

Assignment-06-Logistic Regression

In [1]:

```
# Importing the Libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix
from sklearn.metrics import roc_curve
from sklearn.metrics import roc_auc_score
```

In [2]:

```
# Importing the dataset
bank=pd.read_csv("C:/Users/LENOVO/Documents/Custom Office Templates/bank-full.csv",delimiter=';')
bank
```

Out[2]:

	age	job	marital	education	default	balance	housing	loan	contact	day
0	58	management	married	tertiary	no	2143	yes	no	unknown	5
1	44	technician	single	secondary	no	29	yes	no	unknown	5
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5
4	33	unknown	single	unknown	no	1	no	no	unknown	5
...
45206	51	technician	married	tertiary	no	825	no	no	cellular	17
45207	71	retired	divorced	primary	no	1729	no	no	cellular	17
45208	72	retired	married	secondary	no	5715	no	no	cellular	17
45209	57	blue-collar	married	secondary	no	668	no	no	telephone	17
45210	37	entrepreneur	married	secondary	no	2971	no	no	cellular	17

45211 rows × 11 columns



In [3]:

EDA

bank.info()

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 17 columns):
 #   Column      Non-Null Count  Dtype
---  -
 0   age         45211 non-null  int64
 1   job         45211 non-null  object
 2   marital     45211 non-null  object
 3   education   45211 non-null  object
 4   default     45211 non-null  object
 5   balance     45211 non-null  int64
 6   housing     45211 non-null  object
 7   loan        45211 non-null  object
 8   contact     45211 non-null  object
 9   day         45211 non-null  int64
10  month       45211 non-null  object
11  duration    45211 non-null  int64
12  campaign    45211 non-null  int64
13  pdays      45211 non-null  int64
14  previous    45211 non-null  int64
15  poutcome    45211 non-null  object
16  y           45211 non-null  object
dtypes: int64(7), object(10)
memory usage: 5.9+ MB
```

In [4]:

One-Hot Encoding of categorical variables

```
data1=pd.get_dummies(bank,columns=['job','marital','education','contact','poutcome','month'])
data1
```

Out[4]:

	age	default	balance	housing	loan	day	duration	campaign	pdays	previous	...	m
0	58	no	2143	yes	no	5	261	1	-1	0	...	
1	44	no	29	yes	no	5	151	1	-1	0	...	
2	33	no	2	yes	yes	5	76	1	-1	0	...	
3	47	no	1506	yes	no	5	92	1	-1	0	...	
4	33	no	1	no	no	5	198	1	-1	0	...	
...	
45206	51	no	825	no	no	17	977	3	-1	0	...	
45207	71	no	1729	no	no	17	456	2	-1	0	...	
45208	72	no	5715	no	no	17	1127	5	184	3	...	
45209	57	no	668	no	no	17	508	4	-1	0	...	
45210	37	no	2971	no	no	17	361	2	188	11	...	

45211 rows × 49 columns

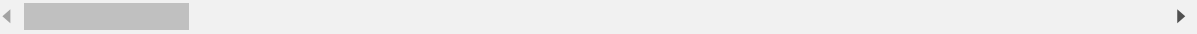
In [5]:

```
# To see all columns
pd.set_option("display.max.columns",None)
data1
```

Out[5]:

	age	default	balance	housing	loan	day	duration	campaign	pdays	previous	y	j
0	58	no	2143	yes	no	5	261	1	-1	0	no	
1	44	no	29	yes	no	5	151	1	-1	0	no	
2	33	no	2	yes	yes	5	76	1	-1	0	no	
3	47	no	1506	yes	no	5	92	1	-1	0	no	
4	33	no	1	no	no	5	198	1	-1	0	no	
...
45206	51	no	825	no	no	17	977	3	-1	0	yes	
45207	71	no	1729	no	no	17	456	2	-1	0	yes	
45208	72	no	5715	no	no	17	1127	5	184	3	yes	
45209	57	no	668	no	no	17	508	4	-1	0	no	
45210	37	no	2971	no	no	17	361	2	188	11	no	

45211 rows × 49 columns



In [6]:

data1.info()

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 49 columns):
 #   Column                Non-Null Count  Dtype
---  -
 0   age                   45211 non-null  int64
 1   default               45211 non-null  object
 2   balance               45211 non-null  int64
 3   housing               45211 non-null  object
 4   loan                  45211 non-null  object
 5   day                   45211 non-null  int64
 6   duration              45211 non-null  int64
 7   campaign              45211 non-null  int64
 8   pdays                45211 non-null  int64
 9   previous              45211 non-null  int64
10   y                     45211 non-null  object
11   job_admin.            45211 non-null  uint8
12   job_blue-collar       45211 non-null  uint8
13   job_entrepreneur      45211 non-null  uint8
14   job_housemaid         45211 non-null  uint8
15   job_management        45211 non-null  uint8
16   job_retired           45211 non-null  uint8
17   job_self-employed     45211 non-null  uint8
18   job_services          45211 non-null  uint8
19   job_student           45211 non-null  uint8
20   job_technician        45211 non-null  uint8
21   job_unemployed        45211 non-null  uint8
22   job_unknown           45211 non-null  uint8
23   marital_divorced      45211 non-null  uint8
24   marital_married       45211 non-null  uint8
25   marital_single        45211 non-null  uint8
26   education_primary     45211 non-null  uint8
27   education_secondary   45211 non-null  uint8
28   education_tertiary    45211 non-null  uint8
29   education_unknown     45211 non-null  uint8
30   contact_cellular      45211 non-null  uint8
31   contact_telephone     45211 non-null  uint8
32   contact_unknown       45211 non-null  uint8
33   poutcome_failure      45211 non-null  uint8
34   poutcome_other        45211 non-null  uint8
35   poutcome_success      45211 non-null  uint8
36   poutcome_unknown     45211 non-null  uint8
37   month_apr             45211 non-null  uint8
38   month_aug             45211 non-null  uint8
39   month_dec             45211 non-null  uint8
40   month_feb             45211 non-null  uint8
41   month_jan             45211 non-null  uint8
42   month_jul             45211 non-null  uint8
43   month_jun             45211 non-null  uint8
44   month_mar             45211 non-null  uint8
45   month_may             45211 non-null  uint8
46   month_nov             45211 non-null  uint8
47   month_oct             45211 non-null  uint8
48   month_sep             45211 non-null  uint8
dtypes: int64(7), object(4), uint8(38)
memory usage: 5.4+ MB

```

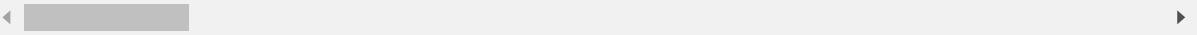
In [7]:

```
# Custom Binary Encoding of Binary o/p variables
data1['default'] = np.where(data1['default'].str.contains("yes"), 1, 0)
data1['housing'] = np.where(data1['housing'].str.contains("yes"), 1, 0)
data1['loan'] = np.where(data1['loan'].str.contains("yes"), 1, 0)
data1['y'] = np.where(data1['y'].str.contains("yes"), 1, 0)
data1
```

Out[7]:

	age	default	balance	housing	loan	day	duration	campaign	pdays	previous	y	join
0	58	0	2143	1	0	5	261	1	-1	0	0	
1	44	0	29	1	0	5	151	1	-1	0	0	
2	33	0	2	1	1	5	76	1	-1	0	0	
3	47	0	1506	1	0	5	92	1	-1	0	0	
4	33	0	1	0	0	5	198	1	-1	0	0	
...
45206	51	0	825	0	0	17	977	3	-1	0	1	
45207	71	0	1729	0	0	17	456	2	-1	0	1	
45208	72	0	5715	0	0	17	1127	5	184	3	1	
45209	57	0	668	0	0	17	508	4	-1	0	0	
45210	37	0	2971	0	0	17	361	2	188	11	0	

45211 rows × 49 columns



In [8]:

data1.info()

```

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45211 entries, 0 to 45210
Data columns (total 49 columns):
 #   Column                                Non-Null Count  Dtype
---  -
 0   age                                  45211 non-null  int64
 1   default                             45211 non-null  int32
 2   balance                             45211 non-null  int64
 3   housing                             45211 non-null  int32
 4   loan                                45211 non-null  int32
 5   day                                  45211 non-null  int64
 6   duration                             45211 non-null  int64
 7   campaign                             45211 non-null  int64
 8   pdays                              45211 non-null  int64
 9   previous                             45211 non-null  int64
10   y                                    45211 non-null  int32
11   job_admin.                           45211 non-null  uint8
12   job_blue-collar                      45211 non-null  uint8
13   job_entrepreneur                     45211 non-null  uint8
14   job_housemaid                       45211 non-null  uint8
15   job_management                       45211 non-null  uint8
16   job_retired                          45211 non-null  uint8
17   job_self-employed                    45211 non-null  uint8
18   job_services                         45211 non-null  uint8
19   job_student                          45211 non-null  uint8
20   job_technician                       45211 non-null  uint8
21   job_unemployed                       45211 non-null  uint8
22   job_unknown                          45211 non-null  uint8
23   marital_divorced                     45211 non-null  uint8
24   marital_married                      45211 non-null  uint8
25   marital_single                       45211 non-null  uint8
26   education_primary                    45211 non-null  uint8
27   education_secondary                  45211 non-null  uint8
28   education_tertiary                   45211 non-null  uint8
29   education_unknown                    45211 non-null  uint8
30   contact_cellular                     45211 non-null  uint8
31   contact_telephone                    45211 non-null  uint8
32   contact_unknown                      45211 non-null  uint8
33   poutcome_failure                     45211 non-null  uint8
34   poutcome_other                       45211 non-null  uint8
35   poutcome_success                     45211 non-null  uint8
36   poutcome_unknown                     45211 non-null  uint8
37   month_apr                            45211 non-null  uint8
38   month_aug                            45211 non-null  uint8
39   month_dec                            45211 non-null  uint8
40   month_feb                            45211 non-null  uint8
41   month_jan                            45211 non-null  uint8
42   month_jul                            45211 non-null  uint8
43   month_jun                            45211 non-null  uint8
44   month_mar                            45211 non-null  uint8
45   month_may                            45211 non-null  uint8
46   month_nov                            45211 non-null  uint8
47   month_oct                            45211 non-null  uint8
48   month_sep                            45211 non-null  uint8
dtypes: int32(4), int64(7), uint8(38)
memory usage: 4.7 MB

```

Model Building

In [9]:

```
# Dividing our data into input and output variables
x=pd.concat([data1.iloc[:,0:10],data1.iloc[:,11:]],axis=1)
y=data1.iloc[:,10]
```

In [10]:

```
# Logistic regression model
classifier=LogisticRegression()
classifier.fit(x,y)
```

C:\Users\LENOVO\anaconda3\lib\site-packages\sklearn\linear_model_logistic.p
y:763: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
<https://scikit-learn.org/stable/modules/preprocessing.html> (<https://scikit-learn.org/stable/modules/preprocessing.html>)
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression (https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression)
n_iter_i = _check_optimize_result(

Out[10]:

LogisticRegression()

Model Predictions

In [11]:

```
# Predict for x dataset
y_pred=classifier.predict(x)
y_pred
```

Out[11]:

array([0, 0, 0, ..., 1, 0, 0])

In [12]:

```
y_pred_df=pd.DataFrame({'actual_y':y,'y_pred_prob':y_pred})
y_pred_df
```

Out[12]:

	actual_y	y_pred_prob
0	0	0
1	0	0
2	0	0
3	0	0
4	0	0
...
45206	1	1
45207	1	0
45208	1	1
45209	0	0
45210	0	0

45211 rows × 2 columns

Testing Model Accuracy

In [13]:

```
# Confusion Matrix for the model accuracy
confusion_matrix = confusion_matrix(y,y_pred)
confusion_matrix
```

Out[13]:

```
array([[39155,  767],
       [ 4127, 1162]], dtype=int64)
```

In [14]:

```
# The model accuracy is calculated by (a+d)/(a+b+c+d)
(39155+1162)/(39155+767+4127+1162)
```

Out[14]:

0.8917520072548716

Accuracy of logistic regression classifier on test set:0.89

In [15]:

```
# As accuracy = 0.8933, which is greater than 0.5; Thus [:,1] Thresold value>0.5=1 else[:,0]
classifier.predict_proba(x)[:,1]
```

Out[15]:

```
array([0.04407932, 0.02469445, 0.01819442, ..., 0.66973993, 0.07913296,
       0.10132428])
```

In [16]:

```
from sklearn.metrics import classification_report
print(classification_report(y,y_pred))
```

	precision	recall	f1-score	support
0	0.90	0.98	0.94	39922
1	0.60	0.22	0.32	5289
accuracy			0.89	45211
macro avg	0.75	0.60	0.63	45211
weighted avg	0.87	0.89	0.87	45211

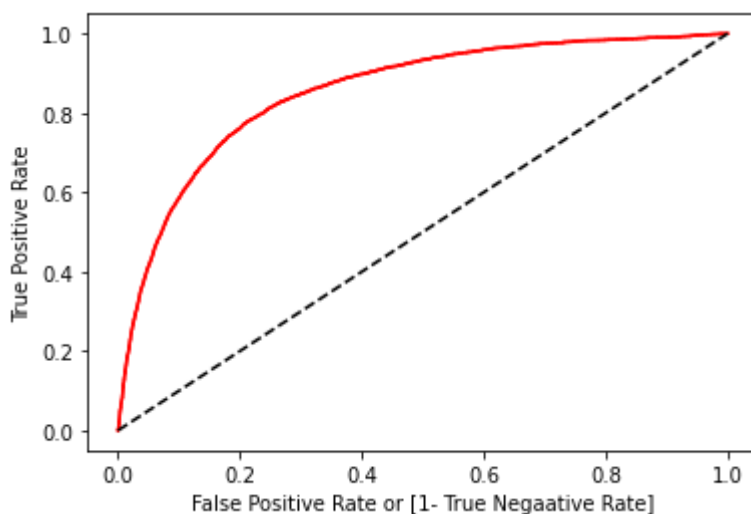
In [17]:

```
# ROC Curve plotting and finding AUC value
fpr,tpr,thresold=roc_curve(y,classifier.predict_proba(x)[:,1])
plt.plot(fpr,tpr,color='red')
auc=roc_auc_score(y,y_pred)

plt.plot(fpr,tpr,color='red',label='logit model(area = %0.2f)'%auc)

plt.plot([0,1],[0,1], 'k--')
plt.xlabel('False Positive Rate or [1- True Negaative Rate]')
plt.ylabel('True Positive Rate')
plt.show()

print('auc accuracy:',auc)
```



```
auc accuracy: 0.6002444012373571
```

In [18]:

```
auc
```

Out[18]:

```
0.6002444012373571
```

In []: