How Much Is My House Worth?
Predicting Housing Prices in McKinney, Texas

Nadin El-Yabroudi, Paul M.R. Harrison

Abstract

The goal was to give homeowners a prediction on their house's closing price which would be put on the market. Classification and regression methods were used as well as natural language processing on each house's remarks. The best set of models found used Lasso L1 regularization. Prediction performed remarkably better than classification, and predicted close prices with 95% confidence with this interval: $-\$54,400$ to $\$51,800$

The goal of this project is to create a model that can best predict the closing price for a house before it has been placed on the market. This is important as it can help homeowners not only estimate a listing price but serve as a second opinion if need be. The project is based on data from the company Opendoor, which uses machine learning to predict housing prices and make selling your home easier. Our dataset is from home sales in the small town of McKenney Texas from late 2014 to mid 2015. Although it has only about 1100 observations, it provides many different features for the data such as area of the house, types of floors, number of bedrooms, etc.

Dataset and Features

Cleaning the data set included eliminating features that were relevant only once the house had been sold since we were only interested in predicting the price that a house should be given when entered into the market. We then converted categorical features such as the type of flooring of the house into binary ones. We also created new features such as binary features for the season in which the house was listed on the market and the age of the house. These modifications to our dataset largely increased the number of features in our training set so that at the end we had 83 features. Once we had cleaned our data set we looked at the shape of the distribution of the target feature, the closing price of the house, to understand how to best fit the data. The closing price seemed to follow a normal distribution skewed left, and the log normal of the distribution looked closer to a normal distribution not skewed in any direction. We decided to test how our models did predicting the log of the closing price as well as just the closing price.

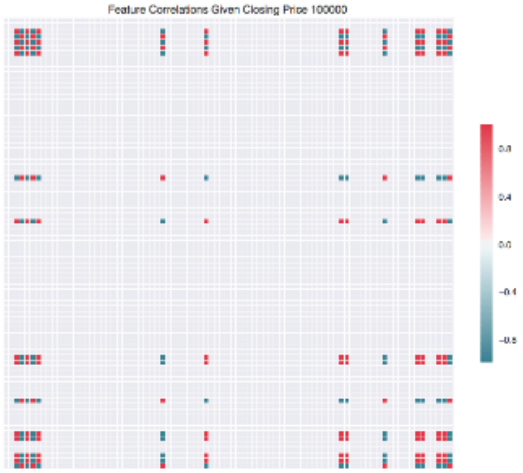
Training Methods

Our training and model selection methods throughout the project were as follows. We split our dataset into a training and testing set by a 70/30 split. Throughout the first part of the project we used K-fold Cross Validation with k=10 to approximate a test error. This proxy test error helped us make decisions about which models to select and what to try next without including bias into the testing phase. During the testing phase we trained on the whole training set and tested on the test set only with the best model found with the cross validation error.

As a baseline model, we decided to use a Naive Bayes classifier for our predictions. In order to use this model we needed to discretize our values including our target variable. Market research on online housing real estate engines such as Zillow, showed that it is common to split houses into brackets of approximately $\$40,000$. Using this value for our bucketing, we were now trying to predict the price of a house within ten possible buckets. First, we implemented two types of Naive Bayes: Gaussian Naive Bayes and Multinomial Naive Bayes. Multinomial Naive Bayes performed much better probably because a greater majority of our features were binary, than continuous. Nonetheless, Multinomial Naive Bayes only achieved 28% accuracy, even with the log of the target values. To explain the performance of Naive Bayes, we obtained correlation plots on features conditional on their actual target price (Figure 1). The sample plot shows high dependency between the continuous features, especially in the more extreme closing price values, where we had less data points.

To improve our classification we needed to relax the model assumption, so we decided to try a Multinomial Logistic Regression, or logit classification on the same bucketed dataset. This model outperformed Naive Bayes reaching an accuracy of 49% after all variables, such as regular-

Figure 1: Correlation Plot of Features Conditional on $\$100,000$ Closing Price Bucket



ization, interaction terms, and time (which we address later in this paper), were maximized. We believed that discretizing our features, by which we were losing information for each sample, was causing the low accuracy with classification. Therefore, we decided to try a regression model.

Our first regression model was ordinary least squares linear Regression without any regularization and found that for certain samples, the model predicted prices up to 1000 times too big, suggesting that regularization was key to this model. Lasso linear regression and ridge linear regression performed much better. To evaluate the regression model we used the metric mean squared percentage error (MSPE). Since the price values were numbers on the range of 1e5 by using a percentage difference metric we could more manageably compare the errors. Using the log of the target values on the lasso regularization we were able to obtain a 0.78% mean squared error.

Finally, we decided to compare whether regression had really done better than classification. We could see that a 0.78% mean squared error was well within the range of $\$40,000$ with which we had bucketed our features before. Yet, we had not tried to see if classification did better with smaller bucket sizes. As such we ran logit classification on a data set with the target values discretize in buckets of $\$20,000$, yielding 20 buckets. However, this model did much worse than with the original bucket size, most likely because creating more buckets meant that each bucket had less samples. Therefore, we could predict with more granularity the price using regression and with a high accuracy.

Optimizing Performance for Model Variables

1. **Time Dependence.** Our goal was to predict the closing price of houses in the future, which meant time was an important factor in our modeling. The dataset was ordered chronologically by when the house was placed on the market, our worry was that if we only trained on houses sold a year ago market changes through time would not be taken into account. Therefore, to test the importance of time in our model we trained our models on a chronologically ordered dataset and on a dataset in random order. For classification it was hard to find a pattern between whether random or chronological did better: for the logged target values chronological did better, but for the others random did better. However, for regression we could clearly see that random dataset was doing far better than chronological one.

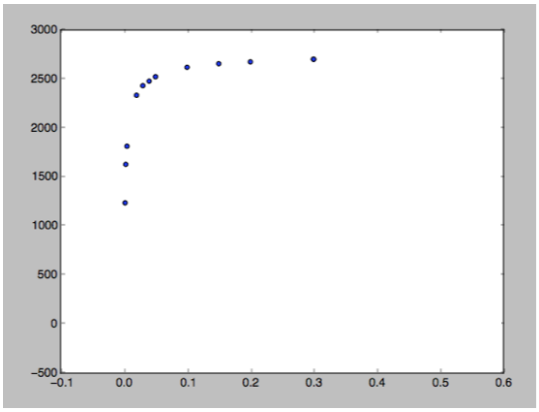
   Therefore, we decided that time did play an important part in our dataset and we wanted to account for it by adding a feature to our training matrix. Using the date in which a house was placed in the market, we created a feature which determined how recent the sample was. For both classification and regression, we saw that the time feature improved the performance of our model under cross validation.

2. **Feature Interaction Terms.** We had learned from Naive Bayes that our features had high dependency, hence, we wanted to use this fact to improve our model. As such we added interaction features of degree two to our dataset, in other words, we created 83 choose 2 new features which represented the product of every pair of features. This brought the number of features in our model up to 3486, which was more than the number of samples in our dataset. To counteract the significant increase in dimensionality of our model, as we will see in the next section, we used aggressive regularization. Using the feature interactions we were able to improve our models slightly. For the logit classifier we obtained 48.2% accuracy without interaction terms, and 49.2% accuracy with interaction terms. For lasso linear regression we obtained 0.81% MSPE without interaction terms, and 0.78% MSPE with interaction terms.

3. **Optimizing Performance with Regularization.** For both the logit classifier, and lasso linear regression, regularization played an important part in model performance.

   First for classification we had to select the appropriate c-value, which was inversely proportional to the intensity of regularization. We found that for the interaction terms we needed to use strong regularization, c=0.0001, while for the dataset without the interaction terms only a small amount of regularization was necessary, c=0.6. This suggests the the interaction terms resulted in overfitting our model, but regularization helped counterbalance this. Similarly, for regression we had a constant to control the regularization, alpha, which in this case was proportional to the data. We see that once again interaction terms require strong regularization, alpha=0.4, where the non-interaction terms data set required only alpha=0.0002. In other words, our new interactions regression model had 2000x more regularization. More interestingly we see that without interaction terms, ridge linear regression does better than lasso, but the opposite is true for the dataset with the interaction terms, further proving that these in-

Figure 2: Num Words Remaining vs. % Threshold for Words to be Removed



teraction terms are overfitting to the data. Additionally it was found that L1 regularization which is stronger i.e more aggressive in how it feature selects than L2 was crucial to finding our optimal model.

Remarks Analysis

The dataset also contained descriptions on each of the sample homes and we wanted to understand whether these could help our model predict the closing price. Therefore, we ran Bag of Words on the remarks and then used TF-IDF to remove the words that appeared more than 5% of the time which would not help us distinguish the comments. We used 5% as we noticed in Figure 2, that after this value there was a steep decrease in the number of common words, and we feared a higher value would eliminate important words. Figure 2 shows this compromise of removing unnecessary words as a function of this threshold percentage.

Next, we wanted to reduce the dimensionality of this matrix to add to our input train matrix and not have too many features. To do so we used two techniques: Principal Components Analysis or PCA and the ANOVA F-statistic. For PCA we selected the top 10% principal components and for the ANOVA F-stat matrix, we selected the top 10% of words that gave the most information about the closing price.

Appending these two matrices, PCA and ANOVA F-values, to our original matrix and running both classification and regression on it had little to no impact on prediction. For regression both remark matrices seemed to be eliminated by regularization since we obtain the same MSPE, and for classification PCA and F-value matrices gave lower accuracy rates, suggesting overfitting. Therefore, we can conclude that regardless of the dimensionality reducing method, the remarks are not useful for predicting closing prices.

Model Comparison

Below we present a summary of the results for the different model iterations that we have discussed previously.

Classification	CV Accuracy
NB Multinomial Log Y	24.8%
NB Gaussian Log Y	15.3%
Logit Log Y Chron	46.7%
Logit Log Y Rand	33.5%
Logit Log Y	48.2%
Logit Inter-Terms Log Y	49.2%
Logit Half Buckets Log Y	48.2%
Logit PCA Log Y	47.9%
Logit F-Stats Log Y	48.2%

It is interesting to note that for both classification and regression, the best model included interaction terms, and the log of the target values. This demonstrate that the relationship between features were very important for prediction as well as having a normal distribution on the target values.

Regression	CV MSPE
Ridge Log Y	0.80%
Lasso Y	10.7%
Lasso Log Y Chron	0.81%
Lasso Log Y Rand	0.80%
Lasso Log Y	0.81%
Lasso Inter-Terms Log Y	0.78%
Lasso PCA Log Y	0.78%
Lasso F-Stats Log Y	0.78%

Testing and Conclusion

Using our test set on the lasso model with interaction terms and the log of our target values i.e. our best model we obtained an MSPE of 0.71%, which was even lower than what we obtained with cross validation on our training set. Because we observed that our model's residuals followed a normal distribution, as in Figure 3, it made sense to find a confidence interval on the mean of these values. The result of this 95% confidence interval on this test set was: $-\$54,400$ to $\$51,800$.

From a first glance looking at Figure 3 and our corresponding 95% confidence interval we can see that we have correctly classified house prices with a far greater accuracy

Figure 3: Normal Distribution of Residuals on the Test Set

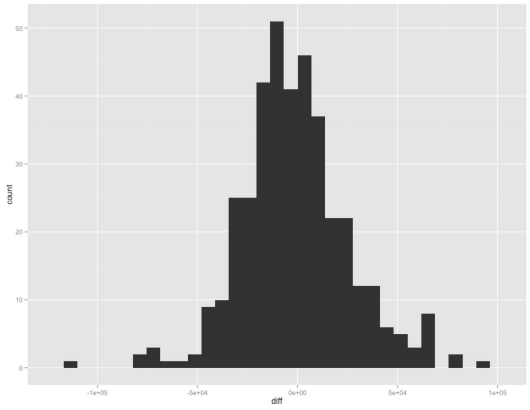


Figure 4: Residual Plot on the Testing Set

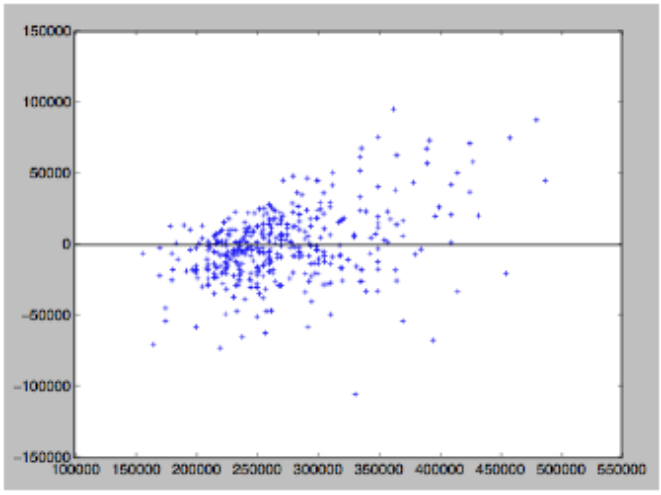


Figure 5: Boxplot for Residuals on the Testing Set

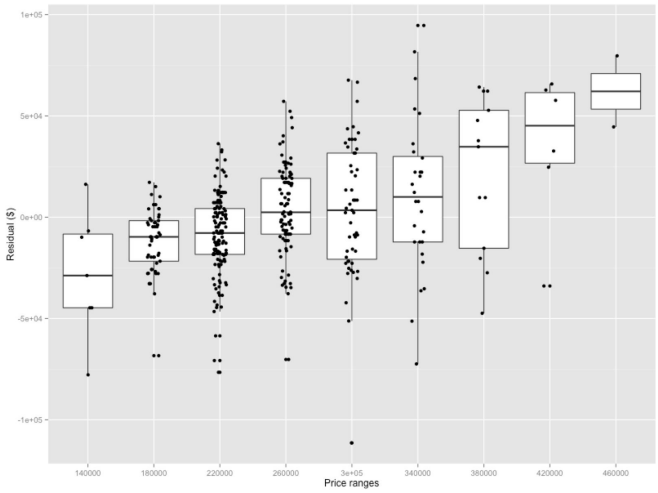
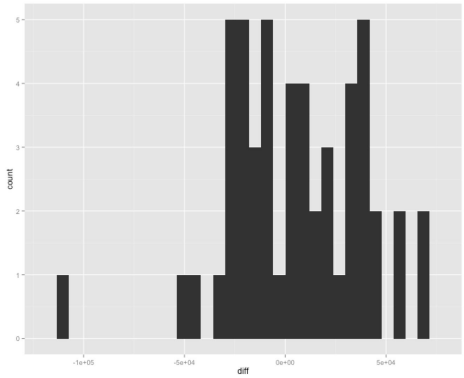


Figure 6: Histogram to Show Assymetry in Residuals for Price ranges $\$300,000$ to $\$400000$



than previous classification methods, which gave at most a 49% accuracy score. However, we wanted to evaluate our model with more granularity this model by examining its performance on each of the buckets that we had used for classification. First, as we can see by the residual plot, Figure 4, our model does well in the middle-to-low price ranges of the houses, but does poorly on expensive houses. Furthermore, the boxplot, Figure 5, shows that the median for the residuals on the price ranges $\$180,000 - 300,000$ is close to zero, which also happens to be the price ranges most common in our training set. This suggests that with more samples for more extreme values, our model could do better. The boxplot also allows us to examine the variance of the residuals for each price range and we can see that for the most common price ranges the variance is much lower than for other ranges. Overall it is also interesting to note that for the highest two price brackets our model is precise but inaccurate, namely: it is offset by a systematic prediction error. Therefore, an easy fix to our model would be to consistently add about $\$50,000$ to our prediction in these bucket ranges.

Although we wanted to also find confidence intervals on the mean of each of the buckets, we can see by Figure 6 that even on the buckets with many samples the distribution of residuals could not be well defined by a normal distribution. Once more, Figure 6, shows that our model predicts lower values than it should. In fact, we can see generally how as the model predicts on more expensive houses our residuals trend higher. We also see overpricing on the first two buckets, which implies a more optimal i.e. flatter set of thetas could do better. Yet, the overpricing is less extreme than the underpricing, which is indicative that perhaps a nonlinear function would model these trends on extreme values more precisely.

Another model which has great potential in improving prediction due to this method modeling feature dependencies is Random Forest Regressors. Preliminary research showed that this model has high potential with an MSPE of 0.5% and 80% R-squared value. Further research on the varying factors of Random Forest Regressors could improve the models performance.

References

[1] De Oliveira, Luke. In person interview. May 14, 2016.

[2] Shirani-Mehr, Houshmand. In person interview. June 1, 2016.