Breast Cancer Prediction Using Machine Learning Algorithms

by

Zeeshan Ali Shahzad

University of Engineering and Technology

B.Sc. Electrical and Computer Engineering, 2012

A Report Submitted in Partial Fulfilment of the Requirements for the Degree of

Master of Engineering

In the Department of Electrical and Computer Engineering

Breast Cancer Prediction Using Machine Learning Algorithms

by

Zeeshan Ali Shahzad

University of Victoria

Supervisory Committee

Dr. T. Aaron Gulliver, Supervisor
Department of Electrical and Computer Engineering

Dr. Mihai Sima, Departmental Member
Department of Electrical and Computer Engineering

Abstract

Breast cancer has become a pressing global health issue with its prevalence increasing worldwide. The rise in breast cancer cases is a cause for concern as it not only affects the physical and emotional well-being of individuals but also places a significant burden on the healthcare system. Early detection and timely intervention are critical factors in effectively combatting this disease. The ability to predict and diagnose breast cancer at its earliest stages can have a profound difference in patient outcomes, potentially saving countless lives. In recent years, the importance of Machine Learning (ML) in the field of healthcare has become paramount. This study considers the utility of supervised ML models to address the challenges posed by breast cancer using the publicly available Breast Cancer Wisconsin (Diagnostic) dataset from the University of California Irvine (UCI) ML repository. The Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Naive Bayes and K-Nearest Neighbors (KNN) classifiers are implemented using Jupyter Notebook with Python programming.

The goal of the proposed methodology is accurate breast cancer prediction. First, data preprocessing is employed to clean the dataset by removing null values and duplicates, and handling missing data. In order to balance the target labels of the dataset, Synthetic Minority Oversampling Technique (SMOTE) is employed. Then, Principal Component Analysis (PCA) is used to reduce the dimensions of the dataset. The number of components is varied (n=2, 5, 10, 15). For training and testing the ML models, five data splits, namely 80/20, 70/30, 50/50, 30/70, and 20/80 are employed to assess the impact on model performance.

The performance of the models is evaluated using the metrics accuracy, precision, recall, F1-score, and execution time. The results obtained show that SVM and Logistic Regression outperform the other models with SVM having an accuracy of 98.2% and an execution time of 9.99 ms with an 80/20 split using 10 features and Logistic Regression having an accuracy of 97.9% and an execution time of 8.42 ms with a 50/50 split using 15 features.

Table of Contents

List of Figures

List of Tables

Glossary

AI….....Artificial Intelligence

ML….....Machine Learning

RL….....Reinforcement Learning

LR….....Logistic Regression

DT….....Decision Tree

RF….....Random Forest

SVM….....Support Vector Machine

NB….....Naive Bayes

KNN….....K-Nearest Neighbors

PCA….....Principal Component Analysis

SMOTE….....Synthetic Minority Oversampling Technique

ms….....Milliseconds

WHO….....World Health Organization

TP….....True Positive

TN….....True Negative

FP….....False Positive

FN….....False Negative

Acknowledgment

I would like to express my deepest gratitude to the Almighty for His countless blessings and consistent guidance throughout this journey. His grace has been my constant source of strength and inspiration.

Heartfelt appreciation goes to my parents and family, whose boundless love, steadfast support, and encouragement have been the pillars of my success. In particular, I owe an immeasurable debt of gratitude to my mother, whose courage and unwavering support have been a guiding light in my life. Her sacrifices and resilience have been my motivation to persevere through challenges.

I am profoundly thankful to Professor Dr. T. Aaron Gulliver for accepting me into his research group. His mentorship, insightful guidance, and resolute support have played a pivotal role in shaping my understanding and approach to the complexities of this project. Without his direction, this journey would not have been possible.

Special thanks to my dear friend Mohammad Khizar Ansari for his unwavering support all along the way. His encouragement and camaraderie have added a valuable dimension to this endeavour.

I extend my sincere appreciation to my younger sister for her valuable contributions and support throughout this endeavor.

I also want to thank my wife for her support during this journey.

This project has been a collective effort, and I am grateful to everyone who has been a part of this transformative experience.

Dedication

This project is dedicated with deep appreciation to my parents, whose unwavering support and limitless love have been the bedrock of my journey. Specifically, my mother, whose courage, sacrifices, and unwavering encouragement have illuminated my path. Her resilience and resolute belief in my abilities have been the driving force behind every stride I have taken.

To my entire family, your collective support has fortified my resolve. Your understanding, patience, and encouragement have cultivated a nurturing environment, enabling me to pursue my academic aspirations.

Chapter 1 Introduction

Breast cancer is the most prevalent and prominent form of cancer among women, and its incidence has seen a concerning rise in recent years. It now accounts for a significant share, approximately 24.5% of all cancer cases and 15.5% of cancer deaths. This underscores the urgent need for heightened awareness, early detection, and research initiatives to combat the growing impact of breast cancer on the health and well-being of women [1].

Breast cancer originates in the breast tissue, where it begins as abnormal replication of cells [2]. This uncontrolled cell growth results in the formation of lumps within the breast and surrounding areas. These lumps, known as tumors, come in two distinct categories: benign and malignant. Benign tumors are non-cancerous and generally pose no immediate threat to health. In contrast, malignant tumors are cancerous and carry a high degree of mortality. Malignant tumors have the potential to invade and spread to neighbouring tissue and organs, making early detection and intervention imperative. The aggressive spread of cancerous cells to other parts of the body is referred to as metastasis [3]. As the disease progresses, it can invade internal organs such as the liver and lungs [4].

The consequences of metastatic breast cancer are far-reaching, as it not only inflicts physical harm but also takes a toll on the mental health of patients. Therefore, understanding the mechanisms underlying breast cancer, as well as developing effective diagnostic and treatment strategies, is important in combating this life-threatening disease [5]. Timely detection, often through routine mammograms and clinical examinations, can significantly improve the chances of successful treatment and enhance the quality of life for those affected. Furthermore, ongoing research into new therapies and targeted treatments offer hope for more effective management of breast cancer [6].

Detecting breast cancer in its early stages is vital for effective treatment. One of the primary indicators is the formation of lumps, particularly in the armpits. Swelling of the breast, dimpling of the skin covering the breast area, and the presence of redness on the skin around the breast are also of concern [7].

Changes in the shape or size of the breast should not be overlooked. Additionally, a discharge of blood or any liquid other than milk from the nipple is a warning sign. Consulting a physician or oncologist for prompt screening is imperative if any of these symptoms persist, as early detection can significantly improve the chances of successful treatment and recovery [8].

1.1 Motivation

Breast cancer has emerged as one of the most prevalent and concerning health issues globally [9]. In the United States, an estimated 290,560 new cases of breast cancer were recorded, with 43,780 deaths attributed to this disease. Moreover, it has been estimated that one out of every 39 women is at risk of breast cancer which underscores the urgency of addressing this healthcare crisis.

Within the realm of healthcare, artificial intelligence, and specifically Machine Learning (ML) algorithms plays a pivotal role in aiding healthcare professionals in the analysis and interpretation of medical data. The goal of this project is to employ ML algorithms to predict breast cancer. This will provide healthcare practitioners with a valuable tool for identifying and addressing breast cancer promptly [10].

1.2 Objectives

The questions considered in this work are as follows.

1. How can real-world clinical data such as that from the Breast Cancer Wisconsin (Diagnostic) dataset be used effectively to address breast cancer challenges? How can data preprocessing, feature selection, and ML models improve the accuracy of breast cancer predictions?
2. How does balancing the dataset using the Synthetic Minority Oversampling Technique (SMOTE) enhance the training and evaluation of breast cancer prediction models?
3. Can Principal Component Analysis (PCA) reduce the dimensionality of breast cancer datasets for more efficient modelling?
4. What is the impact of different data splits on the performance of breast cancer prediction models, and how does this impact model selection?
5. How do the widely used ML algorithms Logistic Regression, Decision Tree, Random Forest, Support Vector Machine (SVM), Naive Bayes, and K-Nearest Neighbors (KNN),

perform in the context of breast cancer prediction, and how does their performance compare in terms of accuracy, precision, recall, F-score, and execution time?

1.3 Contributions

The contributions of this project are as follows.

1. Real-world clinical data from the Breast Cancer Wisconsin (Diagnostic) dataset was used to address breast cancer challenges.
2. A comprehensive methodology, including data preprocessing, feature selection, and ML algorithms was developed for breast cancer prediction.
3. Data preprocessing techniques such as null value and duplicate removal, and missing data handling, were implemented to enhance data quality.
4. The dataset was balanced using SMOTE to improve model training and evaluation.
5. PCA was used to identify highly correlated features and reduce dataset dimensionality.
6. Five data splits (80/20, 70/30, 50/50, 30/70, and 20/80) were considered to assess their impact on model performance.
7. The performance of six popular ML algorithms (Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN) was evaluated using the metrics accuracy, precision, recall, F1-score, and execution time.

1.4 Related Work

The field of medicine is undergoing continuous transformation with a gradual shift towards streamlined and modernized approaches. The progression towards automated systems has brought about a revolution in healthcare practices. As a result, healthcare providers have now the means to plan and deliver more efficient and effective treatments for improved patient care and outcomes [11].

The application of ML for breast cancer prediction was considered in [12] using the Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN algorithms. The effectiveness of these algorithms was evaluated in terms of accuracy, precision, recall, and F1-score. The Breast Cancer Wisconsin (Diagnostic) dataset was used in [13] to evaluate the performance of the ML algorithms Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN. The study assessed these algorithms in terms of accuracy, precision,

recall, F1-score, and execution time. The goal was to enhance early breast cancer detection methods to improve patient outcomes and advance healthcare practices. Breast cancer prediction was explored in [14] using a hybrid ML algorithm with a clinical dataset. The results obtained demonstrate the effectiveness of this approach in accurately identifying breast cancer to enhance early detection and diagnosis.

A non-invasive breast cancer classification system for the detection of metastases was presented in [15]. ML algorithms were evaluated using accuracy and area under Receiver Operating Characteristic (ROC) curve. Statistical significance was determined using the Welch unpaired t-test. A text mining framework for Electronic Medical Records (EMR) was used to segregate blood profile data and identify Metastatic Breast Cancer (MBC) patients. The removal of outliers significantly improved the accuracy of the ML models. The Decision Tree classifier achieved an accuracy of 83% with an AUC of 0.87. It was deployed through Flask to create a web application for robust MBC diagnosis.

The application of ML algorithms, including Random Forest, Gradient Boosting Trees (GBT), and Multi-layer Perceptron (MLP), was explored in [16] to predict the onset of breast cancer. The Random Forest algorithm was the best achieving an accuracy of 80%. These results illustrate the effectiveness of ML techniques in early breast cancer detection to enhance diagnostic capabilities and improve patient care.

1.5 Outline

This report is organized as follows.

Chapter 1: This chapter provided an introduction to breast cancer prediction using ML algorithms including the background, motivation and objectives, research questions, and contributions. The related work was also surveyed.

Chapter 2: This chapter introduces the ML models considered for the prediction of breast cancer. It also presents the Jupyter notebook tool which is used for implementing the breast cancer prediction system.

Chapter 3: This chapter presents the proposed prediction model for breast cancer. It also introduces the testing environment and classifiers used.

Chapter 4: This chapter gives the evaluation metrics and presents the results obtained. A discussion of the results is also provided.

Chapter 5: This chapter provides some concluding remarks and summarizes the results. It also gives some possible directions for future work.

Chapter 2 Machine Learning

ML, a subset of artificial intelligence, has revolutionized healthcare by enabling early disease prediction through data analysis. It is used to process clinical datasets and extract indicators and risk factors to assess disease susceptibility [17]. For breast cancer, ML can be used to analyse mammograms, genetics, and patient history to identify early-stage cases, facilitating timely intervention and improved outcomes. ML models adapt to new data, continually enhancing their prediction accuracy, offering a promising avenue for early diagnosis and more effective patient care [18].

ML algorithms can be divided into three types.

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

2.1 Supervised Learning

Supervised learning algorithms are trained on labelled data. They learn a mapping from input data to known output labels. This type of learning is akin to a teacher supervising the learning process of a student as the algorithm is guided to make predictions based on the provided correct answers [19]. Supervised learning is widely used for both classification and regression tasks. In classification, ML algorithms classify data into distinct categories or classes such as for spam detection or animal recognition. In contrast, regression models predict continuous numerical values like stock prices or housing prices. The accuracy of the predictions depends on the quality and quantity of the labelled training data [20].

2.1.1 Classification

Classification is a type of supervised learning where the goal is to categorize data into predefined classes or categories. This is akin to sorting objects into distinct bins based on their characteristics. For sam, in sentiment analysis, a classification algorithm may determine whether a given text expresses a positive, negative, or neutral sentiment. Classification problems can be binary (two classes) or multi-class (more than two classes). Logistic Regression, Decision Tree, and Neural

Networks were employed for classification in [21]. Image classification using least squares trained filters was presented in [22].

2.1.2 Regression

Regression is a type of supervised learning used to predict continuous values rather than discrete values. Regression models establish relationships between input features and the continuous target variable. They are widely used in forecasting and prediction where the output is continuous [23]. For example, regression models are applied to predict stock prices based on historical data and various market indicators. Linear regression, polynomial regression, and support vector regression are commonly used regression algorithms [24].

2.2 Unsupervised Learning

Unsupervised learning is a branch of ML where the primary focus is on finding patterns, structures, or relationships in data without using labels. Unlike supervised learning, which relies on labelled data to learn from, unsupervised learning algorithms seek to uncover hidden structures and gain insights [25]. This is particularly useful for exploratory data analysis, dimensionality reduction, and identifying natural groupings within datasets. A common type of unsupervised learning is clustering.

2.2.1 Clustering

Clustering is a type of unsupervised learning that involves grouping data into clusters or categories based on their similarities or patterns. The primary goal of clustering is to identify and create partitions within a dataset where data points within a cluster are more similar to each other than to those in other clusters. Clustering algorithms include K-means and hierarchical clustering [26]. This technique is used in various fields including customer segmentation, image processing, text analysis, and anomaly detection. Clustering assists in understanding complex datasets, revealing underlying structures, and making data-driven decisions without prior knowledge of labels or categories.

2.3 Reinforcement Learning

Reinforcement Learning (RL) is a powerful ML paradigm that employs the interaction of an agent with an environment to achieve specific objectives. An agent navigates its surroundings by taking actions, receiving feedback in the form of rewards, and refining its decision-making over time. It

is well-suited for scenarios where training data is scarce, so the agent learns through trial and error [27]. This approach is used in diverse fields from training autonomous robots and optimizing resource allocation in industrial processes to mastering complex games and enhancing recommendation systems. RL can adapt to dynamic environments and discover solutions and so is a valuable tool for tackling complex real-world challenges [28].

2.2 Jupyter Notebook

Jupyter Notebook is a versatile and widely used open-source web application used for ML classification tasks. It has an interactive and user-friendly interface which combines live code execution, visualization, and documentation, making it an indispensable tool for ML practitioners and data scientists [29]. Jupyter Notebook enables users to seamlessly write and execute Python code for building, training, and evaluating ML models. With the integration of popular libraries like Scikit-Learn, TensorFlow, and PyTorch, researchers and developers can effortlessly experiment with a variety of models [30].

Jupyter Notebook allows users to analyse model performance and explore feature engineering with real-time visualization for a more iterative and insightful approach to model development. Moreover, its ability to combine code with rich documentation in Markdown format allows for the creation of comprehensive and easily shareable ML classifier notebooks, facilitating collaboration and reproducibility. In summary, Jupyter Notebook provides an indispensable environment for ML model development, offering an interactive, customizable, and collaborative platform that enhances the efficiency and effectiveness of the model-building process. Jupyter Notebook can be easily accessed and launched through the Anaconda Navigator platform as shown in Figure 2.1.

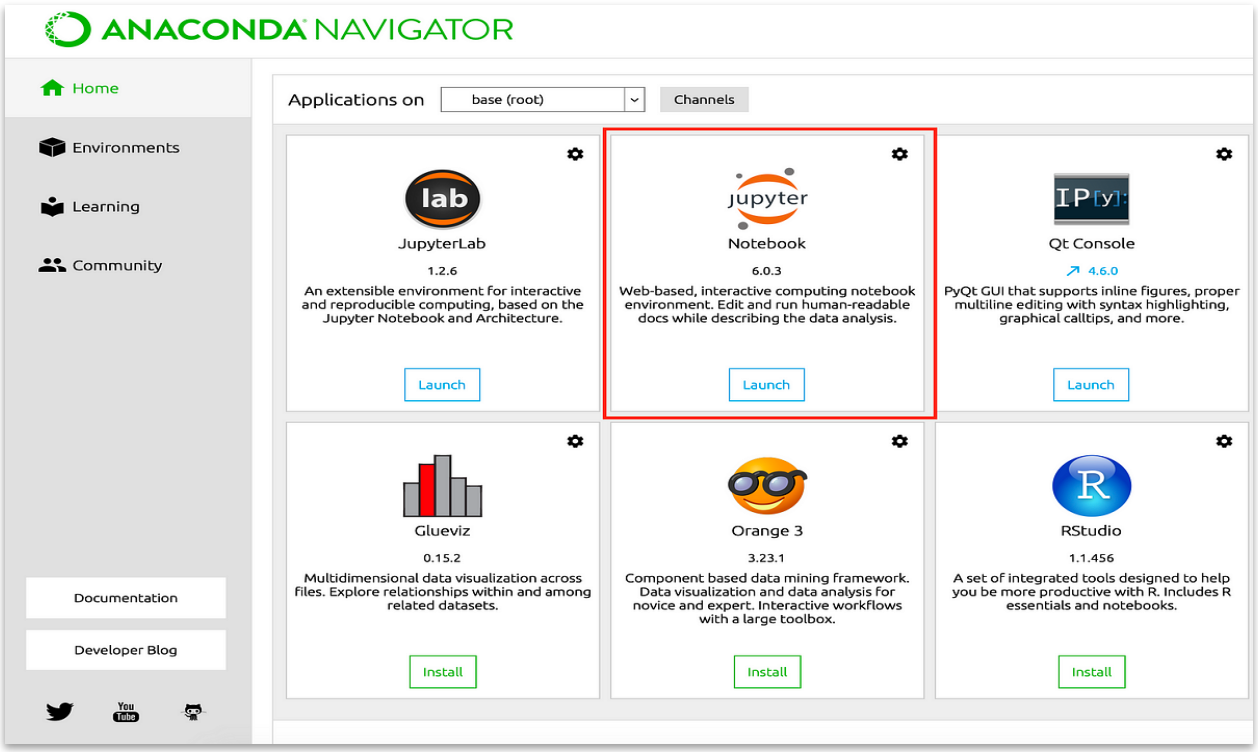

Figure 2.1 The Anaconda platform.

Jupyter Notebook has the following Python libraries. Scikit-Learn is a Python library for ML. It provides tools for various ML tasks such as classification, regression, and clustering [31]. Pandas is a Python library for data manipulation and analysis. It simplifies working with tabular data, offering data frames and series for tasks like cleaning, exploration, and transformation [32]. Matplotlib is a Python library for 2D plotting. It enables the creation of various types of static and interactive plots and serves as a foundation for other visualization libraries [33]. Seaborn is a Python library for data visualization. It enhances Matplotlib by simplifying the creation of aesthetically pleasing statistical graphics [34]. NumPy is a Python library for scientific computing. It supports large multi-dimensional arrays and offers mathematical functions for array operations [35].

Chapter 3 Breast Cancer Prediction System

Figure 3.1 illustrates the architecture of the proposed breast cancer prediction system. In the subsequent sections, the individual components are explained in detail.

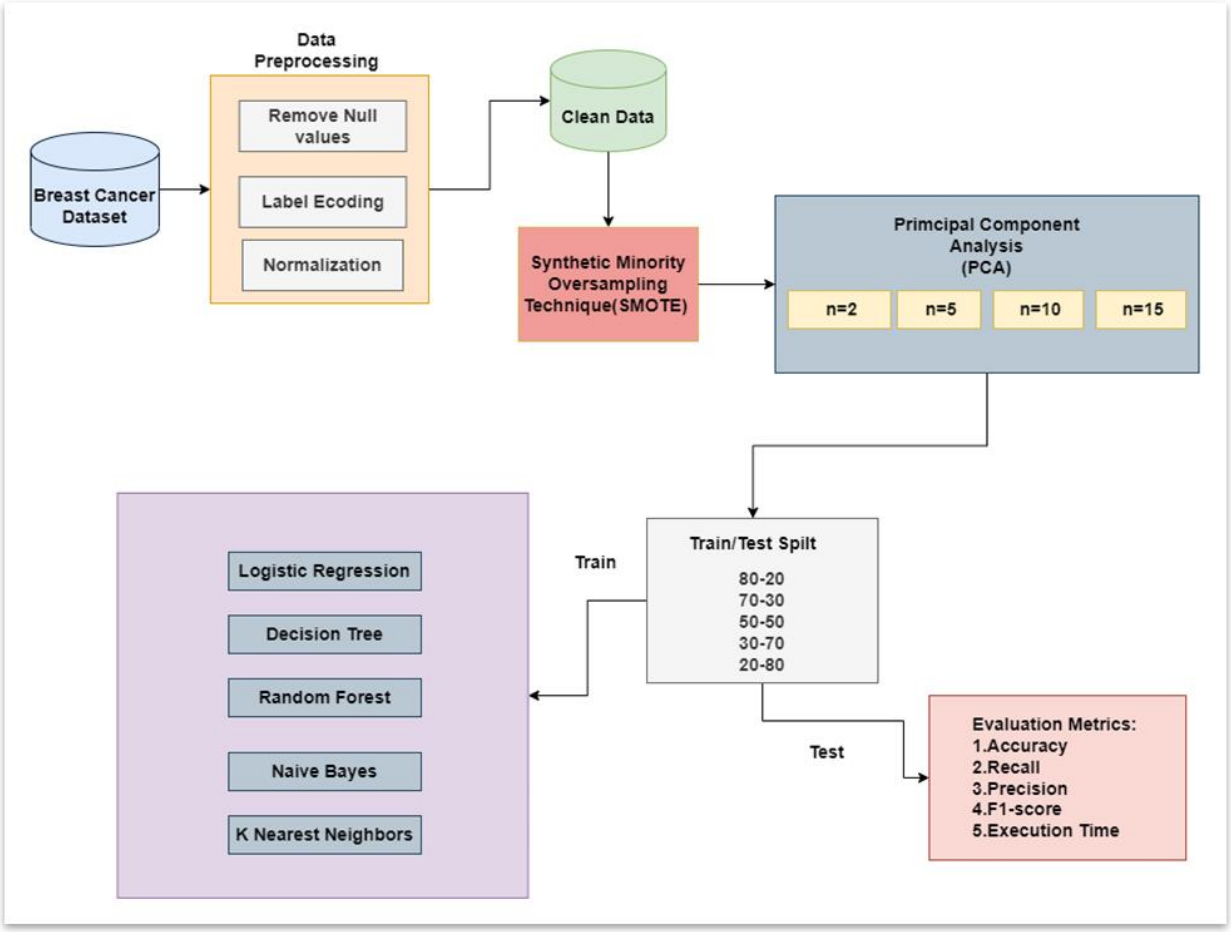

Figure 3.1 The proposed breast cancer prediction system.

3.1 The Breast Cancer Dataset

The breast cancer classification dataset, publicly accessible from the University of California Irvine (UCI) ML repository [36] is employed here. It has 32 features. They can be used as input to an ML model and there are two outputs classes, malignant (M) and benign (B), for diagnostic applications in oncology. The first feature, identification (ID), is a unique identifier for each instance or sample in the dataset. The next 30 features describe the characteristics of cell nuclei present in breast cancer biopsies and the last feature indicates if the patient has breast cancer or

not. There are 569 instances of which 357 belong to the benign class and 212 to the malignant class. The features are described in Table 3.1.

Table 3.1 Feature names and descriptions.

Feature Name	**Description**
ID	Unique identifier for each sample.
Radius_mean	Mean radius of the cell nuclei. This is the average distance from the center to the perimeter of the cell nucleus.
Texture_mean	Mean texture of the cell nuclei. This is the variation in intensity or colour of pixels in the image.
Perimeter_mean	Mean perimeter of the cell nuclei. This is the length of the boundary around the cell nucleus.
Area_mean	Mean area of the cell nuclei. This is the size of the cell nucleus.
Smoothness_mean	Mean smoothness of the cell nuclei. This is the variation in the local spacing of points on the perimeter of the cell nucleus.
Compactness_mean	Mean compactness of the cell nuclei. This is a measure of how closely the points are packed together in the cell nucleus.
Concavity_mean	Mean concavity of the cell nuclei. Concavity is the extent of indentations or depressions in the outline of the cell nucleus.
Concave points_mean	Mean of the concave points in the contour of the cell nuclei. Concave points are the number of inwardly curved points along the boundary of the cell nucleus.
Symmetry_mean	Mean symmetry of the cell nuclei. Symmetry is the similarity or balance between the halves of the cell nucleus.
Fractal_dimension_mean	Mean fractal dimension of the cell nuclei. Fractal dimension describes the complexity or irregularity of a shape at different scales.
Radius_se	Standard error of the radius of the cell nuclei. This is the variability or spread of the radius values in a given sample. Standard error is a measure of how much individual measurements vary from the mean.
Texture_se	Standard error of the texture of the cell nuclei. This is the variability or spread of the texture values in a given sample.

Perimeter_se	Standard error of the perimeter of the cell nuclei. This is the variability or spread of the perimeter values in a given sample.
Area_se	Standard error of the area of the cell nuclei. This is the variability or spread of the area values in a given sample.
Smoothness_se	Standard error of the smoothness of the cell nuclei. This is the variability or spread of the smoothness values in a given sample.
Compactness_se	Standard error of the compactness of the cell nuclei. This is the variability or spread of the compactness values in a given sample.
Concavity_se	Standard error of the concavity of the cell nuclei. This is the variability or spread of the concavity values in a given sample.
Concave points_se	Standard error of the number of the concave points in the contour of the cell nuclei. This is the variability or spread of the concave point values in a given sample.
Symmetry_se	Standard error of the symmetry of the cell nuclei. This is the variability or spread of the symmetry values in a given sample.
Fractal_dimension_se	Standard error of the fractal dimension of the cell nuclei. This is the variability or spread of the fractual dimension values in a given sample.
Radius_worst	Worst (largest) radius among all the measured cells for a patient.
Texture_worst	Worst (largest) texture among all the measured cells for a patient.
Perimeter_worst	Worst (largest) perimeter among all the measured cells for a patient.
Area_worst	Worst (largest) area among all the measured cells for a patient.
Smoothness_worst	Worst (largest) smoothness among all the measured cells for a patient.
Compactness_worst	Worst (largest) compactness among all the measured cells for a patient.
Concavity_worst	Worst (largest) concavity among all the measured cells for a patient.
Concave points_worst	Worst (largest) number of concave points among all the measured cells for a patient.
Symmetry_worst	Worst (largest) symmetry among all the measured cells for a patient.

Fractal_dimension_worst	Worst (largest) fractal dimension among all the measured cells for a patient.
Diagnosis	The outcome of the diagnostic process. This feature is binary with M indicating the presence of malignant (cancerous) cells, while B indicates the absence of malignant (non-cancerous) cells.

3.2 Dataset Preprocessing

Dataset preprocessing plays an important role in ML model development. It includes inspecting for null values, identifying whether each feature is categorical, numerical, or a timestamp, and identifying which feature is the target variable for the ML model. In this work, the target variable is diagnosis. Null values in the dataset were eliminated. Label encoding was employed to encode the target variable with a 0 assigned to benign and 1 to malignant. The bar chart in Figure 3.2 shows the target variable imbalance between benign (blue) with 357 instances and malignant (orange) with 212 instances.

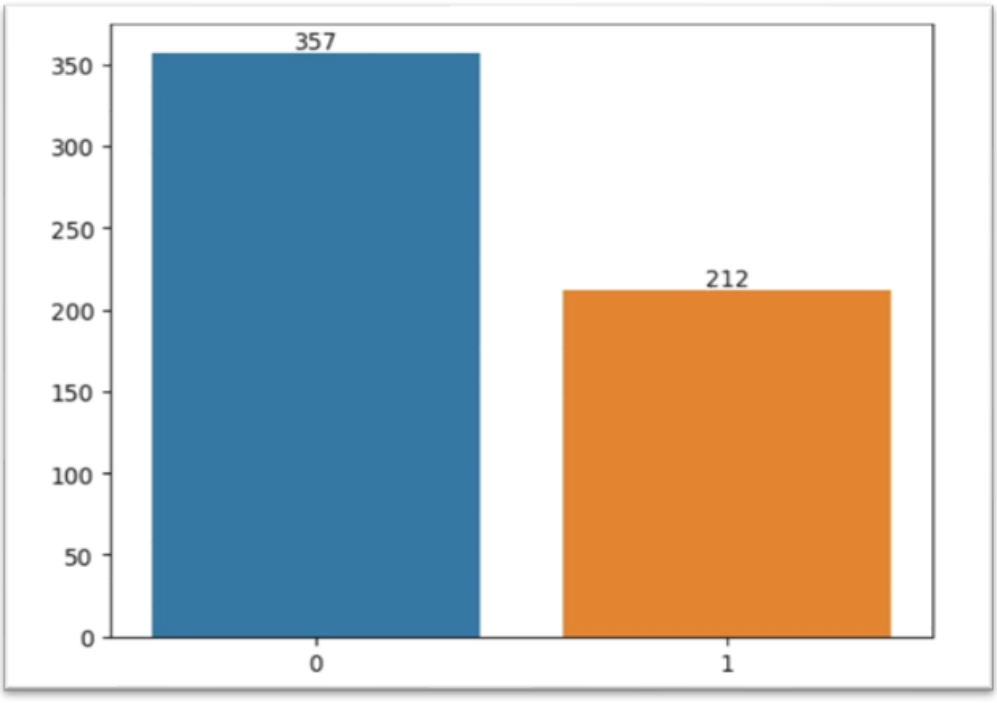

Figure 3.2 Target variable counts.

3.3 Synthetic Minority Oversampling Technique (SMOTE)

SMOTE is used to address the issue of class imbalance which can lead to biased model predictions [37]. SMOTE generates synthetic instances for the minority class to balance the class distribution. They are created by interpolating between existing minority class data points. SMOTE not only increases the overall size of the dataset but also augments the representation of the minority class to enhance the model ability to learn from and classify instances. This ensures ML models do not favour the majority class, leading to more accurate predictions, which is particularly valuable in scenarios where both classes are of equal importance [38]. After applying SMOTE, there were 339 instances in each class as shown in Figure 3.3.

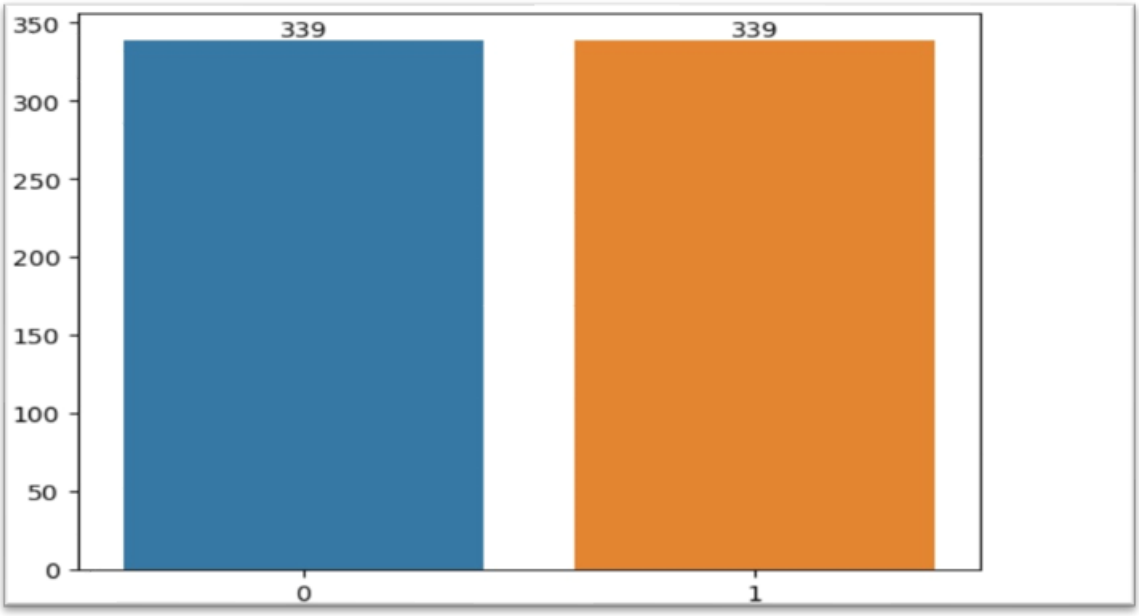

Figure 3.3 Target variable counts after SMOTE.

3.4 Principal Component Analysis (PCA)

PCA is a technique used to reduce the number of dimensions within a dataset while preserving essential information in the data [39]. This is achieved through the identification of principal components, which are the directions along which the data exhibits the most significant variation. The interrelationships between features are discerned using the correlation matrix, and eigen decomposition is used to obtain the eigenvectors and eigenvalues. The eigenvectors represent the principal components, while the eigenvalues signify the variances associated with these components. Typically, the largest eigenvectors, corresponding to the principal components, are

retained, while those with smaller eigenvalues, indicating less variance, are discarded [40]. In this work, the initial 30 features were reduced to 2, 5, 10, and 15 features as a tradeoff between dimensionality reduction and information preservation. To assess the dimensionality reduction and the retention of information, the cumulative variance ratio is employed and this is given in Table 3.2.

Table 3.2 Cumulative variance ratio of the principal components.

Principal Component Number	Cumulative Variance Ratio
1	0.43
2	0.63
3	0.72
4	0.79
5	0.84
6	0.88
7	0.91
8	0.92
9	0.93
10	0.95
11	0.96
12	0.97
13	0.97
14	0.98
15	0.98
16	0.98
17	0.99
18	0.99
19	0.99
20	0.99
21	0.99
22	0.99
23	0.99
24	0.99
25	0.99
26	0.99
27	0.99
28	0.99
29	0.99
30	1.00

The Cumulative variance ratio indicates how much of the total variance is explained by a given number of principal components. It is useful in determining how many principal components to retain in order to achieve as much information preservation as possible while reducing the dimensionality of the data.

For each principal component, the ratio of its eigenvalue to the total sum of all eigenvalues is calculated. This gives the proportion of the total variance explained by that principal component which is also known as the individual variance ratio. Then, the cumulative sum of these variance ratios is computed which indicates the amount of variance explained by the first, second, third and so on, principal components. The goal is to retain as much variance as possible while effectively reducing the dimensionality of the data, ensuring that essential information remains.

Figure 3.4 shows the individual and cumulative variance ratios for the principal components. Principal component 1 contributes the most (0.43), with subsequent components having diminishing contributions. For the cumulative variance ratios, there is a rapid initial increase with the first 15 components having 98% of the variance. This is followed by a slow increase in the cumulative variance ratio.

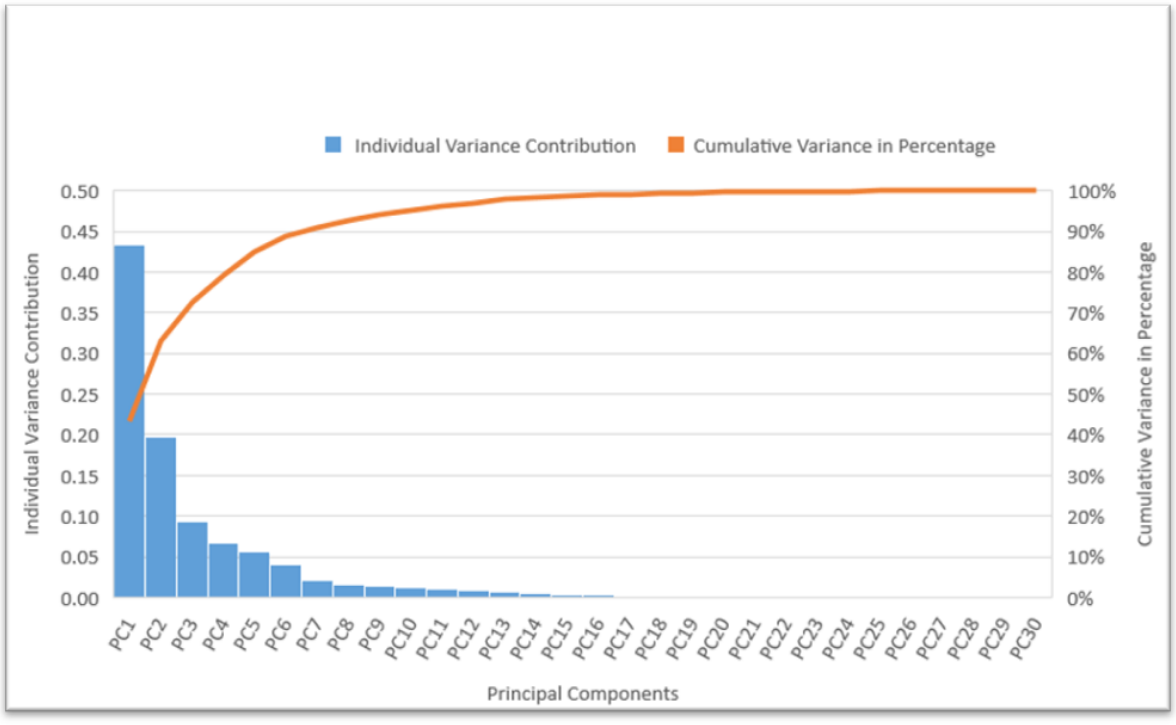

Figure 3.4 Individual and cumulative variance ratios.

3.5 Machine Learning Algorithms

The ML algorithms considered in this project are given below.

3.5.1 Logistic Regression

Logistic Regression is a widely used algorithm due to its simplicity and interpretability. It is efficient for binary classification tasks, where it can help identify the likelihood of a tumor being malignant or benign [41]. The algorithm models the relationship between the dependent binary variable and one or more independent variables by estimating probabilities using the logistic function. The result is a sigmoid-shaped curve that maps any real-valued number into a range between 0 and 1. This output is interpreted as the probability of the instance belonging to the positive (malignant) class.

3.5.2 Decision Tree

Decision Tree is a supervised ML algorithm used for both classification and regression tasks. It recursively splits the dataset into subsets based on the most significant feature at each node, forming a tree-like structure of decisions. Each internal node represents a decision based on a specific feature, and each leaf node represents the predicted outcome. Decision Tree can provide insights into the decision-making process for diagnosing cancer, making it valuable for both healthcare professionals and patients [42].

3.5.3 Random Forest

Random Forest is an ensemble technique that combines multiple decision trees to improve prediction accuracy. Each tree is trained on a random subset of the features and a random subset of the training data which reduces the probability of overfitting and enhances the model robustness. This technique is known for its accuracy, resilience to overfitting, and suitability for a wide range of tasks. Random Forest can handle high-dimensional data effectively making it a robust choice for classification tasks [43].

3.5.4 Support Vector Machine (SVM)

SVM is a powerful algorithm to predict both continuous and discrete data. It works by finding the hyperplane that best separates malignant and benign cases, maximizing the margin between the two classes. SVM is particularly effective when dealing with complex and nonlinear data [44].

3.5.5 Naive Bayes

Naive Bayes is a probabilistic ML algorithm based on Bayes' Theorem. It is particularly effective for classification tasks. The algorithm computes the probability of a class given a set of features and selects the class with the highest probability as the predicted class. Despite its simplicity, it performs well with high-dimensional datasets [45]. It is widely used in text classification, spam filtering, and applications where probability-based decisions are required.

3.5.6 K-Nearest Neighbors (KNN)

KNN is a simple and intuitive machine learning algorithm used for both classification and regression tasks. It classifies tumors based on the class of its nearest neighbors in feature space. The algorithm calculates distances, typically Euclidean distances, to find the K nearest data points and assigns the majority class among them to the new data point. KNN can be effective when data is clustered [46].

Chapter 4 ML Algorithm Performance Evaluation

In this chapter the performance of six supervised ML algorithms, namely Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN are evaluated. The results were obtained using a personal computer with the hardware and software specifications given in Table 4.1.

Table 4.1 The hardware and software parameters.

Item	**Parameter**
Manufacturer	HP
Model	Probook 450 G8 Notebook PC
Operating System	Windows 11 Home 64 bit
Processor Type	11^{th} Gen Intel(R) Core (TM) i5-1135G7
Installed Memory (RAM)	16 GB
Processor Speed	2.42 GHz
Number of Cores	4
Number of Threads	8
Python Version	3.10.9
Jupyter Notebook Version	6.5.2

4.1 Evaluation Metrics

The metrics used to evaluate ML models are described in this section. True Positive (TP) is the number of instances that the model correctly predicts as in the positive class, indicating correct identification of, for instance, malignant tumors in cancer prediction. True Negative (TN) is the number of instances where the model correctly predicts as in the negative class, indicating correct identification of, for instance, benign tumors in cancer prediction. False Positive (FP) occurs when the model wrongly predicts the positive class, indicating an erroneous identification of malignancy in cancer prediction. Conversely, False Negative (FN) occurs when the model incorrectly predicts the negative class, missing instances of actual malignancy in cancer prediction. Accuracy is the ratio of the number of correctly predicted instances to the total number of instances in the dataset and is given by [47]

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

Precision is crucial when the cost of FP is high, as it assesses the model ability to avoid false alarms and is given by [48]

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

Recall is given by [49]

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

F1-score is the harmonic mean of precision and recall, providing a single value that considers both FP and FN and is given by [50]

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Execution time is the time taken to train a model. A shorter execution times is generally preferred, especially in real-time or resource-constrained environments [51].

4.2 Performance with 80/20 split using 2 features

Each model was trained on 80% of the data and tested on the remaining 20% using 2 features. The results are given in Table 4.2. Logistic Regression had the highest accuracy, precision, recall and F1-score at 92.1%, 88.0%, 93.6%, and 90.7%, followed by KNN at 91.2%, 89.3%, 89.4%, 89.3%, Decision Tree at 90.4, 86.0%, 91.5%, 88.7%, Random Forest at 90.4%, 87.5%, 89.4%, 88.4%, Naive Bayes at 90.4%, 87.5%, 89.4%, 88.4%, and SVM at 89.5%, 85.7%, 89.4%, 87.5%. Naive Bayes had the lowest execution time at 2.60 ms, followed by KNN at 3.11 ms, Decision Tree at 6.08 ms, SVM at 10.08 ms, Logistic Regression at 12.10 ms, and Random Forest at 513.6 ms.

Table 4.2 Performance with 80/20 split using 2 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	92.1	88.0	93.6	90.7	12.10
Decision Tree	90.4	86.0	91.5	88.7	6.08
Random Forest	90.4	87.5	89.4	88.4	513.6
SVM	89.5	85.7	89.4	87.5	10.08
Naive Bayes	90.4	87.5	89.4	88.4	2.60
KNN	91.2	89.3	89.4	89.3	3.11

4.3 Performance with 80/20 split using 5 features

Each model was trained on 80% of the data and tested on the remaining 20% using 5 features. The results are given in Table 4.3. Logistic Regression, SVM and KNN had the highest accuracy, precision, recall, and F1-score at 95.6%, 95.7%, 93.6%, 94.6%, and 95.6%, 95.7%, 93.6%, 94.6%, and 95.6%, 100%, 89.4%, 94.4%, followed by Decision Tree at 93.0%, 88.2%, 95.7%, 91.8%, and Random Forest at 93.0%, 91.5%, 91.5%, 91.5%, and Naive Bayes at 92.1%, 93.2%, 87.2%, 90.1%. Naive Bayes had the lowest execution time at 2.00 ms, followed by KNN at 3.01 ms, Decision Tree at 6.01 ms, SVM at 10.99 ms, Logistic Regression at 15.04 ms, and Random Forest at 582.7 ms.

Table 4.3 Performance with 80/20 split using 5 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	95.6	95.7	93.6	94.6	15.04
Decision Tree	93.0	88.2	95.7	91.8	6.01
Random Forest	93.0	91.5	91.5	91.5	582.7
SVM	95.6	95.7	93.6	94.6	10.99
Naive Bayes	92.1	93.2	87.2	90.1	2.00
KNN	95.6	100	89.4	94.4	3.01

4.4 Performance with 80/20 split using 10 features

Each model was trained on 80% of the data and tested on the remaining 20% using 10 features. The results are given in Table 4.4. SVM had the highest accuracy, precision, recall, and F1-score at 98.2%, 100%, 95.7%, and 97.8%, followed by KNN at 96.5%, 100%, 91.5%, 95.6%, and Logistic Regression at 94.7%, 93.6%, 93.6%, 93.6%, and Random Forest at 93.9%, 95.5%, 89.4%, 92.3%, and Naive Bayes at 91.2%, 91.1%, 87.2%, 89.1%, and Decision Tree at 90.4%, 84.6%, 93.6%, 88.9%. Naive Bayes had the lowest execution time at 2.99 ms, followed by KNN at 3.01 ms, SVM at 9.99 ms, Decision Tree at 14.00 ms, Logistic Regression at 14.41 ms, and Random Forest at 707.9 ms.

Table 4.4 Performance with 80/20 split using 10 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	94.7	93.6	93.6	93.6	14.41
Decision Tree	90.4	84.6	93.6	88.9	14.00
Random Forest	93.9	95.5	89.4	92.3	707.9
SVM	98.2	100	95.7	97.8	9.99
Naive Bayes	91.2	91.1	87.2	89.1	2.99
KNN	96.5	100	91.5	95.6	3.01

4.5 Performance with 80/20 split using 15 features

Each model was trained on 80% of the data and tested on the remaining 20% using 15 features. The results are given in Table 4.5. SVM had the highest accuracy, precision, recall, and F1-score at 97.4%, 97.8%, 95.7%, and 96.7%, followed by Logistic Regression at 96.5%, 95.7%, 95.7%, 95.7%, and KNN at 95.6%, 100%, 89.4%, 94.4%, and Random Forest at 93%, 95.3%, 87.2%, 91.0%, and Naive Bayes at 92.1%, 91.3%, 89.4%, 90.3%, and Decision Tree at 91.2%, 86.2%, 93.6%, 89.7%. Naive Bayes and KNN had the lowest execution time at 1.00 ms, followed by SVM at 2.99 ms, Decision Tree at 9.00 ms, Logistic Regression at 10.16 ms, and Random Forest at 252.8 ms.

Table 4.5 Performance with 80/20 split using 15 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.5	95.7	95.7	95.7	10.16
Decision Tree	91.2	86.2	93.6	89.7	9.00
Random Forest	93.0	95.3	87.2	91.0	252.8
SVM	97.4	97.8	95.7	96.7	2.99
Naive Bayes	92.1	91.3	89.4	90.3	1.00
KNN	95.6	100	89.4	94.4	1.00

4.6 Performance with 70/30 split using 2 features

Each model was trained on 70% of the data and tested on the remaining 30% using 2 features. The results are given in Table 4.6. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 93.6%, 89.4%, 93.7%, and 91.5%, followed by Random Forest at 91.2%, 87.5%, 88.9%, 88.1%, and KNN at 91.2%, 88.7%, 87.3%, 88.0%, and Decision Tree at 90.1%, 81.9%, 93.7%, 87.4%, and SVM at 90.1%, 86.0%, 87.3%, 86.7%, and Naive Bayes at 89.5%, 84.6%, 87.3%, 85.9%. Naive Bayes and KNN had the lowest execution time at 0.99 ms, followed by Decision Tree at 1.99 ms, SVM at 2.99 ms, Logistic Regression at 5.99 ms, and Random Forest at 170.6 ms.

Table 4.6 Performance with 70/30 split using 2 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	93.6	89.4	93.7	91.5	5.99
Decision Tree	90.1	81.9	93.7	87.4	1.99
Random Forest	91.2	87.5	88.9	88.1	170.6
SVM	90.1	86.0	87.3	86.7	2.99
Naive Bayes	89.5	84.6	87.3	85.9	0.99
KNN	91.2	88.7	87.3	88.0	0.99

4.7 Performance with 70/30 split using 5 features

Each model was trained on 70% of the data and tested on the remaining 30% using 5 features. The results are given in Table 4.7. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 97.1%, 96.8%, 95.2%, and 96.0%, followed by SVM at 95.9%, 95.1%, 93.7%, 94.4%, and KNN at 95.3%, 96.6%, 90.4%, 93.4%, and Decision Tree at 94.7%, 88.6%, 98.4%, 93.2%, and Random Forest at 93.0%, 89.2%, 92.0%, 90.6%, and Naive Bayes at 91.2%, 90.0%, 85.7%, 87.8%. KNN had the lowest execution time at 0.99 ms, followed by Naive Bayes at 1.00 ms, SVM and Decision Tree at 3.01 ms, Logistic Regression at 7.26 ms, and Random Forest at 194.0 ms.

Table 4.7 Performance with 70/30 split using 5 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	97.1	96.8	95.2	95.0	7.26
Decision Tree	94.7	88.6	98.4	93.2	3.01
Random Forest	93.0	89.2	92.0	90.6	194.0
SVM	95.9	95.1	93.7	94.4	3.01
Naive Bayes	91.2	90.0	85.7	87.8	1.00
KNN	95.3	96.6	90.4	93.4	0.99

4.8 Performance with 70/30 split using 10 features

Each model was trained on 70% of the data and tested on the remaining 30% using 10 features. The results are given in Table 4.8. SVM had the highest accuracy, precision, recall, and F1-score at 97.7%, 98.4%, 95.2%, and 96.8%, followed by KNN at 95.9%, 98.2%, 90.5%, 94.2%, Logistic Regression at 95.9%, 94.8%, 95.23%, 94.5%, Decision Tree at 91.2%, 86.4%, 90.5%, 88.4%, Random Forest at 91.2%, 87.5%, 88.9%, 88.2%, and Naive Bayes at 91.2%, 88.7%, 87.3%, 88.0%. Naive Bayes had the lowest execution time at 0.99 ms, followed by KNN at 1.01 ms, SVM at 3.00 ms, Decision Tree at 3.65 ms, Logistic Regression at 9.96 ms, and Random Forest at 239.1 ms.

Table 4.8 Performance with 70/30 split using 10 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	95.9	93.8	95.2	94.5	9.96
Decision Tree	91.2	86.4	90.5	88.4	3.65
Random Forest	91.2	87.5	88.9	88.2	239.1
SVM	97.7	98.4	95.2	96.8	3.00
Naive Bayes	91.2	88.7	87.3	88.0	0.99
KNN	95.9	98.2	90.5	94.2	1.01

4.9 Performance with 70/30 split using 15 features

Each model was trained on 70% of the data and tested on the remaining 30% using 15 features. The results are given in Table 4.9. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 97.7%, 98.4%, 95.2%, 96.8%, followed by SVM at 97.1%, 96.8%, 95.2%, 96.0%, KNN at 93.6%, 96.4%, 85.7%, 90.7%, Decision Tree at 92.4%, 86.8%, 93.7%, 90.11%, Random Forest at 91.8%, 87.7%, 90.5%, 89.0%, and Naive Bayes at 91.8%, 88.9%, 88.9%, 88.9%. Naive Bayes and KNN had the lowest execution time at 0.99 ms, followed by SVM at 2.01 ms, Decision Tree at 4.99 ms, Logistic Regression at 10.45 ms, and Random Forest at 240.0 ms.

Table 4.9 Performance with 70/30 Split using 15 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	97.7	98.4	95.2	96.8	10.45
Decision Tree	92.4	86.8	93.7	90.11	4.99
Random Forest	91.8	87.7	90.5	89.0	240.0
SVM	97.1	96.8	95.2	96.0	2.01
Naive Bayes	91.8	88.9	88.9	88.9	0.99
KNN	93.6	96.4	85.7	90.7	0.99

4.10 Performance with 50/50 split using 2 features

Each model was trained on 50% of the data and tested on the remaining 50% using 2 features. The results are given in Table 4.10. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 94.4%, 92.0%, 92.0%, and 92.0%, followed by KNN at 91.9%, 90.6%, 86.1%, 88.3%, Random Forest at 91.6%, 88.9%, 87.1%, 88.0%, SVM at 91.6%, 89.7%, 86.1%, 87.9%, Naive Bayes at 91.2%, 88.8%, 86.1%, 87.4%, and Decision Tree at 89.8%, 86.0%, 85.1%, 85.6%. SVM and Naive Bayes had the lowest execution time at 0.99 ms, followed by KNN at 1.00 ms, Decision Tree at 1.51 ms, Logistic Regression at 5.50 ms, and Random Forest at 143.0 ms.

Table 4.10 Performance with 50/50 split using 2 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	94.4	92.0	92.0	92.0	5.50
Decision Tree	89.8	86.0	85.1	85.6	1.51
Random Forest	91.6	88.9	87.1	88.0	143.0
SVM	91.6	89.7	86.1	87.9	0.99
Naive Bayes	91.2	88.8	86.1	87.4	0.99
KNN	91.9	90.6	86.1	88.3	1.00

4.11 Performance with 50/50 split using 5 features

Each model was trained on 50% of the data and tested on the remaining 50% using 5 features. The results are given in Table 4.11. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 96.5%, 97.0%, 93.0%, and 95.0%, followed by SVM at 95.4%, 94.9%, 92.0%, 93.4%, KNN at 95.1%, 95.8%, 90.0%, 92.8%, Random Forest at 93%, 90.9%, 89.1%, 90.0%, Naive Bayes at 93.0%, 93.5%, 86.1%, 89.6%, and Decision Tree at 90.2%, 88.4%, 83.1%, 85.7%. Naive Bayes and KNN had the lowest execution time at 1.00 ms, followed by Decision Tree at 2.09 ms, SVM at 2.99 ms, Logistic Regression at 8.01 ms, and Random Forest at 167.1 ms.

Table 4.11 Performance with 50/50 split using 5 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1- score (%)	Execution Time (ms)
Logistic Regression	96.5	97.0	93.0	95.0	8.01
Decision Tree	90.2	88.4	83.1	85.7	2.09
Random Forest	93.0	90.9	89.1	90.0	167.1
SVM	95.4	94.9	92.0	93.4	2.99
Naive Bayes	93.0	93.5	86.1	89.6	1.00
KNN	95.1	95.8	90.0	92.8	1.00

4.12 Performance with 50/50 split using 10 features

Each model was trained on 50% of the data and tested on the remaining 50% using 10 features. The results are given in Table 4.12. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 97.2%, 96.0%, 96.0%, and 96.0%, followed by SVM at 96.5%, 96.0%, 94.0%, 95.0%, KNN at 94%, 96.7%, 86.1%, 91.0%, Random Forest at 92.3%, 89.9%, 88.1%, 89.0%, and Naive Bayes at 91.9%, 91.5%, 85.1%, 88.1%, and Decision Tree at 90.5%, 86.2%, 87.1%, 86.7%. KNN had the lowest execution time at 0.99 ms, followed by Naive Bayes at 1.00 ms, SVM at 2.00 ms, Decision Tree at 2.99 ms, Logistic Regression at 7.60 ms, and Random Forest at 179.9 ms.

Table 4.12 Performance with 50/50 split using 10 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	97.2	96.0	96.0	96.0	7.60
Decision Tree	90.5	86.2	87.1	86.7	2.99
Random Forest	92.3	89.9	88.1	89.0	179.9
SVM	96.5	96.0	94.0	95.0	2.00
Naive Bayes	91.9	91.5	85.1	88.1	1.00
KNN	94.0	96.7	86.1	91.0	0.99

4.13 Performance with 50/50 split using 15 features

To determine the performance of the models, the same dataset was used for several algorithms. Each model was trained on 50% of the data and tested on the remaining 50% using 15 features. The results are given in Table 4.13. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 97.9%, 98.0%, 96.0%, and 97.0%, followed by SVM at 97.2%, 97.0%, 95.0%, 96.0%, KNN at 94.7%, 97.8%, 87.1%, 92.1%, Naive Bayes at 92.3%, 92.4%, 85.1%, 88.6%, Random Forest at 91.9%, 88.2%, 89.1%, 88.6%, and Decision Tree at 91.2%, 88.8%, 86.1%, 87.4%. KNN had the lowest execution time at 0.98 ms, followed by Naive Bayes at 1.04 ms, SVM at 1.96 ms, Decision Tree at 3.99 ms, Logistic Regression at 8.42 ms, and Random Forest at 203.7 ms.

Table 4.13 Performance with 50/50 split using 15 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	97.9	98.0	96.0	97.0	8.42
Decision Tree	91.2	88.8	86.1	87.4	3.99
Random Forest	91.9	88.2	89.1	88.6	203.7
SVM	97.2	97.0	95.0	96.0	1.96
Naive Bayes	92.3	92.4	85.1	88.6	1.04
KNN	94.7	97.8	87.1	92.1	0.98

4.14 Performance with 30/70 split using 2 features

Each model was trained on 30% of the data and tested on the remaining 70% using 2 features. The results are given in Table 4. 14. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 94.2%, 93.6%, 90.4%, and 92.0%, followed by SVM at 92.7%, 92.7%, 87.0%, 89.8%, KNN at 92.5%, 92.6%, 86.3%, 89.3%, Decision Tree at 92.2%, 92.0%, 86.3%, 89.0%, Random Forest at 92%, 90.1%, 87.7%, 88.9%, and Naive Bayes at 91.2%, 90.5%, 85.0%, 87.7%. KNN had the lowest execution time at 0.94 ms, followed by SVM at 0.98 ms, Naive Bayes at 1.00 ms, Decision Tree at 2.00 ms. Logistic Regression at 3.99 ms, and Random Forest at 98.19 ms.

Table 4.14 Performance with 30/70 split using 2 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	94.2	93.6	90.4	92.0	3.99
Decision Tree	92.2	92.0	86.3	89.0	2.00
Random Forest	92.0	90.1	87.7	88.9	98.19
SVM	92.7	92.7	87.0	89.8	0.98
Naive Bayes	91.2	90.5	85.0	87.7	1.00
KNN	92.5	92.6	86.3	89.3	0.94

4.15 Performance with 30/70 split using 5 features

Each model was trained on 30% of the data and tested on the remaining 70% using 5 features. The results are given in Table 4. 15. Logistic Regression and SVM had the highest accuracy, precision, recall, and F1-score at 95.7%, 97.0%, 91.0%, 94.0%, and 95.7%, 97.0%, 91.0%, 94.0%, followed by KNN at 93.2%, 94.0%, 87.0%, 90.4%, Decision Tree at 92.5%, 92.6%, 86.3%, 89.3%, Random Forest at 91.5%, 88.4%, 88.4%, 88.4%, and Naive Bayes at 91.2%, 90.5%, 85.0%, 87.7%. Decision Tree, Naive Bayes, and KNN had the lowest execution time at 0.99 ms, followed by SVM at 1.00 ms, Logistic Regression at 5.01 ms, and Random Forest at 106.9 ms.

Table 4.15 Performance with 30/70 split using 5 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	95.7	97.0	91.0	94.0	5.01
Decision Tree	92.5	92.6	86.3	89.3	0.99
Random Forest	91.5	88.4	88.4	88.4	106.9
SVM	95.7	97.0	91.0	94.0	1.00
Naive Bayes	91.2	90.5	85.0	87.7	0.99
KNN	93.2	94.0	87.0	90.0	0.99

4.16 Performance with 30/70 split using 10 features

Each model was trained on 30% of the data and tested on the remaining 70% using 10 features. The results are given in Table 4. 16. Logistic Regression the highest accuracy, precision, recall, and F1-score at 96.7%, 97.8%, 93.1%, 95.4%, followed by SVM at 96.2%, 97.1%, 92.5%, 94.7%, KNN at 94.5%, 97.7%, 87.0%, 92.0%, Decision Tree at 92.0%, 91.3%, 86.3%, 88.7%, Random Forest at 91.7%, 89.0%, 88.4%, 88.7%, and Naive Bayes at 91.0%, 91.0%, 83.6%, 87.1%. KNN had the lowest execution time at 1.00 ms, followed by Naive Bayes at 1.01 ms, SVM at 1.07 ms, Decision Tree at 1.90 ms, Logistic Regression 7.09 ms, and Random Forest at 157.6 ms.

Table 4.16 Performance with 30/70 split using 10 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.7	97.8	93.1	95.4	7.09
Decision Tree	92.0	91.3	86.3	88.7	1.90
Random Forest	91.7	89.0	88.4	88.7	157.6
SVM	96.2	97.1	92.5	94.7	1.07
Naive Bayes	91.0	91.0	83.6	87.1	1.01
KNN	94.5	97.7	87.0	92.0	1.00

4.17 Performance with 30/70 split using 15 features

Each model was trained on 30% of the data and tested on the remaining 70% using 15 features. The results are given in Table 4. 17. Logistic Regression the highest accuracy, precision, recall, and F1-score at 96.5%, 97.1%, 93.1%, 95.0%, followed by SVM at 96.0%, 96.4%, 92.5%, 94.4%, KNN at 94.2%, 97.0%, 87.0%, 91.7%, Decision Tree at 92.2%, 92.0%, 86.3%, 89.0%, Random Forest at 91.7%, 89.0%, 88.3%, 88.6%, and Naive Bayes at 90.7%, 91.0%, 82.3%, 86.4%. Naive Bayes and KNN had the lowest execution time at 0.99 ms, followed by SVM at 1.01 ms, Decision Tree at 1.99 ms, Logistic Regression at 5.99 ms, and Random Forest at 181.5 ms.

Table 4.17 Performance with 30/70 split using 15 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.5	97.1	93.1	95.0	5.99
Decision Tree	92.2	92.0	86.3	89.0	1.99
Random Forest	91.7	89.0	88.3	88.6	181.5
SVM	96.0	96.4	92.5	94.4	1.01
Naive Bayes	90.7	91.0	82.3	86.4	0.99
KNN	94.2	97.0	87.0	91.7	0.99

4.18 Performance with 20/80 split using 2 features

Each model was trained on 20% of the data and tested on the remaining 80% using 2 features. The results are given in Table 4. 18. Logistic Regression the highest accuracy, precision, recall, and F1-score at 93.0%, 94.1%, 86.4%, 90.0%, followed by KNN at 92.5%, 92.5%, 87.0%, 90.0%, SVM at 92.3%, 94.0%, 84.6%, 89.0%, Random Forest at 91.5%, 89.6%, 87.0%, 88.2%, Decision Tree at 91.0%, 88.6%, 87.0%, 87.8%, and Naive Bayes at 89.7%, 88.1%, 83.4%, 85.7%. SVM had the lowest execution time at 0.94 ms, followed by KNN at 0.99 ms, Naive Bayes at 1.04 ms, Decision Tree at 1.19 ms, Random Forest at 3.76 ms, and Random Forest at 127.6 ms.

Table 4.18 Performance with 20/80 split using 2 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1- score (%)	Execution Time (ms)
Logistic Regression	93.0	94.1	86.4	90.0	3.76
Decision Tree	91.0	88.6	87.0	87.8	1.19
Random Forest	91.5	89.6	87.0	88.2	127.6
SVM	92.3	94.0	84.6	89.0	0.94
Naive Bayes	89.7	88.1	83.4	85.7	1.04
KNN	92.5	92.5	87.0	90.0	0.99

4.19 Performance with 20/80 split using 5 features

Each model was trained on 20% of the data and tested on the remaining 80% using 5 features. The results are given in Table 4. 19. Logistic Regression the highest accuracy, precision, recall, and F1-score at 96.1%, 96.9%, 92.3%, 94.5%, followed by SVM at 94.3%, 95.0%, 89.3%, 92.0%, KNN at 92.5%, 94.7%, 84.6%, 89.4%, Decision Tree at 91.7%, 90.7%, 86.4%, 88.5%, Random Forest at 91.4%, 89.6%, 87.0%, 88.2%, and Naive Bayes at 90.8%, 89.4%, 85.2%, 87.2%. SVM had the lowest execution time at 0.98 ms, followed by KNN at 0.99 ms, Decision Tree at 1.00 ms, Naive Bayes at 1.02 ms, Logistic Regression at 4.99 ms, and Random Forest at 107.5 ms.

Table 4.19 Performance with 20/80 split using 5 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.1	96.9	92.3	94.5	4.99
Decision Tree	91.7	90.7	86.4	88.5	1.00
Random Forest	91.4	89.6	87.0	88.2	107.5
SVM	94.3	95.0	89.3	92.0	0.98
Naive Bayes	90.8	89.4	85.2	87.2	1.02
KNN	92.5	94.7	84.6	89.4	0.99

4.20 Performance with 20/80 split using 10 features

Each model was trained on 20% of the data and tested on the remaining 80% using 10 features. The results are given in Table 4.20. Logistic Regression had the highest accuracy, precision, recall, and F1-score at 96.7%, 98.1%, 92.9%, 95.4%, followed by SVM at 95.6%, 96.9%, 91.1%, 93.9%, KNN at 92.8%, 95.9%, 84.0%, 90.0%, Decision Tree at 91.7%, 93.3%, 83.4%, 88.0%, Random Forest at 91.0%, 89.0%, 86.4%, 87.7%, and Naive Bayes at 90.4%, 89.3%, 84.0%, 86.6%. Naive Bayes had the lowest execution time at 0.67 ms, followed by KNN at 1.00 ms, Decision Tree at 2.00 ms, SVM at 2.01 ms, Logistic Regression at 6.13 ms, and Random Forest at 131.7 ms.

Table 4.20 Performance with 20/80 split using 10 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.7	98.1	92.9	95.4	6.13
Decision Tree	91.7	93.3	83.4	88.0	2.00
Random Forest	91.0	89.0	86.4	87.7	131.7
SVM	95.6	96.9	91.1	93.9	2.01
Naive Bayes	90.4	89.3	84.0	86.6	0.67
KNN	92.8	95.9	84.0	90.0	1.00

4.21 Performance with 20/80 split using 15 features

Each model was trained on 20% of the data and tested on the remaining 80% using 15 features. The results are given in Table 4.21. Logistic Regression the highest accuracy, precision, recall, and F1-score at 96.3%, 97.0%, 92.9%, 95.0%, followed by SVM at 95.8%, 97.5%, 91.1%, 94.1%, KNN at 93.4%, 97.3%, 84.6%, 90.5%, Random Forest at 91.4%, 90.6%, 85.8%, 88.1%, Decision Tree at 89.7%, 89.1%, 82.2%, 85.5%, and Naive Bayes at 89.5%, 88.0%, 82.8%, 85.3%. KNN had the lowest execution time at 0.78 ms, SVM at 0.89 ms, Naive Bayes at 1.02 ms, Decision Tree at 2.01 ms, Logistic Regression at 3.98 ms, and Random Forest at 107.9 ms.

Table 4.21 Performance with 20/80 split using 15 features.

Classifier	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Execution Time (ms)
Logistic Regression	96.3	97.0	92.9	95.0	3.98
Decision Tree	89.7	89.1	82.3	85.5	2.05
Random Forest	91.4	90.6	85.8	88.1	107.9
SVM	95.8	97.5	91.1	94.1	0.89
Naive Bayes	89.5	88.0	82.8	85.3	1.02
KNN	93.4	97.3	84.6	90.5	0.78

4.22 Discussion

Figure 4.22.1 presents the accuracy of six ML models (Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN) with five data splitting ratios (80-20, 70-30, 50-50, 30-70, and 20-80) using 2 features.

For the 80/20 split using 2 features, Logistic Regression achieved the highest accuracy of 92.1%, with execution time of 12.10 ms. KNN attained the second-highest accuracy of 91.2% with an execution time of 3.11 ms. Decision Tree, Naive Bayes, and Random Forest were all ranked third, and SVM sixth. In terms of execution time, Naive Bayes was the fastest model with 2.60 ms. The slowest model was Random Forest, with an execution time of 513.6 ms.

For the 70/30 split using 2 features, Logistic Regression achieved the highest accuracy of 93.6% with an execution time of 5.99 ms. KNN and Random Forest were second with both having an accuracy of 91.2%. Decision Tree and SVM were fourth, while Naive Bayes was sixth. The fastest models in terms of execution time, both at 0.99 ms, were Naive Bayes and KNN. The slowest model was Random Forest with an execution time of 170.6 ms.

For the 50/50 split using 2 features, Logistic Regression achieved the highest accuracy of 94.4%, with an execution time of 5.50 ms. KNN attained the second-highest accuracy of 91.9% with an execution time of 0.99 ms. SVM and Random Forest were ranked third as both achieved the same accuracy of 91.6%. Naive Bayes was third, while Decision Tree was fourth. In terms of execution time, Naive Bayes was the fastest model with 0.99 ms. The slowest model was Random Forest with an execution time of 143.0 ms.

For the 30/70 split using 2 features, Logistic Regression achieved the highest accuracy of 94.2%, with an execution time of 3.99 ms. SVM attained the second-highest accuracy of 92.7% with an execution time of 0.98 ms. KNN secured the third-highest accuracy of 92.5%, followed by Decision Tree with an accuracy of 92.2%, Random Forest with 92.0%, and Naive Bayes with 91.2% accuracy. In terms of execution time, KNN was the fastest model with 0.94 ms. The slowest model was the Random Forest, with an execution time of 98.19 ms.

For the 20/80 split using 2 features, Logistic Regression achieved the highest accuracy of 93% with an execution time of 3.76 ms, while KNN attained the second-highest accuracy of 92.5% with an execution time of 0.99 ms. SVM secured the third-highest accuracy of 92.3%, followed by Random Forest with 91.5%, Decision Tree with 91.0%, and Naive Bayes with 89.7% accuracy. In terms of execution time, SVM was the fastest model with 0.94 ms. The slowest model was Random Forest, with an execution time of 127.6 ms.

By comparing all splits using 2 features, the highest accuracy is 94.4% obtained by Logistic Regression with 5.50 ms execution time and a 50/50 split. The lowest execution time is 0.94 ms obtained by KNN with an accuracy of 92.5% and a 30/70 split.

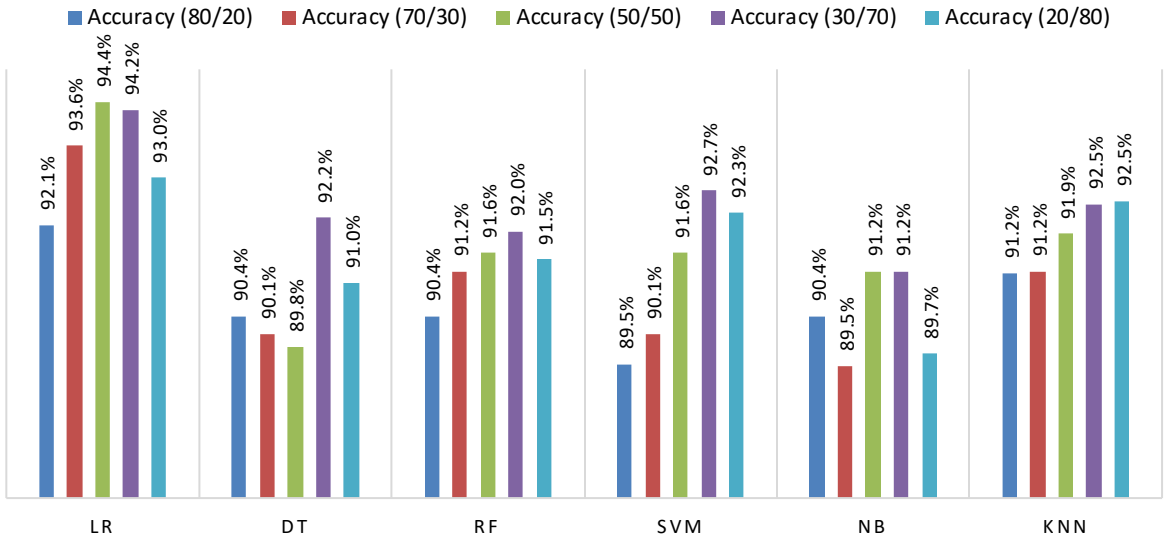

Figure 4.22.1 Machine learning model performance using 2 features.

Figure 4.22.2 presents the accuracy of six ML models (Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN) with five data splitting ratios (80/20, 70/30, 50/50, 30/70, and 20/80) using 5 features.

For the 80/20 split using 5 features, Logistic Regression, SVM, and KNN achieved the highest accuracy of 95.6% with execution times of 15.04 ms, 10.99 ms, and 3.01 ms, respectively. Decision Tree and Random Forest attained the second-highest accuracy of 93.0%, while Naive Bayes achieved the third-highest accuracy of 92.1%. In terms of execution time, KNN was the fastest model with 2.60 ms. The slowest model was Random Forest, with an execution time of 582.7 ms.

For the 70/30 split using 5 features, Logistic Regression achieved the highest accuracy of 97.1%, with an execution time of 7.26 ms, while SVM achieved the second-highest accuracy of 95.9%, with an execution time of 3.01 ms. KNN had the third-highest accuracy of 95.3%, with an execution time of 0.99 ms. Decision Tree ranked fifth with 94.7%, Random Forest placed fifth with 93.0%, and Naive Bayes sixth with 91.2% accuracy. In terms of execution time, KNN was the fastest model with 0.99 ms. Random Forest was the slowest model with an execution time of 194.0 ms.

For the 50/50 split using 5 features, Logistic Regression achieved the highest accuracy of 96.5%, with an execution time of 8.01 ms. SVM had the second-highest accuracy of 95.4%, with an execution time of 2.99 ms. KNN had the third-highest accuracy of 95.1%, with an execution time of 1.00 ms. Random Forest and Naive Bayes were fourth place with 93.0%, while Decision Tree ranked sixth with 90.2% accuracy. In terms of execution time, KNN and Naive Bayes were the fastest models, both having an execution time of 1.00 ms. The slowest model was Random Forest with an execution time of 167.1 ms.

For the 30/70 split using 5 features, SVM and Logistic Regression achieved the highest accuracy of 95.7%, with execution times of 1.00, and 5.01 ms, respectively. KNN attained the third-highest accuracy of 93.2%, with an execution time of 0.99 ms. Decision Tree ranked fourth with 92.5%, Random Forest fifth with 91.5%, and Naive Bayes sixth with 91.2% accuracy. In terms of execution time, KNN and Naive Bayes were the fastest models with both having an execution time of 0.99 ms. Random Forest was the slowest model with an execution time of 106.9 ms.

For the 20/80 split using 5 features, Logistic Regression achieved the highest accuracy of 96.1%, with an execution time of 4.99 ms. SVM attained the second-highest accuracy of 94.3%, with an execution time of 0.98 ms. KNN achieved the third-highest accuracy of 92.5%, with an execution time of 0.99 ms. Decision Tree ranked fourth with 91.7%, Random Forest fifth with 91.4%, and Naive Bayes sixth with 90.8%. In terms of execution time, SVM was the fastest model with 0.98 ms. The slowest model was the Random Forest with an execution time of 107.5 ms.

By comparing all splits using 5 features, the highest accuracy is 97.1% obtained by Logistic Regression with 7.26 ms execution time and a 70/30 split. The lowest execution time is 0.98 ms by SVM with an accuracy of 94.3% and a 20/80 split.

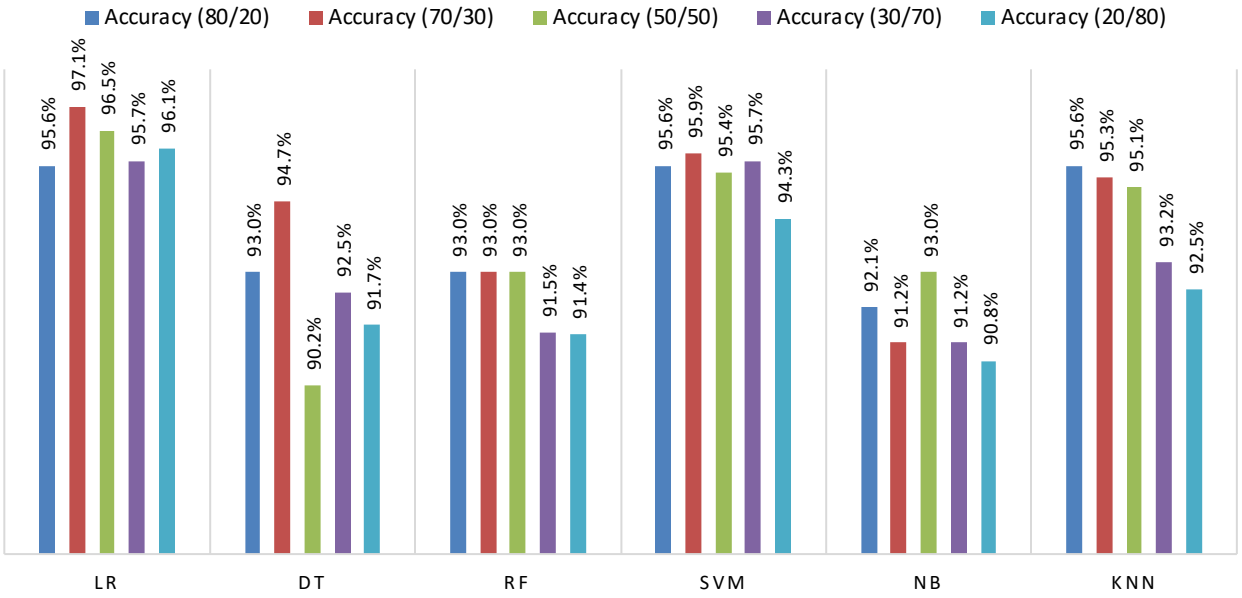

Figure 4.22.2 Machine learning model performance using 5 features.

Figure 4.22.3 presents the accuracy of six ML models (Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN) with five data splitting ratios (80/20, 70/30, 50/50, 30/70, and 20/80) using 10 features.

For the 80/20 split using 10 features, SVM achieved the highest accuracy of 98.2%, with an execution time of 9.99 ms. KNN obtained second-highest accuracy of 96.5%, with an execution time of 3.01 ms. Logistic Regression secured the third-highest accuracy of 94.7%, with execution time of 14.41 ms. Random Forest ranked fourth with 93.9%, Naive Bayes came fifth with 91.2%, and Decision Tree sixth with 90.4% accuracy. In terms of execution time, Naive Bayes was the fastest model with 2.99 ms. Random Forest was the slowest model with an execution time of 707.9 ms.

For the 70/30 split using 10 features, SVM achieved the highest accuracy of 97.7%, with an execution time of 3.00 ms. KNN and Logistic Regression achieved the second-highest accuracy of 95.9%, with execution times of 1.01 ms, and 9.96 ms, respectively. Decision Tree, Naive Bayes, and Random Forest achieved the third highest accuracy of 91.2%. In terms of execution time, Naive Bayes was the fastest model with 0.99 ms. Random Forest was the slowest model with an execution time of 239.1 ms.

For the 50/50 split using 10 features, Logistic Regression achieved the highest accuracy of 97.2%, with an execution time of 7.60 ms. SVM achieved the second-highest accuracy of 96.5%, with an execution time of 2.00 ms. KNN attained the third-highest accuracy of 94%, with an execution time of 1.00 ms. Random Forest ranked fourth with 92.3%, Naive Bayes fifth with 91.9%, and Decision Tree sixth with 90.5% accuracy. In terms of execution time, KNN was the fastest model with 0.99 ms. The slowest model was Random Forest with an execution time of 179.9 ms.

For the 30/70 split using 10 features, Logistic Regression achieved the highest accuracy of 96.7%, with an execution time of 7.09 ms. SVM achieved the second-highest accuracy of 96.2%, with an execution time of 1.01 ms. KNN obtained the third-highest accuracy of 94.5%, with an execution time of 1.00 ms. Decision Tree ranked fourth with 92.0%, followed by Random Forest with 91.7%, and Naive Bayes came last with 91.0% accuracy. In terms of execution time, KNN was the fastest model with 1.00 ms. Random Forest was the slowest model with an execution time of 157.6 ms.

For a 20/80 split using 10 features, Logistic Regression achieved the highest accuracy of 96.7%, with an execution time of 6.13 ms. SVM came in second with an accuracy of 95.6%, with an execution time of 2.01 ms. KNN had the third-highest accuracy of 92.8%, with an execution time of 1.00 ms. Decision Tree was fourth with 91.7%, followed by Random Forest with 91%, and Naive Bayes with 90.4% accuracy. In terms of execution time, Naive Bayes was the fastest model with 0.67 ms. Random Forest was the slowest model with an execution time of 131.7 ms

By comparing all splits using 10 features, the highest accuracy is 98.2% obtained by SVM with 9.99 ms execution time and a 80/20 split. The lowest execution time is 0.67 ms obtained by Naive Bayes with an accuracy of 90.4% and a 20/80 split.

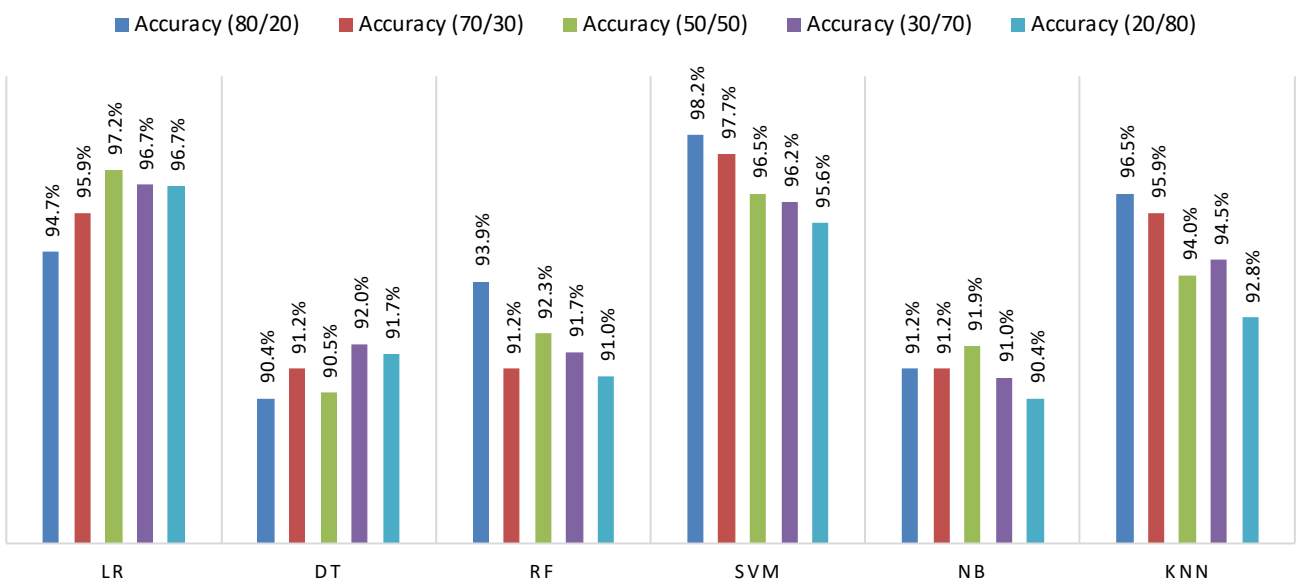

Figure 4.22.3 Machine learning model performance using 10 features.

Figure 4.22.4 presents the accuracy of six ML models (Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN) with five data splitting ratios (80/20, 70/30, 50/50, 30/70, and 20/80) using 15 features.

For the 80/20 split using 15 features, SVM achieved the highest accuracy of 97.4%, with an execution time of 2.99 ms. Logistic Regression achieved the second-highest accuracy of 96.5%, with an execution time of 10.16 ms. KNN obtained the third-highest accuracy of 95.6%, with an execution time of 1.00 ms. Random Forest ranked fourth with 93%, followed by Naive Bayes with 92.1% and Decision Tree with 91.2% accuracy. In terms of execution time, both KNN and Naive Bayes were the fastest models with 1.00 ms. The slowest model was Random Forest with an execution time of 252.8 ms.

For the 70/30 split using 15 features, Logistic Regression achieved the highest accuracy of 97.7%, with an execution of 10.45 ms. SVM achieved the second-highest accuracy of 97.1%, with an execution time of 2.01 ms. KNN achieved the third-highest accuracy of 93.6%, with an execution time 0.99 ms. Decision Tree came fourth place with 92.4% accuracy, and Naive Bayes and Random Forest fifth with 91.8% accuracy. In terms of execution time, both KNN and Naive Bayes were fastest with 0.99 ms. The slowest model was Random Forest with an execution time of 240.0 ms.

For the 50/50 split using 15 features, Logistic Regression achieved the highest accuracy of 97.9%, with an execution time of 8.42 ms. SVM attained the second-highest accuracy of 97.2%, with an execution time of 1.96 ms, while KNN achieved the third-highest accuracy at 94.7%, with an execution time of 0.98 ms. Naive Bayes was fourth with 92.3%, followed by Random Forest with 91.9% and Decision Tree with 91.2% accuracy. In terms of execution time, KNN was the fastest model with 0.98 ms. The slowest model was Random Forest with an execution time of 203.7 ms.

For the 30/70 split using 15 features, Logistic regression achieved the highest accuracy at 96.5%, with an execution time of 5.99 ms. SVM achieved the second-highest accuracy at 96%, with an execution time of 1.01 ms. KNN achieved the third-highest accuracy at 94.2%, with an execution time of 0.99 ms. Decision Tree came was fourth with 92.2% accuracy, while Random Forest and Naive Bayes were fifth and sixth with accuracies of 91.7% and 90.7%, respectively. In terms of execution time, Naive Bayes and KNN were the fastest models with 0.99 ms. The slowest model was Random Forest with an execution time of 181.5 ms.

For the 20/80 split using 15 features, Logistic Regression achieved highest accuracy of 96.3%, with an execution time of 3.00 ms. SVM achieved the second-highest accuracy of 95.8%, with an execution time of 0.89 ms. KNN obtained third-highest accuracy of 93.4%, with an execution time of 0.78 ms. Random Forest was fourth with 91.4%, while Decision Tree and Naive Bayes fifth and sixth with accuracies of 89.7%, and 89.5%, respectively. In terms of execution time, KNN was the fastest model with 0.78 ms. The slowest model was Random Forest with an execution time of 107.9 ms.

By comparing all splits using 15 features, the highest accuracy is 97.9% obtained by Logistic Regression with 8.42 ms execution time and a 50/50 split. The lowest execution time is 0.78 ms obtained by KNN with an accuracy of 93.4% and a 20/80 split.

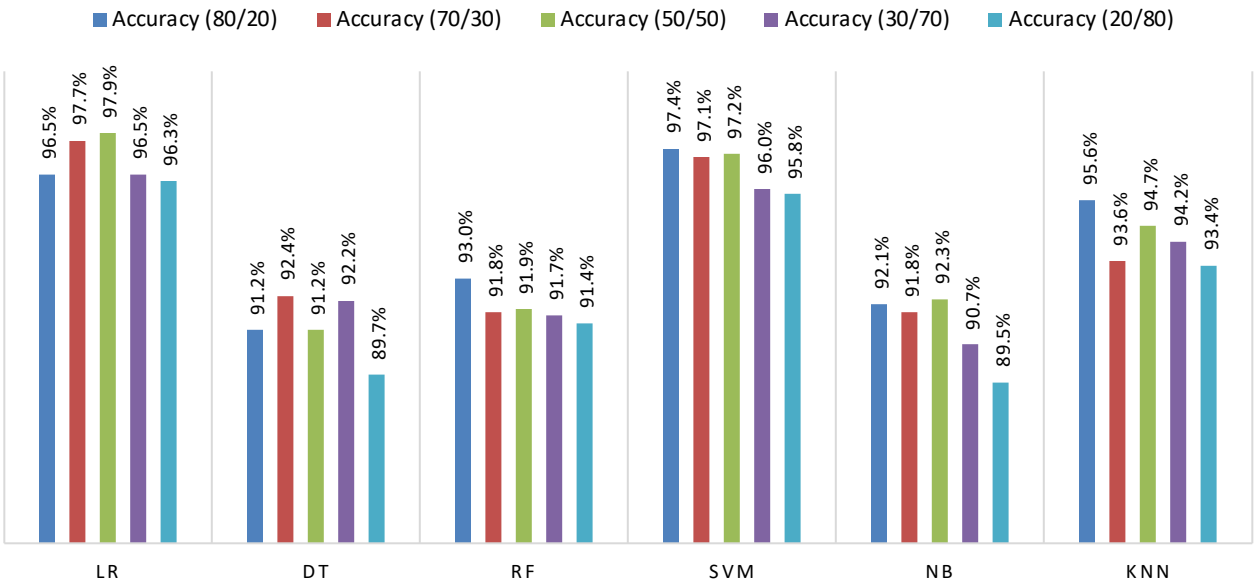

Figure 4.22.4 Machine learning model performance using 15 features.

Chapter 5 Conclusion and Future Work

This project considered breast cancer prediction using the supervised ML models, Logistic Regression, Decision Tree, Random Forest, SVM, Naive Bayes, and KNN. Model effectiveness was assessed using the metrics accuracy, precision, recall, F1-score, and execution time. To remove class imbalance, SMOTE was employed for dataset balancing, and PCA was used to reduce the number of features using eigenvalues and the cumulative variance ratio. Reduction to 2, 5, 10, and 15 components was considered. The performance of the models was evaluated using dataset split percentages 80/20, 70/30, 50/50, 30/70, and 20/80. The models were implemented using the Anaconda Jupyter notebook environment and the Python programming language. The results obtained show that SVM and Logistic Regression outperformed the other classifiers with SVM having an accuracy of 98.2%, and an execution time of 9.99 ms for an 80/20 split using 10 features, and Logistic Regression having an accuracy of 97.9%, and an execution time of 8.42 ms for an 50/50 split using 15 features. Moreover, Naive Bayes was the fastest model with an execution time of 0.67 ms at an accuracy of 90.4% for an 20/80 split using 10 features.

For future work, the proposed system can be utilized with datasets for other diseases. Furthermore, the feasibility of deploying the models in a real-world clinical setting can be assessed. This could involve healthcare professionals providing feedback on the practical applicability of the models and conducting performance evaluations.

References

[1] J. A. Farley, Cancer Statistics For The Year 2020: An Overview, In International Journal of Cancer, vol. 149, pp. 778-789, 2021.

[2] S. M. Winkler, M. Affeneller, S. Schaller, and H. Stekel, Data Based Prediction of Cancer Diagnoses Using Heterogeneous Model Ensembles: A Case Study for Breast Cancer, Melanoma, and Cancer in the Respiratory System, In Annual Conference on Genetic and Evolutionary Computation, pp. 1337-1344, 2014.

[3] A. Russo, L. Incorvaia, E. Capoluongo, P. Tagliaferri, S. Gori, L. Cortesi, M. Genuardi, D. Turchetti, U. De Giorgi, M. Di Maio, M. Barberis, M. Dessena, M. Del Re, A. Lapini, C. Luchini, BA. Jereczek-Fossa, A. Sapino, and S. Cinieri, Implementation of Preventive and Predictive BRCA Testing in Patients with Breast, Ovarian, Pancreatic, and Prostate Cancer: A Position Paper of Italian Scientific Societies, In Journal of European Society for Medical Oncology Open, vol. 7, 2022.

[4] C. Jie, The Whole View of Therapies for Breast Cancer, In International Conference on Biomedical and Bioinformatics Engineering, pp. 176-179, 2020.

[5] L. A. Qiu, B. Kanski, S. Doerksen, R. Winkels, and K. H. Schmitz, Nurse AMIE: Using Smart Speakers to Provide Supportive Care Intervention for Women with Metastatic Breast Cancer, In Computer Human Interaction Conference on Human Factors in Computing Systems, art. 302, 2021.

[6] S. J. Gardezi, A. Elazab, B. Lei, and T. Wang, Breast Cancer Detection and Diagnosis Using Mammographic Data: Systematic Review, In Journal of Medical Internet Research, vol. 21, no. 7, art. e14464, 2019.

[7] P. H. Abreu, M. S. Santos, M. H. Abreu, B. Andrade, and D. C. Silva, Predicting Breast Cancer Recurrence Using Machine Learning Techniques: A Systematic Review, In Journal of Association for Computing Machinery, vol. 49, art. 52, 2016.

[8] X. Wei, R. Yuan, J. Yang, W. Zhang, Y. Jin, M. Wang, J. Jiang, C. Wu, and K. Li. Effects of Baduanjin Exercise on Cognitive Function and Cancer-Related Symptoms in Women with Breast Cancer Receiving Chemotherapy: A Randomized Controlled Trial, In Journal of Supportive Care in Cancer, vol. 30, pp. 6079-6091, 2022.

[9] A. N. Giaquinto, H. Sung, K. D. Miller, J. L. Kramer, L. A. Newman, A. Minihan, A Jemal, and RL. Siegel, Breast Cancer Statistics, In A Cancer Journal for Clinicians, vol. 72, pp. 524-541, 2022.

[10] S. P. Shelomyanov, Applying Three Machine Learning Algorithms to Three Breast Cancer Diagnosis Datasets, In Journal of Computing Sciences in Colleges, vol. 35, pp. 272-274, 2020.

[11] T. A. Pham, Deepcare: A Deep Dynamic Memory Model for Predictive Medicine, In Pacific-Asia Conference on Knowledge Discovery and Data Mining, pp. 30-41, 2016.

[12] R. Karmakar, S. Chatterje, A. K. Das, and A. Mandal, Breast Cancer Prediction Using Machine Learning Approach—A Performance Analysis, In Journal of SN Computer Science, vol. 4, art. 377, 2023.

[13] V. Nemade, and V. Fegade, Machine Learning Techniques for Breast Cancer Prediction, In International Conference on Machine Learning and Data Engineering, vol. 218, pp. 1314-1320, 2023.

[14] O. Iparraguirre-Villanueva, A. Epifania-Huerta, C. Torres-Ceclen, J. Ruiz-Alvarado, and M. Canabillas-Carbonell, Breast Cancer Prediction Using Machine Learning Models, In International Journal of Advanced Computer Science and Applications, vol. 14, pp. 610-620, 2023.

[15] M. A. Botlagunta, M. D. Botlagunta, M. B. Myneni, D. Lakshmi, A. Nayyar, J. S. Gullapalli, and M. A. Shah, Classification and Diagnostic Prediction of Breast Cancer Metastasis on Clinical Data Using Machine Learning Algorithms, In Journal of Scientific Reports, vol. 13, art. 485, 2023.

[16] R. A. Rabiei, S. M. Ayyoubzadeh, S. Sohrabei, M. Esmaeii, and A. Atashi, Prediction of Breast Cancer Using Machine Learning Approaches, In Journal of Biomedical Physics & Engineering, vol. 12, pp. 297-308, 2022.

[17] K. A. Kourou, T. P. Exarchos, K. P. Exarchos, M. V. Karammouzis, and D. I. Fotiadis, Machine Learning Applications in Cancer Prognosis and Prediction, In Journal of Computational and Structural Biotechnology, vol. 13, pp. 8-17, 2014

[18] C. Mair, G. Kadoda, M. Lefley, K. Phalp, C. Schofield, M. Shepperd, and S. Webster, An Investigation of Machine Learning Based Prediction Systems, In Journal of Systems and Software, vol. 53, pp. 23-29, 2000.

[19] T. Jiang, J. L. Gradus, A. J. Rosellini, Supervised Machine Learning: A Brief Primer, In Journal of Behavior Therapy, vol. 51, pp. 675-687, 2020.

[20] S. A. Uddin, A. Khan, Md. E. Hossain, and M. A. Moni, Comparing Different Supervised Machine Learning Algorithms for Disease Prediction, In Journal of BMC Medical Informatics and Decision Making, vol. 19, art. 281, 2019.

[21] F. Y. Osisanwo, J. E. T. Akinsola, O. Awodele, J. O. Hinmikaiye, O. Olakanmi, and J. Akinjobi, Supervised Machine Learning Algorithms: Classification and Comparison, In International Journal of Computer Trends and Technology, vol. 48, pp. 128-138, 2017.

[22] L. Shao, H. Zhang, and G. de Haan, An Overview and Performance Evaluation of Classification-Based Least Squares Trained Filters, In IEEE Transactions on Image Processing, vol. 17, no. 10, pp. 1772-1782, 2008.

[23] Y.-F. Li, H.-W. Zha, and Z.-H Zhou, Learning Safe Prediction for Semi-Supervised Regression, In Association for the Advancement of Artificial Intelligence Conference on Artificial Intelligence, pp. 2217-2223, 2017.

[24] W.-T. Pan, C.-E. Huang, and C.-L. Chiu, Study on The Performance Evaluation of Online Teaching Using The Quantile Regression Analysis and Artificial Neural Network, In Journal of Supercomputing, vol. 72, pp. 789-803, 2016.

[25] K. A. Kourou, Machine Larning Applications in Cancer Prognosis and Prediction, In Journal of Computational and Structural Biotechnology, vol. 13, pp. 8-17, 2015.

[26] S. A. Ferro, D. Bottigliengo, D. Gregori, ASC. Fabricio, M. Gion, I. Baldi, Phenomapping of Patients with Primary Breast Cancer Using Machine Learning-Based Unsupervised Cluster Analysis, In Journal of Personalized Medicine, vol. 11, art. 272, 2021.

[27] V. Francois-Lavet, P. Hendersen, R. Islam, M. G. Bellemare, J. Pineau, An Introduction to Deep Reinforcement Learning, Foundations and Trends in Machine Learning, vol. 11, pp. 219-354, 2018.

[28] R. S. Sutton, Introduction: The Challenge of Reinforcement Learning, Springer International Series in Engineering and Computer Science, vol. 173, 1992.

[29] J. M. Perkel, By Jupyter, It All Makes Sense, In Nature, vol. 563, no. 7729, pp. 145-146, 2018.

[30] J. A. Wang, Better Code, Better Sharing: On The Need of Analyzing Jupyter Notebooks, In ACM/IEEE International Conference on Software Engineering: New Ideas and Emerging Results, pp. 53-56, 2020.

[31] F. A. Pedregosa, Scikit-Learn: Machine Learning in Python, In Journal of Machine Learning Research, vol. 12, pp. 2825-2830, 2011.

[32] W. A. McKinney, Pandas: A Foundational Python Library for Data Analysis and Statistics, 2011, https://api.semanticscholar.org/CorpusID:61539023.

[33] N. Ari and M. Ustazhanov, Matplotlib in Python, In International Conference on Electronics, Computer and Computation, pp. 1-6, 2014.

[34] A. H. Sial, S. M. Rashdi, and A. H. Khan, Comparative Analysis of Data Visualization Libraries Matplotlib and Seaborn in Python, In International Journal of Advanced Trends in Computer Science and Engineering, vol. 10, art. 38, pp. 277-281, 2021.

[35] P. Lemenkova, Processing Oceanographic Data By Python Libraries NumPy, SciPy and Pandas, In Journal of Aquatic Research, vol. 2, art. 19009 pp. 73-91, 2019.

[36] W. Wolberg, O. Mangasarian, N. Street, and W. Street, Breast Cancer Wisconsin (Diagnostic) Data Set, 1995, https://doi.org/10.24432/C5DW2B.

[37] A. A. Fernandez, SMOTE for Learning from Imbalanced Data: Progress and Challenges, In Journal of Artificial Intelligence Research, vol. 61, pp. 863-905, 2018.

[38] R. Blagus and L. Lusa, SMOTE for High-Dimensional Class-Imbalanced Data, In Journal of BMC Bioinformatics, vol. 14, art. 106, 2013.

[39] H. Abdi and L. J. Williams, Principal Component Analysis, In Wiley Interdisciplinary Reviews:Computational Statistics, vol. 2, pp. 433-459, 2010.

[40] L. A. Redmond, D. Foucambert, and L. Libersan, Language Corpora and Principal Components Analysis, In Springer Book of Applied Data Science, Studies in Big Data, vol. 125, pp. 117-132, 2023.

[41] W.-Y. Loh, Logistic Regression Tree Analysis, In Springer Handbook of Engineering Statistics, pp. 593-604, 2023.

[42] V. G. Costa and C. E. Pedreira, Recent Advances in Decision Trees: An Updated Survey, In Artificial Intelligence Review Journal, vol. 56, pp. 4765-4800, 2023.

[43] F. T. Liza, M. C. Das, P. P. Pandit, A. Farjana, A. M. Islam, and F. Tabassum, Machine Learning-Based Relative Performance Analysis for Breast Cancer Prediction, In IEEE Conference of World AI IoT Congress, pp. 7-12, 2023.

[44] Y. A. Wankhade, Machine Learning Approach for Breast Cancer Prediction: A Review, In International Conference on Applied Artificial Intelligence and Computing, pp. 566-570, 2023.

[45] G. Sajiv, G. Ramkumar, A Novel and Robust Breast Cancer Classification Based on Histopathological Images Using Naive Bayes Classifier, In International Conference on Artificial Intelligence and Knowledge Discovery in Concurrent Engineering, 2023.

[46] L. K. Singh, M. Khanna, and R. Singh, Artificial Intelligence Based Medical Decision Support System for Early and Accurate Breast Cancer Prediction, In Journal of Advances in Engineering Software, vol. 175, art. 103338, 2023.

[47] O. A. Risser-Maroix, and B. Chamand, What Can We Learn By Predicting Accuracy?, In IEEE/CVF Winter Conference on Applications of Computer Vision, pp. 2390-2399, 2023.

[48] E. J. Michaud, Z. Liu, and M. Tegmark, Precision Machine Learning, In Journal of Entropy and Information Studies, vol. 25, art. 175, 2023.

[49] P. Franti and R. Mariescu-Istodor, Soft Precision and Recall, In Journal of Pattern Recognition Letters, vol. 167, pp. 115-121, 2023.

[50] A. Humphrey, W. Kuberski, J. Bialek, N. Perrakis, W. Cools, N. Nuyttens, H. Elakhrass, and P. A. C. Cunha, Machine-Learning Classification of Astronomical Sources: Estimating F1-score in The Absence of Ground Truth, In Journal of Monthly Notices of The Royal Astronomical Society: Letters, vol. 517, pp. 116-120, 2022.

[51] M. Buchta, Estimation of Algorithm Execution Time Using Machine Learning, In Student Conference on Innovation, Technology and Science, 2023.