Price Prediction Project

By Amy, Brad, Jainam, Prabhat, Rameez

11 March 2020

Agenda

1. Objectives and Business Case
2. Data Overview and Preparation
3. Exploratory Analysis and Unsupervised Modeling
4. Supervised Modeling
5. Key Takeaways

Objective & Problem Statement

Why?

- To help new and existing owners fix the most competitive price
- This isn’t an easy ask

How?

- A ML model that uses several methods to predict the right price
- Saves huge time, effort and cost on the part of the owner

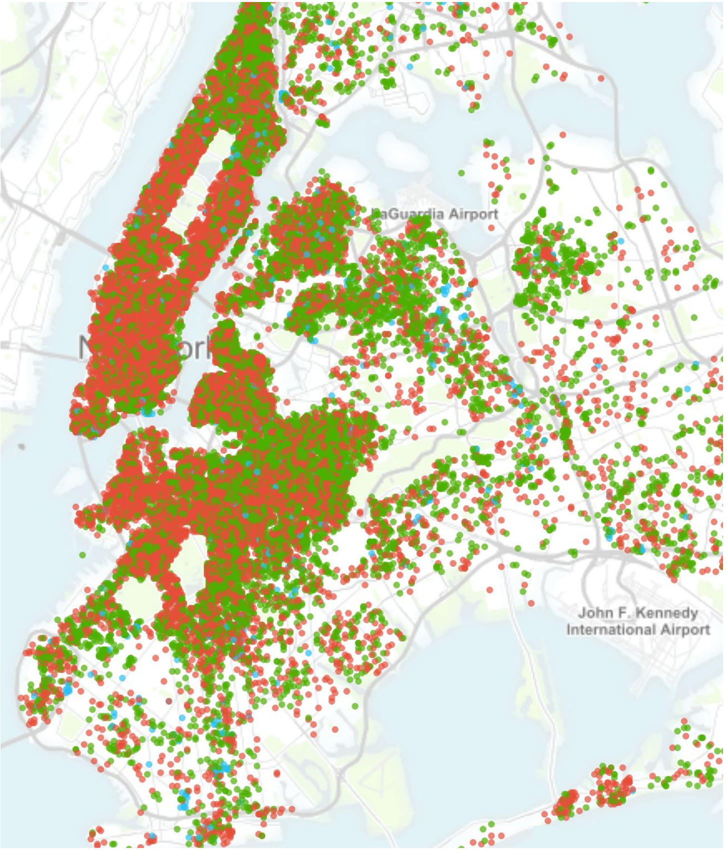
Who?

- To owners (or to hosts with multiple properties looking to list new ones)
- To a consulting firm in this business
- To Airbnb in order to improve their “suggested price” metric

Data Cleaning and Preparation

The Dataset

- New York dataset, with 50,600 Airbnb listings
- Contained a wide range of information including:
    - Location and neighborhood
    - Property and room type
    - No. of guests, bedrooms bathrooms
    - Amenities included (130 types)
    - Price, security deposit, cleaning fee
    - Reviews and rating
- Also contained a lot of additional information which we dropped:
    - Text data: listing name, summary, descriptions
    - URLs, pictures, etc



Data source: http://insideairbnb.com/get-the-data.html

Data Cleaning & Key Assumptions

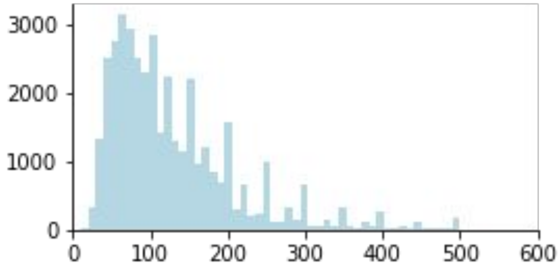
- Additional steps on cleaning and refining the data using Python.
  - Dropped all listings with 0 reviews in the last 12 months.
    Assumption: these are likely 'inactive' properties which don't reflect true price
  - Dropped outliers and listings with extremely high prices (2%), by filtering for price less than $500
  - Extensive missing values in square feet, security deposit and cleaning fee, so those columns were dropped.
    Assumption: security deposit and cleaning fees, are time independent (same amount charged regardless of the length of stay)
- For predictions, we focused on the daily price only, rather than weekly and monthly prices, as this was the most complete data.
- Final data dimensions were (28,268, 37) from (50,600, 107) originally.
- Created train and holdout validation samples with a 70:30 ratio.

Exploratory Analysis and Unsupervised Modeling

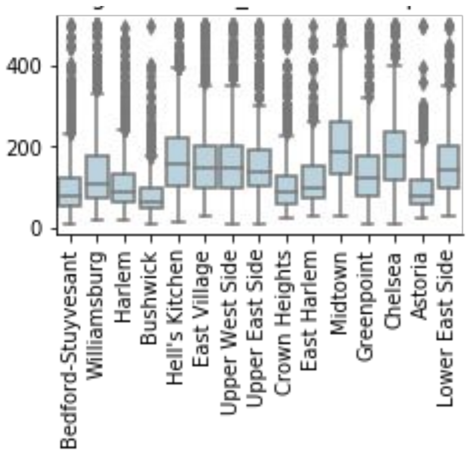
Exploratory Analysis

- Price skewed cheaper with a median of $100
- In the dataset given, 5.7% hosts manage 3 or more listings, which account for 24.7% of total listings in New York
- The most relevant variables are related to locations, quality of room, and amenities

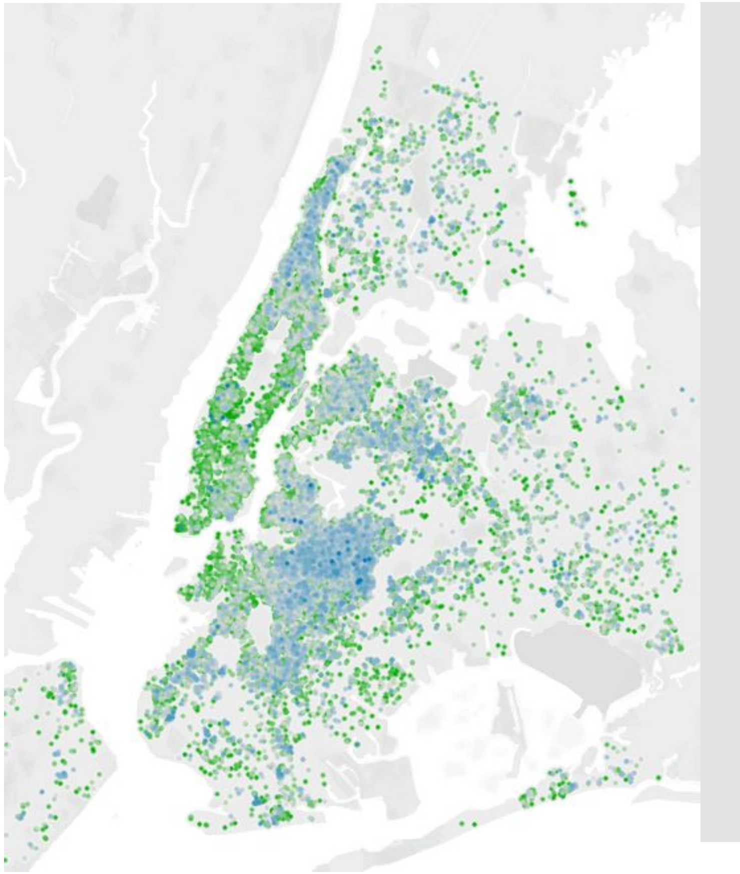
Price Distribution



Price by Neighborhood



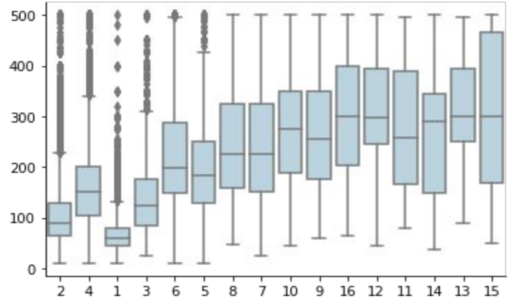
Price Distribution



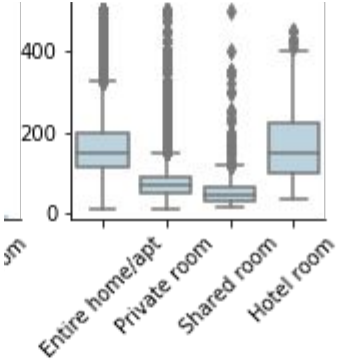
Exploratory Analysis

- The services offered is more relevant than guest rating and review count

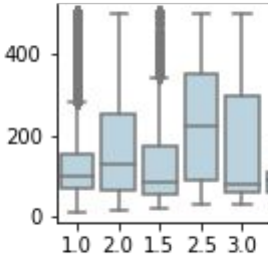
# of Guests



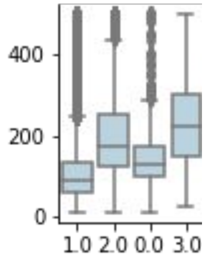
Price by Room Type



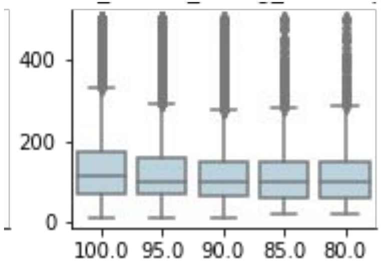
Bathroom



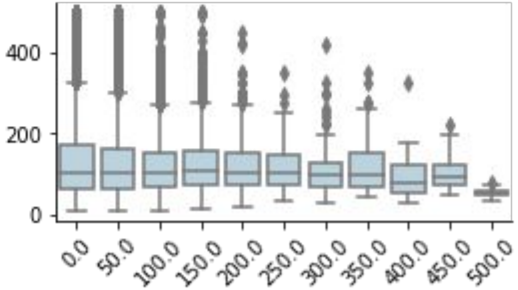
Bedrooms



User Ratings

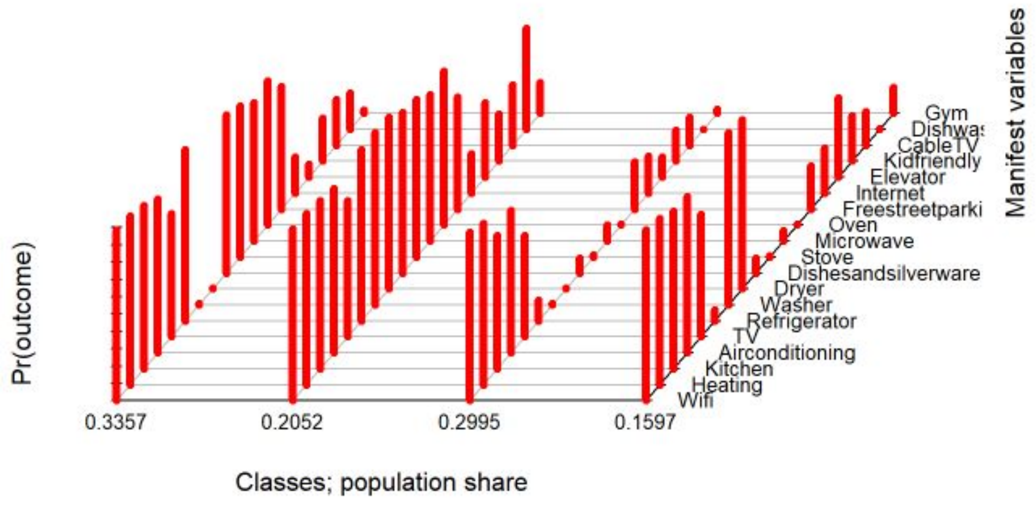


Historical Number of Reviews



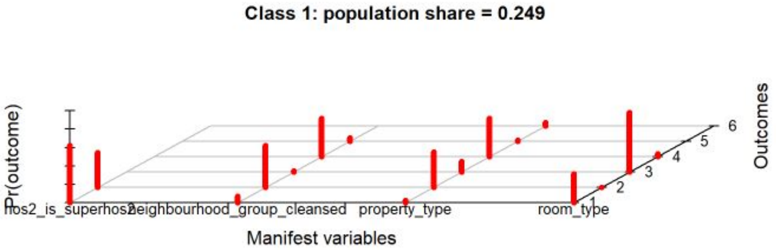
LCA on Amenities

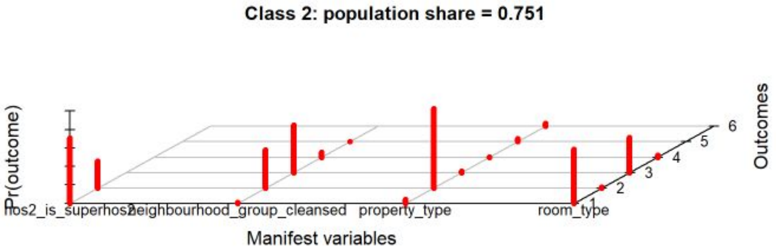
- Selected the 20 most frequent amenities across listings as well as amenities with the greatest effect on price.
- Optimal clustering at 4 clusters with relatively even population share across all classes.
- Cluster Descriptions:
    - Cluster 1 (33.57%): listings with all amenities available except washer and dryer.
    - Cluster 2 (20.52%): listings with all amenities available.
    - Cluster 3 (29.95%): listings that are private rooms with no additional amenities.
    - Cluster 4 (15.97%): listings with all amenities except kitchen related amenities.

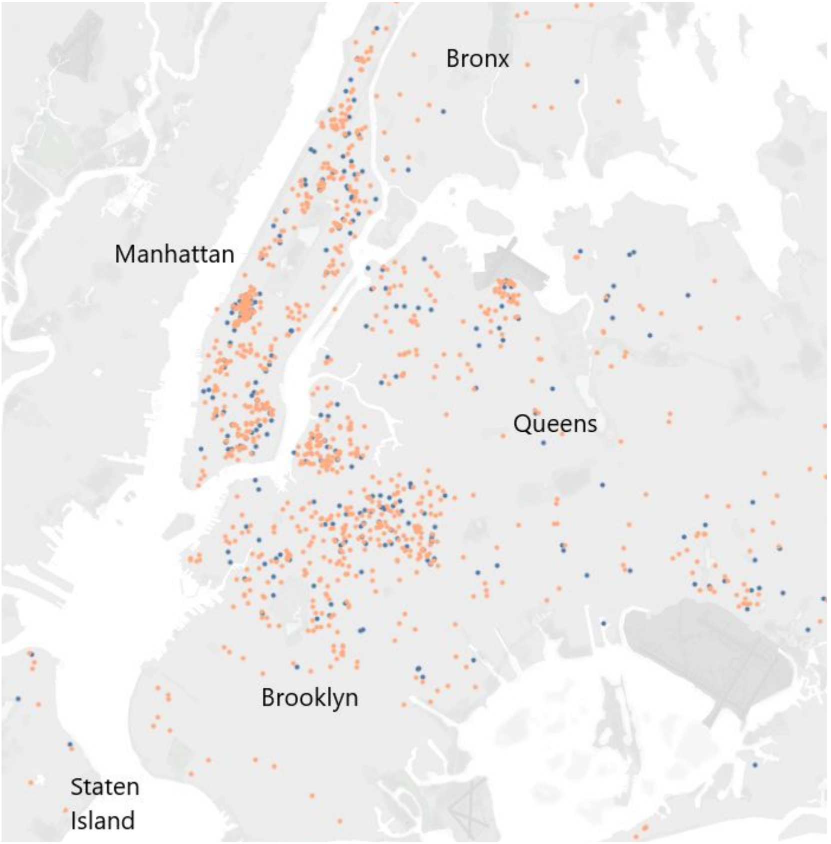


LCA on Property Type and Location

- Selected neighborhood, property type, and room type.
- Cluster Description:
  - Cluster 1 (24.9%): Even distribution of houses and apartments in Queens and Manhattan.
  - Cluster 2 (75.1%): Primarily apartments in Manhattan and Brooklyn





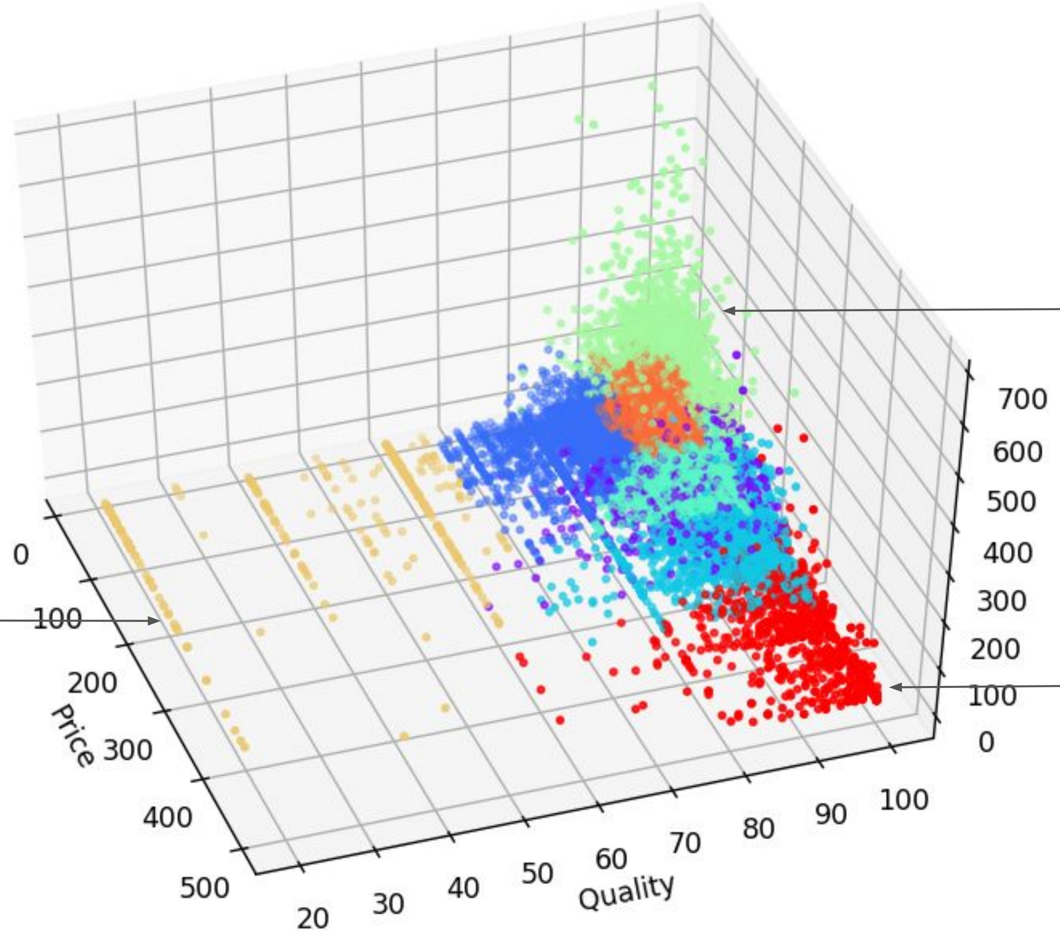


K-means Clustering

- Clustered listings based on the following variables: price, number of bedrooms, number of reviews, overall rating, number of guests.
- 8 clusters explaining ~85% of variance

Cluster	Price	Size	Quality	Occupancy	Number (proportion)
1	Average	Big	Above avg	Average	1746 (6.2%)
2	Very cheap	Small	Below avg	Very low	3150 (11.1%)
3	Very high	Average	Above avg	Very low	2520 (8.9%)
4	Average	Small	Above avg	Very low	6486 (22.9%)
5	Very cheap	Small	Above avg	Extremely high	2412 (8.5%)
6	Very cheap	Small	Terrible	Almost nil	389 (1.4%)
7	Cheapest	Smallest	Above avg	Very low	10623 (37.6%)
8	Ultra luxury	Huge	Above avg	Very low	942 (3.3%)

K-means Clustering



Cluster 5 (very cheap, good quality, highest occupancy)

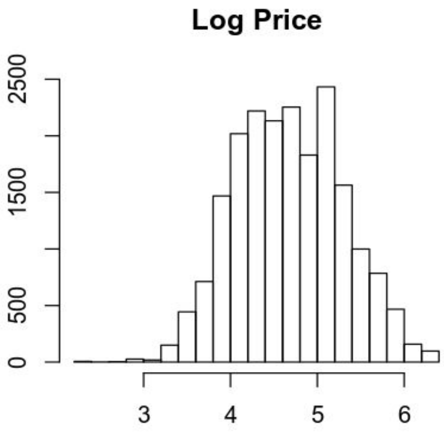
Cluster 6 (very cheap, worst quality, almost no occupancy)

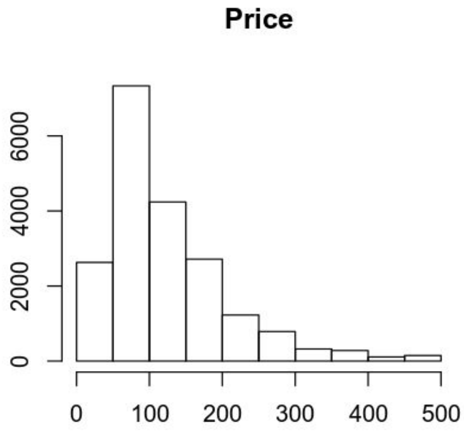
Cluster 8 (luxury, good quality, low occupancy)

Supervised Modeling

Linear Modeling

- Price was converted to logprice to stabilize variance
- Initial model with 11 variables was heavily overfitted and showed signs of multicollinearity.
- Final model: logprice ~ accommodates + zipcode + room_type





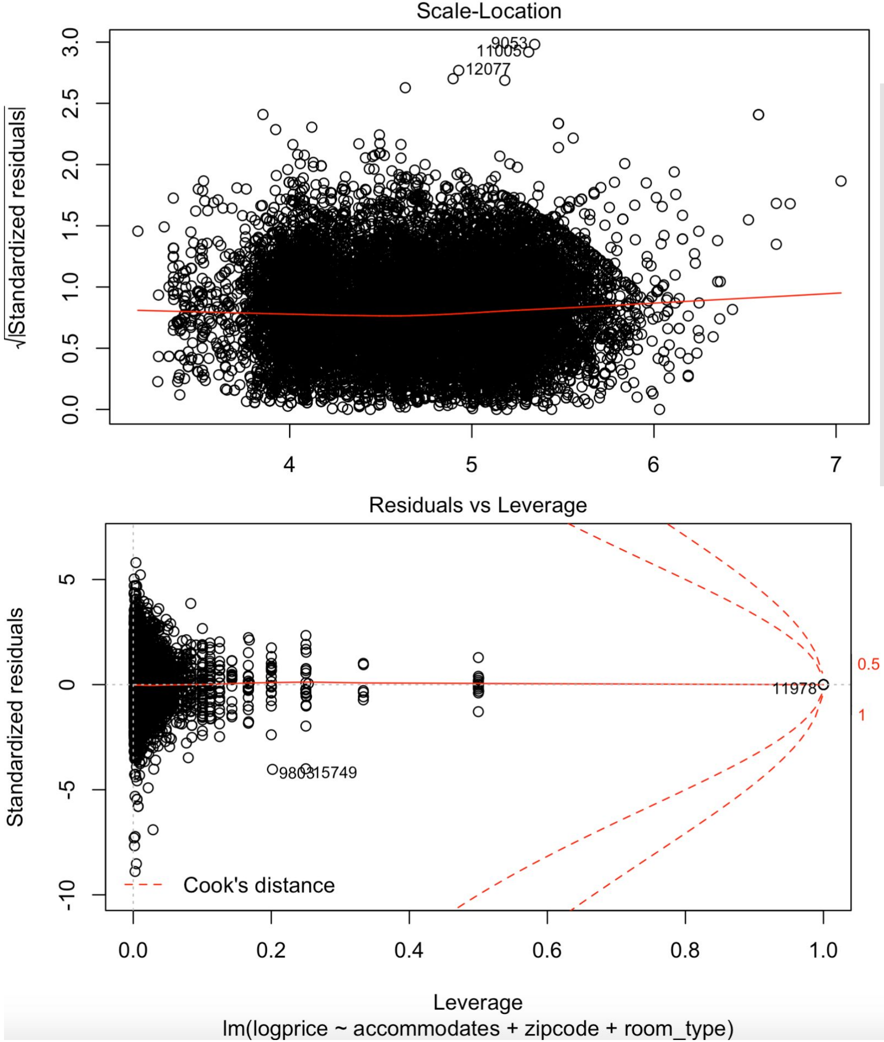
Model Performance

	R2
Train	0.59
Test	0.46

Linear Model Residuals

- The scale-location plot shows the square root of the standardized residuals as a function of the fitted values. The plot shows no obvious trend in this plot.

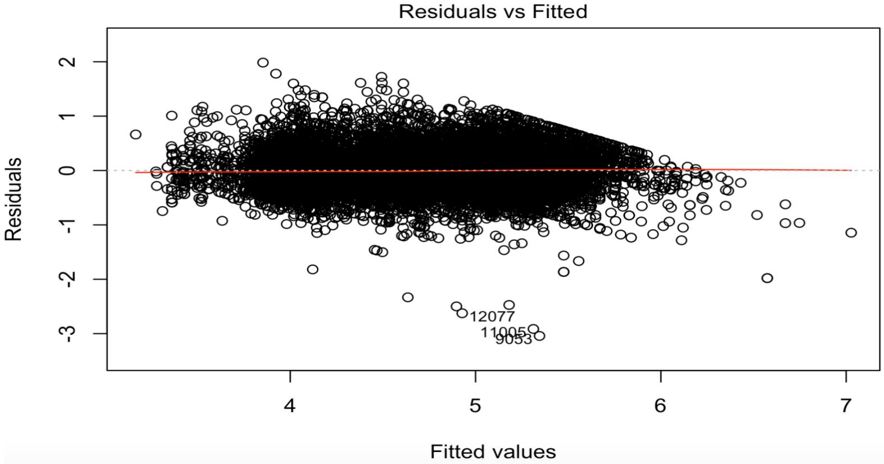
- The residual vs leverage plot shows that the smaller distances means that removing the observation has little effect on the regression results.

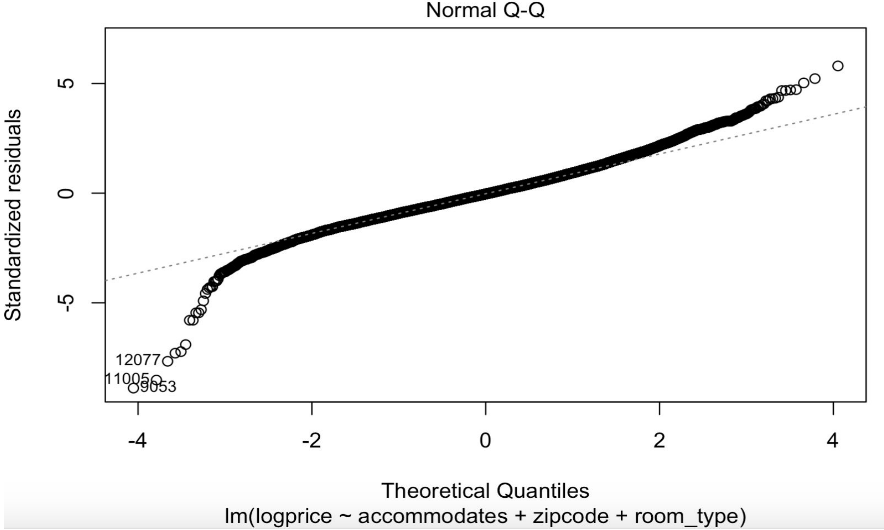


Linear Model Residuals

- The residual errors versus their fitted values show that the residuals are randomly distributed around the horizontal line representing a residual error of zero.

- The standard Q-Q plot shows the residual errors are normally distributed.





Conclusion

- The linear model has an R squared of 0.59 on the training data, but does not perform well in holdout and drops to 0.46. As a result, the Mean Absolute Percentage Error of the model is 52%, whereas the desirable MAPE has to be less than 20%.

- Therefore, we can make this model further rigorous by using more advanced models.

Model Performance

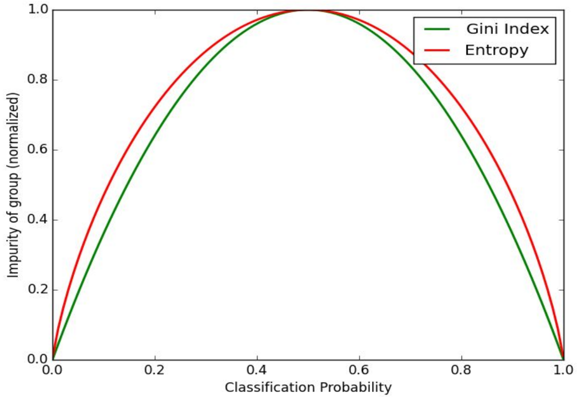
	Initial model	**Final model**
Train	0.72	**0.59**
Test	0.13	**0.46**

Decision Tree

Model Performance (R^2)

	Gini Index	Entropy
Train	0.653	0.653
C.V.	0.601	0.594
Test	**0.604**	0.599

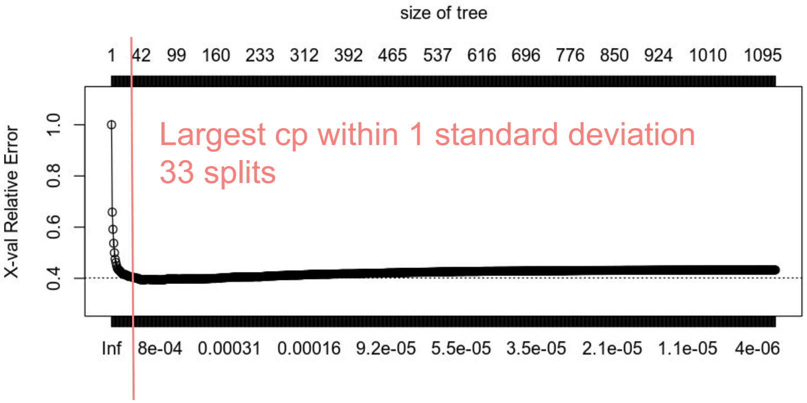
Scaled Gini Index v.s. Entropy



- Gini Index as the loss function
  - Similar performance on the train set; the Gini index stands out on the test set
  - Entropy for classification, Gini for regression
- Conservative selection of the complexity parameter
- 9 features used out of 98
- Advantages of rpart
  - Easy parameter tuning with cp (complexity parameter); printcp presents the threshold cp to avoid heuristic search for the best set of parameters
  - No dummification required as in sklearn
  - Feature selection not required (but could benefit the model) to reach decent performance

Decision Tree

Complexity Parameter Tuning

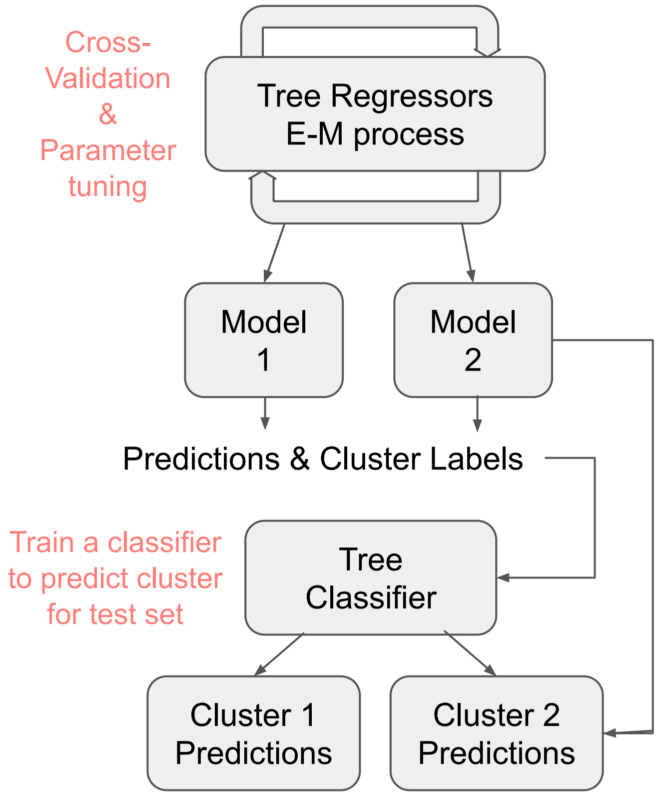


Feature Importance

1. Room_type
2. Accommodates
3. Neighbourhood_cleansed
4. Guests_included
5. Host_neighbourhood
6. Bedrooms
7. Bathrooms
8. Host_location
9. Property_type

- Gini Index as the loss function
  - Similar performance on the train set; the Gini index stands out on the test set
  - Entropy for classification, Gini for regression
- Conservative selection of the complexity parameter
- 9 features used out of 98
- Advantages of rpart
  - Easy parameter tuning with cp (complexity parameter); printcp presents the threshold cp to avoid heuristic search for the best set of parameters
  - No dummification required as in sklearn
  - Feature selection not required (but could benefit the model) to reach decent performance

Cluster-wise Decision Tree



Model Performance

	Class 1 R2	Class 2 R2	**Overall R2**	Class Acc.
Train	0.877	0.753	**0.839**	0.978
Test	0.741	0.661	**0.734**	0.948

- The model performance depends on...
  - Performance of the multi-class classifiers
  - Discrepancy among predictive models of different clusters
    - R2 is -0.79 for the 5% misclassified samples
- Strengths
  - Highly effective
  - Insights into clusters of study subjects
- Weaknesses:
  - Prolonged training time
  - Painstaking parameter tuning and feature selection

Random Forest

- Full model trained with all predictors, ntrees = 1000
- Hyperparameters using GridSearch with cross-validation.
  - Ntrees = 1250, max depth = 60, min sample split = 5, min samples per leaf = 2
- Further improvements were possible by subsetting the data again to include only those listings with >5 reviews in the last 12 months.
- High training R2 values are an indicator that RandomForest is quite prone to overfitting.

Model Performance

	Initial Model	GridSearch Xval	Reviews >5
Train	0.957	0.923	**0.961**
Test	0.654	0.661	**0.694**
Avg. Error	$32.01	$31.41	**$30.22**

Random Forest - Variable Importances

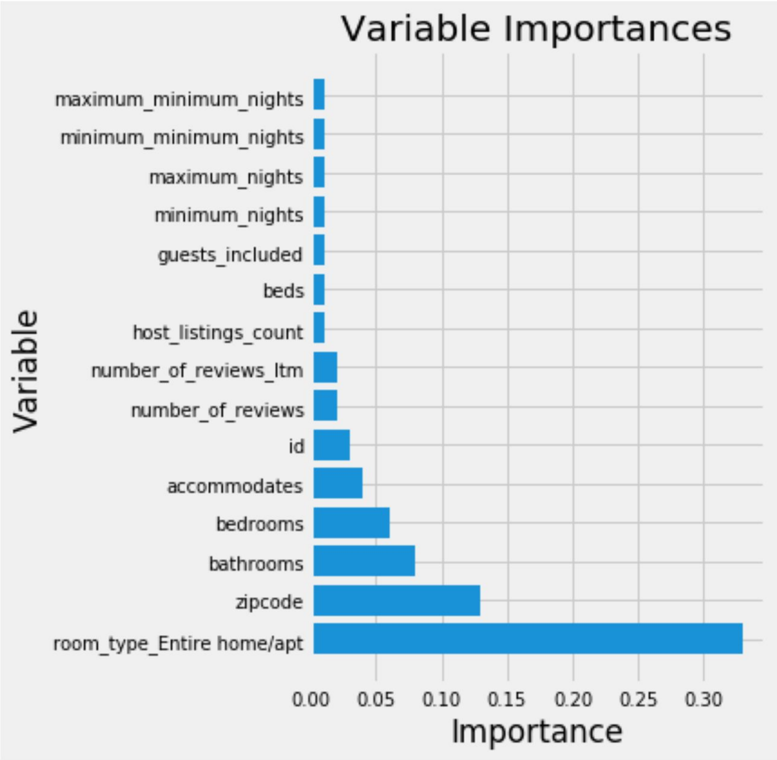
Checking for variable importances in the RandomForest model, we find some interesting results:

- Room type (entire apartment), zipcode, bathrooms and bedrooms are the most important variables
- Number of reviews are important, confirming our initial intuition
- Dishwasher and Dryer are the only important amenities

Retraining the model with just the most important variables, we get a slight drop in R2 and accuracy:

Test R2: 0.652, Accuracy: 71.2%, Avg. Error: $32.68

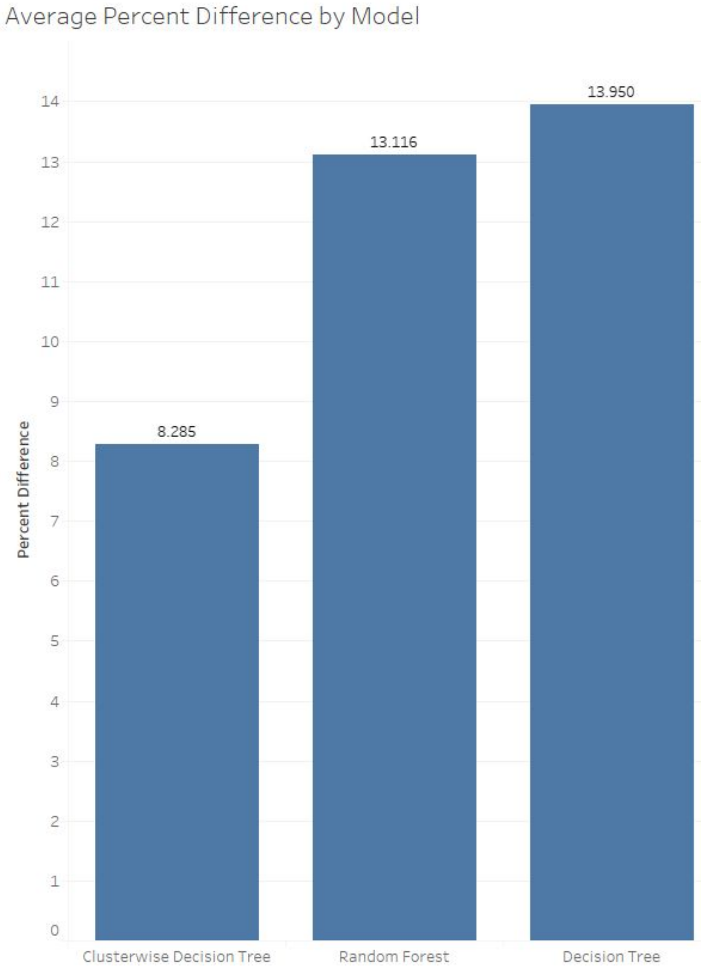
But we have a more interpretable model



Residual Analysis

- Analyzed the percent difference between actual price and predicted price on holdout set across all models.

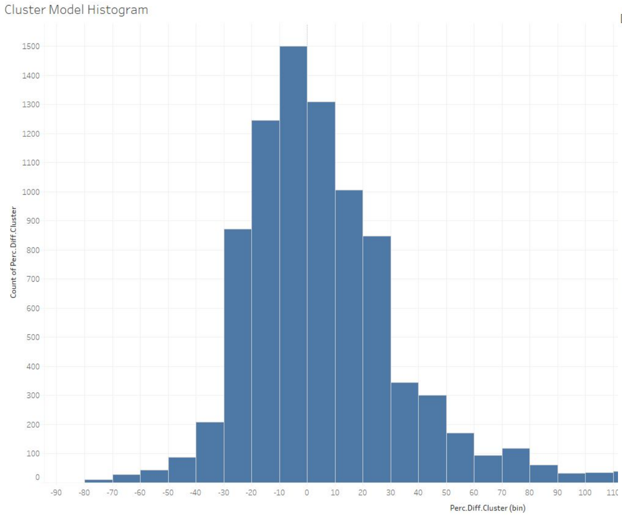
- Cluster-wise Decision Trees returned the smallest percent difference.
  - Cluster-wise Decision Tree: 8.285%
  - Random Forest: 13.116%
  - Decision Tree: 13.950%



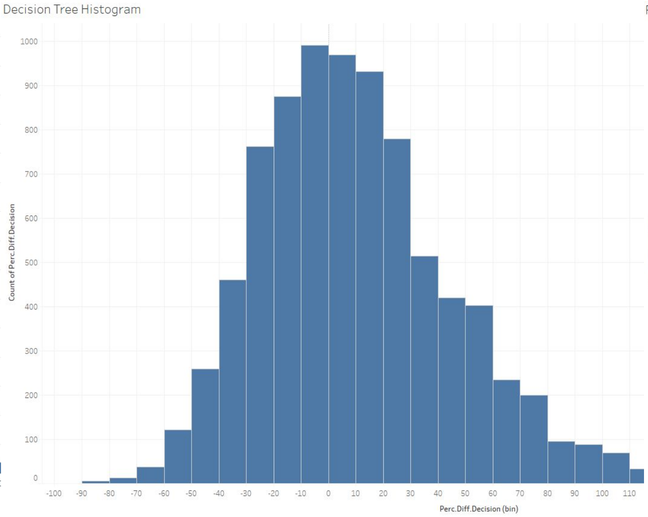
Residual Analysis

- Residuals across all models were normally distributed.

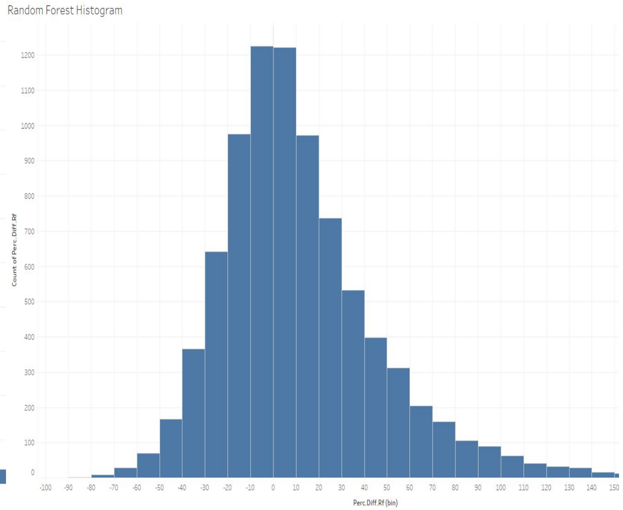
Cluster-wise Decision Tree



Decision Tree

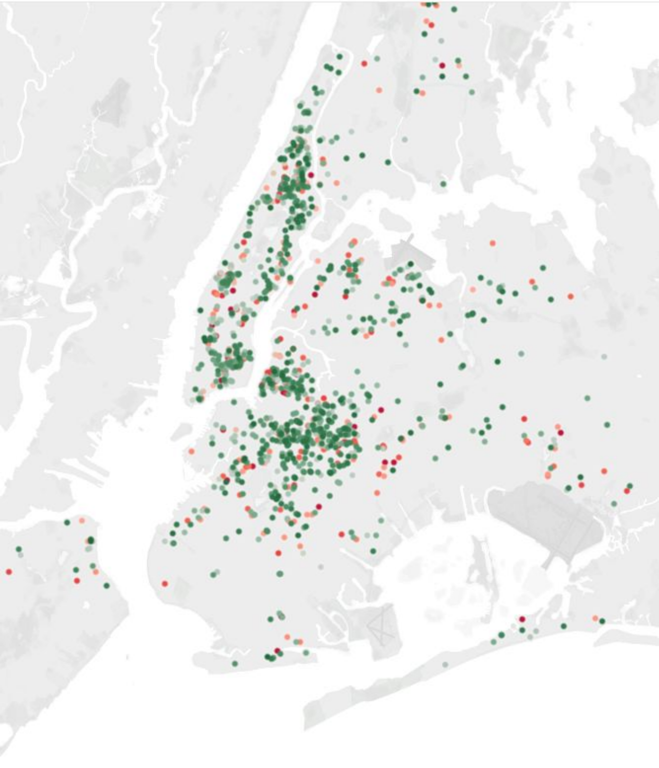


Random Forest

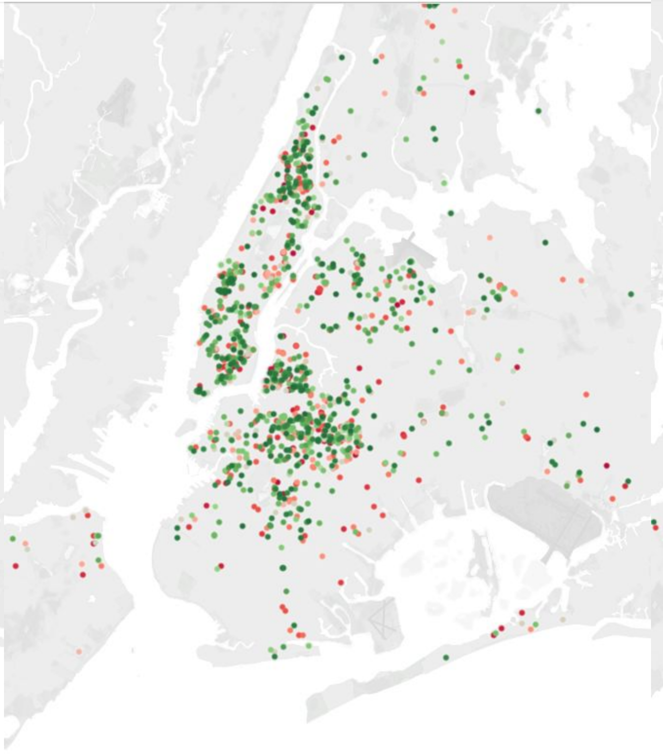


Heat Map of Overpredicted Prices

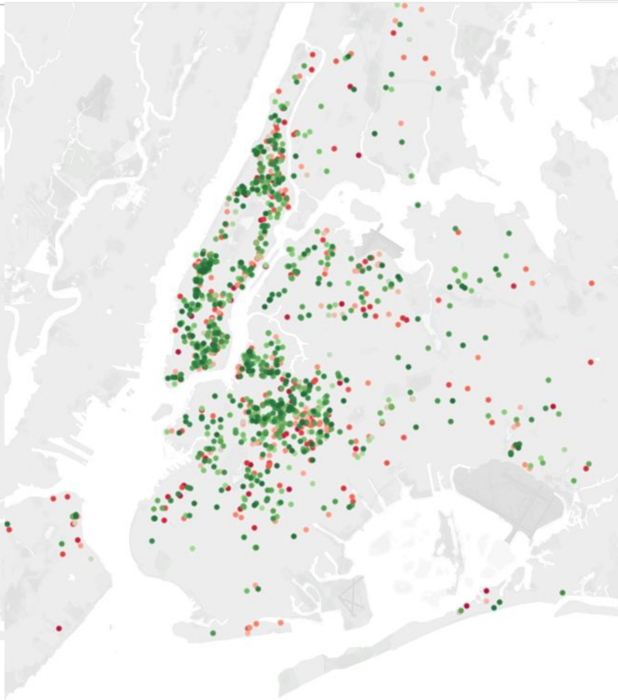
Cluster-wise Decision Tree



Decision Tree

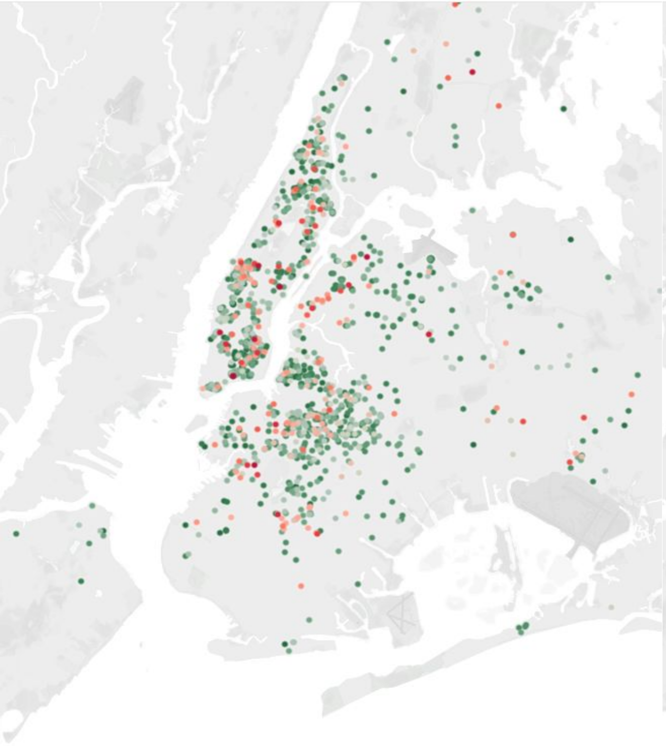


Random Forest

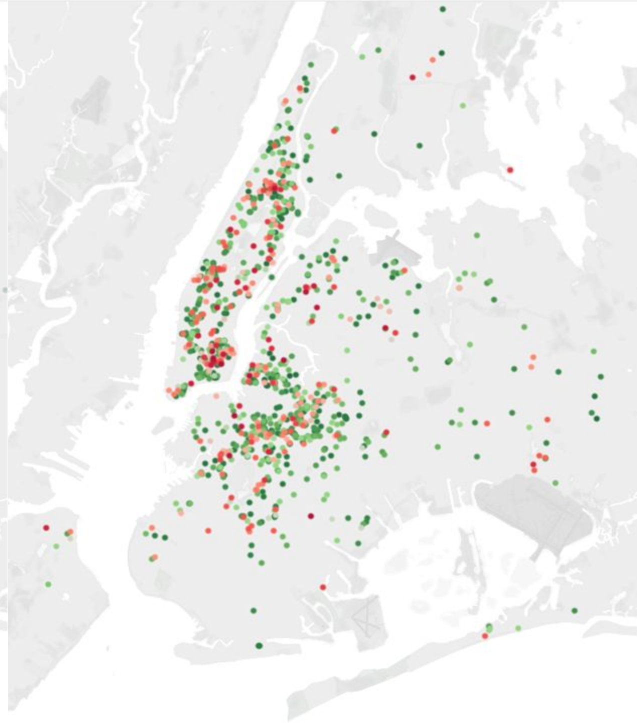


Heat Map of Underpredicted Prices

Cluster-wise Decision Tree



Decision Tree



Random Forest

Key takeaways

Model comparisons and summaries

Linear Model

Avg. Error: $47.21

Test R2: 0.46

- Poor results overall
- Too simplistic to capture the relationship between price and the predictors
- Prone to overfitting

Decision Trees

Avg. Error: $36.07

Test R2: 0.604

- Heavily overfitted
- Train/Test split impacts accuracy greatly

Cluster-wise Decision Trees

Avg. Error: $25.75

Test R2: 0.734

- Best R2
- Relatively stable between train and test results
- Performance dependent on cluster classification

Random Forest

Avg. Error: $31.61

Test R2: 0.661

- Decent performance
- Shows signs of overfitting

Key insights

1. Offer the entire apartment, instead of a private room
2. Always ask guests to leave a review
3. Offer a washer and dryer
4. Offer more beds within your listing

Limitations & Improvement

Limitations

- Predicts a fixed price instead of a seasonal/special price (a concert in the vicinity presents an opportunity for a spike)
- Does not take duration of stay into account (prediction only for a single night)
- Not accurate for super luxury listings (price > $500 excluded)
- On an average, prediction is off by $30
- External validity: probably inaccurate for smaller cities with limited training data
- Price prediction is for the whole of NYC, not neighbourhood-wise

Improvement

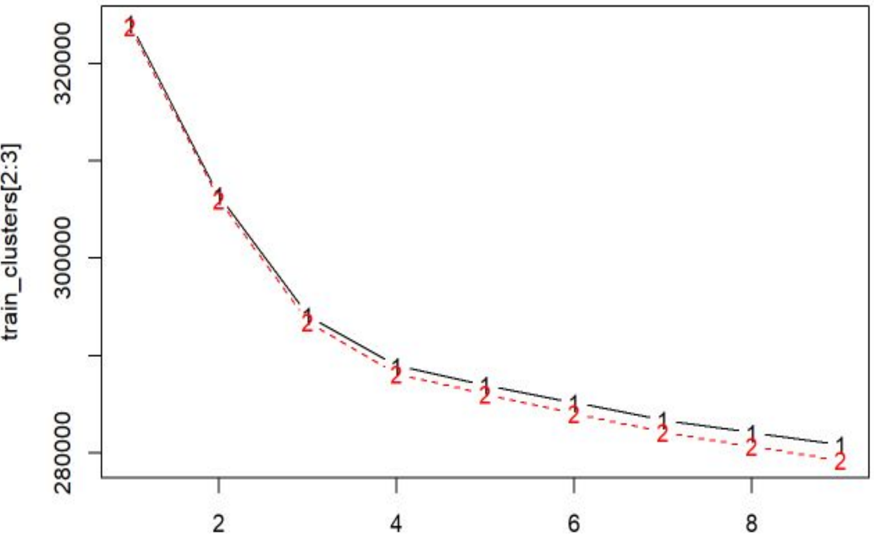
- Give a range of price, instead of a precise number
- Possible model improvements - feature selection in decision tree model, more clusters in cluster-wise decision tree model
- Neighbourhood-wise prediction instead of city-wise prediction (problem of limited data)

Appendix

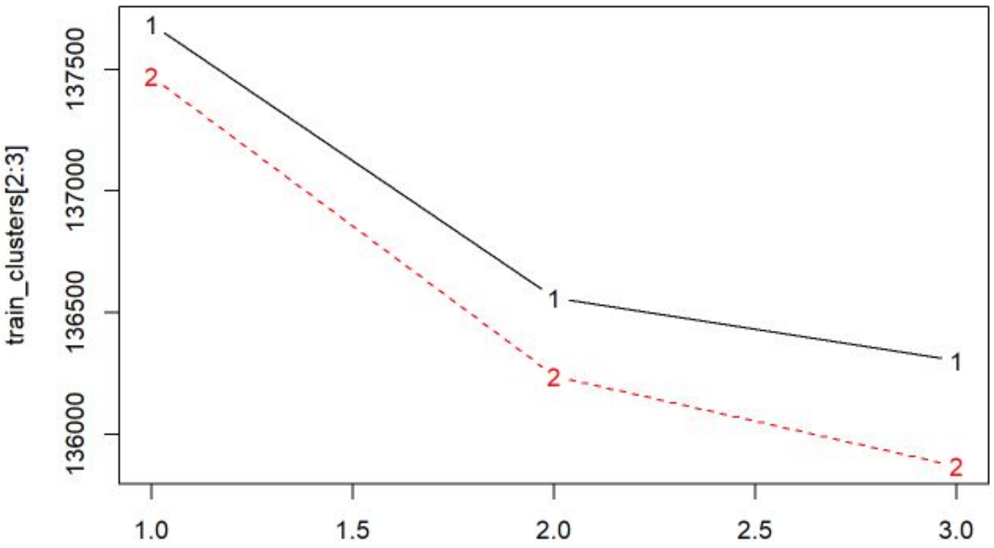
LCA Scree plots

- LCA AIC/BIC Plots

Amenities



Property Type



Regression Model Diagnostics

```
## Evaluate collinearity
vif(LinearModelFinal)
##                    GVIF  Df GVIF^(1/(2*Df))
## accommodates  1.409156   1        1.187079
## zipcode       1.305534 195        1.000684
## room_type     1.660918   3        1.088240
```

The result of vif () shows the GVIF value of each attribute is less than 5 which means the multicollinearity does not exist.