University of Piraeus, School of Information and Communication Technologies,
Department of Digital Systems

M.Sc. Thesis Title:

"In the Heart of Data: Machine Learning Applications for Improved Heart Failure Outcome Prediction"

by

Chelis Apostolos

Dissertation

*submitted in partial fulfilment of the requirements for the degree of
Master of Science in Information Systems & Services
in the specialization of Big Data and Analytics*

Piraeus, February 2024

ACKNOWLEDGMENTS

First and foremost, I would like to thank my supervisor, Professor Filippakis Michael, for his willingness to provide me with his academic support and knowledge.

I would also like to thank the University of Piraeus and the Department of Digital Systems, which hosted this research endeavor, as well as Prof. Kiriazis Dimosthenis and Assoc. Prof. Halkidi Maria as members of my thesis committee.

A debt of gratitude is also owed to both my caring parents and my loving life partner for their moral and physical support during my thesis.

TABLE OF CONTENTS

SUMMARY

The aim of this dissertation is to investigate and evaluate the effectiveness of various machine learning models (Random Forest, Decision Tree, Support Vector Machine, Logistic Regression, k-Nearest Neighbors and Gradient Boosting) and methods (Feature selection, Undersamlig, Oversampling and Principal component analysis), in predicting the occurrence of death events in patients with heart failure. The study employs a diverse set of models on a dataset comprising medical records of heart failure patients, released by Ahmad et al. (2017). Through rigorous analysis, hyperparameter optimization, and exploration of data preprocessing techniques, the research seeks to develop a robust framework capable of accurately classifying heart failure patients based on their risk of experiencing a death event. The findings aim to contribute valuable insights to the evolving landscape of precision healthcare. Specifically, by testing and subsequently selecting the most effective methods/models of machine learning and addressing critical factors/errors present in the datasets, a more efficient determination of results is achieved.

Keywords – Machine learning models, Data mining, Performance Metrics, Hyperparameter tunning, Biomedical informatics, Oversampling, Undersampling, Feature Selection, Principal Component Analysis, Binary Classification.

ΠΕΡΙΛΗΨΗ

Ο στόχος αυτής της διατριβής είναι να εξετάσει και να αξιολογήσει την αποτελεσματικότητα διάφορων μοντέλων μηχανικής μάθηση (Random Forest, Decision Tree, Support Vector Machine, Logistic Regression, k-Nearest Neighbors και Gradient Boosting) και μεθόδων (Feature selection, Undersamplig, Oversampling και Principal component analysis), στον προσδιορισμό της πιθανότητας εμφάνισης συμβάντων θανάτου σε ασθενείς με καρδιακή ανεπάρκεια. Η μελέτη χρησιμοποιεί ένα ποικίλο σύνολο μοντέλων σε ένα σύνολο δεδομένων (Ahmad et al., 2017) που περιλαμβάνει ιατρικές εγγραφές ασθενών με καρδιακή ανεπάρκεια. Μέσω αυστηρής ανάλυσης, βελτιστοποίησης υπερπαραμέτρων και εξερεύνησης τεχνικών προεπεξεργασίας δεδομένων, η έρευνα έχει ως στόχο την ανάπτυξη ενός αξιόπιστου πλαισίου, ικανού να κατηγοριοποιεί με ακρίβεια ασθενείς με καρδιακή ανεπάρκεια, βάσει του κινδύνου εμφάνισης συμβάντος θανάτου. Τα ευρήματα της παρούσας εργασίας συμβάλλουν στον τομέα της υγειονομικής περίθαλψης. Πιο συγκεκριμένα, μέσω της δοκιμής και εν συνεχεία επιλογής των πιο αποτελεσματικών μεθόδων – μοντέλων μηχανικής μάθησης και την αντιμετώπιση των κρίσιμων παράγοντων – σφαλμάτων που παρουσιάζουν τα σύνολα δεδομένων, επιτυγάνεται ο αποτελεσματικότερος προσδιορισμός των αποτελεσμάτων.

Λεξεις Κλειδία – Μοντέλα μηχανικής μάθησης, Εξόρυξη δεδομένων, Μετρικές απόδοσης, Βελτιστοποίηση υπερπαραμέτρων, Βιοϊατρική πληροφορική, Υπερδειγματοληψία, Υποδειγματοληψία, Επιλογή χαρακτηριστικών, Ανάλυση κύριων συνιστωσών, Δυαδική κατηγοριοποίηση.

Chapter 1. INTRODUCTION

Chapter 1 provides an overview of the major issues covered in the current thesis.

1.1 Generally

Cardiovascular diseases refer to conditions affecting the heart and blood vessels, causing approximately seventeen million global fatalities each year (Wu et al., 2023). Specifically, heart failure arises when the heart cannot adequately pump blood due to factors like diabetes, high blood pressure, or other cardiac issues (Heiney et al., 2020). Recognizing the critical role of the heart, medical professionals prioritize predicting heart failure, yet current clinical practices often fall short in achieving high accuracy (Buchan et al., 2019).

Machine learning, when applied to medical records, emerges as a promising tool for predicting survival in patients exhibiting heart failure symptoms (Al'Aref et al., 2019; Al'Aref et al., 2018) and identifying key clinical features or risk factors associated with heart failure (Gallagher et al., 2019; Dunn et al., 2007). Scientists can harness machine learning not only for clinical predictions (Ambale-Venkatesh et al., 2017; Weng et al., 2017) but also for prioritizing features (Shilaskar & Ghatol, 2013). This dissertation delves into the analysis of a dataset comprising medical records of heart failure patients released by Ahmad et al. in July 2017.

This thesis delves into the realm of machine learning applications in the context of heart failure outcome prediction. Employing a diverse set of machine learning models, including Random Forest, Decision Tree, Support Vector Machine (SVM), Logistic Regression, k-Nearest Neighbors (KNN) and Gradient Boosting, the study explores the efficacy of each model in classifying and detecting the critical target: the occurrence of a death event in patients with heart failure.

To optimize the performance of these models, a Grid Search methodology is employed, systematically fine-tuning hyperparameters for enhanced predictive accuracy. Furthermore, the investigation extends to various data preprocessing techniques, such as Oversampling, Undersampling, Feature Selection, Principal Component Analysis (PCA) (and combinations of those techniques) aiming to unravel latent patterns within clinical data.

The core objective is to develop a robust framework capable of accurately classifying heart failure patients based on their risk of experiencing a death event. Through a meticulous analysis of model performances, hyperparameter optimizations, and feature engineering strategies, this research contributes to the evolving landscape of precision healthcare. The findings not only shed light on the most effective machine learning models for heart failure outcome prediction but also provide valuable insights into the critical factors influencing model performance in this domain. This master thesis paves the way for future advancements in the development of tailored predictive models for cardiovascular risk assessment.

1.2 Thesis stimuli and structure

The explanation of the stimuli of this thesis can be found below. The work presented in this thesis was motivated by an increased recognition among academics that:

- The increasing availability of electronic health data presents a major opportunity in healthcare for both discovery and practical applications to improve healthcare (Wiens and Shenoy, 2023).
- Modeling survival for heart failure is still a problem nowadays, both in terms of achieving high prediction accuracy and identifying the driving factors (Chicco & Jurman, 2020).
- Machine learning applied to medical records, can be an effective tool to predict the survival of each patient having heart failure symptoms (Al'Aref et al., 2019).
- Machine learning applied to medical records helps to detect the most important clinical features (or risk factors) that may lead to heart failure (Gallagher et al., 2019).
- Machine learning is an old concept that has recently gained a lot of attention due to the explosion of data generation processes in healthcare (Alanazi, 2022).

The following is a summary of the remaining sections of the current thesis. Chapter 2 presents a detailed literature review of the topics under examination. In Chapter 3, the data and methods utilized in the present thesis are described. Chapter 4 is dedicated to presenting the results, and this is where the discussion of the findings also takes place. Finally, Chapter 5 focuses on the key findings drawn from the study.

Chapter 2. LITERATURE REVIEW

Chapter 2 presents a detailed literature review of the topics and methods under examination.

2.1 Machine Learning and Data Mining

2.1.1 Generally

In the contemporary era, voluminous datasets have become ubiquitous, and their effective utilization through diverse algorithms within the realm of machine learning holds the potential to transmute these datasets into consequential knowledge. Machine learning represents a domain of scholarly investigation that emerges at the confluence of three fundamental domains: statistics, artificial intelligence, and computer science (Muller et al., 2016). The foundational definition of machine learning encompasses a sphere of scholarly inquiry that confers upon computers the capability to learn autonomously, devoid of explicit programming (Awad, 2015).

Historically, software engineering amalgamated rules crafted by humans with data to generate solutions for problems. Conversely, machine learning operates by utilizing data and solutions to unveil the underlying rules governing a problem (Chollet, 2021). In other words, machine learning serves as a mechanism to transform data into comprehensible insights. Over the last five decades, a substantial surge in data generation has taken place. However, this amassed data remains inconsequential unless subjected to thorough analysis to unveil concealed patterns. Machine learning methodologies are employed to autonomously identify meaningful inherent patterns within intricate datasets that would otherwise pose challenges to uncover manually. The latent patterns and discernments pertaining to a given issue hold the potential to prognosticate future occurrences and facilitate multifaceted decision-making processes.

Within this context, three distinct categories of machine learning manifest: Supervised Learning, Unsupervised Learning, and Reinforcement Learning. Subsequent to this delineation, the forthcoming subsection will undertake an exposition of these aforementioned classifications of machine learning.

2.1.2 Contexts and Processes of Machine Learning

There are three major contexts regarding machine learning, the dataset, the features and the model (Amer et al., 2022).

- Dataset: A compilation of data instances, encompassing significant attributes pivotal to addressing the given issue.
- Features: Essential components of information that contribute to the comprehension of a problem. These are inputted into a machine learning algorithm to facilitate its learning process.
- Model: The portrayal (internal framework) of a phenomenon acquired by a machine learning algorithm. This understanding is derived from the data presented during the training phase. The model constitutes the outcome achieved subsequent to algorithmic training. To illustrate, a decision tree algorithm would undergo training and yield a decision tree model as its result.

Additionally, machine learning follows some constituent steps in order to analyze and retrieve information from the data (Naqa & Murphy, 2015). Those steps-processes are analyzed below:

- Data Collection: Acquire the data essential for the algorithm's learning process.
- Data Preparation: Organize and transform the data into an ideal structure, extracting crucial attributes and conducting dimensionality reduction.
- Training: Referred to as the fitting phase, this is when the machine learning algorithm actively learns through exposure to the collected and prepared data.
- Evaluation: Assess the model's performance to determine its efficacy.
- Tuning: Refine the model to optimize its performance.

2.1.3 Machine Learning Approaches

As mentioned earlier, a multitude of approaches are available for conducting machine learning endeavors, often grouped into distinct categories. Among these, Supervised and Unsupervised approaches, having gained established and frequent use, hold prominence. On the other hand, Semi-supervised and Reinforcement Learning, being more contemporary and intricate, have showcased remarkable accomplishments (Muhammad & Yan, 2015).

2.1.3.1 *Supervised Learning*

Supervised learning constitutes a category of machine learning wherein machines are trained utilizing meticulously “labeled” training data. Grounded in this data, machines make “anticipations” about the output. The term "labeled data" refers to input data that is already associated with the accurate output. Within the realm of supervised learning, the data used to train machines functions analogously to a supervisor, imparting the knowledge necessary for accurate output prediction. This mechanism mirrors the educational dynamic of a student guided by a teacher. Supervised learning involves furnishing both input data and corresponding correct output data to the machine learning model. The primary objective of a supervised learning algorithm is to ascertain a mapping function that correlates the input variable (x) with the output variable (y) (Alloghani et al., 2020). In practical applications, supervised learning finds utility in tasks such as Risk Assessment, Image Classification, Fraud Detection, and Spam Filtering, among others.

The steps involved in supervised learning (Burkart & Huber, 2021) are analyzed below:

- Dataset Type Identification: Begin by discerning the nature of the training dataset.
- Data Collection: Assemble the labeled training data through collection efforts.
- Dataset Partitioning: Divide the training dataset into distinct segments: the training dataset itself, a test dataset, and a validation dataset.
- Input Feature Specification: Define the input features within the training dataset. These features should be sufficiently informative for the model to achieve accurate output predictions.
- Algorithm Selection: Choose an appropriate algorithm for the model, such as options like support vector machines or decision trees.
- Algorithm Execution: Apply the chosen algorithm to the training dataset. Occasionally, validation sets might be required to fine-tune control parameters; these sets are subsets of the training dataset.
- Model Accuracy Assessment: Evaluate the model's accuracy by employing the test set. Successful accurate output predictions indicate a reliable model.

Depending on what you want to predict, supervised learning can be used to solve two types of problems: regression or classification (Jiang et al., 2020).

- Regression Problem: When the goal is to forecast continuous values, like predicting house prices or outdoor temperatures, regression comes into play. This kind of problem lacks a rigid value boundary as predictions can encompass any numerical value without limitations.
- Classification Problem: In instances where the inquiry is akin to "Is this a cat?", the scenario constitutes a classification problem. This involves the task of categorizing responses into distinct classes, such as 'yes' or 'no'. In this particular instance, the answer falls under the 'yes' category, making it a binary classification problem.

Supervised learning has some major advantages and disadvantages. More extensively, supervised learning offers a range of advantages that contribute to its efficacy in various applications. Through the utilization of supervised learning, the model gains the ability to predict outputs by drawing upon past experiences and established patterns. This approach provides us with a precise comprehension of object classes within supervised learning, enhancing our understanding of their categorization. Moreover, the supervised learning model emerges as a potent tool for tackling an array of real-world challenges, including the intricate tasks of fraud detection and spam filtering. Its capacity to address these multifaceted problems underscores its practical relevance (Osisanwo et al., 2017). Despite its merits, supervised learning exhibits several limitations that warrant consideration. Supervised learning models demonstrate inefficiency in managing intricate tasks due to their inherent structure and reliance on labeled data. The effectiveness of supervised learning hinges on the assumption of similarity between test and training data. If test data varies significantly from the training dataset, accurate predictions may falter. Training supervised learning models demands substantial computational resources, contributing to prolonged processing times. Effective utilization of supervised learning necessitates a robust understanding of object classes. This prerequisite may hinder the model's applicability in cases with limited class insights (Schrider & Kern., 2018).

2.1.3.2 *Unsupervised Learning*

Unsupervised learning, a technique within machine learning, involves models that aren't guided by a labeled training dataset. Instead, these models autonomously uncover concealed patterns and insights from provided data. This process can be likened to the way the human brain assimilates new information. In essence, unsupervised learning stands as a form of machine learning wherein models are trained using unlabeled datasets and operate on this data without external guidance (Mahesh, 2020).

Unlike supervised learning, which entails possessing input and corresponding output data, unsupervised learning cannot be directly applied to regression or classification problems. This is due to the absence of labeled data. The primary objective of unsupervised learning lies in unearthing the inherent structure of a dataset, clustering data based on similarities, and presenting the dataset in a more condensed representation (Alloghani et al., 2020).

The significance of unsupervised learning is underscored by several key factors (Wang & Biljecki, 2022). Primarily, unsupervised learning proves invaluable in extracting valuable insights from data. This capability to uncover meaningful patterns contributes to its vital role in data analysis. Furthermore, the parallel between unsupervised learning and human cognitive processes enhances its relevance in the realm of genuine artificial intelligence. Similar to how humans learn and think through personal experiences, unsupervised learning autonomously identifies intricate patterns within data, lending it a certain authenticity. The essence of unsupervised learning is magnified by its adeptness at handling unlabeled and unclassified datasets. This unique capability of processing raw, unstructured data heightens the significance of unsupervised learning in data exploration and understanding. In the practical domain, scenarios often arise where input data lacks corresponding output, necessitating the application of unsupervised learning techniques. This aspect of addressing real-world complexities further underscores the indispensability of unsupervised learning in modern machine learning paradigms.

Depending on what you want to group together, unsupervised learning can group data together by clustering or association (Li et al., 2020).

- Clustering: Clustering, involves grouping objects into clusters in a manner that objects sharing the greatest similarities are consolidated within a cluster, while maintaining minimal or no similarities with objects from other clusters. This technique entails identifying shared characteristics among data objects and classifying them based on the presence or absence of these shared traits.

- Association: Association, pertains to an unsupervised learning approach aimed at revealing connections between variables within extensive databases. This method identifies sets of items that frequently co-occur in the dataset. By establishing such associations, the effectiveness of marketing strategies is amplified. For instance, it identifies trends like individuals purchasing item X (e.g., Coffee) often also acquire item Y (e.g., Sugar). A classic illustration of Association is exemplified in Market Basket Analysis.

Unsupervised learning has both advantages and disadvantages. More extensively, unsupervised learning proves advantageous in tackling intricate tasks that surpass the complexity of supervised learning. This distinction arises from the absence of labeled input data in unsupervised learning scenarios. Furthermore, the preference for unsupervised learning stems from the ease of acquiring unlabeled data in contrast to the often laborious task of obtaining labeled data. Conversely, the intrinsic complexity of unsupervised learning presents notable drawbacks. The absence of corresponding output data poses a challenge that differentiates it from supervised learning. As a result, the outcomes yielded by unsupervised learning algorithms might exhibit lower accuracy levels. The absence of labeled input data renders algorithms incapable of anticipating precise output, contributing to potential inaccuracies in results (Fahle et al., 2020).

2.1.3.3 *Semi-supervised Learning*

Semi-supervised learning resides in the intermediary realm between supervised and unsupervised learning paradigms, capitalizing on both labeled and unlabeled data during the training process. This approach typically involves utilizing a modest quantity of labeled data alongside a substantial volume of unlabeled data. Systems adopting this technique can notably enhance their learning accuracy (Zhu, 2005). The preference for semi-supervised learning arises particularly when the available labeled data demands expert and pertinent resources for effective training or learning. In contrast, gathering unlabeled data usually entails minimal supplementary resource allocation (Hady &

Schwenker, 2013). Semi-supervised learning. Handbook on Neural Information Processing, 215-239.).

Semi-supervised learning models are experiencing a surge in popularity across various industries, driven by their effectiveness in addressing complex challenges (Zhou & Zhou, 2021). Several notable applications highlight their utility:

- Speech Analysis: This stands as a quintessential illustration of semi-supervised learning's practicality. Labeling vast quantities of audio data is an arduous task requiring substantial human resources. By employing semi-supervised learning, this hurdle can be surmounted, making the process more feasible.
- Web Content Classification: Labeling every webpage on the expansive internet is a formidable and impractical endeavor due to the extensive human intervention it necessitates. Semi-supervised learning algorithms offer a viable solution to mitigate this challenge, reducing the magnitude of manual effort required.
- Search Engine Ranking: Major players like Google harness the capabilities of semi-supervised learning algorithms to effectively rank webpages in response to user queries, thereby enhancing the accuracy and relevance of search results.
- Protein Sequence Classification: In the domain of DNA strand analysis, where sequences are vast and intricate, the intervention of human experts is indispensable. The advent of semi-supervised models has emerged as a pivotal advancement in this sphere, streamlining the classification process.
- Text Document Classification: Given the impracticality of procuring extensive amounts of labeled text data, semi-supervised learning emerges as an optimal solution. This approach circumvents the challenge by leveraging both labeled and unlabeled data, making it a valuable model for overcoming data scarcity.

2.1.3.4 *Reinforcement Learning*

Reinforcement learning resides within the domain of machine learning, focusing on making optimal decisions to maximize rewards within specific scenarios. This methodology is harnessed by diverse software and machines to ascertain the most favorable actions or paths in given situations (Wiering & Van Otterlo, 2012). Notably distinct from supervised learning, where training data carries the correct answers, reinforcement learning operates without predefined solutions. Instead, a reinforcement agent determines its actions to accomplish assigned tasks, learning from its accumulated experiences due to the absence of a conventional training dataset (Li, 2017).

Reinforcement learning encompasses the realm of decision-making science. Its core objective is to acquire optimal behavioral patterns within an environment to attain the utmost rewards. Unlike supervised or unsupervised machine learning paradigms that rely on input data, reinforcement learning gathers data through trial-and-error procedures, facilitating adaptive learning without predefined datasets (Sutton & Barto, 2018).

Reinforcement learning employs algorithms that deduce suitable actions based on outcomes. Following each action, the algorithm receives feedback to ascertain the correctness, neutrality, or incorrectness of its choice. This approach is particularly effective for automated systems necessitating numerous nuanced decisions devoid of human intervention (François-Lavet et al., 2018).

At its core, reinforcement learning operates as an autonomous, self-educating system that evolves through trial and error. It undertakes actions with the intent of optimizing rewards, effectively learning through practical application to achieve optimal results (Wang et al., 2016).

2.1.4 Knowledge Discovery in Databases

Knowledge Discovery in Databases (KDD), represents a systematic and analytical approach to model data sourced from a database, with the intent of extracting significant and valuable "knowledge" through data mining techniques. Data mining forms the cornerstone of the KDD process and is imperative for its comprehensive methodology. This methodology employs a variety of autonomous learning algorithms to derive meaningful patterns from scrutinized data. Within this cyclical and interconnected process, multiple iterations occur among distinct stages, facilitating continual feedback as mandated by algorithmic requirements and pattern elucidations. KDD process is commonly defined with the stages (Maimon & Rokach, 2005; Fayyad et al., 1996):

1. Problem identification

This marks the preliminary stage in the methodology, demanding a pre-existing grasp and proficiency in the relevant field of application. During this phase, the decision is made regarding the approach to extract insights from processed data and patterns unveiled through data mining. This underlying premise holds paramount importance and any misjudgment can lead to erroneous construals and adverse repercussions for the final user.

2. Data selection

Following the establishment of goals and objectives, a crucial undertaking involves the deliberate selection and categorization of data into meaningful subsets, guided by considerations of availability, accessibility, significance, and quality. These attributes play a foundational role in data mining, significantly influencing the composition of data models that come into fruition.

3. Data preprocessing & Transformation

Our primary objective in this phase is to enhance data efficiency through the elimination of duplicate records, erroneous or noisy data, as well as the identification and removal of outliers. Additionally, we address the strategy for managing missing data attributes and aligning data with its appropriate data types. Tailoring our approach to the specific goals and tasks at hand, we embark on the quest to uncover valuable features that aptly represent the data. This journey may involve applying diverse transformations, each yielding distinct outcomes. Furthermore, we employ mathematical techniques to reduce dimensions and eliminate redundant data. With a refined and functional dataset in place, we then transition to the mining stage, poised to extract meaningful insights.

4. Data mining

Data mining stands as the pivotal process that entails distilling valuable insights from vast datasets through a diverse array of techniques, including regression, clustering, sequence modeling, dependency analysis, and linear scrutiny. This comprehensive methodology involves the instantiation and subsequent fitting of models, leading to the recognition of discernible patterns within the data. These adapted models are analytical tools that help to trace trends from the dataset.

5. Interpretation and evaluation

Data interpretation is the process of interpreting the results that are collected from applying the data mining techniques on the models and ensuring that useful knowledge is derived from the data. In this stage, we try to evaluate and interpret the mined patterns (rules, reliability, etc.) with respect to the goals defined in the first step. This step gives a lot of focus on the usefulness and comprehensibility of the produced model. The interpretation is typically carried out by visualizing the patterns.

2.2 Machine Learning Algorithms

2.2.1 Decision tree

A decision tree is a type of supervised learning technique that's useful for both classification and regression problems, though it's mainly used for classifying things. It's like a tree-shaped guide that helps make decisions. Inside the tree, there are two important kinds of points: Decision Nodes and Leaf Nodes. Decision Nodes help with making choices and have different paths, while Leaf Nodes are the outcomes without any more paths. Figure 1, explains the general structure of a decision tree (Kingsford & Salzberg, 2008).

These choices are based on the features in the data you have. Imagine it as a map that gives you different paths to solve a problem. It's called a decision tree because, much like a tree, it starts with one main point and then branches out into smaller parts.

To create a decision tree, the CART algorithm is used, which stands for classification and regression tree algorithm. The idea is simple: the decision tree asks a question and, depending on whether the answer is Yes or No, it keeps branching out into more detailed questions or outcomes. Absolutely, a decision tree is versatile in handling different types of data. It can handle categorical data where answers might be YES or NO, as well as numeric data, making it a flexible tool for various kinds of information (Kotsiantis, 2013).

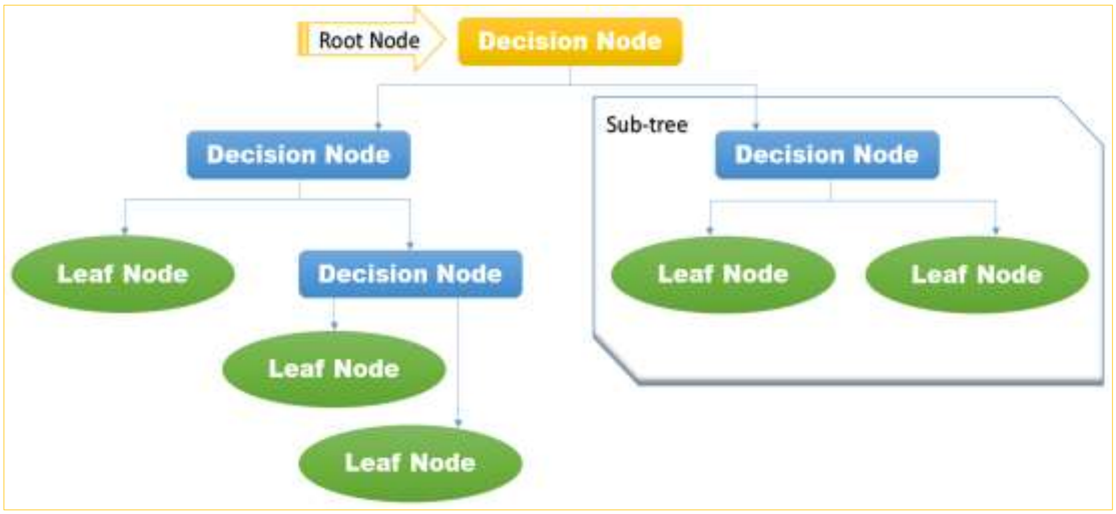


Figure 1. The general structure of a decision tree.

Indeed, decision trees often replicate human thought processes when making decisions, which makes them quite comprehensible. The rationale behind a decision tree is straightforward to grasp since it's presented in a tree-like format, resembling a flowchart that aligns with how we naturally think and make choices. Below, there are some key terms associated with decision trees (Maimon & Rokach, 2014):

- Root Node: This is where the decision tree originates. It represents the whole dataset, which is then divided into two or more similar subsets.
- Leaf Node: These are the endpoints of the tree, and no further divisions occur beyond them. They give the final output.
- Splitting: Splitting refers to dividing a decision node or the root node into smaller sub-nodes based on specific conditions.
- Branch/Sub Tree: A sub-tree forms when the tree is divided by splitting.
- Pruning: Pruning involves removing unnecessary branches from the tree, simplifying it while retaining accuracy. This process is crucial for striking the right balance between complexity and accuracy. When a tree becomes excessively large, it runs the risk of overfitting the training data. Conversely, a small tree might fail to encompass all the vital aspects of the dataset. Pruning serves as a technique to mitigate these issues by reducing the size of the tree while maintaining its accuracy. Two key methods of tree pruning are commonly utilized, Cost Complexity Pruning and Reduced Error Pruning.
- Parent/Child Node: The starting node of the tree is the parent node, and any subsequent nodes are called child nodes.

The Decision Tree algorithm operates (Charbuty & Abdulazeez, 2021) as follows:

✓ Starting Point: The process commences from the root node of the tree. The algorithm compares the root attribute's values with the corresponding attribute values in the real dataset.

✓ Traversing the Tree: Based on this comparison, the algorithm moves along the branches, progressing to the next nodes in the tree.

✓ Iterative Process: At each subsequent node, the algorithm once again compares the attribute value with the sub-nodes' attributes, advancing further as per the outcome.

✓ Reaching Leaf Nodes: This process of comparison and advancement continues until the algorithm reaches a leaf node, which signifies an endpoint and provides the final class prediction.

The algorithm's progression can be outlined through (Priyam et al., 2013) the following steps:

1. Initiation: The tree begins with a root node, labeled as S, encompassing the complete dataset.
2. Attribute Selection: The algorithm identifies the best attribute using an Attribute Selection Measure (ASM).
3. Data Subset Creation: The dataset (S) is divided into subsets, each corresponding to the possible values of the best attribute.
4. Node Generation: Decision tree nodes are generated, each representing the chosen attribute.
5. Recursion: The algorithm recursively constructs new decision trees using the subsets generated in step 3 (Data Subset Creation). This process is repeated until a point is reached where further classification is not feasible, resulting in the creation of leaf nodes. These leaf nodes are considered the final outcomes of the tree.

In this manner, the Decision Tree algorithm maps out decision pathways and class predictions based on comparisons of attribute values, leading to interpretable and actionable results.

Attribute Selection Measures (ASM) play a pivotal role in the Decision Tree algorithm by helping determine the most suitable attributes for both root and sub-nodes. They assist in making informed decisions during tree construction. Two commonly used ASM techniques are:

→ Information Gain / Entropy: Information Gain quantifies how much a particular attribute reduces the uncertainty in predicting the class. It calculates the difference between the uncertainty before and after splitting based on the attribute. Attributes with higher information gain are preferred since they lead to better classification. Information Gain is a critical concept in Decision Trees that gauges the alteration in entropy following the division of a dataset based on a particular attribute. It quantifies the information provided by a feature concerning a class. This value guides the splitting of nodes and the construction of the decision tree. The central goal of the Decision Tree algorithm is to maximize information gain, prioritizing nodes/attributes with the highest information gain for initial splitting (Forman, 2003). The formula to calculate Information Gain is as follows:

$$Information\ Gain = Entropy(S) - [(Weighted\ Avg) * Entropy\ (each\ feature)]$$

Here, Entropy represents the level of impurity in a given attribute, reflecting randomness in data. It can be computed using the formula:

$$Entropy(S) = P(yes) * log2\big(P(yes)\big) - P(no) * log2(P(no))$$

Where, S is the total number of samples, P(yes) is the probability of the positive class and P(no) is the probability of the negative class.

This concept aids in the meticulous selection of attributes for optimal tree construction, ensuring the resultant tree provides accurate classification outcomes.

→ Gini Index: The Gini Index measures the degree of impurity or disorder in a dataset. When choosing an attribute for splitting, the Gini Index evaluates the likelihood of a randomly selected item being misclassified. Lower Gini Index values indicate better attribute choices for effective classification. The Gini Index stands as a crucial metric in Decision Trees, serving as a measure of impurity or purity within the context of the CART (Classification and Regression Tree) algorithm. It assesses the quality of attribute splits during tree construction. When dealing with the Gini Index, attributes demonstrating lower values should be favored over those with higher values. This index operates solely with binary splits, aligning with the binary splitting approach adopted by the CART algorithm (Steinberg & Colla, 2009).

$$Gini\ = 1 - \sum_{i=1}^{n} (\mathrm{p}i)^2$$

Where p*i* is the probability of a particular element belonging to a specific class.

Both Information Gain and Gini Index serve as valuable tools for assessing attribute importance, enabling the Decision Tree algorithm to make optimal attribute selections for creating a more accurate and efficient tree structure. Additionally, decision trees have both advantages and disadvantages (Podgorelec et al., 2002). Table

1, presents the positive and negative aspects of this algorithm.

Table 1. Decision trees' advantages and disadvantages.

Decision tree	
Positive aspects	**Negative aspects**
Compared to other algorithms decision trees require less effort for data preparation during pre-processing.	A small change in the data can cause a large change in the structure of the decision tree causing instability.
A decision tree does not require the normalization of data.	For a Decision tree sometimes calculation can go far more complex compared to other algorithms.
A decision tree does not require scaling of data as well.	Decision tree often involves higher time to train the model.
Missing values in the data also do not affect the process of building a decision tree to any considerable extent.	Decision tree training is relatively expensive as the complexity and time has taken are more.
A Decision tree model is very intuitive and easy to explain to technical teams as well as stakeholders.	The Decision Tree algorithm is inadequate for applying regression and predicting continuous values.

2.2.2 Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both classification and regression problems in machine learning. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model (Belgiu & Drăguţ, 2016).

As the name suggests, “Random Forest” is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset (Biau & Scornet, 2016). Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output. Figure 2, explains how the Random Forest algorithm works.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output (Speiser et al., 2019).

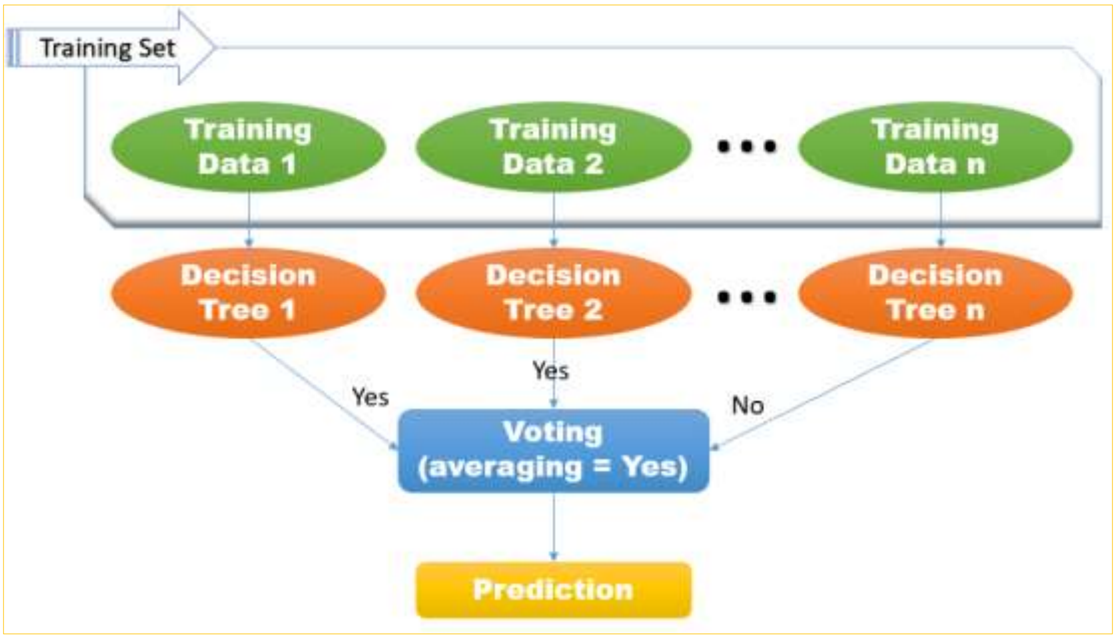


Figure 2. The operation of the Random Forest algorithm.

The procedure for implementing the Random Forest algorithm is briefly explained below:

- Initial Sampling: Begin by selecting random samples from the provided dataset or training set.
- Individual Decision Trees: The algorithm proceeds to build a distinct decision tree for each training data entry.
- Aggregation through Voting: Averaging of the decisions made by the individual decision trees is performed through a voting process.
- Determining Final Prediction: Ultimately, the prediction result with the highest number of votes is chosen as the final prediction outcome.

Table 2, presents the positive and negative aspects of the Random forest algorithm.

Table 2. Random forests' advantages and disadvantages.

Random Forest	
Positive aspects	**Negative aspects**
It reduces overfitting in decision trees and helps to improve the accuracy	It requires much computational power as well as resources as it builds numerous trees to combine their outputs.
It is flexible to both classification and regression problems	It also requires much time for training as it combines a lot of decision trees to determine the class.
It works well with both categorical and continuous values	Due to the ensemble of decision trees, it also suffers interpretability and fails to determine the significance of each variable.
It automates missing values present in the data	
Normalizing of data is not required as it uses a rule-based approach.	

2.2.3 Support Vector Machine

The Support Vector Machine (SVM) stands as one of the most widely utilized algorithms within the realm of Supervised Learning. It boasts applicability not only in Classification but also in Regression problems, though it's primary use is prominent in Classification scenarios within Machine Learning (Huang et al., 2018).

The core objective of the SVM algorithm revolves around the creation of an optimal line or decision boundary. This boundary serves to effectively partition an n-dimensional space into distinct classes, ensuring that forthcoming data points can be accurately categorized. This optimal decision boundary is formally referred to as a hyperplane (Jakkula, 2006).

Put differently, the primary aim of the SVM algorithm centers on identifying the ideal hyperplane within an N-dimensional space. This hyperplane is meticulously positioned to segregate data points belonging to different classes within the feature space. In this pursuit, the hyperplane is designed to maximize the margin between the nearest points from disparate classes. The configuration of this hyperplane aligns with the dimensionality of the feature space – a two-feature input yields a linear hyperplane, while a three-feature input results in a two-dimensional plane. Visualization becomes more intricate as the number of features exceeds three (Meyer & Wien, 2001).

According to Ghosh et al. 2019, Support Vector Machines (SVM) have two distinct forms:

→ Linear SVM: The Linear SVM is adept at handling datasets that are linearly separable. In essence, when a dataset's two classes can be accurately distinguished using a solitary straight line, this classification scenario is labeled as linearly separable data. For this purpose, the classifier employed is known as the Linear SVM classifier.

→ Non-linear SVM: Non-Linear SVM, on the other hand, is tailored for datasets that do not exhibit linear separability. In practical terms, when a dataset defies classification via a single straight line, it is categorized as non-linear data. In such cases, the Non-linear SVM classifier comes into play to effectively handle this type of data and classification complexity.

To gain a better understanding of the aforementioned, in the next paragraph a brief presentation and elaboration regarding the terminology of the Support Vector Machine algorithm is made. Support Vector Machine Terminology (Scholkopf & Smola, 2018, Gu & Han, 2013, Brereton & Lloyd, 2010):

- Hyperplane: The hyperplane serves as the decisive boundary employed to segregate data points belonging to different classes within the feature space. In the context of linear classifications, it takes the form of a linear equation, often represented as wx + b = 0.
- Support Vectors: Support vectors are data points located closest to the hyperplane, playing a pivotal role in determining both the hyperplane's placement and the margin's definition.
- Margin: The margin signifies the spatial gap between the hyperplane and the support vectors. The central objective of the support vector machine algorithm is to

maximize this margin. A broader margin is indicative of superior classification performance.

- Kernel: Kernels are mathematical functions integral to SVM, facilitating the transformation of original input data points into higher-dimensional feature spaces. This maneuver enables the identification of hyperplanes even when data points are not linearly separable in the initial input space. Common kernel functions include linear, polynomial, radial basis function (RBF), and sigmoid.
- Hard Margin: The hard margin hyperplane, also referred to as the maximum-margin hyperplane, effectively distinguishes data points of diverse categories without any classification errors.
- Soft Margin: The soft margin technique becomes relevant when data is not entirely separable or when outliers are present. In such cases, SVM introduces slack variables for each data point, relaxing the stringent margin requirement and accommodating certain misclassifications or deviations. This approach balances margin maximization and misclassification penalties.
- C: The regularization parameter C in SVM strikes a balance between margin maximization and the cost of misclassification. It determines the penalty imposed for exceeding the margin or misclassifying data points. A higher value of C enforces a stricter penalty, potentially leading to a narrower margin and reduced misclassifications.
- Hinge Loss: Hinge loss represents a common loss function within SVMs. It penalizes incorrect classifications and margin violations. Frequently, the objective function in SVM combines hinge loss with a regularization term.
- Dual Problem: The dual problem in optimization pertains to identifying the Lagrange multipliers associated with support vectors. Solving this problem aids in solving the SVM. The dual formulation enables the utilization of kernel tricks and more efficient computations.

Figure 3, explains the general structure of the Support Vector Machine algorithm, whereas in Table 3, presents the positive and negative aspects of the Support Vector Machine.

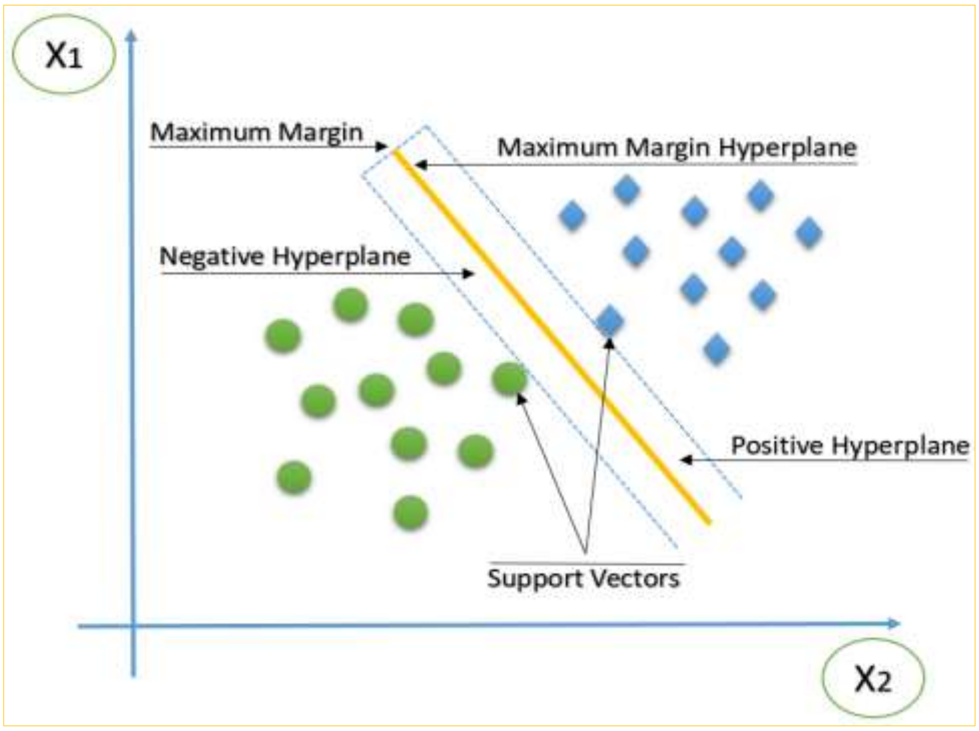


Figure 3. The general structure of a Support Vector Machine.

Table 3. Support Vector Machine advantages and disadvantages.

Support Vector Machine	
Positive aspects	**Negative aspects**
SVM works relatively well when there is a clear margin of separation between classes.	SVM algorithm is not suitable for large data sets.
SVM is more effective in high dimensional spaces.	SVM does not perform very well when the data set has more noise i.e. target classes are overlapping.
SVM is effective in cases where the number of dimensions is greater than the number of samples.	If the number of features for each data point exceeds the number of training data samples, the SVM will underperform.
SVM is relatively memory efficient.	As the support vector classifier works by putting data points, above and below the classifying hyperplane there is no probabilistic explanation for the classification.

2.2.4 Logistic Regression

Logistic regression stands out as one of the most widely employed machine learning algorithms, firmly situated within the domain of supervised learning. Its primary purpose is to forecast categorical dependent variables through the utilization of a specified set of independent variables.

In logistic regression, the objective is to predict the outcome of a categorical dependent variable, necessitating the output to be categorical or discrete in nature. This can manifest as binary choices, such as Yes or No, 0 or 1, true or false, and so forth. However, rather than furnishing exact 0 and 1 values, logistic regression yields probabilistic values that fall within the range of 0 to 1. Comparatively, logistic regression shares similarities with linear regression, diverging mainly in their respective applications. Linear regression finds its utility in addressing regression problems, while logistic regression specializes in addressing classification problems (Wang et al., 2019).

In logistic regression, the process involves fitting an "S"-shaped logistic function rather than a regression line, facilitating predictions of two maximum values, typically denoting 0 or 1. The curve generated by the logistic function conveys the likelihood of specific outcomes, such as identifying whether a patient suffers a medical condition (e.g., heart failure) or not, determining if a person is obese based on their weight and height, and similar scenarios. Logistic regression holds a position of significance within the realm of machine learning due to its capacity to provide probability estimates and effectively classify new data, accommodating both continuous and discrete datasets (Boateng & Abaye, 2019).

Furthermore, logistic regression can be employed to classify observations across diverse types of data, and it readily identifies the most influential variables contributing to the classification process. The Figure 4, depicts the logistic regression function.

The sigmoid function represents a crucial mathematical tool employed to transform predicted values into probability estimates. It possesses the capacity to convert any real number into a value confined within the range of 0 and 1 (Christodoulou et al., 2019).

More extensively, in the context of logistic regression, the central requirement is that the resultant values must fall within the narrow confines of 0 and 1, adhering rigorously to this limit. Consequently, this constraint manifests itself in the formation of a distinctive curve, recognized as the sigmoid function or logistic function, often resembling the shape of the letter "S." Within logistic regression, we rely on the concept of a threshold value, which serves as a critical determinant of the assigned probability, either 0 or 1. In this context, values surpassing the threshold tend to converge toward 1, while values residing below the threshold tend to gravitate towards 0. This pivotal threshold value plays a pivotal role in the classification process (Leukel et al., 2022).

To perform logistic regression, certain assumptions must be satisfied. First and foremost, the dependent variable must exhibit categorical characteristics. Additionally, it is essential that the independent variables do not demonstrate multicollinearity.

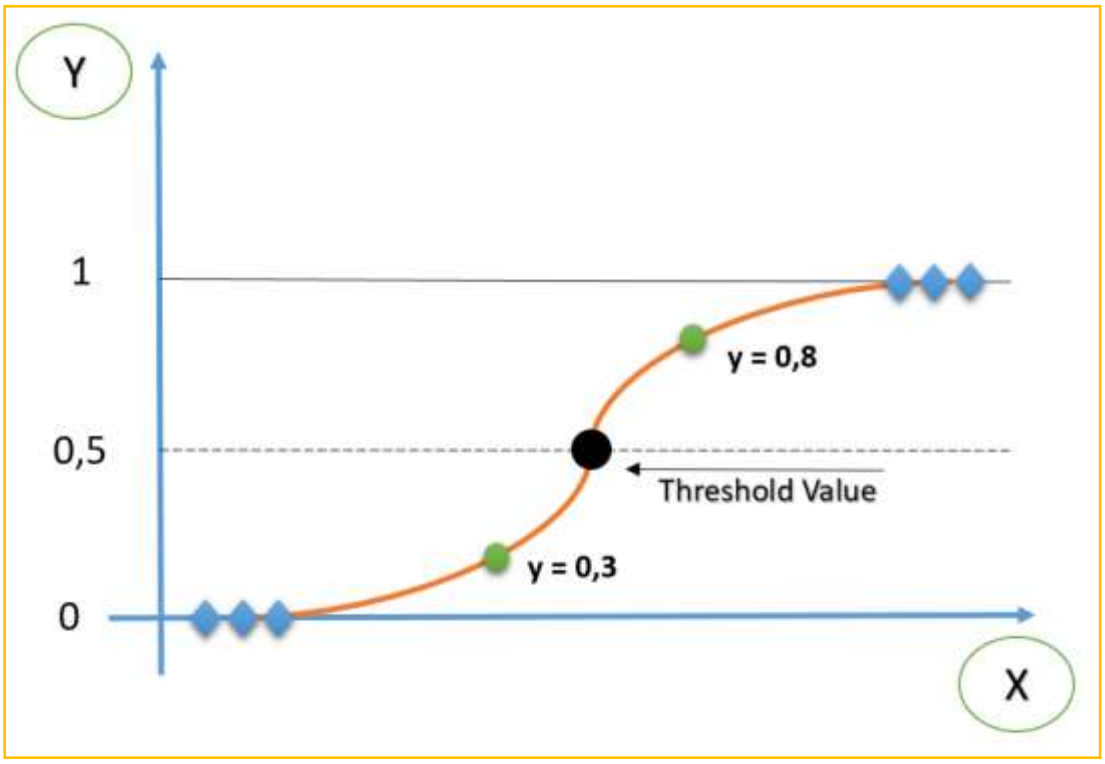


Figure 4. Logistic -Sigmoid function.

The Logistic regression equation can be derived from the Linear Regression equation through a set of mathematical steps. The following outlines the process for obtaining the Logistic Regression equation (Ranganathan et al., 2017). The equation of the straight line can be written as:

$$y = bo += b1 * x1 + b2 * x2 + b3 * x3 + \cdots + bn * xn$$

In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; 0\ for\ y = 0\ and\ infinity\ for\ y = 1$$

But range is needed between -[infinity] to +[infinity], then take logarithm 1of the equation it will become:

$$\log\left[\frac{y}{y-1}\right] = bo + b1 * x1 + b2 * x2 + b3 * x3 + \cdots + bn * xn$$

The above equation is the final equation for Logistic Regression.

Logistic Regression can be categorized into three distinct types based on the nature of the dependent variable (Binder et al., 2019).

→ Binomial Logistic Regression: In binomial logistic regression, the dependent variable can assume only two distinct categories or levels. Examples of such binary outcomes include 0 or 1, Pass or Fail, Yes or No, and so forth. Binomial logistic regression is used when the outcome variable is binary and unordered.

→ Multinomial Logistic Regression: Multinomial logistic regression comes into play when the dependent variable has three or more possible categories or classes that are unordered. For instance, if the outcome variable includes categories like "cat," "dogs," and "sheep," this type of logistic regression is suitable for modeling such scenarios. Multinomial logistic regression addresses cases where the categories are mutually exclusive but lack any inherent order.

→ Ordinal Logistic Regression: Ordinal logistic regression is employed when the dependent variable exhibits three or more categories that are not only unordered but also possess a natural order or hierarchy. Examples could include categories like "low," "Medium," or "High." This type of logistic regression accommodates scenarios where the categories maintain a meaningful rank or progression.

These three types of logistic regression allow for the modeling of different types of categorical dependent variables, depending on the nature of the data and the research question at hand.

Table 4, presents the positive and negative aspects of the Logistic regression.

Table 4. Logistic regression advantages and disadvantages.

Logistic Regression	
Positive aspects	**Negative aspects**
Logistic regression is straightforward to implement, interpret, and efficient in training.	When the number of observations is smaller than the number of features, it is advisable to avoid using Logistic Regression as it can potentially result in overfitting.
It does not assume any specific class distribution in the feature space.	It creates linear boundaries.
It can readily expand to handle multiple classes (multinomial regression) and offers a natural probabilistic perspective on predicting classes.	A primary limitation of Logistic Regression is its assumption of linearity between the dependent variable and the independent variables.
It not only offers insight into the relevance of a predictor (coefficient magnitude) but also its association direction (positive or negative).	Logistic Regression can exclusively be employed to predict discrete functions, implying that the dependent variable in Logistic Regression is constrained to a discrete numerical set.
It efficiently classifies unknown records with speed.	Logistic Regression is ill-suited for addressing non-linear problems due to its reliance on a linear decision surface. In practice, finding linearly separable data in real-world scenarios is a rare occurrence.
It exhibits high accuracy on straightforward datasets and performs effectively when the dataset allows for linear separation.	Logistic Regression necessitates either the absence of multicollinearity or its moderate presence among independent variables.
Model coefficients can be interpreted as indicators of the	Obtaining complex relationships using logistic regression can be challenging.

significance of features.	More robust and concise algorithms, such as Neural Networks, often surpass the performance of logistic regression in such cases.
Logistic regression has a lower tendency to overfit, but in high-dimensional datasets, it can exhibit overfitting.	Logistic Regression requires that the independent variables are linearly related to the log odds (log(p/(1-p)), where 'p' represents the probability of an event occurring.

2.2.5 Linear Regression

Linear regression stands as one of the most accessible and widely used Machine Learning algorithms, primarily employed for predictive analysis. It is a statistical technique tailored for making predictions concerning continuous or numeric variables, such as age, prices, sales etc. (Seber & Lee, 2003).

Linear regression hinges on the establishment of a linear relationship between a dependent variable (usually denoted as 'y') and one or more independent variables (typically denoted as 'x'). Consequently, it is termed "linear regression" because it quantifies how changes in the independent variable(s) correspond to alterations in the dependent variable (Montgomery et al., 2021).

The linear regression model, in essence, generates a straight-line representation, which depicts the relationship between these variables. The Figure 5, depicts the linear regression model.

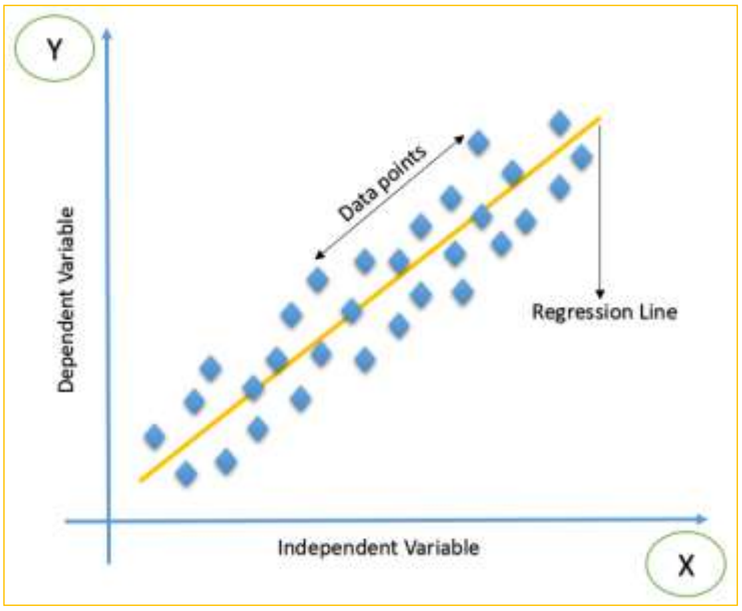


Figure 5. Linear regression model.

Mathematically, a linear regression we can be represented as follows:

$$y = a_0 + a_1 * x + e$$

Where, Y is the dependent Variable (Target Variable), X is the independent Variable (predictor Variable), a0 is the intercept of the line (Gives an additional degree of freedom), a1 is the Linear regression coefficient (scale factor to each input value) and e = random error. The values for x and y variables are training datasets for Linear Regression model representation. (Hope, 2020).

Linear regression can be subdivided into two distinct types of algorithms based on the number of independent variables involved (Maulud & Abdulazeez, 2020).

1. Simple Linear Regression. When a single independent variable is utilized to forecast the value of a numerical dependent variable, the resulting Linear Regression algorithm is termed Simple Linear Regression.
2. Multiple Linear Regression. In cases where more than one independent variable are employed to predict the value of a numerical dependent variable, the corresponding Linear Regression algorithm is referred to as Multiple Linear Regression.

Before running a regression analysis, regression assumptions should be tested. More specifically, the relationship between independent and dependent variables must be linear, there should be no multicollinearity, the values of the residuals must be independent, the variance of the residuals should be constant and the residuals should be normally distributed. When it’s tested that all the regression assumptions were met, a regression analysis can be conducted (Schmidt & Finan, 2018). Table 5, presents the positive and negative aspects of the Linear regression.

Table 5. Linear regression advantages and disadvantages.

Linear Regression	
Positive aspects	**Negative aspects**
Simple implementation.	Prone to underfitting.
Computationally efficient.	Prone to noise and overfitting.
Performance on linearly seperable datasets.	Sensitive to outliers.
Overfitting can be reduced by regularization.	Linear Regression assumes that the data is independent.

2.2.6 k-Nearest Neighbor (kNN)

K-Nearest Neighbors (K-NN) is among the simplest Machine Learning algorithms employed within the realm of Supervised Learning. This algorithm operates on the principle of assuming similarity between new data and existing data points, subsequently placing the new data into the category that most closely resembles the established categories. K-NN is characterized by its approach of retaining all available data and determining the classification of a new data point based on its similarity to existing data. Consequently, when new data emerges, it can be efficiently categorized into an appropriate class through the utilization of the K-NN algorithm (Agrawal, 2014).

K-NN is versatile in its application, serving both as a tool for regression and classification tasks, albeit it is more commonly utilized for classification challenges. One notable feature of K-NN is its non-parametric nature, signifying that it refrains from making any assumptions about the underlying data distribution. This algorithm is often referred to as a "lazy learner" as it abstains from immediate learning from the training set. Instead, it stores the dataset and takes action during the classification phase. During training, K-NN simply retains the dataset, and when presented with new data, it classifies that data into a category closely resembling the new data (Zhang et al., 2017).

The operation of the K-Nearest Neighbors (K-NN) algorithm can be elucidated through the following steps (Dhanabal & Chandramathi, 2011):

Step 1. Select the number K of the neighbors.

Step 2. Calculate the distance of K number of neighbors. There are various methods for calculating the distance between a new point (x) and an existing point (y), the most widely used ones are:

- Euclidean Distance, is calculated as the square root of the sum of the squared differences between a new point (x) and an existing point (y).

$$d(x,y) = \sqrt{\sum_{j=1}^{n}(x_j - y_j)^2}$$

- Manhattan Distance, is the distance between real vectors using the sum of their absolute difference.

$$d(x,y) = \sum_{j=1}^{n} |x_j - y_j|$$

- Hamming Distance: It is used for categorical variables. If the value (x) and the value (y) are the same, the distance D will be equal to 0. Otherwise, D=1.

$$d(x,y) = \sum_{j=1}^{n} |x_j - y_j|$$

$$x = y \rightarrow d(x,y) = 0$$

$$x \neq y \rightarrow d(x,y) = 1$$

- Minkowski Distance: It is a metric intended for real-valued vector spaces. We can calculate Minkowski distance only in a normed vector space, which means in a space where distances can be represented as a vector that has a length and the lengths cannot be negative. There are a few conditions that the distance metric must satisfy:
  1. Non-negativity: d(x, y) >= 0
  2. Identity: d(x, y) = 0 if and only if x == y
  3. Symmetry: d(x, y) = d(y, x)
  4. Triangle Inequality: d(x, y) + d(y, z) >= d(x, z)

$$d(x,y) = \sqrt[p]{\sum_{j=1}^{n} |x_j - y_j|^p}$$

The parameter "p" in the formula can be adjusted to produce various types of distances, such as:

- p = 1, when p is set to 1 we get Manhattan distance,
- p = 2, when p is set to 2 we get Euclidean distance.

Step 3. Take the K nearest neighbors as per the calculated Euclidean distance.

Step 4. Among these k neighbors, count the number of the data points in each category.

Step 5. Assign the new data points to that category for which the number of the neighbor is maximum.

Step 6. Our model is ready.

In order to gain better understanding in the k-Nearest Neighbor algorithm, Figure 6 depicts its function.

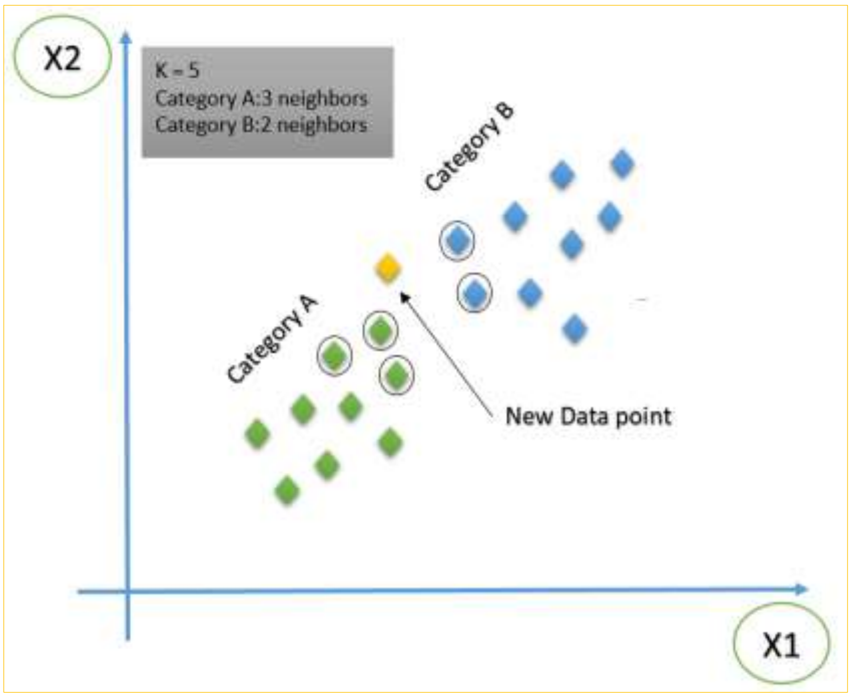


Figure 6. k-Nearest Neighbor model.

Determining the optimal "K" value lacks a definitive method; hence, we must experiment with various values to identify the most favorable one. Typically, a value of 5 is considered the most suitable. In other words, determining the optimal "K" value lacks a definitive method; hence, we must experiment with various values to identify the most favorable one. Typically, a value of 5 is considered the most suitable. Setting "K" to an exceedingly small value, like 1 or 2, can introduce noise and make the model susceptible to the influence of outliers. On the other hand, employing large "K" values has its merits, but it may encounter certain challenges (Tharwat et al., 2018).

Table 6, presents the positive and negative aspects of the k-Nearest Neighbor algorithm.

Table 6. k-Nearest Neighbor advantages and disadvantages.

k-Nearest Neighbor	
Positive aspects	**Negative aspects**
It's easy to understand and simple to implement.	Associated computation cost is high as it stores all the training data.
It can be used for both classification and regression problems.	Requires high memory storage.
It's ideal for non-linear data since there's no assumption about underlying data.	Need to determine the value of K.
It can naturally handle multi-class cases.	The computation cost is high because of calculating the distance between the data points for all the training samples.
It can perform well with enough representative data	Sensitive to irrelevant features.

2.2.7 Gradient Boosting

Gradient Boosting is a combination of two techniques: Gradient Descent and Boosting. In Gradient Boosting, each new model minimizes the loss function from the previous one using the Gradient Descent Method (Bentéjac et al., 2021).

- Boosting is an ensemble method that sequentially combines multiple weak learners to create a robust and powerful learner. In boosting, the predictors are trained one after another, with each subsequent predictor aiming to rectify the errors of its predecessor (Nayak & Sharma, 2023).
- Gradient Descent is widely recognized as one of the most commonly employed optimization algorithms for training machine learning models, with the aim of minimizing the discrepancies between actual and anticipated outcomes. Its primary purpose is to identify the local minimum of a parameterized function, denoted as f(x) (). Defining the local minimum or local maximum of a function using gradient descent entails the following principles:
  - Moving in the direction of the negative gradient or away from the gradient of

the function at the present point leads to the discovery of the local minimum.

- Progressing in the direction of the positive gradient or toward the gradient of the function at the current point results in the identification of the local maximum.

The fundamental objective of employing the gradient descent algorithm is to iteratively diminish the cost function. This cost function serves as a quantification of the dissimilarity or error between actual and anticipated values at the current position, represented as a single real number. It plays a pivotal role in enhancing the efficiency of machine learning models by offering feedback, thereby facilitating error reduction and the identification of local or global minima (Ruder, 2016).

In order to gain better understanding in the Gradient Descent algorithm, Figure 7 depicts its function.

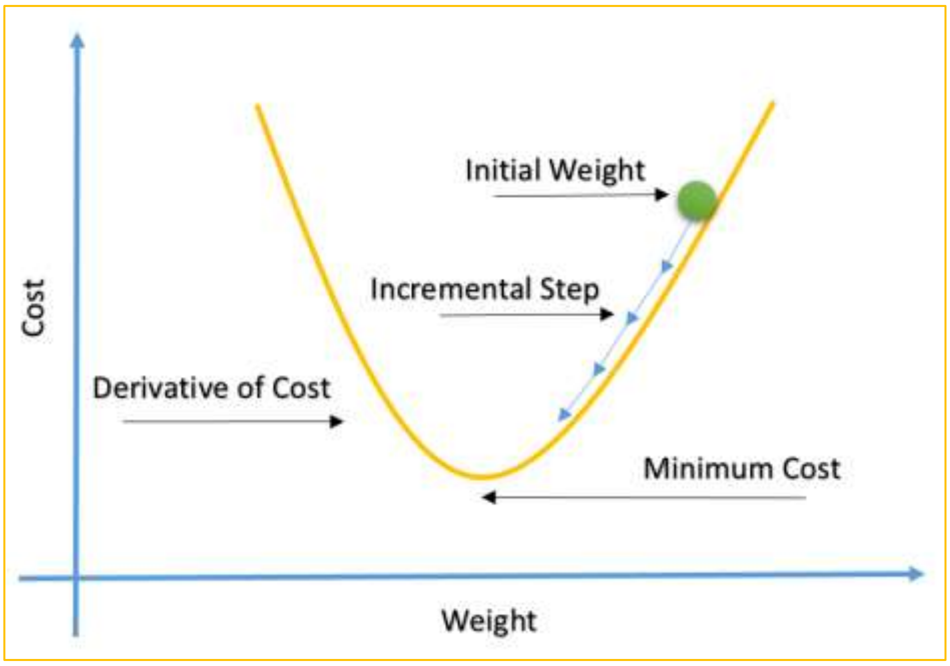


Figure 7. Gradient Descent algorithm.

Briefly, in gradient boosting, the process involves each new model minimizing the loss function in comparison to its predecessor through the use of the Gradient Descent Method. This iterative procedure persists until a more optimal estimation of the target variable is attained. Distinguishing it from other ensemble techniques, the core concept in gradient boosting revolves around constructing a sequence of trees. Each subsequent tree endeavors to rectify the errors made by its predecessor tree (Chen & Shi, 2023).

Table 7, presents the positive and negative aspects of the Gradient Boosting algorithm.

Table 7. Gradient Boosting advantages and disadvantages.

Gradient Boosting	
Positive aspects	**Negative aspects**
Can support various loss functions and provides a number of hyperparameters tuning options which it makes it very flexible.	It minimizes all errors, hence prone to over-fitting. One must use cross-validation to neutralize.
It works great with numerical as well as categorical features as it is.	This technique often requires many trees; hence it can be time and memory exhaustive.
No data imputation is required.	

2.3 Preprocessing Techniques

2.3.1 Generally

The real-world data that we have to proceed and draw information from them is incomplete, inconsistent, inaccurate (contains errors or outliers), and often lacks specific attribute values/trends. This happens due to manual errors, unexpected events, technical issues, or a variety of other obstacles. While working on these data, it is important to know the types of data to process them and get the right results (Vijayarani et al., 2015).

The data can be categorized into two main types:

1. Qualitative or Categorical Data (Hancock & Khoshgoftaar, 2020)

- *Nominal Data*: Nominal data serves to label variables without exhibiting any specific order or numerical value. Examples encompass gender, marital status, nationalities, individual names, and the like.
- *Ordinal Data*: Ordinal data organizes variables into ranked categories, possessing a natural hierarchy based on some scale, such as from high to low. Instances include letter grades in examinations (A, B, C, D, etc.) and educational

levels (Higher, Secondary, Primary).

2. Quantitative or Numerical Data (Famili et al., 1997)

- *Discrete Data*: Discrete data encompasses values represented by integers or whole numbers. An illustration is the total count of students in a class. These values cannot be divided into decimal or fractional parts, and they are finite and countable in nature.
- *Continuous Data*: Continuous data manifests in the form of fractional numbers and can be divided into smaller units. It represents information that can take any value within a specified range. Examples include temperature, height, width, time, speed, and similar variables.

It is evident that nearly any information can be transformed into data. This implies that our data, in addition to potential errors like missing values and outliers, may also exhibit diverse types. Algorithms are typically not equipped to handle incomplete or noisy data, as they can disrupt the accurate representation of the sample. Data preprocessing addresses these issues through a comprehensive treatment of the available data (García et al., 2016).

2.3.2 Tools and Libraries

The process of preparing data for analysis can be streamlined with the aid of tools and libraries, simplifying management and execution. In the absence of specific libraries, crafting concise solutions can become a time-intensive coding task, demanding hours of development and optimization (Jansen et al., 2023).

- Data Preprocessing with Python: Python, a versatile programming language, boasts numerous open-source libraries capable of executing complex operations in just a single line of code. The available functions for data preprocessing are extensive.
- Autumunge: Autumunge is an excellent Python library platform designed to efficiently prepare tabular data for direct application in machine learning algorithms.
- Data Preprocessing with R: R, primarily utilized for research and academic purposes, parallels Python by offering a range of packages, similar to libraries, which significantly support data preprocessing steps.
- Data Preprocessing with Weka: Weka is a comprehensive software solution facilitating data mining and preprocessing, featuring integrated tools for intelligent

mining and machine learning models.

- Data Preprocessing with RapidMiner: Similar to Weka, RapidMiner is an open-source software equipped with a variety of effective tools designed to facilitate data preprocessing.

2.3.3 Purpose of Data Preprocessing

Once the data has been appropriately collected, it must undergo exploration or assessment to identify significant trends and discrepancies. The primary objectives of Data Quality Assessment include (García et al., 2015):

- Get Data Overview: This involves comprehending the data formats and overall structure in which the information is stored. Additionally, it encompasses determining data properties like mean, median, standard quantiles, and standard deviation. These particulars aid in pinpointing irregularities within the data.
- Detect Missing Data: It is common for real-world datasets to contain missing data. This can disrupt the genuine patterns within the data, potentially leading to further loss if entire rows or columns are removed due to a few absent cells in the dataset.
- Identify Outliers or Unusual Data: Certain data points may significantly deviate from the prevailing data patterns, classifying them as outliers. It might be necessary to exclude these points for more accurate predictions, unless the algorithm's primary purpose is to detect anomalies.
- Remove Inconsistencies: Similar to missing values, real-world data often harbors various inconsistencies such as incorrect spellings, erroneously populated columns and rows (e.g., salary inputted in the gender column), duplicated data, and more. At times, automation can address these discrepancies, but frequently they necessitate manual verification.

2.3.4 Dealing with Missing Values

Missing values are a common challenge in real-world datasets due to physical and manual constraints associated with data collection. For instance, if data is gathered through sensors, there may be instances where the sensor temporarily stops working, resulting in missing data. Different datasets may encounter various issues leading to missing data points (Raja & Thangavel, 2020).

To effectively utilize available data, it's essential to address these missing values. Here are some proven strategies (Sessa & Syed, 2016):

- Drop Samples with Missing Values: This approach is beneficial when the number of samples is substantial and the count of missing values in a given sample is high. However, it's not recommended in other cases as it can lead to significant data loss.
- Replace Missing Values with Zero: This technique can be effective for basic datasets, where zero can signify the absence of a value. Nevertheless, in many cases, zero may have its own meaning. For example, in temperature data from a tropical region, zero may not accurately represent a missing value. It's best used when the dataset is independent of its effect, such as in phone bill data.
- Replace Missing Value with Mean, Median, or Mode: Using statistical functions like mean, median, or mode can address the issue of using zero incorrectly. While these values are also assumptions, they tend to provide more meaningful approximations compared to a single value like zero.
- Interpolate Missing Values: Interpolation generates values within a range based on a given step size. For instance, if there are nine missing values between cells with values ranging from 0 to 10, interpolation will fill in the missing cells with numbers from 1 to 9. It's important to ensure the dataset is sorted according to a more reliable variable (like serial number) before applying interpolation.
- Extrapolate Missing Values: Extrapolation populates values that fall outside a given range, such as extreme values of a feature. It relies on another variable, typically the target variable, to compare and populate the variable in question with a guided reference.
- Build a model with other features to predict the missing values: By far the most intuitive of all techniques we've mentioned. Here, an algorithm studies all the variables except the actual target variable (since that would lead to data leakage). The target variable for this algorithm becomes the feature with missing values. The model, if well trained, can predict the missing points and provide the closest approximations.

2.3.5 Scaling

Columns in a dataset can have varying ranges. For instance, one column may represent distances, while another may represent currency units. These columns will exhibit markedly different numerical ranges, which can pose challenges for machine learning models in achieving optimal computations. To address this, several popular scaling techniques are employed (Ahsan et al., 2021):

- Min-Max Scaler: This technique rescales feature values to fit within a specified range, such as between 0 and 5.
- Standard Scaler: The standard scaler assumes that the variable follows a normal distribution. It then scales the data to have a standard deviation of 1, with the distribution centered at 0.
- Robust Scaler: This scaler is particularly effective when the dataset contains outliers. It scales the data based on the inter-quartile range after removing the median.
- Max-Abs Scaler: Similar to the min-max scaler, this technique scales the feature to its maximum absolute value. Notably, it preserves the sparsity of the data by not centering it.

2.3.6 Dealing with outliers

Outliers are data points that do not conform with the predominant pattern observed in the data. They can cause disruptions in the predictions by taking the calculations off the actual pattern (Nnamoko & Korkontzelos, 2020).

Box plots are a valuable tool for detecting and addressing outliers. They enable the identification of key statistical measures like the median, interquartile ranges, and outliers. To effectively manage outliers, it's important to take note of the maximum and minimum ranges, and subsequently filter the variable accordingly (Krishna et al., 2022).

In order to gain better understanding regarding the Box plots, Figure 8 depicts its function.

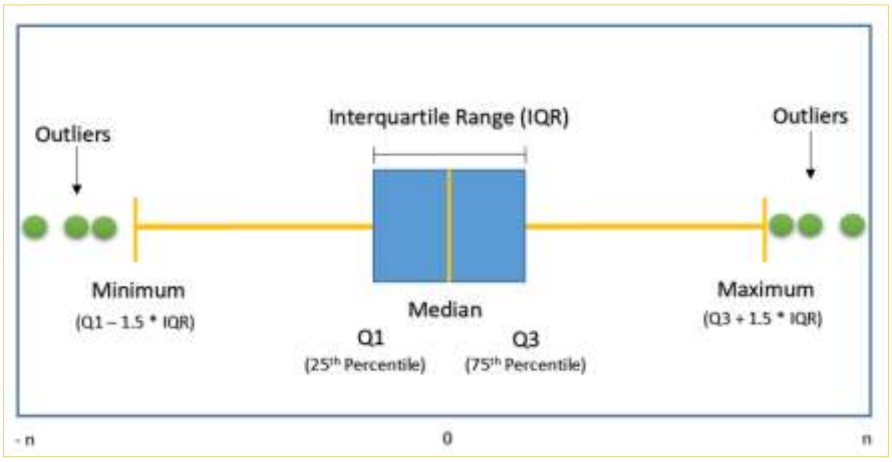


Figure 8. Box Plot.

To better understand the above figure (Williamson et al., 1989),

- median (Q2/50th Percentile): the middle value of the dataset.
- first quartile (Q1/25th Percentile): the middle number between the smallest number (not the "minimum") and the median of the dataset.
- third quartile (Q3/75th Percentile): the middle value between the median and the highest value (not the "maximum") of the dataset.
- **InterQuartile Range (IQR)**: 25th to the 75th percentile. IQR tells how spread the middle values are.
- "maximum": Q3 + 1.5*IQR
- "minimum": Q1 -1.5*IQR
- **Outliers**: (shown as green circles) In statistics, an outlier is an observation point that is distant from other observations.

Additionally, it is essential to clarify that not every outlier is a *wrong* value.

2.3.7 Feature Encoding

At times, data is presented in a format that isn't directly interpretable by machines. For instance, a column containing string values like names might not hold meaning for a model that relies solely on numerical inputs. This necessitates the process of data transformation to facilitate the model's comprehension. This technique is referred to as categorical encoding (Dahouda & Joe, 2021). There are several approaches to encoding categories. Here are some fundamental methods to begin with:

- Label/Ordinal Encoding: This method assigns values from 1 to 'n' in a sequential order. Here, 'n' corresponds to the number of samples in the column. For instance, if a column contains three city names, label encoding will assign values 1, 2, and 3 to the respective cities. While this method is suitable for ordered categories like student grades, it's not recommended for categorical values without a natural order, such as cities.
- One-Hot Encoding: When categorical data lacks a natural order, one-hot encoding is employed. This technique generates a distinct column for each category. A positive value (1) is assigned in the row where the category is present, and 0 denotes its absence. It's worth noting that this method may lead to data expansion, but it's typically not problematic with a manageable number of features.

- Binary Encoding: This method addresses the potential bulkiness associated with one-hot encoding. Each categorical value is transformed into its binary representation, resulting in the creation of new columns for each binary digit. This compresses the number of columns compared to one-hot encoding. For instance, with 100 values in a categorical column, one-hot encoding would generate 100 (or 99) new columns, whereas binary encoding would yield considerably fewer, unless the values are exceedingly large.
- BaseN Encoding: Similar to binary encoding, BaseN encoding utilizes a different base, allowing for a range of options beyond binary (base 2). The choice of base impacts the trade-off between information loss and compression efficiency. Higher bases result in greater compression power, but also entail increased information loss.
- Hashing: This method involves generating values from a category using mathematical functions. It's akin to one-hot encoding, albeit with a more intricate function and fewer dimensions. However, hashing does entail some information loss due to collisions in resulting values.

2.3.8 Dealing with Imbalanced Data

In binary classification problems, imbalanced datasets present a common challenge for machine learning practitioners. Resampling data is a widely favored approach to tackle this issue, with two primary methods: Undersampling and Oversampling. Among these, Synthetic Minority Oversampling Technique (SMOTE) stands out as a popular technique (Garcia et al., 2012).

SMOTE is specifically designed to address imbalanced datasets by generating synthetic samples for the minority class. This technique helps mitigate bias and capture crucial features of the minority class, ultimately leading to more accurate predictions and improved model performance (Hussein et al., 2019).

The mechanism behind SMOTE involves creating synthetic samples along the lines connecting the nearest neighbors in the feature space. The fundamental concept is to generate new samples for the minority class by taking small steps from one of the minority class samples towards one of its k nearest neighbors, where k is a parameter of the algorithm (Fernández et al., 2018).

The algorithm operates in the following steps (Pradipta et al., 2021):
Step 1. Select a minority class sample from the original dataset.
Step 2. Identify its k nearest minority class neighbors in the feature space.
Step 3. Randomly choose one of the k nearest neighbors.
Step 4. Generate a new synthetic sample by interpolating between the selected minority class sample and the randomly chosen neighbor.
Step 5. Repeat steps 1-4 until the desired number of synthetic samples is created.

This process results in new synthetic data that shares similarities with the minority class samples in the feature space, yet is distinct from any existing samples.

2.3.9 Dimensionality Reduction

Dimensionality refers to the number of input features, variables, or columns present within a dataset. The process of reducing these features is known as dimensionality reduction. Datasets often encompass a multitude of input features in various instances, making the task of predictive modeling considerably more complex. Managing and visualizing a high-dimensional training dataset can be challenging. In such scenarios, the employment of dimensionality reduction techniques becomes imperative (Guyon & Elisseeff, 2003).

Dimensionality reduction techniques can be defined as follows: "They are methods for transforming a high-dimensional dataset into a lower-dimensional one, with the objective of retaining similar information". These techniques find widespread application in machine learning, particularly in improving predictive models when dealing with classification and regression problems. Some of the common techniques of dimensionality reduction are: Principal Component Analysis, Backward Elimination, Forward Selection, Score comparison, Missing Value Ratio, Low Variance Filter, High Correlation Filter, Random Forest and Factor Analysis (Huang et al., 2019).

2.4 Evaluation of Machine Learning Models

2.4.1 Generally

Evaluation metrics serve as essential tools for assessing the quality of statistical or machine learning models. The concept behind building machine learning models is rooted in a feedback loop that fosters improvement. Evaluation metrics provide insights into a model's performance (Zhou et al., 2021).

A crucial attribute of evaluation metrics lies in their ability to distinguish between different outcomes generated by a model. Evaluation metrics offer valuable insights about your model, such as whether it has truly learned or merely memorized patterns. This distinction holds great significance because a model that has only memorized is efficient in handling known data but lacks adaptability and efficiency (Dalianis & Dalianis, 2018).

To ensure that the model genuinely learns, it is imperative to employ a variety of evaluation metrics. This approach is essential because a model might perform exceptionally well according to one evaluation metric, but its performance may decline when assessed using a different metric. Utilizing multiple evaluation metrics is pivotal in verifying that the model operates correctly and optimally (Rjoob et al., 2022).

2.4.2 Holdout and Cross validation methods

In the realm of machine learning, the dataset is typically divided into two distinct types: the "Training dataset" and the "Test dataset." The training dataset serves as the foundation for building and training the machine learning model to assess its functionality and performance. However, when it comes to evaluating the model, we rely on the test dataset, which consists of data samples that the model has never encountered during training (Perlaza et al., 2023).

The reason we use a separate test dataset for evaluation is to gauge how well the model performs on previously unseen or unknown data, essentially testing its ability to generalize from the training data to new, unfamiliar examples (Takano & Alaghband, 2019).

If we were to evaluate the model using the same training dataset, it would likely exhibit high accuracy measures for all instances within that dataset. However, this scenario can be misleading because the model is essentially predicting outcomes it has already learned, and it may not perform as effectively on new, real-world data. Therefore, using the training dataset for evaluation doesn't provide a genuine assessment of the model's ability to handle novel scenarios (Mahesh, 2020).

Two common methods for evaluating the performance of a model are the Holdout and the Cross validation methods.

I. Holdout method: The Holdout method is a technique used to assess the performance of a machine learning model. It involves dividing the dataset into two distinct sets: training data and testing data. The training data is used to train the model, while the testing data is employed to evaluate its performance. This method provides insight into how well the model, developed using various algorithm techniques, performs on previously unseen data. The Holdout approach is known for its simplicity, flexibility, and efficiency (Raschka, 2018).

II. Cross validation: Cross-Validation is a more comprehensive evaluation procedure. It entails partitioning the dataset into multiple subsets or "folds." The model is trained on a subset of the data and evaluated on the remaining data. Cross-validation helps assess the model's performance across various data subsets, providing a more robust estimate of its accuracy (Stone, 1978). There are different methods for conducting cross-validation, including:

   a. Validation: The given dataset is split into 50% of training and 50% for testing purpose. The main drawback in this method is that the remaining 50% of data that is subjected to testing may contain some crucial information that may be lost while training the model (Borg et al., 2018).
   b. Leave one out cross validation (LOOCV): All the datasets are trained in the model and a single data point is left for testing purpose. This method aims at exhibiting lower bias, but there are some chances that this method might fail because, the data-point that has been left out may be an outlier in the given data; and in that case we cannot produce better results with good accuracy (Syed, 2011).
   c. K-Fold Cross Validation: Is a popular method used for evaluation of a Machine Learning model. It works by splitting the data into k-parts. Each split of the data is called a fold. Here we train all the k subsets of data to the model, and then we leave out one (k-1) subset to perform evaluation on the trained model. This method results in high accuracy and produces data with less bias (Anguita et al., 2009).

2.4.3 Overfitting and Underfitting

When discussing machine learning models, the focus is often on their performance and accuracy, which is measured by prediction errors. A machine learning model is considered good when it effectively generalizes new input data from the problem domain, enabling accurate predictions for unseen future data (Zhang et al., 2019). However, issues arise in assessing how well a machine learning model learns and generalizes to new data, leading to problems like overfitting and underfitting. These issues are primarily responsible for suboptimal performance in machine learning algorithms (Jabbar & Khan, 2015).

More specifically, on the one hand underfitting in the context of statistical modeling or machine learning refers to a situation where the model's simplicity renders it incapable of adequately grasping the intricacies present in the data. This leads to suboptimal performance not only on the training dataset but also when the model is tested on new, previously unseen data. In simpler terms, an underfit model tends to produce inaccurate results, especially when confronted with novel examples. This issue predominantly arises when an overly simplistic model is employed, often characterized by overly generalized assumptions. To tackle the problem of underfitting, it is essential to opt for more intricate models that possess improved feature representation and employ less regularization (Cunningham & Delany, 2021).

On the other hand, overfitting is a phenomenon observed in statistical models when they fail to make accurate predictions on testing data. This occurs when a model is trained on an extensive dataset to the extent that it begins to learn from noise and inaccuracies present in the data, resulting in high variance when applied to test data. Consequently, the model struggles to correctly categorize data due to an excessive focus on fine details and noise (Roelofs et al., 2019). Overfitting is often associated with non-parametric and non-linear machine learning methods, as these algorithms possess greater flexibility in constructing models based on the dataset, sometimes leading to the creation of unrealistic models. To mitigate overfitting, one strategy is to use linear algorithms for linear data or to apply parameters like maximal depth when employing decision trees. This helps strike a balance between model complexity and generalization (Ying, 2019).

Figure 9, depicts the underfitting, goodfitting and overfitting of Machine learning models.

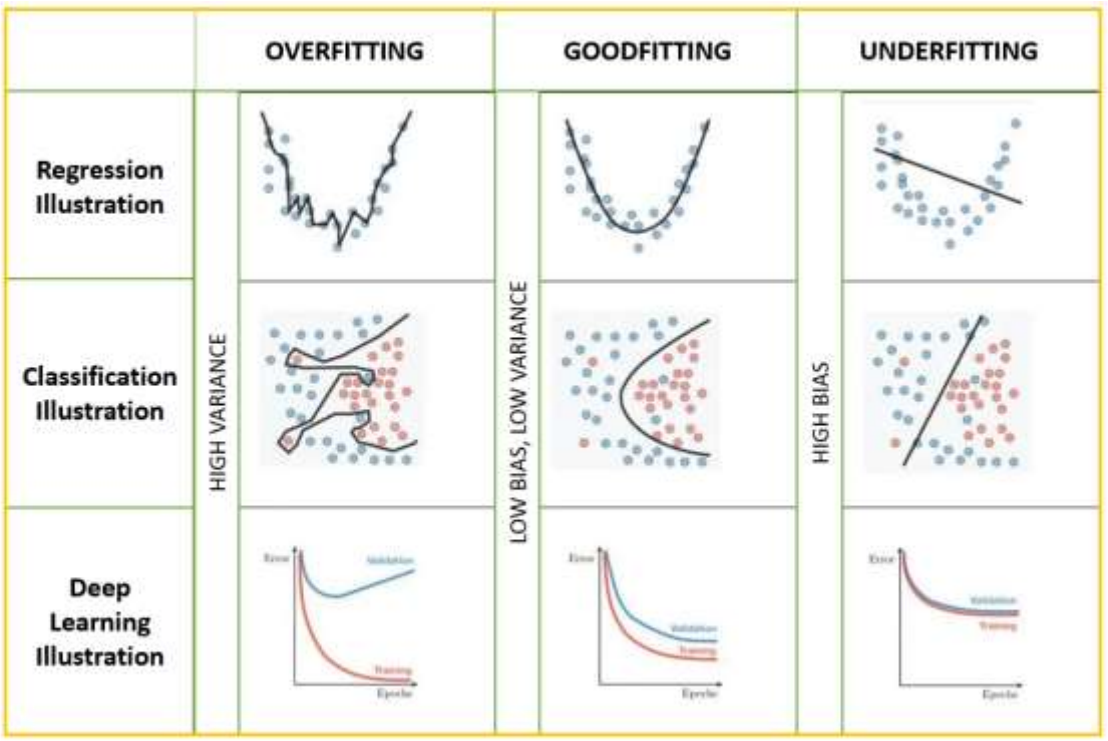


Figure 9. Underfitting, goodfitting and overfitting of Machine learning models.

2.4.3.1 *Bias and Variance in Machine Learning*

Bias refers to the error stemming from overly simplistic assumptions within the learning algorithm. These assumptions, while simplifying the model and its learning process, may not adequately capture the inherent complexities in the data. This error occurs when the model cannot accurately represent the genuine relationship between input and output. High bias is observed when a model performs poorly on both training and testing data, signifying underfitting due to its simplicity (Domingos, 2000).

Conversely, variance pertains to the error resulting from the model's sensitivity to variations within the training data. It quantifies the extent of prediction variability for different training data instances. High variance arises when a model captures the noise and random fluctuations in the training data rather than the underlying pattern. Consequently, the model excels on the training data but falters on the testing data, indicating overfitting (Valentini & Dietterich, 2004).

2.4.4 Classification Metrics

2.4.4.1 *Confusion Matrix*

A confusion matrix is a square NxN matrix structure employed for the evaluation of a classification model's performance. More extensively, N represents the number of classes that the model predicts. This matrix is applied to a test dataset for which the true values are known. Its purpose is to provide insight into the accuracy of a classifier by documenting the count of both correct and incorrect predictions (Choudhary & Gianey, 2017). Within the matrix, we can find key values (Mehrabi et al., 2021) such as:

- True Positives (TP) - The cases in which our predictions are true, and the actual output was also true.
- False Positives (FP) - The cases in which our predictions are true, and the actual output was false.
- True Negatives (TN) - The cases in which our predictions are false, and the actual output was also false.
- False Negatives (FN) - The cases in which our predictions are false, and the actual output was true.

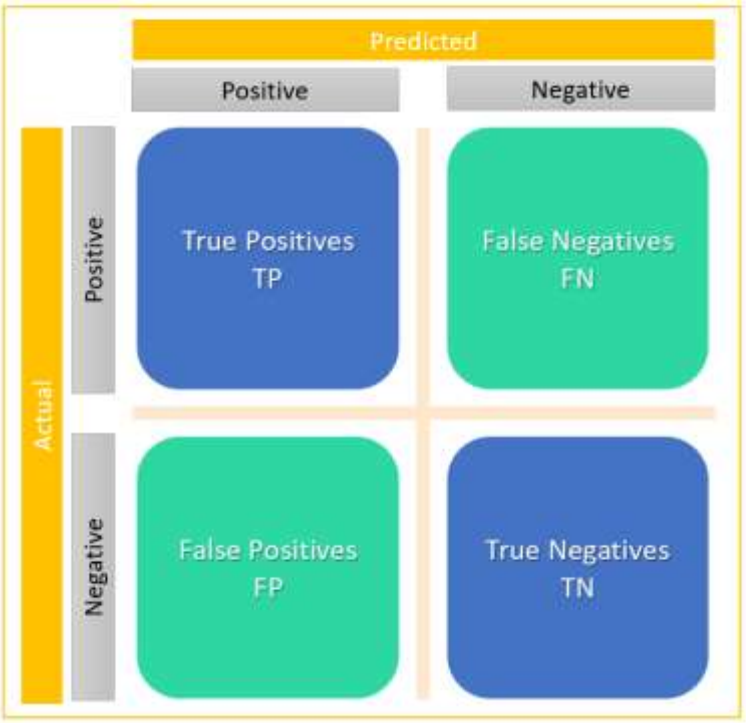


Figure 10. Confusion Matrix.

These elements collectively contribute to assessing the model's correctness and effectiveness in classifying data. Figure 10, depicts the confusion matrix classes.

2.4.4.2 *Accuracy*

Accuracy serves as a fundamental metric for assessing the performance of classification models. Formally, accuracy can be defined as the ratio of the sum of True Positives (correctly predicted positive instances) and True Negatives (correctly predicted negative instances) to the total number of predictions. In other words, accuracy measures how often the model's predictions are correct in relation to the entire dataset (Kotsiantis et al., 2007).

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions} = \frac{TP + TN}{TP + TN + FP + FN}$$

Accuracy alone may not provide a comprehensive understanding of model performance, especially when we are working with a class-imbalanced data set, where there is a significant disparity between the number of positive and negative labels.

2.4.4.3 *Precision*

Precision is a valuable metric in classification evaluation. It quantifies the ratio of True Positives (correctly predicted positive instances) in a sample to the total number of positive samples predicted by the classifier. In essence, precision provides insights into how accurately the model identifies positive samples, highlighting the fraction of predicted positives that are indeed true positives. This metric is particularly relevant in situations where the cost or impact of false positives is a critical consideration (Osisanwo et al., 2017).

$$Precision = \frac{TP}{TP + FP}$$

2.4.4.4 *Recall/ Sensitivity/ True Positive Rate*

The Recall also known as Sensitivity and True Positive Rate, expresses the percentage of positive instances out of the total actual positive instances. Therefore denominator (TP + FN) here is the actual number of positive instances present in the dataset.

$$Recall = \frac{TP}{TP + FN}$$

2.4.4.5 *Specificity*

Specificity, often referred to as the True Negative Rate, is an important metric in classification evaluation. It quantifies the ratio of True Negatives (correctly predicted negative instances) in a sample to the sum of True Negatives and False Positives (actual negative instances that were incorrectly classified as positives) within a given dataset.

In essence, specificity provides insights into how effectively the model identifies actual negative samples from the provided dataset. This metric helps assess the model's ability to correctly recognize instances that are genuinely negative, without misclassifying them as positive. Mathematically, specificity can be expressed as:

$$Specificity = \frac{TN}{TN + FP}$$

So, whether the term specificity or true negative rate is used, both are describing the same evaluation metric that focuses on the model's capacity to accurately identify true negatives within the dataset.

2.4.4.6 *F1 Score*

The F1 score, which is the harmonic mean of precision and recall, offers a balanced assessment of a classification model's performance. A higher F1 score indicates better overall performance. Notably, the F1 score is sensitive to changes in either precision or recall; if one of them decreases, it can significantly impact the final F1 score due to their product relationship.

A good F1 score is achieved when the model excels in both precision (correctly predicting positives among all predicted positives) and recall (not missing actual positives by predicting them as negatives). Mathematically, F1 Score can be expressed as:

$$F1\ Score = \frac{2 * Precision * Recall}{Precision + Recall}$$

2.4.4.7 *Area Under Curve (AUC) - Receiver Operating Characteristic (ROC) Curve*

The AUC-ROC curve serves as a performance measurement tool for classification problems across various threshold settings. The ROC (Receiver Operating Characteristic) curve is a graphical representation of the classifier's performance, while AUC (Area Under the Curve) quantifies the degree of separability between classes.

Specifically, the AUC value indicates how effectively the model can differentiate between classes. A higher AUC corresponds to a better ability of the model to correctly classify instances of class 0 as 0 and instances of class 1 as 1. In essence, a higher AUC signifies superior discrimination by the model.

To create the ROC curve, True Positive Rate (TPR), also known as sensitivity, is plotted on the y-axis, and False Positive Rate (FPR) is plotted on the x-axis. The ROC curve visually illustrates the trade-off between TPR and FPR at different threshold settings, helping to visualize the classifier's performance across various decision boundaries (Narkhede, 2018). Figure 11, depicts the graph the AUC-ROC.

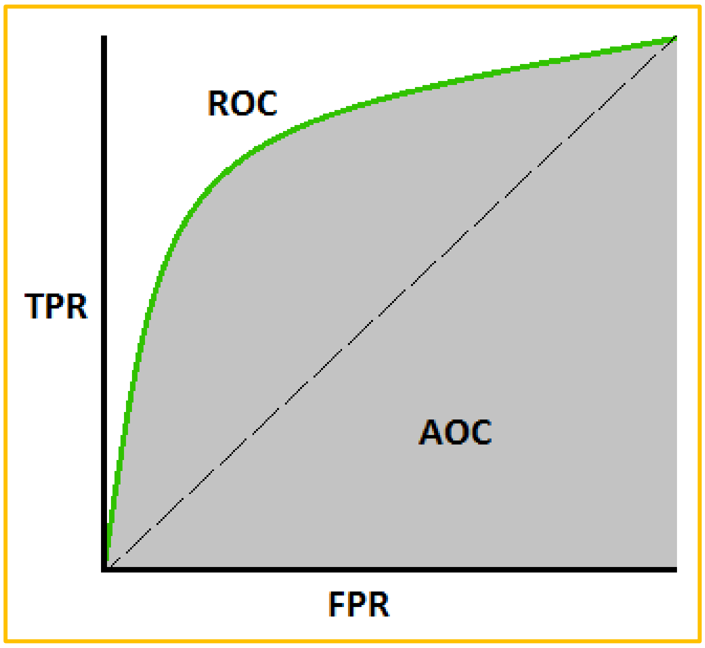


Figure 11. Graph of AUC – ROC.

2.4.5 Regression Metrics

Predicting outcomes with the assistance of correlated independent variables is a fundamental aspect of regression analysis. In this context, three primary metrics serve the purpose of assessing the model's performance and determining whether it suffers from underfitting or overfitting (Botchkarev, 2019). These metrics include:

- Mean Absolute Error (MAE): Mean Absolute Error represents the average discrepancy between predicted values and actual outcomes. It provides insight into the overall prediction accuracy but does not directly address issues related to underfitting or overfitting. Its calculation involves finding the mean of the absolute differences between predictions and actual values.
- Mean Squared Error (MSE): Mean Squared Error shares similarities with Mean Absolute Error but introduces an element of squared differences. This entails computing the average of the squared disparities between original and predicted values. Squaring helps in managing the impact of both small and large errors within the dataset.

- Root Mean Squared Error (RMSE): Root Mean Squared Error is a widely adopted metric for evaluating regression models. It is derived by taking the square root of the mean of the squared differences between predicted and actual values. This metric adheres to a normal distribution assumption and relies on the notion of unbiased errors.

These metrics serve as critical tools for gauging the effectiveness of regression models and guiding decisions on model fit and performance.

Chapter 3. DATA & METHODS

Chapter 3 describes the dataset and methods that are utilized in the present thesis. Specifically, the chapter starts with a description of the dataset and the sample. The chapter concludes with a description of the statistical methods that are used in order to analyze the data.

3.1 Dataset

For the purposes of this research a secondary data analyses is conducted. The examined dataset containing the medical records of 299 heart failure patients from the Faisalabad Institute of Cardiology and the Allied Hospital in Faisalabad, Punjab, Pakistan. The data was collected between April and December 2015 (Ahmad et al., 2017). Among these patients, 105 were women (label 0) and 194 were men (label 1), with ages ranging from 40 to 95 years old. All 299 patients had experienced left ventricular systolic dysfunction and had previous heart failures, placing them in classes III or IV of the New York Heart Association (NYHA) classification for heart failure stages.

This dataset includes 13 features, providing clinical, body, and lifestyle information (see Table 8). Some of these features are binary, indicating whether a patient has conditions like anemia, high blood pressure, diabetes, along with information about their sex and smoking habits. The diagnosis of anemia was based on hematocrit levels below 36%, as determined by the hospital physician. Unfortunately, the original dataset manuscript does not offer a specific definition for high blood pressure (Ahmad et al., 2017).

Among the features, Creatinine Phosphokinase (CPK) indicates the level of the CPK enzyme in the blood, which rises when muscle tissue is damaged (McClellan et al., 2002). Elevated CPK levels may suggest heart failure or injury (Özbay Karakuş & Er, 2022). Ejection fraction measures the percentage of blood the left ventricle pumps out with each contraction. Serum creatinine, a byproduct of muscle breakdown, is closely monitored by doctors to assess kidney function. High levels of serum creatinine can signal renal dysfunction. Sodium, a vital mineral for muscle and nerve function, is assessed through a routine blood test to check for normal levels. Abnormally low sodium levels may be linked to heart failure (Chicco & Jurman, 2020).

The 'death event' feature, used as the target in our binary classification study, indicates whether a patient passed away or survived before the average follow-up period of 130 days (Ahmad et al., 2017). Unfortunately, the original dataset article does not provide information about whether any patients had primary kidney disease or details about the follow-up procedure. In terms of dataset balance, there are 203 patients who survived (death event = 0) and 96 patients who did not (death event = 1). This translates to 32.11% positive cases and 67.89% negative cases in statistical terms.

Table 8. Meanings, measurement units, and intervals of each feature of the dataset.

Feature	Explanation	Measurement	Range
Age	Age of the patient	Years	[40, ..., 95]
Anaemia	Decrease of red blood cells or hemoglobin	Boolean	0, 1
High blood pressure	If a patient has hypertension	Boolean	0, 1
Creatinine phosphokinase (CPK)	Level of the CPK enzyme in the blood	mcg/L	[23, ..., 7861]
Diabetes	If the patient has diabetes	Boolean	0, 1
Ejection fraction	Percentage of blood leaving the heart at each contraction	Percentage	[14, ..., 80]
Sex	Woman or man	Binary	0, 1
Platelets	Platelets in the blood	kiloplatelets/mL	[25.01,...,850.00]
Serum creatinine	Level of creatinine in the blood	mg/dL	[0.50, ..., 9.40]
Serum sodium	Level of sodium in the blood	mEq/L	[114, ..., 148]
Smoking	If the patient smokes	Boolean	0, 1
Time	Follow-up period	Days	[4,...,285]
(target) death event	If the patient died during the follow-up period	Boolean	0, 1

mcg/L: micrograms per liter. mL: microliter. mEq/L: milliequivalents per litre

In the comprehensive analysis of the full dataset comprising heart failure patients, the distribution of key categorical features is detailed in the Table 9. The sample is categorized based on the presence or absence of specific conditions: Anaemia (0: false, 1: true), High blood pressure (0: false, 1: true), Diabetes (0: false, 1: true), Sex (0: woman, 1: man), and Smoking (0: false, 1: true).

Table 9. Statistical quantitative description of the categorical features.

	Full Sample		Dead patients		Survived patients	
Feature	Count	%	Count	%	Count	%
Anaemia (0: false)	170	56.86	50	52.08	120	59.11
Anaemia (1: true)	129	43.14	46	47.92	3	40.89
High blood pressure (0: false)	194	64.88	57	59.38	137	67.49
High blood pressure (1: true)	105	35.12	39	40.62	66	32.51
Diabetes (0: false)	174	58.19	56	58.33	118	58.13
Diabetes (1: true)	125	41.81	40	41.67	85	41.87
Sex (0: woman)	105	35.12	34	35.42	71	34.98
Sex (1: man)	194	64.88	62	64.58	132	65.02
Smoking (0: false)	203	67.89	66	68.75	137	67.49
Smoking (1: true)	96	32.11	30	31.25	66	32.51

Full sample: 299 individuals. Dead patients: 96 individuals. Survived patients: 203 individuals

The presented Table 10, offers a comprehensive statistical analysis of numeric features within a dataset, differentiating between two distinct groups: "Dead patients" and "Survived patients." These features encompass various clinical and demographic characteristics, and the statistics provided offer valuable insights into the central tendencies and variabilities exhibited within each group.

- Age Analysis – The median age for the full sample is 60.00 years, with a mean age of 60.83 years. Notably, the median age for Dead patients is slightly higher at 65.00 years, with a mean age of 65.22 years, while Survived patients exhibit a lower median age of 60.00 years and a mean age of 58.76 years. These figures suggest that Dead patients tend to be older on average, potentially indicating age as a factor worth exploring in the context of patient outcomes.
- Creatinine Phosphokinase (CPK) Insights – The data reveals varying trends in creatinine phosphokinase levels. For the full sample, the median is 250.00, while the mean is substantially higher at 581.80. Dead patients exhibit a median of 259.00 and a mean of 670.20, while Survived patients have a median of 245.00 and a mean of 540.10. This indicates that creatinine phosphokinase levels tend to be higher on average in Dead patients, warranting further investigation into the potential significance of this marker in predicting patient outcomes.
- Ejection Fraction (EF) Findings – Ejection fraction statistics show differences

between the groups. The full sample has a median ejection fraction of 38.00 and a mean of 38.08, while Dead patients present a lower median of 30.00 and a mean of 33.47. In contrast, Survived patients have a similar median of 38.00 but a higher mean of 40.27. These variations highlight the potential role of ejection fraction in distinguishing outcomes, with Dead patients generally showing lower ejection fractions.

- Platelets Examination – The analysis of platelet counts indicates modest differences between the groups. The full sample exhibits a median of 262.00 and a mean of 263.36. Dead patients have a slightly lower median of 258.50 and mean of 256.38, while Survived patients have a slightly higher median of 263.00 and mean of 266.66. This suggests that platelet counts do not significantly differentiate between the two groups.
- Serum Creatinine (SC) Considerations – Serum creatinine values are notable, with the full sample showing a median of 1.10 and a mean of 1.39. Dead patients have a higher median of 1.30 and mean of 1.84, indicating elevated serum creatinine levels, while Survived patients have a lower median of 1.00 and mean of 1.19. These findings suggest that serum creatinine could serve as a valuable predictor of patient outcomes, with higher levels potentially associated with increased mortality risk.
- Serum Sodium (SS) Implication – The analysis of serum sodium levels demonstrates modest variation. The full sample has a median of 137.00 and a mean of 136.60. Dead patients exhibit a slightly lower median of 135.50 and mean of 135.40, while Survived patients present a higher median of 137.00 and mean of 137.20. These findings suggest that serum sodium levels may not be a major distinguishing factor in predicting patient outcomes.
- Time and Its Impact – Time, a variable with significant variance, exhibits interesting patterns. The median time for the full sample is 115.00, but Dead patients have a substantially lower median of 44.50, indicating a shorter time period, while Survived patients have a higher median of 172.00. The mean time for Dead patients is 70.89, while for Survived patients, it is 158.30. These results suggest that the time factor plays a critical role in distinguishing outcomes, with a shorter time interval being associated with higher mortality.

In summary, this analysis of numeric features provides valuable insights into the potential predictors of patient outcomes, highlighting the significance of age, creatinine phosphokinase, ejection fraction, serum creatinine, and time in distinguishing between Dead and Survived patients. These findings may serve as a basis for further research and clinical decision-making in healthcare contexts.

Table 10. Statistical quantitative description of the numeric features.

	Full sample			Dead patients			Survived patients		
Feature	Mdn	M	σ	Mdn	M	σ	Mdn	M	σ
Age	60	60.83	11.89	65	65.22	13.21	60	58.76	10.64
Creatinine phosphokinase	250	581.8	970.3	259	670.2	1316.58	245	540.1	753.8
Ejection fraction	38	38.08	11.83	30	33.47	12.53	38	40.27	10.86
Platelets	262	263.36	97.80	258.5	256.38	98.53	263	266.7	97.53
Serum creatinine	1.10	1.39	1.03	1.30	1.84	1.47	1	1.19	0.65
Serum sodium	137	136.6	4.41	135.5	135.4	5.00	137	137.2	3.98
Time	115	130.3	77.61	44.50	70.89	62.38	172	158.3	67.74

Mdn: Median. M: Mean. σ: Standard deviation Full sample: 299 individuals. Dead patients: 96 individuals. Survived patients: 203 individuals.

Following the presentation of statistical Tables 9 and Table 10, that elucidate the quantitative aspects of our dataset, we embark on a visual exploration of the data through graphical plots (Figure 12, 13, 14). Visualization serves as a powerful complement to numerical summaries, providing a more intuitive understanding of trends, patterns, and relationships within the data. This sequential integration of statistical tables and graphical plots aims to offer a comprehensive perspective on the dataset, enhancing the interpretability and depth of our analysis.

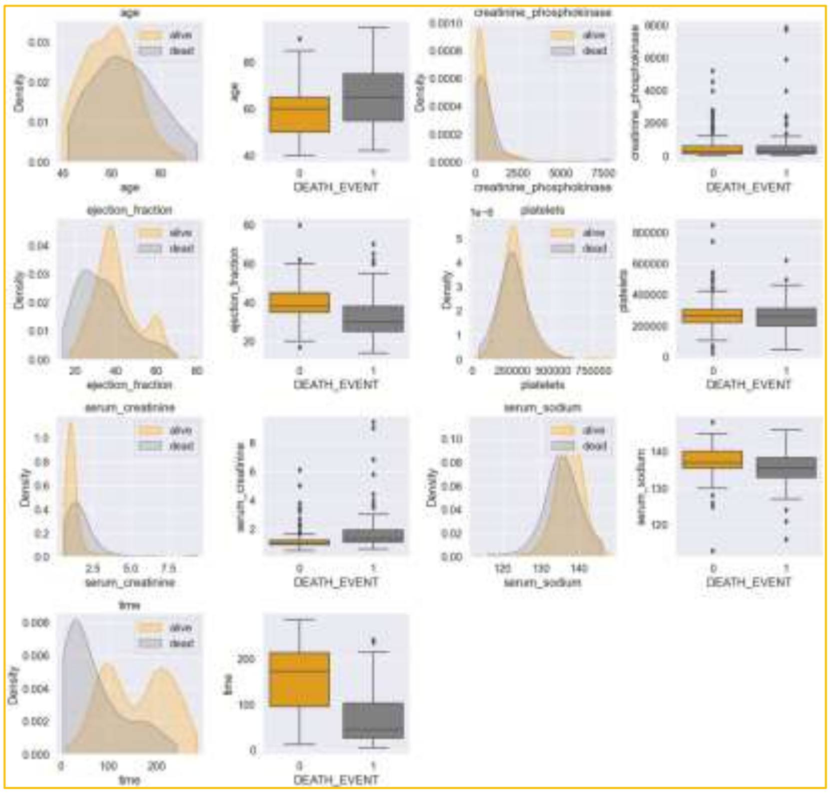


Figure 12. Distributions and Box-plots of numerical features.

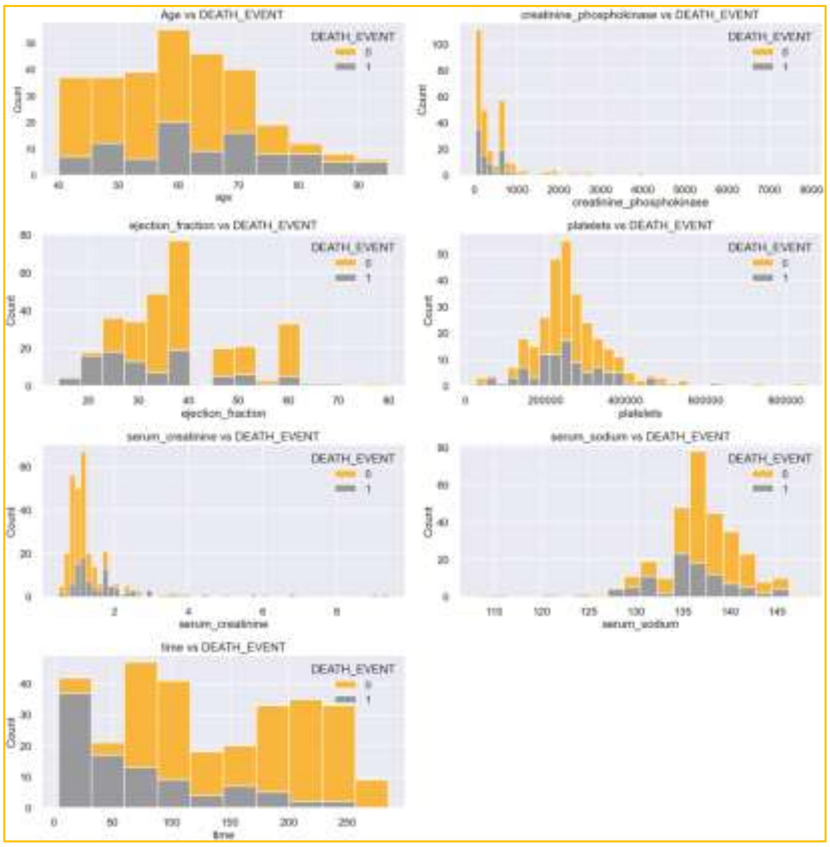


Figure 13. Histograms of numerical features.

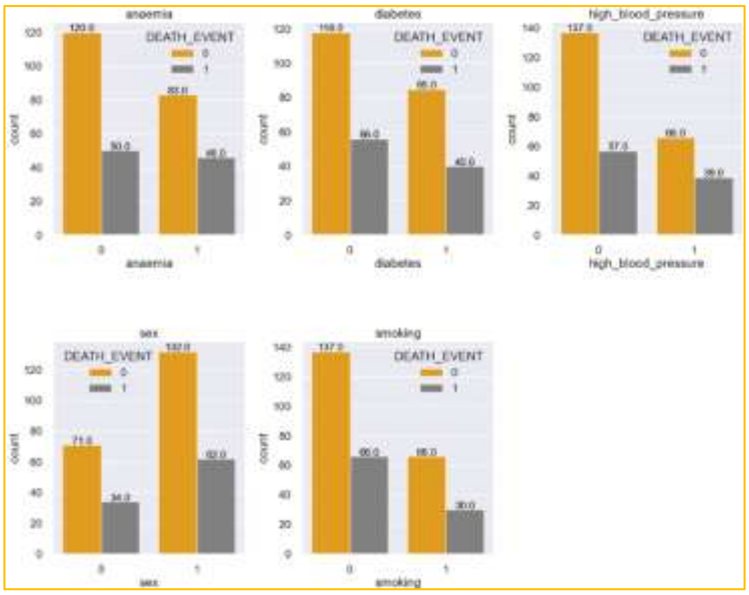


Figure 14. Histograms of categorical features.

3.2 Methodology

In our research on predicting death events in heart failure patients through Machine Learning, the dataset was thoroughly examined for missing values, and it was found to be complete, with no missing entries. An outlier analysis was also conducted, but no modifications were made due to the medical nature of the data, which was determined to have medically plausible values.

Data normalization was applied to ensure consistent feature scaling, a vital step to enhance model performance. Feature selection was carried out using the Extra Tree Classifier in several experimental approaches, helping to identify and retain the most relevant features, thus improving model efficiency and interpretability (see Figure 15). The feature "time", which is the most important feature, has been excluded from all the analyses to highlight the differentiation among the approaches of this study.

For binary classification of patient outcomes, a selection of Machine Learning models was employed, including Random Forest, Decision Tree, Support Vector Machine (SVM), Logistic Regression, k-Nearest Neighbors (KNN), and Gradient Boosting. Hyperparameters for each model were optimized through grid search. The dataset is separated as follows: 70% for training our Machine Learning models and 30% for validation.

A variety of experimental approaches were undertaken to assess model performance under different conditions. These approaches included using:

1. the full dataset,
2. with feature selection to retain the 'Serum creatinine' and 'Ejection fraction' features,
3. employing undersampling techniques for data balance,
4. combining undersampling with feature selection,
5. applying SMOTE for oversampling using the entire dataset,
6. combining SMOTE with feature selection,
7. and Principal Component Analysis (PCA) was employed to reduce dimensionality.

The dataset used for this research consists of clinical data related to heart failure patients, with the primary objective of classifying and predicting death events. Model performance was evaluated using various metrics, such as the Confusion matrix, AUC-ROC curve, Recall, Precision, F1-score, and Accuracy.

This comprehensive methodology provides a solid foundation for our research, enabling a clear presentation of findings and robust conclusions regarding the effectiveness of different Machine Learning models and approaches in predicting death events in heart failure patients.

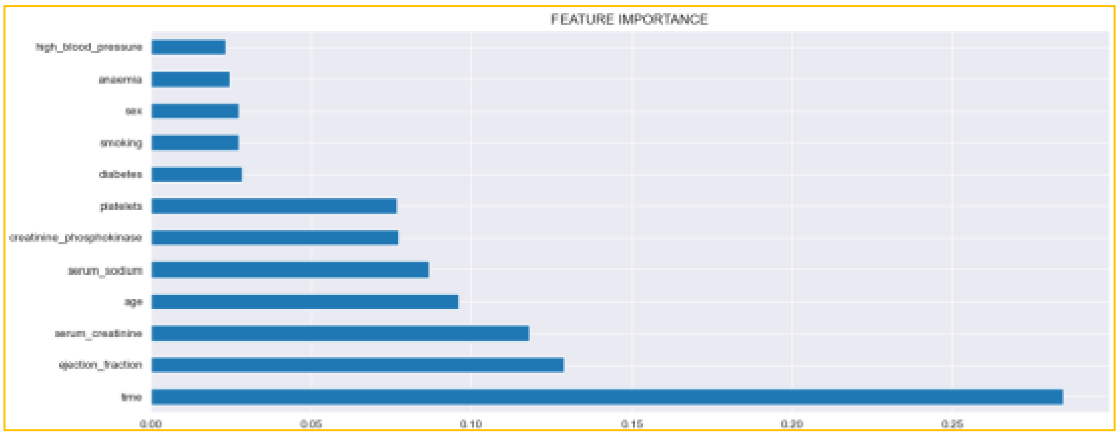


Figure 15. Feature importance using Extra Tree Classifier.

Chapter 4. RESULTS

Chapter 4 presents the results of this thesis, while the discussion of the findings also takes place. More specifically, for each machine learning model, the parameters obtained from the Grid search method are analyzed, and the results are presented through a Confusion matrix, Receiver operating characteristic - Area under the curve and other statistical validation measures.

4.1 Full dataset

In this section all the data features – except the time feature – are used to create the machine learning models.

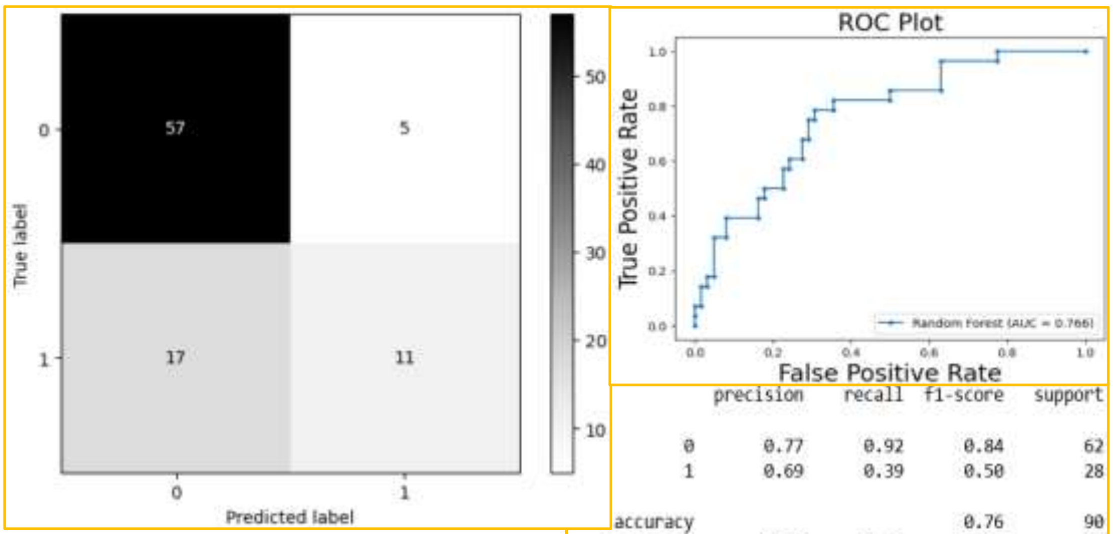
4.1.1 Random forest

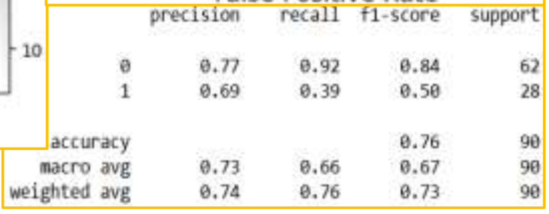
The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to False, 'criterion' utilizing entropy, 'max_depth' limited to 5 levels, 'min_samples_leaf' requiring a minimum of 2 samples per leaf node, 'min_samples_split' set at 5, and an ensemble of estimators consisting of 200 trees.

This confusion matrix provides a clear breakdown of the model's predictions. The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 16). In this case:

- True Positive (TP):
  11 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):
  57 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):
  5 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):
  17 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.766 (Figure 16).





```
              precision    recall  f1-score   support

           0       0.77      0.92      0.84        62
           1       0.69      0.39      0.50        28

    accuracy                           0.76        90
   macro avg       0.73      0.66      0.67        90
weighted avg       0.74      0.76      0.73        90
```

Figure 16. Random forest's confusion matrix, ROC Curve and statistics for the full dataset.

4.1.2 Decision tree

The optimized Decision tree model, obtained through grid search, is characterized by the following key parameters: a Gini criterion for node splitting, a maximum tree depth of 3 levels, a minimum of 7 samples required for leaf nodes, and a minimum of 2 samples for node splitting.

The confusion matrix (Figure 17) for the Decision tree model provides a breakdown of the model's predictions:

- True Positive (TP):

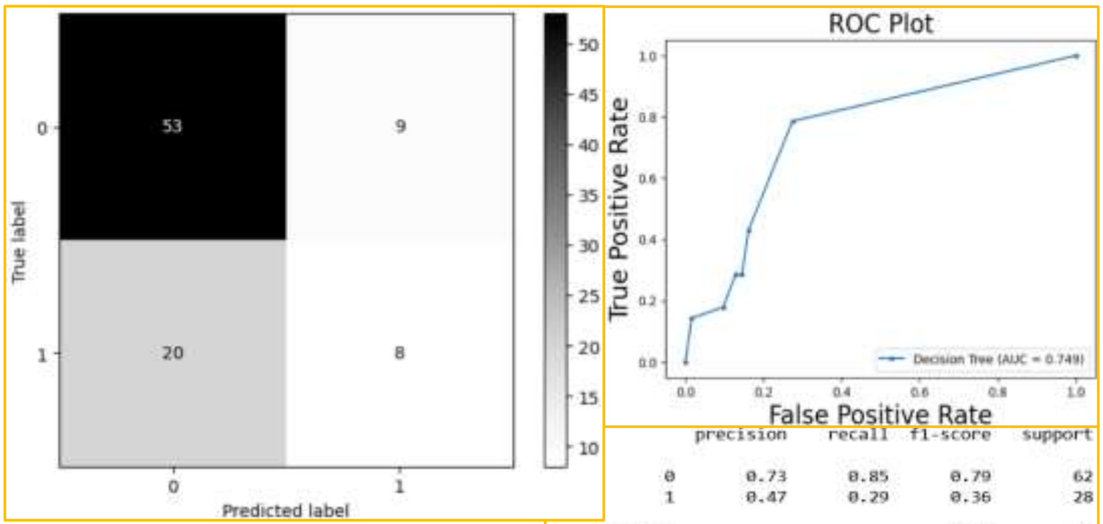
  8 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):

  53 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):

  9 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):

  20 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.749 (Figure 17). The Decision tree model that arises is presented in Figure 18.



```
              precision    recall  f1-score   support

           0       0.73      0.85      0.79        62
           1       0.47      0.29      0.36        28

    accuracy                           0.68        90
   macro avg       0.60      0.57      0.57        90
weighted avg       0.65      0.68      0.65        90
```

Figure 17. Decision tree's confusion matrix, ROC Curve and statistics for the full dataset.

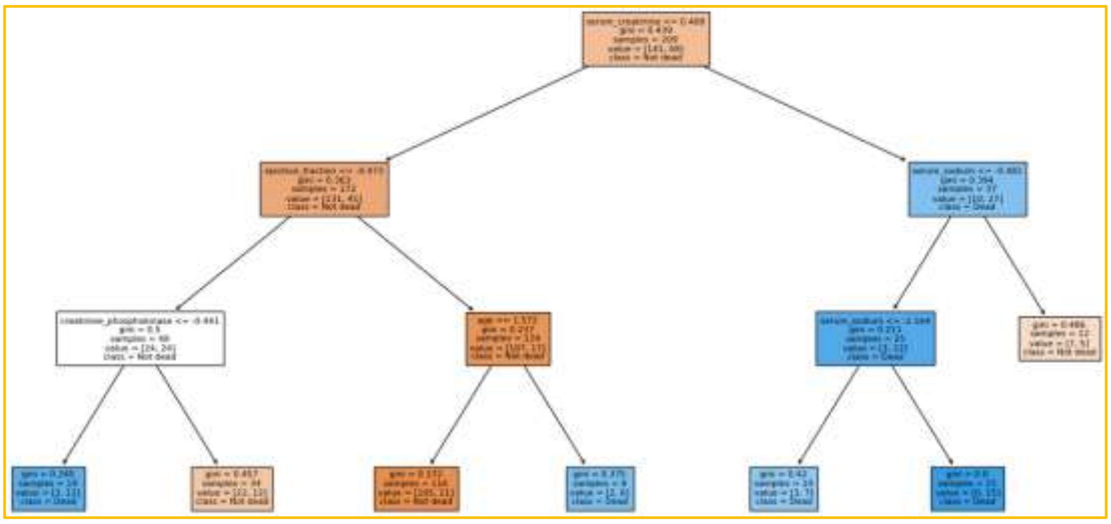


Figure 18. Decision tree for full dataset approach.

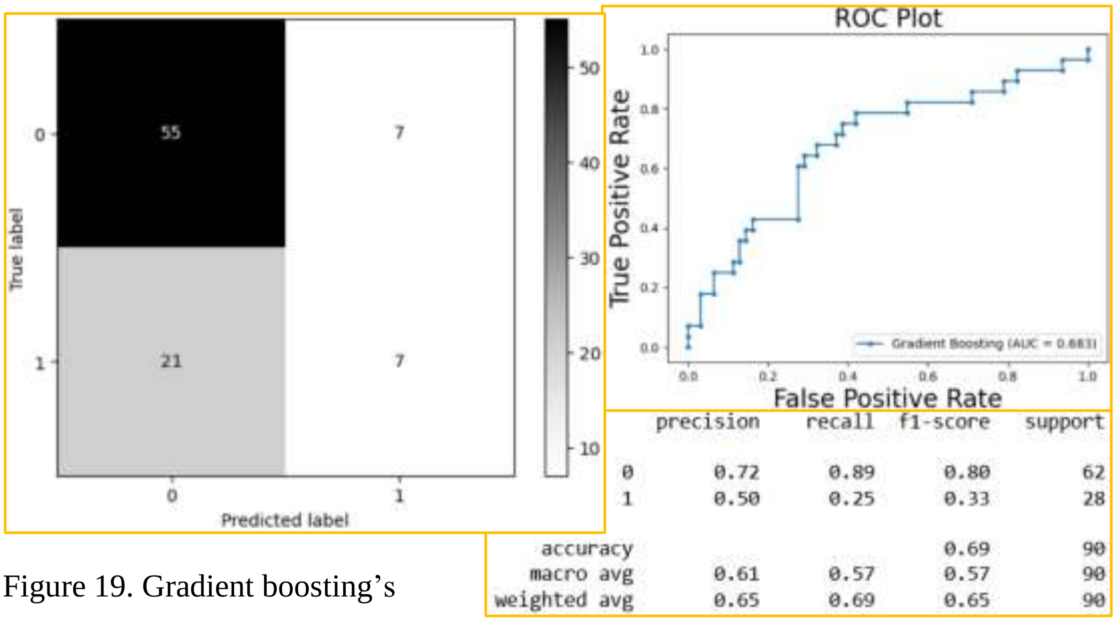
4.1.3 Gradient boosting

The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.1, a maximum tree depth of 10 levels, feature selection based on the logarithm base 2 of total features, 10 boosting stages (estimators), and a subsample fraction of 0.6.

The confusion matrix (Figure 19) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP): 7 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 55 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 21 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.683 (Figure 19).



```
              precision    recall  f1-score   support

           0       0.72      0.89      0.80        62
           1       0.50      0.25      0.33        28

    accuracy                           0.69        90
   macro avg       0.61      0.57      0.57        90
weighted avg       0.65      0.69      0.65        90
```

Figure 19. Gradient boosting's confusion matrix, ROC Curve and statistics for the full dataset.

4.1.4 Logistic regression

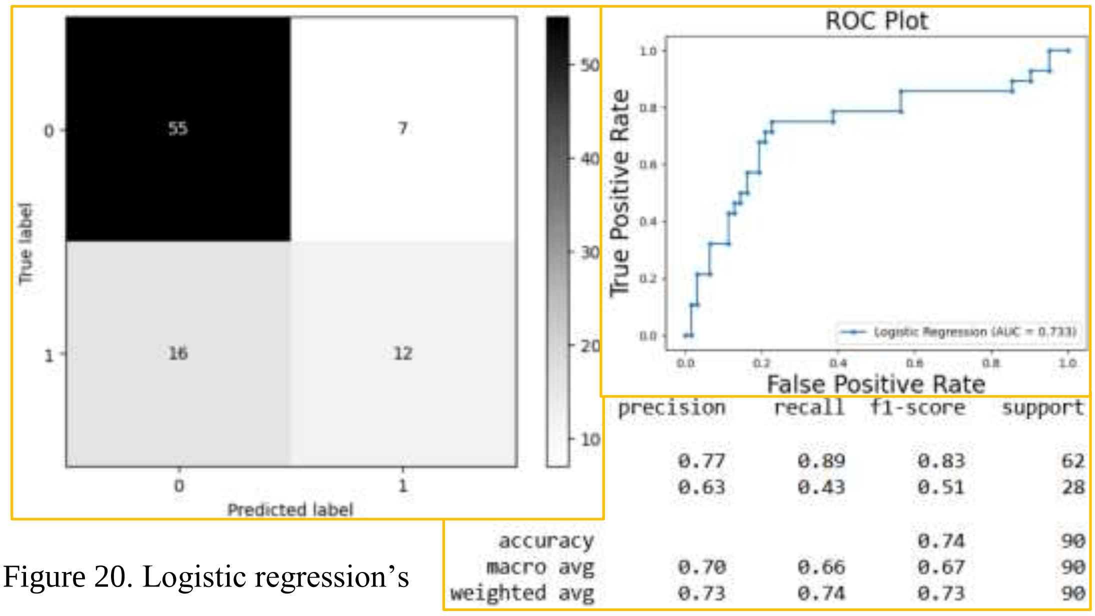
The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 10 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

The confusion matrix (Figure 20) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP): 12 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 55 instances of actual class 0 correctly predicted as class 0.

- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 16 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.733 (Figure 20).



```
              precision    recall  f1-score   support

                   0.77      0.89      0.83        62
                   0.63      0.43      0.51        28

    accuracy                           0.74        90
   macro avg       0.70      0.66      0.67        90
weighted avg       0.73      0.74      0.73        90
```

Figure 20. Logistic regression's confusion matrix, ROC curve and statistics for the full dataset.

4.1.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=3 (Figure 21), achieving a prediction accuracy score of 0.711.

The confusion matrix (Figure 22) for the K-Nearest Neighbors (KNN) model is analyzed:

- True Positive (TP): 10 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 54 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 8 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 18 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.681 (Figure 22).

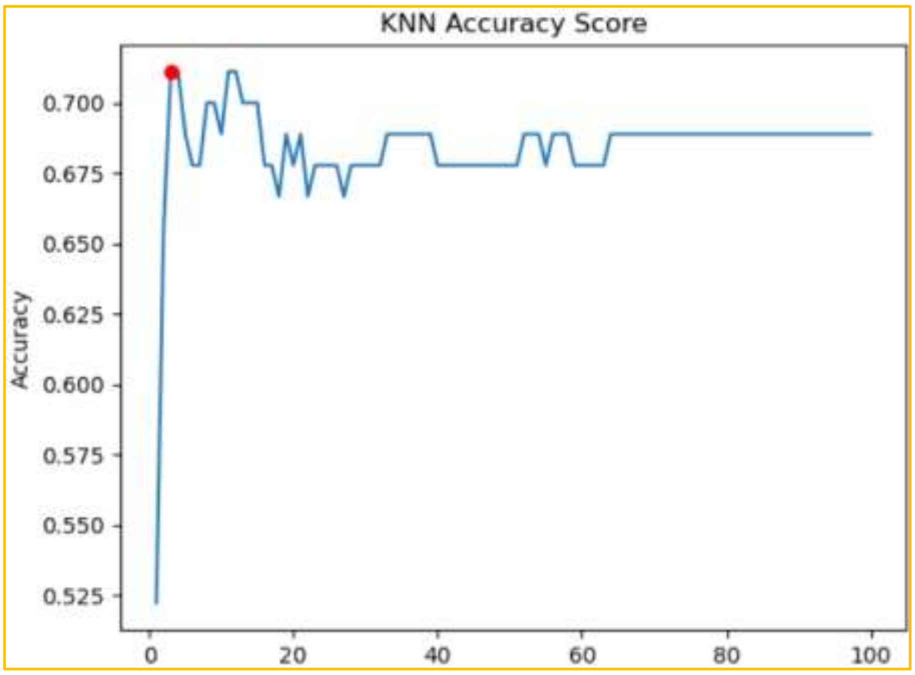
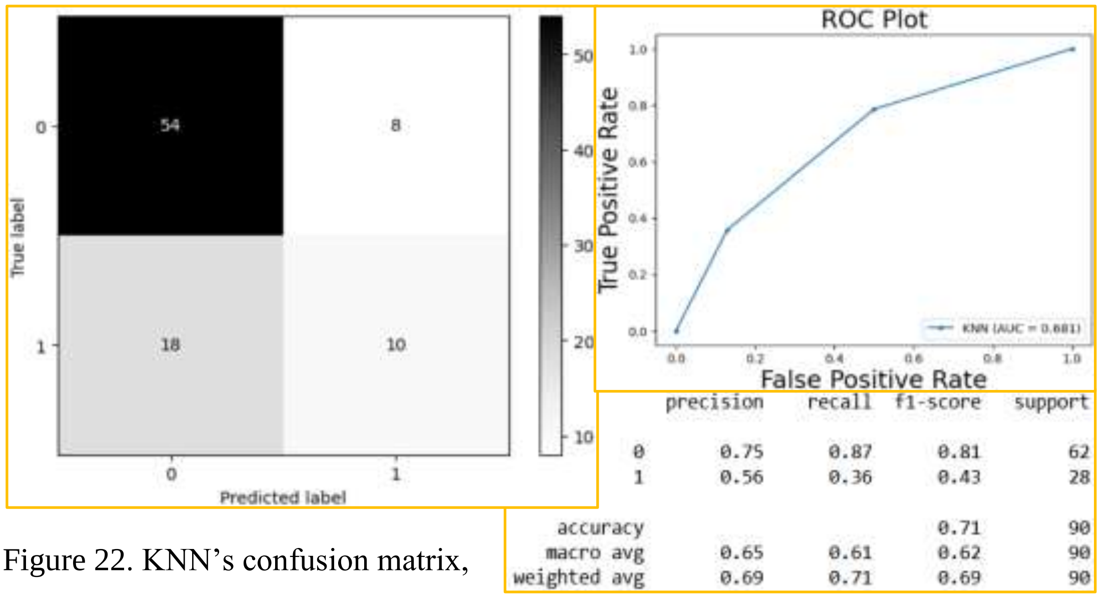


Figure 21. Ideal number of Nearest Neighbors (full dataset approach).



```
              precision    recall  f1-score   support

           0       0.75      0.87      0.81        62
           1       0.56      0.36      0.43        28

    accuracy                           0.71        90
   macro avg       0.65      0.61      0.62        90
weighted avg       0.69      0.71      0.69        90
```

Figure 22. KNN's confusion matrix, ROC curve and statistics for the full dataset.

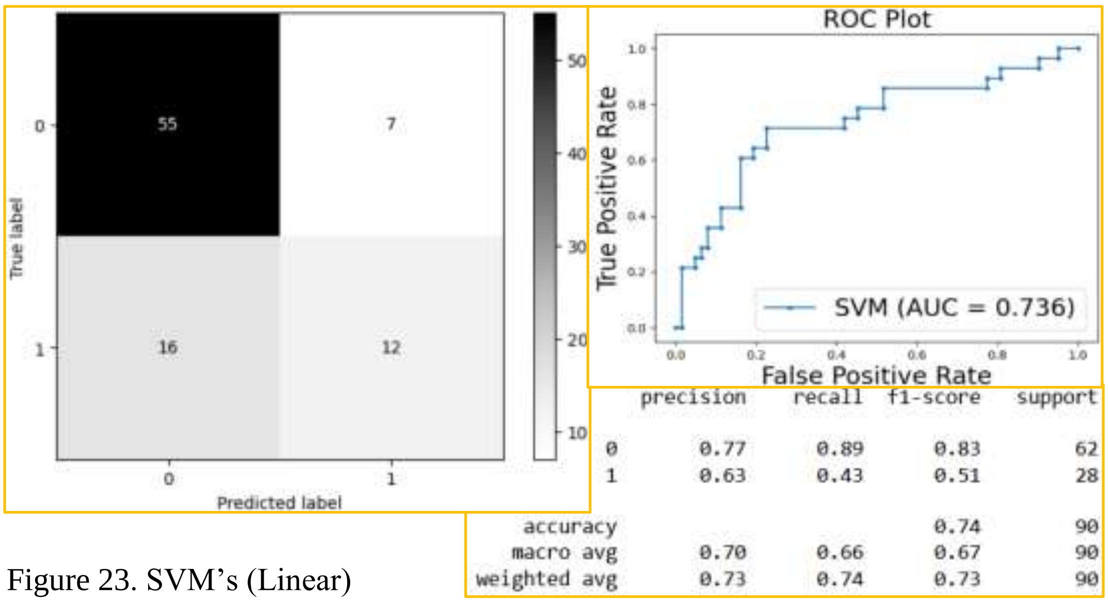
4.1.6 Support vector machines – Linear (SVM Linear)

The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 10, a degree of 1, and a gamma value of 0.0001.

The confusion matrix (Figure 23) for the Support Vector Machine (SVM) – Linear model is examined:

- True Positive (TP): 12 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 55 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 16 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.736 (Figure 23).



```
              precision    recall  f1-score   support

           0       0.77      0.89      0.83        62
           1       0.63      0.43      0.51        28

    accuracy                           0.74        90
   macro avg       0.70      0.66      0.67        90
weighted avg       0.73      0.74      0.73        90
```

Figure 23. SVM's (Linear) confusion matrix, ROC curve and statistics for the full dataset.

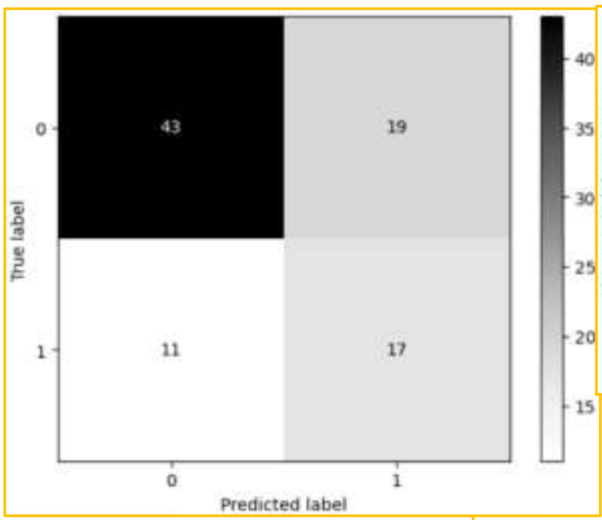
4.1.7 Support vector machines – Radial (SVM Radial)

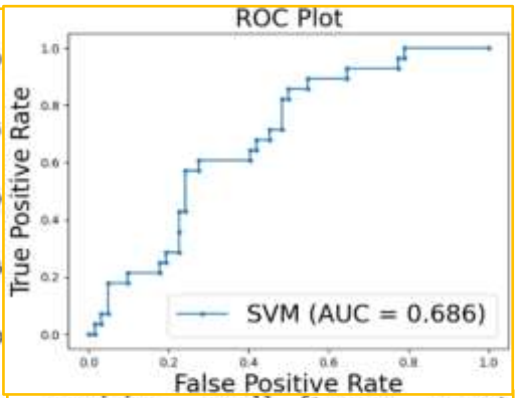
The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 24), obtained through grid search, is characterized by a regularization parameter 'C' of 10 and a gamma value of 0.1.

The confusion matrix for the Support Vector Machine (SVM) – Radial model with a radial kernel is tested:

- True Positive (TP): 17 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 43 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 19 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 11 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.686 (Figure 24).





```
              precision    recall  f1-score   support

           0       0.80      0.69      0.74        62
           1       0.47      0.61      0.53        28

    accuracy                           0.67        90
   macro avg       0.63      0.65      0.64        90
weighted avg       0.70      0.67      0.68        90
```

Figure 24. SVM's (Radial) confusion matrix, ROC curve and statistics for the full dataset.

4.2 Feature selection

In this section, the Extra Tree algorithm was employed to identify the two most significant features. The time feature was excluded from the analysis. As illustrated in Figure 25, serum creatinine and ejection fraction emerged as the two most important features, and they were consequently utilized in the construction of the machine learning models.

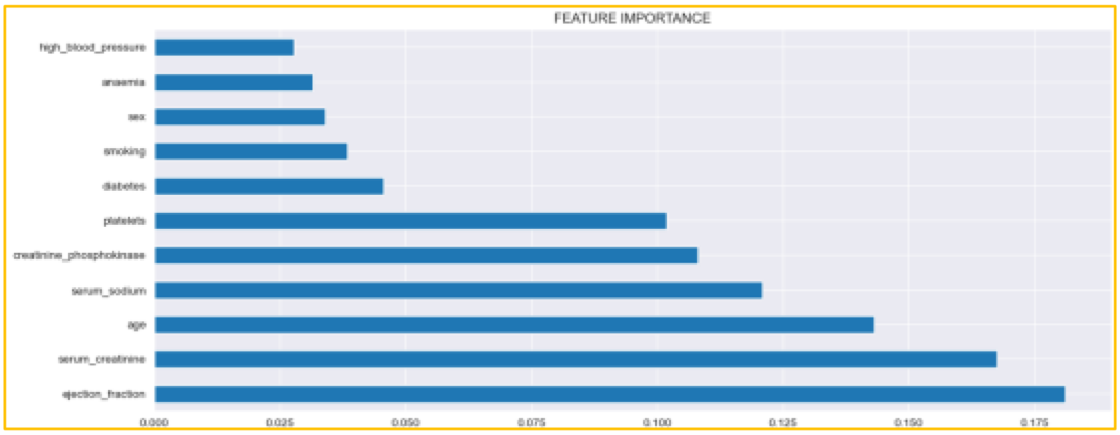


Figure 25. Feature selection for "Feature selection" approach.

4.2.1 Random forest

The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to True, 'criterion' utilizing gini, 'max_depth' limited to 5 levels, 'min_samples_leaf' requiring a minimum of 2 samples per leaf node, 'min_samples_split' set at 2, and an ensemble of estimators consisting of 200 trees.

This confusion matrix provides a clear breakdown of the model's predictions. The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 26). In this case:

- True Positive (TP): 13 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 55 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 15 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.799 (Figure 26).

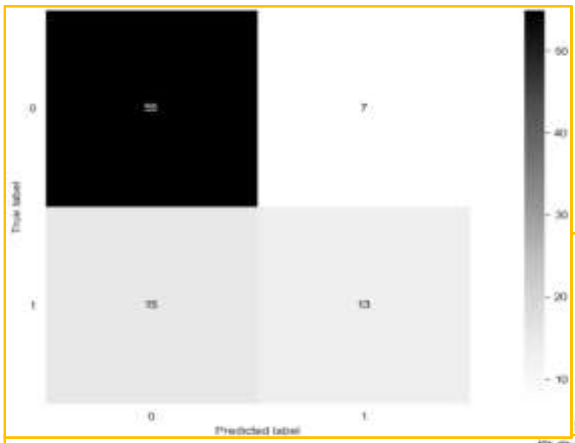
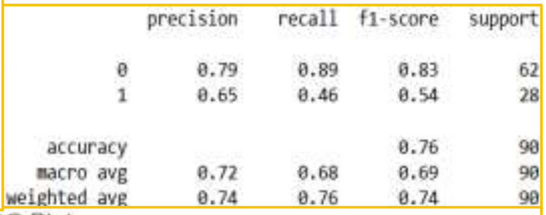


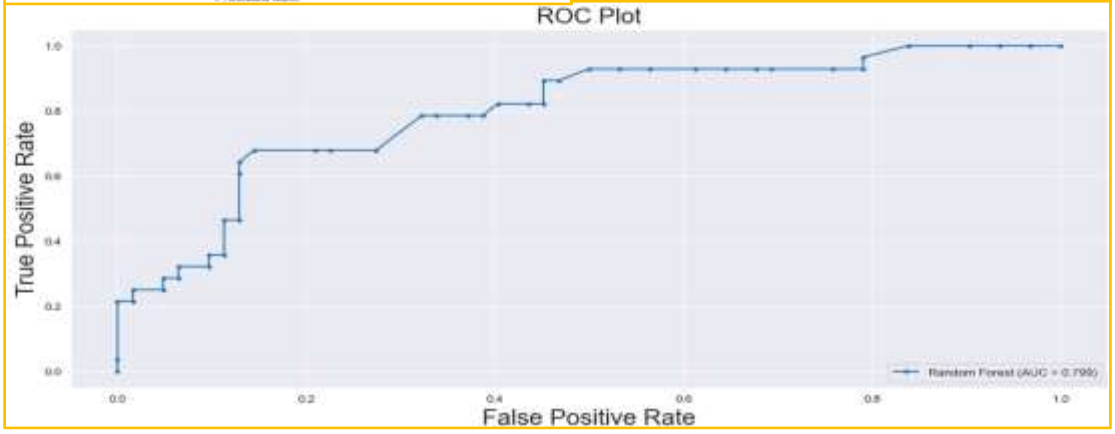
Figure 26. Random forest's confusion matrix, ROC Curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.79      0.89      0.83        62
           1       0.65      0.46      0.54        28

    accuracy                           0.76        90
   macro avg       0.72      0.68      0.69        90
weighted avg       0.74      0.76      0.74        90
```



4.2.2 Decision tree

The optimized Decision Tree model, obtained through grid search, is characterized by the following key parameters: a Entropy criterion for node splitting, a maximum tree depth of 7 levels, a minimum of 5 samples required for leaf nodes, and a minimum of 2 samples for node splitting.

The confusion matrix (Figure 27) for the Decision Tree model provides a breakdown of the model's predictions:

- True Positive (TP): 10 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 58 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 4 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 18 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.746 (Figure 27).

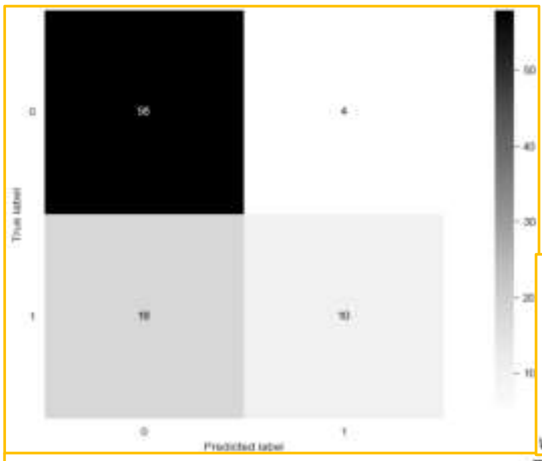
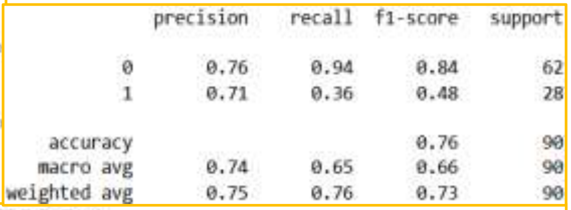


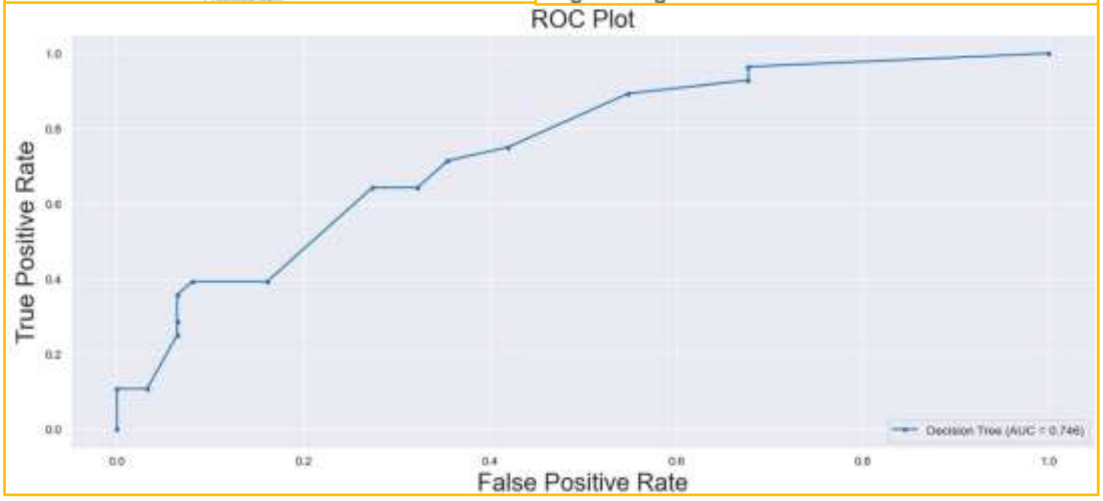
Figure 27. Decision tree’s confusion matrix, ROC Curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.76      0.94      0.84        62
           1       0.71      0.36      0.48        28

    accuracy                           0.76        90
   macro avg       0.74      0.65      0.66        90
weighted avg       0.75      0.76      0.73        90
```



4.2.3 Gradient boosting

The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.1, a maximum tree depth of 6 levels, feature selection based on the logarithm base 2 of total features, 10 boosting stages (estimators), and a subsample fraction of 0.8.

The confusion matrix (Figure 28) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP): 11 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 56 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 17 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.741 (Figure 28).

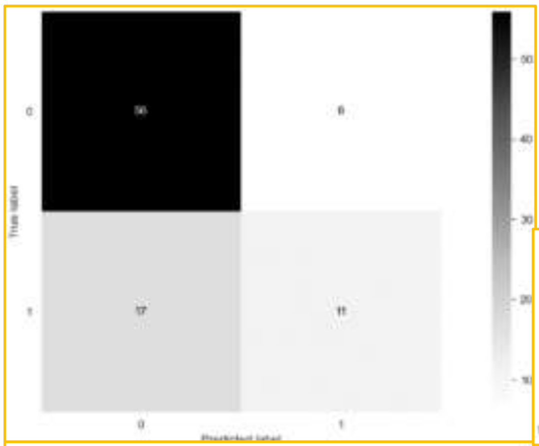
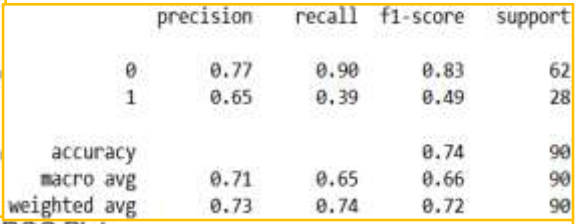


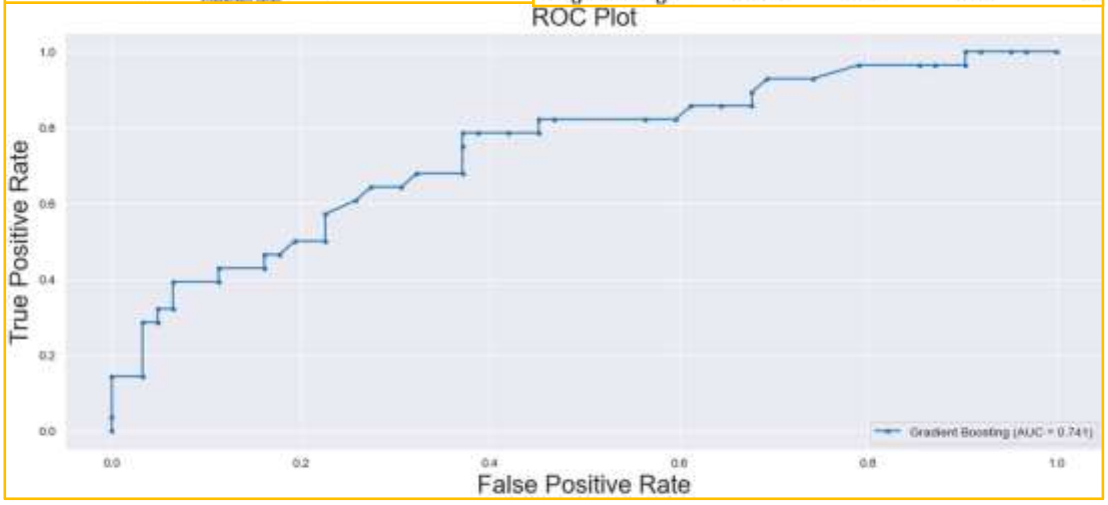
Figure 28. Gradient boosting's confusion matrix, ROC Curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.77      0.90      0.83        62
           1       0.65      0.39      0.49        28

    accuracy                           0.74        90
   macro avg       0.71      0.65      0.66        90
weighted avg       0.73      0.74      0.72        90
```



4.2.4 Logistic regression

The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 1 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

The confusion matrix (Figure 29) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP): 10 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 59 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 3 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 18 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.716 (Figure 29).

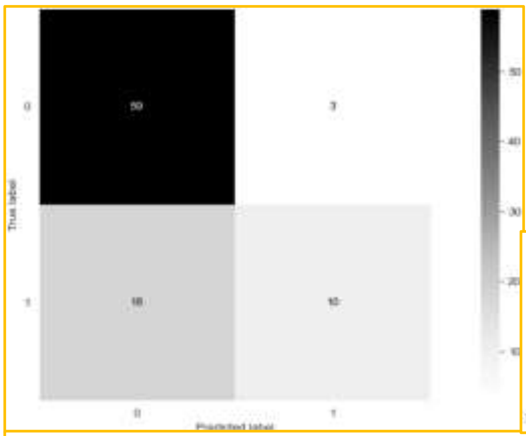
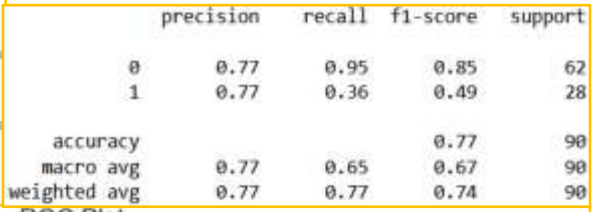


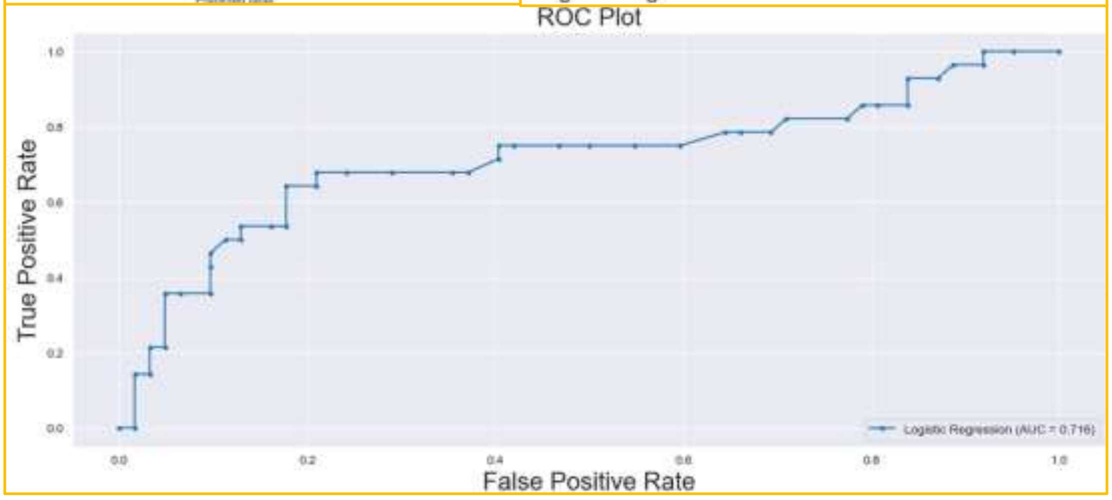
Figure 29. Logistic regression’s confusion matrix, ROC curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.77      0.95      0.85        62
           1       0.77      0.36      0.49        28

    accuracy                           0.77        90
   macro avg       0.77      0.65      0.67        90
weighted avg       0.77      0.77      0.74        90
```



4.2.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=18 (Figure 30), achieving a prediction accuracy score of 0.789. The confusion matrix (Figure 31) for the K-Nearest Neighbors (KNN) model is analyzed:

- True Positive (TP): 14 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 57 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 5 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 14 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.733 (Figure 31).

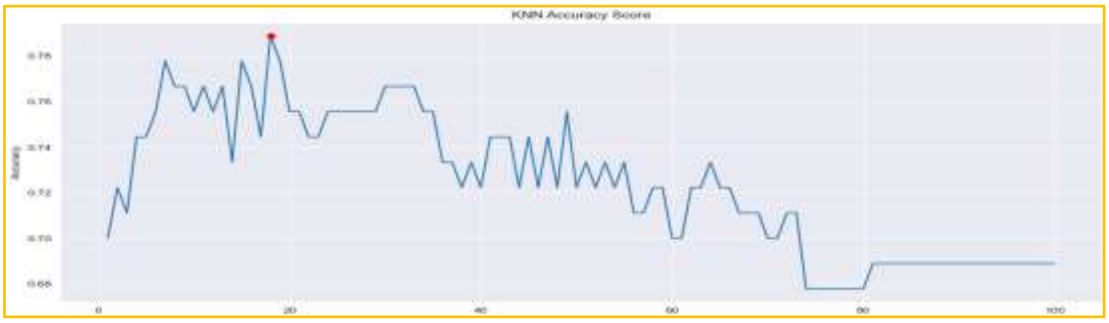


Figure 30. Ideal number of Nearest Neighbors (feature selection approach).

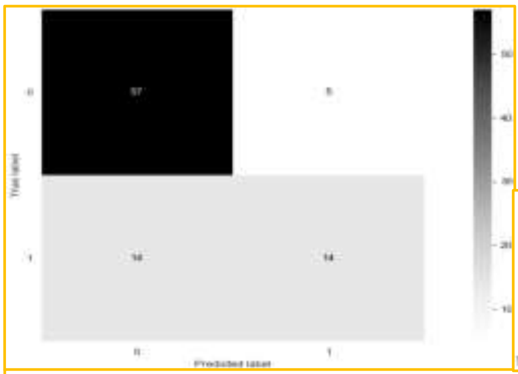
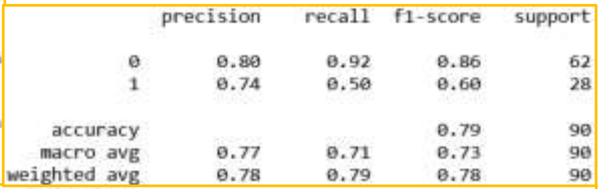


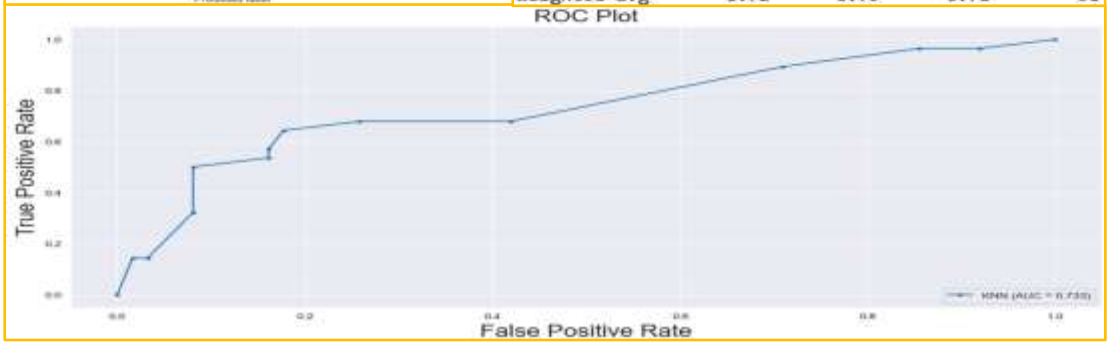
Figure 31. KNN’s confusion matrix, ROC curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.80      0.92      0.86        62
           1       0.74      0.50      0.60        28

    accuracy                           0.79        90
   macro avg       0.77      0.71      0.73        90
weighted avg       0.78      0.79      0.78        90
```



4.2.6 Support vector machines - Linear (SVM Linear)

The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 1, a degree of 1, and a gamma value of 0.0001.

The confusion matrix (Figure 32) for the Support vector machine linear model is examined:

- True Positive (TP): 10 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 59 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 3 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 18 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.718 (Figure 32).

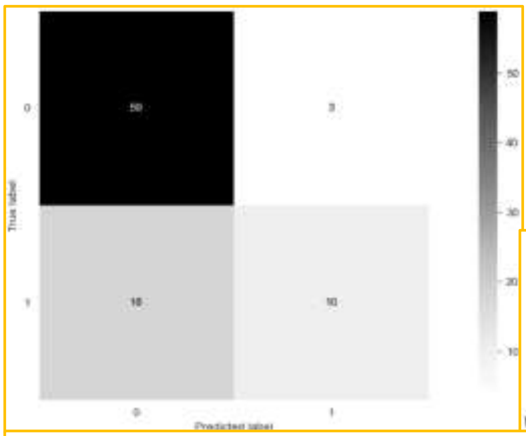
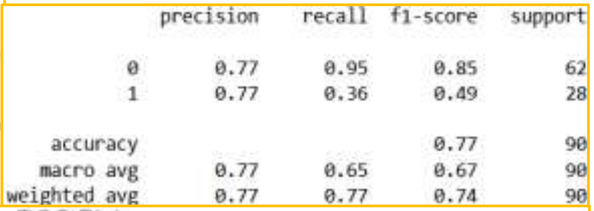


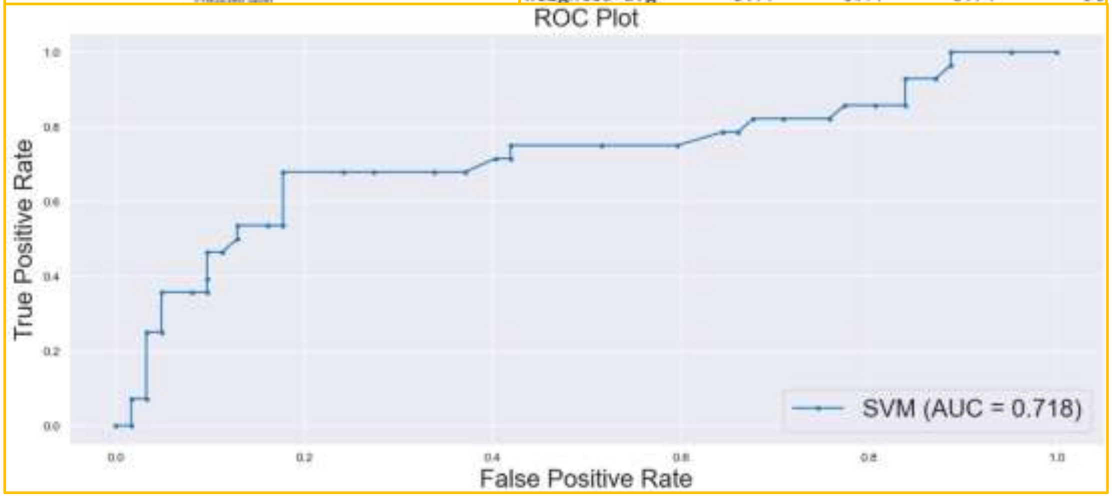
Figure 32. SVM's (Linear) confusion matrix, ROC curve and statistics for the feature selection approach.



```
              precision    recall  f1-score   support

           0       0.77      0.95      0.85        62
           1       0.77      0.36      0.49        28

    accuracy                           0.77        90
   macro avg       0.77      0.65      0.67        90
weighted avg       0.77      0.77      0.74        90
```



4.2.7 Support vector machines - Radial (SVM Radial)

The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 33), obtained through grid search, is characterized by a regularization parameter 'C' of 1 and a gamma value of 10.

The confusion matrix for the Support Vector Machine (SVM) model with a radial kernel is tested:

- True Positive (TP): 13 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 54 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 8 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 15 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.804 (Figure 33).

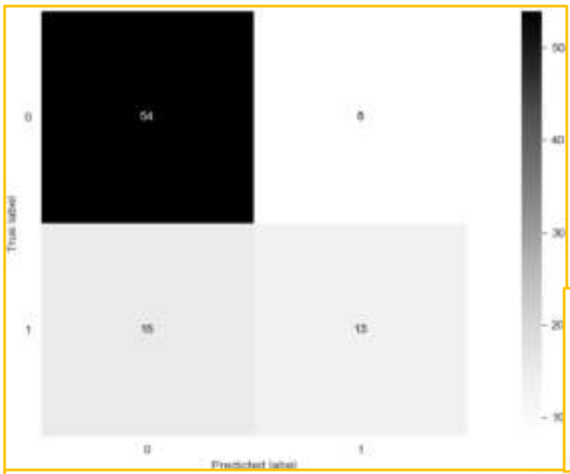
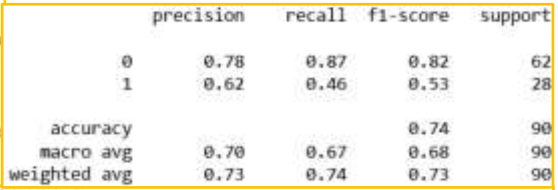


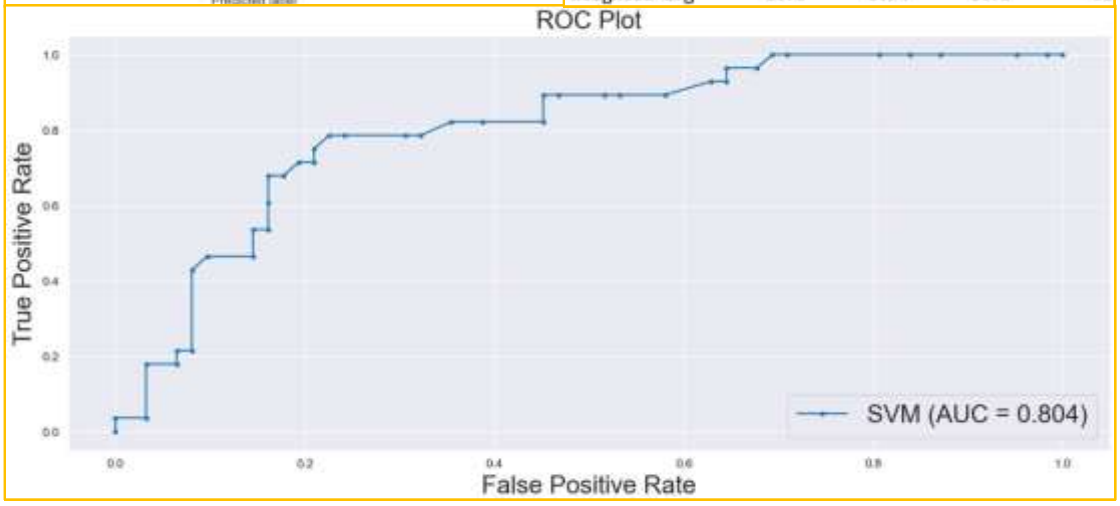
Figure 33. SVM's (Radial) confusion matrix, ROC curve and statistics for the full dataset.



```
              precision    recall  f1-score   support

           0       0.78      0.87      0.82        62
           1       0.62      0.46      0.53        28

    accuracy                           0.74        90
   macro avg       0.70      0.67      0.68        90
weighted avg       0.73      0.74      0.73        90
```



4.3 Undersampling

In response to the imbalanced nature of the dataset, characterized by 203 instances of class 0 (death event not occurred) and 96 instances of class 1 (death event occurred), an undersampling technique was implemented. This involved randomly removing instances from the majority class (class 0) to create a more equitable distribution between the classes. The primary goal of undersampling is to address bias and ensure that the machine learning model does not disproportionately favor the dominant class. The Figure 34, presents how the final classes are formulated after the undersampling method.

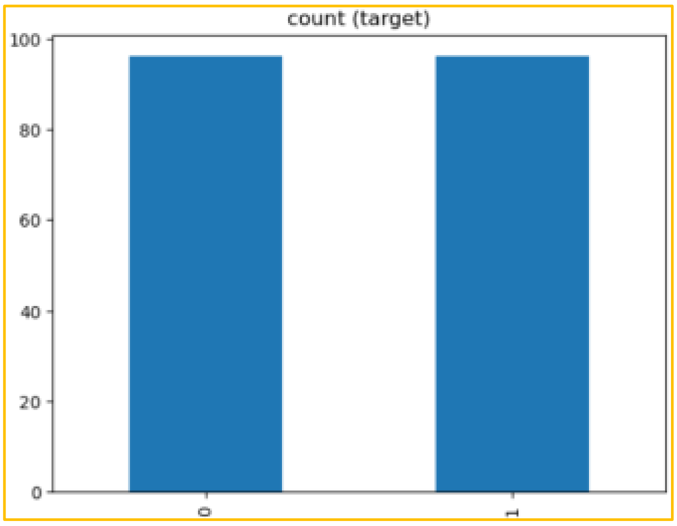


Figure 34. Final classes after the undersampling method.

4.3.1 Random forest

The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to True, 'criterion' utilizing gini, 'max_depth' limited to 6 levels, 'min_samples_leaf' requiring a minimum of 1 samples per leaf node, 'min_samples_split' set at 4, and an ensemble of estimators consisting of 100 trees. This confusion matrix provides a clear breakdown of the model's predictions.

The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 35). In this case:

- True Positive (TP): 23 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 19 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 9 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.833 (Figure 35).

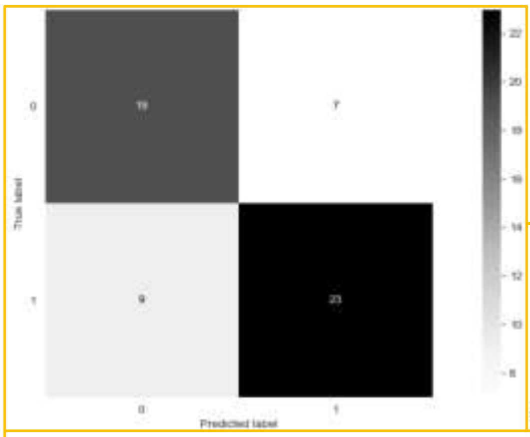
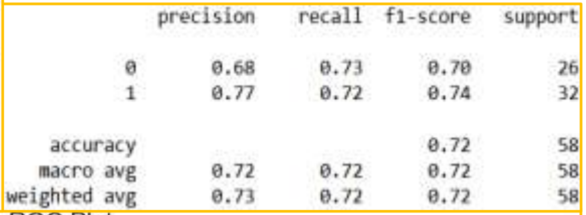


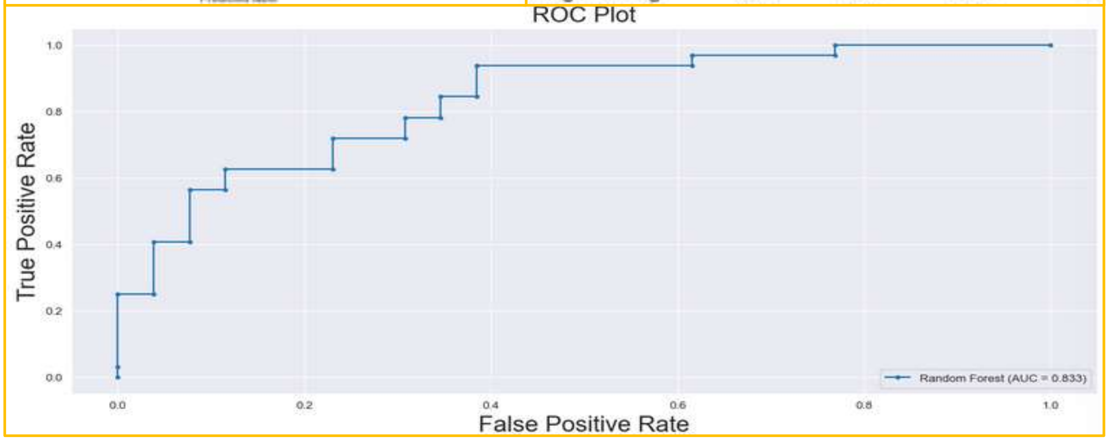
Figure 35. Random forest’s confusion matrix, ROC Curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.68      0.73      0.70        26
           1       0.77      0.72      0.74        32

    accuracy                           0.72        58
   macro avg       0.72      0.72      0.72        58
weighted avg       0.73      0.72      0.72        58
```



4.3.2 Decision tree

The optimized Decision Tree model, obtained through grid search, is characterized by the following key parameters: a Entropy criterion for node splitting, a maximum tree depth of 10 levels, a minimum of 1 samples required for leaf nodes, and a minimum of 9 samples for node splitting.

The confusion matrix (Figure 36) for the Decision Tree model provides a breakdown of the model's predictions:

- True Positive (TP): 22 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 15 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 11 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 10 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.706 (Figure 36).

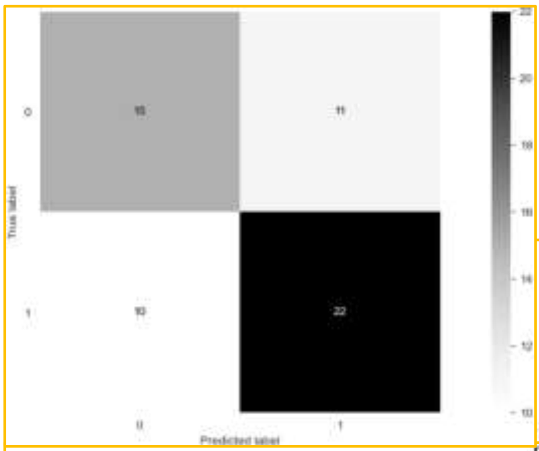
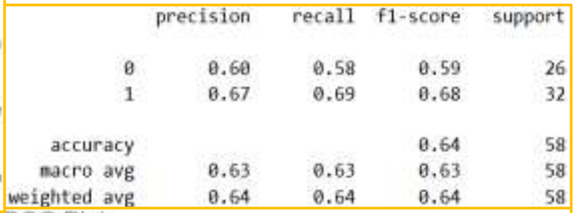


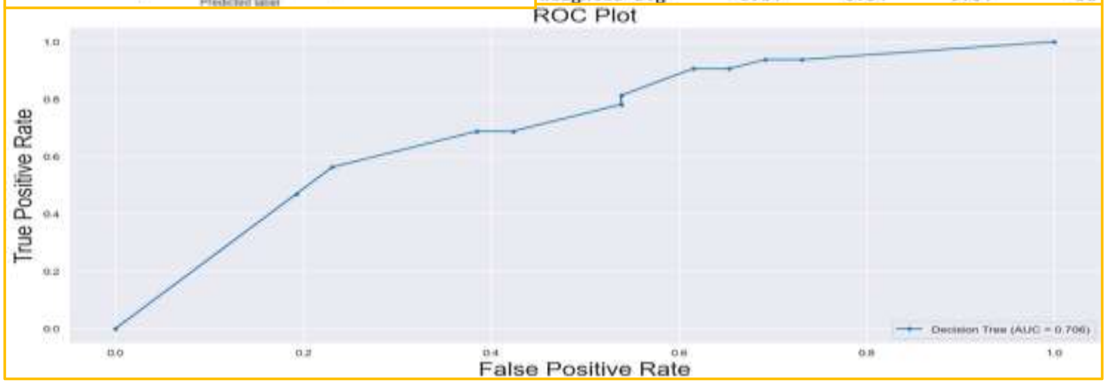
Figure 36. Decision tree's confusion matrix, ROC Curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.60      0.58      0.59        26
           1       0.67      0.69      0.68        32

    accuracy                           0.64        58
   macro avg       0.63      0.63      0.63        58
weighted avg       0.64      0.64      0.64        58
```



4.3.3 Gradient boosting

The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.05, a maximum tree depth of 7 levels, feature selection based on the logarithm base 2 of total features, 100 boosting stages (estimators), and a subsample fraction of 0.5.

The confusion matrix (Figure 37) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP): 19 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 19 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 13 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.774 (Figure 37).

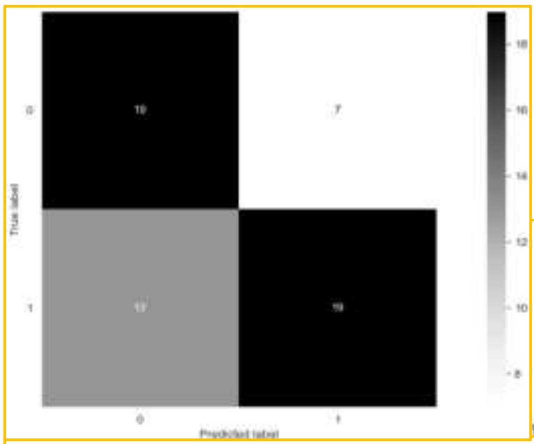
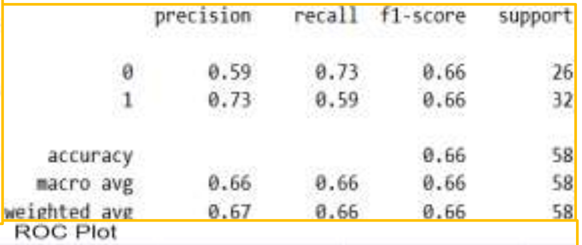


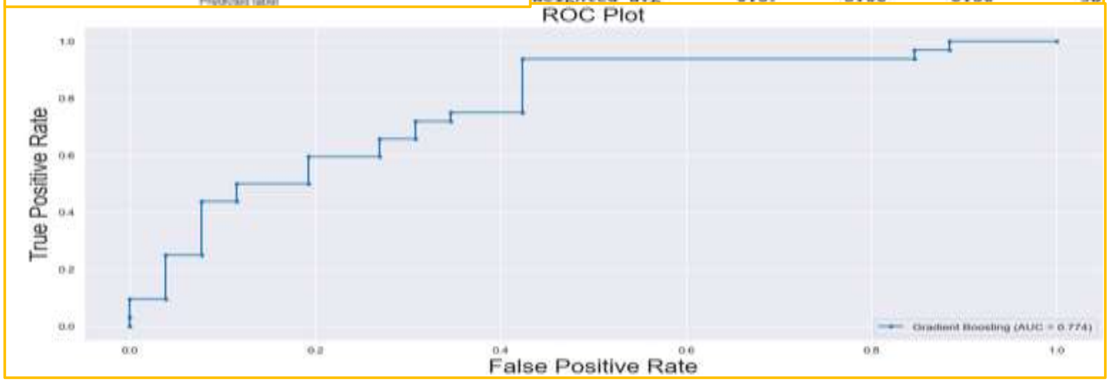
Figure 37. Gradient boosting’s confusion matrix, ROC Curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.59      0.73      0.66        26
           1       0.73      0.59      0.66        32

    accuracy                           0.66        58
   macro avg       0.66      0.66      0.66        58
weighted avg       0.67      0.66      0.66        58
```



4.3.4 Logistic regression

The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 0.01 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

The confusion matrix (Figure 38) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP): 15 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 21 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 5 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 17 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.722 (Figure 38).

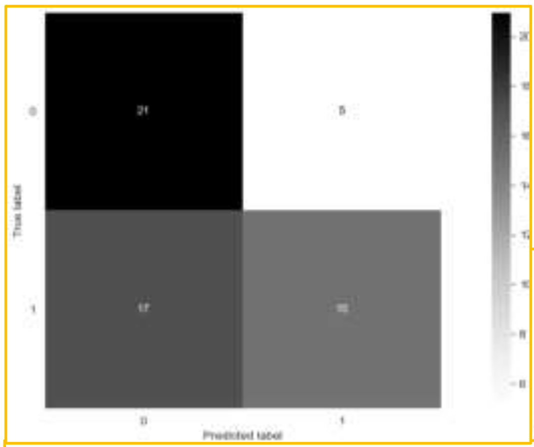
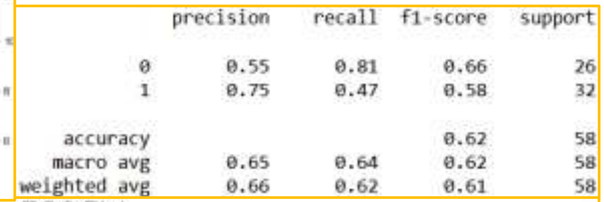


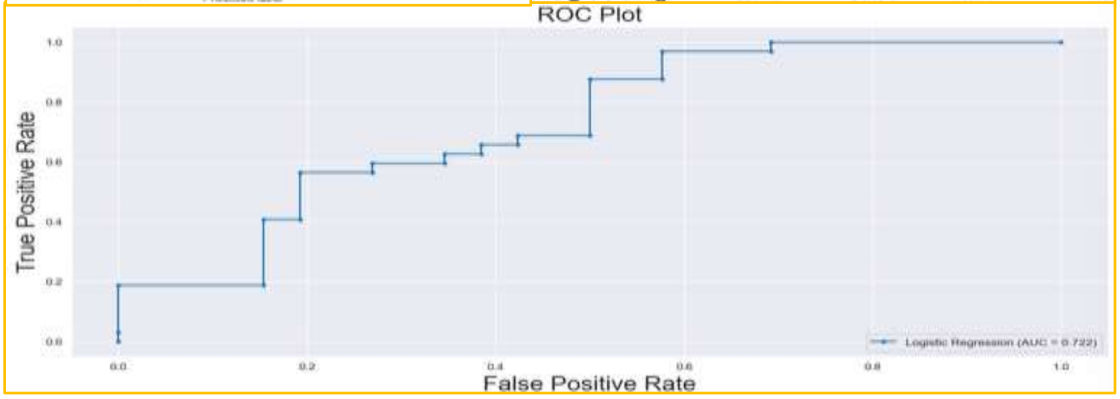
Figure 38. Logistic regression's confusion matrix, ROC curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.55      0.81      0.66        26
           1       0.75      0.47      0.58        32

    accuracy                           0.62        58
   macro avg       0.65      0.64      0.62        58
weighted avg       0.66      0.62      0.61        58
```



4.3.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=5 (Figure 39), achieving a prediction accuracy score of 0.672.

The confusion matrix (Figure 40) for the K-Nearest Neighbors (KNN) model is analyzed:

- True Positive (TP): 18 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 21 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 5 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 14 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.706 (Figure 40).

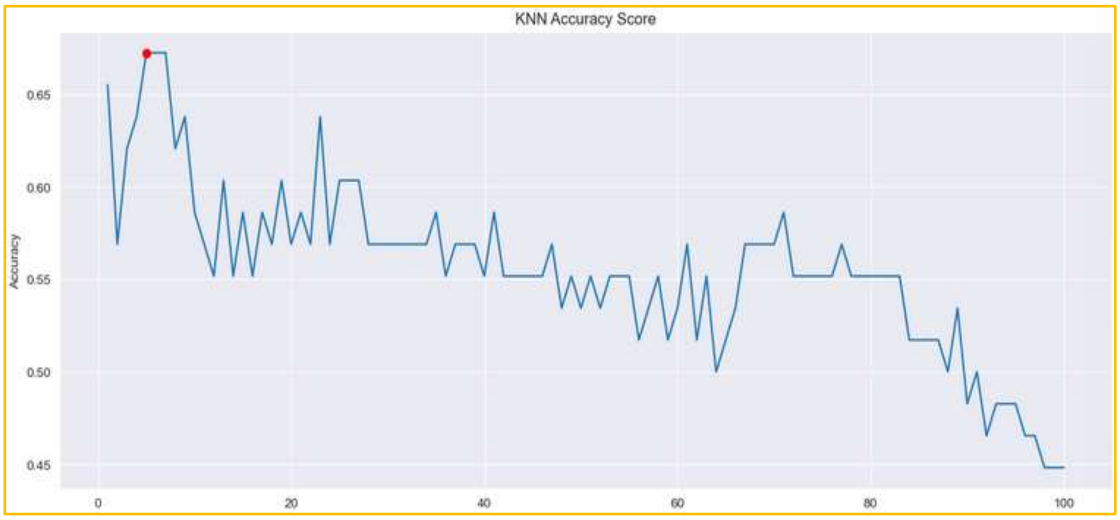


Figure 39. Ideal number of Nearest Neighbors (Undersampling approach).

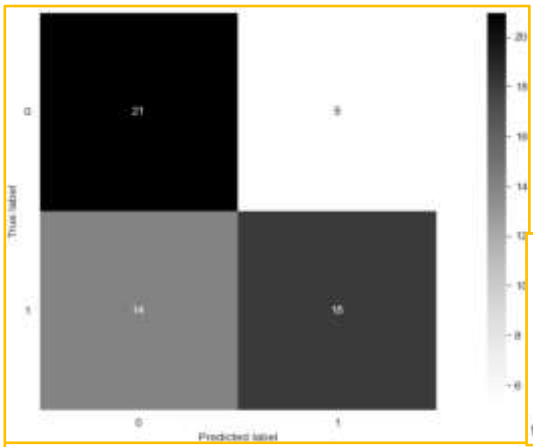
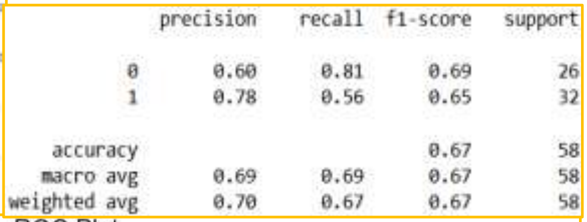


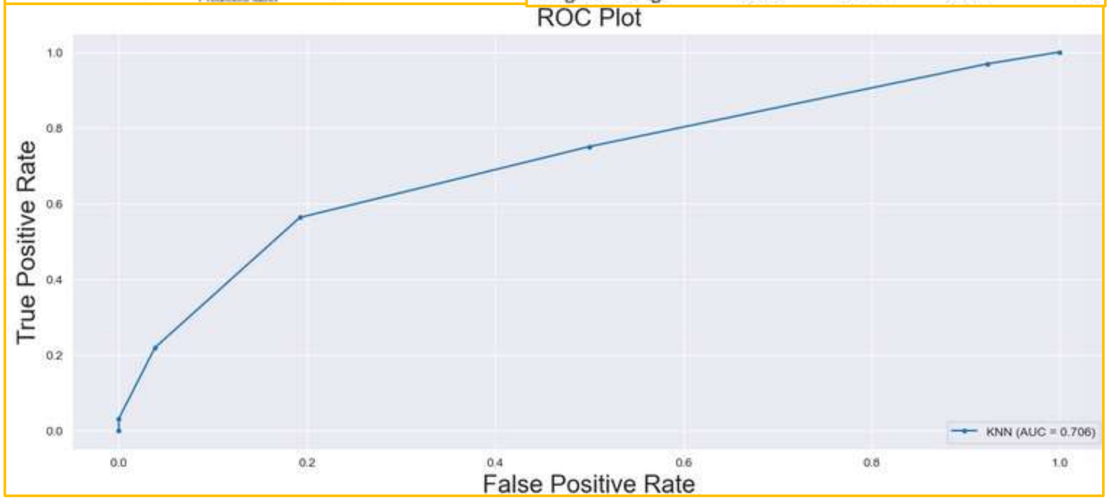
Figure 40. KNN’s confusion matrix, ROC curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.60      0.81      0.69        26
           1       0.78      0.56      0.65        32

    accuracy                           0.67        58
   macro avg       0.69      0.69      0.67        58
weighted avg       0.70      0.67      0.67        58
```



4.3.6 Support vector machines - Linear (SVM Linear)

The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 1, a degree of 1, and a gamma value of 0.0001.

The confusion matrix (Figure 41) for the Support vector machine linear model is examined:

- True Positive (TP): 20 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 20 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 12 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.793 (Figure 41).

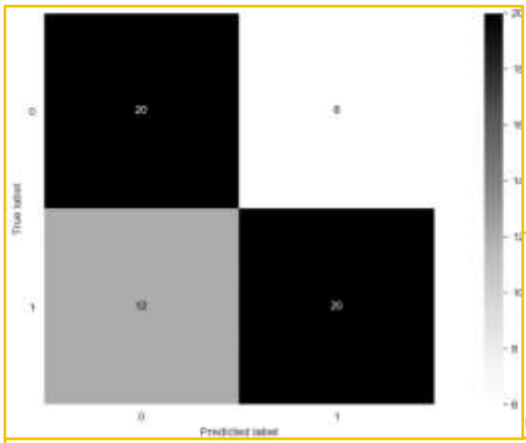
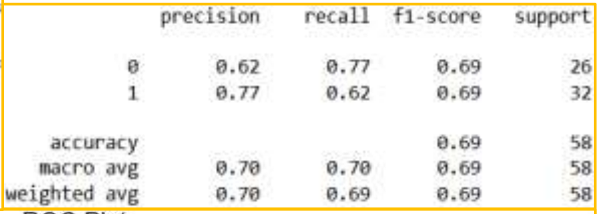


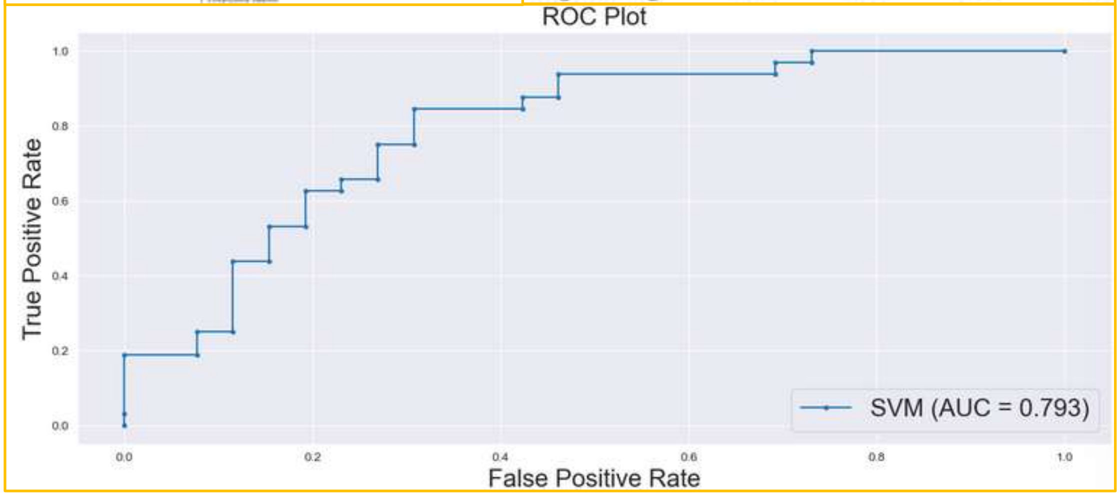
Figure 41. SVM's (Linear) confusion matrix, ROC curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.62      0.77      0.69        26
           1       0.77      0.62      0.69        32

    accuracy                           0.69        58
   macro avg       0.70      0.70      0.69        58
weighted avg       0.70      0.69      0.69        58
```



4.3.7 Support vector machines - Radial (SVM Radial)

The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 42), obtained through grid search, is characterized by a regularization parameter 'C' of 10 and a gamma value of 0.01.

The confusion matrix for the Support Vector Machine (SVM) model with a radial kernel is tested:

- True Positive (TP): 21 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 20 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 11 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.768 (Figure 42).

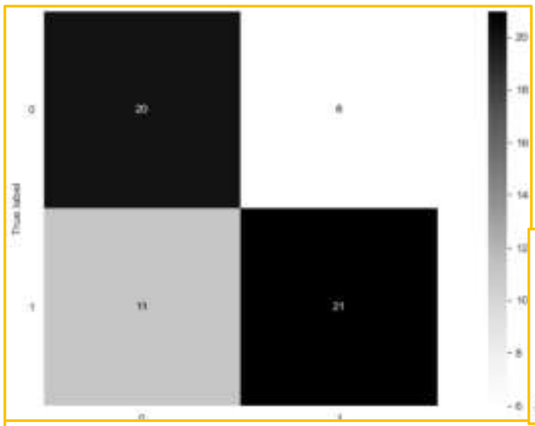
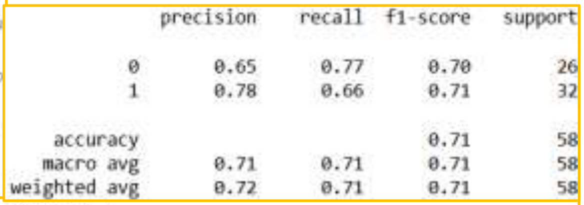


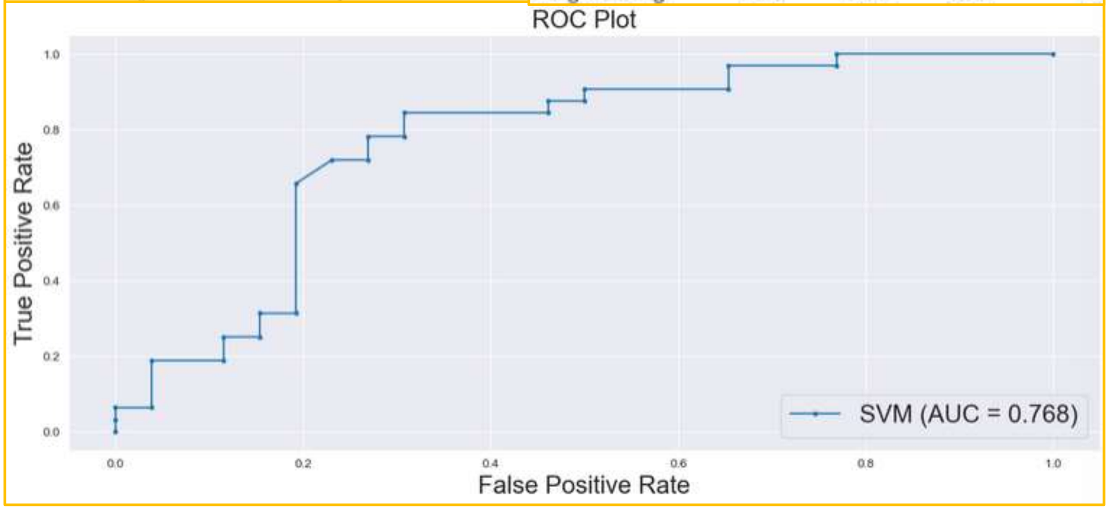
Figure 42. SVM's (Radial) confusion matrix, ROC curve and statistics for the undersampling approach.



```
              precision    recall  f1-score   support

           0       0.65      0.77      0.70        26
           1       0.78      0.66      0.71        32

    accuracy                           0.71        58
   macro avg       0.71      0.71      0.71        58
weighted avg       0.72      0.71      0.71        58
```



4.4 Undersampling and feature selection

In this section, a combination of undersampling and feature selection approaches is applied.

4.4.1 Random forest

The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to True, 'criterion' utilizing gini, 'max_depth' limited to 5 levels, 'min_samples_leaf' requiring a minimum of 2 samples per leaf node, 'min_samples_split' set at 5, and an ensemble of estimators consisting of 200 trees. This confusion matrix provides a clear breakdown of the model's predictions. The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 43). In this case:

- True Positive (TP): 24 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 20 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 8 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.819 (Figure 43).

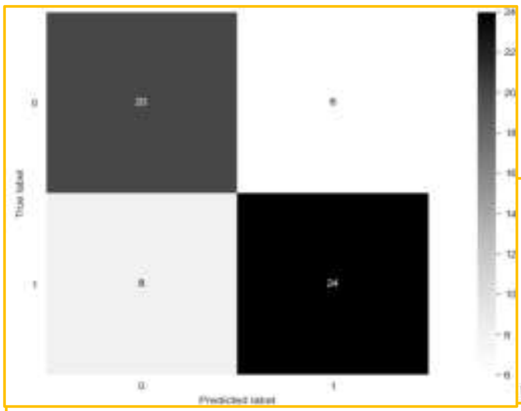
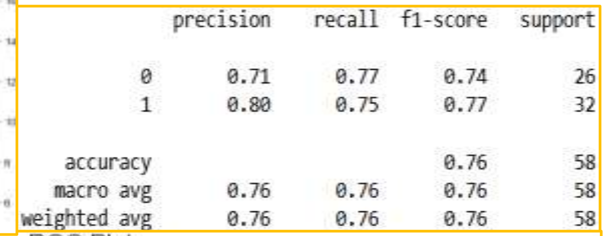


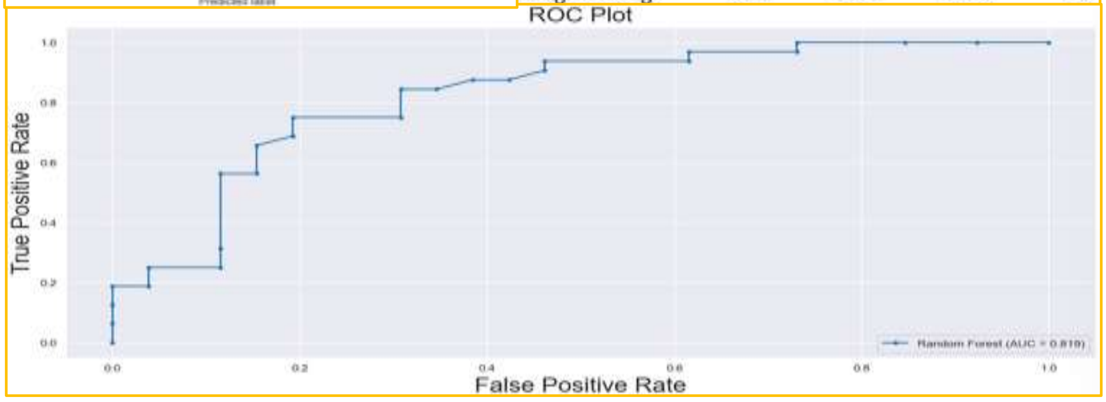
Figure 43. Random forest's confusion matrix, ROC Curve and statistics for Undersampling and feature selection



```
              precision    recall  f1-score   support

           0       0.71      0.77      0.74        26
           1       0.80      0.75      0.77        32

    accuracy                           0.76        58
   macro avg       0.76      0.76      0.76        58
weighted avg       0.76      0.76      0.76        58
```



approach.

4.4.2 Decision tree

The optimized Decision Tree model, obtained through grid search, is characterized by the following key parameters: a Gini criterion for node splitting, a maximum tree depth of 4 levels, a minimum of 5 samples required for leaf nodes, and a minimum of 2 samples for node splitting.

The confusion matrix (Figure 44) for the Decision Tree model provides a breakdown of the model's predictions:

- True Positive (TP): 19 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 22 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 4 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 13 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.745 (Figure 44).

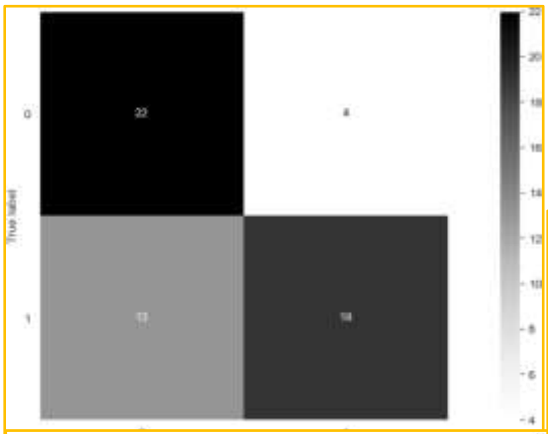
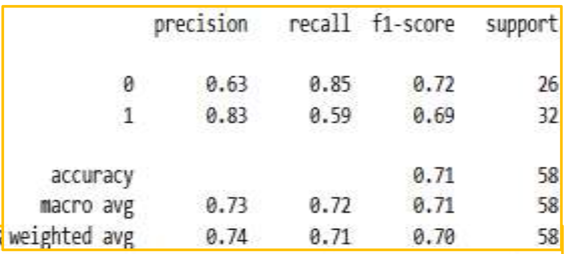


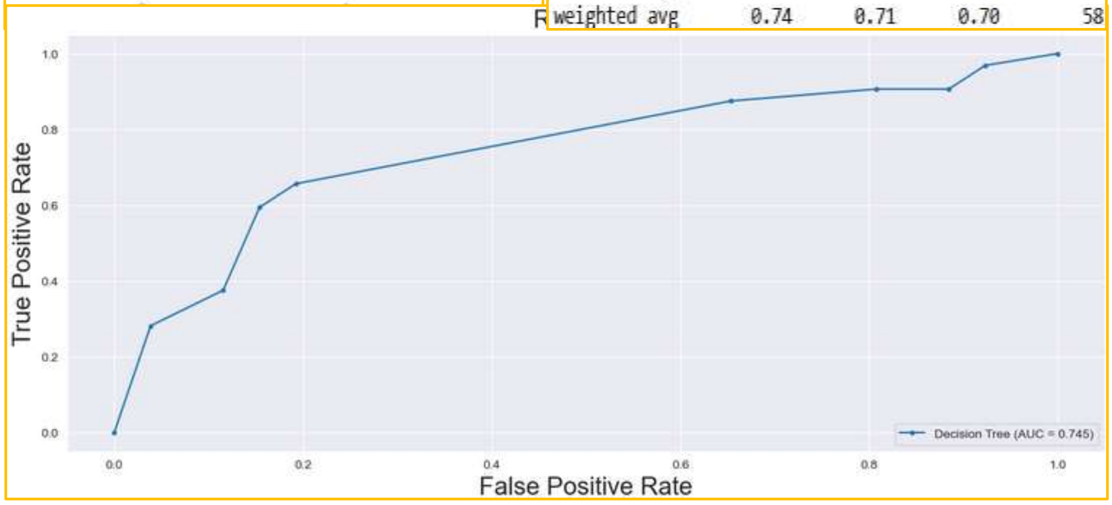
Figure 44. Decision tree’s confusion matrix, ROC Curve and statistics for Undersampling and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.63      0.85      0.72        26
           1       0.83      0.59      0.69        32

    accuracy                           0.71        58
   macro avg       0.73      0.72      0.71        58
weighted avg       0.74      0.71      0.70        58
```



4.4.3 Gradient boosting

The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.01, a maximum tree depth of 2 levels, feature selection based on the logarithm base 2 of total features, 100 boosting stages (estimators), and a subsample fraction of 0.5.

The confusion matrix (Figure 45) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP): 23 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 20 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 9 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.796 (Figure 45).

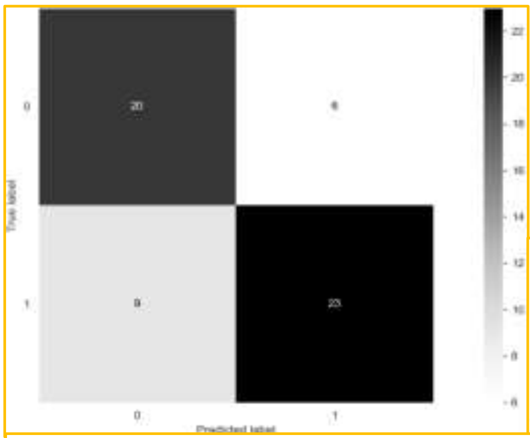
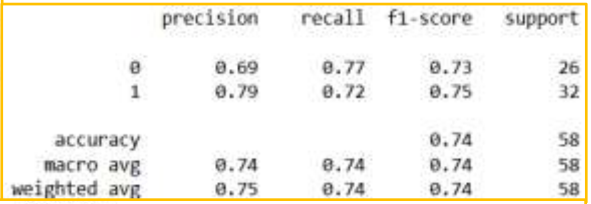


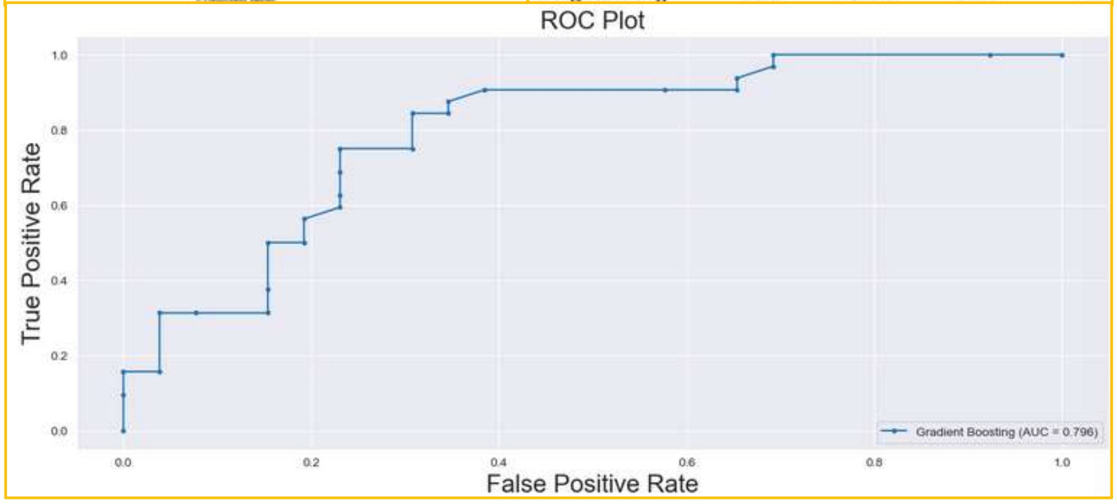
Figure 45. Gradient boosting's confusion matrix, ROC Curve and statistics for Undersampling and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.69      0.77      0.73        26
           1       0.79      0.72      0.75        32

    accuracy                           0.74        58
   macro avg       0.74      0.74      0.74        58
weighted avg       0.75      0.74      0.74        58
```



4.4.4 Logistic regression

The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 10 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

The confusion matrix (Figure 46) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP): 25 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 19 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 7 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.767 (Figure 46).

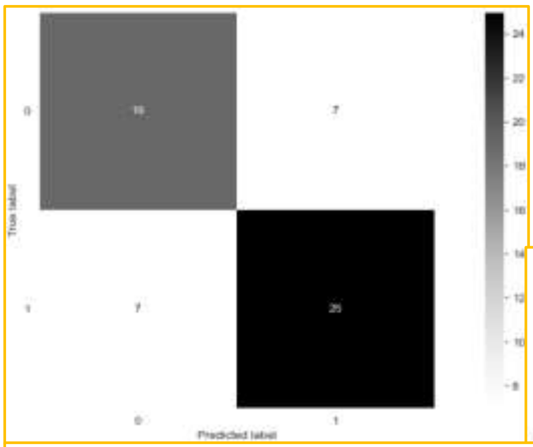
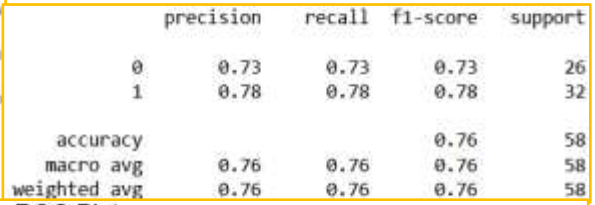


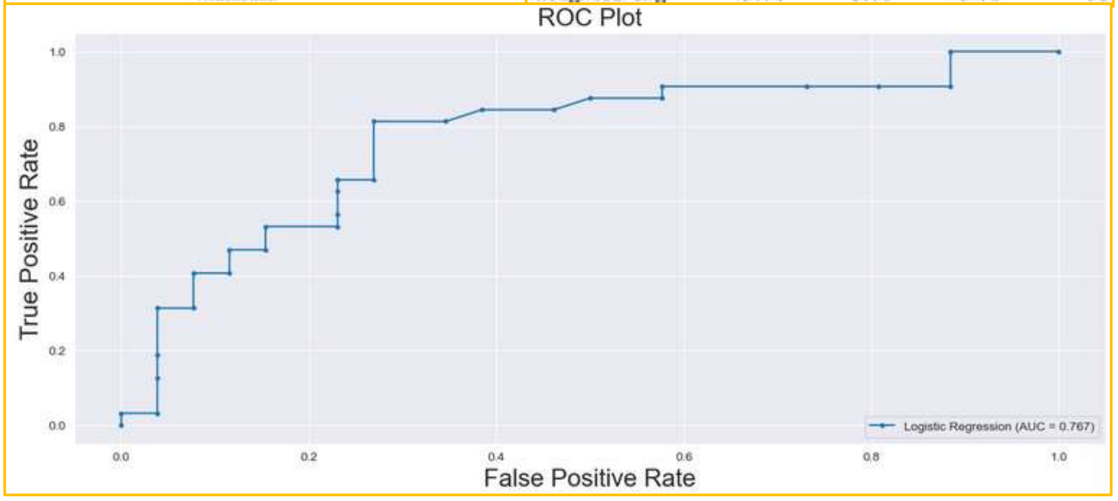
Figure 46. Logistic regression’s confusion matrix, ROC curve and statistics for Undersampling and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.73      0.73      0.73        26
           1       0.78      0.78      0.78        32

    accuracy                           0.76        58
   macro avg       0.76      0.76      0.76        58
weighted avg       0.76      0.76      0.76        58
```



4.4.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=5 (Figure 47), achieving a prediction accuracy score of 0.793. The confusion matrix (Figure 48) for the KNN model is analyzed:

- True Positive (TP): 29 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 17 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 9 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 3 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.864 (Figure 48).

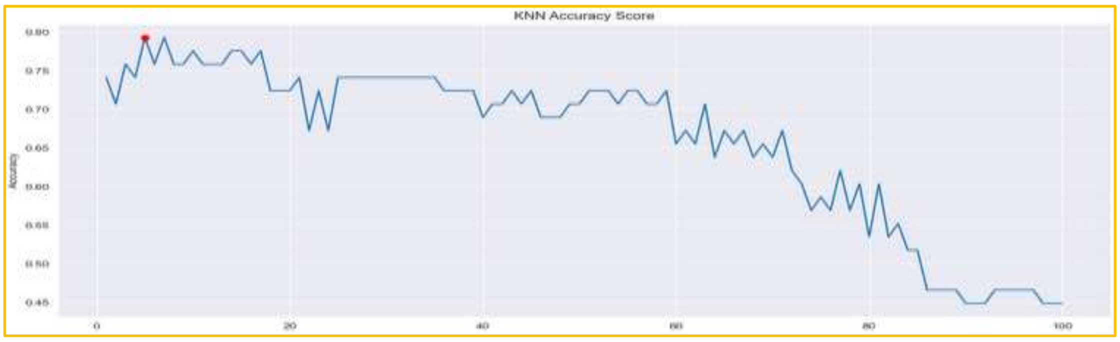
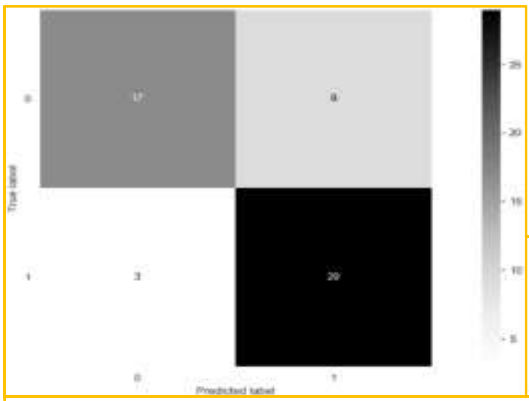
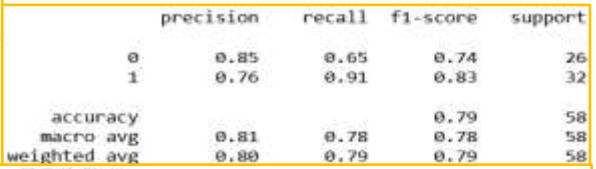


Figure 47. Ideal number of Nearest Neighbors (Undersampling and feature selection approach).





```
              precision    recall  f1-score   support

           0       0.85      0.65      0.74        26
           1       0.76      0.91      0.83        32

    accuracy                           0.79        58
   macro avg       0.81      0.78      0.78        58
weighted avg       0.80      0.79      0.79        58
```

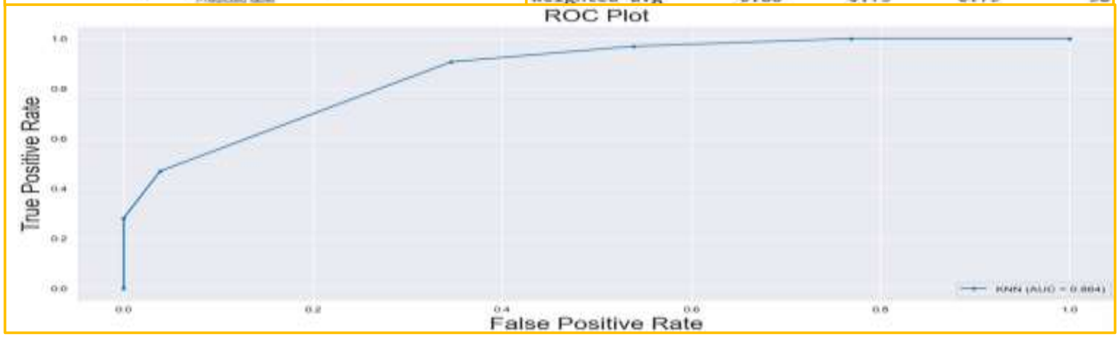


Figure 48. KNN's confusion matrix, ROC curve and statistics for the Undersampling and feature selection approach.

4.4.6 Support vector machines - Linear (SVM Linear)

The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 10, a degree of 1, and a gamma value of 0.0001.

The confusion matrix (Figure 49) for the Support vector machine linear model is examined:

- True Positive (TP): 21 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 19 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 11 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.766 (Figure 49).

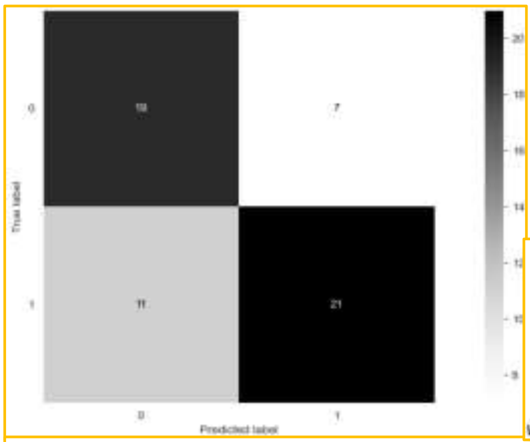
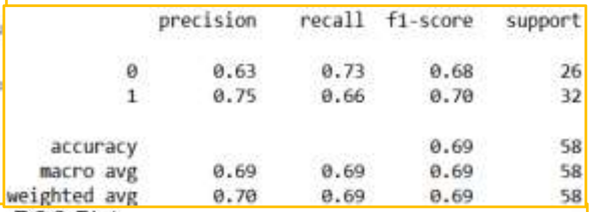


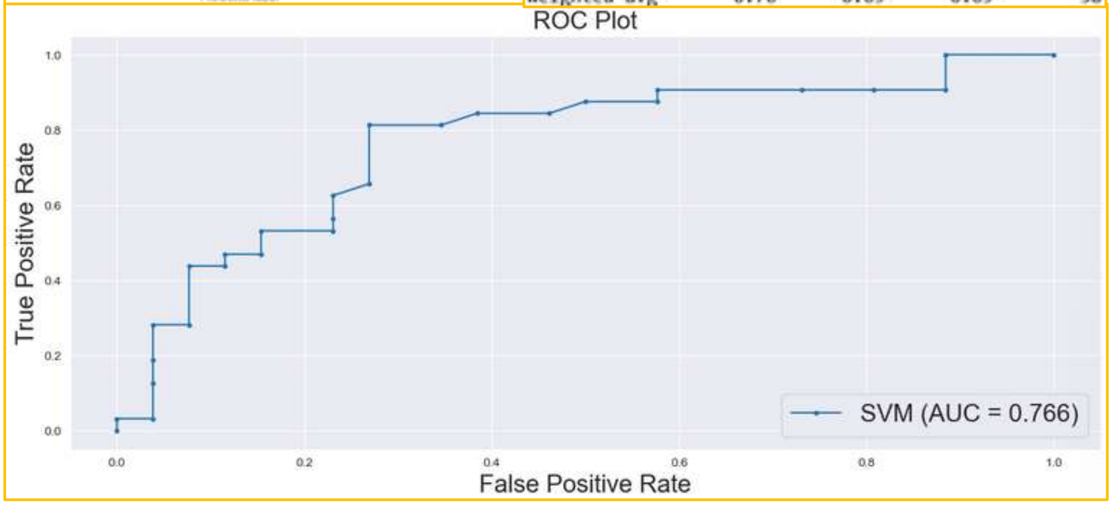
Figure 49. SVM's (Linear) confusion matrix, ROC curve and statistics for the Undersampling and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.63      0.73      0.68        26
           1       0.75      0.66      0.70        32

    accuracy                           0.69        58
   macro avg       0.69      0.69      0.69        58
weighted avg       0.70      0.69      0.69        58
```



4.4.7 Support vector machines - Radial (SVM Radial)

The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 50), obtained through grid search, is characterized by a regularization parameter 'C' of 100 and a gamma value of 0.01.

The confusion matrix for the Support Vector Machine (SVM) model with a radial kernel is tested:

- True Positive (TP): 20 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 20 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 12 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.785 (Figure 50).

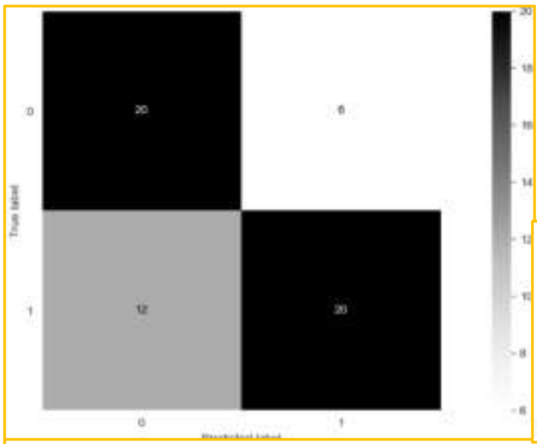
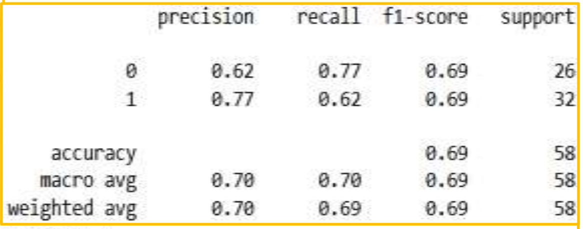


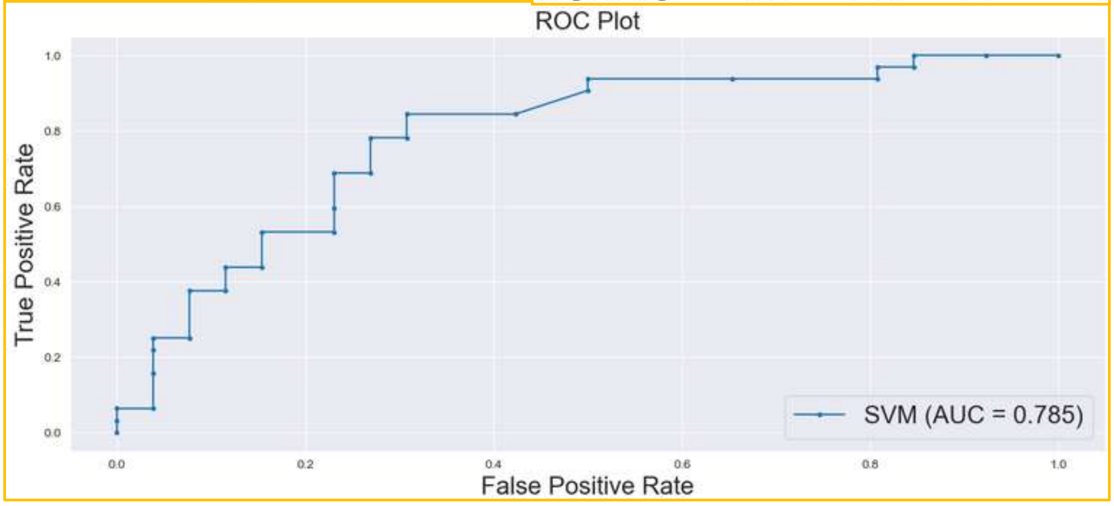
Figure 50. SVM’s (Radial) confusion matrix, ROC curve and statistics for the Undersampling and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.62      0.77      0.69        26
           1       0.77      0.62      0.69        32

    accuracy                           0.69        58
   macro avg       0.70      0.70      0.69        58
weighted avg       0.70      0.69      0.69        58
```



4.5 Oversampling

In response to the imbalanced nature of the dataset, with 203 instances representing the absence of a death event (class 0) and 96 instances indicating the occurrence of a death event (class 1), an oversampling technique was employed.

Oversampling (SMOTE) is a corrective measure that involves artificially boosting the representation of the minority class (class 1) in the dataset (Figure, 51).

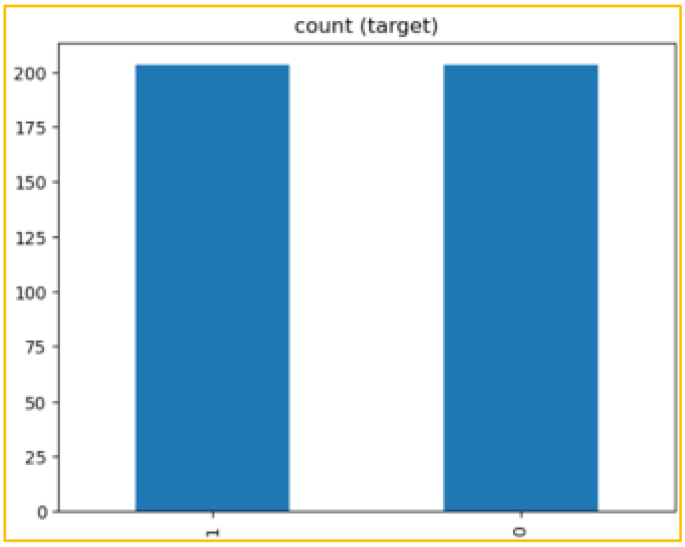


Figure 51. Final classes after the undersampling method.

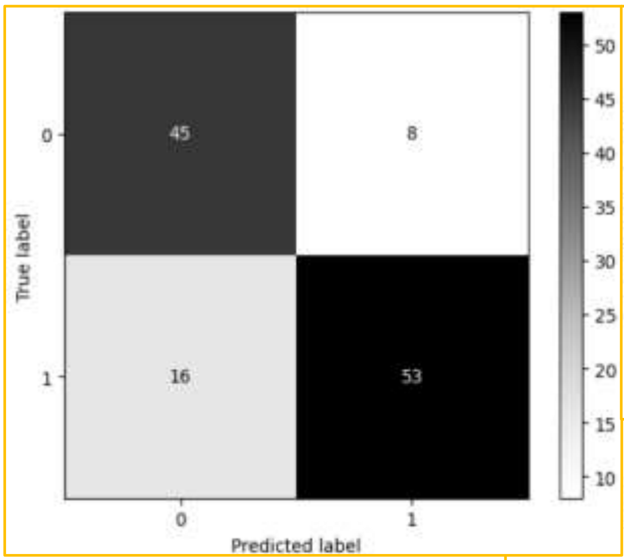
4.5.1 Random forest

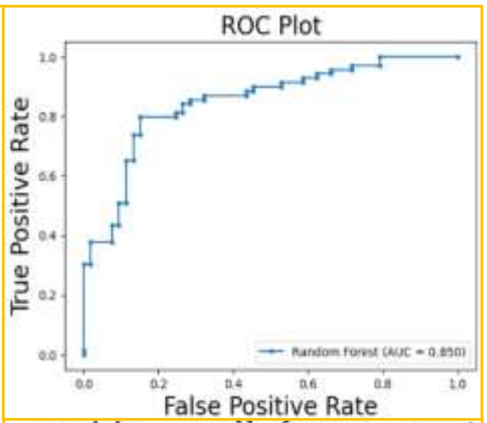
The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to True, 'criterion' utilizing entropy, 'max_depth' limited to 5 levels, 'min_samples_leaf' requiring a minimum of 1 samples per leaf node, 'min_samples_split' set at 4, and an ensemble of estimators consisting of 100 trees.

This confusion matrix provides a clear breakdown of the model's predictions. The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 52). In this case:

- True Positive (TP): 53 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 45 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 8 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 16 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.850 (Figure 52).





```
              precision    recall  f1-score   support

           0       0.74      0.85      0.79        53
           1       0.87      0.77      0.82        69

    accuracy                           0.80       122
   macro avg       0.80      0.81      0.80       122
weighted avg       0.81      0.80      0.80       122
```

Figure 52. Random forest's confusion matrix, ROC Curve and statistics after SMOTE.

4.5.2 Decision tree

The optimized Decision Tree model, obtained through grid search, is characterized by the following key parameters: a Gini criterion for node splitting, a maximum tree depth of 3 levels, a minimum of 1 samples required for leaf nodes, and a minimum of 2 samples for node splitting.

The confusion matrix (Figure 53) for the Decision Tree model provides a breakdown of the model's predictions:

- True Positive (TP):

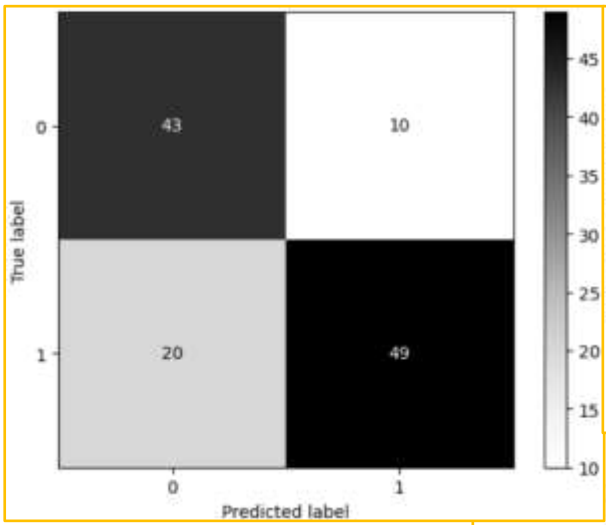
  49 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):

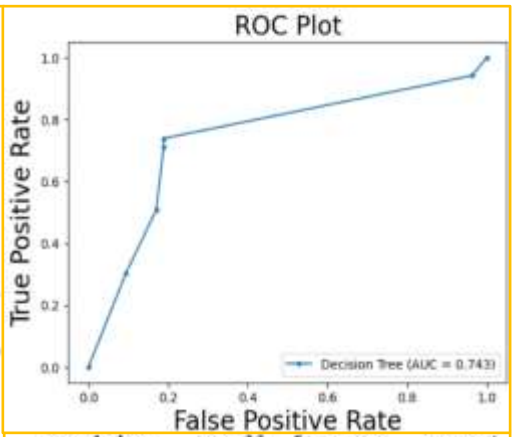
  43 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):

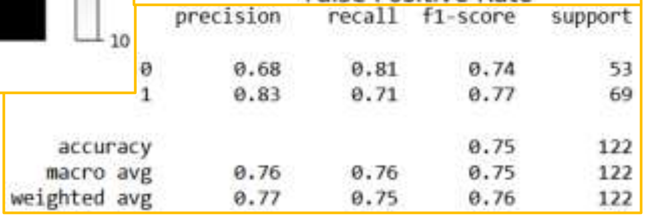
  10 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):

  20 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.743 (Figure 53).






```
              precision    recall  f1-score   support

           0       0.68      0.81      0.74        53
           1       0.83      0.71      0.77        69

    accuracy                           0.75       122
   macro avg       0.76      0.76      0.75       122
weighted avg       0.77      0.75      0.76       122
```

Figure 53. Decision tree's confusion matrix, ROC Curve and statistics after SMOTE.

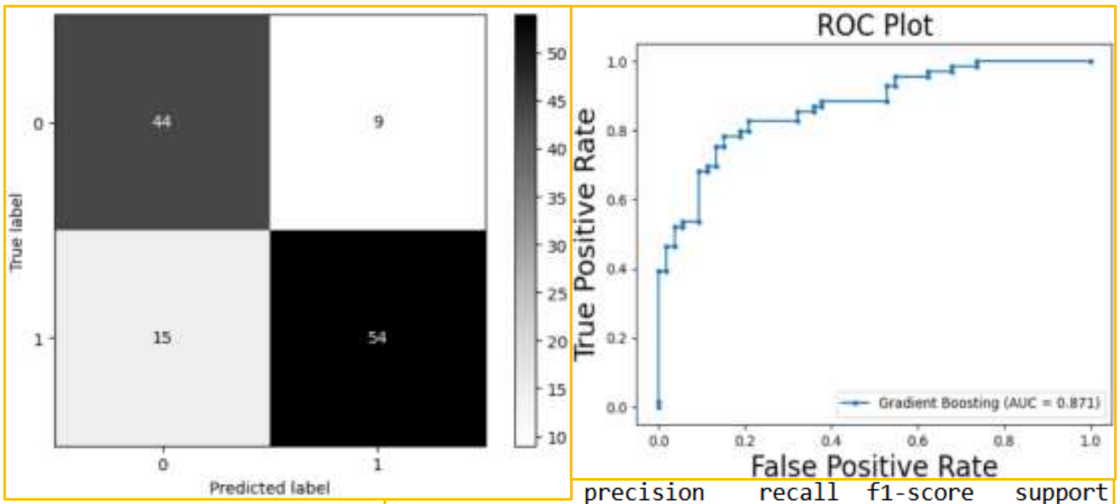
4.5.3 Gradient boosting

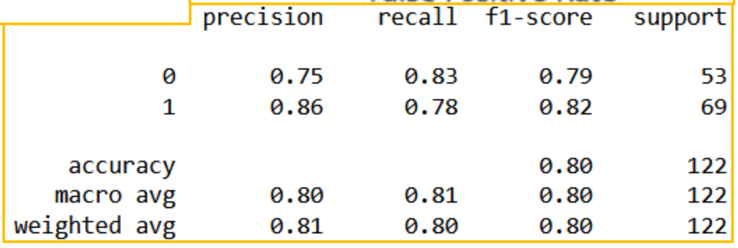
The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.025, a maximum tree depth of 7 levels, feature selection based on the logarithm base 2 of total features, 100 boosting stages (estimators), and a subsample fraction of 0.5.

The confusion matrix (Figure 54) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP):
  54 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):
  44 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):
  9 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):
  15 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.871 (Figure 54).





```
              precision    recall  f1-score   support

           0       0.75      0.83      0.79        53
           1       0.86      0.78      0.82        69

    accuracy                           0.80       122
   macro avg       0.80      0.81      0.80       122
weighted avg       0.81      0.80      0.80       122
```

Figure 54. Gradient boosting's confusion matrix, ROC Curve and statistics after SMOTE.

4.5.4 Logistic regression

The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 0.1 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

The confusion matrix (Figure 55) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP):

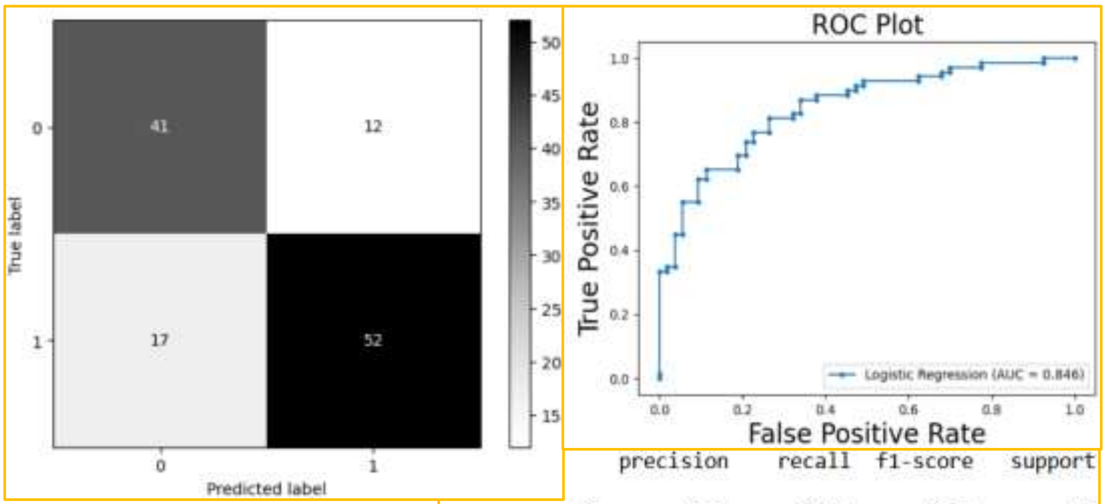
  52 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):

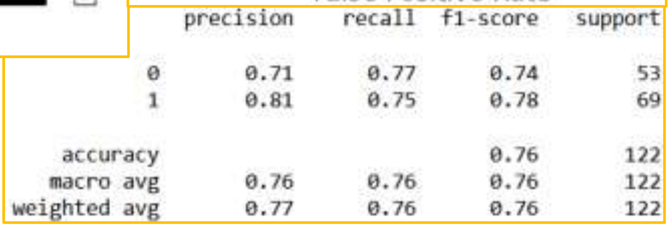
  41 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):

  12 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):

  17 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.846 (Figure 55).





```
              precision    recall  f1-score   support

           0       0.71      0.77      0.74        53
           1       0.81      0.75      0.78        69

    accuracy                           0.76       122
   macro avg       0.76      0.76      0.76       122
weighted avg       0.77      0.76      0.76       122
```

Figure 55. Logistic regression's confusion matrix, ROC curve and statistics after SMOTE.

4.5.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=25 (Figure 56), achieving a prediction accuracy score of 0.811.

The confusion matrix (Figure 57) for the K-Nearest Neighbors (KNN) model is analyzed:

- True Positive (TP):
  52 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):
  47 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):
  6 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):
  17 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.859 (Figure 57).

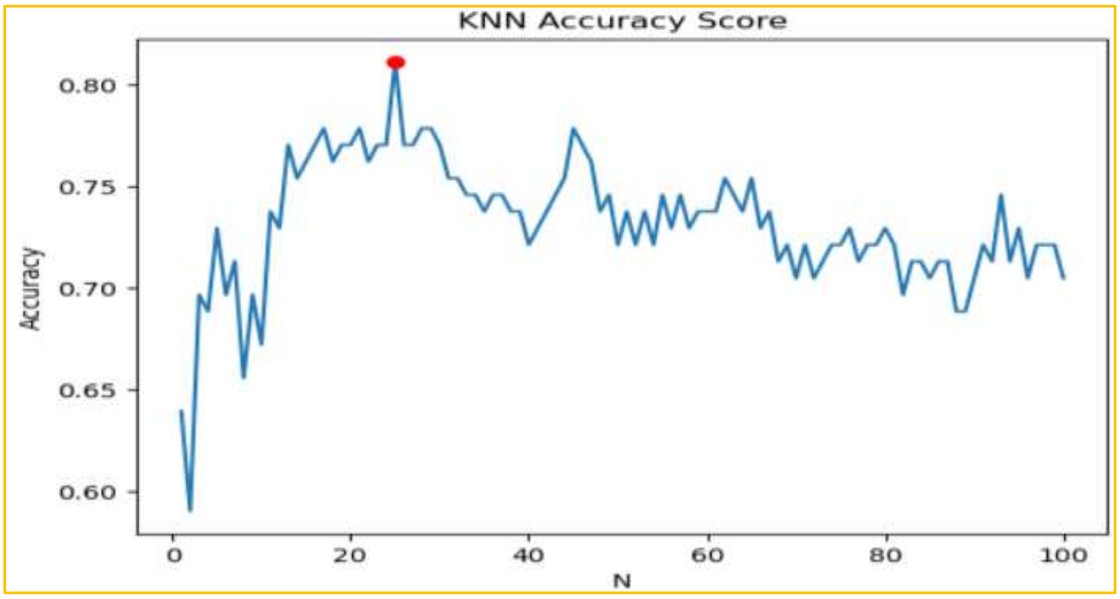
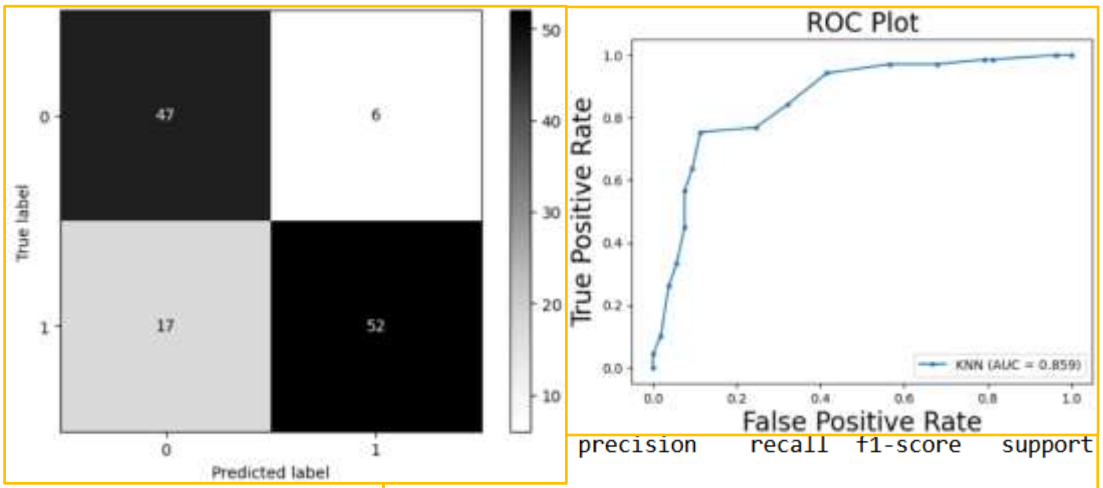


Figure 56. Ideal number of Nearest Neighbors (SMOTE).



```
              precision    recall  f1-score   support

           0       0.73      0.89      0.80        53
           1       0.90      0.75      0.82        69

    accuracy                           0.81       122
   macro avg       0.82      0.82      0.81       122
weighted avg       0.83      0.81      0.81       122
```

Figure 57.KNN's confusion matrix, ROC curve and statistics after SMOTE.

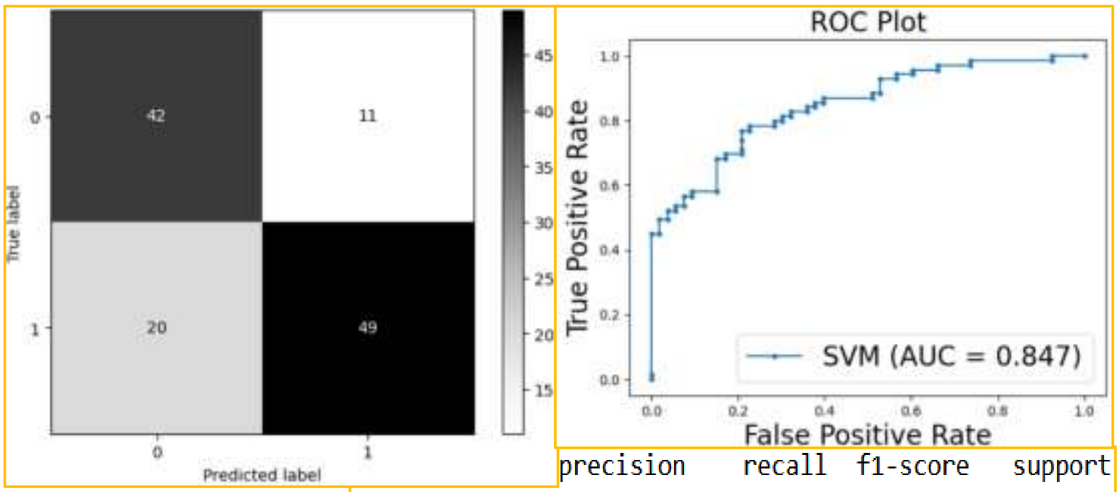
4.5.6 Support vector machines - Linear (SVM Linear)

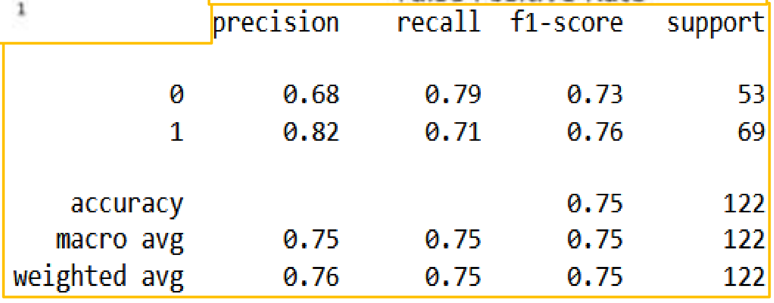
The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 0.01, a degree of 1, and a gamma value of 0.0001.

The confusion matrix (Figure 58) for the Support vector machine linear model is examined:

- True Positive (TP): 49 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 42 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 11 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):20 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.847 (Figure 58).





```
              precision    recall  f1-score   support

           0       0.68      0.79      0.73        53
           1       0.82      0.71      0.76        69

    accuracy                           0.75       122
   macro avg       0.75      0.75      0.75       122
weighted avg       0.76      0.75      0.75       122
```

Figure 58. SVM's (Linear) confusion matrix, ROC curve and statistics after SMOTE.

4.5.7 Support vector machines - Radial (SVM Radial)

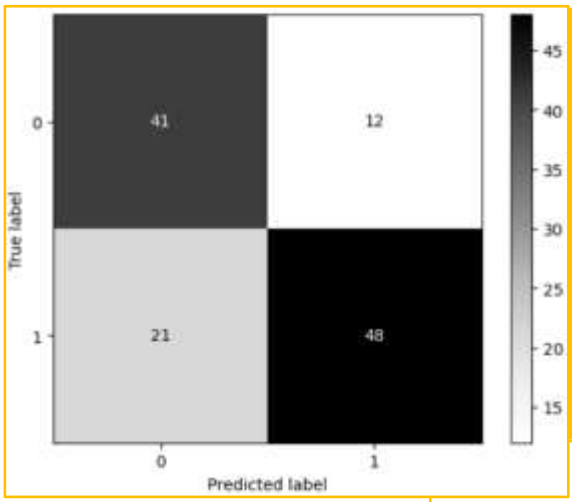
The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 59), obtained through grid search, is characterized by a regularization parameter 'C' of 100 and a gamma value of 0.001.

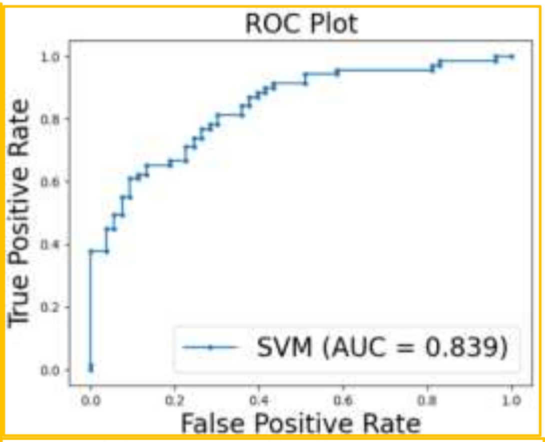
The confusion matrix for the Support Vector Machine (SVM) model with a radial kernel is tested:

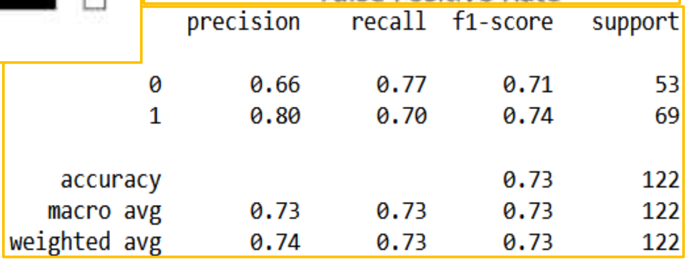
- True Positive (TP): 48 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 41 instances of actual class 0 correctly predicted as class 0.

- False Positive (FP): 12 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 21 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.839 (Figure 59).






```
              precision    recall  f1-score   support

           0       0.66      0.77      0.71        53
           1       0.80      0.70      0.74        69

    accuracy                           0.73       122
   macro avg       0.73      0.73      0.73       122
weighted avg       0.74      0.73      0.73       122
```

Figure 59. SVM’s (Radial) confusion matrix, ROC curve and statistics after SMOTE.

4.6 Oversampling and feature selection

In this section, a combination of oversampling (SMOTE) and feature selection approaches is applied.

4.6.1 Random forest

The optimized Random Forest model, resulting from a grid search, is configured with the following parameters: 'bootstrap' set to True, 'criterion' utilizing gini, 'max_depth' limited to 10 levels, 'min_samples_leaf' requiring a minimum of 4 samples per leaf node, 'min_samples_split' set at 2, and an ensemble of estimators consisting of 500 trees.

This confusion matrix provides a clear breakdown of the model's predictions. The elements along the main diagonal represent correct predictions, while off-diagonal elements indicate misclassifications (Figure 60). In this case:

- True Positive (TP): 53 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 45 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 8 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 16 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.882 (Figure 60).

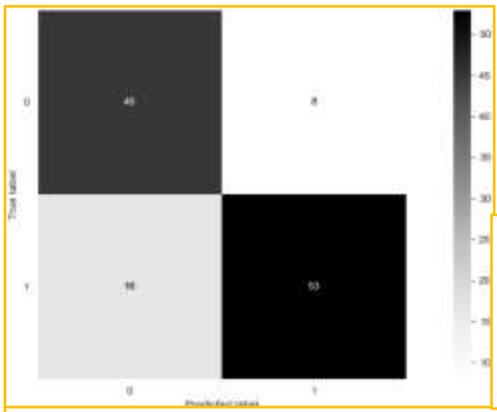
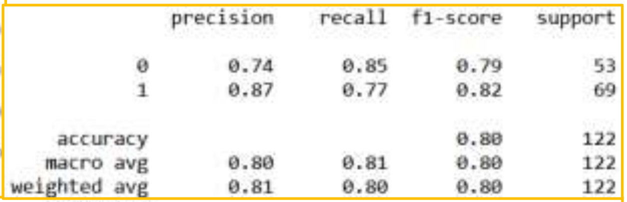


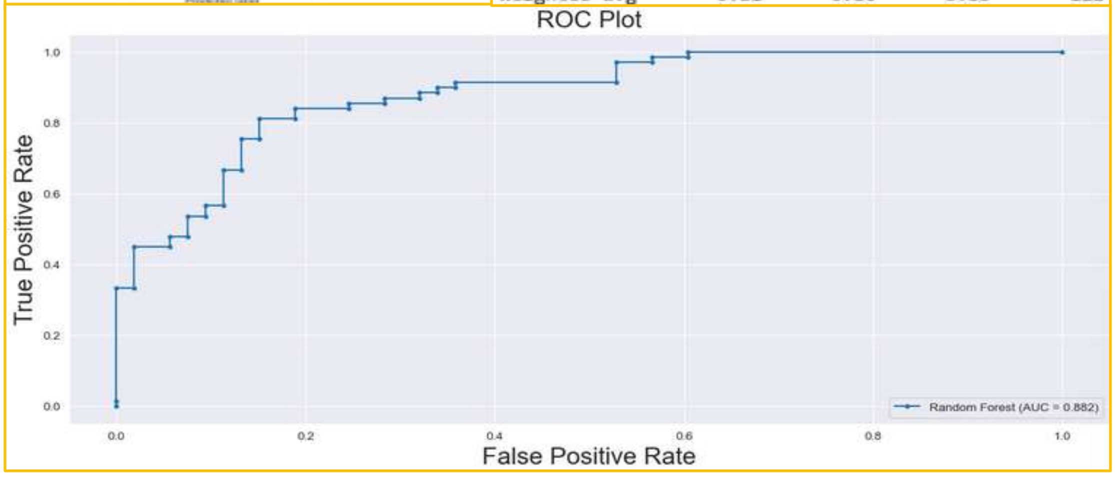
Figure 60. Random forest's confusion matrix, ROC Curve and statistics for SMOTE and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.74      0.85      0.79        53
           1       0.87      0.77      0.82        69

    accuracy                           0.80       122
   macro avg       0.80      0.81      0.80       122
weighted avg       0.81      0.80      0.80       122
```



4.6.2 Decision tree

The optimized Decision Tree model, obtained through grid search, is characterized by the following key parameters: a Gini criterion for node splitting, a maximum tree depth of 9 levels, a minimum of 2 samples required for leaf nodes, and a minimum of 8 samples for node splitting.

The confusion matrix (Figure 61) for the Decision Tree model provides a breakdown of the model's predictions:

- True Positive (TP): 49 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 42 instances of actual class 0 correctly predicted as class 0.

- False Positive (FP): 11 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 20 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.786 (Figure 61).

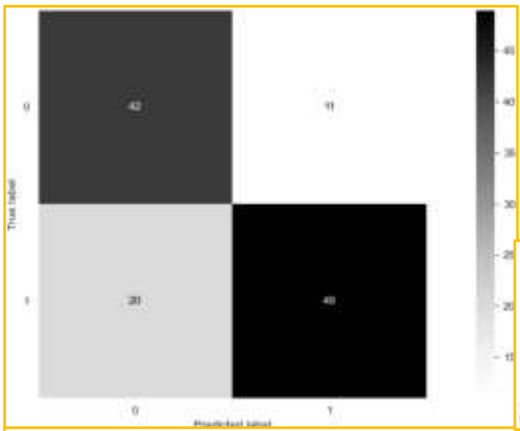
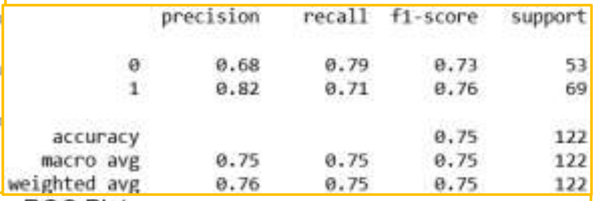


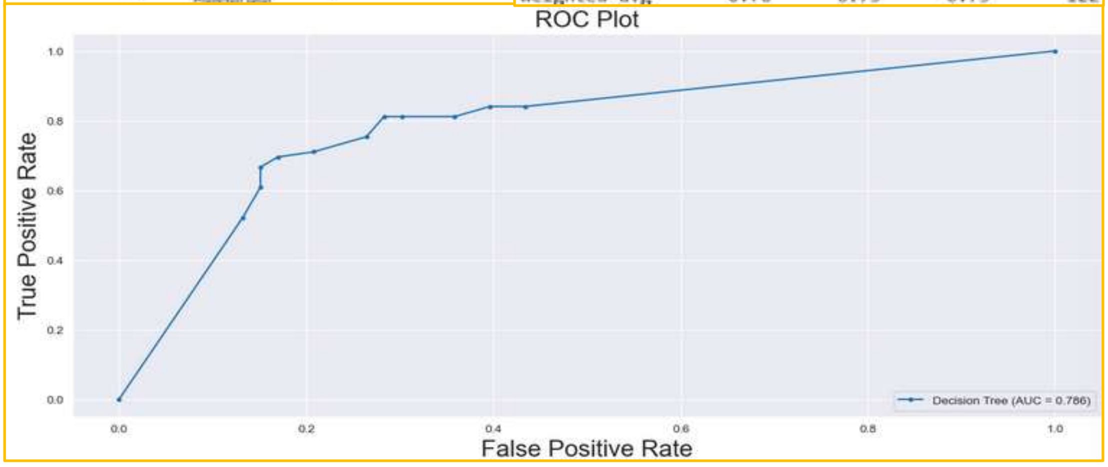
Figure 61. Decision tree's confusion matrix, ROC Curve and statistics for SMOTE and feature selection approach.



```
              precision    recall  f1-score   support

           0       0.68      0.79      0.73        53
           1       0.82      0.71      0.76        69

    accuracy                           0.75       122
   macro avg       0.75      0.75      0.75       122
weighted avg       0.76      0.75      0.75       122
```



4.6.3 Gradient boosting

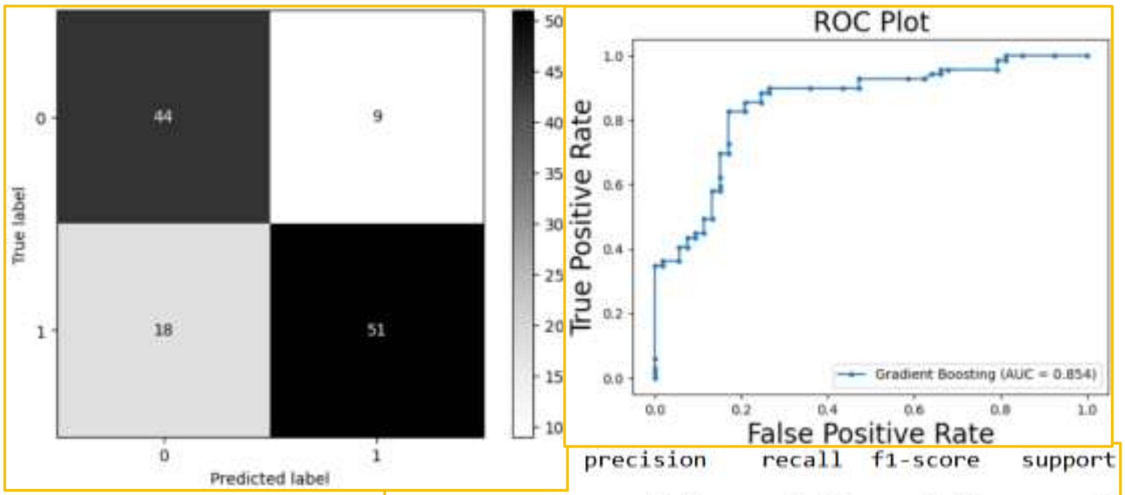
The Gradient Boosting model, refined through grid search, is characterized by the following parameters: a criterion for impurity measurement set to Friedman Mean Squared Error, a learning rate of 0.15, a maximum tree depth of 3 levels, feature selection based on the logarithm base 2 of total features, 10 boosting stages (estimators), and a subsample fraction of 0.8.

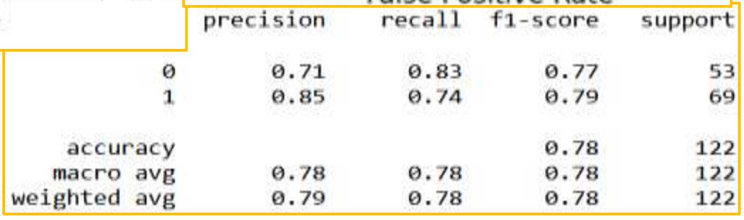
The confusion matrix (Figure 62) for the Gradient Boosting model analyzed below as follows:

- True Positive (TP): 51 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 44 instances of actual class 0 correctly predicted as class 0.

- False Positive (FP): 9 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 18 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.854 (Figure 62).





```
              precision    recall  f1-score   support

           0       0.71      0.83      0.77        53
           1       0.85      0.74      0.79        69

    accuracy                           0.78       122
   macro avg       0.78      0.78      0.78       122
weighted avg       0.79      0.78      0.78       122
```

Figure 62. Gradient boosting’s confusion matrix, ROC Curve and statistics for SMOTE and feature selection approach.

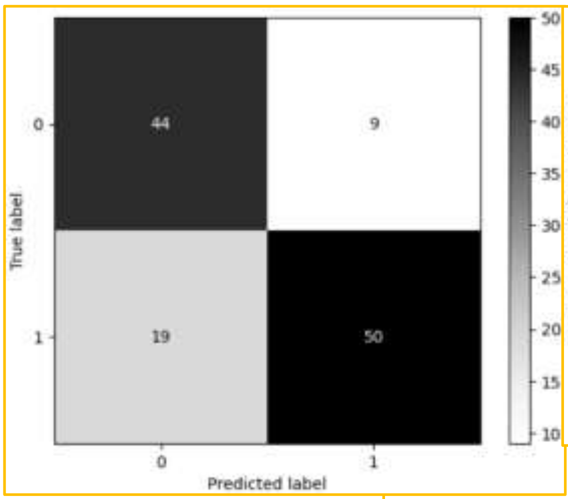
4.6.4 Logistic regression

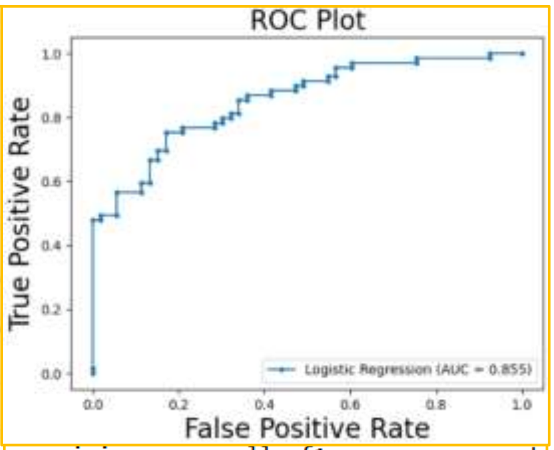
The optimized Logistic Regression model, obtained through grid search, is characterized by a regularization strength 'C' of 0.01 and an 'L2' penalty. This parameter configuration signifies a meticulous fine-tuning process, achieving a balance between model complexity and predictive accuracy tailored to the specific characteristics of the dataset.

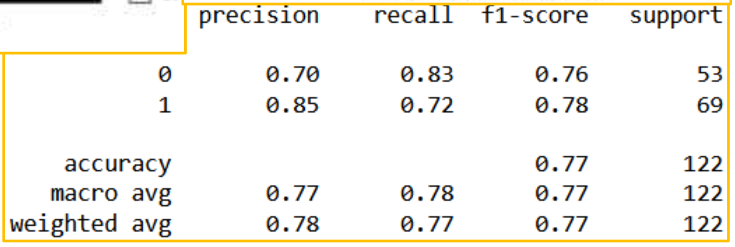
The confusion matrix (Figure 63) for the Logistic Regression model is presented and analyzed as follows:

- True Positive (TP): 50 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN): 44 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP): 9 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN): 19 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.855 (Figure 63).







```
              precision    recall  f1-score   support

           0       0.70      0.83      0.76        53
           1       0.85      0.72      0.78        69

    accuracy                           0.77       122
   macro avg       0.77      0.78      0.77       122
weighted avg       0.78      0.77      0.77       122
```

Figure 63. Logistic regression's confusion matrix, ROC curve and statistics for SMOTE and feature selection approach.

4.6.5 K-Nearest Neighbors (KNN)

After a research for the ideal number of Nearest Neighbors for the K-Nearest Neighbors (KNN) model, it is concluded that the value of the ideal number of the Nearest Neighbors is setting N=25 (Figure 64), achieving a prediction accuracy score of 0.787.

The confusion matrix (Figure 65) for the K-Nearest Neighbors (KNN) model is analyzed:

- True Positive (TP):

  50 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):

  46 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):

  7 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):

  19 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area Under the Curve (ROC AUC) value is 0.850 (Figure 65).

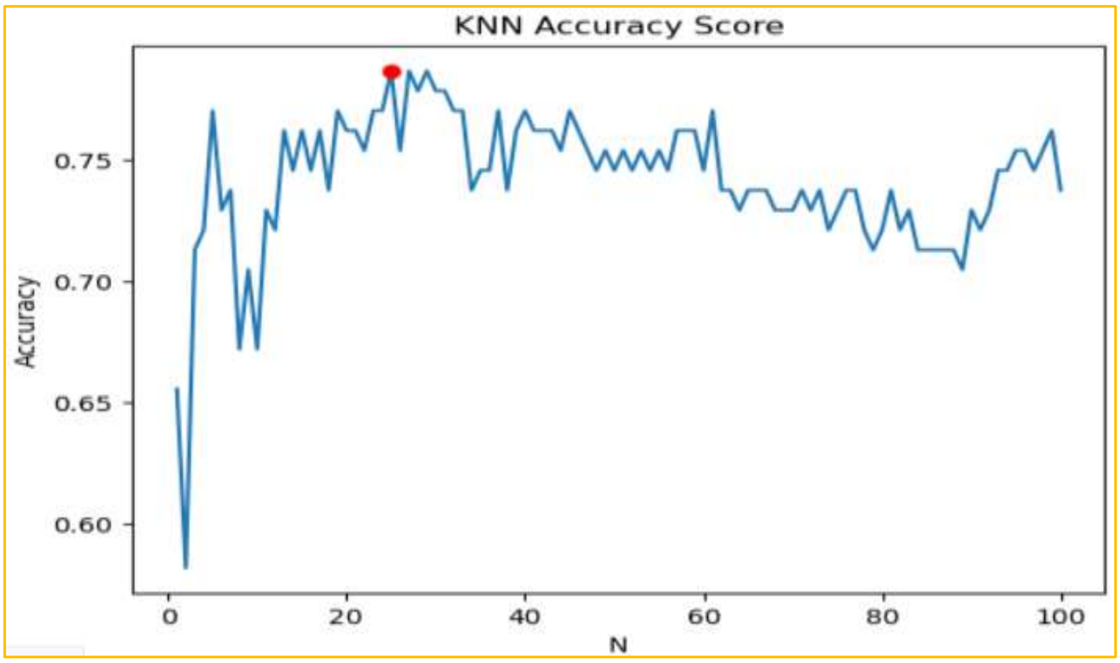
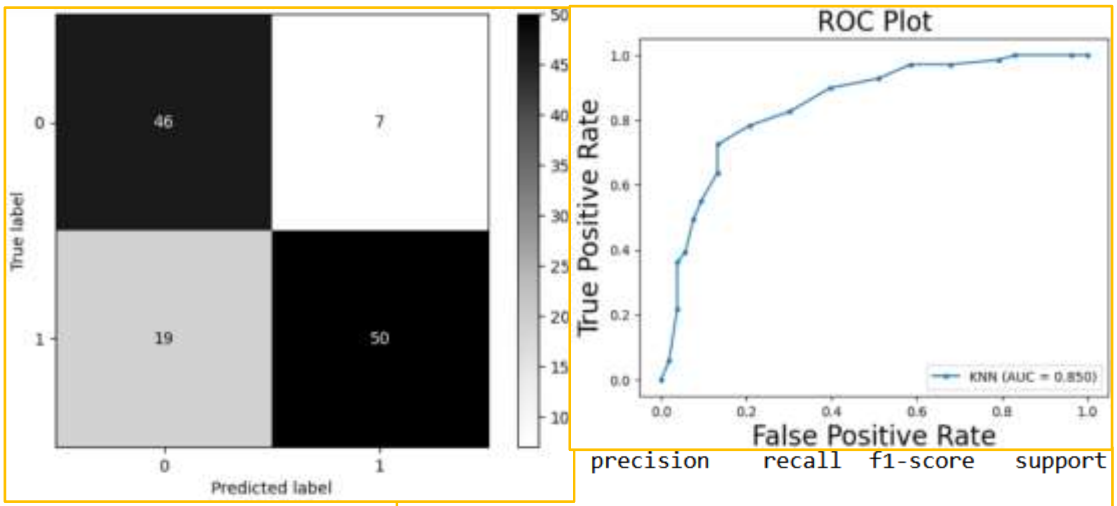
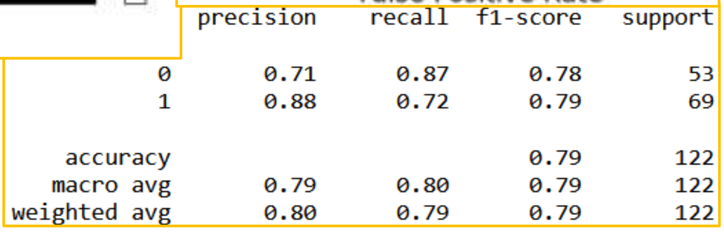


Figure 64.Ideal number of Nearest Neighbors (Smote and feature selection approach).





```
              precision    recall  f1-score   support

           0       0.71      0.87      0.78        53
           1       0.88      0.72      0.79        69

    accuracy                           0.79       122
   macro avg       0.79      0.80      0.79       122
weighted avg       0.80      0.79      0.79       122
```

Figure 65. KNN’s confusion matrix, ROC curve and statistics for SMOTE and feature selection approach.

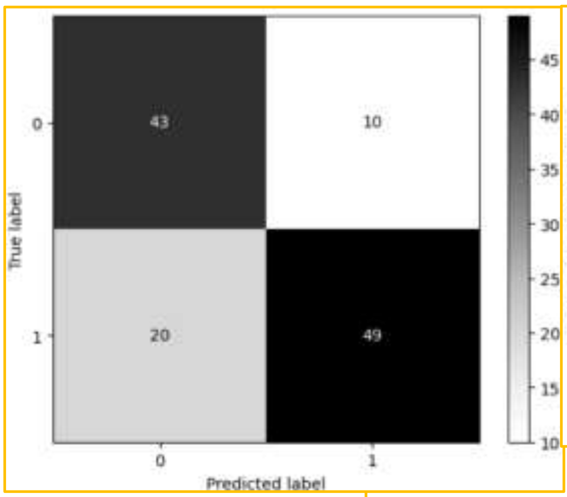
4.6.6 Support vector machines - Linear (SVM Linear)

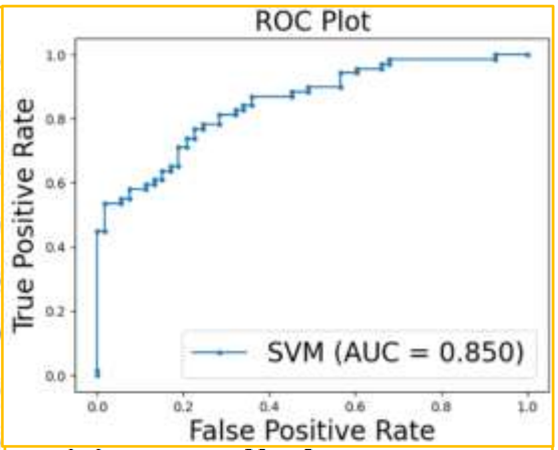
The optimized Support Vector Machine (SVM) model with a linear kernel, obtained through grid search, is characterized by a regularization parameter (C) of 0.01, a degree of 1, and a gamma value of 0.0001.

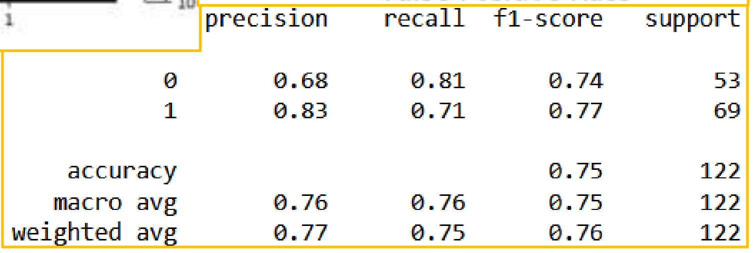
The confusion matrix (Figure 66) for the Support vector machine linear model is examined:

- True Positive (TP):
  49 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):
  43 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):
  10 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):
  20 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.850 (Figure 66).







```
              precision    recall  f1-score   support

           0       0.68      0.81      0.74        53
           1       0.83      0.71      0.77        69

    accuracy                           0.75       122
   macro avg       0.76      0.76      0.75       122
weighted avg       0.77      0.75      0.76       122
```

Figure 66.SVM's (Linear) confusion matrix, ROC curve and statistics for SMOTE and feature selection approach.

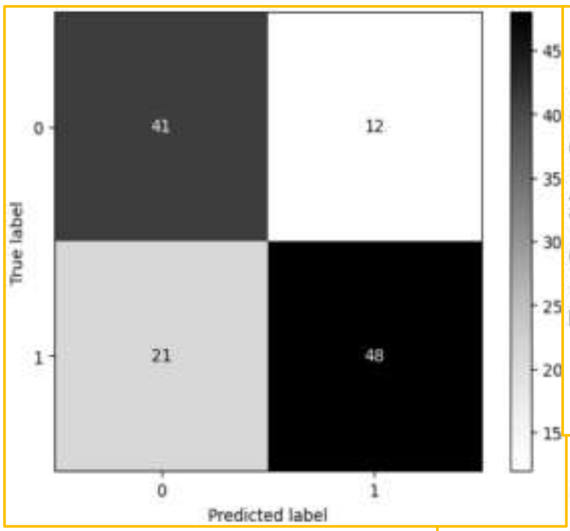
4.6.7 Support vector machines - Radial (SVM Radial)

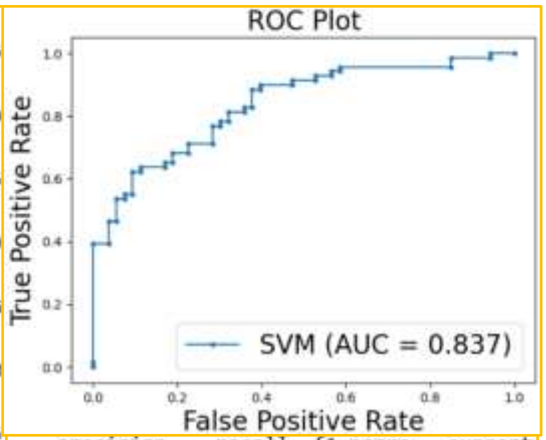
The optimized Support Vector Machine (SVM) model with a radial kernel (Figure 67), obtained through grid search, is characterized by a regularization parameter 'C' of 100 and a gamma value of 0.001.

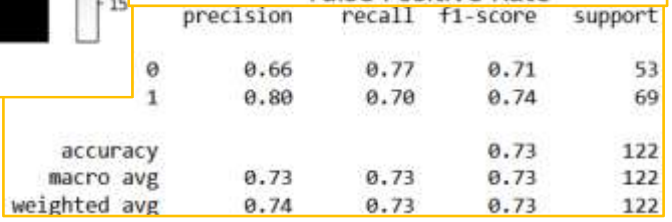
The confusion matrix for the Support Vector Machine (SVM) model with a radial kernel is tested:

- True Positive (TP):
  48 instances of actual class 1 correctly predicted as class 1.
- True Negative (TN):
  41 instances of actual class 0 correctly predicted as class 0.
- False Positive (FP):
  12 instances of actual class 0 incorrectly predicted as class 1.
- False Negative (FN):
  21 instances of actual class 1 incorrectly predicted as class 0.

The Receiver Operating Characteristic - Area under the Curve (ROC AUC) value is 0.837 (Figure 67).







```
              precision    recall  f1-score   support

           0       0.66      0.77      0.71        53
           1       0.80      0.70      0.74        69

    accuracy                           0.73       122
   macro avg       0.73      0.73      0.73       122
weighted avg       0.74      0.73      0.73       122
```

Figure 67. SVM’s (Radial) confusion matrix, ROC curve and statistics for SMOTE and feature selection approach.

4.7 Principal component analysis

Seeking to apply the Principal Component Analysis (PCA) method to the initial dataset under examination in this study, in order to determine the ranking of the coefficients resulting from it, a different data normalization method (Min-Max scaling) was employed. The objective was to minimize deviations among values and standardize all variables to the same scale (0-1).

In Figure 68, the Principal components (PCs) are depicted for a dataset with N=11 variables, illustrating the correlation coefficients of the eigenvectors for each principal component.

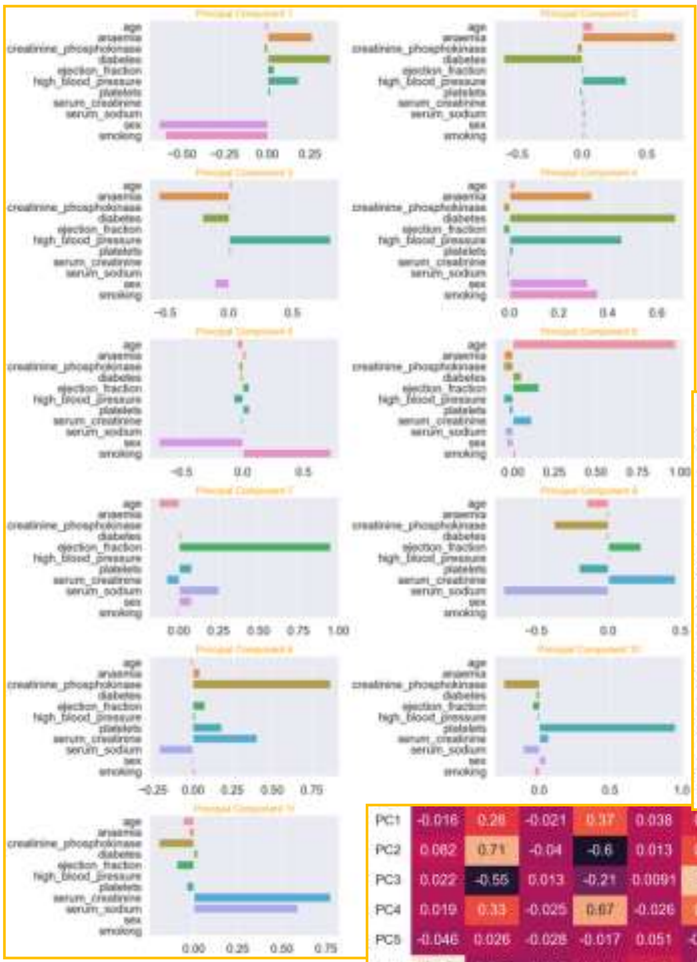
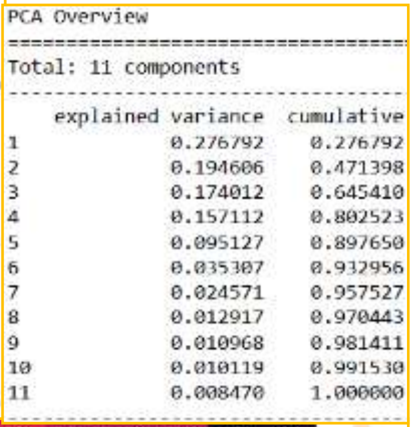
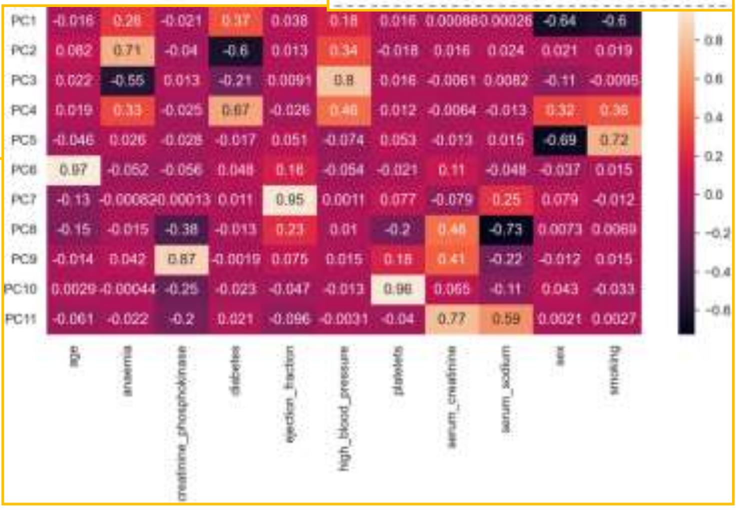


Figure 68.
Principal components (PCs), their values and the cumulative explained variable.


```
PCA Overview
=====================================
Total: 11 components
-------------------------------------
    explained variance   cumulative
1             0.276792     0.276792
2             0.194606     0.471398
3             0.174012     0.645410
4             0.157112     0.802523
5             0.095127     0.897650
6             0.035307     0.932956
7             0.024571     0.957527
8             0.012917     0.970443
9             0.010968     0.981411
10            0.010119     0.991530
11            0.008470     1.000000
-------------------------------------
```



	age	anaemia	creatinine_phosphokinase	diabetes	ejection_fraction	high_blood_pressure	platelets	serum_creatinine	serum_sodium	sex	smoking
PC1	-0.016	0.28	-0.021	0.37	0.038	0.18	0.016	0.00088	0.00026	-0.64	-0.6
PC2	0.082	0.71	-0.04	-0.6	0.013	0.34	-0.018	0.016	0.024	0.021	0.019
PC3	0.022	-0.55	0.013	-0.21	0.0091	0.8	0.016	-0.0061	0.0082	-0.11	-0.0005
PC4	0.019	0.33	-0.025	0.67	-0.026	0.46	0.012	-0.0064	-0.013	0.32	0.36
PC5	-0.046	0.026	-0.028	-0.017	0.051	-0.074	0.053	-0.013	0.015	-0.69	0.72
PC6	0.97	-0.052	-0.056	0.048	0.16	-0.054	-0.021	0.11	-0.048	-0.037	0.015
PC7	-0.13	-0.00082	0.00013	0.011	0.95	0.0011	0.077	-0.079	0.25	0.079	-0.012
PC8	-0.15	-0.015	-0.38	-0.013	0.23	0.01	-0.2	0.46	-0.73	0.0073	0.0069
PC9	-0.014	0.042	0.87	-0.0019	0.075	0.015	0.18	0.41	-0.22	-0.012	0.015
PC10	0.0029	-0.00044	-0.25	-0.023	-0.047	-0.013	0.98	0.065	-0.11	0.043	-0.033
PC11	-0.061	-0.022	-0.2	0.021	-0.096	-0.0031	-0.04	0.77	0.59	0.0021	0.0027

In the bar chart below (Figure 69), the percentage of variance (vertical axis) for each principal component is illustrated. Based on this bar chart, the histogram (Figure 70) was constructed, pinpointing the elbow in the sixth principal component (PC6). The elbow represents the point beyond which the components contribute less variance to our data. Using six components, we can explain 93% of the variability in the original data.

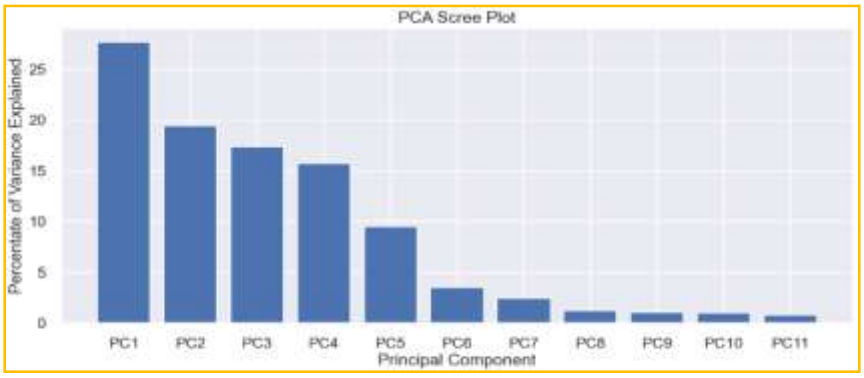


Figure 69. Bar chart of variance in principal components.

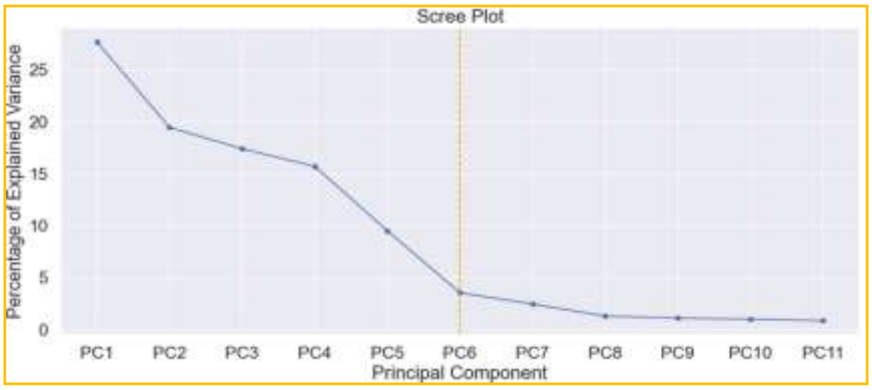


Figure 70. Histogram of Variance in Principal Components.

Using the six Principal components (PC6) derived from the information mentioned earlier, all machine learning models implemented in this study were executed. The summarized results of these models are presented in Table 11. Additionally, the results tables for the statistics of the remaining methods executed are provided, aiming to facilitate a comparison between models and methods (Table 12).

Table 11. Principal component analysis (PCA).

	Accuracy	F1 score	TP rate	TN rate	ROC/AUC	Precision
Random Forest	0.62	0.57	0.14	0.85	0.508	0.56
Decision Tree	0.62	0.52	0.00	0.92	0.338	0.45
Logistic Regression	0.69	0.60	0.10	0.97	0.588	0.66
Gradient Boosting	0.64	0.55	0.03	0.93	0.577	0.52
KNN	0.68	0.60	0.10	0.95	0.531	0.60
SVM (linear-radial)	0.68	0.55	0.00	1.00	0.557	0.46

Table 12. Statistics of the other approaches.

	Accuracy	F1 score	TP rate	TN rate	ROC/AUC	Precision
Full Dataset						
Random Forest	0.76	0.73	0.39	0.92	0.766	0.74
Decision Tree	0.68	0.65	0.29	0.85	0.749	0.65
Logistic Regression	0.74	0.73	0.43	0.89	0.733	0.73
Gradient Boosting	0.69	0.65	0.25	0.89	0.683	0.65
KNN	0.71	0.69	0.36	0.87	0.681	0.69
SVM (Linear)	0.74	0.73	0.43	0.89	0.736	0.73
SVM (Radial)	0.67	0.68	0.61	0.69	0.686	0.70
Feature Selection						
Random Forest	0.76	0.74	0.46	0.89	0.799	0.74
Decision Tree	0.76	0.73	0.36	0.94	0.746	0.75
Logistic Regression	0.77	0.74	0.36	0.95	0.716	0.77
Gradient Boosting	0.74	0.72	0.39	0.90	0.741	0.73
KNN	0.79	0.78	0.50	0.92	0.733	0.78
SVM (Linear)	0.77	0.74	0.36	0.95	0.718	0.77
SVM (Radial)	0.74	0.73	0.46	0.87	0.804	0.73
Undersampling						
Random Forest	0.72	0.72	0.72	0.73	0.833	0.73
Decision Tree	0.64	0.64	0.69	0.58	0.706	0.64
Logistic Regression	0.62	0.61	0.47	0.81	0.722	0.66
Gradient Boosting	0.66	0.66	0.59	0.73	0.774	0.67
KNN	0.67	0.67	0.56	0.81	0.706	0.70
SVM (Linear)	0.69	0.69	0.62	0.77	0.793	0.70
SVM (Radial)	0.71	0.71	0.66	0.77	0.768	0.72
Undersampling and feature selection						
Random Forest	0.76	0.76	0.75	0.77	0.819	0.76
Decision Tree	0.71	0.70	0.59	0.85	0.745	0.74
Logistic Regression	0.76	0.76	0.78	0.73	0.767	0.76
Gradient Boosting	0.74	0.74	0.72	0.77	0.796	0.75
KNN	0.79	0.79	0.91	0.65	0.864	0.80
SVM (Linear)	0.69	0.69	0.66	0.73	0.766	0.70
SVM (Radial)	0.69	0.69	0.62	0.77	0.785	0.70
Oversampling						
Random Forest	0.80	0.80	0.77	0.85	0.850	0.81
Decision Tree	0.75	0.76	0.71	0.81	0.743	0.77
Logistic Regression	0.76	0.76	0.75	0.77	0.846	0.77
Gradient Boosting	0.80	0.80	0.78	0.83	0.871	0.81
KNN	0.81	0.81	0.75	0.89	0.859	0.83
SVM (Linear)	0.75	0.75	0.71	0.79	0.847	0.76
SVM (Radial)	0.73	0.73	0.70	0.77	0.839	0.74
Oversampling and feature selection						
Random Forest	0.80	0.80	0.77	0.85	0.882	0.81
Decision Tree	0.75	0.75	0.71	0.79	0.786	0.76
Logistic Regression	0.77	0.77	0.72	0.83	0.855	0.78
Gradient Boosting	0.78	0.78	0.74	0.83	0.854	0.79
KNN	0.79	0.79	0.72	0.87	0.850	0.80
SVM (Linear)	0.75	0.76	0.71	0.81	0.850	0.77
SVM (Radial)	0.73	0.73	0.70	0.77	0.837	0.74

The cumulative charts corresponding to the above Table are presented in the Appendix.

Chapter 5. CONCLUSIONS

Chapter 5 discusses the implications of the thesis' results and concludes with a discussion of the thesis' limitations and recommendations for future research.

5.1 Thesis conclusions

In conclusion, the exploration into predicting death events in heart failure patients through Machine Learning has provided valuable insights. The dataset's meticulous examination confirmed its completeness, eliminating the need for modifications and identified medically plausible outliers. Successful feature normalization ensured consistent scaling, a crucial step for bolstering model performance. The application of the Extra Tree Classifier for feature selection effectively identified and retained crucial features, enhancing model efficiency and interpretability. Notably, excluding the "time" feature underscored distinctions among our experimental approaches. The array of Machine Learning models includes Random Forest, Decision Tree, SVM, Logistic Regression, KNN, and Gradient Boosting. These models demonstrated multiple important results. Hyperparameter optimization through Grid Search further honed their performance. Experimentation with diverse conditions, including feature selection, undersampling, SMOTE oversampling, and their combinations, provided a nuanced understanding of model behavior.

In this study, the importance of feature selection is notably evident. As indicated in the earlier chapters of the thesis, utilizing the two most crucial features in the dataset—serum creatinine and ejection fraction—results in either comparable or superior outcomes compared to the examined models (refer to subchapter 4.2). This is in contrast to the scenario where all features are employed (refer to subchapter 4.1). Additionally, the significance of undersampling and oversampling is highlighted by the noticeable improvement in classifying the minority class, specifically class 1 (death event = occurred). Combining undersampling-oversampling methods with the selection of the most significant features in the dataset has the potential to bring about substantial improvements in the results of machine learning models. In summary, this thesis establishes a sturdy foundation for comprehending the dynamics of Machine Learning models in predicting death events among heart failure patients. To conclude, it is crucial to experiment with various models and methods to identify the most optimal and suitable one before making the final selection.

5.2 Practical and theoretical implications

This thesis provides useful implications, for the practitioners in the medical and bioinformatics sector. More specifically,

- Helping Doctors Make Decisions: The study's practical side means creating tools that doctors can use. The computer models, especially when we use feature selection and SMOTE, offer practical help for doctors to predict which heart failure patients might face death. This aids in timely actions and better personalized care.
- Making Models Easier to Understand: Picking the right features not only makes the models work better but also makes it easier for doctors to understand why the models predict certain outcomes. This is important for doctors who want to use computer predictions in their decision-making.
- Guiding Healthcare Strategies: The thesis' practical impact extends to how hospitals and healthcare providers use data. The successful use of SMOTE shows its relevance in handling imbalanced data, helping make more accurate predictions. This can guide how healthcare decisions are made in the future.

In summary, this research simplifies how machines predict deaths in heart failure patients. It shows that using the right features and balancing data can improve predictions, making it useful for doctors and healthcare decisions.

This thesis also provides several scholarly contributions. More extensively:

- Advancing Predictive Models: This dissertation boosts the understanding of how machines can predict death events in heart failure patients. It shows that using different computer models, along with picking important features and balancing data, can improve predictions in medical scenarios.
- Importance of Picking Key Features: The thesis highlights how choosing the right features (like specific health indicators) is crucial. This idea can be applied more broadly in medical data analysis, emphasizing the need to focus on important factors for accurate predictions.
- Balancing Data Effectively: By comparing two methods (undersampling and SMOTE), the thesis suggests that adding more data through SMOTE helps computer models perform better, especially when predicting rare events like deaths in heart failure patients.

5.3 Limitations and suggestions for future research

In acknowledging the limitations of this study, it's crucial to note that our dataset was relatively small, with only 299 patients. A larger dataset would have given more reliable results and a better understanding of how well this thesis' methods, especially PCA, work. It would also help to determine if these methods are truly necessary.

Valuable information such as the patients' physical characteristics (like height, weight, and body mass index) and their job histories, could have helped us identify more risk factors for cardiovascular diseases. Additionally, not having an external dataset from a different location limited the ability to validate the thesis findings with a separate group.

Looking ahead, alternative normalization methods can be explored, such as the robust scaler, and fine-tune our approach by adjusting hyperparameters using methods like random search. Furthermore, future research avenues could explore additional techniques for dimensionality reduction in small datasets, ensuring a more comprehensive evaluation of model performance (Chicco & Jurman, 2020). Machine learning methods can be also extend to different datasets related to cardiovascular diseases (Masino et al., 2019; Aushev et al., 20018; Patrício et al., 2018) and other illnesses like cervical cancer (Fernandes et al. 2018), neuroblastoma (Maggio et al., 2018), breast cancer (Yunus et al., 2018), and amyotrophic lateral sclerosis (Kueffner et al., 2019).

REFERENCES

1. Agrawal, R. (2014). K-nearest neighbor for uncertain data. International Journal of Computer Applications, 105(11), 13-16.
2. Ahmad, T., Munir, A., Bhatti, S. H., Aftab, M., & Raza, M. A. (2017). Survival analysis of heart failure patients: A case study. PloS one, 12(7), e0181001.
3. Ahsan, M. M., Mahmud, M. P., Saha, P. K., Gupta, K. D., & Siddique, Z. (2021). Effect of data scaling methods on machine learning algorithms and model performance. Technologies, 9(3), 52.
4. Al'Aref, S. J., Anchouche, K., Singh, G., Slomka, P. J., Kolli, K. K., Kumar, A., & Min, J. K. (2019). Clinical applications of machine learning in cardiovascular disease and its relevance to cardiac imaging. European heart journal, 40(24), 1975-1986.
5. Al'Aref, S. J., Singh, G., van Rosendael, A. R., Kolli, K. K., Ma, X., Maliakal, G., & Minutello, R. M. (2019). Determinants of in-hospital mortality after percutaneous coronary intervention: a machine learning approach. Journal of the American Heart Association, 8(5), e011160.
6. Alanazi, A. (2022). Using machine learning for healthcare challenges and opportunities. Informatics in Medicine Unlocked, 30, 100924.
7. Alloghani, M., Al-Jumeily, D., Mustafina, J., Hussain, A., & Aljaaf, A. J. (2020). A systematic review on supervised and unsupervised machine learning algorithms for data science. Supervised and unsupervised learning for data science, 3-21.
8. Ambale-Venkatesh, B., Yang, X., Wu, C. O., Liu, K., Hundley, W. G., McClelland, R., & Lima, J. A. (2017). Cardiovascular event prediction by machine learning: the multi-ethnic study of atherosclerosis. Circulation research, 121(9), 1092-1101.
9. Amer, E., Kwak, K. S., & El-Sappagh, S. (2022). Context-based fake news detection model relying on deep learning models. Electronics, 11(8), 1255.
10. Aushev, A., Ripoll, V. R., Vellido, A., Aletti, F., Pinto, B. B., Herpain, A., & Bendjelid, K. (2018). Feature selection for the accurate prediction of septic and cardiogenic shock ICU mortality in the acute phase. PloS one, 13(11), e0199089.
11. Awad, M., & Khanna, R. (2015). Efficient learning machines: theories, concepts, and applications for engineers and system designers (p. 268). Springer

nature.

12. Bentéjac, C., Csörgő, A., & Martínez-Muñoz, G. (2021). A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review, 54, 1937-1967.
13. Biau, G., & Scornet, E. (2016). A random forest guided tour. Test, 25, 197-227.
14. Binder, T., Sandmann, A., Sures, B., Friege, G., Theyssen, H., & Schmiemann, P. (2019). Assessing prior knowledge types as predictors of academic achievement in the introductory phase of biology and physics study programmes using logistic regression. International Journal of STEM Education, 6, 1-14.
15. Boateng, E. Y., & Abaye, D. A. (2019). A review of the logistic regression model with emphasis on medical research. Journal of data analysis and information processing, 7(4), 190-207.
16. Borg, M., Englund, C., Wnuk, K., Duran, B., Levandowski, C., Gao, S. & Törnqvist, J. (2018). Safely entering the deep: A review of verification and validation for machine learning and a challenge elicitation in the automotive industry. arXiv preprint arXiv:1812.05389.
17. Botchkarev, A. (2019). A new typology design of performance metrics to measure errors in machine learning regression algorithms. Interdisciplinary Journal of Information, Knowledge, and Management, 14, 045-076.
18. Brereton, R. G., & Lloyd, G. R. (2010). Support vector machines for classification and regression. Analyst, 135(2), 230-267.
19. Buchan, T. A., Ross, H. J., McDonald, M., Billia, F., Delgado, D., Posada, J. D., & Alba, A. C. (2019). Physician prediction versus model predicted prognosis in ambulatory patients with heart failure. The Journal of Heart and Lung Transplantation, 38(4), S381.
20. Burkart, N., & Huber, M. F. (2021). A survey on the explainability of supervised machine learning. Journal of Artificial Intelligence Research, 70, 245-317.
21. Charbuty, B., & Abdulazeez, A. (2021). Classification based on decision tree algorithm for machine learning. Journal of Applied Science and Technology Trends, 2(01), 20-28.
22. Chen, Y., & Shi, C. (2023). Network revenue management with online inverse batch gradient descent method. Production and Operations Management.
23. Chicco, D., & Jurman, G. (2020). Machine learning can predict survival of patients with heart failure from serum creatinine and ejection fraction alone. BMC medical informatics and decision making, 20(1), 1-16.

24. Chollet, F. (2021). Deep learning with Python. Simon and Schuster.
25. Christodoulou, E., Ma, J., Collins, G. S., Steyerberg, E. W., Verbakel, J. Y., & Van Calster, B. (2019). A systematic review shows no performance benefit of machine learning over logistic regression for clinical prediction models. Journal of clinical epidemiology, 110, 12-22.
26. Cunningham, P., & Delany, S. J. (2021). Underestimation bias and underfitting in machine learning. In Trustworthy AI-Integrating Learning, Optimization and Reasoning: First International Workshop, TAILOR 2020, Virtual Event, September 4–5, 2020, Revised Selected Papers 1 (pp. 20-31). Springer International Publishing.
27. Dahouda, M. K., & Joe, I. (2021). A deep-learned embedding technique for categorical features encoding. IEEE Access, 9, 114381-114391.
28. Dalianis, H., & Dalianis, H. (2018). Evaluation metrics and evaluation. Clinical Text Mining: secondary use of electronic patient records, 45-53.
29. Dhanabal, S., & Chandramathi, S. J. I. J. C. A. (2011). A review of various k-nearest neighbor query processing techniques. International Journal of Computer Applications, 31(7), 14-22.
30. Domingos, P. (2000, June). A unified bias-variance decomposition. In Proceedings of 17th international conference on machine learning (pp. 231-238). Morgan Kaufmann Stanford.
31. Dunn, W. B., Broadhurst, D. I., Deepak, S. M., Buch, M. H., McDowell, G., Spasic, I., & Neyses, L. (2007). Serum metabolomics reveals many novel metabolic markers of heart failure, including pseudouridine and 2-oxoglutarate. Metabolomics, 3, 413-426.
32. El Naqa, I., & Murphy, M. J. (2015). What is machine learning? (pp. 3-11). Springer International Publishing.
33. Fahle, S., Prinz, C., & Kuhlenkötter, B. (2020). Systematic review on machine learning (ML) methods for manufacturing processes–Identifying artificial intelligence (AI) methods for field application. Procedia CIRP, 93, 413-418.
34. Famili, A., Shen, W. M., Weber, R., & Simoudis, E. (1997). Data preprocessing and intelligent data analysis. Intelligent data analysis, 1(1), 3-23.
35. Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). From data mining to knowledge discovery in databases. AI magazine, 17(3), 37-37.
36. Fernandes, K., Chicco, D., Cardoso, J. S., & Fernandes, J. (2018). Supervised

deep learning embeddings for the prediction of cervical cancer diagnosis. PeerJ Computer Science, 4, e154.

37. Fernández, A., Garcia, S., Herrera, F., & Chawla, N. V. (2018). SMOTE for learning from imbalanced data: progress and challenges, marking the 15-year anniversary. Journal of artificial intelligence research, 61, 863-905.

38. Forman, G. (2003). An extensive empirical study of feature selection metrics for text classification. J. Mach. Learn. Res., 3(Mar), 1289-1305.

39. François-Lavet, V., Henderson, P., Islam, R., Bellemare, M. G., & Pineau, J. (2018). An introduction to deep reinforcement learning. Foundations and Trends® in Machine Learning, 11(3-4), 219-354.

40. Gallagher, J., McCormack, D., Zhou, S., Ryan, F., Watson, C., McDonald, K., & Ledwidge, M. T. (2019). A systematic review of clinical prediction rules for the diagnosis of chronic heart failure. ESC heart failure, 6(3), 499-508.

41. García, S., Luengo, J., & Herrera, F. (2015). Data preprocessing in data mining (Vol. 72, pp. 59-139). Cham, Switzerland: Springer International Publishing.

42. García, S., Luengo, J., & Herrera, F. (2016). Tutorial on practical tips of the most influential data preprocessing algorithms in data mining. Knowledge-Based Systems, 98, 1-29.

43. García, V., Sánchez, J. S., & Mollineda, R. A. (2012). On the effectiveness of preprocessing methods when dealing with different levels of class imbalance. Knowledge-Based Systems, 25(1), 13-21.

44. Ghosh, S., Dasgupta, A., & Swetapadma, A. (2019). A study on support vector machine based linear and non-linear pattern classification. In 2019 International Conference on Intelligent Sustainable Systems (ICISS) (pp. 24-28). IEEE.

45. Gu, Q., & Han, J. (2013, April). Clustered support vector machines. In Artificial intelligence and statistics (pp. 307-315). PMLR.

46. Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. Journal of machine learning research, 3(Mar), 1157-1182.

47. Hady, M. F. A., & Schwenker, F. (2013). Semi-supervised learning. Handbook on Neural Information Processing, 215-239.

48. Hancock, J. T., & Khoshgoftaar, T. M. (2020). Survey on categorical data for neural networks. Journal of Big Data, 7(1), 1-41.

49. Heiney, S. P., Donevant, S. B., Adams, S. A., Parker, P. D., Chen, H., & Levkoff, S. (2020). A smartphone app for self-management of heart failure in older African

Americans: feasibility and usability study. JMIR aging, 3(1), e17142.

50. Huang, S., Cai, N., Pacheco, P. P., Narrandes, S., Wang, Y., & Xu, W. (2018). Applications of support vector machine (SVM) learning in cancer genomics. Cancer genomics & proteomics, 15(1), 41-51.

51. Huang, X., Wu, L., & Ye, Y. (2019). A review on dimensionality reduction techniques. International Journal of Pattern Recognition and Artificial Intelligence, 33(10), 1950017.

52. Hussein, A. S., Li, T., Yohannese, C. W., & Bashir, K. (2019). A-SMOTE: A new preprocessing approach for highly imbalanced datasets by improving SMOTE. International Journal of Computational Intelligence Systems, 12(2), 1412-1422.

53. Jabbar, H., & Khan, R. Z. (2015). Methods to avoid over-fitting and under-fitting in supervised machine learning (comparative study). Computer Science, Communication and Instrumentation Devices, 70(10.3850), 978-981.

54. Jakkula, V. (2006). Tutorial on support vector machine (svm). School of EECS, Washington State University, 37(2.5), 3.

55. Jansen, B. J., Aldous, K. K., Salminen, J., Almerekhi, H., & Jung, S. G. (2023). Data Preprocessing. In Understanding Audiences, Customers, and Users via Analytics: An Introduction to the Employment of Web, Social, and Other Types of Digital People Data (pp. 65-75). Cham: Springer Nature Switzerland.

56. Jiang, T., Gradus, J. L., & Rosellini, A. J. (2020). Supervised machine learning: a brief primer. Behavior Therapy, 51(5), 675-687.

57. Kingsford, C., & Salzberg, S. L. (2008). What are decision trees?. Nature biotechnology, 26(9), 1011-1013.

58. Kotsiantis, S. B. (2013). Decision trees: a recent overview. Artificial Intelligence Review, 39, 261-283.

59. Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. Emerging artificial intelligence applications in computer engineering, 160(1), 3-24.

60. Krishna, G. S., Supriya, K., & Rao, K. M. (2022, September). Selection of data preprocessing techniques and its emergence towards machine learning algorithms using hpi dataset. In 2022 IEEE Global Conference on Computing, Power and Communication Technologies (GlobConPT) (pp. 1-6). IEEE.

61. Kueffner, R., Zach, N., Bronfeld, M., Norel, R., Atassi, N., Balagurusamy, V.,

& Stolovitzky, G. (2019). Stratification of amyotrophic lateral sclerosis patients: a crowdsourcing approach. Scientific reports, 9(1), 690.

62. Le, Q. V. (2013, May). Building high-level features using large scale unsupervised learning. In 2013 IEEE international conference on acoustics, speech and signal processing (pp. 8595-8598). IEEE.

63. Li, N., Shepperd, M., & Guo, Y. (2020). A systematic review of unsupervised learning techniques for software defect prediction. Information and Software Technology, 122, 106287.

64. Li, Y. (2017). Deep reinforcement learning: An overview. arXiv preprint arXiv:1701.07274.

65. Maggio, V., Chierici, M., Jurman, G., & Furlanello, C. (2018). Distillation of the clinical algorithm improves prognosis by multi-task deep learning in high-risk neuroblastoma. PloS one, 13(12), e0208924.

66. Mahesh, B. (2020). Machine learning algorithms-a review. International Journal of Science and Research (IJSR), 9(1), 381-386.

67. Maimon, O. Z., & Rokach, L. (2014). Data mining with decision trees: theory and applications (Vol. 81). World scientific.

68. Maimon, O., & Rokach, L. (2005). Introduction to knowledge discovery in databases. In Data mining and knowledge discovery handbook (pp. 1-17). Boston, MA: Springer US.

69. Masino, A. J., Harris, M. C., Forsyth, D., Ostapenko, S., Srinivasan, L., Bonafide, C. P., & Grundmeier, R. W. (2019). Machine learning models for early sepsis recognition in the neonatal intensive care unit using readily available electronic health record data. PloS one, 14(2), e0212665.

70. McClellan, W. M., Flanders, W. D., Langston, R. D., Jurkovitz, C., & Presley, R. (2002). Anemia and renal insufficiency are independent risk factors for death among patients with congestive heart failure admitted to community hospitals: a population-based study. Journal of the American Society of Nephrology, 13(7), 1928-1936.

71. Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A survey on bias and fairness in machine learning. ACM computing surveys (CSUR), 54(6), 1-35.

72. Meyer, D., & Wien, F. T. (2001). Support vector machines. R News, 1(3), 23-26.

73. Montgomery, D. C., Peck, E. A., & Vining, G. G. (2021). Introduction to linear regression analysis. John Wiley & Sons.

74. Muhammad, I., & Yan, Z. (2015). Supervised Machine Learning approaches: A survey. ICTACT Journal on Soft Computing, 5(3).

75. Müller, A. C., & Guido, S. (2016). Introduction to machine learning with Python: a guide for data scientists. "O'Reilly Media, Inc.".

76. Narkhede, S. (2018). Understanding auc-roc curve. Towards Data Science, 26(1), 220-227.

77. Nayak, S., & Sharma, Y. K. (2023). A modified Bayesian boosting algorithm with weight-guided optimal feature selection for sentiment analysis. Decision Analytics Journal, 8, 100289.

78. Nnamoko, N., & Korkontzelos, I. (2020). Efficient treatment of outliers and class imbalance for diabetes prediction. Artificial intelligence in medicine, 104, 101815.

79. Osisanwo, F. Y., Akinsola, J. E. T., Awodele, O., Hinmikaiye, J. O., Olakanmi, O., & Akinjobi, J. (2017). Supervised machine learning algorithms: classification and comparison. International Journal of Computer Trends and Technology (IJCTT), 48(3), 128-138.

80. Özbay Karakuş, M., & Er, O. (2022). A comparative study on prediction of survival event of heart failure patients using machine learning algorithms. Neural Computing and Applications, 34(16), 13895-13908.

81. Patrício, M., Pereira, J., Crisóstomo, J., Matafome, P., Gomes, M., Seiça, R., & Caramelo, F. (2018). Using Resistin, glucose, age and BMI to predict the presence of breast cancer. BMC cancer, 18(1), 1-8.

82. Perlaza, S. M., Esnaola, I., Bisson, G., & Poor, H. V. (2023). On the validation of Gibbs algorithms: Training datasets, test datasets and their aggregation. arXiv preprint arXiv:2306.12380.

83. Podgorelec, V., Kokol, P., Stiglic, B., & Rozman, I. (2002). Decision trees: an overview and their use in medicine. Journal of medical systems, 26, 445-463.

84. Pradipta, G. A., Wardoyo, R., Musdholifah, A., Sanjaya, I. N. H., & Ismail, M. (2021, November). SMOTE for handling imbalanced data problem: A review. In 2021 Sixth International Conference on Informatics and Computing (ICIC) (pp. 1-8). IEEE.

85. Priyam, A., Abhijeeta, G. R., Rathee, A., & Srivastava, S. (2013). Comparative

analysis of decision tree classification algorithms. International Journal of current engineering and technology, 3(2), 334-337.

86. Raja, P. S., & Thangavel, K. J. S. C. (2020). Missing value imputation using unsupervised machine learning techniques. Soft Computing, 24(6), 4361-4392.

87. Ranganathan, P., Pramesh, C. S., & Aggarwal, R. (2017). Common pitfalls in statistical analysis: logistic regression. Perspectives in clinical research, 8(3), 148.

88. Raschka, S. (2018). Model evaluation, model selection, and algorithm selection in machine learning. arXiv preprint arXiv:1811.12808.

89. Rjoob, K., Bond, R., Finlay, D., McGilligan, V., Leslie, S. J., Rababah, A., ... & Macfarlane, P. W. (2022). Machine learning and the electrocardiogram over two decades: Time series and meta-analysis of the algorithms, evaluation metrics and applications. Artificial Intelligence in Medicine, 102381.

90. Roelofs, R., Shankar, V., Recht, B., Fridovich-Keil, S., Hardt, M., Miller, J., & Schmidt, L. (2019). A meta-analysis of overfitting in machine learning. Advances in Neural Information Processing Systems, 32.

91. Ruder, S. (2016). An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747.

92. Scholkopf, B., & Smola, A. J. (2018). Learning with kernels: support vector machines, regularization, optimization, and beyond. MIT press.

93. Schrider, D. R., & Kern, A. D. (2018). Supervised machine learning for population genetics: a new paradigm. Trends in Genetics, 34(4), 301-312.

94. Seber, G. A., & Lee, A. J. (2003). Linear regression analysis (Vol. 330). John Wiley & Sons.

95. Sessa, J., & Syed, D. (2016, December). Techniques to deal with missing data. In 2016 5th international conference on electronic devices, systems and applications (ICEDSA) (pp. 1-4). IEEE.

96. Shilaskar, S., & Ghatol, A. (2013). Feature selection for medical diagnosis: Evaluation for cardiovascular diseases. Expert Systems with Applications, 40(10), 4146-4153.

97. Speiser, J. L., Miller, M. E., Tooze, J., & Ip, E. (2019). A comparison of random forest variable selection methods for classification prediction modeling. Expert systems with applications, 134, 93-101.

98. Stone, M. (1978). Cross-validation: A review. Statistics: A Journal of

Theoretical and Applied Statistics, 9(1), 127-139.

99. Sutton, R. S., & Barto, A. G. (2018). Reinforcement learning: An introduction. MIT press.

100. Syed, A. R. (2011). A review of cross validation and adaptive model selection.

101. Takano, N., & Alaghband, G. (2019). Srgan: Training dataset matters. arXiv preprint arXiv:1903.09922.

102. Tharwat, A., Mahdi, H., Elhoseny, M., & Hassanien, A. E. (2018). Recognizing human activity in mobile crowdsensing environment using optimized k-NN algorithm. Expert Systems with Applications, 107, 32-44.

103. Valentini, G., & Dietterich, T. G. (2004). Bias-variance analysis of support vector machines for the development of SVM-based ensemble methods. Journal of Machine Learning Research, 5(Jul), 725-775.

104. Vijayarani, S., Ilamathi, M. J., & Nithya, M. (2015). Preprocessing techniques for text mining-an overview. International Journal of Computer Science & Communication Networks, 5(1), 7-16.

105. Wang, J. X., Kurth-Nelson, Z., Tirumala, D., Soyer, H., Leibo, J. Z., Munos, R. & Botvinick, M. (2016). Learning to reinforcement learn. arXiv preprint arXiv:1611.05763.

106. Wang, J., & Biljecki, F. (2022). Unsupervised machine learning in urban studies: A systematic review of applications. Cities, 129, 103925.

107. Wang, Q. Q., Yu, S. C., Qi, X., Hu, Y. H., Zheng, W. J., Shi, J. X., & Yao, H. Y. (2019). Overview of logistic regression model analysis and application. Zhonghua yu fang yi xue za zhi [Chinese journal of preventive medicine], 53(9), 955-960.

108. Weng, S. F., Reps, J., Kai, J., Garibaldi, J. M., & Qureshi, N. (2017). Can machine-learning improve cardiovascular risk prediction using routine clinical data?. PloS one, 12(4), e0174944.

109. Wiens, J., & Shenoy, E. S. (2018). Machine learning for healthcare: on the verge of a major shift in healthcare epidemiology. Clinical Infectious Diseases, 66(1), 149-153.

110. Wiering, M. A., & Van Otterlo, M. (2012). Reinforcement learning. Adaptation, learning, and optimization, 12(3), 729.

111. Williamson, D. F., Parker, R. A., & Kendrick, J. S. (1989). The box plot: a simple visual method to interpret data. Annals of internal medicine, 110(11),

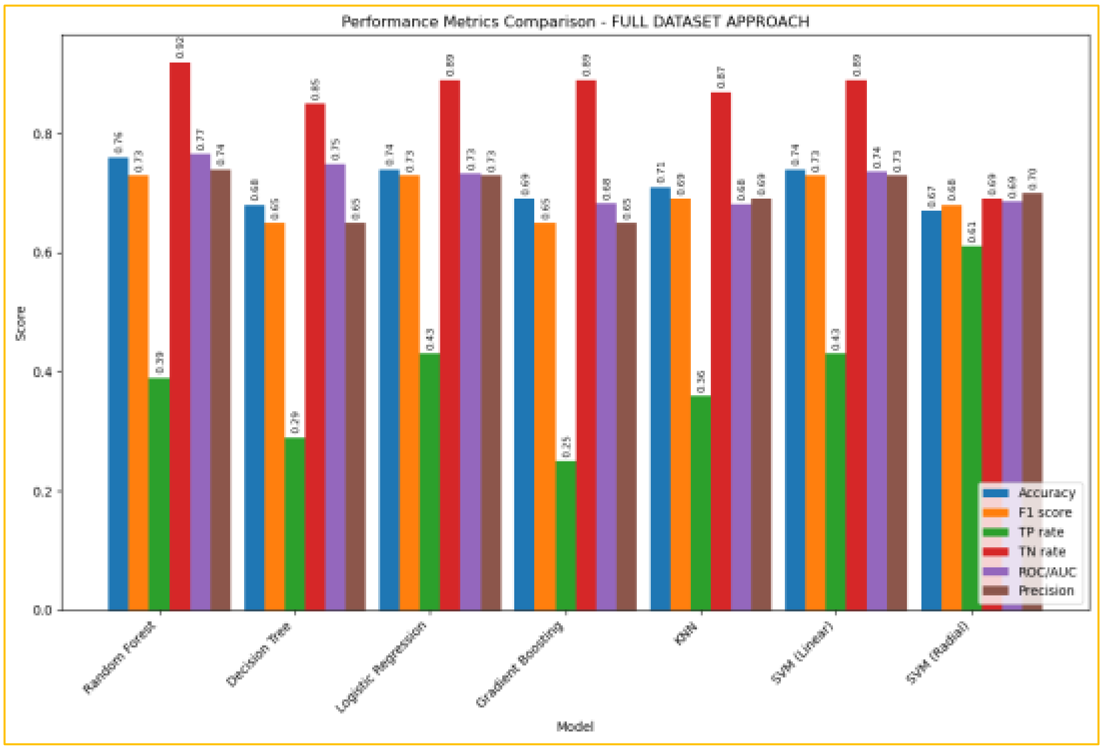
916-921.

112. Wu, Z., Wu, Y., He, X., Wan, L., Tsai, L. C., & Chen, A. (2023, May). Application of Machine Learning for Heart Failure Prediction. In 2023 6th International Conference on Artificial Intelligence and Big Data (ICAIBD) (pp. 276-282). IEEE.
113. Ying, X. (2019, February). An overview of overfitting and its solutions. In Journal of physics: Conference series (Vol. 1168, p. 022022). IOP Publishing.
114. Yunus, I., Fasih, A., & Wang, Y. (2018). The use of procalcitonin in the determination of severity of sepsis, patient outcomes and infection characteristics. PloS one, 13(11), e0206527.
115. Zhang, H., Zhang, L., & Jiang, Y. (2019, October). Overfitting and underfitting analysis for deep learning based end-to-end communication systems. In 2019 11th international conference on wireless communications and signal processing (WCSP) (pp. 1-6). IEEE.
116. Zhang, S., Li, X., Zong, M., Zhu, X., & Cheng, D. (2017). Learning k for knn classification. ACM Transactions on Intelligent Systems and Technology (TIST), 8(3), 1-19.
117. Zhou, J., Gandomi, A. H., Chen, F., & Holzinger, A. (2021). Evaluating the quality of machine learning explanations: A survey on methods and metrics. Electronics, 10(5), 593.
118. Zhou, Z. H., & Zhou, Z. H. (2021). Semi-supervised learning. Machine Learning, 315-341.
119. Zhu, X. J. (2005). Semi-supervised learning literature survey.

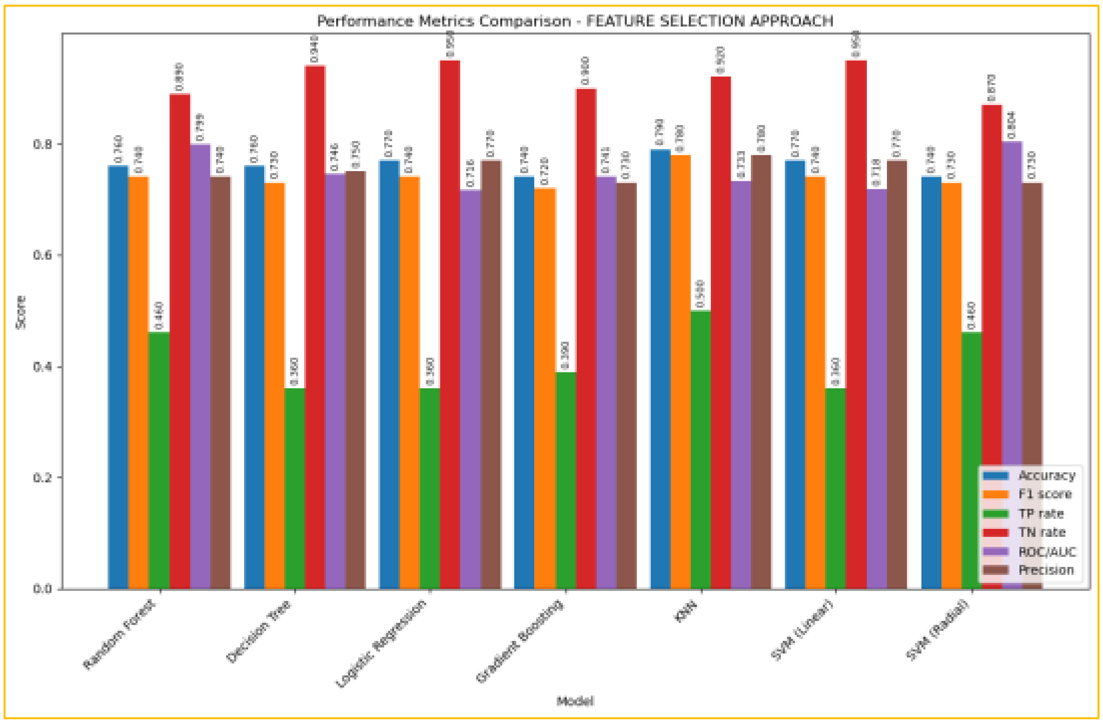
APPENDIX

In the Appendix, all cumulative charts (graphs) of the results are provided.

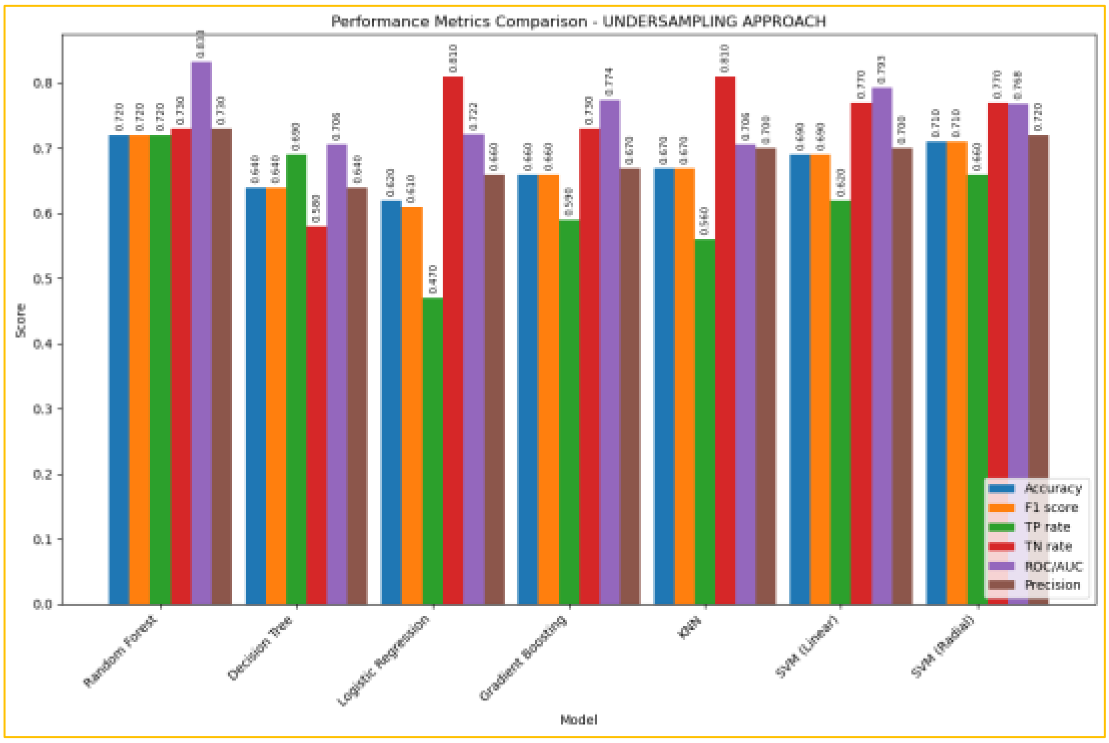
Graph 1. Performance metrics comparison for Full dataset approach.



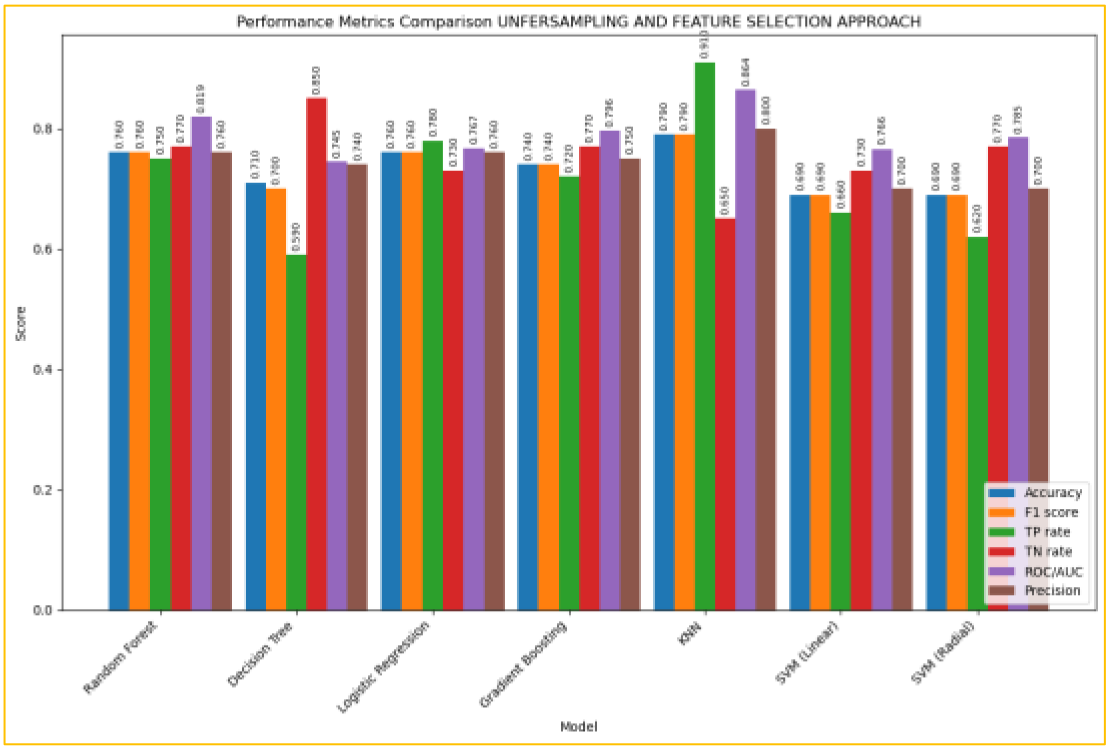
Graph 2. Performance metrics comparison for Feature selection approach.



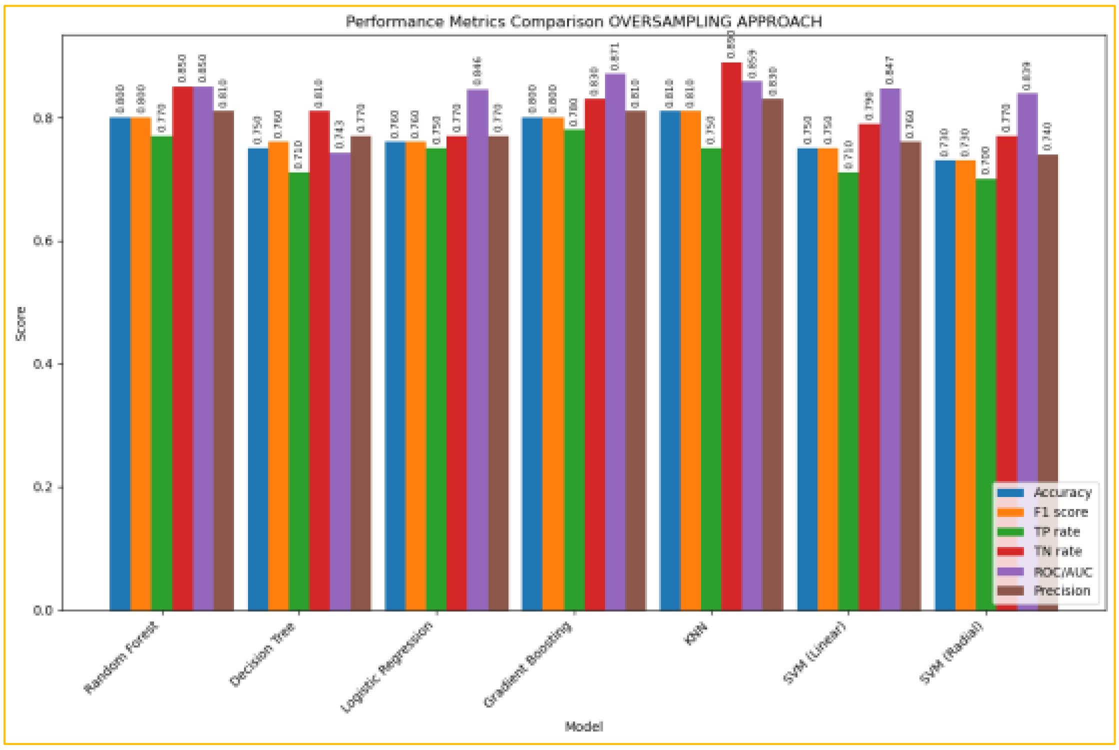
Graph 3. Performance metrics comparison for Undersampling approach.



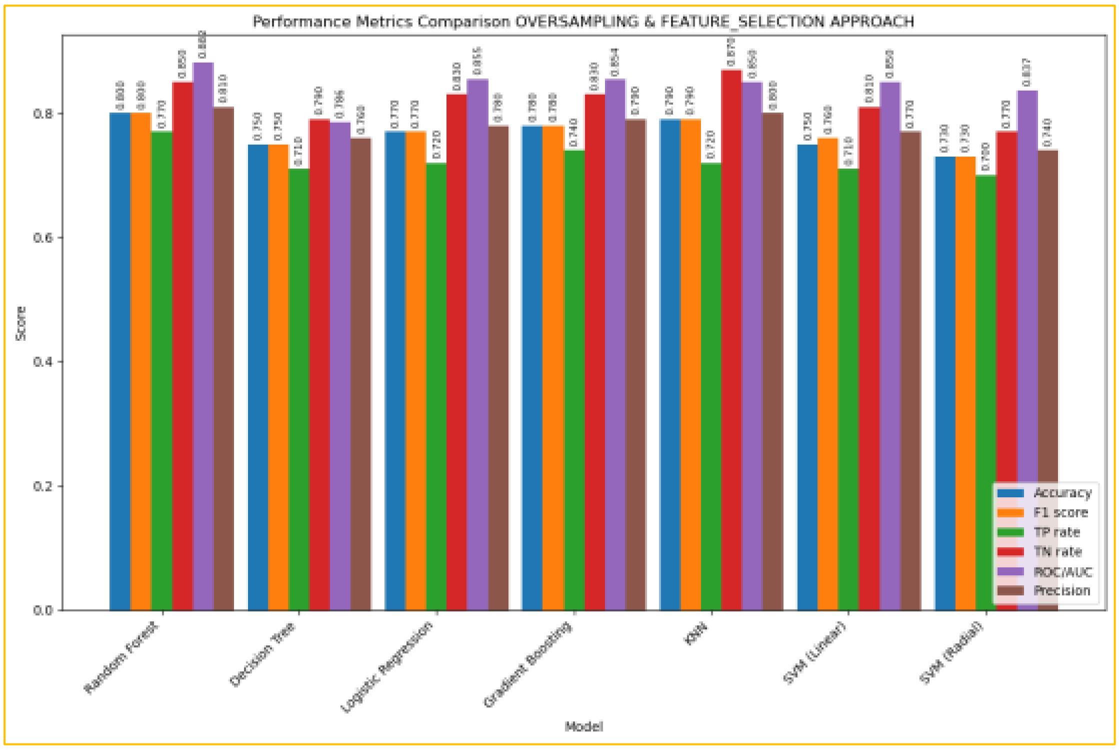
Graph 4. Performance metrics comparison for Undersampling approach and Feature selection approach.



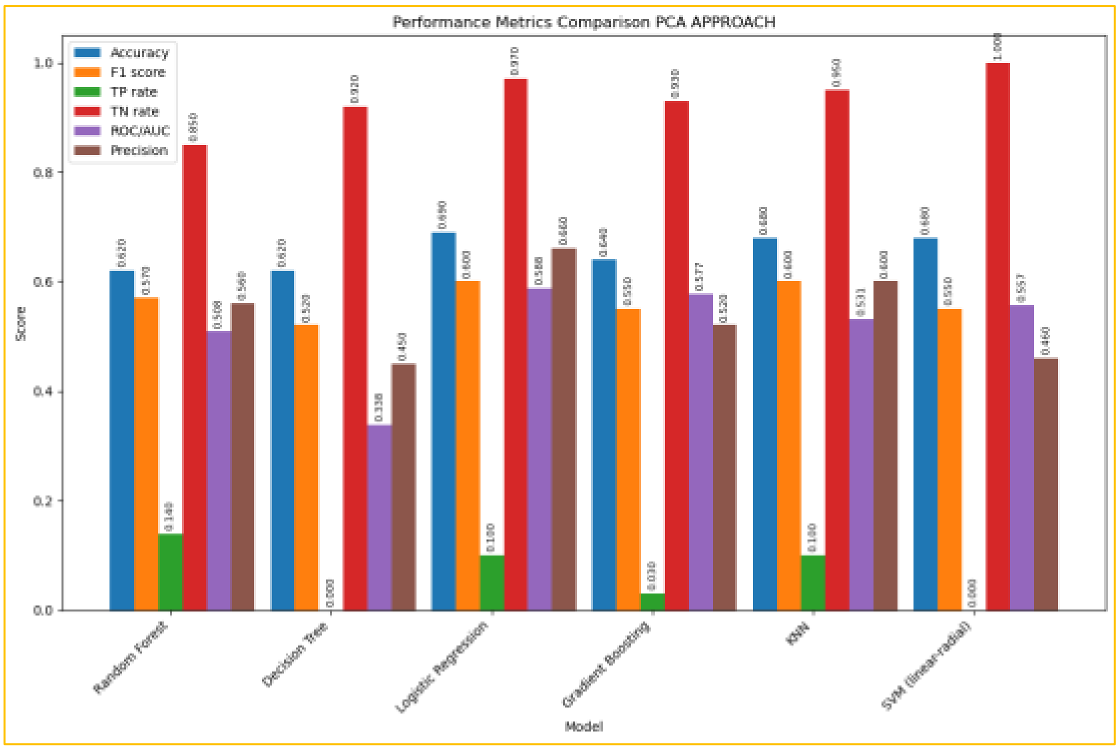
Graph 5. Performance metrics comparison for Oversampling approach.



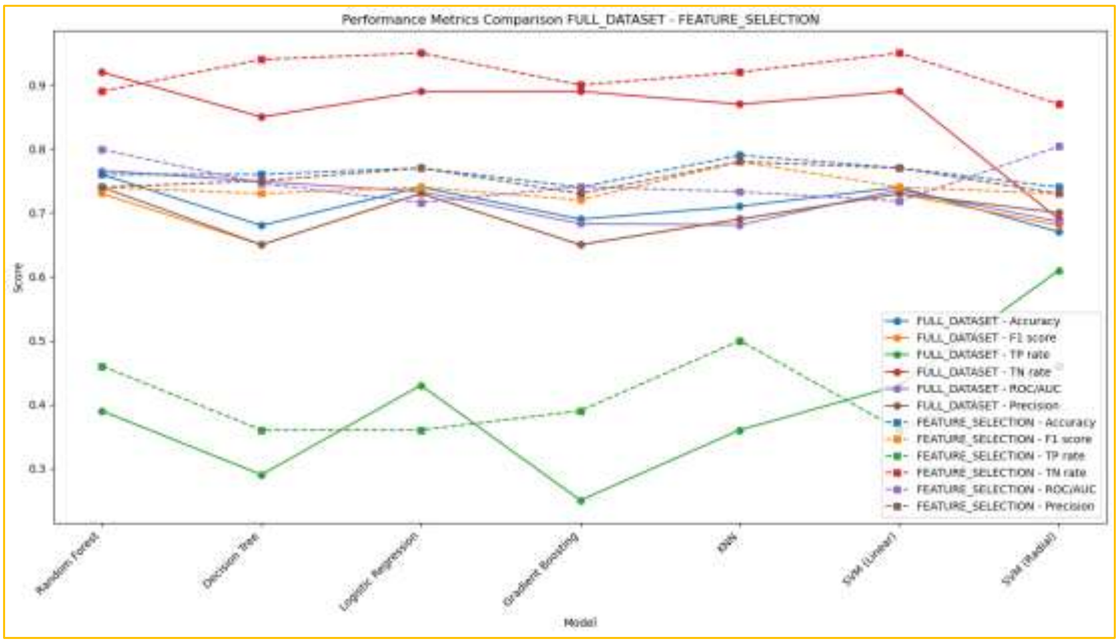
Graph 6. Performance metrics comparison for Oversampling approach and Feature selection approach.



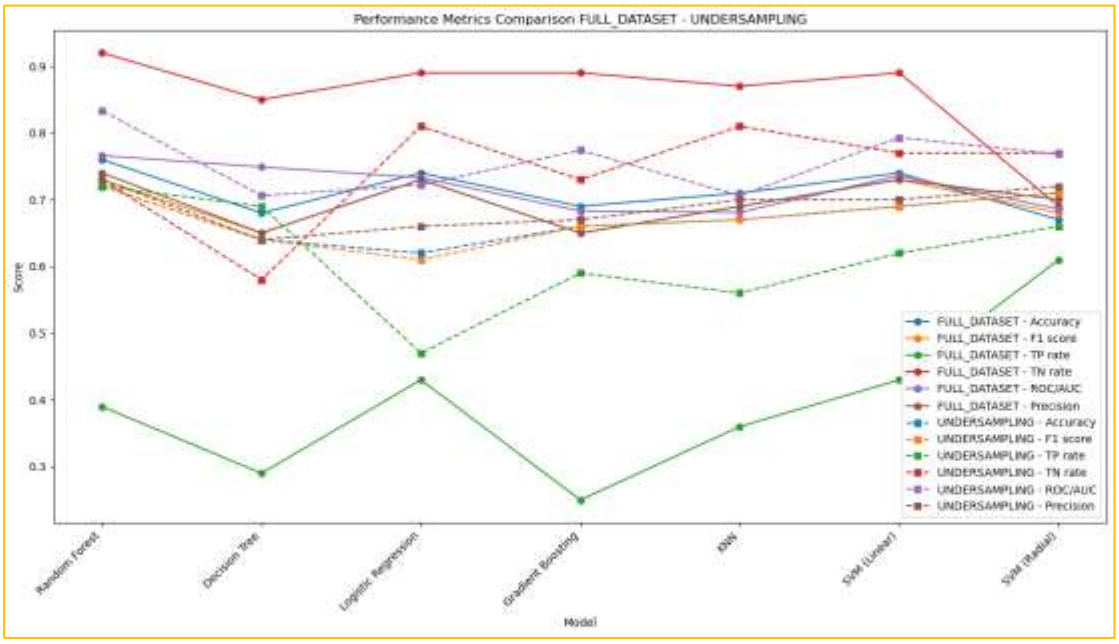
Graph 7. Performance metrics comparison for Principal Component Analysis (PCA) approach.



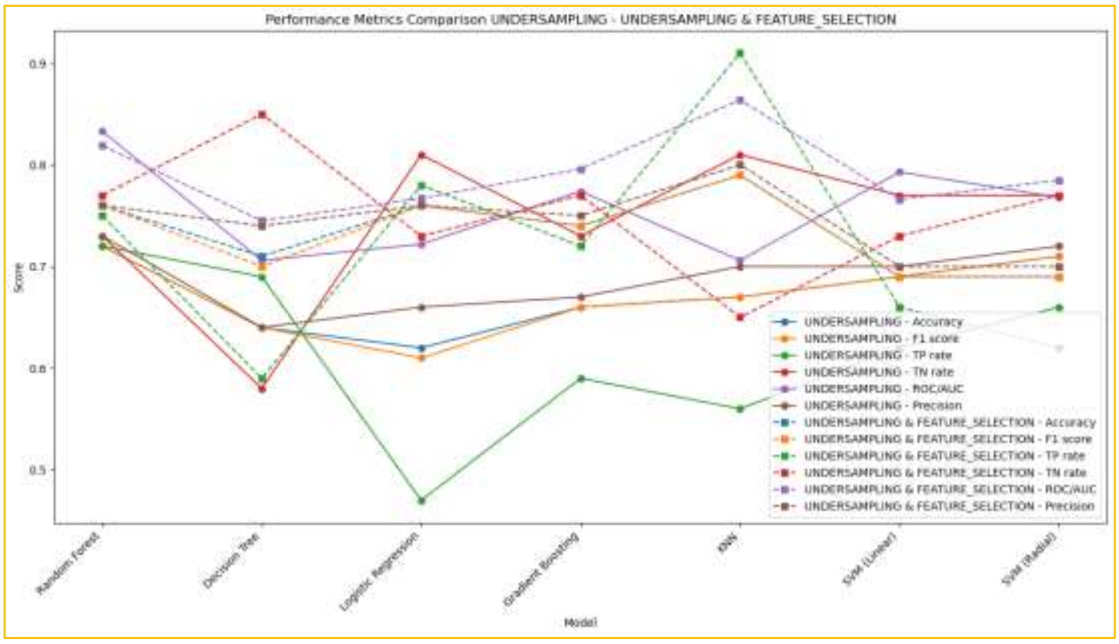
Graph 8. Performance metrics comparison between Full dataset and Feature selection approaches.



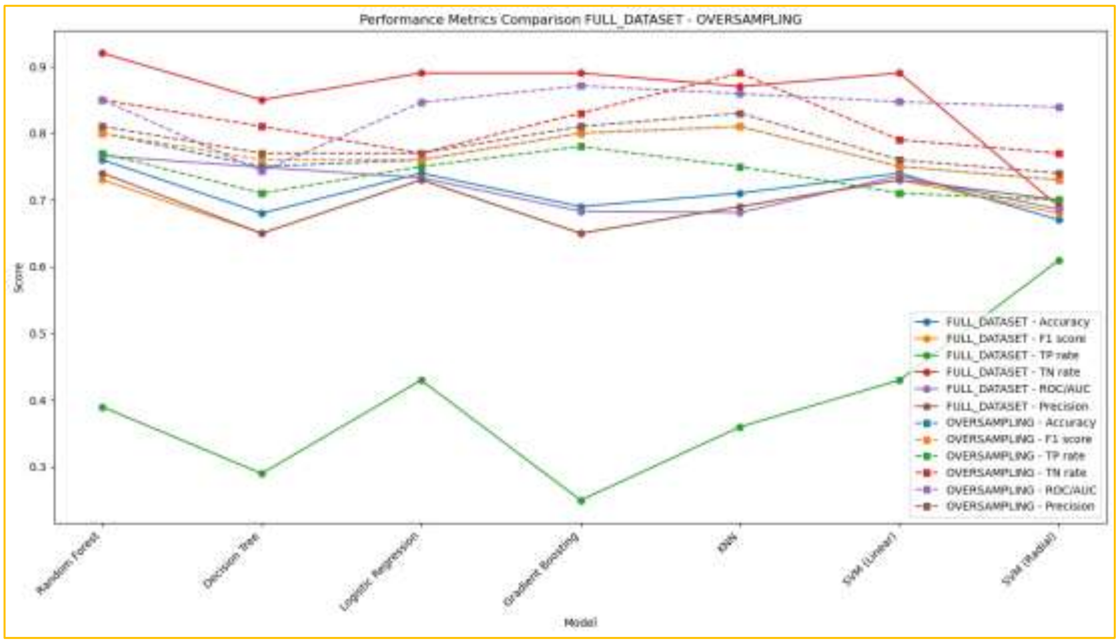
Graph 9. Performance metrics comparison between Full dataset and Undersampling approaches.



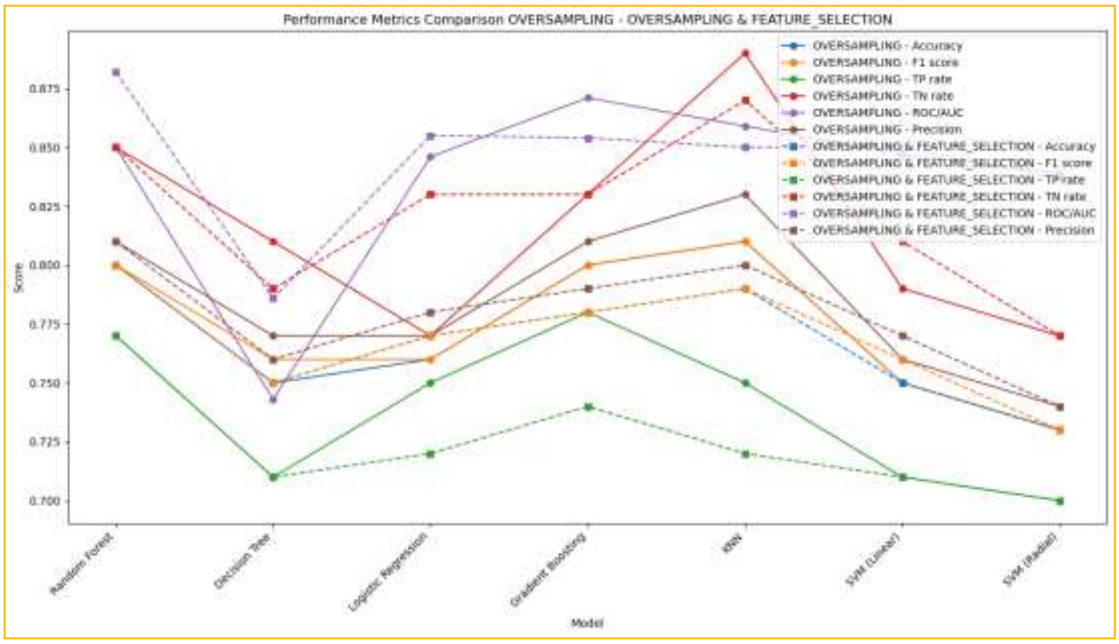
Graph 10. Performance metrics comparison between Undersampling and Undersampling with Feature selection approaches.



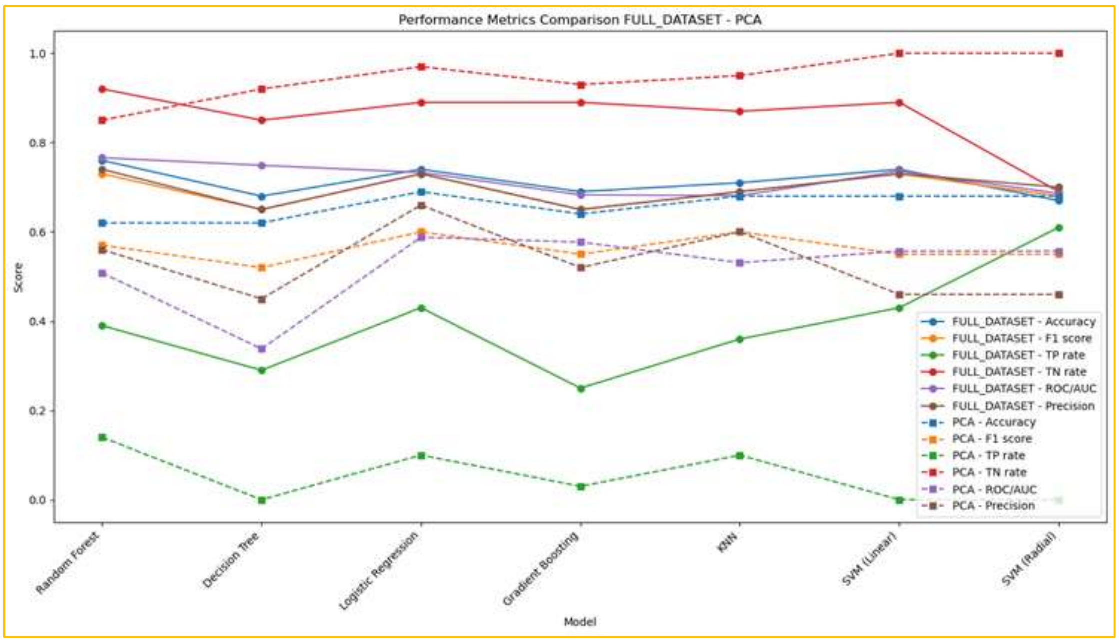
Graph 11. Performance metrics comparison between Full dataset and Oversampling approaches.



Graph 12. Performance metrics comparison between Oversampling and Oversampling with Feature selection approaches.



Graph 13. Performance metrics comparison between Full dataset and Principal Component Analysis (PCA) approaches.


Performance Metrics Comparison FULL_DATASET - PCA
Score
Random Forest
Decision Tree
Logistic Regression
Gradient Boosting
KNN
SVM (Linear)
SVM (Radial)
Model
FULL_DATASET - Accuracy
FULL_DATASET - F1 score
FULL_DATASET - TP rate
FULL_DATASET - TN rate
FULL_DATASET - ROC/AUC
FULL_DATASET - Precision
PCA - Accuracy
PCA - F1 score
PCA - TP rate
PCA - TN rate
PCA - ROC/AUC
PCA - Precision