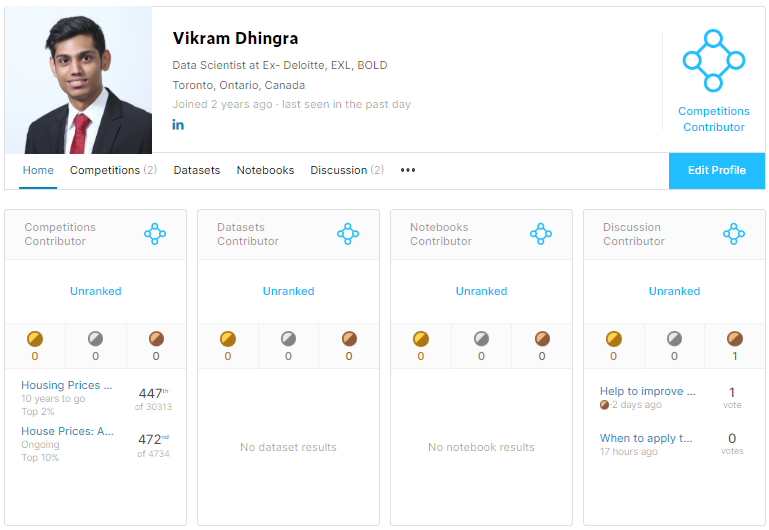
**MMA 867 Individual Assignment 1**

**House prices prediction using regression**



**Kaggle Name: Vikram Dhingra (**<https://www.kaggle.com/vikramdhingra>**)**

**Github repo:** <https://github.com/vikramdhingra/kagglehousing>

**Total teams:** 4734

**My position:** 472**(~top 10%)**

**RMSLE:** 0.11655

**Technique:** GLM with elastic net

**Abstract**

I identified the following three regression competition on Kaggle:

* All state claim: <https://www.kaggle.com/c/allstate-claims-severity/overview>

There are 116 categorical variable and 14 continuous. We have to predict the loss of the insurance claim. Train has ~188K rows with 132 columns. The problem with this competition is that there is no description of the variables.

* Predict Future Sales: <https://www.kaggle.com/c/competitive-data-science-predict-future-sales/data>

There are 9 variables around item sales with respect to time. The task is to predict monthly future sales. The time will be a major regressor for this model and gbm may be the best algorithm to solve this.

* House Prices: Advanced Regression Techniques: <https://www.kaggle.com/c/house-prices-advanced-regression-techniques>

There are 1460 observation with 80 variables in the train and have to predict the sale price of the 1459 houses in the test. The competition is ongoing on Kaggle and is a good use case of regression.

I choose ‘House Prices’ competition on Kaggle for 867 assignment. Below are the reasons:

* Data dictionary for the dataset is well maintained and is helpful for creating new features and model development.
* Dataset is not very large in size and we can easily do exploratory data analysis.
* Good use case of lasso and ridge regression that is very relevant to the course.

1. **Introduction to ‘House Prices’ and its data dictionary**

Housing price competition aims at predicting sale prices for the houses in ‘Ames city’ in Story County, Iowa, United States. The dataset is divided into two halves: testing and training (~50-50%). There are 80 variables in the training dataset. Below is the succinct summary:

Y: Sale Price for the house.

X: Initial variables:

|  |  |  |
| --- | --- | --- |
| **ID** | **Variable** | **Description** |
| 1 | MSSubClass | Identifies the type of dwelling involved in the sale. |
| 2 | MSZoning | Identifies the general zoning classification of the sale. |
| 3 | LotFrontage | Linear feet of street connected to property |
| 4 | LotArea | Lot size in square feet |
| 5 | Street | Type of road access to property |
| 6 | Alley | Type of alley access to property |
| 7 | LotShape | General shape of property |
| 8 | LandContour | Flatness of the property |
| 9 | Utilities | Type of utilities available |
| 10 | LotConfig | Lot configuration |
| 11 | LandSlope | Slope of property |
| 12 | Neighborhood | Physical locations within Ames city limits |
| 13 | Condition1 | Proximity to various conditions |
| 14 | Condition2 | Proximity to various conditions (if more than one is present) |
| 15 | BldgType | Type of dwelling |
| 16 | HouseStyle | Style of dwelling |
| 17 | OverallQual | Rates the overall material and finish of the house |
| 18 | OverallCond | Rates the overall condition of the house |
| 19 | YearBuilt | Original construction date |
| 20 | YearRemodAdd | Remodel date (same as construction date if no remodeling or additions) |
| 21 | RoofStyle | Type of roof |
| 22 | RoofMatl | Roof material |
| 23 | Exterior1st | Exterior covering on house |
| 24 | Exterior2nd | Exterior covering on house (if more than one material) |
| 25 | MasVnrType | Masonry veneer type |
| 26 | MasVnrArea | Masonry veneer area in square feet |
| 27 | ExterQual | Evaluates the quality of the material on the exterior |
| 28 | ExterCond | Evaluates the present condition of the material on the exterior |
| 29 | Foundation | Type of foundation |
| 30 | BsmtQual | Evaluates the height of the basement |
| 31 | BsmtCond | Evaluates the general condition of the basement |
| 32 | BsmtExposure | Refers to walkout or garden level walls |
| 33 | BsmtFinType1 | Rating of basement finished area |
| 34 | BsmtFinSF1 | Type 1 finished square feet |
| 35 | BsmtFinType2 | Rating of basement finished area (if multiple types) |
| 36 | BsmtFinSF2 | Type 2 finished square feet |
| 37 | BsmtUnfSF | Unfinished square feet of basement area |
| 38 | TotalBsmtSF | Total square feet of basement area |
| 39 | Heating | Type of heating |
| 40 | HeatingQC | Heating quality and condition |
| 41 | CentralAir | Central air conditioning |
| 42 | Electrical | Electrical system |
| 43 | 1stFlrSF | First Floor square feet |
| 44 | 2ndFlrSF | Second floor square feet |
| 45 | LowQualFinSF | Low quality finished square feet (all floors) |
| 46 | GrLivArea | Above grade (ground) living area square feet |
| 47 | BsmtFullBath | Basement full bathrooms |
| 48 | BsmtHalfBath | Basement half bathrooms |
| 49 | FullBath | Full bathrooms above grade |
| 50 | HalfBath | Half baths above grade |
| 51 | Bedroom | Bedrooms above grade (does NOT include basement bedrooms) |
| 52 | Kitchen | Kitchens above grade |
| 53 | KitchenQual | Kitchen quality |
| 54 | TotRmsAbvGrd | Total rooms above grade (does not include bathrooms) |
| 55 | Functional | Home functionality (Assume typical unless deductions are warranted) |
| 56 | Fireplaces | Number of fireplaces |
| 57 | FireplaceQu | Fireplace quality |
| 58 | GarageType | Garage location |
| 59 | GarageYrBlt | Year garage was built |
| 60 | GarageFinish | Interior finish of the garage |
| 61 | GarageCars | Size of garage in car capacity |
| 62 | GarageArea | Size of garage in square feet |
| 63 | GarageQual | Garage quality |
| 64 | GarageCond | Garage condition |
| 65 | PavedDrive | Paved driveway |
| 66 | WoodDeckSF | Wood deck area in square feet |
| 67 | OpenPorchSF | Open porch area in square feet |
| 68 | EnclosedPorch | Enclosed porch area in square feet |
| 69 | 3SsnPorch | Three season porch area in square feet |
| 70 | ScreenPorch | Screen porch area in square feet |
| 71 | PoolArea | Pool area in square feet |
| 72 | PoolQC | Pool quality |
| 73 | Fence | Fence quality |
| 74 | MiscFeature | Miscellaneous feature not covered in other categories |
| 75 | MiscVal | $Value of miscellaneous feature |
| 76 | MoSold | Month Sold (MM) |
| 77 | YrSold | Year Sold (YYYY) |
| 78 | SaleType | Type of sale |
| 79 | SaleCondition | Condition of sale |

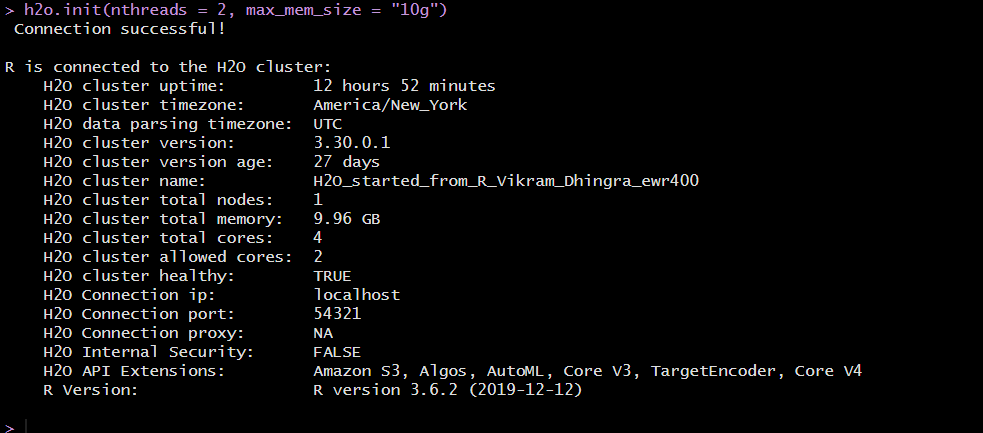
The rich data dictionary will help us to better understand the variables and to create new features.

1. **Loading data and packages**

For this project, we used the following packages:

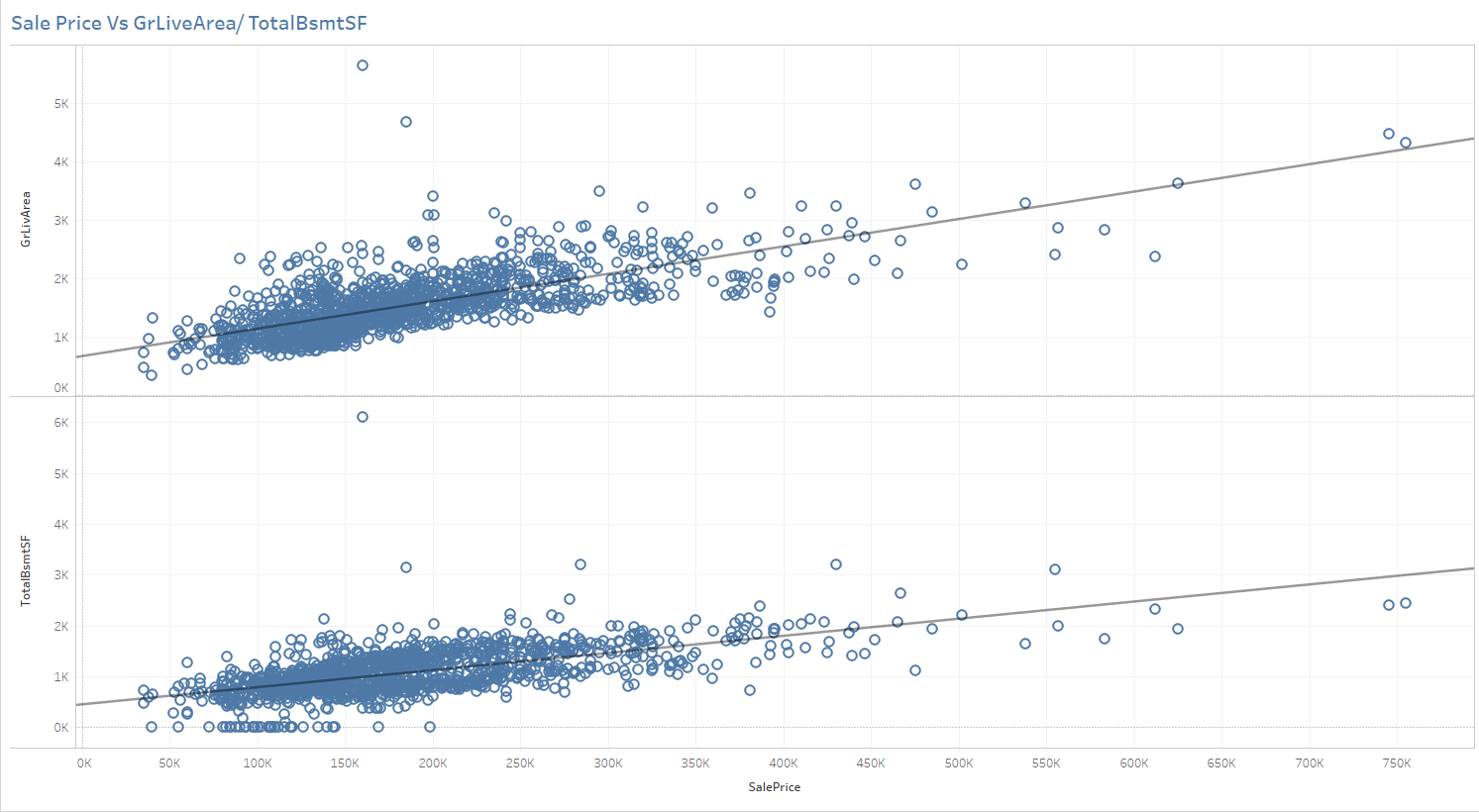
* data.table: for fast reading csv
* dplyr: for data manipulation
* mice: for missing data handling
* h2o: open source machine learning framework that offers ‘parallel computing’

I used h2o to substantially decrease cross validation runtime. H2o is approximately 10 times faster than glmnet. Furthermore, I invested more time in trying different combinations to get the best prediction. 



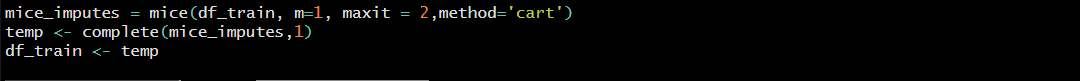
1. **Handling outliers and missing values**

As a part of exploratory analysis, I used tableau for quick insights in the data. I loaded the train data in the tableau. I plotted ‘SalePrice’ on the X and other dimension on the Y on the scatterplot. I plotted all the dimension to see the behavior with the ‘SalePrice’. Below is the screenshot of GrLiveArea and TotBsmtSF.



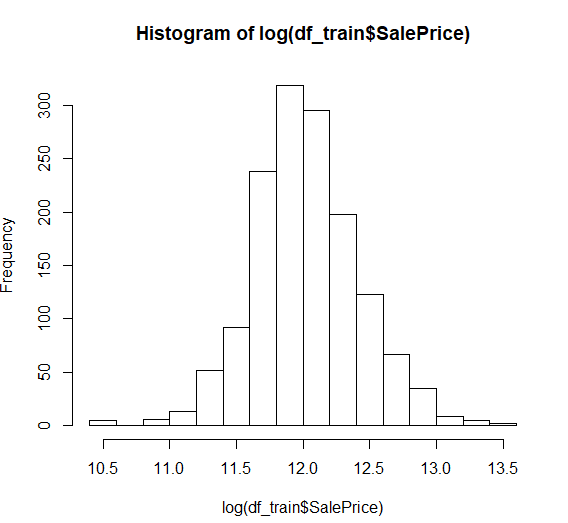
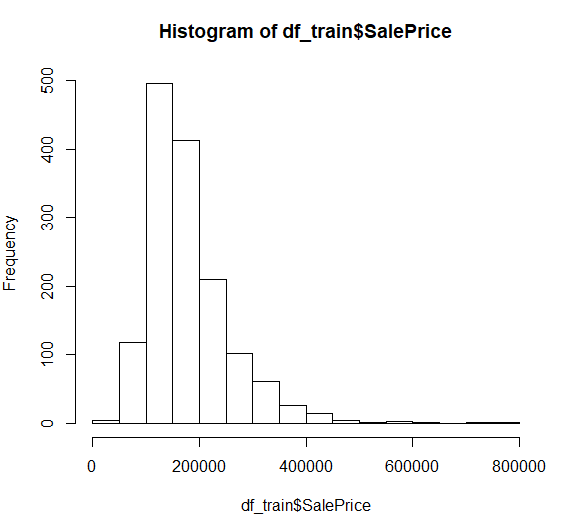
Visualization is best to spot outliers. I realized that there are two outliers in the data. I removed these two outliers from the data. While I was trying multiple models, I modelled with and without outliers. The model without outliers was significantly better than the other.

I converted all the character columns to factor columns. For missing data, I used MICE package and imputed the dataframe using ‘cart’ method and setting max iteration to 2. I used cart as there were missing categorical variables in the dataframe.



1. **Feature engineering**

I plotted the histogram of the ‘SalePrice’ and observed that is it is skewed towards the left. It is understandable as majority of houses should be affordable.



I transformed the saleprice to log of it. I will be cautious of putting exponential in the prediction to get back the predicted price.

I created multiple derived variables. Below is the summary of derived variables:

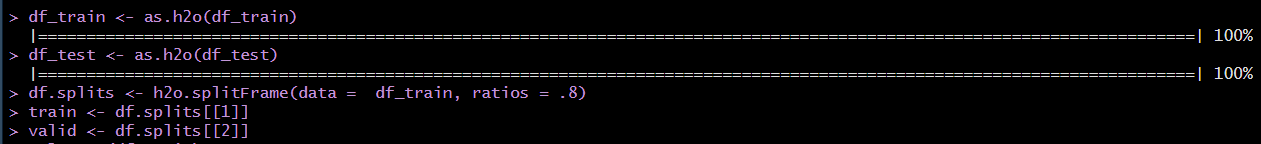
1. House Age: Number of years from YrBuilt. Formula: 2020 – YrBuilt
2. Total Floor SF: Total surface area of floors. 1 Floor + 2 Floor SF
3. Total Bathroom: FullBath+ HalfBath+ BsmtHalfBath+BsmtFullBath
4. Total Area: Total above ground area + Total Basement SF + Total Floor SF
5. Total Porch Area: OpenPorchSF + EnclosedPorch + third\_SsnPorch + ScreenPorch
6. Is house remodeled: Flag 1 if house is remodeled else zero
7. Remodeled Age: Year sold - Remodel date
8. Recession: Average price before 2007 is higher than after 2007. 1 if YrSold is 2008,2009 or 2010 else 0.

Apart from the above 7 variables, I created log(x)/ log(x+1) transformation for 16 variables namely: lg\_totFlrSF, lg\_house\_age, lg\_totbath, lg\_tot\_ar, lg\_tot\_porch, lg\_Age, lg\_MiscVal, lg\_MasVnrArea, lg\_LotArea, lg\_LotFrontage, lg\_ltarea\_front, lg\_BsmtFinSF1, lg\_BsmtUnfSF, lg\_TotRmsAbvGrd, lg\_WoodDeckSF and lg\_OpenPorchSF.

In total, I created 23 new variables. Few these variables like house age and area will be very significant and will make our prediction model more accurate. I converted all the character columns to factor and converted MsSubClass to factor as well.

1. **Modelling**

For modelling in h2o, I need to load everything in h2o environment.



I split the training data into 2 parts, 80% training and 20% validation. The validation frame will be used checking the accuracy. I used h2o.glm() function from the h2o package. The function is extremely powerful as cross validation and best lambda can be achieved in a single command.

H2o.glm() has the following parameters:

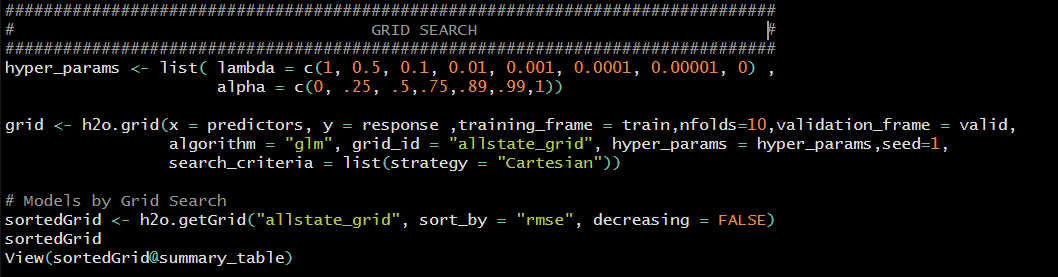
* X = Vector of all the predictors
* Y = Response or dependent variable,
* Training\_frame = Training h2o frame
* Validation\_frame = Validation h2o frame
* Alpha = Value between 0 to 1
* Nfolds = 10
* Lambda = 0.001
* Seed = any number
* Lambda\_search: Set to TRUE. H2o will look for best lambda based on lambda min ratio.
* Interactions: Vector of columns that will interact pairwise among each other

I ran the first model without any inferred variables, missing value imputation and outlier deletion. This gave me a benchmark to start and think on various results. RMSLE for this model is 0.14552

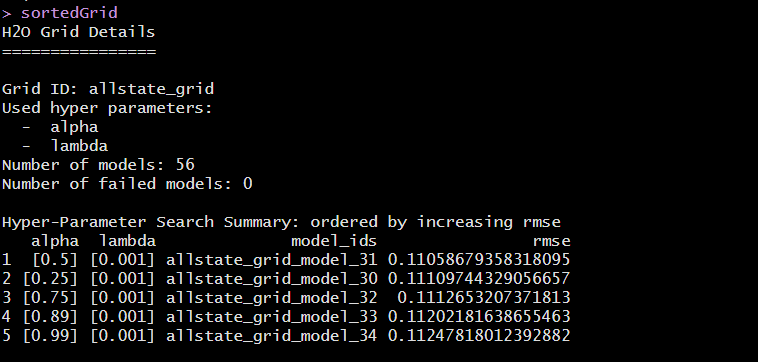


I then added cross validation, nfolds=10 and then decided to tune hyper parameters using H2o.grid() function. Grid search will return the list of all the models of lamda \* alpha combination. Hyperparameters used:

* Lambda: 1, 0.5, 0.1, 0.01, 0.001, 0.0001, 0.00001, 0
* Alpha: 0, .25, .5,.75,.89,.99,1 (Elastic Net)



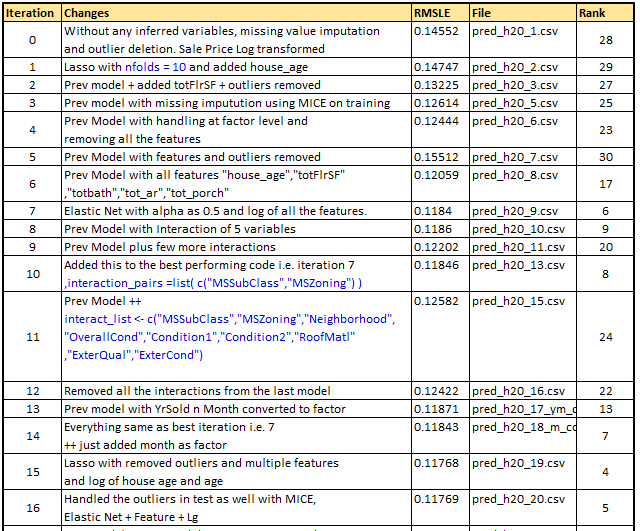
Grid search helps to tune the model and get the best with the current data.

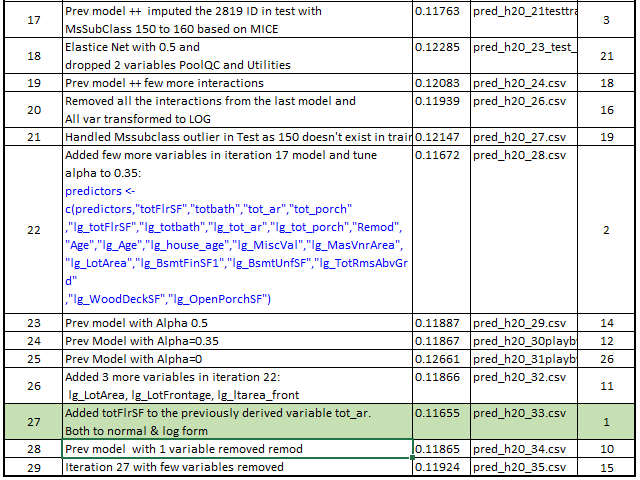


Now, we have benchmark to start, we will carefully add derived variables and see the change in the RMSLE. In the process of getting the best model, I started with variables that are most relevant to the ‘SalePrice’ and then added other variables. My standard operating procedure for making this model better is:

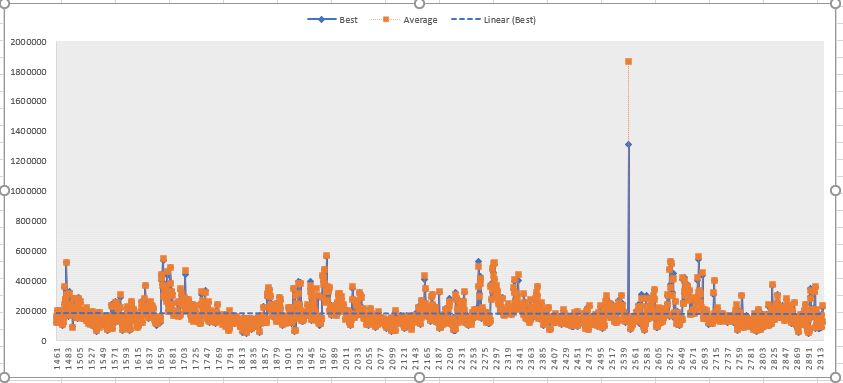
* Add the more relevant variables
* Check for transformation in current variables
* Check for any interactions
* Tune the value of Alpha
* Tune the value of Lambda

Below is the succinct summary of each iteration:

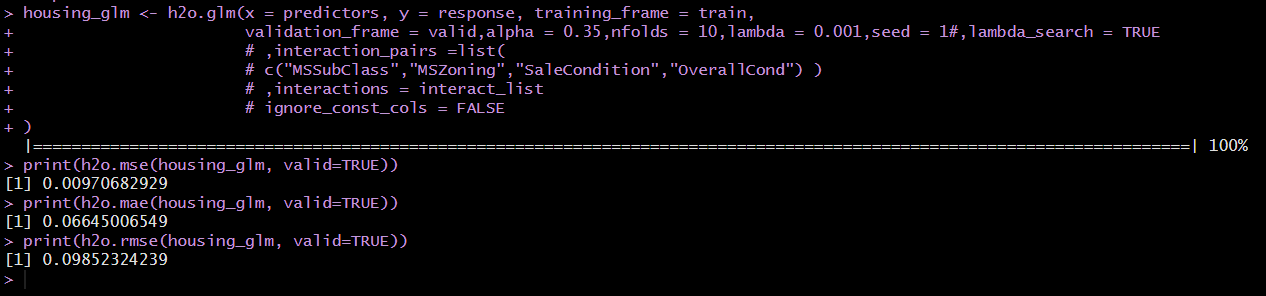


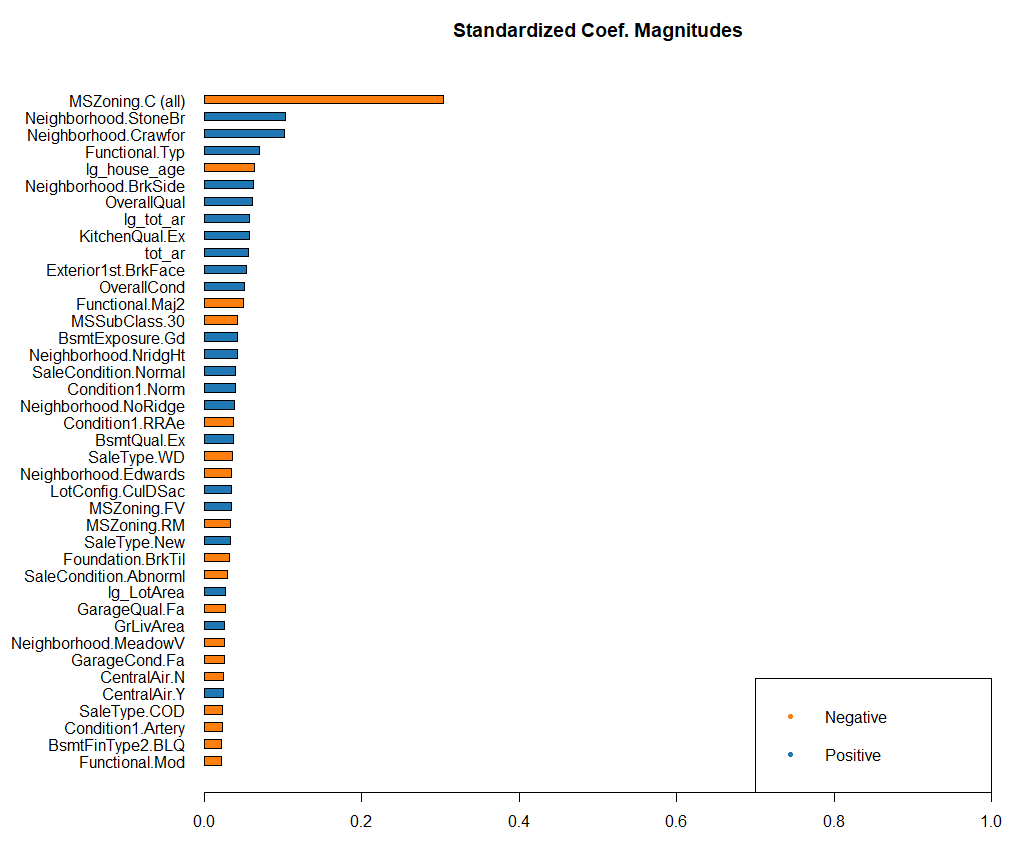


Furthermore, I plotted the results from best (0.11655) and average (0.12444) model to understand the prediction behavior. I observed that for id 2550, results differ a lot. This is the id that we need to take care in the testing dataset. As per my calculated guess, it should be around ~600000.



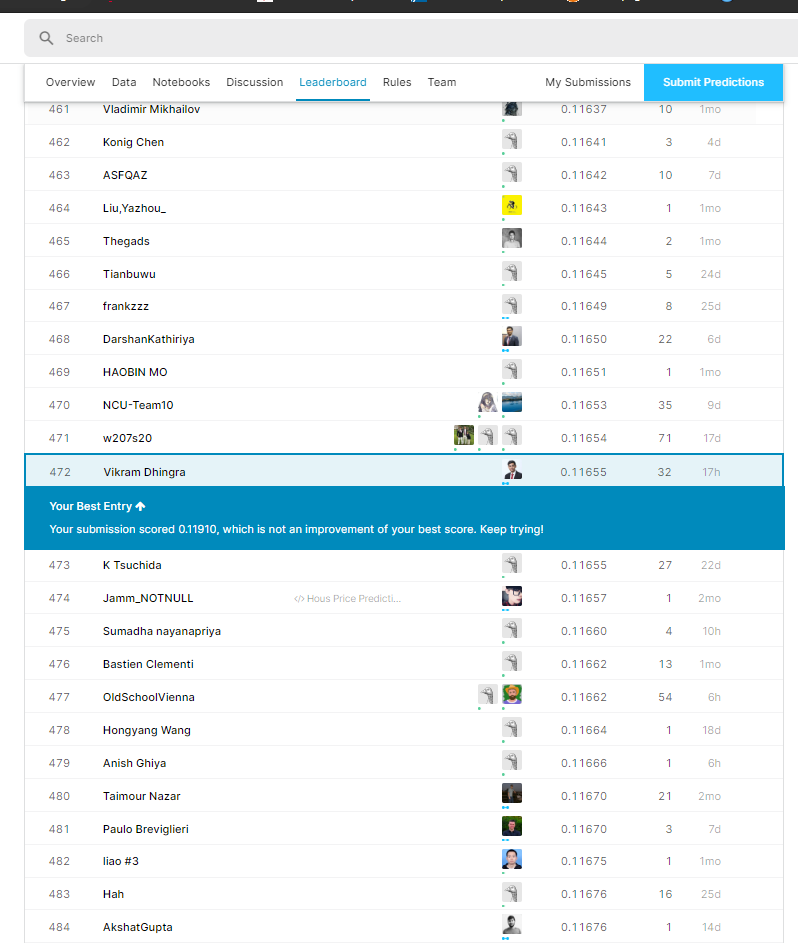
Our best model with RMSLE of 0.11655 is elastic net model with alpha =0.35 and lambda= 0.001 with 101 variables and below are the standardized coefficients magnitude for the same.

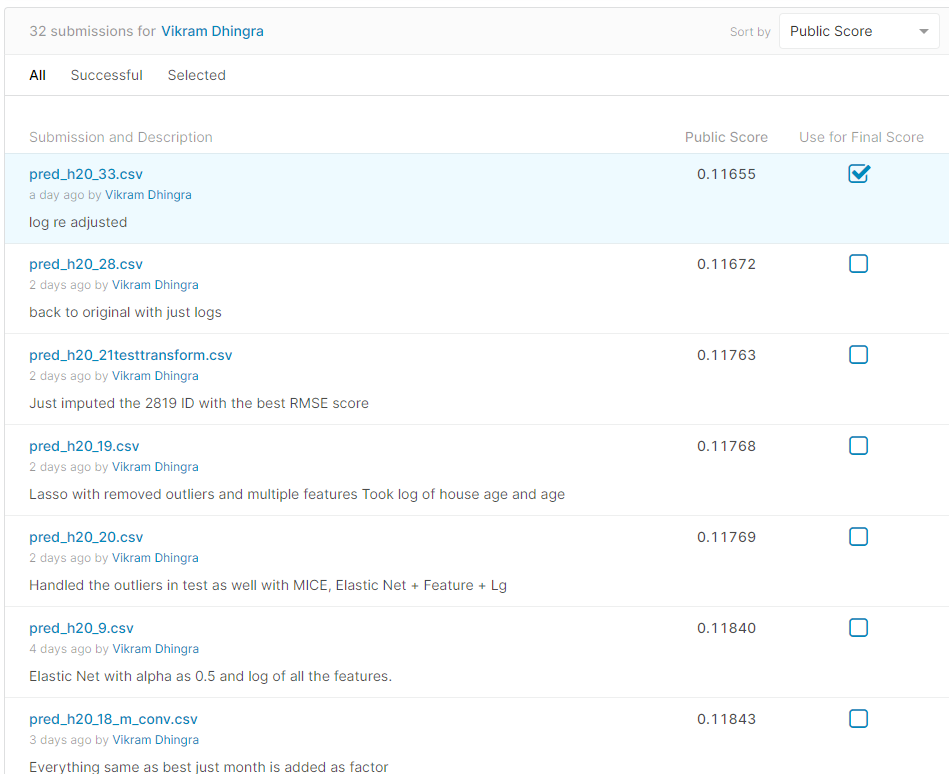




**Appendix**

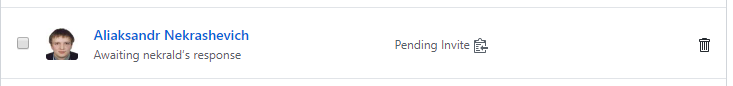
**Kaggle Screenshot:**





**Github Repo for best model code and all submitted csv:** <https://github.com/vikramdhingra/kagglehousing>

Access has been granted to nekrald on github. Please let me know if you are not able to access.



**References:**

1. H2O documentation: <http://docs.h2o.ai/h2o/latest-stable/h2o-docs/data-science/glm.html>
2. Kaggle Discussions: <https://www.kaggle.com/c/house-prices-advanced-regression-techniques/discussion> . Few thoughts have been derived from discussions.