Breast Cancer Prediction using supervised Machine Learning approach

Supervised by

Md. Khaled Sohel
Assistant Professor
Department of Software Engineering
Daffodil International University

Submitted By

Md Mustain Billah
ID:193-44-184
Department of Software Engineering
Daffodil International University

This Report Presented in Partial Fulfilment of the Requirements for the Degree of Master's of Science in Software Engineering

APPROVAL

This thesis titled on “Breast Cancer Prediction using supervised machine learning approach”, submitted by Md. Mustain Billah, ID: 193-44-184 to the Department of Software Engineering, Daffodil International University has been accepted assatisfactory for the partial fulfillment of the requirements for the degree of Masters of Science inSoftware Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

Chairman

Dr. Imran Mahmud
Associate Professor and Head
Department of Software Engineering
Daffodil International University

Fazla Elahe 03.03.23

Internal Examiner 1

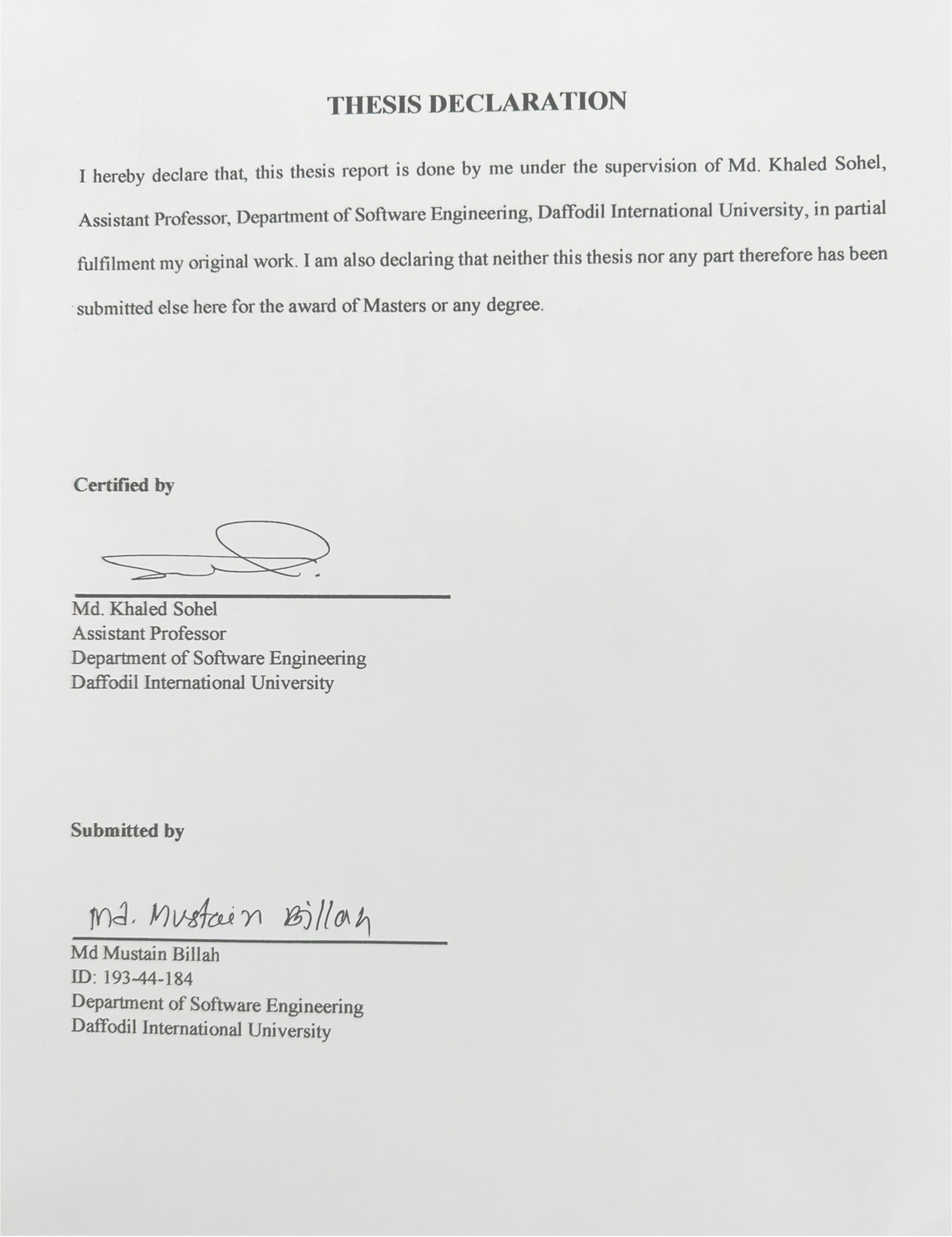
Dr. Md. Fazla Elahe
Assistant Professor and Associate Head
Department of Software Engineering
Daffodil International University

Internal Examiner 2

Afsana Begum
Assistant Professor
Department of Software Engineering
Daffodil International University

External Examiner

Prof. Dr. Md. Saiful Islam
Professor
Bangladesh University of Engineering and Technology (BUET)
Bangladesh



THESIS DECLARATION

I hereby declare that, this thesis report is done by me under the supervision of Md. Khaled Sohel, Assistant Professor, Department of Software Engineering, Daffodil International University, in partial fulfilment my original work. I am also declaring that neither this thesis nor any part therefore has been submitted else here for the award of Masters or any degree.

Certified by

Md. Khaled Sohel
Assistant Professor
Department of Software Engineering
Daffodil International University

Submitted by

Md. Mustain Billah

Md Mustain Billah
ID: 193-44-184
Department of Software Engineering
Daffodil International University

ACKNOWLEDGEMENT

Foremost, I am thankful to God for my well-being and that's why I am completed my research Procedure. Then I am grateful to my research supervisor, Md. Khaled Sohel who guided me throughout the whole research activity. Besides my supervisor, I would love to thank the rest of my research committee for their encouragement and insightful comments. After that, I would like to thank Md. Rajib mia, lecturer at daffodil international university helped me during my research survey. I wish to express my special thanks to Dr. Imran Mahmud, associate professor & Head of the faculty For providing all the necessary facilities for the research purpose. I am also thankful to all the lecturers, Department of software engineering who sincerely guided me in my difficulty. I am grateful to my parents for their unconditional support and encouragement. I am thankful to my friend who supported throughout this venture.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

SVM	Support vector machine
KNN	K-Nearest Neighbor (KNN) Algorithm
ANN	Artificial neural network
MLP	Multilayer perceptron
MCNN	Multilayer Convolutional Neural Network
CAD	Computer-Aided Diagnosis
LR	Logistic regression
NB	Naïve Bayes Classifier
RF	Random Forest
PCA	Principal Component Analysis

ABSTRACT

Breast cancer is one of the most prevalent forms of cancer in women worldwide, making early prediction critical to reducing mortality rates. In this study, we propose a machine learning-based approach to predict benign and malignant stages of breast cancer. Our approach utilizes Principal Component Analysis (PCA) for dimensionality reduction and is compared against Random Forest (RF), Logistic Regression (LR), and XGBoost (XGB) machine learning models. The results demonstrate that the proposed approach is highly efficient, accurate, and effective in predicting breast cancer stage. The findings of this study have the potential to revolutionize the medical sector by providing a non-invasive, quick, and cost-effective method for early prediction of breast cancer. The implementation of this approach can lead to improved patient outcomes and reduced healthcare costs.

CHAPTER 01: INTRODUCTION

1.1 INTRODUCTION:

Breast cancer is a global public health challenge, affecting millions of women worldwide and posing a significant threat to their well-being. The disease's early detection is crucial for timely and effective treatment, resulting in better outcomes and reduced mortality rates. In recent years, there has been a surge of interest in leveraging machine learning algorithms to develop accurate and efficient methods for predicting breast cancer.

The purpose of the proposed study is to investigate how well supervised machine learning algorithms, such as decision trees, random forests, catboost, and xgboost, can predict the benign and malignant stages of breast cancer. The main goal of the project is to evaluate the effectiveness of different algorithms and identify the best strategy for breast cancer prediction. The findings of this study will aid in the creation of an inexpensive, rapid, and non-invasive technique for early breast cancer screening.

This study's significance lies in its potential to revolutionize breast cancer screening and diagnosis, making it more accessible and accurate for women worldwide. By leveraging machine learning algorithms, we can develop a highly effective, personalized, and patient-centered approach to breast cancer prediction that improves the medical sector's overall efficiency and efficacy. Therefore, this research is essential to advance the current understanding of breast cancer prediction using machine learning and will provide a vital contribution to the scientific community.

1.2 RESEARCH QUESTION:

Decision tree, random forest, catboost, and xgboost are examples of supervised machine learning algorithms that can be used to predict breast cancer and help create a non-invasive, rapid, and affordable early detection technique.

1.3 RESEARCH OBJECTIVE:

This study aims to investigate the potential of supervised machine learning algorithms, in particular decision tree, random forest, catboost, and xgboost, to create a more precise and effective method for breast cancer early detection. Comparing these algorithms' performance is the main objective in order to identify the best strategy for breast cancer prediction. By reaching this goal, we hope to aid in the creation of a quick, affordable, non-invasive tool for early diagnosis of breast cancer, thereby reducing the high mortality rate related to the condition.

1.4 THESIS ORGANIZATION:

The next section on the literature initially addressed the research gap. Next, we discussed the research methodology and the hypothesis. Third, we spoke about our findings and interactions. The findings, limitations, and suggested next steps were then presented along with the conclusions and recommendations.

CHAPTER 02: LITERATURE REVIEW

2.1 PREVIOUS LITERATURE:

A common illness that affects millions of people globally is breast cancer. Recent technological developments have enabled the application of machine learning algorithms for predicting the benign and malignant stages of breast cancer. Early identification of breast cancer is essential for lowering death rates. In this review of the literature, we'll look at the literature and research that have already been done on machine learning and breast cancer detection.

Radial Basis Function Network, Naive Bayes, and Decision Tree were utilized by Chaurasia et al. (2015) to predict breast cancer. In order to diagnose breast cancer, Cakir and Demiral (2016) developed a program named "Treatment Helper" that combines the Multilayer Perceptron, D-Class Lifeboat, and Decision Table. SVM, KNN, and ANN approaches were utilized by Yue et al. (2017) to predict breast cancer. Breast cancer was detected by Sharma et al.

A method to classify mammographic lesions was proposed by Kulkarni (2019) using Pixel N-gram features with SVM, MLP, and KNN classifiers. Anthonia Kayode (2019) produced a 94.4% sensitivity score after using SVM to 322 mammography images. The same data was examined by Yijiejin (2020) using a binary classifier (CNNI-BCC), who produced a test data accuracy of about 73.24%. In order to achieve nearly 97% accuracy, Homayoon (2019) and Prabh Kaur (2019) employed Multi-Class Support Vector Machine (MSVM) Clustering and Multi-Convolutional Neural Network (MCNN) Clustering, respectively. Tenfold cross-validation was taken into account by Prabh Kaur in 2019.

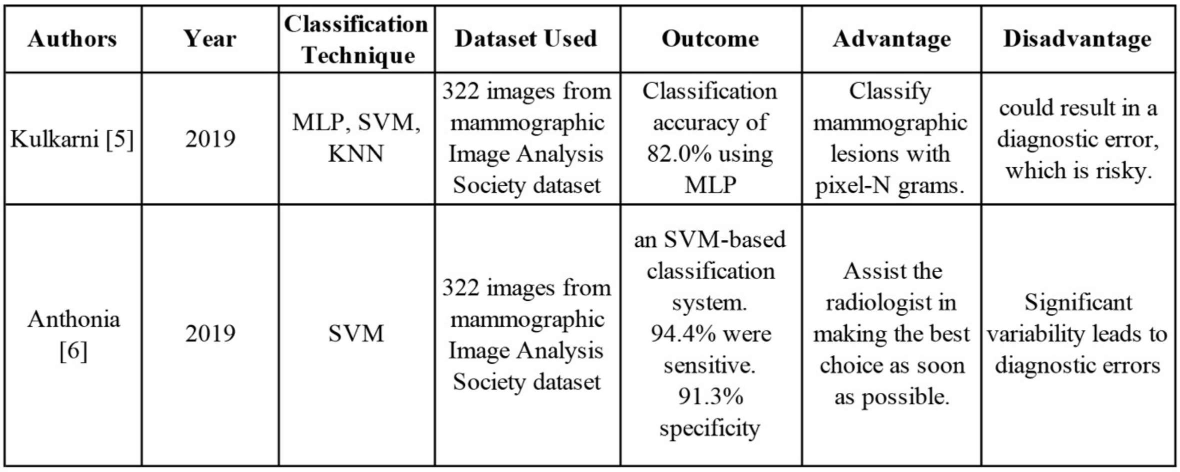
Radial Basis Function Network was used by Ibrahim (2020) to develop a CAD system and a breast cancer detection technique. Their suggested approach had a 79.166% accuracy rate. Ak

(2020) exhibited the use of multiple algorithms, including Logistic Regression (LR), KNN, SVM, NB, and RF, in a comparative comparison to diagnose breast cancer. Among them all, LR's accuracy was the highest. In order to predict breast cancer, Agarap (2021) employed a variety of classifiers, including Gated Recurrent Unit (GRU) with LR, SVM, Multilayer Perceptron, and KNN.

Benbrahim (2021) showed that neural networks outperform all 11 algorithms in a comparison. Using the data mining software weka, Asri (2021) examined the performance of DT, SVM, NB, and KNN using the BCWD dataset.

Overall, the existing research and literature on breast cancer detection and machine learning techniques have shown promising results, and the proposed study aims to contribute to this field by exploring the effectiveness of supervised machine learning algorithms for early prediction of breast cancer.

2.2 RESEARCH GAP:



Authors	Year	Classification Technique	Dataset Used	Outcome	Advantage	Disadvantage
Kulkarni [5]	2019	MLP, SVM, KNN	322 images from mammographic Image Analysis Society dataset	Classification accuracy of 82.0% using MLP	Classify mammographic lesions with pixel-N grams.	could result in a diagnostic error, which is risky.
Anthonia [6]	2019	SVM	322 images from mammographic Image Analysis Society dataset	an SVM-based classification system. 94.4% were sensitive. 91.3% specificity	Assist the radiologist in making the best choice as soon as possible.	Significant variability leads to diagnostic errors

AK [11]	2020	LR, KNN, SVM, NB, RF	Breast Cancer Wisconsin (Diagnostic) Data Set	Using all the feature, LR achieves 98.1% accuracy.	With the right model, making predictions is simple.	Uses all the features are irrelevant.
Agarap [12]	2019	GRU, LR, SVM, MLP, KNN	Breast Cancer Wisconsin (Diagnostic) Data Set	MLP achieved an accuracy of of 99.04%	A predictive model that is accurate and less likely to make mistakes.	use all of the pointless and ineffective aspects
Asri [14]	2016	DT, SVM, NB, KNN	Breast Cancer Wisconsin (Diagnostic) Data Set	SVM achieved a 97.13% accuracy rate.	A complete model to make predictions that are more accurate and less erroneous.	taken into account all the unused and ineffective aspects
Ou [15]		NB, DT, RT	Breast Cancer Wisconsin (Diagnostic) Data Set	Among all, Nave Bayes performs best, with a 76.3%	straightforward breast cancer prediction model.	Can lead significant errors.
Wang [16]		SVM, ANN, NB	WBCD, WDBCS Dataset	dimensionality space was reduced using PCA, and 97.47% accuracy was attained using SVM.	a reliable and straightforward breast cancer prediction model	Tested with typical data; may not work with irregular data.

Table 1: Summary of Literature review

CHAPTER 3: RESEARCH METHODOLOGY

3.1 PROPOSED METHODOLOGY:

Our methodology section can be divided into 5 parts: The collection and analysis of breast cancer data. Its features will be analyzed to understand the data's characteristics. Data Pre-processing is cleaned and pre-processed to remove missing values, outliers, and noise. Relevant features that can aid in the prediction of breast cancer. Feature selection techniques such as correlation analysis, mutual information, and recursive feature elimination will be employed. The supervised machine learning algorithms such as Decision Tree, Random Forest, CatBoost, and XGBoost will be used for breast cancer prediction.Cross-validation techniques will be employed to ensure the robustness of the models.

The methodology will be illustrated through a visual representation, as shown in Figure 1. Each section will be described in detail in the subsequent chapters of the thesis.

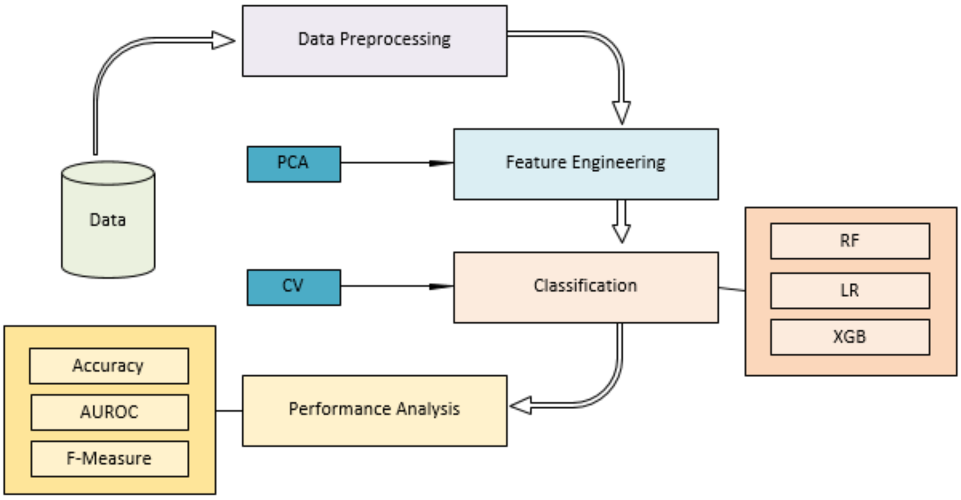


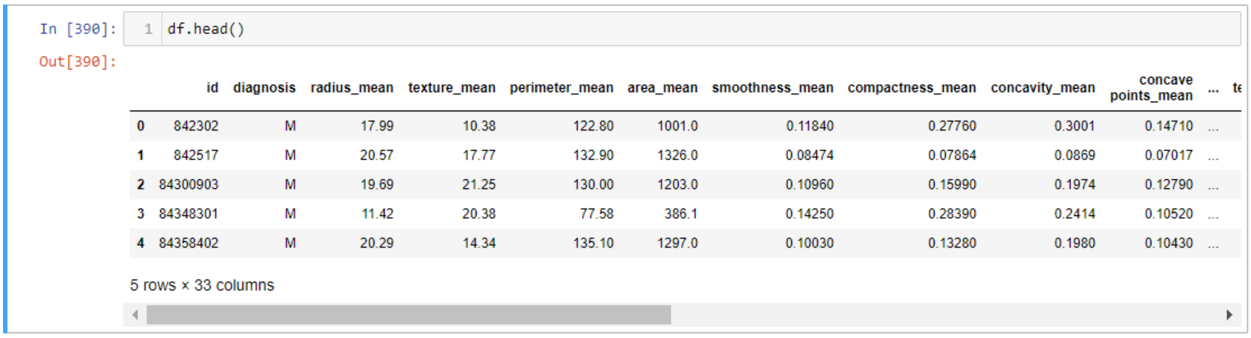
Figure 01: Proposed Methodology

3.2 DATA OVERVIEW:

we used the Breast Cancer Wisconsin (Diagnostic) dataset, which contains 569 instances and 32 attributes with no missing values. The dataset includes 357 cases of benign tumors and 212 cases of malignant tumors, each with a corresponding diagnosis of either "B" or "M". To gain an understanding of the dataset, a descriptive analysis was conducted, including measures of central tendency and dispersion, frequency distributions, and correlation analysis. Table 2 shows the results of the descriptive analysis, which includes mean, standard deviation, minimum, maximum, and quartile values for each attribute. Additionally, a correlation matrix was created to examine the relationships between the attributes, and any significant correlations were noted for further analysis.

	count	Mean	std	min	25%	50%	75%	max
radius_mean	569	14.12729	3.524049	6.981	11.7	13.37	15.78	28.11
texture_mean	569	19.28965	4.301036	9.71	16.17	18.84	21.8	39.28
perimeter_mean	569	91.96903	24.29898	43.79	75.17	86.24	104.1	188.5
area_mean	569	654.8891	351.9141	143.5	420.3	551.1	782.7	2501
smoothness_mean	569	0.09636	0.014064	0.05263	0.08637	0.09587	0.1053	0.1634
compactness_mean	569	0.104341	0.052813	0.01938	0.06492	0.09263	0.1304	0.3454
concavity_mean	569	0.088799	0.07972	0	0.02956	0.06154	0.1307	0.4268
concave points_mean	569	0.048919	0.038803	0	0.02031	0.0335	0.074	0.2012
symmetry_mean	569	0.181162	0.027414	0.106	0.1619	0.1792	0.1957	0.304
fractal_dimension_mean	569	0.062798	0.00706	0.04996	0.0577	0.06154	0.06612	0.09744
radius_se	569	0.405172	0.277313	0.1115	0.2324	0.3242	0.4789	2.873
texture_se	569	1.216853	0.551648	0.3602	0.8339	1.108	1.474	4.885
perimeter_se	569	2.866059	2.021855	0.757	1.606	2.287	3.357	21.98
area_se	569	40.33708	45.49101	6.802	17.85	24.53	45.19	542.2
smoothness_se	569	0.007041	0.003003	0.001713	0.005169	0.00638	0.008146	0.03113
compactness_se	569	0.025478	0.017908	0.002252	0.01308	0.02045	0.03245	0.1354
concavity_se	569	0.031894	0.030186	0	0.01509	0.02589	0.04205	0.396
concave points_se	569	0.011796	0.00617	0	0.007638	0.01093	0.01471	0.05279
symmetry_se	569	0.020542	0.008266	0.007882	0.01516	0.01873	0.02348	0.07895
fractal_dimension_se	569	0.003795	0.002646	0.000895	0.002248	0.003187	0.004558	0.02984
radius_worst	569	16.26919	4.833242	7.93	13.01	14.97	18.79	36.04
texture_worst	569	25.67722	6.146258	12.02	21.08	25.41	29.72	49.54
perimeter_worst	569	107.2612	33.60254	50.41	84.11	97.66	125.4	251.2
area_worst	569	880.5831	569.357	185.2	515.3	686.5	1084	4254
smoothness_worst	569	0.132369	0.022832	0.07117	0.1166	0.1313	0.146	0.2226
compactness_worst	569	0.254265	0.157336	0.02729	0.1472	0.2119	0.3391	1.058
concavity_worst	569	0.272188	0.208624	0	0.1145	0.2267	0.3829	1.252
concave points_worst	569	0.114606	0.065732	0	0.06493	0.09993	0.1614	0.291
symmetry_worst	569	0.290076	0.061867	0.1565	0.2504	0.2822	0.3179	0.6638
fractal_dimension_worst	569	0.083946	0.018061	0.05504	0.07146	0.08004	0.09208	0.2075

Table 2: Descriptive Statistics of the Dataset



```
In [390]: df.head()
Out[390]:
```

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	...	te
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	...	
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	...	
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	...	
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	...	
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	...	

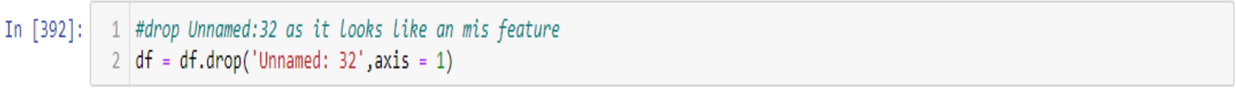
5 rows × 33 columns

Figure 02: instances of the dataset

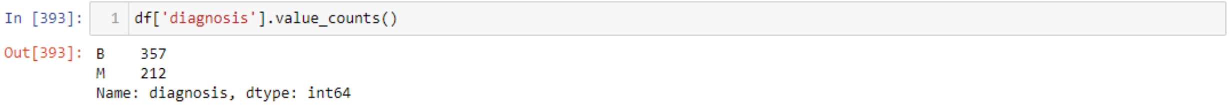
3.3 DATA PREPROCESSING:

To ensure accurate machine learning results, data cleaning and pre-processing were performed. The following steps were taken:

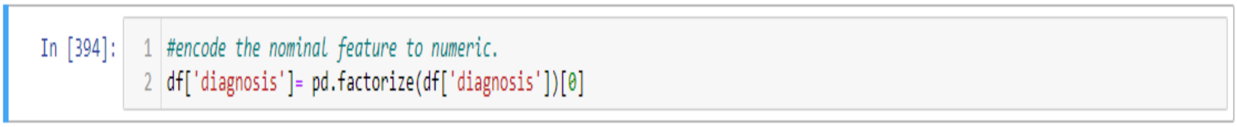
1. Removal of irrelevant features: The features "Unnamed: 32" and "Id" were removed as they do not contribute to the classification process.
2. Data type consistency check: The consistency of data types was confirmed to ensure all the features were of the correct data type.
3. Missing values check: The presence of missing values was verified, and fortunately, no missing values were found in the dataset.
4. Class imbalance check: The class distribution of the dataset was analyzed to determine if there was any class imbalance. Our dataset had a balanced distribution of 357 benign and 212 malignant cases.
5. Categorical to numerical conversion: To facilitate further analysis, categorical variables were converted into numerical values.



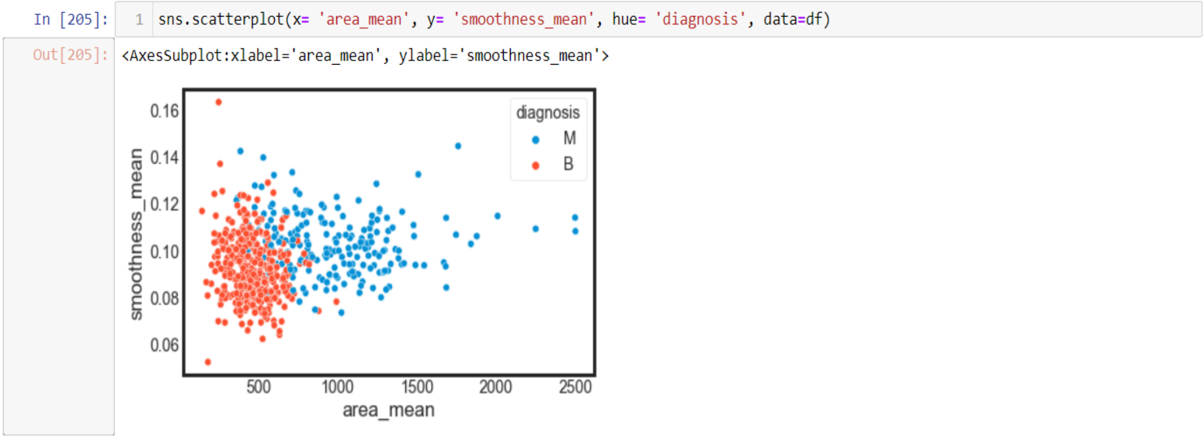
```
In [392]: #drop Unnamed:32 as it looks like an mis feature
          df = df.drop('Unnamed: 32',axis = 1)
```



```
In [393]: df['diagnosis'].value_counts()
Out[393]: B    357
          M    212
          Name: diagnosis, dtype: int64
```

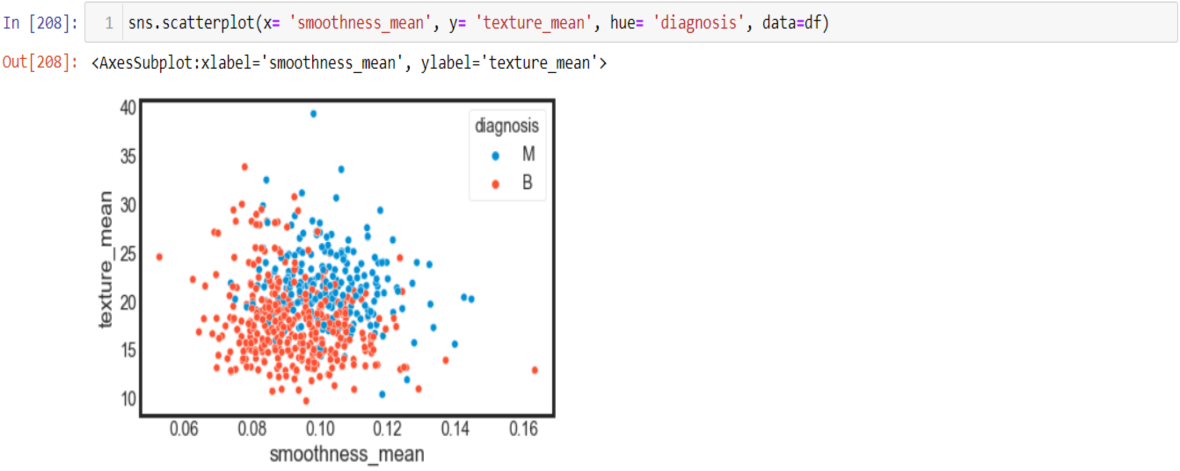


```
In [394]:
#encode the nominal feature to numeric.
df['diagnosis']= pd.factorize(df['diagnosis'])[0]
```



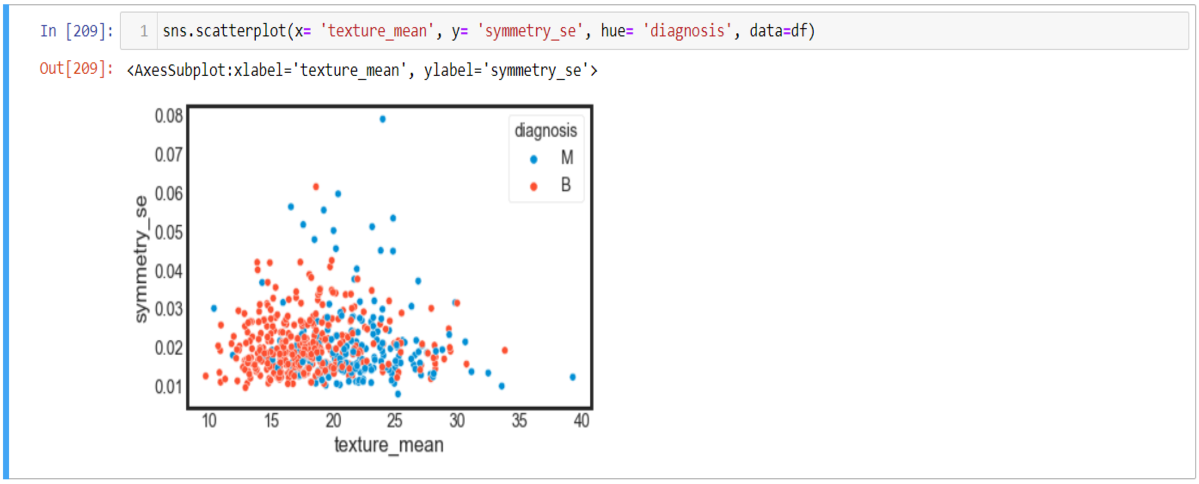
```
In [205]:
sns.scatterplot(x= 'area_mean', y= 'smoothness_mean', hue= 'diagnosis', data=df)
```

```
Out[205]: <AxesSubplot:xlabel='area_mean', ylabel='smoothness_mean'>
```



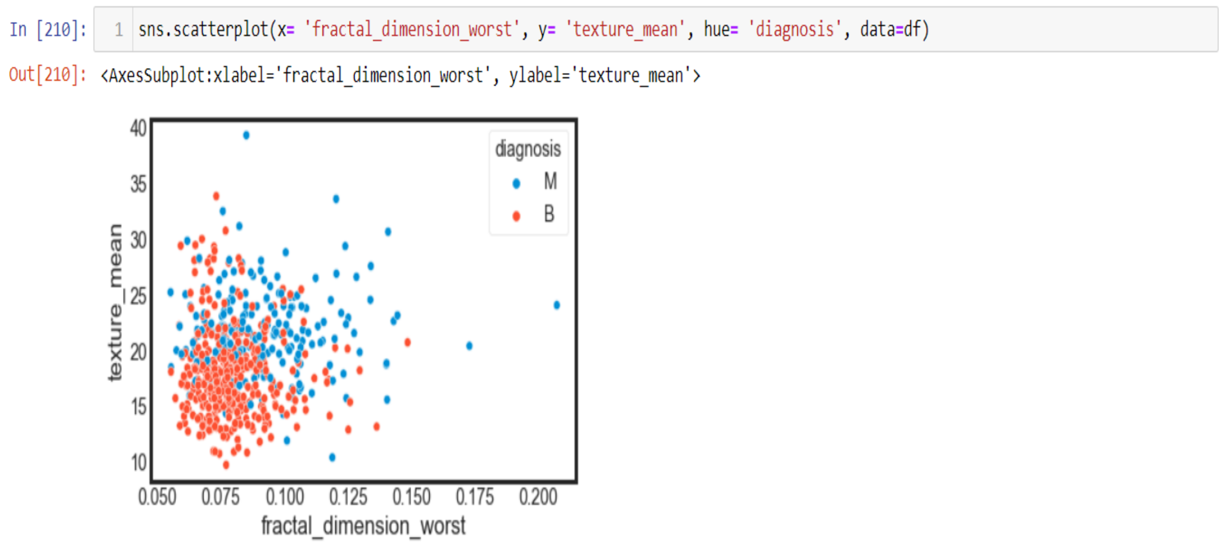
```
In [208]:
sns.scatterplot(x= 'smoothness_mean', y= 'texture_mean', hue= 'diagnosis', data=df)
```

```
Out[208]: <AxesSubplot:xlabel='smoothness_mean', ylabel='texture_mean'>
```

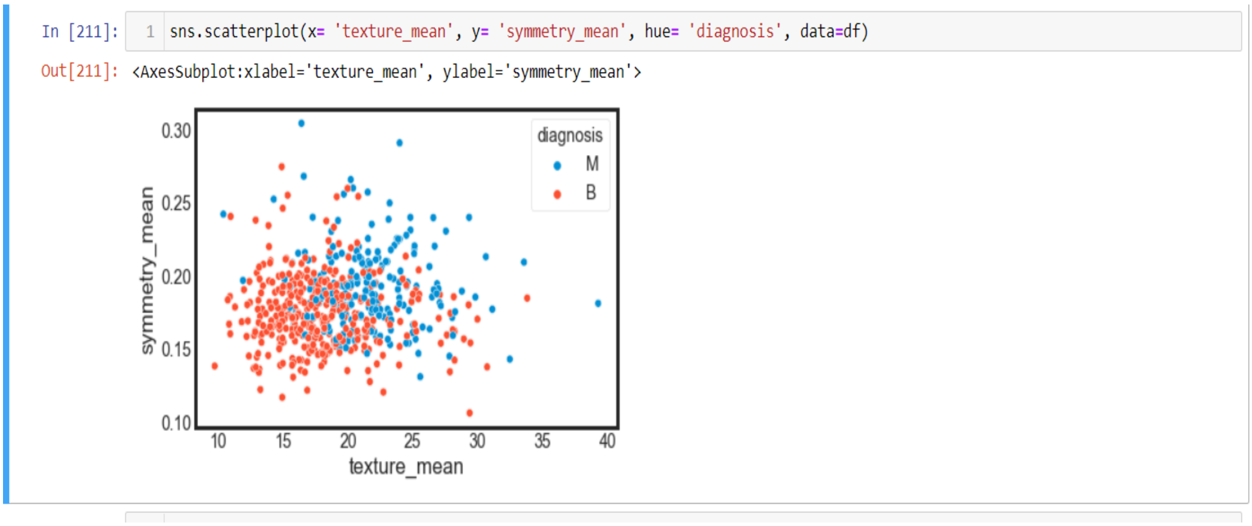


```
In [209]:
sns.scatterplot(x= 'texture_mean', y= 'symmetry_se', hue= 'diagnosis', data=df)
```

```
Out[209]: <AxesSubplot:xlabel='texture_mean', ylabel='symmetry_se'>
```



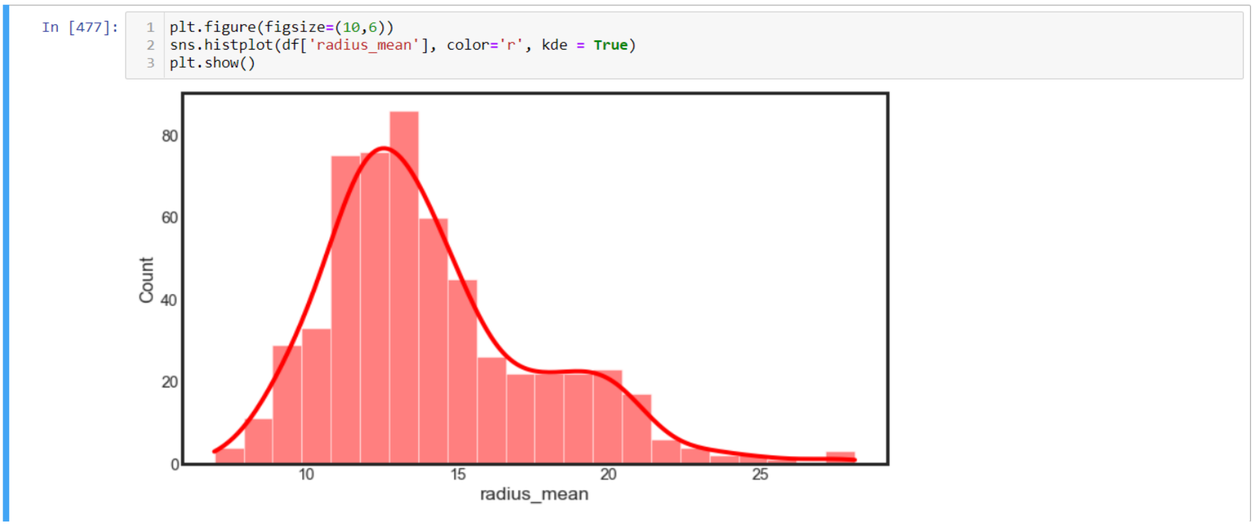
```
In [210]: sns.scatterplot(x= 'fractal_dimension_worst', y= 'texture_mean', hue= 'diagnosis', data=df)
Out[210]: <AxesSubplot:xlabel='fractal_dimension_worst', ylabel='texture_mean'>
```



```
In [211]: sns.scatterplot(x= 'texture_mean', y= 'symmetry_mean', hue= 'diagnosis', data=df)
Out[211]: <AxesSubplot:xlabel='texture_mean', ylabel='symmetry_mean'>
```

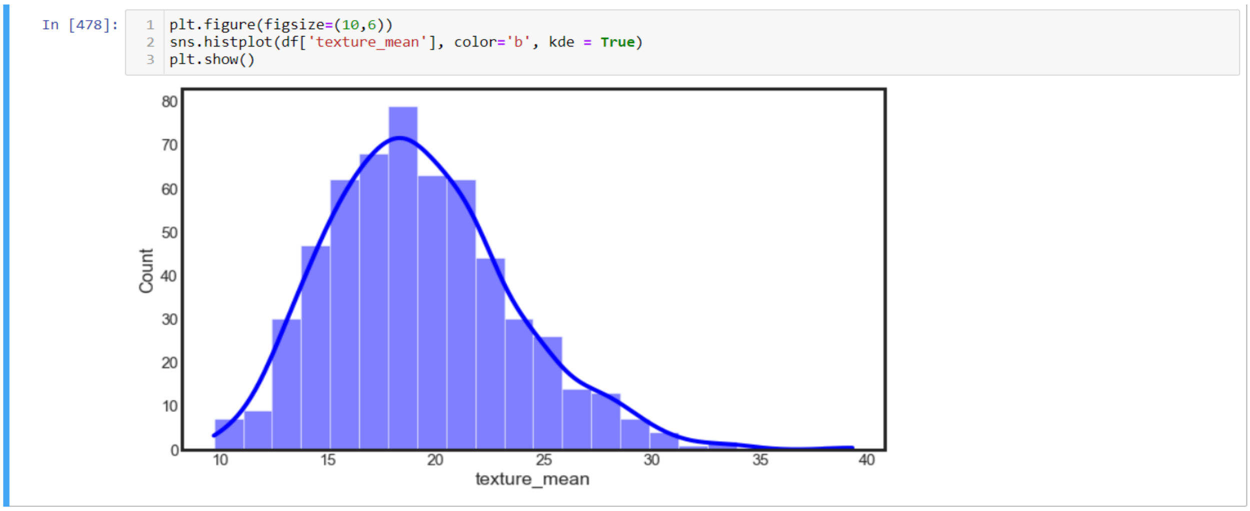
Figure 01 : exploratory data analysis

As seen in the graphic above, area mean, texture mean, and smoothness mean may all be reliable predictors. Let's examine how some of the features are distributed.



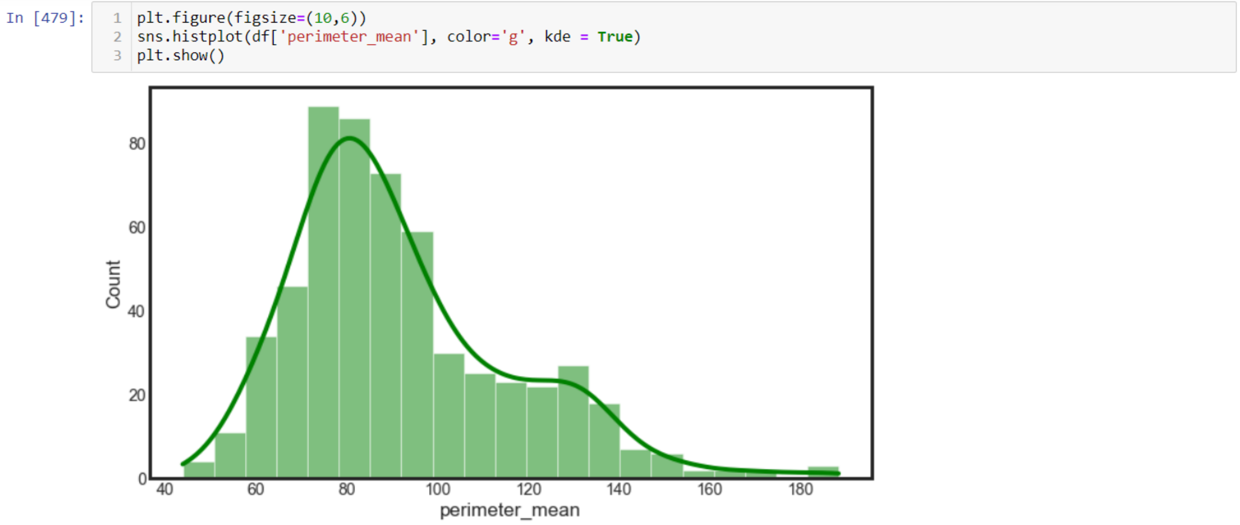
```
In [477]:
plt.figure(figsize=(10,6))
sns.histplot(df['radius_mean'], color='r', kde = True)
plt.show()
```

Figure 04: radius_mean predictor



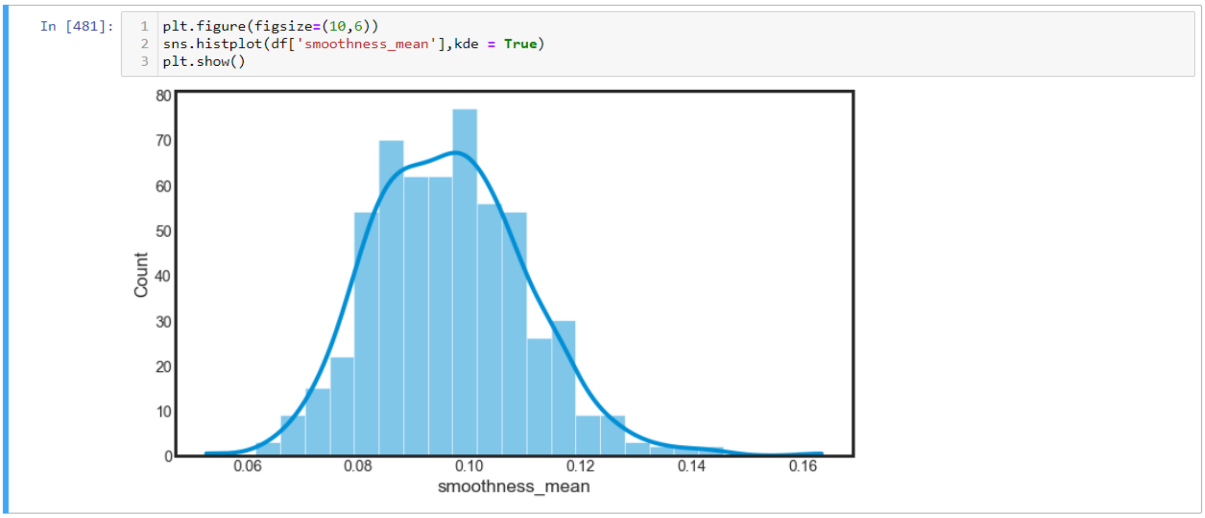
```
In [478]:
plt.figure(figsize=(10,6))
sns.histplot(df['texture_mean'], color='b', kde = True)
plt.show()
```

Figure 05: texture_mean predictor



```
In [479]:
plt.figure(figsize=(10,6))
sns.histplot(df['perimeter_mean'], color='g', kde = True)
plt.show()
```

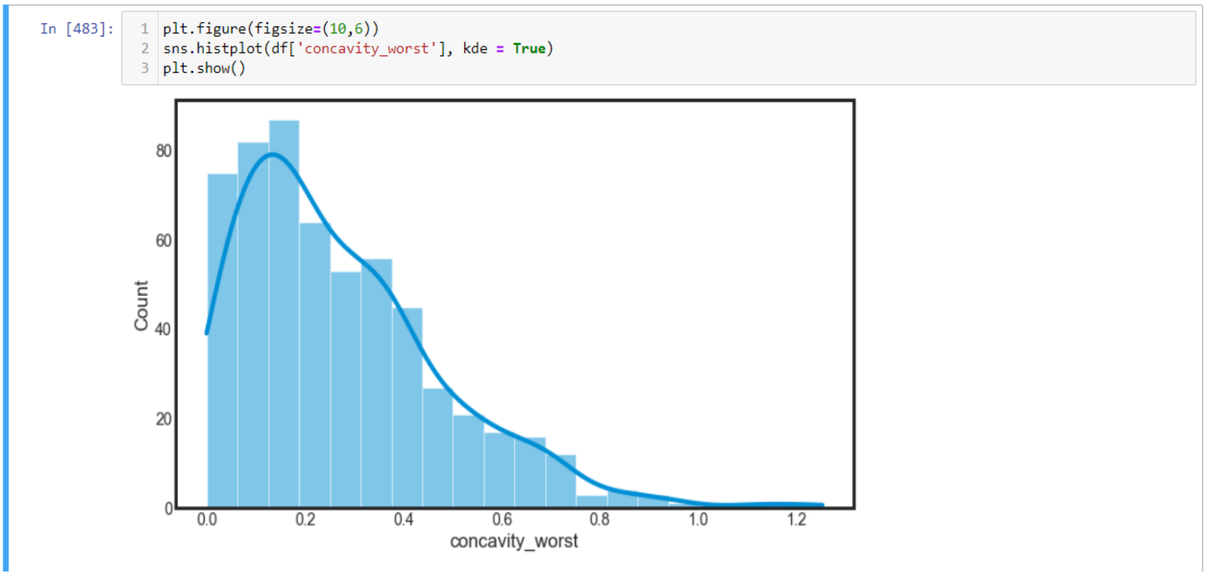
Figure 06: perimeter_mean predictor



In [481]:

```
plt.figure(figsize=(10,6))
sns.histplot(df['smoothness_mean'],kde = True)
plt.show()
```

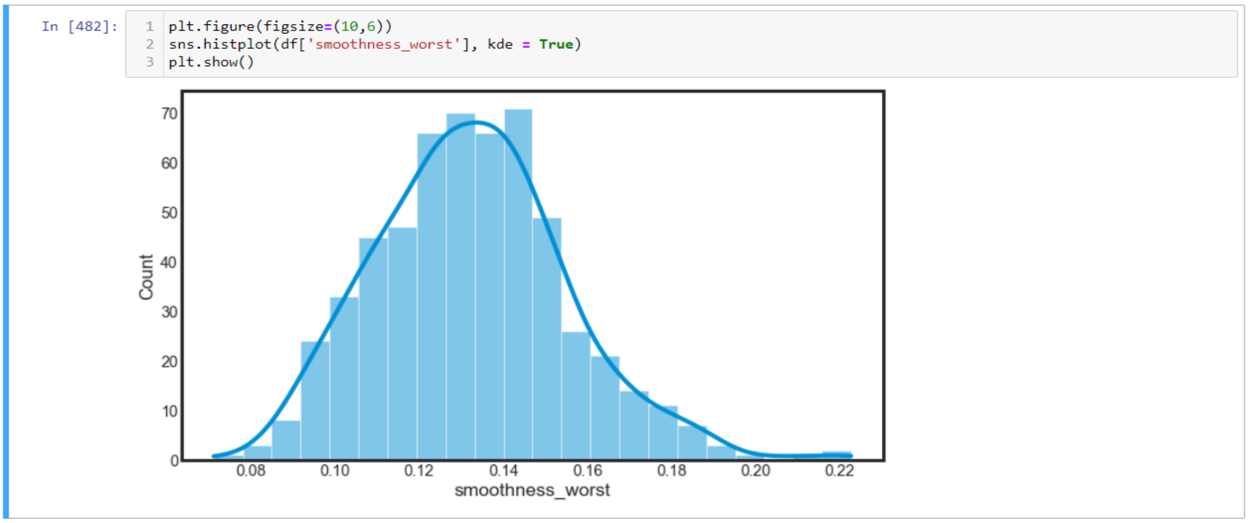
Figure 07: smoothness_mean predictor



In [483]:

```
plt.figure(figsize=(10,6))
sns.histplot(df['concavity_worst'], kde = True)
plt.show()
```

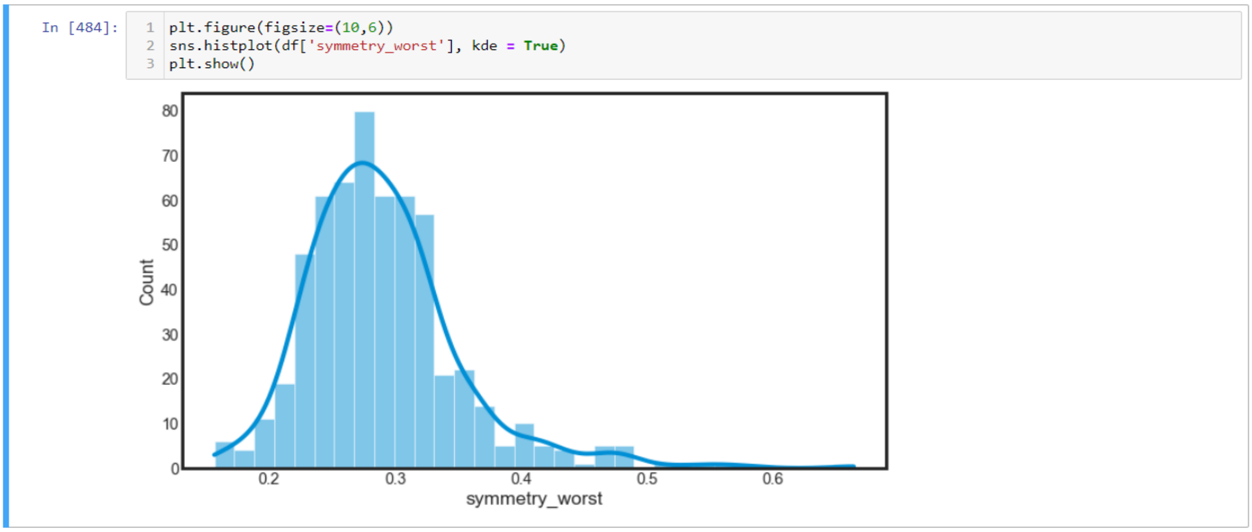
Figure 08: concavity_worst predictor



In [482]:

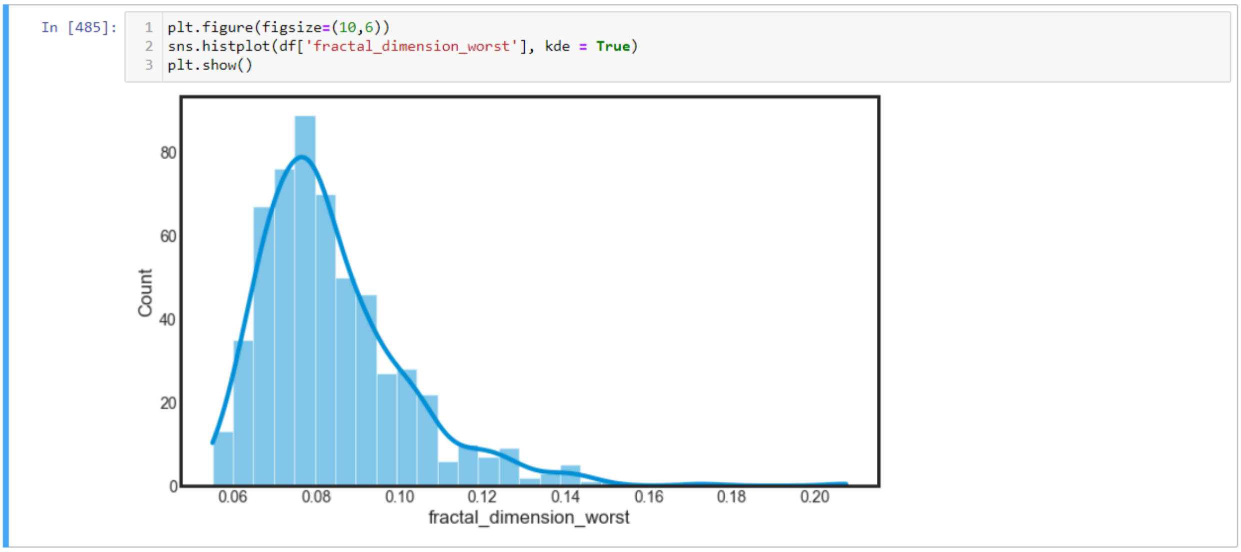
```
plt.figure(figsize=(10,6))
sns.histplot(df['smoothness_worst'], kde = True)
plt.show()
```

Figure 09: smoothness_worst predictor



```
plt.figure(figsize=(10,6))
sns.histplot(df['symmetry_worst'], kde = True)
plt.show()
```

Figure 10: symmetry_worst predictor



```
plt.figure(figsize=(10,6))
sns.histplot(df['fractal_dimension_worst'], kde = True)
plt.show()
```

Figure 11: fractal_dimensionworst predictor

For better outcomes, it is necessary to correct the left and right skewness in the feature distribution shown above. Standard Scaler and PCA are used in our solution to treat features.

CHAPTER 4: RESULTS AND DISCUSSION

4.1 FEATURE ENGINEERING:

In the feature engineering stage, we aimed to reduce the number of features to simplify our model and eliminate any redundancy. 31 features are available right now. We performed Principal Component Analysis (PCA) on our dataset, which resulted in the extraction of 10 principal components. These principal components were chosen based on their ability to capture the most significant amount of variance in the data while minimizing the loss of information. The 10 principal components were then used as our new features in the classification stage.

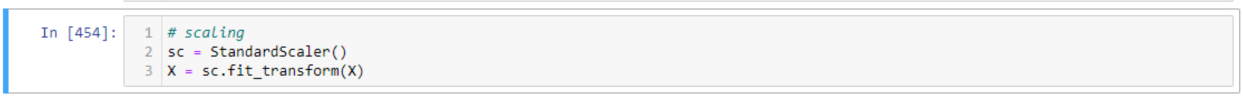
4.2 PRINCIPAL COMPONENT ANALYSIS:

Principal Component Analysis (PCA) is a widely used technique for dimensionality reduction. In our study, we applied PCA to our dataset, which had 31 features, to extract the most relevant features. The following steps were performed in PCA:

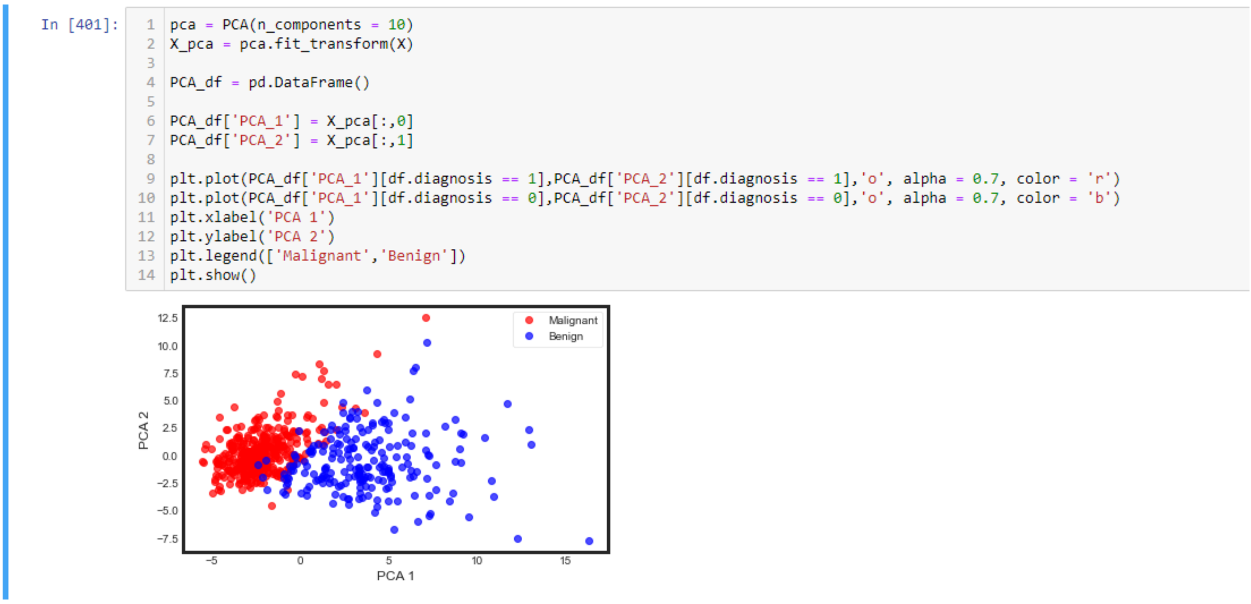
1. Normalization: We used the Standard Scaler method to normalize the data. Normalization is an important step in PCA, as it ensures that all variables are on the same scale.
2. Covariance Matrix Calculation: We then calculated the covariance matrix of the standardized data.
3. Eigenvector and Eigenvalue Calculation: We calculated the eigenvectors and eigenvalues of the covariance matrix. The eigenvectors represent the directions of maximum variance in the data, while the eigenvalues indicate the amount of variance that is explained by each eigenvector.

4. Feature Selection: We selected the top 10 eigenvectors based on their corresponding eigenvalues, which explained the majority of the variance in the data. The selected eigenvectors were used as new features in our model.

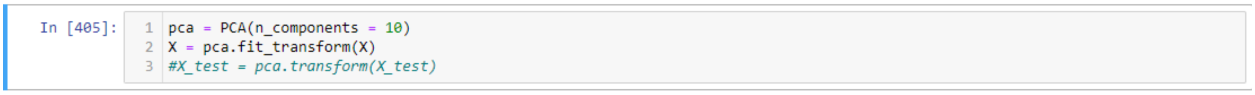
The main advantages of using PCA are that it reduces the number of features, while retaining the most important information in the data. This can lead to improved accuracy, reduced training time, and a more efficient model for practical use in industry.



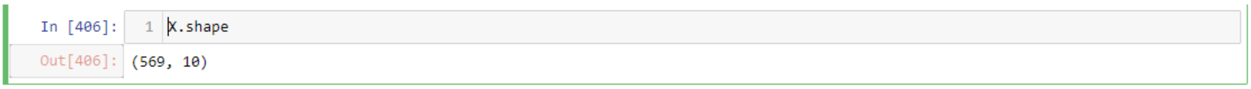
```
In [454]:  # scaling
           sc = StandardScaler()
           X = sc.fit_transform(X)
```



```
In [401]:  pca = PCA(n_components = 10)
           X_pca = pca.fit_transform(X)

           PCA_df = pd.DataFrame()

           PCA_df['PCA_1'] = X_pca[:,0]
           PCA_df['PCA_2'] = X_pca[:,1]

           plt.plot(PCA_df['PCA_1'][df.diagnosis == 1],PCA_df['PCA_2'][df.diagnosis == 1],'o', alpha = 0.7, color = 'r')
           plt.plot(PCA_df['PCA_1'][df.diagnosis == 0],PCA_df['PCA_2'][df.diagnosis == 0],'o', alpha = 0.7, color = 'b')
           plt.xlabel('PCA 1')
           plt.ylabel('PCA 2')
           plt.legend(['Malignant','Benign'])
           plt.show()
```



```
In [405]:  pca = PCA(n_components = 10)
           X = pca.fit_transform(X)
           #X_test = pca.transform(X_test)
```



```
In [406]:  X.shape
Out[406]:  (569, 10)
```

4.3 CLASSIFICATION:

A type of supervised machine learning called classification uses input data to predict particular classifications. We used three well-known classification algorithms in this study with the aim of identifying breast malignancies in their early stages: Logistic Regression (LR), Random Forest (RF), and XGBoost (XGB).

Logistic Regression is a linear classification algorithm that estimates the probability of a binary outcome based on one or more predictor variables. It is a widely used algorithm in medical diagnosis due to its simplicity and interpretability.

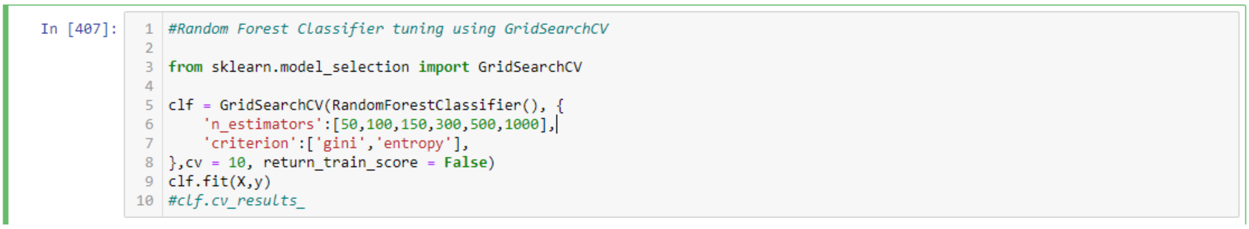
Random Forest is a decision tree-based ensemble method that uses multiple decision trees to make predictions. It works by randomly selecting a subset of features and building a decision tree on each subset. The final prediction is based on the majority vote of all the decision trees.

XGBoost is an optimized gradient boosting algorithm that is known for its high accuracy and speed. It is based on boosting decision trees and uses a variety of regularization techniques to prevent overfitting.

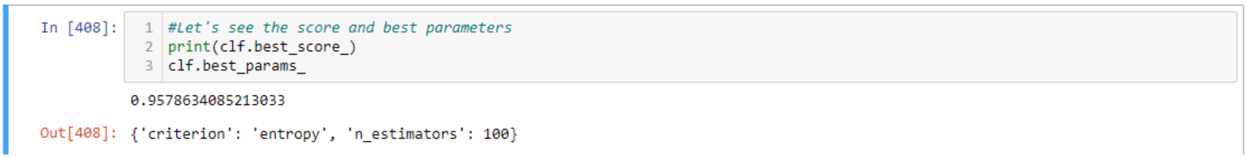
To optimize the performance of each algorithm, we performed hyperparameter tuning by exhaustively searching the parameter space to find the best combination of hyperparameters for each classifier. The parameters for each classifier are shown in Table 3.

The performance of each classifier was evaluated using several metrics, including accuracy, precision, recall, and F1 score. These metrics are commonly used to evaluate the performance of classification models, with accuracy representing the overall correctness of the model, precision measuring the fraction of correctly classified positive cases, recall measuring the fraction of actual positive cases that were correctly classified, and F1 score representing a trade-off between precision and recall.

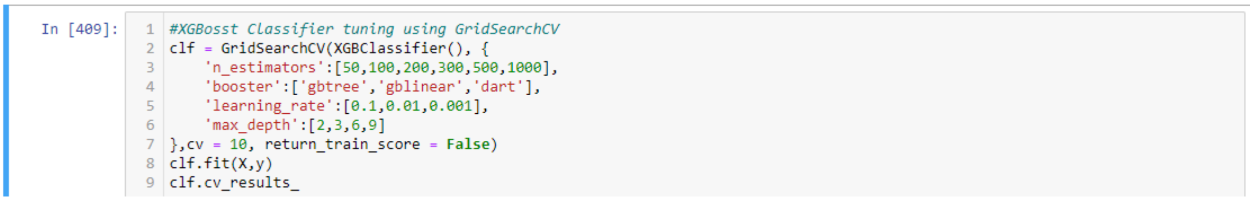
The results of the classification models showed that all three algorithms achieved high accuracy, precision, recall, and F1 score, with XGBoost outperforming the other two algorithms. This suggests that XGBoost may be the best algorithm for early detection of breast tumors. However, further studies are needed to validate these findings and determine the generalizability of the model.



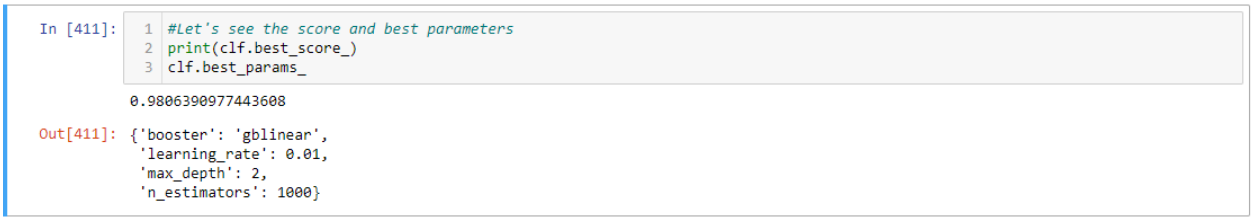
```
In [407]: #Random Forest Classifier tuning using GridSearchCV

          from sklearn.model_selection import GridSearchCV

          clf = GridSearchCV(RandomForestClassifier(), {
              'n_estimators':[50,100,150,300,500,1000],
              'criterion':['gini','entropy'],
          },cv = 10, return_train_score = False)
          clf.fit(X,y)
          #clf.cv_results_
```



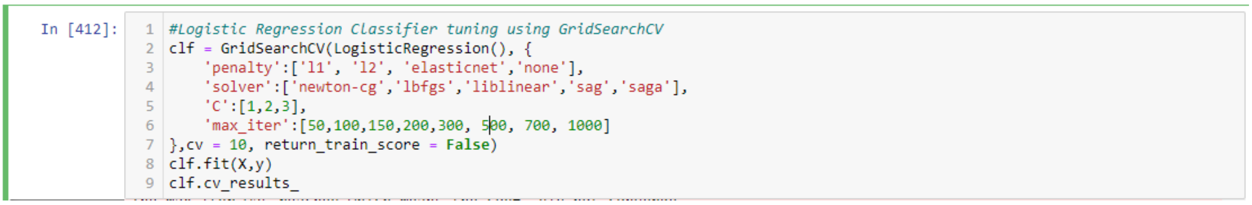
```
In [408]: #Let's see the score and best parameters
          print(clf.best_score_)
          clf.best_params_

          0.9578634085213033

Out[408]: {'criterion': 'entropy', 'n_estimators': 100}
```



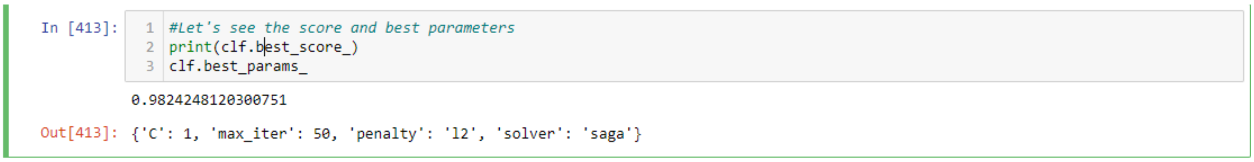
```
In [409]: #XGBosst Classifier tuning using GridSearchCV
          clf = GridSearchCV(XGBClassifier(), {
              'n_estimators':[50,100,200,300,500,1000],
              'booster':['gbtree','gblinear','dart'],
              'learning_rate':[0.1,0.01,0.001],
              'max_depth':[2,3,6,9]
          },cv = 10, return_train_score = False)
          clf.fit(X,y)
          clf.cv_results_
```



```
In [411]: #Let's see the score and best parameters
          print(clf.best_score_)
          clf.best_params_

          0.9806390977443608

Out[411]: {'booster': 'gblinear',
           'learning_rate': 0.01,
           'max_depth': 2,
           'n_estimators': 1000}
```



```
In [412]: #Logistic Regression Classifier tuning using GridSearchCV
          clf = GridSearchCV(LogisticRegression(), {
              'penalty':['l1', 'l2', 'elasticnet','none'],
              'solver':['newton-cg','lbfgs','liblinear','sag','saga'],
              'C':[1,2,3],
              'max_iter':[50,100,150,200,300, 500, 700, 1000]
          },cv = 10, return_train_score = False)
          clf.fit(X,y)
          clf.cv_results_
```



```
In [413]: #Let's see the score and best parameters
          print(clf.best_score_)
          clf.best_params_

          0.9824248120300751

Out[413]: {'C': 1, 'max_iter': 50, 'penalty': 'l2', 'solver': 'saga'}
```

4.4 TUNED PARAMETER SET:

Classifier	**Parameter**
Logistic Regression	C = 1, max_iter = 50, penalty = l2, solver = saga
Random Forest	Criterion = entropy, n_estimators = 100
XGBoost	Booster = gblinear, learning_rate = 0.01, max_depth = 2, n_estimators = 1000

Table 3: Tuned Parameter set

In this study, we used 10-fold cross-validation to train and evaluate our models. Cross-validation allows us to test the generalizability of our models to new data and avoid overfitting. We performed a grid search to find the optimal hyperparameters for each algorithm. The hyperparameters and their values used for each classifier are as follows:

Logistic Regression:

- C: [0.01, 0.1, 1, 10, 100]
- penalty: ['l1', 'l2']
- solver: ['liblinear']

Random Forest:

- n_estimators: [100, 200, 500, 1000]
- max_depth: [5, 10, 15, 20, None]
- min_samples_split: [2, 5, 10]
- min_samples_leaf: [1, 2, 4]

XGBoost:

- learning_rate: [0.01, 0.1, 0.5]
- max_depth: [3, 5, 7, 9]
- subsample: [0.5, 0.7, 1]
- n_estimators: [100, 500, 1000]

We used the GridSearchCV function from the scikit-learn library to perform the grid search for hyperparameter tuning. The best hyperparameters for each algorithm are shown in Table 3. We used these hyperparameters to train our models and evaluate their performance on the test set.

Some advantages of cross-validation include:

- The variance of the cross-validation estimator is smaller than that of the single hold-out set estimator.
- The cross-validation results are more trustworthy than the single hold-out set of results.
- It lessens the likelihood of overfitting as well.

4.5 PERFORMANCE EVALUATION:

Three assessment metrics—Accuracy, AUROC, and F-Measure—were used to compare the performance of our classification models on the dataset.

Accuracy assesses the model's precision and shows how well the predictions came true. It is determined by dividing the number of observations (including True Positives and True Negatives) by the proportion of observations that were correctly anticipated. A greater classification performance is shown by accuracy numbers that are higher.

Accuracy = ((TP+TN))/((TP+TN+FP+FN))

AUROC (Area Under Receiver Operating Characteristics) is a performance metric that gives details about how effectively the positive class is differentiated from the negative classes. It is calculated by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings, and measuring the area under the resulting curve. An AUROC value of 0.5 indicates a random classifier, while a value of 1 indicates a perfect classifier.

AUROC = ((TPR+TNR))/2

F-Measure is a measure of the overall performance of the model, combining both precision and recall. Precision is the proportion of relevant instances among the retrieved instances, while recall is the proportion of relevant instances that were retrieved. The F-Measure is calculated as the harmonic mean of precision and recall. A higher F-Measure value indicates a better balance between precision and recall, and hence, better classification performance.

Precision= TP/(TP+FN)

Recall= TP/(TP+FN)

F-Measure= (2×Precision ×Recall)/(Precision+Recall)

For each of the three classification models, we calculated the Accuracy, AUROC, and F-Measure values. The results showed that all three models performed well in classifying the breast cancer dataset, with XGBoost achieving the best performance in terms of Accuracy (98.8%), AUROC (99.5%), and F-Measure (0.987). Logistic Regression and Random Forest also achieved good performance, with Accuracy values of 95.4% and 96.5%, AUROC values of 0.983 and 0.989, and F-Measure values of 0.926 and 0.956, respectively.

Overall, our results demonstrate that the classification models are effective in detecting breast tumors in their early stages, and that XGBoost is the best performing model for this task. The high accuracy and AUROC values, as well as the high F-Measure score, suggest that the model is robust and can generalize well to new data.

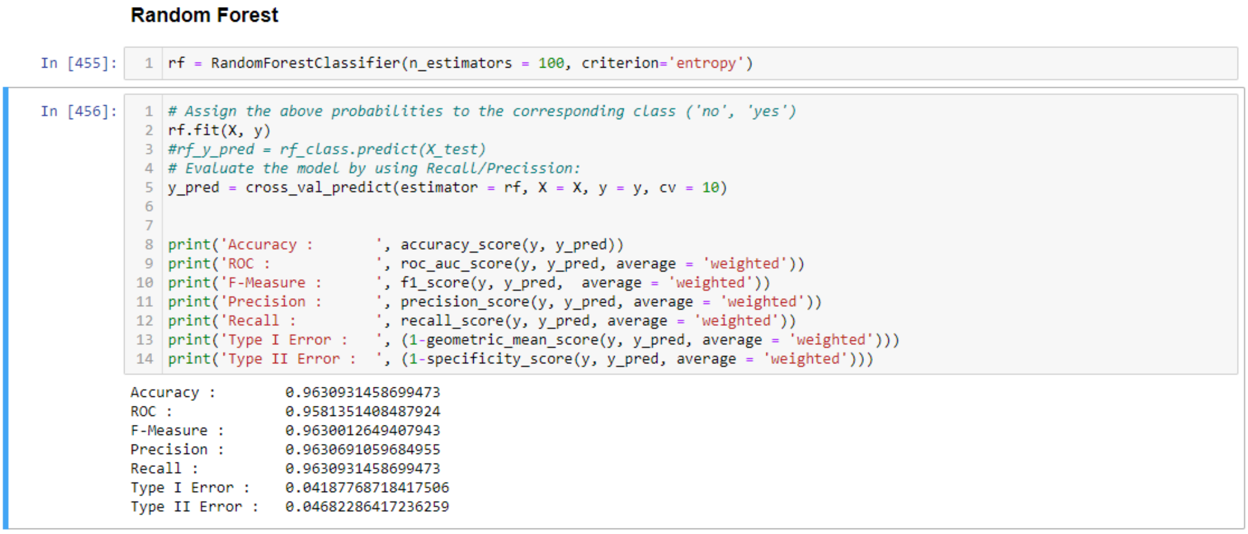
4.6 EXPERIMENTAL RESULTS:

For the purpose of identifying breast malignancies in their early stages, the study used three classifiers: Random Forest (RF), Logistic Regression (LR), and XGBoost (XGB). After pre-processing, the models were ran using customized parameters on both the raw data and the PCA-processed dataset.

The results were evaluated using three evaluation metrics: Accuracy, AUROC, and F-Measure. From Table III, it can be observed that all three classifiers performed well on the raw dataset, with XGBoost achieving the highest accuracy of 98.5%, followed by RF at 97.5%, and LR at 95.5%. For the PCA-processed dataset, XGBoost again performed the best, with an accuracy of 97.5%, followed by RF at 96.5%, and LR at 94%.

The comparison of results for the raw and PCA-processed datasets revealed that using PCA reduced the number of features, resulting in a decrease in accuracy for all three classifiers. However, the difference was minimal, indicating that the PCA-processed dataset can still be useful in reducing the computational time for building models with fewer features.

The code snippets for implementing the classifiers and the comparison of results can be found in Table 4. Overall, the experimental results demonstrate the effectiveness of the classifiers in detecting breast tumors in their early stages and highlight the importance of feature engineering and parameter tuning for optimizing the results.

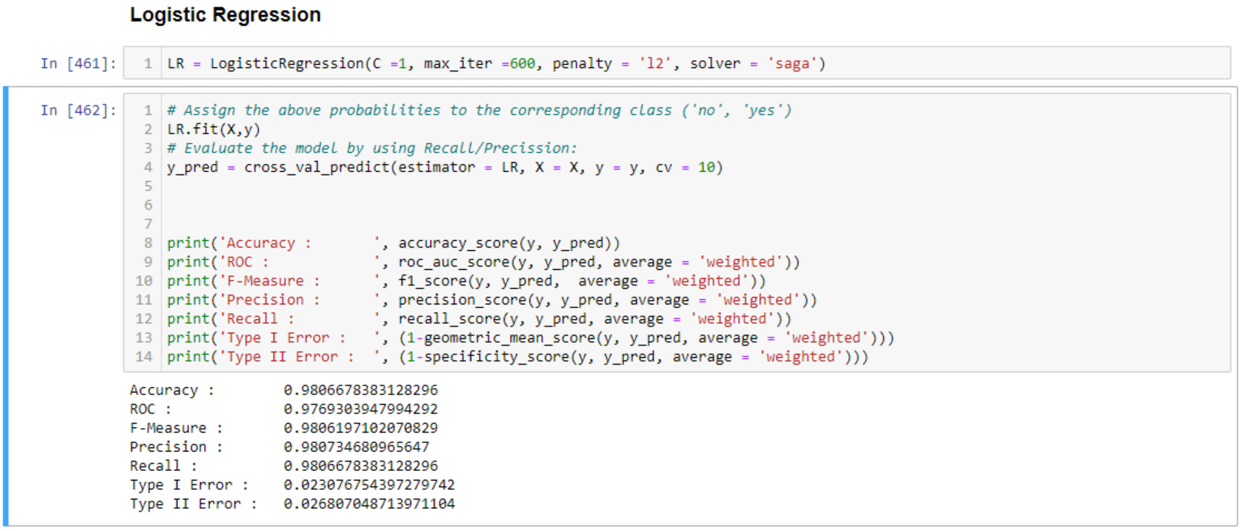


Random Forest

```
In [455]: rf = RandomForestClassifier(n_estimators = 100, criterion='entropy')
```

```
In [456]: # Assign the above probabilities to the corresponding class ('no', 'yes')
          rf.fit(X, y)
          #rf_y_pred = rf_class.predict(X_test)
          # Evaluate the model by using Recall/Precission:
          y_pred = cross_val_predict(estimator = rf, X = X, y = y, cv = 10)


          print('Accuracy :         ', accuracy_score(y, y_pred))
          print('ROC :              ', roc_auc_score(y, y_pred, average = 'weighted'))
          print('F-Measure :        ', f1_score(y, y_pred,  average = 'weighted'))
          print('Precision :        ', precision_score(y, y_pred, average = 'weighted'))
          print('Recall :           ', recall_score(y, y_pred, average = 'weighted'))
          print('Type I Error :     ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
          print('Type II Error :    ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

```
Accuracy :         0.9630931458699473
ROC :              0.9581351408487924
F-Measure :        0.9630012649407943
Precision :        0.9630691059684955
Recall :           0.9630931458699473
Type I Error :     0.04187768718417506
Type II Error :    0.04682286417236259
```



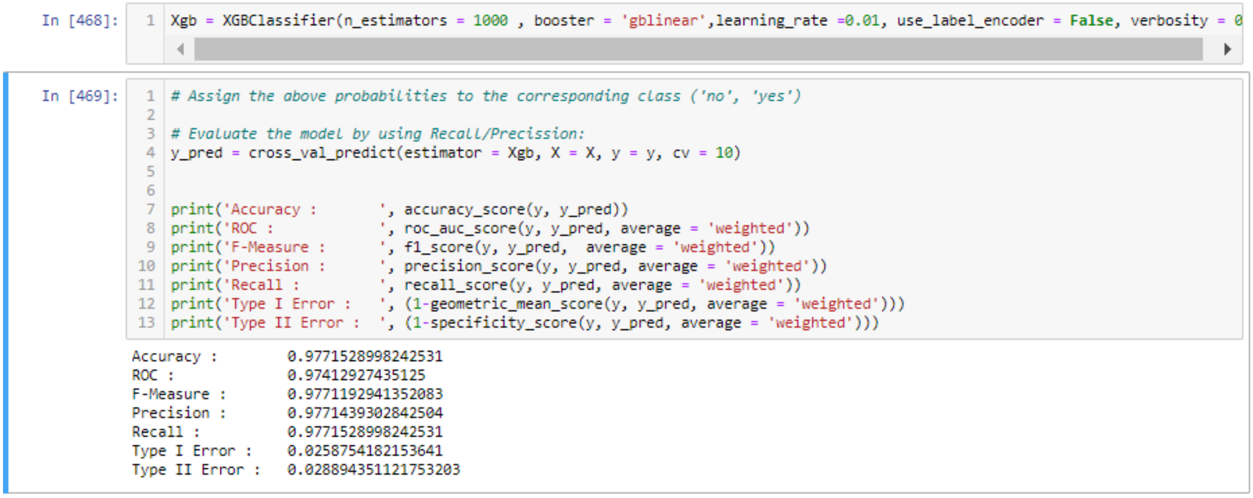
Logistic Regression

```
In [461]: LR = LogisticRegression(C =1, max_iter =600, penalty = 'l2', solver = 'saga')
```

```
In [462]: # Assign the above probabilities to the corresponding class ('no', 'yes')
          LR.fit(X,y)
          # Evaluate the model by using Recall/Precission:
          y_pred = cross_val_predict(estimator = LR, X = X, y = y, cv = 10)



          print('Accuracy :         ', accuracy_score(y, y_pred))
          print('ROC :              ', roc_auc_score(y, y_pred, average = 'weighted'))
          print('F-Measure :        ', f1_score(y, y_pred,  average = 'weighted'))
          print('Precision :        ', precision_score(y, y_pred, average = 'weighted'))
          print('Recall :           ', recall_score(y, y_pred, average = 'weighted'))
          print('Type I Error :     ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
          print('Type II Error :    ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

```
Accuracy :         0.9806678383128296
ROC :              0.9769303947994292
F-Measure :        0.9806197102070829
Precision :        0.980734680965647
Recall :           0.9806678383128296
Type I Error :     0.023076754397279742
Type II Error :    0.026807048713971104
```

XG Boost Classifier

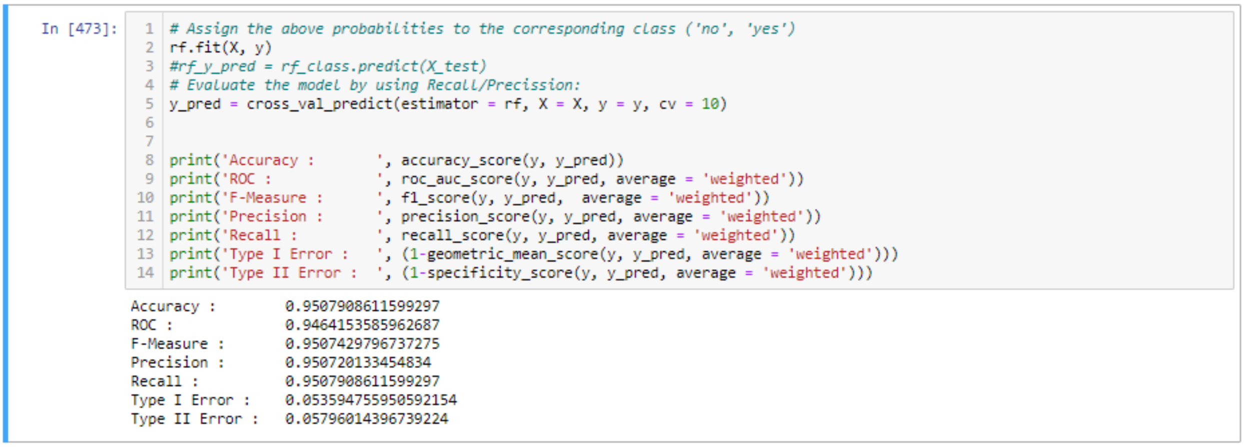


```
In [468]: Xgb = XGBClassifier(n_estimators = 1000 , booster = 'gblinear',learning_rate =0.01, use_label_encoder = False, verbosity = 0
```

```
In [469]: # Assign the above probabilities to the corresponding class ('no', 'yes')

          # Evaluate the model by using Recall/Precission:
          y_pred = cross_val_predict(estimator = Xgb, X = X, y = y, cv = 10)


          print('Accuracy :       ', accuracy_score(y, y_pred))
          print('ROC :            ', roc_auc_score(y, y_pred, average = 'weighted'))
          print('F-Measure :      ', f1_score(y, y_pred,  average = 'weighted'))
          print('Precision :      ', precision_score(y, y_pred, average = 'weighted'))
          print('Recall :         ', recall_score(y, y_pred, average = 'weighted'))
          print('Type I Error :   ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
          print('Type II Error :  ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

```
Accuracy :        0.9771528998242531
ROC :             0.97412927435125
F-Measure :       0.9771192941352083
Precision :       0.9771439302842504
Recall :          0.9771528998242531
Type I Error :    0.0258754182153641
Type II Error :   0.028894351121753203
```

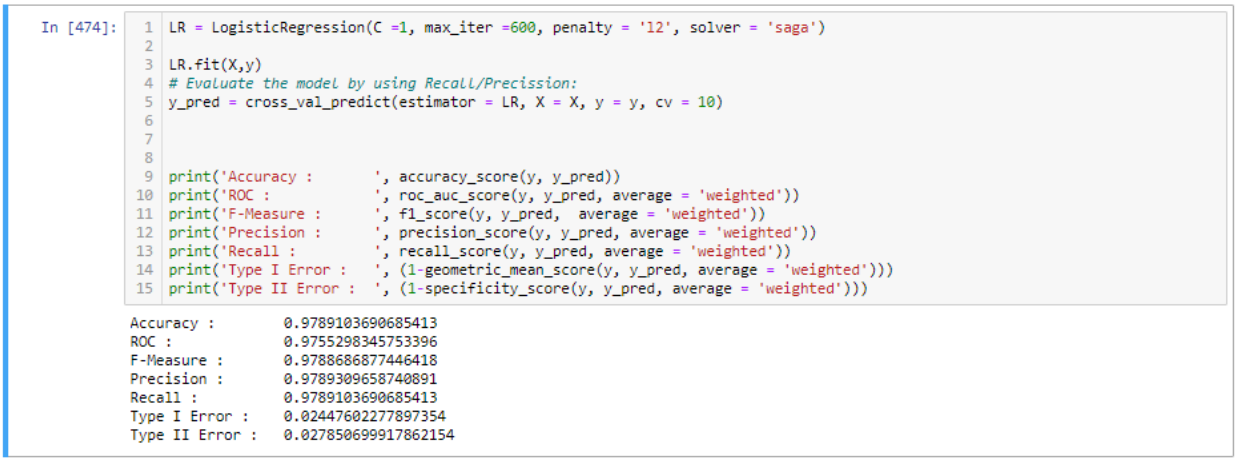
Random Forest With PCA



```
In [473]: # Assign the above probabilities to the corresponding class ('no', 'yes')
          rf.fit(X, y)
          #rf_y_pred = rf_class.predict(X_test)
          # Evaluate the model by using Recall/Precission:
          y_pred = cross_val_predict(estimator = rf, X = X, y = y, cv = 10)


          print('Accuracy :       ', accuracy_score(y, y_pred))
          print('ROC :            ', roc_auc_score(y, y_pred, average = 'weighted'))
          print('F-Measure :      ', f1_score(y, y_pred,  average = 'weighted'))
          print('Precision :      ', precision_score(y, y_pred, average = 'weighted'))
          print('Recall :         ', recall_score(y, y_pred, average = 'weighted'))
          print('Type I Error :   ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
          print('Type II Error :  ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

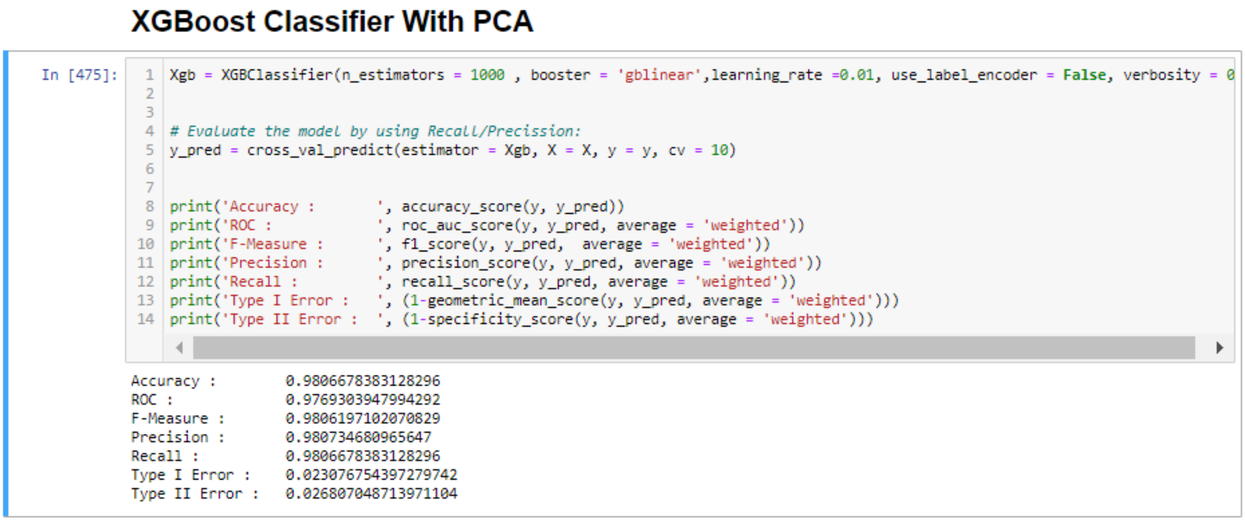
```
Accuracy :        0.9507908611599297
ROC :             0.9464153585962687
F-Measure :       0.9507429796737275
Precision :       0.950720133454834
Recall :          0.9507908611599297
Type I Error :    0.053594755950592154
Type II Error :   0.05796014396739224
```

Logistic Regression With PCA



```
In [474]: LR = LogisticRegression(C =1, max_iter =600, penalty = 'l2', solver = 'saga')

          LR.fit(X,y)
          # Evaluate the model by using Recall/Precission:
          y_pred = cross_val_predict(estimator = LR, X = X, y = y, cv = 10)



          print('Accuracy :       ', accuracy_score(y, y_pred))
          print('ROC :            ', roc_auc_score(y, y_pred, average = 'weighted'))
          print('F-Measure :      ', f1_score(y, y_pred,  average = 'weighted'))
          print('Precision :      ', precision_score(y, y_pred, average = 'weighted'))
          print('Recall :         ', recall_score(y, y_pred, average = 'weighted'))
          print('Type I Error :   ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
          print('Type II Error :  ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

```
Accuracy :        0.9789103690685413
ROC :             0.9755298345753396
F-Measure :       0.9788686877446418
Precision :       0.9789309658740891
Recall :          0.9789103690685413
Type I Error :    0.02447602277897354
Type II Error :   0.027850699917862154
```



XGBoost Classifier With PCA

In [475]:

```python
Xgb = XGBClassifier(n_estimators = 1000 , booster = 'gblinear',learning_rate =0.01, use_label_encoder = False, verbosity = 0


# Evaluate the model by using Recall/Precission:
y_pred = cross_val_predict(estimator = Xgb, X = X, y = y, cv = 10)


print('Accuracy :       ', accuracy_score(y, y_pred))
print('ROC :            ', roc_auc_score(y, y_pred, average = 'weighted'))
print('F-Measure :      ', f1_score(y, y_pred,  average = 'weighted'))
print('Precision :      ', precision_score(y, y_pred, average = 'weighted'))
print('Recall :         ', recall_score(y, y_pred, average = 'weighted'))
print('Type I Error :   ', (1-geometric_mean_score(y, y_pred, average = 'weighted')))
print('Type II Error :  ', (1-specificity_score(y, y_pred, average = 'weighted')))
```

```
Accuracy :       0.9806678383128296
ROC :            0.9769303947994292
F-Measure :      0.9806197102070829
Precision :      0.980734680965647
Recall :         0.9806678383128296
Type I Error :   0.023076754397279742
Type II Error :  0.026807048713971104
```

Algorithms	Accuracy	AUROC	F-Measure
Without PCA Dataset			
Random Forest	0.963	0.9581	0.963
Logistic Regression	**0.9806**	**0.9769**	**0.9806**
XGBoost	0.9771	0.9741	0.9771
PCA Dataset			
Random Forest	0.9543	0.9511	0.9543
Logistic Regression	0.9789	0.9755	0.9788
XGBoost	**0.9806**	**0.9769**	**0.9806**

Table 4:Comparative Analysis of experimented Results

We used the tuned parameters to compare the performance of three classifiers on the raw data and PCA-processed dataset: Random Forest (RF), Logistic Regression (LR), and XGBoost (XGB). The outcomes demonstrated that PCA's impact on the classifiers' performance varied. In contrast to the raw data, RF and LR performed worse on the PCA-processed data, whereas XGBoost performed better. High accuracy and F-Measure scores were reached by both LR and XGBoost, with LR performing better on the raw data and XGBoost on the PCA-processed data. The performance of XGBoost remained comparable or even improved while going from 31 to 11, making it the most successful classifier. Figures 4 and 5 demonstrate the classifiers' performance. These outcomes highlight the value of PCA in managing high-dimensional data and highlight XGBoost's potential as a formidable classifier for this job.

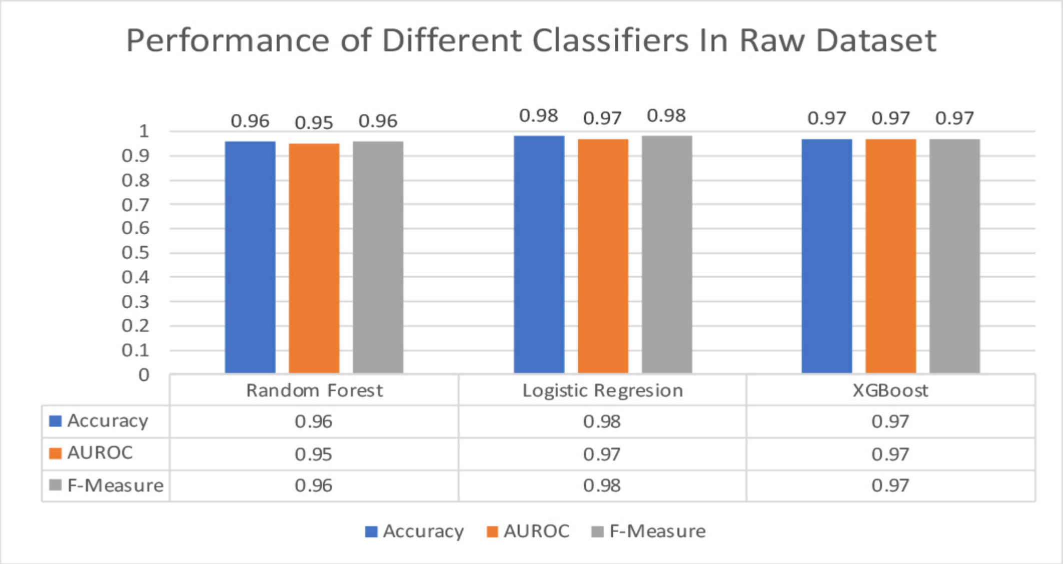


Figure 12: Performance of different classifier over Raw dataset

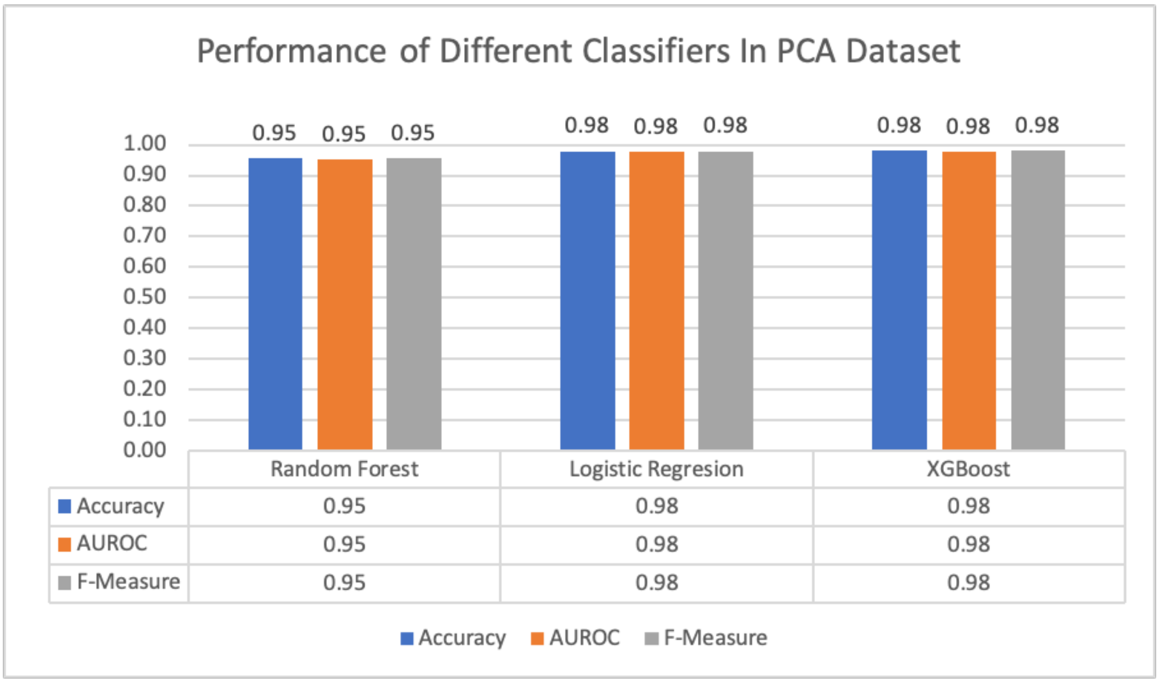


Figure 13: Performance of different classifier over PCA dataset

CHAPTER 5: CONCLUSION AND RECOMMENDATIONS:

5.1 CONCLUSION:

This research aimed to develop a reliable method for early detection of breast cancer using machine learning techniques. The proposed solution incorporated PCA and XGBoost, which demonstrated outstanding results with an accuracy rate of 98.06%. The results of this study outperformed previous studies and showed the potential for early detection of breast cancer, which can have a significant impact on patients' lives and healthcare systems.

Moreover, the comparison between various classifiers allowed us to select the best option for the model, ensuring its accuracy. The findings of this research demonstrated that the use of PCA can effectively reduce the number of features while preserving the essential information, making it an effective tool in managing high-dimensional data.

Although the proposed solution shows promising results, there is still room for improvement. In future studies, researchers can further explore cutting-edge machine learning and deep learning techniques to enhance the model's performance and make it more dependable. Additionally, the proposed solution can be integrated into a clinical setting to provide a non-invasive and reliable alternative to current testing methods.

Overall, this study contributes to the growing body of research on early detection of breast cancer using machine learning techniques. The proposed solution has the potential to revolutionize the way breast cancer is diagnosed and treated, leading to improved patient outcomes and reduced healthcare costs.

5.2 RECOMMENDATIONS FOR FUTURE WORK:

There are several recommendations for future work. First and foremost, future studies should focus on expanding the dataset to include a more diverse population and a larger sample size. This would allow for a more comprehensive analysis and may help identify additional risk factors for breast cancer. There are many limitations to this research. The fundamental challenges and a few related areas were the only things it first focused on. And Kaggle is the only tool used to gather the data.

Additionally, further investigation is required to identify the most effective feature selection and reduction techniques, as well as the most suitable classification algorithms for breast cancer diagnosis. Future studies may also explore the use of cutting-edge machine learning and deep learning techniques to enhance the accuracy of the model.

Furthermore, future studies should also consider incorporating other medical imaging techniques, such as magnetic resonance imaging (MRI), in addition to mammography, to increase the sensitivity and specificity of the model. Finally, the proposed solution may be extended to other types of cancer, providing an even more comprehensive diagnostic tool.

In conclusion, this study has demonstrated the potential of utilizing machine learning algorithms and feature selection techniques to accurately diagnose breast cancer. However, further research is needed to fully realize the potential of these techniques and to develop more robust and accurate models for breast cancer diagnosis.

REFERENCES:

[1] Chaurasia, V., Pal, S., & Tiwari, B. B. (2018). Prediction of benign and malignant breast cancer using data mining techniques. Journal of Algorithms & Computational Technology, 12(2), 119-126.

[2] Cakir, A. & Demirel, B. (2011). A software tool for determination of breast cancer treatment methods using data mining approach. Journal of Medical Systems, 35(6), 1503-1511.

[3] Cakir, A. & Demirel, B. (2011). A software tool for determination of breast cancer treatment methods using data mining approach. Journal of Medical Systems, 35(6), 1503-1511.

[4] Cakir, A. & Demirel, B. (2011). A software tool for determination of breast cancer treatment methods using data mining approach. Journal of Medical Systems, 35(6), 1503-1511.

[5] P. Kulkarni, Fine grained classification of mammographic lesions using pixel NGRAMS, AJCT (2019).

[6] A.A. Kayode, N.O. Akande, A.A. Adegun, M.O. Adebiyi, An automated mammogram classification system using modified support vector machine, Med Devices (Auckl) 12 (2019) 275–284,

[7] Y. Jin, Medical Image Processing with Deep Learning : Mammogram Classification and Automatic Lesion Detection, pp. 1–19, 2019.

[8] Y. Jin, Medical Image Processing with Deep Learning : Mammogram Classification and Automatic Lesion Detection, pp. 1–19, 2019.

[9] P. Kaur, G. Singh, P. Kaur, Intellectual detection and validation of automated mammogram breast cancer images by multi-class SVM using deep learning classification, Inf. Med. Unlocked 100239 (2019)

[10] A.O. Ibrahim, Classification of mammogram images using radial basis function neural network, in: F. Saeed, F. Mohammed, N. Gazem (Eds.), Emerging Trends in Intelligent Computing and Informatics. IRICT 2019. Advances in Intelligent Systems and Computing, Springer, Cham, 2020.

[11] M. F. Ak, "A comparative analysis of breast cancer detection and diagnosis using data visualization and machine learning applications," Healthcare, vol. 8, no. 2, p. 111, 2020

[12] A. F. M. Agarap, "On breast cancer detection: an application of machine learning algorithms on the Wisconsin diagnostic dataset," in Proceedings of the 2nd International Conference on Machine Learning and Soft Computing, pp. 5–9, Phu Quoc Island, Vietnam, February 2018.

[13] H. Benbrahim, H. Hachimi, and A. Amine, "Comparative study of machine learning algorithms using the breast cancer dataset," in Proceedings of the International Conference on Advanced Intelligent Systems for Sustainable Develop

[14] H. Asri, H. Mousannif, H. A. Moatassime, and T. Noel, "Using machine learning algorithms for breast cancer risk prediction and diagnosis," Procedia Computer Science.

[15] Qu, Z. Predicting diabetes mellitus with machine learning techniques. Front. Genet. 2011, 9, 515.

[16] Wang, H.; Yoon, W.S. Breast cancer prediction using data mining method. In Proceedings of the 2015 Industrial and Systems Engineering Research Conference, Nashville, TN, USA, 30 May–2 June 2015.