practical-5

March 10, 2024

[ ]:
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler,LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score,␣
 ↪recall_score
import seaborn as sns
import matplotlib.pyplot as plt
```

[ ]:
```
data = pd.read_csv('/content/Social_Network_Ads.csv')
```

[ ]:
```
data.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
```

[ ]:
```
data.isnull().sum()
```

[ ]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

[ ]:
```
data.info()
```

```
<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
```

[ ]:
```
label_encoder = LabelEncoder()
data['Gender'] = label_encoder.fit_transform(data['Gender'])        #1-Male␣
 ↪0-Female
```

[ ]:
```
data.head()
```

[ ]:
```
    User ID  Gender  Age  EstimatedSalary  Purchased
0  15624510       1   19            19000          0
1  15810944       1   35            20000          0
2  15668575       0   26            43000          0
3  15603246       0   27            57000          0
4  15804002       1   19            76000          0
```

[ ]:
```
X = data.drop(columns=['User ID', 'Purchased'])
y = data['Purchased']
```

[ ]:
```
X
```

[ ]:
```
     Gender  Age  EstimatedSalary
0         1   19            19000
1         1   35            20000
2         0   26            43000
3         0   27            57000
4         1   19            76000
..      …   …              …
395       0   46            41000
396       1   51            23000
397       0   50            20000
398       1   36            33000
399       0   49            36000

[400 rows x 3 columns]
```

[ ]:
```
y
```

[ ]:
```
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
```

```
[ ]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
      ↪random_state=42)
```

```
[ ]: sc = StandardScaler()
     X_train = sc.fit_transform(X_train)
     X_test = sc.transform(X_test)
```

```
[ ]: classifier = LogisticRegression(random_state=42)
     classifier.fit(X_train, y_train)
```

```
[ ]: LogisticRegression(random_state=42)
```

```
[ ]: y_pred = classifier.predict(X_test)
```

```
[ ]: y_pred
```

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

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

```
[ ]: cm
```

```
[ ]: array([[50,  2],
            [ 7, 21]])
```

```
[ ]: print ("Accuracy : ", accuracy_score(y_test, y_pred))
```

```
Accuracy :  0.8875
```

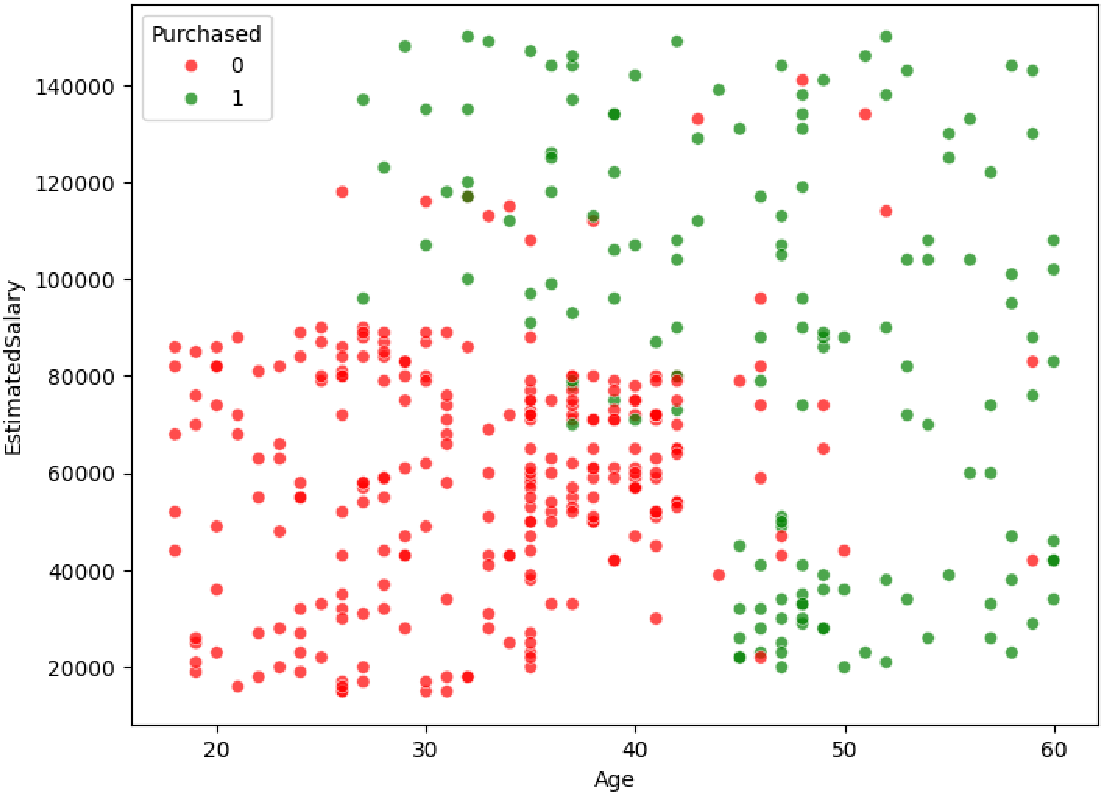
```
[ ]: accuracy = accuracy_score(y_test, y_pred)
     error_rate = 1 - accuracy
     precision = precision_score(y_test, y_pred)
     recall = recall_score(y_test, y_pred)
```

```
[ ]: print("Confusion Matrix:")
     print(cm)
     print("Accuracy:", accuracy)
     print("Error Rate:", error_rate)
     print("Precision:", precision)
     print("Recall:", recall)
```

```
Confusion Matrix:
[[50  2]
 [ 7 21]]
Accuracy: 0.8875
Error Rate: 0.11250000000000004
Precision: 0.9130434782608695
Recall: 0.75
```

```
[ ]: plt.figure(figsize=(8, 6))
     sns.scatterplot(x='Age', y='EstimatedSalary', hue='Purchased', data=data,␣
      ↪palette=['red', 'green'], alpha=0.7)
```

```
[ ]: <Axes: xlabel='Age', ylabel='EstimatedSalary'>
```


Purchased
● 0
● 1
EstimatedSalary
Age