Coláiste na hOllscoile Corcaigh, Éire
University College Cork, Ireland

"Benchmarking Machine Learning Methodologies for Breast Cancer Diagnosis: A Comparative Approach"

Student Name (ID): Dilip Moralwar (122101142)

Supervisor: Dr. Tony Fitzgerald

Course: MSc. Data Science & Analytics

Module Code: ST6090 (Dissertation in Data Analytics)

Table of Contents

1. Introduction 5
   1.1 Aim of the Project 5
   1.2 Breast Cancer & Its Significance 6
   1.3 Dataset Description 8
   1.4 Literature Review / Related Works 8
2. Methodologies 10
   2.1 Data Pre-processing 11
   2.2 Splitting Data into Train and Test Dataset 12
   2.3 Data Visualization 12
   2.4 Splines 13
   2.5 Feature Selection / Choice of Attributes 14
      2.5.1 Recursive Feature Elimination 14
      2.5.2 Random Forest Feature Elimination 15
      2.5.3 Stepwise Selection 15
   2.6 Model Fitting 16
      2.6.1 Support Vector Machine 16
      2.6.2 Random Forest 17
      2.6.3 Logistic Regression 17
      2.6.4 k - Nearest Neighbour 18
      2.6.5 Extreme Gradient Boosting 18
   2.7 Performance Metrics 20
      2.7.1 Confusion Matrix 20
      2.7.2 Accuracy 21
      2.7.3 Precision 21
      2.7.4 Recall 21
      2.7.5 F1 Score 22
      2.7.6 Kappa 22
      2.7.7 AUROC 22
   2.8 Cross Validation 23
      2.8.1 K - Fold CV 23
      2.8.2 Leave One Out CV 24
      2.8.3 Stratified CV 24
3 Results 25
   3.1 Data Pre-processing 25

3.2 Data Visualization 26
3.2.1 Summary Statistics 26
3.2.2 Visualizing each variable 27
3.2.3 Correlation 31
3.3 Feature Selection 32
3.3.1 Recursive Feature Elimination 32
3.3.2 Refinement of Significant Features through Variable Selection Results 33
3.4 Splines 34
3.5 Train - Test Split using Caret 39
3.6 Model Fitting 39
3.7 Classifiers Performance / Performance Metrics 40
4 Discussion 43
4.1 Reasons for High Accuracy 43
5 Conclusion 46
6 Abbreviations 47
8 Bibliography 48
9 Appendices 50
9.1 R Script Explanation 50
9.2 R scripts 51

ABSTRACT

Major public health concern: breast cancer, has highlighted the value of early identification in order to enhance outcomes for treatment and likelihood of survival. Considering the limits of conventional diagnostic techniques, there is growing interest in using technological advancements, particularly machine learning, to facilitate better diagnosis. The Wisconsin Diagnostic Breast Cancer (WDBC) dataset is used in this study to explore the potential of machine learning approaches.[1] The purpose of the study is to investigate and compare the performance of different machine learning models for breast cancer diagnosis like Random Forest, XGBoost, SVM, etc. The process followed as pre-processing, data splitting, feature selection, model training and testing, and evaluation metrics like accuracy, precision, recall. Future research should investigate the use of advanced machine learning techniques to classify breast cancer in other populations and to develop more accurate and efficient classifiers.

1. Introduction

Breast cancer remains a dominant health challenge globally, sadly holding the title as the primary cause of fatalities from cancer among women. Although it has a large impact, low- to middle-income countries are most severely affected. Addressing this, the Global Breast Cancer Initiative has put forth a commendable goal: they aim to cut down breast cancer mortality rates by an impressive 2.5% every year. Achieving this would mean preserving approximately 2.5 million lives over a span of 20 years. [2]

Given its vast prevalence, it becomes imperative to detect and diagnose breast cancer at its earliest. Traditional diagnostic approaches, however, come with their set of challenges, often being resource-intensive both in terms of time and finances. This is where the promise of machine learning comes into the picture. Machine learning approaches can identify tiny patterns indicating breast cancer by using large datasets made up of clinical records and imaging scans. Once properly trained, these advanced algorithms stand to revolutionize how we identify and treat this condition.

The core of this project is the development of models that serve as the benchmark for the early detection of breast cancer. We plan to evaluate and comapre their performance rigorously, using a set of metrics that include accuracy, precision, recall and AUROC.

1.1 Aim of the Project

The primary aim of this study is to delve into and compare different machine learning techniques for breast cancer diagnosis. We're working with Breast Cancer Wisconsin dataset to develop prediction models that are both trustworthy and precise. With a focus on classifying cases of breast cancer as either malignant or benign, the study is rooted in the hope of aiding quicker detection and timely medical action, aiming to positively influence patient results. It is aimed to compare different machine learning techniques.

The research focuses mostly on developing and contrasting various machine learning approaches. The objective is to identify, using our dataset, the model that excels at correctly diagnosing breast cancer. This exploration involves identifying key data features that have a substantial impact on predictions and help us select the appropriate model and its parameters. The classification of breast cancer cases as benign or malignant is essential to this. We anticipate positive results by utilizing a number of machine learning classifiers.

1.2 Breast Cancer & Its Significance

Breast cancer, a grave health concern, arises from the cells within the breast. Predominantly affecting women, its early diagnosis is paramount for ensuring effective treatment and improved survival rates. The advent of machine learning provides an innovative method to diagnose breast cancer, utilizing a plethora of clinical and imaging features.[3]

Understanding Tumors

Tumors are abnormal growths resulting from uncontrolled cell division, and they can manifest in any part of the body. Depending on their nature, tumors can be categorized as benign, meaning they are noncancerous, or malignant, indicating they are cancerous.

Benign (Noncancerous) Tumors

Benign tumors are typically harmless growths that don't spread to surrounding tissues. They remain localized and closely resemble the nearby normal cells. While generally not life-threatening, they may require medical intervention if they:

- Compress neighboring tissues, nerves, or blood vessels
- Occupy crucial spaces like in the brain
- Damage adjacent structures
- Trigger excessive hormone production

Surgeons might consider excising benign tumors to prevent potential complications. Some benign tumors, though harmless, can grow considerably large if left untreated.

Malignant (Cancerous) Tumors

Malignant tumors are characterized by unbridled growth and the potential to invade adjacent tissues. Cells within these tumors differ significantly from their healthy counterparts. Common types of malignant tumors include:[3]

- Female Breast cancer
- Pulmonary cancer
- Colon cancer
- Adenocarcinoma of the prostate
- Carcinoma of the stomach

Some malignant tumors can spread, or metastasize, to distant body parts. For example, even though breast cancer starts in the breast, it can metastasize to vital organs, forming secondary tumors that retain characteristics of the primary tumor. Figure 1: Comparison of Malignant and Benign Breast Tumors.

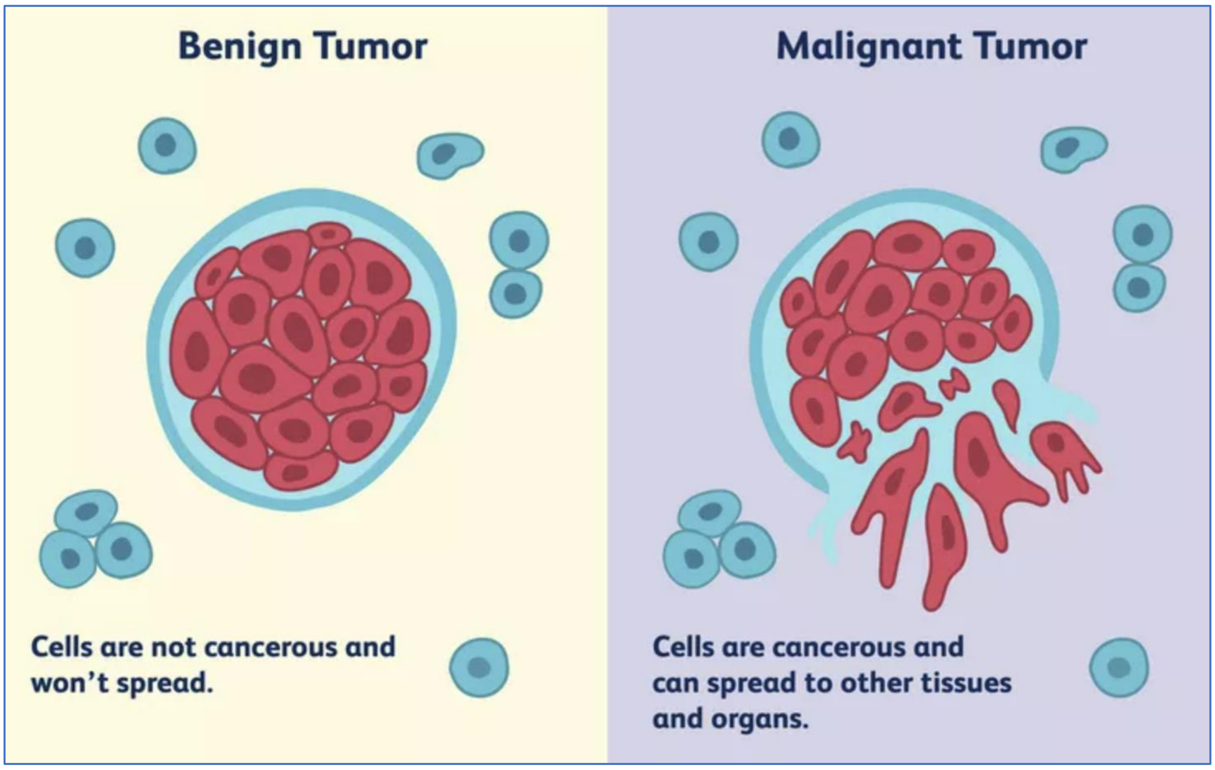

Benign Tumor
Malignant Tumor
Cells are not cancerous and won't spread.
Cells are cancerous and can spread to other tissues and organs.

Figure 1: Comparison of Malignant and Benign Breast Tumors. Adapted from [3], this image showcases the visual contrast between malignant (cancerous) and benign (non-cancerous) breast tumors.

Can a Benign Tumor Develop into Malignancy?

It's a rarity for benign tumors to transform into malignant ones. Nonetheless, certain benign growths, like colon polyps, might evolve into cancer if left unchecked. Early detection and removal of such growths, for example, during a colonoscopy, can be a preventive measure against the onset of cancer. Table 1 represents key differences between benign and malignant tumor.[3]

Benign Tumors	Malignant Tumors
Noncancerous in nature	Positively identified as cancerous
Do not infiltrate surrounding tissues	Can potentially invade neighboring tissues
Typically exhibit slow growth rates	Often demonstrate rapid, aggressive growth patterns
Manifest a smooth and regular shape	Display irregular and asymmetrical shapes
May not need treatment	Needs treatment

Table 1: Contrasting Benign and Malignant Tumors [3]

1.3 Dataset Description

In this research project, we have used the Breast Cancer Wisconsin (Diagnostic) Data Set. It is accessible from the UCI Machine Learning Repository to construct a predictive model for breast cancer diagnosis. The dataset was created by Dr. William H. Wolberg from the University of Wisconsin Hospital in Madison, Wisconsin.[1] It contains 569 instances, each of which is a description of a breast cancer cell. It comprises 357 benign or non-cancerous and 212 malignant or cancerous breast cancer.[4] The 30 features in the dataset are measurements of various properties of the breast cancer cell, such as the radius, texture, and smoothness of the cell nucleus. These characteristics were extracted from digital breast mass FNA photos.[5]

It's worth noting that this dataset has been meticulously curated, containing no instances with missing values. The dataset provides information about cell characteristics, including the following details:

- ID Number: A unique code for nucleus of each cell.
- Diagnosis: shows whether the cell is malignant (M) or benign (B).
- Features 3-32: Each cell nucleus has ten real-valued characteristics defined for it.
  The average, standard deviation, and "worst" value—the highest of the three highest values—of these attributes were calculated for each image. This data provides a thorough perspective of characteristics of cell nuclei in the dataset.[1]

The dataset was collected from a population of women who had undergone breast biopsies. The biopsies were performed at the University of Wisconsin Hospital in Madison, Wisconsin. The women were chosen for the study if they had a breast mass that was suspicious for cancer.

The prevalence of breast cancer in the population from which the dataset was collected is 38.4%. This means that 38.4% of the instances in the dataset are labeled as malignant. However, the prevalence of breast cancer in Ireland is about 1 in 8 women. This means that about 12.5% of women will develop breast cancer in their lifetime.[25]

The prevalence of breast cancer in the dataset is higher than the prevalence in the general population. This is because the dataset only includes women who had breast biopsies. Biopsies are more likely to be performed on women who are at higher risk of breast cancer, such as women who have a family history of breast cancer or who have dense breasts.

1.4 Literature Review / Related Works

In this section, some of the previous studies on breast cancer diagnosis using different machine learning approaches are reviewed. Breast cancer diagnosis has evolved significantly over the decades, leveraging technological advancements to improve early detection and treatment outcomes. Recent developments in machine learning have offered promising avenues in this domain.

Dr. Muhammet Ak provides a comparative analysis of the use of statistics and machine learning applications for breast cancer detection and diagnosis.[6] The author reviewed 21 studies that used these methods, and found that both approaches have their own advantages and disadvantages. It was concluded that the best approach for breast cancer detection and diagnosis is to use a combination of statistics and machine learning methods. This approach can take advantage of the strengths of both approaches and can help to improve the accuracy of diagnosis. The strengths of this paper are that it provides a comprehensive review of the literature on the use of machine learning for breast cancer detection and diagnosis. It also discusses the advantages and disadvantages of each technique and provides recommendations for future research. The limitations of this paper are that it only reviewed studies published between 2015 and 2020, and it did not provide any specific examples of how statistics and machine learning can be used together to improve the diagnosis of breast cancer.

Nitish Biswas et al. proposes an approach based on machine learning for the diagnosis of breast cancer [7]. The approach uses a feature optimization technique to select the most important features from a set of 30 features extracted from digitized breast images. The selected features are then used to train a random forest classifier. The authors concluded that their approach is a promising new method for the diagnosis of breast cancer. The approach is simple to implement and can be used with a variety of machine learning algorithms. The approach also has the potential to be used with other types of cancer. The strengths of the paper include approach which uses a feature optimization technique to select the most important features from a set of 30 features. This helps to improve the accuracy of the classifier. The approach was evaluated on a large and well-known dataset, the WBC dataset. This gives confidence in the results of the study. The approach was compared with other machine learning methods and was found to outperform them in terms of accuracy, precision, recall, and F1 score. The limitations are, it was evaluated only on the WBC dataset. It is important to evaluate the approach on other datasets to confirm the results. The approach was not compared with other feature selection techniques. It is important to compare the approach with other feature selection techniques to determine its effectiveness.

Vincent M. et al. presents a comparative study of machine learning classifiers for the prediction of breast cancer recurrences [8]. The authors also investigated the effect of feature selection on the performance of the classifiers. They found that feature selection can improve the performance of the classifiers, with an average improvement of 1.5%. The strengths of the paper are it presents a comprehensive study of machine learning classifiers for the prediction of breast cancer recurrences. The paper evaluates five different classifiers and investigates the effect of feature selection. The paper uses a large and well-known dataset, the WBC dataset. The limitations are that the paper only uses one dataset. It is important to evaluate the classifiers on other datasets to confirm the results. The paper does not consider the cost of false positives and false negatives. This is an important factor to consider when selecting a classifier for clinical use.

Dr. Marion Olubunmi Adebiyi et al. compared their model with other machine learning techniques, such as random forests and support vector machines (SVMs).[9] They found that their model performed better than the other models in terms of accuracy, precision, recall, and F1 score. The strengths are that the paper proposes a simple and effective machine learning model for the diagnosis of breast cancer. The model was trained on a large and well-known dataset, the WBC dataset. The model achieved high accuracy, precision, recall, and F1 score. The model was able to identify the most important features for breast cancer diagnosis. The limitations are that the paper only uses one dataset. It is important to evaluate the model on other datasets to confirm the results. The paper does not consider the cost of false positives and false negatives. This is an important factor to consider when selecting a model for clinical use.

Kourou, K et al. provides a review of the use of machine learning in cancer prognosis and prediction.[10] The authors found that machine learning can be used to predict the risk of cancer, the prognosis of cancer, and the response to treatment. The authors also found that machine learning can be used to identify patients who are at high risk of cancer recurrence. The authors state that machine learning has the potential to be a powerful tool for cancer prognosis and prediction. However, they also point out that there are a number of challenges that need to be addressed before this can be realized. These challenges include the lack of large, well-annotated datasets, the high dimensionality of cancer data, and the difficulty of interpreting machine learning models. The strengths of the paper are that it provides a comprehensive review of the literature on the use of machine learning in cancer prognosis and prediction. The paper discusses the different machine learning algorithms that have been used for this purpose, as well as the different types of cancer that have been studied. It provides an overview of the challenges and limitations of using machine learning for cancer prognosis and prediction. The limitations are that the paper is somewhat outdated, as it was published in 2015. The paper does not discuss the cost-effectiveness of using machine learning for cancer prognosis and prediction.

2. Methodologies

The analysis and development of machine learning models were executed using the R programming language, with the utilization of several essential libraries:

- **Caret**: The Caret package was used for essential processes such as train-test data splitting and cross-validation, assisting in model evaluation and selection.
- **E1071**: The E1071 package was utilized to implement Support Vector Machine (SVM) algorithms, enabling the creation of effective classification models.
- **randomForest**: The randomForest package was employed for constructing Random Forest models, which are ensemble learning methods known for their predictive accuracy.

- **xgboost**: The xgboost library was used to implement Extreme Gradient Boosting (XGBoost) algorithms, which are powerful for boosting and improving model performance.
- **ggplot2**: This library facilitated the creation of high-quality, customized data visualizations, contributing to insightful data exploration.
- **mgcv**: The mgcv package was potentially used for fitting Generalized Additive Models (GAMs) and splines.
- **ROCR**: ROCR might have been utilized for generating Receiver Operating Characteristic (ROC) curves and evaluating the performance of classification models.

The combination of these software tools and libraries enabled the development of effective machine learning models for breast cancer prediction, enclosing data manipulation, preprocessing, model building, evaluation, and visualization.

2.1 Data Pre-processing

Data preprocessing is a vital step in dataset preparation for analysis or modeling. It entails cleaning, transforming, and organizing raw data into a format that is appropriate for further processing. In the context of diagnosing breast cancer using machine learning, data preprocessing plays a critical role in ensuring accurate and reliable results.

Handling Missing Values:

Missing values in the dataset can impair the performance of machine learning models. Depending on the extent and nature of missing data, various strategies can be used:

- Imputation: Filling missing values with calculated estimates based on other data points.
- Removal: Removing rows or columns with substantial missing data, provided they do not contain critical information.
- Interpolation: Estimating missing values based on the trends present in the dataset.

Dealing with Outliers:

Outliers can adversely affect model training. Techniques include:

Trimming: Capping extreme values to a certain threshold.

Transformation: Applying mathematical functions to reduce the impact of outliers.

Removing Redundant Features:

Features that don't contribute significantly to the model's performance can be removed, reducing noise and improving model efficiency.

2.2 Splitting Data into Train and Test Dataset

Data splitting is a crucial machine learning technique to guarantee accurate model performance assessment. The dataset is split into training and test subsets. The test set is used to evaluate the model's predicted accuracy on fresh, unseen data, while the training set is used to train the model and help it discover patterns.

The requirement to accurately simulate real-world scenarios justifies the separation of training and testing. We measure the model's performance under circumstances akin to those it would experience when used in real-world scenarios by evaluating it on previously unexplored data (the test set). This division reduces the possibility that a model will simply memorize the training data and guarantees that it will be able to make precise predictions on novel, untried samples. Figure 2 depicts the overview of the train-test split.

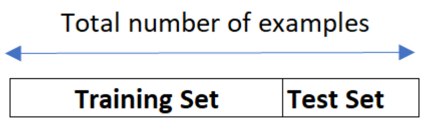


Figure 2: Training & test set

To sum up, data splitting is a critical machine learning phase. The performance of our models will be enhanced by dividing the data into two sets and preventing overfitting.

2.3 Data Visualization

An important phase of exploratory data analysis is data visualization. To better understand the dataset's insights and trends, a variety of plots and graphs are used to visualize it. Common visualization tools like histograms, scatter plots, and bar plots have been employed in this research to comprehend the distribution and connections between various characteristics.[11]

Data is often displayed in rows and columns in tables, which are tabular representations of data. A particular illustration of the distribution of numerical data is called a histogram. It is employed to display the frequency of data within particular ranges or bins. The term "bar chart" refers to a visual representation of categorical data using rectangular bars, where each bar's length is inversely proportional to the value it represents. On a graph, a scatter plot shows the relationship between two variables as points in a two-dimensional coordinate system. Data is graphically represented as a heatmap, where each value is represented by a color. It is frequently used to show how closely two variables are correlated.[11]

2.4 Splines

R's gam() function from the "mgcv" package uses the function s() to define a smooth term. A thin-plate regression spline (TPRS) is the foundation for modeling smooth terms in the "mgcv" s() function. A spline that can capture both linear and nonlinear interactions between the predictor variable and the response variable is called a TPRS. [12] These splines are isotropic, therefore even if the covariate coordinate system is rotated, the smoothing result doesn't change. They also have low rank, which indicates that their coefficients are much smaller than the amount of data points being smoothed.

The formula for a thin plate regression spline is typcally expressed as follows:[13]

$$f(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_k x_k + \sum_{j=1}^{n} f_j(x)$$

Where:

- $f(x)$ is the estimated function you want to find.
- βi's are the coefficients to be estimated for each predictor variable xi's
- n represents the number of data points.
- $f_j(x)$ is the thin plate regression spline for each data point j as a function of predictor variables x.

One benefit of thin-plate regression splines is that they have a lowered rank, which makes calculation quick and lowers the possibility of overfitting. To achieve rank reduction, the thin plate spline penalty is employed, and a truncated eigen-decomposition is used rather than explicitly defining knots as is done in other spline approach.

Instead of depending on a set or user-defined number of knots, we allow the package to automatically decide the number and placement of knots based on the data and the chosen smoothness penalty. The ability to adjust to the intricacy of the relationship between the predictor and the response variable is provided by this behavior. This method can be helpful because it gets rid of the necessity for subjective and difficult human knot selection. The automatic knot selection procedure looks for the best balance between overfitting and model flexibility. This is accomplished by estimating the smooth function using a penalized regression technique, where the smoothness penalty aids in regulating the spline's complexity. Based on the smooth term's effective degrees of freedom (edf) and the stated penalty, the package chooses the right number of knots. The number of knots for thin plate regression splines can be roughly calculated as the edf + 1.[12]

The model is flexible enough to adjust to various data patterns and determine the optimal number of knots to reflect the underlying link between the predictor and the response variable.

2.5 Feature Selection / Choice of Attributes

Variable or feature selection is a pivotal phase in refining datasets for optimal analysis and modeling. Its purpose is to identify the most influential attributes that significantly impact the diagnostic variable. This process aids in reducing dataset complexity and enhancing model performance. Several techniques can be used for variable selection:

2.5.1 Recursive Feature Elimination

Recursive Feature Elimination (RFE) operates as a feature selection technique by systematically eliminating the least influential attributes from a dataset. The importance of each feature is determined by its contribution to the model's predictive power. Employing a backward elimination approach, RFE initiates with all dataset features and progressively discards the least essential ones, iteratively. This process endures until the specified number of features remains.

RFE can be applied across various supervised learning algorithms, although it finds its common use in linear models like logistic regression and linear discriminant analysis. The preference for RFE in these models is attributed to their interpretability, and RFE's capacity to pinpoint the most pivotal attributes.

Recursive feature elimination (RFE) is a feature selection method that can be used to select the most important features from a dataset. RFE works by iteratively removing features that are least important to the model. The Random Forest algorithm is an integrated learning technique that generates numerous decision trees and sums their classification predictions.[14]

Benefits of using RFE:

- Reducing the dataset's dimensionality can help machine learning models perform better, which is one of the advantages of employing RFE. A dataset with fewer dimensions is simpler such that it can be used to train the model and make predictions about.

- Finding redundant or irrelevant features, which can make machine learning models easier to understand.
- Selection of a subset of features that are most important for a particular task, which can improve the generalization performance of machine learning models. Features that are unnecessary or redundant do not affect the model's predictions, so they can be removed without affecting the model's performance. The most crucial features will be used to train the model, increasing the likelihood that it will generalize better to new data.

Limitations of using RFE:

- RFE's performance might be affected by the base model selection.
- RFE can be computationally expensive, especially for large datasets.
- RFE has a tendency to favor highly correlated features.

2.5.2 Random Forest Feature Elimination

The ensemble learning technique Random Forest can also be used to choose features. It determines the significance of each feature depending on how accurate the model becomes without that feature.

Random Forest Feature Elimination (RFFE) utilizes a random forest model to recognize the most crucial attributes in a dataset. The process involves an iterative removal of the least significant features and retraining the random forest model. This continues until the desired number of features is retained.

RFFE falls under the category of wrapper methods, which utilize an estimator (here, a random forest model) to select features. While being computationally intensive compared to filter methods, wrapper methods tend to be more effective in pinpointing essential attributes. Because it is reasonably simple to use and can be implemented in a number of programming languages, RFFE is a well-liked feature selection technique. Furthermore, as a flexible approach, It can be used to solve a variety of machine learning problems, including regression, clustering, and classification.

2.5.3 Stepwise Selection

Stepwise selection is a feature selection technique that progressively includes and excludes features based on their predictive significance. It involves two modes: forward selection (addition of features one by one) and backward elimination (removal of features one by one).

Either a model that contains no characteristics (ahead stepwise) or a model that contains all features (reverse stepwise) is the starting point for stepwise selection. Depending on how features affect the prediction performance of the model, it gradually adds or removes them. This process is pursued until further feature additions would no longer enhance the model's predictive capability. Stepwise selection is relatively straightforward to grasp and implement. However, its computational demands can escalate with larger datasets. The choice of the initial model can also affect its performance.[15]

In summary, these feature selection techniques contribute to the optimization of datasets, improvement of model interpretability, and enhancement of model predictions within the scope of breast cancer diagnosis using machine learning.

2.6 Model Fitting

In our dataset, the outcome variable, also known as the dependent variable (Y), exhibits only two distinct values: M (Malignant) or B (Benign). As a result, we will employ a classification algorithm within the realm of supervised learning to predict and categorize these outcomes. The used machine learning methods are described hereafter and the corresponding R scripts are given in the appendix.

Figure 3 illustrates the overview of classification breast cancer predictive model.

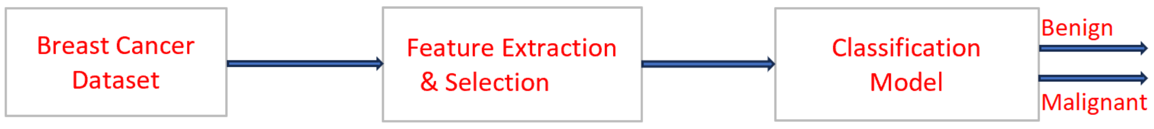


Figure 3: Breast Cancer Classification Model Overview

For the purpose of diagnosing breast cancer, several machine learning classification models can be applied. The following models were utilized in this project:

2.6.1 Support Vector Machine

Support Vector Machine (SVM) is a machine learning technique that can be used for classification tasks. It determines the ideal hyperplane decision boundary for classifying data points into groups. [16] SVM aims to minimize the distance between the nearest data points in each class and the hyperplane. With the help of kernel function, it can manage non-linear relationships between features and classes by mapping data into a higher-dimensional space. SVM is effective in handling high-dimensional data, robust against overfitting, and can work well with small training samples. It is widely used in various domains for tasks such as image recognition and text classification.

2.6.2 Random Forest

Both classification and regression tasks can be accomplish using the machine learning technique called as random forests. By building numerous decision trees and pooling their forecasts, they operate. Overfitting is avoided by training each decision tree on a different random subset of the data. Random forests are renowned for their reliability, capacity to handle large-scale data, and precision. [17] They are extensively utilized in many different fields, such as image recognition, bioinformatics, and finance.

2.6.3 Logistic Regression

Logistic regression is a statistical modeling method used to predict the likelihood of a binary outcome, such as whether or not a patient has a disease. It does this by fitting a linear model to the data, but the predicted probabilities are transformed using the logistic function. This ensures that the predicted probabilities fall between 0 and 1.[18]

Logistic regression is a widely used technique because it is simple to understand and interpret, and it can handle both categorical and continuous variables. It can also be extended to deal with multi-class classification problems. However, it is important to keep in mind that logistic regression assumes linearity between the predictors and the outcome. This means that the relationship between the predictors and the outcome can be modeled as a straight line. In some cases, this assumption may not be met, and the model may not be accurate.

Logistic regression can be extended to include non-linear functions of the predictors by using splines. Splines are piecewise polynomials that can be used to fit a smooth curve to the data. This can help to improve the accuracy of the model in cases where the assumption of linearity is not met. Logistic regression with splines is an example of a generalized additive model (GAM). GAMs are a type of regression model that can be used to fit a non-linear relationship between the response variable and the predictors.

$logit(p(y=1|x)) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_k X_k$ [15]

where: [15]

- *logit(p(y=1/x))* is the log odds of y being equal to 1, given the values of the predictors X1, X2, ..., Xk.

- β_0, β_1, β_2,..., β_k are the coefficients of the linear regression model.
- *X1, X2,...,Xk* are the predictors.

2.6.4 k- Nearest Neighbour

It is possible to use the non-parametric machine learning technique K-nearest neighbors (KNN) for both regression and classification problems. It operates by identifying the k training set instances that are most similar to a new instance. Based on the labels of the k closest neighbors, the new instance's label is then predicted. [19]

The user must select the value of the hyperparameter k. While a big value of k will make the algorithm more resilient to noise but less accurate, a little value of k will make the algorithm more sensitive to noise.

KNN is a straightforward algorithm to comprehend and use, but it has a lot of potential in the real world. It frequently serves as a benchmark algorithm for assessing how other machine learning algorithms perform.

2.6.5 Extreme Gradient Boosting

A machine learning technique called XGBoost combines a number of ineffective prediction models to produce a better and more precise model.[20] It is renowned for its remarkable scalability and predictive performance. With the gradient boosting framework used by XGBoost, new models are created to fix the mistakes produced by older ones. Additionally, it employs a sophisticated optimization approach to reduce the loss function and produce more accurate forecasts. Since it can handle a wide range of data types, including numerical and categorical features, XGBoost is extremely adaptable. Due to its capacity to manage big datasets, handle missing values, and efficiently capture complicated relationships in the data, it is widely used in a various field.

The pros and cons of different machine learning algorithms are summarised in table 2. [21][26]

Sr. No.	ML Algorithm	Advantages	Limitations
1	Random Forest (RF)	RF reduces the risk of overfitting compared to Decision Trees, as it averages outcomes from constituent trees.	Increased complexity and computational demands.
		Empirically, this ensemble classifier often outperforms individual Decision Trees.	Requires setting the number of base classifiers.
		Scales well for large datasets.	Biased towards variables with high value variety for variable importance.
		Can identify important variables in classification.	Prone to overfitting.
2	Support Vector Machine (SVM)	SVM is more resilient than Logistic Regression (LR).	Computational intensity for large, complex datasets.
		Effective in handling diverse feature spaces.	Underperforms with noisy data.
		Lower overfitting risk.	Model outcomes and variable impacts can be intricate.
		Suited for classifying semi-structured data (text, images, etc.).	Limited to binary classification without extensions.
3	Logistic Regression (LR)	Simplicity and ease of implementation.	Accuracy suffers when input variables have intricate relationships.
		Dynamic updates to LR-based models.	Ignores complex variable correlations.
		No presumptions about independent variable distribution.	Prone to overconfidence in logic models.
		Features probabilistic interpretation of model parameters.	Possible exaggeration of prediction accuracy due to sample bias.
			Restricted to binary classification, unless expanded.
4	K-nearest neighbour (KNN)	Simple algorithm with fast instance classification	Computational demands increase with attribute count.
		Tolerant towards noisy instances and missing attributes.	Attributes are uniformly weighted, causing weak classification in some cases.
		Applicable in classification and regression tasks.	Lacks insight into most effective attributes for good classification.
5	Extreme Gradient Boosting (XGBoost)	High efficiency and scalability, often outperforming alternatives in accuracy.	Hyperparameter tuning is essential.
		Manages missing data and feature interactions effectively.	Sensitivity to noisy datasets.
		Built-in regularization curbs overfitting.	Longer training times compared to simpler algorithms.
		Supports parallel processing.	Some algorithms offer better interpretability.
		Proficient in classification and regression.	Categorical variables require feature engineering.

Table 2: Pros and Cons of Different Machine Learning Algorithms [21][26]

There are many papers that have compared the performance of the various models as a diagnostic tool for breast cancer. Dr. Muhammet Ak (2020) [6] compared the performance of four machine learning algorithms, namely support vector machines (SVMs), random forests, decision trees, and k-nearest neighbors (KNN), on a dataset of breast cancer histopathological images.

2.7 Performance Metrics

The suggested methodology uses five classification algorithms to predict breast cancer: Support Vector Machines, Random Forest, Logistic Regression, K-Nearest Neighbors, and XGBoost. The prediction is enhanced by using these classifiers. These classifiers' effectiveness is assessed based on their accuracy, precision, recall, and F1 measure.

2.7.1 Confusion Matrix

A table that lists a classification model's performance is known as a confusion matrix. It demonstrates the proportion of cases that were successfully categorized (true positives and true negatives) and the proportion of instances that were mistakenly categorized (false positives and false negatives).[22]

Accuracy, precision, recall, and F1 score are just a few of the evaluation metrics that may be computed using the confusion matrix. These measures shed light on the trade-offs between true positives and true negatives, false positives and false negatives, and the model's capacity to distinguish between the two groups. Positive is represented by 1 and negative by 0:

	Actual 0	Actual 1
Predicted 0	True Negative (TN)	False Negative (FN)
Predicted 1	False Positive (FP)	True Positive (TP)

- TN: Accurately classified number of negative cases.

  The percentage of instances where a benign (non-cancerous) tumor was properly predicted by the model to be benign.

- TP: Accurately classified positive case count.

  The proportion of instances where the model properly identifies a tumor as being malignant (cancerous).

- FN: The proportion of positive cases that were mistakenly labeled as negative.

  The number of instances that a dangerous tumor was misdiagnosed as a benign one by the model. In other words, the model is unable to appropriately recognize a malignant tumor.

- FP: The proportion of incorrectly classified negative cases as positive.

  The number of instances where a benign tumor is misdiagnosed as malignant by the model. In other words, the model misclassifies a benign tumor as malignant.

  Sensitivity: is a measure of a model's accuracy in detecting breast cancer instances. With a high sensitivity value, the model is capable of identifying breast cancer even in trace amounts. As a result, there are fewer false negatives, or instances of breast cancer that the model misses.

  Specificity: How well a model can accurately detect instances of non-cancerous tissue is measured by its specificity. A high specificity score indicates that the model is effective at differentiating between non-cancerous tissue and breast cancer. As a result, there are fewer false positives, or instances where non-cancerous tissue is mistakenly diagnosed as breast cancer.

2.7.2 Accuracy

A machine learning model's accuracy is a straightforward metric that expresses the proportion of test instances that are properly identified. It is calculated by dividing the total number of test cases by the number of test cases that were correctly categorised.[22]

Accuracy is equal to (TP+TN)/(TP+TN+FP+FN)

2.7.3 Precision

The precision of a model's accurate positive predictions is measured. It is derived by dividing the total number of positive predictions by the number of real positives. The model performs better at recognizing the positive class when the precision is higher.

Precision is equal to TP / (TP + FP)

2.7.4 Recall

Recall, often referred to as sensitivity or true positive rate, is the percentage of real positives that are accurately classified as such. It is calculated by dividing the total of true positives and false negatives by the number of true positives.

Recall is TP / (TP + FN)

2.7.5 F1 Score

The F1 score is calculated as the harmonic mean of precision and recall, which means that it gives equal weight to both metrics. This makes the F1 score a useful measure of accuracy in cases where both precision and recall are important.[22]

F1 Score is equal to 2 * ((precision + recall) / (precision + recall))

2.7.6 Kappa

This is another performance metric that is not affected by class imbalance. It is based on the agreement between different raters.

2.7.7 AUROC

The true positive rate (TPR) vs the false positive rate (FPR) at various categorization criteria is plotted graphically to form a ROC curve. A binary classifier's overall performance is gauged by the area under the ROC curve (AUC). An improved classifier's ability to distinguish between the two classes is indicated by a higher AUC.

Figure 4 shows plot of AUROC Classification Evaluation Metric.

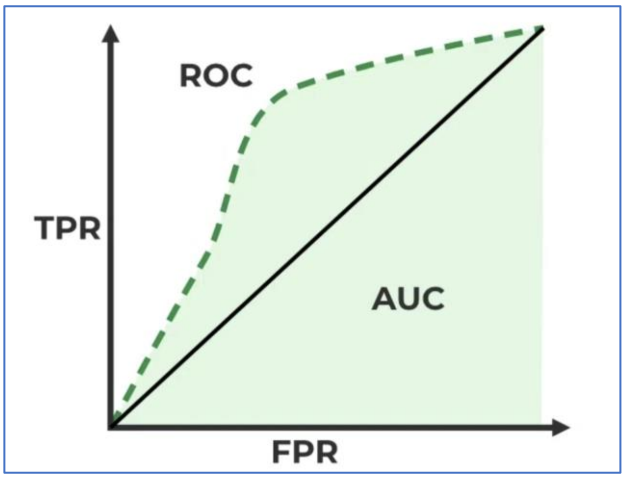


Figure 4: AUROC Classification Evaluation Metric [23]

An area under the curve (AUC) of 0.5 often denotes that the model is unable to distinguish between the two groups. An AUC of 0.5 to 0.7 is regarded as satisfactory, and an AUC of 0.7 or more as excellent. An AUC of 0.95 or greater is regarded as excellent for medical diagnosis models, nevertheless.[24]

2.8 Cross Validation

In statistical modeling and machine learning, cross-validation is a resampling approach used to evaluate a model's performance and reduce overfitting. It entails dividing the given data into numerous subsets, or "folds," in order to train and assess the model numerous times. The procedure aids in determining how well the model will perform on hypothetical data and offers a more trustworthy evaluation of its generalization potential.

Cross-validation is a helpful method for choosing a model, fine-tuning hyperparameters, and evaluating how well machine learning models generalize. It aids professionals in making decisions about a model's suitability for use with unseen data.

Benefits of Cross-Validation:

- Offers a more trustworthy estimation of model performance, particularly when the dataset is modest.
- By evaluating the model on various independent test sets, the risk of overfitting is decreased.
- Aids in hyperparameter adjustment as the model is tested on various data sets.

Common types of cross-validation:

2.8.1 K- Fold CV

The dataset is partitioned into k number of subgroups, and the model is trained and tested k times, using each subset as the validation set once. This process is known as k-fold cross-validation.

A sort of cross-validation called K-fold cross-validation (K-CV) is used to assess how well a machine learning model is working. When using K-CV, the dataset is divided into k folds, after which the model is trained on k-1 folds, and its performance is assessed on the held-out fold. The outcomes are averaged after this process has been carried out k times. Since K-CV does not require training the model k times on the complete dataset, it is a more effective method of model evaluation than LOOCV. K-CV does not evaluate the model using all of the dataset's samples, hence it is less thorough than LOOCV in this regard.

2.8.2 Leave One Out CV

Each data point is utilized as a test set in the Leave-One-Out Cross-Validation (LOOCV) method, which trains the model on all other data points. This is the same as performing k-fold cross-validation, where k is the same as the number of data points.

Cross-validation techniques like leave-one-out cross-validation (LOOCV) are employed to assess how well a machine learning model is performing. When using LOOCV, the model is first trained on all but one of the dataset's samples before being assessed on the held-out sample. Every sample in the dataset goes through this procedure once more, and the results are averaged.

LOOCV employs every sample in the dataset to evaluate the model, making it a very thorough method of model evaluation. LOOCV, however, can be computationally costly, particularly for large datasets.

2.8.3 Stratified CV

Cross-validation techniques like stratified cross-validation (SCV) are employed to stop machine learning models from being overfit. When a model learns the training data too well and is unable to generalize to new data, this is known as overfitting. By ensuring that the folds in the cross-validation are balanced with regard to the target variable, SCV helps to prevent overfitting.

In order for SCV to be effective, the data must first be stratified. This indicates that the target variable is dispersed equally among the folds. If the target variable, for instance, contains two classes, then each fold will have an equal number of samples from each class.

The model is trained on the folds after the data has been stratified, and its performance is assessed using the held-out folds. The results are averaged after this process is done a number of folds.

Overfitting can be avoided with the help of a powerful method i.e., SCV. It is notably helpful for models that are trained using data when the target variable's distribution is skewed.

3 Results

Data pre-processing is a critical initial step to ensure that the dataset is suitable for analysis. It involves addressing missing values and formatting the data appropriately. In this project, the dataset is examined for any missing values and processed accordingly. Additionally, unnecessary columns that do not contribute to the analysis (e.g., "id" column) are removed.

For building the machine learning model, first, we have to segregate feature and target variables. Figure 5 indicates overall workflow of machine learning process.

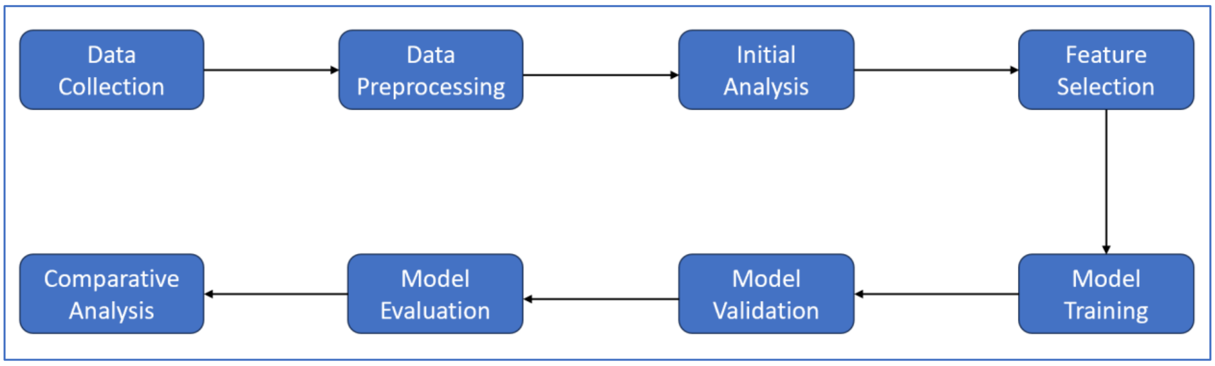


Figure 5: The workflow of a machine learning process

3.1 Data Pre-processing

Pre-processing data is a crucial first step in preparing a dataset for analysis. It includes filling in any missing numbers and correctly formatting the data. The dataset in this project is inspected for any missing values, then processed as necessary. Additionally, irrelevant columns that do not advance the analysis are eliminated, such as the "id" column.

We must first separate the feature and target variables before we can begin to build the machine learning model.

The data is pre-processed to improve the dataset's insights. The dataset could contain imbalances, errors, and omissions, which would reduce the accuracy of the findings. Therefore, the dataset needs to be cleaned before fitting the machine learning model. In this dataset, the diagnosis columns M and B represents malignant and benign, respectively. These textual data are converted to the numeric values 0 and 1, with 0 representing malignant tissue and 1 representing benign tissue.

We intend to improve the dataset's quality and dependability by implementing a complete data pre-processing technique, laying the groundwork for the project's further phases.

3.2 Data Visualization

In this section, each variable is visualized using bar plots, scatterplots, and histograms. Additionally, a table is used to summarize the statistics of the dataset.

3.2.1 Summary Statistics

To determine if the variables are skewed or symmetric, we can compare the mean and median values and assess the distribution of quartiles. If the mean and median are approximately equal and the quartiles are symmetrically distributed, then the variable can be considered symmetric. If the mean and median differ significantly, or if the quartiles are not symmetric, then the variable may be skewed. There are 569 cases in this dataset overall, 357 of which are benign and 212 of which are malignant. Table 2 indicates Summary of cell characteristics.

Cell Characteristics	Mean	Min.	Percentile			Max.
			25th	50th	75th	
Diagnosis						
Radius Mean	14.12	6.98	11.7	13.37	15.78	28.11
Texture Mean	19.29	9.71	16.17	18.84	21.8	39.28
Perimeter Mean	91.97	43.79	75.17	86.24	104.1	188.5
Area Mean	654.9	143.5	420.3	551.1	782.7	2501
Smoothness Mean	0.096	0.052	0.086	0.095	0.105	0.163
Compactness Mean	0.104	0.019	0.064	0.092	0.130	0.345
Concavity Mean	0.088	0	0.029	0.061	0.130	0.426
concave points Mean	0.048	0	0.020	0.033	0.074	0.201
Symmetry Mean	0.181	0.106	0.161	0.179	0.195	0.304
Fractal Dimension Mean	0.062	0.049	0.057	0.061	0.066	0.097
Radius SE	0.405	0.111	0.232	0.324	0.478	2.873
Texture SE	1.216	0.360	0.833	1.108	1.474	4.885
Perimeter SE	2.866	0.757	1.606	2.287	3.357	21.98
Area SE	40.33	6.802	17.85	24.53	45.19	542.2
Smoothness SE	0.007	0.001	0.005	0.006	0.008	0.031
Compactness SE	0.025	0.002	0.013	0.020	0.032	0.135
Concavity SE	0.031	0	0.015	0.025	0.042	0.396
Concave Points SE	0.011	0	0.007	0.010	0.014	0.052
Symmetry SE	0.020	0.007	0.015	0.018	0.023	0.078
Fractal Dimension SE	0.003	0.0008	0.002	0.003	0.004	0.029
Radius Worst	16.27	7.93	13.01	14.97	18.79	36.04
Texture Worst	25.68	12.02	21.08	25.41	29.72	49.54
Perimeter Worst	107.26	50.41	84.11	97.66	125.4	251.2
Area Worst	880.6	185.2	515.3	686.5	1084	4254
Smoothness Worst	0.132	0.071	0.116	0.131	0.146	0.222
Compactness Worst	0.254	0.027	0.147	0.211	0.339	1.058
Concavity Worst	0.272	0	0.114	0.226	0.382	1.252
concave points Worst	0.114	0	0.064	0.099	0.161	0.291
Symmetry Worst	0.290	0.156	0.250	0.282	0.317	0.663
Fractal Dimension Worst	0.083	0.055	0.071	0.080	0.092	0.207

Table 2: Summary of cell characteristics

3.2.2 Visualizing each variable

To assess whether features exhibit skewness or symmetry, a useful approach is to analyze their histograms rather than solely relying on mean and median comparisons. Histograms provide a visual representation of data distribution, allowing us to understand the underlying distribution more intuitively.

Histograms provide insights into the spread, shape, and central tendencies of data. A symmetrical distribution is often characterized by a balanced, bell-shaped histogram, where the data clusters around the center. On the other hand, skewed distributions exhibit a longer tail on one side, indicating the direction of skewness.

By observing histograms, you can identify whether data is right skewed (positively skewed) or left skewed (negatively skewed), or if it maintains a relatively symmetrical pattern. Histogram analysis enables a more holistic understanding of data distribution, helping to make informed decisions about whether transformations or adjustments are necessary based on the shape of the distribution.

The distribution of the cell characteristics is described in below figure 6.1 to 6.5.

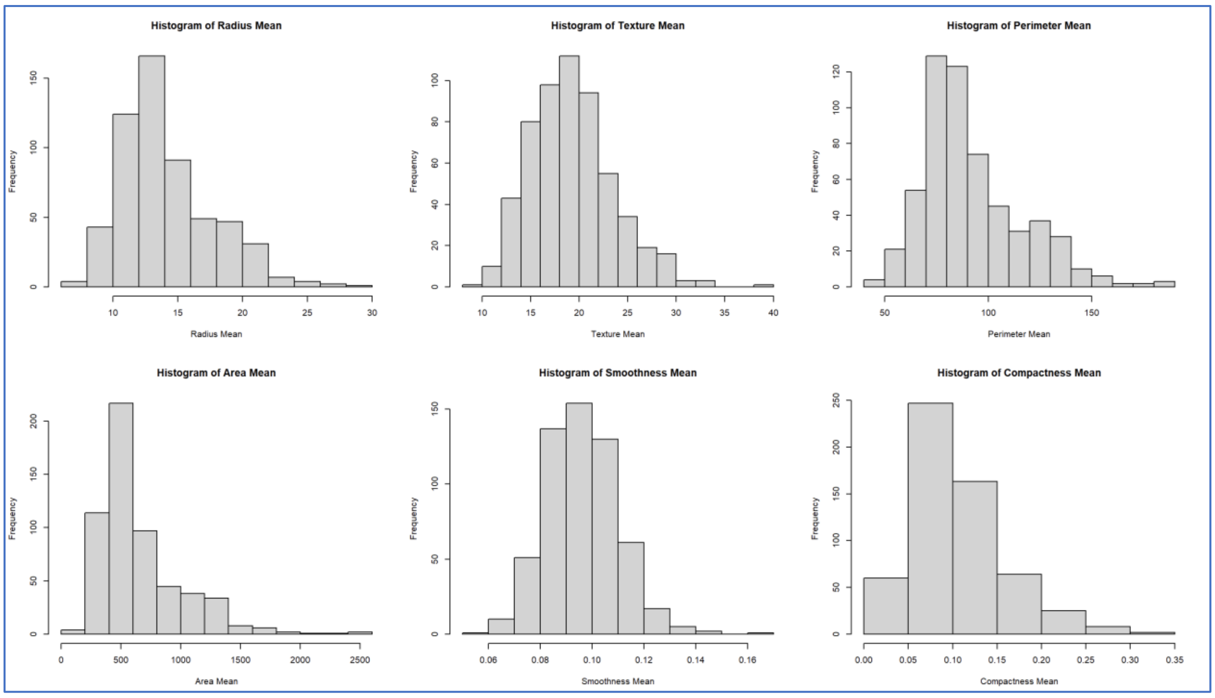


Figure 6.1 Histograms of cell characteristics

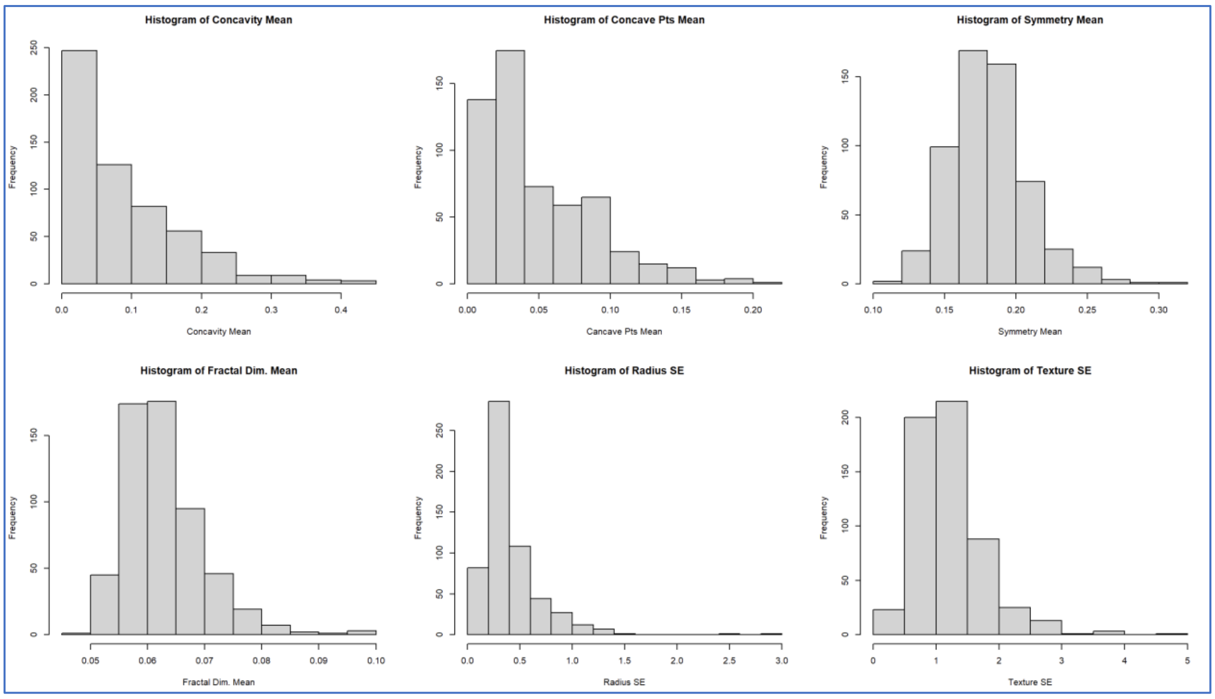


Figure 6.2 Histograms of cell characteristics

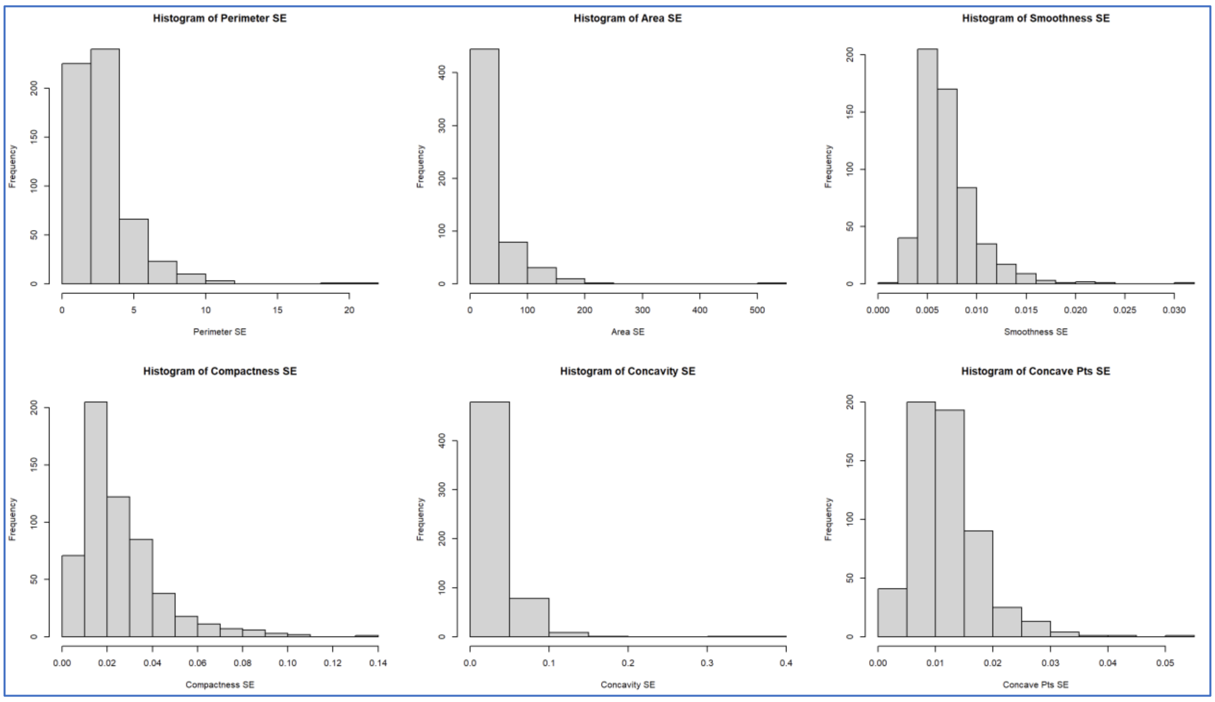


Figure 6.3 Histograms of cell characteristics

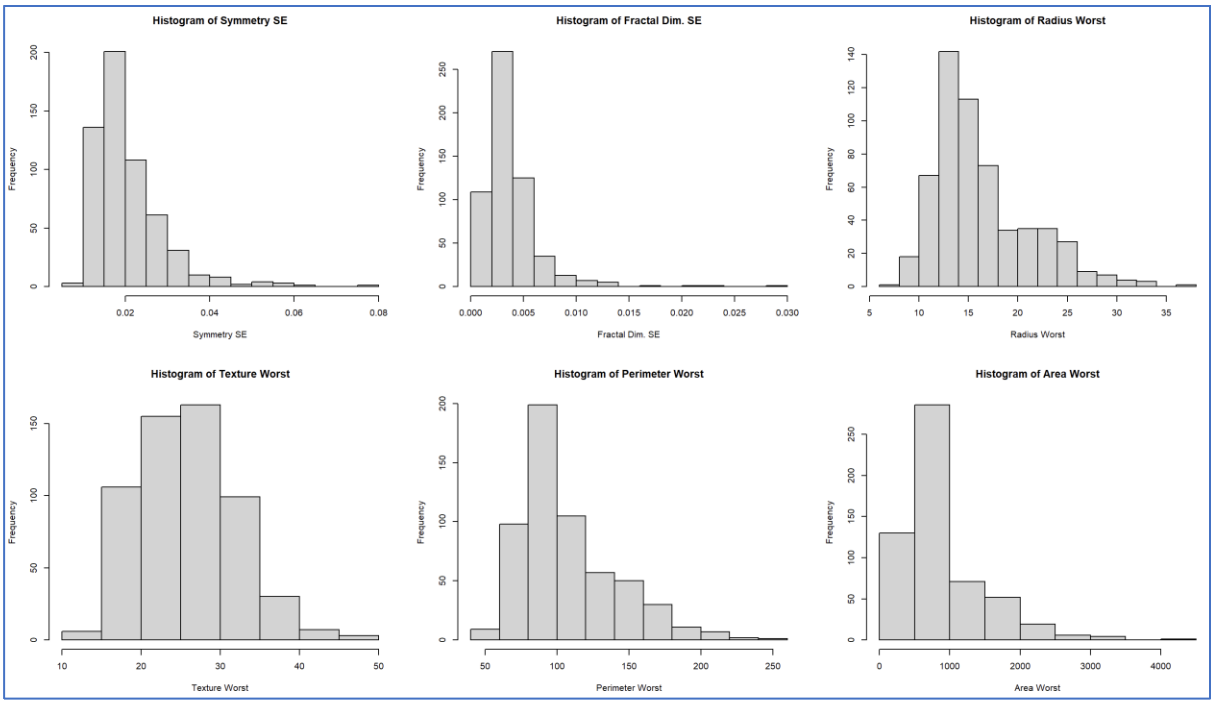


Figure 6.4 Histograms of cell characteristics

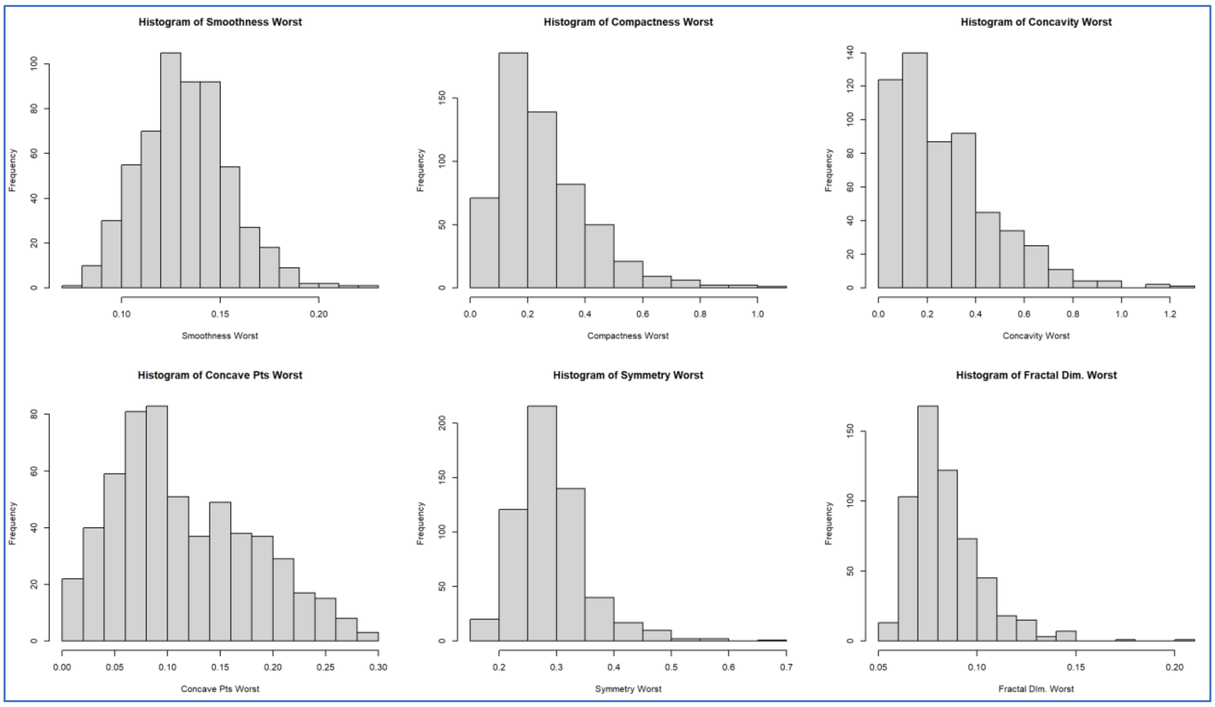


Figure 6.5 Histograms of cell characteristics

The analysis of the dataset features and their distributions yielded the following insights:

1. Most features exhibit a positive skew, indicating that their distributions are shifted towards higher values.

2. Features like Texture Mean, Smoothness Mean and Smoothness Worst exhibit symmetric distributions, as evidenced in the figure and by similar mean and median values.

3. Rest all features show a right skew as evidenced in the figure and exhibit differences between mean and median.

In summary, the analysis shows that certain features have approximately symmetric distributions, while others are skewed. The determination is based on the histogram analysis. This provides valuable intuition into the nature of the dataset's features, helping us understand their underlying distributions and potential implications for subsequent analysis.

3.2.3 Correlation

Figure 7 depicts the Correlation Plot between cell characteristics

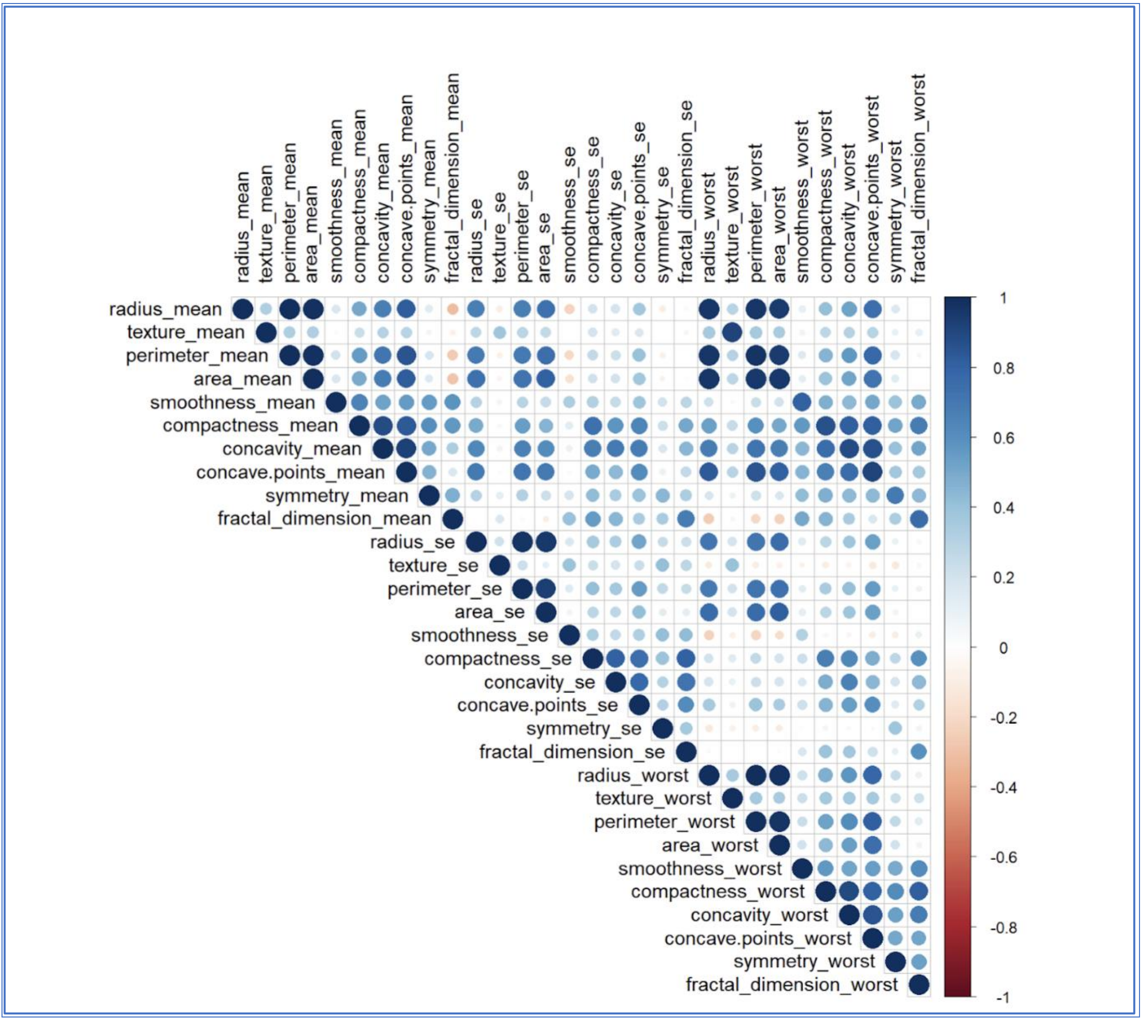


Figure 7: Correlation Plot between cell characteristics

The correlation plot gives valuable intuitions into the relationships within the cell characteristics of the dataset. The strongest correlations were observed between 'Radius Mean' and Perimeter Mean', indicating a significant positive relationship. Additionally, Area Mean' and ‘Radius Mean’ showed a high positive correlation. On the other hand, the weakest correlations were observed between ‘Fractal Dimension SE’ and ‘Perimeter Mean’ as well as ‘Smoothness Mean' and ‘Texture Mean’, indicating a minimal negative relationship.

Overall, the correlation plot helps us identify which cell characteristics are closely related and which are less related, providing crucial insights for further analysis and interpretation of the dataset.

3.3 Feature Selection

Recursive feature selection (RFE) has been used after data exploration on a clean dataset.

3.3.1 Recursive Feature Elimination

Random Forest is an algorithm for handling datasets with correlated features and non-linear associations.

Correlated Features: Random Forest is robust to correlated features because it constructs number of decision trees using various subsets of features and randomization. Every tree is trained independently, which helps in reducing the impact of correlated features. When building a tree at each node, Random Forest randomly picks a subset of features to consider for splitting. This randomness further aids in breaking the correlation patterns and ensures that no single feature dominates the model.

Non-Linear Associations: Random Forest can capture non-linear associations between features and the outcome. It does so by creating complex decision boundaries using the combination of multiple decision trees. These trees work together to capture intricate relationships between variables, enabling the algorithm to learn and make accurate predictions for complex data patterns.

In case of our study, Recursive Feature Selection (RFE) was executed using the Random Forest algorithm. RFE is a methodology intended to extract the most crucial features from a given dataset. This process is achieved by iteratively adding or removing features based on their significance. The Random Forest algorithm, renowned for constructing number of decision trees and amalgamating their predictions for accurate classifications, was employed to implement RFE.

RFE was executed using the following steps:

- The dataset was split into 10 folds for cross-validation to ensure robustness of the feature selection process.
- RFE was performed for subset sizes ranging from 1 to 30 features, encompassing all possible combinations of features.
- Random Forest models were trained on each subset and evaluated using performance metrics.
- The process was repeated for each fold of the cross-validation, and the average performance was calculated.

The RFE analysis revealed the following insights:

- The highest accuracy achieved was 96.66% with a corresponding Kappa value of 0.9289 when using a subset of 12 features.
- The performance of the model generally improved as more features were added, with a few exceptions where the accuracy slightly decreased.
- The accuracy and Kappa values were used as evaluation metrics to assess the effectiveness of each subset size.

3.3.2 Refinement of Significant Features through Variable Selection Results

The selected features based on the RFE analysis are as follows:

1. Perimeter Worst
2. Area Worst
3. Concave Points Worst
4. Radius Worst
5. Concave Points Mean
6. Area Mean
7. Area SE
8. Texture Mean
9. Concavity Worst
10. Texture Worst
11. Smoothness Worst
12. Concavity Mean

These characteristics were found to significantly help in the classification of breast cancer diagnoses. They offer insightful understandings of the underlying patterns and traits of the dataset. A subset of 12 features was found by the Recursive Feature Selection analysis utilizing the Random Forest technique to have the highest accuracy and kappa values. This suggests that these characteristics are essential for an accurate diagnosis of breast cancer and have the greatest capacity of discrimination. For upcoming classification tasks, the chosen features can be used to create a prediction model that will help with breast cancer early diagnosis and treatment.

Based on their Mean Decrease Gini values, the attributes are presented in the table in decreasing order of importance. The more significant a feature is in the categorization process, the greater the Mean Decrease Gini value. Table 3 indicates feature importance in decreasing order.

Feature	Mean Decrease Gini
Perimeter Worst	29.76
Concave Points Worst	28.56
Radius Worst	24.03
Area Worst	17.69
Concave Points Mean	18.35
Concavity Mean	11.63
Area Mean	6.51
Concavity Worst	8.71
Area SE	4.79
Texture Mean	3.94
Texture Worst	3.65
Smoothness Worst	2.45

Table 3: Key Feature Importance

The relative value of each attribute in the random forest model for categorizing tumors is indicated by these rankings. The more significant a feature is in the categorization process, the greater the Mean Decrease Gini value. These rankings suggest that the perimeter worst, concave points worst, and radius worst are the most important features for classifying tumors in the random forest model. This is because these features have the greatest impact on the Gini impurity of the data and are important for distinguishing between malignant and benign tumors.

3.4 Splines

Splines are used to identify and model non-linear relationships. Spline plots are graphical representations that show the relationship between two variables using smoothed curves, which are often created using spline functions. In our case, these plots (Fig 8.1 to 8.4) are used to illustrate the association between the log-odds of malignancy (a measure of the likelihood of a cell being malignant) and individual cell characteristics (features of the cells that might be indicative of malignancy). This helps in understanding which specific cell characteristics might contribute to a higher or lower likelihood of malignancy.

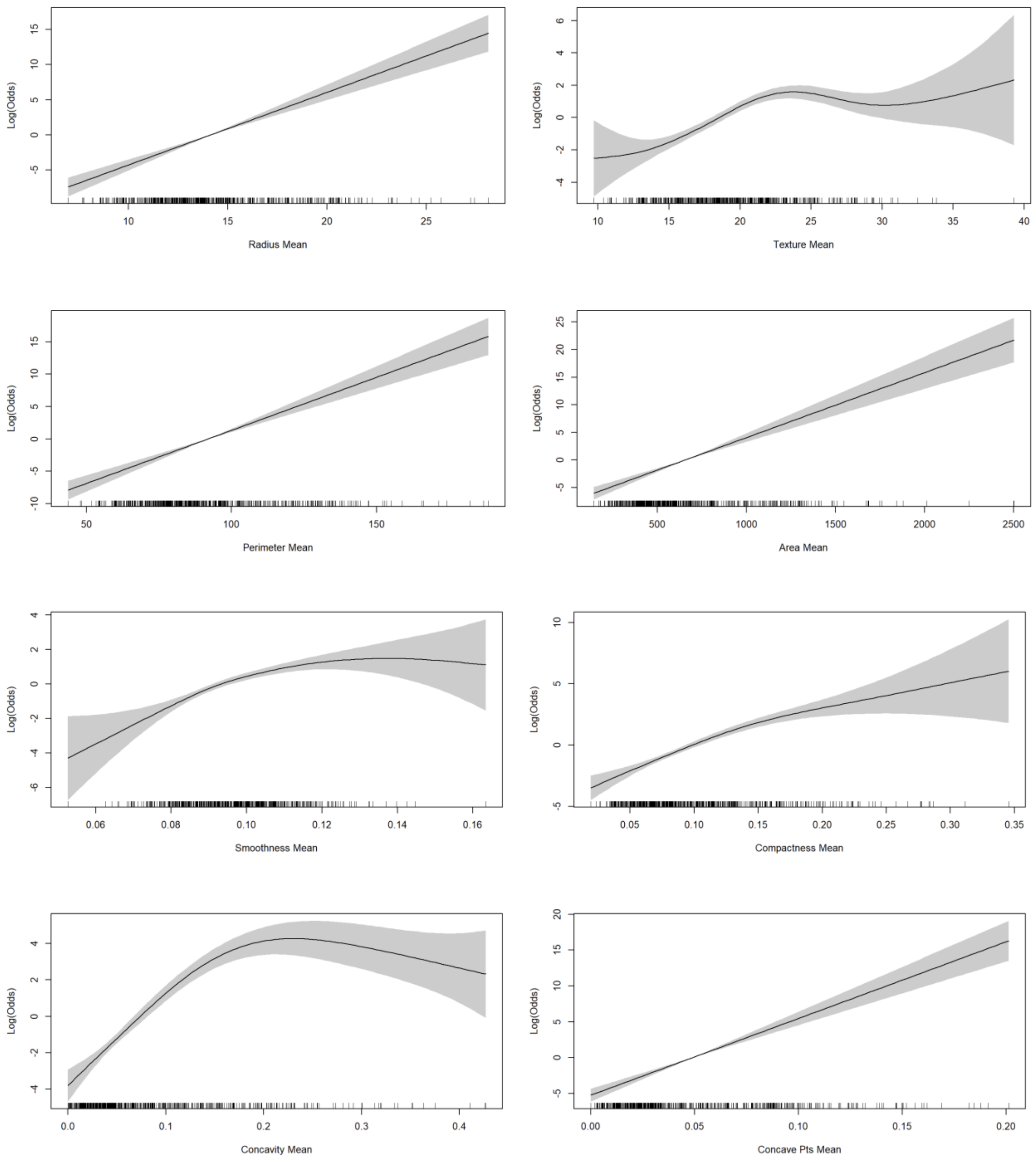


Figure 8.1: Spline plots illustrating association between log-odds malignancy and individual cell characteristics.

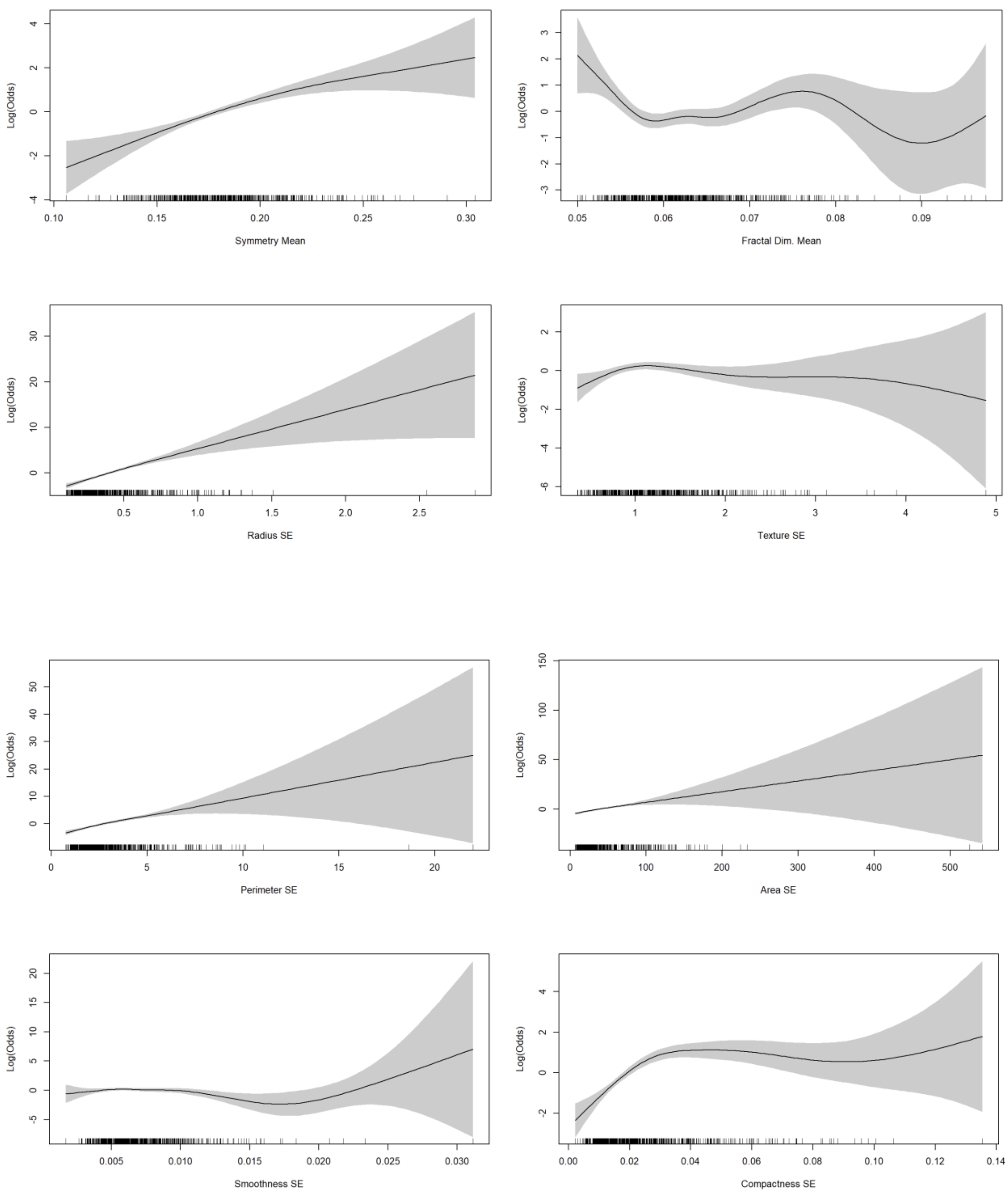


Figure 8.2: Spline plots illustrating association between log-odds malignancy and individual cell characteristics.

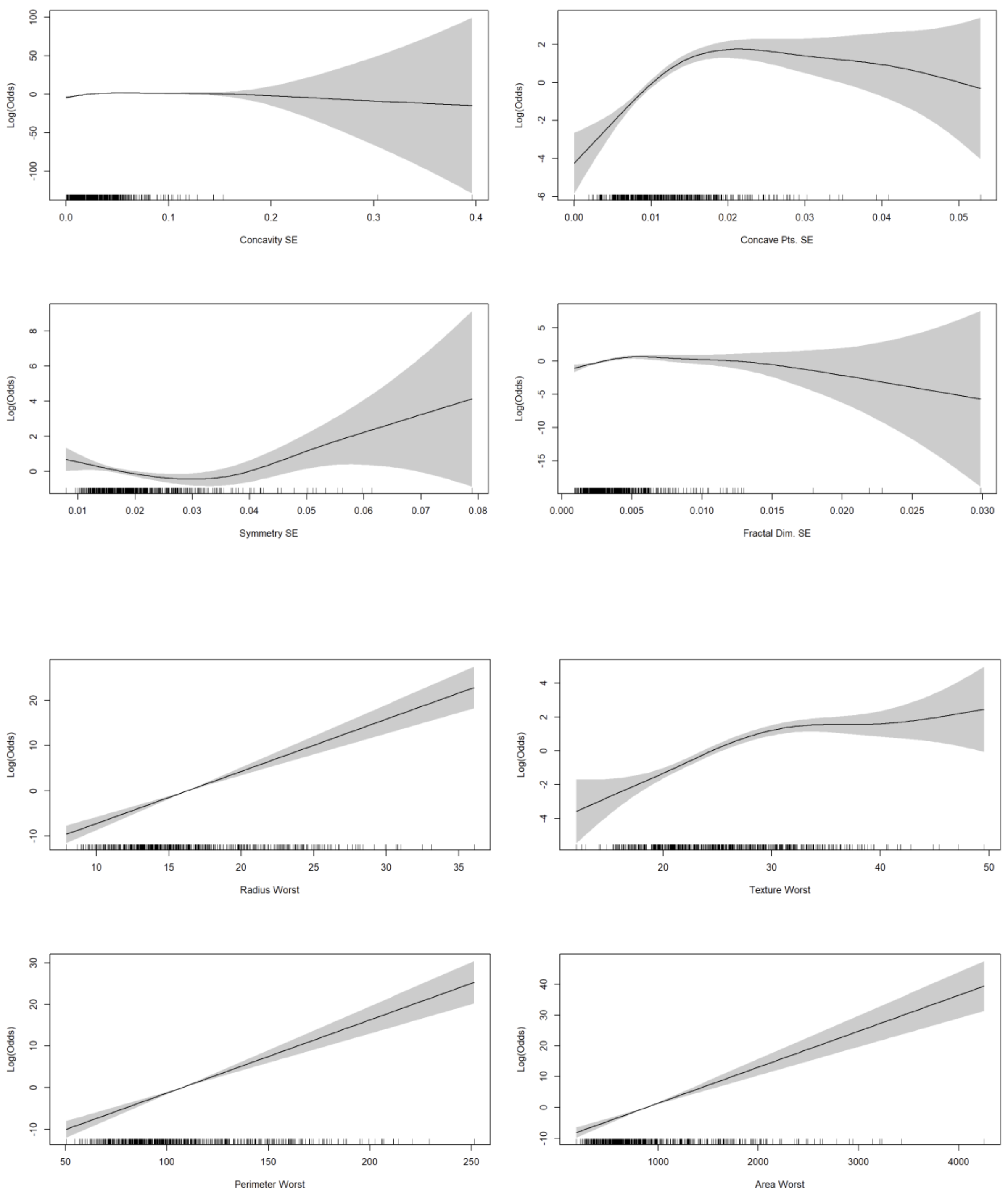


Figure 8.3: Spline plots illustrating association between log-odds malignancy and individual cell characteristics.

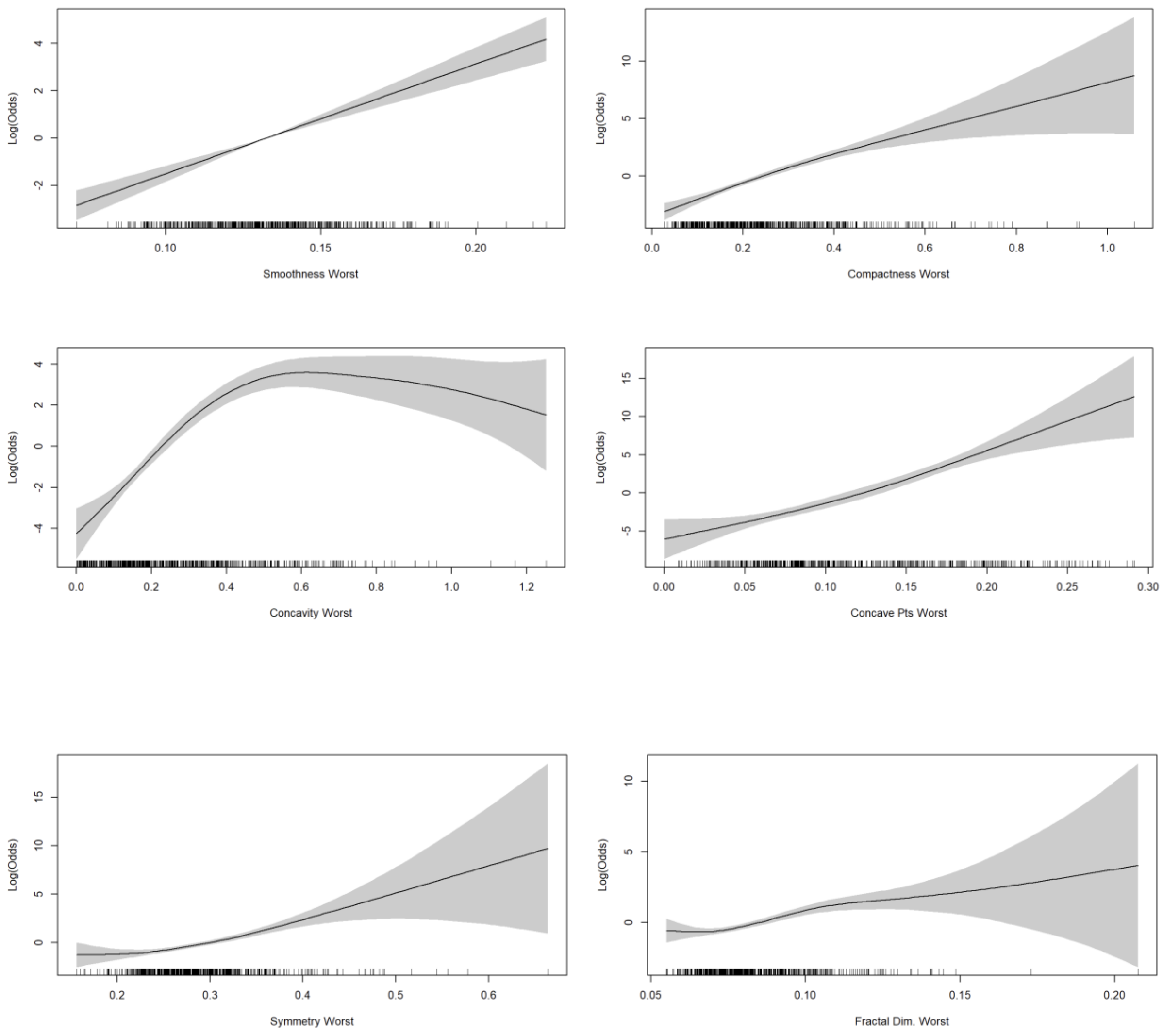


Figure 8.4: Spline plots illustrating association between log-odds malignancy and individual cell characteristics.

The dashed lines on the plot represent the estimated relationship between log-odds of malignancy and the cell characteristics. These lines are smoothed to provide a clear picture of the overall trend. The 95% confidence intervals around these lines indicate the range within which the true relationship is likely to fall with 95% confidence. In other words, they give us an idea of the uncertainty associated with the estimated relationship.

In our analysis, we have utilized thin-plate regression splines to model the intricate relationship between each individual cell characteristic and the probability of cell malignancy. The figure provided visually represents the log-odds of malignancy as it relates to these individual characteristics.

Notably, our findings reveal that certain cell characteristics exhibit non-linear associations with the log-odds of malignancy. The specific characteristics where non-linearity is prominent include Texture mean, Smoothness mean, Concavity mean, Fractal dimension mean, Texture SE, Smoothness SE, Compactness SE, Concave Points SE, Symmetry SE, Fractal Dimension SE, Texture Worst, Concavity Worst, Symmetry Worst, and Fractal Dimension Worst.

This observation underscores the complexity of the relationships between these features and malignancy. The use of thin-plate regression splines is particularly apt for capturing such non-linear associations, as they allow for more flexible modeling compared to traditional linear methods.

The non-linearity in the tails of the distribution of cell characteristics is likely due to the fact that there are fewer observations in these regions. This can be a problem for logistic regression, as this model assumes a linear relationship between the response and dependent variables. To address this problem, use a different model, such as a generalized linear model (GLM). GLMs are more flexible than logistic regression and can accommodate non-linear relationships.

3.5 Train- Test Split using Caret

To improve model performance, data splitting is a regular method in machine learning. The training set and the test set are two separate sets that make up the dataset. The test set is used to evaluate the model's performance, while the training set is used to train the model.

Overfitting is avoided by using data splitting. Overfitting happens when a model learns training data too completely and is unable to generalize to fresh input. Due to the data being split into two sets, the model is only tested on the test set and evaluated on the training set. The model can be kept from overfitting the training set of data thanks to this. The 80/20 split of the data is a standard procedure. This indicates that the model is trained using 80% of the data, and tested using 20% of the data. Because it offers a good balance between training the model and assessing its effectiveness, this split is frequently utilized.

For this study, we divided the training and test sets at a ratio of 4:1. This indicates that the model was trained using 80% of the data, and tested using the remaining 20%. This split was selected because it strikes a nice mix between training the model and assessing its effectiveness on a smaller sample. In machine learning, the data splitting procedure is a crucial stage. We can prevent overfitting and enhance the effectiveness of our models by dividing the data into two sets.

3.6 Model Fitting

We have employed six distinct machine learning models - Support Vector Machine with Radial Basis Function Kernel (SVM), Random Forest (RF), Logistic Regression (LR), k-Nearest

Neighbors (k-NN), and XGBoost - all used with the selected features to predict the diagnosis of breast cancer. Non-linear relationships are often modeled using splines. Support vector machines (SVM), random forests, and XGBoost are examples of machine learning models that can naturally handle and model complex nonlinear relationships found in the data without explicitly introducing splines.

3.7 Classifiers Performance / Performance Metrics

We conducted an extensive evaluation of five machine learning classifiers. The classifiers were trained on selected features to predict breast cancer diagnosis. The evaluation's findings are shown in Table 4, which shows the performance of each classifier on a variety of evaluation criteria, including accuracy, precision, recall, F1 score, and AUC ROC. Table 4 represents the Performance of various Classifiers.

Evaluation Criteria	**Classifiers**				
	SVM	**RF**	**LR**	**k-NN**	**XGBoost**
Accuracy (%)	97.35%	99.12%	96.46%	91.15%	97.35%
Precision	0.9535	0.9767	0.9318	0.8636	0.9535
Recall	0.9762	1.00	0.9762	0.9048	0.9762
F1 Score	0.9647	0.9881	0.9535	0.8838	0.9647
AUROC	0.974	0.993	0.967	0.9101	0.9746

Table 4: Classifiers Performance

The table shows that the performance of the classifiers is very similar, with all of them achieving high accuracy, precision, recall, F1 score, and AUC ROC. This suggests that all of the classifiers are capable of accurately classifying the data. There are a few small differences in the performance of the classifiers. RF has the highest accuracy, precision, F1 score, recall and AUC ROC. k-NN has lower accuracy, precision, recall, F1 score, and AUC ROC that rest all. It's crucial to remember that accuracy may not be ideal choice if the dataset is very unbalanced. AUC ROC may be more important to take into account in this situation.

Overall, all of the classifiers perform well and are capable of accurately classifying the data. The choice of which classifier to use may depend on the specific application and the desired trade-off between accuracy and other factors, such as computational complexity.

Figure 9 shows AUROC plots for various classifiers.

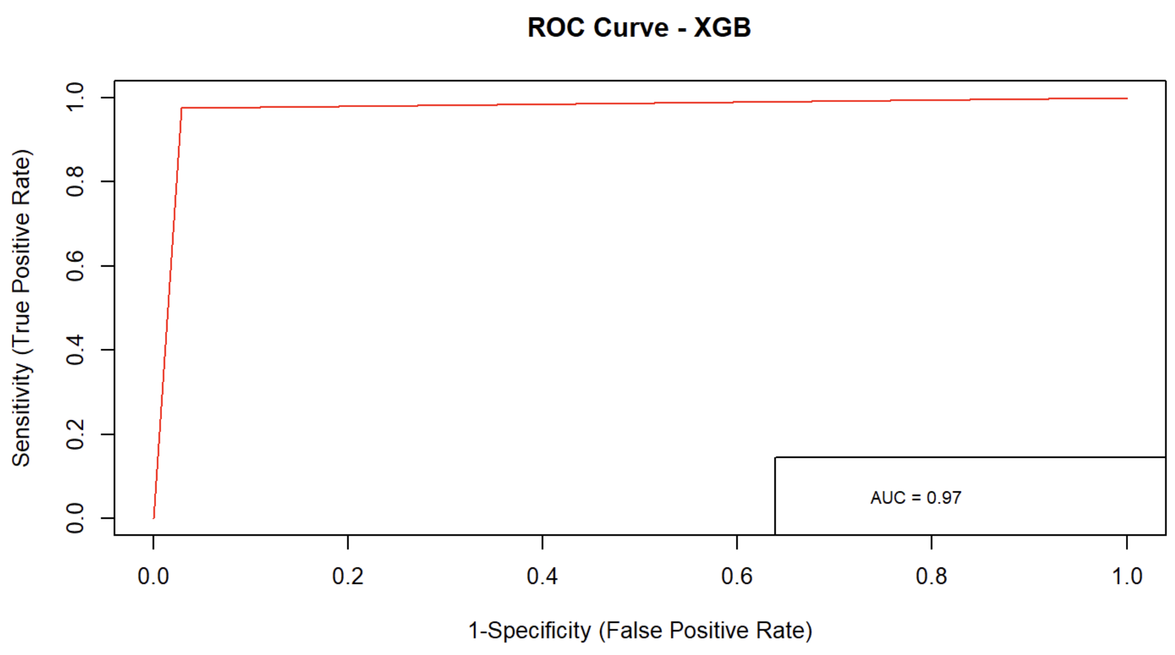
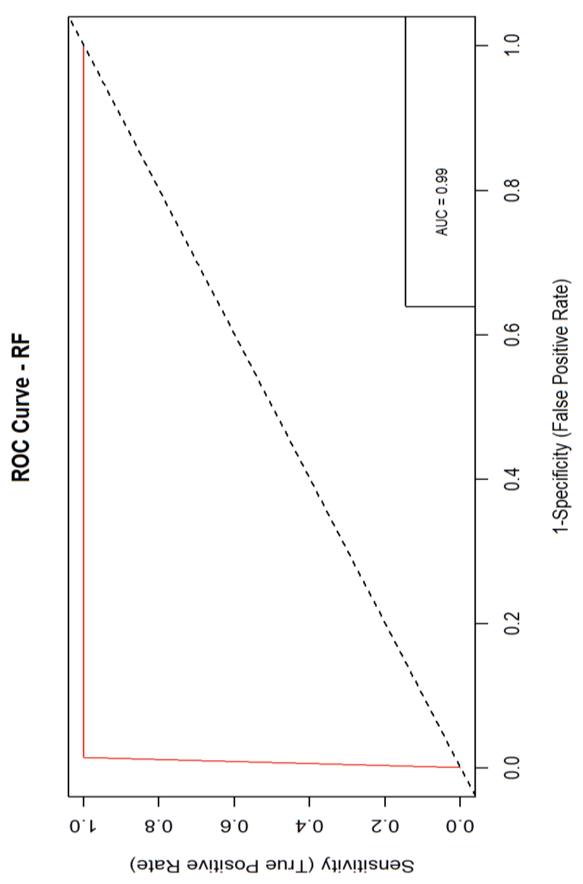
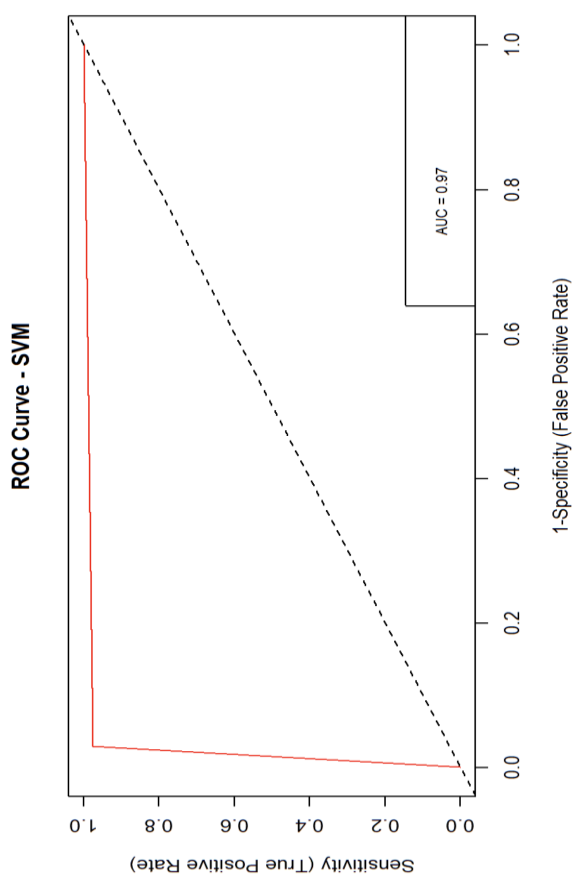
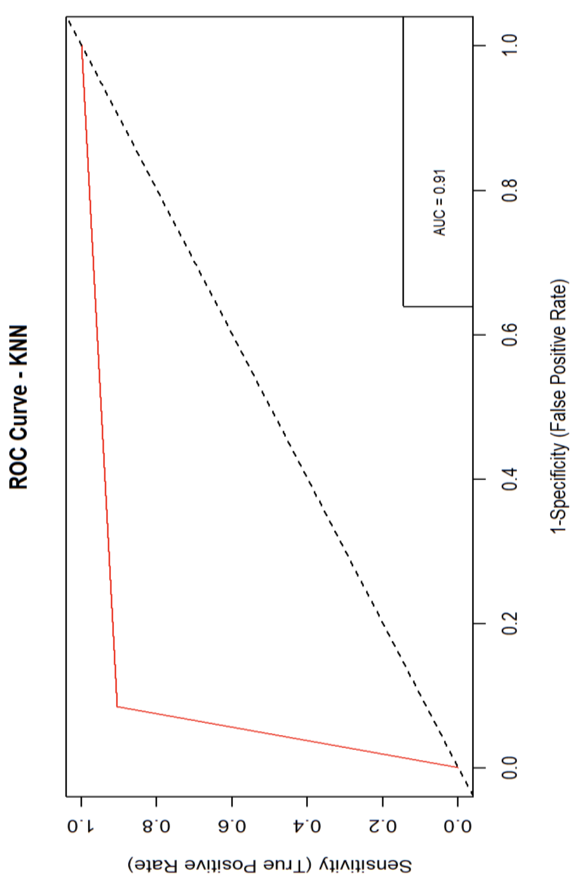
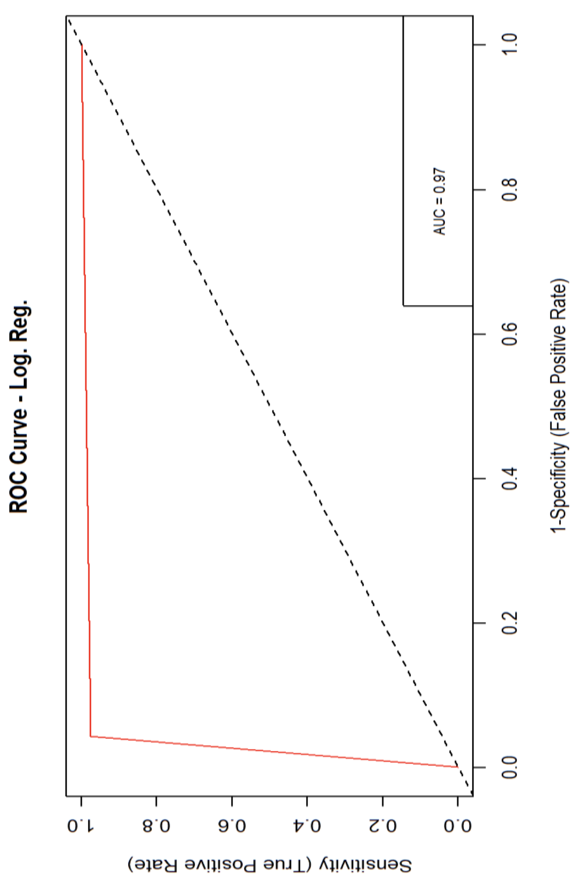


Figure 9: AUROC for various classifiers


ROC Curve - RF
AUC = 0.99
1-Specificity (False Positive Rate)
Sensitivity (True Positive Rate)
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve - SVM
AUC = 0.97
1-Specificity (False Positive Rate)
Sensitivity (True Positive Rate)
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve - KNN
AUC = 0.91
1-Specificity (False Positive Rate)
Sensitivity (True Positive Rate)
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curve - Log. Reg.
AUC = 0.97
1-Specificity (False Positive Rate)
Sensitivity (True Positive Rate)
0.0
0.2
0.4
0.6
0.8
1.0

4 Discussion

The fact that the performance was excellent for all methods making it difficult to compare. It suggests that breast cancer can be accurately classified using machine learning techniques. The slightly better performance of RF suggests that it may be a better choice for this application. However, the difference in performance is not large, so other factors, such as computational complexity, may also need to be considered when choosing a classifier. The high accuracy of the classifiers suggests that they could be used to help doctors make decisions about breast cancer diagnosis. However, it is important to note that the classifiers were trained on a specific dataset, and their performance may not be the same on other datasets. The results of this study are consistent with the findings of other studies that have used machine learning to diagnose breast cancer. This study also has some limitations. First, the sample size is relatively small. This could have led to overfitting of the models. Second, the dataset is not balanced, with more instances of benign tumors than malignant tumors. This could have biased the results of the study. Third, the study only used a single dataset. It would be useful to test the models on other datasets to see if the results are consistent. The dataset was collected in the 1980s, and the features may not be as relevant to modern breast cancer diagnosis. Future research could study the use of advanced machine learning techniques such as deep learning to classify breast cancer in other populations, such as men or younger women.

4.1 Reasons for High Accuracy

The remarkable accuracy achieved by our models prompts a natural inquiry: why are we observing such high accuracy? The figure 10.1 to 10.4 helps to visualize the relationship between the high importance variables and how it relates to the diagnosis of the samples. The different colors will indicate the two classes of the diagnosis.

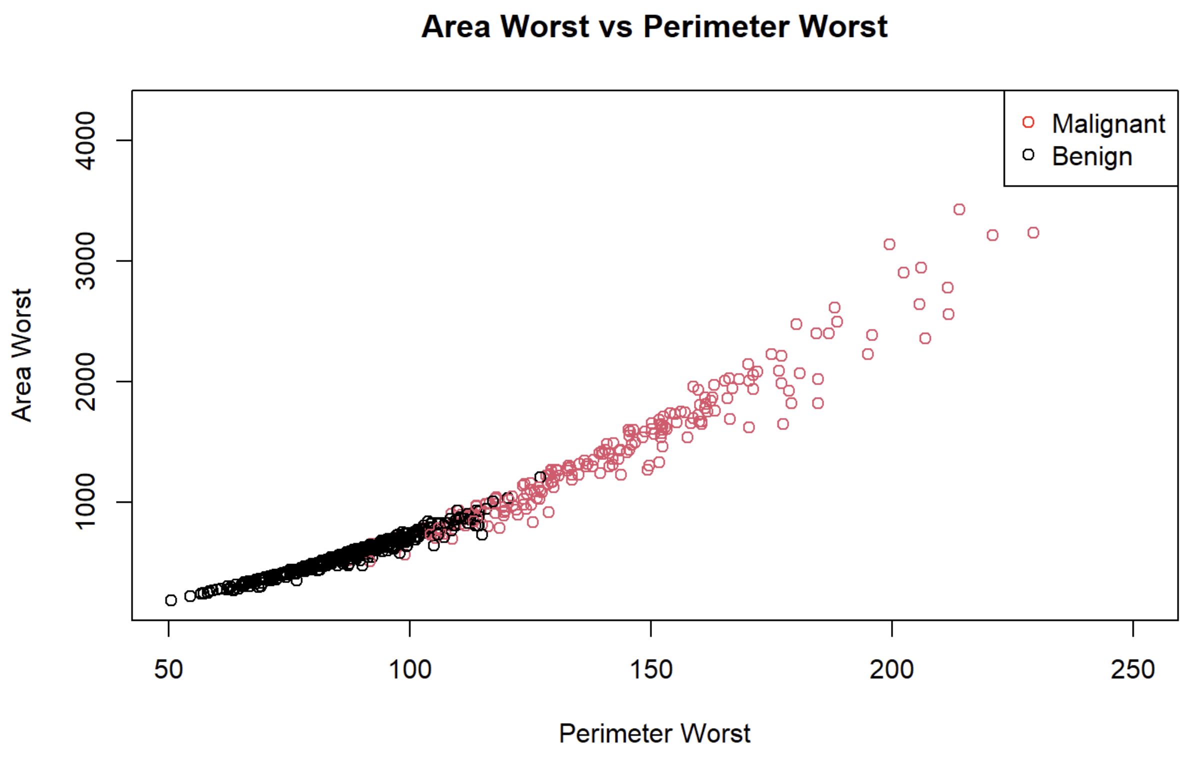


Figure 10.1: Area Worst vs Perimeter Worst Relationship

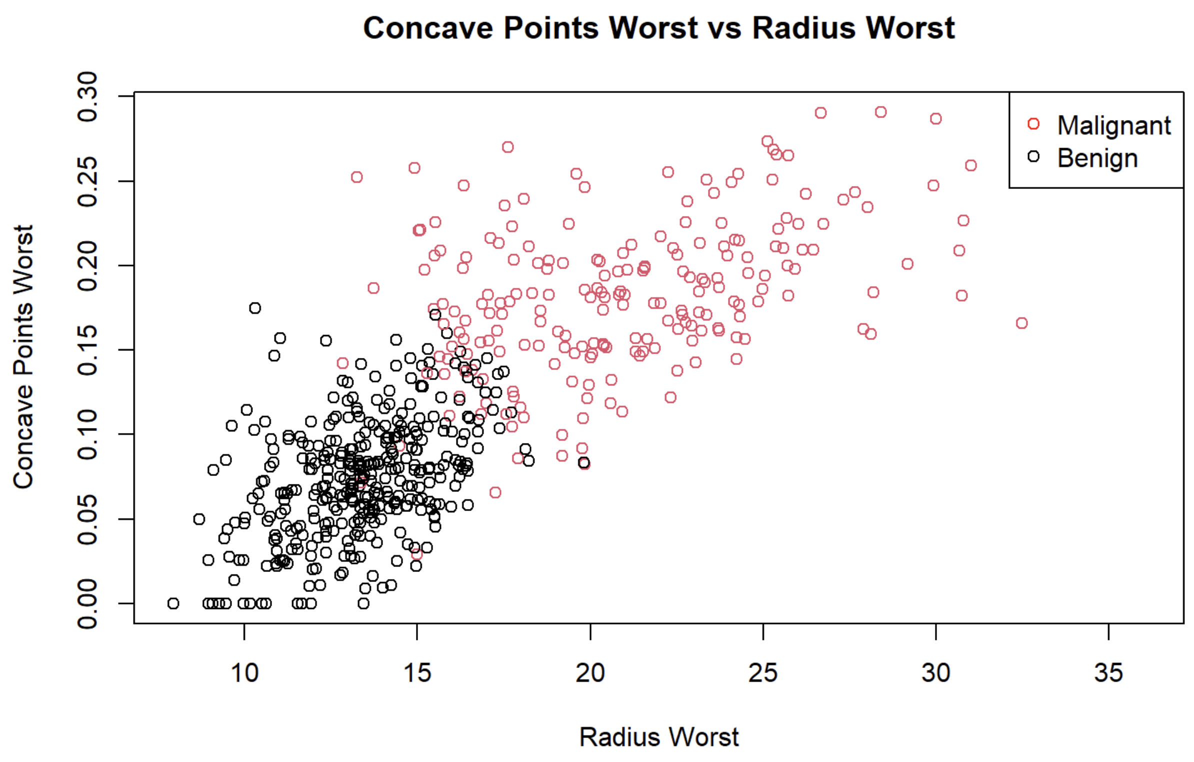


Figure 10.2: Concave Points Worst vs Radius Worst Relationship

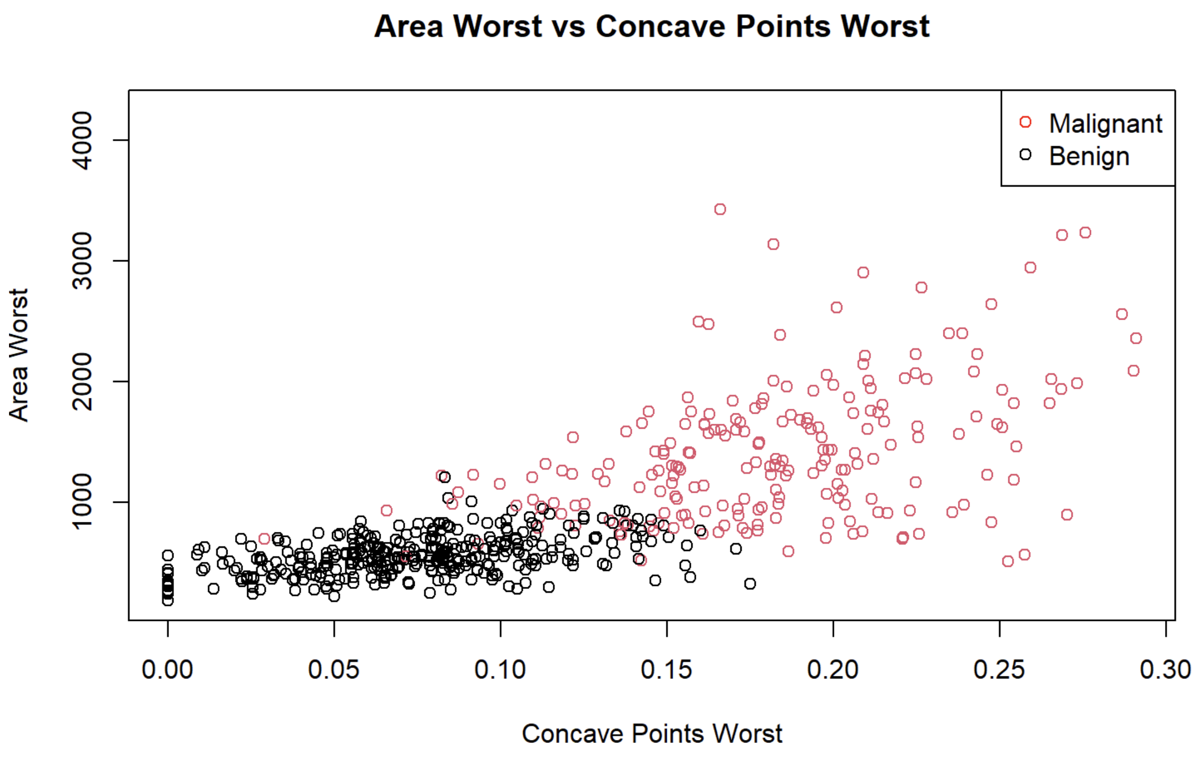


Figure 10.3: Area Worst vs Concave Points Worst Relationship

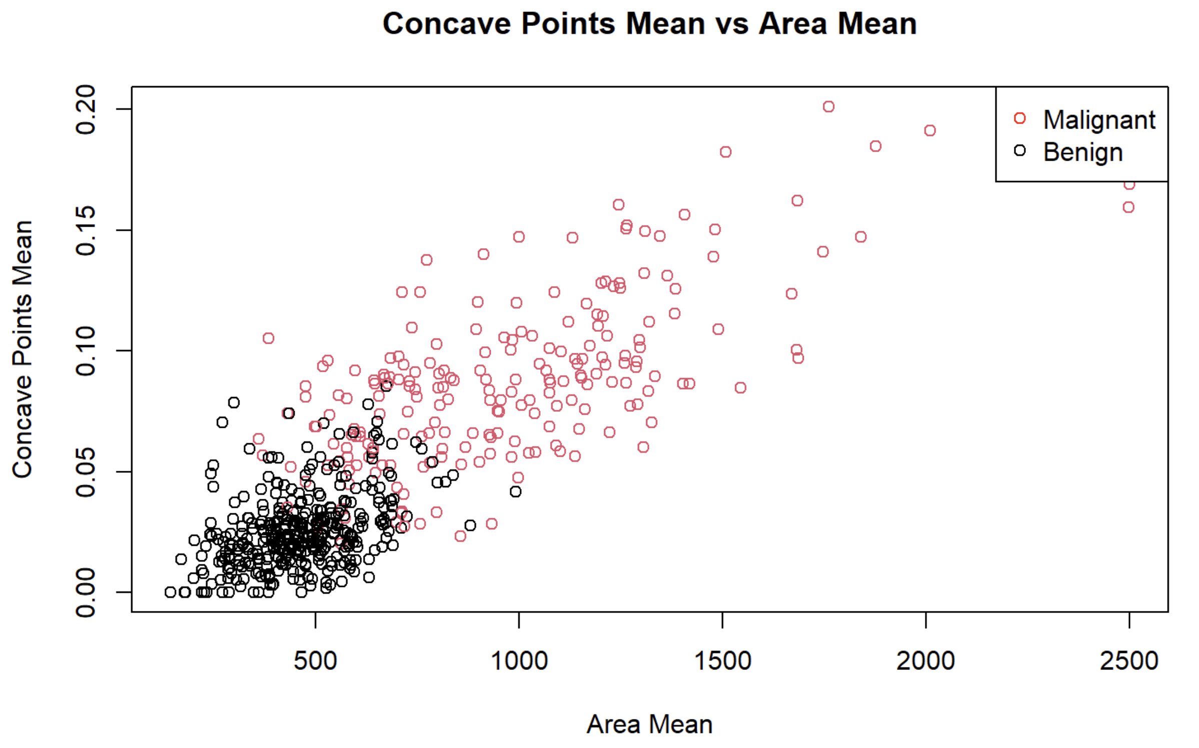


Figure 10.4: Concave Points Mean vs Area Mean Relationship

By examining above scatter plots, we can observe clear separation between the two classes based on the variables, it indicates that the classes form distinct clusters or groups with minimal overlap. This clear separation implies that there are distinct patterns or features in the data that can be used to distinguish between the classes.

5 Conclusion

In our study, the pre-processing of dataset was done and then divided into a training set and a testing set. Recursive feature elimination (RFE) is a feature selection method that was used to select the most important features for a model. RFE works by iteratively removing features that are the least important for the model. Training and testing the models on the training and test data, respectively was done. It is important to evaluate the performance of the models. The performance of the models was evaluated using metrics such as accuracy, precision, and recall. In conclusion, the results of this study suggest that machine learning can be used to accurately diagnose breast cancer. The performance was not comparable as it was excellent for all methods. The study was limited by the small sample size. A larger sample size would be needed to confirm the findings of the study. The study only used a single dataset. It would be useful to test the models on other datasets to see if the results are consistent. The study did not investigate the use of other machine learning techniques, such as deep learning. Future research could investigate the use of these techniques for breast cancer diagnosis. The prevalence of breast cancer in Ireland is about 1 in 8 women. This means that about 12% of women in Ireland will develop breast cancer in their lifetime. The incidence of breast cancer is increasing in Ireland, as it is in many other countries. This is likely due to a combination of factors, including the aging population, increased obesity rates, and changes in lifestyle.[25] The best way to detect breast cancer early is through regular screening. Mammography is the most effective screening method for breast cancer, but MRI can also be used in women with dense breasts or who have a high risk of breast cancer. Most common modern-day methods for diagnosing breast cancer are Breast Ultrasound, Fine Needle Aspiration, Core Needle Biopsy. Techniques like SMOTE can help balance the data and improve model accuracy. Although there's still a long way to go, each step brings us closer to a future where breast cancer diagnoses are not only quick but also reliable and accurate.

6 Abbreviations

CAD - Computer-Aided Diagnosis

Support Vector Machine (SVM)

ANN - Artificial Neural Network

PCA - Principal Component Analysis

SVM-SMO - support vector machine-sequential minimal optimization

KNN - k-Nearest Neighbour

RF - Random Forest

GRU - Gated Recurrent Unit

XGB - Extreme Gradient Boost

RFE - Recursive Feature Elimination

RFFE - Random Forest Feature Elimination

RBF - Radial Basis Function

ROC - Receiver Operating Characteristic

AUC - Area Under Curve

LOOCV - Leave-one-out cross-validation

TP - True Positive

FP - False Positive

TN - True Negative

FN - False Negative

GAM - Generalized Additive Model

8 Bibliography

The list of references and sources consulted during the project is provided.

1. archive.ics.uci.edu. (n.d.). UCI Machine Learning Repository. [online] Available at: https://archive.ics.uci.edu/dataset/17/breast+cancer+wisconsin+diagnostic.

2. www.who.int. (n.d.). Global breast cancer initiative implementation framework: assessing, strengthening and scaling up of services for the early detection and management of breast cancer: executive summary. [online] Available at: https://www.who.int/publications/i/item/9789240067134.

3. https://www.facebook.com/verywell (2019). Differences Between a Malignant and Benign Tumor. [online] Verywell Health. Available at: https://www.verywellhealth.com/what-does-malignant-and-benign-mean-514240.

4. Uddin, K.M.M., Biswas, N., Rikta, S.T. and Dey, S.K. (2023). Machine learning-based diagnosis of breast cancer utilizing feature optimization technique. Computer Methods and Programs in Biomedicine Update, [online] 3, p.100098. doi:https://doi.org/10.1016/j.cmpbup.2023.100098.

5. Soni, R., Shaik, S.Z. and Latha, Y.L.M. (2023). Unlocking the Potential of Machine Learning for Accurate Diagnosis of Breast Cancer. [online] IEEE Xplore. doi:https://doi.org/10.1109/CONIT59222.2023.10205897.

6. Ak, M.F. (2020). A Comparative Analysis of Breast Cancer Detection and Diagnosis Using Data Visualization and Machine Learning Applications. Healthcare, 8(2), p.111. doi:https://doi.org/10.3390/healthcare8020111.

7. Biswas, N., Uddin, K.M.M., Rikta, S.T. and Dey, S.K. (2023). Machine learning-based diagnosis of breast cancer utilizing feature optimization technique. Computer Methods and Programs in Biomedicine Update, [online] 3, p.100098. doi:https://doi.org/10.1016/j.cmpbup.2023.100098.

8. Vincent M. and Magboo, Ma.S.A. (2021). Machine Learning Classifiers on Breast Cancer Recurrences. Procedia Computer Science, 192, pp.2742–2752. doi:https://doi.org/10.1016/j.procs.2021.09.044.

9. Adebiyi, M.O., Arowolo, M.O., Mshelia, M.D. and Olugbara, O.O. (2022). A Linear Discriminant Analysis and Classification Model for Breast Cancer Diagnosis. Applied Sciences, 12(22), p.11455. doi:https://doi.org/10.3390/app122211455.

10. Kourou, K., Exarchos, T. P., Exarchos, K. P., Karamouzis, M. V., & Fotiadis, D. I. (2015). Machine learning applications in cancer prognosis and prediction. Computational and Structural Biotechnology Journal.

11. www.ibm.com. (n.d.). What is Data Visualization? | IBM. [online] Available at: https://www.ibm.com/topics/data-visualization#:~:text=Data%20visualization%20is%20the%20representation.

12. Columbia University Mailman School of Public Health. (2016). Thin Plate Spline Regression. [online] Available at: https://www.publichealth.columbia.edu/research/population-health-methods/thin-plate-spline-regression#:~:text=R%20can%20be%20used%20to [Accessed 1 Aug. 2023].

13. Eberly, D., Tools, G. and Wa, R. (n.d.). Thin-Plate Splines. [online] Available at: https://www.geometrictools.com/Documentation/ThinPlateSplines.pdf [Accessed 13 Aug. 2023].

14. Brownlee, J. (2020). Recursive Feature Elimination (RFE) for Feature Selection in Python. [online] Machine Learning Mastery. Available at: https://machinelearningmastery.com/rfe-feature-selection-in-python/.

15. Dr. Michael Cronin, 2023, Generalized Linear Model Techniques, 2023

16. Gandhi, R. (2018). Support Vector Machine — Introduction to Machine Learning Algorithms. [online] Towards Data Science. Available at: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47.

17. E R, S. (2021). Random Forest | Introduction to Random Forest Algorithm. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2021/06/understanding-random-forest/.

18. Lawton, G. (2022). What is Logistic Regression? - Definition from SearchBusinessAnalytics. [online] SearchBusinessAnalytics. Available at: https://www.techtarget.com/searchbusinessanalytics/definition/logistic-regression.

19. Srivastava, T. (2019). Introduction to KNN, K-Nearest Neighbors : Simplified. [online] Analytics Vidhya. Available at: https://www.analyticsvidhya.com/blog/2018/03/introduction-k-neighbours-algorithm-clustering/.

20. Simplilearn.com. (2022). What is XGBoost? An Introduction to XGBoost Algorithm in Machine Learning | Simplilearn. [online] Available at: https://www.simplilearn.com/what-is-xgboost-algorithm-in-machine-learning-article.

21. Uddin, S., Khan, A., Hossain, M.E. and Moni, M.A. (2019). Comparing different supervised machine learning algorithms for disease prediction. BMC Medical Informatics and Decision Making, [online] 19(1). doi:https://doi.org/10.1186/s12911-019-1004-8.

22. Beheshti, N. (2022). Guide to Confusion Matrices & Classification Performance Metrics. [online] Medium. Available at: https://towardsdatascience.com/guide-to-confusion-matrices-classification-performance-metrics-a0ebfc08408e#:~:text=Confusion%20matrices%20can%20be%20used.

23. GeeksforGeeks. (2020). AUC-ROC Curve. [online] Available at: https://www.geeksforgeeks.org/auc-roc-curve/.

24. Vidiyala, R. (2020). Performance Metrics for Classification Machine Learning Problems. [online] Medium. Available at: https://towardsdatascience.com/performance-metrics-for-classification-machine-learning-problems-97e7e774a007.

25. Zeeshan, M., Salam, B., Khalid, Q.S.B., Alam, S. and Sayani, R. (2018). Diagnostic Accuracy of Digital Mammography in the Detection of Breast Cancer. Cureus, [online] 10(4). doi:https://doi.org/10.7759/cureus.2448.

26. Uddin, S., Khan, A., Hossain, M.E. and Moni, M.A. (2019). Comparing different supervised machine learning algorithms for disease prediction. BMC Medical Informatics and Decision Making, [online] 19(1). doi:https://doi.org/10.1186/s12911-019-1004-8.

9 Appendices

9.1 R Script Explanation

Below is a breakdown of the different sections in the code:

1. **Data Preprocessing and Exploration**: The code starts by loading the dataset, performing some initial data preprocessing such as converting the diagnosis column to a factor and removing the "id" column. It then generates summary statistics and plots histograms of various features.
2. **Correlation Plot**: The code creates a correlation plot using the **corrplot** library to visualize the correlation between different features.
3. **Spline Fitting**: The code uses the **mgcv** library to perform generalized additive modeling (GAM) with spline functions to understand the relationship between individual features and the diagnosis. It creates plots for several features' log-odds against their respective predictors.
4. **Random Forest Importance**: The code utilizes the **randomForest** library to calculate variable importance scores using the Random Forest algorithm.
5. **Feature Selection**: The code applies recursive feature elimination (RFE) using the **caret** library with different methods (e.g., **rfFuncs** and **treebagFuncs**) to select a subset of important features.
6. **Model Fitting and Evaluation**: The code proceeds to fit and evaluate various machine learning models such as Support Vector Machine (SVM), Random Forest, Logistic Regression, Lasso Regression, k-Nearest Neighbors (k-NN), and XGBoost. It uses the **train** function from the **caret** library to train these models.
7. **ROC Curve and AUC**: The code calculates and plots the Receiver Operating Characteristic (ROC) curves for the trained models, then calculates the Area Under the Curve (AUC) values for each model's ROC curve.
8. **Performance Metrics**: The code calculates and prints various evaluation metrics such as accuracy, precision, recall (sensitivity), and F1 score for each model.
9. **Overall Performance Summary**: The code prints a summary of the evaluation metrics for each model.

9.2 R scripts

```
# Set working directory and load the data
setwd("C:/Users/dilip/Downloads")
df = read.csv("BC_Diagnostics.csv")

# Convert diagnosis column to a factor
df$diagnosis = as.factor(df$diagnosis)
df$id = NULL

# Display summary and structure of the data
summary(df)
str(df)

# Calculate proportion of each diagnosis category
prop.table(table(df$diagnosis))

# Load required libraries
library(corrplot)
library(mgcv)
library(caret)
library(randomForest)

# Create a correlation plot
library(corrplot)
corrplot(cor(df[,2:31]), method = "circle", type = "upper", tl.col =
"black")
cor_matrix <- cor(df[, 2:31])

# Create histograms for various features
```

```
par(mfrow = c(2, 3))

for (i in 2:21) {

  hist(df[, i], main = paste("Histogram of", colnames(df)[i]), xlab
= colnames(df)[i])

}


# Create the count plot

plot(df$diagnosis, xlab = "Diagnosis (0 - B, 1 - M)", ylab =
"Count")


# Create spline plots for selected features

par(mfrow = c(2, 2))

# Radius Mean

model1 <- gam(diagnosis ~ s(radius_mean), data = df,
family=binomial())

#summary(model1)

#plot(model1)

plot(model1, select = 1, scheme = 1, se = TRUE, xlab = "Radius
Mean", ylab = "Log(Odds)")

# Texture Mean

model2 <- gam(diagnosis ~ s(texture_mean), data = df,
family=binomial())

#summary(model2)

#plot(model2, ylab = "Log(Odds)")

plot(model2, select = 1, scheme = 1, se = TRUE, xlab = "Texture
Mean", ylab = "Log(Odds)")


# Perimeter Mean

model3 <- gam(diagnosis ~ s(perimeter_mean), data = df,
family=binomial())

#summary(model3)

#plot(model3, ylab = "Log(Odds)")

plot(model3, select = 1, scheme = 1, se = TRUE, xlab = "Perimeter
Mean", ylab = "Log(Odds)")
```

```
# Area Mean

model4 <- gam(diagnosis ~ s(area_mean), data = df,
family=binomial())

#summary(model4)

#plot(model4, ylab = "Log(Odds)")

plot(model4, select = 1, scheme = 1, se = TRUE, xlab = "Area Mean",
ylab = "Log(Odds)")


# Smoothness Mean

model5 <- gam(diagnosis ~ s(smoothness_mean), data = df,
family=binomial())

#summary(model5)

#plot(model5, ylab = "Log(Odds)")

plot(model5, select = 1, scheme = 1, se = TRUE, xlab = "Smoothness
Mean", ylab = "Log(Odds)")


# Compactness Mean

model6 <- gam(diagnosis ~ s(compactness_mean), data = df,
family=binomial())

#summary(model6)

#plot(model6, ylab = "Log(Odds)")

plot(model6, select = 1, scheme = 1, se = TRUE, xlab = "Compactness
Mean", ylab = "Log(Odds)")


# Concavity Mean

model7 <- gam(diagnosis ~ s(concavity_mean), data = df,
family=binomial())

#summary(model7)

#plot(model7, ylab = "Log(Odds)")

plot(model7, select = 1, scheme = 1, se = TRUE, xlab = "Concavity
Mean", ylab = "Log(Odds)")


# Concave Points Mean

model8 <- gam(diagnosis ~ s(concave.points_mean), data = df,
family=binomial())

#summary(model8)
```

```
#plot(model8, ylab = "Log(Odds)")

plot(model8, select = 1, scheme = 1, se = TRUE, xlab = "Concave Pts
Mean", ylab = "Log(Odds)")


# Symmentry Mean

model9 <- gam(diagnosis ~ s(symmetry_mean), data = df,
family=binomial())

#summary(model9)

#plot(model9, ylab = "Log(Odds)")

plot(model9, select = 1, scheme = 1, se = TRUE, xlab = "Symmetry
Mean", ylab = "Log(Odds)")


# Fractal Dimension Mean

model10 <- gam(diagnosis ~ s(fractal_dimension_mean), data = df,
family=binomial())

#summary(model10)

#plot(model10, ylab = "Log(Odds)")

plot(model10, select = 1, scheme = 1, se = TRUE, xlab = "Fractal
Dim. Mean", ylab = "Log(Odds)")



# Radius SE

model11 <- gam(diagnosis ~ s(radius_se), data = df,
family=binomial())

#summary(model11)

#plot(model11, ylab = "Log(Odds)")

plot(model11, select = 1, scheme = 1, se = TRUE, xlab = "Radius SE",
ylab = "Log(Odds)")


# Texture SE

model12 <- gam(diagnosis ~ s(texture_se), data = df,
family=binomial())

#summary(model12)

#plot(model12, ylab = "Log(Odds)")

plot(model12, select = 1, scheme = 1, se = TRUE, xlab = "Texture
SE", ylab = "Log(Odds)")
```

```
# Perimeter SE

model13 <- gam(diagnosis ~ s(perimeter_se), data = df,
family=binomial())

#summary(model13)

#plot(model13, ylab = "Log(Odds)")

plot(model13, select = 1, scheme = 1, se = TRUE, xlab = "Perimeter
SE", ylab = "Log(Odds)")


# Area SE

model14 <- gam(diagnosis ~ s(area_se), data = df, family=binomial())

#summary(model14)

#plot(model14, ylab = "Log(Odds)")

plot(model14, select = 1, scheme = 1, se = TRUE, xlab = "Area SE",
ylab = "Log(Odds)")


# Smoothness SE

model15 <- gam(diagnosis ~ s(smoothness_se), data = df,
family=binomial())

#summary(model15)

#plot(model15, ylab = "Log(Odds)")

plot(model15, select = 1, scheme = 1, se = TRUE, xlab = "Smoothness
SE", ylab = "Log(Odds)")


# Compactness SE

model16 <- gam(diagnosis ~ s(compactness_se), data = df,
family=binomial())

#summary(model16)

#plot(model16, ylab = "Log(Odds)")

plot(model16, select = 1, scheme = 1, se = TRUE, xlab = "Compactness
SE", ylab = "Log(Odds)")


# Concavity SE

model17 <- gam(diagnosis ~ s(concavity_se), data = df,
family=binomial())
```

```
summary(model17)

#plot(model17, ylab = "Log(Odds)")

plot(model17, select = 1, scheme = 1, se = TRUE, xlab = "Concavity
SE", ylab = "Log(Odds)")


# Concave Points SE

model18 <- gam(diagnosis ~ s(concave.points_se), data = df,
family=binomial())

summary(model18)

#plot(model18, ylab = "Log(Odds)")

plot(model18, select = 1, scheme = 1, se = TRUE, xlab = "Concave
Pts. SE", ylab = "Log(Odds)")


# Symmentry SE

model19 <- gam(diagnosis ~ s(symmetry_se), data = df,
family=binomial())

#summary(model19)

#plot(model19, ylab = "Log(Odds)")

plot(model19, select = 1, scheme = 1, se = TRUE, xlab = "Symmetry
SE", ylab = "Log(Odds)")


# Fractal Dimension SE

model20 <- gam(diagnosis ~ s(fractal_dimension_se), data = df,
family=binomial())

#summary(model20)

#plot(model20, ylab = "Log(Odds)")

plot(model20, select = 1, scheme = 1, se = TRUE, xlab = "Fractal
Dim. SE", ylab = "Log(Odds)")



# Radius Worst

model21 <- gam(diagnosis ~ s(radius_worst), data = df,
family=binomial())

#summary(model21)

#plot(model21, ylab = "Log(Odds)")
```

```
plot(model21, select = 1, scheme = 1, se = TRUE, xlab = "Radius
Worst", ylab = "Log(Odds)")

# Texture Worst

model22 <- gam(diagnosis ~ s(texture_worst), data = df,
family=binomial())

#summary(model22)

#plot(model22, ylab = "Log(Odds)")

plot(model22, select = 1, scheme = 1, se = TRUE, xlab = "Texture
Worst", ylab = "Log(Odds)")


# Perimeter Worst

model23 <- gam(diagnosis ~ s(perimeter_worst), data = df,
family=binomial())

#summary(model23)

#plot(model23, ylab = "Log(Odds)")

plot(model23, select = 1, scheme = 1, se = TRUE, xlab = "Perimeter
Worst", ylab = "Log(Odds)")


# Area Worst

model24 <- gam(diagnosis ~ s(area_worst), data = df,
family=binomial())

#summary(model24)

#plot(model24, ylab = "Log(Odds)")

plot(model24, select = 1, scheme = 1, se = TRUE, xlab = "Area
Worst", ylab = "Log(Odds)")


# Smoothness Worst

model25 <- gam(diagnosis ~ s(smoothness_worst), data = df,
family=binomial())

#summary(model25)

#plot(model25, ylab = "Log(Odds)")

plot(model25, select = 1, scheme = 1, se = TRUE, xlab = "Smoothness
Worst", ylab = "Log(Odds)")


# Compactness Worst
```

```
model26 <- gam(diagnosis ~ s(compactness_worst), data = df,
family=binomial())

#summary(model26)

#plot(model26, ylab = "Log(Odds)")

plot(model26, select = 1, scheme = 1, se = TRUE, xlab = "Compactness
Worst", ylab = "Log(Odds)")


# Concavity Worst

model27 <- gam(diagnosis ~ s(concavity_worst), data = df,
family=binomial())

#summary(model27)

#plot(model27, ylab = "Log(Odds)")

plot(model27, select = 1, scheme = 1, se = TRUE, xlab = "Concavity
Worst", ylab = "Log(Odds)")


# Concave Points Worst

model28 <- gam(diagnosis ~ s(concave.points_worst), data = df,
family=binomial())

#summary(model28)

#plot(model28, ylab = "Log(Odds)")

plot(model28, select = 1, scheme = 1, se = TRUE, xlab = "Concave Pts
Worst", ylab = "Log(Odds)")


# Symmentry Worst

model29 <- gam(diagnosis ~ s(symmetry_worst), data = df,
family=binomial())

#summary(model29)

#plot(model29, ylab = "Log(Odds)")

plot(model29, select = 1, scheme = 1, se = TRUE, xlab = "Symmetry
Worst", ylab = "Log(Odds)")


# Fractal Dimension Worst

model30 <- gam(diagnosis ~ s(fractal_dimension_worst), data = df,
family=binomial())

#summary(model30)

#plot(model30, ylab = "Log(Odds)")
```

```
plot(model30, select = 1, scheme = 1, se = TRUE, xlab = "Fractal
Dim. Worst", ylab = "Log(Odds)")



# Create a model for feature selection using randomForest

model <- randomForest(diagnosis~., data=train_data)

importance(model)

varImpPlot(model)


# Feature selection using Recursive Feature Elimination (RFE)

set.seed(1000)

ctrl <- rfeControl(functions = rfFuncs, method = "cv", number = 10)


rfe_result <- rfe(x = df[,-1],

                  y = df$diagnosis,

                  sizes = c(1:30),

                  rfeControl = ctrl)

selected_features <- predictors(rfe_result)

print(selected_features)


# Data for selected features

cancer_data =
df[,c("diagnosis","area_worst","concave.points_worst","perimeter_wor
st","radius_worst","texture_worst","concave.points_mean","area_se","
texture_mean","concavity_worst","smoothness_worst","concavity_mean",
"area_mean")]


# Split the data into training and testing sets

set.seed(123)

train_indices <- createDataPartition(cancer_data$diagnosis, p = 0.8,
list = FALSE)

train_data <- cancer_data[train_indices, ]

test_data <- cancer_data[-train_indices, ]


# Convert the data into matrix format required by glmnet
```

```
x_train <- as.matrix(train_data[, -1])

y_train <- train_data$diagnosis

x_test <- as.matrix(test_data[, -1])


# Define the control parameters for model training

ctrl <- trainControl(method = "cv", number = 5)


# Fit Support Vector Machine (SVM) model

svm_model <- train(diagnosis ~ ., data = train_data, method = 
"svmRadial",

                   trControl = ctrl)


# Fit Random Forest model

rf_model <- train(diagnosis ~ ., data = train_data, method = "rf",

                  trControl = ctrl)


# Fit Logistic Regression model

logreg_model <- train(diagnosis ~ ., data = train_data, method = 
"glm",

                      trControl = ctrl, family = "binomial")


# Fit K-Nearest Neighbors model

knn_model <- train(diagnosis ~ ., data = train_data, method = "knn",

                   trControl = ctrl)


# Fit XGBoost model

xgb_model <- train(diagnosis ~ ., data = train_data, method = 
"xgbTree",

                   trControl = ctrl)


# Print the model results

print(svm_model)

print(rf_model)

print(logreg_model)
print(knn_model)


# Predict on the test data using the trained models
svm_pred <- predict(svm_model, newdata = test_data)
rf_pred <- predict(rf_model, newdata = test_data)
logreg_pred <- predict(logreg_model, newdata = test_data)
knn_pred <- predict(knn_model, newdata = test_data)
xgb_pred <- predict(xgb_model, newdata = test_data)


# Evaluate model performance
# Accuracy: The percentage of tumors whose malignancy was correctly
predicted
svm_accuracy <- mean(svm_pred == test_data$diagnosis)
rf_accuracy <- mean(rf_pred == test_data$diagnosis)
logreg_accuracy <- mean(logreg_pred == test_data$diagnosis)
knn_accuracy <- mean(knn_pred == test_data$diagnosis)
xgb_accuracy <- mean(xgb_pred == test_data$diagnosis)


# Confusion Matrix
svm_cm <- confusionMatrix(svm_pred, test_data$diagnosis, positive =
"1" )
rf_cm <- confusionMatrix(rf_pred, test_data$diagnosis, positive =
"1")
logreg_cm <- confusionMatrix(logreg_pred, test_data$diagnosis,
positive = "1")
knn_cm <- confusionMatrix(knn_pred, test_data$diagnosis, positive =
"1")
xgb_cm <- confusionMatrix(xgb_pred, test_data$diagnosis, positive =
"1")


accuracy = cbind(svm_accuracy, rf_accuracy, logreg_accuracy,
knn_accuracy, xgb_accuracy)
colnames(accuracy) = c("SVM","RF","Log. Reg.", "KNN", "XGB")
round(accuracy, 4)
```

```
# Precision: TP / (TP + FP)

svm_precision <- svm_cm$byClass['Precision']

rf_precision <- rf_cm$byClass['Precision']

logreg_precision <- logreg_cm$byClass['Precision']

knn_precision <- knn_cm$byClass['Precision']

xgb_precision <- xgb_cm$byClass['Precision']


precision = cbind(svm_precision, rf_precision, logreg_precision,
knn_precision, xgb_precision)

colnames(precision) = c("SVM","RF","Log. Reg.", "KNN", "XGB")

round(precision, 4)


# Recall: TP / (TP + FN)

svm_recall <- svm_cm$byClass['Sensitivity']

rf_recall <- rf_cm$byClass['Sensitivity']

logreg_recall <- logreg_cm$byClass['Sensitivity']

knn_recall <- knn_cm$byClass['Sensitivity']

xgb_recall <- xgb_cm$byClass['Sensitivity']


recall = cbind(svm_recall, rf_recall, lasso_recall, knn_recall,
xgb_recall)

colnames(recall) = c("SVM","RF","Log. Reg.","KNN", "XGB")

round(recall, 4)


# F1 Score: (Precision * Recall) / (Precision + Recall)

svm_f1 <- svm_cm$byClass['F1']

rf_f1 <- rf_cm$byClass['F1']

logreg_f1 <- logreg_cm$byClass['F1']

knn_f1 <- knn_cm$byClass['F1']

xgb_f1 <- xgb_cm$byClass['F1']


f1 = cbind(svm_f1, rf_f1, logreg_f1, knn_f1, xgb_f1)

colnames(f1) = c("SVM","RF","Log. Reg.", "KNN", "XGB")
```

```
round(f1, 4)


# Convert the predicted labels to a prediction object

pred_svm <- prediction(as.numeric(svm_pred),
as.numeric(test_data$diagnosis))

pred_rf <- prediction(as.numeric(rf_pred),
as.numeric(test_data$diagnosis))

pred_logreg <- prediction(as.numeric(logreg_pred),
as.numeric(test_data$diagnosis))

pred_knn <- prediction(as.numeric(knn_pred),
as.numeric(test_data$diagnosis))

pred_xgb <- prediction(as.numeric(xgb_pred),
as.numeric(test_data$diagnosis))


# Create the performance object

perf_svm <- performance(pred_svm, "tpr", "fpr")

perf_rf <- performance(pred_rf, "tpr", "fpr")

perf_logreg <- performance(pred_logreg, "tpr", "fpr")

perf_knn <- performance(pred_knn, "tpr", "fpr")

perf_xgb <- performance(pred_xgb, "tpr", "fpr")


# Calculate the AUC ROC

auc_svm <- performance(pred_svm, "auc")@y.values[[1]]

auc_rf <- performance(pred_rf, "auc")@y.values[[1]]

auc_logreg <- performance(pred_logreg, "auc")@y.values[[1]]

auc_knn <- performance(pred_knn, "auc")@y.values[[1]]

auc_xgb <- performance(pred_xgb, "auc")@y.values[[1]]


auc_roc = cbind(auc_svm, auc_rf, auc_logreg, auc_lasso, auc_knn,
auc_xgb)

colnames(auc_roc) = c("SVM","RF","Log. Reg.", "KNN", "XGB")

round(auc_roc, 4)


par(mfrow=c(2,3))

# Plot the ROC curve
```

```
plot(perf_svm, col = "red", main = "ROC Curve - SVM", xlab = "1-
Specificity (False Positive Rate)", ylab = "Sensitivity (True
Positive Rate)")

legend("bottomright", legend = paste("AUC =", round(auc_svm, 2)),
col = "black", bg = "white", cex = 1)

abline(0, 1, lty = 2, col = "black")


plot(perf_rf, col = "red", main = "ROC Curve - RF", xlab = "1-
Specificity (False Positive Rate)", ylab = "Sensitivity (True
Positive Rate)")

legend("bottomright", legend = paste("AUC =", round(auc_rf, 2)), col
= "black", bg = "white", cex = 0.8)

abline(0, 1, lty = 2, col = "black")


plot(perf_logreg, col = "red", main = "ROC Curve - Log. Reg.", xlab
= "1-Specificity (False Positive Rate)", ylab = "Sensitivity (True
Positive Rate)")

legend("bottomright", legend = paste("AUC =", round(auc_logreg, 2)),
col = "black", bg = "white", cex = 0.8)

abline(0, 1, lty = 2, col = "black")


plot(perf_knn, col = "red", main = "ROC Curve - KNN", xlab = "1-
Specificity (False Positive Rate)", ylab = "Sensitivity (True
Positive Rate)")

legend("bottomright", legend = paste("AUC =", round(auc_knn, 2)),
col = "black", bg = "white", cex = 0.8)

abline(0, 1, lty = 2, col = "black")


plot(perf_xgb, col = "red", main = "ROC Curve - XGB", xlab = "1-
Specificity (False Positive Rate)", ylab = "Sensitivity (True
Positive Rate)")

legend("bottomright", legend = paste("AUC =", round(auc_xgb, 2)),
col = "black", bg = "white", cex = 0.8)

abline(0, 1, lty = 2, col = "black")



# Print the AUC ROC values

print(paste("SVM AUC:", auc_svm)) # 0.9740
print(paste("Random Forest AUC:", auc_rf)) # 0.9929

print(paste("Logistic Regression AUC:", auc_logreg)) # 0.9669

print(paste("k-NN AUC:", auc_knn)) # 0.9110

print(paste("XGB AUC:", auc_xgb)) # 0.9740


auc = cbind(auc_svm, auc_rf, auc_logreg,  auc_knn, auc_xgb)

colnames(auc) = c("SVM","RF","Log. Reg.", "KNN", "XGB")

round(auc, 4)


# Print the performance metrics

print("Evaluation Metrics:")


print(paste("SVM Accuracy:", svm_accuracy)) # 0.9735

print(paste("Random Forest Accuracy:", rf_accuracy)) # 0.9911

print(paste("Logistic Regression Accuracy:", logreg_accuracy)) #
0.9646

print(paste("k-NN Accuracy:", knn_accuracy)) # 0.9115


print(paste("SVM Precision:", svm_precision)) # 0.9857

print(paste("Random Forest Precision:", rf_precision)) # 1

print(paste("Logistic Regression Precision:", logreg_precision)) #
0.9855

print(paste("k-NN Regression Precision:", knn_precision)) # 0.9855


print(paste("SVM Recall:", svm_recall)) # 0.9718

print(paste("Random Forest Recall:", rf_recall)) # 0.9859

print(paste("Logistic Regression Recall:", logreg_recall)) # 0.9577

print(paste("k-NN Recall:", knn_recall)) # 0.9154
```