Semester Project - House Prices: Advanced Regression Techniques

Rishikesh Pandey[1]*, Rohit Surve[2]**

Abstract
This article explains our approach at trying to solve one of the competitions listed on Kaggle. The problem deals with predicting the sale price of houses in Ames, Iowa. The data given consists of a total of 80 attributes (including the sale price). In this paper, we will describe how we used the 79 attributes of the training set to come up with a prediction of the sale price. In the upcoming sections, we will be describing our analysis of data, and the algorithms we used and why we chose them. For predictions, we have used random forests, regression techniques and Xgboost.

[1] *Computer Science, School of Informatics and Computing, Indiana University, Bloomington, IN, USA*
[2] *Computer Science, School of Informatics and Computing, Indiana University, Bloomington, IN, USA*
***Corresponding author**: ripandey@umail.iu.edu
****Corresponding author**: rohsurve@umail.iu.edu

Contents

Introduction

The problem we are dealing with in this paper is that of using regression (and other) techniques to come up with a prediction of an attribute. We have been a total of 79 attributes that describe a house's condition, and the aim is to observe a pattern in these attributes' values and based on that predict the sales price of more houses.

The problem has its roots in people trying to come up with the price tag of properties for hundreds of years. The price for a piece of land almost always depends on the land's condition, the surrounding area and many other factors. Often, the price predicted for a house comes out to be highly inaccurate. Hence, we used data mining techniques to use the values of these attributes to predict the sale price of a property. This problem hold significance, since if one can come up with an efficient data mining technique to predict house prices accurately, it will greatly reduce the aberrations observed in prices of not only houses, but also a lot of other areas.

In our approach, we have used the 79 variables available to us and analyzed every one of them. We had to convert many of these variables containing categorical data into numerical equivalents, so they fit into mathematical expressions. We removed a few of the attributes that showed very little variance. For the variables with highly skewed data, we had to transform them to the natural logarithms of their values. We also evaluated the importance of variables based on their correlation with the sale price of the training data.

Though our approach is not perfect, we are confident that the results yielded by our data analysis gives reasonably good results.

1. Background

In this section, we shall introduce some of the technical terms used throughout the paper. We shall also be visiting some of these terms again when we use them later in the paper.

1. **Regression**
   Regression refers to the study of the relationship that two or more variables exhibit. The simplest form of regression is linear regression, where one attribute can be interpolated from another. A more complex form of regression is multiple regression, where we predict a single variable by observing multiple variables.
2. **Random Forests**
   Random Forest is a technique used for both regression and classification, which works by constructing multiple decision trees for a problem, and bases its output upon this.

3. **Clustering**
   Clustering is a technique often employed in Data Mining that aims at classifying data elements into classes such that the most similar elements fall in the same category. For the purposes of this paper, we have used *k*-Means clustering, which is a special case of clustering that uses Floyd's algorithm.
4. **Cross Validation**
   Cross Validation is a technique used for predicting how efficient a model is, by testing its results on different sample training sets.
5. **Feature Engineering**
   Feature Engineering refers to analyzing the data, its types, its (range of) values, and modifying features' values to better suit a model. We have used this extensively in our model.

2. Problem Evaluation

As of now, several methods (and Kaggle kernels) exist for this problem. Most of these proposed solutions use Random Forests, Regression models and XGBoost. Regression techniques attempt at expressing (and hence calculating) the sale price as an equation of the form

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} + \varepsilon \quad (1)$$

for i = 1, 2, ... n.
The second approach, Random Forests, can be explained thus. A Random Forest is an ensemble of Decision Trees. For a classifier $h_k(X)$, we can define the parameters to be $\theta_k = (\theta_{k1}, \theta_{k2}, ...\theta_{kp})$. These parameters also include the layout of the tree. The same can also be written as $h_k(X) = h(x|\Theta_k)$, which describes the Random Forest classifier.

As of now, both of these techniques have been heavily used for the present problem. These techniques achieve reasonably good results. Our aim in this paper is to see if these results can be improved with feature engineering.

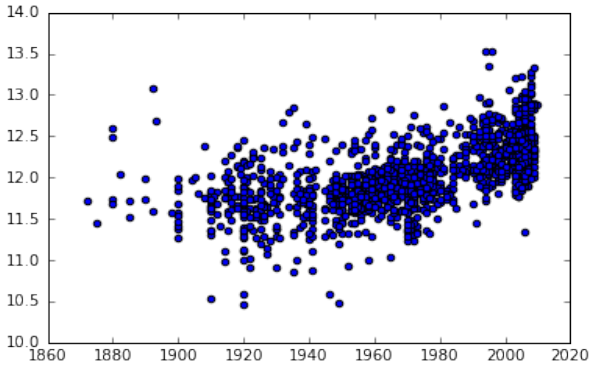
3. Data Analysis

The data in this problem contains a total of 79 variables, both categorical and continuous. We played around with these a lot, and noted the results these had on our data.

There were a lot of categorical variables (like MSSubClass) for which we narrowed down the number of categories the exhibited. For example, for MSSubClass, we replaced values of houses with 1 and 1.5 dwelling types with 1, 2 and 2.5 with 3 and the rest with 0. We adopted this approach for a number of other categorical variables. In some places, we simplified the distribution of categories, based on the variable. For example, we substituted the values of the variable MSZoning with 1 or 0 simply based on whether or not the house was located in a residential area.

There were some sets of variables whose data showed an equal or equivalent distribution (such as Condition1 and Condition2). In some of these cases, we only used one of the variables, since using both seemed redundant.

Figure 1. Distribution of YearBuilt with log of SalePrice



For the YearBuilt variables, we observed the correlation of the years with SalePrice. We show this in the graph below.

Our aim here was to observe what periods of time showed a significant amount of deviation in SalePrice. Based on that, we substituted the YearBuilt variable's values with one of four categorical values. We tried to do something similar for other variables pertaining to years (like YearRemodAdd), but there wasn't any distribution of vales in other variables that would make each period of time unique. Hence, in those, we simply substituted years in a certain interval by a number. We compared our results before and after doing this, and this approach certainly seems to help.

For the Neighborhood variables, we wanted to come up with an approach to group the most similar localities together. For this, we gave each neighborhood a label, which would later be used as distance metrics for the k-Means algorithm. We tried to allot these numbers in such a way that the localities with the highest number of houses received integer labels that were closer to one another than comparatively sparsely populated localities.

We had to drop some of the variables like Utilities that contained almost zero deviation in their distribution. There were some continuous variables like SalePrice that contained highly skewed data. For these variables we used the natural logarithm of their values, to correct the error.

For missing values in continuous data, we used replacing the values with their mean, their median, and lastly, we used Pandas *interploate()* method. We compared the results achieved from all three approaches, and found the third method to work the best.

We also performed one-hot encoding using *sklearn* to convert categorical features into sparse matrices that could be fed into our algorithms.

4. Experiments and Results

Below are the details of the tools we used for the purpose of this experiment:

1. **Language**: Python

2. **Language version**: 3.4.3

3. **IDE**: Enthought Canopy, Kaggle's code editor

4. **Libraries**: Pandas, Numpy, scikit-learn, Matplotlib, XGBoost

5. **OS**: Windows 10

Once we were done cleaning the data, we decided to go with some of the most popular machine learning techniques used for prediction. We used Ridge Regression, Random Forests and XGBoost.

4.1 Ridge Regression

We began our experiment with the most straightforward algorithm available to us - regression. We performed Ridge Regression, with multiple values of *alpha*. We started with the value 1.0 for *alpha*. Through multiple cross validation on sets of 100, we finally arrived at a value of 0.00099 for *alpha*. We verified this by running the Ridge Regression model obtained through this *alpha* on the train set, and calculating the Root Mean Squared Error, which verified that this value of *alpha* was indeed pretty good for our purposes. Our first Kaggle submission was based on the results obtained from this simple classifier. This attained a score of 0.15919 on Kaggle.

4.2 Random Forest Regression

Our next approach was to deploy Random Forests. Once again, we were not sure of how to configure the model. We started with the most intuitive choices (for example, keeping 100 *n_estimators*). This performed better than the results obtained from Ridge Regression, but not significantly better. So we fine tuned the model further. Through multiple runs and cross validation, we finally arrived at the value 500 for *n_estimators*, which improved our results further. Once again, we verified the efficiency of this model by testing it on the training data and calculating the Root Mean Squared Error.

4.3 XGBoost

For our next approach, we decided to try and further improve our results through XGBoost. However, in our case, prediction with XGBoost gave a considerably worse outcome than either of the two approaches we have mentioned above. Hence we disposed of the results.

4.4 Final Results

For our final submission, we averaged the results obtained from the first and second methods. This simple approach gave our score a boost of 0.019 on Kaggle. As of now, we are trying to come up with a better alternative to merge the results obtained from two different models, and compare it with the result obtained from the averaging approach.

5. Results

All three approaches mentioned by us above performed decently. We scored Kaggle score of 0.16792 with using just Ridge Regression, 0.15919 with using just Random Forest Regression, and 0.14328 by using the averaging technique. We are confident that we obtained satisfactory results without having to tweak our models too much because of the extensive feature analysis we performed prior to using our algorithms. Through our efforts, we managed to bring down the number of variables in play from 79 to 52. We are positive that this had a role to play in achieving these results despite having used simple models.

We would also like to mention here that the results we obtained from XGBoost were far from satisfactory. This could be due to two reasons:

1. We could not fine tune the XGBoost model well enough.

2. We did not use XGBoost for evaluating feature importance, something XGBoost is extensively used for. This if done successfully might have had some positive impact on the results.

6. Summary and Conclusions

To summarize, we would like to cite our results.

1. We showed that even on such a large dataset with a large number of features, simple algorithms like Ridge Regression can be made to work if the features have been sufficiently worked with.

2. We showed that combining the results obtained from popular algorithms like Random Forest Regression with the results obtained from simple algorithms like Ridge Regression can significantly affect the results in a positive way.

3. We showed that without fine tuning a model through adjusting its parameters, even a widely successful model like XGBoost might fail to give satisfactory results.

Thus, through our project, we have hopefully established the importance of feature engineering. We compared our results with results from many other Kaggle teams that have used much more complicated models than ours apparently without having conducted enough data analysis first. We scored better than quite a few of these.

Also, we are positive that combining results obtained from two or more models should definitely help make score better. To improve our score further, we are trying to first obtain acceptable results through XGBoost, and then merge together the results obtained from several different models.

Note: The code for this project can be found here

7. Future Work

As mentioned previously, there is scope for some future work in our project. For starters, we are not satisfied with our technique to cluster neighborhoods (mainly because *k*-Means is not usually used for one dimensional data), and think that something better can be used for this purpose.

Of course, there are a lot more models that we haven't tried yet, that can be used for prediction.

We also think that our method to group years into four sections based on the distribution of SalePrice in a period of time can be improved upon.

There is always XGBoost, that we are currently trying to tweak to suit our purposes. We are confident that if done right, XGBoost can dramatically improve our results.

We are also looking for other ways to merge data from two different models than averaging them.

Acknowledgments

We would like to acknowledge the guidance, ideas and support of Prof. Mehmet Dalkilic throughout this course, without which this project wouldn't be possible. We would also like to acknowledge the helping hand of all our AIs, for which we are thankful.

Appendices

Packages/libraries used:

1. Pandas - `http://pandas.pydata.org/`
2. Numpy - `http://www.numpy.org/`
3. Matplotlib - `http://matplotlib.org/`
4. Scikit-learn - `http://scikit-learn.org/`
5. XGBoost - `http://xgboost.readthedocs.io/en/latest/model.html`

We used the Ames housing dataset available here for this project. We accessed this data for the first time at November 25th, 2016, 5:00 PM.

References

1. Dean De Cock *Ames, Iowa: Alternative to the Boston Housing Data as an End of Semester Regression Project* Journal of Statistics Education Volume 19, Number 3(2011)
2. www.kaggle.com/humananalog/house-prices-advanced-regression-techniques/xgboost-lasso/code
3. http://www.ccs.neu.edu/home/mirek/classes/2012-S-CS6220/Slides/Lecture2-ClassificationPrediction.pdf
4. Jiawei Han, Micheline Kamber, Jian Pei *Data Mining Concepts and Techniques*