

**MH 8111**

**Group Project**

**House Prices: Advanced Regression Techniques**

|  |  |  |
| --- | --- | --- |
| Prepared by: | Chen Jun Xiang | G1800609E |
| Gayus Yonathan | G1800636H |
| Low Chuan Peng | G1800662C |
| Tan Ying Li Cheryl | G1800663L |
| Submitted to: | A/Professor Li Xiaoli | NTU / SPMS |

**Table of Contents Page**

[**Problem Statement**](#_kxvi2wsyyois) **3**

[**Data Description**](#_jk4crbn39xjs) **3**

[**Methodology and Implementation**](#_5ymh1pp6hpg5) **3**

[Methodology](#_ufkcb88qf5d8) 3

[Data Imputation](#_si8qi8n9ag8k) 4

[Data Transformation](#_68wqm52oha14) 6

[Feature Engineering](#_9xrkripk36mf) 7

[Implementation](#_5xjrh0sv06i6) 10

[Regularised Regression](#_i95hjlh2s7gp) 10

[Random Forest](#_r51g1wxuknv5) 12

[XGBoost](#_404rysh0geqc) 13

[**Results and Discussions**](#_585v3mxsvrvy) **13**

[Analysis, Discussions and Interesting Findings](#_hq1b8ychm6x4) 13

[**Comparison**](#_xrp8ojsbd0dd) **19**

[**Conclusion**](#_crypd92yogry) **20**

[**Appendix**](#_wxvee2j91qm9) **21**

[**References**](#_khy6qgkgoaqc) **23**

## 

## Problem Statement

Purchasing a property is probably one of the most important and expensive decision for a large number of people, especially if these people reside in countries where real estate prices are significantly higher. As such, the notion of a “dream house” is ingrained deeply in many homeseekers, in order for them to justify the cost of the house they will be purchasing. However, if you ask a home buyer to describe how would their “dream house” be like, majority of them would likely miss out specific details like the height of the basement or the type of foundation used to build the house. Although these specific details are not explicitly mentioned during prices negotiations, these factors might in turn have significant impacts on the final prices of the house purchased. Understanding the various correlations between the multiple aspects of a house and the final price that particular house fetches can bring forth benefits to parties in a housing transactions. In a typical example, a real estate agent can push for higher prices that in turn lead him/her raking in higher commissions and higher profit margins for home sellers. Also, a home seeker can make more informative decisions before investing in a house.

Hence, the above-mentioned points motivate one to embark on the Kaggle competition - **House Prices: Advanced Regression Technique**. The purpose of this competition is to analyse almost every aspect of the residential homes in Ames, Iowa, USA, in order to predict the final price of each home.

## Data Description

In this dataset, 79 explanatory characters or numeric variables such as the facilities present, type of road access, type of foundation, house style, neighborhood, condition of sale and month and year sold, are provided to describe homes in Ames, Iowa.

The training set “train.csv” consists of the above-mentioned 79 variables for 1460 houses as well as the ID and the property’s sale price in dollars. On the other hand, the test set “test.csv” consists of the same 79 variables for 1459 houses and their corresponding IDs. The training set was used to train the machine learning algorithm and the target of this competition is to predict the sales price (aka “SalePrice”) for each house in the test set.

## Methodology and Implementation

# **Methodology**

For this project, 4 machine learning algorithms, namely **Regularised Regression (including Ridge and Lasso)**, **Random Forest** and **XGBoost** were evaluated to decide which method will provide the most accurate predicted Sale Price by calculating the Root Mean Squared Error (RMSE). Prior to implementing the machine learning algorithms, however, the following data preparations were done:

1. Combine both training and test sets using rbind() such that both sets can be subjected to the same data treatment.
2. Removing the ID column as it does not provide useful information.
3. Check the variables for missing data. Table 1 shows the category and its corresponding number of missing values. In addition to the categories listed in Table 1, there were also 1459 missing values in “SalePrice” which matches the number of instances in the test set. After which each category in Table 1 was individually accessed and sound methodologies were proposed to replace the missing values. Refer to the following subsections for the respective methods used for fixing the missing values.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Category** | **No.** | **Category** | **No.** | **Category** | **No.** |
| PoolQC | 2909 | BsmtCond | 82 | Exterior1st | 1 |
| MiscFeature | 2814 | BsmtExposure | 82 | Exterior2nd | 1 |
| Alley | 2721 | BsmtQual | 81 | BsmtFinSF1 | 1 |
| Fence | 2348 | BsmtFinType2 | 80 | BsmtFinSF2 | 1 |
| SalePrice | 1459 | BsmtFinType1 | 79 | BsmtUnfSF | 1 |
| FireplaceQu | 1420 | MasVnrType | 24 | TotalBsmtSF | 1 |
| LotFrontage | 486 | MasVnrArea | 23 | Electrical | 1 |
| GarageYrBlt | 159 | MSZoning | 4 | KitchenQual | 1 |
| GarageFinish | 159 | Utilities | 2 | GarageCars | 1 |
| GarageQual | 159 | BsmtFullBath | 2 | GarageArea | 1 |
| GarageCond | 159 | BsmtHalfBath | 2 | SaleType | 1 |
| GarageType | 157 | Functional | 2 |  |  |

*Table 1 summarises the number of missing values in each category.*

# Data Imputation

The variables split into 3 types for data imputation which are ordinal values, nominal values and numeric (ratio) values. In order to decide if a particular variable is ordinal or nominal, the following code is used:

|  |
| --- |
| *attach(<name\_of\_dataset>)*  *aggregate(<name\_of\_dataset>[,c("<variable>","SalePrice")],by=list(<variable>),FUN=mean,na.rm=TRUE)* |

If the tibble output shows a correlation between the values and the mean sale price, the variable is said to be ordinal, if not the value will be considered to be nominal. Refer to Table 2 for the data type for each feature.

|  |  |  |
| --- | --- | --- |
| **Ordinal** | **Nominal** | **Ratio** |
| PoolQC | MiscFeature | LotFrontage |
| FireplaceQu | Alley | MasVnrArea |
| GarageYrBlt | Fence | BsmtFullBath |
| GarageFinish | Garage Type | BsmtHalfBath |
| GarageQual | MSZoning | BsmtFinSF1 |
| GarageCond | Functional | BsmtFinSF2 |
| BsmtCond | Exterior1st | BsmtUnfSF |
| BsmtExposure | Exterior2nd | TotalBsmtSF |
| BsmtQual | Electrical | GarageCars |
| BsmtFinType2 | SaleType | GarageArea |
| BsmtFinType1 |  |  |
| MasVnrType |  |  |
| Utilities |  |  |
| KitchenQual |  |  |

*Table 2 summarises the data type in each feature.*

Generally, for ordinal data types, the variable will be converted to the ordinal class, e.g for PoolQC, the values were converted as follows:

|  |
| --- |
| *# Converting ordinal variables to ordinal class*  *values = c("No Pool" = 0,"Po" = 1,"Fa" = 2,"TA" = 3,"Gd" = 4,"Ex" = 5)*  *all$PoolQC = as.integer(revalue(all$PoolQC, values))* |

For nominal data types, those values with NA will be evaluated as either (1) the house having no such feature or (2) missing data for that particular feature. If (1) occurs, the NA will be replaced by “None” or “No *specific feature*” before converting the variables into factors, e.g. the NA values in “MiscFeature” indicates that that particular house had no additional facilities and hence all the NA were replaced by None before converting into factor form:

|  |
| --- |
| *# The NAs are due to the house having no other additional facilities*  *# Replacing 'NA' with 'None' and this feature should be in factor form.*  *all$MiscFeature[is.na(all$MiscFeature)] = 'None'*  *# Convert the variable to factor class*  *all$MiscFeature = as.factor(all$MiscFeature)* |

If (2) occurs, the NA will be replaced by the mode of the feature before converting it to factor form, e.g. For “MSZoning”, it is not possible for a house to be without the general zoning classification of the sale, hence it is likely that the NA is due to missing data. Hence, the mode of “MSZoning” was obtained using the getmode() to replace the NA in “MSZoning” and after which the values in “MSZoning” were converted to factors as shown below:

|  |
| --- |
| *# Input missing values with mode and convert to factor*  *all[is.na(all$MSZoning), "MSZoning"] = getmode(all$MSZoning)*  *all$MSZoning = as.factor(all$MSZoning)* |

For numeric (ratio) features, a similar approach as for the nominal data types were used, i.e. if the house had that particular feature, the NA would be replaced by the mode of that feature and if the house did not have that feature, the NA would be replaced by 0. One notable exception was the feature “LotFrontage”. For “LotFrontage”, the values of the “LotFrontage” were grouped by each neighbourhood to calculate the mean of the “LotFrontage” in each neighbourhood. Then, the NA values of the “LotFrontage” were replaced by the mean of that in each neighbourhood.

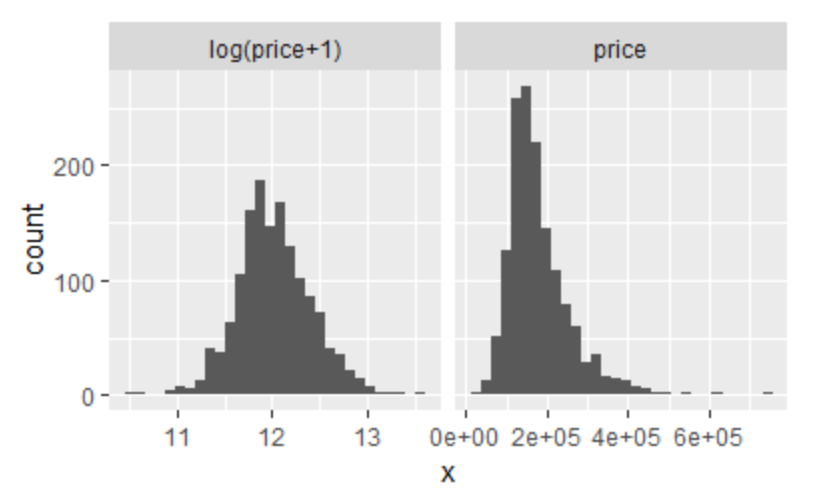
For more detailed data treatment of each variable, refer to the attached R Codes.

# Data Transformation

After all the missing data were imputed, the pairs of variables that were highly correlated with each other were examined and one of each pair that had the lowest correlation with “SalePrice” was dropped. For example, “GarageCars” and “GarageArea” had high correlation (which is logical as given a larger garage area, more cars could be placed) and since “GarageArea” had a lower correlation with the “SalePrice”, “GarageArea” was removed and “GarageCars” was kept. In total, 7 such pairs of variables were observed and 7 variables namely, "YearRemodAdd", "GarageYrBlt", "GarageArea", "GarageCond", "TotalBsmtSF", "TotRmsAbvGrd" and "BsmtFinSF1" were dropped.

Next, the numeric variables and the factor variables were separated to do data treatment for each set of variables.

For the numeric variables, some of the variables were found to be skewed (see diagram on the right) and hence these variables were required to be corrected using the logarithmic transformation (see diagram on the left). After correcting the skew, these numeric data were normalised.



|  |
| --- |
| *Figure 1. Illustrates the skewed response variable before and after the application of logarithmic function.* |

Lastly, the factor variables were converted into numeric which was required by machine learning models by using model.matrix() in r.

Before implementation, the training and test sets were separated.

# Feature Engineering

4 new features were engineered. They were created to account for the number of bathrooms within the house, the age of the house, whether the house has been remodeled, is it a newly built house and the total area within the house.

* 1. *Total Number of Bathrooms*

There are 4 variables related to bathroom. They are the number of full bathrooms above grade (“FullBath”), half bathrooms above grade (“HalfBath”), full bathrooms at basement (“BsmtFullBath”) and half bathrooms at basement (“BsmtHalfBath”). A half bathroom has two of the four full bathroom components-typically a toilet and sink. Individually, these variables are not highly correlated with the Sale Price. As most buyers would focus more on the total number of bathrooms than each bathroom’s individual specification, we combined these features in an attempt to create a stronger correlated predictor. A factor of half was multiplied to the number of half bathrooms, followed by the sum of it with the number of full bathrooms.

|  |
| --- |
| *# Multiply half value (0.5) to half bathrooms and full value (1.0) to full bathrooms*  *all$Total\_Bathrooms = all$BsmtFullBath + 0.5\*all$BsmtHalfBath + all$FullBath + 0.5\*all$HalfBath* |

* 1. *Age of House , Remodeled House, New House*

There are 3 variables related to the age of the house (“Age”). These are the year in which the house was built (“YearBlt”), the year where remodelling was done to the house (“YearRemodAdd”) and the year the house was sold (“YrSold”). Since in data imputation, we had replaced the “NA” in YearRemodAdd with YearBuilt. We derived the “Age” using the following code:

|  |
| --- |
| *# The age of the house is determined by subtracting the year it was sold with the year it was*  *# remodeled. Previously, the YearRemodAdd column has been adjusted such that those with # NA were replaced with the year it was built.*  *all$Age = as.numeric(all$YrSold - all$YearRemodAdd)* |

The next variable created was to complement the feature “Age”. In most cases, old constructions are retained and only parts of the house are renovated. Remodeled houses are usually less pricer compared to new houses.This feature engineered helps to differentiate remodelled from new houses.

|  |
| --- |
| *# creating a new feature that indicates that whether the house has been remodeled*  *all[all$YearBuilt == all$YearRemodAdd, "Remodeled"] = 0*  *all[all$YearBuilt != all$YearRemodAdd, "Remodeled"] = 1* |

Furthermore, we created a feature to distinguish new houses from the old houses.

|  |
| --- |
| *# If the house is new, the year it was sold and built should the same*  *all[all$YearBuilt == all$YrSold, "New"] = 1*  *all[all$YearBuilt != all$YrSold, "New"] = 0* |

* 1. *Neighborhood Classification*

The houses in different neighborhoods were usually priced differently. For example, in Singapore, the houses in the central district are usually pricer compared to most other districts. We first computed the mean of the price of houses in each neighborhood.

|  |
| --- |
| *# Find the mean of the price of the houses in each neigborhood*  *sort(tapply(all$SalePrice[1:1460], all$Neighborhood[1:1460], mean), decreasing = TRUE)*  *table(all$Neighborhood)* |

This was followed by setting the neighborhoods into 5 different categories according to the tabulated mean prices of the houses in different neighborhoods.

|  |
| --- |
| *# Create 5 categories out of it*  *all[all$Neighborhood == "NoRidge"|all$Neighborhood == "NridgHt" | all$Neighborhood == "StoneBr" | all$Neighborhood == "Timber" | all$Neighborhood == "Veenker", "Neighborhood\_Category"] = 4*  *all[all$Neighborhood == "Somerst"|all$Neighborhood == "ClearCr" | all$Neighborhood == "Crawfor"|all$Neighborhood == "CollgCr" | all$Neighborhood == "Blmngtn", "Neighborhood\_Category"] = 3*  *all[all$Neighborhood == "Gilbert"|all$Neighborhood == "NWAmes" | all$Neighborhood= = "SawyerW" | all$Neighborhood == "Mitchel" | all$Neighborhood == "NAmes", "Neighborhood\_Category"] = 2*  *all[all$Neighborhood == "NPkVill"|all$Neighborhood == "SWISU" | all$Neighborhood == "Blueste" | all$Neighborhood == "Sawyer" | all$Neighborhood == "OldTown", "Neighborhood\_Category"] = 1*  *all[all$Neighborhood == "Edwards" | all$Neighborhood == "BrkSide" | all$Neighborhood == "BrDale" | all$Neighborhood == "IDOTRR" | all$Neighborhood == "MeadowV", "Neighborhood\_Category"] = 0* |

* 1. *Total Living Area*

The total living area (“Total\_Living\_Area”) could possibly be one of the factors that buyers consider as well when choosing a house. Hence, we created this feature by summing the living area above grade (“GrLivArea”) and at basement (“TotalBsmtSF”).

|  |
| --- |
| *# Total area within the house*  *all$Total\_Living\_Area = all$GrLivArea + all$TotalBsmtSF* |

# **Implementation**

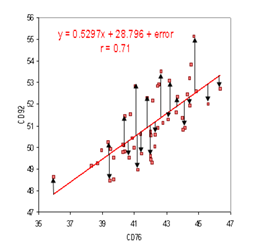
# After data preparation was completed, 4 machine learning algorithms, namely **Regularised Regression (Ridge and Lasso)**, **Random Forest** and **XGBoost** were implemented for prediction of the Sale Price.

# Regularised Regression

Regression is explaining the relationship between numeric variables in a dataset. Variables can be divided into two groups, which are dependent variable (variable will be predicted) and independent variables (predictors). The simplest form of regression is a linear straight line which can be represented in mathematical form as

y=α+βx, α=slope intercept, β= slope

In another model of regression, such as multilinear regression α is considered as residual or sum of squares of error (shown in figure 2) between the real value of data and regression model fitted. This error is sometimes expressed in RMSE (Root Mean Square Error), is the more preferable representation to explain, how different it is the regression model from the actual value. In this method, **Lasso Regression** and **Ridge Regression** are used. Bothhave regularized regression analytics technique which enhances the accuracy of prediction. Lasso Regression is type of linear regression and is developed from initial of Least Square method has the extensive relationship with Ridge Regression. Both Ridge and Lasso are categorized as Regularized Regression and Lasso are further shrinking the operator.



*Figure 2. Illustrates the plot of residual error vs actual points.*

Regularized Regression is the way to avoid overfitting model into data by penalizing high-value coefficient on more complex data. Lower coefficient tends to have a better predictive power of the dependent variable. It is necessarily be done on data due to minimizing residual in ordinary regression become unstable, especially when the model becomes complex and multicollinear.

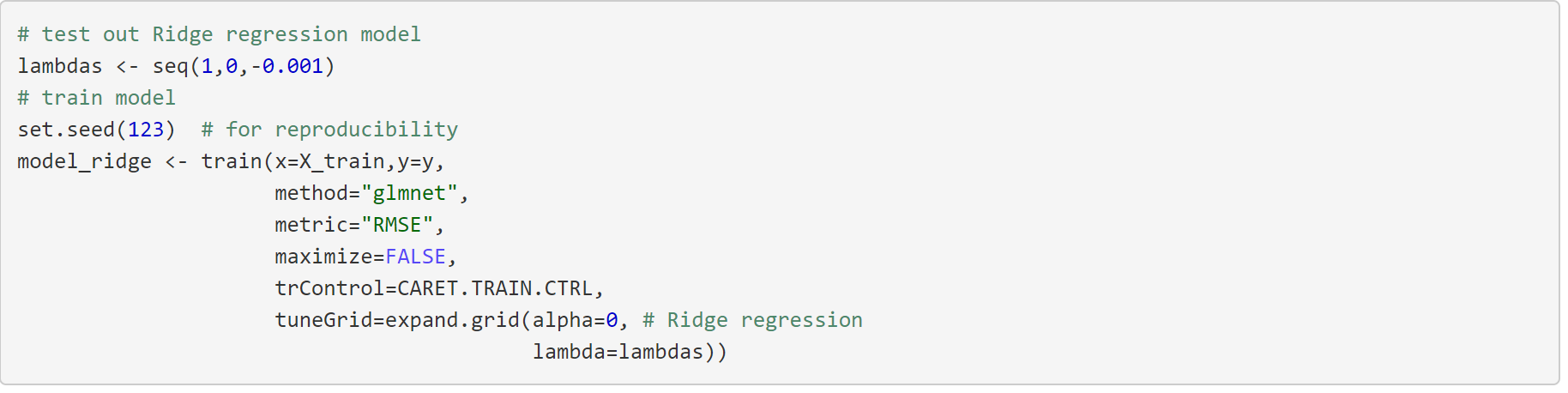
For example, there are three lines to fit 2 data points (straight line, polynomial line and logarithmic line), a straight line may be underfitting the data. In addition, if there is the third point which fits the model, it cannot be assumed to be applying the straight line will pass through the third point. In short, it is not suitable to fit small data as it will make the design to overfit. While underfit may give a sparse performance of predictions. Compensating regularization, there are two rules to be applied to this dataset. L1 regularization (Lasso Regression) limit the size of the coefficient, it will give penalty by making it equal to the absolute magnitude (zero or eliminating the variables). L2 regularization (Ridge Regression), a non-eliminating method by making high valued coefficients into the square of the magnitude of coefficients, all coefficients are shrunk by the same factor.

Shrinkage estimation is done on this house price data to simplify the high coefficient value. Shrinkage is done by centralizing the value into sample mean, this will not be advantageous for the atypical mean data-like data with extreme outlier. Models which include prior distribution will improved in accuracy by doing shrinkage estimation. Shrinkage estimation is using Ridge Regression and Lasso Regression.

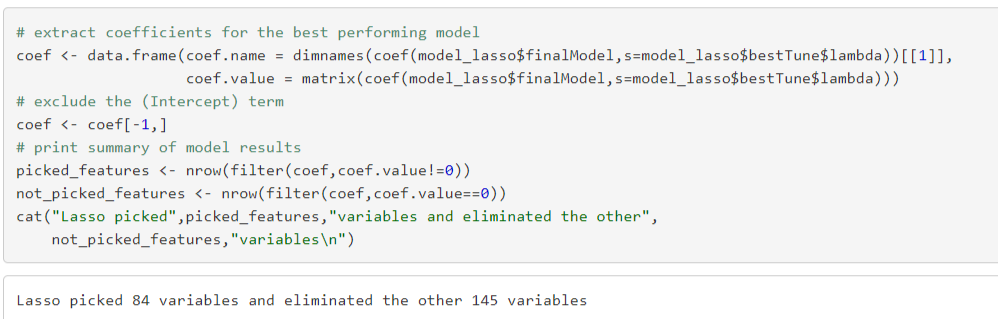
Ridge Regression is to create a simplified model of prediction, which data set has variables have multicollinearity (correlation between predictors). As explained L2 regularizer using shrinking method by using ridge estimator with λ as the tuning parameter. Ridge Regression is putting lambda to convert OLS equation into Ridges equation. X is centered, scaled matrix, x’x is almost singular. When X column is highly correlated, I is identity matrix, Y is dependent variable matrix. While λ is tuning parameter, which needs to be found.

|  |  |
| --- | --- |
| **OLS** | **Ridge regression** |
|  |  |

When λ = 0, dataset methodology will be yielding the same result as OLS (Ordinary Least Square), while λ = ∞, it will shrink all the coefficients to zero. By taking sequences from 0 to 1 with interval 0.001, Ridge Regression is done accordingly. However, an optimum value of λ so that it produces accurate model (measured by RMSE), will be discussed more in next part.



Lasso Regression is the least absolute shrinkage and selection operator. As it named, Lasso Regression simplifies dataset before data is used for prediction by the principle of regularization. However, since Lasso Regression is eliminating out variables, it needs to be preceded with Ridge Regression to do the shrinking process. Lasso then continues to pick variables with lower coefficients as predictors. Below shows that 84 variables picked and 145 variables eliminated due to high coefficients.



# Random Forest

Random Forest (RF) is an ensemble of decision trees. Each Decision Tree (DT) returns its own prediction and the final of it is based on majority voting. The term *Random* of its name comes from two sources. Each DT built in a RF model is based on a subset of randomly chosen predictors and each tree is grown using a bootstrap sample of training data. A DT is mostly used for classification task but in this work it was employed for regression purpose.

A DT breaks down a dataset into smaller subset while incrementally develop an associated DT. It comprises of decision nodes and leaf nodes. A decision node could have a single or multiple branches, each of which represents an attribute value. On the other hand, a leaf node indicates the decision on the numerical target. The best predictor, also known as the root node, is usually at the top of the DT. The primary algorithm used to build a DT is called ID3 which is based on greedy method. It also uses Standard Deviation Reduction rather than Information Gain that is commonly used for DT build for classification. As from its name, each decision node is formulated based on the highest reduction in standard deviation for that instance.

Some of the advantages of using RF includes its computation speed, easily parallelized and ease in handling high dimensional data. Below table illustrates some of the parameters set for the RF method used in this work:

|  |  |
| --- | --- |
| **Parameters** | **Values** |
| Number of trees to grow | 1000 |
| Algorithm | Standard Deviation Reduction |
| No. of variables tried at each split | 79 |

Table 3. A summary of the parameters used in the RF model.

# XGBoost

XGBoost stands for eXtreme Gradient Boosting and it is one of the fastest implementations of the Gradient Boosted Decision Trees algorithm. Gradient Boosting is a type of ensemble technique in which the predictors are made sequentially by learning the mistakes of previous predictors. As such, the observations with the highest error will have greatest probability of appearing in subsequent model and in turn this algorithm will take less time or iterations to reach the predictions with the lowest error rate. In Gradient Boosting, the minimum error is obtained through the use of gradient descent, which is an optimisation algorithm for finding a function’s minimum, and by updating predictions based on a learning rate. However, one of the most severe inefficiencies of gradient boosted trees is that it is required to consider potential loss for all the possible splits before creating a new branch. This is especially time-consuming if there are huge number of features present in the data sets.

XGBoost has several advantages over gradient boosted trees, one of the few notable ones are: 1) XGBoost provides more regularisation options, therefore reducing overfitting. 2) XGBoost is faster as it uses parallel processing. 3) XGBoost has an in-built cross-validation to allow users to run a cross-validation at each iteration and this allows it to obtain exact number of boosting iterations in one run.

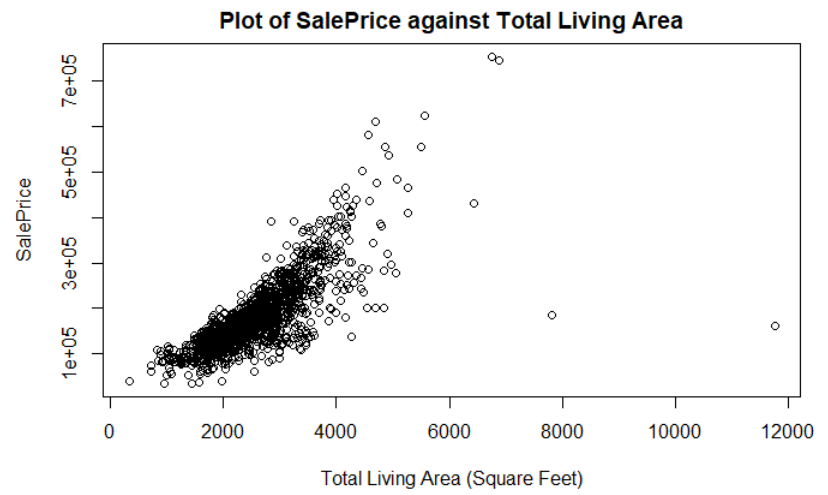
For the maximised model performance, the best parameters have to be chosen. In XGBoost, the parameters can be divided into three categories, 1) general parameters - that controls the type of booster in the model that drives the overall functioning, 2) booster parameters - that controls the performance of the booster selected and 3) learning task parameters - that sets and evaluates the learning process of the booster using the data provided. In this instance, caret package was used to find the best parameters values for “max\_depth”, “min\_child\_weight” and “eta” which are the learning rate, using a 5-fold cross validation.

After the best parameters were found, the best number of rounds was determined using the cross validation function “xgb.cv()” in the XGBoost package.

## Results and Discussions

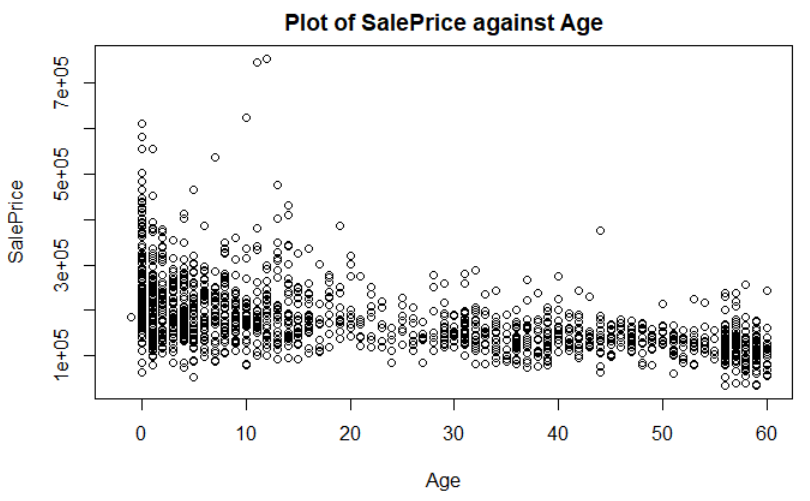
# **Analysis, Discussions and Interesting Findings**

In feature engineering, 6 new variables were introduced. It was found out that the “Total\_Living\_Area” and “SalePrice” had the highest correlation of 0.779 among all other comparisons. (See Figure 3) This is a logical phenomenon given that most buyers would be willing to pay more for houses that have a large total living area. A correlation plot was plotted to see the relationship between “Total\_Living\_Area” and “SalePrice” as shown below:



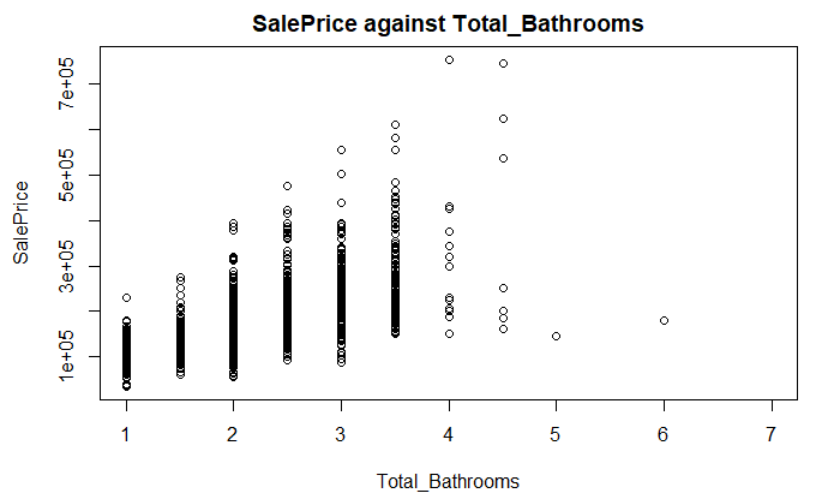
*Figure 3. Illustrates the plot of “SalePrice” against “Total\_Living\_Area”.*

Furthermore, a negative correlation of -0.509 was also found between the the feature “Age” and “SalePrice” (See Figure 4). This is expected as older houses tend to be cheaper compared to newer houses.



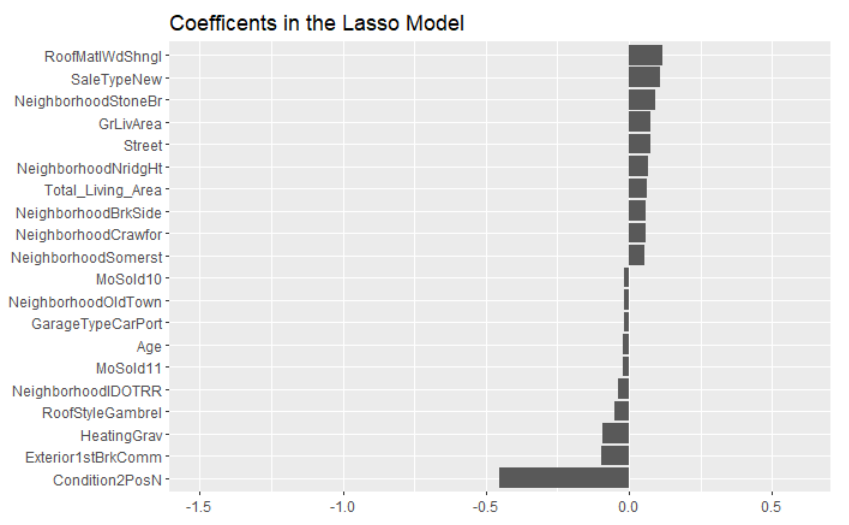
*Figure 4. Illustrates the plot of “SalePrice” against “Age”.*

Lastly, a correlation of 0.632 was found between the “SalePrice” and “Total\_Bathrooms” (See Figure 5).



*Figure 5. Illustrates the plot of “SalePrice” against “Total\_Bathrooms”.*

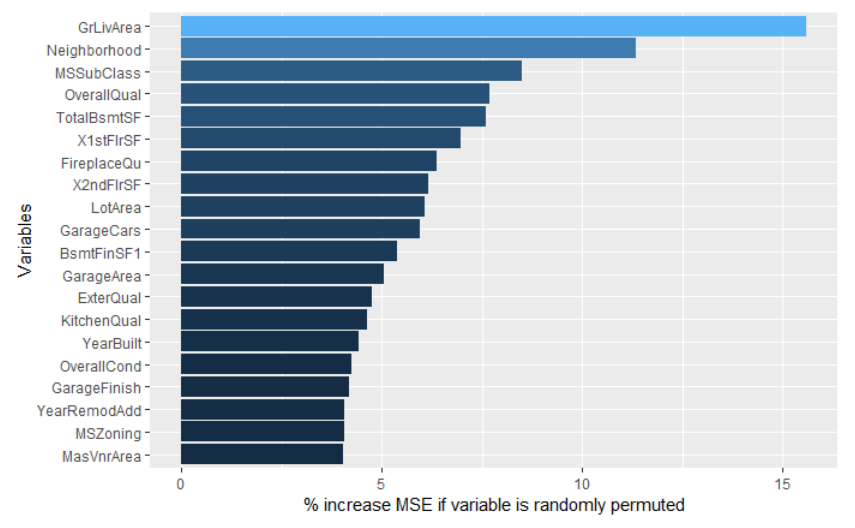
From Lasso Regression, the top 10 and bottom 10 variables that were correlated with the “SalePrice” were identified as shown in the plot below:



*Figure 6. Illustrates the top 10 and bottom 10 variables that were correlated with the “SalePrice” from Lasso Regression.*

The top most variable is “RoofMatlWdShngl” i.e. roof material - Wood Shingles. The cost of a new roof can range from relatively affordable to unbelievably expensive. The main reason that Wood Shingles roofs are in fact related to higher home prices is because Wood Shingles material has the highest material price as compared to most other materials. The next top variable “SaleTypeNew”, being home just constructed and sold. makes best logical relationship with SalePrice. In the top variable list, also include a neighborhood within Ames, called “NeighborhoodStoneBr” (Stone Brook). According to Ames City Budget Report, Stone Brook has been planned to close to facilities like Ames Golf Club and other Heritage Parks. On the other hand, the variable with the lowest correlation is proximity to positive off-site feature--park, greenbelt, etc. (“Condition2PosN”) which did not attract buyers.

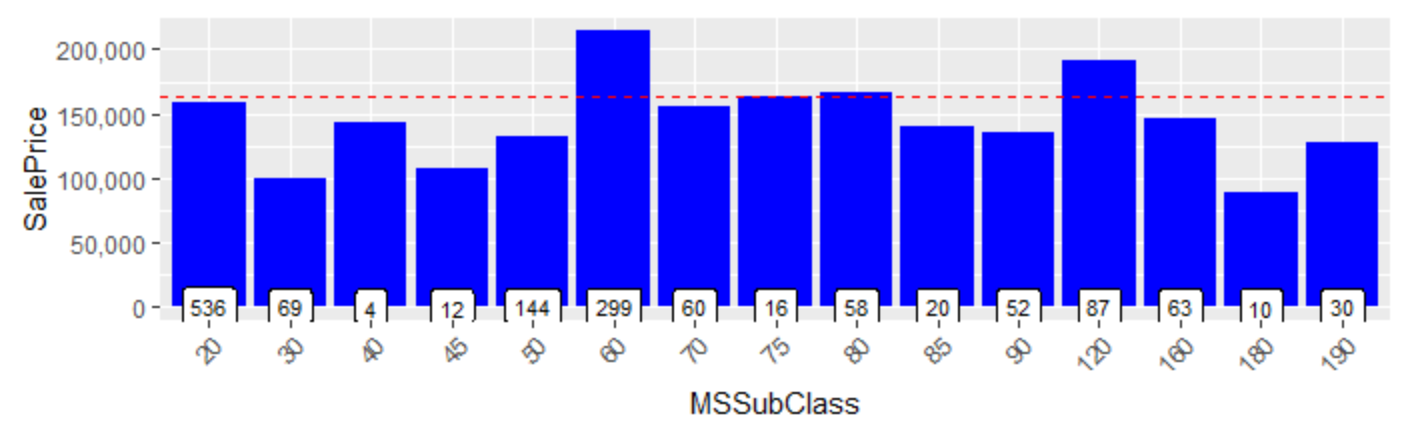
Next, from Random Forest, it was found that apart from size of ground level living area and the neighbourhood, the type of dwelling (“MSSubClass”) and several garage variables were also identified as highly important factors. (See Figure 7)



*Figure 7. Illustrates the top 20 variables that were correlated with the “SalePrice” from Random Forest method.*

This is logical given that buyers will be willing to pay more for the type of dwelling they desire. From the plot below, it can be seen that houses that “MSSubClass 60” and “MSSubClass 120” which are “2 storey houses - 1946 & newer” and “1-storey planned unit development - 1946 & newer” respectively, they can fetch higher prices as compared to the median amount (as shown in the red line).

However, the highest number of the type of house sold is “MSSubClass 20” (“1 storey house - 1946 & newer”) which was significantly higher than the rest of the type of dwellings. (See Figure 8) This could probably indicate that majority of the houses in Ames, Iowa are of this type of dwelling.

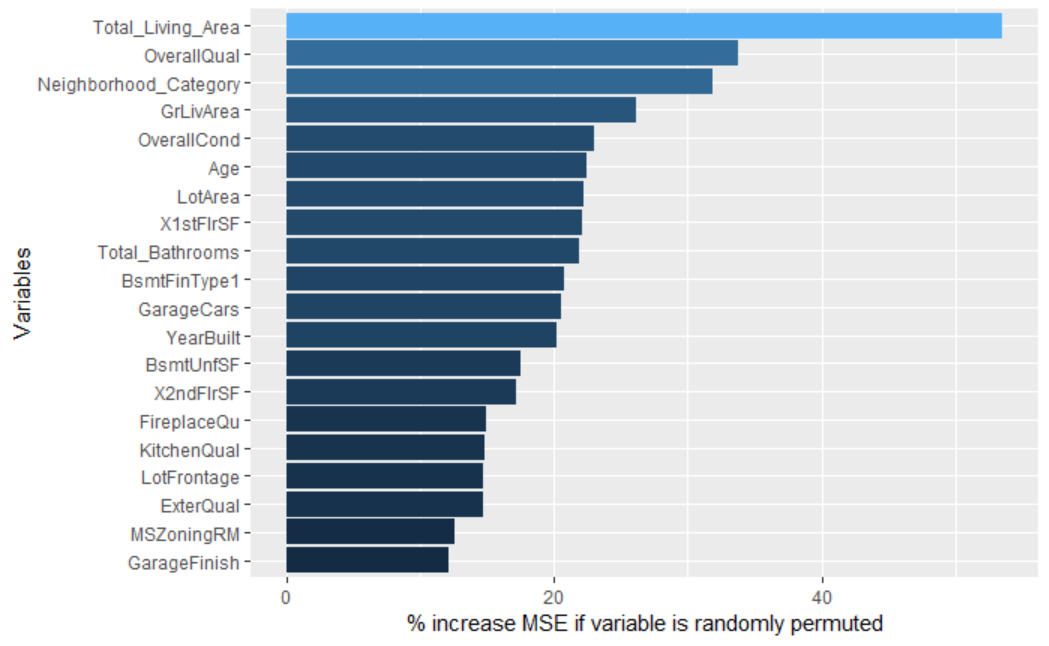


*Figure 8. Illustrates the chart of SalePrice against MSSubClass.*

From Random Forest, it could be seen that garage variables (“GarageCars”, “GarageArea” and “GarageType”) were highly correlated with “SalePrice”. This is also understandable as residents in Ames, Iowa were likely to commute using cars. Ames, Iowa has a land size of 62.86 km² and a population of 66,191 which indicate a low population density, as compared to Singapore which has a land size of 719.9 km² and a population of 5.607 million (Singapore is approximately 7 times denser than Ames, Iowa). Hence, it could be inferred that amenities or facilities are more spread out in Ames, Iowa and it would be more convenient for the residents to own car(s).

In addition, from Random Forest, “OverallQual” (overall quality), “ExterQual” (quality of the exterior material), “KitchenQual” (kitchen quality), “BsmtQual” (height of the basement) and “FireplaceQu” (fireplace quality) all have high correlations with “SalePrice”. This evidence is an indication of the house quality importance to the buyers and is definitely true as buyers will be willing to pay more for a house with better quality.

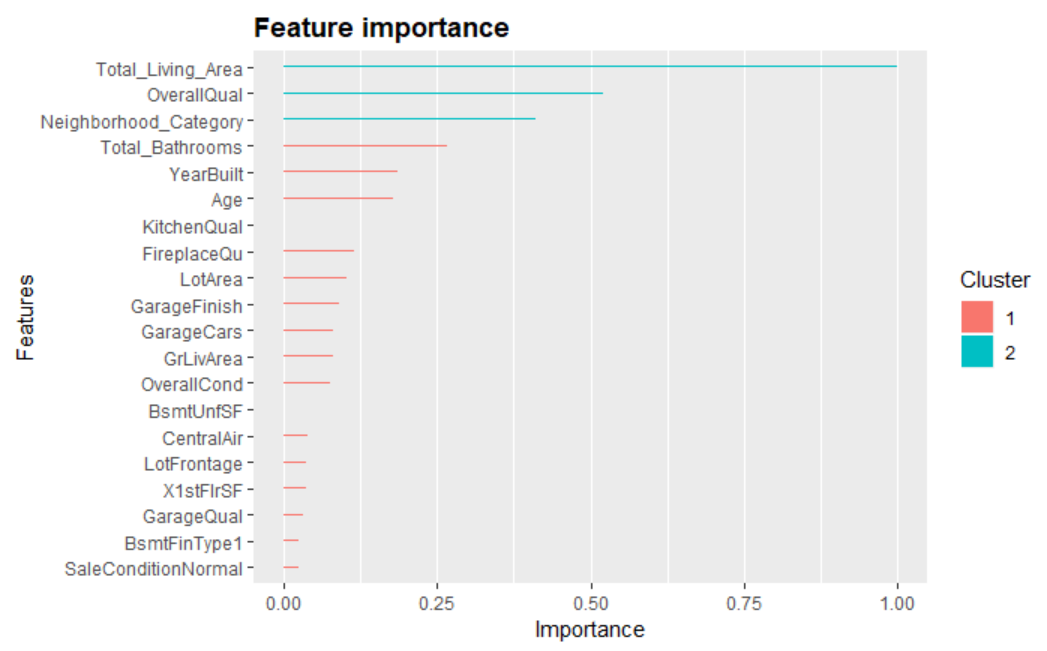
When Random Forest is re-run after engineered features were added in, the results turned out to be interestingly different from above. The ranking of the variable are as follow:



*Figure 9. Illustrates the top 20 variables that were correlated with the “SalePrice” from Random Forest (with engineered features added).*

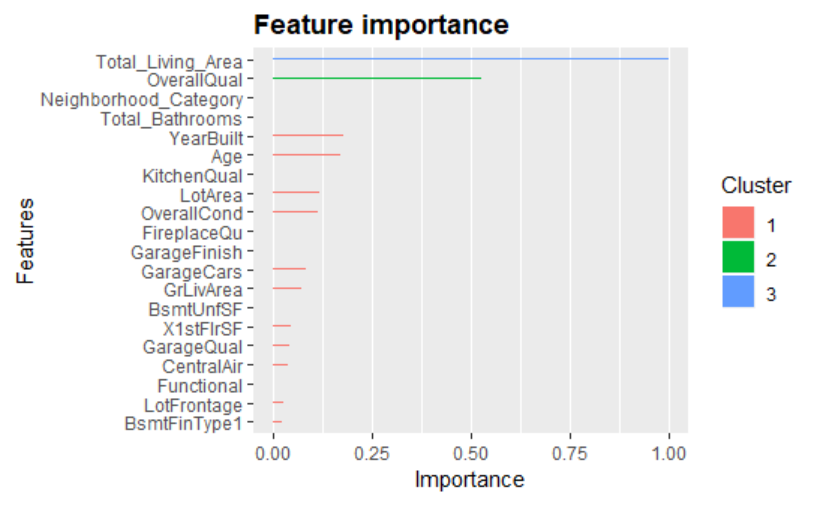
The top 20 most important variables have changed from prior Random Forest modelling. Among the top 20 variables, all 3 new variables created during our earlier feature engineering had all top the list. These variables are “Total\_Living\_Area” (Total area above ground and below ground = basement area), “Age” (difference between the year it was sold and the year it was remodeled) and “Total\_Bathrooms” (total number bathrooms). Naturally, buyers are concerned if the homes had been remodeled recently and are more willing to pay more for more bathrooms, thus the importance to “SalePrice”. Earlier on, as explained, most buyers would pay more for larger total living area. This result helps to re-validate our feature engineering hypothesis on new variables creation.

Further to the exploration of above simpler algorithms, we proceed to explore more complex one other than Random Forest, XGBoost was applied to the given “training dataset” (a subset of train dataset at 80% and test dataset at 20% is created from this given “training dataset”). Using the optimised parameters of max depth = 4, eta = 0.01 and min\_child\_weight = 4 after running the caret (with nrounds set to 1000) to find the best hyperparameters using 5-fold cv. Subsequently, when nrounds is set to 1000, from the cross validation, the best number of rounds is 1313. The cross validation RMSE is 0.124112. The plot also showed the variable importance:



*Figure 10. Illustrates the variable importance from XGBoost with nrounds set to 10.*

When nrounds is set to 10, the results are very different from above (nrounds set to 1000). As best tune parameters are max depth = 2, eta = 0.1 and min\_child\_weight = 1, the best number of rounds stop early at 10. Hence, the cross validation RMSE is 4.037670. The plot also showed the variable importance:



*Figure 11. Illustrates the variable importance from XGBoost with nrounds set to 1000.*

From above 2 outcomes, we can conclude that using the best parameters from grid search, the model with nrounds set to 1000 (cv RMSE = 0.124112) returned the lowest errors for both datasets as compared to model with nrounds set to 10 (cv RMSE = 4.037670). The important variables had also changed as illustrated from the 2 plots above. Other than 2 of the top variables are no longer significant in the nrounds = 10 run, the list of top 20 variables had also changed.

In summary, after applying simple algorithms from Regularized Regression to more complex algorithms (automated parameter tuning), we made additional findings:

* The variable which consistently tops the rank of importance across all models is **“Total\_Living\_Area”** (Total area above ground and below ground = basement area); this outcome is very much supported universally as total living area to most buyers is an important feature.
* All **3 new variables** **(i.e. “Total\_Living\_Area”, “Total Bathrooms”, “Age”)** yielded high importance across all models.
* **Ames-specific variables** are also unique to our analysis, such as Neighborhood related variables. This helps us to learn the importance of local knowledge for data analysis.

## Comparison

This section summarizes the 4 machine learning algorithms’ performances which were used to predict the “SalePrice” of each home. The RMSE is a good measure of how accurately the model predicts the “SalePrice” and it is the most important criterion for fit, hence is the key basis for our evaluation. The R-squared is also listed to show how well the actual and predicted values are correlated.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Regularised  Regression (Ridge / Lasso)** | **Random Forest** | **XGBoost** |
| **RMSE**  (for Training dataset) | 0.128496 / 0.1244038 | 0.1383465 | 0.124112 |
| **R-squared** | 0.932 | 0.936 | 0.943 |

*Table 4. Illustrates the RMSE and R-squared from the different methods.*

* Of the 4 machine learning algorithms, XGBoost model provided the lowest RMSE score. This is achieved for the model, with nrounds set to 1000 and using a subset of the Training dataset, with a cross validation RMSE of 0.124112.
* Though for Lasso Regression, using caret cross validation, the model derived a RMSE of 0.1244038, it does not select a substantial number of the given variables in its model, as it is supposed to do.
* As the XGBoost cross validation RMSE is better than Lasso’s CV RMSE, we recommend as part of the future improvement that averaging the predictions is likely to improve the final predictions and we can weigh the XGBoost results more.
* For all the models, the R-squared values are very comparable as each model accounted for over 93% of the variation in prediction of “SalePrice”. Again, XGBoost model presented the best R-squared value of 0.943.

## Conclusion

In the course to better predict the “SalePrice” through advanced regression techniques, different machine learning algorithms are applied to the dataset to help the learning process.

First of all, the use of traditional regression methods, i.e. Regularized Regressions, help to understanding the correlations between the existing variables and the “SalePrice”. Starting with the basic relationships, new variables were created by adding interactive effects among using existing variables. When these new variables were imputed into the various models, more insights are gained. This makes us further wonder the depth of data around us.

As the problem statement for our Group Project is to predict the “SalePrice” for residential homes in Ames, Iowa, USA, it is compelling to use statistics such as RMSE to determine the accuracy of the predictions. RMSE is the square root of the variance of the residuals. It indicates the absolute fit of the model to the data–how close the observed data points are to the model’s predicted values. From the 4 machine learning algorithms used in our Group Project, XGBoost provides the best RMSE as compared to the 3 others (Ridge, Lasso, Random Forest). XGBoost only provides the best RMSE when its best iteration is achieved. However, to run XGBoost effectively and efficiently, it is key to good CPU Processing Power to ensure the outputs are generated.

Furthermore, to better predict the “SalePrice”, many steps are critical to the success of good predictions, such as during **Data Preprocessing** step from converting character variables to numeric variables in order to perform regression, treating missing values (either by replacing these values with mode, median, mean or “None” or “0”), normalizing the data. The rigorous application of the **Data Mining** step is a good start to extract the different insights of the dataset. With the thoroughness in **Post Processing** step, improvements can be achieved.

Lastly, for future explorations, our Group recommends the following:

* Try out more machine learning algorithms e.g. Neural Network and SVM.
* Submit on Kaggle for more learning opportunities, especially Leaderboard.
* Use these techniques to explore Singapore Housing Data to predict which factors will influence the home prices the most.
* Similarly, explore other property markets as well.

# 

## Appendix

The following table illustrates all the accompanying Files and their respective purposes:

The R Codes are splitted into several files to speed up our processing. Hence, it is recommended to run these **R Codes (in Notebook format)** in the following sequence. For ease of reference, **R Markdown docs (in HTML)** are also enclosed in the submission.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **S/N** | **Category** | **Filename** | **Purpose** | **Input File** | **Output File** |
| 1 | R Codes | data\_preparation.Rmd | Data Preparation |  | all\_DP.rds |
|  | R Codes | analytics.Rmd | Data Analysis | all\_DP.rds |  |
| 2 | R Codes | feature\_engineering.Rmd | Feature Engineering | all\_DP.rds | all\_DP\_FE.rds |
| 3 | R Codes | Pre\_Processing.Rmd | Data Pre Processing | all\_DP\_FE.rds | train.rds  test.rds |
| 4 | R Codes | regularized regression.Rmd | Regularized Regression  (cover both Ridge and Lasso) | train.rds  test.rds | ridge-sol.csv |
| 5 | R Codes | random\_forest.Rmd | Random Forest | train.rds  test.rds |  |
| 6 | R Codes | xgboost.nrounds1000.Rmd | XGBoost where nrounds = 1000  (long CPU Processing) | train.rds  test.rds |  |
| 7 | R Codes | xgboost.nrounds10.10.Rmd | XGBoost where nrounds = 10 | train.rds  test.rds |  |
| 8 | Additional R packages: | library(knitr) library(ggplot2) library(plyr) library(dplyr) library(corrplot) library(caret) library(gridExtra) library(scales) library(Rmisc) library(ggrepel) library(rand omForest) library(psych) library(xgboost)  library(moments)  library(glmnet)  library(elasticnet)  library(Metrics) | To run above R Codes,  load this list of Additional R packages (besides base R) |  |  |
| 8 | R Markdown docs (in HTML) | \*.html | To preview our R outputs (for each of the R Codes) |  |  |
| 9 | Data | train.csv | Training Dataset |  |  |
| 10 | Data | test.csv | Testing Dataset |  |  |
| 11 | Data | data\_description.txt | Data - description file |  |  |

## 

## References

<https://medium.com/mlreview/gradient-boosting-from-scratch-1e317ae4587d>

<https://medium.com/@gabrieltseng/gradient-boosting-and-xgboost-c306c1bcfaf5>

[ttps://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/](https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/)

<https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/>

<http://download.smkn4padalarang.sch.id/Ebook/sysadm/Machine%20Learning%20with%20R,%202nd%20Edition.pdf>   
<https://archive.ics.uci.edu/ml/datasets.html>   
<https://cran.r-project.org/>   
<https://github.com/slopp/nyr-tips-tricks>   
<https://bookdown.org/yihui/rmarkdown/>   
<https://rpubs.com/>   
<https://www.theanalysisfactor.com/linear-models-r-improving-our-regression-model/>   
<https://uc-r.github.io/linear_regression>   
<https://web.ma.utexas.edu/users/mks/statmistakes/skeweddistributions.html>   
<https://data.library.virginia.edu/>

<https://web.stanford.edu/~hastie/ElemStatLearn/printings/ESLII_print12.pdf>

<http://www.statisticshowto.com/regularization/>

<http://www.statisticshowto.com/ridge-regression/>

<http://appliedpredictivemodeling.com/toc/>

<https://www.hackerearth.com/practice/machine-learning/machine-learning-algorithms/beginners-tutorial-on-xgboost-parameter-tuning-r/tutorial/>

<https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/>

<http://www.statisticshowto.com/lasso-regression/>

<https://www.roofcostestimator.com/concrete-vs-clay-roof-tile-cost/>

<https://www.stat.berkeley.edu/~breiman/randomforest2001.pdf>

<https://en.wikipedia.org/wiki/Ames,_Iowa>   
<https://www.cityofames.org>/   
<https://www.cityofames.org/Home/ShowDocument?id=12571> (Ames City Budget report)