PREDICTION OF SURGICAL OPERATION DURATIONS USING SUPERVISED MACHINE LEARNING TECHNIQUES

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF INFORMATICS OF
THE MIDDLE EAST TECHNICAL UNIVERSITY
BY

EHSAN ZABARDAST

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF MEDICAL INFORMATICS

AUGUST 2017

PREDICTION OF SURGICAL OPERATION DURATIONS USING SUPERVISED MACHINE LEARNING TECHNIQUES

Submitted by EHSAN ZABARDAST in partial fulfillment of the requirements for the degree of **Master of Science in The Department of Medical Informatics Middle East Technical University** by,

Prof. Dr. Deniz Zeyrek Bozşahin _____
Director, **Graduate School of Informatics**

Assoc. Prof. Dr. Yeşim Aydın Son _____
Head of Department, **Health Informatics**

Assist. Prof. Dr. Aybar Can Acar _____
Supervisor, **Health Informatics**

Assist. Prof. Dr. Marie Persson _____
Co-Supervisor, **Computer Science**

Examining Committee Members:

Prof. Dr. Tolga Can _____
Computer Engineering, Middle East Technical University

Assist. Prof. Dr. Aybar Can Acar _____
Health Informatics, Middle East Technical University

Assist. Prof. Dr. Marie Persson _____
Computer Science, Blekinge Institute of Technology

Assist. Prof. Dr. Rahime Belen Sağlam _____
Computer Engineering, Yıldırım Beyazıt University

Assist. Prof. Dr. Nurcan Tunçbağ _____
Health Informatics, Middle East Technical University

Date: _____

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this wok.

Name, Last name : Ehsan Zabardast

Signature : _____

ABSTRACT

PREDICTION OF SURGICAL OPERATION DURATIONS USING SUPERVISED MACHINE LEARNING TECHNIQUES

Zabardast, Ehsan

MSc., Department of Medical Informatics

Supervisor: Assist. Prof. Dr. Aybar Can Acar

Co-Supervisor: Assist. Prof. Dr. Marie Persson

August 2017, 99 pages

There's an ever increasing number of patients referred to healthcare facilities and hospitals. The healthcare facilities have two main options to deal with this situation. They have to either employ and acquire more resources or they should use the existing staff and resources more efficiently and effectively. The first option is not always feasible due to the fact that the healthcare facilities have limitations on both the staff they can employ and the resources they can acquire. Given the fact that these resources are expensive and extra resources provide diminishing returns, it is important to make the best use of resources available. Operating rooms and surgeons are the most expensive and scarce resources in hospitals; so it is crucial to optimize their performance and avoid under and over utilized operating rooms. The aim of this study is to employ supervised machine learning techniques and probabilistic graphical models to predict the duration of surgical operations using historical data. We have used a wide spectrum of different models ranging from regression methods, classification methods, and Bayesian Networks to predict the surgical operation durations. The models built based on Bayesian Networks, in general, produce more accurate results with lower errors. Naive Bayes, however, outperforms the other Bayesian-Network based models with an average accuracy of 66.9% and root mean square error of 998 seconds (16.6 minutes) from the true duration of the operation. Provided with accurate estimation of surgical operation durations, it is possible to build optimization models to utilize healthcare facility resources. This allows healthcare facilities' managers to create tactical (medium term) plans and to increase efficient utilization of operating rooms and surgeons.

Keywords: Surgery Duration Prediction, Supervised Machine Learning, Probabilistic Graphical Models, Bayesian Networks

ÖZ

AMELİYAT SÜRELERİNİN GÜDÜMLÜ MAKİNE ÖĞRENME TEKNİKLERI İLE TAHMİNİ

Zabardast, Ehsan

Yüksek Lisans, Sağlık Bilişimi Bölümü

Tez Yöneticisi: Assist. Prof. Dr. Aybar Can Acar

Ortak Tez Yöneticisi: Assist. Prof. Dr. Marie Persson

Agustos 2017, 99 sayfa

Hastanelerdeki hasta sayısı giderek artmaktadır. Sağlık kuruluşlarının bununla başedebilmek için iki seçeneği bulunmaktadır: daha fazla cerrah ve kaynak edinmek veya var olan kaynağı daha verimli kullanmak. İstihdam edilebilecek cerrahi personel ve kaynaklarda reel bütçeden dolayı kısıtlar olması ve daha fazla kapasitenin getirisinin giderek azalan yapıda olması nedeniyle var olanı daha verimli kullanmak daha önemli hale gelmektedir. Ameliyathane ve cerrahlar hastanelerdeki en kıt kaynaklar oldukları için ise bunların zaman performansını optimize edecek zaman çizelgelerinin çıkartılabilmesi kritiktir. Bu çalışmanın amacı geçmiş veriden ameliyat sürelerini daha doğru tahmin edecek bir modelin güdümlü makine ögrenme teknikleri ile geliştirilmesidir. Geniş bir gamda regresyon ve sınıflandırma modeli denenmiş, bunlar arasında Bayes Ağı tabanlı modellerin genel olarak daha az hata gösterdikleri bulunmuştur. Bunun yanında Saf (Naive) Bayes modellerin ortalama %66.87 doğruluk ve gerçek ameliyat süresinden 998 saniye (16.6 dk.) karesel ortalama hata ile diğer Bayes modellerinden daha iyi sonuç verdiği görülmüştür. Ameliyat sürelerinin doğru tahmini ile sağlık kurumu kaynaklarının kullanımını optimize edebilecek modeller geliştirmek mümkün olacaktır. Bu da sağlık kurumu yöneticilerinin taktik (orta vade) planlamalarını ameliyathane ve cerrah kullanımının verimini artıracak şekilde yapabilmelerine imkan kılacaktır.

Anahtar Sözcükler: Ameliyat Süresi Tahmini, Güdümlü Makine Öğrenmesi, Olasılıksal Çizge Modelleri, Bayes Ağları

To my mother, father, and sister for all their love and support and putting me through the best education possible. I appreciate their sacrifices and I would not have been able to get to this stage without them.

And to Amin, my brother, for his undeniable support and help. I would not have gotten through this if it was not for him.

ACKNOWLEDGMENTS

My deepest gratitude goes first to my supervisor Assist. Prof. Dr. Aybar Can Acar and my co-supervisor Assist. Prof. Dr. Marie Persson, who expertly guided me through my graduate education and who shared the excitement for this project. Their unwavering enthusiasm kept me constantly engaged with my research, and their personal generosity helped make my time at Middle East Technical University and Blekinge Institute of Technology enjoyable and unforgettable. I was honored to work with them on this project.

I would like to thank Prof. Dr. Tolga Can, Assist. Prof. Dr. Rahime Belen Sağlam, and Assist. Prof. Dr. Nurcan Tunçbağ for taking time out from their busy schedules to attend the examining committee and provide insightful comments.

My appreciation also extends to my project colleague, Jan-Tobias Matysik. His encouragement and support has been especially valuable for me. I had a fruitful time working with him and I learned a lot while doing so. I am grateful to all of my friends, Mina, Rana, Nasim, Milad, Lisa, Farhad, Alireza, and everyone else, for all their moral and emotional support throughout my studies.

Above all, I am indebted to my family, whose value to me only grows with age. I would like to thank them for their unconditional support, encouragement, and infinite trust.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
ML	Machine Learning
OR	Operating Room
BN	Bayesian Network
NLP	Natural Language Processing
PGM	Probabilistic Graphical Models
DAG	Directed Acyclic Graph
CPT	Conditional Probability Table
EHR	Electronic Health Records
RMSE	Root Mean Squared Error
MAE	Mean Absolute Error
MAD	Mean Absolute Deviation
LR	Linear Regression
MARS	Multivariate Adaptive Regression Splines
RF	Random Forest
ANN	Artificial Neural Network
SVM	Support Vector Machine
kNN	k Nearest Neighbor
MMHC	Max-Min Hill-Climbing
FC	Feature Correlation
TS	Tabu Search
HC	Hill-Climbing
BMI	Body Mass Index
ASA	American Society of Anesthesiologists

CHAPTER 1

INTRODUCTION

Computers have become an inseparable part of our everyday life that it is hard to imagine living without them. They are very well integrated to the industry and are the reason for many technological advances. It is fair to say that without computers humanity could not have come this far. One of the main branches of Computer Science is Artificial Intelligence (AI). Artificial Intelligence is easily one of the most prevalent themes in all of science. The idea that a machine could exhibit the same level of intelligence and sentience as a human being has captivated scientists for decades. The primary aim of artificial intelligence is to create intelligent machines that can work and react like humans. Artificial intelligence is considered an essential part of the technology and industry.

Machine Learning (ML) is a subfield of artificial intelligence. It is the study of algorithms that learn from examples and experience instead of relying on hard-coded rules. Machine learning includes the methods for data analysis that automates analytical model building. Using algorithms that iteratively learn from data. Machine learning is a powerful tool that allows computers to find hidden insights and information without being explicitly programmed where to look.

In the broad sweep of AI's current worldly ambitions, machine learning healthcare applications seem to top the list in recent years. Machine learning has provided the healthcare industry with a wide variety of applications including diagnosis in medical imaging, drug discovery, robotic surgery, diagnosis decision support systems, personalized medicine and many more [1], [2].

This study aims to use machine learning and more specifically predictive analytics to predict surgical operation durations. Surgical operations are inherently and intrinsically stochastic and heteroscedastic [3]–[5]. This makes it very difficult to build a comprehensive model to predict the surgical operation durations. If provided with an acceptable prediction, it is possible to build optimization models in order to have better schedules in hospitals and healthcare facilities. This, in the end, will help to utilize resources and improve patient satisfaction; and it will, evidently, have enormous financial benefits.

1.1. Problem Statement

Many countries are struggling to reduce the healthcare costs. The healthcare costs are high in general, but the costs related to surgical operations are higher when compared to the other healthcare related expenditures [6], [7]. The reason why surgical operations are considered one of the most expensive areas in healthcare is because they are in dire need of many expensive and scarce resources such as specialized and professional staff, high-tech and sophisticated equipment, and other medical resources [8]–[10]. These resources need to be distributed among the different departments as well [11]. It is also important to mention that not all the healthcare facilities can perform specialized surgical operations because they either lack the equipment and facilities related to that specific surgical operation or they are not specialized in that specific area.

It is worth mentioning that the population of most of the developed countries are consist of elderly and the developing countries are going towards having an old population composition [12], [13]. The countries with old population will require more healthcare facilities to take care of their elderly in the near future. The mentioned problems and limitations necessitates the optimal use of available resources [14].

There are, in general, two ways to resolve this problem. The first approach would be to acquire more resources. This means building more facilities, training specialized staff, and buying the necessary equipment. This, in theory, is a good solution but it is not feasible in reality. Building new facilities requires substantial financial resources and more importantly the construction of these facilities is time consuming. The other problem of this approach is that the newly constructed facilities need specialized staff to run the facility and training new surgeons, doctors, and nurses takes time.

The second approach would be to utilize the resources and staff required in operating rooms (OR). This can be achieved by better scheduling the surgical operations and eventually increasing the throughput. In order to have a reliable schedule we need to have a good prediction of surgical operation durations. The problem is that surgical operations are very complicated and intrinsically hard to anticipate [3] – [5], and the duration of each specific surgery can vary a lot depending on the patient, patient's condition, type of surgery, facility, operating staff, and other factors [3] – [5], [15], [16].

There are many aspects that affect the performance of operating rooms. These include: costs, patient waiting time, operating room utilization, patient throughput, surgical operation cancellation, surgical operation delay, and many more [17]. These aspects need to be considered when managing and scheduling the hospital resources. Each hospital has a couple of operating rooms and the managers need to schedule surgeries based on demand and the available resources they have. If the managers have reliable estimations for the planned surgical operations, they can have a better and more precise schedule for the hospital or the healthcare facility. The previous studies have shown that it is possible to

come up with a model to predict the duration of surgical operations [3]–[7], [11], [15], [17].

1.2. Objective

The main objective of this study is to use predictive analytic methods including machine learning, statistical, and probabilistic modeling to predict the duration of surgical operations. This study uses historical data to build different models, hence all of the models can be categorized as supervised learning methods. The available data mainly consist of three clusters of features including the features related to the patient, the features related to the healthcare facility, and the features related to the illness. With this aim in mind, the forthcoming objective is to build an optimal scheduling model based on these predictions. It is important to remember that for building a suitable and successful scheduling model, the predictions should be as precise as possible and their error should be minimum. Figure 1 illustrates the simplified schematic representation of the methodology of this work.

1.3. Scope

This study utilizes predictive analytic techniques including machine learning, statistical, and probabilistic methods to predict the duration of surgical operations. The surgical operation durations are very diverse and stochastic since there are a lot of factors that affect them. This means that in order to build a comprehensive model, the data set should be as comprehensive as possible.

We tried to build comprehensive models as extensively as possible. This was because we were not sure which model would have better results. A wide spectrum of models has been tested during this study and they range from regression models, conventional classification models, and finally probabilistic graphical models. We started with regression models then moved to classification models and finally concentrated on Bayesian-Network (BN) based classification models. After trying out different models, it was concluded that classification models work best it this area.

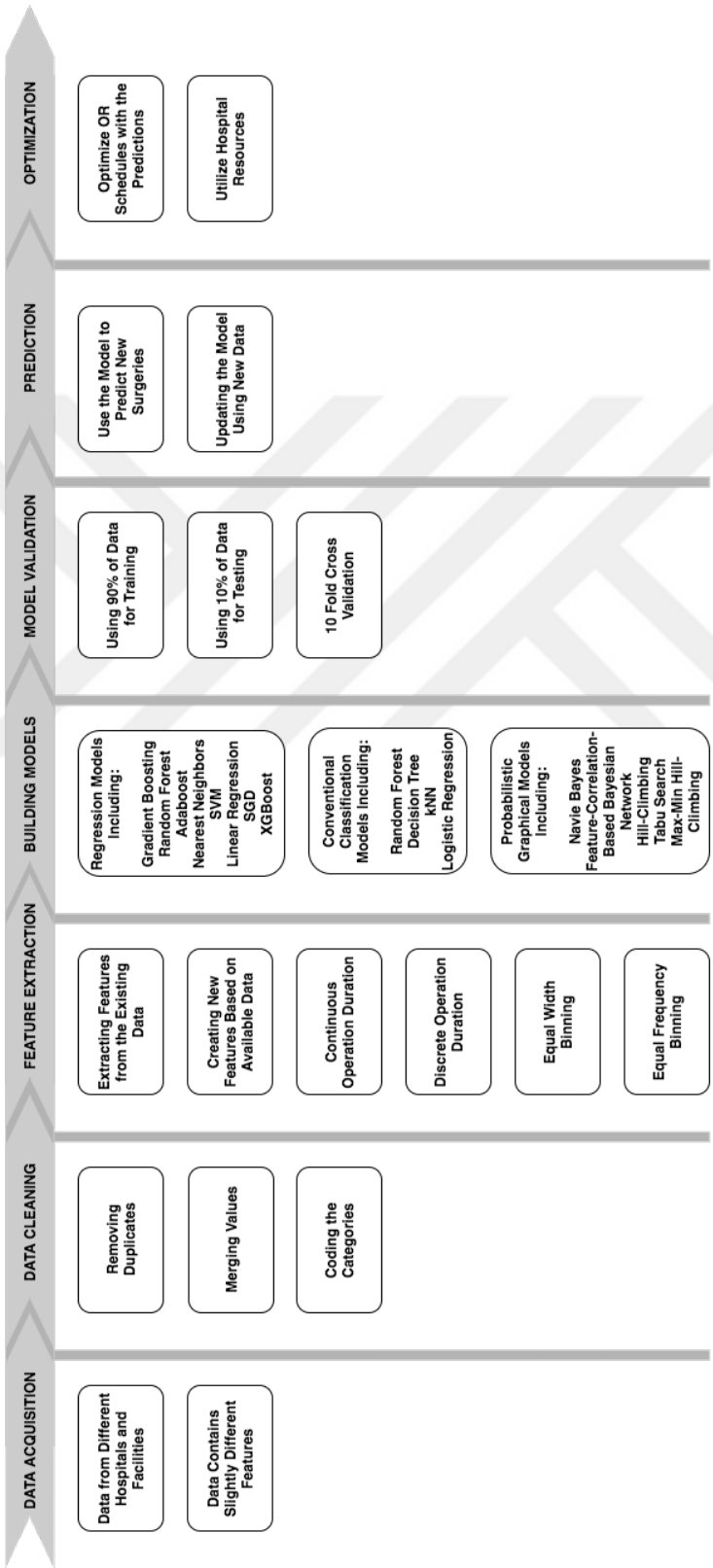

Figure 1: Schematic Representation of the Methodology of This Study

1.4. Thesis Organization and Outline

The outline of this thesis is as follows. There are a total number of seven chapters including (1) Introduction, (2) Background, (3) Data Preparation, (4) Methodology and Results, and (5) Conclusion.

Chapter two is dedicated to the explanation of the technologies and methods which are used to build and prepare different models. It demonstrates the main ideas behind methods and approaches used in this study. It also includes the literature review and related works which have been conducted so far in this spectrum. Chapter three provides a detailed description of data preparation process. This chapter also contains feature selection and feature extraction which are essential prerequisites prior to building the models. This chapter delves deeper to explain how feature selection and feature extraction works and how the fitting features should be selected that can represent the model properly. Chapter four introduces the methodologies used in this study and illustrates how different algorithms work and what can be expected from them. This chapter also includes all the models that have been built in this study and their respective results. The major methods can be categorized in three different sections namely, regression models, conventional classification models, and Bayesian-Network based classification models. And finally chapter five is an overview of the results of different models and their collation. It includes the summary of the thesis and its major findings. This chapter is the conclusion of this work and it provides the provisions for the future studies.

CHAPTER 2

BACKGROUND

This chapter is dedicated to the background information necessary for this study. Here we cover and explain the technology which is used during this study. First, we will cover the basics of artificial intelligence and predictive analytics. Later, we will go over their applications in healthcare informatics. This chapter will end with the demonstration of how these techniques can be used in order to predict the operation durations.

2.1. Artificial Intelligence

The field of artificial intelligence was officially created by John McCarthy in 1956 in a workshop. The goal of the workshop was to investigate the ways in which machines could be made to simulate aspects of any intelligence. It was McCarthy who coined the word "artificial intelligence" with his co-authors in their proposal [18]. One of the most influential people with inspiring ideas who proposed the formal model of computing in his classic essay *Computing Machinery and Intelligence*, was Alan Turing [19]. His famous ideas and theories included how intelligence might be tested (the Turing test) and how machines might automatically learn.

Artificial intelligence is the science of making intelligent machines or more specifically intelligent computers. It also tries to find similarities between computers' and humans' intelligence, but artificial intelligence does not restrict itself with methods which are biologically observable. Intelligence is the ability to acquire and apply the gained knowledge. The problem is that we cannot yet characterize in general what kind of computational procedures we want to call intelligent. We understand some of the mechanisms of intelligence and not others. Another problem is that we cannot ask a yes or no question "Is this machine intelligent or not?" Intelligence involves mechanisms, and artificial intelligence research has discovered how to make machines carry out some of them and not others. If doing a task requires only mechanisms that are very well understood today, computer programs can give very impressive performances on these tasks. It is also important to remember that artificial intelligence is not trying to simulate human intelligence in all cases.

Artificial intelligence branches to many areas including logical AI, search, pattern recognition, representation, inference, common sense knowledge and reasoning, learning

from experience, planning, epistemology, ontology, heuristics, genetic programming, and more. The applications of AI include game playing, speech recognition, Natural Language Processing (NLP), computer vision, expert systems, heuristic classification, etc.

2.2. Machine Learning and Predictive Analytics

Machine learning is part of artificial intelligence that enables computers with the abilities to learn without being explicitly programmed or hard-coded beforehand. Machine learning focuses on the development of computer programs that are able to adapt themselves when they are exposed to new data, hence making them very powerful when tailoring and conforming their patterns. Machine learning is very similar to data mining. They go through data to find patterns hidden in data. What makes machine learning different from data mining is that in data mining the extracted data is used for human comprehension whereas machine learning algorithms use the data to find patterns and adjust the program actions accordingly.

Machine learning can be categorized into two major parts, supervised machine learning and unsupervised machine learning. In supervised machine learning the data is already labeled with "input" and "target" labels. The model will be trained based on these labeled data. On the other hand, unsupervised machine learning deals with the unlabeled data and has to find relations and associations itself. Supervised learning algorithms can be further subcategorized into regression and classification. Regression models are generally used for continuous target variables whereas classification models try to work with discrete target variables. Unsupervised learning algorithms can be further subcategorized into clustering and association. Clustering models try to discover the inherent groupings in data whereas association models are used when we want to discover rules that describe the data better.

Predictive analytics can be defined as the use of data, machine learning, and statistical and probabilistic models and techniques to identify the likelihood of future events based on historical data. Larose & Larose define predictive analytics as "... the process of extracting information from large datasets in order to make predictions and estimates about future outcomes." [20] Predictive analytics, in short, is understanding the future. Predictive analytics is part of the supervised learning.

2.2.1. Procedure

We need to have a concrete procedure when it comes to machine learning and data mining projects. According to CRISP-DM, a project will go through six different phases. These phases are, *Business/Research Understanding Phase, Data Understanding Phase, Data Preparation Phase, Modeling Phase, Evaluation Phase,* and *Deployment Phase* [21]. These phases and their relationships are illustrated in Figure 2 The process is not

sequential, it is adaptive, meaning that the next step in the sequence will be determined by the results and outcomes of the previous phase.

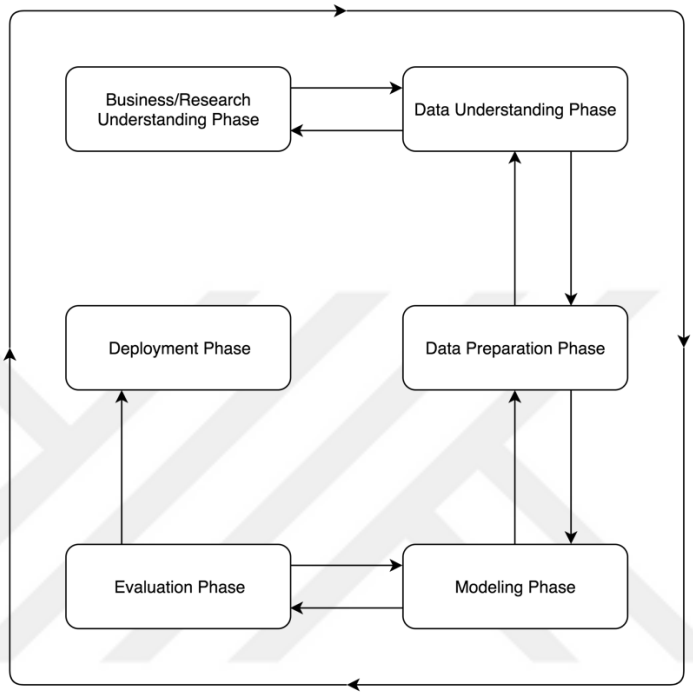

Figure 2: CRISP-DM Phases and Procedure

Business/Research Understanding Phase: The first step is to understand the research project objectives clearly as a whole and translate these objectives into restrictions and formulations of a machine learning problem definition. Finally, having a strategy for achieving the goals and objectives of the research project.

Data Understanding Phase: This phase begins with the data collection, that is if the data hasn't already been collected. The data, then, should be explored to gain new information and insights related to the research subject. The quality of the data should also be evaluated. The last step is to sample the data and get the subsets to test the models with.

Data Preparation Phase: This is one of the laborious phases of machine learning procedures. It includes all the aspects of preparing the final data set which is going to be used in the subsequent phases. The cases and variables that are going to be analyzed will be selected. These cases should be appropriate for the analysis phase. The transformation and conversion of the variables will happen in this phase, if needed. The output of this phase is the data ready for the modeling tools and algorithms.

Modeling Phase: This phase begins with the model selection. The selected model should be calibrated in order to optimize the results. Several models can be built and tested during this phase. This phase may need a loop back to data preparation phase to redefine and

reevaluate some of the features with the specific requirements of a particular machine learning algorithm.

Evaluation Phase: The modeling phase will deliver one or several models. The performances of these models need to be evaluated to identify the best working model before the deployment phase. The results of the selected model should also be compared with the objectives of the first phase to determine if it meets the expectations or not. Finally, the use of the proposed model should be decided.

Deployment Phase: Having the model does not signify the end of the project. In order to complete the project, the model needs to be deployed in a working setting. It is suggested that the model will be deployed in a controlled setting to evaluate the results in action [20]. Once the results are confirmed, the selected model will be fully deployed.

2.3. Probabilistic Graphical Models and Bayesian Networks

There are many ways to build models based on the available data, one of which is probabilistic graphical models. A Probabilistic Graphical Model (PGM) is the mathematical way of capturing the relationships, like conditional dependencies, between the variables. The nodes in the graph represent the variables and the edges represent the dependencies. PGMs are divided into many different categories, but in this study we are focusing on Bayesian Networks.

Bayesian Networks (BN) are among the most frequently used and popular models when it comes to reasoning under uncertainty [22], [23]. A Bayesian network is a Directed Acyclic Graph (DAG). In such graph, directed arcs connect the variable nodes. The conditional probability tables (CPT) can be calculated based on these relationships. Together the CPTs represent the full joint distribution. [24]

After the network is created, the values of the combinations of the nodes will be fed to the network to produce the posterior probability distribution for each node in the network. There are many algorithms both for exact inference and approximate inference of these probabilistic updating, providing a powerful combination of predictive, diagnostic, and explanatory reasoning [24].

2.4. Machine Learning Applications in Healthcare

Automated data collection routines have been subject to a rapid and brisk evolution in the recent years and have led to a huge amount of patient-related data stored in distributed, heterogeneous, and large databases [1]. We live in an era where data is abundant and this data needs to be put to good use. Furthermore, according to Dua et al., with the easy access

to high speed networks coupled with the advances in knowledge discovery boosted by mobile technologies has "amplified the emphatic demand for a unifying, coherent computing resources designed to accommodate, enhance, and empower multidisciplinary, and multi-institutional healthcare informatics research". [1]

Healthcare related data is usually unstructured, complex, context-independent, highly dimensional, and inherently heterogeneous [1], [25]. Working with such data requires a lot of effort because of its nature and it is challenging to interpret and extract insightful knowledge and information from it. Healthcare data related resources are comprised of a wide variety of data types as well, ranging from free text notes to complex image types. Having such prolific and diverse data resources present a good opportunity for implementation of classical and novel machine learning algorithms that are "in the heart of the current data-rich but information-poor diagram." [26] Having a consolidated, expanded, and immerse view of the data will accommodate for the opportunity to tackle previously impossible clinical discoveries.

2.4.1. Healthcare Informatics

Healthcare informatics is a big part of healthcare system now and without a doubt it is one of the reasons for such high quality services available these days. Healthcare informatics has the responsibility to acquire, transmit, process, store, and retrieve the information relevant and pertinent to healthcare [27]. This information can be used for early detection, diagnosis, and treatment of disease. Healthcare informatics is enclosed with electronic health records (EHR), the data related to diseases, and the technologies associated with handling of such data.

With the investment in superior and improved technologies to support researchers, medical practitioners, and patients, healthcare informatics can yield in affordable and high quality healthcare system. With this aim in mind, it is crucial to invest and start using computational tools and technologies for referral and prescription aids, the management of electronic health records and other areas.

"Machine learning is a natural extension of artificial intelligence." [1] Machine learning is often used for intricate and elaborated statistical analytics. A big part of health informatics, evidently, is the combination of health data and machine learning. Healthcare informatics aims to detect and distinguish the hidden patterns in data and evolve with it [28].

2.5. Predicting Surgical Operation Durations

Every manager in healthcare facilities tries to run the operating rooms effectively and efficiently. A successful schedule can be measured by the overall utilization of the

operating rooms. It is crucial to remember that utilization is closely linked to cost efficiency [29]. Staff satisfaction is another important factor when it comes to operating rooms' effectiveness. "Satisfaction is greatly influenced by hours of overtime, OR utilization that extends beyond the scheduled hours of surgery." [30] It would be ideal to have no delays or dead times during the scheduled hours in an OR, but because of the nature of surgical operations the variability is inevitable.

Surgeries are stochastic in their nature and they have an intrinsic variability. The scheduling of such activities are harder and more challenging than the tasks that can clearly be defined in mechanical steps studied in traditional scheduling applications. It is globally unanimous and agreed upon that "one of the most essential factors in reducing the element of variability in the schedules is to improve the estimate of surgical operation durations." [31]

As mentioned before in numerous articles and studies, operating rooms are considered to be the most crucial of hospital resources. They need to be managed efficiently and effectively to deliver a high quality of care and a high throughput with a sensible and plausible cost [32]. The reason ORs need to be cost-efficient is that the average cost for operating room time per hour is extremely high (roughly 4000$ per hour in US) [33], [34]. While the cost of ORs are intrinsically high due to their complicated nature, mismanaged ORs with cancelled, delayed, or prolonged surgeries can have degrading effect on the healthcare facilities' throughput [30], [35], [36].

Attempts to predict the surgical operation durations range from very simple guesses made by surgeons and OR teams to very complicated mathematical models. These models try to anticipate the duration by considering different factors. The simpler models consider basic factors such as type of surgical operation, identity of the surgeon, date of the surgery, etc. to predict the surgical operation durations. Different studies have shown that by taking advantage of richer sets of information and creating detailed models, it is possible to increase the accuracy of predictions. All of the effort for predicting surgical operation durations is for but one reason, to increase the quality of operating room schedules. The more accurate the estimation is, the more reliable the scheduling will be.

2.6. Literature Review and Related Works

The notion of predicting surgical operation durations is not a novel idea. Initially, hospitals and healthcare facilities relied on the predictions made by surgeons and OR teams. Given the fact that surgical operations are fundamentally random and unpredictable, even the experienced and professional surgeons are unable to provide accurate prognostications. There have been many studies trying to find the optimal model for predicting the surgical operation durations. They have used various models trying to tackle this problem with different approaches.

As discussed before the main reason for attempting to predict the surgical operation durations is to best utilize the operating rooms. The goal is to minimize overutilization and underutilization of the operating rooms. According to Fügener et al., when it comes to predicting surgical operation durations by surgeons, "they consistently plan too long (too short) in cases where the optimal duration is below (above) the average duration of a surgery." [37] The same study hypothesize that the surgeons tend to overestimate the duration and avoid overutilization rather than underutilization [37]. This makes the predictions made by the surgeons on average longer than what they really are.

In the past decades there have been many studies on managing the operating rooms [38]. Magerlein and Martin had reviewed the literature on surgical demand scheduling and distinguish between advanced scheduling and allocation scheduling [39]. Blake and Carter added the domain of external resources to previously mentioned taxonomy [40]. Przasnyski categorizes the research studies based on general areas of concern such as cost containment and scheduling of specific resources [41]. There are many other reviews which cover the operating room management as part of global healthcare service [42] – [44].

In a recent study Ng et al. have tried to predict the surgical operation durations using neural heteroscedastic regression. Their work is based on a collection of surgeries recorded at the University of California, San Diego Health System. For each surgery they have considered two sets of features, patient attributes such as age, gender, weight, etc. and attributes of the clinical environment such as the surgeon, location, and time. Their working data included 86,796 surgery instances. Their best model had been able to predict the surgical operation durations with RMSE of 44.28 minutes and MAE of 23.53 minutes. [3]

In another study Kayis et al. combined five different statistical methods and a hybrid of them to improve the surgical operation duration estimations. In their study, they have "constructed a statistical model with relevant operational, temporal, and staff-related factors to adjust the estimates generated by SchDur method to achieve higher accuracy." [16] Their hybrid method helped to reduce mean absolute deviation (MAD) from 43.00 minutes to 41.01 minutes in the training data. [16]

In a similar study, ShahabiKargar et al. have tried to predict the surgical operation durations using Linear Regression (LR), Multivariate Adaptive Regression Splines (MARS), and Random Forest (RF) methods. They have used similar features to predict the surgical operation durations based on three main categories of features including patient characteristics, surgical operation characteristics, and surgery team characteristics. The data they have used contained over 60,000 surgical operation instances scattered in four years. They managed to drop the RMSE of the currently used model in hospital (27.88 minutes of RMSE) by approximately five minutes using the Random Forest method (22.78 minutes of RMSE). [4]

Burgette et al. had tried to estimate the surgical operation durations with linear median regression models using anesthesia billing data and operating room records [45]. Devi et al. have used Artificial Neural Networks (ANN), Regression Analysis, and Adaptive Neuro Fuzzy Inference System to predict the surgical operation durations [4]. Dexter and Ledolter used Bayesian prediction bounds to estimate the surgical operation durations [46]. Kayis et al. used operational and temporal factors to predict the surgical operation durations [15].

In conclusion, there have been many studies and research trying to find the optimal model to predict the surgical operation durations. To best of our knowledge, there have not been any attempts to predict the surgical operation durations using Bayesian Network Classification methods.

CHAPTER 3

DATA PREPARATION

This chapter is dedicated to the data preparation process. All the learning algorithms rely on data. It is critical to have good data and to feed your algorithm with the proper data relevant to the problem you want to solve and this is not restricted to these conditions. You need to have good data and it needs to be in the right scale and format. It should also consist of appropriate features. The following chapter will demonstrate the effort and process that went into preparing the data.

3.1. Data Cleaning

Preparing data which includes cleaning, merging, and imputation is a critical step in any data related research project. Before starting to work on the data for a machine learning project, it is extremely important to comprehend the nature of the available data and what you want to achieve. Without understanding the data, there will be no basis from which we can make the decisions about what data is relevant as we clean and prepare the data.

To put it in simple words, data cleaning can be defined as the process of standardising and formatting the data to make it ready for the analysis phase. Data cleaning is, often, a laborious task and takes a large portion of time spent on the project. The collected data will contain discrepancies unless it has been purposefully collected for the machine learning or data mining project at hand and even in this case the data needs to be cleaned because of missing data, errors, and invalid inputs. The cleaning phase is a crucial step in any given data related research project. The result and the accuracy of these results depend profoundly on the data and the features used in the research project.

As mentioned before the data needs to be cleaned for a couple of reasons. The data can be collected and captured from different and various resources. These resources might not have the same format even if they include the same information. The data from different resources needs to be cleaned and merged so it will represent the information in the correct and usable format for the machine learning algorithm. The data might also contain special characters, stop words, HTML tags, etc. these anomalies need to be removed as well.

3.1.1. Missing Data and Data Imputation

One of the prominent problems of data analytics in most scientific research domains is missing data and how to handle this problem [47]–[50]. Missing data occurs for a couple of reasons such as mishandling of samples, low signal-to-noise ratio, measurement errors, non-response, or deleted aberrant value [47]. Another important part of data cleaning is data imputation. There is a certain ambiguity with the datasets with missing data entries and this will affect properties of statistical estimators. Large datasets will contain missing entries no matter the data source. These missing values need to be handled properly because some of the learning algorithms might not be able to handle missing data. There are a variety of methods for substituting missing values and they are generally referred to as "missing data imputation" [51]–[53]. Data imputation, then, according to OECD Glossary of Statistical Terms can be defined as "the substitution of estimated values for missing or inconsistent data items (fields). The substituted values are intended to create a data record that does not fail edits." [54]

3.1.2. The Available Data for This Study

The raw data contains 377,685 entries. It contains three categories of information related to each surgical operation including the starting and finishing times of the tasks related to each surgical operation, the information regarding the patient, illness, and diagnosis, and the data related to healthcare facility and department for which the surgical operation is carried out. The detailed information on the data is available in Table 1.

Table 1: Features of the Original Dataset Available for This Study

377,685 entries divided into three main categories		
Timing information related to each surgical operation	Patient and disease information	Facility and department information
Called to department	Age	Operating room number
Patient start time	Gender	Clinic
Anesthesia start time	ASA physical status classification	Personnel ID
Blocked start time	Operation card method	Personnel Category
Anesthesia ready	Diagnosis code	
Ready for operation	Operation code	
Operation start	Anesthesia method	
Operation end	Treatment count	
Anesthesia end		
Operation date		

The original data set contained missing and duplicated values throughout the whole spreadsheet. It also contained special character, descriptive texts, and codes for some of the features. The initial approach for removing the duplicates included merging the data as well. Even though the data contained duplicate entries for each surgical operation, some of the values of the features were not duplicated meaning that for some features the values were the same but for the others they were different and contained unique information.

This is the reason why we could not just remove the duplicate entries and had to go through every instance of the data to find the differences and merge them into single feature value.

3.2. Feature Selection and Extraction

Feature selection and extraction is the next step after cleaning the data. Feature selection is simply selecting the features that are already in the correct format for the model and they do not need to be changed. These features can be directly used in the model as long as they provide significant benefits and advantages to the results. Feature extraction, however, is more complicated. It is the construction of new features from the raw data. These features will be created manually after considering the underlying information and form of the problem, structures in the data, and how to best expose them to a learning algorithm.

The features which will be used in the learning algorithm directly and immensely influence the results of the predictive models. The better the available features, the better the results that will be achieved. The previous statement can be misleading; good features will not guarantee better results. The achieved results are generally influenced by three different factors including the chosen learning algorithm and model, the nature of available data, and lastly the features which are prepared.

The main goal of feature selection and extraction is to find features that describe the structures inherent in the data. Better features, in the end, will help to have flexibility in choosing and modifying the model, a simpler model, and lastly better results.

3.2.1. Features Extracted From the Data

We have calculated the surgical operation duration for each surgery by subtracting the ending time from starting time of the surgical operation. The time is calculated in seconds but as we will discuss in section 3.4 in this chapter, we have used the time as a discrete target value for classification algorithms and discretized it into time bins. The process and methods for discretizing the time will be covered in details in section 3.4 in this chapter. We have also calculated the starting time of the surgical operation. The initial approach was to have each hour as a separate category, but later we decided to further categorize the starting time to four distinct bins as follows: from 0 to 7 as category 1, from 7 to 12 as category 2, from 12 to 18 as category 3, and from 18 to 24 as category 4.

We have also categorized the patients' ages into four distinct categories including 0 to 20 as category 1, 20 to 40 as category 2, 40 to 60 as category 3, and older than 60 as category 4. We have calculated the number of treatments that had been done on the patient. Even though a significant amount of the surgical operations had only one treatment, there were cases with 2 or 3 treatments in one surgical operation as well. We have calculated the day

in which the surgical operation was carried out as the *operation weekday* feature and created a boolean feature specifically to indicate if the surgical operation had been carried out in the weekend or not.

In a significant portion of the original data, each surgical operation had more than one anesthesia method used in the surgical operation. We decided to merge the anesthesia methods used in each case and coded them into one single feature. The number of anesthesia methods used in each surgical operation is also calculated and used as a separate feature. This allowed us to categorize and utilize the anesthesia methods as a strong feature more coherently. The operation code in the data is already coded, but like the anesthesia methods a considerable amount of surgical operations had more than one operation code. Following the previous attempt to merge the anesthesia methods, we treated the operation code in the same way. We have merged all the operation codes included in each case into one single feature and further calculated the number of operations carried out in each surgical operation.

The original data contained information about the staff present in the surgical operations, this information include the personnel ID and categories. We extracted eight distinct features out of these two variables. Initially we extracted all of the available categories of staff and selected seven influential categories including *general surgeon, anesthesiologist, orthopedist, urologist, gynecologist, radiologist,* and *medical student*. These features are all boolean and they represent the presence of each category in the surgical operation e.g. if the feature "anesthesiologist" is *1* it means that at least one anesthesiologist was present in the surgical operation and if it is *0* it means that there was no anesthesiologist present in the surgical operation. We have also extracted the total number of staff that were present during the surgical operation as a separate feature.

There were a total number of eighteen extracted features from the original data. The rest of the features were used in their original form. Table 2 summarizes the extracted features from the data.

Table 2: Summary of Extracted Features From the Original Data

	Original Data		Cleaned Data	
Feature	**Type***	**Description**	**Type***	**Description**
OP Duration	TS	Start and end of the operation are stored separately in different variables	C	Operation Duration extracted as seconds by subtracting the ending time from the starting time of each surgery
Patient Start	TS	Stored values contain date and time of the surgery	D	Four distinct categories for the start time of surgery
OP Day	TS	Stored values contain date and time of the surgery	D	Seven categories to indicate days of week
Weekend	TS	Stored values contain date and time of surgery	B	Boolean values indicating the weekend
Anesthesia Method	T	Names of anesthesia methods stored as text	D	Anesthesia methods coded into discrete values
Anesthesia Count	T	Names of anesthesia methods stored as text	D	Number of anesthesia methods used in each surgery as categories
OP Code	T	Codes of operations stored as text	D	Operation codes merged together into discrete values
OP Count	T	Codes of operations stored as text	D	Number of operations carried out in each surgeries as categories
General Surgeon	T	Categories stored as text	B	Presence of general surgeon in the surgery as a boolean
Anesthesiologist	T	Categories stored as text	B	Presence of anesthesiologist in the surgery as a boolean
Orthopedist	T	Categories stored as text	B	Presence of orthopedist in the surgery as a boolean
Urologist	T	Categories stored as text	B	Presence of Urologist in the surgery as a boolean
Gynecologist	T	Categories stored as text	B	Presence of gynecologist in the surgery as a boolean
Radiologist	T	Categories stored as text	B	Presence of radiologist in the surgery as a boolean
Medical Student	T	Categories stored as text	B	Presence of medical student in the surgery as a boolean
Staff Count	T	Personnel IDs stored for each surgery	D	Number of staff present in surgeries as categories

* T: Text - TS: Timestamp - C: Continuous - D: Discrete - B: Boolean

The final dataset included 24,723 unique entries after cleaning, merging, and imputing the data. We did not to use the the data entries which had missing values throughout the spreadsheet.

3.3. Data Distribution

After cleaning the data and extracting the features, it is good to take a look at data distribution to find patterns and gain extra information. This information can later help developing more suitable and more comprehensive models. Understanding the underlying data distribution in a dataset before applying any machine learning algorithm or statistical modelling approach is crucially important. Many of the machine learning projects fail because of not considering the data distribution before building the models. Learning algorithms have, either explicitly or implicitly, a certain number of assumptions on the data they are going to use. It is important that the data abide to these assumptions. That being said, the data should satisfy these assumptions or it needs to be converted to concede these assumptions. The rest of this section provides the distribution for the important features in the available data.

Predicting the surgical operation durations is the main goal of this study. What we are trying to predict is time and it is a continuous variable in its nature. However, we base the units of time as seconds and treat it as a continuous variable. Figure 3 demonstrates the distribution of *surgical operation duration* of available cases in the data. The horizontal axis indicates the duration of surgeries and the vertical axis indicates the number of surgeries. The distribution is skewed to right. It can be inferred from the histogram that most of the surgical operations take less than 8,000 seconds.

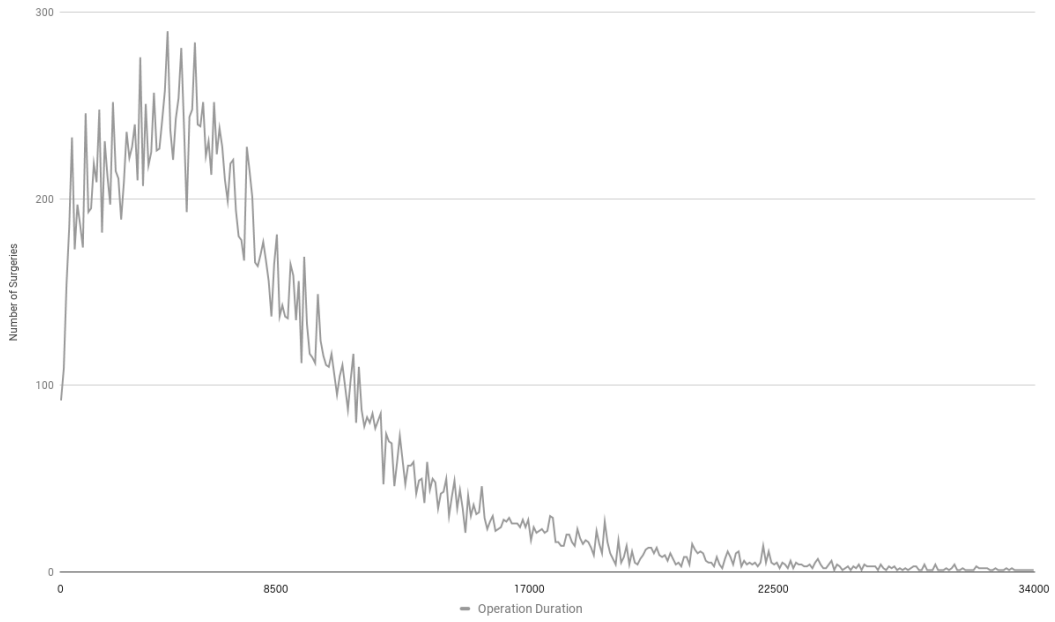

Figure 3: The Histogram of Surgical Operation Duration

Figure 4 shows the distribution of *age* categories in the data. [0, 20) as category 0, [20, 40) as category 1, [40, 60) as category 2, and finally [60, inf) as category 3.

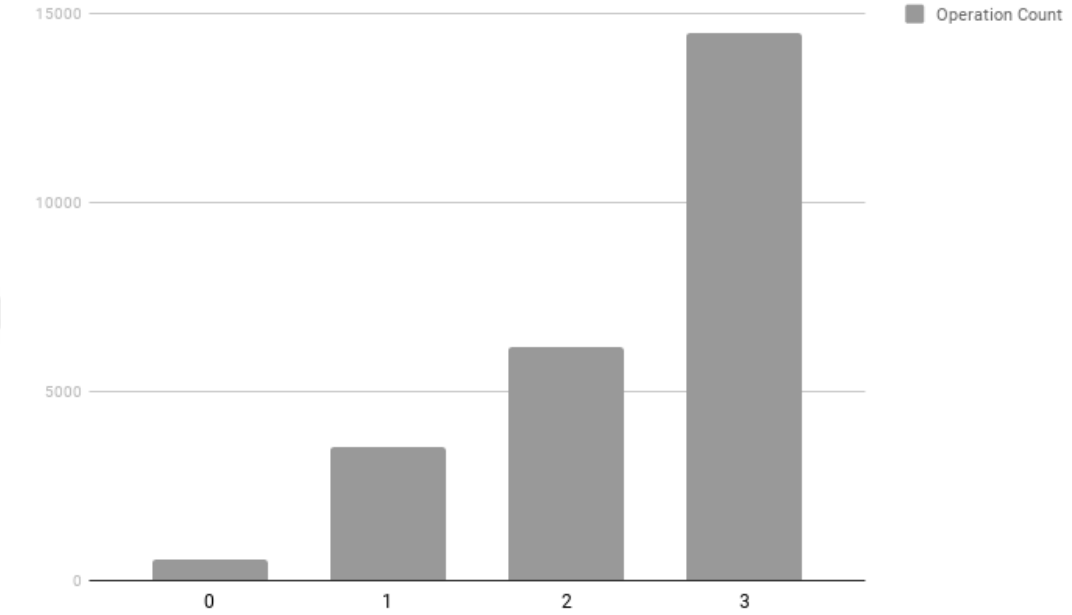

Figure 4: The Distribution of Age Categories

Figure 5 shows the distribution of the *clinics* categories.

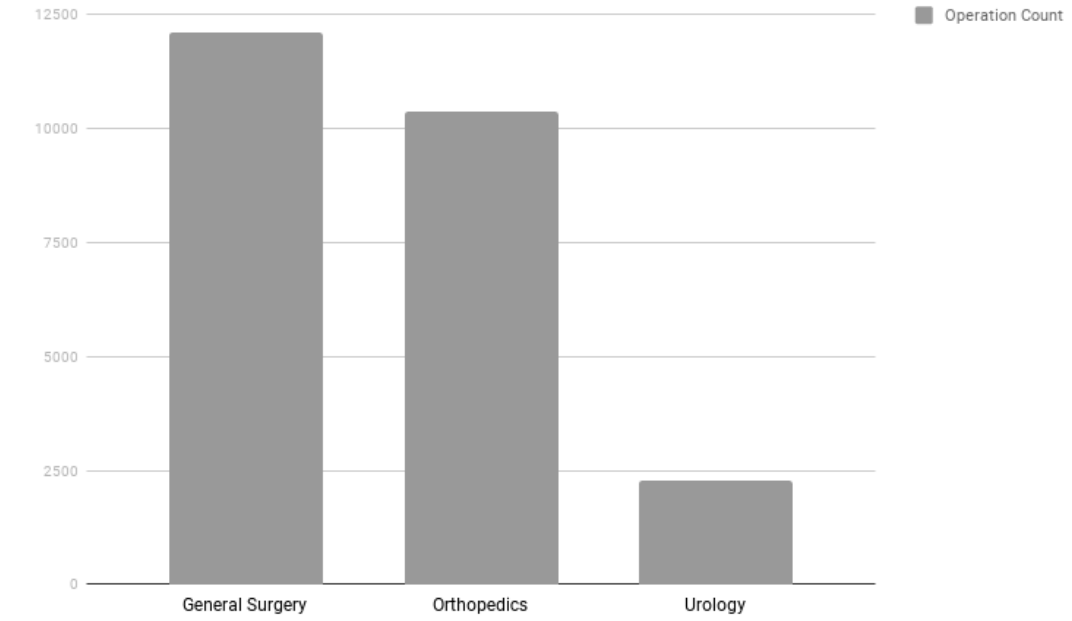

Figure 5: The Distribution of Clinics Categories

Figure 6 shows the distribution of *days of operations*.

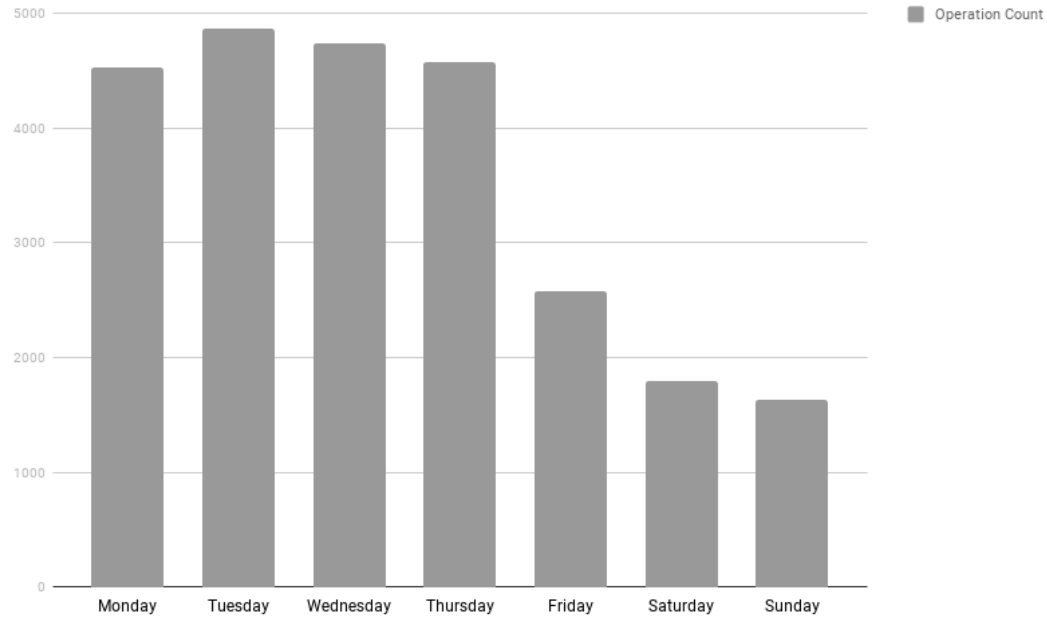

Figure 6: The Distribution of Days of Operation

Figure 7 shows the distribution of *patient start time* in data. [0, 7) as category 0, [7, 12) as category 1, [12, 18) as category 2, and finally [18, 24) as category 3.

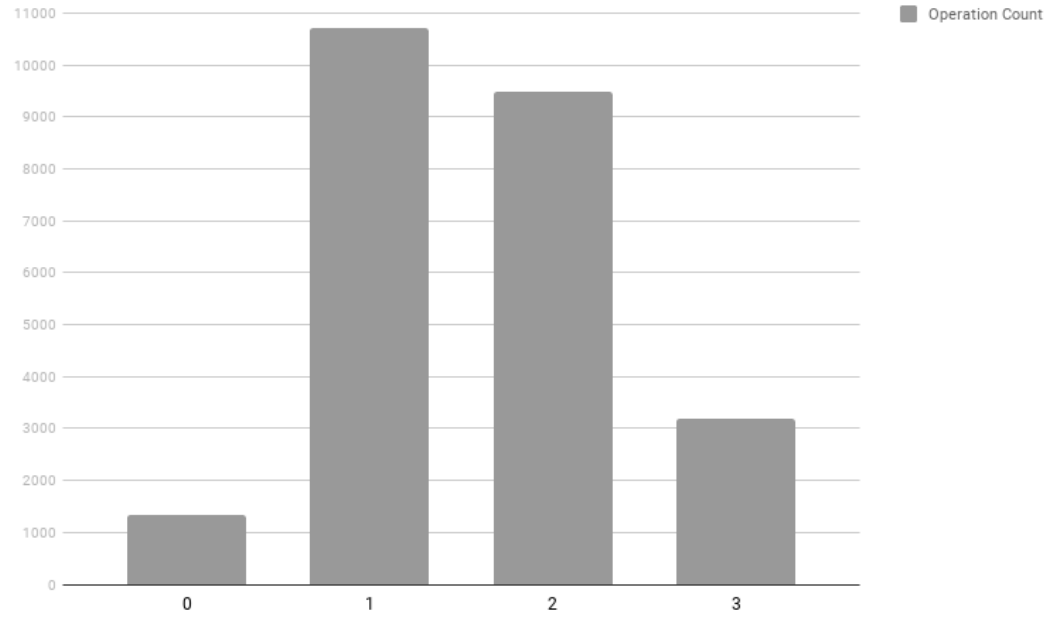

Figure 7: The Distribution of Patient Start Time

Figure 8 shows the distribution of *ASA physical status classification* categories.

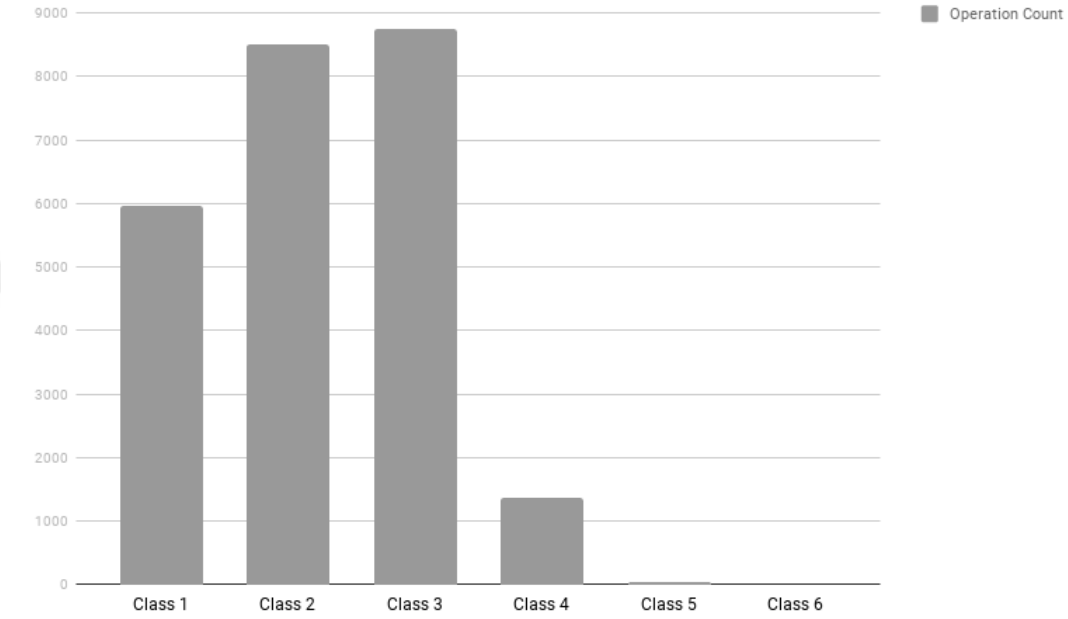

Figure 8: The Distribution of ASA Physical Status Classification

Figures 9, 10, and 11 show the distributions of *anesthesia count, operation count,* and *staff count* respectively.

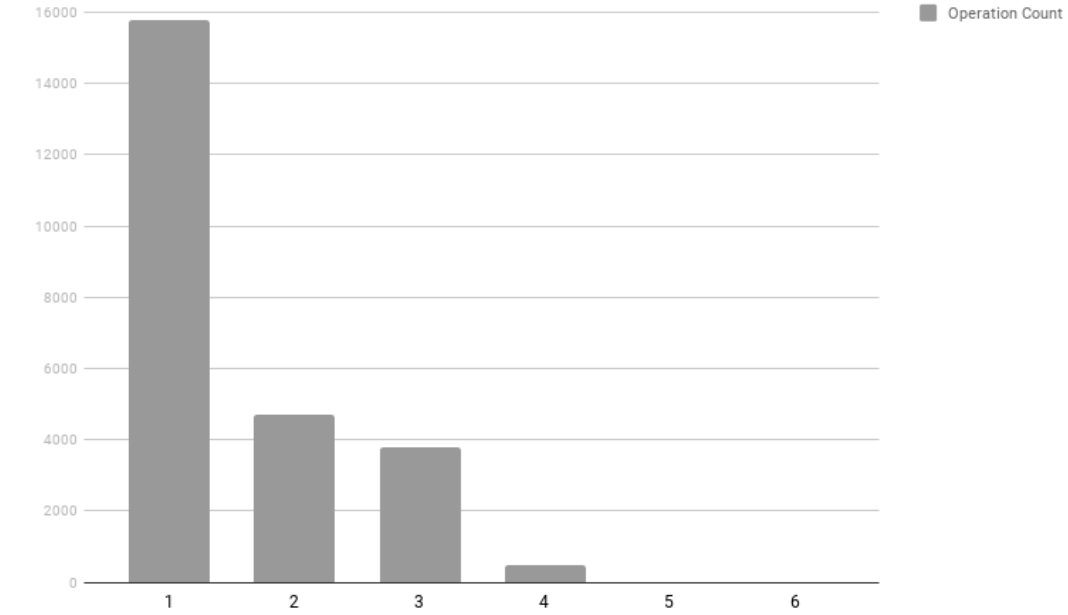

Figure 9: The Histogram of Anesthesia Count

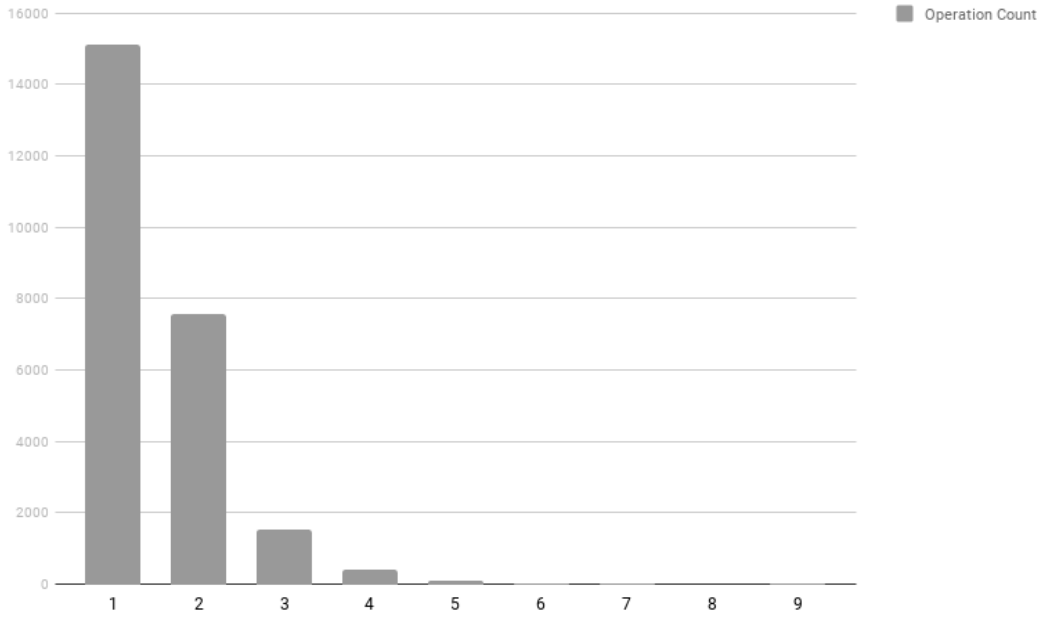

Figure 10: The Histogram of Operation Count

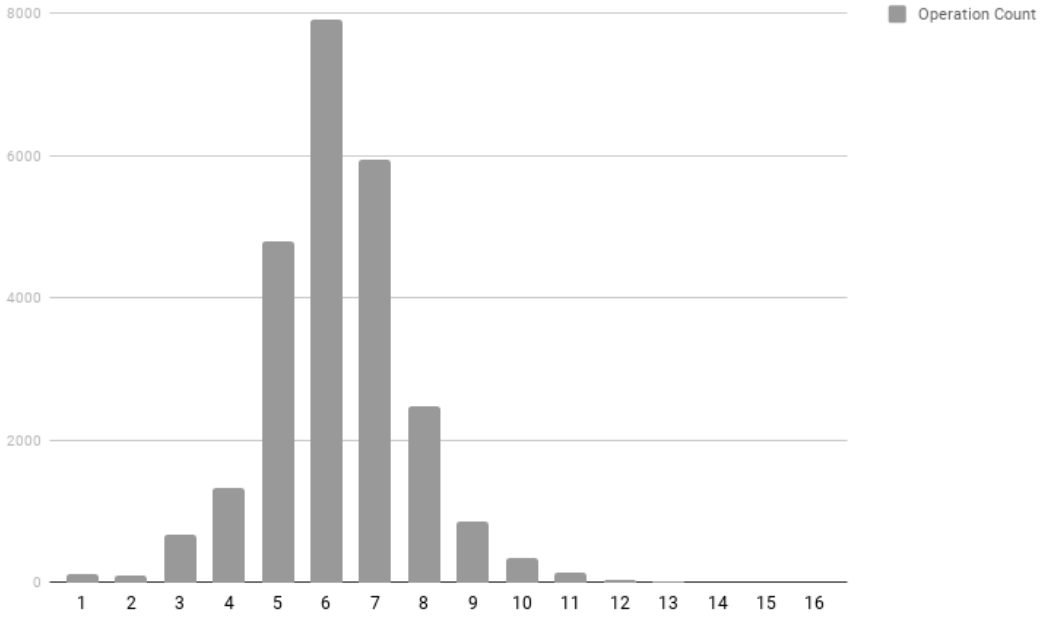

Figure 11: The Histogram of Staff Count

3.4. Discretizing Surgical Operation Durations

In data sciences discretization is referred to as the conversion of continuous values of features into nominal or discrete values. Data discretization is one of the prevalent methods in the data preparation phase [55]. Data discretization is a reduction mechanism because it diminishes the wide spectrum of continuous values into subsets of values and categories. Many machine learning algorithms need discrete values as inputs to be able to be implemented [56]. Data discretization is not restricted to only continuous variables. Nominal variables can also be discretized to reduce the original categories into comprehensive and robust categories. Machine learning algorithms which use discrete data can have remarkable improvements in learning speed and accuracy when they are provided with better categorization of the data [56], [57]. It is important to mention that discretizing the data generally leads to a loss of information and such information loss should be minimized.

There are four general steps when it comes to discretizing the data. These steps can be changed, skipped, or modified depending on the data. The first step is to sort the values of the selected feature. The second step is to find cut points for splitting the data or adjacent intervals for merging the data. The third step is for splitting the intervals in accordance to the defined criterion. And finally, the fourth step is to find the stopping point for a specific split.

Surgical operation duration in this study, as discussed before, is considered a continuous variable. It was used and treated as such in the regression methods. But for classification models and methods it needed to be discretized. We have discretized the operation durations into time bins and used two methods, Equal Width Binning and Equal Frequency Binning. There are two reasons for using time bins. First, the binning may improve the accuracy of the predictive models by reducing the noise or nonlinearity. And second, binning allows the easy identification of outliers, invalid, and missing values of numerical variables.

3.4.1. Equal Width Binning Approach

In equal width binning, as the name suggests, we use bins with equal intervals i.e. if the width of the first bin is 1500 seconds, the rest of the bins must have the same width. In this approach each bin will have different number of instances and some of the bins may end up being empty or with very few instances. We can use $w = (max - \min)/k$, w being the width of the bin and k being the number of bins. To determine k we can consider the data distribution and the histogram of the data.

3.4.2. Equal Frequency Binning Approach

In equal frequency or equal probability binning we use a different method. In this approach the widths of the bins can vary and are not the same but each bin should have the same number of instances. Similar to the previous case, the best way to determine the number of instances per each bin is to consider the data distribution and the histogram of the data. Figure 12 shows the difference of equal width binning and equal frequency binning with a small example.

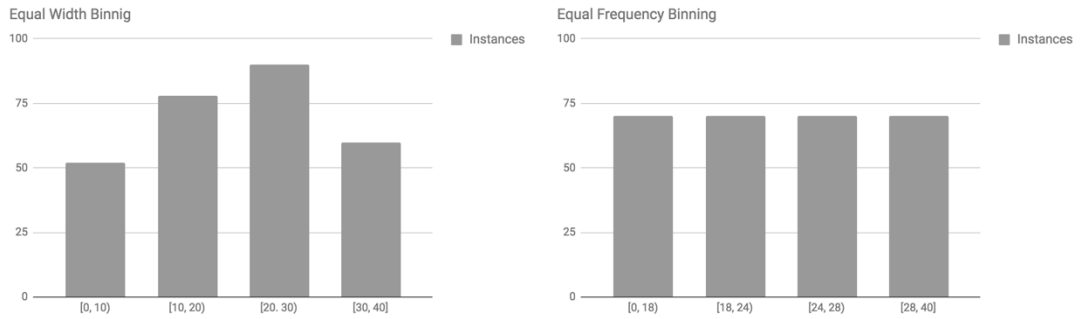

Figure 12: Equal Width Binning vs. Equal Frequency Binning

3.4.3. Hybrid Binning Approach

We tried to discretize surgical operation durations with both equal width binning and equal frequency binning and used them separately in different models. Both approaches had their benefits and advantages, but they were not quite satisfying. As mentioned before each method has its pros and cons. After considering the data distribution and the results of tests we decided to combine these approaches and use a hybrid binning technique. We used equal width binning to discretize the first two third of the data and used equal frequency binning to discretize the last one third. This approach allowed us to have a more concrete and coherent distribution over the bins.

3.5. Finalized Features

Data cleaning process is an essential part of any data related research, especially in machine learning and artificial intelligence. We can consider the features as the workhorses of machine learning [58]. After cleaning the data and feature selection process we ended up with a total number of 23 features to build the models. It is important to mention that the features used in each model varied slightly. The reason is that we updated the features simultaneously with the models and tried to modify them to best fit the selected model. The differences of features, if any, in each model will be mentioned in next chapter in the relative section of that model.

The finalized feature set for this study included three main categories of features. The features related to the patient, the features related to the disease and diagnosis, and finally the features related to the healthcare facility. Table 3 summarizes the finalized features.

Table 3: Finalized Feature Set

24,723 entries divided into three main categories		
Features related to patient	Features related to disease and diagnosis	Features related to the healthcare facility
Gender	Diagnosis Code	Starting time of surgery
Age	Treatment count	Operating room number
ASA physical status classification	Anesthesia method	Operation card method
	Anesthesia count	Clinic
	Operation code	Operation Day
	Operation count	Weekend
		General surgeon
		Anesthesiologist
		Orthopedist
		Urologist
		Gynecologist
		Radiologist
		Medical student
		Staff count

CHAPTER 4

METHODOLOGY AND RESULTS

This chapter is dedicated to the methodology used in this study. We have tried to best predict the operation duration with a wide spectrum of models. These models range from regression models, classical classification models, and finally the classification models based on probabilistic graphical models and Bayesian networks. Given the nature of the available features and the fact that almost all of the features are nominal, or can be easily converted to nominal values, we decided to concentrate on classification models. In this chapter we will go through the regression models first then continue with the classification models.

4.1. Regression Models

Regression models are among the most prevalent and studied algorithms in machine learning and statistical analysis. They have a deep route in statistics and they are generally used for predictive analytics. A large proportion of machine learning algorithms are used for predictive modeling as well. Regression models in machine learning try to predict the outcome and are primarily concerned with minimizing the error of the models and to have the most accurate predictions possible. Machine learning models borrow and reuse a lot of algorithms from different fields including statistics. Regression models were developed for statistics. They try to find the relationships and understand how different variables are related to each other.

We have used four different regression models to predict the surgical operation duration. These models include *Random Forest Regression, SVM Regression, Nearest Neighbors Regression,* and *Adaboost Regression*. Since regression models were the first to be tried in our experiments the features we used were slightly different from the finalized version of features. The cleaned data included 24,773 surgery instances. Table 4 summarizes the features used for regression models.

Table 4: Feature Set Used for Regression Models

24,773 entries divided into three main categories		
Features related to patient	Features related to disease and diagnosis	Features related to the healthcare facility
Gender	Diagnosis Code	Operating room number
Age	Treatment count	Operation card method
ASA physical status classification	Anesthesia method	Clinic
	Anesthesia count	Operation Day
	Operation code	Weekend
	Operation count	General surgeon
		Anesthesiologist
		Orthopedist
		Urologist
		Gynecologist
		Radiologist
		Medical student
		Staff count

There are a total number of 22 features used for regression models. The feature *starting time of surgery* was added later on in the study. The *age* feature was also used without categorization i.e. the inputs were the original values of the data. The following seven features, *general surgeon, anesthesiologist, orthopedist, urologist, gynecologist, radiologist,* and *medical student* were continuous variables indicating the number of that category present in the surgery. The rest of the features were the same.

4.1.1. Random Forest Regression

Random Forest method develops lots of decision trees based on random selection of data and random selection of variables. It provides the class of dependent variable based on many trees. As the trees are based on random selection of data as well as variables, we call them random trees. After creating many of such random trees, we will get a random forest. There are two major principles in random forest method. First, most of the trees can provide correct prediction of class for most part of the data. Second, these trees make mistakes in different places. [59]–[61]

We have created a random forest regression model based on the data using Scikit Learn package in Python and Orange. The obtained results are as follows. The model predicts the operation durations with Root Mean Squared Error (RMSE) of 2399.31 seconds (39.98 minutes) and Mean Absolute Error (MAE) of 1569.73 seconds (26.16 minutes).

4.1.2. SVM Regression

Support Vector Machines (SVMs) try to categorize the data by mapping it to a high dimensional feature space. SVMs can handle the data which is linearly separable as well

as the data which is not linearly separable. The goal of support vector machines is to design or find a hyperplane that classifies the training vectors. SVMs, then, will draw a separator between the categories and transform the data into separable groups. There can be different hyperplanes and separators that can correctly classify the data, but the best choice will be the hyperplane that leaves the maximum margin from all of the classes. After finding the hyperplane, the model can be used to predict the new data into groups. [62], [63]

We have created a SVM regression model based on the data using Scikit Learn package in Python and Orange. The obtained results are as follows. The model predicts the operation duration with Root Mean Squared Error (RMSE) of 3345.82 seconds (55.76 minutes) and Mean Absolute Error (MAE) of 2300.46 seconds (38.34 minutes).

4.1.3. Nearest Neighbors Regression

The Nearest Neighbors algorithm, also called K Nearest Neighbors (kNN), is one of the simple machine learning algorithms. kNN classifies the unlabeled observations by assigning them to the class of the most similar labeled example. The algorithm works by comparing the similarities. The idea is that the entry belongs to the category which it has the most resemblance. When the algorithm wants to classify a point, it will check the K-closest (most similar) neighbors and picks the closest one. kNN works of two principles, a metric to compute the distance between two points and the value of k, the number of neighbors to consider. It is important to remember that if k is a small number the algorithm might overfit and if k is a large number it might underfit. kNN can be used for both regressions as well as classification problems. The average value for the k nearest neighbors will be the prediction value for that entry. [64] – [67]

We have created a kNN regression model based on the data using Scikit Learn package in Python and Orange. The obtained results are as follows. The model predicts the operation duration with Root Mean Squared Error (RMSE) of 2657.72 seconds (44.29 minutes) and Mean Absolute Error (MAE) of 1773.92 seconds (29.56 minutes).

4.1.4. Adaboost Regression

Adaboost is an ensemble, and boosting is an ensemble technique. The goal of boosting is to create a stronger classifier from a number of weak classifiers. Boosting is an iterative method. It is done by creating and training a model over and over again, finding the flaws of each iteration and attempting to correct those errors. This process is terminated once the training set can be predicted perfectly or a maximum number of models are reached. Adaboost was initially developed for binary classification problems and decision trees to boost their performance. [68] – [71]

We have created an Adaboost regression model based on the data using Scikit Learn package in Python and Orange. The obtained results are as follows. The model predicts the operation duration with Root Mean Squared Error (RMSE) of 2344.46 seconds (39.07 minutes) and Mean Absolute Error (MAE) of 1513.25 seconds (25.22 minutes).

4.1.5. Regression Models Summary of Results

Out of the four regression models built based on the data Random Forest regression and Adaboost outperform the other models. But the results are far from satisfactory. The reason why regression models are not performing good can be related to the nature of available features. Most of the features used in this study are nominal. Working with nominal data is more challenging and regression models tend to work better with continuous variables. Table 5 summarizes the results of regression models.

Table 5: Summary of Regression Models Results

Model	**Root Mean Squared Error (RMSE)**	**Mean Absolute Error (MAE)**
Random Forest	2399.31 seconds (39.98 minutes)	1569.73 seconds (26.16 minutes)
SVM	3345.82 seconds (55.76 minutes)	2300.46 seconds (38.34 minutes)
kNN	2657.72 seconds (44.29 minutes)	1773.92 seconds (29.56 minutes)
Adaboost	2344.46 seconds (39.07 minutes)	1513.25 seconds (25.22 minutes)

4.2. Classification Models

Classification algorithms' goal is to identify the correct class of each entry based on the features. In order for a classifier to work, the target variable needs to be categorical. The operation duration, in this study, was a continuous variable. Initially, we categorized the target variable using equal width binning technique with the bin widths of three-minutes, five-minute, and ten-minute bins but the initial results were not good at all. In the second approach we categorized the target variable using equal frequency binning. Each time bin had 3000 surgery instances and we had a total number of nine categories. Table 6 illustrates the bin categories and their respective information.

Figure 13 illustrates the bin intervals. Given the data distribution and the method for binning the operation duration we observed that the bin intervals do not have the same range. The first half of the bins have similar ranges but after the fifth bin the bins start expanding. The reason behind this difference is the surgical operation duration distribution. Most of the surgeries take less than 8000 seconds (the first two third of the data), but the rest of the surgeries take longer and the number of these surgeries are fewer.

Since the method for discretizing the operation duration is equal frequency binning, and in this method we want to have bins with equal sizes the bins expand towards the end.

Table 6: Bins Used for Classification Methods

Bin	Span (seconds)	Span (minutes)	Bin Intervals (minutes)
0	60 - 1020	1 -17	16
1	1020 - 1800	17 - 30	13
2	1800 - 2580	30 - 43	13
3	2580 - 3300	43 - 55	12
4	3300 - 4140	55 - 69	14
5	4140 - 5220	69 -87	18
6	5220 - 6900	87 - 115	28
7	6900 - 12000	115 - 200	85
8	12000 - 33780	200 - 563	363

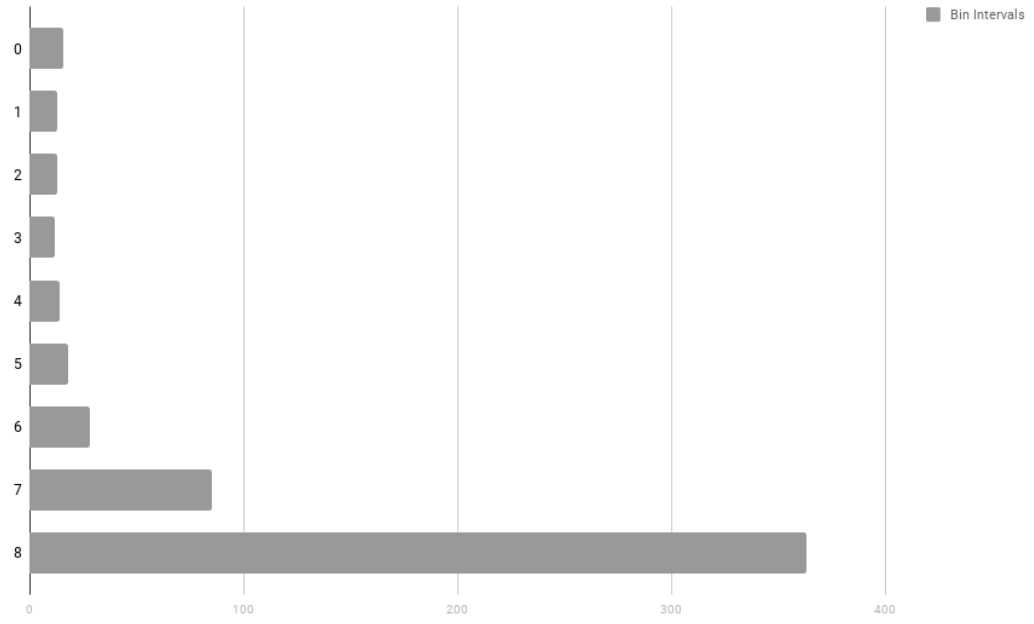

Figure 13: Bin Intervals Distribution

For this trial, we have built three different models based on the following algorithms: Logistic Regression classifier, K Nearest Neighbors classifier, and Naive Bayes classifier. The processed data for these trials included 24,751 surgery instances. Table 7 summarizes the features used for these classification models.

Table 7: Feature Set Used for Classification Models

24,751 entries divided into three main categories		
Features related to patient	Features related to disease and diagnosis	Features related to the healthcare facility
Gender	Diagnosis Code	Operating room number
Age	Treatment count	Operation card method
ASA physical status classification	Anesthesia method	Clinic
	Anesthesia count	Operation Day
	Operation code	Weekend
	Operation count	General surgeon
		Anesthesiologist
		Orthopedist
		Urologist
		Gynecologist
		Radiologist
		Medical student
		Staff count

We used Scikit Learn package in Python and Orange to train the models and predict the outcomes. The models were tested using the ten-fold stratified cross validation technique to avoid overfitting and minimizing the models' bias and variance. Table 4.5 summarizes the classification models' results.

Table 8: Classification Models Results

Method	**Accuracy**	**F1**	**Precision**
Naive Bayes Classifier	35.7%	0.346	0.353
K Nearest Neighbors Classifier	29.0%	0.284	0.291
Logistic Regression Classifier	37.5%	0.368	0.366

4.2.1. Calculating Accuracy and Error in Classification Models

There are different evaluation methods and metrics in machine learning algorithms. These evaluation methods depend on the type and nature of the method. Regression methods are usually evaluated by Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE) whereas classification methods are usually evaluated by classifier accuracy, precision, and F1 score. Root mean squared error is the square root of the mean of the square of all of the errors in the model. RMSE is an excellent general purpose error metric for numerical predictions. RMSE amplifies and severely punishes large errors. RMSE is calculated using the following formula where y_i is the actual value and $\hat{y}_i$ is the predicted value.

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}$$

Mean absolute error is used to measure how close forecasts or predictions are to the eventual outcomes. MAE is calculated using the following formula where y_i is the actual value and $\hat{y}_i$ is the predicted value.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

As mentioned before classification methods use different evaluation criteria to evaluate the results. Here are their definitions and how they are calculated.

Terms:

- Positive (P): the observation is positive
- Negative (N): the observation is negative
- True Positive (TP): the observation is positive and is predicted to be positive
- False Negative (FN): the observation is positive but is predicted negative
- True Negative (TN): the observation is negative and is predicted to be negative
- False Positive (FP): the observation is negative but is predicted positive

Error:

Proportion of all predictions that are incorrect.

$$Error = \frac{FP + FN}{FP + FN + TP + TN} = \frac{incorrect\ predictions}{all\ predictions}$$

Accuracy:
Proportion of all predictions that are correct.

$$Accuracy = \frac{TP + FN}{TP + FN + FP + TN} = \frac{correct\ predictions}{all\ predictions}$$

False Positive Rate:
Proportion of all negative observations that are predicted correctly.

$$FPR = \frac{FP}{FP + TN} = \frac{FP}{N} = \frac{predicted\ to\ be\ positive}{all\ negative\ observations}$$

True Positive Rate:
Proportion of all positive observations that are predicted correctly.

$$TPR = \frac{TP}{TP + FN} = \frac{TP}{P} = \frac{predicted\ to\ be\ positive}{all\ positive\ observations}$$

Precision:
Proportion of all positive predictions that are correct.

$$Precision = \frac{TP}{TP + FP} = \frac{positive\ predicted\ correctly}{all\ positive\ predictions}$$

Recall:
Proportion of all real positive observations that are correct.

$$Recall = TPR = \frac{TP}{TP + FN} = \frac{TP}{P} = \frac{predicted\ to\ be\ positive}{all\ positive\ observations}$$

F1 Score:
F1 score is the harmonic mean of precision and recall.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

The target value in this study is surgical operation duration and as mentioned before it is a continuous variable by its nature. We discretized the surgical operation duration into time bins and used classification models to predict the duration. This being said, we can use evaluation methods from both regression evaluation methods and classification evaluation methods. We have decided to use RMSE, MAE, and accuracy to evaluate the models. Because of the nature of the data and the target variable, we have slightly changed how these criteria metrics are calculated.

We consider the center of each bin as the average time for that specific bin. When calculating RMSE we get the error value by subtracting the actual duration of each surgery from the predicted bin's midpoint. We use the same calculated error to get the MAE as well. In normal cases, accuracy is just the number of correct hits divided by the number of all instances. Since the target value in this study is time and it is a continuous variable, the actual time of each surgical operation can fall anywhere in the bin or in some cases very close to the actual bin but not in it. The accuracy in this manner can be defined depending on the situation. An accurate result would be, therefore, not only the fact that we have predicted the correct bin but also how far we are off from the actual value. In figure 14, as an example, there's a surgery with the duration of 56 minutes which belongs to the 14th bin. In a normal case, if a classifier predicts the 13th bin, it would be inaccurate. Since the error of this particular prediction is 7 minutes, depending on the definition of accuracy or expectations of the model, this can be a correct hit as well, e.g. if the definition of a correct hit is "all the predictions which have an error of less than or equal to ten minutes", this prediction is still considered a correct bin regardless of not being in the correct bin.

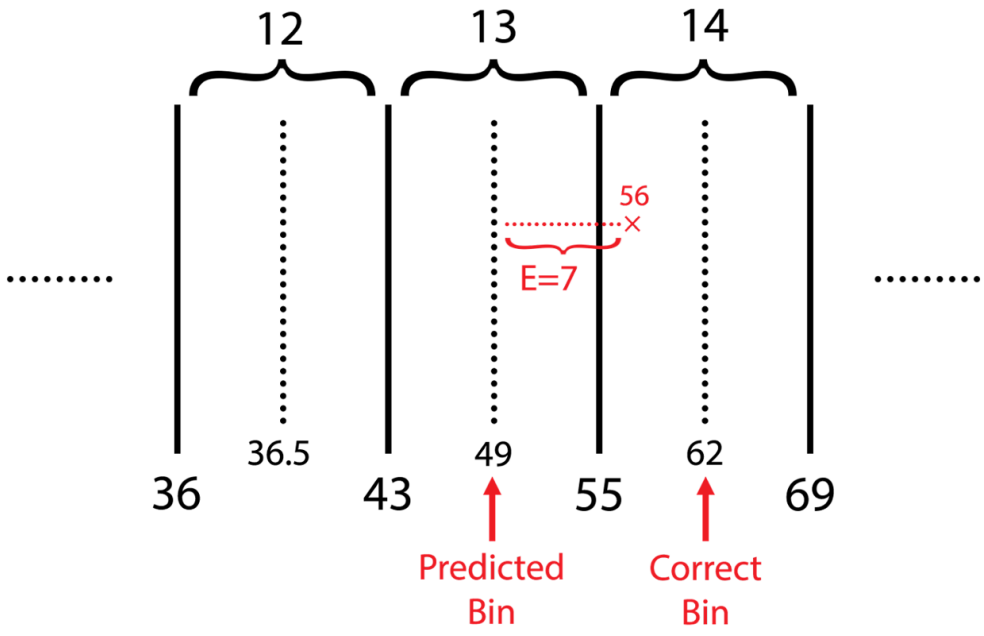

Figure 14: Illustration of a Hypothetical Prediction Example

4.2.2. Hybrid Binning Approach for Classification Models

In the next step we tried categorising the target value using a hybrid of equal width binning and equal frequency binning techniques. The reason for this decision was to have a more lenient and similar bin ranges overall the whole dataset. After categorising the surgery durations using this method we ended up having 21 distinct time bins. We also added one new feature to the feature list. *Patient start,* it is, as explained before, the time when the patient was admitted to the healthcare facility for the surgery. At this point, however, it wasn't categorised to four distinct groups. It was used as extracted, a number indicating the starting hour of the procedure. Table 9 illustrates the bin categories and their respective information.

The hybrid binning approach divided the large bins in the end of the dataset into smaller bins. Figure 15 illustrates bin ranges for each bin.

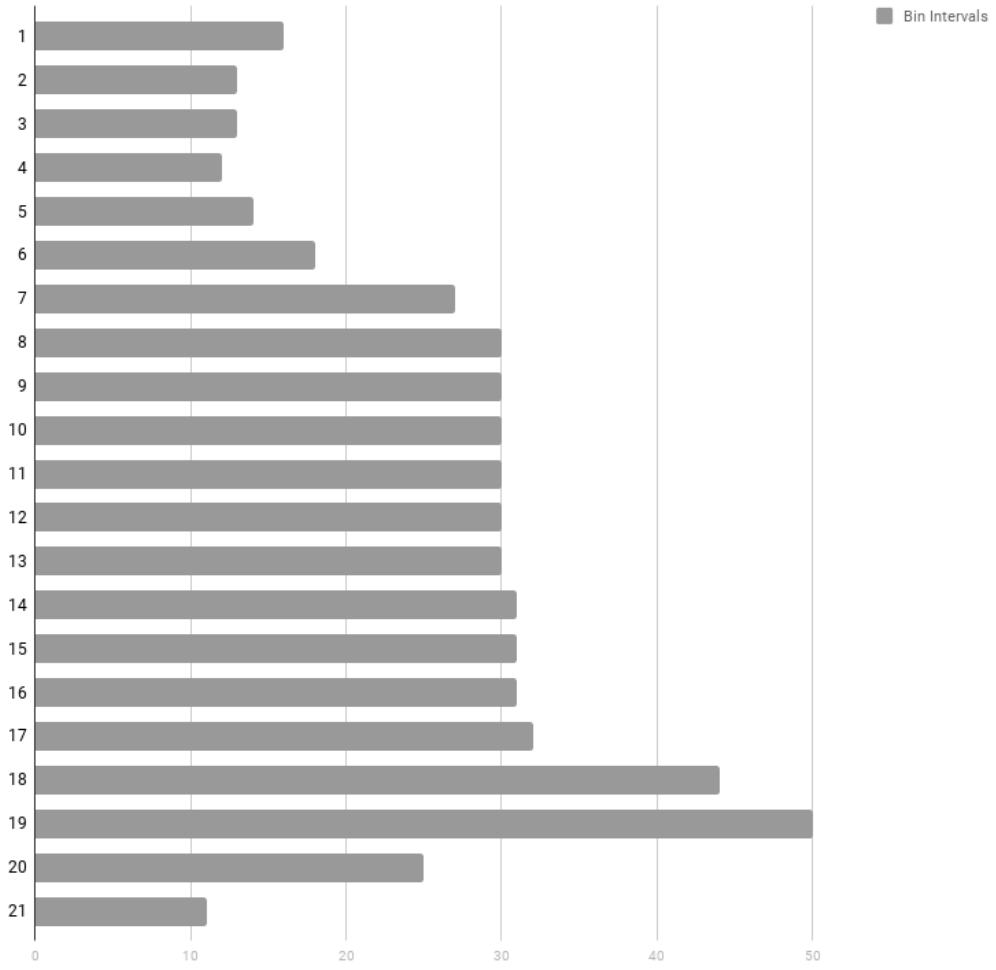

Figure 15: Bin Intervals Distribution Using Hybrid Binning Techniques

Table 9: Created Bins Using Equal Frequency Binning and Equal Width Binning Techniques

Bin	Span (seconds)	Span (minutes)	Bin Intervals (minutes)	Number of Instances
0	60 -1020	1 - 17	16	3000
1	1020 - 1800	17 - 30	13	3000
2	1800 - 2580	30 - 43	13	3000
3	2580 - 3300	43 - 55	12	3000
4	3300 - 4140	55 - 69	14	3000
5	4140 - 5220	69 - 87	18	3000
6	5220 - 6840	87 - 114	27	2996
7	6900 - 8760	115 - 146	30	1712
8	8760 - 10620	146 - 177	30	877
9	10620 - 12480	177 - 208	30	506
10	12480 - 14340	208 - 239	30	264
11	14340 - 16200	239 - 270	30	172
12	16800 - 18060	270 - 301	30	89
13	18060 - 19980	301 - 333	31	57
14	19980 - 21900	333 - 365	31	29
15	21900 - 23820	365 - 397	31	16
16	23820 - 25800	397 - 430	32	9
17	25800 - 28500	430 - 475	44	6
18	28500 - 31560	475 - 526	50	4
19	31560 - 33060	526 - 551	25	3
20	33060 - 33780	551 - 563	11	1

Table 10 summarizes the features used for training Naive Bayes and logistic regression models using the hybrid binning techniques.

Table 10: Feature Set Used for Classification Models with Hybrid Binning Techniques

24,751 entries divided into three main categories		
Features related to patient	Features related to disease and diagnosis	Features related to the healthcare facility
Gender	Diagnosis Code	Starting time of surgery
Age	Treatment count	Operating room number
ASA physical status classification	Anesthesia method	Operation card method
	Anesthesia count	Clinic
	Operation code	Operation Day
	Operation count	Weekend
		General surgeon
		Anesthesiologist
		Orthopedist
		Urologist
		Gynecologist
		Radiologist
		Medical student
		Staff count

Using the hybrid binning approach improved the results and lowered the overall errors of the models. Naive Bayes classifier and Logistic Regression classifier were chosen to train and test the models. We used Scikit Learn package in Python and Orange to build and test the models. The models were tested using the ten-fold stratified cross validation technique to avoid overfitting and minimize the models' bias and variance. Logistic Regression classifier had an accuracy of 38.88%, RMSE of 43.73 minutes, and MAE of 26.47 minutes. Naive Bayes classifier had an accuracy of 49.0%, RMSE of 38.77 minutes, and MAE of 25.52 minutes. These new models had major improvements in accuracy and error. RMSE was dropped, on average, around 14 minutes.

4.3. Bayesian-Network Based Classification Models

Naive Bayes classifier was one of the promising models in previous attempts to predict the surgical operation duration. Naive Bayes classifier can be considered as one of the simplest models of the whole Bayesian Network (BN) models which by itself is part of probabilistic graphical models. Given the fact that the results of the models were promising when using Naive Bayes classifier, we decided to continue working with Bayesian-Network based classification models. The use of Bayesian Networks have increased in recent years [23], [72], [73]. According to Gamez et al. there are four reasons for the prevalence of Bayesian Networks: "(1) Bayesian Networks mathematical basis is rigorously justified; (2) they deal in an innate way with uncertainty (modeled as a joint probability distribution); (3) they are understandable because of their graphical

representation; and (4) they take advantage of locality both in knowledge representation and during inference." These models are built based on the prior distribution of the data and create a posterior distribution based on the feature relations and their conditional properties in a Bayesian Network. Once the network has been trained, the predictions can be made by using the posterior distributions of the features.

There are many different algorithms that allow us to build Bayesian Networks form the raw data. In this study we used five different algorithms to build Bayesian Networks and tested the models based on these created Bayesian Networks. These algorithms include *Feature-Correlation based BN, Tabu Search based BN, Hill-Climbing based BN, Max-Min Hill-Climb based BN,* and finally *Naive Bayes based BN*. We created the Bayesian networks based on the abovementioned algorithms and trained and tested the data with these models.

Before beginning to create Bayesian Network models we redid the data preparation process and changed the features slightly. *Patient start time* was further categorized to four categories; *age* was further categorised to four categories; *general surgeon, anesthesiologist, orthopedist, urologist, gynecologist, radiologist,* and *medical student* were converted to boolean features.

Table 11: New Feature Configurations for Bayesian Network Models

Feature	**Previous Configuration**	**Updated Configuration**
Patient Start Time	1 category per each hour	4 categories: [0, 7) [7, 12) [12, 18) [18, 24)
Age	1 category per each age	4 categories: [0, 20) [20, 40) [40, 60) [60, Inf.)
General Surgeon Anesthesiologist Orthopedist Urologist Gynecologist Radiologist Medical Student	Number personnel of each feature present in the surgery	Boolean values 1: present in the surgery 0: not present in the surgery

The processed data for these trials included 24,723 surgery instances. Table 12 summarizes the features used for these classification models. We created, trained, and tested the models using R and bnlearn package.

Table 12: Finalized Feature Set Used for Bayesian Network Models

24,723 entries divided into three main categories		
Features related to patient	Features related to disease and diagnosis	Features related to the healthcare facility
Gender	Diagnosis Code	Starting time of surgery
Age	Treatment count	Operating room number
ASA physical status classification	Anesthesia method	Operation card method
	Anesthesia count	Clinic
	Operation code	Operation Day
	Operation count	Weekend
		General surgeon
		Anesthesiologist
		Orthopedist
		Urologist
		Gynecologist
		Radiologist
		Medical student
		Staff count

4.3.1. Feature-Correlation Based Bayesian Network

The created Bayesian Network is based on the partial correlation of the features. In order to create this model, the correlation of the features was calculated using Chi-Square test. We used Cramer's V to compare the magnitude of effect between features, then we built the correlation matrix out of these values. Later, we used the correlation matrix to calculate the partial correlation between the features. Partial correlation is a measure of the strength and direction of a linear relationship between two variables whilst controlling for the effect of one or more other variables. We used *Pearson, Spearman,* and *Kendall's* estimates to create the partial correlation matrix of the features, then created a heat map based on these outputs.

We have created a Bayesian Network based on the partial correlation of features based on Spearman's estimators heat map using R and bnlearn package. Figure 16 represents the heat map created based on Spearman's estimation of partial correlations and Figure 17 represents the created network generated by bnlearn package.

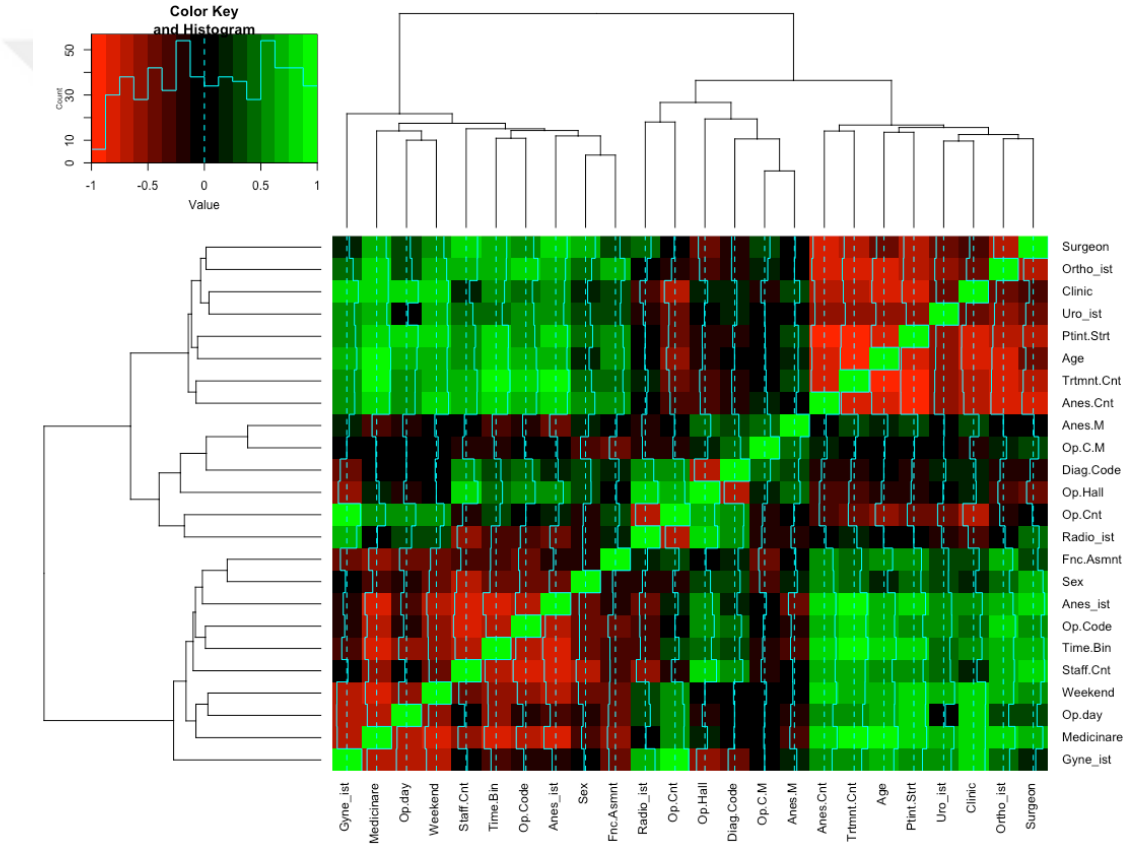

Figure 16: The Heat Map Created Based on Spearman's Estimation of Partial Correlations

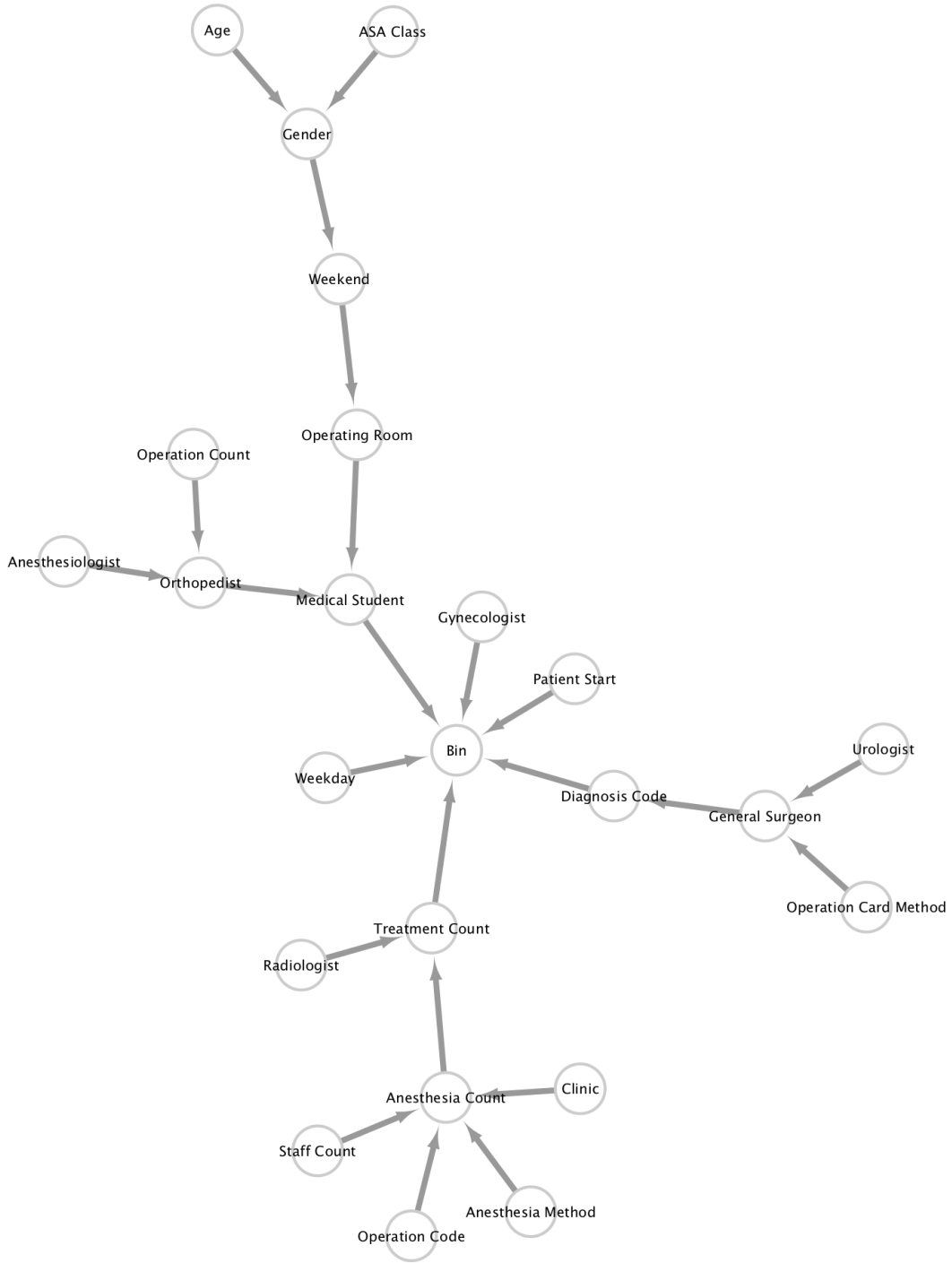

Figure 17: Bayesian Network Created Based on Spearman's Correlation Estimations

We have trained and tested the model in ten iterations and reported the average as the result. We have used bootstrapping when creating the subsets of the data for the training dataset and testing dataset. Table 13 summarizes the ten iteration results for this model.

Table 13: Ten-Fold Cross Validation Results for Partial-Correlation Based Bayesian Network

Iteration Number	Accuracy	RMSE (seconds)	MAE (seconds)
1	61.65%	2334.81	1567.92
2	60.03%	2288.14	1516.02
3	58.88%	2312.54	1577.41
4	59.53%	2226.54	1487.54
5	62.34%	2202.76	1443.38
6	61.73%	2300.87	1500.73
7	61.20%	2387.61	1580.73
8	60.88%	2349.01	1508.12
9	59.89%	2318.63	1533.28
10	60.06%	2230.33	1489.06
Average	60.61%	2295.12 (38.25 minutes)	1520.41 (25.34 minutes)

4.3.2. Tabu Search Based Bayesian Network

Tabu Search is a score-based structure learning algorithm. According to Nagarajan et al. "Score-based structure learning algorithms (also known as search-and-score algorithms) represent the application of general heuristic optimization techniques to the problem of learning the structure of a Bayesian Network. Each candidate network is assigned a *network score* reflecting its goodness of fit, which the algorithm then attempts to maximize." [74] Tabu Search can further be categorized as a greedy search algorithm. "These algorithms explore the search space starting from a network structure (usually the empty graph) and adding, deleting, or reversing one arc at a time until the score can no longer be improved." [74], [75]

Algorithm 1: Greedy Search Algorithm

1. Choose a network structure G over $\mathbf{V}$, usually (but not necessarily) empty.
2. Compute the score of G, denoted as $Score_G = \text{Score}(G)$.
3. Set $maxscore = Score_G$.
4. Repeat the following steps as long as *maxscore* increases:
   a. for every possible arc addition, deletion or reversal not resulting in a cyclic network:
      i. compute the score of the modified network G^*, $Score_{G^*} = \text{Score}(G^*)$:
      ii. if $Score_{G^*} > Score_G$, set $G = G^*$ and $Score_G = Score_{G^*}$.
   b. update *maxscore* with the new value of $Score_G$.
5. Return the directed acyclic graph G.

We have created a Bayesian Network based on Tabu Search algorithm using R and bnlearn package. Figure 18 represents the created network generated by bnlearn package.

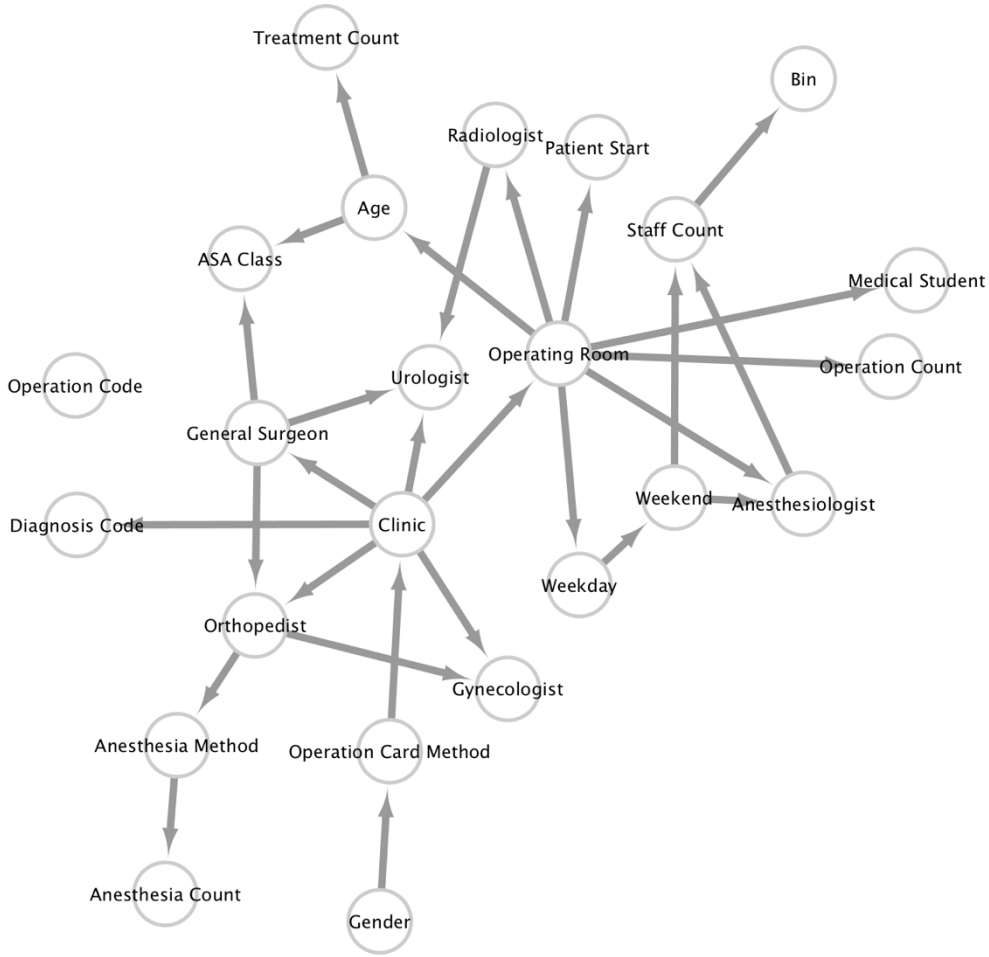

Figure 18: Bayesian Network Created by Tabu Search Algorithm

We have trained and tested the model in ten iterations and reported the average as the result. We have used bootstrapping when creating the subsets of the data for the training dataset and testing dataset. Table 14 summarizes the ten iteration results for this model.

Table 14: Ten-Fold Cross Validation Results for Tabu Search Algorithm

Iteration Number	**Accuracy**	**RMSE (seconds)**	**MAE (seconds)**
1	35.76%	3301.95	2407.54
2	34.02%	3378.09	2414.98
3	38.12%	3214.76	2356.76
4	37.01%	3311.54	2400.87
5	37.47%	3356.13	2406.80
6	35.03%	3312.41	2387.04
7	36.49%	3480.13	2443.65
8	35.89%	3312.11	2356.45
9	34.86%	3288.74	2378.64
10	35.96%	3277.51	2365.87
Average	36.06%	3323.34 (55.39 minutes)	2391.86 (39.86 minutes)

4.3.3. Hill-Climbing Based Bayesian Network

Hill-climbing is also a score-based structure learning algorithm. Hill-climbing search traverses the search space by starting from an initial solution and performing a finite number of steps. At each step the algorithm only considers local changes, i.e. neighbor DAGs, and chooses the one resulting in the greatest improvement the scoring metric. [76]

Algorithm 2: Hill-Climbing Algorithm

Input: D: A dataset defined over variables $\mathbf{V} = \{X_1, \ldots, X_n\}$
Input: $\mathcal{G}_0$: A DAG defined over $\mathbf{V}$ used as the starting point for the search
Output: A DAG $\mathcal{G}$ being the graphical part of network $\mathcal{B}$

$\mathcal{G} \leftarrow \mathcal{G}_0$;
$f_{\mathcal{G}} \leftarrow f(\mathcal{G} : D)$ //*
[r]f: decomposable scoring metric improvement $\leftarrow$ true;
while *improvement* **do**
    improvement $\leftarrow$ false;
    //*
    [h]neighbors generated by addition
    For each node X_i and each node $X_j \notin Pa_{\mathcal{G}}(X_i)$ such that $X_j \rightarrow X_i$ does not introduce a directed cycle in $\mathcal{G}$, compute the difference $diff = f(\mathcal{G} + \{X_j \rightarrow X_i\} : D) - f_{\mathcal{G}}$. Store the change which maximizes $diff$ in $\langle change_a, diff_a \rangle$;
    //*
    [h]neighbors generated by deletion
    For each node X_i and each node $X_j \in Pa_{\mathcal{G}}(X_i)$, compute the difference $diff = f(\mathcal{G} - \{X_j \rightarrow X_i\} : D) - f_{\mathcal{G}}$. Store the change which maximizes $diff$ in $\langle change_d, diff_d \rangle$;
    //*
    [h]neighbors generated by reversal
    For each node X_i and each node $X_j \in Pa_{\mathcal{G}}(X_i)$ such that reversing $X_j \rightarrow X_i$ does not introduce a directed cycle in $\mathcal{G}$, compute the difference $diff = d_1 + d_2$ where d_1 corresponds to $f(\mathcal{G} - \{X_j \rightarrow X_i\} : D) - f_{\mathcal{G}}$ and d_2 corresponds to $f(\mathcal{G} + \{X_j \rightarrow X_i\} : D) - f_{\mathcal{G}}$. Store the change which maximizes $diff$ in $\langle change_r, diff_r \rangle$;
    //*
    [h]Checking if improvement
    Let $d^* = max_{k=a,d,r} diff_k$ and $move^*$ its corresponding change;
    if $d^* > 0$ **then**
        improvement $\leftarrow$ true;
        $\mathcal{G} \leftarrow$ apply $move^*$ over $\mathcal{G}$;
        $f_{\mathcal{G}} \leftarrow f_{\mathcal{G}} + d^*$;
    end
end
return $\mathcal{G}$;

We have created a Bayesian Network based on Hill-Climbing Search algorithm using R and bnlearn package. Figure 19 represents the created network generated by bnlearn package.

Figure 19: Bayesian Network Created by Hill-Climbing Algorithm

We have trained and tested the model in ten iterations and reported the average as the result. We have used bootstrapping when creating the subsets of the data for the training dataset and testing dataset. Table 15 summarizes the ten iteration results for this model.

Table 15: Ten-Fold Cross Validation Results for Hill-Climbing Algorithm

Iteration Number	Accuracy	RMSE (seconds)	MAE (seconds)
1	36.29%	3309.16	2395.10
2	37.05%	3280.53	2365.43
3	36.83%	3300.42	2370.12
4	35.27%	3340.61	2301.87
5	36.76%	3279.54	2348.73
6	36.14%	3266.99	2333.18
7	37.78%	3246.47	2284.35
8	36.80%	3310.43	2307.44
9	34.93%	3349.01	2337.90
10	36.41%	3311.54	2323.80
Average	36.38%	3299.47 (54.99 minutest)	2336.79 (38.94 minutes)

4.3.4. Max-Min Hill-Climbing Based Bayesian Network

Max-Min Hill-Climbing (MMHC) is a hybrid structure learning algorithm. Hybrid structure learning algorithms are a combination of constraint-based and score-based algorithms. MMHC algorithms combine constraint-based and score-based algorithms to offset their weaknesses and produce reliable networks structures in a wide variety of situations. MMHC works based on two steps called restrict and maximize. The first step reduces the candidate set for the parents of each node from the whole node set to a smaller set. The second step tries to find the network that maximizes a given score function. [74]

We have created a Bayesian Network based on Max-Min Hill-Climbing Search algorithm using R and bnlearn package. Figure 20 represents the created network generated by bnlearn package.

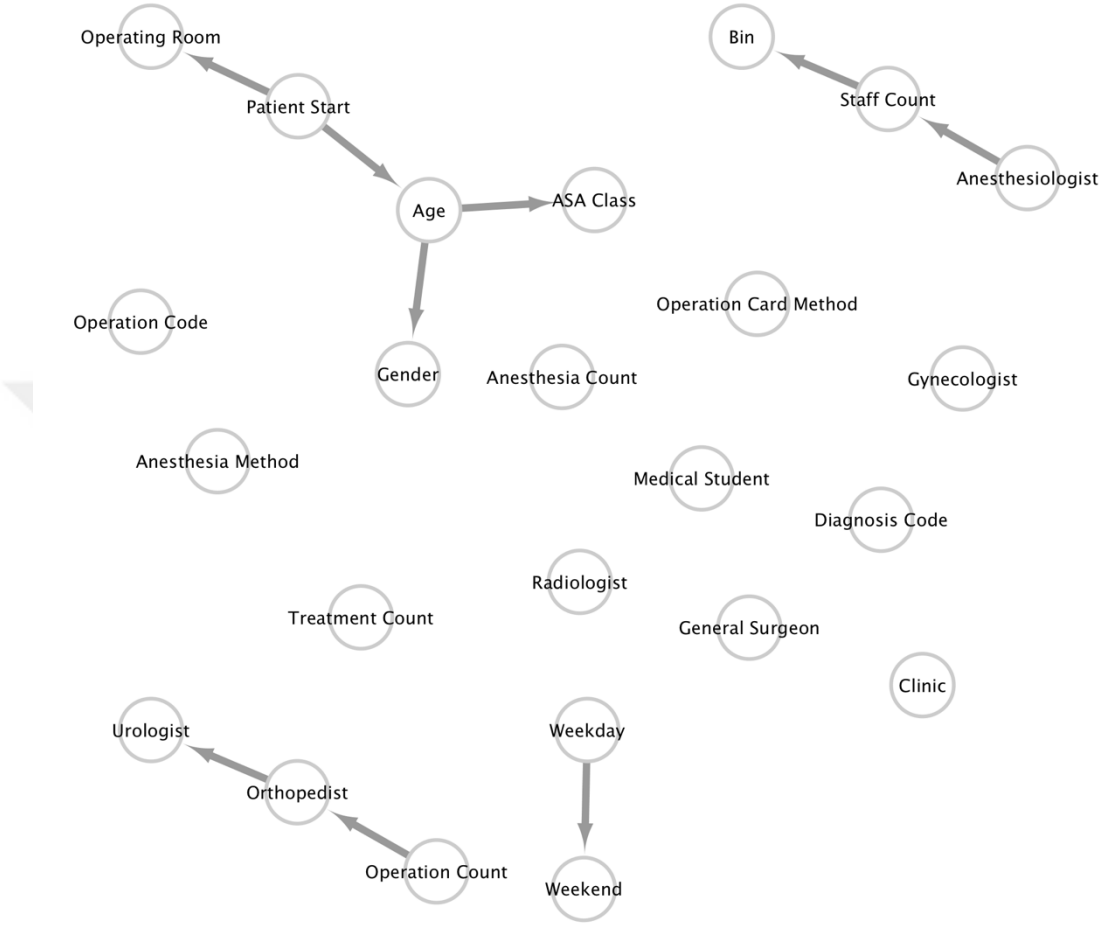

Figure 20: Bayesian Network Created by Max-Min Hill-Climbing Algorithm

We have trained and tested the model in ten iterations and reported the average as the result. We have used bootstrapping when creating the subsets of the data for the training dataset and testing dataset. Table 16 summarizes the ten iteration results for this model.

Table 16: Ten-Fold Cross Validation Results for Max-Min Hill-Climbing Algorithm

Iteration Number	Accuracy	RMSE (seconds)	MAE (seconds)
1	37.94%	3126.91	2291.28
2	38.31%	3100.14	2220.45
3	37.65%	3145.76	2234.50
4	36.70%	3109.44	2189.56
5	37.19%	3121.47	2245.12
6	39.01%	3091.89	2266.31
7	37.26%	3169.83	2270.38
8	37.00%	3180.11	2231.83
9	36.86%	3178.42	2246.39
10	37.28%	3102.12	2239.71
Average	37.52%	3132.61 (52.21 minutes)	2243.55 (37.39 minutes)

4.3.5. Naïve Bayes Based Bayesian Network

As mentioned before, Naive Bayes is by far the simplest Bayesian Network. Naive Bayes makes two assumptions about the feature set. First, all the features are independent, and second, the effect of the value of a predictor on a given class is independent of other predictors. Naive Bayes models have outperformed all the models that we have created so far. Perhaps the best explanation come from Zhang [77]:

> *"The basic idea comes from the observation as follows. In a given dataset, two attributes may depend on each other, but the dependence may distribute evenly in each class. Clearly, in this case, the conditional independence assumption is violated, but Naive Bayes is still the optimal classifier. Further, what eventually affects the classification is the combination of dependencies among all attributes. If we just look at two attributes, there may exist strong dependence between them that affects the classification. When the dependencies among all attributes work together, however, they may cancel each other out and no longer affect the classification. Therefore, we argue that it is the distribution of dependencies among all attributes over classes that affects the classification of naive Bayes, not merely the dependencies themselves."*

In short Naive Bayes is good not only when the features are independent, but also when dependencies of features from each other are similar between features.

We have created a naive Bayesian Network using R and bnlearn package. Figure 21 represents the created network generated by bnlearn package.

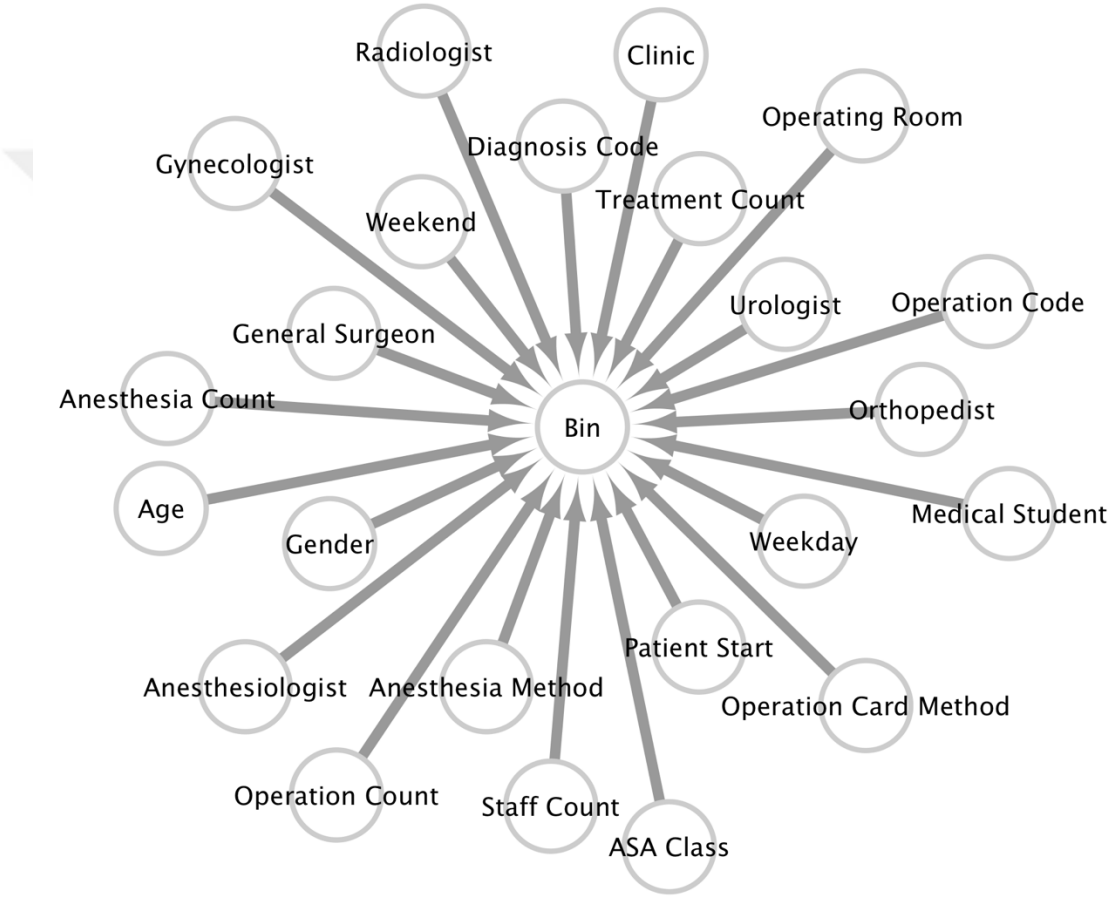

Figure 21: Naive Bayesian Model of the Features

We have trained and tested the model in ten iterations and reported the average as the result. We have used bootstrapping when creating the subsets of the data for the training dataset and testing dataset. In case of this model, we tried the sampling without bootstrapping as well. Table 17 summarizes the ten iteration results for this model with bootstrapping and Table 18 summarizes the ten iteration results for this model without bootstrapping.

Table 17: Ten-Fold Cross Validation with Bootstrapping Results for Naive Bayes

Iteration Number	Accuracy	RMSE (seconds)	MAE (seconds)
1	67.00%	1045.25	1284.44
2	66.79%	843.08	1277.65
3	67.60%	957.66	1317.05
4	67.27%	1049.38	1280.31
5	66.42%	942.11	1263.16
6	67.60%	912.30	1277.73
7	65.61%	789.25	1366.84
8	67.07%	1154.35	1293.45
9	65.49%	1269.30	1331.21
10	67.80%	1027.75	1245.43
Average	66.87%	998.84 (16.64 minutes)	1293.72 (21.56 minutes)

Table 18: Ten-Fold Cross Validation without Bootstrapping Results for Naive Bayes

Iteration Number	Accuracy	RMSE (seconds)	MAE (seconds)
1	55.13%	1068.27	2088.45
2	54.69%	1266.70	2077.25
3	55.82%	1101.18	2017.45
4	54.04%	1102.30	2089.62
5	56.06%	1120.27	2043.30
6	56.75%	1465.98	2124.18
7	55.46%	1213.58	2039.52
8	53.92%	1253.13	2090.24
9	56.43%	1122.47	1990.08
10	56.53%	1286.49	2096.20
Average	55.48%	1200.04 (20 minutes)	2056.63 (34.42 minutes)

CHAPTER 5

CONCLUSION

The purpose of this research study was to predict and estimate the duration of individual surgical operations based on the available historical data. We have experimented numerous supervised machine learning algorithms including regression, classification, and probabilistic graphical models. These models were trained with the data of surgical operations scattered throughout a year. This study was mainly composed of five phases. We began by preparing the data in data preparation phase. We cleaned the data, merged the duplicate values, removed the anomalies, replaced special characters in texts, and converted the dates to timestamp. The second phase included selecting and extracting the features from the original data. We have extracted eighteen features and used the rest of features as they were in the original data. We have discretized the surgical operation durations into time bins as well. We have built and tested four different regression models in the third phase including Random Forest, Support Vector Machine, k Nearest Neighbors, and Adaboost. We have moved to classification models in the fourth phase and tried out Naive Bayes classifier, kNN classifier, and Logistic Regression Classifier. We used three different approaches to discretize the surgical operation durations namely, equal width binning, equal frequency binning, and a hybrid of abovementioned binning techniques. Finally, in the last phase we have tried the Bayesian-Network based classification models. We built five distinct networks and computed the posterior probability distribution of the target features and used these distributions to predict the duration of surgical operations.

5.1. Summary of Result

Here in this section the results of all models are summarized. We began testing with regression models and built models based on Random Forest, Support Vector Machine, k Nearest Neighbors, and Adaboost algorithms. Models based on Random Forest and Adaboost had around 40 minutes of RMSE and generated better results in comparison to SVM and kNN. These results, however, were not satisfying in application. There errors were still relatively high when compared to the duration of surgical operations and they needed to be lower, otherwise the estimations are not applicable in real-world applications. Figure 22 summarizes the results of regression models.

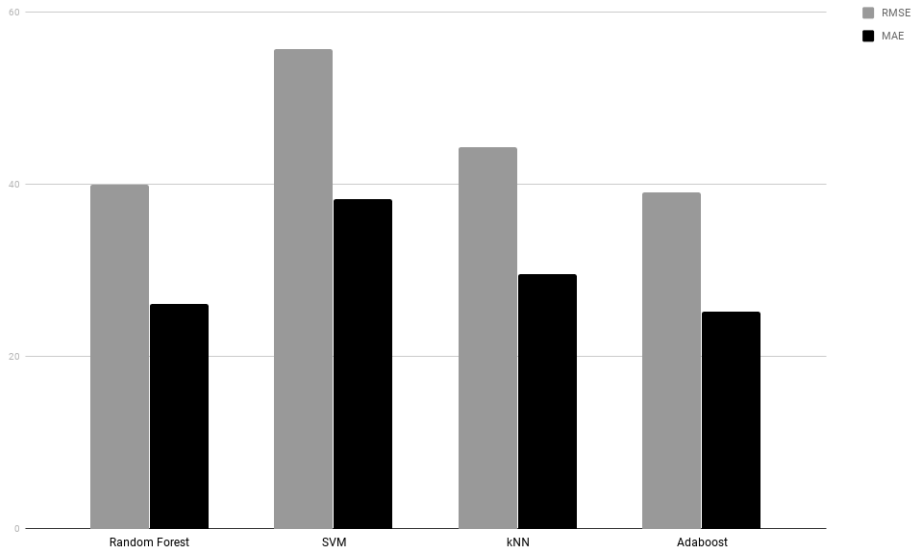

Figure 22: Summary of Regression Models' Results

The next attempt to predict the surgical operation durations was to build models based on the classification algorithms. In order to convert the target feature to conform the prerequisites of classification algorithms, we discretized the surgical operation durations into time bins. We started with equal frequency binning and equal width binning techniques but later used a hybrid of mentioned methods to have a more consistent distribution of surgical operations in time bins. We used three different algorithms to predict the surgical operation durations namely, Naive Bayes, K Nearest Neighbors, and Logistic Regression. KNN and Logistic Regression methods had relatively better results. They had, on average, an accuracy of 36% which was boosted up to 49% with the better configuration of features and binning techniques. The RMSE for the Naive Bayes Classifier was 38.88 minutes which was, on average, 14 minutes less than the previous models. Even though the results had improved over the attempt, they were still far from a usable and practical amount. Figures 23 summarizes the results of classification models with equal width binning and hybrid binning.

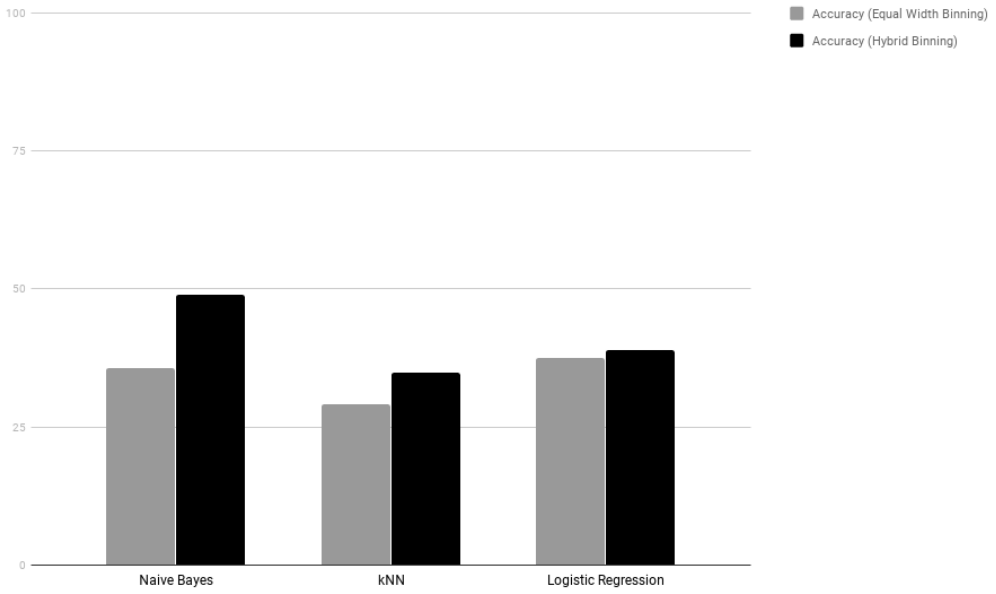

Figure 23: Summary of Classification Models' Results

In the last attempt, we have created Bayesian-Network Based classification models to predict the surgical operation durations. We created five different Bayesian Networks based on different algorithms and trained these networks using the data, then we used the posterior distributions from the trained models to predict the surgery durations. Figure 24 summarizes the results of these models.

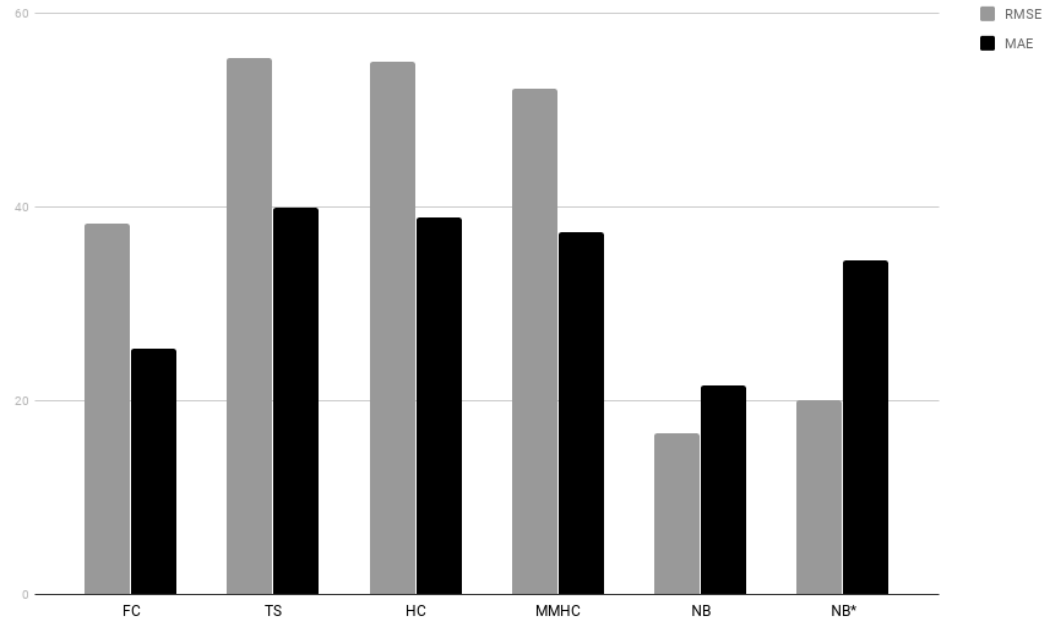

Figure 24: Summary of Bayesian-Network-Based Classification Models' Results

The networks created based on three existing algorithms namely, Tabu Search (TS), Hill-Climbing (HC), and Max-Min Hill-Climbing (MMHC) did not perform better than previous attempts. They had on average 38 minutes of RMSE. The model based on the network created by the feature correlations has better results but at the end Naive Bayes (NB) outperformed all of the models. We used bootstrapping when sampling the data for training and testing the models but we tried the conservative sampling (without bootstrapping) for Naive Bayes (NB*) which still performed better than the other models.

5.2. Discussion

Since the target value of this study was the duration of surgical opearions, and time in its nature is a continuous variable, the initial approach to predict the time was intrinsic. We tried four different regression methods before we decided to move on to classification methods. The main reason behind this decision was not just because we could not get promising results. It may be possible to get accurate results with regression models as well. We changed the direction of this study after considering the circumstances of the target value and the goal of the study. Even though the goal of this study is to predict the duration of surgical operations and the optimal model would be the model with minimum error, there is a tradeoff between how accurate the expected model should be and error of each prediction. Given the fact that all of the models are prone to have errors and a threshold of error is acceptable, we decided to discretize the surgical operation durations.

We tried various classification models and selected the best three to further configure them in the case of this study. We tried two different approaches to discretize the surgical operation durations to have a better representation of the time bins. Since the results of Naive Bayes classifier were promising, we decided to build different classifiers based on probabilistic graphical models and Bayesian networks.

After considering the results of different models created for this study, it is obvious that classification methods outperform regression methods. It is either because of the fact that the surgery durations are very stochastic and heteroscedastic due to their complicated and intrinsic nature [15], making it hard for regression algorithms to perform optimally; or it is because of the nature of data (the data is mostly comprised of nominal and categorical features).

In the comparison between the classification models, Naive Bayes has the most accurate results with lower errors and this is not surprising. Naive Bayes models have a history of outperforming other models. There have been extensive studies to determine why Naive Bayes works better [77] – [80]. Naive Bayes generally performs better than other models under two general circumstances; first, the features don't have significant correlation between each other making it redundant and counterproductive to try and find relations between independent features; and second, the dependencies between the features are the

same among all features which makes the correlations of features ineffective when it comes to building the model [77].

5.3. Future Works

We solely focused on estimating the surgical operation duration in this research study. The same models using the same features can be adapted in order to predict the duration of pre-surgical and post-surgical operations. That being said, we believe that the current model can still be improved by adding related features and changing the configuration of current features. We believe that by adding two extra features, namely body mass index (BMI) and the type of surgery being acute (emergency operations) or elective (planned operations), the predictions can be more accurate with lower errors. According to Eijkemans et al. BMI has a positive correlation with the duration of surgical operations, the higher the BMI the longer the duration of each surgical operation [5]. The mentioned features were not available in working data for this study. BMI can easily be calculated by dividing the weight of the patient (in kilograms) by the square of height (in meters), if it is not recorded in patients' health records.

The operation code feature was expected to have a significant effect on the models but it was left out in the Bayesian Networks created by Tabu Search, Hill-Climbing, and Max-Min Hill-Climbing algorithms. This feature contains more than 1200 distinct values and this effects the how the algorithms treat this feature, generally leaving it out because of its diverse values. We propose to change this feature and reconfigure it with a better taxonomy to be able to use the feature properly.

Additionally, the models created in this study can be tested using new data from other healthcare facilities to confirm the obtained results. Given the adaptive nature of these models, they will adjust their estimations based on the new data inputs. This process allows the models to find new patterns in the data and adjust the respective posterior distributions. Lastly, with obtaining better and more accurate estimations for all three parts of surgical operation durations it is possible to build optimization models using the available software and constraint modeling languages such as MiniZinc.

REFERENCES

[1] S. Dua, U. R. Acharya, and P. Dua, Eds., *Machine Learning in Healthcare Informatics*, vol. 56. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014.

[2] G. D. Magoulas and A. Prentza, "Machine Learning in Medical Applications," 2001, pp. 300–307.

[3] N. H Ng, R. A. Gabriel, J. McAuley, C. Elkan, and Z. C. Lipton, "Predicting Surgery Duration with Neural Heteroscedastic Regression," *ICLR Work. 2017*, 2017.

[4] S. P. Devi, K. S. Rao, and S. S. Sangeetha, "Prediction of Surgery Times and Scheduling of Operation Theaters in Optholmology Department," *J. Med. Syst.*, vol. 36, no. 2, pp. 415–430, Apr. 2012.

[5] M. J. C. Eijkemans, M. van Houdenhoven, T. Nguyen, E. Boersma, E. W. Steyerberg, and G. Kazemier, "Predicting the unpredictable: a new prediction model for operating room times using individual characteristics and the surgeon's estimate.," *Anesthesiology*, vol. 112, no. 1, pp. 41–9, Jan. 2010.

[6] M. Persson and N. Lavesson, "Identification of surgery indicators by mining hospital data: A preliminary study," in *Proceedings - International Workshop on Database and Expert Systems Applications, DEXA*, 2009, pp. 323–327.

[7] E. Kostadinova, V. Boeva, and N. Lavesson, "Clustering of Multiple Microarray Experiments Using Information Integration," 2011, pp. 123–137.

[8] L. H. Aiken *et al.*, "Patient safety, satisfaction, and quality of hospital care: cross sectional surveys of nurses and patients in 12 countries in Europe and the United States," *BMJ*, vol. 344, no. mar20 2, pp. e1717–e1717, Mar. 2012.

[9] S. Yin, X. Wang, J. Wu, and G. Wang, "Grey Correlation Analysis on the Influential Factors the Hospital Medical Expenditure," 2010, pp. 73–78.

[10] A. MILLS, "The economics of hospitals in developing countries. Part I: expenditure patterns," *Health Policy Plan.*, vol. 5, no. 2, pp. 107–117, 1990.

[11] M. Persson and J. A. Persson, “Health economic modeling to support surgery management at a Swedish hospital,” *Omega*, vol. 37, no. 4, pp. 853–863, Aug. 2009.

[12] S. Kudo, E. Mutisya, and M. Nagao, “Population Aging: An Emerging Research Agenda for Sustainable Development,” *Soc. Sci.*, vol. 4, no. 4, pp. 940–966, Oct. 2015.

[13] R. Palacios, “The future of global ageing,” *Int. J. Epidemiol.*, vol. 31, no. 4, pp. 786–791, Aug. 2002.

[14] D. A. Etzioni, J. H. Liu, M. A. Maggard, and C. Y. Ko, “The Aging Population and Its Impact on the Surgery Workforce,” *Ann. Surg.*, vol. 238, no. 2, pp. 170–177, Aug. 2003.

[15] E. Kayis *et al.*, “Improving prediction of surgery duration using operational and temporal factors.,” *AMIA ... Annu. Symp. proceedings. AMIA Symp.*, vol. 2012, pp. 456–62, 2012.

[16] E. Kayış, T. T. Khaniyev, J. Suermondt, and K. Sylvester, “A robust estimation model for surgery durations with temporal, operational, and surgery team effects,” *Health Care Manag. Sci.*, vol. 18, no. 3, pp. 222–233, Sep. 2015.

[17] M. J. Persson and J. A. Persson, “Analysing management policies for operating room planning using simulation,” *Health Care Manag. Sci.*, vol. 13, no. 2, pp. 182–191, Jun. 2010.

[18] J. McCarthy, M. L. Minsky, N. Rochester, and C. E. Shannon, “A proposal for the dartmouth summer research project on artificial intelligence, august 31, 1955,” *AI Mag.*, vol. 27, no. 4, p. 12, 2006.

[19] A. M. Turing, “Computing machinery and intelligence,” *Mind*, vol. 59, no. 236, pp. 433–460, 1950.

[20] D. T. Larose and C. D. Larose, *Data Mining and Predictive Analytics*. Wiley, 2015.

[21] C. Shearer, “The CRISP-DM model: the new blueprint for data mining,” *J. data Warehous.*, vol. 5, no. 4, pp. 13–22, 2000.

[22] K. B. Korb and A. E. Nicholson, *Bayesian Artificial Intelligence*, 2nd Revise. Taylor & Francis Inc, 2011.

[23] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1988.

[24] L. C. van der Gaag and A. J. Feelders, Eds., *Probabilistic Graphical Models*, vol. 8754. Cham: Springer International Publishing, 2014.

[25] A. Holzinger, Ed., *Machine Learning for Health Informatics*, vol. 9605. Cham: Springer International Publishing, 2016.

[26] S. Goodwin, "Data rich, information poor (DRIP) syndrome: is there a treatment?," *Radiol. Manage.*, vol. 18, no. 3, pp. 45–9, 1996.

[27] Y. T. Zhang and C. C. Y. Poon, "Editorial Note on Bio, Medical, and Health Informatics," *IEEE Trans. Inf. Technol. Biomed.*, vol. 14, no. 3, pp. 543–545, May 2010.

[28] D. A. Clifton, J. Gibbons, J. Davies, and L. Tarassenko, "Machine Learning and Software Engineering in Health Informatics," in *Proceedings of the First International Workshop on Realizing AI Synergies in Software Engineering*, 2012, pp. 37–41.

[29] D. G. McQuarrie, "Limits to efficient operating room scheduling: Lessons from computer-use models," *Arch. Surg.*, vol. 116, no. 8, pp. 1065–1071, 1981.

[30] A. Macario, F. Dexter, and R. D. Traub, "Hospital profitability per hour of operating room time can vary among surgeons," *Anesth. Analg.*, vol. 93, no. 3, pp. 669–675, 2001.

[31] D. M. Hamilton and S. Breslawski, "Operating Room Scheduling," *AORN J.*, vol. 59, no. 3, pp. 665–680, Mar. 1994.

[32] N. Master, Z. Zhou, D. Miller, D. Scheinker, N. Bambos, and P. Glynn, "Improving predictions of pediatric surgical durations with supervised learning," *Int. J. Data Sci. Anal.*, May 2017.

[33] A. Macario, "What does one minute of operating room time cost?," *J. Clin. Anesth.*, vol. 22, no. 4, pp. 233–6, Jun. 2010.

[34] R. D. Shippert, "A study of time-dependent operating room fees and how to save $100 000 by using time-saving products," *Am. J. Cosmet. Surg.*, vol. 22, no. 1, pp. 25–34, 2005.

[35] F. Dexter, E. Marcon, R. H. Epstein, and J. Ledolter, "Validation of statistical methods to compare cancellation rates on the day of surgery.," *Anesth. Analg.*, vol. 101, no. 2, p. 465–73, table of contents, Aug. 2005.

[36] F. Dexter, J. T. Blake, D. H. Penning, and D. A. Lubarsky, "Calculating a potential

increase in hospital margin for elective surgery by changing operating room time allocations or increasing nursing staffing to permit completion of more cases: a case study," *Anesth. Analg.*, vol. 94, no. 1, pp. 138–142, 2002.

[37] A. Fügener, S. Schiffels, and R. Kolisch, "Overutilization and underutilization of operating rooms - insights from behavioral health care operations management," *Health Care Manag. Sci.*, vol. 20, no. 1, pp. 115–128, Mar. 2017.

[38] B. Cardoen, E. Demeulemeester, and J. Beliën, "Operating room planning and scheduling: A literature review," *Eur. J. Oper. Res.*, vol. 201, no. 3, pp. 921–932, Mar. 2010.

[39] J. M. Magerlein and J. B. Martin, "Surgical demand scheduling: a review.," *Health Serv. Res.*, vol. 13, no. 4, p. 418, 1978.

[40] J. T. Blake and M. W. Carter, "Surgical process scheduling: a structured review.," *J. Soc. Health Syst.*, vol. 5, no. 3, pp. 17–30, 1997.

[41] Z. H. Przasnyski, "Operating room scheduling: A literature review," *AORN J.*, vol. 44, no. 1, pp. 6770747882--68727679, 1986.

[42] Y. Yang, K. M. Sullivan, P. P. Wang, and K. D. Naidu, "Applications of computer simulation in medical scheduling," in *Proceedings of the joint conference on information sciences*, 2000, vol. 227.

[43] V. L. Smith-Daniels, S. B. Schweikhart, and D. E. Smith-Daniels, "Capacity management in health care services: Review and future research directions," *Decis. Sci.*, vol. 19, no. 4, pp. 889–919, 1988.

[44] D. Boldy, "A review of the application of mathematical programming to tactical and strategic health and social services problems," *J. Oper. Res. Soc.*, vol. 27, no. 2, pp. 439–448, 1976.

[45] L. F. Burgette, A. W. Mulcahy, A. Mehrotra, T. Ruder, and B. O. Wynn, "Estimating Surgical Procedure Times Using Anesthesia Billing Data and Operating Room Records," *Health Serv. Res.*, vol. 52, no. 1, pp. 74–92, Feb. 2017.

[46] F. Dexter and J. Ledolter, "Bayesian prediction bounds and comparisons of operating room times even for procedures with few or no historic data.," *Anesthesiology*, vol. 103, no. 6, pp. 1259–167, Dec. 2005.

[47] S. P. Mandel J, "A Comparison of Six Methods for Missing Data Imputation," *J. Biom. Biostat.*, vol. 6, no. 1, 2015.

[48] O. Troyanskaya *et al.*, "Missing value estimation methods for DNA microarrays.,"

Bioinformatics, vol. 17, no. 6, pp. 520–5, Jun. 2001.

[49] "Missing Data in Clinical Trials," *N. Engl. J. Med.*, vol. 367, no. 26, pp. 2557–2558, Dec. 2012.

[50] T. Schneider, "Analysis of incomplete climate data: Estimation of mean values and covariance matrices and imputation of missing values," *J. Clim.*, vol. 14, no. 5, pp. 853–871, 2001.

[51] R. J. A. Little and D. B. Rubin, *Statistical analysis with missing data*. John Wiley & Sons, 2014.

[52] D. B. Rubin, Ed., *Multiple Imputation for Nonresponse in Surveys*. Hoboken, NJ, USA: John Wiley & Sons, Inc., 1987.

[53] J. G. Ibrahim, M.-H. Chen, S. R. Lipsitz, and A. H. Herring, "Missing-Data Methods for Generalized Linear Models," *J. Am. Stat. Assoc.*, vol. 100, no. 469, pp. 332–346, Mar. 2005.

[54] *OECD Glossary of Statistical Terms*. OECD Publishing, 2008.

[55] S. Garcia, J. Luengo, J. A. Sáez, V. López, and F. Herrera, "A Survey of Discretization Techniques: Taxonomy and Empirical Analysis in Supervised Learning," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 4, pp. 734–750, Apr. 2013.

[56] S. Ramírez-Gallego *et al.*, "Data discretization: taxonomy and big data challenge," *Wiley Interdiscip. Rev. Data Min. Knowl. Discov.*, vol. 6, no. 1, pp. 5–21, Jan. 2016.

[57] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," 1993.

[58] P. Flach, *Machine Learning*. Cambridge: Cambridge University Press, 2012.

[59] A. Liaw, M. Wiener, and others, "Classification and regression by randomForest," *R news*, vol. 2, no. 3, pp. 18–22, 2002.

[60] V. Svetnik, A. Liaw, C. Tong, J. C. Culberson, R. P. Sheridan, and B. P. Feuston, "Random Forest: A Classification and Regression Tool for Compound Classification and QSAR Modeling," *J. Chem. Inf. Comput. Sci.*, vol. 43, no. 6, pp. 1947–1958, Nov. 2003.

[61] T. F. Cootes, M. C. Ionita, C. Lindner, and P. Sauer, "Robust and Accurate Shape Model Fitting Using Random Forest Regression Voting," 2012, pp. 278–291.

[62] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Stat.*

Comput., vol. 14, no. 3, pp. 199–222, 2004.

[63] M. Martin, "On-line support vector machine regression," in *ECML*, 2002, vol. 2, pp. 282–294.

[64] Z. Zhang, "Introduction to machine learning: k-nearest neighbors," *Ann. Transl. Med.*, vol. 4, no. 11, pp. 218–218, Jun. 2016.

[65] K. Q. Weinberger, J. Blitzer, and L. K. Saul, "Distance metric learning for large margin nearest neighbor classification," in *Advances in neural information processing systems*, 2006, pp. 1473–1480.

[66] L. E. Peterson, "K-nearest neighbor," *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009.

[67] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *IEEE Trans. Syst. Man. Cybern.*, no. 4, pp. 580–585, 1985.

[68] D. L. Shrestha and D. P. Solomatine, "Experiments with AdaBoost.RT, an Improved Boosting Scheme for Regression," *Neural Comput.*, vol. 18, no. 7, pp. 1678–1710, Jul. 2006.

[69] W. Hu, W. Hu, and S. Maybank, "Adaboost-based algorithm for network intrusion detection," *IEEE Trans. Syst. Man, Cybern. Part B*, vol. 38, no. 2, pp. 577–583, 2008.

[70] H. Drucker, "Improving regressors using boosting techniques," in *ICML*, 1997, vol. 97, pp. 107–115.

[71] M. Collins, R. E. Schapire, and Y. Singer, "Logistic regression, AdaBoost and Bregman distances," *Mach. Learn.*, vol. 48, no. 1, pp. 253–285, 2002.

[72] F. V. Jensen and T. D. Nielsen, *Bayesian Networks and Decision Graphs*. New York, NY: Springer New York, 2007.

[73] D. Heckerman, "Bayesian networks for data mining," *Data Min. Knowl. Discov.*, vol. 1, no. 1, pp. 79–119, 1997.

[74] R. Nagarajan, M. Scutari, and S. L?bre, *Bayesian Networks in R*. New York, NY: Springer New York, 2013.

[75] R. R. Bouckaert, "Bayesian belief networks: from construction to inference," Utrecht University, The Netherlands, 1995.

[76] J. A. Gamez, J. L. Mateo, and J. M. Puerta, "Learning Bayesian networks by hill climbing: efficient methods based on progressive restriction of the neighborhood,"

Data Min. Knowl. Discov., vol. 22, no. 1–2, pp. 106–148, Jan. 2011.

[77] H. Zhang, "The optimality of naive Bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[78] D. J. Hand and K. Yu, "Idiot's Bayes—not so stupid after all?," *Int. Stat. Rev.*, vol. 69, no. 3, pp. 385–398, 2001.

[79] P. N. Bennett, "Assessing the calibration of Naive Bayes posterior estimates," 2000.

[80] P. Domingos and M. Pazzani, "Beyond independence: Conditions for the optimality of the simple bayesian classi er," in *Proc. 13th Intl. Conf. Machine Learning*, 1996, pp. 105–112.

APPENDICES

APPENDIX A

PARTIAL CORRELATION HEAT MAPS

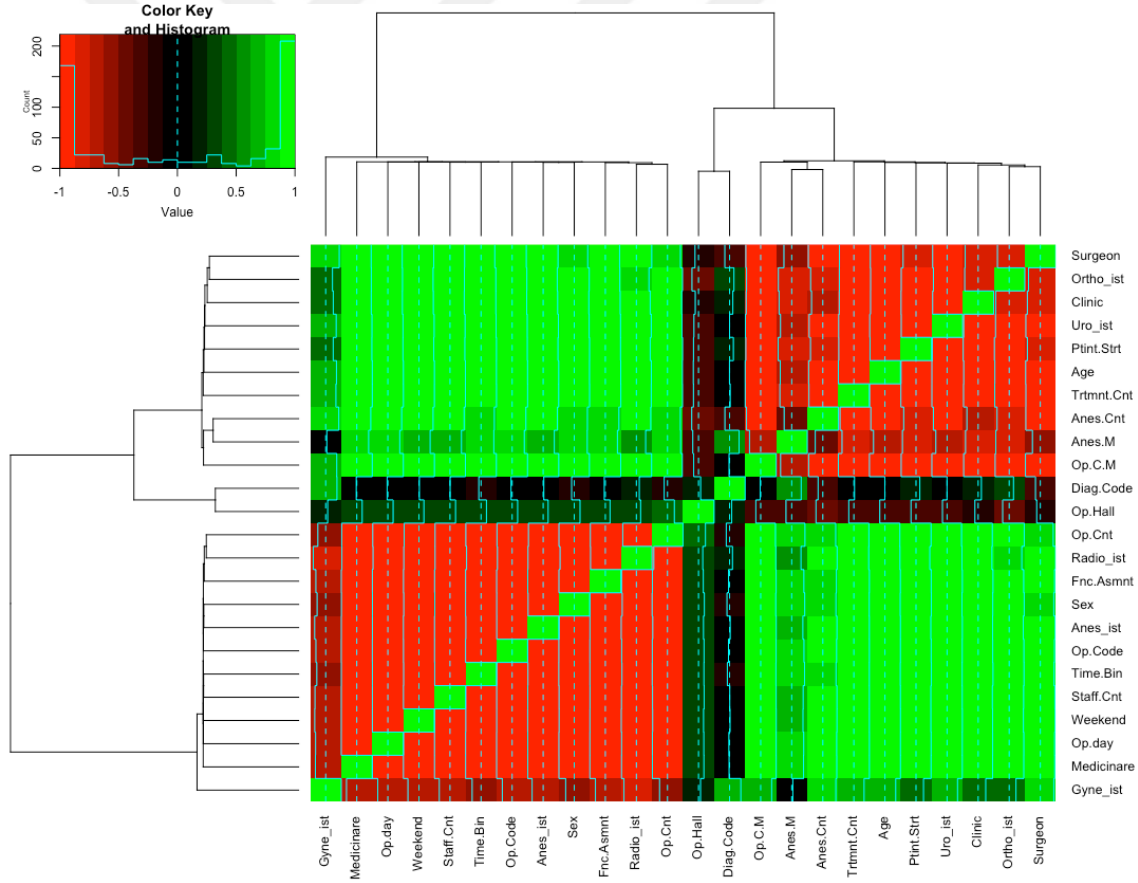

Figure 25: The Heat Map Created Based on Pearson's Estimation of Partial Correlations

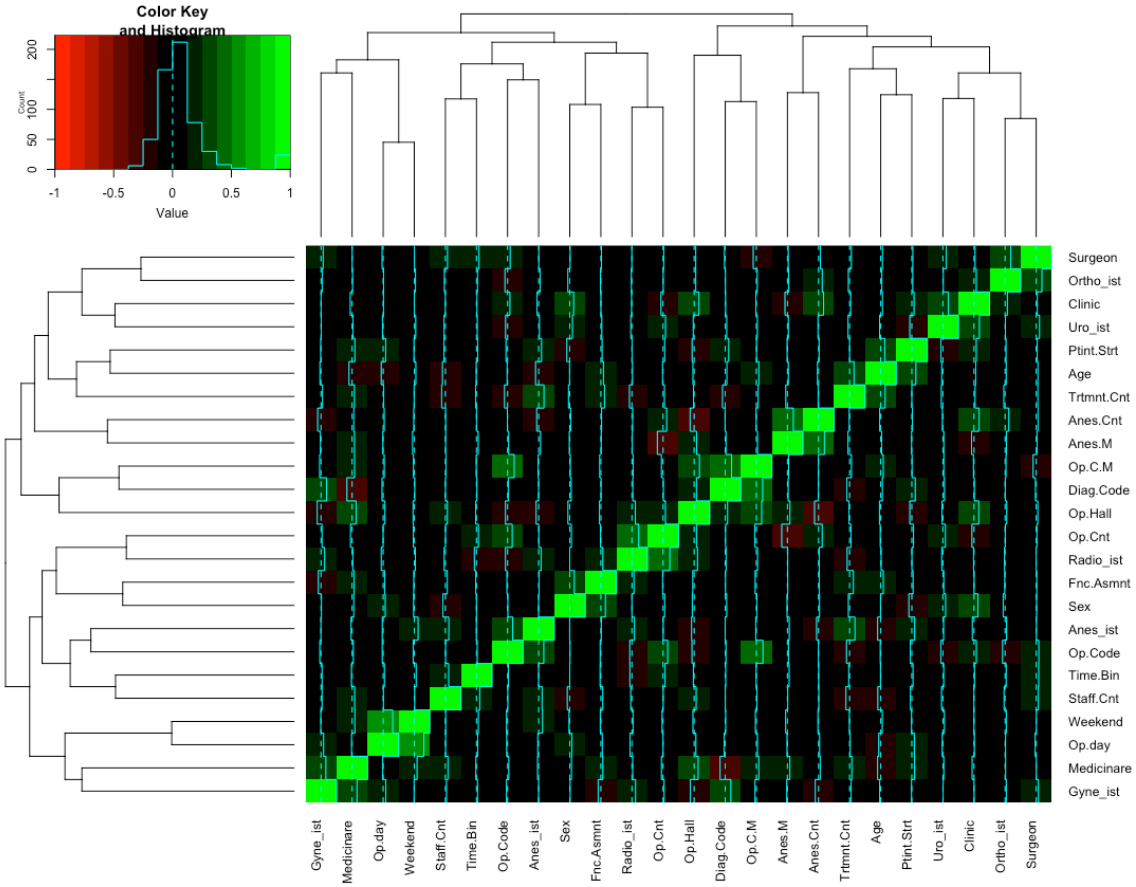

Figure 26: The Heat Map Created Based on Kendall's Estimation of Partial Correlations

APPENDIX B

DATA DISTRIBUTION CHARTS

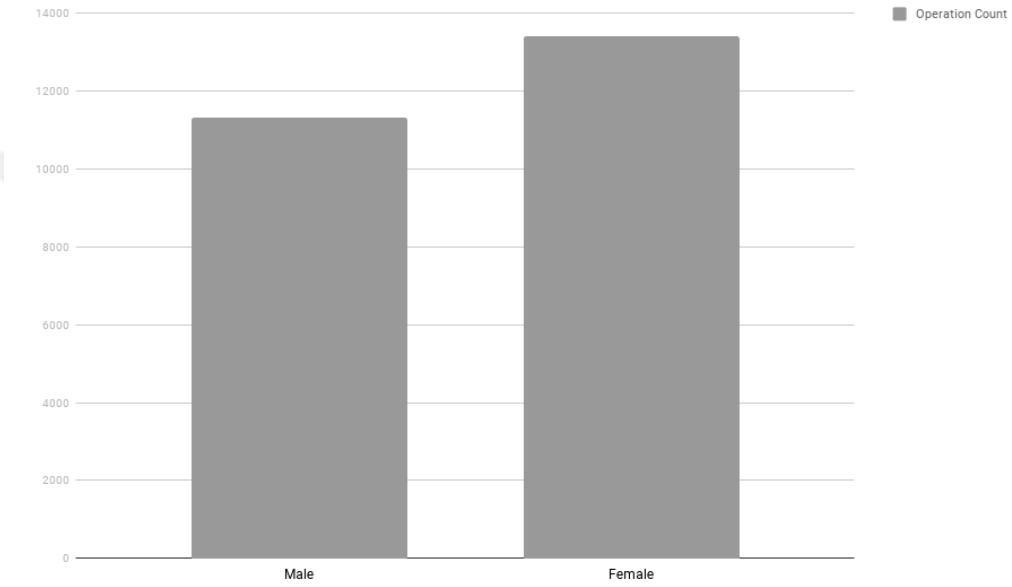

Figure 27: The Distribution of Gender Categories

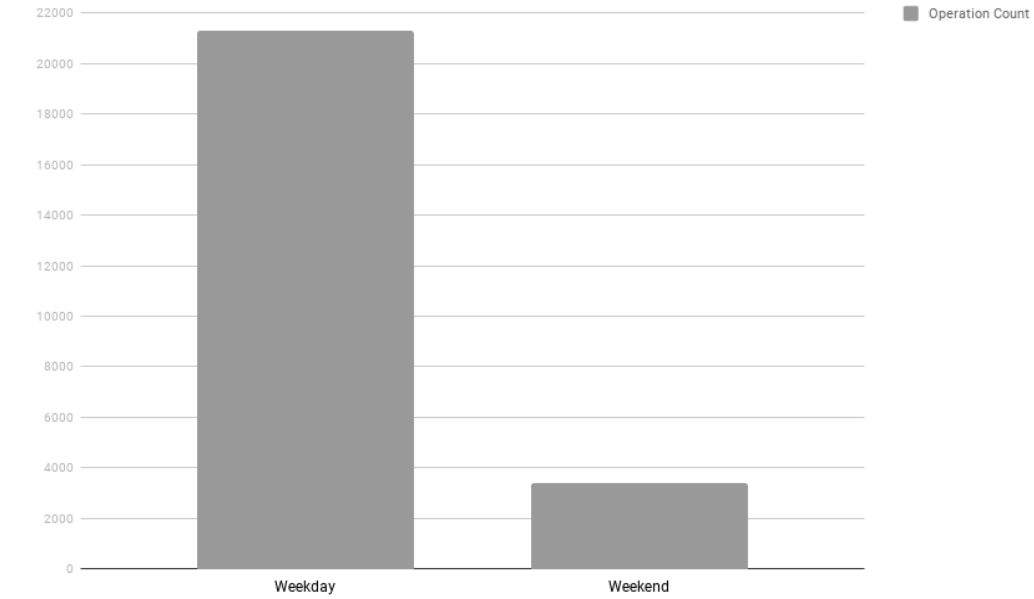

Figure 28: The Distribution of Weekend Category

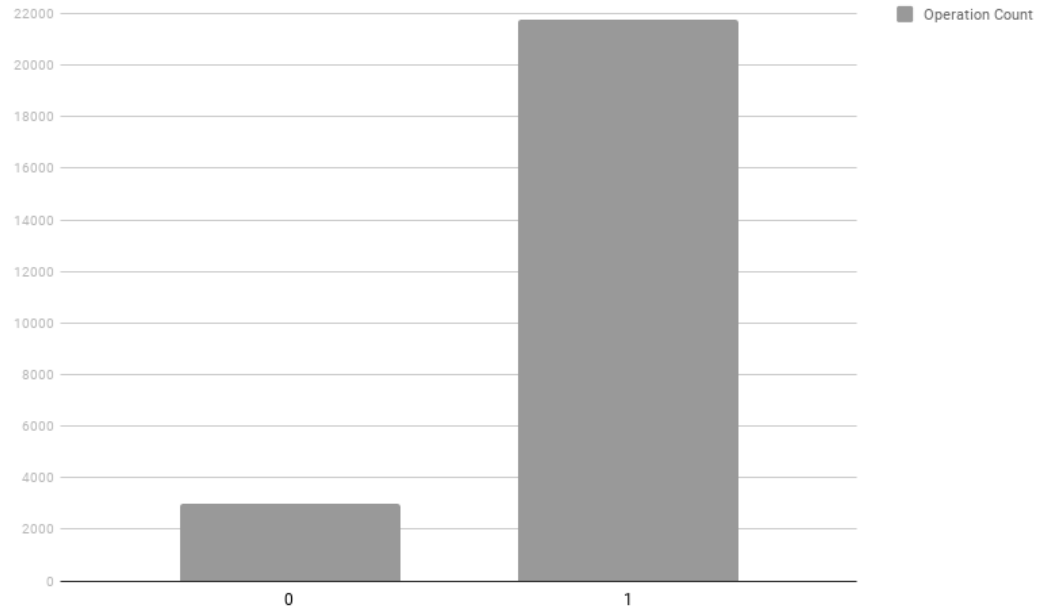

Figure 29: The Distribution of Anesthesiologist

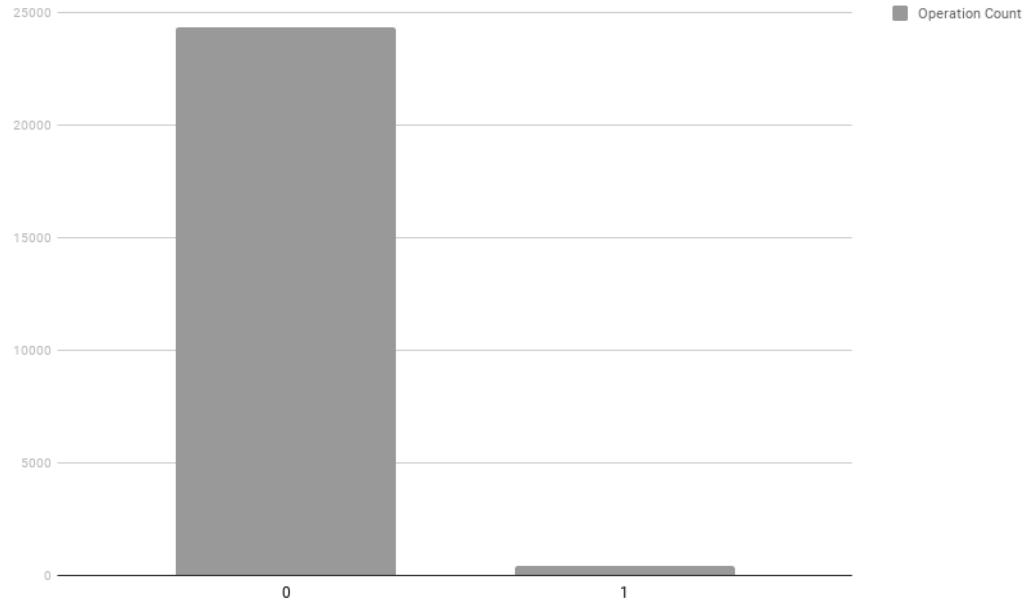

Figure 30: The Distribution of Gynecologist

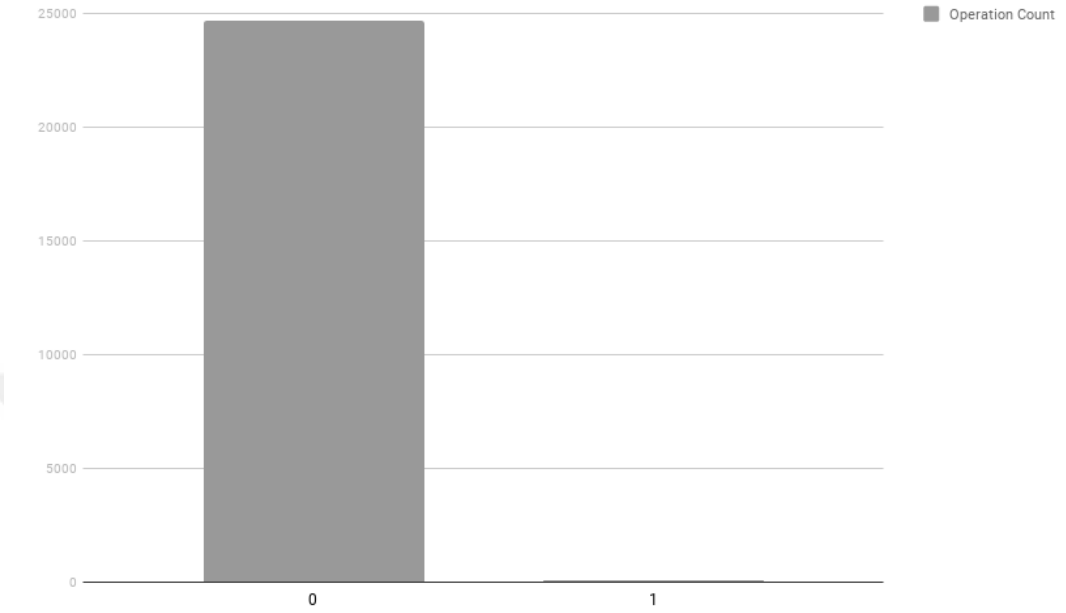

Figure 31: The Distribution of Medical Students

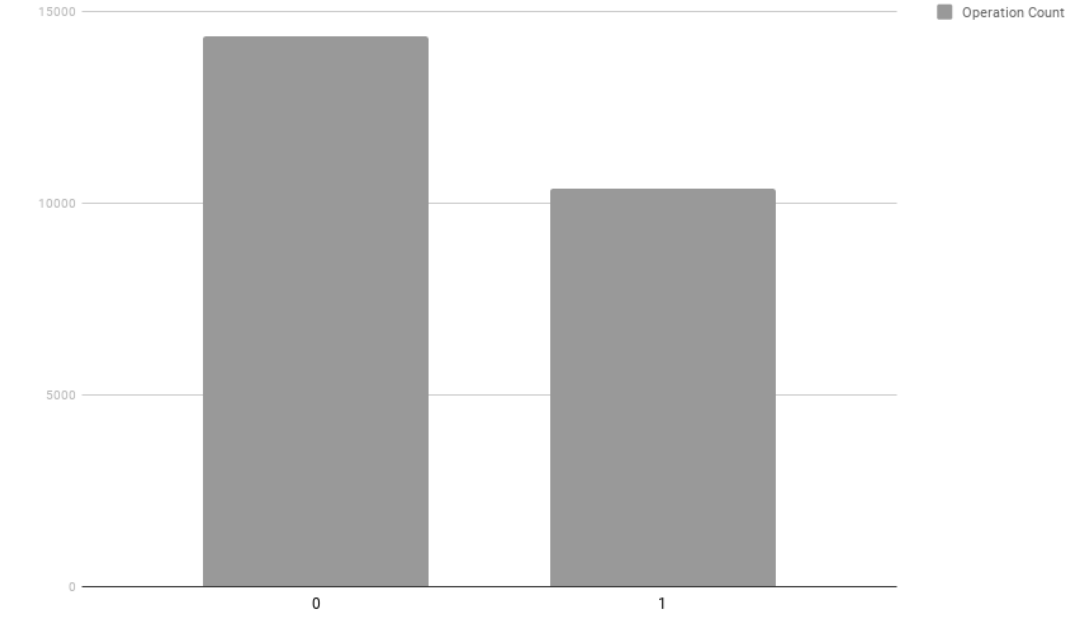

Figure 32: The Distribution of Orthopedist

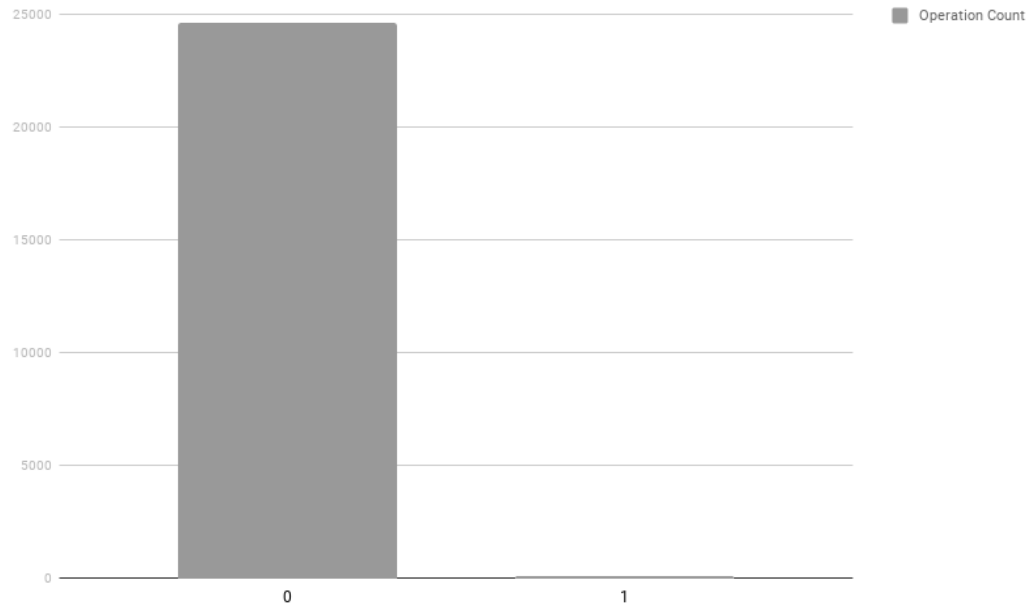

Figure 33: The Distribution of Radiologist

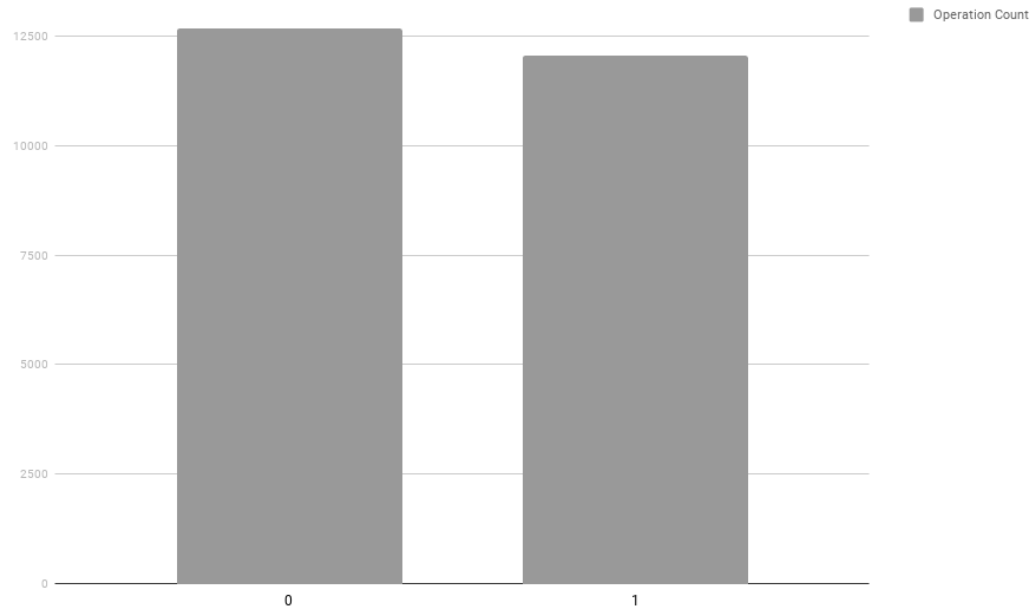

Figure 34: The Distribution of General Surgeon

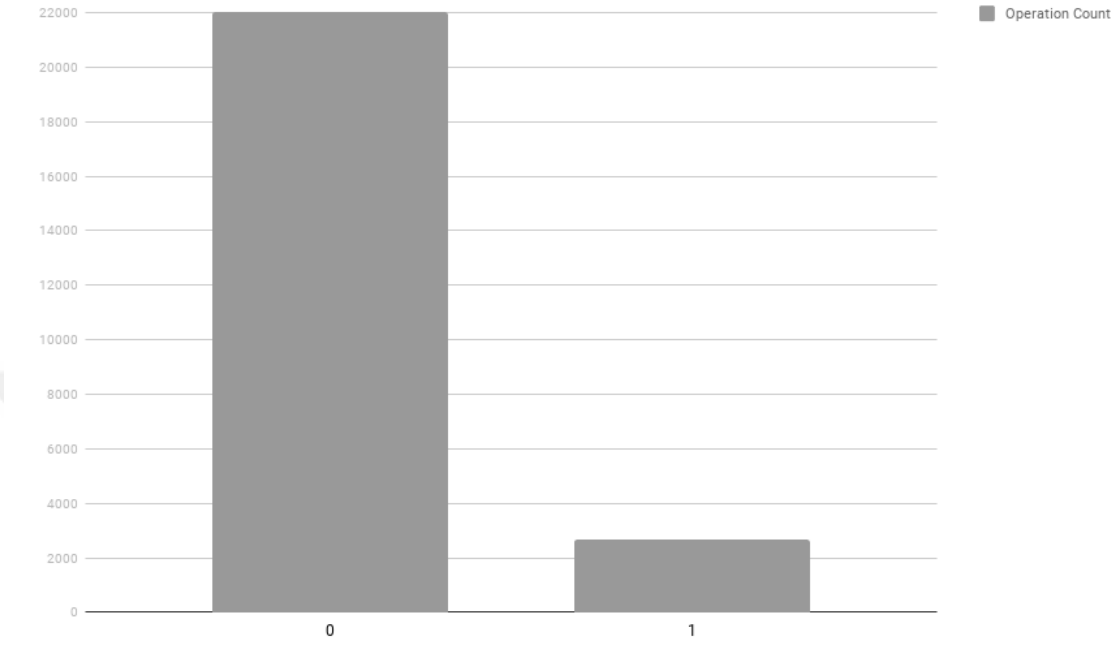

Figure 35: The Distribution of Urologist

APPENDIX C

SOURCE CODES

FILE CLEANING SCRIPT:

```
# The main script
# Reads the raw data and does the cleaning process


from settings import INPUT_FILE_NAME, OUTPUT_FILE_NAME, FILE_HEADER_V2
import openpyxl
import csv
from libs.Patient import Patient

wb = openpyxl.load_workbook(INPUT_FILE_NAME, read_only=True, data_only=True,
guess_types=False)
ws = wb.active
iterator = ws.iter_rows()
next(iterator)

tn_temp = None
# count = 0
flag = False

with open(OUTPUT_FILE_NAME, 'w') as result_file:
    wr = csv.writer(result_file, dialect='excel')
    wr.writerow(FILE_HEADER_V2)
    for row in iterator:
        if not (not (row[0].value is None) and not (row[2].value is None) and
not (row[3].value is None) and not (
                row[6].value is None)) or row[7].value is None or row[8].value
is None or row[9].value is None:
            continue
        if row[0].value != tn_temp:
            if flag:
                result = patient.render_data()
                wr.writerow(result)
            flag = False
            # assigning the first treatment number (tn)
            tn_temp = row[0].value
            patient = Patient(row[0].value)
            patient.set_patient_start(row[2].value)
            patient.set_operation_start(row[7].value)
            patient.set_operation_end(row[8].value)
            patient.set_sex(row[10].value)
            patient.set_age(row[11].value)
            patient.set_functioning_assessment(row[12].value)
            patient.set_operation_hall(row[13].value)
            patient.set_operation_card_method(row[16].value)
            patient.set_diagnosis_code(row[17].value)
            patient.set_clinic(row[19].value)
            patient.set_treatment_count(row[22].value)
```

```
            patient.set_operation_date(row[14].value)
            patient.set_anesthesia_method(row[15].value)
            patient.set_operation_code(row[18].value)
            patient.set_personal_id(row[20].value)
            patient.set_personal_category(row[21].value)
        elif row[0].value == tn_temp:
            patient.set_anesthesia_method(row[15].value)
            patient.set_operation_code(row[18].value)
            patient.set_personal_id(row[20].value)
            patient.set_personal_category(row[21].value)
            flag = True
        # if count > 100:
        #     break
        # count += 1
result_file.close()
```

PATIENT CLASS:

```
# Class: Patient
#
# Usage: reading, cleaning, merging, and coding the data


class Patient:

    def __init__(self, treatment_number):
        self._treatment_number = treatment_number
        self._patient_start = None
        self._operation_start = None
        self._operation_end = None
        self._sex = None
        self._age = None
        self._functioning_assessment = None
        self._operation_hall = None
        self._operation_card_method = None
        self._diagnosis_code = None
        self._clinic = None
        self._treatment_count = None
        self._operation_date = None
        self._operation_duration = None
        self._personal_id = list()
        self._personal_category = list()
        self._anesthesia_method = list()
        self._operation_code = list()
        self._patient = []
        self._kirug = 0
        self._anestesiolog = 0
        self._ortoped = 0
        self._urolog = 0
        self._gynekolog = 0
        self._medicinare = 0
        self._radiolog = 0
        self._staff_count = 0
        self._call_service = None
        self._anesthesia_start = None
        self._blockade_start = None
        self._anesthesia_clear = None
        self._clear_for_operation = None
        self._anesthesia_end = None
```

```
# setter and getter functions to initialize the values

def set_call_service(self, data):
    self._call_service = data

def set_anesthesia_start(self, data):
    self._anesthesia_start = data

def set_blockade_start(self, data):
    self._blockade_start = data

def set_anesthesia_clear(self, data):
    self._anesthesia_clear = data

def set_clear_for_operation(self, data):
    self._clear_for_operation = data

def set_anesthesia_end(self, data):
    self._anesthesia_end = data

def set_patient_start(self, data):
    self._patient_start = data

def set_operation_start(self, data):
    self._operation_start = data

def set_operation_end(self, date):
    self._operation_end = date

def set_sex(self, data):
    self._sex = data

def set_age(self, data):
    self._age = data

def set_functioning_assessment(self, data):
    self._functioning_assessment = data

def set_operation_hall(self, data):
    self._operation_hall = data

def set_operation_card_method(self, data):
    self._operation_card_method = data

def set_diagnosis_code(self, data):
    self._diagnosis_code = data

def set_clinic(self, data):
    self._clinic = data

def set_treatment_count(self, data):
    self._treatment_count = data

def set_operation_date(self, data):
    self._operation_date = data

def set_personal_id(self, data):
    self._personal_id.append(data)
```

```
    def set_personal_id_m(self, data):
        if data not in self._personal_id:
            self._personal_id.append(data)
            return True
        return False

    def _get_personal_id(self):
        return ' '.join(str(self._personal_id))

    def set_personal_category(self, data):
        self._personal_category.append(data)

    def set_personal_category_m(self, data):
        if data not in self._personal_category:
            self._personal_category.append(data)
            return True
        return False

    def _get_personal_category(self):
        return ' '.join(self._personal_category)

    def set_anesthesia_method(self, data):
        if data not in self._anesthesia_method:
            self._anesthesia_method.append(data)
            return True
        return False

    def _get_anesthesia_method(self):
        return ' '.join(self._anesthesia_method)

    def set_operation_code(self, data):
        if data not in self._operation_code:
            self._operation_code.append(data)
            return True
        return False

    def _get_operation_code(self):
        return ' '.join(self._operation_code)

    def _render_personal_categories(self, personal_id, personal_category):
        """
        This function counts the number of the staff for each category included
in every surgery.
        :param personal_id:
        :param personal_category:
        :return:
        """
        pid = personal_id
        pcat = personal_category
        pid_temp = list()
        for i in range(0, len(pid)):
            if pid[i] not in pid_temp:
                pid_temp.append(pid[i])
                self._staff_count += 1
                if pcat[i] == 'Kirurg':
                    self._kirug += 1
                elif pcat[i] == 'Anestesiolog':
                    self._anestesiolog += 1
                elif pcat[i] == 'Ortoped':
                    self._ortoped += 1
```

```
                elif pcat[i] == 'Urolog':
                    self._urolog += 1
                elif pcat[i] == 'Gynekolog':
                    self._gynekolog += 1
                elif pcat[i] == 'Medicinare':
                    self._medicinare += 1
                elif pcat[i] == 'Radiolog':
                    self._radiolog += 1

    def render_data(self):
        """
        This function puts all the cleaned and merged data in to a list in
order to be written in a csv file
        :return:
        """
        operation_duration = self._operation_end - self._operation_start
        operation_weekday = self._operation_date.weekday()
        if operation_weekday > 4:
            is_weekend = 1
        else:
            is_weekend = 0
        self._render_personal_categories(self._personal_id,
self._personal_category)
        self._patient.append(self._treatment_number)
        self._patient.append(operation_duration.total_seconds())

        # self._patient.append(round(operation_duration.total_seconds() / 180))
        # self._patient.append(round(operation_duration.total_seconds() / 300))
        # self._patient.append(round(operation_duration.total_seconds() / 600))

        # 4 categories for starting hours:
        #   [0, 7) = 0
        #   [7, 12) = 1
        #   [12, 18) = 2
        #   [18, 24) = 3
        if 0 <= self._patient_start.hour < 7:
            self._patient.append("0")
        elif 7 <= self._patient_start.hour < 12:
            self._patient.append("1")
        elif 12 <= self._patient_start.hour < 18:
            self._patient.append("2")
        elif 18 <= self._patient_start.hour < 24:
            self._patient.append("3")

        self._patient.append(self._sex)

        # 4 categories for age:
        #   [0, 20) = 0
        #   [20, 40) = 1
        #   [40, 60) = 2
        #   [60, inf) = 3
        if 0 <= self._age < 20:
            self._patient.append("0")
        elif 20 <= self._age < 40:
            self._patient.append("1")
        elif 40 <= self._age < 60:
            self._patient.append("2")
        elif 60 <= self._age:
            self._patient.append("3")
```

```
        self._patient.append(self._functioning_assessment)
        self._patient.append(self._operation_hall)
        self._patient.append(self._operation_card_method)
        self._patient.append(self._diagnosis_code)
        self._patient.append(self._clinic)
        self._patient.append(self._treatment_count)
        self._patient.append(operation_weekday)
        self._patient.append(is_weekend)
        self._patient.append(self._get_anesthesia_method())
        self._patient.append(len(self._anesthesia_method))
        self._patient.append(self._get_operation_code())
        self._patient.append(len(self._operation_code))

        # boolean values from kirurg to radiolog
        # 0 = not in the surgery
        # 1 = in the surgery
        if self._kirug == 0:
            self._patient.append("0")
        elif self._kirug > 0:
            self._patient.append("1")

        if self._anestesiolog == 0:
            self._patient.append("0")
        elif self._anestesiolog > 0:
            self._patient.append("1")

        if self._ortoped == 0:
            self._patient.append("0")
        elif self._ortoped > 0:
            self._patient.append("1")

        if self._urolog == 0:
            self._patient.append("0")
        elif self._urolog > 0:
            self._patient.append("1")

        if self._gynekolog == 0:
            self._patient.append("0")
        elif self._gynekolog > 0:
            self._patient.append("1")

        if self._medicinare == 0:
            self._patient.append("0")
        elif self._medicinare > 0:
            self._patient.append("1")

        if self._radiolog == 0:
            self._patient.append("0")
        elif self._radiolog > 0:
            self._patient.append("1")

        self._patient.append(self._staff_count)
        return self._patient

    def render_data_marie(self):
        """
        WARNING: Do NOT use this function in normal cases. It's only for
special circumstances where specific data
        is needed
        :return:
```

```
        """
        self._patient.append(self._treatment_number)
        self._patient.append(self._call_service)
        self._patient.append(self._patient_start)
        self._patient.append(self._anesthesia_start)
        self._patient.append(self._blockade_start)
        self._patient.append(self._anesthesia_clear)
        self._patient.append(self._clear_for_operation)
        self._patient.append(self._operation_start)
        self._patient.append(self._operation_end)
        self._patient.append(self._anesthesia_end)
        self._patient.append(self._sex)
        self._patient.append(self._age)
        self._patient.append(self._functioning_assessment)
        self._patient.append(self._operation_hall)
        self._patient.append(self._operation_date)
        self._patient.append(self._get_anesthesia_method())
        self._patient.append(self._operation_card_method)
        self._patient.append(self._diagnosis_code)
        self._patient.append(self._get_operation_code())
        self._patient.append(self._clinic)
        self._patient.append(self._get_personal_id())
        self._patient.append(self._get_personal_category())
        self._patient.append(self._treatment_count)
        return self._patient
```

BINNING SURGICAL OPERATION DURATION SCRIPT:

```
# This script reads the data from the source file and does the binning process
for surgery times
# assigning different bin codes for each surgery depending on the duration it
took
#
# The following methods of binning are used: Equal Width Binning and Equal
Frequency Binning


import csv

with open('data/result_operation_duration_sorted_V2.csv', 'r', encoding="ISO-
8859-1") as input_file:
    with open('data/result_operation_duration_sorted_mixed_bins_3000i_V2.csv',
'w') as output_file:
        writer = csv.writer(output_file, dialect='excel')
        reader = csv.reader(input_file)

        header = next(reader)
        header.append('bin')
        writer.writerow(header)
        counter = 0
        last_bin = 0
        start_point = 6900
        for row in reader:
            # equal width binning
            if counter < 20996:
                row.append(int(counter/3000))
                writer.writerow(row)
                last_bin = int(counter/3000)
            # equal frequency binning
            else:
```

```
                if int(row[1]) > start_point + 1800:
                    last_bin += 1
                    start_point = int(row[1])
                row.append(last_bin + 1)
                writer.writerow(row)
            counter += 1
output_file.close()
```

CALCULATING ACCURACY SCRIPT:

```
import csv

with open('orange/predictions.csv', 'r', encoding="ISO-8859-1") as input_file:
    iterator = csv.reader(input_file)
    next(iterator)
    next(iterator)
    next(iterator)
    counter = 0
    hit = 0
    threshold = 15

    # indexes:
    # 23 -> actual bin
    # 25 -> operation duration (seconds)
    # 26 -> Naive Bayes Predicted Bin
    # 27 -> Logistic Regression Predicted Bin

    for row in iterator:
        counter += 1
        if row[26] == "0":
            if abs(8 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "1":
            if abs(23.5 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "2":
            if abs(36.5 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "3":
            if abs(49 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "4":
            if abs(62 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "5":
            if abs(78 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "6":
            if abs(101 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "7":
            if abs(157.5 - (float(row[25]) / 60)) < threshold:
                hit += 1
        elif row[26] == "8":
            if abs(381.5 - (float(row[25]) / 60)) < threshold:
                hit += 1
    print(hit / counter)
```

CALCULATING RMSE SCRIPT:

```
import csv
import math

with open('orange/prediction_mix_bin.csv', 'r', encoding="ISO-8859-1") as
input_file:
    iterator = csv.reader(input_file)
    next(iterator)
    next(iterator)
    next(iterator)
    counter = 0
    MSE = 0

    # indexes:
    # 23 -> actual bin
    # 25 -> operation duration (seconds)
    # 26 -> Naive Bayes Predicted Bin

    for row in iterator:
        counter += 1
        if str(row[26]) == "0.0":
            MSE += ((float(row[25]) / 60) - 8) * ((float(row[25]) / 60) - 8)
        elif row[26] == "1.0":
            MSE += ((float(row[25]) / 60) - 23.5) * ((float(row[25]) / 60) -
23.5)
        elif row[26] == "2.0":
            MSE += ((float(row[25]) / 60) - 36.5) * ((float(row[25]) / 60) -
36.5)
        elif row[26] == "3.0":
            MSE += ((float(row[25]) / 60) - 49) * ((float(row[25]) / 60) - 49)
        elif row[26] == "4.0":
            MSE += ((float(row[25]) / 60) - 62) * ((float(row[25]) / 60) - 62)
        elif row[26] == "5.0":
            MSE += ((float(row[25]) / 60) - 78) * ((float(row[25]) / 60) - 78)
        elif row[26] == "6.0":
            MSE += ((float(row[25]) / 60) - 100.5) * ((float(row[25]) / 60) -
100.5)
        elif row[26] == "7.0":
            MSE += ((float(row[25]) / 60) - 130) * ((float(row[25]) / 60) -
130)
        elif row[26] == "8.0":
            MSE += ((float(row[25]) / 60) - 161) * ((float(row[25]) / 60) -
161)
        elif row[26] == "9.0":
            MSE += ((float(row[25]) / 60) - 192) * ((float(row[25]) / 60) -
192)
        elif row[26] == "10.0":
            MSE += ((float(row[25]) / 60) - 223) * ((float(row[25]) / 60) -
223)
        elif row[26] == "11.0":
            MSE += ((float(row[25]) / 60) - 254) * ((float(row[25]) / 60) -
254)
        elif row[26] == "12.0":
            MSE += ((float(row[25]) / 60) - 285) * ((float(row[25]) / 60) -
285)
        elif row[26] == "13.0":
            MSE += ((float(row[25]) / 60) - 316) * ((float(row[25]) / 60) -
316)
        elif row[26] == "14.0":
```

```
            MSE += ((float(row[25]) / 60) - 348.5) * ((float(row[25]) / 60) -
348.5)
        elif row[26] == "15.0":
            MSE += ((float(row[25]) / 60) - 380.5) * ((float(row[25]) / 60) -
380.5)
        elif row[26] == "16.0":
            MSE += ((float(row[25]) / 60) - 413) * ((float(row[25]) / 60) -
413)
        elif row[26] == "17.0":
            MSE += ((float(row[25]) / 60) - 452.5) * ((float(row[25]) / 60) -
452.5)
        elif row[26] == "18.0":
            MSE += ((float(row[25]) / 60) - 500.5) * ((float(row[25]) / 60) -
500.5)
        elif row[26] == "19.0":
            MSE += ((float(row[25]) / 60) - 539) * ((float(row[25]) / 60) -
539)
        elif row[26] == "20.0":
            MSE += ((float(row[25]) / 60) - 557) * ((float(row[25]) / 60) -
557)
    print(math.sqrt(MSE / counter))
```

CALCULATING MAE SCRIPT:

```
import csv

with open('orange/prediction_mix_bin.csv', 'r', encoding="ISO-8859-1") as
input_file:
    iterator = csv.reader(input_file)
    next(iterator)
    next(iterator)
    next(iterator)
    counter = 0
    AE = 0

    # indexes:
    # 23 -> actual bin
    # 25 -> operation duration (seconds)
    # 26 -> Naive Bayes Predicted Bin

    for row in iterator:
        counter += 1
        if row[26] == "0.0":
            AE += abs((float(row[25]) / 60) - 8)
        elif row[26] == "1.0":
            AE += abs((float(row[25]) / 60) - 23.5)
        elif row[26] == "2.0":
            AE += abs((float(row[25]) / 60) - 36.5)
        elif row[26] == "3.0":
            AE += abs((float(row[25]) / 60) - 49)
        elif row[26] == "4.0":
            AE += abs((float(row[25]) / 60) - 62)
        elif row[26] == "5.0":
            AE += abs((float(row[25]) / 60) - 78)
        elif row[26] == "6.0":
            AE += abs((float(row[25]) / 60) - 100.5)
        elif row[26] == "7.0":
            AE += abs((float(row[25]) / 60) - 130)
        elif row[26] == "8.0":
```

```
            AE += abs((float(row[25]) / 60) - 161)
        elif row[26] == "9.0":
            AE += abs((float(row[25]) / 60) - 192)
        elif row[26] == "10.0":
            AE += abs((float(row[25]) / 60) - 223)
        elif row[26] == "11.0":
            AE += abs((float(row[25]) / 60) - 254)
        elif row[26] == "12.0":
            AE += abs((float(row[25]) / 60) - 285)
        elif row[26] == "13.0":
            AE += abs((float(row[25]) / 60) - 316)
        elif row[26] == "14.0":
            AE += abs((float(row[25]) / 60) - 348.5)
        elif row[26] == "15.0":
            AE += abs((float(row[25]) / 60) - 380.5)
        elif row[26] == "16.0":
            AE += abs((float(row[25]) / 60) - 413)
        elif row[26] == "17.0":
            AE += abs((float(row[25]) / 60) - 452.5)
        elif row[26] == "18.0":
            AE += abs((float(row[25]) / 60) - 500.5)
        elif row[26] == "19.0":
            AE += abs((float(row[25]) / 60) - 539)
        elif row[26] == "20.0":
            AE += abs((float(row[25]) / 60) - 557)
    print(AE / counter)
```

COLLECT ANESTHETIC METHODS SCRIPT:

```
# collecting all the available anesthesia methods in the data set


from settings import INPUT_FILE_NAME, ANESTHETIC_METHODS
import openpyxl

wb = openpyxl.load_workbook(INPUT_FILE_NAME, read_only=True, data_only=True,
guess_types=False)
ws = wb.active
iterator = ws.iter_rows()
next(iterator)
anesthetic_methods = list()
for row in iterator:
    if row[15].value not in anesthetic_methods:
        anesthetic_methods.append(row[15].value)
with open(ANESTHETIC_METHODS, 'w') as file:
    file.write('\n'.join(anesthetic_methods))
```

COLLECT OPERATION CODES SCRIPT:

```
# collecting all the available operation codes in the data set


from settings import INPUT_FILE_NAME, OPERATION_CODES
import openpyxl

wb = openpyxl.load_workbook(INPUT_FILE_NAME, read_only=True, data_only=True,
guess_types=False)
ws = wb.active
iterator = ws.iter_rows()
```

```
next(iterator)
operation_codes = list()
for row in iterator:
    if row[18].value not in operation_codes:
        operation_codes.append(row[18].value)
with open(OPERATION_CODES, 'w') as file:
    file.write('\n'.join(operation_codes))
```

COLLECT PERSONNEL CATEGORIES SCRIPT:

```
# collecting all the available personal categories in the data set


from settings import INPUT_FILE_NAME, PERSONAL_CATEGORIES
import openpyxl

wb = openpyxl.load_workbook(INPUT_FILE_NAME, read_only=True, data_only=True,
guess_types=False)
ws = wb.active
iterator = ws.iter_rows()
next(iterator)
categories = list()
for row in iterator:
    if row[21].value not in categories:
        categories.append(row[21].value)
with open(PERSONAL_CATEGORIES, 'w') as file:
    file.write('\n'.join(categories))
```

CALCULATING THE PAIRWISE CORRELATION OF FEATURES SCRIPT:

```
# calculating the pairwise correlation of features using Chi-Square Test
# Cramer's V for comparing the magnitude of effect
# creating the correlation matrix

library("lsr")
dataset <-
read.csv("~/Documents/GitProjects/DV2542/4/R/result_operation_duration_sorted_m
ixed_bins_3000i.csv", header = TRUE)
dataset$treatment_number <- NULL
dataset$time_bin_3m <- NULL
dataset$time_bin_5m <- NULL
dataset$time_bin_10m <- NULL
dataset$operation_duration <- NULL
chi_sq = matrix(0, 24, 24)
chi_p_value = matrix(0, 24, 24)
cramersv = matrix(0, 24, 24)
for (i in 1:ncol(chi_sq)){
  for (j in 1:ncol(chi_sq)){
    output = chisq.test(dataset[[i]], dataset[[j]], simulate.p.value = TRUE)
    chi_sq[i, j] = output$statistic
    chi_p_value[i, j] = output$p.value
    if (output$p.value < 0.05) {
      cramersv[i, j] = cramersV(dataset[[i]], dataset[[j]])
    }
    else {
      cramersv[i, j] = 0
    }
  }
}
```

```
write.table(chi_p_value, file =
"~/Documents/GitProjects/DV2542/4/R/correlation_matrix/chi_p_value.csv", sep =
",", row.names = FALSE, col.names = FALSE)
write.table(chi_sq, file =
"~/Documents/GitProjects/DV2542/4/R/correlation_matrix/chi_sq.csv", sep = ",",
row.names = FALSE, col.names = FALSE)
write.table(cramersv, file =
"~/Documents/GitProjects/DV2542/4/R/correlation_matrix/cramersv.csv", sep =
",", row.names = FALSE, col.names = FALSE)
```

CALCULATING PARTIAL CORRELATION OF FEATURES SCRIPT:

```
# calculating partial correlation of the features based on the values obtained
from Chi-Square Test
# using Pearson, Spearman, and Kendall estimates
# generating the heatmap to visualize the clusters

library("ppcor")
library("gplots")

feature_names <- c("Ptint Strt", "Sex", "Age", "Fnc Asmnt", "Op Hall", "Op C
M", "Diag Code", "Clinic", "Trtmnt Cnt", "Op day", "Weekend", "Anes M", "Anes
Cnt", "Op Code", "Op Cnt", "Surgeon", "Anes_ist", "Ortho_ist", "Uro_ist",
"Gyne_ist", "Medicinare", "Radio_ist", "Staff Cnt", "Time Bin")
dataset <-
read.csv("~/Documents/GitProjects/DV2542/4/R/correlation_matrix/cramersv.csv",
header = FALSE, col.names = feature_names, row.names = feature_names)

kendall_output <- pcor(dataset, method = "kendall")
kendall_estimates <- kendall_output$estimate

pearson_output <- pcor(dataset, method = "pearson")
pearson_estimates <- pearson_output$estimate

spearman_output <- pcor(dataset, method = "spearman")
spearman_estimates <- spearman_output$estimate

#heatmap.2(kendall_estimates, col = redgreen(16))
#heatmap.2(pearson_estimates, col = redgreen(16))
heatmap.2(spearman_estimates, col = redgreen(16))

#write.table(pearson_estimates, file =
"~/Documents/GitProjects/DV2542/4/R/partial_correlation/pearson_estimates.csv",
sep = ",", row.names = FALSE, col.names = FALSE)
#write.table(spearman_estimates, file =
"~/Documents/GitProjects/DV2542/4/R/partial_correlation/spearman_estimates.csv"
, sep = ",", row.names = FALSE, col.names = FALSE)
#write.table(kendall_estimates, file =
"~/Documents/GitProjects/DV2542/4/R/partial_correlation/kendall_estimates.csv",
sep = ",", row.names = FALSE, col.names = FALSE)
```

NAÏVE BAYES MODEL SCRIPT:

```
library("bnlearn")
library("MASS")
dataset <-
read.csv("~/Documents/GitProjects/DV2542/4/R/result_operation_duration_sorted_m
ixed_bins_3000i_V2.csv", header = TRUE)

# removing the unnecessary columns
```

```
dataset$treatment_number <- NULL
# dataset$operation_duration <- NULL

for (i in 1:ncol(dataset)){
 dataset[[i]] <- as.factor(dataset[[i]])
}

randomSample = function(df,n) {
  return (df[sample(nrow(df), n),])
}

binMidPoints = function(df, n){
  tempdf <- data.frame()
  tempv <- vector()
  for (i in 0:n){
    for (j in 1:nrow(df)){
      if (as.character(i) == df[j, 25])
        tempdf <- rbind(tempdf, df[j,])
    }
    min.value <- as.numeric(as.character(tempdf[1,1]))
    max.value <- as.numeric(as.character(tempdf[nrow(tempdf),1]))
    mid.point <- as.numeric(as.character(tempdf[1,1])) + ((max.value -
min.value) / 2)
    tempv <- c(tempv, mid.point)
    tempdf <- NULL
  }
  return(tempv)
}
shuffled.dataset <- dataset[sample(nrow(dataset)),]
test.data <- head(shuffled.dataset, 2472)
train.data <- tail(shuffled.dataset, 22251)

train.data$operation_duration <- NULL

op_duration <- test.data$operation_duration
test.data$operation_duration <- NULL

bin.mid.points <- c(540, 1410, 2190, 2940, 3720, 4680, 6030, 7770, 9660, 11520,
13380, 15240, 17070, 18960, 21420)

nb.model <- naive.bayes(train.data, "bin")
plot(nb.model)
nb.fittedbn <- bn.fit(nb.model, data = train.data)
pred <- predict(nb.fittedbn, node = "bin", data = test.data)
test.data[, "pred"] <- pred
test.data[, "operation_duration"] <- op_duration

hit = 0

threshold <- 1200 #20 Mins

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
    if (abs(bin.mid.points[1] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 1){
```

```
    if (abs(bin.mid.points[2] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 2){
    if (abs(bin.mid.points[3] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 3){
    if (abs(bin.mid.points[4] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 4){
    if (abs(bin.mid.points[5] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 5){
    if (abs(bin.mid.points[6] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 6){
    if (abs(bin.mid.points[7] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 7){
    if (abs(bin.mid.points[8] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 8){
    if (abs(bin.mid.points[9] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 9){
    if (abs(bin.mid.points[10] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 10){
    if (abs(bin.mid.points[11] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 11){
```

```
    if (abs(bin.mid.points[12] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 12){
    if (abs(bin.mid.points[13] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 13){
    if (abs(bin.mid.points[14] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 14){
    if (abs(bin.mid.points[15] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
}

acc <- hit / nrow(test.data)

MSE = 0

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
    MSE + (bin.mid.points[1] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[1] - as.numeric(as.character(test.data[i,26]))))-> MSE
  }
  else if (test.data[i,25] == 1){
    MSE + (bin.mid.points[2] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[2] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 2){
    MSE + (bin.mid.points[3] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[3] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 3){
    MSE + (bin.mid.points[4] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[4] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 4){
    MSE + (bin.mid.points[5] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[5] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 5){
    MSE + (bin.mid.points[6] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[6] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 6){
    MSE + (bin.mid.points[7] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[7] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 7){
```

```
    MSE + (bin.mid.points[8] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[8] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 8){
    MSE + (bin.mid.points[9] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[9] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 9){
    MSE + (bin.mid.points[10] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[10] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 10){
    MSE + (bin.mid.points[11] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[11] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 11){
    MSE + (bin.mid.points[12] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[12] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 12){
    MSE + (bin.mid.points[13] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[13] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 13){
    MSE + (bin.mid.points[14] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[14] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
  else if (test.data[i,25] == 14){
    MSE + (bin.mid.points[15] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[15] - as.numeric(as.character(test.data[i,26])))) -> MSE
  }
}

MSE / nrow(test.data) -> MSE
sqrt(MSE) -> RMSE

MAE = 0

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
    MAE + (abs(bin.mid.points[1] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 1){
    MAE + (abs(bin.mid.points[2] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 2){
    MAE + (abs(bin.mid.points[3] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 3){
    MAE + (abs(bin.mid.points[4] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 4){
    MAE + (abs(bin.mid.points[5] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 5){
```

```
    MAE + (abs(bin.mid.points[6] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 6){
    MAE + (abs(bin.mid.points[7] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 7){
    MAE + (abs(bin.mid.points[8] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 8){
    MAE + (abs(bin.mid.points[9] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 9){
    MAE + (abs(bin.mid.points[10] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 10){
    MAE + (abs(bin.mid.points[11] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 11){
    MAE + (abs(bin.mid.points[12] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 12){
    MAE + (abs(bin.mid.points[13] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 13){
    MAE + (abs(bin.mid.points[14] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 14){
    MAE + (abs(bin.mid.points[15] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
}

MAE / nrow(test.data) -> MAE
```

FEATURE CORRELATION MODEL SCRIPT:

```
library("bnlearn")
library("MASS")
dataset <-
read.csv("~/Documents/GitProjects/DV2542/4/R/result_operation_duration_sorted_m
ixed_bins_3000i_V2.csv", header = TRUE)

# removing the unnecessary columns
dataset$treatment_number <- NULL
# dataset$operation_duration <- NULL

for (i in 1:ncol(dataset)){
  dataset[[i]] <- as.factor(dataset[[i]])
}

randomSample = function(df,n) {
```

```
  return (df[sample(nrow(df), n),])
}

binMidPoints = function(df, n){
  tempdf <- data.frame()
  tempv <- vector()
  for (i in 0:n){
    for (j in 1:nrow(df)){
      if (as.character(i) == df[j, 25])
        tempdf <- rbind(tempdf, df[j,])
    }
    min.value <- as.numeric(as.character(tempdf[1,1]))
    max.value <- as.numeric(as.character(tempdf[nrow(tempdf),1]))
    mid.point <- as.numeric(as.character(tempdf[1,1])) + ((max.value -
min.value) / 2)
    tempv <- c(tempv, mid.point)
    tempdf <- NULL
  }
  return(tempv)
}

test.data <- randomSample(dataset, nrow(dataset)/10)
train.data <- randomSample(dataset, nrow(dataset)*(9/10))

train.data$operation_duration <- NULL

op_duration <- test.data$operation_duration
test.data$operation_duration <- NULL

bin.mid.points <- c(540, 1410, 2190, 2940, 3720, 4680, 6030, 7770, 9660, 11520,
13380, 15240, 17070, 18960, 21420)

node.names <- c("clinic", "staff_count", "anesthesia_method", "operation_code",
"anesthesia_count", "radiolog",
                "operation_card_method", "urolog", "kirurg",
"operation_weekday", "treatment_count", "diagnosis_code",
                "gynekolog", "patient_start", "age", "functioning_assessment",
"sex", "is_weekend", "operation_count",
                "anestesiolog", "operation_hall", "ortoped", "medicinare",
"bin")

arc.set = matrix(c("clinic", "anesthesia_count",
                   "staff_count", "anesthesia_count",
                   "anesthesia_method", "anesthesia_count",
                   "operation_code", "anesthesia_count",
                   "anesthesia_count", "treatment_count",
                   "radiolog", "treatment_count",
                   "operation_card_method", "kirurg",
                   "urolog", "kirurg",
                   "kirurg", "diagnosis_code",
                   "operation_weekday", "bin",
                   "treatment_count", "bin",
                   "diagnosis_code", "bin",
                   "gynekolog", "bin",
                   "patient_start", "bin",
                   "age", "sex",
                   "functioning_assessment", "sex",
                   "sex", "is_weekend",
                   "is_weekend", "operation_hall",
                   "operation_count", "ortoped",
```

```
                  "anestesiolog", "ortoped",
                  "operation_hall", "medicinare",
                  "ortoped", "medicinare",
                  "medicinare", "bin"),
                ncol = 2, byrow = TRUE, dimnames = list(NULL, c("from",
"to")))

corr.model = empty.graph(node.names)
arcs(corr.model) = arc.set

plot(corr.model)
corr.fittedbn <- bn.fit(corr.model, data = train.data)
pred <- predict(corr.fittedbn, node = "bin", data = test.data)
test.data[, "pred"] <- pred
test.data[, "operation_duration"] <- op_duration

hit = 0

threshold <- 1200 #20 Mins

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
    if (abs(bin.mid.points[1] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 1){
    if (abs(bin.mid.points[2] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 2){
    if (abs(bin.mid.points[3] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 3){
    if (abs(bin.mid.points[4] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 4){
    if (abs(bin.mid.points[5] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 5){
    if (abs(bin.mid.points[6] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 6){
    if (abs(bin.mid.points[7] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
```

```
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 7){
    if (abs(bin.mid.points[8] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 8){
    if (abs(bin.mid.points[9] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 9){
    if (abs(bin.mid.points[10] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 10){
    if (abs(bin.mid.points[11] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 11){
    if (abs(bin.mid.points[12] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 12){
    if (abs(bin.mid.points[13] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 13){
    if (abs(bin.mid.points[14] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
  else if (test.data[i,25] == 14){
    if (abs(bin.mid.points[15] - as.numeric(as.character(test.data[i,26]))) <=
threshold || test.data[i,24] == test.data[i,25]){
      hit + 1 -> hit
    }
  }
}

acc <- hit / nrow(test.data)

MSE = 0

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
```

```
    MSE + (bin.mid.points[1] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[1] - as.numeric(as.character(test.data[i,26])))-> MSE
  }
  else if (test.data[i,25] == 1){
    MSE + (bin.mid.points[2] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[2] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 2){
    MSE + (bin.mid.points[3] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[3] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 3){
    MSE + (bin.mid.points[4] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[4] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 4){
    MSE + (bin.mid.points[5] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[5] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 5){
    MSE + (bin.mid.points[6] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[6] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 6){
    MSE + (bin.mid.points[7] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[7] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 7){
    MSE + (bin.mid.points[8] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[8] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 8){
    MSE + (bin.mid.points[9] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[9] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 9){
    MSE + (bin.mid.points[10] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[10] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 10){
    MSE + (bin.mid.points[11] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[11] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 11){
    MSE + (bin.mid.points[12] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[12] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 12){
    MSE + (bin.mid.points[13] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[13] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 13){
    MSE + (bin.mid.points[14] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[14] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
  else if (test.data[i,25] == 14){
    MSE + (bin.mid.points[15] - as.numeric(as.character(test.data[i,26])) *
(bin.mid.points[15] - as.numeric(as.character(test.data[i,26]))) -> MSE
  }
}
```

```
MSE / nrow(test.data) -> MSE
sqrt(MSE) -> RMSE

MAE = 0

for (i in 1:nrow(test.data)){
  if (test.data[i,25] == 0){
    MAE + (abs(bin.mid.points[1] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 1){
    MAE + (abs(bin.mid.points[2] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 2){
    MAE + (abs(bin.mid.points[3] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 3){
    MAE + (abs(bin.mid.points[4] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 4){
    MAE + (abs(bin.mid.points[5] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 5){
    MAE + (abs(bin.mid.points[6] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 6){
    MAE + (abs(bin.mid.points[7] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 7){
    MAE + (abs(bin.mid.points[8] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 8){
    MAE + (abs(bin.mid.points[9] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 9){
    MAE + (abs(bin.mid.points[10] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 10){
    MAE + (abs(bin.mid.points[11] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 11){
    MAE + (abs(bin.mid.points[12] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 12){
    MAE + (abs(bin.mid.points[13] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 13){
```

```
    MAE + (abs(bin.mid.points[14] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
  else if (test.data[i,25] == 14){
    MAE + (abs(bin.mid.points[15] - as.numeric(as.character(test.data[i,26]))))
-> MAE
  }
}

MAE / nrow(test.data) -> MAE
```