**Linear Regression:**

The boston dataset is imported from the sklearn.datasets. This dataset has its own inbuilt functions and data members.

* A scatterplot of the target feature, which is the prices, is plotted using the matplotlib.pyplot module.
* Then, the whole of the Boston dataset is converted to a dataframe. Then, a lmpot is done with the number of rooms per dwelling as x axis and the cost as y axis. the lmplot automatically fits a linear regression model for the given scatter plot.
* The method called the sum of least squares, is that the best fit line equation for the given scatter plot is found out by finding the parameters of the line equation for which the sum of the squares of the distance between the y coordinate of the line and the actual y value for the same x value is least.
* Seaborn library automatically does this. But we can also find out the best fit using the numpy library:
* The series is first converted to a 2 dimensional array using the ‘vstack()’ in numpy.
* For every row in the array X, a new column with value 1 is created.
* Now , using the ‘np.linalg.lstsq()’ we find  the values of m,b by supplying the matrix X and Y(Y is the array containing all the prices, the target feature).Now, we can plot the best fit line whose equation is y=m\*x + b
* Now, we plot this on top of the scatter plot using the plot function in plt. The total error is found out as the value in index 1 of the list object returned by the ‘np.linalg.lstsq()’ method.Through this we can find the root mean square error
* A Linear Regression object is created after importing The sklearn module and the sklearn.linear\_model.
* The training data consists of the training values and the target data consists of the target values.Using the ‘.fit()’ method , the linear fit is modelled.
* The coefficients of the fitted line can be obtained using the ‘.coef\_’ attribute and §  the intercept coefficient can be obtained using the ‘.intercept\_’ attribute of the Linear Regression object.
* In order to predict the prices, we need to split our available dataset into a training set and a testing set. This is done using the ‘sklearn.model\_selection.train\_test\_split()’ function. This method takes in the data used for prediction and the the target variable data as X and Y. It can also have an argument as the fraction of the given data to be split into training/testing set.
* The function also  gives four different indexable objects as output that are stored as X\_train,X\_test,Y\_train and Y\_test respectively.X\_train and Y\_train are used in the ‘.fit()’ method to model the fitted line
* To predict the target set by returning a predicted set and takes in the dataset as input, we use the method ‘.predict()’
* The error is the difference between this predicted set and the target set. Sum of the  squares of all these errors gives us the root mean square error. The coefficient of determination is also measure of the accuracy of the model. It is the square of the correlation coefficient between the predicted and the target random variables.
* Then, a residual plot is plotted to check for the residual error pattern. A residual data point is the difference between the predicted data point to the corresponding target data point. It has the residuals in the y axis and the predicted points as the x values.
* If the scatter plot centred on y=0 with no detectable patterns, it is a proper model.
* If the errors are due to complete randomness of the real world and there are no deterministic variables that have been missed while making the predictive model,it means that the scatter plot is completely stochastic