New methods for algorithm evaluation and cluster initialisation with applications to healthcare

Henry David Wilde
School of Mathematics

Submitted in partial fulfillment of
the requirements for the degree of
Doctor of Philosophy

2021

Abstract

This thesis explores three themes related to modern operational research: evaluating the objective performance of an algorithm, combining clustering with concepts of mathematical fairness, and developing insightful healthcare models despite a lack of fine-grained data.

The established evaluation procedure for algorithms — and particularly machine learning algorithms — lacks robustness, potentially inflating the success of the methods being assessed. To tackle this, the evolutionary dataset optimisation method is introduced as a supplementary evaluation tool. By traversing the space in which datasets exist, this method provides the means of attaining a richer understanding of the algorithm under study.

This method is used to investigate a novel initialisation method for a centroid-based clustering algorithm, k-modes. The initialisation makes use of the game theoretic concept of a matching game to allocate the starting centroids in a mathematically fair way. The subsequent investigation reveals the conditions under which the new initialisation improves upon two other initialisation methods.

An extension to the k-modes algorithm is utilised to segment an administrative dataset provided by the co-sponsors of this project, Cwm Taf Morgannwg University Health Board. The dataset corresponds to the patient population presenting a specific chronic disease, and comprises a high-level summary of their stays in hospital over a number of years. Despite the relative coarseness of this dataset, the segmentation provides a useful profiling of its instances. These profiles are used to inform a multi-class queuing model representing a hypothetical ward for the affected patients. Following a novel validation process for the queuing model, actionable insights into the needs of the population are found.

In addition to these research pursuits, several open-source software packages have been developed to accompany this thesis. These pieces of software were developed using best practices to ensure the reliability, reproducibility, and sustainability of the research in this thesis.

Acknowledgements

This thesis is the culmination of the first few years of my academic journey. However, I have been afforded the want and resilience to get to this point by a great many people. Here, I would like to acknowledge a non-exhaustive list of those people.

First and most importantly, I offer my utmost gratitude to my supervisors, Dr Jonathan Gillard and Dr Vincent Knight. Your support, encouragement, and guidance throughout this project have been invaluable. You have made a great team for me, and your distinct brand of tutelage has instilled in me equal measures of curiosity, determination, and scepticism. I would also like to thank Mr Kendal Smith and many others at Cwm Taf Morgannwg University Health Board for allowing me to work on a project that has inspired great motivation and intrigue for me.

I offer my thanks to my family, who have been a source of inspiration and support during my studies. Thank you to my parents, Susan and John, who have always pushed me to excel. Thank you to my siblings for facilitating a healthy amount of competition growing up. Finally, I offer a special thanks to my older brother, Matthew, for looking out for me when I have needed it most.

To my dearest friends, Noah Atkin, Jessica Lockwood, Cameron Pearson, and Edward Priest, I offer my sincere thanks and admiration. Your unwavering kindness, warmth, and solace have been vital to keeping my head above water at the worst of times, and have filled me with a great sense of joy otherwise.

Finally, I turn my attention to my academic home for the last six or so years, Cardiff University. I wish to thank my peers in the School of Mathematics, especially Emma Aspland, Lorenzo De Biase, Nikoleta Glynatsi, Emily O'Riordan, Geraint Palmer, Chris Seaman, Álvaro Torras Casas, and Emily Williams, for your stimulating conversation, resolute solidarity and welcomed distraction. Lastly, thank you to the School of Mathematics staff for maintaining a nourishing environment in which to study and work.

Dissemination

Publications

- [376] H. Wilde, V. Knight, and J. Gillard. **Evolutionary dataset optimisation: learning algorithm quality through evolution**. *Applied Intelligence*, 50(4):1172–1191, 2020. doi:10.1007/s10489-019-01592-4.

- [377] H. Wilde, V. Knight, and J. Gillard. **Matching: A Python library for solving matching games**. *Journal of Open Source Software*, 5(48):2169, 2020. doi:10.21105/joss.02169.

Under review

- [378] H. Wilde, V. Knight, and J. Gillard. **A novel initialisation based on hospital-resident assignment for the k-modes algorithm**. Preprint available at arXiv:2002.02701 — *submitted to Knowledge and Information Systems*.

- [379] H. Wilde, V. Knight, J. Gillard, and K. Smith. **Segmentation analysis and the recovery of queuing parameters via the Wasserstein distance: a study of administrative data for patients with chronic obstructive pulmonary disease**. Preprint available at arXiv:2008.04295 — *under review at the European Journal of Operational Research*.

In preparation

- H. Wilde. **A review of current literature covering the intersections of clustering, algorithm evaluation and healthcare modelling**. Available on GitHub at github:daffidwilde/literature-review.

- H. Wilde, V. Knight, and L. Roach. **Automatic final-year project allocation in a School of Biosciences**.

Talks and posters

Unless otherwise stated, copies of the following items (and their source code) are available on GitHub at `github:daffidwilde/talks`.

- **Benchmark datasets are not the only option when understanding algorithm performance (poster)**. *SIAM UKIE Annual Meeting*, University of Edinburgh. 2020. Available at `github:daffidwilde/siam-ukie`.
- **Learning algorithm quality through evolution**. *Welsh Mathematics Colloquium*, Gregynog Hall (Powys). 2019.
- **Parallelisation in Python**. *Advanced Python Workshop*, Cardiff University. 2019.
- **Evolutionary dataset optimisation: data synthesis and algorithm quality**. *Data Science Campus Seminar Series*, Office for National Statistics. 2019.
- **Understanding variation with clustering**. *NHS Wales Modelling Collaborative*, Cardiff. 2019.
- **What is Dask?** *PyDiff*, Cardiff. 2017.

Software contributions

- `edo`: a Python library for generating artificial datasets through evolution. The main application of the library is to better understand an algorithm's quality by studying the generated datasets. Available at `github:daffidwilde/edo`. *Contributions:* main developer.
- `edolab`: an open-source framework for conducting reproducible experiments with `edo`. Available at `github:daffidwilde/edolab`. *Contributions:* main developer.
- `matching`: a Python library for facilitating and solving various matching games. Available at `github:daffidwilde/matching`. *Contributions:* main developer.
- `kmodes`: a Python implementation of the k-modes and k-prototypes algorithms. Available at `github:nicodv/kmodes`. *Contributions:* implemented epoch cost recording and matching-based initialisation.
- `ciw`: a simulation library for open queuing networks. Available at `github:CiwPython/Ciw`. *Contributions:* assisted in refactoring probability distributions and added to documentation.

Contents

List of Figures

List of Tables

List of Algorithms

List of Code Snippets

Chapter 1

Introduction

Operational research (OR) is the scientific process of deriving insights from data to better inform decision-making processes. Since its origins during the Second World War, OR has been applied in all manner of organisations, including those relating to logistics, engineering, and government [160]. Techniques from OR are often designed to optimise an objective function, which may relate to quantities such as costs or efficiency, but the overarching purpose of OR is to make sense of a system that is too complex to understand without a thorough, scientific study of its inner workings.

The technological expansion observed throughout the second half of the 20^{th} century brought about an almost universal increase in industrial and organisational complexity. Among the affected industries was healthcare. With ever-growing issues like increased population size and density, longer life expectancy, and growing socioeconomic disparity, healthcare has become one of the most morally essential applications of OR.

Alongside this rise in complexity came the advent of accessible computational power, and with that, the field of machine learning. Machine learning can be defined in broad terms as a machine (computer) learning patterns and characteristics from data (through the use of statistics) without explicit instructions. This definition aligns machine learning squarely with OR, in that both take data and extract value from it. As such, methodologies employed in contemporary OR projects are increasingly making use of machine learning techniques.

The National Health Service (NHS) is one of the many organisations to adopt machine learning into its operational pursuits, with half of all NHS Trusts engaging in machine learning projects [169]. Despite its promise, there are some commonly occurring problems with applying machine learning to healthcare. These issues include

ensuring ethical integrity, appropriately modelling intricate systems, and having access to sufficient data sources. These challenges are addressed in this thesis, and the solutions depend on an intuitive use of machine learning.

The remainder of this chapter is as follows: Section 1.1 states the primary objective of the thesis; Section 1.2 sets out the structure of this thesis and its chapters; Section 1.3 outlines the novel contributions of the thesis; Section 1.4 provides an overview of the best practices used in developing software for the research presented in this thesis, as well as signposting the software projects themselves.

1.1 Thesis objective

The primary purpose of this thesis is to utilise machine learning to better understand a healthcare population. This purpose is achieved through the novel methods for clustering initialisation and algorithm evaluation presented herein. The co-sponsors of this project, NHS Wales Cwm Taf Morgannwg University Health Board (UHB), are seeking to reveal new insights into their patient population through the use of machine learning. In the context of machine learning, the term 'utilisation' typically refers to the application of some existing machine learning apparatus to a dataset. This thesis considers a more nuanced definition. Here, 'utilisation' is considered as the culmination of three parts: *creation*, *evaluation*, and *application*. Each of these components is essential to properly utilising machine learning, and the process is comparable to honing a craft.

First, a tool is fashioned for a particular purpose, as a machine learning method would be for a problem — or class thereof. Once a prototype has been created, the tool must be evaluated and adjusted. This process improves both the skilled use of the tool and the tool itself. Evaluating a machine learning method extensively is analogous to this process. With a refined tool and skilled hand, expertise in that craft can be demonstrated by applying it for its intended purpose; this is the final (and often only) stage when utilising machine learning in the real world.

Cwm Taf Morgannwg UHB are particularly concerned with understanding the operational characteristics of their patients presenting chronic obstructive pulmonary disease (COPD). COPD is a respiratory condition with known links to deprivation, and often presents as a comorbidity, i.e. in concurrence with at least one other condition. These affiliations make living with and treating COPD inherently difficult, emphasising the importance of studying it closely.

This thesis considers how clustering can be utilised to better understand the COPD population treated by Cwm Taf Morgannwg UHB. In order to carry this out, this

thesis incorporates three themes from modern OR: algorithm evaluation, clustering and segmentation, and operational healthcare modelling. These seemingly disparate themes are presented as a triptych, where each theme corresponds to its own chapter. With each themed chapter, there is a clear contribution to a component of utilising machine learning.

In Chapter 4, an extension to an existing clustering algorithm is created. Following its definition, the method is thoroughly evaluated using a novel framework for understanding the objective quality of an algorithm. This framework is presented in Chapter 3. Through this evaluation, the clustering method is identified as an appropriate tool for segmenting the COPD population in Chapter 5. In turn, the work in each chapter contributes to an effective operational methodology, which addresses the concerns of Cwm Taf Morgannwg UHB and their COPD population, providing useful and tangible insights into the population.

1.2 Thesis structure

Including this introduction, this thesis contains six chapters, which together cover the research topics of this thesis. A brief summary of each chapter is given below:

- Chapter 2 comprises a literature review covering the principal topics of this thesis: clustering, healthcare modelling, and model evaluation. In addition to surveying each topic individually, their intersections are considered.
- Chapter 3 presents a novel approach to understanding an algorithm's quality according to a particular metric. The presented method allows for an exploration of the space in which 'good' datasets exist by use of an evolutionary algorithm.
- Chapter 4 describes a new initialisation method for an existing clustering algorithm. This method models the initialisation as a matching game, incorporating a mathematical notion of fairness. The chapter concludes with an evaluation of the method against two initialisations, making use of the approach set out in Chapter 3, and reveals the cases in which the new initialisation improves upon two existing methods.
- Chapter 5 combines the initialisation from Chapter 4 with the findings of the analysis in Appendix B to produce a segmentation of a healthcare population, using another administrative dataset from Cwm Taf Morgannwg UHB. This segmentation is used to inform a multi-class queuing model, and subsequent

adjustments to that model provide actionable insights into the needs of the population under study.

- Chapter 6 summarises the research presented in the previous chapters and establishes avenues for further work.

In addition to these chapters, this thesis contains several appendices. Of these, two appendices (Appendix A and Appendix B) provide additional context to two of the later chapters — Chapter 4 and Chapter 5, respectively. These appendices are not presented as chapters because they do contain a significant amount of novel mathematics.

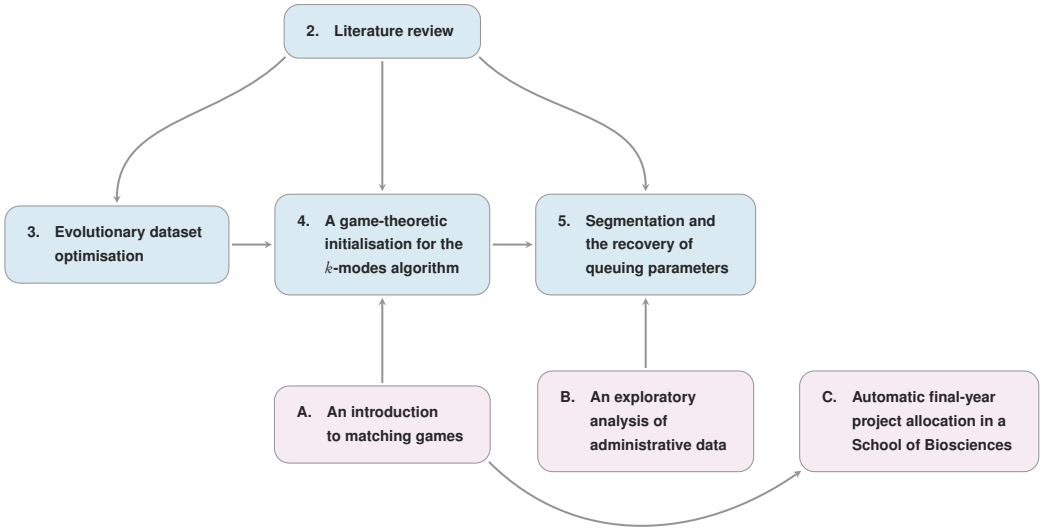


Figure 1.1: A graph of the chapters, appendices, and their connections

The logical connections between the chapters and appendices of this thesis are demonstrated in Figure 1.1. An arrow from one chapter (or appendix) to another indicates that some part of the research presented in that chapter contributes to the research in the other.

1.3 Novel contributions of the thesis

This section lists aspects of this thesis which are novel to its principal themes of algorithm evaluation, clustering, and operational healthcare modelling. The contributions of each chapter are presented separately with a concise description of the problem, existing literature surrounding that problem, and how that problem is addressed through this thesis.

Chapter 3 addresses the issue of how algorithms are evaluated. The standard procedure for algorithm evaluation consists of measuring the performance of an algorithm

on a small number of examples and metrics. This procedure is referred to as a confirmation process. Such processes offer little evidence upon which conclusions can be based about the quality of an algorithm [282]. The method presented in Chapter 3, called evolutionary dataset optimisation (EDO), expands on the familiar concept of a confirmation process. The EDO method generates datasets for which an algorithm performs well by optimising some fitness function. While some recent works into synthetic data generation champion the promise of deep learning [25, 280, 356], the EDO method is a bespoke evolutionary algorithm and promotes transparency in the data generation process. Two case studies that apply this methodology are given in this thesis: one in Chapter 3 and the other in Chapter 4.

The k-modes initialisation presented in Chapter 4 extends an existing method from the seminal work on the k-modes algorithm [168]. While this initialisation is novel itself, the chapter contributes to the growing body of research around fair machine learning practices [30, 81], and particularly those related to clustering such as [10, 71]. These practices aim to reframe machine learning to focus on collective benefit, and are often based on (or share some common root with) game theory. Game theory is a branch of mathematics which applies rules and logic to resolve and analyse scenarios involving conflict, cooperation and competition among rational agents. The novel method in Chapter 4 incorporates objects from game theory directly, offering another approach to 'fair' machine learning.

The research reported in Chapter 5 makes two major contributions to OR literature. Firstly, the methodology comprises a novel combination of machine learning and classical OR techniques to provide insight into a healthcare population. The use of clustering to inform a healthcare queuing model does not appear in literature, despite its use to study the results of queuing models — as in [291, 300]. Secondly, the methodology circumvents the common issue of applying OR to areas such as healthcare where sufficiently detailed data is not always available. The dataset used in the chapter is a routinely gathered, administrative dataset, from which a well-fitting replica is derived via the Wasserstein distance.

1.4 Software development and best practices

Conducting research without software is seemingly becoming a thing of the past. In 2014, the Software Sustainability Institute surveyed researchers (from across the disciplinary spectrum) at 15 Russell Group universities. Their analysis revealed that 92% of respondents use software to conduct their research, and 69% responded that "their research would not be practical without" software [158]. The research conducted in this thesis is no different, and relies on the use of software. As with all

scientific pursuits, researchers who make use of software are obliged to ensure their work is correct and reproducible. This section provides a brief overview of the software developed for this thesis, and the methods of best practice used to develop that software in a responsible manner.

1.4.1 Code snippets

Throughout this thesis, snippets of code are shown. These snippets are either of source code, as in Snippet 1.1, or uses of code. The first type of code snippet is presented on a darker background and is used to display some part of the source code of an existing piece of software. In general, the source code in these snippets is written in the open-source language, Python [353], as that is the default language for the software developed for this thesis. The second type of snippet can be distinguished by its lighter background and is used to display a series of commands to run; where these commands should be run is indicated by the preceding symbols.

```
1  def main():
2      """ Say hello. """
3
4      return "Hello world."
5
6  if __name__ == "__main__":
7      main()
```

Snippet 1.1: An example of some Python source code

A snippet whose commands begin with >>>, as in Snippet 1.2, should be run in a Python interpreter while those with commands beginning with >, as in Snippet 1.3, should be run in a shell. In each of these cases, the output of a command (or series of commands) is displayed directly beneath it without any preceding symbols.

```
>>> print("Hello world.")
Hello world.
```

Snippet 1.2: An example of some code run in a Python interpreter

```
> echo "Hello world."
Hello world.
```

Snippet 1.3: An example of some code run in a shell

1.4.2 Methods of best practice

Best practices are guidelines to ensure that research methods are reliable, reproducible, and transferable. In essence, the proper adoption of best practices sustains the lifespan of a piece of research. The same is true of research software. In Chapter 2, the ethical implications of best practices are discussed, as well as briefly mentioning the analogous practices for research data. Examples of existing software best practices include [1, 40, 185, 382]. The following subsections provide overviews of four fundamental methods of best practice that are used throughout the software developed for this thesis: version control, virtual environments, automated testing and documentation.

Version control

A *version control system* records all files within a software project, typically on a line-by-line basis. As the name suggests, the system also keeps a record of all the versions of that project. This record of a project is called a *repository* and offers some transparency into how the software was developed. Full accounts of the history and benefits of version control systems and their features may be found in [313, 399].

A number of version control systems exist, each with their own objectives and specialities, but all of the software for this thesis was developed using Git [352]. Created by Linus Torvalds in 2005, Git is a free, open-source version control system that has been widely adopted by large tech companies including Google, Facebook, and Microsoft. The primary objectives of Git are to be uncomplicated and to provide frictionless, low-latency versioning.

Several services exist for hosting Git repositories online, the most popular of which is GitHub [135]. Each of the repositories used in this thesis is publicly hosted on GitHub, and links to them are listed in Table 1.1. In addition to the benefits of the underlying version control system, hosting services afford software developers the ability to make their software accessible beyond their local machine. Furthermore, GitHub has features which encourage collaboration between developers, allowing users to interact through their repositories by reporting issues, commenting and liking, and (perhaps most importantly) requesting to make changes.

Virtual environments

When using or developing a piece of software, it is almost a certainty that it will have *dependencies*. A dependency is a version of some existing software required by the newly developed software to run. Occasionally, there will be clashes in the

dependencies of two or more pieces of software, or another developer may wish to install that software exactly as it was created. These are two examples of motivations for organising and separating project dependencies; *virtual environments* provide a means of achieving this. A virtual environment is a self-contained, independent copy of some dependencies that can be activated and deactivated at will. By activating an environment, only the specific versions of the dependencies are available.

```yaml
name: thesis
channels:
- defaults
- conda-forge
dependencies:
 - python>=3.6
 - dask=2.30.0
 - ipykernel=5.3.2
 - matplotlib=3.2.2
 - numpy=1.18.5
 - pandas=1.0.5
 - scikit-learn=0.23.1
 - scipy=1.5.0
 - statsmodels=0.11.1
 - tqdm=4.48
 - pip=20.1.1
 - pip:
   - alphashape==1.0.1
   - bibtexparser==1.2.0
   - descartes==1.1.0
   - edo>=0.3
   - git+https://github.com/daffidwilde/kmodes@v0.9.1
   - graphviz==0.14.1
   - invoke==1.4.1
   - matching==1.3.2
   - pygments>=2.5.2
   - shapely==1.6.4.post2
   - yellowbrick==1.1
```

Snippet 1.4: The Anaconda environment file for this thesis

Each of the repositories in this thesis includes an Anaconda virtual environment configuration file named `environment.yml`. Anaconda [15] is a free and open-source distribution of various pieces of software, including the Python, R and Julia programming languages. This distribution has been specialised for scientific computing, hence its use in this thesis. Included with Anaconda are tools to simplify package management such as the virtual environments created using environment configuration files.

Snippet 1.4 shows the contents of an overarching environment file for this thesis. The environment file lists the name of the environment (`thesis`), its dependencies, and the locations from which those dependencies should be installed (under `channels` and `pip`). Beside each dependency is the specific version (or bounds on the version) required to recreate the environment.

Automated testing

Testing code is essential to ensuring that a piece of software works as intended, and that it is robust and sustainable. *Automated testing* is the de facto tool used by software developers to test their code, consisting of *test suites* that run parts of the code base to ensure they behave as expected. The importance of testing cannot be understated in producing good software, and is the basis of the software development practice known as test-driven development (TDD). A thorough tutorial on how to adopt TDD may be found in [285]. This book informed much of the process by which the software was developed for this thesis.

Included in each of the software package repositories are test suites composed of two types of test: *functional* and *unit* tests. A functional test asserts that the software (or a part thereof) behaves as expected from the perspective of a user, while a unit test checks the behaviour of a small (potentially isolated) part of the code base from an internal viewpoint. Unit tests allow a developer to ensure that their software is free from any bugs, and streamline the process of finding the source of any bugs.

All of the test suites associated with this thesis were written using the Python library, `pytest`. The `pytest` framework is designed to write scalable test suites, and comes with a number of plugins, including one to automatically test for *coverage*, `pytest-cov`. Coverage is a measure of what proportion of the code base for a project is 'hit' (executed) when running the test suite, indicating the robustness of the suite. All of the test suites associated with this thesis achieve 100% coverage.

To regularly test code that is going to be merged into the main code base (through version control), continuous integration (CI) systems exist. CI systems run the test suite and coverage checks at regular prompts (e.g. when a new version is pushed to the online repository, prior to new releases of the software, according to a schedule, etc.), minimising any potential issues during development and collaboration as well as providing another layer of transparency. Given that the code for this thesis is hosted on GitHub, the CI used is GitHub Actions [136].

Documentation

In addition to testing, another crucial appendix to a software code base is its *documentation*. Software documentation can take many forms — text, websites, illustrations, demonstrations — but regardless of how it is presented, the purpose is to explain to a user how to use a piece of software.

All of the repositories associated with this thesis include (at a minimum) a `README` file, detailing what the repository is for, and (if appropriate) instructions on how

to reproduce the results with the code therein. Each Python function, method and class defined in the source code includes its own inline documentation in the form of a *docstring*. Furthermore, the variables and defined objects have been assigned informative, sensible names, making the software self-documenting.

For the larger, free-standing software packages developed during this thesis, fully fledged documentation websites have been written. Each of these is hosted on Read the Docs and adheres to the so-called 'Grand Unified Theory of Documentation' [290], which separates documentation into four categories: tutorials, how-to guides, explanation and reference.

1.4.3 Summary of software

As stated throughout this section, the software to accompany this thesis has been written according to best practices, and their associated repositories are available online. These practices have been adopted to ensure the reliability, reproducibility and sustainability of the software described throughout this thesis.

In addition to these GitHub repositories, the specific versions of the source code used in each chapter have been archived online via Zenodo [118]. Each archive is assigned a digital object identifier (DOI) name, further reinforcing the longevity of the software. Table 1.1 details the repositories and archives associated with each chapter.

Chapter	GitHub repository	Source code archive	Data archive(s)
2	github:daffidwilde/literature-review	doi:10.5281/zenodo.4320050	doi:10.5281/zenodo.4320050
3	github:daffidwilde/edo-paper	doi:10.5281/zenodo.4000316	doi:10.5281/zenodo.4000327
4	github:daffidwilde/kmodes-paper	doi:10.5281/zenodo.3639282	doi:10.5281/zenodo.3639282
5	github:daffidwilde/copd-paper	doi:10.5281/zenodo.4457902	doi:10.5281/zenodo.3908167 doi:10.5281/zenodo.4457808

Table 1.1: The repositories and archives associated with each chapter

This thesis and its supporting files are also hosted online at `github:daffidwilde/thesis`. It has been prepared using LaTeX and it is regularly tested using the GitHub Actions CI. The tests include checking that the document can be compiled, that it is without spelling errors, and that the Python usage code snippets are correct.

Chapter 2

Literature review

The survey reported in this chapter will form part of a future work entitled:

"A review of current literature covering the intersections of clustering, algorithm evaluation and healthcare modelling"

Available online at: `github:daffidwilde/literature-review`
Associated data and source code: `doi:10.5281/zenodo.4320050`

The research reported here will form the opening sections of the manuscript listed above. In addition to the classic literature review, the manuscript will include a software-driven, bibliometric study of the wider literature corpus. Due to the time constraints of this project, this part of the study is not included here. This chapter also includes a more extensive review of clustering techniques, as this form of analysis proves instrumental in the research presented in Chapters 4 and 5.

2.1 Introduction

Modern research, in all respects, is highly dependent on the use of data. The authors of [384] consider a stratified sample of several thousand articles from leading journals across the natural sciences, and suggests that perhaps as many as 80% of all scientific articles published in 2014 utilised research data directly. That article is one of many in an increasing body of publications over the last decade that concern themselves with the state of research data [159], its utilisation and openness [23, 400], and the importance of best practices [79, 82]. These publications focus on using this set of characteristics as a measure of the quality of a research data source, which also acts as a mark of the associated research's quality; this is in contrast to other

recent trends where the pursuit of more voluminous and varied data sources has taken precedence [32]. In [336], the authors argue for the widespread use of a concept (used already in the geosciences) that unifies these characteristics: that research data should be *Findable, Accessible, Interoperable and Reusable* (FAIR) — an acronym coined in [380].

In tandem with this shift was the rise in the use of software to implement and compute algorithms, and so electronic data became essential. Similarly to the FAIR guidelines for research data, best practices for research software development have been adopted [1, 40, 185]. As the size and complexity of research data increased, so did the span of the field *machine learning*. To reiterate the opening section of this thesis, machine learning is the process of applying statistics to glean insights from a potentially large source of data, without following explicit instructions. Owing in part to this broad definition, machine learning comprises a great many techniques that were borne out of statistics, including regression, classification and clustering.

Healthcare modelling is one crucial (albeit broad) branch of research in which decisions are increasingly being informed by data-driven methodologies [37, 192, 302], often based around machine learning techniques and big data [11, 17]. One such technique is *clustering* — the task of identifying a partition of a dataset with homogeneous parts. Its efficacy in identifying otherwise unknown structures in a dataset, as well as straightforwardly garnering the attention of non-technical stakeholders has proven it to be an essential part of healthcare modelling research [355, 392], as is discussed in Section 2.3.

Regardless of the particular methods that are to be implemented, the quality of a methodology must be evaluated as being fit-for-purpose before it can be used in a real world setting. The ways in which this evaluation is carried out are reliant on both the methods themselves and the setting in which they are applied. However, there are some approaches used across research fields such as consensus by literature or corroboration via some metric on some dataset(s). In addition to objective measurements, there are broader considerations to be made about a model, including its adequacy-for-purpose [282], reusability [304], and — particularly in healthcare — whether it is ethical [143].

This survey considers the literature surrounding these three components of modern operational and mathematical research — clustering, the evaluation of models, and healthcare modelling — as well as their intersections. Given the potentially vast nature of the literature under study, summarising a slice that encompasses the state of the research requires organisation. As such, the structure of this chapter is as follows:

- Section 2.2 summarises literature on clustering paradigms and algorithms;
- Section 2.3 addresses some select aspects of operational healthcare literature;
- Section 2.4 reviews current literature on the evaluation of models, clustering algorithms and healthcare models.

2.2 Clustering

Clustering, or *cluster analysis*, is a generic term used to describe a number of techniques whose objective is to partition some dataset into parts (clusters) according to some distance (similarity) measure. The generated partition should be such that the members of one cluster are more similar to one another than the rest of the data [119]. This form of analysis has found applications in an array of fields, including the optimisation of energy systems [186, 350], wireless sensor networking [142], and the biological sciences [60, 200].

Clustering originated in the social sciences around the turn of the 20$^{\text{th}}$ century in a similar manner to several other (now ubiquitous) statistical tools, including hypothesis testing and p-values, correlation, and factor analysis. Early work on cluster analysis, such as [68, 102], focused on its applications in attempting to identify traits which led to cultural and psychological differences between groups of people, as opposed to studying the methods used to identify any partitions.

The focus of clustering research shifted to the statistical nature of the methods following work on clustering under the name *numerical taxonomy* [333, 334]. These works were highly influential and led to an abundance of research into clustering methods, including [99] and [155] that each formalised the principles of clustering as a part of statistics. Furthermore, this period witnessed the advent of seminal work on clustering algorithms such as k-means [156] and hierarchical clustering [93, 327] which are still considered fundamental methods [8, 385]. Since then, with an increase in the data under study, clustering has fallen more squarely into the category of machine learning techniques that perform *unsupervised learning* — this is because clustering relies solely on the entirety of the observed data and some a priori knowledge such as a set of parameters [91].

The remainder of this section offers a summary of the three principal types of clustering algorithms: those belonging to the k-means (centroid-based) paradigm, hierarchical models, and, finally, density-based algorithms. In addition to these, fuzzy clustering and model-based clustering are popular. For overviews and reviews of the methods belonging to the former, consider [122, 141, 218], and [53, 128, 242] for the latter.

2.2.1 Centroid-based clustering

The concept of k-means clustering is to partition a real-valued dataset into $k \in \mathbb{N}$ homogeneous parts (clusters), where each data point is assigned to the cluster to which the point is closest. The sum of these within-cluster distances is often referred to as the *inertia* of the clustering. The distance from a point to a cluster is defined as the distance from the point to the mean of the elements in that cluster, also referred to as the cluster centre or centroid; this alternative name gives another name to the paradigm, *centroid-based clustering*. Typically, the distance measure used is the Euclidean distance, with the assumption that several preprocessing techniques are used to mitigate any scaling discrepancies in the attributes of the data.

Although k-means clustering has independently been discovered a number of times since as early as the 1950s (comprehensive histories on the subject can be found in [49, 180]), its formulation is commonly attributed to [156]. Given its lengthy standing, there are a multitude of algorithms that perform k-means clustering. However, the most commonly used is Lloyd's algorithm [222]. This procedure is remarkably uncomplicated in its statement, and has now become so synonymous with k-means clustering that it is referred to as 'the k-means algorithm'. To summarise it in a sentence, the algorithm iteratively partitions the numerical data into k parts, reassigning data points and adjusting cluster means until no point moves cluster on a full cycle of the dataset. The algorithm has proven popular owing to this simplicity. Furthermore, it is scalable, can be parallelised [28], and its implementations are concise [267, 385].

However, centroid-based clustering is not without its drawbacks. The most notable of these is its dependency on being presented with data that can be naturally partitioned into k clusters of a particular type [275]. Namely, the true clusters should be isotropic, convex and linearly separable. The reason these conditions exist, particularly for k-means, is because the k-means (Lloyd's) algorithm is a heuristic for identifying the centroidal Voronoi tessellation of a dataset [103]. Hence, the true objective of the algorithm is to divide the attribute space into k equally weighted parts using the dataset as a sample, rather than focusing on dividing the dataset itself.

Regardless of the true nature of a centroid-based clustering algorithm, there is the issue of choosing a suitable value of k prior to clustering. A common approach to this problem is the so-called 'elbow' method where a clustering algorithm is run using a range of values for k [8]. This range is typically informed by what is considered reasonable in the problem domain. From these runs, the value of k is plotted against some objective and the plot is inspected for a kink or 'elbow'. Typically, inertia is used, but another popular metric is the silhouette coefficient: a measure for both the

intra-cluster tightness and inter-cluster separation of a clustering [310].

Irrespective of the objective used, the definition of what constitutes an elbow is not necessarily well-defined. The works [343, 346] both employ the elbow method with final inertia as the objective. In each case, the authors offer little explanation of their methodologies other than choosing k where 'drastic' changes occur in the objective. Attempts to formalise this method have been made, however. For instance, the authors of [319] provide a 'knee-point' detection algorithm that finds the point of maximal curvature in the domain of a function when provided with some basic properties of that function (whether it is convex, monotonic, etc.). This algorithm has been adapted to cleanly carry out the elbow method in software packages, such as Yellowbrick [39], by interpolating the objective values.

In addition to the elbow method, extensions to k-means itself have also been presented to alleviate its dependency on a choice for k. For example, in [272], the authors propose an automatic decision scheme for k-means based on a vector comprised of the values that constitute the inertia of a clustering. The process determines that if this vector meets some criterion attached to its standard deviation, then each cluster is sufficiently unimodal and Gaussian, fulfilling the condition that clusters be roughly spherical.

Another limitation of k-means is that it is dependent on its initial solution. The standard initialisation process is to select k points in the dataset at random, introducing a stochastic element to the algorithm. Studying methods to find sensible (ideally, deterministic) initial solutions has formed a substantial part of centroid-based clustering research. For instance, the authors of [232] provide a novel initialisation that pursues the Voronoi tessellation by use of a 'divide-and-conquer' approach. Under the presented scheme, the initial centroids are selected based on a rough partition of the attribute space according to their extreme values. Likewise, the initialisation presented in [331] assigns each centroid using the arithmetic mean and dividing the remaining attribute space in two. Meanwhile, the method introduced in [191] initialises k-means by sorting the instances in the dataset according to their distance from the origin. Once sorted, the instances are split into k disjoint sets and the mean of each set is assigned as a centroid. A likely issue with this approach is that there are new assumptions about the data, i.e. that it is centred about the origin in all dimensions (rectifiable through feature scaling), and that all points equidistant from the origin are similar to one another.

A crucial part of centroid-based clustering research is dedicated to alternative measurements for the centre of a cluster. A cluster centroid is broadly regarded as a purely geometric object, and is measured using a distance metric and a measure of

central tendency. Using the Euclidean mean as the centre of a cluster is a sensible choice when clustering continuous, normalised data. However, if the data is not scalable (if it is categorical, for instance) then this approach is not suitable. Further, the Euclidean distance is linear, leading k-means to be prone to the effects of outliers in the data. As such, numerous extensions to Lloyd's algorithm exist in the literature that are designed to be more robust to these limitations. For continuous (or at least ordered) data, k-medians and k-medoids exist. The former uses the median value of each attribute in a cluster to form the centre [21, 55], mitigating the effect of outliers.

In k-medoids, the centre of a cluster is taken to be the data point which is closest to all other points in the cluster [193]. Typically, k-medoids makes use of the Manhattan distance, but is generic in its schema. If Euclidean distance is used, then k-medoids is identical to k-means, with the restriction that the centre of the cluster is an actual point in that cluster. Specifying that the centre be a real point is seen as the primary benefit of k-medoids, and continues to be actively studied [321, 363]. There are cases where a virtual cluster centre is not sufficient, such as facility allocation [70, 373] and the clustering of gene sequences [187].

When considering a categorical or mixed-type dataset, none of the aforementioned clustering techniques will work as they were intended because the sense of order does not exist across the entire attribute space. While it is possible to use k-means in these situations, by converting any categorical attributes into pseudo-numerical (binary) attributes by use of dummy coding, this is not recommended. One potential failing is the inflated effect of categorical variables with many distinct values in any distance calculation, as these will produce more binary variables. Another is that by introducing potentially many binary variables, the dimensionality of the dataset increases substantially, and so the points in the higher-dimension form of the data become intrinsically further from one another, thus loosening any sense of cluster homogeneity.

Therefore, another approach should be taken that handles the data directly. The k-modes and k-prototypes algorithms (introduced in [168]) are capable of handling categorical and mixed-type data, respectively. The k-modes algorithm is an extension to Lloyd's algorithm, like k-medians and k-medoids, where the cluster centre is defined as a point whose attribute values are most frequent among those of the points in a cluster. [168] provides a so-called 'matching dissimilarity' measure to expedite the centre calculations, making k-modes scalable and computationally efficient [226]. The k-prototypes algorithm is essentially a blended form of k-modes and k-means where each method is applied to the categorical and numeric attributes at each iteration, before their respective costs are combined linearly according to some

parameter, γ. Work into k-modes clustering is largely focused on the initial choice of centroids, as in [64, 184, 197, 198, 347, 378], and improving on the original distance measure. Two popular dissimilarity measures for k-modes clustering were presented in [65] and [259]. Each of these novel measures improve on the basic metric by taking into account the relative frequency of the values of each attribute. The latter measure [259] focuses locally to the cluster at hand, while the former [65] considers their frequencies from a universal perspective.

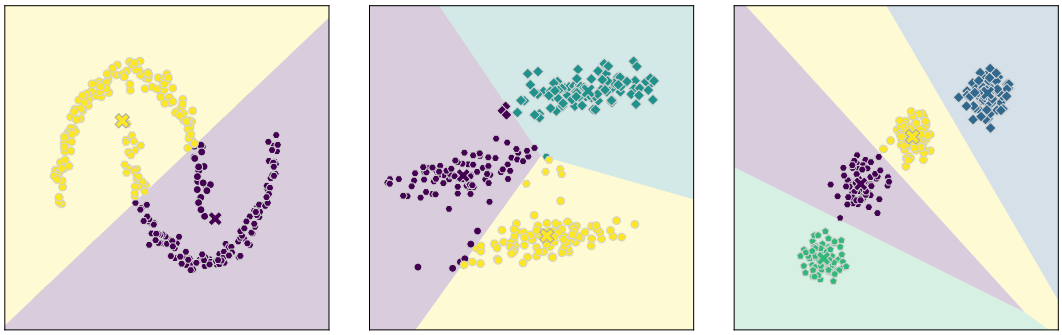
While centroid-based clustering is dominated by Lloyd-like algorithms, recent work has sought out alternative approaches. For instance, the authors of [71] revise centroid clustering from an expectation-maximisation process to one that considers its data points as agents that act with preferences based on their locale. The overall aim of this approach is to incorporate a sense of mathematical fairness by identifying a solution that can not be justifiably rejected by any subset of its agents. Such a solution is found through a proportional (i.e. Pareto-optimal) distribution of the cluster centres according to some distance metric.

At this point, the survey returns to the effect of outliers on Lloyd-like algorithms, especially k-means, and the literature around reducing that effect. The k-means-sharp algorithm was presented in [273]; this revision to k-means embeds an outlier detection feature which relies solely on the underlying structures from the original k-means algorithm. The benefit of handling adverse noise in this way is twofold: first, it preserves the computational efficiencies that come with using the mean (as opposed to sorting the attributes of each cluster to find a median in k-medians, say); and, second, there are no additional parameters to determine. In [146], the authors provide another integration with k-means for detecting outliers by way of a local search, but in order to do so, they introduce another parameter determining the cardinality of the neighbourhood of a point. In contract to these extensions, are examples of works that exploit this weakness of k-means clustering [216, 375]. In each case, the k-means algorithm partitions a dataset, and outliers are identified according to some neighbourhood threshold associated with the elements of a cluster.

The final remark in this section of the survey is concerned with the relaxation of the constraints on cluster geometry by centroid-based clustering algorithms. This property is fundamental to how these algorithms operate and are designed, and so little research has been done on the subject. Having said that, the work in [345] demonstrates how the Mahalanobis distance can be used in place of the Euclidean distance to circumvent the dependency on spherical (i.e. isotropic) clusters by k-means. However, the resultant clusters must still be convex and linearly separable. The Mahalanobis distance is commonly used to detect outliers in other settings, and is dependent on the eigenvalues of the covariance matrix derived from a dataset [227].

Name	Summary	No. clusters	Convex	Separable	Isotropic
Moons	Interlocking crescent moons	2	✗	✗	✗
Ellipses	Elongated globules	3	✓	✓	✗
Spheres	Rough-edged balls	4	✓	✓	✓

Table 2.1: A summary of the synthetic datasets



(a) (b) (c)

Figure 2.1: Clusters identified by k-means on the (a) moons, (b) ellipses, and (c) spheres datasets

A potential downside to this measure is the singularity of that matrix, which is particularly likely in high-dimensional data. Another exemplar work is [338], where a variant of k-means is presented as being agnostic to its distance measure, and removes the requirement for the attribute space to be a metric space at all.

Figure 2.1 demonstrates the nuances of centroid-based clustering discussed in this section, and shows how the k-means algorithm performs on three synthetic datasets. A brief summary of these datasets is provided in Table 2.1. The datasets were created using the Python library Scikit-learn [283], and the source code to generate them (and the plots included in this survey) is available at the GitHub repository for this manuscript (`github.com/daffidwilde/literature-review`), and has also been archived online under `doi:10.5281/zenodo.4320050`.

Each plot in the figure shows a grouped scatter plot of the identified clusters, with the true clusters indicated by the marker shape. In addition to this scatter, the cluster centres are shown (as crosses), as well as the Voronoi cells defined by those centres (as shaded regions). Note that the axes are presented without labels. The name and scale of each attribute detracts from the purpose of this figure, which is to demonstrate the clustering. As such, all unnecessary information has been removed.

Figure 2.1a shows a stark example of where centroid-based clustering fails; despite the clusters in the dataset being distinct, k-means is unable to identify them. Instead,

the attribute space is divided to create two equally weighted regions. Likewise, in Figure 2.1b, the placement of the cell boundaries has been determined by weighting, rather than the cluster locations. Finally, with Figure 2.1c, k-means performs well, correctly partitioning the dataset into its four clusters.

2.2.2 Hierarchical clustering

Hierarchical clustering seeks to identify a hierarchy of the relationships between subsets of points in a dataset. A hierarchical clustering algorithm depends on a distance metric for measuring inter-point similarity, and a *linkage criterion* to determine the distance between two sets of points. For the most part, algorithms for hierarchical clustering are one of two types: agglomerative or divisive [194]. Agglomerative algorithms, originating with [188, 374], begin with each point in its own cluster and work to merge clusters together, constructing the hierarchy from the bottom up. Meanwhile, divisive algorithms, with [108] being an early example, take a top-down approach and seek to split the dataset into reasonably homogeneous clusters.

Since its origins, hierarchical clustering research has been dominated by agglomerative methods; this is due, largely, to the computational issues associated with how to best split a heterogeneous dataset into parts recursively. For a dataset of n points, a complete search of the possible splits takes 2^n calculations, and so divisive methods require efficient heuristics to be even remotely plausible on real-world data. A common choice is the k-means algorithm [251, 286] which was discussed in Section 2.2.1.

In addition, hierarchical clustering has been successfully reframed as a combinatorial optimisation problem, as is done with divisive clustering in [89], which allows for the application of optimisation heuristics such as linear programming [311], local search [12] and probabilistic estimation [120] to identify a clustering in reasonable time. Furthermore, related works exist which concern themselves with determining the quality of a hierarchical structure with respect to some cost function, like [46, 225]. In [78], the authors combine these methodologies to produce a divisive clustering algorithm that achieves logarithmic complexity.

While heuristic methods exist for agglomerative clustering [12, 120], its methods do not bear the same burdens as divisive methods; the grouping together of parts can be achieved with more straightforward tools. Generally, the definition of an agglomerative algorithm is reliant on their linkage criterion. Meanwhile, the distance metric used is largely dependent on the type of data presented [263]. Two of the fundamental criteria for hierarchical clustering are complete linkage, which defines the CLINK algorithm [93], and single linkage, providing the SLINK algorithm [327].

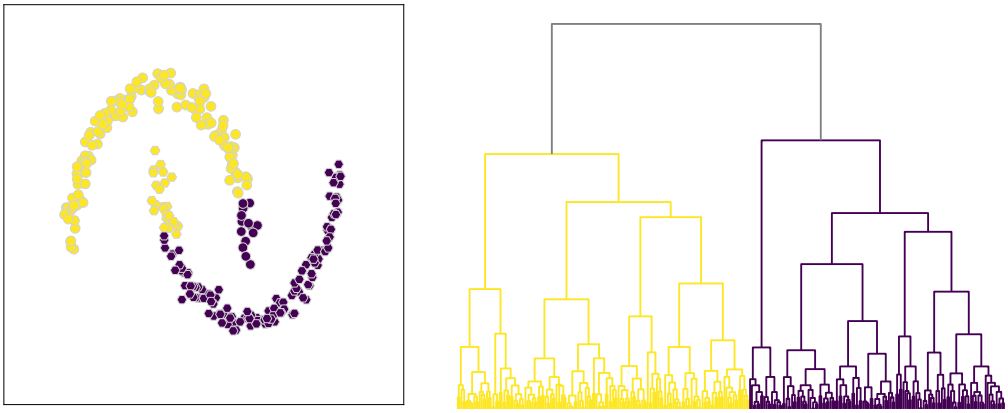
These linkage criteria also have the colloquial names *farthest neighbour* and *nearest neighbour* clustering, respectively.

Another popular linkage criterion is average-linkage, which defines the distance between two clusters as the mean distance between any point in one cluster and any point in the other. Defining linkage criteria with point-wise distances allows a hierarchical (agglomerative or divisive) algorithm to be run using only a point-wise distance matrix, as opposed to a dataset directly. Doing so allows for increases in computational efficiency by use of caching [263]. However, hierarchical methods continue to be prone to outliers and are biased towards globular clusters because of definitions such as these. Ongoing reviews on the subject of hierarchical (although predominantly agglomerative) clustering methods are available [253, 254, 255].

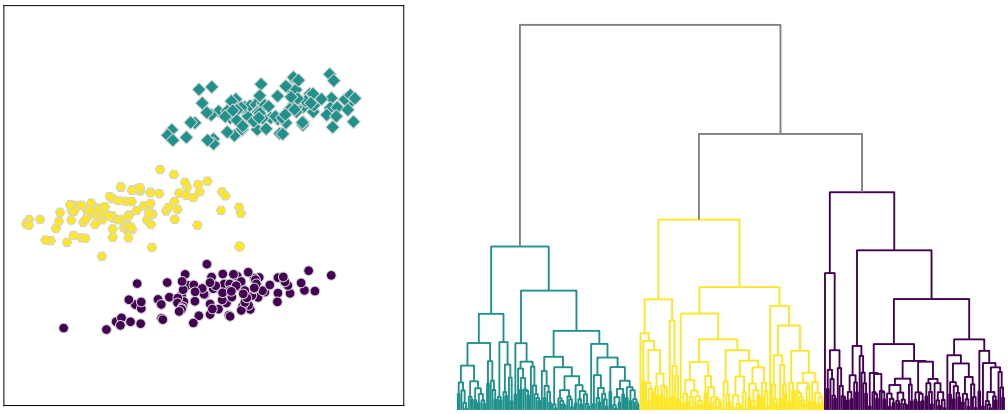
Like centroid-based clustering, a principal issue with hierarchical clustering is identifying the appropriate number of clusters. Hierarchical algorithms terminate when all points exist in a single cluster (for agglomerative methods) or are all separate (for divisive). A key advantage of clustering in this way is the retention of a complete history of how the points relate to one another. These histories are used to visualise the clustering process via a dendogram. A dendogram is an acyclic graph (tree) arranged into levels corresponding to the stages at which two clusters are merged (or split).

With such a visualisation, and the underlying tree itself, an appropriate number of clusters can be identified after running the algorithm once, as opposed to the ranges of runs required for the elbow method in centroid-based clustering. The authors of [351] present a hierarchical clustering schema that incorporates complete-linkage clustering with the minimum variance criterion presented in [374]. In doing so, the method automates the process of choosing an appropriate number of clusters; a decision that is usually made post hoc according to another metric, such as the silhouette coefficient, evaluated at each level of the tree.

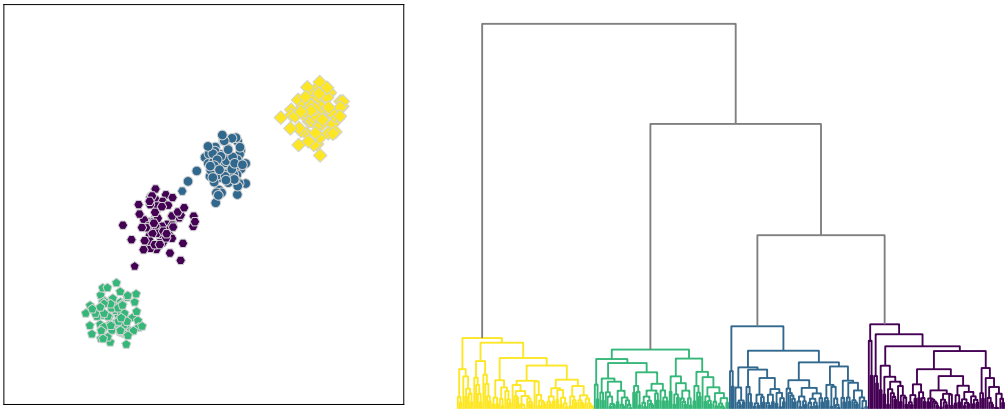
Figure 2.2 shows the output of agglomerative clustering with average linkage on the three synthetic datasets described in Table 2.1. For each example, both the grouped scatter plot (left) and a dendogram (right) are shown. To obtain the scatter plot, the clustering was stopped when the required number of clusters had been identified. The same clustering can be identified by drawing a horizontal line on the dendogram at the level where the number of vertical links equals the number of clusters. The subtrees below that line form the clusters in the scatter plot. In the figure, this level has been indicated by colouring the requisite links with the corresponding cluster colour. Note again, that the figures are deliberately bare to focus on the outcome of the clustering, rather than the artificial attribute values.



(a) The moons dataset



(b) The ellipses dataset



(c) The spheres dataset

Figure 2.2: Clusters and dendograms identified by average-linkage clustering on the synthetic datasets

Like k-means, average-linkage clustering struggled to identify the moons in Figure 2.2a, but other linkage criteria may be robust to shapes such as these. For the remaining examples, hierarchical clustering appears to perform well at identifying convex, linearly separable regions in the datasets. However, there are a handful of incorrectly clustered points along the borders of some regions.

2.2.3 Density-based clustering

Density-based clustering differs from centroid-based methods in that it actively seeks more of the underlying structure of a dataset. In centroid-based clustering, decisions are made using the distance between two points, which provides a *flat* clustering without an underlying structure. Meanwhile, hierarchical methods make use of identification and aggregation devices, which are derived from point-wise distances, to measure and organise the connectivity of subsets in a dataset, providing insight into the structure of the data. Arguably, density-based clustering takes this structural approach a step further and considers the attribute space of a dataset as a density surface. Then, a cluster is defined as a high-density region of that surface.

By considering a dataset in this way, the clusters can be of arbitrary shape, overcoming the major shortfall of the other paradigms considered here [299]. Furthermore, recognising data points in sparse areas of the surface (i.e. with low density) often affords density-based methods the automatic filtration of noise, without having to implement a specific schema. As such, several methods splicing density-based clustering with the other paradigms have been offered. For example, (Fast)DPeak [74], HDBSCAN [62] and DenPEHC [387] incorporate hierarchical structures with density-based decision making. Likewise, KIDBSCAN [358] utilises k-means clustering to identify centres with a high density.

Density-based clustering is relatively young when compared to other paradigms, only emerging at the end of the 20$^{\text{th}}$ century with the Density-Based Spatial Clustering of Applications with Noise (DBSCAN) algorithm [115]. Given its youth, contemporary clustering literature surveys (such as [181, 312]) lack thorough study of density-based clustering methods. Over time, specific surveys have been published, culminating in a most recent survey [43] which provides an exceptionally detailed review of density-based clustering algorithms. This survey includes a descriptive taxonomy of 32 density-based methods, categorised by their density definition, parameter sensitivity, mode of execution and the nature of the data being clustered.

DBSCAN is a point-based, parameter-sensitive clustering algorithm that has proven popular, and is now synonymous with density-based clustering. The initial work has spurred a slew of extensions to DBSCAN which aim to bypass its shortcomings,

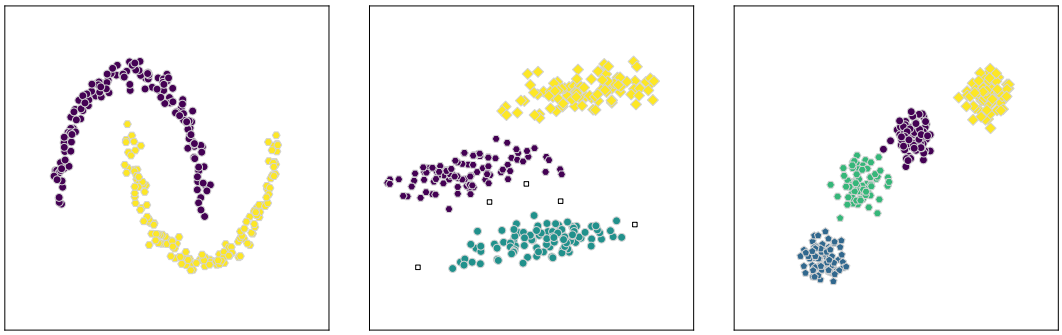
including Generalised DBSCAN [316], Incremental DBSCAN [29, 116], Improved DBSCAN [52], and various parallelised versions of the algorithm [51, 157, 223, 388] to overcome its time-complexity. The primary issues with DBSCAN relate to its parameters: a radius, $\epsilon > 0$, defining the neighbourhood of a point, and an integer, Minpts $\in \mathbb{N}$, that specifies the minimum cardinality of a neighbourhood required for a point to be considered dense. These parameters correspond to DBSCAN being point-based: regions of high density are determined using the points directly. The first disadvantage of DBSCAN is in estimating these parameters. The seminal work on DBSCAN [115] includes some default parameters based on dimensionality, but a good understanding of the dataset at hand is required for effective use. This issue leads to the second, which is that the parameters are global in scope, meaning that DBSCAN is unable to identify clusters that are of varying densities. Finally, the performance of DBSCAN is limited in high-dimensional space, because of the so-called *curse of dimensionality* [196] — the same is true of other clustering algorithms, including k-means.

However, these limitations are not without an alternative. For instance, CLIQUE [9] and OPTIGRID [161] each provide density-based clustering algorithms that are designed to perform well in high-dimensional space. Each of these algorithms is grid-based (as opposed to point-based), so the attribute space is discretised into rectangles. These rectangles are then used to identify dense regions. DCore [75] provides powerful point-based clustering in arbitrarily high dimensions.

To identify clusters of variable densities, DVBSCAN [294] and HDBSCAN [62] are available. The latter aggregates outputs from DBSCAN using a range of radii, making it more resilient to changes in either parameter. The former runs DBSCAN with the addition of allowing a cluster to expand provided its core points' local (ϵ-neighbourhood) density satisfy some criterion; because of this, it requires careful consideration of its parameters to perform well.

Lastly, any parameter-adaptive, density-based algorithm will do away with the trouble of choosing precise parameters. Examples include the popular OPTICS algorithm [16], DBCLASD [386] for spatial data, and DSets-DBSCAN [164]. The last provides a clustering that is independent of the parameters provided, but is exclusively for the clustering of images.

Figure 2.3 shows the equivalent plots to the other paradigms discussed here, with the clustering being performed by the DBSCAN algorithm. Unlike k-means or average-linkage, DBSCAN is able to identify the crescent moon shapes in Figure 2.3a. DBSCAN performs well at clustering the globular datasets. However, Figure 2.3b exemplifies the sensitivity of DBSCAN and its parameters; a small number of points



(a) (b) (c)

Figure 2.3: Clusters identified by DBSCAN on the (a) moons, (b) ellipses, and (c) spheres datasets

have been deemed outliers with the particular value of ϵ used here. DBSCAN narrowly improves upon the hierarchical method for the spheres dataset, but is unable to correctly identify all of the data points. Section 2.4.1 touches on the relevance of this occurrence.

2.3 Healthcare modelling

Healthcare modelling is a broad term that encompasses a plethora of techniques from a number of disciplines such as financial modelling and forecasting, and operational problems like vehicle routing or staff rostering. This review focuses on these operational pursuits, and particularly those that are concerned with patients directly. The decision to narrow the literature in this way is both practical and conscientious. Practical by reducing the span of literature on 'healthcare modelling' to something less cumbersome, and conscientious as it allows this review to make a small contribution to the research of the progressively more commonplace concept of patient-centred care. However, an array of comprehensive reviews are available, including [57, 131, 211, 277].

This form of healthcare, formally defined in [303], demands that the perspective of a healthcare system should align itself with its patients' needs and lived experiences. The alternative to patient-centred care would be a system in which patients are treated in an exact and unvaried way, according to the needs of the system, say. Some form of patient-centred care has been adopted in healthcare systems around the world including both state-funded and private systems [96, 98, 224]. Unsurprisingly, its application has been commended by patients, advocates and practitioners alike for improving condition-specific populations [125, 132, 139, 359] and more broadly [171, 301, 317].

Operational modelling and simulation form an integral part of advancing patient-centred care, and have been used to improve a great number of its facets, including patient safety [190], identifying patients in need of critical care [6], reducing length of stay [145], and implementing personalised monitoring [366]. The remainder of this section considers two techniques whose applications directly contribute to the advancement of patient-centred care: segmentation analysis and the modelling of queues. The former is often achieved through clustering, and could be regarded as a direct application of cluster analysis to healthcare populations. The latter is a broader form of analysis with use cases in a variety of healthcare problems, including resource utilisation [291], estimating bed capacity [381], and the taxonomy of patient pathways [305].

2.3.1 Segmentation analysis

Segmentation analysis allows for the targeted analysis of otherwise heterogeneous healthcare datasets and encompasses several techniques from operational research, statistics and machine learning. One of the most desirable qualities of this kind of analysis is the ability to glean and communicate simplified summaries of patient needs to stakeholders within a healthcare system [110, 355, 369, 393]. For instance, clinical profiling often forms part of the broader analysis where each segment is summarised in a phrase or infographic [370, 390].

The survey identified three commonplace groups of patient characteristics used to segment a patient population: system utilisation metrics; clinical attributes; and the pathway. The last is not used to segment the patients directly, but instead groups patients' movements through a healthcare system; this is typically done using a technique known as process mining. This technique originates in business analytics, and has been used to study the efficiency of hospital systems [18, 94]. The remaining characteristics can be segmented in a variety of ways, but recent works tend to favour unsupervised methods — typically latent class analysis (LCA) or clustering [389].

LCA is a statistical, model-based method used to identify groups (called latent classes) in data by relating the data instances to some unobserved (latent), categorical attribute. This attribute has multiple possible categories, each corresponding to a latent class. The discovered relations enable the observations to be separated into latent classes according to their maximum likelihood class membership [151, 215]. This method has proved useful in the study of comorbidity patterns (as in [212, 214]) where combinations of demographic and clinical attributes are related to various subgroups of chronic diseases.

As demonstrated in Section 2.2, clustering includes a wide variety of methods where the common theme is to maximise homogeneity within, and heterogeneity between, each cluster. Of those methods, k-means clustering (or a variant thereof) is the most widely used in healthcare [111, 152, 266, 318, 328, 370]; this is likely due to its simplicity and scalability. Hierarchical clustering methods have also been applied to operational healthcare problems in recognising patient utilisation patterns [396], profiling their healthcare preferences [220], and in mapping out effective healthcare leadership models [153].

Furthermore, in [370], hierarchical clustering is utilised as a part of the formative analysis, identifying a suitable number of clusters. In addition, supervised hierarchical segmentation methods such as classification and regression trees (as in [154, 210]) have been used where an existing, well-defined label or attribute is of particular significance. A crucial and attractive reason for using hierarchical methods in healthcare settings comes from the dendogram visualisation. The use of effective visualisation tools encourages (and allows, in some cases) involvement by key, expert stakeholders in the simulation process. The same is true of k-means for its straightforward construction. The authors of [179] present an analysis of the factors that result in a low level of engagement from stakeholders with healthcare simulation work. The key findings indicate that complexity and communication are the limiting factors for stakeholders, and so the onus resides on researchers to make their models informative, effective and transferable.

2.3.2 Queuing models

A queue models the arrival (and exit) of customers to (and from) a point of service. A queuing network describes how two or more queues may interact with each other. Since the seminal works by Erlang [113, 114] established the core concepts of queuing theory, the application of queues and queuing networks to real services has become commonplace, including healthcare. By applying these models to healthcare settings, many aspects of the underlying system can be studied. A common area of study in healthcare settings is of service capacity. The study [235] is an early example of such work where acute bed capacity was determined using hospital occupancy data. Meanwhile, more modern works consider more extensive sources of data to build their queuing models [278, 287, 381]. Moreover, the output of a model is catered more towards being actionable — as is the prerogative of operational research. For instance, in [287], the authors devise new categorisations for both hospital beds and arrivals that are informed by the queuing model. A further example is [205] where queuing models are used to measure and understand satisfaction among patients and staff.

In addition to these theoretic models, healthcare queuing research has expanded to include computer simulation models. The simulation of queues, or networks thereof, have the benefit of adeptly capturing the stochastic nuances of hospital systems over their theoretic counterparts. Example areas include the construction and simulation of Markov processes via process mining [18, 291, 300], and phase-type patient flow analysis [44, 236].

Regardless of the advantages of simulation models, a prerequisite for simulation research is having access to reliable software with which to construct those simulations. A common approach to building simulation models of queues is to use a graphical user interface such as Simul8. These tools are designed to be highly visual, making them attractive to organisations looking to implement queuing models without necessary technical expertise, including the NHS. The authors of [56] discuss the issues around operational research and simulation being taken up in the NHS despite the availability of intuitive software packages like Simul8.

However, they do not address a core principle of good simulation work: reproducibility. The ability to reliably reproduce a set of results is of great importance to scientific research but continues to be an issue in simulation research generally [124, 349]. When considering issues with reproducibility in scientific computing (simulation included), the source of any concerns is often with the software used [175]. Using well-developed open-source software can alleviate issues around reproducibility and reliability as how it is used involves less uncertainty and requires more rigour than 'drag-and-drop' software. One example of such a piece of software is the discrete event simulation library, Ciw [276].

The simulation of queues (or networks thereof) also makes the modelling of complex scenarios more practical and accessible. One scenario of interest in real-world settings is a queue with multiple customer classes. In a multi-class queue, each class of customer has its own set of attributes, which may include levels of priority for servicing, a specific arrival discipline or a particular service rate. Incorporating these characteristics allows for more realistic modelling and addresses the variety exhibited by agents in real-world queues.

Defining attributes such as these is straightforward in a computer simulation, but this scenario is certainly not restricted to empirical studies. For instance, in [306], the authors provide empirical, simulation-based evidence to dispel the idea that customers in multi-class queues experience identical waiting time distributions. Meanwhile, the authors of [288] demonstrate how a purely theoretical model of a motorway tollgate is improved by the use of multiple customer classes, and is verified using simulation. Finally, in [314], the authors derive a model that groups arrivals from the same class

together by way of a two-stage arrival process. The motivation for grouping arrivals is to provide a schema which supports the common assumption that reducing the number of times a server must change the task they are completing will influence the overall throughput of that server; the study succeeds in demonstrating this.

2.3.3 Queuing and clustering

The beginning of Section 2.3.2 briefly mentions two methods for investigating patient pathways — process mining and patient flow analysis — that are underpinned by a queuing network. In each case, the healthcare system is represented as a network of nodes through which patients and resources move at certain rates, as in a queuing network. Patient flow analysis considers the entire network, while process mining attempts to identify critical parts of the network.

The modelling of patient pathways is of great importance to understanding patient care in a healthcare system. Recording and studying how a patient moves through the system may provide insights into the dynamics of the characteristics presented by individuals, and how their true pathway differs from any previously defined clinical ideal. Two recent surveys on the literature surrounding the modelling of clinical pathways are [24, 121] with the latter focusing on process management tools such as process mining.

Often, clustering forms a fundamental part of these methodologies, creating a significant overlap. Clustering is a common resort in scenarios such as this where something must be implemented to rationalise an otherwise vast, heterogeneous set of behaviours. As an example, the authors of [291] use hierarchical clustering to group clinical pathways which, in turn, informs a broader, adaptive patient flow model. Meanwhile, the authors of [300] consider a pathway as a sequence of events sampled from a Markov chain. These chains (and their respective transition matrices) define the clusters in the sequence data, in accordance with the methodology presented in [61].

Other than process mining, clustering has acted as a catalyst for optimising other healthcare processes which relate to the efficient treatment of patients — namely, the scheduling and planning of resources, as exemplified in [256, 340, 394]. In each of these works, some form of clustering is used to categorise patients according to their requirements. This clustering then informs the scheduling model, providing a richer and more sophisticated picture of the overall system. With the other methods discussed in this survey, clustering exclusively sits in the post hoc stage of analysis. In process mining, a generalisation is formed for patient pathways. In segmentation analysis, clustering provides a patient profiling scheme.

Recalling the benefits of multi-class queues — being able to better model the variety of agents in real-world queues — it appears that clustering would be a natural fit to inform a model. Segmentation analysis would be able to quantise the idiosyncrasies of patients and quantify their effect on attributes of queuing networks, including arrival behaviour, transition probabilities, priority levels, and service requirements. However, the literature survey revealed scarce research into the application of clustering to inform a multi-class queuing model, theoretical or otherwise.

2.4 Evaluating a model

In [67], the author presents a seminal work discussing the proper evaluation of scientific models, drawing attention to a component of modelling that (even now) is often skimmed over. The author establishes a need for a clear distinction between *validation* and *corroboration* when evaluating a model; given that a model characterises a part of a wider reality through some parameterisation, no model can be 'valid', i.e. true. Instead, a model can be *confirmed* and support some hypothesis about the reality which it models. This sentiment is echoed by the adage, 'all models are wrong, but some are useful' [54], in that good models can and should illuminate a particular research problem. Works of this era raise important points, cautioning researchers about relying too heavily on their models, but the question still stands: how can (and should) a model be evaluated?

With the large-scale use of electronic data and the advent of computer simulation models, the importance of this question grows. The authors of [274] presented a landmark work that reiterates the concerns of earlier works, framing the question of model evaluation more squarely for computer simulation models. Again, one focus of [274] is on the use of particular language when discussing the quality of a model, i.e. that no model can be validated, but rather a model can be confirmed when its output aligns with some observed data. The authors go a step further in stating that any corroborative support offered by model confirmation is 'inherently partial' given the assumptions made to construct a model. However, this confirmation process has become the standard response for evaluating computer simulation models, i.e. choose a metric and dataset (or sets thereof) and use these to quantify the quality of the model either alone or against some competitor(s).

These works indicate that the evaluation of a model should not be purely mathematical or practical, and as such, it invokes a need for some philosophical thinking. In [281], the author consolidates the concerns of practitioners and philosophers regarding model evaluation, defining scientific models as representational tools constructed to achieve a particular purpose or goal; earlier and more contemporary

works concur with this definition [33, 67, 84]. The author suggests that model evaluation should be considered as the process of determining whether a model is *adequate* for a purpose, i.e. whether a model is highly likely to achieve a goal. The author notes the link between their notion of adequacy and a model being sufficient, following the principle of Occam's Razor, which is fundamental to scientific theory [372].

The author distinguishes the property of adequacy from model fitness and from individual adequacy, where a model achieves a purpose in a particular instance. The model presented in [203] exemplifies this last case. The authors present a model consisting of multiple phases in a specific, time-dependent process where the best-fitted model (i.e. that with the highest objective quality) produced the same results as a simpler one with fewer components on the data for which the model would be applied. The authors argue the importance of not doing more than is necessary to achieve their objective, and opt for the simpler model.

In [282], the same author elaborates on this adequacy-for-purpose view of model evaluation, highlighting the contrasts between determining adequacy of a model and measuring the accuracy of its representation against some target. In particular, the author emphasises the real-world dangers of overconfidence in a model's performance against any confirmation process. By defining models as tools constructed from parameters and simplifications, they are not generally boolean. Therefore, in general, they cannot be the subject of true confirmation. However, in practice, if a model is accredited through some confirmation process, the general merit and confidence in that model increases, potentially leading to the collective idea that the model has been 'confirmed' and may be 'better' than some other(s) without regard for its assumptions or limitations.

The author offers four practical methods for assessing adequacy-for-purpose that address evaluation at the point of model construction, and as a part of post hoc analysis. However, they conclude by recommending that any sufficient assessment suite should include a synthesis of the presented strategies, leaving open questions regarding the criteria of how each assessment should be aggregated to form a single conclusion. The author presents several examples and addresses potential concerns, indicating that the rigorous evaluation of any model requires careful, measured processes.

One quick alternative to this due diligence is to rephrase a purpose in terms of a reasonable limit on some criterion. For instance, rather than asking whether a method is adequate for predicting some quantity, the purpose could be rephrased as 'predicting some quantity with an accuracy of x%'. In these terms, the power of the statement is diminished, but practically, this question becomes much more straightforward to an-

swer. In addition, rephrasing the purpose in this way results in a conclusion which is broadly equivalent to a confirmation process, indicating little about the true quality or robustness of the method.

So, while these works are highly valuable in expanding the discussion of model evaluation, without concrete and readily applied methods with which to quantify their approaches, they are unlikely to be adopted widely. This unlikelihood is not necessarily the fault of the methods, but rather that (in some respects) convenience and speed have been allowed to outweigh merit in the evaluation of models. In an increasingly competitive environment where becoming the 'state-of-the-art' has become the norm, regard for robust, thoroughly evaluated models has declined. The remainder of this section summarises the established evaluation processes for clustering algorithms and healthcare models, highlighting their limitations and the avenues for improvement in the literature.

2.4.1 Clustering

Algorithms for clustering, like many machine learning tasks, fall into the category of models that are primarily evaluated using confirmation processes. As discussed at the beginning of this section, a confirmation process provides support for a model based on its alignment with a set of observed data according to some metric. Often lumped in with the task of classification, clustering is distinct in that it reveals the underlying structure of a dataset with knowledge that is fixed from the outset exclusively. However, the metrics used to evaluate clustering algorithms are often concerned with the retrieval of information such as accuracy, precision, and recall [231]. Examples of this method of evaluation are abundant, including many of the cited works from Section 2.2 such as [9, 12, 29, 64, 168].

Information retrieval metrics can be considered *external criteria* in that they measure an aspect of a model's clustering according to an external labelling of the dataset [247]. The quality of any clustering is predicated on the data generation process, and if a clustering algorithm cannot replicate the external labelling, it will not be deemed fit using external, label-dependent criteria. When discussing the illustrative figures in Section 2.2, an informal form of this evaluation was made when referring to a 'mislabelling' of points. In reality, the data synthesis produced a structure within the data that is not necessarily congruent with the generative process. Therefore, the clustering algorithms (which identify unknown structures) should not be penalised for not recognising these anomalies.

Without the use of an external truth, *internal criteria* exist to evaluate clustering algorithms. A criterion is considered internal if its sole source of information is

the clustering produced by the algorithm. For centroid-based and divisive clustering algorithms, a common internal measure is the inertia of a clustering, i.e. some loss function regarding the cluster centres [119]. Using inertia as a quality measure is convenient given that practical implementations of centroid-based methods use inertia as their objective function, meaning it is readily available [283]. However, interpreting inertia values can be difficult as they are dependent on the number of clusters and are scale-variant. As such, the inertia can be presented as a normalised measure using the all-cluster loss (i.e. the inertia when all points are contained in a single cluster), for instance.

Another popular internal quality measure is the silhouette coefficient [310] which presents a ratio between the tightness and separation of a set of clusters. This measure has the benefit of taking values in a finite range (from -1 to 1), making it interpretable across any given array of methods. Furthermore, it does not rely on cluster centres, and so can be applied to other clustering paradigms [133, 144, 337]. However, the primary drawback of the silhouette coefficient (as with inertia) is that it strongly favours linearly separable, spherical clusters. The authors of [126] present two cluster-quality measures that avoid this preference, but rely on knowing the 'true' clusters of a dataset.

Regardless of the nature of an objective metric, the evaluation of a method through corroboration can leave a method susceptible to societal biases. The recent widespread adoption of machine learning for human-related tasks has left society and machine learning inextricably linked, prompting discourse around its ethics, including these biases [30, 143, 243, 265, 315]. In [30], the authors welcome machine learning as a tool that may discover and incorporate subtle characteristics of a dataset that are otherwise imperceptible to humans, streamlining complex decision-making processes. However, they note that any evidence-based method cannot guarantee to produce a reliable or fair output; this is because of the inherent — and often unconscious — biases introduced in all aspects of the analysis. For instance, biases exist in datasets through a number of avenues, including (but not limited to) improper and unethical data collection practices [289], and existing socioeconomic inequities and prejudices [83, 296].

The other components which can be subjected to biases are the model design, and the evaluation of those models. To remedy any biases in these components, fair machine learning practices have been developed for designing algorithms [81, 127, 137, 195]. For clustering, these include the centroid-based method introduced in [71] and the hierarchical method from [10]. These methods tend to treat the data points as agents in the model, and rely on internal mechanisms that maximise their communal utility. Hence, they are in some respects self-validating. However, to be comparable,

objective metrics are applied showing that the costs of fairness are small losses in objective quality measures.

Further to particular metrics — objective or otherwise — clustering evaluation has been generalised to study more intrinsic properties of clustering instances such as clusterability [5, 275, 397]. These works often rely on an axiomatic description of clustering (examples of which include [41, 182, 201]). In [201], the author defines clustering as a function, f, over a dataset, X, which maps a distance metric, d, also over X to some partition, C, of X. The primary result of [201] is a theorem stating that no clustering function exists which satisfies all three of the following 'axiomatic' properties:

- Scale-invariance: the function is insensitive to changes in the unit distance
- Richness: all partitions of the dataset are possible outputs of the function (by use of an alternative distance function)
- Consistency: the outputted clustering is unchanged when within-cluster distances are decreased and between-cluster distances are increased

However, small changes to these properties accommodate various well-known clustering algorithms, including single-linkage and k-means clustering (discussed in Section 2.2). The authors of [38] expand on these axioms to derive necessary and desirable properties of cluster-quality measures that are entirely agnostic to the clustering function at hand. The closing sections of that work offer a number of exemplar cluster-quality measures based on loss, cluster centres and linkage, including measures that correspond to the all-cluster normalised mean-squared error, and a within-versus-between variability measure. The latter can be considered as a generalisation of the silhouette coefficient with respect to its loss function or distance measure.

2.4.2 Healthcare settings

As discussed in Section 2.3, data-driven methodologies have become prevalent in healthcare modelling. Incorporating computer-based modelling into decision-making processes allows healthcare practitioners and researchers to make better use of the volume of information available in increasingly complex healthcare settings [11, 37, 348, 355, 360].

However, since these methods directly impact the wellbeing of humans — perhaps in an unaudited manner — any data-driven model applied in healthcare requires a richer evaluation process to be deemed fit-for-purpose [48, 134, 162]. In healthcare settings, the entire application must be evaluated, not only the model itself. How-

ever, introducing machine learning techniques raises new questions (and reinforces existing questions) about the ethical quality of an application in terms of things like accountability [237, 307, 395] and patient safety [148, 238, 332]. Like machine learning more generally, this aspect of modern ethics research has been active in recent years, and extensive reviews on the subject are available as part of works seeking to map out points of concern [69, 143, 264, 293].

Ethical considerations are instrumental in developing trust in an application by stakeholders, which in turn marks its quality. In [85], the authors describe the perspectives of researchers and care providers on the ethical application of 'machine intelligence' in healthcare. They summarise that trust (for these stakeholders) requires confidence at all stages of an application: in the relevance of the data, the robustness and reproducibility of the system (i.e. the model itself), thorough auditing of the model output, and continual review of the entire workflow. They also make several calls for consistency across the application pipeline, including systematic review processes of existing methods, congruous terminology and nomenclature, and regularly reproducing existing model outputs on other data sources.

These calls for unified best practices are matched in recent literature [221, 249, 250, 368, 383]. For instance, the authors of [368] present a list of questions to ask of an application regarding its ethical quality. They state that despite the burgeoning promise of data-driven methods, there is a distinct lack of best practice guidance, stifling the calibre of the research topic more generally. Their questions address concerns for researchers and practitioners, as well as editors and patients, from the point of inception up to long-term implementation guidelines.

Another work attempting to consolidate these ethical considerations is [69]. Therein, the authors present a systematic framework for evaluating the ethical quality of a machine learning application in healthcare. Their framework, which has proven influential in other recent works (such as [123, 206]), is distilled from the collated literature on the ethics of machine learning in healthcare and non-healthcare settings. The framework considers an array of concerns from across the application pipeline which are relevant to healthcare, including recognising and mitigating the effect of societal biases, transparency, harm minimisation and accountability.

While this aspect of modelling research strives to improve and protect the standards of patient care, the presented solutions are primarily concerned with how data-driven healthcare applications can be improved for stakeholders within the healthcare system: practitioners and researchers. Patients form another group of stakeholders who are pivotal in the lifespan of any data-driven model. Patient-centred care relies on the involvement of patient experiences and feedback, and the authors of [123] argue

that the same ethos should apply with the application of new data-assisted methods. They offer a critique of the process introduced in [69], arguing that it concentrates too heavily on technical challenges, and does not provide sufficient consideration of the core ethical issues regarding the subject of the application: the patients.

The authors discuss how the perspective of a patient affects the real-time application of a data-driven healthcare model. Citing [100] as motivation, they lay out examples demonstrating patients' distrust of automated, algorithmic decisions despite their potential improvements over practitioners in terms of objective measures like diagnostic accuracy. In turn, they argue that this aversion may degrade the patient-practitioner relationship, leading patients to believe their agency is being undermined, and breaking down trust in the system. Ultimately, the authors conclude that any successful application of data-driven decision-making processes requires transparency and disclosure to patients by practitioners to be considered ethical, let alone fit-for-purpose. The authors do not, however, provide any practical solutions to these issues, unlike the authors of [206]. In that work, the authors emphasise the importance of trust in relationships in healthcare systems between patients, practitioners, and 'machines'. The authors offer additions to the process from [69] which include questions to protect and enhance the quality of those relationships.

Given the seeming inevitability that healthcare is only going to become more reliant on data-driven processes, a crucial and convenient avenue by which to maintain patient trust is to ensure the privacy of their data [172, 364]. In the EU, the introduction of strict data protection regulations has meant that the guidelines have been set out, but incidents of data mishandling continue to occur [35, 117]. The authors of [163] highlight the importance of properly using patient data (i.e. securely, ethically and effectively) in maintaining the unwritten social contract between a healthcare system and its population. Failing to go beyond the legal requirements and enact some social good can lead to the downfall of an initiative, such as with the NHS England *care.data* project [66].

A workaround to the potential misuse of patient data is to synthesise new datasets from existing, confidential data. The methods by which data synthesis is achieved are various, but include bespoke, often statistical models [88, 105, 241, 361] and, recently, generative adversarial networks (GANs) [25, 280, 356]. Using synthetic datasets provides another level of anonymity to the patients, but their validity as representations of real data has been contested. In a recent paper [297], the authors report consistent, although marginal, losses in accuracy by various supervised learning methods on synthetic datasets when compared with their performance on the corresponding real datasets. The authors of [88] find similar results, but demon-

strate how their method of data synthesis may improve certain objective metrics when the amount of real data available is small. This scarcity is a common issue in modern operational research, where data-hungry methods (i.e. those from machine learning) are becoming increasingly mainstream, with healthcare research included [279].

2.5 Chapter summary

This chapter has consisted of a review of existing literature, covering three central components of modern operational research: clustering, healthcare modelling and the evaluation of models. As part of this survey, the intersections of these topics have been considered, but there appears to be a distinct lack of literature covering all three topics simultaneously.

In the following chapter, a novel approach to understanding algorithm quality is presented. The motivation for this method stems from the literature reviewed in Section 2.4. Throughout Chapter 3, there is further discussion of the state of algorithm evaluation, particularly in machine learning. The focus of this discussion relates to the dangers of being too reliant on the confirmation processes reviewed in this chapter. A conformation process applies a method to a specific example and quantifies its success using some metric. The literature considered in this chapter revealed that this is the default approach to algorithm evaluation in machine learning — clustering included.

Of the clustering approaches available, this review confirms that centroid-based clustering (k-means, in particular) continues to be the most popular in healthcare applications. This popularity prevails in spite of its known limitations such as requiring spherical, evenly sized clusters. In part, this appears to be because of its straightforward, scalable and readily explained design, making it an appealing option when non-technical stakeholders are involved in a modelling process. Owing to this, centroid-based clustering algorithms are the focus of the clustering approaches used in Chapters 4 and 5.

Beyond healthcare applications, this survey recognises the growing body of work into fair machine learning practices, including in clustering. These new approaches seek to reframe machine learning techniques to focus on some measure of fairness that is often derived from some game-theoretic notion. In Chapter 4, a novel method is presented that would not necessarily be categorised as a fair machine learning algorithm, but rather adapts an existing algorithm to incorporate some concepts from game theory. The purpose of studying fair machine learning practices is to more

roundly quantify their quality, often from an ethical standpoint where people (and their data) are the subject of the algorithms.

Despite the widespread use of clustering in healthcare, the survey for this review identified no existing works where clustering was used to inform a queuing model. Indeed, clustering has been used to understand and categorise the outputs of healthcare queuing models, but does not appear to have been applied at the front-end of a model. In Chapter 5, administrative data is utilised to profile spells associated with a healthcare population via clustering. These profiles are then included as classes in a queue which models the population in a hypothetical hospital system.

The works introduced in Chapters 3 and 4 are vital in producing the model used in Chapter 5. As such, they exist as a whole, singular piece of research, forming this thesis. Importantly, each chapter provides novel contributions to the literature of their respective theme: evaluation, clustering, and healthcare modelling. However, and perhaps more importantly, this thesis acts as a substantial and unified contribution to the literature at the intersection of these themes, which is lacking.

Chapter 3

Evolutionary dataset optimisation

The research reported in this chapter has led to a publication [376] entitled:

"Evolutionary dataset optimisation: learning algorithm quality through evolution"

Available online at: doi:10.1007/s10489-019-01592-4
Associated data: doi:10.5281/zenodo.3492228
Source code: doi:10.5281/zenodo.3492236

The abstract of the publication is as follows:

In this paper we propose a novel method for learning how algorithms perform. Classically, algorithms are compared on a finite number of existing (or newly simulated) benchmark datasets based on some fixed metrics. The algorithm(s) with the smallest value of this metric are chosen to be the 'best performing'. We offer a new approach to flip this paradigm. We instead aim to gain a richer picture of the performance of an algorithm by generating artificial data through genetic evolution, the purpose of which is to create populations of datasets for which a particular algorithm performs well on a given metric. These datasets can be studied so as to learn what attributes lead to a particular progression of a given algorithm. Following a detailed description of the algorithm as well as a brief description of an open source implementation, a case study in clustering is presented. This case study demonstrates the performance and nuances of the method which we call Evolutionary Dataset Optimisation. In this study, a number of known properties about preferable datasets for the clustering algorithms known as k-means and DBSCAN are realised in the generated datasets.

The differences between this chapter and the publication are an extended discussion of the motivation behind the Evolutionary Dataset Optimisation method (in Section 3.1) and its components (Section 3.2), as well as a revised case study which concludes the chapter (Section 3.3).

The source code used to generate the plots and datasets in this chapter have been archived online under doi:10.5281/zenodo.4000316. The datasets themselves have been archived under doi:10.5281/zenodo.4000327.

3.1 Introduction

This chapter introduces a novel method called Evolutionary Dataset Optimisation (EDO). At its core, EDO is an evolutionary algorithm that acts on datasets to optimise some real-valued function. While it is possible to perform classical optimisation tasks with this method, its primary application is in learning the quality of an algorithm. The concept of an algorithm's quality here refers to some combination of its robustness, and its strengths and weaknesses.

When developing an algorithm to solve a given problem, questions are raised about its performance, both objectively and relative to existing methods. Determining convincing answers to these questions is an inherently difficult task. However, under the current regime, there is a standard response: take benchmark datasets and a common metric (or set thereof) amongst the proposed method, and its competitors, then assess the methods based on this metric and deem those with the smallest value to be 'best'.

Objectively, there is nothing wrong about comparing methods in this way except for the semantics of the outcome, i.e. outperforming a method on a dataset with a metric is insufficient evidence to categorise one method as 'better' than another. Each case can be qualified with something along the lines of "Method A performs better than Method B under the given conditions", but there are concerns about this process that persist beyond linguistic hair-splitting.

A significant concern presented by this process is in how benchmark examples are selected; there is no real measure of their reliability other than their frequent use. There do exist benchmark dataset suites that are curated to be relevant, diverse and comprehensive for some problem domains — such as machine learning [104, 271] and time series [90] — but it is often the case that a dataset becomes a benchmark for merely being long-standing and used many times. This title awards the dataset

with the accolades of being reliable and trustworthy. However, this is not guaranteed.

Computer vision is one such domain where these questionable de facto benchmarks have come to exist. In [289], the authors dissect the unethical and problematic practices used in the creation and aggregation of several benchmark datasets from computer vision, including the renowned *ImageNet* datasets [95]. These practices pose serious questions about the credibility of the models trained using these benchmarks, both morally and as a matter of their performance. The exposition highlights questions of consent and privacy as well as revealing a valid moral quandary given that the social, cultural and racial biases transferred from these datasets to the models will then diffuse into systems that are synonymous with life in the age of 'Big Data'.

As an example of the reality of these systemic biases, in 2015, it was made public that the automatic classifier developed as part of *Google Photos* had been incorrectly labelling images of people of colour as gorillas. Google publicly apologised and vowed to fix the problem, but since then the only action taken to mitigate this has been to remove several primates from the set of labels available to their model [330].

Leaving computer vision aside, the authors of [63] raise questions about the availability and suitability of benchmark datasets in the field of unsupervised outlier detection. The authors point out that even though systematic approaches exist for the generation of benchmark datasets, the approaches are not sufficiently documented to be reproducible, thus rendering them scientifically moot.

In addition to this, the authors discuss the troubles that come with co-opting datasets designed for another task (classification in their case) in the absence of existing benchmarks designed for outlier detection. This practice is indicative of another issue with this aspect of the current paradigm where convenience has become a driving force for benchmark selection rather than merit.

Striving for convenience may well be an issue that stems from the competitive nature of algorithm design. In order for a method to become 'state-of-the-art', there has to be some comparable evaluation with existing methods. However, this should not be the end of the line when discussing the quality of any method. More extensive work is required to understand an algorithm truly and to quantify its quality, which leads to the other source of concern in the established process: the methods themselves.

Holding a method to account on a finite number of example datasets — regardless of their reliability or diversity — limits the amount of learning one can gain about that method. In particular, it limits the understanding of the characteristics which lead to good or bad performance to those attributes present in the set of example

datasets. Another example from computer vision [357] shows that Support Vector Machines (SVMs) — the use of which is ubiquitous in classification — fail to perform well when tested on a dataset containing comparable and broadly equivalent items to the one on which they have been trained. So, despite the abundant use of SVMs, even in the then-'best' image classifier, there should be concerns about the robustness of the model.

Taking a step back from examples of empirical algorithm evaluation, consider the space between algorithms and data more generally. To evaluate an algorithm, i.e. a fixed point in the space of algorithms, one maps it to a finite subset of points in the space of datasets using some metric(s). How that subset is determined is what has been discussed thus far. The process when travelling in the opposite direction is not so standardised, but it appears more rigorous.

Suppose that the object of interest was not an algorithm but rather a dataset. In this case, the objective is to determine a preferable algorithm to complete some task on the data. There exist many ways of achieving this that appear in a range of disciplines. However, each takes into account the constraints and characteristics of the data and the context of the research problem. These methods are often equivalent to asking questions of the data and can include the use of diagnostic tests. For instance, in the case of clustering, if the data displayed an indeterminate number of non-convex blobs, then one could recommend that an appropriate clustering algorithm would be DBSCAN [115]. Otherwise, for scalability, k-means may be chosen [385, 398].

The EDO method belongs to a new paradigm that aims to flip the process described here by allowing the data itself to be unfixed. EDO achieves this fluidity by generating data for which the algorithm of interest performs well (or better than some other) through the use of an evolutionary algorithm (EA). The purpose of doing so is not only to create a bank of useful datasets, but rather to allow for the subsequent studying of those datasets. Undergoing this study reveals the attributes and characteristics which lead to the success (or failure) of the algorithm, giving a broader understanding of the algorithm on the whole. Figure 3.1 provides a diagram of this framework. On the right: the current path for selecting some algorithm(s) based on their validity and performance for a given dataset; on the left: the proposed flip to better understand an algorithm by exploring the space in which 'good' datasets exist for that algorithm.

As will be seen later in this chapter, the most substantial limitation of this paradigm is in defining a meaningful fitness function for the EA. Section 3.3 acts as a tutorial on how to resolve this issue using an iterative approach where the fitness function

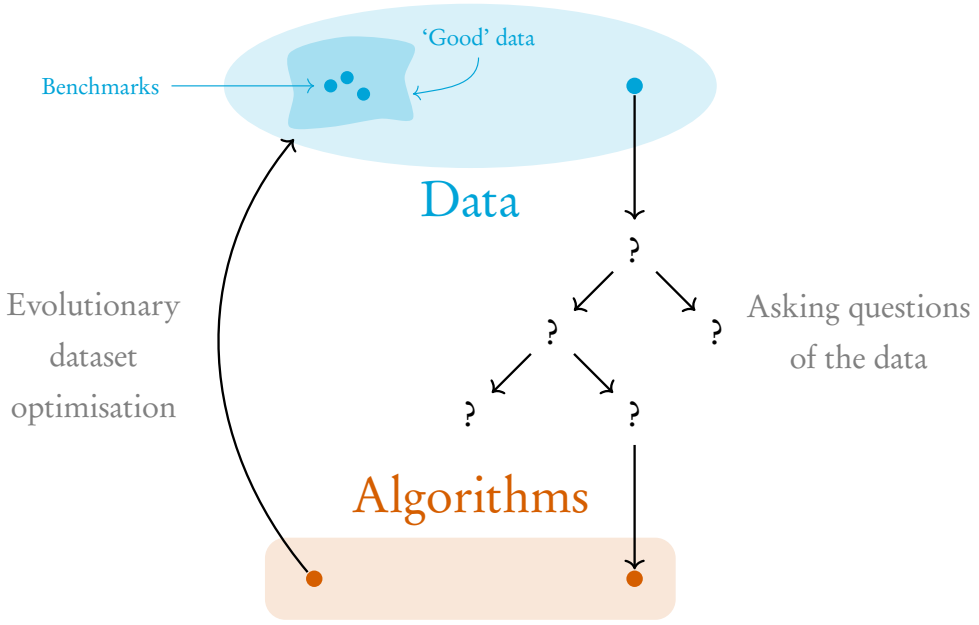


Figure 3.1: A diagram of the current and proposed paradigms for algorithm evaluation

is adjusted so as to avoid redundant, trivial or reductive results. Although the case study in that section of this chapter focuses on clustering, the generic form of the EA means that any method which acts on tabular data is suitable for study using EDO.

It is also important to note that the method described in this chapter is merely one element of this new paradigm, and is one that utilises evolution. EAs have been applied successfully to solve a wide array of problems — particularly where the complexity of the problem or its domain is significant. These methods are highly adaptive, and their population-based construction (displayed in Figure 3.2) allows for the efficient solving of problems that are otherwise beyond the scope of traditional search and optimisation methods. An EA approach has been chosen here as they are uncomplicated in design, yet their capabilities encompass the difficulties of the flipped paradigm set out above.

The use of EAs to generate artificial data is not a new concept, however. Applications of EAs to data generation have included developing methods for the automated testing of software [204, 246, 324] and the synthesis of existing or confidential data [72]. Such methods also have a long history in the parameter optimisation of algorithms, and recently in the automated design of convolutional neural network (CNN) architecture [342, 344].

Other methods for the generation or synthesis of artificial data are numerous and range from simple concepts such as simulated annealing [234] to swarm-based learn-

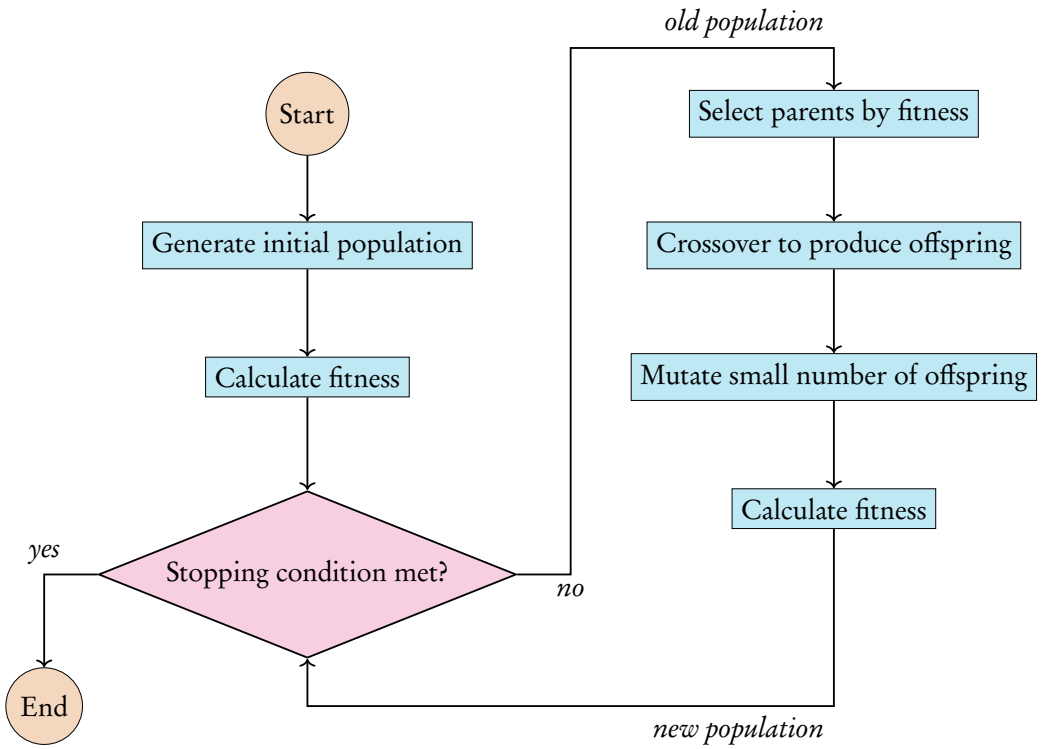


Figure 3.2: A general schematic for an evolutionary algorithm

ing techniques [4] or generative adversarial networks (GANs) [140]. The unconstrained learning style of methods like CNNs and GANs aligns with those in the proposed paradigm, and with EDO in particular. By allowing the EA to explore and learn about the search space in an organic way, an unprejudiced insight can be established that is not necessarily reliant on any particular framework or agenda.

Note that there is no necessary restriction on the search space to be of a fixed dimension or data type such as the method described in [72]. The shape of a dataset is considered a part of the search space itself that can be traversed through the EA.

The remainder of this chapter is structured as follows:

- Section 3.2 describes the parameterisation, structure and components of the EDO method.
- Section 3.3 contains a case study examining the success and failure of k-means clustering using EDO. Included also is a comparison between k-means and another clustering algorithm DBSCAN.
- Section 3.4 summarises the chapter.

In addition to the case study at the end of this chapter, the EDO method is instrumental in evaluating the algorithm presented in Chapter 4.

3.2 The evolutionary algorithm

This section presents the details of the EDO algorithm. As stated previously, the EDO method is an EA. The EA follows a typical schema with the addition of some features that align with the overall objective of artificial data generation. With that, there are a number of parameters that are passed to EDO. These include the typical parameters of an evolutionary algorithm: a fitness function, f, which maps from an individual to a real number (or some quantity which can be ordered), a population size, $N \in \mathbb{N}$, a maximum number of iterations, $M \in \mathbb{N}$, a selection parameter, $b \in [0,1]$, and a mutation probability, p_m.

In addition to these, EDO takes a number of parameters which allow the EA to act on datasets of varying sizes and forms. The required parameters are as follows:

- A set of probability distribution families, $\mathcal{P}$. Each family in this set has some parameter limits which form a part of the overall search space. For instance, the family of normal distributions, denoted by $N(\mu, \sigma^2)$, would have limits on values for the mean, μ, and the standard deviation, σ.
- A maximum number of *subtypes* for each family in $\mathcal{P}$. A subtype is an independent copy of the family distribution that progresses separately from the other subtypes in that family. These are the actual distribution objects which are traversed in the optimisation and that are passed to the individuals.
- A probability vector to sample distributions from $\mathcal{P}$, $w = (w_1, \ldots, w_{|\mathcal{P}|})$.
- Limits on the number of rows an individual dataset can have,

$$R \in \left\{(r_{\min}, r_{\max}) \in \mathbb{N}^2 \mid r_{\min} \leq r_{\max}\right\}$$

- Limits on the number of columns a dataset can have,

$$C := (C_1, \ldots, C_{|\mathcal{P}|}) \text{ where } C_j \in \left\{(c_{\min}, c_{\max}) \in (\mathbb{N} \cup \{\infty\})^2 \mid c_{\min} \leq c_{\max}\right\}$$

  for each $j = 1, \ldots, |\mathcal{P}|$. That is, C defines the minimum and maximum number of columns a dataset may have from each distribution in $\mathcal{P}$.

The remaining two parameters are optional, but can be useful for introducing and focusing exploration, respectively:

- A second selection parameter, $l \in [0,1]$, to allow for a small proportion of 'lucky' individuals to be carried forward.
- A shrink factor, $s \in [0,1]$, defining the relative size of a component of the search space to be retained after adjustment.

Choosing an optimal set of parameters for any given run of EDO is not an objective exercise — as is the case with all EAs — and some experimentation may be required. For the dataset-specific parameters — like $\mathcal{P}$, R, and C — these decisions are practical in nature for the most part. Large datasets (controlled by R and C) will require more computational power and storage, for instance. For the EA-specific parameters, there are some rules of thumb such as:

- Choosing a population size, N large enough to take a representative sample from the search space. $N = 100$ is a good starting point, but larger populations should provide better samples.
- Setting the selection pressure, b, so as to encourage the preservation of favourable characteristics in the individuals, and not to dilute the next population. Typical values for achieving this with other EAs range between 0.1 and 0.25.
- Including enough mutation (through p_m) to force some heterogeneity in the population and to further explore the search space. However, heavy-handed mutation will likely cause the EA to diverge, leading to reductive results. Typical mutation probabilities are between 0.01 and 0.05.

Algorithm 3.1 provides a high-level description of the EDO algorithm, presenting its general structure. This section comprises more detailed discussion — along with relevant examples, diagrams and algorithm statements — for the processes mentioned there: the creation of individuals, the evolutionary operators and the 'shrinkage' process. In addition, an overview of a software implementation is provided.

Note that there are no defined processes for how to stop the algorithm or adjust the mutation probability, p_m, in this chapter. This generality is deliberate and is down to the particular use case. Some examples include:

- Regular decreasing in mutation probability across the available attributes [207].
- Stopping when no improvement in the best fitness is found within some K consecutive iterations [217].
- Utilising global behaviours in fitness to indicate a stopping point [233].

3.2.1 The software implementation

The remainder of this section discusses the components and mechanisms of the EDO method in a largely mathematical manner. However, a Python implementation of EDO has been developed as part of this thesis, `edo`. The software is freely available online under the Massachusetts Institute of Technology (MIT) licence, and can be found hosted on GitHub (at `github:daffidwilde/edo`) or archived on Zen-

Algorithm 3.1: The evolutionary dataset optimisation algorithm

Input: $f, N, R, C, \mathcal{P}, w, M, b, l, p_m, s$
Output: A full history of the populations and their fitnesses.

```
1  begin
2      create initial population of individuals
3      find fitness of each individual
4      record population and its fitness
5      while current iteration less than the maximum and stopping condition not met
        do
6          select parents based on fitness and selection proportions
7          use parents to create new population through crossover and mutation
8          find fitness of each individual
9          update population and fitness histories
10         if adjusting the mutation probability then
11             update mutation probability
12         end
13         if using a shrink factor then
14             shrink the mutation space based on parents
15         end
16     end
17 end
```

Algorithm 3.2: Creating a new population

Input: parents, $N, R, C, \mathcal{P}, w, p_m$
Output: A new population of size N

```
1  begin
2      add parents to the new population
3      while the size of the new population is less than N do
4          sample two parents at random
5          create an offspring by crossing over the two parents
6          mutate the offspring according to the mutation probability
7          add the mutated offspring to the population
8      end
9  end
```

odo (at doi:10.5281/zenodo.2552890). Furthermore, edo is registered on the Python Package Index and can be installed through standard Python practices, including via pip — as in Snippet 3.1.

```
> pip install edo
```

Snippet 3.1: Installing the edo library via pip

The edo library is built on the scientific Python stack [240, 270] and has been developed to be consistent with the current best practices of open-source software development; these are discussed in Section 1.4. As such, the software is modular, automatically tested and fully documented. The documentation for edo is available at edo.readthedocs.io, and Figure 3.3 contains a screenshot of one of the tutorial pages.

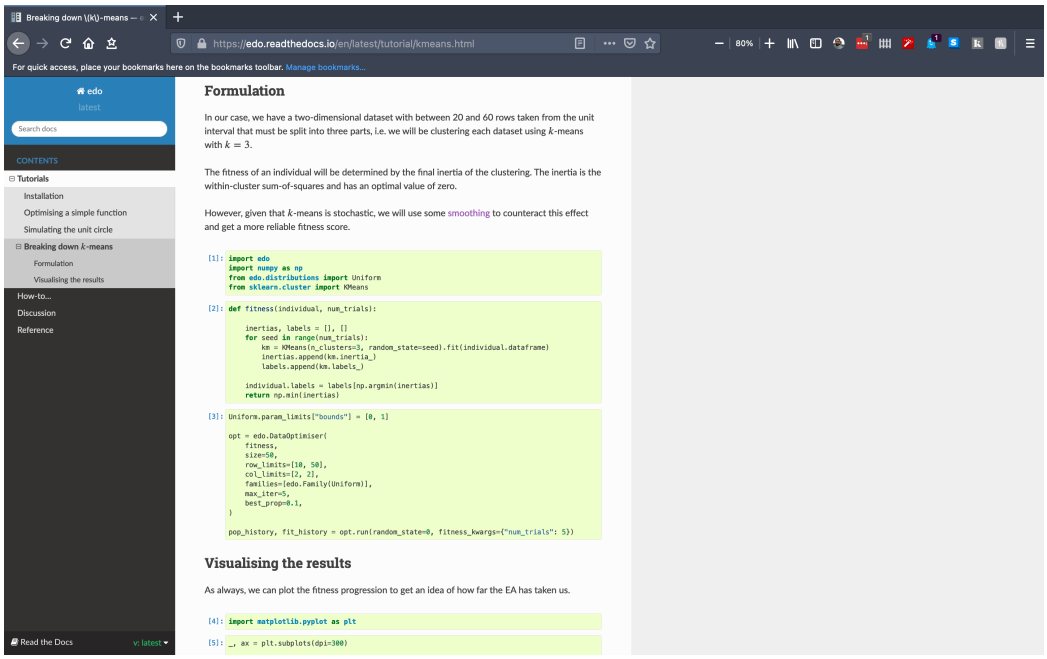


Figure 3.3: A screenshot of one of the edo library tutorials

In Section 3.3, the edo implementation is used to carry out an investigation into k-means clustering. The tutorial shown in Figure 3.3 is a simplified version of the example in Section 3.3.1. Given the size and number of examples in the closing section of this chapter, a command-line tool, edolab, is used to run the experiments. To demonstrate how edo should be used directly, consider the following scenario.

Let X be a dataset with two columns, X_r and X_a, which each contain $n \in [50, 100]$ real values. The values of X_a are drawn uniformly from the interval $[-2\pi, 2\pi]$, while

those in X_r may take values from the interval $[0,2]$. Again, these values are sampled uniformly.

The objective is to find a dataset, X, which maximises the following function:

$$f(X) = \frac{\mathrm{Var}(X_a)}{\max_{x \in X_r} |x - 1|} \tag{3.1}$$

That is, to find a dataset with maximal variance in one column, and minimal maximum distance from one in the other. Such a dataset would describe the polar coordinates of some set of points along the unit circle. The points in X_r correspond to the radii, and those in X_a correspond to the angle from the origin in radians.

The first step in implementing this scenario in `edo` is to choose appropriate distribution classes. These classes go on to form the families in EDO, $\mathcal{P}$. The `edo` documentation includes detailed instructions on how to implement new distribution classes (`edo.readthedocs.io/.../how-to/new_column.html`), but several common distributions are already implemented; these include the uniform distribution. Since both X_r and X_a sample their values uniformly and from different bounds, separate subclasses of the `edo.distributions.Uniform` class must be created. Snippet 3.2 shows how this should be done.

```
>>> import numpy as np
>>> from edo.distributions import Uniform
>>>
>>> class RadiusUniform(Uniform):
...     """A uniform distribution for radii"""
...     name = "RadiusUniform"
...     param_limits = {"bounds": [0, 2]}
>>>
>>> class AngleUniform(Uniform):
...     """A uniform distribution for angles"""
...     name = "AngleUniform"
...     param_limits = {"bounds": [-2 * np.pi, 2 * np.pi]}
```

Snippet 3.2: An `edo` implementation for separate uniform distribution classes

With the column distributions implemented, the fitness function needs defining. Snippet 3.3 contains the code to implement the fitness function given in (3.1). Since each column of the dataset is of a specific type, they must be recovered properly; this is done using the `split_individual` function.

Now all the required components are ready, the EDO algorithm can be run using the `edo.DataOptimiser` class. Snippet 3.4 shows the code required to run the EDO algorithm on this scenario across several random seeds. These seeds make the runs repeatable, and their outputs reproducible.

```
>>> def split_individual(individual):
...     """ Separate the columns of an individual's dataframe. """
...     df, metadata = individual
...     names = [m.name for m in metadata]
...     radii = df[names.index("RadiusUniform")]
...     angles = df[names.index("AngleUniform")]
...     return radii, angles
>>>
>>> def fitness(individual):
...     """ Determine the similarity of the dataset to the unit circle. """
...     radii, angles = split_individual(individual)
...     return angles.var() / (radii - 1).abs().max()
```

Snippet 3.3: Implementing the circle fitness function in `edo`

```
>>> import edo
>>> import pandas as pd
>>>
>>> pop_histories, fit_histories = [], []
>>> for seed in range(5):
...     families = [edo.Family(RadiusUniform), edo.Family(AngleUniform)]
...     opt = edo.DataOptimiser(
...         fitness,
...         size=100,
...         row_limits=[50, 100],
...         col_limits=[(1, 1), (1, 1)],
...         families=families,
...         max_iter=30,
...         best_prop=0.1,
...         maximise=True,
...     )
...     pops, fits = opt.run(random_state=seed)
...     fits["seed"] = seed
...     pop_histories.append(pops)
...     fit_histories.append(fits)
>>>
>>> fit_history = pd.concat(fit_histories)
```

Snippet 3.4: Running the circle scenario in `edo` across five seeds

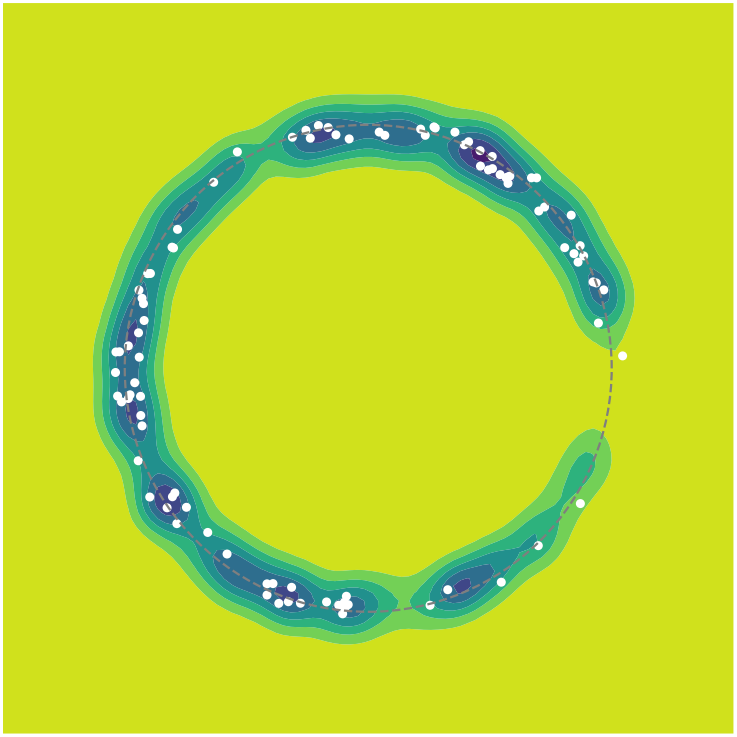


Figure 3.4: A contour and scatter plot of the circle example results

Here, the remaining parameters — population size, selection parameters, etc. — are mainly illustrative and such that this example can run in good time. However, despite this potential roughness, the results are good. Figure 3.4 shows a contour of the top 50% of individuals, which is well-distributed about the unit circle. In addition, the best-performing individual from across these runs is shown as a scatter.

3.2.2 Individuals

Evolutionary algorithms operate in an iterative process. At each iteration, the EA acts on a population (generation) of *individuals*. Each individual corresponds to a solution to the problem in question according to some representation or encoding. In a genetic algorithm, an individual is a solution encoded as a bit string of typically fixed length and is treated as a chromosome-like object to be manipulated.

In EDO, individuals are represented primarily as the dataset that defines them, without an encoding. This is because the objective of EDO is to generate datasets and explore the space in which datasets exist. Therefore, to design meaningful operators on these solutions, this form is preserved. Creation is one such operator that is governed by this representation. Figure 3.5 shows this process diagrammatically and Algorithm 3.3 provides a simplified statement of the individual-creation process.

In addition to the dataset, an individual is represented by a list of probability distributions. These distributions are created using the elements of $\mathcal{P}$ and correspond to the columns of the dataset. This list is referred to as the individual's *metadata*. The metadata acts as a set of instructions for sampling new values for the columns (as in mutation). Also, the metadata is a record of how that column was created.

However, one should not assume that the columns are a reliable representative of the distribution associated with them or vice versa; this is particularly true of 'shorter' datasets with only a few rows, whereas confidence in the pair could be given more liberally for 'longer' datasets with a more significant number of rows. In any case, appropriate methods of analysis should be employed before formal conclusions are made about these relationships.

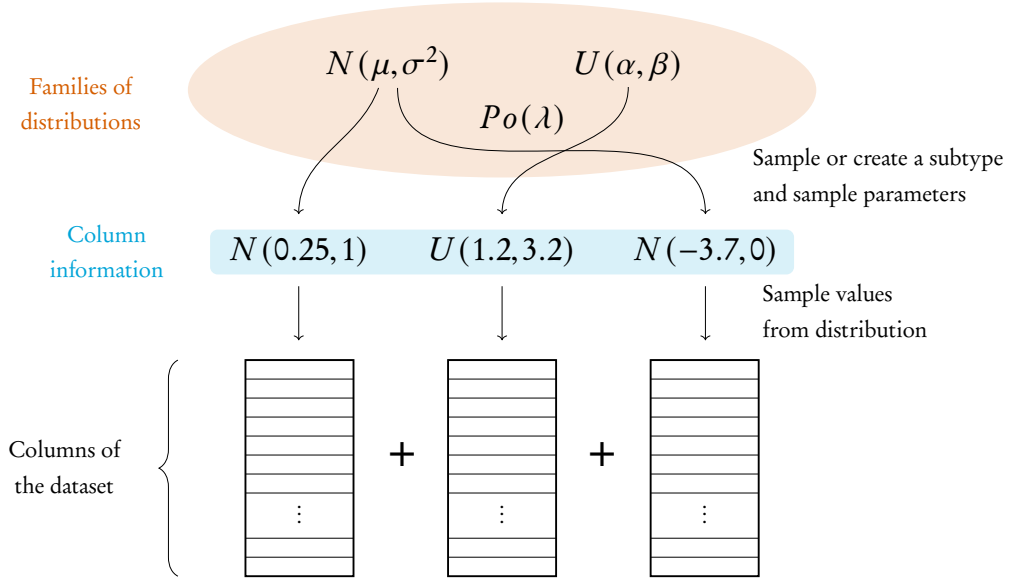


Figure 3.5: An example of how an individual is first created

Algorithm 3.3: Creating an individual

Input: $R, C, \mathcal{P}, w$
Output: An individual defined by a dataset and some metadata

```
1  begin
2      sample a number of rows and columns
3      create an empty dataset
4      for each column in the dataset do
5          sample a distribution from P
6          create an instance of the distribution
7          fill in the column by sampling from this instance
8          record the instance in the metadata
9      end
10 end
```

3.2.3 Selection

The *selection* operator describes the process by which individuals are chosen from the current population to generate the next. Almost always, the fitness of an individual determines the likelihood of their selection to be a parent. By selecting individuals in this way, the hope is for the preservation of some favourable qualities — thus improving the population — and to encourage some homogeneity within future generations [27].

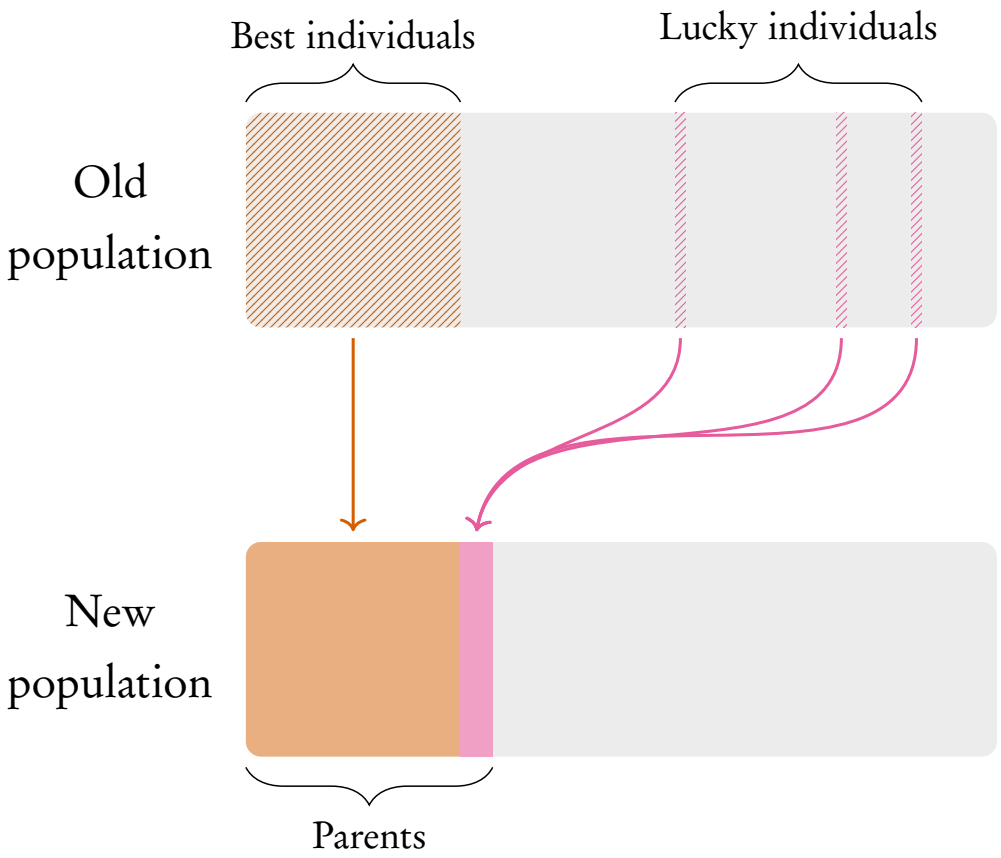


Figure 3.6: The selection process with the inclusion of some lucky individuals

A modified truncation selection method is used in EDO, as is illustrated in Figure 3.6. Truncation is perhaps the simplest selection method wherein a fixed number, $n_b = \lceil bN \rceil$, of the fittest individuals in a population are taken forward and used as the *parents* of the next generation. These parent individuals are also referred to as the 'best' individuals in their population, as in Figure 3.6. Note that this selection process is 'memoryless' in a sense, and an individual could potentially be present throughout the entirety of the EA.

Despite its efficiency as a selection operator, truncation selection can lead to premature convergence at local optima [183, 252]. EDO provides an optional modification to counteract this where, after the best individuals have been chosen, some number,

Algorithm 3.4: The selection process

Input: population, population fitness, b, l
Output: A set of parent individuals

```
1 begin
2     calculate n_b and n_l
3     sort the population by the fitness of its individuals
4     take the first n_b individuals and make them parents
5     if there are any individuals left then
6         take the next n_l individuals and make them parents
7     end
8 end
```

$n_l = \lceil lN \rceil$, of the remaining individuals can be selected uniformly to be carried forward. The purpose of taking forward a small number of 'lucky' individuals is to introduce some diversity in the genetic pool of the parent individuals, thus adding to the exploration of the search space.

After the parents have been selected, there are two adjustments made to the current search space. The first is that the subtypes for each family in $\mathcal{P}$ are updated to include only those present in the parents. The second adjustment is a process which acts on the distribution parameter limits for each subtype in $\mathcal{P}$ and takes place once the new generation has been created. This adjustment gives the ability to 'shrink' the search space about the region observed in a given population. This method is based on a power law described in [14] that relies on a shrink factor, s. At each iteration, t, every distribution subtype which is present in the parents has its parameter's limits, (l_t, u_t), adjusted. This adjustment is such that the new limits, (l_{t+1}, u_{t+1}) are centred about the mean observed value, μ, for that parameter:

$$l_{t+1} = \max\left\{l_t,\ \mu - \frac{1}{2}(u_t - l_t)s^t\right\} \tag{3.2}$$

$$u_{t+1} = \min\left\{u_t,\ \mu + \frac{1}{2}(u_t - l_t)s^t\right\} \tag{3.3}$$

The shrinking process is given explicitly in Algorithm 3.5. Note that the behaviour of this process can produce reductive results where early convergence is achieved at the cost of extensive exploration. For these reasons, shrinking is an optional component of EDO.

Algorithm 3.5: Shrinking the mutation space

Input: parents, current iteration, $\mathcal{P}, M, s$
Output: A new mutation space focussed around the parents

```
1  begin
2      for each distribution subtype in P do
3          for each parameter of the distribution do
4              get the current values for parameter over all parent columns
5              find the mean of the current values
6              find the new lower (3.2) and upper (3.3) bounds around the mean
7              set the parameter limits
8          end
9      end
10 end
```

3.2.4 Crossover

Crossover is the operation of combining two individuals in order to create a new individual (or individuals). It is also the opportunity to have the favourable qualities preserved through selection interact with one another in potentially new ways. The term *crossover* originates from its application in genetic algorithms where it is quite literal. In genetic algorithms, two bit strings are crossed at a point to create two new bit strings.

Another popular method is uniform crossover, where the components of two parents are sampled uniformly to create a new individual. This method is efficient and is effectual in combining individuals to preserve homogeneity in both bit string and matrix representations [73, 322]. EDO makes use of a form of uniform crossover that has been adapted to support the representation of individuals in the EA. Put simply: a new offspring is created by uniformly sampling each of its components (i.e. dimensions and columns) from a set of two parent individuals, as depicted in Figure 3.7 and described in Algorithm 3.6.

Observe that there is no requirement on the dimensions of the parents to be of similar or equal shapes. This laxness is allowed because the proposed method allows for individuals of different shapes, and their combination can be reconciled because of how individuals are represented. Where there is an incongruence in the lengths of the two parents, missing values may appear in a shorter column that has been sampled. New values are sampled from the probability distribution associated with that column to fill in these gaps. Conversely, surplus values are trimmed from the bottom of all longer columns.

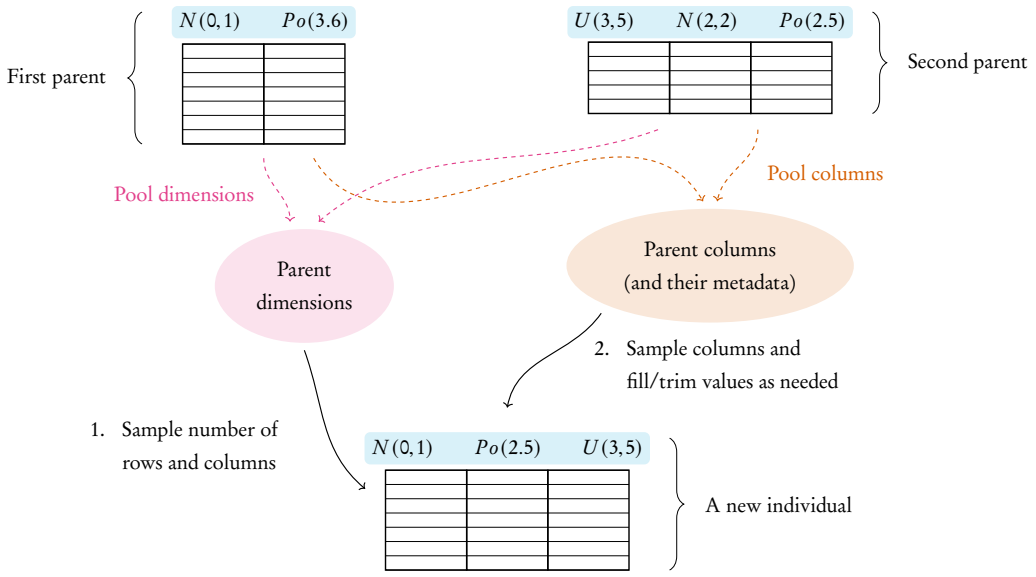


Figure 3.7: The crossover process between two individuals with different dimensions

Algorithm 3.6: The crossover process

Input: Two parents
Output: An offspring made from the parents ready for mutation

```
1  begin
2      collate the columns and metadata from each parent in a pool
3      sample each dimension from between the parents uniformly
4      form an empty dataset with these dimensions
5      for each column in the dataset do
6          sample a column (and its corresponding metadata) from the pool
7          if this column is longer than required then
8              randomly select entries and delete them as needed
9          end
10         if this column is shorter than required then
11             sample new values from the metadata and append them to the
                 column as needed
12         end
13         add this column to the dataset and record its metadata
14     end
15 end
```

3.2.5 Mutation

The *mutation* operator is used in EAs to maintain a level of variety in a population. This operator effectively forces the algorithm to explore more of the search space at each generation. It is typical of mutation operators to affect all aspects of an individual. In genetic algorithms, this is as straightforward as running along a bit string and swapping a zero to a one (or one to zero). Under the EDO framework, the mutation process manipulates the phenotype of an individual by potentially modifying its dimensions and the entries of its dataset. Figure 3.8 gives a diagrammatic description of this process, and a formal statement of the algorithm is described in Algorithm 3.7.

In the publication that initially presented EDO, this process included a penultimate stage where the metadata of an individual could be mutated. This manipulation sampled new parameter values for each distribution in the metadata with the mutation probability, p_m. Following subsequent testing, it became apparent that this kind of mutation led to confusing results. In particular, studying the resultant individuals became more complicated when individuals retained values in their columns that were now beyond any reasonable bounds of the associated distribution, for instance. Since removing this stage of the process, no noticeable impact has been identified on the ability of the EA to traverse the search space compared with its inclusion.

Each of the potential mutations occurs with the same probability p_m. However, the way in which columns are formed and stored (with their associated metadata) ensures that even multiple mutations in the dataset will only result in some incremental change in the individual's fitness relative to, say, a completely new individual. This assertion relies on appropriate choices for f and $\mathcal{P}$.

The following section addresses how over-sensitivity and observable weak points in a fitness function impact the performance of the method. Addressing how to make a good choice for distribution families, $\mathcal{P}$, is not as clear-cut a process, but all use cases of this method have indicated that even a basic choice of distribution is sufficient. The following case study requires a continuous variable so the uniform distribution is used. Despite its simplicity, the EDO method is able to generate datasets with interesting structural properties. The analysis in Chapter 4 makes use of a discrete uniform distribution and, likewise, the EDO method is able to offer up datasets with more interest than a random cloud, as could be expected with a uniform distribution.

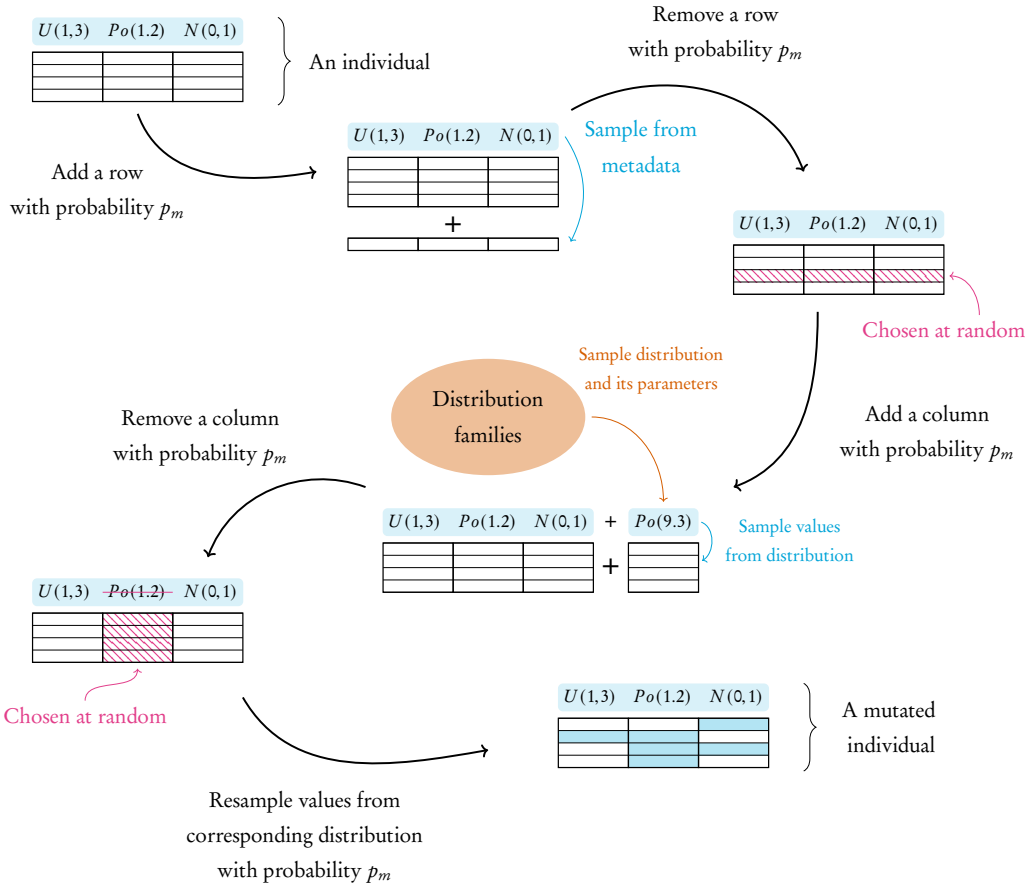


Figure 3.8: The mutation process

Algorithm 3.7: The mutation process

Input: An individual, p_m, R, C, $\mathcal{P}$, w
Output: A mutated individual

```
1  begin
2      sample a random number r ∈ [0, 1]
3      if r < p_m and adding a row would not violate R then
4          sample a value from each distribution in the metadata
5          append these values as a row to the end of the dataset
6      end
7      sample a new r ∈ [0, 1]
8      if r < p_m and removing a row would not violate R then
9          remove a row at random from the dataset
10     end
11     sample a new r ∈ [0, 1]
12     if r < p_m and adding a new column would not violate C then
13         create a new column using P and w
14         append this column to the end of the dataset
15     end
16     sample a new r ∈ [0, 1]
17     if r < p_m and removing a column would not violate C then
18         remove a column (and its associated metadata) at random from the
             dataset
19     end
20     for each entry in the dataset do
21         sample a random number r ∈ [0, 1]
22         if r < p_m then
23             sample a new value from the associated column distribution
24             update the entry with this new value
25         end
26     end
27 end
```

3.3 A case study in clustering

The following case study contains three examples that act as a form of validation for EDO. These examples also highlight some of the nuances in its use. This case study uses the proposed method to reproduce some known results about the clustering of data in the absence of any external forces and examines how clustering algorithms are typically evaluated. In particular, the focus will be on the well-known k-means (Lloyd's) algorithm.

Clustering has been chosen here as it is a well-understood problem that is accessible — most notably when restricted to two dimensions. However, the overall approach taken in this case study is generic and would apply to any machine learning method. First, choose an algorithm (or set of algorithms) and set the initial fitness function to be a metric of interest for that method. The remainder of the process is a cycle of studying the generated datasets and adjusting the fitness function to reveal new insights into the algorithms under study.

3.3.1 Inertia and k-means clustering

The k-means algorithm is an iterative, centroid-based method that aims to minimise the *inertia* of the current partition, $Z = \{Z_1, \ldots, Z_k\}$, of some dataset X:

$$I(Z, X) := \frac{1}{|X|} \sum_{j=1}^{k} \sum_{x \in Z_j} d(x, z_j)^2 \tag{3.4}$$

A full statement of the algorithm to minimise (3.4) is given in Algorithm 3.8.

As this inertia function is the objective of the k-means algorithm, it is often used for evaluating the quality of the final clustering it produces. However, since it is not a normalised measure, other metrics are often used. Many of these metrics — such as accuracy, recall and precision — are used under the assumption that clustering is some sort of unsupervised classification task. This assumption is fundamentally wrong. Therefore, as a starting point, the first example uses inertia as the fitness function in EDO. That is, EDO is used to find datasets that minimise the final inertia found by k-means clustering.

For visualisation purposes, these examples will restrict EDO to datasets that are two-dimensional, i.e. $C = ((2, 2))$. For simplicity, each dataset will be clustered into three parts, i.e. $k = 3$, and have its columns formed from uniform distributions, denoted by U, enclosed by the unit interval. Thus, the search space is the unit square, and

Algorithm 3.8: k-means (Lloyd's algorithm)

Input: a dataset X, a number of centroids k, a distance metric d
Output: a partition of X into k parts, Z

begin
  select k initial centroids, $z_1, \ldots, z_k \in X$
  while *any point changes cluster or some stopping criterion is not met* **do**
    assign each point, $x \in X$, to cluster Z_{j^*} where:

$$j^* = \arg\min_{j=1,\ldots,k} \left\{ d\left(x, z_j\right)^2 \right\}$$

    recalculate all centroids by taking the intra-cluster mean:

$$z_j = \frac{1}{|Z_j|} \sum_{x \in Z_j} x$$

  end
end

the only element of $\mathcal{P}$ is:

$$\mathcal{U} := \{U(\alpha, \beta) \mid \alpha, \beta \in [0, 1]\} \tag{3.5}$$

The remaining parameters are as follows: $N = 100$, $R = (50, 100)$, $M = 100$, $b = 0.1$, $l = 0$, $p_m = 0.01$, and shrinkage is excluded. This set of parameters has been adapted from that used in [376]. The changes are: omitting the trivial case where the number of rows equals k; shortening the run time to reduce computational resources; and, finally, increasing the selective pressure (by reducing b) to mitigate the effect of noise in later generations.

In addition to these parameter changes, the fitness function has been altered. In this study, every individual is scaled using a min-max scaler so their values are in the interval $[0, 1]$. This makes the values of the limits in (3.5) arbitrary and eliminates the pinching effect observed in [376] where well-performing individuals were disproportionately compact. Following this scaling, the number of initialisations for the k-means algorithm has been increased from ten to 50 so that there is greater confidence in any given fitness score.

The examples in this study make use of a command-line tool, `edolab`, for running experiments with the library. This tool allows for a lot of otherwise repeated code to be replaced by an *experiment* script, configuring the parameters of the experi-

```python
""" /path/to/experiments/kmeans_inertia.py """

from edo.distributions import Uniform
from sklearn.cluster import KMeans
from sklearn.preprocessing import MinMaxScaler


def fitness(individual, max_seed=5):
    """ Return the lowest final inertia of k-means on the individual
    across the given number of trials with k=3. """

    data = MinMaxScaler().fit_transform(individual.dataframe, copy=True)

    inertias = []
    for seed in range(max_seeds):
        km = KMeans(n_clusters=3, random_state=seed).fit(data)
        inertias.append(km.inertia_)

    return min(inertias)


size = 100
row_limits = [50, 100]
col_limits = [2, 2]
max_iter = 100
best_prop = 0.1
lucky_prop = 0
mutation_prob = 0.01

Uniform.param_limits["bounds"] = [0, 1]
distributions = [Uniform]
```

Snippet 3.5: An abridged version of the experiment configuration script used in the first example

```
> cd /path/to/experiments
> edolab run --seeds=10 --cores=4 kmeans_inertia.py
> edolab summarise --tarball kmeans_inertia.py
```

Snippet 3.6: Example usage of the `edolab` command-line tool

ment. The source code for the `edolab` package is hosted on GitHub (`github:daffidwilde/edolab`) and the tool itself is registered on the Python Package Index. Snippet 3.5 shows the experiment script used for this example, and Snippet 3.6 shows how to use that script with the command-line tool. Other than the fitness function definition, this script is identical to that of every example in this section.

Once the EDO algorithm has terminated, a body of datasets, and information about those datasets, is recorded. This output is referred to as a *history*. A history created by EDO can be exceptionally large; some of the preliminary experiments conducted for this chapter produced hundreds of gigabytes of data.

The potentially storage-hungry nature of an EDO history can be seen as follows.

Given a population of size 100 and a maximum number of iterations of 100, then $100 \times 100 = 10,000$ datasets will be written to file. If each dataset takes up 1kB of disc space, then the history will use at least 10MB to write the datasets themselves. In addition to these, EDO saves information about each individual's metadata as well as the distribution subtypes, families and the states of the pseudo-random number generators used by the method. All of this information is essential to thoroughly study the history and to recover its individuals, but it is plain to see how this all adds up.

Each experiment in Chapter 4 that uses EDO produced tens of thousands of unique datasets. Having volumes of data of these sizes certainly provides a rich source for study, but they can be cumbersome to the point of being completely infeasible. As such, one should be mindful of the storage capacity of the computer being used. Further to that, if fitting the data comfortably into memory is a concern then not all of the data must be studied; the analysis in Chapter 4, for instance, only considers the fittest percentile of datasets produced.

Figure 3.9 shows the progression of the fitness function (inertia) and the number of rows at ten generation intervals across the history generated with the parameters defined above. There is a steep learning curve here; within the first ten generations the population fitness gains substantially, and although there is constant improvement to the median and best fitness scores, the pace slows over the remaining generations.

The same quick convergence is evident in the number of rows where it is there is a clear preference for datasets with fewer rows. Wanting fewer rows is expected given that inertia is the sum of the mean error from each cluster centre. Then, with k fixed a priori, a quick (although not guaranteed) way of reducing this mean error is to reduce the number of points in each cluster; doing this reduces the number of terms in the second summation of (3.4). [376] included the case where $r_{min} = 3$ in this example, and the EA successfully identified it before promptly getting stuck there.

Aside from these progressions, a more focused look may be taken at the generated datasets. Figure 3.10 shows the individuals with a fitness closest to the lowest, median and highest values across the entire history after min-max scaling. These individuals correspond to the best-, most-middling- and worst-performing individuals in said history. It should be noted that any individual from any generation may be retrieved and studied with this implementation. The summary provided here is one particular way of studying the body of datasets that have been generated. This transparency in the history and progression of the EDO method sets it apart from other

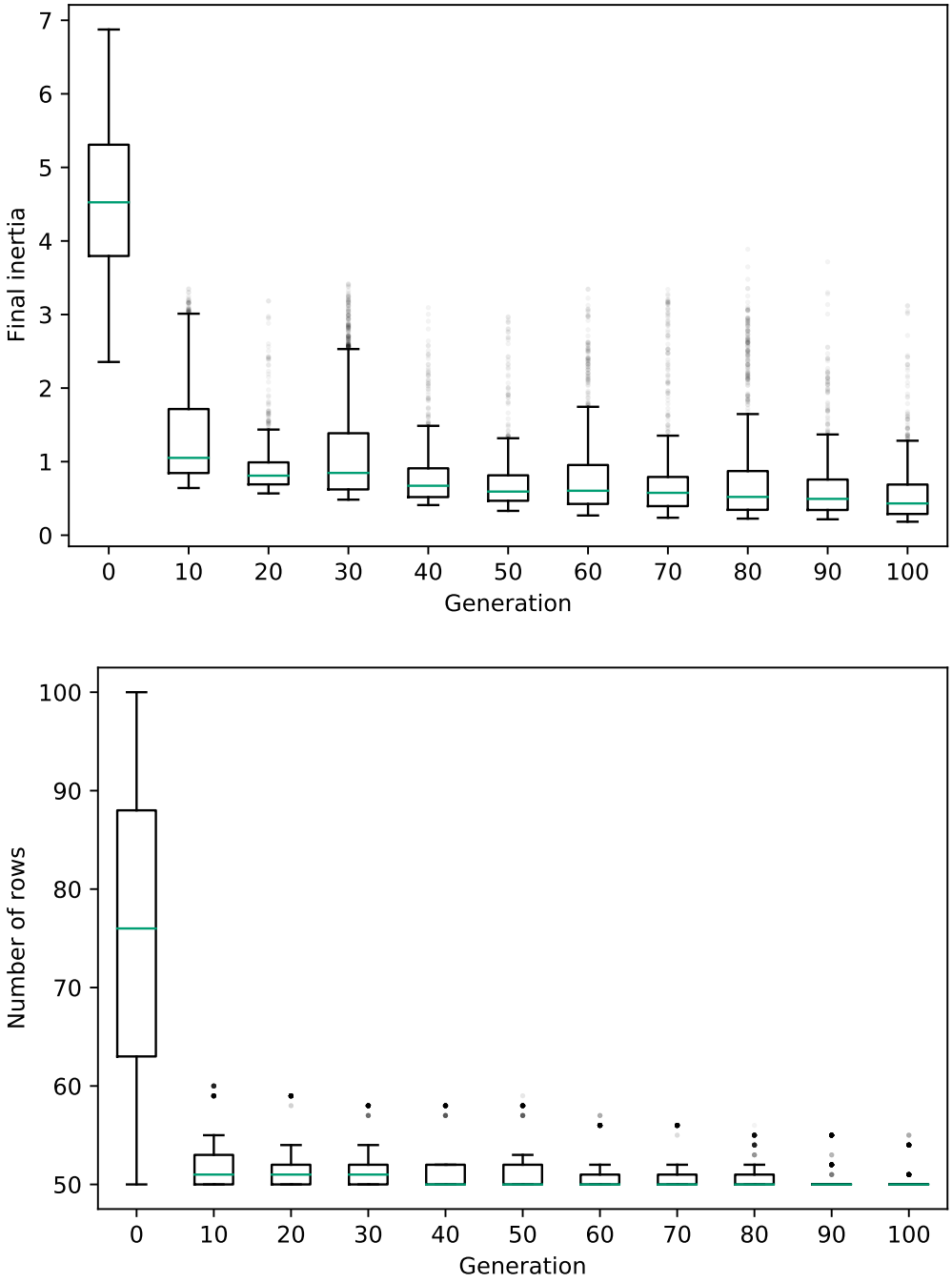


Figure 3.9: Progressions for final inertia and the number of rows

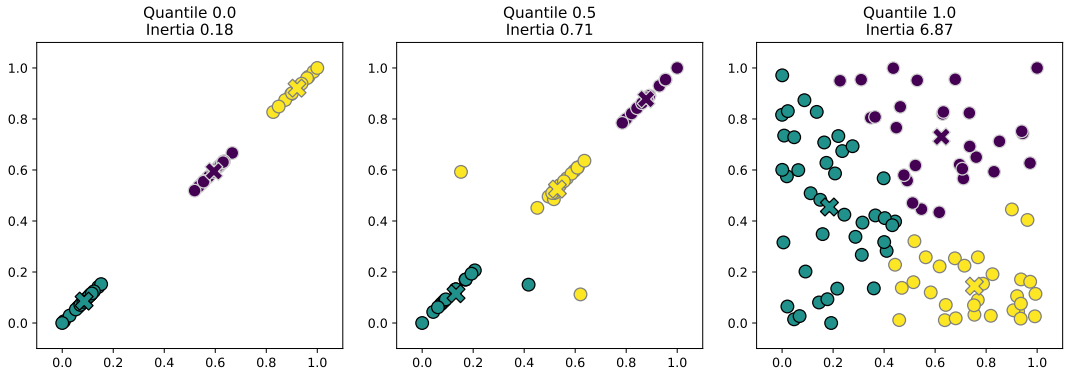


Figure 3.10: Representative individuals from EDO trials with inertia

methods such as GANs which have a reputation of providing so-called 'black box' solutions.

In this case, as may have been expected, the worst individuals take the form of random clouds with no distinct clusters. However, there are some patterns in the better-performing individuals. It is clear that there is a preference for tight clusters, but also it appears that well-performing datasets have columns with a strong positive correlation. Having such a relationship may seem irrelevant to the success of k-means, but in doing so, the dataset becomes one-dimensional. Removing a dimension reduces the search space of the algorithm considerably, and makes it easier for the k-means algorithm to achieve its real goal of finding the centroidal Voronoi tessellation of a dataset [103]. When restricted to a single dimension and with $k = 3$, the optimal tessellation is equivalent to finding a clustering that matches the tertiles of a set of numbers, and when $k = 2$ this is the same as finding the median.

In the first example of [376], the best and median individuals showed clusters that were all virtually the same point. Although having exceptionally compact clusters provides optimal values of I, it leaves little else to be learnt. That kind of behaviour was exhibited, in part, because it was allowed; the fitness function did nothing to penalise the proximity of the inter-cluster means. There is no need for this penalty currently because of the scaling step before the application of the k-means algorithm. By scaling the dataset, the entire unit square must be used by every individual, thus reducing the effect of that otherwise dominant behaviour.

In this way, inertia could be considered a flawed fitness function. Without the scaling step, for instance, inertia produces near-trivial results, which begs the question: is there anything else to be learnt here? The answer is, "probably". There are two options available to draw more learning from this algorithm. Either change the parameters passed to EDO, or modify the fitness function. Given that useful results have already been found with this parameter set, and to avoid cherry-picking any fur-

ther results by tweaking parameters, this study considers the latter for the remaining examples.

3.3.2 The silhouette coefficient

In the example above, the presence of strong positive correlations stood out when using inertia as the fitness function — as such, counteracting that effect is the focus of this example.

This study aims to understand k-means clustering, and therefore, the fitness function(s) used should somehow measure the efficacy of the identified clustering. The silhouette coefficient is one such metric. The silhouette coefficient evaluates what is colloquially referred to as the 'appropriateness' of a particular clustering of a dataset [310]. The metric considers both the intra-cluster cohesion and inter-cluster separation of a clustering. The silhouette coefficient of a clustering, Z, is given by the mean of the silhouette value, $S(x)$, of each point $x \in Z_j$ in each cluster:

$$\begin{aligned} A(x) &:= \frac{1}{|Z_j| - 1} \sum_{y \in Z_j \setminus \{x\}} d(x, y), \\ B(x) &:= \min_{k \neq j} \frac{1}{|Z_k|} \sum_{w \in Z_k} d(x, w), \\ S(x) &:= \begin{cases} \frac{B(x) - A(x)}{\max\{A(x), B(x)\}} & \text{if } |Z_j| > 1 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{3.6}$$

The optimisation of the silhouette coefficient is analogous to finding a dataset which maximises both cohesion (the inverse of A) and separation (B). Hence, the silhouette coefficient addresses the overall objective of minimising inertia by maximising cohesion. Meanwhile, the silhouette coefficient has the added benefit of being normalised and takes values in the interval $[-1, 1]$. Although, k-means will not (in general) produce a clustering with a negative silhouette score since the clusters are formed from a partition of the plane and cannot overlap, meaning that the range of expected silhouette scores should be $[0, 1]$.

The silhouette fitness function, with the same EDO parameters, yields the results summarised in Figure 3.11, and the individuals shown in Figure 3.12. Note that the order of the individuals is from worst to best here since the fitness function should be maximised.

As was the case in the previous example, there is a steep learning curve followed by steady, incremental improvements to the population fitness. Moreover, the produced

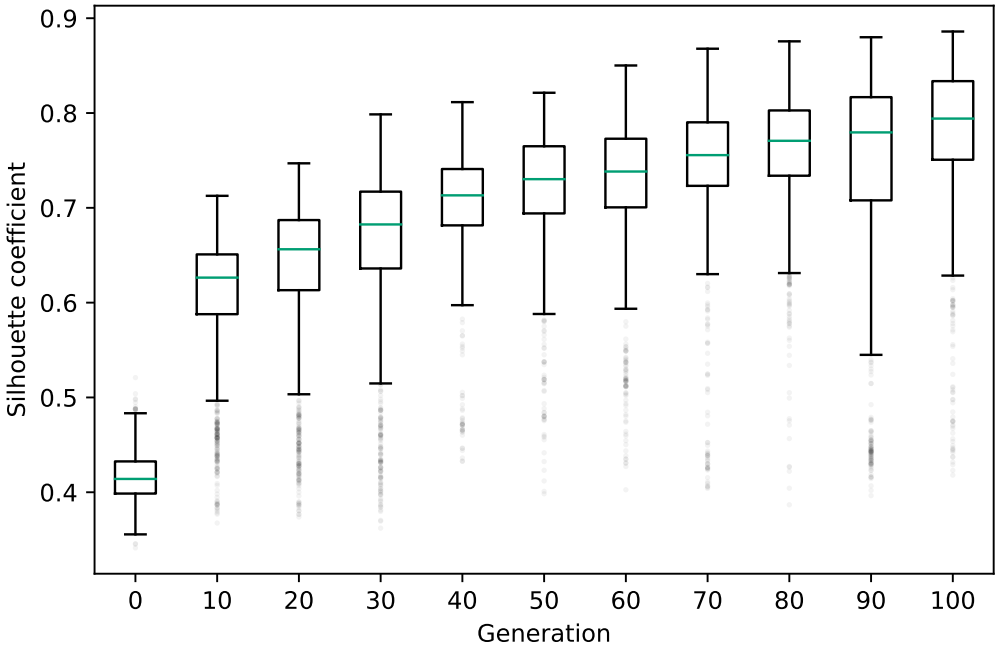


Figure 3.11: Progression plot for the silhouette fitness function

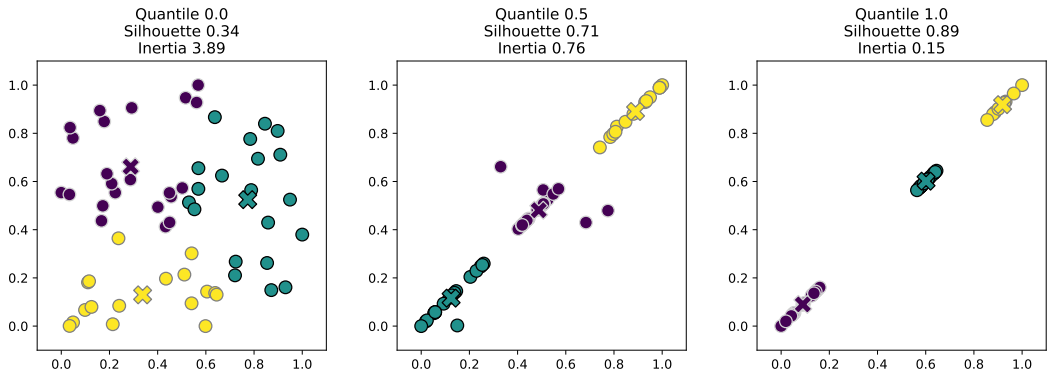


Figure 3.12: Representative individuals from EDO trials with silhouette

datasets are markedly similar to those created using inertia. The clusters identified in [376] showed an "increased separation from one another whilst maintaining low values in the final inertia" when using this fitness function. In this case, the produced datasets appear to be no different from those made using inertia, for which the scaling step is mostly responsible. By preprocessing the data to fill the unit interval, the notion of cluster separation has, in effect, been maximised, leaving only cohesion to be optimised. Although this is is only strictly true for the outer bounds of each cluster, this observation means silhouette fitness function is broadly equivalent to the previous inertia fitness function. This equivalence is seen further by the close fit of inertia scores in the better-performing representative individuals displayed in Figure 3.12.

So, solely using the silhouette coefficient provides no further insight. However, given

that it is normalised, it can be discounted in a meaningful way. The previous example found that positive correlation was a driving force in the learning achieved by EDO. The same is evident here. Therefore, a sensible adjustment to the fitness function would be to penalise positive correlation directly. As such, the adjusted fitness function is:

$$f(X) := S(X) - |\rho(X)| \tag{3.7}$$

where $S(X)$ is the silhouette coefficient of a dataset X when clustered by k-means and $\rho(X)$ is the Pearson correlation coefficient of the columns of X. This function will be referred to as the *discounted silhouette* fitness function.

The same method could be used with inertia, but the effect of the discounting term would be lost when inertia is high — as is common in the early stages of the EA (see the top plot of Figure 3.9) — rendering the exercise pointless. However, its effect integrates well with the silhouette coefficient:

- The optimal score is the same ($1 - 0 = 1$) as the silhouette fitness function while the worst score is similar ($0 - 1 = -1$).
- A score of zero still indicates there is little to be gained in that (at the extremes) the dataset has a perfect silhouette and perfect correlation or no silhouette with no correlation, indicating a random cloud.
- Negative scores indicate a dataset with a low silhouette coefficient and high correlation, i.e. high inertia but correlated and, therefore, unwanted.

Figures 3.13 and 3.14 show a summary of the results generated using this discounted silhouette function with the same parameters as used in the previous examples. The fitness progression shows a steady increase for the best individuals at each generation — as has been the case with the first two cases — and there is the same preference for datasets with fewer rows. This trend in the fitness means that EDO is indeed optimising across the search space. However, there appears to be some variation in the population fitness here, which may indicate that the parameter set requires some tweaking or, simply, that the environment in which individuals exist is more competitive. Since the EA is still producing passable results, the parameters will not be adjusted.

Figure 3.14 highlights the impact of a well-adjusted fitness function. Consider the leftmost frame of either plot, where the worst-performing individual is shown. This kind of individual would have been selected immediately with either of the previous fitness functions, and its cluster centres separated along the diagonal. Instead, the simple modification to the fitness function has relegated it to the bottom of the pop-

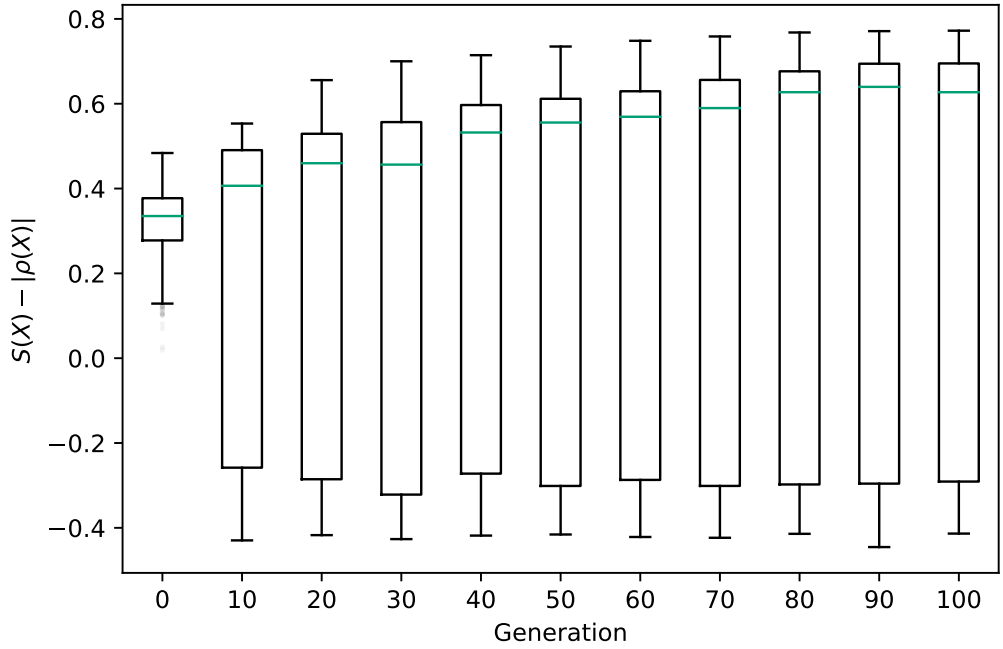
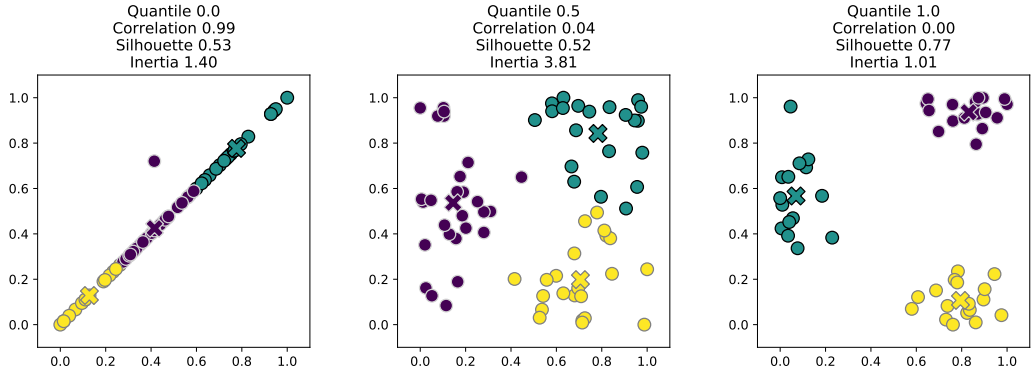
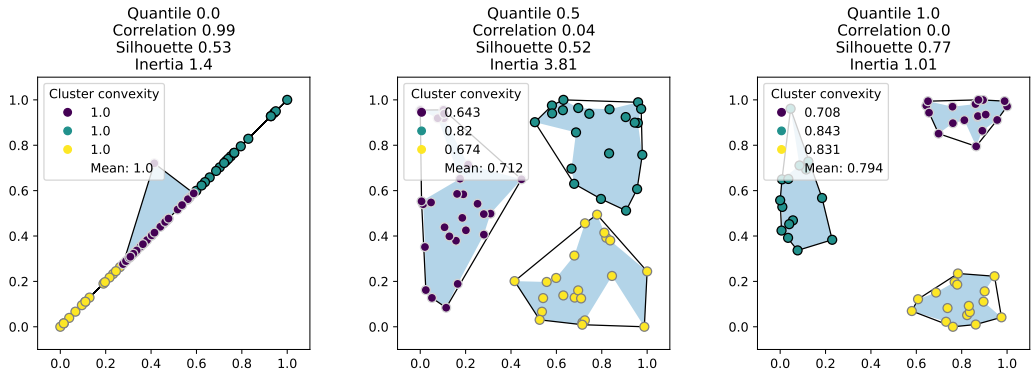


Figure 3.13: Progression plot for the discounted silhouette fitness function



(a) clusters and their centres



(b) clusters and their hulls

Figure 3.14: Representative individuals from EDO trials with discounted silhouette

ulation. Then, inspecting the other two datasets shows that EDO has generated more 'realistic' individuals that offer a more polished silhouette without being reductive. Although this is a somewhat simplistic example, it demonstrates how a genuinely useful and well-formed dataset may be created objectively.

Another point of interest here is the convexity of the clusters. A known condition for the success of k-means is that the presented clusters are of roughly equal size and are convex. Without this condition, up to the correct choice of k, the algorithm will fail to produce satisfactory results for either inertia or silhouette. This condition is derived from the link between k-means and Voronoi tessellation. In [335], the authors define the convexity of a set of points (such as a cluster), denoted C, as the ratio of the areas of its concave and convex hulls, denoted H_c and H_v, respectively:

$$C := \frac{\text{area}\,(H_c)}{\text{area}\,(H_v)} \tag{3.8}$$

Here, a cluster's *concave hull* is taken to be the α-shape of the cluster's data points [107], where $\alpha \in \mathbb{R}$ is the smallest value such that all points in the cluster are contained in a single polygon.

With this definition, it should be clear that a perfectly convex cluster, such as a single point, line or convex polygon, would have $C = 1$. Also, it appears that the mean convexity of the clustering increases as fitness increases (save for the 'invalid' individual) and suggests that this condition for convex clusters is sought out during the optimisation process.

3.3.3 Comparison with DBSCAN

The extent of the capabilities EDO holds as a tool to better understand an algorithm are especially apparent when comparing an algorithm against another (or a set of others) simultaneously. To compare a set of algorithms, one must utilise the freedom of choice in a fitness function for EDO. Consider two algorithms, A and B, and some common metric between them, g. Then their similarities and contrasts can be explored by considering the differences in this metric on the two algorithms, i.e. using $f = g_A - g_B$, $f = g_B - g_A$ or $f = |g_B - g_A|$ as the fitness function. Doing so can highlight pitfalls, edge cases or fundamental conditions for the method(s). Overall, this process affords a deeper level of learning about the method of interest beyond the traditional empirical approach of comparison on a particular example.

The final example in this case study considers a comparison of k-means with another clustering algorithm of a different form: Density-Based Spatial Clustering of Applications with Noise (DBSCAN). The objective of the first part of this example is to

find datasets for which k-means outperforms its alternative, DBSCAN. There is no concept of inertia in DBSCAN as it is based on density (as opposed to raw distance) and identifies outliers. A full statement of the algorithm is given in [115]. Without inertia, a valid metric must be chosen. Again, the silhouette coefficient is one such metric.

The standard silhouette function is used here over that defined in (3.7) for two reasons. First, the relationship between correlation and DBSCAN has not yet been established in this study, and second, to simplify the final fitness function. An adjustment to (3.6) must be made since the silhouette coefficient requires at least two clusters and DBSCAN need only cluster a subset of a dataset (referred to as the *core points*), labelling the remainder as outliers. Let $S_k(X)$ and $S_D(X)$ denote the silhouette coefficients of the clustering found by k-means and DBSCAN respectively, and let Z_D be the clustering found by DBSCAN. Then the *k-means-preferable* fitness function is defined to be:

$$f(X) := \begin{cases} S_k(X) - S_D(X), & \text{if } |Z_D| > 1 \\ -\infty & \text{otherwise} \end{cases} \tag{3.9}$$

There are three remarks to be made here. First, note the order of the subtraction indicates that this fitness function should be maximised. Second, while f can have a value of $-\infty$, 'valid' individuals provide values in the range $[-1,2]$ where 2 is the best, i.e. $S_D(X) = -1$ and $S_k(X) = 1$. Likewise, -1 is the worst score, occurring when $S_D(X) = 1$ and $S_k(X) = 0$. Finally, any individual that is not clustered into at least two parts by DBSCAN is penalised heavily under this fitness function when, in fact, that clustering may be of high quality. As such, this fitness function may require more nuanced adjustment.

It should also be acknowledged that k-means and DBSCAN share no common parameters, and so direct comparison is difficult. This example only uses one set of parameters, but a more thorough investigation should include a parameter sweep. The parameters being used are $k = 3$ for k-means, and $\epsilon = 0.14$ and $MinPoints = 5$ for DBSCAN. Investigating the datasets from the other examples in this study informed this parameter set. In particular, a $MinPoints$-nearest-neighbour distance plot was constructed for each dataset (as is commonly done) using the Python library Scikit-learn [283], which confirmed that an appropriate value for ϵ was just less than 0.15.

Figure 3.15 shows a summary of the progression of EDO using the k-means-preferable fitness function and the same parameters otherwise. As with the previous examples,

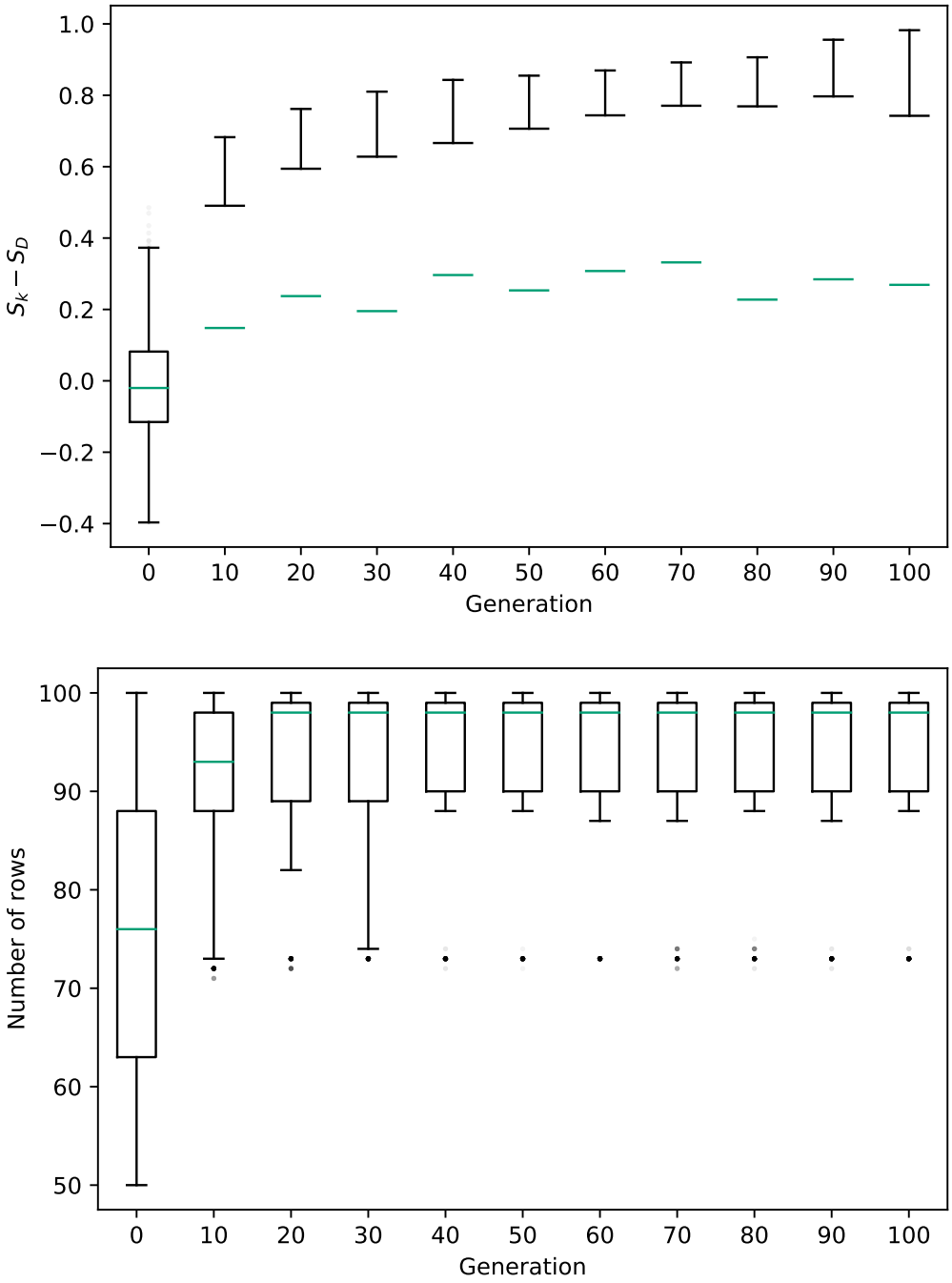
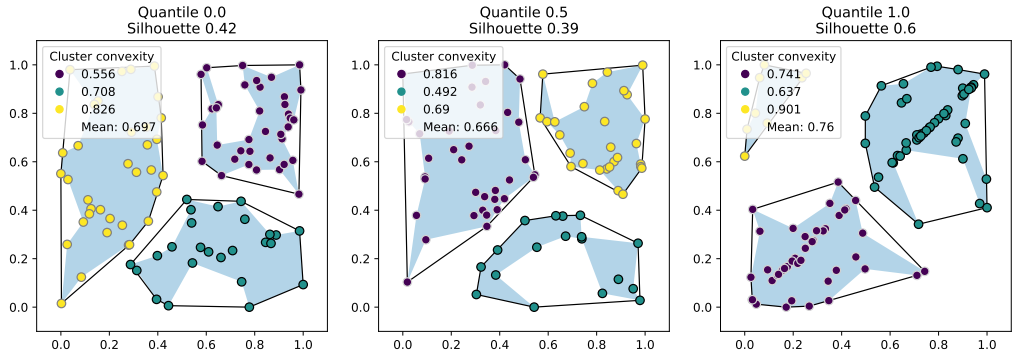
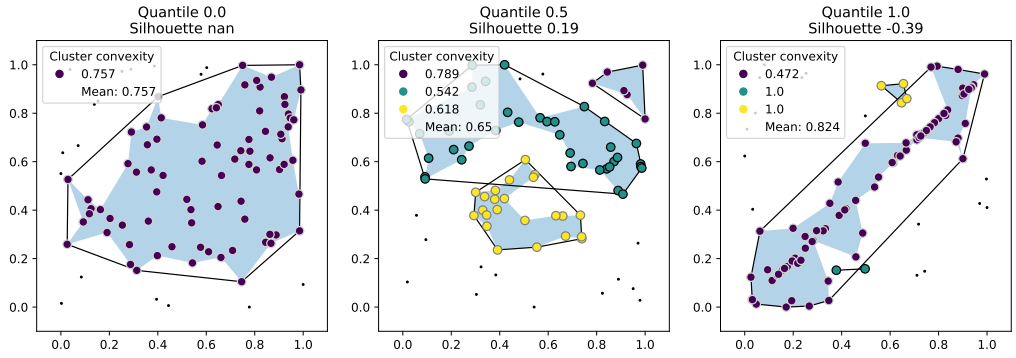


Figure 3.15: Progressions for the (k-means preferable) difference in silhouette and dimension



(a) clustered by k-means



(b) clustered by DBSCAN

Figure 3.16: Representative individuals from the k-means-preferable trials

there is a clear trend of improvement in the best individuals throughout the run. However, the variation in the population is unstable, with at least a quarter of each generation having an 'invalid' fitness score. There is also a convergence seen in the number of rows of a dataset. In this example, however, the convergence is toward the upper limit of 100 rows. Both of these observations are suggestive of a more competitive environment where slight changes to an individual can drastically alter their fitness.

Figure 3.16 exemplifies these consequences, which shows the representative individuals for this example from worst to best. Each dataset is shown with its clustering (and associated silhouette) by (a) k-means and (b) DBSCAN. In addition to a scattering of the cluster's data points, the latter figure displays the concave and convex hulls of each cluster using shading and outline, respectively.

The best-performing individual, when clustered by k-means, shows three distinct clusters, one of which is nicely separated from the others. The dimension-collapsing effect of positive correlation has come into play for the two closer clusters, although it has not dominated. In contrast, when DBSCAN clusters the same dataset, the method identifies three clusters of core points that exist within the convex hulls of one another, meaning there are overlapping clusters and, hence, a negative silhouette

coefficient.

The final observation from Figure 3.16 is that there are no truly distinct patterns in the convexity of the individuals. It is true that the best-performing individual by k-means is more convex than the others (according to the mean cluster convexity), but there is a slight dip for the median individual.

This pattern is mirrored by those clusters found by DBSCAN. Furthermore, the clusters by k-means do not show any strong, visual improvements to the convexity of each cluster. The absence of this continuous improvement to convexity may be caused by the sort of clustering found in the median individual by DBSCAN. In this case, there is a distinct overlap between the two largest clusters, but the clusters themselves are comfortably bowed. While the clustering by k-means exhibits a similar crookedness in its concave hulls, that is merely coincidental and the boundary between the clusters is more closely defined by the convex hulls. For DBSCAN, this is not the case, and highlights one of its strengths: that a cluster need not be convex to be appropriate.

To add to the discussion above, the opposing optimisation should be considered, i.e. using the same parameters to find the datasets for which DBSCAN outperforms k-means with some altered silhouette coefficient. Given the success of the k-means-preferable fitness function, a sensible starting point for the fitness function in this case would be to alter f slightly. To be specific, the objective is the reverse of f and so the fitness function should be $-f$. However, the same penalty of $-\infty$ is required for the case set out in (3.9), where DBSCAN produces only one cluster. This altered fitness function is referred to as the *DBSCAN-preferable* fitness function.

Figure 3.17 shows the same summary as above with the revised, DBSCAN-preferable fitness function, while Figure 3.18 shows the analogous individuals. Inspecting the former reveals, again, that there is some instability in the population fitness over the epochs. Also, the figure suggests that the best fitness found is worse than in the k-means-preferable case. However, this shortfall is due to the non-overlapping property of any clustering produced by k-means mentioned in Section 3.3.2. Despite that property, the k-means algorithm can readily produce results with small silhouette scores where the cluster decision boundaries are relatively close to some of the data points. Therefore, a realistic best fitness score is 1 (when $S_D(X) = 1$ and $S_k(X) = 0$) whereas the worst is -2 (when $S_D(X) = -1$ and $S_k(X) = 1$).

Upon inspection of the latter figure, it is apparent that there is a move toward less densely packed datasets — something that may be inferred from the reduced number of rows exhibited in the lower plot of Figure 3.17. Here, EDO has recognised that DBSCAN identifies and clusters only the core points of a dataset, whereas k-means

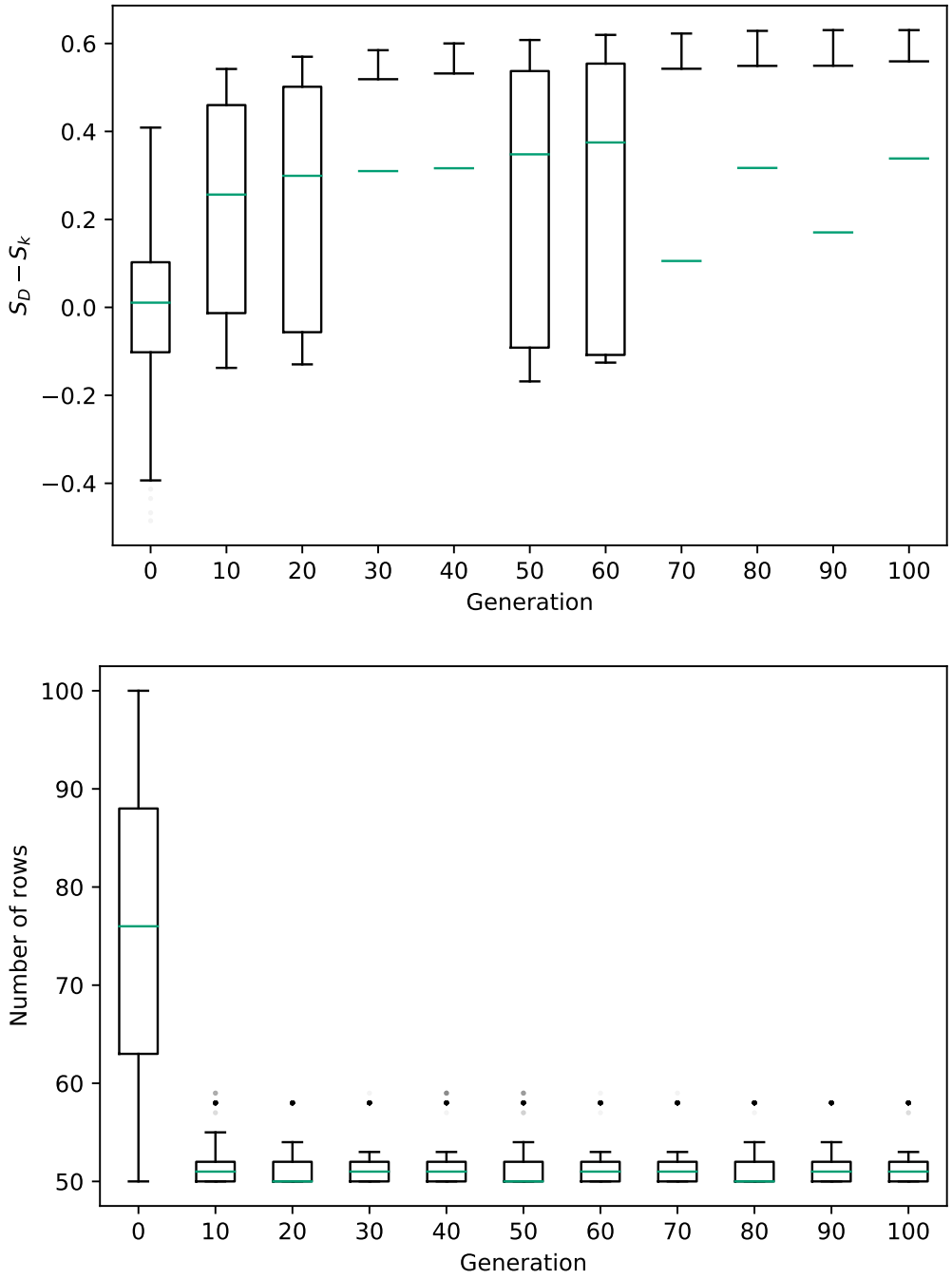
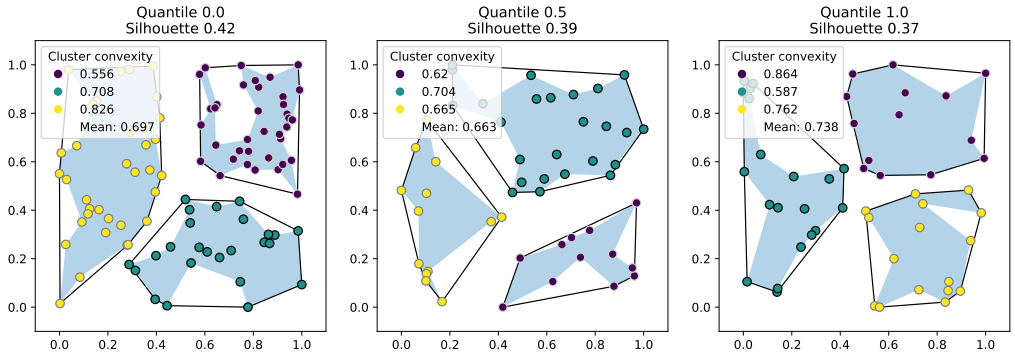
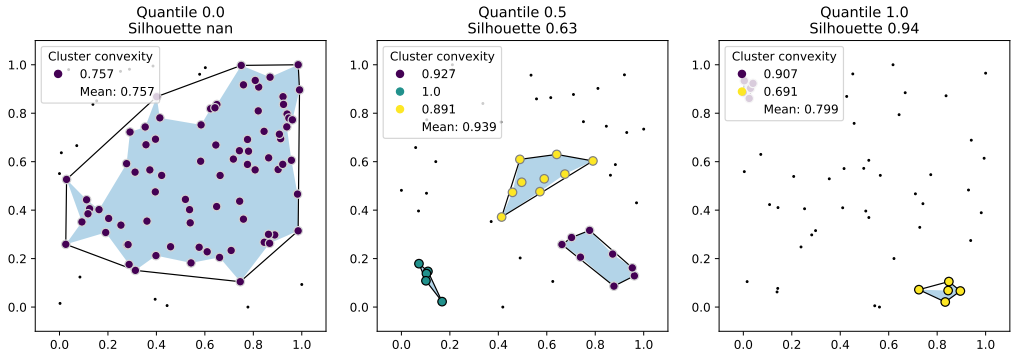


Figure 3.17: Progressions for the (DBSCAN-preferable) difference in silhouette and dimension



(a) clustered by k-means



(b) clustered by DBSCAN

Figure 3.18: Representative individuals from the DBSCAN-preferable trials

must partition the entire sample space. As a result, k-means is forced to have clusters with somewhat lower cohesion, and DBSCAN can ignore significant parts of the sample space to identify a small number of dense hotspots with impunity. Then, as fitness improves, the clusters become smaller, contain fewer points and are further apart. All of these factors serve to maximise the silhouette of the DBSCAN clustering while leaving that of k-means mostly unchanged. In general, being able to identify outliers is one of DBSCAN's core strengths. However, its success here relies on being able to ignore the majority of the data points. Hence, approaches for identifying other conditions for the success of DBSCAN should adapt the fitness function to mitigate this behaviour, perhaps with a penalty.

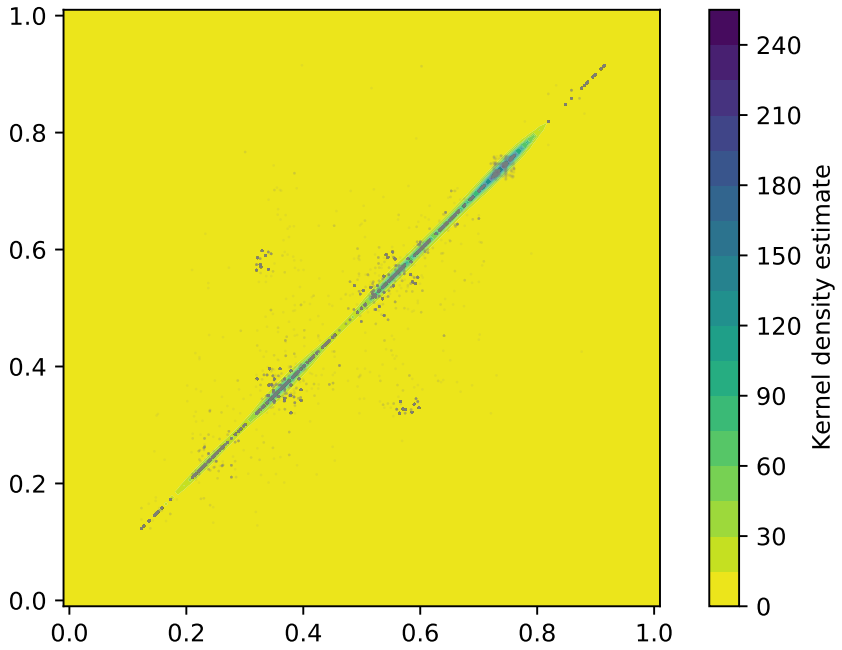
This point concludes the case study into clustering. By following a manual, iterative process of running EDO and adjusting the fitness function, several known properties of the k-means and DBSCAN algorithms have been revealed. However, to reiterate the beginning of this section: this case study acts as a tutorial on how to use EDO to study machine learning tools. A similar study could have been conducted on classification, taking an SVM as the algorithm of choice, and beginning with accuracy, say. Comparable issues may have come up such as the dimension reduction with the non-discounted fitness functions. Again, these phenomena could be handled in a similar fashion using penalties on correlation, for instance.

3.4 Chapter summary

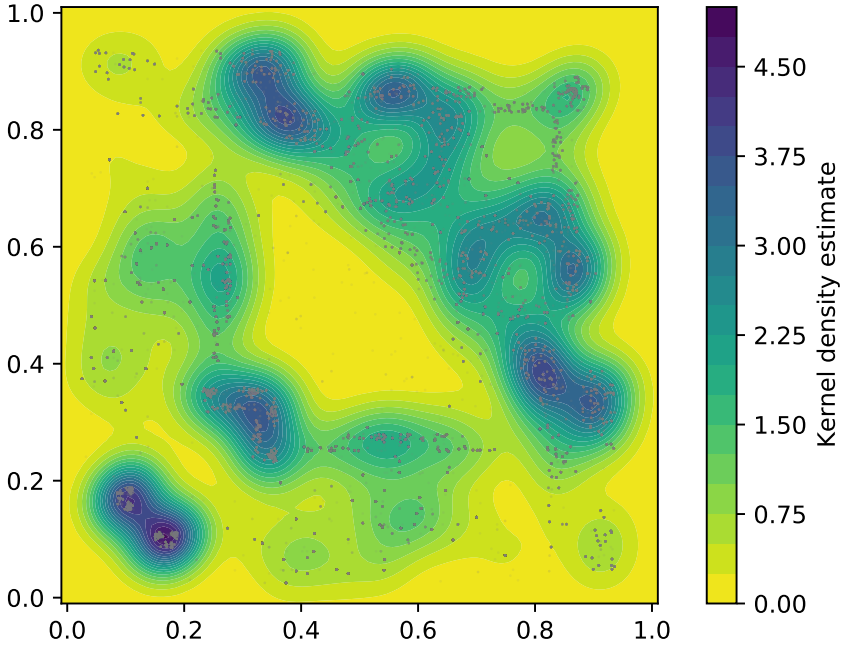
This chapter has introduced a novel approach to understanding the quality of an algorithm by exploring the space in which their well-performing datasets exist. Following a detailed explanation of its internal mechanisms, a case study in k-means clustering was offered as validation for the proposed method known as Evolutionary Dataset Optimisation. The EDO method was able to reveal some known results without prior knowledge when investigating k-means in several scenarios, and again when comparing k-means and another prominent clustering method, DBSCAN. This application of the EDO method is used again in the closing sections of Chapter 4 to compare a proposed algorithm with an established contemporary. Ultimately, it is this method that provides useful insights into the limitations of each algorithm — as well as where they are most appropriate.

The method itself is an EA and utilises biological operators to traverse a potentially broad region of the space of all possible datasets. This optimisation occurs with a minimal external framework attached, and without a need for large banks of training data. The generative nature of the proposed method also provides transparency and richness to the solution when compared to other contemporary techniques for artificial data generation as the entire history of individuals is preserved. While other search and optimisation methods exist, the decision to use an EA here was down to this transparency and the ease with which to implement biological operators that are both meaningful and understandable.

A well-known downside to EAs is that they might terminate at a local optimum — as occurred in the early examples of the case study in this chapter — and, as such, an EA may not be able to traverse the entire sample space [367] or even a sufficient part of it. This limited exploration would be even more problematic under the framework of EDO, given that the sample space is not of a fixed size or data type. By thoroughly studying the methods and results at hand, this limitation did not occur in most of the examples considered in this chapter. As an example, Figure 3.19 shows how the distribution of the parents from the examples in Section 3.3.2. In particular, each part of the figure shows density and scatter plots of all of the parent datasets. The datasets are presented without scaling to demonstrate how the EA explored the sample space. Consider the first plot (corresponding to the raw silhouette coefficient example). Here, the EDO method got stuck and failed to adequately explore the unit square as indicated by the distinct diagonal line through the square. However, by properly considering the limitations of that fitness function, the EDO method was able to explore a large proportion of the unit square, as shown in the second plot (with the discounted silhouette).



(a) with the silhouette fitness function



(b) with the discounted silhouette fitness function

Figure 3.19: Scatter and density plots of the selected parents at 10 epoch intervals from the examples in Section 3.3.2

Although this does provide evidence to say that the current design of the EA can sufficiently explore its given search space with an appropriate fitness function, it does not provide any guarantee that this will happen, even in expectation, with any fitness function.

Another weakness of EAs is their tendency to find the 'easy' way out. That is, reducing down to the most straightforward solution which solves the given problem. In most cases, that is not a problem and is often, in fact, favourable. The concentrated diagonal in Figure 3.19a shows this sort of reductive behaviour. In that particular example, the most natural solution for the EA (i.e. to maximise the silhouette coefficient with k-means) was to attempt to collapse one dimension of the search space to make the problem one-dimensional. This kind of behaviour is not necessarily a bad thing as trivial, basic and straightforward cases are of great importance when understanding an algorithm's quality.

However, should that be a problem, then the objective function could be adjusted accordingly. The case study in this chapter examined several iterations of fitness functions, but each was adjusted according to what was apparent at the time. These adjustments were possible because of the architecture of the implementation of this method. A similar strategy could be employed automatically by a more sophisticated fitness function that retains some information about the datasets generated from previous runs of EDO on a particular (or at least similar) parameter set. In this way, the currently completely unsupervised learning conducted by the EA could be ushered away from less helpful solutions (via some penalty, say) and towards previously unexplored behaviours. This automatic, iterative application of the proposed method would likely reveal more sophisticated insights into a particular algorithm.

This iterative approach in adjusting the fitness function also suggests that obtaining an optimal parameter set is less important than in some applications of EAs. For instance, the only probability distribution family used in the case study was the uniform distribution. However, in the most successful cases, the distributions of the individual datasets were not at all uniform; this is a direct result of the fitness function being so freely modifiable. By adjusting the fitness function appropriately, the EA was able to avoid structural properties in the data it was producing, thus transcending the other parameters.

The EDO method is merely a tool that demonstrates the benefit of the flipped paradigm set out in the beginning of this chapter. The concept of where 'good' datasets exist is not well-documented in literature, and this thesis established Evolutionary Dataset Optimisation as a starting point for further works to come.

Chapter 4

A game-theoretic initialisation for the k-modes algorithm

The research reported in this chapter has led to a manuscript entitled:

"A novel initialisation based on hospital-resident assignment for the k-modes algorithm"

Submitted to: *Knowledge and Information Systems*

Available online at: arXiv:2002.02701
Associated data and source code: doi:10.5281/zenodo.3639282

The abstract of the manuscript is as follows:

This paper presents a new way of selecting an initial solution for the k-modes algorithm that allows for a notion of game theoretic fairness that classic initialisations, namely those by Huang and Cao, do not. The method, which utilises the Hospital-Resident Assignment Problem to find the set of initial cluster centroids, is compared with two initialisation methods for k-modes: the original presented in [168] and the next most popular method present in the literature [64]. In order to highlight the merits of the proposed method two stages of analysis are presented. The paper concludes with an analysis of these methods against the proposed and it is demonstrated that the proposed method is able to outperform them both. The aim of this analysis is two-fold: first, to highlight the merits of the method in a familiar setting by clustering well-known benchmark datasets; and second, to provide a deeper insight into how the methods perform against one another by generating artificial datasets using the method set out in [376].

This chapter differs from the manuscript by including a more detailed description of the k-modes algorithm and its setting (in Section 4.1) and omits much of the discussion around matching games (Section 4.2) which has been expanded to form parts of Appendix A.

4.1 Introduction

The case study at the end of Chapter 3 examined the k-means algorithm — a method for clustering a set of numeric data into k parts using Euclidean distance and the cluster means. This chapter considers another algorithm in the k-means paradigm, k-modes, described in a set of seminal papers by Huang [166, 167, 168]. The k-modes algorithm makes use of similar principles to k-means, i.e. adjusting the partition toward some Voronoi tessellation according to a measure of centrality. However, the method allows for the clustering of categorical data by considering a cluster's modal values rather than its mean.

The k-modes algorithm will be used to cluster patient records based on a number of features in Chapter 5, of which some are categorical. To cluster data of mixed type in the k-means paradigm, the k-prototypes algorithm may be used. The k-prototypes algorithm was introduced in [168] with k-modes and works by separating the data into numeric and categorical attributes before performing k-means on the numeric data while performing k-modes on the categorical.

The focus of this chapter is on the initialisation of the k-modes algorithm — its initialisation is also used as the initialisation of k-prototypes. Like k-means, there is no guarantee that any two runs of k-modes will provide the same clusters; this is due to the stochastic nature of its initialisation. Therefore, the performance of the algorithm is contingent upon the quality of this initial solution [168].

This chapter introduces a novel initialisation for the k-modes algorithm that extends the method presented in [168], referred to as Huang's method. This new initialisation simulates an instance of the hospital-resident assignment problem (HR) to incorporate mathematical fairness to the initial solution and to eliminate the greedy component of Huang's method.

In addition to Huang's method, the literature revealed that the next most commonly cited initialisation was presented by Cao et al. in [64]. This initialisation (referred to as Cao's method) forms the basis of many other initialisations where a notion of density is central. These two initialisations form the set of established methods against which the proposed shall be measured.

This chapter concludes with an analysis of all three initialisations and demonstrates that the proposed method can outperform both of the established initialisations using the traditional approach with benchmark datasets. In addition to this, the method introduced in Chapter 3 provides a more in-depth examination of the methods and how they perform against one another. The first stage of this analysis serves the purpose of highlighting the merits of each method in a familiar setting, and while the second stage bolsters these observations, it provides a vigorous defence of the proposed method and exposes the scenarios in which it excels. In turn, this second analysis not only allows us to confirm the merit of the novel initialisation method as used in Chapter 5, but it also serves as a further case study highlighting the importance of the paradigm described in Chapter 3.

The chapter is structured as follows:

- Section 4.1 introduces the k-modes algorithm and its components.
- Section 4.2 provides an overview of the established initialisation methods before a statement of the proposed initialisation.
- Section 4.3 presents analyses of the initialisations on benchmark and artificial datasets.
- Section 4.4 concludes the chapter.

4.1.1 The k-modes algorithm

The following notation will be used throughout this chapter to describe the objects associated with clustering a categorical dataset:

- Let $\mathcal{A} := A_1 \times \cdots \times A_m$ denote the *attribute space*. In this chapter, only categorical attributes are considered. So, for each $j = 1, \ldots, m$, it follows that $A_j := \left\{a_1^{(j)}, \ldots, a_{d_j}^{(j)}\right\}$ where $d_j = |A_j|$ is the size of the j^{th} attribute.
- Let $\mathcal{X} := \left\{X^{(1)}, \ldots, X^{(N)}\right\} \subset \mathcal{A}$ denote a *dataset* where each $X^{(i)} \in \mathcal{X}$ is defined as an m-tuple $X^{(i)} := \left(x_1^{(i)}, \ldots, x_m^{(i)}\right)$ where $x_j^{(i)} \in A_j$ for each $j = 1, \ldots, m$. The elements of $\mathcal{X}$ are referred to as *data points* or *instances*.
- Let $\mathcal{Z} := (Z_1, \ldots, Z_k)$ be a partition of a dataset $\mathcal{X} \subset \mathcal{A}$ into $k \in \mathbb{Z}^+$ distinct, non-empty parts. Such a partition $\mathcal{Z}$ is called a *clustering* of $\mathcal{X}$.
- Each cluster Z_l has associated with it a *mode* (see Definition 4.2) which is denoted by $z^{(l)} = \left(z_1^{(l)}, \ \ldots, \ z_m^{(l)}\right) \in \mathcal{A}$. These points are also referred to as *representative points* or *centroids*. The set of all current cluster modes is denoted as $\overline{Z} = \left\{z^{(1)}, \ldots, z^{(k)}\right\}$.

As discussed in Chapter 2, the k-modes algorithm relies on a different metric to the traditional Euclidean distance since it breaks down in a categorical attribute space; if the space were ordinal, like the natural numbers, then some sense of Euclidean distance could be recovered. However, in the absence of direction or what comes between two points, this distance is not well-defined.

There are numerous measures available for the handling of categorical data. Definition 4.1 describes what is perhaps the simplest of those measures, and is the measure used in the seminal works by Huang [166, 167, 168]. The pitfall of this measure is that it democratises the attribute space. In doing so, it does not fully consider the frequency (and, thus, density) of the attributes' values, which may result in some loss of learning by the algorithm. Examples of categorical distance measures that do consider such properties of the attribute space include Ng's distance [259] and, more recently, the metric introduced in [65].

Definition 4.1. Let $\mathcal{X} \subset \mathcal{A}$ be a dataset and consider any $X^{(a)}, X^{(b)} \in \mathcal{X}$. The dissimilarity between $X^{(a)}$ and $X^{(b)}$, denoted by $d\left(X^{(a)}, X^{(b)}\right)$, is given by:

$$d\left(X^{(a)}, X^{(b)}\right) := \sum_{j=1}^{m} \delta\left(x_j^{(a)}, x_j^{(b)}\right) \quad \text{where} \quad \delta(x,y) = \begin{cases} 0, & \text{if } x = y \\ 1, & \text{otherwise.} \end{cases} \tag{4.1}$$

Theorem 4.1. *Consider a categorical attribute space, $\mathcal{A}$. Then the dissimilarity measure given in (4.1) is a metric on $\mathcal{A}$, i.e. d satisfies the following properties for all $A, B, C \in \mathcal{A}$:*

1. *Identity:* $d(A,B) = 0 \iff A = B$
2. *Positivity:* $d(A,B) \geq 0$
3. *Symmetry:* $d(A,B) = d(B,A)$
4. *The triangle inequality:* $d(A,C) \leq d(A,B) + d(B,C)$

Proof. Let $\mathcal{A}$ be a categorical attribute space and consider any $A, B, C \in \mathcal{A}$

1. Let $A, B \in \mathcal{A}$ be such that $d(A,B) = 0$. Then:

$$\begin{aligned} d(A,B) = 0 &\iff \delta\left(a_j, b_j\right) = 0, \text{ for all } j = 1, \ldots, m \\ &\iff a_j = b_j, \text{ for all } j = 1, \ldots, m \\ &\iff A = B \end{aligned}$$

2. If $A = B$, then it follows from the above that $d(A,B) = 0$. Otherwise, $\delta\left(a_j, b_j\right) = 1$ for at least one $j = 1, \ldots, m$. Therefore, $d(A,B) \geq 1 > 0$.

3. This follows immediately from the definition of δ in (4.1).

4. Let $\epsilon(A,B) = \{j : a_j \neq b_j\}$ for every $A,B \in \mathcal{A}$, i.e. let us consider the set of indices where two points differ. Then it follows that $d(A,B) = |\epsilon(A,B)|$. Hence:

$$\begin{aligned} d(A,C) = |\epsilon(A,C)| &\leq |\epsilon(A,B) \cup \epsilon(B,C)| \\ &\leq |\epsilon(A,B)| + |\epsilon(B,C)| \\ &= d(A,B) + d(B,C) \end{aligned}$$

Therefore, d satisfies the required conditions and is a metric. □

With the definition of a categorical distance metric, the notion of a representative point of a cluster can be addressed. When considering numeric data with k-means, a representative point is the mean of the points within the cluster. With categorical data, however, the mode is used as the measure for central tendency. This change follows from the concept of dissimilarity defined in (4.1). Here, the point that best represents (i.e. is closest to) those in a cluster is one with the most frequent attribute values of the points in the cluster. The following definitions formalise this notion, and Theorem 4.2, which has been adapted from [168], provides a method to find such a point.

Definition 4.2. Let $\mathcal{X} \subset \mathcal{A}$ be a dataset and consider some point $z = (z_1, \ldots, z_m) \in \mathcal{A}$. Then z is called a *mode* of $\mathcal{X}$ if it minimises the summed dissimilarity:

$$D(\mathcal{X}, z) = \sum_{i=1}^{N} d\left(X^{(i)}, z\right) \tag{4.2}$$

Definition 4.3. Let $\mathcal{X} \subset \mathcal{A}$ be a dataset. Then $n\left(a_s^{(j)}\right)$ denotes the *frequency* of the s^{th} category $a_s^{(j)}$ of A_j in $\mathcal{X}$, i.e. for each $A_j \in \mathcal{A}$ and each $s = 1, \ldots, d_j$, it follows that:

$$n\left(a_s^{(j)}\right) := \left|\left\{X^{(i)} \in \mathcal{X} : x_j^{(i)} = a_s^{(j)}\right\}\right| \tag{4.3}$$

Furthermore, $\frac{n\left(a_s^{(j)}\right)}{N}$ is called the *relative frequency* of category $a_s^{(j)}$ in $\mathcal{X}$.

Theorem 4.2. *Consider a dataset $\mathcal{X} \subset \mathcal{A}$ and some $U = (u_1, \ldots, u_m) \in \mathcal{A}$. Then $D(\mathcal{X}, U)$ is minimised if and only if $n\left(u_j\right) \geq n\left(a_s^{(j)}\right)$ for all $s = 1, \ldots, d_j$ and each $j = 1, \ldots, m$.*

A proof of this theorem can be found in the Appendix of [168].

Theorem 4.2 defines the process by which cluster modes are updated in k-modes — specifically, in Algorithm 4.3. Therefore, the final component from the k-means

paradigm to be configured is the objective (cost) function. This function is defined in Definition 4.4, and following that a practical statement of the k-modes algorithm is given in Algorithm 4.1 as set out in [168].

Definition 4.4. Let $\mathcal{Z} = \{Z_1, \ldots, Z_k\}$ be a clustering of a dataset $\mathcal{X}$, and let $\overline{Z} = \left\{z^{(1)}, \ldots, z^{(k)}\right\}$ be the corresponding cluster modes. Then $W = (w_{i,l})$ is an $N \times k$ *partition matrix* of $\mathcal{X}$ such that:

$$w_{i,l} = \begin{cases} 1, & \text{if } X^{(i)} \in Z_l \\ 0, & \text{otherwise.} \end{cases}$$

With this, the *cost function* is defined to be the summed within-cluster dissimilarity:

$$C\left(W, \overline{Z}\right) := \sum_{l=1}^{k} \sum_{i=1}^{N} \sum_{j=1}^{m} w_{i,l}\, \delta\left(x_j^{(i)}, z_j^{(l)}\right) \tag{4.4}$$

Algorithm 4.1: The k-modes algorithm

Input: a dataset $\mathcal{X}$, a number of clusters to form k
Output: a clustering $\mathcal{Z}$ of $\mathcal{X}$

Select k initial modes $z^{(1)}, \ldots, z^{(k)} \in \mathcal{X}$
$\overline{Z} \leftarrow \left\{z^{(1)}, \ldots, z^{(k)}\right\}$
$\mathcal{Z} \leftarrow \left(\left\{z^{(1)}\right\}, \ldots, \left\{z^{(k)}\right\}\right)$
for $X^{(i)} \in \mathcal{X}$ **do**
    $Z_{l^*} \leftarrow \textsc{SelectClosest}\left(X^{(i)}\right)$
    $Z_{l^*} \leftarrow Z_{l^*} \cup \left\{X^{(i)}\right\}$
    $\textsc{Update}\left(z^{(l^*)}\right)$
end
repeat
    for $X^{(i)} \in X$ **do**
        Let Z_l be the cluster $X^{(i)}$ currently belongs to
        $Z_{l^*} \leftarrow \textsc{SelectClosest}\left(X^{(i)}\right)$
        if $l \neq l^*$ **then**
            $Z_l \leftarrow Z_l \setminus \left\{X^{(i)}\right\}$ and $Z_{l^*} \leftarrow Z_{l^*} \cup \left\{X^{(i)}\right\}$
            $\textsc{Update}\left(z^{(l)}\right)$ and $\textsc{Update}\left(z^{(l^*)}\right)$
        end
    end
until *No point changes cluster*

Algorithm 4.2: SELECTCLOSEST

Input: a data point $X^{(i)}$, a set of current clusters $\mathcal{Z}$ and their modes $\overline{Z}$
Output: the cluster whose mode is closest to the data point Z_{l^*}

1 Select $z^{l^*} \in \overline{Z}$ that minimises: $d\left(X^{(i)}, z_{l^*}\right)$
2 Find their associated cluster Z_{l^*}

Algorithm 4.3: UPDATE

Input: an attribute space $\mathcal{A}$, a mode to update $z^{(l)}$ and its cluster Z_l
Output: an updated mode

1 Find $z \in \mathcal{A}$ that minimises $D(Z_l, z)$
2 $z^{(l)} \leftarrow z$

4.2 Initialisation processes

The k-modes algorithm is stochastic, in that it is dependent on a stochastic choice of starting point. The process of finding an initial solution or set of modes for the algorithm is called the initialisation. As with many algorithms in the centroid-based paradigm, the standard initialisation for k-modes is to randomly sample k distinct points in the dataset, call them the initial modes and assign each data point to its closest mode.

The initial set of modes must be instances in the dataset to ensure that there are no empty clusters in the first iteration of the algorithm. However, this set of points need not be determined entirely at random. The remainder of this section describes two well-established initialisations for k-modes that aim to lever the structure of the data at hand preemptively to select these initial modes. The section then closes with the motivation and definition of the proposed initialisation method.

4.2.1 Huang's method

Among the original works by Huang, an alternative initialisation method was presented that selects modes by distributing frequently occurring values from the attribute space among k potential modes [168]. The process (referred to as Huang's method) is described in full in Algorithm 4.4. Huang's method considers a set of potential modes, $\widehat{Z} \subset \mathcal{A}$, that is then replaced by a set of real initial modes, $\overline{Z} \subset \mathcal{X}$.

The process of selecting this set of potential modes is ambiguous in the original paper — as is alluded to in [184]. In [168], the author states that the potential modes should be created by assigning to them the "most frequent categories equally". Here, as is done in practical implementations of k-modes, this assignment is interpreted

as a weighted random sample. Algorithm 4.5 contains a description of the sampling process using the relative frequencies of the values (categories) in the attribute space.

Algorithm 4.4: Huang's method

Input: a dataset $\mathcal{X} \subset \mathcal{A}$, a number of modes to find k
Output: a set of k initial modes $\overline{Z}$

$\overline{Z} \leftarrow \emptyset$
$\widehat{Z} \leftarrow$ SamplePotentialModes $(\mathcal{X})$
for $\hat{z} \in \widehat{Z}$ **do**
    Select $X^{(i^*)} \in \mathcal{X} \setminus \overline{Z}$ that minimises $d\left(X^{(i)}, \hat{z}\right)$
    $\overline{Z} \leftarrow \overline{Z} \cup \left\{X^{(i^*)}\right\}$
end

Algorithm 4.5: SamplePotentialModes

Input: a dataset $\mathcal{X} \subset \mathcal{A}$, a number of modes to find k
Output: a set of k potential modes $\widehat{Z}$

$\widehat{Z} \leftarrow \emptyset$
for $j = 1, \ldots, m$ **do**
    for $s = 1, \ldots, d_j$ **do**
        Calculate $\frac{n\left(a_s^{(j)}\right)}{N}$
    end
end
while $\left|\widehat{Z}\right| < k$ **do**
    Create an empty m-tuple $\hat{z}^{(l)}$
    for $j = 1, \ldots, m$ **do**
        Sample $a_{s^*}^{(j)}$ from A_j with respect to the relative frequencies of A_j
        $\hat{z}_j^{(l)} \leftarrow a_{s^*}^{(j)}$
    end
    $\widehat{Z} \leftarrow \widehat{Z} \cup \left\{\hat{z}^{(l)}\right\}$
end

4.2.2 Cao's method

The second initialisation considered in this chapter is known as Cao's method [64]. This method differs significantly from Huang's method, and has formed the basis of several other initialisations. In essence, the method selects initial modes according to their density in the dataset whilst forcing dissimilarity between them. Definition 4.5 formalises the concept of categorical density. The definition also considers the re-

lationship between density and relative frequency when using the measure defined in (4.1).

Further to its alternative density-based approach, the method (which is described in Algorithm 4.6) is deterministic. Therefore, when using k-modes with Cao's initialisation, only one run of the algorithm is required to identify a clustering, making it an attractive option if computational time is critical.

Definition 4.5. Consider a dataset $\mathcal{X} \subset \mathcal{A} = \{A_1, \ldots, A_m\}$. Then [64] defines the *average density* of any point $X^{(i)} \in \mathcal{X}$ with respect to $\mathcal{A}$ as:

$$\text{Dens}\left(X^{(i)}\right) = \frac{\sum_{j=1}^{m} \text{Dens}_j\left(X^{(i)}\right)}{m} \tag{4.5}$$

where

$$\text{Dens}_j\left(X^{(i)}\right) = \frac{\left|\left\{X^{(t)} \in \mathcal{X} : x_j^{(i)} = x_j^{(t)}\right\}\right|}{N} = \frac{n\left(x_j^{(i)}\right)}{N} \tag{4.6}$$

That is, $\text{Dens}_j\left(X^{(i)}\right)$ is the relative frequency of $x_j^{(i)}$ in $\mathcal{X}$. Also, observe that:

$$\left|\left\{X^{(t)} \in \mathcal{X} : x_j^{(i)} = x_j^{(t)}\right\}\right| = \sum_{t=1}^{N}\left(1 - \delta\left(x_j^{(i)}, x_j^{(t)}\right)\right) \tag{4.7}$$

Hence, an alternative definition for (4.5) can be derived:

$$\text{Dens}\left(X^{(i)}\right) = \frac{1}{mN}\sum_{j=1}^{m}\sum_{t=1}^{N}\left(1 - \delta\left(x_j^{(i)}, x_j^{(t)}\right)\right) = 1 - \frac{1}{mN} D\left(\mathcal{X}, X^{(i)}\right) \tag{4.8}$$

This alternative definition highlights how identifying high-density data points would mean finding data points with a low summed dissimilarity, i.e. true modes of the dataset.

4.2.3 The proposed method

Both of the initialisation methods described in this section have a greedy component. Cao's method essentially chooses the point with highest density that has not already been chosen whilst forcing some separation between the set of initial modes. In the case of Huang's, the greediness only comes toward the end of the method. When the set of potential modes is identified, it is replaced by a set of instances in the dataset. Specifically, this means that in any practical implementation of this method, the order in which a set of potential modes is iterated over can affect the set of initial

Algorithm 4.6: Cao's method

Input: a dataset $\mathcal{X}$, a number of modes to find k
Output: a set of k initial modes $\overline{Z}$

$\overline{Z} \leftarrow \emptyset$
for $X^{(i)} \in \mathcal{X}$ **do**
    Calculate Dens $\left(X^{(i)}\right)$
end
Select $1 \leq i_1 \leq N$ which maximises Dens $\left(X^{(i)}\right)$
$\overline{Z} \leftarrow \overline{Z} \cup \left\{X^{(i_1)}\right\}$
while $\left|\overline{Z}\right| < k$ **do**
    Select $X^{(i^*)} \notin \overline{Z}$ which maximises $\min_{z^{(l)} \in \overline{Z}} \left\{\text{Dens}\left(X^{(i)}\right) \times d\left(X^i, z^{(l)}\right)\right\}$
    $\overline{Z} \leftarrow \overline{Z} \cup \left\{X^{(i^*)}\right\}$
end

modes. Thus, there is no guarantee of consistency. The same is true for any arbitrary tie breaks that may occur.

The initialisation proposed in this chapter extends Huang's method to be order-invariant in the final allocation. Doing so eliminates its greedy component and provides a more intuitive starting point for the k-modes algorithm by constructing and solving a matching game between the set of potential modes and some subset of the data.

Matching games are a construct from game theory that allow for the allocation of partnerships between agents (players) such that no pair of players is rationally envious of any partnership. Appendix A provides an extensive introduction to several fundamental matching games, the most commonly found of which is the stable marriage problem (SM). SM, like many matching games, involves two distinct sets of players (parties), each of whom have a strict ranking of the players in the other set. Resolving these player preferences to find a matching in which no two players would rather be matched to one another than their current matches is called *solving* the game. The authors of [130] formalised this problem and provided an algorithm that guarantees to find an stable, party-optimal matching to any instance of SM. This algorithm is given in Appendix A, and algorithms following a similar structure are commonly referred to as Gale-Shapley algorithms.

Despite its simplicity, using SM in this scenario would reduce the proposed method to Huang's method. This can be seen as follows. Both parties in the game must be of the same size, and so k data points must be considered along with the potential modes. The most straightforward way of selecting these points is to allow each po-

tential mode to add its closest point that has not already been selected. Then, upon solving the game, and up to an arbitrary breaking of any ties in the preference lists, each potential mode is assigned to its chosen data point.

A popular generalisation of SM is HR, a matching game that addresses the practical problem from which it gets its name: assigning medical students to hospital placements. A variant of this game is still used by the National Resident Matching Program (NRMP) in the United States. A thorough description of HR, and the algorithms to solve instances of it, is provided in Appendix A. However, a brief overview is provided here.

Consider two distinct sets of players, R and H, and call them the residents and hospitals, respectively. In HR, every resident $r \in R$ provides a strict ranking of some nonempty subset of the hospitals, denoted $f(r)$. Then, each hospital $h \in H$ provides a preference list, $g(h)$, strictly ranking all and only those residents that have ranked it, i.e. $g(h)$ is some permutation of the set $\{r \in R \mid h \in f(r)\}$. In addition to these preference lists, HR stipulates that every hospital $h \in H$ has a capacity, $c_h \in \mathbb{N}$, associated with them. This capacity is a strict upper limit on the number of residents that a hospital may be matched to at once. A general assumption, although not necessary, is that there is sufficient capacity across the hospitals for all of the residents to be assigned, i.e. that $\sum_{h \in H} c_h \geq |R|$. This construction of residents, hospitals, preference lists and capacities form a *game*, denoted by (R, H).

The objective of the game is to identify a many-to-one mapping (referred to as a *matching*), M, between the residents and the hospitals. In particular, the aim is to find a matching such that no resident-hospital pair could be, and would rather be, matched to one another than their current match(es). Such a matching is considered *stable*. Finding a resident- or hospital-optimal, stable matching to an instance of HR is called *solving* the game. In addition to SM, the authors formalised HR in [130], and presented a Gale-Shapley algorithm that guaranteed such a solution to any instance of the game. Subsequent studies [106, 308] have improved on the original algorithm by leveraging structures within the game, such as the strict nature of the preference lists.

Returning to the problem at hand, where the k potential modes must be allocated to a nearby and unique data point, it can be seen that HR mirrors the constraints well. In particular, this scenario can be modelled by an instance of HR where each hospital has a capacity of one. A contextual summary of the remaining HR game components and those of the proposed initialisation is given in Table 4.1.

A limitation of using HR is the common occurrence of ties when using the distance measure defined in (4.1). A tie in the distance between points corresponds to a tie

Object in k-modes initialisation	Object in a matching game
Potential modes	The set of residents
Data points closest to potential modes	The set of hospitals
Similarity between a potential mode and a point	Position in preference lists
The data point to replace a potential mode	A pair in a matching

Table 4.1: Links between the initialisation and the components of a game

in the preference lists. As is discussed in Appendix A, extensions to HR do exist for handling ties in preferences that would alleviate the reliance on having to break ties. However, the notion of stability becomes tiered under such extensions, and the guarantee of a stable matching does not exist. Therefore, they are not considered here. Instead, the proposed method employs the standard form of HR and aims to mitigate the effect of ties by considering a potentially large number of candidate points (up to k^2). Algorithm 4.7 provides a formal statement of the proposed method.

Algorithm 4.7: The proposed initialisation method

Input: a dataset $\mathcal{X} \subset \mathcal{A}$, a number of modes to find k
Output: a set of k initial modes $\overline{Z}$

$\overline{Z} \leftarrow \emptyset$
$H \leftarrow \emptyset$
$R \leftarrow$ SamplePotentialModes$(\mathcal{X})$
for $r \in R$ **do**
    Find the set of k data points $H_r \subset \mathcal{X}$ that are the least dissimilar to r
    Arrange H_r into descending order of similarity with respect to r, denoted by H_r^*
    $H \leftarrow H \cup H_r$
    $f(r) \leftarrow H_r^*$
end
for $h \in H$ **do**
    $c_h \leftarrow 1$
    Sort R into descending order of similarity with respect to h, denoted by R^*
    $g(h) \leftarrow R^*$
end
Solve the matching game defined by (R, H) to obtain a matching M
for $r \in R$ **do**
    $\overline{Z} \leftarrow \overline{Z} \cup \{M(r)\}$
end

	Breast cancer	Mushroom	Nursery	Soybean
N	699	8124	12960	307
m	10	22	8	35
No. classes	2	2	5	19
Missing values	True	True	False	True
Adjusted N	683	5644	12960	266
Adjusted no. classes	2	2	5	15
No. clusters found	8	17	23	8

Table 4.2: A summary of the benchmark datasets

4.3 Experimental results

This section provides several analyses of the initialisation processes presented in this chapter. Each analysis herein requires the use of software, and specifically an implementation of the k-modes algorithm. The implementation used, called `kmodes`, is available on the Python Package Index and its source code is hosted on GitHub [92]. Huang's and Cao's methods are implemented already in `kmodes`, and the proposed matching initialisation has been included in a separate fork of the GitHub repository, available at `doi:10.5281/zenodo.3713170`. This version of the k-modes implementation makes use of a Python library, `matching`, the details of which can be found in Appendix A. The version of the library used for these analyses is available at `doi:10.5281/zenodo.2711847`. Furthermore, all of the source code used in this section is available at `doi:10.5281/zenodo.3639282`.

To give comparative results on the quality of the initialisation processes considered in this chapter, four well-known, categorical, labelled datasets — breast cancer, mushroom, nursery, and soybean (large) — will be clustered by the k-modes algorithm with each of the initialisation processes. These datasets have been chosen to fall in line with the established literature, and particularly the works in which Huang's and Cao's methods were introduced. Further, the datasets exhibit a range of sizes, complexities and applications. Each dataset is openly available under the University of California Irvine (UCI) Machine Learning Repository [104], and their characteristics are summarised in Table 4.2. This analysis excludes all incomplete instances (i.e. where data is missing), and the remaining dataset characteristics are reported as 'adjusted'.

This analysis does not consider evaluative metrics related to classification such as ac-

curacy, recall or precision — although this is commonly done. A selection of articles from the last 20 or so years that employ this approach include [19, 64, 65, 168, 259, 268, 320, 325]. Instead, only internal measures are considered, such as the cost function defined in (4.4). Like inertia with k-means, this metric is label-invariant, and its values are comparable across the different initialisation methods on the same dataset. Furthermore, the cost function captures the effect of each initialisation method on the initial and final clustering of each dataset.

An additional, and often useful, metric for clustering is the silhouette coefficient. In Chapter 3, the silhouette served as an extension to the cost function to ensure intra-cluster cohesion and inter-cluster separation. Therefore, it would be sensible to believe that it could provide insight into the effect of each initialisation method to the k-modes algorithm. However, this metric loses its intuition under the distance measure employed here. As such, it has been omitted. The remaining performance measures used are the number of iterations for the k-modes algorithm to terminate after the initialisation and the time to terminate in seconds.

The final piece of information required in this analysis is a choice for k for each dataset. An immediate choice is the number of classes that are present in a dataset, but this is not necessarily an appropriate choice since the classes may not be representative of true clusters [244]. However, this analysis will consider this case as there may be practical reasons to limit the value of k. The other strategy for choosing k considered in this chapter uses the knee point detection algorithm introduced in [319]. This strategy was chosen over other popular methods such as the 'elbow' method as its results are definitive.

The knee point detection algorithm was employed using values of k from 2 up to $\lfloor\sqrt{N}\rfloor$ for each dataset. The number of clusters determined by this strategy is reported in the final column of Table 4.2.

4.3.1 Using the knee point detection algorithm for k

Table 4.3 summarises the results of each initialisation method on the benchmark datasets. In each case, the number of clusters was determined using the knee point detection algorithm. Each column shows the mean value of each metric and its standard deviation across 250 repetitions of the k-modes algorithm.

By examining these tables, it would seem that the proposed method and Huang's method are fairly comparable across the board. On the contrary, the proposed method is faster despite taking more iterations in general which suggests a more intuitive initialisation. More importantly, though, it appears that Cao's method performs the best out of the three initialisation methods. In terms of initial and final

	Initial cost	Final cost	No. iterations	Time
Cao	3118.00 (0.000)	2774.00 (0.000)	4.00 (0.000)	0.30 (0.012)
Huang	2856.50 (104.245)	2748.83 (64.514)	2.68 (0.817)	0.22 (0.046)
Matching	2870.11 (101.869)	2752.59 (52.387)	2.72 (0.760)	0.16 (0.021)

(a) the breast cancer dataset with $k = 8$

	Initial cost	Final cost	No. iterations	Time
Cao	20381.00 (0.000)	20376.00 (0.000)	2.00 (0.000)	4.68 (0.205)
Huang	23027.24 (1209.753)	21869.06 (747.766)	2.90 (0.934)	5.11 (1.138)
Matching	23279.36 (1498.324)	21855.50 (751.641)	3.02 (0.936)	2.77 (0.325)

(b) the mushroom dataset with $k = 17$

	Initial cost	Final cost	No. iterations	Time
Cao	35544.00 (0.000)	35544.00 (0.000)	1.00 (0.000)	4.98 (0.152)
Huang	37535.06 (372.596)	37535.06 (372.596)	1.00 (0.000)	3.58 (0.121)
Matching	37484.29 (327.467)	37484.29 (327.467)	1.00 (0.000)	3.14 (0.141)

(c) the nursery dataset with $k = 23$

	Initial cost	Final cost	No. iterations	Time
Cao	1654.00 (0.000)	1585.00 (0.000)	4.00 (0.000)	0.28 (0.014)
Huang	1829.31 (92.308)	1708.55 (69.740)	3.58 (1.019)	0.28 (0.063)
Matching	1827.76 (86.852)	1711.49 (73.319)	3.42 (0.963)	0.17 (0.022)

(d) the soybean dataset with $k = 8$

Table 4.3: Metric results when using the knee point detection algorithm

costs Cao's method improves by roughly 10% against the next best method for the three datasets with which it succeeds. Also, while the number of iterations is comparable, the computation time is substantially less than the other two methods. This last observation is expected given it is a deterministic method and need only be run once to achieve this performance.

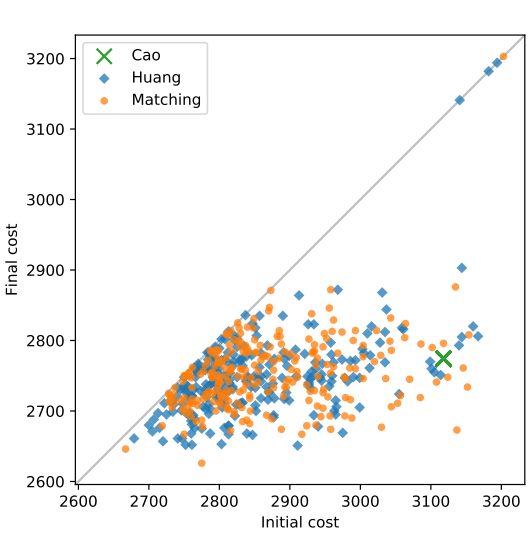
As useful as these metrics are, in the centroid-based clustering paradigm, a particular clustering is selected based on it having the minimal final cost over several runs of the algorithm — not the mean. While Cao's method is reliable in this regard, in that there is no variation at all, it does not always produce the best clustering possible. There is a trade-off to be made between computational time and performance here.

In order to gain more insight into the performance of each method, a less granular form of analysis is required. Figures 4.1—4.4 display the cost function results for each dataset in the form of a scatter plot and two empirical cumulative density function (CDF) plots, highlighting the breadth and depth of the behaviours exhibited by each initialisation method.

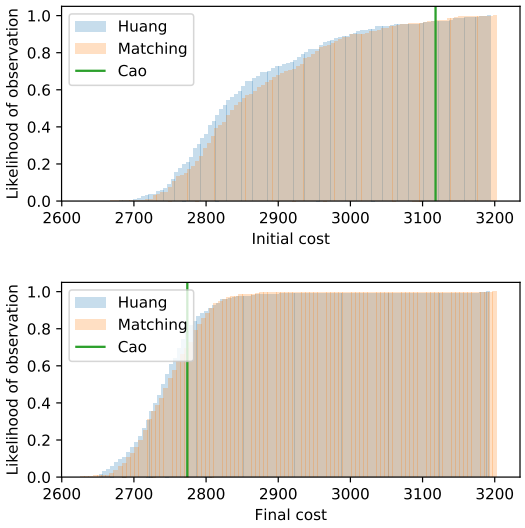
Looking at Figure 4.1, it is clear that in terms of the final cost, Cao's method is middling when compared to the other methods. This so-so performance is apparent from Table 4.3a and, indeed, Huang's and the proposed method appear indistinguishable when looking at the body of the results. However, since the criterion for the best clustering (in practical terms) is having the minimal final cost, the proposed method is superior; that the method produces clusterings with a broader range of costs (indicated by the trailing right-hand side of each CDF plot) is irrelevant for the same reason.

This pattern of broadly similar behaviour between Huang's and the proposed method is apparent in each of the figures here, and in all cases, the proposed method outperforms Huang's. In fact, for all cases except for the nursery dataset, the proposed method achieves the lowest final cost of all the methods. As such, it performs the best in practical terms on these particular datasets.

In the case of the nursery dataset, Cao's method is unquestionably the best performing initialisation method. There are two things of note here: first, not one of the methods was able to find an initial clustering that k-modes could improve upon; and, second, the nursery dataset exactly describes the entire attribute space in which it exists. This property could be why the other methods fall behind Cao's method so decisively. In such a scenario, Cao's method can definitively choose the k most dense-whilst-separated points from that space as the initial cluster centres. Meanwhile, the two remaining methods are, in essence, randomly sampling from this space. That each initial solution in these repetitions is locally optimal remains a mystery.

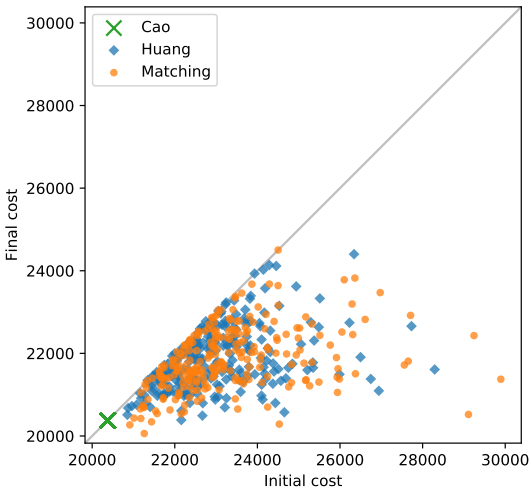


(a) Scatter plot of initial and final costs.

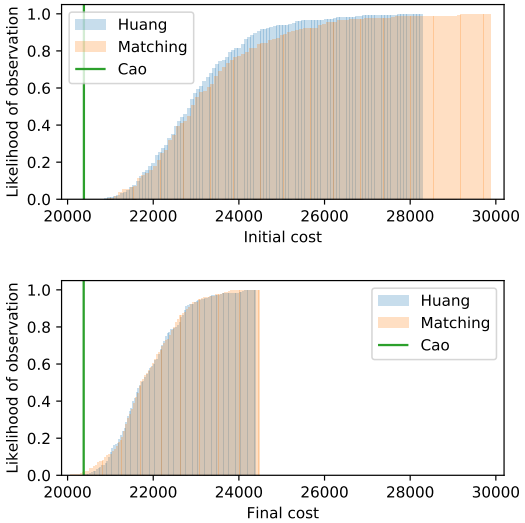


(b) Empirical CDF plots for initial (top) and final (bottom) costs.

Figure 4.1: Summary plots for the breast cancer dataset with $k = 8$

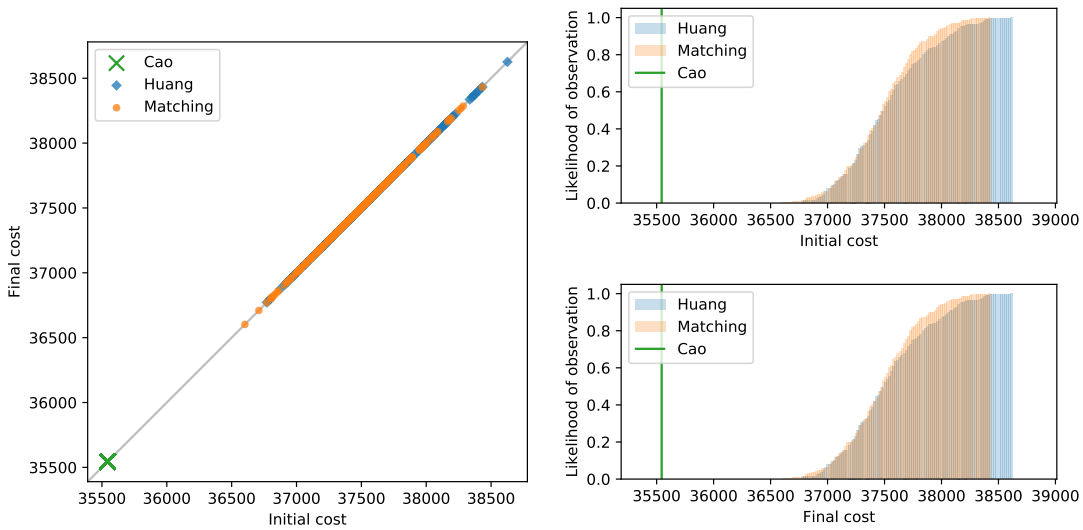


(a) Scatter plot of initial and final costs.



(b) Empirical CDF plots for initial (top) and final (bottom) costs.

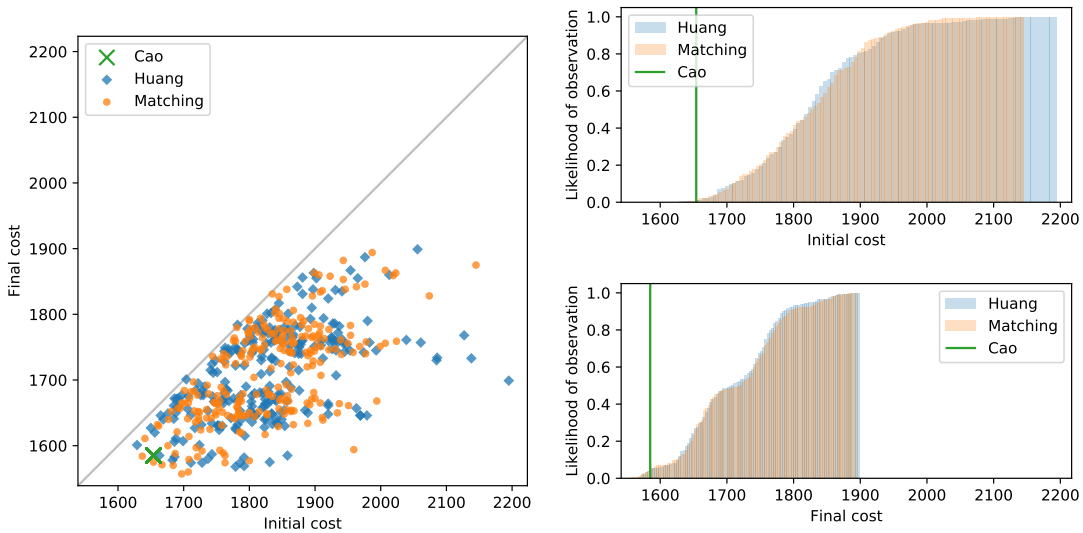
Figure 4.2: Summary plots for the mushroom dataset with $k = 17$



(a) Scatter plot of initial and final costs

(b) Empirical CDF plots for initial (top) and final (bottom) costs

Figure 4.3: Summary plots for the nursery dataset with $k = 23$



(a) Scatter plot of initial and final costs

(b) Empirical CDF plots for initial (top) and final (bottom) costs

Figure 4.4: Summary plots for the soybean dataset with $k = 8$

4.3.2 Using the number of classes for k

As is discussed above, the often automatic choice for k is the number of classes present in the data. This subsection repeats the analysis from the previous subsection but with this traditional choice for k. Table 4.4 contains the analogous summaries of each initialisation method's performance on the benchmark datasets over the same number of repetitions.

An immediate comparison to the previous tables is that the mean costs are significantly higher, and the computation times are shorter, for all datasets bar the soybean dataset. These effects come directly from the choice of k in that higher values of k will require more checks (and thus computational time) but will typically lead to more homogeneous clusters, reducing their within-cluster dissimilarity and therefore cost.

Looking at these tables on their own reveals that Cao's method is the superior initialisation method on average: the means are substantially lower in terms of initial and final cost; there is no deviation in these results; again, the total computational time is a fraction of the other two methods. It is apparent again that Huang's and the proposed method are comparable on average. Then, as before, a more nuanced investigation will require finer visualisations. Figures 4.5—4.8 show the analogous plots as in the previous subsection except with this traditional choice for k.

Figures 4.5 and 4.6 indicate that a particular behaviour emerged during the runs of the k-modes algorithm. Specifically, each solution falls into one of (predominantly) two types: effectively no improvement on the initial clustering, or terminating at some clustering with a cost that is bounded below across all such solutions. Invariably, Cao's method achieves this lower bound, and unless Cao's method is used, these particular choices for k mean that the performance of the k-modes algorithm is highly susceptible to its initial clustering. Moreover, the other two methods are effectively indistinguishable in these cases, and so if a robust solution is required, Cao's method is the only viable option.

Figure 4.7 corresponds to the nursery dataset results with $k = 5$. In this set of runs, the same pattern emerges as in Figure 4.3 where sampling the initial centres from amongst the densest points (via Huang's method and the proposed) is an inferior strategy to one considering the entire attribute space such as with Cao's method. Again, no method can improve on the initial solution (except for one repetition with the matching initialisation method) although the disparity between Cao's method and the others is significantly less.

The primary conclusion from this analysis of benchmark datasets is that while Huang's

	Initial cost	Final cost	No. iterations	Time
Cao	3315.00 (0.000)	3172.00 (0.000)	2.00 (0.000)	0.13 (0.005)
Huang	3393.80 (120.772)	3348.51 (144.849)	1.54 (0.653)	0.10 (0.024)
Matching	3406.73 (111.686)	3355.56 (144.621)	1.61 (0.638)	0.09 (0.018)

(a) the breast cancer dataset with $k = 2$

	Initial cost	Final cost	No. iterations	Time
Cao	37662.00 (0.000)	37662.00 (0.000)	1.00 (0.000)	0.94 (0.035)
Huang	41974.07 (2393.889)	39226.25 (2483.933)	3.11 (1.430)	1.92 (0.679)
Matching	42175.54 (2520.163)	39617.53 (2637.574)	3.03 (1.439)	1.38 (0.491)

(b) the mushroom dataset with $k = 2$

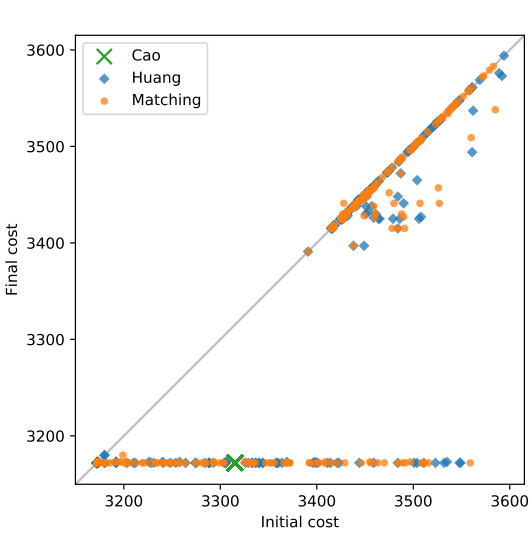
	Initial cost	Final cost	No. iterations	Time
Cao	49060.00 (0.000)	49060.00 (0.000)	1.00 (0.000)	1.80 (0.090)
Huang	51229.45 (902.503)	51229.45 (902.503)	1.00 (0.000)	1.72 (0.116)
Matching	51107.52 (910.258)	51101.95 (903.525)	1.00 (0.063)	1.37 (0.128)

(c) the nursery dataset with $k = 5$

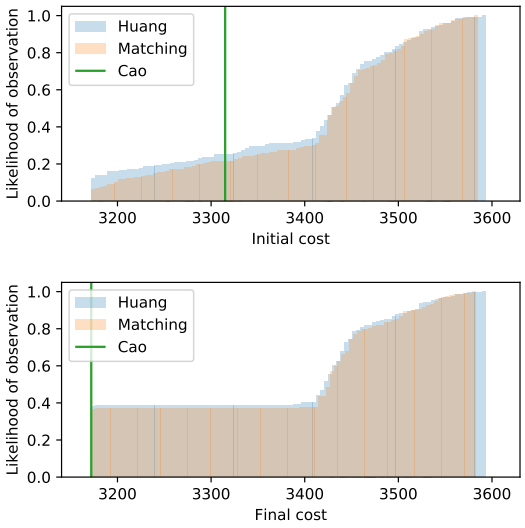
	Initial cost	Final cost	No. iterations	Time
Cao	1364.00 (0.000)	1314.00 (0.000)	2.00 (0.000)	0.33 (0.009)
Huang	1588.89 (83.682)	1446.22 (59.844)	4.02 (1.081)	0.45 (0.085)
Matching	1582.56 (87.418)	1447.08 (60.154)	4.01 (1.128)	0.24 (0.025)

(d) the soybean dataset with $k = 15$

Table 4.4: Metric results when using the number of classes

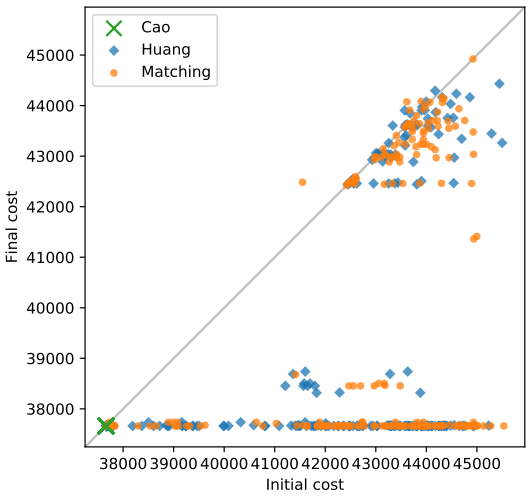


(a) Scatter plot of initial and final costs.

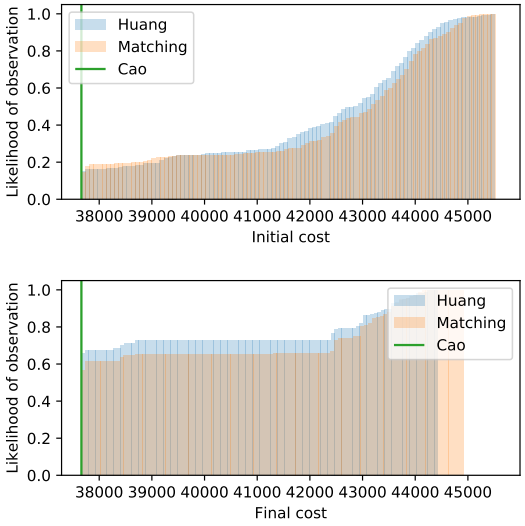


(b) Empirical CDF plots for initial (top) and final (bottom) costs

Figure 4.5: Summary plots for the breast cancer dataset with $k = 2$

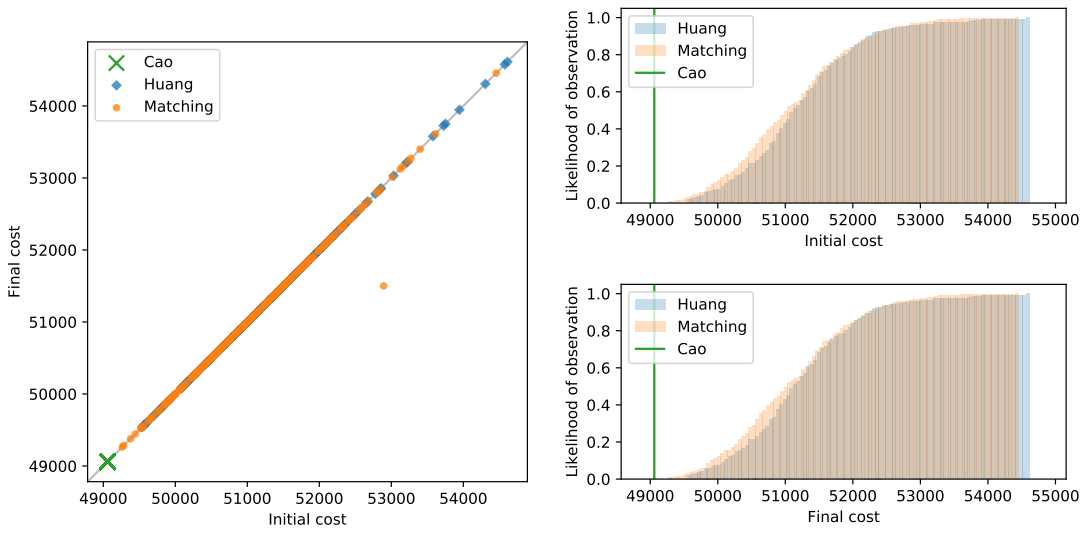


(a) Scatter plot of initial and final costs



(b) Empirical CDF plots for initial (top) and final (bottom) costs

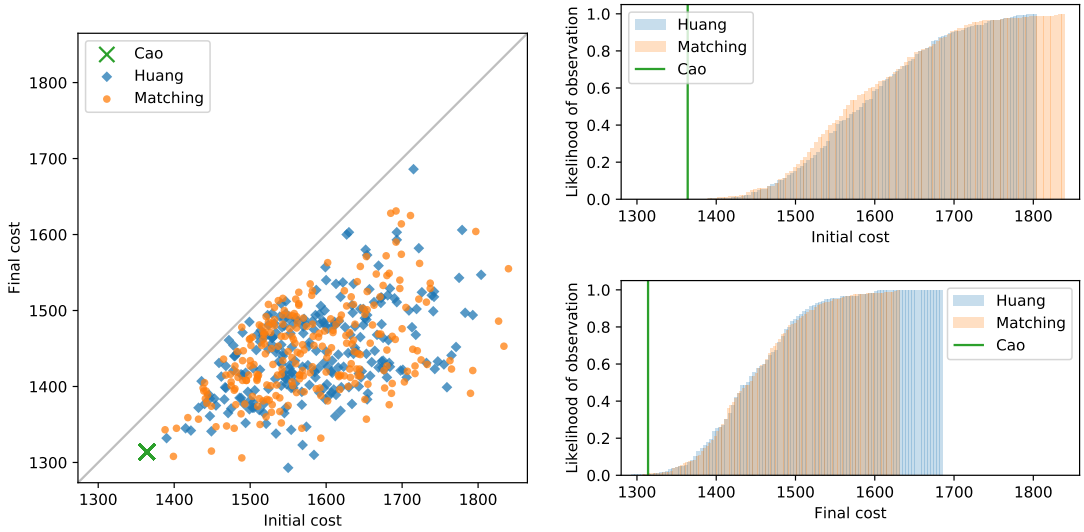
Figure 4.6: Summary plots for the mushroom dataset with $k = 2$



(a) Scatter plot of initial and final costs

(b) Empirical CDF plots for initial (top) and final (bottom) costs

Figure 4.7: Summary plots for the nursery dataset with $k = 5$



(a) Scatter plot of initial and final costs

(b) Empirical CDF plots for initial (top) and final (bottom) costs

Figure 4.8: Summary plots for the soybean dataset with $k = 15$

method is mostly comparable to the proposed extension, there is no substantial evidence to use Huang's method over the one proposed in this chapter. Figure 4.8 shows the only instance where Huang's method was able to outperform the proposed method. Other than this, the proposed method consistently performs better (or as well as) Huang's method in terms of minimal final costs and computational time. This success occurs when an external framework is imposed on the data (by choosing k to be the number of classes) and not. Furthermore, though not discussed here, the matching initialisation method has the scope to allow for expert or prior knowledge to be included in an initial clustering by using some expedient or specific preference list mechanism.

4.3.3 Using the EDO method

All of the results leading up to this point were created using benchmark datasets. The discussion at the start of Chapter 3 does not condemn this entirely as there are certainly benefits to comparing methods in this way such as familiarity and direct comparison. However, as elaborated on in that discussion, it does not afford a rich understanding of how any of the methods perform more generally. This stage of the analysis relies on the method for generating artificial datasets introduced in Chapter 3, and, as was made clear in that chapter's case study, the critical component of this method is a well-defined fitness function.

In order to expose the nuances in the performance of Cao's method and the proposed initialisation on a particular dataset, two cases are considered: where Cao's method outperforms that proposed, and vice versa. Both cases use the same fitness function, although the latter uses its negative. The function is defined as follows:

$$f(\mathcal{X}) = C_{\text{cao}} - C_{\text{match}} \tag{4.9}$$

Here, C_{cao} and C_{match} are the final costs when a dataset $\mathcal{X}$ is clustered using Cao's method and the proposed matching method respectively with $k = 3$. Given that costs should be minimised, this function is given to EDO to be minimised. As such, the case that uses f will be referred to as Cao-preferable and $-f$ will be referred to as matching-preferable. For the sake of computational time, the proposed initialisation is given 25 repetitions.

Apart from the sign of f, each case uses identical EDO parameters: a population size of $N = 500$; row and column limits of $R = (50, 500)$ and $C = (2, 50)$, respectively; selection parameters $b = 0.2$ and $l = 0$; a mutation probability of $p_m = 0.01$; a maximum number of iterations of $M = 100$. In addition to these parameters, the only

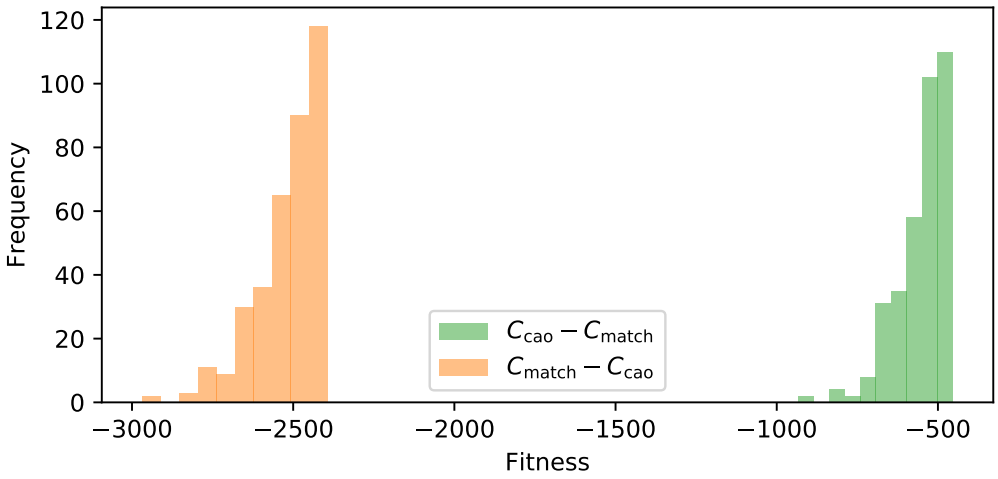


Figure 4.9: Histograms of fitness for the top performing percentile in each case

distribution family included is a discrete uniform family with up to ten categories. This means that all columns in the generated datasets will have at least one category but no more than ten.

This process yielded approximately 35,000 unique datasets for each case, and the subsequent analysis only considers the top-performing percentile of datasets from each. The typical computational time for this analysis is in the order of hours and potentially days, even when the spread over multiple cores. This time consumption may be reduced by considering a smaller run of EDO — perhaps by reducing the maximum number of iterations or the population size. Such an approach can also be useful for determining the efficacy of the fitness function and parameter set in reasonable time when applying EDO.

The datasets considered here are archived online at doi:10.5281/zenodo.3638035. Figure 4.9 shows the fitness distribution of the top percentile in each case. It should be clear from (4.9) that large negative values are preferable here. With that, and bearing in mind that the generation of these datasets was parameterised consistently, it appears that the attempt to outperform Cao's method proved somewhat more straightforward than the reverse situation; this is indicated by the substantial difference in the locations of the fitness distributions.

Given the quantity of data available, to understand the patterns that have emerged, they must be summarised — as in Chapter 3. In this case, univariate statistics are used. Despite the datasets all being of similar shapes, there are some discrepancies. With the number of rows, this is less of an issue, but any comparison of statistics across datasets of different widths is difficult without prior knowledge of the datasets. Moreover, there is no guarantee of contingency amongst the attributes, and the comparison

of more than a handful of variables becomes complicated even when the attributes are identifiable. To combat this and bring uniformity to the datasets, each dataset is represented as their first principal component obtained via centred Principal Component Analysis (PCA) [189]. While some subtleties may be lost, this representation captures the essential characteristics of each dataset in a single variable, meaning they can be compared directly.

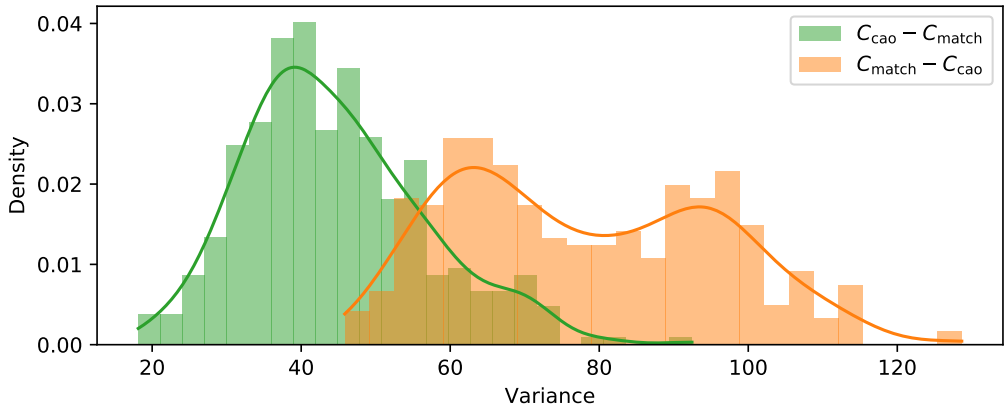
Since the transformation by PCA is centred, all measures for central tendency are moot. The mean and median are not interpretable here anyway, given that the original data is categorical. As such, the univariate statistics used here describe the spread and shape of the principal components and are split into two groups: central moments (variance, skewness and kurtosis) and empirical quantiles (the interquartile range, and lower and upper deciles).

Figures 4.10 and 4.11 show the distributions of the six univariate statistics across all of the principal components in each case. In addition to this, they show a fitted Gaussian kernel density estimate [31] to accentuate the general shape of the histograms. What is immediately clear from each of these plots is that for datasets where Cao's method performs better, the general spread of their first principal component is much tighter than in the case where the proposed initialisation method succeeds. This observation is particularly evident in Figure 4.10a where relatively low variance indicates a higher level of density in the original categorical data.

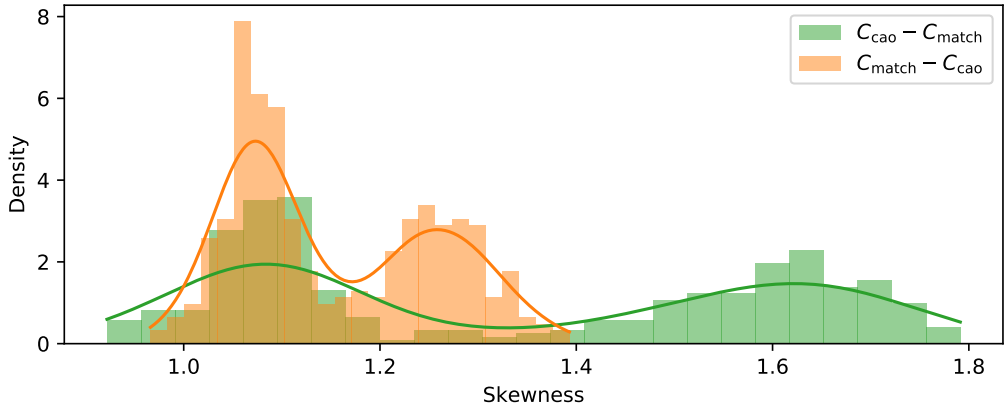
The quantiles echo this same observation. Although Figure 4.11a suggests that the components of Cao-preferable datasets can have higher interquartile ranges than in the second case, the lower and upper deciles tend to be closer together as is seen in Figures 4.11b and 4.11c, suggesting that despite the body of the component being spread, its extremities are not.

In Figures 4.10b and 4.10c, the most notable contrast between the two cases is the range in values for both skewness and kurtosis, which supports the evidence thus far that individual datasets have higher densities and lower variety (i.e. tighter extremities) when Cao's method succeeds over the proposed initialisation. In particular, larger values of skewness and kurtosis translate to a high level of similarity between the instances in a categorical dataset which is equivalent to having high density.

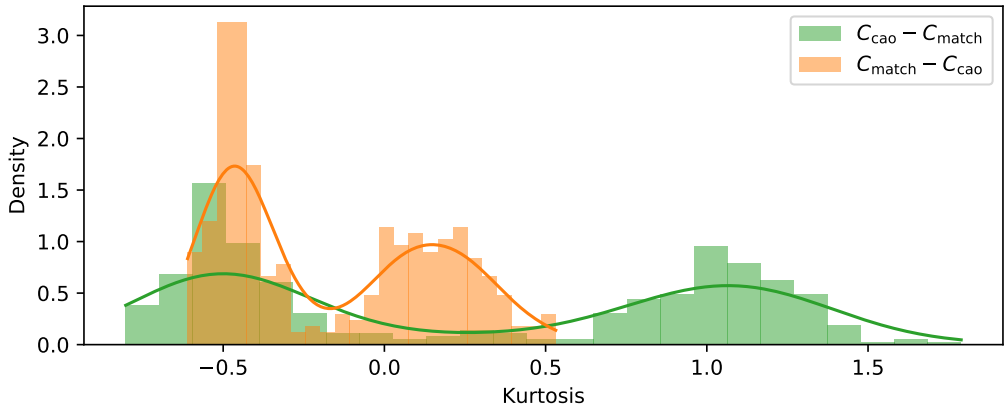
Overall, this analysis has revealed that if a dataset shows clear evidence of high-density points, then Cao's method should be used over the proposed method. However, if there is no such evidence, the proposed method can find a substantially better clustering than Cao's method.



(a)

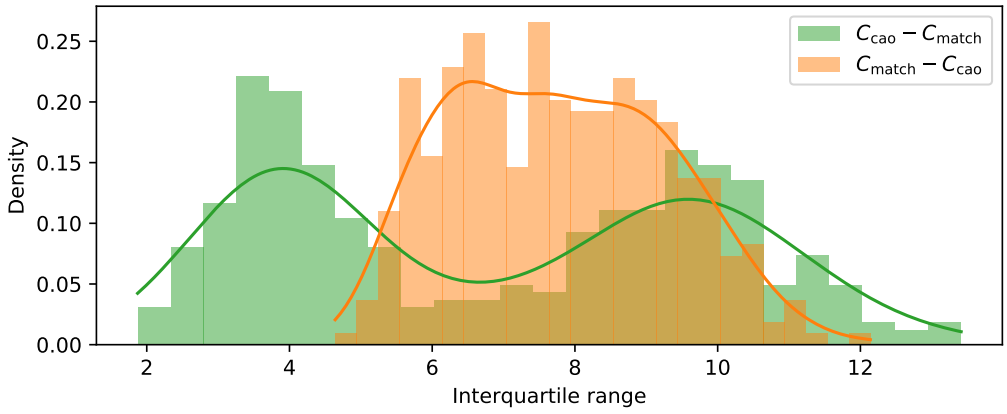


(b)

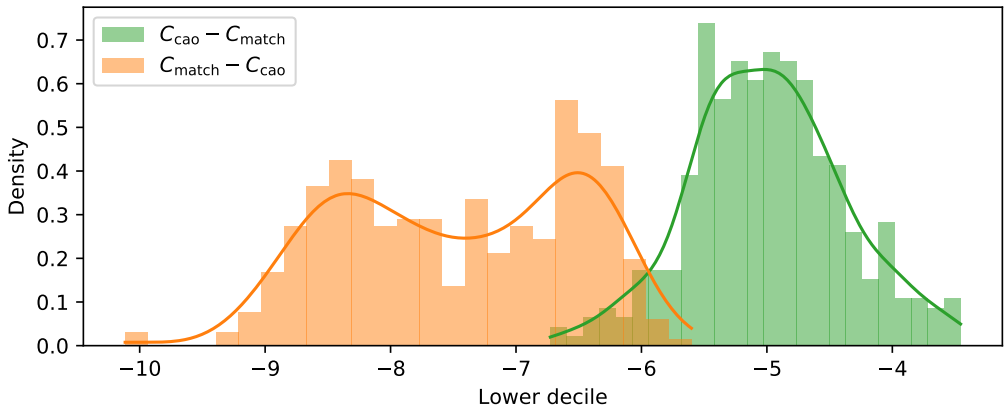


(c)

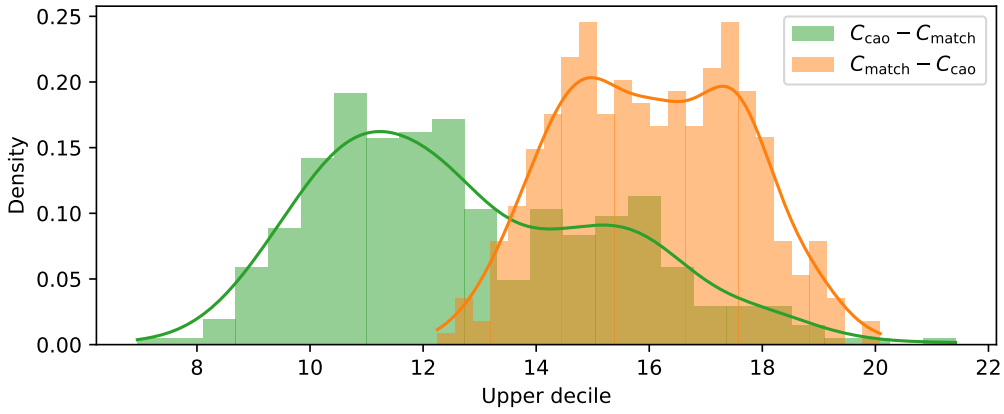
Figure 4.10: Distribution plots for the (a) variance, (b) skewness and (c) kurtosis of the first principal components in each case



(a)



(b)



(c)

Figure 4.11: Distribution plots for the (a) interquartile range, (b) lower decile and (c) upper decile of the first principal components in each case

4.4 Chapter summary

This chapter has introduced a novel initialisation method for the k-modes algorithm that builds on the method set out in the seminal paper [168]. The new method models the final replacement process in the original as an instance of the hospital-resident assignment problem that may be solved using a Gale-Shapley algorithm. The use of game theory here allows for a more mathematically fair initialisation to k-modes, adding to the literature around so-called 'fair' machine learning.

Following a thorough description of the k-modes algorithm and the established initialisation methods, a comparative analysis was conducted among the three initialisations using benchmark datasets. In addition, the EDO method described in Chapter 3 was utilised to extend this analysis to artificial datasets.

The first of these analyses revealed that the proposed initialisation was able to consistently outperform both of the other methods when the choice of k had been optimised somewhat. However, the proposed method was unable to beat Cao's method when an external, and frankly separate, framework was imposed on each dataset by choosing k to be the number of classes present.

The latter analysis, which was entirely dependent on EDO, showed that the proposed method should be employed over Cao's if there is no direct evidence that the dataset at hand has some notion of high density. Otherwise, Cao's method remains the most reliable initialisation in terms of computational time and final cost. In the next chapter, this proposed method is used to cluster a healthcare dataset because of this property.

Chapter 5

Segmentation and the recovery of queuing parameters

The research reported in this chapter has led to a manuscript entitled:

"Segmentation analysis and the recovery of queuing parameters via the Wasserstein distance: a study of administrative data for patients with chronic obstructive pulmonary disease"

Under review at: *European Journal of Operational Research*

Available online at: arXiv:2008.04295
Associated data: doi:10.5281/zenodo.3924716
Source code: doi:10.5281/zenodo.3937548

The abstract of the manuscript is as follows:

This work uses a data-driven approach to analyse how the resource requirements of patients with chronic obstructive pulmonary disease (COPD) may change, quantifying how those changes impact the hospital system with which the patients interact. This approach is composed of a novel combination of often distinct modes of analysis: segmentation, operational queuing theory, and the recovery of parameters from incomplete data. By combining these methods as presented here, this work demonstrates that potential limitations around the availability of fine-grained data can be overcome. Thus, finding useful operational results despite using only administrative data.

The paper begins by finding a useful clustering of the population from this granular data that feeds into a multi-class $M/M/c$ model, whose parameters are recovered from the data via parameterisation and the Wasserstein distance. This model is then

used to conduct an informative analysis of the underlying queuing system and the needs of the population under study through several what-if scenarios.

The analyses used to form and study this model consider, in effect, all types of patient arrivals and how those types impact the system. With that, this study finds that there are no quick solutions to reduce the impact of COPD patients on the system, including adding capacity to the system. In this analysis, the only effective intervention to reduce the strain caused by those presenting with COPD is to enact external policies which directly improve the overall health of the COPD population before they arrive at the hospital.

This chapter differs from the manuscript by expanding the literature review into Chapter 2, providing a detailed discussion of the clustering algorithm in Section 5.1, and using an improved parameterisation for the model in Sections 5.3 and 5.4.

5.1 Introduction

Population health research is increasingly based on data-driven methods for patient-centred care — as opposed to those designed solely by clinical experts. This movement is borne from the advent of accessible software and a relative abundance of electronic data. However, many such methods rely heavily on detailed data about both the healthcare system and its population, which may limit research where sophisticated data pipelines are not yet in place.

The only healthcare datasets used in this thesis are administrative hospital records. These records offer little detail as to the exact nature of a patient's time in hospital other than a surface-level summary. Appendix B carries out an exploratory analysis of a population-wide administrative dataset, revealing the presence of high variation in the population. This variability stifles the possibility of uncovering valuable insights about the whole population. However, some benefits are made apparent by considering a condition-specific population. This chapter utilises another administrative dataset of patients presenting COPD, and demonstrates how actionable insights can be identified by thoroughly extracting information from the dataset through the use of machine learning and operational research techniques.

This chapter presents a method of overcoming this, using routinely gathered, administrative hospital data to build a clustering which feeds into a multi-class queuing model, allowing for better understanding of the healthcare population and the system with which they interact. COPD is a condition of particular interest to population health research, and to Cwm Taf Morgannwg UHB, as it is known to often

present as a comorbidity in patients [165], increasing the complexity of treatments among those with the condition. Moreover, an internal report by NHS Wales found Cwm Taf Morgannwg UHB had the highest prevalence of the condition across all the Welsh health boards.

The research in this chapter draws upon several overlapping sources within mathematical research, and acts as the final stage (application) in utilising machine learning for healthcare data — the primary objective of this thesis. In the previous chapter, a method for clustering data was devised and thoroughly evaluated using both confirmation processes methods and the novel EDO approach introduced in Chapter 3. It is through this latter evaluation that the clustering initialisation described in Chapter 4 can be identified as an appropriate candidate when considering the administrative data at hand.

Furthermore, this chapter contributes to the literature in three ways: to theoretical queuing research by the estimation of missing queuing parameters with the Wasserstein distance; to operational healthcare research through the weaving together of the combination of methods used in this chapter despite data constraints; and to public health research by adding to the growing body of mathematical and operational work around a condition that is vital to understand operationally, socially and medically.

The remainder of this chapter is structured as follows:

- Section 5.1 provides an overview of the dataset and its clustering;
- Section 5.2 offers a concise introduction to queuing theory;
- Section 5.3 describes the queuing model and the estimation of its parameters;
- Section 5.4 presents several what-if scenarios with insight provided by the model parameterisation and the clustering;
- Section 5.5 summarises the chapter.

5.1.1 Overview of the dataset and its clustering

Cwm Taf Morgannwg UHB provided the dataset used in this chapter. The dataset contains an administrative summary of 5,231 patients presenting COPD from February 2011 through to March 2019, covering 10,861 hospital spells. A patient (hospital) spell is defined as the continuous stay of a patient using a hospital bed on premises controlled by a healthcare provider, and is made up of one or more patient episodes [261]. A patient episode is defined to be any continuous period of care provided by the same consultant [260]. Figure 5.1 contains an illustrative example of

the relationship between patient episodes and patient spells.

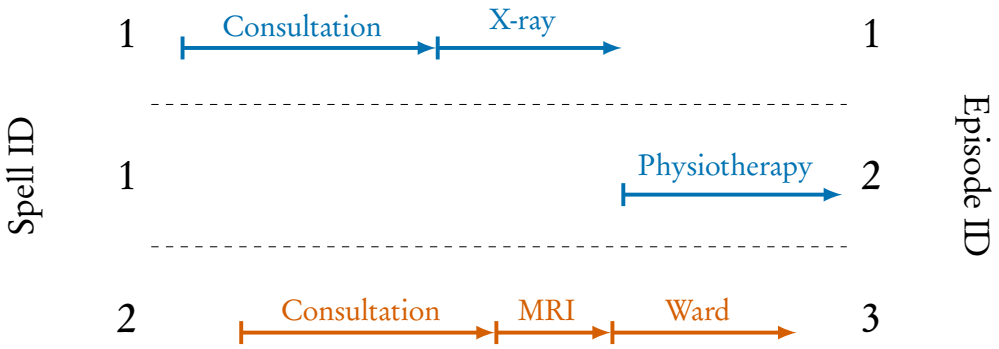


Figure 5.1: A Gantt chart of two patient spells across three episodes

In the example, the first patient begins their first episode with a consultation and an X-ray. Following this, they are referred to a physiotherapist for specialist treatment; this is their second episode, and concludes their spell. The second patient spell consists of consultation, an MRI, and monitoring on a ward. These are all overseen by the same consultant, and so they form one episode. The analysis in Appendix B considers another dataset consisting of patient episodes. However, as is often the case with administrative datasets, the order of procedures within the same episode is not recorded.

The following attributes describe the spells included in the dataset studied in this chapter:

- Personal identifiers and information, i.e. patient and spell ID numbers, and identified gender;
- Admission/discharge dates and approximate times;
- Attributes summarising the clinical path of the spell including admission/discharge methods, and the number of episodes, consultants and wards in the spell;
- International Classification of Diseases (ICD) codes and primary Healthcare Resource Group (HRG) codes from each episode;
- Indicators for any COPD intervention. The value for any given instance in the dataset (i.e. a spell) is one of no intervention, pulmonary rehabilitation (PR), specialist nursing (SN), and both interventions;
- Charlson Comorbidity Index (CCI) contributions from several long term conditions (LTCs) as well as indicators for some other conditions such as sepsis and obesity. CCI is useful in anticipating hospital utilisation as a measure for the burdens associated with comorbidity [329];

- Rank under the 2019 Welsh Index of Multiple Deprivation (WIMD), indicating relative deprivation of the postcode area the patient lives in which is known to be linked to COPD prevalence and severity [80, 323, 339].

In addition to the above, the following attributes were engineered for each spell:

- Age and spell cost data were linked to approximately half of the spells in the dataset from the administrative dataset analysed in Appendix B;
- The presenting ICD codes were generalised to their categories according to NHS documentation and counts for each category were attached. This reduced the number of values from 1,926 codes to 21 categories;
- A measure of admission frequency was calculated by taking the number of COPD-related admissions in the last twelve months linked to the associated patient ID number.

Although there is a fair amount of information here, it is limited to COPD-related admissions. Therefore, rather than segmenting the patients themselves, the spells will be segmented.

The attributes included in the clustering encompass both utilisation metrics and clinical attributes relating to the spell. They comprise the summary clinical path attributes, the CCI contributions and condition indicators, the WIMD rank, length of stay (LOS), COPD intervention status, and the engineered attributes (not including age and costs due to lack of coverage between the two datasets).

With these attributes selected, a clustering algorithm must be chosen. Two critical specifications of the algorithm used are that it must handle mixed-type data, and that it should be interpretable by stakeholders. As such, the k-prototypes algorithm is a strong candidate. The k-prototypes algorithm was mentioned in Chapter 4 and is a mixed-type extension to the k-modes and k-means algorithms; in effect, the k-prototypes algorithm separates the given dataset into its numeric and categorical attributes before applying k-means and k-modes on the respective parts. The statement of the k-prototypes algorithm has been omitted since it is equivalent to that of k-modes (given in Algorithm 4.1) with the exceptions that:

- The SelectClosest function uses the dissimilarity measure given in (5.2);
- The Update function is as given in Algorithm 5.1.

These parts are combined using a modified dissimilarity function, defined in (5.2). This function is a linear combination of the squared Euclidean distance and the dissimilarity function defined in (4.1) according to a weight, $\gamma \in \mathbb{R}$. The notation and terminology for clustering mixed-type data is much the same as in Chapter 4. How-

ever, there are substantial differences: first, representative points are referred to as *prototypes*; and, second, an attribute space $\mathcal{A}$ of m mixed-type attributes can be written as the product of its numeric and categorical components:

$$\mathcal{A} = \prod_{j=1}^{p} A_j^{(n)} \times \prod_{j=p+1}^{m} A_j^{(c)} = \mathcal{A}^{(n)} \times \mathcal{A}^{(c)} \tag{5.1}$$

Here, $A^{(n)}$ and $A^{(c)}$ denote individual numeric and categorical attributes, respectively. Meanwhile, $\mathcal{A}^{(n)}$ and $\mathcal{A}^{(c)}$ denote the numeric and categorical components of the space. With this notation, the dissimilarity between two points, $X, Y \in \mathcal{A}$, is defined to be:

$$d(X,Y) = \sum_{j=1}^{p} \left(x_j - y_j\right)^2 + \gamma \sum_{j=p+1}^{m} \delta\left(x_j, y_j\right) \tag{5.2}$$

Algorithm 5.1: UPDATE (k-prototypes)

Input: an attribute space $\mathcal{A} = \mathcal{A}^{(n)} \times \mathcal{A}^{(c)}$, a prototype to update $z^{(l)}$ and its cluster Z_l

Output: an updated prototype

1 Find $z_n \in \mathcal{A}^{(n)}$, the mean numeric attribute vector in Z_l:

$$z_n := \left(\frac{1}{|Z_l|} \sum_{u \in Z_l} u_j : j = 1, \ldots, p \right)$$

2 Find $z_c \in \mathcal{A}^{(c)}$ that minimises $D\left(Z_l^{(c)}, z_c\right)$ where:

$$Z_l^{(c)} := \left\{ (u_j : j = p+1, \ldots, m) : u \in Z_l \right\}$$

i.e. find the modal categorical attribute vector in Z_l

3 Update the prototype to be the concatenation of these vectors:

$$z^{(l)} \leftarrow z_n \frown z_c$$

In addition to this dissimilarity function, k-prototypes has a cost function that uses the same linear combination as its dissimilarity function to consider the numeric and categorical attributes. This function combines the categorical cost function (defined in (4.4)) and inertia (defined in (3.4)) according to the same value of γ. A proof that minimising this linear combination also minimises the intra-cluster k-prototypes dissimilarity is given in [166].

The choice of γ is of particular importance as it balances the contribution of each data type to the objective function. The seminal work by Huang [166] investigated the effect of various γ values when clustering with k-prototypes. This investigation determined that a sensible and robust value for γ is the average of the standard deviations for the numeric attributes. The analysis that informed the clustering in this chapter found that this value for γ provided a useful clustering; as such, no further modifications were made.

To determine the optimal number of clusters, k, the knee point detection algorithm used at the end of Chapter 4 was used with a range of potential values for k from two to ten. This range was chosen based on what may be considered feasibly informative to stakeholders. Applying this algorithm revealed an optimal value for k of four, but both three and five clusters were considered. Both of these cases were eliminated due to a lack of clear separation in the characteristics of the clusters.

Although the dataset is confidential and may not be published, a synthetic analogue which illustrates the clustering has been archived under `doi:10.5281/zenodo.3908167`. A summary of the dataset and its clustering is provided in Table 5.1. Note that a negative length of stay indicates that the patient had passed away prior to arriving at the hospital and so these spells have been omitted from further analysis. This table separates each cluster and the overall dataset (referred to as the population). From this table, helpful insights can be gained about the segments identified by the clustering. For instance, the needs of the spells in each cluster can be summarised succinctly:

- Cluster 0 represents those spells with relatively *low clinical complexity but high resource requirements*. The mean spell cost is almost four times the population average, and the shortest spell is almost two weeks long. Moreover, the median number of COPD-related admissions in the last year is elevated, indicating that patients presenting in this way require more interactions with the system.
- Cluster 1, the second-largest segment, represents the spells with *complex clinical profiles despite lower resource requirements*. Specifically, the spells in this cluster have the highest median CCI and number of LTCs, and the highest condition prevalence across all clusters but the second-lowest length of stay and spell costs.
- Cluster 2 represents the majority of spells and those where *resource requirements and clinical complexities are minimal*; these spells have the shortest lengths, and the patients present with fewer diagnoses and a lower median CCI than any other cluster. In addition to this, the spells in Cluster 2 have the highest intervention prevalence. However, they have the lowest condition prevalence

		Cluster				Population
		0	1	2	3	
Characteristics	**Percentage of spells**	9.90	19.27	69.39	1.44	100.00
	Mean spell cost, £	8051.23	2309.63	1508.41	17888.43	2265.40
	Percentage of recorded costs	29.01	19.38	48.20	3.40	100.00
	Median age	77.00	77.00	71.00	82.00	73.00
	Minimum LOS	12.82	0.01	0.00	48.82	0.00
	Mean LOS	25.31	6.47	4.11	75.36	7.69
	Maximum LOS	51.36	30.86	16.94	224.93	224.93
	Median COPD adm. in last year	2.00	1.00	1.00	2.00	1.00
	Median no. of LTCs	2.00	3.00	1.00	3.00	1.00
	Median no. of ICDs	9.00	8.00	5.00	11.00	6.00
	Median CCI	9.00	20.00	4.00	18.00	4.00
Intervention prevalence	**None, %**	80.19	83.42	65.76	89.74	70.94
	PR, %	15.81	13.43	27.98	8.97	23.69
	SN, %	3.81	2.87	4.63	1.28	4.16
	Both, %	0.19	0.29	1.63	0.00	1.21
LTC prevalence	**Pulmonary disease, %**	100.00	100.00	100.00	100.00	100.00
	Diabetes, %	19.07	28.14	14.84	25.00	17.97
	AMI, %	13.86	22.93	8.76	16.03	12.10
	CHF, %	12.47	53.80	0.00	26.28	11.98
	Renal disease, %	7.53	19.54	1.92	17.95	6.11
	Cancer, %	7.53	12.28	2.93	10.90	5.30
	Dementia, %	6.88	21.26	0.00	26.92	5.17
	CVA, %	8.65	13.33	0.70	19.87	4.20
	PVD, %	4.37	7.69	2.27	5.77	3.57
	CTD, %	5.12	4.25	3.11	4.49	3.55
	Obesity, %	2.51	3.01	1.49	7.69	1.97
	Metastatic cancer, %	1.49	4.54	0.00	0.64	1.03
	Paraplegia, %	1.30	3.73	0.24	0.64	1.02
	Diabetic compl., %	0.19	0.86	0.48	1.92	0.54
	Peptic ulcer, %	1.58	0.81	0.23	1.28	0.49
	Sepsis, %	1.77	0.91	0.15	1.92	0.48
	Liver disease, %	0.28	0.48	0.23	0.00	0.28
	C. diff, %	0.74	0.10	0.01	0.64	0.11
	Severe liver disease, %	0.19	0.43	0.00	0.00	0.10
	MRSA, %	0.28	0.05	0.03	1.28	0.07
	HIV, %	0.00	0.00	0.03	0.00	0.02

Table 5.1: A summary of clinical and condition-specific characteristics for each cluster and the population

across all clusters.

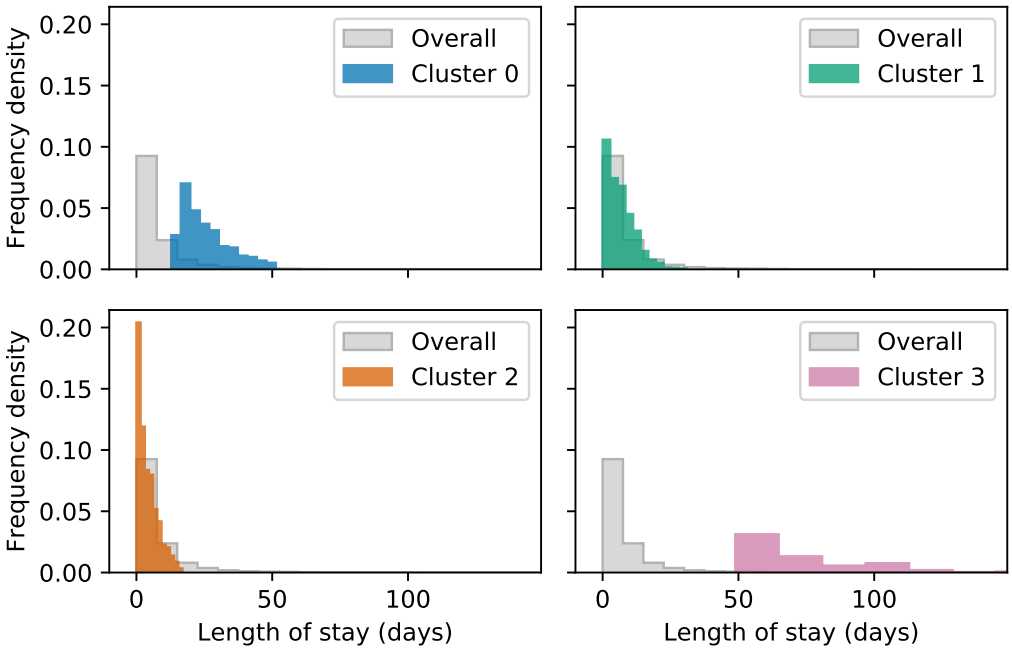
- Cluster 3 represents the smallest section of the population but perhaps the most critical: spells with *high complexity and high resource needs.* The patients within Cluster 3 are the oldest in the population and are some of the most frequently returning despite having the lowest intervention rates. The lengths of stay vary between seven and 32 weeks, and the mean spell cost is almost eight times the population average. This cluster also has the second-highest median CCI, and the highest median number of concurrent diagnoses.

The attributes listed in Table 5.1 can be studied beyond summaries such as these, however. Figures 5.2 through 5.6 show the distributions for some clinical characteristics for each cluster. Each of these figures also shows the distribution of the same attributes when splitting the population by intervention. While this classical approach — of splitting a population based on a condition or treatment — can provide some insight into how the different interventions are used, it has been included to highlight the value added by segmenting the population via data without such a prescriptive framework.

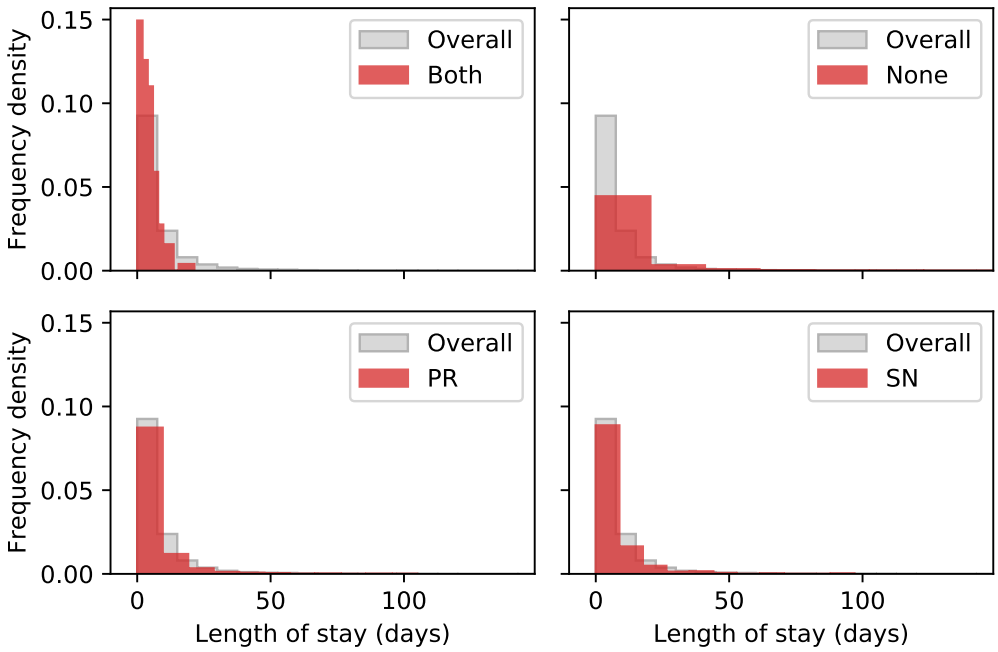
Figure 5.2 shows the length of stay distributions as histograms. Figure 5.2a demonstrates the different bed resource requirements well for each cluster — better than Table 5.1 might — in that the difference between the clusters is not only a matter of varying means and ranges, but entirely different shapes to their respective distributions. Indeed, they are all positively skewed, but there is no real consistency beyond that. When comparing this to Figure 5.2b, there is undoubtedly some variety, but the overall shapes of the distributions are generally similar. The exception is the spells with no COPD intervention, where binning could not improve the visualisation due to the widespread distribution of their lengths of stay.

The same conclusions can be drawn about spell costs from Figure 5.3; there are distinct patterns between the clusters in terms of their costs, and they align with the patterns seen in Figure 5.2. Such patterns are expected given that length of stay is a driving force of healthcare costs. Equally, there does not appear to be any immediately discernible difference in the distribution of costs when splitting by intervention.

Similarly to the previous figures, Figure 5.4 shows that clustering has revealed distinct patterns in the CCI of the spells within each cluster, whereas splitting by intervention does not. All clusters other than Cluster 2 show clear, heavy tails, and in the cases of Clusters 1 and 3, the body of the data exists far from the origin as indicated in Table 5.1. In contrast, the plots in Figure 5.4b all display similar, highly skewed distributions regardless of intervention.

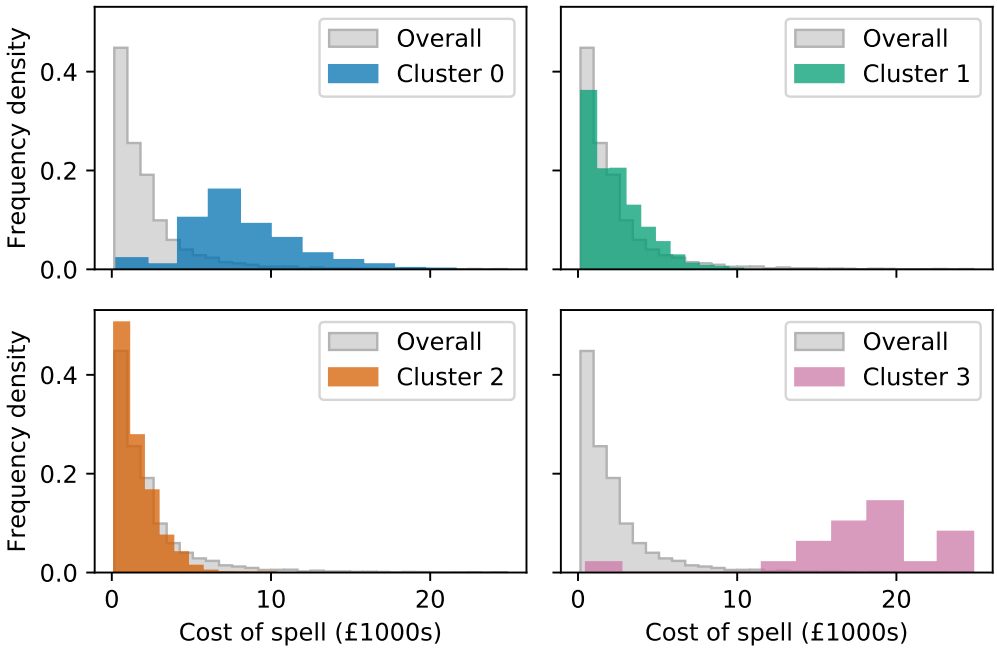


(a)

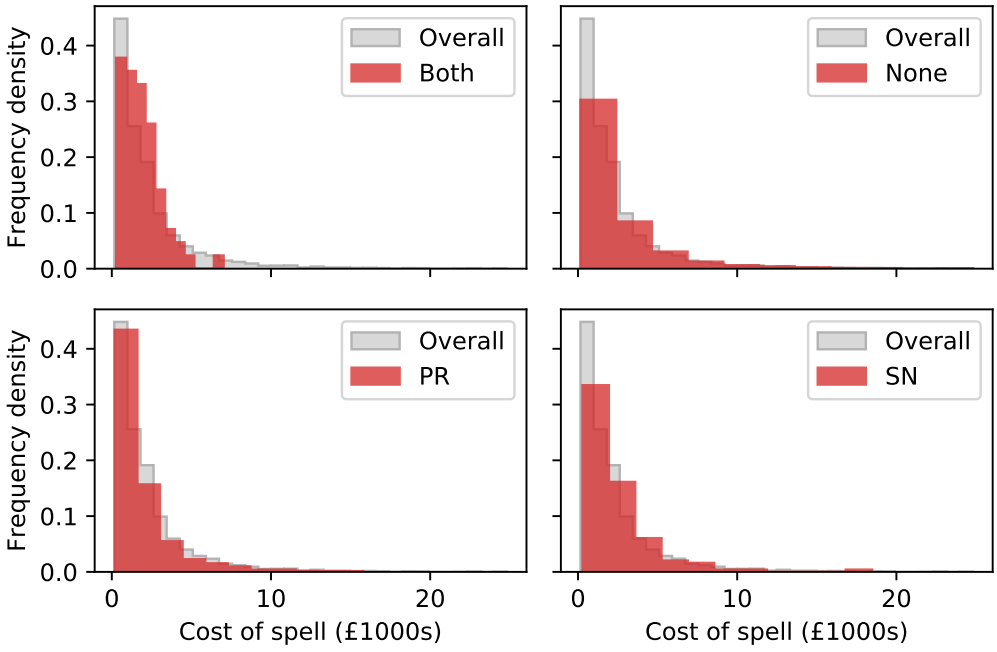


(b)

Figure 5.2: Histograms for length of stay by (a) cluster and (b) intervention

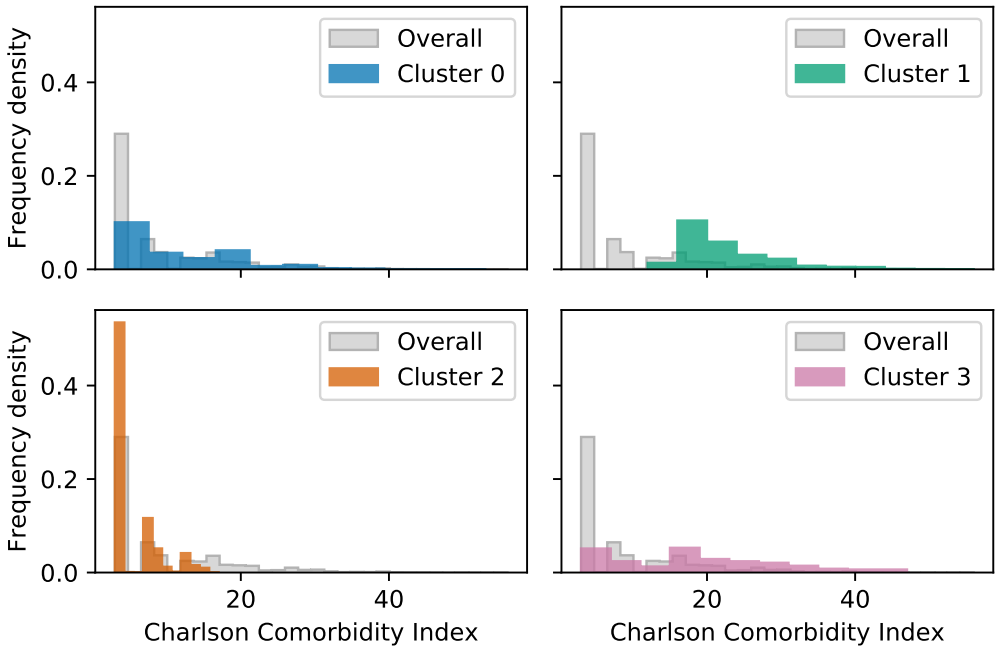


(a)

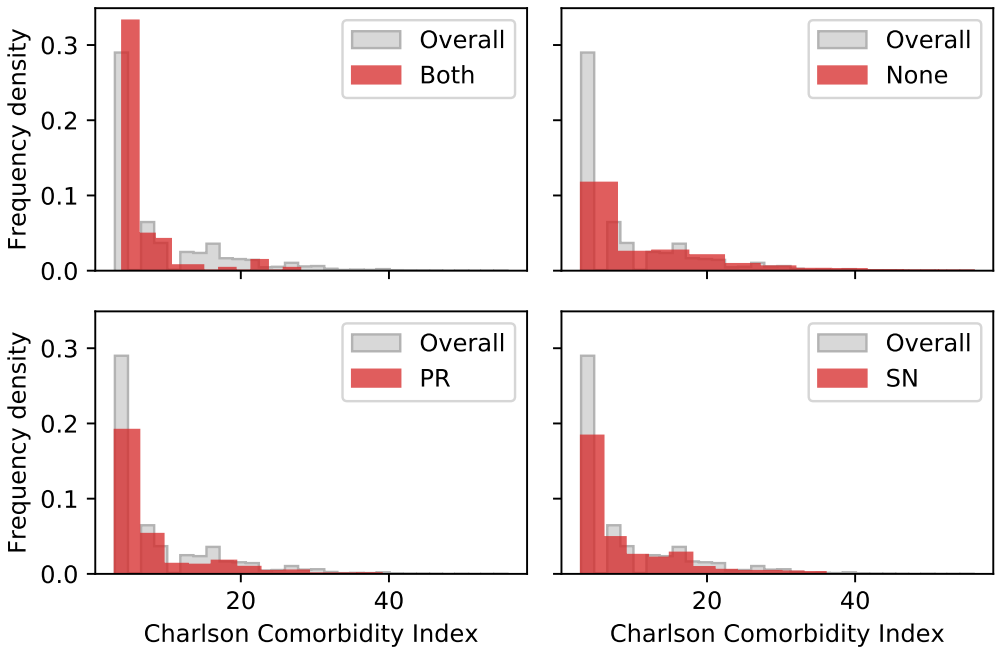


(b)

Figure 5.3: Histograms for spell cost by (a) cluster and (b) intervention

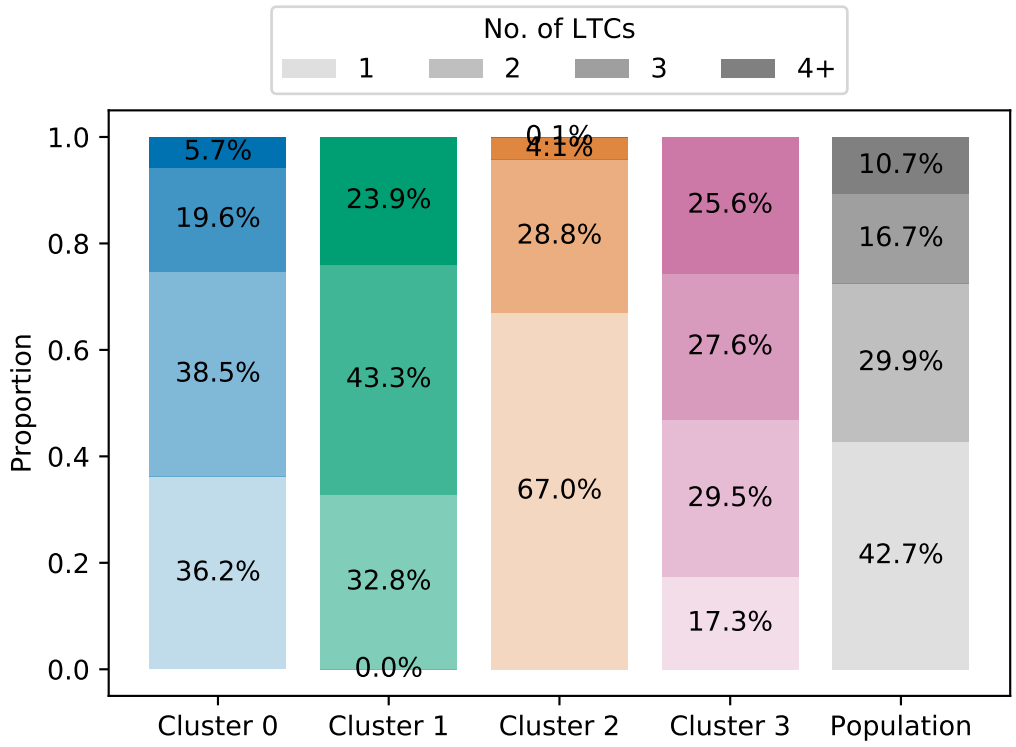


(a)

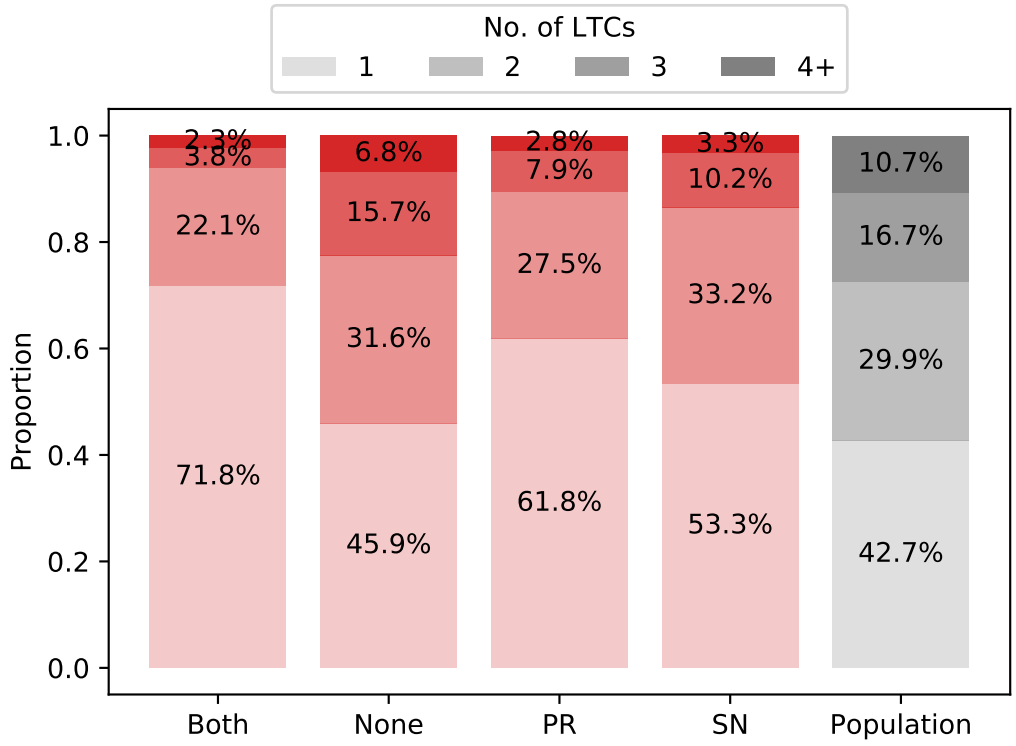


(b)

Figure 5.4: Histograms for CCI by (a) cluster and (b) intervention

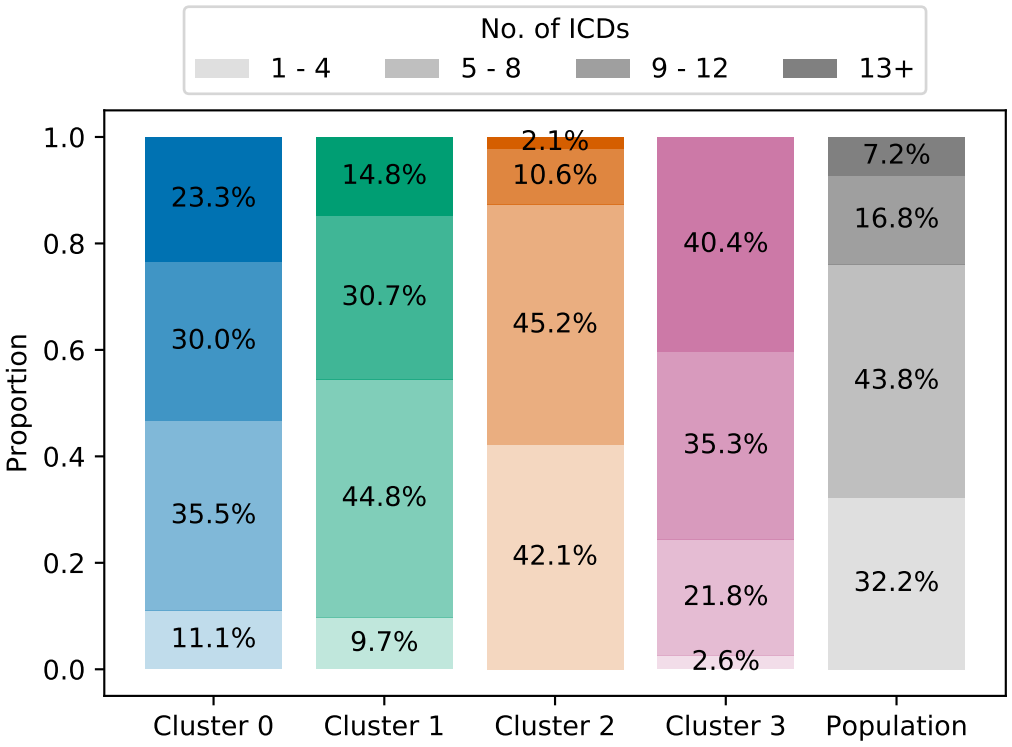


(a)

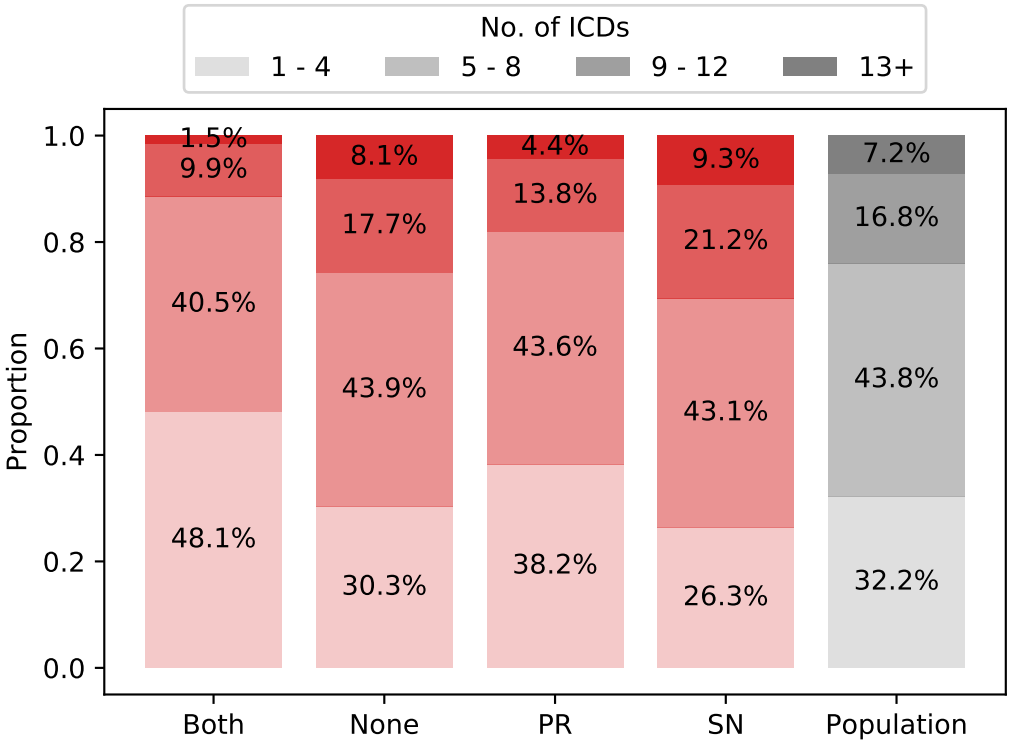


(b)

Figure 5.5: Proportions of the number of concurrent LTCs in a spell by (a) cluster and (b) intervention



(a)



(b)

Figure 5.6: Proportions of the number of concurrent ICDs in a spell by (a) cluster and (b) intervention

Figures 5.5 and 5.6 show the proportions of each grouping presenting levels of concurrent LTCs and ICDs, respectively. By exposing the distribution of these attributes, some notion of the clinical complexity for each cluster can be captured better than with Table 5.1 alone. In Figure 5.5a, for instance, there are distinct LTC count profiles among the clusters: Cluster 0 is typical of the population; Cluster 1 shows that no patient presented COPD solely as an LTC in their spells, and more than half presented at least three; Cluster 2 is similar in form to the population but is strongly biased towards patients presenting COPD as the only LTC; Cluster 3 has the closest-to-uniform spread among the four bins despite the increased length of stay and CCI, suggesting a diverse array of patients in terms of their long term medical needs.

Figure 5.6a largely mirrors these cluster profiles with the number of concurrent ICDs. There are some points of interest, however. Firstly, Cluster 1 has a relatively low-leaning distribution of ICDs that does not marry up with the high rates of LTCs. Secondly, the vast majority of spells in Cluster 3 present with at least nine ICDs suggesting a likely wide range of conditions and comorbidities beyond the LTCs used to calculate CCI.

However, little can be drawn from the intervention counterparts to these figures (i.e. Figures 5.5b and 5.6b), regarding the corresponding spells. One thing of note is that patients receiving both interventions for their COPD (or either, in fact) have disproportionately fewer LTCs and concurrent ICDs when compared to the population. Aside from this, the profiles of each intervention are similar to one another.

As discussed earlier, the purpose of this chapter is to construct a queuing model for the data described here. Insights have already been gained into the needs of the segments that have been identified in this section. However, to glean further insights, some parameters of the queuing model must be recovered from the data. The following two sections briefly introduce queues, and describe how these parameters are derived using the dataset at hand, respectively.

5.2 An introduction to queues

Queues facilitate the orderly provision of services. Examples include lining up to board a bus, assembly lines in a factory, or patients arriving at a hospital. In all queues there are two types of agent: those providing the service (a bus driver), and those demanding it (the passengers). The generic terminology for these agents are *servers* and *customers*, respectively.

As well as individual queues, networks of interconnecting queues can be described in a similar manner. In a *queuing network*, each individual queue is called a *node*. For

instance, a hospital could be considered a network of queues, where patients arrive into triage, are processed, and are redirected throughout their spell.

The observed characteristics of a queuing system can be used to construct a mathematical model. Such a model would describe things like the process by which customers arrive to a queue, the rules that allow customers to be served, and the time taken to serve a customer. Queuing theory is the branch of mathematics concerned with the analysis of these models.

The remainder of this section defines some of the fundamental elements necessary to discuss the queues used in this chapter. Queuing theory is a mature discipline with many facets that extend beyond the needs of this chapter, and the scope of this thesis. Comprehensive and informative introductions to queuing models, queuing theory, and the simulating of queues can be found in [42, 326, 341]. Further, applications of queuing models to healthcare systems are plentiful, but examples include [47, 77, 248, 340, 381, 391].

5.2.1 Elements of a queue

A queue is made up several components: the service facility, a number of servers within that facility, a line in which customers wait to be served, and a stream of arriving customers. Figure 5.7 shows a diagram of a queue. The characteristics associated with the components of a queue are often summarised using Kendall's notation [341]. The exact notation varies somewhat, but here it shall be denoted $A/S/c/m/K/Q$. This notation also defines the *parameters* of the queue. The process in Section 5.3 estimates some unknown parameters of a queue.

Kendall's notation acts as a shorthand to fully describe a queue, and is as follows. Customers arrive to the queue according to an *inter-arrival time distribution*, A, and wait in line to be served according to a *queuing discipline*, Q. Typically, customers are served as they arrive. This discipline is called FIFO (first in first out). Other disciplines include LIFO (last in first out) and priority scheduling — as used in emergency triage. At the service facility there are c parallel servers, who each serve customers according to the *service time distribution*, S. Sometimes it is beneficial to attach a capacity to the system, denoted by $K \geq c$. If omitted, an unlimited system capacity is presumed. The *system capacity* can also be distinguished from an optional *queue capacity*, $m < K$, which limits the number of customers allowed to wait in line. Again, this is assumed to be ∞, unless specified.

The distributions used to model inter-arrival and service times are numerous, but some commonly studied examples are:

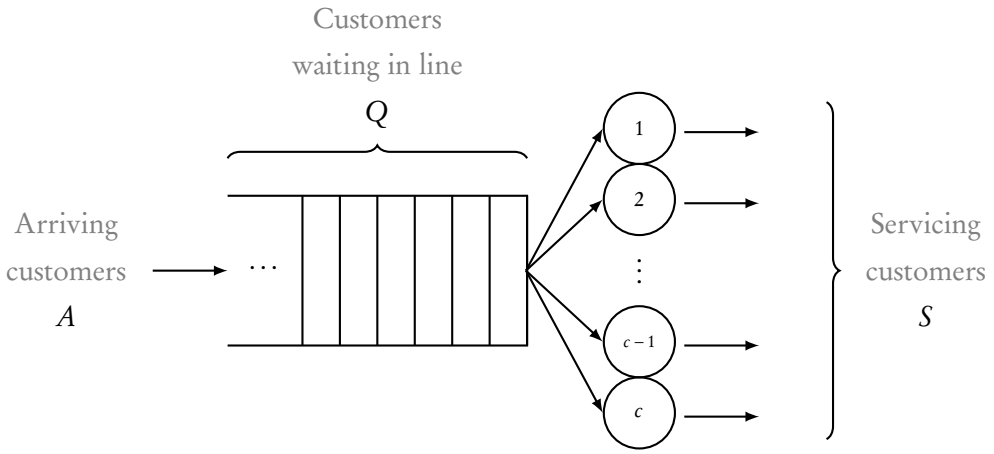


Figure 5.7: The anatomy of a queue

- Markovian (denoted M). Customers arrive according to a Poisson process. The inter-arrival or service times follow an exponential distribution with rate $\alpha > 0$, i.e. they have probability density function:

$$f(t) = \alpha e^{-\alpha t}; \quad t \geq 0 \tag{5.3}$$

- Deterministic (denoted D). Inter-arrival or service times are non-stochastic and are of fixed length.

- General (denoted G). Arrivals are random, and inter-arrival or service times follow a general probability distribution.

5.2.2 Some classical queues

The $M/M/1$ queue

One of the best-known queues is the $M/M/1$ queue. In this queue, customers arrive according to a Poisson process at a rate of λ. The customers are served by a single server exponentially, at a rate of μ. Owing to the memoryless property of the exponential distribution, this queue can be represented as a continuous-time Markov chain over the state space $\mathbb{N}_0 = \{0, 1, 2, \ldots\}$.

A stochastic process, (X_t), which is defined over a countable state space, $\mathcal{S}$, is considered a *Markov chain* if and only if for all $n \in \mathbb{N}$ and for any $(n+1)$-tuple of states, $s_0, \ldots, s_n \in \mathcal{S}^n$, the process satisfies:

$$\mathbb{P}(X_n = s_n \mid X_{n-1} = s_{n-1}, \ldots, X_0 = s_0) = \mathbb{P}(X_n = s_n \mid X_{n-1} = s_{n-1}) \tag{5.4}$$

That is, the probability of being in state s_n is dependent only on the previous state, s_{n-1}. The Markov chain underlying an $M/M/1$ queue is also known as a *birth-death process*. Figure 5.8 shows a diagram of the birth-death process of an $M/M/1$ queue.

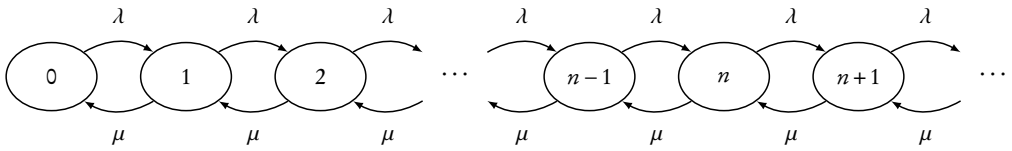


Figure 5.8: A state space diagram for an $M/M/1$ queue

An important property of the $M/M/1$ queue is its *traffic intensity*, which is defined as $\rho = \frac{\lambda}{\mu}$. This quantity also represents the proportion of time that the server spends serving customers, and so measures their *utilisation*. The $M/M/1$ queue is considered *stable* (i.e. the underlying process will become stationary eventually) if $\rho < 1$. In an unstable queue, customers arrive at a faster rate than they are served, and so the line grows indefinitely.

Many other properties of the $M/M/1$ queue can be explicitly derived from this representation, including steady-state solutions, the expected size of the system, and average response times. The calculation of these quantities also makes use of Little's Law [219], which relates average system size, L, with average waiting time, W, in stationary processes such that:

$$L = \lambda W \tag{5.5}$$

The $M/M/c$ queue

The $M/M/c$ queue is an extension of the $M/M/1$ queue where there are $c \in \mathbb{N}$ independent servers working in parallel. Customers still arrive according to a Poisson process with rate λ, and customer service times follow an exponential distribution with a mean of $\frac{1}{\mu}$. The traffic intensity of the $M/M/c$ queue is also $\rho = \frac{\lambda}{\mu}$, but server utilisation is measured as the mean traffic intensity across the servers, i.e. $\frac{\rho}{c}$. The stability condition for the $M/M/c$ queue is $\rho < c$.

As with the $M/M/1$ queue, the $M/M/c$ queue can be represented as a birth-death process on the state space $\mathbb{N}_0$, as shown in Figure 5.9. Since there are multiple servers, some servers may be idle when there are customers in the system. Once all servers are active, arriving customers join the line and wait for service.

The $M/G/c$ queue

As useful as memoryless service times are in deriving properties of queues, they are not always representative of real systems. The $M/G/c$ queue extends the $M/M/c$

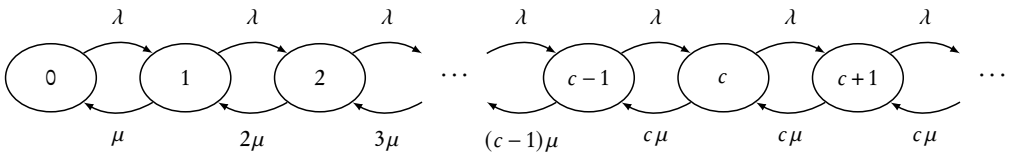


Figure 5.9: A state space diagram for an $M/M/c$ queue

queue to allow for a general service time distribution. Again, there are c parallel servers, and customers arrive randomly at a rate of λ.

The $M/G/c$ queue cannot be represented as a Markov chain, but it is a stochastic process on the same state space as the other queues in this section. Given the generic nature of customer departure times, a lot of the structure of the underlying process is lost. As such, deriving exact values for many properties of the $M/G/c$ queue continues to be an open problem [199]. Despite the theoretical challenges posed by the $M/G/c$ queue, simulating these generic queues can still be of great benefit — as is done in Section 5.3.

5.2.3 Simulation tools

As well as theoretical results, queues provide a valuable basis for computer simulation. Theoretical models are limited when studying complex queuing systems, where the parameterisation of a system requires complicated notation or derivations to produce useful results. When properly utilising computer simulation, the stochastic intricacies of a system may be more readily observed. Examples of such systems include the multi-class queuing networks studied in [77].

There are numerous tools available for simulating queues, but many leave the associated research prone to issues like reproducibility — as mentioned in Section 2.3.2. A recent review on the subject and its tools is [87]. One of the defining features of a simulation tool is whether it has a *graphical user interface* (GUI) or not. GUIs provide accessibility to the non-technical members of a simulation project, but can also foster poor simulation practices [36].

The simulation framework of choice in this chapter is the discrete event simulation library, Ciw [276]. Ciw is written in Python, and is a well-developed piece of open-source software, adhering to the same best practices as the other software packages developed during this project. In [276], the authors stress how ensuring sustainable and reproducible simulation work are at the core of their development process.

5.3 Constructing the queuing model

The data available in this chapter is not as detailed as in comparative projects. Without access to such data — but intending to gain insight from what is available — it is imperative to bridge the gap left by the incomplete data. Figure 5.10 provides a diagrammatic depiction of the process described in this section.

It is often the case that in practical situations where suitable data is not (immediately) available, further inquiry in that line of research will stop. Queuing models in healthcare settings appear to be such a case; the line ends at incomplete queue data. The bibliographic work [22] collates articles on the estimation of queuing system characteristics — including their parameters. Despite its breadth of almost 300 publications from 1955, only two articles have been identified as being applied to healthcare: [248, 391]. Both works are concerned with customers who can re-enter services during their time in the queuing system, which is mainly of value when considering the effect of unpredictable behaviour in intensive care units, for instance. In [248], the authors seek to approximate service and re-service densities through a Bayesian approach and by filtering out those customers seeking to be serviced again. Meanwhile, the approach in [391] considers an extension to the $M/M/c$ queue with direct re-entries. The devised model is then used to determine resource requirements in two healthcare settings.

Aside from healthcare-specific works, the approximation of queue parameters has formed a part of relevant modern queuing research. However, the scope is primarily focused on theoretic approximations rather than simulation. For instance, two recent works [101, 138] consider an underlying process to estimate a general service time distribution in single server and infinite server queues respectively.

While these solutions are interesting, they do not necessarily tackle the issue in this scenario where information about the system is also missing. With that, there is a precedent for simplifying healthcare systems to a single node with parallel servers that emulate overall resource availability. Two studies [340, 381] provide examples of how this approach, when paired with discrete event simulation, can expose the resource needs of a system beyond deterministic queuing theory models. In particular, the authors of [381] show how a single node, multiple server queue can be used to accurately predict bed capacity and length of stay distributions in a critical care unit using administrative data.

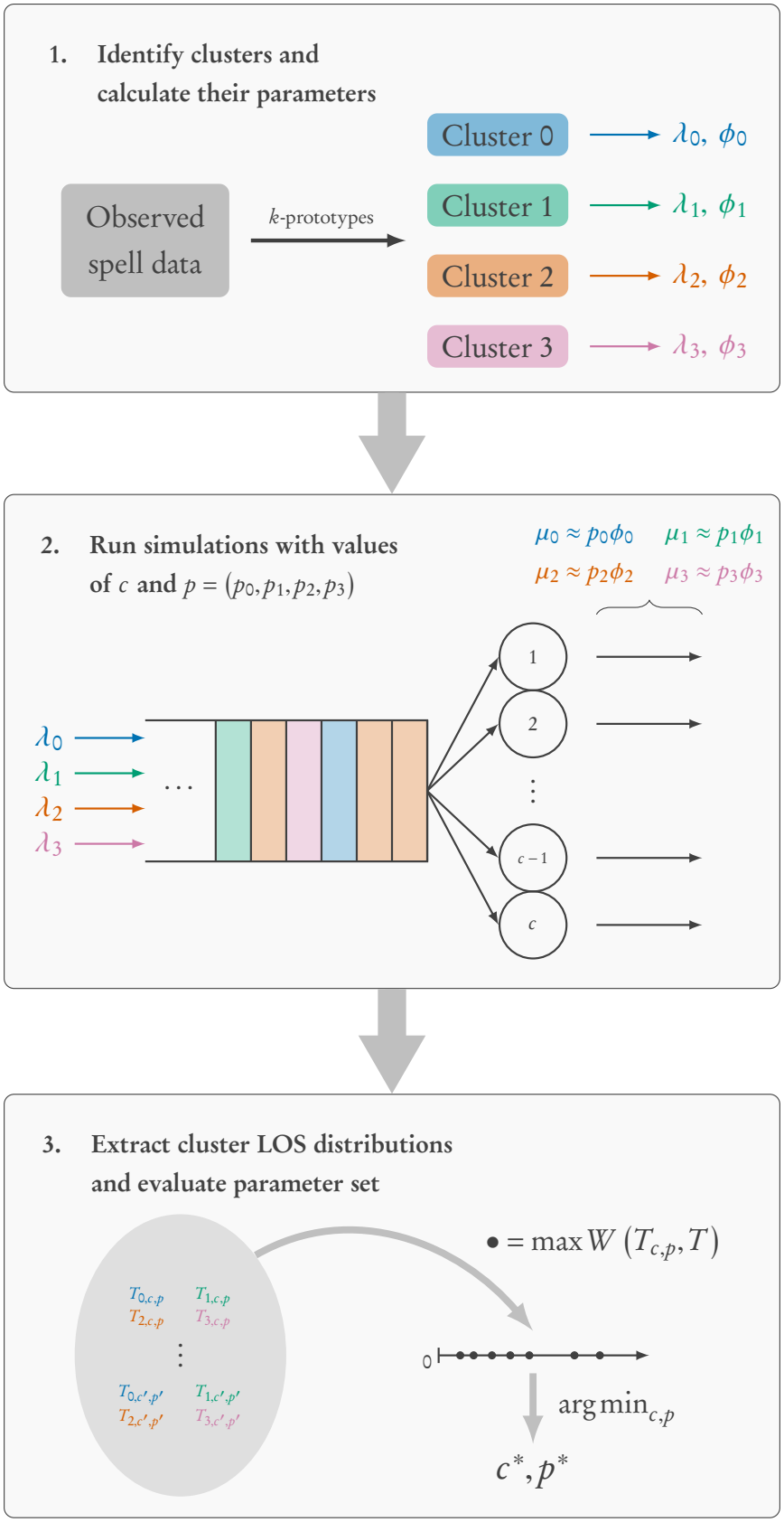


Figure 5.10: A diagrammatic depiction of the queuing parameter recovery process

5.3.1 Deriving the model parameters

Following in the suit of recent literature [340, 381], this chapter employs a single node using the $M/M/c$ queue to model a hypothetical ward of patients presenting COPD. In addition to this, the grouping found in Section 5.1.1 provides a set of patient classes in the queue. Under this model, the following assumptions are made:

1. Inter-arrival and service times of patients are each exponentially distributed with some mean. This distribution is used to simplify the model parameterisation.

2. There are $c \in \mathbb{N}$ servers available to arriving patients at the node representing the overall resource availability, including bed capacity and hospital staff.

3. There is no queue or system capacity. In [381], a queue capacity of zero is set under the assumption that any surplus arrivals would be sent to another suitable ward or unit. As this hypothetical ward represents the sole unit for COPD patients within the health board, this assumption is not held.

4. Without the availability of expert clinical knowledge, a first-in-first-out service policy is employed in place of some patient priority framework.

Each group of patients has its arrival distribution, the parameter of which is the reciprocal of the mean inter-arrival times for that group. This parameter is denoted by λ_l for each cluster l.

Like arrivals, each group of patients has its service time distribution. Without full details of the process order or idle periods during a spell, some assumption must be made about the actual 'service' time of a patient in the hospital. It is assumed here that the mean service time of a group of patients may be approximated via their mean length of stay, i.e. the mean time spent in the system. As indicated by the distributions in Figure 5.2a, the length of stay distributions require shifting prior to fitting an exponential distribution.

Let T_l denote the set of observed lengths of stay for cluster l, and let $m_l = \max\{0, \min T_l\}$ be its feasible minimum. Thus, the *shifted times* for cluster l, denoted $\widehat{T}_l$, are:

$$\widehat{T}_l := \{t - m_l : t \in T_l\} \tag{5.6}$$

An exponential distribution may be fitted to these shifted system times by using their mean, denoted by $\frac{1}{\phi_l}$, as the distribution parameter. For the sake of simplicity, it is assumed that for each cluster l, the mean shifted service time of that cluster, $\frac{1}{\mu_l}$,

is proportional to the corresponding mean shifted system time such that:

$$\mu_l = p_l \phi_l \tag{5.7}$$

where $p_l \in (0, 1]$ is a *service proportion* parameter to be determined for each group.

With these definitions, the service time for cluster l, denoted S_l, is distributed by a *shifted exponential distribution* with a mean of $\frac{1}{\mu_l}$ and shift of m_l. The probability density function of this distribution is as follows:

$$f(s) = \begin{cases} \mu_l e^{-\mu_l (s - m_l)} & \text{if } s \geq m_l \\ 0 & \text{otherwise} \end{cases} \tag{5.8}$$

Since this distribution is geometrically identical to the exponential distribution with rate μ_l except for a shift of m_l, its memoryless property holds for $s \geq m_l$. However, since this model allows for multiple classes and the shift terms are not the same for each cluster, this model technically should be reclassified as a $M/G/c$ model. Regardless of this, the mean service time for spells in cluster l is given by:

$$\mathbb{E}(S_l) = \int_{m_l}^{\infty} \mu_l s e^{-\mu_l (s - m_l)} \mathrm{d}s = m_l + \frac{1}{\mu_l} \tag{5.9}$$

5.3.2 Validating the model

One of the few ground truths available in the provided data is the observed length of stay distribution. Given that the length of stay and resource availability are connected, the approach here will be to simulate the length of stay distributions for a range of values p_l and c, to find the parameters that best match the observed data.

Several methods are available for the statistical comparison of two or more distributions, such as the Kolmogorov-Smirnov test, a variety of discrepancy approaches such as summed mean-squared error, and f-divergences. A popular choice among the last group (which may be considered distance-like) is the Kullback-Leibler divergence which measures relative information entropy from one probability distribution to another [209]. A key issue with many of these methods is that they lack interpretability, something which is paramount when conveying information to stakeholders, not only for explaining how something works, but also how its results may be explained.

As such, a reasonable candidate is the (first) Wasserstein metric, also known as the

'earth mover' or 'digger' distance [365]. The Wasserstein metric satisfies the conditions of a formal mathematical metric — like Euclidean distance or the dissimilarity measure given in Definition 4.1. Also, the values of the Wasserstein metric take the units of the distributions under comparison (in this case: days). These characteristics can aid understanding and explanation. The distance measures the approximate 'minimal work' required to move between two probability distributions where 'work' can be loosely defined as the product of how much of the distribution's mass moves and the distance by which it must be moved. More formally, the Wasserstein distance between two probability distributions U and V is defined as:

$$W(U,V) = \int_0^1 \left|F^{-1}(t) - G^{-1}(t)\right| dt \tag{5.10}$$

Here, F and G are the cumulative density functions of U and V, respectively. A proof of (5.10) is presented in [295].

Each trial used here takes a parameter set and simulates the ward across a series of independent repetitions. The parameter set with the smallest maximum distance between the simulated system time distribution and the observed length of stay distribution is taken to be the most appropriate. To be specific, let $T_{c,p}$ denote the system time distribution obtained from a simulation with c servers and $p := (p_0, p_1, p_2, p_3)$, and let T denote the observed length of stay distribution. Then the optimal parameter set (c^*, p^*) is given by:

$$(c^*, p^*) = \arg\min_{c,p} \left\{\max\left\{W\left(T_{c,p}, T\right)\right\}\right\} \tag{5.11}$$

The parameter sweep included values of each p_l from 0.5 to 1.0 with a granularity of 5.0×10^{-2} and values of c from 30 to 50 at steps of five. These choices were informed by the assumptions of the model and formative analysis to reduce the parameter space given the computational resources required to conduct the simulations. Each parameter set was repeated 50 times, with each simulation running for four years of virtual time. The warm-up and cool-down periods were taken to be approximately one year each, leaving two years of simulated data from each repetition.

The results of this parameter sweep are summarised in Figures 5.11 through 5.13. Each plot shows a comparison of the observed lengths of stay across all groups and the newly simulated data with the best, median and worst parameter sets, respectively. These figures highlight the importance of choosing good parameters under this model as the differences in the quality of the fits are stark. In the best case the fit is uncanny, whereas the median case shows a distribution that inflates the pres-

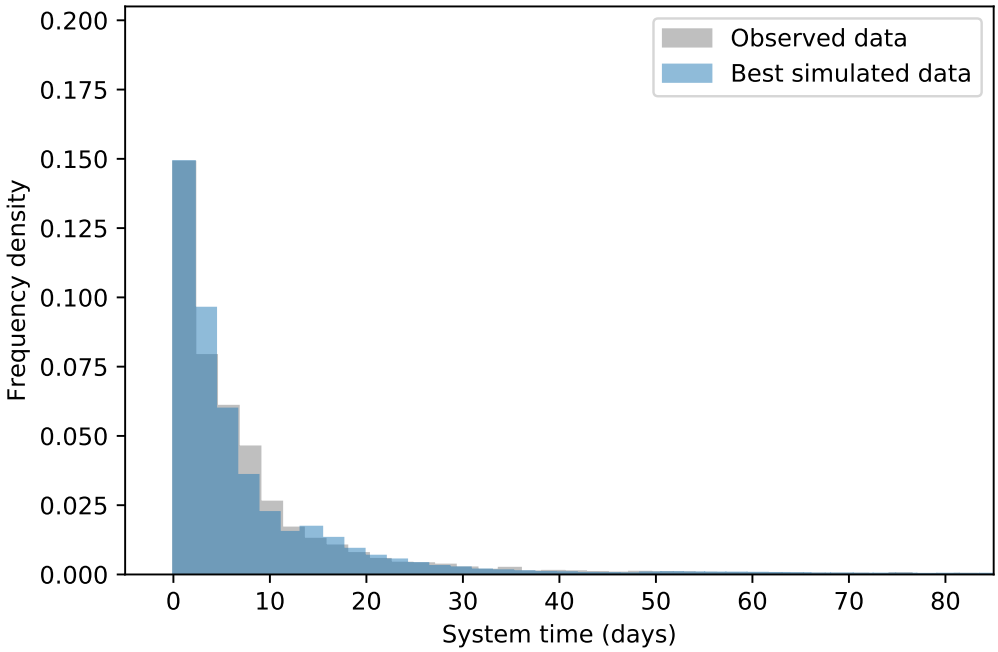


Figure 5.11: Histograms of the best-simulated and observed LOS data

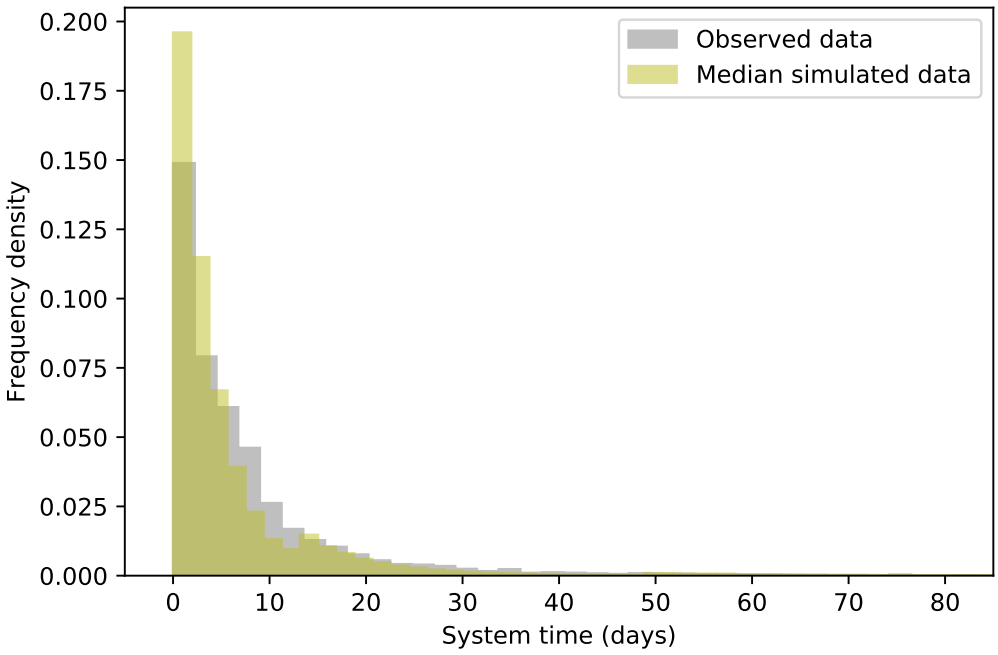


Figure 5.12: Histograms of the median-simulated and observed LOS data

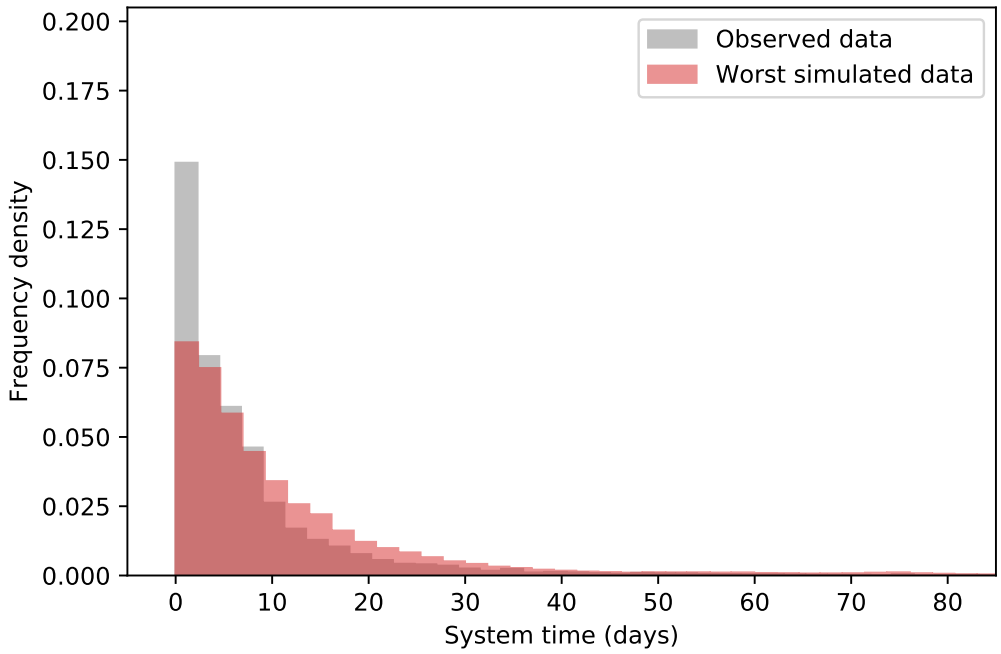


Figure 5.13: Histograms of the worst-simulated and observed LOS data

ence of short-stay patients despite an otherwise good fit. Meanwhile, Figure 5.13 displays a distribution that only resembles the observed distribution in its positive skew; the worst-case distribution lacks the distinctive 'exponential' nose and has a considerably heavier tail corresponding to a disproportionate amount of long-stay patients. Table 5.2 reinforces these results numerically, showing a precise fit by the best parameter set across all measures, except the maximum recorded stay.

In this section, the previously identified clustering enriched the overall queuing model and was used to recover the parameters for several classes within that. Now, using this model, the next section details an investigation into the underlying system by adjusting the parameters of the queue with the clustering.

5.4 Adjusting the queuing model

This section comprises several what-if scenarios — a classic component of healthcare operational research — under the novel parameterisation of the queue established in Section 5.3. The outcomes of interest in this work are server (resource) utilisation and system times. These metrics capture the driving forces of cost and the state of the system. Specifically, the objective of these experiments is to address the following questions:

- How is the system affected by a change in overall patient arrivals?

		Observed	Best simulated	Median simulated	Worst simulated
Model characteristic	p_0	NaN	0.80	0.70	1.00
	p_1	NaN	1.00	0.55	1.00
	p_2	NaN	1.00	0.85	0.95
	p_3	NaN	0.85	0.70	0.90
	c	NaN	35.00	40.00	30.00
	Max. distance	0.00	0.68	1.95	44.25
LOS statistic	**Mean**	7.70	7.56	6.23	11.56
	Std.	11.86	11.44	10.45	14.81
	Min.	0.00	0.00	0.00	0.00
	25%	1.49	1.60	1.16	3.00
	Med.	4.20	3.90	2.90	6.90
	75%	8.93	8.81	6.54	14.21
	Max.	224.93	219.92	187.78	230.49

Table 5.2: A comparison of the observed and simulated data based on the model parameters and summary statistics for length of stay

- How is the system affected by a change in resource availability?
- How is the system affected by patients moving between clusters?

Given the nature of the observed data, the queuing model parameterisation and its assumptions, the effects on the chosen metrics in each scenario are in relative terms with respect to the base case. The base case being those results generated from the best parameter set recorded in Table 5.2. In particular, the data from each scenario is scaled by the corresponding median value in the base case, meaning that a metric having a value of 1 is 'normal'.

As mentioned in Section 5.1, the source code used throughout this chapter has been archived online under doi:10.5281/zenodo.4457902. Also, the datasets generated from the simulations in this section, and the parameter sweep, have been archived online doi:10.5281/zenodo.4457808.

5.4.1 Changes to overall patient arrivals

Changes in overall patient arrivals to a queue reflect real-world scenarios where some stimulus is improving (or worsening) the condition of the patient population. Examples of stimuli could include an ageing population or independent life events that lead to a change in deprivation, such as an accident or job loss. Within this model, overall patient arrivals are altered using a scaling factor denoted by $\sigma > 0$. This scaling factor is applied to the model by multiplying each cluster's arrival rate by σ. That is, the new arrival rate for a cluster, l, denoted $\hat{\lambda}_l$, is given by:

$$\hat{\lambda}_l = \sigma \lambda_l \tag{5.12}$$

Figure 5.14 shows the effects of changing patient arrivals on (a) relative system times and (b) relative server utilisation for values of σ from 0.5 to 2.0 at a precision of 1.0×10^{-2}. Specifically, each plot in the figure (and the subsequent figures in this section) shows the median and interquartile range (IQR) of each relative attribute. These metrics provide an insight into the experience of a typical user (or server) in the system. Furthermore, they reveal the stability and variation of the body of users (or servers).

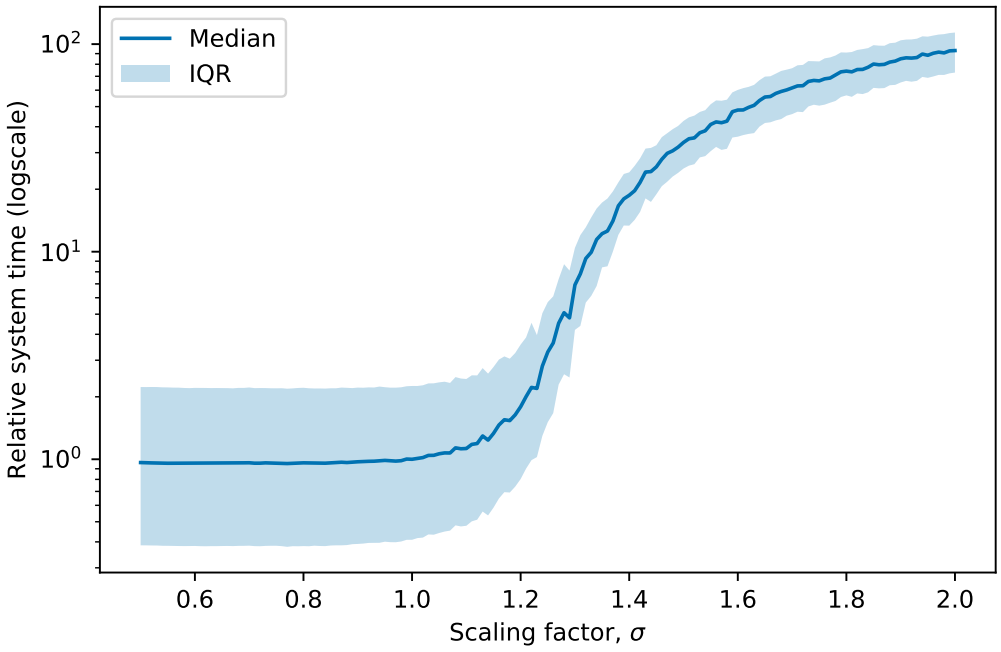
What is evident from these plots is that things are happening as one might expect: as arrivals increase, the strain on the system increases. However, it should be noted that it also appears that the model has some amount of slack relative to the base case. Looking at Figure 5.14a, for instance, the relative system time distribution stays unchanged up to $\sigma \approx 1.2$, or an approximate 20% increase in arrivals of COPD patients. Beyond that, relative system times quickly rise to an untenable point where the median time becomes orders of magnitude above the norm.

However, Figure 5.14b shows that the situation for the system's resources reaches its worst-case near to the start of that spike in relative system times (at $\sigma \approx 1.3$). That is, the median server utilisation reaches a maximum (this corresponds to constant utilisation) at this point, and the variation in server utilisation disappears entirely. The reality of this situation is that the system has no slack at all, and all parts of the system are under equal load, which is not preferable given the differences in resource requirements for the parts of a hospital system. For instance, if surgical theatres were in constant use but administrative processing required an equivalent amount of resources to continue running, the system would likely falter or deteriorate entirely.

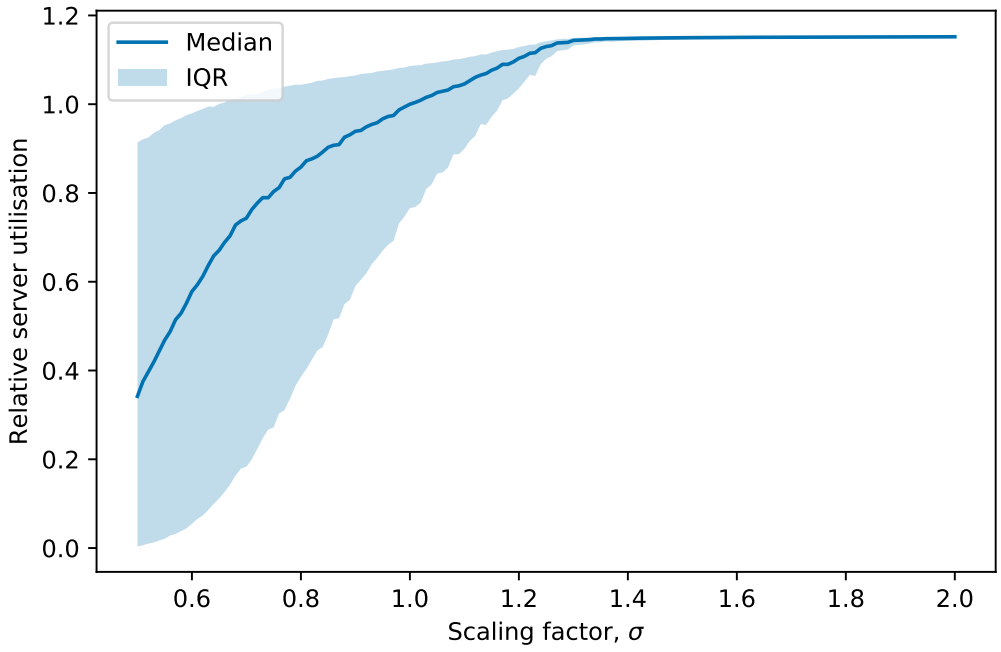
5.4.2 Changes to resource availability

As is discussed in Section 5.3, the resource availability of the system is captured by the number of parallel servers, c. Therefore, to modify the overall resource availability, only the number of servers needs to be changed. This kind of sensitivity analysis is usually done to determine the opportunity cost of adding service capacity to a system, e.g. would an increase of n servers sufficiently increase efficiency without exceeding a budget?

To reiterate the beginning of this section: all suitable parameters are given in relative terms, including the number of servers here. By doing this, the changes in resource



(a)



(b)

Figure 5.14: Plots of σ against relative (a) system time and (b) server utilisation

availability are more evident, and do away with any concerns as to what a particular number of servers precisely reflects in the real world, be it any combination of hospital beds, equipment availability and medical staff.

Figure 5.15 shows how the relative resource availability affects relative system times and server utilisation. In this scenario, the relative number of servers took values from 0.5 to 2.0 at an equivalent step size of one in the number of servers, i.e. c takes values from 17 to 70. Overall, these figures fortify the claim from the previous scenario that there is some room to manoeuvre so that the system runs 'as normal', but pressing on those boundaries results in massive changes to both resource requirements and system times.

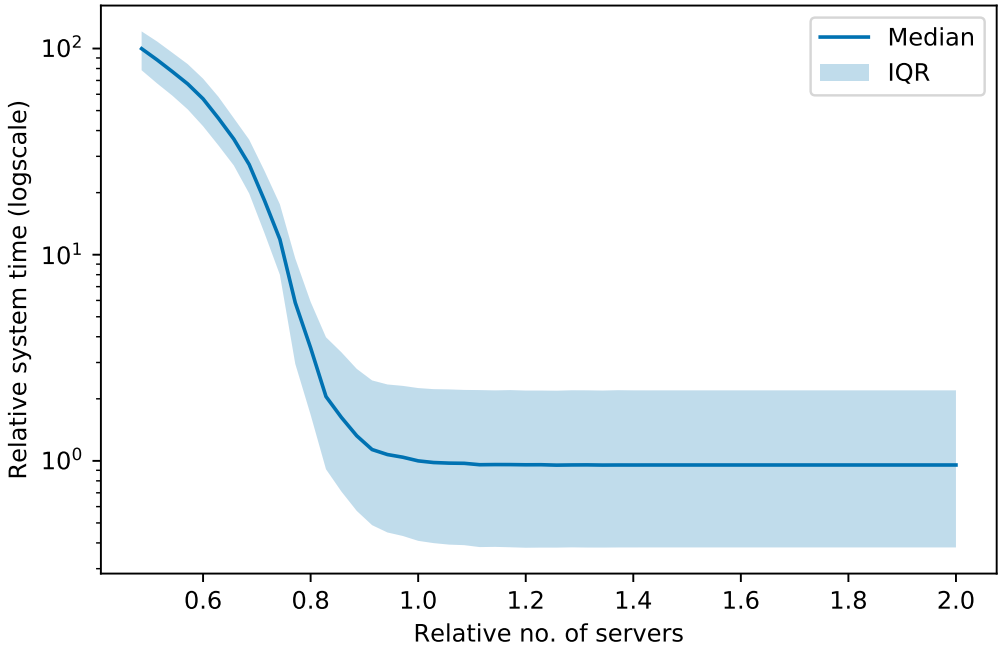
In Figure 5.15a this amounts to a maximum of 10% slack in resources before relative system times are substantially affected; further reductions quickly result in a potentially tenfold increase in the median system time, and up to 100 times once resource availability falls by 50%. Moreover, the variation in the body of the relative times (i.e. the IQR) decreases as resource availability decreases. The reality of this is that patients arriving at a hospital are forced to consume more significant amounts of resources (by merely being in a hospital) regardless of their condition, putting added strains on the system. Figure 5.15b mirrors these observations on the small amount of slack in resource requirements, but (as with the previous scenario) constant utilisation occurs quickly.

Meanwhile, it appears that there is no tangible change in relative system times given an increase in the number of servers. This indicates that the model carries sufficient resources to cater to the population under normal circumstances and that adding service capacity will not necessarily improve system times.

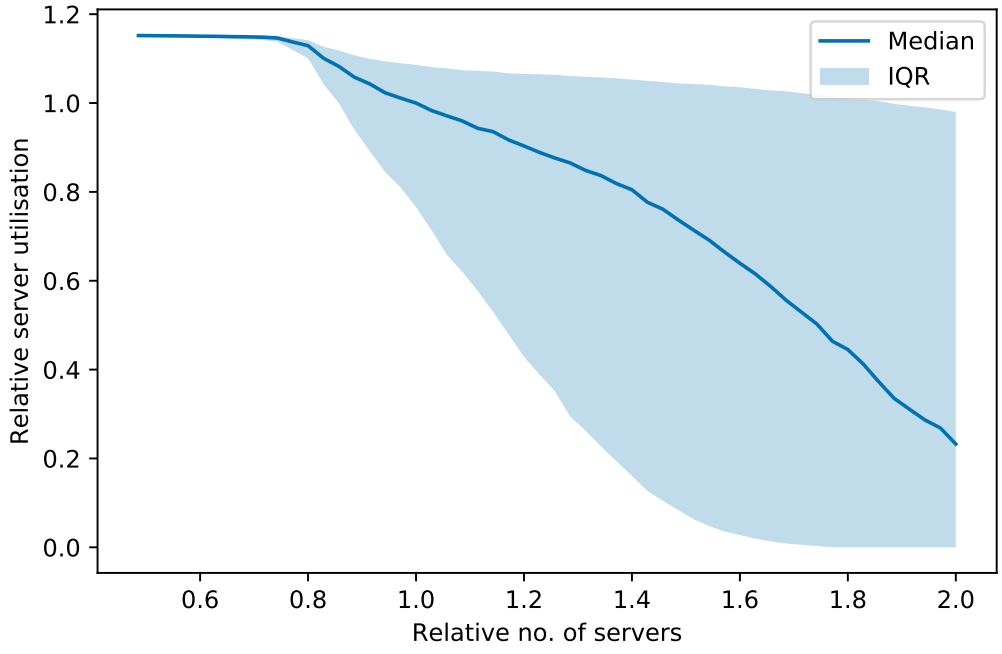
Again, Figure 5.15b shows that there is a substantial change in the variation in the relative utilisation of the servers. In this case, the variation dissipates as resource levels fall, and increases as resources increase. While the relationship between real hospital resources and the number of servers is not exact, having variation in server utilisation would suggest that small parts of an existing system may be configured or partitioned away in the case of some significant public health event (such as a global pandemic) without overloading the system.

5.4.3 Moving arrivals between clusters

This scenario is perhaps the most relevant to actionable public health research of those presented here. The clusters identified in this chapter could be characterised by their clinical complexities and resource requirements, as done in Section 5.1.1. Therefore, being able to model the movement of some proportion of patient spells



(a)



(b)

Figure 5.15: Plots of the relative number of servers against relative (a) system time and (b) server utilisation

from one cluster to another will reveal how those complexities and requirements affect the system itself. The reality is then that if some public health policy could be implemented to initiate that movement informed by a model such as this, then change would be seen in the real system.

In order to model the effects of spells moving between two clusters, the assumption is that each cluster's service time distribution stays the same (and so does each cluster's p_l), but their arrival rates are altered according to some transfer proportion. Consider two clusters indexed at l and m, and their respective arrival rates, λ_l, λ_m. Let $\delta \in [0, 1)$ denote the proportion of arrivals to be moved from cluster l to cluster m. Then the new arrival rates for each cluster, denoted by $\hat{\lambda}_l, \hat{\lambda}_m$ respectively, are:

$$\hat{\lambda}_l = (1-\delta)\,\lambda_l \quad \text{and} \quad \hat{\lambda}_m = \delta\,\lambda_l + \lambda_m \tag{5.13}$$

By moving patient arrivals between clusters in this way, the overall arrivals are left the same since the sum of the arrival rates is the same. Hence, the (relative) effect on server utilisation and system time can be measured independently.

Figures 5.16 and 5.17 show the effect on relative system time and relative server utilisation, respectively, of moving patient arrivals between clusters. In each figure, the median and IQR for the corresponding attribute is shown, as in the previous scenarios. Each scenario was simulated using values of δ from 0.0 to 0.98 at steps of 2.0×10^{-2}.

Considering Figure 5.16, it appears that each type of transfer falls into one of two categories: either completely derailing the system (such as moving any cluster to Cluster 3) or improving system times, albeit mildly. The latter case occurs in the following transfers:

- Cluster 0 to Clusters 1 or 2
- Cluster 1 to Cluster 2
- Cluster 3 to any other cluster

A finer look at the effect of these transfer types on relative system times is given in Table 5.3. Likewise, their effects on relative server utilisation is given in Table 5.4.

The message delivered by these transfers is that in order to improve system times in hospitals, the only solution is for the patients arriving at hospital to present with fewer resource requirements. Meanwhile, the complexity of their condition is less influential. Achieving such reductions in resource requirements is certainly no mean feat, but could be addressed by investing in more advanced medical infrastructure in

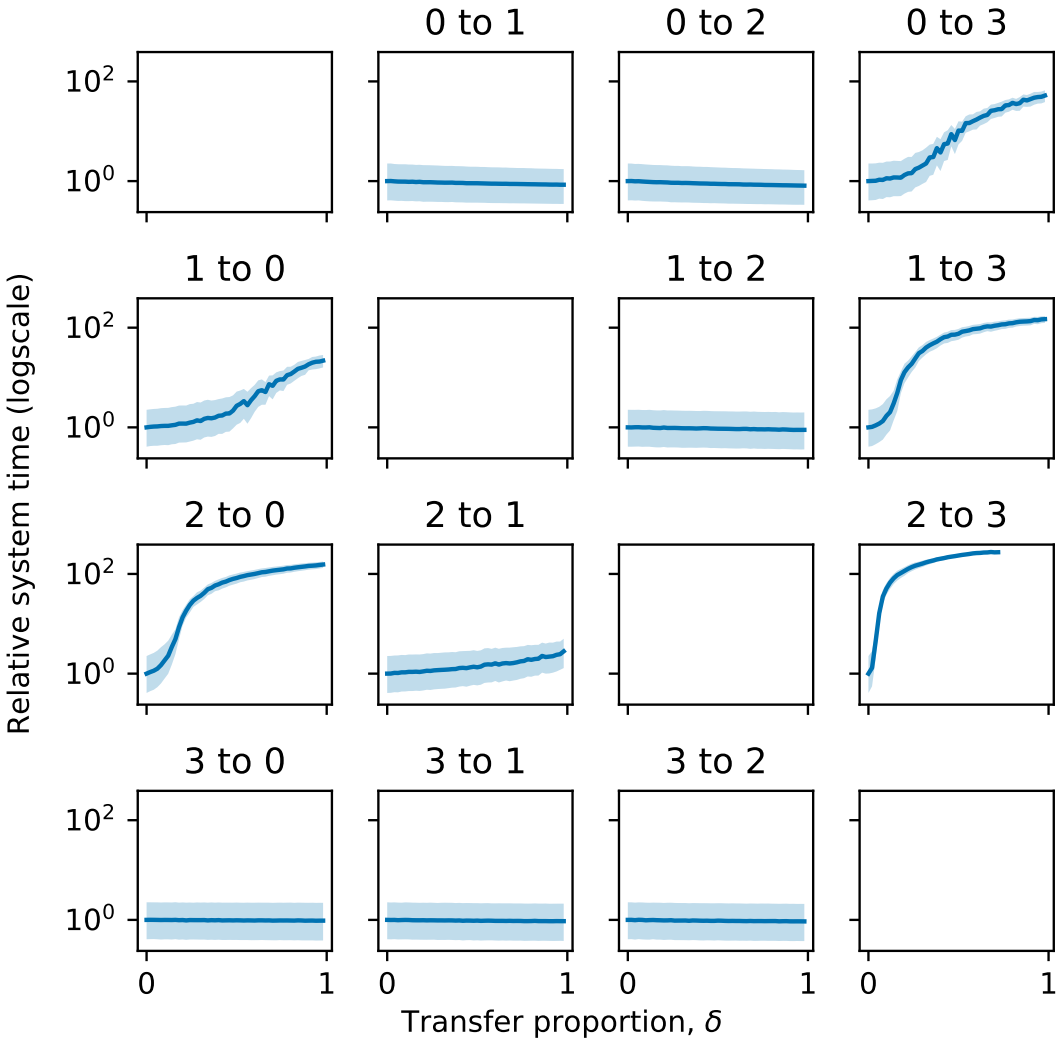


Figure 5.16: Plots of proportions of each cluster moving to another against relative system time

	δ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Origin	**Destination**										
0	1	0.0	-0.0251	-0.0511	-0.0657	-0.0794	-0.0944	-0.1117	-0.1230	-0.1357	-0.1484
	2	0.0	-0.0287	-0.0556	-0.0841	-0.1034	-0.1214	-0.1354	-0.1527	-0.1663	-0.1789
1	2	0.0	-0.0048	-0.0072	-0.0393	-0.0452	-0.0606	-0.0762	-0.0761	-0.0909	-0.1058
3	0	0.0	-0.0024	-0.0066	-0.0111	-0.0102	-0.0186	-0.0292	-0.0333	-0.0292	-0.0325
	1	0.0	-0.0021	-0.0156	-0.0229	-0.0257	-0.0327	-0.0443	-0.0486	-0.0521	-0.0583
	2	0.0	-0.0182	-0.0242	-0.0298	-0.0365	-0.0337	-0.0487	-0.0554	-0.0530	-0.0646

Table 5.3: Proportional changes in median relative system time for selected cluster transfers

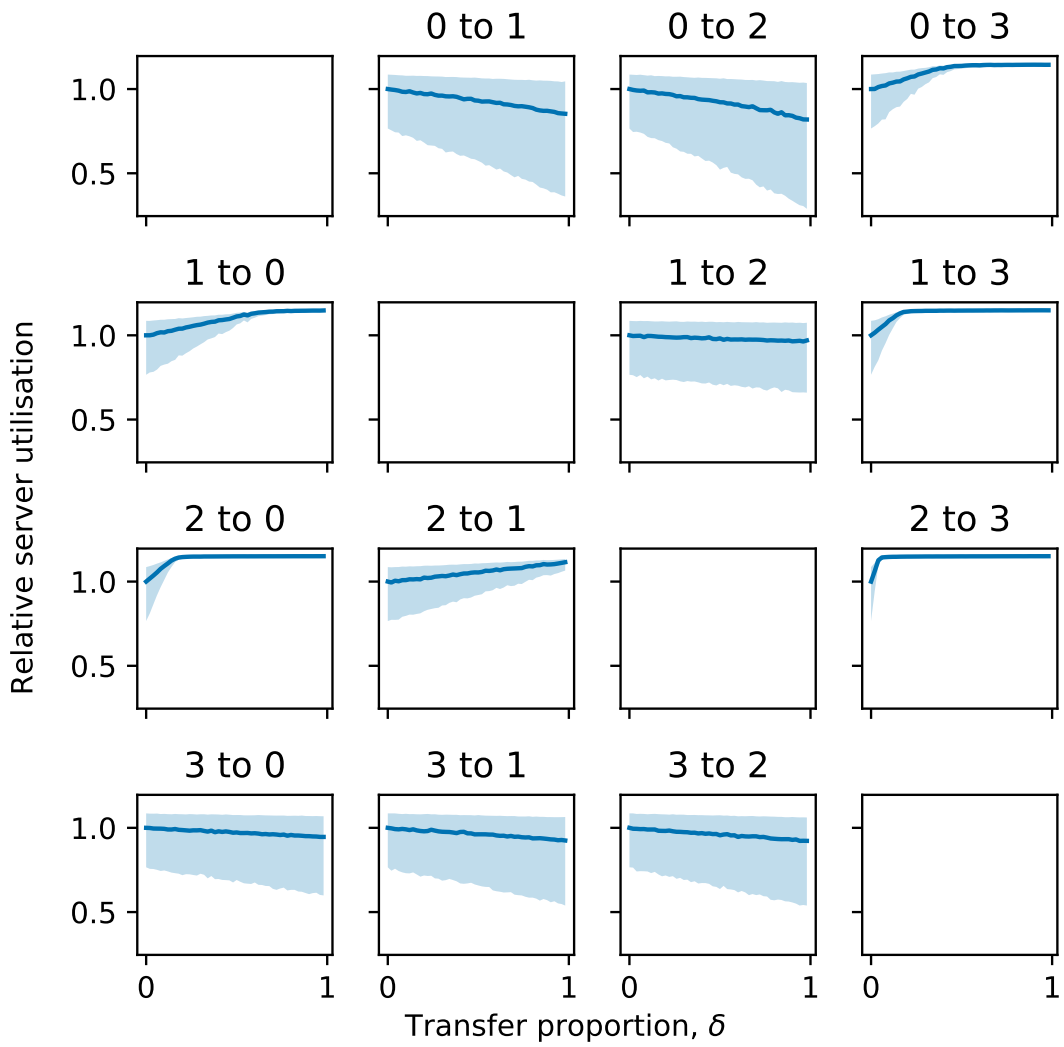


Figure 5.17: Plots of proportions of each cluster moving to another on relative server utilisation

	δ	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
Origin	**Destination**										
0	1	0.0	-0.0176	-0.0299	-0.0391	-0.0535	-0.0693	-0.0824	-0.1001	-0.1129	-0.1325
	2	0.0	-0.0197	-0.0290	-0.0488	-0.0627	-0.0782	-0.0919	-0.1140	-0.1384	-0.1592
1	2	0.0	-0.0035	-0.0108	-0.0108	-0.0180	-0.0181	-0.0249	-0.0256	-0.0302	-0.0357
3	0	0.0	-0.0060	-0.0132	-0.0137	-0.0206	-0.0274	-0.0320	-0.0380	-0.0422	-0.0494
	1	0.0	-0.0089	-0.0206	-0.0232	-0.0246	-0.0384	-0.0451	-0.0532	-0.0626	-0.0685
	2	0.0	-0.0100	-0.0184	-0.0254	-0.0314	-0.0443	-0.0542	-0.0504	-0.0649	-0.0714

Table 5.4: Proportional changes in median relative utilisation for selected cluster transfers

other parts of the healthcare system, beyond hospitals. Furthermore, this could be achieved by implementing some preventive policy that would help improve the overall health of the COPD population, with particular targeting for those most-affected by the condition.

Conversely, the concern arises when either of the low resource requirement clusters moves to Cluster 0 or Cluster 3. Even as few as one in ten of the low-complexity, low-resource-needs arrivals in Cluster 2 moving to either cluster results in large jumps in the median system time for all arrivals. Soon after, as in the previous scenario, any variation in the system times disappears, indicating an overborne system.

With relative server utilisation, the story is much the same. The ordinary levels of high-complexity, high-resource arrivals from Cluster 3 are absorbed by the system and moving these arrivals to another cluster bears little effect on resource consumption levels. Likewise, either of the low-resource needs clusters moving even slightly toward high resource requirements completely overruns the system's resources. However, the relative utilisation levels of the system resources can be substantially reduced by moving arrivals from Cluster 0 to either Cluster 1 or Cluster 2, i.e. by reducing the overall resource requirements of such spells.

In essence, this entire analysis offers two messages. Firstly, that there are several ways in which the system can get worse and even overwhelmed. Secondly, and more importantly, that any meaningful impact on the system must come from a stimulus outside of the system that results in a higher proportion of healthy patients arriving at the hospital. This conclusion is non-trivial; the first two scenarios in this analysis show that there are no quick solutions to reduce the effect of COPD patients on hospital capacity and length of stay. The only effective intervention for improving the system on the whole is found through inter-cluster transfers.

5.5 Chapter summary

This chapter presents a novel approach to investigating a healthcare population that encompasses the topics of segmentation analysis, queuing models, and the recovery of queuing parameters from incomplete data. This investigation is done despite characteristic limitations in operational research concerning the availability of fine-grained data, and this chapter only uses administrative hospital spell data from patients presenting COPD from Cwm Taf Morgannwg UHB.

By considering a variety of attributes present in the data, and engineering some, a useful clustering of the spell population is identified that successfully feeds into a multi-class $M/G/c$ queue to model a hypothetical COPD ward. This clustering

was generated using the initialisation presented in Chapter 4, which in turn was effectively evaluated using the EDO method from Chapter 3. The culmination of these three features from this thesis fulfil its objective: to utilise machine learning through creation, evaluation and, finally, application.

With this model, several insights are gained by investigating purposeful changes in the parameters of the model that have the potential to inform actual public health policy. In particular, since neither the resource capacity of the system nor the clinical processes of the spells are evident in the data, service times and resource levels are not available. However, the length of stay is. Using what is available, this chapter assumes that mean service times can be parameterised using mean lengths of stay. By using the Wasserstein distance to compare the distribution of the simulated lengths of stay data with the observed data, a best performing parameter set is found via a parameter sweep.

This parameterisation ultimately recovers a surrogate for service times for each cluster, and a universal number of servers to emulate resource availability. The parameterisation itself offers its strengths by being straightforward and effective. Despite its simplicity, a good fit to the observed data is found, and — as is evident from the closing section of this chapter — substantial and useful insights can be gained into the needs of the population under study.

This mode of analysis, in effect, considers all types of patient arrivals and how they each impact the system in terms of resource capacity and length of stay. By investigating changes in both overall patient arrivals and resource capacity, it is clear that there is no quick solution to be employed from within the hospital to improve COPD patient spells. The only effective, non-trivial intervention is to improve the overall health of the patients arriving at the hospital, as is shown by moving patient arrivals between clusters. In reality, this would correspond to an external, preventive policy that improves the overall health of COPD patients.

Chapter 6

Conclusions

This chapter serves to summarise and reflect on the work reported in this thesis. The summaries here are deliberately brief since each chapter concludes with a detailed summary. In addition to these summaries, this chapter outlines the contributions to literature made by this thesis, and describes some potential avenues for further work.

6.1 Research summary

Chapter 1 described the research questions associated with this thesis, laying out its principle subjects of algorithm evaluation, clustering, and operational healthcare modelling. With this last subject, there was a particular interest in overcoming a common issue with machine learning applications in healthcare: not necessarily having sufficiently detailed and voluminous data with which to create meaningful, actionable models.

Chapter 2 presented a survey of the literature spanning these principle topics and their intersections. Motivated by the apparent gaps in the collated literature, the subsequent chapters of the thesis presented novel methods for assessing the quality of an algorithm (or algorithms), and for incorporating mathematical fairness into an existing clustering algorithm. These methods later fed into the case study for Cwm Taf Morgannwg UHB which characterised, analysed and modelled a subsection of their patient population.

In Chapter 3, a new paradigm by which algorithms may be assessed was described, and a method from that paradigm presented. This method, known as evolutionary dataset optimisation (EDO), explores the space in which 'good' datasets exist for an algorithm according to some metric. This exploration is achieved via a bespoke

evolutionary algorithm which acts on datasets of unfixed shapes, sizes and data types. The chapter presented descriptions and illustrations of the internal mechanisms of the EDO method, as well as briefly describing a Python implementation. Finally, the chapter concluded with an extensive case study, demonstrating the capabilities and nuances of EDO in gaining a richer picture of an algorithm's abilities independently, and against a competitor.

Following the discussion of 'fair' machine learning practices in the literature review, Chapter 4 offered a novel initialisation to the k-modes algorithm which made use of game theory. The new initialisation extended a commonly used method, but replaced its greedy component with a solvable matching game. In the evaluative section of this chapter, traditional assessment techniques suggested that the new method improved upon the original, and so the original was discarded.

However, the new method did not consistently outperform another well-known initialisation. To better understand the conditions under which either of the remaining initialisations would succeed, a similar setting to Chapter 3 was used. This analysis revealed that there were distinct sets of properties for which one method was more likely to succeed than the other according to the metric under study.

Chapter 5 presented a novel framework with which to model the resource needs of a condition-specific healthcare population — despite a lack of fine-grained data. In this case, that population were those suffering from COPD. The corresponding dataset, provided by Cwm Taf Morgannwg UHB, consisted of high-level, administrative details about the spells associated with the patients, and lacked the depth that many contemporary operational models require.

The presented framework utilised the clustering algorithm described in Chapter 4 to segment a subset of existing and engineered attributes in the dataset. These attributes included hospital utilisation metrics, and proxy measures of clinical complexity and resource needs. The segmentation successfully characterised the instances of the dataset, and the ensuing analysis of the identified segments revealed clear profiles for each segment. Included in these profiles were distinctly shaped distributions for length of stay. With an aim to extract as much as possible from the available data, and to provide further practical insights, these distributions were utilised to construct a multi-class queuing model.

The queue, although minimal in structure, produced a well-fitting replica of the true lengths of stay observed in the data. The quality of this model was dependent on a novel parameterisation, which derived the unknown service time distributions for each cluster from the data according to the Wasserstein distance. In turn, this model was adjusted to answer several 'what-if' scenarios associated with changes in resource

capacity and requirements for the population under study. These adjustments revealed actionable insights into the most-impactful segments of the population. The most important of these results was demonstrating the futility of attempting to implement quick, blanket solutions for that population, such as only increasing resource capacity without improving patient well-being.

6.2 Contributions

This thesis has made novel contributions across each of its three principal themes: algorithm evaluation, clustering, and healthcare modelling. This section summarises these contributions with reference to their respective chapters.

The EDO method introduced in Chapter 3 provided an example approach from a novel paradigm in which the objective performance of algorithms can be assessed by exploring the space in which 'good' or 'bad' datasets exist. The proposed paradigm expands the commonly used approach for evaluation where a method's quality is 'confirmed' by taking a small number of benchmark datasets and comparing them with its contemporaries. By exploring the space of datasets, it was demonstrated that a more robust assessment can be made of a method — or set thereof.

Chapter 4 added to the growing body of literature where game-theoretic concepts are combined with machine learning techniques, of which clustering is included. In general, pursuits of this kind reformulate existing techniques to be mathematically fair. The initialisation presented in Chapter 4, instead, incorporated game theory directly into an existing algorithm. In doing so, an improvement over the existing method was shown, using both traditional confirmation processes and the EDO method.

Supplementing the research reported in Chapters 3 and 4 are two free-standing software packages, `edo` and `matching`. Descriptions of these packages and their locations are as follows:

- The `edo` library comprises a Python implementation of the EDO method.
  - GitHub repository: `github:daffidwilde/edo`
  - Zenodo archive: `doi:10.5281/zenodo.2552890`
  - Documentation: `edo.readthedocs.io`
- The `matching` package provides a framework for facilitating and solving various matching games.
  - GitHub repository: `github:daffidwilde/matching`

- Zenodo archive: doi:10.5281/zenodo.2553125
- Documentation: matching.readthedocs.io

The framework used in Chapter 5 contributed to healthcare modelling literature in three ways. First, the estimation of queuing parameters via the Wasserstein distance has expanded a relatively scarce area of queuing research. Second, by making COPD the subject of the methodology, the framework has added to a body of literature surrounding a condition that is vital to understand given its prevalence, as well as its links to deprivation and comorbidity. Lastly, the framework provided a solution to the common issue of data availability in modern operational research. By combining the various individual methods, valuable insights were extracted from a relatively unsophisticated data source, which is a result seldom seen in operational research.

In addition to the work directly included in the chapters of this thesis, the research associated with this thesis has resulted in the production of numerous auxiliary research items. These include several well-developed pieces of research software — two of which have been listed here, while the rest are summarised at the end of Dissemination) — and a number of useful, publicly available datasets (listed in Table 1.1).

6.3 Reflections on research direction

The original objective of this thesis was to utilise machine learning to better understand variability in the NHS. Driven by the needs of the co-sponsors of this project, Cwm Taf Morgannwg UHB, the hope was to apply some technique(s) from machine learning to reveal insights into the patient population within their hospital system.

With an administrative dataset provided by Cwm Taf Morgannwg UHB, exploratory analysis (available in Appendix B) found that the population in question was deeply varied and heterogeneous — as expected. However, the dataset was insufficiently detailed to construct meaningful models of the entire population or system. The variety in the data opened up an interest in population segmentation techniques, eventually feeding into Chapters 4 and 5. Meanwhile, requests were made for larger, more detailed datasets — so that contemporary, machine learning techniques could be applied more readily — and the gathering of literature began. Even with the expansive nature of machine learning literature, two clear patterns emerged.

First, the vast majority of publications that introduced a novel machine learning method contained a boilerplate evaluation section. In each article, the proposed method would be pitted against a few of its contemporaries by comparing a handful

of metrics on a handful of datasets. Typically, these metrics and datasets would be taken from a small pool which was relevant to that technique; this makes sense. There are appropriate measures for various techniques, and it is important to make comparisons relative to a fixed point.

This approach to algorithm evaluation exhibited two issues: one, a lack of diversity in the resources used to assess algorithm performance; and, two, an incongruence between the power of the evaluation process and the conclusions drawn from that process. Often, a method would be deemed 'state-of-the-art' or 'better' based exclusively on a process that offered little insight into the actual quality of that method. This reliance on a narrow assessment process prompted research into how else an algorithm could be evaluated objectively, directly leading to the work in Chapter 3.

The second observation was that the ethics of data and machine learning algorithms were being discussed, but the discourse appeared separately from the machine learning publications. An exception to this was the development of fair machine learning practices. These methods consider new formulations and objectives based on mathematical fairness, a concept either derived from or with common roots in game theory. The reinventing of techniques and paradigms to be fair raised questions about how far an addition of some game theory could improve the performance of an existing algorithm, as opposed to creating an entirely new one, leading to the work in Chapter 4.

Over the course of this project, it became clear that the requests for more detailed datasets would not be completed in time. Hence, the research methodology would have to prioritise extracting as much useful information as possible from the data at hand. Simultaneously, Cwm Taf Morgannwg UHB provided further administrative datasets which related to the members of their population who suffer from the respiratory condition, COPD. An internal report by NHS Wales found that Cwm Taf Morgannwg UHB had the highest prevalence of the condition out of all the Welsh health boards. Given the known links between COPD, socioeconomic deprivation and the coincidence of multiple comorbidities, the objective of this project was revised to focus on understanding the needs of that population.

Overcoming each of these challenges culminated in the methodology presented in Chapter 5. The high-level, administrative dataset was analysed and processed using machine learning to extract distinct profiles within the COPD population. These profiles fed into a classic operational research method — queuing — to provide a rich, insightful model of the population under study, and its needs.

6.4 Further work

EDO as a data synthesiser

As demonstrated in the case study in Chapter 3 and the closing section of Chapter 4, the EDO method is capable of facilitating richer insights into an algorithm's performance. Having said that, a limitation of the method is that there is no standardised way to guarantee relationships between different columns in a dataset, or the families passed to EDO, $\mathcal{P}$. Currently, the only way to do this is to include measures of the desired relationships in the fitness function. Given the success in the chapters of this thesis, this level of control is not necessary when looking at an algorithm (or algorithms) in a general sense, and so is considered beyond the scope of this thesis.

However, there are cases where automatically ensuring the relationship between the elements of $\mathcal{P}$ could be beneficial to a user of EDO. For instance, if the algorithm of interest is bespoke to a particular task or dataset. Using EDO in this way would be analogous to synthesising an existing dataset, which is another example of when this would be useful. In such a scenario, it may be beneficial to capture the essence of a dataset by loosely fitting the elements of $\mathcal{P}$ to the existing dataset. Fitting the parameters of the distribution families would be relatively straightforward, but incorporating the relationships between them is less so.

This capability has been one of the major attractions of using GANs for data synthesis, but their black-box nature defeats the object of EDO. Another option is to use copulas. Copulas are functions that join multivariate distribution functions to their one-dimensional margins [258]. For EDO, this would mean $\mathcal{P}$ would contain a single element: a copula function fitted to the existing dataset. In this case, the technical aspects of an individual's representation would need adjusting to accommodate this change. Likewise, the crossover and mutation processes would require some changes to account for the lack of distinct distribution families.

A Python implementation of copulas for data synthesis exists [13], and incorporating this as a dependency of the `edo` library would reduce the work required to implement this feature. Studying the impact of copulas in EDO would provide a valuable opportunity to demonstrate the capabilities of EDO as a fully fledged data synthesis method.

Expanding the COPD queuing framework

As discussed at various points in this thesis, the framework presented in Chapter 5 is novel in its ability to circumvent the need for fine-grained data. However, as discussed in Section 6.2, there are other aspects to its novelty such as the use of clustering

to inform a queuing model, and the estimation of unknown queuing parameters. Extending the reach of this work into the COPD population would be possible with even slightly more detailed data. For instance, episode-level data (such as the dataset analysed in Appendix B) could allow for a queuing network with multiple nodes to be developed, separating the various departments in the hospital. However, that data would need to be well-ordered to understand the actual pathway of patients at the spell level, which routinely gather administrative datasets are not.

Cwm Taf Morgannwg UHB, in partnership with Swansea University, has been developing a new system for recording the clinical activity and vital information of their patients in real time [86]. This system replaces the physical whiteboards in hospital wards with an electronic equivalent. The 'e-whiteboard' and its drag-and-drop software overcomes some of the issues associated with traditional whiteboards such as the accurate recording of data to the existing electronic system. In addition, the internal software records the exact time that information is recorded, allowing for an extremely high level of detail in terms of the processes undergone by patients. Access to such a data source would certainly open up more sophisticated models, including both the clustering and queuing aspects of the framework used in Chapter 5.

Weighted student-project allocation

In tandem with the work presented in Appendix C, another school at Cardiff University expressed an interest in implementing a matching-based allocation for their final year student projects. The attraction of using matching games was the mathematical fairness of its solution when compared with their current allocation process. However, their final year students are of two classes: those on a three-year course and those on a four-year course. Projects for shorter courses are worth fewer credits and require less commitment from supervisors than those for longer courses.

Effectively, this variety equates to the students having different weights. A potential line of research then would be to formulate the weighted student-project allocation problem (WSA). WSA would be a generalisation of the student-project allocation problem (SA) — described in Appendix A — where each student, s, would have a weight associated with them, $w_s > 0$. Then, the size of a project or supervisor matching would be the sum of their students' weights, as opposed to the cardinality of their matching. Under this formulation, an instance of SA could be restated as an instance of WSA where $w_s = 1$ for every student, s.

In addition to the formulation, further work would include adapting the existing Gale-Shapley algorithms for SA to accommodate for student weights, and proving whether those algorithms guarantee a unique, stable matching.

Bibliography

[1] M. Aberdour. Achieving quality in open-source software. *IEEE Software*, 24(1):58–64, 2007. doi:10.1109/MS.2007.2.

[2] D. J. Abraham, R. W. Irving, and D. F. Manlove. The student-project allocation problem. In *Algorithms and Computation*, pages 474–484. Springer Berlin Heidelberg, 2003. doi:10.1007/978-3-540-24587-2_49.

[3] D. J. Abraham, R. W. Irving, and D. F. Manlove. Two algorithms for the student-project allocation problem. *Journal of Discrete Algorithms*, 5(1):73 – 90, 2007. doi:10.1016/j.jda.2006.03.006.

[4] L. M. Abualigah, A. T. Khader, and E. S. Hanandeh. Hybrid clustering analysis using improved krill herd algorithm. *Applied Intelligence*, 48(11):4047–4071, 11 2018. doi:10.1007/s10489-018-1190-6.

[5] M. Ackerman and S. Ben-David. Clusterability: A theoretical study. In D. van Dyk and M. Welling, editors, *Proceedings of the Twelth International Conference on Artificial Intelligence and Statistics*, volume 5, pages 1–8. Proceedings of Machine Learning Research, 2009. URL: http://proceedings.mlr.press/v5/ackerman09a.html.

[6] S. Adeyemi, E. Demir, and T. Chaussalet. Towards an evidence-based decision making healthcare system management: Modelling patient pathways to improve clinical outcomes. *Decision Support Systems*, 55(1):117–125, 2013. doi:10.1016/j.dss.2012.12.039.

[7] N. Agarwal. Policy analysis in matching markets. *American Economic Review*, 107(5):246–50, 2017. doi:10.1257/aer.p20171112.

[8] C. C. Aggarwal and C. K. Reddy. *Data Clustering: Algorithms and Applications*. Chapman & Hall/CRC, 1st edition, 2013.

[9] R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. *SIGMOD Rec.*, 27(2):94–105, 1998. doi:10.1145/276305.276314.

[10] S. Ahmadian, A. Epasto, M. Knittel, R. Kumar, M. Mahdian, B. Moseley, P. Pham, S. Vassilvitskii, and Y. Wang. Fair hierarchical clustering, 2020. arXiv:2006.10221.

[11] C. A. Alexander and L. Wang. Big data and data-driven healthcare systems. *Journal of Business and Management Sciences*, 6(3):104–111, 2018. doi:10.12691/jbms-6-3-7.

[12] I. Aljarah, M. Mafarja, A. A. Heidari, H. Faris, and S. Mirjalili. Clustering analysis using a novel locality-informed grey wolf-inspired clustering approach. *Knowledge and Information Systems*, 62(2):507–539, 2019. doi:10.1007/s10115-019-01358-x.

[13] M. Alvarez, C. Sala, J. D. Pérez, A. Y. Sun, A. Montanez, K. Veeramachaneni, paulolimac, K. A. Zhang, and G. Bonomi. Copulas. URL: https://github.com/sdv-dev/Copulas.

[14] A. Amirjanov. Modeling the dynamics of a changing range genetic algorithm. *Procedia Computer Science*, 102:570 – 577, 2016. doi:10.1016/j.procs.2016.09.444.

[15] Anaconda, Inc. Anaconda. URL: https://www.anaconda.com/.

[16] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander. OPTICS: Ordering points to identify the clustering structure. *SIGMOD Record*, 28(2):49–60, 1999. doi:10.1145/304181.304187.

[17] J. Archenaa and E. M. Anita. A survey of big data analytics in healthcare and government. *Procedia Computer Science*, 50:408–413, 2015. doi:10.1016/j.procs.2015.04.021.

[18] I. V. Arnolds and D. Gartner. Improving hospital layout planning through clinical pathway mining. *Annals of Operations Research*, 263:453–477, 2018. doi:10.1007/s10479-017-2485-4.

[19] D. Arthur and S. Vassilvitskii. *k*-means++: The advantages of careful seeding. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 1027–1035, 2007. URL: http://dl.acm.org/citation.cfm?id=1283383.1283494.

[20] A. Arulselvan, Ágnes Cseh, M. Groß, D. F. Manlove, and J. Matuschke. Matchings with lower quotas: Algorithms and complexity, 2016. arXiv:1412.0325.

[21] V. Arya, N. Garg, R. Khandekar, A. Meyerson, K. Munagala, and V. Pan-

dit. Local search heuristic for k-median and facility location problems. In *Proceedings of the Thirty-Third Annual ACM Symposium on Theory of Computing*, STOC '01, page 21–29. Association for Computing Machinery, 2001. doi:10.1145/380752.380755.

[22] A. Asanjarani, Y. Nazarathy, and P. Pollett. Parameter and state estimation in queues and related stochastic models: A bibliography, 2017 (accessed Friday 9th April, 2021). URL: https://people.smp.uq.edu.au/PhilipPollett/papers/Qest/QEstAnnBib.pdf.

[23] M. A. Aslam and N. R. Aljohani. SPedia. *International Journal on Semantic Web and Information Systems*, 13(1):128–147, 2017. doi:10.4018/ijswis.2017010108.

[24] E. Aspland, D. Gartner, and P. Harper. Clinical pathway modelling: a literature review. *Health Systems*, 0(0):1–23, 2019. doi:10.1080/20476965.2019.1652547.

[25] L. Aviñó, M. Ruffini, and R. Gavaldà. Generating synthetic but plausible healthcare record datasets, 2018. arXiv:1807.01514.

[26] P. Aylin, S. Williams, A. Bottle, and B. Jarman. Counting hospital activity: spells or episodes? *BMJ*, 329(7476):1207, 2004. doi:10.1136/bmj.329.7476.1207.

[27] T. Bäck. Selective pressure in evolutionary algorithms: a characterization of selection mechanisms. In *Proceedings of the First IEEE Conference on Evolutionary Computation. IEEE World Congress on Computational Intelligence*, pages 57–62, 1994. doi:10.1109/ICEC.1994.350042.

[28] B. Bahmani, B. Moseley, A. Vattani, R. Kumar, and S. Vassilvitskii. Scalable k-means++, 2012. arXiv:1203.6402.

[29] A. M. Bakr, N. M. Ghanem, and M. A. Ismail. Efficient incremental density-based algorithm for clustering large datasets. *Alexandria Engineering Journal*, 54(4):1147 – 1154, 2015. doi:10.1016/j.aej.2015.08.009.

[30] S. Barocas, M. Hardt, and A. Narayanan. *Fairness and Machine Learning*. fairmlbook.org, 2019 (accessed Friday 9th April, 2021). URL: http://www.fairmlbook.org.

[31] D. M. Bashtannyk and R. J. Hyndman. Bandwidth selection for kernel conditional density estimation. *Computational Statistics and Data Analysis*, 36:279–298, 2001.

[32] S. Batistič and P. der Laken. History, evolution and future of big data and analytics: A bibliometric analysis of its relationship to performance in organizations. *British Journal of Management*, 30(2):229–251, 2019. doi:10.1111/1467-8551.12340.

[33] C. Baumberger, R. Knutti, and G. Hirsch Hadorn. Building confidence in climate model projections: an analysis of inferences from fit. *Wiley Interdisciplinary Reviews: Climate Change*, 8(3):e454, 2017.

[34] S. Bayat, Y. Li, L. Song, and Z. Han. Matching theory: Applications in wireless communications. *IEEE Signal Processing Magazine*, 33:103–122, 11 2016. doi:10.1109/MSP.2016.2598848.

[35] BBC News. Google DeepMind NHS app test broke UK privacy law, 2017 (accessed Friday 9th April, 2021). URL: https://www.bbc.co.uk/news/technology-40483202.

[36] P. C. Bell and R. M. O'Keefe. Visual interactive simulation — history, recent developments, and major issues. *SIMULATION*, 49(3):109–116, 1987. doi:10.1177/003754978704900304.

[37] A. Belle, R. Thiagarajan, S. Soroushmehr, F. Navidi, D. A. Beard, and K. Najarian. Big data analytics in healthcare. *BioMed research international*, 2015, 2015. doi:10.1155/2015/370194.

[38] S. Ben-David and M. Ackerman. Measures of clustering quality: A working set of axioms for clustering. In *Advances in Neural Information Processing Systems 21 - Proceedings of the 2008 Conference*, pages 121–128, 2008.

[39] B. Bengfort and R. Bilbro. Yellowbrick: Visualizing the Scikit-Learn Model Selection Process. *The Journal of Open Source Software*, 4(35), 2019. doi:10.21105/joss.01075.

[40] F. C. Y. Benureau and N. P. Rougier. Re-run, repeat, reproduce, reuse, replicate: Transforming code into scientific contributions. *Frontiers in Neuroinformatics*, 11, 2018. doi:10.3389/fninf.2017.00069.

[41] J. C. Bezdek and J. D. Harris. Fuzzy partitions and relations; an axiomatic basis for clustering. *Fuzzy sets and systems*, 1(2):111–127, 1978.

[42] U. N. Bhat. *An Introduction to Queueing Theory*. Birkhäuser Boston, 2015. doi:10.1007/978-0-8176-8421-1.

[43] P. Bhattacharjee and P. Mitra. A survey of density based clustering al-

gorithms. *Frontiers of Computer Science*, 15(1), 2020. doi:10.1007/s11704-019-9059-3.

[44] P. Bhattacharjee and P. K. Ray. Patient flow modelling and performance analysis of healthcare delivery processes in hospitals: A review and reflections. *Computers & Industrial Engineering*, 78:299–312, 2014. doi:10.1016/j.cie.2014.04.016.

[45] J. Billings, J. Dixon, T. Mijanovich, and D. Wennberg. Case finding for patients at risk of readmission to hospital: development of algorithm to identify high risk patients. *BMJ*, 333(7563):327, 2006. doi:10.1136/bmj.38870.657917.AE.

[46] Y. Bilu and N. Linial. Are stable instances easy? *Combinatorics, Probability and Computing*, 21(5):643–660, 2012. doi:10.1017/S0963548312000193.

[47] O. Bittencourt, V. Verter, and M. Yalovsky. Hospital capacity management based on the queueing theory. *International journal of productivity and performance management.*, 67(2):224–238, 2018.

[48] A. Boaz and D. Ashby. Fit for purpose?: assessing research quality for evidence based policy and practice, 2003.

[49] H.-H. Bock. *Clustering Methods: A History of k-Means Algorithms*, pages 161–172. Springer-Verlag Berlin Heidelberg, 2007. doi:10.1007/978-3-540-73560-1_15.

[50] A. Bogomolnaia and H. Moulin. Random matching under dichotomous preferences. *Econometrica*, 72(1):257–279, 2004. doi:10.1111/j.1468-0262.2004.00483.x.

[51] C. Böhm, R. Noll, C. Plant, and B. Wackersreuther. Density-based clustering using graphics processors. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, page 661–670, New York, NY, USA, 2009. Association for Computing Machinery. doi:10.1145/1645953.1646038.

[52] B. Borah and D. K. Bhattacharyya. An improved sampling-based DBSCAN for large spatial databases. In *International Conference on Intelligent Sensing and Information Processing*, pages 92–96, 2004. doi:10.1109/ICISIP.2004.1287631.

[53] C. Bouveyron, G. Celeux, T. B. Murphy, and A. E. Raftery. *Model-based*

clustering and classification for data science: with applications in R, volume 50. Cambridge University Press, 2019.

[54] G. Box. Robustness in the strategy of scientific model building. In *Robustness in Statistics*, pages 201–236. Elsevier, 1979. doi:10.1016/b978-0-12-438150-6.50018-2.

[55] P. S. Bradley, O. L. Mangasarian, and W. N. Street. Clustering via concave minimization. *Advances in Neural Information Processing Systems*, 9:368–374, 1997.

[56] S. C. Brailsford, T. B. Bolt, G. Bucci, T. M. Chaussalet, N. A. Connell, P. R. Harper, J. H. Klein, M. Pitt, and M. Taylor. Overcoming the barriers: A qualitative study of simulation adoption in the NHS. *Journal of the Operational Research Society*, 64(2):157–168, 2013.

[57] S. C. Brailsford, P. R. Harper, B. Patel, and M. Pitt. *An Analysis of the Academic Literature on Simulation and Modeling in Health Care*, pages 231–251. Palgrave Macmillan UK, London, 2016. doi:10.1007/978-1-137-57328-5_11.

[58] J. A. Briffa and S. Lygo-Baker. Enhancing student project selection and allocation in higher education programmes. In *2018 28th EAEEIE Annual Conference (EAEEIE)*, pages 1–6, 2018. doi:10.1109/EAEEIE.2018.8534298.

[59] British Medical Association. Models for paying providers of NHS services, 2020 (accessed Friday 9th April, 2021). URL: https://www.bma.org.uk/advice-and-support/nhs-delivery-and-workforce/funding/models-for-paying-providers-of-nhs-services.

[60] H. Bulut, A. Onan, and S. Korukoğlu. An improved ant-based algorithm based on heaps merging and fuzzy c-means for clustering cancer gene expression data. *Sādhanā*, 45(1), 2020. doi:10.1007/s12046-020-01399-x.

[61] I. Cadez, D. Heckerman, C. Meek, P. Smyth, and S. White. Visualization of navigation patterns on a web site using model-based clustering. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 280–284, 2000.

[62] R. J. G. B. Campello, D. Moulavi, and J. Sander. Density-based clustering based on hierarchical density estimates. In *Advances in Knowledge Discovery and Data Mining*, pages 160–172. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-37456-2_14.

[63] G. Campos, A. Zimek, J. Sander, R. Campello, B. Micenková, E. Schubert, I. Assent, and M. Houle. On the evaluation of unsupervised outlier detection: measures, datasets, and an empirical study. *Data Mining and Knowledge Discovery*, 30(4):891–927, 2016. doi:10.1007/s10618-015-0444-8.

[64] F. Cao, J. Liang, and L. Bai. A new initialization method for categorical data clustering. *Expert Systems with Applications*, 36:10223–10228, 2009. URL: https://pdfs.semanticscholar.org/1955/c6801bca5e95a44e70ce14180f00fd3e55b8.pdf.

[65] F. Cao, J. Liang, D. Li, L. Bai, and C. Dang. A dissimilarity measure for the *k*-modes clustering algorithm. *Knowledge-Based Systems*, 26:120–127, 2012. doi:10.1016/j.knosys.2011.07.011.

[66] P. Carter, G. T. Laurie, and M. Dixon-Woods. The social licence for research: why care.data ran into trouble. *Journal of Medical Ethics*, 41(5):404–409, 2015. doi:10.1136/medethics-2014-102374.

[67] H. Caswell. *The Validation Problem*, volume 4, pages 313–325. Academic Press, Cambridge, MA, 1976.

[68] R. B. Cattell. The description of personality: basic traits resolved into clusters. *The Journal of Abnormal and Social Psychology*, 38(4):476–506, 1943. doi:10.1037/h0054116.

[69] D. S. Char, M. D. Abràmoff, and C. Feudtner. Identifying ethical considerations for machine learning healthcare applications. *The American Journal of Bioethics*, 20(11):7–17, 2020. doi:10.1080/15265161.2020.1819469.

[70] A. Y. Chen and T.-Y. Yu. Network based temporary facility location for the emergency medical services considering the disaster induced demand and the transportation infrastructure in disaster response. *Transportation Research Part B: Methodological*, 91:408 – 423, 2016. doi:10.1016/j.trb.2016.06.004.

[71] X. Chen, B. Fain, C. Lyu, and K. Munagala. Proportionally fair clustering. *CoRR*, abs/1905.03674, 2019. arXiv:1905.03674.

[72] Y. Chen, M. Elliot, and J. Sakshaug. A genetic algorithm approach to synthetic data production. In *PrAISe@ECAI*, 2016.

[73] Y. Chen, M. Elliot, and D. Smith. The application of genetic algorithms to data synthesis: A comparison of three crossover methods. In *Privacy in*

Statistical Databases, pages 160–171. Springer International Publishing, 2018. doi:10.1007/978-3-319-99771-1_11.

[74] Y. Chen, X. Hu, W. Fan, L. Shen, Z. Zhang, X. Liu, J. Du, H. Li, Y. Chen, and H. Li. Fast density peak clustering for large scale data based on kNN. *Knowledge-Based Systems*, 187:104824, 2020. doi:10.1016/j.knosys.2019.06.032.

[75] Y. Chen, S. Tang, L. Zhou, C. Wang, J. Du, T. Wang, and S. Pei. Decentralized clustering by finding loose and distributed density cores. *Information Sciences*, 433-434:510 – 526, 2018. doi:10.1016/j.ins.2016.08.009.

[76] M. Chiarandini, R. Fagerberg, and S. Gualandi. Handling preferences in student-project allocation. *Annals of Operations Research*, 275(1):39–78, 2017. doi:10.1007/s10479-017-2710-1.

[77] J. K. Cochran and K. T. Roche. A multi-class queuing network analysis methodology for improving hospital emergency department performance. *Computers & Operations Research*, 36(5):1497–1512, 2009. doi:10.1016/j.cor.2008.02.004.

[78] V. Cohen-Addad, V. Kanade, F. Mallmann-Trenn, and C. Mathieu. Hierarchical clustering: Objective functions and algorithms. In *Proceedings of the 2018 Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 378–397, 2018. doi:10.1137/1.9781611975031.26.

[79] G. Colavizza, I. Hrynaszkiewicz, I. Staden, K. Whitaker, and B. McGillivray. The citation advantage of linking publications to research data. *PLOS ONE*, 15(4):e0230416, 2020. doi:10.1371/journal.pone.0230416.

[80] P. F. Collins, R. J. Stratton, R. J. Kurukulaaratchy, and M. Elia. Influence of deprivation on health care use, health care costs, and mortality in COPD. *International Journal of Chronic Obstructive Pulmonary Disease*, 13:1289–1296, 2018. doi:10.2147/COPD.S157594.

[81] S. Corbett-Davies and S. Goel. The measure and mismeasure of fairness: A critical review of fair machine learning, 2018. arXiv:1808.00023.

[82] L. Corti, V. Van den Eynden, L. Bishop, and M. Woollard. *Managing and sharing research data: a guide to good practice*. SAGE Publications Limited, 2019.

[83] K. Crawford. The hidden biases in big data, 2013 (accessed Friday 9th April, 2021). URL: https://hbr.org/2013/04/the-hidden-biases-in-big-data.

[84] A. Currie. From models-as-fictions to models-as-tools. *Ergo, an Open Access Journal of Philosophy*, 4(20201021), 2017. doi:10.3998/ergo.12405314.0004.027.

[85] C. M. Cutillo, K. R. Sharma, L. Foschini, S. Kundu, M. Mackintosh, and K. D. Mandl. Machine intelligence in healthcare—perspectives on trustworthiness, explainability, usability, and transparency. *npj Digital Medicine*, 3(1), 2020. doi:10.1038/s41746-020-0254-2.

[86] Cwm Taf Morgannwg UHB Informatics. Transforming ward experience for NHS staff and patients at Cwm Taf Morgannwg through e-whiteboards, 2020 (accessed Friday 9th April, 2021). URL: https://cwmtafmorgannwg.wales/transforming-ward-experience-...-at-cwm-taf-morgannwg-through-e-whiteboards/.

[87] G. Dagkakis and C. Heavey. A review of open source discrete event simulation software for operations research. *Journal of Simulation*, 10(3):193–206, 2016. doi:10.1057/jos.2015.9.

[88] J. Dahmen and D. Cook. SynSys: A synthetic data generation system for healthcare applications. *Sensors*, 19(5):1181, 2019. doi:10.3390/s19051181.

[89] S. Dasgupta. *A Cost Function for Similarity-Based Hierarchical Clustering*, page 118–127. Association for Computing Machinery, New York, NY, USA, 2016. doi:10.1145/2897518.2897527.

[90] H. A. Dau, E. Keogh, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, Yanping, B. Hu, N. Begum, A. Bagnall, A. Mueen, G. Batista, and Hexagon-ML. The UCR time series classification archive, 2018. URL: https://www.cs.ucr.edu/~eamonn/time_series_data_2018/.

[91] P. Dayan, M. Sahani, and G. Deback. Unsupervised learning. *The MIT encyclopedia of the cognitive sciences*, pages 857–859, 1999.

[92] N. J. de Vos. kmodes: Categorical clustering library, 2015. URL: https://github.com/nicodv/kmodes.

[93] D. Defays. An efficient algorithm for a complete link method. *The Computer Journal*, 20(4):364–366, 1977. doi:10.1093/comjnl/20.4.364.

[94] P. Delias, M. Doumpos, E. Grigoroudis, P. Manolitzas, and N. Matsatsinis. Supporting healthcare management decisions via robust clustering of

event logs. *Knowledge-Based Systems*, 84:203–213, 2015. doi:10.1016/j.knosys.2015.04.012.

[95] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[96] Department of Health. Equity and excellence: Liberating the NHS, 2010 (accessed Friday 9th April, 2021). URL: https://www.gov.uk/government/publications/liberating-the-nhs-white-paper.

[97] S. Deschênes, R. Burns, and N. Schmitz. Associations between diabetes, major depressive disorder and generalized anxiety disorder comorbidity, and disability: Findings from the 2012 Canadian Community Health Survey – Mental Health (CCHS-MH). *Journal of Psychosomatic Research*, 78(2):137 – 142, 2015. doi:10.1016/j.jpsychores.2014.11.023.

[98] W. N. Dewi, D. Evans, H. Bradley, and S. Ullrich. Person-centred care in the Indonesian health-care system. *International Journal of Nursing Practice*, 20(6):616–622, 2013. doi:10.1111/ijn.12213.

[99] E. Diday and J. Simon. Clustering analysis. In *Digital Pattern Recognition*, pages 47–94. Springer, 1976.

[100] B. J. Dietvorst, J. P. Simmons, and C. Massey. Algorithm aversion: People erroneously avoid algorithms after seeing them err. *Journal of Experimental Psychology: General*, 144(1):114–126, 2015. doi:10.1037/xge0000033.

[101] Y. Djabali, B. Rabta, and D. Aissani. Approximating service-time distributions by phase-type distributions in single-server queues: A strong stability approach. *International Journal of Mathematics in Operational Research*, 12:507–531, 2018. doi:10.1504/IJMOR.2018.10005095.

[102] H. E. Driver and A. L. Kroeber. Quantitative expression of cultural relationships, 1932.

[103] Q. Du, M. Emelianenko, and L. Ju. Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations. *SIAM Journal on Numerical Analysis*, 44(1):102–119, 2006. doi:10.1137/040617364.

[104] D. Dua and C. Graff. UCI Machine Learning Repository, 2017 (accessed Friday 9th April, 2021). URL: http://archive.ics.uci.edu/ml.

[105] K. Dube and T. Gallagher. Approach and method for generating realistic synthetic electronic healthcare records for secondary use. In *Foundations of Health*

Information Engineering and Systems, pages 69–86. Springer Berlin Heidelberg, 2014. doi:10.1007/978-3-642-53956-5_6.

[106] L. E. Dubins and D. A. Freedman. Machiavelli and the Gale-Shapley algorithm. *The American Mathematical Monthly*, 88(7):485–494, 1981.

[107] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel. On the shape of a set of points in the plane. *IEEE Transactions on Information Theory*, 29(4):551–559, July 1983. doi:10.1109/TIT.1983.1056714.

[108] A. W. F. Edwards and L. L. Cavalli-Sforza. A method for cluster analysis. In *Phylogenetic Inference, Selection Theory, and History of Science*, pages 53–67. Cambridge University Press, 1965. doi:10.1017/9781316276259.006.

[109] A. H. A. El-Atta and M. I. Moussa. Student project allocation with preference lists over (student, project) pairs. In *2009 Second International Conference on Computer and Electrical Engineering*. IEEE, 2009. doi:10.1109/iccee.2009.63.

[110] E. El-Darzi, R. Abbi, C. Vasilakis, F. Gorunescu, M. Gorunescu, and P. Millard. *Length of Stay-Based Clustering Methods for Patient Grouping*, pages 39–56. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009. doi:10.1007/978-3-642-00179-6_3.

[111] M. Elbattah and O. Molloy. Clustering-aided approach for predicting patient outcomes with application to elderly healthcare in Ireland. In *AAAI Workshops*, 2017.

[112] A. Erdil and H. Ergin. Two-sided matching with indifferences. *Journal of Economic Theory*, 171:268–292, 2017. doi:10.1016/j.jet.2017.07.002.

[113] A. K. Erlang. Solution of some problems in the theory of probabilities of significance in automatic telephone exchanges. *Post Office Electrical Engineer's Journal*, 10:189–197, 1917.

[114] A. K. Erlang. Telephone waiting times. *Matematisk Tidsskrift, B*, 31:25, 1920.

[115] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, 1996.

[116] M. Ester and R. Wittmann. Incremental generalization for mining in a data warehousing environment. In *International Conference on Extending Database Technology*, pages 135–149. Springer, 1998.

[117] European Data Protection Board. Norwegian DPA imposes administrative fine to Østfold HF Hospital, 2020 (accessed Friday 9th April, 2021). URL: https://edpb.europa.eu/news/.../norwegian-dpa-imposes-...-fine-ostfold-hf-hospital_en.

[118] European Organization For Nuclear Research and OpenAIRE. Zenodo, 2013. doi:10.25495/7GXK-RD71.

[119] B. S. Everitt, S. Landau, M. Leese, and D. Stahl. *Cluster analysis*. John Wiley & Sons, 2011.

[120] J. Fan. OPE-HCA: an optimal probabilistic estimation approach for hierarchical clustering algorithm. *Neural Computing and Applications*, 31(7):2095–2105, 2015. doi:10.1007/s00521-015-1998-5.

[121] A. D. R. Fernández, D. R. Fernández, and Y. S. García. Business process management for optimizing clinical processes: A systematic literature review. *Health Informatics Journal*, 26(2):1305–1320, 2019. doi:10.1177/1460458219877092.

[122] M. B. Ferraro and P. Giordani. A review and proposal of (fuzzy) clustering for nonlinearly separable data. *International Journal of Approximate Reasoning*, 115:13 – 31, 2019. doi:10.1016/j.ijar.2019.09.004.

[123] J. Findley, A. Woods, C. Robertson, and M. Slepian. Keeping the patient at the center of machine learning in healthcare. *The American Journal of Bioethics*, 20(11):54–56, 2020. doi:10.1080/15265161.2020.1820100.

[124] B. G. Fitzpatrick. Issues in reproducible simulation research. *Bulletin of Mathematical Biology*, 81:1–6, 2019. doi:10.1007/s11538-018-0496-1.

[125] S. Foster, D. Balmer, M. Gott, R. Frey, J. Robinson, and M. Boyd. Patient-centred care training needs of health care assistants who provide care for people with dementia. *Health & Social Care in the Community*, 27(4):917–925, 2019. doi:10.1111/hsc.12709.

[126] G. Frederix and E. J. Pauwels. Shape-invariant cluster validity indices. In *Advances in Data Mining*, pages 96–105. Springer Berlin Heidelberg, 2004. doi:10.1007/978-3-540-30185-1_11.

[127] S. A. Friedler, C. Scheidegger, and S. Venkatasubramanian. On the (im)possibility of fairness, 2016. arXiv:1609.07236.

[128] S. Fruhwirth-Schnatter, G. Celeux, and C. Robert. *Handbook of Mixture Anal-*

ysis. Chapman & Hall/CRC Handbooks of Modern Statistical Methods. CRC Press, 2019.

[129] T. Fuku, A. Namatame, and T. Kaizouji. Collective efficiency in two-sided matching. In *Lecture Notes in Economics and Mathematical Systems*, pages 115–126. Springer Berlin Heidelberg, 2006. doi:10.1007/3-540-28547-4_10.

[130] D. Gale and L. Shapley. College admissions and the stability of marriage. *The American Mathematical Monthly*, 69(1):9–15, 1962. doi:10.2307/2312726.

[131] P. Galetsi and K. Katsaliaki. A review of the literature on big data analytics in healthcare. *Journal of the Operational Research Society*, 71(10):1511–1529, 2020. doi:10.1080/01605682.2019.1630328.

[132] T. Gambling and A. F. Long. The realisation of patient-centred care during a 3-year proactive telephone counselling self-care intervention for diabetes. *Patient Education and Counseling*, 80(2):219 – 226, 2010. doi:10.1016/j.pec.2009.11.007.

[133] L. A. García-Escudero, A. Gordaliza, C. Matrán, and A. Mayo-Iscar. Exploring the number of groups in robust model-based clustering. *Statistics and Computing*, 21(4):585–599, 2011.

[134] H. Gerhards, K. Weber, U. Bittner, and H. Fangerau. Machine learning healthcare applications (ML-HCAs) are no stand-alone systems but part of an ecosystem – a broader ethical and health technology assessment approach is needed. *The American Journal of Bioethics*, 20(11):46–48, 2020. doi:10.1080/15265161.2020.1820104.

[135] GitHub, Inc. GitHub. URL: https://github.com/.

[136] GitHub, Inc. GitHub Actions. URL: https://docs.github.com/en/free-pro-team@latest/actions.

[137] P. Goelz, A. Kahng, and A. D. Procaccia. Paradoxes in fair machine learning. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32, pages 8342–8352. Curran Associates, Inc., 2019. URL: https://proceedings.neurips.cc/paper/2019/file/bbc92a647199b832ec90d7cf57074e9e-Paper.pdf.

[138] A. Goldenshluger. Nonparametric estimation of the service time distribution

in the M/G/∞ queue. *Advances in Applied Probability*, 48(4):1117–1138, 2016. doi:10.1017/apr.2016.67.

[139] D. Gondek, J. Edbrooke-Childs, T. Velikonja, L. Chapman, F. Saunders, D. Hayes, and M. Wolpert. Facilitators and barriers to person-centred care in child and young people mental health services: A systematic review. *Clinical Psychology & Psychotherapy*, 24(4):870–886, 2016. doi:10.1002/cpp.2052.

[140] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems 27*, pages 2672–2680, 2014 (accessed Friday 9th April, 2021). URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

[141] A. Gosain and S. Dahiya. Performance analysis of various fuzzy clustering algorithms: A review. *Procedia Computer Science*, 79:100 – 111, 2016. Proceedings of International Conference on Communication, Computing and Virtualization (ICCCV) 2016. doi:10.1016/j.procs.2016.03.014.

[142] P. Goswami, Z. Yan, A. Mukherjee, L. Yang, S. Routray, and G. Palai. An energy efficient clustering using firefly and HML for optical wireless sensor network. *Optik*, 182:181 – 185, 2019. doi:10.1016/j.ijleo.2018.12.191.

[143] T. Grote and P. Berens. On the ethics of algorithmic decision-making in healthcare. *Journal of Medical Ethics*, 46(3):205–211, 2020. doi:10.1136/medethics-2019-105586.

[144] C. Guan, K. K. F. Yuen, and F. Coenen. Particle swarm optimized density-based clustering and classification: Supervised and unsupervised learning approaches. *Swarm and Evolutionary Computation*, 44:876 – 896, 2019. doi:10.1016/j.swevo.2018.09.008.

[145] M. Gul and A. F. Guneri. A computer simulation model to reduce patient length of stay and to improve resource utilization rate in an emergency department service system. *International Journal of Industrial Engineering*, 19(5):221–231, 2012.

[146] S. Gupta, R. Kumar, K. Lu, B. Moseley, and S. Vassilvitskii. Local search methods for k-means with outliers. *Proc. VLDB Endow.*, 10:757–768, 2017.

[147] D. Gusfield and R. W. Irving. *The Stable Marriage Problem: Structure and Algorithms*. MIT Press, Cambridge, MA, USA, 1989.

[148] I. Habli, T. Lawton, and Z. Porter. Artificial intelligence in health care: accountability and safety. *Bulletin of the World Health Organization*, 98(4):251–256, 2020. doi:10.2471/blt.19.237487.

[149] G. Haeringer and V. Iehlé. Two-sided matching with one-sided preferences. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC '14, pages 353–353. ACM, 2014. doi:10.1145/2600057.2602853.

[150] G. Haeringer and V. Iehlé. Two-sided matching with (almost) one-sided preferences. *American Economic Jounral: Microeconomics*, 11(3):155–190, 2019. doi:10.1257/mic.20170115.

[151] J. A. Hagenaars. *Applied Latent Class Analysis*. Cambridge University Press, 2002. doi:10.1017/CBO9780511499531.

[152] R. A. Haraty, M. Dimishkieh, and M. Masud. An enhanced k-means clustering algorithm for pattern discovery in healthcare data. *International Journal of Distributed Sensor Networks*, 11(6):615740, 2015. doi:10.1155/2015/615740.

[153] C. Hargett, J. Doty, J. Hauck, A. Webb, S. Cook, N. Tsipis, J. Neumann, K. Andolsek, and D. Taylor. Developing a model for effective leadership in healthcare: a concept mapping approach. *Journal of Healthcare Leadership*, Volume 9:69–78, 2017. doi:10.2147/jhl.s141664.

[154] P. R. Harper and D. Winslett. Classification trees: A possible method for maternity risk grouping. *European Journal of Operational Research*, 169:146–156, 2006. doi:10.1016/j.ejor.2004.05.014.

[155] J. A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[156] J. A. Hartigan and M. A. Wong. Algorithm AS 136: A k-means clustering algorithm. *Journal of the Royal Statistical Society. Series C (Applied Statistics)*, 28(1):100–108, 1979.

[157] Y. He, H. Tan, W. Luo, H. Mao, D. Ma, S. Feng, and J. Fan. MR-DBSCAN: An efficient parallel density-based clustering algorithm using MapReduce. In *2011 IEEE 17th International Conference on Parallel and Distributed Systems*, pages 473–480, 2011. doi:10.1109/ICPADS.2011.83.

[158] S. Hettrick. UK Research Software Survey, 2014. doi:10.5281/zenodo.1183562.

[159] R. Higman, D. Bangert, and S. Jones. Three camps, one destination: the

intersections of research data management, FAIR and open. *Insights the UKSG journal*, 32, 2019. doi:10.1629/uksg.468.

[160] F. S. Hillier and G. J. Lieberman. *Introduction to Operations Research and Revised CD-ROM 8*. McGraw-Hill Science/Engineering/Math, 2005.

[161] A. Hinneburg and D. A. Keim. Optimal grid-clustering: Towards breaking the curse of dimensionality in high-dimensional clustering. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 506–517, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[162] C. W.-L. Ho. Deepening the normative evaluation of machine learning healthcare application by complementing ethical considerations with regulatory governance. *The American Journal of Bioethics*, 20(11):43–45, 2020. doi:10.1080/15265161.2020.1820106.

[163] R. Horn and A. Kerasidou. Sharing whilst caring: solidarity and public trust in a data-driven healthcare system. *BMC Medical Ethics*, 21(1), 2020. doi:10.1186/s12910-020-00553-8.

[164] J. Hou, H. Gao, and X. Li. DSets-DBSCAN: A parameter-free clustering algorithm. *IEEE Transactions on Image Processing*, 25(7):3182–3193, 2016. doi:10.1109/tip.2016.2559803.

[165] S. Houben-Wilke, F. J. J. Triest, F. M. Franssen, D. J. Janssen, E. F. Wouters, and L. E. Vanfleteren. Revealing methodological challenges in chronic obstructive pulmonary disease studies assessing comorbidities: A narrative review. *Chronic Obstructive Pulmonary Diseases: Journal of the COPD Foundation*, 6(2):166–177, 2019. doi:10.15326/jcopdf.6.2.2018.0145.

[166] Z. Huang. Clustering large data sets with mixed numeric and categorical values. In *The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, pages 21–34, 1997.

[167] Z. Huang. A fast clustering algorithm to cluster very large categorical data sets in data mining. In *Proceedings of the SIGMOD Workshop on Research Issues on Data Mining and Knowledge Discovery*, pages 1–8, 1997.

[168] Z. Huang. Extensions to the *k*-means algorithm for clustering large data sets with categorical values. *Data Mining and Knowledge Discovery*, 2(3):283–304, 1998. doi:10.1023/A:1009769707641.

[169] O. Hughes. 'More than half' of NHS trusts engaged in AI projects, report suggests. *Digital Health*, 2019 (accessed Friday 9th April, 2021). https:

//www.digitalhealth.net/2019/12/half-of-nhs-trusts-... -report-suggests/.

[170] S. Hussain, K. A. A. Gamage, M. H. Sagor, F. Tariq, L. Ma, and M. A. Imran. A systematic review of project allocation methods in undergraduate transnational engineering education. *Education Sciences*, 9(4):258, 2019. doi:10.3390/educsci9040258.

[171] International Alliance of Patients' Organizations. Patient-centred healthcare indicators review, 2012 (accessed Friday 9th April, 2021). URL: https://iapo.org.uk/sites/default/files/files/IAPO%20Patient-Centred%20Healthcare%20Indicators%20Review.pdf.

[172] B. E. Iott, C. Campos-Castillo, and D. L. Anthony. Trust and privacy: How patient trust in providers is related to privacy behaviors and attitudes. In *AMIA Annual Symposium Proceedings 2020*, pages 487–493, 2019.

[173] R. W. Irving. An efficient algorithm for the "stable roommates" problem. *Journal of Algorithms*, 6(4):577 – 595, 1985. doi:10.1016/0196-6774(85)90033-1.

[174] R. W. Irving. Stable marriage and indifference. *Discrete Applied Mathematics*, 48(3):261 – 272, 1994. doi:10.1016/0166-218X(92)00179-P.

[175] P. Ivie and D. Thain. Reproducibility in scientific computing. *ACM Computing Surveys*, 51(3), 2018. doi:10.1145/3186266.

[176] K. Iwama and S. Miyazaki. A survey of the stable marriage problem and its variants. In *International Conference on Informatics Education and Research for Knowledge-Circulating Society*, pages 131–136, 1 2008. doi:10.1109/ICKS.2008.7.

[177] K. Iwama and S. Miyazaki. *Stable Marriage with Ties and Incomplete Lists*, pages 2071–2075. Springer New York, 2016. doi:10.1007/978-1-4939-2864-4_805.

[178] K. Iwama, S. Miyazaki, Y. Morita, and D. Manlove. Stable marriage with incomplete lists and ties. In *Automata, Languages and Programming*, pages 443–452. Springer Berlin Heidelberg, 1999. doi:10.1007/3-540-48523-6_41.

[179] M. Jahangirian, S. Borsci, S. G. S. Shah, and S. J. E. Taylor. Causal factors of low stakeholder engagement: a survey of expert opinions in the context of healthcare simulation projects. *SIMULATION*, 91(6):511–526, 2015. doi:10.1177/0037549715583150.

[180] A. K. Jain. Data clustering: 50 years beyond k-means. *Pattern Recognition Letters*, 31(8):651–666, 2010. doi:10.1016/j.patrec.2009.09.011.

[181] A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering. *ACM Computing Surveys*, 31(3):264–323, 1999. doi:10.1145/331499.331504.

[182] N. Jardine and R. Sibson. *Mathematical Taxonomy*. Wiley Series in Probability and Mathematical Statistics. Wiley, 1971.

[183] K. Jebari. Selection methods for genetic algorithms. *International Journal of Emerging Sciences*, 3:333–344, 12 2013.

[184] F. Jiang, G. Liu, J. Du, and Y. Sui. Initialization of *k*-modes clustering using outlier detection techniques. *Information Sciences*, 332:167–183, 2016. doi:10.1016/j.ins.2015.11.005.

[185] R. C. Jiménez, M. Kuzak, M. Alhamdoosh, M. Barker, B. Batut, M. Borg, S. Capella-Gutierrez, N. Chue Hong, M. Cook, M. Corpas, M. Flannery, L. Garcia, J. L. Gelpí, S. Gladman, C. Goble, M. González Ferreiro, A. Gonzalez-Beltran, P. C. Griffin, B. Grüning, J. Hagberg, P. Holub, R. Hooft, J. Ison, D. S. Katz, B. Leskošek, F. López Gómez, L. J. Oliveira, D. Mellor, R. Mosbergen, N. Mulder, Y. Perez-Riverol, R. Pergl, H. Pichler, B. Pope, F. Sanz, M. V. Schneider, V. Stodden, R. Suchecki, R. SvobodováVařeková, H.-A. Talvik, I. Todorov, A. Treloar, S. Tyagi, M. van Gompel, D. Vaughan, A. Via, X. Wang, N. S. Watson-Haigh, and S. Crouch. Four simple recommendations to encourage best practices in research software. *F1000Research*, 6:ELIXIR–876, 6 2017. doi:10.12688/f1000research.11407.1.

[186] R. Jing, M. Wang, Z. Zhang, X. Wang, N. Li, N. Shah, and Y. Zhao. Distributed or centralized? Designing district-level urban energy systems by a hierarchical approach considering demand uncertainties. *Applied Energy*, 252:113424, 2019. doi:10.1016/j.apenergy.2019.113424.

[187] M. G. Johnson, L. Pokorny, S. Dodsworth, L. R. Botigué, R. S. Cowan, A. Devault, W. L. Eiserhardt, N. Epitawalage, F. Forest, J. T. Kim, J. H. Leebens-Mack, I. J. Leitch, O. Maurin, D. E. Soltis, P. S. Soltis, G. K. shu Wong, W. J. Baker, and N. J. Wickett. A universal probe set for targeted sequencing of 353 nuclear genes from any flowering plant designed using k-medoids clustering. *Systematic Biology*, 68(4):594–606, 2018. doi:10.1093/sysbio/syy086.

[188] S. C. Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[189] I. T. Jolliffe. *Principal Component Analysis and Factor Analysis*, pages 115–128. Springer New York, 1986. doi:10.1007/978-1-4757-1904-8_7.

[190] G. T. Jun, J. Ward, and P. J. Clarkson. Systems modelling approaches to the design of safe healthcare delivery: ease of use and usefulness perceived by healthcare workers. *Ergonomics*, 53(7):829–847, 2010. doi:10.1080/00140139.2010.489653.

[191] J. Katara and N. Choudhary. A modified version of the k-means clustering algorithm. *Global Journal of Human-Social Science Research*, 15, 2015.

[192] K. Katsaliaki and N. Mustafee. Applications of simulation within the healthcare context. *Journal of the Operational Research Society*, 62(8):1431–1451, 2011. doi:10.1057/jors.2010.20.

[193] L. Kaufman and P. J. Rousseeuw. Clustering by means of medoids, 1987.

[194] L. Kaufman and P. J. Rousseeuw, editors. *Finding Groups in Data*. John Wiley & Sons, Inc., 1990. doi:10.1002/9780470316801.

[195] M. Kearns. Fair algorithms for machine learning. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3033274.3084096.

[196] E. J. Keogh and A. Mueen. Curse of dimensionality. In C. Sammut and G. I. Webb, editors, *Encyclopedia of Machine Learning and Data Mining*, pages 314–315. Springer, 2017. doi:10.1007/978-1-4899-7687-1_192.

[197] S. Khan and A. Ahmad. Cluster center initialization algorithm for k-modes clustering. *Expert Systems with Applications*, 40:7444–7456, 2013. doi:10.1016/j.eswa.2013.07.002.

[198] S. Khan and S. Kant. Computation of initial modes for k-modes clustering algorithm using evidence accumulation. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, page 2784–2789, 2007.

[199] J. F. C. Kingman. The first Erlang century—and the next. *Queueing Systems*, 63(1-4):3–12, 2009. doi:10.1007/s11134-009-9147-4.

[200] V. Y. Kiselev, T. S. Andrews, and M. Hemberg. Challenges in unsupervised clustering of single-cell RNA-seq data. *Nature Reviews Genetics*, 20(5):273–282, 2019. doi:10.1038/s41576-018-0088-9.

[201] J. Kleinberg. An impossibility theorem for clustering. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Informa-*

tion Processing Systems, volume 15, pages 463–470. MIT Press, 2003. URL: https://proceedings.neurips.cc/paper/2002/file/43e4e6a6f341e00671e123714de019a8-Paper.pdf.

[202] P. Klimek, A. Kautzky-Willer, A. Chmiel, I. Schiller-Frühwirth, and S. Thurner. Quantification of diabetes comorbidity risks across life using nation-wide big claims data. *PLOS Computational Biology*, 11(4):1–16, 04 2015. doi:10.1371/journal.pcbi.1004125.

[203] V. A. Knight and P. R. Harper. Modelling emergency medical services with phase-type distributions. *Health Systems*, 1(1):58–68, 2012. doi:10.1057/hs.2012.1.

[204] C. Koleejan, B. Xue, and M. Zhang. Code coverage optimisation in genetic algorithms and particle swarm optimisation for automatic software test data generation. *2015 IEEE Congress on Evolutionary Computation (CEC)*, pages 1204–1211, 2015.

[205] A. Komashie, A. Mousavi, P. J. Clarkson, and T. Young. An integrated model of patient and staff satisfaction using queuing theory. *IEEE Journal of Translational Engineering in Health and Medicine*, 3:1–10, 2015. doi:10.1109/JTEHM.2015.2400436.

[206] S. A. Kraft. Respect and trustworthiness in the patient-provider-machine relationship: Applying a relational lens to machine learning healthcare applications. *The American Journal of Bioethics*, 20(11):51–53, 2020. doi:10.1080/15265161.2020.1820108.

[207] M. Kuehn, T. Severin, and H. Salzwedel. Variable mutation rate at genetic algorithms: Introduction of chromosome fitness in connection with multi-chromosome representation. *International Journal of Computer Applications*, 72:31–38, 07 2013. doi:10.5120/12636-9343.

[208] G. D. Kuh. The national survey of student engagement: Conceptual and empirical foundations. *New Directions for Institutional Research*, 2009(141):5–20, 2009. doi:10.1002/ir.283.

[209] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951. doi:10.1214/aoms/1177729694.

[210] A. Kumar and H. Anjomshoa. A two-stage model to predict surgical patients' lengths of stay from an electronic patient database. *IEEE Journal of Biomedical and Health Informatics*, 23(2):848–856, 2019. doi:10.1109/JBHI.2018.2819646.

[211] M. Kunc, P. Harper, and K. Katsikopoulos. A review of implementation of behavioural aspects in the application of OR in healthcare. *Journal of the Operational Research Society*, 71(7):1055–1072, 2018. doi:10.1080/01605682.2018.1489355.

[212] J. P. Kuwornu, L. M. Lix, and S. Shooshtari. Multimorbidity disease clusters in Aboriginal and non-Aboriginal Caucasian populations in Canada. *Chronic Diseases and Injuries in Canada*, 34(4):218–225, 2014.

[213] A. Kwanashie, R. W. Irving, D. F. Manlove, and C. T. S. Sng. Profile-based optimal matchings in the student/project allocation problem. In *Combinatorial Algorithms*, pages 213–225, 2015. doi:10.1007/978-3-319-19315-1_19.

[214] F. B. Larsen, M. H. Pedersen, K. Friis, C. Glümer, and M. Lasgaard. A latent class analysis of multimorbidity and the relationship to socio-demographic factors and health-related quality of life. a national population-based study of 162,283 Danish adults. *PLoS One*, 12(1), 2017. doi:10.1371/journal.pone.0169426.

[215] P. F. Lazarsfeld and N. W. Henry. *Latent structure analysis*. Houghton Mifflin Co., 1968.

[216] D. Lei, Q. Zhu, J. Chen, H. Lin, and P. Yang. Automatic k-means clustering algorithm for outlier detection. In *Lecture Notes in Electrical Engineering*, pages 363–372. Springer London, 2012. doi:10.1007/978-1-4471-2386-6_47.

[217] Y.-W. Leung and Y. Wang. An orthogonal genetic algorithm with quantization for global numerical optimization. *IEEE Transactions on Evolutionary Computation*, 5(1):41–53, 2001. doi:10.1109/4235.910464.

[218] J. Li and H. W. Lewis. Fuzzy clustering algorithms — review of the applications. In *2016 IEEE International Conference on Smart Cloud (SmartCloud)*, pages 282–288, 2016. doi:10.1109/SmartCloud.2016.14.

[219] J. D. C. Little. A proof for the queuing formula: $L = \lambda W$. *Operations Research*, 9(3):383–387, 1961. doi:10.1287/opre.9.3.383.

[220] S. S. Liu and J. Chen. Using data mining to segment healthcare markets from patients' preference perspectives. *International Journal of Health Care Quality Assurance*, 22(2):117–134, 2009. doi:10.1108/09526860910944610.

[221] X. Liu, S. C. Rivera, L. Faes, L. Ferrante di Ruffano, C. Yau, P. A. Keane, H. Ashrafian, A. Darzi, S. J. Vollmer, J. Deeks, L. Bachmann, C. Holmes,

A. W. Chan, D. Moher, M. J. Calvert, and A. K. Denniston. Reporting guidelines for clinical trials evaluating artificial intelligence interventions are needed. *Nature Medicine*, 25(10):1467–1468, 2019. doi:10.1038/s41591-019-0603-3.

[222] S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982. doi:10.1109/tit.1982.1056489.

[223] W.-K. Loh, Y.-S. Moon, and Y.-H. Park. Erratum: Fast density-based clustering using graphics processing units. *IEICE TRANSACTIONS on Information and Systems*, 97(7):1947–1951, 2014.

[224] K. Luxford, D. G. Safran, and T. Delbanco. Promoting patient-centered care: a qualitative study of facilitators and barriers in healthcare organizations with a reputation for improving the patient experience. *International Journal for Quality in Health Care*, 23(5):510–515, 2011. doi:10.1093/intqhc/mzr024.

[225] V. Lyzinski, M. Tang, A. Athreya, Y. Park, and C. E. Priebe. Community detection and classification in hierarchical stochastic blockmodels. *IEEE Transactions on Network Science and Engineering*, 4(01):13–26, 2017. doi:10.1109/TNSE.2016.2634322.

[226] R. Madhuri, M. R. Murty, J. V. R. Murthy, P. V. G. D. P. Reddy, and S. C. Satapathy. Cluster analysis on different data sets using k-modes and k-prototype algorithms. In *ICT and Critical Infrastructure: Proceedings of the 48th Annual Convention of Computer Society of India- Vol II*, pages 137–144. Springer International Publishing, 2014. doi:10.1007/978-3-319-03095-1_15.

[227] P. C. Mahalanobis. On the generalised distance in statistics. *Proceedings of the National Institute of Sciences of India*, 2(1):49–55, 1936.

[228] D. F. Manlove, R. W. Irving, K. Iwama, S. Miyazaki, and Y. Morita. Hard variants of stable marriage. *Theoretical Computer Science*, 276(1):261–279, 2002. doi:10.1016/S0304-3975(01)00206-7.

[229] D. F. Manlove, I. McBride, and J. Trimble. "Almost-stable" matchings in the hospitals / residents problem with couples. *Constraints*, 22(1):50–72, 2016. doi:10.1007/s10601-016-9249-7.

[230] D. F. Manlove and G. O'Malley. Student-project allocation with preferences over projects. *Journal of Discrete Algorithms*, 6(4):553–560, 2008. doi:10.1016/j.jda.2008.07.003.

[231] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Re-*

trieval. Cambridge University Press, 2008. URL: http://nlp.stanford.edu/IR-book/.

[232] J. J. Manoharan and S. H. Ganesh. Initialization of optimized k-means centroids using divide-and-conquer method. *ARPN Journal of Engineering and Applied Sciences*, 11(2), 2016.

[233] L. Martí, J. García, A. Berlanga, and J. M. Molina. A stopping criterion for multi-objective optimization evolutionary algorithms. *Information Sciences*, 367-368:700 – 718, 2016. doi:https://doi.org/10.1016/j.ins.2016.07.025.

[234] J. Matejka and G. Fitzmaurice. Same stats, different graphs: Generating datasets with varied appearance and identical statistics through simulated annealing. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*, CHI '17, pages 1290–1294. ACM, 2017. doi:10.1145/3025453.3025912.

[235] J. O. McClain. Bed planning using queuing theory models of hospital occupancy: A sensitivity analysis. *Inquiry*, 13(2):167–176, 1976.

[236] S. McClean, M. Barton, L. Garg, and K. Fullerton. A modeling framework that combines markov models and discrete-event simulation for stroke patient care. *ACM Trans. Model. Comput. Simul.*, 21(4), 2011. doi:10.1145/2000494.2000498.

[237] M. D. McCradden, J. A. Anderson, and R. Z. Shaul. Accountability in the machine learning pipeline: The critical role of research ethics oversight. *The American Journal of Bioethics*, 20(11):40–42, 2020. doi:10.1080/15265161.2020.1820111.

[238] M. D. McCradden, S. Joshi, J. A. Anderson, M. Mazwi, A. Goldenberg, and R. Z. Shaul. Patient safety and quality improvement: Ethical principles for a regulatory approach to bias in healthcare machine learning. *Journal of the American Medical Informatics Association*, 27(12):2024–2027, 2020. doi:10.1093/jamia/ocaa085.

[239] G. McDonald. Does size matter? The impact of student-staff ratios. *Journal of Higher Education Policy and Management*, 35(6):652–667, 2013. doi:10.1080/1360080X.2013.844668.

[240] W. McKinney. Data structures for statistical computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the*

9th Python in Science Conference, pages 56–61, 2010. doi:10.25080/Majora-92bf1922-00a.

[241] S. McLachlan, K. Dube, and T. Gallagher. Using the CareMap with health incidents statistics for generating the realistic synthetic electronic healthcare record. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*. IEEE, 2016. doi:10.1109/ichi.2016.83.

[242] P. McNicholas. Model-based clustering. *Journal of Classification*, 33:331–373, 2016.

[243] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan. A survey on bias and fairness in machine learning, 2019. arXiv:1908.09635.

[244] F. Mémoli. Metric structures on datasets: Stability and classification of algorithms. In *Computer Analysis of Images and Patterns*, pages 1–33. Springer Berlin Heidelberg, 2011. doi:10.1007/978-3-642-23678-5_1.

[245] K. Menzel. Large matching markets as two-sided demand systems. *Econometrica*, 83(3):897–941, 2015. doi:10.2307/43616957.

[246] C. C. Michael, G. McGraw, and M. Schatz. Generating software test data by evolution. *IEEE Trans. Software Eng.*, 27:1085–1110, 2001.

[247] T. M. Mitchell. *Machine Learning*. McGraw-Hill, Inc., USA, 1 edition, 1997.

[248] A. Mohammadi and M. R. Salehi-Rad. Bayesian inference and prediction in an $M/G/1$ with optional second service. *Communications in Statistics - Simulation and Computation*, 41(3):419–435, 2012. doi:10.1080/03610918.2011.588358.

[249] T. Monks. Operational research as implementation science: definitions, challenges and research priorities. *Implementation Science*, 11(1), 2015. doi:10.1186/s13012-016-0444-0.

[250] T. Monks, C. S. M. Currie, B. S. Onggo, S. Robinson, M. Kunc, and S. J. E. Taylor. Strengthening the reporting of empirical simulation studies: Introducing the STRESS guidelines. *Journal of Simulation*, 13(1):55–67, 2018. doi:10.1080/17477778.2018.1442155.

[251] B. Moseley and J. Wang. Approximation bounds for hierarchical clustering: Average linkage, bisecting k-means, and local search. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30, pages 3094–3103. Curran Associates, Inc.,

2017. URL: https://proceedings.neurips.cc/paper/2017/file/d8d31bd778da8bdd536187c36e48892b-Paper.pdf.

[252] T. Motoki. Calculating the expected loss of diversity of selection schemes. *Evolutionary Computation*, 10(4):397–422, 2002. doi:10.1162/106365602760972776.

[253] F. Murtagh. A survey of recent advances in hierarchical clustering algorithms. *The Computer Journal*, 26(4):354–359, 1983. doi:10.1093/comjnl/26.4.354.

[254] F. Murtagh and P. Contreras. Algorithms for hierarchical clustering: an overview. *WIREs Data Mining and Knowledge Discovery*, 2(1):86–97, 2012. doi:10.1002/widm.53.

[255] F. Murtagh and P. Contreras. Algorithms for hierarchical clustering: an overview, II. *WIREs Data Mining and Knowledge Discovery*, 7(6):e1219, 2017. doi:10.1002/widm.1219.

[256] J. A. Nasir, S. Hussain, and C. Dang. An integrated planning approach towards home health care, telehealth and patients group based care. *Journal of Network and Computer Applications*, 117:30 – 41, 2018. doi:10.1016/j.jnca.2018.05.009.

[257] National Institute for Health and Care Excellence (NICE). Guidance, advice and quality standards on diabetes and other endocrinal nutritional and metabolic conditions, 2011—present (accessed Friday 9th April, 2021). URL: https://www.nice.org.uk/guidance/conditions-and-diseases/.../diabetes.

[258] R. B. Nelsen. *An Introduction to Copulas*. Springer New York, 1999. doi:10.1007/978-1-4757-3076-0.

[259] M. K. Ng, M. J. Li, J. Z. Huang, and Z. He. On the impact of dissimilarity measure in *k*-modes clustering algorithm. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):503–507, 2007. doi:10.1109/TPAMI.2007.53.

[260] NHS Data Model and Dictionary. NHS Business Definitions: Consultant Episode, 2020 (accessed Friday 9th April, 2021). URL: https://www.datadictionary.nhs.uk/data_dictionary/nhs_business_definitions/c/consultant_episode_(hospital_provider)_de.asp.

[261] NHS Data Model and Dictionary. NHS Business Definitions: Hos-

pital Provider Spell, 2020 (accessed Friday 9th April, 2021). URL: https://www.datadictionary.nhs.uk/data_dictionary/nhs_business_definitions/h/hospital_provider_spell_de.asp.

[262] NHS England. The NHS long term plan, 2019 (accessed Friday 9th April, 2021). URL: https://www.longtermplan.nhs.uk/.

[263] F. Nielsen. Hierarchical clustering. In *Introduction to HPC with MPI for Data Science*, pages 195–211. Springer International Publishing, 2016. doi:10.1007/978-3-319-21903-5_8.

[264] Z. Obermeyer and E. J. Emanuel. Predicting the future — big data, machine learning, and clinical medicine. *New England Journal of Medicine*, 375(13):1216–1219, 2016. doi:10.1056/nejmp1606181.

[265] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan. Dissecting racial bias in an algorithm used to manage the health of populations. *Science*, 366(6464):447–453, 2019. doi:10.1126/science.aax2342.

[266] G. Ogbuabor and F. Ugwoke. Clustering algorithm for a healthcare dataset using silhouette score value. *International Journal of Computer Science & Information Technology*, 10(2):27–37, 2018.

[267] S. Olafsson, X. Li, and S. Wu. Operations research and data mining. *European Journal of Operational Research*, 87(3):1429–1448, 2008. doi:10.1016/j.ejor.2006.09.023.

[268] A. Olaode, G. Naghdy, and C. Todd. Unsupervised image classification by Probabilistic Latent Semantic Analysis for the annotation of images. In *International Conference on Digital Image Computing: Techniques and Applications*, 2014. doi:10.13140/2.1.1909.4086.

[269] S. Olaosebikan and D. Manlove. Super-stability in the student-project allocation problem with ties. *Journal of Combinatorial Optimization*, 2020. doi:10.1007/s10878-020-00632-x.

[270] T. Oliphant. NumPy: A guide to NumPy. USA: Trelgol Publishing, 2006–present (accessed Friday 9th April, 2021). URL: http://www.numpy.org/.

[271] R. S. Olson, W. La Cava, P. Orzechowski, R. J. Urbanowicz, and J. H. Moore. PMLB: a large benchmark suite for machine learning evaluation and comparison. *BioData Mining*, 10(1):36, 12 2017. doi:10.1186/s13040-017-0154-4.

[272] P. O. Olukanmi, F. Nelwamondo, and T. Marwala. Learning the k in k-

means via the Camp-Meidell inequality. In *2019 6th International Conference on Soft Computing Machine Intelligence (ISCMI)*, pages 211–216, 2019. doi:10.1109/ISCMI47871.2019.9004417.

[273] P. O. Olukanmi and B. Twala. K-means-sharp: Modified centroid update for outlier-robust k-means clustering. In *2017 Pattern Recognition Association of South Africa and Robotics and Mechatronics (PRASA-RobMech)*, pages 14–19, 2017. doi:10.1109/RoboMech.2017.8261116.

[274] N. Oreskes, K. Shrader-Frechette, and K. Belitz. Verification, validation, and confirmation of numerical models in the earth sciences. *Science*, 263(5147):641–646, 1994.

[275] R. Ostrovsky, Y. Rabani, L. Schulman, and C. Swamy. The effectiveness of Lloyd-type methods for the k-means problem. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*. IEEE, 2006. doi:10.1109/focs.2006.75.

[276] G. I. Palmer, V. A. Knight, P. R. Harper, and A. L. Hawa. Ciw: An open-source discrete event simulation library. *Journal of Simulation*, 13(1):68–82, 2019. doi:10.1080/17477778.2018.1473909.

[277] R. Palmer, N. J. Fulop, and M. Utley. A systematic literature review of operational research methods for modelling patient flow and outcomes within community healthcare and other settings. *Health Systems*, 7(1):29–50, 2018. doi:10.1057/s41306-017-0024-9.

[278] R. K. Palvannan and K. L. Teow. Queueing for healthcare. *Journal of Medical Systems*, 36:541–547, 2012. doi:10.1007/s10916-010-9499-7.

[279] T. Panch, H. Mattie, and L. A. Celi. The "inconvenient truth" about AI in healthcare. *npj Digital Medicine*, 2(1), 2019. doi:10.1038/s41746-019-0155-4.

[280] N. Park, M. Mohammadi, K. Gorde, S. Jajodia, H. Park, and Y. Kim. Data synthesis based on generative adversarial networks. *Proc. VLDB Endow.*, 11(10):1071–1083, 2018. doi:10.14778/3231751.3231757.

[281] W. S. Parker. Scientific models and adequacy-for-purpose. *The Modern Schoolman*, 87(3):285–293, 2010. doi:10.5840/schoolman2010873/410.

[282] W. S. Parker. Model evaluation: An adequacy-for-purpose view. *Philosophy of Science*, 87(3):457–477, 2020. doi:10.1086/708691.

[283] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel,

M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[284] L. Penn, A. Rodrigues, A. Haste, M. M. Marques, K. Budig, K. Sainsbury, R. Bell, V. Araújo-Soares, M. White, C. Summerbell, E. Goyder, A. Brennan, A. J. Adamson, and F. F. Sniehotta. NHS Diabetes Prevention Programme in England: formative evaluation of the programme in early phase implementation. *BMJ Open*, 8(2):e019467, 2018. doi:10.1136/bmjopen-2017-019467.

[285] H. J. W. Percival. *Test-Driven Development with Python: Obey the Testing Goat Using Django, Selenium, and JavaScript*. O'Reilly Media, Inc., 2nd edition, 2017.

[286] A. D. Peterson, A. P. Ghosh, and R. Maitra. Merging *k*-means with hierarchical clustering for identifying general-shaped groups. *Stat (International Statistical Institute)*, 7(1):e172, 2018. doi:10.1002/sta4.172.

[287] L. R. Pinto, F. C. C. de Campos, I. H. O. Perpétuo, and Y. C. N. M. B. Ribeiro. Analysis of hospital bed capacity via queuing theory and simulation. In *Proceedings of the Winter Simulation Conference 2014*, pages 1281–1292, 2014.

[288] A. Pompigna and R. Mauro. A multi-class time-dependent model for the analysis of waiting phenomena at a motorway tollgate. *Journal of Traffic and Transportation Engineering (English Edition)*, 2020. doi:10.1016/j.jtte.2020.09.001.

[289] V. U. Prabhu and A. Birhane. Large image datasets: A pyrrhic win for computer vision?, 2020. arXiv:2006.16923.

[290] D. Procida. Documentation system: The Grand Unified Theory of Documentation. URL: https://documentation.divio.com/.

[291] E. S. Prokofyeva, S. V. Maltseva, N. Y. Fomichev, and A. G. Kudryashov. Data-driven approach to patient flow management and resource utilization in urban medical facilities. In *2020 IEEE 22nd Conference on Business Informatics (CBI)*, volume 2, pages 71–77, 2020. doi:10.1109/CBI49978.2020.10060.

[292] Public Health England (PHE). Health matters: preventing Type 2 Diabetes, 2018 (accessed Friday 9th April, 2021). URL: https://www.gov.uk/government/publications/

health-matters-preventing-type-2-diabetes/health-matters-preventing-type-2-diabetes.

[293] A. Rajkomar, J. Dean, and I. Kohane. Machine learning in medicine. *New England Journal of Medicine*, 380(14):1347–1358, 2019. doi:10.1056/nejmra1814259.

[294] A. Ram, S. Jalal, A. S. Jalal, and M. Kumar. A density based algorithm for discovering density varied clusters in large spatial databases. *International Journal of Computer Applications*, 3(6):1–4, 2010. doi:10.5120/739-1038.

[295] A. Ramdas, N. G. Trillos, and M. Cuturi. On Wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47, 2017. doi:10.3390/e19020047.

[296] B. S. Rangvid. Gender discrimination in exam grading? Double evidence from a natural experiment and a field experiment. *The B.E. Journal of Economic Analysis & Policy*, 19(2), 2019. doi:10.1515/bejeap-2018-0084.

[297] D. Rankin, M. Black, R. Bond, J. Wallace, M. Mulvenna, and G. Epelde. Reliability of supervised machine learning using synthetic data in health care: Model to preserve privacy for data sharing. *JMIR Medical Informatics*, 8(7):e18910, 2020. doi:10.2196/18910.

[298] B. Rastegari, P. Goldberg, and D. Manlove. Preference elicitation in matching markets via interviews: A study of offline benchmarks. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*, AAMAS '16, pages 1393–1394, 2016.

[299] Y. P. Raykov, A. Boukouvalas, F. Baig, and M. A. Little. What to do when k-means clustering fails: A simple yet principled alternative algorithm. *PLOS ONE*, 11(9):1–28, 2016. doi:10.1371/journal.pone.0162259.

[300] Á. Rebuge and D. R. Ferreira. Business process analysis in healthcare environments: A methodology based on process mining. *Information Systems*, 37(2):99–116, 2012. doi:10.1016/j.is.2011.01.003.

[301] T. Richards, A. Coulter, and P. Wicks. Time to deliver patient centred care. *BMJ*, page h530, 2015. doi:10.1136/bmj.h530.

[302] D. Rios-Zertuche, A. Gonzalez-Marmol, F. Millán-Velasco, K. Schwarzbauer, and I. Tristao. Implementing electronic decision-support tools to strengthen healthcare network data-driven decision-making. *Archives of Public Health*, 78(1), 2020. doi:10.1186/s13690-020-00413-2.

[303] J. H. Robinson, L. C. Callister, J. A. Berry, and K. A. Dearing. Patient-centered care and adherence: Definitions and applications to improve outcomes. *Journal of the American Academy of Nurse Practitioners*, 20(12):600–607, 2008. doi:10.1111/j.1745-7599.2008.00360.x.

[304] S. Robinson, R. E. Nance, R. J. Paul, M. Pidd, and S. J. Taylor. Simulation model reuse: definitions, benefits and obstacles. *Simulation Modelling Practice and Theory*, 12(7-8):479–494, 2004. doi:10.1016/j.simpat.2003.11.006.

[305] E. Rojas, J. Munoz-Gama, M. Sepúlveda, and D. Capurro. Process mining in healthcare: A literature review. *Journal of Biomedical Informatics*, 61:224–236, 2016. doi:10.1016/j.jbi.2016.04.007.

[306] R. Romero-Silva and M. Hurtado. The difference of mean waiting times between two classes of customers in a single-server FIFO queue: An experimental study. *Cogent Engineering*, 4(1), 2017. doi:10.1080/23311916.2017.1321082.

[307] C. Ross and I. Swetlitz. IBM pitched its Watson supercomputer as a revolution in cancer care. It's nowhere close, 2017 (accessed Friday 9th April, 2021). URL: https://www.statnews.com/2017/09/05/watson-ibm-cancer/.

[308] A. Roth. The evolution of the labor market for medical interns and residents: A case study in game theory. *Journal of Political Economy*, 92(6):991–1016, 1984. doi:10.1086/261272.

[309] A. E. Roth and M. Sotomayor. Two-sided matching. *Handbook of game theory with economic applications*, 1:485–541, 1992.

[310] P. J. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. doi:10.1016/0377-0427(87)90125-7.

[311] A. Roy and S. Pokutta. Hierarchical clustering via spreading metrics. *J. Mach. Learn. Res.*, 18(1):3077–3111, 2017.

[312] Rui Xu and D. Wunsch. Survey of clustering algorithms. *IEEE Transactions on Neural Networks*, 16(3):645–678, 2005. doi:10.1109/TNN.2005.845141.

[313] N. B. Ruparelia. The history of version control. *SIGSOFT Softw. Eng. Notes*, 35(1):5–9, 2010. doi:10.1145/1668862.1668876.

[314] B. Réveil, D. Claeys, T. Maertens, J. Walraevens, and H. Bruneel. Impact

of class clustering in a multiclass FCFS queue with order-dependent service times. *Computers & Operations Research*, 51:90 – 98, 2014. doi:10.1016/j.cor.2014.05.016.

[315] J. Saltz, M. Skirpan, C. Fiesler, M. Gorelick, T. Yeh, R. Heckman, N. Dewar, and N. Beard. Integrating ethics within machine learning courses. *ACM Trans. Comput. Educ.*, 19(4), 2019. doi:10.1145/3341164.

[316] J. Sander, M. Ester, H.-P. Kriegel, and X. Xu. Density-based clustering in spatial databases: The algorithm GDBSCAN and its applications. *Data Mining and Knowledge Discovery*, 2(2):169–194, 1998. doi:10.1023/a:1009745219419.

[317] M.-J. Santana, S. Ahmed, D. Lorenzetti, R. J. Jolley, K. Manalili, S. Zelinsky, H. Quan, and M. Lu. Measuring patient-centred system performance: a scoping review of patient-centred care quality indicators. *BMJ Open*, 9(1), 2019. doi:10.1136/bmjopen-2018-023596.

[318] P. Santhi, V. M. Bhaskaran, et al. Performance of clustering algorithms in healthcare database. *International Journal for Advances in Computer Science*, 2(1):26–31, 2010.

[319] V. Satopaa, J. Albrecht, D. Irwin, and B. Raghavan. Finding a 'kneedle' in a haystack: Detecting knee points in system behavior. In *Proceedings of the 2011 31st International Conference on Distributed Computing Systems Workshops*, pages 166–171, 07 2011. doi:10.1109/ICDCSW.2011.20.

[320] S. E. Schaeffer. Graph clustering. *Computer Science Review*, 1(1):27–64, 2007. doi:10.1016/j.cosrev.2007.05.001.

[321] E. Schubert and P. J. Rousseeuw. Faster k-medoids clustering: Improving the PAM, CLARA, and CLARANS algorithms. In *Similarity Search and Applications*, pages 171–187. Springer International Publishing, 2019.

[322] E. Semenkin and M. Semenkina. Self-configuring genetic algorithm with modified uniform crossover operator. In *Advances in Swarm Intelligence*, pages 414–421, 2012.

[323] E. Sexton and D. Bedford. GP supply, deprivation and emergency admission to hospital for COPD and diabetes complications in counties across Ireland: An exploratory analysis. *Irish Journal of Medical Science*, 185(2):453–461, 2016. doi:10.1007/s11845-015-1359-5.

[324] H. Sharifipour, M. Shakeri, and H. Haghighi. Structural test data generation using a memetic ant colony optimization based on evolution strategies.

Swarm and Evolutionary Computation, 40:76 – 91, 2018. doi:10.1016/j.swevo.2017.12.009.

[325] N. Sharma and N. Gaud. *k*-modes clustering algorithm for categorical data. *International Journal of Computer Applications*, 127(17):1–6, 2015. doi:10.5120/ijca2015906708.

[326] J. F. Shortle, J. M. Thompson, D. Gross, and C. M. Harris. *Fundamentals of Queueing Theory*. John Wiley & Sons, Inc., 2018. doi:10.1002/9781119453765.

[327] R. Sibson. SLINK: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973. doi:10.1093/comjnl/16.1.30.

[328] P. Silitonga. Clustering of patient disease data by using k-means clustering. *International Journal of Computer Science and Information Security*, 15(7):219–221, 2018.

[329] T. Simon-Tuval, S. M. Scharf, N. Maimon, B. J. Bernhard-Scharf, H. Reuveni, and A. Tarasiuk. Determinants of elevated healthcare utilization in patients with COPD. *Respiratory Research*, 12(7), 2011. doi:10.1186/1465-9921-12-7.

[330] T. Simonite. When it comes to gorillas, Google Photos remains blind. *Wired*, 2018 (accessed Friday 9th April, 2021). URL: https://www.wired.com/story/when-it-comes-to-gorillas-google-photos-remains-blind/.

[331] H. Singh and K. Kaur. New method for finding initial cluster centroids in k-means algorithm. *International Journal of Computer Applications*, 74:27–30, 2013.

[332] D. F. Sittig and H. Singh. A new socio-technical model for studying health information technology in complex adaptive healthcare systems. In *Health Informatics*, pages 59–80. Springer International Publishing, 2015. doi:10.1007/978-3-319-17272-9_4.

[333] P. H. Sneath and R. R. Sokal. *Numerical taxonomy: The principles and practice of numerical classification*. W.H. Freeman, 1973.

[334] R. R. Sokal. Numerical taxonomy. *Scientific American*, 215(6):106–117, 1966.

[335] M. Sonka, V. Hlavac, and R. Boyle. *Image Processing, Analysis and Machine Vision*. Springer US, 1993. doi:10.1007/978-1-4899-3216-7.

[336] S. Stall, L. Yarmey, J. Cutcher-Gershenfeld, B. Hanson, K. Lehnert, B. Nosek, M. Parsons, E. Robinson, and L. Wyborn. Make scientific data FAIR. *Nature*, 570(7759):27–29, 2019. doi:10.1038/d41586-019-01720-7.

[337] A. Starczewski and A. Krzyżak. Performance evaluation of the silhouette index. In L. Rutkowski, M. Korytkowski, R. Scherer, R. Tadeusiewicz, L. A. Zadeh, and J. M. Zurada, editors, *Artificial Intelligence and Soft Computing*, pages 49–58, Cham, 2015. Springer International Publishing.

[338] A. Statman, L. Rozenberg, and D. Feldman. k-means+ + +: Outliers-resistant clustering. *Algorithms*, 13(12):311, 2020. doi:10.3390/a13120311.

[339] M. C. Steiner, D. Lowe, K. Beckford, J. Blakey, C. E. Bolton, S. Elkin, W. D. C. Man, C. M. Roberts, L. Sewell, P. Walker, and S. J. Singh. Socioeconomic deprivation and the outcome of pulmonary rehabilitation in England and Wales. *Thorax*, 72(6):530–537, 2017. doi:10.1136/thoraxjnl-2016-209376.

[340] K. Steins and S. Walther. A generic simulation model for planning critical care resource requirements. *Anaesthesia*, 68(11):1148–1155, 2013.

[341] W. J. Stewart. *Probability, Markov Chains, Queues, and Simulation*. Princeton University Press, 2009. doi:10.2307/j.ctvcm4gtc.

[342] M. Suganuma, S. Shirakawa, and T. Nagao. A genetic programming approach to designing convolutional neural network architectures. In *GECCO*, 2017.

[343] S. Sujatha and A. Sona. New fast k-means clustering algorithm using modified centroid selection method. *International Journal of Engineering Research and Technology*, 2(2), 2013.

[344] Y. Sun, B. Xue, M. Zhang, and G. G. Yen. Automatically designing CNN architectures using genetic algorithm for image classification. *CoRR*, abs/1808.03818, 2018.

[345] K.-K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):39–51, 1998. doi:10.1109/34.655648.

[346] M. A. Syakur, B. K. Khotimah, E. M. S. Rochman, and B. D. Satoto. Integration k-means clustering method and elbow method for identification of the best customer profile cluster. *IOP Conference Series: Materials Science and Engineering*, 336:012017, 2018. doi:10.1088/1757-899x/336/1/012017.

[347] L. Tao-ying, C. Yan, J. Zhi-hong, and L. Ye. Initialization of k-modes clus-

tering for categorical data. In *2013 International Conference on Management Science and Engineering 20th Annual Conference Proceedings*, pages 107–112, 2013. doi:10.1109/ICMSE.2013.6586269.

[348] I. Țăranu. Data mining in healthcare: decision making and precision. *Database Systems Journal*, 6(4):33–40, 2016.

[349] S. J. E. Taylor, T. Eldabi, T. Monks, M. Rabe, and A. M. Uhrmacher. Crisis, what crisis – does reproducibility in modeling simulation really matter? In *2018 Winter Simulation Conference (WSC)*, pages 749–762, 2018. doi:10.1109/WSC.2018.8632232.

[350] H. Teichgraeber and A. R. Brandt. Clustering methods to find representative periods for the optimization of energy systems: An initial framework and comparison. *Applied Energy*, 239:1283 – 1293, 2019. doi:10.1016/j.apenergy.2019.02.012.

[351] P. Tellaroli, M. Bazzi, M. Donato, A. R. Brazzale, and S. Drăghici. Cross-clustering: A partial clustering algorithm with automatic estimation of the number of clusters. *PLOS ONE*, 11(3):1–14, 2016. doi:10.1371/journal.pone.0152333.

[352] The Git developers. Git. URL: https://git-scm.com/.

[353] The Python Software Foundation. Python. URL: https://www.python.org/.

[354] J. Tilly and N. Janetos. MatchingR: Matching algorithms in R and C++, 2018. URL: https://github.com/jtilly/matchingR.

[355] D. Tomar and S. Agarwal. A survey on data mining approaches for healthcare. *International Journal of Bio-Science and Bio-Technology*, 5(5):241–266, 2013.

[356] A. Torfi and E. A. Fox. CorGAN: Correlation-capturing convolutional generative adversarial networks for generating synthetic healthcare records, 2020. arXiv:2001.09346.

[357] A. Torralba and A. A. Efros. Unbiased look at dataset bias. Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition, 2011. doi:10.1109/CVPR.2011.5995347.

[358] C.-F. Tsai and C.-W. Liu. KIDBSCAN: A new efficient data clustering algorithm. In *Artificial Intelligence and Soft Computing – ICAISC 2006*, pages 702–711. Springer Berlin Heidelberg, 2006. doi:10.1007/11785231_73.

[359] V. Tsianakas, G. Robert, J. Maben, A. Richardson, C. Dale, and T. Wise-

man. Implementing patient-centred cancer care: using experience-based co-design to improve patient experience in breast and lung cancer services. *Supportive Care in Cancer*, 20(11):2639–2647, 2012. doi:10.1007/s00520-012-1470-3.

[360] K. L. Tsui, N. Chen, Q. Zhou, Y. Hai, and W. Wang. Prognostics and health management: A review on data driven approaches. *Mathematical Problems in Engineering*, 2015:1–17, 2015. doi:10.1155/2015/793161.

[361] A. Tucker, Z. Wang, Y. Rotalinti, and P. Myles. Generating high-fidelity synthetic patient data for assessing machine learning healthcare software. *npj Digital Medicine*, 3(1), 2020. doi:10.1038/s41746-020-00353-9.

[362] UK Cabinet Office. Open public services white paper, 2011 (accessed Friday 9th April, 2021). URL: https://www.gov.uk/government/publications/open-public-services-white-paper.

[363] A. V. Ushakov and I. Vasilyev. Near-optimal large-scale k-medoids clustering. *Information Sciences*, 545:344 – 362, 2021. doi:10.1016/j.ins.2020.08.121.

[364] T.-P. van Staa, B. Goldacre, I. Buchan, and L. Smeeth. Big health data: the need to earn public trust. *BMJ*, page i3636, 2016. doi:10.1136/bmj.i3636.

[365] L. N. Vaserstein. Markov processes over denumerable products of spaces describing large systems of automata. *Problemy Peredačhi Informatsii*, 5(3):64–72, 1969.

[366] M. Velikova, J. T. van Scheltinga, P. J. Lucas, and M. Spaanderman. Exploiting causal functional relationships in bayesian network modelling for personalised healthcare. *International Journal of Approximate Reasoning*, 55(1):59–73, 2014. doi:10.1016/j.ijar.2013.03.016.

[367] P. A. Vikhar. Evolutionary algorithms: A critical review and its future prospects. In *2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC)*, pages 261–265, Dec 2016. doi:10.1109/ICGTSPICC.2016.7955308.

[368] S. Vollmer, B. A. Mateen, G. Bohner, F. J. Király, R. Ghani, P. Jonsson, S. Cumbers, A. Jonas, K. S. L. McAllister, P. Myles, D. Grainger, M. Birse, R. Branson, K. G. M. Moons, G. S. Collins, J. P. A. Ioannidis, C. Holmes, and H. Hemingway. Machine learning and artificial intelligence research for patient benefit: 20 critical questions on transparency, replicability, ethics, and effectiveness. *BMJ*, page l6927, 2020. doi:10.1136/bmj.l6927.

[369] S. I. Vuik, E. K. Mayer, and A. Darzi. Patient segmentation analysis offers significant benefits for integrated care and support. *Health Affairs*, 35(5):769–775, 2016. doi:10.1377/hlthaff.2015.1311.

[370] S. I. Vuik, E. K. Mayer, and A. Darzi. A quantitative evidence base for population health: Applying utilization-based cluster analysis to segment a patient population. *Population Health Metrics*, 14, 2016. doi:10.1186/s12963-016-0115-z.

[371] J. Walker, N. Halbesma, N. Lone, D. McAllister, C. Weir, and S. Wild. Socio-economic status, comorbidity and mortality in patients with type 2 diabetes mellitus in Scotland 2004 – 2011: a cohort study. *Journal of Epidemiology & Community Health*, 70(6):596–601, 2016. doi:10.1136/jech-2015-206702.

[372] D. Walsh. Occam's Razor: A principle of intellectual elegance. *American Philosophical Quarterly*, 16(3):241–244, 1979. URL: http://www.jstor.org/stable/20009764.

[373] P. Wang, W. Chen, and L. Zhao. Towards effective top-k location recommendation for business facility placement. In *Knowledge Science, Engineering and Management*, pages 51–63. Springer International Publishing, 2020. doi:10.1007/978-3-030-55393-7_5.

[374] J. H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. doi:10.1080/01621459.1963.10500845.

[375] Y. Wei, J. Jang-Jaccard, F. Sabrina, and T. R. McIntosh. MSD-Kmeans: A novel algorithm for efficient detection of global and local outliers. *CoRR*, abs/1910.06588, 2019. arXiv:1910.06588.

[376] H. Wilde, V. Knight, and J. Gillard. Evolutionary dataset optimisation: learning algorithm quality through evolution. *Applied Intelligence*, 50(4):1172–1191, 2020. doi:10.1007/s10489-019-01592-4.

[377] H. Wilde, V. Knight, and J. Gillard. Matching: A Python library for solving matching games. *Journal of Open Source Software*, 5(48):2169, 2020. doi:10.21105/joss.02169.

[378] H. Wilde, V. Knight, and J. Gillard. A novel initialisation based on hospital-resident assignment for the k-modes algorithm, 2020. arXiv:2002.02701.

[379] H. Wilde, V. Knight, J. Gillard, and K. Smith. Segmentation analysis and the recovery of queuing parameters via the Wasserstein distance: a study of

administrative data for patients with chronic obstructive pulmonary disease, 2020. arXiv:2008.04295.

[380] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, G. Appleton, M. Axton, A. Baak, N. Blomberg, J.-W. Boiten, L. B. da Silva Santos, P. E. Bourne, J. Bouwman, A. J. Brookes, T. Clark, M. Crosas, I. Dillo, O. Dumon, S. Edmunds, C. T. Evelo, R. Finkers, A. Gonzalez-Beltran, A. J. Gray, P. Groth, C. Goble, J. S. Grethe, J. Heringa, P. A. 't Hoen, R. Hooft, T. Kuhn, R. Kok, J. Kok, S. J. Lusher, M. E. Martone, A. Mons, A. L. Packer, B. Persson, P. Rocca-Serra, M. Roos, R. van Schaik, S.-A. Sansone, E. Schultes, T. Sengstag, T. Slater, G. Strawn, M. A. Swertz, M. Thompson, J. van der Lei, E. van Mulligen, J. Velterop, A. Waagmeester, P. Wittenburg, K. Wolstencroft, J. Zhao, and B. Mons. The FAIR guiding principles for scientific data management and stewardship. *Scientific Data*, 3(1), 2016. doi:10.1038/sdata.2016.18.

[381] J. Williams, S. Dumont, J. Parry-Jones, I. Komenda, J. Griffiths, and V. Knight. Mathematical modelling of patient flows to predict critical care capacity required following the merger of two district general hospitals into one. *Anaesthesia*, 70(1):32–40, 2015. doi:10.1111/anae.12839.

[382] G. Wilson, D. A. Aruliah, C. T. Brown, N. P. C. Hong, M. Davis, R. T. Guy, S. H. D. Haddock, K. D. Huff, I. M. Mitchell, M. D. Plumbley, B. Waugh, E. P. White, and P. Wilson. Best practices for scientific computing. *PLoS Biology*, 12(1):e1001745, 2014. doi:10.1371/journal.pbio.1001745.

[383] E. R. Witjas-Paalberends, L. P. M. van Laarhoven, L. H. M. van de Burgwal, J. Feilzer, J. de Swart, E. Claassen, and W. T. M. Jansen. Challenges and best practices for big data-driven healthcare innovations conducted by profit–non-profit partnerships – a quantitative prioritization. *International Journal of Healthcare Management*, 11(3):171–181, 2017. doi:10.1080/20479700.2017.1371367.

[384] R. P. Womack. Research data in core journals in biology, chemistry, mathematics, and physics. *PLOS ONE*, 10(12):e0143460, 2015. doi:10.1371/journal.pone.0143460.

[385] X. Wu and V. Kumar. *The top ten algorithms in data mining*. CRC press, 2009.

[386] Xiaowei Xu, M. Ester, H. . Kriegel, and J. Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *Proceedings 14th International Conference on Data Engineering*, pages 324–331, 1998. doi:10.1109/ICDE.1998.655795.

[387] J. Xu, G. Wang, and W. Deng. DenPEHC: Density peak based efficient

hierarchical clustering. *Information Sciences*, 373:200 – 218, 2016. doi:10.1016/j.ins.2016.08.086.

[388] X. Xu, J. Jäger, and H.-P. Kriegel. A fast parallel clustering algorithm for large spatial databases. In *High Performance Data Mining: Scaling Algorithms, Applications and Systems*, pages 263–290. Springer US, Boston, MA, 2002. doi:10.1007/0-306-47011-X_3.

[389] S. Yan, Y. H. Kwan, C. S. Tan, J. Thumboo, and L. L. Low. A systematic review of the clinical application of data-driven population segmentation analysis. *BMC Medical Research Methodology*, 18(121), 2018. doi:10.1186/s12874-018-0584-9.

[390] S. Yan, B. J. J. Seng, Y. H. Kwan, C. S. Tan, J. H. M. Quah, J. Thumboo, and L. L. Low. Identifying heterogeneous health profiles of primary care utilizers and their differential healthcare utilization and mortality – a retrospective cohort study. *BMC Family Practice*, 20(54), 2019. doi:10.1186/s12875-019-0939-2.

[391] G. B. Yom-Tov and A. Mandelbaum. Erlang-R: A time-varying queue with reentrant customers, in support of healthcare staffing. *Manufacturing & Service Operations Management*, 16(2):283–299, 2014. doi:10.1287/msom.2013.0474.

[392] I. Yoo, P. Alafaireet, M. Marinov, K. Pena-Hernandez, R. Gopidi, J.-F. Chang, and L. Hua. Data mining in healthcare and biomedicine: A survey of the literature. *Journal of Medical Systems*, 36(4):2431–2448, 2011. doi:10.1007/s10916-011-9710-5.

[393] S. Yoon, H. Goh, Y. H. Kwan, J. Thumboo, and L. L. Low. Identifying optimal indicators and purposes of population segmentation through engagement of key stakeholders: A qualitative study. *Health Res Policy Syst.*, 18(1):26, 2020. doi:10.1186/s12961-019-0519-x.

[394] N. Yousefi, F. Hasankhani, and M. Kiani. Appointment scheduling model in healthcare using clustering algorithms, 2020. arXiv:1905.03083.

[395] M. Zawati and M. Lang. What's in the box?: Uncertain accountability of machine learning applications in healthcare. *The American Journal of Bioethics*, 20(11):37–40, 2020. doi:10.1080/15265161.2020.1820105.

[396] C. E. Zayas, Z. He, J. Yuan, M. Maldonado-Molina, W. Hogan, F. Modave, Y. Guo, and J. Bian. Examining healthcare utilization patterns of elderly middle-aged adults in the United States. In *Proceedings of the... International*

Florida AI Research Society Conference. Florida AI Research Symposium, volume 2016, page 361. NIH Public Access, 2016.

[397] B. Zhang. Dependence of clustering algorithm performance on clustered-ness of data. *Hewlett-Packard Labs, Palo Alto, CA, USA, Tech. Rep*, 20010417, 2001.

[398] W. Zhao, H. Ma, and Q. He. Parallel k-means clustering based on MapReduce. In *Cloud Computing*, pages 674–679. Springer Berlin Heidelberg, 2009. doi:10.1007/978-3-642-10665-1_71.

[399] N. N. Zolkifli, A. Ngah, and A. Deraman. Version control system: A review. *Procedia Computer Science*, 135:408–415, 2018. doi:10.1016/j.procs.2018.08.191.

[400] A. Zuiderwijk, R. Shinde, and W. Jeng. What drives and inhibits researchers to share and use open research data? A systematic literature review to analyze factors influencing open research data adoption. *PLOS ONE*, 15(9):1–49, 2020. doi:10.1371/journal.pone.0239283.

Appendix A

An introduction to matching games

Matching games form a part of game theory that were formally introduced by Gale and Shapley in their seminal work [130]. These games allow for the allocation of resources and partnerships in a mathematically fair way. Typically, a matching game is defined by two sets of players (referred to as parties) that each have preferences over at least some of the elements of the other set. The objective of the game is then to find a mapping between the sets of players in which everyone is *happy enough* with their match(es).

This appendix does not contain any novel mathematics, but it does offer an introduction to matching games and their variants. Studying this branch of mathematics has contributed to a significant amount of the research conducted for this thesis, hence its inclusion here. That research has culminated in the development of a Python library for solving various matching games, `matching`. Among other uses, the `matching` library proves instrumental in the practical implementation of the novel method described in Chapter 4.

The `matching` library has been developed as a research tool and adheres to the best practices discussed in Chapter 1. The current version of Matching has also been archived on Zenodo under `doi:10.5281/zenodo.3931026`. Along with the source code being modularised and fully tested (using example, integration and property-based unit tests) with 100% coverage, the library is documented extensively. Like the `edo` library developed for the work in Chapter 3, the `matching` documentation is hosted online at `matching.readthedocs.io`. The documentation has been written to maximise its effect as a resource for learning about matching games as well as for the software itself. Furthermore, the library is registered on the Python

Package Index and is installable using standard Python practices.

```
> pip install matching
```

Snippet A.1: Installing the `matching` library via `pip`

Matching games have applications in many fields where relationships between rational agents must be arranged. Some example applications include: being able to inform on healthcare finance policy [7]; helping to reduce the complexity of automated wireless communication networks [34]; and education infrastructure [76, 170]. Thus, having access to software implementations of algorithms that are able to solve such games is essential.

The only current software alternative to `matching` is `MatchingR` [354]. `MatchingR` is a package written in C++ with an R interface and its content overlaps well with that of `matching`. However, the lack of a Python interface makes it less relevant to researchers and other users as Python's popularity grows both in academia and industry.

At the time of writing, the `matching` library offers facilities to handle and solve four types of matching games:

- The stable marriage problem (SM) [130];
- the hospital-resident assignment problem (HR) [130, 308];
- the student-project allocation problem (SA) [2, 3]; and
- the stable roommates problem (SR) [173].

This appendix goes through the details of the games for SM, HR and SA, the second of which is used in Chapter 4. A further piece of work conducted during the production of this thesis that uses SA is provided in Appendix C.

A.1 The stable marriage problem

One of the most ubiquitous matching games is the stable marriage problem (SM). SM describes the problem of finding a bijection between two distinct, equally sized sets of players that is stable according to the players' preferences. The notion of stability is broadly similar across all matching games, albeit up to the context of the game at hand. Definitions A.1 through A.4 formally introduce the components of SM.

Definition A.1. Consider two distinct sets, S and R, each of size $N \in \mathbb{N}$. These sets are the players of the game and are referred to as *suitors* and *reviewers*, respectively. Each element of S and R has a strict ranking of the other set's elements associated with it, and this ranking is called their *preference list*. The preference lists for each player set can be considered as a function which takes an element from the set and produces a permutation of the other set's elements:

$$f : S \to R^N; \quad g : R \to S^N \tag{A.1}$$

This construction of suitors, reviewers and preference lists is called a *game* of size N, denoted (S, R), and is used to model instances of SM.

Definition A.2. Consider a game (S, R). A *matching* M is any bijection between S and R. If a pair $(s, r) \in S \times R$ are matched in M, then that relationship is denoted $M(s) = r$ and, equivalently, $M^{-1}(r) = s$.

A matching is considered *valid* only if every player in (S, R) is matched to another player uniquely.

Definition A.3. Let (S, R) be an instance of SM, and consider $s \in S$ and $r, r' \in R$. Then s *prefers* r to r' if r appears before r' in $f(s)$. The definition of preference is equivalent for reviewers.

Definition A.4. Let (S, R) be an instance of SM and let M be a matching of (S, R). A pair $(s, r) \in S \times R$ is said to *block* M if:

- s and r are not matched by M, i.e. $M(s) \neq r$;
- s prefers r to $M(s) = r'$; and
- r prefers s to $M^{-1}(r) = s'$.

A matching M is said to be *stable* if it has no blocking pairs, and *unstable* otherwise.

This final definition envelopes the critical differences between the various matching games in existence. Despite their differences, however, the spirit is the same: a pair of players blocks a matching if their envy is *mutually rational*. Irrational envy would be where one player wishes to be matched to another over their current match but the other player does not (or cannot) reciprocate. The social outcome of acting irrationally in SM is that a player would be betraying their partner for another player, thus destabilising the group, without any reward of a 'better' partner.

A.1.1 Example instance

Consider the game of size three shown in Figure A.1 as an edgeless graph with suitors on the left and reviewers on the right. This representation of a matching game finds its origin in the bipartite matching problems of graph theory. Beside each vertex is the name of the player and their associated ranking of the complementary set's elements.

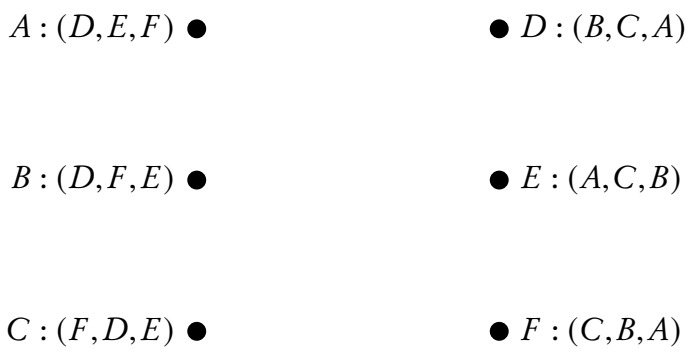


Figure A.1: A game of size three

In this representation, a matching M creates a bipartite graph where an edge between two vertices (players) indicates that they are matched by M. Figure A.2 shows an example of a valid matching.

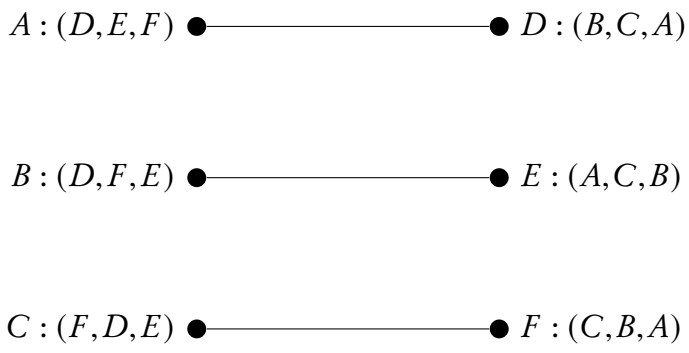


Figure A.2: An unstable matching to the game

In this matching, players A, C and F are all matched to their favourite player while B, D and E are matched to their least favourite. In particular, B and D form a blocking pair since they would both rather be matched to one another than their current match. Hence, this matching is unstable. As an attempt to rectify this instability, swap the matches for the first two rows, as shown in Figure A.3. This move does not form another blocking pair despite A having a worse match since D ranks A at the bottom of its preference list. Therefore, the envy exhibited by A is not reciprocated, and the matching is stable.

Upon closer inspection of this matching, it appears that no suitor can improve on

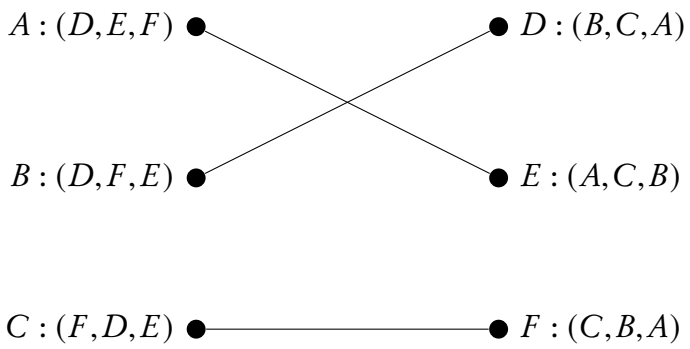


Figure A.3: A stable, suitor-optimal solution to the game

their current match without forming a blocking pair. In fact, the only suitor improvement would be for A and D to be matched again. This kind of stable matching is called *suitor-optimal*. Similarly, no reviewer can improve their match without forming a blocking pair and so this matching is also *reviewer-optimal*.

A.1.2 Solving instances of SM

Finding a party-optimal, stable matching to an instance of a matching game is referred to as *solving* the game. When there are only a handful of players to deal with, solving a game (or even finding a party-suboptimal, stable matching) is relatively straightforward with pen, paper and some time. However, solving the example above in two steps was little more than a coincidence. In the seminal paper on matching games [130], Gale and Shapley presented an algorithm for finding a unique, stable and suitor-optimal matching to any instance of SM. This algorithm has since become known as the Gale-Shapley algorithm, and is given in Algorithm A.1. The matching this algorithm produces is shown in Figure A.3.

As an instance of SM requires S and R to be of equal size, the reviewer-optimal algorithm is equivalent to Algorithm A.1 with the roles of suitors and reviewers reversed.

Even with the process described in the Gale-Shapley algorithm, solving an instance of SM soon becomes infeasible to do by hand in good time as the size of the game increases. Furthermore, instances of other matching games tend to have more players (and relationships between them) than SM and require the use of software to be solved in reasonable time. Hence, the development of the `matching` library which computes the matching as shown in Snippet A.2.

Algorithm A.1: The suitor-optimal algorithm for SM

Input: a set of suitors S, a set of reviewers R, two preference list functions f and g
Output: a stable, suitor-optimal matching M between S and R

for $p \in S \cup R$ **do**
    Set player p to be unmatched
end
while *there exists an unmatched suitor* $s \in S$ **do**
    Take any such suitor s and their favourite reviewer r
    if r *is matched to some other suitor* s' **then**
        Set r and s' to be unmatched
    end
    Match s and r, i.e. $M(s) \leftarrow r$
    for *each successor,* $t \in g(r)$*, to* s **do**
        DeletePair(r, t)
    end
end

Algorithm A.2: DeletePair

Input: two players p, q and their respective party's preference list functions f, g
Output: updated preference lists

$f(p) \leftarrow f(p) \setminus \{q\}$
$g(q) \leftarrow g(q) \setminus \{p\}$

```
>>> from matching.games import StableMarriage
>>> suitor_preferences = {
...     "A": ["D", "E", "F"], "B": ["D", "F", "E"], "C": ["F", "D", "E"]
... }
>>> reviewer_preferences = {
...     "D": ["B", "C", "A"], "E": ["A", "C", "B"], "F": ["C", "B", "A"]
... }
>>> game = StableMarriage.create_from_dictionaries(
...     suitor_preferences, reviewer_preferences
... )
>>> game.solve()
{A: E, B: D, C: F}
```

Snippet A.2: Solving the game from Figure A.1 in `matching`

A.1.3 Problem variants

Since the publication of [130], several other matching games have come into vogue, as well as variants to the fundamental games like SM. However, the accompanying algorithms for solving these games are still often structured to be party-oriented and aim to maximise some form of social or party-based optimality [129, 130]. In turn, these algorithms tend to follow a similar structure to Algorithm A.1, which has given the family of such matching game algorithms the name 'Gale-Shapley' algorithms.

A common and valuable extension to SM is the allowing of ties in a preference list; this is sometimes called indifference. Such an extension is straightforward enough to implement but the notion of stability becomes tiered; a matching is one of unstable, weakly stable, super-stable, or strongly stable [174, 177, 178]. In each case of stability, if such a matching exists, then a polynomial-time algorithm will find one that is optimal for one set of players. However, there is no guarantee that such a level-of-stable matching exists and, even in that case, the notion of party-optimality is lost [112].

Further to allowing ties, how preference lists are constructed is a point of interest in many applications of matching games [176, 228]. Often this is a contextual problem and may be addressed in a number of ways. As is briefly discussed in Chapter 4, bespoke preference list functions may be derived from some expert knowledge a priori to discourage particular matchings. Meanwhile, if the game forms part of a larger, long-standing or otherwise complex model, introducing flexibility in preferences (as in [7, 245]) may be helpful where streaming information should inform the preference lists. Likewise, [298] shows that estimating preference lists on the fly in the absence of complete information aids obtaining meaningful matchings.

A.2 The hospital-resident assignment problem

In addition to SM, [130] presented a game that modelled the college admission process. Since then, this game has been widely rebranded as the hospital-resident assignment problem (HR). This rebranding comes from it providing a practical solution to the problem that gives it its namesake: assigning medical students to resident positions at hospitals in the United States. A variant of the algorithm given in this section is used to this day by the National Resident Matching Program (NRMP).

HR is, in fact, a generalisation of SM. The game that models HR relaxes the conditions that the two player parties be the same size, and allows for multiple concurrent matches by the reviewing party (the hospitals in this case). Definitions A.5 through A.7 describe the components that make up the HR game.

Definition A.5. Consider two distinct sets, R and H, and refer to them as *residents* and *hospitals*. Each hospital $h \in H$ has a capacity $c_h \in \mathbb{N}$ associated with them that specifies their maximum number of concurrent matches. Each player $r \in R$ and $h \in H$ has associated with them a strict *preference list* of the other party's elements such that:

- Each resident $r \in R$ ranks a non-empty subset of H, denoted by $f(r)$; and
- each $h \in H$ ranks all and only those residents that have ranked it, i.e. the preference list of h, denoted $g(h)$, is a permutation of the set $\{r \in R \mid h \in f(r)\}$. If no such residents exist, h is removed from H.

This construction of residents, hospitals, capacities and preference lists is called a *game* and is denoted by (R, H). The notion of preference here is the same as in SM.

Definition A.6. Consider a game (R, H). A *matching* M is any mapping between R and H. If a pair $(r, h) \in R \times H$ are matched in M then this relationship is denoted $M(r) = h$ and $r \in M^{-1}(h)$.

A matching is only considered *valid* if for all $r \in R, h \in H$:

- $M(r) \in f(r)$ if r is matched;
- $M^{-1}(h) \subseteq g(h)$; and
- h is not over-subscribed, i.e. $\left|M^{-1}(h)\right| \leq c_h$.

Definition A.7. Consider a game (R, H). Then a pair $(r, h) \in R \times H$ is said to *block* a matching M if:

- There is mutual preference, i.e. $r \in g(h)$ and $h \in f(r)$;
- either r is unmatched or they prefer h to $M(r)$; and
- either h is under-subscribed or h prefers r to at least one resident in $M^{-1}(h)$.

A valid matching M is considered *stable* if it contains no blocking pairs, and *unstable* otherwise.

A.2.1 Example instance

Using games such as HR in practical settings has all the same social benefits as SM, and, in the case of assigning hospital residencies, HR allows for the fair distribution of talent. Attempting to assign medical students in a competitive market without such a system would encourage nepotism and backroom deals between hospitals and prospective applicants. Moreover, any social mobility afforded to students with fewer resources and opportunities is at risk without the protection of a stable

matching. These concerns are particularly important given the scale of many assignment problems. However, for illustrative purposes, consider the game shown in Figure A.4.

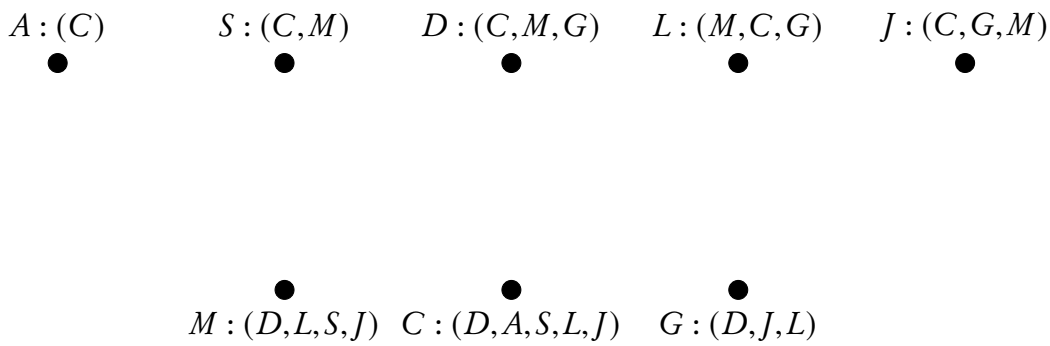


Figure A.4: An instance of HR

A similar representation to SM is used for instances of HR. Here, there are five applicants (along the top) and three hospitals (along the bottom). Although not shown, this example allows each hospital to accept no more than two residents. The benefit of visualising the game in this way is that the status of the solution is readily seen. For instance, consider the matching shown in Figure A.5.

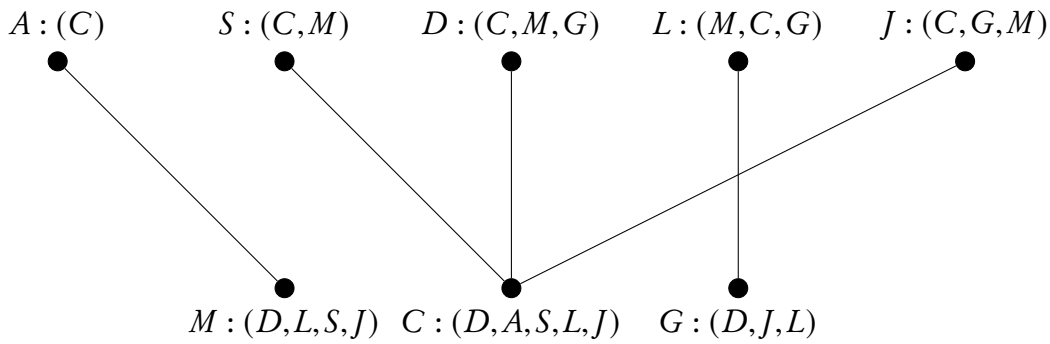


Figure A.5: An invalid matching for the instance

This matching is a mapping between the residents and hospitals, but it is not valid. In fact, none of the conditions for validity have been met: resident A has been matched to a hospital outside of their preferences; likewise for hospital M; and hospital C is over-subscribed with three residents. Correcting these issues could give something like the matching in Figure A.6.

While this matching is valid, it is unstable since resident L and hospital M form a blocking pair: there is mutual preference, L prefers M to G, and M has space available. Figure A.7 shows the now-stable matching following this move. Close inspection of this matching reveals that it is both resident- and hospital-optimal.

This particular example also demonstrates how robust Gale-Shapley algorithms are

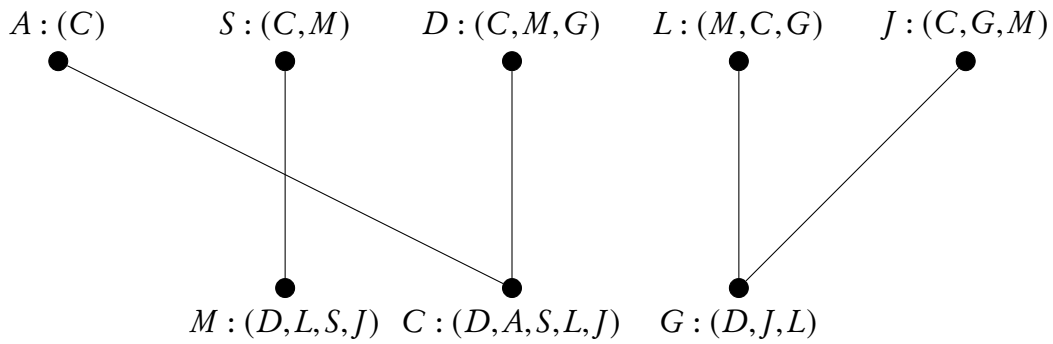


Figure A.6: An unstable matching for the instance

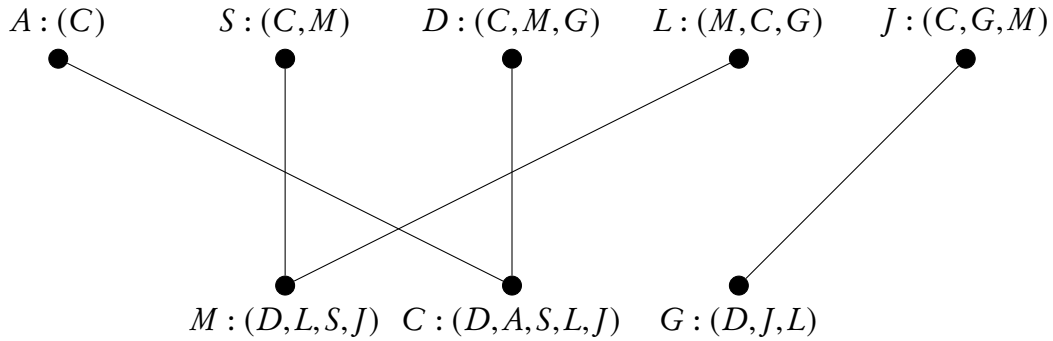


Figure A.7: A resident-optimal, stable matching for the instance

for solving real-world matching games. Suppose this was a real application pool, then resident A has decided that the only acceptable hospital placement is at hospital C, perhaps falsely assuming that this will guarantee them a place at C. On the contrary, the rules of the HR game do not stipulate that a stable matching must match all residents, and so a situation could arise where A will not be assigned to a hospital. For instance, if C swapped A and S in its preference list (because A did not meet certain academic requirements, say) then (S,C) would form a blocking pair under this matching. The only resolution that gives a stable matching then is to leave A without a match, and for $M(C) = \{S,D\}$.

A.2.2 Solving instances of HR

Like SM, [130] presented an algorithm that provides a unique, resident-optimal, stable matching to any instance of HR. However, further work (in [106, 308]) improved the Gale-Shapley algorithm for HR to take advantage of the structure of the game. This adapted algorithm is given in Algorithm A.3. An analogous hospital-optimal algorithm is omitted but follows a similar structure of considering available hospitals and removing successors from their favourite resident's preference list.

As in Section A.1, this algorithm produces the matching from the example in this section, i.e. that shown in Figure A.7. The code for solving this problem instance in

Algorithm A.3: The resident-optimal algorithm for HR

Input: an instance of HR (R, H)
Output: a stable, resident-optimal matching M between R and H

for *each resident* $r \in R$ **do**
  Set r to be unmatched
end
for *each hospital* $h \in H$ **do**
  Set h to be totally unsubscribed, i.e. $M^{-1}(h) \leftarrow \emptyset$
end
while *there exists any unmatched resident* $r \in R$ *with a nonempty preference list* **do**
  Take any such resident r and consider their favourite hospital h
  Match the pair, i.e. $M(r) \leftarrow h$ and $M^{-1}(h) \leftarrow M^{-1}(h) \cup \{r\}$
  if $\left|M^{-1}(h)\right| > c_h$ **then**
    Find their worst match $r' \in M^{-1}(h)$
    Unmatch the pair, i.e. r' is unmatched and $M^{-1}(h) \leftarrow M^{-1}(h) \setminus \{r'\}$
  end
  if $\left|M^{-1}(h)\right| = c_h$ **then**
    Find their worst match $r' \in M^{-1}(h)$
    for *each successor,* $s \in g(h)$, *to* r' **do**
      DeletePair(s, h)
    end
  end
end

`matching` is given in Snippet A.3.

```
>>> from matching.games import HospitalResident
>>> resident_preferences = {
...     "A": ["C"],
...     "S": ["C", "M"],
...     "D": ["C", "M", "G"],
...     "L": ["M", "C", "G"],
...     "J": ["C", "G", "M"],
... }
>>> hospital_preferences = {
...     "M": ["D", "L", "S", "J"],
...     "C": ["D", "A", "S", "L", "J"],
...     "G": ["D", "J", "L"],
... }
>>> hospital_capacities = {hospital: 2 for hospital in hospital_preferences}
>>> game = HospitalResident.create_from_dictionaries(
...     resident_preferences, hospital_preferences, hospital_capacities
... )
>>> game.solve()
{M: [L, S], C: [D, A], G: [J]}
```

Snippet A.3: Solving the instance from Figure A.4 in `matching`

A.2.3 Problem variants

The same extensions to SM exist for HR where indifference and custom preference list constructors are included; the NRMP uses its own ranking system for the hospital agents, for instance. In a sense, the generalisation of SM to HR includes allowing for a form of indifference by allowing incomplete preference lists by residents. Not ranking any subset of the hospitals is equivalent to ranking them all the same: as unacceptable. Allowing ties in a preference list would have its practical benefits, such as in the initialisation presented in Chapter 4. In that setting, the categorical dissimilarity measure corresponds to the ranking between potential representative points in a cluster and real data points. A shortcoming of that measure is the potentially high incidence of ties in the distances between a set of points. As such, allowing for ties to be reconciled in the matching game would be desirable. However, as no matching is guaranteed for any of the levels of stability in such a framework, it cannot be considered.

Further to these extensions, HR has given rise to its own contextual problems. One of these is allowing for couples in the resident party. In [229], the authors show that no stable matching is guaranteed to exist when couples are permitted. In response, a related, NP-hard problem is considered, where the objective is to identify an almost-stable matching that minimises the number of blocking pairs.

Another method used to construct preference lists is to discount the preference lists presented by players. For instance, where acceptability of another player is the only

criterion, binary preferences (i.e. incomplete preference lists with ties) can create games that are invulnerable to manipulative players' strategies [50]. This approach can be adapted to cater for larger games, such as student-school allocation (a special case of HR). In this scenario, each student submits a set of acceptable schools and the schools form strict rankings of the students. The result of this is a simpler game (in the practical sense) and a reduction in the set of possible stable matchings [149, 150].

A.3 The student-project allocation problem

The game for SA considers three sets of players — students, projects and supervisors — and seeks to assign students to projects according to preferences (from students and supervisors) and capacities (for supervisors and projects). Like HR, the practical importance of SA is to avoid unfair, ad hoc allocations in the eponymous problem of project allocation. There is substantial historical evidence and analysis indicating the dangers of allowing individual deal-making in matching scenarios [147, 308, 309].

Despite having three sets of players, SA is a two-sided matching (like SM and HR) as the projects and supervisors act as a single party. The applying party are the students, while the reviewing party consists of the supervisors with their projects. Any instance of HR can be stated as an instance of SA by replacing each hospital with a supervisor-project pair, making SA a generalisation of HR [2]. Definitions A.8 through A.10 describe the components that make up the SA game.

Definition A.8. Consider three distinct sets, S, P and U, and refer to them as *students*, *projects* and *supervisors*, respectively. Each project, $p \in P$, has a single supervisor, $u \in U$, associated with them. This association is described as a surjective function, $L : P \to U$, where the supervisor, $u \in U$, for a project, $p \in P$, can be written as $L(p) = u$. Note that since L is surjective, a supervisor may have multiple projects associated with them. The set of projects belonging to a supervisor, u, is written as $L^{-1}(u)$.

In addition to these supervisor-project associations, each project, $p \in P$, and supervisor, $u \in U$, has a *capacity*, denoted $c_p, c_u \in \mathbb{N}$, respectively. These capacities are such that for all $u \in U$:

$$\max\left\{c_p \mid p \in L^{-1}(u)\right\} \leq c_u \leq \sum_{p \in L^{-1}(u)} c_p \tag{A.2}$$

That is, every supervisor must be able to accommodate their largest project, but may

not offer more spaces than their projects sum to.

As with other matching games, each player has a *preference list* associated with them. In the case of SA, these preferences are strict and satisfy the following conditions:

- Each student, $s \in S$, ranks a non-empty subset of P, denoted by $f(s)$.
- Each supervisor, $u \in U$, ranks all and only those students who have ranked at least one of their projects, i.e. the preference list of u, denoted $g(h)$, is a permutation of the set $\{s \in S \mid L^{-1}(u) \cap f(s) \neq \emptyset\}$. If no students have ranked any of a supervisor's projects then u is removed from U.
- The preference list of each project, $p \in P$, is governed by its supervisor, $u = L(p)$. This preference, denoted $g_p(u)$, is identical to $g(u)$, but only includes those students who ranked p. If no students have ranked a project then that project is removed from P.

This construction of students, projects, supervisors, capacities and preference lists is called a *game* and is denoted by (S, P, U). The notion of preference here is the same as in SM and HR.

Definition A.9. Consider a game (S, P, U). A *matching* M is any mapping between S and P. If a pair $(s, p) \in S \times P$ are matched in M, then this is denoted $M(s) = p$ and $s \in M^{-1}(p)$.

Since each supervisor, $u \in U$, oversees their projects, their matching is taken as the union of its projects' matchings, i.e.

$$M^{-1}(u) = \bigcup_{p \in L^{-1}(u)} M^{-1}(p) \subseteq S \tag{A.3}$$

A matching is only considered *valid* if for all $s \in S, p \in P, u \in U$:

- $M(s) \in f(s)$ if s is matched;
- $M^{-1}(p) \subseteq g_p(L(p))$;
- p is not over-subscribed, i.e. $\left|M^{-1}(p)\right| \leq c_p$;
- $M^{-1}(u) \subseteq g(u)$; and
- u is not over-subscribed, i.e. $\left|M^{-1}(u)\right| \leq c_u$.

Definition A.10. Consider a game (S, P, U). Consider a pair $(s, p) \in S \times P$ and let $u = L(p)$. The pair is said to *block* a matching M if:

- There is mutual preference, i.e. $p \in f(s)$ and $s \in g_p(u)$;

- either s is unmatched or they prefer p to $M(s) = p'$; and
- at least one of the following is true:
  - Both p and u are under-subscribed, i.e. $\left|M^{-1}(p)\right| < c_p$ and $\left|M^{-1}(u)\right| < c_u$;
  - p is under-subscribed and u is at capacity, and either $M(s) = p' \in L^{-1}(u)$ or u prefers s to their worst current match $s' \in M^{-1}(u)$; or
  - p is at capacity and u prefers s to the project's least favourite student in $M^{-1}(p)$.

A valid matching M is considered *stable* if it contains no blocking pairs, and *unstable* otherwise.

A.3.1 Example instance

The definitions for this game are broadly similar to those for HR, with the exception of a blocking pair. Here, the definition of what may be considered *rational* is more complicated given the relationships between projects and their supervisors. To demonstrate these relationships, consider the game shown in Figure A.8.

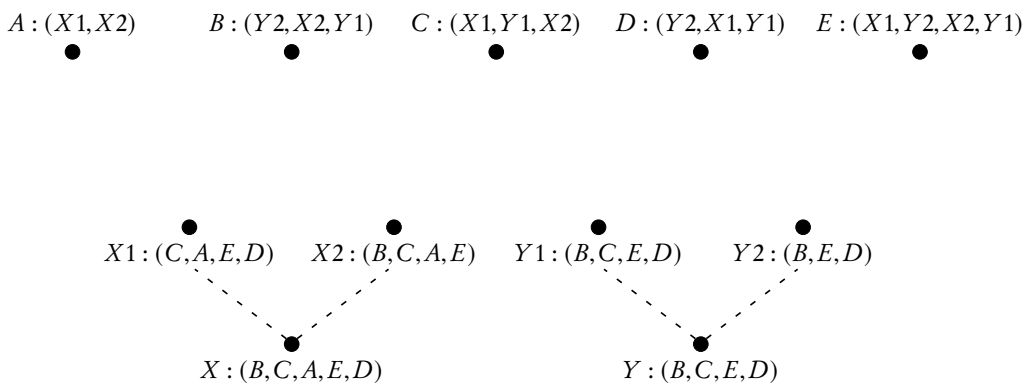


Figure A.8: An instance of SA

This game has five students (arranged along the top of Figure A.8) and two supervisors with two projects each (shown at the bottom of the figure). Here, the links between a supervisor and their projects is indicated by a dashed line. Although not shown, each supervisor has a capacity of three, while each project has a capacity of two.

Note also that the students are ranked in the same order by both supervisors. This practice is common in real-world applications of SA, including the case study in Appendix C. Drawing supervisor preferences from a complete ranking of the students strengthens the centralised decision-making of this approach.

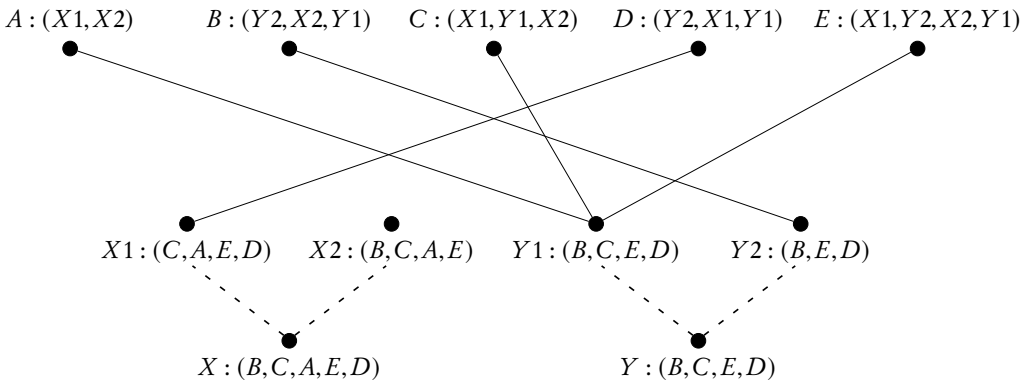


Figure A.9: An invalid matching for the instance

Suppose students were assigned to projects arbitrarily, ignoring preferences and capacities, like the allocation in Figure A.9. This matching meets none of the conditions for validity. Specifically:

- A has been assigned $Y1$ despite not ranking it. Likewise, the supervisor Y has not ranked A.
- $Y1$ has also been assigned three students, which exceeds its capacity of two.
- Y has been allocated a fourth student, violating their capacity constraint.

Correcting these issues requires several changes, and the following suggestions also try to maximise the quality of the matching. This manual approach is not uncommon in higher education institutions [170]. First, moving A and C to $X1$ addresses all three violations. Simultaneously, this move means both students will undertake their favourite project, and assigns favourable students to supervisor X.

However, this move introduces a new violation, where $X1$ is assigned three students. To rectify this, move D to their favourite project, $Y2$. The matching produced by these changes is shown in Figure A.10.

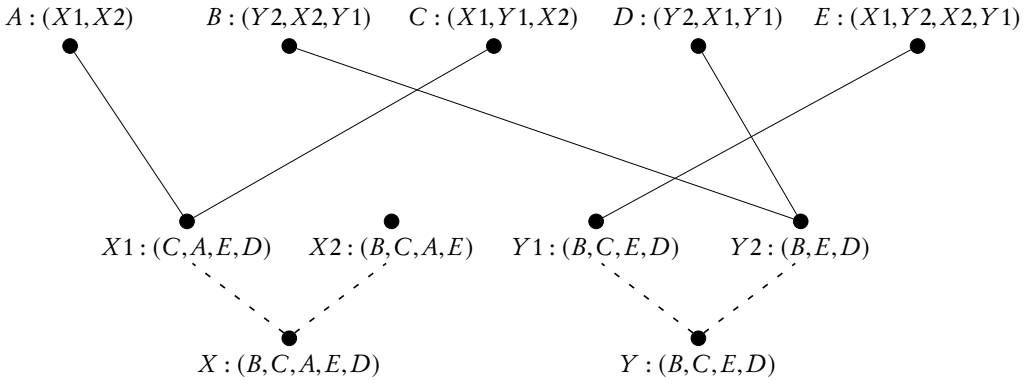


Figure A.10: An unstable matching for the instance

Despite the efforts to accommodate the preferences of the players, this matching

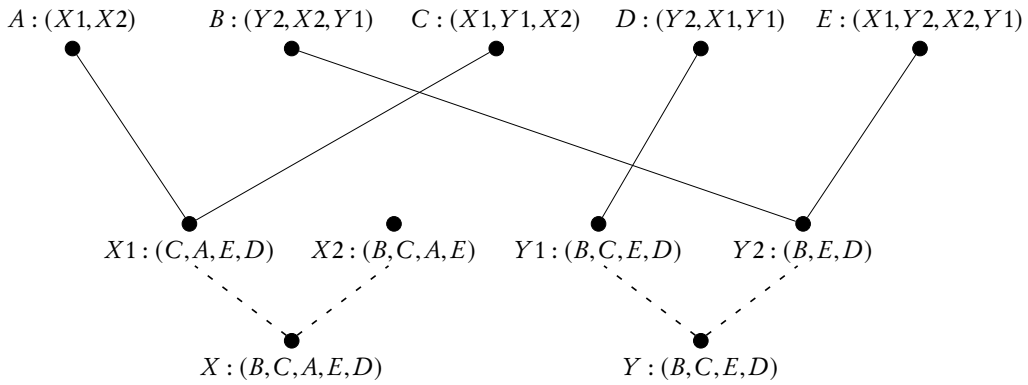


Figure A.11: A student-optimal, stable matching for the instance

is not stable. Here, there are two blocking pairs, $(E, X2)$ and $(E, Y2)$. Although student E prefers $X1$ to either of these projects, $(E, X1)$ does not form a blocking pair as $X1$ is at capacity and its supervisor, X, prefers both of its matches, A and C, to E.

So, to avoid these blocking pairs without creating more, students D and E must be swapped. Keeping fairness in mind, this is a sensible move given that E outranks D in the supervisors' preferences. Figure A.11 shows the resultant matching from this swap. Inspecting the matching, it is clear that no student may improve their allocation without creating a blocking pair, and so the matching is student-optimal.

A.3.2 Solving instances of SA

Game-theoretic work on the problem SA addresses has only come into interest relatively recently, beginning with the works [2, 3]. In the latter of these works, the authors present two Gale-Shapley algorithms for SA. These algorithms are party-oriented such that they each produce a unique, stable matching for an instance of SA that is student- or supervisor-optimal, respectively.

As with HR, the reviewing-party-optimal algorithm is omitted, but the student-optimal algorithm is given in Algorithm A.4. This algorithm, and its supervisor-optimal counterpart follow a similar structure to the others presented in this appendix: members of the applying party make temporary proposals, leading to the removal of infeasible pairs, until no acceptable applicants are left.

This algorithm produces the matching shown in Figure A.11, and can be solved in Python using the `matching` library, as shown in Snippet A.4. Here, there are several dictionaries required to describe the preferences, affiliations and capacities of the players in the game.

Algorithm A.4: The student-optimal algorithm for SA

Input: an instance of SA (S, P, U)
Output: a stable, student-optimal matching M between S and P

for *each student* $s \in S$ **do**
  Set s to be unmatched
end
for *each supervisor* $u \in U$ **do**
  Set u (and their projects) to be totally unsubscribed, i.e. $M^{-1}(u) \leftarrow \emptyset$
end
while *there exists any unmatched student* $s \in S$ *with a nonempty preference list* **do**
  Take any such student s and consider their favourite project p. Let $u = L(p)$
  Match the pair, i.e. $M(s) \leftarrow p$ and $M^{-1}(p) \leftarrow M^{-1}(p) \cup \{s\}$
  if $\left|M^{-1}(p)\right| > c_p$ **then**
    Find their worst match $s' \in M^{-1}(p)$
    Unmatch the pair, i.e. s' is unmatched and $M^{-1}(p) \leftarrow M^{-1}(p) \setminus \{s'\}$
  end
  else if $\left|M^{-1}(u)\right| > c_u$ **then**
    Find their worst match $s' \in M^{-1}(u)$ and let $p' = M(s')$
    Unmatch the pair, i.e. s' is unmatched and $M^{-1}(p') \leftarrow M^{-1}(p') \setminus \{s'\}$
  end
  if $\left|M^{-1}(p)\right| = c_p$ **then**
    Find their worst match $s' \in M^{-1}(p)$
    for *each successor,* $t \in g_p(u)$, *to* s' **do**
      DeletePair(t, p)
    end
  end
  if $\left|M^{-1}(u)\right| = c_u$ **then**
    Find their worst match $s' \in M^{-1}(u)$
    for *each successor,* $t \in g(u)$, *to* s' **do**
      for *each project,* $p' \in L(u) \cap f(t)$ **do**
        DeletePair(t, p')
      end
    end
  end
end

```
>>> from matching.games import StudentAllocation
>>> student_preferences = {
...     "A": ["X1", "X2"],
...     "B": ["Y2", "X2", "Y1"],
...     "C": ["X1", "Y1", "X2"],
...     "D": ["Y2", "X1", "Y1"],
...     "E": ["X1", "Y2", "X2", "Y1"],
... }
>>> supervisor_preferences = {
...     "X": ["B", "C", "A", "E", "D"], "Y": ["B", "C", "E", "D"]
... }
>>> project_supervisors = {"X1": "X", "X2": "X", "Y1": "Y", "Y2": "Y"}
>>> supervisor_capacities = {
...     supervisor: 3 for supervisor in supervisor_preferences
... }
>>> project_capacities = {project: 2 for project in project_supervisors}
>>> game = StudentAllocation.create_from_dictionaries(
...     student_preferences,
...     supervisor_preferences,
...     project_supervisors,
...     project_capacities,
...     supervisor_capacities,
... )
>>> game.solve()
{X1: [C, A], X2: [], Y1: [D], Y2: [B, E]}
```

Snippet A.4: Solving the instance from Figure A.8 in `matching`

A.3.3 Problem variants

In a similar fashion to SM and HR, generalising HR into SA admits many of the same extensions, including indifference [269]. Other variants of SA often arise from the constraints presented by the particular applications of project allocation. For instance, in [3], the authors leave open questions about solving instances of SA where each project has a minimum number of assignees. This extension would handle the common issue of projects with multiple independent parts, or those that require group efforts [20].

A natural continuation of this is to allow students to apply to projects in groups. Allowing for student coalitions complicates the allocation process. The authors of [76] study this variant through mathematical programming. This technique has also been applied to other intricate allocation variants, including the maximisation of other optimality measures [213].

Another extension to SA is in how preferences are constructed. For instance, in [230], the authors present a variant of SA in which supervisors hold preferences over their projects — as opposed to the students. This change in preferences allows supervisors to prioritise working on projects more closely related to their own research, and removes any practical challenges with ranking students. Another variant, which combines supervisor-project priority with their preferences over students, is presented in [109].

Appendix B

An exploratory analysis of administrative data

This appendix provides an exploratory analysis of a patient-episode dataset provided by the Cwm Taf Morgannwg University Health Board (UHB). This dataset details, among other administrative quantities, the costs associated with treating patients during their time in hospital.

The purpose of this analysis is to locate any surface-level sources of variation in these costs. In particular, this analysis considers a selection of attributes associated with costs, their distributions across the whole dataset, and how they interact with one another. These attributes are comprised of non-trivial cost components and a set of clinical attributes that are known to drive costs.

The subsequent analysis reveals that, while the bulk of the data corresponds to short-stay and relatively low-impact spells of treatment, there are long, heavy tails with high levels of variation in each of these variables. As such, a more homogeneous part of the population should be considered to find more actionable results.

In aid of this, an approach for the analysis of slices within the data is established, using the diabetic population as an example. This framework provides another dimension to the overall analysis through the use of comparison and contrast, but the intended impact is ultimately lost due, again, to high levels of variation.

The remainder of this appendix is structured as follows:

- Section B.1 provides an overview of the dataset and its key attributes
- Section B.2 explores the subset of the data corresponding to diabetic patients
- Section B.3 summarises the findings of this analysis

B.1 An overview of the data

B.1.1 Data structure

Before any analysis can be conducted, the structure of the data must be understood, as well as how it has been prepared. This dataset comprises approximately two and a half million episode records for patients from across Wales who were treated in the Prince Charles and Royal Glamorgan hospitals (South Wales) from April 2012 through April 2017. This dataset contains some personal information, and given that sensitivity, it has not been made available.

An episode is defined to be any continuous period of care provided by the same consultant in the same place [260]. For instance, if a patient is admitted to a general medical ward for diagnostic testing, and then is referred to a specialist consultant in oncology, then their first episode would end with their testing, and a second episode of care would begin on the oncology ward. Each of these episodes would correspond to a row in the dataset. If the patient was then immediately discharged, they would have completed a spell with two episodes.

Looking at the episodes directly will be avoided in this analysis. Instead, this analysis favours aggregating a patient's episodes into spells. Statistics associated with these aggregates are referred to as *spell-level* statistics. The reason for this level of aggregation is that it has been seen that episode-level statistics can lead to an overestimation of the resource or 'activity' consumed by a hospital to treat a patient during that time [26].

Furthermore, the processes of paying for treatments and reimbursing other organisations in NHS Wales is much simpler than in England. Within NHS England, a more transparent, albeit complex, method is used wherein an NHS organisation takes 'payment-by-results' according to a National Tariff [59, 362]. This system for financial flows is not necessary in Wales since patients tend to receive treatment within the jurisdiction of the Health Board they reside in. Figure B.1 shows an approximation for the geographic distribution of the patients in this dataset, and it is evident that the vast majority live in the Cwm Taf Morgannwg UHB area of South Wales. Since patients are treated locally, inter-organisation disputes are uncommon and NHS Wales continues to utilise block contracts for the majority of its payments [59]. The critical issue with block contracts is that they lack the precision of a payment-by-results system that can attribute costs (and accountability) to an organisation on a fine scale. Therefore, aggregating the episodes into spells may smooth out any unpredictability in the episode costs.

Each episode is recorded as a row of roughly 260 attributes or columns, includ-

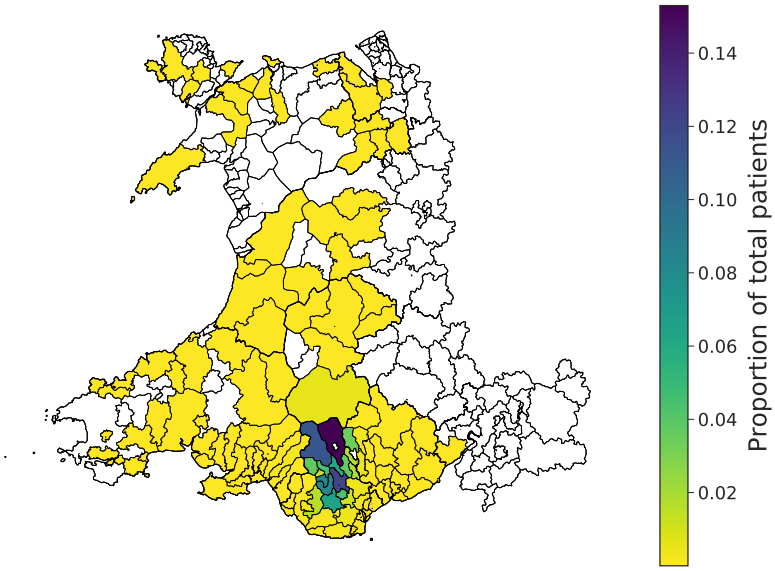


Figure B.1: The proportion of patients by their postcode district

ing:

- Personal information such as unique identifiers, age, and registered GP practice;
- Clinical quantities such as the number of diagnoses made and procedures conducted in that episode, admission and discharge dates and methods, and length of stay;
- A number of cost components, including the costs associated with the hospital departments, overall medical and ward costs, and overhead costs;
- Diagnosis (HRG, ICD-10) and procedure (OPCS-4) codes, as well as Charlson Comorbidity Index (CCI) scores for the appropriate chronic conditions.

Of the attributes listed here, this analysis considers the total, net and component costs, and a selection of other clinical variables. This selection pays particular attention to those attributes which are considered to be linked to an overall contribution to the cost of care. Those attributes are:

- length of stay;
- the maximum number of diagnoses during a spell;
- the total number of procedures during a spell;

- and (separately) the number of spells associated with any given patient.

B.1.2 Cleaning the data

As with any data analysis, a substantial amount of preprocessing is required to make the data sufficiently consistent and suitable for analysing. With the dataset at hand, this process included the removal of some superfluous attributes which added unwanted redundancy to the dataset, and a number of rows that had been corrupted or coded incorrectly. In addition to this, some columns have been reformatted; namely those whose entries were intended to be used as date-time objects such as admission and discharge dates.

It has already been stated that the majority of the attributes in the dataset will not be considered in this analysis. By ignoring these attributes, the focus is purely on how the costs of care appear in the data. The subset of chosen attributes will frequently be referred to as the set of *key attributes*. However, this name does not imply that the remaining attributes are not of interest nor that they are in any way unimportant.

The key attributes provide a base for understanding how the costs and resources consumed by a patient in a spell originate. Cost components give direct information on which departments are being utilised, and by how much; the length of stay can offer an indication of the nature of the spell and any costs that may be incurred automatically by merely spending more time in hospital; and considering the maximum and total number of diagnoses and procedures (respectively) in a spell allow for some insight into the severity or complexity of a patient's spell in hospital.

B.1.3 Distributions and summary statistics

When looking at the distributions of the key attributes on the whole dataset, as displayed in Figures B.2 through B.6, it is clear that the data is weighted towards low-cost, short-stay, and otherwise low-impact patients. This is especially clear in Figures B.2 and B.3. Here, it is clear that, of all the spells provided under the care of the health board, the majority are day-cases. Also, the patients being treated are one-time users of the hospital system.

In general, the distributions themselves have long tails, suggesting an adverse effect from cases that are severe but rare. Moreover, although the length and returning frequency of the spells are minimal and tightly packed, their associated net costs are wildly variant. This variation is shown in Figure B.6. It appears that there is a distinct peak in the distribution, but closer inspection of the scale indicates that

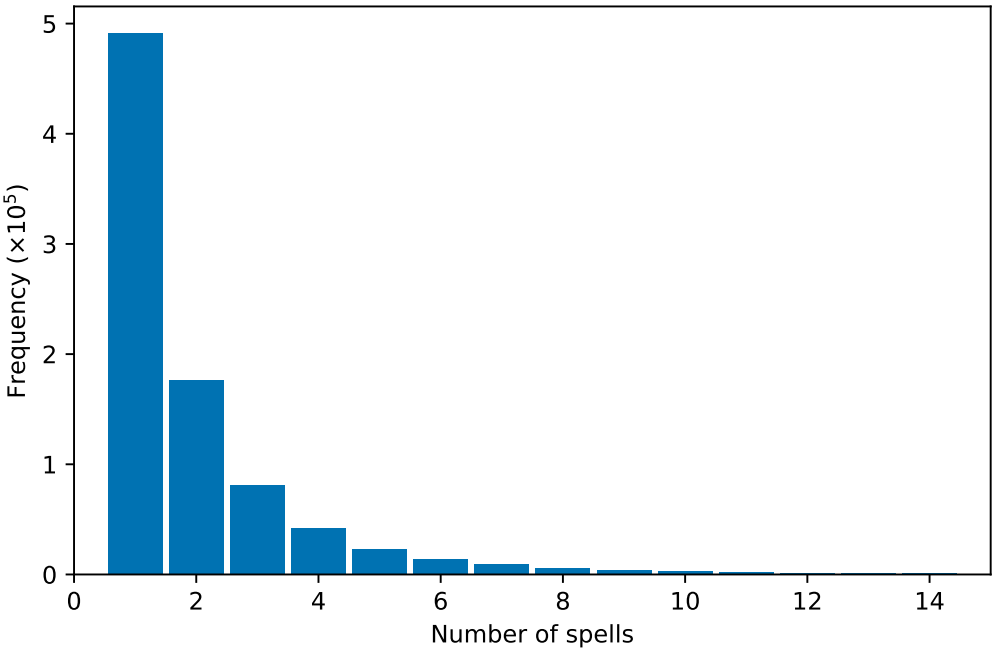


Figure B.2: Number of spells associated with each patient

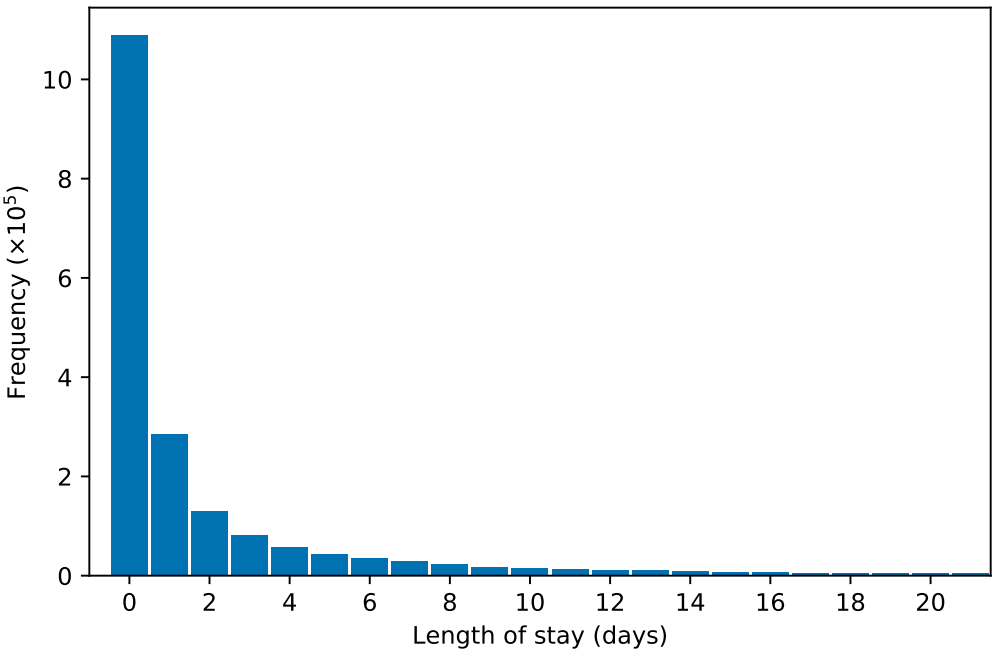


Figure B.3: Bar chart for length of stay

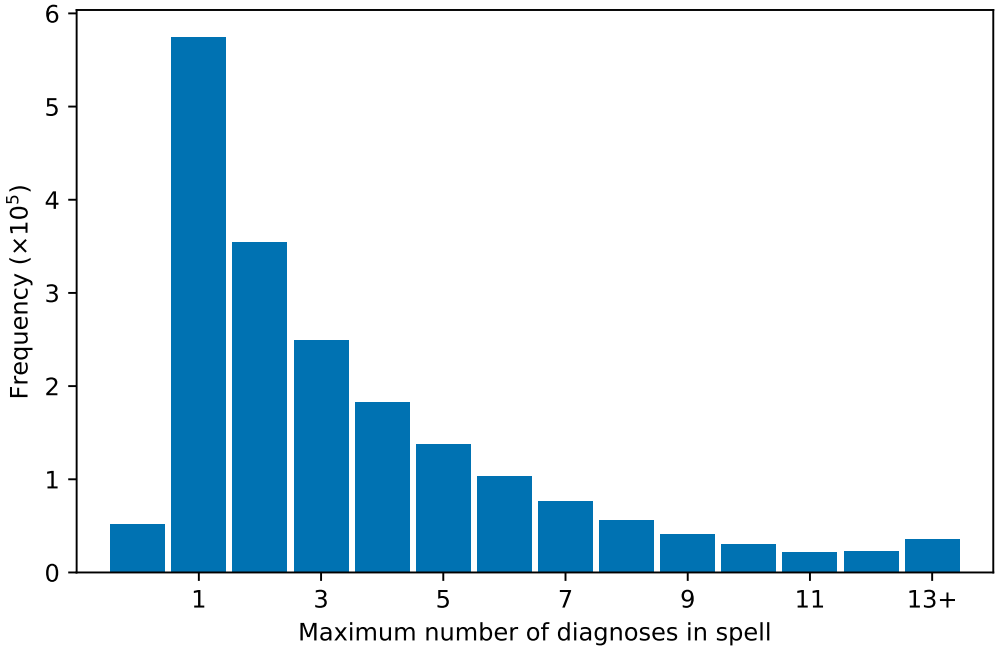


Figure B.4: Maximum number of diagnoses in each spell

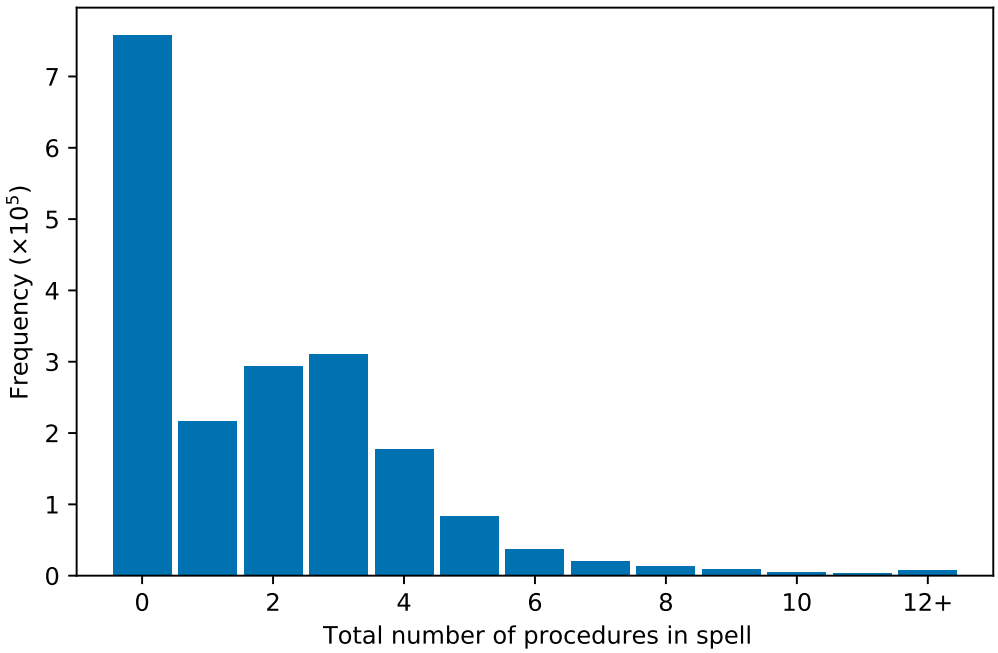


Figure B.5: Total number of procedures in each spell

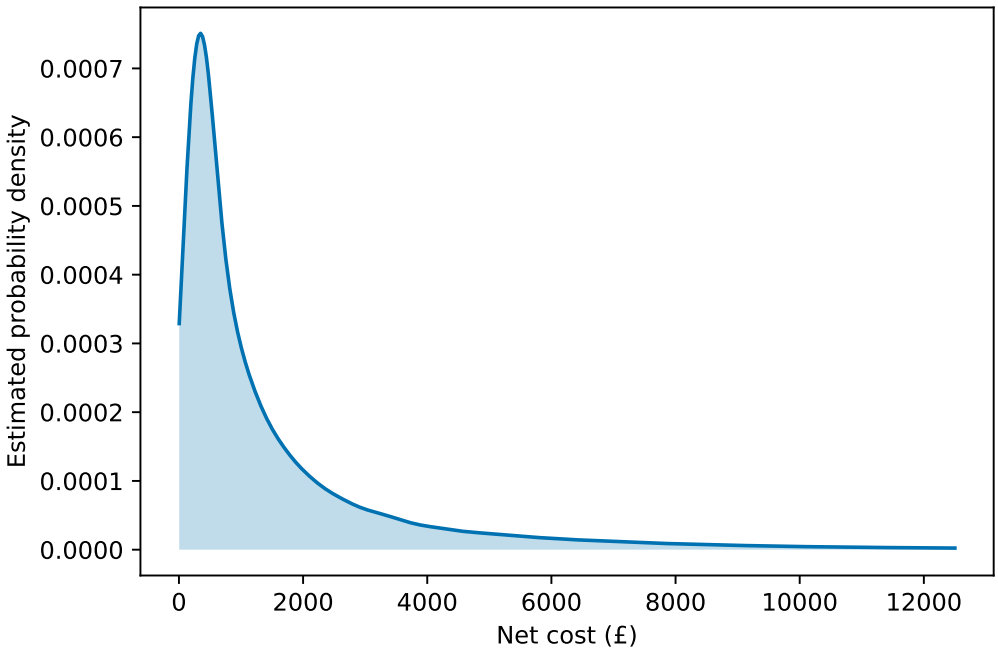


Figure B.6: Kernel density estimate for the net cost of a spell

this peak is little more than a blip; the most probable net cost has a likelihood of less than one tenth of a percent. The remaining values are distributed in a way that, given the scale, is near uniform, spanning from approximately £6,000 up to £369,000. A more detailed look at the skeleton of this distribution, and those of the remaining key attributes, is given in Table B.1.

Analyses of healthcare populations will canonically categorise patients by grouping ages together to aid the calculation of risk factors and projected costs. This approach has proven to be particularly helpful when looking at older patients [45], but is limited in scope as will be discussed in Section B.2. Bearing this in mind, however, studying the distribution of age among the patients can provide another valuable insight into how costs may appear.

Figure B.7 shows this distribution in contrast to a UK population estimate in 2016 from the Office for National Statistics (ONS). Following the graph from left to right, the UK estimate is roughly uniform from birth up until the late fifties where a decline appears as older people become less prevalent. Looking instead at the distribution belonging to the patients, it is clear that there are several peaks and troughs. The largest trough corresponds to adolescents, which makes sense anecdotally since some of the least likely people to visit a hospital would be reaching their peak healthiness. Similarly, the distinct peaks around infancy and in the older age range often correspond to those people who are most vulnerable in terms of their health. Thus, a hospital should expect to see a disproportionate number of people at those ages.

	mean	std	min	1%	25%	50%	75%	99%	max
COST	1,829.12	3,745.76	4.50	62.55	347.35	748.67	1,882.59	15,858.60	369,168.93
NetCost	1,737.65	3,160.53	4.50	62.55	347.07	745.51	1,859.00	14,183.24	369,168.93
CRIT	-91.48	1,327.49	-250,000.61	-2,205.96	0.00	0.00	0.00	0.00	0.00
DRUG	75.20	314.88	-0.57	0.00	7.18	19.93	59.88	837.10	63,430.52
EMER	1.24	29.15	0.00	0.00	0.00	0.00	0.00	1.13	33,347.89
ENDO	21.21	92.64	0.00	0.00	0.00	0.00	0.00	453.85	11,855.95
HCD	20.90	210.78	0.00	0.00	0.00	0.23	4.83	435.40	94,411.85
IMG	32.60	143.41	0.00	0.00	0.00	0.08	10.93	535.69	46,708.66
IMG_OTH	20.51	118.06	0.00	0.00	0.00	0.00	0.31	386.22	46,708.66
MED	346.40	735.11	0.00	0.00	44.45	130.63	374.93	2,947.14	116,449.90
NCI	-30.86	85.33	-12,960.21	-316.65	-29.75	-11.64	-3.03	0.00	0.00
NID	94.38	245.33	0.00	1.84	14.99	32.18	83.12	976.00	84,374.21
OCLST	13.27	58.62	0.00	0.00	0.00	0.77	5.43	263.86	12,358.37
OPTH	160.17	479.74	0.00	0.00	0.00	0.00	0.04	2,105.19	97,783.22
OTH	1.37	11.65	0.00	0.00	0.00	0.00	0.00	54.70	1,248.83
OTH_OTH	0.97	10.14	0.00	0.00	0.00	0.00	0.00	19.23	1,248.83
OUTP	0.58	26.81	0.00	0.00	0.00	0.00	0.00	0.00	10,632.15
OVH	353.72	726.91	0.00	25.86	84.86	139.47	320.24	3,243.31	91,511.45
PATH	36.05	135.06	0.00	0.00	0.00	4.60	31.77	399.14	70,008.12
PATH_OTH	23.22	122.42	0.00	0.00	0.00	0.00	13.71	315.59	70,008.12
PHAR	30.32	86.29	0.00	0.00	2.25	7.20	26.09	321.91	25,087.73
PROS	40.63	342.58	0.00	0.00	0.00	0.00	0.00	1,296.09	33,930.70
RADTH	0.65	8.02	0.00	0.00	0.00	0.00	0.00	0.00	227.64
SECC	0.87	27.45	0.00	0.00	0.00	0.00	0.00	10.42	2,177.74
SPS	11.82	149.54	0.00	0.00	0.00	0.00	0.00	208.62	68,029.58
THER	28.42	181.09	0.00	0.00	0.09	0.62	10.44	438.29	125,249.49
WARD	494.94	1,227.92	0.00	0.00	10.33	141.15	462.18	5,162.36	203,854.11
TRUE_LOS	2.84	8.57	0.00	0.00	0.00	0.00	2.00	38.00	705.00
DIAG_NO	3.47	2.95	0.00	0.00	1.00	2.00	5.00	13.00	13.00
PROC_NO	1.90	2.20	0.00	0.00	0.00	1.00	3.00	10.00	70.00

Table B.1: Spell-level statistics for each of the key attributes.

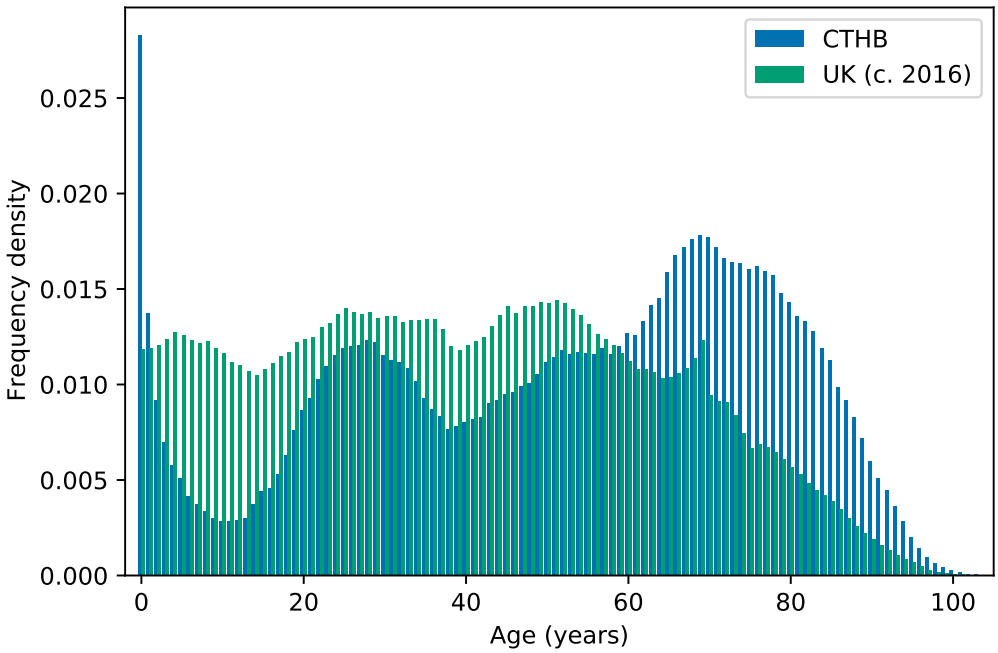


Figure B.7: Age of patients in the dataset compared with the estimated UK population in 2016

Looking at these key attributes, it would appear things are as expected: people tend to go to their local hospitals, and historically vulnerable people are more likely to go. However, there is significant spread in the costs, severity and lengths of hospital visits. Moreover, the likelihood of returning to hospital seems relatively low for the vast majority of the population served by the health board. While this analysis does not provide any unexpected insights, it is not a fruitless exercise as getting to grips with any body of data is essential. In fact, the analysis thus far has shown that this population is typical, in some (broadly anecdotal) respects, of many other populations.

B.1.4 Pairwise correlation

Looking at the univariate distributions of the key attributes in the previous section gave a good base for understanding the scope of the data. The next step is to investigate how these key attributes interact with one another. In this analysis, correlation coefficients will be used to give a sense of this interaction.

Figure B.8 shows the Pearson correlation coefficient between all the pairs of key attributes. These correlation coefficients have been presented in the form of a heat map with a colour bar, indicating the scale of the correlation between any two variables. The attributes themselves have been arranged into descending order according to their summed absolute correlation coefficient. This reordering makes it easier to

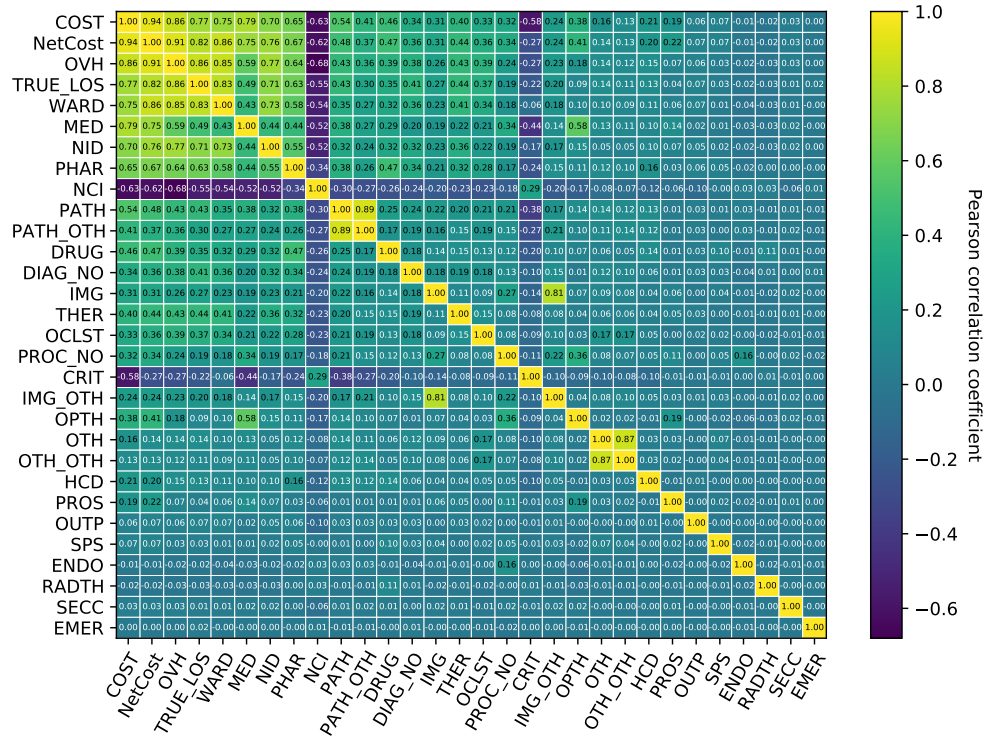


Figure B.8: Pairwise correlation coefficients for the key cost attributes

deduce which variables have the most prominent levels of interaction.

Definition B.1. Consider a dataset with $m \in \mathbb{N}$ columns, $A = \{A_1, \ldots, A_m\}$. Attribute A_j has associated with it a *summed absolute correlation coefficient*, c_j, given by:

$$c_j = \sum_{k=1}^{m} \left\| \rho_{A_j, A_k} \right\| \tag{B.1}$$

Here, ρ_{A_j,A_k} is the Pearson's correlation coefficient between attributes A_j and A_k.

Upon inspection of the heat map, there are many cost components that have no substantial linear correlation with any of the other attributes. The absence of correlation here only adds to the evidence that the patients in the data present themselves to hospital with a wide array of needs. Having said that, there are clear correlations between several of the attributes; some of these are easier to discern than others.

For instance, ignoring the main diagonal, the largest value is that between total costs (`COST`) and net costs (`NetCost`) with a value of 0.94. This high value indicates almost total positive linear correlation between these two variables, which makes sense given that the net cost of a spell is the total cost corrected for any reimbursable costs such as critical care costs (`CRIT`) and non-contracted income (`NCI`). Reimbursable costs are given as negative values in the dataset — hence their distinctly negative cor-

relation coefficients with the other variables. Typically, these deductible costs are small (see Table B.1) so a strong correlation between costs and net costs is to be expected.

Other examples of strong correlation are those between length of stay (TRUE_LOS) and ward and overhead costs (WARD and OVH respectively). These are well-known relationships that can be justified succinctly. The longer a patient spends in hospital, the more time they are likely to spend on a ward, incurring overheads like administrative work, cleaning costs and a larger proportion of rental costs. It should also be clear that these three attributes all share a strong linear correlation with the net cost of a spell, suggesting that these costs and the length of stay are strong indicators of the net cost of treating someone, and may suggest that the remaining cost components make up a substantially smaller part of the net cost.

B.1.5 Variation and relative importance

The broader purpose of this appendix is to better understand the factors leading to variation in the cost of treating patients. Therefore, it would be fitting to investigate how this variation can be attributed to each of the cost components. By doing so, a high-level indication of which departments and procedures that create more (or less) variation can be identified. Once a level of variation has been determined, the relative importance of that component and its variation can be assessed by considering the overall contribution that component makes to net costs.

In this section, and throughout this analysis, a dimensionless measure of variation will be used so that the cost components can be compared against one another. This measure is known as the coefficient of variation and is effectively the standard deviation scaled by the mean. While the sample variance, for instance, is a perfectly valid estimator for the variation of a variable, it is dependent on the scale of the data being considered. The effect of this non-scaling is evident in the standard deviations of Table B.1.

Definition B.2. Consider a population with mean μ and standard deviation σ. Then the *coefficient of variation*, denoted by C_v, is defined to be:

$$C_v := \frac{\sigma}{\mu} \tag{B.2}$$

If only a sample of the data from a population is available then the coefficient of variation can be estimated using the sample standard deviation and the sample mean.

Figure B.9 shows the coefficient of variation for each of the cost components. The

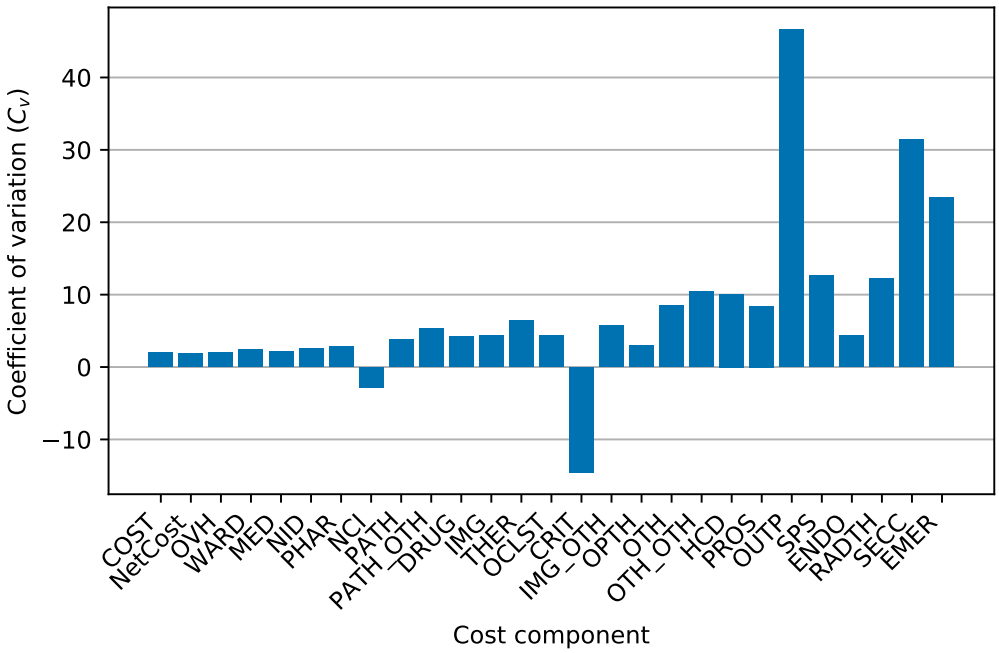


Figure B.9: Coefficient of variation of each cost component, and the net and total costs

components have been ranked as in Figure B.8 from the most to least correlated. It is immediately clear that there are a number of highly variant cost components. Take outpatient costs (OUTP) as an example: its standard deviation is over thirty times the size of its mean. This relative heterogeneity could go some way in explaining why there seemed to be no linear correlation with the other variables in Figure B.8.

At the other end of Figure B.8, ward and overhead costs have some of the smallest variations. This would suggest that they are in some way consistent or predictable, as was commented on in Section B.1.4. Despite this, the dominant conclusion is that all the cost components are still quite highly varied when considering the entire dataset since the majority of coefficients of variation found have size far greater than one.

Knowing which of the cost components are the most highly varied is not sufficient to decide whether they are worth pursuing further. To determine the importance of these components, the contribution of each cost component to the net cost of a spell should be considered. Then, with a sense of the scale of the variation acquired, the components that make the most significant impacts on net costs can be isolated. These quantities are calculated by taking each cost component in a spell, dividing it by its corresponding net cost and taking the mean over all of these values. This mean is referred to as the average contribution (or proportion) to the net cost, although it is more accurately an average of the spell-wise ratios between each cost component

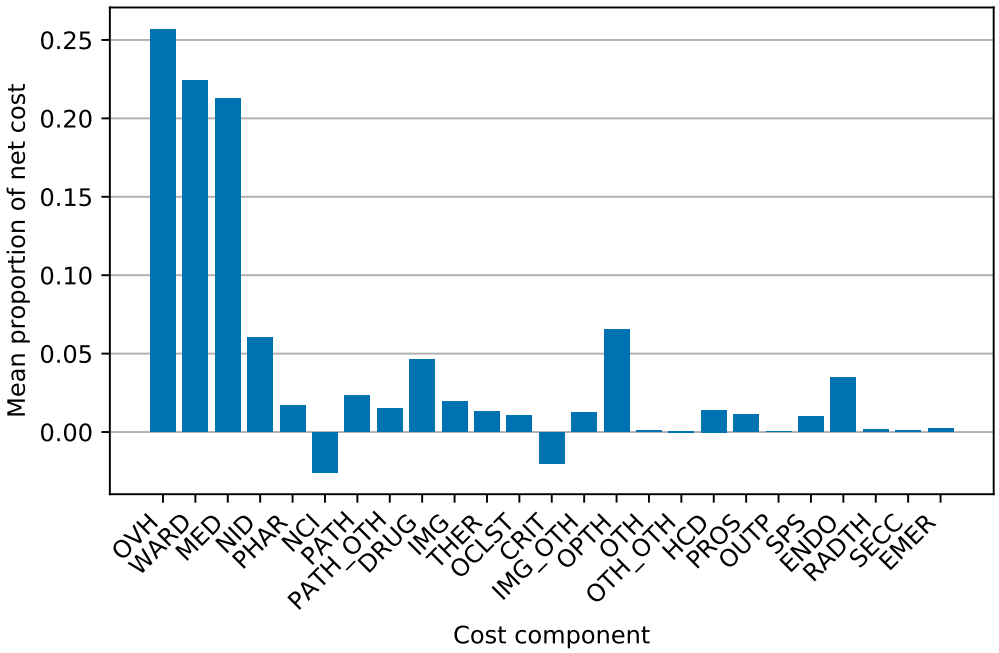


Figure B.10: Average contribution of each cost component to the net cost of a spell

and the net cost.

By inspecting Figure B.10, it is seen that ward, overhead and medical (MED) costs are the largest contributors to the net cost of a spell by a significant margin. When looking across the remaining bars, the contribution is substantially smaller for the department-specific cost components. Not only that but it appears that the most varied components (from Figure B.9) have near negligible average contributions to the net cost of a spell.

So the question left to be answered is: can these small but highly varied components be considered especially important? And what about the other components? The midriffs of each of these figures contain many of the same components but the relationships are less clear. In order to visualise how these two quantities relate to one another, a bubble plot is used. Such a plot allows for three-dimensional data to be displayed in the two-dimensional plane; by running their common variable along the horizontal axis, both of the quantities can be visualised simultaneously. The bubble plot in Figure B.11 uses the vertical axis and marker size to show net cost contribution and variation, respectively. The same ordering has been used for the components here as in the rest of the analysis.

This figure can be interpreted either by first reading along the vertical axis to find the components that make the most considerable contribution to treating a patient, and then investigating the variation that component holds by looking at the size of its outer marker. The reverse of this process is also perfectly logical since the objective

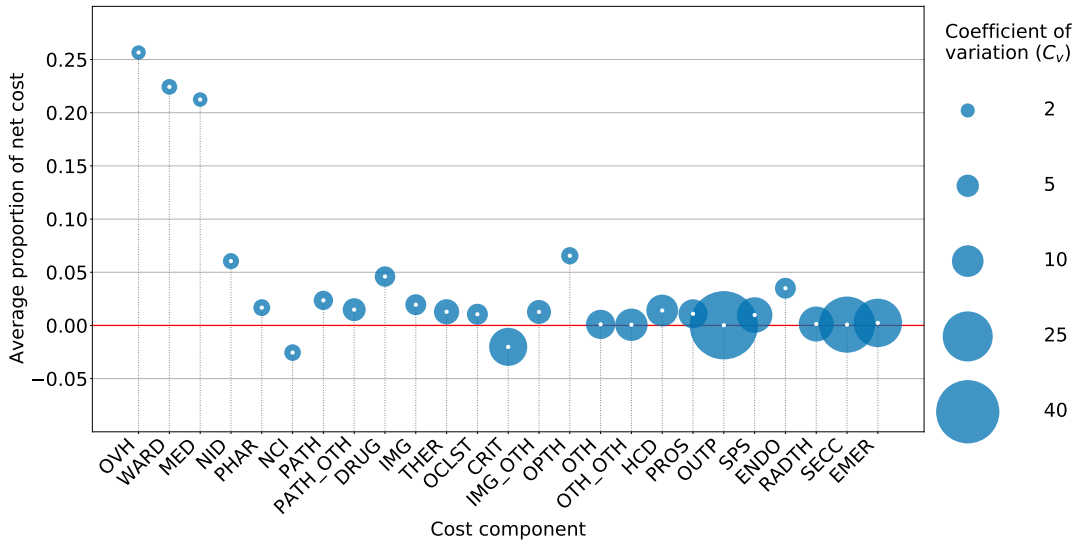


Figure B.11: A bubble plot showing the average contribution to the net cost of a spell and the coefficient of variation for each cost component

is to determine where the variation exists, and then how much of an impact that has on the net cost, as has been done above. The crux of interpreting this plot is that the further away a large marker is from the zero line, the more important that component is to be considered. However, small markers are also of interest since these components indicate that the level of variation is relatively low there, perhaps indicating the component has been optimised somehow.

This figure indicates that the conclusions made previously still hold: that the largest contributors have some of the smallest measures of variation. Meanwhile, the smallest average contributors are more strongly varied. What is of interest is the jump between these components and the others. There does not seem to be any particular component in the midriff of contributors that has large, or small, variation. As such, a deeper investigation is required to properly analyse individual components and their relationships with specific types of patient.

B.2 Diabetic patient analysis

The main conclusion to be taken away from the previous analysis was that the dataset contains a significant amount of variation. Therefore, in order to conduct more meaningful analysis, more homogeneous subsets of the data must be considered.

Classically, patients are categorised by age or condition. However, doing so often gives an unrepresentative slice of patients [370]. In this section, the focus will be on the diabetic population within the dataset, despite this potential danger, as it provides

a good example of condition-based slicing. Furthermore, diabetes is a condition of growing interest to public health research.

Since diabetes is recorded only as a primary or secondary condition in the dataset and is not distinguished by type, the diabetic population is considered to be any instance where diabetes is present.

The following analysis will provide evidence that the diabetic population is increasing in the Cwm Taf Morgannwg UHB, and that, despite this, the relative resource consumption by diabetic patients has been stagnant over the data period. It will also be seen that this population holds too much variation to make meaningful conclusions about the population on the whole. However, by considering a subset based on a condition such as this, there is a natural opportunity to compare the subset with its complement; by considering the differences and similarities between these two groups, a new dimension is added to the analysis.

B.2.1 Distributions and summary statistics

In much the same way as in Section B.1.3, taking an overview of the key attributes provides some idea about how costs are represented in the data. Figures B.12 through B.16 show the same statistics as in the summary analysis, although these figures have two additional components: (a) in the case of bar charts, separate plots for overall frequency and frequency density, and (b) a comparison with the non-diabetic population on the same axes. The purpose of the separate bar charts is to show the relative sizes of the groups, and then to be able to directly compare their distributions.

As before, the distributions of the diabetic population have long tails, but they are often heavier than the general or non-diabetic populations which are arguably interchangeable given their sizes. This extra weight in the tails suggests that diabetic patients are more likely to experience severe periods of illness, and this is bolstered by the complete difference in the shape of the distribution of maximum spell diagnoses pictured in Figure B.14.

Other than diagnosis numbers, the shapes of the distributions here are broadly equivalent, but the tails are heavier across the board for the diabetic population. With that being true, it follows that the noses are substantially lighter, which is most evident in Figure B.13 and Figure B.16. These figures (and the others in this group) imply that diabetic patients are more likely to return, have more procedures and stay longer in the hospital. As a result, they will typically incur higher costs than non-diabetic patients. All of these observations suggest that diabetic patients represent a population whose spells are more severe on average than the typical patient. Therefore, they will likely have a larger effect on the hospital system on the whole. A more

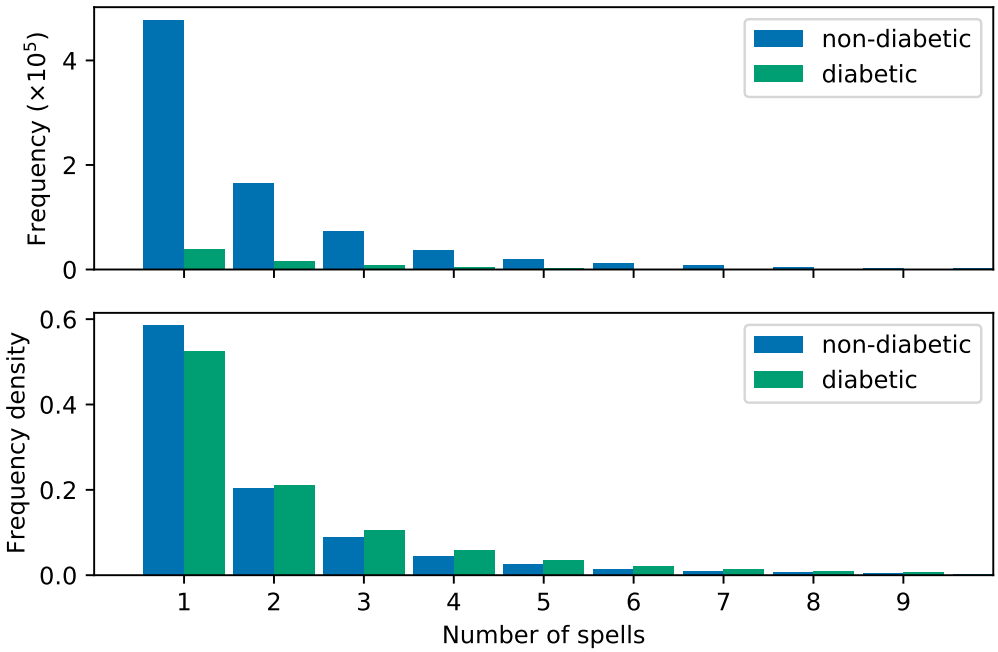


Figure B.12: Bar chart for the number of spells associated with a patient in the presence and absence of diabetes

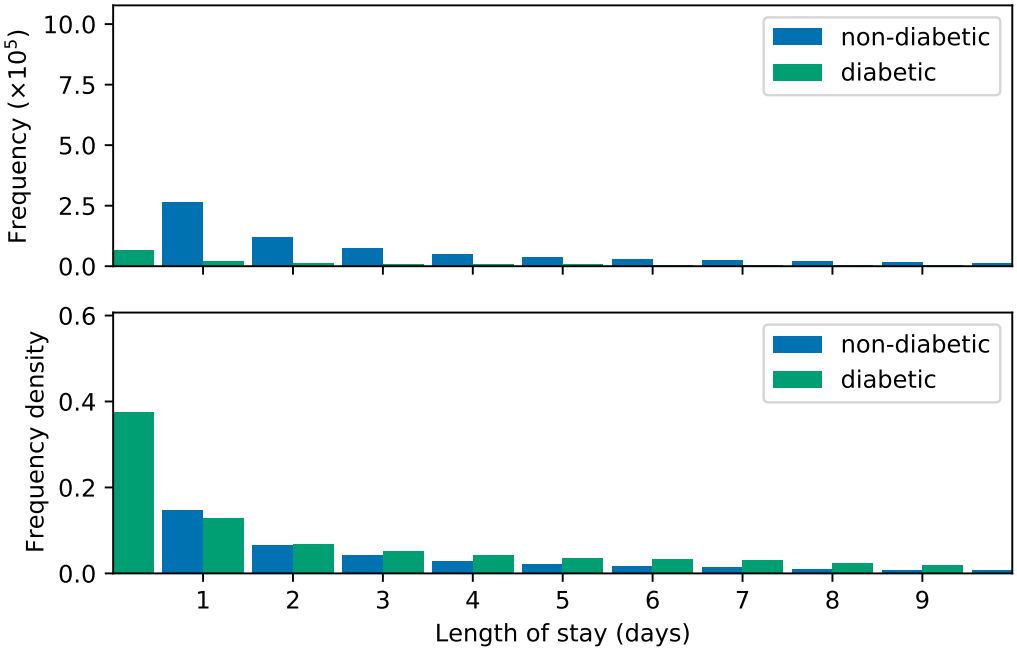


Figure B.13: Bar chart for the total length of a spell in the presence and absence of diabetes

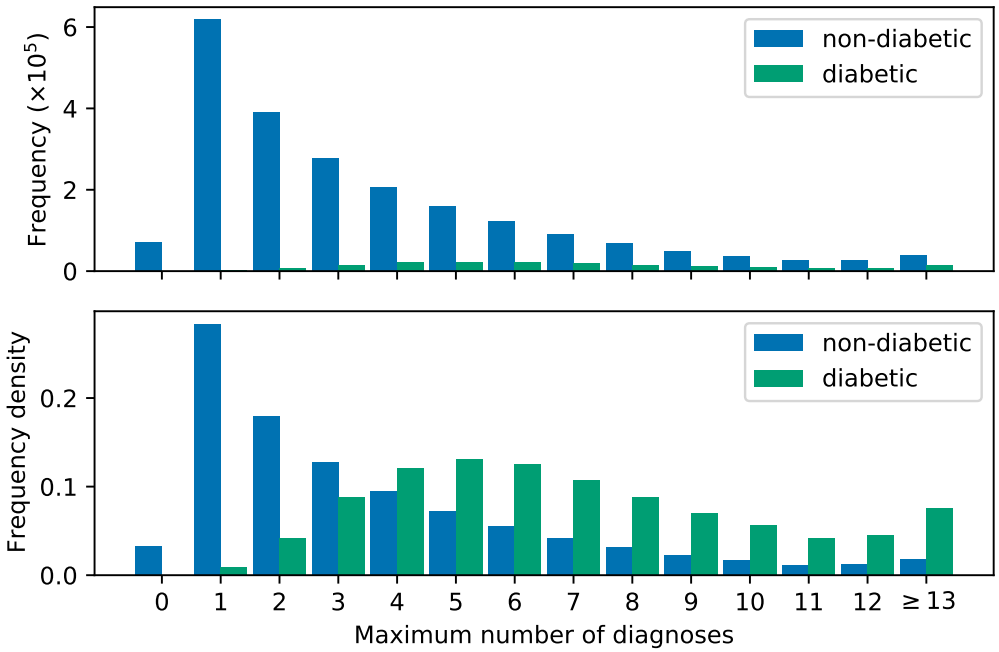


Figure B.14: Bar chart for the maximum number of diagnoses in a spell in the presence and absence of diabetes

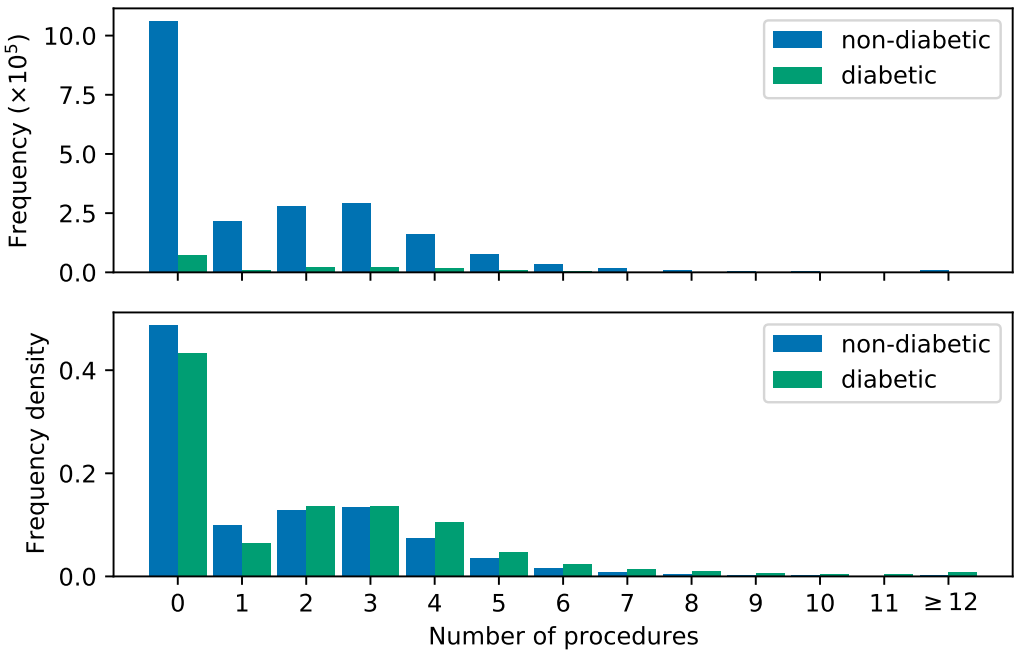


Figure B.15: Bar chart for the total number of procedures in a spell in the presence and absence of diabetes

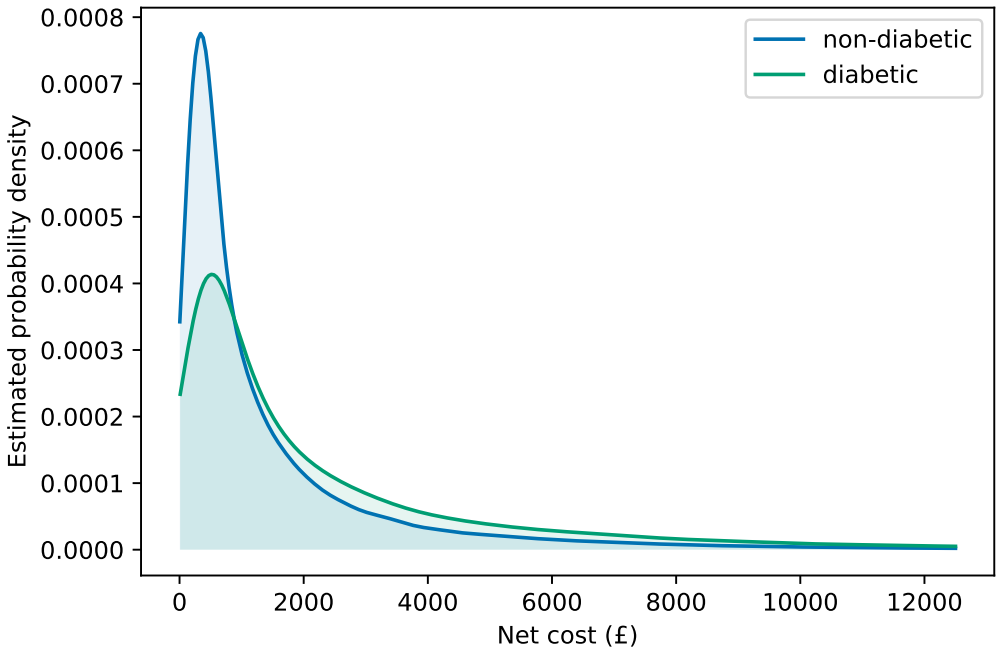


Figure B.16: Kernel density estimate for the net cost of a spell in the presence and absence of diabetes

detailed breakdown of the skeleton for each of these attributes as well as the other key attributes is given in Table B.2. This table also shows a comparison between both populations being considered in this section.

The patient age distribution for each group is given in Figure B.17. This figure shows how unrepresentative a slice the diabetic population can be. When looking at the frequency density plot, all the intricacies in the shape of the non-diabetic population are dropped. Instead, the distribution has a distinct negative skew with a disproportionate number of older patients. Thus, studying the entire diabetic population is akin to considering some subset of all older patients. In effect, this would ignore the younger diabetic population.

Conversely, the small number of younger diabetic patients left could be confusing the population somehow, and then any subsequent analysis of that population. A remedy for this would be to consider two or more diabetic populations based on their age and perhaps a combination of other attributes including severity or total cost. Deciding meaningful populations like these would require a significant amount of potentially arbitrary splitting on, or estimation of, such attributes. In Chapter 5, an automatic approach to separating a condition-specific population on features like these is used without the need for such clinical expertise or exertion.

	mean	std	min	1%	25%	50%	75%	99%	max
COST	2,801.26 (1,732.47)	4,755.10 (3,604.26)	10.91 (4.50)	140.16 (62.55)	493.10 (339.15)	1,242.98 (713.45)	3,191.26 (1,777.71)	21,380.12 (15,007.47)	273,450.30 (369,168.93)
NetCost	2,648.98 (1,647.00)	4,152.20 (3,019.53)	10.91 (4.50)	139.65 (62.55)	490.64 (338.67)	1,227.95 (709.32)	3,106.44 (1,756.90)	19,128.45 (13,414.48)	273,450.30 (369,168.93)
CRIT	-152.28 (-85.47)	1,543.66 (1,302.48)	-193,076.19 (-250,000.61)	-4,351.60 (-1,947.99)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)
DRUG	117.66 (70.98)	308.05 (314.59)	-0.24 (-0.57)	0.03 (0.00)	11.98 (6.70)	41.73 (18.97)	125.24 (55.12)	1,077.62 (790.91)	39,100.44 (63,430.52)
EMER	1.49 (1.22)	18.94 (29.92)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	12.06 (1.13)	1,274.44 (33,347.89)
ENDO	17.92 (21.49)	86.49 (93.10)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	459.95 (452.73)	2,930.77 (11,855.95)
HCD	30.88 (19.90)	282.12 (202.23)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.78 (0.20)	8.47 (4.18)	538.46 (421.83)	31,451.98 (94,411.85)
IMG	57.82 (30.12)	173.69 (139.60)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.96 (0.07)	38.02 (5.68)	760.00 (496.25)	8,097.57 (46,708.66)
IMG_OTH	37.11 (18.88)	137.35 (115.64)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	14.20 (0.31)	622.04 (359.49)	8,097.57 (46,708.66)
MED	442.80 (336.51)	823.33 (723.61)	0.00 (0.00)	2.33 (0.00)	67.48 (42.63)	193.30 (125.47)	478.28 (364.67)	3,630.58 (2,853.92)	58,673.47 (116,449.90)
NCI	-47.74 (-29.19)	111.85 (81.90)	-6,663.12 (-12,960.21)	-462.48 (-297.09)	-48.25 (-28.27)	-18.62 (-11.36)	-5.62 (-2.95)	0.00 (0.00)	0.00 (0.00)
NID	156.84 (88.22)	350.59 (230.71)	0.00 (0.00)	2.65 (1.84)	21.22 (14.52)	51.42 (31.14)	169.79 (76.98)	1,396.24 (916.69)	68,821.61 (84,374.21)
OCLST	23.79 (12.24)	86.84 (54.85)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	1.83 (0.77)	12.30 (5.06)	356.95 (243.31)	5,155.60 (12,358.37)
OPTH	157.82 (160.10)	554.75 (471.42)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.04)	2,310.35 (2,083.16)	97,783.22 (51,651.76)
OTH	3.03 (1.20)	17.35 (10.92)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.25 (0.00)	94.37 (38.46)	787.82 (1,248.83)
OTH_OTH	2.09 (0.86)	14.90 (9.53)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	79.99 (10.10)	787.82 (1,248.83)
OUTP	1.44 (0.49)	50.43 (23.29)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	10,632.15 (9,989.54)
OVH	578.90 (331.46)	983.48 (689.86)	0.00 (0.00)	43.77 (20.22)	107.56 (83.78)	230.05 (135.46)	663.48 (296.93)	4,548.67 (3,037.17)	57,647.29 (91,511.45)
PATH	63.95 (33.31)	175.98 (129.62)	0.00 (0.00)	0.00 (0.00)	0.67 (0.00)	20.01 (3.72)	71.03 (28.55)	589.39 (370.62)	28,621.00 (70,008.12)
PATH_OTH	42.12 (21.37)	159.98 (117.55)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.74 (0.00)	35.24 (12.38)	486.22 (290.02)	28,621.00 (70,008.12)
PHAR	58.15 (27.60)	124.21 (80.90)	0.00 (0.00)	0.02 (0.00)	3.75 (2.13)	16.13 (6.74)	71.52 (23.22)	479.20 (295.96)	14,812.14 (25,087.73)
PROS	54.56 (39.22)	435.57 (331.92)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	1,569.75 (1,263.77)	28,955.99 (33,930.70)
RADTH	0.50 (0.67)	7.24 (8.08)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	227.64 (227.64)
SECC	1.00 (0.86)	21.45 (27.94)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	20.83 (10.42)	1,813.69 (2,177.74)
SPS	21.49 (10.87)	190.25 (144.70)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	799.16 (208.62)	14,008.47 (68,029.58)
THER	57.23 (25.61)	207.44 (177.75)	0.00 (0.00)	0.00 (0.00)	0.18 (0.08)	7.53 (0.50)	47.84 (8.43)	684.15 (407.23)	17,643.81 (125,249.49)
WARD	843.02 (460.63)	1,673.72 (1,165.64)	0.00 (0.00)	0.00 (0.00)	59.64 (9.04)	271.67 (136.97)	986.61 (429.02)	7,244.42 (4,855.75)	173,963.47 (203,854.11)
TRUE_LOS	6.07 (2.57)	12.55 (8.13)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	1.00 (0.00)	7.00 (2.00)	57.00 (35.00)	705.00 (690.00)
DIAG_NO	6.89 (3.14)	3.15 (2.72)	1.00 (0.00)	2.00 (0.00)	4.00 (1.00)	6.00 (2.00)	9.00 (4.00)	13.00 (13.00)	13.00 (13.00)
PROC_NO	2.05 (1.88)	2.58 (2.16)	0.00 (0.00)	0.00 (0.00)	0.00 (0.00)	2.00 (1.00)	3.00 (3.00)	12.00 (9.00)	43.00 (70.00)

Table B.2: Spell-level statistics for each of the key attributes in the diabetic population (and non-diabetic population in parentheses)

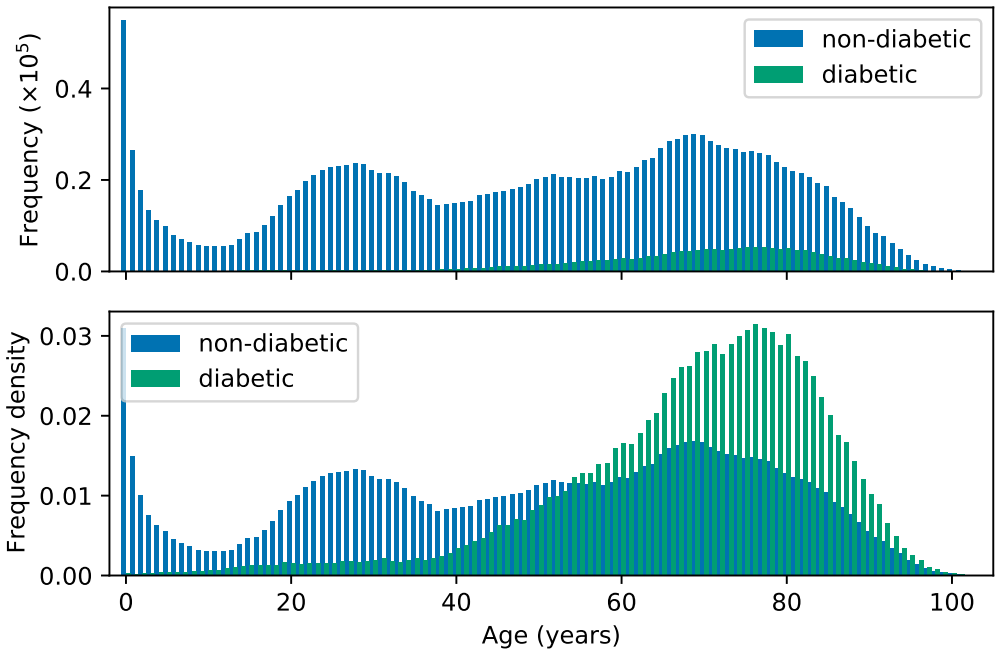


Figure B.17: Bar chart for the age of patients in the presence of diabetes and not

B.2.2 Pairwise correlation

With an overview of how the key attributes are distributed in mind, as before, it is a good idea to see how these attributes interact with one another. In Figure B.18, the Pearson correlation coefficients are shown between each of the pairs of the key attributes in the diabetic population.

Again, the attributes have been ranked in descending order according to their summed absolute correlation coefficient (B.1) to determine those with the highest levels of interaction. The correlation matrix for the non-diabetic population has been omitted as it is similar to that of the general population.

To more directly see the distinctions between these correlation coefficients and those of the general population (shown in Figure B.8), another heat map has been included to show their differences in Figure B.19. This heat map utilises a different colour map to reflect this, and the attributes have been ranked in descending order of their summed absolute differences.

This figure indicates that drug and therapy costs (DRUG and THER respectively) have the largest total difference in correlation coefficients. In fact, the signs of each coefficient are the same, and so we can say that the diabetic population has more strongly correlated drug and therapy costs. The same pattern occurs with a number of other cost components, including the core components from the general population: ward, medical and overhead costs. These increased correlation coefficients may be indica-

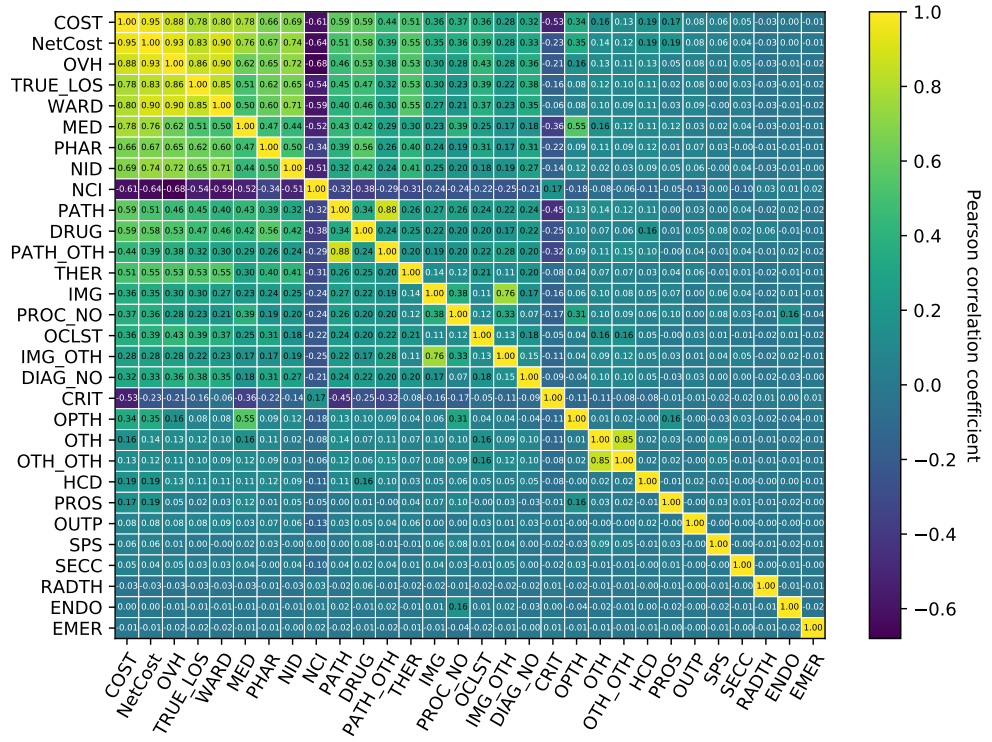


Figure B.18: A heat map of the pairwise correlation coefficients for the key attributes in diabetic patients

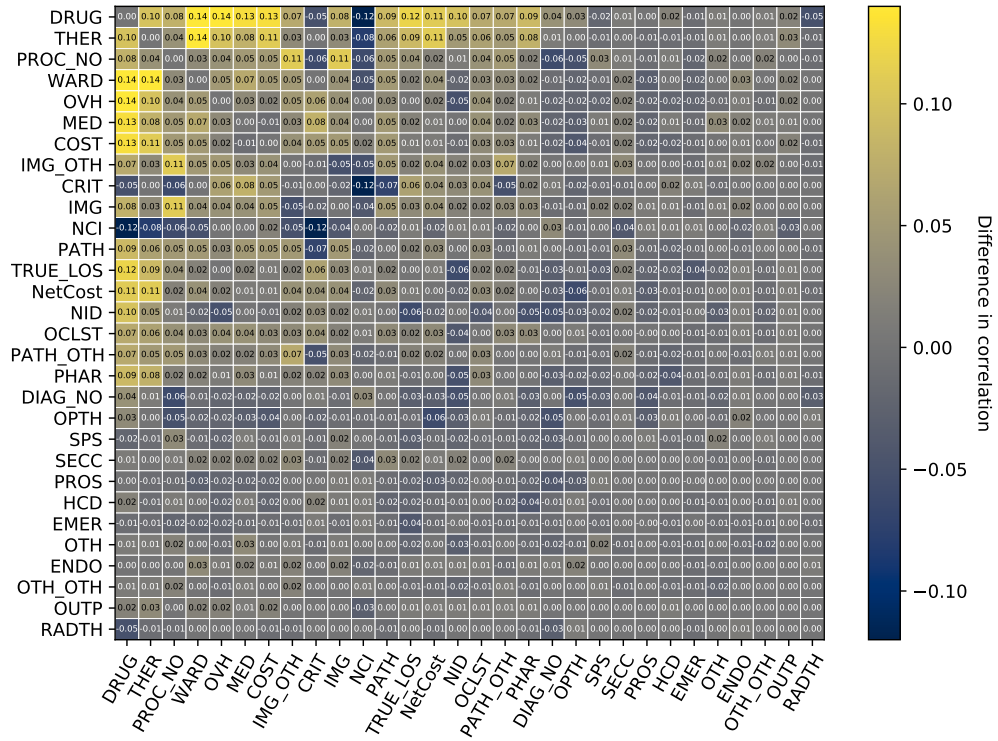


Figure B.19: A heat map of the difference in pairwise correlation coefficients between the diabetic and general populations

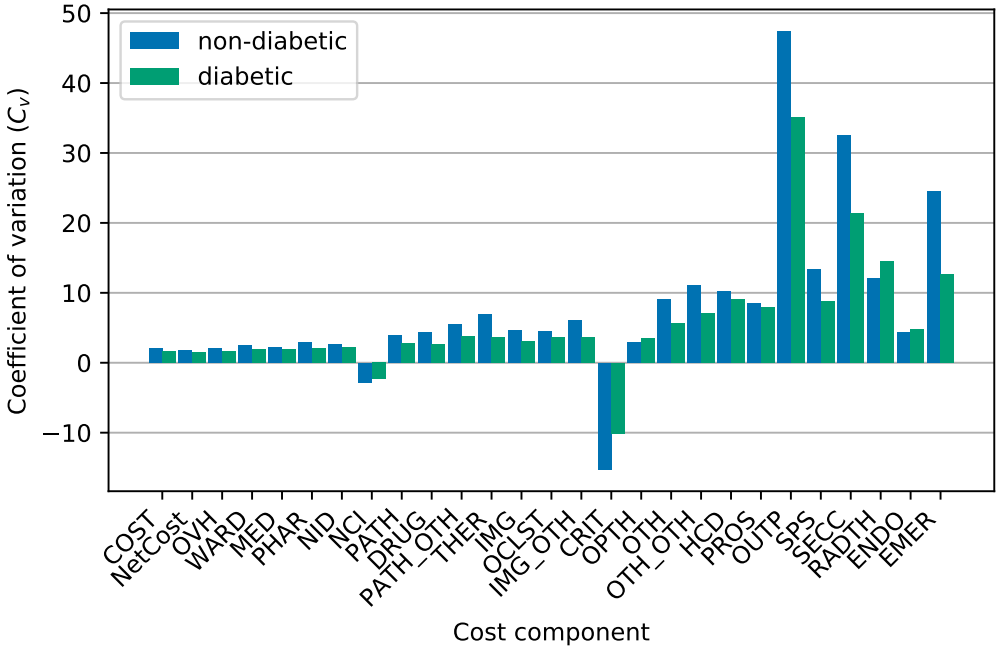


Figure B.20: Bar chart showing the coefficient of variation C_v of each cost component, and the net and total costs, in the presence of diabetes and not

tive of some intrinsic property the diabetic population holds. However, they could merely be a coincidence of considering a subset of the dataset that is, by default, more homogeneous.

Other than the few attributes at the top, this heat map shows that the vast majority of correlation coefficients are unaffected by considering the diabetic population alone. Given the large amounts of variation and low levels of correlation seen in Section B.1.4, this is unsurprising, but where there are differences suggests potential areas of interest when comparing the corresponding diabetic variation with the non-diabetic and general populations.

B.2.3 Variation and relative importance

The distributions of the key attributes, and some notion of their interactions, have been established. The remaining stage of the methodology established in Section B.1 is to investigate variation and the relative importance of the cost components themselves. Figures B.20 and B.21 show these quantities, and are ranked as in Figure B.18.

Aside from the change in the order of the attributes compared with Figure B.9, this plot is largely similar: more weakly correlated attributes tend to be more highly varied and the overall level of relative variation is high. Having said that, the diabetic population is consistently less, or similarly, varied than the non-diabetic population in each instance; this is true except for operating theatre (OPTH), radiotherapy

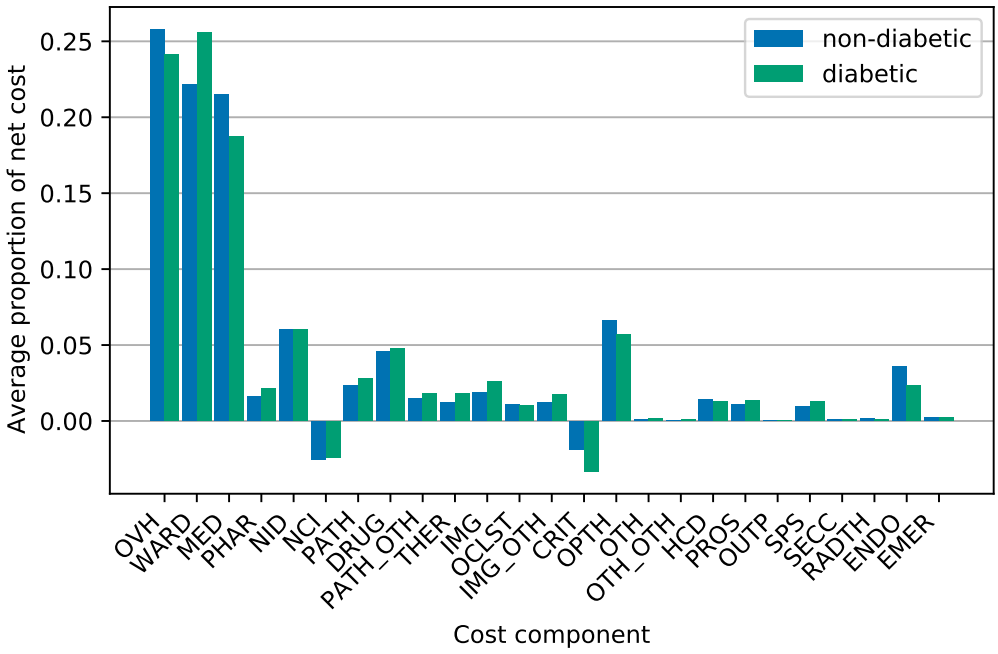


Figure B.21: Bar chart showing the average contribution of each cost component to the net cost of a spell in the presence of diabetes and not

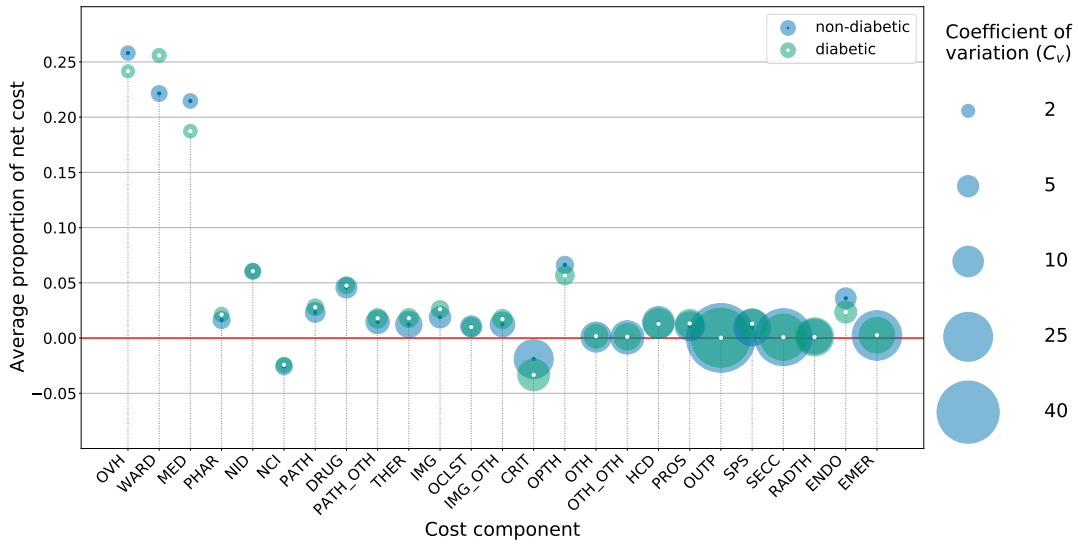


Figure B.22: A bubble plot showing a comparison between the diabetic and non-diabetic populations' average contribution to the net cost of a spell, and the coefficient of variation, for each cost component

(RADTH) and endoscopy (ENDO) costs. Regardless, the decrease in overall cost variation indicates that this subset of the dataset is in fact somewhat more homogeneous, as has been noted, and as is desired.

Inspecting Figure B.21 tells a similar story to that of the general population: the dominant cost components are still overheads, medical and ward costs, and the least correlated (and often most varied) components are insignificant in their contributions to net costs. However, there is a certain interest in the increased contribution from ward costs and those from specific departments such as pharmacy (PHAR), pathology (PATH), and imaging (IMG). The apparent increase in the likelihood, severity and length of diabetic patient spells seen in Table B.2 — and the figures in Section B.2.1 — seems to be linked to a rise in costs more generally. This increase can be rationalised given that the patients within this population all exhibit at least one chronic condition that is known to have several comorbidities and knock-on effects more widely associated with the well-being of a patient [97, 202, 371].

As in the previous section, the bubble plot in Figure B.22 allows these quantities to be considered simultaneously. Again, and despite the efficacy of the visualisation itself, there is little insight to be gained. There are no distinctly important components here and the system seems to be optimised for both the diabetic and non-diabetic populations. This 'optimisation' is only up to the point where the smallest relative variation of a component is still twice its mean.

B.2.4 Resource consumption

The types of comparisons made between the non-diabetic and diabetic populations throughout this analysis are useful for observing their similarities in a direct way, and in understanding how the groups may relate to one another. However, these are not the only devices available for examining such a subset of the data. Particularly when looking at costing data such as this, another useful way of evaluating a subset is to quantify its contributions to those costs over time within the general population. The attributes used to identify these contributions can give a sense of the level and nature of the resources that are consumed by the population in question. This stage of the analysis considers the following attributes: the proportion of net costs and admissions, and the length of stay.

For these purposes, the data must be manipulated into a chronological form. Here, each of the chosen attributes is given with respect to a particular admission date, and has been calculated in the following way for each admission date:

- Proportion of total admissions. Take the number of unique spells for diabetic patients admitted on that day, n_d, and the total number of unique spells with

that admission date, N. The proportion of total admissions on that day from diabetic patients is given by $\frac{n_d}{N}$.

- Average length of stay. Take the mean over all lengths of stay from the diabetic spells with that admission date.

- Proportion of net costs. Take the net cost for each diabetic spell beginning on that admission date and sum them, denote this by c_d. Do the same with the net cost of all spells with that admission date and denote this by C. Then the proportion of net costs spent on diabetic patients is given by $\frac{c_d}{C}$.

The key benefit of taking the quantities in this way is that it allows for the data to be arranged with some sense of time. However, there is a glaring issue: that the data will be misrepresented when manipulated in this way. For instance, the length of a spell has no definitive connection to the admission date of that spell; by grouping all the spells starting on that day together and taking their mean, any adversely long spells will push the mean upwards. Also, there is a time-related error when taking the net cost of a spell on any one day in that spell since that cost was not necessarily spent or incurred on that day.

Irrespective of these misrepresentations, Figures B.23 through B.25 show how these quantities evolve over the entire data period. In each case, the weekly, monthly and yearly means are shown. The data has been aggregated in this way, rather than using the daily data, in an attempt to smooth out the misrepresentation that is described above. In addition to these plotted points, the data has been fitted with a standard least-squares linear regression model.

Figures B.23 and B.24 suggest that the amount of resources consumed by the diabetic population is increasing slowly. The former indicates that, on average, the number of diabetic patients visiting the hospital is increasing at a rate of approximately one percent over five years. From the latter, it is seen that the yearly average proportion of net spending on diabetic patients has also experienced a shallow increase of roughly half a percent over the same period.

In addition to this, both figures show some form of divergence over time, as shown by the spread in the weekly and monthly averages. This is an interesting phenomenon; there seems no apparent reason for this variability to increase in recent years with improved policy on prevention, diagnosis, management and treatment [262, 257, 284, 292]. The drops toward the end of the period in Figures B.24 and B.25 are a result of both the chronological encoding and the data period ending. Towards the end of the period, there are fewer long-term patients included in the dataset as they have not completed their spell yet.

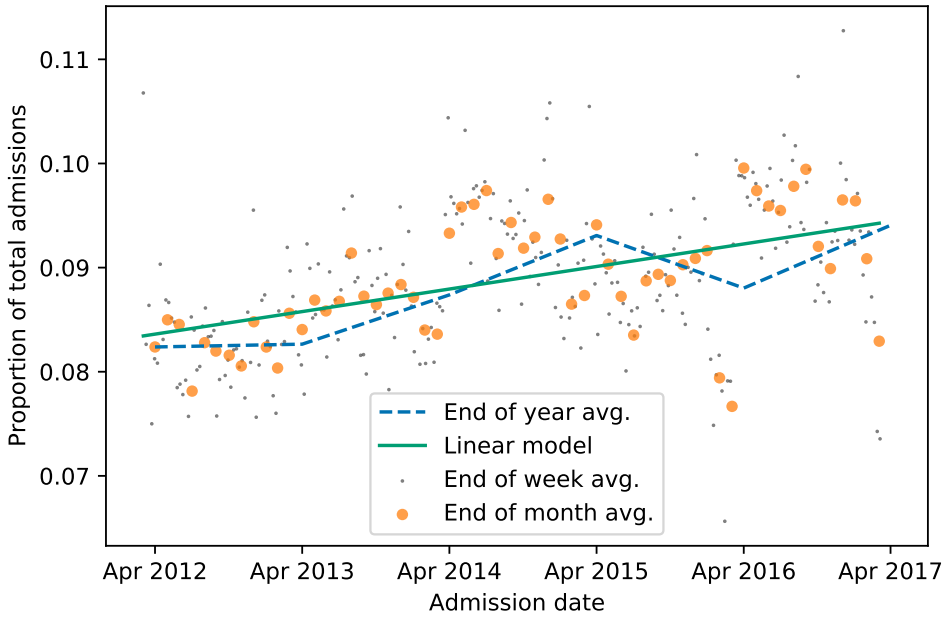


Figure B.23: Monthly averages for the proportion of daily admissions presenting diabetes

With Figure B.25, it is clear that — despite the slight increase in the proportion of net costs and the number of diabetic admissions over the last five years — there has been a slight decline in the average length of stay for diabetic patients in the same period. This average has fallen from one week to roughly five and a half days. This decrease is likely due, in part, to the changes in NHS policy referenced above but also the ever-increasing pressure put on the hospital system to move patients through the system efficiently in order to save on idle costs such as ward costs and overheads.

Across each of the models displayed here, there appears to be some seasonality. The handling of seasonal behaviour in a regression model has more to do more with the semantics of finding a "good" regression model than was intended here but it is an important concept nonetheless. If the purpose of this exercise was to accurately predict the quantities being plotted, rather than only seeing the general trend, then a more elaborate model should have been fitted.

B.3 Conclusion

This appendix presented a descriptive analysis of an administrative dataset provided by the Cwm Taf Morgannwg UHB. This dataset describes the episodes of all patients being treated at two of their hospitals over a five-year period. The scope of this exploration of the dataset was thwarted by the span and diversity of the episodes therein. In fact, there was little insight to be gained beyond it conforming to some

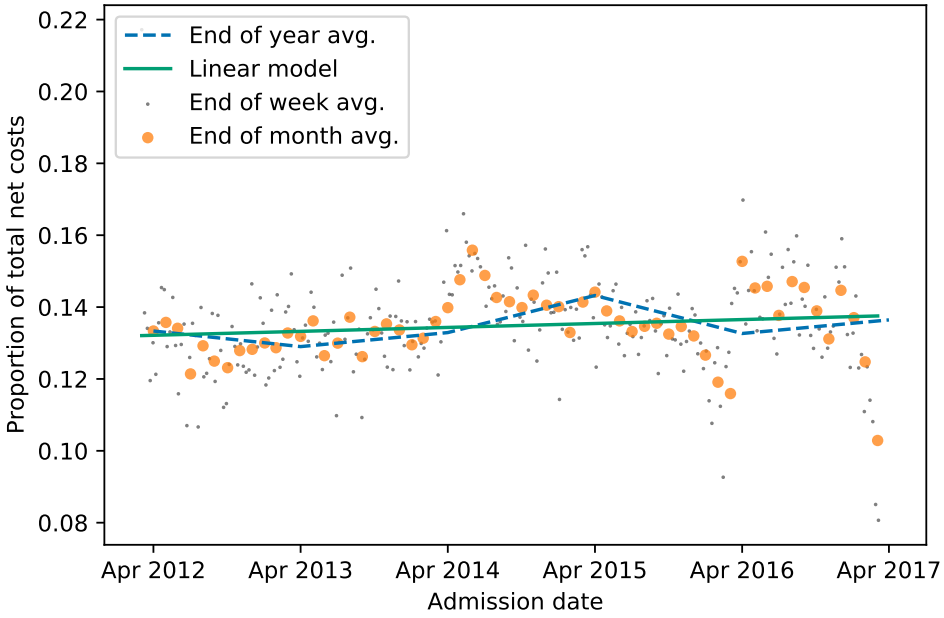


Figure B.24: Monthly averages for the proportion of daily net cost spending toward diabetic patients given their admission date

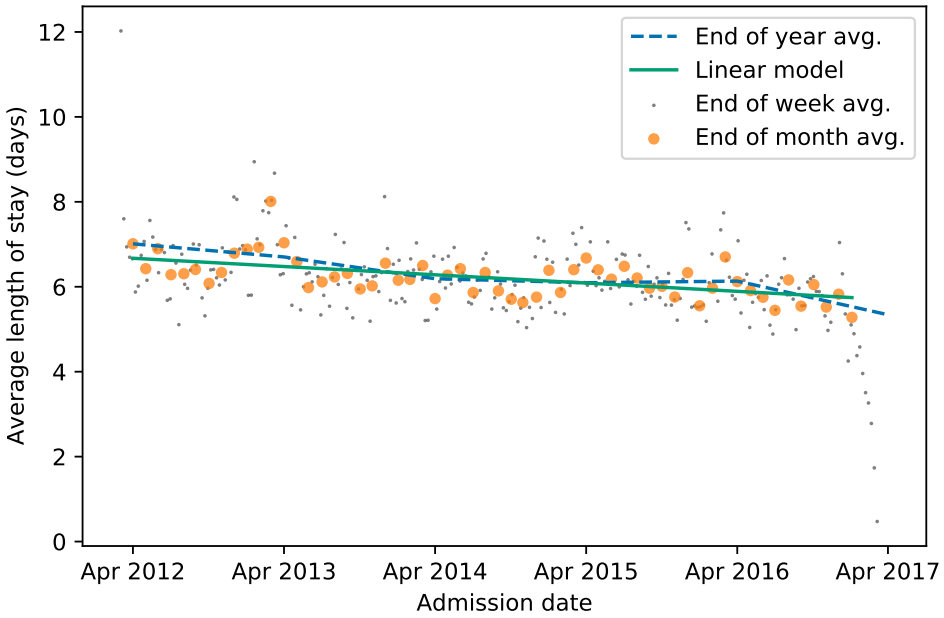


Figure B.25: Monthly averages for the average length of a diabetic patient's spell given their admission date

loosely anecdotal concept of an administrative healthcare dataset.

After this initial analysis, a similar methodology was employed on a slice of the dataset. Following advice from the subject matter expert, the diabetic population was used. The hope of that analysis was that taking a slice according to some characteristic would result in a more homogeneous dataset, which was confirmed, although only marginally. Other than this, there were few actionable insights to be gained from this sort of exploratory analysis.

So what was there to be gained by looking at the diabetic population? The lack of novel insights may have been expected since the decision to look at diabetic patients was effectively arbitrary. Moreover, the decision framework was not descriptive enough to indicate that any particular kind of patient was being investigated other than that they must exhibit this one condition. So, in that way, there was little to gain. However, as has been noted throughout this chapter, taking a subset of the population allows for some comparison with its complement, as well as the general population.

Chapter 5 considers another administrative dataset comprised of patients presenting another chronic condition. The work in that chapter relies on the clustering algorithm presented in Chapter 4, and builds on the surface-level analyses employed here. The most important of these analyses is the slicing of a dataset to identify more homogeneous parts, and then comparing those. The process by which this is done incorporates more attributes of the data than the condition alone, or age, as is commonly done. Ultimately, Chapter 5 provides genuine, actionable insights into the healthcare population under study.

Appendix C

Automatic final-year project allocation in a School of Biosciences

C.1 Introduction

For many undergraduate students, a crucial part of their degree is their final-year project (FYP). This piece of work characterises their interests and allows the student to demonstrate their command of a chosen subject. Being assigned a favourable FYP topic is of great importance. Good allocation affects the student experience, improving student-supervisor relationships, engagement, and, eventually, satisfaction [58, 208].

However, as the ratio between students and university staff increases [239], so does the need for fair and efficient FYP allocation systems. This need is both practical and pedagogic. Practical in that as cohort sizes increase, the demands on administrative staff grow, meaning manual systems eventually become infeasible. Pedagogic, given the impacts of good project allocation on learning.

FYP allocation is a resource allocation problem with a specific set of constraints. Typically, these correspond to student preferences, supervisor preferences, and workload capacities. There are many techniques available for finding solutions to this allocation problem, and [170] provides a review of current FYP allocation methodologies. A common approach is linear programming [76, 213]. This appendix uses the student-project allocation problem (SA) to carry out FYP allocation. Implementing FYP allocation as an instance of SA grants access to a Gale-Shapley algorithm, which produces a unique, student-optimal, mathematically fair allocation.

C.2 Using the `matching` library

This section is in preparation, and some of the details are covered in Section A.3, where SA is introduced. However, a full tutorial on how to implement this FYP allocation process is available in the `matching` documentation:

`matching.readthedocs.io/.../project_allocation/main.html`

C.3 Case study

This section is in preparation, but will include a case study of FYP allocation in the School of Biosciences at Cardiff University (BIOSI). The automatic, matching-based process has been used since the 2019/20 academic year to allocate projects to final-year students in BIOSI. The School accepts applications from all of their final-year students, of which there are approximately 360 in any given year.

C.4 Conclusion

This section is in preparation. However, there are two major findings from implementing this FYP allocation.

First, the automatic allocation process reduces the work hours required by the staff dramatically. There are half a dozen or so staff members within the FYP team, and their manual allocation process would typically take at least one week to create a draft allocation to be distributed to the supervisors. Following that, adjustments are made until the BIOSI staff are satisfied, after which the projects are released to the students. Even at this point, project allocations may change owing to students making individual requests.

With the new implementation, the initial matching completes within a few seconds. Included in the process are several figures for analysing the allocation, including supervisor-project utilisation and allocation-rank quality.

Second, the allocation itself is guaranteed to be fair and optimal for students. By using the Gale-Shapley algorithm, it becomes far easier to justify particular allocations to students and staff. This resolves many of the adjustment issues experienced by the School prior to implementing this allocation process.