Athlete Health Prediction Using Machine Learning Methods

by

Md Raihan Sharif

PhD, Nanyang Technological University, Singapore, 2008

MPhil, The Hong Kong University of Science and Technology, 2001

BSc, Bangladesh Institute of Technology, Rajshahi, 1995

A thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Applied Science

in the Program of

Electrical and Computer Engineering

Toronto, Ontario, Canada, 2019

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my dissertation may be made electronically available to the public.

Athlete Health Prediction Using Machine Learning Methods

Master of Applied Science 2019

Md Raihan Sharif

Electrical and Computer Engineering

Ryerson University

Abstract

Due to an increase in sports activities, the prediction of athletes' health (AH) has recently become an important research topic. However, it is a challenging task to predict AH because of the nature of the data and the limitations of predictive models. The main objective of this work is to develop appropriate models that can forecast AH using historical data. This work will enable sport organizations to monitor the well-being of their athletes.

In this thesis, we explore the applicability of various machine learning (ML) methods for predicting AH. Traditional ML methods do not perform well for class-imbalanced data as these methods are biased towards the majority class. In this work, we propose to use ensemble-based methods which utilize downsampling, bootstrap sampling, and boosting techniques to improve the classification performance. Various metrics are used to evaluate and to compare the model performance. Our results show the superiority of ensemble-based methods over traditional approaches. The random forest and the RUSBoost classier models are in particular found to produce the best performance in handling imbalanced classes.

Acknowledgements

First, I would like to thank my supervisor, Prof. Xiao-Ping Zhang, for his support throughout my MASc study. It would have been difficult for me to finish the thesis without his help and encouragement. I have also been benefitted from his course and other technical knowledge.

I would also like to thank the members of my thesis committee for their insightful comments to improve my thesis.

Special thanks are given to the Yeates School of Graduate Studies for the QEII-GSST award as a financial support during my research.

I also want to thank my fellow lab-mates as well as my friends at Ryerson University. Special thanks are due to Lidong, Dr. Tuan, and Bob Monro for their help.

Finally, I would like to thank my parents, son, and many others for their love and unconditional support.

Contents

Declaration ii

Abstract iii

Acknowledgements iv

List of Tables ix

List of Figures xi

1 Introduction **1**

1.1 Introduction and Motivation 1

1.2 Literature Review 2

1.3 Research Objectives and Contributions 6

1.4 Organization of the Thesis 7

2 Data Exploration and Preparation for Machine Learning **9**

2.1 Data Exploration 11

2.2 The Purpose of Data Cleansing 11

2.3 Dataset and Feature Description 12

2.4 Missing Values Treatment 13

2.5 Choosing the Right Interpolation Technique 14
2.6 Outliers Detection and Treatment 16
2.7 Generating Health Index or Output Label 17
2.8 Handling Imbalanced Classes 19
2.8.1 Oversampling 22
2.8.2 Undersampling 22
2.8.3 Synthetic Data Generation 23
2.9 Underfitting and Overfitting Problems 24
2.10 Cross-validation to Avoid Overfitting 25
2.11 Pruning to Reduce Overfitting 26
2.12 Model Tuning 26
2.13 Summary 27

3 Statistical Analysis for Feature Selection and Evaluation Metrics 28

3.1 Statistical Analysis for Feature Selection 28
3.1.1 Correlation Matrix 29
3.1.2 t-Value and p-Value 30
3.2 Performance Evaluation of Machine Learning 35
3.3 Summary 38

4 Athlete Health Prediction Using Traditional ML Methods 39

4.1 Introduction 39
4.2 Problem Formulation 40

4.3 Traditional Machine Learning Methods 41
4.3.1 Athlete Health Prediction Using Logistic Regression 41
4.3.2 Athlete Health Prediction Using Naive Bayes 42
4.3.3 Athlete Health Prediction Using k-Nearest Neighbor 43
4.3.4 Athlete Health Prediction Using Artificial Neural Network 44
4.3.5 Athlete Health Prediction Using Support Vector Machine 46
4.3.6 Athlete Health Prediction Using Decision Tree 47
4.4 Parameter Tuning for ML Methods 48
4.5 Model Evaluation Results and Analysis 50
4.6 Summary and Discussion 55

5 Improving Classification Performance Using Ensemble Methods 57

5.1 Athlete Health Prediction Using Boosting Algorithms 58
5.2 Athlete Health Prediction Using Random Forest Classifier 60
5.3 Parameter Tuning for Ensemble Methods 62
5.4 Combined Results and Discussion Considering All Features 67
5.5 Evaluation Results After Eliminating Insignificant Features 71
5.6 Evaluation Results After Removing the Health Feature 75
5.7 Performance Evaluation Results Using the ARMA Model 75
5.8 Training Time 82
5.9 Summary and Discussion 83

6 Conclusions and Future Work 85

6.1 Conclusions 85

6.2 Future Work 87

List of Abbreviations **88**

Bibliography **102**

List of Tables

2.1 Features and their respective ratings. 13

2.2 Seven omit days output labels for a single user. 18

3.1 Correlation matrix. 30

3.2 Descriptive statistics and p-values for all features. 31

3.3 Descriptive statistics and p-values for ten features. 32

3.4 Descriptive statistics and p-values for nine features. 33

3.5 Descriptive statistics and p-values for eight features. 33

3.6 Descriptive statistics and p-values for seven features. 34

3.7 Descriptive statistics and p-values for six features. 34

3.8 Descriptive statistics and p-values for five features. 35

3.9 Example of a confusion matrix. 36

3.10 Performance evaluation metrics. 37

4.1 Parameters for experiments and the optimal parameters for each model. 49

4.2 Performance measures for various ML models (seven omit days). 50

4.3 Performance measures for various ML models (fourteen omit days). 51

4.4 Regression equations for LR models for different omit days. 55

4.5 Bias vectors and weight matrices for ANN algorithm. 56

5.1 Parameters for experiments and the optimal parameters for each ensemble model. 66

5.2 Comparative analysis of various ML models (seven omit days). 68

5.3 Comparative analysis of various ML models (fourteen omit days). 68

5.4 Evaluation results for various models after removing the health feature. 76

5.5 Descriptive statistics and p-values for all features. 78

5.6 Descriptive statistics and p-values for all features. 79

5.7 Evaluation results using the ARMA model (seven omit days). 80

5.8 Regression equations for the LR models for different features. 81

5.9 Training time (in sec.). 82

List of Figures

2.1 A simple flowchart of machine learning. 10

2.2 Comparison of various interpolation techniques. 15

2.3 Histogram of nutrition, sleep, irritability, and hydration features for all athletes. 20

2.4 Histogram of stress, rest, energy, and soreness features for all athletes. 20

2.5 Histogram of health, enjoyment, and exertion features for all athletes. 21

2.6 Examples of underfitting (left), balanced (middle), and overfitting (right) models. 24

3.1 ROC curves showing two test results. 38

4.1 An example of a neural network structure. (Credit: TeXample.net) 45

4.2 ROC curves for various ML models (seven omit days). 52

4.3 ROC curves for various ML models (fourteen omit days). 53

4.4 Comparison of AUC for different omit days. 53

4.5 Comparison of F-score for different omit days. 54

5.1 An illustration of random forests. (Credit: William Koehrsen) 61

5.2 Impact of number of cycles on the classification error for AdaBoost. 63

5.3 Impact of number of cycles on the classification error for LogitBoost. 64

5.4 Impact of number of cycles on the classification error for RUSBoost. 64

5.5 Impact of number of trees on the classification error for RF. 65

5.6 Impact of number of features on the classification error for RF. 66

5.7 ROC curves for various models (seven omit days). 69

5.8 ROC curves for various models (fourteen omit days). 69

5.9 Comparison of AUCs for different omit days. 71

5.10 Comparison of F-scores for different omit days. 72

5.11 Model comparison results in terms of AUC. 72

5.12 Model comparison results in terms of F-score. 73

5.13 Impact of number of features on AUC for various models. 74

5.14 Impact of number of features on F-score for various models. 74

5.15 Impact of the health feature on the prediction performance. 76

5.16 ROC curves using the ARMA model (seven omit days). 80

Chapter 1

Introduction

1.1 Introduction and Motivation

The benefits of exercise and sport participation are well-documented and have been discussed and reported in the literature [1, 2, 3, 4, 5, 6, 7]. Regular exercise through sport participation can help in reducing stress and in improving physical and mental health. It is also helpful for obese people. Typically, athletes need to undergo high levels of physical training in order to improve their performance. Thus, there is always a chance of getting injured or sick. Injuries occur quite often at sport competitions and can be a serious concern for sport organizations.

Participation in sports continues to grow globally at every level. For example, college football programs increased by over one hundred programs in just 16 years [8]. As the number of participants in sports increases, it is reasonable that an increasing number of injuries may occur. Around 1.4 million injuries happen annually in the United States among high school athletes [9]. Researchers have found injury rates to be between 65% and 95% annually in Swedish soccer

populations [10]. Pre-injury analysis has found several factors that might increase injury risk among athletes. Researchers in [11] developed a theoretical model in which internal risk factors are related to an athlete's tendency towards increased injury risk.

It is evident from the references that athletes are prone to get injured or sick. This fact motivates us to investigate suitable models for predicting athletes' health (AH) beforehand so that precautionary measures can be taken before participating in an event. However, it is a challenging task to predict AH due to the nature of the psychometric data and the limitations of predicting models.

Before the advent of data science, sport organizations mainly depended on human experience which was neither accurate nor efficient. To overcome these problems, the use of machine learning (ML) and statistical techniques in sports has started to become popular recently. ML methods have been used for many purposes in sports. This thesis compares various supervised ML methods such as logistic regression (LR), naive Bayes (NB), k-nearest neighbor (k-NN), artificial neural network (ANN), support vector machine (SVM), decision tree (DT), and ensemble-based algorithms like AdaBoost, LogitBoost, RUSBoost, and random forest (RF) to construct suitable predictive models. This will enable sport organizations to monitor the health risks of their athletes. Our research emphasizes imbalanced data which has rarely been addressed in sports data analysis.

1.2 Literature Review

Due to an increase in sports activities, the prediction of AH has recently become an important research topic. One can find much research in the fields of sports. However, the application

of ML methods in this area is still limited. Most of the research in different areas of sports has been carried out using traditional statistical methods like regression analysis. For example, logistic regression is used in [12] to identify program applicant characteristics in order to predict academic success in professional academic training programs. However, the purpose of that study was to recruit and select the best students from a pool of candidates rather than predicting their health status.

Psychosocial variables are related to athletic injury and time missed from participation in sports. The influence of psychosocial variables and time to injury in college athletes is examined in [13]. However, their research focuses more on the psychosocial data that could be used for psychosocial screening of athletes.

Receiver operating characteristic and logistic regression analysis are used in [14] to identify dichotomized predictive factors that distinguished injured from uninjured status in college football players. Gary et al. in [15] address the prevalence of metabolic syndrome among collegiate football players and develop a clinical prediction rule to identify players who possess a high level of cardiometabolic risk. They used chi-square analysis and logistic regression in their study. Researchers in [16] use logistic regression in conjunction with the Wald test for the prediction of injuries and injury types to determine the relationship among the results in different groups of armies and soldiers. Smith et al. in [17] apply a univariate and step-wise logistic regression, F-test, and chi-square test to determine both the incidence of injury and the influence of physical, situational, and psychological factors in high school ice hockey players. The authors in [18] used machine learning and sensor fusion for fatigue prediction in outdoor runners. Their results claim to be useful and represent a solid start for moving into a real-world application

for monitoring the fatigue level of outdoor runners using wearable sensors. Although these papers achieved their respective research objectives, none of them, however, addressed the class imbalance problem that we often encounter in real-life applications.

ML methods have been used particularly for sport result prediction. In the paper [19], Josip and Alen used ML techniques to develop software in order to predict the results of football matches. However, the accuracy of the most successful classifier has reached a limit of 68% only. In [20], the authors develop a hybrid fuzzy-SVM model by integrating the fuzzy approach and the SVM technique for predicting basketball game outcomes that help the players to enhance their performance. They also claim that their proposed model can be applied to other sports such as soccer, baseball and golf. The paper [21] provides a critical analysis of the literature in machine learning, focusing on the application of ANN to sport result prediction. The article also proposes a novel sport result prediction framework through which machine learning can be used as a learning strategy. Machine learning techniques such as ANN, random forests and logistic regression are used in [22] to achieve an effective team strategizing analysis. Note, however, that all this research were conducted for sport result prediction using some popular ML methods. The crucial issue of imbalanced classes was not taken into consideration in those studies.

ML methods have also been used in many areas of medical science including breast cancer detection, cardiovascular risk prediction, stroke prediction, Parkinson's disease prediction, and medical imaging, to name a few [23, 24, 25, 26, 27, 28, 29]. For example, researchers in [29] use naive Bayes and logistic regression models for early detection of breast cancer. In the paper, the authors claim that computer-based quantitative methods improve diagnosis on breast

ultrasound and have the potential to reduce the number of biopsies. Wang et al. [28] use the multilinear sparse logistic regression model for the prediction of clinical risks of Alzheimer's disease and heart failure. Unlike logistic regression, which requires the inputs to be in vector form, the proposed method in [28] can directly take data matrices or tensors as inputs to perform prediction. Although these methods showed good performance in detecting breast cancer, Parkinson's and Alzheimer's diseases, they perform poorly where the minority class needs to be detected with reasonable accuracy.

Using ML techniques of boosted logistic regression, classification trees, Bayes Net and multilayer perceptron, researchers in [27] develop an improved approach for the prediction of Parkinson's disease for real life applications. The paper [24] compares deep neural network (DNN) with three other ML methods for predicting 5-year stroke occurrence using a large population-based electronic medical claims database. The article claims that the DNN achieves optimal results by using lesser amounts of patient data when compared to the gradient boosting decision tree. Stephen et al. [26] conducted a study which show that ML algorithms perform better at predicting the number of cardiovascular disease cases correctly. It is demonstrated in a large patient population using routinely collected electronic health data. The importance and applications of ML techniques in various healthcare industries have been addressed and explained in [25]. This research work demonstrates the usefulness of ML methods in various areas.

Inspired by the above-mentioned articles, in this thesis, we apply ML methods to our psychometric data in order to build suitable models for predicting AH. Traditional statistical models like regression analysis or ML methods such as LR, DT, etc. perform poorly in classifying

imbalanced data. In our research, we aim to improve the classification performance by using ensemble-based methods like AdaBoost, LogitBoost, RUSBoost, and random forest. We have chosen several supervised ML methods for prediction purposes and a comparative performance analysis of these methods has also been made by various performance evaluation metrics.

1.3 Research Objectives and Contributions

The main goal of this thesis is to build effective classifier models in order to predict athletes' health (illness/wellness) using a new dataset with imbalanced classes. This broad goal can be accomplished through the objectives and contributions outlined as follows:

- Clean and prepare the raw data for machine learning methods.
- Find the best interpolation technique to fill in the missing values in the dataset.
- Perform statistical analysis for feature selection purposes.
- Find the best possible parameters for the classifier models through optimization and fine-tuning techniques.
- Apply the traditional ML methods to the dataset for predicting athletes' health.
- Use ensemble-based methods to improve the classification performance.
- Use the ARMA model to further improve the classification performance.
- Analyze the results, compare the performance of the models with each other, and propose the best classifier model(s) for the prediction of athletes' health.

1.4 Organization of the Thesis

The remaining chapters of this thesis is organized as follows.

Chapter 2 presents a brief description of the dataset and explains various methods and techniques for data cleaning and preparation that are vital for constructing good predictive models. Predictive performance of ML models can be improved by properly handling the dataset. As such, this chapter talks about missing values treatment, outlier detection and treatment, handling imbalanced classes, and so on. It also explains some of the key issues related to machine learning that are useful for improving the classification performance.

Chapter 3 introduces the use of the correlation matrix and p-values to find correlated and significant features for feature selection. The aim is to reduce the size of the original dataset in an efficient way so that we can save time by not computing unnecessary features. This chapter also talks about various performance evaluation metrics, especially for imbalanced classes.

Chapter 4 first outlines the traditional ML methods that are applied to our data for predicting AH. The technique of fine-tuning is then applied to the models to find the optimal parameters for each model. Then, experiments using 10-fold cross-validation are performed followed by a discussion of the results. The performance of the classifier models is also compared with each other.

Chapter 5 first presents an overview of the ensemble-based methods used in this thesis in order to improve the classification performance. Model tuning is then performed to obtain the best possible parameters for each model. An ensemble approach called RUSBoost, which uses downsampling and boosting, is utilized to alleviate the class imbalance problems in the

dataset. This chapter also provides evaluation results through various experiments followed by a comparative discussion and analysis. The models are then tested after eliminating insignificant features and the health feature. The ML models are further tested using the ARMA model to see if the classification performance can be improved. Finally, a comparative discussion of the training time of each model is presented followed by a summary on the chapter.

Finally, Chapter 6 concludes the thesis by providing a summary of the research along with suggestions for future work.

Chapter 2

Data Exploration and Preparation for Machine Learning

Machine learning (ML) is a comparatively new method of data analysis that enables computers to search hidden insights while not being explicitly programmed [30, 31]. It is a branch of artificial intelligence (AI) that focuses on prediction, based on known properties learned from training information. Figure 2.1 shows a simple flowchart of how the ML system works to solve a problem.

ML algorithms are divided into supervised, unsupervised, and reinforcement learning tasks [31]. The basic idea of supervised learning is that the machine (computer) is provided with both training examples (inputs) and their correct labels (outcomes). Labelled data contains input feature variables and output target variables. The goal is to build a generalized model which can predict the outcome of new data. In other words, the machine learns to predict the outcome of new data based on the past examples [32]. Here, the users determine which

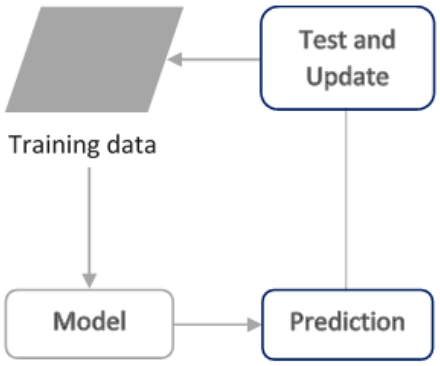

Figure 2.1: A simple flowchart of machine learning.

features the model should analyze and use to develop predictions. After finishing the learning or training stage, the model can be applied to new examples.

Supervised learning algorithms are widely used in many real-life applications such as bioinformatics, spam detection, fraud detection, database marketing, pattern recognition, speech recognition, and so on. Based on the desired output of a machine-learned system, ML algorithms can further be categorized as classification, regression, clustering, association rules learning, and dimensionality reduction. Classification algorithms are employed when the output labels are categorical such as healthy/sick, positive/negative, or male/female. In this thesis, we used supervised ML algorithms for predicting athletes' health (AH) and for classifying their health status.

Predictive performance of ML models can be improved both at the data level and at the algorithm level. In the process of building an efficient predictive model, we first address the following important topics that are used at the data level.

2.1 Data Exploration

Data exploration is a preliminary investigation process of a dataset in machine learning. In most cases, data are collected in an unstructured way and in large bulk. These raw data need to be prepared in an organized manner for better analysis. For these reasons, data exploration is necessary to have a better understanding of our data [33]. If we understand the characteristics of our data well, we can use them efficiently.

There are two ways to explore a dataset: statistical techniques and visualization techniques. Statistical techniques explore the mean, median, mode, and standard deviation of data. These values help us to better understand the dataset. Whereas, visualization techniques assist us in visualizing our data graphically. There are several types of plots that can be used to visualize data. Some of the most common plotting techniques include histogram, line plot, scatter plot, box plot, and density plot. Each type of plot is used for different purposes depending on the data at hand. By visualizing data using graphs and charts, we are better able to discover the most relevant aspects of our datasets.

2.2 The Purpose of Data Cleansing

Data quality is one of the major issues in machine learning and other related areas. Data cleansing is the process of identifying and rectifying inaccurate or corrupted data from a database to achieve high quality data. It is a valuable process that can help organizations save time and increase their efficiency. Maintaining excellent quality data is also essential since the performance of a predictive model could be impacted if the missing values are not appropriately

handled. Moreover, invalid or inconsistent data can lead to false conclusions. Therefore, we should ensure that the database is free from corruption. As a part of the data cleansing process, we will talk about missing values and outliers in the next sections after introducing our dataset.

2.3 Dataset and Feature Description

In this thesis, we have used a psychometric dataset from a company. Our dataset contains several features that include nutrition, sleep, irritability, hydration, stress, rest, energy, soreness, health, enjoyment, and exertion. The dataset also contains the athletes' names, IDs, and date of data entry. Athletes enter the data for these features on a daily basis. For each feature, they enter a number from 1 to 7. For example, in the health feature, if athletes feel very good for the day, they enter 7. When they are very sick, the entry is usually 1. When they feel moderate in health, the score varies from 4 to 6. The features of our dataset are presented in Table 2.1 with their respective ratings.

It has been documented in research-based articles that health and wellness are dependent upon several factors or features. The selection of features is very important in ML as it helps improve the model performance. Most of the features in the dataset we used in this thesis (e.g., nutrition, stress, rest, enjoyment, hydration, irritability, energy, etc.) are well-established in the literature [34, 35, 36]. For example, there is a clear relationship between nutrition and health. Similarly, hydration, rest, enjoyment, sleep, irritability, energy, etc. are also directly or indirectly related to health. On the other hand, stress affects both psychological and physiological functioning. There is also a strong positive connection between enjoyment and sense of well-being, including physical as well as mental health.

Table 2.1: Features and their respective ratings.

Feature	Ratings
Nutrition (x_1)	1 = very bad, 7 = very good
Sleep (x_2)	1 = very bad, 7 = very good
Irritability (x_3)	1 = worst, 7 = no irritability
Hydration (x_4)	1 = very bad, 7 = very good
Stress (x_5)	1 = overwhelmed, 7 = no stress
Rest (x_6)	1 = no rest, 7 = very good rest
Energy (x_7)	1 = very low, 7 = very high
Soreness (x_8)	1 = very sore, 7 = no soreness
Health (x_9)	1 = very bad, 7 = very good
Enjoyment (x_{10})	1 = no enjoyment, 7 = enjoyed very well
Exertion (x_{11})	1 = very low effort, 7 = very high effort

The original raw data contain a total of 1,116 athletes (or users). Out of 1,116 athletes, only 542 files contain valid records. The remaining athletes did not have any record at all. So, we have discarded those athletes whose records are empty. Moreover, many of them have missing data. The missing values were filled in by interpolation techniques. The athletes whose missing data rate (before interpolation/after interpolation) is more than 40% were also removed. After the initial stage of data cleaning, we have ended up with 265 athletes with many missing values.

2.4 Missing Values Treatment

Missing values in raw data are a common phenomenon in real-world problems. It may occur for several reasons and affects the quality of data. The impact of missing values can lead to many problems including biased estimates of parameters, loss of information, and weakened generalizability of findings.

The treatment of missing values has been a major concern in machine learning and data science in recent years. Missing values in data can be handled in various ways [37, 38, 39, 40, 41, 42]. First, if the number of cases of missing values are much smaller than the total number of observations, then the observations (rows) that contain missing values can be deleted without losing much information. Second, if a feature contains more missing values than the rest of the features in the dataset and if it is not a significant predictor, then that feature can be removed from the dataset. Third, the missing values can be replaced with the mean or median or mode of the respective feature. Fourth, the missing values can be replaced by interpolation techniques. Given the nature of our dataset, we have used interpolation techniques to fill in the missing values.

2.5 Choosing the Right Interpolation Technique

Interpolation is the process of estimating unknown values within the range of a discrete set of known data points. Many interpolation techniques exist in the literature [43]. They include linear interpolation, modified Akima cubic Hermite interpolation, nearest-neighbor interpolation, next-neighbor interpolation, shape-preserving piecewise cubic interpolation, previous-neighbor interpolation, spline interpolation, etc. A comparison of these interpolation techniques is shown in Figure 2.2 for an artificially generated signal.

The choice of an interpolation technique depends on the nature of the dataset and its application. There is no one best interpolation technique for any specific type of data or industry. To find the best technique for a specific dataset, the best thing to do is to experiment with different techniques until a desired result is obtained. In our case, we have applied different

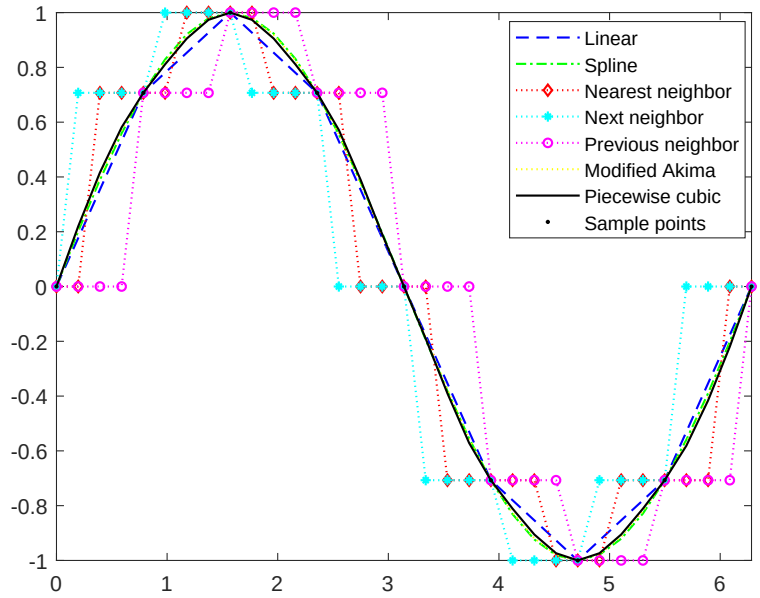

Figure 2.2: Comparison of various interpolation techniques.

interpolation techniques to our data to produce AUC (area under the ROC curve) using several ML methods. Then, we selected the interpolation technique which gave the best AUC. According to our experiments, the nearest-neighbor interpolation technique worked the best for our data. In nearest-neighbor interpolation, the interpolated value at a query point is the value at the nearest sample grid point which yields a piecewise-constant interpolant. It requires less memory and computation time than linear interpolation. After completing interpolation, we finally ended up with around 67,766 observations.

We have also used z-score to standardize the data values having a mean of zero and a standard deviation of 1. It is always beneficial to standardize or normalize the data. The optimization algorithm converges faster for normalized data, and, as a result, the algorithm run-time decreases.

2.6 Outliers Detection and Treatment

An outlier is an extreme value that is distant from other observations. Outlier detection is the process of detecting outliers from a given dataset. Most real-world datasets contain a certain number of outliers. These observations substantially deviate from the general trend. Therefore, we should isolate these outliers in order to improve the quality of original data and reduce the adverse impact on data analysis. Outliers in data can also distort predictions and affect accuracy. However, there is no standardized mathematical method for determining an outlier as it depends on the data and applications at hand.

Some of the most common causes of outliers in a dataset are data entry errors, measurement errors, experimental errors, data processing errors, and sampling errors. Graphical methods like boxplot and scatterplot are commonly used for detecting outliers. Some of the most popular statistical methods such as z-score, linear regression models (PCA), and k-means clustering are also used for outlier detection purposes [44]. A detailed survey of outlier detection methodologies can be found in [38].

The outliers can be treated in several ways. One way is to soften the impact of outliers by transforming data using some expressions such as square roots and logarithms. When the outliers lie significantly beyond the range of the data, they should be deleted. However, in the case of natural outliers that have significant impact on the dataset, they are treated separately. Another effective way of dealing with outliers is to use methods that provide a robust treatment of outliers, such as tree-based methods like random forest and boosting techniques. Regarding our dataset, some users have mistakenly inserted 8 or 9 instead of 7 in the exertion feature.

The number of these apparent outliers is very small compared to the total number of instances. Moreover, these few outliers do not have any significant impact on any model that we used in this thesis. However, we have replaced those outliers with the maximum rating 7.

2.7 Generating Health Index or Output Label

The original data did not have output labels. Therefore, we generated output labels for supervised learning algorithms. The criterion for predicting the health index of athletes was decided based on the health feature in a way that if the athletes are sick for at least one day in a period of one week, they are considered sick, otherwise they are assumed healthy. The AH is represented by a binary number referred to as the output label. If the value of the health feature is larger than 3.5, it is assigned 0 (healthy), otherwise 1 (sick) is assigned. This can be represented using OR logical function in a time period of one week as follows:

$$HInd_t = 1, \;\; if \;\; OR(Health_{t+1} < 3.5, Health_{t+2} < 3.5, Health_{t+3} < 3.5, Health_{t+4} < 3.5,$$
$$Health_{t+5} < 3.5, Health_{t+6} < 3.5, Health_{t+7} < 3.5)$$
$$Otherwies, \;\; HInd_t = 0 \quad (2.1)$$

where $HInd$ is the health index or output label and t is the time.

According to our dataset, most of the time the athletes have reported from 5 to 7. It makes the negative (healthy) cases much higher compared to the positive (sick) cases. An example of generating health index (output label) for a single athlete based on (2.1) is presented in Table 2.2 (right-hand column). By inspecting the features, most of the time the user has entered 5

Table 2.2: Seven omit days output labels for a single user.

Date	Nutrition	Sleep	Irritability	Hydration	Stress	Rest	Energy	Soreness	Health	Enjoyment	Exertion	$HInd$
13	4	4	4	4	4	4	4	4	4	5	7	-
12	5	6	5	3	5	5	3	4	5	5	7	-
11	5	5	4	5	5	5	3	3	3	3	7	-
10	6	4	5	5	4	4	4	4	4	5	5	-
9	4	4	4	4	4	4	4	4	4	4	6	-
8	6	6	5	6	6	6	5	6	5	5	5	-
7	5	6	5	3	4	5	5	3	5	3	7	-
6	5	6	3	3	5	6	5	7	5	5	7	1
5	5	5	5	4	6	5	3	3	5	5	5	1
4	5	5	5	4	6	5	3	3	5	5	5	1
3	5	6	5	3	3	6	3	3	5	2	4	0
2	6	6	6	4	5	5	5	3	5	5	5	0
1	5	4	5	5	6	5	3	5	4	5	7	0

or above. It is also observable that there are no 1 or 2 ratings for this user. About the labels, there can be found 3 healthy (negative) cases and 3 sick (positive) cases.

Note that instead of using individual athlete data, we have used the panel data which is the combination of all the athlete data. Panel data usually refers to data containing time series observations of a number of individuals [45, 46]. In our case, the panel data can be represented in the matrix form as:

$$\mathbf{X} = \begin{bmatrix} X_1 \\ X_2 \\ \vdots \\ X_N \end{bmatrix} \tag{2.2}$$

where N represents the total number of athletes, $\mathbf{X}$ is the panel data matrix, and $X_1, X_2, ..., X_N$

represent individual athlete data matrix. An individual data matrix can be expressed as:

$$X = \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{m1} \\ x_{12} & x_{22} & \cdots & x_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1T_j} & x_{2T_j} & \cdots & x_{mT_j} \end{bmatrix} \tag{2.3}$$

where m is the number of features (columns) in the dataset, T_p is the time variable, and x_{iT_p} refers to the data value of feature i in T_p time. The advantage of panel data is that it gives us a large number of data points which increase the degrees of freedom and reduces the collinearity among explanatory variables. As a result, panel data offers more accurate inference of model parameters [45].

Our initial investigation suggests that the positive (sick) and negative (healthy) cases in our dataset are not balanced in that there are many more healthy cases (over 90%) than sick ones. It means that our dataset contains imbalanced classes. It can also be seen in histograms in Figures 2.3 to 2.5. The histogram of the health feature makes it clear that most of the good scores (healthy cases) are on the right-hand side of the histogram. In the next section, we address dealing with imbalanced classes.

2.8 Handling Imbalanced Classes

The existence of imbalanced data has been a relatively new challenge to the data scientist community. Most of the real-world datasets commonly exhibit an unbalanced distribution, where one of the classes outnumbers other classes by a large proportion. This kind of problem

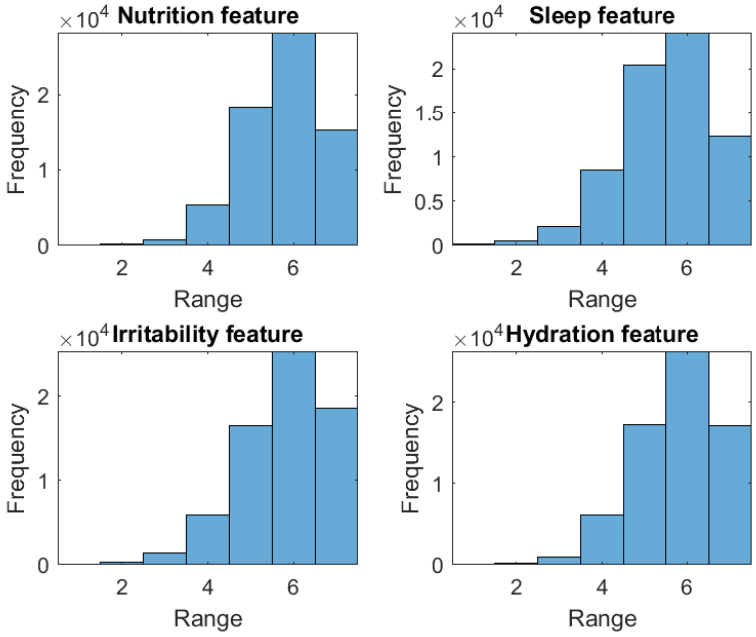

Figure 2.3: Histogram of nutrition, sleep, irritability, and hydration features for all athletes.

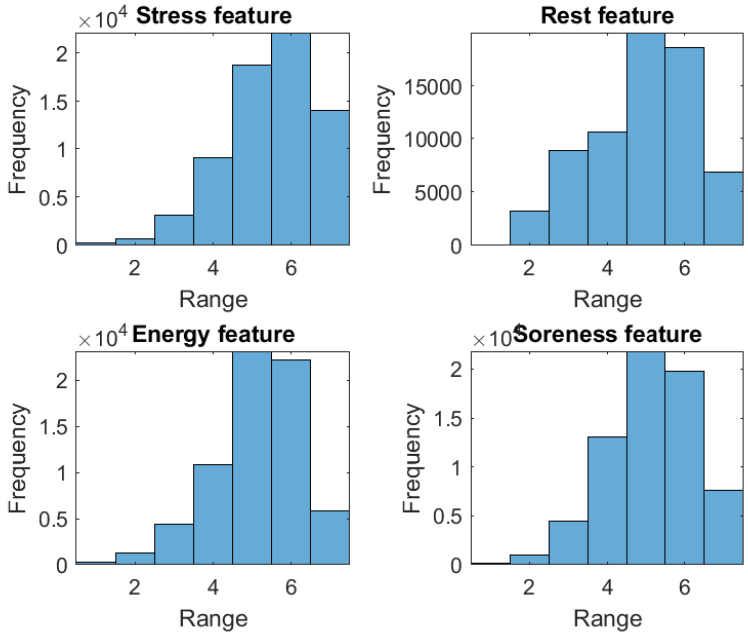

Figure 2.4: Histogram of stress, rest, energy, and soreness features for all athletes.

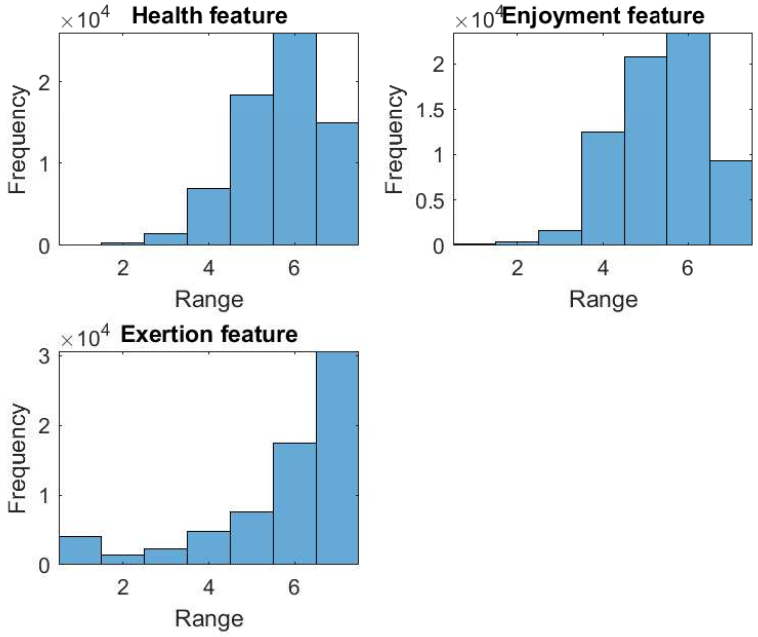

Figure 2.5: Histogram of health, enjoyment, and exertion features for all athletes.

is encountered more frequently in binary classification problems than multi-class classification problems. The imbalanced classes can be found in many applications such as fraudulent transaction detection, spam filtering, or identifying cancer, hepatitis, and cardiovascular disease.

Traditional ML methods will not perform well for class-imbalanced datasets as these methods will be biased towards the majority class [47]. Several techniques have been proposed both at the data level and at the algorithmic level to solve class-imbalance problems [48, 49, 50, 51]. Data level techniques for handling imbalanced datasets include undersampling, oversampling, and feature selection. Algorithm level techniques for dealing with imbalanced datasets include threshold methods and cost-sensitive learning. In addition to these techniques, combining methods are also used for the same purposes.

To deal with imbalanced data, some simple steps can be applied first. For example, more

data can be collected as a larger dataset might have more balanced classes. The performance evaluation metric can be changed. Instead of using traditional classification accuracy, we can use AUC and F-score as performance measures. In the next step, we can apply re-sampling techniques at the data level to combat imbalanced data. These techniques aim to change or modify the dataset using some mechanisms to have more balanced data. Resampling of imbalanced data is performed mainly in three ways: oversampling, undersampling, and synthetic data generation. At the algorithmic level, we can try different techniques and models to evaluate what works best. The most used of all these techniques are briefly explained in the subsequent sections.

2.8.1 Oversampling

The idea of oversampling is to increase the size of the minority class until the number of observations balance the majority class. This technique is used when the quantity of data is insufficient. An advantage of the oversampling technique is that it leads to no loss of information. However, this technique may lead to overfitting and can also introduce an additional computational cost.

2.8.2 Undersampling

Undersampling is an efficient technique for balancing imbalanced data. It is performed by deleting instances from the majority class to balance the data. This technique is usually used for a large dataset. However, reducing observations may lose some information from the training set.

2.8.3 Synthetic Data Generation

The synthetic data generation technique aims to overcome imbalances by generating synthetic data. A simple way to generate synthetic samples is to randomly sample the instances in the minority class. Using this concept, SMOTE (Synthetic Minority Oversampling Technique) has been a powerful and widely used technique in machine learning to combat imbalanced data. Instead of creating mere copies, the SMOTE algorithm creates synthetic samples from the minority class. It first selects two or more similar samples from the minority class using a distance measure. Then, it introduces synthetic samples along the line segments connecting any/all the k minority class nearest neighbors. Neighbors from the k-nearest neighbors are selected randomly depending on the amount of required oversampling. Since there is no replication of minority class records, SMOTE does not suffer from overfitting problems seriously.

In addition to the above techniques, penalized algorithms or ensemble methods are also used to handle the classification problem of imbalanced data [52, 53, 54, 55, 56]. These methods impose a penalty for misclassification on the minority class during training. These penalties can push the model in a way that it pays more attention to the minority class.

In the process of building a robust predictive model, we encounter many challenging problems. In addition to the above-mentioned methods and techniques, the model performance can also be improved through some changes or modifications or tuning at the algorithm level. In the next sections, we will present some of the most common problems that can be removed or minimized at this level to enhance model performance.

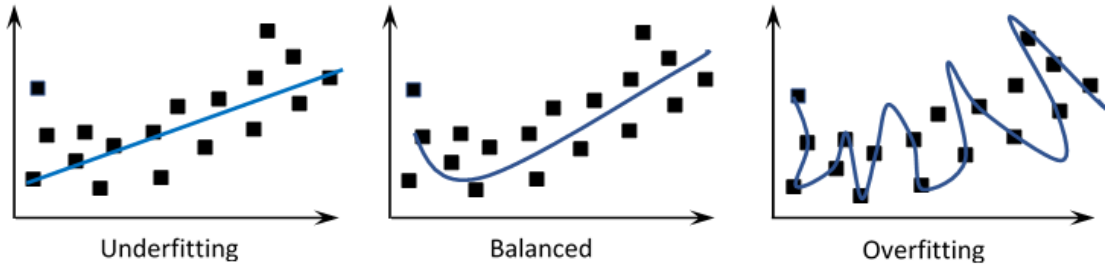

Figure 2.6: Examples of underfitting (left), balanced (middle), and overfitting (right) models.

2.9 Underfitting and Overfitting Problems

Underfitting and overfitting are two common problems in machine learning and data science. A machine learning model is said to have underfitting when it cannot capture the underlying trend of the training data. It is usually a result of an extremely simple model. The picture on the left of Figure 2.6 shows an underfitting case. As can be seen, the line does not cover all the points in the graph. Such an underfit model will have high training and high testing error. Underfitting can be avoided in several ways such as by using more data and/or by adding more features and/or by increasing the degree of the polynomials or by using alternative models.

A model is said to be overfitting when it performs very well on training data but performs poorly on unseen data. It is typically a result of an overly complicated model. The graph on the right in Figure 2.6 shows that the predicted line covers all the points in the graph. This kind of model will have an extremely low training error but a high testing error.

Both underfitting and overfitting can lead to poor prediction performance on new datasets. To know how well a model will perform on new data, we can split our initial dataset into separate training and test subsets. If the model performs very well on the training set compared to its performance on the test set, then it is most likely an overfitting case.

When we build a predictive model, we want to avoid overfitting as it performs poorly on the unseen data. However, in the process of reducing overfitting, we may end up having an underfit model which is another problem in machine learning. This is known as the bias-variance dilemma. We need to minimize both bias and variance as much as possible in building a model which generalizes well to new data. Cross-validation is a powerful preventative measure against overfitting [57]. The methods of pruning and regularization are also used to prevent overfitting in some cases. We will explain these methods in the following sections.

2.10 Cross-validation to Avoid Overfitting

One simple way to avoid overfitting is to divide the dataset into a training set to train the model, and a validation or test set to evaluate its performance. This technique is called cross-validation in its simplest form. This simple technique helps avoid overfitting as the model is tested against unseen data. However, the most popular and widely used method is k-fold cross-validation [58, 59, 60, 61]. In k-fold cross-validation, the original dataset is partitioned into k subsets or k folds. Of the k subsets, a single subset is kept for validation, and the remaining k-1 subsets are used for training the model. The process is then repeated k times until each of the k subsets is used as the validation set. The average of k results is then used to produce a single estimation.

Choosing the right value of k is a difficult task. A lower value of k can produce more biased results. Whereas, a higher value of k can reduce bias but can suffer from large variability. In practice, the best choice for the value of k is 5 or 10 [61]. One obvious disadvantage of this method is that it can be computationally intensive. In the case of a smaller dataset, it is better

to apply cross-validation. Otherwise, a simple train-test split could be enough for a larger dataset.

2.11 Pruning to Reduce Overfitting

Pruning is a technique used in machine learning for dealing with overfitting in tree-based algorithms [62, 63]. Pruning can be of two types: pre-pruning and post-pruning. Pre-pruning is the process that stops the tree-building process before it becomes a fully-grown tree. One obvious disadvantage of early stopping is that it may underfit the data by stopping too early. However, early stopping may save computation time. Whereas, post-pruning allows the tree to grow fully with no size limit, and then prune or remove the nodes that do not provide additional information. Pruning in general reduces the complexity and the overfitting of the final classifier without reducing predictive accuracy significantly.

2.12 Model Tuning

So far, we addressed many methods and techniques for improving the results of a machine learning model. Model tuning is another strategy that can be employed intelligently to boost model performance. This technique is also called parameter tuning or parameter optimization. ML models require us to set different parameters. We can tune or optimize those parameters until the best possible result is achieved.

Our ML-based prediction algorithm can be summarized as follows:

- Clean and prepare the dataset.

- Generate the output labels.
- Divide the dataset into training and test sets.
- Build/train the models using the training dataset.
- Evaluate the models using the test dataset.
- Predict the output and make a decision (healthy or sick).

2.13 Summary

In this chapter, we presented some useful methods and techniques for data cleaning and preparation and to enhance the predictive power of ML algorithms. We also addressed some important issues related to machine learning that are useful for improving the classification performance. In the next chapter, statistical analysis for feature selection along with evaluation metrics to evaluate the performance of ML models are presented.

Chapter 3

Statistical Analysis for Feature Selection and Evaluation Metrics

3.1 Statistical Analysis for Feature Selection

Our dataset might contain an excess number of features. All those features might not be useful in constructing a machine learning (ML) model. In fact, some features might even make the model worse. So, feature selection plays an important role in building ML models. We want to explain the data in the simplest possible way and thus the redundant features should be removed. Feature selection is the process by which we can reduce the size of the original dataset by selecting the best features out of many. By doing this, we can save time and/or money by not measuring unnecessary features. In this thesis, we use correlation matrix and p-value as measures to select the right features.

3.1.1 Correlation Matrix

Correlation means the degree of association between two independent variables or features. When there are more than two features, the collection of pair-wise correlations is succinctly represented in a correlation form which is called the correlation matrix. In short, a correlation matrix is a table showing correlation coefficients among all features. The purpose of examining correlations is to identify the collinearity between features.

The values of the correlation coefficients always range from -1 to 1. The positive or negative sign tells us the direction of the relationship, and the number tells us the strength of the relationship. If there is perfect positive relationship between two features, the correlation coefficient will be 1. If there is perfect negative connection between two features, the correlation coefficient is -1. A correlation coefficient of zero means that there is no link between the features. The most common way to quantify this relationship is the Pearson's correlation coefficient.

Typically, a correlation matrix is square, with the same features shown in the rows and columns. This matrix is symmetrical, with identical correlation shown on top of the main diagonal being a mirror image of those below the main diagonal. The line of 1s going from the top left to the bottom right is the main diagonal, which shows that each feature always perfectly correlates with itself.

Highly correlated features are more linearly dependent and hence have almost the same effect on the output variable. So, when two features have high correlation, we can drop one. For our data, a correlation matrix is used to investigate the relationship among multiple features together. The correlation matrix of our data is shown in Table 3.1. Besides the main diagonal elements, it can be observed that the highest correlation (0.53) occurs between stress and

Table 3.1: Correlation matrix.

	Nutrition	Sleep	Irritability	Hydration	Stress	Rest	Energy	Soreness	Health	Enjoyment	Exertion
Nutrition	1	0.41	0.43	0.52	0.37	0.33	0.33	0.27	0.49	0.30	0.07
Sleep	0.41	1	0.42	0.39	0.38	0.47	0.50	0.32	0.46	0.27	0.07
Irritability	0.43	0.42	1	0.36	0.53	0.34	0.35	0.29	0.45	0.30	0.05
Hydration	0.52	0.39	0.36	1	0.30	0.30	0.33	0.25	0.40	0.28	0.10
Stress	0.37	0.38	0.53	0.30	1	0.32	0.37	0.31	0.39	0.25	0.04
Rest	0.33	0.47	0.34	0.30	0.32	1	0.42	0.33	0.39	0.18	0.13
Energy	0.33	0.50	0.35	0.33	0.37	0.42	1	0.46	0.44	0.29	0.05
Soreness	0.27	0.32	0.29	0.25	0.31	0.33	0.46	1	0.34	0.21	0.01
Health	0.49	0.46	0.45	0.40	0.39	0.39	0.44	0.34	1	0.31	0.07
Enjoyment	0.30	0.27	0.30	0.28	0.25	0.18	0.29	0.21	0.31	1	0.07
Exertion	0.07	0.07	0.05	0.10	0.04	0.13	0.05	0.01	0.07	0.07	1

irritability features. But the lowest correlation (0.01) can be found between soreness and exertion. However, there is no perfect or high correlation between any two features in our dataset.

3.1.2 t-Value and p-Value

The t-value is a test statistic. A test statistic is the result of a statistical test or t-test. The exact formula for t depends on the type of t-test. For two sample tests, it is a standardized difference between the two means. That is, it is a measure of how far apart the two means are. Every t-value has a p-value (probability value) associated with it to measure the statistical significance of the difference. There can be found tables, spreadsheet programs and statistical software to help calculate the p-value. p-Value is used to decide if a feature is statistically significant based on a significance level α (usual values used for $\alpha = 0.05, 0.01, 0.001$). If $p \leq \alpha = 0.05$ for a specific feature, then it is considered as statistically significant, otherwise it is assumed as

Table 3.2: Descriptive statistics and p-values for all features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4335 (β_0)	0.0147	-165.48	0
Nutrition	-0.0418 (β_1)	0.0142	-2.9556	0.00312*
Sleep	-0.0131 (β_2)	0.0150	-0.8706	0.38398
Irritability	-0.0087 (β_3)	0.0151	-0.5756	0.56491
Hydration	-0.0464 (β_4)	0.0141	-3.2972	0.00098*
Stress	0.0105 (β_5)	0.0151	0.6923	0.48876
Rest	-0.0638 (β_6)	0.0150	-4.2533	2.11e-05*
Energy	-0.0174 (β_7)	0.0157	-1.1145	0.26508
Soreness	-0.0045 (β_8)	0.0143	-0.3153	0.75257
Health	-0.4248 (β_9)	0.0129	-32.905	1.9e-237*
Enjoyment	-0.0024 (β_{10})	0.0140	-0.1747	0.86136
Exertion	-0.0401 (β_{11})	0.0137	-2.9336	0.00335*
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0 * represents significant p-value with $\alpha \leq 0.05$				

statistically insignificant. Table 3.2 presents the estimated feature coefficients and p-values for our dataset. The regression model used to obtain the statistics in Table 3.2 is given in (3.1) for eleven features.

$$\widehat{HInd} = \beta_0 + \beta_1 Nutrition + \beta_2 Sleep + \beta_3 Irritability + \beta_4 Hydration + \beta_5 Stress + \beta_6 Rest + \beta_7 Energy + \beta_8 Soreness + \beta_9 Health + \beta_{10} Enjoyment + \beta_{11} Exertion \quad (3.1)$$

where β_0 is the intercept and the parameters $\{\beta_1, \beta_2, ..., \beta_{11}\}$ represent the coefficients of the respective features.

Considering $\alpha = 0.05$ (5% significance level), the p-values of 0.00312, 0.00098, 2.11e-05, 1.9e-237, and 0.00335 indicate that the coefficients of the features nutrition, hydration, rest, health,

Table 3.3: Descriptive statistics and p-values for ten features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4335	0.0147	-165.48	0
Nutrition	-0.0419	0.0141	-2.9624	0.00305
Sleep	-0.0131	0.0151	-0.8703	0.38413
Irritability	-0.0089	0.0150	-0.5942	0.55236
Hydration	-0.0464	0.0141	-3.3019	0.00096
Stress	0.0104	0.0151	0.6861	0.49264
Rest	-0.0637	0.0150	-4.2498	2.14e-05
Energy	-0.0177	0.0156	-1.1356	0.25611
Soreness	-0.0046	0.0143	-0.3238	0.74606
Health	-0.4250	0.0129	-32.961	2.9e-238
Exertion	-0.0403	0.0137	-2.9454	0.00323
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0				

and exertion are statistically significant. The features sleep and energy can be considered as relatively less significant. On the other hand, the features irritability, stress, soreness, and enjoyment are not significant at the 5% significance level given the other terms in the model. Therefore, we can remove these insignificant features to test our model performance. Backward elimination technique can be applied for this purpose. According to this technique, we start with all the features in the model. Then, we remove the feature with highest p-value greater than α. We then refit the model and repeat the previous step. The process is stopped when all p-values are less than α. The results are presented in Tables 3.3 to 3.8. In Table 3.3, the feature enjoyment (p=0.86136) was removed. In Table 3.4, the feature soreness (p=0.74606) was removed. In a similar manner, the feature related to the maximum p-value greater than α was removed in every subsequent Table. Based on these data, we will present experimental results in Chapter 5. Note that the significant features may not be automatically good predictors [64].

Table 3.4: Descriptive statistics and p-values for nine features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4334	0.0147	-165.48	0
Nutrition	-0.0419	0.0141	-2.9621	0.00306
Sleep	-0.0131	0.0151	-0.8678	0.38549
Irritability	-0.0091	0.0150	-0.6063	0.54435
Hydration	-0.0465	0.0141	-3.3060	0.00095
Stress	0.0101	0.0151	0.6714	0.50196
Rest	-0.0641	0.0149	-4.3010	1.70e-05
Energy	-0.0186	0.0153	-1.2123	0.22541
Health	-0.4251	0.0129	-33.007	6.5e-239
Exertion	-0.0401	0.0137	-2.9354	0.00333
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0				

Table 3.5: Descriptive statistics and p-values for eight features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4334	0.0147	-165.48	0
Nutrition	-0.0425	0.0141	-3.0138	0.00258
Sleep	-0.0138	0.0150	-0.9205	0.35730
Hydration	-0.0470	0.0140	-3.3500	0.00081
Stress	0.0071	0.0142	0.4965	0.61953
Rest	-0.0643	0.0149	-4.3104	1.63e-05
Energy	-0.0195	0.0153	-1.2809	0.20022
Health	-0.4258	0.0128	-33.181	2.0e-241
Exertion	-0.0401	0.0137	-2.9311	0.00338
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0				

Table 3.6: Descriptive statistics and p-values for seven features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4335	0.0147	-165.48	0
Nutrition	-0.0420	0.0141	-2.9841	0.00284
Sleep	-0.0131	0.0149	-0.8794	0.37919
Hydration	-0.0467	0.0140	-3.3312	0.00086
Rest	-0.0638	0.0149	-4.2861	1.82e-05
Energy	-0.0186	0.0151	-1.2285	0.21925
Health	-0.4253	0.0128	-33.246	2.3e-242
Exertion	-0.0401	0.0137	-2.9303	0.00339
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0				

Table 3.7: Descriptive statistics and p-values for six features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4334	0.0147	-165.48	0
Nutrition	-0.0428	0.0140	-3.0483	0.00230
Hydration	-0.0474	0.0140	-3.3915	0.00070
Rest	-0.0666	0.0145	-4.5873	4.49e-06
Energy	-0.0223	0.0145	-1.5367	0.12436
Health	-0.4265	0.0127	-33.548	9.6e-247
Exertion	-0.0401	0.0137	-2.9316	0.00337
Chi^2-statistic vs. constant model: 1.56e+03, p-value = 0				

Table 3.8: Descriptive statistics and p-values for five features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.4332	0.0147	-165.49	0
Nutrition	-0.0439	0.0140	-3.1374	0.00171
Hydration	-0.0486	0.0140	-3.4816	0.00050
Rest	-0.0722	0.0140	-5.1403	2.74e-07
Health	-0.4309	0.0124	-34.725	3.3e-264
Exertion	-0.0397	0.0137	-2.9054	0.00367
Chi^2-statistic vs. constant model: 1.55e+03, p-value = 0				

3.2 Performance Evaluation of Machine Learning

After implementing a machine learning model and getting some results, the next step is to see how effective the model is based on some metrics. The evaluation of a model is normally done by splitting the dataset into a training set and a test set. The model is then trained on the training set, while the test set is used to calculate performance indicators to evaluate the quality of the model. Various performance evaluation metrics are used to evaluate different models. In this thesis, we focus on the ones that are used for classification problems [65]. The choice of metrics is also important because it influences how the performance of ML algorithms is measured and compared.

The most commonly used metric to evaluate classifier performance is accuracy, which is the percent of correct prediction. Accuracy is a good measure when the target variable classes in the dataset are nearly balanced. However, accuracy gives a false impression about the classifier in cases with highly skewed dataset, i.e., where the classes are not balanced. Therefore, we need to use additional metrics such as AUC (area under the ROC curve), precision, recall, and

Table 3.9: Example of a confusion matrix.

		Predicted Class	
		Positive	Negative
Actual Class	Positive	TP	FN
	Negative	FP	TN

F-score for evaluation. The best-known evaluation metrics including the confusion matrix are presented below.

Confusion Matrix: A matrix that contains information about true and predicted values of a classifier. It is useful for measuring evaluation metrics such as accuracy, precision, recall, specificity, F-score, etc. Table 3.9 shows the confusion matrix for a binary class classifier.

True Positives (TP): The number of cases when the actual class and the predicted class are both positive.

True Negatives (TN): The number of cases when the actual class and the predicted class are both negative.

False Positives (FP): The number of cases when the actual class is negative, but the predicted class is positive. It is also known as the false alarm.

False Negatives (FN): The number of cases when the actual class is positive, but the predicted class is negative.

Accuracy: The ratio of correctly predicted observations to the total observations, also called the probability of a correct classification.

Precision: The ratio of correctly predicted positive cases to the total predicted positive cases. High precision relates to the low false positive rate.

Table 3.10: Performance evaluation metrics.

Measure	Formula
AUC	Area under the ROC curve
Accuracy	$\frac{TP+TN}{TP+FN+FP+TN}$
Precision	$\frac{TP}{TP+FP}$
Recall	$\frac{TP}{TP+FN}$
Specificity	$\frac{TN}{TN+FP}$
F-score	$2 \times (\frac{Precision \times Recall}{Precision+Recall})$

Recall or Sensitivity: The number of positive predictions divided by the number of positive cases in the dataset, also known as the true positive rate (TPR).

Specificity: The proportion of actual negative cases which are correctly predicted, also known as the true negative rate (TNR).

F-score: The weighted average of precision and recall. It conveys the balance between the precision and the recall by taking both false positives and false negatives into account. In the case of imbalanced classes, F-score is more useful than accuracy. It tells us how precise a classifier is. An F-score of 1 means perfect precision and recall, and 0 means the worst case.

The formulae for performance evaluation metrics are tabulated in Table 3.10.

The receiver operating characteristic (ROC) curve is a plot of the true positive rate (TPR) against the false positive rate (FPR). It can be used to select a threshold for a classifier which maximizes the true positives, while minimizing the false positives. ROC curves also give us the ability to assess the performance of the classifier over its entire operating range [66, 67]. The most widely used measure is the AUC, which is the area under the ROC curve. The greater

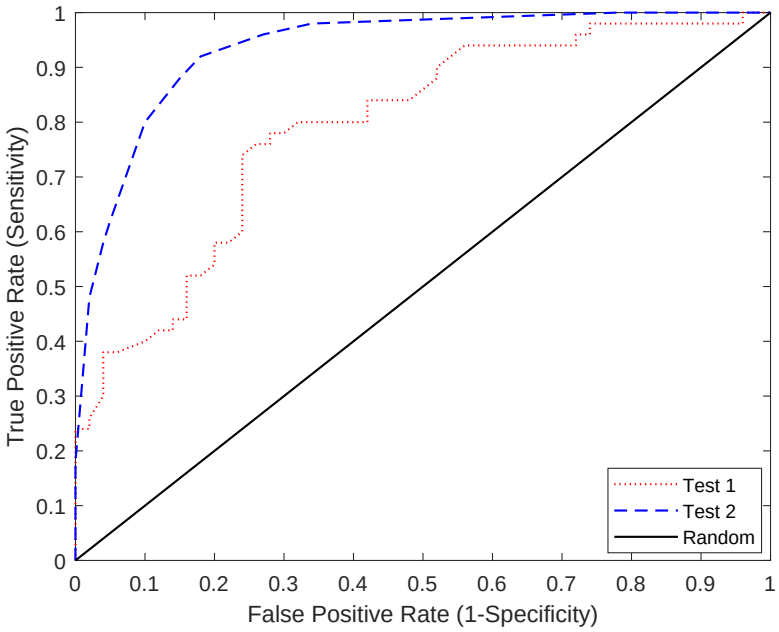

Figure 3.1: ROC curves showing two test results.

the AUC, the more accurate the test. A perfect test has an AUC of 1. The AUC can also be used to compare the performance of two or more classifiers. If one ROC curve dominates all others, then the best method is the one that produced the dominant curve, which is the curve with the largest AUC. Figure 3.1 shows ROC curves of two test results. In terms of AUC, Test 2 is better than Test 1.

3.3 Summary

In this chapter, we presented a statistical analysis of our data using a correlation matrix and p-values for an efficient way of feature selection. We also discussed performance evaluation metrics, especially for imbalanced classes. In the next chapter, traditional ML methods and relevant results are presented for the prediction of athletes' health.

Chapter 4

Athlete Health Prediction Using Traditional ML Methods

4.1 Introduction

The primary aim of this study was to analyze a dataset and to build suitable classification models in order to predict athletes' health (AH). The secondary purpose was to find the best classifier for our class-imbalanced dataset. This is the first known study that compares various supervised ML methods for predicting AH. We evaluated the performance of the models by six measures: AUC, accuracy, precision, recall, specificity, and F-score. ROC curves were also used for a better visualization between the true positive rate (recall/sensitivity) and the false positive rate.

The rest of this chapter is organized as follows. Section 4.2 states the formulation of the problem. Section 4.3 introduces the traditional ML methods namely logistic regression, naive

Bayes, k-nearest neighbor, artificial neural network, support vector machine, and decision tree. Section 4.4 discusses about parameter tuning of the models. Experiments are then conducted, and the results and analysis are given in Section 4.5 to evaluate the performance of the models. Section 4.6 summarizes the chapter.

4.2 Problem Formulation

Getting injured or sick is a common phenomenon among sport participants and can be a serious concern for sport organizations. Millions of injuries occur annually throughout the world among sports participants. As such, the prediction of health earlier could help sport organizations to monitor illness/wellness of their athletes before participating in an event. However, most of the research on this area of sports has been carried out using traditional statistical methods like regression analysis. The research in this study, on the other hand, aims to use ML methods for predicting athlete health. To do this, we used a psychometric dataset from an organization to perform training and to test the models to measure their classification performance.

Let $\mathbf{X}$ be the input matrix of our dataset whose $i-th$ row is the input vector x_i. If there are n rows or instances and m features or predictors, $\mathbf{X}$ is a matrix with dimension $n \times m$. Also, let $\mathbf{Y}$ be the $n \times 1$ vector of health labels or target values and $HInd_i$ be the $i-th$ value of $\mathbf{Y}$. Given the input matrix $\mathbf{X}$ and the health label vector $\mathbf{Y}$, the goal is to build suitable ML models in order to predict AH label. In relation to our dataset, the prediction of AH label

is made based on eleven features which can be expressed as:

$$\widehat{HInd} = \beta_0 + \beta_1 Nutrition + \beta_2 Sleep + \beta_3 Irritability + \beta_4 Hydration + \beta_5 Stress + \beta_6 Rest + \beta_7 Energy + \beta_8 Soreness + \beta_9 Health + \beta_{10} Enjoyment + \beta_{11} Exertion \quad (4.1)$$

where β_0 is the intercept and the parameters $\{\beta_1, \beta_2, ..., \beta_{11}\}$ represent the coefficients of the respective features.

4.3 Traditional Machine Learning Methods

A prime objective of learning is to generalize from its experience. In machine learning, generalization is the ability of a learning machine to perform reasonably well on unseen examples after having experienced a training dataset. The aim is to build a generalized model based on the training examples that enable it to produce sufficiently accurate predictions in new cases. The traditional ML methods that were used in this thesis are briefly introduced in the next subsections in the context of our dataset.

4.3.1 Athlete Health Prediction Using Logistic Regression

Logistic regression (LR) is a type of classification model for analyzing a dataset in which there are one or more independent variables that determine an outcome [28]. We applied LR to our data as the binary logistic model has two possible outcomes such as sick or healthy. To understand logistic regression, it is important to understand the sigmoid function. The sigmoid function can take any real input, whereas the output always takes values between 0

and 1 [68]. Given the parameter vector $\beta = (\beta_0, \beta_1, ..., \beta_m)^T$ and an input set of features $x = (x_0, x_1, ..., x_m)^T$, $x_0 = 1$, the hypothesis for logistic regression can be expressed as:

$$h_\beta(x) = \frac{1}{1 + e^{-(\beta_0 x_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_m x_m)}} = \frac{1}{1 + e^{-\beta^T x}} \tag{4.2}$$

where in relation to our dataset: x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8, x_9, x_{10}, and x_{11} ($m = 11$) respectively represent the features nutrition, sleep, irritability, hydration, stress, rest, energy, soreness, health, enjoyment, and exertion. The parameters in the vector β are determined by minimizing the cost function using the gradient descent algorithm. The output $h_\beta(x)$ in (4.2) has the value in the range (0,1) and is interpreted as the probability of the class variable $HInd$. The new input example x is classified as healthy when $h_\beta(x) \geq 0.5$, else it is classified as sick. The threshold 0.5 can be changed depending on our requirements.

4.3.2 Athlete Health Prediction Using Naive Bayes

Naive Bayes (NB) is a supervised ML method used for binary and multi-class classification problems [69], which is also useful for our data. For a binary classification problem like ours, given a health label $HInd \in (0, 1)$ and a vector of features $x = x_1, x_2, ..., x_m$, Bayes' theorem can be stated mathematically as follows:

$$P(HInd|x) = \frac{P(HInd)P(x|HInd)}{P(x)} \tag{4.3}$$

where P denotes probability, $P(HInd|x)$ is the posterior probability (outcome) of health label $HInd$ given input feature x, $P(HInd)$ is the prior probability of $HInd$, $P(x|HInd)$ is the

likelihood which is the probability of x given $HInd$, and $P(x)$ is the prior probability of x. With the naive independence assumption among the m features, (4.3) can be rewritten as:

$$P(HInd|x_1, ..., x_m) = \frac{1}{C} P(HInd) \prod_{i=1}^{m} P(x_i|HInd) \tag{4.4}$$

where $C = P(x) = \sum P(HInd)P(x|HInd)$ is a constant for a given input. To create a classifier model, we need to find the probability of a given input set for all possible values of the class variable $HInd$. According to the *maximum a posteriori* or MAP decision rule, we can pick up the output with maximum probability. This can be expressed mathematically as:

$$\widehat{HInd} = \underset{HInd}{\operatorname{argmax}} P(HInd) \prod_{i=1}^{m} P(x_i|HInd) \tag{4.5}$$

Using (4.5), we can predict AH for new instances. For numerical features, a common strategy is to fit a distribution over the data and use this to get estimates of $P(x_i|HInd)$. An advantage of naive Bayes classifier is that it is computationally fast, and it requires limited training data to estimate the parameters necessary for classification.

4.3.3 Athlete Health Prediction Using k-Nearest Neighbor

k-Nearest neighbor (k-NN) is a non-parametric supervised method of machine learning that can be applied to our data. Its purpose is to predict the classification of a new input instance with the help of a training dataset. For a new sample x, predictions are carried out by searching through the whole training dataset for the k most similar neighbors based on some distance metric [69]. The distance is then used to find the k-closest neighbors to a new sample.

Selecting the value of k in k-NN is the most critical problem. We used the technique of parameter tuning in that we computed classification errors for different values of k to obtain the best value of k. Given a positive integer k, the k-NN algorithm can be summarized as follows:

- Select an appropriate distance metric for the training data.
- Compute distances between a new sample x and each training instance $x_i, i = 1, ..., n$.
- Arrange the distances in ascending order.
- Take the first k entries from this sorted list which are closest to the new sample.
- Find the output labels of the selected k nearest points.
- Determine the AH label based on the majority voting.

k-NN has some disadvantages, especially being computationally intensive for very large data and prone to overfitting. However, its simplicity can make it a relatively good benchmark for more complicated methods.

4.3.4 Athlete Health Prediction Using Artificial Neural Network

Artificial neural network (ANN) is a branch of computational intelligence and one of the main tools used in machine learning for classification purposes [69]. Depending on the applications, there can be found multiple types of neural networks. We chose to apply the feedforward neural network to our data for the prediction of AH. The basic architecture of a neural network is shown in Figure 4.1.

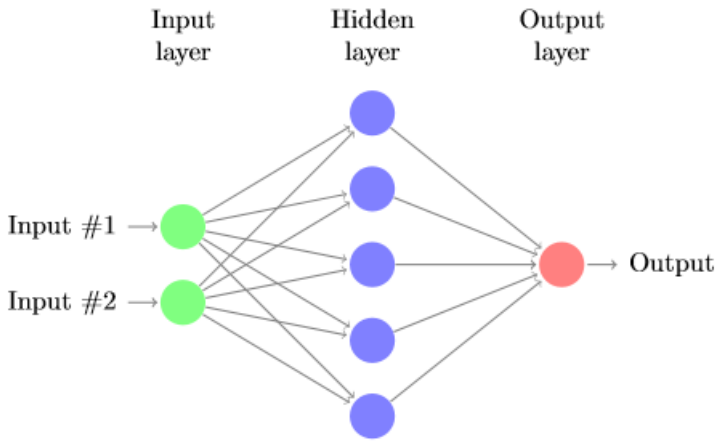

Figure 4.1: An example of a neural network structure. (Credit: TeXample.net)

In order to train a neural network to perform some tasks, the error between the desired output and the actual output should be reduced by adjusting the weights of each unit. This process requires the computation of the error derivative of the weights. The output $HInd$ of a single neuron is calculated as the weighted sum of its input as:

$$\widehat{HInd} = b + \sum_{j=1}^{m} w_j x_j \tag{4.6}$$

where b is the bias and w_j is the weight corresponding to the input feature x_j. For the hidden layer, the output $HInd$ is passed through an activation function. The values of the bias and the weight parameters need to be determined to classify input vectors into two categories. The process of finding the parameters is called training the network. We trained the network using the Levenberg-Marquardt optimization algorithm [70] as it is the fastest backpropagation algorithm in the MATLAB Toolbox. We used a set of training data from our dataset to train the network. The training algorithm goes as follows:

- Select a network architecture.
- Initialize weight matrices to have small random values.
- Perform forward propagation to get initial prediction $HInd_i$ for any $x_i, i = 1, ..., n$.
- Calculate the error of the final layer.
- Do backpropagation and compute the error in each layer.
- Update the weights.
- Repeat the previous four steps on the whole training dataset.
- Stop the process when the error converges within an acceptable limit.

After completing training, we can test the model using new data and the desired output class can be determined. Based on the output, we can decide if an athlete is healthy or not. One of the disadvantages of neural networks is that they suffer from local minima. Training a neural network can also be computationally intensive.

4.3.5 Athlete Health Prediction Using Support Vector Machine

In binary classification, a support vector machine (SVM) is based on the idea of finding the best line or hyperplane that represents the largest separation or margin between the two classes [69]. If the training data are not linearly separable, SVMs handle this by using a non-linear kernel function to map the data into a different space. This technique is called kernel-trick [71]. For n training examples where each training example is a tuple of $(x_i, HInd_i)$, the margin separating

two classes can be maximized by solving the following optimization problem [72]:

$$\min_{w,b,\xi}\left(\frac{1}{2}w^T w + C\sum_{i=1}^{n}\xi_i\right) \tag{4.7}$$

Subject to

$$HInd_i\big(w^T\Phi(x_i)+b\big) \geq 1-\xi,\ \xi \geq 0 \tag{4.8}$$

where the function $\Phi(\cdot)$ maps training instances x_i into higher-dimensional space, w is a weight vector, b is a scalar bias, C is a soft-margin constant which regularizes the trade-off between training error and margin maximization, and ξ is an error parameter (or slack variable) to denote margin violation. The optimization problem in (4.7) and (4.8) can be solved by using the Lagrange multipliers method. After finding a solution, the resulting score function can be given by:

$$\hat{f}(x) = \sum_{i=1}^{n}\hat{\alpha}_i HInd_i K(x,x_i) + \hat{b} \tag{4.9}$$

where $\hat{\alpha}$ is an estimate of the Lagrange multiplier α and K denotes kernel function. After performing experiments with different kernel functions, we chose a Gaussian kernel for our data. Based on (4.9), we can predict AH for new observations.

4.3.6 Athlete Health Prediction Using Decision Tree

A decision tree (DT) is a graphical representation of the possible outcomes to a decision based on certain conditions [30, 69]. A decision tree usually starts with a single node. Then, it branches into possible outcomes. Each of the outcomes leads to new nodes, and it goes on. The

result is a tree-like structure with many nodes. A general algorithm for a decision tree can be stated as follows:

- Start with a training dataset which contains features and labels.
- Using a feature selection measure, determine the best feature which best splits the data.
- Make that feature the root node and break the dataset into smaller subsets.
- Start tree building by repeating this process recursively until a stopping criterion is met.
- Once the decision tree is trained, predict AH based on the new observations.

For the selection of the best split feature at each node, the standard CART algorithm is used which makes use of the Gini index as an impurity measure. Note that decision trees still do not find wide acceptance because of their problem with overfitting.

4.4 Parameter Tuning for ML Methods

Model selection varies depending on the type and the number of parameters. Setting parameters for each model leads to good results. Four of our methods described in this chapter are associated with a few parameters that need to be fine-tuned during training for the best possible performance. For example, for NB, we need to decide the type of distribution function, the type of kernel, and the width of the window. For k-NN, the number of k neighbors and the distance function must be known. For ANN, we need to find the number of hidden layers and the number of neurons for each hidden layer. The network training function and the number

Table 4.1: Parameters for experiments and the optimal parameters for each model.

Model	Parameters for experiments	Optimal parameter
LR	—	—
ANN	Hidden layers $\in \{1, 2, ..., 3\}$	1
	Hidden neurons $\in \{3, 7, ..., 51\}$	11
	Network training functions: Levenberg-Marquardt, Bayesian regularization, Gradient descent	Levenberg-Marquardt
NB	Data distributions: Kernel smoothing density estimate, Gaussian	Kernel
	Kernel smoother type: Box, Gaussian, Triangle	Gaussian
	Kernel smoothing window width $\in \{0.001, 0.01, ..., 1\}$	0.01
k-NN	$k \in \{1, 3, ..., 25\}$	k=3
	Nearest neighbor search methods: Kdtree, Exhaustive	Exhaustive
	Distance metrics: Euclidean, Manhattan, Minkowski	Manhattan
SVM	Kernel functions: Linear, Gaussian, Polynomial	Gaussian
	Kernel scale $\in \{0.1, 0.5, ..., 2\}$	0.6
	Regularization parameter, C $\in \{0, 0.5, ..., 5\}$	1
DT	—	—

of epochs for the backpropagation algorithm should also be determined. For SVM, we need to find the type of kernel function, kernel scale, and the regularization parameter.

One quick way to evaluate the performance of a model is to split the dataset into two sets: the training set and the test or validation set. It is common to use 80% of the data for training and 20% for testing [31]. A parameter grid was used in order to find the optimal setting using the test set. Table 4.1 shows the parameters used for the experiments and the optimal parameters for each model.

After obtaining the optimal parameters, the evaluation of the models was conducted through 10-fold cross-validation to avoid overfitting. In the next sections, we will present the results and will compare the performance of the classifiers using ROC and performance evaluation metrics.

Table 4.2: Performance measures for various ML models (seven omit days).

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.625	0.870	0.251	0.232	0.933	0.241
ANN	0.640	0.859	0.227	0.245	0.919	0.235
NB	0.829	0.835	0.291	0.605	0.857	0.393
k-NN	0.752	0.909	0.471	0.278	0.970	0.350
SVM	0.813	0.936	0.845	0.333	0.994	0.478
DT	0.774	0.905	0.459	0.416	0.952	0.436

4.5 Model Evaluation Results and Analysis

In this section, we evaluate the models considering all the features. To perform 10-fold cross-validation, for the NB classifier, we used the Gaussian kernel with a window width of 0.01. For the k-NN classifier, we chose the three nearest neighbors and the Manhattan distance function to measure the distance. For the ANN classifier, one hidden layer with 11 neurons were selected. We also used Levenberg-Marquardt as the network training function for the backpropagation algorithm to train the network. For the SVM classifier, the Gaussian kernel function with a kernel scale of 0.6 and the regularization parameter of one were chosen.

Tables 4.2 and 4.3 summarize the performance measures of various classifiers for the seven omit days and the fourteen omit days health labels, respectively. In terms of the accuracy metric, the classifiers seem to perform reasonably well with SVM giving the best performance with an accuracy of 93.6%. Whereas, NB gave the lowest accuracy of 83.5%. Note that for the seven omit days output labels, our data contain around 91% negative cases and only 9% positive cases. This may mean that some classifiers act like naive predictors which predict majority cases most of the time and hence show high accuracy which may be confusing. Therefore, accuracy

Table 4.3: Performance measures for various ML models (fourteen omit days).

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.578	0.647	0.189	0.441	0.682	0.265
ANN	0.602	0.656	0.199	0.449	0.691	0.274
NB	0.830	0.805	0.393	0.642	0.833	0.488
k-NN	0.767	0.860	0.523	0.350	0.946	0.420
SVM	0.819	0.899	0.880	0.346	0.992	0.497
DT	0.789	0.863	0.526	0.510	0.923	0.518

cannot be a good evaluation metric for skewed data. Instead, we should investigate other metrics such as AUC and F-score.

Now, if we see other metrics in Table 4.2, LR and ANN classifiers performed poorly with AUCs of only 0.625 and 0.64, respectively. It is because LR is a generalized linear model and it is almost impossible for real-life data to be linearly separable. However, we can use the LR model as a baseline for comparison. The poor performance of ANN is probably due to the local minimum problem. On the other hand, NB and SVM performed better than other classifiers in terms of AUC and F-score. The k-NN and DT classifiers showed average performance.

For a clear view of the overall predictive performance of the models, we also plot the corresponding ROC curves that are shown in Figures 4.2 and 4.3. The horizontal axis corresponds to the false positive rate (FPR) and the vertical axis represents the true positive rate (TPR) or sensitivity. The characteristic of a good classifier is that it will produce high TPR that corresponds to low FPR. Thus, at any given FPR, an ROC curve with a higher TPR corresponds to better classification performance. The computed AUC scores show that LR and ANN indeed performed very poorly, whereas NB yielded the best results with an AUC of 0.83. SVM, DT, and k-NN classifiers produce good performance with an AUC of 0.813, 0.774, and 0.752, respec-

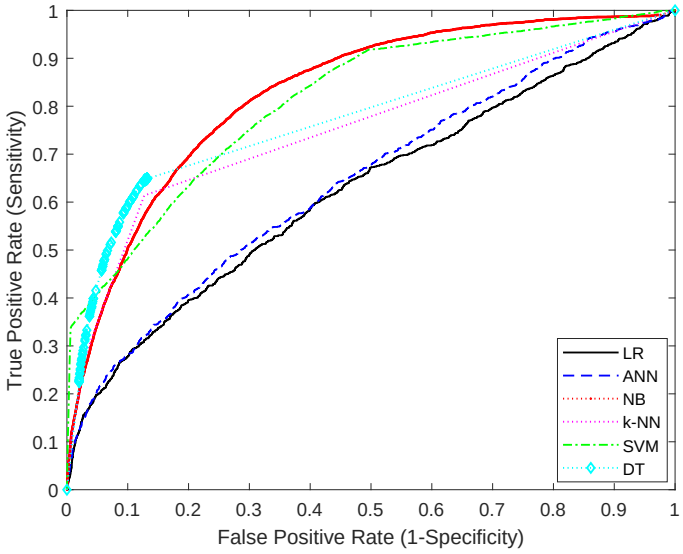

Figure 4.2: ROC curves for various ML models (seven omit days).

tively. However, the overall performance of these classifiers is not that impressive. Note that AUC measures the ability of the model to correctly classify sick and healthy classes, where an AUC of 1.0 represents perfect classification performance and 0.5 means random prediction. Because each point on a ROC curve corresponds to a threshold for distinguishing between healthy and sick, the selected threshold could be set according to the individual needs of the athlete. Note that depending on our requirements, we can adjust precision and recall (hence F-score) by adjusting the threshold.

In summary, we evaluated six traditional ML classifiers using 10-fold cross-validation. The LR and ANN classifiers performed very poorly. Other classifiers did not perform very well either. This means that these classifiers are probably not good for imbalanced data. It is also clear from the results that the accuracy metric cannot be used as a measure for imbalanced classes. Therefore, we should give emphasis on other metrics for evaluation purposes.

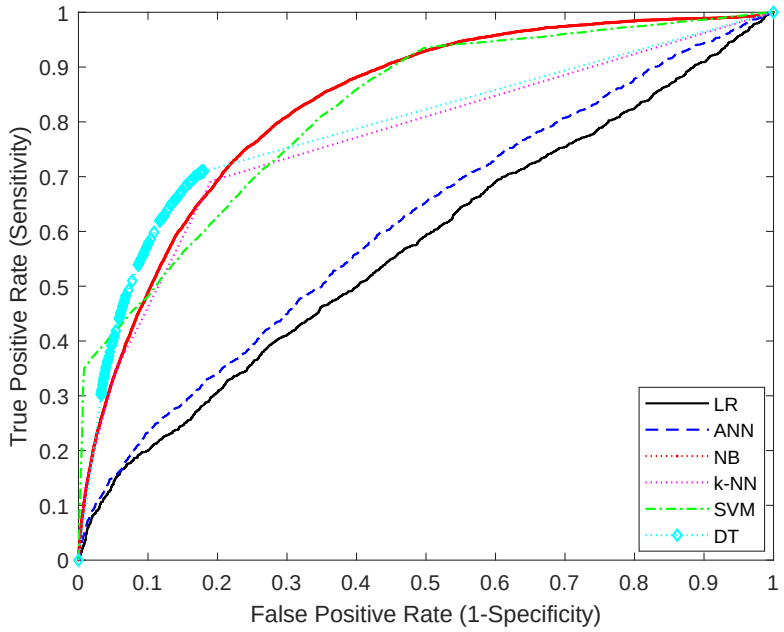

Figure 4.3: ROC curves for various ML models (fourteen omit days).

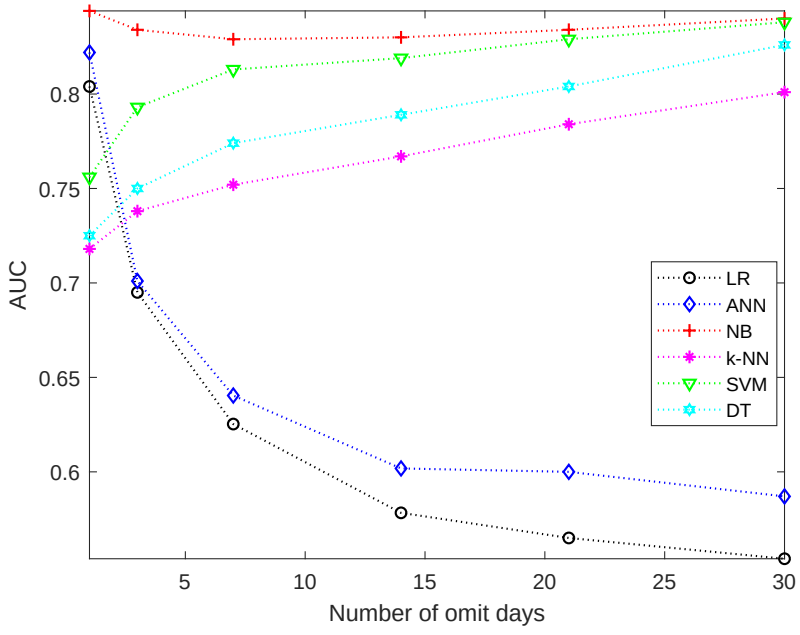

Figure 4.4: Comparison of AUC for different omit days.

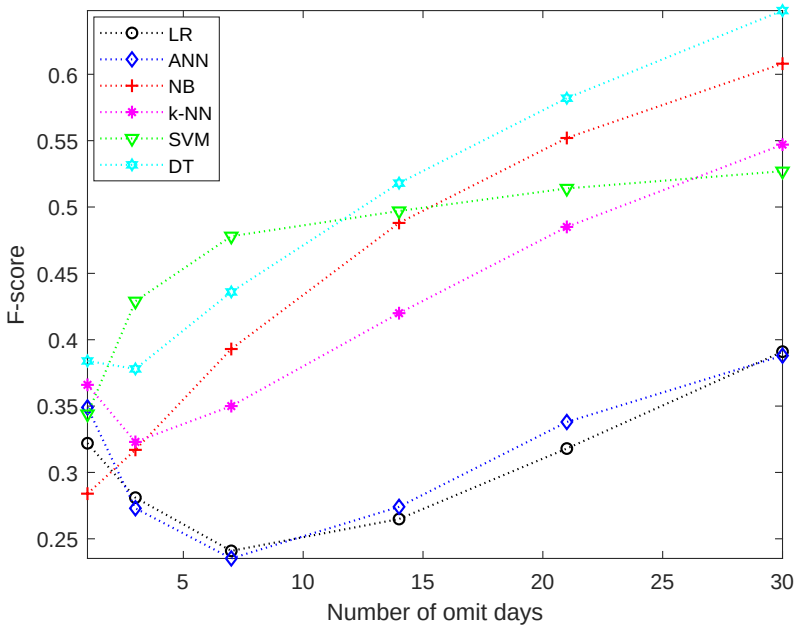

Figure 4.5: Comparison of F-score for different omit days.

We also present the results showing a relationship between AUC and the number of omit days to generate output labels which are shown in Figure 4.4. In a similar manner, Figure 4.5 shows the relationship between the F-score and the number of omit days. As can be seen in Figure 4.4, in the case of the NB model, AUC values are almost the same in all cases. Whereas, for k-NN, SVM, and DT, AUC values increase slightly with the increase of the number of omit days. However, for LR and ANN, AUC values decrease rapidly from day one to day fourteen, and then there is a gradual decrease of AUC. This was expected because as the number of omit days is increased, the number of positive cases decreases, which makes the data more imbalanced. LR and ANN performed poorly in classifying imbalanced data. Now, if we investigate Figure 4.5, it is obvious that the F-score values increase gradually from the third day onward for all the classifiers.

Table 4.4: Regression equations for LR models for different omit days.

Omit days	Regression equation
1	$\widehat{HInd} = -4.192 - 0.060x_1 + 0.059x_2 - 0.080x_3 - 0.066x_4 + 0.019x_5 - 0.023x_6$ $-0.153x_7 + 0.051x_8 - 0.942x_9 - 0.015x_{10} - 0.091x_{11}$
3	$\widehat{HInd} = -3.205 - 0.056x_1 + 0.002x_2 - 0.021x_3 - 0.045x_4 - 0.006x_5 - 0.051x_6$ $-0.065x_7 + 0.017x_8 - 0.646x_9 - 0.022x_{10} - 0.053x_{11}$
7	$\widehat{HInd} = -1.810 - 0.033x_1 - 0.032x_2 + 0.025x_3 - 0.032x_4 - 0.002x_5 - 0.074x_6$ $+0.003x_7 + 0.002x_8 - 0.257x_9 + 0.002x_{10} - 0.045x_{11}$
14	$\widehat{HInd} = -2.427 - 0.023x_1 - 0.008x_2 - 0.014x_3 - 0.041x_4 - 0.002x_5 - 0.062x_6$ $-0.017x_7 - 0.004x_8 - 0.426x_9 - 0.004x_{10} - 0.044x_{11}$
21	$\widehat{HInd} = -1.470 - 0.032x_1 - 0.024x_2 + 0.015x_3 - 0.042x_4 - 0.005x_5 - 0.094x_6$ $+0.012x_7 - 0.012x_8 - 0.205x_9 + 0.0024x_{10} - 0.050x_{11}$
30	$\widehat{HInd} = -1.168 - 0.027x_1 - 0.019x_2 + 0.049x_3 - 0.032x_4 - 0.021x_5 - 0.099x_6$ $-0.003x_7 + 0.001x_8 - 0.149x_9 + 0.0001x_{10} - 0.055x_{11}$

Note that we pruned our DT model for different levels. The difference between before pruning and after pruning was not sufficiently significant to be included in the thesis. However, pruning reduced the model complexity.

The estimated regression equations are tabulated in Table 4.4 which can be used to reproduce the results presented in this chapter for the LR models. This Table shows the number of omit days to generate output labels and the corresponding regression equation for our dataset considering all the eleven features. Similarly, Table 4.5 presents the bias vectors, the weight matrices, and the regression equation for the ANN algorithm for seven omit days health label.

4.6 Summary and Discussion

So far, we presented the necessary methods and techniques for data cleaning and preparation. We also addressed some important issues related to ML methods that are useful for improving

Table 4.5: Bias vectors and weight matrices for ANN algorithm.

Coefficient	Values
B_2	-0.7786
B_1	-1.884 -1.380 -0.936 -1.550 0.210 0.021 -0.471 0.720 0.655 -1.072 1.762
IW	-0.191 1.177 0.181 -0.499 1.419 -0.546 0.224 0.561 -0.014 0.724 0.515 0.784 0.408 -0.411 0.922 -0.654 0.307 0.609 0.579 0.133 0.248 0.155 1.552 -0.795 -0.242 0.262 -0.748 -1.549 -0.727 -0.050 -2.836 -0.073 0.802 0.060 0.184 -0.098 -0.020 -1.448 -0.484 -0.916 0.063 0.051 -0.014 0.232 -0.910 -0.391 -0.311 -1.779 0.506 0.007 0.186 -0.613 0.331 0.530 -0.186 1.024 -0.343 -0.168 -0.067 -1.040 -2.521 -1.137 0.168 0.301 -0.413 -0.736 -0.357 0.321 0.208 -0.456 1.119 0.311 -0.822 -0.423 -0.411 0.462 -0.468 0.352 -0.198 -0.666 -0.677 -0.483 0.466 -0.790 -0.602 -0.453 -0.505 0.010 1.198 -0.633 0.782 0.806 -1.578 1.150 -1.174 0.081 7.514 -0.218 -0.189 -0.780 0.997 -0.176 -0.990 1.406 -2.000 2.681 -1.005 -3.774 0.161 0.201 0.978 -1.528 -0.962 0.757 -0.479 0.201 -2.039 1.154 0.246 -0.110 -0.539
LW	-0.939 0.351 0.090 -0.609 0.136 0.177 0.044 -0.119 -0.858 -1.121 -1.368
Regression equation: $\widehat{HInd} = B_2 + LW * tanh(B_1 + IW * x)$ where B_2 is the output bias, B_1 is the input bias vector, IW is the input weight matrix, and LW is the layer weight matrix	

the classification performance. The concept of machine learning, its applications, and the supervised ML methods which were used for our data analysis have also been introduced.

In this chapter, we first introduced six ML methods in order to predict AH. Then, we applied these methods to our dataset and presented results in various forms. We have been able to show that the ML classifiers are useful for the prediction of AH. We compared each classifier in terms of AUC, accuracy, F-score, ROC, and other metrics. The NB model gave superior performance to other models in terms of AUC. However, if we investigate other metrics, none of the classifiers performed very well in classifying the two imbalanced classes. This is probably because these classifiers are not good in handling imbalanced data effectively.

In the next chapter, advanced ensemble-based methods are investigated in order to further improve the classification performance of the imbalanced data.

Chapter 5

Improving Classification Performance Using Ensemble Methods

In the previous chapter, traditional ML methods were used to predict athletes' health (AH). Using these methods, we achieved some good results. However, the results may not be satisfactory in applications in which a higher classification performance is expected. Most of the traditional ML methods failed to provide sufficiently high performance. This is probably because our dataset has an unbalanced distribution of classes between the healthy class and the unhealthy (sick) class. The healthy cases outnumbered the sick ones by a margin of 11:1.

In this chapter, we aim to explore different approaches to handling the class-imbalanced dataset in order to improve classification performance. As discussed earlier, several strategies

exist for dealing imbalanced data for classifier training, including resizing the training dataset and adjusting misclassification costs. Ensemble-based ML methods are used for these purposes. Ensemble methods work by combining predictions from multiple models that are trained on samples from the original dataset. Tree-based ensemble techniques can be either bagging-based trees or boosting-based trees. In bagging-based trees, the data is divided into N samples with replacement. Then, out of N samples, an N number of trees are built to be combined to train the model. Random forest is a special case of the bagging technique where a fraction of the predictor variables is used for each tree. In boosting-based trees, however, prediction accuracy is improved in each iteration by focusing on the misclassified instances from the previous iteration.

The remainder of this chapter is organized as follows. Sections 5.1 and 5.2 introduce the ensemble-based ML methods which are used in this thesis. Section 5.3 addresses the parameter tuning for ensemble methods. Section 5.4 presents experimental results and analysis to evaluate the performance of the models. The effects of removing insignificant features is demonstrated in Section 5.5. Section 5.6 presents evaluation results after eliminating the health feature. To further improve the prediction performance, Section 5.7 presents results using the ARMA model. Section 5.8 gives a comparative analysis of training time for each model. Section 5.9 summarizes the chapter with discussion.

5.1 Athlete Health Prediction Using Boosting Algorithms

Boosting is a sequential ensemble technique that strategically combines a collection of weak learners to form a stronger model. It is suitable for low variance and high bias (i.e., under-fitting) models. In this thesis, we consider the three most popular ensemble methods such

as AdaBoost (Adaptive Boosting), LogitBoost (Adaptive Logistic Regression), and RUSBoost (Random Undersampling Boosting) for our imbalanced data. These boosting algorithms are briefly explained below.

AdaBoost (AB) is a popular boosting algorithm proposed by Freund and Schapire for binary and multiclass classifications. It aims to convert a set of weak classifiers into a strong one. It is used with a short decision tree, and it changes the weight of samples to let the algorithm pay more attention to wrongly classified samples. For example, training data that is hard to predict is given more weight, whereas easy to predict instances are given less weight. Following these rules, models are created sequentially one after the other, each updating the weights of the training instances that affect the learning performed by the next tree in the sequence. After a certain number of iterations, we can get the final prediction by a majority vote over the weighted predictions of weak classifiers. The AdaBoost algorithm goes as follows. Given a training dataset $(x_i, HInd_i)$ containing n observations:

- Initialize the weight for each data point as $w(x_i, HInd_i) = 1/n, i = 1, ..., n$.
- For iteration $r = 1, ..., T$, fit weak classifiers to the data and select the one with lowest weighted classification error ε_r.
- Calculate the weight for the $r - th$ weak classifier as $\alpha_r = \frac{1}{2} ln \frac{1-\varepsilon_r}{\varepsilon_r}$.
- Update the weights.

- After finishing the training, compute prediction for new data using

$$\hat{f}(x) = sign\left(\sum_{r=1}^{T} \alpha_r h_r(x)\right) \tag{5.1}$$

  where $h_r(x)$ is the output of weak classifier r for input x.

LogitBoost (LB) is one of the boosting algorithms for binary classification. It works similarly to AdaBoost, except it optimizes the cost function of logistic regression.

RUSBoost (RUSB) is also a boosting algorithm particularly effective in handling the class imbalance in the training data with discrete class labels. As such, this algorithm is useful for our imbalanced data. The RUSB algorithm combines the undersampling and the standard boosting procedure AdaBoost, providing a simple and efficient method for improving classification performance when training data is imbalanced. The algorithm reduces majority class samples through undersampling. It is computationally less expensive than other boosting algorithms and results in a simpler algorithm with faster model training time [73, 74]. A detailed description of these algorithms can be found in [75, 76, 77].

5.2 Athlete Health Prediction Using Random Forest Classifier

Random forest (RF) is an ensemble-based learning method used for classification and regression purposes [78]. It is relatively new but is one of the most widely used algorithms for classification. The RF algorithm starts with the technique called decision tree. For a given dataset, the RF algorithm builds a set of decision trees. Each tree is developed individually from a bootstrap sample from the training dataset. The idea is to combine many decision trees into a single

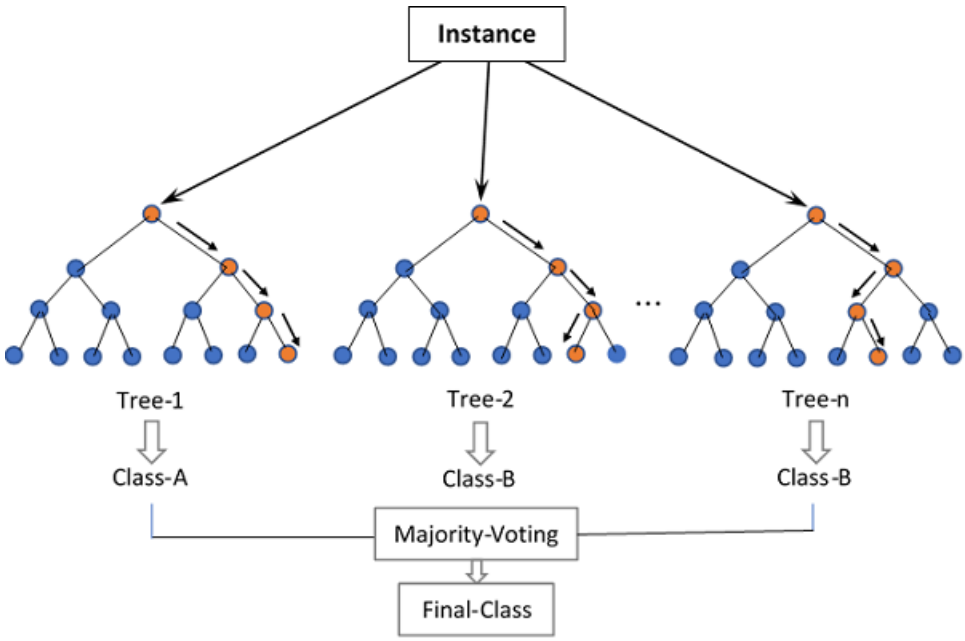

Figure 5.1: An illustration of random forests. (Credit: William Koehrsen)

model. The prediction of the RF algorithm is obtained by a majority vote over the predictions of the individual trees.

RF is a robust machine learning algorithm. It reduces the overfitting problems associated with decision trees by averaging different outputs of the individual decision trees. The logic behind this is that each of the models used is weak when employed on its own but gets stronger when put together in an ensemble. The RF algorithm can also deal with imbalanced classes [79] and missing data. Figure 5.1 shows an example of the RF algorithm. The main parameters to be determined for the RF algorithm are the number of trees and the number of features to use for each tree. The algorithm can be stated as follows:

- Take N random samples to create N subsets of the data.
- At each node, randomly select k features from total m features, where $k < m$. The feature

that provides the best split is used to do a binary split on that node. A common setting for k is equal to the square root of m.

- At the next node, randomly choose another set of k features and do the same.
- Repeat the previous steps to create N number of trees.
- Predict new examples by a majority vote over the predictions of the N trees.

We applied the above-mentioned ensemble methods to our dataset in order to improve classification performance. In the next section, we will address parameter tuning of these methods.

5.3 Parameter Tuning for Ensemble Methods

The ensemble methods described above are associated with some parameters that need to be tuned to achieve the best possible performance. For example, for AB, LB, and RUSB algorithms, we need to determine the number of learning cycles for boosting. The learning rates for these algorithms should also be determined. For the RF algorithm, we need to select the number of trees (NoT) for the forest and the number of features (NoF) to be used in random selection in order to grow each tree. NoF must be lower than the total number of features in the dataset [78]. We applied the same rules and technique as in Section 4.4 for training and testing the ensemble models to find the optimal parameters.

Figures 5.2, 5.3, and 5.4 respectively show a series of AB, LB, and RUSB models with different numbers of learning cycles. In the case of AB and LB models, the ensemble achieves

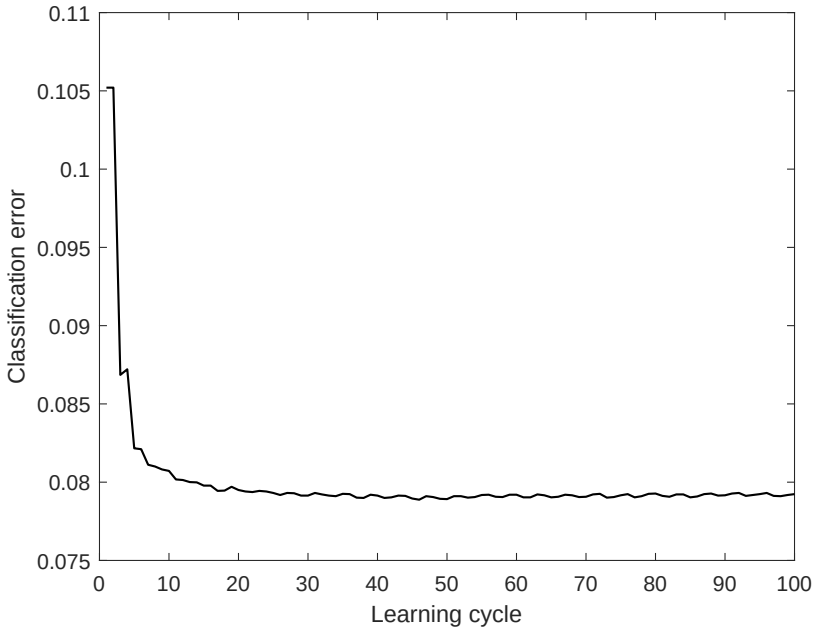

Figure 5.2: Impact of number of cycles on the classification error for AdaBoost.

an error rate of around 7.5% after accumulating about 30 weak learners. For the RUSB model, the ensemble achieves an error rate of around 14% using 30 or more weak learners. In all cases, however, after approximately 50 learning cycles, no significant benefit is gained by including more weak learners. Thus, to apply these models, we used 90 learning cycles which is a sufficiently large number. For these algorithms, we also selected 0.1 for the learning rate through the trial and error method. For the RUSB algorithm, we experimented with different values of sampling proportion with respect to the minority class and finally came up with 1 as the best value.

In the case of the RF algorithm, we optimized two important parameters, NoT and NoF. We first set the NoF as three (square root of the number of features), and then plotted the error for the different number of trees as shown in Figure 5.5. After about 200 trees, the error decreases

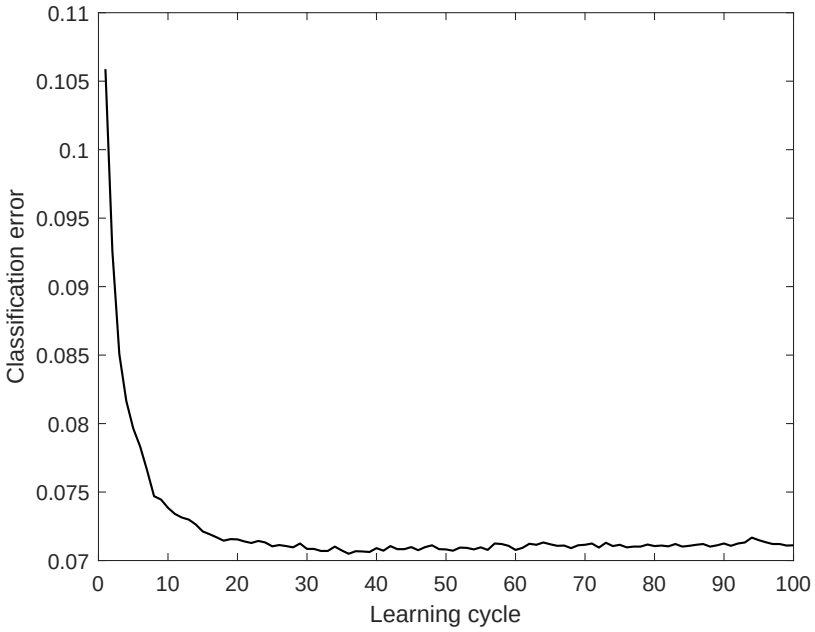

Figure 5.3: Impact of number of cycles on the classification error for LogitBoost.

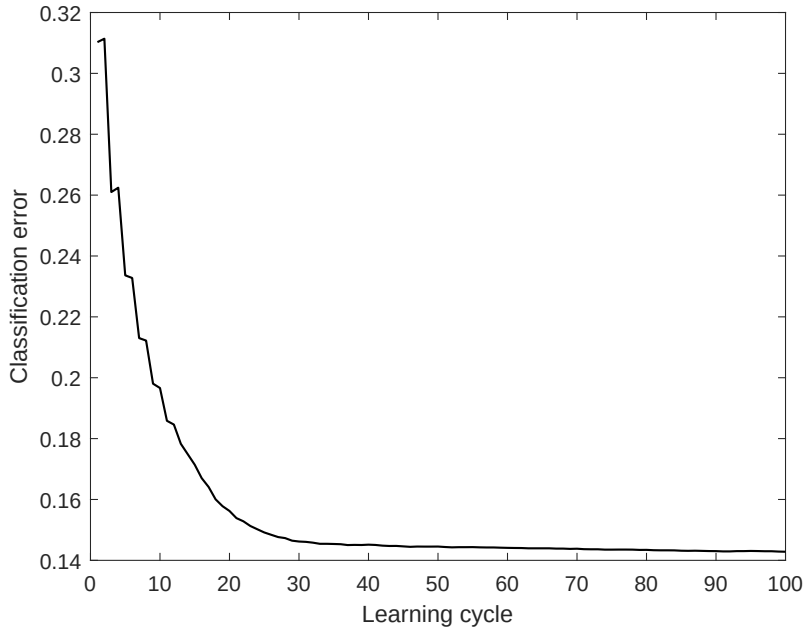

Figure 5.4: Impact of number of cycles on the classification error for RUSBoost.

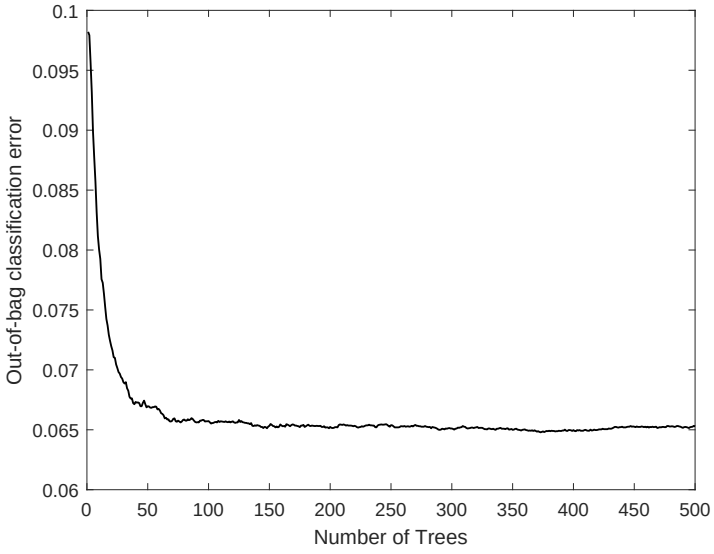

Figure 5.5: Impact of number of trees on the classification error for RF.

very slowly with the number of trees. This slow rate ensures that no substantial benefit is achieved by including more than 200 trees. Thus, we used 300 trees for our experiments which can be considered a large number. For this algorithm, we also needed to determine the NoF. To do that, we set the NoT as 300, and then the error for the different number of NoF are computed which can be seen in Figure 5.6. Since a total of 11 features was used, a total of 11 RF models was constructed to find the best NoF. The far right of Figure 5.6 shows the results of the bagging approach, where all the 11 features were used. The minimum error rate was obtained with two features in use. However, we selected three features for this thesis.

Table 5.1 shows the parameters used for the experiments and the optimal parameters obtained for each ensemble model. Using these parameters, we evaluated the models through 10-fold cross-validation. In the next sections, we will present results and analysis of each classifier using various performance evaluation metrics.

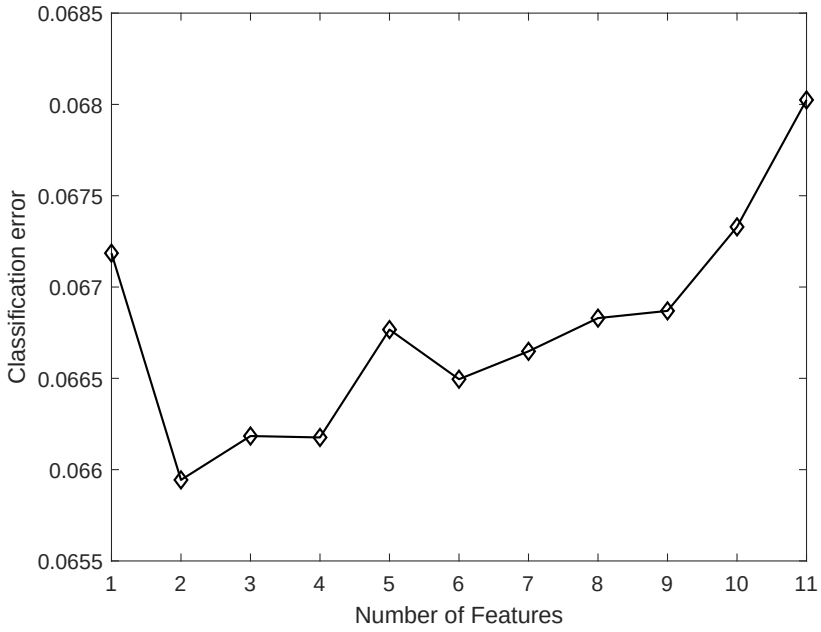

Figure 5.6: Impact of number of features on the classification error for RF.

Table 5.1: Parameters for experiments and the optimal parameters for each ensemble model.

Model	Parameters for experiments	Optimal parameter
AdaBoost	Number of ensemble learning cycles $\in \{10, 20, ..., 100\}$	90
(AB)	Learning rate $\in \{0.05, 0.1, ..., 1\}$	0.1
LogitBoost	Number of ensemble learning cycles $\in \{10, 20, ..., 100\}$	90
(LB)	Learning rate $\in \{0.05, 0.1, ..., 1\}$	0.1
RUSBoost	Number of ensemble learning cycles $\in \{10, 20, ..., 100\}$	90
(RUSB)	Learning rate $\in \{0.05, 0.1, ..., 1\}$	0.1
	Sampling proportion with respect to the minority class $\in \{1, ..., 5\}$	1
Random Forest	Number of trees $\in \{10, 20, ..., 500\}$	300
(RF)	Number of features to select at random $\in \{1, 2, ..., 11\}$	3

5.4 Combined Results and Discussion Considering All Features

In this section, we evaluate the models considering all the features. In the next section, we will examine the models after removing insignificant features from the dataset. We will then test our models after eliminating the health feature from the dataset.

At first, we conducted experiments using the SMOTE algorithm which creates synthetic samples from the minority class to overcome the problem with imbalanced data. We experimented with oversampling the minority (sick) class in the training set. However, the oversampling approach did not improve the classification performance when compared with no oversampling. Moreover, oversampling makes the algorithms computationally intensive. Therefore, we did not include these results in the thesis. We then further experimented with undersampling the majority class in the training set. We selected the RUSB algorithm for this purpose. The RUSB algorithm combines undersampling and boosting techniques for improving classification performance when training data contains imbalanced classes. It is also computationally much faster than the SMOTE algorithm. The results are summarized in Tables 5.2 and 5.3 for all the classifier models. For a better visualization, the results are also presented in Figures 5.7 and 5.8 in the form of ROC curves.

The results demonstrate that the ensemble-based methods outperform the traditional methods by a clear-cut margin in terms of AUC and F-score. In particular, the RF model shows the best performance in terms of AUC followed by RUSB, LB, AB, NB, SVM, DT, k-NN, ANN, and LR. In the case of traditional ML methods, LR and ANN are similar in performance, whereas k-NN and DT are similar. For ensemble methods, AB and LB show similar

Table 5.2: Comparative analysis of various ML models (seven omit days).

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.625	0.870	0.251	0.232	0.933	0.241
ANN	0.640	0.859	0.227	0.245	0.919	0.235
NB	0.829	0.835	0.291	0.605	0.857	0.393
k-NN	0.752	0.909	0.471	0.278	0.970	0.350
SVM	0.813	0.936	0.845	0.333	0.994	0.478
DT	0.774	0.905	0.459	0.416	0.952	0.436
AB	0.877	0.933	0.689	0.433	0.981	0.531
LB	0.891	0.932	0.683	0.435	0.980	0.531
RUSB	0.901	0.871	0.381	0.736	0.884	0.502
RF	0.905	0.937	0.793	0.385	0.990	0.518

Table 5.3: Comparative analysis of various ML models (fourteen omit days).

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.578	0.647	0.189	0.441	0.682	0.265
ANN	0.602	0.656	0.199	0.449	0.691	0.274
NB	0.830	0.805	0.393	0.642	0.833	0.488
k-NN	0.767	0.860	0.523	0.350	0.946	0.420
SVM	0.819	0.899	0.880	0.346	0.992	0.497
DT	0.789	0.863	0.526	0.510	0.923	0.518
AB	0.892	0.899	0.708	0.507	0.965	0.591
LB	0.904	0.900	0.713	0.518	0.965	0.600
RUSB	0.911	0.873	0.545	0.739	0.896	0.627
RF	0.914	0.904	0.802	0.448	0.981	0.575

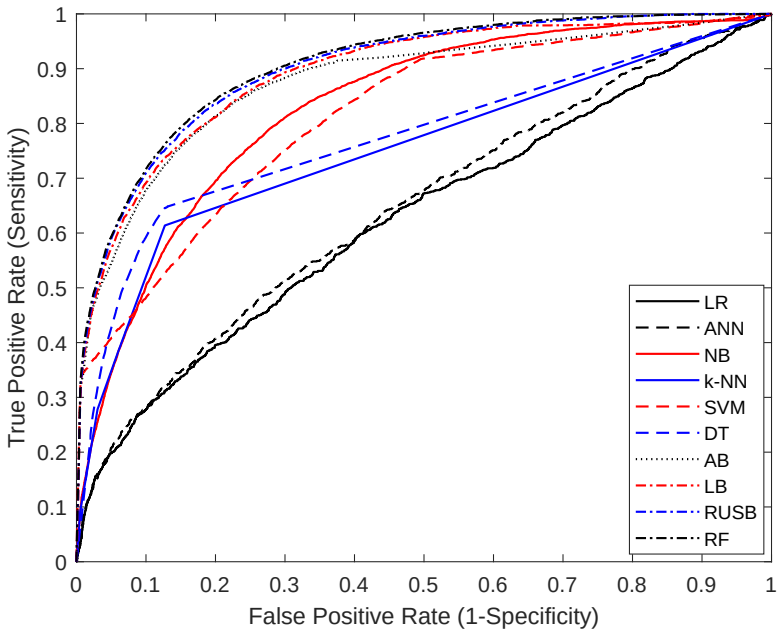

Figure 5.7: ROC curves for various models (seven omit days).

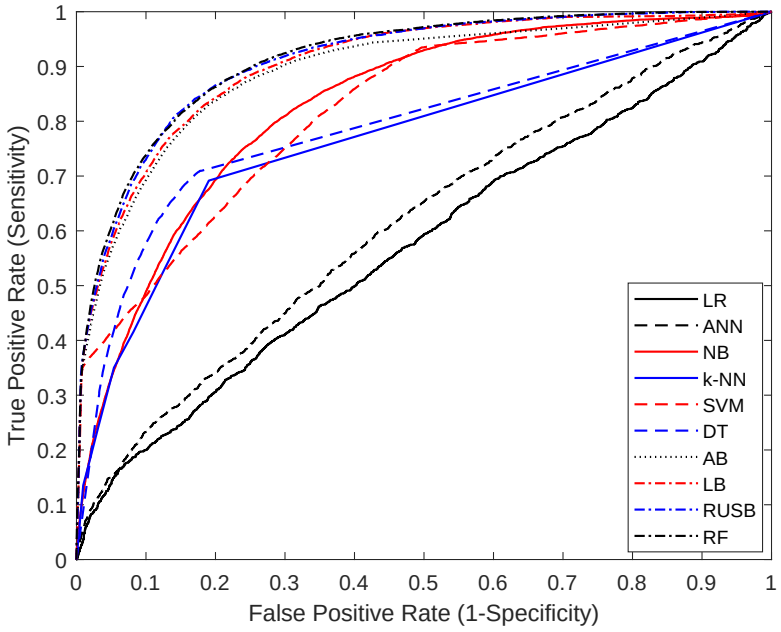

Figure 5.8: ROC curves for various models (fourteen omit days).

performance, whereas RUSB and RF are very close; however, the latter group outperformed the former with a small margin.

The superior performance by ensemble methods over traditional methods is not surprising. As explained earlier, the dataset used in our study contains imbalanced classes. In such a case, traditional ML methods are unable to separate imbalanced classes with reasonable accuracy, whereas ensemble methods are designed to handle this kind of data. Among the four ensemble methods used in this study, the RUSB model achieved the best performance in terms of recall. However, this was to be expected. It was explained that the RUSB algorithm uses downsampling and boosting techniques which ensures the robustness of the model. The worst classification performance comes from the LR model which assumes a linear separation of the two classes. Note that there are 5893 sick (positive) cases in the dataset. The LR model predicted only 1348 positive cases correctly, resulting in a recall/sensitivity of a mere 23.2%. Whereas, the RUSB model predicted 4370 positive cases correctly with a recall/sensitivity of 73.6%, which is 50.4% more than the LR model.

Figures 5.9 and 5.10 show how AUC and F-score vary with the number of omit days to generate output labels. As can be seen, AUC does not vary much over the number of omit days in the case of the ensemble methods, NB, k-NN, SVM, and DT. The only exceptions are LR and ANN, where AUC decreases as the number of omit days increases up to day 15 and then it remains almost the same. Regarding Figure 5.10, especially after seven omit days, the F-score increases with the increase of the number of omit days in all cases.

The model comparison results can also be seen as bar graphs in Figures 5.11 and 5.12 in terms of AUC and F-score, respectively. These results are obvious and self-explanatory. From

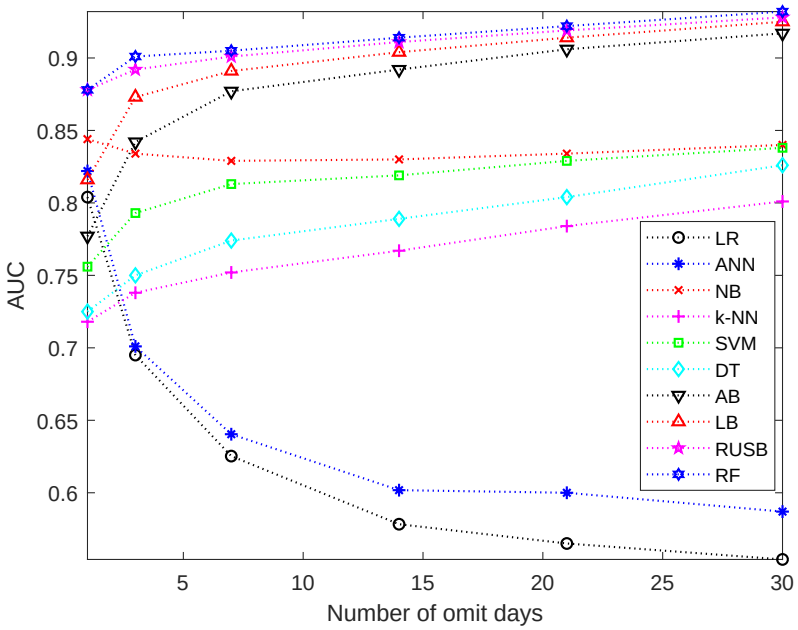

Figure 5.9: Comparison of AUCs for different omit days.

these graphs we can easily identify the best model and the worst model based on AUC and F-score values.

We note here that the precision and the recall values depend on the selection of the threshold based on which the healthy and sick cases are distinguished. By changing the threshold, we can adjust the precision and the recall, and hence the F-score. As far as the specificity is concerned, it is the true negative rate that we are less concerned with. In the case of our imbalanced data, we gave more emphasis on the AUC and the F-score metrics.

5.5 Evaluation Results After Eliminating Insignificant Features

So far, we have considered all the features for our dataset. In this section, we present model evaluation results after eliminating insignificant features that we have discussed in Chapter 3.

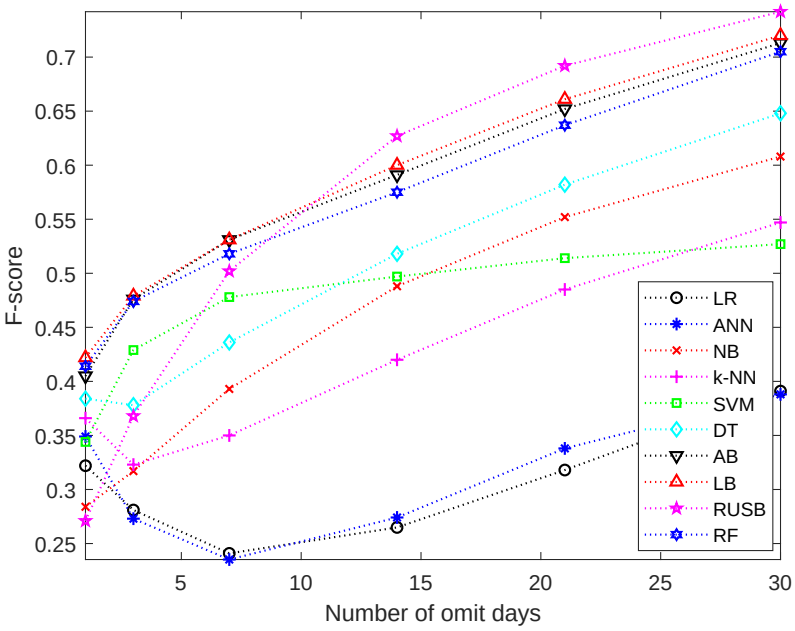

Figure 5.10: Comparison of F-scores for different omit days.

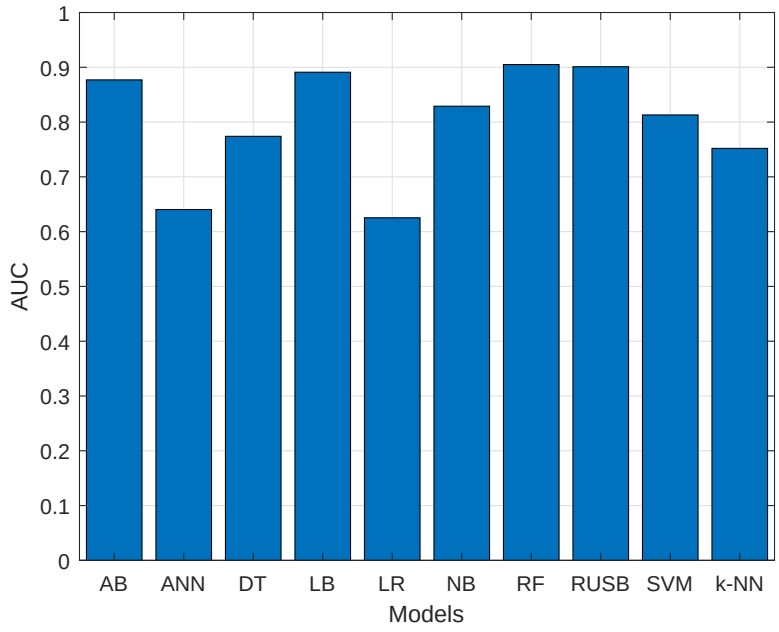

Figure 5.11: Model comparison results in terms of AUC.

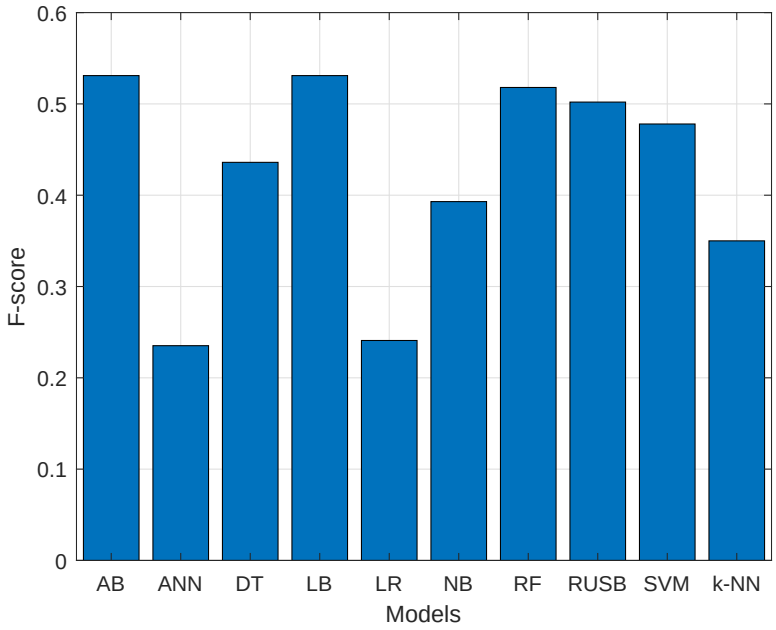

Figure 5.12: Model comparison results in terms of F-score.

Starting from all the eleven features, we eliminated one insignificant feature at a time. The results are shown in Figures 5.13 and 5.14 in terms of AUC and F-score, respectively, for the seven omit days to generate output labels.

Firstly, it is clear from the results that the performance of most of the classifiers decreases slowly with the decrease of the number of features (from right to left). This was expected, since information is reduced in the dataset after removing the insignificant features. Therefore, as far as our dataset is concerned, it is probably not a good idea to remove the insignificant features. In the case of DT, however, the AUC increases gradually with the removal of the insignificant features. Therefore, the DT model can be a good choice when we want to eliminate insignificant features.

Secondly, even after removing insignificant features, the results demonstrate that the en-

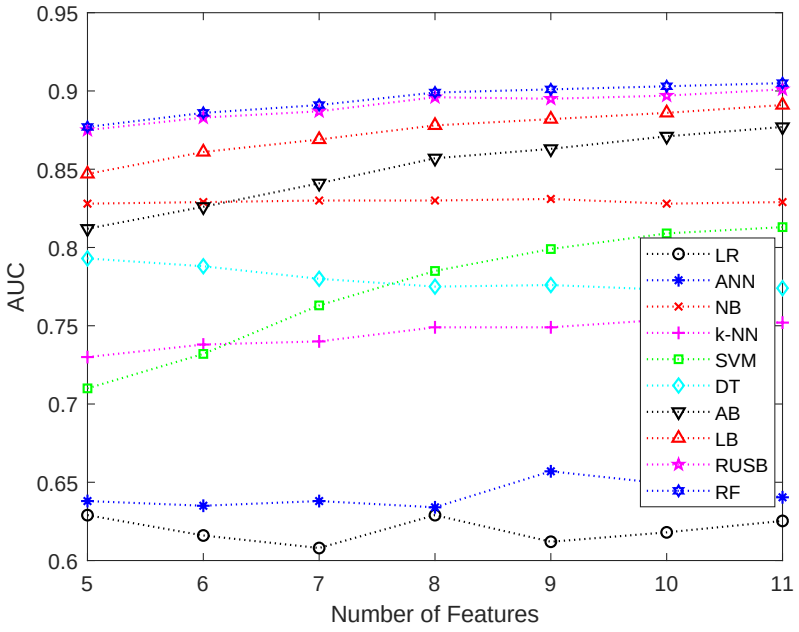

Figure 5.13: Impact of number of features on AUC for various models.

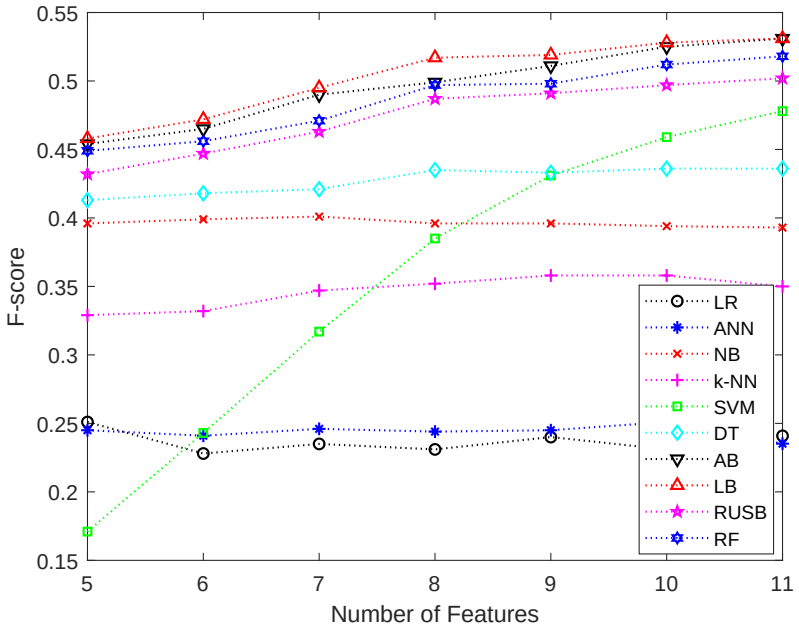

Figure 5.14: Impact of number of features on F-score for various models.

semble models still outperform the traditional models. Similarly, AB and LB yield similar performance, whereas RUSB and RF are very similar. It is also obvious that the latter group (RUSB and RF) performs better than the former group (AB and LB).

Based on our experiments, it can also be said that removing insignificant features makes the algorithms faster with little penalty. Therefore, depending on our requirements, we can use a trade-off strategy of removing or not removing the insignificant features from a dataset.

5.6 Evaluation Results After Removing the Health Feature

Up to this point, we have presented the results by including the health feature in the dataset based on which the output labels were generated. To check the impact of the health feature on the results, we tested the models after eliminating this feature. Table 5.4 and Figure 5.15 present the results after removing the health feature from the dataset. Comparing the results in Tables 5.2 and 5.4, it can be concluded that the overall performance of the models has been degraded a bit after removing the health feature. However, the difference is not sufficiently significant to be mentioned.

5.7 Performance Evaluation Results Using the ARMA Model

All the results presented thus far were produced by using the present day's observations to predict the next day's health. In this section, we also consider the past six days observations in order to further improve the prediction performance. The concept here is that the next value will be dependent on the previous observations. This may improve the prediction performance.

Table 5.4: Evaluation results for various models after removing the health feature.

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.577	0.896	0.210	0.063	0.977	0.096
ANN	0.597	0.868	0.178	0.129	0.941	0.148
NB	0.816	0.836	0.282	0.553	0.864	0.373
k-NN	0.738	0.905	0.439	0.258	0.968	0.325
SVM	0.794	0.933	0.823	0.309	0.994	0.450
DT	0.750	0.900	0.424	0.373	0.951	0.397
AB	0.863	0.928	0.647	0.409	0.978	0.501
LB	0.881	0.930	0.673	0.406	0.981	0.507
RUSB	0.886	0.851	0.338	0.714	0.864	0.459
RF	0.892	0.934	0.773	0.359	0.990	0.491

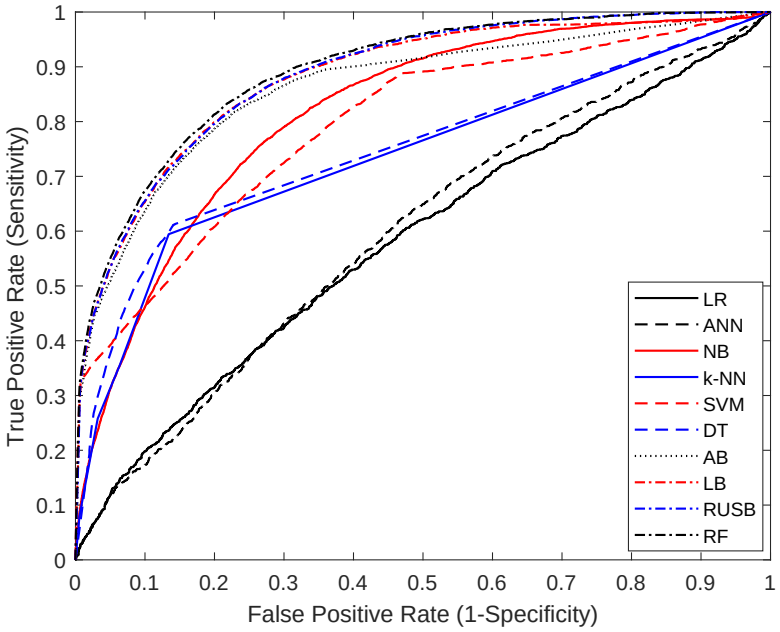

Figure 5.15: Impact of the health feature on the prediction performance.

The model used for this purpose is called the ARMA model. In relation to our dataset, the ARMA model can be represented in mathematical term as:

$$\widehat{HInd}_{t+1} = c + \sum_{k=0}^{p} \alpha_k Nut_{t-k} + \sum_{k=0}^{p} \beta_k Sle_{t-k} + \sum_{k=0}^{p} \gamma_k Irr_{t-k} + \sum_{k=0}^{p} \delta_k Hyd_{t-k} + \sum_{k=0}^{p} \theta_k Str_{t-k}$$
$$+ \sum_{k=0}^{p} \lambda_k Rest_{t-k} + \sum_{k=0}^{p} \mu_k Ene_{t-k} + \sum_{k=0}^{p} \xi_k Sor_{t-k} + \sum_{k=0}^{p} \rho_k Hea_{t-k} + \sum_{k=0}^{p} \sigma_k Enj_{t-k} + \sum_{k=0}^{p} \phi_k Exe_{t-k} \tag{5.2}$$

where α, β, γ, δ, θ, λ, μ, ξ, ρ, σ, and ϕ are the coefficients of the corresponding features, c is a constant, and p is the lag variable which is 6 in our case. The names of the features in (5.2) are written in short form. Based on (5.2), the estimated coefficients and p-values for all the 77 features are presented in Tables 5.5 and 5.6.

Considering the 5% significance level and also analyzing the p-values in Tables 5.5 and 5.6, the coefficients of the features Nut_t, Hyd_t, Hea_t, Exe_t, Hea_{t-1}, Irr_{t-6}, and Hea_{t-6} are considered to be statistically significant. The feature Sle_{t-6} is also very close to the significance level. It is important to note that the health feature Hea appeared to be the most significant one among all the features. However, most of the features are statistically insignificant. So, we extracted the eight most significant features from the dataset to test our ML models. Based on these features, we present results in Table 5.7 and in Figure 5.16 using 10-fold cross-validation.

The aim was to improve the classification performance using the ARMA model. However, comparing the results between Table 5.2 and Table 5.7, we did not see any improvement. In fact, the model performance is degraded a bit. One of the possible reasons behind this result is that the psychometric data that we have used in this thesis could be problematic.

Table 5.5: Descriptive statistics and p-values for all features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Intercept	-2.439	0.014772	-165.11	0
Nut_t	-0.031807	0.015611	-2.0374	0.04161*
Sle_t	-0.008826	0.016601	-0.53169	0.59494
Irr_t	-0.019086	0.016678	-1.1444	0.25246
Hyd_t	-0.041443	0.015985	-2.5927	0.00952*
Str_t	0.002276	0.017983	0.12655	0.8993
Rest_t	-0.035019	0.022988	-1.5233	0.12767
Ene_t	-0.02192	0.017861	-1.2273	0.21972
Sor_t	-0.010744	0.016697	-0.64347	0.51992
Hea_t	-0.387991	0.015479	-25.065	1.2e-138*
Enj_t	-0.011735	0.015266	-0.7687	0.44207
Exe_t	-0.039158	0.015052	-2.6015	0.00928*
Nut_{t-1}	-0.011658	0.016915	-0.68921	0.49069
Sle_{t-1}	-0.009215	0.017825	-0.51699	0.60516
Irr_{t-1}	0.009462	0.018044	0.52437	0.60002
Hyd_{t-1}	-0.016025	0.017482	-0.91668	0.35931
Str_{t-1}	0.014485	0.019931	0.72674	0.46738
Rest_{t-1}	-0.011751	0.025464	-0.46144	0.64448
Ene_{t-1}	0.007648	0.019286	0.39655	0.6917
Sor_{t-1}	0.004189	0.018599	0.22527	0.82177
Hea_{t-1}	-0.070046	0.017874	-3.9189	8.90e-05*
Enj_{t-1}	0.008475	0.016222	0.52246	0.60135
Exe_{t-1}	0.015862	0.016309	0.97262	0.33074
Nut_{t-2}	-0.018228	0.017043	-1.0696	0.28482
Sle_{t-2}	0.001072	0.017887	0.059939	0.9522
Irr_{t-2}	0.012838	0.018166	0.70667	0.47977
Hyd_{t-2}	0.007539	0.017698	0.42595	0.67014
Str_{t-2}	0.003471	0.019948	0.17398	0.86188
Rest_{t-2}	-0.012369	0.025622	-0.48274	0.62928
Ene_{t-2}	0.003078	0.019301	0.1595	0.87328
Sor_{t-2}	0.000208	0.018619	0.01119	0.99107
Hea_{t-2}	-0.023407	0.018536	-1.2628	0.20666
Enj_{t-2}	0.009049	0.016263	0.55637	0.57796
Exe_{t-2}	0.006873	0.016349	0.42042	0.67418
Nut_{t-3}	-0.008569	0.017189	-0.49854	0.6181
Sle_{t-3}	0.006305	0.017999	0.3503	0.72612
Irr_{t-3}	-0.007244	0.018115	-0.39988	0.68925
Hyd_{t-3}	0.005401	0.017793	0.30357	0.76145
Str_{t-3}	0.005997	0.019968	0.30033	0.76392
* represents significant p-value with $\alpha \leq 0.05$				

Table 5.6: Descriptive statistics and p-values for all features.

Feature	Estimate	Standard error	t-Statistic	p-Value
Rest_{t-3}	0.000525	0.025658	0.02048	0.98366
Ene_{t-3}	-0.004854	0.019328	-0.25116	0.80169
Sor_{t-3}	0.001182	0.018642	0.063437	0.94942
Hea_{t-3}	-0.014012	0.018943	-0.73969	0.45949
Enj_{t-3}	0.002814	0.016246	0.17321	0.86248
Exe_{t-3}	0.001718	0.016258	0.10571	0.91581
Nut_{t-4}	-0.007485	0.017171	-0.43592	0.66289
Sle_{t-4}	-0.009782	0.017968	-0.54442	0.58616
Irr_{t-4}	0.005376	0.018185	0.29564	0.7675
Hyd_{t-4}	0.001112	0.017733	0.062729	0.94998
Str_{t-4}	-0.011507	0.019872	-0.57905	0.56256
Rest_{t-4}	0.000565	0.025512	0.02216	0.98232
Ene_{t-4}	-0.011970	0.01929	-0.62054	0.5349
Sor_{t-4}	0.007712	0.018606	0.41452	0.67849
Hea_{t-4}	0.021853	0.019287	1.1331	0.25718
Enj_{t-4}	0.005254	0.016279	0.32275	0.74689
Exe_{t-4}	-0.013363	0.016124	-0.82876	0.40724
Nut_{t-5}	0.003351	0.017249	0.19424	0.84598
Sle_{t-5}	-0.004707	0.017965	-0.26201	0.79331
Irr_{t-5}	0.000547	0.018241	0.030007	0.97606
Hyd_{t-5}	0.003331	0.017766	0.18749	0.85127
Str_{t-5}	-0.011416	0.019896	-0.57375	0.56614
Rest_{t-5}	-0.000795	0.025516	-0.031133	0.97516
Ene_{t-5}	-0.002949	0.019289	-0.15291	0.87847
Sor_{t-5}	0.000938	0.018579	0.050522	0.95971
Hea_{t-5}	0.015121	0.019428	0.77828	0.4364
Enj_{t-5}	-0.003062	0.016248	-0.18844	0.85053
Exe_{t-5}	-0.006094	0.016154	-0.37727	0.70597
Nut_{t-6}	0.008377	0.016323	0.51325	0.60778
Sle_{t-6}	-0.030252	0.016969	-1.7828	0.074621^*
Irr_{t-6}	0.035302	0.017349	2.0348	0.041873^*
Hyd_{t-6}	0.001361	0.016606	0.08197	0.93467
Str_{t-6}	-0.000266	0.018191	-0.014638	0.98832
Rest_{t-6}	-0.012681	0.023269	-0.54497	0.58577
Ene_{t-6}	0.009649	0.018089	0.53347	0.59371
Sor_{t-6}	0.004248	0.016782	0.25316	0.80015
Hea_{t-6}	0.045555	0.017825	2.5556	0.0106^*
Enj_{t-6}	0.010244	0.015507	0.66063	0.50885
Exe_{t-6}	-0.020645	0.015235	-1.3551	0.17539
* represents significant p-value with $\alpha \leq 0.05$				

Table 5.7: Evaluation results using the ARMA model (seven omit days).

Model	AUC	Accuracy	Precision	Recall	Specificity	F-score
LR	0.629	0.872	0.254	0.226	0.935	0.239
ANN	0.642	0.866	0.251	0.247	0.927	0.249
NB	0.821	0.849	0.302	0.542	0.878	0.388
k-NN	0.688	0.905	0.421	0.195	0.974	0.267
SVM	0.703	0.916	0.738	0.073	0.997	0.133
DT	0.714	0.893	0.384	0.342	0.947	0.362
AB	0.805	0.913	0.518	0.288	0.974	0.370
LB	0.845	0.916	0.544	0.319	0.974	0.402
RUSB	0.859	0.821	0.293	0.722	0.831	0.417
RF	0.858	0.919	0.631	0.216	0.988	0.322

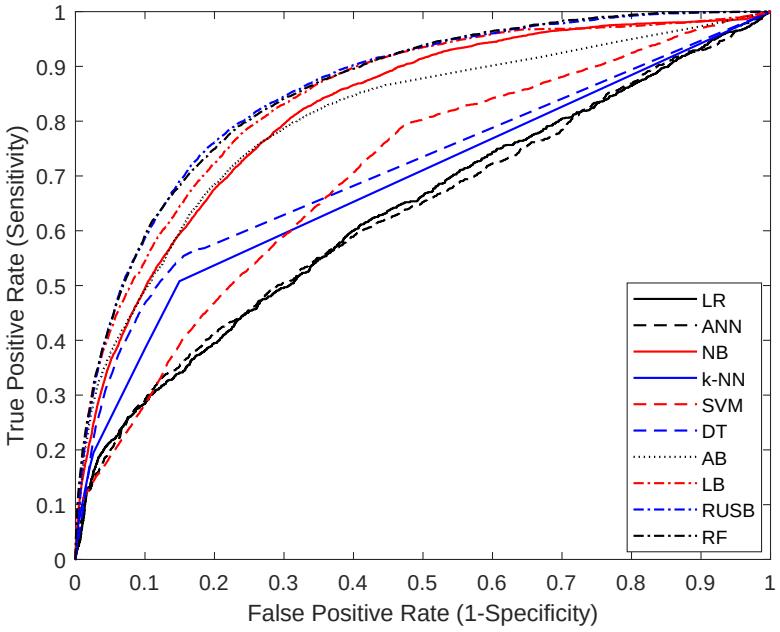

Figure 5.16: ROC curves using the ARMA model (seven omit days).

Table 5.8: Regression equations for the LR models for different features.

No. of features	Regression equation
11	$\widehat{HInd} = -2.434 - 0.042x_1 - 0.013x_2 - 0.009x_3 - 0.046x_4 + 0.011x_5$ $-0.064x_6 - 0.017x_7 - 0.005x_8 - 0.425x_9 - 0.002x_{10} - 0.040x_{11}$
10	$\widehat{HInd} = -2.438 - 0.029x_1 - 0.019x_2 - 0.004x_3 - 0.045x_4 + 0.005x_5$ $-0.061x_6 - 0.025x_7 - 0.002x_8 - 0.417x_9 - 0.043x_{11}$
9	$\widehat{HInd} = -2.420 - 0.056x_1 - 0.023x_2 - 0.015x_3 - 0.037x_4 + 0.026x_5$ $-0.048x_6 - 0.021x_7 - 0.413x_9 - 0.047x_{11}$
8	$\widehat{HInd} = -2.440 - 0.043x_1 - 0.014x_2 - 0.044x_4 + 0.012x_5 - 0.083x_6$ $-0.009x_7 - 0.448x_9 - 0.030x_{11}$
7	$\widehat{HInd} = -2.433 - 0.036x_1 - 0.020x_2 - 0.040x_4 - 0.060x_6 - 0.021x_7$ $-0.427x_9 - 0.032x_{11}$
6	$\widehat{HInd} = -2.430 - 0.050x_1 - 0.046x_4 - 0.070x_6 - 0.017x_7 - 0.425x_9$ $-0.040x_{11}$
5	$\widehat{HInd} = -2.443 - 0.050x_1 - 0.039x_4 - 0.064x_6 - 0.434x_9 - 0.045x_{11}$
Excluded the health feature (x_9)	$\widehat{HInd} = -2.370 - 0.118x_1 - 0.070x_2 - 0.043x_3 - 0.067x_4 - 0.010x_5$ $-0.087x_6 - 0.085x_7 - 0.017x_8 - 0.022x_{10} - 0.038x_{11}$
ARMA model	$\widehat{HInd} = -2.439 - 0.032x_1 - 0.041x_4 - 0.388x_9 - 0.039x_{11} - 0.070046x_{20}$ $-0.030252x_{68} + 0.035x_{69} + 0.045555x_{75}$

Table 5.8 presents the estimated regression equations which can be used to reproduce the results presented in this chapter for the LR models. This Table shows the number of features and the corresponding regression equation for seven omit days health label.

Table 5.9: Training time (in sec.).

NoF	LR	ANN	NB	k-NN	SVM	DT	AB	LB	RUSB	RF
11	0.25	3.13	0.12	0.014	157.93	1.21	23.56	18.11	6.85	52.99
9	0.21	1.91	0.10	0.013	139.44	0.84	19.59	15.15	6.32	53.44
7	0.18	3.89	0.08	0.013	106.25	1.48	15.80	14.50	6.16	56.77
5	0.19	2.73	0.07	0.012	90.21	1.38	12.50	9.31	4.99	45.02

5.8 Training Time

In our experiments, all the algorithms were implemented with MATLAB. Table 5.9 summarizes the training time of each model for different number of features. For a fair comparison, we used the same training set to compute the runtime for all the models. As can be seen, the traditional ML models are much faster than the ensemble models with k-NN takes the shortest training time followed by NB, LR, DT, ANN, RUSB, LB, AB, and RF, whereas the SVM model takes the longest training time. However, the performance of the traditional models is not as good as the ensemble models.

Among the four ensemble models, the much-reduced training time of RUSB was expected as it performs downsampling the majority class which reduces the length of the training data. Therefore, it can be said that in terms of both accuracy and speed, RUSB outperforms all the models used in this thesis. Note that the execution time decreases with the reduction of the number of insignificant features which ensures faster performance. Among the traditional methods, relatively better performance of SVM comes with the cost of high training time. Thus, the selection of a model depends on the application and our requirements. Note here that the training time of the ensemble methods can be reduced by reducing the number of learning cycles for AB, LB, and RUSB models and the number of trees for the RF model.

5.9 Summary and Discussion

In this study, we used a dataset from a company to predict AH using ML methods. The growing interest in this topic is because AH could have a negative impact on team performance. Hence, an accurate prediction of AH could have beneficial effects on team performance.

There are many ways to construct classification models, each with its own merits and demerits. This study used ten ML methods to predict AH using a new dataset. The classifier models were trained through supervised learning. The classifiers were also evaluated using fine-tuning and 10-fold cross-validation technique. The results presented in this chapter demonstrated that it is possible to predict AH using ML methods.

ML methods such as LR and ANN appeared to be faster, but they showed poor performance in classifying imbalanced data. Probably because of its overfitting problem, the DT classifier also did not do a good job. Although SVM showed slightly better performance than LR, ANN, and DT, however, it is computationally intensive. It is very common to use the Euclidian distance function to compute the distance in k-NN algorithm. This thesis experimented with different distance functions and selected the Manhattan function as the best for our dataset. Using the Manhattan distance as a distance measure, k-NN achieved better performance. Among the six ML methods, NB proved to be the best classifier in terms of classification performance and speed with the highest AUC of 0.83.

However, the primary goal of this chapter was to further improve classification performance of the imbalanced data. Ensemble-based methods that address the problem of class imbalance were used for this purpose. To improve the model performance, techniques such as bootstrap

sampling, undersampling, and boosting were combined with ML methods. With this, we chose four ensemble-based methods (AdaBoost, LogitBoost, RUSBoost, and random forest) to apply to our data. Using the technique of undersampling the majority class, the RUSB ensemble technique was exploited to alleviate the class imbalance in the training data. The results demonstrated that ensemble-based methods can indeed classify imbalanced data with higher performance compared to the traditional ML methods used in this study (Tables 5.2 and 5.3, Figures 5.7, 5.8, 5.11, and 5.12).

As a matter of fact, it was found that the ensemble-based RF classifier offered the best performance in terms of AUC, whereas the RUSB classifier gave the best performance in terms of recall. However, if we consider the predictive performance and the training time together, then the RUSB model could be a better option for our data.

We also performed experiments to demonstrate an efficient way of feature selection by discarding insignificant features. Although removing insignificant features did not have any significant impact on the results, however, removing those features made the algorithms faster which may be good for some applications.

The models are also tested after eliminating the health feature from the dataset. It has been shown that the overall performance of the models did not degrade significantly even after removing the health feature. The performance of ML models is further tested using the ARMA model to see if the classification performance can be improved. Finally, a comparative discussion of the training time of each model is given.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Sport organizations will find it useful to monitor the performance of their athletes by predicting their health beforehand. Therefore, it is important to develop suitable models using athletes' historical data. However, there are two primary problems with the dataset we used in this study. The first one is the missing values and the second one is the imbalanced classes, which make it difficult for ML methods to accurately predict two classes.

To fill in the missing values, we compared several interpolation techniques and eventually selected the nearest-neighbor interpolation technique for our data. Then, we tackled the problem of classifying imbalanced data. Initially, we started the research with LR as a baseline method and applied it to our data to predict athletes' health (AH). However, the LR model performed very poorly yielding an AUC of 0.625 and an F-score of 0.241. We then tried with more complex ML methods. The algorithms were fine-tuned to obtain the best possible parameters. We also

experimented with different kernel functions to improve the classification performance of the NB and the SVM models. The results show that the traditional ML methods are useful for the prediction of AH. Among all the methods, the NB classifier performed the best, resulting in an AUC of around 0.83 and an F-score of around 0.4. Even though we managed to improve the prediction performance, the results still may not be satisfactory in applications where a higher performance is required.

Hence, to improve the classification performance further, we used ensemble-based methods such as AdaBoost, LogitBoost, RUSBoost, and random forest. To improve the model performance, techniques such as sampling and parameter tuning are utilized. These approaches helped to alleviate the class imbalance problem in the training data. Our results show that the ensemble-based methods outperformed the traditional methods in terms of all the metrics. The results also demonstrate that among the four ensemble methods, the RUSBoost model has the best overall performance. While the LR model predicted 1348 positive cases correctly out of 5893 positive cases, resulting in a recall/sensitivity of a mere 23.2%, the RUSBoost model predicted 4370 positive cases correctly with a recall/sensitivity of 73.6%, which is 50.4% more than the LR model. In terms of the training time, the RUSBoost algorithm also appeared to be faster than the other ensemble methods.

We also used the correlation matrix and p-values to identify and eliminate insignificant features. It is demonstrated that by using p-values, the backward elimination technique can be applied to remove insignificant features which reduce the size of the original dataset. Removing insignificant features did not improve the classification performance in our case. It did, however, make the algorithms faster with little penalty.

6.2 Future Work

This study will influence future work in predicting athletes' health. Based on the conclusions of this study, possible future work could be expanded as follows.

The dataset used in this thesis does not contain enough data. The number of features is also limited. In the future, more data could be collected by incorporating more features to test the models. The models could also be tested in other applications such as spam detection and fraud detection where datasets usually contain imbalanced classes.

In this thesis, the model parameters were tuned by using grid search method. In the future, the parameters could be tuned by using optimization algorithms.

This thesis used AUC as a measure to select the best interpolation technique. In the future, a study could be conducted to find a better way of selecting the best interpolation technique.

In the future, the binary classification problem of this thesis could be extended to ternary or multi-class classification problems.

Although the classification performance obtained in this work is reasonably acceptable, other methods could be sought to improve the model performance further.

List of Abbreviations

AH	Athletes' Health
ML	Machine Learning
AI	Artificial Intelligence
LR	Logistic Regression
NB	Naive Bayes
k-NN	k-Nearest Neighbor
ANN	Artificial Neural Network
MLP	Multilayer Perceptron
SVM	Support Vector Machine
DT	Decision Tree
DNN	Deep Neural Network
RF	Random Forest
AdaBoost (AB)	Adaptive Boosting
LogitBoost (LB)	Adaptive Logistic Regression
RUSBoost (RUSB)	Random Undersampling Boosting

Bagging	Bootstrap Aggregating
ROC	Receiver Operating Characteristic
AUC	Area Under the ROC Curve
SMOTE	Synthetic Minority Oversampling Technique
CART	Classification And Regression Trees
PCA	Principal Component Analysis
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
TPR	True Positive Rate
FPR	False Positive Rate
TNR	True Negative Rate
MAP	Maximum A Posteriori

Bibliography

[1] E. Winslow, N. Bohannon, S. A. Brunton, and H. E. Mayhew, "Lifestyle modification: Weight control, exercise, and smoking cessation," in *American Journal of Medicine*, vol. 101, no. 4 A, 1996, pp. 25S–33S.

[2] T. K. Houston, L. A. Meoni, D. E. Ford, F. L. Brancati, L. A. Cooper, D. M. Levine, K. Y. Liang, and M. J. Klag, "Sports ability in young men and the incidence of cardiovascular disease," *American Journal of Medicine*, vol. 112, no. 9, pp. 689–695, 2002.

[3] N. Smart and T. H. Marwick, "Exercise training for patients with heart failure: A systematic review of factors that improve mortality and morbidity," pp. 693–706, 2004.

[4] R. V. Milani and C. J. Lavie, "Reducing Psychosocial Stress: A Novel Mechanism of Improving Survival from Exercise Training," *American Journal of Medicine*, vol. 122, no. 10, pp. 931–938, 2009. [Online]. Available: http://dx.doi.org/10.1016/j.amjmed.2009.03.028

[5] W. E. Boden, B. Franklin, K. Berra, W. L. Haskell, K. J. Calfas, F. H. Zimmerman, and N. K. Wenger, "Exercise as a therapeutic intervention in patients with stable ischemic

heart disease: An underfilled prescription," pp. 905–911, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.amjmed.2014.05.007

[6] H. B. Simon, "Exercise and Health: Dose and Response, Considering Both Ends of the Curve," pp. 1171–1177, 2015. [Online]. Available: http://dx.doi.org/10.1016/j.amjmed.2015.05.012

[7] T. Capel, J. F. Schnittert, S. Snow, and D. Vyas, "Exploring Motivations of Young Adults to Participate in Physical Activities," in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems - CHI EA '15*, 2015, pp. 1409–1414. [Online]. Available: http://dl.acm.org/citation.cfm?doid=2702613.2732800

[8] J. Agel, T. A. Evans, R. Dick, M. Putukian, and S. W. Marshall, "Descriptive epidemiology of collegiate men's soccer injuries: National Collegiate Athletic Association Injury Surveillance System, 1988-1989 Through 2002-2003," *Journal of Athletic Training*, vol. 42, no. 2, pp. 270–277, 2007.

[9] E. E. Yard, C. L. Collins, and R. D. Comstock, "A comparison of high school sports injury surveillance data reporting by certified athletic trainers and coaches," *Journal of athletic training*, vol. 44, no. 6, pp. 645–652, 2009. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/19911092{%}5Cnhttp://www.ncbi.nlm.nih.gov/pmc/articles/PMC2775367/pdf/i1062-6050-44-6-645.pdf

[10] A. Ivarsson, U. Johnson, M. Lindwall, H. Gustafsson, and M. Altemyr, "Psychosocial stress as a predictor of injury in elite junior soccer: A latent growth curve analysis,"

Journal of Science and Medicine in Sport, vol. 17, no. 4, pp. 366–370, 2014. [Online]. Available: http://dx.doi.org/10.1016/j.jsams.2013.10.242

[11] R. Bahr and T. Krosshaug, "Understanding injury mechanisms: A key component of preventing injuries in sport," *British Journal of Sports Medicine*, vol. 39, no. 6, pp. 324–329, 2005.

[12] S. L. Bruce, E. Crawford, G. B. Wilkerson, D. Rausch, R. B. Dale, and M. Harris, "Prediction Modeling for Academic Success in Professional Master's Athletic Training Programs," *Athletic Training Education Journal*, vol. 11, no. 4, pp. 194–207, 2016.

[13] J. Sibold and S. Zizzi, "Psychosocial variables and time to injury onset: A hurdle regression analysis model," *Journal of Athletic Training*, vol. 47, no. 5, pp. 537–540, 2012.

[14] G. B. Wilkerson, J. L. Giles, and D. K. Seibel, "Prediction of core and lower extremity strains and sprains in collegiate football players: A preliminary study," *Journal of Athletic Training*, vol. 47, no. 3, pp. 264–272, 2012.

[15] G. B. Wilkerson, J. T. Bullard, and D. W. Bartal, "Identification of cardiometabolic risk among collegiate football players," *Journal of Athletic Training*, vol. 45, no. 1, pp. 67–74, 2010.

[16] J. M. Sefton, K. R. Lohse, and J. S. McAdam, "Prediction of injuries and injury types in army basic training, infantry, armor, and cavalry trainees using a common fitness screen," *Journal of Athletic Training*, vol. 51, no. 11, pp. 849–857, 2016.

[17] A. M. Smith, M. J. Stuart, D. M. Wiese-Bjornstal, and C. Gunnon, "Predictors of injury in Ice Hockey players: A multivariate, multidisciplinary approach," *American Journal of Sports Medicine*, vol. 25, no. 4, pp. 500–507, 1997.

[18] T. Op De Beéck, W. Meert, K. Schütte, B. Vanwanseele, and J. Davis, "Fatigue Prediction in Outdoor Runners Via Machine Learning and Sensor Fusion," *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining - KDD '18*, pp. 606–615, 2018. [Online]. Available: http://dl.acm.org/citation.cfm?doid=3219819.3219864

[19] J. Hucaljuk and A. Rakipovic, "Predicting football scores using machine learning techniques," *2011 Proceedings of the 34th International Convention MIPRO*, vol. 48, pp. 1623–1627, 2011.

[20] H. Kaur and S. Jain, "Machine learning approaches to predict basketball game outcome," *Proceedings - 2017 3rd International Conference on Advances in Computing, Communication and Automation (Fall), ICACCA 2017*, vol. 2018-Janua, pp. 1–7, 2018.

[21] R. P. Bunker and F. Thabtah, "A machine learning framework for sport result prediction," *Applied Computing and Informatics*, vol. 15, no. 1, pp. 27–33, 2019. [Online]. Available: http://linkinghub.elsevier.com/retrieve/pii/S2210832717301485

[22] V. Rao and A. Shrivastava, "Team strategizing using a machine learning approach," *2017 International Conference on Inventive Computing and Informatics (ICICI)*, no. Icici, pp. 1032–1035, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/8365296/

[23] T. J. W. Dawes, A. de Marvao, W. Shi, T. Fletcher, G. M. J. Watson, J. Wharton, C. J. Rhodes, L. S. G. E. Howard, J. S. R. Gibbs, D. Rueckert, S. A. Cook, M. R. Wilkins, and D. P. O'Regan, "Machine Learning of Three-dimensional Right Ventricular Motion Enables Outcome Prediction in Pulmonary Hypertension: A Cardiac MR Imaging Study," *Radiology*, vol. 283, no. 2, pp. 381–390, 2017. [Online]. Available: http://pubs.rsna.org/doi/10.1148/radiol.2016161315

[24] C. Y. Hung, W. C. Chen, P. T. Lai, C. H. Lin, and C. C. Lee, "Comparing deep neural network and other machine learning algorithms for stroke prediction in a large-scale population-based electronic medical claims database," *Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS*, pp. 3110–3113, 2017.

[25] B. Nithya and V. Ilango, "Predictive analytics in health care using machine learning tools and techniques," *Proceedings of the 2017 International Conference on Intelligent Computing and Control Systems, ICICCS 2017*, vol. 2018-Janua, pp. 492–499, 2018.

[26] S. F. Weng, J. Reps, J. Kai, J. M. Garibaldi, and N. Qureshi, "Can Machine-learning improve cardiovascular risk prediction using routine clinical data?" *PLoS ONE*, vol. 12, no. 4, pp. 1–14, 2017.

[27] K. N. R. Challa, V. S. Pagolu, G. Panda, and B. Majhi, "An Improved Approach for Prediction of Parkinson's Disease using Machine Learning Techniques," 2016, pp. 1446–1451. [Online]. Available: http://arxiv.org/abs/1610.08250

[28] F. Wang, P. Zhang, B. Qian, X. Wang, and I. Davidson, "Clinical risk prediction with multilinear sparse logistic regression," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014, pp. 145–154.

[29] C. M. Sehgal, T. W. Cary, A. Cwanger, B. J. Levenback, and S. S. Venkatesh, "Combined Naïve Bayes and logistic regression for quantitative breast sonography," *IEEE International Ultrasonics Symposium, IUS*, pp. 1686–1689, 2012.

[30] N. J. Nilsson, *Introduction to Machine Learning*, 2005.

[31] Aurélien Géron, *Hands-on Machine Learning with Scikit-Learn& TensorFlow*, 2017.

[32] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Supervised Machine Learning : A Review of Classification Techniques General Issues of Supervised Learning Algorithms," pp. 501–520, 2007.

[33] A. F. Zuur, E. N. Ieno, and C. S. Elphick, "A protocol for data exploration to avoid common statistical problems," *Methods in Ecology and Evolution*, vol. 1, no. 1, pp. 3–14, 2010.

[34] J. E. Myers, T. J. Sweeney, and J. M. Witmer, "The Wheel of Wellness Counseling for Wellness :," *Journal of Counseling & Development*, vol. 78, pp. 251–266, 2000.

[35] H. P. Dijkstra, N. Pollock, R. Chakraverty, and J. M. Alonso, "Managing the health of the elite athlete: A new integrated performance health management and coaching model," *British Journal of Sports Medicine*, vol. 48, no. 7, pp. 523–531, 2014.

[36] D. Caine, T. Walch, and T. Sabato, "The elite young athlete: strategies to ensure physical and emotional health," *Open Access Journal of Sports Medicine*, vol. Volume 7, pp. 99–113, 2016.

[37] E. Acuña and C. Rodriguez, "The Treatment of Missing Values and its Effect on Classifier Accuracy," *Classification, Clustering, and Data Mining Applications*, pp. 639–647, 2004.

[38] V. J. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artificial Intelligence Review*, vol. 22, no. 2, pp. 85–126, 2004.

[39] W. J. Long and W. X. Zhang, "A novel measure of compatibility and methods of missing attribute values treatment in decision tables," in *Proceedings of 2004 International Conference on Machine Learning and Cybernetics*, vol. 4, 2004, pp. 2356–2360.

[40] K. Penny and T. Chesney, "Imputation methods to deal with missing values when data mining trauma injury data," *28th International Conference on Information Technology Interfaces, 2006.*, pp. 213–218, 2006.

[41] R. Jagannathan and S. Petrovic, "Dealing with missing values in a clinical case-based reasoning system," *Proceedings - 2009 2nd IEEE International Conference on Computer Science and Information Technology, ICCSIT 2009*, pp. 120–124, 2009. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5234442

[42] Y. Zou, A. An, and X. Huang, "Evaluation and automatic selection of methods for handling missing data," *Granular Computing, 2005 IEEE International Conference on*, vol. 2, pp. 728–733 Vol. 2, 2005.

[43] T. M. Lehmann, C. Gönner, and K. Spitzer, "Survey: Interpolation methods in medical image processing," *IEEE Transactions on Medical Imaging*, vol. 18, no. 11, pp. 1049–1075, 1999.

[44] A. T. Andrade, C. Montez, R. Moraes, A. R. Pinto, F. Vasques, and G. L. Da Silva, "Outlier detection using k-means clustering and lightweight methods for Wireless Sensor Networks," *IECON Proceedings (Industrial Electronics Conference)*, pp. 4683–4688, 2016.

[45] C. Hsiao, "Panel data analysis-advantages and challenges," *Test*, vol. 16, no. 1, pp. 1–22, 2007.

[46] T. Zheng, D. Hu, X. Wang, and B. Yu, "Panel data clustering and its application to discount rate of B stock in China," *2009 2nd International Conference on Information and Computing Science, ICIC 2009*, vol. 1, pp. 163–166, 2009.

[47] B. Krawczyk, "Cost-sensitive one-vs-one ensemble for multi-class imbalanced data," *Proceedings of the International Joint Conference on Neural Networks*, vol. 2016-Octob, pp. 2447–2452, 2016.

[48] N. V. Chawla, "Data Mining for Imbalanced Datasets: An Overview," *Data Mining and Knowledge Discovery Handbook*, pp. 875–886, 2009.

[49] Y. Sun, M. S. Kamel, and Y. Wang, "Boosting for learning multiple classes with imbalances class distribution," in *Proceedings - IEEE International Conference on Data Mining, ICDM*, 2006, pp. 592–602.

[50] Y. Pristyanto, I. Pratama, and A. F. Nugraha, "Data level approach for imbalanced class handling on educational data mining multiclass classification," *2018 International Conference on Information and Communications Technology, ICOIACT 2018*, vol. 2018-Janua, pp. 310–314, 2018.

[51] F. Shakeel, A. S. Sabhitha, and S. Sharma, "Exploratory review on class imbalance problem: An overview," *8th International Conference on Computing, Communications and Networking Technologies, ICCCNT 2017*, 2017.

[52] N. Thai-Nghe, Z. Gantner, and L. Schmidt-Thieme, "Cost-sensitive learning methods for imbalanced data," in *Proceedings of the International Joint Conference on Neural Networks*, 2010.

[53] E. Lloyd, "The foundations of learning," in *Children & Society*, vol. 18, no. 3, 2004, pp. 248–249.

[54] Y. Sun, M. S. Kamel, A. K. Wong, and Y. Wang, "Cost-sensitive boosting for classification of imbalanced data," Ph.D. dissertation, 2007.

[55] Y. Zhang and D. Wang, "A cost-sensitive ensemble method for class-imbalanced datasets," *Abstract and Applied Analysis*, vol. 2013, 2013.

[56] Z. Sun, Q. Song, X. Zhu, H. Sun, B. Xu, and Y. Zhou, "A novel ensemble method for classifying imbalanced data," *Pattern Recognition*, vol. 48, no. 5, pp. 1623–1637, 2015.

[57] W. M. Van Der Aalst, V. Rubin, H. M. Verbeek, B. F. Van Dongen, E. Kindler, and C. W. Günther, "Process mining: A two-step approach to balance between underfitting and overfitting," *Software and Systems Modeling*, vol. 9, no. 1, pp. 87–111, 2010.

[58] L. Prechelt, "Automatic early stopping using cross validation: Quantifying the criteria," *Neural Networks*, vol. 11, no. 4, pp. 761–767, 1998.

[59] C. Alippi and M. Roveri, "Virtual k-fold cross validation: An effective method for accuracy assessment," in *Proceedings of the International Joint Conference on Neural Networks*, 2010.

[60] S. Yadav and S. Shukla, "Analysis of k-Fold Cross-Validation over Hold-Out Validation on Colossal Datasets for Quality Classification," in *Proceedings - 6th International Advanced Computing Conference, IACC 2016*, 2016, pp. 78–83.

[61] R. Kohavi, "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection," *International Joint Conference of Artificial Intelligence*, 1995.

[62] R. Reed and S. Member, "Pruning Algorithms-A," vol. 4, no. 5, pp. 740–747, 1993.

[63] J. Chen, X. Wang, and J. Zhai, "Pruning Decision Tree Using Genetic Algorithms," *Proceedings of the International Conference on Artificial Intelligence and Computational Intelligence, AICI '09*, vol. 3, pp. 244–248, 2009.

[64] A. Lo, H. Chernoff, T. Zheng, and S. H. Lo, "Why significant variables aren't automatically good predictors," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 112, no. 45, pp. 13 892–13 897, 2015.

[65] J. Lever, M. Krzywinski, and N. Altman, "Points of Significance: Classification evaluation," *Nature Methods*, vol. 13, no. 8, pp. 603–604, 2016.

[66] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *ACM International Conference Proceeding Series*, vol. 148, 2006, pp. 233–240.

[67] E. Chandra Blessie, E. Karthikeyan, and B. Selvaraj, "Empirical study on the performance of the classifiers based on various criteria using ROC curve in medical health care," *Proceedings of 2010 International Conference on Communication and Computational Intelligence, INCOCCI-2010*, pp. 515–518, 2010.

[68] M. Weerasinghe, G. K. Dias, A. Dharmaratne, D. Sandaruwan, A. Nisansala, C. Keppitiyagama, and N. Kodikara, "Computer aid assessment of muscular imbalance for preventing overuse injuries in athletes," in *ACM International Conference Proceeding Series*, 2016, pp. 244–248.

[69] S. Ben-David and S. Shalev-Shwartz, *Understanding Machine Learning: From Theory to Algorithms*, 2014.

[70] C. Kanzow, N. Yamashita, and M. Fukushima, "Levenberg-Marquardt methods with strong local convergence properties for solving nonlinear equations with convex constraints," *Journal of Computational and Applied Mathematics*, vol. 173, no. 2, pp. 321–343, 2005.

[71] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "Training algorithm for optimal margin classifiers," in *Proceedings of the Fifth Annual ACM Workshop on Computational Learning*

Theory, 1992, pp. 144–152.

[72] A. Jahangiri and H. A. Rakha, "Applying Machine Learning Techniques to Transportation Mode Recognition Using Mobile Phone Sensor Data," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 5, pp. 2406–2417, 2015.

[73] C. Seiffert, T. M. Khoshgoftaar, J. Van Hulse, and A. Napolitano, "RUSBoost: Improving classification performance when training data is skewed," in *Proceedings - International Conference on Pattern Recognition*, 2008.

[74] ——, "RUSBoost: A hybrid approach to alleviating class imbalance," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, vol. 40, no. 1, pp. 185–197, 2010.

[75] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 904, pp. 23–37, 1995.

[76] L. Wei, Y. Yang, R. M. Nishikawa, and Y. Jiang, "A study on several machine-learning methods for classification of malignant and benign clustered microcalcifications," *IEEE Transactions on Medical Imaging*, vol. 24, no. 2, pp. 371–380, 2005.

[77] J. Friedman, T. Hastie, and R. Tibshirani, "Additive logistic regression: A statistical view of boosting," *Annals of Statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[78] L. Breiman, "Random Forrests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001. [Online]. Available: https://www.stat.berkeley.edu/{~}breiman/randomforest2001.pdf

[79] A. S. More and D. P. Rana, "Review of random forest classification techniques to resolve data imbalance," *Proceedings - 1st International Conference on Intelligent Systems and Information Management, ICISIM 2017*, vol. 2017-Janua, pp. 72–78, 2017.