House sale price prediction with variable selection based on Lasso

Zheming Lian| Jiahui Jiang| Yuying Yan

Introduction

The goal of this project is to predict house selling price based on several features of house. The dataset we use is from Kaggle.com[1], which contains 1460 observations and 80 variables. In this project, we use LASSO and coefficient path plot to do variable selection task and then fit a model to predict the house price.

Data wrangling

Dealing with missing value:

We combine the training set and testing set together to maintain the consistency. After generally viewing the data, we find that the dataset contains a lot of missing data. So we try to see if there are some patterns of these NAs. According to the data description, most of the NA means "None". For example, the NA in "GarageType" variable means "No garage", so we substitute NA in these kind of variables into "None" for categorical variables and 0 for numerical variables, respectively. In addition, a little number of the missing values does not show any pattern, so we decide to change them into mode for categorical and median numerical variables, respectively. Additionally, there are some datas that the value of certain variable is inconsistent with the value of another obvious correlated variable. For instance, in line 2039, the variable 'BsmtCond' has value NA (no basement),but in the same line other variable related to basement all has value other than NA. Thus we believe that there are typing error in certain cases, and update the certain value from NA to a reasonable variable.

Updating data type:

We update the data type for some variables. For instance, OverallCond is treated as numerical when it was read into R. It will be more appropriate to change it's type to ordinal. In another case, it is inappropriate to do similar change to variable reflects year of built. This is because in this way the number of levels will be too much. Thus for certain variables (there are more multiple variables describing the year of built) we segment the value into 10-year interval and use the interval as the new variable.

Log transformation :

Since skewness will influence the performance of our model, we need to avoid this kind of influence. If the skewness of certain numeric variable is larger than 1, then we take the log transformation of certain variable. We also take a look at our response variable, sale price. The density plot shows distribution of sale price is right skewed. To apply linear model, we need to do log response variable to meet the assumption. After we log the sale price, the distribution of it seems more normal.

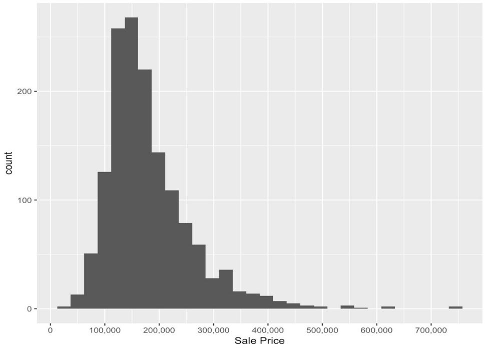
Detecting near-zero variance predictor:

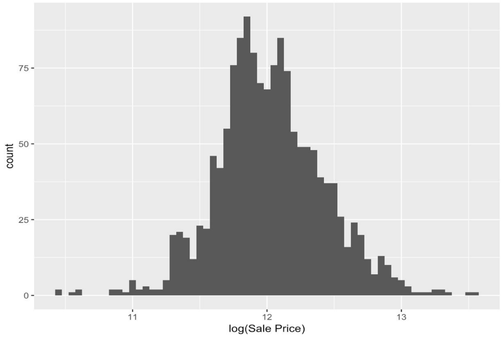
There are 80 variables in the dataset to predict sale price, however, they are not all useful. For some near-zero variance variables, fraction of unique values over the sample size is low, and the ratio of frequency of the most prevalent value to the frequency of the second most prevalent value is large. These kind of predictors do not provide much information, therefore we use nearZerovar() function to detect and remove these useless predictors.

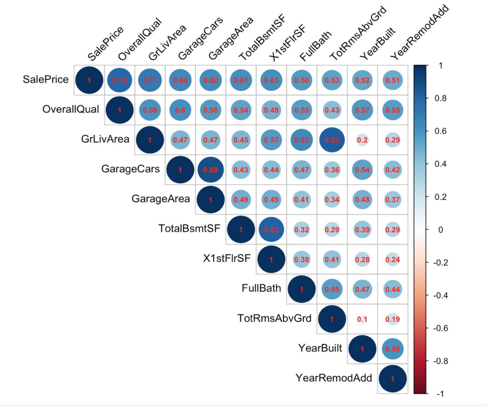
[1] https://www.kaggle.com/c/house-prices-advanced-regression-techniques

- **Visualization**

  We first visualize SalePrice and find the data is right skewed.







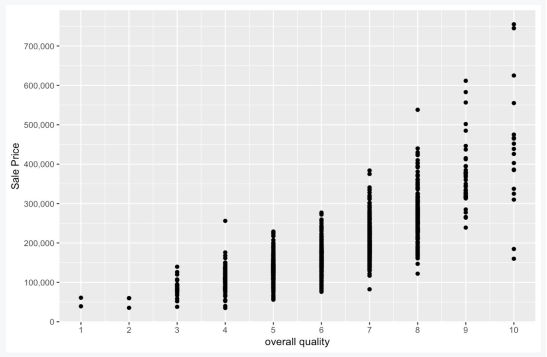
	SalePrice	OverallQual	GrLivArea	GarageCars	GarageArea	TotalBsmtSF	X1stFlrSF	FullBath	TotRmsAbvGrd	YearBuilt	YearRemodAdd
SalePrice	1	0.79	0.71	0.64	0.62	0.61	0.61	0.56	0.53	0.52	0.51
OverallQual		1	0.59	0.6	0.56	0.54	0.48	0.55	0.43	0.57	0.55
GrLivArea			1	0.47	0.47	0.45	0.57	0.63	0.83	0.2	0.29
GarageCars				1	0.88	0.43	0.44	0.47	0.36	0.54	0.42
GarageArea					1	0.49	0.49	0.41	0.34	0.48	0.37
TotalBsmtSF						1	0.82	0.32	0.29	0.39	0.29
X1stFlrSF							1	0.38	0.41	0.28	0.24
FullBath								1	0.55	0.47	0.44
TotRmsAbvGrd									1	0.1	0.19
YearBuilt										1	0.59
YearRemodAdd											1

After log-transformation: The distribution of SalePrice approximately goes to normal.

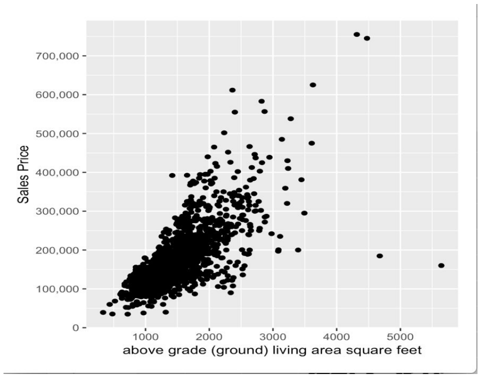
We decide to see which numerical variables has higher correlation with Sale Price first and find there are 10 numerical variables that have correlation at least >0.5 with Sale Price.

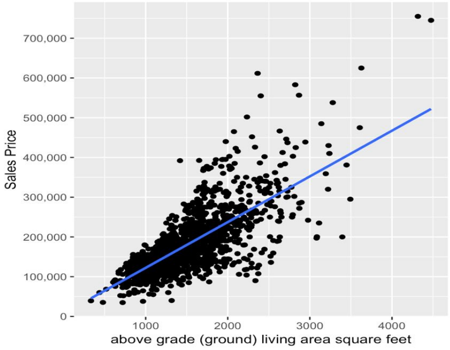
Obviously multicollinearity is an issue. For example: the correlation between GarageCars and GarageArea is high(0.89).

We try to examine deeper in two variables having highest correlation with SalePrice: OverallQuality with SalePrice(0.79) and 'Above Grade' Living Area with SalePrice(0.71)



There exists a positive trend between overall quality and SalePrice and the positive correlation is shown in the graph.





The right side picture shows there are two houses with big living rooms and low Sales Price. We try to examine the reason of low price of two houses and guess Overall Quality may attribute to the low Sale Price. However, we find these two houses score maximum points on Overall Quality. Therefore, we decide to take these two houses as outliers. The left side pictures is after taking out outliers.

- **Variable selection with Lasso**

After deleting some near-zero variance variables, we still have 58 predictors. In order to find the variables that is important in predicting house price, we need to lower the dimension of the dataset. Thus, we decide to use Lasso and coefficient path plot to further select the features.

Lasso:

When minimize the RSS, The Lasso method also puts a constraint on the sum of the absolute values of the model parameters(L1 norm): the sum has to be less than a fixed value (upper bound). In order to do so the method applies a shrinking process where it penalizes the coefficients of the regression variables shrinking some of them to zero. Lambda is the tuning parameter which controls the strength of penalty. If lambda is large enough, then the coefficient are forced to be zero, resulting the reduction of dimensionality.

$$\sum_{i=1}^{n}\left(y_i-\beta_0-\sum_{j=1}^{p}\beta_j x_{ij}\right)^2+\lambda\sum_{j=1}^{p}|\beta_j| = \mathrm{RSS}+\lambda\sum_{j=1}^{p}|\beta_j|.$$

We first fit 58 predictors in a Lasso model, and then we draw the coefficient path plot for each variable in Lasso model

Coefficient Path:

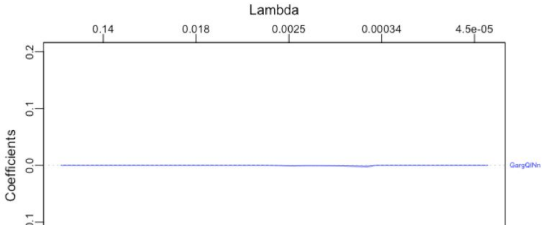
To obtain different models, Lasso constantly adjusts the level of tuning parameter Lambda. Apparently, the coefficient estimate of each variable changes with this adjustment. we resort to the pattern of such change to select the variable:

We devise three requirements and remove the variable if it doesn’t meet any of the requirements:

1. The coefficient of a variable constantly equals to or near to zero.

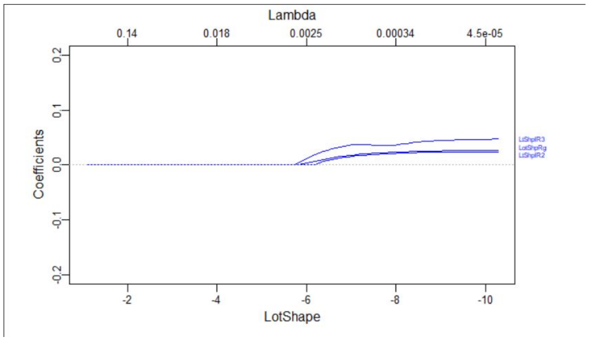
2. The variable enters the model too late. (split the graph into two part, the coefficient of certain variable only deviates from zero in the right part of graph)

3. The impact of variable on model is not consistent. (The line doesn’t maintain a stable change of slope as the lambda changes.) To make this more concrete, let’s take a look at two graphs:

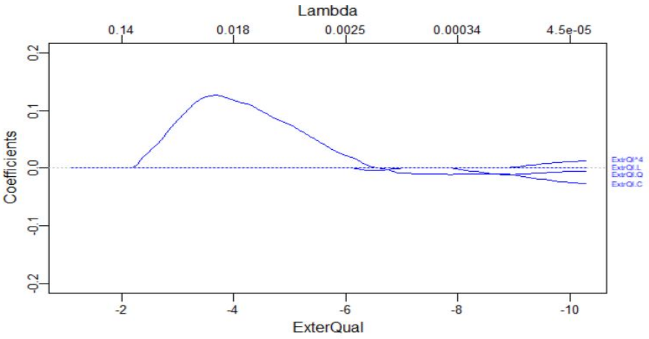


From this graph, we can see the coefficient is zero all the time due to Lasso shrinking process. This kind of predictor is abandoned by Lasso, so we remove it correspondingly.

From this graph, it is clear that the variable LotShape enters the model too late, so we decide to remove it.



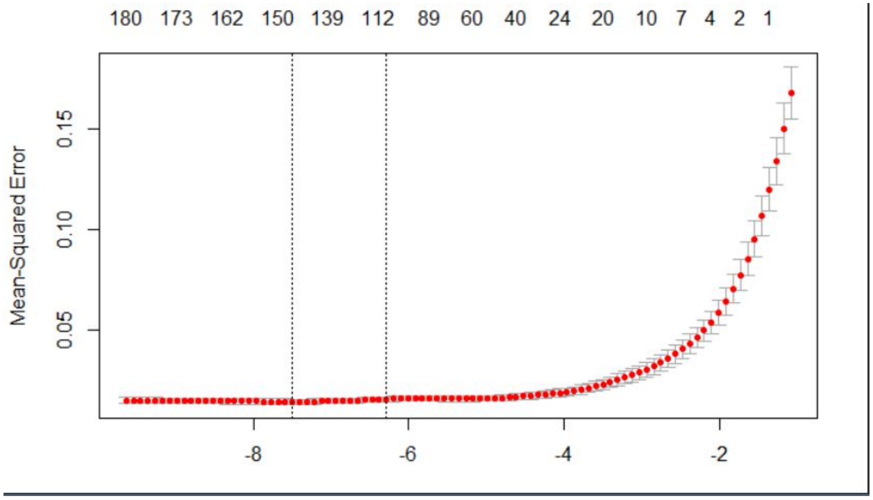
From this graph, we can see that even though the ExterQual enters model very early, it’s coefficient drop dramatically as Lambda changes. Thus we decide to remove it



(For the record, each graph only represents one variable. The appearance of multiple lines in a graph suggests that it is a categorical variable with multiple levels)

Result: In this way, we further remove twenty variables.
More on Lasso:
Besides selecting variables,the reasons of choosing LASSO are as follows:
1. LASSO can provide a very good prediction accuracy, because shrinking and removing the coefficients adjust the model to achieve a balanced point between bias and variance trade-off.
2. LASSO helps us to interpret the model more easily since we eliminate those irrelevant variables that are not connected to the response variable. In this way, overfitting is also reduced.

- **Model and its performance.**



With selected variables, we use cross-validation (we set seed to stabilize the performance)to find the optimal lambda that minimize mean cross validation error. As the graph suggests, the MSE doesn’t change too much as the log of lambda changes. Thus we decide to also built a model with lambda 1 Standard deviation away from the estimated optimal level.
As for the performance of model, according to the test score provided by Kaggle.com, the model with estimated optimal level of lambda has score 0.1188(top 20%), whereas the model with lambda 1 SE from optimal level has score 0.1182(top 20%). So basically the performance are the same between two models.