# Udacity Machine Learning Engineer Nanodegree
# Capstone Project

Amrish Purohit July 07, 2018

## Table of Contents

#  1. Definition

## 1.1 Project Overview

 Estimating price of a house is quite challenging and important process as housing prices are difficult to predict, and can be influenced by a very large quantity of factors. Everyone want better home in lease amount. This is lifetime decision and people spends hundreds of thousands of dollars to buy a house (at least in Los Angeles and major cities of California and US). To make such important decision, buyer want to make sure that fair price is placed on the property, particularly price of the house is not inflated.

Now days there are many real estate companies provide data and own define algorithm to determine best price of the house. One of such company is Zillow. Zillow has millions of data on homes across United States. Zillow has machine learning algorithm called "Zestimate". The Zestimate home value is Zillow's estimated market value for an individual home and is calculated for about 100 million homes nationwide. The Zestimate is calculated from public and user-submitted data, considering special features, location, and market conditions. More information about Zestimate can be obtained from here Zestimate.

## 1.2. Problem Statement

Based on data set provided on Kaggle, https://www.kaggle.com/c/zillow-prize-1, goal of the project is to predict price of new property going to sold in Los Angeles, Orange and Ventura county of California. Here price of home is evaluated based on features of home, it is a regression problem.

## 1.3 Datasets and Inputs

For this project, I will use data set provided by Zillow on Kaggle competition, https://www.kaggle.com/c/zillow-prize-1. The train data has all the transactions before October 15, 2016, plus some of the transactions after October 15, 2016 and test data in the public leaderboard has the rest of the transactions between October 15 and December 31, 2016. About 58 features provided in properties file like size, neighborhood, tax and location.

The data set will be used to train model to estimate price of home and prediction will be compared with Kaggle board.

## 1.4. Evaluation Metrics

Submissions are evaluated on Mean Absolute Error between the predicted log error and the actual log error. The log error is defined as

$$logerror=\log(Zestimate)-\log(SalePrice)$$

**Mean Absolute Error (MAE)** is a measure of difference between two continuous variables. Assume *X* and *Y* are variables of paired observations that express the same phenomenon. Examples of *Y* versus *X* include comparisons of predicted versus observed, subsequent time versus initial time, and one technique of measurement versus an alternative technique of measurement. Consider a scatter plot of *n* points, where point *i* has coordinates ( $x_i$, $y_i$ ).

Mean Absolute Error (MAE) is the average vertical distance between each point and the Y=X line, which is also known as the One-to-One line. MAE is also the average horizontal distance between each point and the Y=X line. The solution for the problem will be written in python using scikit and other packages. As, the problem is a regression problem and because the evaluation of the solution is based on MEA, 'neg_mean_absolute_error' is chosen for model evaluation. This 'regression' scoring metric provided by scikit is used to measure the MAE value which is used to measure the Zillow problem. MAE output is non-negative floating point.

The metric measures the distance between the model and the data and in case of 'neg_mean_absolute_error', the lower return values are better than the higher return values. The best value is 0.0.

## 2. Analysis

### 2.1. Data Exploration

The provided dataset is a tabular, comma separated values sheet with nearly 3 million samples that have 58 features to describe them. It is actual real estate information from 3 counties (Los Angeles, Orange and Ventura) in California. Zillow's data is obtained from publicly available sources, and since different cities and counties track statistics in different ways

We are provided with two sets of training data and their corresponding "properties" file: (i) "train_2016_v2.csv" and "properties_2016", (ii) "train_2017.csv" and "properties_2017". The main difference between "properties_2016" and "properties_2017" are in tax features.

When performing missing data analysis, we can observe that most of the variables have a high ratio of missing data. There are 25 variables which have a missing ratio of > 70%.

From correlation analysis, we can observe that following pairs are highly correlated (bathroomcnt, calculatedbathnbr and fullbathcnt)

(calculatedfinishedsquarefeet and finishedsquarefeet12)

(fips, rawcensustractandblock, censustrackandblock) and

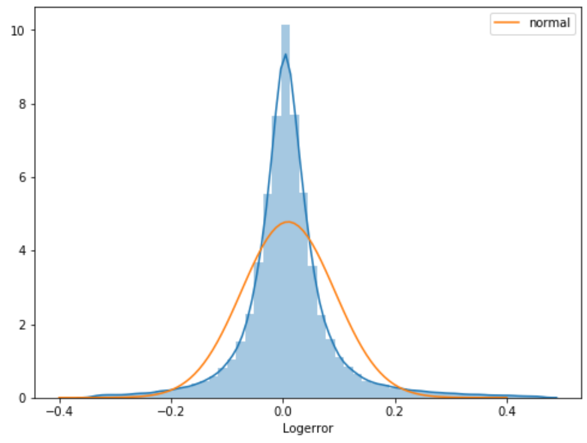
(taxvaluedollarcnt, taxamount, landvaluedollarcnt).

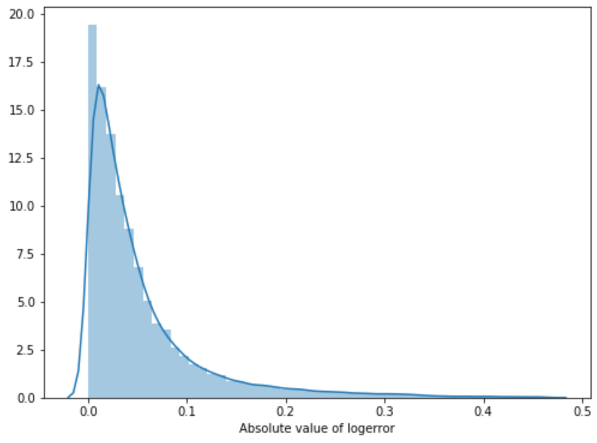
The data has been divided into continuous, discrete and categorical data. By plotting histograms on the continuous variables, we can observe that most of the data is normally distributed.

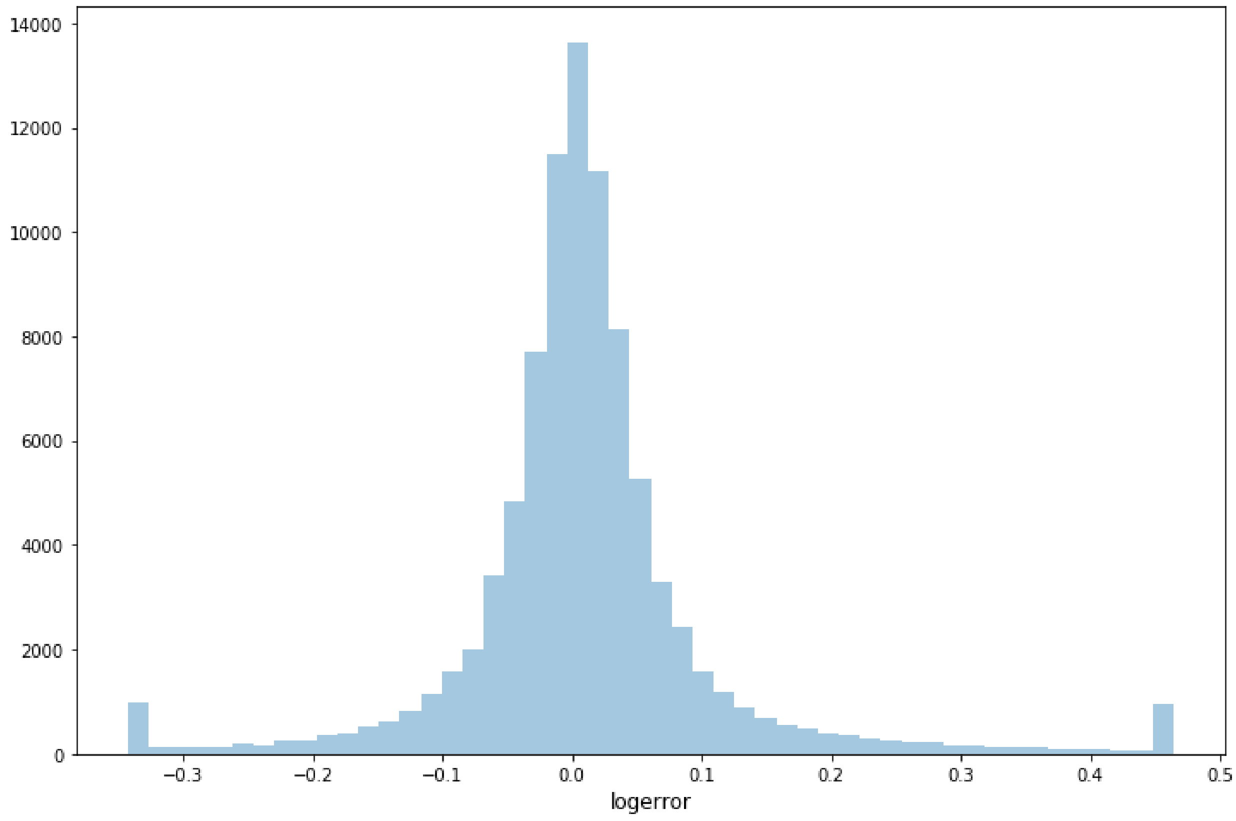
Preprocessing for this dataset will have to include alterations to the features like replacing missing values (where applicable) and changing the way some of the features are represented via encoding strategies.

## 2.2. Exploratory Visualizations

The target variable ‘logerror’ has few outliers but is mostly normally distributed.







The data set "properties_2016" consists of 58 features and 2,985,217 observations. The features include information about the size, neighborhood, tax, and location of the properties.

Data has been largely divided into three categories.

1. continuous

'taxamount','garagetotalsqft', 'calculatedfinishedsquarefeet', 'lotsizesquarefeet','structuretaxvaluedollarcnt'

continuous variables are depicted through display plot

2. discrete

'bathroomcnt', 'bedroomcnt','garagecarcnt', 'roomcnt', 'unitcnt','assessmentyear', 'yearbuilt'

discrete variables are depicted through count plot

3. categorical

'airconditioningtypeid', 'buildingqualitytypeid', 'heatingorsystemtypeid', 'propertylandusetypeid', 'fips', 'regionidcounty'

categorical variables are depicted through bar plot.

## 2.3. Algorithms and Techniques

Initially I was planning to use deep learning technique **CNN** to implement the algorithm as I am very impressed with deep learning during course of the program. As feature set is relative small in context of deep learning and comment I got from my connect program teacher, It may tend to over fitting on training set.

I have tried several tree based ensemble methods and KNN and finally chose XGBoost algorithms for solving the problem. XGBoost is an ensemble method that makes predictions by combining the output of several trees. XGBoost is fast compared to other implementations of gradient boosting. XGBoost is very good with structured or tabular datasets on classification and regression predictive modeling problems.

XGBoost optimizes an objective function. The objective function is a summation of a loss function and a regularization function which depends on the leaf weights and the number of leaves in a tree.

Below are the main parameters that was used to optimize the model in this project:

Nthread = To utilize all cores of the processor

max_depth (9) = the maximum depth of each tree.

min_child_weight (100) = the minimum number of samples that must be present in each node

n_estimators (100) = Number of boosted trees to fit.

learning_rate (0.1) = this parameter scales down the steps along the gradient. It can be used to make the boosting process more conservative.

colsample_bytree (1) = the ratio of the features that is used to construct a tree.

Subsample (0.8) = ratio of the training data points that are randomly selected to construct each tree.

Once the base score is achieved by the model, I have tuned the same by using Grid Search and Cross Validation methods.

### 2.4. Benchmark

As I have mentioned in my proposal, I have used Linear regression as benchmark model. Linear regressor got score of -0.06850. I have used other tree base ensemble methods. I have observed that XGBoost provides better result.

## 3. Methodology

### 3.1. Data Preprocessing

preliminary assessment of the dataset for missing values is required before fitting data into the model as dataset contains features with lots of missing values more than 70%.

Merge training data file (train_2016_v2.csv) and properties file(properties_2016.csv).

Cleaning data by dropping columns which have a missing ratio of more than 70%. Here is list variable dropped due to missing values

- architecturalstyletypeid,
- basementsqft
- buildingclasstypeid
- decktypeid
- finishedfloor1squarefeet

- finishedsquarefeet13
- finishedsquarefeet15
- finishedsquarefeet50
- finishedsquarefeet6
- fireplacecnt
- hashottuborspa
- poolcnt
- poolsizesum
- pooltypeid10
- pooltypeid2
- pooltypeid7
- storytypeid
- threequarterbathnbr
- typeconstructiontypeid
- yardbuildingsqft17
- yardbuildingsqft26
- numberofstories
- fireplaceflag
- taxdelinquencyflag
- taxdelinquencyyear

On correlation analysis, it was observed that the following pairs are highly correlated.

- fips, rawcensustractandblock, censustrackandblock
- taxvaluedollarcnt, taxamount, landvaluedollarcnt
- bathroomcnt, calculatedbathnbr and fullbathcnt
- calculatedfinishedsquarefeet and finishedsquarefeet12

Among above variables, variable with high missing value is dropped.

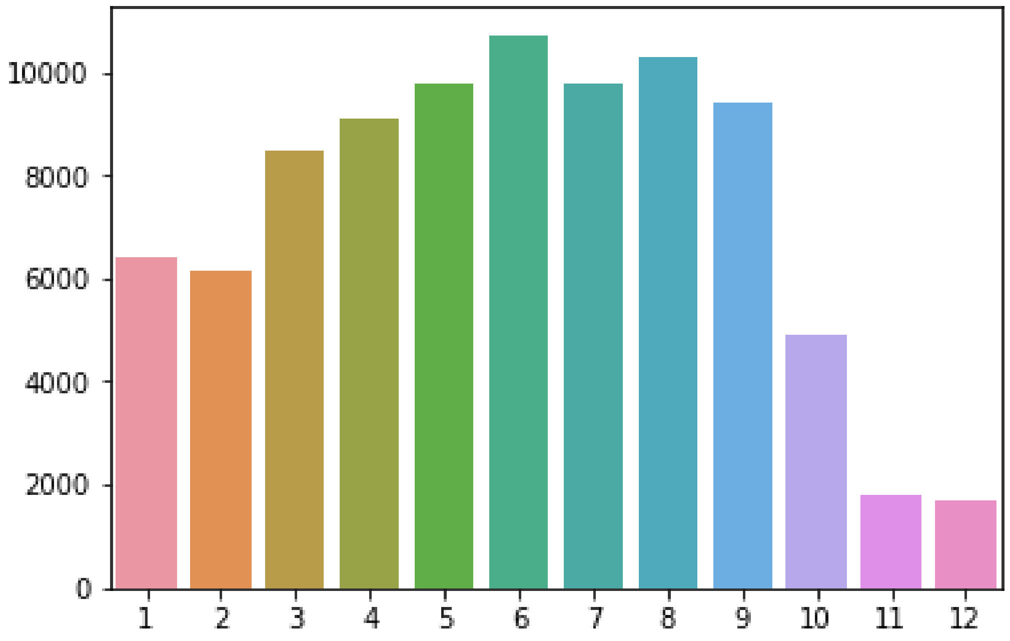
Numeric variables with null columns, data was imputed with median values.

Variables with null columns, data was imputed with -1.

Outliers were eliminated from target variable ‘log_error’.

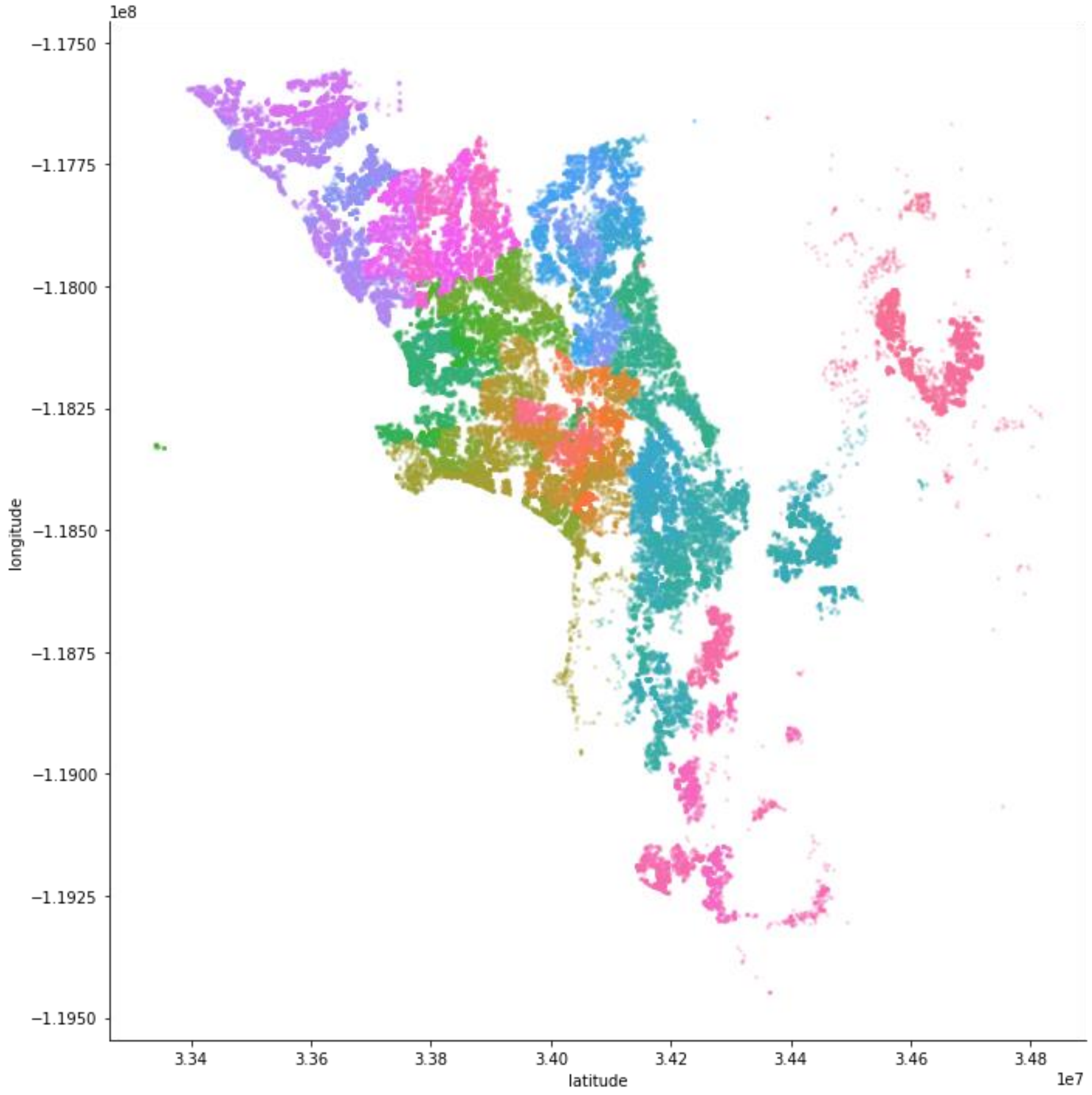
## 3.2 Feature Engineering.

A bar plot is generated to understand transactions per month.



New_Transaction_Month column is added to the training model.

New_zip_count : Number of properties in the zip



New_LivingAreaProp : Proportion of living area is added.

New_city_count : Number of properties in the city is added.

## 3.2. Implementation

- **Packages and Dataset :** I have used Jupyter notebooks with Python3 for implementation. Libraries and packages (numpy, pandas, matplotlib, seaborn, sklearn, catboost, xgboost) are imported prior to implementation and number of CPU cores are calculated with multiprocessing package. Other packages are imported as needed.
- **Benchmark Model:** Initial benchmark model is created with Linear regression as already mentioned in proposal. Linear regression yield score of 0.06850.
- **Read and prepare dataset:** Read and merge training data and properties data on unique field 'parcelid'. Columns are dropped where Nan values are more than 70%. Null values are imputed and median values are replaced in numeric column
- **Correlation Analysis:** Correlation between features are plotted with heat map and columns with high missing values are dropped from the training set.
- **Removing Outliers:** Outliers are plotted with target variable and removed from training data set.
- **Feature Engineering :** New feature as described in previous section is added with feature engineering.
- **Model Selection:** I have tried following models with default parameters.

| Model Name | Score |
|---|---|
| Bayesian Ridge Regression | -0.05314 |
| Linear Regression | -0.05315 |
| Ridge Regression | -0.05315 |
| Lasso Regression | -0.05314 |
| Decision Trees | - 0.08308 |
| Random Forest | -0.05925 |
| KNN | -0.06117 |
| XGB Regression | -0.05280 |
| Gradient Boosted Regressor | -0.05285 |
| CatBoostRegressor Regressor | -0.05280 |

Among above models, XGB Regression is selected with score -0.052796701349816476.

- **Model tuning :** Selected model is tuned with GridSearchCV. Detail is provided in next refinement section.

### 3.2.1. Cross-Validation

I have used used the k-fold cross-validation scheme with k=10. In k-fold cross-validation method, we train a model k times on (1-1/k)*100 percent of the training data.

The remaining 1/k*100 percent of the data is used for validation. The validation dataset is randomly selected from the training dataset. Those k models are then used to make a prediction for the test dataset.

After cross validation final score obtained from tuned model is Final score : -0.05277.

## 3.3. Refinement

Benchmark Model Linear regression provide accuracy of -0.06850. This score was based before preprocessing of the data. After preprocessing, cleaning and feature engineering on data. I ran Linear regression again and yield score of -0.05315. I ran all the model described above and chose XGB regressor as a final model.

I created the GridSearchCV object to tune the parameters for the XGB regressor. Parameters that I have considered for tuning are as follow.

```
params = {
        "nthread" : [ncpu],
        "max_depth": [3,4,5,7,9],
        "min_child_weight": [1,10,50,100],
        "n_estimators": [10,100,500,1000],
        "subsample" : [0.5, 0.8, 1],
        "colsample_bytree" : [0.5, 1],
        "learning_rate":[0.1, 0.01]
      }
```

I ran grid search object for hours with permutations and combinations to find out right parameters. Final parameters chosen are as follow, output from Grid Search.

```
Best Score : -0.052802641831849545
Best Parameters : {'colsample_bytree': 1, 'learning_rate': 0.1, 'max_depth': 3, 'min_child_weight': 100,
'n_estimators': 100, 'nthread': 8, 'subsample': 0.8}
Best Estimator : XGBRegressor(base_score=0.5, booster='gbtree', colsample_bylevel=1,
    colsample_bytree=1, gamma=0, learning_rate=0.1, max_delta_step=0,
    max_depth=3, min_child_weight=100, missing=None, n_estimators=100,
    n_jobs=1, nthread=8, objective='reg:linear', random_state=0,
    reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
    silent=True, subsample=0.8)
```

# 4. Results

## 4.1 Model Evaluation and Validation

The final model was chosen after analyzing all major regression techniques, finalizing on using XGBoost, tuning the XGBoost using Grid Search and Cross Validation techniques. The robustness of the model and its solution were tested by using Cross Validation with number of folds as 10. When the cv argument is an integer, cross_val_score uses the KFold or StratifiedKFold strategies by default. Cross validation technique is generally used to assess the predictive ability of any regression model. The concept of cross-validation relies on the principle that a large enough dataset can split into two or more sub-groups, the first being used to derive the model and the additional data set(s) reserved for model testing and validation. This would let us know when our model might be over or under fitting on the dataset that we have employed.

## 4.2 Justification

The final model scored the following on the CV and testing sets:

Benchmark Score = -0.06850
Final score after Model tuining = -0.05277

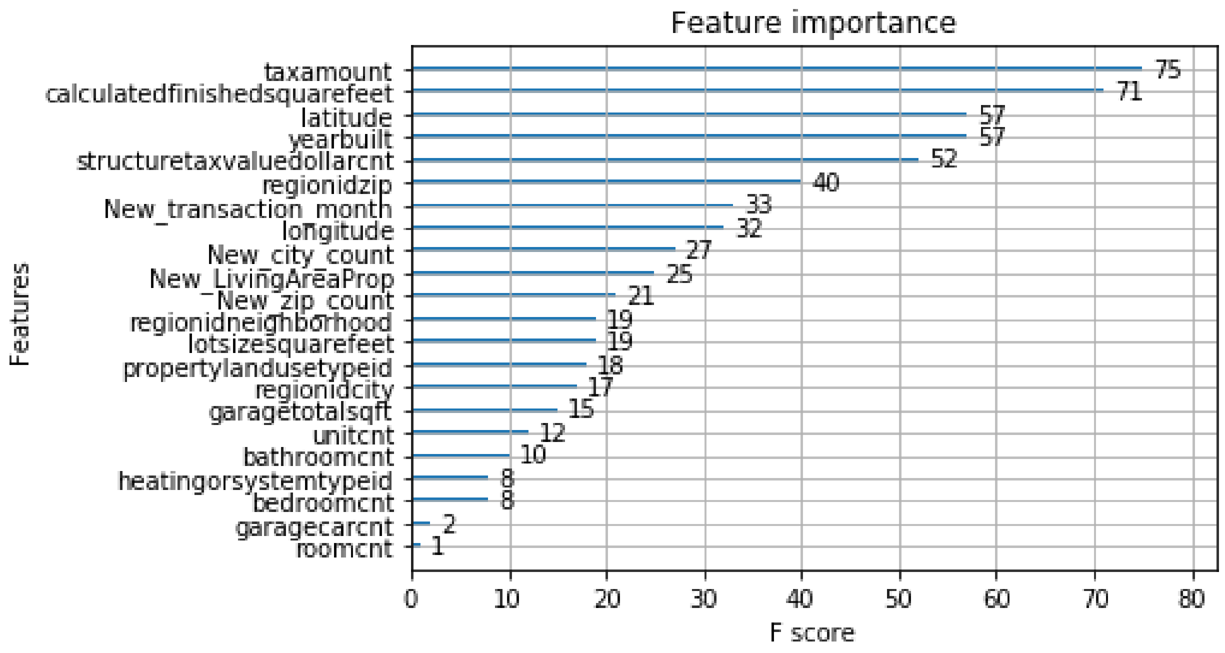
This is an improvement over the benchmark model. Steps followed in getting the result add significance and validity to the final solution obtained. Difference between Linear regression model and tuned XGB regressor obtained through data processing, removing outliers and feature engineering.

## 4.3 2017 Data

The 2017 data were released on October 2, 2017. This data included transactions from January through mid-September, 2016. The teams were also provided with properties_2017 dataset. The 2017 dataset has the same parcel_id as in the properties_2016 file but the tax assessment values are updated based on the data that were collected in 2017. The features that we engineered for the 2016 data can seamlessly be applied to the 2017 data without any needs for further adjustments.

## 5. Conclusion

### 5.1 Free-Form Visualization



The XGBoost library provides a built-in function to plot features ordered by their importance.

The function is called plot_importance() which maps features to corresponding F score. The above mentioned diagram is obtained from using the plot_importance() feature of XGBoost. Based on domain knowledge of the Housing Industry I can state that the features generated by the model are in sync with the real world housing market. For example, Most of the home buyers are interested in paying low tax amounts, look for homes with high calculatedfinishedsquarefeet, are interested in location of the house (latitute, longitude), age of the house - depicted by year built etc. All these features have been correctly marked as important by the model.

## 5.2 Reflection

The following process was used followed to create this project.

- Import Packages
- Benchmark Model - Linear Regression
- Read and Prepare Dataset
- Correlation Analysis
- Remove Outliers
- Univariate analysis
- Bivariate analysis
- Feature Engineering
- Model Selection
- Model Tuning
- Cross Validation

Many of the above steps were iterative, as I made small adjustments a number of times to see if better scores could be achieved. I enjoyed and learnt a lot by exploratory data analysis of data, and application of different regression models and techniques such as gridsearch and cross-validation.

This capstone project helped me a lot to sharpen my knowledge and skills that I have learnt during the Udacity course.

## 5.3 Improvements

- During model selection process, 1 out of 4 times, CatBoostRegressor was chosen as best model. I think CatBoostRegressor is very close match to XGB regressor. CatBoostRegressor could yield better result than XGB if we tune it.
- I could use KNN to impute missing values to get better result on final score.
- As I mentioned in proposal, I was wanted to try CNN to implement this project. CNN can be good candidate for model selection.
- One can use PCA and ICA to combine and transform some of the numerical variables into a low-dimensional space.

## 5.4 References

- https://www.kaggle.com/c/zillow-prize-1
- https://github.com/dmlc/xgboost
- http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingRegressor
- https://en.wikipedia.org/wiki/Feature_engineering