# Random Forest

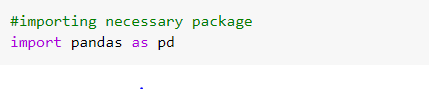
**Random Forest:** It is one of supervised learning algorithm where we use both classification and regression i.e., by building decision trees on several samples and considering the majority for classification and average for regression.

This model is preferred for classification problems as we get better results for them.

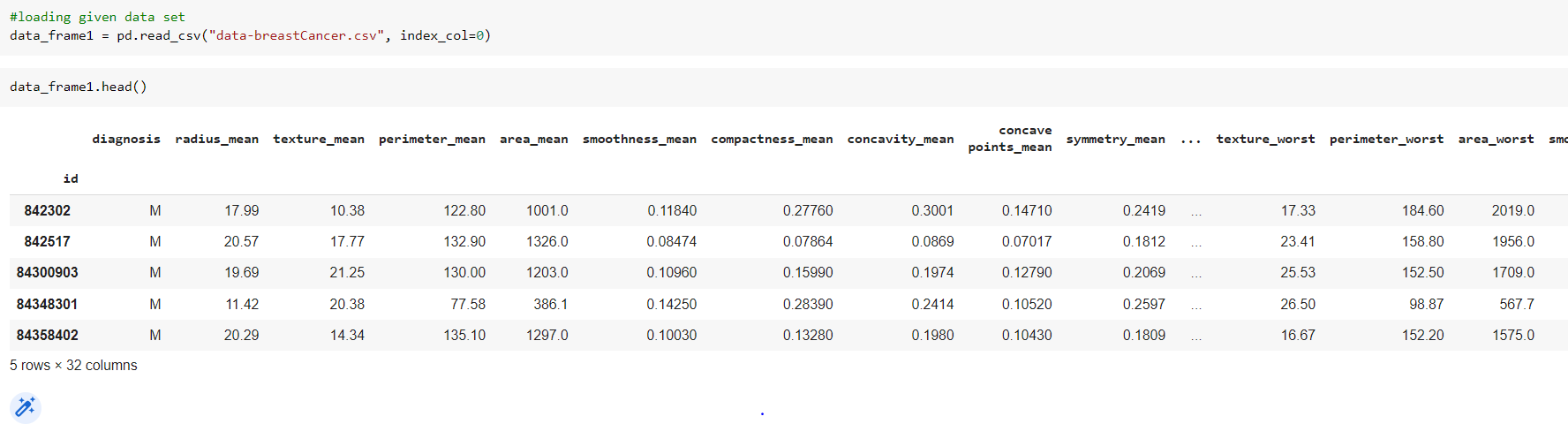
**Given DataSet:**

Here we have a total of 29 features that were computed for each cell nucleus with an ID Number and the Diagnosis (Later converted to binary representations: Malignant = 1, Benign = 0).

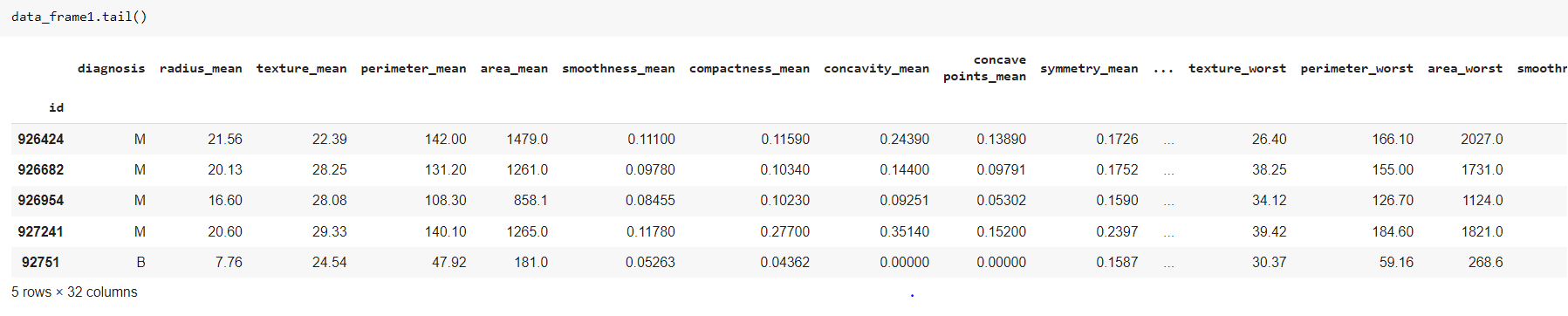
**Code:**



Here we are importing necessary packages

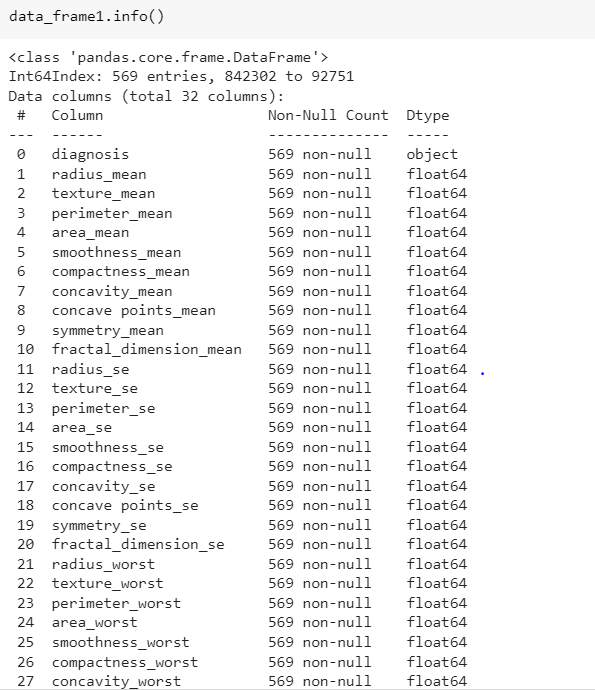


Here we are loading given data set and then by using head function we are getting first 5 rows in the given data set.

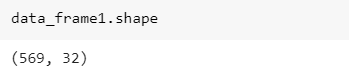


Here we are getting last 5 rows of the given dataset by using tail function.

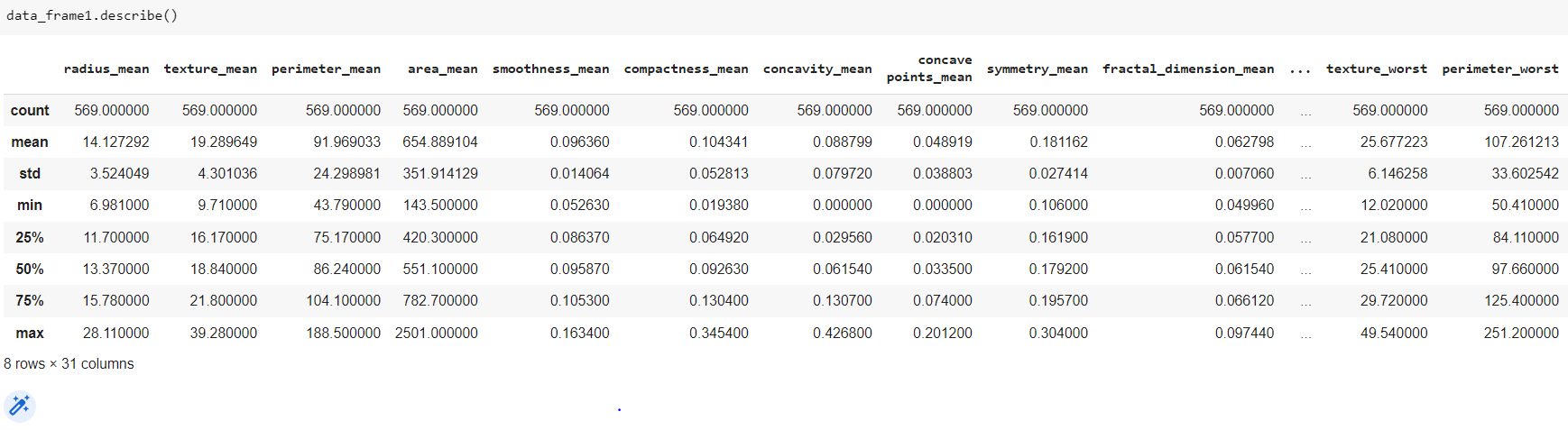
**Data Pre-Processing:**



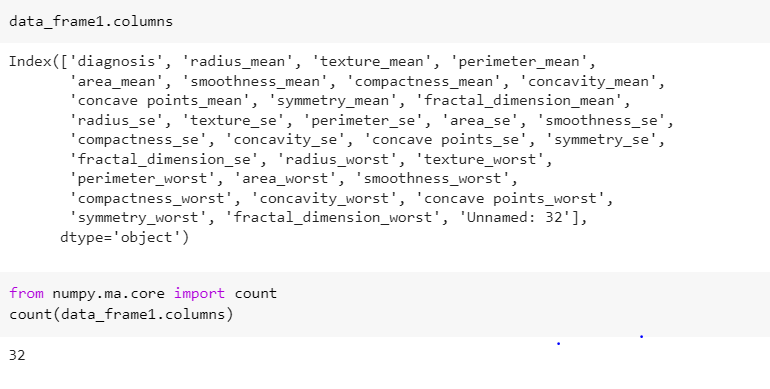
Here we are getting information such as datatype, count etc., of all the 32 columns present in the dataset.



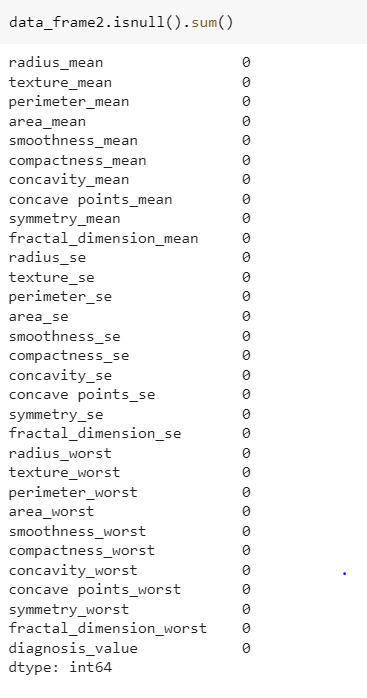
Here by using shape function we are obtaining the count of rows and columns in the given data set.



Here we are computing and displaying summary statistics of the given dataset using Describe function.

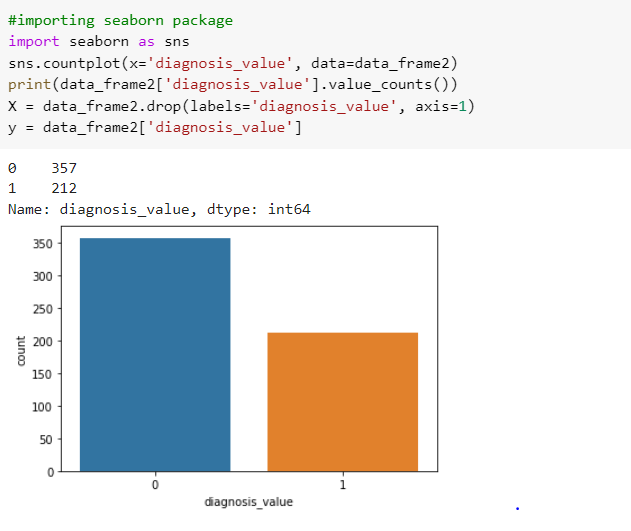


Here we are getting details of columns



Here we are checking if there are any null values in the given dataset.

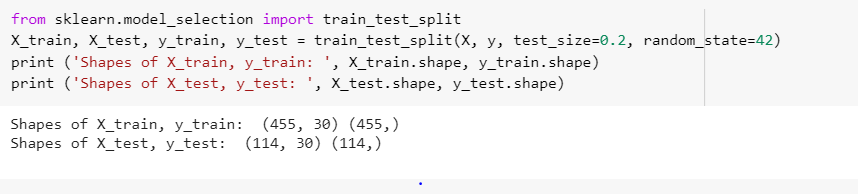
**Data Visualization:**



Here M as 1(malignant) -212 and 0(benight) -357 diagnosis are the binary values and then we are splitting the given data into X and Y variables.

Here X has all the values except the diagnosis\_value and Y has diagnosis\_value only.

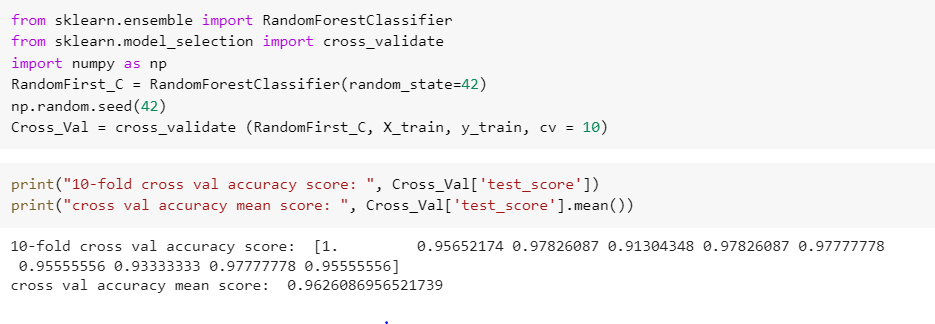
**Splitting data:**



Here train size is 80% and test size is 20% and we are using train\_test\_split model for creating train data and test data.

**Applying Model:**

**K-fold cross validation:**

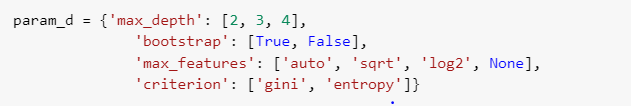


Here in K-fold cv, we are using training data to further split it into K number of subsets, called folds, then iteratively we are fitting the model k times, and then each time we are training the data on k-1 of the folds and then we evaluate it on the kth fold. Finally, here we are averaging the performance on each folds to get final validation metrics for the model.

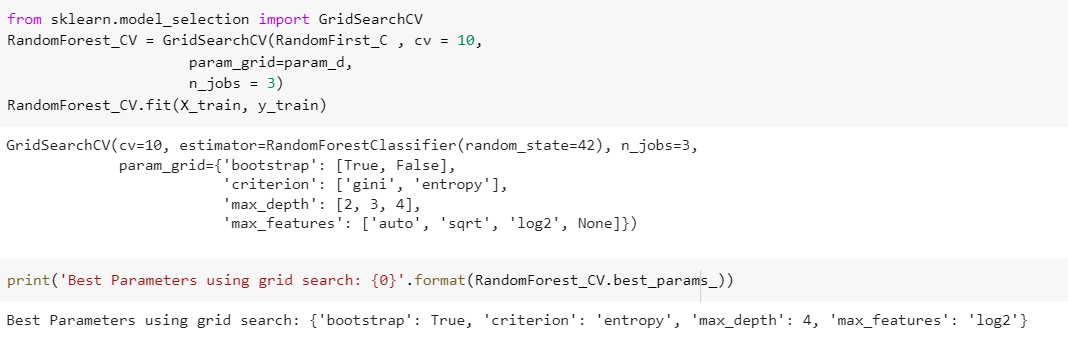
By using this model we obtained an accuracy score of 0.96. Additionally, we can still improve the performance by tuning the hyper parameters.

**Tuning of Hyper-Parameters:**

By using different model setting every time, we are performing many iterations of the KV fold procedure.



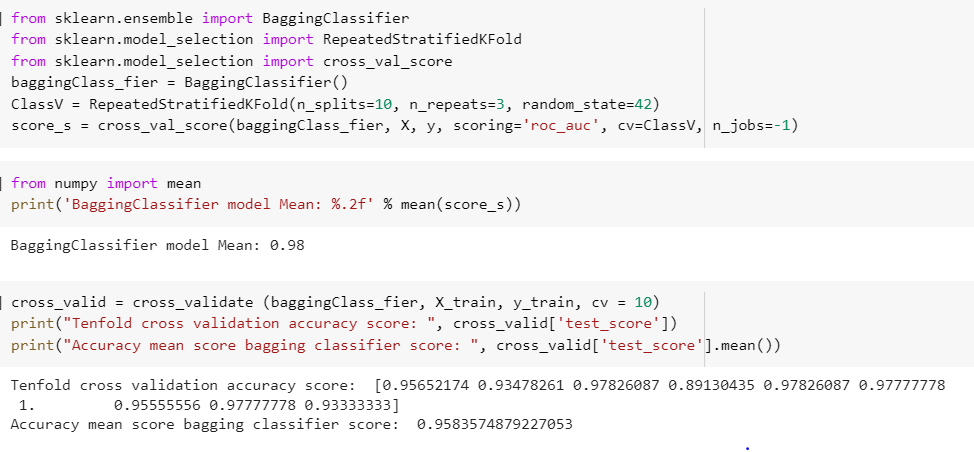
Here we are considering max\_depth, bootstrap, max\_features and criterion features for tuning to enhance the parameters.



Here we are passing the given values through random forest classifier to get best parameters using best\_params of GridSearchCV in sklearn package before tuning.

**Bagging Classifier:**

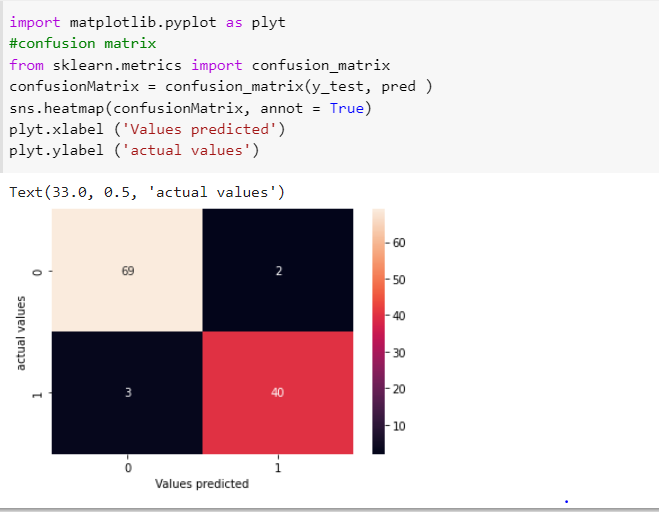
Here in bagging classifier we are forming final prediction by fitting the base classifiers each on random subsets of original data and then aggregate their individual predictions either by majority vote or through averaging.



Here we are using Bagging Classifier model and we obtained accuracy score as 0.95

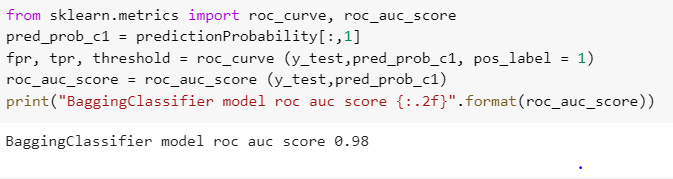


Here we obtained accuracy score of test set which is 0.96.

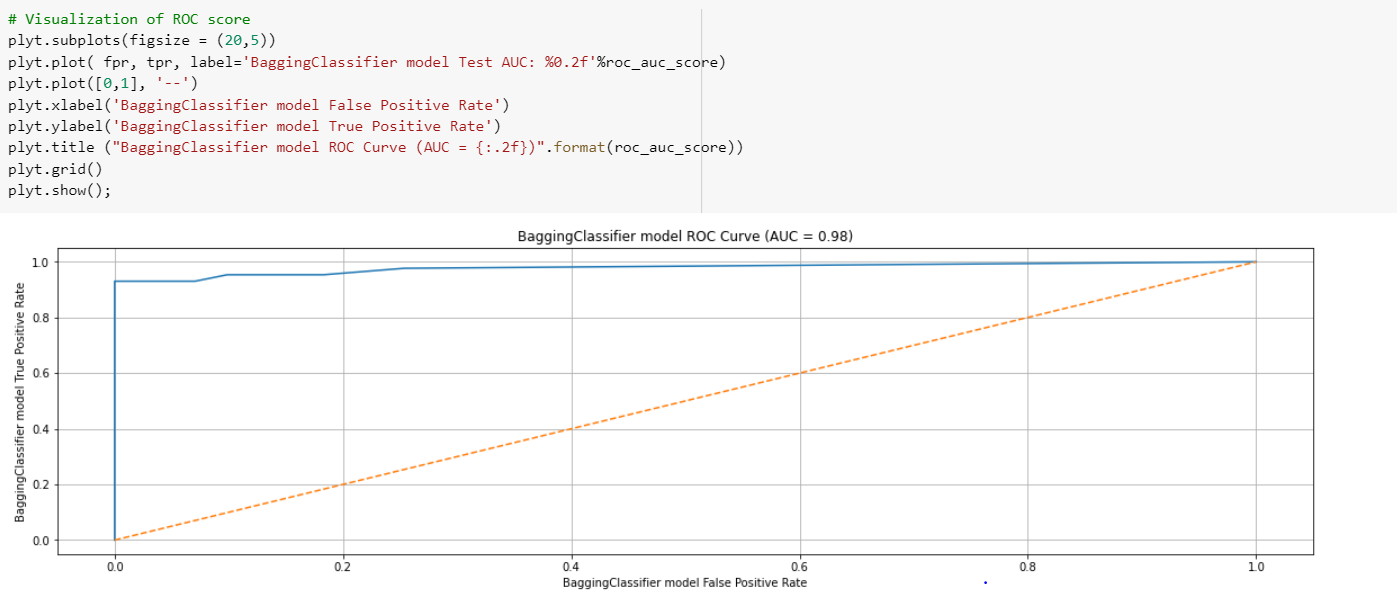


Here by using confusion matrix, we can conclude that in 114 patient records 71 are actual zeroes and 43 are actual 1s.

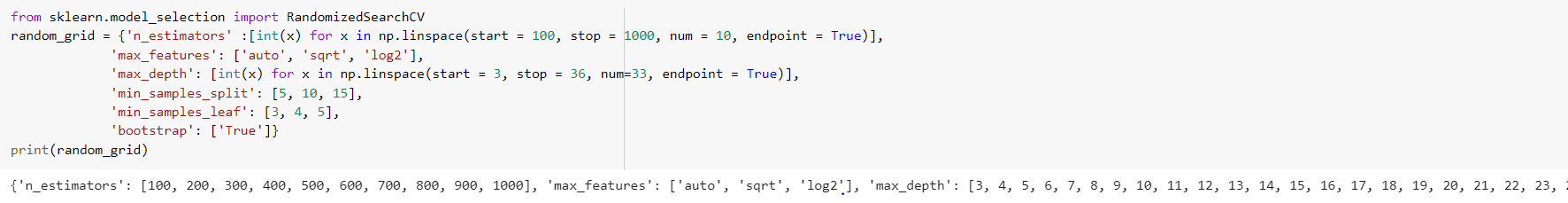
**Metrics:**



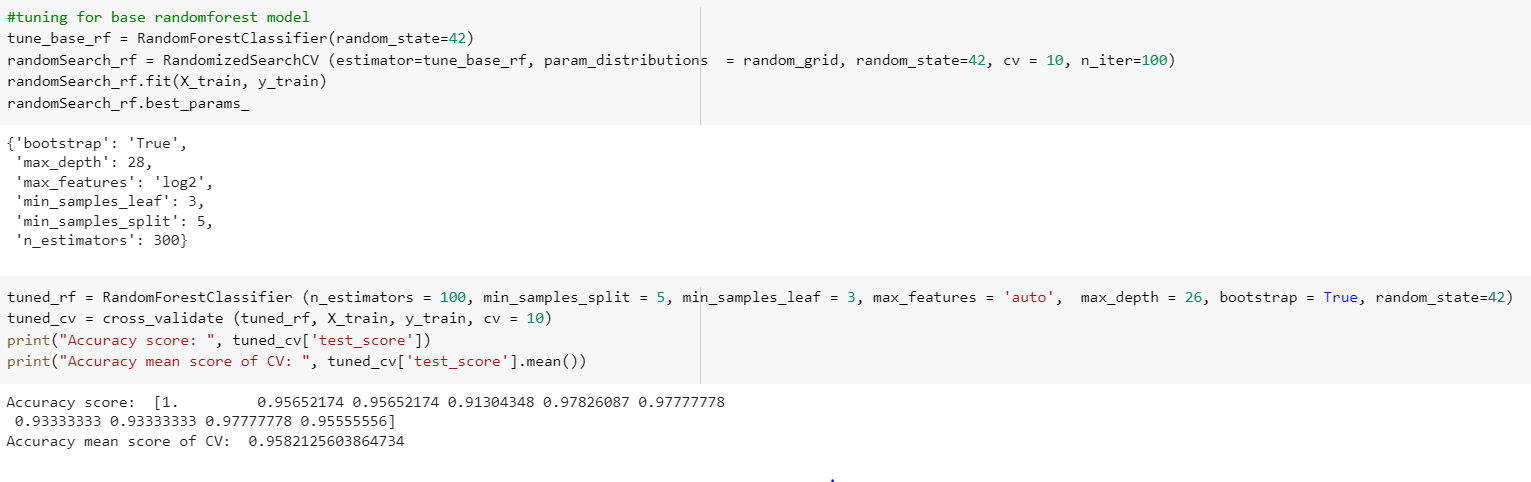
Here we are taking probability of class\_1 on test set, True positive rate and false positive rate and then we have to note that Receiver Operating characteristic curve(ROC) is used to plot the performance of classification model at all classification thresholds. In order to have a better model we have to make sure AUC is more than 0.5.



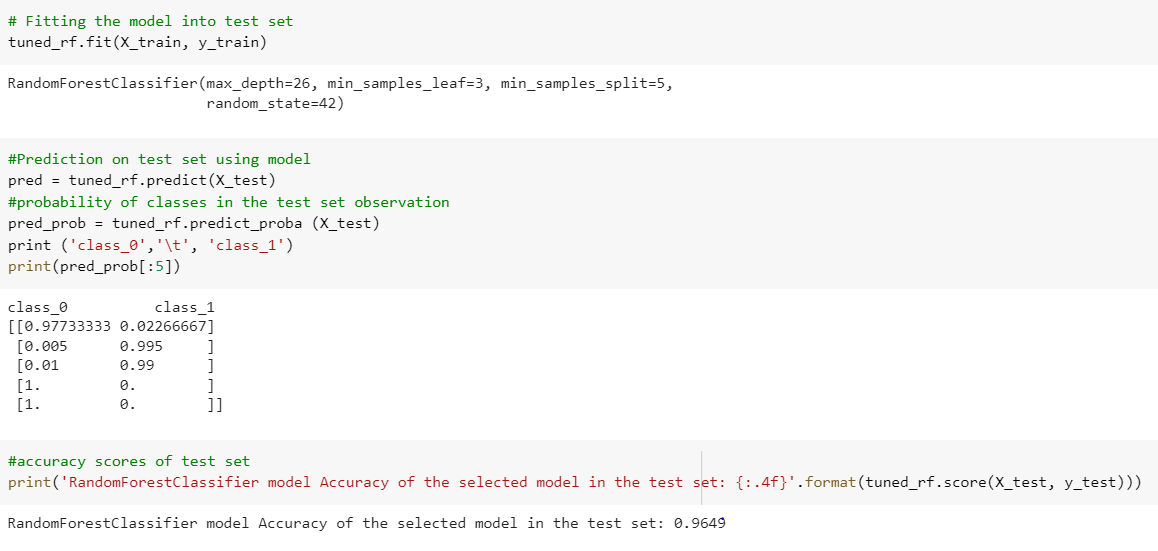
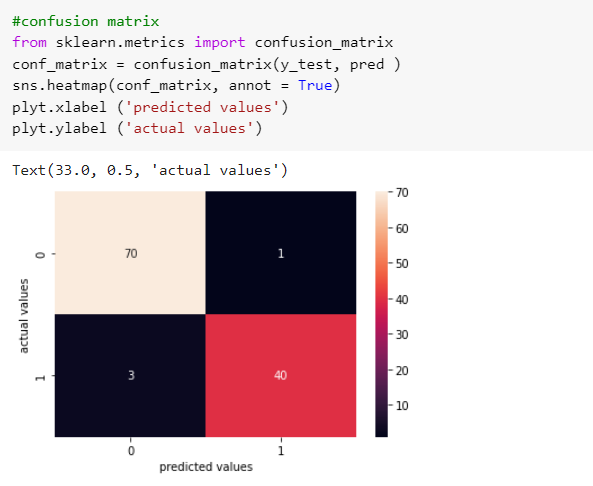
Here we are visualizing the ROC score by plotting the curve. False Positive Rate is considered on X axis and True positive rate is considered on Y axis.



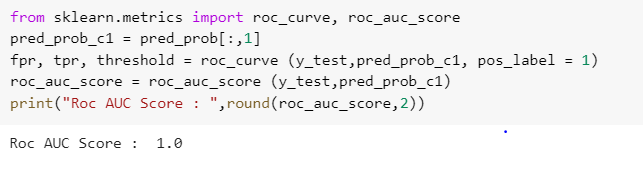
Here we are considering estimators, max\_features, max\_depth, min\_samples\_split, min\_samples\_leaf and bootstrap features.

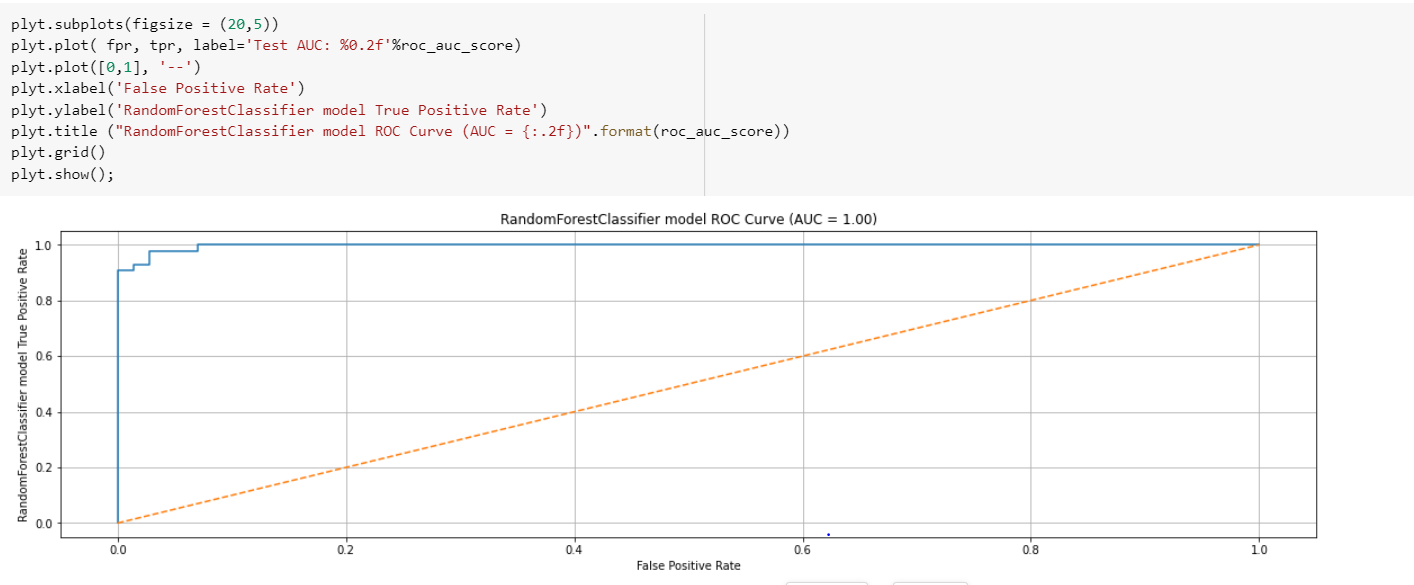


Here we are tuning using random forest model.

Here we are fitting the model into test set and we are calculating accuracy of selected model in test set. 

Here we are using confusion matrix and get actual values and predicted values.





**Conclusion:**

The Bagging Classifier accuracy is 98% where the random forest classifier is 100.0%. The hyperparameters are useful for avoiding the overfitting or high variance of the model on the unseen data. We decide to go with the random forest classifier as our final model as the accuracy is high for it.