**BOARD INFINITY PROJECT**

**Project: Analysis and Prediction of Airbnb Listing Prices**

**Project Done By:** Jithendranath Reddy Keshavareddy  
**Date:**10/07/2024

**Submitted to:** Board Infinity

**INTRODUCTION**

Welcome to my Airbnb listing price prediction project. In this project, I aim to analyse and predict the prices of Airbnb listings based on various features and attributes associated with each listing. By building a predictive model, we can gain insights into the factors that influence listing prices and create a tool that can estimate the price of a listing given its characteristics.

**PROJECT OVERVIEW**

The goal of this project is to perform an exploratory data analysis (EDA) and predictive modelling on Airbnb listing data using the R programming language. The project will encompass all stages of the data science lifecycle, from data import and cleaning to visualisation and modelling**.**

**DATA**Dataset link:  
<https://www.kaggle.com/datasets/dgomonov/new-york-city-airbnb-open-data/data>  
  
**Dataset Description:**

The dataset used in this project contains information about Airbnb listings. It includes various attributes that describe each listing such as the listings ID, name, Host name, Host Id, Neighbourhood group, Neighbourhood, Latitude, Longitude, Room type, Price, Minimum nights, Last review, Reviews per month, Calculated host listings count, availability 365.  
  
**Numeric Columns:**

* **availability 365:**

Number of days the listing is available for booking throughout the year.

* **calculated host listings count:**

Number of listings owned by the host.

* **reviews per month:**

Average number of reviews received per month.

* **price:**

Price of the listing per night.

* **minimum nights:**

Minimum number of nights required for a stay.

* **longitude:**

Longitude coordinate of the listing's location.

* **latitude:**

Latitude coordinate of the listing's location.

**Categorical Columns:**

* **room type:**

Type of room available (e.g., entire home, private room, shared room).

* **neighbourhood:**

Neighbourhood where the listing is located.

* **neighbourhood group:**

grouping of neighbourhoods within a city or region

* **name:**

Name of the listing.

* **host name:**

Name of the host.

**Date Columns:**

* **last review:**

Date of the last review.

**Identifiers:**

* **id:**

Unique identifier of the listing.

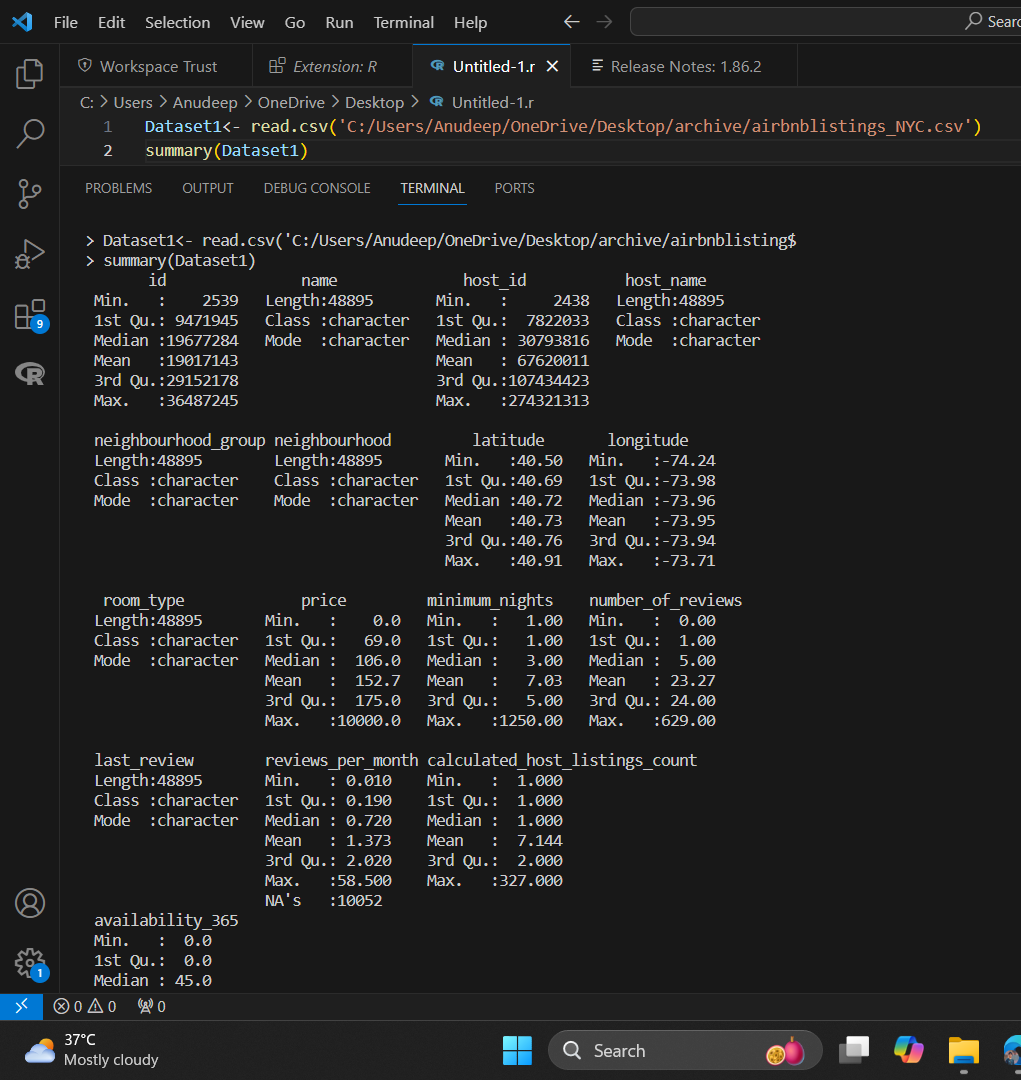
* **host id:**

Unique identifier of the host.

**Data Importing:**

Dataset1<- read.csv('C:/Users/Anudeep/OneDrive/Desktop/archive/airbnblistings\_NYC.csv')

summary(Dataset1)



**Data Cleaning and Transformation:**#load necessary packages  
library(dplyr)

# Data Cleaning: Remove 'neighbourhood\_group' column and rows with NA values

Dataset1 <- Dataset1 %>%

  select(-neighbourhood\_group) %>%

  na.omit()

# Calculate the interquartile range (IQR) for 'price'

Q1 <- quantile(Dataset1$price, 0.25, na.rm = TRUE)

Q3 <- quantile(Dataset1$price, 0.75, na.rm = TRUE)

IQR <- Q3 - Q1

# Calculate the lower and upper whiskers

lower\_whisker <- Q1 - 1.5 \* IQR

upper\_whisker <- Q3 + 1.5 \* IQR

# Impute outliers in 'price' with the median price

median\_price <- median(Dataset1$price, na.rm = TRUE)

Dataset1 <- Dataset1 %>%

  mutate(price = ifelse(price < lower\_whisker | price > upper\_whisker, median\_price, price))

# View the cleaned dataset summary

summary(Dataset1)

**Explanation:**

1. **Remove 'neighbourhood\_group' column and rows with NA values:**
   * The **select(-neighbourhood\_group)** removes the **neighbourhood\_group** column.
   * The **na.omit()** removes rows with any **NA** values.
2. **Calculate the IQR for price:**
   * **Q1** and **Q3** are the 25th and 75th percentiles of the **price** column, respectively.
   * **IQR** is the difference between **Q3** and **Q1**.
3. **Calculate the lower and upper whiskers:**
   * **lower\_whisker** is **Q1 - 1.5 \* IQR**.
   * **upper\_whisker** is **Q3 + 1.5 \* IQR**.
4. **Impute outliers with the median price:**
   * Outliers (values below **lower\_whisker** or above **upper\_whisker**) are replaced with the median price.
5. **View the cleaned dataset summary:**
   * The **summary(Dataset1)** command provides a summary of the cleaned dataset.

**Exploratory Data Analysis:**#load necessary packages

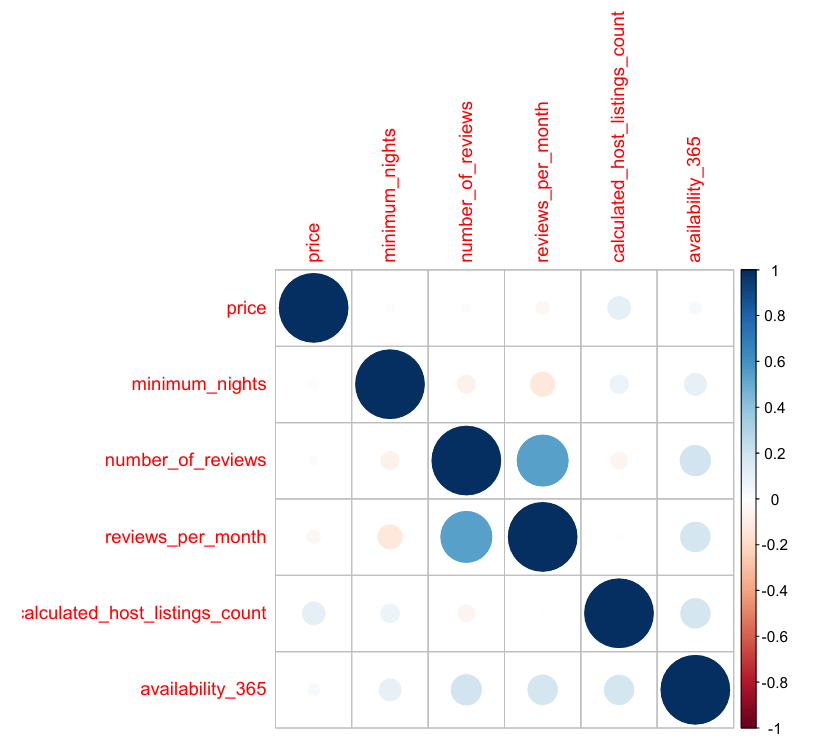
library(corrplot)

# Calculate correlations for numerical variables

correlations <- cor(Dataset1 %>% select(price, minimum\_nights, number\_of\_reviews, reviews\_per\_month, calculated\_host\_listings\_count, availability\_365), use = "complete.obs")

# Plot correlations

corrplot(correlations, method = "circle")

****

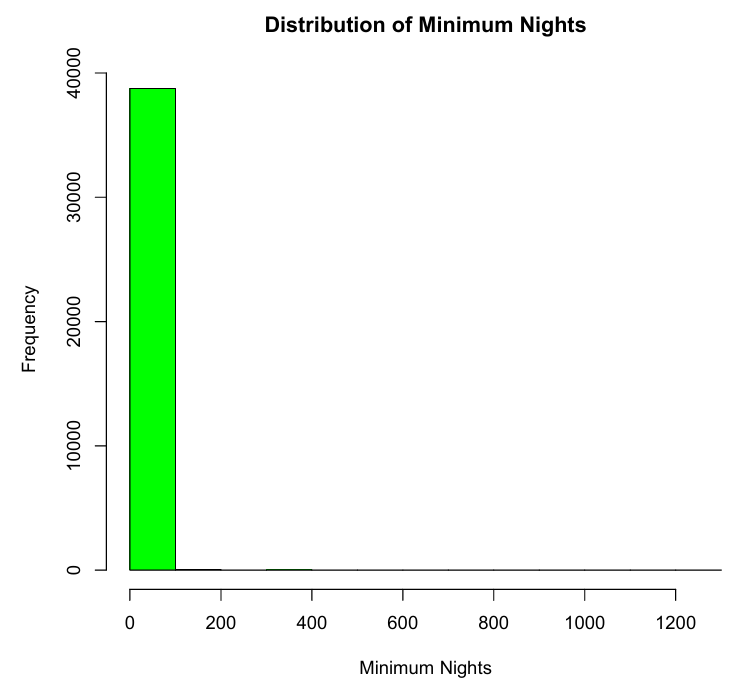
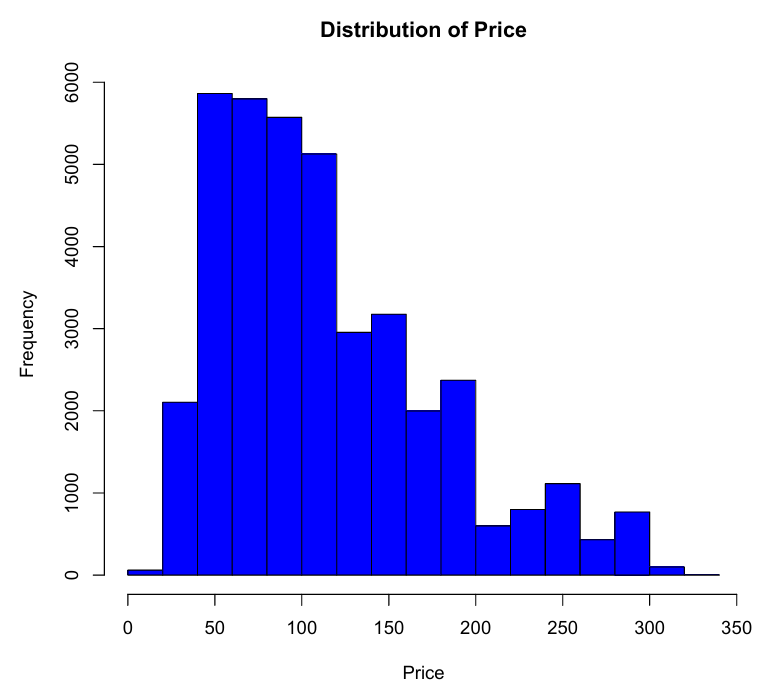
# Histograms for numerical variables

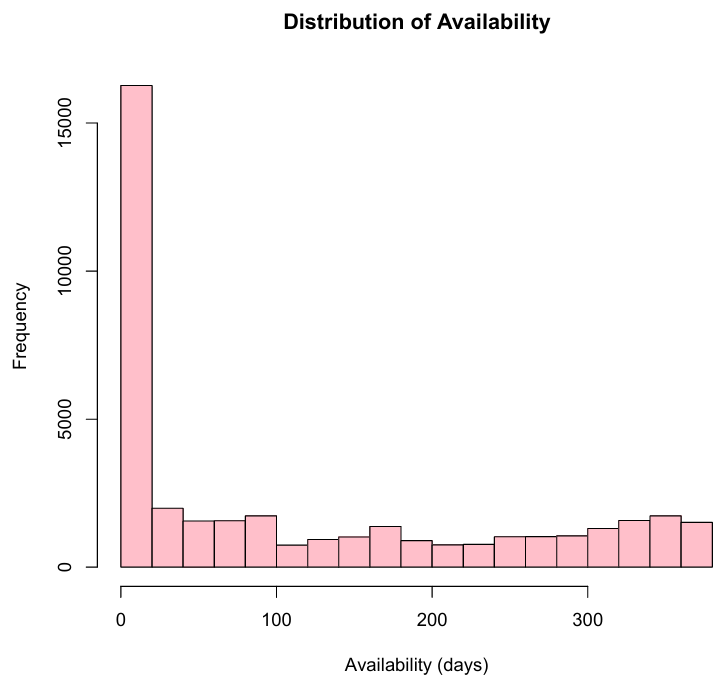
par(mfrow = c(2, 3))  # Set up a 2x3 grid for plotting

hist(Dataset1$price, main = "Distribution of Price", xlab = "Price", col = "blue")

hist(Dataset1$minimum\_nights, main = "Distribution of Minimum Nights", xlab = "Minimum Nights", col = "green")

hist(Dataset1$availability\_365, main = "Distribution of Availability", xlab = "Availability (days)", col = "pink")

****

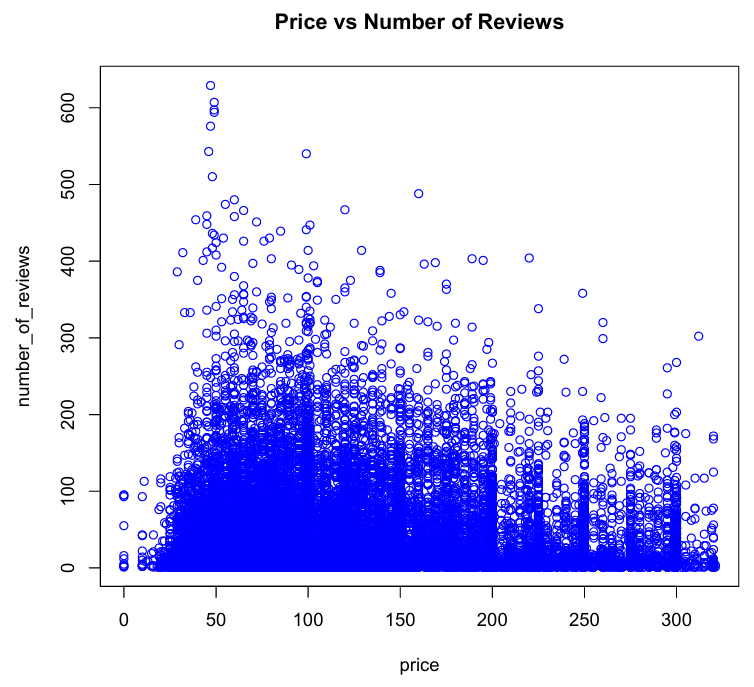


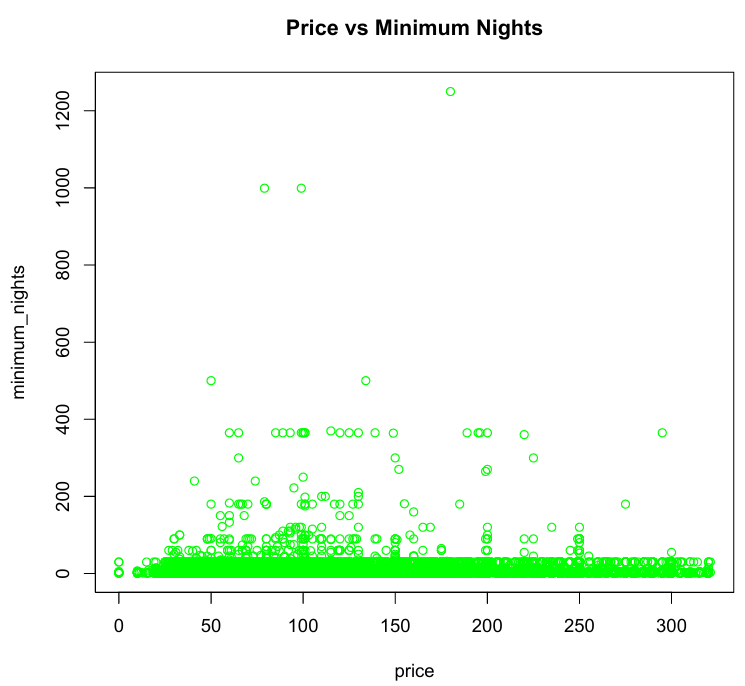
# Scatter plots to examine relationships between variables

par(mfrow = c(2, 3))  # Set up a 2x3 grid for plotting

plot(number\_of\_reviews ~ price, data = Dataset1, main = "Price vs Number of Reviews", col = "blue")

plot(minimum\_nights ~ price, data = Dataset1, main = "Price vs Minimum Nights", col = "green")

****

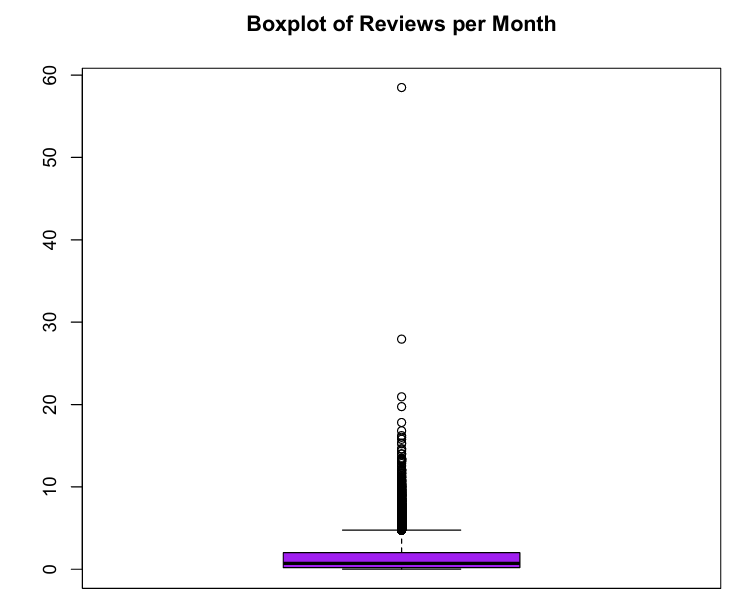
****

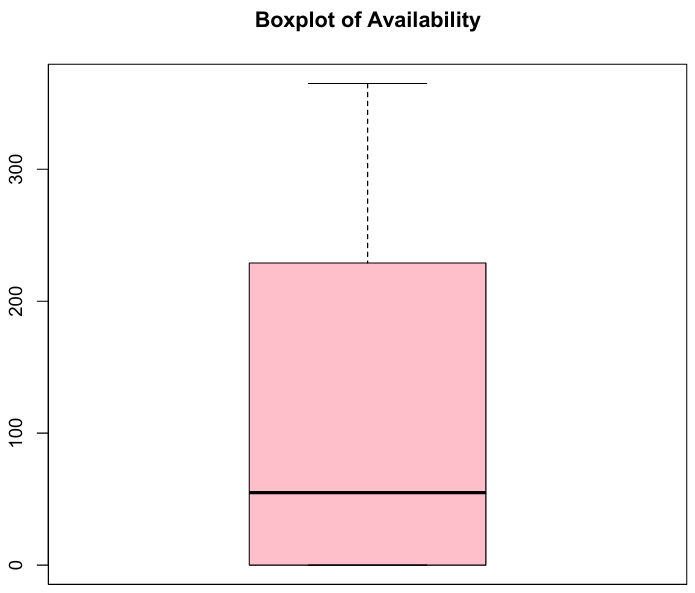
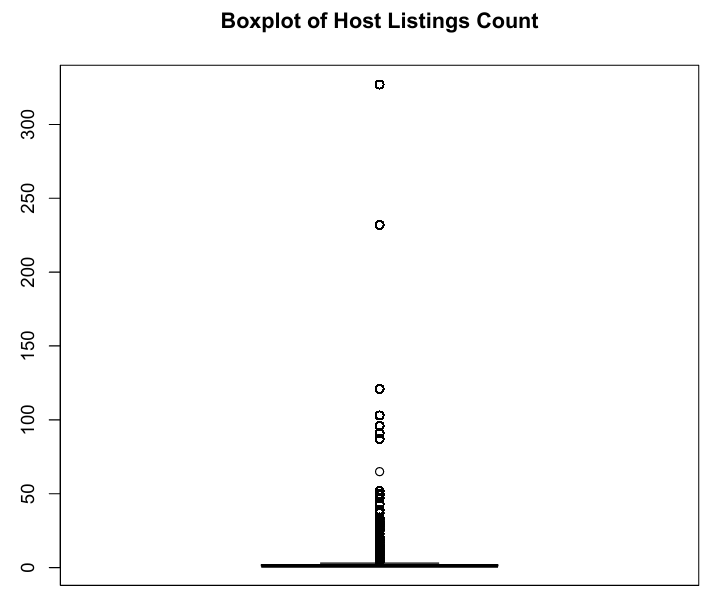
# Boxplots for numerical variables

par(mfrow = c(2, 3))  # Set up a 2x3 grid for plotting

boxplot(Dataset1$reviews\_per\_month, main = "Boxplot of Reviews per Month", col = "purple")

boxplot(Dataset1$calculated\_host\_listings\_count, main = "Boxplot of Host Listings Count", col = "orange")

boxplot(Dataset1$availability\_365, main = "Boxplot of Availability", col = "pink")



**Correlation Plot**

**Purpose:**  
To understand the relationships between different numerical variables by calculating and visualizing the correlation matrix. Correlation values range from -1 to 1, where:

* 1 means a perfect positive relationship,
* -1 means a perfect negative relationship,
* 0 means no relationship.

**Explanation:**

* cor(): Calculates the correlation matrix for the selected numerical variables.
* corrplot(): Visualizes the correlation matrix using circles to represent the strength and direction of the relationships.

**Histograms**

**Purpose:**

To visualize the distribution of individual numerical variables, giving insights into their central tendency, spread, and any potential outliers.

**Explanation:**

* par(mfrow = c(2, 3)): Sets up a grid of 2 rows and 3 columns for the histograms.
* hist(): Plots histograms for each selected numerical variable, with appropriate titles, axis labels, and colors.

**Scatter Plots**

**Purpose:**

To examine the relationships between the price variable and other numerical variables. These plots help in identifying patterns, trends, and potential correlations

**Explanation:**

* plot(): Creates scatter plots to show how price varies with number\_of\_reviews, minimum\_nights, availability\_365, reviews\_per\_month, and calculated\_host\_listings\_count. Each plot includes appropriate titles and colors.

**Boxplots**

**Purpose:**

To visualize the distribution of numerical variables and identify the presence of outliers. Boxplots provide a summary of the minimum, first quartile, median, third quartile, and maximum of a dataset.

**Explanation:**

par(mfrow = c(2, 3)): Sets up a grid of 2 rows and 3 columns for the boxplots.

boxplot(): Plots boxplots for each selected numerical variable, with appropriate titles and colors to help in identifying the spread and outliers in the data.

**Feature Engineering:**

#install or load necessary packages

install.packages("geosphere")

library(geosphere)

library(dplyr)

# Coordinates of Times Square, NYC

times\_square\_lat <- 40.7580

times\_square\_lon <- -73.9855

# Perform feature engineering

Dataset1 <- Dataset1 %>%

  mutate(

    # Calculate distance from each listing to Times Square

    distance\_to\_times\_square = distHaversine(

      matrix(c(longitude, latitude), ncol = 2),

      matrix(c(times\_square\_lon, times\_square\_lat), ncol = 2)

    ) / 1000,  # Distance in kilometers

    # Convert 'last\_review' to Date type

    last\_review = as.Date(last\_review),

    # Create a feature representing the number of days since the last review

    days\_since\_last\_review = as.numeric(Sys.Date() - last\_review),

    # Create a feature representing the average number of reviews per month

    review\_rate\_per\_month = ifelse(number\_of\_reviews > 0, number\_of\_reviews / days\_since\_last\_review \* 30, 0),

    # Encode 'room\_type' as a numerical variable

    room\_type\_encoded = as.numeric(factor(room\_type))

  )

# Inspect the new features

head(Dataset1 %>% select(distance\_to\_times\_square, days\_since\_last\_review, review\_rate\_per\_month, room\_type\_encoded))

# Print the structure of the updated dataset to verify new features

str(Dataset1)

**Explanation:**

* Distance from Times Square: Uses the geosphere package to calculate the Haversine distance between each listing's latitude and longitude coordinates and Times Square's coordinates.
* Days since the last review: Converts last\_review to Date format and calculates the difference in days from the current date (Sys.Date()).
* Review rate per month: Calculates the average number of reviews per month since the earliest review date.
* Room type encoding: Converts the categorical room\_type variable into numerical factors using factor() and assigns numerical codes.

**Modelling:**

# Install and load necessary packages

install.packages("caTools")

install.packages("randomForest")

library(caTools)

library(randomForest)

# Set seed for reproducibility

set.seed(123)

# Split the data into training (70%) and testing (30%) sets

split <- sample.split(Dataset1$price, SplitRatio = 0.7)

training\_set <- subset(Dataset1, split == TRUE)

testing\_set <- subset(Dataset1, split == FALSE)

# Build Linear Regression model

linear\_model <- lm(price ~ distance\_to\_times\_square + days\_since\_last\_review + review\_rate\_per\_month + room\_type\_encoded, data = training\_set)

# Build Random Forest model

random\_forest\_model <- randomForest(price ~ distance\_to\_times\_square + days\_since\_last\_review + review\_rate\_per\_month + room\_type\_encoded, data = training\_set, ntree = 100)

# Predict on the test set

linear\_predictions <- predict(linear\_model, newdata = testing\_set)

random\_forest\_predictions <- predict(random\_forest\_model, newdata = testing\_set)

# Calculate RMSE for both models

linear\_rmse <- sqrt(mean((linear\_predictions - testing\_set$price)^2))

random\_forest\_rmse <- sqrt(mean((random\_forest\_predictions - testing\_set$price)^2))

# Print RMSE values

cat("Linear Regression RMSE:", linear\_rmse, "\n")

cat("Random Forest RMSE:", random\_forest\_rmse, "\n")

# Visualize Linear Regression Performance

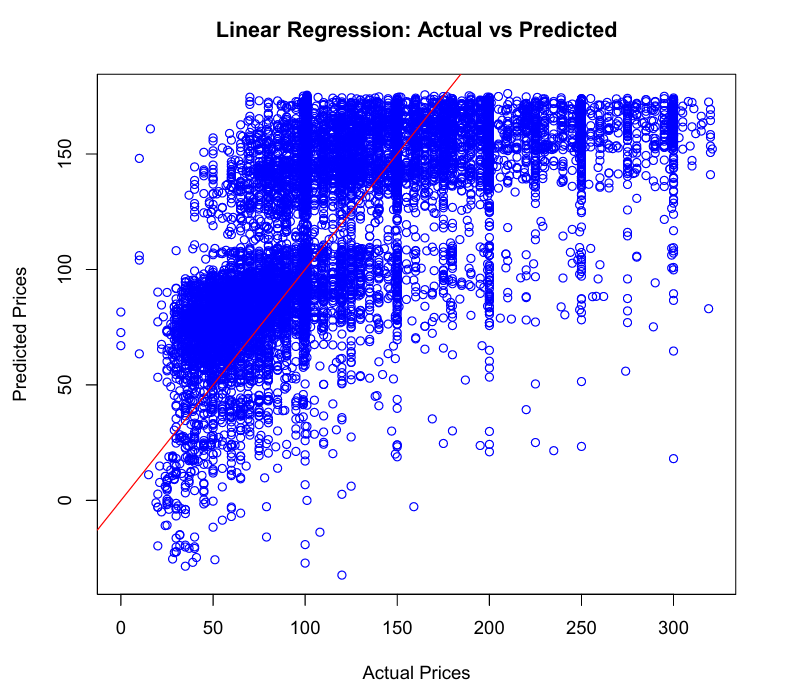
plot(testing\_set$price, linear\_predictions, main = "Linear Regression: Actual vs Predicted", xlab = "Actual Prices", ylab = "Predicted Prices", col = "blue")

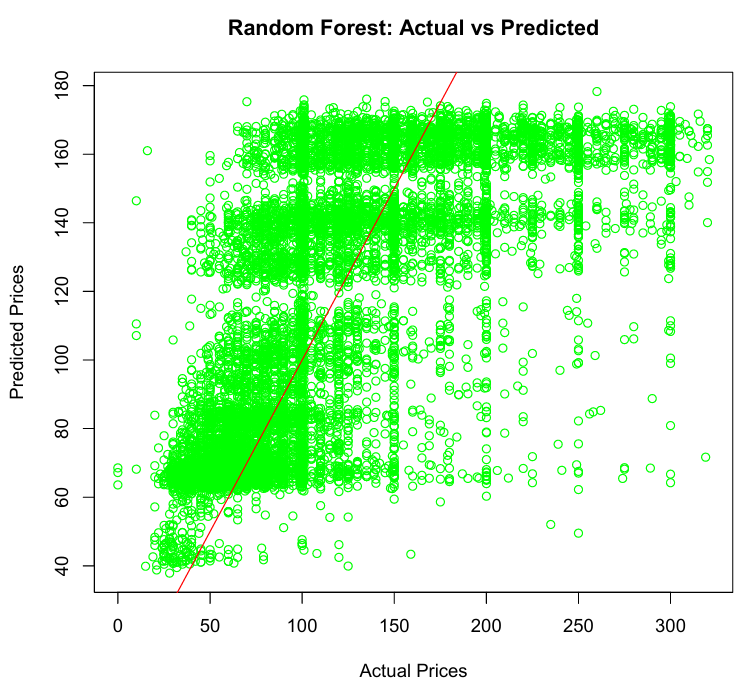
abline(0, 1, col = "red")  # y = x line for reference

# Visualize Random Forest Performance

plot(testing\_set$price, random\_forest\_predictions, main = "Random Forest: Actual vs Predicted", xlab = "Actual Prices", ylab = "Predicted Prices", col = "green")

abline(0, 1, col = "red")  # y = x line for reference

****

****

**Explanation:**

* **Data Splitting:** The dataset (**Dataset1**) is split into training (**training\_set**) and testing (**testing\_set**) sets using a 70%-30% ratio.
* **Model Building:**
  + **Linear Regression (linear\_model):** Uses the **lm()** function to build a linear regression model predicting **price** based on the engineered features (**distance\_to\_times\_square**, **days\_since\_last\_review**, **review\_rate\_per\_month**, **room\_type\_encoded**).
  + **Random Forest (random\_forest\_model):** Uses the **randomForest()** function to build a random forest model predicting **price** using the same features, with **ntree = 100** specifying the number of trees in the forest.
* **Model Prediction and Evaluation:**
  + Predictions are made on the **testing\_set** using both models.
  + Root Mean Squared Error (RMSE) is calculated to evaluate the prediction accuracy of each model.
* **Visualization:**
  + Scatter plots (**plot()**) are used to visualize how well the predicted prices (**linear\_predictions** and **random\_forest\_predictions**) align with the actual prices (**testing\_set$price**).
  + The red line (**abline(0, 1, col = "red")**) indicates perfect prediction (where actual = predicted).

**THE END**