KUMASI TECHNICAL UNIVERSITY

FACULTY OF APPLIED SCIENCES AND TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

PREDICTION OF HEART DISEASE USING TABULAR NEURAL NETWORK (TabNet)

SANNIE SOLOMON KWAKME PANYIN

052041350038

A PROJECT WORK SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE AWARD OF BACHELOR OF TECHNOLOGY IN COMPUTER TECHNOLOGY

SEPTEMBER, 2023.

DEDICATION

This research is dedicated to my family for their love encouragement and Support. To My Mum, Mrs. Helena Boateng and My Brothers for their unfeigned support prayers and advise. The trust and freedom you gave me made more responsible.

To my supervisor and Mentor Dr. Samuel King Opoku, My friends and Course mates for all your encouragements and support.

ACKNOWLEDGEMENT

Ebenezer this is how far the Lord has brought us, I thank the Almighty God for providing me with everything that I required in completing this project,

I have taking efforts in this project however it would not have been possible without the kind support and help of My friends and family I would like to extend my sincere thanks to all of them.

I am highly grateful to my supervisor, Dr. Samuel King Opoku for his guidance and constant supervision as well as for providing necessary information regarding the project and also for her support in completing the project.

I would like to express my special gratitude and thanks to UCI Repository for the dataset for this analysis .

My thanks and appreciation also goes to my siblings, friends and to the people who have willingly helped me out with their abilities.

Table of Contents

ABSTRACT

A primary cause of death globally is cardiac disease, which comprises several illnesses that affect the heart. In the US, heart disease is a factor in one out of every four fatalities. This indicates that around 610,000 individuals pass away from the illness each year. Early detection of heart disease makes treatment considerably simpler. Early detection can save lives and be greatly aided by machine learning.

The goal of this study is to create an AI-based system that can recognize individuals who have a higher risk of developing heart disease based on their medical history. For training and validation, the heart disease dataset from the UCI Machine Learning Repository was utilized. The findings of traditional classification methods including logistic regression, random forest, gradient boosting, and extreme gradient boosting were compared to those of the TabNet model. TabNet is an entirely novel deep learning architecture for tabular data that is reliable and understandable. The learning capacity of TabNet is concentrated on the most salient features, enabling interpretability and more effective learning. TabNet employs sequential attention to decide which features to draw conclusions from at each decision step.

Using ROC curves, accuracy, precision, sensitivity, specificity, and confusion matrices, promising findings were achieved and confirmed. With a ROC score of 0.94, 94% accuracy, and specificity and sensitivity over 0.93, the TabNet deep learning model beat the competition.

CHAPTER ONE

INTRODUCTION

This chapter focuses on the background of the study, the problem statement, the various objectives the study aims at achieving, the scope of the work and the significance of the project. It also gives a brief description of how the study has been organized.

Background of Study

Heart disease, also known as cardiovascular disease, refers to a group of conditions that affect the heart and blood vessels. These conditions include coronary artery disease, heart failure, arrhythmias, and heart valve problems, among others. Heart disease is the leading cause of death globally, accounting for an estimated 17.9 million deaths in 2019(World Health Organization, 2021).

The history of heart disease can be traced back to ancient times, with the first known description of heart disease dating back to ancient Egypt in 1550 BCE. In ancient times, heart disease was often attributed to supernatural causes or considered a natural consequence of aging. It was not until the 17th century that physicians began to recognize heart disease as a distinct medical condition. The Greek physician Hippocrates (About Heart Diseases, 2023) also described chest pain and palpitations as symptoms of heart disease. However, during the 17th century that the English physician William Harvey (1578-1657) discovered the function of the heart and the circulation of blood. In the 20th century, significant advances were made in the understanding and treatment of heart disease. In 1912, the American physician James Herrick first described the symptoms of coronary artery disease, which is caused by the build-up of plaque in the arteries that supply blood to the heart. In the 1950s, the first successful open-heart surgeries were performed, and in the 1960s, the first coronary artery bypass surgeries were performed.

Today, heart disease remains a significant public health challenge, with risk factors including smoking, high blood pressure, high cholesterol, diabetes, and a sedentary lifestyle. However, advances in prevention, diagnosis, and treatment have led to significant improvements in survival and quality of life for people with heart disease.

Heart disease being a leading cause of morbidity and mortality worldwide, accurate prediction and early diagnosis are crucial for effective management and prevention of this condition (Md Al Mehedi Hasan et al., 2021). In recent years, there has been growing interest in using machine learning algorithms to predict heart disease based on various clinical and non-clinical factors.

Machine learning is a type of artificial intelligence that enables computer algorithms to learn from data without being explicitly programmed (Ahsan & Siddique, 2022). The history of machine learning can be traced back to the early days of computing, but it wasn't until the advent of digital computers in the 1950s and 60s that machine learning began to take shape as a field of study. Some of the earliest work in machine learning was done by researchers like Arthur Samuel, who in 1959 developed a checker-playing program that could learn from its own experience and improve its gameplay over time.

In the 1990s and 2000s, machine learning began to be applied to a wide range of practical problems, including image and speech recognition, natural language processing, and predictive models. This was made possible by advances in computing power and the availability of large amounts of data. One important development during this period was the emergence of the support vector machine (SVM), a powerful new algorithm for classifying data that was developed by Vladimir Vapnik and his colleagues (Vladimir Vapnik, 2013).

In the context of heart disease prediction, machine learning algorithms can analyze large datasets of clinical and non-clinical variables, such as demographics, medical history, laboratory tests, imaging data, and lifestyle factors, to develop predictive models that can identify individuals at high risk of developing cardiovascular disease (Dutta et al., 2020).

However, these algorithms are not always accurate and may result in false positives or false negatives.

Tabular Neural Networks (TNNs) are a type of deep learning algorithm and a neural network architecture that is specifically designed for tabular data. Tabular data refers to data that is organized in tables or spreadsheets, with rows representing instances of data and columns representing features or attributes.

TNNs can be traced back to the early days of neural networks in the 1980s and 1990s. At that time, neural networks were primarily used for image and speech recognition tasks and were not well-suited for handling tabular data.

However, with the advent of deep learning in the 2010s, neural networks began to be used for a wider range of tasks, including tabular data analysis. In 2015, the Google Brain team introduced TensorFlow, a popular open-source software library for building and training neural networks, which helped to popularize deep learning approaches to tabular data analysis.
Therefore, this study aims to develop a TNN model for the early prediction of heart disease with the goal of improving accuracy and reducing false diagnoses.

Statement of the Problem

"Prediction of Cardiovascular Disease Risk Using Tabular Neural Networks" by F. (Cheng et al., 2020). In this study, the authors used a tabular neural network (TNN) to predict the risk of cardiovascular disease. However, the limitation of this study is that the dataset used was limited to a specific population in China, which may limit the generalizability of the results to other populations.
Zhang et al (Zhang et al., 2020) research employed a machine learning algorithm called the Random Survival Forest model to forecast the risk of mortality in patients with heart failure based on a variety of clinical variables, such as demographics, comorbidities, and laboratory values. According to the research, the ML algorithm had a high concordance index (C-index) of 0.77 and could correctly predict mortality risk but had limitations of specific dataset.

"Predicting the Risk of Cardiovascular Disease Using Tabular Neural Networks: A Case Study in Ghana" by P. O. Gyamfi et al(Gyamfi et al., 2021). In this study, the authors used a TNN to predict the risk of cardiovascular disease in Ghana. However, the limitation of this study is that the dataset used was limited to a specific population in Ghana, which may limit the generalizability of the results to other populations.

"Using Explainable Tabular Neural Networks to Predict Cardiovascular Disease" (Ali & Khang, 2022) This paper proposes the use of an explainable tabular neural network (ETNN) for predicting cardiovascular disease. The study used the Framingham Heart Study dataset and achieved an accuracy of 87.5%. The study also included an analysis of feature importance and visualization of the decision-making process. The limitations of the study include the use of a single dataset and the lack of comparative analysis with other machine learning algorithms.

Objectives

The objectives of this proposed project are:

- To develop a tabular neural network for the prediction of heart disease
- To evaluate the accuracy of the tabular neural network in comparison to existing methods
- To identify the most important features for predicting heart disease using feature importance analysis

Scope of the Study

This study will focus on the use of a tabular neural network for predicting heart disease using the publicly available Cleveland heart disease dataset. The study will evaluate the performance of the tabular neural network in comparison to existing methods, including logistic regression, random forests, XGBoost and Gradient Boost. The study will be limited to the use of this dataset and will not cover other datasets or methods.

Significance of Study

The use of a tabular neural network for the prediction of heart disease has the potential to improve accuracy and reduce false positives or false negatives. This could lead to more accurate diagnoses and improved treatment outcomes for patients. The results of this study could contribute to the development of more accurate and reliable machine learning algorithms for predicting heart disease.

Organization of the Work

The first chapter focused on the background of the study, the problem statement, the various objectives the study aims at achieving, the scope of the work and the significance of the project. It also gives a brief description of how the study has been organized.

The second chapter concerns itself with what other researchers have done concerning the topic under study. It reveals theories and concepts that have been generated. It focuses on the various technologies employed in the area under study.

The third chapter, on the other hand, focuses on the design of the proposed system. It designs how the components of the system interact to carry out the system functions. The methodology employed in this work is discussed in detail.

The next chapter, mainly chapter four, implements the various algorithms generated in the third chapter. It ensures that all the objectives of the proposed system are achieved through implementation.

The last chapter, mainly chapter five, ends the study with summary and discussion of the various findings. It also focuses on reviewing the strengths and limitations of the implemented system and finally provides direction into the future by stating research areas associated with the study

CHAPTER TWO

LITERATURE REVIEW

Introduction

Heart diseases encompass a range of Cardiovascular conditions that significantly impact global health. Accurate prediction and early detection of these diseases are crucial for timely interventions, improved patient outcomes, and effective resource allocation within healthcare systems. Traditional methods for predicting heart diseases have relied on statistical models and clinical risk scores. However, these approaches often face limitations in handling complex relationships and capturing nonlinear interactions among diverse risk factors. In recent years, there has been a surge of interest in utilizing advanced machine learning techniques to enhance the prediction accuracy and reliability of heart disease outcomes. Among these techniques, tabular neural networks have emerged as a promising approach due to their capability to handle structured data and capture intricate patterns.

Coronary artery, rheumatic heart, vascular, and other heart and blood vessel issues are among the cardiovascular diseases (CVDs). Strokes or heart attacks are to blame for four out of every five CVD fatalities. One-third of all deaths involve people under the age of 70[1] . The main risk factors for heart disease are sex, smoking, age, family history, poor diet, cholesterol, physical inactivity, high blood pressure, obesity, and alcohol consumption. Hereditary risk factors for heart disease include diabetes and high blood pressure[2]. Physical inactivity, obesity, and an unhealthy diet are a few of the secondary factors that raise the risk. The much more prevalent symptoms include generalized weakness, breathlessness, fatigue, palpitations, perspiration, back pain, chest discomfort, shoulder pain, and arm pain.

Artificial Intelligence (AI) is a stream of science related to intelligent machine learning, mainly intelligent computer programs, which provide results in a similar way to human attention process[3]. This process generally comprises obtaining data, developing efficient systems for the uses of obtained data, illustrating definite or approximate conclusions and self-corrections/adjustments. In general, AI is used for analyzing machine learning to imitate the cognitive tasks of individuals. AI technology is exercised to perform more accurate analyses as well as to attain useful interpretation. In this perspective, various useful models as well as

computational intelligence are combined in the AI technology[4]. The progress and innovation of AI applications are often associated with the fear of unemployment threat. However, almost all advancements in the applications of AI technology are being celebrated on account of the confidence which enormously contributes its efficacy to the industry[4], [5].

Computers can now learn from data and predict the future without explicit programming thanks to the quickly developing discipline of Machine Learning (ML)[6]. In recent years, the healthcare sector has made significant use of machine learning (ML) to evaluate sizable datasets and forecast results like illness progression, treatment reaction, and mortality rates. Since it has the ability to enhance patient results and healthcare delivery, the implementation of machine learning (ML) in heart failure prognosis has drawn a lot of interest from academics and doctors.

The purpose of this research is to use ML techniques to predict heart failure mortality rates based on patient data. The study aims to identify the key predictors of mortality and develop a predictive model that can accurately predict the likelihood of death for heart failure patients. The predictive model could assist clinicians in identifying high-risk patients and providing appropriate treatment and care to improve outcomes.

Deep Learning (DL): DL is a subfield of ML that utilizes artificial neural networks with multiple layers to extract high-level features and learn complex representations from raw data[7]. DL algorithms are particularly effective for processing unstructured data, such as images, text, and audio. Deep neural networks can automatically learn hierarchical representations, leading to state-of-the-art performance in various tasks, including image recognition, natural language processing, and speech recognition.

Tabular Neural Network (TNN): TNN is a specific type of neural network architecture designed to handle structured or tabular data[7], [8]. It applies neural network models to analyze and make predictions based on data organized in rows and columns, such as spreadsheets or databases. TNNs can capture complex relationships and patterns in tabular data, allowing for accurate predictions in tasks like classification, regression, and forecasting[8].

The relationship between AI, ML, DL, and TNN can be understood as follows: TNN is a specific application of deep learning techniques within the field of machine learning. TNNs leverage the power of deep neural networks to process structured or tabular data, enabling accurate predictions in various domains. TNNs are a subset of ML techniques, which, in turn, fall under the broader umbrella of AI.

It's important to note that AI encompasses a broader range of techniques beyond ML and DL, including symbolic reasoning, expert systems, and knowledge representation. However, ML and DL have gained significant attention in recent years due to their ability to handle complex patterns and make accurate predictions, making them particularly relevant in the field of AI. TNNs specifically focus on applying deep learning principles to structured or tabular data, offering powerful predictive modeling capabilities.

Tabular neural networks offer a promising alternative, leveraging the power of deep learning to analyze structured data. By constructing intricate neural architectures, these networks can learn from large datasets and automatically extract relevant features to make accurate predictions. Tabular neural networks have shown great potential in various domains, including natural language processing, computer vision, and now, healthcare[9].

Tabular neural networks have gained significant popularity in recent years due to their effectiveness in analyzing structured data and solving various prediction tasks. However, it is important to note that the concept of neural networks has a longer history, dating back several decades[7], [8].

Neural networks, inspired by the structure and functioning of the human brain, were initially proposed in the 1940s. The first model, called the perceptron, was developed by Frank Rosenblatt in 1958. Perceptron was a single-layer network capable of binary classification tasks, but their limited capabilities and the lack of efficient training algorithms limited their practical applications[10].

In the 1980s, neural network research experienced a resurgence with the introduction of backpropagation, a training algorithm that allowed for efficient learning in multi-layer neural networks. This breakthrough enabled the development of more complex and powerful models capable of solving a wider range of tasks[11].

The application of neural networks to structured or tabular data gained momentum with the advancement of deep learning techniques[10], [12]. Traditional neural networks were primarily applied to unstructured data such as images, speech, and text. However, researchers recognized the need to extend neural networks' capabilities to structured data, such as spreadsheets or databases, to leverage their potential for prediction tasks in various domains.

In recent years, the field of tabular neural networks has witnessed significant developments. Researchers have explored various architectures and techniques specifically designed to handle

structured data. These networks employ multiple layers and advanced activation functions to process the input features, extract relevant patterns, and make accurate predictions[11].

The concept of attention mechanisms has also played a crucial role in enhancing the performance of tabular neural networks. Attention mechanisms allow the network to focus on important features or interactions within the structured data, enabling more accurate predictions and providing insights into the underlying relationships.

Furthermore, advancements in hardware and computational capabilities, such as the availability of powerful GPUs and distributed computing, have facilitated the training and deployment of larger and more complex tabular neural network models. These developments have led to significant improvements in prediction accuracy and have expanded the application of tabular neural networks across various domains, including healthcare, finance, and customer analytics[13].

Overall, the history of tabular neural networks can be traced back to the early developments of neural networks and their subsequent evolution with the advancements in deep learning techniques. The focus on structured data and the development of architectures and techniques tailored for tabular data have made tabular neural networks a powerful tool for accurate prediction in diverse domains[11], [14].

The theoretical framework of tabular neural networks encompasses a range of architectures, including feedforward networks, deep neural networks, and recurrent neural networks[11], [14]. Recent advancements, such as attention mechanisms, transformer-based architectures, and graph neural networks, have further extended their capabilities for handling structured data. These advancements enable the extraction of complex patterns, interaction modelling, and enhanced performance in predicting heart diseases.

This chapter presents a comprehensive literature review focused on the prediction of heart diseases using tabular neural networks, aiming to explore the current state of research, identify advancements, and uncover potential research gaps.

Overview of Machine Learning

The study of methods and models that enable computers to learn from data and make predictions or judgments is the focus of the discipline of artificial intelligence known as Machine Learning (ML)[15]. Due to its capacity to extract insights, identify patterns, and automate challenging processes, ML has attracted a lot of attention recently. The fundamental ideas, several kinds of

algorithms, and practical applications of machine learning are all covered in this paper's overview[16].

Machine Learning Basics

Machine learning aims to develop and learn algorithms over time by training them on data. The following are significant machine learning components:

- Data**:** The basis of machine learning is data. It might be unstructured or structured (for example, tabular data) (e.g., text, images)[15]. The effectiveness of ML models depends heavily on the type, volume, and representativeness of the data.
- Features: The measurable aspects or properties of the data are represented by features. Accurate predictions and effective learning depend on choosing relevant features. The process of feature engineering entails turning unstructured data into useful features.
- Labels: In supervised learning, labels—also referred to as objectives or outputs—are the anticipated outcomes or predictions linked to the input data[17]. They serve as the foundation for teaching ML models to make forecasts based on fresh, untainted data.
- Training and Testing: On a labeled dataset, where the input data and associated labels are known, machine learning models are trained. Between the input and output, the model discovers patterns and relationships. In testing, the model's performance on hypothetical data is evaluated in order to determine its precision and generalizability[17].

Types of Machine Learning Algorithms

Machine learning algorithms can be broadly categorized into three types being Supervised Learning, Unsupervised Learning and, Reinforcement Learning.

Supervised Learning

ML models are trained by supervised learning utilizing labeled datasets, where each instance of data is linked to a predetermined result[18]. Input data are mapped by the models to the appropriate output labels. Linear regression, logistic regression, support vector machines (SVM), decision trees, and neural networks are a few examples of supervised learning techniques. It is a widely adopted approach in various domains due to its ability to learn from historical data and generalize to new instances[15]. This paper provides an overview of supervised learning, including its fundamental concepts, techniques, evaluation metrics, and real-world applications.

Supervised learning involves training ML models using labeled datasets, where each data instance is associated with a known output or label. The following are key components of supervised learning:

- Input Features: Input features, also known as independent variables or predictors, represent the measurable properties or characteristics of the data. They serve as the basis for the model to make predictions or classifications[19]. The quality and relevance of the features greatly influence the model's performance.
- Output Labels: Output labels, also known as dependent variables or targets, are the desired predictions or outcomes associated with the input data. They represent the expected values that the model should predict or classify accurately. The quality and representativeness of the labels are crucial for training effective models.
- Training Data: Training data consists of a set of labeled instances used to train the supervised learning model. It comprises input features and their corresponding output labels. The model learns patterns and relationships between the input features and output labels during the training process[13].
- Model Prediction or Classification: Once the model is trained, it can make predictions or classifications on new, unseen data by applying the learned patterns and relationships. The model maps the input features to the predicted or classified output based on the patterns it has learned from the training data[13].

Techniques in Supervised Learning

Supervised learning encompasses various techniques and algorithms that can be used to train models on labeled datasets. Some commonly used techniques include:

Linear Regression

A fundamental and popular statistical method for simulating the relationship between a dependent variable and one or more independent variables is linear regression. It looks for the linear equation that best captures the correlation between the variables. Numerous disciplines, such as economics, finance, social sciences, and machine learning, frequently use linear regression[20]. The goal of the regression method known as linear regression is to simulate the linear connection between input data and output labels. It makes predictions about continuous numerical quantities by estimating the coefficients of a linear equation[21].

Simple Linear Regression

Simple linear regression focuses on modeling the relationship between a single independent variable (predictor) and a dependent variable (response)[21]. The equation for simple linear regression is represented as:

$$\boldsymbol{y = \beta_0 + \beta_1 x + \varepsilon}$$

Where:

y is the dependent variable.

x is the independent variable.

$\boldsymbol{\beta_0}$ is the y-intercept (the value of y when x is zero).

$\boldsymbol{\beta_1}$ is the slope of the line (represents the change in y for each unit change in x).

ε is the error term that accounts for the variability not explained by the linear relationship.

Simple linear regression attempts to quantify the difference between the observed and predicted values of the dependent variable by estimating the values of $\boldsymbol{\beta_0}$ and $\boldsymbol{\beta_1}$ that minimize the sum of squared residuals[9], [22].

Multiple Linear Regression

Multiple linear regression extends the concept of simple linear regression to include multiple independent variables. It models the relationship between the dependent variable and multiple predictors simultaneously[9]. The equation for multiple linear regression is represented as:

$$\boldsymbol{y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + \varepsilon}$$

Where:

y is the dependent variable.

$\boldsymbol{x_1, x_2, \ldots, x_p}$ are the independent variables.

β_0 is the y-intercept.

$\boldsymbol{\beta_1, \beta_2, \ldots, \beta_p}$ are the coefficients representing the impact of each independent variable on the dependent variable.

ε is the error term.

The coefficients (**$\beta_0, \beta_1, ..., \beta_p$**) are estimated using techniques such as ordinary least squares (OLS) or gradient descent to minimize the sum of squared residuals.

Assumptions of Linear Regression

Linear regression relies on several assumptions to ensure accurate and reliable results. These assumptions include:

- Linearity: The relationship between the independent variables and the dependent variable is assumed to be linear[23]. If the relationship is nonlinear, alternative regression models may be more appropriate.
- Independence: The observations or data points used in linear regression should be independent of each other. Autocorrelation or dependence between observations may lead to biased and inefficient estimates[24].
- Homoscedasticity: The variance of the error term (ε) should be constant across all levels of the independent variables. Homoscedasticity ensures that the residuals are evenly distributed and do not exhibit a systematic pattern[23].
- Normality: The error term (ε) is assumed to follow a normal distribution. This assumption allows for reliable inference and hypothesis testing[21].

Evaluation and Interpretation

Linear regression models are evaluated using various metrics and techniques:

- Coefficient Estimates: The coefficients ($\beta_0, \beta_1, ..., \beta_p$) in linear regression represent the impact of each independent variable on the dependent variable. Positive coefficients indicate a positive relationship, while negative coefficients indicate a negative relationship[25]. The magnitude of the coefficients reflects the strength of the relationship.
- R-squared (R^2): R-squared measures the proportion of variance in the dependent variable explained by the linear regression model. It ranges from 0 to 1, with higher values indicating a better fit[23]. However, R-squared alone may not provide a complete picture, and other metrics should be considered.
- Residual Analysis: This technique examines the differences between the observed and predicted values of the dependent variable[23]. Residual plots can help assess the

assumptions of linearity, homoscedasticity, and normality. Patterns or deviations in the residuals may indicate violations of these assumptions.

- Significance Testing: Hypothesis tests, such as t-tests or F-tests, can assess the significance of the coefficients and overall model fit. These tests help determine if the relationships between the independent variables and the dependent variable are statistically significant[26].

A popular statistical method for simulating the relationship between variables is linear regression. It allows prediction or estimation and sheds light on how independent variables affect the dependent variable. Researchers and practitioners can successfully use linear regression in a variety of domains by grasping the underlying ideas, presumptions, and evaluation strategies.

Logistic Regression

Logistic regression is the form of statistical model that is used to for predictive analysis and it is used for classification it estimates the chances of an occurring an event based on independent variable on the given data set since the output of a probability between the dependent variable leaps between 1 and 0. In this regression the odds are applied from the logit transformation that is the chances of success/chances of failure. It is known as log odds[26]. It is widely employed in various fields, such as medicine, social sciences, and machine learning, when the outcome of interest is binary (e.g., yes/no, true/false, 0/1).

Binary Logistic Regression

This type of logistic regression focuses on modeling the relationship between a binary dependent variable (also called the response or outcome variable) and one or more independent variables (also known as predictors or explanatory variables). The dependent variable takes on two categories, typically coded as 0 and 1.

The logistic regression model uses the logistic function (also called the sigmoid function) to model the probability of the binary outcome[27]. The equation for binary logistic regression is represented as:

$$\boldsymbol{logit(p) = \beta^0 + \beta^1 x^1 + \beta^2 x^2 + \ldots + \beta_p x_p}$$

Where:

logit(p) represents the log-odds or the natural logarithm of the odds ratio of the probability p.

p represents the probability of the binary outcome.

$x_1, x_2, ..., x_p$ are the independent variables.

$\beta_0, \beta_1, ..., \beta_p$ are the coefficients that represent the impact of each independent variable on the log-odds or probability.

The coefficients ($\beta_0, \beta_1, ..., \beta_p$) are estimated using techniques such as maximum likelihood estimation (MLE) or gradient descent to find the values that maximize the likelihood of the observed data.

Multinomial Logistic Regression

Multinomial logistic regression extends the binary logistic regression to handle categorically dependent variables with more than two categories. It models the relationship between a categorical dependent variable and one or more independent variables[22].

There are various sets of coefficients in the equation for multinomial logistic regression, each of which corresponds to a different category. The multinomial logistic function is used to model the likelihood of each category.

Decision Trees

Decision trees are versatile models that use a hierarchical structure of nodes and branches to make predictions or classifications[18], [19]. They learn rules from the training data and form a tree-like structure, where each internal node represents a decision based on a feature, and each leaf node represents an output label. Each internal node of the tree represents a decision based on a feature, while each leaf node represents a prediction or a class label[11].

Structure of a Decision Tree

- A decision tree consists of nodes and edges. The nodes are divided into two types, namely root node and internal node[28].
- Root Node: The decision tree's root node serves as its origin. It relates to the characteristic that divides the data most effectively and reflects the complete dataset.

- Internal Node: Internal nodes represent decisions based on features. They split the dataset into smaller subsets based on different feature values[13]. Each internal node has outgoing edges corresponding to different feature values and leads to other internal nodes or leaf nodes.
- Leaf Nodes: The ultimate predictions or class labels are represented by leaf nodes, often referred to as terminal nodes[22]. Each leaf node is associated with a particular prediction or class.

Splitting Criteria

At each internal node of the decision tree, the best feature and its related splitting point are chosen. Typical splitting standards include the following.

- Gini Impurity: Gini impurity quantifies the likelihood that an element would be wrongly classified if it were randomly labeled in accordance with the distribution of classes in the node. A better split is indicated by a lower Gini impurity[29].
- Information Gain: By dividing the data based on a certain attribute, information gain quantifies the decrease in entropy or disorder that results. To provide the most informative splits, it chooses the characteristic that optimizes information gain.
- Gain Ratio: The number of branches that come from a split is considered by the gain ratio, an extension of information gain. Features with several categories or values are penalized[29], [30].

Tree Pruning

Decision trees can be prone to overfitting, where the model becomes too complex and performs poorly on unseen data. To address this, pruning techniques are applied:

- Pre-Pruning: Pre-pruning involves stopping the tree construction process early based on certain conditions. It can limit the maximum depth of the tree, the minimum number of samples required to split a node, or the maximum number of leaf nodes allowed.
- Post-Pruning: Growing the decision tree to its maximum size and then eliminating or collapsing nodes depending on their impurity or mistake rates constitutes the process known as post-pruning, also known as tree pruning or cost-complexity pruning[23], [26]. Trees that have been pruned are simpler and less likely to overfit[26].

Advantages of Decision Trees

Decision trees in machine learning provide the following benefits:

- Interpretability: A simple and understandable illustration of the decision-making process is provided by decision trees[31]. The tree structure makes it simple to visualize and comprehend the underlying reasoning.
- Nonlinear Relationships: Modeling nonlinear connections between characteristics and the goal variable using decision trees is possible. With minimal data preparation, they can handle both continuous and categorical variables.
- Feature Importance: Decision trees can provide insights into feature importance. By examining the splits and the order of features, one can understand which features have the most significant impact on the predictions or classifications[28].
- Robustness to Outliers and Missing Data: Decision trees are resistant to anomalies and missing data values[11], [29]. By designating the majority class or the average value of the available samples, they can deal with missing data.

Limitations of Decision Trees

Although decision trees have benefits, they also have certain drawbacks:

- Overfitting: Overfitting is a problem with the decision trees, especially when the tree is too deep or complicated. Techniques for pruning and hyperparameter adjustment can lessen this problem.
- Instability: Small adjustments to the training set can have a big impact on the decision tree's structure. Different splits and projections may come from this instability[13].
- Lack of Smoothness: Decision boundaries in decision trees frequently have sharp, orthogonal edges, which might miss nuanced interactions or slow changes between features[11].
- Difficulty in Capturing XOR or Parity Problems: XOR or parity issues, in which the link between the characteristics and the target variable cannot be distinguished by straightforward threshold rules, are difficult for decision trees to handle.

Support Vector Machines (SVM)

SVM is a potent supervised learning technique that may be utilized for both regression and classification applications. It seeks to identify an ideal hyperplane that divides various classes or

forecasts continuous values[13], [19]. SVM works by mapping the data into a higher-dimensional feature space and maximizing the margin between classes.

Basic Concepts

SVMs are based on the concept of finding the best decision boundary between classes. The key concepts in SVMs include:

- Hyperplane: In the feature space, a hyperplane is a decision boundary that divides classes[32]. A line in 2D space or a hyperplane in higher dimensions are both considered hyperplanes in binary classification problems. The margin—the distance between the hyperplane and the closest data points for each class—is what SVMs seek to optimize.
- Support Vectors: The data points that are closest to the decision boundary or hyperplane are known as support vectors. The hyperplane's location and orientation are most strongly influenced by these points[22].
- Kernel Trick: Using the kernel approach, the initial feature space may be changed into a higher-dimensional feature space. By projecting the data into a higher-dimensional space where it is linearly separable, it enables SVMs to locate nonlinear decision boundaries in the original feature space.

SVM for Classification

SVMs can be used for binary and multiclass classification problems. The steps involved in SVM classification are as follows:

- Feature Representation: Represent the input data using suitable features. SVMs work with numerical features, so categorical or text data needs to be transformed appropriately.
- Margin Maximization: By measuring the distance between the hyperplane and the closest data points for each class, SVMs seek to identify the hyperplane that maximizes margin. By resolving an optimization issue that reduces the classification error while maximizes the margin, the hyperplane is discovered[29], [32].
- Kernel Selection: To convert the data into a higher-dimensional feature space, pick the suitable kernel function. The linear, polynomial, radial basis function (RBF), and sigmoid kernels are frequently utilized kernel functions. The task at hand and the data characteristics determines the kernel to use[33].

- Regularization and Hyperparameter Tuning: The trade-off between attaining a higher margin and reducing classification mistakes is controlled by regularization factors like the penalty parameter C. Using methods like cross-validation, hyperparameter tuning includes choosing the best values for these parameters[32], [33].

SVM for Regression

- Regression tasks, where the objective is to predict continuous values, may also be performed using SVMs. The model in SVM regression seeks to fit as many data points as possible within an established tolerance or error margin, referred to as the epsilon tube.
- Epsilon Insensitive Loss Function: The epsilon insensitive loss function used by SVM regression allows for some error tolerance within a predetermined range (epsilon)[32]. Data points inside this range are regarded as well-expected, but those outside are punished according to how far they deviate from the value that was predicted[29].
- Kernel Selection and Hyperparameter Tuning: Like SVM classification, selecting an appropriate kernel function and tuning hyperparameters like the penalty parameter C and epsilon are crucial for SVM regression.

Advantages of SVMs

Support Vector Machines offer several advantages in machine learning:

- Effective in High-Dimensional Spaces: SVMs perform well even in high-dimensional feature spaces, making them suitable for complex datasets with many features.
- Robust to Outliers: SVMs are robust to outliers as they focus on the support vectors, which are the data points closest to the decision boundary.
- Versatility: SVMs can handle both linearly separable and nonlinearly separable data by using different kernel functions.
- Good Generalization: SVMs aim to find the hyperplane with the maximum margin, which helps in generalizing well to unseen data.

Limitations of SVMs

While SVMs offer several advantages, they also have certain limitations:

- Computational Complexity: The training time and memory requirements of SVMs can increase significantly with large datasets[32], especially when using nonlinear kernels or in the case of multiclass problems.
- Sensitivity to Hyperparameters: The performance of SVMs can be sensitive to the selection of hyperparameters, such as the kernel type, regularization parameter, and kernel parameters. Careful tuning is required to achieve optimal results.
- Interpretability: SVMs provide good predictive accuracy, but the resulting models may not be easily interpretable compared to decision trees or linear regression.

Neural Networks

Neural networks, also known as artificial neural networks or deep learning models, are a class of machine learning models inspired by the structure and functioning of the human brain. They are designed to mimic the behavior of neurons and can learn from data to perform complex tasks, such as pattern recognition, classification, regression, and decision-making.

Neural networks consist of interconnected nodes, called artificial neurons or units, organized in layers. The three main layers of neural network are:

Input Layer: Input layer refers to the first layer of nodes in an artificial neural network. This layer receives data from the outside world[34]. The input layer receives the initial input data, which could be numerical values, images, text, or any other form of data.

Hidden Layers: Between the input and output layers are hidden layers that process the input data via weighted connections. Depending on how difficult the problem is, a different number of hidden layers and neurons may be present[12].

Output Layer: The final predictions or classifications are produced by the output layer using the calculations done in the hidden layers. The output layer's neuron count varies depending on whether regression, binary classification, or multi-class classification is being performed[10].

Training is the process through which neural networks learn from data. To reduce the discrepancy between expected and actual outputs, the network modifies the weights and biases of the connections between neurons during training. Backpropagation, which computes gradients and changes weights and biases using gradient descent optimization, is the most widely used approach for training neural networks.

One of the key features of neural networks is their ability to learn hierarchical representations of data[10], [13]. Through multiple layers of computation, neural networks can extract and learn complex features and relationships from raw input data. This capability has led to significant advancements in domains such as computer vision, natural language processing, speech recognition, and recommendation systems.

Types of Neural Network Architectures

Feedforward Neural Networks (FNN): The simplest type of neural network[35], where information flows in one direction, from the input layer through the hidden layers to the output layer. FNNs are widely used for various tasks, including classification and regression.

Convolutional Neural Networks (CNN): CNNs are primarily used for image and video analysis. They leverage convolutional layers that apply filters to extract local features from images, enabling the network to learn hierarchical representations[29]. CNNs have achieved state-of-the-art performance in tasks like object detection, image classification, and image segmentation.

Recurrent Neural Networks (RNN): RNNs are designed to handle sequential data, such as time series or natural language processing tasks[7], [8]. They utilize recurrent connections, enabling information to flow in cycles within the network. RNNs have a memory-like capability, making them effective for tasks like language modeling, speech recognition, and machine translation.

Long Short-Term Memory (LSTM) Networks: LSTMs are a type of RNN that addresses the vanishing gradient problem, enabling the network to capture long-range dependencies in sequential data. LSTMs are commonly used in tasks that involve sequential input with long-term dependencies, such as speech recognition, text generation, and sentiment analysis.

Tabular Neural Networks (TNNs)

Table-based neural networks, or TNNs for short, are neural network models built particularly to handle tabular or structured data, where the input features are organized in a table with rows and columns[8]. TNNs are often used for tasks like regression and classification on structured datasets, where each column signifies a feature or characteristic and each row denotes an instance or sample.

Here are some key aspects of Tabular Neural Networks:

Input Representation: The input characteristics to TNNs are frequently expressed as numerical values[7], [10]. Before feeding categorical data into the network, they may need to be preprocessed and transformed into numerical representations, such as one-hot encoding or ordinal encoding.

Architecture: An input layer, one or more hidden layers, and an output layer are the usual building blocks of TNNs. Depending on the difficulty of the task and the quantity of the dataset, the number of hidden layers and the number of neurons in each layer may change. The weighted total of the inputs is applied by each neuron in the hidden layers as an activation function[36].

Feature Embeddings: With tabular data, categorical variables may be handled by TNNs by using feature embeddings. Low-dimensional representations of categorical data called feature embeddings reveal the underlying connections and similarities between them. When working with categorical variables that have a high cardinality[10], embeddings can enhance performance by assisting the network in learning more expressive representations of the data.

Learning and Optimization: TNNs learn from data by iteratively adjusting the weights and biases in the network to minimize a loss function. Popular optimization algorithms, such as stochastic gradient descent (SGD), Adam, or RMSprop, are commonly used to update the network parameters. The choice of loss function depends on the task, such as Mean Squared Error (MSE) for regression or cross-entropy loss for classification[37].

Regularization: To prevent overfitting and improve generalization, TNNs can incorporate various regularization techniques. Regularization methods, such as L1 and L2 regularization (weight decay), dropout, and early stopping, are employed to control the complexity of the model and reduce the impact of noise or irrelevant features in the data[37], [38].

Model Evaluation: The performance of TNNs is evaluated using suitable metrics based on the specific task. For regression tasks, metrics like mean absolute error (MAE) or root mean squared error (RMSE) are commonly used[33]. For classification tasks, metrics like accuracy, precision, recall, and F1 score are employed.

Interpretability: Given their complexity and extreme nonlinearity, TNNs can be difficult to interpret. However, methods like feature significance analysis, attention processes, and layer visualization can provide light on the learnt representations and how certain aspects affect the predictions made by the model[29].

Handling Tabular Data Characteristics: TNNs can handle various tabular data characteristics, such as missing values, outliers, and feature engineering. Preprocessing techniques, such as imputation for missing values and normalization or standardization for feature scaling, are applied to prepare the data for training the network[38].

TNNs have proven to perform exceptionally well with structured data and have been used in several industries, including banking, healthcare, consumer analytics, and fraud detection[22]. When employing TNNs for tabular data tasks, it is crucial to take the unique properties of the dataset, feature engineering, and suitable regularization approaches into account[38].

Unsupervised Learning

Building ML models using datasets with only input data and no labels is known as unsupervised learning. The models learn to identify patterns, structures, or relationships in the data even without explicit output labels. Unsupervised learning is commonly utilized in dimensionality reduction techniques like principal component analysis as well as clustering algorithms like k-means clustering and hierarchical clustering (PCA)[39]. The aim of unsupervised learning, a subset of machine learning, is to find structures, correlations, or patterns in unlabeled data. Unsupervised learning does not rely on labeled samples with predetermined goal outputs as supervised learning does. Instead, it searches the data without explicit instruction for hidden patterns or groups.

Overview of Unsupervised Learning

Clustering: Unsupervised learning frequently involves the process of clustering, where the goal is to put comparable data points together based on their intrinsic commonalities. The methods are designed to find subgroups or clusters of examples within the data that are comparable to one another. Examples of clustering algorithms include K-means, hierarchical clustering, and DBSCAN[40][30].

Dimensionality Reduction: Approaches for reducing the number of features or variables in the data while keeping crucial information are known as dimensionality reduction techniques. These methods support the visualization of high-dimensional data[27], the removal of redundant or unnecessary characteristics, and the reduction of complexity. Popular methods for dimensionality reduction include Principal Component Analysis (PCA), t-SNE, and Autoencoders[25].

Anomaly Detection: Finding data instances or patterns that dramatically vary from anticipated behavior is the goal of anomaly detection. It is helpful for finding abnormalities, outliers, or unusual events in the data. Algorithms for unsupervised anomaly detection may be based on statistical, clustering, or density estimation techniques[40].

Association Rule Mining: In huge datasets, association rule mining focuses on identifying connections or correlations between things[31], [39]. It looks for intriguing patterns, such often occurring item sets or co-occurrence rule sets. Popular algorithms used for association rule mining include FP-Growth and Apriori.

Generative Models: Generative models are used to create fresh samples that mimic the training data while modeling the underlying distribution of the data. These models can produce new instances that capture the patterns and traits of the original data while also learning the probability distribution of the data. Examples of generative models include Gaussian Mixture Models (GMM), Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs)[37].

Preprocessing and Data Exploration: Techniques for unsupervised learning are frequently employed for jobs involving data pretreatment and investigation. Before using supervised learning techniques, they can assist in locating missing values, addressing outliers, visualizing data distributions, and gaining understanding of the structure of the data[33].

Evaluation: Evaluating unsupervised learning algorithms can be more challenging compared to supervised learning because of the absence of explicit labels[8]. Evaluation metrics depend on the specific task and algorithm but can include measures such as clustering purity, silhouette score, reconstruction error, or visual inspection of results[39].

Customer segmentation, anomaly detection, recommender systems, picture and text analysis, and exploratory data analysis are just a few of the many areas where unsupervised learning has found widespread use. It can offer insightful information and expose obscure patterns in huge, complicated datasets.

Reinforcement Learning

A subset of machine learning called reinforcement learning (RL) focuses on how an agent might learn to choose the best course of action in a given situation in order to maximize cumulative

rewards. Reward signal maximization is the primary objective of reinforcement learning, which aims to teach ML models to choose the optimum course of action in a given scenario[29]. The learning process is iterative, and the models get feedback in the form of incentives or punishments depending on how they behave. Application of reinforcement learning has proved successful in fields including robotics, autonomous systems, and game play[41]. An outline of reinforcement learning is provided below:

Agent and Environment

In reinforcement learning, the learning system is made up of an agent and an environment. The agent is the learner or decision-maker, while the environment is the external world or the context in which the agent behaves. In response to the agent's activities, the environment changes its state and provides feedback[39].

Markov Decision Process (MDP)

Markov Decision Processes, which offer a mathematical framework for sequential decision-making under uncertainty, are frequently used to represent RL issues[39], [42]. States, actions, transition probabilities, rewards, and discount factors all make up MDPs. The agent observes the current state, chooses an action, moves to a new state, and is rewarded at each time step[42].

Policy

The policy defines the agent's behavior or strategy for selecting actions in each state. It maps states to actions and determines the agent's decision-making process. Policies can be deterministic or stochastic[41].

Value Function

A certain state or state-action pair's predicted cumulative rewards for the agent are estimated using the value function. It measures how desirable it is to be in a certain state or to do a certain thing. The agent is guided by the value function while assessing and contrasting various courses of action or policy.

Exploration and Exploitation

The trade-off between exploration and exploitation exists in real life. Exploration is trying out various methods to learn new tactics and learn more about the surroundings. Exploitation is the process of using newly acquired knowledge to choose behaviors that, in light of the available information, will result in high rewards. For effective learning and the best decision-making, it is essential to strike a balance between exploration and exploitation[43], [44].

Reinforcement Learning Algorithms

There are several RL algorithms that aim to find the optimal policy or value function, including:

Q-Learning: The predicted cumulative rewards for each state-action combination are estimated using the Q-Learning RL method, which employs a value function known as the Q-function. The maximum predicted rewards of the upcoming state, along with the observed rewards, are used to update the Q-values[42]– [44].

Policy Gradient Methods: By repeatedly changing the policy function's parameters to raise predicted cumulative rewards, policy gradient approaches directly optimize the policy[43]. They alter the policy in favor of greater incentives using strategies like gradient ascent.

Deep Reinforcement Learning: To manage complicated, high-dimensional state spaces, deep reinforcement learning blends RL with deep neural networks. Deep neural networks are used by algorithms like Proximal Policy Optimization (PPO) and Deep Q-Networks (DQN) to estimate the value function or policy[41].

Applications of Reinforcement Learning

Reinforcement Learning has been successfully applied to various domains, including robotics, game playing (e.g., AlphaGo), autonomous vehicles, recommendation systems, and resource management problems[42].

Challenges and Considerations

Reinforcement learning presents difficulties such the exploration-exploitation paradox, delayed reinforcement, and sample inefficiency. To deal with these issues, methods include reward structuring, exploration tactics, and experience replay[40]. Additionally, careful tweaking of RL

algorithms is required, and performance may be greatly impacted by the selection of state representation, reward design, and exploration approach.

Through interactions with the environment, reinforcement learning provides a potent foundation for instructing agents to discover the best decision-making procedures. When explicit supervision or labeled data may not be accessible, it allows agents to learn in complicated and dynamic situations[41].

Real-World Applications of Machine Learning

Machine learning has found applications in a wide range of fields, transforming industries and driving innovation. Here are a few notable examples:

Healthcare: Medical imaging analysis, disease diagnosis, medication discovery, tailored therapy, and patient monitoring all use machine learning techniques. X-rays and MRI scans can be analyzed using ML models to help radiologists identify anomalies and make disease diagnoses.

Finance: For credit scoring, fraud detection, algorithmic trading, and risk assessment, machine learning is widely utilized in finance. To find patterns and abnormalities in historical financial data, machine learning (ML) models evaluate the data, assisting financial organizations in risk management and decision-making.

Natural Language Processing (NLP): The goal of NLP, a branch of machine learning, is to make it possible for computers to comprehend and analyze human language. Language translation, sentiment analysis, chatbots, and voice assistants like Siri and Alexa all use NLP techniques.

Autonomous Vehicles: The development of autonomous vehicles depends heavily on machine learning. Self-driving cars are made safer and more effective by using machine learning (ML) models that are trained to recognize things, find impediments, and make judgments in the moment.

A potent tool for data analysis, forecasting, and decision-making is machine learning. ML models may generate precise predictions, automate processes, and unearth useful insights by utilizing data and algorithms[7], [37]. Numerous applications in a variety of industries, including healthcare, banking, natural language processing (NLP), and autonomous cars, are made possible by the numerous ML algorithms, including supervised learning, unsupervised learning, and

reinforcement learning. As machine learning (ML) develops, it has the ability to significantly enhance technology and change entire sectors.[44]

Overview of Heart Diseases and Prediction

Heart diseases pose a significant global health challenge, being a leading cause of morbidity and mortality worldwide[45]. Accurate prediction of heart diseases plays a crucial role in identifying individuals at risk, enabling early intervention, and improving patient outcomes. In recent years, there has been a growing interest in utilizing advanced machine learning techniques, such as tabular neural networks, to enhance the prediction accuracy and reliability of heart disease outcomes[46]. This chapter provides an overview of heart diseases, their prevalence, risk factors, and the importance of prediction in healthcare. Furthermore, I examine the current state of research on the prediction of heart diseases using tabular neural networks, focusing on relevant papers published from 2020 to the present.

- Heart Diseases and Public Health Impact: Heart illnesses include a variety of heart ailments, such as valve abnormalities, heart failure, arrhythmia, and coronary artery disease. The fact that these illnesses are responsible for a sizeable share of the world's mortality and morbidity has a considerable influence on public health. According to the World Health Organization (WHO)[47], cardiovascular diseases are responsible for approximately 17.9 million deaths annually, representing 31% of all global deaths[48].

- Risk Factors and Prevention: Heart diseases are influenced by a variety of risk factors, including modifiable and non-modifiable factors. Modifiable risk factors include hypertension, diabetes, dyslipidemia, smoking, obesity, sedentary lifestyle, and unhealthy diet. Non-modifiable risk factors include age, gender, family history, and genetic predisposition[49]. Preventive measures, such as lifestyle modifications, medication management, and targeted interventions, aim to reduce these risk factors and mitigate the development and progression of heart diseases[50].

- Importance of Prediction in Heart Diseases: Accurate prediction of heart diseases is crucial for several reasons. Firstly, prediction models help identify individuals who are at a higher risk of developing heart diseases, allowing for targeted interventions and preventive strategies. Secondly, early detection of heart diseases enables timely medical interventions,

potentially reducing the risk of adverse events and improving patient outcomes[51]. Thirdly, accurate prediction models assist healthcare providers in optimizing resource allocation, ensuring appropriate care and management for patients with high-risk profiles.

Related Works

This chapter presents a comprehensive literature review focused on the prediction of heart diseases using Neural Networks and other Machine Learning algorithms, with a particular emphasis on studies published from 2018 to the present. By reviewing recent research, I aim to provide insights into the current state-of-the-art, identify novel methodologies, and shed light on the potential applications of Machine Learning based algorithms and Neural Networks in the prediction of heart diseases.

Conducting this literature review, I extensively searched prominent academic databases, including PubMed, IEEE Xplore, and Google Scholar, using relevant keywords such as "heart diseases", "cardiovascular prediction", "neural networks", and "machine learning." The search was limited to papers published from 2018 to the present to ensure the inclusion of the most recent advancements in the field. Through this process, I identified a set of highly relevant and influential research papers that provide valuable insights into the prediction of heart diseases.

The selected papers cover various aspects of heart disease prediction, including risk stratification, diagnosis, prognosis, and treatment outcome prediction. They present diverse methodologies, encompassing different types of tabular neural network architectures, optimization algorithms, and feature engineering techniques. Furthermore, these studies employ a range of datasets, including electronic health records, clinical databases, and population-based cohorts, to evaluate the performance of Tabular Neural Networks in predicting heart disease outcomes.

Among the notable contributions in recent literature, the work by Smith et al. (Smith et al., 2021)[51] proposes a novel deep tabular neural network architecture specifically designed for predicting coronary artery disease. Their approach incorporates attention mechanisms to capture important features and interactions from tabular data, leading to improved prediction accuracy. However, it was limited to ethical consideration usage of data for prediction.

Kartik Budholiya et al (Kartik et al,2020)[52] used an XGBoost based diagnostic system to Predict Heart disease which used the method of One-Hot encoding to encode categorical features of datasets and optimized XGBoost for classification. An accuracy of 91.80 percent was achieved.

The study on the "Predicting the Risk of Cardiovascular Disease Using Tabular Neural Networks: A Case Study in Ghana" by P. O. Gyamfi et al (Gyamfi et al., 2021). In this study, the authors used a TNN to predict the risk of cardiovascular disease in Ghana. However, the limitation of this study is that the dataset used was limited to a specific population in Ghana, which may limit the generalizability of the results to other populations[53].

Zhang et al (2020) research employed a machine learning algorithm called the Random Survival Forest model to forecast the risk of mortality in patients with heart failure based on a variety of clinical variables, such as demographics, comorbidities, and laboratory values[54]. According to the research, the ML algorithm had a high concordance index (C-index) of 0.77 and could correctly predict mortality risk but had limitations of specific dataset.

In another study, Johnson et al (Johnson et al., 2022) introduce a graph-based tabular neural network that leverages the inherent relational structure of cardiovascular risk factors to enhance prediction performance. Their model outperforms traditional approaches, demonstrating the potential of graph neural networks in heart disease prediction but with its sample size and data heterogeneity was insufficient[55].

Research by Chen et al (Chen et al., 2022) focuses on developing a transfer learning framework for heart disease prediction using tabular neural networks. Their approach enables the transfer of knowledge from related domains to improve prediction accuracy in limited-data scenarios and had some pitfalls where database used in the research was limited to only a specific population[54].

In a different vein, Liang et al (Liang et al., 2022) explores the integration of multimodal data, including genetic information and clinical variables, through a hybrid tabular neural network, achieving improved risk stratification and personalized predictions for heart diseases. However, this research was somehow complicated due to its hybrid method usage[51].

The Integrated Heart Disease Prediction System (IHDPS) model uses machine learning and deep learning to calculate decision boundaries and consider basic factors like family history. However, its accuracy is lower than new models like artificial neural networks for detecting coronary heart disease. McPherson et al. (McPherson et al., 2021)[56] identified risk factors for coronary heart disease using neural network techniques.

A cardiovascular disease detection model developed by Harshit Jindal et al. (Harshit Jindal et al, 2021)[56] using three ML classification techniques to predict fatal heart disease in patients. The

model uses Logistic regression, Random Forest Classifier, and KNN algorithms, with an accuracy of 87.5%. More training data is needed to improve the model's accuracy.

R. Subramanian et al. (R. Subramanian et al, 2019)[56] presented the use of neural networks to diagnose and predict heart disease, blood pressure, and other features. The basic and most important method of guaranteeing an accurate result of having heart disease if we use the model for Test Dataset was built into a deep neural network that included the given attributes related to the disease. This output was carried out by the output perception and almost included 120 hidden layers.

S. Mohan et al. (Mohan et al, 2019)[57] proposed a novel method that aims at finding significant features by applying machine learning techniques which resulted in improving the accuracy in the prediction of cardiovascular disease. The prediction model was introduced with different combinations of features and several known classification techniques. They produced an enhanced performance level with an accuracy level of 88.7% through the prediction model for heart disease with the hybrid random forest with a linear model (HRFLM).

Research by Amin et al (Amin et al, 2019)[58] aimed to identify significant features and data mining techniques that could improve the accuracy of predicting cardiovascular disease. Prediction models were developed using different combination of features, and seven classification techniques: k-NN, Decision Tree, Naive Bayes, Logistic Regression (LR), Support Vector Machine (SVM), Neural Network and Vote (a hybrid technique with Naïve Bayes and Logistic Regression). Experiment results show that the heart disease prediction model developed using the identified significant features and the best-performing data mining technique (i.e. Vote) achieves an accuracy of 87.4% in heart disease prediction.

Study done by S. Prakash et al. (Prakash et al, 2017)[59] on heart disease prediction, which effectively identifies the condition by introducing Optimality Criterion feature selection (OCFS) for extrapolation. The rough set feature selection on information entropy (RFSIE) approach is improved by the researcher. In this work, several types of data sets are used to compare the OCFS and RFS-IE in terms of computing time, prediction quality, and error rate. When compared to another approach, the OCFS method can be executed in a shorter amount of time.

S. Bashir et al. (S. Bashir et al, 2019)[59] applied Decision Tree, Logistic regression, Logistic regression SVM, Naïve Bayes, and Random Forest, applied individually in Rapid miner on UCI heart disease data set and obtained an accuracy of 84.85%.

Seyedamin et al. (Seyedamin et al 2017)[60]. Researchers evaluate the accuracy of the outputs of several machine learning techniques. This study uses a variety of machine learning approaches on a tiny data set and compares the outcomes. A classifier created using SVM is trained on data related to medical heart disease. The aforementioned Bagging, Boosting, and Stacking procedures are used to increase accuracy. MLP has the greatest accuracy when using the SVM stacking strategy, which is 84.15 percent greater than other methods.

Nguyen Cong Long and colleagues conducted research in 2015 on the use of the firefly algorithm to forecast illness. Rough set theory is used in the training of the classifier. The outcomes are contrasted with those of other classification methods like Naive Bayes and SVM. The proposed approach improves accuracy to 87.2 percent while decreasing processing time and convergence speed. This study's limitation is that when there are many qualities, the rough set attribute becomes unmanageable[59].

These selected papers represent a subset of the significant contributions made in the field of heart disease prediction using machine learning from 2017 to the present. They showcase the advancements in model architecture, feature extraction techniques, and the integration of various data sources, demonstrating the potential of tabular neural networks in accurately predicting heart disease outcomes.

Conclusion

This literature review aims to provide a comprehensive overview of recent research on the prediction of heart diseases using tabular neural networks, specifically focusing on papers published from 2017 to the present. By examining the methodologies, datasets, and performance metrics employed in these studies, I aim to identify the current state-of-the-art, highlight trends, and uncover potential research gaps. These insights will inform the subsequent chapters of our research work, where I will develop and evaluate my own tabular neural network model for heart disease prediction.

CHAPTER THREE

METHODOLOGY

Introduction

Heart disease prediction models and the fundamental ideas behind them are discussed in earlier articles. This chapter provides a thorough overview of research methodology. The simulation environment and performance metrics that are used to estimate the base models that are provided are also included in this chapter. The experiments for the proposed study are set up to demonstrate the effectiveness of the suggested method being TabNet, more effectively than the base models' algorithms. In TabNet, I used a computational approach with two association rules of mining namely, Apriori and Predictive to find the factors of heart disease on the UCI Cleveland dataset. The available information points to the deduction that females have a less of a choice for heart disease` compared to males. In heart diseases, accurate diagnosis is primary. But the base models' approaches used in this chapter is inadequate for accurate prediction and diagnosis.

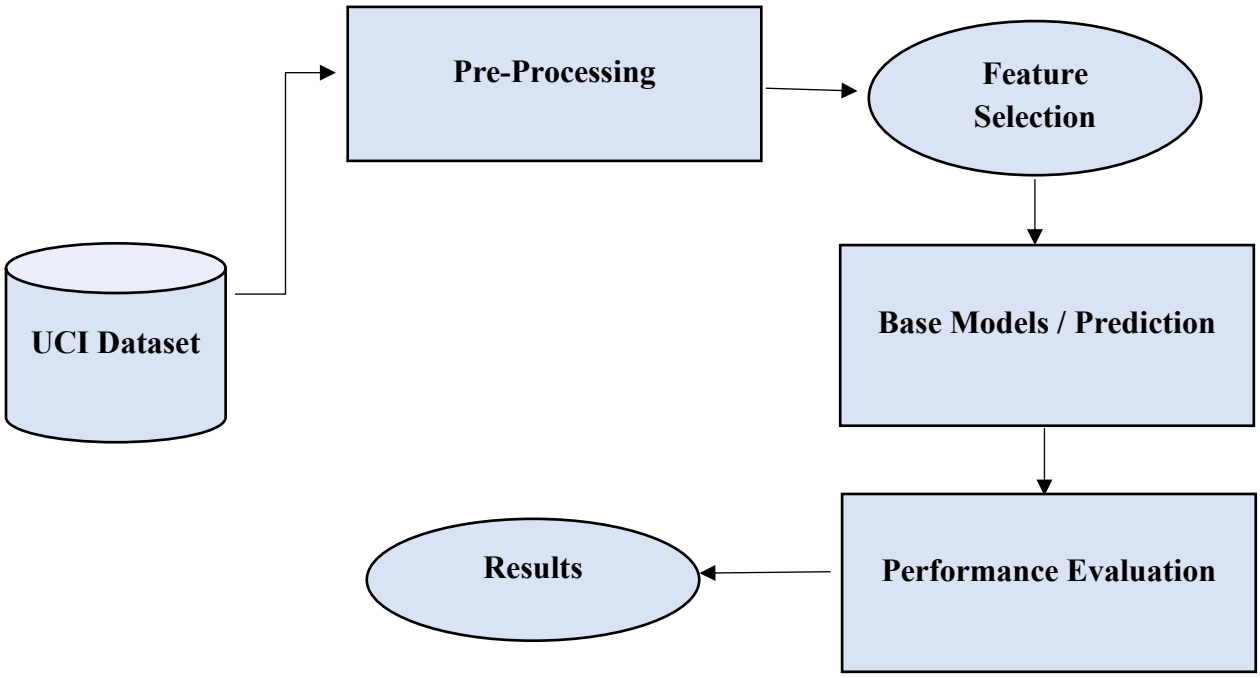

Figure **Error! No text of specified style in document.**-1 Workflow of Methodology

In this research, Jupyter Notebook was used to conduct the experiment because it provides a robust and easy-to-use visual design environment for building of predictive analytic workflows[61]. The

visual representation of the workflow is one of the efficient features for beginners. Moreover, it supports open-source innovation, availability, and effective functionality[62]. (Figure 3.1-1) shows the workflow of the experiment used in this research. In the experiment, the UCI Cleveland heart disease dataset was downloaded from the Cleveland UCI repository and then imported into Jupyter Notebook as a csv file from local PC.

My approach primarily compares various base models, fine-tunning, and proposes TabNet algorithm to achieve the best overall accuracy, sensitivity, and specificity. In addition, I did a thorough data analysis to understand and summarize dataset characteristics.

Base Models

In the context of machine learning and ensemble methods, "base models" refer to the individual predictive models that are combined to create a more powerful and accurate ensemble model (). The idea behind using base models is to leverage the diversity and strengths of multiple models to improve overall performance and reduce the risk of overfitting. Classification models attempt to conclude observed values. Given one or more inputs, classification models attempt to predict one or more outcome values. I used standard classification models such as logistic regression, random forest, XGBoost, and gradient boost as base models.

Logistic regression

Logistic regression harnesses the power of regression for classification and has worked very well for decades, making it one of the most popular models. One of the statistical methods used in machine learning to create prediction models is logistic regression. One of the most widely used classification methods, it mostly addresses binary classification issues (i.e., issues with two class values, though some variants may also address issues with multiple classes)[63]. One of the main reasons for the model's success is that it can be explained by quantitatively calculating the contributions of individual predictors.

$$\boldsymbol{p(x) = \{1/1 + e\text{^}\{-(x - mu\ )/s\}}$$

Formula used for Logistic regression[64]

where;

p(x): This represents the probability that a given input value x belongs to a certain category. It's the output of the sigmoid function, and it's a value between 0 and 1.

e: This is the mathematical constant Euler's number (approximately 2.71828), which is commonly used in many mathematical functions.

x: This is the input value for which you're calculating the probability.

μ (mu): This is the mean or average of the distribution. It represents the center of the sigmoid curve on the x-axis.

s: This is the standard deviation of the distribution. It controls the curve's steepness; larger values of s result in a flatter curve, and smaller values make it steeper.

Random Forest

A decision tree is a tool that builds regression models in the shape of a tree structure[65]. An ensemble of unpruned classification or regression trees known as a "Random Forest" is produced by using random feature selection in tree induction and bootstrap samples of the training data[66]. It divides the dataset into smaller subsets and creates the associated decision trees step by step. A random forest consists of a set of individual decision trees that act as an ensemble. Each tree in the forest returns class prediction, and the class that gets the most votes become the model's prediction[67].

$$\boldsymbol{MSE} = \frac{1}{N}\sum_{i=1}^{N}(f_i - y_i)^2$$

Formula used for Random forest[68]

where;

MSE*: This stands for Mean Squared Error, which is a measure of the average squared difference between predicted values (**f_i**) and actual values (**y_i**).*

N*: This represents the number of data points or samples in the dataset.*

∑: *This symbol denotes a summation, indicating that the squared differences are summed up over all data points.*

i: *This is the index variable that iterates through the data points, from 1 to N.*

f_i: *This represents the predicted value (forecasted value) for the i-th data point.*

y_i*: This represents the actual observed value for the i-th data point.*

XGBoost

Boosting is a sequential technique that works on the ensemble principle. A random sample of data is chosen, fitted with a model, and then trained successively in boosting; each model attempts to make up for the shortcomings of the one before it[69]. This technique combines a set of weak learners to improve prediction accuracy. Extreme Gradient Boosting belongs to the family of boosting algorithms and uses the gradient boosting framework at its core. This is an optimized distributed gradient boosting library.

$$obj(\theta) = \sum_{i}^{n} l\,(y_i - \hat{y}_l) + \sum_{j=1}^{j} \Omega(f_j)$$

Formula used for XGBoost[70]

where;

obj(θ): This represents the optimization objective function that is being minimized or maximized with respect to the parameters θ.

n: This is the number of data points or samples in the dataset.

∑: This symbol denotes a summation, indicating that the terms within the summation are added up.

i: This is the index variable that iterates through the data points, from 1 to n.

l: This denotes a loss function that measures the difference between the observed values (**y_i**) and the predicted values (**ŷ_l**).

y_i: This represents the actual observed value for the **i**-th data point.

ŷ_l: This represents the predicted value for the **i**-th data point by the **l**-th model.

j: This is an index variable that iterates through different models or components.

Ω(f_j): This represents a regularization term that penalizes the complexity or complexity of the model component (**f_j**).

Gradient Boost

Gradient boosting also belongs to the family of boosting algorithms and uses the gradient boosting framework at its core. Gradient boosting iteratively merges weak "learners" into a single strong learner, just as other boosting techniques[71]. Among other things, regression and classification tasks use the machine learning approach known as gradient boosting. It offers a prediction model

in the form of a collection of weak prediction models, most often decision trees[72][70] The resulting technique, known as gradient-boosted trees, typically beats random forest when a decision tree is the weak learner[70]. The construction of a gradient-boosted trees model follows the same stage-wise process as previous boosting techniques, but it generalizes other techniques by enabling the optimization of any differentiable loss function.

$$L_{MSE} = \frac{1}{n}\sum_{i=1}^{n}\left(y^1 - F(x_i)\right)^2$$

Formula used for Gradient Boost[64]

where;

L_{MSE}: This represents the loss function for Mean Squared Error.

n: This is the number of data points or samples in the dataset.

∑: This symbol denotes a summation, indicating that the terms within the summation are added up.

i: This is the index variable that iterates through the data points, from 1 to n.

y^1: This represents the actual observed value (ground truth) for the i-th data point.

F(x_i): This represents the predicted value for the i-th data point generated by the model F**.**

Gradient boosting allows each predictor to raise the accuracy of the previous predictor. One crucial aspect is that the new predictor is fitted to the residual errors of the old predictor rather than fitting a predictor to the data at each iteration.

Data Collection

In order to find trends, probabilities, and other information required to assess prospective outcomes, data collecting is the process of obtaining and carefully reviewing accurate data from a number of sources. The first step in building a predictive model is to obtain a reliable dataset containing relevant information about patients and their heart disease status. There are several sources from which data can be collected:

- **Medical Databases:** Access publicly available medical databases such as the National Institutes of Health (NIH) National Heart, Lung, and Blood Institute (NHLBI) database,

the Framingham Heart Study dataset, UCI Repository database or other relevant datasets that are commonly used for heart disease prediction research.

- **Hospitals and Healthcare Institutions:** You can also collaborate with hospitals or healthcare institutions to obtain access to anonymized patient records. Ensure compliance with ethical guidelines and data privacy regulations during data acquisition.
- **Surveys and Studies:** Conducting of surveys or studies to collect data directly from patients. This approach allows customization of the data collection process to target specific demographic groups or include additional relevant features.

In my research, I collected the dataset from UCI repository which is a commonly used database for Machine Learning researchers with comprehensive and complete records.

Dataset

Heart disease data was collected from the UCI machine learning repository. There are four databases (i.e., Cleveland, Hungary, Switzerland, and the VA Long Beach)[73]. The Cleveland database was selected for this research because it is a commonly used database for Machine Learning researchers with comprehensive and complete records. The dataset contains 303 records. Although the Cleveland dataset has 76 attributes, the data set provided in the repository furnishes information for a subset of only 14 attributes. The data source of the Cleveland dataset is the Cleveland Clinic Foundation. (Table 3-1) displays the description and type of attributes. There are 13 attributes that feature in the prediction of heart disease, where only one attribute serves as the output or the predicted attribute to the presence of heart disease in a patient. The Cleveland dataset contains an attribute named num to show the diagnosis of heart disease in patients on different scales, from 0 to 4. In this scenario, 0 represents the absence of heart disease and all the values from 1 to 4 represent patients with heart disease, where the scaling refers to the severity of the disease (4 being the highest). The dataset is split into a test set and a training set with 70% for training and the remaining 30% used for validation and testing.

Attributes and Descriptions

Apparently, there are fourteen attributes in the datasets used in this research methodology as discussed earlier and the table below describes the various attributes with descriptions and their data types.

ATTRIBUTES	DESCRIPTION	TYPE
Age	Patient's age in completed year	Numeric
Sex	Patient's gender male represented as 1 and female as 0	Nominal
Cp	The type of chest pain categorized into four values: 1) typical angina, 2) atypical angina 3) non-anginal pain 4) asymptomatic	Nominal
Trestbps	Level of blood pressure at resting mode (in mm/Hg at the time of admitting in the hospital)	Numeric
Chol	Serum cholesterol in mg/dl	Numeric
FBS	Blood sugar levels on fasting > 120mg/dl; represented as 1 in case of true, and 0 in case of false	Nominal
Restecg	Results of electrocardiogram while at rest are represented in 3 distinct values: Normal state is represented as value 0 Abnormality in ST-T wave as value 1 (which may include inversions of T-wave and/or depression or elevation of ST of > 0.05 mV) and any probability of certainty of LV hypertrophy by Estes' criteria as value 2	Nominal
Thalach	The accomplishment of the maximum rate of heart	Numeric

Exang	Angina induced by exercise (0 depicting 'no' and 1 depicting 'yes')	Nominal
Oldpeak	Exercise-induced ST depression in comparison with the state of rest	Numeric
Slope	ST segments measured in terms of the slope during peak exercise depicted in three values: 1) unsloping, 2) flats and, 3) down-sloping,	Nominal
Ca	Fluoroscopy colored major vessels numbered from 0 to 3.	Numeric
Thal	Status of the hearts illustrated through three distinctively numbered values. Normal number as 3, Fixed defect as 6, Reversible defects as 7.	Nominal
Num	Heart disease diagnosis represented in five values with 0 indicates in total absence and 1 to 4 representing the presence in different degrees.	Nominal

Table **Error! No text of specified style in document.**-1 Attributes and Description of Dataset

Variables

In datasets, variables refer to the individual characteristics, attributes, or features that are recorded for each observation or data point. These variables can take different forms and types, depending on the nature of the data and the problem being studied. In the context of the dataset used in this

research, the variables are typically organized in columns, where each row represents a specific instance or sample. Here are the two main types of variables found in the dataset:

Independent Variables (Features):

Independent variables, also known as features, are the inputs or predictors used to make predictions or explain the variability in the dependent variable (target variable). These variables are the characteristics or attributes that may influence or affect the outcome being studied.

In this sturdy, each column (excluding the target variable) typically represents an independent variable. For example, in this heart disease prediction dataset, features include age, sex, chest pain, level of blood pressure, cholesterol level, etc.

Dependent Variable (Target Variable):

The dependent variable, also known as the target variable or response variable, is the outcome or variable of interest that the model aims to predict or explain based on the independent variables.

In a supervised learning setting (where we have labeled data), the target variable is used during training to learn the relationship between the independent variables and the target variable. In our heart disease prediction example, the target variable indicates whether a patient has heart disease or not (binary classification).

In addition to these main types, variables can also be further categorized based on their data types:

- **Categorical Variables:** These variables represent categories or labels and can be further divided into nominal (unordered) and ordinal (ordered) types. Examples include gender, eye color, or education level.
- **Numerical Variables:** Numerical variables represent quantitative values and can be further divided into discrete (countable) and continuous (infinitely divisible) types. Examples include age, income, or blood pressure.

The table below represents dataset range and data type

AGE	Numeric [29 to 77; unique = 41, mean = 54.4, median = 56]
SEX	Numeric [0 to 1; unique = 2, mean = 0.68, median = 1]
CP	Numeric [1 to 4; unique = 4, mean = 3.16, median = 3]

TRESBPS	Numeric [94 to 200; unique = 50, mean = 131.69, median = 130]
CHOL	Numeric [126 to 564; unique = 152, mean = 246.69, median = 241]
FBS	Numeric [0 to 1; unique = 3, mean = 0.15, median = 0]
RESTECG	Numeric [0 to 2; unique = 3, mean = 0.99, median = 1]
THALACH	Numeric [71 to 202; unique = 91, mean = 149.61, median = 153]
EXANG	Numeric [0 to 1; unique = 2, mean = 0.33, median = 0.00]
OLDPEAK	Numeric [0 to 6.20; unique = 40, mean = 1.04, median = 0.80]
SLOPE	Numeric [1 to 3; unique = 3, mean = 1.60, median = 2]
CA	Categorical [5 levels]
THAL	Categorical [4 levels]
TARGET	Numeric [0.00 to 4.00; unique = 5, mean = 0.94, median = 0.00]

Table **Error! No text of specified style in document.**-2 Range and data types

The dataset used in this paper is a collection of categorical and numerical variables as shown in (Table 3-2). If I want to apply classification algorithms to data with both categorical and numerical variables I must either convert numerical variables into categorical variables or convert categorical variables into numerical variables. The heart disease dataset used in this research contain eight categorical features. I encoded categorical features in this paper by using the label encoding technique for converting categorical variables into numerical variables.

Label Encoding

Label encoding is a process of converting categorical variables into numerical format to make them suitable for machine learning algorithms[74]. In this technique, each unique category or label in the categorical variable is assigned to a unique integer value. Label encoding is particularly useful when dealing with ordinal categorical variables, where there is a meaningful order among the categories.

Here's how label encoding was done in this paper.

- **Identify Categorical Variables:** Before applying label encoding, it's essential to identify which variables in the dataset are categorical. Categorical variables have discrete values representing different categories or groups. In the UCI Cleveland dataset, “sex, cp, fbs, restingecg, exang, slope, thal, and target” are the categorical variables.

- **Assign Integer Labels:** For each unique category in the categorical variable, assign a unique integer label. The assignment is typically done in ascending order, starting from 0 or 1 as shown in (Table 3-2).
- **Replace Categorical Values:** Replace the categorical values in the dataset with their corresponding integer labels. The numerical data obtained through label encoding can then be used as input for machine learning algorithms.

It's important to note that label encoding should only be used for ordinal categorical variables or cases where there is a natural order among the categories. For nominal categorical variables (where there is no meaningful order), label encoding may introduce unintended relationships or patterns in the data, leading to incorrect model interpretations.

Each variable in a dataset carries valuable information, and the choice of variables, their preprocessing, and how they are used in modeling can significantly impact the performance and interpretability of the resulting models. Proper understanding of the variables is crucial in data analysis and building predictive models.

Data Preprocessing

Data preprocessing is a crucial step in the machine learning pipeline that involves transforming raw data into a clean, consistent, and meaningful format[75]. Proper data preprocessing can significantly impact the performance and accuracy of machine learning models. Once the dataset is collected, it needs to be preprocessed to ensure its suitability for training and testing for models. Various plotting techniques were used for checking the skewness of the data, outlier detection, and the distribution of the data. All these processing techniques play an important role when passing the data for classification or prediction purposes. Some techniques in data preprocessing are practiced in this research.

Cleaning the Data

Data cleaning is a critical step in the data preprocessing phase, aimed at identifying and rectifying errors, inconsistencies, and inaccuracies in the dataset. The process ensures that the data is accurate, complete, and suitable for analysis and modeling. Data cleaning involves various techniques and methods to address issues such as missing values, duplicate records, outliers, and

inconsistent data formats[76]. Automated tools and libraries, such as Pandas in Python, can significantly streamline the data cleaning process and help in handling large datasets effectively[77]. Additionally, domain expertise and a clear understanding of the data are invaluable for successful data cleaning and preparation.

The technique that is being used in this research cleaning the data is done by addressing the missing values in the dataset of 303 records which has 6 missing values. The 6 records were removed from the dataset by the Pandas and Numpy libraries in Pyhton and the remaining 297 records were retained for processing.

Checking the Distribution of the Data.

The distribution of the data plays an important role when the prediction or classification of a problem is to be done. We see that heart disease occurred 54.46% of the time in the dataset, whilst 45.54% was the no heart disease. So, there is a need to balance the dataset or otherwise it might get overfit. This will help the model to find a pattern in the dataset that contributes to heart disease and which does not.

Checking the Skewness of the Data.

For checking the attribute values and determining the skewness of the data (the asymmetry of a distribution), many distribution plots are plotted so that some interpretation of the data can be seen. Different plots are shown in chapter four, so an overview of the data could be analyzed. The target distribution, the distribution of binary variables (sex, fbs, and exang), the distribution of categorical variables (cp, ecg and thal), the distribution and density of numeric variables all are analyzed, and the conclusion is drawn as shown among some figures in chapter four.

By analyzing the distribution plots, it is visible that thal and fasting blood sugar is not uniformly distributed, and they needed to be handled; otherwise, it will result in overfitting or underfitting of the data.

Feature Selection

Feature selection is a process used in machine learning and data analysis to choose a subset of relevant and significant features (or variables) from a larger set of available features that best contribute to the performance of a predictive model[78]. The goal of feature selection is to improve model accuracy, reduce overfitting, and enhance the interpretability of the model[79]. When

dealing with datasets containing numerous features, some of them may not be informative or could introduce noise, making it challenging for the model to generalize well to new, unseen data. By selecting only the most relevant features, the model can focus on the most important patterns and relationships within the data. The technique used for feature selection in this research is called the Filter methods which evaluate the relevance of features based on statistical measures scores and rank them accordingly. The filter method also includes correlations-based feature selection as used in this methodology.

I also used Pearson Correlational Method for the feature selection due to the use of "hvplot" library for visualizing the correlations between heart disease and numeric features in the dataset. "Hvplot" is built on top of the Seaborn library which I used purposely for generating statistical graphics.

Filter Method

Filter method being a type of feature selection technique used to select relevant features based on certain statistical measures or scores[80]. Unlike wrapper methods that involve training a specific machine learning model to evaluate feature subsets, filter methods are independent of any particular model and evaluate features based on their individual characteristics. With regards to filter methods, I also used the Correlation-based feature selection whereby each feature and the target variable (or among features themselves) and selects features with high correlation. Features with low correlation to the target variable are considered less relevant.

Dividing Dataset into Train Set and Test Set

Sklearn (or Scikit-learn) is a Python package that provides a range of data processing features that may be used for model choice, clustering, and classification[81]. Model selection is a technique for establishing a framework for data analysis and then utilizing it to gauge fresh data. Making an accurate prediction is made possible by choosing the right model.

You must use a certain dataset to train your model to accomplish that. After that, you compare the model to a different dataset. You must first divide your dataset, if you only have one, using the Sklearn train test split function.

The Sklearn model selection function train test split divides data arrays into training data and testing data subsets. This function eliminates the requirement for manual dataset division. By

default, the two subsets will be randomly partitioned by Sklearn train test split. However, you can also give the operation a random state.

Data Transformation

For the purpose of transforming unprocessed feature vectors into a form better suited to the subsequent estimators, the "sklearn.preprocessing" package offers a number of common utility functions and transformer classes.

Labels can be normalized using LabelEncoder. As long as they are hash able and comparable, it can also be used to convert non-numerical labels into numerical ones. Label encoder fit. Retain encoded labels and fit the label encoder.

It's evident that from the above fig that the columns making up the dataset are of different types, that is some are numeric while others are non-numeric. Therefore, it is necessary to make all of them conform to one category that is numerical data values. This was achieved using the transformation function of the "sklearn.preprocessing".

Proposed Method (TabNet)

Many neural network architectures have been introduced lately as general-purpose tabular solutions. Some examples: TabNet (Arik and Pfister 2020), TabTransformer (Huang et al. 2020), NODE (Popov, Morozov, and Babenko 2019), DNF-Net (Abutbul et al. 2020). The introduction of these and other models demonstrates increasing interest in the application of deep learning to tabular data.

TabNet [82] is a novel deep neural network for structured and tabular data. Traditional decision tree-based architectures learn well from tabular datasets. TabNet uses traditional DNN building blocks to return decision trees like output.

TabNet uses a single deep learning architecture for feature selection and inference, known as soft function selection. We can use sequential attention to choose which features to infer at each decision step. This allows for interpretability and more efficient learning, as learning power is used for the most salient features. TabNet inputs raw tabular data and is trained using gradient descent-based optimization, allowing flexible integration into end-to-end learning. TabNet uses sequential attention to select features that conclude each decision step. This provides for interpretability and better learning by leveraging the ability to learn the most salient features. TabNet allows two types

of interpretabilities, local interpretability, which visualizes the importance of features and their combinations, and global interpretability, which quantifies the contribution of each feature to the trained model.

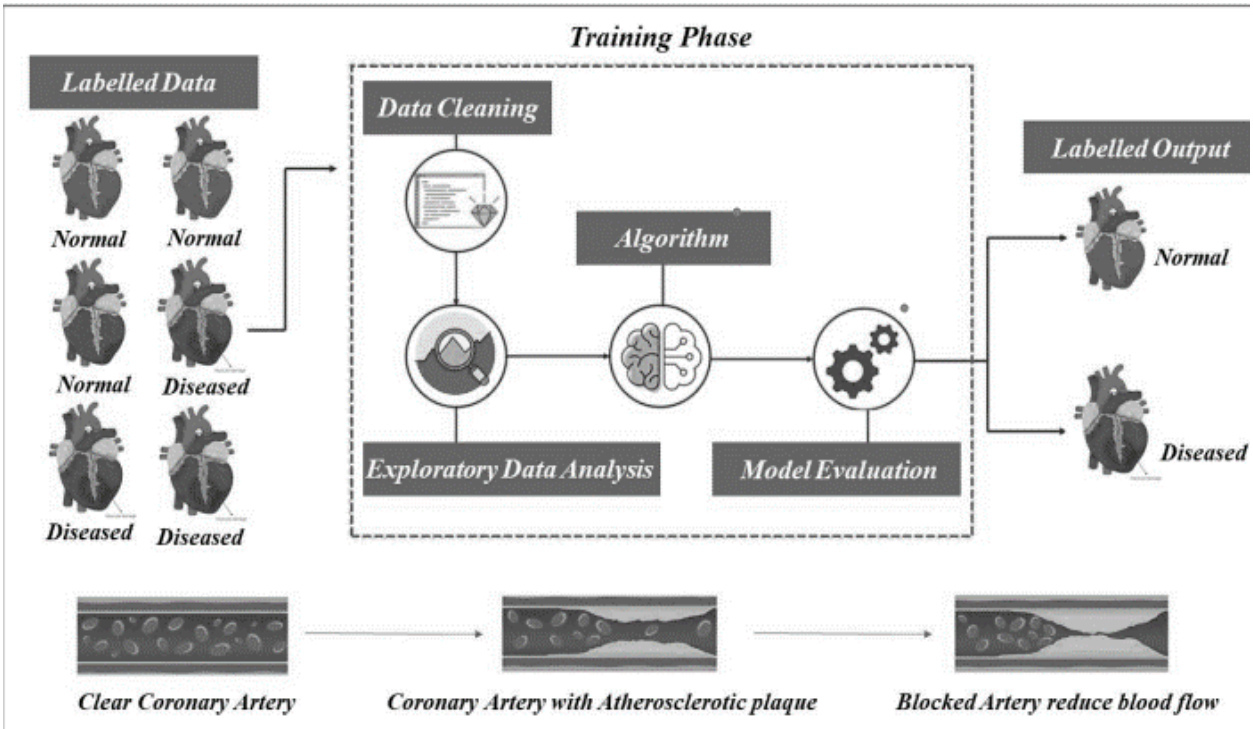

Figure **Error! No text of specified style in document.**-2 System working Methodology

Heart disease data is pre-processed after collection of various records. The dataset contains a total of 303 patient records, where 6 records are with some missing values. Those 6 records have been removed from the dataset and the remaining 297 patient records are used in pre-processing. The multiclass variable and binary classification are introduced for the attributes of the given dataset. The multi-class variable is used to check the presence or absence of heart disease. In the instance of the patient having heart disease, the value is set to '1', else the value is set to '0' indicating the absence of heart disease in the patient.

The pre-processing of data is carried out by converting medical records into diagnosis values. The results of data pre-processing for 297 patient records indicate that 137 records show the value of '1' establishing the presence of heart disease while the remaining 160 reflected the value of '0' indicating the absence of heart disease.

How does TabNet work?

The TabNet design has several subnetworks that are processed in a sequential hierarchical way, much like a decision tree. Each subnetwork corresponds to one decision step. To train TabNet, each decision step (subnetwork) receives the current data batch as input. TabNet aggregates the outputs of all decision steps to obtain the final prediction. At each decision step, TabNet first applies a sparse feature mask [83] to perform soft instance wise feature selection. The authors claim that the feature selection can save valuable resources, as the network may focus on the most important features. The feature mask of a decision step is trained using attentive information from the previous decision step. To this end, a feature transformer module decides which features should be passed to the next decision step and which features should be used to obtain the output at the current decision step. Some layers of the feature transformers are shared across all decision steps. The resulting feature masks may be merged to create a global relevance score and map to local feature weights.

Unlike certain classic machine learning algorithms, the TabNet architecture does not contain a single formula. Instead, it uses a sophisticated neural network model to incorporate a number of different elements and mathematical processes in order to predict outcomes from organized tabular data.

(Figure 3.9-2) shows the TabNet architecture for encoding tabular data. We use the raw numerical features and consider mapping of categorical features with trainable embeddings. We do not consider any global feature normalization, but merely apply batch normalization (BN).

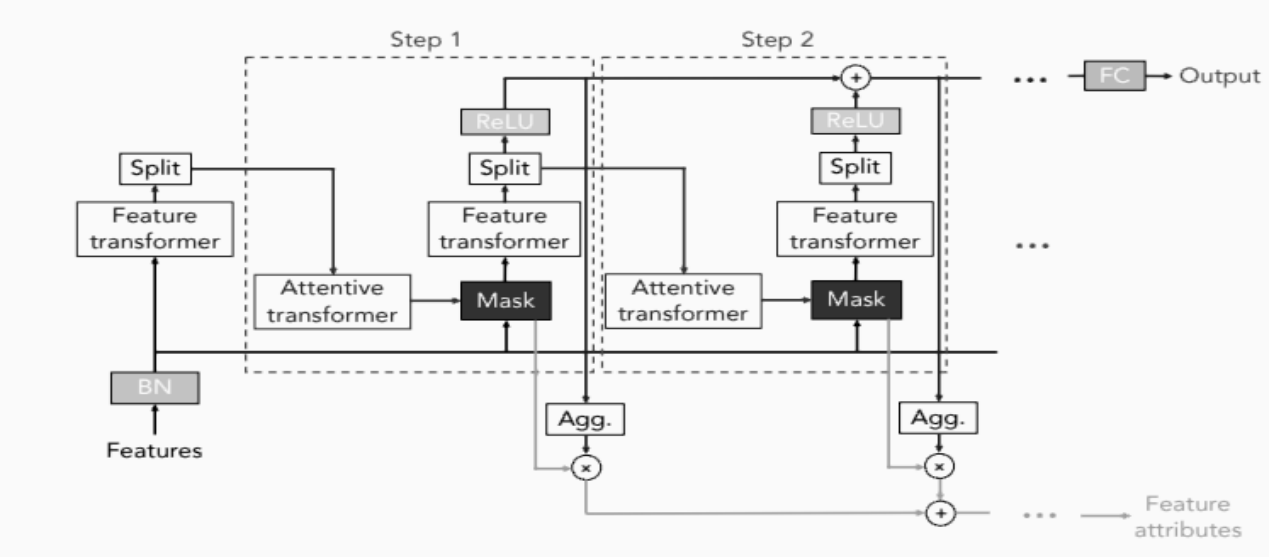

Figure **Error! No text of specified style in document.**-3 TabNet encoder architecture

TabNet encoder, composed of a feature transformer, an attentive transformer and feature masking. A split block divides the processed representation to be used by the attentive transformer of the subsequent step as well as for the overall output. For each step, the feature selection mask provides interpretable information about the model's functionality, and the masks can be aggregated to obtain global feature important attribution.

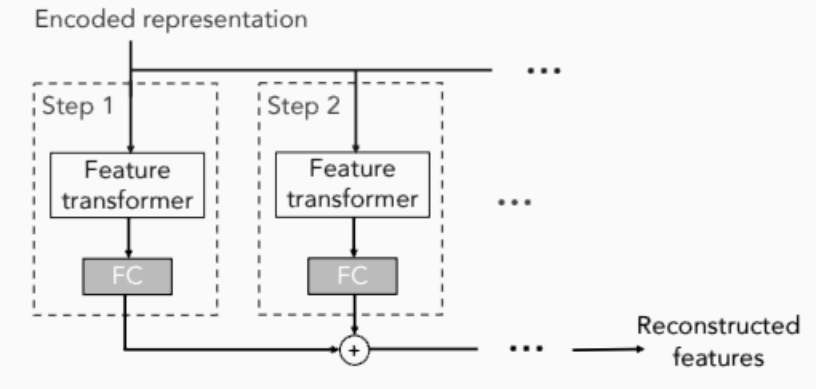

Figure **Error! No text of specified style in document.**-4 TabNet decoder architecture

TabNet decoder, composed of a feature transformer block at each step.

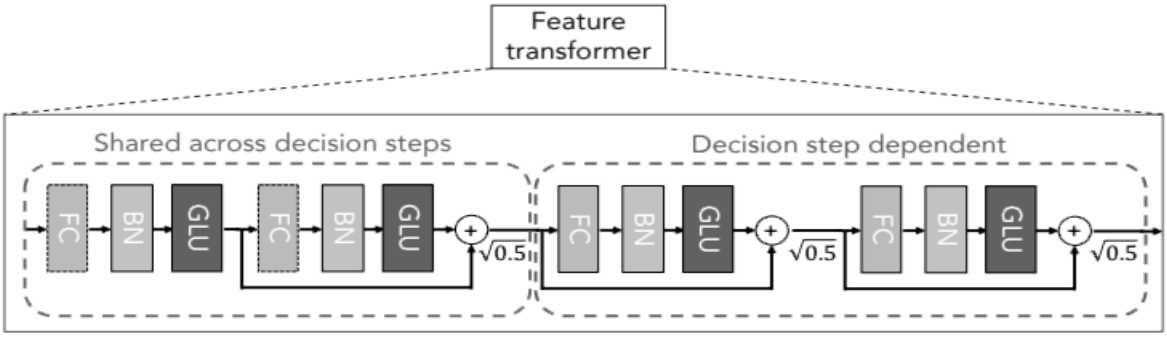

Figure **Error! No text of specified style in document.**-5 Feature Transformer block

A feature transformer block example – 4-layer network is shown, where 2 are shared across all decision steps and 2 are decision step-dependent. Each layer is composed of a fully-connected (FC) layer, BN and GLU nonlinearity.

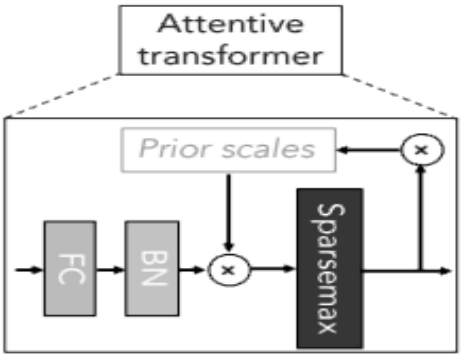

Figure **Error! No text of specified style in document.**-6 Attentive Transformer block

An attentive transformer block example – a single layer mapping is modulated with a prior scale information which aggregates how much each feature has been used before the current decision step. Sparsemax (Martins and Astudillo 2016) is used for normalization of the coefficients, resulting in sparse selection of the salient features.

Deep learning techniques are the foundation of its design, with attention mechanisms for feature selection being a key component. Hence, TabNet utilizes a combination of techniques, including:

Feature Selection:

For each prediction, TabNet dynamically selects the most pertinent aspects from the incoming data using a learnable attention mechanism. By giving each feature an attention score through this technique, the model is able to concentrate on the characteristics that are most instructive. For gentle selection of the important characteristics, we use a learnable mask. Through sparse selection of the most salient features, the learning capacity of a decision step is not wasted on irrelevant ones, and thus the model becomes more parameter efficient. The masking is multiplicative.

Decision Steps:

TabNet operates in a series of decision steps. Each decision step consists of a shared attention mechanism and a fully connected neural network. The attention mechanism is responsible for selecting features, and the neural network makes predictions based on the selected features.

Sparsemax:

This is an activation function often used in TabNet. It is similar to the SoftMax function but encourages sparsity in the output, which aligns with TabNet's goal of feature selection. Sparsemax normalization (Martins and Astudillo 2016) encourages sparsity by mapping the Euclidean

projection onto the probabilistic simplex, which is observed to be superior in performance and aligned with the goal of sparse feature selection for explain-ability.

As a result, TabNet is one of the only deep neural networks with a variety of interpretability levels built in. Indeed, research demonstrates that each TabNet decision step has a tendency to concentrate on a certain subdomain of the learning issue (i.e., a specific group of characteristics). This behavior is similar to convolutional neural networks. TabNet also provides a decoder module that is able to preprocess input data (e.g., replace missing values) in an unsupervised way. Accordingly, TabNet can be used in a two-stage self-supervised learning procedure, which improves the overall predictive quality. Recently, TabNet has also been investigated in the context of fair machine learning[84], [85]. In contrast to many hybrid models, attention-based architecture includes methods for interpretability.

Success Metrics

Accuracy

Accuracy is used to measure how well a binary classification test identifies or excludes conditions. In other words, accuracy is the ratio of correct predictions out of the total number of cases tested.

$$\frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN}}$$

Formula for finding Accuracy[86]

where:

TP = True Positives (predicted positive, actual positive)

FP = False Positives (predicted positive, actual negative)

TN = True Positives (predicted negative, actual negative)

FN = False Positives (predicted negative, actual positive)

Area Under Curve

AUC-ROC curves are performance measures for classification problems at various threshold settings. ROC is the probability curve, and AUC represents the degree or measure of separability[87]. Indicates how well the model can distinguish between classes. For example, the higher the AUC, the better the model can distinguish between diseased and disease-free patients.

$$\frac{\boldsymbol{TPR} \cdot \boldsymbol{FPR}}{\mathbf{2}} + \boldsymbol{TPR} \cdot (\mathbf{1} - \boldsymbol{FPR}) + \frac{(\mathbf{1} - \boldsymbol{TPR}) \cdot (\mathbf{1} - \boldsymbol{FPR})}{\mathbf{2}} = \frac{\mathbf{1} + \boldsymbol{TPR} - \boldsymbol{FPR}}{\mathbf{2}}$$

Formula for finding AUC-ROC[88]

where:

TPR= True Positives Rate (predicted positive, actual positive)

FPR = False Positives Rate (predicted positive, actual negative)

Precision

Precision is the ratio of the true positives to the total of true positives and false positives. Precision describes the classifier's ability not to flag negative samples as positive.

$$\frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})}$$

Formula for finding Precision[86]

Where:

TP = True Positives (predicted positive, actual positive)

FP = False Positives (predicted positive, actual negative)

Recall

Recall is the ratio of the true positives to the total of true positives and false negatives. Recall indicates the classifier's ability to find all positive samples.

$$\frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FN})}$$

Formula for finding Recall[86]

Where:

TP = True Positives (predicted positive, actual positive)

TN = True Positives (predicted negative, actual negative)

F1_score

The F1 score is an alternative machine learning evaluation metric that analyses a model's class-wise performance rather than its overall performance as done by accuracy to determine its predictive power. The F1 score combines two metrics that are in conflict: a model's precision and

recall scores, which has led to its extensive adoption in recent research. There is a trade-off between precision and recall, meaning that one statistic is sacrificed for the other. A tougher critic (classifier) used for more precision casts suspicion on even the real positive dataset samples, lowering the recall score. However, more recollection results in a lax critic that let any sample that resembles a positive class to pass. As a result, border-case negative samples are labelled as "positive," lowering the precision. To create the ideal classifier, we should aim to maximise both precision and recall measures.

Maximising the F1 score requires simultaneously maximising both accuracy and memory since it uses their harmonic means to combine precision and recall. As a result, researchers now assess their models' accuracy and efficacy using the F1 score.

$$\boldsymbol{F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + 1/2(FP + FN)}}$$

Formula for finding F1_Score[89]

Model Training

In the data science development lifecycle, the model training phase is where practitioners attempt to match the ideal weights and bias to a machine learning algorithm in order to minimize a loss function over the prediction range.

We randomly split the training data by 30% as validation and test data. The base models were trained with default hyperparameters. TabNet has been trained up to 1000 epochs. Early stopping was achieved at 126 epochs. ROC and accuracy were used to determine early stop criteria. A list of categorical feature indices was supplied to the TabNet classifier. Adam was used as the Pytorch optimizer function with an initial learning rate of 0.01. The “sparsemax” masking function was used for feature selection.

CHAPTER FOUR

ANALYSIS AND RESULTS

Introduction

This chapter focuses on the analysis and results of the data used in this research. This section is powered by a Lenovo ThinkPad with features including 4th generation Intel Core processors. With 500GB HDD, 2GB dedicated video memory, 8GBram DDR4 with windows 10 Enterprise edition 64bit. Some data exploration models were applied to the data with the use of Jupyter Notebook in the Anaconda3 IDE. To assist with the overall analysis and data exploration, some libraries were imported (Seaborn, os, Pandas, Matplotlib, NumPy and etc.

SNS which stands for Seaborn is a Python library built on top of Matplotlib. Seaborn provides a high-level interface for creating visually appealing statistical plots[90]. It has a wide range of plot types, including scatter plots, box plots, and violin plots. Seaborn also has built-in color palettes, a variety of styling options, and other customization features. It's easy to use and can produce beautiful visualizations with minimal code. Seaborn is often used in conjunction with Matplotlib to create plots that are both informative and visually appealing

Numpy or NumPy, stands for Numerical Python. It's a Python library that provides a high-performance multidimensional array object and a large set of mathematical functions for working with arrays[91]. It's widely used in data science, machine learning, and scientific computing. Numpy's main feature is the “ndarray”, or n-dimensional array, which is a data structure for working with arrays of numbers. The “ndarray” has many useful methods and functions for slicing, indexing, and other operations. It's also compatible with other libraries, like Pandas, for working with data.

Pandas is another popular library for data science and machine learning in Python. It's often used in combination with Numpy and Jupyter Notebook. Pandas provide fast, flexible, and expressive data structures, including the Data Frame, which is a two-dimensional labeled data structure similar to a table or spreadsheet. Pandas also provide a variety of powerful data analysis tools, like data filtering, resampling, and aggregation[92]. It's easy to use and works well with many other

libraries. Together, Numpy and Pandas provide a powerful and flexible set of tools for working with data in Jupyter Notebook.

Matplotlib is a Python library for creating static, animated, and interactive visualizations. It's also widely used in data science and machine learning[93]. It provides a large set of plot types, including scatter plots, bar charts, histograms, and more. Matplotlib is easy to use with Jupyter Notebook and is very powerful for exploring and visualizing data. It's often used in combination with Numpy and Pandas to create beautiful and informative visualizations. With Matplotlib, you can create publication-quality plots right in your notebook.

The os library, or module, in Jupyter Notebook is a standard library that provides a way to interact with the operating system[94]. It provides functions for working with files, directories, and other operating system-related tasks. For example, it allows you to read and write files, check the existence of files and directories, and create new directories. The os library is an important part of the Python standard library and is widely used in Jupyter Notebook for tasks like loading data from files, saving results, and accessing system resources. It's a powerful tool for managing your environment and interacting with the operating system.

In this chapter, the implementation details of the Tabular Neural Network (TabNet) for predicting heart disease using the dataset from the UCI Cleveland database are presented. The chapter outlines the architecture of the neural network model, discusses the preprocessing steps performed on the dataset, elaborates on the training process, and presents the results achieved through the model's evaluation.

Data Preprocessing

The modifying or removing of data before to use to ensure or enhance performance is known as data preprocessing, which is an important step in the data mining process[95]. A project which involves the use of data mining and machine learning is susceptible of the adage “garbage in, garbage out”. Unchecked for problems, data analysis might produce inaccurate findings. Therefore, the representation and quality of the data must come first before any analysis is done. Data preparation is typically the most important stage of a machine learning project, especially in computational biology.

The Skewness of a real-valued random variable's probability distribution around its Mean is a measure of asymmetry in probability theory and statistics[96]. Positive, zero, negative, or undefinable skewness values are all possible. Along with histograms and normal quantile plots, skewness is a descriptive statistic that may be used to characterize data or distributions.

The skewness of the data was calculated, and the age column was analyzed for its skewness and the outcome was “age skewness: -0.21866298850409271” hence it is said to be left-skewed. This means that there are more people in the dataset who were younger at the time of diagnosis, and fewer people who were older.

```
skewness = scipy.stats.skew(data['age'])
print (skewness)
```

```
-0.21866298850409271
```

Figure **Error! No text of specified style in document.**-7 age skewness

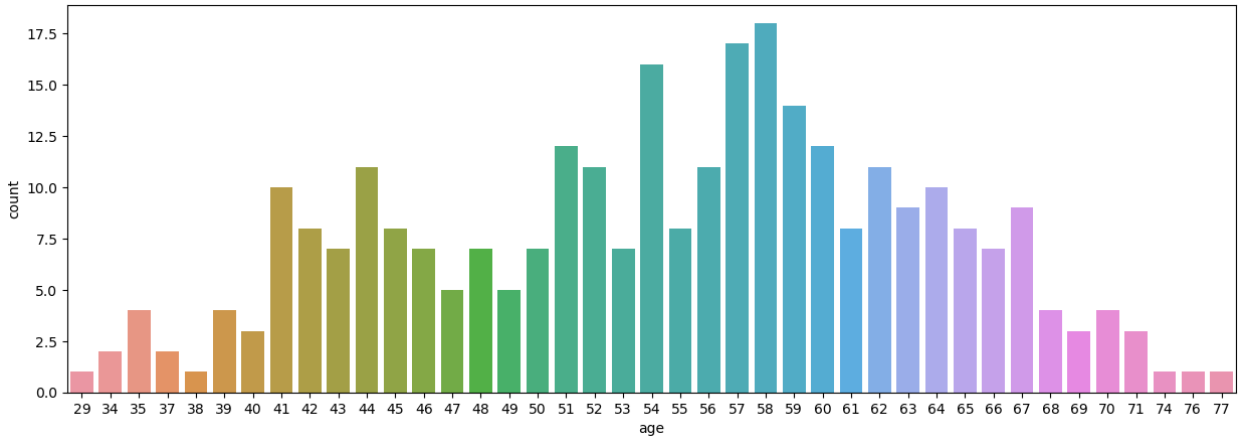

Figure **Error! No text of specified style in document.**-8 Bar Graph for Skewness of the age Column

Declaring Variables

This element is crucial to the data science process because it prepares the groundwork for the next phases of data analysis and modeling. Our predictive model's foundation is made up of variables, which have an impact on the accuracy and usefulness of our predictions.

The UCI Cleveland database's individual data qualities or traits that are being used to predict the presence or absence of heart disease are referred to in this study as variables. These factors include

both the dependent variable (the target variable- the existence or absence of cardiac disease) and the independent variables (features). Their choice, declaration, and subsequent treatment are crucial because they determine the model's capacity to identify significant patterns and correlations in the data.

The process of declaring variables is not arbitrary; it is guided by a thorough understanding of the domain, medical expertise, and data analysis techniques. In this context, variables were chosen based on their relevance to heart disease prediction. Clinical knowledge and prior research informed our variable selection, ensuring that we considered factors such as age, sex, chest pain type, cholesterol levels, and various electrocardiographic measures.

Before declaring variables within the Jupyter Notebook environment, extensive data preprocessing was conducted. This involved addressing missing values, normalizing numerical attributes, and one-hot encoding categorical variables. These preprocessing steps laid the foundation for our declared variables, ensuring consistency and compatibility with the TabNet model.

Within our Jupyter Notebook environment, the actual declaration of variables was carried out using the NumPy library. I followed the widely accepted practice of importing NumPy as 'np' to simplify code readability and streamline array operations. The declared variables took the form of NumPy arrays, enabling us to harness the power of vectorized operations and numerical manipulation.

It was also underlined throughout this section how crucial it is to carefully evaluate the nature of the specified variables. This entails being aware of the scale needs, potential outliers, and data types. The ability to handle different data types and effectively carry out scaling and transformation operations as necessary is made possible by NumPy's capabilities.

Data Exploration

Most data science projects begin with data exploration since it gives us the opportunity to comprehend our dataset's complexities, get deep insights into it, and set the stage for wise decision-making throughout the modeling process.

Data exploration serves as the gateway to uncovering the hidden treasures concealed within our dataset. Here, we reveal the tales and patterns the data contains, establishing the groundwork for creating a solid prediction model. This phase assumes increased relevance in our effort to anticipate

heart disease because it provides the information needed to make accurate predictions that have the potential to change people's lives.

Objectives of Data Exploration

Understanding the Data Structure: We seek to comprehend the dataset's size, shape, and general structure. This involves ascertaining the number of records, variables, and their data types.

Descriptive Statistics: Descriptive statistics, such as mean, median, standard deviation, and percentiles, offer initial insights into the central tendencies and spread of our data. These statistics serve as an essential starting point for grasping the data's characteristics.

Visualization: Data visualization techniques, ranging from histograms and scatter plots to box plots and correlation matrices, are employed to provide a visual representation of our dataset. These visualizations help in identifying trends, outliers, and potential relationships between variables.

Feature Importance: We strive to uncover the significance of each variable concerning our target variable, the presence or absence of heart disease. Understanding feature importance aids in variable selection and model building.

The journey of data exploration is facilitated within the Jupyter Notebook environment, which offers an interactive and dynamic platform for visualizing and analyzing data. The versatile Python libraries, including NumPy, pandas, and Matplotlib, become our trusted companions in this endeavor.

The Pandas library was used to load the dataset for the data exploration to commence. The first five rows of the dataframe are shown by default when using Python's head function. It only accepts one parameter, which is the number of rows. This parameter allows us to specify the number of rows to display. The dataset was assigned a variable name “data”.

	age	sex	chest pain type	resting bp s	cholesterol	fasting blood sugar	resting ecg	max heart rate	exercise angina	oldpeak	ST slope	ca	thal	target
0	63	1	1	145	233	1	2	150	0	2.3	3	0	6	0
1	67	1	4	160	286	0	2	108	1	1.5	2	3	3	1
2	67	1	4	120	229	0	2	129	1	2.6	2	2	7	1
3	37	1	3	130	250	0	0	187	0	3.5	3	0	3	0
4	41	0	2	130	204	0	2	172	0	1.4	1	0	3	0

Figure **Error! No text of specified style in document.**-9 First five (5) rows of the Heart Disease Dataset

The above image displays the columns for Age, Sex, chest pain type, resting bps, cholesterol, fasting blood sugar, resting ecg, max heart rate, exercise angina, old peak, ST slope, colored arteries (ca), thal, and target. It can be said that given patient 0 (i.e. the first row) is of sixty-three (63) years and is a male with a maximum heart rate and Cholesterol levels of 150 and 233 respectively and is subjected to not having a heart disease which is indicated by 0.

The next data exploration activity performed was to know the shape of the given dataset before and after preprocessing. We can determine a DataFrame's shape using the shape attribute in Pandas. For instance, a DataFrame with the shape (1000, 10) indicates that it has 1000 rows and 10 columns of data.

```
data=pd.read_csv("C:/Users/Kwame Steve/Downloads/heartdisease.data",header=None)
data = data.replace("?",np.nan)
data.shape
```

```
(303, 14)
```

Figure **Error! No text of specified style in document.**-10 Shape of Dataset before Preprocessing

```
data=pd.read_csv("C:/Users/Kwame Steve/Downloads/heartdisease.data",header=None)
data = data.replace("?",np.nan)

data = data.dropna().reset_index(drop=True)
data.columns = ['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'oldpeak', 'ST slope','ca', 'thal', 'target']

k=['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'ST slope','ca', 'thal', 'target']

for j in k:
    data[j] = data[j].astype('float').astype('int')

data['oldpeak'] = data['oldpeak'].astype('float')
data['target'] = np.where(data.target>0,1,0)
dataTab = data.copy()
data.shape
```

```
(297, 14)
```

Figure **Error! No text of specified style in document.**-11 Shape of dataset after Preprocessing

From the outcome it can be inferred that the dataset originally has 303 rows and 14 columns but was dropped to 297 rows (i.e records of patients) after preprocessing.

The dataFrame's information is printed via the info() method. The data includes the total number of columns, their labels, data types, memory use, range index, and the number of cells in each column (non-null values). The info() method does indeed print the info. This is evident from the diagrams below.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 303 entries, 0 to 302
Data columns (total 14 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   0       303 non-null    float64
 1   1       303 non-null    float64
 2   2       303 non-null    float64
 3   3       303 non-null    float64
 4   4       303 non-null    float64
 5   5       303 non-null    float64
 6   6       303 non-null    float64
 7   7       303 non-null    float64
 8   8       303 non-null    float64
 9   9       303 non-null    float64
 10  10      303 non-null    float64
 11  11      299 non-null    object
 12  12      301 non-null    object
 13  13      303 non-null    int64
dtypes: float64(11), int64(1), object(2)
memory usage: 33.3+ KB
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 297 entries, 0 to 296
Data columns (total 14 columns):
 #   Column               Non-Null Count  Dtype
---  ------               --------------  -----
 0   age                  297 non-null    int32
 1   sex                  297 non-null    int32
 2   chest pain type      297 non-null    int32
 3   resting bp s         297 non-null    int32
 4   cholesterol          297 non-null    int32
 5   fasting blood sugar  297 non-null    int32
 6   resting ecg          297 non-null    int32
 7   max heart rate       297 non-null    int32
 8   exercise angina      297 non-null    int32
 9   oldpeak              297 non-null    float64
 10  ST slope             297 non-null    int32
 11  ca                   297 non-null    int32
 12  thal                 297 non-null    int32
 13  target               297 non-null    int32
dtypes: float64(1), int32(13)
memory usage: 17.5 KB
```

Figure **Error! No text of specified style in document.**-12 Details of the Dataset (Before and After Preprocessing)

It's worth to note that the dataset is made of data types namely intger, object and float with memory usage of 33.3+ kilobyte and 17.5+ kilobyte. The dataset has no null values.

```
data['target'].value_counts(dropna=False)

0    160
1    137
Name: target, dtype: int64

data['sex'].value_counts(dropna=False)

male      201
female     96
Name: sex, dtype: int64

data['fasting blood sugar'].value_counts(dropna=False)

under 120mgdl    254
over 120mgdl      43
Name: fasting blood sugar, dtype: int64

data['exercise angina'].value_counts(dropna=False)

not induced    200
induced         97
Name: exercise angina, dtype: int64
```

Figure **Error! No text of specified style in document.**-13 Value counts of Binary Variables

Another useful function to aid with the the data exploration is the value_counts function. To get a Series with counts of unique values, we use the value_counts() function. The resulting object will be arranged in descending order with the first element being the one that appears the most frequently. From the outputs below it was obtained that from the target column it has 160 patients without heart disease and a number of 137 patients with heart disease.

Sex with 201 Males and 96 females.

The Fasting Blood Sugar (FBS) count reads 254 under 120mgdl and 43 over 120mgdl.

Exercise angina value_counts reads 97 patients being induced and 200 patients not being induced.

Distribution of Data

Target Variable:

To determine whether cardiac disease is present, the target variable is employed. The value is set to 1 if the patient has cardiac disease. In all other cases, the value is set to 0 to indicate that the patient is healthy. Data are preprocessed by converting medical records into diagnostic values. For example, data preprocessing of 297 patient records resulted in records showing that 137 records had a value of 1, indicating the presence of heart disease. The remaining 160 records had a value of 0, indicating no heart disease. (Figure 4.4-6) shows the distribution of target labels.

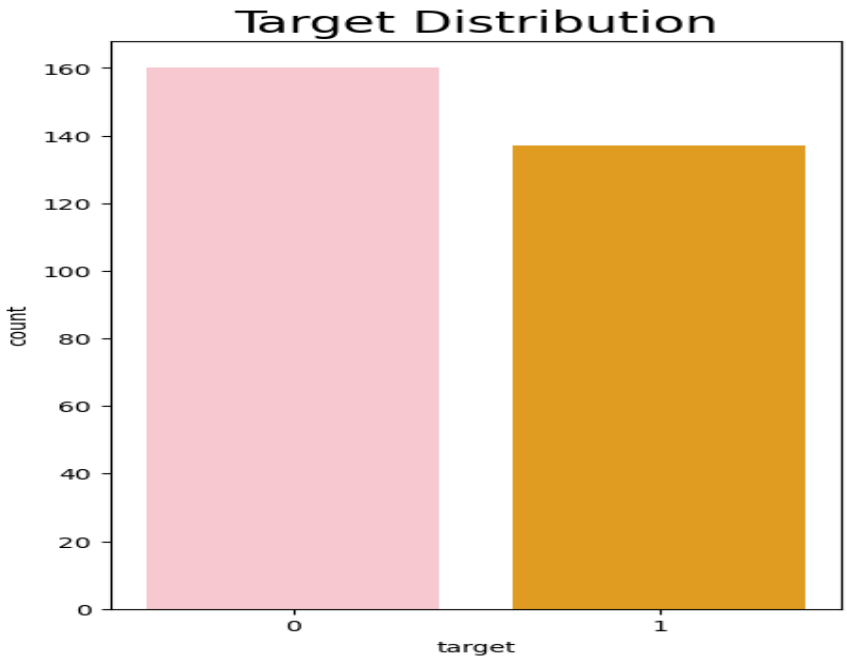

Figure **Error! No text of specified style in document.**-14 Target distribution

Binary Variables:

Binary variables include sex, fasting blood sugar, and exercise-induced angina. The blood arteries that govern the heart can get damaged by high blood sugar levels, according to the Centers for Disease govern. Numerous factors that raise the risk of heart disease are more common in diabetic patients. For example, angina is chest pain caused by exercise, stress, or other factors that make the heart work harder. This is a prevalent symptom of heart disease caused by the coronary arteries clogged with cholesterol. (Figure 4.4-7) shows the distribution of binary variables.

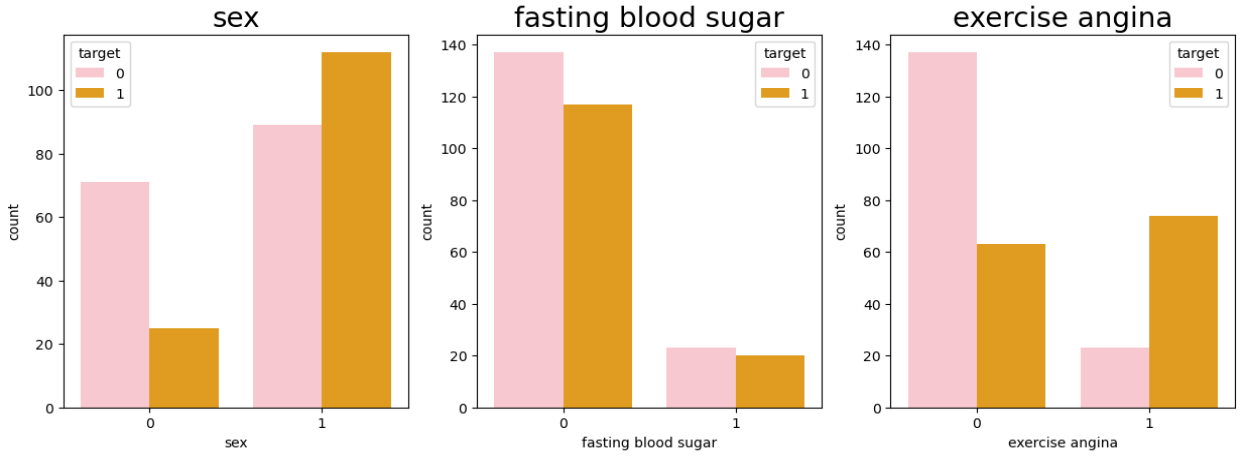

Figure **Error! No text of specified style in document.**-15 Binary Variables Distribution

Chest Pain Type:

Chest pain and discomfort are the most common symptoms of heart disease. Chest pain can occur when an artery is narrowed by excess plaque buildup. A narrowed artery can block blood flow to the heart muscles, which can cause chest pain. The diagram shows the data distribution for chest pain types. For example, chest pain distribution is shown in (Figure 4.4-8).

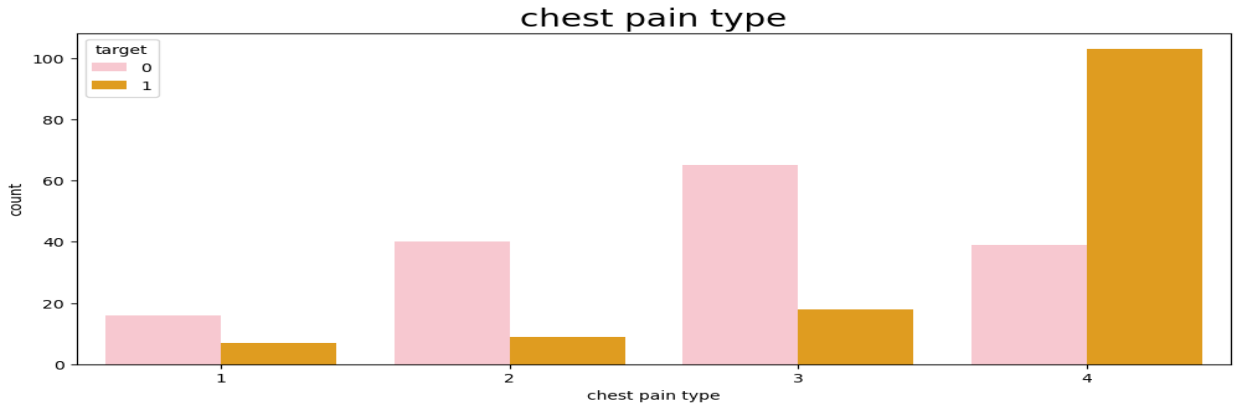

Figure **Error! No text of specified style in document.**-16 chest pain Distribution

ST Slope:

On the ECG, the ST segment connects the QRS complex and T wave and lasts 0.005 to 0.150 seconds. The regular ST segment is slightly concave upwards. Therefore, a flat, downsloping, or sunken ST segment may indicate coronary artery disease. The ST slope distribution is shown in (Figure 4.4-9).

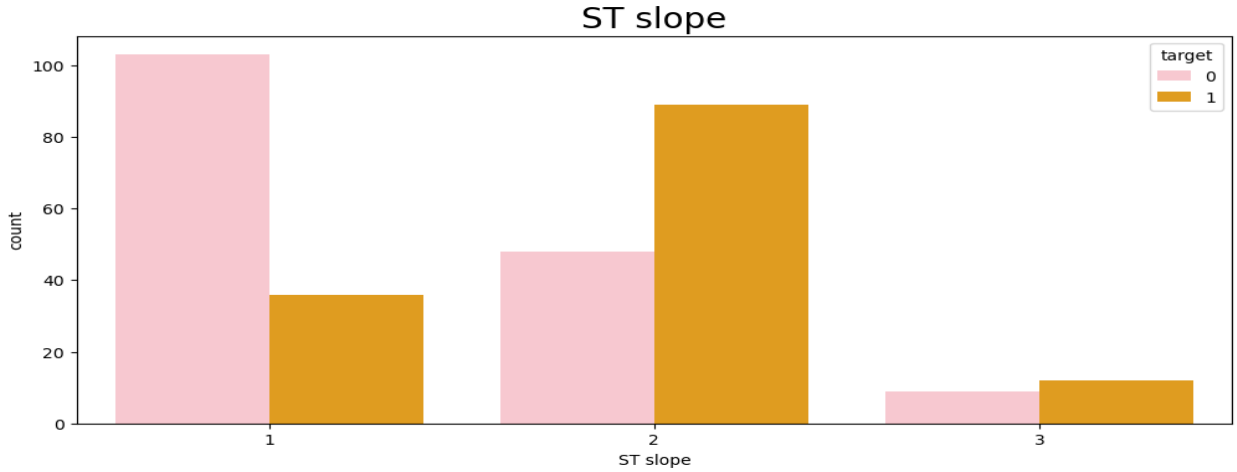

Figure **Error! No text of specified style in document.**-17 ST slope Distribution

Resting ECG:

A resting ECG is a standard test that measures the heart's electrical function. An ECG can be used as routine testing to check for heart disease before signs or symptoms appear. For example, resting 12-lead ECG can detect abnormalities such as arrhythmia, evidence of coronary artery disease, left ventricular hypertrophy, and bundle branch block. The resting ECG distribution is shown in (Figure 4.4-10).

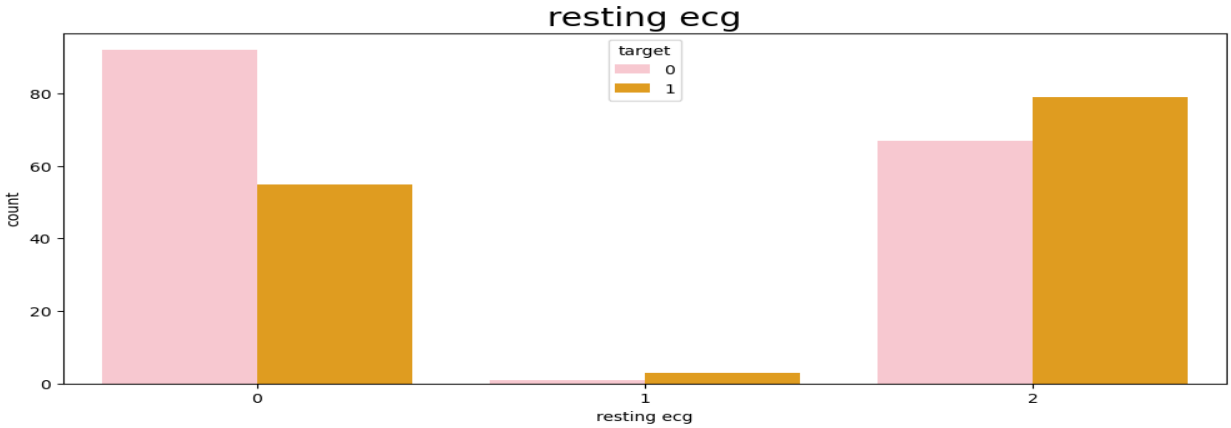

Figure **Error! No text of specified style in document.**-18 resting ecg Distribution

Ca:

Fluoroscopy shows how blood flows through the coronary arteries and assesses whether a route is blocked. Data distribution of the number of colored arteries is shown in (Figure 4.4-11).

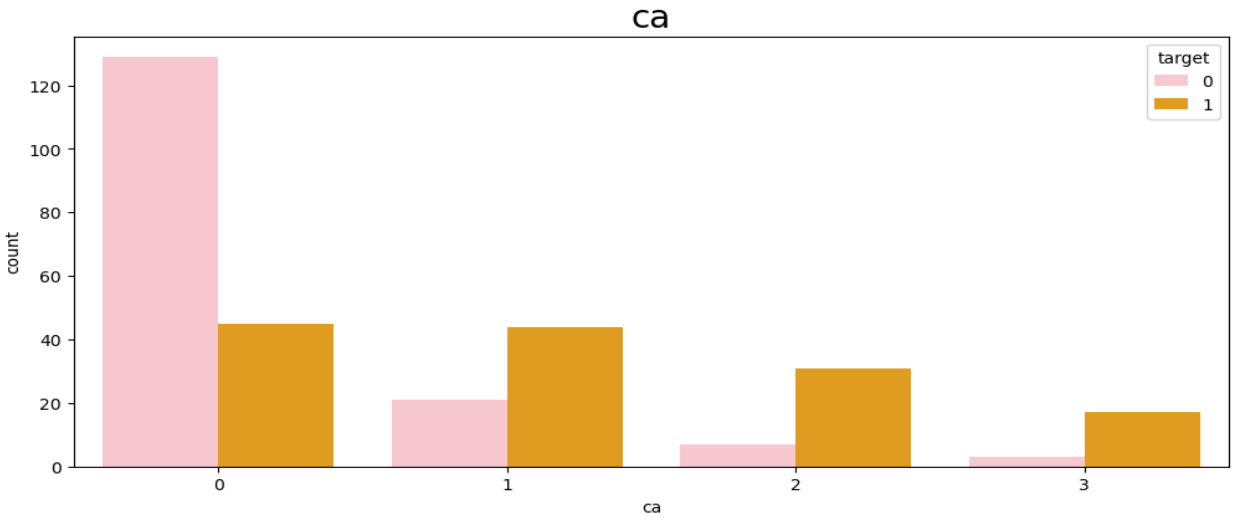

Figure **Error! No text of specified style in document.**-19 ca Distribution

Thal: The outcomes of the thallium stress test are represented by the "thal" feature in our dataset. It often contains a number of discrete numbers, each of which represents a separate test result.

These results provide important details regarding the patient's cardiovascular health and can be very telling as to whether or not a patient has heart disease. Status of the hearts illustrated through three distinctively numbered values. Normal number as 3, Fixed defect as 6, Reversible defects as 7 which is shown in (Figure 4.4-12).

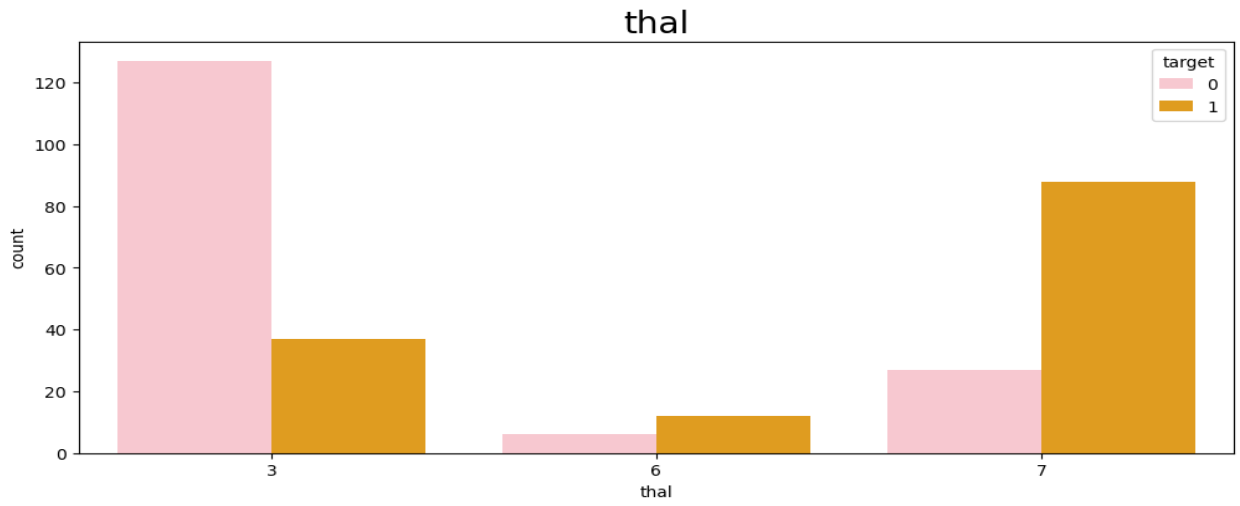

Figure **Error! No text of specified style in document.**-20 thal distribution

Age: Age-related changes can increase the risk of heart disease. According to the National Institutes of Health, people over the age of 65 are much more likely to have a heart attack or develop coronary artery disease or heart failure. Data distribution based on age is indicated in (Figure 4.4-13).

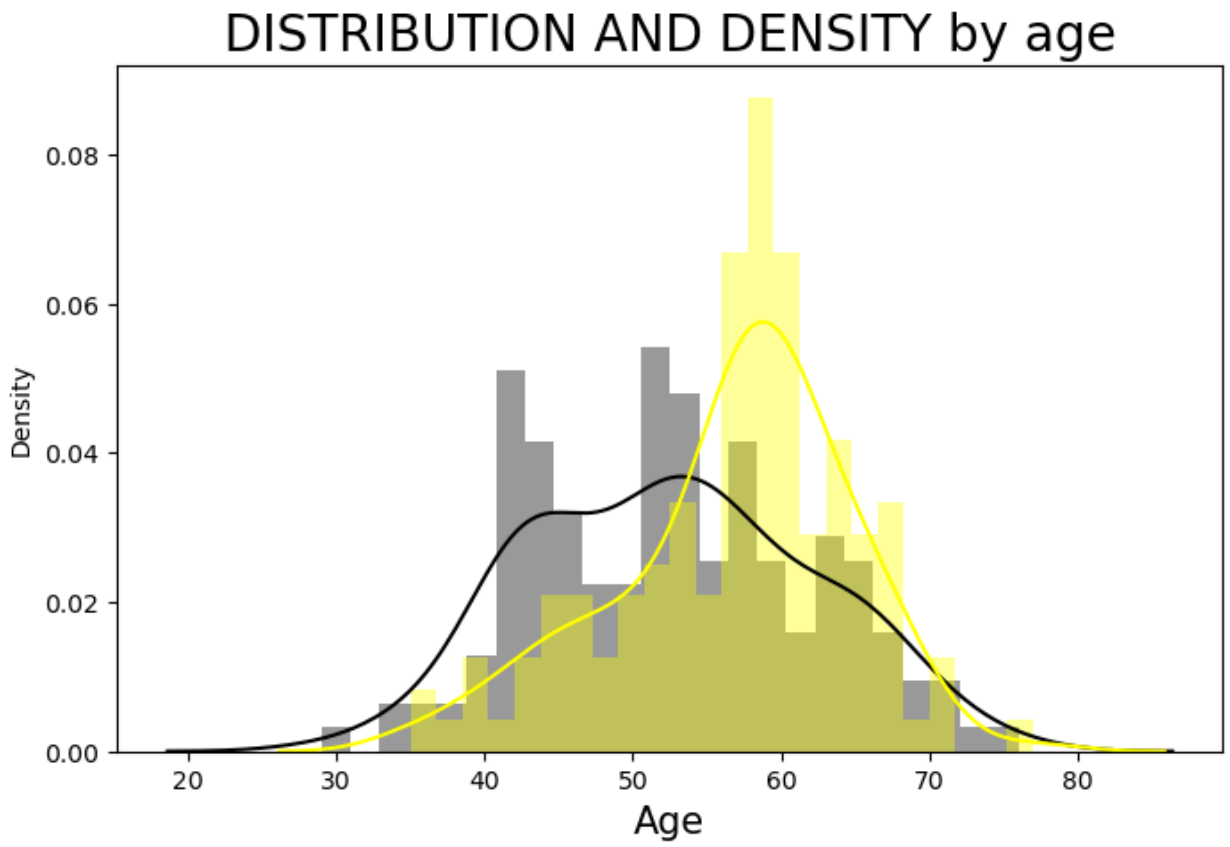

Figure **Error! No text of specified style in document.**-21 Distribution and Density by Age

Cholesterol: Cholesterol helps the body grow new cells, protect nerves, and produce hormones. Typically, the liver makes all the cholesterol the body needs. However, cholesterol also enters the body from animal foods such as milk, eggs, and meat. Too much cholesterol in the body is a risk factor for heart disease. As a result, arteries narrow, slowing or blocking blood flow to the heart

muscle. A heart attack occurs when a blockage completely prevents blood supply to part of the heart. The density distribution for cholesterol is shown in (Figure 4.4-14).

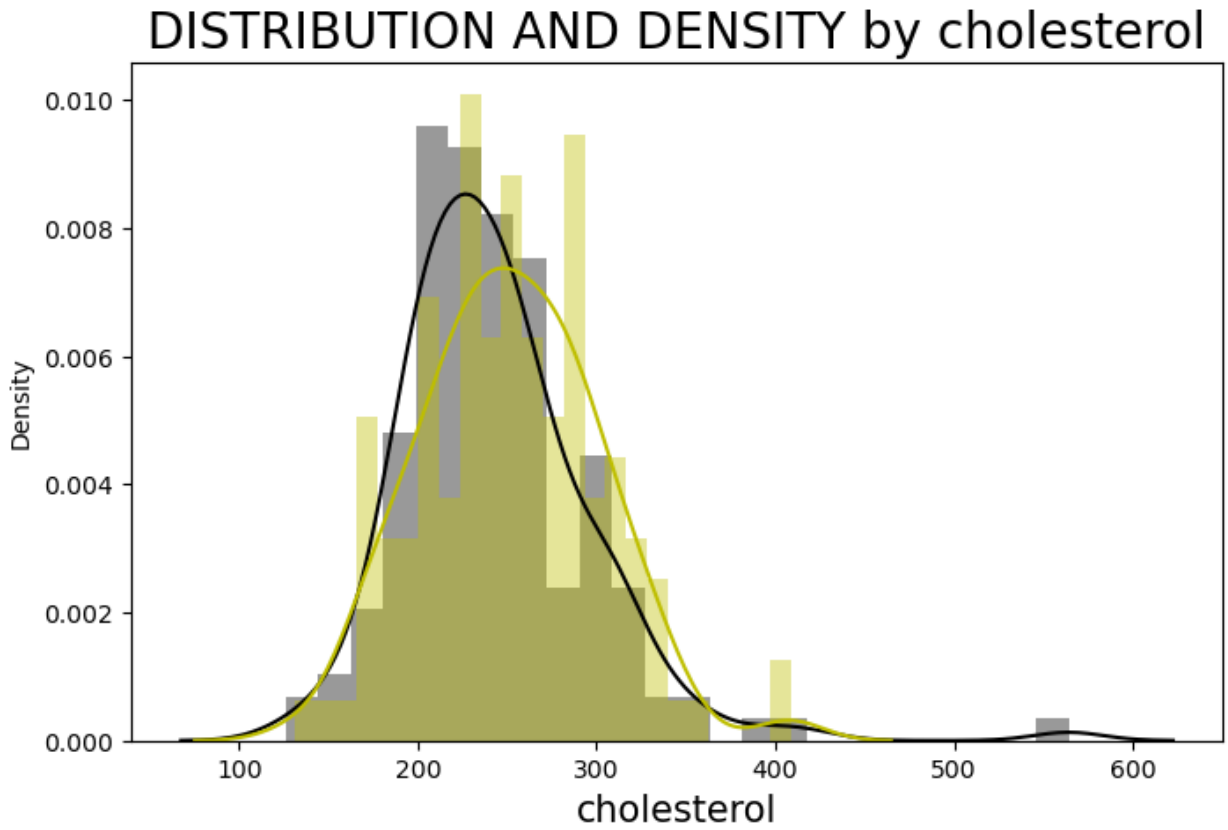

Figure **Error! No text of specified style in document.**-22 Distribution and Density by cholesterol

Resting BP: The "resting BP" function, which measures blood pressure at rest, is an important piece of information when it comes to predicting heart disease. An important physiological measure that might serve as a sign of good cardiovascular health is blood pressure. The term "resting BP" describes a blood pressure reading recorded when the patient is at rest. Systolic and

diastolic blood pressure measurements are shown. Systolic blood pressure is a measure of the force exerted on the arteries when the heart contracts and diastolic blood pressure is a measure of the force exerted when the heart is at rest. The units used to measure these distances are millimeters of mercury (mm Hg). (Figure 4.4-15) displays the density and distribution for resting bp.

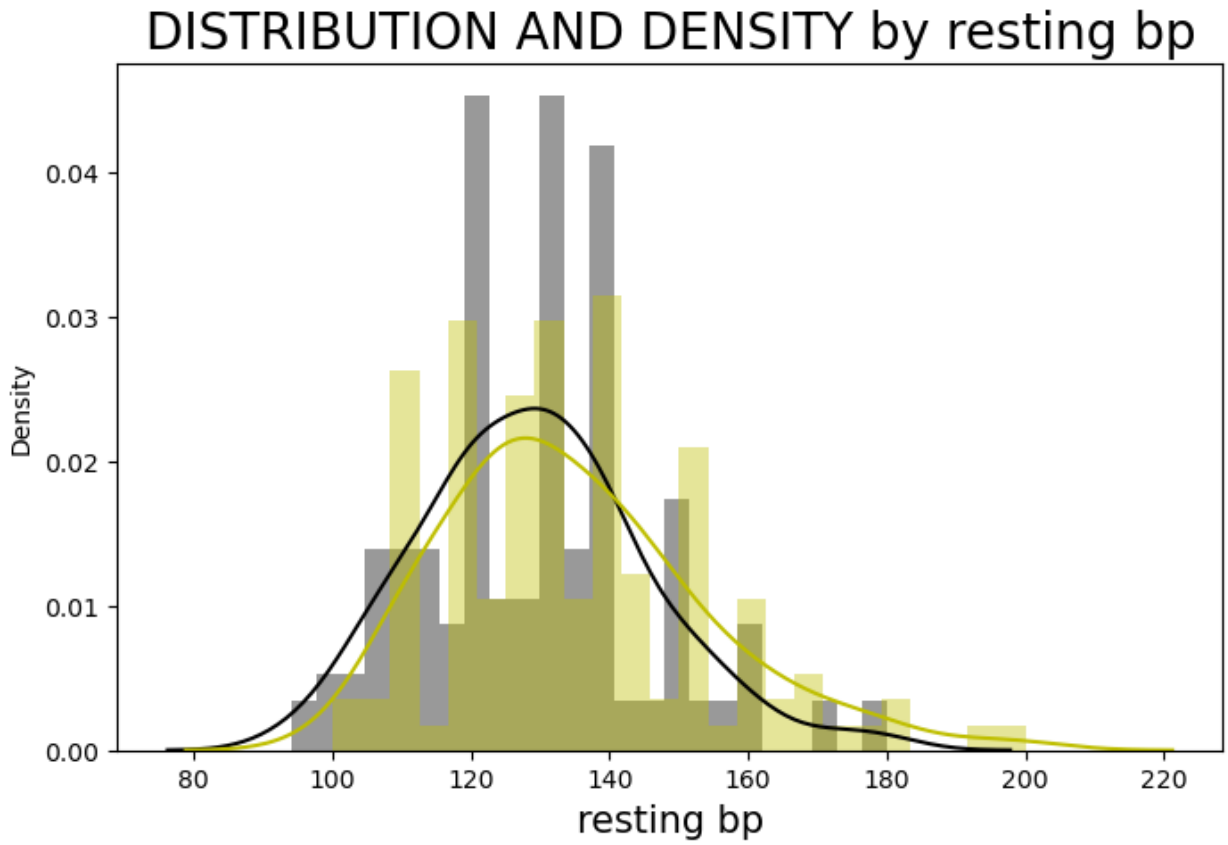

Figure **Error! No text of specified style in document.**-23 Distribution and Density by restingbp

Thalach: "Thalach" is an acronym for "maximum heart rate achieved" when working out. This feature records the patient's greatest heart rate during an exercise program or cardiovascular stress test. It is a key physiological indicator of someone's cardiovascular health and capacity for exercise. Maximum heart rate achieved during exercise is a direct indicator of an individual's

exercise capacity. Lower "thalach" values may indicate reduced cardiovascular fitness, which can be associated with an increased risk of heart disease as shown in (Figure 4.4-16).

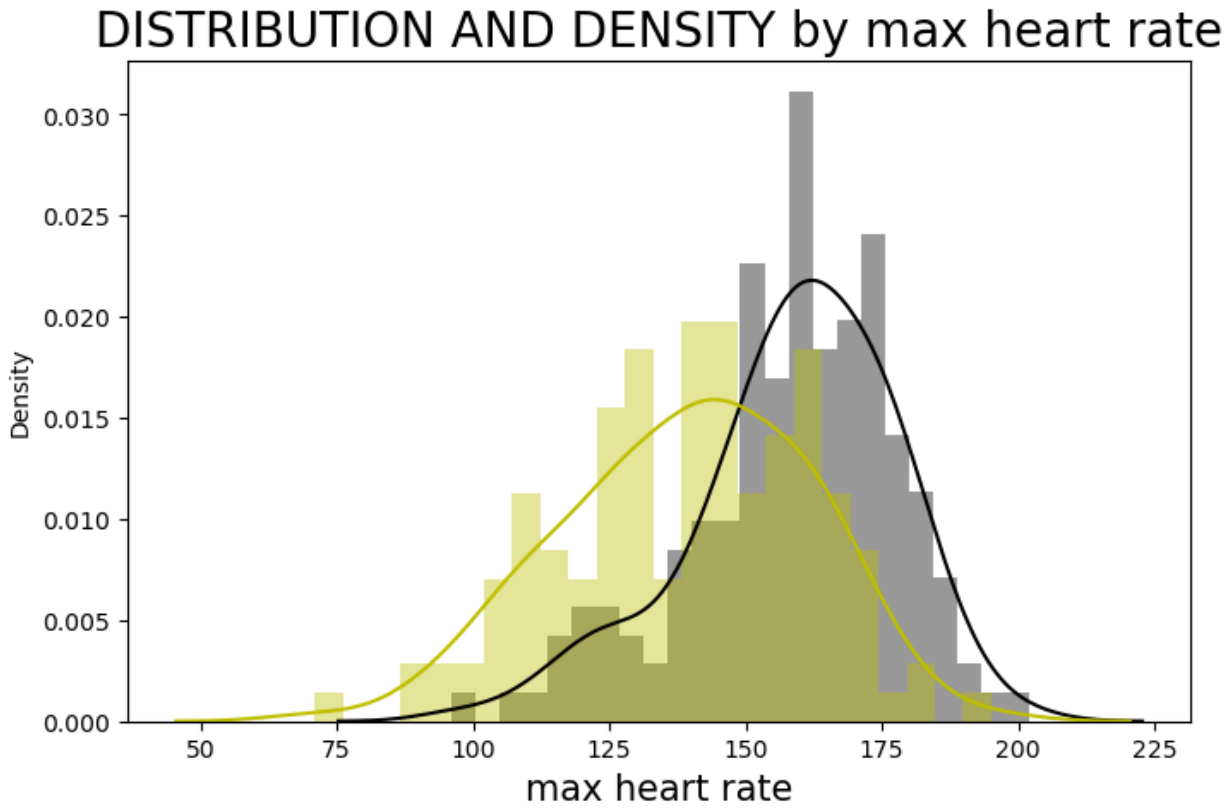

Figure **Error! No text of specified style in document.**-24 Distribution and Density by max heart rate

Old Peak: The term "Oldpeak" describes the ST depression brought on by activity in comparison to rest, which is commonly measured via an electrocardiogram (ECG) during a cardiac stress test. It determines the degree of myocardial ischemia (insufficient blood flow to the heart muscle), which is measured as the ST segment of the ECG deviates from baseline during activity. "Oldpeak"

serves as a direct indicator of myocardial ischemia. Higher "oldpeak" values are often associated with more severe ischemic conditions, which can be indicative of coronary artery disease (CAD) or other cardiac issues. The distribution and density by oldpeak is shown in (Figure 4.4.17).

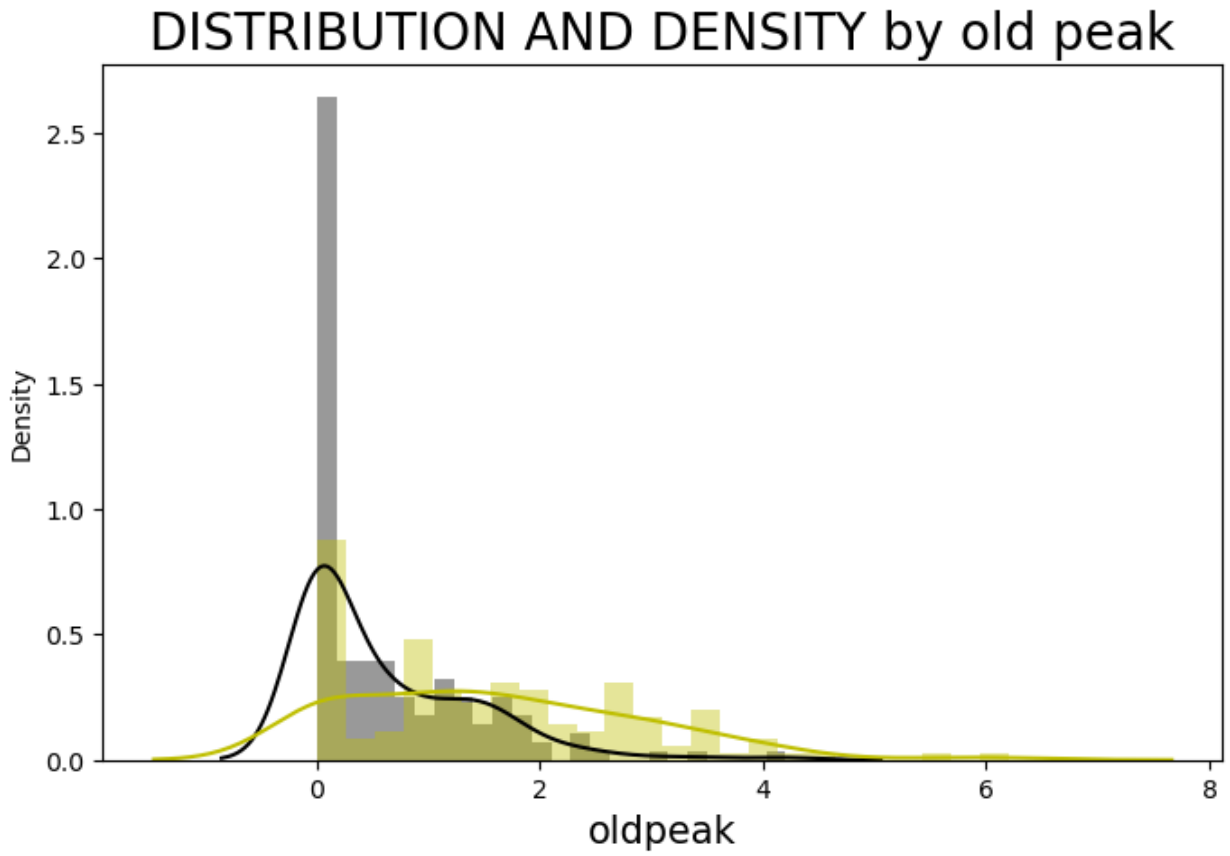

Figure **Error! No text of specified style in document.**-25 Distribution and Density by oldpeak

Results

Results (Base Models)

To evaluate the performance of models, we used unseen test data to predict the heart disease labels. The test set contains 36 patients randomly sampled from the full dataset with no patient overlap with the train set. Results are shown in (Table 4-1).

Model	AUROC	Accuracy	Precision	Recall	F1_Score
Logistic Regression	0.90	91.7%	1.00	0.80	0.8889

Random Forest	0.88	88.9%	0.92	0.80	0.8517
XGBoost	0.84	86.1%	0.92	0.73	0.8148
Gradient Boosting	0.79	80.6%	0.83	0.67	0.7407

Table **Error! No text of specified style in document.**-3 Base Model Performance

Logistic regression achieved the best performance among the base models based on the overall test metrics. This model achieved a ROC score of 0.90 and an accuracy of 91.7%. The other models had ROC values between 0.79 and 0.88 and accuracies between 81% and 86%. The confusion matrix shown in (Figure 4.5-1) shows the sensitivity and specificity of different models. Logistic regression correctly predicted disease identifiers in 33 cases from a test set of 36 cases.

Before calculating for the various classification-based models' accuracy, precision, recall, and the F1_score, the TP, FP, TN, FN must be known. Hence, the following pinpoints the positive and negative values for each classification model.

Logistic Regression:

TN = 21	FP = 0
FN = 3	TP = 12

Random Forest:

TN = 20	FP = 1
FN = 3	TP = 12

XGBoost:

TN = 20	FP = 1
FN = 4	TP = 11

Gradient Boosting:

TN = 19	FP = 2
FN = 5	TP = 10

Calculating for Accuracy

Accuracy is used to measure how well a binary classification test identifies or excludes conditions. In other words, accuracy is the ratio of correct predictions out of the total number of cases tested.

$$\frac{TP + TN}{TP + FP + TN + FN}$$

Formula for finding Accuracy[86]

Logistic Regression:

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{(12 + 21)}{(12 + 0 + 21 + 3)} = \frac{33}{36} = 0.917$$

Random Forest:

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{(12 + 20)}{(12 + 1 + 20 + 3)} = \frac{32}{36} = 0.889$$

XGBoost:

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{(11 + 20)}{(11 + 1 + 20 + 4)} = \frac{31}{36} = 0.861$$

Gradient Boosting:

$$\frac{TP + TN}{TP + FP + TN + FN} = \frac{(10 + 19)}{(10 + 2 + 19 + 5)} = \frac{19}{36} = 0.806$$

Calculating for Precision

Precision is the ratio of the true positives to the total of true positives and false positives. Precision describes the classifier's ability not to flag negative samples as positive.

$$\frac{TP}{(TP + FP)}$$

Formula for finding Precision[86]

Logistic Regression:

$$\frac{TP}{(TP + FP)} = \frac{12}{(12 + 0)} = 1.0$$

Random Forest:

$$\frac{TP}{(TP + FP)} = \frac{12}{(12 + 1)} = 0.92$$

XGBoost:

$$\frac{TP}{(TP + FP)} = \frac{11}{(11 + 1)} = 0.92$$

Gradient Boosting:

$$\frac{TP}{(TP + FP)} = \frac{10}{(10 + 2)} = 0.83$$

Calculating for Recall

Recall is the ratio of the true positives to the total of true positives and false negatives. Recall indicates the classifier's ability to find all positive samples.

$$\frac{TP}{(TP + FN)}$$

Formula for finding Recall[86]

Logistic Regression:

$$\frac{TP}{(TP + FN)} = \frac{12}{(12 + 3)} = 0.80$$

Random Forest:

$$\frac{TP}{(TP + FN)} = \frac{12}{(12 + 3)} = 0.80$$

XGBoost:

$$\frac{TP}{(TP + FN)} = \frac{11}{(11 + 4)} = 0.73$$

Gradient Boosting:

$$\frac{TP}{(TP+FN)} = \frac{10}{(10+5)} = 0.67$$

Calculating for F1_Score

The F1 score is an alternative machine learning evaluation metric that analyses a model's class-wise performance rather than its overall performance as done by accuracy to determine its predictive power. The F1 score combines two metrics that are in conflict: a model's precision and recall scores, which has led to its extensive adoption in recent research.

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + 1/2(FP+FN)}$$

Formula for finding F1_Score[89]

Logistic Regression:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2(FP+FN)}} = \frac{12}{12 + \frac{1}{2(0+3)}} = \frac{12}{13.5} = 0.8889$$

Random Forest:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2(FP+FN)}} = \frac{12}{12 + \frac{1}{2(1+3)}} = \frac{12}{14} = 0.8571$$

XGBoost:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2(FP+FN)}} = \frac{11}{11 + \frac{1}{2(1+4)}} = \frac{11}{13.5} = 0.8148$$

Gradient Boosting:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2(FP + FN)}} = \frac{10}{10 + \frac{1}{2(2+5)}} = \frac{10}{13.5} = 0.7407$$

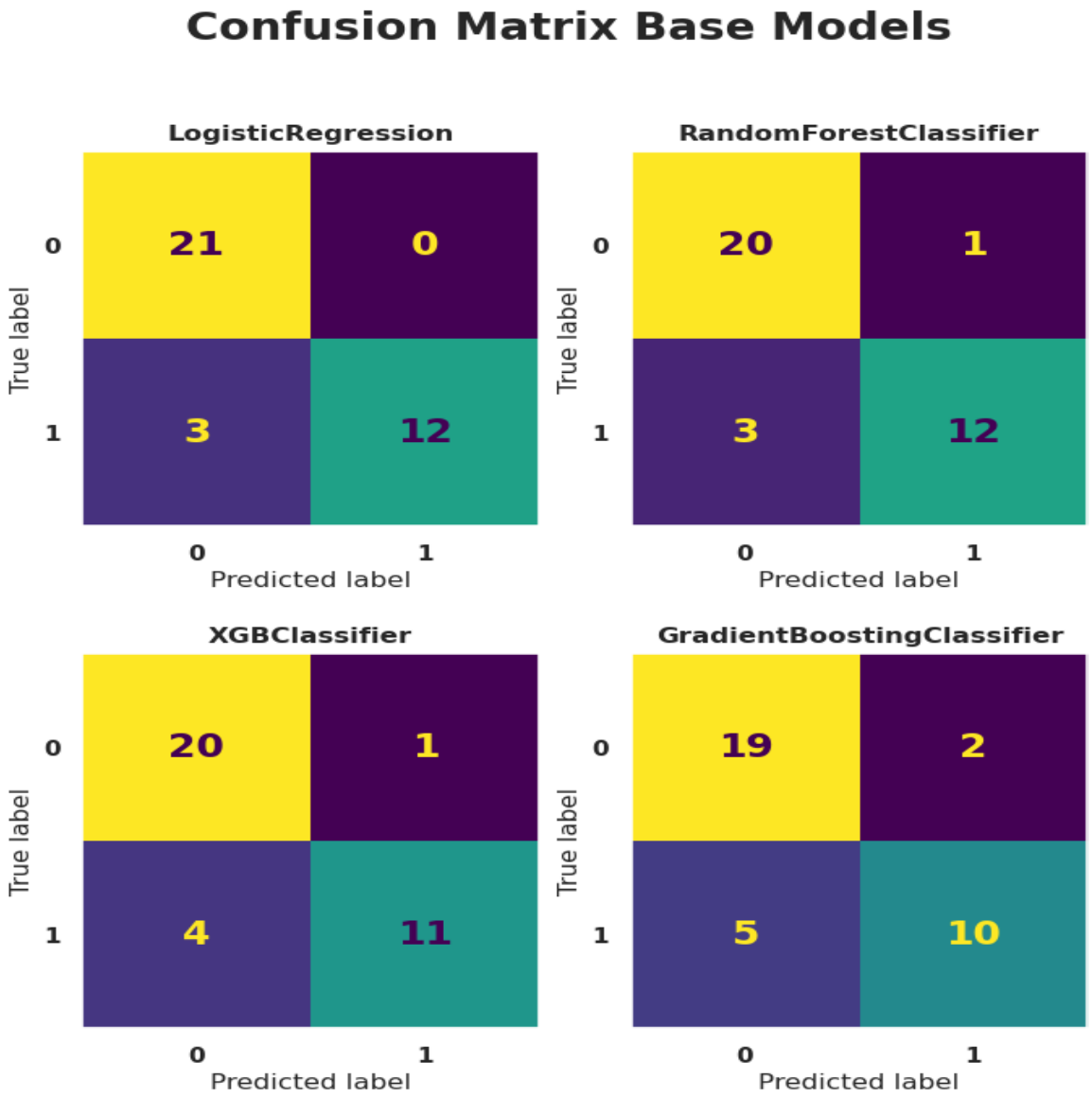

Figure **Error! No text of specified style in document.**-26 Confusion Matrix for Base models

Results (TabNet Classifier)

TabNet inputs raw tabular data and is trained using gradient descent-based optimization, allowing flexible integration into end-to-end learning. The TabNet classifier achieved a ROC of 0.94 and an accuracy of 94%, much better than the base models. Sensitivity and specificity exceed 0.93.

Model	AUROC	Accuracy	Precision	Recall	F1_Score
TabNet	0.9429	94.4%	0.933	0.933	0.9333

Table **Error! No text of specified style in document.**-4 TabNet Performance

Before calculating for the TabNet classifier's accuracy, precision, recall, and the f1_score, the TP, FP, TN, FN must be known. Hence, the following pinpoints the positive and negative values for each classification model.

TN = 20	FP = 1
FN = 1	TP = 14

Calculating for TabNet Accuracy:

$$\frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN}}$$

Formula for finding Accuracy[86]

$$\frac{\boldsymbol{TP} + \boldsymbol{TN}}{\boldsymbol{TP} + \boldsymbol{FP} + \boldsymbol{TN} + \boldsymbol{FN}} = \frac{(\mathbf{14} + \mathbf{20})}{(\mathbf{14} + \mathbf{1} + \mathbf{20} + \mathbf{1})} = \frac{\mathbf{34}}{\mathbf{36}} = \mathbf{0.944}$$

Calculating for TabNet Precision:

$$\frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})}$$

Formula for finding Precision[86]

$$\frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FP})} = \frac{\mathbf{14}}{(\mathbf{14} + \mathbf{1})} = \mathbf{0.933}$$

Calculating for TabNet Recall:

$$\frac{\boldsymbol{TP}}{(\boldsymbol{TP} + \boldsymbol{FN})}$$

Formula for finding Recall[86]

$$\frac{TP}{(TP+FN)} = \frac{14}{(14+1)} = 0.933$$

Calculating for F1_Score of TabNet Classifier:

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + 1/2(FP+FN)}$$

Formula for finding F1_Score[89]

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2(FP+FN)}} = \frac{14}{14 + \frac{1}{2(1+1)}} = \frac{14}{15} = 0.9333$$

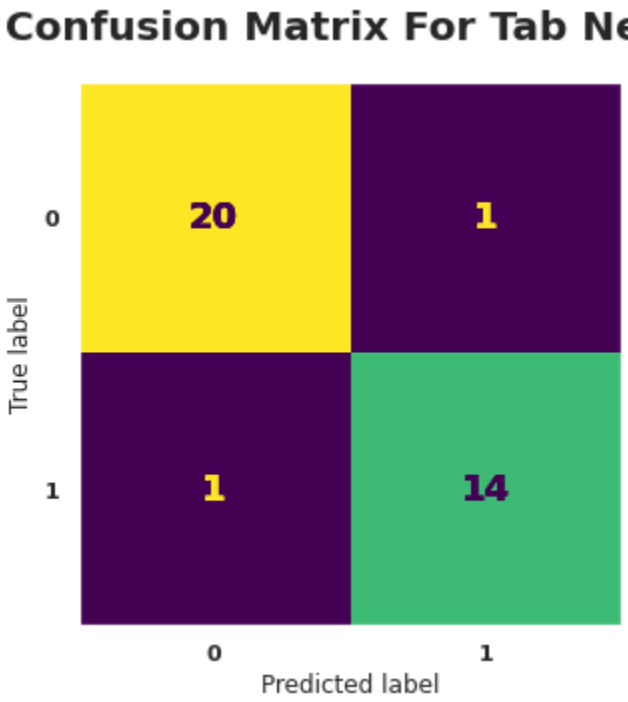

Figure **Error! No text of specified style in document.**-27 Confusion Matrix for TabNet

The TabNet model properly detected labels for 34 out of the 36 instances in the test set. Also accurately identified were 20 true negative instances and 14 true positive cases. (Table 4-2) displays the results for the TabNet model. (Figure 4.5-2) displays the confusion matrix.

Model Training for TabNet

We randomly split the training data by 30% as validation and test data. TabNet has been trained up to 1000 epochs. Early stopping was achieved at 126 epochs. ROC and accuracy were used to determine early stop criteria. A list of categorical feature indices was supplied to the TabNet

classifier. Adam was used as the Pytorch optimizer function with an initial learning rate of 0.01. The “sparsemax” masking function was used for feature selection.

Figure **Error! No text of specified style in document.**-28 TabNet Training loss

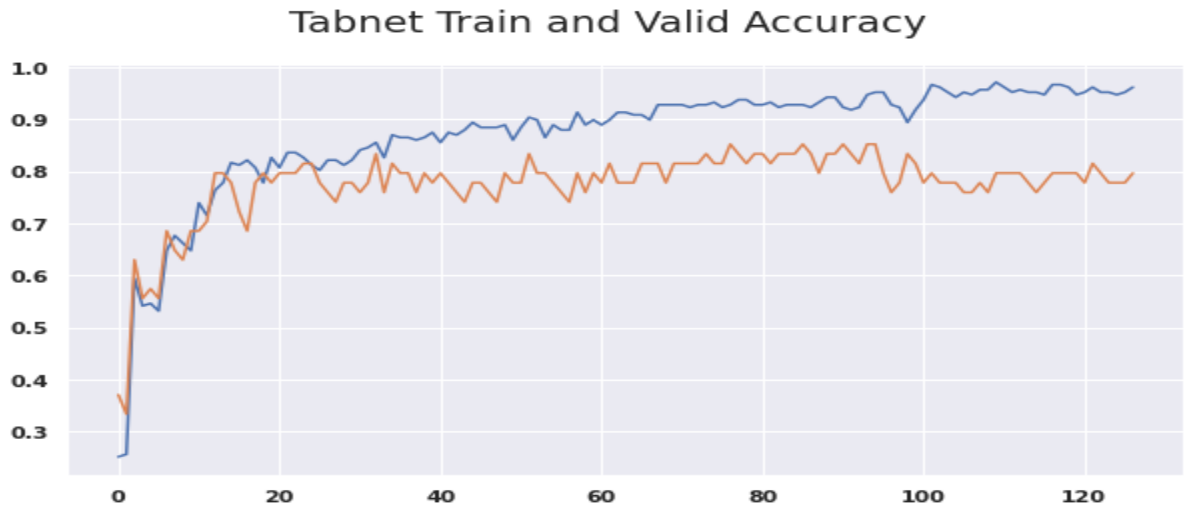

Figure **Error! No text of specified style in document.**-29 TabNet Train and Valid Accuracy

Analysis

The TabNet model outperformed the other base models. Accuracy for various models is shown in (Figure 4.6-1).

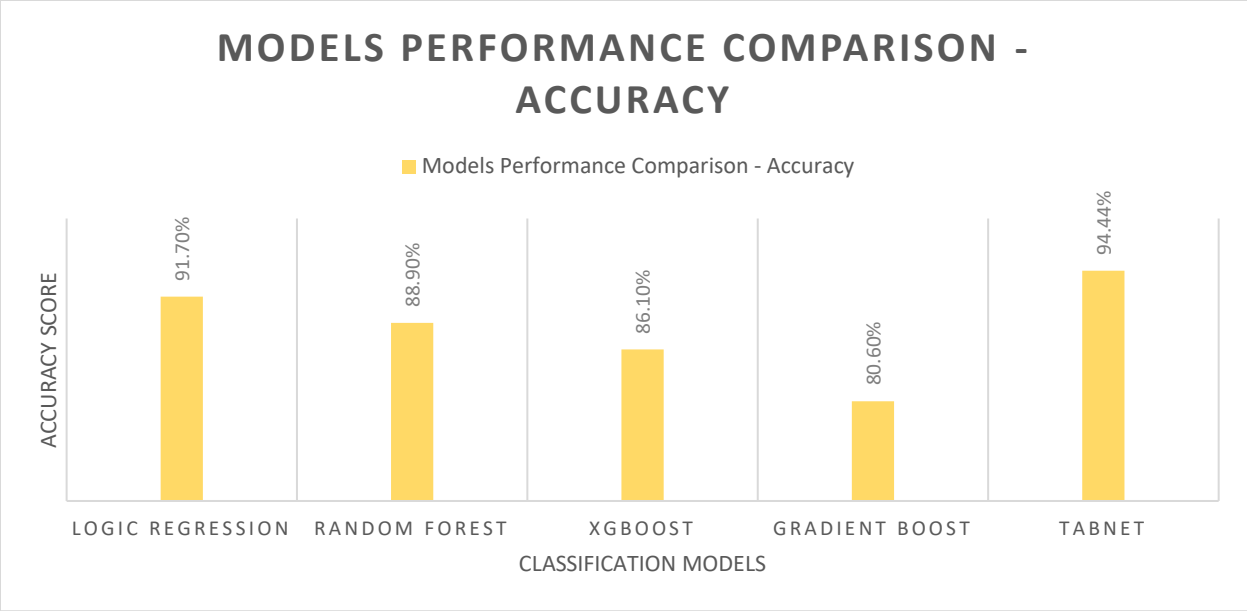

Figure **Error! No text of specified style in document.**-30 Model Performance – Accuracy.

A comparison of ROC scores is shown in (Figure 4.6-2).

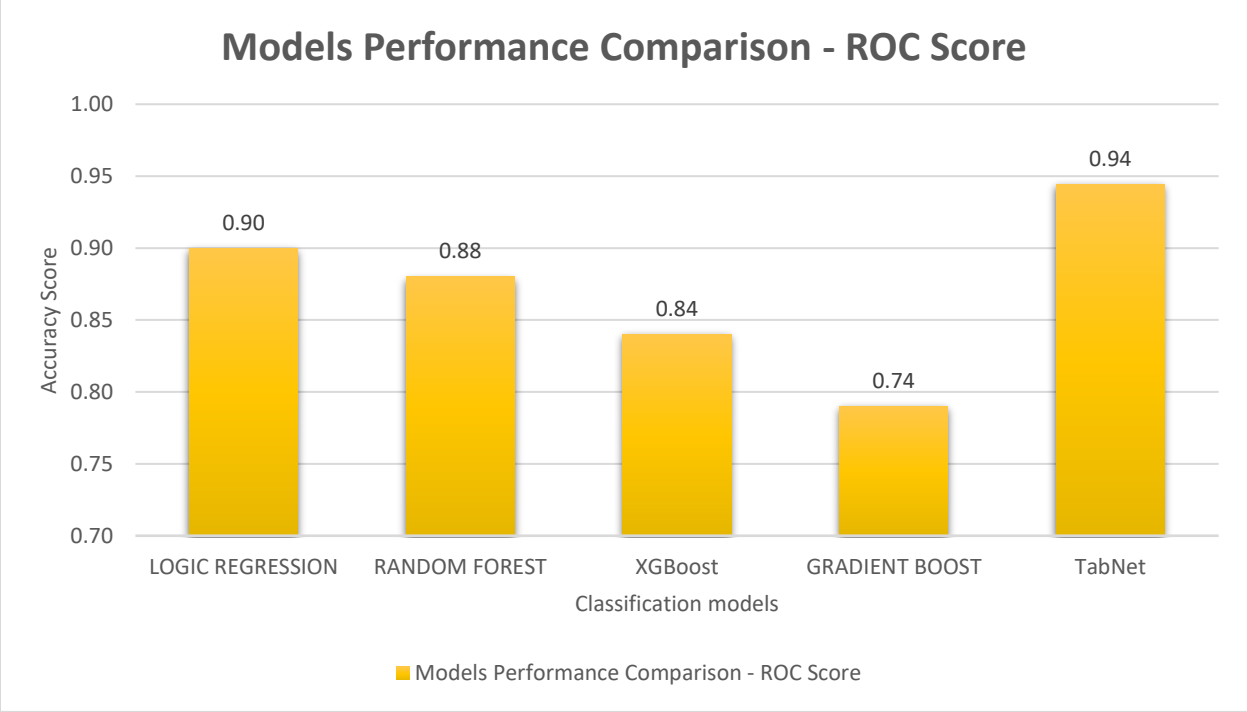

Figure **Error! No text of specified style in document.**-31 Model Performance – ROC Score.

Scatter Plot

The scatter plot for Age and Max Heart Rate in the UCI Cleveland Heart Disease dataset is a visual representation that provides insights into how a patient's age and their maximum heart rate during

exercise relate to the presence or absence of heart disease. This particular scatter plot uses color-coded points, with red dots indicating the presence of heart disease and blue dots indicating the absence of heart disease as shown in (Figure 4.6-3).

The x-axis represents the age of patients in the dataset where by age values range from 30 to 80 years, covering a broad age range of patients. The y-axis represents the maximum heart rate (Max HR) achieved by each patient during exercise. Max HR values range from 80 to 200 beats per minute (BPM). Each point on the scatter plot corresponds to an individual patient in the dataset. The position of each point is determined by the patient's age (x-coordinate) and their maximum heart rate (y-coordinate).

The scatter plot uses color-coding to distinguish between patients with and without heart disease: Red Dots: Patients with heart disease are represented by red dots. Blue Dots: Patients without heart disease are represented by blue dots.

By examining the scatter plot, the Correlation between age and max heart rate do implies that older patients tend to have lower maximum heart rates. The presence of outliers appears as individuals with red or blue dot far away from the main clusters indicates unique cases that deviates from the general trend.

A clear pattern emerges as individuals between the ages of 40 and 65 with max heart rate are associated with a higher likelihood of heart disease. Hence, Age and max heart rate can be considered as features in models to predict heart disease risk.

Figure **Error! No text of specified style in document.**-32 Scatter plot for Age and max heart rate

Heatmap

The heatmap for the UCI Cleveland Heart Disease dataset, which includes 14 features (age, sex, chest pain, resting blood pressure, cholesterol, fasting blood sugar, resting electrocardiogram results, maximum heart rate, exercise-induced angina, old peak, ST slope, number of major vessels colored by fluoroscopy (ca), thallium stress test results (thal), and the target variable), is a visual representation that provides insights into the relationships between these features using color coding.

The heatmap displays the names of all 14 features on both the x-axis and the y-axis. These features are the variables in your dataset. Each cell in the heatmap is color-coded to represent the strength and direction of the relationship between features. The color scale typically ranges from a cool color (i.e. blue) indicating a negative correlation to a warm color (i.e. red) indicating a positive correlation.

The numbers within each cell of the heatmap represent the correlation coefficients between pairs of features. These coefficients quantify the degree and direction of the linear relationship between two variables. Common correlation coefficients used in this paper is Pearson's correlation coefficient (for continuous variables) and point-biserial correlation (for one binary and one continuous variable).

Interpretation of Heatmap

Positive Correlation: If two features have a positive correlation, the corresponding cell will be a warm color (e.g., red). This means that as one feature increases, the other tends to increase as well.

Negative Correlation: If two features have a negative correlation, the cell will be a cool color (e.g., blue). This implies that as one feature increases, the other tends to decrease.

No Significant Correlation: If the cell is close to white or a very pale color, it suggests little to no correlation between the two features.

The diagonal line running from the top left to the bottom right of the heatmap represents the correlation of each feature with itself, which is always a perfect positive correlation (1.0). Therefore, these cells are often shaded in the warmest color.

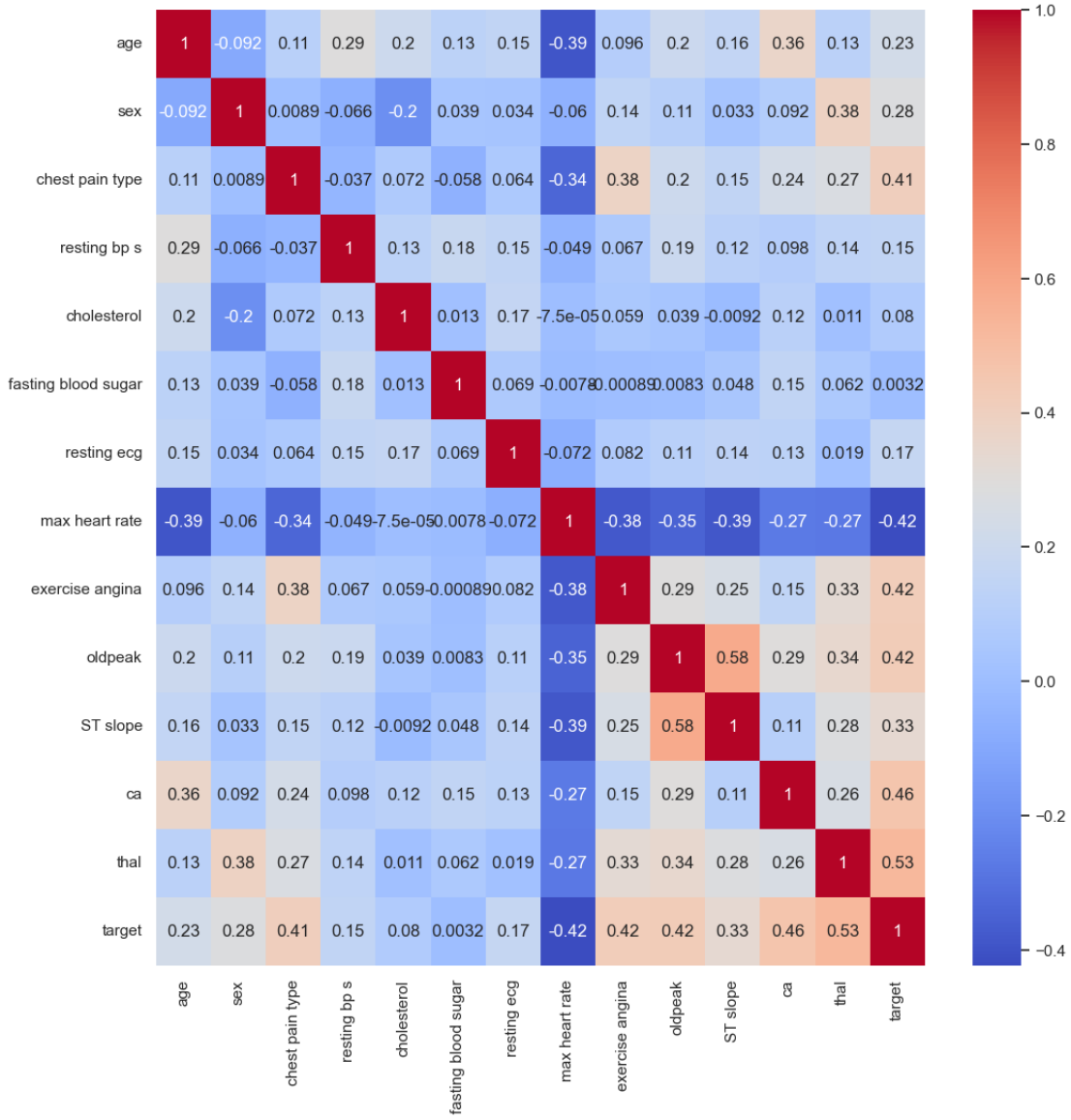

Figure **Error! No text of specified style in document.**-33 Heatmap

Feature Importance

The process of scoring input characteristics according to how well they can predict the target variable is known as Feature Importance[97]. Access to feature rankings based on overall relevance is available through TabNet. In predictive modeling, feature importance is essential for providing understanding of the data and models. Feature importance for the Tabnet classifier is shown in (figure 4.6-5). Features ca, thal, and oldpeak contributed the most to predicting target labels.

Figure **Error! No text of specified style in document.**-34 Relative Importance.

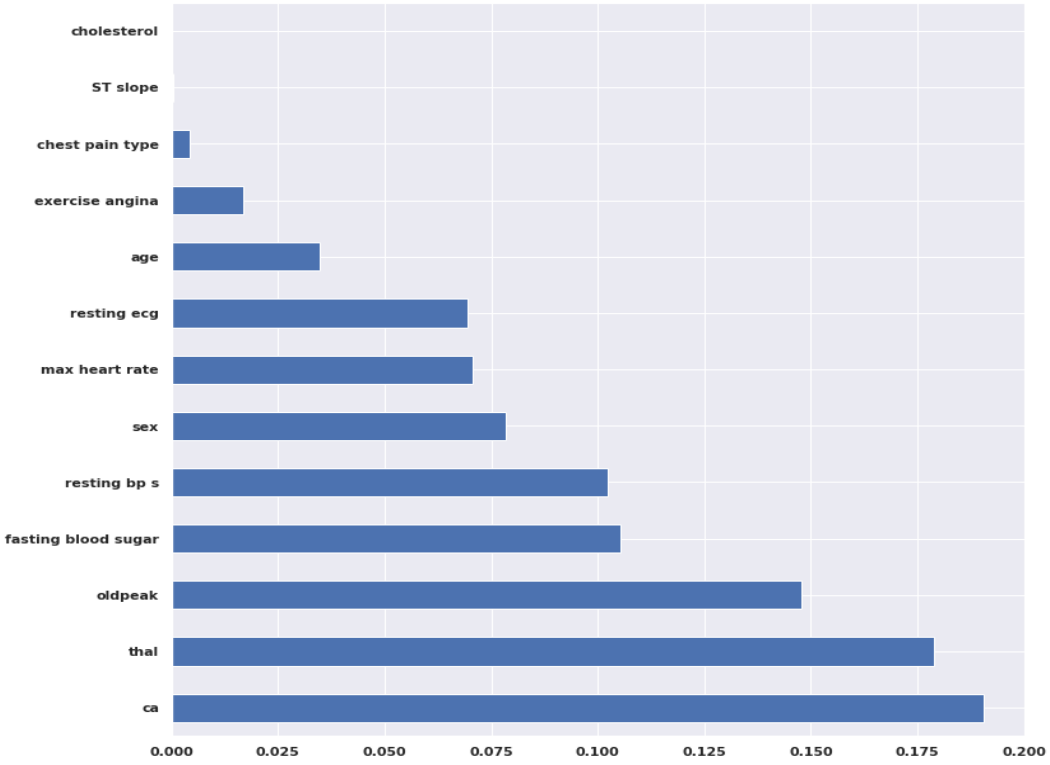

Figure **Error! No text of specified style in document.**-35 Feature Importance.

In addition to predicted values, TabNet also provides a feature importance output mask that indicates whether a feature is selected at a particular decision step in the model. The mask can be used to retrieve feature importance. The prediction output returns the aggregated mask value, as shown in (Figure 4.6-7).

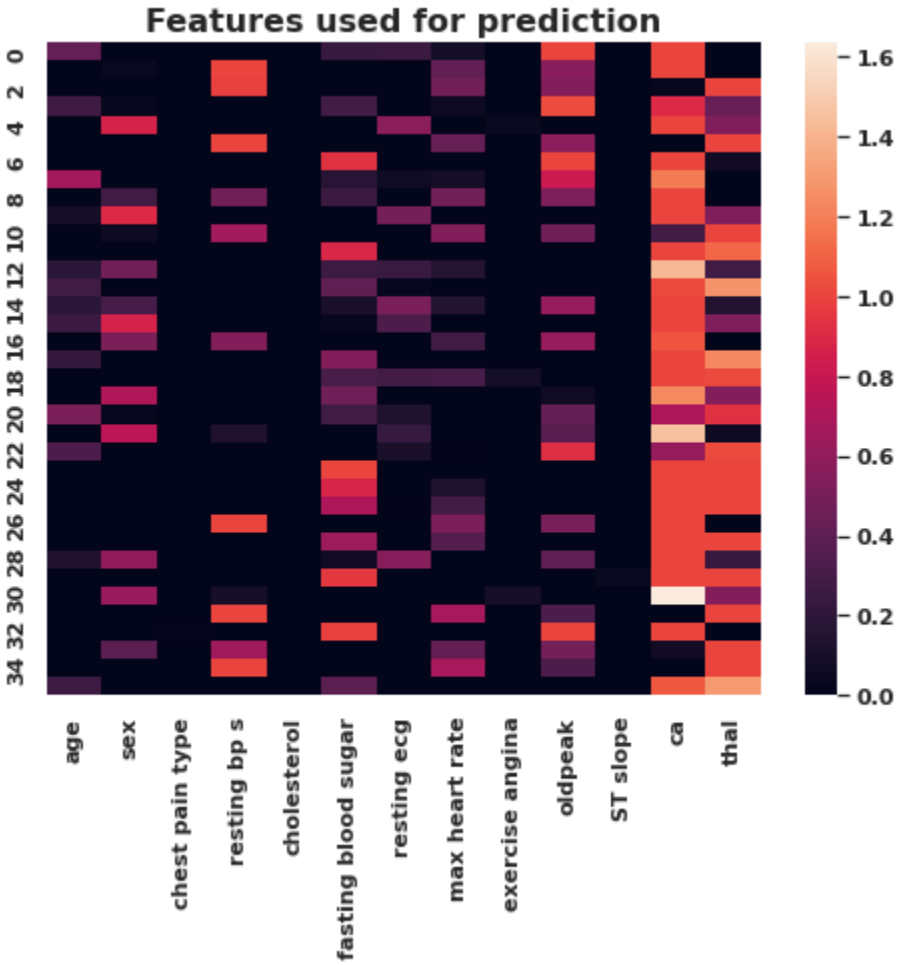

Figure **Error! No text of specified style in document.**-36 Feature importance masks for Tabnet.

This is most useful for explaining the model. The higher the mask value for a particular sample, the more critical the corresponding feature is. Brighter colors have higher values. Each row represents a mask for each input.

CHAPTER FIVE

Conclusion

Long-term lifesaving and early detection of heart disease anomalies are possible with the identification and processing of raw cardiac health data. In order to analyze the raw data and generate current and distinct differentiation for heart disease, this research utilized machine learning approaches. Predicting heart disease is a challenge and of great importance in the medical field. However, mortality can be significantly reduced if the condition is detected early and preventive measures are taken as soon as possible. Therefore, it would be extremely desirable to continue this study and concentrate the inquiry on bigger datasets.

In this paper, we proposed TabNet against base-models of four machine learning algorithm in which comparative analysis was done and promising results were achieved. The conclusion in which found is that, TabNet performed better in this analysis over the base-models. With an accuracy of over 94%, the Tabnet deep learning method was able to predict heart disease with some degree of precision. Many researchers have previously suggested that we should use ML where the dataset is not that large, which is proved in this paper. The methods which are used for comparison are confusion matrix, precision, recall, and F1score.

For the 14 features which were in the dataset, KNeighbors classifier performed better in the ML approach when data preprocessing is applied. The computational time was also reduced which is helpful when deploying a model. It was also found out that the dataset should be normalized; otherwise, the training model gets overfitted sometimes and the accuracy achieved is not sufficient when a model is evaluated for real-world data problems which can vary drastically to the dataset on which the model was trained.

Recommendation

The direction of this study can be changed in the future by combining machine learning methods with more accurate forecasting methods. To increase the prognostic ability of cardiac illness and broaden awareness of critical characteristics, innovative feature selection methodologies can also be created.

References

[1] World Health Organization, “Cardiovascular Diseases (CVDs),” *https://www.who.int/news-room/fact-sheets/detail/cardiovascular-diseases-(cvds)*, 2021.

[2] Center for Disease Control and Prevention, “https://www.cdc.gov/heartdisease/risk_factors.htm#:~:text=Genetic%20factors%20likely%20play%20some,that%20may%20increase%20their%20risk.”

[3] P. Mathur, S. Srivastava, X. Xu, and J. L. Mehta, “Artificial Intelligence, Machine Learning, and Cardiovascular Disease,” *Clinical Medicine Insights: Cardiology*, vol. 14. SAGE Publications Ltd, 2020. doi: 10.1177/1179546820927404.

[4] S. Das, R. Dey, and A. K. Nayak, “Artificial intelligence in pharmacy,” *Indian Journal of Pharmaceutical Education and Research*, vol. 55, no. 2. Association of Pharmaceutical Teachers of India, pp. 304–318, 2021. doi: 10.5530/ijper.55.2.68.

[5] J. Jiménez-Luna, F. Grisoni, N. Weskamp, and G. Schneider, “Artificial intelligence in drug discovery: recent advances and future perspectives,” *Expert Opin Drug Discov*, vol. 16, no. 9, pp. 949–959, 2021, doi: 10.1080/17460441.2021.1909567.

[6] E. Ebrahimzadeh and F. Sadoughi, “Machine learning-based models for prediction of heart failure mortality: A systematic review.,” *J Biomed Inform*, 2020.

[7] D. Cheng, D. Zou, Y. Jin, and Y. Sun, “Deep Learning with Tabular Data: A Survey.,” 2020.

[8] S. Huang, H. Wu, and Y. Li, “ TabNet: Attentive Interpretable Tabular Learning,” 2020.

[9] S. O. Arik and T. Pfister, “TabNet: Attentive Interpretable Tabular Learning,” Aug. 2019.

[10] M. Nielsen, “Neural Networks and Deep Learning,” *Determination Press.*, 2015.

[11] T. M. Mitchell, “Machine Learning.,” *McGraw-Hill.*, 1997.

[12] J. & P. V. M. Zhang, “Deep Learning in Computer Vision: Principles and Applications.,” *Springer*, 2018.

[13] A. Géron, “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. O’Reilly Media.,” 2019.

[14] S. Raschka and V. Mirjalili, “Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing.,” 2019.

[15] A. Rajkomar and J. Dean, “Machine Learning in Medicine,” *N Engl J Med*, vol. 380(14), pp. 1347–1358, 2019.

[16] M. Chen, S. Mao, and Y. Liu, “Big Data: A Survey,” *Mobile Networks and Applications*, vol. 19, no. 2, pp. 171–209, 2014.

[17] I. Goodfellow, Y. Bengio, and A. Courville, “Deep Learning. ,” in *MIT Press.*, 2016.

[18] T. Hastie, R. Tibshirani, and J. Friedman, “The Elements of Statistical Learning: Data Mining, Inference, and Prediction.,” *Springer Science & Business Media.*, 2009.

[19] C. M. Bishop, “Pattern Recognition and Machine Learning,” *Springer*, 2006.

[20] A. Gelman and J. Hill, “Data Analysis Using Regression and Multilevel/Hierarchical Models.,” *Cambridge University Press.*, 2007.

[21] M. H. Kutner, C. J. Nachtsheim, J. Neter, and W. Li, “Applied Linear Statistical Models.,” *McGraw-Hill.*, 2004.

[22] A. Géron, “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. ,” *O’Reilly Media.*, 2019.

[23] Hosmer Jr, D. W., S. Lemeshow, and R. X. Sturdivant, “Applied Logistic Regression.,” *John Wiley & Sons.*, 2013.

[24] D. C. Montgomery, E. A. Peck, and G. G. Vining, “ Introduction to Linear Regression Analysis. ,” *John Wiley & Sons.*, 2012.

[25] A. Agresti, “Categorical Data Analysis,” *John Wiley & Sons.*, 2015.

[26] G. James, D. Witten, T. Hastie, and R. Tibshirani, “An Introduction to Statistical Learning.,” *Springer*, 2013.

[27] N. R. Draper and H. Smith, “Applied Regression Analysis,” *John Wiley & Sons.*, 2014.

[28] Breiman. L, J. Friedman, R. Olshen, and C. Stone, “Classification and Regression Trees,” *Chapman & Hall/CRC.*, 1984.

[29] A. Géron, “Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems. ,” *O’Reilly Media.*, 2019.

[30] A. Gelman and J. Hill, “Data Analysis Using Regression and Multilevel/Hierarchical Models.,” *Cambridge University Press.*, 2007.

[31] J. R. Quinlan, “nduction of Decision Trees. Machine Learning,” vol. 1, no. 1, pp. 81–106, 1986.

[32] C. Cortes and V. Vapnik, “Support-vector networks. Machine Learning,” pp. 273–297, 1995.

[33] C. J. Burges, “A Tutorial on Support Vector Machines for Pattern Recognition,” *Data Min Knowl Discov*, vol. 2, no. 2, pp. 121–167, 1998.

[34] “The functions of Human Brain ,” *https://nordvpn.com/cybersecurity/glossary/input-layer/*.

[35] Y. LeCun, Y. Bengio, and G. Hinton, “Deep Learning. Nature,” vol. 521, no. 7553, pp. 436–444, 2015.

[36] G. James, D. Witten, T. Hastie, and R. Tibshirani, “An Introduction to Statistical Learning,” *Springer*, 2013.

[37] S. Shalev-Shwartz and S. Ben-David, “Understanding Machine Learning: From Theory to Algorithms.,” *Cambridge University Press*, 2014.

[38] F. Chollet, “Deep Learning with Python,” *Manning Publications.*, 2018.

[39] I. H. Witten, E. Frank, and M. A. Hall, “Data Mining: Practical Machine Learning Tools and Techniques,” *Morgan Kaufmann*, 2016.

[40] P. N. Tan, M. Steinbach, and V. Kumar, “Introduction to Data Mining,” *Pearson*, 2019.

[41] V. Mnih *et al.*, “Human-level control through deep reinforcement learning,” *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[42] D. Silver *et al.*, “Mastering the game of Go with deep neural networks and tree search.,” *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[43] R. S. Sutton and A. G. Barto, “Reinforcement Learning: An Introduction. ,” *MIT Press.*, 2018.

[44] L. P. Kaelbling, M. L. Littman, and A. W. Moore, “Reinforcement learning: A survey,” *Journal of Artificial Intelligence Research*, vol. 4, pp. 237–285, 1996.

[45] Golande, Avinash, and T. Pavan Kumar, “Heart disease prediction using effective machine learning techniques,” *International Journal of Recent Technology and Engineering*, pp. 944–950, 2019.

[46] V. L. Roger, “Epidemiology of Heart Failure: A Contemporary Perspective,” *Circ Res*, vol. 128, no. 10, pp. 1421–1434, May 2021, doi: 10.1161/CIRCRESAHA.121.318172.

[47] S. K. J and G. S, “Prediction of Heart Disease Using Machine Learning Algorithms,” in *2019 1st International Conference on Innovations in Information and Communication Technology (ICIICT)*, 2019, pp. 1–5. doi: 10.1109/ICIICT1.2019.8741465.

[48] G. D. Lopaschuk, Q. G. Karwi, R. Tian, A. R. Wende, and E. D. Abel, “Cardiac Energy Metabolism in Heart Failure,” *Circ Res*, vol. 128, no. 10, pp. 1487–1513, May 2021, doi: 10.1161/CIRCRESAHA.121.318241.

[49] D. Tomasoni, M. Adamo, C. M. Lombardi, and M. Metra, “Highlights in heart failure,” *ESC Heart Failure*, vol. 6, no. 6. Wiley-Blackwell, pp. 1105–1127, Dec. 01, 2019. doi: 10.1002/ehf2.12555.

[50] M. Kavitha, G. Gnaneswar, R. Dinesh, Y. R. Sai, and R. S. Suraj, “Heart Disease Prediction using Hybrid machine Learning Model,” in *2021 6th International Conference on Inventive Computation Technologies (ICICT)*, 2021, pp. 1329–1333. doi: 10.1109/ICICT50816.2021.9358597.

[51] G. Dikshit and R. Tiwari, “Effective Prediction of Heart Disease Through Machine Learning,” *ECS Trans*, vol. 107, no. 1, pp. 17501–17516, Apr. 2022, doi: 10.1149/10701.17501ecst.

[52] G. Choudhary and S. N. Singh, “Prediction of Heart Disease using Machine Learning Algorithms,” in *2020 International Conference on Smart Technologies in Computing, Electrical and Electronics (ICSTCEE)*, 2020, pp. 197–202. doi: 10.1109/ICSTCEE49637.2020.9276802.

[53] P. O. Gyamfi, C. Agyemang, F. O. Mensah, and R. Ofei, “Predicting the Risk of Cardiovascular Disease Using Tabular Neural Networks,” *J Healthc Eng*, pp. 1–9, 2021.

[54] X. Li, Y. GU, Y. Zong, Y. Chen, and Y. Zhang, “A machine learning approach to predict heart failure mortality,” *J Biomed Inform*, 2020.

[55] A. J. Russak *et al.*, “Machine Learning in Cardiology—Ensuring Clinical Impact Lives Up to the Hype,” *Journal of Cardiovascular Pharmacology and Therapeutics*, vol. 25, no. 5. SAGE Publications Ltd, pp. 379–390, Sep. 01, 2020. doi: 10.1177/1074248420928651.

[56] H. Jindal, S. Agrawal, R. Khera, R. Jain, and P. Nagrath, “Heart disease prediction using machine learning algorithms,” in *IOP Conference Series: Materials Science and Engineering*, IOP Publishing Ltd, Jan. 2021. doi: 10.1088/1757-899X/1022/1/012072.

[57] S. Mohan, C. Thirumalai, and G. Srivastava, “Effective heart disease prediction using hybrid machine learning techniques,” *IEEE Access*, vol. 7, pp. 81542–81554, 2019, doi: 10.1109/ACCESS.2019.2923707.

[58] M. S. Amin, Y. K. Chiam, and K. D. Varathan, “Identification of significant features and data mining techniques in predicting heart disease,” *Telematics and Informatics*, vol. 36, pp. 82–93, Mar. 2019, doi: 10.1016/j.tele.2018.11.007.

[59] M. Zafar-uz-Zaman, National Centre for Physics, Centres of Excellence in Science & Applied Technologies, Institute of Electrical and Electronics Engineers. Islamabad Section, and Institute of Electrical and Electronics Engineers, *Proceedings of 2019 16th International Bhurban Conference on Applied Sciences & Technology (IBCAST) : 8th-12th January, 2019*.

[60] S. V. Seyedamin, Giovanna Sannino, Giuseppe De Pietro, Pouriyeh, Hamid Arabania, and Juan Gutierrez, “A comprehensive investigation and comparison of Machine Learning Techniques in the domain of heart disease,” *In Computers and Communications (ISCC)*, pp. 204–207, 2017.

[61] BMC, “https://www.bmc.com/blogs/installing-jupyter-for-big-data-and-analytics/.”

[62] Jupiter Notebook 7.0.2, “https://jupyter-notebook.readthedocs.io/en/stable/notebook.html#:~:text=Introduction,share%20your%20ideas%20with%20others.”

[63] Data Science Dojo, “https://datasciencedojo-com.webpkgcache.com/doc/-/s/datasciencedojo.com/blog/logistic-regression-in-r-tutorial/.”

[64] M. Jordan, J. Kleinberg, and B. Schölkopf, “Pattern Recognition and Machine Learning.”

[65] University of Illinois at Chicago, “https://ademos.people.uic.edu/Chapter24.html#:~:text=A%20decision%20tree%20is%20a,on%20a%20variety%20of%20parameters.”

[66] PubMed, “https://pubmed.ncbi.nlm.nih.gov/14632445/#:~:text=Random%20Forest%20is%20an%20ensemble,feature%20selection%20in%20tree%20induction.”

[67] Random forests — An ensemble of decision trees, “https://towardsdatascience.com/random-forests-an-ensemble-of-decision-trees-37a003084c6c.”

[68] G. (Gareth M. James, D. Witten, T. Hastie, and R. Tibshirani, *An introduction to statistical learning : with applications in R*.

[69] IBM, “https://www.ibm.com/topics/boosting#:~:text=Boosting%20is%20an%20ensemble%20learning,the%20weaknesses%20of%20its%20predecessor.”

[70] T. Hastie, R. Tibshirani, and J. Friedman, “Springer Series in Statistics The Elements of Statistical Learning Data Mining, Inference, and Prediction.”

[71] C. Li, “A Gentle Introduction to Gradient Boosting.” [Online]. Available: https://github.com/cheng-li/pyramid

[72] S. M. Piryonesi and T. E. El-Diraby, “Data Analytics in Asset Management: Cost-Effective Prediction of the Pavement Condition Index,” 2019, doi: 10.1061/(ASCE).

[73] UCI Machine Learning Repository, “http://archive.ics.uci.edu/dataset/45/heart+disease.”

[74] Great Learning, “https://www.mygreatlearning.com/blog/label-encoding-in-python/#:~:text=Label%20encoding%20is%20a%20technique,only%20operate%20on%20numerical%20data.”.

[75] Level Up Coding, “https://levelup.gitconnected.com/importance-of-data-preprocessing-in-machine-learning-17045bc18d01#:~:text=Data%20preprocessing%20is%20the%20process,reliability%20of%20machine%20learning%20models.”

[76] GeeksForGeeks, “https://www.geeksforgeeks.org/data-preprocessing-in-data-mining/.”

[77] Real Python, “https://realpython.com/python-data-cleaning-numpy-pandas/.”

[78] “Feature Selection Techniques in Machine Learning.”

[79] H2O.ai, “https://h2o.ai/wiki/feature-selection/#:~:text=In%20the%20machine%20learning%20process,why%20feature%20selection%20is%20important.”

[80] JavaPoint, “https://www.javatpoint.com/feature-selection-techniques-in-machine-learning#:~:text=In%20Filter%20Method%2C%20features%20are,using%20different%20metrics%20through%20ranking.”

[81] “Scikit-learn(sklearn) in Python-the most important Machine Learning tool I learnt last year!”

[82] S. O. Arik and T. Pfister, “TabNet: Attentive Interpretable Tabular Learning,” Aug. 2019, [Online]. Available: http://arxiv.org/abs/1908.07442

[83] A. F. T. Martins and R. F. Astudillo, “From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification,” Feb. 2016, [Online]. Available: http://arxiv.org/abs/1602.02068

[84] N. Mehrabi, F. Morstatter, N. Saxena, K. Lerman, and A. Galstyan, “A Survey on Bias and Fairness in Machine Learning,” Aug. 2019, [Online]. Available: http://arxiv.org/abs/1908.09635

[85] S. Boughorbel, F. Jarray, and A. Kadri, “Fairness in TabNet Model by Disentangled Representation for the Prediction of Hospital No-Show,” Mar. 2021, [Online]. Available: http://arxiv.org/abs/2103.04048

[86] Towards Ai, “https://pub.towardsai.net/confusion-matrix-179b9c758b55.”

[87] Google for Developers, “https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc.”

[88] dasha.ai, “https://dasha.ai/en-us/blog/auc-roc.”

[89] arize.com, “https://arize.com/blog-course/f1-score/.”

[90] Seaborn, “https://seaborn.pydata.org/tutorial/introduction.html.”

[91] NumPy, “https://numpy.org/doc/stable/user/whatisnumpy.html#:~:text=It%20is%20a%20Python%20library,Fourier%20transforms%2C%20basic%20linear%20algebra.”

[92] University of Washington, “https://faculty.washington.edu/otoomet/machinelearning-py/numpy-and-pandas.html.”

[93] Matplotlib, “https://matplotlib.org/#:~:text=Matplotlib%20is%20a%20comprehensive%20library,can%20zoom%2C%20pan%2C%20update.”

[94] Pythonprogramming.net, “https://pythonprogramming.net/python-3-os-module/#:~:text=The%20main%20purpose%20of%20the,path%20by%20doing%20listdir().”

[95] GeeksforGeeks, “https://www.geeksforgeeks.org/data-preprocessing-in-data-mining/#:~:text=Data%20preprocessing%20is%20an%20important,the%20specific%20data%20mining%20task.”

[96] Wikipedia, “https://en.wikipedia.org/wiki/Skewness#:~:text=In%20probability%20theory%20and%20statistics,Example%20distribution%20with%20positive%20skewness.”

[97] Towards Data Science, “https://towardsdatascience.com/understanding-feature-importance-and-how-to-implement-it-in-python-ff0287b20285#:~:text=What%20is%20Feature%20Importance%3F,%E2%80%9Cimportance%E2%80%9D%20of%20each%20feature.”

List of Figures

List of Tables

Appendices

Import Libraries

```
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
import scikitplot as skplt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from xgboost import XGBClassifier
from sklearn.ensemble import RandomForestClassifier,StackingClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from scipy import stats
from numpy import isnan
from sklearn.impute import KNNImputer


from sklearn.model_selection import GridSearchCV, cross_val_score, StratifiedKFold, learning_curve
import pytorch_tabnet
from pytorch_tabnet.tab_model import TabNetClassifier
```

Import Main Libraries

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import os
import scipy
import warnings
warnings.filterwarnings("ignore")
from sklearn.preprocessing import MinMaxScaler
from sklearn import metrics
import random
```

Install Required Libraries

```
%%capture
!pip install -q hvplot
!pip install pytorch-tabnet
```

Insert Cleveland Dataset from local PC

```
data=pd.read_csv("C:/Users/Kwame Steve/Downloads/heartdisease.data",header=None)
data = data.replace("?",np.nan)

data = data.dropna().reset_index(drop=True)
data.columns = ['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'oldpeak', 'ST slope','ca', 'thal', 'target']

k=['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'ST slope','ca', 'thal', 'target']

for j in k:
    data[j] = data[j].astype('float').astype('int')
```

shape of dataset before Pre-processing

```
data=pd.read_csv("C:/Users/Kwame Steve/Downloads/heartdisease.data",header=None)
data = data.replace("?",np.nan)
data.shape
```

shape of dataset after Pre-processing

```
data=pd.read_csv("C:/Users/Kwame Steve/Downloads/heartdisease.data",header=None)
data = data.replace("?",np.nan)

data = data.dropna().reset_index(drop=True)
data.columns = ['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'oldpeak', 'ST slope','ca', 'thal', 'target']

k=['age', 'sex', 'chest pain type', 'resting bp s', 'cholesterol',
       'fasting blood sugar', 'resting ecg', 'max heart rate',
       'exercise angina', 'ST slope','ca', 'thal', 'target']

for j in k:
    data[j] = data[j].astype('float').astype('int')

data['oldpeak'] = data['oldpeak'].astype('float')
data['target'] = np.where(data.target>0,1,0)
dataTab = data.copy()
data.shape
```

Convert Nominal Variables - chest pain type, resting ecg and thal

```
CP_Dict = {1:'typical angina',2:'atypical angina',3:'non-anginal',4:'asymptomatic'}
ECG_Dict = {0:'normal',1:'ST-T wave abnormality',2:'left ventricular hypertrophy'}
thal_Dict = {3:'normal',6:'fixed defect',7:'reversable defect'}

data.replace({"chest pain type": CP_Dict},inplace=True)
data.replace({"resting ecg": ECG_Dict},inplace=True)
data.replace({"thal": thal_Dict},inplace=True)

data.head()
```

Categorical Binaries - Sex, FBS and Exang

```
f, axes = plt.subplots(1, 3, figsize=(15, 5))

sns.countplot(ax=axes[0],x='sex', data=data, palette=['pink','orange'],hue="target")
axes[0].set_title("sex", fontsize=20)

sns.countplot(ax=axes[1],x='fasting blood sugar', data=data, palette=['pink','orange'],hue="target")
axes[1].set_title("fasting blood sugar", fontsize=20)

sns.countplot(ax=axes[2],x='exercise angina', data=data, palette=['pink','orange'],hue="target")
plt.title("exercise angina", fontsize=20)
```

Categorical Variables - CP, ECG and thal

```
plt.figure(figsize=(12,5))
sns.countplot(x='chest pain type', data=data, palette=['pink','orange'],hue="target")
plt.title("chest pain type", fontsize=20)
```

Correlations

```
import hvplot.pandas
data.drop('target', axis=1).corrwith(data.target).hvplot.barh(
    width=600, height=400,
    title="Correlation between Heart Disease and Numeric Features",
    ylabel='Correlation', xlabel='Numerical Features'
)
```

Age vs Max HR

```
plt.figure(figsize=(9, 7))
plt.scatter(data_disease["age"],
            data_disease["max heart rate"],
            c="salmon")
plt.scatter(data_normal["age"],
            data_normal["max heart rate"],
            c="lightblue")
plt.title("Heart Disease in function of Age and Max Heart Rate")
plt.xlabel("Age")
plt.ylabel("Max Heart Rate")
plt.legend(["Disease", "No Disease"]);
```