**Assignment No :- 04**

**Title :-**

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 feature.

**Theory :-**

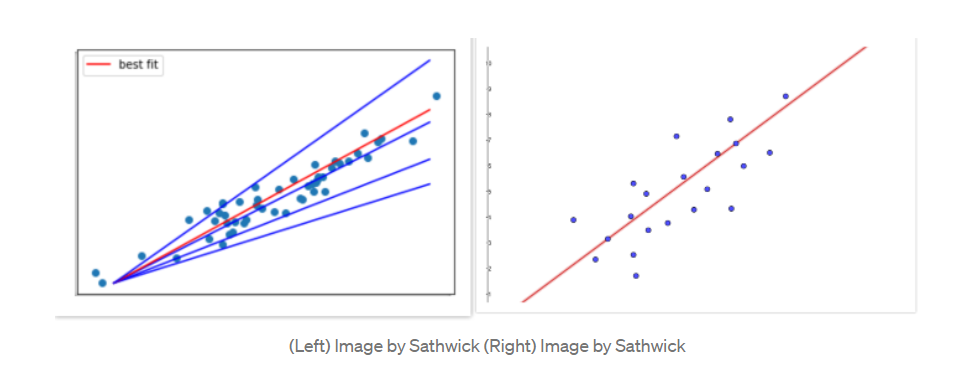
**What is a Linear Regression?**

It tries to find out the best possible linear relationship between the input features and the target variable(y).

That’s it! This is what Linear Regression does. Pretty simple right?

In machine learning jargon the above can be stated as “It is a supervised machine learning algorithm that best fits the data which has the target variable (dependent variable) as a linear combination of the input features(independent variables). ”

**How to visualise linear regression?**



For now, when you think of linear regression think of fitting a line such that the distance between the data points and the line is minimum. As shown above, the red line best fits that data than the other blue lines.

The linear relation between the input features and the output in 2D is simply a line.

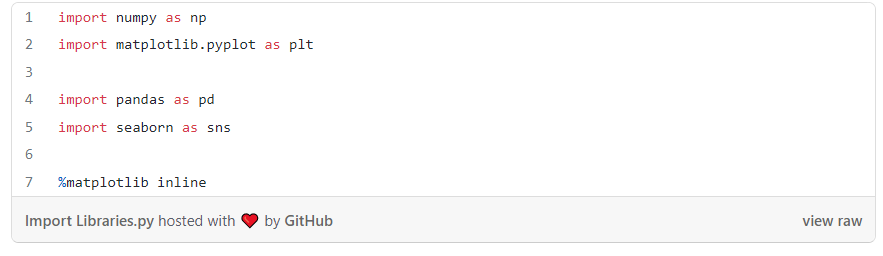
Yes! The linear regression tries to find out the best linear relationship between the input and output.

y = θx + b # Linear Equation

The goal of the linear regression is to find the best values for θ and b that represents the given data.

let’s get started.

First, we will import the required libraries.



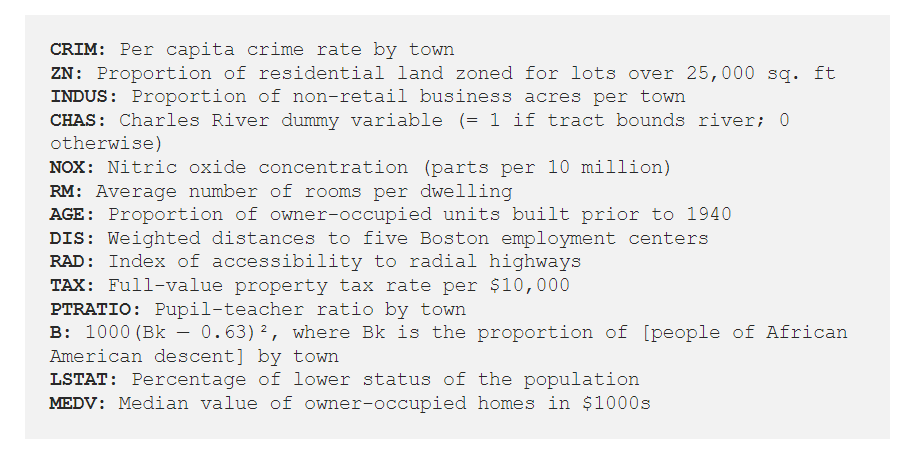
Next, we will load the housing data from the scikit-learn library and understand it.

We print the value of the boston\_dataset to understand what it contains. print(boston\_dataset.keys()) gives

dict\_keys(['data', 'target', 'feature\_names', 'DESCR'])

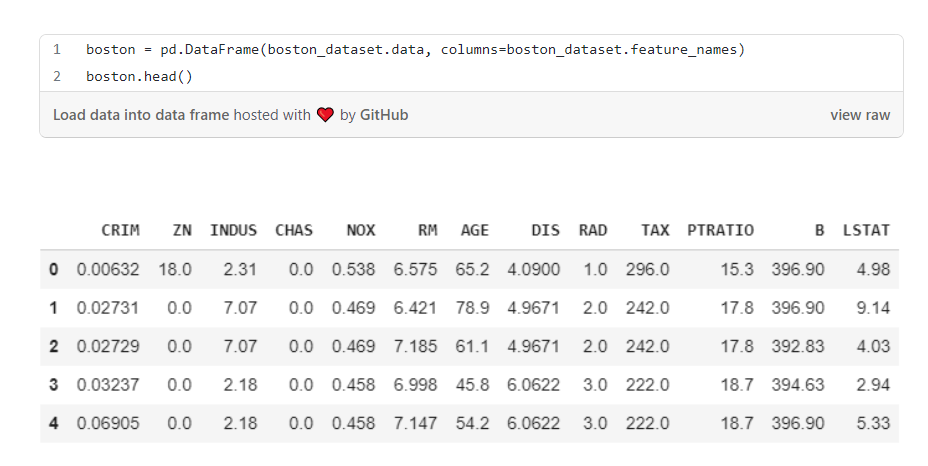
* *data*: contains the information for various houses
* *target*: prices of the house
* *feature\_names*: names of the features
* *DESCR*: describes the dataset

To know more about the features use boston\_dataset.DESCR The description of all the features is given below:



The prices of the house indicated by the variable MEDV is our **target variable** and the remaining are the **feature variables** based on which we will predict the value of a house.

We will now load the data into a pandas dataframe using pd.DataFrame. We then print the first 5 rows of the data using head()



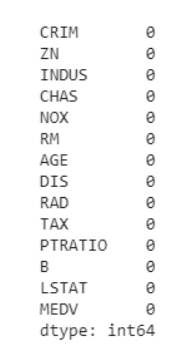
We can see that the target value MEDV is missing from the data. We create a new column of target values and add it to the dataframe.

boston['MEDV'] = boston\_dataset.target

**Data preprocessing**

After loading the data, it’s a good practice to see if there are any missing values in the data. We count the number of missing values for each feature using isnull()

boston.isnull().sum()

However, there are no missing values in this dataset as shown below. 

## ****Exploratory Data Analysis****

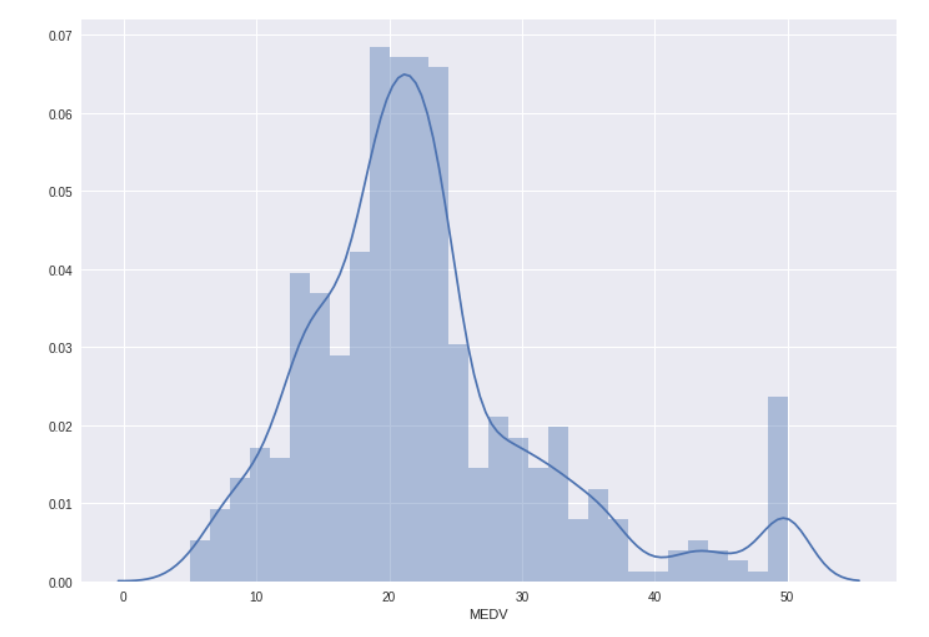
Exploratory Data Analysis is a very important step before training the model. In this section, we will use some visualizations to understand the relationship of the target variable with other features.

Let’s first plot the distribution of the target variable MEDV. We will use the distplot function from the seaborn library.

sns.set(rc={'figure.figsize':(11.7,8.27)})

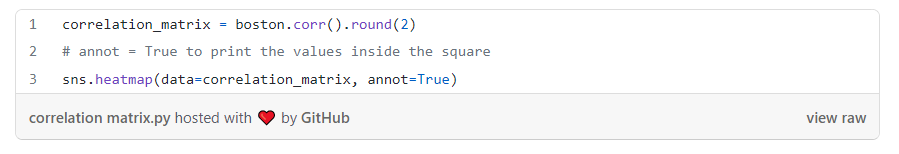
sns.distplot(boston['MEDV'], bins=30)

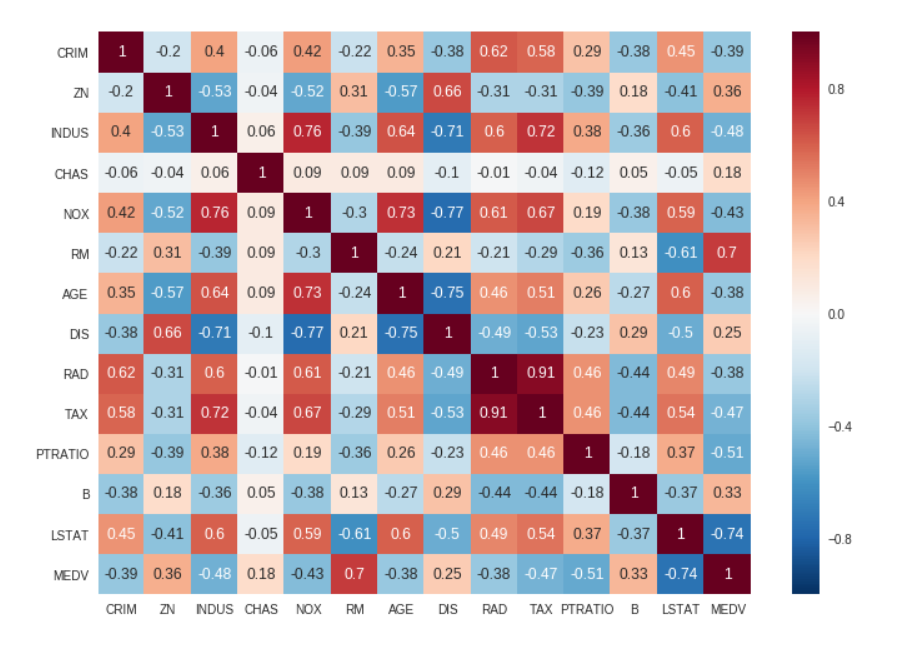
plt.show()



We see that the values of MEDV are distributed normally with few outliers.

Next, we create a correlation matrix that measures the linear relationships between the variables. The correlation matrix can be formed by using the corr function from the pandas dataframe library. We will use the heatmap function from the seaborn library to plot the correlation matrix.

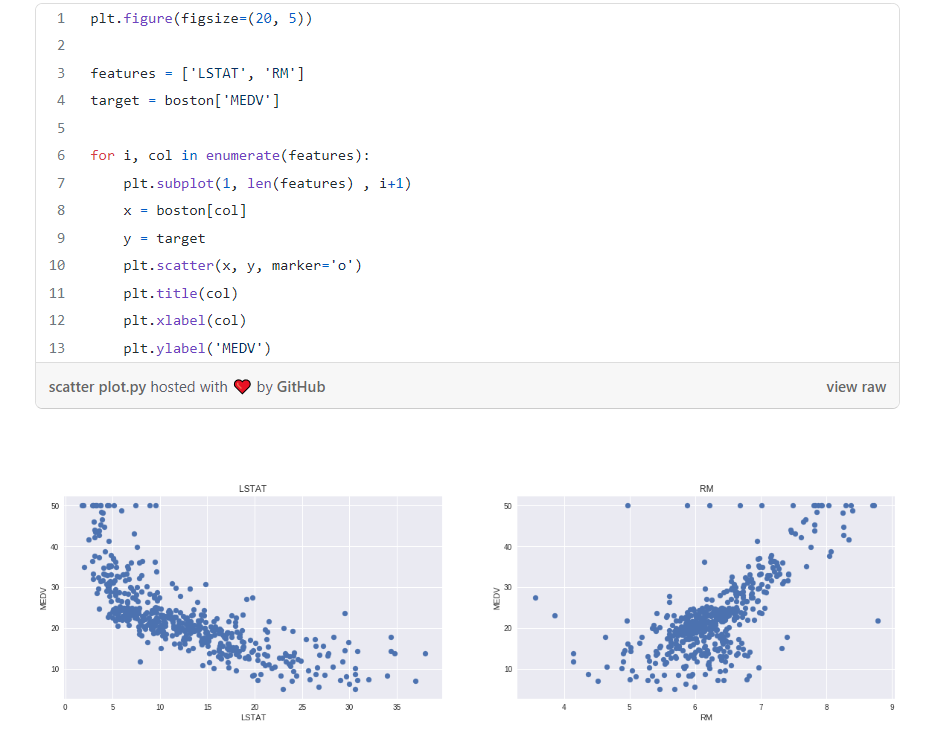




The correlation coefficient ranges from -1 to 1. If the value is close to 1, it means that there is a strong positive correlation between the two variables. When it is close to -1, the variables have a strong negative correlation.

## ****Observations:****

* To fit a linear regression model, we select those features which have a high correlation with our target variable MEDV. By looking at the correlation matrix we can see that RM has a strong positive correlation with MEDV (0.7) where as LSTAT has a high negative correlation with MEDV(-0.74).
* An important point in selecting features for a linear regression model is to check for multi-co-linearity. The features RAD, TAX have a correlation of 0.91. These feature pairs are strongly correlated to each other. We should not select both these features together for training the model. Check [this](https://stats.stackexchange.com/a/1150) for an explanation. Same goes for the features DIS and AGE which have a correlation of -0.75.
* Based on the above observations we will RM and LSTAT as our features. Using a scatter plot let’s see how these features vary with MEDV.

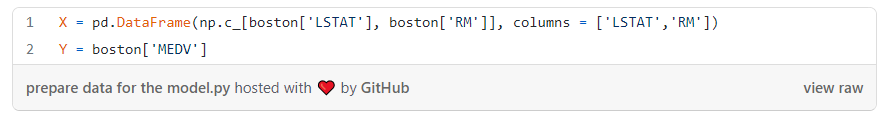


## ****Observations:****

* The prices increase as the value of RM increases linearly. There are few outliers and the data seems to be capped at 50.
* The prices tend to decrease with an increase in LSTAT. Though it doesn’t look to be following exactly a linear line.

## ****Preparing the data for training the model****

We concatenate the LSTAT and RM columns using np.c\_ provided by the numpy library.



## ****Splitting the data into training and testing sets****

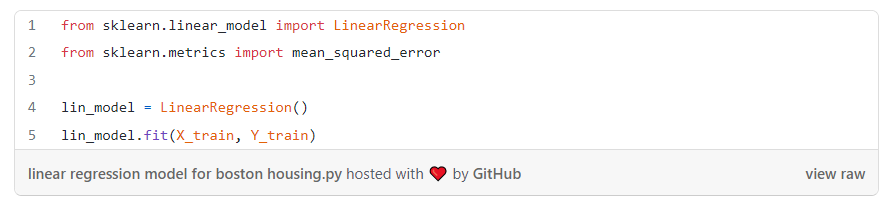
Next, we split the data into training and testing sets. We train the model with 80% of the samples and test with the remaining 20%. We do this to assess the model’s performance on unseen data. To split the data we use train\_test\_split function provided by scikit-learn library. We finally print the sizes of our training and test set to verify if the splitting has occurred properly.



(404, 2)   
(102, 2)  
(404,)  
(102,)

## ****Training and testing the model****

We use scikit-learn’s LinearRegression to train our model on both the training and test sets.

**Model evaluation**

We will evaluate our model using RMSE and R2-score.



**The model performance for training set**   
--------------------------------------   
RMSE is 5.6371293350711955   
R2 score is 0.6300745149331701 **The model performance for testing set**   
--------------------------------------   
RMSE is 5.137400784702911  
R2 score is 0.6628996975186952

## ****Conclusion****

In this experiment, we applied the concepts of linear regression on the Boston housing dataset.