**Application of Machine Learning for identifying the presence of Cardiovascular disease.**

**Student name: Ali Ahmed**

**Student number: 16010572**

**Supervisor: DR Kevin Golden**

**Module code: UFCF9Y-60-M**

**Programme: MSc Data Science**

**Abstract**

Cardiovascular disease (CVD) is the primary cause of death worldwide. Current diagnostic tools are hampered by several limitations, including invasiveness, interpretability and accessibility, this necessitates the requirement for alternative approaches. Machine learning (ML) has emerged as an alternative tool for screening CVD, by analysing patterns in existing medical data, ML algorithms can predict disease outcomes with high accuracy, offering a quicker, more accessible and non-invasive approach for detecting CVD. Numerous studies have shown that ML algorithms are more than capable of accurately predicting patients with CVD, However, many of these studies have neglected fundamental techniques such as feature selection and hyperparameter optimization. This project aims to investigate the use of supervised classification ML algorithms in predicting patients with CVD, while also incorporating feature selection and hyperparameter optimisation in a bid to enhance performance. In this project Five ML algorithms (KNN, RF, LR, adaboost, XGBoost) were trained, tuned and evaluated on the Cleveland heart disease dataset obtained from the UCI ML repository. The results reveal all algorithm performed well achieving accuracy scores > 80%, furthermore, the best performing algorithms was found to be the XGBoost classifier achieving highest scores across all germane evaluation metrics with a notable accuracy score of 85%. The findings from this project can be implemented into clinical settings, providing physicians an alternative tool to identify CVD.

**Keywords:** Cardiovascular disease (CVD), Machine learning (ML), K nearest neighbors (KNN), Random Forest (RF), Logistic regression (LR).

Table of Contents

[1: Introduction, Rationale and Objectives 6](#_Toc129882713)

[1.1 Introduction to cardiovascular diseases 6](#_Toc129882714)

[1.2 Current methods for diagnosing CVD. 6](#_Toc129882715)

[1.3 Machine learning 7](#_Toc129882716)

[1.3.1 How machine learning can be applied for detecting CVD. 7](#_Toc129882717)

[1.4 Significance 8](#_Toc129882718)

[1.5 Aims and objectives. 8](#_Toc129882719)

[1.5.1 Success criteria 8](#_Toc129882720)

[1.6 Scope of study 8](#_Toc129882721)

[1.7 Gaps in research 8](#_Toc129882722)

[1.8 Ethical considerations 8](#_Toc129882723)

[1.9 Code Availability 8](#_Toc129882724)

[2: Literature review 9](#_Toc129882725)

[2.1 Applications of machine learning in healthcare 9](#_Toc129882726)

[2.2 ML for disease diagnosis. 9](#_Toc129882727)

[2.3 Related works 10](#_Toc129882728)

[2.3.1 Random Forest 10](#_Toc129882729)

[2.3.2 Logistic regression 10](#_Toc129882730)

[2.3.4 K Nearest Neighbour 10](#_Toc129882731)

[2.3.5 Boosting algorithms 11](#_Toc129882732)

[2.3.6 XGBoost 11](#_Toc129882733)

[2.3.7 Systematic reviews 11](#_Toc129882734)

[2.3.8 Importance of hyperparameter optimisation 12](#_Toc129882735)

[2.3.9 Importance of feature selection 12](#_Toc129882736)

[2.3.10 Potential flaws with ML 13](#_Toc129882737)

[2.4 Literature Review Summary 13](#_Toc129882738)

[3: Methodology 16](#_Toc129882739)

[3.1 Dataset 17](#_Toc129882740)

[3.1.1 Importing, cleaning and formatting data. 17](#_Toc129882741)

[3.1.2 Dataset description 18](#_Toc129882742)

[3.2 Exploratory data analysis 19](#_Toc129882743)

[3.2.1 Univariate analysis 19](#_Toc129882744)

[3.2.2 Bivariate analysis 22](#_Toc129882745)

[3.2.3 Multivariate analysis 23](#_Toc129882746)

[3.3 Data pre-processing 25](#_Toc129882747)

[3.3.1 One hot Encoding 25](#_Toc129882748)

[3.3.2 Feature scaling 26](#_Toc129882749)

[3.3.3 Feature selection 26](#_Toc129882750)

[3.4 Machine learning algorithms used. 28](#_Toc129882751)

[3.4.1 K nearest Neighbors 28](#_Toc129882752)

[3.4.2 Logistic regression 29](#_Toc129882753)

[3.4.3 Random Forest 29](#_Toc129882754)

[3.4.4 Adaptive boost 29](#_Toc129882755)

[3.4.5 XGBoost 30](#_Toc129882756)

[3.5 Cross-validation 30](#_Toc129882757)

[3.6 Hyperparameter Optimisation. 31](#_Toc129882758)

[3.6.1 Hyperparameters tuned for each algorithm. 31](#_Toc129882759)

[3.7 Evaluation methodology 32](#_Toc129882760)

[3.7.1 Accuracy 32](#_Toc129882761)

[3.7.2 Confusion matrix 32](#_Toc129882762)

[3.7.3 Precision 32](#_Toc129882763)

[3.7.4 Recall 32](#_Toc129882764)

[3.7.5 Precision recall trade off 33](#_Toc129882765)

[3.7.6 F1 score 33](#_Toc129882766)

[3.7.7 ROC-AUC 33](#_Toc129882767)

[3.8 Model building workflow. 34](#_Toc129882768)

[4: Results and Analysis 36](#_Toc129882769)

[4.1 K-Nearest Neighbors results 37](#_Toc129882770)

[4.2 Logistic regression results 37](#_Toc129882771)

[4.3 Random Forest results 38](#_Toc129882772)

[4.4 Adaboost results 38](#_Toc129882773)

[4.5 XGBoost results 39](#_Toc129882774)

[5: Evaluation and Discussion 40](#_Toc129882775)

[5.1 Findings and Discussion 40](#_Toc129882776)

[5.1.1 Comparison with previous studies 40](#_Toc129882777)

[5.2 Limitations 41](#_Toc129882778)

[5.3 Future work 41](#_Toc129882779)

[5.4 Reflection 41](#_Toc129882780)

[5.5 Conclusion 42](#_Toc129882781)

[6: References 43](#_Toc129882782)

[7: Appendix 46](#_Toc129882783)

[7.1 Appendix A: Abbreviations 46](#_Toc129882784)

[7.2 Appendix B: Supervisory meetings 47](#_Toc129882785)

[7.3 Appendix C: Gantt Chart 48](#_Toc129882786)

[7.4 Appendix D: Supplementary Figures 49](#_Toc129882787)

[7.5 Appendix E: Code 50](#_Toc129882788)

**List of Figures:**

**Figure 1:** Workflow diagram of proposed methodology

**Figure 2:** Screenshot of Data cleaning

**Figure 3:** Screenshot of Data Transformations

**Figure 4:** Descriptive statistics of dataset

**Figure 5:** Histograms and box plots illustrating distributions of continuous features.

**Figure 6:** Bar chart illustrating distributions of categorical features.

**Figure 7:** Class distribution of Target variable

**Figure 8:** Bar chart of categorical features with target as hue

**Figure 9:** Grouped box plots of numeric features.

**Figure 10:** Corelation matrix

**Figure 11:** Multivariate Pair plot

**Figure 12:** Screenshot of feature importance rankings

**Figure 13:** Screenshot of model building workflow.

**Figure 14:** ROC curves

**Figure 15:** KNN Confusion matrix

**Figure 16:** LR Confusion matrix

**Figure 17:** RF Confusion matrix

**Figure 18:** adaboost Confusion matrix

**Figure 19:** XGBoost Confusion matrix

**List of Tables:**

**Table 1**: Comparative summary of studies mentioned in literature review.

**Table 2:** Dataset attributes with description.

**Table 3:** Summary Table of evaluation scores for each model.

# 1: Introduction, Rationale and Objectives

This chapter delivers an introduction to cardiovascular disease, including current diagnostic methods and their limitations. This chapter also introduces the concept of machine learning (ML) along with its applications in CVD diagnosis. Finally, this chapter concludes by specifying the aims, objectives and scope of this project, along with the ethical considerations involved.

## 1.1 Introduction to cardiovascular diseases

The term cardiovascular disease (CVD) serves to describe a collection of disorders that impact the heart and blood vessels. Approximately 32% of all casualties globally and around 17.9 million deaths annually are attributable to CVD (WHO, 2022).

Heart attacks and cerebral strokes account for 85% of all CVD death's (WHO, 2022). Coronary heart disease is the primary trigger of heart attacks and transpires when there is a reduction in oxygen rich blood supply to the heart muscle cells. The reduction in blood supply is brought on by the reoccurring accumulation of fatty deposits in the coronary arteries called atherosclerotic plaques, these plaques cause the narrowing of the arteries therefore reduce blood flow to the heart (NHS, 2022a). Angina, a malady marked by sensations of chest pain is a clear symptom of coronary heart disease (NHS, 2022a). During the latter stages of coronary heart disease, the blood vessels can become completely blocked by a blood clot, this is what causes the heart attack. In the midst of a heart attack, cardiac muscle cells are starved of oxygen and begin to die. The death of these cells causes the heart muscle to deteriorate and lose its ability to pump blood. If immediate treatment is not provided the heart stops pumping blood, resulting in heart failure and eventually death (NHS, 2022b).

Cerebral strokes are caused by disruptions in the oxygen rich blood supply to the brain. Ischemic strokes are the most common form of strokes, accounting for roughly 85% of all strokes (NHS, 2022c). Ischemic stroke arises when a blood clot obstructs the artery supplying the brain, without blood flow brain cells called neurons begin to die due to a lack of oxygen (NHS ,2022c). A stroke can be identified through symptoms such as a drooping face, weak arms and slurred speech (NHS, 2022c). The timely identification of a stroke is crucial as a delayed response can result in lifelong brain damage and potentially death.

## 1.2 Current methods for diagnosing CVD.

CVDs can be diagnosed using a variety of techniques, these techniques include analysing biomarkers like cholesterol through blood tests or observing electrocardiograms (ECG’s) in search for abnormal heart rhythms (NHS, 2022a). Nevertheless, the most effective and reliable method currently used to diagnose CVDS is an angiography (NHS, 2022d). An angiography is an invasive procedure that is only employed when other tests are unable to deliver a clear-cut diagnosis. During an angiography a radiocontrast agent is injected into the blood vessels. Once injected, doctors can view images of the blood vessel using X-rays, this allows any abnormalities such as atherosclerotic plaques or blood clots to be identified (NHS, 2022d).

Even though angiography tests are effective for identifying CVD’S, they come with several limitations that can impair the precision and accessibility of diagnosis. One drawback of an angiography is its invasiveness, since it involves inserting a catheter into a blood vessel, issues like bleeding, bruising and infection can occur. Additionally, angiography’s also carry the risk of an allergic reaction to the radiocontrast agent, in most cases this manifests as a mild rash, However, in rare circumstances it can lead to kidney damage and heart attacks (NHS, 2022d).

Angiography tests are also affected by the issue of interobserver variability, which occurs when different observers interpret the same images differently. This has been documented in a study by Zir et al., (1976), in which four angiographers independently examined 20 angiograms and discovered a 42% mean disagreement between observers. The likelihood of interobserver variability rises, especially when dealing with inexperienced medical personnel. Interobserver variability can result in larger percentages of false positives or false negatives, culminating in unnecessary or delayed therapy and ultimately serious harm to the patient.

In developing countries, especially in southeast Asia, there is limited access to angiographies (Qanitha et al., 2019). This is caused by many reasons such as lack of specialised training and high cost of medical equipment. Alternative methods that can swiftly and accurately identify CVDs are therefore needed.

Overall, the shortcomings of existing methods for diagnosing CVDs emphasise the need for alternative approaches that are more accurate, convenient, and accessible. ML offers promising possibilities for improving CVD diagnosis and eventually improving outcome of patients.

## 1.3 Machine learning

The concept of ML was initially defined by Samuel (1959) as “the field of study that gives computers the ability to learn without being explicitly programmed”. ML can be broadly classified into three main categories: Supervised, Unsupervised and reinforced learning.

Supervised learning involves training an algorithm using labelled data, labelled data refers to data where the output (target) is already known (Raschka and Mirjalili, 2019). Supervised learning is further divided into two subcategories called classification and regression, both of which are utilised for distinct types of problems. Classification is used when a problem requires the prediction of discrete categories, examples of classification problems include disease diagnostics, spam detection and image classification. Regression is used when the task at hand is to predict a continuous numeric outcome, regression is frequently used by businesses to forecast future sales, regression can also be used to predict the price of assets such as cars or houses (Raschka and Mirjalili, 2019).

Unsupervised learning involves training an algorithm on data without labels. Unsupervised learning is commonly used for tasks such as clustering, association and dimensionality reduction (Raschka and Mirjalili, 2019). Two common real-world applications of unsupervised learning are customer segmentation and recommendation engines.

The final ML category is reinforcement learning. Reinforcement learning involves rewarding correct behaviour with positive reinforcement and punishing bad behaviour with negative reinforcement. Examples of reinforced learning include self-driving cars and chess game algorithms (Raschka and Mirjalili, 2019).

### 1.3.1 How machine learning can be applied for detecting CVD.

ML algorithms can be utilised as non-invasive tools for detecting and diagnosing CVD. This is because Supervised ML algorithms can be trained on existing patient medical data, during the training process these algorithms analyse the data and identify patterns. Once trained the algorithm can then be applied to new data and make predictions classifying patients as either having the disease or not. This can help healthcare professionals identify patients who may not have previously displayed any symptoms, allowing for early discovery and more time for patients to receive preventative treatments.

## 1.4 Significance

CVD is a deadly and burdensome condition that affects people across the world. Given the limitations of current diagnostic tools, ML shows promise as an alternative approach to detect CVD in people who may not have previously manifested symptoms. This early diagnosis can help healthcare providers make informed decisions regarding the necessary treatment required to avert premature mortality.

## 1.5 Aims and objectives.

The central aims of this project are to train, tune and evaluate several supervised classification ML algorithms for identifying patients with CVD.

### 1.5.1 Success criteria

1. Develop multiple ML models using data collected from the UCI ML repository.
2. Employ feature selection and hyperparameter optimisation to enhance each model's performance.
3. Evaluate and compare performances of each algorithm to determine the best algorithm at predicting patients with CVD.

## 1.6 Scope of study

The scope of this project will comprise building, fine-tuning and evaluating the performance of several supervised ML algorithms for predicting the presence of CVD using data from the UCI ML repository. Additionally, this project will integrate feature selection (FS) and hyperparameter optimisation in a bid to augment performance. This project will be constrained to the use of conventional ML algorithms and will not include deep learning algorithms such as artificial neural networks. This is because deep learning algorithms require larger datasets and copious computational power, neither of which I have access to.

## 1.7 Gaps in research

There have been numerous studies exploring the applications of ML for CVD diagnosis, with many studies ignoring data pre-processing techniques such as FS or hyperparameter optimization. The novelty of my project resides in the use of embedded FS using random forest as well as hyperparameter optimization to maximise recall for each model. With these techniques, I intend to close the gap in research and develop models with enhanced performance.

## 1.8 Ethical considerations

The dataset utilised in this project was collected from the publicly available UCI ML Repository and does not contain any personal data, Hence, there are no ethical or legal concerns associated with the use of this data. Furthermore, this project was approved by the university research ethic committee.

## 1.9 Code Availability

All code from this project can be found on the GitLab repository using the following link: <https://gitlab.uwe.ac.uk/aa25-ahmed/research-project.git>

# 2: Literature review

This chapter will review the current literature on the applications of ML in healthcare and disease diagnosis, additionally, this chapter will also discuss the various algorithms employed for CVD detection.

## 2.1 Applications of machine learning in healthcare

ML offers a broad range of applications in healthcare, with numerous studies investigating its practical applications. Habehh and Gohel (2021) conducted research addressing the different fields in which ML is being employed in healthcare. This study revealed how ML is currently being applied to electronic health records for the prognosis of diseases such as heart and kidney disease, Furthermore, this study also discussed how ML is also being applied in the field of medical imaging to assist radiologists in promptly detecting abnormalities in MRI, CT, and X-ray images.

In a similar study Javaid et al., (2022) explored a wider range of ML-related applications in healthcare, this study, like the previous one, included the topics of utilising ML on patient data to detect the early signs of diseases such as diabetes, heart disease and sepsis. However, this study also highlighted additional potential applications for ML, such as how ML can help in choosing candidates for clinical trials, which could shorten the length of the trial and result in a quicker drug release.

Dalal, (2020) investigated the implementations of ML in clinical settings. This study investigated several novel uses of ML in healthcare, including the development of personalised medications, smart health records and drug discovery using unsupervised learning. This study also addressed the existing hurdles of integrating ML into healthcare settings, with the most difficulty stemming from acquiring access to the data due to the numerous privacy policies adopted by governments.

## 2.2 ML for disease diagnosis.

Employing ML for disease diagnosis is supported by numerous studies. Kumar et al., (2022) conducted a systematic literature review to investigate the various uses of artificial intelligence (AI) for disease diagnosis. In this review they found that AI approaches are being used to help diagnose a range of diseases including Alzheimer's, cancer, heart disease, and diabetes. This is done by leveraging data collected from various sources such as electronic health records, MRI, and ultrasound images, by analysing these various sources of data, the algorithm can identify nonlinear patterns in the data and diagnose patients who may have gone unnoticed by clinicians.

The utility of ML in disease prediction was further proven in a study by Weng et al., (2017). In this study, four ML algorithms were applied to routine clinical data of 378,000 patients who previously displayed no signs of CVD’s. They compared the performance of these algorithms to an established algorithm set by the American college of cardiology to predict first cardiovascular event over a 10-year period. They discovered all four models achieved significant improvements in area under Receiver Operating Characteristic (ROC -AUC) score over the established algorithm with the neural networks (NN) model achieving the highest ROC -AUC score of 0.754, They concluded that ML is a valuable approach for identifying patients with CVD, with it being more accurate than the old school risk prediction methods, hence resulting in an increased number of patients benefiting from preventive treatments.

In a different study, ML has been demonstrated to reduce the cost and time of diagnosis by eliminating the time spent by clinicians on repetitive and laborious activities (Khanna et al., 2022). Early diagnosis of CVD can thwart premature mortality by promoting the use of preventative treatments thereby slowing disease progression. ML has also shown promise as a disease diagnostic tool in low-income nations where resources are scarce (Alabdaljabar et al., 2023). This is important since developing nations often lack access to conventional diagnostic instruments, necessitating alternative solutions.

## 2.3 Related works

In order to establish the most effective algorithm for CVD prediction, numerous studies have been conducted comparing the performance of various ML algorithms. Differences in findings have been influenced by variations in datasets, pre-processing techniques and hyperparameter tunings. In this section, I will review the current literature analysing the various algorithms, datasets, FS techniques and hyperparameter optimisation methodologies utilised.

### 2.3.1 Random Forest

The Random Forest Classifier (RF) is a supervised ML algorithm that consists of a forest of decision trees (DT). RF employs the ensemble learning approach known as bagging to generates an uncorrelated forest of DT’s, this decreases variance and consequently produces a model that is robust and resistance to overfitting (Raschka and Mirjalili, 2019). Many studies have been undertaken investigating the effectiveness of RF in predicting CVD. Singh et al., (2017) applied a RF algorithm to the Cleveland heart disease dataset and using 10-fold cross validation achieved an accuracy score of 85.81%, similarly Pal and Parija., (2021) employed RF on the same dataset and obtained an accuracy score of 86.9%, furthermore this study also evaluated performance using an AUC-ROC score and achieved a score of 93.3%. These studies show how well RF performs in predicting CVD, nevertheless one limitation with these two studies is the absence of details on pre- processing methods employed.

### 2.3.2 Logistic regression

Logistic regression (LR) is a supervised ML algorithm that is frequently used to solve binary classification problems. Logistic regression utilises a sigmoid logistic function to maps input features to take probability values between 0 and 1, this allows it to predict discrete categorical outcomes (Belyadi and Haghighat, 2021). In a recent study, Randhwan et al., (2022) employed a LR model to predict heart disease using the Cleveland Heart Disease dataset. This study's results showed that the LR model had an accuracy score of 85%, suggesting that it can be a useful algorithm for predicting the presence of CVD. Some drawbacks associated with this study include the lack of pre-processing techniques such as feature scaling, FS and one-hot encoding, as well as the absence of hyperparameter tuning.

### 2.3.4 K Nearest Neighbour

K nearest neighbour (KNN) is a non-parametric supervised ML algorithm commonly used for classification problems, KNN classifies new data point based on the geometric distance from those already stored in memory (Raschka and Mirjalili, 2019). The potential of KNN as an effective algorithm for CVD prediction has been demonstrated in a study by Shah et al., (2020), in this study they applied many different algorithms such as RF, KNN and DT to the Cleveland heart disease data set, they discovered the KNN algorithm achieves the highest accuracy score out of all algorithms tested with a score of 90.79%. Some limitations of this study were the lack of details in pre-processing techniques as well as hyperparameter optimisation techniques utilised.

### 2.3.5 Boosting algorithms

Boosting is an ensemble learning technique that entails constructing multiple weak learner decision trees in order to obtain a strong one. Models are trained iteratively, one after the other, this allows each model to learn from the mistakes of the prior model, consequently resulting in a powerful model with low bias (Raschka and Mirjalili, 2019). There are currently various boosting algorithms, each with minor intrinsic differences. The most prominent boosting algorithms are adaboost and gradient boost (GB).

The effectiveness of boosting algorithms on predicting coronary disease was demonstrated in a study by Habib et al., (2022), in this study the Framingham heart disease dataset was used to train five distinct boosting algorithms (adaboost, XGBoost, GB, LightGBM and CATBoost). The findings of this study revealed that four out of the five algorithms attained testing accuracies in the region of 87% with LightGBM being the sole algorithm that scored lower. Several shortcomings of this study include its disregard for cross validation and hyperparameter tuning. Furthermore, this study employed an imbalanced dataset and did not adopt a resampling technique like SMOTE. In the field of ML, imbalanced datasets are frowned upon and are notorious for producing models with misleading accuracy scores (He and Garcia, 2009).

### 2.3.6 XGBoost

XGBoost, originally created by Chen and Guestrin, (2016) is an open-source package that employs a range of hardware and software optimization methods to improve the GB algorithm. Budholiya et al., (2022) demonstrates the strength of XGBoost for CVD prediction, in this study they employed XGBoost with hyperparameter optimisation and one hot encoding on the Cleveland heart disease dataset. They compared the performance of the XGBoost model with two other tree-based ensemble techniques. According to this study, the XGBoost model outperformed the other models by 3.28% in terms of prediction accuracy, achieving a score of 91.8%.

In a similar study, Rajadevi et al., (2021) trained 5 algorithms, including an XGBoost model, on the Cleveland heart disease dataset. A major difference in this study compared to the previous one was the adoption of FS using the black hole algorithm. Nevertheless, this study found that the XGBoost model achieved the highest testing accuracy out of the proposed models, attaining an accuracy score of 92.3%. This study further demonstrates the strength of XGBoost in accurately predicting CVD. One important weakness of this study is that it did not address the use of hyperparameter optimisation, this could have improved accuracy even further.

### 2.3.7 Systematic reviews

A thorough analysis of 41 publications that employed ML to predict CVD was carried out in a recent systematic review by Azmi et al., (2022). In this review, they thoroughly investigated the ML algorithms and datasets most frequently used in the field of CVD prediction, furthermore they also highlighted the advantages and limitations of each study. One key discovery from this review was that the RF algorithm was the most consistent top performing algorithm, outperforming other algorithms in 13 of the 41 studies, thus, demonstrating its superiority for the accurate prediction of CVD.

One common problem they noticed throughout these studies were that the size of the dataset had a significant impact on the model's accuracy, with larger sample sizes generally producing worse performing models. As a result, many studies utilized small sample size datasets, inflating the accuracy scores and giving the model's performance a positive appearance. Another common problem they found was the lack of a consistent split between training and testing sets, with splits ranging from 80%-20% to 50%-50% in some studies. The split of a dataset is crucial and may have a detrimental impact on algorithm performance since dividing the data too thin can result in insufficient data for the algorithm to learn, resulting in an underfit model. Furthermore, they also uncovered that many studies did not employ FS, FS is a technique where irrelevant features are removed from a model, FS has been shown to reduce overfitting and improve model performance (Chandrashekar and Sahin, 2014). The lack of efficient FS strategies was also identified as a major issue faced by researchers in a systematic review by Kumar et al., (2022). Finally, the last issue they discovered was that many studies failed to identify the hyperparameter optimisation approach utilised, if at all. Hyperparameter optimisation is a well-known technique for enhancing the performance of an algorithm (Hashi and Zaman, 2020). Therefore, failing to tune hyperparameters can result in suboptimal model performance.

### 2.3.8 Importance of hyperparameter optimisation

Hyperparameters are a set of adjustable learning parameters of a ML algorithm that must be established before training a ML model, fine tuning these parameters in a specific combination can have a noticeable impact on model performance (Raschka and Mirjalili, 2019). The two most popular hyperparameter optimisation techniques are grid search and random search. The key difference between the two is that while random search randomly attempts different combinations, grid search exhaustively searches through all possible combinations (Raschka and Mirjalili, 2019). In a study by Bergstra and Bengio, (2012), random search was proven to be preferable to grid search due to it being less computationally expensive, allowing it to discover hyperparameters just as good as grid search in a considerably shorter timeframe.

The effectiveness of hyperparameter optimisation on the performance of ML models was investigated in a study by Hashi and Zaman (2020). In this study, five different ML algorithms (LR, KNN, Support vector machines (SVM), DT, RF) were applied to the Cleveland heart disease dataset and performance was evaluated both with and without the use of GridSearch hyperparameter optimization. They discovered that following hyperparameter tuning each algorithm showed significant improvements in performance, with accuracy’s ranging from 81.97% - 90.16% before tuning to 85.25% - 91.80% after tuning. This study demonstrates how GridSearch hyperparameter optimization can boost model accuracy, leading to better CVD prediction. Some limitations with this study are that it uses GridSearch as its hyperparameter optimisation technique, this has been shown by many studies including one I previously cited by Bergstra and Bengio (2012) to be inferior to a technique called random search, Moreover, this study does not make use of FS, which could potentially further enhance model performance.

### 2.3.9 Importance of feature selection

FS is a pre-processing procedure where redundant and irrelevant features are removed, leaving behind a subset of relevant features to be used in the model. FS techniques can be split into three categories named filter, wrapper and embedded. Filter methods incorporate statistical tests and have the advantages of being easy to implement and low in computational cost, however the downsides of filter methods are their inability to account for feature interactions (Chandrashekar and Sahin, 2014). Wrapper methods involve building many models with various feature combinations, this allows wrapper methods to select the optimum features for a specific algorithm, resulting in maximum algorithm performance, however the drawbacks with wrapper methods are that they are extremely computationally expensive. Embedded methods involve finding the optimal features while training the model, this allows them to be much faster than wrapper methods and more robust than filter methods. Overall, each category has its strengths and weaknesses, with embedded methods typically being the most efficient as they incorporate the benefits of the other two (Chemmakha et al., 2022).

The value of FS on model performance was assessed in a study by Mohammed et al., (2020). In this study 10 distinct ML algorithms were initially applied to the Cleveland and Hungarian heart disease datasets. Following performance evaluation, the best performing algorithm was re-trained with a smaller dataset that had undergone FS. They discovered that the algorithm with best performance on the full set of features was the Extra Trees (ET) classifier with an accuracy score of 92%, However, after using FS, the accuracy of the ET classifier climbed to 94.41%, with the maximum accuracy coming from the relief FS technique. This research highlights the advantages of FS in enhancing model performance.

Chemmakha et al., (2022) investigated the effectiveness of embedded FS in improving model performance for malware detection. This study found that embedded FS using RF and XGBoost provides a way of reducing computational time and complexity of ML models, Furthermore, this study revealed that the RF was the best embedded FS technique, achieving an accuracy score of 99.47%.

Doki et al., (2022) assessed the efficacy of five algorithms (RF, adaboost, LR, SVM, XGBoost) on the Cleveland heart disease dataset. Furthermore, two FS techniques (one wrapper and one embedded) along with grid search hyperparameter tuning were also employed. From the five algorithms tested the XGBoost model achieved the highest ROC-AUC score of 82%. However, after implementing hyperparameter optimization and recursive FS, the ROC-AUC score improved to 85.3%. This study demonstrates the significant influence that FS and hyperparameter optimization may have on model performance.

### 2.3.10 Potential flaws with ML

Some studies have shown that ML may not be significantly superior to traditional statistical methods in disease prediction, for instance, a study by Nusinovici et al., (2020) compared several ML algorithms (SVM, RF, NN, LR, KNN, GB) to a classical logistic regression model for the risk prediction of CVD, chronic kidney diseases, diabetes and hypertension. They discovered that logistic regression was equally as effective as the ML models with their only being a non-significant 1% difference between them. However, some drawbacks of this study were that they only utilised a limited number of predictors as well as there being a class imbalance in the dataset; both of these factors are known to impair the performance of ML models.

## 2.4 Literature Review Summary

To summarise this literature review, the recent research on employing ML algorithms to predict CVDs has yielded promising findings. Several studies have demonstrated that ML has a wide range of applications and that many ML algorithms are more than capable of delivering accurate CVD predictions. From the algorithms discussed RF, KNN, LR and boosting-based algorithms appear to have the most potential, typically outperforming the others. The research also demonstrated the benefits of techniques such as FS and hyperparameter optimization in enhancing model performance, with embedded FS and random search hyperparameter optimization showing the most promise. In this project, I will attempt to bridge gaps in the research by combining embedded FS with hyperparameter optimisation to enhance model performance. Furthermore, to determine the most effective algorithm, I'll be building, optimising and comparing the performance of the leading algorithms mentioned in this literature review, which include KNN, RF, LR, adaboost and XGBoost.

**Table 1.**  Comparative summary of studies mentioned in literature review.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Author / year** | **Title** | **Dataset** | **Algorithms applied** | **Best algorithm found** | **Testing Accuracy (%)** | **Limitations with study** |
| Randhawan, (2022) | Heart Disease Prediction Using Logistic Regression Algorithm | Cleveland Heart disease dataset | LR | LR | 85 | Inadequate specification of pre-processing techniques implemented.  Only one algorithm utilised. |
| Mohammed et al., (2020) | Early and accurate detection and diagnosis of heart disease using intelligent computational model | Cleveland and Hungarian heart disease dataset | DT, KNN, ANN, RF, SVM,  ET, GB, AB, Naïve Bayes, LR | ET | 94.41 | Hyperparameter optimization not considered. |
| Singh et al., (2017) | Heart Disease Prediction System Using Random Forest | Cleveland Heart disease dataset | RF | RF | 85.81 | One algorithm utilised.  No details on FS and hyperparameter optimisation techniques utilised |
| Pal and Parija, (2021) | Prediction of Heart Diseases using Random Forest | Cleveland Heart disease dataset | RF | RF | 86.9 | One algorithm utilised.  No details on FS or hyperparameter optimisation techniques utilised |
| Shah et al., (2020) | Heart Disease Prediction using Machine Learning Techniques | Cleveland Heart disease dataset | RF, KNN, DT, Naïve Bayes | KNN | 90.79 | No details on FS or hyperparameter optimisation techniques utilised |
| Habib et al., (2022) | A Study on Coronary Disease Prediction Using Boosting-based Ensemble Machine Learning Approaches | Framingham heart disease dataset | adaboost, XGBoost, GB, LightGBM, CATBoost | XGBoost | 87.62 | Imbalanced dataset,  no cross validation, hyperparameter tuning not included. |
| Budholiya et al., (2022) | An optimized XGBoost based diagnostic system for effective prediction of heart disease | Cleveland heart disease dataset | XGBoost, ET, RF | XGBoost | 91.8 | FS not employed |
| Rajadevi et al., (2021) | Feature Selection for Predicting Heart Disease Using Black Hole Optimization Algorithm and XGBoost Classifier | Cleveland heart disease dataset | XGBoost, KNN, SVM, DT, RF | XGBoost | 92.3% | Hyperparameter optimisation not utilised. |
| Doki et al., (2022) | Heart Disease Prediction Using XGBoost | Cleveland heart disease dataset | RF, adaboost, LR, SVM, XGBoost | XGBoost | 85.96% | Grid search hyperparameter tuning used rather than random search |
| Hashi and Zaman, (2020) | Developing a Hyperparameter Tuning Based Machine Learning Approach of Heart Disease Prediction | Cleveland heart disease dataset | LR, KNN, SVM, DT, RF | KNN | 91.8% | No FS |

# 3: Methodology

In this chapter, I will describe the methodological approaches used in this dissertation, I will also discuss information about the dataset as well as techniques used to clean and pre-process the data, Additionally, I will discuss the types of exploratory data analysis implemented and the insights discovered, finally I will discuss the ML algorithms, hyperparameter optimisation method and appropriate evaluation metrics applied.

Diagram

Description automatically generated

**Figure 1.** Workflow diagram of proposed methodology

## 3.1 Dataset

The dataset, known as the Cleveland heart disease dataset, was obtained from the UCI ML Repository. This dataset was compiled by Dr Robert Detrano and was originally released in 1988 for the Cleveland Clinic Foundation. The Cleveland heart disease dataset is one of the most frequently used data sources in the field of CVD research and has been utilised in numerous studies to train and test ML models for CVD prediction. This data source is extensively used since it encompasses a diverse set of patient characteristics and is representative of the general population.

### 3.1.1 Importing, cleaning and formatting data.

The raw data file named “processed.cleveland.data” was imported directly from the UCI ML repository into a Python Jupyter notebook, where it was then transformed into a Pandas data frame, this allowed the data to be cleaned and ready for ML analysis. The first step of cleaning the data was to format the columns as the raw file did not include column names, the next step of data cleaning was to identify and eliminate any missing values, this is necessary since most ML algorithms cannot operate with missing values (Géron, 2019).

The Pandas “isnull().sum()” function was employed to calculate the number of missing values for each attribute. This initially revealed that there were no missing values, however upon further investigation, it was revealed that some missing values were represented using the symbol '?'. 6 records were identified to contain missing values, these records were subsequently dropped from the dataset (Fig 2).

Table

Description automatically generated with medium confidence

**Figure 2.** Screenshot of functions employed to detect missing values.

After eliminating missing values, each attribute was converted to the appropriate datatype. This was done to enable accurate data visualisations, improve data performance, and ensure precise numerical calculations.

The final step of data cleaning was transforming the target attribute into a binary format, this transformation was carried out because the goal of this project was to develop ML models that would be able to accurately predict whether patients have CVD or not. This conversion would allow the data to be used for binary classification.

Initially, the target variable contained 5 classes with labels ranging from 0 to 4, however, through information relayed on the UCI ML repository I discovered that a label of 0 represented the absence of CVD and labels 1-4 indicated various types of CVD. To facilitate the data to be used for binary classification, records with a class label other than 0 were promptly converted to 1 (Fig 3).

Graphical user interface, text, application, email

Description automatically generated

**Figure 3**. Screenshot showing the target variable being reduced from five categories to two.

### 3.1.2 Dataset description

14 attributes and 297 instances make up the cleaned dataset. The attributes are divided into 13 features and one target variable. The features are made up of numeric continuous variables such as age and maximum heart rate, in addition to discrete categorical variables such as gender and chest pain type. The target variable is discrete and has two classes; a value of 1 denotes that the person has CVD, while a value of 0 denotes that they do not.

**Table 2.** Dataset attributes with description

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute name** | **Abbreviation** | **Data type** | **Description** |
| Age | age | Continuous | Age of patient in years |
| Sex | sex | Discrete | Gender: Female = 0, male = 1 |
| Chest pain | cp | Discrete | Chest pain type: typical angina = 0, atypical angina =1, non- anginal pain = 3, asymptotic = 4 |
| Resting blood pressure | trestbps | Continuous | Resting blood pressure (mmHg) on admission to hospital |
| Serum cholesterol | chol | Continuous | Serum cholesterol (mg/dl) |
| Fasting blood sugar | fbs | Discrete | Is fasting blood sugar > 120 mg/dl:  0 = false, 1 = true |
| Rest Electrocardiograph | restecg | Discrete | ECG results at rest: 0 = normal, 1 = ST wave abnormality, 2 = left ventricular hypertrophy |
| Max Heart rate | thalach | Continuous | Maximum heart rate achieved (bpm) |
| Exercise induced angina | exang | Discrete | Presence of angina after exercise: 0 = false, 1 = true |
| Old peak | oldpeak | Continuous | ST depression induced by exercise relative to rest |
| Slope | slope | Discrete | The slope of the peak exercise ST segment: 0 = upslope, 1 =flat, 2 = downslope |
| Number of major vessels coloured | ca | Discrete | The number of major vessels coloured by fluoroscopy, 0-3 |
| Thalassemia | thal | Discrete | Type of thalassemia: 0 = normal, 1 = fixed defect, 2 = reversable defect |
| Target | target | Discrete | Target class: 0 = no CVD, 1 = presence of CVD |

## 3.2 Exploratory data analysis

To discover more about the trends, distributions and correlations of the dataset, exploratory data analysis (EDA) was carried out. EDA involves summarising and visualising the dataset, this makes it possible to understand relationships and patterns that might not be immediately clear.

### 3.2.1 Univariate analysis

Univariate analysis was carried out to summarise and visualise the range and distribution of specific attributes, as well as spotting potential outliers. The first step was to print out summary descriptive statistics (fig 4), this helped in assessing the measurements of central tendency and dispersion for each field in the dataset. A notable finding from the descriptive statistics was that patients ages ranged from 29 to 77, with an average age of 55.

Table

Description automatically generated

**Figure 4.** Descriptive statistics

Secondly histograms and box plots were constructed for continuous variables, histograms were used to illustrate the distribution of specific variables, allowing me to identify whether they were normally distributed or skewed, this helped me determine if any appropriate log transformations needed to be performed. Box plots allowed for the visualisation of the spread, the viewing of the median, and the identification of any outliers from each feature. Examining the continuous variables in (fig 5), it appears the majority of numeric features are normally distributed with slight positive or negative skews. However, the only exception is old peak which exhibits a highly positive skew.

Chart, histogram

Description automatically generated

**Figure 5.** Histograms and box plots revealing Distributions of continuous features.

Bar graphs were utilized to visualise the distribution of categorical features (fig 6), the key observation from figure 6 Is that:

* There is a higher number of male patients compared to female.
* The majority of patients have a fasting blood sugar level below 120mg/ml.
* The most prevalent type of chest pain is asymptomatic.
* The incidence of ST-T wave abnormality is very low, and the distribution of rest ECG is uniformly split between left ventricular hypertrophy and normal.
* There are significantly more patients without exercise induced angina compared to with it.
* The majority of patients have 0 coloured blood vessels by fluoroscopy, with the number of patients decreasing as the number of coloured blood vessels increases.

Chart, treemap chart

Description automatically generated

**Figure 6**. Bar chart showing distribution of categorical features.

The distribution of the target variable was visualised using a bar and pie chart (fig 7). From figure 7 it can be observed that the target variable is binary, split 160:137 with slightly more individuals displaying cvd absence than presence. It is important to determine the class ratio in the target variable since imbalanced classes can lead to misleading accuracy scores (He and Garcia, 2009). With this knowledge, I can decide the appropriate evaluation metrics to employ.

Chart

Description automatically generated

**Figure 7.** Bar chart and Pie chart of class distribution of Target variable.

### 3.2.2 Bivariate analysis

A bivariate analysis is one that looks at the relationship between two variables. In this project a bivariate analysis was conducted to examine the relationships and distributions between the features and the target variable. This would help identify the features that have the greatest influence on the target, providing a deeper understanding of feature interactions and ultimately assisting in the selection of features for ML models.

Figure 8 depicts the relationship between the categorical features and the target. From this figure the key observation I can make are that

* Individuals with cvd often experience asymptomatic chest pain.
* Males are more prone to CVD compared to females.
* The presence of exercise indued angina is a clear sign of the manifestation of CVD.
* Patients with CVD are more likely to have a defective thallium test.

Chart, bar chart, treemap chart, PowerPoint

Description automatically generated

**Figure 8.** Bar charts of categorical feature with target as hue

The use of box plots assisted in comparing the range of distributions in numeric features between healthy and CVD patients (Fig 9). Figure 9 demonstrates that:

* Patients with CVD have a higher median and a narrower range of distribution of age compared to those without CVD.
* CVD patients have a significantly lower median maximum heart rate and higher median for oldpeak when compared to healthy individuals.

These findings suggest that age, maximum heart rate and oldpeak may potentially serve as strong predictors for ML models. On the contrary, the median and range of distribution for both serum cholesterol and resting blood pressure appears to be similar in both healthy and CVD patients, suggesting that these features may not be strong predictors in our models.

Chart, box and whisker chart

Description automatically generated

**Figure 9.** Grouped Box plots of numeric features.

### 3.2.3 Multivariate analysis

Multivariate analysis was conducted to determine the relationships between each of the attributes. First, a correlation matrix was constructed to visualise the strength and direction of any linear correlations between the attributes. A correlation matrix displays the magnitude of the Pearson correlation coefficient between each pair of attributes in a color-coded format, making it easier to identify significant correlations (Raschka and Mirjalili, 2019). The Pearson correlation coefficient ranges from -1 to 1 with positive values signifying a positive correlation and negative values representing a negative correlation, When the coefficient value is zero, it indicates that there is no linear correlation between the variables (Géron, 2019).

In this project, a correlation matrix was utilized to determine the correlations between the features and the target, this would provide insight into which features are likely to be strong predictors for our ML models. The correlation matrix also aids in multicollinearity detection by revealing the magnitude of the correlation between each pair of features.

From the correlation matrix (fig 10), we can see the features with strongest correlation to the target variable are type of thalassemia (thal; 0.52), the number of major vessels coloured by fluoroscopy (ca; 0.46), maximum heart rate (thalach; -0.42) and ST depression induced by exercise (oldpeak; 0.42). The features with the weakest correlation to the target were identified to be fasting blood sugar (fbs; 0.0032) and serum cholesterol (chol; 0.08)

Treemap chart

Description automatically generated with medium confidence

**Figure 10.** Correlation matrix.

The next step of the multivariate analysis was to construct a pair plot. A pair plot is a graph that displays scatter plots for every combination of numeric attributes in the data set, this provides a visual representation of the relationships and type of correlations between the attributes. Additionally, a pair plot also showcases the distribution of each feature through a kernel density estimate plot.

The pair plot in figure 11 reveals several weak positive and negative relationships from the scatter plots between the features. The kernel density estimate plots reveals that although the distributions of most features between CVD and healthy patients are similar, there is a marked discrepancy for the features oldpeak and thalach. This indicates that these two features will likely be strong predictors of CVD in the ML models.

Chart

Description automatically generated

**Figure 11.** Pair plot with Target as Hue

## 3.3 Data pre-processing

Following data cleaning and EDA, the data required pre-processing for ML analysis.

### 3.3.1 One hot Encoding

ML algorithms do not permit the use of categorical data directly. Categorical data must first be transformed into numeric data before it can be used. One approach commonly used to transform categorical data is called one-hot encoding. During one-hot encoding, features of categorical string data are transformed into multiple features of binary numeric data, this allows the data to be used for ML algorithms (Raschka and Mirjalili, 2019).

In Python, one-hot encoding can be accomplished in a variety of ways. The Scikit learn library includes several functions for transforming categorical data, such as OneHotEncoder and Column Transformer. However, the simplest technique and the method I opted for was to use the get\_dummies function from the Pandas library. In this project, the execution of one hot encoding grew the feature space from 14 to 29 features.

### 3.3.2 Feature scaling

Following one hot encoding the next step of data pre-processing was feature scaling. Feature scaling is one of the most vital steps in data pre-processing and is defined as the transformation of features to a homogeneous scale. Feature scaling is employed when numerical features have different units, this is because differing units can cause many ML algorithms to underperform (Géron, 2019).

Feature scaling can improve performance of gradient decent based algorithms since it can help them to converge more quickly, feature scaling is also important in geometric distance calculating algorithms like KNN, this is because the similarity measure known as Euclidean distance is sensitive to magnitude, therefore by scaling the features to a uniform scale allows all the features to contribute equally (Raschka and Mirjalili, 2019).

There are several techniques available to scale features, each with advantages and disadvantages. The two most widely used methods are normalisation and standardisation. Normalisation, also referred to as min-max scaling, rescales the features to a specific range typically between 0 and 1. This is useful when the data does not follow gaussian distribution, however one drawback of normalisation is that it is susceptible to outliers (Raschka and Mirjalili, 2019). Standardization entails rescaling features to a normal distribution with a mean of 0 and standard deviation of 1, standardisation is beneficial if you know the data follows a gaussian distribution, standardisation also preserves pertinent data about outliers, therefore reducing the algorithms susceptibility to them (Raschka and Mirjalili, 2019).

In this project feature scaling was implemented utilizing the standardisation technique. This is due to EDA analysis revealing the majority of the numerical features follow a gaussian distribution, additionally standardisation is generally considered a more robust method as it is less influenced by outliers. Standardisation was implemented in python using Scikit learns standard scaler function.

### 3.3.3 Feature selection

Not every feature in a dataset will be helpful for a model, some features may even hinder a model’s performance. FS is a process where redundant and irrelevant features are removed, leaving behind a subset of relevant features to be used in the model. FS has several benefits, including reduced training time, improved model performance and enhanced model interpretability (Chandrashekar and Sahin, 2014). Additionally, FS reduces the dimensionality of a dataset, where dimensionality relates to the number of input features. This is important because too many input features can lead to the curse of dimensionality, this is when a model's performance suffers as a result of having too many features. The Dimensionality of a dataset is important for algorithms that calculate geometric distance between data points such as KNN, this is due to the fact that in a large multidimensional space the geometric distances between instances become similar, thus making it extremely difficult to classify new instances. (Raschka and Mirjalili, 2019)

Filter, wrapper, and embedded methods are three groups of techniques commonly used for FS. Filter methods involve statistical tests such as Pearson’s correlation, fisher score or chi squared and are performed prior to building a model. Features are ranked based on the results of these tests, features with high correlation with the target are given a high rank while those with low correlation given a low rank, low-ranking features can subsequently be chosen to be eliminated from the dataset (Chandrashekar and Sahin, 2014). Some advantages of filter methods include their low computational cost, algorithm independence and ability to prevent overfitting. (Kolukisa and Bakir-Gungor, 2023). However, one notable disadvantage of filter methods is that they do not consider interactions between features, this may result in the elimination of features that though maybe not valuable on their own, are quite significant when paired with other features. (Chandrashekar and Sahin, 2014).

Wrapper methods entail building multiple models using various combinations of input features, after which features are selected from the model with the highest performance for a specific evaluation metric. Examples of wrapper methods include forward selection and backwards elimination. Starting with no features in the model, forward selection adds in the pre-determined best features one by one until the performance of the model stops progressing. Backwards elimination begins with all features and progressively eliminates the least relevant features one by one until model performance plateaus. The primary drawbacks with wrapper methods are that they are extremely time consuming and computationally expensive as they require training multiple models using various feature subsets, additionally wrapper methods render the algorithm susceptible to overfitting (Kolukisa and Bakir-Gungor, 2023).

Embedded methods find the best set of features while training the algorithm, Embedded methods include regularisation algorithms like Lasso as well as algorithms based on decisions trees such as RF.

FS with RF involves the use of feature importance, feature importance is a score and is calculated from the average impurity decrease from all the decision trees in the forest (Raschka and Mirjalili, 2019). The feature importance score refers to how important each feature is at predicting the target variable, with a higher score given to more significant features, this allows low importance feature to be identified and discarded.

In this project the FS approach chosen was the embedded method. This method was selected because it provides a good middle ground between filter and wrapper methods, for example, they are less prone to overfitting and require less computation than wrapper methods, additionally they generally outperform filter methods while also considering feature interactions (Kolukisa and Bakir-Gungor, 2023).

RF FS was implemented by first, initialising and fitting a RF model to the training data, next a feature importance ranking graph was printed out (fig 12). This graph revealed the most and least important features in producing the RF model. The feature “restecg\_1” was identified as being the only feature with a feature importance of 0, hence was subsequently removed.

Chart, bar chart, histogram

Description automatically generated

**Figure 12**. Graphical representation of RF feature importance.

## 3.4 Machine learning algorithms used.

Following data pre-processing and exploratory data analysis, ML algorithms were trained and evaluated using the scikit learn library from python. In the next section, I will delve into the background of some of the ML algorithms used in this project.

### 3.4.1 K nearest Neighbors

KNN is a non-parametric, instance based, supervised ML algorithm that can be applied for both classification and regression tasks. KNN belongs to a group of algorithms known as "lazy" learners because rather than developing a model, it compares new data points to those already stored in memory (Raschka and Mirjalili, 2019). The fundamental concept of KNN is that new instances are classified by a majority vote based on their geometrical distance from their k nearest neighbours. A similarity measure, such as Euclidean distance is employed to compute the geometric distance between the new point and its neighbours; the new instance is then admitted to the class by majority vote (Raschka and Mirjalili, 2019).

The number of neighbours the model takes into account and computes geometric distances for is determined by the hyperparameter k. Choosing the right value of k is crucial as it can greatly affect the performance of the model. A small k value can lead to a model with high variance and low bias, resulting in an overfit model that does not generalise well to new data. A k value that is too big can lead to a model with high bias and low variance, this will result in a underfit model that performs poorly on both training and testing data. The ideal model will have a k value that achieves the optimum equilibrium between variance and bias, producing a model that performs well on both training and test data (Raschka and Mirjalili, 2019).

The advantages of employing a KNN model are that it is simple and easy to implement, additionally as it is non-parametric, KNN does not make any assumptions about the data. The disadvantages of KNN are it can get computationally expensive with large data sets. Furthermore, it does not work well with high dimension data due to the curse of dimensionality, finally KNN requires the precise tuning of the hyperparameter k for optimal performance.

### 3.4.2 Logistic regression

Logistic regression is a supervised ML algorithm that is frequently used to solve binary classification problems. In contrast to the widely known linear regression model, which uses a straight line to predict continuous values, logistic regression uses a sigmoid logistic function to predict discrete categorical outcomes. The sigmoid logistic function maps input features to take probability values between 0 and 1, predictions that have probability larger than 0.5 are put into one class, while the remainder are put into a separate class (Belyadi and Haghighat, 2021).

One advantage of logistic regression is that it is a “white box” model, white box models are algorithms that are known for being intuitive and easy to interpret (Géron, 2019). The disadvantage of logistic regression is that it performs poorly when there is multicollinearity in the dataset (Senaviratna and Cooray, 2019).

### 3.4.3 Random Forest

RF is a versatile supervised ML algorithm that can be applied for both classification and regression problems. RF is a type of algorithm that employs ensemble learning. The concept of ensemble learning is based on the adage "wisdom in crowds." In this context, a group of weak learners combine forces to create a strong learner, thus increasing the performance of the algorithm (Géron, 2019).

The RF algorithm is composed of hundreds of uncorrelated decision trees that work collaboratively as an ensemble, each tree in the forest is trained on a random subset of the data, every tree then yields a class prediction, The final prediction is then determined by a majority vote (Raschka and Mirjalili, 2019).

The primary advantage of RF algorithms is that they are resistant to overfitting, this is because they employ the ensemble learning technique called bagging which reduces variance. Bagging short for bootstrap aggregating entails creating numerous bootstrap samples from training data, after which models are built from each sample in parallel and independent of other samples, finally they are aggregated, and a majority vote is taken. As a result of its resistance to overfitting, RF frequently result in models with excellent generalisation (Raschka and Mirjalili, 2019).

The major disadvantages with RF are that they are slow to compute and computationally expensive, this is largely due to their recruitment of many decision trees. Furthermore, RFs are recognised as being “black box” algorithms, this is because they are more difficult to interpret than conventional algorithms like the simple decision tree (Géron, 2019).

### 3.4.4 Adaptive boost

Adaboost short for adaptive boost is a decision tree-based supervised ML algorithm that utilizes the ensemble learning technique called boosting. Each of the decision trees used by Adaboost has one node and one split, these trees are referred to as stumps. Weak learner models are trained sequentially using error-based weighting that gives wrong predictions additional weight. This is performed a set number of times, and the results are combined using a weighted total (Géron, 2019).

The advantages of adaboost are that it is simple to implement and tune with just a few hyperparameters to optimise. The disadvantages with adaboost is their lack of interpretability due to their “black box” nature.

### 3.4.5 XGBoost

Extreme gradient boost, or XGBoost, is an open-source library that aims to enhance the gradient boosting paradigm using a variety of hardware and software optimization techniques. Gradient boost is a boosting technique similar to adaboost, however unlike adaboost, gradient boost seeks to fit the new predictor to the residual error of the prior predictor rather than changing weights for incorrect predictions (Géron, 2019). Initially developed by work done by Chen and Guestrin (2016), XGBoost is widely regarded as one of the best ML algorithms renowned for its ability to swiftly build models that are extremely accurate without being overfit. Parallelization, tree pruning, regularisations and cross validation are a few examples of the optimisation techniques implemented by XGBoost.

## 3.5 Cross-validation

Training and testing an algorithm using the entire dataset can result in a highly complex model with high variance, this type of model is referred to as overfit. An overfit model is one that learns the detail and noise in the training data too well. Overfit models perform exceptionally well on training data however perform poorly on unseen test data (Géron, 2019). One simple technique to prevent this from happening is to split the data into training and testing sets, this enables the algorithm to be trained on a subset of data, typically 80%, known as training data, and evaluated on the remaining unseen data, known as testing data, ultimately resulting in a model that generalises better. Train test split can be done in python using sklearns library’s train\_test\_split function.

One drawback of the train-test split technique is that you can't choose which subset of the data is utilised for training and testing. This is where the cross-validation technique comes in handy. Cross-validation divides the dataset into blocks and uses each data block one at a time, summarising the results at the end, this eliminates the burden of worrying about which part of data is used as the train or test set, The primary objective of cross validation is to prevent overfitting and encourage model generalisation.

There are numerous cross-validation methods, with k-fold cross validation being the most prominent. During k-fold cross validation, the dataset is divided into (k) number of folds of equal size, k-1 folds are used for training and 1 fold used for testing. This is repeated k number of times until each fold has served as a testing set once. The results from the evaluation of each test fold are pooled into an array of scores from which a mean is computed (Raschka and Mirjalili, 2019), this score provides a greater insight of the model’s overall performance.

Cross-validation can be used for both regression and classification tasks, however, when dealing with classification problems with unequal class ratios the optimal technique to use is stratified k fold, this is because it ensures that the target class ratio in each fold is the same as the ratio in the overall dataset. Research conducted by Kohavi, (1995) has demonstrated that stratified k fold cross validation is the best cross validation technique because it leads to reductions in bias and variance hence produces models that are more accurate and generalizable.

in this project I have decided to first split the data 80:20 using train\_test\_split function. I've also opted to employ stratified K-Fold cross-validation as my cross-validation approach of choice. With this method, each fold will have a balanced representation of each class, allowing for a more realistic assessment of the model's performance.

## 3.6 Hyperparameter Optimisation.

Hyperparameters are a group of parameters of a learning algorithm that are set by the user before training a model, it is essential to appropriately fine-tune these hyperparameters with the optimal combinations as they can have a significant impact on how well an algorithm performs. There are several approaches of tuning hyperparameters with the two most well-known techniques being GridSearch and RandomizedSearch (Raschka and Mirjalili, 2019).

GridSearch is a comprehensive search technique that builds models for all conceivable combinations of the provided hyperparameters, each model is then evaluated using cross validation with the hyperparameters from the best performing model spit out. GridSearch has the downside of being extremely time-consuming, with the amount of time required increasing exponentially with each additional hyperparameter (Raschka and Mirjalili, 2019).

RandomSearch differs to GridSearch as rather than analysing every possible combination of hyperparameters, it instead randomly tries different combinations of hyperparameters. This allows RandomSearch to test a wider range of hyperparameters in substantially less time than GridSearch (Raschka and Mirjalili, 2019). Numerous studies have shown the benefits of employing random search over GridSearch, with a study by Bergstra and Bengio (2012) stating that RandomSearch has a 95% chance of producing results that are within 5% of the optimal performance when considering 60 hyperparameter combinations. As a result of these findings, as well as time and computing constraints, RandomSearch will be the hyperparameter optimisation technique used for this project.

### 3.6.1 Hyperparameters tuned for each algorithm.

The most important hyperparameter for the KNN algorithm is “n\_neighbors”, this defines the number of neighbours for which distances are calculated. It is critical to carefully adjust this hyperparameter, since a low value can result in high variance and overfitting while a large value can increase bias resulting in underfitting (Belyadi and Haghighat, 2021).

Three common hyperparameters, “solver”, “penalty”, and “C” require tuning for the LR algorithm. “Solver” is the type of optimization algorithm employed, “Penalty” refers to the type of regularisation applied, either l1 or l2. Finally, “C” represents the strength of regularisation applied, with a smaller value invoking stronger regularisation (Belyadi and Haghighat, 2021).

The decision tree-based ensemble algorithms, such as RF, adaboost and XGBoost, are composed of many decision trees. The number of trees in the model is governed by the “n\_estimators” hyperparameter. Increasing the number of trees decreases variance and leads to a more reliable model, however more trees also carry the burden of a longer computational time.

The “max depth” hyperparameter limits the depth of the decision tree, a high tree depth can increase risk of overfitting, hence by restricting the depth of a tree we can avoid overfitting (Belyadi and Haghighat, 2021).

“Learning rate”, also referred to as shrinkage, is a hyperparameter used in boosting algorithms. It controls the extent to which each tree contributes to the final prediction and is utilised to prevent overfitting. Decreasing the learning rate usually necessitates increasing the number of trees in the model to compensate (Géron, 2019).

## 3.7 Evaluation methodology

The performance of each classification algorithm was assessed using several evaluation metrics, namely accuracy, precision, recall, f1 score and ROC -AUC score, furthermore confusion matrices and ROC curves were constructed to visualise these metrics. By employing a range of evaluation metrics, I will be able to provide a comprehensive assessment of each model's performance since a model may perform well using one metric but poorly with another.

### 3.7.1 Accuracy

Accuracy is the simplest evaluation metric for classification algorithms and is the optimal choice when the classes of a dataset are equally balanced. Accuracy is calculated by dividing number of correct predictions by the total number of predictions. (Belyadi and Haghighat, 2021).

Accuracy = (1)

Accuracy is frequently divided into two categories, training accuracy and testing accuracy. Testing accuracy, commonly known as simply accuracy, is the model's accuracy on unseen data. Training accuracy is the accuracy when the model is evaluated using the same data that was used to train the model. The training accuracy of a model can give insight to whether the model is overfit or not, a model with very high training accuracy but low testing accuracy will indicate an overfit model.

### 3.7.2 Confusion matrix

Confusion matrices were constructed to visualise and identify the specific types of errors committed by each model. Each confusion matrix revealed the count of a classifier's true positive, true negative, false positive, and false negative predictions, this facilitated the direct derivation of evaluation metrics including accuracy, precision, and recall.

With the aid of a confusion matrix, accuracy can be worked out as:

Accuracy = (2)

In the context of this project a True Positive (TP) is the algorithms correct prediction of an individual as having CVD, a True Negative (TN) is the algorithms correct prediction of an individual not having CVD. A false positive (FP) referred to as a type 1 error is when the algorithm incorrectly predicts an individual as having CVD, finally a False Negative (FN) referred to as type 2 error is when the algorithm incorrectly predicts an individual as not having CVD.

### 3.7.3 Precision

Precision is defined as the proportion of true positives from all positives; it gives an insight of how accurate the positives predictions are, it is calculated using the following formula.

Precision = (3)

### 3.7.4 Recall

Recall also known as sensitivity is defined as the proportion positives correctly classified,  
recall is calculated using the formula (Géron, 2019).

Recall = (4)

Recall Is a valuable metric in medical diagnostics, as it measures the percentage of true positives correctly classified from all patients with the disease (Bolin and Lam, 2013). Recall is especially important when dealing with life threating conditions such as CVD, this is because a model with low recall will result in many patients remaining undiagnosed, this can lead to serious health consequences and potentially death, hence by maximising recall we can result in less patients remaining undiagnosed, allowing preventative treatment and potentially saving lives.

### 3.7.5 Precision recall trade off

Precision and recall have a trade-off, with a larger value of one translating into a lower value of the other. In the context of this project a model with high precision and low recall will strive to minimise false positives hence will only classify individuals as having CVD if it is highly confident, this can have potentially fatal ramifications since it can lead to many patients with CVD remaining undiagnosed. A model with high recall and low precision aims to minimise false negatives, it does this by classifying individuals as having cvd even when it is not entirely confident. This can be advantageous because it would mean no individual with CVD would go untreated, However, one consequence of a high recall model is that it can lead to the algorithm misclassifying healthy individuals as having CVD, resulting in additional costs and unnecessary treatments for individuals who do not have CVD.

My goal for this project is to develop models that can accurately predict patient with CVD while also ensuring the fewest possible patient with CVD remain undetected. This is important because CVD is a devastating disease that if left untreated can result in serious health complications and ultimately death. To achieve this goal, I will focus on constructing models with high accuracy as well as recall. To attain high recall models, I will optimise model hyperparameters for recall, this will be accomplished by specifying recall as the scoring parameter of the RandomSearch object. This will allow me to identify the best hyperparameters that will yield models with the highest possible recall.

### 3.7.6 F1 score

F1 score is defined as the harmonic mean of precision and recall and is calculated using the following formula.

F1 score = 2 x (4)

The model's F1 score, which is only high if the precision and recall values are high, will provide insights into how well the model balances these two factors, I believe f1 score will be a more informative metric than precision and recall values alone when comparing performance between the different algorithms utilised in this project.

### 3.7.7 ROC-AUC

ROC curves are graphs which plot the True positive rate against False positive rate, they are constructed to provide graphical illustrations of the performance of each algorithm at different classification thresholds (Raschka and Mirjalili, 2019). The area under the ROC curve (ROC-AUC), which ranges from 0.5 to 1, provides a numerical metric of the performance of the classifier. A flawless classifier will have a ROC curve at the top left corner of the plot and an AUC score of 1. A ROC curve close to the diagonal line with an AUC score of 0.5 represents a very poor classifier that performs as well as a random guess (Raschka and Mirjalili, 2019). ROC-AUC score is known to be a very robust evaluation metric, with a study by Bradley (1997) recommending that ROC-AUC score to be used rather than accuracy as a single number evaluation metric for ML algorithms.

## 3.8 Model building workflow.

Following data pre-processing, which involved FS, feature scaling and splitting the data into training and testing sets, the data was finally ready to be used for ML analysis.

First, for each algorithm, a unique hyperparameter grid was established, each grid consisted of a dictionary containing a list with a range of values for each relevant hyperparameter that would aid in enhancing the performance of the model.

Next the RandomizedSearchCV function was employed to generate the optimal set of hyperparameters for each model. This function took in several arguments including: the algorithm to be used, the related hyperparameter grid, the type of cross validation (In this project stratified K-fold with 10 splits was used) and the scoring metric we are striving to maximise (recall was prioritized in this project).

Each random search object was subsequently fit to the training data and optimal hyperparameters were printed out. New models incorporating these optimal hyperparameters were initialized and fit to the training data.

Finally, predictions were made using the test data and evaluation metrics such as accuracy, precision, recall and f1 score were printed out, additionally confusion matrices and ROC curves were constructed to provide a visual representation of performance.

Figure 13 displays a screenshot of the RF classifier construction process.

Text

Description automatically generated

**Figure 13.** screenshot of building, tuning and evaluating RF Classifier.

# 4: Results and Analysis

This chapter delivers a comparative analysis of the experimental results obtained from evaluating each ML algorithm employed in this project.

**Table 3.** Summary Table of evaluation scores for each model

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Classifier** | **Train Accuracy %** | **Test Accuracy**  **%** | **Precision**  **%** | **Recall**  **%** | **F1 score**  **%** | **ROC-AUC Score** |
| **KNeighbors** | 86.5 | 81.7 | 76.7 | 85.2 | 80.7 | 0.915 |
| **Logistic Regression** | 86.1 | 83.3 | 79.3 | 85.2 | 82.1 | 0.898 |
| **RandomForest** | 88.2 | 81.7 | 80.8 | 77.8 | 79.2 | 0.897 |
| **AdaBoost** | 85.7 | 83.3 | 81.5 | 81.5 | 81.5 | 0.872 |
| **XGBoost** | 86.5 | 85.0 | 84.6 | 81.5 | 83.0 | 0.919 |

Chart, line chart

Description automatically generated

**Figure 14**. Graphical illustration of each model’s performance via ROC curves.

## 4.1 K-Nearest Neighbors results

The KNN model achieved a training accuracy of 86.5% and testing accuracy of 81.7%, this testing accuracy is tied for the worst amongst the evaluated models suggesting that this model did not generalise as well to new data, additionally this model produced an f1 score of 80.7% which was the second lowest amongst models indicating that the model's balance between precision and recall is weaker compared to the other models. However, the model achieved the second highest ROC-AUC score of 0.915. The confusion matrix uncovers the type of errors produced by the model, in this case, the confusion matrix (fig 15) reveals this model has a relatively high number of false positives and a low number of false negatives, this can explain why the model has high recall and low precision. Overall, this model's performance is mixed, with its strengths in ROC-AUC and recall but weaknesses in testing accuracy and f1 score.

Chart

Description automatically generated

**Figure 15.** Confusion matrix for KNN model

## 4.2 Logistic regression results

The LR model achieved a training accuracy of 86.1% and a testing accuracy of 83.3%, this testing accuracy was ranked joint second highest among the models evaluated, signifying good generalization performance. Additionally, the model achieved an f1 score of 82.1%, This was also second highest among the models assessed, signifying a good balance between precision and recall, furthermore this model attained an ROC-AUC score of 0.898. Overall, the logistic regression model performed well across all evaluation metrics, demonstrating that it is a consistent and robust model.

Chart

Description automatically generated

**Figure 16**. Confusion matrix for LR model

## 4.3 Random Forest results

The RF model achieved a training accuracy of 88.2% and a testing accuracy of 81.7%, this large discrepancy in training and testing accuracy suggests that this model was overfit. Additionally, this model obtained a subpar F1 score of 79.2%, placing it last among the examined models. This indicates that the model's precision and recall were inadequately balanced. Moreover, this model attained a below average ROC-AUC score of 0.897, The confusion matrix (fig 17) reveals this model obtained a relatively high number of false negatives contributing to its low recall. Overall, the RF classifier performed worst in terms of testing accuracy, f1 score and recall, additionally it also underperformed in AUC-ROC, further emphasising that this was one of the worst performing models.

Chart

Description automatically generated

**Figure 17.** Confusion matrix for RF model.

## 4.4 Adaboost results

The adaboost model achieved a training accuracy of 85.7% and testing accuracy of 83.3%, this testing accuracy was joint second, indicating that this model was able to generalise effectively to new data. Additionally, the model obtained a f1 score of 81.5%, ranking it in the median of the models examined. furthermore, the adaboost model achieved an ROC-AUC score of 0.872, this ROC-AUC score was the worst amongst evaluated algorithms. The confusion matrix reveals that the adaboost model obtained the same number of false positives and false negatives, hence providing a consistent balance between precision and recall. Overall, the adaboost classifier performed reasonable in terms of generalisability and f1 score, but poorly in terms of ROC-AUC score therefore yielding a below average performing model.

Chart

Description automatically generated

**Figure 18.** Confusion matrix for adaboost model.

## 4.5 XGBoost results

The XGBoost model achieved a training accuracy of 86.5% and a testing accuracy of 85%, this testing accuracy was the highest amongst all models evaluated, demonstrating the model’s exceptional ability to generalise to new data. Additionally, The XGBoost classifier achieved the highest f1 score of 83% and ROC-AUC score of 0.919, further demonstrating its superiority over the other models. Overall, the XGBoost model was the top performer across all germane evaluation metrics, confirming its place as the best performing model while also demonstrating its viability as a trustworthy and powerful algorithm for CVD prediction.

Chart

Description automatically generated

**Figure 19.** Confusion matrix for XGBoost model.

# 5: Evaluation and Discussion

This chapter summarises the project's findings and compares it with previous studies. Furthermore, this chapter addresses the project's limitations and suggest recommendations for future work. Finally, this chapter includes my personal reflections on this endeavour, as well as a succinct conclusion.

## 5.1 Findings and Discussion

In this project, five ML algorithms (KNN, RF, LR, adaboost, XGBoost) were trained, tuned and evaluated using the Cleveland heart disease dataset to predict CVD. The performance of each model was assessed using a variety of evaluation metrics with testing accuracy, f1 score and ROC-AUC score taking the most value. The results reveal all models performed well, achieving testing accuracy’s > 80%, in layman’s terms this means each algorithm was able to correctly predict an individual as either having or not having CVD > 80% of the time. The best performing model was found to be the XGBoost classifier attaining the highest scores in testing accuracy (85%), f1 score (83%) and ROC-AUC score (0.919). I believe the XGBoost model outperformed other models due to its ensemble approach with integrated optimisation strategies such as regularisation to prevent overfitting (Chen and Guestrin, 2016), this can be evidenced by the minor difference between its training and testing accuracy scores. The worst performing model was found to be the RF classifier achieving the lowest results in testing accuracy (81.7%) and f1 score (79.2%), I believe this was due to the RF classifier being overfit, as evidenced by the significant disparity between its training and testing accuracy, this was a surprising discovery given RF's notoriety for tolerance to overfitting.

Overall, these findings justify the use of ML as an alternative tool for CVD detection. All models exhibit a high level of predictive performance, with XGBoost ranking best. This bodes well for future research seeking to implement these models into clinical settings in order to enhance a physicians’ ability to diagnose CVD.

### 5.1.1 Comparison with previous studies

In this project, I attempted to address some of the research gaps identified by Azmi et al., (2022), by incorporating both FS and hyperparameter optimization techniques. The FS technique chosen was embedded FS with RF, this was due to many studies, particularly one by Chemmakha et al., (2022) demonstrating that it is an acceptable middle ground between filter and wrapper methods. Random search was chosen as the hyperparameter optimization approach as it was shown by Bergstra and Bengio (2012) to be just as good as grid search yet more time efficient.

The top performing algorithm in my project was the XGBoost classifier, this result is consistent with previously mentioned studies (Doki et al., 2022; Rajadevi et al., 2021; Budholiya et al., 2022) which also achieved optimal results with the XGBoost classifier. However, in contrast to these studies, my testing accuracy scores were slightly lower, with a maximum testing accuracy of 85%, I believe this was due to differences in data pre-processing, FS, as well as hyperparameter optimisation techniques utilised.

The KNN model was among the weakest performers in my project, while other studies (Shah et al., 2020; Hashi and Zaman., 2020) reported that KNN produced very accurate models with testing accuracy scores > 90%. I believe the poor performance of KNN in my project was caused by one-hot encoding, which I conducted as part of data pre-processing. One hot encoding is a technique where categorical features are encoded into multiple features of binary data. This increases the dimensionality of a dataset and since KNN is susceptible to the curse of dimensionality, it resulted in a model that performed worse.

Overall, the results of my project exhibited slightly worse performance than previous studies. I believe this was in part due to differences in the FS and hyperparameter optimisation techniques utilised. However, I suspect that the primary cause for the differences stemmed from the intrinsic variations in the data pre-processing techniques utilised.

## 5.2 Limitations

One limitation of my project was my dataset's relatively small sample size, this was a result of the limited availability of public large-scale CVD datasets due to privacy concerns. A larger dataset would allow the algorithms to capture more underlying patterns in the data and further improve generalisability. Furthermore, a larger dataset would have also enabled me to employ deep learning algorithms, this would have enabled me to produce models with superior performance.

Another limitation of my project was the lack of hybrid FS techniques utilised. Hybrid FS would have allowed me to combine the use of filter and wrapper methods, resulting in a more powerful selection of features, however this was unfeasible for me due to wrapper methods taking up significant computational costs.

## 5.3 Future work

The findings from this project adds to the growing corpus of research on the use of ML for CVD prediction, nevertheless there are still various avenues future work can take to enhance this topic area. One route for future work is the experimentation with Generative Adversarial Networks (GAN) to generate synthetic data, this can help tackle the privacy concern expressed by Dalal., (2020) over the use real patient data. Future work could also incorporate more advanced modelling techniques, such as ensemble modelling with stacking, this can lead to more robust models with improved performance. Additionally, future studies should refine and test these algorithms on larger and more diverse populations, allowing the models to be adopted globally rather than for a specific demographic. Finally, future work should concentrate on developing tools that can allow these models to be easily implemented in healthcare settings. This is important because the earlier they are implemented, the earlier CVD can be detected in patients allowing for the administration of preventative medication, avoiding premature mortality.

## 5.4 Reflection

Reflecting on this project I was able to hone my data analysis skills in python, this included data cleaning, visualisation and the construction of ML models. Furthermore, I was able to become accustomed with the intricacies of the Pandas, Seaborn and Scikit learn libraries. This endeavour taught me the value of data pre-processing, a significant amount of time was spent cleaning and transforming data in preparation for ML analysis, If I had been more diligent with data pre-processing, this time could have been better utilised by constructing and learning about other algorithms. Moreover, unnecessary time and effort was spent investigating and testing a wide range of hyperparameter combinations, with little progress in results. Instead, I could have chosen a few key hyperparameters and worked on tuning them. Finally, the vast quantity of sources on the subject of ML for CVD prediction made the literature review a daunting task, with countless hours spent reading through various journals. However, by staying organised and making concise notes I was able to find the relevant sources needed for the literature review.

## 5.5 Conclusion

CVD is a debilitating disease affecting all parts of the world, complications with current diagnostic methods such as speed and accessibility require the need for alternative approaches. ML offers an alternative approach which can allow for early detection facilitating prophylactic treatment, thereby preventing premature deaths. The primary objective of this project was to train, tune and evaluate several ML algorithms that will be able to accurately predict patients with CVD. More specific objectives included incorporating embedded feature selection in conjunction with hyperparameter optimisation. Furthermore, the final objective was to evaluate and compare the performances of each algorithm in order to identify which was the most effective. I believe these objectives were successfully met as I was able to develop and train five ML algorithms (KNN, RF, LR, adaboost, XGBoost) on the Cleveland heart disease dataset, additionally I was able to implement embedded feature selection using RF to remove irrelevant features as well as employ random search hyperparameter tuning to augment the performance of each algorithm. Finally, after evaluating each algorithm's performance I discovered that XGBoost was the best performing model, achieving highest scores across all relevant evaluation metrics with a distinguished testing accuracy of 85%. The findings of this project highlight the application of ML in accurately detecting patients with CVD, with the addition of FS and hyperparameter optimization approaches to further increase performance. Future research can make use of these findings and implement into clinical settings to assist physicians in diagnosing CVD earlier, providing additional time for protective treatment and mitigating premature mortality.

# 6: References

Alabdaljabar, M. S., Hasan, B., Noseworthy, P. A., Maalouf, J. F., Ammash, N. M. and Hashmi, S. K. (2023) Machine learning in cardiology: A potential real-world solution in low- and middle-income countries. *Journal of Multidisciplinary Healthcare*. 16, pp. 285–295.

Azmi, J., Arif, M., Nafis, M.T., Alam, M.A., Tanweer, S. and Wang, G. (2022) A systematic review on machine learning approaches for cardiovascular disease prediction using medical big data. *Medical Engineering and Physics. 105*

Belyadi, H. and Haghighat, A. (2021) *Machine Learning Guide for oil and gas using python: A step-by-step breakdown with data, algorithms, codes, and applications.* London: Gulf Professional Publishing

Bergstra, J. and Bengio, Y. (2012) Random Search for Hyper-Parameter Optimization*. J. Mach. Learn. Res.* 13, 281-305.

Bolin, E. and Lam, W. (2013) A review of sensitivity, specificity, and likelihood ratios: Evaluating the utility of the electrocardiogram as a screening tool in hypertrophic cardiomyopathy. *Congenital heart disease*. 8(5), pp. 406–410.

Bradley, A.P. (1997) The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*. 30(7), pp. 1145-1159.

Budholiya, K., Shrivastava, S.K. and Sharma, V. (2022) An optimized XGBoost based diagnostic system for effective prediction of heart disease. *Journal of King Saud University - Computer and Information Sciences*. 34(7), pp. 4514–4523.

Chandrashekar, G. and Sahin, F. (2014) A survey on feature selection methods. *Computers and Electrical Engineering*. 40(1), pp. 16-28.

Chemmakha, M., Habibi, O. and Lazaar, M. (2022) Improving machine learning models for malware detection using embedded feature selection method. *IFAC-PapersOnLine*. 55(12), pp. 771–776.

Chen, T. and Guestrin, C. (2016) XGBoost. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. Pp, 785–794

Dalal, K.R. (2020) Analysing the implementation of machine learning in Healthcare. *2020 International Conference on Electronics and Sustainable Communication Systems (ICESC)*.   
pp. 133-137

Doki, S., Devella, S., Tallam, S., Reddy Gangannagari, S.S., Sampathkrishna Reddy, P. and Reddy, G.P. (2022) Heart Disease Prediction Using XGBoost. *Proceedings of the 2022 3rd International Conference on Intelligent Computing, Instrumentation and Control Technologies: Computational Intelligence for Smart Systems, ICICICT 2022*. pp. 1317.

Géron, A. (2019). *Hands-on machine learning with ScikitLearn and TensorFlow*. Sebastopol, CA: O'Reilly.

Habehh, H. and Gohel, S. (2021) Machine learning in healthcare. *Current Genomics.* 22(4), 291-300.

Hashi, E.K. and Zaman, S.U. (2020) Developing a hyperparameter tuning based machine learning approach of heart disease prediction. *Journal of Applied Science and Process Engineering*. 7(2), pp. 631–647.

He, H. and Garcia, E.A. (2009) Learning from Imbalanced Data. *IEEE Transactions on Knowledge and Data Engineering*. 21(9), pp. 1263–1284.

Javaid, M., Haleem, A., Singh, RP., Suman, R. and Rab, S. (2022) Significance of machine learning in Healthcare: Features, pillars and applications. *International Journal of Intelligent Networks*. 3, pp. 58–73.

Khanna, N.N., Maindarkar, M.A., Viswanathan, V., Fernandes, J.F.E., Paul, S., Bhagawati, M., Ahluwalia, P., Ruzsa, Z., Sharma, A., Kolluri, R., Singh, I.M., Laird, J.R., Fatemi, M., Alizad, A., Saba, L., Agarwal, V., Sharma, A., Teji, J.S., Al-Maini, M., Rathore, V., Naidu, S., Liblik, K., Johri, A.M., Turk, M., Mohanty, L., Sobel, D.W., Miner, M., Viskovic, K., Tsoulfas, G., Protogerou, A.D., Kitas, G.D., Fouda, M.M., Chaturvedi, S., Kalra, M.K. and Suri, J.S. (2022) Economics of Artificial Intelligence in healthcare: Diagnosis vs. treatment. *Healthcare*. 10(12), pp. 2493.

Kohavi, R. (1995) A study of cross-validation and bootstrap for accuracy estimation and model selection*. In Proceedings of the international joint conference on artificial intelligence (IJCAI).* 1137–1143.

Kolukisa, B. and Bakir-Gungor, B. (2023) Ensemble feature selection and classification methods for machine learning-based coronary artery disease diagnosis*. Computer Standards and Interfaces*. 84

Kumar, Y., Koul, A., Singla, R. and Ijaz MF. (2022) Artificial Intelligence in disease diagnosis: A systematic literature review, synthesizing framework and future research agenda*. Journal of Ambient Intelligence and Humanized Computing*

Muhammad, Y., Tahir, M., Hayat, M. and Chong, K.T. (2020) Early and accurate detection and diagnosis of heart disease using intelligent computational model. *Scientific Reports*. 10(1)

National Health Service (2022a) *coronary heart disease*. Available from: <https://www.nhs.uk/conditions/coronary-heart-disease/> (Accessed 01 December 2022).

National Health Service (2022b) *Heart attack*. Available from:  
<https://www.nhs.uk/conditions/heart-attack/> (Accessed 01 December 2022).

National Health Service (2022c) *Stroke.* Available from:  
<https://www.nhs.uk/conditions/stroke/> (Accessed 02 December 2022).

National Health Service (2022d) *Cardiac catheterisation and coronary angiography*. Available from: <https://www.nhs.uk/conditions/coronary-angiography/> (Accessed 3 December 2022).

Nusinovici, S., Tham, Y.C., Chak Yan, M.Y., Wei Ting, D.S., Li, J., Sabanayagam, C., Wong, T.Y. and Cheng, C. (2020) Logistic regression was as good as machine learning for predicting major chronic diseases. *Journal of Clinical Epidemiology*. 122, pp. 56–69.

Pal, M. and Parija, S. (2021) Prediction of heart diseases using Random Forest. *Journal of Physics: Conference Series*. 1817(1), pp. 012009.

Qanitha, A., Uiterwaal, C.S.P.M., Henriques, J.P.S., Mappangara, I., Amir, M., Saing, S.G. and de Mol, B.A.J.M. (2019) Adherence to guideline recommendations for coronary angiography in a poor south-East Asian setting: Impact on short- and medium-term clinical outcomes. *Scientific Reports*. 9(1).

Rajadevi, R., Devi, E.M.R., Shanthakumari, R., Latha, R.S., Anitha, N. and Devipriya, R. (2021) Feature selection for predicting heart disease using black hole optimization algorithm and XGBoost classifier. *2021 International Conference on Computer Communication and Informatics (ICCCI).* pp. 1–7.

Randhawan, B., Jagtap, R., Bhilawade, A. and Chaure, D. (2022) Heart disease prediction using logistic regression algorithm. *International Journal for Research in Applied Science and Engineering Technology*. 10(4), pp. 2667–2672.

Raschka, S. and Mirjalili, V. (2019) *Python Machine Learning.* 3rd ed. Birmingham: Packt Publishing.

Samuel, A.L. (1959) Some studies in machine learning using the game of Checkers. *IBM Journal of Research and Development*. 3(3), pp. 210–229.

Senaviratna, N.A.M.R. and Cooray, T.M.J.A. (2019) Diagnosing multicollinearity of logistic regression model. *Asian Journal of Probability and Statistics*. 5(2), pp. 1–9.

Shah, D., Patel, S. and Bharti, S.K. (2020) Heart Disease Prediction using Machine Learning Techniques. *SN Computer Science*. 1(6).

Singh, Y.K., Sinha, N. and Singh, S.K. (2017) Heart disease prediction system using random forest. *Communications in Computer and Information Science*. 721, pp. 613–623.

Sultan Bin Habib, A. -., Tasnim, T. and Billah, M. M. (2019) A study on coronary disease prediction using boosting-based ensemble machine learning approaches. *ICIET 2019 - 2nd International Conference on Innovation in Engineering and Technology.*

Weng, S. F., Reps, J., Kai, J., Garibaldi, J. M. and Qureshi, N. (2017) Can machine-learning improve cardiovascular risk prediction using routine clinical data?. *Plos One.* 12(4).

World Health Organization (2022) *cardiovascular diseases (CVDs).* Available from: <https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)> (Accessed: 01 December 2022).

Zir, L.M., Miller, S.W., Dinsmore, R.E., Gilbert, J.P. and Harthorne, J.W. (1976) Interobserver variability in coronary angiography. *Circulation*. 53(4), pp. 627-632.

# 7: Appendix

## 7.1 Appendix A: Abbreviations

AUC - Area under curve

CVD - Cardiovascular disease

DT - Decision Tree

ECG - Electrocardiogram

EDA - Exploratory Data Analysis

ET - Extra Trees classifier

FN - False Negative

FP - False Positive

FS - Feature selection

GAN - Generative Adversarial Networks

GB - Gradient boost

KNN - K nearest neighbour

LR - Logistic regression

ML - Machine learning

NN - Neural Networks

RF - Random Forrest

ROC - Receiver Operating Characteristic

SMOTE - Synthetic Minority Oversampling Technique.

SVM - Support vector machines

TN - True Negative

TP - True Positive

XGBoost - Extreme Gradient Boost

## 7.2 Appendix B: Supervisory meetings

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Supervisor** | **Notes** | **Action** |
| 16-02-2023 | Kevin Golden | * Initial discussion on project topic and progress made so far. * Discussed whether I have completed the ethics approval form | * Complete ethics approval form * Finish write-up of background research. |
| 23-02-2023 | Kevin Golden | * Talked about aims and objectives. * Discussion on SMART Success criteria to set achievable goals. * Discussion of improving literature review by adding more sources about the application of ML in healthcare. * Discussed the possibility of reducing the number of ML models being built, due to similarities between them. | * Clarify aims and objectives. * Research more applications of ML in healthcare * Decrease the number of algorithms from 8 to 5, due to similarities between them. |
| 02-03-2023 | Kevin Golden | * Discussion on structure of literature review and critical analysis. * Discussed referencing sources in Harvard format, which can be found on the UWE website. | * Restructure literature review based on themes. * Evaluate strengths and weakness of each study. * Highlight gaps in research and propose ways to address with my project. * Rewrite bibliography in Harvard style. |
| 09-03-2023 | Kevin Golden | * Supervisor suggested attending espresso maths to help with statistical analysis of results. * Supervisor suggested comparing my results with studies mentioned in the literature review, highlighting the differences and providing possible explanations for why. * Discussed evaluating the project with limitations, future work and conclusions * The supervisor recommended going to the library to look at previous dissertations in order to grasp the structure and layout of the conventional dissertation. | * Attend espresso maths to help with results analysis. * Compare my results with previous studies. * Reorganise word document in conventional format. |

## 7.3 Appendix C: Gantt Chart

Chart

Description automatically generated

## 7.4 Appendix D: Supplementary Figures

Chart, line chart, scatter chart

Description automatically generated

**Figure 20.** Screenshot of hyperparameter k against model accuracy in KNN model. We can see as k increases accuracy decreases this is due to increase in Bias resulting in underfitting.

Diagram, Teams

Description automatically generated

**Figure 21.** Confusion matrices for each model

## 7.5 Appendix E: Code

All data cleaning, pre-processing, EDA and ML modelling were carried out in a Jupyter notebook using the Python programming language. The following link will take you to the repository where you can find the code.

**Code:** <https://gitlab.uwe.ac.uk/aa25-ahmed/research-project.git>

The repository contains two files: one for EDA and one for ML model building.

**Word count: 12905** (Excluding Title page, Abstract, Table of contents, References and Appendix)