Artificial Neural Network, XgBoost, Random Forest, Support Vector Regression

Zillow Zestimate – Neural Network For Minimizing Log Errors

"Using Artificial Neural Networks to minimize prediction volatility" (Volak Sin, August 18, 2017)

I Introduction

Zillow is one of the largest real estate informational websites. One of its most highly sought-after services is providing real estate valuations for potential home buyers and sellers. These valuations are often the only means of price comparison for many consumers, often superseding the importance of traditional real estate "comps" given by real estate agents. According to Zillow, their real valuations, called Zestimates have been as accurate as to be within 5% of the actual transaction pricing.

This is a considerable feat seeing that when they first started out, their initial valuations produced an accuracy rate within 14% of the actual sales price. Even an error rate of 5% could still mean a fairly large variance from the actual sales price of the home. For a home whose initial valuation estimate of $300,000, this could mean a difference from the selling price of almost $15,000. These inaccuracies of 5% have actually led to lawsuits brought against the company by both the buyers and sellers of homes. In both cases, the alleged parties argued that the inaccurate price predictions caused economic harm to the respective positions. It is of strategic importance for Zillow to constantly try to improve their prediction algorithms not only for the sake of reliability and trust among their user base but to also avoid potentially expensive lawsuits.

II Data Limitations

The data provided by Zillow for their competition seems to have all features related to the physicality of the home. Some of these features are of course vital to the actual pricing of the home. The difficulty in predicting the target features comes from trying to find the sources of volatility and whether that volatility is direction specific, i.e., over valuations versus under valuations.

The biggest limitation to the current dataset relates to real estate's "golden rule" of "location, location, location." Though the latitude and longitude coordinates are provided, we can only make limited location valuation assessment; whether a clump of homes has better or worse log error rates. The assessment of a property's location value will require other features such as quality of schools, crime rate, parks, etc.

The challenge summary mentions an error of 5%. After running our machine learning algorithms, we may or may not be able to achieve this 5% at this stage of the data challenge. There is mention of a second round where presumably these other features relating to location are released.

III Data Exploration

The properties dataset consists of 2985217 samples indexed by parcelid. The index provides little information since it is described in the data dictionary as "Unique identifier for parcels (lots)" with no reference to how the unique identifier was determined. The training dataset consisting of 90811 was also indexed by parcelid. This dataset contains our response variable, the logError estimates, that we are trying to minimize.

Key Highlights from the Data exploration part:

- There are 57 distinct features (not including parcelid) in the properties dataset and 2 distinct features in the training data set. Since the total number of combined features is quite high, we do need to worry about false positives created by the curse of dimensionality. We will use techniques to reduce the dimensionality when possible.
- 53 of the features are float 64, 5 are object, 1 date/time and 1 integer 64. We may have to encode the 5 object features since most ML algorithms cannot correctly process these features. The ways of encoding and the difference among them will be discussed later on.
- There are many missing values in the dataset. Some features have 100% of their values missing. This fact shows that Zillow's data may not be as highly processed and friendly to use as most other Zillow competitions.
- The current strategy to deal with the missing values is to average the values that are currently available and substitute them into the missing fields, imputing the values. Though this solution holds for most of the values missing in the features, certain features cannot take advantage of this strategy. These features often have values that are descriptive in nature and cannot be mathematically averaged. For these values, we will try other strategies.
- Some of the float 64 features are really categorical. Features such as Architectural type and Air conditioning type are really labels of categories. We will need to process these feature correctly in order for the Machine Learning algorithms to work.
- The target feature has been preemptively converted to the log normal of the error between the actual selling price and the Zillow estimate of the selling price (Zestimate)
- The target feature is normally distributed but also has some outlier values.
- Some of the features contain extreme outliers. Coincidently, many of these features that have these extreme outliers are correlated. This correlation makes perfect sense since some of the features are in essence physical components of the others. For example, increasing the number of bedrooms and bathrooms automatically means you increase the square footage and cost of the home. If you have an extreme number for either of these features, it will create outlier values for the latter features. We use a correlation matrix to show all the related features. Unfortunately, this will mean that we have multicollinearity in our dataset. This will lessen the predictive power of the linear

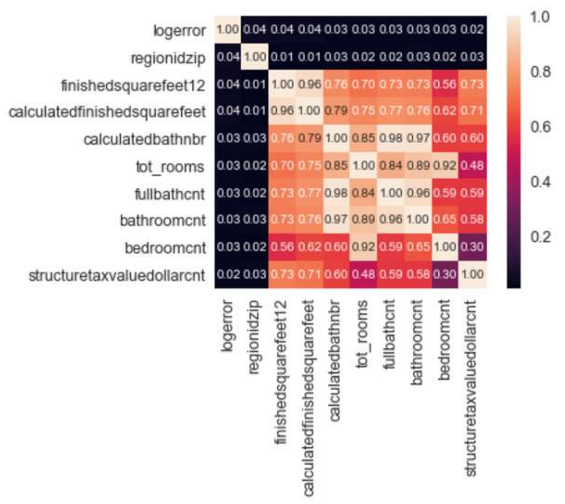
IV Evaluation Method

Since this project serves as the first major capstone, we will use Root Mean Square error as a means of evaluating the different models. Since we are focused on optimizing for model performance, we will not focus so much on model interpretability. Thus, we did not emphasize on various Feature Selection techniques such as the Filter Method, Wrapper Method or Embedded method. We did a preliminary analysis with the Filter method but avoided the Wrapper and Embedded method altogether. We instead put our emphasis on Feature Extraction techniques; Principal Component Analysis.

V Feature Selection

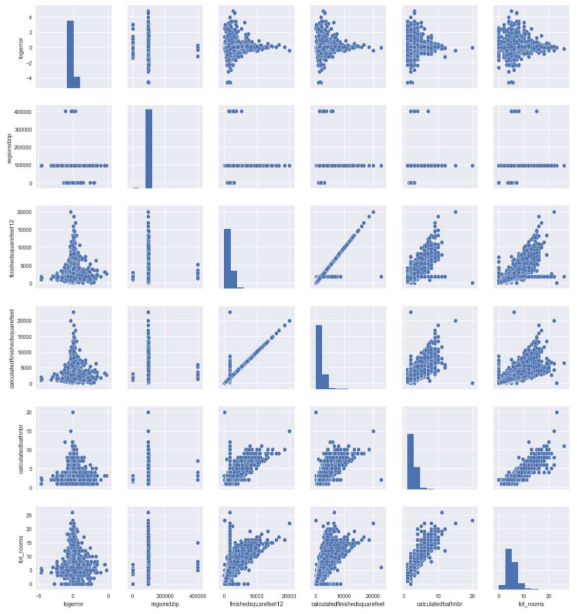
Correlation Matrix

From the result of the correlation matrix below, we see that the "regionzip" had the highest correlation with the target feature. We also noticed that "calculated finished square feet " and "finished square feet12" also made it to the top ten correlated features. Unfortunately, those two features are also correlated with each other. This makes intuitive sense, and therefore one of the variables will need to be eliminated from our final model. The R-squared of running only those significant features produced using the Ordinary Least Squares and Correlation matrix produced a model with an R-squared of less than 1%. Even though the first data set provided by Zillow excluded some important features, we were still surprised to get R-Squared of less than 1%.



OLS Regressions

Even though we are aware that the features were not independent, we ran Ordinary Least Square regression with the target feature just to get a baseline of which features were important. There were 10 categorical variables that were statistically significant with a p-value less than 0.05 and 8 continuous variables with 8 statistically significant variables. Below is a scatter plot of the top 5 correlated features.



Feature Selection Results

When we ran Ordinary Least squares independently on all the features, we saw that there were a handful of features that seemed to be statistically significant to the model. However, when we took the features most correlated to our response variables we produced a model whose R-squared was less than 1%. Since we are more focused on producing a model that has maximum accuracy, we are not as worried about the interpretability of our model.

VI Feature Extraction and Regression

We originally planned on doing Feature Extraction using Principal Component Analysis (PCA), Linear Discriminant Analysis, and Kernel Principal Component Analysis. Even after setting up a desktop to run computations using the GPU, the desktop was not able to handle the computational needs of the latter two feature extraction techniques, and we were left to run just PCA on our model.

Principal Component Analysis

Before using the various regression models, we had to use a feature extraction technique due to the limited computer power available. After creating dummy variables for our categorical features, we were left with a dataset with over 3000 features. Unlike feature selections techniques, we will not be able to interpret the features extracted. We will leverage the features extracted using PCA and feed it into our regression models. We will vary the number of Principal components needed; using 10, 20, and 50 respectively. We will then feed these number of component into our various regression models to see if choosing a number of principal components has any significant effect on the model performance.

Multiple Linear Regression

We use the multiple linear regression as our baseline model to compare the other regression models. Ideally, we would like to feed our entire list of features into our multiple regression model, but due to computational constraints, we will only feed it the principal components we extracted earlier.

Support Vector Regression

We will use a Support Vector regression to check for both linear and nonlinear models. We will use a linear kernel to check for linearity in our model. For nonlinear models, we will use a Gaussian kernel. Before we can use our Support Vector Regression, we have to make sure our features are all scaled. Some regression classes include a feature scaling algorithm, like the multiple regression above, but our Support Vector Regression does not. We had to fit transform both the feature and response variables before we produced our predictions.

Decision Tree Regression

The Decision Tree regression is built on nonlinear models. Like the classification Decision Tree, it uses entropy as a metric of determining whether certain features provide added information to the model. The Decision Tree builds regression model by breaking down a dataset into smaller and smaller subsets. At the same time, an associated decision tree is incrementally developed. The overall structure of a decision tree includes a decision node and a leaf node. The decision has two or more branches that each represent the value of the feature tested while the leaf node represents the decision target. Decision trees can handle both numerical and categorical data.

Random Forest Regressions

Similar to the Decision Tree, the first step of a Random forest is first to find the feature that gives the most information gain when splitting on a given value of that feature. The second step is to find the next feature that gives the most information gain. Unlike a Decision tree that continuously adds branches by adding features, a Random forest will randomly choose a feature and start another tree with a different combination of features to start the tree. This will help counter the tendency to overfitting. We chose 10 estimators for our Random Forest algorithm.

VII Artificial Neural Network

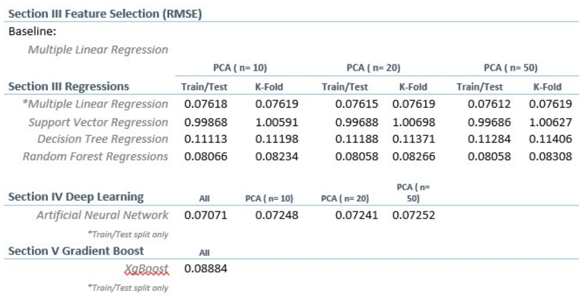
We were able to have access to a computer that had an Nvidia Graphics GeForce 770 graphics card. Even though we can access the GPU to run the neural network, we will standard scale the features variables to help run the computations. For our Neural Network, we ran it with two hidden layers, with 6 nodes in each layer. We chose a batch size of 10 and an epoch of 50. We would have normally chosen an epoch of 100, but we wanted to an optimal solution that could be found with our computational constraints. We ran this Neural

Network multiple times. In the first, we included our entire feature set into the neural network. Later, we first ran principal component analysis and then used those components as input features/nodes for our artificial neural network. To minimize the problem of overfitting, we included a 10% drop out between each hidden layer.

VIII XgBoost

XgBoost is one of the most popular models in machine learning. It is also the most powerful implementation of gradient boosting. One of the major advantages of XgBoost besides having high performance and fast execution speed, you can keep the interpretation of the original problem. This also means you do not need to do feature scaling. We will still do a train and test split to check our model. Even though it is even better to do a K-fold cross validation, we are restricted by our computational power of our desktop.

IX Results



Section III Feature Selection (RMSE)						
Baseline:						
Multiple Linear Regression						
	PCA (n= 10)		PCA (n= 20)		PCA (n= 50)	
Section III Regressions	Train/Test	K-Fold	Train/Test	K-Fold	Train/Test	K-Fold
**Multiple Linear Regression*	0.07618	0.07619	0.07615	0.07619	0.07612	0.07619
Support Vector Regression	0.99868	1.00591	0.99688	1.00698	0.99686	1.00627
Decision Tree Regression	0.11113	0.11198	0.11188	0.11371	0.11284	0.11406
Random Forest Regressions	0.08066	0.08234	0.08058	0.08266	0.08058	0.08308
Section IV Deep Learning	All	PCA (n= 10)	PCA (n= 20)	PCA (n= 50)		
Artificial Neural Network	0.07071	0.07248	0.07241	0.07252		
**Train/Test split only*						
Section V Gradient Boost	All					
XgBoost	0.08884					
**Train/Test split only*						

Results Summary

From the results above, we see that the Artificial Neural Network performed the best out of all the algorithms. Since we were able to use the GPU to run the Neural Network, we were able to use our entire feature set in our algorithm. Unfortunately, the regression algorithms utilize the less powerful CPU to process the data, and as a result, we could not run our models with all the features without running into "memory errors." For a better comparison, we ran all the algorithms after we did a Principal Component Analysis, varying the number of principal components from 10 to 50. After running the models through this procedure, we see that Neural Networks produced the lowest root mean square error. The second best result came from a Multivariate Linear Regression. This could mean that the model is more linear than nonlinear. As expected, the Random Forest outperformed the Decision Tree. This makes sense since a Random Forest requires multiple iterations of Decision Trees.

X Conclusions

The response variable is the log of the errors between the actual selling price and the estimated selling price of the home. This is different than the more traditional model that predicts the price of a home and compares it to the actual selling price. Under this model assumption, variables such as the size of the home and number of bedrooms have a significant correlation with the sales price. Since we are dealing with the log of the errors, the question now becomes which variable, size of home or number of bedrooms, makes our guess more accurate. This log error model has no natural intuition. As such we must treat each feature provided as a potentially important model feature.

We are concerned about our result since the Root Mean Square Error is an order of magnitude higher than the mean log errors. We can account for this discrepancy since the root mean square error will cumulatively add both the positive and negative values together while the mean of log errors would have the positive and negative values negating each other, thereby reducing the magnitude.

We were not surprised that a deep learning model produced the best result. We were a little surprised that a simple multivariate linear regression produced the second best result.

XI Improving The Model

The simplest way to improve our model is to run the algorithms on more powerful computers. This would improve the results of our Artificial Neural Network since we could utilize Grid Search to tune our Hyper Parameters. We could also increase the batch size, number of hidden layers, number of nodes in each hidden layer, as well as the number of epochs ran. A more powerful computer would also allow try other feature extraction algorithms such as Linear Discriminant Analysis and Kernel PCA. Even though the results may be similar, it helps to verify the potency of the models further.

For the first round of the competition, Zillow provided a sample of their full dataset. We can assume more features would be provided in the second round. Common real estate related variables such as average household income, quality of schools, and crime rate were not included in the first dataset. These variables should improve our current models and reduce our root mean square error of our predictions even further

XII Alternative Strategies

One alternative strategy for this project could be to take the absolute value of the log error as a new response variable. Though this method helps simplify things by focusing on just the magnitude of the errors, it would probably not increase our prediction accuracy in the actual use case. We could narrow the absolute error rate, but because we do not know the direction of the error, positive or negative, we could make our predictions twice as bad if we had the incorrect sign.

We could have also spent more time engineering different features. For this project, we created two new features, “age of home” and “total number of rooms.” These new features probably did contribute to the accuracy of the model, but since the baseline model included both new features, we will not be able to detect the importance of adding these new features.