**COVID-19**

**CLASSIFICATION**

**BY**

Hanan Fared Mohamed Omara 20398559

**Supervisor**

Marwa Elsayed

CONTENTS

[1. ALGORITHMS 3](#_Toc128031965)

[1.1 Data Loading 3](#_Toc128031966)

[1.2 Visualize Data 4](#_Toc128031967)

[1.3 Preprocessing Data 6](#_Toc128031968)

[1.4 Create ML Model 6](#_Toc128031969)

[2. Final Result 17](#_Toc128031970)

# ALGORITHMS

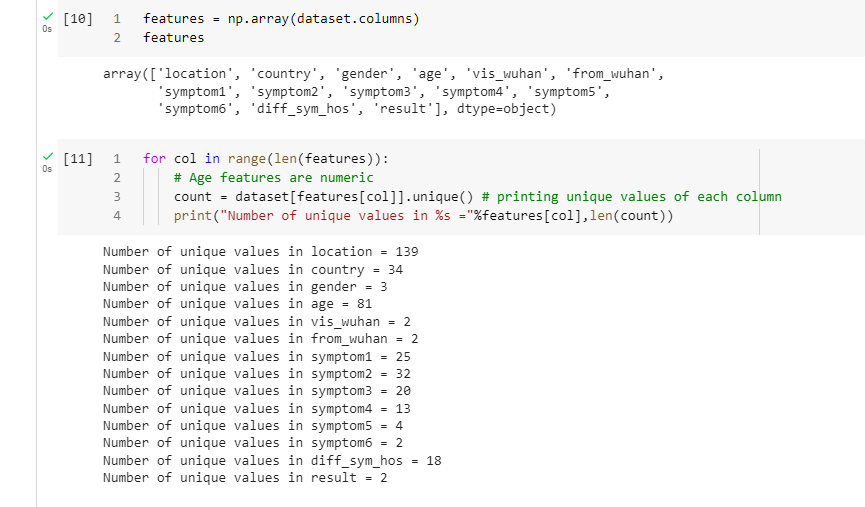
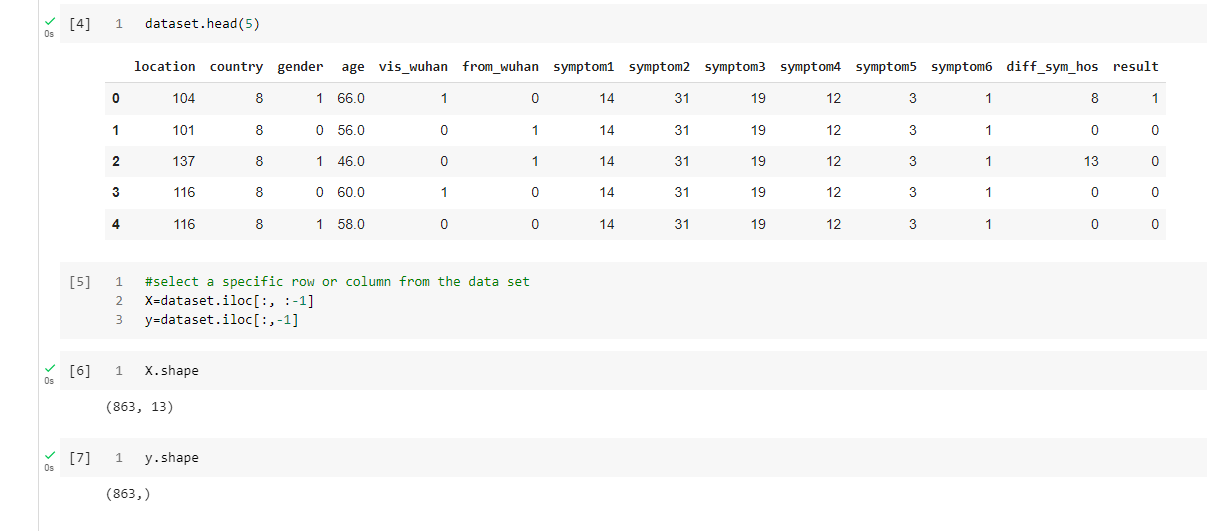
## Data Loading

### Load data

Import different libraries and Load data from csv file to the work as a Data Frame using pandas.



### View more information about data

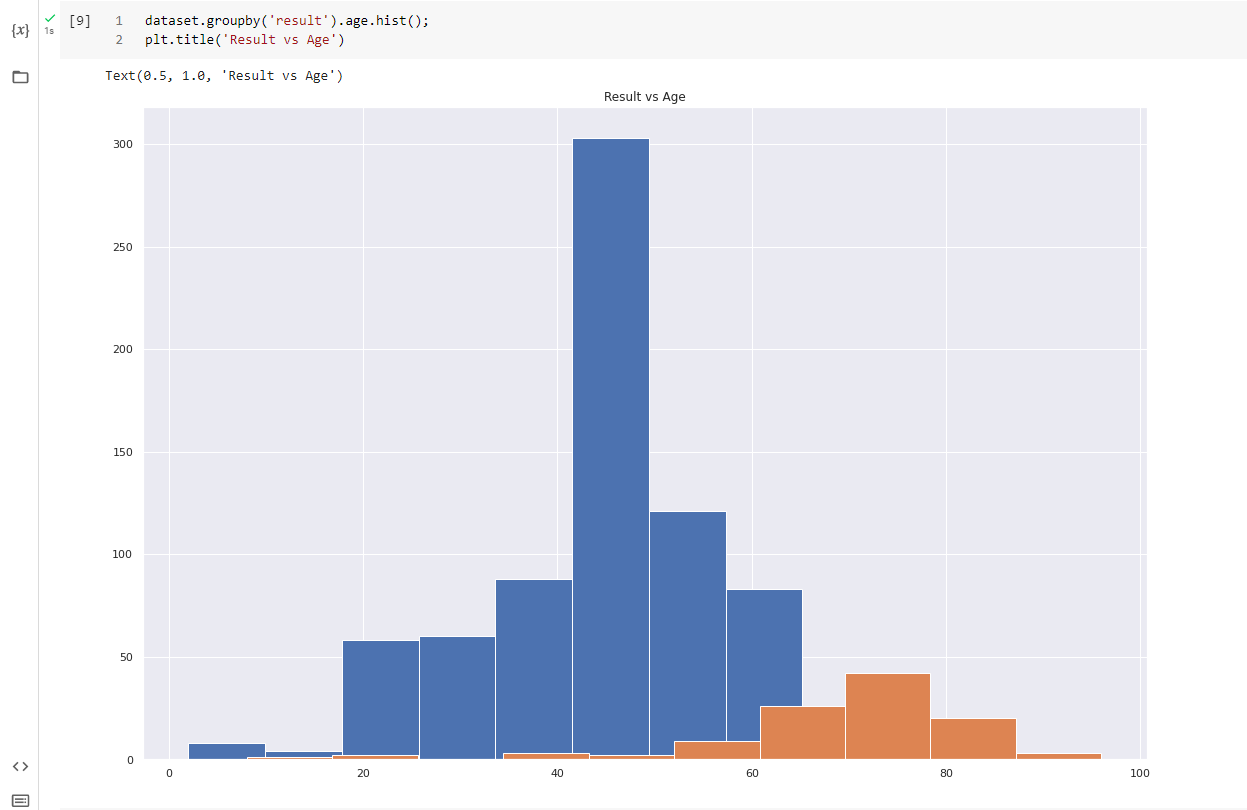


## Visualize Data

In these step we need to view more plots about our data to figure out more information about data set we work with

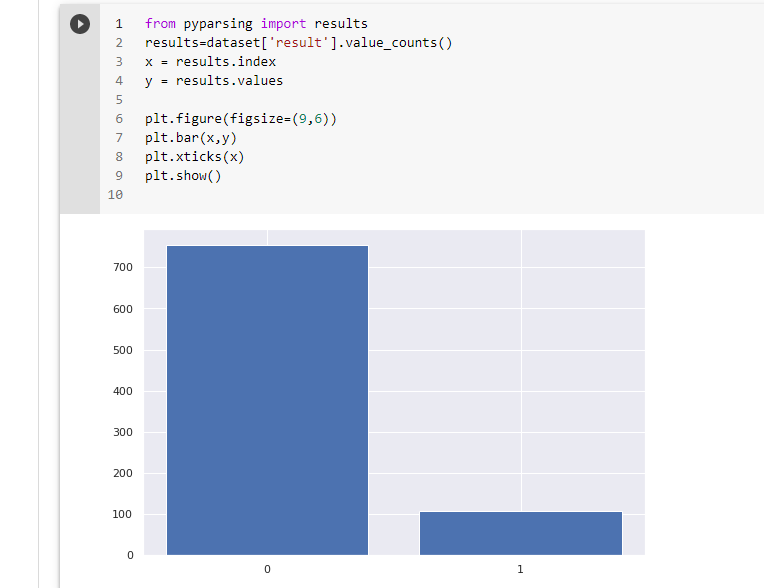
### View the most frequent age get an infection

From chars we can notice that the most cases in age between **40 and 60** but cases above **50** is the greatest possibility of death (Now we are certain that 'Age' is a key feature for predicting the results.)

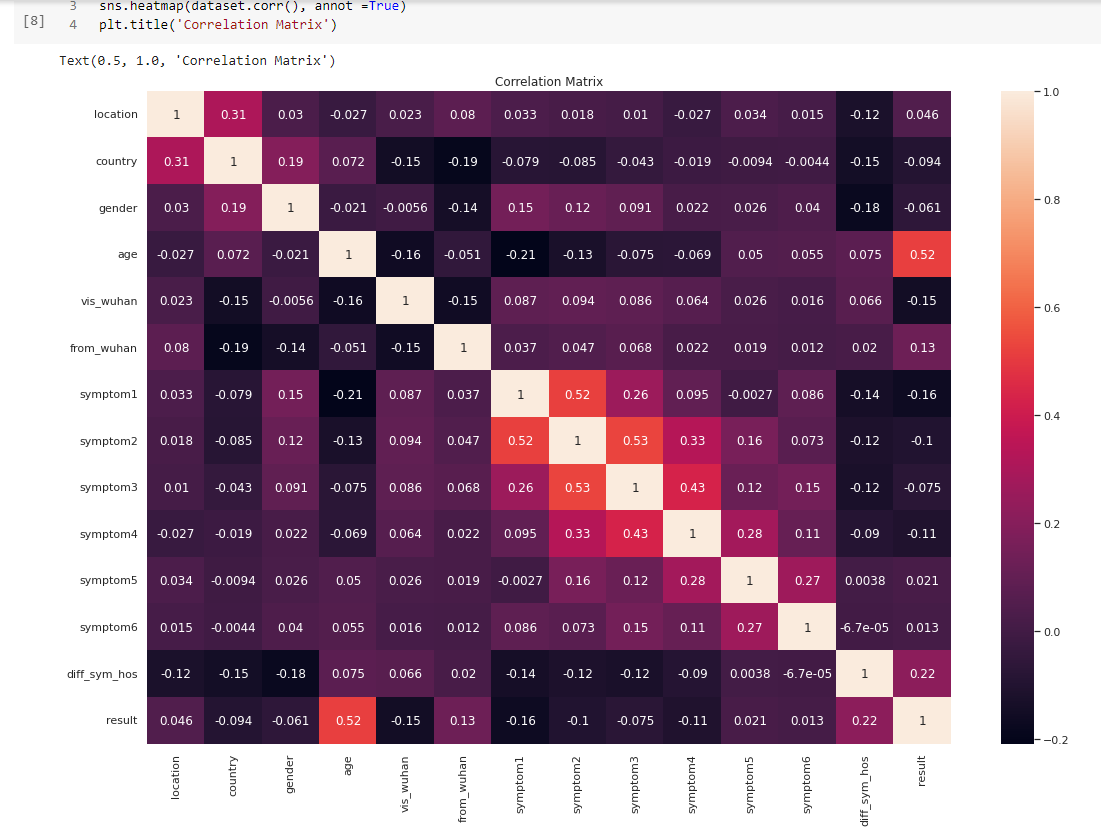


### View if data balanced or not

We can figure that the data is not balanced as number of Recovered is higher than death cases.



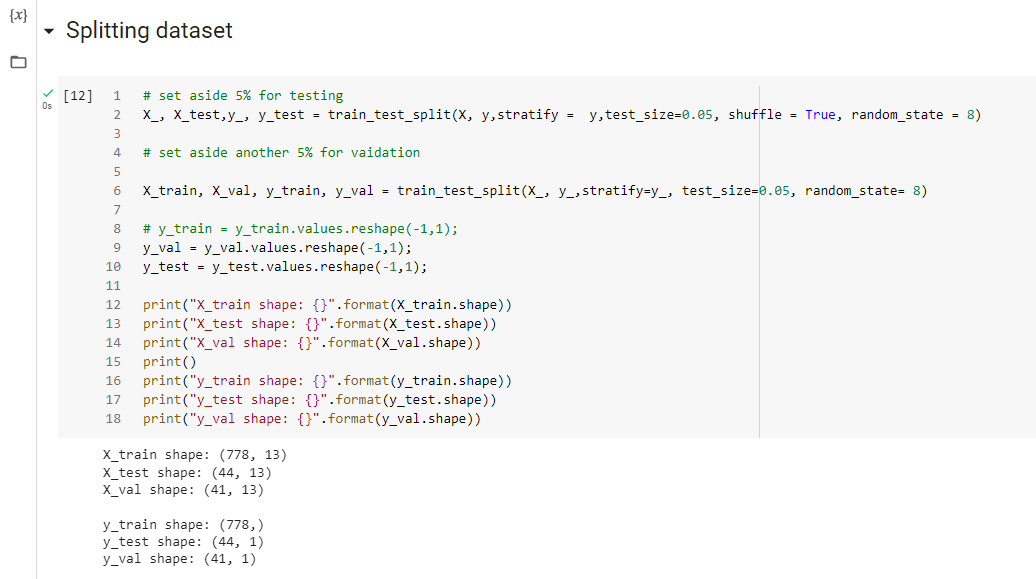
### View correlation between all features



## Preprocessing Data

### Split data

Now we need to split our data set into train and test but from our visualization we find that data is imbalanced so we will need to split data with stratify method to be sure that percentage of each class in train & test data set is equally



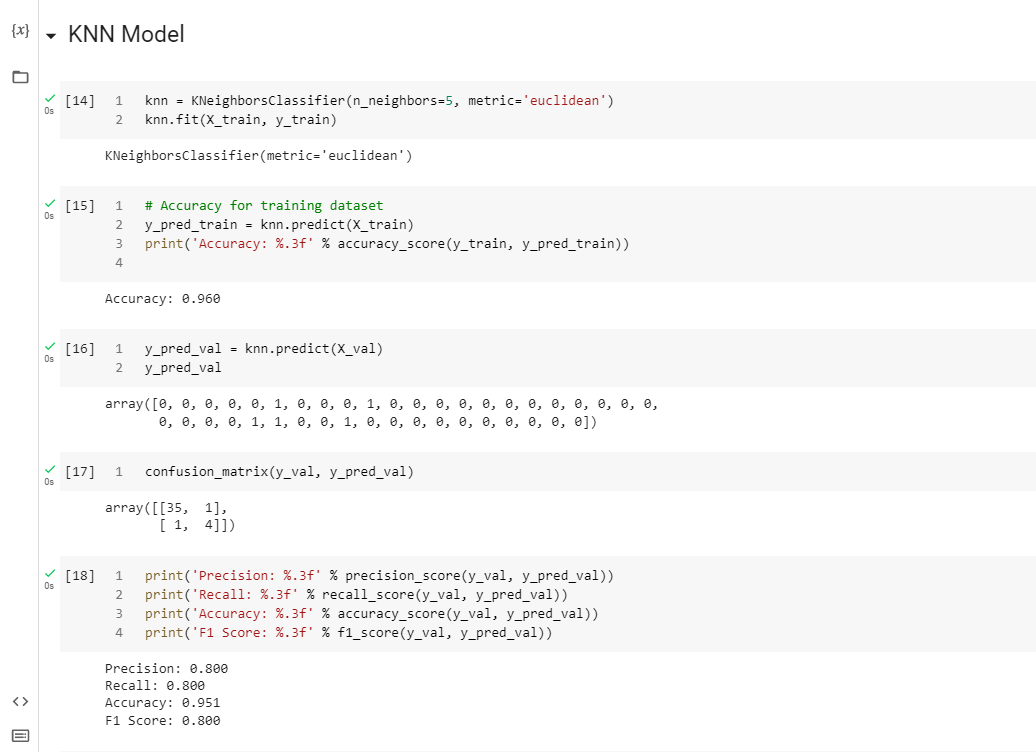
## Create ML Model

Now we will start to create our model we will build four different models then train them at our data set to find what is the most suitable model for our case also we will use grid search technique to tune our hyper parameters

### K Neighbors Classifier

#### Create Model

We create KNN model and use k=5

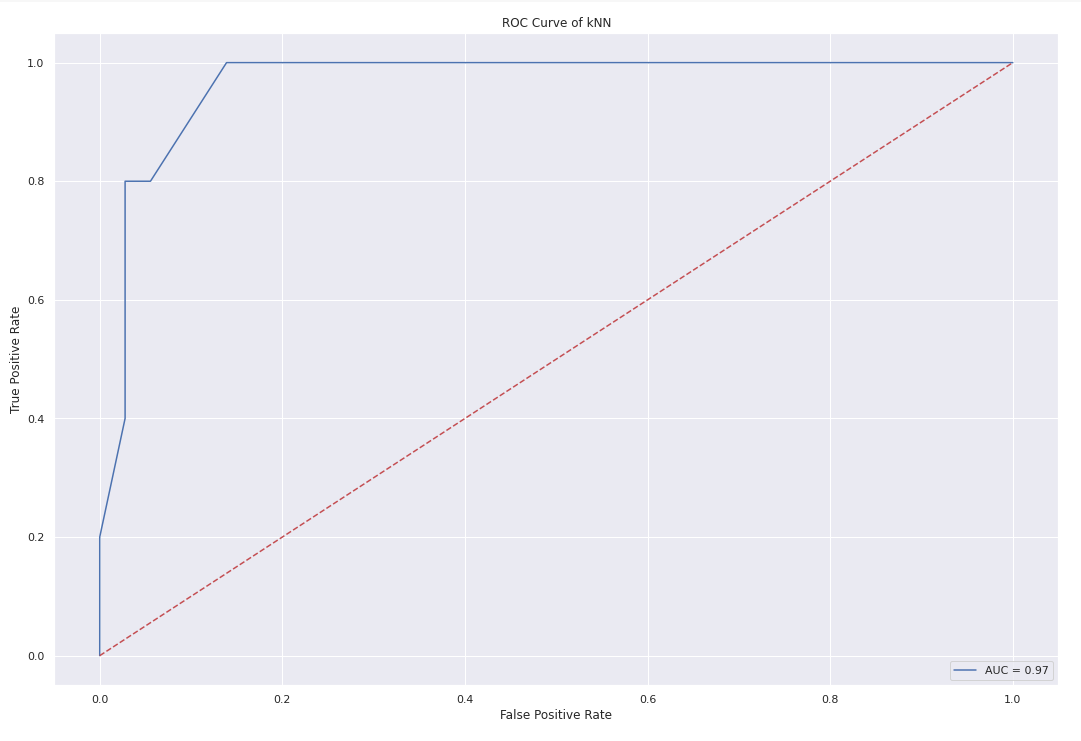


#### The Model Results

For KNN model we find that the recall and precision accuracy is not good



#### Calculate AUC & plot ROC Curve

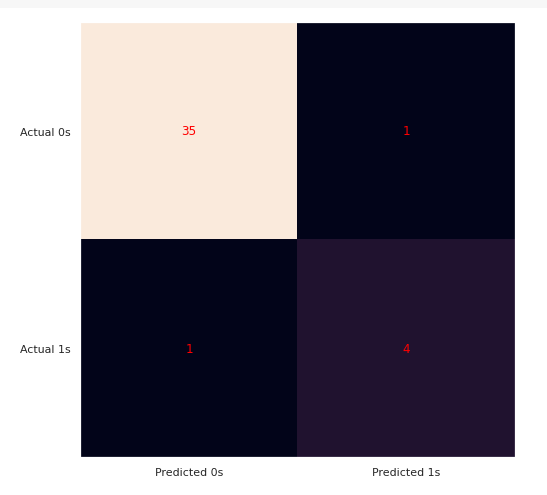
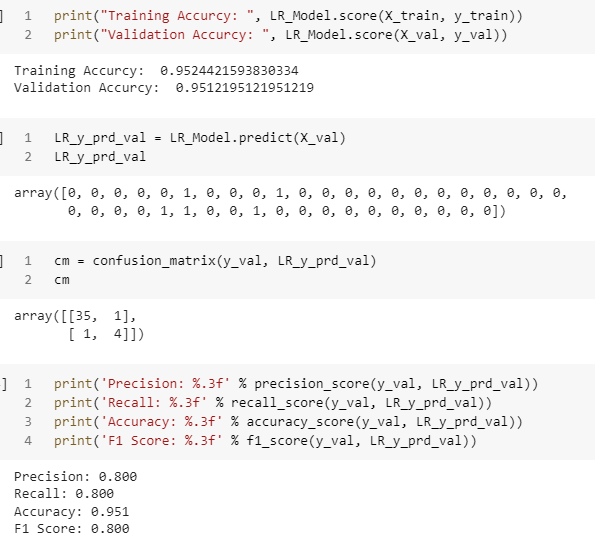


### Logistic Regression

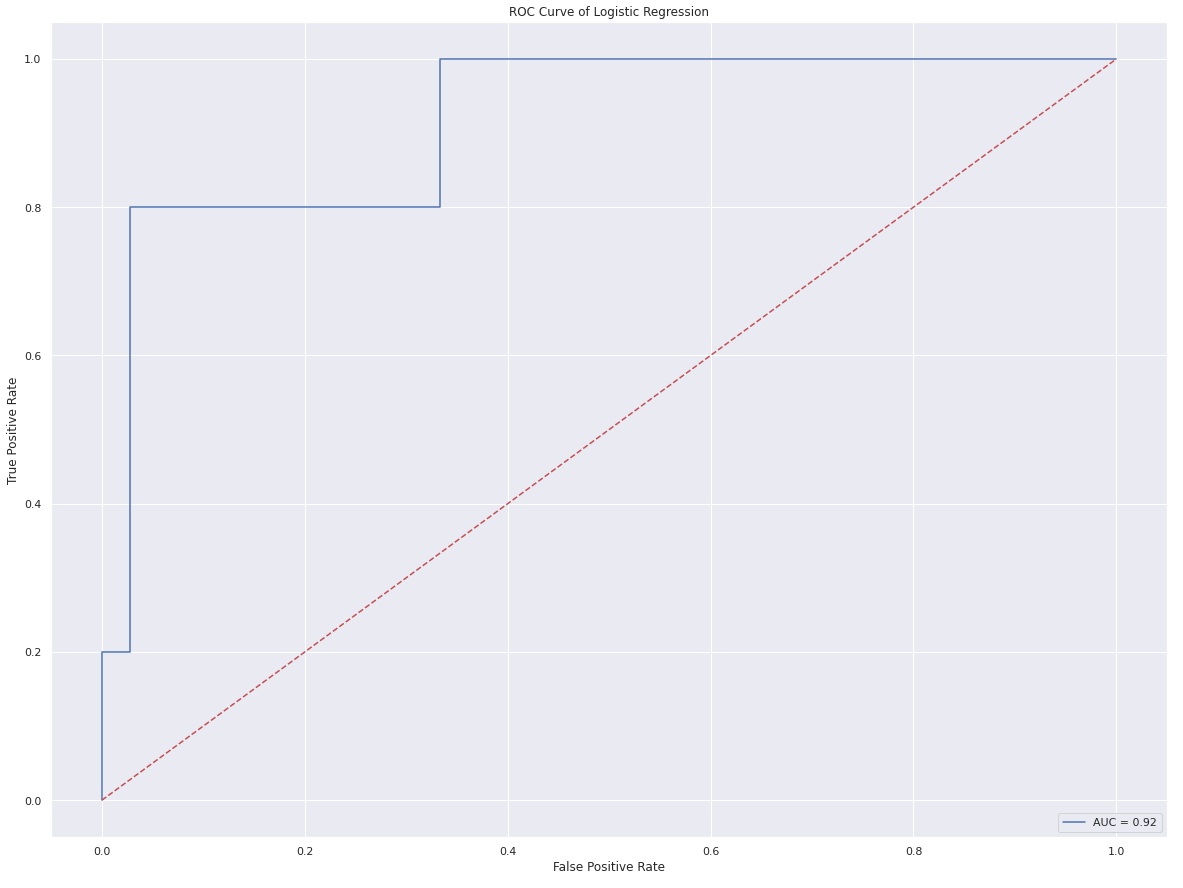
#### Create Model

We create Logistic Regression model and the hyper parameters are (random\_state=0, solver='liblinear')

#### The Model Results



#### Calculate AUC & plot ROC Curve

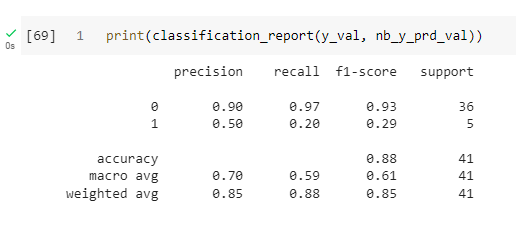
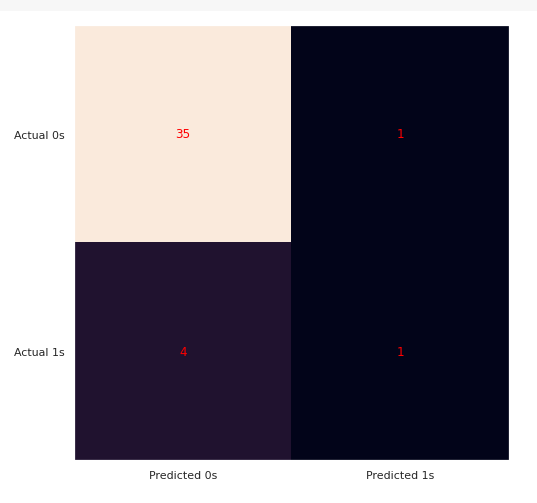
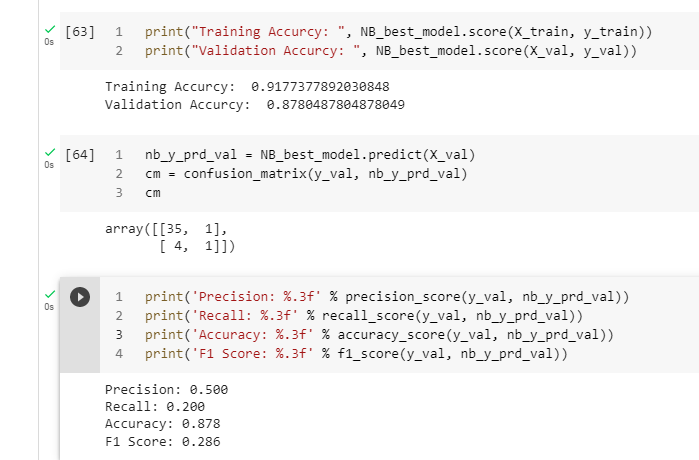


### Naive Bayes

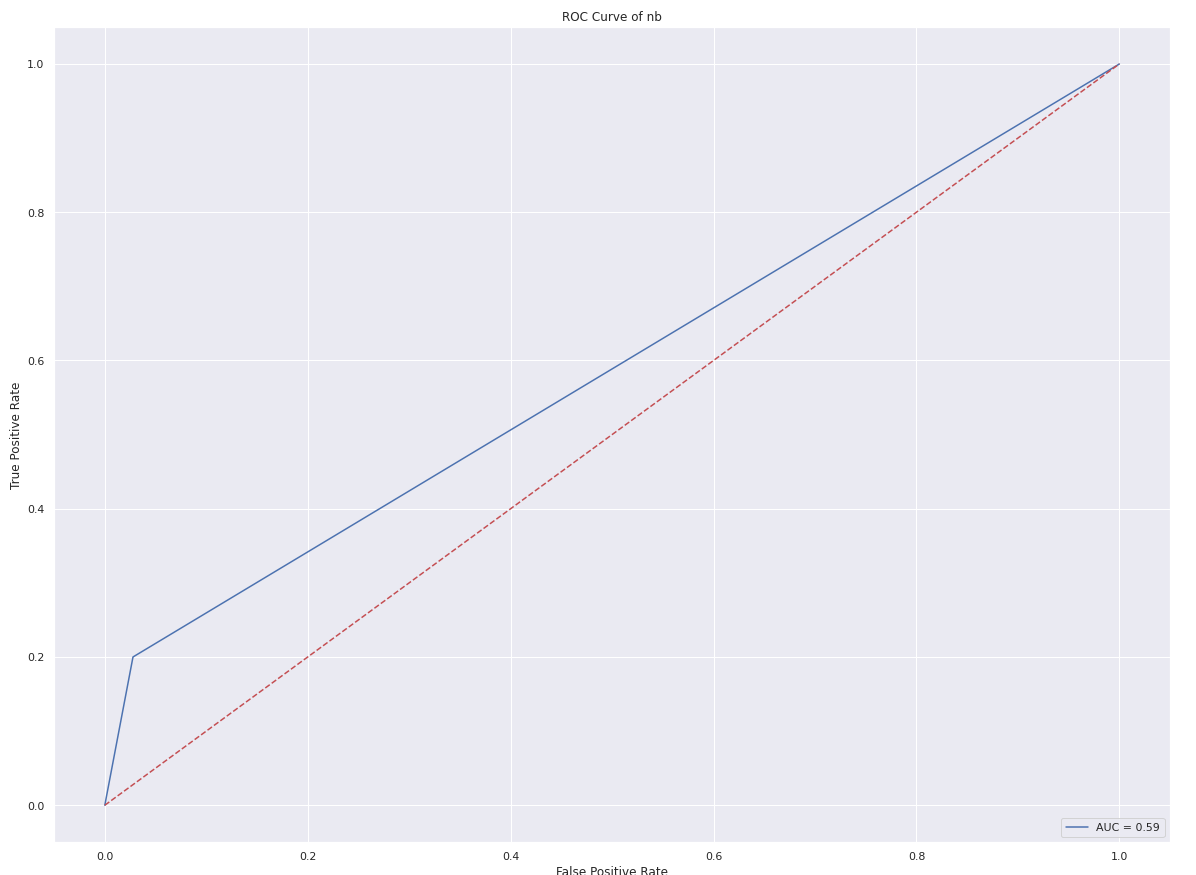
#### Create Model

We create Naive Bayes model and we used grid search technique to tune our hyper parameters after the grid search we find that the best hyper parameters are ('var\_smoothing': np.logspace(0,-9, num=300)

#### The Model Results



#### Calculate AUC & plot ROC Curve



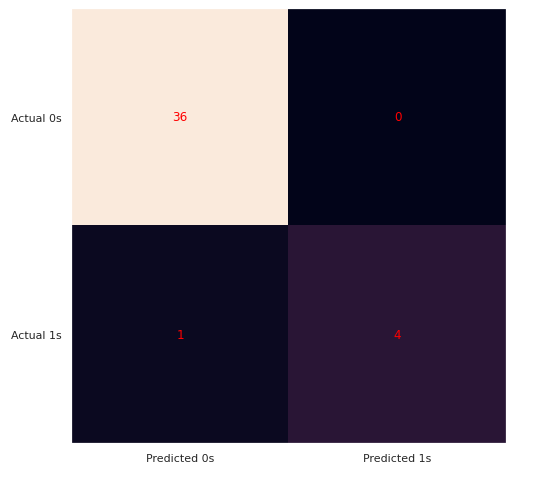
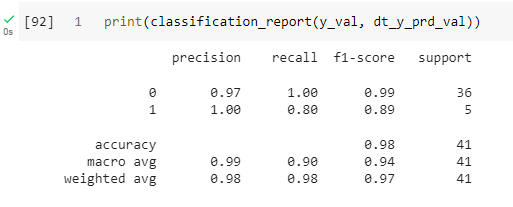
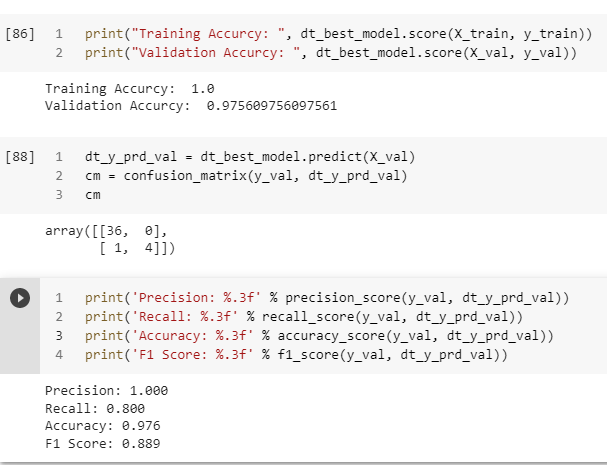
### Decision Tree Classifier

#### Create Model

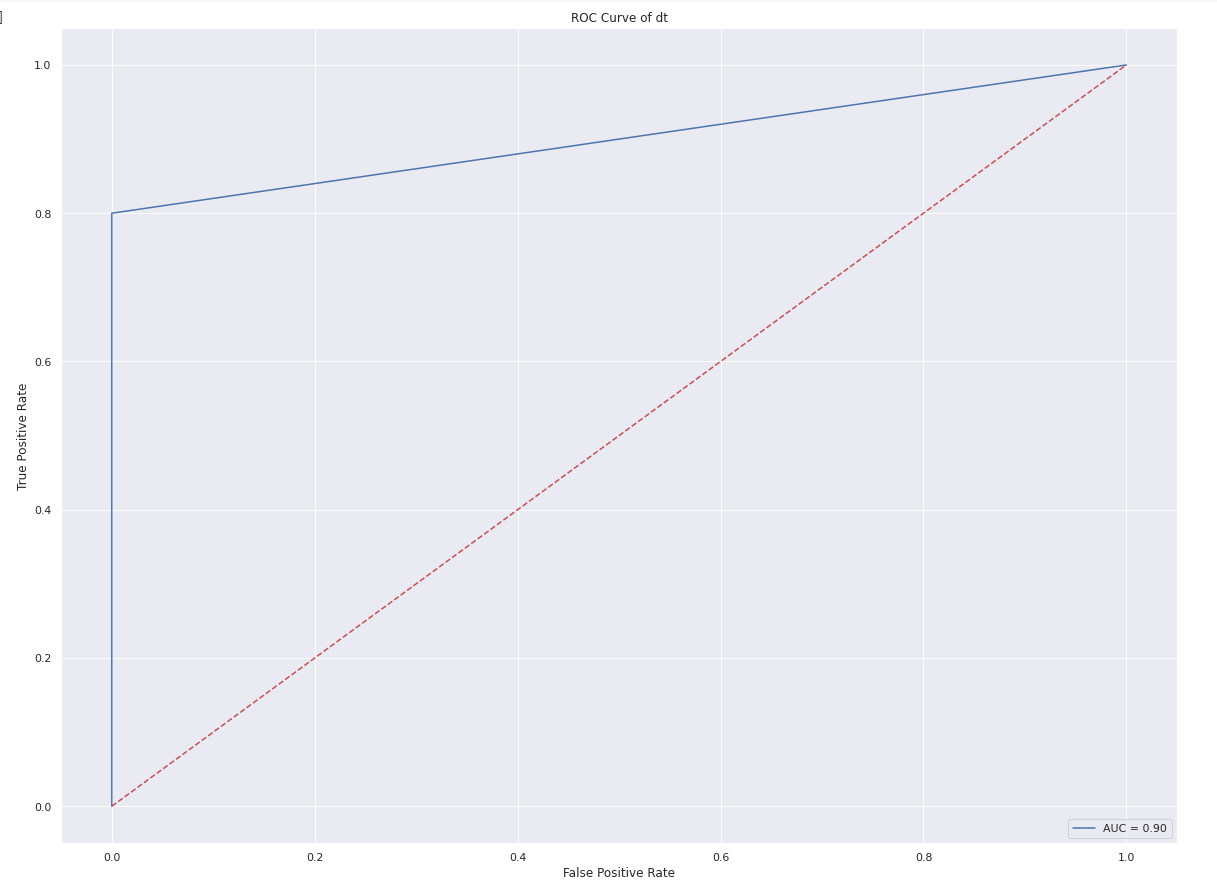
We create Decision Tree model using hyper parameters are (tree\_param = {'criterion':['gini','entropy'],'max\_depth':[4,5,6,7,8,9,10,11,12,15,20,30,40,50,70,90,120,150]}

)

#### The Model Results



#### Calculate AUC & plot ROC Curve



### Support Vector Machines

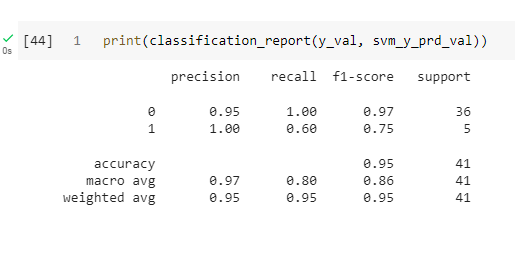
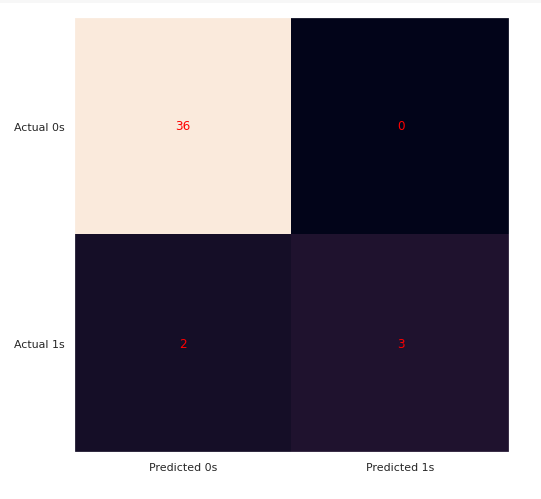
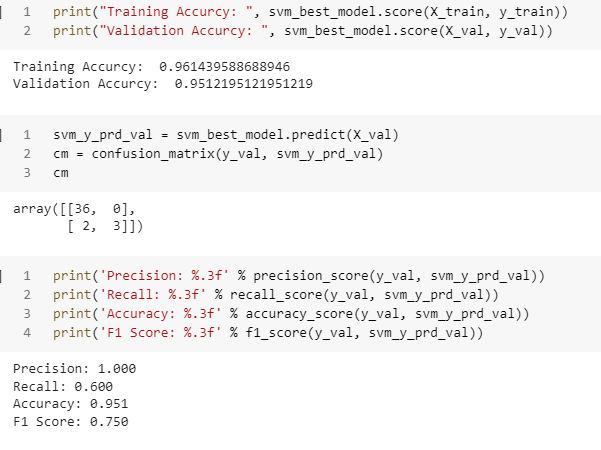
#### Create Model

We create Support vector machine model and we used grid search technique to tune our hyper parameters after the grid search we find that the best hyper parameters are param\_grid = {'C': [ 100, 1000],

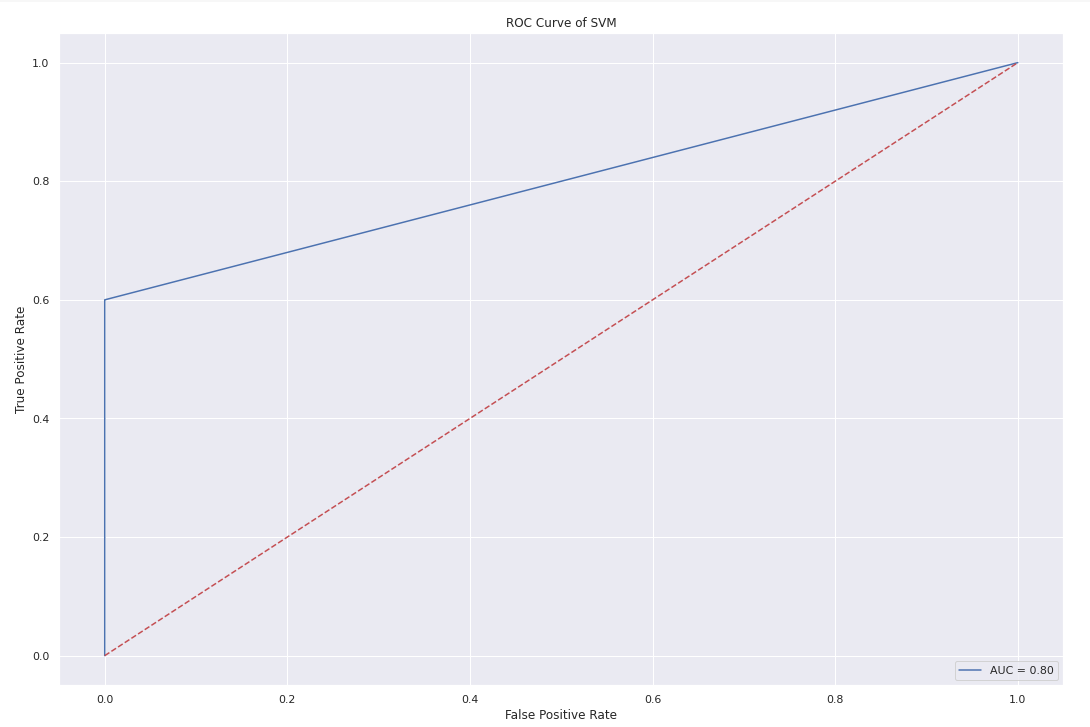
              'gamma': [1, 0.1, 0.01],

              'kernel': ['linear']}

The Model Results



#### Calculate AUC & plot ROC Curve



# **Final Result**

At the end when we compare between all the models we find that the best model for our data set is **Decision tree model** because it has the highest precision and recall accuracy. So we applied it on test dataset and got the test accuracy as shown

