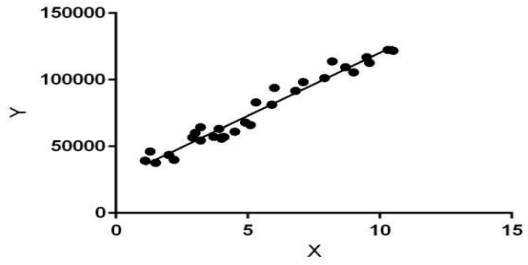
Assignment No: 4

Aim/Problem Statement:- Data Analytics I

Create a Linear Regression Model using Python/R to predict home prices using Boston Housing Dataset (https://www.kaggle.com/c/boston-housing). The Boston Housing dataset contains information about various houses in Boston through different parameters. There are 506 samples and 14 feature variables in this dataset.
The objective is to predict the value of prices of the house using the given features.

Theory:-

Linear Regression: It is a machine learning algorithm based on **supervised learning**. It performs a **regression task**. Regression models a target prediction value based on independent variables. It is mostly used for finding out the relationship between variables and forecasting. Different regression models differ based on – the kind of relationship between dependent and independent variables, they are considering and the number of independent variables being used.



Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x). So, this regression technique finds out a linear relationship between x (input) and y(output). Hence, the name is Linear Regression.

In the figure above, X (input) is the work experience and Y (output) is the salary of a person. The regression line is the best fit line for our model.

Hypothesis function for Linear Regression :

$$y = \theta_1 + \theta_2 . x$$

While training the model we are given :
x: input training data (univariate – one input variable(parameter))
y: labels to data (supervised learning)

When training the model – it fits the best line to predict the value of y for a given value of x. The model gets the best regression fit line by finding the best θ_1 and θ_2 values.
θ_1: intercept
θ_2: coefficient of x
Once we find the best θ_1 and θ_2 values, we get the best fit line. So when we are finally using our model for prediction, it will predict the value of y for the
input value of x.

How to update θ_1 and θ_2 values to get the best fit line ?
Cost Function (J):
By achieving the best-fit regression line, the model aims to predict y value such that the error difference between predicted value and true value is minimum. So, it is very important to update the θ_1 and θ_2 values, to reach the best value that minimize the error between predicted y value (pred) and true y value (y).

$$minimize \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

$$J = \frac{1}{n}\sum_{i=1}^{n}(pred_i - y_i)^2$$

Cost function(J) of Linear Regression is the **Root Mean Squared Error (RMSE)** between predicted y value (pred) and true y value (y).
Gradient Descent:
To update θ_1 and θ_2 values in order to reduce Cost function (minimizing RMSE value) and achieving the best fit line the model uses Gradient Descent. The idea is to start with random θ_1 and θ_2 values and then iteratively updating the values, reaching minimum cost.

Algorithm:-

Step 1: Download the data set of Boston Housing Prices
(https://www.kaggle.com/c/boston-housing).
Step 2: Importing Libraries
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```
Step 3: Importing Data
```
from sklearn.datasets import load_boston
boston = load_boston()
```
Step 4: Converting data from nd-array to data frame and adding feature names to the data
```
data = pd.DataFrame(boston.data)
data.columns = boston.feature_names
```
Step 5: Adding 'Price' (target) column to the data

```
data['Price'] = boston.target
```

Step 6: Getting input and output data and further splitting data to training and testing dataset.

```
# Input Data
x = boston.data

# Output Data
y = boston.target
```

Step 7: splitting data to training and testing dataset.

```
#from sklearn.cross_validation import train_test_split
#the submodule cross_validation is renamed and reprecated to model_selection
from sklearn.model_selection import train_test_split

xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size =0.2,random_state = 0)
```

Step 8: #Applying Linear Regression Model to the dataset and predicting the prices.

```
# Fitting Multi Linear regression model to training model
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(xtrain, ytrain)

# predicting the test set results
y_pred = regressor.predict(xtest)
```

Step 9: Plotting Scatter graph to show the prediction

```
# results - 'ytrue' value vs 'y_pred' value
plt.scatter(ytest, y_pred, c = 'green')
plt.xlabel("Price: in $1000's")
plt.ylabel("Predicted value")
plt.title("True value vs predicted value : Linear Regression")
plt.show()
```

Step 10: Results of Linear Regression.

```
from sklearn.metrics import mean_squared_error
mse = mean_squared_error(ytest, y_pred)
print("Mean Square Error : ", mse)
```

Input: Dataset of Boston Housing Prices. This dataset concerns the housing prices in the housing city of Boston. The dataset provided has 506 instances with 14 features.

Output: Prediction of Boston Housing Prices by plotting graph to show prediction

Conclusion:-Housing Prices of Boston city predicted using Linear Regression

Questions:

1) What is Linear regression
2) What are different types of linear regressions
3) Applications where linear regression is used
4) What are the limitations of linear regression