Politecnico di Torino

Breast Cancer Classification

Prashant Kumar Ray

s271098, Computer Engineering

1 Introduction

Breast cancer is a dominant cancer in women worldwide and is increasing in developing countries where most cases are diagnosed in late stages. An early diagnosis can greatly improve the prognosis and chance of survival for patients. Breast Cancer occurs as the result of abnormal growth of the cells in the breast tissue commonly referred to as tumors. A tumor does not mean cancer-tumors. It can be benign (non-cancerous), pre-malignant (pre -cancerous) or malignant(cancerous). In this work I plan to build a model which accurately classifies tumours as Benign or Malignant based on certain features. The code is available at https://github.com/pkr076/DataSpaceExamProject.git.

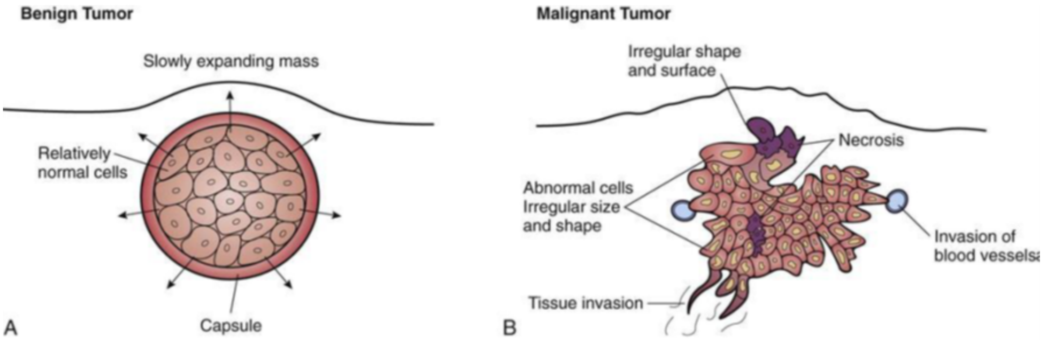


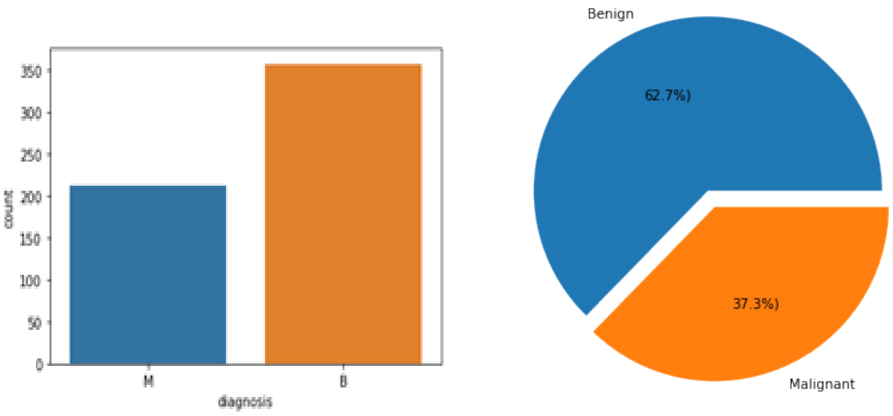
Figure 1: Breast Tumors

2 Exploratory Data Analysis

This is an analysis of the Breast Cancer Wisconsin (Diagnostic) Dataset, available at both UCI and Kaggle websites. This data set was created by Dr. William H. Wolberg. It contains 596 rows and 32 columns of tumour shape and specifications. The tumor is ultimately classified as benign or malignant based on its geometry and shape. Features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image. There are 212 malignant and 357 benign tumours in the dataset.

The features of the dataset include:

1. tumour radius (mean of distances from center to points on the perimeter)



(a) Count plot

(b) Class-wise Proportion

Figure 2: Class-wise distribution For Breast Cancer Wisconsin (Diagnostic) Dataset

2. texture (standard deviation of gray-scale values)
3. perimeter
4. area
5. smoothness (local variation in radius lengths)
6. compactness (perimeter$^{\mathbf{2}}$ / area — 1.0)
7. concavity (severity of concave portions of the contour)
8. concave points (number of concave portions of the contour)
9. symmetry
10. fractal dimension

The mean, standard error and ”worst” or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 numerical features. The only categorical feature in the dataset is the target feature ‘Diagnosis’. There are no missing values in the dataset. I have mapped this categorical feature into a numerical feature as summarized in Table 1.

Target Class	Code
Malignant	1
Benign	0

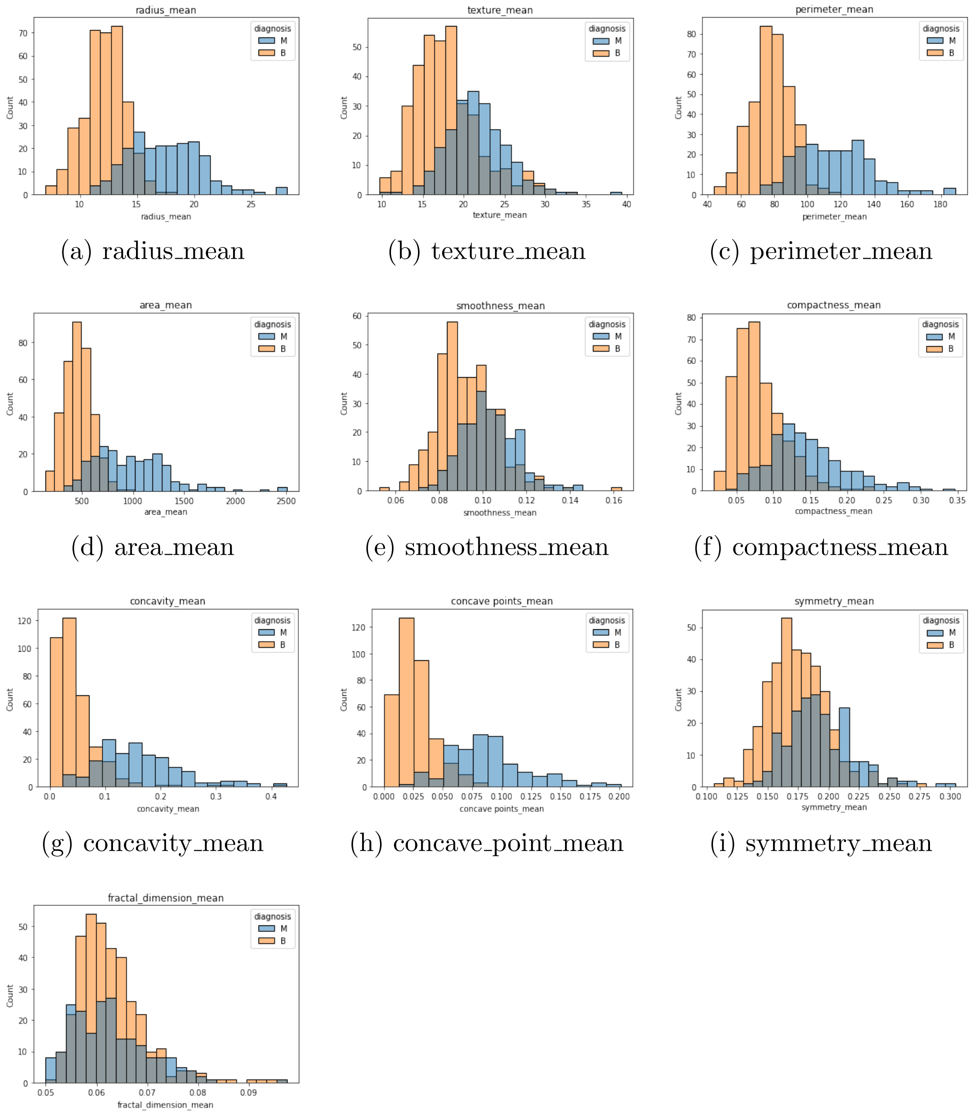
Table 1: Label Encoding for target feature

From the Fig[3,4,5] we can observe that most of the features are normally distributed. Also, we can observe that there is some separation in data distribution for the two classes for many features which might be good for classification. Although some features, for example: smoothness_se, texture_se etc, have almost the same data distribution for the two classes.

The separation of the two classes for a given feature can also be inferred by box plot. If the median line of one box lies outside of second box entirely, then there is likely to be a difference between the two groups. For example, In Fig[6], we can see that for the most of the feature the medians of malignant and benign group are well separated. These features can be good features for the classification.

Box plot is also useful for visualizing outliers present in the data. We can observe in Fig.[6,7,8], many outliers here but, in the paper the authors mention that "the features are numerically modelled such that larger values will typically indicate a higher likelihood of malignancy". That is why I am not going to remove the outliers because data has been already processed and confirmed by the experts of this sector.

Correlation is a statistical technique which determines how one variables is related with the other variable. It gives us the idea about the degree of the relationship of the two variables. It's a bi-variate analysis measure which describes the association between different variables. Two features can be positively correlated or negatively correlated or can have no correlation at all. In Fig.9 shows the correlation matrix for the features of the dataset. We can say that the set of features having correlation greater than 0.8 are highly correlated.



(a) radius_mean

(b) texture_mean

(c) perimeter_mean

(d) area_mean

(e) smoothness_mean

(f) compactness_mean

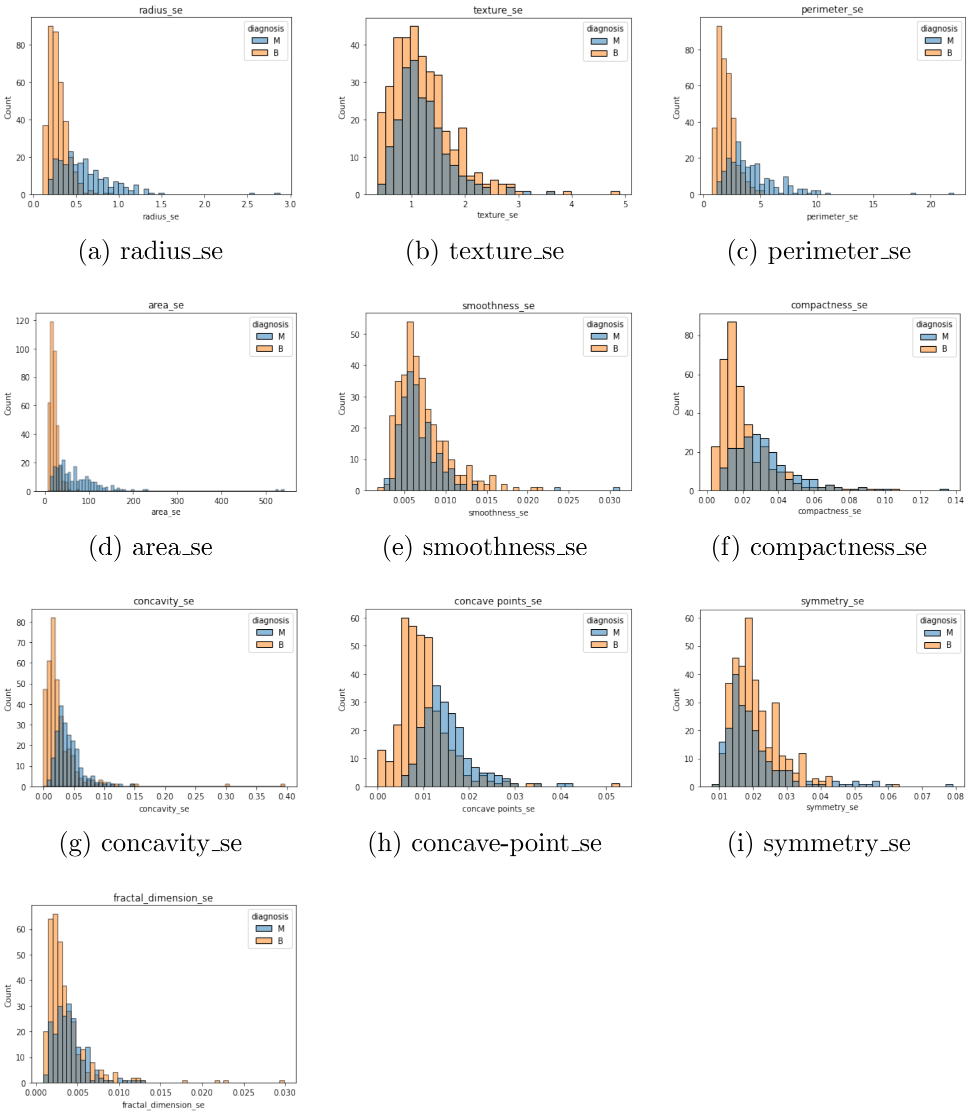
(g) concavity_mean

(h) concave_point_mean

(i) symmetry_mean

(j) fractal_dimension_mean

Figure 3: Histogram for 'mean' group features



(a) radius_se

(b) texture_se

(c) perimeter_se

(d) area_se

(e) smoothness_se

(f) compactness_se

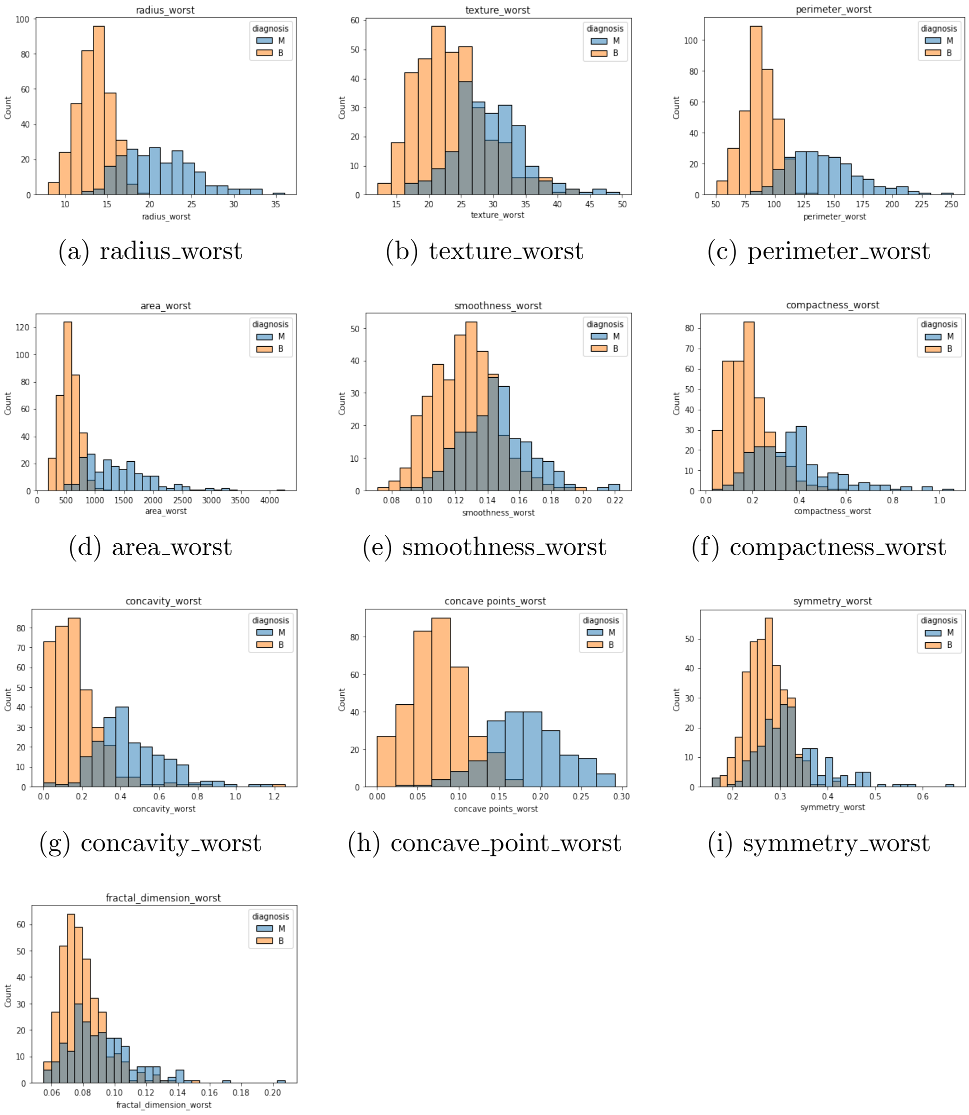
(g) concavity_se

(h) concave-point_se

(i) symmetry_se

(j) fractal_dimension_se

Figure 4: Histogram for 'se' group features



(a) radius_worst

(b) texture_worst

(c) perimeter_worst

(d) area_worst

(e) smoothness_worst

(f) compactness_worst

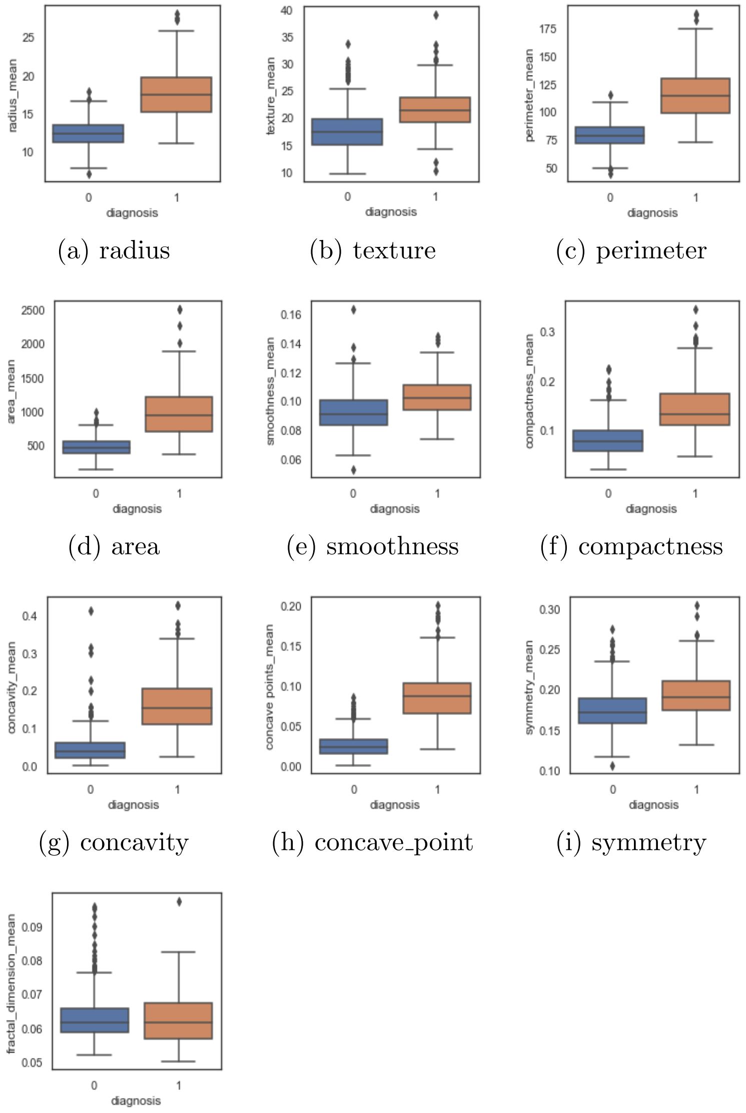
(g) concavity_worst

(h) concave_point_worst

(i) symmetry_worst

(j) fractal_dimension_worst

Figure 5: Histogram for 'Worst' group features



(a) radius

(b) texture

(c) perimeter

(d) area

(e) smoothness

(f) compactness

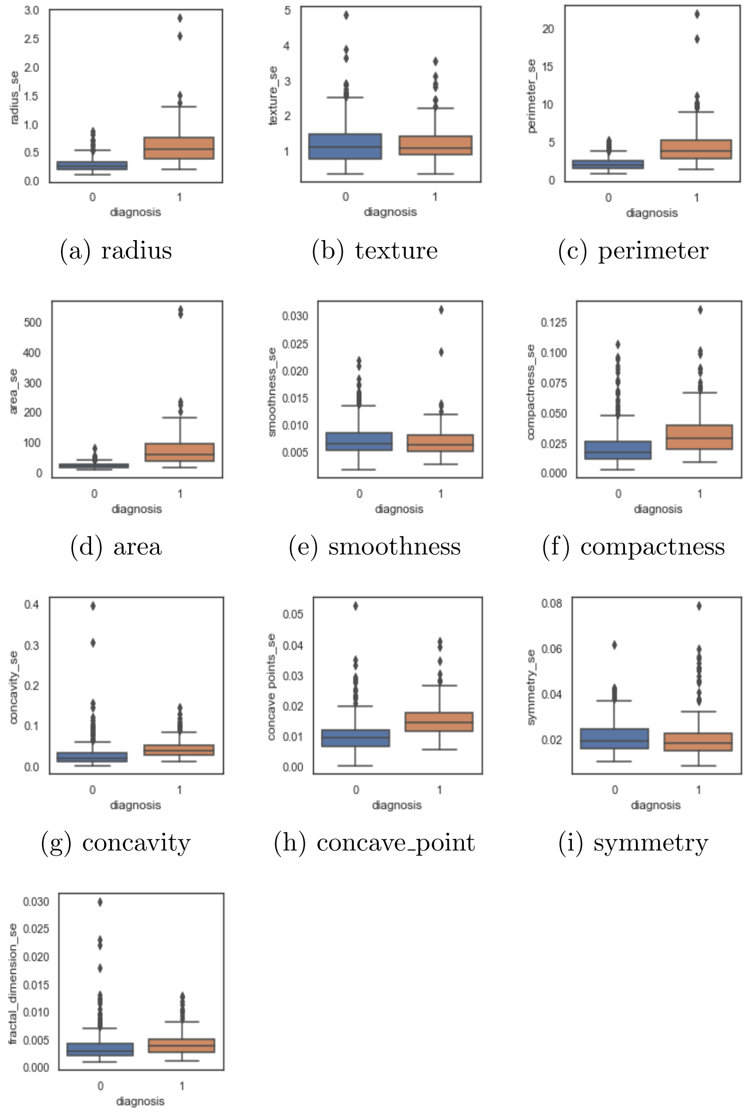
(g) concavity

(h) concave_point

(i) symmetry

(j) fractal_dimension

Figure 6: Box Plot for 'mean' group of features



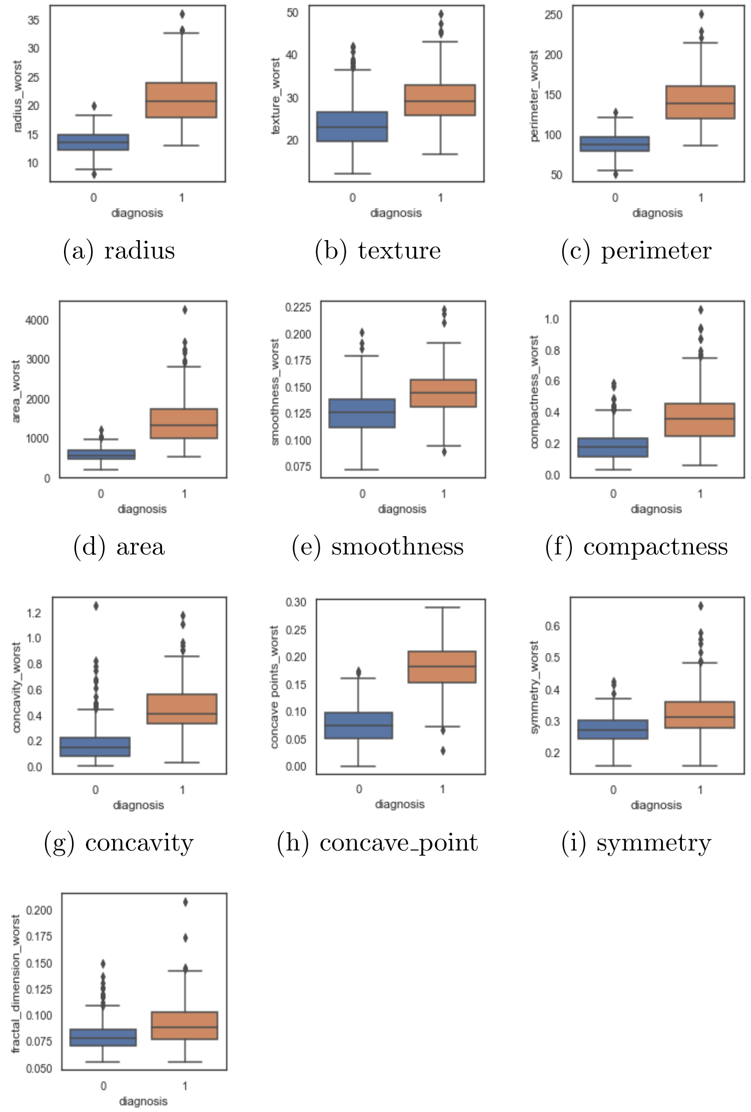
(a) radius (b) texture (c) perimeter

(d) area (e) smoothness (f) compactness

(g) concavity (h) concave_point (i) symmetry

(j) fractal_dimension

Figure 7: Box Plot for 'se' group of features



(a) radius (b) texture (c) perimeter

(d) area (e) smoothness (f) compactness

(g) concavity (h) concave_point (i) symmetry

(j) fractal_dimension

Figure 8: Box Plot for 'Worst' group of features

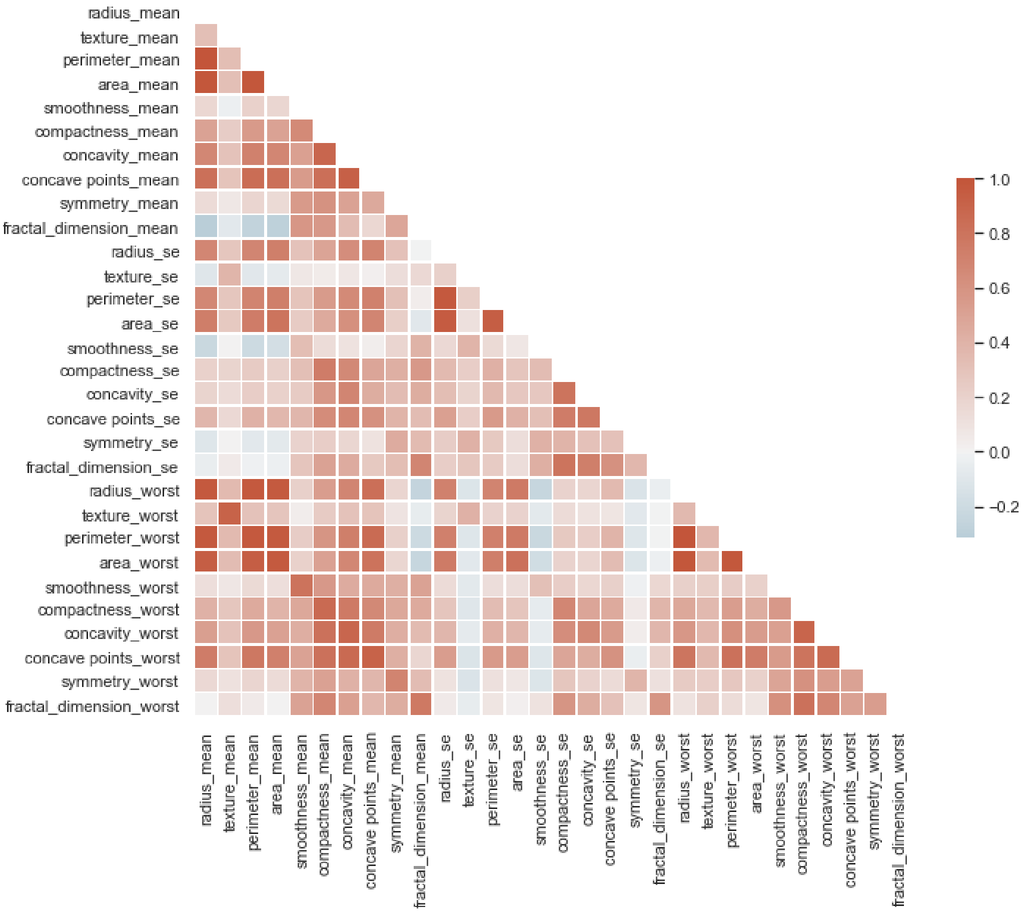


Figure 9: Correlation Matrix of features

3 Decision Tree

The decision tree algorithm is a powerful algorithm that can be used for classification or regression problems. In our case we will use this algorithm for classification. Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree. The algorithm selects one or more features and split the input data considering the values taken by these examples in the selected features. This is done until the tree reaches a desired max_depth, the splits contains only examples of a single class or the split contains a minimum number of samples min_samples_split.
In Decision Tree the major challenge is to identification of the attribute for the root node in each level. This process is known as attribute selection. We

have two popular attribute selection measures:

1. Information Gain
2. Gini Index

3.1 Information Gain

Information Gain = Entropy of the full dataset - Entropy of the dataset given some feature

Definition: Suppose S is the set of instances, A is an attribute,S_v is the subset of S with A=v, and values(A) is the set of all possible values of A, then

$Gain(S, A) = Entropy(S) - \sum_{v\epsilon Values(A)} \frac{|S_v|}{|S|} \cdot Entropy(S_v)$

Entropy represents the uncertainty in our dataset or measure of disorder. It characterizes the impurity of an arbitrary collection of examples. The higher the entropy more the information content.

$Entropy = -\sum_{i=0}^{n} p_i \cdot log(p_i)$

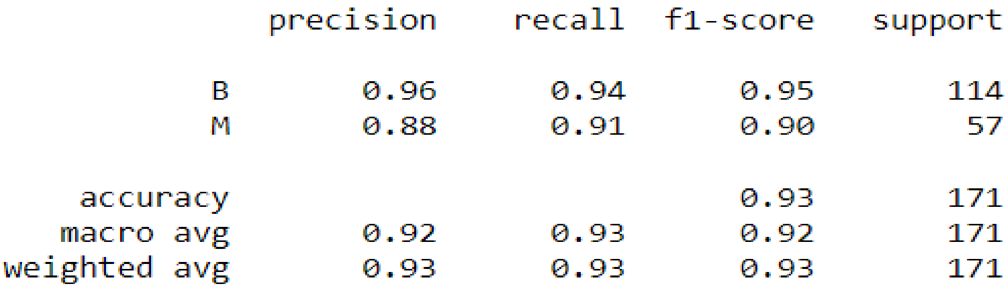
3.2 Gini Index

Definition: Gini Impurity is a measurement of the likelihood of an incorrect classification of a new instance of a random variable, if that new instance were randomly classified according to the distribution of class labels from the data set. In simple terms, Gini Index is a metric to measure how often a randomly chosen element would be incorrectly identified. It means an attribute with lower Gini index should be preferred.

$G = \sum_{k=1}^{K} pmk(1 - pmk)$

where K is the number of classes and pmk is the proportion of examples of the region m belong to the k class.

After training and fine-tuning the decision tree model, the achieved accuracy for the model on the test data is around 93%.

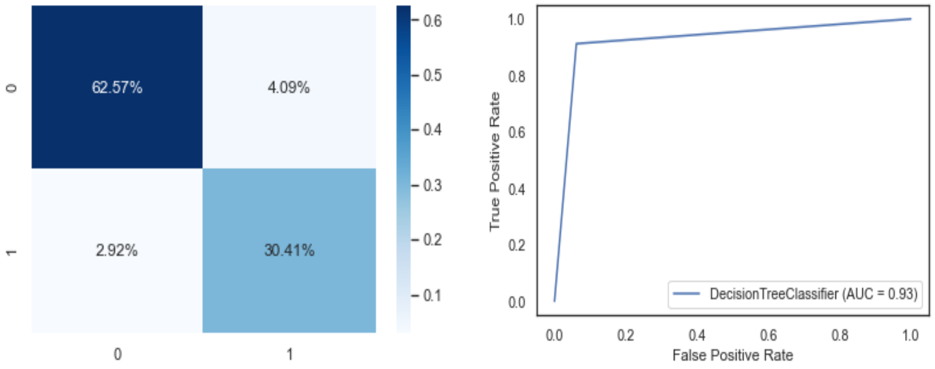


```
              precision    recall  f1-score   support

           B       0.96      0.94      0.95       114
           M       0.88      0.91      0.90        57

    accuracy                           0.93       171
   macro avg       0.92      0.93      0.92       171
weighted avg       0.93      0.93      0.93       171
```

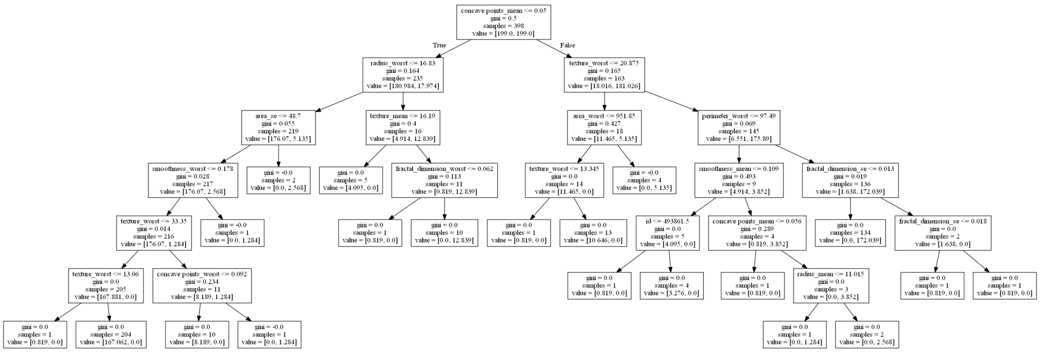
Figure 10: Decision Tree model classification report



(a) Confusion Matrix

(b) ROC

Figure 11: Decision Tree Confusion Matrix and ROC curve

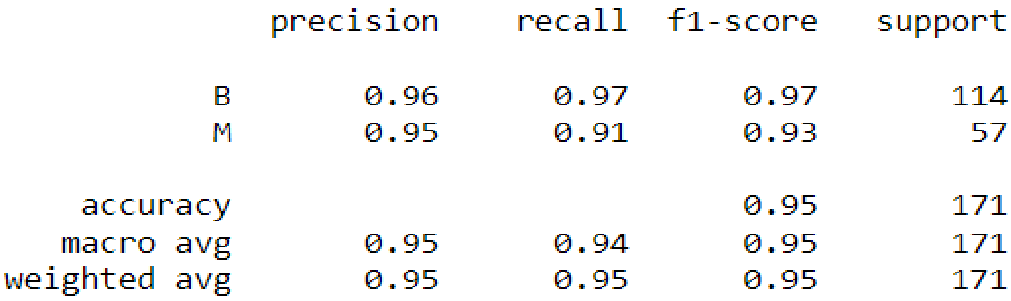


concave points_mean <= 0.05 gini = 0.5 samples = 398 value = [199.0, 199.0]
True
False
radius_worst <= 16.83 gini = 0.164 samples = 235 value = [180.984, 17.974]
texture_worst <= 20.875 gini = 0.165 samples = 163 value = [18.016, 181.026]
area_se <= 48.7 gini = 0.055 samples = 219 value = [176.07, 5.135]
texture_mean <= 16.19 gini = 0.4 samples = 16 value = [4.914, 12.839]
area_worst <= 951.85 gini = 0.427 samples = 18 value = [11.465, 5.135]
perimeter_worst <= 97.49 gini = 0.069 samples = 145 value = [6.551, 175.89]
smoothness_worst <= 0.178 gini = 0.028 samples = 217 value = [176.07, 2.568]
gini = -0.0 samples = 2 value = [0.0, 2.568]
gini = 0.0 samples = 5 value = [4.095, 0.0]
fractal_dimension_worst <= 0.062 gini = 0.113 samples = 11 value = [0.819, 12.839]
texture_worst <= 13.345 gini = 0.0 samples = 14 value = [11.465, 0.0]
gini = -0.0 samples = 4 value = [0.0, 5.135]
smoothness_mean <= 0.109 gini = 0.493 samples = 9 value = [4.914, 3.852]
fractal_dimension_se <= 0.013 gini = 0.019 samples = 136 value = [1.638, 172.039]
texture_worst <= 33.35 gini = 0.014 samples = 216 value = [176.07, 1.284]
gini = -0.0 samples = 1 value = [0.0, 1.284]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 10 value = [0.0, 12.839]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 13 value = [10.646, 0.0]
id <= 493861.5 gini = 0.0 samples = 5 value = [4.095, 0.0]
concave points_mean <= 0.056 gini = 0.289 samples = 4 value = [0.819, 3.852]
gini = 0.0 samples = 134 value = [0.0, 172.039]
fractal_dimension_se <= 0.018 gini = 0.0 samples = 2 value = [1.638, 0.0]
texture_worst <= 13.06 gini = 0.0 samples = 205 value = [167.881, 0.0]
concave points_worst <= 0.092 gini = 0.234 samples = 11 value = [8.189, 1.284]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 4 value = [3.276, 0.0]
gini = 0.0 samples = 1 value = [0.819, 0.0]
radius_mean <= 11.015 gini = 0.0 samples = 3 value = [0.0, 3.852]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 1 value = [0.819, 0.0]
gini = 0.0 samples = 204 value = [167.062, 0.0]
gini = 0.0 samples = 10 value = [8.189, 0.0]
gini = -0.0 samples = 1 value = [0.0, 1.284]
gini = 0.0 samples = 1 value = [0.0, 1.284]
gini = 0.0 samples = 2 value = [0.0, 2.568]

Figure 12: Decision Tree

4 Random Forest

Random forest is a supervised learning algorithm. It can be used for both classification and regression. The “forest” it builds is an ensemble of decision trees, usually trained with the “bagging” method. The general idea of the bagging method is that a combination of learning models increases the overall result. Random forest builds multiple decision trees and merges them together to get more accurate and stable predictions.
Instead of searching for the most important feature while splitting a node , it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. Unlike a normal decision tree, It is also possible to add some random threshold for each feature rather than searching for the best possible thresholds.
After training and fine-tuning the Random Forest model, the achieved accuracy for the model on the test data is about 95%.

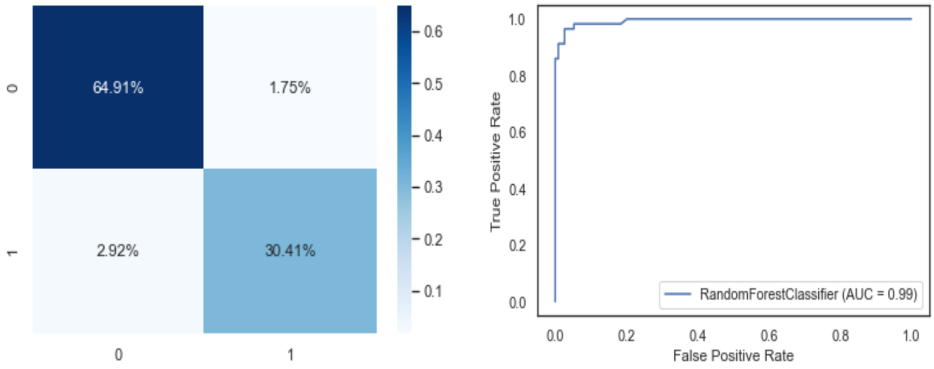


```
              precision    recall  f1-score   support

           B       0.96      0.97      0.97       114
           M       0.95      0.91      0.93        57

    accuracy                           0.95       171
   macro avg       0.95      0.94      0.95       171
weighted avg       0.95      0.95      0.95       171
```

Figure 13: Random Forest model classification report



(a) Confusion Matrix

(b) ROC

Figure 14: Random Forest Confusion Matrix and ROC curve

5 PCA

Dimensionality reduction techniques address the "Curse of Dimensionality" by extracting new features from the data, rather than removing low-information features. The new features are usually a weighted combination of existing features. Principal component analysis (PCA) is a technique for reducing the dimensionality of such datasets but at the same time minimizing information loss. It does so by creating new uncorrelated variables that successively maximize variance.

There are two benefits for applying PCA , firstly it reduces the curse of dimensionality and it makes the data uncorrelated and a lot of algorithms work better if the data is uncorrelated. Reduced dimension also reduces the storage requirement.

Before applying PCA algorithm, it is important to scale the data.Feature scaling is essential for machine learning algorithms that calculate distances between data. If not scale, the feature with a higher value range starts dominating when calculating distances.

I have used the PCA algorithm such that the new components would keep at least 90% of original variance. The algorithm outputs 7 new principal components which is shown in the Fig.15.

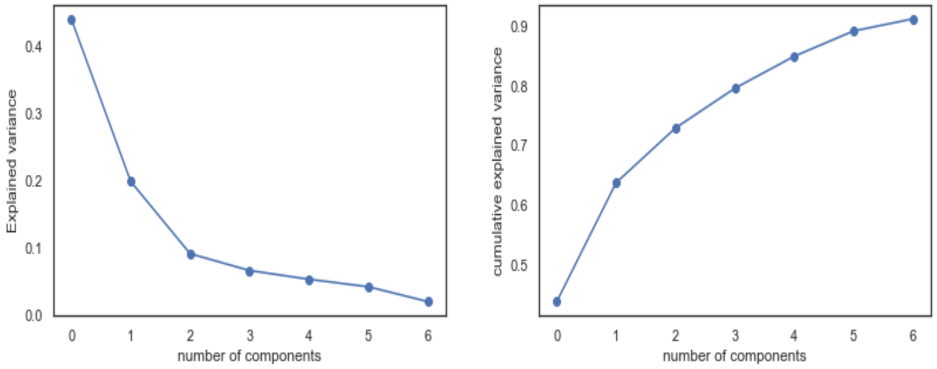


Figure 15: PCA: Explained Variance and Cumulative Explained Variance

6 KNN

K - Nearest Neighbors is a supervised machine learning algorithm that assumes that the behaviour of a data point is similar to the behaviour of the nearest neighbours. K in KNN is a parameter that refers to the number of the nearest neighbors to include in the majority voting process.

For a given value of K, algorithm will find the K nearest neighbors of unseen data point and then it will assign the class to unseen data point by having the class which has the highest number of data points out of all classes of K neighbors. The selection of optimal K value is essential as the small value of K leads to unstable decision boundary. The substantial K value is better for smoother decision boundary.

For selecting optimal value of K, I have derived a plot between error-rate and K. Then I have chosen the K=5 which has low error rate. The achieved accuracy for the KNN model for K=5 on the test data is about 95%.

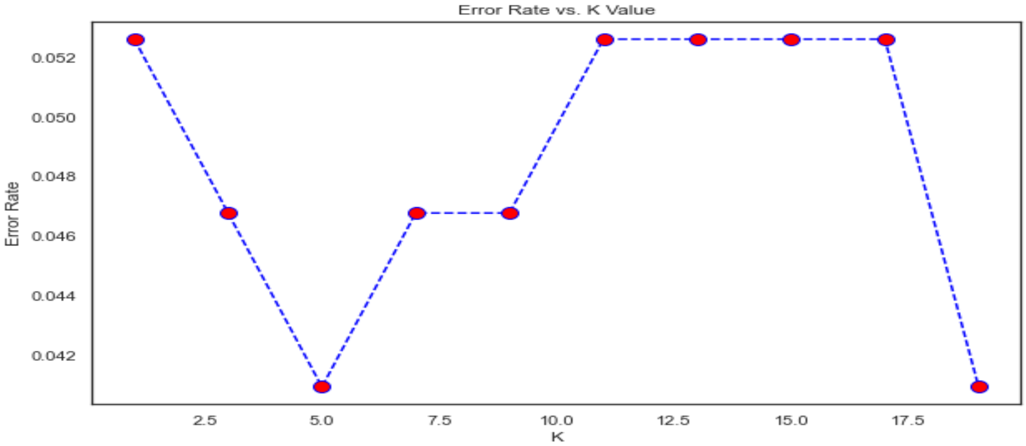
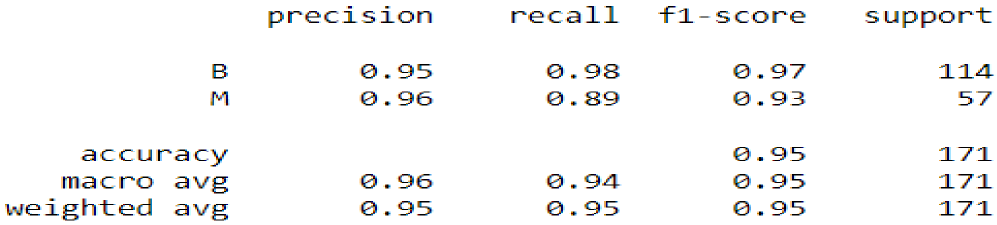


Figure 16: KNN: Error rate vs K

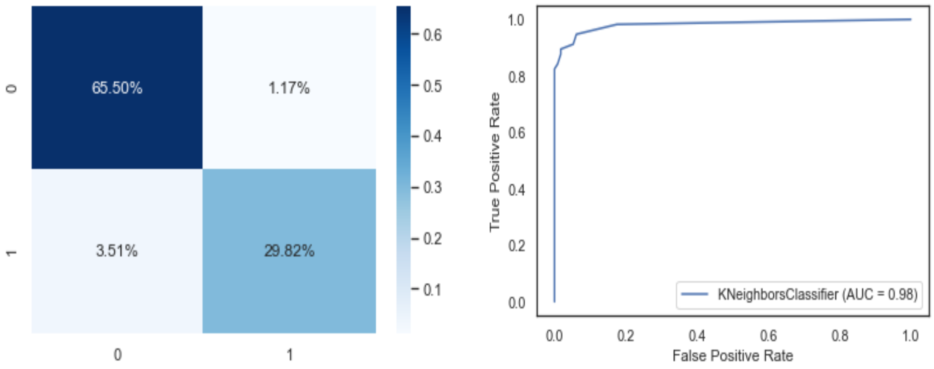


```
              precision    recall  f1-score   support

           B       0.95      0.98      0.97       114
           M       0.96      0.89      0.93        57

    accuracy                           0.95       171
   macro avg       0.96      0.94      0.95       171
weighted avg       0.95      0.95      0.95       171
```

Figure 17: KNN model classification report



(a) Confusion Matrix

(b) ROC

Figure 18: KNN: Confusion Matrix and ROC curve

7 Logistic Regression

Logistic regression is a classification algorithm used to assign observations to a discrete set of classes. Unlike linear regression which outputs continuous number values, logistic regression transforms its output using the logistic sigmoid function to return a probability value which can then be mapped to two or more discrete classes.
Given a data(X,Y), X being a matrix of values with m examples and n features and Y being a vector with m examples. The objective is to train the model to predict which class the future values belong to. Primarily, we create a weight matrix with random initialization. Then we multiply it by features.
$a = w_0 + w_1x_1 + w_2x_2 + ... + w_nx_n$
As the values that we obtain ranges from negative infinity to positive infinity based on the linear model, we need to narrow it down to a score that is in between zero and one as probabilities always are in that range and logistic regression talks about probabilities. The sigmoid function takes care of this work.
The exponent present in the sigmoid function ensures that the value always remains positive as probability is always greater than zero and the property of exponents takes care of this aspect. Then we need to worry about the limiting the values less than one, which is done by dividing the value in the numerator by value greater than it.
$P(X) = \frac{1}{1+e^{-a}}$
So,we pass the output obtained from above equation to a sigmoid function.

$$P(X) = \frac{1}{1+e^{w_0+w_1x_1+w_2x_2+......+w_nx_n}}$$

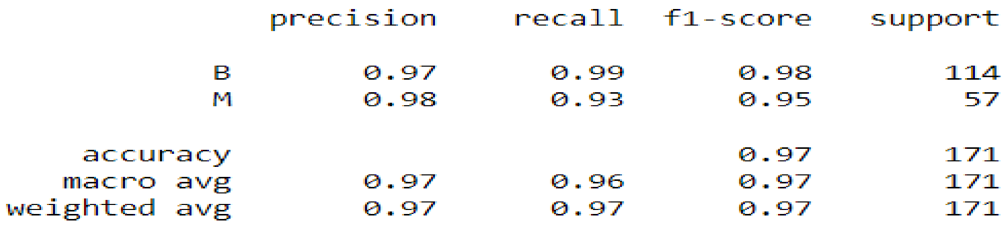
For the probability of the alternate class we just have to subtract the value obtained above by 1. After doing some calculation, we can find the following equation:

$$log(\frac{P(X)}{(1-P(X))}) = w_0 + w_1 x_1 + w_2 x_2 + ... + w_n x_n$$

The coefficients $(w_0, w_1, ..., w_n)$ are estimated based on the available training data. Maximum Likelihood is the preferred approach for this. A mathematical formalization is in the form of a likelihood function:

$$l(w_0, w_1, ..., w_n) = \prod_{i:y_i=1} p(x_i) \prod_{j:y_j=0} 1 - p(x_j)$$

We choose coefficients $(w_0, w_1, ..., w_n)$ that maximizes the above equation. The achieved accuracy for the Logistic regression model on the test data is about 97%.

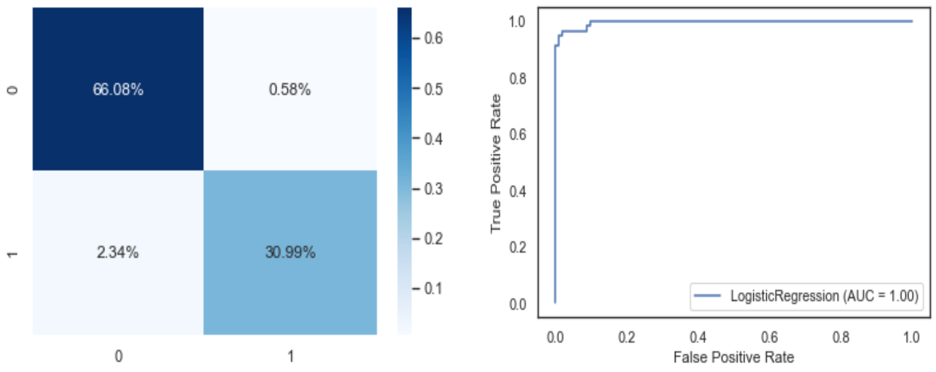


```
              precision    recall  f1-score   support

           B       0.97      0.99      0.98       114
           M       0.98      0.93      0.95        57

    accuracy                           0.97       171
   macro avg       0.97      0.96      0.97       171
weighted avg       0.97      0.97      0.97       171
```

Figure 19: Logistic Regression model classification report



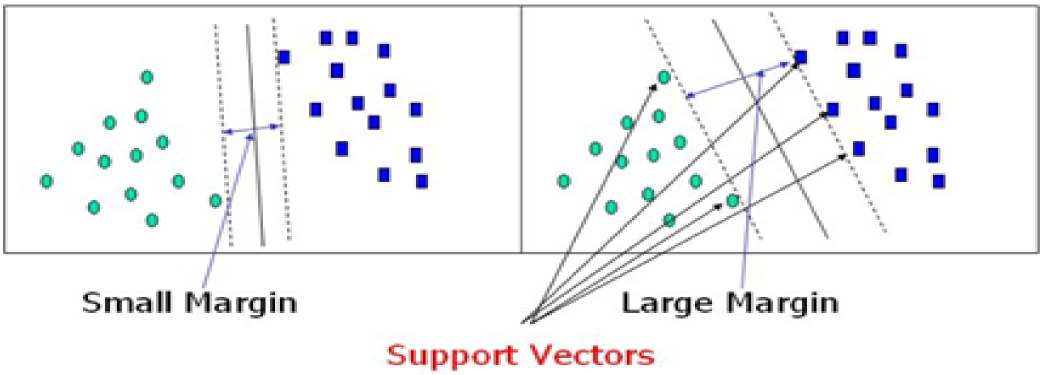
(a) Confusion Matrix

(b) ROC

Figure 20: Logistic Regression: Confusion Matrix and ROC curve

8 Support Vector Machine (SVM)

Support Vector Machine(SVM) is a supervised machine learning algorithm used for both classification and regression. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points. The dimension of the hyperplane depends upon the number of features. To separate the two classes of data points, there are many possible hyperplanes that could be chosen. The objective is to find a plane that has the maximum margin, i.e the maximum distance between data points of both classes.

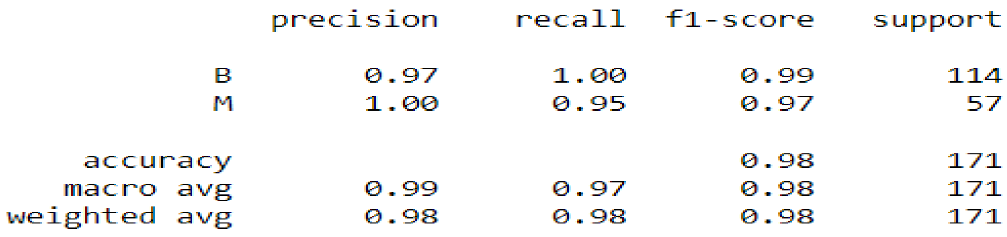


Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier.
If data is not linearly separable, the hard margin SVM (i.e., no misclassification allowed) is not feasible. Soft margin SVM overcomes this problem by introducing slack variables i:

$$\min_{w,b} \frac{1}{2}||w||^2 + C\sum_i \xi_i, \text{ subject to } y_i[< w, x_i > +b] \geq 1 - \xi_i, \xi_i \geq 0$$

We will tune an hyper-parameter 'C' which represents how much we want to avoid misclassifying each training sample. Small values of C cause the optimizer to look for larger-margin separating hyperplanes, even though this comes at the cost of misclassifying more points. Conversely, large values of C correspond to smaller-margin hyperplanes if they do a better job at classifying all the points correctly. This strongly affects the decision boundaries of the model.
The SVM kernel is a function that takes low dimensional input space and transforms it into higher-dimensional space, ie it converts not separable prob-

lem to separable problem. It is mostly useful in non-linear separation problems.
The achieved accuracy for the SVM model on the test data is about 98%.

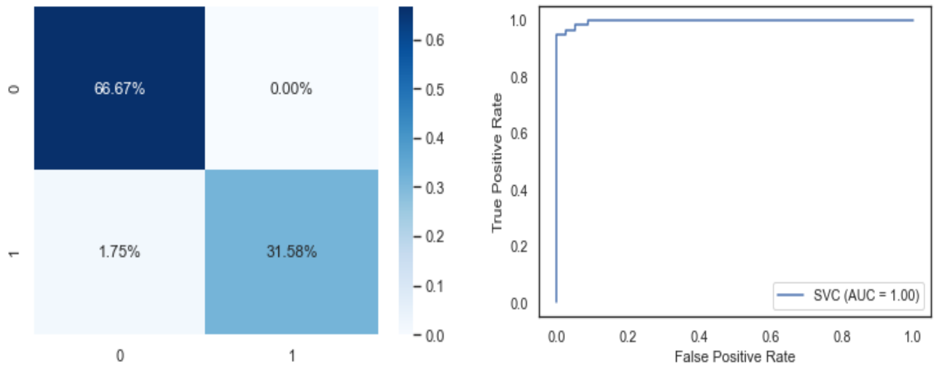


```
              precision    recall  f1-score   support

           B       0.97      1.00      0.99       114
           M       1.00      0.95      0.97        57

    accuracy                           0.98       171
   macro avg       0.99      0.97      0.98       171
weighted avg       0.98      0.98      0.98       171
```

Figure 21: SVM model classification report



(a) Confusion Matrix

(b) ROC

Figure 22: SVM: Confusion Matrix and ROC curve

9 Conclusion

I have used a variety of algorithms for the correct classification of tumors as malignant or benign. Among all the algorithms used, the Logistic Regression and Support Vector Classifier gave maximum accuracies and minimum misclassification for the positive class. In this particular classification problem we can not afford to classify a malignant tumor as benign, hence the False negative case has more importance. Therefore Our goal should be to maximize the recall values so as to avoid misclassification of FN type.

10 Tools:

I have used Jupyter Notebook to write the code. The libraries that are used is listed below:

1. Numpy
2. Pandas
3. Matplotlib
4. Seaborn
5. Scikit learn