**EXPERIMENT 3**

**AIM:** Implementation of Linear Regression

1. Single Variate

2. Multi variate

**THEORY:**

Linear Regression was developed in the field of statistics and is studied as a model for understanding the relationship between input and output numerical variables, but it has been borrowed by machine learning. It is both a statistical algorithm and a machine learning algorithm.

In linear regression, the relationships are modelled using linear predictor functions whose unknown model parameters are estimated from the data. Such models are called linear models.

The representation is a linear equation that combines a specific set of input values (X) the solution to which is the predicted output for that set of input values (y). As such, both the input values (X) and the output value are numeric. The linear equation assigns one scale factor to each input value or column, called a **coefficient,** and represented by the capital Greek letter Beta (B). One additional coefficient is also added, giving the line an additional degree of freedom (e.g. moving up and down on a two-dimensional plot) and is often called the intercept or the **bias coefficient**.

For example, in a simple regression problem (for X and y), the form of the model would be:

y = B0 + B1\*x … univariate

y = B0 + B1\*x1 + B2\*x2 +……+ Bn\*xn …. multivariate

**PART A: Univariate Salary Dataset**

Salary Dataset is a univariate dataset having 2 columns 'YearsExperience' and Salary. I have trained a Linear Regression model that predicts the Salary based on the Years of Experience of the Employee.

**CODE:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

%matplotlib inline

from sklearn.linear\_model import LinearRegression

from google.colab import files

from sklearn.metrics import mean\_squared\_error, r2\_score

from google.colab import files

import io

uploaded = files.upload()

# Load the diabetes dataset

salary\_data = pd.read\_csv(io.BytesIO(uploaded['Salary\_Data.csv']))

salary\_data.head()

sal\_X = salary\_data[['YearsExperience']]

sal\_y = salary\_data.Salary

# Test Train Split

from sklearn.model\_selection import train\_test\_split

X\_train , X\_test , y\_train , y\_test = train\_test\_split(sal\_X , sal\_y, test\_size = 0.3 , random\_state= 2 )

# Create linear regression object

sal\_reg = LinearRegression()

# Train the model using the training sets

sal\_reg.fit(X\_train, y\_train)

# PREDICT UNSEEN

y\_pred = sal\_reg.predict(X\_test)

print('PREDICT UNSEEN SAMPLES : \n' , y\_pred)

# Coefficients

print('\nCoefficient (C0): ', sal\_reg.coef\_)

# MSE

print('Mean squared error(MSE) : %.2f' % mean\_squared\_error(y\_test, y\_pred))

# R2 Score

print('R2 SCORE : %.2f' % r2\_score(y\_test, y\_pred))

# Plot outputs

plt.scatter(X\_test, y\_test, color='green')

plt.plot(X\_test, y\_pred, color='blue', linewidth=3)

plt.title("Salary vs Experience")

plt.xlabel("Years of Experience")

plt.ylabel("Salary")

plt.xticks(())

plt.yticks(())

plt.show()

**OUTPUT:**

DataSet

Table

Description automatically generated with medium confidence

Text

Description automatically generated

PLOT Linear Regression Model

Chart, scatter chart

Description automatically generated

**PART B: Multivariate Dataset**

**Melbourne Housing Prices**

This is a multivariate dataset that depends on attributes like Area, Number of Rooms, Real Estate Agent, Distance, Council Area, Type of House and other features. I have implemented a Linear Regression Model along with data pre-processing before it.

**CODE:**

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

import seaborn as sns

from sklearn.model\_selection import train\_test\_split

dataset = pd.read\_csv('./Melbourne\_housing\_FULL.csv')

dataset.head()

# DATA CLEANING

cols\_to\_use = ['Suburb' , 'Rooms' , 'Type' , 'Method' , 'SellerG' , 'Regionname' , 'Propertycount' , 'Distance' , 'CouncilArea' , 'Bedroom2' ,'Bathroom' , 'Car' , 'Landsize' , 'BuildingArea' , 'Price' ]

dataset = dataset[cols\_to\_use]

# N.A values in the dataset

dataset.isna().sum()

# Fill with Zero

cols\_to\_fill\_zero = ['Propertycount' , 'Distance' , 'Bedroom2' , 'Bathroom' , 'Car']

dataset[cols\_to\_fill\_zero] = dataset[cols\_to\_fill\_zero].fillna(0)

# Fill with mean

dataset['Landsize'] = dataset['Landsize'].fillna(dataset.Landsize.mean())

dataset['BuildingArea'] = dataset['BuildingArea'].fillna(dataset.BuildingArea.mean())

# Drop the remaining

dataset.dropna(inplace=True)

dataset = pd.get\_dummies(dataset , drop\_first = True)

# X and Y data columns

X = dataset.drop('Price' , axis=1)

Y = dataset['Price']

train\_X , test\_X , train\_Y , test\_Y = train\_test\_split(X , Y, test\_size = 0.3 , random\_state= 2 )

# LINEAR REGRESSION

from sklearn.linear\_model import LinearRegression

from sklearn.metrics import mean\_squared\_error, r2\_score

lr = LinearRegression().fit(train\_X ,train\_Y)

lr\_r2 = lr.score(test\_X , test\_Y)

# PREDICT UNSEEN

y\_pred = lr.predict(test\_X)

print('PREDICT UNSEEN SAMPLES(First 20) : \n' , y\_pred[:20])

# Coefficients

# print('\nCoefficient (C0): ', lr.coef\_)

# MSE

print('\nMean squared error(MSE) : %.2f' % mean\_squared\_error(test\_Y, y\_pred))

# R2 Score

print('R2 SCORE : %.2f' % lr\_r2)

# Graph Actual Values VS Predicted Values

f =plt.figure(figsize= (6,6))

ax = plt.axes()

ax.plot(test\_Y , y\_pred , marker='o' , ls='' , ms=3.0 , label= 'Linear')

leg = plt.legend(frameon =True)

leg.get\_frame().set\_edgecolor('black')

leg.get\_frame().set\_linewidth(1.0)

lim = (0 , test\_Y.max())

ax.set(xlabel = 'Actual Price', ylabel='Predicted Value' , xlim =lim , ylim =lim , title= 'Housing Price Predictions Model')

**OUTPUT:**

**Load Melbourne Housing Dataset**

A picture containing calendar

Description automatically generated

**DATA Preprocessing and Cleaning (isNA , isNull)**

Before After

Table

Description automatically generated with low confidence A picture containing table

Description automatically generated

Text

Description automatically generated

**PLOT Actual vs Predicted Prices**

Chart, scatter chart

Description automatically generated

**CONCLUSION:** In this experiment I implemented Linear Regression on Univariate as well as Multivariate dataset. I have predicted the salary for unseen samples and also plotted the Linear Regression for the Salary Dataset. In Salary Dataset the linear model had an accuracy of 90%. In the Melbourne Dataset, I performed data pre-processing by filtering out the superfluous columns which does not affect the pricing. Then I performed data cleaning by removing the null values and filling the N/A values by the mean. I have also predicted the house prices for unseen samples and displayed the first 20 values. Finally, the Melbourne House Prediction linear model had an accuracy of 74%.