Practical No. 05

by Kartik Deshpande

Data Analytics II

1. Implement logistic regression using Python/R to perform classification on

Social_Network_Ads.csv dataset. 2. Compute Confusion matrix to find TP, FP, TN, FN, Accuracy, Error rate, Precision, Recall on the given dataset.

In [14]:
```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, reca
```

In [15]:
```
df = pd.read_csv("/home/kartik/Documents/Python Notebooks/Social_Network_Ads.csv")
```

In [16]:
```
df['Gender'].replace({"Male":0,"Female":1},inplace = True)
df
```

Out[16]:

	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	0	19	19000	0
1	15810944	0	35	20000	0
2	15668575	1	26	43000	0
3	15603246	1	27	57000	0
4	15804002	0	19	76000	0
...	...	...	...	...	...
395	15691863	1	46	41000	1
396	15706071	0	51	23000	1
397	15654296	1	50	20000	1
398	15755018	0	36	33000	0
399	15594041	1	49	36000	1

400 rows × 5 columns

In [25]:
```
df.columns
```

Out[25]:
```
Index(['User ID', 'Gender', 'Age', 'EstimatedSalary', 'Purchased'], dtype='objec
t')
```

In [24]:
```
x = df[['User ID', 'Gender', 'Age', 'EstimatedSalary']]
y = df[['Purchased']]
```

In [23]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=29
```

In [27]:
```
model = LogisticRegression()
model.fit (x_train,y_train)
```

```
/home/kartik/anaconda3/lib/python3.11/site-packages/sklearn/utils/validation.py:114
3: DataConversionWarning: 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().
  y = column_or_1d(y, warn=True)
```

Out[27]:
```
▾ LogisticRegression

LogisticRegression()
```

In [28]:
```
y_pred = model.predict(x_test)
```

In [29]:
```
y_pred
```

Out[29]:
```
array([0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0,
       1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0])
```

In [30]:
```
model.score(x_train,y_train)
```

Out[30]:
```
0.7833333333333333
```

In [31]:
```
model.score(x,y)
```

Out[31]:
```
0.785
```

In [32]:
```
cm = confusion_matrix(y_test,y_pred)
cm
```

Out[32]:
```
array([[64,  5],
       [16, 15]])
```

In [33]:
```
tn, fp, fn, tp = confusion_matrix(y_test,y_pred).ravel()
```

In [34]:
```
print (tn, fp, fn, tp)
```

```
64 5 16 15
```

In [35]:
```
a = accuracy_score(y_test,y_pred)
a
```

Out[35]:
```
0.79
```

In [36]:
```
e = 1 - a
e
```

Out[36]: 0.20999999999999996

In [39]:
```
precision_score(y_test,y_pred)
```

Out[39]: 0.75

In [40]:
```
recall_score(y_test,y_pred)
```

Out[40]: 0.4838709677419355

In [ ]: