Assignment-8

Analytic tools, Techniques and Methods

Amruth Jaligama

Student Id:11396245

**Topic Chosen**

The topic that I have chosen is regarding breast cancer prediction that is Efficacy of machine learning algorithms in predicting whether cancer is benign or malignant using machine learning algorithm

Breast Cancer Wisconsin (diagnostic) dataset was used to comment on the effectiveness of machine learning to classify breast cancer.

Source: https://www.kaggle.com/uciml/breast-cancer-wisconsin-data

**Information Regarding Dataset**

The dataset comprises 33 columns and 569 rows. The 'diagnosis' column is the target label for the dataset and features are the characteristics of the cell nuclei computed from a digitized image of fine needle aspirate (FNA) of a breast mass.

**Columns In The Dataset:**

diagnosis - The diagnosis of breast tissues (M = malignant, B = benign)

radius_mean - mean of distances from centre to points on the perimeter

texture_mean - standard deviation of gray-scale values

perimeter_mean - mean size of the core tumor

area_mean - mean area of the core tumor

smoothness_mean - mean of local variation in radius lengths

compactness_mean - mean of perimeter^2 / area - 1.0

concavity_mean - mean of severity of concave portions of the contour

concave points_mean - mean for number of concave portions of the contour

symmetry_mean

fractal_dimension_mean -mean for "coastline approximation" - 1

radius_se - standard error for the mean of distances from centre to points on the perimeter

texture_se - standard error for standard deviation of gray-scale values

perimeter_se -

area_se -

smoothness_se - standard error for local variation in radius lengths

compactness_se - standard error for perimeter^2 / area - 1.0

concavity_se - standard error for severity of concave portions of the contour

concave points_se - standard error for number of concave portions of the contour

symmetry_se

fractal_dimension_se - standard error for "coastline approximation" - 1

radius_worst - "worst" or largest mean value for mean of distances from centre to points on the perimeter

texture_worst - "worst" or largest mean value for standard deviation of gray-scale values

perimeter_worst

Area_worst -

smoothness_worst - "worst" or largest mean value for local variation in radius lengths

compactness_worst - "worst" or largest mean value for perimeter^2 / area - 1.0

concavity_worst - "worst" or largest mean value for severity of concave portions of the contour

concave points_worst - "worst" or largest mean value for number of concave portions of the contour

symmetry_worst

fractal_dimension_worst - "worst" or largest mean value for "coastline approximation" - 1

## Exploratory Data Analysis

Exploratory data analysis was carried out primarily using matplotlib and seaborn libraries. The shape of the data didn't match the dataset description. Further, when I plotted the count of empty rows in each column I found the extra empty column in the dataset file owing to maybe unnoticed spaces in the original file.

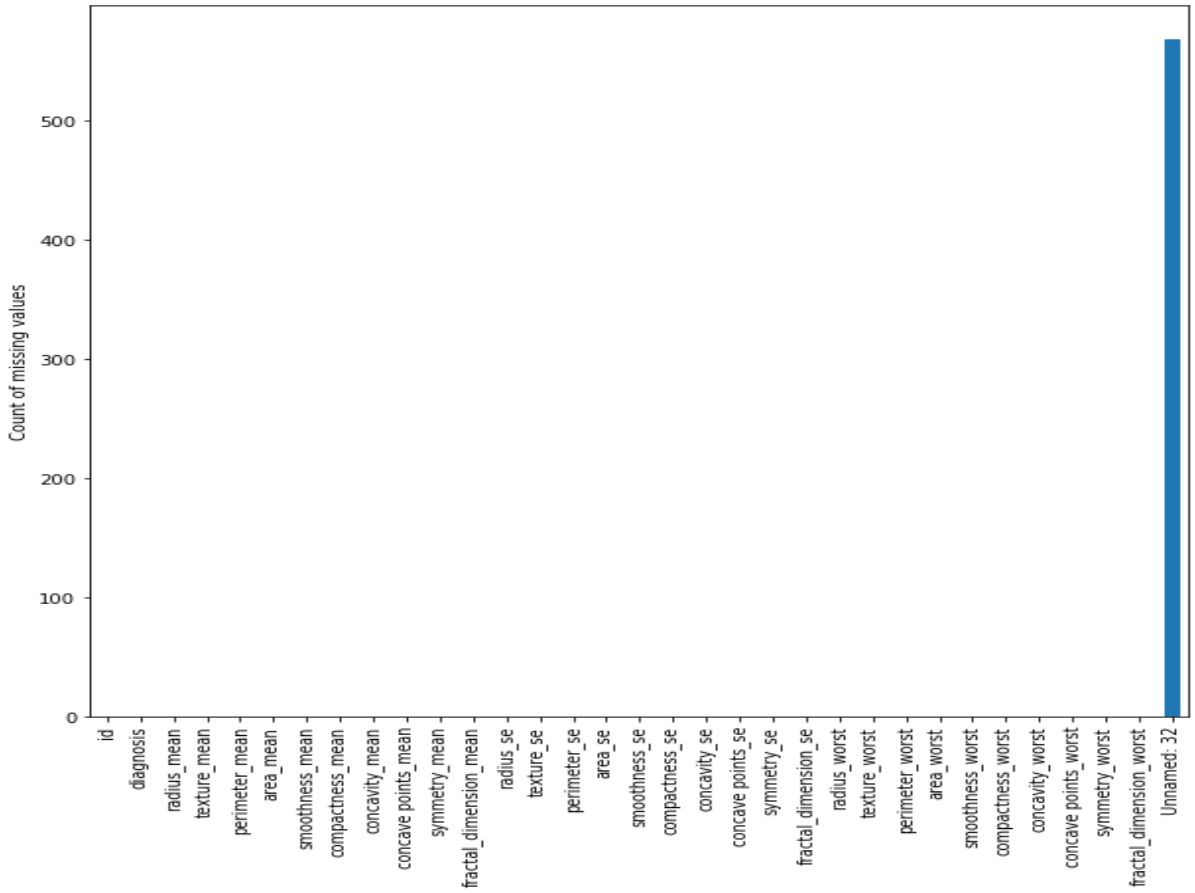


Fig 1: Count of empty rows in each column

This column along with the 'id' column was dropped from the dataset. Further, I looked at the summary statistic for the remaining columns

| | diagnosis | radius_mean | texture_mean | perimeter_mean | area_mean | smoothness_mean | compactness_mean | concavity_mean | concave points_mean |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 569 | 569.000000 | 569.000000 | 569.000000 | 569.000000 | 569.000000 | 569.000000 | 569.000000 | 569.000000 |
| **unique** | 2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **top** | B | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **freq** | 357 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **mean** | NaN | 14.127292 | 19.289649 | 91.969033 | 654.889104 | 0.096360 | 0.104341 | 0.088799 | 0.048919 |
| **std** | NaN | 3.524049 | 4.301036 | 24.298981 | 351.914129 | 0.014064 | 0.052813 | 0.079720 | 0.038803 |
| **min** | NaN | 6.981000 | 9.710000 | 43.790000 | 143.500000 | 0.052630 | 0.019380 | 0.000000 | 0.000000 |
| **25%** | NaN | 11.700000 | 16.170000 | 75.170000 | 420.300000 | 0.086370 | 0.064920 | 0.029560 | 0.020310 |
| **50%** | NaN | 13.370000 | 18.840000 | 86.240000 | 551.100000 | 0.095870 | 0.092630 | 0.061540 | 0.033500 |
| **75%** | NaN | 15.780000 | 21.800000 | 104.100000 | 782.700000 | 0.105300 | 0.130400 | 0.130700 | 0.074000 |
| **max** | NaN | 28.110000 | 39.280000 | 188.500000 | 2501.000000 | 0.163400 | 0.345400 | 0.426800 | 0.201200 |

11 rows × 31 columns

Fig 2: Summary statistics for the dataset obtained using the describe function (9 features)

This seems to be a very clean dataset i.e. it has no missing values and no outliers can be seen in the summary statistics.

The ‘diagnosis’ or the target variable was converted to a column with numerical categories instead of strings. The labels are now 0, 1 which can be used by the downstream preprocessing steps and machine learning algorithm.

Correlation among the variables was observed in a bid to reduce feature set size before further analysis was done during the EDA.

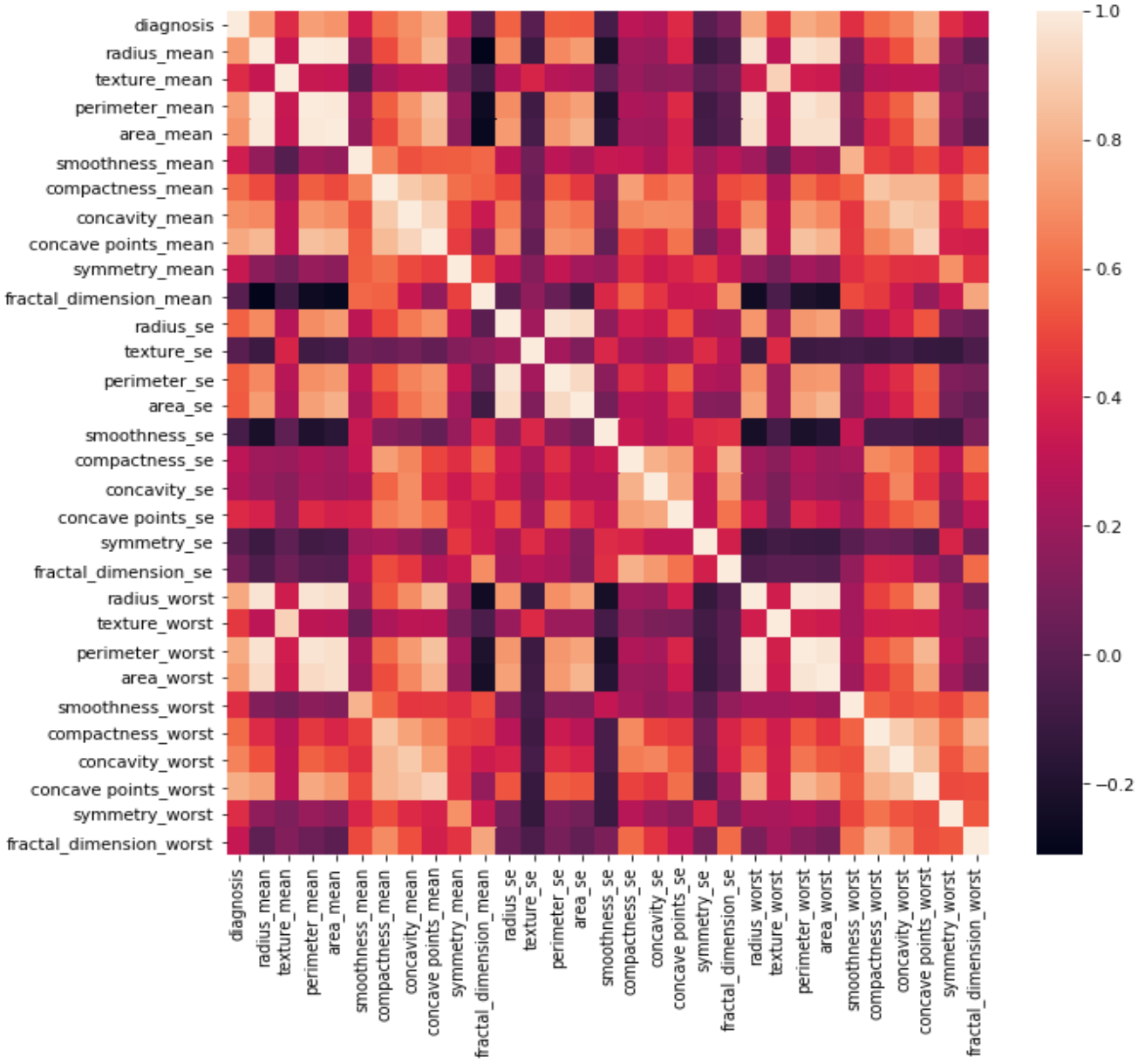


Fig 3. Correlation Matrix

It was observed that some columns were highly correlated and hence from each pair of columns having greater than 0.95 correlation coefficient one column was removed. 7 columns i.e. ['perimeter_mean', 'area_mean', 'perimeter_se', 'area_se', 'radius_worst', 'perimeter_worst', 'area_worst'] were hence removed

This was done as correlated features tend to overstate the importance of underlying characteristics.

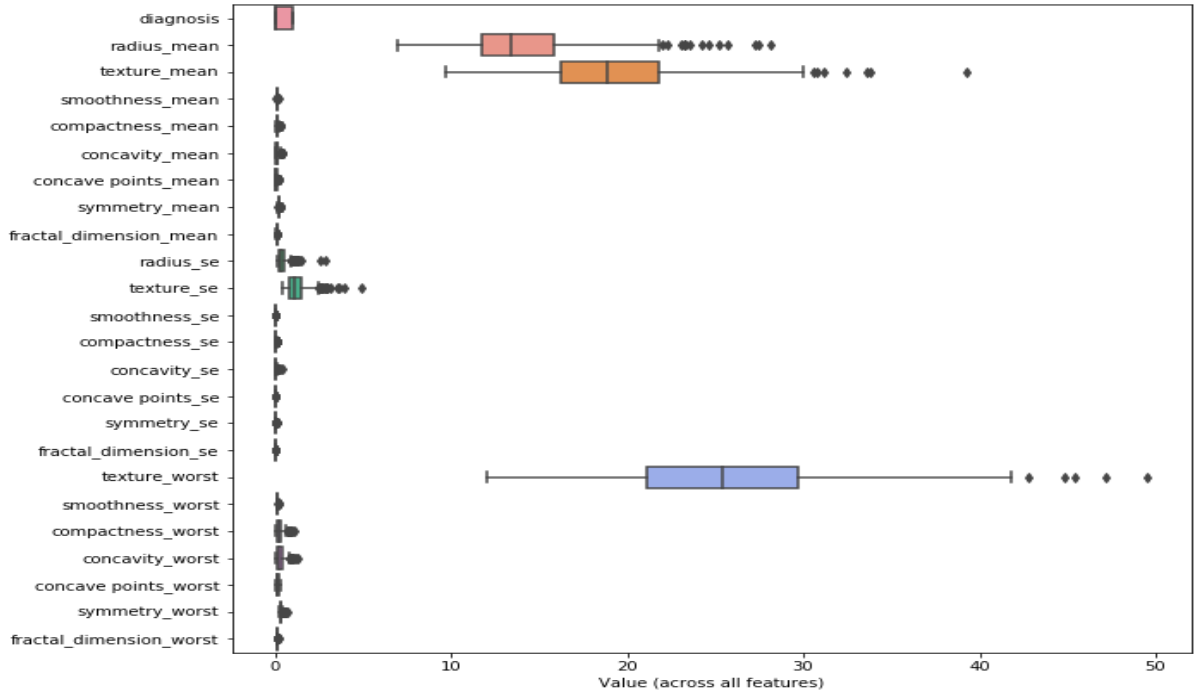


Fig 4: Box plot depicting data distribution in each column

For each remaining column the data seems to be nicely distributed (as was observed using the summary statistics) hence no further modifications were done.

Finally, the target column distribution was checked to find out whether both the labels were adequately represented.

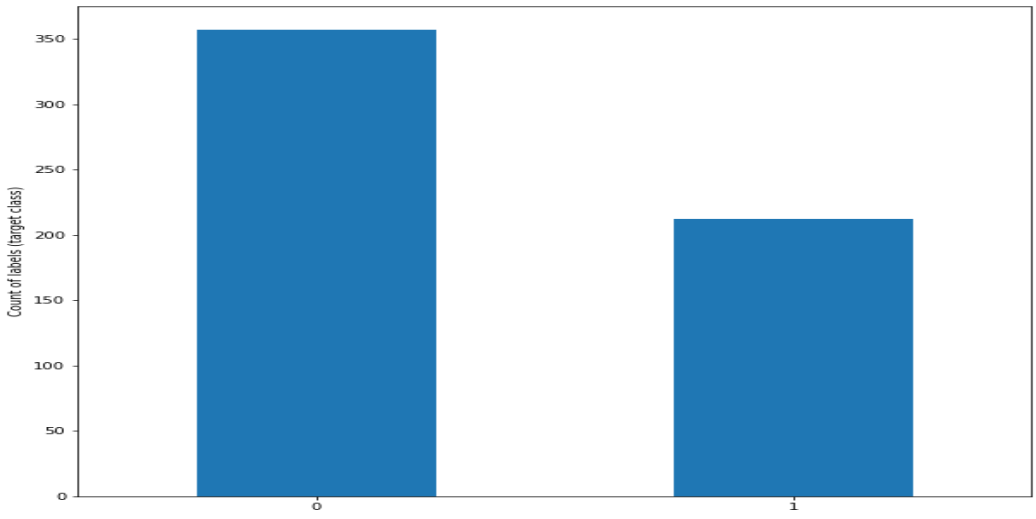


Fig 5: Data distribution in diagnosis class

Plot of different column pairs were plotted for bivariate and univariate analysis.

Blue dots represent ‘0’ Outcome while orange ones indicate ‘1’ as Outcome class

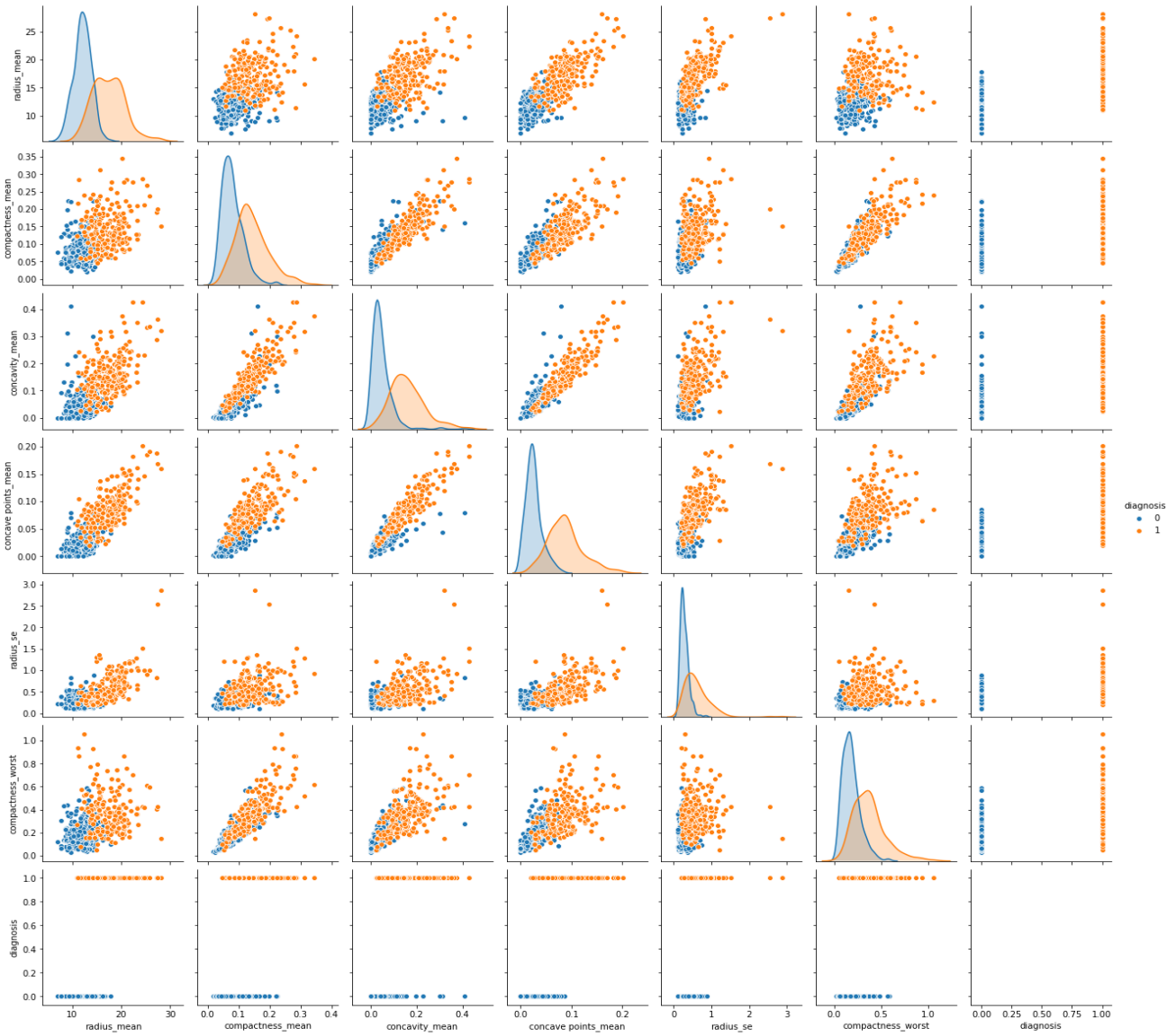


Fig 6: Pair plots of all the variables with diagonal plots being univariate

From the bivariate plots we can see how a clear decision boundary can be observed. All the variables sampled on the basis of the correlation coefficient value with the diagnosis column seem to have very high predictive power.

Data appears to be sufficient for the task, hence no additional data was searched. The feature set seems to be highly relevant.

**Machine Learning Algorithm Used: KNN Classifier**

K-nearest neighbors classifier defined in the scikit learn library was used as the machine learning algorithm to classify the unknown samples.

K-nearest neighbors, even though has no training involved, typically performs very well on such tasks, is easy to interpret and thus is widely used in academia. The algorithm can be defined as follows:

As the name suggests, it uses class information of the k-nearest neighbors sorted by a similarity metric and assigns class to the row under consideration during a prediction task. No training takes place and hence no loss function has to be defined, only the distance to the row has to be calculated and stored in the memory.

For our use case:

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + \ldots + (x_n - x'_n)^2}$$

Similarity Metric Used - Euclidean Distance

Weights - ‘uniform

K value - decided using cross validation

Example: predicting for any sample x using training data denoted by T

1. Distance to x from each row in T is calculated using the feature set and decided distance measure which typically is Euclidean distance but often cosine similarity, minkowski distance is used (depends on the end application)
2. Top K neighbours are identified by sorting the rows in set T by distance calculated in the previous step
3. Class for the sample is then calculated on the basis of the number and weights of the neighbors belonging to each class. The weights can be uniform (our case) where all the neighbours are equal , inversely proportional to distance or calculated using any other function.

**KNN Training and Prediction Methodology**

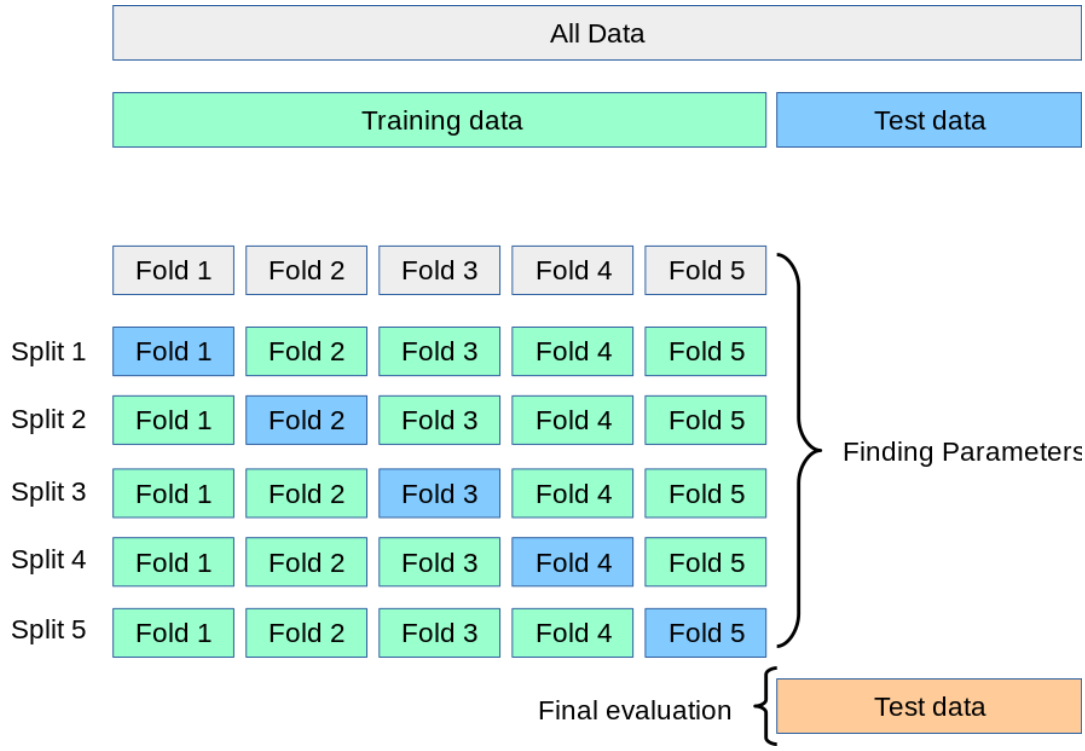


Fig 7: Training and Testing methodology (source: https://scikit-learn.org/stable/modules/cross_validation.html)

Using this methodology, it was ensured that the test set is unseen to the model till the parameters are finalized and model is trained. This prevents data leakage i.e. over optimistic performance on the test set and poor performance on unseen data.

Post the train/ test split the remaining features in the training as well as testing dataset were standardized using the StandardScaler function defined in the scikit-learn library

$$x_{scaled} = \frac{x - mean}{sd}$$

Fig 8: Standardization formula

K value was finalized using 5 fold cross validation and was then used to predict the target class i.e. diagnosis for the test set.

Accuracy score was used as the metric during cross validation and mean accuracy score across all the folds was used to finalize K.

Odd values of k were tested by cross validation as mentioned above KNN with uniform weights uses majority voting to assign a class to the row. An even number can result in ties and reduce the accuracy.

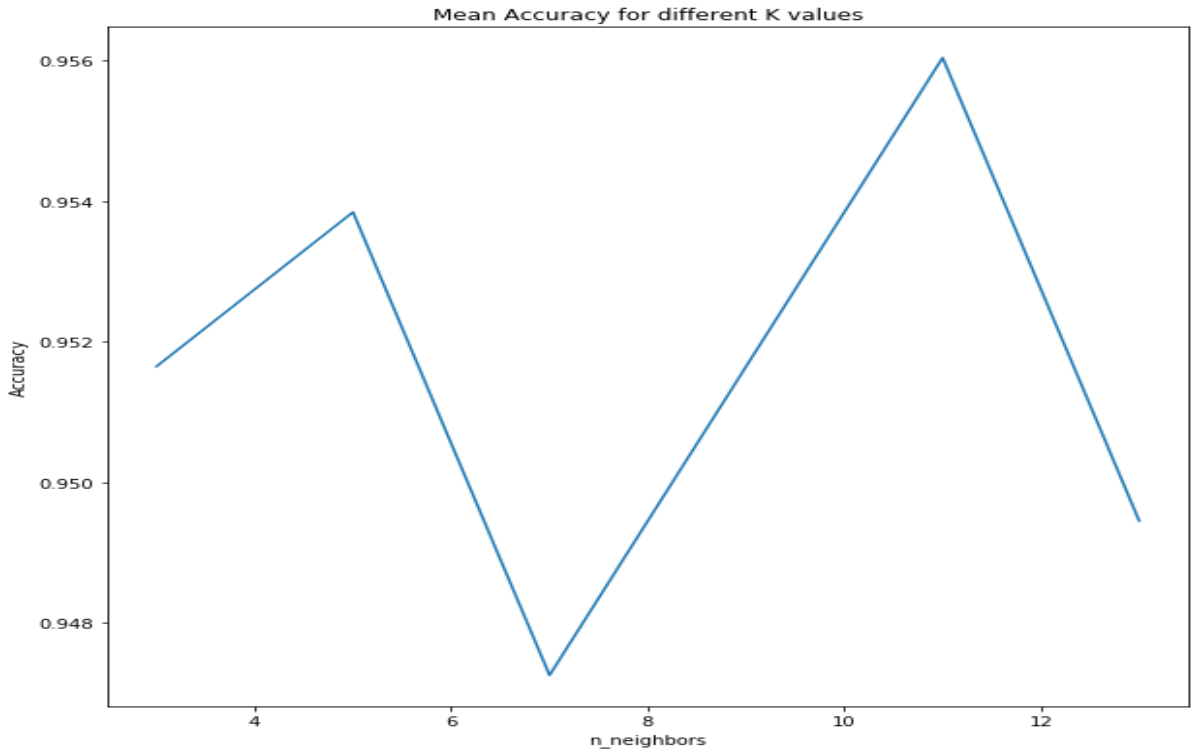


Fig 9: K value vs mean accuracy plot

Most Optimal Value of K - 11

Using K=11 prediction accuracy of 95.6 % was obtained

## Results

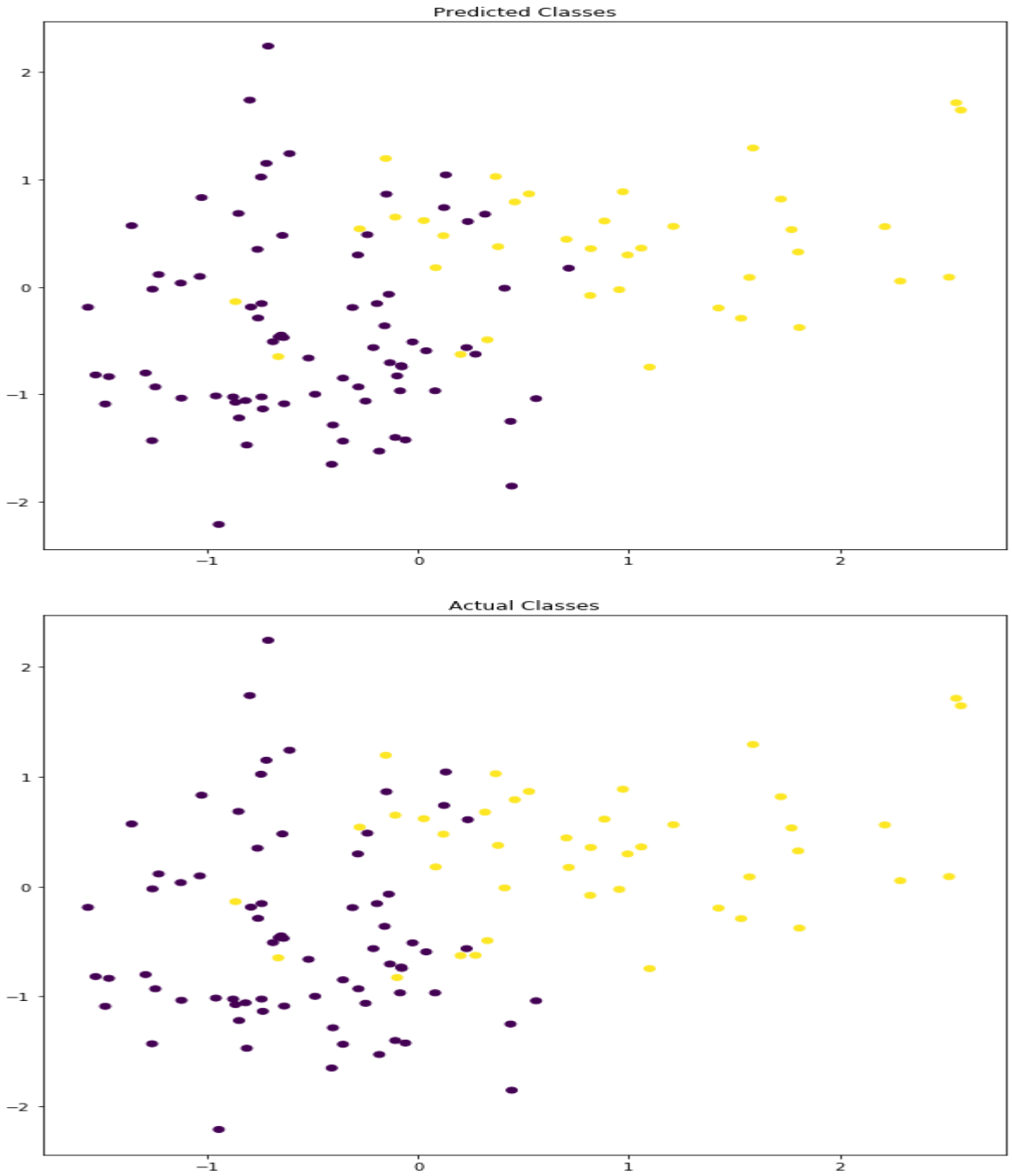


Fig 10: Classifier Performance Evaluation

A 2-dimensional plot using the first two features in the dataset was created for comparison of the classifier performance. The actual and the predicted classes are largely similar barring some points which can be easily seen in the plot.

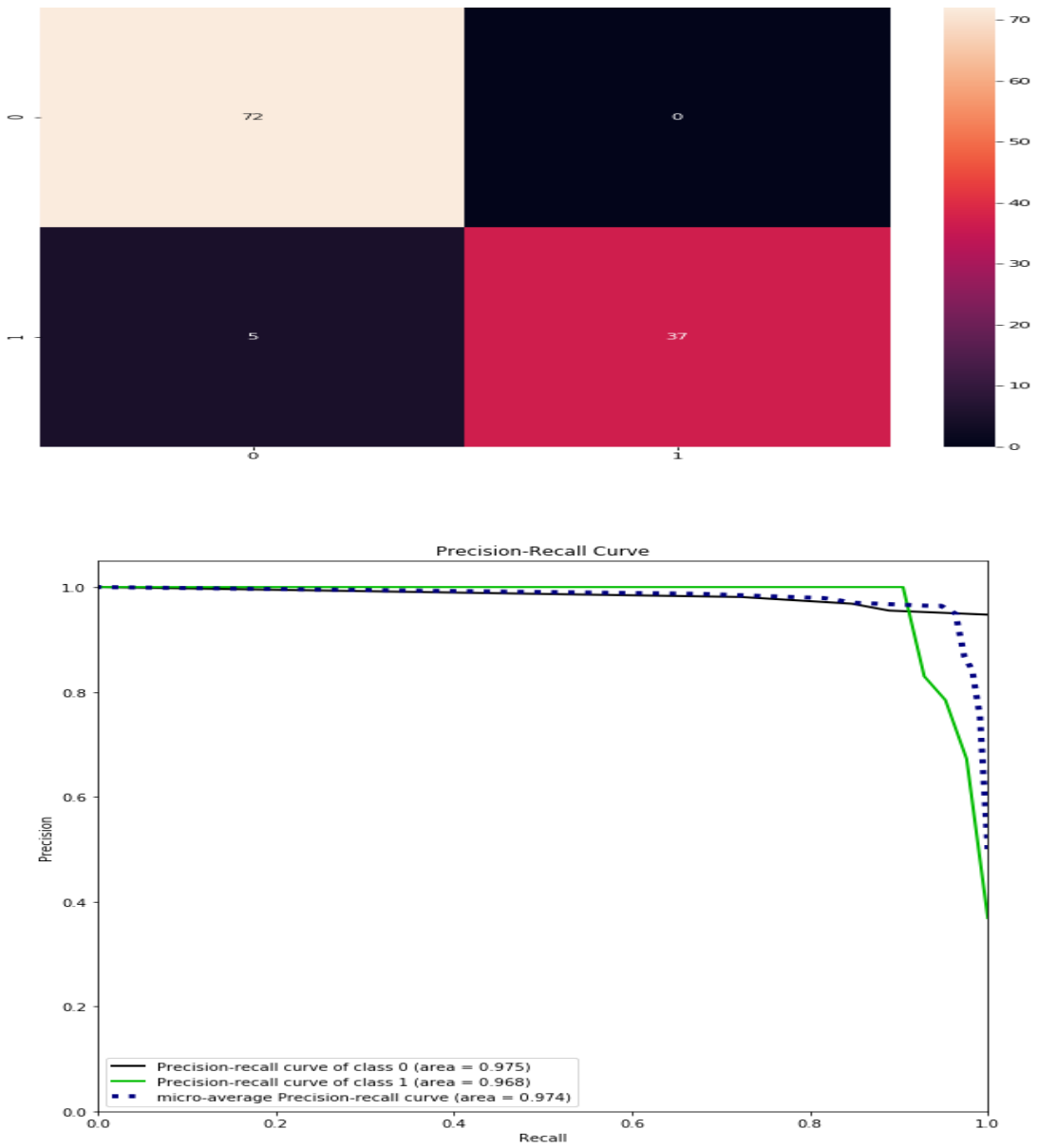


Fig 11: Confusion Matrix and precision recall curve

As can be observed from the confusion matrix there were no false positives which is a great thing but the algorithm predicted 5 false negatives. Still, it can be said that machine learning can be an effective first layer of screening for breast cancer based on the input features. Also, the very high precision along with a very high recall percentage can be observed from the precision recall curve indicative of very good classifier performance.

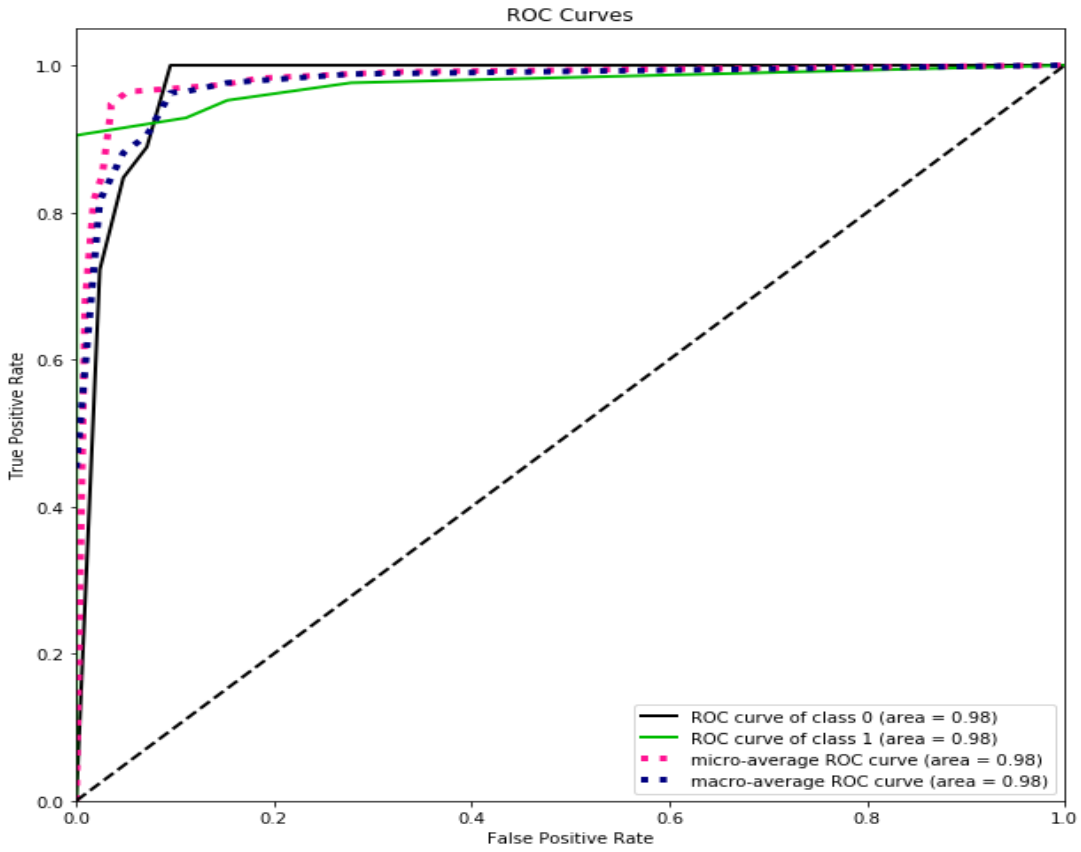


Fig 12. ROC Curve

ROC curve is plotted using difference threshold values and observing the True positive rate and false positive rate for each of those thresholds as we move along the curve i.e. each point represents a sensitivity/specificity for a particular decision threshold. The area under the ROC curve (AUC) measures the ability of the model to distinguish between the two classes which for our model is near perfect i.e. 0.98

**Evaluation of the methodology**

Using the Exploratory data analysis, the data was observed for any anomalies and one feature from every highly correlated feature pair was removed. This ensured that highly correlated features didn't overstate their predictive power during prediction.

Further, the variables were scaled using the standardscaler function to ensure that all the features were having the same scale. Post the completion of preprocessing, robust 5 fold cross validation was used to find out the optimal value of k. The accuracy score over 5 folds during cross validation and prediction accuracy on the test set was similar indicating similar distribution of classes in the sets. This was intentionally done using the 'stratify' option in the train test split function and using validation sets for parameter tuning instead of test set.

**Limitations Of Knn**

Considering KNN is a relatively simplistic model, the model performed really well on the dataset. Perhaps the removal of correlated variables could have been done using the target variables which may have given additional boost to the accuracy.