House Pricing Prediction

Project Report

Presented by

Vignesh_Sakthivel

Abstract

This paper is the report of a Kaggle competition, House Prices: Advance Regression Techniques. Buying a house has long been a staple of the American dream. This project is about building a model to predict the house prices of different houses in Ames, Iowa based on related features through advanced regression techniques. The data set being used for the purpose is Ames Housing Dataset (I), compiled by Dean De Dock..

1 Introduction

In this project, we used the Ames Housing dataset to predict the final sales price of each home in the test data set, with a large number of features. The Ames Housing Dataset (I) was used to build a model to predict the house prices of different houses by combining the train and test data which held 1460 observations and 79 explanatory variables describing the various related features of the houses. The goal is to predict the Sale Price of the House with given dataset. For achieving the set target, our approach involved exploring the data, cleaning and imputation of the NA values, merging the test and train data sets, finding correlation, achieving normalization, and applying different models that would benefit better result

The models and data preprocessing is based on R language and we also building our models for better data visualizations. We first got to know our data set and all the variables through both univariate and multivariate approaches, and then we applied data cleaning techniques to reduced noisy data and checked for basic assumptions of regression. Moreover, to avoid overfitting problem, we selected variables that are truly correlated with the dependent variable and dealing with outlier bias. Afterwards, we conducted a comparative study on different advanced regression technique, in which we compare the performance of linear regression, random forest, LASSO regression, Xtreme Gradient Boost and Neural Net. Eventually, we use the model to predict the final sales prices of test data set and the procedures will be discussed in greater details in the later sections.

2 Pre-Project Works

Before implementing regression models, We successfully completed online course “Feature Engineering” recommended by our professor. We did some researches on dataset their data limitations, restrictions and complexity. We found that we need to focus on, which are linearity, normality, and no multicollinearity in the regression model. Also, we believe Neural Network will be a perfect fit for performing house price prediction. Since the task of predicting house price does not require a step-by-step break down of how the model gets the final predicted value. Moreover, Neural Network is able to hierarchically learn the importance and relations among all the input features. We believe that it can better capture the subtle relationship between features and the target sales price.

3 Methodology

The first part of the project is mainly focused on the Data Preprocessing and feature selecting, which is accomplished through correlation matrix, scatter plot, histogram, and statistical summary are frequently used.

In the second part of our project, we build and compared types of regressions: linear regression, Xtreme Gradient Boost, Lasso regression and random forest.

Linear regression is the base model that commonly used for predictive analysis and we want to get the first glance of our data by feed in this simple prediction model. The second mode that we used is Random Forest and Clustering which is a slightly advanced regression model that takes each driven factor into account when doing the calculation and enhance the result. By implementing the cluster and tree model we found out that the performance of the model is not as good as our expected, then we tried the third model Lasso regression. Lasso regression is a shrinkage and selection method for linear regression, and it minimizes the usual sum of squared error because of its ability to perform subset selection and regulation. And for the last regression model, we used the Xtreme Gradient Boost. Random forest possesses significant effect on this model. Random forest is a supervised learning algorithm and could be used for both classification and regression problems. Random forest is robust to errors and
outliers and it will not lead to overfitting since error for a forest converges if the number of trees in the forest is large

For the third part of our project, we constructed a layer-feedforward network and preformed hyperparameter tunning to reduce the validation error. Such a multilayer feedforward neural network turns out to be quite robust in predicting house price. It is the best model among all the model we created and has secured a place on the public leaderboard.

Finally, we compare the performance between all the model we have created and use the best performance for predicting the final house price.

4 Experiment

4.1 Data Preprocessing

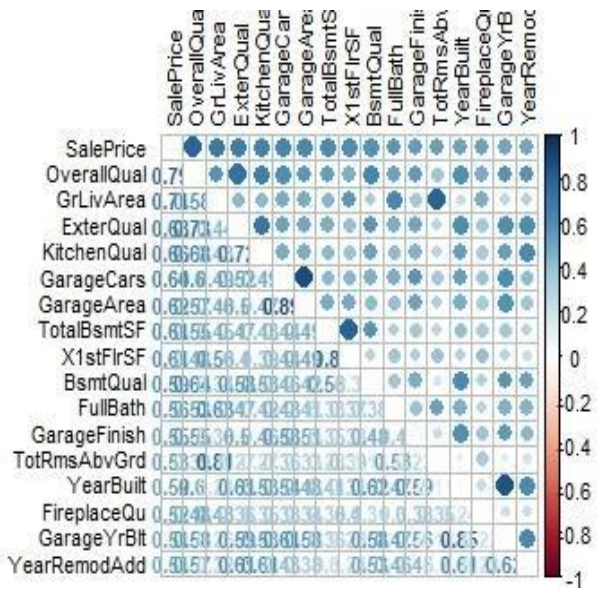
Before we start to apply our regression model to the data set, it is essential for us to have a clean and tide training data, the better training data we can get, the better performance of our regression model could behave.

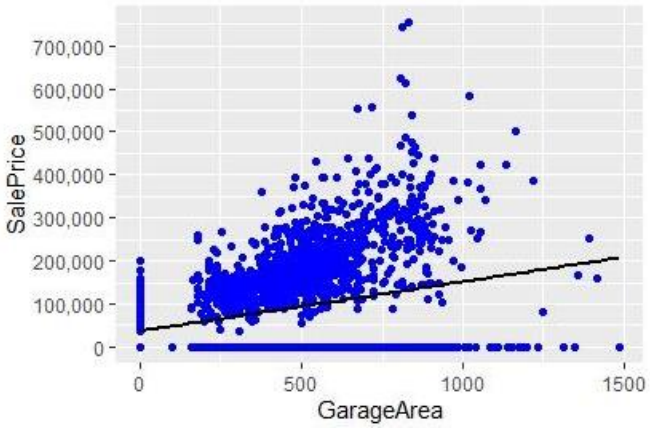
Our data preprocessing contains major steps:

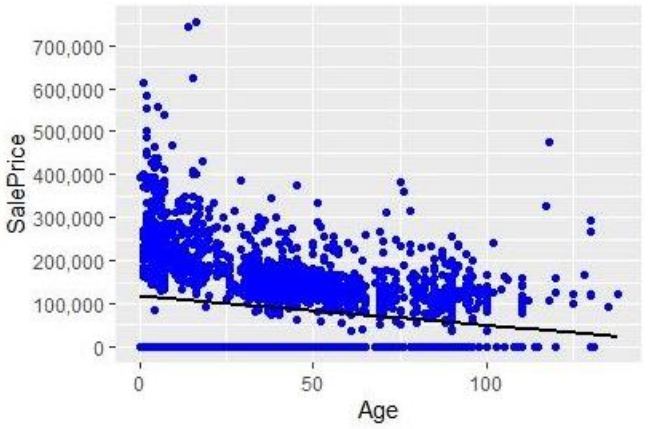
- Detecting and dealing with missing values.
- Handling outliers.
- Normalizing distribution of variables.
- Checking for regression assumptions.

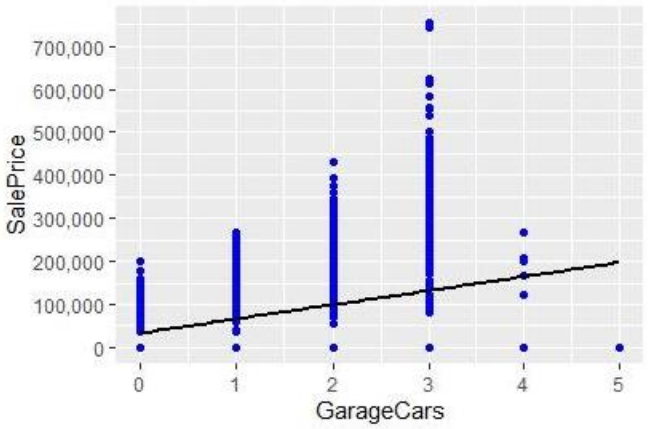
First, we detected all the missing values in each column and sorted them with their percentage. We found out that there are more missing values in Categorical variables than numeric variables. For dealing with the variables, it is important for us to find a threshold to decide whether to drop the variable or replace them using other imputing methods. By observing the data, we found out that there exists a huge gap in the percentage of missing data volume. The number 200 can classify the missing variable into to clusters, higher missing value cluster and lower missing value cluster. After doing so, we drop the variables which have more than 200 missing data and then we tried running linear regression to predict the missing value in our low missing value clusters. However, the regression of each variable doesn't have a good performance and each of their R squares are no more than 50%, so we decide to use the mean value to fill out the gap.

After deleting and replacing the missing variables, the second step we took is to handle the outliers. The first big issue that we met is there are too many variables and very difficult to handle. So in order to make our process more efficient and not losing important information, we constructed a correlation map for all the existing numeric variables.





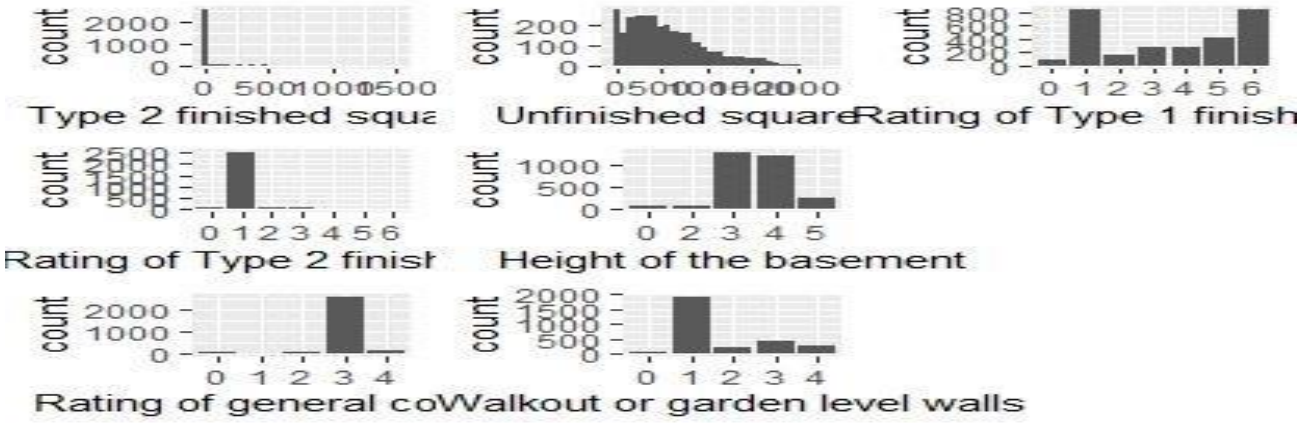




The correlation map above shows the correlation between each selected numeric variables. In order to reduce the variables that we used in our regression, we only choose the top ten variables which has a correlation above 50%.(The correlation between each independent variables and dependent variables are at the bottom line of the correlation heat map). Then it is much easy for use to plot the scatter plot of each variables to detect the outliers and remove them. I have added few example plot relating with the Sale price to show variation.

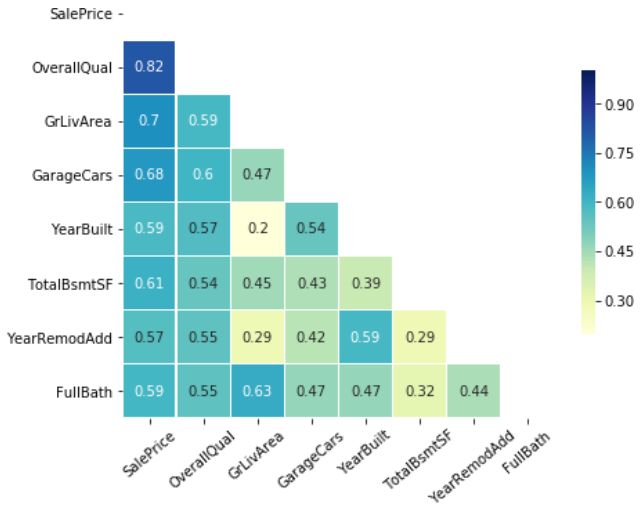
Comparing the sample Garage multiple variables for better understanding of their type and importance

After normalizing the data, we need to check for the other very important criteria for the regression.



The next step is to check for the regression assumption. Since we are using the dataset to build a regression model, we want to ensure that the basic assumptions for regression are satisfied. First, the regression requires all variables in the model to be multivariate normal. Therefore, we created histograms for every selected variable so that we could analyze the distributions and we started by plotting the distribution of sales price. We could see the distribution of sale price clearly skewed to the right and one way to deal with skewed data is to perform natural logarithm transformation to the data set. The histogram is plotted again after the sales price is log-transformed and its distribution is almost the shape of the normal distribution. After analyzing the distributions for the selected independent variables, we found that there is no not much transformation that can help to squeeze the data.

This correlation heat map can help us noticed that there do exist multicollinearity between some of our chosen variables. For few instance like Garage interconnected variables have a correlation nearly 90% which means there exists a strong collinearity between these two variables. After detecting & imputing the data the changes can be noticed from the heat map. The correlation heat map after changing shows below:



From the heat map, we can see that the multicollinearity of our data significantly reduced.

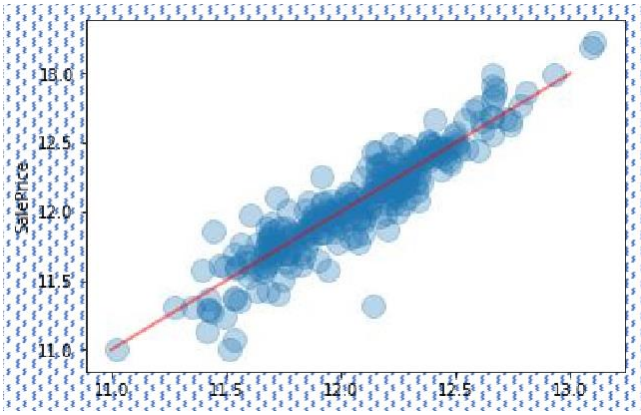
After doing all these process, the final thing that we have done before building our model is change all the categorical variables into dummy variables, and now we have a clean and tidy training dataset.

4.2 Regression Model

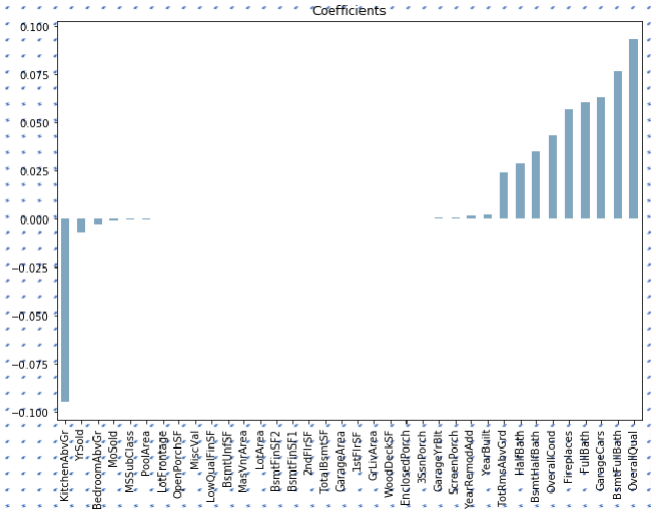
In this sector, we will discuss more details of our regression models. In order to have a general idea of our feature selection quality, we first choose to use linear regression and see its performance. In order to align with the Kaggle rating criteria, we also use the root mean square log error also write as RMSLE. This error measurement is calculated base on MSE, and the formula is

$$\sqrt{(\sum(\boldsymbol{log}(\mathbf{1}+\square\square\square\square\square\square\square)-\square\square\square(\square+\square\square\square\square)/\square)}$$

This measurement punishes underestimate more than overestimate and makes sense in the house price prediction situation that when buying a house the worse case is more valuable than the lucky situation. Back to our linear regression model, our RMSLE and the scatter plot between the predicted value and test value are shown below:

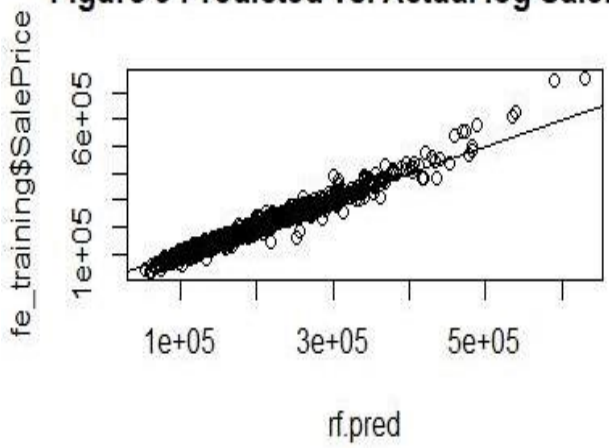


In this case, we think the model didn't handle the cheaper house price as well as the expensive price, and false values appeared more frequently in the lower price range. In order to have a deeper understanding of the price-driven feature we plot out the coefficient of each variable in our linear regression:



Due to the coefficients plot, the second model that we tried is the Cluster and Random forest, which shrinks the coefficient of each independent variables and automatically handles the model along with the cluster. We applied clustering to the model made two cluster and ran random forest on the model, comparing the final model values the models will be selected.

Figure 9 Predicted vs. Actual log SalePric

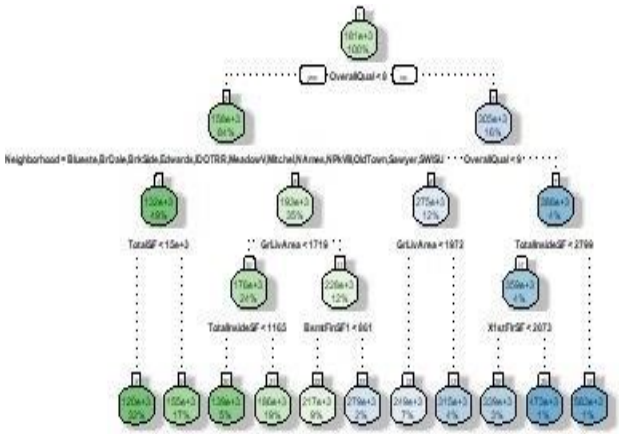


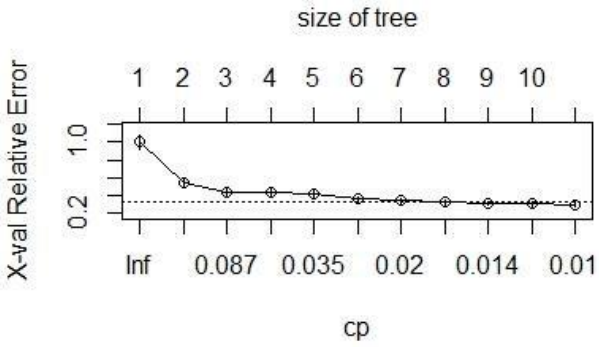
The graph shows the predicted and actual log Sale price.

From the coefficients graph, we have the rightful concern that the variables who has a positive coefficient are probably still existed some level of collinearity and recall the sub heat map we can

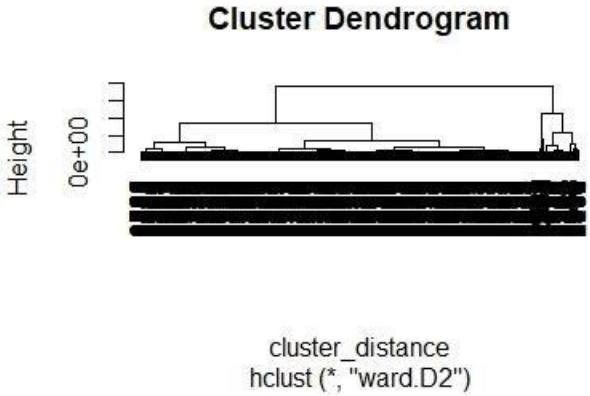
easily found that there are still some variables who has a correlation over 50% and not being excluded

The random forest is also known as random decision tree forest, which operates by constructing a multitude of decision tree at training time and mean prediction of the individual tree. Random forest also corrects for decision trees' habit of overfitting to their training set(). Because it's specialty to avoid overfitting, we decide using the training set which we generate only after handling the missing data but before feature select. After tunning the hyperparameters, we found that its performance is not as good as the previous models. However, when using the data after feature selecting is worse, which R square is less. The reason that we speculate is that the training data dimensionality is way more than the testing data due to the many dummy variables that we added. In this case, it is also explained that why even linear regression performed well compared to the random forest.

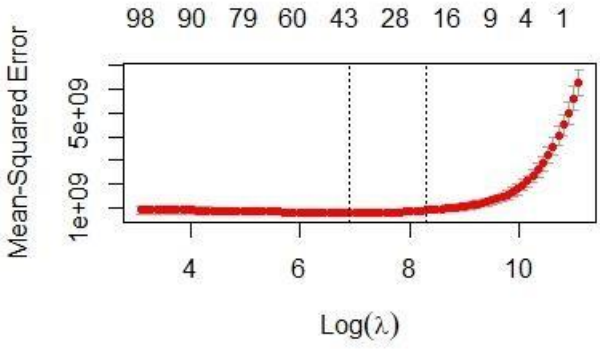




From the cluster model we have made the dendrogram

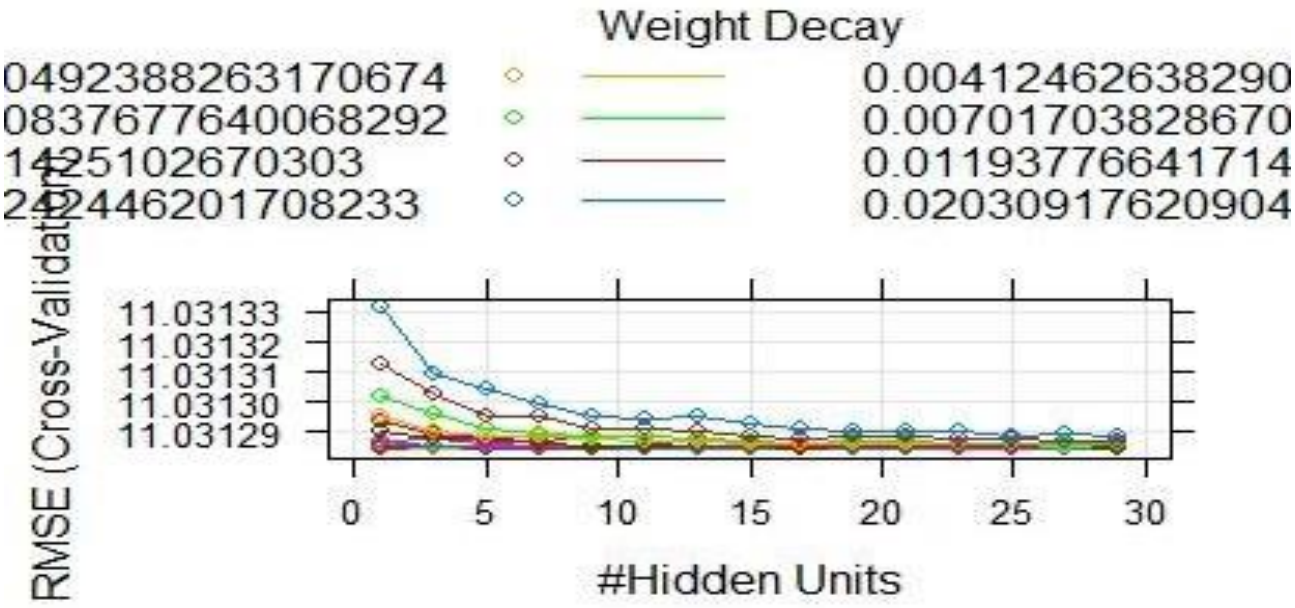


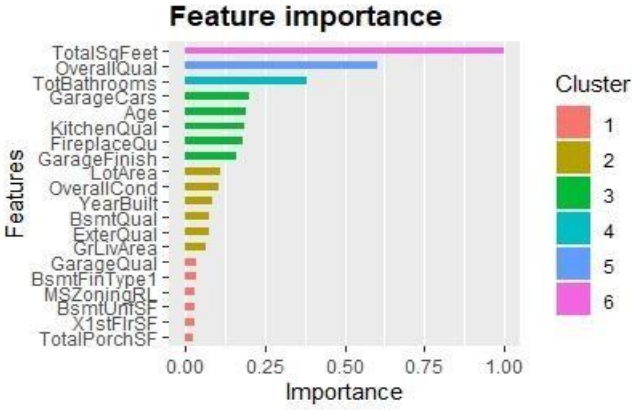
Lasso Regression model gives us better results compared to other since it selects and runs on the important variables with the fixed lambda better for the model. The lambda min graph of the model is plotted.



Xtreme Gradient Boosting is also one of the effective model but didn't perform better than lasso regression model in this case. The feature

importance of the model is plotted.



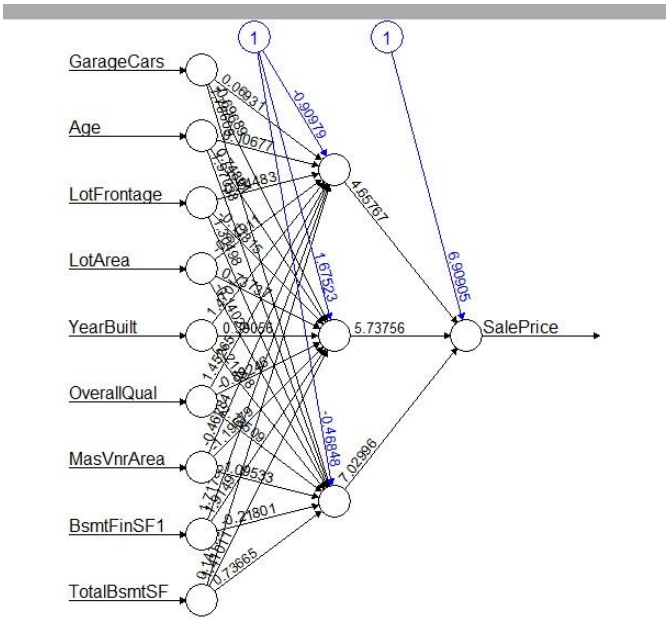


4.3 Neural Network

The neural network that we use to predict house price is a 3-layers feedforward neural network. Since we believe all the numerical features, except the "Id", convey some information about the house price. We select all the numerical features, except from "Id", from the training sets and perform Minmax normalization from each feature. After doing so, we obtained a dataset with 33 normalized features and 1 normalize target value, the final sales price. Thus, the models have 33 input neural units and 1 output neural unit.

After performing hyper-parameter tuning, we discovered that our best performance model is one with neural units in the first hidden unit and neural units in the second neural units

Here is the diagram for our Neural Network:



In order increase the final accuracy of our model, we explored three different update criterion to calculate the training and validation error: MSE (Mean Square Root Error), RMSE (Root Mean Square Error) and RMSLE (Root Mean Square Logarithmic Error). The picture below are some example of the training curves for models using three update criterion mentioned above:

Fig 1: MSE

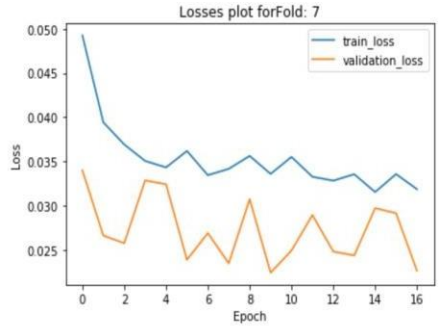


Fig 2: (RMSE)

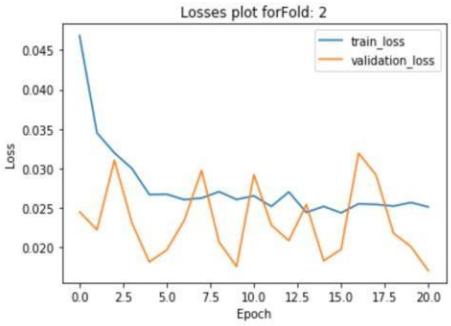


Fig 3: RMLSE

Fig 1 is the lost curves for the model using RMSE to calculate loss; Fig 2 is the lost curves for the model using MSE to calculate loss; Fig 3 is the lost curves for the model using RMSLE to calculate loss. As we can see from the figures, the curves of Fig 1 has a smoother decrease than those of the other two. Besides, the accuracy is higher for the model utilizing MSE to calculate loss. Thus, using MSE to calculate loss allow our model to have more generation power to predict unseen data.

By having a hypothesis that using MSE as a loss function will yield a greater outcome on the testing dataset. We started to perform the Hyperparameter tunning on a 3-layer-feed-forward network with MSELoss and Adam optimizer in order to find the model that has the least validation loss. That means this model will perform best on the testing data. We designed our Hyper-parameter tunning with a greedy search approach. Below is the algorithms for our greedy-based Hyperparameter tunning:

I ← Normalized Data Set □□ ←{0.001, 0.0066, 0.01, 0.1}
D_7 ← {30,28,24,20,12}
D_B ← {26, 24, 16, 9, 7, 3} $Model_{abcd}$ ← □□□ $Error_{abcd}$ ← ∞ for d_7 in D_7 do

for d_B in D_B
do for
lr in LR
do
para ← {□7, d_B, lr}
Model, Error ←
Train(I, para) ifError < $Error_{abcd}$
□*h*□□ $Error_{abcd}$ ← Error
$Model_{abcd}$ ← Model

After finishing the algorithm, we have our final model: it is a 3-layers feedforward neural network with 28 neural units in the first hidden layer and 16 units in the second hidden layer. Such model achieves a high accuracy while predicting the house price of unseen examples. The validation loss calculated by MSELoss is 5.51 and the training loss of the last epoch is 0.0203. Form the RMSLE graph we can identify the loss is nearly 0.01. We speculate that the reason why the training lost is higher than the validation loss is the undetected outlier in the training set.

Most importantly, we have submitted out predicted outcome to Kaggle. The outcome of the predicted by this model has better performance than those of 50% of teams on the public leaderboard.

5 Conclusion

After trying different models we get a simple comparison matrix of models. From the table we listed below, it showed that the Neural Net is the most accurate one. So that we used the neural net to predict the final test set which leads us to achieve better score over Kaggle leaderboard as Top 3% with 0.11309 score.

However, there are still lots of works that need to do to improve the regression model. The first thing that is highly approachable is to do the feature selection process more accurately and changing the numeric categorical variables into dummies as well. For the random forest, reduce the dimensionality of the training data seems urgent and necessary, using PCA and other methods to reduce dimensionality will be the next step for improving the performance of random forest. Besides, although we haven't included the SVM in our models, trying SVM is also a good choice when the data has more dimensionalities. For the Neural Networks, we have captured the subtle relationship between numerical features and the final house price. However, the categorical features also carry some information about the response house price. Therefore, the next step of using deep learning approaches is to explore the encoder-decoder network for the categorical features. We hope to use a separate encode rdecoder network to extract some information about the relationship between categorical features and the house price. Then, we concatenate such information with the output of the last hidden unit of our current model to build a multimodal early fusion network. In doing so, we hope that we could further improve our model's accuracy.

6 Reference

1. House prices: Lasso, XGBoost, and a detailed EDA by Erik Bruin
2. *Selva Prabhakaran*. *Assumptions of Linear Regression*. Retrieved May 6, 2018 from http://r-statistics.co/Assumptions-of-Linear-Regression.html
3. "KDnuggets." *KDnuggets Analytics Big Data Data Mining and Data Science*, www.kdnuggets.com/2017/11/10-statistical-techniques-data-scientists-need-master.html.
4. Techniques you should know! https://www.analyticsvidhya.com/blog/2015/08/comprehensive-guide-regression/
5. **House-Prices-Advanced-Regression-Techniques** by **Pau Roger Puig-Sureda**