assignment-5

April 13, 2024

Data Analytics - II : Logistic Regression

Problem Statement

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.

[18]:
```
#imports
import numpy as np
import pandas as pd
import seaborn as sns
import warnings
import matplotlib.pyplot as plt
warnings.filterwarnings("ignore")
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix,classification_report
```

[19]:
```
data = pd.read_csv("Social_Network_Ads.csv")
```

[20]:
```
data.sample(5)
```

[20]:

	User ID	Gender	Age	EstimatedSalary	Purchased
34	15724858	Male	27	90000	0
236	15660541	Male	40	57000	0
261	15680587	Male	36	144000	1
288	15649668	Male	41	79000	0
225	15622171	Male	37	53000	0

[21]:
```
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
```

[22]:
```
data
```

[22]:

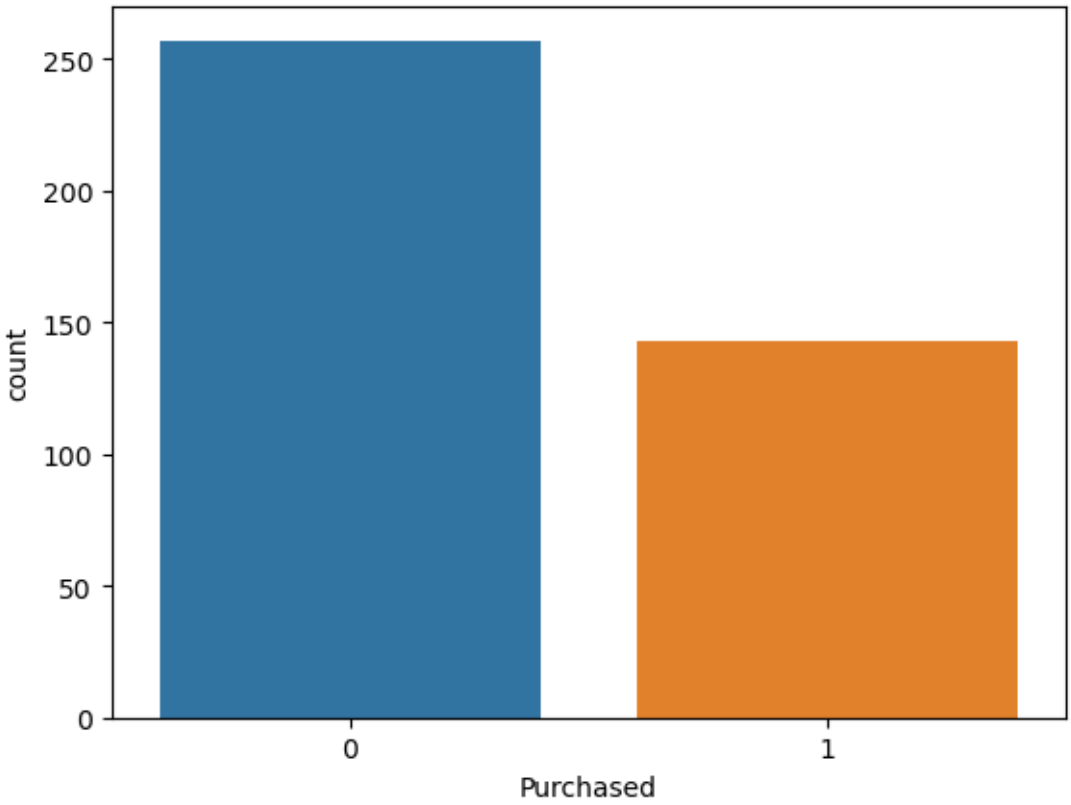
	User ID	Gender	Age	EstimatedSalary	Purchased
0	15624510	Male	19	19000	0
1	15810944	Male	35	20000	0
2	15668575	Female	26	43000	0
3	15603246	Female	27	57000	0
4	15804002	Male	19	76000	0
..	…	…	…	…	…
395	15691863	Female	46	41000	1
396	15706071	Male	51	23000	1
397	15654296	Female	50	20000	1
398	15755018	Male	36	33000	0
399	15594041	Female	49	36000	1

[400 rows x 5 columns]

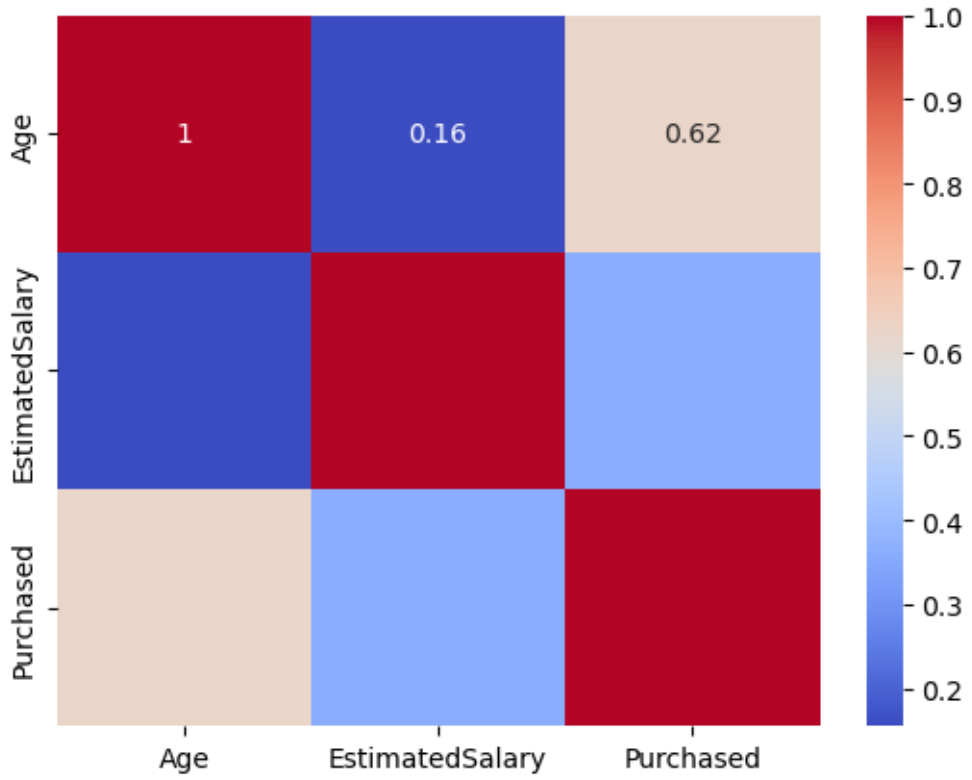
[23]:
```
data.isna().sum()
```

[23]:
```
User ID            0
Gender             0
Age                0
EstimatedSalary    0
Purchased          0
dtype: int64
```

[24]:
```
# Target label : 'Purchased'
sns.countplot(data = data, x = 'Purchased');
```



[25]:
```
# Finding useful features
sns.heatmap(data[['Age','EstimatedSalary','Purchased']].corr(), annot = True,␣
↪cmap= 'coolwarm' );
```



[26]:
```
features = data[['Age', 'EstimatedSalary']]
label = data['Purchased']
```

[27]:
```
scaler = StandardScaler()
features = scaler.fit_transform(features)
```

[28]:
```
x = features
y = label
```

[29]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.2,␣
↪random_state=42)
```

0.0.1 Model

[30]:
```
model = LogisticRegression()
model.fit(x_train, y_train)
```

[30]:
```
LogisticRegression()
```

0.0.2 Prediction

[31]:
```
y_pred = model.predict(x_test)
```

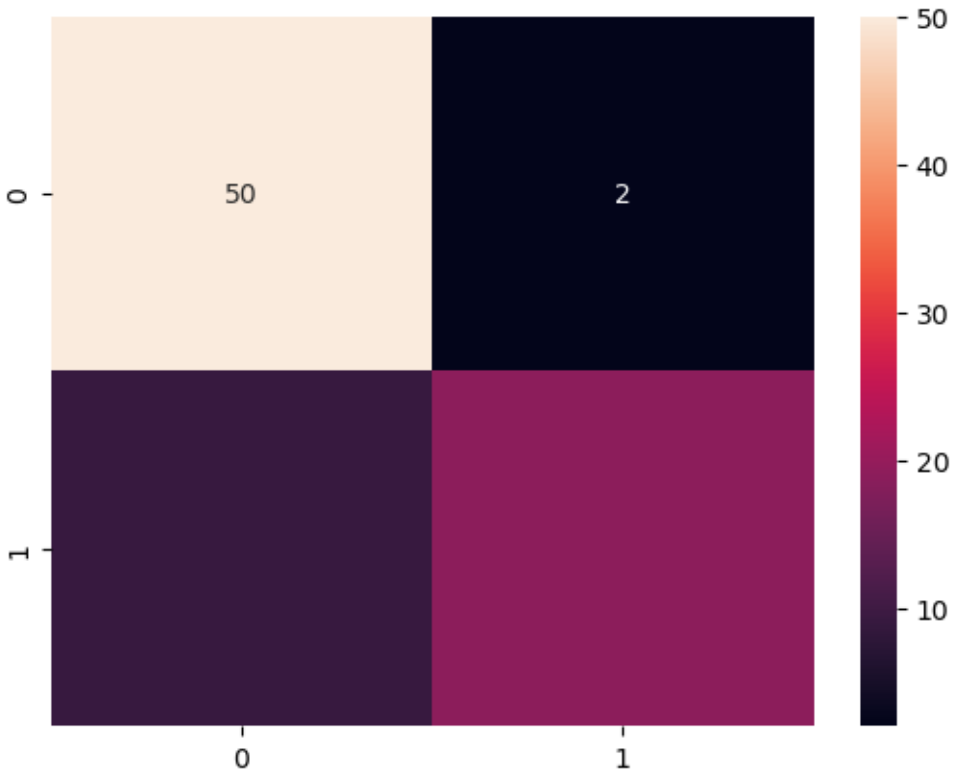
[32]:
```
y_pred
```

[32]:
```
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, 0, 1, 0, 1, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 1, 0, 0, 0, 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], dtype=int64)
```

0.0.3 Evaluation

[33]:
```
sns.heatmap(confusion_matrix(y_test, y_pred), annot= True)
```

[33]:
```
<Axes: >
```



[34]:
```
print(classification_report(y_test, y_pred))
```

```
              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