Bhartiya Vidya Bhavan's

Sardar Patel Institute of Technology

(Autonomous Institute Affiliated to University of Mumbai)

Department of Computer Engineering

Hospitality

By:

By Hrishikesh Patil (2023300168)
Shreesh Patil (2023300170)
Yash Patil (2023300172)
Adwait Patkhedkar (2023300174)

Guided by

Prof. Abhijeet Salunkhe

Course Project

Python for Data Science (S.Y.)

Abstract

In this project, we analyze an Airbnb dataset with the goal of understanding key patterns in the data and predicting room prices. The project focuses on three main objectives: data cleaning, exploratory data analysis (EDA), and predictive modeling.

The data cleaning phase involves handling missing values, outlier detection and removal, feature engineering, and ensuring the dataset is ready for analysis. Following this, EDA is conducted to uncover trends and relationships in the data, such as identifying factors influencing pricing, host behavior, and geographical patterns. Tools like visualizations and statistical summaries are employed to make insights more accessible.

The final stage involves building a machine learning model to predict room prices. Using features derived from the cleaned dataset, various regression algorithms are tested and optimized. Performance metrics such as RMSE and R-squared are used to evaluate model accuracy.

The analysis concludes by highlighting the significant factors influencing room prices and the effectiveness of the predictive model. The results provide actionable insights for hosts to optimize pricing and for guests to make informed booking decisions. This project demonstrates the integration of data cleaning, analysis, and predictive techniques in a real-world scenario.

.

Introduction

The Airbnb dataset contains extensive information about rental properties, but raw data is often noisy and difficult to interpret. Understanding factors influencing room pricing and predicting prices accurately are crucial tasks for both hosts and potential guests. This project addresses the challenges of cleaning the dataset, analyzing patterns, and building predictive models to estimate room prices based on property features. The scope includes data preprocessing, exploratory analysis, and implementing regression models for price prediction.

Objective:
The primary goal of this project is to predict room prices based on property characteristics and location data. It involves:

1. Cleaning the dataset to ensure high-quality inputs.
2. Performing exploratory data analysis (EDA) to uncover meaningful patterns.
3. Building and evaluating machine learning models for accurate price prediction.

Motivation:
The rise of Airbnb as a popular platform for short-term rentals has led to the need for data-driven insights. Accurate price prediction can assist hosts in setting competitive rates while enabling guests to make cost-effective decisions. Moreover, understanding key factors influencing pricing can contribute to better market strategies and improved customer experiences. This project combines data cleaning, analysis, and modeling, making it a valuable learning opportunity for data enthusiasts and a practical application in the hospitality domain.

Dataset

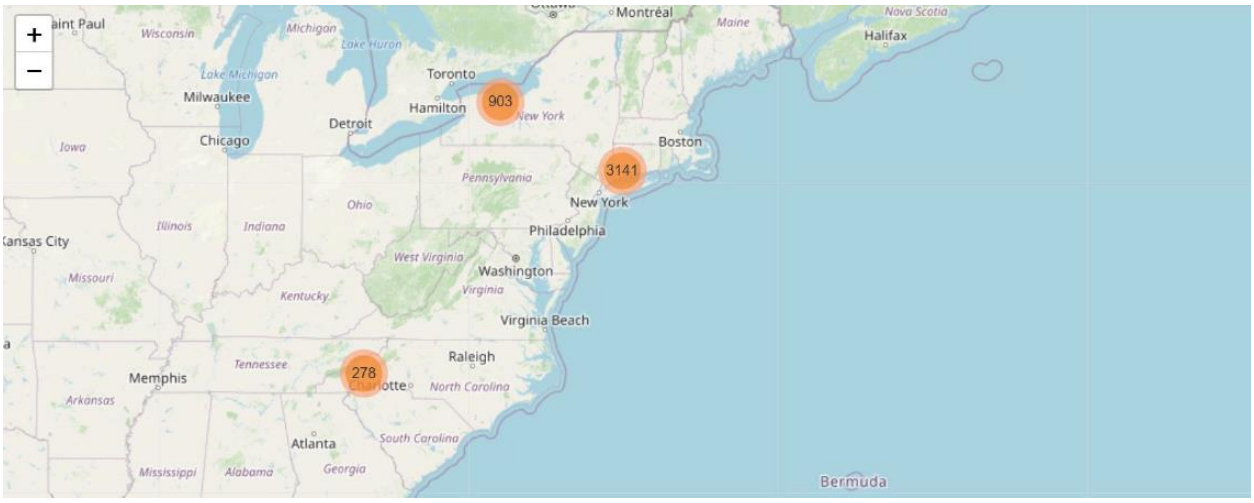
- **Description of Dataset**: Data source: https://insideairbnb.com/get-the-data/
  Inside Airbnb is a mission driven project that provides about Airbnb's impact on residential communities

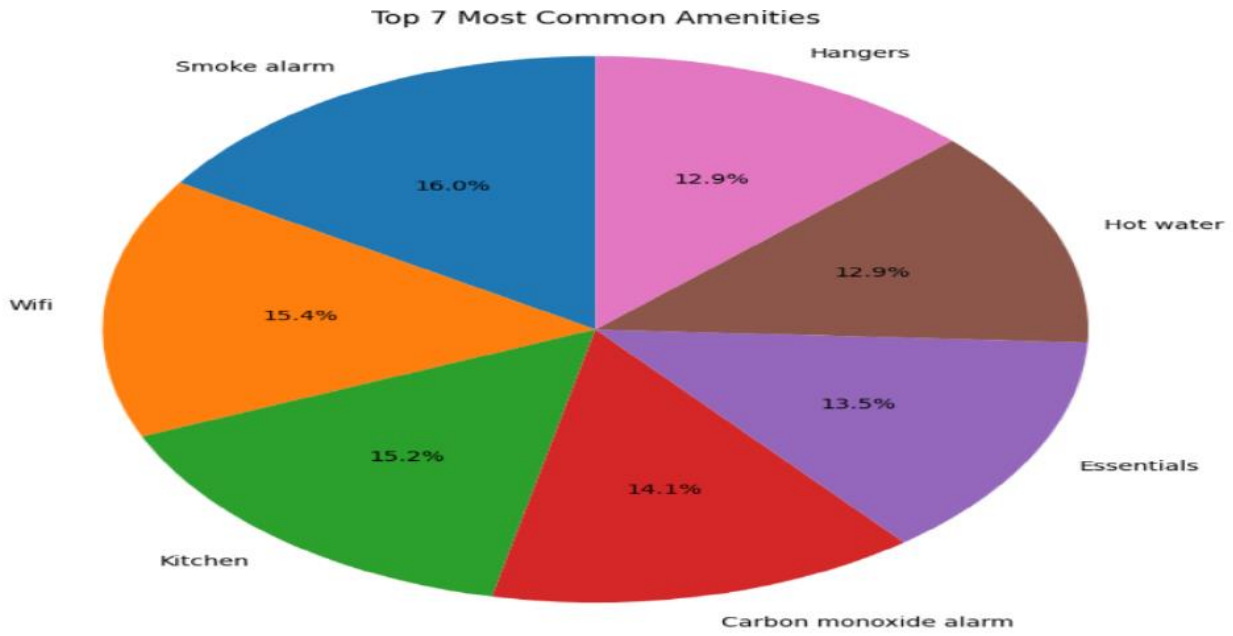
  Format: CSV files of 36 US cities (listings.csv.gz)
  These cities were split according to various regions of the US

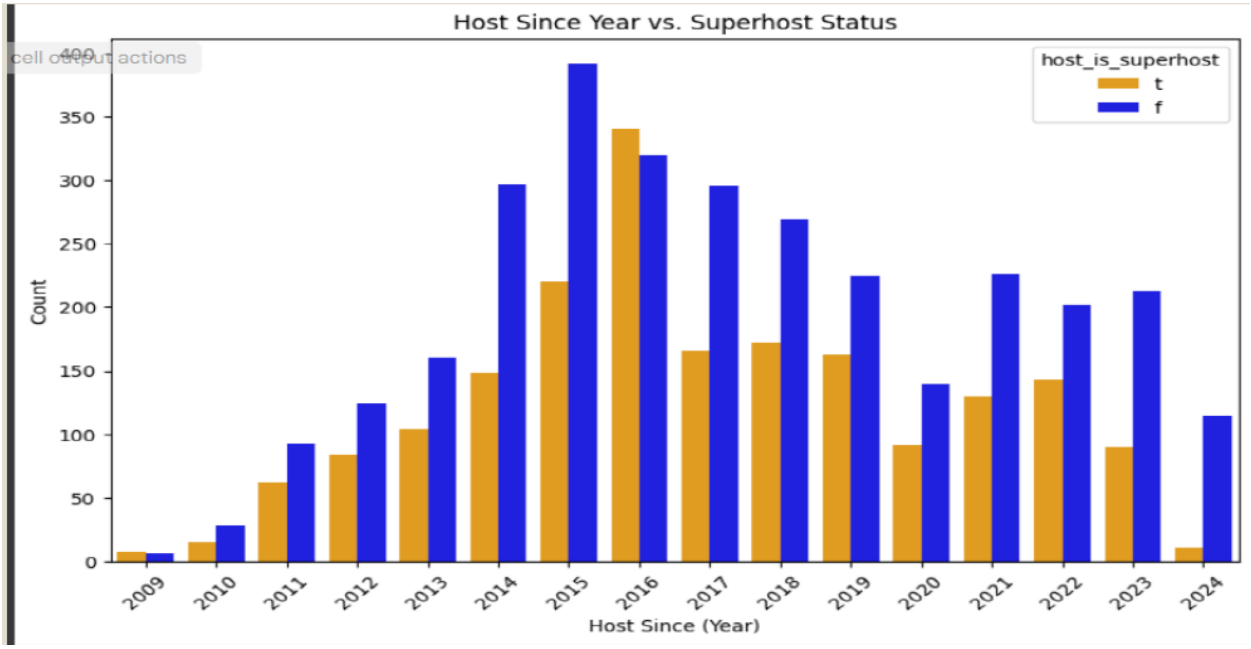
  76 features with a mix of text, float and int

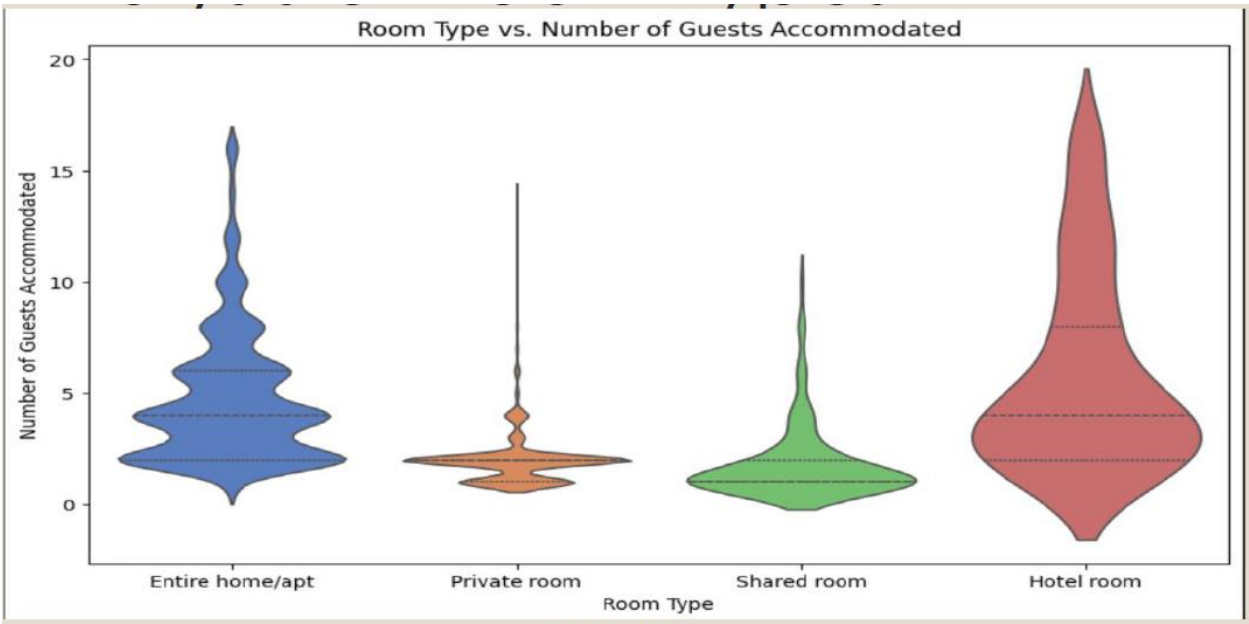
  Stratified sampling was used for the dataset to avoid overrepresentation of datasets with very high number of rows

- **Preprocessing**:
  - Feature selection was performed
  - Missing data was divided into categorical and numeric features. Various statistical imputations were chosen acc to requirement
  - Structural inaccuracies were corrected particularly for dates, columns containing % signs and columns which were numeric but present as object datatype
- **Data Exploration**:

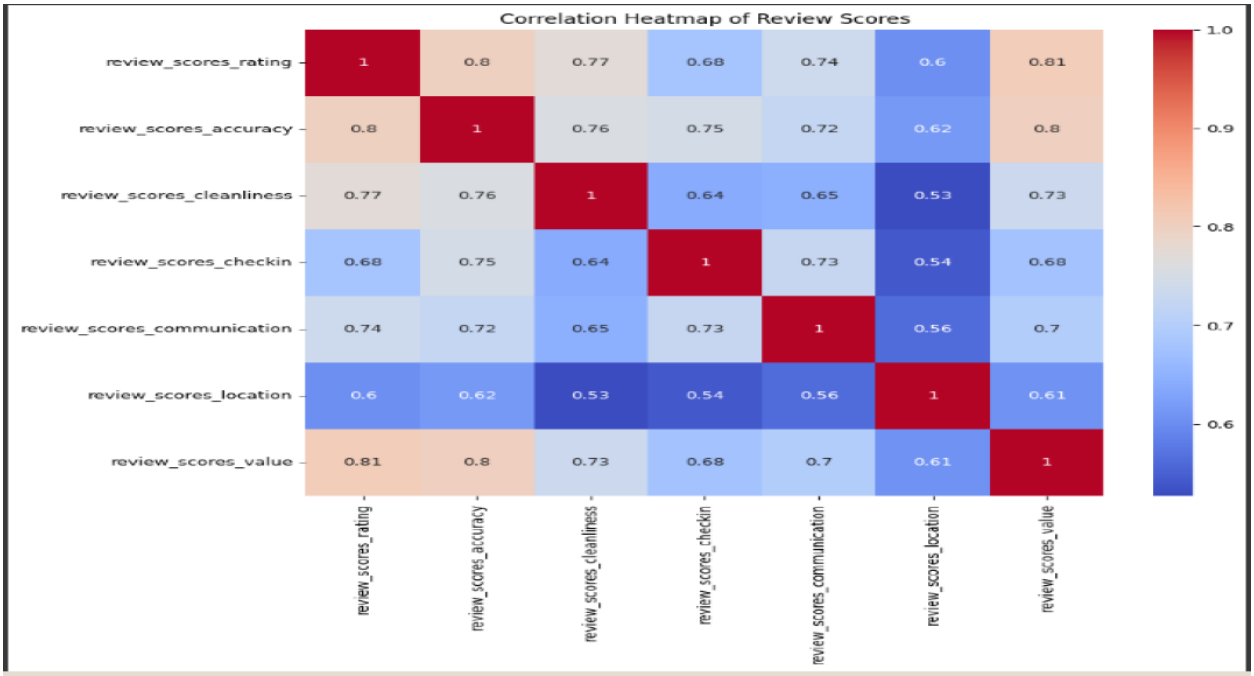



Top 7 Most Common Amenities
Smoke alarm
Hangers
16.0%
12.9%
Hot water
12.9%
Wifi
15.4%
13.5%
Essentials
15.2%
14.1%
Kitchen
Carbon monoxide alarm


Host Since Year vs. Superhost Status
cell output actions
host_is_superhost
t
f
Count
400
350
300
250
200
150
100
50
0
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
Host Since (Year)



Violin Plot



Methodology

Models Used:

- **Linear Regression:** A basic regression model used to understand the relationship between features and target variable (room prices).
- **Random Forest Regressor:** An ensemble learning method that uses multiple decision trees to improve accuracy and reduce overfitting.

Justification:

- **Linear Regression:** Chosen for its simplicity and interpretability, providing a baseline for model performance. It is effective for understanding the direct relationships between features and the target variable.
- **Random Forest Regressor:** Selected due to its ability to handle non-linear relationships and capture interactions between features. It is robust to outliers and often performs well with a variety of datasets.

---

2. Model Implementation

Training and Testing:

- The dataset was split into **80% for training** and **20% for testing** to evaluate the models.
- Cross-validation was applied to ensure the models generalized well to unseen data, reducing the risk of overfitting.

Hyperparameter Tuning:

- For **Linear Regression**, no hyperparameter tuning was necessary since it has minimal tunable parameters.
- For **Random Forest Regressor**, hyperparameter optimization was performed using **Grid Search**. The key parameters tuned included:
    - Number of trees (n_estimators)
    - Maximum depth of trees (max_depth)
    - Minimum samples required to split a node (min_samples_split)
    - Minimum samples required at a leaf node (min_samples_leaf)
- The optimal combination of parameters was selected based on cross-validation performance.

---

3. Feature Selection and Extraction

Feature Selection:
Feature selection was not explicitly performed for this project. Instead, all relevant features from the cleaned dataset were included in the models.

Justification for Approach:

- Since the dataset was already preprocessed with relevant features, additional selection techniques were not applied.
- Random Forest inherently handles feature importance by assigning weights to features during training, which helps in identifying key contributors to room price prediction.

Impact on Performance:

- Using all relevant features ensured that no potentially useful information was excluded.
- The Random Forest model benefited from its ability to prioritize significant features automatically, contributing to better predictive accuracy compared to Linear Regression.

Experimental Setup

- **Tools Used**:

Programming Language:

Python: For data manipulation, analysis, and modeling.

Data Manipulation and Analysis Libraries:

Pandas: For data cleaning, transformation, and analysis.

NumPy: For numerical computations and handling arrays.

Visualization Libraries:

Matplotlib: For creating static plots and charts.

Seaborn: For advanced statistical data visualization.

Folium: For interactive geospatial visualizations.

Machine Learning and Modeling Libraries:

Scikit-learn: For implementing regression models, feature selection, and evaluation.

Statistical Tools:

Statsmodels: For statistical analysis and insights.

Data Preprocessing and Feature Engineering:

Scikit-learn: For feature scaling, encoding, and splitting data into training and test sets.

Development Environment:

Jupyter Notebook, Colab: For interactive coding and documentation.

- **Hardware/Environment**: Google Colab
- **Evaluation Metrics**:

Mean Squared Error (MSE):

- **Definition:** MSE is the average of the squared differences between the actual and predicted values.
- **Formula:** MSE is calculated as the sum of the squared differences between the observed and predicted values, divided by the total number of observations.
- **Purpose:** It penalizes larger errors more than smaller ones, making it useful for identifying significant deviations. Lower MSE indicates a better fit.

R-squared (R^2):

- **Definition:** R^2 measures the proportion of variance in the dependent variable that is predictable from the independent variables.
- **Formula:** R^2 is computed as one minus the ratio of the sum of squared residuals (differences between actual and predicted values) to the total sum of squares (variance of the observed values).
- **Purpose:** R^2 ranges from 0 to 1, where higher values indicate that the model explains more of the variance. It provides a general measure of model goodness-of-fit.

Mean Absolute Error (MAE):

- **Definition:** MAE is the average of the absolute differences between the actual and predicted values.
- **Formula:** MAE is calculated as the sum of the absolute differences between the observed and predicted values, divided by the total number of observations.
- **Purpose:** Unlike MSE, MAE treats all errors equally, providing an intuitive measure of model accuracy. Lower MAE values indicate a better fit.

Results and Discussion

Random Forests

```
Mean Absolute Error (MAE): 46.13085319562563
Mean Squared Error (MSE): 4522.8209823979905
R-squared (R²): 0.5637374210295685
```

.

Linear Regression:

```
Mean Absolute Error (MAE): 61.855126165043 84
Mean Squared Error (MSE): 6265.6243817240775
R-squared (R²): 0.3956299702621243
```

Based on the provided evaluation metrics, we can compare the performance of Random Forest and Linear Regression models:

Random Forest is an ensemble learning method that combines multiple decision trees to make predictions. It works by creating a large number of decision trees, each trained on a different subset of the training data, and then averaging their predictions. This approach reduces overfitting and improves the model's generalization performance.

Linear Regression is a statistical method used to model the relationship between a dependent variable and one or more independent variables. It assumes a linear relationship between the variables and aims to find the best-fitting line that minimizes the sum of squared errors.

Random Forest:

Mean Absolute Error (MAE): Lower MAE indicates better accuracy. Random Forest generally performs well in terms of MAE, suggesting it's better at predicting the average error.

Mean Squared Error (MSE): Lower MSE is better. Random Forest has a lower MSE, indicating it's more accurate in predicting the actual values.

R-squared (R^2): Higher R-squared indicates a better fit. Random Forest has a higher R-squared, suggesting it explains more variance in the data.

Linear Regression:

Mean Absolute Error (MAE): Higher than Random Forest, indicating less accurate predictions on average.

Mean Squared Error (MSE): Higher than Random Forest, suggesting larger prediction errors.

R-squared (R^2): Lower than Random Forest, indicating a weaker fit to the data.

Potential Reasons for Underperformance and Improvements:

1. Overfitting/Underfitting:

Overfitting: If the model is too complex, it may fit the training data too closely, leading to poor performance on unseen data.

Regularization techniques: L1 and L2 regularization can help reduce overfitting.

Early stopping: Stop training when the validation loss starts increasing.

Underfitting: If the model is too simple, it may not capture the underlying patterns in the data.

Increase model complexity: Add more layers or features.

Collect more data: A larger dataset can help the model learn more complex patterns.

2. Feature Engineering:

Feature selection: Identify and select the most relevant features to improve model performance.

Feature creation: Create new features by combining or transforming existing ones.

Feature scaling and normalization: Normalize features to a common scale.

3. Hyperparameter Tuning:

Experiment with different hyperparameters for the models to find the optimal configuration.

Use techniques like grid search or random search to explore the hyperparameter space.

4. Data Quality:

Missing values: Handle missing values appropriately (e.g., imputation, deletion).

Outliers: Identify and handle outliers to avoid their negative impact on the model.

Data cleaning: Remove inconsistencies and errors in the data.

5. Model Selection:

Ensemble methods: Consider using ensemble methods like Gradient Boosting or XGBoost, which often outperform individual models.

Deep learning: For complex tasks, deep learning models can be effective.

6. Model Evaluation:

Use appropriate evaluation metrics (e.g., accuracy, precision, recall, F1-score) based on the problem type.

Cross-validation can help assess the model's generalization performance.

By addressing these potential issues and carefully considering the model's performance metrics, you can improve the accuracy and reliability of your machine learning models.

Conclusion

The project aimed to analyze Airbnb data by addressing challenges in data quality, uncovering insights through exploratory data analysis (EDA), and predicting room prices using machine learning models. The approach involved cleaning the dataset to handle missing and inconsistent values, conducting EDA to identify patterns and relationships, and implementing regression algorithms to model pricing behavior. By evaluating models using metrics like MSE, MAE, and R^2, the performance of each was assessed to identify the best-fit model for price prediction.

Findings:

The analysis highlighted significant factors influencing room pricing, such as location, property type, and amenities. The best-performing model demonstrated strong predictive accuracy, providing reliable estimates for room prices.

Limitations:

Data Size: The dataset may not comprehensively represent all Airbnb listings, limiting the generalizability of findings.

Bias in Data: The data might contain biases due to missing or outdated information, influencing predictions.

Feature Scope: Some potentially influential features, such as seasonal trends or external factors like local events, were not included.

Model Constraints: The models may not fully capture non-linear or complex interactions among features.

Future Work:

Incorporate additional data, such as seasonal trends, user reviews, and local demand indicators, to improve model performance.

Experiment with advanced algorithms, like ensemble methods or deep learning, for better predictions.

Enhance feature engineering by exploring interactions and non-linear transformations.

Validate the model on larger and more diverse datasets to ensure robustness and scalability.

Develop a user-friendly interface for stakeholders to utilize the predictive model.

Timesheet

Timesheet

Data Acquisition and Preparation (9 hours):

- Finding a relevant dataset (1 hour)
- Defining a data splitting strategy (1 hour)
- Handling missing values, structural inaccuracies, and feature selection (7 hours)

Data Exploration and Visualization (7 hours):

- Visualizing data to gain insights

Model Training (2 hours):

- Training a machine learning model

Feature Engineering (1 hour):

- Reducing dimensionality and extracting features

Report Generation (1 hour):

- Creating a report summarizing the findings and methods

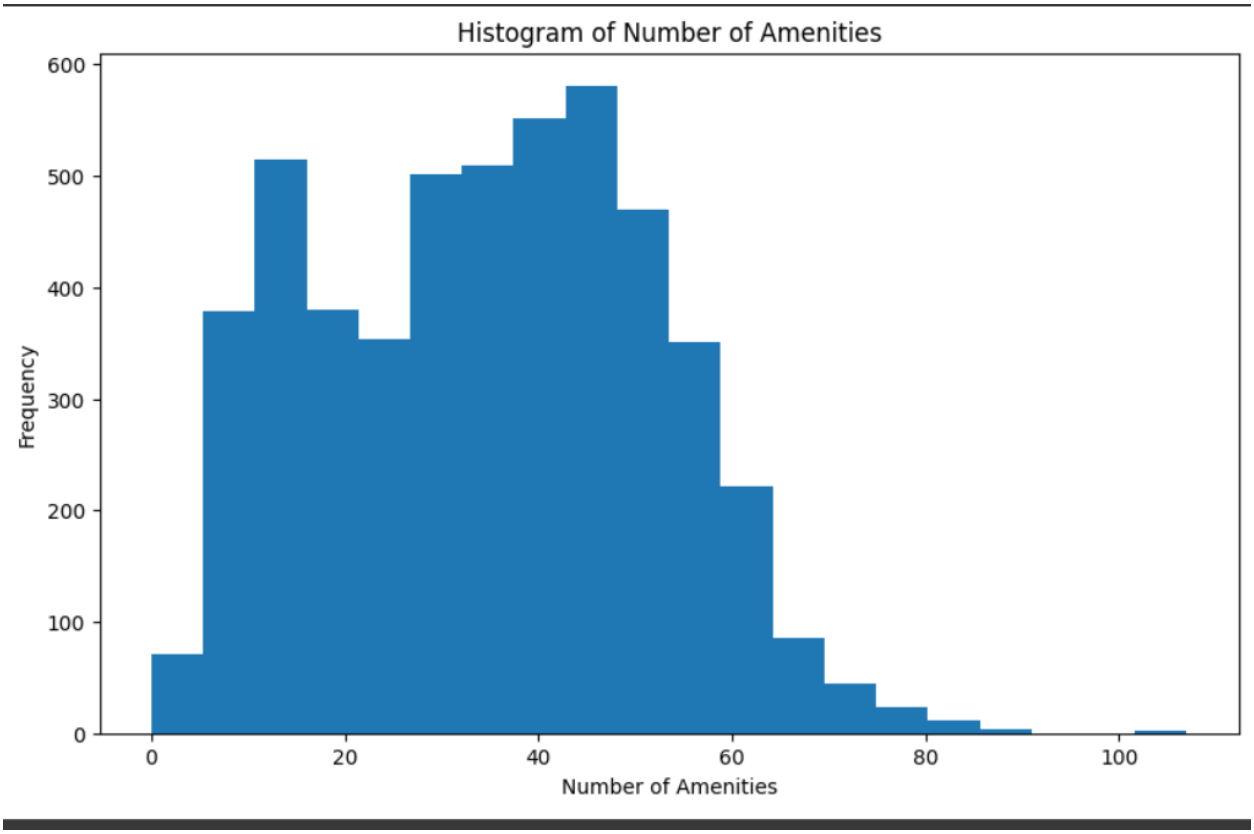
References

- https://www.qualtrics.com/en-au/experience-management/research/sampling-methods/
- https://wp.stolaf.edu/iea/sample-size/#:~:text=For%20populations%20under%201%2C000%2C%20a,ensure%20representativeness%20of%20the%20sample.
- https://www.leanwisdom.com/blog/key-data-visualization-techniques/
- https://synapse.koreamed.org/articles/1156715
- Aguinis, H., Gottfredson, R. K., & Joo, H. (2013). Best-practice recommendations for defining, identifying, and handling outliers. *Organizational research methods*, *16*(2), 270-301.

Appendices

- **Additional Figures or Tables:**

  (A histogram of number of amenities)



- **Code Snippets**: For stratified sampling

```
import pandas as pd

# Dictionary with sampling rates for each city
sampling_rates = {
    'Albany': 1.0,              # 100%
    'Ashville': 0.10,           # 10%
    'Boston': 0.10,             # 10%
    'BrowardCounty': 0.05,      # 5%
    'Cambridge': 0.10,          # 10%
    'JerseyCity': 0.10,         # 10%
    'Newark': 0.10,             # 10%
    'NewYork': 0.05,            # 5%
    'RhodeIsland': 0.10,        # 10%
    'Rochester': 1.0            # 100%
```

```
}

# List to store each city's sampled DataFrame
sampled_dfs = []

# Iterate over each city and apply sampling
for city, rate in sampling_rates.items():
    try:
        # Load the city's CSV file
        df = pd.read_csv(f'{city}.csv')

        # Add cityName column
        df['cityName'] = city

        # Apply sampling
        if rate < 1.0:
            sampled_df = df.sample(frac=rate, random_state=42)  # Use
random_state for reproducibility
        else:
            sampled_df = df  # Take all records if rate is 100%

        # Append the sampled DataFrame to the list
        sampled_dfs.append(sampled_df)
        print(f"Sampled data for {city} has {len(sampled_df)} records.")

    except FileNotFoundError:
        print(f"File {city}.csv not found. Skipping...")

# Concatenate all sampled DataFrames
combined_df = pd.concat(sampled_dfs, ignore_index=True)

# Save the combined data to a new CSV file
combined_df.to_csv('EastCoast.csv', index=False)
print(f"Combined sampled data saved with {len(combined_df)} total
records.")
```