Claire Lundy, Claire Razanauskas, Katie Rudins
CSC 272
Dr. Treu
28 April 2025

Airbnb Analysis: A Deeper Look into the Contributors to Prices and Satisfaction

Introduction

Airbnb properties offer a wide range of experiences for guests, and the unique characteristics of each rental can influence the cost of the stay. As Airbnbs are growing in popularity compared to the traditional hotel option, we chose to investigate European Airbnb prices. The dataset we used was downloaded from Kaggle and included the weekday Airbnb prices for various European cities (The Devastator). We chose to integrate the datasets corresponding to Athens, Greece and Lisbon, Portugal. In addition, we found another dataset depicting TripAdvisor restaurant data for these two cities (Leone), which was integrated with the Airbnb data. The Airbnb dataset included attributes that described the characteristics of each Airbnb such as price, person capacity, distance from metro stations, distance from the city center, cleanliness, and guest satisfaction. The restaurant data included attributes detailing the type of cuisine, culinary awards received, and special dietary accommodations. After exploring our data, we were interested in predicting the price of the Airbnb, the price of the Airbnb per guest, guest satisfaction, and the price level of a given restaurant.

To investigate the above attributes, classification algorithms ZeroR, OneR, J48, PRISM, JRip (RIPPER), IBk (linear search, KD tree, and ball tree), and simple Naïve Bayes were used. Apriori was also used to observe potential associations between attributes. Three different attribute selection methods were utilized to maximize model accuracy.

Dataset Description

The datasets for the weekday Airbnb prices for Athens and Lisbon were downloaded from Kaggle (https://www.kaggle.com/datasets/thedevastator/airbnb-prices-in-european-cities?select=athens_weekdays.csv). The datasets contained attributes that described characteristics of the Airbnb properties and their relation to locations such as the city center, metro stations, restaurants, and popular attractions, which are detailed below in Table 1.

Table 1. Airbnb Dataset Attributes

Attribute Name	Description	Type
realSum	the full price of accommodation two nights in Euros	numeric
room_type	the type of the accommodation	nominal
room_shared	dummy variable for shared rooms	nominal
room_private	dummy variable for private rooms	nominal
person_capacity	the maximum number of guests	numeric
host_is_superhost	dummy variable for superhost status	nominal
multi	dummy variable if the listing belongs to hosts with 2-4 offers	nominal
biz	dummy variable if the listing belongs to hosts with more than 4 offers	nominal
cleanliness_rating	cleanliness rating for Airbnb	numeric
guest_satisfaction_overall	overall rating of the listing	numeric
bedrooms	number of bedrooms (0 for studios)	numeric
dist	distance from city center in km	numeric
metro_dist	distance from nearest metro station in km	numeric
attr_index	attraction index of the listing location	numeric
attr_index_norm	normalized attraction index (0-100)	numeric
rest_index	restaurant index of the listing location	numeric
rest_index_norm	normalized restaurant index (0-100)	numeric
lng	longitude of the listing location	numeric
lat	latitude of the listing location	numeric

The TripAdvisor restaurant dataset was also downloaded from Kaggle (https://www.kaggle.com/datasets/stefanoleone992/tripadvisor-european-restaurants). This data was scraped from TripAdvisor, and the dataset included over one million instances of restaurants with forty-two different attributes such as cuisine type, number of reviews, awards, hours of operation, and average rating, among others. These attributes are described in Table 2.

Table 2. Tripadvisor Restaurant Dataset Attributes

Attribute Name	Description	Type
Restaurant Link	TripAdvisor restaurant link	nominal
Restaurant Name	name of the restaurant on TripAdvisor	nominal
Original Location	original location displayed on TripAdvisor	nominal
Country	country name retrieved from original_location	nominal
Region	region name retrieved from original_location	nominal
Province	province name retrieved from original_location	nominal
City	city name retrieved from original_location	nominal
Address	address displayed on TripAdvisor	nominal
Latitude	latitude coordinate	numeric
Longitude	longitude coordinate	numeric
Claimed	notes if restaurant business is “claimed” on TripAdvisor	nominal
Awards	award names	nominal
Popularity Detailed	popularity detailed ranking	nominal
Popularity Generic	popularity generic ranking (among all places to eat in the area	nominal
Top Tags	top tag names	nominal
Price Level	level of prices in currency (Euros)	nominal
Price Range	more specific range of prices in currency (Euros)	nominal
Meals	types of meals	nominal
Cuisines	types of cuisines	nominal
Special Diets	types of special diets	nominal
Features	restaurant features (takeout, reservations, outdoor seating)	nominal
Vegetarian Friendly	if the restaurant is vegetarian friendly (Y/N)	nominal
Vegan Options	if the restaurant offers vegan options (Y/N)	nominal
Gluten Free	If the restaurant offers gluten-free options (Y/N)	nominal
Original Open Hours	original open hours on TripAdvisor	nominal
Open Days Per Week	number of open days per week retrieved from original_open_hours	numeric
Open Hours Per Week	number of open hours per week retrieved from original_open_hours	numeric

Working Shifts Per Week	number of working shifts days per week retrieved from original_open_hours	numeric
Average Rating	average restaurant rating	numeric
Total Reviews Count	total reviews count	numeric
Default Language	default language displayed while scraping	nominal
Reviews Count in Default Language	total reviews count in default language	numeric
Excellent	excellent reviews count in default language	numeric
Very Good	very good reviews count in default language	numeric
Average	average reviews count in default language	numeric
Poor	poor reviews count in default language	numeric
Terrible	terrible reviews count in default language	numeric
Food	food rating	numeric
Service	service rating	numeric
Value	value rating	numeric
Atmosphere	atmosphere rating	numeric
Keywords	popular keywords	nominal

Data Preparation

The Athens and Lisbon datasets were downloaded and integrated using Excel. In each dataset, we created a new attribute called "City" and entered the respective city. The first column contained an index or identification number, which did not have predictive value, so we removed this attribute. The dataset also contained normalized versions of the attributes "attr_index" and "rest_index," which we also chose to remove to avoid redundancy. When booking an Airbnb, it is common for a group of individuals to book together and split the overall cost. Therefore, we decided to create a new price per guest attribute by dividing the price of the Airbnb by the number of people it could accommodate. While the Airbnb dataset contained no missing values, the restaurant dataset required some cleaning, reformatting, and attribute modification. We first filtered the dataset to only restaurants located in Athens or Lisbon, and removed all attributes except for the restaurant name, city, latitude, longitude, awards, and price level. We removed instances that had missing values for any of these attributes, and we changed the symbols used for the TripAdvisor price ranges into words: "cheap eats," "medium price," and

"fine dining." To integrate this dataset with each of the Airbnb datasets, we utilized R to write a nested for loop that calculated the distance between a particular Airbnb and each restaurant using the latitude and longitude coordinates (Figures 1, 2).

$$distance = \sqrt{(lat_A - lat_R)^2 + (long_A - long_R)^2}$$

$$A = Airbnb$$
$$R = Restaurant$$

Figure 1. Euclidean distance formula to find the distance between an Airbnb and a restaurant.

The restaurant with the minimum distance, and its corresponding attributes, were added to the Airbnb instance. This process looped through all Airbnb latitudes and longitudes.

```
for (i in 1:length(Athens$Airbnb_lat)){
  min_distance = 100
  for (j in 1:length(Athens$Airbnb_lat)){
      new_min_distance = sqrt((((Athens$Airbnb_lat[i]) -
                          (Athens$OrigRest_lat[j]))^2) +
      (((Athens$Airbnb_long[i]) -
          (Athens$OrigRest_long[j]))^2))

    if(isTRUE(new_min_distance < min_distance)==TRUE){
      min_distance = new_min_distance
      Athens$Rest_lat[i] <- Athens$OrigRest_lat[j]
      Athens$Rest_long[i] <- Athens$OrigRest_long[j]
      Athens$Restaurant[i] <- Athens$Old_RestName[j]
      Athens$PriceLevel[i] <- Athens$old_price_level[j]
      Athens$NumCertExcel[i] <- Athens$old_NumCertificatesExcellence[j]
      Athens$TravellersChoice[i] <- Athens$old_Traveller_Choice[j]
      Athens$Michelin_TwoStar[i] <- Athens$old_Michelin_2_Star[j]
      Athens$Michelin_Very_Comfortable[i] <- Athens$old_Michelin_Very_Comfortable[j]
      Athens$Michelin_Plate[i] <- Athens$old_Michelin_Plate[j]
      Athens$Michelin_Simple_Rest[i] <- Athens$old_Michelin_Simple_Restaurant[j]
      Athens$Michelin_OneStar[i] <- Athens$old_Michelin_1_Star[j]
      Athens$Michelin_Bib_Gourmand[i] <- Athens$old_Michelin_Bib_Gourmand[j]
      Athens$Michelin_Extremely_Comfortable[i] <- Athens$old_Michelin_Extremely_Comfortable[j]
    }
  }
}
```

Figure 2. R code denoting nested for loop to find the restaurant with the minimum distance to each Airbnb in Athens and add the restaurant attributes to the Airbnb instance. An identical block of code was used to integrate the restaurants and Airbnbs in Lisbon.

From R, we exported both new datasets as csv files and opened them in Excel (Figure 3).

```
<<>>=
write.csv(Athens,"~/Library/CloudStorage/OneDrive-FurmanUniversity/Senior/Spring 2025/CSC 272/Term
Project/AthensRestaurants.csv", row.names = TRUE)
write.csv(Lisbon,"~/Library/CloudStorage/OneDrive-FurmanUniversity/Senior/Spring 2025/CSC 272/Term
Project/LisbonRestaurants.csv", row.names = TRUE)
@
```

Figure 3. R code for exporting the integrated datasets as csv files.

We then copied the entire Lisbon dataset and pasted it below the Athens dataset to complete the integration process. Next, we removed any special characters in the restaurant name and removed one of the "City" attributes because we had a duplicate after the integration process. The last piece of pre-processing involved changing the format of the awards attribute. The different awards were listed as comma separated values, but we created a new nominal, binary attribute for each of the individual awards: Michelin_TwoStar, Michelin_Very_Comfortable, Michelin_Plate, Michelin_Simple_Rest, Michelin_OneStar, Michelin_Bib_Gourmand, and Michelin_Extremely_Comfortable, and TripAdvisor's Traveler's Choice Award. There was one award, Certificate of Excellence, that had individual years associated with the award. We decided to represent this attribute as numeric, with the attribute value being the total number of certificates a restaurant received.

All numeric attributes were discretized using the optimal equal-width binning in Weka so that classification algorithms could be appropriately analyzed. When discretizing the realSum attribute to categorize the Airbnb prices into price ranges, we noticed only two buckets were generated, with seemingly all the values in one bucket. Upon closer investigation, there was one Airbnb in Athens that had a price value of 18,545 euros per two nights. We figured this was an error since the next highest value was 1,681 euros, as observed in the dataset and in a scatterplot (Figure 4). Because the dataset contained a large volume of instances, we mitigated this issue by removing the outlier, leaving us with 5509 instances, 32 attributes, and no missing values.

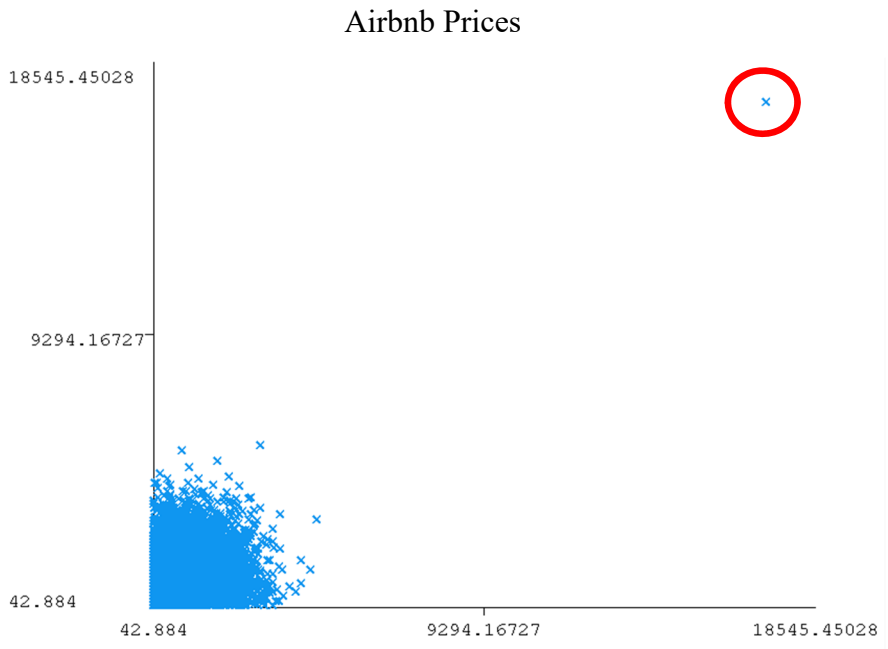

Figure 4. Scatterplot of all the Airbnb prices, with the outlier (18,545 euros) circled.

Furthermore, some of the attributes, specifically guest satisfaction rating and cleanliness rating, were very unbalanced, with more than 90% of the data values falling into the highest categories. This was important to address in the training data before using classification algorithms to predict guest satisfaction, especially since we suspected cleanliness rating was highly correlated with guest satisfaction. Synthetic Minority Oversampling (SMOTE) and resampling techniques were explored to address the data imbalance. First, SMOTE increases the frequency of the minority class by relying on information from the *k* nearest neighbors to generate new instances. This algorithm had to be applied multiple times since there were multiple class values requiring much higher frequencies. As a result of SMOTE, the dataset was balanced. However, several missing values resulted in other variables, some binary variables then contained decimal points, and the algorithm failed with initial frequencies of less than ten. We concluded SMOTE was not an ideal algorithm due to the severity of the imbalance, which at times required more than 10,000% increases in the number of values.

Next, we resampled guest satisfaction to generate a random subsample with a uniform class distribution, resulting in 549 instances per value in the training set. This process was repeated for cleanliness rating as well, and then once more for guest satisfaction to create

somewhat balanced data distributions for both attributes (Figure 5). In future research, under-sampling techniques could also be explored and compared to the results of oversampling. This would require only changing the number of instances on the majority class value, rather than all other class values.

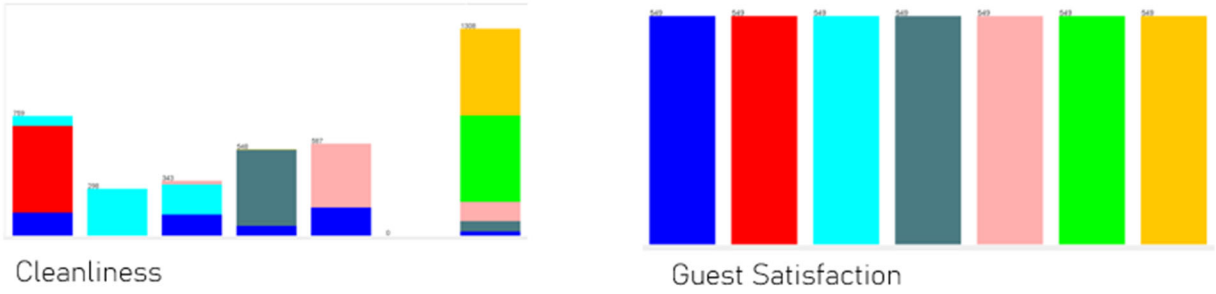

Figure 5. Distribution of cleanliness (left) and guest satisfaction (right) values after resampling techniques were applied.

Data Analysis

Before beginning analysis, the dataset was split into training and test data using Weka. The entire integrated dataset was loaded into Weka and was randomized to ensure that the Airbnb properties of both cities were adequately mixed. Weka was then used to remove 30% (1653) of the data, which was saved as a separate ARFF file for testing. The remaining 70% (3856 instances) of the data was saved to be used as the training data. When modeling, the supplied test set option was selected, and the test set of data was indicated for use. It is important to note that when modifying the class attribute or removing attributes in the training data, the same had to be performed on the testing data as well to ensure that Weka supplied the desired test set. The algorithms used to investigate the Airbnb and restaurant data included ZeroR, OneR, J48, PRISM, JRip (RIPPER), IBk, simple Naïve Bayes, and Apriori. The attribute selection filters CsfSubset Eval, Information Gain, and WrapperSubset Eval were also used to identify attributes or a subset of attributes suggested by Weka to be the most predictive.

ZeroR

ZeroR was used as our baseline algorithm, to which we compared the performance of the remaining algorithms. This algorithm always selects the majority attribute and does not consider any other information when determining the class value. The accuracy of ZeroR indicates the percentage of our class attribute values that are the majority value. For the four class attributes, Airbnb price, price per guest, guest satisfaction, and restaurant price, the ZeroR results were lowest price range (63.2%), lowest price range (71.4%), lowest satisfaction range (0.3025%), and mid-range priced restaurant (57.4%), respectively. The most surprising result was the very low accuracy and prediction of the guest satisfaction. This is likely a result of the resampling technique used. Resampling was implemented so that the guest satisfaction category had a uniform distribution. Therefore, all categories of guest satisfaction were equally likely to be selected; however, our test set only had a few low-value satisfaction ratings, thus resulting in a very low accuracy.

OneR

OneR was then used to identify the most predictive attributes for each of our four class attributes. This algorithm selects the attribute that has the highest accuracy of predicting the class attribute value correctly.

Apriori

In the beginning stages of the data analysis, the Apriori algorithm was used to identify attributes that were potentially correlated with one another. This algorithm identifies rules with confidence levels at or above the threshold, decreasing the minimum support as the algorithm progresses (Weka). Apriori helps us identify potential multicollinearity and overfitting that could skew results, while also offering insight into potential trends in the data. For the Apriori algorithm, support was set to 75% and confidence was set to 90%. The results showed the Michelin awards were highly correlated with one another, the type of room and if the room was shared were correlated, and high guest satisfaction related to high cleanliness ratings. These associations all seemed logical, with the relationship between guest satisfaction and cleanliness having the most potential for future experimentation.

Attribute Selection: Information Gain, Subset Selection, Wrapper Method

Before running more specified algorithms, various attribute selections were performed in Weka to identify the most predictive attributes and groups of attributes with the goal of eliminating noise and generating more accurate models in future analysis. Information gain measures the improvement in the model when each attribute is added, considering how well the attribute splits and correctly predicts the class attribute values ("Information Gain"). Each attribute is then ranked based on its contribution to the model. Subset Selection considers the predictive ability of various subsets of attributes while also considering if any redundancy exists (Weka). Finally, the Wrapper Method identifies subsets of attributes and then builds the selected model (i.e. J48) with each subset, testing the accuracies of the models using cross validation.

J48 Decision Tree

After the initial explorations were complete, more advanced algorithms were considered, with the first being J48. This algorithm builds a predictive tree using the attributes that best divide the class attribute values, considering both the overall accuracy of the predictive attribute and the number of imperfect nodes remaining after the attribute is added to the tree. This algorithm appeared most useful for the Airbnb price and price per guest attributes, resulting in only twenty-two and twelve leaves, respectively, in each of the decision trees. In contrast, the guest satisfaction tree had over 150 leaves (with cost sensitive analysis) and over 200 leaves (without cost sensitive analysis), and restaurant price level had only four leaves, and failed to classify one value of the class attribute.

PRISM

Two rule-generating algorithms were applied to the data, one of which was PRISM. This algorithm requires nominal values and identifies rules and rule sets for each class attribute value, focusing on one attribute at a time. A rule is completed when the ratio of positive instances (ie. number correct for rule) to total instances (i.e. all to which the rule is applied) is equal to one. If multiple attributes result in a ratio of one, the attribute with greater coverage is preferred. A rule set is complete when all instances of the desired class attribute value are covered. For all four

class attributes, the accuracies of the PRISM algorithm were approximately 60-70% with large sets of rules.

JRip (RIPPER)

The second rule-generating algorithm explored was the JRip algorithm, which implements pruning techniques to improve the accuracy and efficiency of the rule sets. The algorithm accomplishes this through a repeated process of growing, pruning, and optimizing. The algorithm increases the components of the antecedents to create rules with 100% accuracy with the use of information gain, followed by pruning them to reduce overfitting and improve the applicability of the model to future data (Weka). This algorithm was applied to the guest satisfaction class attribute since the PRISM algorithm resulted in over 120 rules, with approximately 12% of the data being unclassified. The JRip algorithm significantly improved these results, increasing the accuracy from 62% to 74% and decreasing the number of rules from 122 to 48.

IBk

This nearest neighbor algorithm considers the *k* closest neighbors (or instances) to the one attempting to be classified using the Euclidean distance formula with normalized input values. If *k* is greater than one, the algorithm considers the majority vote of all the neighbors. Increasing the neighborhood can help overcome the negative impacts of noise in large datasets. For this project, the neighborhood sizes utilized ranged from one to ten. A larger neighborhood size was included because the dataset was quite large. It was hypothesized that noise may be present, potentially impacting the obtained results. However, as the neighborhood size was increased, the accuracy changed minimally, indicating little noise existed.

There are also different search methods by which the algorithm searches for the nearest neighbors. For this project, the linear search, KD tree, and ball tree methods were used. The linear search method completes a linear scan of the data to locate the instances in the training data that are most similar to the instance being classified. The KD tree method divides the data into a binary decision tree in multidimensional space by choosing a dimension at each level of the tree in which split the data to create nodes (“Ball Tree”). This process continues until a specified termination condition is met, providing a classification for the instance (“Ball Tree”).

The ball tree method divides the data into spheres structured as a binary tree, and searches for the nearest neighbors of the instance to be classified by dividing the data into smaller and smaller circles based upon similarities ("Ball Tree").

Naïve Bayes

The Naïve Bayes algorithm identifies the probability of each class value given the specifics of the instance using statistical calculations. This algorithm can be particularly helpful when needing to understand "close seconds" for predictions.

Cost-Sensitive Analysis

Cost-sensitive analysis was applied during the analysis of the guest satisfaction attribute. This process involves assigning different weights to incorrectly classified values, forcing the algorithm to more strongly focus on correctly predicting those class attribute values. The importance of using this technique became evident when running OneR for guest satisfaction. The accuracy was about 0.4%, and the confusion matrix indicated most of the highest rated guest satisfactions were classified in the second to highest category (Figure 6).

Low-Low	Low-Med	Low-High	Med-Med	High-Low	High-Med	High-High
0	4	1	0	0	0	0
0	0	0	0	0	0	0
0	0	1	0	3	1	0
0	0	0	1	0	1	0
0	0	0	0	3	9	0
0	0	0	0	1	2	0
0	0	0	0	8	1618	0

Figure 6. Confusion matrix of One R algorithm, with guest satisfaction as the class attribute.

Once we increased the cost of mis-classifying high-high as high-med to 5 (rather than 1), the accuracy of OneR jumped to 98%, and the confusion matrix improved (Figure 7).

Low-Low	Low-Med	Low-High	Med-Med	High-Low	High-Med	High-High
1	4	0	0	0	0	0
0	0	0	0	0	0	0
3	0	1	0	0	0	1
0	0	0	1	0	0	1
3	0	0	0	0	0	9
1	0	0	0	0	0	2
8	0	0	0	0	0	1618

Figure 7. Confusion matrix after increasing the cost for incorrectly mis-classifying high-high as high-med (cost = 5).

This matrix resulted in much better accuracy, but it is important to note that all high-low and high-med instances were misclassified, so future experimentation is needed to better predict these values. However, there existed a high success rate for predicting high guest satisfaction ratings. While not as drastic as the OneR results, this cost sensitive analysis also improved the results of the other algorithms explored.

Data Analysis Results

Airbnb Price and Price per Guest

Before creating models to predict the Airbnb price, the price per guest attribute was removed to avoid redundancy as it was created from two attributes already existing in the dataset (overall price divided by person capacity). All the above-described classification algorithms were used to obtain preliminary results. Then, the three attribute selection methods were employed to find the most predictive attributes: room type, rest_index, bedrooms, restaurant, and city. The classification algorithms were repeated using this subset of attributes. The average accuracy of the models was around 90%. During this process, the various algorithms seemed to be biased towards the restaurant attribute, likely because it has 1,216 potential values, nearly one for every three Airbnbs. Therefore, the restaurant attribute was removed because overfitting was a concern, and attribute selection was conducted a second time. The set of most predictive attributes was city, rest_index, bedrooms, person capacity, multi, attr_index, and room type. This set of attributes was subjected to the classification algorithms, giving an average accuracy of

approximately 80% (Table 3). The J48 algorithm had the highest accuracy of classifying instances.

Table 3. Airbnb Price Results

Algorithm	Accuracy	Notes
ZeroR	63.2002 %	To obtain base line. Algorithm predicted that the Airbnb price would fall into the bin with the lowest price range.
OneR	70.6594 %	Rest_index was selected as the attribute used to make classification rule.
J48	80.5203 %	The generated decision tree had 22 leaves with the city attribute as the root node. The tree also included all of the attributes.
PRISM	63.9443 %	Did not appear to rely heavily on one attribute to generate rules for covering the instances.
Simple Naïve Bayes	79.0684 %	Most probable class attribute value was determined to be a price that falls into the lowest price range bin.
IBk (linear search)	k = 1 79.6733 % k = 3 80.0363 % k = 5 79.9153 % k = 10 80.5203 %	Accuracy across neighborhood sizes and search methods remained around 80%.
IBk (KD Tree)	k = 1 79.6733 % k = 3 80.0968 % k = 5 80.0968 % k = 10 80.4598 %	
IBk (Ball Tree)	k = 1 79.6733 % k = 3 80.0363 % k = 5 79.9153 % k = 10 80.5203 %	

The price per guest attribute was handled very similarly to the Airbnb price. First, the Airbnb price and person capacity attributes were removed because they were redundant. Initial modeling and attribute selection were performed. The most predictive attributes were restaurant, city, and rest_index. Several of the models had an accuracy of 95%. The results also indicated that the restaurant attribute likely should be removed to prevent overfitting. After removal of restaurants, the same algorithm re-running and re-selection of attributes was conducted. The most predictive attributes were determined to be city, rest_index, bedrooms, host_is_superhost, and cleanliness rating. The average accuracy of the models was around 74% with IBk (k=1) being the most accurate (Table 4).

Table 4. Price per Guest Results

Algorithm	Accuracy	Notes
ZeroR	71.4212 %	To obtain base line. Algorithm predicted that the price per guest would fall in the lowest bin.
OneR	72.7697 %	Selected the rest_index attribute to generate classification rule.
J48	73.8849 %	The generated decision tree had 12 leaves with the city attribute as the root node. The other attributes included were rest_index, host_is_superhost, and bedrooms.
PRISM	71.6027 %	Did not appear to rely heavily on one attribute to generate rules for covering the instances.
Simple Naïve Bayes	69.9429 %	Most probable class attribute value was determined to be a price per guest that falls into the lowest price range bin.
IBk (linear search)	k = 1 74.0923 % k = 3 74.0145 % k = 5 73.9108 % k = 10 73.8849 %	Accuracy across the neighborhood sizes and search methods remained around 74%. As neighborhood size increased, accuracy slightly decreased.
IBk (KD Tree)	k = 1 74.0923 % k = 3 73.9886 % k = 5 73.8849 % k = 10 73.8589 %	

IBk (Ball Tree)	k = 1	74.0923 %	
	k = 3	74.0145 %	
	k = 5	73.9108 %	
	k = 10	73.8849 %	

Guest Satisfaction

The data mining results for guest satisfaction indicated cleanliness, more than any other attribute, strongly influences the satisfaction levels of guests. The various algorithms all aligned with the initial OneR prediction (Table 5). Overall, OneR seemed to sufficiently predict the guest satisfaction using cleanliness scores, which was expected, given the correlation between the two attributes (Figure 8). According to the rules generated, one exception we noticed was that a shared room caused a decrease in satisfaction ratings, even for places with high cleanliness ratings. We attributed this to potentially disruptive and incompatible roommates.

Table 5. Guest Satisfaction Results (with Cost-Sensitive Analysis)

Algorithm	Accuracy	Notes
ZeroR	0.3025 %	To obtain base line. Algorithm predicted the guest satisfaction level fall into the bin with the lowest ratings. However, ZeroR was not very useful in this scenario due to the uniform distribution of the class values.
OneR	98.0641%	Selected the cleanliness rating attribute to generate classification rule.
J48	80.6413%	The generated decision tree had 167 leaves, with predictive attributes including cleanliness, bedrooms, biz, and restaurant price.
PRISM	59.5886 %	Resulted in 122 rules, with 12.6437% of the data remaining unclassified.
RIPPER (JRip)	72.47%	Resulted in 48 rules and higher accuracy relative to PRISM, demonstrating the benefit of pruning the rules.

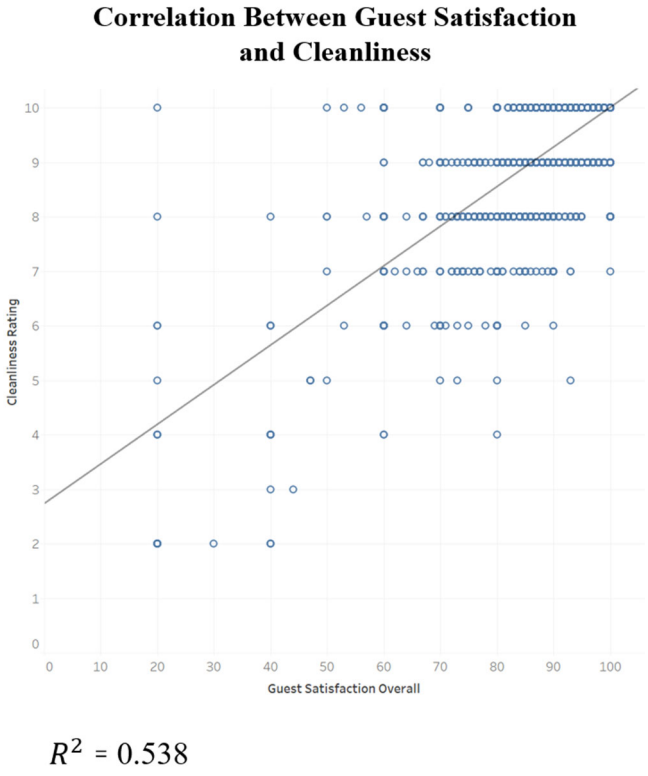

Figure 8. Scatterplot of guest satisfaction and cleanliness ratings. There is a positive correlation between the two attributes, and the $R^2 = 0.538$.

Restaurant Price Level

Upon exploring our data in Tableau, we noticed the more expensive Airbnbs appeared to be clustered closer to the fine dining restaurants in both Athens and Lisbon (Figure 9), so we were curious to explore the predictive ability of Airbnb attributes on restaurant price. We removed the latitude, longitude, restaurant, and price per guest attributes to minimize bias and remove redundancies. With twenty-six attributes remaining, we conducted attribute selection to determine the most predictive subset of attributes and then ran the classification algorithms on a reduced dataset containing only the Michelin_Plate, Michelin_One_Star, Michelin_TwoStar, NumCertExcel, and TravellersChoice attributes. Naïve bayes and IBk were the best-performing

algorithms (Table 6). OneR, J48, and Prism gave similar accuracies to the baseline ZeroR, although PRISM did cover more of the class attribute values.

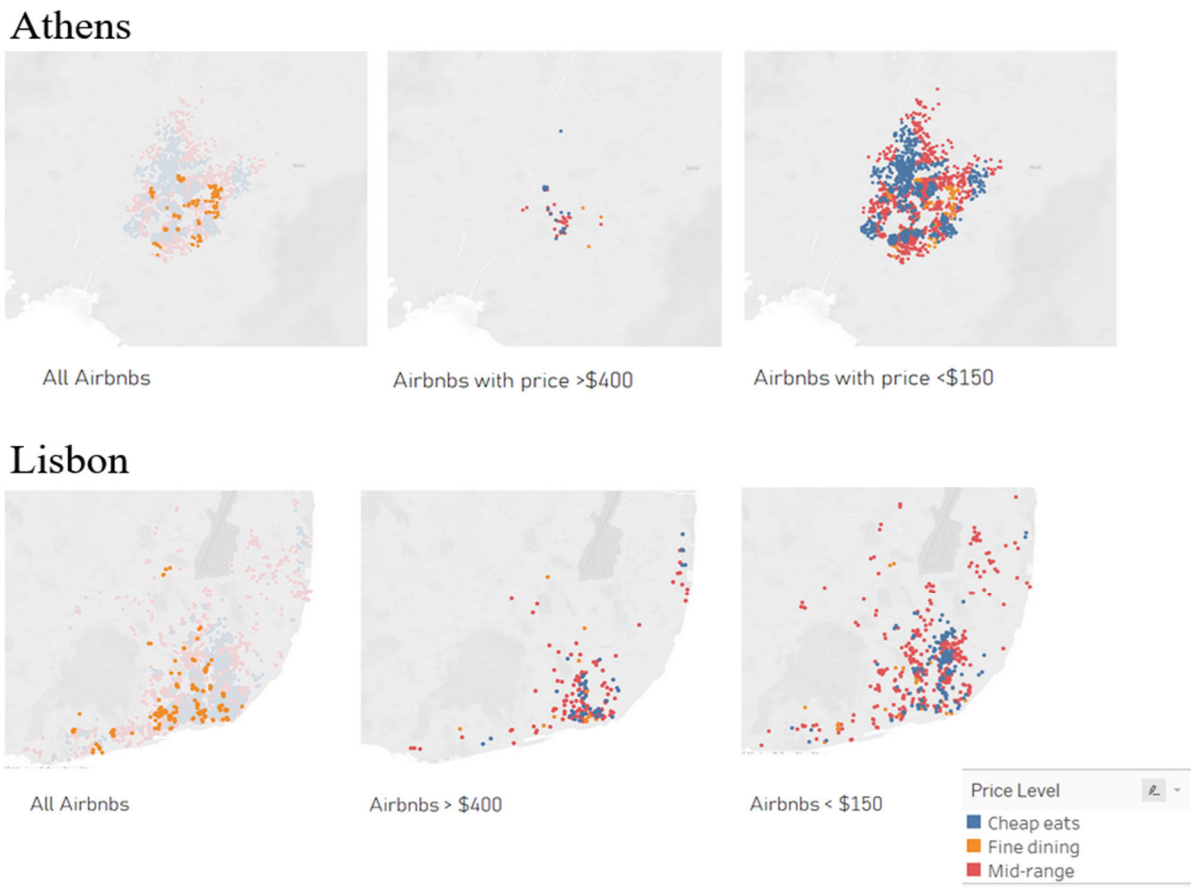

Figure 9. Geographical mapping of the Airbnbs, color-coded by the price level of the nearest restaurant: blue for cheap eats, red for mid-range, and orange for fine dining.

Table 6. Restaurant Price Level Results

Algorithm	Accuracy	Notes
ZeroR	57.4108 %	To obtain base line. Algorithm predicted the restaurant price level would be mid-range.
OneR	58.3182 %	Selected the Michelin_Plate attribute to generate classification rule.
J48	58.8627 %	The generated decision tree had 4 leaves with Michelin_Plate as the root node. The tree also split on city, and metro_dist near the bottom of the tree.

PRISM	59.7096 %		Even though accuracy was not that much better than OneR, the confusion matrix showed that each class attribute was represented.
Simple Naïve Bayes	63.8838 %		Most probable class attribute value was the mid-range price level. (P) = 0.57605597
IBk (linear search)	k = 1 k = 3 k = 5 k = 10	64.4283 % 62.9764 % 62.7949 % 62.8554 %	Accuracy across the neighborhood sizes and search methods remained around 63 %.
IBk (KD Tree)	k = 1 k = 3 k = 5 k = 10	64.7913 % 63.8838 % 63.4604 % 63.7024 %	
IBk (Ball Tree)	k = 1 k = 3 k = 5 k = 10	64.4283 % 62.9764 % 62.7949 % 62.8554 %	

Changing either the neighborhood size or the search algorithm did not impact the accuracy very much at all, signifying there was little noise in the dataset. While we had developed a hypothesis that more expensive, high-quality Airbnbs may be located near expensive restaurants, we discovered that the Airbnb attributes were not predictive of the restaurant price level. The only attribute that appeared in the attribute selection was metro distance, which was not necessarily characteristic of the Airbnb itself, but more a function of location. We think we could have had better results had we kept some of the other attributes from the dataset, such as type of cuisine and type of meal served (bar food, café food, lunch, dinner).

Conclusion

In conclusion, while we were able to obtain relatively acceptable accuracies when predicting some class attributes, this was not the case for all the attributes in which we were interested. After removing redundant attributes, eliminating attributes that may have resulted in

overfitting, and conducting attribute selection, the most predictive attributes of the price of the Airbnb were city, attr_index, room_type, person_capacity, rest_index, bedrooms, and multi. The J48 algorithm was able to obtain an accuracy of 80.5203 %, making it the best model in predicting Airbnb price. Via the same process, the most predictive attributes for the price per guest were determined to be city, rest_index, bedrooms, host_is_superhost, and cleanliness_rating. The IBk algorithm with a neighborhood size k = 1 performed the best when predicting the class attribute and had an accuracy of 73.8849 %.

The modeling process and results obtained for the predictions of the restaurant price level and guest satisfaction were found to be a bit more interesting. When investigating restaurant price level as the class attribute, we found that the characteristics of the Airbnb properties were not predictive of the price of the restaurants at all. The only attribute from the Airbnb dataset that appeared in the attribute selection methods was metro distance, which is a function of the Airbnb's location rather a quality of the Airbnb itself. For guest satisfaction, significant work had to be done to balance the data. Cleanliness was found to be the dominant attribute in the prediction of guest satisfaction. However, it was interesting to note that even if a high cleanliness rating was present, having a shared room decreased guest satisfaction ratings. We theorized this was most likely a result of having to share the space with potentially disruptive roommates. Cost sensitive analysis allowed relatively high accuracies to be obtained, but this was likely also because the data was quite skewed.

Overall, we concluded that our models for Airbnb price were best at predicting the lowest price range. For example, based on the confusion matrix, J48 resulted in 88.87% accuracy for the lowest price range, 66.7% accuracy for the middle price range, and 50% accuracy for the highest price range. This may be due to the moderate data imbalance, favoring the lowest category. Similarly, the models were best at predicting the highest guest satisfaction ratings, relative to lower categories, for likely the same reasons.

One limitation of this study was the inaccurate recordings in the Traveler's Choice attribute, which was noticed after the completion of the data analysis. Oddly, at some point the values changed from binary ones and zeros to seemingly random numbers between zero and ten. When attempting to correct this error, the spreadsheet values changed mid-analysis once again. Further investigation is needed. This error may have impacted the results slightly, but this is not a major concern since Traveler's Choice was not a primary predictive attribute. Perhaps,

however, with the correction, the attribute will be more predictive. Additional limitations included only analyzing two cities, not yet exploring the results of ensemble learning algorithms, using only two sources of data, and removing instances with missing values. Future studies could address these limitations.

If we were to expand this project further, we would like to experiment with more of the attributes from the restaurant data, such as the cuisine type, and whether the restaurant is primarily a lunch, dinner, bar, or café establishment. In addition, we could integrate data that includes the weekend pricing of the Airbnbs to see how this would impact the results. It would also be interesting to investigate Airbnb locations in other cities to determine if there is a difference in the impact of Airbnb characteristics on the Airbnbs across cities. In the pre-processing stages, further experimentation with various binning techniques could be completed to potentially achieve more accurate results. Furthermore, it would be worthwhile to explore numeric estimation in comparison to classification to predict the price of the Airbnb and price per guest, as these attributes were originally numeric. These data mining techniques have the potential to suggest optimal pricing strategies for owners and help guests identify which Airbnb attributes are affecting the costs of their stays.

This project challenged us to integrate two datasets in a creative and meaningful way to learn more about the price relationships between Airbnbs and nearby restaurants. In addition to learning some new algorithms, such as JRip, we explored a variety of algorithms covered in the course on a deeper level, and analyzed the impacts of resampling techniques and cost-sensitive analysis.

Works Cited

Leone, Stefano. “TripAdvisor European restaurants.” *Kaggle*, 2021, https://www.kaggle.com/datasets/stefanoleone992/tripadvisor-european-restaurants.

The Devastator. “Airbnb Prices in European Cities.” *Kaggle*, 2024, www.kaggle.com/datasets/thedevastator/airbnb-prices-in-european-cities?select=athens_weekdays.csv.

“Information Gain and Mutual Information for Machine Learning.” *GeeksforGeeks*, 15 Apr. 2024, www.geeksforgeeks.org/information-gain-and-mutual-information-for-machine-learning/.

“Ball Tree and KD Tree Algorithms.” *GeeksforGeeks*, 9 Dec. 2023, www.geeksforgeeks.org/ball-tree-and-kd-tree-algorithms/.

“Waikato Environment for Knowledge Analysis (WEKA).” 3.8.6, The University of Waikato, 1999.