Zillow Home Value Prediction

(<https://www.kaggle.com/c/zillow-prize-1/overview>)

**Contents:**

* Introduction to the dataset
* Competition Question
* Data Download
* Data Description
* Answering Professor’s Questions
* Exploratory Data Analysis
* Implementing Machine Learning Algorithm
* Results
* Conclusion
* References

**Introduction to the dataset:**

Ever since Zillow released its Home Value Prediction 11 years ago, it was a great hit amongst the first-time home buyers. Home being the largest and most expensive purchase a person makes in his lifetime, it ensured homeowners had access to reliable information, free of cost, at the click of a button.

**Competition Question:**

Zillow is predicting the sale price of every house, which they call as the ‘Zestimate’, comparing it with the actual sale price of the house and getting a log error of the difference. Given all the features of the house, Zillow is asking Kaggle participants to predict this log error for other houses.

**Data Download:**

To programmatically download the data from [www.kaggle.com](http://www.kaggle.com) please follow the below steps:

* Step 1: Sign up for a Kaggle account at [https://www.kaggle.com](https://www.kaggle.com/).
* Step 2: Go to the 'Account' tab of your user profile (https://www.kaggle.com/<username>/account) and select 'Create API Token'. This will trigger the download of kaggle.json, a file containing your API credentials.
* Step 3: Place this file in the location ~/.kaggle/kaggle.json (on Windows in the location C:\Users\<Windows-username>\.kaggle\kaggle.json).

Download data using os.popen command and use zillow-prize-1 as competition name

**Data Description:**

Below is the description about the files used in this analysis/prediction:

• properties\_2016.csv - all the properties with their home features for 2016

• train\_2016.csv - the training set with transactions from 1/1/2016 to 12/31/2016

• sample\_submission.csv - a sample submission file in the correct format

These files have been manually downloaded and placed in a 'Data' folder.

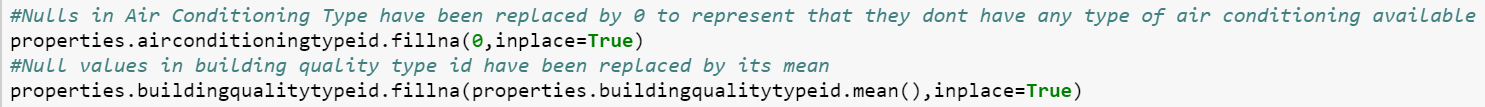
Output files shall be placed in an 'Output' folder. Hence, I have started with writing a code to traverse through the file system and create a new folder.

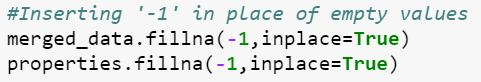
Bokeh library is used to create graphs and plots in a new tab in the browser. These plots are also saved in html format in the Output folder.

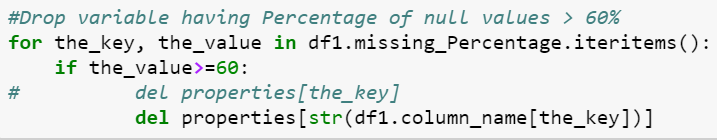
**Answers to Professor’s Questions with relevant code snippets:**

* Data cleaning
  + Are there missing values?

Yes, some of them are logically replaced by ‘-1’. Most of the missing values have been dropped.





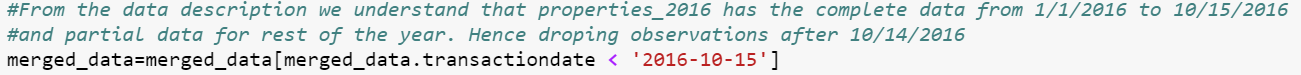


* + Are there inappropriate values?

No. There are no inappropriate values

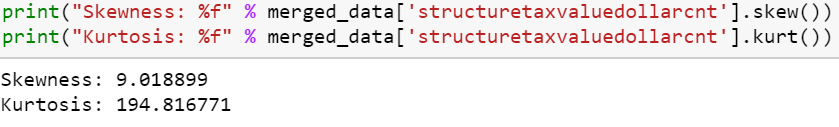
* + Remove or impute any bad data.

Some of the bad data has been dropped.



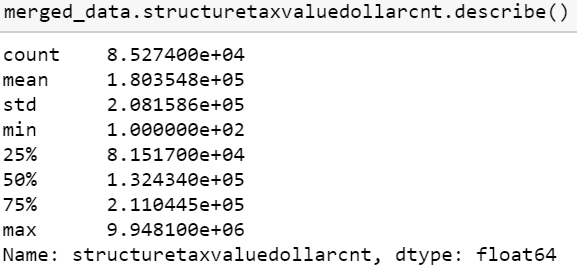
* Answer the following questions for the data in each column:
  + How is the data distributed?

The data is right schewed.



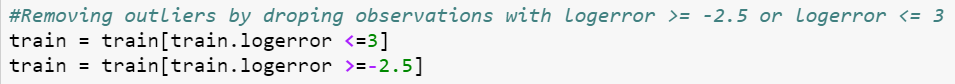
* + What are the summary statistics?

The summary statistics is as follows:



* + Are there anomalies/outliers?

Yes, there are anomalies. They have been dropped



* Plot each column as appropriate for the data type:
  + Write a summary of what the plot tells you

Explanation of each plot is done in Exploratory Data Analysis.

* Are any of the columns correlated?

The columns are not correlated

* Write a clear summary of what the EDA tells you

**Exploratory Data Analysis** is as follows:

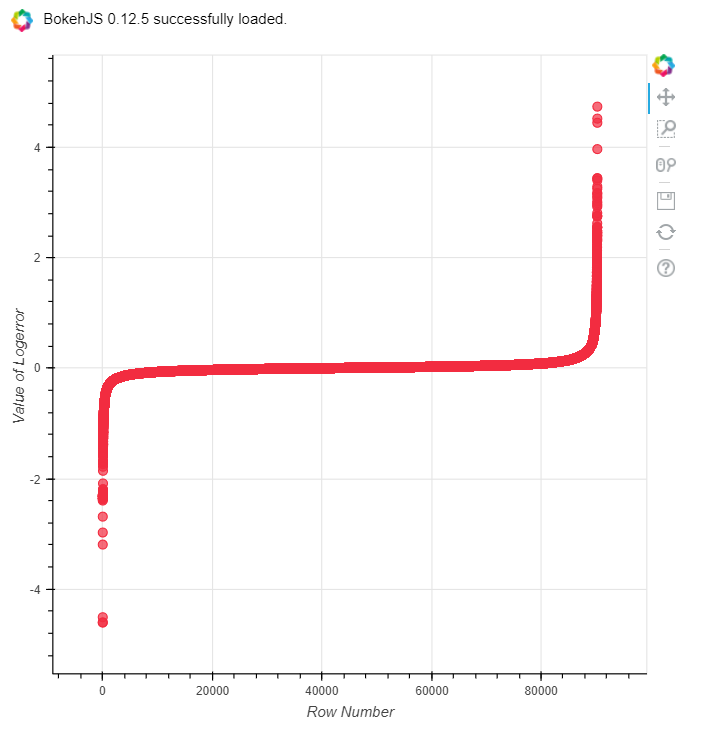
Understanding the dataset:

• properties\_2016.csv has 2985217 observations with 53 variables for each observation.

• train\_2016.csv has 90275 observations with 3 variables for each observation.

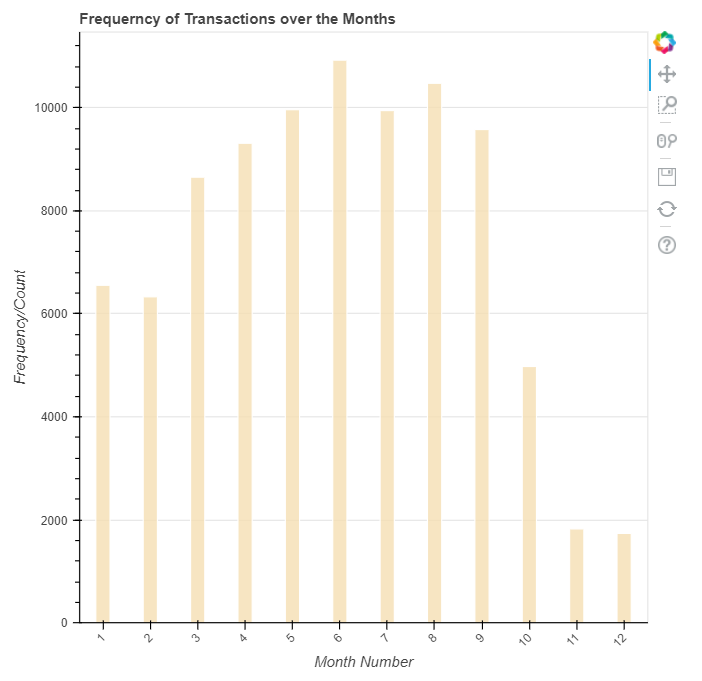
• sample\_submission.csv has 2985217 observations with 2 relevant variables for each observation.

We start with plotting a scatter plot to understand the distribution of ‘logerror’ in the train\_2016 dataset.



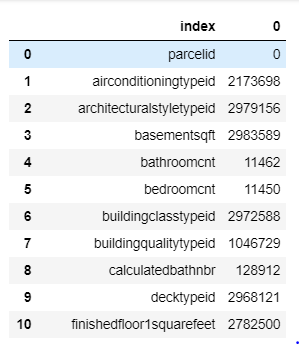
On plotting this graph, I truncated the outliers i.e. observations with ‘logerror’ >= 2.5 or ‘logerror’ <= 3 were dropped.

Then I moved onto plotting the month-wise frequency of these observations.

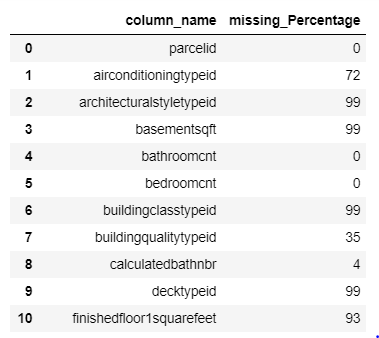


This revealed that there was a significant drop in the number transactions during November and December 2016. Also, the data description mentioned that data for November and December 2016 were not complete. Hence, we move on to dropping observations that have transactions dates greater than 10/15/2016.

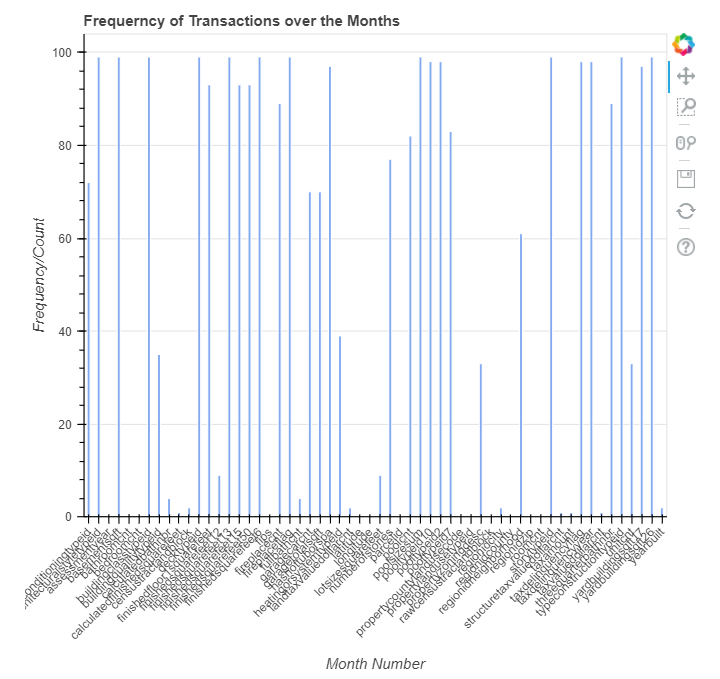
Now we move on to understanding the properties\_2016 dataset. Manual overview reveals that there are null values for most of the variables. Hence, we start with finding the frequency of null for each variable. The first 10 variables with their null count is as follows:



Since the count seems to be quite large we move on to counting the percentage of null values for each variable. The first 10 variables with their respective null percentage is as follows:

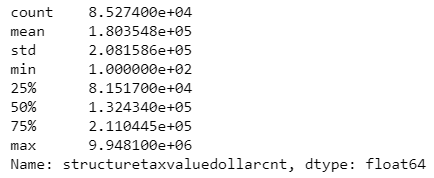


Since there are 53 variables, it difficult to get an overall view. Hence, we move on to plotting a bar graph.

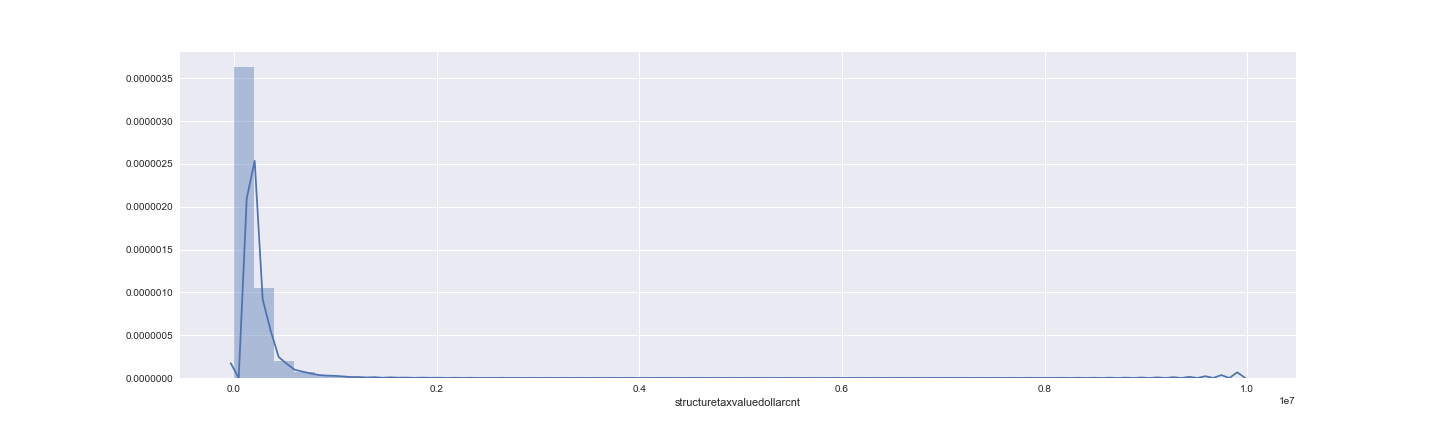


We move on to dropping variables with null percentage greater than 60%. Next, we merge the cleaned properties\_2016 and train\_2016 datasets to get a total of 85652 observations with 31 variables for each.

Out of these 31 variables, data description mentions that 'structuretaxvaluedollarcnt' reveals 'The assessed value of the built structure on the parcel.' A simple describe functions gives the below details regarding the variable.

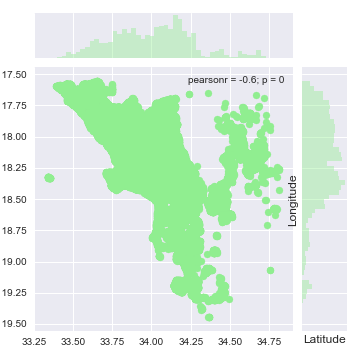


Seeing the difference in the minimum, mean and maximum values for this variable we plot a skewness graph to get a get the distribution.



The skewness of a graph describes how the data is distributed across the mean. The above data is right (positively) schewed i.e. most of the data is greater than the mean.

We also observe that the observations have latitude and longitudes. The data description mentions that the observations are for Los Angeles, Orange and Ventura, California counties. We plot a map to verify the same.



Finally, we save this dataset in the Data folder to further use it as an input to implement machine learning algorithms.

**Data Pre-processing:**

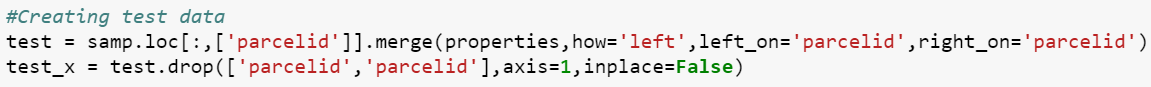
We start with importing the above created ‘merged\_data’, properties\_2016 dataset and the ‘sample\_submission’ dataset. Then we take variables with numerical values and ignore the variables with strings as string values cannot be used in this machine learning model. Finally, we replace the empty values with '-1'.

**Implementing Machine Learning Algorithms**

• The first step in implementing machine learning algorithms is to Create a training dataset I have done that by including all variables except ‘parcelid’ and ‘logerror’ from the ‘merged\_data’ dataset. Sample output i.e. ‘logerror’ in the form of a series of values form the ‘train\_y’ dataset.



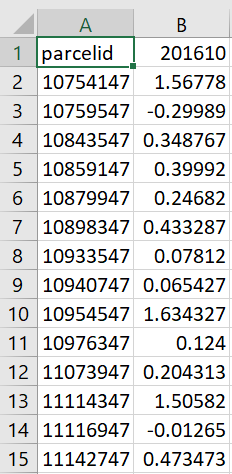
• Next step is to Create a testing dataset I am using the sample\_submission dataset to create a test dataset. test\_x dataset contains variables similar to train\_x.



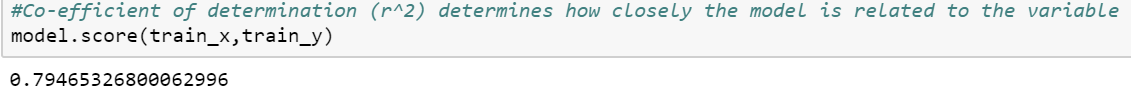
• We then implement Random Forest Regressor for 5,7 and 10 variables and check the prediction of log error. Top 1000 rows of the output are saved in .csv file named LogErrorPrediction with a time stamp. Only top 1000 rows are considered due to space constraints.

**Results:**

We have successfully predicted the logerror for each house in the sample dataset. Top 1000 rows of the output are saved in .csv file named LogErrorPrediction with a time stamp. Only top 1000 rows are considered due to space constraints. Top 15 parcelid with their predicted logerror is as follows:



R^2 value of the model is 79% i.e. the model determines 79% of the variables. It is obtained by:



**Conclusion:**

It was a very interesting dataset to work with. High percentage of null value was a challenge, but even after ignoring them, my machine learning model had enough data to train and get the desired results.

**References:**

* https://www.kaggle.com/c/zillow-prize-1
* <https://www.kaggle.com/arjanso/simple-starter-randomforest-regressor>
* <https://github.com/Kaggle/kaggle-api>