In [82]:
```
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, recall_score
```

In [83]:
```
df = pd.read_csv("Social_Network_Ads.csv")
df
```

Out[83]:

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

400 rows × 5 columns

In [84]:
```
df["Gender"] = df["Gender"].replace({"Male": 0, "Female": 1})
df
```

```
C:\Users\Ritesh Kolte\AppData\Local\Temp\ipykernel_19336\3117893451.py:1: FutureWarning: Downcasting behavior in `replace` is deprecated and will be removed in a future version. To retain the old behavior, explicitly call `result.infer_objects(copy=False)`. To opt
-in to the future behavior, set `pd.set_option('future.no_silent_downcasting', True)`
  df["Gender"] = df["Gender"].replace({"Male": 0, "Female": 1})
```

Out[84]:

	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 [85]:
```
df.columns
```

Out[85]:
```
Index(['User ID', 'Gender', 'Age', 'EstimatedSalary', 'Purchased'], dtype='object')
```

In [86]:
```
x = df[['User ID', 'Gender', 'Age', 'EstimatedSalary']]
y = df['Purchased']
```

In [87]:
```
x
```

Out[87]:

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

400 rows × 4 columns

In [88]:
```
y
```

Out[88]:
```
0      0
1      0
2      0
3      0
4      0
      ..
395    1
396    1
397    1
398    0
399    1
Name: Purchased, Length: 400, dtype: int64
```

In [89]:
```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.25,random_state=29)
```

In [90]:
```
model = LogisticRegression()
model.fit(x_train,y_train)
```

Out[90]:
```
LogisticRegression()
```

In [91]:
```
y_pred = model.predict(x_test)
```

In [92]:
```
y_pred
```

Out[92]:
```
array([0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 0,
       0, 0, 0, 0, 1, 1, 0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0,
       0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
       1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1])
```

In [93]:
```
model.score(x_train,y_train)
```

Out[93]:
```
0.8633333333333333
```

In [94]:
```
model.score(x,y)
```

Out[94]:
```
0.85
```

In [96]:
```
cm = confusion_matrix(y_test, y_pred)
cm
```

Out[96]:
```
array([[63,  6],
       [13, 18]])
```

In [99]:
```
tn, fp, fn, tp = confusion_matrix(y_test, y_pred).ravel()
```

In [100…
```
print(tn,fp,fn,tp)
```

```
63 6 13 18
```

In [101…
```
a = accuracy_score(y_test,y_pred)
a
```

Out[101…
```
0.81
```

In [103…
```
e = 1 - a    #error value
e
```

Out[103…
```
0.18999999999999995
```

In [104…
```
precision_score(y_test,y_pred)
```

Out[104…
```
0.75
```

In [105…
```
recall_score(y_test,y_pred)
```

Out[105… 0.5806451612903226

In [ ]: