Apartment Sale Price Prediction in Manhattan

Jieyu Lu (jl8570), Dongning Fang (df1352), Yuxin Tang (yt1526), Jiayu Qiu (jq429)

OutOfIndex, Center for Data Science, New York University

I. Business Understanding

We consider ourselves as a research team employed by Marketing Analytics department under Realtor.com, an online real estate database company. Comparing with our large amount of properties listed for sale, the monthly visits to our website is relatively low. We notice that Zillow.com, the real estate website with the largest monthly visits, has a few interesting features besides its delicate and well-designed website. One of them is a price estimator called 'Zestimate'[4]. This is really important and helpful because living in a metropolitan like New York City, housing expense can be the largest expense for a family. Also, the real estate market is always fluctuating so that both buyer and sellers (renters and owners) do not have a clear idea about what the fair price is. We believe the price estimator has a positive contribution to the monthly visits because customers always prefer a reasonable price.

Therefore, the goal of our project is to design a real estate estimation system based on public sales data for Realtor.com. This could help Realtor to better compete with Zillow by attracting more customers to visit the website but more importantly, sellers and buyers will be provided with an opportunity to compare and learn what a "reasonable price" is and make wiser decisions.

II. Data Understanding

Initially, we obtained data for apartment sales in Manhattan from October 2017 to September 2018 from NYC.gov[1], which includes features that can possibly affect the sale price of an apartment such as square feet, built year, address, etc. There are numerical data such as built year, apartment number, apartment size in square feet, while the information contained in the address, building class, tax class is categorical. Also, there are some data missing from apartment size in square feet and apartment number. Extreme values of sale prices exist and a small number of them are missing as well, which we suspect to be apartments being inherited.

In order to improve our baseline model, which will be explained later, we scraped data from Streeteasy.com[3], where we browsed more than 9,000 pages by searching corresponding addresses in Google. To avoid anti-bot detection, we mimic the behavior of normal browser with *selenium* and *time* library. The raw HTML page was then parsed by *BeautifulSoup*. Coordinates data were obtained by geopy library with *Nominatim*, an open source geocoding from OpenStreetMap[5] data.

The graphs (**Figure 1**) below show the distribution of sales price in our original data (for the baseline model). The sales price concentrates at $1,000,000 to $ 1,500,000 and right-skewed.

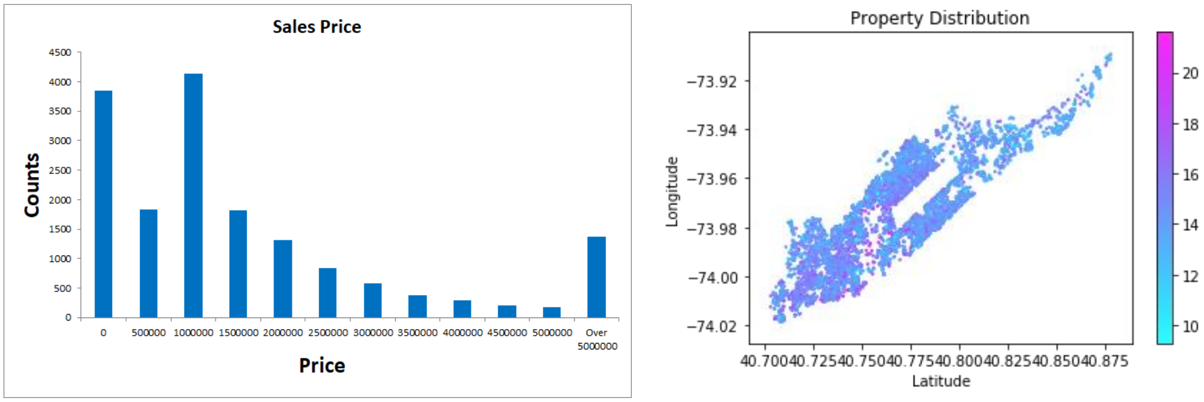

Figure 1. Target distribution. Left: Raw data from NYC.gov. Right: Heatmap of property. Each point represents an instance.

From the heat map (Figure 1), most apartments with high sale prices are concentrated at lower Manhattan as shown by those purple points. Thus, we believe location is an important factor of sale price. We took log of the sale price for a clearer plot.

Although our data includes fairly well records of transactions in Manhattan, our data selection is very likely to be biased. The dataset contains transaction records only from October 2017 to September 2018, thus any apartments not being transacted within this time period are not taken into account. Also, we deleted all transactions with missing sale price or with sale price equals to 0. However, these apartments may share some characteristics that cannot be represented by our data.

III. Data Preparation

The target variable, in this case, is the sale price of a personal residential apartment in Manhattan. It is numeric and non-negative and thus regression is used to generate the prediction model. Missing values in the target variable (sale price) are deleted. We also excluded sale prices under $10,000 because it is highly unlikely these prices reflected the true value of the property.

An instance is a transaction record from our original dataset. Because many features in the original dataset are categorical, we transformed these features into dummy variables. Such transformation includes:

1. 11 dummy variables used to indicate 12 sale months

2. 3 dummy variables to indicate apartments built in 1800-1900,1900-2000, and 2000-later (missing built years are represented by 0).

3. 5 dummy variables used to indicate “One Family Dwellings”, “Two Family Dwellings”, “Walk Up Apartments”, “Elevator Apartments” , “Condominiums” and “Mixed Use”. [2]

4. 2 dummy variables used to indicate tax class 1,2 and 3 (we noticed that our dataset did not include apartments with tax class 4)

5. Missing value dummy variables used to indicate missing values of size in square feet

6. 38 dummy variables used to indicate 39 neighborhoods in Manhattan

Besides the dummy variables, we also used the StandardScaler in sklearn to standardize the block and lot numbers, and size in square feet (of which missing values are replaced by mean value.

We split the dataset above into 85% training and 15% testing. The performance of the data on the baseline model (i.e. linear regression) is not promising. We found that apartment size in square feet had too many missing values that hurt the quality of the feature. Sale months does not present a strong relation to the target variable. Too many dummy variables used to indicate one feature is not informative.

Hence we scraped new data from Streeteasy.com. We selected transactions in Manhattan during the same period of time. We made some adjustments based on the new data:

1. More data in apartment size are provided by Streeteasy.com[3].

2. We were also able to grab new features - the number of bedrooms and bathrooms - that we believed can also affect the target variable.

3. We replaced building class and tax class with two dummy variables that indicated whether the apartment had doorman or elevator. Such replacement is based on our observation that doorman and elevator are more straightforward features.

4. Block and lot numbers, as well as neighborhoods, were replaced by the numerical coordinates (i.e. latitude and longitude) that gave a more specific location.

5. We also selected number of stories and number of units in the building as new features.

6. Finally, dummy variables indicating building year was again included in the new dataset.

In terms of feature engineering, as mentioned earlier in this section, we came up with some features that we believed are related to the target variable. We checked the performance of these features in the baseline model and improved feature selection. During the process, to improve the accuracy of the model, dummy variables indicating missing values are included when missing values are replaced by the mean value. Besides, numeric variables are also normalized.

IV. Modeling & Evaluation

Given our goal is to predict the sales price, regression is the main method we relied on. We chose linear regression as our baseline model since linear regression is simple and straightforward. The data used to train the baseline model is the ones from NYC.gov. The value of R^2 between predicted prices and true prices we got from our model was 0.286, which was considerably low. This is possibly due to the simplicity of linear regression, which makes it hard to capture internal relationships between our features and targets. On the other hand, our model contained too many variables and some of them are correlated to some extent while linear regression is sensitive to correlated features. Since we have a large number of dummy variables, high dimensionality also influences the model performance. We considered the size of the apartment as an important feature on its sale price, but we were missing many data for this feature.

We improved our baseline model from three aspects: feature selection, algorithm selection, and hyper-parameters selection. As explained in the data preparation section, the first thing we did was adding more relevant features and obtaining more data for apartment size. We combined this process with feature selection as we replaced existing features with better alternatives and deleted several redundant features which were not very significant.

A. Linear regression:

We fit our updated data with Linear regression again and we observed R^2 value between prediction values and true values of 0.67090. It is the lowest value among all the models due to its low complexity. The R^2 value for both training and testing data are not high, implying that a simple linear model cannot address the problem well, and that is why we turned to tree-based models.

B. Decision Tree Regressor:

Alternatively, we chose Decision Tree Regressor which provides flexible complexity based on hyper-parameter settings. The regressor is able to naturally deal with correlated features and does not require normalization. However, Decision Tree Regressor is unstable. For hyper-parameters selection, we implemented GridSearch and set maximum depth ranging from 6 to 18 (step=3), minimum samples leaf size ranging from 1 to 11 (step=2). (See **Figure 2** below). The best combination is shown in the table and graph below. R^2 (=0.75649) for test data is significantly higher than that of linear regression. R^2 for training data is a bit higher than that of testing data, implying that there is somewhat overfitting. In order to reduce the variance, we also tried Random Forest Regressor.

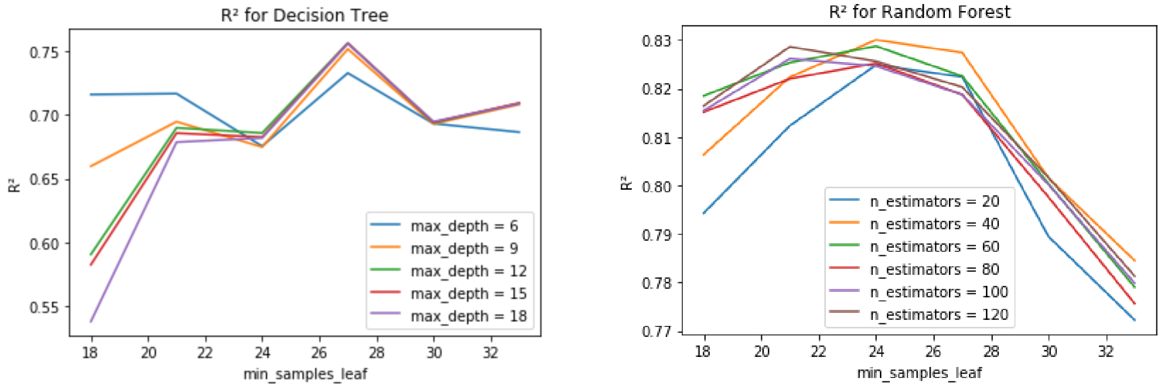

Figure 2. Hyper-parameter searching. Left: Decision Tree; Right: Random Forest.

C. Random Forest Regressor:

Compared with Decision Tree Regressor, it is more robust, but it needs to do a lot of hyper-parameters searching. We set maximum depth ranging from 6 to 21 (step=3), minimum samples leaf size ranging from 1 to 21 (step=4) and n_estimators ranging from 20 to 120 (step=20). (See **Figure 2** above). We observed a higher R^2 (=0.83002) for test data and there is less overfitting comparing with Decision Tree.

D. Gradient Boosting Regressor:

Gradient Boosting Regressor is derived by optimizing the objective function so it should serve our goal. But it is more sensitive to overfitting if our data is noisy. We set minimum samples leaf size ranging from 18 to 33 (step=3), maximum depth ranging from 6 to 18 (step=3), learning rate as [0.01, 0.05, 0.1, 0.2], n_estimators ranging from 120 to 180 (step=20), subsample as [0.8, 0.9, 1], max_features as ['auto', 'sqrt']. See (**Figure 3**) for parts of the results. Through GridSearch, the best combination is shown in the chart, except that n_estimator is determined by the position of "sweet spot" shown in the graph to avoid overfitting. The feature importance graph (**Figure 4**, left) shows that the size, number of stories, number of bathrooms and location are the most significant factors for determining sale price of an apartment.

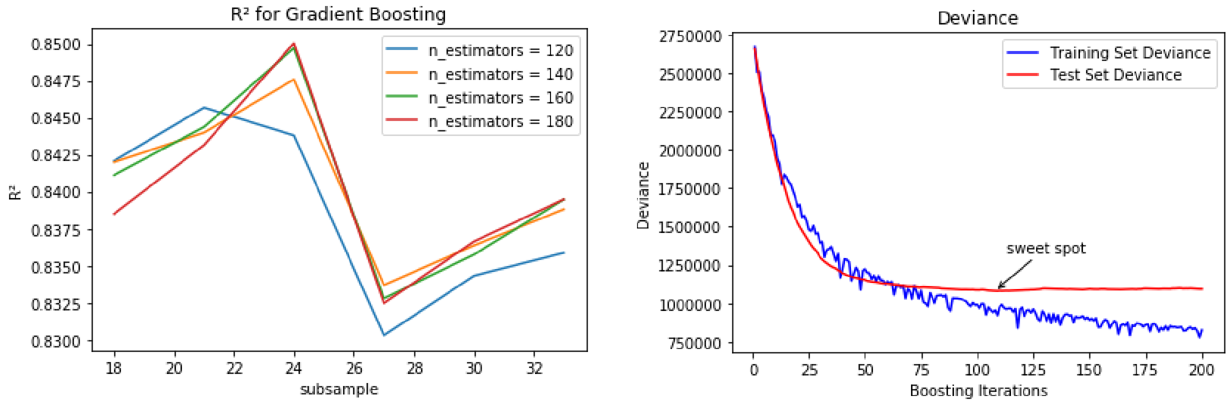

Figure 3. Hyper-parameter searching for Gradient Boosting Regressor. Left: R^2 vs. subsample with different numbers of estimators. Right: Early stopping.

The best result of each regressor is as follows:

Model	Hyperparameters	R^2 for testing data	R^2 for training data
Linear Regression	Normalize = TRUE	0.67090	0.61636

DecisionTreeRegressor	Max_depth = 12
	Min_samples_leaf = 7
		0.75649	0.83911

RandomForestRegressor	Max_depth = 9
	Min_samples_leaf = 9
	N_estimators = 40
		0.83002	0.84366

Gradient Boosting Regressor	Max_depth = 12
	Min_samples_leaf = 21
	n_estimators=108
	learning_rate=0.05,
	random_state=8570,
	subsample=0.9
		0.85004	0.89845

Table 1. Best hyper-parameters for different regressors.

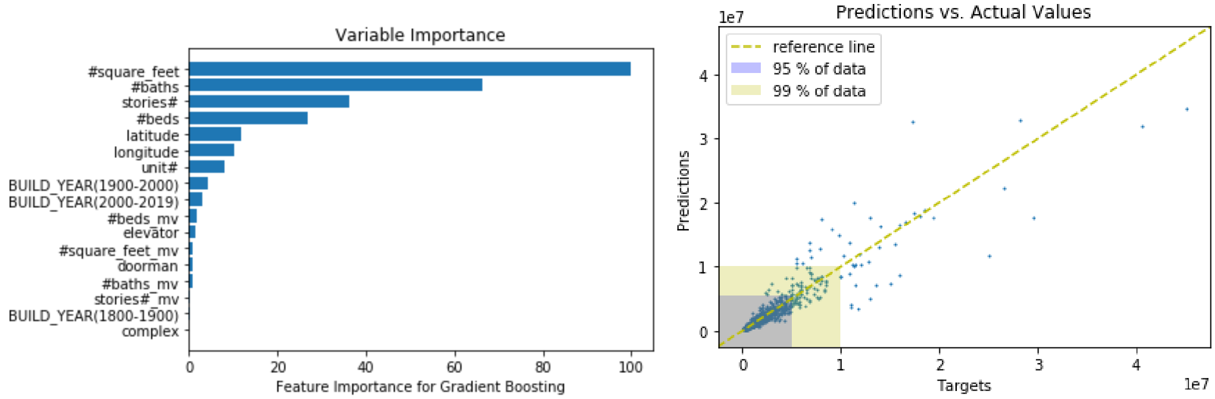

Figure 4. Analysis based on Gradient Boosting Regressor. Left: Feature importance; Right: Visualization of predictions.

From the results, Gradient Boosting Regressor performs the best among the four, with the highest R^2 value of 0.85004. Comparison between predicted sale price and actual sale price in our testing data is shown in the graph above (**Figure 4**, right). From which we can see 99% of our target data is under 10 million dollars and therefore the model is not performing so well in predicting price for high-end apartments. However, Gradient Boosting Regressor usually trains longer since the trees are built sequentially. So if the company concerns much about computational cost, Random Forest Regressor could potentially outperform Gradient Boosting Regressor.

V. Deployment

As mentioned in the goal of our project, our real estate price estimation model has its practical deployment. It helps eliminate asymmetric information and promote a transparent and fair transaction. The price estimation system could be added to other real estate websites like Realtor for both buyers and seller' reference. It provides the users as well as the website itself with a better idea of the value of an apartment and an opportunity to compare other estimations like Zestimate[4].

The real estate market in Manhattan is always active and fluctuating therefore once the estimation model is deployed, the company should update the model frequently to match the most recent market status.

The main risk of our model is prediction error. There are deficiencies in our model since our estimation is based on past sales data which does not include other systematic factors that can directly or indirectly affect sale prices (for example, interest rates). But it should be a good estimator to compare prices horizontally. Also, the firm should be aware that the estimation model relies on past data of Manhattan only. Thus it might not be appropriate to apply it to places other than Manhattan.

Our project should not raise any ethical concerns. Since the target variable is the apartment’s price, there is not anything about human that was studied here. One possible issue here is that some sellers may not want us to show the prediction of their apartment price because lower estimations may undermine their interest. Therefore complaints regarding the accuracy of the price estimations may occur and the firm should be prepared to mitigate this issue by explaining the benefits of the estimation system to these users.

Appendix

Reference:

[1]. New York City Department of Finance (2018). Retrieved from: https://www1.nyc.gov/site/finance/taxes/property-rolling-sales-data.page. Accessed 10 Oct 2018.

[2]. NYC Building Class. PropertyShark. Retrieved from: https://www.propertyshark.com/mason/text/nyc_building_class.html. Accessed 15 Sep 2018.

[3]. StreetEasy. Retrieved from: https://streeteasy.com. Accessed 1 Dec 2018.

[4]. Zestimate. Retrieved from: https://www.zillow.com. Accessed 15 Sep 15 2018.

[5]. OpenStreetMap. Free Wiki World Map: https://www.openstreetmap.org. Accessed 20 Sep 2018.

Contributions:

Most of the work are evenly distributed.

Jieyu Lu: Notebooks to scrape the data and geocoding. Set up of the machine learning framework (model selection). Plotting of hyper-parameter searching and feature importance, paper write-up

Dongning Fang: Data cleaning and processing, scraping the data, hyper-parameter searching, analysis and paper write-up

Yuxin Tang: Data cleaning and processing, scraping the data, hyper-parameter searching, analysis and paper write-up

Jiayu Qiu: Data cleaning and processing, scraping the data , hyper-parameter searching, analysis and paper write-up