ASSIGNMENT - 5

In [1]:
```
import pandas as pd
```

In [2]:
```
df = pd.read_csv("Social_Network_Ads.csv")
```

In [3]:
```
df
```

Out[3]:

	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 [4]:
```
df.head(6)
```

Out[4]:

	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
5	15728773	Male	27	58000	0

In [5]:
```
df.tail(4)
```

Out[5]:

	User ID	Gender	Age	EstimatedSalary	Purchased
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

In [7]:
```
df.shape
```

Out[7]: (400, 5)

In [8]:
```
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

In [9]:
```
df = df.drop(columns=['User ID'])
```

In [38]:
```
#In above command we drop User ID column so total reamaing coloumns are 4 .L
df.shape
```

Out[38]: (400, 4)

In [12]:
```
df.head(1) #successfully deleted column- User ID
```

Out[12]:

	Gender	Age	EstimatedSalary	Purchased
0	Male	19	19000	0

In [13]:
```
encoder = LabelEncoder()
df['Gender'] = encoder.fit_transform(df['Gender'])
```

In [14]:
```
from sklearn.model_selection import train_test_split
```

In [37]:
```
X = df.drop(columns=['Purchased'])
y = df['Purchased']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rar
```

In [16]:
```
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

In [17]:
```
from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_train, y_train)
```

Out[17]:

▾ LogisticRegression

LogisticRegression()

In [33]:
```
y_pred = model.predict(X_test)
```

In [36]:
```
from sklearn.metrics import confusion_matrix, accuracy_score, precision_scor
```

In [24]:
```
cm = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='binary')
recall = recall_score(y_test, y_pred, average='binary')
```

In [25]:
```
error_rate = 1 - accuracy
```

In [27]:
```
print("Confusion Matrix:\n", cm)
```

```
Confusion Matrix:
 [[50  2]
 [ 7 21]]
```

In [28]:
```
print(f"Accuracy: {accuracy:.2f}")
print(f"Precision: {precision:.2f}")
print(f"Recall: {recall:.2f}")
print(f"Error Rate: {error_rate * 100:.2f}%")
```

```
Accuracy: 0.89
Precision: 0.91
Recall: 0.75
Error Rate: 11.25%
```

In [ ]:
```

```