Informative Features for Predicting NYC Property Value

Abstract— **Our work primarily aims at selecting important features to predict New York City (NYC) property sale price in five different boroughs using random forests. The secondary goal is to predict property sale price by building an artificial neural network model.**

Keywords: NYC Property Sales, Random Forests, Artificial Neural Networks

I. INTRODUCTION

Nowadays, buying or selling a property is of great concern for most of US families, especially for people living around metropolitan areas like New York City (NYC). To find a reliable way to determine the value of a property is not an easy task. Machine learning techniques provide us valuable tools to predict sale price according to different features such as the location of property, the age of property etc.

II. RESEARCH QUESTIONS

In order to get better prediction of the value of different properties, we would like to know which factors contribute most to the sale price. In addition, it is worthwhile to explore various models to find one with most predictive power.

III. DATA PREPARTION AND FEATURE ENGINEERING

We integrate five data files representing rolling sales data from five different boroughs in NYC including Manhattan, Bronx, Brooklyn, Queens and Staten Island. This is annual data from Feb 2017 to Jan 2018. There are 21 variables including response variable sale price, boroughs, neighborhood, building class category etc. We will do some data inspection and clean the data in next step.

Since we are only interested in residential properties, tax class type is restricted to either 1 or 2. After removing several duplicates, choosing the building class type at present equals to that at time of sale, removing missing values in response variable, there are 55414 observations in total. We observe that the response variable sale price is highly skewed, and therefore, log transformation is applied to sale price to achieve normality and the post-transformation distribution plot is displayed in Figure 1. Next, we will prepare our features by data cleaning and feature engineering.

Preliminary visualization shows that land square feet and gross square feet, total units and residential units or commercial units are highly correlated, and hence we only keep one of them accordingly. Additionally, outliers (> 98% or < 2% quantile cutoff) are removed for some numeric features. The sales dates are mapped to four seasons (winter, spring, summer, fall). By investigating the building year, we notice that there are three peaks in the distribution and hence after restricting our sample to the properties after 1700 we categorize this feature to three categories: 1700-1920, 1920-1960 and 1960-2018. In addition, we need only general class of building and thus keep first position of the class letters.

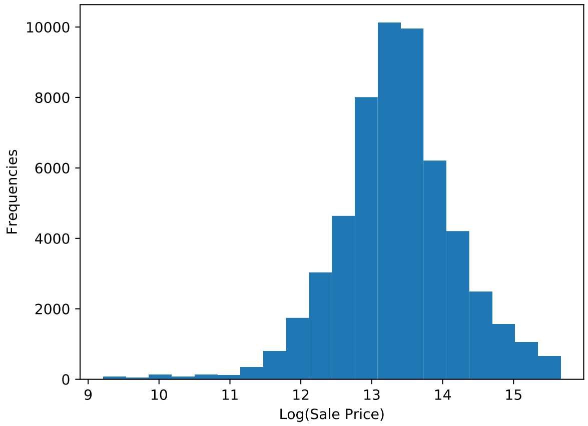


Figure 1. Log-Transformation of Sale Price

Moreover, we drop features like address or block and keep informative features in predictive modeling. After data preparation and feature engineering, 25359 observations are in our sample and our feature dimension is 8. Categorical features contain borough, neighborhood, building class, built year category and season. Land square feet is continuous and total units is ordinal (treated as continuous). The final step in data engineering is to create dummies for categorical features and we have 244 features in total.

IV. DESCRIPTIVE ANALYSIS AND HYPOTHESIS

In this section, we explore the relationship between some features and our response variable and display the descriptive visualization results. In the boxplots below (Figure 2-4), we show the relationship between log sale price and some categorical (ordinal) features. Borough is discriminant in terms of log sale price as the property values in Manhattan and Bronx are clearly different. The number of total units is strongly related to log sale price because with the increasing of total units, the property value increases as well. Building year category does not have enough power to predict the property value. Building class might have weak capacity in terms of predicting property value. Similarly, sale season does not seem to have association with the value property. The scatterplot of log sale price versus land square feet indicates some weak relationship between them.

From descriptive results, we assume that our response variable log sale price is normally distributed and there are some association or interaction between features collected and the response variable. Next we form our hypothesis that borough, total units and building class are informative factors in predicting property value in NYC. To test our hypothesis, we will build some machine learning models.

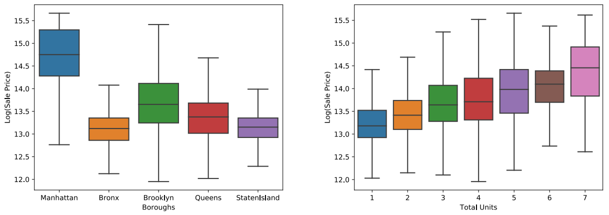


Figure 2. Boxplots (Price by Boroughs / Total Units)

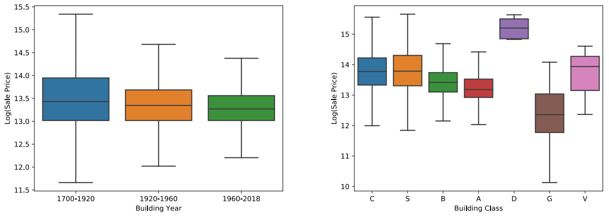


Figure 3. Boxplots (Price by Year / Building Class)

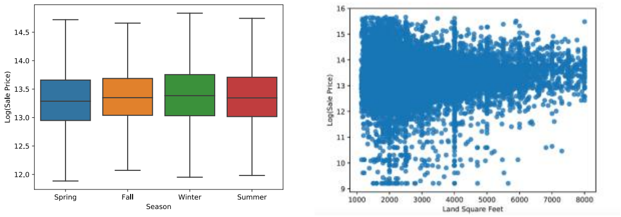


Figure 4. Boxplot /Scatterplot
(Price by Sale Season / Land Square Feet)

V. Methods

Random forest regression is used to predict the value property thanks to its robustness. We split the sample into train and test set according to 70/30 proportion. Parameters including maximum of feature, minimum sample leaves and number of trees are tuned with grid search and best parameters are selected by five-fold cross-validation. Mean square error (MSE) is used as the loss criterion. Moreover, a random forest regression with best tuning parameters is performed on whole train set to get important features and then the model is evaluated on the test set.

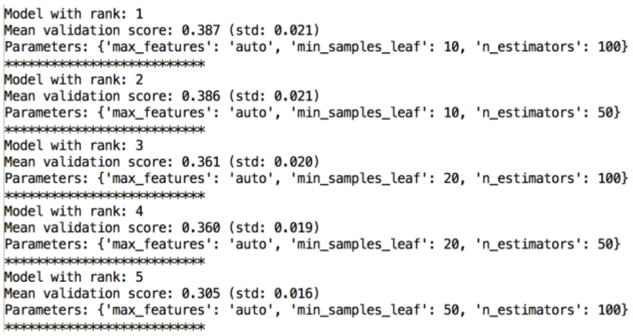
Furthermore, we build a feed-forward artificial neural network model (ANN) to see if predictive power can be improved by neural networks.

VI. Experiment Results

Random Forest

From cross-validation results, the best parameters selected are the following parameter setting: maximum of features = square of root of total features, minimum samples of leaves = 10 and number of trees = 100. The best validation score (coefficient of determination R^2) is 0.387 with standard error 0.021 (Figure 5). The R^2 in test set is 0.397, which demonstrates that our model is reasonably well. We fit the selected model on the whole train set again to get importance feature rank and the R^2 is 0.502 in the train set.

For random forests regression, we get top 5 importance variables, these variables are the most important in building the trees (as splitting criteria): land square feet, borough (Brooklyn), total units, borough (Manhattan), borough (Queens). The feature importance is displayed in Figure 6.



```
Model with rank: 1
Mean validation score: 0.387 (std: 0.021)
Parameters: {'max_features': 'auto', 'min_samples_leaf': 10, 'n_estimators': 100}
**************************
Model with rank: 2
Mean validation score: 0.386 (std: 0.021)
Parameters: {'max_features': 'auto', 'min_samples_leaf': 10, 'n_estimators': 50}
**************************
Model with rank: 3
Mean validation score: 0.361 (std: 0.020)
Parameters: {'max_features': 'auto', 'min_samples_leaf': 20, 'n_estimators': 100}
**************************
Model with rank: 4
Mean validation score: 0.360 (std: 0.019)
Parameters: {'max_features': 'auto', 'min_samples_leaf': 20, 'n_estimators': 50}
**************************
Model with rank: 5
Mean validation score: 0.305 (std: 0.016)
Parameters: {'max_features': 'auto', 'min_samples_leaf': 50, 'n_estimators': 100}
**************************
```

Figure 5. Cross Validation Results from RF

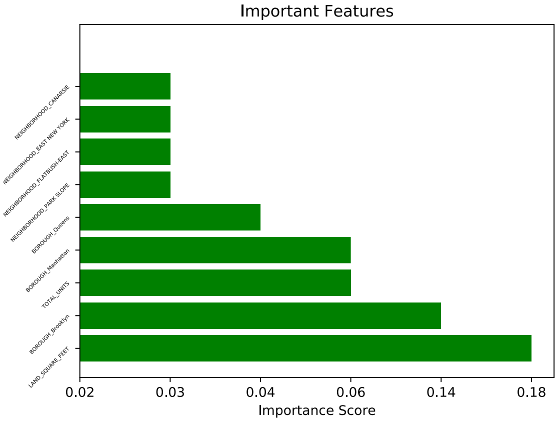
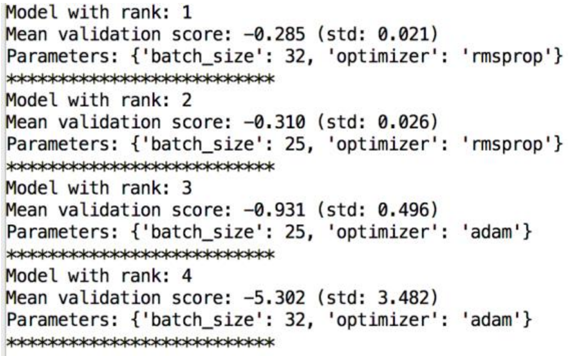


Figure 6. Feature Importance Ranking

Moreover, we build an ANN model to predict value of properties in NYC. Similar to the previous random forest model, tuning parameters (batch size and optimizer) are selected using grid search with five-fold cross validation.

Two hidden layers are specified and a dropout layer for each hidden layer is added with dropout rate of 0.2. The loss function is MSE (actually negative MSE in Python keras library). The cross-validation results are shown in Figure 7.

The best parameters selected are batch size of 32 and optimizer of ‘rmsprop’. R^2 in the test set equals to 0.398, which slightly outperforms that of random forest model.



```
Model with rank: 1
Mean validation score: -0.285 (std: 0.021)
Parameters: {'batch_size': 32, 'optimizer': 'rmsprop'}
**************************
Model with rank: 2
Mean validation score: -0.310 (std: 0.026)
Parameters: {'batch_size': 25, 'optimizer': 'rmsprop'}
**************************
Model with rank: 3
Mean validation score: -0.931 (std: 0.496)
Parameters: {'batch_size': 25, 'optimizer': 'adam'}
**************************
Model with rank: 4
Mean validation score: -5.302 (std: 3.482)
Parameters: {'batch_size': 32, 'optimizer': 'adam'}
**************************
```

Figure 7. Cross Validation Results from ANN

In Conclusion, land square feet, borough and total units are important informative features in predicting value property in NYC. This is not exactly the same as what we hypothesize after running some descriptive analyses but very close. Random forest regression provides us good interpretation of the association by feature importance ranking. In addition, ANN has the capacity to improve predictive power in modeling.

VII. Discussion and Extension

Missing data was not handled in this work due to its complexity. In the future analysis, we can consider more precise missing data mechanism and implement multiple imputation for some features. For the missing values in response variable, selection bias can be addressed using inverse probability weighting.

Random forests regression is a valuable tool to select important informative features in prediction task. It can handle high dimensionality and avoid the issue of multicollinearity. As an extension, we could use adaptive boosting using decision tree as base estimators for comparison purpose regarding important features.

ANN is not fast due to computational power of my personal laptop and it is tested on CPU. To improve ANN results, we could implement the model on GPU and tune more parameters such as number of epochs (specified as 100 in this example). Moreover, we can build more complicated architecture of neural networks such as better-designed dropout regularization and hidden layer dimension.

The Python code is attached on the next page.