Report On

Group Project

Airbnb Listing Clustering

INSY-5339: Principle of Business Data Mining
Fall 2025

Group: Backbencher

Members:

Bhuwan Bokati
Pawan Ojha
Ashwin Menezes

Airbnb Listing Clustering

Introduction

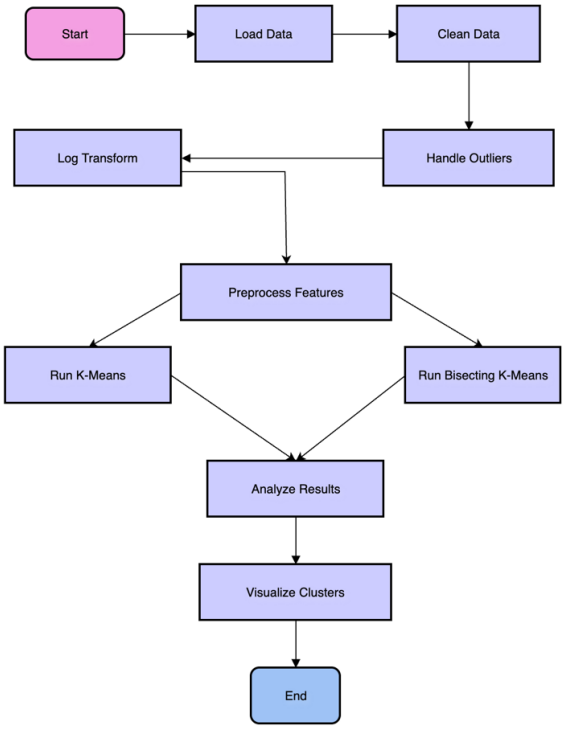
The project implements a pipeline for clustering Airbnb listings in New York City using two custom clustering algorithms: standard K-Means and Bisecting K-Means. This project analyzes Airbnb property listings using unsupervised learning to group similar rentals based on their features. By implementing a custom K-Means clustering algorithm, we aim to discover natural patterns.

Objectives

a. Develop a custom K-Means implementation from first principles
b. Implement a hierarchical Bisecting K-Means variant
c. Systematically determine optimal cluster counts
d. Generate actionable insights about Airbnb listing patterns

Mythology

The project employed an unsupervised learning pipeline to cluster Airbnb listings using a custom K-means algorithm and Bisecting K-Means. The methodology began with exploratory data analysis to identify correlations, missing values and spatial distributions. Data preprocessing included log transformations for skewed variables. RobustScaler normalization to handle outliers, and one-hot encoding for categorical features. Below flowchart shows main process of the project.



Data Exploration

Dataset Overview

The dataset contains 48,895 listings with 16 original features. After initial cleaning:

a. Removed 758 listings with missing/invalid values and outliers
b. Final cleaned dataset: 48137 listings
c. Key features retained: price, location, room type, neighborhood group, availability and review metrics

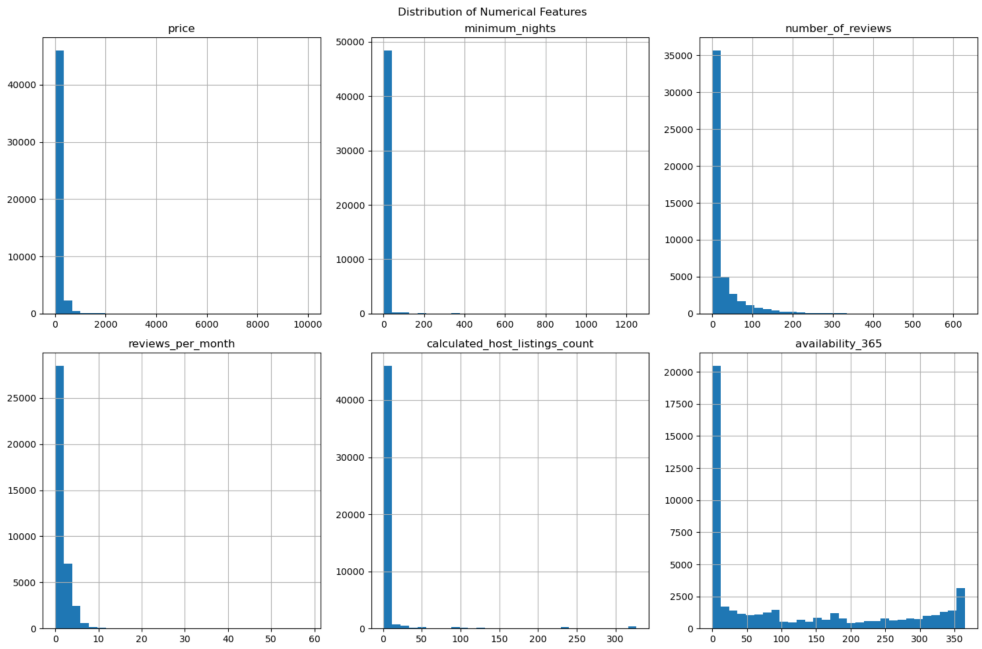
Missing Value Analysis

a. last_review: 20.6% missing (created new field called days_since_last_review and assigned max value in the column for missing ones and cutoff date taken as 2019-07-15 as the max date is 2019-07-08)
b. no_of_reviews and reviews_per_month: 20.6% missing (filled with 0)
c. host_name: 0.04% missing (column dropped)
d. name: 0.03% missing (column dropped)

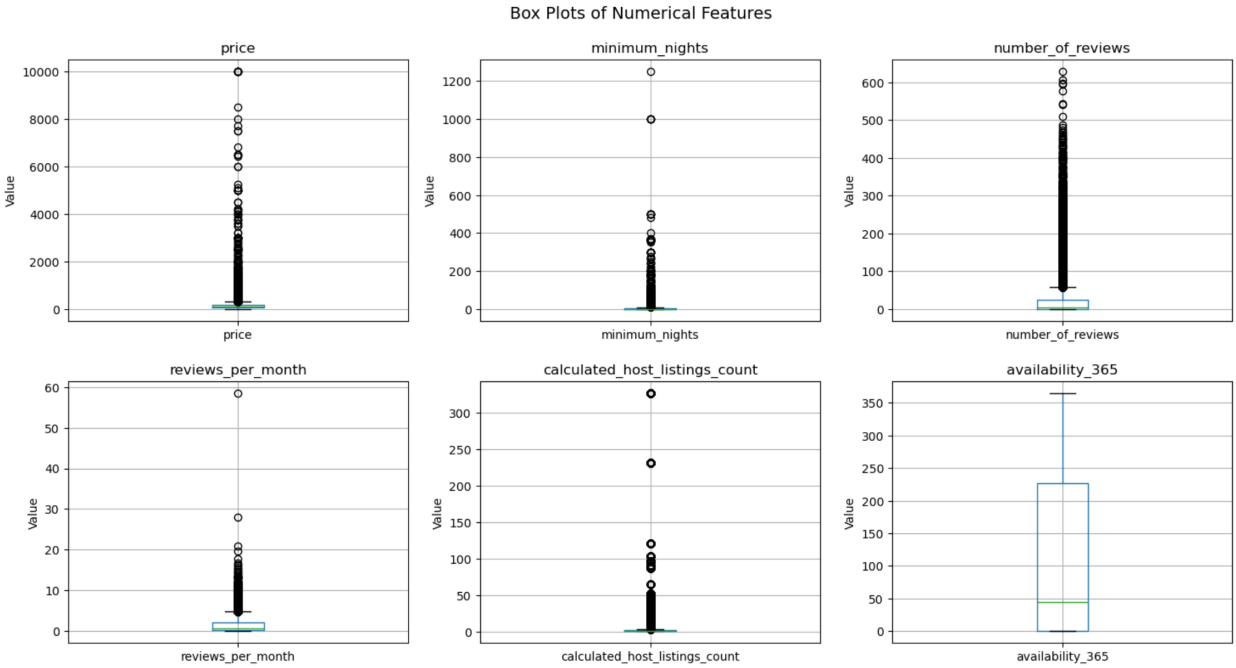
Visualization:

The data exploration crucial part is different visualization technique to detect outliers, skewness, majority and relationship between deferent features.

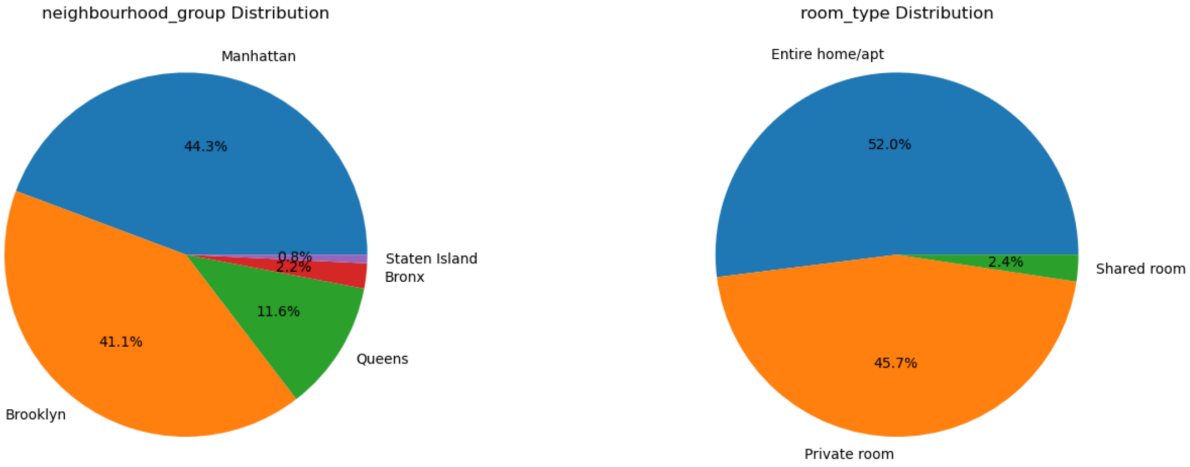
Histogram: Most of the numerical features are right skewed which need log transformation for normal spread of the data.



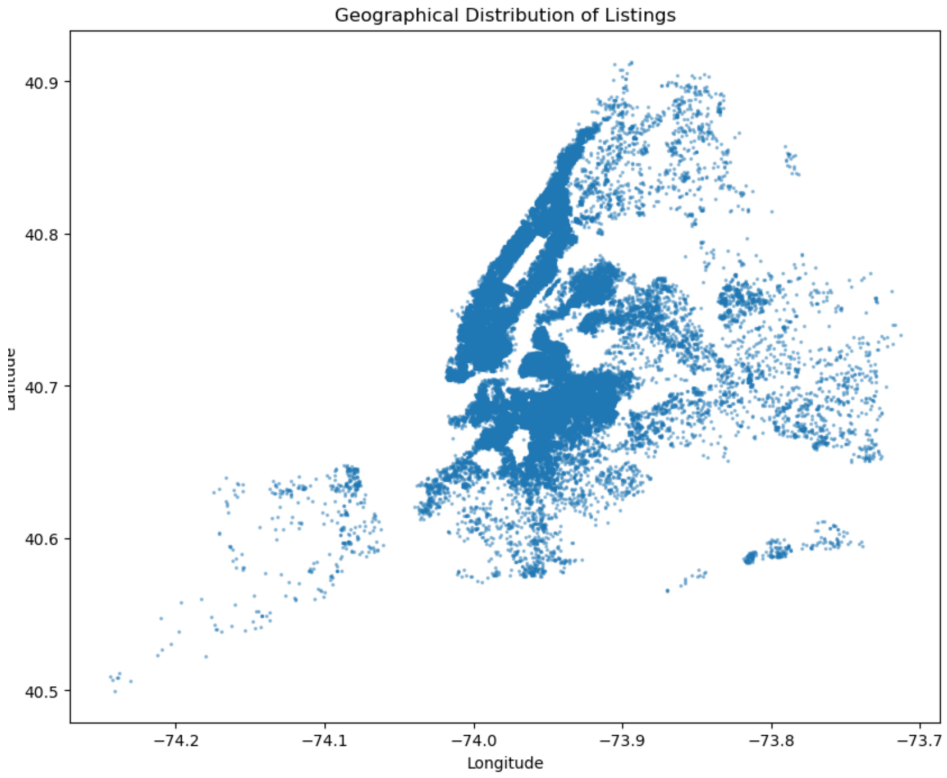
Boxplots: The box plots show the features expects availability_365 have outliers.



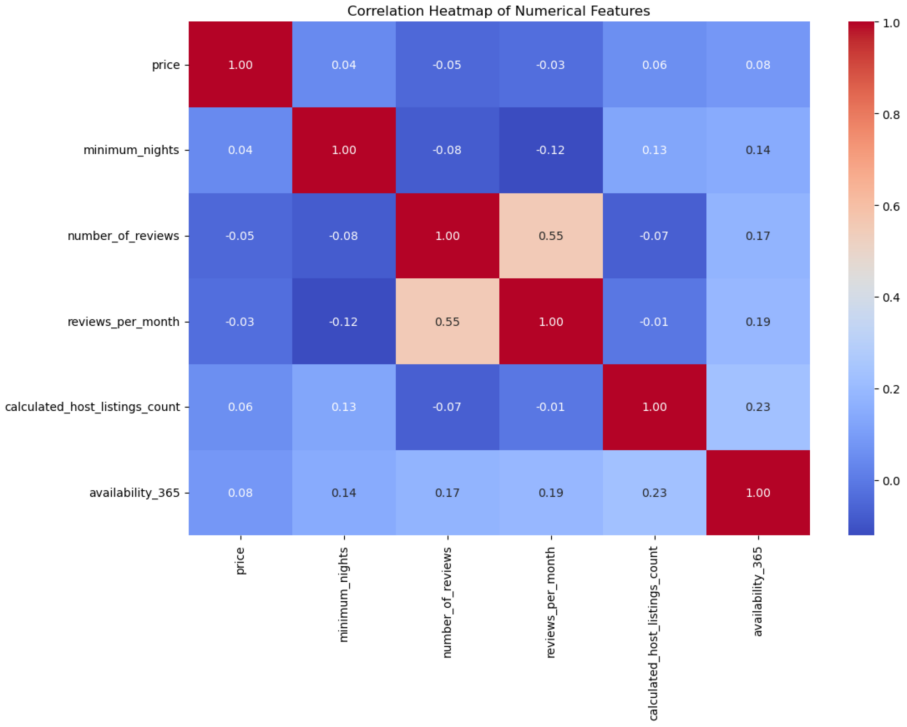
Pie charts: The pie charts show data portion in different categorical features.



Scatter Plot: The scatter plot shows the data point scatter among longitude and latitude.

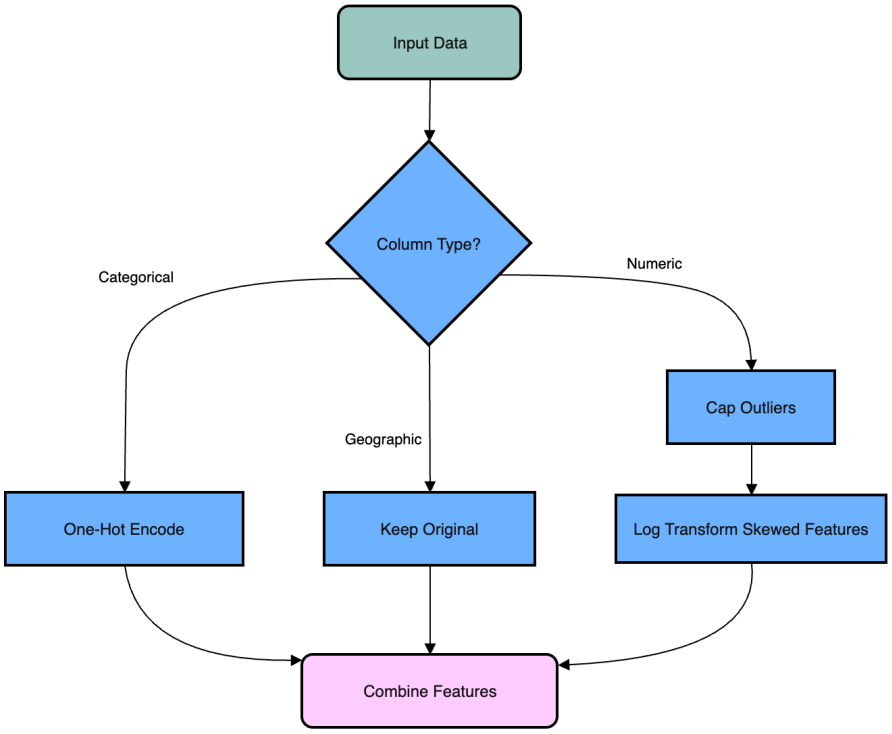


Correlation Heat Map: The numerical features are not highly correlated. Highly corelated features increased weight in distance calculations, potentially skewing cluster formation.



Preprocessing Pipeline

The basic flow chart of preprocessing of feature in this project is shown below:



Capping features:

The dataset was streamlined by dropping non-predictive columns including id, host_id, host_name, and name, as these unique identifiers offered no meaningful patterns for clustering. The project employs feature capping to handle extreme values in numerical features like price greater than zero and minimum_nights greater than zero and less than equal to thirty.

Feature Engineering

The days_since_last_review feature was engineered to quantify listing activity by calculating the time difference (in days) between the dataset's reference date (July 15, 2019) and each property's last review date.

Outlier Detection

The outlier detection function in this project uses a flexible, automated approach to handle extreme values that could distort clustering results. The code implements three robust strategies (IQR for skewed data, Z-score for normal distributions, and percentile clipping) with an intelligent auto mode that selects the best method based on statistical tests. For each numeric column, it calculates data-driven bounds (e.g., Q1-1.5IQR to Q3+1.5IQR for IQR method) and caps values beyond these thresholds, preserving dataset size while reducing outlier impact.

Transformation

The log transformation (log1p) was applied to right-skewed features like price and minimum_nights to normalize their distributions and reduce the impact of extreme values. By taking the natural logarithm of values plus one (log(x+1)), the transformation compresses the scale of high-magnitude numbers while preserving zeros/negative values, making the data more suitable for distance-based clustering algorithms like K-Means. This preprocessing step helps prevent skewed features from dominating the cluster analysis.

Scaling Methodology

The project employs feature scaling to standardize numerical features, ensuring fair comparison during clustering. For outlier-prone variables like price and minimum_nights, RobustScaler is used while geographic coordinates (latitude/longitude) are processed with StandardScaler since they follow near-normal distributions and MinMax maintains the data's distribution shape while ensuring equal feature contribution which is used for availability_365. This multiple approach prevents high-magnitude features from dominating distance calculations in K-Means, while maintaining meaningful spatial relationships.

a. RobustScaler: For price,minimum and reviews (resistant to outliers)
b. StandardScaler: For geographic coordinates
c. MinMaxScaler: For normalized percentages (availability_365)

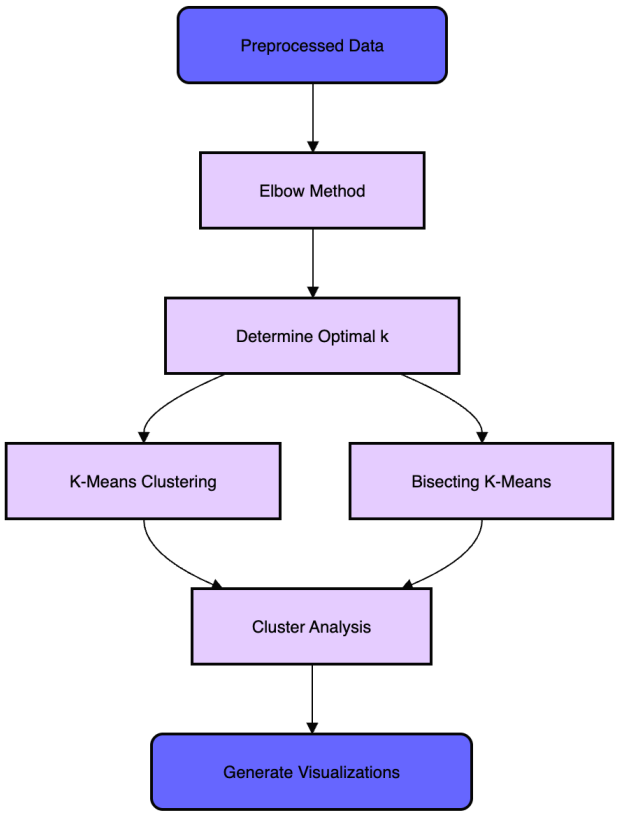
Encoding

One-hot encoding was applied to categorical features like room_type and neighbourhood_group to convert them into a numerical format suitable for machine learning algorithms. Since K-Means relies on distance calculations, categorical variables must be transformed into binary columns, where each category becomes a separate feature indicating presence or absence.

Clustering Implementation

For standard K-means, the core algorithm begins by randomly selecting initial centroids from the dataset. In each iteration, it calculates Euclidean distances between all points and centroids using vectorized operations for efficiency. The code assigns each point to its nearest centroid, then updates centroids as the mean of their assigned points. This loop continues until centroids stabilize (movement below tolerance) or max iterations is reached. The implementation includes tracking of inertia (total within-cluster SSE) for elbow analysis.

The bisecting K-means variant implements a hierarchical approach starting with one cluster containing all points. It maintains a list of current centroids and iteratively splits the cluster with highest SSE. For each split, it isolates the cluster's points and applies 2-means clustering. The implementation carefully manages cluster labels and centroid updates during splits, ensuring correct indexing as the cluster count grows. The splitting continues until reaching the target k clusters.

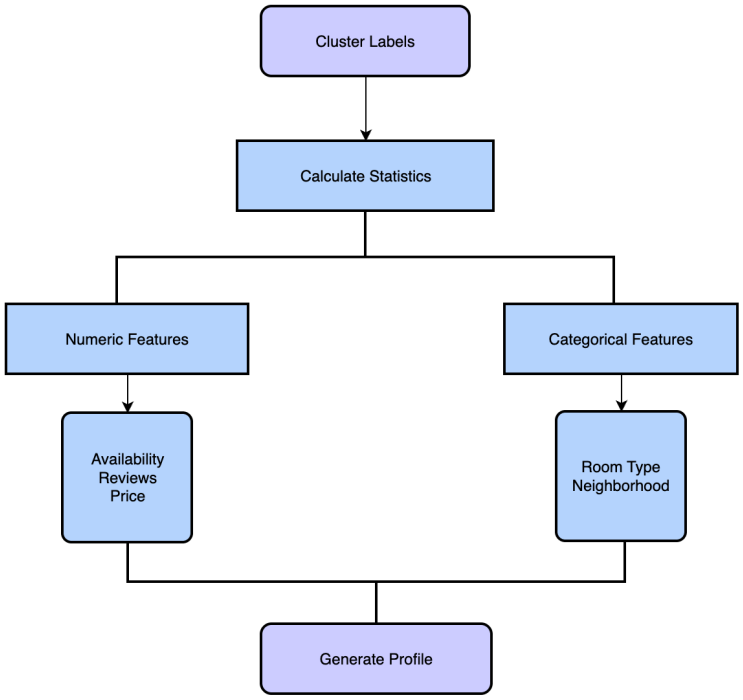


Cluster Evaluation and Optimization

The elbow method implementation systematically tests k values from 1 to 10. For each k, it runs the selected algorithm (standard or bisecting) and records the inertia. The code uses KneeLocator to automatically detect the optimal k from the inertia curve. For visualization, we reduce dimensions using PCA (to 2 components) and plot clusters with distinct colors. The implementation includes geographic mapping of clusters using the original latitude/longitude coordinates for spatial interpretation.

Result Analysis and Interpretation

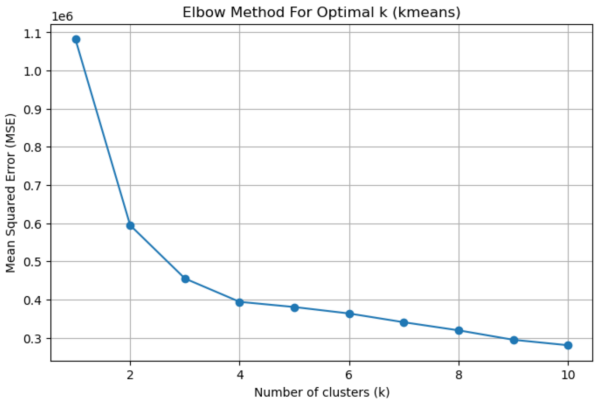
Post-clustering, the code generates comprehensive cluster profiles. For each cluster, it calculates key statistics: median price, most common room type, average availability, and review activity levels. These profiles help interpret the practical meaning of each cluster. The implementation also analyzes review patterns, categorizing listings as active, dormant or inactive based on days since last review. All results are formatted into clear tables showing cluster sizes, dominant characteristics, and behavioral patterns.

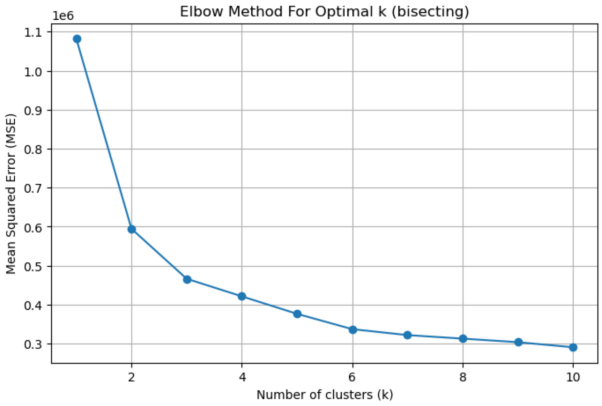


Flexible Feature Selection for Clustering

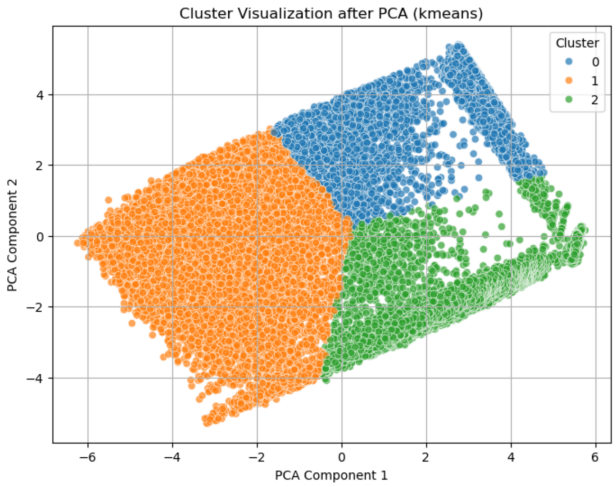
The code allows you to run customizable clustering on Airbnb listings by selecting specific columns for analysis. It supports geographic data (latitude/longitude), numeric features (price, reviews, availability), and categorical variables (room type, neighborhood), automatically preprocessing each type appropriately, log transforming skewed values, scaling numeric data, or one-hot encoding categories. By adjusting the selected_cols parameter, you can focus on key aspects like pricing trends, geographic distribution, or room-type preferences, ensuring the clustering results align with your analysis goals. The pipeline adapts dynamically, skipping missing columns and preserving interpretability, making it easy to experiment with different feature combinations.

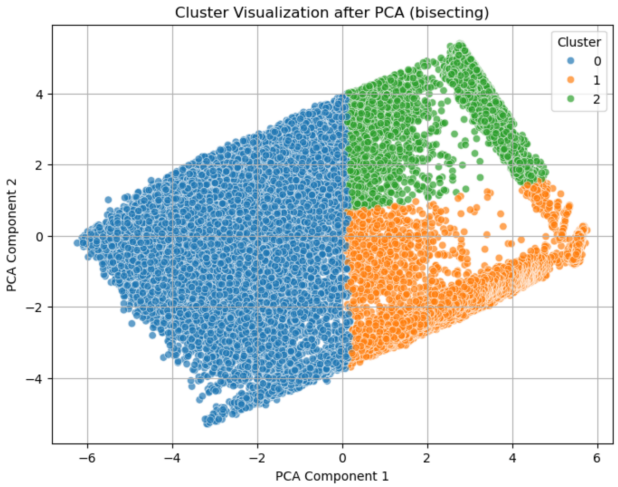
Elow Plot:



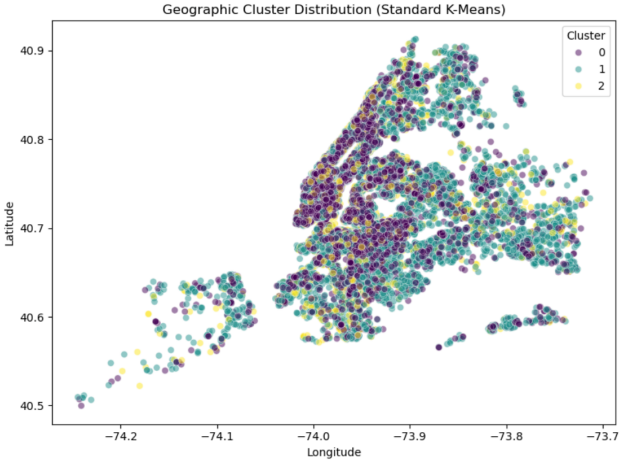


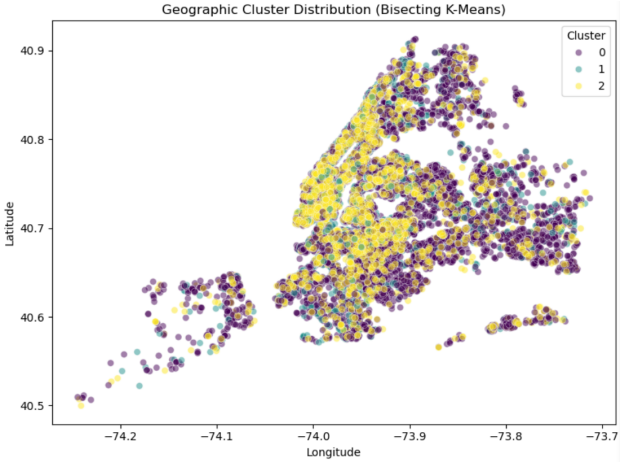
PCA Visualization:





Geographic cluster distribution:





Cluster Characteristics:

```
Optimal k: 3

=== Cluster Profiles (kmeans) ===
                  Size Avg_Price Min_Price Max_Price Usual_price  \
Cluster
0        8,063 (16.8%)      $130       $10      $334        $334
1       22,686 (47.1%)      $104       $10      $334        $334
2       17,388 (36.1%)      $100       $10      $334        $100

        Most_minimum_nights number_of_reviews Dominant_Room_Type  \
Cluster
0                        30                 0    Entire home/apt
1                         2                59    Entire home/apt
2                         1                 0    Entire home/apt

        Dominant_Neighbourhood Avg_Availability Most_Review_Activity_Status
Cluster
0                    Manhattan         217 days             Inactive (>1yr)
1                     Brooklyn         160 days               Recent (≤90d)
2                    Manhattan           0 days             Inactive (>1yr)

=== Review Activity (kmeans) ===
review_status  Inactive (>1yr)  Older (91–365d)  Recent (≤90d)
Cluster
0                        0.79             0.21           0.00
1                        0.00             0.07           0.93
2                        0.75             0.20           0.05
```

```
Optimal k: 3

=== Cluster Profiles (bisecting) ===
                  Size Avg_Price Min_Price Max_Price Usual_price  \
Cluster
0       25,053 (52.0%)      $105       $10      $334        $334
1       16,905 (35.1%)      $100       $10      $334        $100
2        6,179 (12.8%)      $143       $10      $334        $334

        Most_minimum_nights number_of_reviews Dominant_Room_Type  \
Cluster
0                         2                59    Entire home/apt
1                         1                 0    Entire home/apt
2                        30                 0    Entire home/apt

        Dominant_Neighbourhood Avg_Availability Most_Review_Activity_Status
Cluster
0                     Brooklyn         163 days               Recent (≤90d)
1                    Manhattan           0 days             Inactive (>1yr)
2                    Manhattan         208 days             Inactive (>1yr)

=== Review Activity (bisecting) ===
review_status  Inactive (>1yr)  Older (91–365d)  Recent (≤90d)
Cluster
0                        0.01             0.13           0.86
1                        0.77             0.21           0.02
2                        0.97             0.03           0.00
```

K-means Clustering:

Cluster 0 (16.8%):

a. High-priced listings (Avg $130) with long minimum stays (30 nights).
b. Inactive hosts (79% inactive >1yr), mostly Manhattan entire homes.
c. Low availability (217 days/year), suggesting seasonal or infrequent rentals.

Cluster 1 (47.1%):

a. Moderate prices (Avg $104), short stays (2-night min), and frequent reviews (59 avg).
b. Active hosts (93% recent reviews), mostly Brooklyn entire homes.
c. Moderate availability (160 days/year), likely professionally managed.

Cluster 2 (36.1%):

a. Lowest prices (Avg $100), short stays (1-night min), no reviews.
b. Inactive hosts (75% inactive >1yr), mostly Manhattan entire homes.
c. Zero availability, suggesting listings may be inactive or delisted.

Bisecting k-means Clustering:

Cluster 0 (52.0%):

Like Cluster 1 (k-means): Moderate prices ($105), short stays, active hosts (86% recent reviews), Brooklyn-focused.

Cluster 1 (35.1%):

Like Cluster 2 (k-means): Low prices ($100), short stays, inactive hosts (77% inactive), Manhattan-based, zero availability.

Cluster 2 (12.8%):

High-priced ($143), long stays (30-night min), extremely inactive (97% >1yr), Manhattan-based, low availability (208 days).

Key Observations

Two distinct "active" vs. "inactive" host groups:

Active hosts (k-means Cluster 1 / bisecting Cluster 0):

a. Lower prices, frequent reviews, short stays.
b. Likely professional hosts optimizing for occupancy.

Inactive hosts (k-means Clusters 0 & 2 / bisecting Clusters 1 & 2):

a. Either high-priced (long-term rentals) or low-priced (possibly abandoned listings).
b. Minimal reviews and high inactivity rates.

Manhattan vs. Brooklyn Divide:

a. Manhattan: Dominates inactive, high-priced, or zero-availability listings.
b. Brooklyn: More active, affordable, and frequently reviewed listings.

Pricing Strategy Differences:

a. Active listings cluster around (100-105).
b. Inactive listings vary, some are premium-priced (130-143), others are low-cost but abandoned.

Business Implications

a. Encourage inactive hosts (especially zero-availability ones) to update listings or delist.
b. Promote Brooklyn listings as they seem more reliable and frequently booked.
c. Investigate high-priced, low-activity clusters could be illegal rentals or speculative listings.

For Hosts:

a. Active hosts: Maintain competitive pricing (100–105) and responsiveness to attract bookings.
b. Inactive hosts: Consider repricing or improving availability to regain visibility.

For Travelers:

a) For short stays: Brooklyn listings are more reliable.
b) For long stays: High-priced Manhattan listings exist but may have inactive hosts.

Algorithm Comparison

K-means vs. Bisecting k-means: Both methods identified similar groups, but bisecting k-means split the inactive listings more clearly (separating high-priced vs. low-priced inactive clusters).

Conclusion

The data shows a clear split in Airbnb listings between two types. On one side, there are active, affordable rentals mostly in Brooklyn with frequent bookings and good host engagement. On the other side, there are inactive or overpriced listings often in Manhattan that get little attention and may even be abandoned. To improve the platform, Airbnb should address these inactive listings, while travelers looking for reliable stays may want to focus on Brooklyn options.

Team Contributions

Feature Engineering & Clustering (Bhuwan Bokati):

- Created advanced data transformation processes to normalize different types of information
- Designed and implemented a custom grouping algorithm with intelligent starting point selection
- Incorporated an automated method to determine the ideal number of groupings
- Optimized the core algorithm for efficiency and accuracy

Data Preprocessing & Cleaning (Pawan Ojha):

- Led the initial data cleaning process by addressing incomplete entries and removing unnecessary information
- Developed sophisticated techniques to identify and properly handle unusual data points
- Established protocols to maintain the integrity of location-based information throughout processing
- Ensured all data met quality standards before analysis

Visualization & Analysis (Ashwin Menezes):

- Developed intuitive graphical representations of the grouping results
- Created mapping solutions to display geographical patterns in the data
- Designed comprehensive reporting of group characteristics and statistics
- Structured clear summaries highlighting key findings and patterns