CAPSTONE PROJECT
SUPERVISED MACHINE LEARNING

HEART DISEASE PREDICTION

KARAN KISHOR KARLE
BATCH :- PCAI

Contents

Introduction

- Machine learning is a subfield of Data Science,
- Machine learning is an approach to data analysis that involves building and adapting models, which allow programs to "learn" through experience
- Machine Learning has two types i.e. Supervised and Unsupervised Learning,
- Here we have to build Supervised regression model for predict the probability of a patient having Heart Disease or not
- Logistic Regression, Decision Tree , Random Forest algorithm are used for creating model
- One another model created by using Hyper parameter tuning with GridSearchCV for Random Forest model
- Final model selection is based on accuracy score all models

Objectives

To build a predictive Classification model which predict the probability of a patient having Heart Disease or not, based on the symptom's like Chest Pain type , BP , Cholesterol etc.

It will be benefiting for making decisions and precautions on right time .

Python Packages

- Pandas, Numpy, Seaborn, Matplotlib
- LabelEncoder, train_test_split, MinMaxScaler, accuracy_score

- Algorithms
    - o Logistic Regression
    - o Decision Tree
    - o Random Forest

- GridSearchCV and all other needed dependencies

The Solution Approach

270 entries

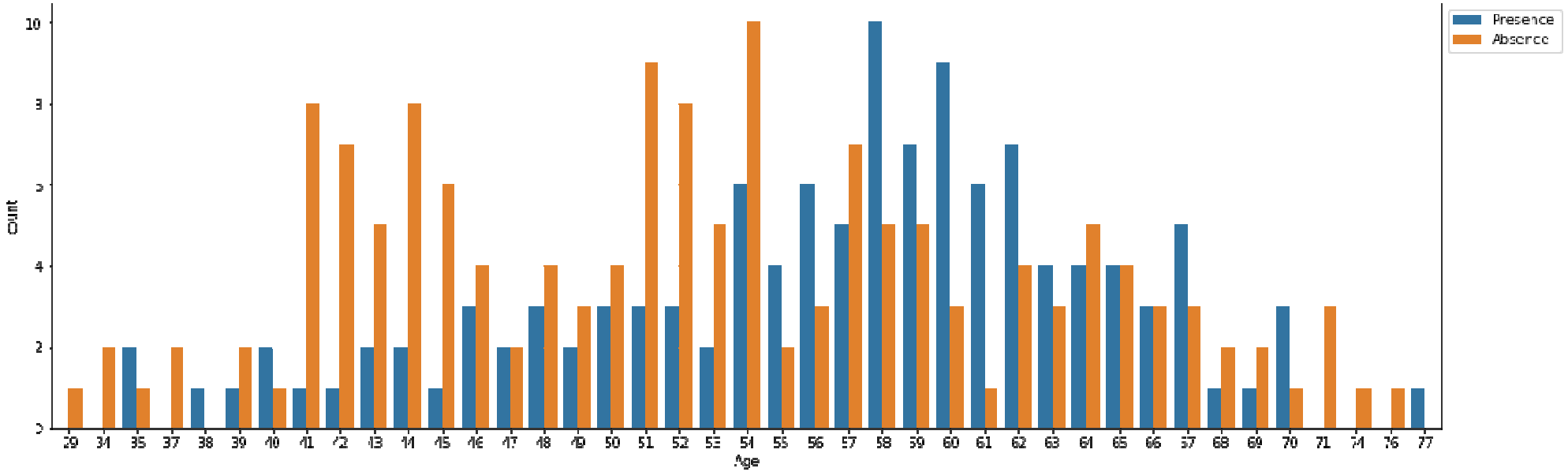
Data

```
df = pd.read_csv('Heart_Disease_Prediction.csv')
df
```

#270 entries are there in dataset

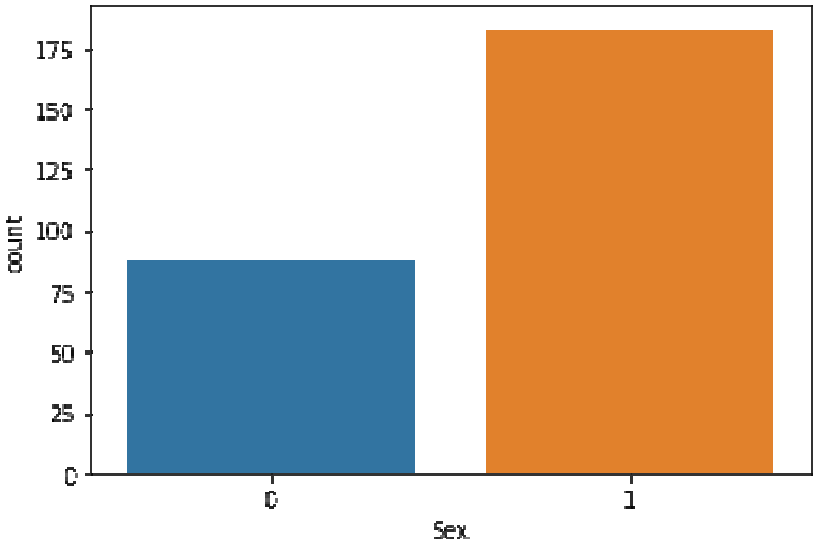
	Age	Sex	Chest pain type	BP	Cholesterol	FBS over 120	EKG results	Max HR	Exercise angina	ST depression	Slope of ST	Number of vessels fluro	Thallium	Heart Disease
0	70	1	4	130	322	0	2	109	0	2.4	2	3	3	Presence
1	67	0	3	115	564	0	2	160	0	1.6	2	0	7	Absence
2	57	1	2	124	261	0	0	141	0	0.3	1	0	7	Presence
3	64	1	4	128	263	0	0	105	1	0.2	2	1	7	Absence
4	74	0	2	120	269	0	2	121	1	0.2	1	1	3	Absence
5	65	1	4	120	177	0	0	140	0	0.4	1	0	7	Absence
6	56	1	3	130	256	1	2	142	1	0.6	2	1	6	Presence
7	59	1	4	110	239	0	2	142	1	1.2	2	1	7	Presence
8	60	1	4	140	293	0	2	170	0	1.2	2	2	7	Presence
9	63	0	4	150	407	0	2	154	0	4.0	2	3	7	Presence
10	59	1	4	135	234	0	0	161	0	0.5	2	0	7	Absence
11	53	1	4	142	226	0	2	111	1	0.0	1	0	7	Absence
12	44	1	3	140	235	0	2	180	0	0.0	1	0	3	Absence
13	61	1	1	134	234	0	0	145	0	2.6	2	2	3	Presence
14	57	0	4	128	303	0	2	159	0	0.0	1	1	3	Absence
15	71	0	4	112	149	0	0	125	0	1.6	2	0	3	Absence
16	46	1	4	140	311	0	0	120	1	1.8	2	2	7	Presence
17	53	1	4	140	203	1	2	155	1	3.1	3	0	7	Presence
18	64	1	1	110	211	0	2	144	1	1.8	2	0	3	Absence
19	40	1	1	140	199	0	0	178	1	1.4	1	0	7	Absence

Exploratory Data Analysis & Data Visualization

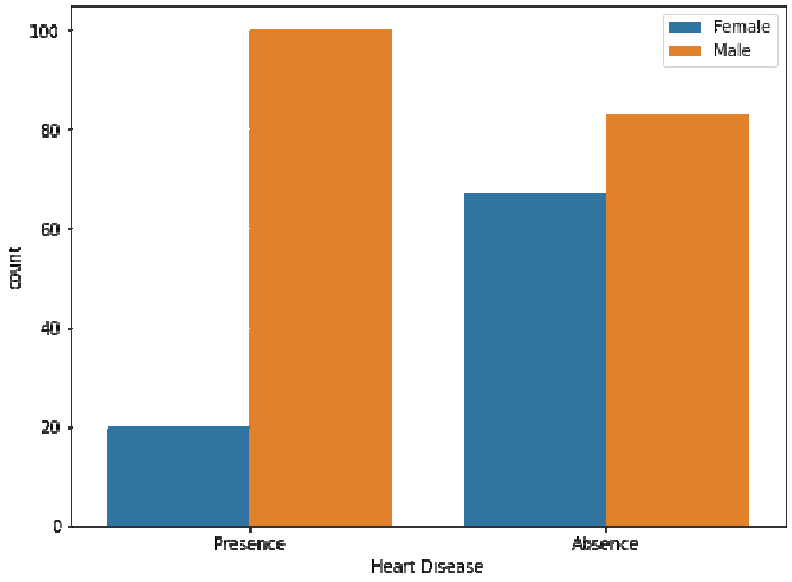


Above count plot showing Persons who are 35+ years age can be diagnosed with Heart Disease

Exploratory Data Analysis & Data Visualization

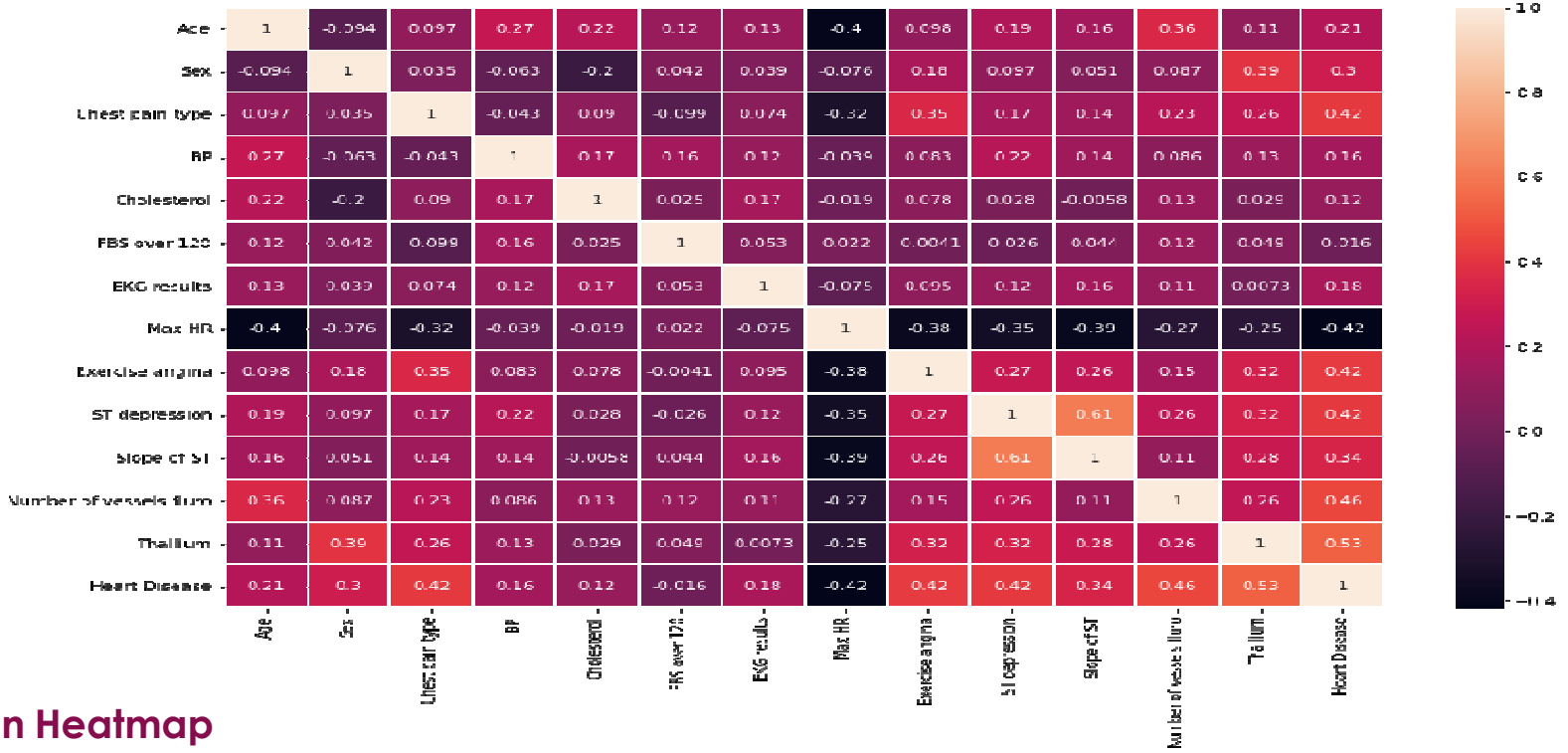


There are 87 Females and 183 Males in data



Above plot shows that Heart problems is more in Males than Females

Exploratory Data Analysis & Data Visualization



Correlation Heatmap

Model Building and Performance Evaluation

- **Data trained is by different Classification Algorithms**
  - Logistic Regression
  - Decision Tree
  - Random Forest
  - Random Forest with Hyper Parameter Tuning using GridsearchCV

1) Logistic Regression

```
1 from sklearn.linear_model import LogisticRegression
2 lr_model =LogisticRegression()
3 lr_model.fit(X_train,y_train)
```

```
LogisticRegression()
```

```
1 print("Test data accuracy is : ",lr_model.score(X_test, y_test))
2 print("Train data accuracy is : ",lr_model.score(X_train, y_train))
```

```
Test data accuracy is :  0.8271604938271605
Train data accuracy is :  0.8571428571428571
```

```
1 predicted =lr_model.predict(X_test)
```

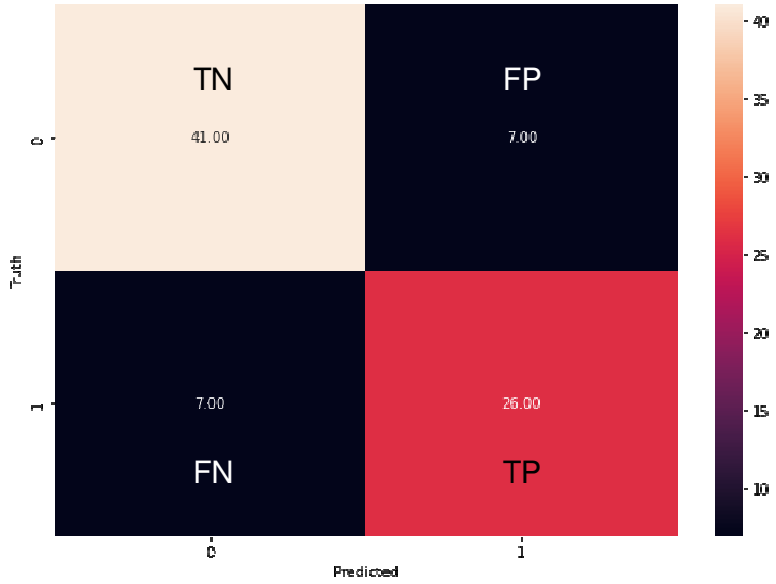
```
1 pred_prob= lr_model.predict_proba(X_test)
```

```
1 from sklearn.metrics import accuracy_score
2 logi = accuracy_score(y_test, predicted)
3 print ("Accuracy for Logistic Regression: {}".format(logi))
```

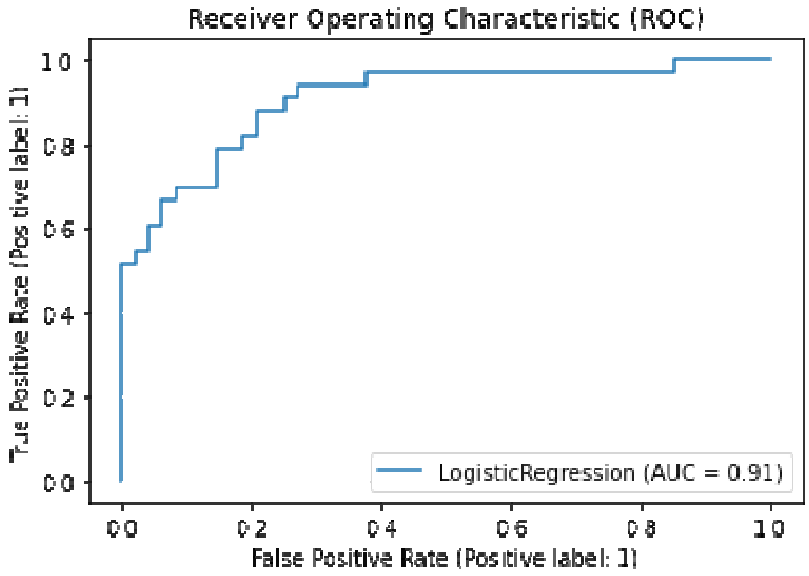
```
Accuracy for Logistic Regression: 0.8271604938271605
```

Performance Evaluation

Logistic Regression Model



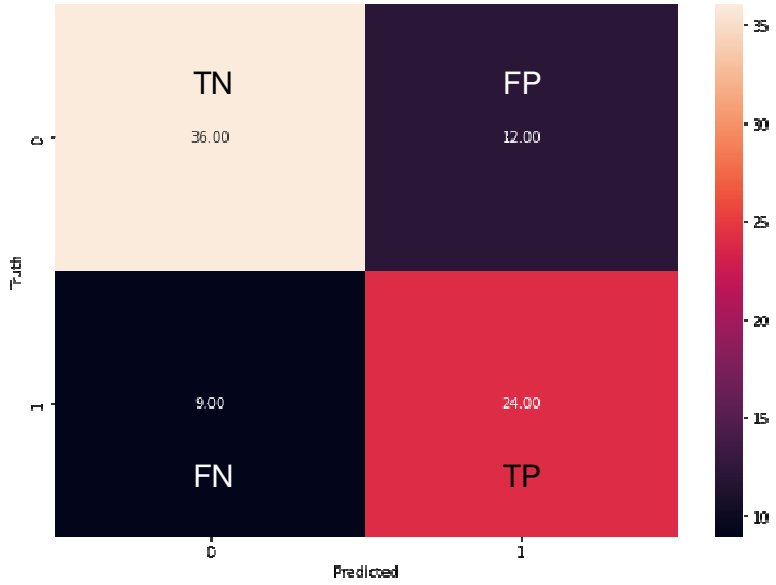
Confusion Matrix



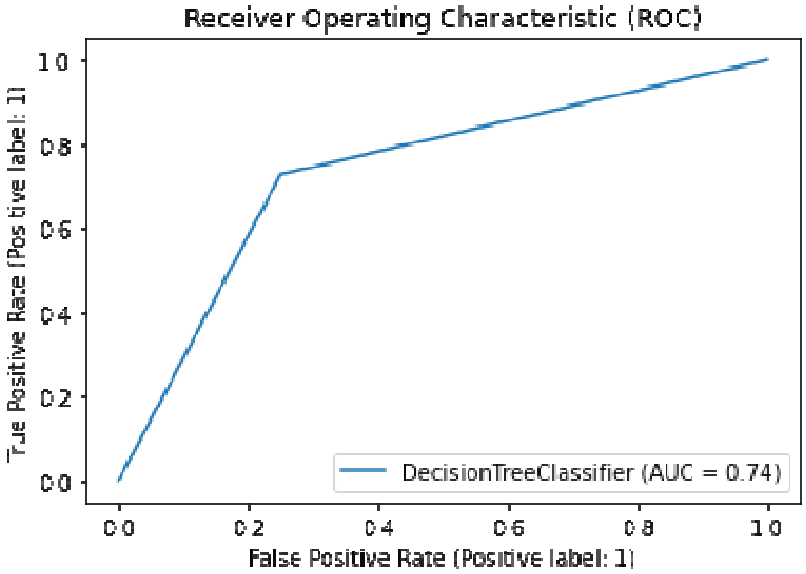
AUC – ROC Curve

Performance Evaluation

Decision Tree Model



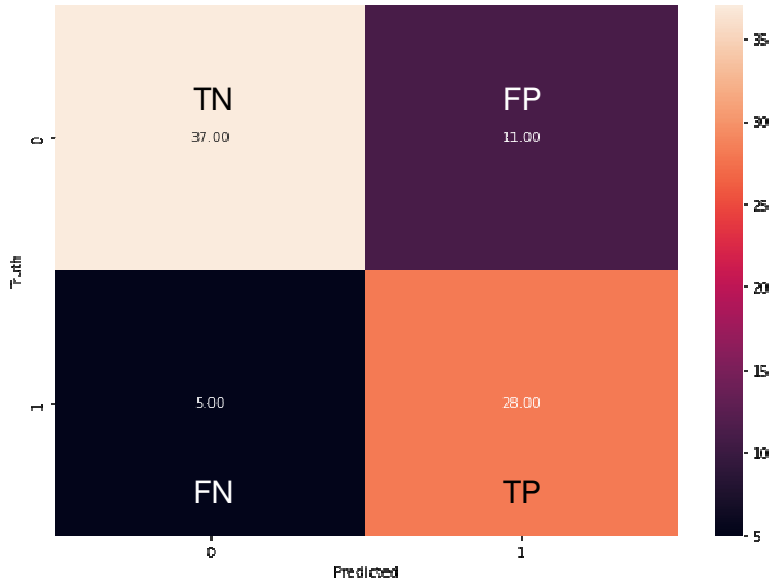
Confusion Matrix



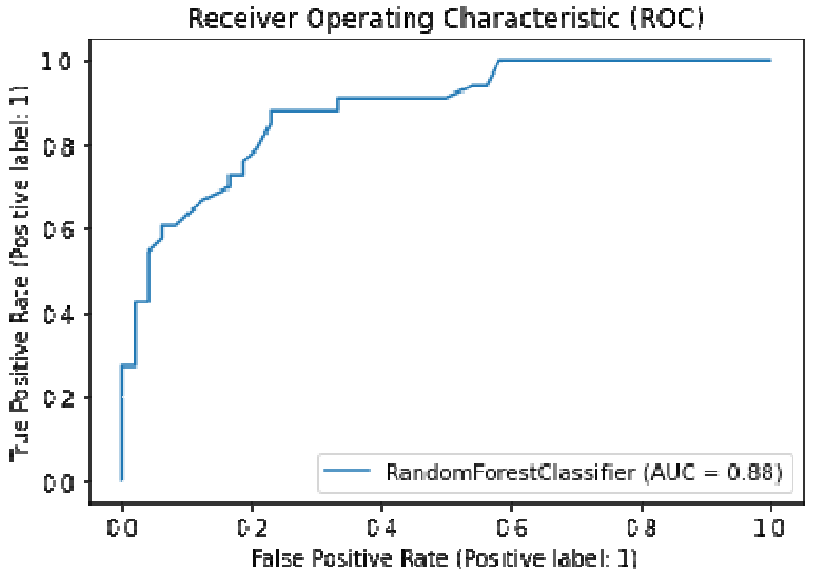
AUC – ROC Curve

Performance Evaluation

Random Forest Model



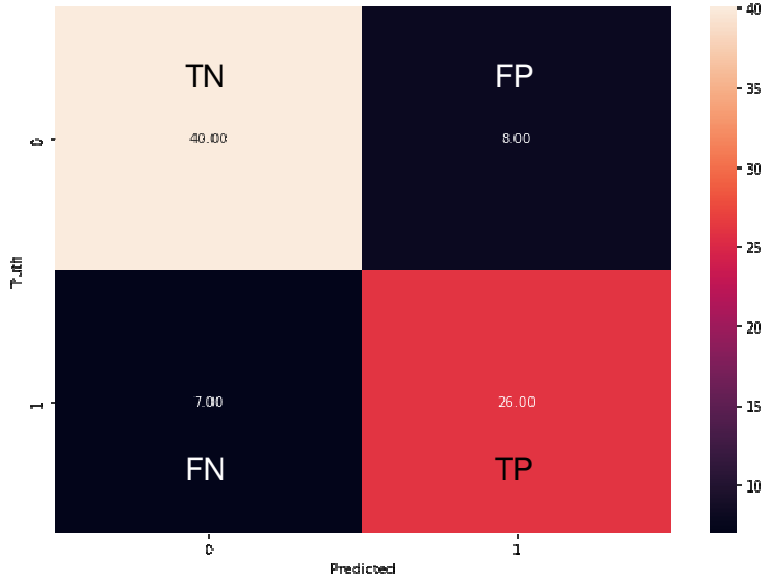
Confusion Matrix



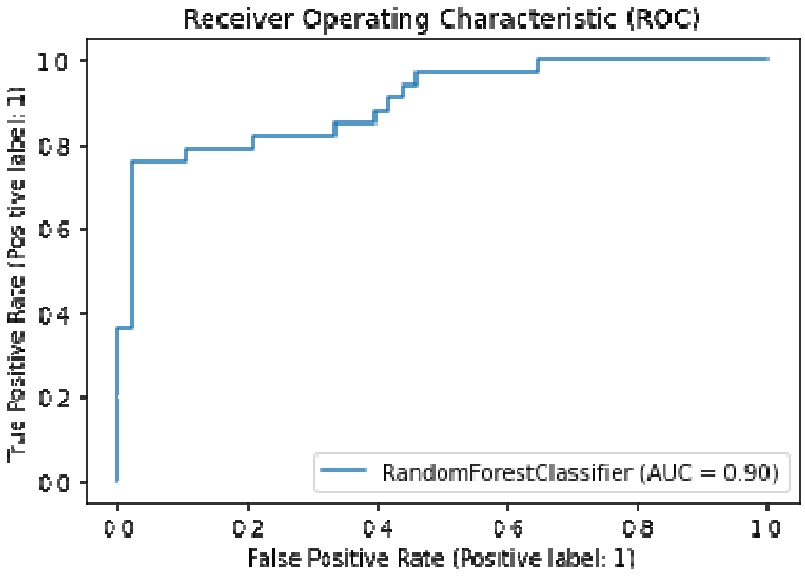
AUC – ROC Curve

Performance Evaluation

Random Forest (Hyper Parameter Tuned) Model



Confusion Matrix



AUC – ROC Curve

Model Comparison and Conclusion

Results :- Model Comparison

```
data = {'Models':['Logistic Regression',
                  'Decision Tree',
                  'Random Forest',
                  'Random Forest Tuned'],
        "Accuracy":[logi*100,
                    deci*100,
                    random*100,
                    random_grid*100]
        }

data = pd.DataFrame(data)

data.sort_values('Accuracy', ascending=False)
```

	Estimators	Accuracy
0	Logistic Regression	82.716049
3	Random Forest Tuned	81.481481
2	Random Forest	80.246914
1	Decision Tree	74.074074

Conclusion :-

Accuracy with 'Decision Tree' is Best here without having Overfitting or Underfitting Problem

Thank you