Sophie Ngo, Ting Chen, Michael Peeler
CSC 272 – Data Mining Final Project Writeup
December 2, 2022
Dr. Kevin Treu

The Price is Right: Estimating Room Rates in Various Major Cities

Introduction

What do we want to do? Predict Airbnb prices. Why do we want to do this? Those who want to list their property as an Airbnb may be able to find the most appropriate nightly price, or perhaps those who are wanting to travel may want to know which types of properties yield the most affordable, yet exciting experiences. Having an accurate and predictive model of pricing based on room types, locations, and review frequency would give someone unfamiliar with the market a much better understanding of if the price they are setting or paying is fair.

Our motivation is using the Airbnb data from NYC to build a model and generalize Airbnb prices in Chicago and Washington DC.

Since we are trying to predict prices, we of course used numeric estimation for our project. The key findings of our results are that using a variety of linear regression and ensemble learning techniques, we were able to achieve fairly high levels of correlation with the expected pricing data. We conclude that prices can be reasonably predicted using data mining, but also that finding the most effective data source for this task was more nuanced than we expected.

Data Attributes

Attribute	Description
neighbourhood_group * † : *Nominal*	Indicates the region of a city that a rental is in; only includes data in New York City, where it indicates which of the 5 boroughs the rental is in. 1) Brooklyn 2) Manhattan 3) Queens 4) Bronx 5) Staten Island
neighbourhood ** † ” : *Nominal*	Indicates the neighborhood which the rental is located; 244 different options in New York City, 39 different options in D.C. and no options for Chicago.
latitude † : *Numeric*	Latitude of the rental; positive indicates degrees North, negative degrees South.
longitude † : *Numeric*	Longitude of the rental; positive indicates degrees East, negative degrees West.
room_type : *Nominal*	Indicates whether the room was a private room, an entire home/apartment, a shared room, or a hotel room.

number_of_reviews : *Numeric*	Count of total number of reviews received on the rental.
reviews_per_month : *Numeric*	Average number of reviews per month for the rental since it was first listed.
calculated_host_listing_count: *Numeric*	How many rental listings in the geographic area had the same host when the data was collected.
availability_365: *Numeric*	The number of days, in the past year, this rental has been available to rent.
number_of_reviews_ltm : *Numeric*	The number of reviews the rental has received in the last 12 months.
days_since_last_review : *Numeric*	The time since the last review was given to the rental.
dist_from_airport_in_miles : *Numeric*	The distance in miles to the nearest large or medium-sized airport.
dist_from_subway_in_miles : *Numeric*	The distance in miles to the nearest station of the local subway.
dist_from_city_center : *Numeric*	The distance of the rental from a central point in the city; Times Square in New York City, the Loop in Chicago, and the Capitol in D.C.
price: class *Numeric*	The price per night, in USD, of the rental.

* Only in New York City dataset; ** Not in Chicago Dataset; † Excluded from cross-city datasets; " For some algorithms, the NYC's 244 separate nominal values are too many to handle, and this attribute must be excluded.

Table 1. Original Cleaned Dataset Attributes

Data Preparation

While looking for data, we found information from Airbnb locations in multiple major cities across the United States which had multiple characteristics that thought may be predictive (Table 1). We already know of many factors that may influence Airbnb prices. This, of course, includes property type (e.g., apartment, condo, house), availability, number of reviews, etc. This type of information can be easily gathered from our data source. However, we hypothesized that an Airbnb's distance from the nearest metro station and airport may also affect price because a shorter distance may be more convenient, or imply closeness to the city center, where tourist attractions and fun activities are more common.

We decided we needed to add three extra attributes to our Airbnb dataset: distance from nearest airport (in miles), distance from nearest metro station (in miles), and distance from the city center (in miles). For our data on airports, we found a dataset containing locations for every

airport in North America, from major international hubs to small local airstrips. In our analysis, we limited comparisons to airports identified as either “large”- or “medium”-sized in the dataset; “large” included hubs like Chicago O’Hare and Los Angeles International, and “medium” included smaller airports like Greenville-Spartanburg, which seemed to be a good range of sizes to include. Subway locations were slightly harder to find, but we found location data for three cities—New York City, Chicago, and Washington, D.C.—and so decided to focus our research on those cities. Our source for the data scrapped the New York City data from Airbnb on September 7th of 2022, and the Washington, D.C. and Chicago data on September 14th of 2022, so any attributes relative to time are in relation to those dates.

For each city, we also selected a “city center”, based on locations commonly held to be important for the city. For New York City, we selected Times Square for its cultural and business importance; for Washington, D.C. we chose the U.S. Capital Building, because of the impact it has on the layout of D.C. and its importance for both tourism and governmental purposes; for Chicago, we selected the Loop, the home of Chicago’s downtown and business districts. New York City data also had to be parsed from a GEOM object-style string.

To determine the distances, we looped through all location values in each of the sets of values (subways, airports, and city centers) and selected the minimum distance. Distance was calculated using a modified Euclidean distance formula. While one degree of latitude is always approximately 69 miles, because of the spherical shape of the Earth the distance of one degree longitude shrinks as latitude increases. At 40 degrees latitude, which all our city centers were within 1.1 degrees latitude of, one degree of longitude is approximately 53 miles. Typically, when calculating distance on a sphere, one would use the Haversine formula, which works well with the longitude-latitude system. However, for optimization reasons in our attempt to use R scripts to calculate the nearest of 470 subway stations for nearly 40,000 rental locations, we opted not to implement the Haversine formula, but instead approximate the distance by multiplying the difference in longitude by 69 miles and the difference in latitude by (due to an arithmetic mistake) 54.6 miles. Though this mistake was found too late in the process to do anything about, since it was consistent across all data it would not affect any of our conclusions in any notable way; further, even with the error, the calculated distance was lower than a hundredth of a mile for distances even as large as 8 miles, compared to the calculation one would get with the Haversine formula. Thus, we remain confident in our data, though the correct value of 53 has been corrected in the codebase if anyone were to run it again.

Across our 3 datasets, New York City had the most instances with 39,881, Chicago had 7,414, and D.C. had 6,473. There was a total of 872 large or medium airports, though the full dataset included 29,519. New York City had 473 subway stations, Chicago had 626 entrances to 145 stations, and Washington, D.C. had 91 stations. After this processing, and the removal of all non-categorical nominal attributes, we ended up with a set of 15 potentially predictive attributes for our models to use (Table 1), though one of those attributes (neighborhood_group) only had data for New York City, and Chicago was missing data for a second attribute (neighborhood).

Using Weka, we created a randomized 70-30 split into training and testing files, respectively, for each city. In cross-city testing, we eliminated 4 attributes (neighborhood_group,

neighborhood, latitude, and longitude) based on their specificity to the original city. To be able to both make models applicable across cities and create models using all available attributes, each dataset was further split into a full dataset and a "cross_city" dataset, which removed the incomparable attributes. This meant for each of the 3 cities, we ended up with 6 datasets. In creating models for each city, we used both the full and "cross_city" versions of the training data to create two models specific to a city, and the respective testing dataset to test each model. However, to compare the models against each other, we tested them using the full dataset of cities that the model was not built upon.

While conducting analysis, we discovered an extended version of our original dataset, which contained all the previously mentioned data, with an additional 23 attributes that could be used in modeling, relating to the person who created the listing for the rental, the average ratings for various review statuses for the rental, the data regarding the maximum and minimum length of a stay at the rental, the current availability status, and the number of people the rental accommodates (see Table 2). This data was treated in the same way, being broken into testing and training sets with copies for cross-city comparisons and all attribute available model building.

Attribute	**Description**
host_is_superhost: *Nominal*	True or False. Indicates whether the host is a member of the Airbnb "super host" program, based on various criteria such as a minimum number of stays and a high review rate.
host_listing_count : *Numeric*	How many rentals the host has listed.
host_has_profile_pic : *Nominal*	True or False. Whether or not the host's Airbnb account has a profile picture.
host_identity_verified : *Nominal*	True or False. Whether or not the host's Airbnb account has had their purported identity verified.
accommodates: *Numeric*	The maximum capacity of the rental.
bedrooms: *Numeric*	The number of bedrooms in the rental.
beds: *Numeric*	The number of beds in the rental.
minimum_nights : *Numeric*	The minimum length of stay, in days, that a renter could use the rental.
maximum_nights : *Numeric*	The maximum length of stay, in days, that a renter could use the rental.
has_availability : *Nominal*	True or False. Whether or not a room was available when the data was collected.
avilability_30: *Numeric*	How many of the 30 days before the data was collected that the room was available for rental.

avilability_60: *Numeric*	How many of the 60 days before the data was collected that the room was available for rental.
avilability_90: *Numeric*	How many of the 90 days before the data was collected that the room was available for rental.
number_of_reviews_l30d: *Numeric*	The number of reviews the rental has received in the 30 days before the data was collected.
review_scores_rating: *Numeric*	The average review score of the rental.
review_scores_accuracy : *Numeric*	The average review score of how accurate the rental's listing was.
review_scores_cleanliness: *Numeric*	The average review score of the how clean the rental was.
review_scores_checkin: *Numeric*	The average review score of how well check-in went.
review_scores_communicaiton: *Numeric*	The average review score of how well the rental's host communicated with the renter.
review_scores_location: *Numeric*	The average review score of how well renters rated the location of the rental.
instant_bookable: *Nominal*	True or False. Whether or not a room can be booked for immediate use.
calculated_host_listing_count_ entire_homes: *Numeric*	How many rental listings classified as "entire home" in the geographic area had the same host when the data was collected.
calculated_host_listing_count_ private_rooms: *Numeric*	How many rental listings classified as "private rooms" in the geographic area had the same host when the data was collected.
calculated_host_listing_count_ Shared_rooms: *Numeric*	How many rental listings classified as "shared rooms" in the geographic area had the same host when the data was collected.

Table 2. Additional attributes of extended dataset.

Data Visualization

Since New York City had the largest dataset, we decided to conduct pre-modeling analysis to identify patterns in the data which may indicate attributes conductive to accurate modeling. One topic we thought might be interesting was the distribution of room types. We first sought to find how room types were distributed between accommodation sizes, and as Figure 1 demonstrates, for all rentals with an accommodation larger than one, entire home/apartment rentals dominate all other forms of rooms. Private rooms are more common at the two extremes of the scale, while hotel rooms exclusively accommodated less than 6 people. There are also very few hotel rooms in general, perhaps because Airbnb likes to style itself as a competitor to the hotel industry.

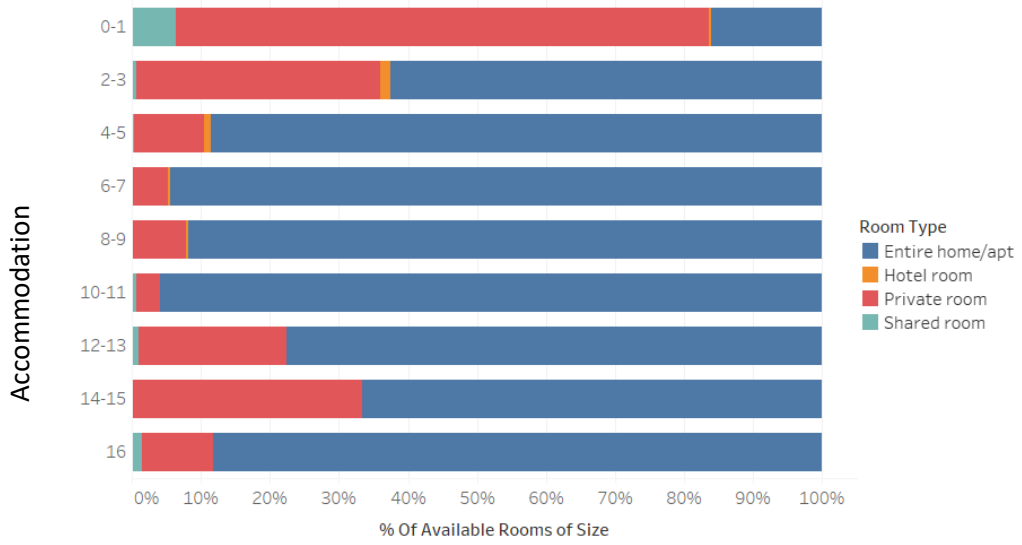

Figure 1. The distribution of room types by accommodation size in New York City.

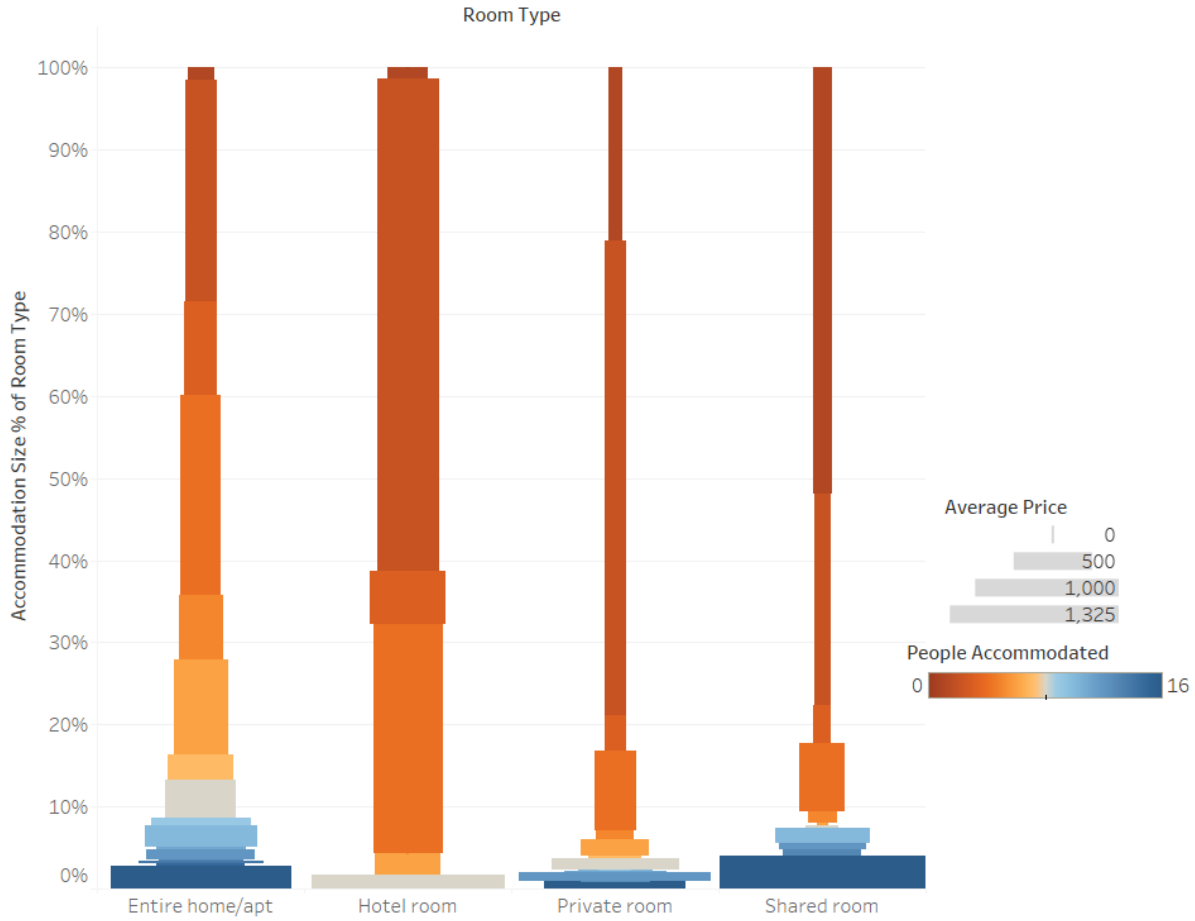

Figure 2. Room type by accommodation size and price in New York City.

We then moved on to looking for relations between accommodation size, room type, and price. As Figure 2 demonstrates, in all cases, as accommodation increased, price does as well. Figure 2 also shows the extent to which small-accommodation homes dominate the market, as hotels contain no availabilities greater than 9 people. Hotel rooms are also much more expensive than other room types with the same accommodation size, though their price tends to remain constant across various accommodation sizes, while the prices of other room types prices increase as accommodation increases. Further, we see that small-accommodation rooms dominate the market and are far more frequent than mid-sized or large rooms; rooms with an accommodation level below 4 make up most of rental availabilities for all room types except whole-house rentals.

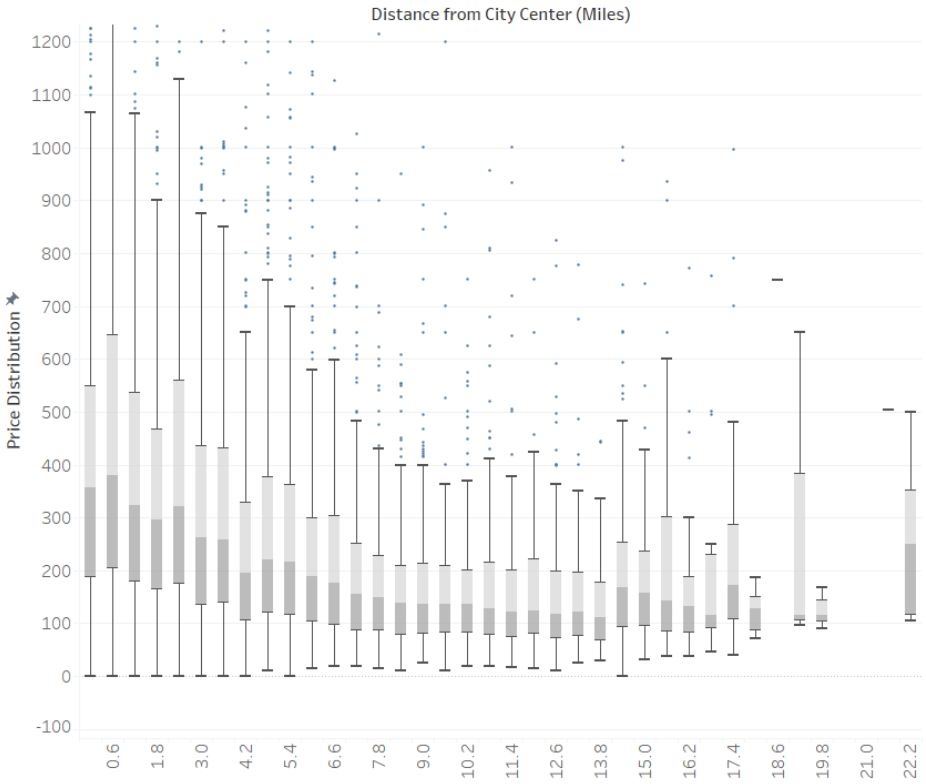

Figure 3. Distribution of prices by distance to city center in New York City.

Next, we looked for evidence of our hypothesis regarding proximity to key locations like the city center. From Figure 3, we can see that, in New York City at least, the average prices of Airbnb rentals decrease as the distance to the city center in Time Square increases. We hypothesized calculating the distance between the Airbnb and the city center would be predictive for the nightly fee and the graph shows a clear negative correlation between the distance and the average price, which confirms our hypothesis. This visualization suggests to us that Linear Regression modeling may be quite good with this attribute, since price seems to follow such a stark linearly decreasing trend in distance from city center.

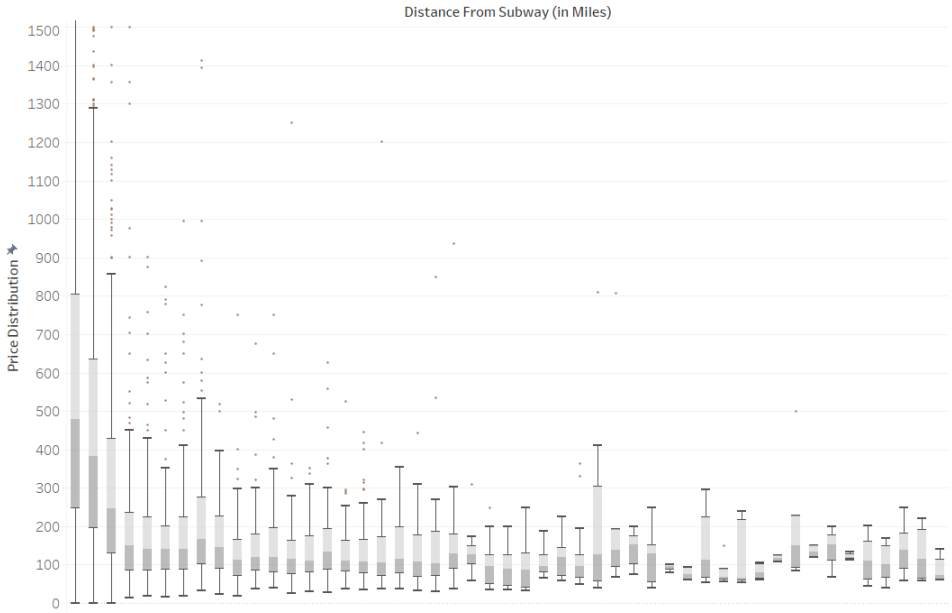

Figure 4. Distribution of prices by distance to subways in New York City.

From Figure 4, we can see the closest to the subway, the price was highest. As the distance between the Airbnb and the subway increased, the price of the Airbnb decreased within 1.2 miles. The mean price for Airbnb located with distance greater than 2 miles stays consistent between 50 to 150 dollars. Although there are many outliers in this figure, they also show a negative correlation between distance to subway and price of Airbnb. For Airbnb located 4 miles or further away do not have clear correlation with the price. We expect this is because we do not have enough data for those Airbnb and distance from the closest subway would not be predictive for Airbnb located outside of any effect radius. The effects seen this this table may be an effect of the distance to the city center as well; as seen in Figure 5, where color represents the distance to the city center and size represents the distance to the nearest subway station. It is extremely noticeable how small each of the light-colored circles at the center are compared to the dark-colored circles on the edges of the figure. This makes sense because most of the subways in New York City are centered around the city center and traveling further from a city center would mean you are further from a subway station.

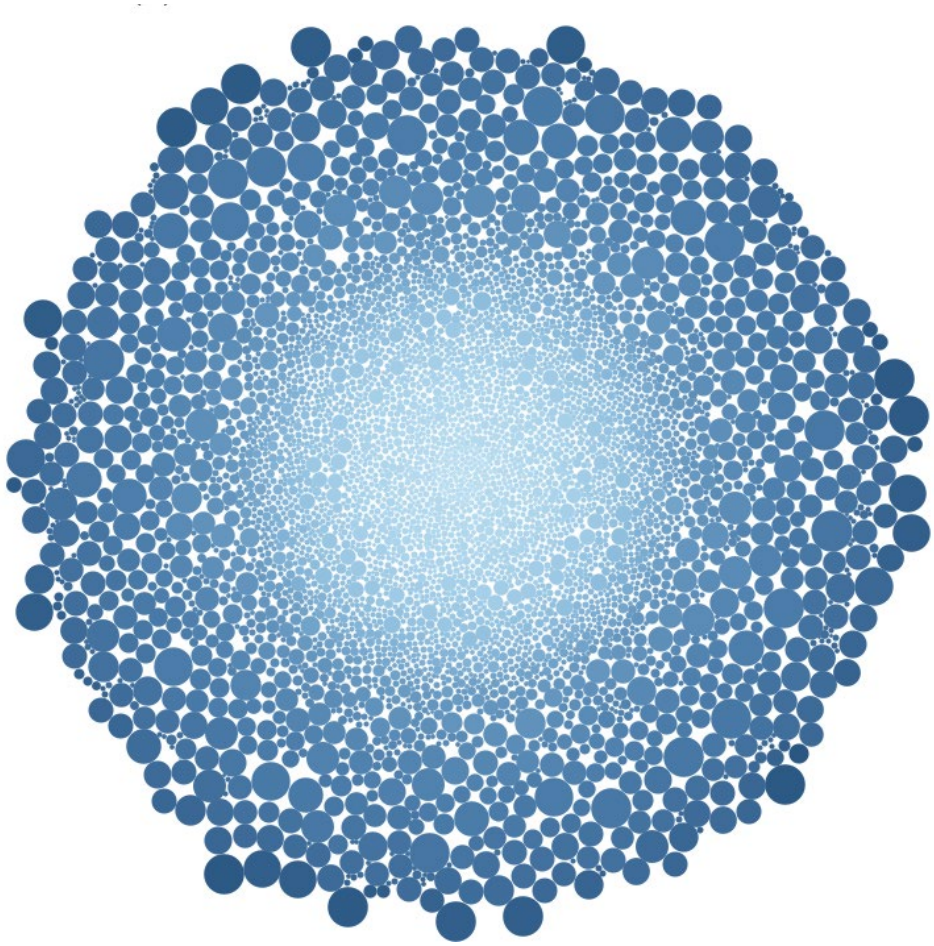

Figure 5. Relation between distance to the city center and distance to the nearest subway station, where lighter circles are closer to the city center and smaller circles are closer to the nearest subway station.

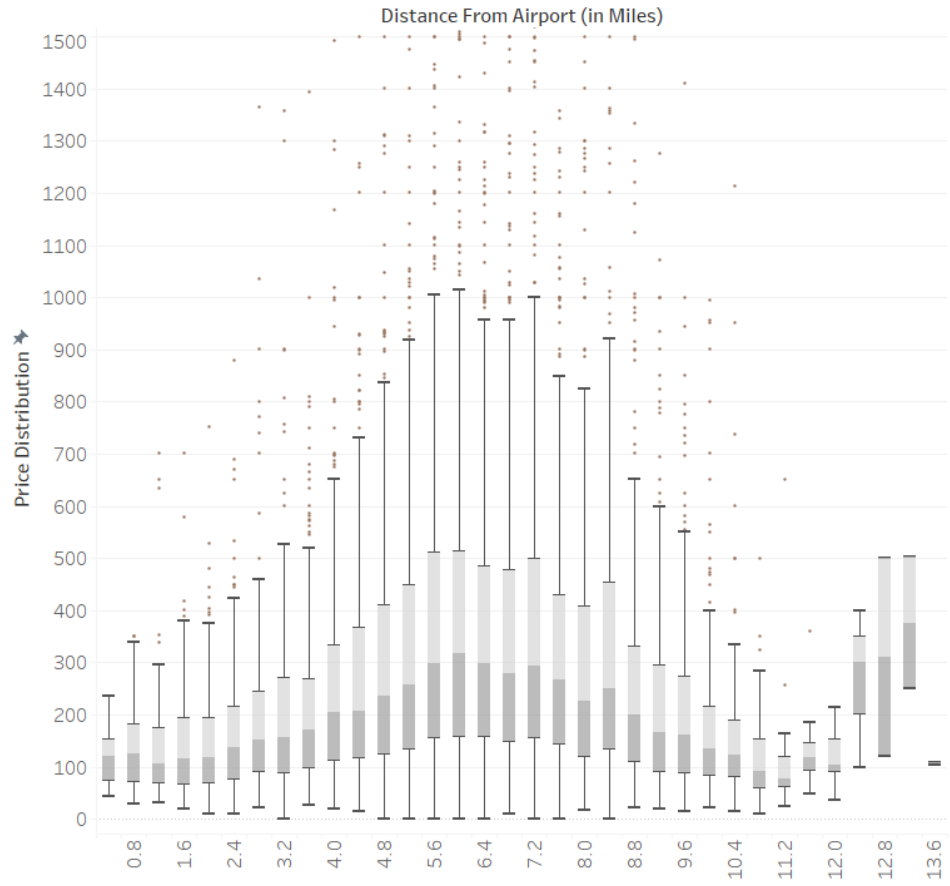

Figure 6. Distribution of prices by distance to airports in New York City.

Figure 6 displays the distance from each Airbnb in New York City to the nearest airport and the distance is between 0 to 13.6 miles. It is interesting that the mean distance around 6 miles has the highest mean price for each night, and the prices fall out in both directions from that distance. It is also quite telling that the as-the-crow-flies distance from Le Guardia Airport to Times Square is approximately 6.1 miles, meaning that the peak of the average pricing falls directly on the city center.

Data Analysis

Once we had prepared our data, we had to decide which learning algorithms would be most applicable for our purposes. Of course, the only ones we were able to use were numeric estimation algorithms, of which we chose four. A description of each is briefly given below.

Nearest-Neighbor (IBk)

This algorithm is one of the simpler algorithms that we used. It works by taking each instance in the dataset and giving it a “position” in space, based on the values that it has. When you give the algorithm a new instance it has never seen before, it gives it a position just like before, then compares it against all the other instances. Out of these, it will then try to find the

closest point to the new point (this is the “nearest neighbor”) by calculating the distance between the two positions. The algorithm will estimate the new instance with the class value of the nearest neighbor. If desired, the number of neighbors can be increased, and doing so will allow the model to estimate the average of the class values.

We experiment with this model because we know that it works for both classification and numeric estimation, and it could be used as a baseline for evaluating the effectiveness of other numeric estimation algorithms. When we experiment deeper into setting the KNN value, number of neighbors to find and take the average of. Since our dataset contains large of number of instances (>7000 instances), we experiment with large of neighbors to find.

REP Tree

An algorithm that builds a regression or decision tree. This tree is created by branching out attributes with the most information gain. The tree reduces overfitting by pruning itself, using a pruning method called reduced-error pruning (REP). It works by percolating bottom-up through the tree and attempting to replace each node with its most popular class as a leaf, effectively pruning the subtree. If the accuracy did not go down, then the change is kept. The advantage of REP Tree is that it is fast and simple compared to other pruning algorithms.

First, we were testing all the tree algorithms that are available in Weka, REP tree happens to perform well on the original dataset with 15 attributes, including the extra 3 distances attributes that we calculated. We hypothesized that the REP Tree algorithm would also be relatively accurate in the extended dataset with 38 attributes.

Random Forest

Random Forest works by first creating a random subset of the attributes, then builds a decision tree from each one. It can be used for classification or regression. When used for regression, a new instance is assigned the average of the output for all the trees it has built. This can allow for less error since the forest is spread over more decision trees, and therefore any flaw in an individual decision tree is much less detrimental.

We are interested in using this ensemble learning method because other available trees for numeric estimation perform better (higher correlation coefficient) than the nearest neighbor algorithm which is our baseline model. We think building multiple tree models would minimize the flaws that each tree has, therefore improving the overall accuracy measured in correlation coefficients and mean absolute error.

Linear Regression

Linear Regression builds a line of best fit through the data. The purpose of this is to see if there exists a linear relationship between the class attribute and all other attributes. When the algorithm is run, the result is a formula that represents the line of best fit. We can tell how strong

the correlation is by looking at the correlation coefficient: a value between 0 and 1, with 0 being no correlation and 1 being a perfect linear relationship. This serves as an "accuracy" measure of our model. A new instance is predicted by inputting all the values of numeric attributes into the formula and selected values of the nominal attributes included in the formula, then assigning it the result of the calculation.

Linear Regression is one of the most used algorithms for numeric estimation in this class and it is the simplest to understand. We decided to build a model to determine its effectiveness for our dataset. It makes sense that for many of the attributes for the Airbnb dataset to have a linear relationship with the price. For example, as the number of accommodates increases the price would also increase since the space would be larger. Another possible negative correlation could be that as the distance to the city center decreased, the price will increase. Based on these hypotheses, we included Linear Regression as one of the models.

Vote

Vote is another ensemble learning algorithm. It works by creating models from a collection of learning algorithms. Then, when given a new instance, the Vote algorithm will apply it to each model that it has built. The output for each model is then averaged, and this average is used as the final value to assign to the new instance. In our case, we use the above four algorithms (IBk, REP Tree, Random Forest, and Linear Regression) for the classifiers used in the Vote model.

From the previous four algorithms we used, the results were all unsatisfactory, specifically, the correlation coefficients were less than 0.5, meaning the accuracy is lower than 50%. We wanted to combine the best of the models to improve accuracy, therefore we attempted to build an ensemble learning model using Vote algorithm embedded in another ensemble learning model (Random Forest model).

MultiScheme

This algorithm takes in multiple different classifiers and, using folds of the training data, determines when the output of each model should be preferred. This selection is done by minimizing the mean-squared error. In our analysis, we used the same four algorithms as in the Vote model, with 5 folds used to judge performance of the internal classifiers. This will let us build a more tailored ensemble function that Vote would, as it considers previous rates of success in the training data.

Results

Since we used two major data sources (original and extended), we compared them to explore why the models built from the original dataset with 14 attributes far underperformed the models built from the extended dataset with 38 attributes. The table below summarizes the performance of the above 5 models we built from the original dataset with 15 attributes.

	New York City		Washington, D.C.		Chicago	
	Corr.	MAE	Corr.	MAE	Corr.	MAE
Neighbor (K=5)	0.2356	144.38	0.088	125.05	0.2115	154.05
Random Forest	0.2796	150.39	0.085	139.87	0.3311	131.54
REP Tree	0.1522	131.45	0.0924	103.24	0.1815	162.42
Lin. Reg.	0.2967	107.94	0.1525	106.47	0.2667	133.23
Vote *	0.2893	127.13	0.1315	113.00	0.2887	127.18

* The Vote model was built using the above four algorithms (nearest-neighbor, random forest, REP tree, and linear regression).

Table 3. Comparison of differing model accuracies for full original dataset by city.

To find better-performing models, we created numerous models for each city on the original dataset. Table 3 shows the results of building models using the five different algorithms, then testing them using city-specific testing data, repeated for each city. The full versions of the datasets were used to train the models because we aimed to create the most comprehensive version of the model for the data. Since we are seeking to predict the cost of an Airbnb location more accurately, the lower values of mean absolute error (MAE) indicate much stronger performance for what we are selecting for. Linear regressions, across the board, seem to have a much stronger performance in this regard than any of the other models, with MAEs towards the bottom of ranges in each of the three cities. However, Random Forest had good success for the Chicago model. The Vote model was not particularly strong, though based on that, we sought to optimize on the original linear regression model to see if we could construct stronger versions of the model that perform well, both across cities (when models are trained and tested on the same city) and between cities (when models are trained on one city and tested on another).

Operating on the extended dataset, we used a cross-validation parameter selection method to restrict the number of parameters, and then generated linear regression models for each of the cities.

Table 4a - New York City Model

Summary	New York City *	Chicago	D.C.
Correlation	0.5156	0.5089	0.3671
MAE	118.65	190.88	179.99
RMSE	286.29	293.23	296.87
RAE	86.96 %	152.29 %	163.55 %
RRSE	85.29 %	101.20 %	109.16 %

Table 4b - Chicago Model

Summary	New York City	Chicago *	D.C.
Correlation	0.0011	0.5012	0.0155
MAE	1337.10	107.09	139.30
RMSE	239005.35	266.12	2774.256
RAE	998.60 %	76.90 %	118.54 %

RRSE	67641.37 %	86.11 %	1017.36 %

Table 4c – Washington, D.C. Model

Summary	New York City	Chicago	D.C. *
Correlation	0.3104	0.4809	0.5122
MAE	111.9983	111.7221	94.9767
RMSE	338.1516	253.3523	218.9686
RAE	88.8044 %	90.0148 %	92.1966 %
RRSE	95.5858 %	87.2876 %	87.2018 %

* The city a model was trained on.
MAE = Mean Absolute Error, RMSE = Root Mean Squared Eror, RAE = Relative Squared Error, RRSE = Root Relative Square Error

Table 4. Performance of model accuracies across cities using extended cross-city datasets.

As seen in Table 4, models built using the extended datasets had much-improved levels of correlation, compared to the models demonstrated in Table 3, though they did not necessarily have lower MAE values. Models built for other cities performed exceptionally well on Chicago data, and even rivaled the model built specifically from Chicago data. The Chicago mode, meanwhile, was only predictive of the Chicago dataset, and was extremely ill-suited to predicting values in New York City or Washington, D.C.

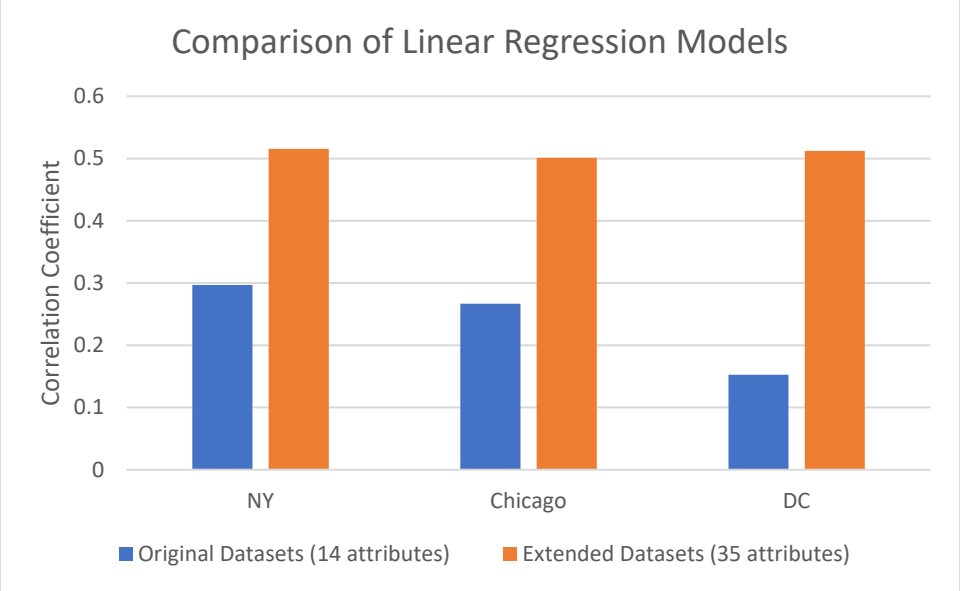

Figure 7. A side-by-side bar chart showing the difference between the correlation coefficients of the linear regression model that was built from the original datasets, seen in Table 3, vs the one built from the extended datasets, seen in Table 4. For clarification, the correlation coefficient used for the extended datasets is the one obtained from testing the model using 70% split. This is to match how the numbers in Table 3 were obtained, as they were also tested using a 70% split.

As seen in Figure 7, there is no question that the linear regression models built using the extended datasets far outperform the linear regression models built using our datasets. The bar chart puts into perspective the drastic difference in correlation coefficients between the two datasets. This leads us to conclude that the extended datasets are a better choice for attempting to predict prices, and therefore gave us enough reason to continue our research using only the extended datasets.

However, it is also worth noting the disparities in the predicative capacities of the created datasets. While the extended datasets were roughly equal in their predicative capability between cities, the original datasets had high levels of variation, with the D.C. dataset being the least predictive of the price. This would imply that the New York City pricing is much better predicted by the attributes included in the original dataset than the D.C. data, and pricing predictivity acts differently in the two cities.

We decided to try building the linear regression model again, but this time with feature selection. This is because we believed that reducing noise in the form of non-predictive attributes may increase the performance of our results. Utilizing Weka's wrapper subset evaluation, conducting a greedy stepwise search with a linear regression evaluator, we determined that 13 attributes should be removed from the data. The removed attributes were:

1) host_has_profile_pic
2) host_identity_verified
3) room_type
4) beds
5) maximum_nights
6) availability_90
7) availability_365
8) review_scores_rating
9) review_scores_accuracy
10) review_scores_communication
11) review_scores_location
12) review_scores_value
13) calculated_host_listings_count_private_rooms

	New York City *		Chicago		D.C.	
	Full	Subset	Full	Subset	Full	Subset
Correlation	0.5156	0.5162	0.5089	0.5098	0.3671	0.3676
MAE	118.65	118.77	190.88	191.03	179.99	180.38
RMSE	286.29	286.09	293.23	293.56	296.87	297.36
RAE	86.96 %	87.05 %	152.29 %	152.41%	163.55 %	163.91 %
RRSE	85.29 %	85.23 %	101.20 %	101.31 %	109.16 %	109.37 %

* The city the model was trained in.

Table 5. Comparison between Linear Regression model described in Table 4a, called the "full" model, and the model built using a subset which excluded the above-mentioned 13 attributes.

Using the remaining 22 attributes, we generated a Linear Regression model from the expanded New York City cross-city dataset, and tested it on each of the three cities, as described in Table 5. This model demonstrated a performance almost identical to the model described in Table 4a, with differences so marginal that they do not bear worthy of mention. This indicates the excluded attributes had no significant benefit to the model and were not helpful in predicting the price of the Airbnb rentals. It serves as a validation for the subset evaluator's potential benefits to removing noise while having very low risk of ruining the existing model. It should be noted that this does not mean the attributes removed could not be predictive; but if they are, their predictive capability is fully covered by some other attribute.

	New York City – MultiScheme		
	New York City *	Chicago	D.C.
Correlation	0.5556	0.5224	0.3269
MAE	98.73	197.38	218.00
RMSE	278.34	318.83	348.18

* The city the model was trained on.

Table 6. Performance of meta-selected model using extended cross-city datasets.

Finally, we sought to do a multi-scheme classifier using the four types of models we had been using, those being the IBk, REP Tree, Random Forest, and Linear Regression algorithms. We generated a model using the New York City training data, and ran the analysis with the data from each of the other cities, resulting in Table 6. This proved to be the best model we had created throughout the entire process, with correlation coefficients higher than any previous model and mean absolute errors lower than any model as well. The correlation coefficient improved on the NYC test data and the full Chicago dataset compared to using a subset of attributes. The models remain very predictive for Chicago, though the D.C. data remains elusive. Although both Chicago dataset and D.C. dataset had a higher mean absolute error and root mean squared error than the New York City data, it was acceptable because we used the New York City training data, and our goal is to generalize to the other two cities. This final model truly shows the power of ensemble learning, as it was well ahead of every other model we had generated to that point.

Conclusion

Our aim for this project was to see if we could use Airbnb data to predict Airbnb's nightly fee. We would consider us to have achieved this goal using the extended dataset, but not with the original dataset with less attributes. The linear regression models built upon the extended datasets had a substantial increase in average correlation coefficients compared to the linear regression models built upon original datasets, therefore we can conclude that the extended dataset we found was more appropriate to explore the goal of predicting price. However, it is also worth noting how well the predictivity of the original dataset was for New York City and Chicago, especially compared to Washington, D.C. where even a model trained on data from the district was not able to predict price nearly as well as many of the models, we would go on to create from data in other cities.

For testing the model using Airbnb data from the other two cities, we focused on building the model using the New York City Airbnb data and tested with Airbnb data from Chicago and Washington D.C. Since New York City has the largest instances of Airbnb, we thought it was a better suited dataset for building an accurate model. When testing the model with Chicago dataset, the result was very similar to the accuracy of testing data from NYC. From the similar accuracy result, we can conclude our model generalized well to the Airbnb data from the other two cities.

Data Source

Original datasets were found at these following sites:

- Airports - data.humdata.org/dataset/ourairports-usa
- Airbnb - http://insideairbnb.com/
- DC Metro - https://opendata.dc.gov/datasets/DCGIS::metro-stations-regional/explore?location=38.903376%2C-76.740656%2C10.00
- Chicago Metro - https://github.com/ChicagoCityscape/gis-data/blob/master/stations_metra/metra_entrances.geojson
- New York City Metro - https://catalog.data.gov/dataset/subway-station

Location data for city center latitude and longitudes was obtained from Google Maps.

Appendix

All code and datasets can be found at https://github.com/m-peeler/DataminingFinalProject.

Linear Regression Models built using New York City training data in Table 4a:

price =

```
 16.0604 * host_is_superhost=t +
  0.0464 * host_listings_count +
  0.0717 * host_total_listings_count +
 39.2538 * room_type=Shared room,Entire home/apt,Hotel room +
-25.0309 * room_type=Entire home/apt,Hotel room +
 59.6165 * room_type=Hotel room +
 35.207  * accommodates +
 53.4856 * bedrooms +
 -0.2632 * minimum_nights +
-34.6018 * has_availability=t +
  6.0056 * availability_30 +
  0.8301 * availability_60 +
 -0.2661 * number_of_reviews +
 -0.7695 * number_of_reviews_ltm +
 26.627  * review_scores_cleanliness +
-39.2583 * review_scores_checkin +
```

20.42 * review_scores_location +
18.3005 * instant_bookable=t +
-2.7423 * calculated_host_listings_count +
2.0364 * calculated_host_listings_count_entire_homes +
2.5496 * calculated_host_listings_count_private_rooms +
-11.8869 * calculated_host_listings_count_shared_rooms +
11.1624 * reviews_per_month +
14.7946 * dist_from_subway_in_miles +
8.8422 * dist_from_airport_in_miles +
-22.7702 * dist_from_city_center +
18.683

Linear Regression Models built using Chicago training data in Table 4b:

price =

-0.0994 * host_listings_count +
-15.2982 * room_type=Entire home/apt +
12.8966 * accommodates +
63.542 * bedrooms +
25.1525 * beds +
-0.1991 * minimum_nights +
-0.0222 * maximum_nights +
-28.9121 * has_availability=t +
5.2163 * availability_30 +
-1.3373 * availability_60 +
0.2616 * availability_90 +
-0.1712 * number_of_reviews +
-5.2143 * number_of_reviews_l30d +
57.9512 * review_scores_cleanliness +
-36.1286 * review_scores_checkin +
25.4897 * review_scores_location +
-29.4017 * review_scores_value +
18.848 * instant_bookable=t +
0.7843 * calculated_host_listings_count +
-1.459 * calculated_host_listings_count_private_rooms +
-23.1329 * dist_from_subway_in_miles +
9.1437 * dist_from_airport_in_miles +
-15.9237 * dist_from_city_center +
-74.0992

Linear Regression Models built using Washington DC training data in Table 4c:

price =
0.045 * host_listings_count +
-0.0103 * host_total_listings_count +
47.8274 * room_type=Private room,Entire home/apt +
-15.6446 * room_type=Entire home/apt +
24.4659 * accommodates +
65.5869 * bedrooms +
94.0433 * has_availability=f +
1.9723 * availability_30 +
0.3081 * availability_60 +
-0.3563 * availability_90 +
0.0949 * availability_365 +
-0.1852 * number_of_reviews +
-1.1151 * number_of_reviews_ltm +
17.6158 * review_scores_rating +
34.4088 * review_scores_cleanliness +
102.1303 * review_scores_location +
-56.178 * review_scores_value +
-13.2632 * instant_bookable=f +
-0.6558 * calculated_host_listings_count +
0.2456 * calculated_host_listings_count_entire_homes +
-3.2588 * calculated_host_listings_count_shared_rooms +
14.086 * reviews_per_month +
-529.824