Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

School of Computer Science and Statistics

MACHINE LEARNING CONDUCTED ON PHYSIOTHERAPIST DATA SET

Caleb Teo
[redacted]

Supervisor: Dr. Lucy Hederman

April 30, 2020

A dissertation submitted in partial fulfilment
of the requirements for the degree of
MAI (Computer Engineering)

Declaration

I hereby declare that this project is entirely my own work and that it has not been submitted as an exercise for a degree at this or any other university.

I have read and I understand the plagiarism provisions in the General Regulations of the University Calendar for the current year, found at `http://www.tcd.ie/calendar`.

I have also completed the Online Tutorial on avoiding plagiarism 'Ready Steady Write', located at `http://tcd-ie.libguides.com/plagiarism/ready-steady-write`.

Signed:  Date: 30-04-2020

Abstract

Machine Learning applications have a huge impact on various medical areas for diagnosis and treatment with great success. Machine Learning algorithms are used to aid physicians in the medical care of patients. The aim of this work is to investigate the Machine Learning algorithms that can predict the outcome of a physiotherapy patient's treatment. In this research work, a physiotherapist data set from a physiotherapist clinic was used to train Machine Learning algorithms in predicting the outcome of patients. Data processing techniques were applied to clean and prepare the data for the algorithms. The processed data was then used to train the Machine Learning algorithms. The findings were a successful implementation using Random Forest algorithm obtaining a high accuracy score in predicting patient's treatment outcome. The final approach included key mechanisms of SMOTETOMEK; a combined undersampling and oversampling technique, followed by Target encoding; encoding using Empirical Bayesian of a target. The work includes a thorough investigation of approaches using other data processing techniques and Machine Learning models such as Decision Trees, K-Nearest Neighbours, and Neural Networks. Several recommendations were given to further improve the results of the implementation. Recommendations were also given to the physiotherapist, which are the continuation of gathering data, standardising entry of data, and investigation of correlated features.

Acknowledgements

I would like to thank Dr Lucy Hederman, Dr. Kevin Koidl, and Dr Kristina Kopanova for their supervision in my dissertation. They have provided wonderful advise and insight to the research.
I am forever grateful to my family, who have always supported me with every possible need I have. Thank you for encouraging me through all the years of my time in Trinity.
I am thankful to my friends who have always supported me with their friendship and prayers. They have helped and spurred me on, especially through the last few weeks with the abrupt change in circumstances.
Finally I am forever thankful to God, the rock and foundation of my life. His everyday presence allows me to complete this work.

Contents

1 Introduction **1**

1.1 Research Goal . . . 3

2 Literature Review **4**

2.1 Machine Learning Overview . . . 4

2.1.1 Supervised Learning and Classification . . . 4

2.1.2 Data Types . . . 6

2.1.3 Machine Learning Algorithms . . . 7

2.1.4 Evaluation . . . 8

2.2 Data Processing Techniques . . . 9

2.2.1 Preprocessing Missing Values . . . 9

2.2.2 Encoding Methods . . . 10

2.2.3 Oversampling and Undersampling . . . 12

2.2.4 Combined Sampling Techniques: SMOTE with Tomek Links . . . 13

2.3 Machine Learning Algorithms . . . 14

2.3.1 Decision Trees . . . 14

2.3.2 Random Forest . . . 14

2.3.3 K-Nearest Neighbours . . . 15

2.3.4 Neural Networks . . . 16

2.3.5 Grid Search . . . 18

2.4 Machine Learning in the Medical Field . . . 19

3 Methodology **25**

3.1 Description of Physiotherapist Data Set . . . 25

3.2 Language Choice and Libraries . . . 27

3.3 Data Analysing . . . 27

3.4 Data Transformation . . . 30

3.4.1 Preprocessing - Cleaning . . . 30

3.4.2 Preprocessing - Label Encoding . . . 31

3.4.3 Sampling . . . 32

3.4.4 Splitting the Data . . . 32
3.4.5 Target Encoding . . . 33
3.5 Model Training . . . 33
3.5.1 Neural Network . . . 35
3.6 Model Evaluation . . . 37

4 Results & Discussion **39**
4.1 Model Accuracy . . . 39
4.2 Key Machine Learning Mechanisms . . . 40
4.3 Neural Network Model Loss & Accuracy Graphs . . . 43
4.4 Encoders and Synthetic Data . . . 48
4.5 Further Improvements . . . 49

5 Recommendations **51**

6 Conclusion **53**

List of Figures

3.1 Neural Network Architecture 36

4.1 Model Loss and Accuracy Graph for 1st Fold 44
4.2 Model Loss and Accuracy Graph for 2nd Fold 44
4.3 Model Loss and Accuracy Graph for 3rd Fold 44
4.4 Model Loss and Accuracy Graph for 4th Fold 45
4.5 Model Loss and Accuracy Graph for 5th Fold 45
4.6 Model Loss and Accuracy Graph for 6th Fold 45
4.7 Model Loss and Accuracy Graph for 7th Fold 46
4.8 Model Loss and Accuracy Graph for 8th Fold 46
4.9 Model Loss and Accuracy Graph for 9th Fold 46
4.10 Model Loss and Accuracy Graph for 10th Fold 47

List of Tables

3.1 The Physiotherapist Data Set 26
3.2 Distribution of Target Variable OUTCOME 28
3.3 New Distribution of Target Variable OUTCOME 32
3.4 Hyper-parameters search range for Decision Tree 34
3.5 Hyper-parameters search range for Random Forest 34
3.6 Hyper-parameters search range for K-Nearest Neighbours 34
3.7 Hyper-parameters used for Models 34

4.1 Average Accuracy of Machine Learning Models across 10 Folds 40
4.2 Average Accuracy of Machine Learning Models across 10 Folds without sampling and Target encoding 41
4.3 Average Accuracy of Machine Learning Models across 10 Folds without Target encoding 41

Nomenclature

E(x, w)	Error Function
b_{ij}	Bias Constant in Neuron
$d(x_i, x_k)$	Euclidean Distance
d_{XY}	Manhattan Distance
f	Activation Function in Neural Network
o_{ij}	Output of Neuron in Neural Network
w	Weight/Model parameters
x	Inputs vector
y'(x, w)	Model prediction value with x and w
y(x)	True value with x inputs
α	Learning Rate for Gradient Descent Function
FP	False Positive
FN	False Negative
KNN	K-Nearest Neighbour
ML	Machine Learning
NN	Neural Network
SMOTE	Synthetic Minority Oversampling Technique
SMOTETOMEK	SMOTE with Tomek Links
TP	True Positive
TN	True Negative

1 Introduction

The world has transformed dramatically and continues to transform due to the rapid evolution and adoption of Machine Learning [1]. To most humans, Machine Learning has integrated to all parts of our lives, whether it is clearly identified or not. Machine Learning has revolutionised how people interact with everyday life and technology. The application of Machine Learning spans from life changing technology like autonomous driving [2] and credit card fraud detection [3], to small changes like suggested time of our morning alarms. The history of Machine Learning comes from a mathematical statistical root and is part of Artificial Intelligence. It's early developments were in the late 20th Century, but followed by this new age of rich data, its development has major breakthroughs. Rich and plentiful data has become the catalyst in which Machine Learning has been rapidly pushed forward and proven substantial results. This makes the current topic of Machine Learning an exciting area of research and application in today's age of technology. With Machine Learning and the explosive availability of data, countless applications can be developed in transforming mankind and in benefiting all human life [4, 5, 6, 7].

There are a large number of Machine Learning applications which cover a range of different areas. Machine Learning is able to solve many problems when applied correctly and countless applications have been implemented. There are applications in different industry from medical, businesses, and genetic science. The work of Galatzer et al. [5] applied Machine Learning models to aid in forecasting PTSD. Their research was able to identify PTSD diagnosis and the relating factors. In the business industry, supply and demand is essential in the operation of a business, which is why Carbonneau et al. [8] incorporated Machine Learning to their benefit. They were able to test a range of Machine Learning algorithms, from Linear Regression, Support Vector Machines, Neural Network, and Recurrent Neural Network. They were able to apply the benefits of Machine Learning to their business. Although not every application is successful as proven by Libbrecht & Noble [9]. Libbrecht & Noble researched Machine Learning implementations for genetic and genomics studies and found that Machine Learning can work with large amounts of data. The issue in

which they ran into was that to successfully apply the techniques, it cannot be applied arbitrarily. They recognised that as technology develops, with large amounts of data output, machine learning techniques and expert knowledge will be essential.

Image recognition is another large area where Machine Learning is applied to. This area became very popular especially after the implementation of LeNet [10] which was a Convolutional Neural Network (a type of Neural Network). This then led to the advancement of image recognition and classification with Convolutional Neural Networks. A noteworthy Convolutional Neural Network is AlexNet [11] which produced very successful results. The deep Convolutional Neural Network can then applied to different areas. One example of image recognition which is applied to autonomous driving was researched by Bechtel et al. [2]. They were able to implement a low powered Neural Network for autonomous cars, which is able to take camera images as inputs and output steering angles. Another example of Convolutional Neural Network achieve in the area of medicine was by Van et al. [12]. This Convolutional Neural Network was able to detect Hemorrhages. This shows the wide range of applications currently being tested and applied.

This dissertation is research in the realms of Machine Learning and real data from the health domain. In the past two decades, machine learning has been trending in the world as the future to solve many different problems. Machine Learning is within the context of Artificial Intelligence and is an area in which provides predictive abilities. The focus of Machine Learning is the process in which an algorithm recognises patterns or trends in data and rules are enforced or finds weights values in mathematical equations. The weights are used along with input vectors in a linear equation and the algorithm's equation outputs predictions [13]. Once these rules are set or weights are found through a training process, the algorithm can then output predictions based on inputs. For a model to be trained, data is needed. The data is the bases of Machine Learning as this is what the algorithm will use understand the pattern that causes events [13]. In this dissertation, Machine Learning methods will be applied to a real-life data set. The data is from a physiotherapist clinic.

This dissertation is in collaboration with a physiotherapist clinic where the source of the data originates. The data was collected by the physiotherapist clinic of past patients. The data is compromised of patient's information, collected during their visit to the clinic with a particular injury. The data includes general information such as the patient's age, gender, occupation, and how they found out about the clinic. Part of the data set includes the patient's sports activities, how regularly physically active they are, indicators of the amount of pain and mobility, whether it is a

recurrent problem or not, type of injury sustained, the prescribed treatment. The data also includes labels of outcome with the number of consultations of treatment, and outcome of treatment. The data set records the physiotherapist assigned to the treatment of the patient. This data is used in a Machine Learning application.

This dissertation will describe how Machine Learning algorithm can be applied and utilised to predict the outcomes of the data set, in aid of the work of the physiotherapist clinic. The first chapter will describe the background, research question, and goals and aims of the dissertation. The following chapters will then be a literature review of Machine Learning overall and in the context of human health, the methodological approach applied based on the research question, the results of the approach, the discussion of the results, and a concluding chapter summarising the research completed.

1.1 Research Goal

There are a number of research investigations that can be explored. The motivation of the collaboration with the physiotherapist clinic was to increase the awareness of data collection and how data and application of Machine Learning can positively impact the practice. The physiotherapist clinic view that collecting this data and analysing it will help improve the quality of their work. From another angle, with Machine Learning, there are a number of sub-topics to consider from data analysing, data transformation, algorithms selection, and evaluation. All these sub-areas could be thoroughly investigated into, in the context of this practical problem.

For this dissertation, the objective is to explore what Machine Learning techniques work effectively for the physiotherapist data set in predicting the patient's outcome. From the data, the Machine Learning algorithms will predict on the Outcome of the patient. The research goal will be to answer: What is the best approach to predicting this outcome. To answer this question, Machine Learning will be applied to the data set and the different techniques implemented and their impact will be reported. The approaches will then be measured with Machine Learning performances metrics. The best approach will be determined by comparing the performances.

2 Literature Review

2.1 Machine Learning Overview

Machine Learning is the subject of algorithms observing and setting rules or following weight-adjusted mathematical equations based on patterns and trends from data. The computers can then used these rules or equations to make decisions or predictions. This is a powerful concept as computers can spot patterns that is not apparent to humans. The overall process of Machine Learning starts with data. The data is first processed into a form that can be processed by a machine and set as inputs to the machine. The algorithm will be trained with this processed data. The process of training the algorithm involves Machine Learning algorithms which each have their own parameters. Depending on the algorithms selected, the Machine will calculate its parameters differently and fit the parameters to the data. Following the training of the Machine Learning models, new inputs can be passed into the model and the model will produce an output or result based on its parameters and algorithm. The model can be evaluated with several metrics to verify the validity of the predictions. This summarises the basic approach to Machine Learning research and development. The area of Machine learning is a mature topic and this first section will highlight the overall ideas which will be involved in this dissertation.

2.1.1 Supervised Learning and Classification

In the area of Machine Learning, there are a number of different categories and approaches to solving problems. Machine Learning has three categories, Supervised Learning, Unsupervised Learning, and Reinforcement Learning. Supervised Learning is where the data set comes with labelled data. Burkov [14] explains that in Supervised Learning the labelled data is split into two parts, the feature vector x_i, and the label y_i. The value of i ranges from 0 to the number of data points N. In the feature vector, there could be more than one feature. The vector size can be denoted by D, the number of dimensions, which is also the number of features. The label data y_i can be either a real continuous value or a class. This labelled data is what the

Machine Learning algorithm learns and predicts on. Unsupervised Learning is when there is no labelled data. The algorithm will then group data samples together and they will be classed as one group. With this approach, there is no defined answer but the machine tries to find a pattern. In the final type, Reinforcement Learning is where the machine recognises the correct pattern through trial and error and is rewarded for successful outcomes. In my dissertation, the type of Machine Learning will be Supervised Learning as the physiotherapist data set has labelled data.

There are different approaches to Machine Learning which is determined by the problem in the domain. For Supervised Learning, the desired outcome will be to produce a model that can accurately predict the labelled data and to produce an accurate model the type of prediction must be defined. There are two types of predictions for Supervised Learning, Classification and Regression [14]. Regression is prediction in the realm of continuous values e.g. person's salary, age, temperature for the next week. Classification is prediction with types of classes e.g. Male or Female, Sunny, Cloudy, or Rain. The type of predictions will determine the approach and algorithms that should be used. In regards to the physiotherapist labelled data, the values are categorical which characterises this as a classification problem.

In the case of Regression problems, a suitable algorithm would be an algorithm that outputs a continue value e.g. Linear Regression. Linear Regression is suited for Regression problems as it can produce a real number. The algorithm trains with the labelled data and calculates weights that match the pattern in the data. The algorithm can then take inputs and return a real number predictions. In the other case with Classification, the output should be a category e.g. K-Nearest Neighbours. For K-Nearest Neighbours, the algorithm determines the classification of a new input through the 'K' closest data points using a distance metric. The algorithm takes a data point, and using its feature vector, calculates the distances to all the other points. The pre-determined 'K' number of closest points, measured using the distance metric is made into a set. From the set of 'K' closest points, the data point is classified by the vote of the closest points. The highest voted class is then determined as the class of the data point. The examples given are models that are suited and designed for those particular Machine Learning problems. The selection for the correct algorithm is important as not all algorithms are designed for every problem. By understanding the problem and data is the way to ensure you have an effective model.

In training the Machine Learning models, when the model is not accurately capturing the pattern in the data then this is called Underfitting [14]. When the model is being trained, the model takes the data points and runs its algorithm to find the

weights or rules that fit the data. When it is completed then the model can predict on new data points. If the predictions of the model on the training data are inaccurate, the training accuracy is low, the algorithm has not been able to accurately model the trends in the data. In this case, the algorithm is Underfitting the data. The way to overcome Underfitting is to use a more complex model. This way the more complex model can fit the data accurately but this could lead to Overfitting.

Overfitting is the opposite of Underfitting when the model fits the training data but does not fit the test data. Overfitting can be seen when the training accuracy is high but the validation or test accuracy is much lower. This means the model is predicting poorly on the test data or new data that has not been seen during training. The model is still able to achieve a high training accuracy which means the model is fitted to the noise in the data. Overfitting is a more likely case and can be overcome in many ways. The first way is to use a simpler model which stops the model from fitting to the noise in the data. The next appropriate approach is to obtain more data (if available). A third method is to add Regularisation which in effect keeps the model as simple as possible. Burkov [14] gives examples of using L1 and L2 Regularisation for Linear Regression which mitigates Overfitting. There are a wide number of different ways to stop Overfitting and is dependant on the model and approach to which is effective.

2.1.2 Data Types

Data is the key factor to Machine Learning and there are different types of data which play a factor in the approaches in handling data. Data are split into two main types, Numerical and Categorical. Numerical data can be continuous or discrete. With numerical data, it can mostly be passed to the Machine Learning algorithms as numerical data is easy for machines to understand. Categorical data are usually string form and are separated into classes. These categories are usually a defined split e.g. Male or Female, Primary School, Secondary School or University. Machines do not know how to interpret categorical data and the data must be handled to be converted into a numerical representation. This process is called encoding which is representing categorical data by a numerical value. There are a wide range of encoding methods and each have their strengths and weaknesses. Of these methods, not every method is suitable for different types of data and it is an important process to determine the most suitable technique. In proceeding sections, different types of encoding will be described, including details of the encoding method that suited for different types of data.

2.1.3 Machine Learning Algorithms

The history of Machine Learning algorithms started back in the 1950s and has continued to develop to this day. In an article stated by Igor Kononenko [4], he mentioned how three branches of research were explored that gives different types of Machine Learning algorithms. He mentions, statistical methods, neural networks, and symbolic learning. In later advancements, statistical methods would give Machine Learning algorithm such as Bayesian Networks, where symbolic learning which would include Machine Learning algorithm of Decision Tree, which are both applied algorithms to this day [1, 15]. The third is a Neural Network which is created based on the structure of a network of neurons in the brain. Kononenko states that Nilsson [16] was the one who initiated the Machine Learning techniques relating to statistical methods. He mentions names such as Rosenblatt [17] and Hunt et al. for the development of neural networks and symbolic learning respectively. All these areas have advanced over the years and many more algorithms were developed.

Loss Function and Gradient Descent

In all Machine Learning algorithms, the necessary components of a loss function and an optimiser is used to enable the training process. To be able to train the Machine Learning algorithms for the purpose of prediction, then a measurement of the number of predictions that were wrong is needed. This is the loss function which is a measure of the error based on a prediction [18]. The loss function is a calculation of the difference between the algorithm's prediction and the true value. The loss function can be calculated on a single data point but can also be summarised with an average value across multiple data points. Burkov [14] states the loss function as a building block in learning algorithms. The goal of the Machine Learning algorithm is to minimise this loss value. The loss function is then needed to train a Machine Learning algorithm. The common loss function used in Machine Learning algorithm is the squared loss function (1). The squared loss function is the prediction minus the true value squared. Burkov mentions the use of optimiser as another essential building block which is used to minimise the loss function. An optimisation method is need to update the algorithm's parameters in order to minimise the loss function. This is repeated until a minima is found for the loss function. An optimiser example would be Gradient Descent. Gradient Descent is an iterative process of updating parameters to lower the loss. The Gradient Descent takes the value of the parameter of the algorithm and updates it by subtracting of the product of the partial differential of the loss function and the learning rate (2). This is mentioned in both Bishop's and Burkov's book [14, 18] in relation to the gradient descent. This equation

(2), in essence, is updating the parameters one step at a time towards the minima. This is then the process in which Machine Learning algorithms are trained with data and how the correct rules or parameters are found.

$$E(x, w) = (y'(x, w) - y(x))^2 \tag{1}$$

where:

$E(x, w)$ = Error/Loss Function
$y'(x, w)$ = Model prediction of x (as inputs) with w (as parameters)
$y(x)$ = True value

$$w^{t+1} = w^t - \alpha \frac{\partial}{\partial E(w)} E(x, w) \tag{2}$$

where:

$E(x, w)$ = Error/Loss Function
w^t = Parameters at iteration t
w^{t+1} = Parameters at iteration t+1
α = The Learning Rate

2.1.4 Evaluation

There are standard Machine Learning Evaluation method in which different algorithms' performance can be compared. In Burkov's book on Machine Learning [14], a number of metrics assessments of classifiers are given. These metrics are specific to the problem in the same ways as Machine Learning algorithms. For this Machine Learning dissertation, the prediction target is a Multi-class classification. The metric which is relevant to this is accuracy. Accuracy (3) is the number of correct predictions from all predictions.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

where:

TP = True Positive
TN = True Negative
FP = False Positive
FN = False Negative

This metric is suitable for a multi-class classification as the denominator will the total number of prediction samples and the numerator will be the number of correct predictions (in equation (3). This metric is a commonly understood and is used in many other research papers.

2.2 Data Processing Techniques

The following sections details the Data Processing techniques. This section will review the techniques which will be used in the experimental phase. The following subsections will go over a several preprocessing techniques, encoding methods, and sampling techniques.

2.2.1 Preprocessing Missing Values

From the work of Reedy [6], in the event of missing data, there must be some method to fill in this missing data. In Burkov's book [14], it suggests a number of ways in which missing data can be dealt with. The first method is removing the rows with missing data but he prefaces that it is only suitable for large data sets. This is also mentioned by Gelman & Hill [19] and they state a number of possible issues with removing rows of data. The first issue is if the removed data points remove a trend in the data set then the model will not learn this trend. This is why Burkov prefaces the technique with large data sets but even with large data sets, this issue could still cause a negative effect. The other issue that Gelman and Hill mention is by removing more features and points the model might become too simple and not perform accurately to new data. Gelman and Hill proceed to present other methods, one method involves removing whole features depending on the percentage of missing values. Although this could have the same issue as the first mentioned in removing data points. An additional method is to retain all the information but to fill in the missing data. This fill method could use the mean value and impute that value into the missing points. This requires to calculate the mean of a feature and then fill in the missing values. In the case of categorical features, the model (most frequent) category can be used to fill in the missing value. An alternative approach is to fill in the missing values with an assigned value. This assigned value would be outside the range of possible values and in hopes that the model will learn that this value indicates a particular pattern. The model will learn from this missing variable value. Both these techniques are also mention in Burkov's book as suitable approaches. The last possible technique that could be used to impute the missing data is to use regression or model. This technique is to build a model in which can take in existing inputs and output a predicted value for the missing values. This is an advance

technique and requires additional cost. This method also requires you to investigate correlated features to use as inputs and these inputs must be all present to be used to train. The final remarking comment by Burkov on missing data was that the methods are data set dependant and cannot be determined unless through trials. There are a number of different methods to impute missing data and deciding which method will be most effective will be done through experiments.

2.2.2 Encoding Methods

Encoding method is the process of transforming nominal data into numerical data for Machine Learning algorithms. There are many different encoding methods and every encoding methods has strengths and weaknesses. For different data types, different encoding methods are more suited to be used to transform the data. It is important that the features are encoded with the best representation so that the model can observe the pattern.

One encoding method is Label Encoding which is a simple and quick method. Label Encoding is done by representing each unique category in a feature list with a number. The process starts with 0 and increments for each new category. In this method, there is no meaning in the encoding to the numerical representation. This method is simple to use as it does not require any prior knowledge of the values in the feature list and how they relate to each other or the prediction class. The biggest flaw with this encoding method is that the method gives an order to the categories in the feature list. When the Machine Learning algorithm receives the Label encoded data, the numbers could translate into an 'order', and the algorithm would then train to give some values a more significant impact. This 'order' from the encoding method would be misrepresented as this order does not exist in real-life. For example, when Label Encoding is used on animals, giving dog = 0 and cat = 1. This example will then give the cat a high significance than the dog and this would be a misrepresentation when neither animal is better. The algorithm will then be trained poorly and would have a lower performance. Label encoding could be applied to features with a natural ordering such as education level; primary, secondary, university. Although this is useful, Label encoding is usually limited in contributing to the prediction of Machine Learning algorithms.

Another encoding method is One-hot Encoding [14] which represents all features equally. One-hot encoding takes the number of unique categories in the feature and creates a vector. This vector's size is equal to the number of categories in the feature. Each column in the vector represents one of the categories and in the case of a present

category then a value of 1 is given. The remaining columns are assigned 0. This creates a binary feature vector that represents each category with a 1 at the assigned nth column. Burkov [14] gives an example with colour. There are three colours in the feature space; red, yellow, and green. The One-hot encoding transforms each category into the following:

- red = [1,0,0]
- yellow = [0,1,0]
- green = [0,0,1]

The colour red is corresponding to the first column, yellow with the second, and green with the third. This is how One-hot encoding transforms the nominal data. The advantage of One-hot encoding is that it overcomes the weakness of Label encoding by not introducing an ordering. One-hot encoding represents the categories equally and allows the Machine Learning algorithms to determine the pattern from the features in the data. The one weakness of One-hot encoding is that for every new category, the dimensionality of the vector increases. This makes One-hot encoding memory intensive which is something to consider when implementing the data processing.

Target Encoding [20] is an encoding method in which give meaning to the encoded categorical features and benefits the Machine Learning algorithm. Target Encoding is the method in which utilising a target to represent the categories. This method was purposed by Micci [20], which tackled the problem of high cardinality categorical features. Micci's approach is encoding with an Empirical Bayesian. The output of the Target encoding is an encoding that relates the values in the feature being encoded and the values in the target. The process of Target encoding starts with gathering all the instances of a single category value in a data set. A specific target is selected from the data i.e. another feature or the label data which predictions are made. The corresponding target values to each instance is then gathered. The average of all the instances' target values is found and this is the value to be encoded to that categorical value. This is repeated for each unique categorical value in the feature. With this approach, the range of the output encoding is the range of the target variables. This averaging of all the occurred value is a simple Empirical Bayesian approach which encodes the features with a probabilistic representation. This is the strength of the encoding method as it encodes meaning to the categorical values which benefit the Machine Learning algorithm. When the prediction labels are used as the target, the encoded values will point in some direction to the most probabilistic label. A Machine Learning algorithm can then use this information and have a higher chance

to train to a higher accuracy.

2.2.3 Oversampling and Undersampling

In a data set, having a balanced target data can better train the models and there are two types of techniques to balance data. In some data sets, the target feature in which the models predict on, would not always have an equal amount of each class. In a classification problem, the Machine Learning algorithms expect to have a balanced target with an approximately equal number of each class. This leads to the implementation of techniques called 'Sampling' which creates balanced data sets. There are two types of sampling techniques, 'Oversampling' and 'Undersampling'. Oversampling is the approach in which you create more minority case examples to match the number of majority case examples. This technique will then increase the number of data points from the original. An example of this is Synthetic Minority Oversampling Technique (SMOTE) [14, 21]. The other type is undersampling which is the opposite of Oversampling. Undersampling is where a number of samples are chosen (restricted by the maximum in the minority cases) and equal numbers are taken from minority and majority cases. This removes data points from the majority class. This means that the output would be an equal distribution but with the majority cases under sampled. This means the output training data is a small size compared to the original data set. There are many different ways to do undersampling. A basic approach is to do random undersampling which randomly removes samples from the majority class. A more advance method is using Condensed Nearest Neighbours [22]. This method involves sampling the majority classes for data points that are hard to classify (points near the separation border between classes), and grouping them with the minority classes. This approach then produces a data set that is evenly distributed and a data set that has data points that have a significant impact. This undersampling technique essentially picks the majority class points that have the biggest impact on the model. Upon training, the new samples will then produce a more accurate model. From the research of Mazurowski et al. [23], they concluded that from a Neural Network point of view, that there is an increase in performance after applying sampling techniques. In their conclusion, they stated that between undersampling and oversampling there is no better solution. They recommended the developer to choose a solution depending on the characteristics of the data set, class distribution, number of data points, and number of features. This informs the study that in the case of imbalanced data for the physiotherapist data set, sampling techniques will need to be applied.

2.2.4 Combined Sampling Techniques: SMOTE with Tomek Links

There are sampling techniques which combined the two types of sampling and produces a better result than either type. These techniques use some oversampling and undersampling and apply each method's strength. An example of this is the combination of oversampling technique; SMOTE and undersampling technique; Tomek Links. The SMOTE technique was developed by Chawla et al. [21], an oversampling technique that uses KNN to oversample the minority class. The technique introduces synthetic sample points by taking the minority class points as inputs. The synthetic sample points are created along the feature space of the minority classes between K Nearest Neighbours. A KNN of the minority case is constructed (Chawla et al. [21] suggest K = 5), and depending on how much percentage of sampling is needed, a different number of neighbours are used. The newly generated points are created by taking the distances between the current point and their KNN. Taking these distances and multiplying each distance by a random number between 0,1; this produces a new point randomly shifted between the KNN. Chawla et al. describe this producing of points done in the feature space. They compared the effect of SMOTE on a number of data sets and found that in most data sets with imbalanced data the model predicted more accurately. They commented that there is a bias introduced for the minority class and it could be seen from their data set which had the lowest amount of imbalanced classes. This is the first part of the combined sampling technique. The second part is the undersampling technique using Tomek Links. Tomek Links [24] created by Tomek, is a pair of points, where one point is a minority class and the other is a majority class. This method is a modification to Condensed Nearest Neighbours. The Tomek Links are formed from the boundary points between the two classes and the closest point to the minority class, identified as a majority class is the Tomek link. This identifies the boundary between two classes and in the undersampling method, the majority cases in the Tomek Links are removed until a satisfactory level of sampling has been achieved. In the combined sampling technique of SMOTE with Tomek Links [25], the data set is first passed to SMOTE. This creates an evenly distributed data set with synthetic data points. Tomek Links are then applied to the data set and the Tomek Links are removed from the data set. This is to remove any possible noise around the class bounders. In this combined approach, both points of the Tomek Links are removed because after the processing of SMOTE the distribution is even and thus both can be removed without losing even distribution. From the work of Batista et al. [25], it can be seen that using the same Decision Tree, and data set SMOTE with Tomek Links produces the best AUC result (at 92%). It can be seen that using SMOTE alone resulted in a slightly lower result at 91%. There is a small difference between the two

approaches but this was for this particular data set and could have a more significant improvement on another data set. These methods can be a powerful processing step in enabling the model to be trained more effectively and given a more rigorous model.

2.3 Machine Learning Algorithms

2.3.1 Decision Trees

Decision Tree is a Machine Learning algorithm which models the data like a tree with nodes as decision points and bottom leaves as classifications. There are many implementations of Decision Tree from CART, ID3, and C4.5 [14, 18]. The Decision Tree starts at a root and this root is labelled with the first decision. The decision splits the data into two parts, one part will be captured by one leaf and the rest by the other. The proceeding leaves can then become another decision node or an end leaf. This repeats until all data points belong to an end leaf. The algorithm is then represented as a tree and has a depth and number of leaves. The end result is a set of rules that the Decision Tree follows for classification of data points. The Decision Tree will then have sets of the training data spread amount the leaves. A number of hyper-parameters can be set for a Decision Tree to control the training of the algorithm. One hyper-parameter is the minimum number of samples split, which is the number of samples needed in a decision node (node with parents and children) to be split. This means that all nodes except leaves will have a minimum number of samples. The other hyper-parameter is the minimum samples leaf which is the minimum number of samples for an end leaf node. This is for all end leaf nodes, there is a minimum number of samples that are assigned to that node. Another important hyper-parameter is the maximum depth of the Decision Tree. This is the maximum number of the levels of the Decision Tree. In the implemented algorithm, these parameters can be adjusted to combat overfitting or underfitting. Decision Tree is good to use in cases where there is no domain knowledge and can perform exact classification [1]. Decision Tree can become very complex as the tree grows in depth and number of leaves. This will then cause the model to overfit the data. A way to reduce the overfitting is to tune the hyper-parameters of the model.

2.3.2 Random Forest

Random Forest [26] is an ensemble of Decision Trees which vote on the classification. In a Random Forest model, a number of Decision Trees are created from the data set. This is done by making multiple data sets which are samples of the

training data set [14]. Each tree is then trained with one of those sampled data set. Each Decision Tree is designed to be uncorrelated with each other and each tree outputs a classification. After each tree has made a prediction the class with the most votes is the Random Forest prediction. Similarly in a Decision Tree, Random Forest has hyper-parameters that can be selected. As Random Forest is a collection of Decision Trees, the hyper-parameters are the same. One new hyper-parameter that a Random Forest has is selecting the number of Decision Trees; the number of estimators. With the same purpose, these hyper-parameters can be determined to create the best fit. Through the voting in the Random Forest, the model is more powerful in predictions than the Decision Tree as the collection of Decision Trees help overcome the overfitting and errors of a single Decision Tree. The important aspect of the Random Forest is to ensure that the Decision Trees are uncorrelated with each other as they will then be able to predict more generically which help stop overfitting. Random Forest then becomes a slower model to train over a Decision Tree but can produce more accurate predictions.

2.3.3 K-Nearest Neighbours

In the K-Nearest Neighbours algorithm [14, 18], a K number of data points that are closest to the prediction point is determined by the most common class out of the K points. The model takes in a data point and calculates the distances to every point. The K points that are closest are then put into a list and the highest count of a class is the predicted class. For example, with a new data point, for K = 5, one point is class 1, one point is class 2 and three points are Class 3, the model will classify the point as Class 3. The distance metric is usually Euclidean Distance [14, 27] but any distance can be used i.e. Manhattan Distance [28]. The Euclidean and Manhattan distance formulas can be found in equation (4) and (5) The main variable in the KNN is the value of K. Choosing K is an important task as having smaller values of K will lead to overfitting the data. Having a larger value for K will lead to underfitting and inaccurate classification.

$$d(x_i, x_k) = \sqrt{\sum_{j=1}^{D} (x_i^j - x_k^j)^2} \tag{4}$$

where:

$d(x_i, x_k)$ = Euclidean Distance
x_i^j = x value of dimension j for coordinate i
x_k^j = x value of dimension j for coordinate k
D = The number of dimension

$$d_{XY} = \mid (X_i k - X_j k) \mid \tag{5}$$

where:

d_{XY} = Manhattan Distance
$X_i k$ = Vector X for i-coordinate
$X_j k$ = Vector X for j-coordinate

2.3.4 Neural Networks

Neural Networks [18] is a collection of neuron nodes organised in layers, the network feeds the data through layers until the final layer, where a prediction is made. There are many different types of Neural Networks e.g. Feed Forward, Convolutional, Recurrent. The simplest is a Feed Forward Network [18]. The Feed Forward Network is made up of layers and there are three types of layers. The first layer is the input layer, the final layer is the output layer and layers in between are the hidden layers. The input layer takes in the input from the training data. The output layer takes the final values from the hidden layers and outputs a prediction. The hidden layers are layers in between, that takes inputs, calculates a new value and passes this to the next hidden layer. Each layer is made up of a number of neurons. Each neuron takes all the values from the previous layer (in a Feed Forward Network) and does a linear operation. The values passed to the neurons is multiplied by weights. The sum of the products between the weights and inputs is then passed to an activation function. The output value of this activation function is the final output to the neuron 6.

$$o_{i,j} = f(\sum w_{i,j} * x_{i,j} + b_{i,j}) \tag{6}$$

where:

$o_{i,j}$ = Output of Neuron
f = Activation Function
$w_{i,j}$ = Weights for input i,j
$x_{i,j}$ = Input Value for i,j
$b_{i,j}$ = Bias constant

There are a number activation functions that can be used e.g. Sigmoid Function, Rectified Linear Unit Function, Hyperbolic Tan. Each activation function has a different effect on the Network and is data dependant on which one to use.

A Neural Network is trained through an algorithm called Back Propagation [29]. In the process of Back Propagation, the error is calculated with a metric Mean Squared Error (MSE) or Root Mean Squared Error (RMSE). This calculation process is called the forward propagation. Using this error and partial differentiation, new weights can be calculated. The key to the Back Propagation algorithm is the Chain Rule [14]. Using Back Propagation and Gradient Descent (2), the best new weight value is found. In the Back Propagation algorithm, the gradients of each neuron are calculated through connected neurons and using the Chain Rule. The Chain rule is shown in equation (9). From equations (7), (8), (9), it can be seen the Back Propagation algorithm using the Chain rule [29]. The Back Propagation is started with the total error from the output neurons. The total error is the difference between the predicted state and the desired state for every output neuron (7).

$$E = \frac{1}{2}\sum_{c}\sum_{j}(y_{j,c} - d_{j,c})^2 \tag{7}$$

Through partial differentiation, the gradient of the error with respect to the output neuron value (y_j) is calculated (8). This can be done for all output layer neurons. To be able to calculate the hidden layers' gradients then the Chain rule is needed. The values can be calculated by using Chain rule (9). In this equation, assuming x_j is feed into y_j then the error (E) differential in respect to x_j can be solved through the Chain rule. This value of $\frac{\partial E}{\partial y_j}$ is then passed back to the previous neuron that it is connected to, which is represented by equation (9). This can then be applied to multiple connecting neurons. The gradient (10) of each neuron with respect to the loss function is then the sum of the partial differentiation values (where the partial differentiation value are determined by the connected neurons). This gradient is then used in the Gradient Descent formula (2) and the new weights are calculated. The Neural Network continues to update weights for each training instance (data point). The Neural Network can then repeat this training for a set number of epochs. Through Back Propagation and Gradient Descent, Neural Network becomes a viable solution in Machine Learning.

$$\frac{\partial E}{\partial y_j'} = y_j' - d_j \tag{8}$$

$$\frac{\partial E}{\partial x_j} = \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial x_j} \tag{9}$$

$$\frac{\partial E}{\partial x_i} = \sum_j \frac{\partial E}{\partial x_j} \frac{\partial x_j}{\partial x_i} \tag{10}$$

E = Error of network
y'_j = Predicted output of neuron j
d_j = Desire output of j
x_i = Hidden Layer Neuron i
x_j = Hidden Layer Neuron j

In Gradient Descent [30] every data point is used in the updating of the new weights, which can lead to slow training speeds. This can be solved through batch updates in a technique called Stochastic Gradient Descent. In Stochastic Gradient Descent, the gradient is calculated over a batch of data points and that gradient is then used for the following batch. In equation (11), the gradient of the error with respect to the weights is averaged over a batch of 16. The batch size can be varied depending on the required speed of training. Another consideration for Stochastic Gradient Descent is that with smaller batch sizes the gradient can vary with each batch and led the poor conversion in the training i.e. the minima is not found. Therefore a suitable batch size must be selected for the Neural Network.

$$\frac{\partial E}{\partial x_i} = \frac{1}{16} \sum_{k=j}^{j+15} \frac{\partial E}{\partial x_k} \frac{\partial x_i}{\partial x_i} \tag{11}$$

2.3.5 Grid Search

For each of the models, some hyper-parameters can be selected and one simple and effective way to select hyper-parameters is using Grid Search. Grid Search [14] is an algorithm which takes a model and a list of hyper-parameter values. The algorithm would use the training data set and validation data set. The algorithm will then take the model and set the value of the first set of parameters and train the model. Then it would test its performance using the validation set. This will be repeated for each combination of parameters. The best performing model will then be saved and the parameters returned. This method is a way to find the most suitable hyper-parameters. This method can also be time-consuming as it goes through every possible combination and therefore the selected range of hyper-parameters to test is essential. This range can be found by starting with an approximate search and then eventually a fine search to narrow it down to the best parameters.

2.4 Machine Learning in the Medical Field

There are many Machine Learning research papers in the field of medicine. In the majority of cases, the goal is to produce a model in which can predict knowledge that is beneficial to the patient. The research touches on many different areas from trying to predict on heart diseases, liver diseases, kidney diseases stages, to breast cancer. There is research which investigates into mental states by classifying the stress of an employee at work [6]. Each research paper investigates Machine Learning techniques and evaluates the impact of the models. There are a number of popular models among the papers which will be mention in the proceeding paragraphs. Each investigation has its own approach to the preprocessing of data. In each problem domain, the overall aim for the Machine Learning algorithm is to aid physicians in the process of diagnosis by providing this tool which was concluded by Ayeldeen et al. [31]. Across a number of papers, different methods can be compared and analysed to review the current approach to Machine Learning in the medical field.

As previously mentioned, preprocessing is a necessary step in transforming nominal data into numerical data that machines can interpret. Preprocessing is dependant on the data set and the categorical data that is present and preprocessing can be conducted in a number of ways. In the work completed by Reedy [6], the data preprocessing techniques used for the Machine Learning algorithms were a combination of One-Hot encoding and Label encoding. One-Hot encoding was applied to a number of features and Label encoding was used for the rest of the categorical features. This is one of the rare few papers in which explicitly dictates the encoding process, wherein many other papers it is not stated. In another Machine Learning application by Ayeldeen et al. [31], where the state of Liver Fibrosis is predicted on, it is only explicit that all attributes were transformed into numbers. It can be assumed, from the basic knowledge of Machine Learning algorithms, that preprocessing was done on the data set before training of the model.

A part of the preprocessing stage is the steps to fill in missing data and additional techniques that would represent the data in a more accurate way. In many data sets, there are usually missing data points. Gelman and Hill, [19], describes that there are a number of different reasons as to why there is missing data. The missing data could be related to the specific feature or it could be related to another feature within the data set. In Reedy's work [6], there is a brief mention of missing nominal data being filled in with a zero value (making it equivalent to a 'No' answer). This method of filling missing data point is mentioned by Gelman and Hill but the logic to this approach was not described. This can be seen as a necessary step in the preprocessing

stage depending on the data set. There are two more techniques mentioned in other papers that can be part of the preprocessing stages. The first technique is normalisation. This was used by Panwong and lam-On [7] in their research to use Machine Learning to predict the 'Transitional Interval' for Kidney Diseased patients. They used this technique in addressing sugar, fat, and waste levels in the blood. This technique relates to a single feature and normalising the data points to a range between zero and one. This will transform the data to be represented on a scale between zero and one, with the highest value being represented by one and the lowest being represented by zero. This is a standard way of normalisation [14]. The second technique that Panwong and Iam-On mentioned was binning. Binning technique is taking a continuous feature and sorting them into sections called bins. These bins are a sub-ranges within the continuous feature space. Panwong and Iam-On used this method on the ages of the patients and split the ages into three bins. The first bin between under 57 years old, second bin 57-74 years old and finally above 74 years. With these bins, Panwong and Iam-On transformed them into numerical features space with One-hot encoding (not explicitly stated but an assumption based on the feature table and value domain). These preprocessing steps, dealing with missing data, normalisation and binning are only a small number of techniques that can be applied to the data set before training. Through the literature, it can be seen that not many researchers reveal the way the data was prepared.

Among the works reviewed in this dissertation, there are several of Machine Learning models which can cover different diagnosis problems. When reviewing models, it would be worth comparing the different models and the performance recorded. Exploring the different models and their impact can give insights into what models might be a suitable match for the physiotherapist data.

In many research works, Decision Tree is an algorithm that is tested and performance is evaluated. In the survey conducted by Tomar and Agarwal [1], Decision Trees are used in main data mining implementation as they state the need for no domain knowledge. This advantage of not needing domain knowledge makes Decision Trees a useful model to test. Tomar and Agarwal mention how Decision Trees are used in the areas of breast cancer and chronic diseases detection. An example of Decision Trees used is the work completed by Hassan et al. [32]. In their work, they used a number of models on heart disease and hepatitis disease data set. They employed Decision Trees and recorded an accuracy of 78% and 72% for the heart and hepatitis data set respectively. Hassan et al. also reported on Precision, Recall, and F1 Score. In a more specific study by Tu et al. [15], predicting on patients with heart disease, they recorded to have used Decision Trees. From their results,

they reported on Precision, Recall, F1 Score, True Positive Rate and False Positive Rate. The precision for their Decision Tree was 79% and comparing to Hassan et al [32], whom obtained a precision of 80% in their heart disease data set, shows that they both achieved similar scores. In comparing another paper, where Machine Learning algorithms were used to predict stress levels [6], the Decision Trees implemented obtained a precision of 81% and an accuracy score of 70%. Across these papers [6, 15, 32], Decision Trees can be applied as an algorithm and can provide significant results in accuracy and precision.

Random Forest is another well-implemented algorithm, that provides effective prediction results. Hassan et al. [32], along with the Decision Tree, implemented a Random Forest. The Random Forest outperforms the Decision Tree and obtained an accuracy of 83% and 85% for heart and hepatitis data set respectively. This is a logical result as having a collection of Decision Trees would be more powerful to a single Decision Tree. The Random Forest precision score for the heart disease data set was 84%. This can be compared to the work of Reddy [6] who also implemented a Random Forest and obtained a precision of 80%. Even though these two implementations have different problem domains, it can still be accredited to the Random Forest algorithm in achieving a relatively high accuracy score. Finally, in the works of Pangwong and Iam-On [7], predicted with an accuracy score of 60& in their Random Forest implementation. In two of the three examples, Random Forest displays an impressive classification power and in the third case [7], the Random Forest's result was improved with an additional transformation. This leads to concluding that Random Forest is a model to consider in discovering a solution. In all three cases, the Random Forest algorithm outperformed the Decision Tree. In Hassan et al. paper [32], the improved between the Decision Tree and Random Forest was 5% to 13% in terms of accuracy score. This is similar in Panwong and Iam-On where the increase was 5% and a little lower at 3% for Reddy's models [6]. With this insight in mind, proceeding into the implementation of this dissertation, an approximate idea of the performance difference can be expected.

In Panwong and Iam-On's work [7], they applied a data transformation called SMOTE which impacted positively the results. This algorithm transforms the data set by oversampling the minority cases in a data set. This creates a data set which is made up of the original data set and additional data points that balance the number of classes in the target. The results reported by Panwong and Iam-On after applying SMOTE, increased the accuracy of the Decision Tree from 55% to 80%. The increase in Random Forest performance went from 60% to 87%. Panwong and Iam-On stated that the imbalance in the data targets cause the low accuracy of their initial results

and it can be proven that this was the case. This insight shows that for a data set, having a balanced data target is important as it increases the metric performance of the algorithm and this balance data set can be achieved through sampling techniques.

Other algorithms are applied as comparisons to Tree type algorithms and they perform significantly well. In Tomar & Agarwal [1] survey, other algorithms evaluated were K-Nearest Neighbours (KNN), Bayesian Networks, and Neural Networks. Tomar & Agarwal describes how KNN was used for early warning systems and a way to correlate the relationship between heart diseases and hypertension. KNN was another model tested by Hassan et al. alongside Decision Trees and Random Forest. They produced a result of 67% (heart data set) and 72% (hepatitis data set) for KNN. In Reddy's paper [6], KNN is applied and achieves an accuracy of 73% which is comparable to the Random Forest model in the same paper. In both works, KNN can produce predictions that are similar in performance to Tree type models. Another algorithm that is mentioned is Naïve Bayesian, which is a network-based on conditional probabilities [18]. Hassan et al. implemented a Naïve Bayesian Network and was able to achieve 65% and 77% precision on the data sets. This could be seen as significantly lower than the other models tested. In contrast to the work by Tu et al. [15], who applied a Bayesian Network and obtained a precision of 82.5%. Tu et al. were able to apply Bayesian Network which produced a higher precision score compared to Hassan et al. There are two factors that could have contributed to this difference besides the clear factor that the two data sets are different. The first factor is that in the work completed by Tu et al. the record result is with an additional bagging technique which improves the precision metric. The second factor is stated by Tomar and Agarwal, that Naïve Bayesian Network models have an assumption that features are independent of each other. This assumption cannot be assumed with patient data as there would be some correlation between features. This is important to consider as with most patient data, the features will be correlated and so Naïve Bayesian Network would not be the most suitable model to obtain a high prediction performance.

Neural Network is a Machine Learning algorithm which has gained a lot of attention in the last decade and features in medical Machine Learning literature. Tomar and Agarwal [1] mention Neural Networks in the survey and give example applications. The example included a classification on the diagnosis of chest diseases i.e. Lung Cancer, Asthama etc. The example illustrates that a Neural Network is a possible solution for applying medical patient data to target diagnosis classification. In another piece of work by Mazurowski et al. [23], who explore Neural Networks

and imbalance classification and how it can affect medical machine decisions. Both papers comment on the way Neural Networks can overfit the data. Overfitting is the concept that the Neural Network will be trained in a way that can predict well on training data but does not perform well on real and new data. Mazurowski et al. has many results and compares between varying features, sample numbers, and different optimisers. For the scenario using 10 features, and a ratio of 1:9 split on minority classes, the AUC achieved was approximately 0.87. This is a good result as values of AUC closest to 1 shows good performance. This Neural Network was able to predict well on breast cancer patients.

In this chapter, an overview of Machine Learning, data processing, Machine Learning algorithms, and an overview of the Machine Learning applications in the Medical Field was researched. The broad concepts of Machine Learning were introduced from the idea of Supervised Learning and Classification problems to the essential role of data and Machine Learning algorithms, and how algorithms are evaluated. Data is a critical part of the Machine Learning process and must be handled with care. Several data processing techniques were investigated from imputing missing data, encoding methods of categorical data, and sampling techniques to balance data sets. All these are important techniques to prepare the data before applying to the algorithms. A number of Machine Learning algorithms were reviewed, those being Decision Trees, Random Forest, K-Nearest Neighbours, and Neural Networks. All these algorithms were explained and evaluated in relation to their implementations. Grid Search was another concept researched as it provided the method to tune hyper-parameters for each algorithm. From this review of Machine Learning research in the Medical Field, there are a wide range of applications using many different algorithms. It can be reported that the number of areas covers span from breast cancer data and heart problem data [23, 32], to kidney disease data [7], and even to stress of employees [6]. The Machine Learning process starts with a preprocessing step that prepares the data. The techniques that are used are encoding data into numerical representations, handling missing data, and separating data i.e. binning. From the range of research works, the Machine Learning algorithms that were used were Decision Trees, Random Forests, K-Nearest Neighbours, Naïve Bayesian and Neural Networks. In the domain of Tree type algorithms, their advantage of not having a need for domain knowledge makes it a suitable model to test. KNN and Neural Network are models that could be tested as they produced good results demonstrated by other implementations. Naïve Bayesian Networks is shown to be a good Machine Learning algorithm (with the proper accompanying techniques) but would be hard to train as the data features are closely dependant on each other. From reviewing approaches from other people, the

approach to answer what techniques and model works most effectively on the physiotherapist data set is narrowed and given direction.

3 Methodology

The following chapter describes in detail the data, the data analysis, data transformation, Machine Learning model training and model evaluation. The data will be described for clarity in further discussions. The chapter will describe how the data was analysed and what insights contributed to the data transformation and model training. There will be descriptions of insights that did not have any benefits to the preprocessing. There will be details of the methods in which was applied to the data. These methods will be described, the purpose in which they were chosen, and their impact. The details of the models used will be reported and how the models were trained to the data. The final section will detail the model evaluation and the method in which the accuracy of the model was measured.

3.1 Description of Physiotherapist Data Set

The physiotherapist data set is a mixture of categorical and numerical data. Each data point is a patient who has attended the physiotherapist clinic for some type of treatment. The data points are associated with one person and there are a number of features for each person. Table 3.1 illustrates the data set's feature name, description of the feature, and data type.

Name	Description	Data Type
CLIENTNO	Random assign client number	Numerical
SEX	The sex of the person	Categorical
OCCUPATION	The occupation of the person.	Categorical
INTROSOURC	The source in which they found out about the physiotherapist clinic	Categorical
Activity Level	A standard measure of how active the patient is	Categorical
Activity Days	A description of what kind of physical activities the patient does	Text String
Imaging	Yes/No to imaging of injury obtained by patient	Categorical
DOB	Date of Birth of patient	String Date
STARTDATE	Start date of consultation	String Date
ENDDATE	End date of consultation	String Date
DIAGNOSIS	The Diagnosis given by physiotherapist	Categorical
FIRSTRECUR	If injury is First or Recurrent for patient	Categorical
ACUTECHRON	A measure of level of pain	Categorical
TREATCAT	The treatment category for the patient	Categorical
DISCHARGE	The final discharge by the physiotherapist	Categorical
OUTCOME	The measuring outcome by the patient	Categorical
PSFS Initial	PSFS Initial score	Numerical
PSFS Final	PSFS Final score	Numerical
PSFS Change	PSFS difference between the Initial and Final	Numerical
Consultation in Episodes	Number of consultations	Numerical
PRACTITNER	The name of the physiotherapist	Categorical
Research Consent	Consent to use data for Research	Categorical
GDPR Consent	Consent for GDPR for clinic	Categorical

Table 3.1: The Physiotherapist Data Set

From the data set, there are many features that can be used to train the model. In all the cases of categorical data, there will need of encoding before training the models. The data will be analysed and missing values will be handled with appropriate methods.

3.2 Language Choice and Libraries

In the implementation of this dissertation, the chosen language used was Python [1]. Python is a high-level programming language which can be used for Machine Learning applications. Python has a range of data analysis and manipulation libraries in which can be used to complete the data processing. In the method, Numpy[2] and Pandas [3] were imported to the code for data analysis and manipulation. Numpy is a library used for array and matrix operations and Pandas is used for data structuring and row and table operations. The main library used to integrate Machine Learning models was Scikit-Learn [33], which included the function calls for many algorithms and data processing techniques. The final central library imported was Keras [4] as it supplied the Neural Network algorithms. These were the main libraries used in the python code that implemented the methodology.

3.3 Data Analysing

Data analysis is a key step in understanding the data which can give insights to the approach of data transformation and model hyper-parameters tuning. Through the process of reading in the data and examining the values of the features, there could be some insights to the process in which to clean, transformed, and encode the data. Using visualisation, it can be possible to see distributions of feature values and correlations between features.

Through manual inspection, it was observed that the data had multiple cases of matching values but entered with trailing characters, blank values, and null values. By displaying the unique values in each feature, it was made apparent that many values had leading or trailing characters which made the number of unique values larger. There were cases of lower and upper case entries which further increased the unique values. An example of this would have been for an occupation entry such as student:

- Value 1: " Student"
- Value 2: "Student"
- Value 3: "student"

[1] https://www.python.org/
[2] https://numpy.org/
[3] https://pandas.pydata.org/
[4] https://keras.io/

In the three examples, semantically they are the same but the machine interprets them as different values. This apparent higher number of values would cause an issue when encoding the data as each example would be encoded with a different value. This insight determined part of the cleaning process in the Data Transformation process.

The other noticeable value from the features were blanks and null values. There were a large number of blank values in each feature which were separate to null values. The blank values would usually be white space. From the null values, there were a few types of nulls that needed to be caught.

- Value 1: NaN
- Value 2: 'na'
- Value 3: 'n/a'

From the list, the first value is a null value which is a true empty value. The other two examples are equivalent null values represented by a string. Each of these cases will need to be handled in the Data Cleaning steps of the preprocessing stage.

Through the counting and displaying of the target variable 'OUTCOME,' a critical insight in the distribution is observed. By listing the count of each category in the OUTCOME feature, the distribution of each category can be seen from Table 3.2.

Category	**Count**	**Percentage of Total**
"missing"	43	2.0%
"return to patient desired activity level"	114	5.3%
"no goals achieved"	442	20.4%
"goals partially achieved"	637	29.4%
"all goals achieved"	930	42.9%

Table 3.2: Distribution of Target Variable OUTCOME

It can be seen that there is an uneven distribution in the feature. There is a high percentage in a single class, "all goals achieved". There is a gap in occurrences after the majority class, a 13.5% drop in the 'OUTCOME' feature. This imbalance distribution will need to be addressed, based on the research; Machine Learning models predict better with a balanced target variable.

From examining the correlations between different features a number of remarks could be seen. There were correlations between the target variable and numerical values. In the data set, there were trends between input features which both had contributions to the prediction of the target. It could be seen key features in positive effects on model prediction.

There was a correlation observed between PSFS Final and target variable OUTCOME. This correlation between PSFS Final and the target variable is a logical trend as the PSFS Final score determines the patient's final mobility score. As the PSFS Final score is closer to 30 (the maximum PSFS score) then the more positive the result of OUTCOME. This shows that the target variables are significantly impacted by the PSFS Final score. Although this is helpful in understanding the practices of the physiotherapist but is not beneficial to the Machine Learning process as the PSFS Final score is not a suitable input variable. PSFS Final score is not realistic to be used in prediction as the PSFS Final score would only be known at the same time as the target variables. This would be further described in future paragraphs.

The next trend in which gives knowledge into the features is between FIRSTRECUR and ACUTECHRON. Through observing the correlation between these variables it can be noted that there is a positive correlation. The correlation between FIRSTRECUR and ACUTECHRON is the 'recurrent' visit and the 'chronic' pain. It can be seen a high number of data points being both of those mentioned values. This is the same for 'first' and 'sub-acute' in FIRSTRECUR and ACUTECHRON features respectively. These pairs are logical trends as it can be assumed if a patient were to come to visit a physiotherapist clinic for a recurrent injury then it would be of a serious calibre and such having a higher level of pain. This can be said of the first injury and a lower pain level. The correlation between these data points can reveal the features to use in the Machine Learning algorithm.

For the target variable OUTCOME, there are a number of features that have a correlation that has a higher impact. The features are Treatment (TREATCAT), Diagnosis (DIAGNOSIS), First-Recurrent (FIRSTRECUR), Acute or Chronic pain (ACUTECHRON) and Practitioner (PRACTITNER). These features are critical in the prediction of the classification. These features would be the bare minimum to include in the input features list as they have an important impact on the prediction.

In the data set, there are a number of features that can be removed as they have no contribution to the prediction of the classes. These features are Client Number (CLIENTNO), Imaging, Research Consent, and GDPR Consent. In the case of the

consent features and client number, it is clear that they have no indication on the outcome of the patient from a logical standpoint. The Client Number is a randomly assigned number with no connection to a patient's circumstances and the consents are for clinic record only. In the case of Imaging, there was no correlation observed between the outcome of the patients. This observation could be reasonable as it would suggest that imaging does not contribute to the assessment of the patient's mobility, diagnosis, and treatment. There are another set of features in which were removed as they were not realistic in predicting the outcome of the patient at an initial consultation. These features could be viewed as labelled data in themselves. These features are DISCHARGE, PSFS Change, PSFS Final and Consultation in Episode. These four features are only recorded at the end of the treatment with the patient and the removal of these features will enable a realistic prediction of a new patient. These were then features that were removed in the Data Transformation stages.

3.4 Data Transformation

This following section describes the data transformation steps in which was applied to achieve the best prediction model. The data transformation stage includes preprocessing steps where the data was cleaned, missing values were handled, the transformation of features, and the removal of features. Included in the preprocessing steps was an intermediate encoding stage to facilitate the sampling stage. In the sampling stage, the data were re-sampled to balance the data. Once the data was re-sampled then the data split into training and testing data. The final stage of the data transformation was then encoding through the Target encoding method. This Data Transformation section details how the data was prepared for model training.

3.4.1 Preprocessing - Cleaning

The first preprocessing step in the data transformation was cleaning the data. In this first step, there are three stages, in which the first stage is handling the duplicate values. In this cleaning step, the data had a lot of duplicate values. These values, as mentioned with examples in the previous section had to be handled. The method in which was used to fix this was to remove leading and trailing white space and transforming all values into lower case only. This was done by looping through every column that was a categorical column and removing the unnecessary white space and transforming the characters to lower cases. This cleaned up the duplicate values.

The next stage of the cleaning process was to handle the missing values. There were a number of different null values. In the implementation, each null value was found and replaced with a suitable value. For the categorical data, all values were replaced with a 'missing' string. This 'missing' string would be an indicator to the Machine Learning algorithms of the blank values. This would aggregate all the missing values into a single category. For the numerical values, the null values were replaced with a -1. This was the case for Age, PSFS Initial, and PSFS Final. A –1 value is a suitable value as the range of these features are positive integers. By choosing a value outside the range, this will give a pattern to the Machine Learning model that this value is missing. In the case of PSFS Change, this feature's range is between [-30, 30], as the possible values are the difference between the PSFS values. This meant that a value of -1 would not be appropriate as it is within the range. The next best value would be the average of the PSFS Change values. The average was then filled into the missing values for PSFS Change. This is done in the assumption that an average value is the best value as it would be closet to a possible true value.

The following stage was obtaining the age feature of the patients. Through the features, Start Date of the consultation (STARTDATE) and Date of Birth (DOB), the age of the patient is calculated. The function takes the year of the start date and finds the difference between the year of the Date of Birth. This is how the age of the patient is calculated. In this approach, it can be assumed that either the start date or end date is suitable to be used. The assumption is that consultation rarely span multiple years and thus either feature can be used to calculate the age. In the implementation, the STARTDATE was used.

The final stage in the preprocessing stage is to remove the unnecessary feature column. The columns that were removed were Client Number (CLIENTNO), Imaging, Research Consent, GDPR Consent, Discharge of the patient (DISCHARGE), PSFS Change, PSFS Final, and Consultation in Episode. These 8 columns were removed with Pandas' Dataframe Drop function. This function is given a list of columns and removes them from the data set. The data is then completely cleaned and ready to be encoded.

3.4.2 Preprocessing - Label Encoding

The second step in the preprocessing step is encoding categorical features. This encoding phase is a small transformation that allows the proceeding sampling step to function correctly. In this phase, the data set is transformed with Label encoding. The Label encoding transformed all the categorical data into numerical values. Each

unique category in the categorical feature is given a numerical represented value. In the implementation, a custom function was written to do the Label encoding as the Scikit-Learn library for Label encoding was not able to handle unknown values. The implementation of this function was done by using the Scikit-Learn One-Hot encoding function which returned a One-Hot sparse matrix. With this matrix and at each data point, the index with the value of '1' was found. Using the value of this index, a new array was made and for each data point, the index value was assigned. This constructed a Label encoding of the features.

3.4.3 Sampling

The next step is the sampling stage which balances the data set. In this sampling step, the Label encoded data is taken as an input to the sampling technique. The sampling technique used was SMOTETOMEK which is the combined sampling technique. The output of the SMOTETOMEK balances the data set from evaluating the label data. The new distribution of the label data is displayed in Table 3.3. It can be seen that the count of each category is balanced and this new balanced data set will improve the model training process.

Category	**Count**
"missing"	925
"return to patient desired activity level"	920
"no goals achieved"	874
"goals partially achieved"	836
"all goals achieved"	817

Table 3.3: New Distribution of Target Variable OUTCOME

3.4.4 Splitting the Data

Once the data has been sampled into a balanced form, the data is split into training and testing data. To split the data into the two parts, K-folds was used as a method to complete this. K-fold was used to ensure that an unbiased accuracy score could be obtained. In the chosen method, the value of K equals 10 was selected. This meant that the data was split into 10 equal parts and trained and tested 10 times. The data was shuffled and split randomly into the 10 parts.

3.4.5 Target Encoding

Proceeding the splitting of data, the data is then transformed with Target encoding to give a better input to the Machine Learning models. The data is encoded with Target encoding which will encode the data with a meaningful numerical value. The target used was the label encoded values of OUTCOME of the patients. The range of the OUTCOME target was between 0 and 4 and the Target encoding encoded the features based on these OUTCOME target values. The output of the encoding completed the preprocessing stage and the data was ready to be used to train the models.

3.5 Model Training

For the Model Training process, the four models selected will be trained with the training data and tested with the testing data. The four models trained were Decision Tree, Random Forest, KNN, and Neural Network. At each fold, the models were trained and fitted with the training data. This will then created Machine Learning model that could predict the OUTCOME classification of a patient. The test set data was then feed into models and the models completed a classification prediction. The prediction was then compared to the true results and a test accuracy score was obtained for that K-fold. This score was then saved to then obtain an average accuracy score. To record the training accuracy, once the model was trained, the training data was passed into the model and predicted on at each fold. By outputting the training accuracy, it can be used to investigate the presence of underfitting and overfitting which could be seen by comparing training and test accuracy. Through this investigation, the models are evaluated and adjusted which is described in the next section.

For each model, the hyper-parameters are different and thus can be tuned before training. At the initial implementation, the models were constructed without any hyper-parameters input. This meant the default hyper-parameters for models were used (set by the library implementation). To obtain suitable hyper-parameters that would fit the data better, a Grid Search algorithm was implemented. In this Grid Search algorithm, a list of parameter values, model, and data were passed to it. The Grid Search tries every combination of parameter values with the model and data. The list of hyper-parameters tried for each algorithm are listed in Tables 3.4 - 3.6. The results are the best hyper-parameters values to use for this model and data. The Grid Search method was applied to the Decision Tree, Random Forest, and KNN models. The approach in tuning hyper-parameters for Neural Network was a separate

approach. From the Grid Search, the following hyper-parameters were selected for each model can be seen in Table 3.7.

Hyper-parameters Detail	**Hyper-parameters**
Max Depth	[20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
Min Sample Split	[6, 8, 10, 12]
Min Sample Leaf	[2, 3, 4, 5]

Table 3.4: Hyper-parameters search range for Decision Tree

Hyper-parameters Detail	**Hyper-parameters**
Max Depth	[20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30]
Min Sample Split	[6, 8, 10, 12]
Min Sample Leaf	[2, 3, 4, 5]
Number of Decision Tree	[100, 200, 300, 1000]

Table 3.5: Hyper-parameters search range for Random Forest

Hyper-parameters Detail	**Hyper-parameters**
Nearest Neighbours	[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16, 17,18,19,20,21,22,23,24,25,26,27,28,29,30]

Table 3.6: Hyper-parameters search range for K-Nearest Neighbours

Model	**Hyper-parameters**
Decision Tree	max depth=23, min samples split=6, min samples leaf=2
Random Forest	max depth=25, min samples split=6, min samples leaf=2, n estimators=300
KNN	n neighbors = 2

Table 3.7: Hyper-parameters used for Models

3.5.1 Neural Network

The same process of training was applied to the Neural Network and using K-folds to cross-validate the results. The Neural Network model took in the features as inputs and was feed into the network. The output of the Neural Network was a classification of the OUTCOME of the patients. For each fold, the training data set which was passed to the model was split into a training set and a validation set. The validation set was used to evaluate the Neural Network at each epoch which helps the overall training. The Neural Network was designed with the architecture displayed in Figure 3.1. An input layer with 12 input neurons, followed by 2 dense layers with 600 and 370 neurons respectively, which was then followed by a dropout layer. The following three hidden layers were dense layers followed by dropout with 200, 150, and 70 neurons respectively. A final dense layer with 40 neurons completed the hidden layers and the output layer was a softmax layer with 5 neurons. The dropout layers were applied to try and contest against overfitting. The Neural Network design was achieved through trial and error by running through a single training and test set. Once a reasonable attempt was made, the designed network was then run through K-folds algorithm. The model used a Stochastic Gradient Descent with a learning rate of 0.01, a batch size of 256, and was trained over 600 epochs.

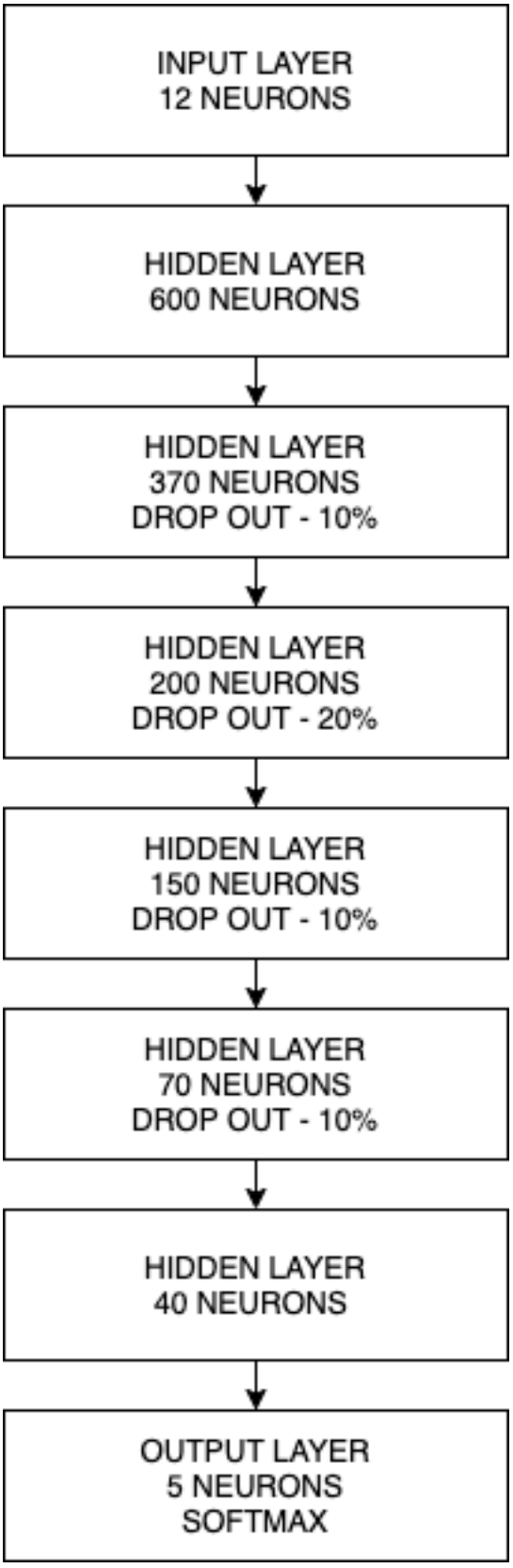

Figure 3.1: Neural Network Architecture

3.6 Model Evaluation

During the 10-folds the test data set is feed into models and the models would output predictions. The test data was not seen in the training step by any of the models which means the data was new for the models to predict on. The models were tested as it tried to predict on the data that it had not trained for. These output predictions were then compared to the true values. The number of correctly predicted values were counted. This number was then divided by the total number of predictions and a percentage accuracy score was obtained. This value was then saved and represented one accuracy score for a fold.

At the end of the 10-folds, the accuracy of each model at each run is then averaged. The test accuracy score at each run was saved to an array. The test accuracy were summed up and divided by 10 to obtain the average prediction accuracy of the model. This value was then displayed to be viewed for each model. Through displaying each value, each model's performance can be measured by the accuracy score and compared. The accuracy that is closest to 100% is the best model as 100% accuracy determines a perfect model. The models can then be ranked in terms of it performance. The results and models are then saved to disk on the local machine. This saved results and models can then be re-evaluated and compared to future attempts.

In evaluating the training accuracy (displayed during the training) and the testing accuracy, the models can be determined to have under fitted or over fitted the data. The training accuracy score was particularly helpful in observing underfitting. When the training accuracy was low, underfitting can be seen which would then indicate a need for a more complex model. When the training accuracy was sufficiently high then it can be determined that a suitable model is being used. To then validate the model, the testing accuracy was evaluated. Through the evaluation of the testing data, it can be concluded whether the model was overfitting. In the case of overfitting, the model parameter's would need to have been tuned. This would mean to re-run the Grid Search Algorithm in search of better hyper-parameters. The evaluation of the testing accuracy determined if there were more work to be done on the preprocessing of data, model selection, and model tuning. Any adjustments or implementations of Data Process/Machine Learning techniques were added or taken away from this flow based on the evaluation of the models.

This chapter goes into the detail of the method in which was taken to preprocess the data, train the algorithms, and measure the performance of the algorithms. The

methodology was conducted with Python and various libraries. The data was initially analysed to understand the data and gain insights into trends in the data set. These insights would then contribute to the data transformation. The physiotherapist data is a mixture of categorical and numerical data and a number of techniques were used to transform the data. The method started with preprocessing the data, where missing values were imputed, features calculated, and removal of unnecessary features. This was then followed by Label encoding which was a needed step to prepare for the sampling technique. The data was then balanced with sampling technique SMOTETOMEK. Once the data was balanced, the data was split into K-folds for cross-validation. The training set would be passed to the algorithms and the algorithms would be trained. The algorithms' hyper-parameters can be set before training. The values would be determined by the result of the Grid Search algorithm. In the case of the Neural Network, the training data was split further into training and validation set to improve the training. The algorithms would then be tested with the test set and the accuracy was saved. The overall model's accuracy score was calculated as an average from each K-fold. The models' accuracy can then be compared and reviewed.

4 Results & Discussion

The following chapter will report the results of the approach described by the methodology chapter. Included will be a discussion of the results and details of the key mechanisms behind how the results were achieved. The results were achieved through iterations in the experiments and the key mechanisms of sampling and Target encoding largely impacted the results. The results reported include the intermediate results that were improved on which eventually reach the final result. Through examining these intermediate results, the key mechanisms were identified. This chapter details the explanation of the key mechanisms' impact and describing alternative techniques that did not return a positive result. These alternative techniques would be in terms of other encoders and other approaches with data. The experiment will be discussed in relation to other strengths of the methodology and possible further implementations that could further improve the results.

4.1 Model Accuracy

The average accuracy score of each model is shown in Table 4.1. This accuracy is an average of 10-folds predicting on the testing data set. These were the results of the prediction of the best approach described in the Methodology.

In Table 4.1, it shows that the best model was the Random Forest model with an accuracy score of 77.69%. The Random Forest was the best model because of its bagging technique. In the Random Forest algorithm, the ensemble of Decision Trees was able to fit the data and not overfit on the noise in the data set. Accredited to the weakly correlated Decision Trees, the model was less affected by the noise in the data and produced better predictions. This reasoning would then validate Decision Trees being the lowest scoring model, as the model would be sensitive to the noise. The noise would then affect the performance of the prediction for the Decision Tree. The accuracy achieved is commendable with 67.35% but due to being a single Decision Tree, the model produced the lowest score. The Neural Network achieved the second high score which is shown in Table 4.1. The Neural Network was able to achieve a

high score as it was a suitably complex model. By comparing the training accuracy and the validation accuracy it can be seen that the model was overfitting as the training accuracy was high but the validation accuracy was flattening. Investigating the Neural Network, the main component as to why the model did not achieve as well as due to the low amount of data. For the training, there are approximately 2800 training points (after removing test and validation data sets) and this could be seen as not enough data for a Neural Network to perform at its optimal. Alternative techniques of overfitting could be used to improve the score but with the current model, the Neural Network is second best. The KNN was able to achieve an accuracy score of 73.87% which was completed with K=2. The success of KNN is likely due to the sampling technique of SMOTETOMEK. In SMOTETOMEK, Nearest Neighbours are used in the algorithm and the Tomek Links cleans up the data around the decision boundaries. This means the separation between class is clear and a K=2 Nearest Neighbour can predict moderately well.

Model	**Accuracy**
Random Forest	77.69%
Neural Network	74.64%
KNN	73.87%
Decision Tree	67.35%

Table 4.1: Average Accuracy of Machine Learning Models across 10 Folds

4.2 Key Machine Learning Mechanisms

The results in Table 4.2 and 4.3, are intermediate results during the exploration of different machine learning techniques. Table 4.2 is the results of data processing described in the methodology without any SMOTETOMEK and Target encoding. The only encoding applied was One-hot encoding. Table 4.3 is the same data preprocessing but including sampling technique SMOTETOMEK. This results are without any Target encoding and is still using One-hot encoding.

Model	Accuracy
Random Forest	48.8%
Neural Network	42.93%
KNN	43.26%
Decision Tree	44.46%

Table 4.2: Average Accuracy of Machine Learning Models across 10 Folds without sampling and Target encoding

The results were achieved through key mechanisms that had a positive impact on accuracy. The scores presented in Table 4.1 was reached after the sampling technique of SMOTETOMEK and Target Encoding was applied to the data. These two techniques can be seen as key mechanisms in the Machine Learning approach. In the first iteration of experiments, without sampling and Target encoding, the accuracy score was between 43% and 50% (Table 4.2). This was the method with the cleaning process described in the methodology and using K-folds for unbiased accuracy scoring. The encoding applied to the features was a One-hot encoding at that phase of the experimenting. The first key mechanism to be applied was SMOTETOMEK which outputted the sampled data before feeding it to the Machine Learning models. Once this technique was applied, the new results from the models were in the range of 62% and 72% (Table 4.3). This was a significant impact which identified SMOTETOMEK as a key technique. The success of SMOTETOMEK on improving the prediction score was due to the re-balancing of the imbalanced nature of the data. As SMOTETOMEK transformed the data to balance data, the Machine Learning models were then able to predict more accurately.

Model	Accuracy
Random Forest	71.82%
Neural Network	49.33%
KNN	66.56%
Decision Tree	62.10%

Table 4.3: Average Accuracy of Machine Learning Models across 10 Folds without Target encoding

The other key mechanism discovered in the experimentation is the Target encoding technique. Proceeding from the addition of SMOTETOMEK, the Target encoding was

implemented after the splitting of the data. The data was then Target encoded and passed to the model training. Upon applying Target encoding, the accuracy score increased to a range between 64% and 76%. The improvement of the accuracy score shows the impact of Target encoding and it can be determined that Target encoding is the most suitable encoding technique. The effect of Target encoding on the accuracy is due to the encoding giving a meaningful represented value. The categorical values are given a value that is insightful to the labelled data and enables the models to be trained better. This significant increase in accuracy can then determine that Target encoding is an important technique for predicting on this physiotherapist data set.

The model tuning through the Grid Search algorithm helped increase the model accuracy to the final results. The Grid Search algorithm was used with the processed data to find a suitable model hyper-parameters that would fit the data best. The Grid Search returned the values show in Table 3.7 and the model results are the accuracy in Table 4.1. Model tuning gave a small amount of improvement after SMOTETOMEK and Target encoding. This step is still important even with a small impact. The Grid Search helps improve the model's accuracy as the model's hyper-parameters are set at the creation of the model. These hyper-parameters are set at a default value set by the Python libraries and thus the Grid Search algorithm can be used to test different values. The results gave hyper-parameters that are best suited to the data set (hyper-parameters referred in Table 3.7). If the training data was different then different hyper-parameters would be produced by Grid Search and therefore it can be concluded that if more different training data was added then the hyper-parameters would need to be re-tuned. If the data is not different in trends and correlation but just additional, then these hyper-parameters would likely be suitable.

In Table 4.3, which are accuracy results for the models without Target encoding, it can be noted that the Neural Network did significantly worse than the other models. The Neural Network was only able to achieve 49.33% while the other models were able to achieve +60%. From investigating, this result was caused by underfitting of the Neural Network. The model was noted to be underfitting as the training accuracy would not increase any further. This would mean that the Neural Network model would need to more complex. From inspecting the Neural Network design (Figure 3.1), it could be argued that it is already a complex network but the results based on Table 4.3 had an input vector of 395 features. This meant the input layer for the Neural Network corresponding to Table 4.2 and 4.3 has 395 input neurons compared to the 12 input neurons from Fig 3.1. This difference in input layers is caused by the different encoding methods, as the encoding methods output different size feature

vectors. The 395 input neurons are caused by the One-hot encoding, where 12 input neurons are due to Target encoding. This is the reason for the network not being complex enough to capture the pattern in the data. The first layer has only 600 neurons which is suspected to be too small to capture the features optimally. To increase this value, the first solution is to remove the underfitting and increase the complexity of the Network. Upon doing a quick single fold, a model with increase neurons (increasing 1st hidden layer to 900 neurons) the training was above to achieve an improved result of 56%. Once an improved result is achieved, then solutions to mitigate overfitting might be required depending on the training process.

4.3 Neural Network Model Loss & Accuracy Graphs

The graphs 4.1 to 4.10 are related to the results produced on Table 4.1 for the Neural Network. Each pair of graphs relate to a Kth fold from the K-fold algorithm. Each graph has a sub-graph, the sub-graph on the left is the Model Loss graph which plots the training loss value and the validation loss value at each epoch. The sub-graph on the right is the Model Accuracy graph which is similar to the Model Loss graph but with accuracy. In each graph, the blue line represents the training loss/accuracy (calculated from the training set) and the orange line represents the validation loss/accuracy (calculated from the validation set). These values are obtained after the forward propagation of the Neural Network and with respective data sets. These models were run for 600 epochs. These graphs show the training process for the Neural Network. In these graphs, it can be seen how at each epoch the training improves as the weights of each neuron is adjusted by Stochastic Gradient Descent. The levelling off of the curves show that the Neural Network is converging. From these graphs, it can be identified when underfitting or overfitting occurs. In all cases, the training losses reach below 0.25 and the validation losses finish at approximately between 1 and 1.5. The Model Loss graphs show a steady decrease in training losses but the same cannot be said for the validation losses. In the validation losses, there is a steady decrease until the 150 epoch where the curve starts to increase again. In the Model Accuracy graphs, the training accuracy improves steadily and obtains an approximate accuracy of 0.9+. For the validation accuracy, there is also a steady increase but diverges and flattens earlier than the training accuracy. The validation accuracy diverges at roughly 240 epochs and reaches high 0.7.

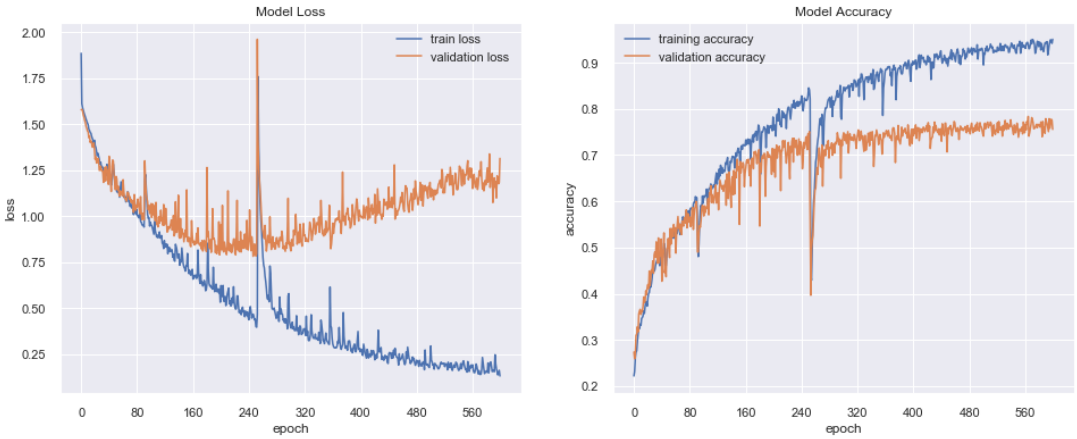

Figure 4.1: Model Loss and Accuracy Graph for 1st Fold

Figure 4.2: Model Loss and Accuracy Graph for 2nd Fold

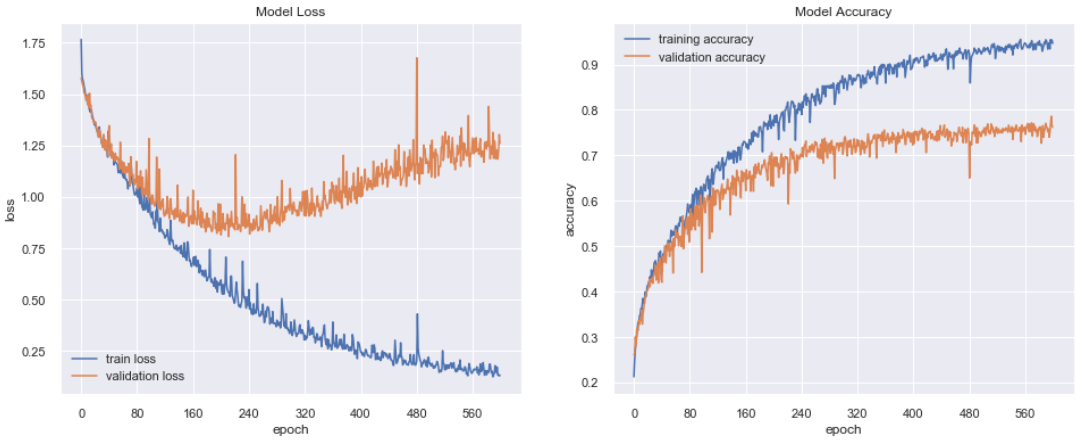

Figure 4.3: Model Loss and Accuracy Graph for 3rd Fold

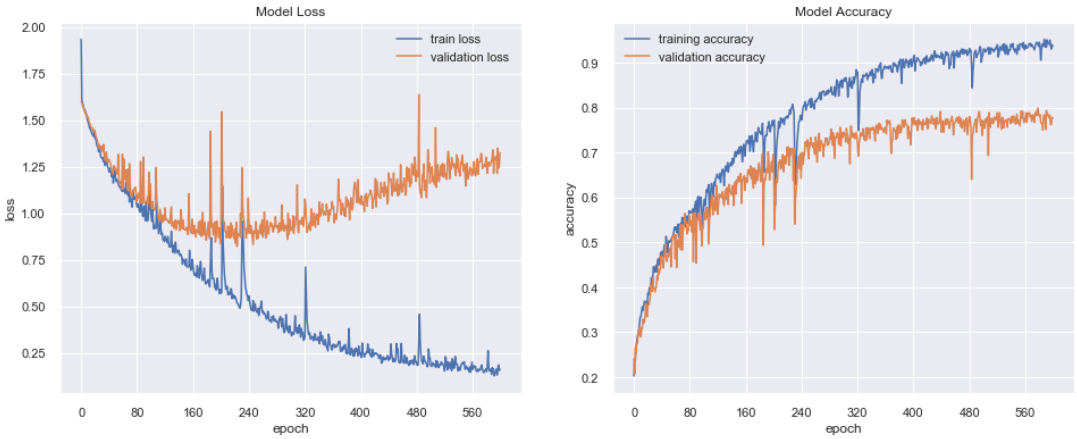

Figure 4.4: Model Loss and Accuracy Graph for 4th Fold

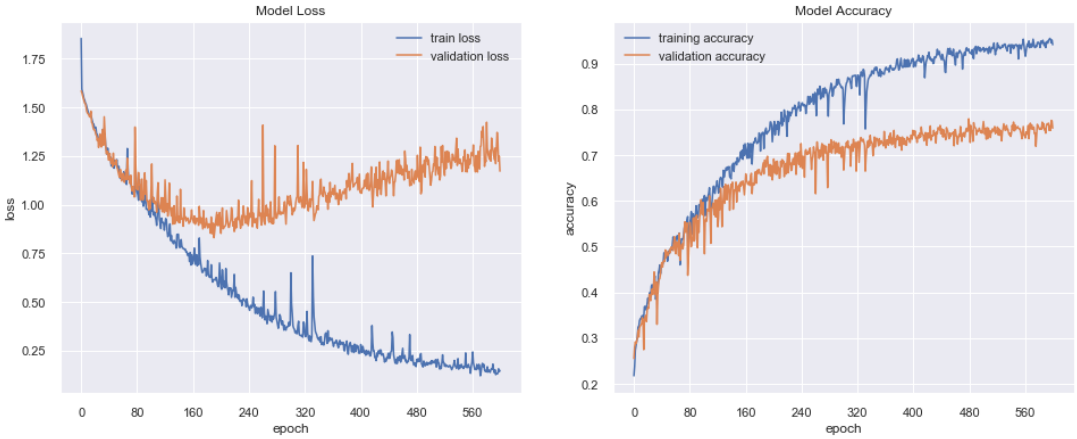

Figure 4.5: Model Loss and Accuracy Graph for 5th Fold

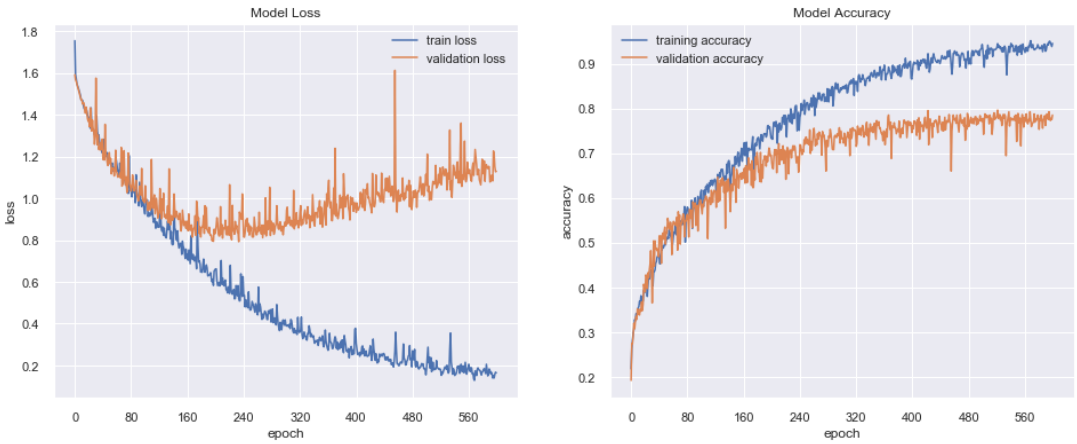

Figure 4.6: Model Loss and Accuracy Graph for 6th Fold

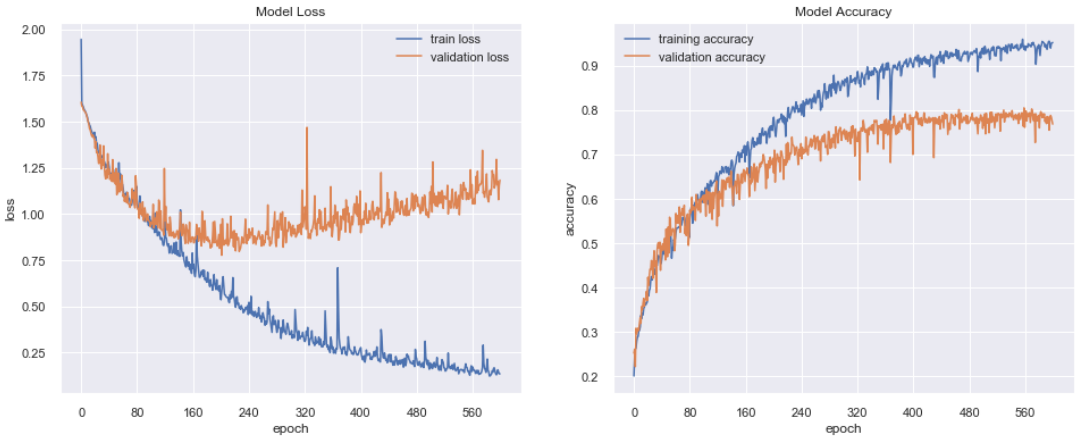

Figure 4.7: Model Loss and Accuracy Graph for 7th Fold

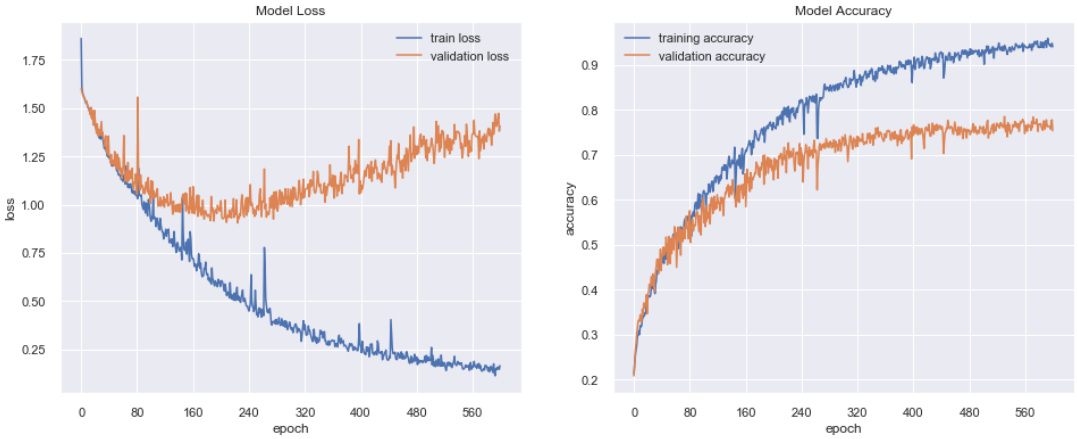

Figure 4.8: Model Loss and Accuracy Graph for 8th Fold

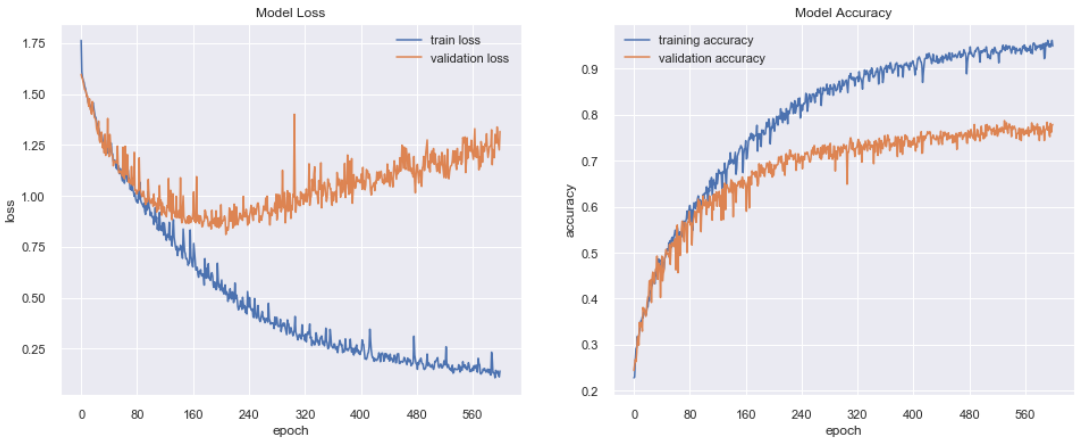

Figure 4.9: Model Loss and Accuracy Graph for 9th Fold

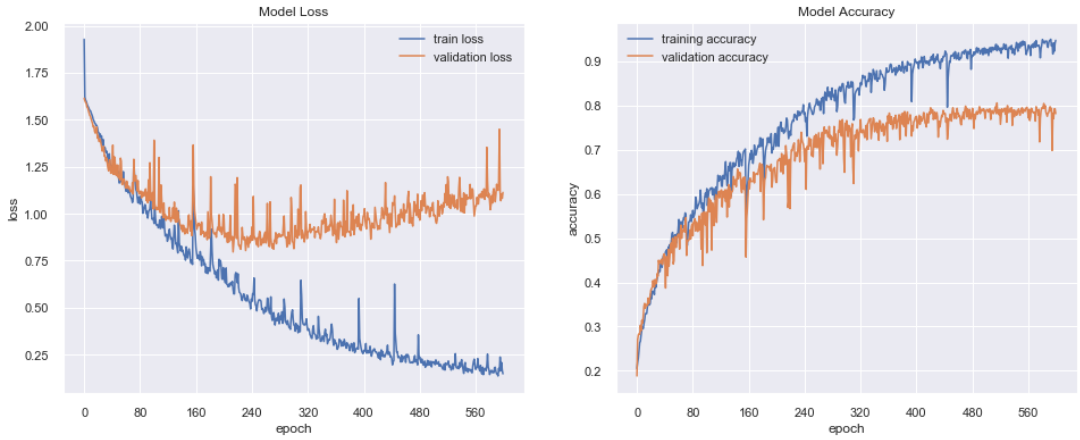

Figure 4.10: Model Loss and Accuracy Graph for 10th Fold

The graphs of the Neural Network training (Figures 4.1 - 4.10), shows how the Neural Network was trained to obtain the results on Table 3. Across all the graphs, it can be noted that there is a good curve for both the Model Loss and Model Accuracy. The Model Loss curve starts with a steep decrease and at a steady rate. The curve eventually starts to level off and the rate of change decreases. For the training loss curve, the values continue to decrease but the validation does not and starts to rise in most cases. This pattern in the training loss curve shows that the Neural Network is able to fit to the training data, which means the model is not underfitting. When the validation loss curve starts to increase, it illustrates that the model is overfitting as the training loss is decreasing but the validation loss does not decrease any further. The same evidence can be seen in the Model Accuracy graphs. Across the majority of the graphs, both training and validation accuracy increases with a steep rate but eventually starts to level off. At approximately 160 epochs, the training and validation accuracy values diverge. This shows the beginning of the Neural Network model overfitting in the training process. From the graphs (Figures 4.1 - 4.10), it can be used as feedback to understand the training process of the Neural Network. By analysing the curves, it can be seen that overfitting is prevalent and that some solutions to mitigate this is needed. One solution to solve the overfitting of the Neural Network is using more Regularisation. This will stop the model from fitting to the noise in the training data and be trained more generically. Another possible method to lower the overfitting is to use more drop out layers. Drop out layers will create the effect of more noise in the data and allow the model to train more generically. One final solution which could be used is by stopping the training early. This method means to train the model up to or before it starts to overfit. This will then lower the chance for the model to overfit.

4.4 Encoders and Synthetic Data

Through the experiment, a number of different techniques were applied but were replaced or removed as they were not effective. The first set of techniques that was replaced was the encoding method. The final encoding method used was Target encoding on all the features, where the initial investigation used a range of encoders from Label encoding, One-hot encoding, and Target encoding. These were replaced in the final encoding method. One method which was tested but removed was creating synthetic data. This method was to use a Python library to help create more data to then allow for a better prediction from the models. This was removed as the Python library used did not create similar data to the original data set and had no impact on the predictions.

For the approach in selecting the best encoding method, Label, One-hot and Target encoding were tested. In the investigation, Label encoding and One-hot encoding were the first iterations of the Machine Learning pipeline. It was seen that the effect of Label encoding was weaker than One-hot encoding. The models were able to predict better when One-hot encoding was applied over Label encoding. This was the same for Target encoding relatively to One-hot encoding as previously described Target encoding was the strongest method. The reason for Label encoding being not as strong as One-hot encoding is that there is no meaningful relationship between the encoding and the categorical values. The encoding from Label encoding does not give any encoded information about the category and the encoded values can be randomly assigned. In Label encoding, there is also the weakness of accidental ordering. The encoded value might train the model in a way that causes the model to have an order of importance for different categorical values. If there is no ordering or ranking of importance then this makes the training of the model more difficult and a weaker prediction result. This is why Label encoding was the weakest method. In examining One-hot encoding, the returned sparse matrix gives the models an input that can be used better. It is better as One-hot encoding does not allow any bias of order because of the sparse matrix. The One-hot encoding treats all categories equally and the model can be trained from the data in a more accurate way. With the One-hot encoding method, the categories in the data set are represented clearly without any bias but do not give any categorical meaning. This is where Target encoding is superior to One-hot encoding as Target encoding encodes the categories with a target meaning i.e. the classification class of the data. Target encoding works over the other two methods as it does not introduce any order or bias and gives meaning to the encoded categories.

The method of creating synthetic data was implemented but ultimately removed as it had no contributing impact on the model's prediction output. During the initial data analysing, it was noted that the data has a low number of samples, approximately 2000. One way in which was attempted to increase this number was creating synthetic data. A Python library was used called DataSynthesizer which was developed by Ping et al. [34, 35]. This library was able to produce new synthetic data based on the original data. In applying this library, an extra 4000 samples were produced. This data was then feed through the described method and results were produced. The performance of the model was slightly worse than the results in Table 4.1. With the additional data, it would have been expected that an increase in accuracy from the models could be achieved as more data would help train the models better. In this case, it was not true and the cause of the lower than expected performance was that the synthetic data was not true to the original data. The reason for the lower performance was that the Python library generated data which was similar to the original data set but with some added noise. This noise in the synthetic data did not contribute positively to the prediction models and had an impact where the model's accuracy was lower. This meant the method was not effective and removed from the methodology.

4.5 Further Improvements

Several possible techniques that could have been applied to improve the results but were not applied due to time constraints. The approaches in which could have a significant impact on the prediction was in removing outliers from the data set, increasing the data set size either through synthetic data or gathering more data. Even though the technique of creating synthetic data was tested, using a different library or approach could bare improve results.

In the data set, there are a number of small occurring values in Activity Days and Activity Levels which can be seen as outliers. These values in Activity Days and Activity Levels could be adding noise to the model prediction. These noise values can then make it hard for the model to train on the data and worsen the prediction accuracy. The Machine Learning models could be improved if the outliers in the data set were removed in both Activity Days and Activity Levels. Although this approach could worsen the result even further as the number of data points is low and removal of outliers will lower this number further. There could be possible negative effects in lower data examples which is not ideal and it would be more desirable to retain all data points. The method of removing outliers can be a viable solution if there were more data points.

Another solution in increasing the data set in aid to better train the Machine Learning models. As data is key to Machine Learning algorithms in recognising the patterns in data. This is a generally accepted approach in Machine Learning and has proven to increase the prediction of Machine Learning models. With the high number of samples, the model will be able to train better to the data. As previously discussed, this approach was attempted with using Python library but did not produce positive results. This was due to the poor synthetic creation as the purpose of the imported library was to anonymous the original data and not centre around the creation of new data. One solution to creating more data is searching for a better Python library which is centred on creating data for Machine Learning models. This will then create synthetic data that is closely matched to the original. If in the case that another library can not be found that suits the problem, then it would be viable to create a Machine Learning model in which creates the synthetic data. This model would be trained on the original data set and create new data. This can be done through one feature at a time and thus creating a cascading synthetic data creation. The output of one model will then be used as an input for the next synthetic data model. This will then create synthetic data that is based on the original data. This approach would need to be tested as the models will be created might be inaccurate and this can affect the model in predicting the outcome. To test the synthetic data creation, it would be vital to have a set of data that is a test set, which is unseen to the models. The addition of new data through acquiring or synthesising would vastly improve the model's performance, especially in the case of the Neural Network which would improve significantly more.

5 Recommendations

Through the experiment, many insights were obtained which can be made into recommendations to the physiotherapist clinic in their method of collecting data. The first insight from the data set was the quality of the data which leads to recommending standardising data entries. From the data processing, there were a number of unique values which semantically had the same representation. These values could have arisen because of physiotherapist entries into the system. The other possible causes could be a flaw in the digital system while entering values or during the output process where the values were incorrectly formatted. One recommendation for the physiotherapist clinic is to standardise the input values for every feature. This is a necessary need as there are different practitioners who enter into the system and to ensure a higher data quality the data should be succinct. In the case of the system being at fault, a report could be given to the software provider to update the outputting process, in the removal of syntactically different values.

The second insight is from the evaluation of the models' accuracy and the recommendation to continue collecting more data. To contribute to improving machine learning models, more data is essential. In having more data, there are more examples of treatment and outcomes. This will then capture more cases that represent the full sets of possible classifications. A better represented training data set will then produce a better trained model which can classify better on the patient's outcome.

The third recommendation is to investigate new data features that could be added to the features list. From the features, there were a number of features that had moderate correlations but no feature that has a strong correlation (to the outcome variable). The clinic could continue to investigate other possible data about the patient which could contribute to the success or fail of the patient's treatment. One suggestion could be some data on how the patient applied their treatment at home. This feature could indicate the patient's motivation or attitude in taking the treatment which could be a feature with a strong correlation. This investigation can only be

done by the physiotherapist as they have the expert domain in the field. The practitioners have expert insight to identify what data is vital. These are the recommendations that I would suggest to the physiotherapist clinic based on my exploring of the data and applying it to Machine Learning models.

6 Conclusion

Machine Learning is a powerful tool which can be applied to a wide range of problems and add a multitude of benefits. In this research, the most suitable Machine Learning algorithm in classifying the outcome of a patient for a physiotherapist data set was found. The Machine Learning model with the highest accuracy was the Random Forest with an accuracy score of 77.69%. The Random Forest was able to perform the best as it was an ensemble of Decision Trees and it was able to fit best to the data. The Random Forest achieved a high score as it was not sensitive to the noise in the data which was accredited to the low correlated ensemble of Decision Trees. The key mechanisms which were recognised to have a significant impact on the preprocessing stages were SMOTE with Tomek Links (SMOTETOMEK) and Target encoding. SMOTETOMEK was essential as it balanced the data set which had majority classes that made the data hard to train with. Target encoding had a significant impact as it encoded the categorical features with meaningful numerical representations based on the OUTCOME feature. This made the data easier to train and enabled the Random Forest to achieve a higher accuracy score. These were the significant findings from this dissertation.

The overall research included a number of concepts in which contributed to the achievement of the research goal. The overall concept of Machine Learning was explored from Supervised Learning, Overfitting, and Underfitting. A number of data processing techniques were examined and evaluated. The processing techniques related to cleaning and data transformation, which were necessary steps to prepare the data for the Machine Learning algorithms. The cleaning steps involved were techniques on replacing missing data and removal of data features. In regards to data transformation, the encoding methods of categorical data such as Label encoding, One-hot encoding, and Target Encoding were investigated. The final data processing techniques which were researched were sampling which included undersampling and oversampling techniques. Synthetic Minority Oversampling Technique was thoroughly investigated. The Machine Learning models of Decision Trees, Random Forest, K-Nearest Neighbours, and Neural Networks were studied and reviewed.

These concepts were examined in a theoretical and practical view. The algorithms' process were explained and several examples of the algorithms applied to the medical field were recorded. All these techniques were applied to the methodology and results were produced. These results were then discussed and a number of key mechanisms could be concluded. Further inspections could be drawn from the results and suggested improvements were identified. A number of recommendations for the physiotherapist were also drawn from the work of the research.

The main suggested improvement which would have a high contributing factor would be to gather more data. In Machine Learning models, the quantity and quality of the data are essential for the success of a model. The first suggested improvement is to gather more data in hopes to improve the classification accuracy of the models. Having more data will allow the models to train with more examples and be able to recognise the patterns in the data more efficiently and accurately. This suggested improvement strategy also overlaps with the recommendation for the physiotherapist clinic. It is recommended to the physiotherapist clinic to collect more data in the aid to improve the predictions of the models. Another recommendation is to ensure the quality of the data as there was a high demand for cleaning the data. The recommendations was to standardise the data entries through formal standards or updates in the software. The final recommendation were to continue to find strongly correlated features in which could significantly improve the prediction of the patient's outcome. Through the research, these numbers of recommendations could be drawn.

Through the application of Machine Learning as a prediction tool, benefits could be given to the physiotherapist and the patients in the overall treatment process. This application can help physiotherapist assess the overall outcome of the patient before the start of the treatment. This will then enable the practitioner to identify high-risk patients in which are in need of extra attention. Practitioners can also use this model to prepare their patients for the treatment and overall progress. This Machine Learning prediction tool can then assist the physiotherapist and give benefits to the patient in care.

Bibliography

[1] Divya Tomar and Sonali Agarwal. A survey on data mining approaches for healthcare. *International Journal of Bio-Science and Bio-Technology*, 5(5):241–266, 2013.

[2] Michael G Bechtel, Elise McEllhiney, Minje Kim, and Heechul Yun. Deeppicar: A low-cost deep neural network-based autonomous car. In *2018 IEEE 24th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, pages 11–21. IEEE, 2018.

[3] Sam Maes, Karl Tuyls, Bram Vanschoenwinkel, and Bernard Manderick. Credit card fraud detection using bayesian and neural networks. In *Proceedings of the 1st international naiso congress on neuro fuzzy technologies*, pages 261–270, 2002.

[4] Igor Kononenko. Machine learning for medical diagnosis: history, state of the art and perspective. *Artificial Intelligence in medicine*, 23(1):89–109, 2001.

[5] Isaac R Galatzer-Levy, Karen-Inge Karstoft, Alexander Statnikov, and Arieh Y Shalev. Quantitative forecasting of ptsd from early trauma responses: A machine learning application. *Journal of psychiatric research*, 59:68–76, 2014.

[6] U Srinivasulu Reddy, Aditya Vivek Thota, and A Dharun. Machine learning techniques for stress prediction in working employees. In *2018 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, pages 1–4. IEEE, 2018.

[7] Patcharaporn Panwong and Natthakan Iam-On. Predicting transitional interval of kidney disease stages 3 to 5 using data mining method. In *2016 second Asian conference on defence technology (ACDT)*, pages 145–150. IEEE, 2016.

[8] Real Carbonneau, Kevin Laframboise, and Rustam Vahidov. Application of machine learning techniques for supply chain demand forecasting. *European Journal of Operational Research*, 184(3):1140–1154, 2008.

[9] Maxwell W Libbrecht and William Stafford Noble. Machine learning applications in genetics and genomics. *Nature Reviews Genetics*, 16(6):321–332, 2015.

[10] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

[11] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] Mark JJP Van Grinsven, Bram van Ginneken, Carel B Hoyng, Thomas Theelen, and Clara I Sánchez. Fast convolutional neural network training using selective data sampling: Application to hemorrhage detection in color fundus images. *IEEE transactions on medical imaging*, 35(5):1273–1284, 2016.

[13] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. Springer series in statistics New York, 2001.

[14] Andriy Burkov. *The hundred-page machine learning book*. Andriy Burkov Quebec City, Can., 2019.

[15] My Chau Tu, Dongil Shin, and Dongkyoo Shin. A comparative study of medical data classification methods based on decision tree and bagging algorithms. In *2009 Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing*, pages 183–187. IEEE, 2009.

[16] Nils J Nilsson. Learning machines. 1965.

[17] Frank Rosenblatt. Principles of neurodynamics. perceptrons and the theory of brain mechanisms. Technical report, Cornell Aeronautical Lab Inc Buffalo NY, 1961.

[18] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[19] Andrew Gelman and Jennifer Hill. *Data analysis using regression and multilevel/hierarchical models*. Cambridge university press, 2006.

[20] Daniele Micci-Barreca. A preprocessing scheme for high-cardinality categorical attributes in classification and prediction problems. *ACM SIGKDD Explorations Newsletter*, 3(1):27–32, 2001.

[21] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16:321–357, 2002.

[22] Peter Hart. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3):515–516, 1968.

[23] Maciej A Mazurowski, Piotr A Habas, Jacek M Zurada, Joseph Y Lo, Jay A Baker, and Georgia D Tourassi. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural networks*, 21(2-3):427–436, 2008.

[24] Ivan Tomek et al. Two modifications of cnn. 1976.

[25] Gustavo EAPA Batista, Ana LC Bazzan, Maria Carolina Monard, et al. Balancing training data for automated annotation of keywords: a case study. In *WOB*, pages 10–18, 2003.

[26] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[27] Sahibsingh A Dudani. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, (4):325–327, 1976.

[28] Punam Mulak and Nitin Talhar. Analysis of distance measures using k-nearest neighbor algorithm on kdd dataset. *International Journal of Science and Research*, 4 (7):2101–2104, 2015.

[29] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533–536, 1986.

[30] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[31] Heba Ayeldeen, Olfat Shaker, Ghada Ayeldeen, and Khaled M Anwar. Prediction of liver fibrosis stages by machine learning model: A decision tree approach. In *2015 Third World Conference on Complex Systems (WCCS)*, pages 1–6. IEEE, 2015.

[32] Ch Anwar Ul Hassan, Muhammad Sufyan Khan, and Munam Ali Shah. Comparison of machine learning algorithms in data classification. In *2018 24th International Conference on Automation and Computing (ICAC)*, pages 1–6. IEEE, 2018.

[33] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[34] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthesizer: Privacy-preserving synthetic datasets. In *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, pages 1–5, 2017.

[35] Haoyue Ping, Julia Stoyanovich, and Bill Howe. Datasynthezier github. `https://github.com/DataResponsibly/DataSynthesizer`. Accessed: 2020-01-02.