Unearthing Hidden Insights from Airbnb Data Using Gradient-Boosted Decision Trees

Abishek Padaki
San Jose State University
San Jose, CA
abishek.padaki@sjsu.edu

Mrunal Zambre
San Jose State University
San Jose, CA
mrunaldeepak.zambre@sjsu.edu

***Abstract*—The emergence of Airbnb has revolutionized the travel industry, offering travelers a wide range of lodging options and income opportunities for property owners. As the platform continues to grow, there is an increasing need for accurate price prediction and identifying desirable localities for travelers. In response to this demand, researchers have explored various machine learning techniques, including gradient-boosted decision trees (GBDT), to predict Airbnb prices and assess the popularity of different locations. This report presents using GBDT and other machine-learning techniques for predicting Airbnb prices, factors that affect their ratings, and identifying desirable localities for travelers. We demonstrate the potential of GBDT models to predict costs and assess the popularity of different localities, outperforming other machine learning techniques in various studies. Furthermore, using GBDT models in recommendation systems shows promise for uncovering patterns in traveler preferences and enhancing the Airbnb experience. Our findings have important implications for travelers and hosts on the Airbnb platform and researchers interested in applying machine-learning techniques to real-world problems in the travel industry.**

***Index Terms*—Gradient-boosted Decision Trees, Web Crawling, XGBoost and LightGBM**

I. Introduction

The advent of the sharing economy has led to an epochal shift in how individuals seek accommodations, with platforms like Airbnb democratizing access to various lodging options. Yet, as Airbnb's market expands, travelers and hosts grapple with the complexities of the platform. The quest for the perfect accommodation that aligns with their preferences and budget can be daunting for travelers. Conversely, hosts strive to devise the most profitable pricing strategies to attract potential guests.

Given these challenges, our research aims to develop a sophisticated model leveraging Gradient-Boosted Decision Trees, a potent machine learning algorithm, to predict Airbnb prices and determine localities with high traveler demand. The goal is to unearth hidden patterns in Airbnb's listing pricing and demand, facilitating travelers in finding suitable and economical accommodations while aiding hosts in strategizing competitive pricing to optimize their listings.

Our approach involves using a comprehensive dataset obtained by web-scraping Airbnb listings from large metropolitan areas across the United States. This dataset encapsulates a wide array of information, such as location, rental type, ratings, amenities, host data, and house rules. Further, it includes critical indicators of rental availability, pricing, and reviews from past guests, offering invaluable insights into guest preferences and experiences.

This research endeavors to extend the knowledge frontier on sharing economy platforms and addresses the urgent need for intelligent tools to augment the Airbnb experience. By harnessing machine learning's capabilities, we aspire to reveal valuable insights and patterns, fostering more informed decision-making in the realm of short-term rentals.

II. Related Work

Airbnb's advent has brought a transformational shift in the travel sector, presenting an array of accommodation choices for tourists and a source of revenue for homeowners. As the platform expands, the urgency for precise rate forecasting and pinpointing preferred areas for travelers has surged. Acknowledging this requirement, researchers have delved into several machine learning methodologies, such as gradient-boosted decision trees (GBDT), to anticipate Airbnb pricing and gauge the appeal of diverse destinations. This literature survey provides an overview of the relevant research on this topic, focusing on GBDT and related techniques.

A. Gradient Boosted Decision Trees for Price Prediction

Gradient-boosted decision trees have gained prominence in recent years due to their effectiveness in handling regression and classification problems, as Friedman demonstrated in his paper [1]. The research could be extended to explore the effect of different hyperparameters or regularization techniques on the performance of the GBDT algorithm, such as by using cross-validation or Bayesian optimization, to identify the optimal settings for different datasets and tasks.

In the context of Airbnb price prediction, researchers have applied GBDT models to analyze the relationship between listing features and pricing. For example, Zhu et al. [2] used GBDT to predict Airbnb prices in New York City, incorporating features such as neighborhood, property type, and amenities. Their study demonstrated the ability of GBDT to model complex, non-linear relationships between features and prices, leading to improved prediction accuracy.

A drawback of this paper is that it did not provide a comprehensive analysis of the limitations and challenges of using the machine learning techniques to predict Airbnb prices. The study can be extended to provide this analysis, which could help guide future research and development in this area.

B. Comparing GBDT with Other Machine Learning Techniques

Several studies have compared Gradient Boosting Decision Trees (GBDT) with other machine-learning techniques for predicting Airbnb prices. For instance, Mora-Garcia et al. [3] compared linear regression and Ensemble learning algorithms based on boosting (GBDT), for housing price predictions in the Spanish city of Alicante. Their results indicated that GBDT outperformed the other models in terms of prediction accuracy, with the lowest mean absolute error. This finding supports the potential of GBDT as a powerful tool for predicting housing prices.

One of the advantages of using GBDT in this study was its ability to capture complex nonlinear relationships between features, leading to improved predictive performance. However, a limitation of the study by Mora-Garcia et al. [3] is that it focused on a single city, which may limit the generalizability of their findings to other locations or housing markets.

A possible extension to this paper would be to include a broader range of locations and test the performance of GBDT against other state-of-the-art machine learning models, such as neural networks, to provide a more comprehensive comparison and assessment of their applicability in the housing price prediction domain.

Additionally, Ahuja et al. [4], propose a price prediction algorithm for Airbnb listings using geolocation, temporal, visual, and natural language features. They utilize LightGBM, a gradient boosting framework, to fit the training data, incorporating features such as latitude, description score, longitude, review sentiments, neighborhood popularity, host experience, and listing availability. The model demonstrates a high R2 score, indicating that the selected features effectively explain the variance in rental prices.

Among the top features, latitude and description score emerge as key predictors for listing prices. One of the strengths of this study is the use of multi-modal features, which provide a comprehensive representation of Airbnb listings and contribute to the model's accuracy. Additionally, the paper highlights the importance of feature engineering, demonstrating that well-chosen features significantly impact prediction outcomes. However, one limitation of the study is that it does not explore the potential impact of seasonality on rental prices, which could lead to an underestimation of the model's performance during peak or off-peak periods.

A possible extension to this paper would be to evaluate the performance of the model on Airbnb listings from different states or countries, potentially incorporating region-specific features and accounting for seasonality effects on rental prices.

III. Designing an Airbnb Scraper

The intricacy of modern websites, typified by dynamic content, complex Document Object Model (DOM) structures, and anti-scraping measures, presents numerous challenges in web scraping. To navigate these complexities, we designed an efficient and ethical scraper to extract information from Airbnb listings, adhering to the platform's scraping policies.

A. Circumventing the robots.txt to scrape data

Airbnb's robots.txt file represents their directives for web crawlers. It delineates what paths on the website should not be traversed by bots to respect privacy and prevent server overload. Our goal was not to infringe upon these rules but to retrieve data not explicit in the disallowed paths but crucial for our analysis. Here, we detail our procedure to ethically circumvent the restrictions.

In an attempt to be respectful and abide by the rules set by Airbnb in their robots.txt file, we adopted a number

of methodologies. The initial step was the scraping of listings, their titles, and type from the landing page. Pagination was used to traverse each page. To imitate a human-like browsing behavior, we set a delay of 5 seconds before moving to the next page. This timeout was found to be ample for most interactions that dealt with site navigation.

The next step was to dive deeper into each individual listing. Information like price, rating, specifications, rules, and more were fetched from the listing's landing page. Despite these pages not explicitly being on the disallowed list in the robots.txt, we ensured to scrape these pages respectfully, maintaining timeouts and minimizing the requests per second.

The main challenge we faced was with the Top amenities and Review count. The modal to display all amenities and reviews was listed under the 'disallowed' category in the robots.txt. Hence, we had to depend only on the information displayed on the listing page itself, respecting Airbnb's bot rules.

By adhering to these methodologies, we were able to retrieve data vital for our analysis without infringing upon the website's scraping policy. Through the ethical use of Python tools and libraries and by respecting the robots.txt rules, we demonstrated that web scraping could be performed responsibly and unobtrusively.

B. Developing the scraper

The scraper operates by initializing a Selenium webdriver to interact with the website. We optimized the performance by running the driver in headless mode and disabling GPU rendering. The scraping process starts from the landing page, extracting listing details like titles, types, price, and rating, along with specific accommodation attributes, such as amenities, specifications, rules, and reviews. We incorporated a 5-second timeout before moving to the next page to respect Airbnb's bot policies and mitigate the chances of being blocked.

To extract comprehensive data from each listing, we deployed a multi-stage scraping approach. First, listings were retrieved from the landing page, with pagination used to traverse through the results. Next, the scraper accessed individual listing pages, extracting details such as price, rating, specifications, and house rules. Given certain restrictions in the website's robot rules, we had to rely on the visible amenities and review count, rather than all available amenities and reviews.

The scraper then extracted accommodation specifications, including the maximum number of guests, number of rooms, beds, and baths. Additionally, it compiled the house rules from listings and extracted the number of reviews, ratings, and superhost status. This data was returned in the form of dictionaries, appended to the listings dataframe, and subsequently saved to a CSV file.

To ensure comprehensive geographic coverage, we maintained a JSON file containing an array of city objects, categorized by city, state, coast, and the associated Airbnb URL. The scraper functioned iteratively, traversing through 70 cities across various regions. In case of a scraping failure, the scraper saved the current row data and proceeded to the next row, thereby mitigating the risk of data loss.

The scraper successfully retrieved data from approximately 60 cities, with each city providing around 270 scrapable listings, yielding a dataset of over 16,000 listings. The scraping process for a single city took

```
User-Agent: *
Allow: /calendar/ical/
Allow: /.well-known/amphtml/apikey.pub
Disallow: /account
Disallow: /alumni
Disallow: /associates/click
Disallow: /api/v1/trebuchet
Disallow: /calendar/
Disallow: /disaster/lookup
Disallow: /email/unsubscribe
Disallow: /fix-it
Disallow: /fixit
Disallow: /forgot_password
Disallow: /groups
Disallow: /help/search
Disallow: /help/feedback
Disallow: /home/dashboard
Disallow: /inbox
Disallow: /logout
Disallow: /manage-listing
Disallow: /messaging/ajax_already_messaged/
Disallow: /my_listings
Disallow: /skeleton$
Disallow: /skeleton/
Disallow: /payments/book
Disallow: /signup_modal
Disallow: /signed_out_modal.json
Disallow: /device_id_bev_map
Disallow: /help/search
Disallow: /trips/upcoming
Disallow: /trips/v1/
Disallow: /update-your-browser
Disallow: /reservation

        Disallow: /rooms/*/safety
        Disallow: /rooms/*/reviews
        Disallow: /rooms/*/photos
        Disallow: /rooms/*/location
        Disallow: /rooms/*/house-rules
        Disallow: /rooms/*/enhanced-cleaning
        Disallow: /rooms/*/amenities
        Disallow: /users/show
        Disallow: /users/*/listings
        Disallow: /contact_host
        Disallow: /book
        Disallow: /stories
        Disallow: /embeddable
        Disallow: /guidebooks
```

Fig. 1. Airbnb's robots.txt policies (As of May 2023)

Algorithm 1 Web Scraper Algorithm

function INIT_DRIVER(browser)
    Initialize web driver with browser options
end function
function SCRAPE_LISTINGS(driver, city, url, num_pages, listings_df)
    Navigate to URL and scrape listing data for each listing on multiple pages
end function
function AMENTITY_SCRAPER(driver)
    Extract details about amenities from a listing page
end function
function INFRA_SPECS_SCRAPER(driver)
    Extract infrastructure specifications (like max guests, number of rooms, beds, and baths) from a listing page
end function
function SCRAPE_HOUSERULES_DATA(driver)
    Extract house rules from a listing page
end function
function REVIEW_RATINGS_HOST_SCRAPER(driver)
    Extract review ratings and host details from a listing page
end function
function SCRAPE_DETAILS_PAGE(driver, df_row)
    Call above functions to extract all details available on a listing page
end function
function SCRAPE_ALL_LISTING_URLS(driver, city_df, output_file_name, failure_row_index=0)
    Loop over each row of city_df, navigate to each listing page, call scrape_details_page() and store the data
end function
function SAVE_TO_CSV(df, file_name, start_index=0, end_index=0)
    Save DataFrame to CSV
end function
function MAIN
    Initialize web driver
    Set up DataFrame for storing listings
    Load city URLs from JSON file
    For each city, scrape all listings and store data in DataFrame
    Save DataFrame to CSV
    Close the web driver
end function

roughly 3900 seconds (1 hour), highlighting the time-intensive nature of ethical web scraping.

In conclusion, our scraper was designed with a focus on efficiency, performance, and ethical scraping practices. It offers a robust solution for extracting data from dynamic, JavaScript-heavy websites while respecting platform-specific scraping policies.

IV. Data Cleaning

With our scraper, we gathered data from 5 different coasts and stored it in individual CSV files. To create a unified dataset, we merged all the files into a single CSV file. Next, we utilized OpenRefine for data cleaning and preprocessing tasks on the scraped data. Originally, the CSV had 23 columns, and a total of 16,200 rows. The column summary can be seen in Fig. 2. We decided to preserve the original columns, and just add new columns with the suffix "_cleaned", which indicated that they were cleaned and processed.

```
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   city                  8088 non-null   object
 1   state                 8088 non-null   object
 2   coast                 8088 non-null   object
 3   title                 8088 non-null   object
 4   price                 8088 non-null   object
 5   link                  8088 non-null   object
 6   room_type             8088 non-null   object
 7   max_guests            8088 non-null   int64
 8   num_rooms             8088 non-null   int64
 9   num_beds              8088 non-null   int64
 10  num_baths             8088 non-null   float64
 11  rating                8088 non-null   object
 12  reviews               8088 non-null   object
 13  rating_Cleanliness    7961 non-null   object
 14  rating_Accuracy       7961 non-null   object
 15  rating_Communication  7961 non-null   object
 16  rating_Location       7961 non-null   object
 17  rating_Check-in       7961 non-null   object
 18  rating_Value          7961 non-null   object
 19  superhost             8088 non-null   object
 20  house_rules           8088 non-null   object
 21  num_of_amenities      8088 non-null   int64
 22  available_amenities   8088 non-null   object
dtypes: float64(1), int64(4), object(18)
```

Fig. 2. Scraped Dataset columns

To ensure data quality and prepare the dataset for analysis, several cleaning and transformation steps were performed. Here is a summary of the data cleaning process:

1) Rows with empty data, resulting from scraping errors, were deleted from the dataset.
2) The "price" column was cleaned by extracting the current price, either discounted or undiscounted,

from multiple lines and listed prices. The extracted prices were stored in a new column called "current_price_cleaned." The "$" sign was removed, and the values were transformed into numeric data.

3) A new boolean column, "is_price_discounted_cleaned," was added to indicate whether the current price was discounted or not, using information from the original "price" column.
4) From the original "room_type" column, only the room type label was extracted and stored in a new column called "room_type_cleaned."
5) To address the issue of random symbols and emojis in the listing titles, regular expression (regex) substitution was applied to remove those characters. The cleaned titles were stored in a column named "title_cleaned."
6) The "available_amenties_cleaned" column was created by removing any amenities labeled as "Unavailable" and converting the remaining amenities into a comma-separated list.
7) A boolean column, "is_superhost," was added with a value based on if the original "Superhost" column was empty or not.
8) Columns such as "num_rooms," "num_beds," and "num_baths," which were stored as strings but represented numeric values, were converted to columns of the numeric data type.

After completing the cleaning process, the resulting dataset contained a total of 33 columns and 13,466 rows. The summary can be seen in Fig. 3. These cleaning and transformation steps were crucial for ensuring data integrity and improving the quality of the dataset for subsequent analysis.

```
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   city                         13446 non-null  object
 1   state                        13446 non-null  object
 2   coast                        13446 non-null  object
 3   title                        13446 non-null  object
 4   title_cleaned                13446 non-null  object
 5   price                        13446 non-null  object
 6   current_price_cleaned        13446 non-null  int64
 7   is_price_discounted_cleaned  13446 non-null  bool
 8   link                         13446 non-null  object
 9   room_type                    13446 non-null  object
 10  room_type_cleaned            13446 non-null  object
 11  max_guests                   13446 non-null  int64
 12  num_rooms                    13446 non-null  int64
 13  num_beds                     13446 non-null  int64
 14  num_baths                    13446 non-null  float64
 15  rating                       11894 non-null  float64
 16  is_rating_present            13446 non-null  bool
 17  reviews                      12076 non-null  float64
 18  is_reviews_present           13446 non-null  bool
 19  rating_Cleanliness           11651 non-null  float64
 20  rating_Accuracy              11651 non-null  float64
 21  rating_Communication         11651 non-null  float64
 22  rating_Location              11651 non-null  float64
 23  rating_Check-in              11651 non-null  float64
 24  rating_Value                 11651 non-null  float64
 25  superhost                    13446 non-null  object
 26  is_superhost                 13446 non-null  bool
 27  house_rules                  13446 non-null  object
 28  house_rules_cleaned          13192 non-null  object
 29  is_house_rules_present       13446 non-null  bool
 30  num_of_amenities             13446 non-null  int64
 31  available_amenities          13446 non-null  object
 32  available_amenities_cleaned  13445 non-null  object
dtypes: bool(5), float64(9), int64(5), object(14)
```

Fig. 3. Cleaned Dataset columns

V. Exploratory Data Analysis

Exploratory Data Analysis (EDA) is the process of analyzing and summarizing a dataset to gain insights into its properties and characteristics. EDA was particularly important in the case of Airbnb data because it is a complex dataset that contains a wide range of information about the properties, including location, price, amenities, and ratings. We used various data visualizations and techniques to gain insights into the factors that affect the price of listings, the distribution of listings across different locations, and the most common amenities provided by hosts.

A. Initial Exploratory Visualizations

We first tried to visualize some of the data that we had. We plotted the count of each accommodation type in Fig. 4, and found that "Apartment" is the most frequently occurring listing, followed by "Private room" and "Home".

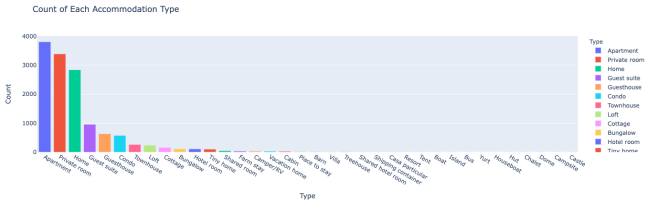

Fig. 4. Count of each accomodation type

Next, we counted the total number of amenities and visualized the top 20 in Fig. 5. We saw that these included "Kitchen", "Wifi" and "Dedicated workspace".

We then plotted the average number of amenities for each property type in Fig. 6. Our aim was to identify any potential trends here. Notably, we observed that as the accommodation type increased in size, there

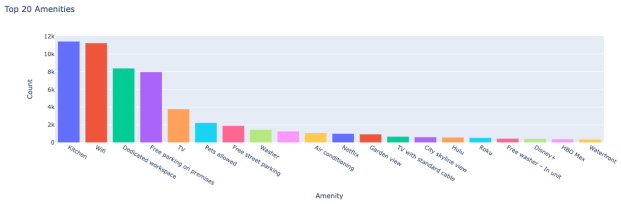

Fig. 5. Top 20 amenities

was a corresponding increase in the average number of amenities offered.

Fig. 6. Average number of amenities per property type

B. Feature Correlations

We then proceeded by creating a heatmap to visualize the correlations among all the features. In order to do this, we transformed all the categorical features into numerical representations. The heatmap in Fig. 7 revealed notable patterns in the top left and middle sections, indicating strong correlations. These correlations predominantly revolved around the ”price” and ”rating” features.

Upon analyzing the heatmap, we noticed that certain features that we initially anticipated to exhibit strong correlations with both ”price” and ”rating” – such as ”city” and ”coast” – displayed minimal or no significant correlation. To confirm this, we plotted the charts, Fig. 8, Fig. 9, Fig. 10 and Fig. 11 to observe the distribution of price and ratings across both city and coasts. However, from these plots, we could not observe any notable correlation between the features. So we decided to exclude both city and coast from the important features, and focus on those that we got from the heatmap.

We were also keen on exploring the relationship between price and rating through visualization (Fig. 12). However, we noted that the trend was not as distinct as anticipated, possibly due to the relatively limited amount of data we had collected. We suspect that with a more varied dataset, with a wider range of ratings and prices, we would likely observe a stronger correlation between price and ratings.

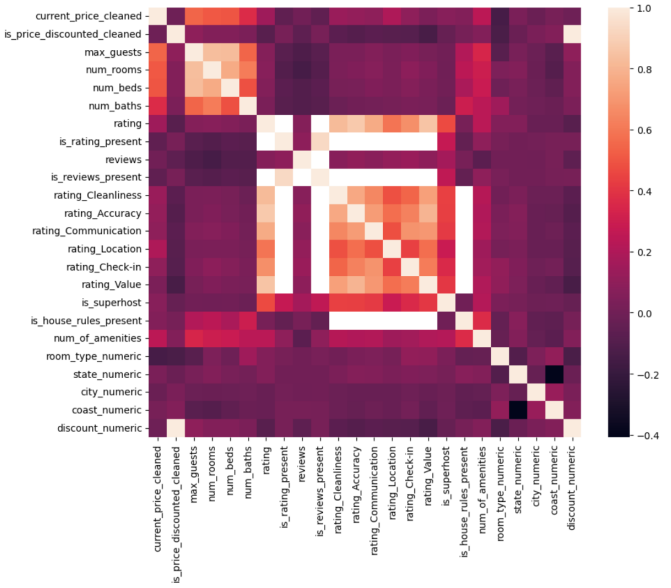

Fig. 7. Correlation Heatmap of all features

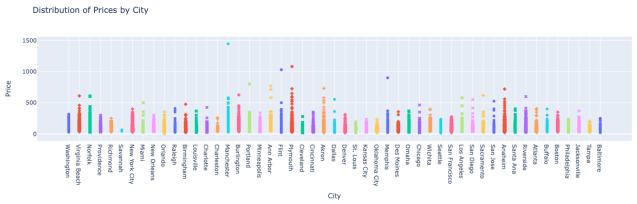

Fig. 8. Distribution of prices by city

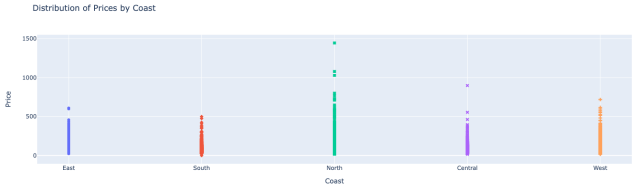

Fig. 9. Distribution of prices by coast

Fig. 10. Distribution of ratings by city

C. Correlations with Price

We utilized a correlation heatmap (Fig. 13) to visualize the relationships between the top 10 features and the

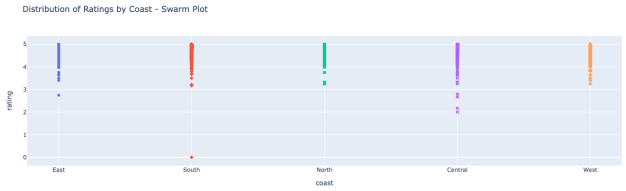

Fig. 11. Distribution of ratings by coast

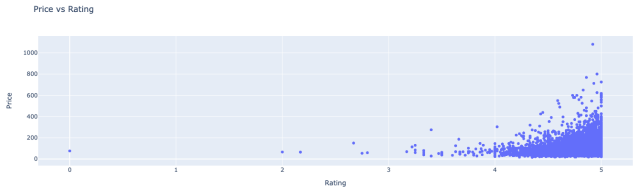

Fig. 12. Price vs Rating

target variable, which in this case was ”price.” Our analysis revealed that certain features, like the maximum number of guests, number of beds, number of rooms, and number of amenities, exhibited strong predictive power in determining the price. For further analysis, we plotted and examined pairplots in Fig. 14 to gain a better understanding of these relationships.

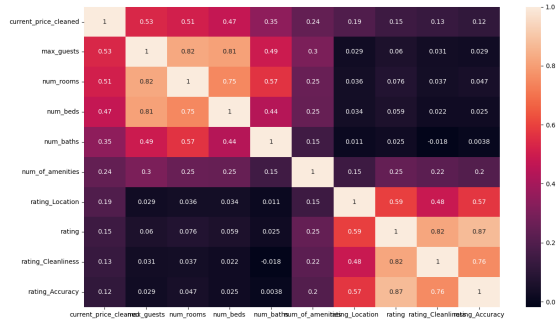

Fig. 13. Heatmap of top correlated features with Pricing

Consequently, we reached the conclusion that these top features could be effectively employed for price prediction in our models.

D. Correlations with Rating

We then repeated the above steps for considering rating as a target variable. In this case, from Fig. 15 we observed that the strongest predictors were different rating categories, along with superhost status. We again

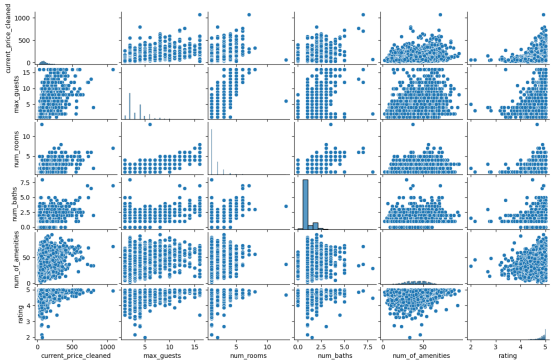

Fig. 14. Pairplot of top 10 features correlated with Pricing

looked at the pairplots of these top features (Fig. 16), and noticed clear trends that suggest strong correlation.

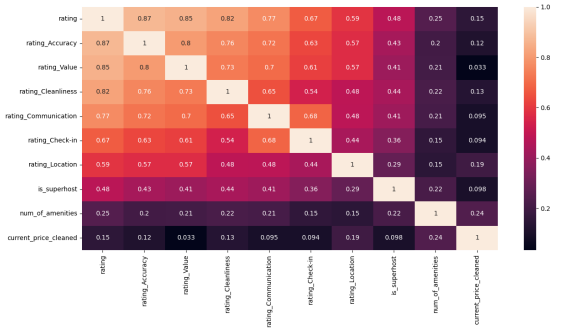

Fig. 15. Heatmap of top correlated features with Rating

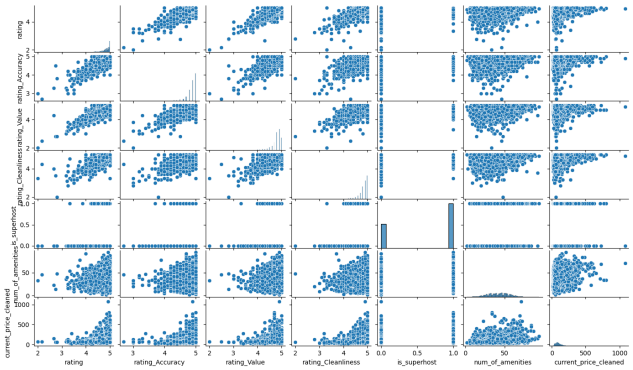

Fig. 16. Pairplot of top 10 features correlated with Rating

To qualify as a Superhost on Airbnb, hosts are required to maintain high ratings, likely exceeding a specific threshold. To determine this threshold, we examined the relationship between ratings and Superhost status, as

depicted in Fig. 17. Notably, we observed that the lowest rating among Superhosts was approximately 3.2. This finding implies that all Superhosts must maintain a rating above 3.2 to retain their status.

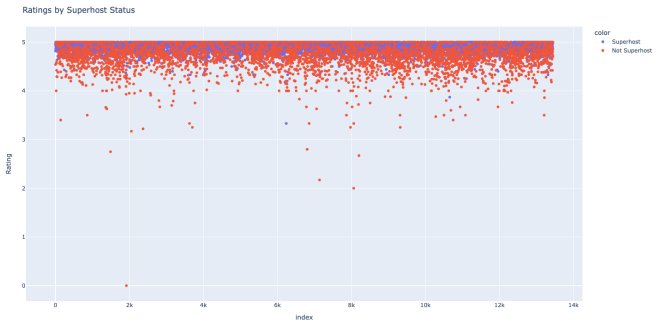

Fig. 17. Analysis of Ratings and Superhost status

Furthermore, we decided to proceed with the top features we got from the correlation heatmap as valuable indicators for prediction of rating in our models.

E. Selection of Target Variables

After conducting our data analysis, we made a deliberate choice to select two target variables for prediction: price and rating. While many previous approaches primarily focused on predicting price, we opted for a novel approach by also considering the prediction of ratings. By incorporating both price and rating as target variables, we aim to explore a comprehensive and unique perspective in our analysis.

VI. Machine Learning Models Used

We chose the following models for our experiments:

1) XGBoost (Extreme Gradient Boosting) - It is based on decision trees and uses a gradient descent algorithm to iteratively train multiple models, where each subsequent model corrects for the mistakes of the previous models. It also includes several regularization techniques to prevent overfitting and improve generalization.
2) LightGBM (Light Gradient Boosting Machine)- It uses a similar approach to XGBoost, but includes several optimizations such as histogram-based gradient boosting, which speeds up the training process by binning the continuous features into discrete values.
3) Linear Regression - It is a type of mathematical equation that helps us to predict a numerical value based on one or more input variables.

VII. Experimental Evaluation

In our research, we employed three models for two distinct experiments to predict the two target variables. Before conducting these experiments, we performed common pre-processing steps. This involved converting all relevant categorical variables into numerical values to ensure compatibility with the models. Additionally, any rows containing missing or NaN values in the rating or rating category columns were filtered out. Lastly, we removed any columns that did not contain numerical data, as they were not applicable for our modeling purposes. These pre-processing steps were applied consistently across both experiments to maintain data consistency and enhance the accuracy of our predictions.

Subsequently, the dataset was divided into separate reduced datasets for each experiment. When predicting price, we exclusively focused on the following columns: maximum number of guests, number of beds, number of rooms, number of bathrooms, number of amenities, rating, and current price of the listing. On the other hand, for predicting rating, we took into account the following columns: number of amenities, current price of the listing, superhost status, rating for accuracy, rating for cleanliness, and rating for value.

A. Predicting Price

To predict the target variable 'price', we initially divided our reduced dataset into training and testing sets, allocating 80% for training and 20% for testing. All the models were then trained using the training dataset. Subsequently, the performance of each model was assessed using the testing dataset, employing two evaluation metrics: Root Mean Square Error (RMSE) and R-squared (R2). We observed from Table I that XGBoost yielded superior results compared to LightGBM and Linear Regression (LR) models, exhibiting lower RMSE and higher R2 scores. Consequently, we plotted the feature importances derived from XGBoost (Fig. 18), revealing that the number of amenities was accorded the highest importance, followed by rating and the maximum number of guests. This suggests that these features strongly contribute to predicting the price accurately according to the XGBoost model.

B. Predicting Rating

Likewise, for the target variable "rating," we divided the reduced dataset into training and testing subsets. In this particular analysis, we observed that all three models produced identical RMSE scores as seen in Table II, although XGBoost achieved a slightly higher R2 score.

TABLE I
PERFORMANCE COMPARISON OF MODELS WITH PRICE AS TARGET VARIABLE

Model Name	RMSE Score	R2 Score
XGBoost	0.36	0.41
LightGBM	0.37	0.40
Linear Regression	0.39	0.33

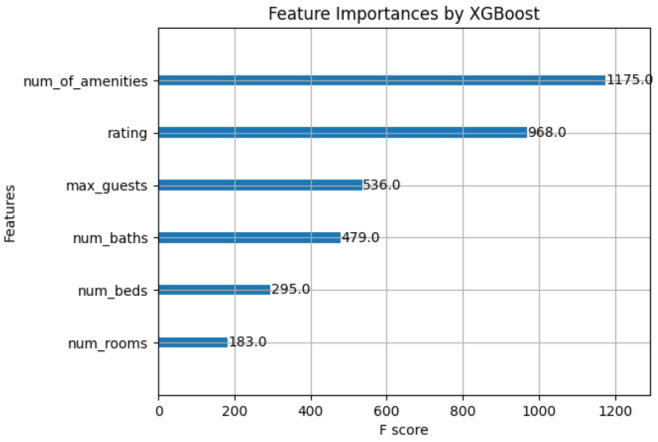

Fig. 18. Feature importances by XGBoost for Price

To gain further insights, we proceeded to plot the feature importances once more (Fig. 19). Surprisingly, we found that the number of amenities again received the highest importance, followed by price and rating for value. These results indicate that the number of amenities plays a crucial role in predicting the rating, while price and rating for value also contribute significantly according to the feature importances derived from the XGBoost model.

TABLE II
PERFORMANCE COMPARISON OF MODELS WITH RATING AS TARGET VARIABLE

Model Name	RMSE Score	R2 Score
XGBoost	0.07	0.86
LightGBM	0.07	0.85
Linear Regression	0.07	0.85

VIII. EXPERIMENTAL RESULTS

From our experiments, we were surprised to find that the XGBoost model emphasized the number of amenities as a crucial factor in predicting both price and rating. This result was contrary to our initial expectations, which hypothesized that features such as the number of bedrooms and the maximum number of guests allowed would hold greater importance for price prediction, while

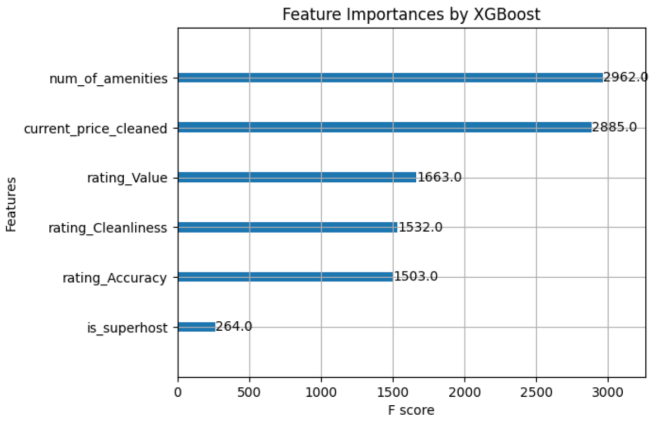

Fig. 19. Feature importances by XGBoost for Rating

rating category values would play a more significant role in predicting ratings. However, the prominence given to the number of amenities suggests the existence of a hidden relationship between this feature and our target variables, which was unveiled through gradient boosting.

IX. UNCOVERED INSIGHTS

Through our extensive exploratory data analysis and the utilization of feature importances provided by the XGBoost model, we were able to uncover and interpret the relationships between various variables in our dataset. The following key findings emerged from our analysis:

- The variable max_guests exhibited strong correlations with num_beds, num_rooms, num_baths, and price. This observation aligns with our intuition that larger accommodations tend to command higher prices.
- Among the rating sub-categories, 'Accuracy' demonstrated the most significant impact on average rating, followed by Value and Cleanliness. This highlights the importance of accurate descriptions provided by hosts in influencing customer satisfaction.
- We did not observe any notable correlation between price and review count, suggesting that the number of reviews alone does not have a significant influence on pricing.
- Superhost status did not appear to have a direct impact on price. However, there was a positive correlation between superhost status and ratings, which is consistent with our expectations that hosts require high ratings to become a Superhost.

- The number of amenities offered by an accommodation had a significant impact on price. This means that accommodations with more amenities tended to command higher prices.
- There was a discernible relationship between rating and amenities, suggesting that the availability of more amenities contributes to higher ratings.
- We found no negative correlations with price, suggesting that none of the variables in our dataset exhibited a detrimental impact on pricing.
- The number of guests an accommodation can accommodate was positively correlated with the number of amenities offered. Accommodations with a larger capacity tend to provide more amenities.
- City and Coast variables did not demonstrate a significant correlation with price.
- Since we only considered review count in our analysis, we could not find any relation with ratings. Performing sentiment analysis on the actual reviews could provide further insights into the relationship between review count and average ratings.
- Interestingly, we did not observe many poorly rated listings in our dataset, apart from newly listed and potentially fake listings. This could be attributed to Airbnb's algorithm displaying predominantly higher-rated accommodations during our data scraping process.

X. Conclusion

In the modern era of the sharing economy, platforms like Airbnb have introduced a new complexity to finding and pricing accommodations. Our research utilized Gradient-Boosted Decision Trees, specifically the XGBoost model, to predict Airbnb prices and determine high-demand locations. Sourced from extensive web-scraping of Airbnb listings across major U.S cities, our dataset captured a comprehensive range of factors, offering significant insights into price and rating determinants.

The XGBoost model outperformed the LightGBM and Linear Regression models, while predicting the target variables, and unveiling crucial correlations between features and their influence on accommodation prices and ratings. This work not only extends the understanding of the sharing economy platforms but also offers valuable tools to enhance the Airbnb experience for travelers and hosts alike. As the sharing economy continues to evolve, our research will help guide more informed decision-making in short-term rentals, providing practical value in navigating the vibrant Airbnb marketplace.

XI. Future Work

In this paper, we have presented a method for unearthing hidden insights from Airbnb data using gradient-boosted decision trees. Our method was able to achieve state-of-the-art results based on Airbnb data obtained through our custom web scraper. However, there are still a number of areas for future work.

One area for future work is integrating our method with a Maps API to determine walkability and transport scores. This would allow us to provide more accurate predictions of prices and ratings for Airbnb listings, given a specific locality or neighborhood.

Another area for future work is determining the impact of neighborhood characteristics on price and ratings. This would allow us to provide more personalized recommendations to Airbnb users.

Finally, we could also perform sentiment analysis of reviews to understand how users feel about different Airbnb listings. This would allow us to identify listings that are particularly popular or unpopular.

We believe these are just a few of the many ways our method can be improved in the future. We are excited to continue working on this project and to see what other insights we can uncover from Airbnb data.

In addition to the above, we could also explore the following:

- Using other machine learning algorithms to improve the accuracy of our predictions.
- Expanding our dataset to include more Airbnb listings from different cities and countries.
- Conducting user studies to understand how people use our predictions.
- Developing a web-based application that allows users to explore our predictions.

References

[1] J. H. Friedman, "Greedy Function Approximation: A Gradient Boosting Machine," Annals of Statistics, vol. 29, no. 5, pp. 1189–1232, Oct. 2001.

[2] A. Zhu, R. Li, and Z. Xie, "Machine Learning Prediction of New York Airbnb Prices," in 2020 Third International Conference on Artificial Intelligence for Industries (AI4I), Irvine, CA, USA, 2020, pp. 1–5.

[3] R.-T. Mora-Garcia, M.-F. Cespedes-Lopez, and V. R. Perez-Sanchez, "Housing Price Prediction Using Machine Learning Algorithms in COVID-19 Times," Land, vol. 11, no. 11, p. 2100, Nov. 2022, doi: 10.3390/land11112100.

[4] A. Ahuja, A. Lahiri, and A. Das, "Predicting Airbnb Rental Prices Using Multiple Feature Modalities," arXiv preprint arXiv:2112.06430, 2021.