House Prices: Advanced Regression Techniques

Hangyu Liu

June 2019

1 Introduction

Already possessing some experience of Python and machine learning basics, I entered this competition on kaggle.com. The competition seemed practical, as its goal was to predict house prices in Ames, Iowa using data covering the different features of the houses collected in 2010 and available on the competition platform. There were 79 explanatory features describing detailed aspects of residential homes in Ames, Iowa, with explanations of the features provided to the participants. The competition turned out to be a valuable opportunity for me to practice machine learning, such as creative feature engineering and advanced regression techniques like random forest and gradient boosting.

In four weeks, I performed EDA, feature engineering, ensembling, stacking, and had constantly fine-tuned the parameters. The algorithms are score-oriented to achieve a better score in the competition. Eventually, after multiple submissions, I was placed in the top 4% out of more than 4,500 teams on the competition's leader board.

2 WorkFlow

1. Data Pre-processing: correlation analysis, data distribution pattern, missing value imputation, feature engineering.
2. Algorithm: Lasso, Kernel Ridge, Elastic Neural Network, Gradient Boosting
3. Tune Parameters: grid search, manual regulation.
4. Model ensemble: Base model (KRR, ENet, GBM), Meta Model (Lasso).

3 Data Preprocessing

3.1 Data Distribution pattern

Our aim is to predict the house price (abbreviated as HP), thus it's important to determine the distribution of the HP of the sample.

As shown in figure 1, the shaded blue area represents the HP density histogram plot, and the blue and black lines represent the HP density line and fitted HP density line respectively. It's obvious that the distribution of the HP is not a normal one, which may cause considerable errors in our predictions. We can also infer this from figure 2, which compares the theoretical quantile value with the real quantile value. We can see that most of the points are beyond the boundary, meaning that the distribution of HP is non-normal.

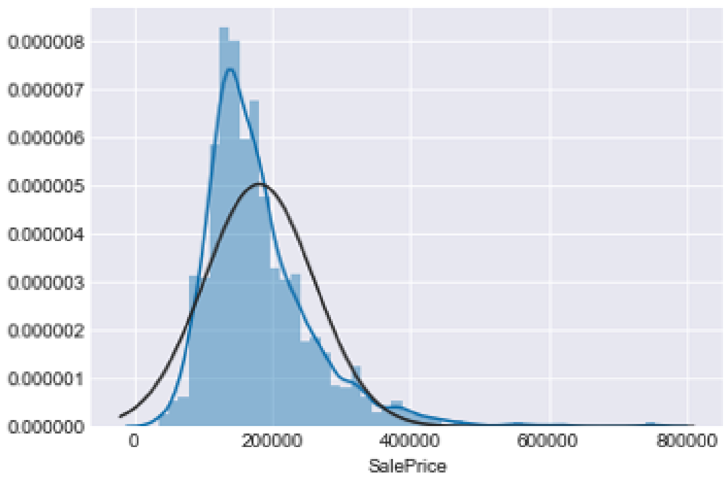


Figure 1: Original Distribution of the Sale Price

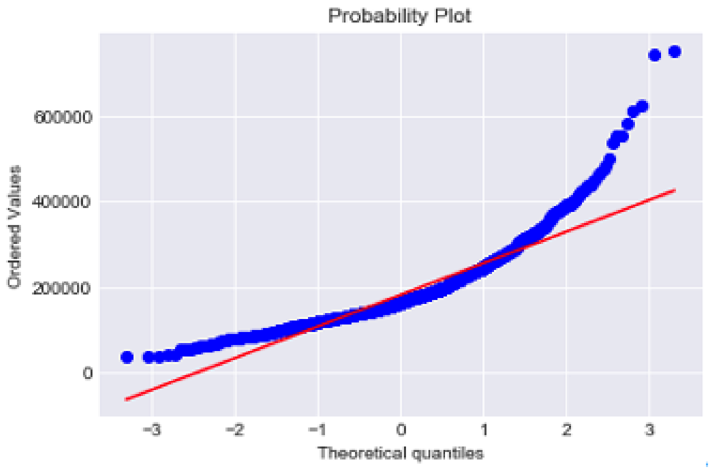


Figure 2: Original Probability Plot

This non-normal distribution shows a trailing trend, namely that most of the samples are clustering around the lesser values. Therefore, we can employ the cox-box method to handle this problem. We can observe that the samples are close to the normal distribution after using the cox-box method, in figure 3 and 4.

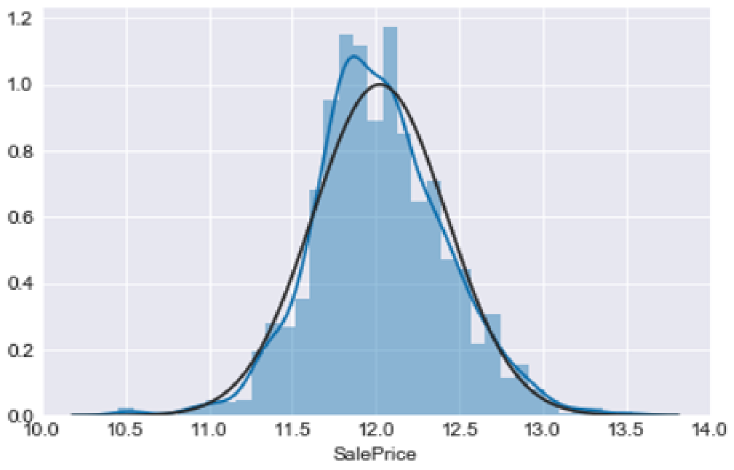


Figure 3: Distribution of the Sale Price after Normalization

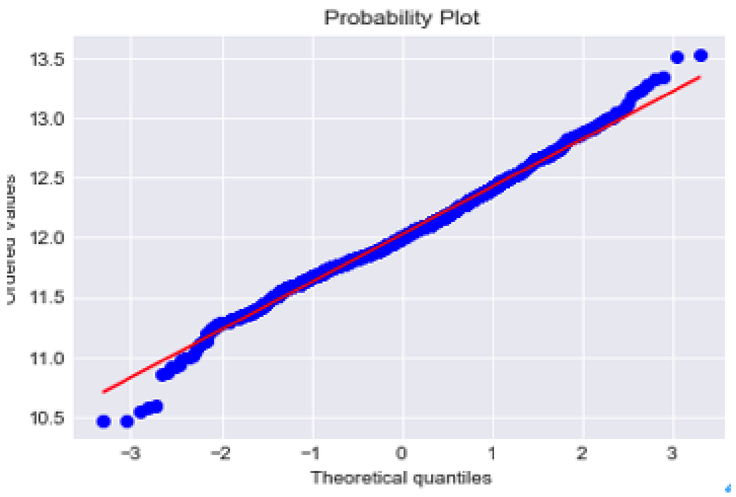
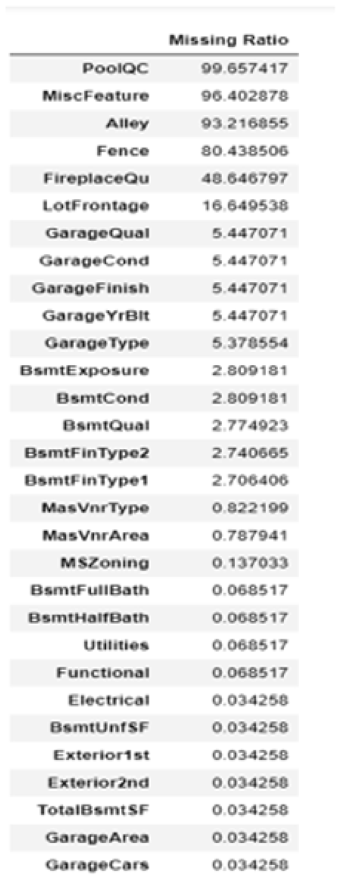


Figure 4: Probability Plot after Normalization

3.2 Missing Values

The next important process is dealing with missing values. In figure 5, it is easy to see that 30 columns contained missing values. We dealt with the missing values as table 3 shows.



	Missing Ratio
PoolQC	99.657417
MiscFeature	96.402878
Alley	93.216855
Fence	80.438506
FireplaceQu	48.646797
LotFrontage	16.649538
GarageQual	5.447071
GarageCond	5.447071
GarageFinish	5.447071
GarageYrBlt	5.447071
GarageType	5.378554
BsmtExposure	2.809181
BsmtCond	2.809181
BsmtQual	2.774923
BsmtFinType2	2.740665
BsmtFinType1	2.706406
MasVnrType	0.822199
MasVnrArea	0.787941
MSZoning	0.137033
BsmtFullBath	0.068517
BsmtHalfBath	0.068517
Utilities	0.068517
Functional	0.068517
Electrical	0.034258
BsmtUnfSF	0.034258
Exterior1st	0.034258
Exterior2nd	0.034258
TotalBsmtSF	0.034258
GarageArea	0.034258
GarageCars	0.034258

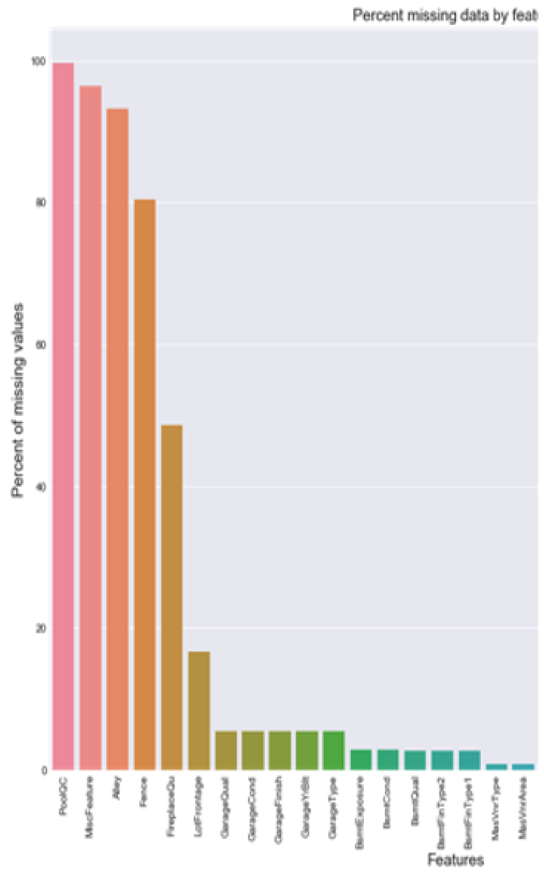


Figure 5: Missing Values

Method	Feature
None	PoolQC, Alley, Fence, FireplaceQu, MiscFeature, GarageType, GarageFinish, GarageQual, GarageCond, BsmtQual, BsmtCond, BsmtExposure, BsmtFinType1, BsmtFinType2, MasVnrType, Functional
0	GarageYrBlt, GarageArea, GarageCars, BsmtFinSF1, BsmtFinSF2, BsmtUnfSF,TotalBsmtSF, BsmtFullBath, BsmtHalfBath, MasVnrArea.
Most Frequent Class	MSZoning, Electrical, KitchenQual, Exterior1st
Median Replacement	LotFrontage
Drop	Utilities

Table 1: Missing Value Imputation

For the “None” group, which includes features such as ‘PoolQC’, ‘MiscFeature’, ‘Alley’, ‘Fence’, ‘FireplaceQu’, etc, missing values occurred mainly because most of the houses didn’t have these facilities. So it is safe to fill NA with “None”. Also, for houses with no basement, garage, or wall cladding, missing values may show up in numerical features like ’GarageYrBlt, ’BsmtFinSF1’, ’BsmtFinSF2’, etc. So we can fill those missing values with 0.

By observing the feature ‘MSZoning’, ’Electrical’, ’KitchenQual’, ’Exterior1st’, ’SaleType’, ’Exterior2nd’, I discovered that all of them frequently showed a value not determined by the sample’s other features. Therefore I filled those missing values with the most frequent value within the sample.

The ‘LotFrontage’ usually showed a definite value in a definite area, thus we can simply fill in this feature’s missing value with a median value of this area which is grouped by ‘Neighborhood’. For the feature ‘Utilities’, all records are ”AllPub”, except for one ”NoSeWa” and two NA’s . Since the house with ’NoSewa’ is in the training set, this feature won’t benefit our predictive model and could thus be removed.

Now we have filled in all of the missing values.

3.3 Correlation Analysis

In order to analyze the correlation between each feature and the housing price, we plotted the correlation heat map of these features in Figure 6. We discovered that some numerical features, which are correlate highly with the price. In particular, there are strong positive correlations between “TotaLBsmtSF” and “1stFlrSF”, as well as “GarageAreas” and “GarageCars”. Such information is useful in determining the correlation of features and offering evidences beneficial to performing imputations. We also know that multi-collinearity may make it more difficult to make inferences about the relationships between our independent and dependent variables.

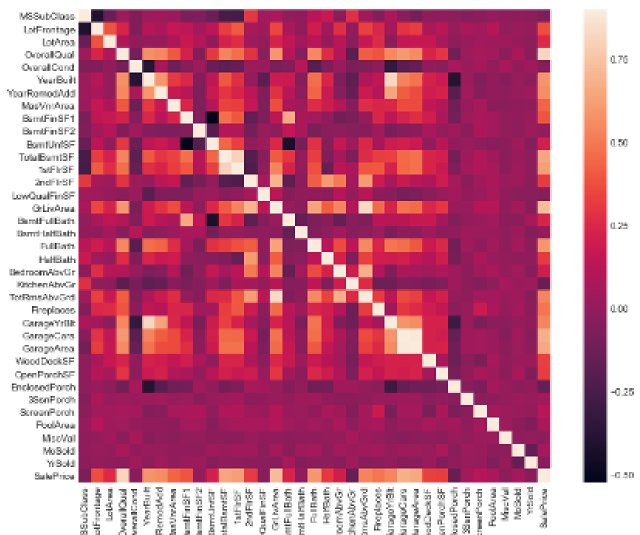


Figure 6: Correlation Before Imputation

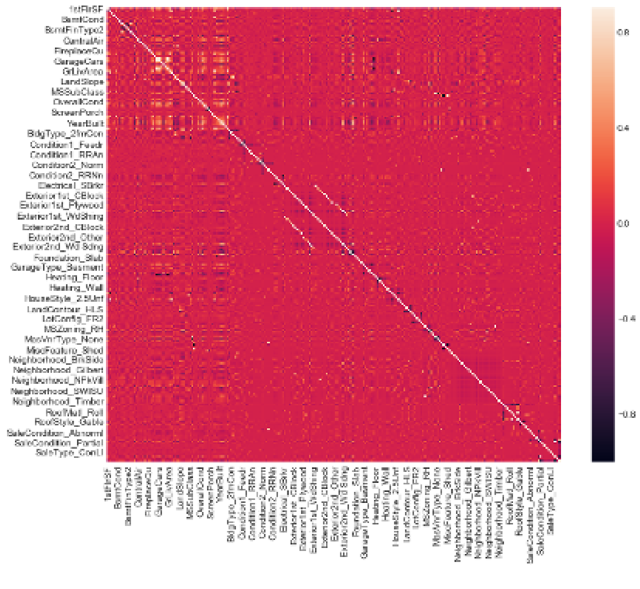


Figure 7: Correlation After Imputation

3.4 Feature Engineering

3.4.1 TRANSFORMING THE DATA TYPE

In the algorithm to be applied next, the data type should be legal, which means that we should transform some of this data.

In one method, we transferred some numeric features to categorical types including ’MSSubClass’, ’OverallCond’, ‘’YrSold’, ’MoSold’.

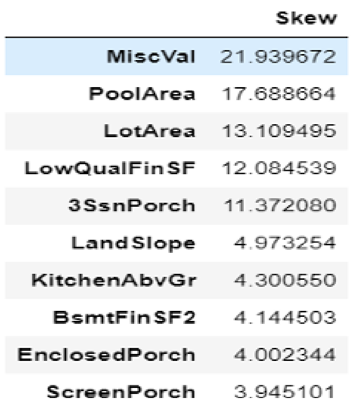
Another way, which allows for categorical features with three or more classes, is to employ the method of ‘LabelEncoder’ to transform them, which includes ’FireplaceQu’, ’BsmtQual’, ’BsmtCond’, ’GarageQual’, ’GarageCond’, ’ExterQual’, ’ExterCond’,’HeatingQC’, ’PoolQC’, ’KitchenQual’, ’BsmtFinType1’, ’BsmtFin-Type2’, ’Functional’, ’Fence’, ’BsmtExposure’, ’GarageFinish’, ’LandSlope’,’LotShape’, ’PavedDrive’, ’Street’, ’Alley’, ’CentralAir’, ’MSSubClass’, ’OverallCond’, ’Yr-Sold’, ’MoSold’. At last we implement ‘One-Hot-Encoding’ in order to make dummy variables out of all the categorical data which will have better effects on the algorithms.

3.4.2 Adding a New Feature

According to general knowledge, the mean house price for different parts or floors of the house are the same. In this way, we can combine the features ”TotalBsmtSF”, ”1stFlrSF”, and ”2ndFlrSF” into a new feature, namely ”To-talSF”.

3.4.3 Feature Skewness

In most algorithms, data with Gauss distribution can minimize error. Thus, it was preferable to discover the skewing features and then correct them by calculating the relative asymmetry of the data.



	Skew
MiscVal	21.939672
PoolArea	17.688664
LotArea	13.109495
LowQualFinSF	12.084539
3SsnPorch	11.372080
LandSlope	4.973254
KitchenAbvGr	4.300550
BsmtFinSF2	4.144503
EnclosedPorch	4.002344
ScreenPorch	3.945101

Figure 8: Skewness of the Feature

As shown in figure 8, I selected the top 10 skewing features, all of which display high rates of skewness. For all of the numerical data within the data set, I set 0.75 as the threshold of the skewness. In this condition, I found 59 features in total to be corrected. After histograming the density map of the features, I employed box-cox method to correct the skewing (LAM = 0.15).

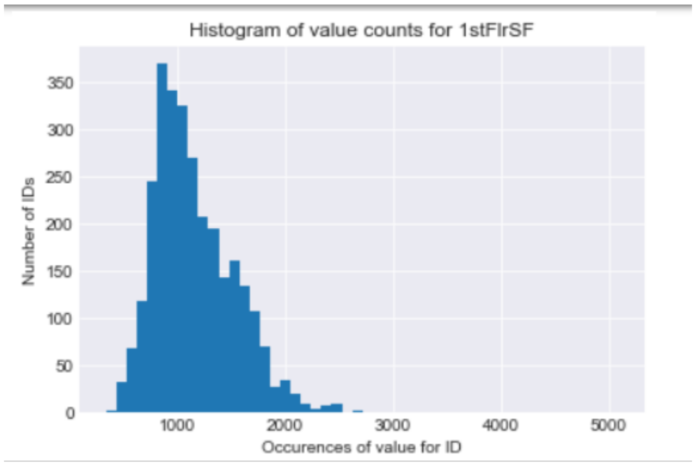


Figure 9: Skewness of 1stFirSF Before Boxcox

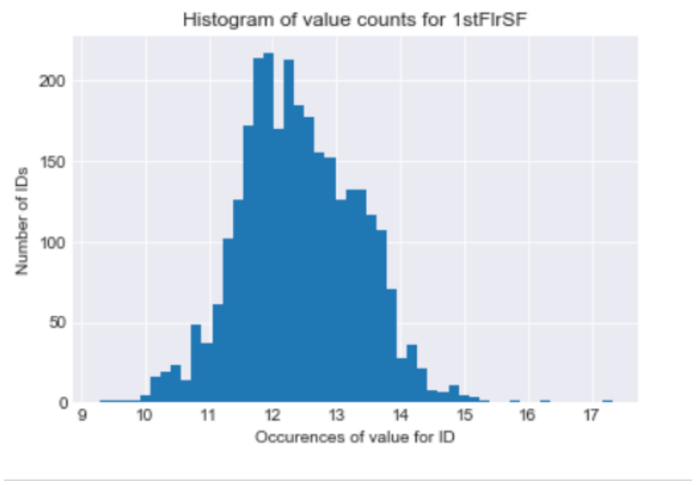


Figure 10: Skewness of 1stFirSF After Boxcox

4 Models

4.1 Elastic Net

In statistics, and in particular the fitting of linear or logistic regression models, the elastic net is a regularized regression method that linearly combines the lasso and ridge methods. There are two parameters we need to tune for: alpha, I1_ratio.

- Alpha: the constant that multiplies the penalty
- l1_ratio: the ElasticNet mixing parameter, with 0 <= l1_ratio <= 1. For text l1_ratio = 0, the penalty is an L2 penalty. For l1_ratio = 1 it is an L1 penalty. For 0 < l1_ratio < 1, the penalty is a combination of L1 and L2.

Based on the process, we decided: Alpha[0.001,0.1], I1_ratio[1,2]

We then added the new parameters "random_state", the seed of the pseudo random number generator that selects a random feature to update.

We used the MSE in order to judge the score and to discover the most ideal parameters.

4.2 BGM

The next step is to apply gradient boosting. There are 6 tuning parameters: n_estimators, learning_rate, max_depth, max_features, min_samples_leaf, min_samples_split.

Parameter	Note
n_estimators	The number of sequential trees to be mod-eled.
learning_rate	This determines the impact of each tree on the outcome. GBM works by starting with an initial estimate which is updated using the output of each tree. The learning parameter controls the magnitude of this change in the estimates.
max_depth	The maximum depth of a tree.
max_features	The number of features to consider while searching for the best split.
min_samples_leaf	Defines the minimum samples (or observations) required by a terminal node or leaf
min_samples_split	Defines the minimum number of samples (or observations) which are required by a node to be considered for splitting.

Table 2: Missing Value Imputation

There are two types of parameter that we need to tune here, tree based and boosting parameters. We knew that there were no optimum values for learning rates as low values always work better, given that we trained on a sufficient number of trees. While GBM is robust enough to prevent over-fitting within an increased number of trees, a high number for a particular learning rate can still lead to over-fitting. As we reduced the learning rate and increased trees, the computation took a long time to run and produce results.

In such a case, we adopted the following approach. Initially, we used 0.05, however it did not produce a favorable result. Thus, our second attempt saw us choosing the default value of 0.1. Meanwhile, we lowered the learning rate and increased the estimators proportionally in order to produce more robust models. Additionally, it is worth mentioning that you should pay careful attention to the order of tuning variables. For example, the parameters that exert greater impacts on the result should be tuned firstly, such as max_depth and min_samples_split, and then you could tune min_samples_leaf and max_features.

4.3 Kernel Ridge

Let's have a look at our liner machine learning model, Kernel Ridge Regression (KRR). KRR combines ridge regression (linear least squares with l2-norm regularization) with the kernel trick. It thus learns and adapts a linear function in the space induced by the respective kernel and the data. For non-linear kernels, this corresponds to a non-linear function within the original space. In our case, there were four parameters, including alpha, kernel, degree and coef0.

4.4 Lasso

Lasso is a powerful technique generally used for creating parsimonious models in the presence of a large number of features. The loss function is based on the 'Least squares'. Lasso adds the L1 penalty in order to penalize a large number of coefficients. What's more, it can also minimize the coefficient to 0 when certain features fail to work within the model. We used the grid search method to determine the parameter's alpha value. Lasso is a linear method, and therefore it can be used to enhance the generalization of the model in the sections of the samples where the price is highly related to the area of the house. But the actual model is so much more complicated than that, and thus we chose to use Lasso as the meta model for the stacking model, and the implementation has been a success, ultimately producing very accurate predictions.

4.5 Final Parameters

- GradientBoostingRegressor: $\bar{2}$000, learning_rate=0.1, max_depth=4, max_features='sqrt', min_samples_leaf=13, min_samples_split=9, loss='huber', random_state =5
- Lasso: alpha =0.0011, random_state=1
- Elastic net: alpha=0.0018, l1_ratio=1.0, random_state=9
- KernelRidge: alpha=0.2, kernel='polynomial', degree=2, coef0=5.0

4.6 Summary

Model	Description	Pros	Cons
Lasso	Lasso regression is a powerful techniques generally used for creating parsimonious models in the presence of a 'large' number of features	regularization term restrains the efficiency brought about by the increased data size. The reg. term can minimize to zero to totally move the non-relative features.	Sensitive to the outliers, which might cause over-fitting.
Elastic Net	In statistics, and in the fitting of linear or logistic regression models in particular, the elastic net is a regularized regression method that linearly combines the L1 and L2 penalties of the lasso and ridge methods	Enforces sparsity. No limitation on the number of selected variables. Encourages a grouping effect in the presence of highly correlated predictors	Native elastic net suffers from double shrinkage
KRR	kernel ridge regression (KRR) combines ridge regression (linear least squares with l2-norm regularization) with the kernel trick. It thus learns a linear function within the space induced by the respective kernel and the data. For non-linear kernels, this corresponds to a non-linear function within the original space.	Strong theoretical guarantees. Generalization to outputs in the single matrix inversion. Uses kernels.	Solution not sparse. Training time for large matrices is a low-rank approximation of kernel matrix.
GBM	Produces a prediction model in the form of an ensemble of decision trees. Builds the model in a stage-wise fashion. Generalizes results by allowing for the optimization of an arbitrary differentiable loss function	High-performing. Uses all features. Lower possibility of over-fitting with more trees.	Susceptible to outliers. Lack of interpretability and higher complexity. Harder to tune the parameters. Slow training speed.

Table 3: Missing Value Imputation

5 Averaging Based Ensemble Methods and Stacked Ensemble Methods

The implementation of only one model cannot independently produce the best score, so we decided to ensemble all the models in order to figure out the best way to predict the outcomes.

5.1 Simple averaging model

Within the simple averaging method, for every sample of test datasets, the average predictions are calculated. This method often reduces over-fitting and creates a smoother regression model. The scoring method is MSE, and it turn out that the more linear the regression, the better the score is. Ultimately, the best score was produced by the combined efforts of lasso, elastic net, KRR, and GBoost

5.2 Stacked

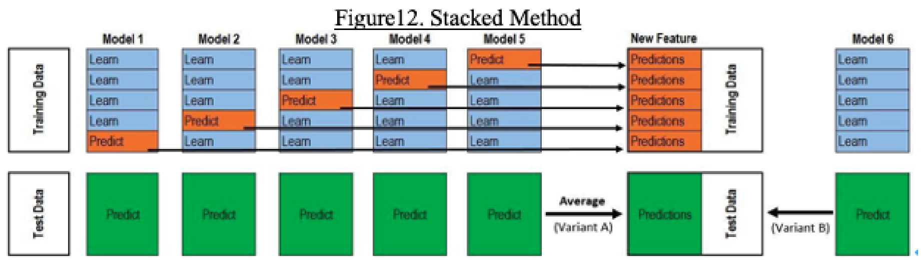
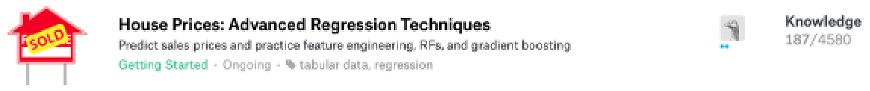


Figure 11: Stacked Method

As shown from the two layers of stacking in figure 11, the first layer has five models and the second has only one. The role of the first layer model is to train an $R_{n\times m}$ feature matrix for inputting the second layer model training, where n is the number of training data rows and m is the number of first layer models. We used lasso for the second model and the other models for the first layer. If the first layer has less than five models, we can ensemble the left model by changing the weight. We employed the following equation: “ensemble = stacked_pred*0.70 + averaged_pred*0.2 +xgb_pred*0.1”, and after careful processing, the results were as follows: the first layer is made up of KRR, elastic net, and GBoost, and the second layer is lasso. In this way, the best score is achieved.

6 Final Score



House Prices: Advanced Regression Techniques
Predict sales prices and practice feature engineering, RFs, and gradient boosting
Getting Started · Ongoing · tabular data, regression

Knowledge
187/4580

7 Appreciation

I would like to thank Shuo Zhang (Ph.D, Columbia University) for answering question from me throughout the project, and making it possible for me to complete the project so thoroughly.