Practical No-5

Date of Conduction : **Date of Checking:**

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.

Python Code:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.impute import SimpleImputer
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
from sklearn.metrics import confusion_matrix, accuracy_score,
precision_score, recall_score, f1_score
# Load the Social_Network_Ads dataset
social_data = pd.read_csv('Social_Network_Ads.csv')

# Check for NaN values
print(social_data.head(25))

# Separate features (X) and target variable (y)
X = social_data[['Age', 'EstimatedSalary']]
y = social_data['Purchased']

# Handle NaN values by imputing the mean
imputer = SimpleImputer(strategy='mean')
X = imputer.fit_transform(X)

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Feature scaling
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Create a Logistic Regression model
model = LogisticRegression(random_state=42)

# Train the model
model.fit(X_train, y_train)

# Make predictions on the test set
y_pred = model.predict(X_test)
```

```
# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print(f'Accuracy: {accuracy}')
print('Confusion Matrix:')
print(conf_matrix)

# Compute Confusion Matrix
conf_matrix = confusion_matrix(y_test, y_pred)

# Extract values from the confusion matrix
TN, FP, FN, TP = conf_matrix.ravel()

# Compute Performance Metrics
accuracy = accuracy_score(y_test, y_pred)
error_rate = 1 - accuracy
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

# Print the results
print("Confusion Matrix:")
print(conf_matrix)
print("\nTrue Positive (TP):", TP)
print("False Positive (FP):", FP)
print("True Negative (TN):", TN)
print("False Negative (FN):", FN)
print("\nAccuracy:", accuracy)
print("Error Rate:", error_rate)
print("Precision:", precision)
print("Recall:", recall)
print("F1 Score:", f1)

# Plot the decision boundary
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Age', y='EstimatedSalary', hue='Purchased',
data=social_data, palette='viridis')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')

# Plot the decision boundary
plt.figure(figsize=(10, 6))
sns.scatterplot(x='Age', y='EstimatedSalary', hue='Purchased',
data=social_data, palette='viridis')
plt.xlabel('Age')
plt.ylabel('Estimated Salary')

# Plotting decision boundary
h = 0.5
x_min, x_max = X[:, 0].min() - 1, X[:, 0].max() + 1
y_min, y_max = X[:, 1].min() - 1, X[:, 1].max() + 1

xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max,
h))
Z = model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.3)
```

```
plt.title('Logistic Regression Decision Boundary')
plt.show()
```

OUTPUT:

"C:\Users\Ram Kumar Solanki\PycharmProjects\pythonProject\venv\Scripts\python.exe" "C:\Users\Ram Kumar Solanki\PycharmProjects\MBA_BFS\main.py"

	Age	EstimatedSalary	Purchased
0	19	19000	0
1	35	20000	0
2	26	43000	0
3	27	57000	0
4	19	76000	0
5	27	58000	0
6	27	84000	0
7	32	150000	1
8	25	33000	0
9	35	65000	0
10	26	80000	0
11	26	52000	0
12	20	86000	0
13	32	18000	0
14	18	82000	0
15	29	80000	0
16	47	25000	1
17	45	26000	1
18	46	28000	1
19	48	29000	1
20	45	22000	1
21	47	49000	1
22	48	41000	1
23	45	22000	1

24 46 23000 1

Accuracy: 0.8625

Confusion Matrix:

[[50 2]

[9 19]]

Confusion Matrix:

[[50 2]

[9 19]]

True Positive (TP): 19

False Positive (FP): 2

True Negative (TN): 50

False Negative (FN): 9

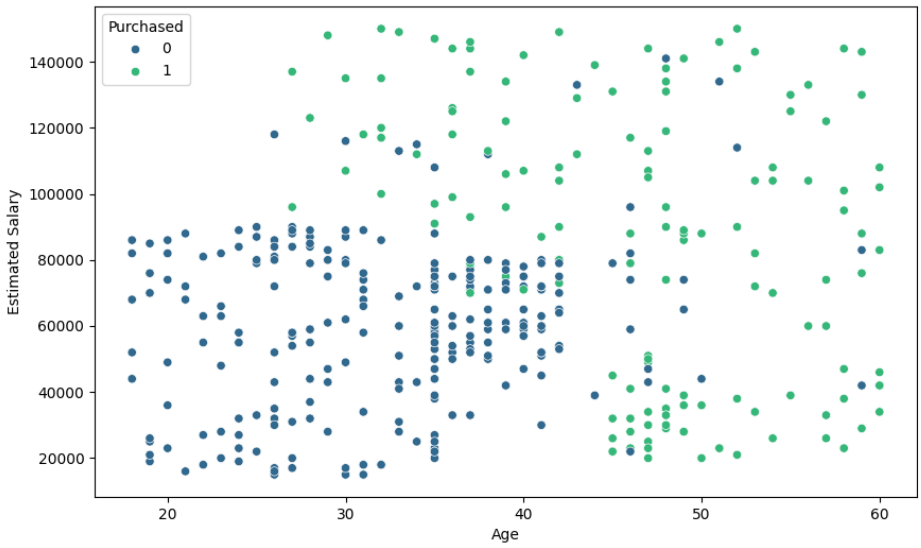
Accuracy: 0.8625

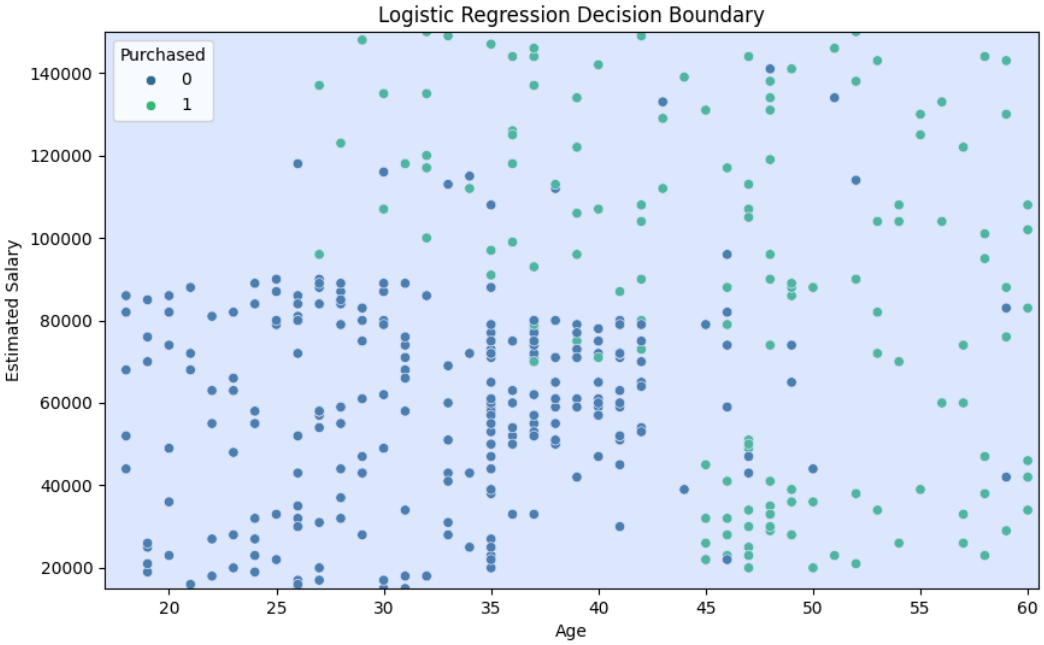
Error Rate: 0.13749999999999996

Precision: 0.9047619047619048

Recall: 0.6785714285714286

F1 Score: 0.7755102040816326




Logistic Regression Decision Boundary
Purchased
0
1
Estimated Salary
140000
120000
100000
80000
60000
40000
20000
20
25
30
35
40
45
50
55
60
Age