CSE 847: Machine Learning—Final Report

An Exploration and Implementation of Automated Valuation Models to Learn and Predict the Value of Real Estate

Mick Langford
langfo37@msu.edu

Mike Lingg
linggmic@egr.msu.edu

Jordi Lucero
luceroj2@msu.edu

March 24, 2017

1 Problem Description

Automated Valuation Models (AVM) have become increasingly popular as the real estate market has embraced the World Wide Web as a source of accurate, up to the minute data.[3] Banks have also shown great interest in using AVMs to help mitigate fraud by human appraisal.[4] Our goal is to explore various machine learning techniques to implement an AVM and predict the true value of a house based on features commonly found on real estate listings. Our data will be drawn from various data sets, including from the Nashville, TN housing market, using a dataset posted on Kaggle[1].

We will begin by exploring linear and logistic regression models that take into account physical attributes of each house and location. Further work will be performed exploring nonlinear models, such as deep learning with neural networks and decision trees, which can be compared and contrasted. Additional work was performed to explore missing feature estimation and other data preprocessing.

2 Related Work

An obvious and popular example is Zillow's proprietary Zestimate®. Zillow uses a closed source AVM that takes into account special features of the home, location, and market conditions. Zillow admits to using features such as physical attributes, tax assessments, and prior transactions. Zillow claims to have data on 110 million homes and estimates on approximately 100 million homes.[5]

Relevant papers include the doctoral dissertation of Lowrance which explores and compares various linear models on housing data for the Los Angeles County.[6] Park and Bae explore machine learning algorithms such as C4.5, RIPPER, Naive Bayesian, and AdaBoost.[7] Bin performed a study that estimates a hedonic price function using a semi-parametric regression.[8] This may be particularly useful for real estate listings that are incomplete or for data that is entered erroneously. Bourassa et al. consider the spatial dependence of house prices, which is intuitively an important factor.[9][10] Kauko et al. research neural network models to help investigate segmentation in the housing market of Helsinki, Finland.[11] Azadeh et al. present an algorithm based on fuzzy linear regression and a fuzzy cognitive map to handle uncertainty in the housing market and improve the analysis of housing price fluctuations.[12] Fan et al. introduce a decision tree approach for modeling and predicting house prices.[13]

3 Project Data

For this project we are working with multiple data sources pulled from real housing sales data.

Nashville Housing Data The Nashville housing data set is a list of home sales in the Nashville, Tennessee area, provided by Kaggle[1]. This data set includes 29 fields of data for 56635 entries. However, nearly half of the entries have gaps in information, which will have to be accounted for. We further augmented this data set by using an geocoding service provided by the United State Census Bureau to add the zip code, latitude, and longitude for entries where a match could be found.

King County Housing Data The King County housing data set is a list of home sales in the King County, Washington area, provided by Kaggle[2]. This data set includes 20 fields of data for 21614 entries, with none of the entries missing any data.

Advanced Regression Techniques Data The Advanced Regression Techniques data is a list of home sales, provided by Kaggle. This data includes 79 features of housing data for 1460 homes. The data has no gaps, except for some N/A data.

Grand Rapids Data The Grand Rapids data is a list of home sales in the Grand Rapids area, provided by the real estate listing service Redfin. This data includes 16 fields for 9646 entries. The data set also has gaps in information for about half of the entries.

The different data sets provide a variety of input to test machine learning techniques against. Data with more features should provide more accurate results, if the additional features provide useful information for the prediction.

4 Preprocessing Data

For pre-processing the data, we will be normalizing the data set, by dividing each feature value by the difference between that feature's maximum and minimum values. Another problem being considered is how to properly manage fields containing categorical values. Our approach will be to treat a field with n categories as n binary features, indicating whether that entry is of the associated category or not. Finally we provide the ability to fill in holes in the input data using the feature mean, closest value, by way of Euclidean distance, or closest mean.

	Preprocessing Results
No Preprocessing	1.2487
Value Normalizing	0.0499
Normalize and Binary Features	0.0012

Table 1: Linear regression preprocesssing results.

Table 1 shows the results of data preprocessing with a simple closed form linear regression, using the Advanced Regression Techniques Data. First we look at no preprocessing, though this row does normalize the sale prices so the mean squared error can be compared between the results. Second we look at the change with normalizing the features and home prices. Third we look at splitting category fields into binary features, in addition to normalizing. Each additional method of preprocessing improves the mean squared error significantly. As a result of this analysis we preprocessing all of our data by normalizing the values and splitting category values into binary features.

	Hole Filling
None	NaN
Mean	1.6e-3
Closest Value	1.2e-3
Closest Mean	1.1e-3

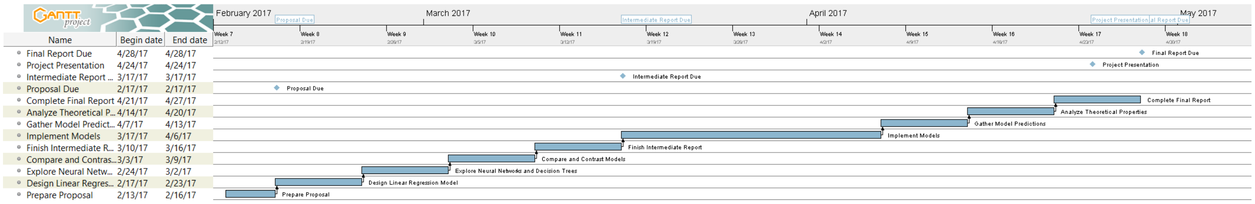
Table 2: Linear regression preprocesssing hole filling results.

Additionally we look at the best method of filling in holes in the input data. The Advanced Regression Techniques Data has no holes so we look at the Grand Rapids Data from Redfin for hole filling. Table **??** shows that with no holes filled, the basic closed form linear regression cannot be solved with Matlab. Filling via the mean value of the missing feature works but does not produce the best error. In closest value replacement we look at the set of homes with no missing features and fill the missing values from the closest neighbor, using euclidean distance. Closest value improves on mean value replacement but the best method is closest mean where the value is taken from the average of the 10 closest homes, using the same method as closest value. We will continue to use the closest mean method throughout the project for filling in holes in the data.

During the project we noticed that the mean squared error was not producing expected results between data sets. Models that appeared to produce a good match were resulting in higher error than models that appeared to produce weaker results. It was observed that the data sets producing lower error were ones with a large number of lower value houses and a few high value corner cases. This issue was addressed by filtering out any homes with price above 1 million to eliminate some corner cases. This upper limit was applied before filling in data holes, so the larger data does not impact the calculations for hole filling. Additionally when normalizing house prices, the same maximum value was used for all data sets. With these changes, the mean squared error is producing consistent results between the data sets.

5 Models

Figure 1 shows our initial project milestones and timeline to completing them.



Name	Begin date	End date
Final Report Due	4/28/17	4/28/17
Project Presentation	4/24/17	4/24/17
Intermediate Report ...	3/17/17	3/17/17
Proposal Due	2/17/17	2/17/17
Complete Final Report	4/21/17	4/27/17
Analyze Theoretical P...	4/14/17	4/20/17
Gather Model Predict...	4/7/17	4/13/17
Implement Models	3/17/17	4/6/17
Finish Intermediate R...	3/10/17	3/16/17
Compare and Contras...	3/3/17	3/9/17
Explore Neural Netw...	2/24/17	3/2/17
Design Linear Regres...	2/17/17	2/23/17
Prepare Proposal	2/13/17	2/16/17

Figure 1: Gantt Chart showing our project milestones and expected completion dates.

5.1 Linear Regression

For linear regression we are using a closed form ridge regression. We tested standard and stochastic gradient descent but these methods did not produce a significant improvement in processing time for the data we are using, and do not improve on the prediction error.

For the ridge regression regularization value we developed a simple automated approach to identify the ideal regularization value. We start with a test regularization value of 1 and perform five fold cross validation on the training data to determine the mean square error with this test value. Then we adjust the test value by 0.1 in the positive direction, in this case moving from 1 to 1.1, and repeat the cross validation. If the error decreases, we continue to adjust the test value in the same way. If the error increases, we reverse the sign of the test value adjustment and divide it by 10. This is like a simplistic gradient descent, but when we overshoot the minimum we reverse direction and lower the step size. This continues until the error between two cross validations matches within 1e-7.

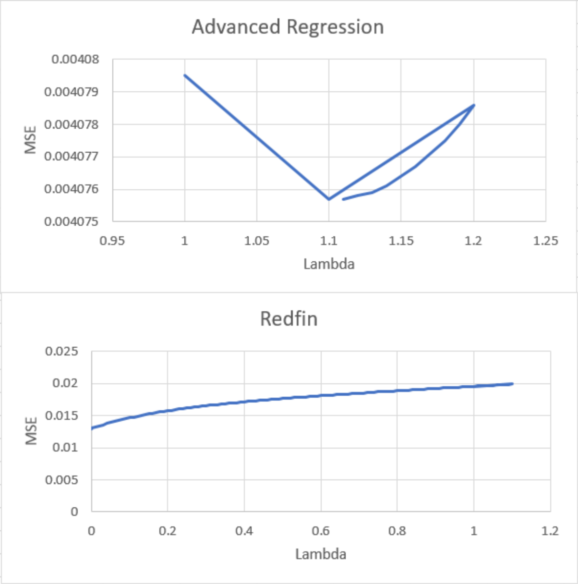


Figure 2: Linear regression cross training.

Figure 2 shows a sample of our cross training method on the Advanced Regression and Redfin data sets. The advanced regression data initially shows a reduced error when moving from 1 to 1.1 so it increases to 1.2. At 1.2 the error goes back up again so it reverses direction and moves back in increments of 0.01, until it reaches a stable point. This shows a place where the algorithm works but could possibly have been made more efficient, but still works for our current purposes. The Redfin data starts at a regularization value of 1 but sees an increase of error at 1.1 so decreases at a lower rate. What can be seen in the data, but is not visible in the graph, is around 0.013 lambda the regularization search reverse direction, an dincreases precision, several times to find the minimum solution.

Table 3 shows the mean square error of linear regression for each dataset. As expected the advanced regression techniques dataset provides the least error as this data provides the most complete features for each house. Nashville performs the weakest as there are very large gaps in the data, which must be filled in by preprocessing. The interesting result is between Redfin and King's county data. King's county has the same number of features as redfin and more homes in the data set. Preprocessing the data shows that the redfin features have more categories and the redfin house prices are concentrated more at the lower end, whereas the King's county features have few categories and have prices fairly evenly distributed over the normalized range. As a result, redfin has more information in the features and the prices are easier to predict, which appears to be what produces the lower error.

	Mean Squared Error
Nashville	22.1e-03
Advanced Regression	1.0e-03
Redfin	6.3e-03
King's County	13.1e-03

Table 3: Observed performance from linear regression model.

We also tried two ensemble methods with linear regression. The first method divides the features into groups of 4, performs linear regression on each group and adds the group results together. Linear regression is then performed on the summation of the groups. The second method picks random groups of 10 features with 100 groups and then an ensemble is produced in the same method as the groups of 4 in the previous method. Additionally each ensemble method was tested with PCA feature reduction providing 25%, 50%, 75% and 100% of features. This method was tested against all of the data sets except for the Nashville data as an SVM on the Nashville data exceeded matlab's available memory.

Figure 3 shows the results of the ensemble methods. The advanced regression techniques and King's county data performed close to the pure linear regression. The redfin data set showed a slight improvement over linear regression.

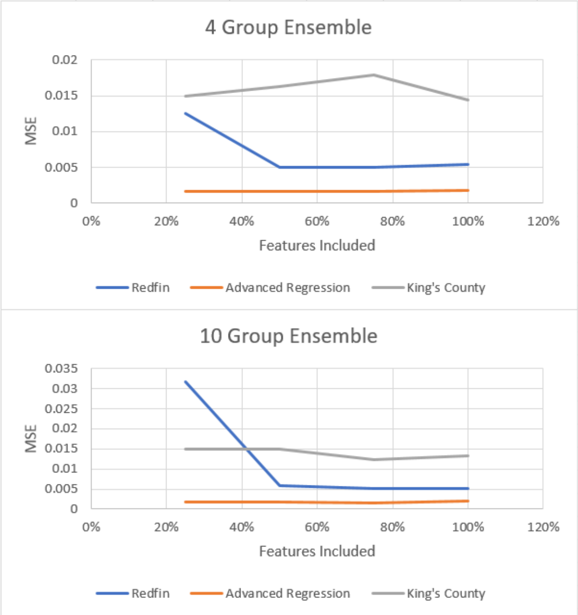


Figure 3: Linear regression ensemble results.

5.2 Classification

We are also approaching this problem as a classification problem for our linear regression and neural network models. The range of sale prices for the entire data set will be partitioned into k classes, each class can include the same range of values, or be variable in size to accommodate ranges with similar input data. The classification solutions will then be fitted to a training data set, and finally predict each entry in the test data set by fitting it into one of the k classes.

5.3 Logistic Regression

During initial logistic regression development, we have used two different approaches to perform classification.

The first approach attempt to classify which of the k class ranges the house most likely falls in based on input data. This model is composed of a matrix of independent regression models. It is trained for each input data point by setting the Y value corresponding with the class the house value falls under to 1 and all others to 0. The current training method uses stochastic gradient descent with a reducing step size. Once trained, the logistic regression class with the most confident result for the given input data is the one under which we classify a house.

The second approach also uses a matrix of independent regression models but instead the results classify if the house is worth more than the lower value of the class k of each model. The training method only differes from the first approach by setting the Y value corresponding with the class equal to 1 if the value of the current house is higher than the lower value of the class. Then classification is performed by summing up the range represented by each class with a confidence higher than 0.5.

Table 4 shows a comparison using the Advanced Regression Techniques data of the linear regression vs the two logistic regression approaches. The table shows that the second approach for logistic regression, determining if the house is worth more than each class, works better than the first approach, determining which class the house would fall under.

	Mean Squared Error
Linear Regression Model	1.0e-03
Logistic Regression Model 2	4.8e-03
Logistic Regression Model 1	13e-03

Table 4: Observed performance from logistic regression model.

Figure 4 shows the accuracy is best for a small number of classes, except for Advanced Regression Techniques. For Advanced Regression Techniques, 5 classes causes most houses to fall right on a boundary between classes, which causes poor performance. Advanced Regression Techniques performs better for 10 classes, where most of the house values happens to fall into classes more evenly. While accuracy decreases with an increase in classes, the mean squared error difference between the lowest value of the predicted class and the true value decreases. This is caused by predictions being for a narrower range with more classes, increasing the precision.

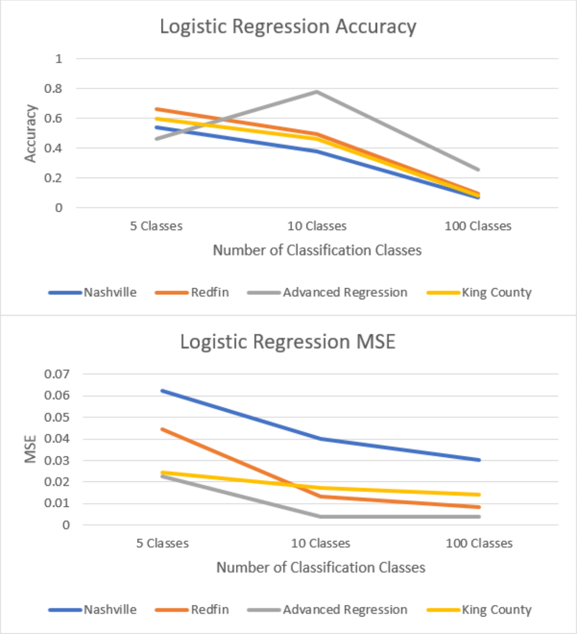


Figure 4: MSE and accuracy for Logistic Regression.

Finally we implemented an ensemble that performed linear regression on top of the logistic results. This ensemble used the 100 class logistic regression test predictions for training and testing. Similar to previous runs, the Nashville data was skipped as it produces too large of a matrix for matlab. As shown in table 5, the ensemble provides at least a small improvement on each data set as compared to the base logistic regression.

	Original Logistic	Ensemble
Redfin	8.5e-03	8.1e-03
Advanced Regression	4.5e-03	1.7e-03
King's County	14.4e-03	13.2e-03

Table 5: Observed performance from logistic regression ensemble.

5.4 Decision Tree

For our Decision Tree approach, we used a Classification Decision Tree and a Regression Decision Tree. For the Classification Tree, since the data was not already classified, it was split into k equally sized classes where each class represents a range of prices. The average value for every feature column is also calculated and used to subsitute for any missing data points in the observations. A classification model was built for the KC data set, Redfin data set, and Nashville data set. For each data set, initially a classification tree model was built using crossfold validation with 10 folds using 90% of the data for training. This model used the default parameters for tree size, which are maximum number of splits of n-1, minimum leaf size of 1, and minimum parent size of 1.

Another classification model was built by using 10 fold cross validation and generating trees for many possible parameters and choosing the model that provides the best training MSE, again using 90% of the data for training. The results comparing these two models and their training and testing MSEs for the Nashville data set is shown in Table 6 below. The MSE indicates the averae squared distance between the predicted class and the actual class, where the classes are represented by a range of integers from 1 to k. In the results below, the number of classes that are being classified is 8.

	Mean Squared Error
CTree Train Error	0.2890
CTree Test Error	0.5247
Optimized CTree Train Error	0.5052
Optimized CTree Test Error	0.4350

Table 6: Observed performance from Classification Tree models.

For the Classification Tree, the optimized parameters result in much higher testing error, but also a much better testing error. This seems to indicate that default parameters with crossfold validation result in slight overfitting compared to crossfold validation with the optimized parameters.

Regression Tree models were generated following a similar pattern, creating one 10 fold cross validation Regression Tree model for each data set using default parameters and creating another Regression Tree model by varying the tree parameters to find the best MSE training error. Table 7 below shows a table of the MSE results for the Regression Tree approach on the Nashville data set.

	Mean Squared Error
RTree Train Error	5.4526e+10
RTree Test Error	6.8929e+10
Optimized RTree Train Error	5.2162e+10
Optimized RTree Test Error	6.5855e+10

Table 7: Observed performance from Regression Tree models.

The Regression models perform similarly, with slightly better results using the optimized parameters for the regression tree compared to the default parameters. Our goal for the Decision Tree approach is to take the best approach and vary the feature selection to create a Random Tree Forest. Then use proximity trees to better estimate any missing values.

5.5 Neural Network

Our initial approach for implementing a neural network learning model is to use a simple feed-forward network with three hidden fully-connected layers. Each data point will have d features extracted from the data. Each feature in the input will have its value sent as input to each neuron in the first layer of the network, where each neuron has its own associated weight and activation function. Each neuron's output from the first layer will be sent as input to each neuron in the second layer, which will also have its own weight values and activation function. Finally, the output from each neuron in the second layer will be sent as input to each neuron in the third layer, again having its own weights and activation function. The final hidden layer will have k outputs, representing which class the data point falls into.

Our current method is to use a sigmoid function as the activation function for each layer, with k neurons in the third layer, $2k$ neurons in the second layer, and $4k$ neurons in the third layer. Further experimentation is being conducted with varying sizes and architectures.

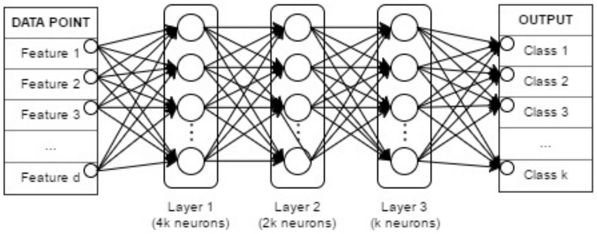


Figure 5: Architecture of our network.

The input data is randomly shuffled and split into training and testing data sets, with a ratio of $9:1$. The model is trained for 50 epochs, using an Adaptive Gradient Descent (Adagrad) algorithm to fit and optimize weights throughout the model, with an initial learning rate $\eta = 0.1$. Techniques are being considered that will decay the learning rate with relation to the loss calculated for each epoch.

Displayed in Table 8 and Figures 6 and 7 are the results of running the data sets through the current prototype network. Two different trials were run on each data set, with the first trial learning and predicting against 5 classes of target values, and the second trial against 10 classes of target values. Each target class represents an equally sized partition of the data's target values. Predictions of a target class indicate that the given data point's sale price will fall within the boundaries that define that class's partition.

	5 classes		10 classes	
	Train	Test	Train	Test
Nashville	61.53%	60.41%	39.80%	38.71%
King County	51.92%	51.92%	30.90%	29.94%
Grand Rapids	57.05%	56.34%	33.74%	32.75%

Table 8: Observed performance from neural network.

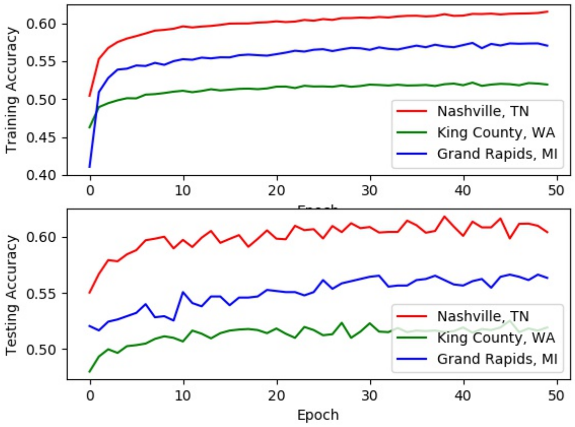


Figure 6: Results for classifying into 5 classes.

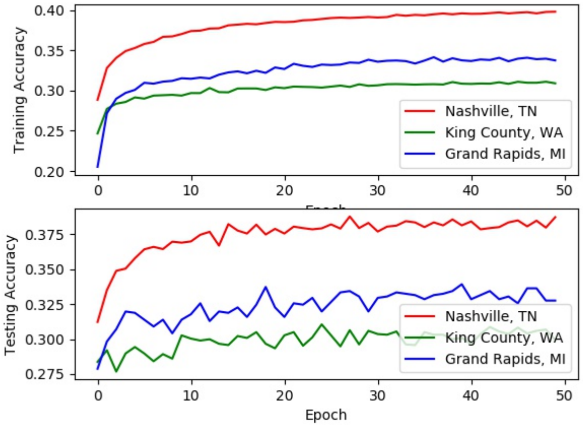


Figure 7: Results for classifying into 10 classes.

The results show that the network performs better when classifying data points into a smaller number of target classes. Our goal from this point forward is to attempt to improve accuracy by investigating further improvements on preprocessing the input data, experimenting with different hyper-parameters for the network, and experimenting with varying network architectures. When accuracy improves, the goal with then be to increase the number of target classes in order to provide a more valuable estimation of the sales price of any given data point.

6 Conclusion

We set out to look at how different machine learning approaches work when predicting or classifying home selling values. We implmented models that included linear regression, logistic regression, decision trees, neural networks and ensemble methods. The methods themselves provided some interesting comparisons but one consistent thing we found with all models is the data the model is built on is one of the most important considerations.

First processing the data for use by the models is extremely important. This includes cleaning up the data to a format the models handle well, such as splitting categorical features up into a binary category for each category in the feature. Additionaly filling holes in the data can significantly improve a model's ability to predict or classify data.

Ultimately while there are things that can be done to get all of the information available from the data, a machine learning model is limited by how much information is actually in the data. Table 9 shows the results across the various models. In all cases the Advanced Regression Techniques data performs the best. This data is feature complete and has the largest number of features. We believe this data was created specifically for developing machine learning models. Of the remaining models, Nashville performs the worst as this data has the biggest holes in feature values, limiting the amount of actual information provided by the data.

	Linear Regression	Linear Ensemble	Logistic Regression	Logistic Ensemble	Decision Tree 1?	Decis
Nashville	22.1e-03	N/A	N/A	N/A	?	
Advanced Regression	1.8e-03	4.5e-03	1.7e-03	?	?	
Redfin	6.3e-03	5.1e-03	8.5e-03	8.1e-03	?	
King County	13.1e-03	13.1e-03	14.4e-03	13.2e-03	?	

Table 9: Combined Results.

References

[1] *Nashville Housing Data: Home value data for the booming Nashville Market* Retrieved from `https://www.kaggle.com/tmthyjames/nashville-housing-data/`

[2] *House Sales in King County, USA: Predict house price using regression* Retrieved from `https://www.kaggle.com/harlfoxem/housesalesprediction`

[3] *Data-driven property valuations: the real deal?* Retrieved from `http://blog.kaggle.com/2010/06/21/data-inc-are-avms-soothsayers-or-the-real-deal/`

[4] Schroeder, Steve. *Fighting Fraud: A combination of collateral assessment and AVMs can maximize mortgage-fraud management* Retrieved from `http://www.scotsmanguide.com/Residential/Articles/2005/10/Fighting-Fraud/`

[5] *What is a Zestimate? Zillow's Home Value Forecast.* Retrieved from `http://www.zillow.com/zestimate/`

[6] Lowrance, R. E. (2015). *Predicting the Market Value of Single-Family Residential Real Estate* (Doctoral Dissertation). New York University. Retrieved from `http://gradworks.umi.com/36/85/3685886.html`

[7] Park, B., & Bae, J. K. (2015). *Using machine learning algorithms for housing price prediction: The case of Fairfax County, Virginia housing data.* Expert Systems with Applications, 42(6), 2928-2934. doi:10.1016/j.eswa.2014.11.040

[8] Bin, O. (2004). *A prediction comparison of housing sales prices by parametric versus semi-parametric regressions.* Journal of Housing Economics, 13(1), 68-84. doi:10.1016/j.jhe.2004.01.001

[9] Bourassa, S. C., Cantoni, E., & Hoesli, M. (2010). *Predicting House Prices with Spatial Dependence: Impacts of Alternative Submarket Definitions.* SSRN Electronic Journal. doi:10.2139/ssrn.1090147

[10] Bourassa, S. C., Cantoni, E., & Hoesli, M. (2007). *Spatial Dependence, Housing Submarkets, and House Prices.* SSRN Electronic Journal. doi:10.2139/ssrn.771867

[11] Kauko, T., Hooimeijer, P., & Hakfoort, J. (2002). *Capturing Housing Market Segmentation: An Alternative Approach based on Neural Network Modelling.* Housing Studies, 17(6), 875-894. doi:10.1080/02673030215999

[12] Azadeh, A., Ziaei, B., & Moghaddam, M. (2012). *A hybrid fuzzy regression-fuzzy cognitive map algorithm for forecasting and optimization of housing market fluctuations.* Expert Systems with Applications, 39(1), 298-315. doi:10.1016/j.eswa.2011.07.020

[13] Fan, G., Ong, S. E., & Koh, H. C. (2006). *Determinants of House Price: A Decision Tree Approach.* Urban Studies, 43(12), 2301-2315. doi:10.1080/00420980600990928