Forecasting Housing Prices

Matthew Salvati
500681878
Computer Science
Ryerson University
Toronto, Canada
msalvati@ryerson.ca

Shah Moiz Alam
500578704
Computer Science
Ryerson University
Toronto, Canada
moiz.alam@ryerson.ca

***Abstract*—This paper aims to understand the chosen forecasting models, and more specifically, their implementation for predicting the pricing of houses. A dataset was chosen, and a multitude of different strategies were attempted in order to discover the most accurate results. This paper observes the impact of feature pruning, normalization of data, and also the unpruned data's affect on the accuracy. The findings from the paper concluded that certain, more obvious selection of data did not result in the highest accuracy. The entire set with some pre-processing was able to do product results with sufficient accuracy. However, normalization of the data increased the accuracy of all models the most with models such as Gradient Boost and Neural Network (Multi-layer perceptron) able to yield results close to 90%.**

***Index Terms*—forecasting, regression, linear, ridge, normalization, gradient boost, lasso, random forest, ElasticNet, neural network, outliers, normalization**

I. Introduction

The prediction of housing prices is a prime example of a useful implementation of machine learning and regression. Regression models are very useful using continuous data and through a dataset, with a plethora of attributes and features, there is the potential to find a correlation in the data which can be used to train a model to estimate the most likely price of a house given only the features. Assuming the data is correct and actually has correlation this can easily be applied to any housing dataset and can have a multitude of uses.

This document will explain the process of how various machine learning algorithms were used to predict housing prices based on features from the dataset. The dataset used in this implementation contains house sales for King County in the United States of America. It includes the homes sold from May 2014 to May 2015. Based on the type of the data, it was deemed appropriate to use regression models to predict housing prices because the data in this set was continuous.

This demonstration implements the use of linear regression, ridge regression, random forests, lasso regression, ElasticNet regression, and gradient boosting regression. A lot of linear regressors were used and the reason for that is cost and the data being used has continuous values. Linear regressors are quick to implement and calculate and provide relatively accurate results within a very short period of time. The biggest issue that was a concern during the implementation was that linear regressors are prone to over fitting. The data had to be clean and caution must be taken in order to avoid fitting the data too heavily resulting in inaccurate results. That is the reason for the implementation of multiple regressors also. For one, they insure that they are all able to produce reliable results, and another reason was simply to analyze the data according to each type of regressor and understand why certain ones are better than others. This paper will go more in depth about each type of regressor used for the forecasting and why it may or may not have been an ideal design choice for a forecasting algorithm in this case.

Ultimately, this paper will establish the best approach for determining the prices of the houses in King's County. Multiple linear regressors were used on the datasets and the product of each method and the resulting best approach will be discussed throughout the rest of the rest of this report.

II. Related Work

A. Predicting Housing Prices with Machine Learning

As stated previously, predicting housing prices is an ideal problem for an algorithm to effectively solve. The report, [1] , focuses on predicting local

housing prices. The difference with this and with many problems in machine learning is the processing of information and the algorithm used to solve the problem. This report focused heavily on pruning and pre-processing the data as one would assume it would result in a higher accuracy.

III. Research Methodology

The dataset used in for this implementation relied on regression models because of its continuous structure. The structure used to create this forecasting model was to use the data how it was presented and then apply various techniques on the data to see how it improves or worsens the accuracy of prediction. To begin the process of training our model, the dataset was split into training and testing sets. The ratio chosen for this project was 80:20, with 80% of the data is used for training and 20% is used for testing. A larger section of the data was used for our training to make sure that under fitting was prevented, while also making sure that the set is large enough to yield statistically sound results. To get a strong understanding of the data, a graph plotting the distribution of housing prices was used. This helped gain an of the density of housing prices in the dataset. Following will be a

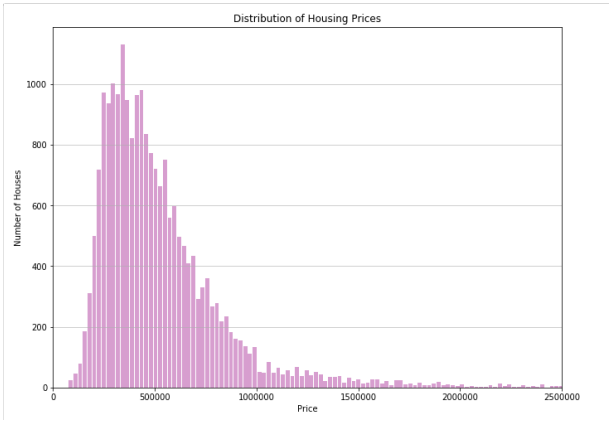

Fig. 1. Distribution of Housing Prices.

list of procedures and algorithm considered and used during the forecasting implementation. The general idea of these algorithm and the reason for use will be discussed as well.

A. Pre-Processing

Data pre-processing is a very important part in Machine Learning. In any real world dataset, there will be values that may be null, or need to be pruned to make sure all the data that is being used is strictly relevant. Often time, data is pre-processed to remove values that may possibly hinder the performance of the learning model. The date label in this dataset was trimmed

To perform an efficient and relevant pre-process in this machine learning implementation, correlation between all the labels was produced and only the highest values were kept. In this case, they were; prices, bedroom, bathrooms, sqft living, sqft above, view and sqft basement.

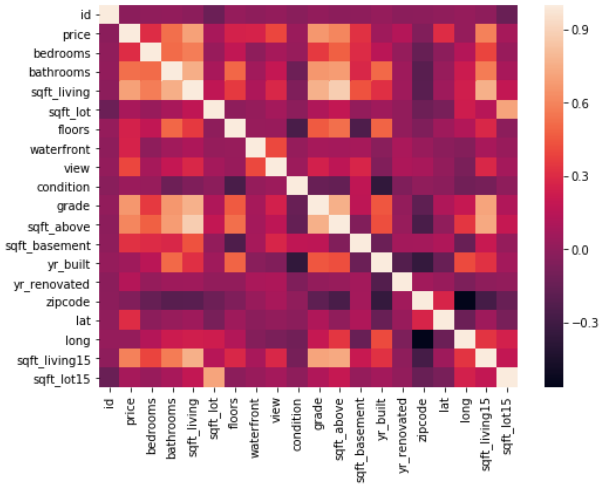

Fig. 2. Correlation between the data.

As the image suggest, labels that were highly correlated to price were the ones that were kept, and the ones below a threshold of 0.3 were discarded. There were some relatively lower correlated labels that were kept in the data because there was an assumption made that those labels should be correlated. For example, bedrooms has a correlation (0.308) with price that is just meeting the threshold, however, it would make sense that the number of bedrooms would influence the price of the house.

B. Normalization

As part of the pre-processing stage of machine learning, normalization was applied to the dataset to ensure that the numeric values in the columns are changed to a common scale, without distorting the weights of the values. Normalization is not a necessary step in all machine learning implementations, however, the dataset used here had a wide range in its features.

C. Linear Regression

The idea with linear regression [2] is that the algorithm attempts to create a linear model that fits a line of best fit through all of the data. Many would consider, and would be correct to assume that linear regression is the simplest form of regression. Linear regression observes one dependant variable and one or more independent variable in order to make an assumption on what the model should look like. The main practical use for linear regression is for forecasting or to assist in better error reduction. It is able to easily and quickly train a predictive model that will likely be accurate if the data is somewhat correlated to the dependant variable. Mathematically, the simplest linear regression model would be represented by the following equation:

$$Y_i = \beta_0 + \beta_1 X_i + \epsilon_1$$

Where:

- Y represents the dependent variable.
- X represents the independent variable.
- b is the slope of the actual line of regression.
- e represents the random error term. This is the difference between the regressions predicted value and the actual value.
- b+bX as a function represents the entirety of the linear component.

There are many forms of linear regression but simple linear regression uses the least squares method to fitting. This type of fitting where the line of best fit would look for the minimum distance between all the samples. Linear regression is one of the simplest models and a staple when forecasting so it must be considered when trying to determine the prices for houses. The greatest issue with this is that outliers will heavily impact the accuracy of the model which is why data correlation is so important. As seen in Fig. 1, the line of best fit is somewhat in the center which is the minimum distance from all of those points to the line.

D. Ridge Regression

A large dilemma with many regression algorithms is overfitting and underfitting. Ridge regression [2] helps to mitigate some of this problem by regularizing to solve the issue of overfitting. A penalty is added to some of the features and data in order to ensure that the model doesn't try to fit all of the data too heavily. The penalty and regularizer looks like this:

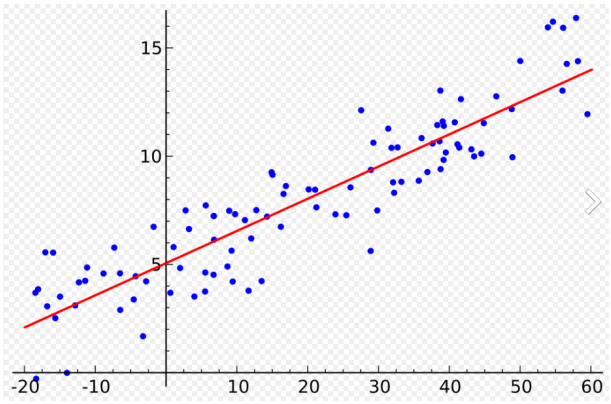

Fig. 3. Linear Regression using least squares to minimize the distance from the line to each point.

$$\lambda \sum \beta^2$$

The sklearn library imports ridge regression using L2 regularization. L2 regularization is where the model tries to minimize the dependency by adding a penalty to the loss function. The difference between simple linear regression and ridge regression is this added restraint.

E. Random Forest Regression

Random forest regression [2] is a prime example of ensemble learning. Ensemble learning is when multiple learning algorithms are used in combination with each other in order to provide a more accurate result from the total information. Random forest constructs multiple decision trees for the data and collectively makes a mean prediction for each of the trees. The benefit of the ensemble learning and the multiple decision trees is that it will not result in overfitting like having only 1 decision tree would. All the trees use their averages to improve the accuracy and better control overfitting of the model. The default number of trees using the sklearn library is 10.

F. Lasso Regression

Lasso regression [2] is very similar to ridge regression however its primary difference is that a different penalty function or regularizer is used. The equation aims to shrink the data values towards a more centered point. Where ridge regression uses an L2 regularizer, lasso, and the sklearn library used,

uses an L1 regularizer. Instead of squaring the value it just takes the absolute value.

$$\lambda \sum |\beta|$$

G. Gradient Boost Regression

Gradient boost [2] is a strong regressor that was implemented in this forecasting approach. Gradient boosting builds trees as the regressor and corrects errors based on each iteration. It is a reliable regressor and will likely result in the best accuracy for forecasting housing prices which is the reason it was chosen. Many related projects refrain from using this regressor but it should be a key staple for many forecasting projects.

Gradient boost starts by making a single leaf of a tree which represents a starting price. This estimated price is around the average value of all the prices so the leaf is not just an arbitrary number. From there, gradient boost builds a tree upwards. When the model has errors, during the next iteration it corrects those errors by adding weights to incorrect predictions making them not as relevant, and vice-versa for correct predictions. If the next iteration did not correctly correct errors from the last tree it will not have a large impact on the further improvement of the model. Although time consuming, this regressor is an ideal candidate for forecasting and that will be seen later in this report.

H. ElasticNet Regression

ElasticNet regression [2] is often a large improvement to a lot of other models. ElasticNet is a combination of both ridge and lasso regression, but is excels at dealing with data that has highly correlated independent variables. It uses a combination between L1 and L2 regularization to add a penalty:

$$\lambda_1 \sum \beta^2 + \lambda_2 \sum |\beta|$$

The hybrid of lasso and ridge regression takes the regularizers of both regression models. Lasso focuses on taking one independent variables and elimination the others to train the model, where ridge regression shrinks all of the correlated variables together. ElasticNet combines these penalties by grouping and shrinking each of the parameters of the correlated variables. Although likely more costly, this is an ideal implementation for this problem as that have highly correlated independent variables. There are many attributes associated with the housing prices and the if the features are correlated with each other, excluding the price, then ElasticNet would have a higher score. This means the opposite if the independent variables are not too highly correlated with one another.

I. Multi-Layer Perceptron Regressor

Finally, the last attempted implementation was a multi-layer perceptron [2] which is a neural network. This is another regression implementation that similar projects do not attempt to implement. The idea with this implementing a neural network is that they have a very strong learning model and learning rate so they may be able to produce a high accuracy with not too complex of a network.

To put things it somewhat simpler terms, neural network is a bunch of neurons corresponding to a value. Each neuron has an activation which is the certainty that it is correct. These neurons pass forward their values to a hidden layer and these values are affected by weights depending on how correct the network thinks that neuron is. This neural network uses an optimized version of stochastic gradient descent which is a loss function that progressively updates often using the L2 regularizer. The network continuously forward propagates and back propagates adjusting each neuron and the weights connecting them until it finds an accuracy that is acceptable. The sklearn library has a parameter called ”early_ stopping” which allows for the neural network to stop training once it sees that the validation score is no longer improving. Neural networks do get more complex but for the sake of simplicity it is being using in the forecasting of housing prices because of the ability to update the weights itself and the optimal learning rate they bring.

IV. Findings

A. Pruned Dataset

Using the pruned dataset of labels that were either only highly correlated to price or assumed to be important based on common knowledge, for example, more bathroom and bedrooms may increase the price of the house) the predictions for each model turned out to be quite inaccurate.

The underlying relations showed us that the initial assumption of including the number of bedrooms and bathrooms to train our model were incorrect. The following heatmap (figure 3.) is a result of the pruned dataset and it clearly displays the lack of correlation between most of the labels that were initially chosen. The models used on the pruned dataset

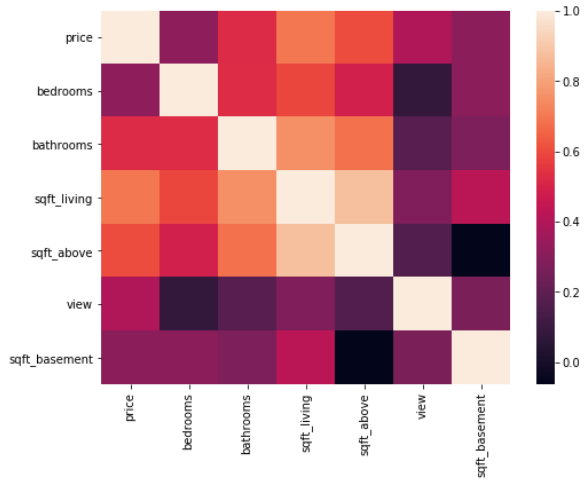

Fig. 4. Correlation between the data.

were Linear Regression, Ridge Regression, Random Forest Regression, Lasso Regression, Gradient Boost Regression and ElasticNet Regression. The accuracy for each model was; 56.6%, 56.6%, 54.0%, 56.6%, 56.5% and 54.9%, respectively. The low accuracy score of each model clearly matches what is observed from the heatmap in Figure 3. Figure 4.

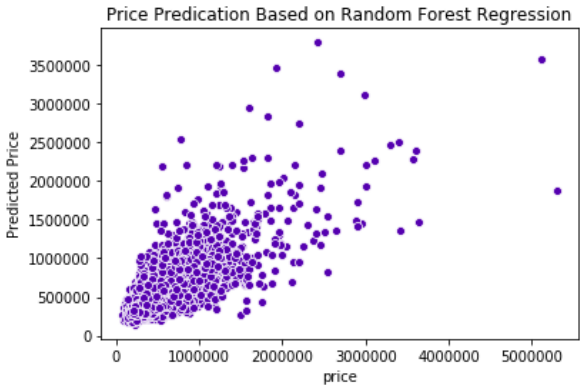

Fig. 5. Random Forest Regression on pruned dataset.

and Figure 5. show the best predicted price on the pruned dataset. The discussion on the comparison of these predicted prices made with different data (Entire dataset and Normalized dataset) will be done below. Based on the accuracy score and heatmap to prove this, it was obvious that methodology and technique used to prune the data only hindered the performance of the models.

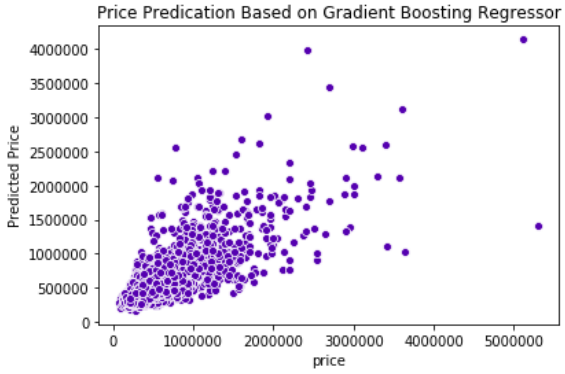

Fig. 6. Gradient Boost Regression on pruned dataset.

Therefore, the entire dataset was used in the next iteration of the project. There was still some data that was pre-processed and outliers had been removed to make sure the data was uniform and meaningful. This technique resulted in significantly higher scores for each model used earlier. The results from the entire dataset proved to be more accurate drastically.

B. Entire Dataset

Fig. 7. Linear Regression on entire dataset.

Following the implementation of the models on the pruned data, the models were then trained on the entire dataset. The data was still pruned for outliers however it included all of the features instead of just the 6 that were included on the pruned data. The pruned data did not result in as high of an accuracy as one would hope for so the entire dataset

was used to see how all the features collectively were able to predict the pricing. There were drastic improvements from the pruned data with some regression models improving the accuracy by over 30%. Every regressor, although some not as much as others, did see an improvement and that was a significant finding because one would think that the common attributes of a house, like the ones chosen in the pruned data, would be the only impactful ones but in reality it seems that all the data cumulatively is what produces the best results.

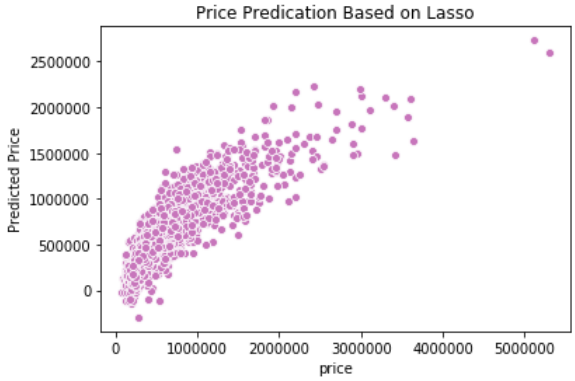

Fig. 8. Lasso Regression on entire dataset.

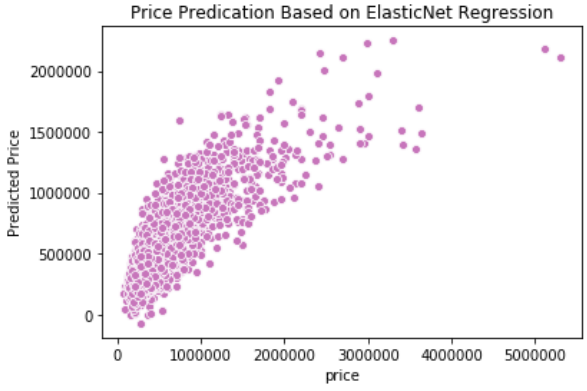

Fig. 9. ElasticNet Regression on entire dataset.

The notebook shows the entire set of graphs for both the pruned data and the whole dataset but by looking and Figure 6 through 8, it's much more clear to see that the predictions vs the actual data more closely resemble something linear. The pruned data is much more wide meaning the predictions were somewhat random, where the predictions for the entirety of the data was far more precise. The same regressors were used for this dataset as well which were all the regressors discussed above, that includes Linear Regression, Ridge Regression, Random Forest Regression, Lasso Regression, Gradient Boost Regression, and ElasticNet Regression. The resulting accuracies for each of these models were; 72.6%, 64.5%, 88.1%, 72.6%, 90.0%, and 63.6% respectively.

C. Normalized Dataset

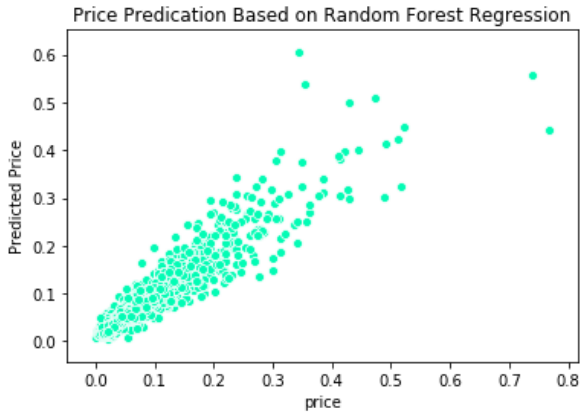

Fig. 10. Random Forest Regression on normalized dataset.

Building on top of the results observed on the entire set, the goal was to find a way to improve on those results. This came in the shape of normalizing the dataset to make sure all ranges were brought down to a common scale without disrupting the influence of the weights of those values. Although the results did not improve drastically, as they did when the entire dataset was used over the pruned data.

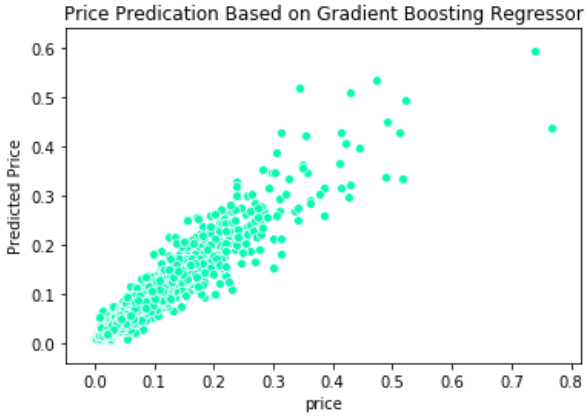

Fig. 11. Gradient Boost Regression on normalized dataset.

There were still improvement seen across the models. The models with the highest accuracy score

for the predicted prices were Gradient Boost and Random Forest Regression with their score being 90.1% and 88.3%, respectively. These scores were significantly higher than the initial implementation of these models on the pruned dataset. Since the data was normalized, there were more models used that showed a different relationship amongst the labels. Neural Networks were used to confirm the high accuracy produced by Gradient Boost regression. The multilayer perceptron was able to produce results with an accuracy of 87.4% and confirmed the relationship between prices and predicted_prices.

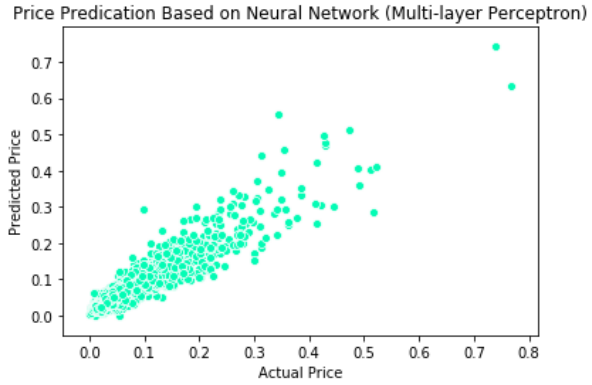

Fig. 12. NeuralNet Multilayer Perceptron Regressor on normalized dataset.

Normalizing the dataset allowed the neural net to learn optimal parameters for each node in the hidden layer more quickly and efficiently. Figure 13. shows the final results for each regressor on the dataset.

Regressor	Pruned Data (%)	Entire Data (%)	Normalized Data (%)
Linear	56.6	72.5	72.6
Ridge	56.6	64.5	N/A
Random Forest	54	88.1	88.3
Lasso	56.6	72.6	72.5
Gradient Boost	56.8	89.9	90.1
Elastic Net	54.9	63.6	N/A
Neural Net (Multi layer)	N/A	N/A	87.4

Fig. 13. Predictions for datasets.

V. Conclusion

The initial ideology behind the data pruning was to take the attributes that were the popular features people would look for when considering a house. Looking at the data, the correlation of these features to the price were not very high so using only a few features for the predictions resulted in a very poor accuracy score. When returning to using the entire dataset with all the features the accuracy score saw a drastic increase which means that although some of the correlation isn't the strongest, all of the data in combination is important for the actual prediction of the pricing. Outlier removal was still a necessary addition as it got rid of unrealistic data and some data that would cause the model to attempt to overfit. The removal of just a few outliers in the end resulted in increasing accuracy from 2% to 5% on some of the models.

The models chosen for this machine learning problem were all regression problem and the reason for this was because the dataset being used was continuous. After using the entire normalized dataset, the findings proved that the best models for this dataset were Gradient Boost, Random Forest and Neural Networks (Multilayer perceptron). The multi-layer perceptron in this implementation used a large hidden-layer size. This was reflective in the score as it allowed the model to adjust the weights of each of the neurons in this layer. A layer with a good amount of neurons allows for the neural net to make a large amount of slight changes instead of drastically changing the weights of a few connections. Random Forest was also able to perform well because it tries to minimize variance, since it's an ensemble technique. By using the bagging technique it was able to train each decision tree on a different sample from the dataset. This allowed the model to eventually combine a series of decision tree together, rather than relying on one decision tree. Random Forest is able to handle missing values and misclassification without training the model on those values, thus maintaining the accuracy of the larger portion of data. Lastly, Gradient boost; the best performing model for this forecasting problem. This was another ensemble learning technique which proved to produce excellent results. Gradient Boost Regression is able to perform better than most models in this project because it uses the boosting technique. It's able to use weak learners and gradually improving them into stronger learners. The weights of the observations that were misclassified or inaccurately classified are increased. Each tree is built on top of the previous tree, thus, each

tree ends up improving. Bias and variance trade-offs plague all machine learning problems and this implementation is no different. It is not shocking to see that the best performing regressors were the ones that minimized variance and bias.

A variety of models were trained to see the impacts of each of them but not all of them performed very well. It seems as though the two that performed the worst were Ridge regression and ElasticNet regression. Lasso regression also performed subpar however it was not terrible. There is a link between Ridge and ElasticNet with is likely the reason that these two regressors had similar scores. ElasticNet uses the same regularizer as Ridge in combination with Lasso so if one of those didn't perform well then the defects would also be apparent with ElasticNet. Lasso could not be performed on normalized data and for that reason neither could ElasticNet, as stated before because it uses the same regularizer, so the effects of normalization could not be visualized or calculated for there two regressors. Ridge did perform significantly better with normalized data however it was still the worst performing regressor. The largest issue with Ridge regression is that the dimensionality reduction may be too heavily impacting the data. This leads to a high bias error which may be why the results are so poor.

Normalizing the data was an ideal adjustment to the data. In practice it helps to normalize the data as it can limit the range of everything between 0 and 1 instead of having absolutely outrages values that could sometimes be considered outliers. This was the best change made to the data and it resulted in all of the models performing better. It did however restrict the ability to use some regression models such as Lasso and ElasticNet, though it did allow for the use of a neural network which performed extremely well. Related works that were done on the forecasting of housing prices did not make use of neural networks but the results show that it is nearly one of the best regressors for this data. The performance of the model was excellent and should be implemented in any similar forecasting implementations or wherever regression is needed because even the cost was relatively low.

After discussing and analyzing the findings from this forecasting problem, it is evident that in any given dataset, the most obvious labels may not be the best features. It is through various machine learning algorithms that one can find the underlying relations data points have between each other that can be used to make accurate predictions. Upon reflection of the dataset, it may also be beneficial to find a better dataset. There were problems encountered at the start of this implementation as the dataset did not have descriptions for certain labels. Furthermore, this data was restricted within the boundaries of Kings County which means that this may not be an accurate predictor for other parts, even within the country. A larger, more diverse dataset could solve this issue and would be able to provide more universal predictions. Overall, the forecasting models used on this dataset proved to be excellent predictors with Gradient Boost, Random Forest and the Multilayer Perceptron having a far greater performance than the simple linear regressors.

REFERENCES

[1] Eric Kim, "Predicting Hose Prices with Machine Learning", Kaggle. 2017.

[2] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011