Phishing Detector using LR

The document has to specify the requirements for the project “Build a detector for Phishing websites (LR).” Apart from specifying the functional and non-functional requirements for the project, it also serves as an input for project scoping.

Importing the libraries

In [1]:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
import seaborn as sns
```

Importing the dataset phishing.txt

In [2]:

```
data = pd.read_csv('phishing.txt',header =None)
```

Data understanding and Exploration

In [3]:

```
data.head()
```

Out[3]:

	0	1	2	3	4	5	6	7	8	9	...	21	22	23	24	25	26	27	28	29	30
0	-1	1	1	1	-1	-1	-1	-1	-1	1	...	1	1	-1	-1	-1	-1	1	1	-1	-1
1	1	1	1	1	1	-1	0	1	-1	1	...	1	1	-1	-1	0	-1	1	1	1	-1
2	1	0	1	1	1	-1	-1	-1	-1	1	...	1	1	1	-1	1	-1	1	0	-1	-1
3	1	0	1	1	1	-1	-1	-1	1	1	...	1	1	-1	-1	1	-1	1	-1	1	-1
4	1	0	-1	1	1	-1	1	1	-1	1	...	-1	1	-1	-1	0	-1	1	1	1	1

5 rows × 31 columns

In [4]:

```
data.shape
```

Out[4]:

```
(11055, 31)
```

Adding columns to the dataset

In [5]:

```
data.columns = ['having_IP_Address','URL_length','Shortining_Service','having_At
_Symbol',
               'double_slash_redirecting','Prefix_suffix','having_Sub_Domain','SS
Lfinal_State','Domain_registration_length',
                'Favion','Port','HTTPS_token','Request_URL','URL_of_Anchor','Link
s_in_tags','SFH',
                'Submitting_to_email','Abnormal_URL','Redirect','on_movesover','R
ightClick','PopUpwindow','IFrame',
                 'age_of_domain','DNSRecord','web_traffic','Page_Rank','Google_In
dex',
                 'Links_pointing_to_page','Statistical_Report','Result']
```

In [6]:

```
data.head()
```

Out[6]:

	having_IP_Address	URL_length	Shortining_Service	having_At_Symbol	double_slash_redir
0	-1	1	1	1	
1	1	1	1	1	
2	1	0	1	1	
3	1	0	1	1	
4	1	0	-1	1	

5 rows × 31 columns

In [7]:

```
data.columns
```

Out[7]:

```
Index(['having_IP_Address', 'URL_length', 'Shortining_Service',
       'having_At_Symbol', 'double_slash_redirecting', 'Prefix_suffi
x',
       'having_Sub_Domain', 'SSLfinal_State', 'Domain_registration_l
ength',
       'Favion', 'Port', 'HTTPS_token', 'Request_URL', 'URL_of_Ancho
r',
       'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_UR
L',
       'Redirect', 'on_movesover', 'RightClick', 'PopUpwindow', 'IFr
ame',
       'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
       'Google_Index', 'Links_pointing_to_page', 'Statistical_Repor
t',
       'Result'],
      dtype='object')
```

In [8]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 31 columns):
having_IP_Address              11055 non-null int64
URL_length                     11055 non-null int64
Shortining_Service             11055 non-null int64
having_At_Symbol               11055 non-null int64
double_slash_redirecting       11055 non-null int64
Prefix_suffix                  11055 non-null int64
having_Sub_Domain              11055 non-null int64
SSLfinal_State                 11055 non-null int64
Domain_registration_length     11055 non-null int64
Favion                         11055 non-null int64
Port                           11055 non-null int64
HTTPS_token                    11055 non-null int64
Request_URL                    11055 non-null int64
URL_of_Anchor                  11055 non-null int64
Links_in_tags                  11055 non-null int64
SFH                            11055 non-null int64
Submitting_to_email            11055 non-null int64
Abnormal_URL                   11055 non-null int64
Redirect                       11055 non-null int64
on_movesover                   11055 non-null int64
RightClick                     11055 non-null int64
PopUpwindow                    11055 non-null int64
IFrame                         11055 non-null int64
age_of_domain                  11055 non-null int64
DNSRecord                      11055 non-null int64
web_traffic                    11055 non-null int64
Page_Rank                      11055 non-null int64
Google_Index                   11055 non-null int64
Links_pointing_to_page         11055 non-null int64
Statistical_Report             11055 non-null int64
Result                         11055 non-null int64
dtypes: int64(31)
memory usage: 2.6 MB
```

In [9]:

```
data.describe()
```

Out[9]:

	having_IP_Address	URL_length	Shortining_Service	having_At_Symbol	double_slash_r
count	11055.000000	11055.000000	11055.000000	11055.000000	110
mean	0.313795	-0.633198	0.738761	0.700588	
std	0.949534	0.766095	0.673998	0.713598	
min	-1.000000	-1.000000	-1.000000	-1.000000	
25%	-1.000000	-1.000000	1.000000	1.000000	
50%	1.000000	-1.000000	1.000000	1.000000	
75%	1.000000	-1.000000	1.000000	1.000000	
max	1.000000	1.000000	1.000000	1.000000	

8 rows × 31 columns

In [10]:

```
data.corr()
```

Out[10]:

	having_IP_Address	URL_length	Shortining_Service	having_At_Syml
having_IP_Address	1.000000	-0.052411	0.403461	0.1586
URL_length	-0.052411	1.000000	-0.097881	-0.0751
Shortining_Service	0.403461	-0.097881	1.000000	0.1044
having_At_Symbol	0.158699	-0.075108	0.104447	1.0000
double_slash_redirecting	0.397389	-0.081247	0.842796	0.0869
Prefix_suffix	-0.005257	0.055247	-0.080471	-0.0117
having_Sub_Domain	-0.080745	0.003997	-0.041916	-0.0589
SSLfinal_State	0.071414	0.048754	-0.061426	0.0312
Domain_registration_length	-0.022739	-0.221892	0.060923	0.0155
Favion	0.087025	-0.042497	0.006101	0.3048
Port	0.060979	0.000323	0.002201	0.3648
HTTPS_token	0.363534	-0.089383	0.757838	0.1045
Request_URL	0.029773	0.246348	-0.037235	0.0279
URL_of_Anchor	0.099847	-0.023396	0.000561	0.0579
Links_in_tags	0.006212	0.052869	-0.133379	-0.0708
SFH	-0.010962	0.414196	-0.022723	-0.0086
Submitting_to_email	0.077989	-0.014457	0.049328	0.3701
Abnormal_URL	0.336549	-0.106761	0.739290	0.2039
Redirect	-0.321181	0.046832	-0.534530	-0.0281
on_movesover	0.084059	-0.045103	0.062383	0.2796
RightClick	0.042881	-0.013613	0.038118	0.2195
PopUpwindow	0.096882	-0.049381	0.036616	0.2908
IFrame	0.054694	-0.013838	0.016581	0.2844
age_of_domain	-0.010446	0.179426	-0.052596	-0.0054
DNSRecord	-0.050733	-0.040823	0.436064	-0.0478
web_traffic	0.002922	0.008993	-0.047074	0.0329
Page_Rank	-0.091774	0.183518	0.014591	-0.0647
Google_Index	0.029153	0.002902	0.155844	0.0370
Links_pointing_to_page	-0.339065	-0.022987	-0.198410	-0.0060
Statistical_Report	-0.019103	-0.067153	0.085461	-0.0803
Result	0.094160	0.057430	-0.067966	0.0529

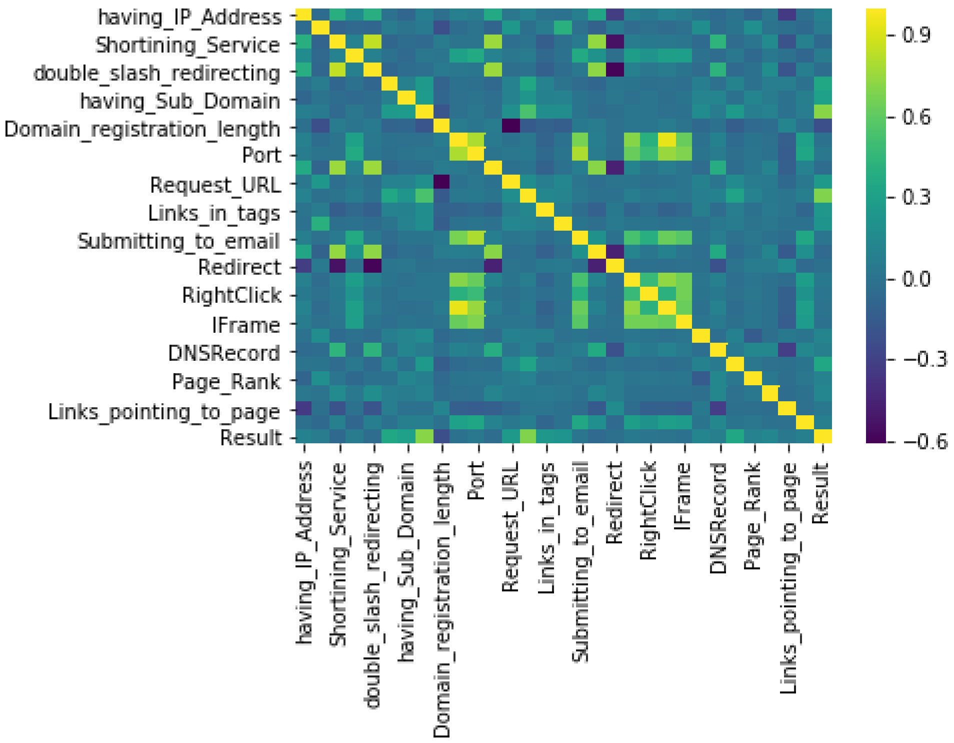
31 rows × 31 columns

In [11]:

```
sns.heatmap(data.corr(),cmap ='viridis',linecolor = 'black')
```

Out[11]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a24eda390>
```



Splitting the dataset into independent variables and dependent variables

In [12]:

```
X = data.iloc[:,0:30].values
```

In [13]:

```
X.shape
```

Out[13]:

```
(11055, 30)
```

In [14]:

```
y = data.loc[:,['Result']].values
```

In [15]:

```
y.shape
```

Out[15]:

```
(11055, 1)
```

Spitting the dataset into training and testing dataset with ratio 70:30

70% of the dataset is goes for training 30% of the dataset is goes for testing

In [16]:

```
# Importing the train_test_split
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X,y,test_size =0.30,random_stat
e =1)
```

In [17]:

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(7738, 30)
(3317, 30)
(7738, 1)
(3317, 1)
```

Logistic Regression Model

In [18]:

```
# Importing the classifier from linear model
from sklearn.linear_model import LogisticRegression
classifier = LogisticRegression()
```

In [19]:

```
classifier.get_params().keys()
```

Out[19]:

```
dict_keys(['C', 'class_weight', 'dual', 'fit_intercept', 'intercept_
scaling', 'max_iter', 'multi_class', 'n_jobs', 'penalty', 'random_st
ate', 'solver', 'tol', 'verbose', 'warm_start'])
```

In [20]:

```
# applying grid search to find best performing parameters
from sklearn.model_selection import GridSearchCV
parameters = [{'C':[0.001,0.01,0.1,1,10,100,1000],
               'penalty':['l1','l2']
              }]
grid_search = GridSearchCV(classifier,parameters,cv=5,n_jobs =-1)
grid_search.fit(X_train,y_train)
# Printing best parameters
print('Best Accuracy =',(grid_search.best_score_))
print('Best parameters = ',(grid_search.best_params_))
```

```
Best Accuracy = 0.9280175756009305
Best parameters =  {'C': 0.1, 'penalty': 'l1'}
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:7
61: DataConversionWarning: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```

In [21]:

```
# Instantiate the classifier Logistic Regression
classifier = LogisticRegression(C=0.1,penalty ='l1')
```

In [22]:

```
# Fitting the classifier or model on training dataset to train
classifier.fit(X_train,y_train)
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:7
61: DataConversionWarning: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[22]:

```
LogisticRegression(C=0.1, class_weight=None, dual=False, fit_interce
pt=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l1', random_state=None, solver='war
n',
          tol=0.0001, verbose=0, warm_start=False)
```

In [23]:

```
# Predicting the values on test dataset
y_pred = classifier.predict(X_test)
```

In [24]:

```
# Confusion matrix for the LR classifier
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[24]:

```
array([[1338,  152],
       [  91, 1736]])
```

In [25]:

```
TP = cm[0][0]
print('True Postive  = ',TP)
FP = cm[0][1]
print('False Postive = ',FP)
FN = cm[1][0]
print('False Negative = ',FN)
TN = cm[1][1]
print('True Negative = ',TN)
```

```
True Postive  =  1338
False Postive =  152
False Negative =  91
True Negative =  1736
```

In [26]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test,y_pred)
print ('The Accuracy of the LR model : ',round(accuracy*100,ndigits =2),'%')
```

```
The Accuracy of the LR model :  92.67 %
```

Random Forest Classification Model

In [89]:

```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators =700,
                                   criterion ='entropy',
                                   max_features ='sqrt',
                                   random_state=0)
```

In [90]:

```
classifier.fit(X_train,y_train)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: Data
ConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples,), for example
using ravel().
  """Entry point for launching an IPython kernel.
```

Out[90]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion
='entropy',
            max_depth=None, max_features='sqrt', max_leaf_nodes=Non
e,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=700, n_jobs=N
one,
            oob_score=False, random_state=0, verbose=0, warm_start=F
alse)
```

In [91]:

```
y_pred = classifier.predict(X_test)
```

In [92]:

```
from sklearn.metrics import confusion_matrix
confusion_matrix(y_test,y_pred)
```

Out[92]:

```
array([[ 996,  494],
       [   3, 1824]])
```

In [93]:

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test,y_pred)
accuracy
```

Out[93]:

```
0.8501658124811576
```

In [ ]:

```

```

XGBoost Model

In [32]:

```
%time
# Importing the XGBoost model from scikit learn ensemble
from xgboost import XGBClassifier
classifier = XGBClassifier(n_estimator =1000)
classifier.fit(X_train,y_train)
```

```
CPU times: user 4 µs, sys: 1 µs, total: 5 µs
Wall time: 7.87 µs

/anaconda3/lib/python3.6/site-packages/sklearn/preprocessing/label.p
y:219: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples, ), f
or example using ravel().
  y = column_or_1d(y, warn=True)
/anaconda3/lib/python3.6/site-packages/sklearn/preprocessing/label.p
y:252: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples, ), f
or example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[32]:

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_ste
p=0,
       max_depth=3, min_child_weight=1, missing=None, n_estimator=10
00,
       n_estimators=100, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=True,
       subsample=1)
```

In [33]:

```
# Predicting the values on independent variables testing dataset
y_pred = classifier.predict(X_test)
```

In [34]:

```
# Confusion matrix for evaluation to get the accuracy of the XGBoost model
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[34]:

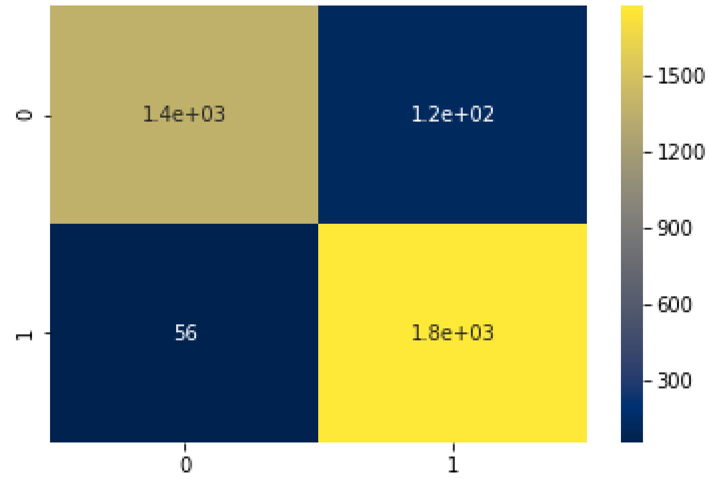
```
array([[1370,  120],
       [  56, 1771]])
```

In [35]:

```
sns.heatmap(cm,annot = True,cmap="cividis")
```

Out[35]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a27074780>
```



In [36]:

```
# Accuracy of the XGBoost model is base on Actual values and predicting values b
y the model
from sklearn.metrics import accuracy_score
model_accuracy = accuracy_score(y_test,y_pred)
model_accuracy
```

Out[36]:

```
0.9469400060295448
```

The Champion Model out of LR,RF and XGBoost is Random Forest of accuracy 96%

Exercies 2

Train with only two input parameters - parameter Prefix_Suffix and 13 URL_of_Anchor. Check accuracy using the test data and compare the accuracy with the previous value.

In [37]:

```
data.head()
```

Out[37]:

	having_IP_Address	URL_length	Shortining_Service	having_At_Symbol	double_slash_redirect
0	-1	1	1	1	
1	1	1	1	1	
2	1	0	1	1	
3	1	0	1	1	
4	1	0	-1	1	

5 rows × 31 columns

In [38]:

```
data.columns
```

Out[38]:

```
Index(['having_IP_Address', 'URL_length', 'Shortining_Service',
       'having_At_Symbol', 'double_slash_redirecting', 'Prefix_suffi
x',
       'having_Sub_Domain', 'SSLfinal_State', 'Domain_registration_l
ength',
       'Favion', 'Port', 'HTTPS_token', 'Request_URL', 'URL_of_Ancho
r',
       'Links_in_tags', 'SFH', 'Submitting_to_email', 'Abnormal_UR
L',
       'Redirect', 'on_movesover', 'RightClick', 'PopUpwindow', 'IFr
ame',
       'age_of_domain', 'DNSRecord', 'web_traffic', 'Page_Rank',
       'Google_Index', 'Links_pointing_to_page', 'Statistical_Repor
t',
       'Result'],
      dtype='object')
```

In [39]:

```
X1 = data.loc[:,['Prefix_suffix','URL_of_Anchor']].values
```

In [40]:

```
y1 = data.loc[:,['Result']].values
```

In [41]:

```
X = pd.DataFrame(X1)
```

In [42]:

```
y = pd.DataFrame(y)
```

Data understanding and Exploration

In [43]:

```
X.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 11055 entries, 0 to 11054
Data columns (total 2 columns):
0    11055 non-null int64
1    11055 non-null int64
dtypes: int64(2)
memory usage: 172.8 KB
```

In [44]:

```
X.describe().transpose()
```

Out[44]:

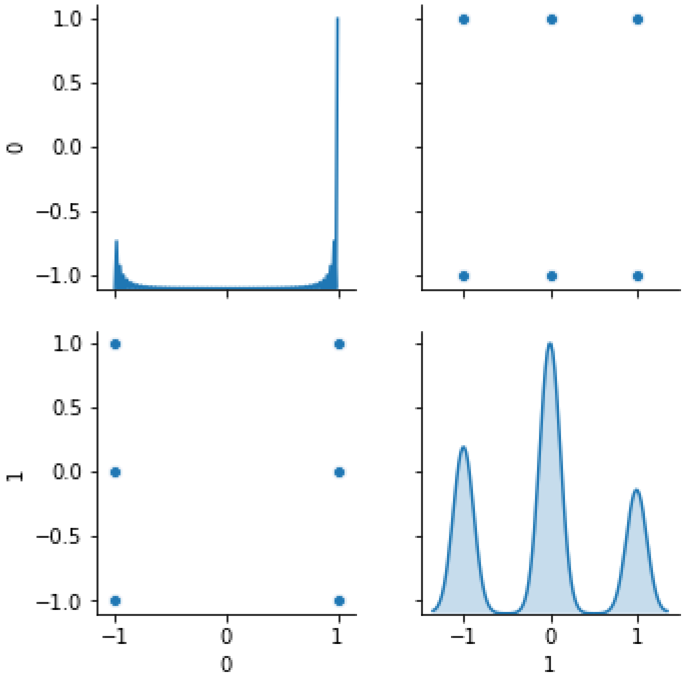
	count	mean	std	min	25%	50%	75%	max
0	11055.0	-0.734962	0.678139	-1.0	-1.0	-1.0	-1.0	1.0
1	11055.0	-0.076526	0.715138	-1.0	-1.0	0.0	0.0	1.0

In [45]:

```
sns.pairplot(X,diag_kind = 'kde')
```

Out[45]:

```
<seaborn.axisgrid.PairGrid at 0x1a27b662e8>
```

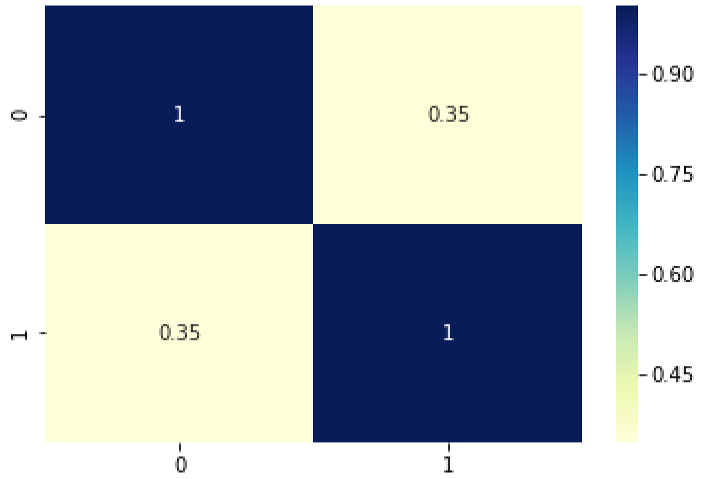


In [46]:

```
sns.heatmap(X.corr(),cmap = 'YlGnBu',annot =True)
```

Out[46]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x1a27e20390>
```



In [ ]:

```

```

In [47]:

```
y1 = y.values
```

Spitting the dataset into training and testing dataset with ratio 70:30

In [48]:

```
# Importing the train_test_split
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X1,y1,test_size =0.30,random_st
ate =1)
```

In [49]:

```
print(X_train.shape)
print(X_test.shape)
print(y_train.shape)
print(y_test.shape)
```

```
(7738, 2)
(3317, 2)
(7738, 1)
(3317, 1)
```

Logistic Regression Model

In [66]:

```
# Importing the classifier from linear model
from sklearn.linear_model import LogisticRegression
```

In [67]:

```
# Instantiate the classifier Logistic Regression
classifier = LogisticRegression()
classifier.fit(X_train,y_train)
```

```
/anaconda3/lib/python3.6/site-packages/sklearn/utils/validation.py:7
61: DataConversionWarning: A column-vector y was passed when a 1d ar
ray was expected. Please change the shape of y to (n_samples, ), for
example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[67]:

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_interce
pt=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='war
n',
          tol=0.0001, verbose=0, warm_start=False)
```

In [68]:

```
# Predicting the values on independent variables testing dataset
y_pred = classifier.predict(X_test)
```

In [69]:

```
# Confusion matrix for evaluation to get the accuracy of the XGBoost model
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[69]:

```
array([[ 996,  494],
       [   3, 1824]])
```

In [70]:

```
TP = cm[0][0]
print('True Postive  = ',TP)
FP = cm[0][1]
print('False Postive = ',FP)
FN = cm[1][0]
print('False Negative = ',FN)
TN = cm[1][1]
print('True Negative = ',TN)
```

```
True Postive  =  996
False Postive =  494
False Negative =  3
True Negative =  1824
```

In [73]:

```
# Accuracy of the XGBoost model is base on Actual values and predicting values b
y the model
from sklearn.metrics import accuracy_score
model_accuracy = accuracy_score(y_test,y_pred)
model_accuracy
```

Out[73]:

```
0.8501658124811576
```

Random Forest Classification Model

In [72]:

```
from sklearn.ensemble import RandomForestClassifier
classifier = RandomForestClassifier(n_estimators =700,
                                   criterion ='entropy',
                                   max_features ='sqrt',
                                   random_state=0)
```

In [74]:

```
classifier.fit(X_train,y_train)
```

```
/anaconda3/lib/python3.6/site-packages/ipykernel_launcher.py:1: Data
ConversionWarning: A column-vector y was passed when a 1d array was
expected. Please change the shape of y to (n_samples,), for example
using ravel().
  """Entry point for launching an IPython kernel.
```

Out[74]:

```
RandomForestClassifier(bootstrap=True, class_weight=None, criterion
='entropy',
            max_depth=None, max_features='sqrt', max_leaf_nodes=Non
e,
            min_impurity_decrease=0.0, min_impurity_split=None,
            min_samples_leaf=1, min_samples_split=2,
            min_weight_fraction_leaf=0.0, n_estimators=700, n_jobs=N
one,
            oob_score=False, random_state=0, verbose=0, warm_start=F
alse)
```

In [75]:

```
# Predicting the values on independent variables testing dataset
y_pred = classifier.predict(X_test)
```

In [76]:

```
# Confusion matrix for evaluation to get the accuracy of the XGBoost model
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[76]:

```
array([[ 996,  494],
       [   3, 1824]])
```

In [77]:

```
TP = cm[0][0]
print('True Postive  = ',TP)
FP = cm[0][1]
print('False Postive = ',FP)
FN = cm[1][0]
print('False Negative = ',FN)
TN = cm[1][1]
print('True Negative = ',TN)
```

```
True Postive  =  996
False Postive =  494
False Negative =  3
True Negative =  1824
```

In [94]:

```
# Accuracy of the XGBoost model is base on Actual values and predicting values b
y the model
from sklearn.metrics import accuracy_score
model_accuracy = accuracy_score(y_test,y_pred)
model_accuracy
```

Out[94]:

```
0.8501658124811576
```

XGBoost Model

In [79]:

```
%time
# Importing the XGBoost model from scikit learn ensemble
from xgboost import XGBClassifier
classifier = XGBClassifier(n_estimator =1000)
classifier.fit(X_train,y_train)
```

```
CPU times: user 4 µs, sys: 1 µs, total: 5 µs
Wall time: 10 µs

/anaconda3/lib/python3.6/site-packages/sklearn/preprocessing/label.p
y:219: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples, ), f
or example using ravel().
  y = column_or_1d(y, warn=True)
/anaconda3/lib/python3.6/site-packages/sklearn/preprocessing/label.p
y:252: DataConversionWarning: A column-vector y was passed when a 1d
array was expected. Please change the shape of y to (n_samples, ), f
or example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[79]:

```
XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
       colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_ste
p=0,
       max_depth=3, min_child_weight=1, missing=None, n_estimator=10
00,
       n_estimators=100, n_jobs=1, nthread=None,
       objective='binary:logistic', random_state=0, reg_alpha=0,
       reg_lambda=1, scale_pos_weight=1, seed=None, silent=True,
       subsample=1)
```

In [80]:

```
# Predicting the values on independent variables testing dataset
y_pred = classifier.predict(X_test)
```

In [81]:

```
# Confusion matrix for evaluation to get the accuracy of the XGBoost model
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[81]:

```
array([[ 996,  494],
       [   3, 1824]])
```

In [96]:

```
# Accuracy of the XGBoost model is base on Actual values and predicting values by the model
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test,y_pred)
accuracy
```

Out[96]:

```
0.8501658124811576
```

In [ ]:

In [ ]:

In [ ]: