181B069 BALAJI G

In [1]:

```
import numpy as np
import pandas as pd
```

In [2]:

```
data=pd.read_csv('data.csv')
```

In [3]:

```
data.head()
```

Out[3]:

	y_act	y_pred_random_forest	y_pred_logistic
0	1	0.639816	0.531904
1	0	0.490993	0.414496
2	1	0.623815	0.569883
3	1	0.506616	0.443674
4	0	0.418302	0.369532

In [5]:

```
#rounding the numbers
data['y_pred_random_forest']=data['y_pred_random_forest'].apply(lambda x:round(x))
data['y_pred_logistic']=data['y_pred_logistic'].apply(lambda x:round(x))
```

In [9]:

```
def confusion_matrix(actual,prediction):
    matrix = np.zeros([2,2])

    TP,FP,TN,FN = 0,0,0,0
    for i in range(len(actual)):
        if actual[i]==prediction[i]==1.0:
            TP += 1
        if prediction[i]==1.0 and actual[i]!=prediction[i]:
            FP += 1
        if actual[i]==prediction[i]==0.0:
            TN += 1
        if prediction[i]==0 and actual[i]!=prediction[i]:
            FN += 1
    matrix[0][0]=TP
    matrix[0][1]=FP
    matrix[1][0]=FN
    matrix[1][1]=TN
    return matrix
```

In [10]:

```
#confusion Matrix for random Forest
rf=confusion_matrix(data['y_act'],data['y_pred_random_forest'])
print("Confusion Matrix for Random Forest \n {}".format(rf))
```

```
Confusion Matrix for Random Forest
 [[5047. 2360.]
 [2832. 5519.]]
```

In [11]:

```
#confusion Matrix for logistic Regression
lr=confusion_matrix(data['y_act'],data['y_pred_logistic'])
print("Confusion Matrix for logistic Regression \n {}".format(lr))
```

```
Confusion Matrix for logistic Regression
 [[4279. 2454.]
 [3600. 5425.]]
```

In [12]:

```
#Precision for random Forest
prec_rf = rf[0][0]/(rf[0][0]+rf[0][1])
print("Precision Score for Random Forest {}".format(round(prec_rf,4)))
```

```
Precision Score for Random Forest 0.6814
```

In [13]:

```
#precision for logistic regression
prec_lr= lr[0][0]/(lr[0][0]+lr[0][1])
print("Precision Score for Logic Regression {}".format(round(prec_lr,4)))
```

```
Precision Score for Logic Regression 0.6355
```

In [14]:

```
#Recall Score for random Forest
recall_rf = rf[0][0]/(rf[0][0]+rf[1][0])
print("Recall Score for Random Forest {}".format(round(recall_rf,4)))
#Recall Score for Logistic Regression
recall_lr = lr[0][0]/(lr[0][0]+lr[1][0])
print("Recall Score for Logic Regression {}".format(round(recall_lr,4)))
```

```
Recall Score for Random Forest 0.6406
Recall Score for Logic Regression 0.5431
```

In [15]:

```
#Normal F1 Score for random Forest
f1_rf = rf[0][0]/(rf[0][0]+(rf[0][1]+rf[1][0])/2)
print("Normal F1 Score for Random Forest {}".format(round(f1_rf,4)))
#Normal F1 Score for Logistic Regression
f1_lr = lr[0][0]/(lr[0][0]+(lr[0][1]+lr[1][0])/2)
print("Normal F1 Score for Logistic Regression {}".format(round(f1_lr,4)))
```

```
Normal F1 Score for Random Forest 0.6603
Normal F1 Score for Logistic Regression 0.5857
```

In [ ]: