Name:Kaushik Kotian Roll No.:30 Div:D15B Batch:B

EXPERIMENT: 05

AIM: Implementation of Regression Analysis.

PROBLEM STATEMENT:

1. Perform Logistic regression to find out the relation between variables
2. Apply the regression model technique to predict the data on the selected dataset.

THEORY:

Regression

Regression is a statistical method used to investigate the relationship between a dependent variable and one or more independent variables. The dependent variable is the variable that is being predicted or explained, while the independent variables are the variables that are used to make predictions about the dependent variable.

Regression analysis can be used in many fields, such as finance, economics, psychology, and engineering, to name a few. It is often used to identify relationships between variables, make predictions, and inform decision-making processes.

There are many types of regression analysis, each suited to different types of data and research questions. Some common types of regression analysis include:

- Linear regression: A method for modeling the relationship between a continuous dependent variable and one or more independent variables.
- Logistic regression: A method for modeling the relationship between a binary dependent variable and one or more independent variables.
- Polynomial regression: A method for modeling the relationship between a dependent variable and an independent variable that is best represented by a polynomial function.
- Ridge regression: A type of linear regression that uses regularization to prevent overfitting by adding a penalty term to the model.
- Lasso regression: A type of linear regression that also uses regularization to prevent overfitting, but with a different penalty term that can lead to a more sparse model.
- Stepwise regression: A method for selecting the most important independent variables to include in a regression model by sequentially adding or removing variables based on statistical tests.
- Multivariate regression: A method for modeling the relationship between a dependent variable and multiple independent variables.
- Poisson regression: A method for modeling the relationship between a count variable and one or more independent variables.

Linear Regression

Linear regression is a statistical technique used to model the relationship between a dependent variable and one or more independent variables. The relationship between the variables is assumed to be linear, meaning that the change in the dependent variable is proportional to the change in the independent variable(s).

Mathematically, we can represent a linear regression as:

$y= a0+a1x+ \varepsilon$

Here,

Y= Dependent Variable (Target Variable)

X= Independent Variable (predictor Variable)

a0= intercept of the line (Gives an additional degree of freedom)

a1 = Linear regression coefficient (scale factor to each input value).

ε = random error

The values for x and y variables are training datasets for Linear Regression model representation.

Linear regression can be further divided into two types of the algorithm:

- Simple Linear Regression:
  If a single independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Simple Linear Regression.
- Multiple Linear regression:
  If more than one independent variable is used to predict the value of a numerical dependent variable, then such a Linear Regression algorithm is called Multiple Linear Regression.

A regression line is a linear line showing the relationship between the dependent and independent variables. A regression line can show two types of relationship:

- Positive Linear Relationship:
  If the dependent variable increases on the Y-axis and independent variable increases on X-axis, then such a relationship is termed as a Positive linear relationship.

- Negative Linear Relationship:
  If the dependent variable decreases on the Y-axis and independent variable increases on the X-axis, then such a relationship is called a negative linear relationship.

Linear regression has many applications in various fields. Some examples include:

- Economics: Linear regression can be used to model the relationship between two economic variables, such as the relationship between inflation and interest rates.

- Finance: Linear regression can be used to predict stock prices based on historical data, as well as to model the relationship between financial variables such as interest rates, asset prices, and economic growth.
- Marketing: Linear regression can be used to model the relationship between advertising spending and sales, as well as to predict customer behavior based on demographic data.
- Healthcare: Linear regression can be used to model the relationship between health outcomes and patient characteristics, such as age, gender, and medical history.

Logistic Regression

- Logistic regression is a statistical technique used to model the relationship between a binary dependent variable and one or more independent variables. The binary dependent variable can only take on two possible values, such as yes or no, 0 or 1, or true or false.

- Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

- In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1).

- The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

- Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification. The below image is showing the logistic function:

Logistic Function (Sigmoid Function):
f(x) = 1 / (1 + e^(-x))

- The sigmoid function is a mathematical function used to map the predicted values to probabilities.
- It maps any real value into another value within a range of 0 and 1.
- The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the Sigmoid function or the logistic function.
- In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

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

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

Logistic regression has many applications in various fields. Some examples include:

- Medical research: Logistic regression can be used to model the probability of a patient developing a certain disease based on their demographic and health factors.
- Marketing: Logistic regression can be used to predict whether a customer is likely to buy a product based on their demographic and behavioral data.
- Finance: Logistic regression can be used to model the probability of a credit card user defaulting on their payments based on their credit history and other factors.
- Social sciences: Logistic regression can be used to model the probability of a person exhibiting a certain behavior or attitude based on their demographic and socioeconomic factors.
- Ecology: Logistic regression can be used to model the probability of a species being present in a certain area based on environmental factors such as temperature, precipitation, and vegetation.
- Sports: Logistic regression can be used to predict the outcome of a game based on the teams' previous performance, player stats, and other factors.
- Image recognition: Logistic regression can be used as a classification algorithm in image recognition tasks, such as identifying whether an image contains a certain object or not.

IMPLEMENTATION:

Loading dataset into Google collab.

```
[7]  import pandas as pd
     import numpy as np
     import matplotlib.pyplot as plt
     from sklearn.metrics import r2_score, mean_squared_error
     from math import sqrt
     from sklearn import metrics
```

```
df = pd.read_csv('/content/sample_data/loan_data_set.csv')
df.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
0	LP001002	Male	No	0	Graduate	No	5849	0.0	NaN	360.0	1.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y

Dropping Null values

```
[9] df.dropna(inplace=True)
```

```
df.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	LP001003	Male	Yes	1	Graduate	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0.0	141.0	360.0	1.0	Urban	Y
5	LP001011	Male	Yes	2	Graduate	Yes	5417	4196.0	267.0	360.0	1.0	Urban	Y

```
[11] data=df.replace('Graduate' ,0)
```

A] LOGISTIC REGRESSION

1. Setting the target variables in Binary format

```
data=df.replace('Graduate' ,0)
data=data.replace('Not Graduate',1)
data.head()
```

	Loan_ID	Gender	Married	Dependents	Education	Self_Employed	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History	Property_Area	Loan_Status
1	LP001003	Male	Yes	1	0	No	4583	1508.0	128.0	360.0	1.0	Rural	N
2	LP001005	Male	Yes	0	0	Yes	3000	0.0	66.0	360.0	1.0	Urban	Y
3	LP001006	Male	Yes	0	1	No	2583	2358.0	120.0	360.0	1.0	Urban	Y
4	LP001008	Male	No	0	0	No	6000	0.0	141.0	360.0	1.0	Urban	Y
5	LP001011	Male	Yes	2	0	Yes	5417	4196.0	267.0	360.0	1.0	Urban	Y

2. **Setting the dependent and independent variables, splitting the dataset in train and test set.**

```
[12] x = data[['ApplicantIncome',  'LoanAmount']]
     y = data['Loan_Amount_Term']
```

```
[14] from sklearn.model_selection import train_test_split
     x_train, x_test, y_train, y_test = train_test_split(x,y, random_state = 0)
```

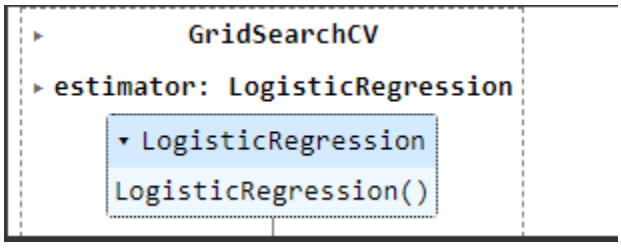
3. **Model Development**

```
[15] from sklearn.linear_model import LogisticRegression
     reg=LogisticRegression()
```

```
[16] from sklearn.model_selection import GridSearchCV
     parameter={'penalty': ['l1','l2', 'elasticnet'], 'C': [1,2,3,4,5,6, 10, 20, 30,40,50], 'max_iter': [100,200,300]}
```

```
[17] reg_model=GridSearchCV(reg,param_grid=parameter, scoring='accuracy', cv=5)
```

```
[20] reg_model.fit(x_train,y_train)
```



4. **Model Evaluation**

```
[21] predicted=reg_model.predict(x_test)
```

```
[22] from sklearn.metrics import accuracy_score, classification_report
```

```
[23] score=accuracy_score(predicted,y_test)
     print(score)

     0.8083333333333333
```

```
[24] df=pd.DataFrame(y_test,predicted)
```

```
df=pd.DataFrame({"Actual Term":y_test, 'Predicted Term':predicted})
print(df)
```

```
     Actual Term  Predicted Term
18         360.0           360.0
161        360.0           360.0
182        180.0           360.0
340        360.0           360.0
216        360.0           360.0
..           ...             ...
593        180.0           360.0
253        180.0           360.0
200        360.0           360.0
280        360.0           360.0
575         84.0           360.0

[120 rows x 2 columns]
```

5. **Prediction**

```
[27] feat = np.array([[1632540.0, 28641.0]])
     print("Predicted: {}".format(reg_model.predict(feat)))
     print("0 represents Graduate, 1 represents Not Graduate")
```

```
Predicted: [360.]
0 represents Graduate, 1 represents Not Graduate
```

B] LINEAR REGRESSION

1.Spliting dataset in features and target variable and accordingly splitting into train and test sets.

```
[33] x= data[['ApplicantIncome','Credit_History']]
     y= data[['LoanAmount']]
```

```
[34] from sklearn.model_selection import train_test_split
     x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.2, random_state=42)
```

```
[35] x_train.shape
```

```
(384, 2)
```

```
[36] x_test.shape
```

```
(96, 2)
```

2.Model Development

```
[37] from sklearn.linear_model import LinearRegression
     model = LinearRegression()
     model.fit(x_train, y_train)
```

```
▾ LinearRegression
LinearRegression()
```

3.Calculating the learned coefficients and intercept of the linear model

```
[38] print(model.coef_)
     print(model.intercept_)
```

```
[[ 0.00608366 -0.27263999]]
[110.9335726]
```

4. Model prediction

```
[39] y_pred = model.predict(x_test)
```

```
[40] y_pred
```

```
array([[130.57273729],
       [151.92636833],
       [132.01456366],
       [129.1845393 ],
       [130.63357384],
       [124.44537162],
       [129.41684273],
       [123.33318716],
       [122.68831967],
       [130.93775662],
       [130.12863043],
```

5. Converting y_test and y_pred to 1-D array

```
[41] Y_pred=y_pred.ravel()
```

```
[42] Y_pred
```

```
array([130.57273729, 151.92636833, 132.01456366, 129.1845393 ,
       130.63357384, 124.44537162, 129.41684273, 123.33318716,
       122.68831967, 130.93775662, 130.12863043, 139.55829656,
       143.92636126, 138.43890393, 132.56209266, 128.65030213,
       137.09329154, 123.94042821, 145.02750291, 199.98724731,
       161.35603446, 133.9795844 , 145.30735107, 125.87007153,
       136.212286  , 130.32330741, 137.14916895, 148.01953694,
       144.04803437, 138.54232608, 129.87920055, 121.93394638,
       222.19259013, 130.93775662, 161.35603446, 166.42371954,
       166.4298032 , 135.05030778, 183.66479942, 126.75107708,
       126.87995836, 146.06172436, 173.97961976, 199.37888175,
       136.39479566, 124.00734842, 123.83204692, 125.77273304,
       126.47843709, 130.20050978, 114.58489046, 133.47464099,
       136.7050621 , 172.34311641, 129.52026487, 129.96437173,
       129.22824941, 138.54232608, 159.76820035, 171.49748829,
       134.13775945, 201.91576612, 136.13928213, 144.49101671,
       152.48606464, 122.78565816, 124.36740861, 125.87007153,
       148.68377991, 134.95296929, 128.50925354, 121.36208276,
       135.57350216, 183.66479942, 177.58114385, 148.68377991,
       133.27388036, 128.30353376, 146.76742841, 132.68376577,
       149.69975039, 140.39175737, 161.35603446, 208.72946036,
       168.82067984, 127.4214037 , 134.18642869, 130.17729967,
       138.96209831, 128.15752603, 132.79214706, 134.55144803,
       122.91341493, 145.19784527, 131.9537271 , 132.6655148 ])
```

```
[43] Y_pred.shape
```

```
(96,)
```

```
[44] Y_test = y_test['LoanAmount'].values
     y_test
```

	LoanAmount
92	81.0
529	130.0
505	243.0
358	100.0
512	148.0
...	...
281	112.0
299	113.0
522	100.0
33	114.0
537	107.0

96 rows × 1 columns

6.Displaying the the actual value and predicted value using a Dataframe.

```
[45] Y_test=Y_test.ravel()

     df=pd.DataFrame(Y_test, Y_pred)
     df=pd.DataFrame({"Actual LoanAmount":Y_test, 'Predicted LoanAmount':Y_pred})
     print(df)

     Actual LoanAmount  Predicted LoanAmount
0                 81.0           130.572737
1                130.0           151.926368
2                243.0           132.014564
3                100.0           129.184539
4                148.0           130.633574
..                 ...                  ...
91               112.0           134.551448
92               113.0           122.913415
93               100.0           145.197845
94               114.0           131.953727
95               107.0           132.665515

[96 rows x 2 columns]
```

7. Calculating the accuracy score

```
[47] from sklearn.metrics import confusion_matrix, accuracy_score
     model.score(x_test, Y_test)*100

     27.00700397153979
```

8. Calculating the predicted monthly debt for Annual income and Current credit balance entered.

```
[48] trial = np.array([[1184194.0,122170.0]])
     print("New Predicted: {}".format(model.predict(trial)))

     New Predicted: [[-25993.26542687]]
```

CONCLUSION:

In this experiment we studied our test dataset thoroughly with the help of Python libraries. Then we used linear regression for establishing the strength of the relationship between two variables. We also used logistic regression for predicting binomial categorical classification.