Purchase Prediction

Using Logistic Regression

Team Members:

Sudharsana Suresh[2019506100]

R Thanuja Varshini[2019506105]

V Yashwanth[2019506117]

Objective

To apply the logistic regression algorithm of machine learning to check whether the launched product would be purchased by a group of users.

We thereby, predict and calibrate the probability of its success in the market.

Motivation behind the work

The initial spark was ignited based on an article that suggested that many products failed to bloom in the market due to incorrect market of sale.

This project would aid in commercial online release and sale of a new product to a targeted group of interest and give a clear cut idea of how receptive the group would be to the product.

We can also derive a conclusion of the kind of people and their taste in our targeted set of interest.

We would also like to thank Mr.V.Sathiesh Kumar for his immense support and guidance which enabled us to successfully complete the project.

Methodology

We assume our product of interest to be *Nike unisex shoes*.

We have a data set consisting of 200 user details along with their history of categories they are most interested in.

Category IDs Index:

Clothing---100
Shoes---101
Grocery---102
Accessories---103
Bags---104
Sports---105
Toys---106
Beauty---107
Stationary---108
Appliances---109
Art---110

Data set:

User_ID	Age	Gender	Category	Purchased
180	19	F	100	0
102	70	M	101	1
126	23	M	102	0
100	44	M	100	0
119	55	F	102	0
106	67	M	102	0
117	70	F	101	1
190	20	F	103	0
120	21	F	104	0
321	27	M	105	0
432	26	F	106	0
563	33	M	106	0
231	25	F	107	0
111	45	F	107	0

Data is read and divided into input and output.

Data read is split into train and test data sets.

We perform feature scaling, as the values of age and category lie in different ranges.

We train the logistic regression model and fit it on the train set.

We perform the prediction on the testing data.

We evaluate the performance of the model using a confusion matrix.

We visualise the result in the form of a confusion matrix.

We calibrate the accuracy of the model using the confusion matrix.

An accuracy of greater than eighty percentage is considered as an ideal classification rate.

An accuracy of eighty-four percent was achieved by the model we had developed which speaks for its high level of performance and efficiency.

Source Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
import seaborn as sns

dataset=pd.read_csv(r'C:/Users/govin/.spyder-py3/Proj.csv.')
print(dataset.head())

#input
x=dataset.iloc[:,[1,3]].values

#output
y=dataset.iloc[:,4].values

#splitting dataset into train and test
x_train, x_test, y_train, y_test= train_test_split(
        x, y, test_size = 0.25, random_state = 0)

#model development
logreg=LogisticRegression()
```

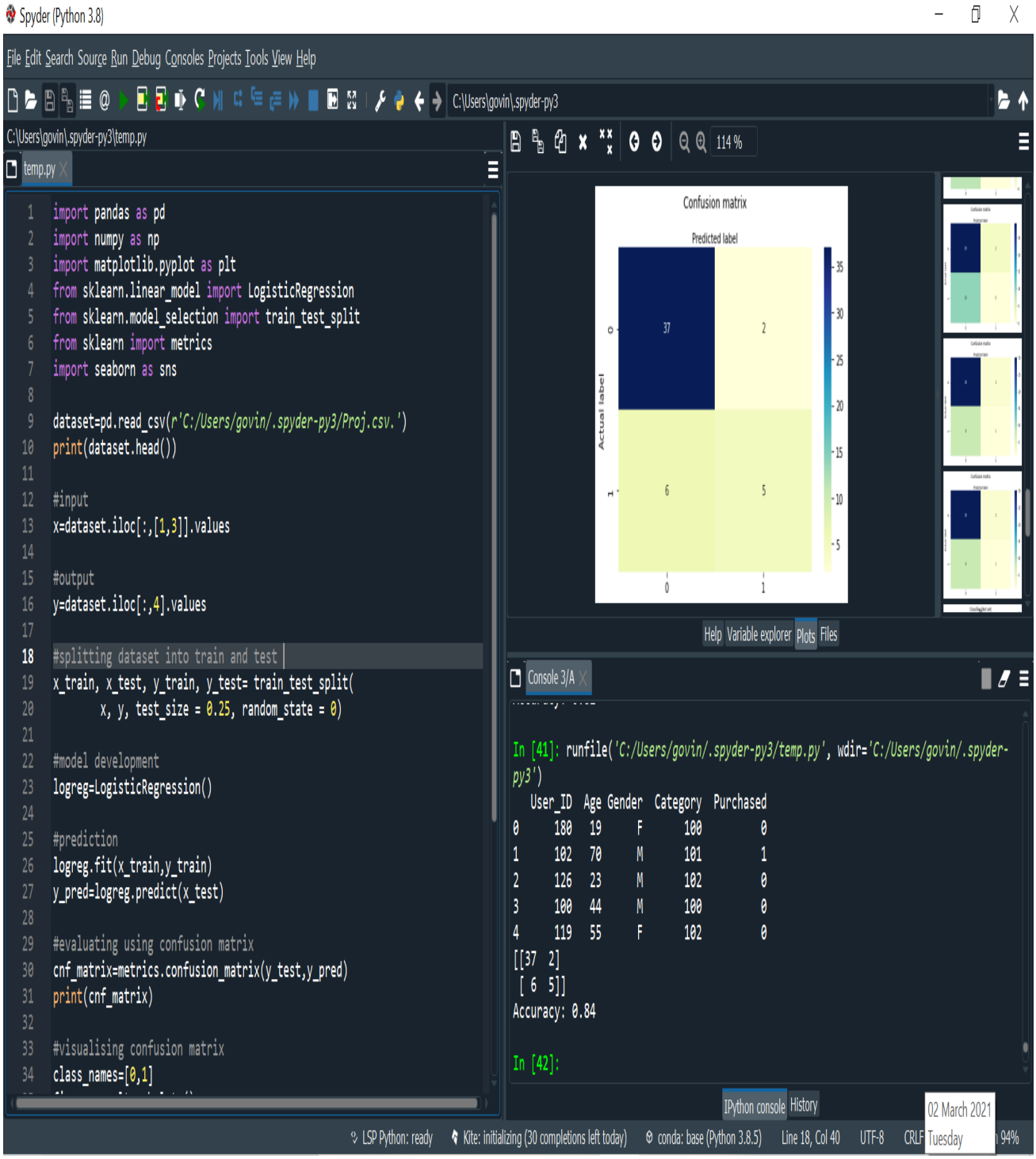
```
#prediction
logreg.fit(x_train,y_train)
y_pred=logreg.predict(x_test)

#evaluating using confusion matrix
cnf_matrix=metrics.confusion_matrix(y_test,y_pred)
print(cnf_matrix)

#visualising confusion matrix
class_names=[0,1]
fig, ax = plt.subplots()
tick_marks = np.arange(len(class_names))
plt.xticks(tick_marks, class_names)
plt.yticks(tick_marks, class_names)
# create heatmap
sns.heatmap(pd.DataFrame(cnf_matrix), annot=True, cmap="YlGnBu"
,fmt='g')
ax.xaxis.set_label_position("top")
plt.tight_layout()
plt.title('Confusion matrix', y=1.1)
plt.ylabel('Actual label')
plt.xlabel('Predicted label')

print("Accuracy:",metrics.accuracy_score(y_test, y_pred))
```

Output Screenshots



```
In [40]: runfile('C:/Users/govin/.spyder-py3/temp.py', wdir='C:/Users/govin/.spyder-py3')
   User_ID  Age Gender  Category  Purchased
0      180   19      F       100          0
1      102   70      M       101          1
2      126   23      M       102          0
3      100   44      M       100          0
4      119   55      F       102          0
[[-1.40502485  0.87917081]
 [ 0.10770533  0.87917081]
 [-0.55000345 -1.26329451]
 [ 1.22581025 -1.26329451]
 [ 1.09426849 -1.26329451]
 [-1.53656661  0.57310434]
 [-1.33925398 -1.26329451]
 [ 0.7654141   1.18523729]
 [-0.6815452   0.87917081]
 [-0.41846169  0.57310434]]
Confusion Matrix:
 [[37  2]
 [ 7  4]]
Accuracy: 0.82
```