Using Machine Learning Algorithms to Predict Property Prices

Armon Singh

Introduction:

In a society where investing in properties has become commonplace in the lives of most Americans, it is crucial for homebuyers to be knowledgeable of housing trends before they make a purchase. Historically, however, the housing market has been extremely volatile – a pattern that can be attributed to financial restraints[1], down-payment effects[2], and departures from rational expectations[3]. This unpredictable movement within the real estate sector makes it extremely difficult for people to know the right times to sell or purchase properties. Fluctuating mortgage rates and high prices limit options for prospective buyers[4], which in turn affects the ability of sellers to complete sales, establishing that housing market volatility affects both parties through their indirect relationship. Beyond individual benefits, having an accurate forecaster of real estate prices is important because the housing market reflects overall market conditions and is a reliable indicator of economic health.

To combat the issue of fluctuations within real estate, it is beneficial to investigate the specific factors that most heavily influence prices. I selected a dataset from Kaggle that included properties across New York City, specifically their prices and other characteristics. I expected that each aspect of a property would have some significance in influencing its price, yet I knew it would vary from feature to feature. After preprocessing the dataset, I used various algorithms to visualize my data in different graphs. This provided a preliminary view of how the property prices seemed to correlate with individual features. Finally, I trained different computational models with my preprocessed data and evaluated their accuracies in properly predicting the property prices. After selecting the most accurate model, I used a specific function to determine the most influential feature in forecasting the prices.

Background:

Section I: General History in Predicting Real Estate Prices

Throughout history, there have been numerous studies that delved into predicting real estate prices through different methods. Kashan used linear regression analysis in conjunction with text mining to forecast real estate prices in Dubai, and was able to successfully reduce prediction errors[5]. Del Cacho used a model tree bagged algorithm that produced results with a

15% deviation from the quoted prices of properties[6]. Taking the research one step further, Grether and Mieszkowksi used mathematical models to identify the strongest determinants of property price as structural characteristics, such as the number of stories in a house or the spatial qualities of a garage and other exterior features.

Section 2: King County Research

In recent years, researchers from Seattle University studied rapidly-fluctuating residential real estate trends in King County, Washington. They noticed that from 2008 to 2015, the local housing market experienced a steep crash followed by a sudden surge.

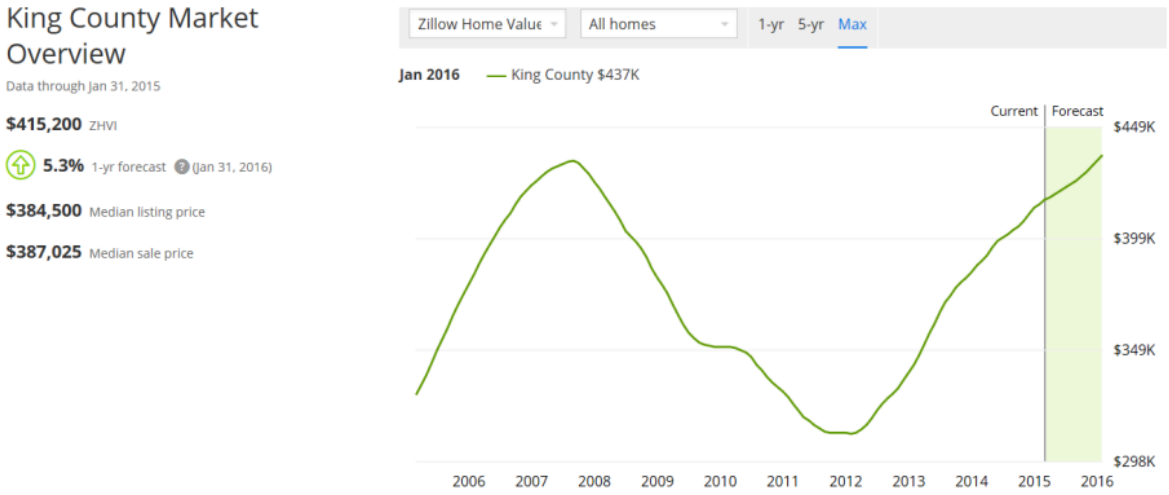


Figure 1. Real Estate Trend in King County per Zillow

Noting that Zillow's Zestimate® algorithm predicted a 5.3% increase in property prices for King's County from 2015 to 2016, the researchers decided to create their own models to predict a drop or rise in real estate values. The Zillow Zestimate® uses market data and statistics in conjunction with its own, proprietary formula to estimate trends. The researchers obtained their dataset from Zillow, which contained real estate data from King County during January 2015.

Table 1. Factors used for Analysis of King County

Data format	**Factor**
Categorical	Direction Prefix Fraction Street Name Street Type

	Zip Code Daylight Basement View Utilization
Numerical	Stories Building Grade Building Grade Variation Square Feet 1st Floor Square Feet Half Floor Square Feet 2nd Floor Square Feet Upper Floor Square Feet Unfinished Full Square Feet Total Living Square Feet Total Basement Finished Basement Grade Square Feet Garage Basement Square Feet Garage Attached Square Feet Open Porch Square Feet Enclosed Porch Square Feet Deck Heat System Heat Source Percent Brick Stone Bedrooms Bath: Half Count Bath: 3 Qtr Count Bath: Full Count Fireplace: Single Story Fireplace: Multiple Story Fireplace: Freestanding Fireplace: Additional Year Built Year Renovated Percent Complete Percent Net Condition Obsolescence Additional Cost Condition

This dataset includes only residential homes, neglecting commercial and office buildings.

The researchers constructed and compared Decision Tree and Neural Network Models through Microsoft SQL Server 2014 – Databases and Analysis Servers. The models were trained on the dataset above. The Microsoft Decision Trees algorithm is a classification and regression tool used in predictive analysis for both discrete and continuous variables. Meanwhile, the

Microsoft Neural Network was composed of three layers of neurons: an input layer, a hidden layer, and an output layer.

Prediction Error after Adjustment	**Decision Trees**	**Neural Networks**
Mean Absolute Error (MAE)	102,968	83,579
Standard Deviation	108,474	73,595

Table 2. Comparison of Prediction Error Results

Using the "Mining Model Prediction" tool within the Microsoft SQL Server, the researchers used their models to predict the prices in King's County for 2013, and compared their results to the actual data. After calculating the mean absolute errors (MAE) and standard deviations for each model in accordance with properties in different price ranges, it was found that the Neural Networks model was more accurate in predicting values for all price ranges, except for the $300,000 - $400,000 range. Their results contradicted those of del Cacho, who found that the Decision Trees model performed better than Neural Networks. This variation may have come from the fact that del Cacho examined properties in Madrid, Spain, a more urban setting, during a different time period, from 1991 to 2001.

Dataset:

My dataset, titled "AirBnB," was procured from Inspirit AI's database. It includes both numerical and categorical variables. In all, there are 48,895 different samples and 16 different features represented in the dataset. Using the is.na().sum() function, I was able to locate a total of 20,141 null values located within my dataset, which I filled with either empty strings or zeroes depending on if the variable was qualitative or quantitative. The dataset contained 10 numerical variables and 6 categorical variables. Of the 6 categorical variables, I converted "neighborhood_group," "neighborhood," and "room type" to numerical values using the Label Encoder. After creating 3 new variables to hold the numerical form of these categorical features, I dropped the original columns that held their qualitative values. I excluded the categorical features of "name" and "host name" from the analysis as it was difficult to convert them to binary values for numerical analysis. After finishing the preprocessing and storing my modified dataset under a new variable, I used the matplotlib and seaborn libraries to visualize my data in scatterplots, histograms, and boxplots.

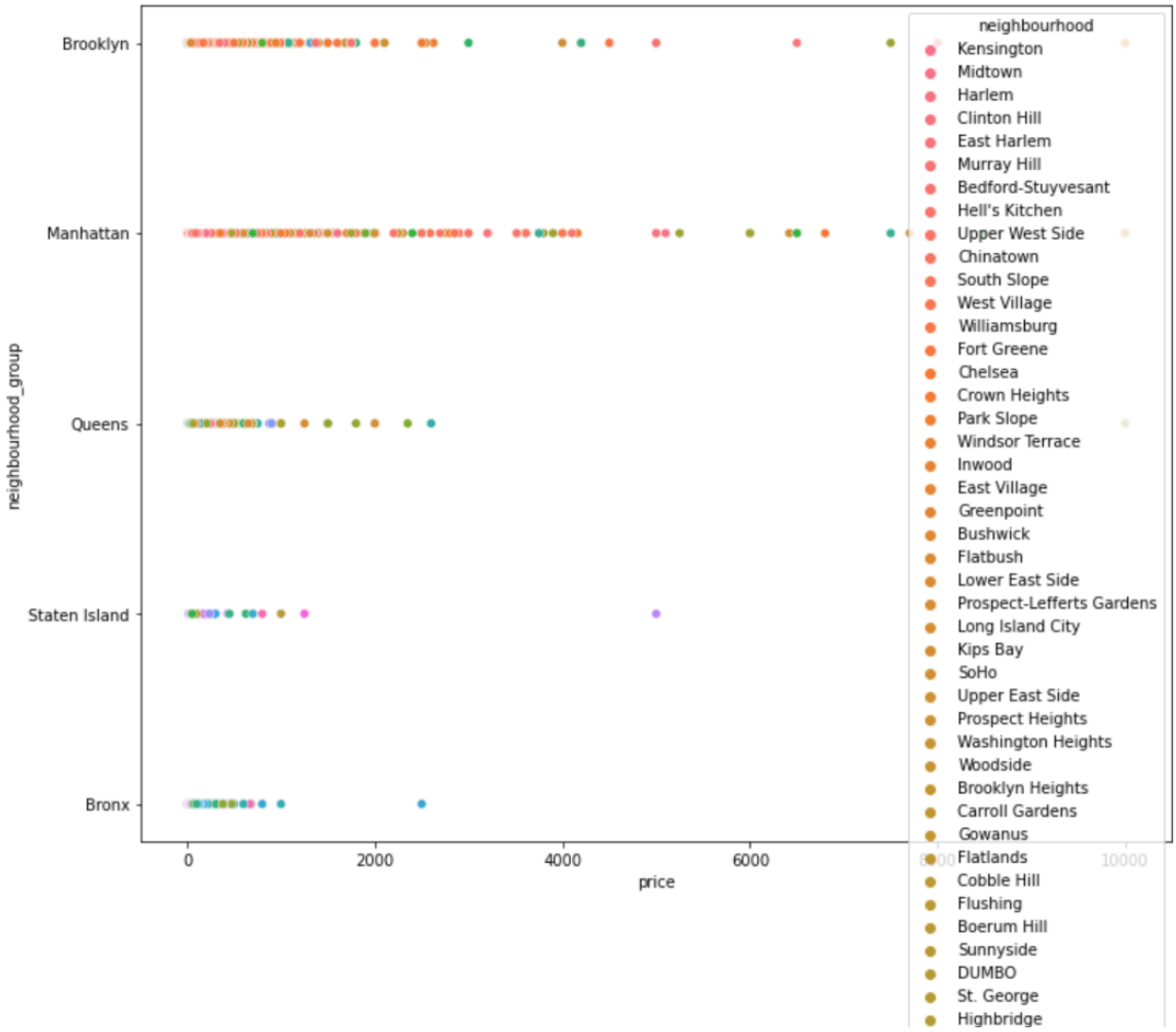


Figure 2. Scatterplot of Preprocessed Data Comparing Price to Neighborhood Group and Neighborhood

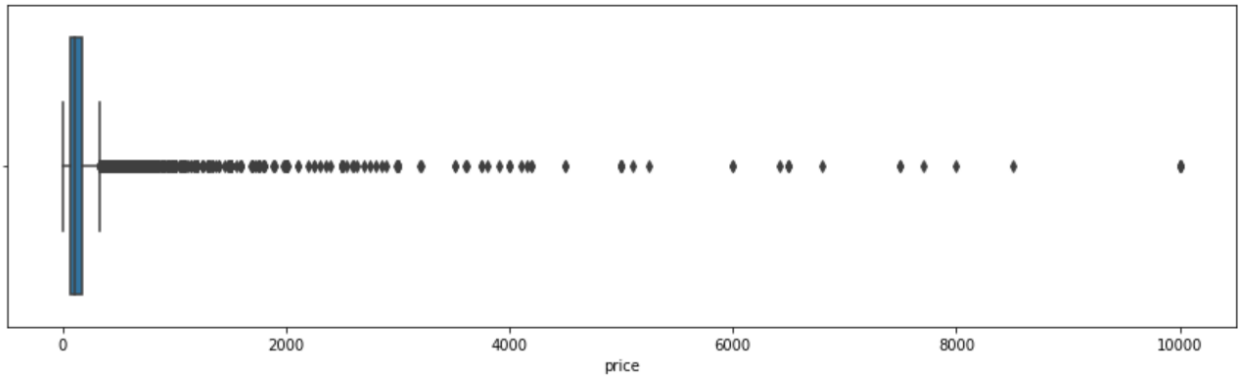


Figure 3. Boxplot Representing Distribution of Prices of NYC Properties in Dataset

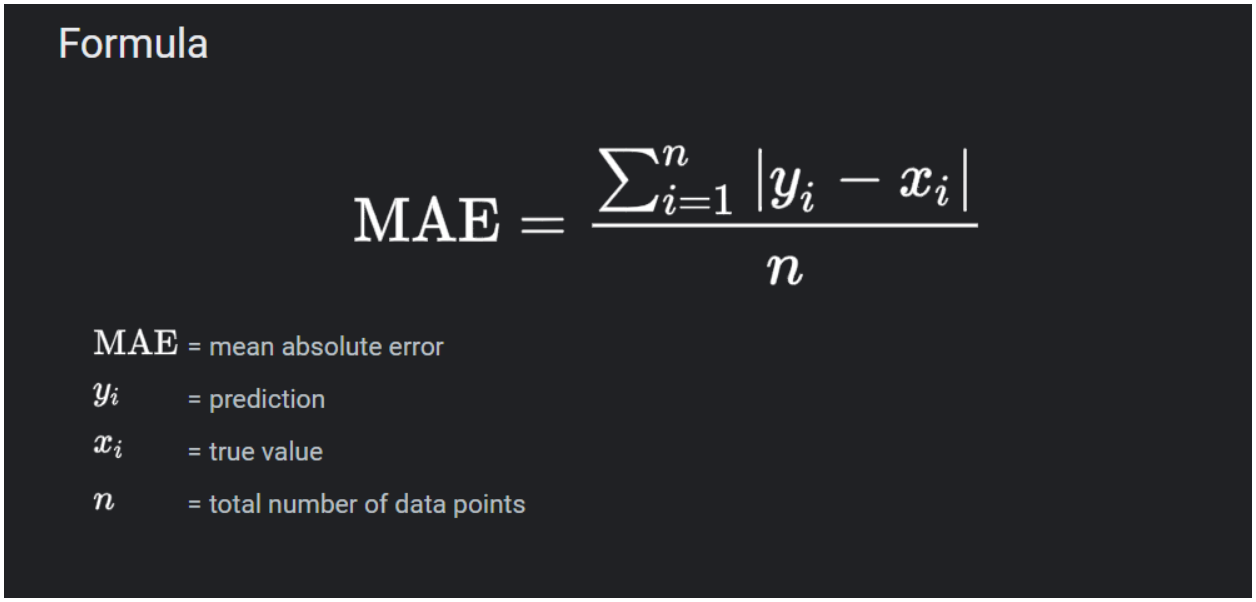
Methodology/Models:

Prior to running the baseline models, I needed to modify my data into a format that could be understood by the machine learning packages. I divided my data into an X matrix, which contained all the samples (rows) and features (columns) in the dataset, and a y vector, which contained the labels and values which correlated to each sample. Then, to split my X and y groups into training and testing groups, I had to use the train_test_split() function, as this is a supervised machine learning problem. The test_size parameter was set to 0.2, meaning that 20% of the data was used for testing.

	Linear Regression	**Decision Trees**	**Neural Network**	**Random Forest**
Hyperparamenters	n/a	Random state: 0	Hidden layers: 3x3x3 Max iterations: 50	Max depth: 2 Random State: 0

During the model creation step in my research, I drew inspiration from my findings during my literature review and decided to create a Linear Regression, Random Forest, Decision Trees, and Neural Network model. In developing each model, I follow the four standard steps of initialization, training, prediction, and evaluation.

In creating my linear regression model, I used the sklearn library to initialize the model. Then, using the regression_model.fit() function and the X and y training groups as parameters, I fit the model. I established a variable called linear_predictions and set it equal to the function regression_model.predict(), which used the X test group as a parameter, as that is the data I am feeding into the model to make an estimate based off. Finally, I imported mean_absolute_error from sklearn.metrics to evaluate the accuracy of the linear regression model, using the parameters of y_test and linear_predictions in order to compare the actual and estimated data.



Equation 1. Formula for Calculating Mean Absolute Error

I followed a similar process to build the Random Forest, Decision Tree, and Neural Network models. For the Random Forest, the parameters were max_depth and random_state. For this project, the max_depth was set to 2. The max_depth refers to the maximum depth of the tree, or the number of splits that each decision tree is permitted to make. Creating too many or too little splits in the decision tree can lead to overfitting or underfitting.

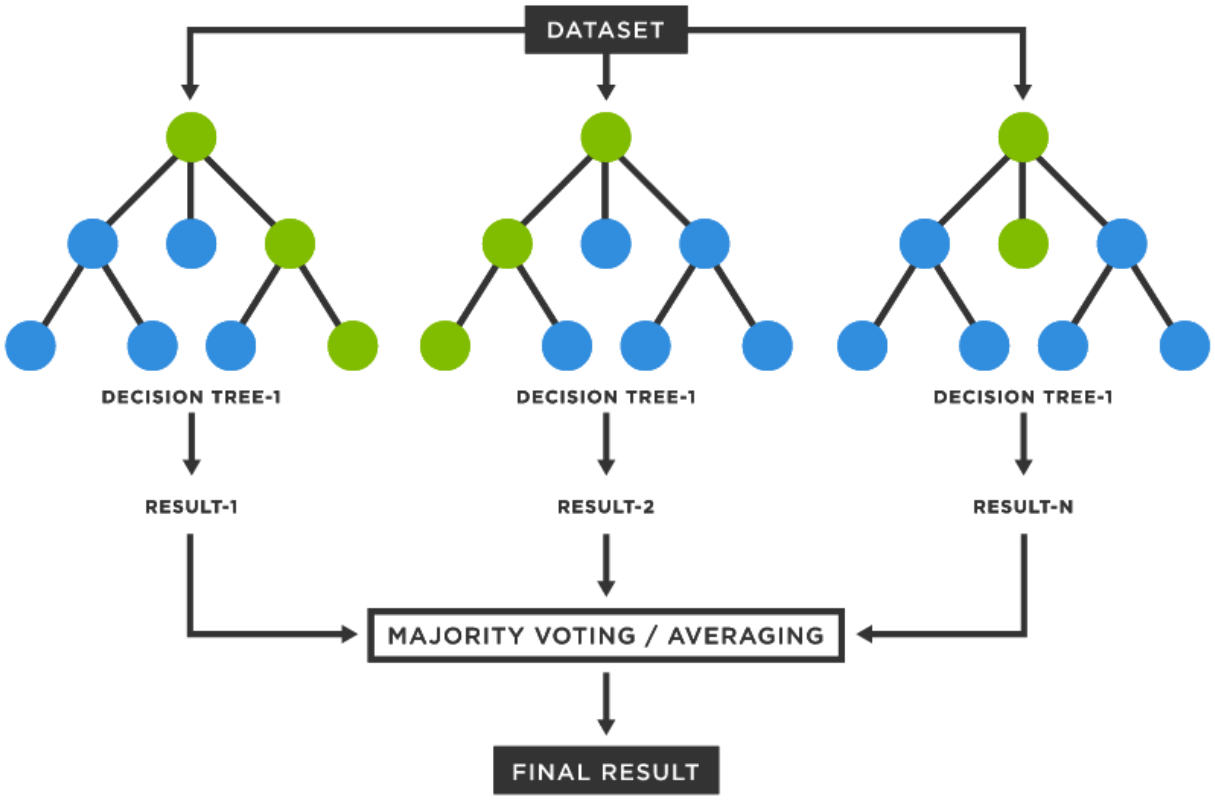


Figure 4. Random Forest Model with Splits

In building the Decision tree, the only parameter that I controlled was random_state. I set this parameter equal to 0, meaning that the train and test sets will remain consistent among different executions of the decision tree model.

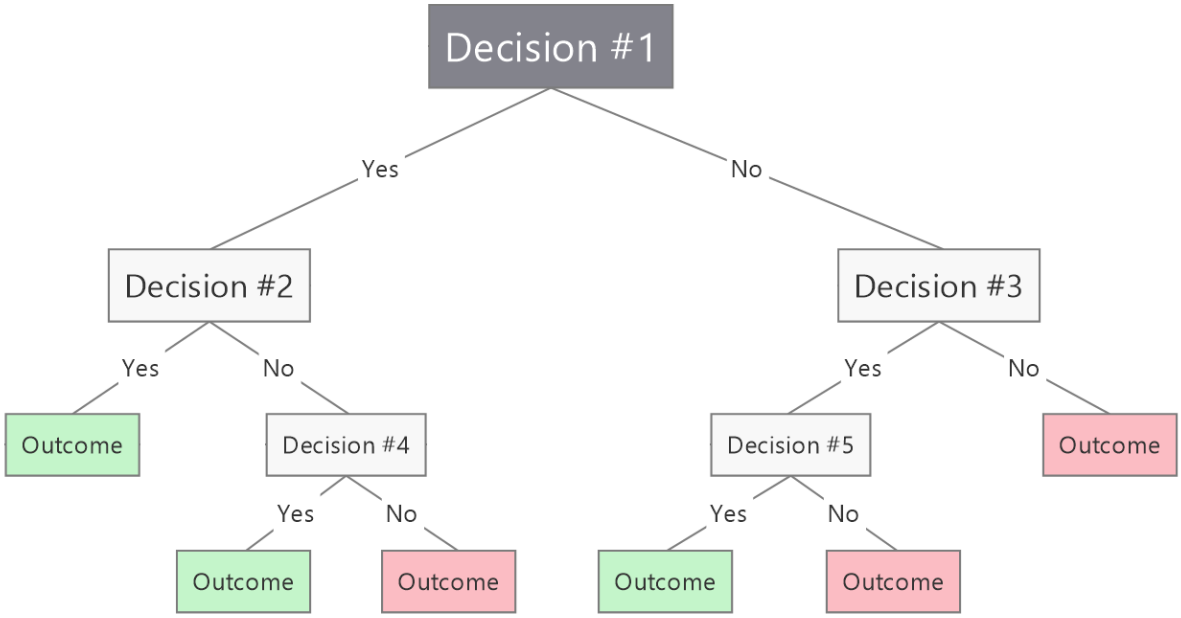


Figure 5. Decision Tree Model Diagram

Much like the aforementioned models, the neural network was built in the same fashion. I imported an MLP classifier to initialize and fit the model. I set hidden_layer_sizes to (2,2) as a baseline, meaning there are 2 hidden layers between the input and output layers with 2 neurons in each one, and the maximum iterations to 50. Again, making the maximum iterations too high can result in overfitting.

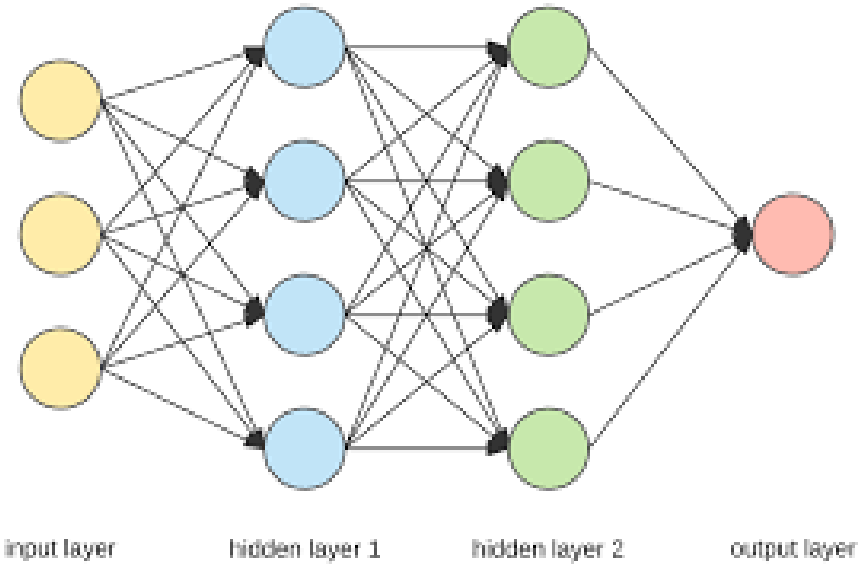


Figure 6. Neural Network Model with 2 Hidden Layers

Results and Discussion:

Prediction Error after Adjustment	**Linear Regression**	**Decision Trees**	**Neural Network**	**Random forest**
Best Mean Absolute Error (MAE)	75.236	88.360	69.229	64.759

The initial MAE calculations show that the neural network performed the best, while the Decision Tree model was the most inaccurate in its predictions. However, I wanted to try and improve the accuracies of the Random Forest, Decision Trees, and Neural Network models by iterating through different values for their parameters.

Initially, the MAE of the Random Forest model was 72.869, but by iterating through the values of the max_depth parameter for the Random Forest, I found that its best MAE, 64.759, occurred when the max_depth was set to 11, thus outperforming the initial error of the neural network.

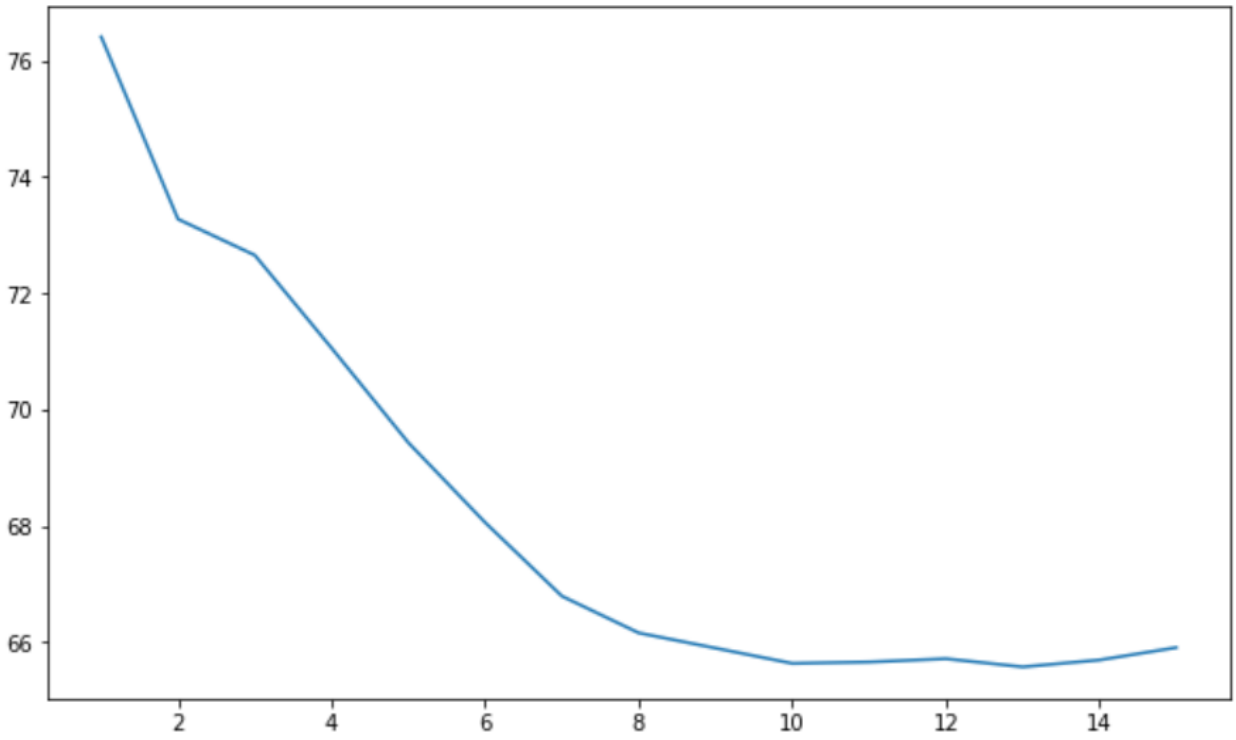


Figure 7. Iteration through Values of max_depth Parameter for Random Forest

In addition, I used the features_importances_ function to determine which characteristic in the dataset had the biggest influence on the property price. As represented in Figure 8, the function determined the latitude and longitude, essentially location, played the biggest role in determining the value of a property. After that, characteristics relating to availability had the strongest influence.

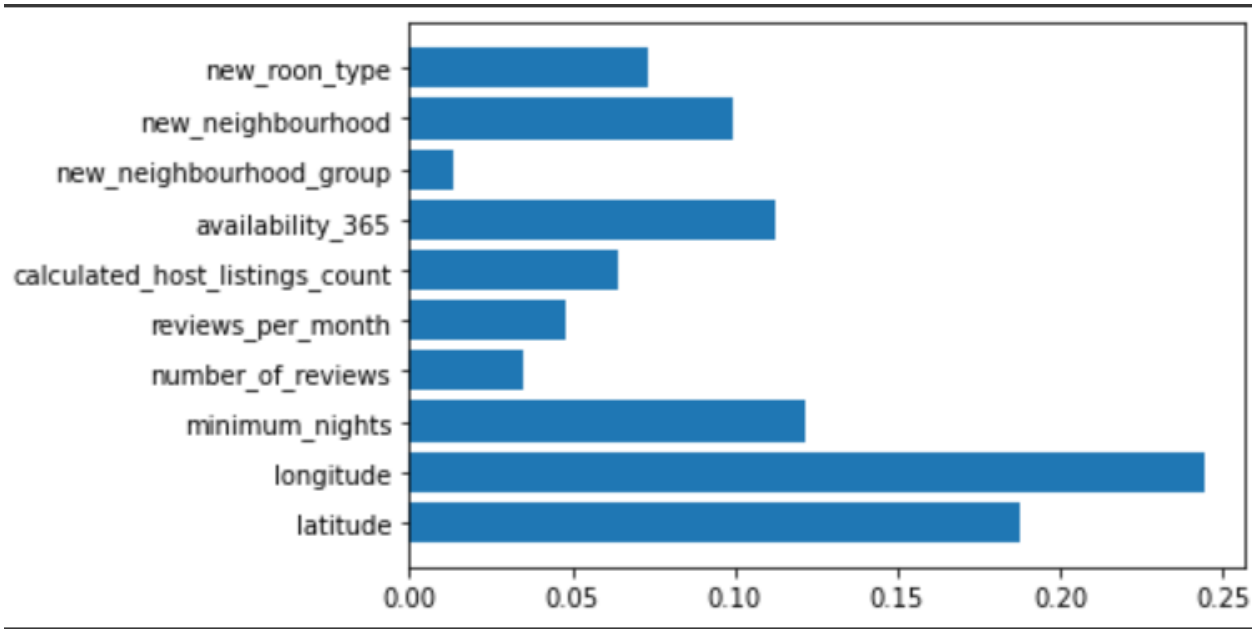


Figure 8. Results of Feature Importances Function for Random Forest

In determining the best MAE produced by the Decision Tree model, I employed the same tactic of using a for loop to iterate through the values 1 to 20 for the random_state parameter. The loop returned the best MAE, 85.827, which occurred when random_state was set to 19. After plotting the graph of the Decision Tree model, I noticed that it did not represent a normal loss curve, so in the future I would look more into that.

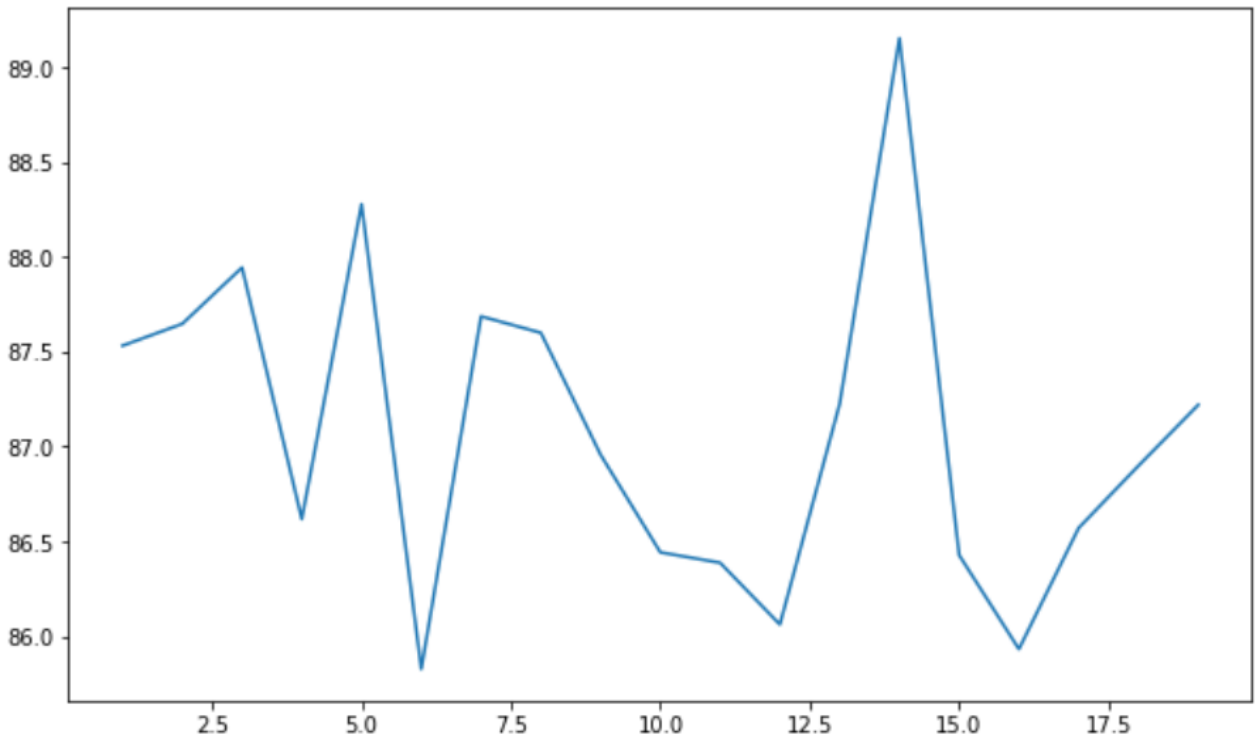


Figure 9. Iterating through Values of random_state for Decision Tree Model

I once again used the features_importances function to evaluate which property characteristics were most influential in predicting prices. The Decision Tree revealed that it depended most heavily on location and availability in its predictions, which is consistent with the Random Forest model.

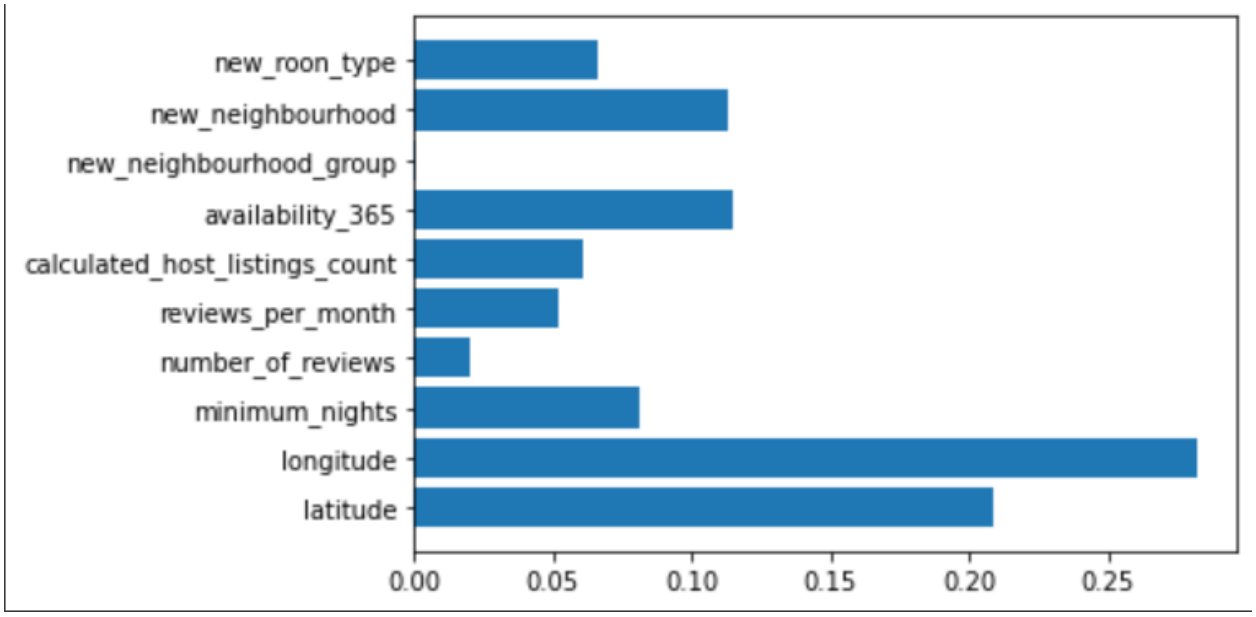


Figure 10. Results of Features Importances Function for Decision Tree

Finally, in evaluating the neural network, I once again used a for loop to iterate through a parameter to find the best MAE. This time, I iterated through the number of hidden layers and the number of neurons in each. For the for loop, the max iterations was set to 200 rather than 50. The loop found that the best MAE was 69.229, which occurred when the number of hidden layers and the number of neurons in each was set to 8. When this value changed to 9, the MAE actually got worse, indicating overfitting.

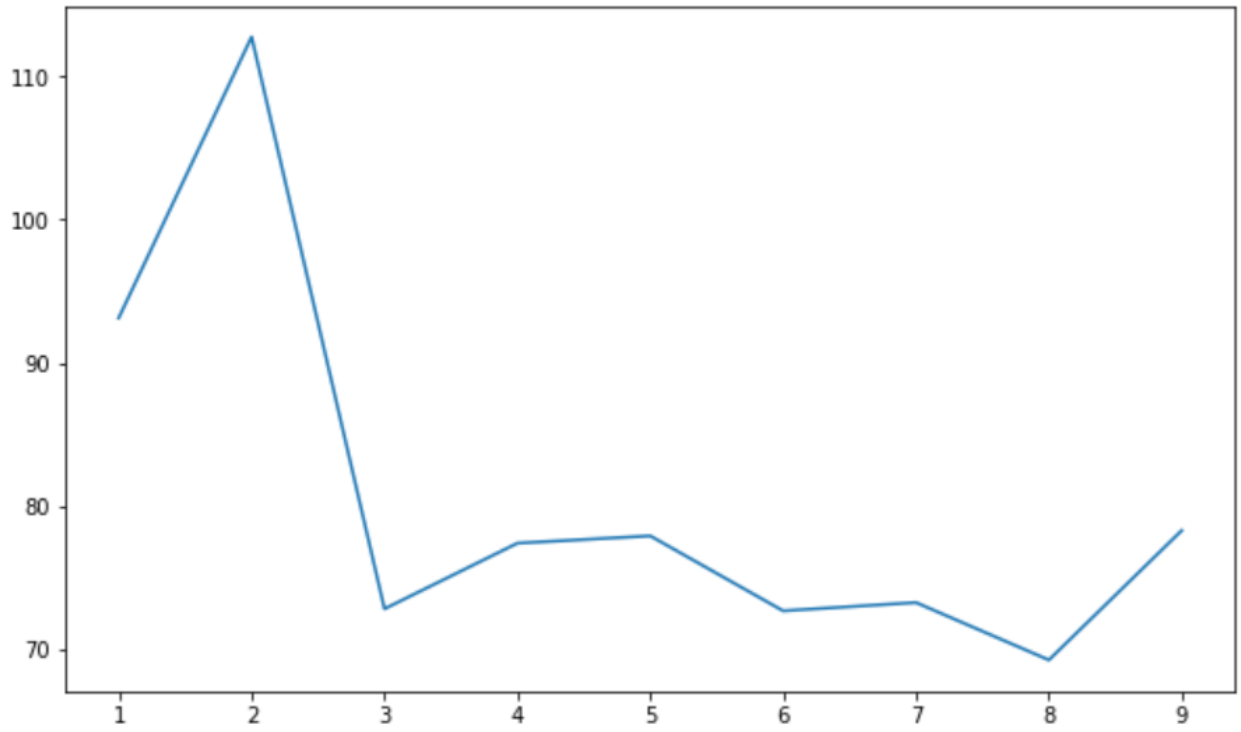


Figure 10. Iterating Through Values of hidden_layer_size for Neural Network

By finding the best MAE values of each model, I was able to objectively determine that the neural network was most effective in predicting the property prices based off of the features in the dataset. In addition, using the features importances function enabled me to learn that location was most significant in influencing real estate valuations, while availability details came in second.

Conclusion:

After analyzing the different types of models and their accuracies, I was able to conclude that the neural network was most effective in predicting real estate property prices, with the Random Forest model being second-most effective. Also, using the feature_importance function, I was able to determine that the most influential characteristics in determining property prices are location and availability. This information will help home buyers and sellers better interpret the housing market and make more informed purchases. Looking to the future, researchers can analyze the language aspects of properties, such as the marketing diction and host names, to see how they impact the prices.

References

1 - F. Francois and S. Rady, “Housing Market Dynamics: On the Contribution of Income Shocks and Credit Constraints,” *Review of Economic Studies*, vol. 73, no. 2, 2006.

2 – Stein, Jeremy C. 1995. “Prices and Trading Volume in the Housing Market: A Model with Down-Payment Effects.” Quarterly Journal of Economics 110 (May): 379-406.

3 – K. E. Case, R. J. Shiller, and J. M. Quigley, “Comparing Wealth Effects: The Stock Market versus the Housing Market,” *Advances in Macroeconomics*, vol. 5, no. 1, 2005.

4 - S. O'Brien, “Jobs, home prices and market volatility are among clients' big concerns right now, advisors say,” *CNBC*, 03-Oct-2022. [Online]. Available: https://www.cnbc.com/2022/10/03/jobs-home-prices-market-volatility-are-client-concerns-advisors-say.html. [Accessed: 21-Dec-2022].