Machine Learning Engineer Nanodegree

Capstone Project

I. Definition

Project Overview

A home is often the largest and most expensive purchase a person makes in his or her lifetime. Ensuring homeowners have a trusted way to monitor this asset is incredibly important. The Zestimate was created to give consumers as much information as possible about homes and the housing market, marking the first time consumers had access to this type of home value information at no cost. Zillow's Zestimate home valuation has shaken up the U.S. real estate industry since first released 11 years ago. "Zestimates" are estimated home values based on 7.5 million statistical and machine learning models that analyze hundreds of data points on each property. And, by continually improving the median margin of error (from 14% at the onset to 5% today), Zillow has since become established as one of the largest, most trusted marketplaces for real estate information in the U.S. and a leading example of impactful machine learning.

All the real estate transactions in the U.S. are publicly available. We are provided with a full list of real estate properties in three counties (Los Angeles, Orange and Ventura, California) data in

2016. The train data has all the transactions before October 15, 2016, plus some of the transactions after October 15, 2016. The dataset is available in the Kaggle website under the competition “ Zillow Prize: Zillow’s Home Value Prediction (Zestimate)”. The shape of the training data set is 90275,3 and that of the properties data set is 2985217,58. The shape of the merged dataset is 90275, 60. Target variable for this competition is variable "log error". There are few outliers but the logerror has a normal distribution of data. Out of the 60 variables; 53 are float, 5 objects, 2 ints and 1 data time. The target variable is no unbalanced, also we will have to convert objects to integers by transforming the data before feeding the same to the model.

Link to Dataset :

https://www.kaggle.com/c/zillow-prize-1/data

Problem Statement

The problem statement is to build a model to improve the Zestimate residual error by engineering new features. The strategy will involve reading and analyzing the data, preprocess data including removing outliers, standardizing data etc, applying various machine learning regression models to get a better performing model, tuning to the model by using techniques such as grid search and cross validation. I will be using XGBoost machine learning technique to push the accuracy of the Zestimate further. The model will make more accurate predictions about the future sale prices of homes by primarily improving the Zestimate residual error.

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.

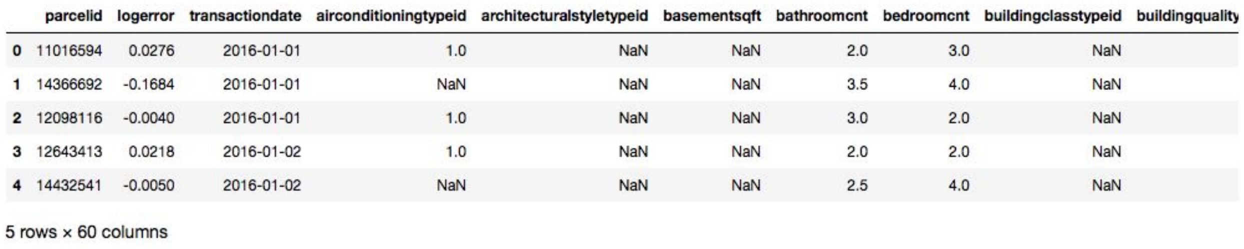
II. Analysis

Data Exploration

We can observe from the dataset that most of the variables in the data set are float variables. The breakup is as follows : int64 - 1, float64 - 53, datatime64[ns] - 1 and object -5. 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 The 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. By plotting countplots on the discrete variables we can see that the data some of the values have high counts for discrete variables. By plotting barplots on categorical variables we can observe that some of the data values are present in high percentages for few variables. The target variable logerror has few outliers, which need to be removed.

The dataset is available in the Kaggle website under the competition “ Zillow Prize: Zillow’s Home Value Prediction (Zestimate)”. The shape of the training data set is 90275,3 and that of the properties data set is 2985217,58. The shape of the merged dataset is 90275, 60. Target variable for this competition is variable "logerror". There are few outliers but the logerror has a normal distribution of data. Out of the 60 variables; 53 are float, 5 objects, 2 ints and 1 datatime. The target variable is not unbalanced, also we will have to convert objects to integers by transforming the data before feeding the same to the model.

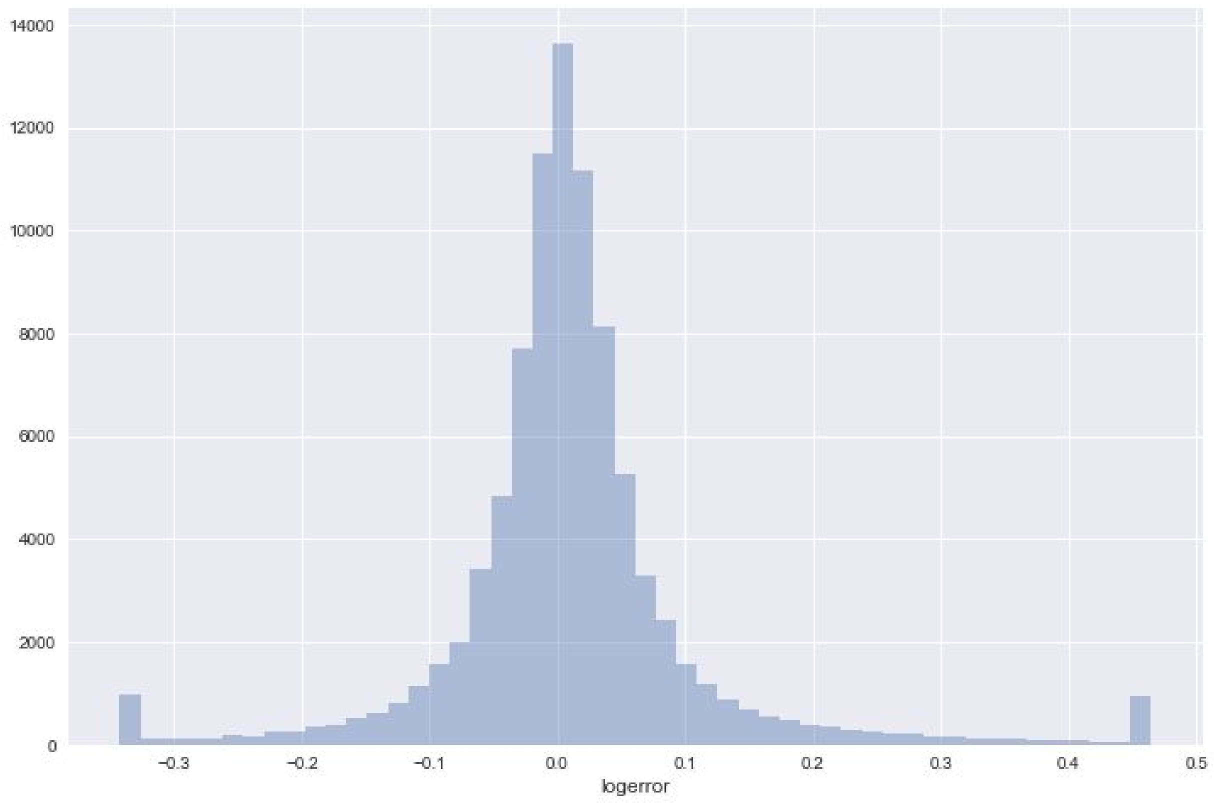
Sample Data :



	parcelid	logerror	transactiondate	airconditioningtypeid	architecturalstyletypeid	basementsqft	bathroomcnt	bedroomcnt	buildingclasstypeid	buildingquality
0	11016594	0.0276	2016-01-01	1.0	NaN	NaN	2.0	3.0	NaN	
1	14366692	-0.1684	2016-01-01	NaN	NaN	NaN	3.5	4.0	NaN	
2	12098116	-0.0040	2016-01-01	1.0	NaN	NaN	3.0	2.0	NaN	
3	12643413	0.0218	2016-01-02	1.0	NaN	NaN	2.0	2.0	NaN	
4	14432541	-0.0050	2016-01-02	NaN	NaN	NaN	2.5	4.0	NaN	

5 rows × 60 columns

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

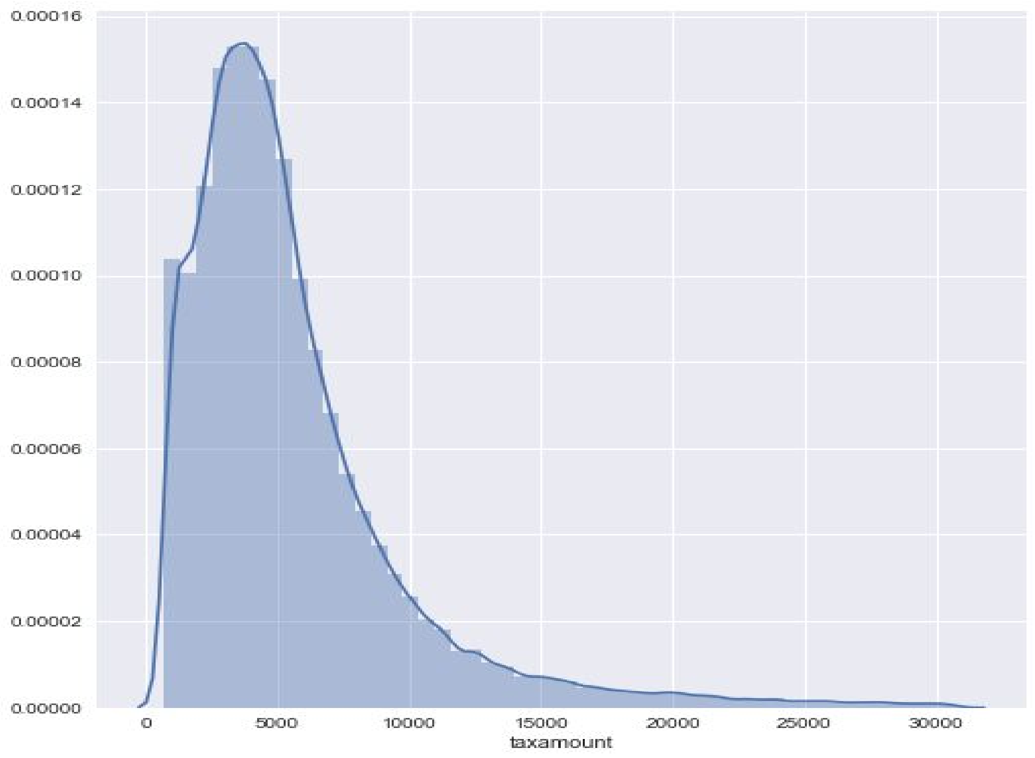


Exploratory Visualization

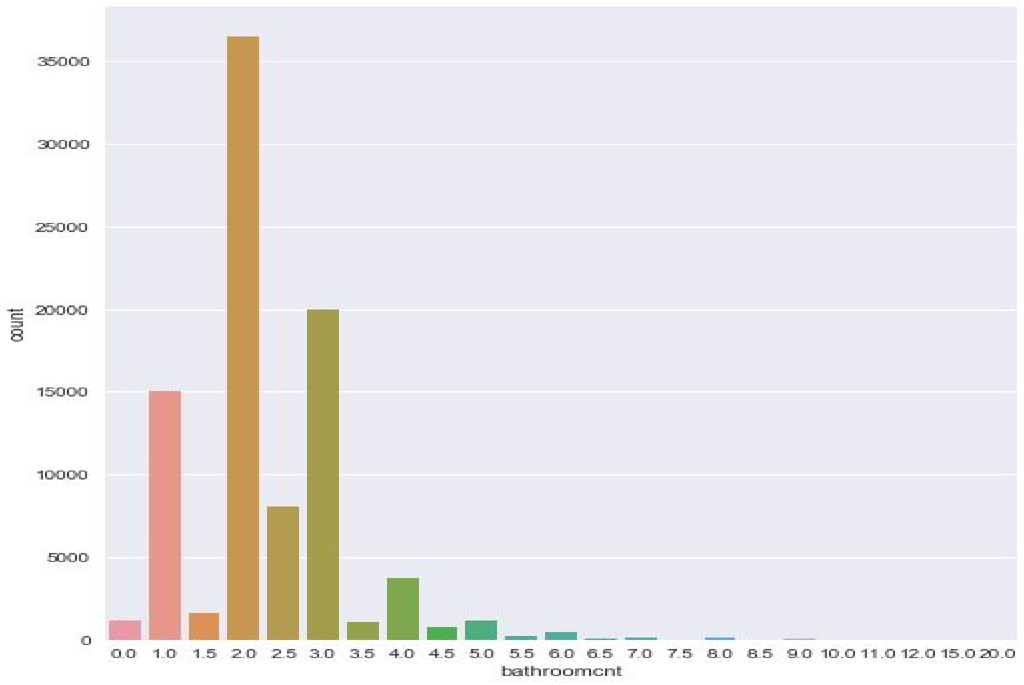
The data has been divided into continuous, discrete and categorical data. Histograms have been plotted for histograms, count plots for discrete variables, barplots for categorical variables and headmap for correlation analysis have been plotted. Most of the variable have been visualized and thoroughly analyzed.

Exploring few of the variables from the dataset:

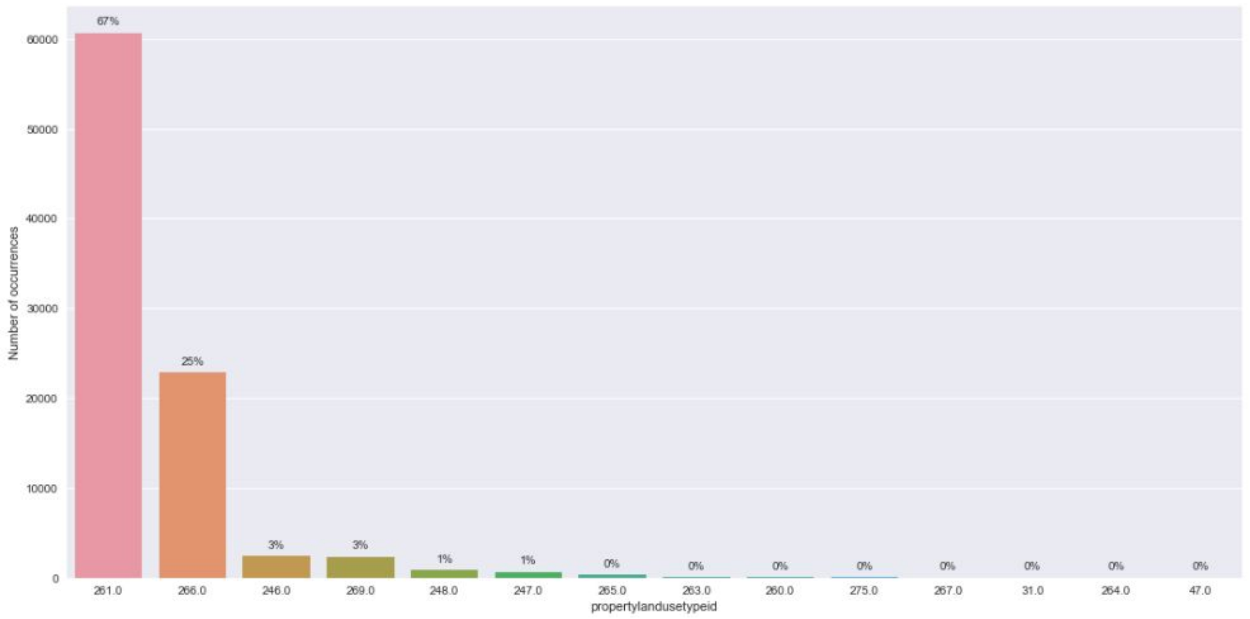
Histogram of continuous variable : taxamout. We can observe that the mean tax amount is around 5983.97 , with a range of [49.08, 321936.09].



Count plot of discrete variable : bathroomcnt. This indicates that most of the house have 2 or 2.5 or 3 bathrooms.



Bar plot of discrete variable : propertylandusetypeid. We can clearly observe that most of the data (92%) has propertylandusetypeid as either 261 or 266.

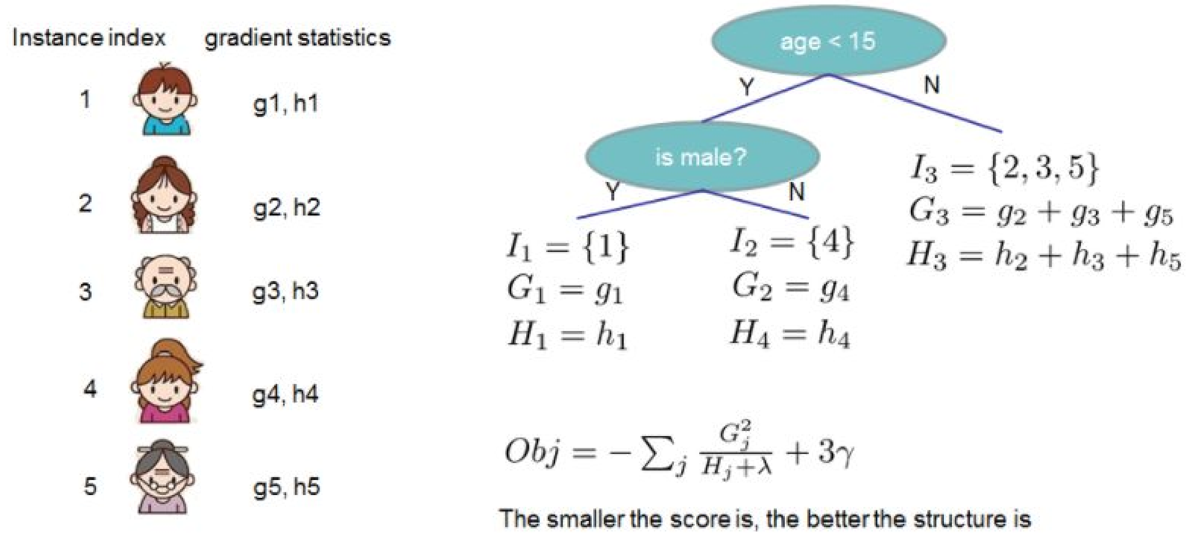


Algorithms and Techniques

I have used XGBoost algorithms for solving the problem. XGBoost stands for e**X**treme **G**radient **B**oosting and is an implementation of gradient boosted decision trees and is designed for speed and performance and is very flexible. 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 is enabled with parallel processing making it at least ten times faster than any other tree based models. It avoids overfitting by Regularization. XGBoost handles Missing values internally. Due, to all above mentioned features I decided to go with the XGBoost algorithm - fast, handles regression, optimized, good on tabular datasets, flexible parameters, handles missing values etc. Once the base score is achieved by the model, I have tuned the same by using Grid

Search and Cross Validation methods. The only parameter explicitly pass is the ‘scoring’ parameter - ‘neg_mean_absolute_error’, which will help in generating the MAE values. All other parameters are generated from Grid Search. The same are then used in cross validation.

The model of XGBoost is “tree ensembles”. The tree ensemble model is a set of classification and regression tress (CART). The following formula and figure are derived from XGBoost documentation which show how a person based on inputs such as age, gender, occupation etc. can be classified as a person who likes computer games or not.


Instance index
gradient statistics
1
g1, h1
2
g2, h2
3
g3, h3
4
g4, h4
5
g5, h5
age < 15
Y
N
is male?
Y
N
$I_1 = \{1\}$ $G_1 = g_1$ $H_1 = h_1$
$I_2 = \{4\}$ $G_2 = g_4$ $H_4 = h_4$
$I_3 = \{2,3,5\}$ $G_3 = g_2 + g_3 + g_5$ $H_3 = h_2 + h_3 + h_5$
$Obj = -\sum_j \frac{G_j^2}{H_j+\lambda} + 3\gamma$
The smaller the score is, the better the structure is

Benchmark

I have used RandomForest Regressor to generate the Benchmark score (-0.08261). However, as part of my model analysis I have used all other Regression model and observed that XGBoost gives the best results.

III. Methodology

Data Preprocessing

Data preprocessing involved the following activities :

I. Merging data file and properties file

II. Cleaning data by dropping columns which have a missing ratio of more than 70%

   A. 25 variables were dropped as part of this process.

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

   A. bathroomcnt, calculatedbathnbr and fullbathcnt

   B. calculatedfinishedsquarefeet and finishedsquarefeet12

   C. fips, rawcensustractandblock, censustrackandblock

   D. taxvaluedollarcnt, taxamount, landvaluedollarcnt

   E. The variables with high percentage of missing values were dropped. Hence, calculatedbathnbr, fullbathcnt, finishedsquarefeet12, rawcensustractandblock, censustrackandblock, taxvaluedollarcnt, and landvaluedollarcnt were dropped.

IV. For the numeric variables with null columns, data was imputed with median values

V. For the object variables with null columns, data was imputed with -1.

VI. The target variable ‘logerror’ had a few outliers which were eliminated.

VII. Variables : ‘propertyzoningdesc’ and ‘propertycountylandusecode’ which had random values were dropped before feeding the data to the models.

VIII. As part of feature engineering new variables were added

   A. New_Transaction_Month : Month of the transaction

   B. New_LivingAreaProp : Proportion of living area derived by (calculatedfinishedsquarefeet) / (lotsizesquarefeet)

   C. New_zip_count : Number of properties in the zip.

   D. New_city_count : Number of properties in the city.

Implementation

The Model was Implemented in Python. The core steps involved

1. Importing required Libraries, provided below are the same with corresponding version.

   A. seaborn 0.7.1

   B. scipy 0.19.1

   C. scikit-learn 0.19.0

   D. pandas 0.20.3

   E. numpy 1.13.3

   F. matplotlib 2.0.2

   G. xgboost 0.6

2. Benchmarking Model : Used RandomforestRegressor
3. Data preprocessing
4. Feature Engineering
5. Model Selection

   The following models were tried with default parameters.

Sno	Model	Score Obtained
1	XGB Regression	-0.0530361127108
2	Linear Regression	-0.0533859948903
3	Bayesian Ridge Regression	-0.0533737062301
4	Ridge Regression	-0.0533859935965
5	Lasso Regression	-0.0533532241913
6	Decision Trees	-0.0846246402558
7	Random Forest	-0.0594988842515
8	KNN	-0.0615386087447
9	Gradient Boosted Regressor	-0.0530904245252

6. Model Tuning using GridSearchCV
7. Cross Validation

Some of the difficulties encountered during the coding process include :

i) Trying SVM but dropped the model as it was taking too long to be trained.

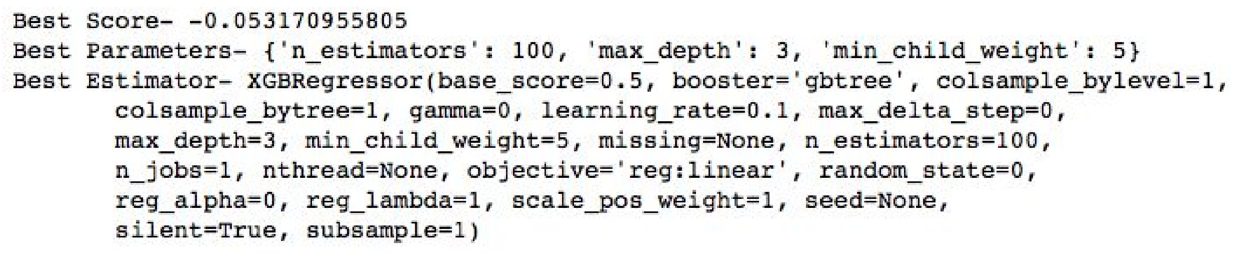
ii) Decision on variables to be dropped based on high missing values ratio...60% vs 70% vs 80%)

iii) Decision on imputing technique to be used.

Refinement

The Benchmark Model RandomForestRegressor() gave us an accuracy of -0.08261. This score after data preprocessing improved to -0.05949. But the model XGBRegressor was chosen as it had a better score of -0.05303. On GridSearch the score had a minor improvement of -0.05317. The parameter optimization was done on 3 tree-specific hyperparameters : ‘n_estimators’, ‘max_depth’ and ‘min_child_weight’. GridSearch estimated the best parameters as : n_estimators = 100 , max_depth = 3 and min_child_weight = 5. Using these parameter and upon cross validation the score improved to -0.05.

Result from Gridsearch



```
Best Score- -0.053170955805
Best Parameters- {'n_estimators': 100, 'max_depth': 3, 'min_child_weight': 5}
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=5, missing=None, n_estimators=100,
       n_jobs=1, nthread=None, objective='reg:linear', random_state=0,
       reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=None,
       silent=True, subsample=1)
```

IV. Results

Model Evaluation and Validation

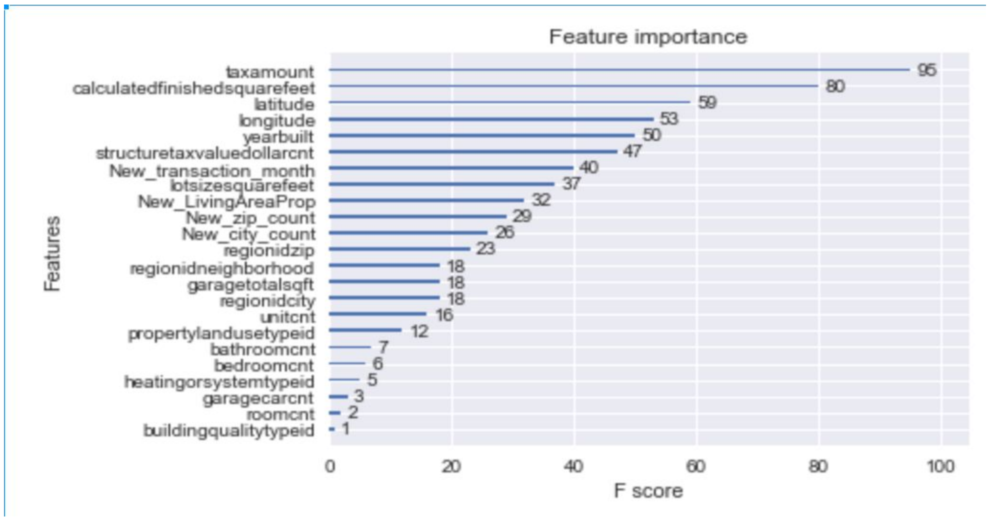
The final model was chosen after analyzing all major regression techniques, finalizing on using XGBoost, tuning the XGBoost using Grid Search and Cross Validation techniques. The robustness of the model and its solution were tested by using Cross Validation with number of folds as 10. When the cv argument 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.

Justification

The final result of -0.05 generated by XGBoost is a very good compared to -0.08261 generated by benchmark model RandomForestRegressor. The final score has parameters of learning rate as 0.1, max_depth of 3, min_child_weight of 5 and n_estimators of 100. I strongly feel that the steps followed in getting the final result add significance and validity to the final solution obtained.

V. Conclusion

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.

Reflections

This has been a very hard project to do as we don’t know the correct answers as the competition is still in progress. However, I was surprised with insights generated from the exploratory data analysis of data, and application of different regression models and techniques such as gridsearch and cross-validation. These steps helped me in tuning my code to come up with better scores and also helped me in better understanding of the problem.

Improvement

Some of the improvements i wanted to use by could not due to time and lack of knowhow, and also think that the new model generated might be better than my model

i) Improve algorithm by using KNN to impute missing values

ii) Wanted to try LightGBM, Catboost algorithms and compare with XGBoost

iii) Use CNN and ensemble it with LightGBM, Catboost algorithms and XGBoost