ML_LAB_05

2403A52006

Md.Mustafa

Import Required Libraries

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

Load the Dataset

```
# Upload Social_Network_Ads.csv in Colab before running this
df = pd.read_csv("/content/Social_Network_Ads.csv")

df.head()
```

	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

Next steps: Generate code with df New interactive sheet

Select Features and Target

```
X = df[['Age', 'EstimatedSalary']]   # Features
y = df['Purchased']                 # Target
```

Apply Feature Scaling

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

Function to Train & Evaluate Model for Different Splits

```
def train_and_evaluate(test_size):
    X_train, X_test, y_train, y_test = train_test_split(
        X_scaled, y, test_size=test_size, random_state=42
    )

    model = LogisticRegression()
    model.fit(X_train, y_train)

    y_pred = model.predict(X_test)

    print(f"\nTrain/Test Split: {int((1-test_size)*100)}/{int(test_size*100)}")
    print("Accuracy:", accuracy_score(y_test, y_pred))
    print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
    print("Classification Report:\n", classification_report(y_test, y_pred))

    return model
```

Train Models with Different Train/Test Splits

```
model_65 = train_and_evaluate(0.35)
model_75 = train_and_evaluate(0.25)
model_80 = train_and_evaluate(0.20)
```

```

Train/Test Split: 65/35
Accuracy: 0.8357142857142857
Confusion Matrix:
 [[82  2]
 [21 35]]
Classification Report:
               precision    recall  f1-score   support

           0       0.80      0.98      0.88        84
           1       0.95      0.62      0.75        56

    accuracy                           0.84       140
   macro avg       0.87      0.80      0.81       140
weighted avg       0.86      0.84      0.83       140


Train/Test Split: 75/25
Accuracy: 0.86
Confusion Matrix:
 [[61  2]
 [12 25]]
Classification Report:
               precision    recall  f1-score   support

           0       0.84      0.97      0.90        63
           1       0.93      0.68      0.78        37
    accuracy                           0.86       100
   macro avg       0.88      0.82      0.84       100
weighted avg       0.87      0.86      0.85       100


Train/Test Split: 80/20
Accuracy: 0.8625
Confusion Matrix:
 [[50  2]
 [ 9 19]]
Classification Report:
               precision    recall  f1-score   support

           0       0.85      0.96      0.90        52
           1       0.90      0.68      0.78        28

    accuracy                           0.86        80
   macro avg       0.88      0.82      0.84        80
weighted avg       0.87      0.86      0.86        80
```

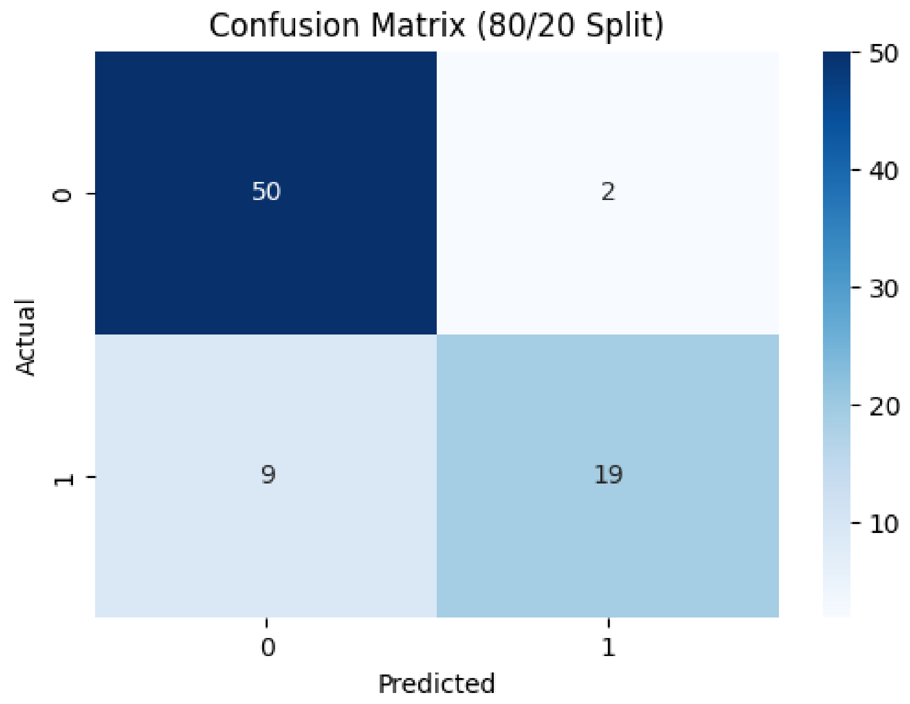
Confusion Matrix Heatmap (Best Split – 80/20)

```
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.20, random_state=42
)

best_model = LogisticRegression()
best_model.fit(X_train, y_train)
y_pred = best_model.predict(X_test)

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(6,4))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel("Predicted")
plt.ylabel("Actual")
plt.title("Confusion Matrix (80/20 Split)")
plt.show()
```



Plot Decision Boundary (2D) – Best Split

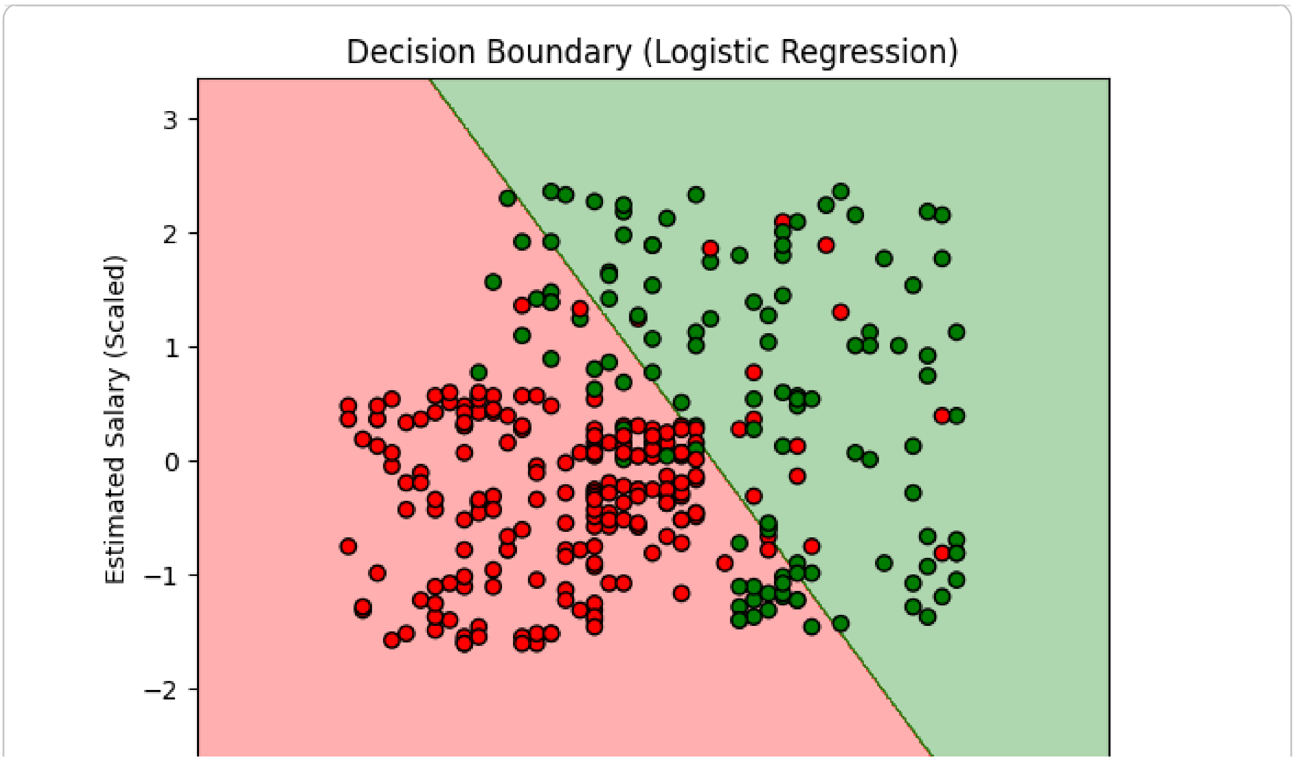
```
from matplotlib.colors import ListedColormap

X_set, y_set = X_train, y_train
X1, X2 = np.meshgrid(
    np.arange(X_set[:, 0].min()-1, X_set[:, 0].max()+1, 0.01),
    np.arange(X_set[:, 1].min()-1, X_set[:, 1].max()+1, 0.01)
)

plt.contourf(
    X1, X2,
    best_model.predict(np.array([X1.ravel(), X2.ravel()]).T).reshape(X1.shape),
    alpha=0.3,
    cmap=ListedColormap(('red', 'green'))
)

plt.scatter(
    X_set[:, 0], X_set[:, 1],
    c=y_set,
    cmap=ListedColormap(('red', 'green')),
    edgecolor='k'
)

plt.xlabel('Age (Scaled)')
plt.ylabel('Estimated Salary (Scaled)')
plt.title('Decision Boundary (Logistic Regression)')
plt.show()
```


Decision Boundary (Logistic Regression)
Estimated Salary (Scaled)
3
2
1
0
−1
−2