smvegppxq

March 31, 2025

[137]:
```
# Import necessary libraries
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

# Import machine learning tools
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import precision_score, confusion_matrix, accuracy_score,␣
↪recall_score
```

[143]:
```
# Load dataset (Replace '.csv' with actual dataset path)
df = pd.read_csv("social_network_ads.csv")

# Display first few rows
print(df.head())

# Check dataset information
df.info()

# Check for missing values
print(df.isnull().sum())
```

```
    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
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 5 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   User ID          400 non-null    int64
 1   Gender           400 non-null    object
```

```
 2   Age              400 non-null    int64
 3   EstimatedSalary  400 non-null    int64
 4   Purchased        400 non-null    int64
dtypes: int64(4), object(1)
memory usage: 15.8+ KB
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

[175]:
```
# Convert categorical variables to numerical (if applicable)
if 'Gender' in df.columns:
    df['Gender'] = df['Gender'].map({'Male': 0, 'Female': 1})
```

[187]:
```
# Assume 'Target' is the dependent variable (Replace with actual column name)
X = df[['Age', 'EstimatedSalary']]  # Features
y = df['Purchased']  # Target variable
```

[189]:
```
# Split dataset (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# Display dataset shapes
print("Training Data Shape:", X_train.shape)
print("Testing Data Shape:", X_test.shape)
```

```
Training Data Shape: (320, 2)
Testing Data Shape: (80, 2)
```

[191]:
```
# Initialize scaler
scaler = StandardScaler()

# Fit and transform training data
X_train = scaler.fit_transform(X_train)

# Transform test data
X_test = scaler.transform(X_test)
```

[193]:
```
# Initialize model
logreg = LogisticRegression()

# Train model
logreg.fit(X_train, y_train)

# Predict output for test data
```

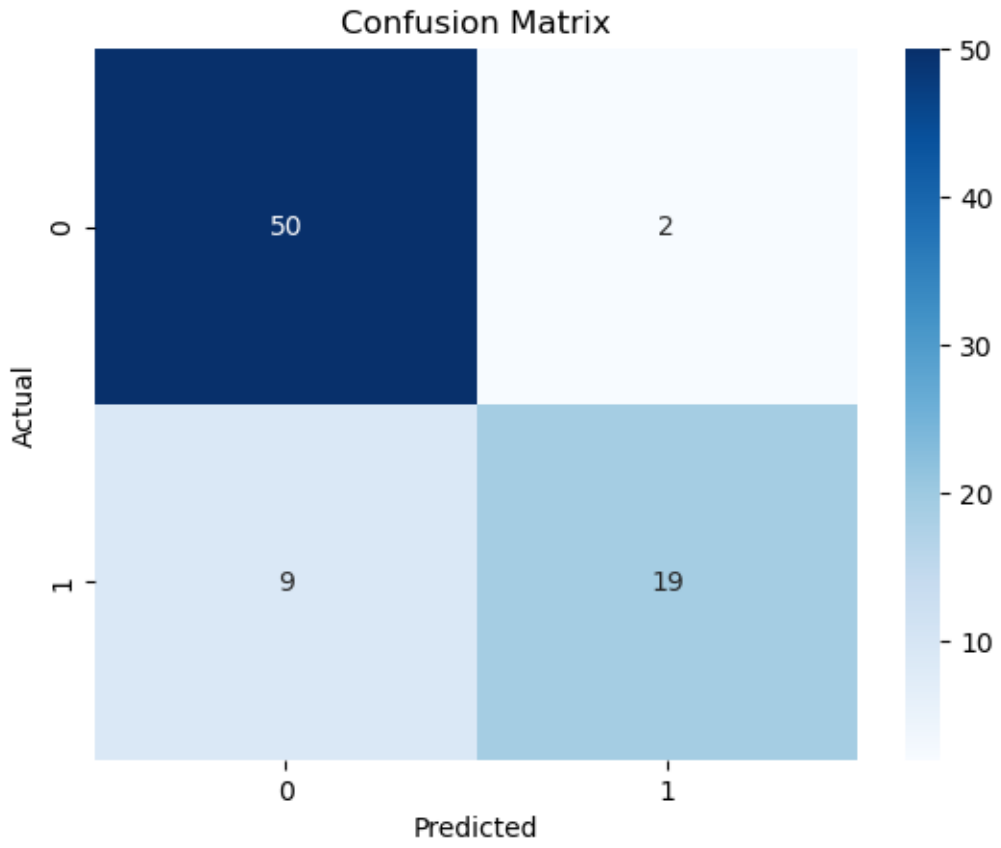
```
y_pred = logreg.predict(X_test)
```

[195]:
```
# Evaluating Model Performance
cm = confusion_matrix(y_test, y_pred)
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)

# Printing evaluation metrics
print("\nConfusion Matrix:\n", cm)
print("Accuracy Score:", accuracy)
print("Precision Score:", precision)
print("Recall Score:", recall)
```

```
Confusion Matrix:
 [[50  2]
 [ 9 19]]
Accuracy Score: 0.8625
Precision Score: 0.9047619047619048
Recall Score: 0.6785714285714286
```

[197]:
```
# Optional: Visualizing the confusion matrix
import seaborn as sns
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



[205]:
```
print(df.columns)
```

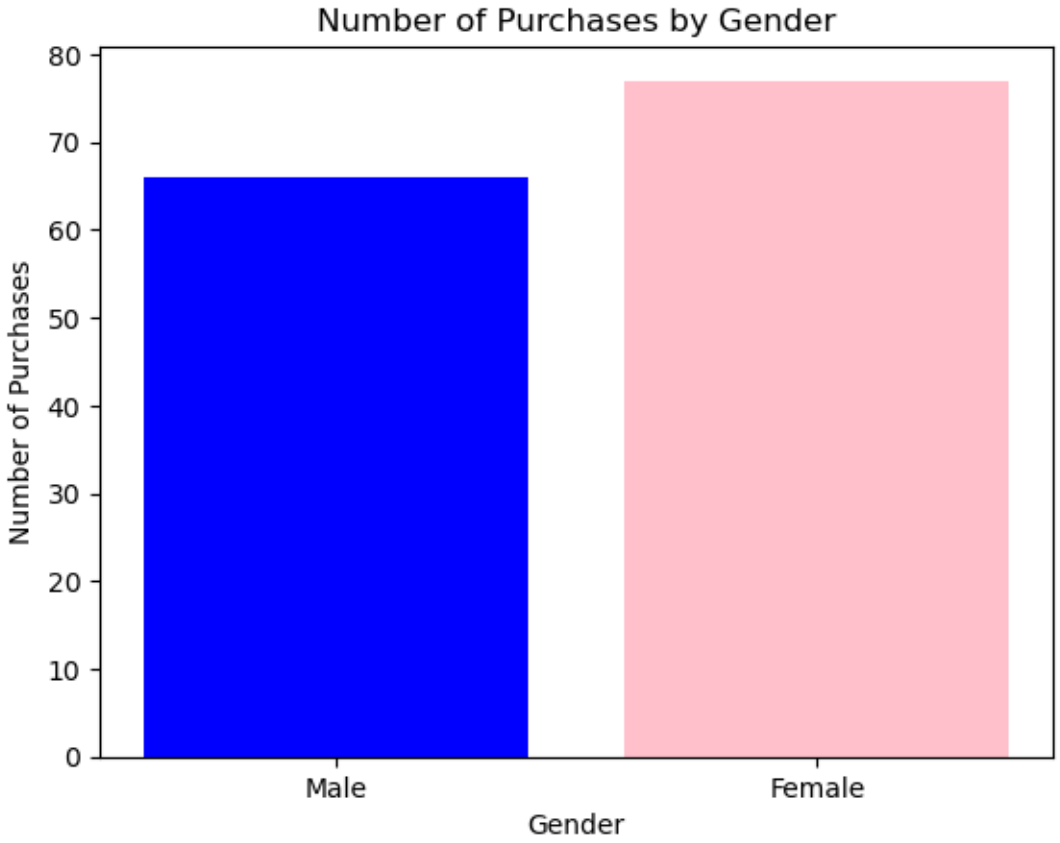
```
Index(['User ID', 'Age', 'EstimatedSalary', 'Purchased', 'Gender_Male'],
dtype='object')
```

[209]:
```
# Adjusted Gender Column Handling
if 'Gender_Male' in df.columns and 'Purchased' in df.columns:
    # Calculate the number of purchases for males and females
    male_purchases = df[df['Gender_Male'] == 1]['Purchased'].sum()
    female_purchases = df[df['Gender_Male'] == 0]['Purchased'].sum()

    # Bar chart for gender-wise purchases
    gender_labels = ['Male', 'Female']
    purchase_counts = [male_purchases, female_purchases]

    plt.bar(gender_labels, purchase_counts, color=['blue', 'pink'])
    plt.xlabel('Gender')
    plt.ylabel('Number of Purchases')
    plt.title('Number of Purchases by Gender')
    plt.show()
```

```
else:
    print("Gender_Male or Purchased column not found in dataset.")
```



[201]:
```
print(df.columns)
```

```
Index(['User ID', 'Age', 'EstimatedSalary', 'Purchased', 'Gender_Male'],
dtype='object')
```

[ ]:
```

```