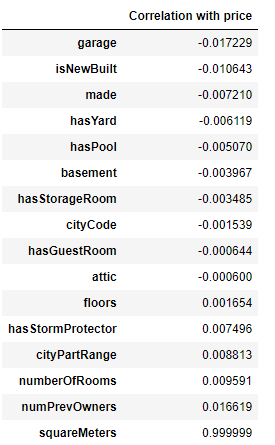
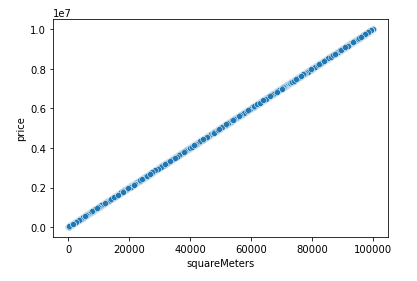
**Regression**

**Single Output Regression**

The dataset from ParisHousing.csv, has 10000 rows and 17 columns. The column ‘price’ is the label we want to predict. More details of the feature names and definition is given in the assignment question. Calculating the correlation of each feature with price, we get





From this we can see that other than the squareMeters feature (which varies linearly with price), all other features barely have any correlation with price. We will later compare the accuracy of keeping all features and if we drop all columns other than squareMeters.

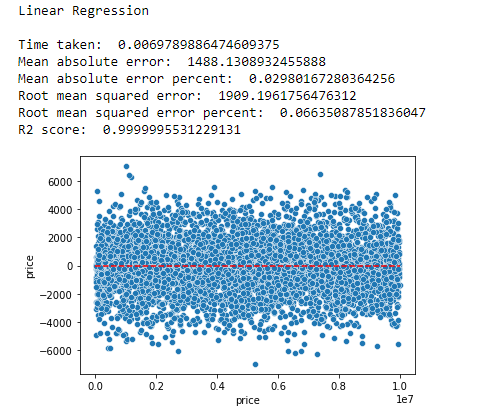
**1) Comparing different regression algorithms**

For each of linear regression, ridge regression, lasso regression, elastic net regression, SVM (with default values), SVM (with grid search) and random forest regression (all implemented using sklearn), I have calculated the following

* Time taken to train
* Mean absolute error
* Mean absolute error as a percent of mean value of price
* Root mean squared error
* Root mean squared error as a percent of standard deviation of price
* The R2 score
* In addition to these I have also plotted residue vs price.

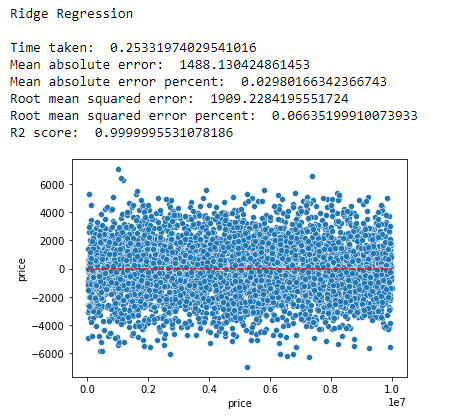
**Linear Regression:**

Using the default values, I got



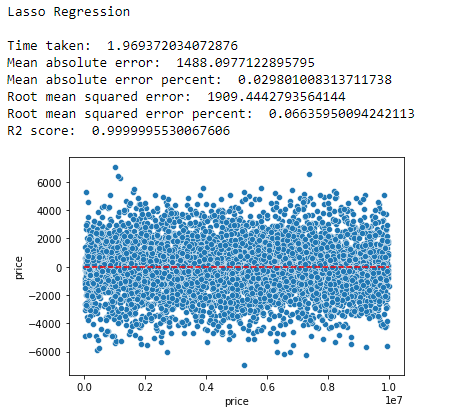
**Ridge Regression:**

Using k fold cross validation over alphas = [0.01,0.1,0.5,1,5,10] with k=10, I got

­­

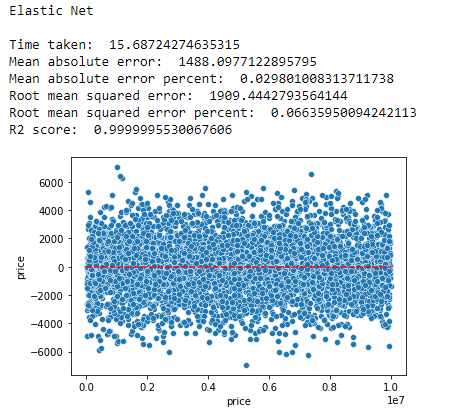
**Lasso Regression:**

Using k fold cross validation with epsilon = 0.0000001, number of alphas = 1000 and k=10, I got



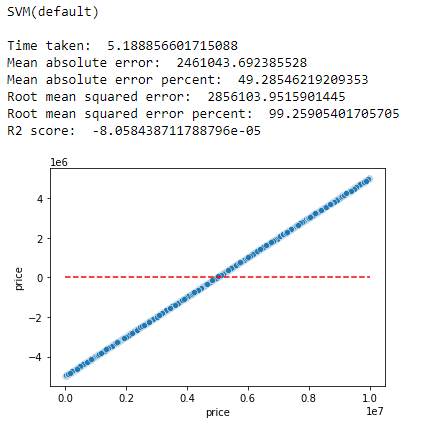
**Elastic Net:**

Using k fold cross validation with l1\_ratio = [.1, .5, .7, .9, .95, .99, 1], epsilon = 0.0000001, number of alphas=1000 and k=10, I got



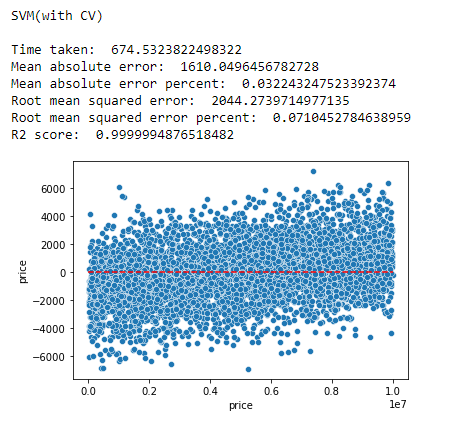
**SVM (default values)**

Using the default values, I got



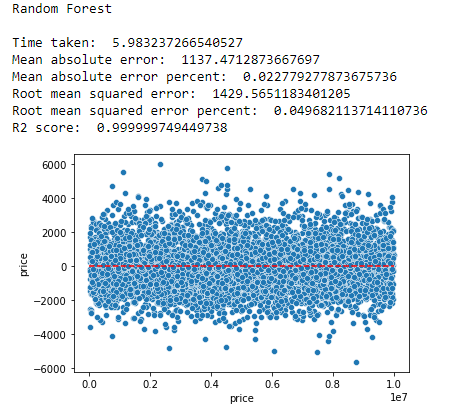
**SVM (with grid search)**

Using a grid search with parameters kernel = [‘rbf’, ’linear’], C = [0.01,0.1,1,5,10,100,1000], Gamma = ['auto’, ‘scale'] and k value = 5, I got



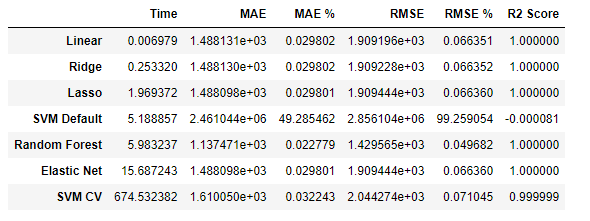
**Random Forest Regression:**

Using the default values, I got



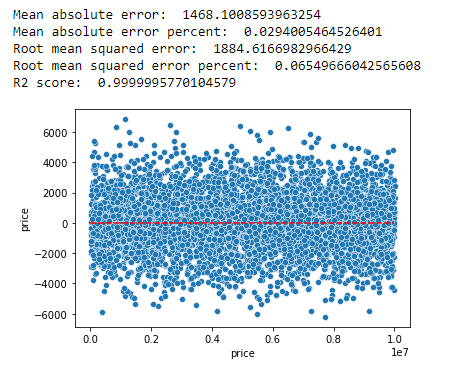
**Which to choose?**

Comparing all the data (sorted by time taken to train), we get



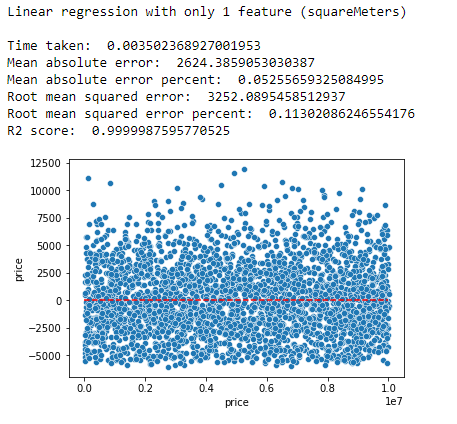
From this we can see that in most algorithms, we get similar error values (svm default is probably has a lot of error because SVM’s accuracy is highly dependent on the C and gamma values), but the time for linear regression is the least by a large margin.

That is why we should probably use linear regression model here, which on a holdout test set gives



**2) Keeping only squareMeters feature:**

As we had seen earlier, only the squareMeters feature had very high correlation with price. We have seen the performance of Linear Regression when we had all the features. Now let us see how it performs if we only have the squareMeters feature.



If we compare it with the previous models, we can see that it is a pretty good estimate and runs faster. In this case with not as many examples, it may not be worth dropping the other features but if the dataset became very large it could be considered to just keep the squareMeters feature.

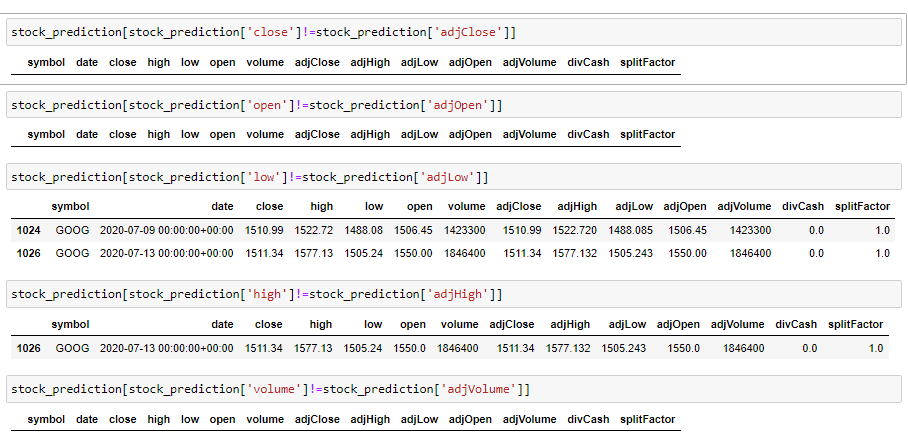
**Multiple Output Regression**

The dataset from GoogleStockPrediction.csv, has 1258 rows and 14 columns. The columns ‘close’ and ‘open’ are the labels we want to predict. More details of the feature names and definition is given in the assignment question. Calculating the correlation of each feature with close and open, we get

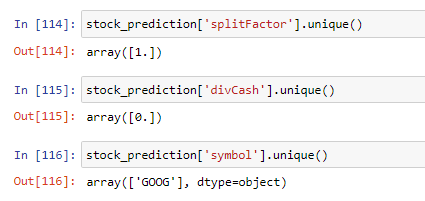


From this we can see some strange things. adjClose is perfectly correlated with close and adjOpen is perfectly correlated with open. Also, high and low are highly correlated with close and open.

Checking the data set we get



This tells us that all of the adj versions are almost the exact same as the non-adj versions. If we keep them then the algorithm, it will almost be the same as duplicating columns. Also since adjClose and adjOpen are the exact same as close and open, the algorithm can simply set the predicted values as adjClose and adjOpen and be extremely accurate. We also get

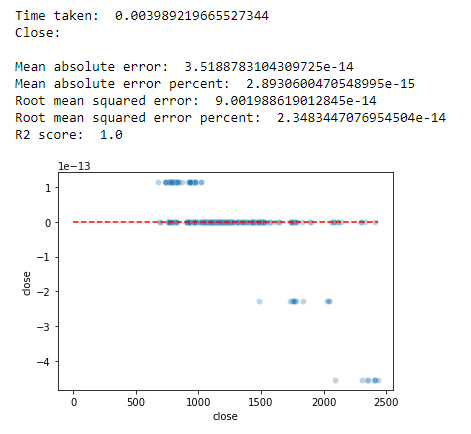


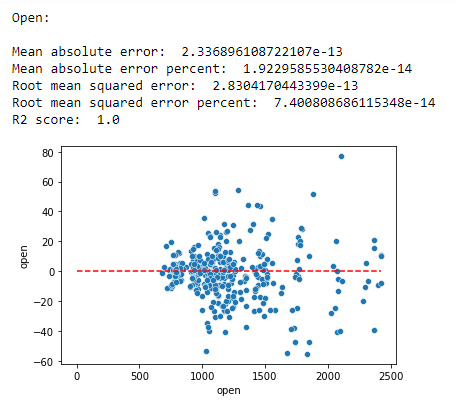
This tells us that all the data have these features in common so they wouldn’t affect the algorithm. Also ‘date’ and ‘symbol’ are non-numeric variables.

I will first drop the features, ‘symbol’, ’divCash’, ‘splitFactor’ and ‘date’. Then let us compare the results of keeping the adj values and dropping them.

**1) Keeping the adj values:**

Using the default values for linear regression, we get the following performance for the close and open values:

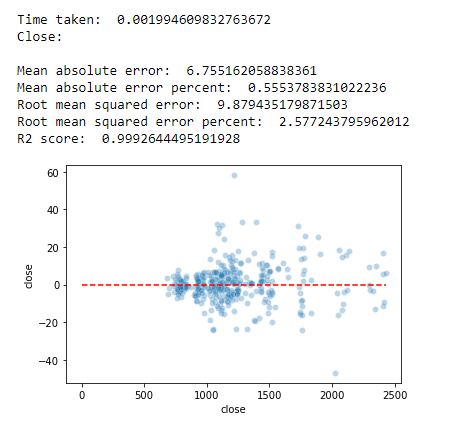


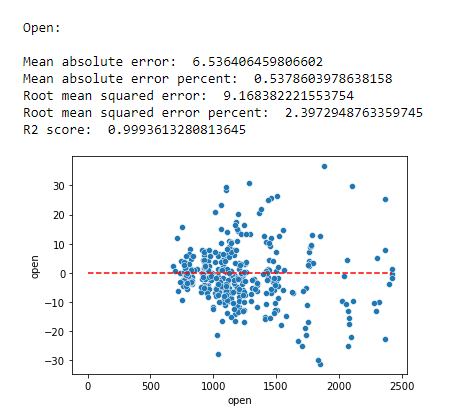


We can see that they are pretty much perfect.

**2) Without adj values:**

Using the default values for linear regression, we get the following performance for the close and open values:





We can see that although we have only used 3 features (high, low and volume), we are getting very good performance.