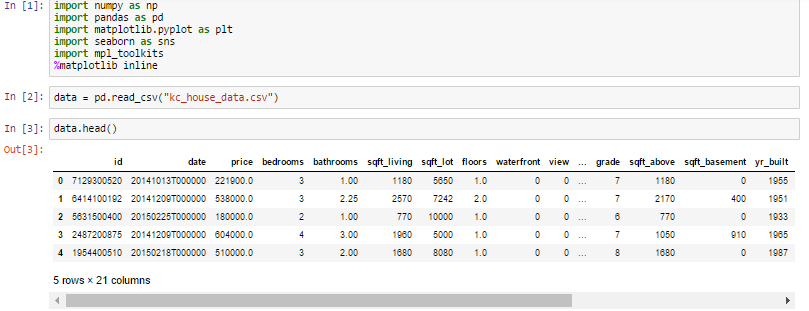
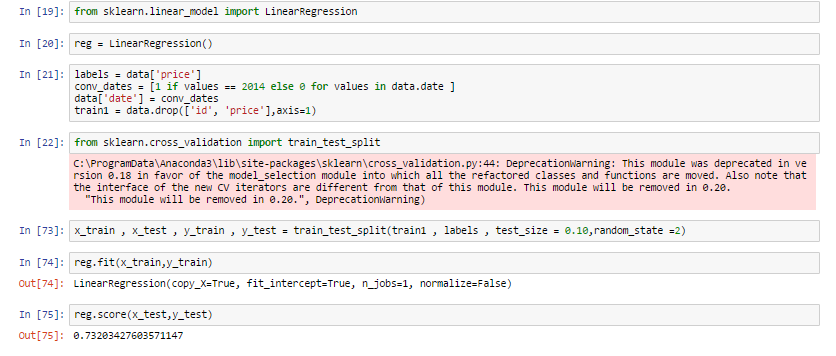
Create a model to predict house prices using Python

we import our libraries and dataset and then we see the head of the data to know how the data looks like and use describe function to see the percentile’s and other key statistics.



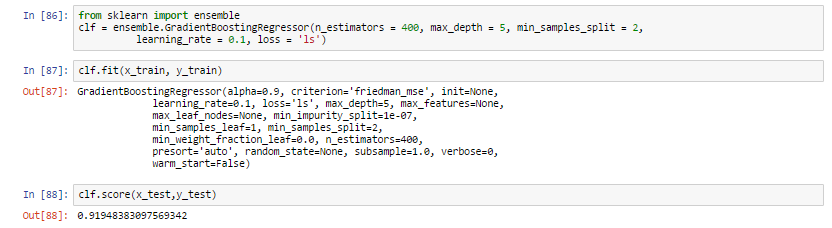
Linear regression on the data to predict prices:



1. We import our dependencies, for linear regression and import linear regression from it.
2. We then initialize Linear Regression to a variable reg.
3. Now we know that prices are to be predicted , hence we set labels (output) as price columns and we also convert dates to 1’s and 0’s so that it doesn’t influence our data much . We use 0 for houses which are new that is built after 2014.
4. We again import another dependency to split our data into train and test.
5. I’ve made my train data as 90% and 10% of the data to be my test data , and randomized the splitting of data by using random\_state.
6. So now , we have train data , test data and labels for both let us fit our train and test data into linear regression model.
7. After fitting our data to the model we can check the score of our data ie., prediction. in this case the prediction is **73%**

The accuracy of the model is lower than our aim of 85

So we use **gradient boosting regression:** It is a machine learning technique for regression and classification problems, which produces a prediction model in the form of an ensemble of weak prediction models, typically decision trees.



1. We first import the library .
2. We create a variable where we define our gradient boosting regressor and set parameters to it , here

n\_estimator — The number of boosting stages to perform. We should not set it too high which would overfit our model.

max\_depth — The depth of the tree node.

learning\_rate — Rate of learning the data.

Loss — loss function to be optimized. ‘ls’ refers to least squares regression

minimum sample split — Number of sample to be split for learning the data

. We then fit our training data into the gradient boosting model and check for accuracy

4. We got an accuracy of **91.94%**