## **Experiment No: 5**

Aim: Perform Regression Analysis using Scipy and Sci-kit learn.

### <u>Theory</u>

### 1. Linear Regression:

Linear Regression is a supervised learning algorithm used to predict continuous numerical values.

- It assumes a linear relationship between the independent (input) and dependent (target) variables.
- The model minimizes the sum of squared errors to determine the best-fitting line.
- It is evaluated using performance metrics like Mean Squared Error (MSE) and R-squared (R<sup>2</sup>).
- It works best when data follows a linear trend but is sensitive to outliers.

#### 2. Logistic Regression:

Logistic Regression is a classification algorithm used for predicting binary outcomes (0 or 1).

- It applies the sigmoid function to convert linear outputs into probabilities.
- A threshold (typically 0.5) is used to classify data points.
- The model is trained using gradient descent to minimize the log loss function.
- Performance is assessed using accuracy, precision, recall, F1-score, and confusion matrices.

# 1. Import necessary libraries and load the dataset

```
import pandas as pd

# Load the dataset
file_path = "/content/Loan_default.csv"
df = pd.read_csv(file_path)

# Display basic info
print(df.head()) # Show first few rows
print(df.info()) # Show dataset summary
```

# 2. Check for missing values and anomalies in the dataset.

											-	
<b>∓</b>		LoanID	Age	Income	Loan		Cre	editScore	Mont			
	0	I38PQUQS96	56	85994		50587		520		;	80	
	1	HPSK72WA7R	69	50432		124440		458			15	
	2	C10Z6DPJ8Y	46	84208		129188		451			26	
	3	V2KKSFM3UN	32	31713		44799		743			0	
	4	EY08JDHTZP	60	20437		9139		633			8	
		NumCreditLines		InterestRate		LoanTerm		DTIRatio			\	
	0		4		15.23		36	0.44	Bachelor's			
	1		1		4.81		60	0.68	Master's			
	2		3	21.17			24 0.31		Master's			
	3	3		7.07			24 0.23					
	4		4	6.51			48 0.73		Bachelor's			
					asMortgage HasDepend		lents LoanPurpose					
	0	Full-time		Divorced		Yes		Yes Other		her		
	1	Full-time		Married		No			No Other			
	2	Unemployed		Divorced		Yes			Yes	A	uto	
	3	Full-time		Married		No			No	No Business		
	4	Unemployed		Divorced		No			Yes	A	uto	
HasCoSigner Default												
	0	Yes		0								
	1	Yes		0								
	2	No		1								
	3	No		0								
	4	No		0								
	<pre><class 'pandas.core.frame.dataframe'=""></class></pre>											
RangeIndex: 255347 entries, 0 to 255346												
Data columns (total 18 columns):												

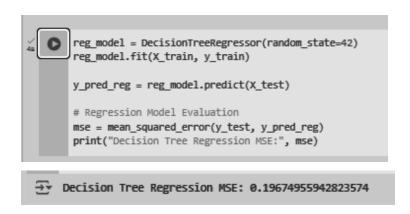
- a) Perform Logistic regression to find out relation between variables
- 3. Train the dataset using Linear Regression

```
+ Code + Text
25s D import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.preprocessing import LabelEncoder
        from sklearn.linear_model import LogisticRegression
        from sklearn.tree import DecisionTreeRegressor
        from sklearn.metrics import accuracy_score, classification_report, confusion_matrix, mean_squared_error
        # Load dataset
        file_path = "/content/Loan_default.csv"
        df = pd.read_csv(file_path)
        # Drop LoanID as it's just an identifier
        df.drop(columns=["LoanID"], inplace=True)
        # Encode categorical variables
        categorical_cols = ["Education", "EmploymentType", "MaritalStatus", "HasMortgage",
                            "HasDependents", "LoanPurpose", "HasCoSigner"]
        label_encoders = {}
        for col in categorical_cols:
           le = LabelEncoder()
            df[col] = le.fit_transform(df[col])
           label_encoders[col] = le
        # Split data into features and target
        X = df.drop(columns=["Default"])
        y = df["Default"]
        # 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=42, stratify=y)
        # (a) Perform Logistic Regression
        log_reg = LogisticRegression(max_iter=1000, random_state=42)
        log_reg.fit(X_train, y_train)
        y_pred = log_reg.predict(X_test)
        # Logistic Regression Evaluation
        print("Logistic Regression Accuracy:", accuracy_score(y_test, y_pred))
        print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
        print("Classification Report:\n", classification_report(y_test, y_pred))
```

- 4. Train the dataset using Logistic Regression and compute:
  - Accuracy Score
  - Precision, Recall, and F1-score

```
→ Logistic Regression Accuracy: 0.8854709222635598
   Confusion Matrix:
    [[45028 111]
[5738 193]]
    Classification Report:
                 precision recall f1-score support
                    0.89 1.00
0.63 0.03
                                        0.94
                                                 45139
              1
                                        0.06
                                                  5931
       accuracy
                                         0.89
                                                  51070
    macro avg 0.76
weighted avg 0.86
                               0.52
                                         0.50
                                                  51070
                                0.89
                                         0.84
                                                  51070
```

- b) Apply regression model technique to predict the data on the above dataset.
  - Find Mean Squared Error (MSE) and R-squared score.



```
from sklearn.metrics import r2_score

y_prob = log_reg.predict_proba(X_test)[:, 1]
r_squared = r2_score(y_test, y_prob)

print("R-squared value:", r_squared)
```

R-squared value: 0.3017

**Conclusion:** The logistic regression model achieves 88.54% accuracy, but its recall for loan defaulters (Class 1) is very low (0.03), indicating poor detection of actual defaults. The decision tree regression model has an MSE of 0.1967 and an R<sup>2</sup> score of 0.3017, showing moderate predictive power. Improving class balance and feature selection could enhance model performance.