**House Price in the City of Ames**

Xizi Zhu, Zhongyuan Ma

**Abstract**

The city of Ames, Iowa provides the dataset of 2,929 houses sale prices with 79 attributes involved in assessing the house values. Dataset is split into two portions: 80% of data are used to train the models, and the rest 20% are used for testing. This project has two objectives: 1. Create a linear model that predicts the house price with the given variables. 2. Create a model that predicts what houses have the sale price below or above $200,000, which is the average price in this city.

**1. Data Cleaning**

The first problem with this dataset is many variables have the data field “NA”, but “NA” here doesn’t mean missing data. For example, for variable “Alley”, “NA” means no alley access. Therefore, we had to change all the “NA” values in order to allow Python to process the data correctly.

The second problem we have dealt with is the unbalanced variables in this dataset. Unbalanced variables can make it difficult for us to conduct cross-validation (some field occurs in testing dataset but not the training). For example, variable MSZoning identifies the general zoning classification of the sale. “Residential” zoning consists of more than 90% of the data, and the other 4 zoning attributes consist less than 10%. When we encounter problems like this, depends on the significance of the variable in predicting the sale price, we either delete the variables or merge some of the attributes of such variable to make it less unbalanced.

Lastly, we use one-hot encoding to transform all the nominal variables. As for the ordinal variables, we assume that the distance between the adjacent level is equal and transform into this format: high (1, 1, 1), medium (1, 1, 0), and low (1, 0, 0).

**2. Model Selection**

**2.1 Linear Model**

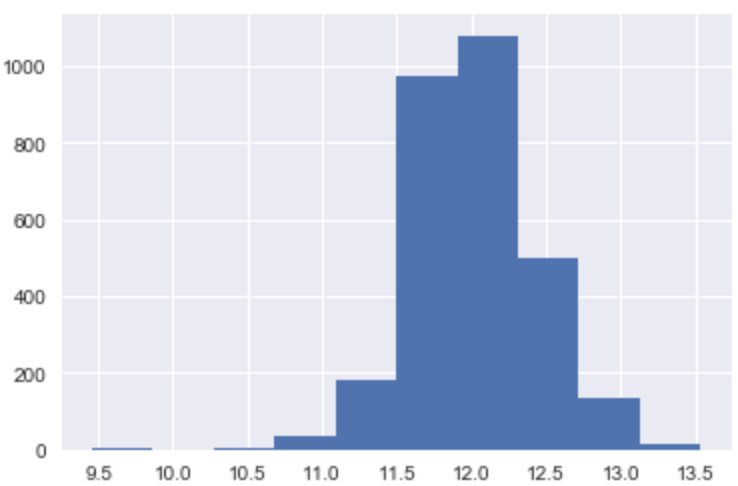
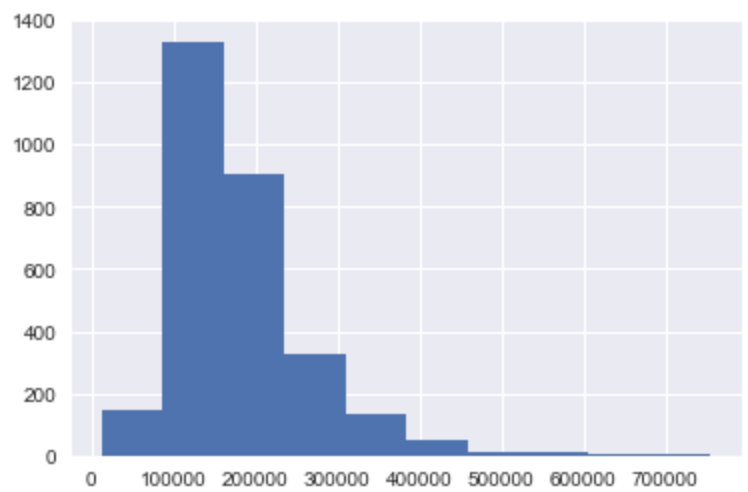
We run a 5-fold cross-validation on the training dataset in order to choose the best model.

We started with a vanilla linear model with all of the variables provided. This model, as we have imagined, surely runs a high risk of overfitting the data as after variable transformation, there are more than 400 columns with merely 3,000 data points. This baseline linear model (in sample *R2* 0.887, estimated out of sample *R2* 0.840, **mean outcome 180276**) surprisingly gives a relatively good result. We expected this model to have a very low bias given it contains all the variables, and to have a very high variance.

Based on the coefficients and the p-value from the variables in the baseline model, we decided to hand-pick some variables that we believe might affect the sale price the most, hoping doing this can greatly lower the variance of the model. Out of the 79 variables, we selected 20 variables that we decided to be meaningful and significant in predicting sale price. For example, *Overall Quality, Overall Condition, Gr Living Area*, etc. After deleting so many variables, this model still has a pretty good performance: in sample *R2* 0.862, estimated out of sample *R2* 0.813.

Given the previous two models, we decided that hand picking the variables might not be a good choice. It is likely that we dropped some variables that might be significant in predicting the house price. Therefore, we want to use the *Lasso* and *Ridge* regularizer to help to help us find the most important variables. By forcing the sum of the absolute value of the regression coefficients to be less than a fixed value, which forces certain coefficients to be set to zero, Lasso regularizer effectively chooses a simpler model that does not include those coefficients. The model drops more than 150 of 400 columns and achieves a decent result: in sample *R2* 0.881, estimated out of sample *R2* 0.865. The *Ridge* regularizer, which shrinks some of the large coefficients to reduce overfitting, has in sample *R2* 0.883 and estimated out of sample *R2* 0.827.

Both of these models give relatively good result, but we decided to improve upon the *Lasso* model. Though *Ridge* model also gives a decent result, we choose to use *Lasso* to performs variable selection, and thus makes the model more interpretable. After seeing the cross-validation result, it is also possible that the *Lasso* will be more resilient when it is used on the test data.



**Figure 3.1.1 SalePrice Histogram** **Figure 3.1.2 log(SalePrice)Histogram**

One interesting fact we noticed is that Sale Price is highly skewed. Most of the sale prices are on the lower spectrum, and after conducting the log transformation on the sale price, they have a relatively normal distribution. We then fit a linear model with *Lasso* regularizer, which has in sample *R2* 0.912 and estimated out of sample *R2* 0.882. This model is the best model we have so far. By doing log transformation, we make sure that large sale price won’t influence the linear model by too much.

**2.2 Classification Model**

As is in the previous case, a 5-fold cross-validation is ran on the training dataset in order to choose the best model.

The setup of this classification problem is different. We create a binary entry, which is 1 if the sale price is larger than 200,000 and 0 otherwise. The reason we are setting up this entry is that we would like to look into whether a housing price is likely to be above threshold. The application of it can be: if real estate is heavily taxed above certain threshold and it is difficult to come up with the current pricing of the house, we can apply our model to see whether this house should be heavily taxed.

We considered **0-1 loss function** in our first baseline model in the classification problem but did not use it as our objective because 0-1 loss function does not punish the large price difference enough and a sale price way below 200,000 and one that is close to 200,000 (still below) are totally different in real world, and giving them the same weight does not make sense.

Therefore, our standard of evaluating whether this model is good is ***Recall***, also known as True Positive rate. It is adopted as metric because if one were to purchase the house, he would rather not buy a house that was not identified as having the potential to be taxed.

We used 6 different models in this case:

base linear (out-of-sample ***Recall*** = 0.82),

selected features linear(out-of-sample ***Recall*** = 0.84),

knn (out-of-sample ***Recall*** = 0.19),

logistic(out-of-sample ***Recall*** = 0.91),

Lasso Selected logistic (out-of-sample ***Recall*** = 0.84),

logistic with selected features(out-of-sample ***Recall*** = 0.85),

Our most basic model is a linear classification model. Since we so many features, we want to extrapolate some information through their linear combination. Also, because this binary outcome variable is derived from the linear outcome variable “SalePrice”, and the linear model performed great in predicting the “SalePrice”, it is reasonable to use the linear model to predict the derived binary outcome.

We then performed some classic classification models on the data: KNN and logistic regression. We used cross-validation to pick the best k value, but KNN still performs poorly. As for logistic regression, we tried different inputs. “logistic” model takes all the features as input, “Lasso Selected logistic” takes the important picked by Lasso in the linear model as inputs, and in “logistic with selected features” we hand-pick some important features as input. The reason we did the last two is to reduce the variance of this model. Selected variable model does not differ a lot from baseline model in term of ***Recall***, but it has less variance.

The reason the last one is the best model is because logistic is a good model for classification for its less emphasis on the outliners and more focus on what is on the edge. It is also influenced by a lot of features, these features might not be important to predicting the exact numbers, but they are more important when it comes to prediction of the edge. However, despite being the best model, it might miss when predicting on held-out test set because 200,000 is not a real-life threshold, and this dataset is an unbalanced corpus (more prices below 200,000 and fewer prices above it).

**3. Test Result and Future Improvement**

**3.1 Linear Model**

We picked the linear model with *Lasso* regularizer and log transformation to be our best model and tested it on the test data. The result is very promising: . This model gives a very accurate performance of the house sale price in this city. If we look at the data, the average sale price is around $200,000, the RSME of this model is quite small compared to the average sale price.

However, there are multiple things I would like to try if I am given more time and resources. First, I would like to fit linear models with different loss functions, for example, Huber loss. We don’t want the expensive houses to skew our model. Huber loss can do this by punishing small errors with quadratic loss, and larger errors with a linear and more robust loss function. The second thing I want to explore is the variable neighborhood. I think if the dataset is large enough, we really should fit different linear models for different neighborhood. Intuitively, it makes sense that different neighborhood or people from different income classes might value things differently. Some neighborhood may value things that are practical and some might value things that are luxury.

**3.2 Classification Model**

base linear (***Recall*** = 0.82 v.s. estimated out-of-sample ***Recall*** =0.82)

logistic(***Recall*** = 0.84 v.s. estimated out-of-sample ***Recall*** = 0.91)

Comparatively, logistic (best model) still performs better than the base linear.

Since this binary variable is created out of the thin air by us, we want to know what is the price threshold that concerns people living in Ames. Maybe houses above $250,000 are the ones that are getting higher taxes, or maybe houses below $100,000 can receive subsidy from the government. We want to find a more reasonable binary response variable for this dataset if we are given the chance to talk to the people who live in Ames.

**4. Conclusion**

A good prediction model can help prospective buyers to evaluate house values prior to their purchases. Among the linear models we explored, quadratic loss with *Lasso* regularizer appears to capture the price-determining factors well and offer a good prediction. However, in order to make this model more robust, I suggest fitting the model with new house sales data every 3 months. Seasonal effect and time really play a large role in real estate market, and the price fluctuation of this market can sometimes be very volatile. In addition, in order to produce a better result in predicting the house price, more details and explanations need to be provided in some of the variables. For example, *Overall Cond* has a rating from 1 to 10 (poor to excellent). The process taken to give out these ratings can be essential for us to construct a good model.

In term of classification problem, the result is relatively good with training cross-validation data. The test is off to certain extent, but due to the nature of the dataset, it is likely to happen. Further down the road, I would like to explore more techniques in machine learning, such as SVM and CNN, which can allow us to better leverage the numbers of our data.

Code are attached below

The first part is for linear models, and the second part is for classification models