Class: BE **Subject: Machine Learning** **Sem-VII**

Experiment No:4

Course Outcome: CO3 **Blooms Level:L3**

Aim: To implement Logistic Regression using Python.

Abstract: Logistic Regression is a fundamental statistical method used for binary classification problems, where the objective is to predict a categorical outcome based on one or more independent variables. Logistic regression models the probability that a given input belongs to a particular class using the logistic (sigmoid) function. This function ensures output values between 0 and 1, which can be interpreted as probabilities.

Sample Input and Output:

Case 1: Input Features:

- **ApplicantIncome (in $)**
- **Credit_History (1 = Good, 0 = Bad)**

Output:

- **Loan_Status (1 = Approved, 0 = Rejected)**

Output

Input:

ApplicantIncome = 5000

Credit_History = 1

Output:

Loan_Status = 1 (Approved)

Theory:

Logistic Regression:

Logistic Regression is a supervised machine learning algorithm used for binary classification.

It estimates the probability of a binary outcome based on predictor variables.

The sigmoid function transforms the linear combination of predictor variables to a probability value between 0 and 1.

Preprocessing Data:

Similar to linear regression, data preprocessing includes handling missing values, scaling, and splitting into training and testing sets.

Implementing Logistic Regression:

Libraries like NumPy, pandas, and scikit-learn are commonly used for implementing Logistic Regression.

The logistic regression model is trained using available data and the logistic function.

Model Evaluation:

Metrics like accuracy, precision, recall, and F1-score are used to evaluate the performance of the logistic regression model.
Confusion matrices and ROC curves provide additional insights into the model's performance.

Program:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')
```

```
[13] from google.colab import drive
     drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=Tr
```

```
[16] df = pd.read_csv('/content/sample_data/2.01.+Admittance.csv')
     df.head()
```

```
[17] df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 168 entries, 0 to 167
Data columns (total 2 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   SAT       168 non-null    int64
 1   Admitted  168 non-null    object
dtypes: int64(1), object(1)
memory usage: 2.8+ KB
```

```
[18] df.shape
```

```
(168, 2)
```

```
[19] from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LogisticRegression
     from sklearn import metrics
```

```
[20] df.Admitted=df.Admitted.map({'Yes':1,'No':0})
     df['Admitted'].value_counts()
```

Admitted	count
1	94
0	74

dtype: int64

```
[21] X = df[['SAT']]          # Feature as DataFrame
     y = df['Admitted']       # Target
```

```
[22] # Split into training and test sets
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=
```

```
[23] model = LogisticRegression()
     model.fit(X_train, y_train)
```

```
LogisticRegression()
```

```
[27] y_pred = model.predict(X_test)
```

```
[28] from sklearn import metrics
```

```
[29] print("y_predicted",y_pred)
     print("y_test",y_test)
```

```
y_predicted [0 0 1 1 1 1 1 1 1 0 1 1 0 0 1 1 1 0 1 0 1 0 1 0 0 0 1 1 0 0 1 1 0
y_test 137    0
30     0
119    1
29     1
142    1
161    1
164    1
51     1
105    1
60     0
15     1
156    1
133    0
45     0
68     1
85     1
24     1
109    0
75     1
108    0
19     1
16     0
31     1
18     1
12     1
9      0
82     1
98     1
```

Output:

```
from sklearn.metrics import accuracy_score, classification_report

print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pre
```

```
Accuracy: 0.9117647058823529

Classification Report:
              precision    recall  f1-score   support

           0       0.87      0.93      0.90        14
           1       0.95      0.90      0.92        20

    accuracy                           0.91        34
   macro avg       0.91      0.91      0.91        34
weighted avg       0.91      0.91      0.91        34
```

Conclusion:

The implementation of Logistic Regression using Python is valuable for predicting binary outcomes, such as whether an email is spam or not, customer churn, and more. By preprocessing data, training the model, and evaluating its performance, you can gain practical experience in building and assessing logistic regression models for classification tasks.

Exercise 1:

You work in the loan approval department of a financial institution and would like to create a model that can help predict whether a loan application will be approved based on one of the features of the applicant. Feature selection should be done based on the results of correlation matrix. You decide that a Logistic Regression model might be a good candidate to solve this binary classification problem.

Students shall draw flowchart of exercise question in the writeup and submit.

Exercise 2:

Examine the Smarket data, which is part of the ISLP library. This data set consists of percentage

Class: BE **Subject: Machine Learning** **Sem-VII**

returns for the S&P 500 stock index over 1,250 days, from the beginning of 2001 until the end of 2005. For each date, we have recorded the percentage returns for each of the five previous trading days, Lag1 through Lag5. We have also recorded Volume (the number of shares traded on the previous day, in billions), Today (the percentage return on the date in question) and Direction (whether the market was Up or Down on this date). Use the full data set with Direction as the response and the five lag variables plus Volume as predictors.Select which predictors can be used and perform Logistic Regression on it.

Students shall draw flowchart of exercise question in the writeup and submit.