Experiment No. 2

AIM: To study and implement Logistic Regression.

THEORY

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a givenset of independent variables.

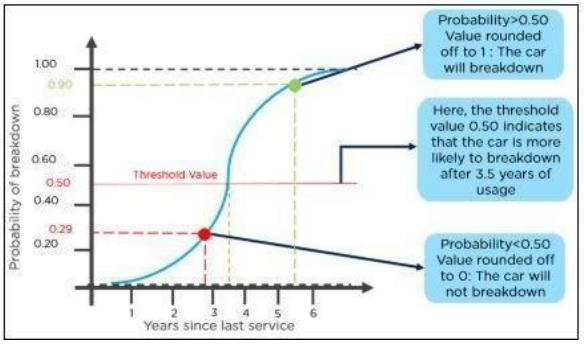
Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome mustbe a categorical or discrete value. It can be either Yes or No, 0 or 1, True or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Type of Logistic Regression

On the basis of the categories, Logistic Regression can be classified into three types:

1) Binomial: In binomial Logistic regression, there can be only two possible types of thedependent variables, such as 0 or 1, Pass or Fail, etc.
2) Multinomial: In multinomial Logistic regression, there can be 3 or more possible unorderedtypes of the dependent variable, such as "cat", "dogs", or "sheep"
3) Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types ofdependent variables, such as "low", "Medium", or "High".



The following is an example of a logistic function we can use to find the probability of a vehicle breaking down, depending on how many years it has been since it was serviced last.

PROCEDURE

1) Install Python 3.8 or above.
2) Install the required python libraries using the pip command:
    a) numpy
    b) pandas
    c) sklearn
    d) matplotlib
3) Run the below code on Jupyter notebook

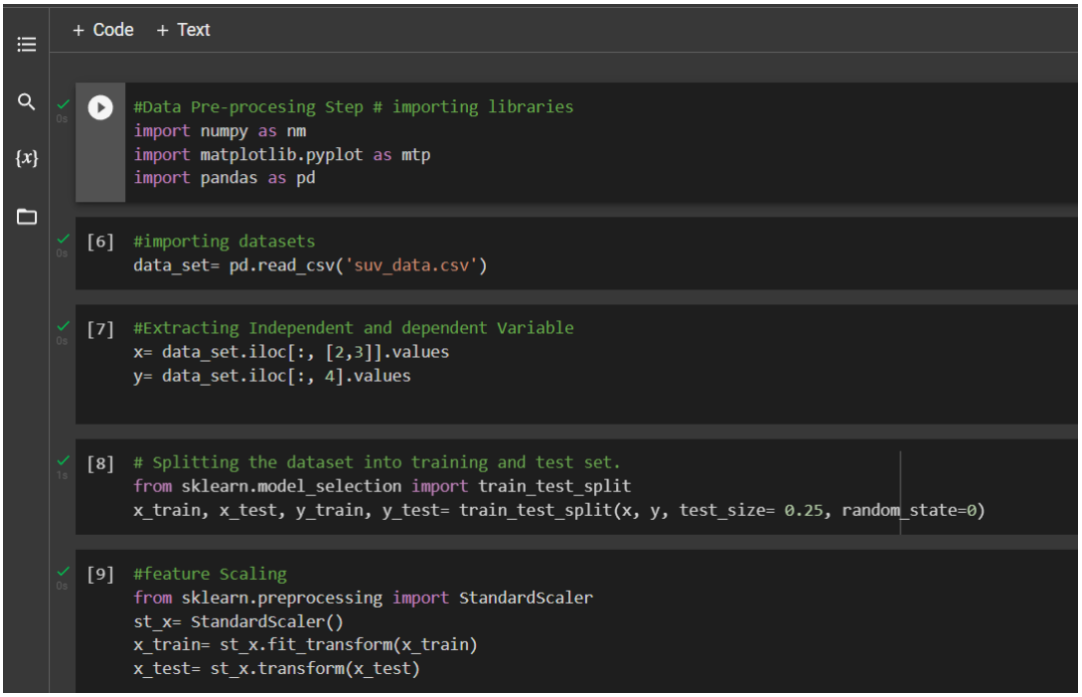
PYTHON PROGRAM

```
#Data Pre-procesing Step # importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

#importing datasets
data_set= pd.read_csv('suv_data.csv')

#Extracting Independent and dependent Variable
x= data_set.iloc[:, [2,3]].values
y= data_set.iloc[:, 4].values

# Splitting the dataset into training and test set.
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)

#feature Scaling
from sklearn.preprocessing import StandardScaler
st_x= StandardScaler()
x_train= st_x.fit_transform(x_train)
 x_test= st_x.transform(x_test)

#Fitting Logistic Regression to the training set
from sklearn.linear_model import LogisticRegression
classifier= LogisticRegression(random_state=0)
classifier.fit(x_train, y_train)
```

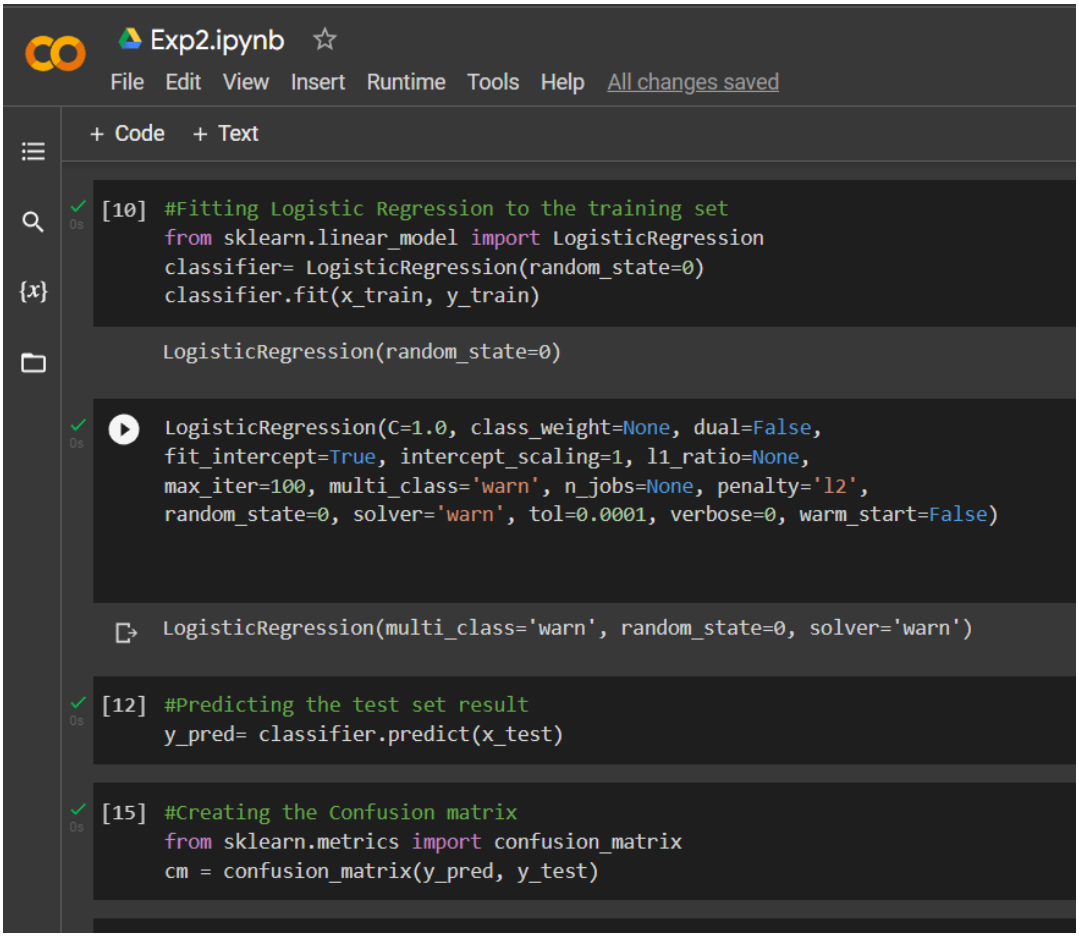
```
LogisticRegression(
C=1.0,
class_weight=None, dual=False, fit_intercept=True, intercept_scaling=1, l1_ratio=None, max_iter=100,
multi_class='warn', n_jobs=None, penalty='l2', random_state=0,
solver='warn', tol=0.0001,
```

```
verbose=0, warm_start=False
)

#Predicting the test set result
y_pred= classifier.predict(x_test)
#Creating the Confusion matrix
from sklearn.metrics import confusion_matrix
cm= confusion_matrix(y_pred, y_test)

#Visualizing the training set result
from matplotlib.colors import ListedColormap x_set, y_set = x_train, y_train
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step =0.01),
nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape), alpha = 0.75,
cmap = ListedColormap(('purple','green' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)): mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],
c = ListedColormap(('purple', 'green'))(i), label = j) mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary') mtp.legend()
mtp.show()
```

IMPLEMENTATION:



```
+ Code  + Text

#Data Pre-procesing Step # importing libraries
import numpy as nm
import matplotlib.pyplot as mtp
import pandas as pd

[6]  #importing datasets
     data_set= pd.read_csv('suv_data.csv')

[7]  #Extracting Independent and dependent Variable
     x= data_set.iloc[:, [2,3]].values
     y= data_set.iloc[:, 4].values

[8]  # Splitting the dataset into training and test set.
     from sklearn.model_selection import train_test_split
     x_train, x_test, y_train, y_test= train_test_split(x, y, test_size= 0.25, random_state=0)

[9]  #feature Scaling
     from sklearn.preprocessing import StandardScaler
     st_x= StandardScaler()
     x_train= st_x.fit_transform(x_train)
     x_test= st_x.transform(x_test)
```



```
Exp2.ipynb
File  Edit  View  Insert  Runtime  Tools  Help  All changes saved

+ Code  + Text

[10] #Fitting Logistic Regression to the training set
     from sklearn.linear_model import LogisticRegression
     classifier= LogisticRegression(random_state=0)
     classifier.fit(x_train, y_train)

     LogisticRegression(random_state=0)

     LogisticRegression(C=1.0, class_weight=None, dual=False,
     fit_intercept=True, intercept_scaling=1, l1_ratio=None,
     max_iter=100, multi_class='warn', n_jobs=None, penalty='l2',
     random_state=0, solver='warn', tol=0.0001, verbose=0, warm_start=False)

     LogisticRegression(multi_class='warn', random_state=0, solver='warn')

[12] #Predicting the test set result
     y_pred= classifier.predict(x_test)

[15] #Creating the Confusion matrix
     from sklearn.metrics import confusion_matrix
     cm = confusion_matrix(y_pred, y_test)
```

Exp2.ipynb

File Edit View Insert Runtime Tools Help

+ Code + Text

```
#Visualizing the training set result
from matplotlib.colors import ListedColormap
x_set, y_set = x_train, y_train
x1, x2 = nm.meshgrid(nm.arange(start = x_set[:, 0].min() - 1, stop = x_set[:, 0].max() + 1, step =0.01),
                     nm.arange(start = x_set[:, 1].min() - 1, stop = x_set[:, 1].max() + 1, step = 0.01))
mtp.contourf(x1, x2, classifier.predict(nm.array([x1.ravel(), x2.ravel()]).T).reshape(x1.shape), alpha = 0.75,
             cmap = ListedColormap(('purple','green' )))
mtp.xlim(x1.min(), x1.max())
mtp.ylim(x2.min(), x2.max())
for i, j in enumerate(nm.unique(y_set)):
  mtp.scatter(x_set[y_set == j, 0], x_set[y_set == j, 1],c = ListedColormap(('purple', 'green'))(i), label = j)
mtp.title('Logistic Regression (Training set)')
mtp.xlabel('Age')
mtp.ylabel('Estimated Salary')
mtp.legend()
mtp.show()
```

OUTPUT:

CONCLUSION: Thus, we studied and implemented Logistic Regression.