Experiment No. 4

NAME: PARTH PAREKH **SAP ID:60004200006**
BATCH:A1 **BRANCH:COMPUTER ENGINEERING**

Aim:

Implementation of Linear Regression for Single Variate and Multi-VariateX

Perform the experiment in Python.

Read any dataset from UCI dataset repository

Part A:

Program Single variate using inbuilt functions.

Predict for unseen samples

Plot the regression

Part B:

Program Multi variate using inbuilt functions.

Predict for unseen samples

Theory:

- It is simplest form of regression. Linear regression attempts to model the relationship between two variables by fitting a linear equation to observe the data.
- Linear regression attempts to find the mathematical relationship between variables.
- If outcome is straight line then it is considered as linear model and if it is curved line, then it is a non linear model.
- The relationship between dependent variable is given by straight line and it has only one independent variable.
- $Y = \alpha + B X$
- Model 'Y', is a linear function of 'X'.
- The value of 'Y' increases or decreases in linear manner according to which the value of 'X' also changes.
- The best-fit line is achieved using the Least-Squared method. This method minimizes the sum of the squares of the deviations from each of the data points to the regression line. The negative and positive distances do not get cancelled out here as all the deviations are squared. There are also divisions under linear regression in data mining named simple regression and multiple regression. Simple linear regression is where a singular predictor variable is known. However, in most real-world cases, the number of predictor variables is more than one, which is why multiple Regression data mining is used more than the simple one

Single Variate Linear regression

In statistics, simple linear regression is a linear regression model with a single explanatory variable. That is, it concerns two-dimensional sample points with one independent variable and one dependent variable (conventionally, the x and y coordinates in a Cartesian coordinate system) and finds a linear function (a non-vertical straight line) that, as accurately as possible, predicts the dependent variable values as a function of the independent variable. The adjective simple refers to the fact that the outcome variable is related to a single predictor.

It is common to make the additional stipulation that the ordinary least squares (OLS) method should be used: the accuracy of each predicted value is measured by its squared residual (vertical distance between the point of the data set and the fitted line), and the goal is to make the sum of these squared deviations as small as possible. Other regression methods that can be used in place of ordinary least squares include least absolute deviations (minimizing the sum of absolute values of residuals) and the Theil–Sen estimator (which chooses a line whose slope is the median of the slopes determined by pairs of sample points). Deming regression (total least squares) also finds a line that fits a set of two-dimensional sample points, but (unlike ordinary least squares, least absolute deviations, and median slope regression) it is not really an instance of simple linear regression, because it does not separate the coordinates into one dependent and one independent variable and could potentially return a vertical line as its fit.

Multi Variate Linear regression

Multivariate Regression is a supervised machine learning algorithm involving multiple data variables for analysis. A Multivariate regression is an extension of multiple regression with one dependent variable and multiple independent variables. Based on the number of independent variables, we try to predict the output.

Multivariate regression tries to find out a formula that can explain how factors in variables respond simultaneously to changes in others.

Code:

Single Variate

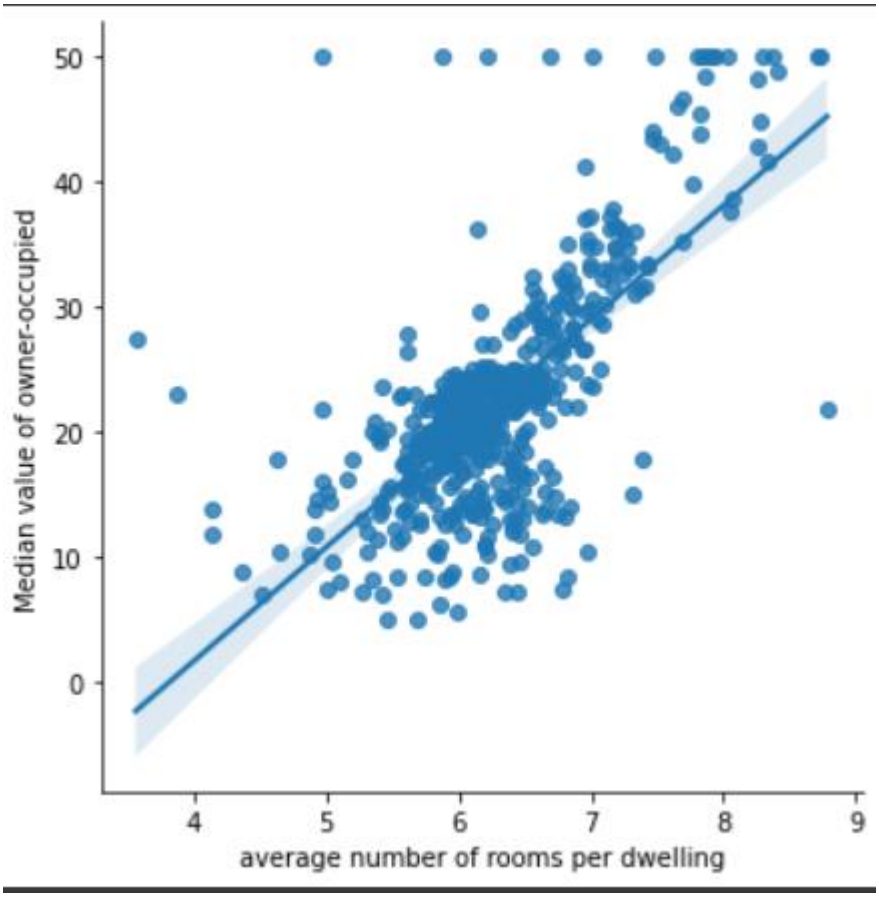
```
from sklearn.datasets import load_boston
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
%matplotlib inline
df = pd.read_csv('bostonhousing.csv')
df.head()
X = pd.DataFrame(df,columns=['rm'])
y = df['medv']
sns.lmplot(x='rm', y='medv',data=df)
plt.xlabel('Average number of rooms per dwelling')
plt.ylabel('Median value of owner-occupied')
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta
te=101)
lm = LinearRegression()
lm.fit(X_train, y_train )
predictions = lm.predict(X_test)
print(lm.coef_)
plt.scatter(X_test,y_test,color='b')
plt.plot(X_test,predictions,color='k')
plt.show()
print('MAE: {}'.format(metrics.mean_absolute_error(y_test, predictions)))
print('MSE: {}'.format(metrics.mean_squared_error(y_test, predictions)))
print('RMSE: {}'.format(np.sqrt(metrics.mean_squared_error(y_test, predictions))))
```

```
print(lm.score(X_test,y_test))
```

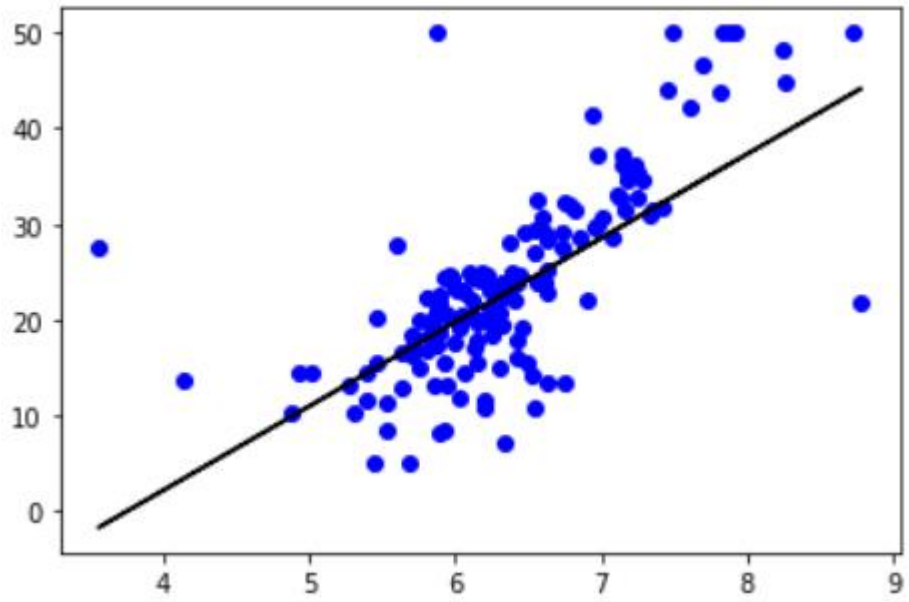
OUTPUT:
Dataset five rows

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	b	lstat	medv
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2

Before dataset split



After dataset split



MAE: 4.75614486976047
MSE: 48.11240589160621
RMSE: 6.936310683036495
Lm score: 0.5152898496246172

Code:
Multi-Variate

```
from sklearn.datasets import load_boston
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn import metrics
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import numpy as np
%matplotlib inline
df = pd.read_csv('bostonhousing.csv')
df.head()
X = pd.DataFrame(df,columns=['rm','age','tax','crim'])
y = df['medv']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_sta
te=101)
lm = LinearRegression()
lm.fit(X_train, y_train )
predictions = lm.predict(X_test)
print(lm.coef_)
print('MAE: {}'.format(metrics.mean_absolute_error(y_test, predictions)))
print('MSE: {}'.format(metrics.mean_squared_error(y_test, predictions)))
print('RMSE: {}'.format(np.sqrt(metrics.mean_squared_error(y_test, predictions))))
print(lm.score(X_test,y_test))
```

OUTPUT:
MAE: 4.030100871526964
MSE: 39.12720506943922
RMSE: 6.255174263714738
Lm score:0.6058115760059078

Conclusion:
Linear Regression is one of the most trivial machine algorithms. Interpretability and easy-to-train traits make this algorithm the first steps in Machine Learning. Being a little less complicated, Linear Regression acts as one of the fundamental concepts in understanding higher and complex algorithms.
We found that MAE for RM is better than Age and Tax with Medv in single variate analysis
Also combination of Rm ,tax ,age ,crim is better performing than Induz, zn, b , lstat in Multi-variate analysis in terms of MAE.
Some problems faced with linear regression are:
1. Non-Linearity of the response-predictor relationships
2. Correlation of error terms
3. A non-constant variance of the error term [Heteroscedasticity]
4. Collinearity
5. Outliers and High Leverage Points