Analysis on Airbnb Dataset

Group2

11/13/2024

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```r
#Install Package
install.packages("readxl")
install.packages("tidyverse")
install.packages("leaflet")
install.packages("corrplot")

#Load libraries
library(readxl)
library(tidyverse)  # This includes dplyr, ggplot2, and tidyr
library(tidyr)      # Explicitly load tidyr for pivot_longer
library(leaflet)    # Load leaflet for mapping
library(ggcorrplot)   # Load corrplot for correlation visualization
```

Load data into datframe

```r
# Read the Excel file
airbnb <- readxl::read_xlsx("AirbnbLA_2023.xlsx")

# View the data
head(airbnb) #use head for a sample

## # A tibble: 6 × 32
##      Id `Host Id` `Host Name`    `Host Is Superhost` `Host Acceptance
Rate`
##   <dbl>     <dbl> <chr>          <lgl>               <chr>
## 1   109       521 Paolo          FALSE               50%
## 2  2708      3008 Chas.          TRUE                100%
## 3  2732      3041 Yoga Priestess FALSE               42%
## 4 63416    309512 Vincenzo       TRUE                96%
## 5 67089    210344 Brenna         TRUE                95%
## 6  5728      9171 Sanni          FALSE               79%
## # ℹ 27 more variables: `Host Response Rate` <chr>, `Host Response Time`
<chr>,
```

```
## #   `Host Since` <dttm>, `Neighbourhood Group` <chr>, Neighbourhood <chr>,
## #   Latitude <dbl>, Longitude <dbl>, `Room Type` <chr>, Accommodates
<dbl>,
## #   Beds <dbl>, Price <dbl>, `Instant Bookable` <lgl>, `First Review`
<dttm>,
## #   `Last Review` <dttm>, License <chr>, `Reviews Per Month` <dbl>,
## #   `Minimum Nights` <dbl>, `Number Of Reviews` <dbl>,
## #   `Number Of Reviews L30D` <dbl>, `Number Of Reviews Ltm` <dbl>, …
```

Perform Data Cleaning

```
# Initial data cleaning and renaming columns
airbnb_cleaned <- airbnb %>%
  rename('id' = 'Id',
         'host_id' = 'Host Id',
         'host_name' = 'Host Name',
         'host_is_superhost' = 'Host Is Superhost',
         'host_acceptance_rate' = 'Host Acceptance Rate',
         'host_response_rate' = 'Host Response Rate',
         'host_response_time' = 'Host Response Time',
         'host_since' = 'Host Since',
         'neighbourhood_group' = 'Neighbourhood Group',
         'neighbourhood' = 'Neighbourhood',
         'latitude' = 'Latitude',
         'longitude' = 'Longitude',
         'room_type' = 'Room Type',
         'accommodates' = 'Accommodates',
         'beds' = 'Beds',
         'price' = 'Price',
         'instant_bookable' = 'Instant Bookable',
         'first_review' = 'First Review',
         'last_review' = 'Last Review',
         'license' = 'License',
         'reviews_per_month' = 'Reviews Per Month',
         'minimum_nights' = 'Minimum Nights',
         'number_of_reviews' = 'Number Of Reviews',
         'number_of_reviews_l30d' = 'Number Of Reviews L30D',
         'number_of_reviews_ltm' = 'Number Of Reviews Ltm',
         'review_scores_rating' = 'Review Scores Rating',
         'review_scores_accuracy' = 'Review Scores Accuracy',
         'review_scores_checkin' = 'Review Scores Checkin',
         'review_scores_cleanliness' = 'Review Scores Cleanliness',
         'review_scores_communication' = 'Review Scores Communication',
         'review_scores_location' = 'Review Scores Location',
         'review_scores_value' = 'Review Scores Value'
  )

#drop licence column
airbnb_cleaned <- select(airbnb_cleaned, -license)

# Convert "N/A" values to NA
```

```r
airbnb_cleaned <- airbnb_cleaned %>%
  mutate(
    host_acceptance_rate = if_else(host_acceptance_rate == "N/A", NA,
host_acceptance_rate),
    host_response_rate = if_else(host_response_rate == "N/A", NA,
host_response_rate),
    host_response_time = if_else(host_response_time == "N/A", NA,
host_response_time)
  )

# Impute missing review scores with the mean value of each column
airbnb_cleaned <- airbnb_cleaned %>%
  mutate(
    review_scores_accuracy = if_else(is.na(review_scores_accuracy),
mean(review_scores_accuracy, na.rm = TRUE), review_scores_accuracy),
    review_scores_checkin = if_else(is.na(review_scores_checkin),
mean(review_scores_checkin, na.rm = TRUE), review_scores_checkin),
    review_scores_cleanliness = if_else(is.na(review_scores_cleanliness),
mean(review_scores_cleanliness, na.rm = TRUE), review_scores_cleanliness),
    review_scores_communication = if_else(is.na(review_scores_communication),
mean(review_scores_communication, na.rm = TRUE),
review_scores_communication),
    review_scores_location = if_else(is.na(review_scores_location),
mean(review_scores_location, na.rm = TRUE), review_scores_location),
    review_scores_value = if_else(is.na(review_scores_value),
mean(review_scores_value, na.rm = TRUE), review_scores_value)
  )

# Check for missing values again
colSums(is.na(airbnb_cleaned))
```

```
##                          id                     host_id
##                           0                           0
##                   host_name           host_is_superhost
##                           0                           0
##        host_acceptance_rate          host_response_rate
##                        4903                        6076
##          host_response_time                  host_since
##                        6076                           0
##          neighbourhood_group               neighbourhood
##                           0                           0
##                    latitude                   longitude
##                           0                           0
##                   room_type                accommodates
##                           0                           0
##                        beds                       price
##                           0                           0
##            instant_bookable                first_review
##                           0                           0
##                 last_review           reviews_per_month
```

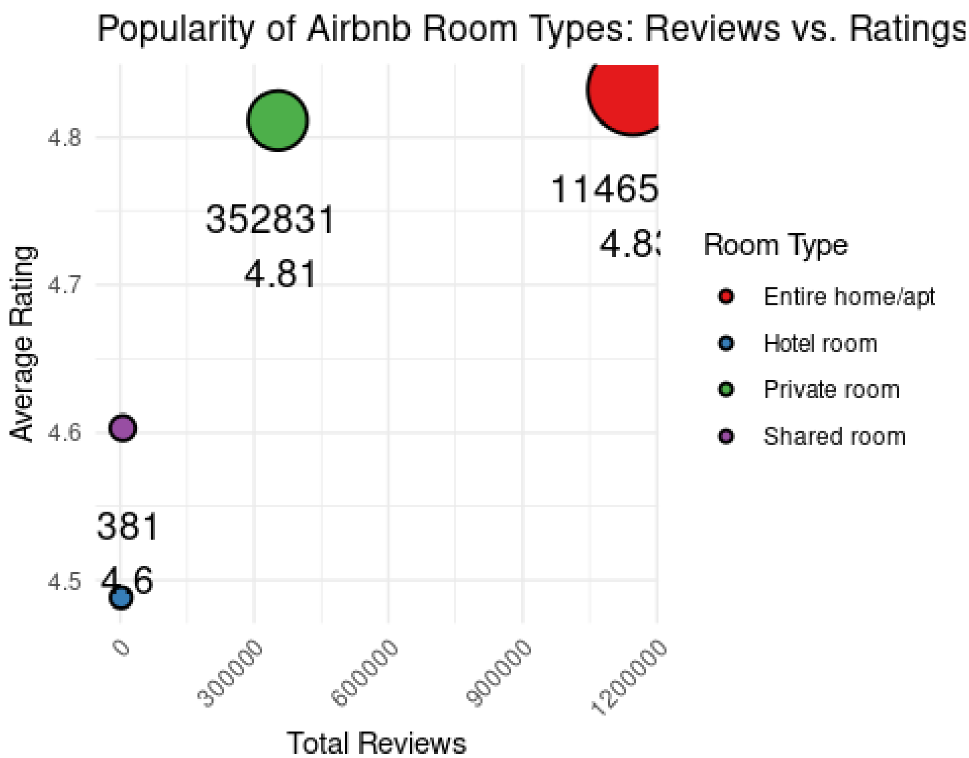
```
##                            0                             0
##               minimum_nights             number_of_reviews
##                            0                             0
##        number_of_reviews_l30d        number_of_reviews_ltm
##                            0                             0
##          review_scores_rating       review_scores_accuracy
##                            0                             0
##         review_scores_checkin    review_scores_cleanliness
##                            0                             0
## review_scores_communication       review_scores_location
##                            0                             0
##           review_scores_value
##                            0
```

```
sum(is.na(airbnb_cleaned))
```

```
## [1] 17055
```

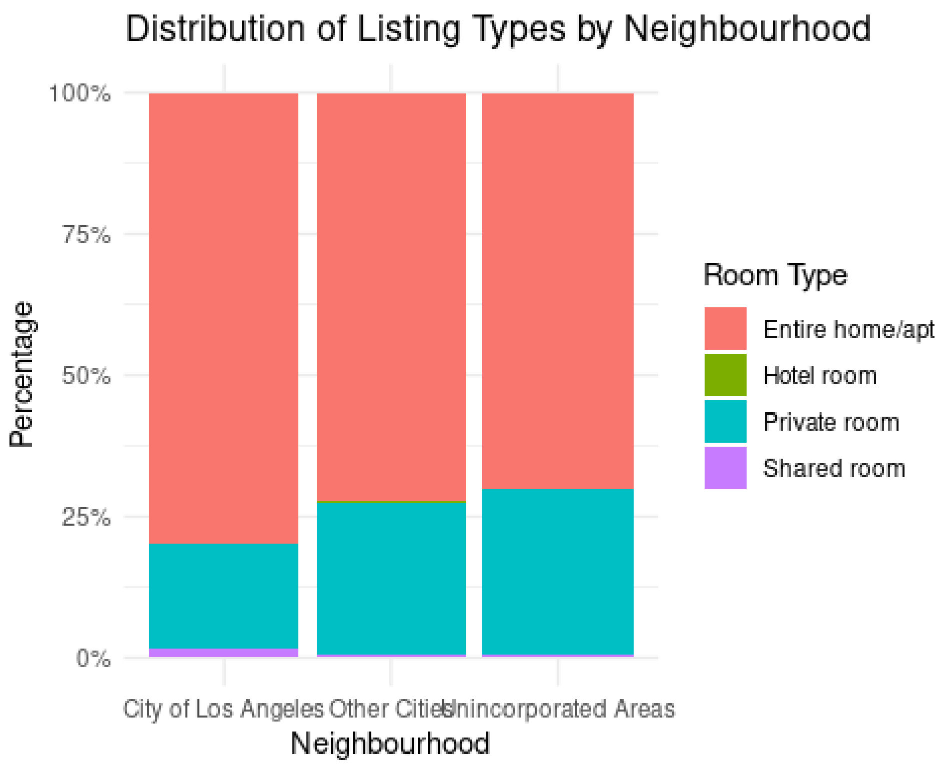
Question 1: Which type of Airbnb properties garner the most reviews, indicating popularity?

```
#Summarise the total review, avg rating, no. of reviews across room type

airbnb_popularity <- airbnb_cleaned %>%
  group_by(room_type) %>%
  summarise(total_reviews = sum(number_of_reviews),
            avg_rating = sum(review_scores_rating * number_of_reviews) /
sum(number_of_reviews))

# Bubble chart for total reviews and avg_rating

ggplot(airbnb_popularity, aes(x = total_reviews, y = avg_rating, size =
total_reviews, fill = room_type)) +
  geom_point(shape = 21, color = "black", stroke = 1) +
  geom_text(aes(label = paste(total_reviews, "\n", round(avg_rating, 2))),
            vjust = 2, color = "black", size = 5) +
  scale_size(range = c(3, 15), guide = "none") +
  scale_fill_brewer(palette = "Set1") +
  labs(title = "Popularity of Airbnb Room Types: Reviews vs. Ratings",
       x = "Total Reviews",
       y = "Average Rating",
       fill = "Room Type") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Question 2: How does the distribution of listing types vary across different neighborhoods or regions?

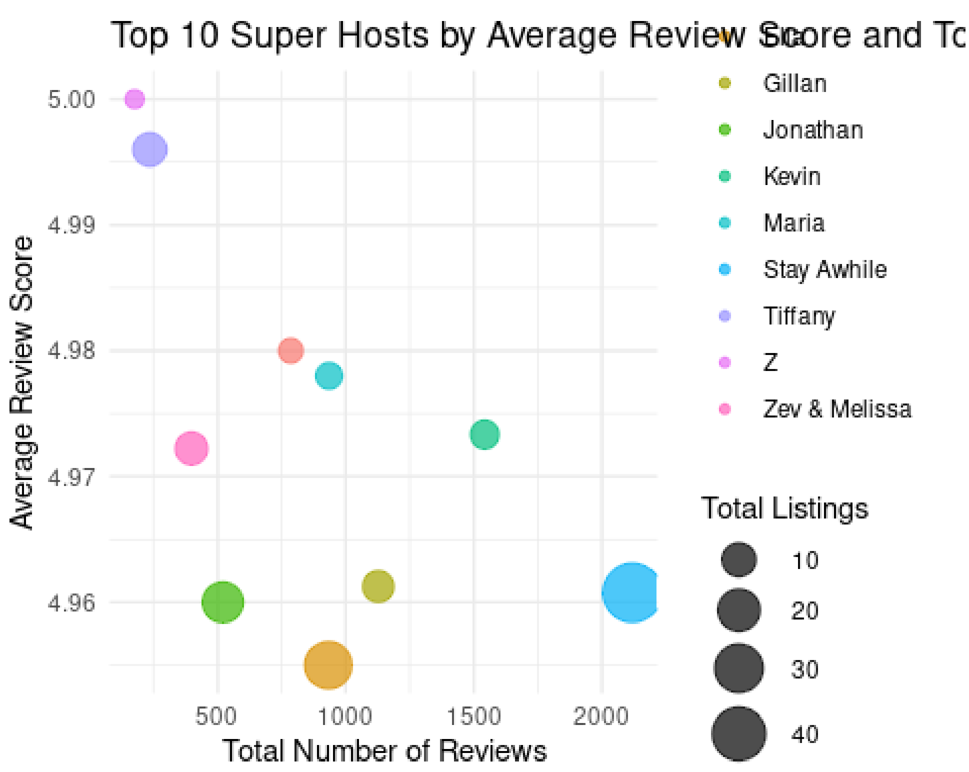
```
airbnb_summary <- airbnb_cleaned %>%
  group_by(neighbourhood_group, room_type) %>%
  summarise(count = n(), .groups='drop') %>%
  mutate(percentage = count / sum(count) * 100)

# Plot the 100% stacked bar chart for distribution of listing types vary
across different neighborhoods or regions

ggplot(airbnb_summary, aes(x = neighbourhood_group, y = percentage, fill =
room_type)) +
  geom_bar(stat = "identity", position = "fill") +
  scale_y_continuous(labels = scales::percent) +
  labs(x = "Neighbourhood", y = "Percentage", fill = "Room Type",
       title = "Distribution of Listing Types by Neighbourhood") +
  theme_minimal()
```



Question 3: Who are the top 10 Super hosts based on listings, review scores, and number of reviews? How do their listing and review score distributions vary?

```
#Assign the rank to each host based on review score rating, number of reviews
and total listings
airbnb_groupby_unique_host <- airbnb_cleaned %>%
  filter(host_is_superhost = TRUE) %>%
  group_by(host_id, host_name) %>%
  summarise(avg_review_score = mean(review_scores_rating),
            total_reviews = sum(number_of_reviews),
            total_listing = n()) %>%
  ungroup() %>%  # Corrected to 'ungroup()'
  mutate(
    rank_review_score = rank(-avg_review_score),  # Rank by average review
score rating
    rank_number_of_reviews = rank(-total_reviews), # Rank by number of
reviews
    rank_number_of_listings = rank(-total_listing) # Rank by maximum number
of listings
  )


# Combine the ranks (e.g., by summing them up for an overall rank)
filtered_data <- airbnb_groupby_unique_host %>%
  mutate(overall_rank = rank_review_score + rank_number_of_reviews +
```

```
rank_number_of_listings) %>%
  arrange(overall_rank) # Sort by the combined rank

# Select the top 10 super hosts
top_10_hosts <- filtered_data %>%
  slice_head(n = 10) %>%
  select(host_id, host_name, avg_review_score, total_reviews, total_listing,
overall_rank)


#Visualize top 10 Super hosts based on listings, review scores, and number of
reviews
ggplot(top_10_hosts, aes(x = total_reviews, y = avg_review_score, color =
host_name, size=total_listing)) +
  geom_point(alpha = 0.7) +
  scale_size_continuous(range = c(3, 10)) +
  labs(title = "Top 10 Super Hosts by Average Review Score and Total
Reviews",
       x = "Total Number of Reviews",
       y = "Average Review Score",
       size = "Total Listings",
       color = "Super Host Name") +
  theme_minimal()
```

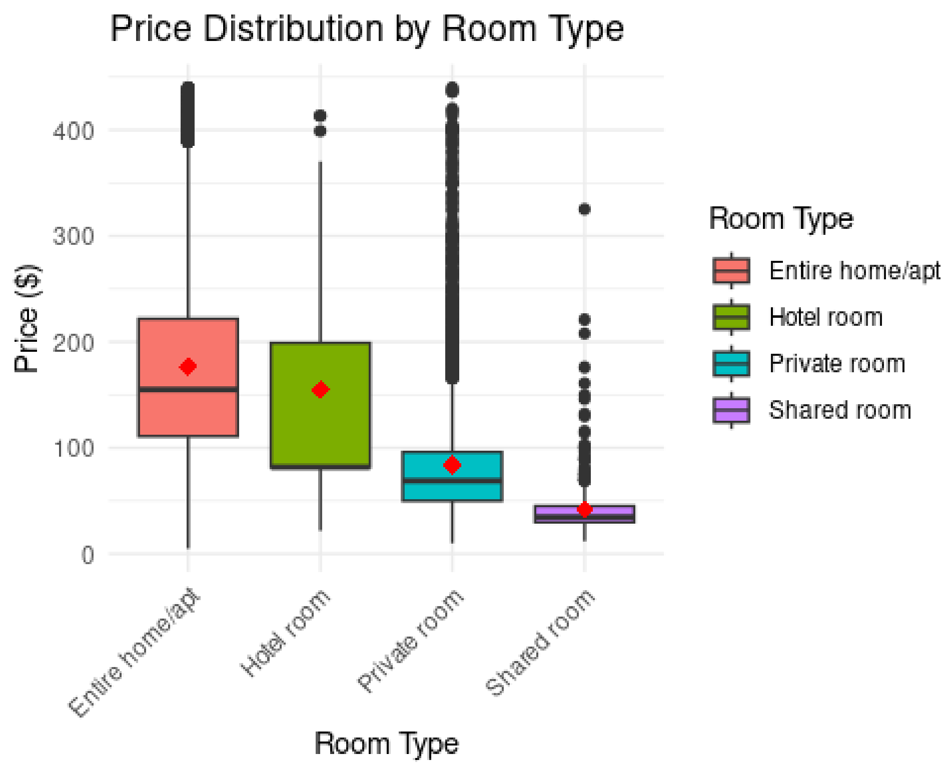


Question 4: What is the overall price trend for different room types on Airbnb?

```
# Calculate summary statistics for room types and see for price outliers

summary_stats_by_roomType <- airbnb_cleaned %>%
  group_by(room_type) %>%
  summarise(
    Average_Price = mean(price, na.rm = TRUE),
    Median_Price = median(price, na.rm = TRUE),
    Min_Price = min(price, na.rm = TRUE),
    Max_Price = max(price, na.rm = TRUE),
    SD_Price = sd(price, na.rm = TRUE)
  )

print(summary_stats_by_roomType)

## # A tibble: 4 × 6
##   room_type       Average_Price Median_Price Min_Price Max_Price SD_Price
##   <chr>                   <dbl>        <dbl>     <dbl>     <dbl>    <dbl>
## 1 Entire home/apt         268.          170          5     99999    777.
## 2 Hotel room              798.          100.        22      9999   2439.
## 3 Private room            118.           69         10     99999   1204.
## 4 Shared room              53.7          35         12      1200     95.3

# Data Cleaning: Remove outliers in price using the IQR method
remove_price_outliers <- function(data) {
  Q1 <- quantile(data$price, 0.25, na.rm = TRUE)
  Q3 <- quantile(data$price, 0.75, na.rm = TRUE)
  IQR_value <- Q3 - Q1
  lower_bound <- Q1 - 1.5 * IQR_value
  upper_bound <- Q3 + 1.5 * IQR_value
  data %>% filter(price >= lower_bound & price <= upper_bound)
}

airbnb_filtered <- remove_price_outliers(airbnb_cleaned)
```

Plot Analysis Question4

```
# Box plot of Price Distribution by Room Type
ggplot(airbnb_filtered, aes(x = room_type, y = price, fill = room_type)) +
  geom_boxplot() +
  stat_summary(fun = "mean", geom = "point", shape = 18, size = 3, color =
"red", fill = "red",
               position = position_dodge(width = 0.75)) +  # Adjusts the
position of the mean marker
  labs(title = "Price Distribution by Room Type",
       x = "Room Type",
       y = "Price ($)",
       fill = "Room Type") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



Question 5: How does the average price of Airbnb listings vary across different neighborhoods in Los Angeles?

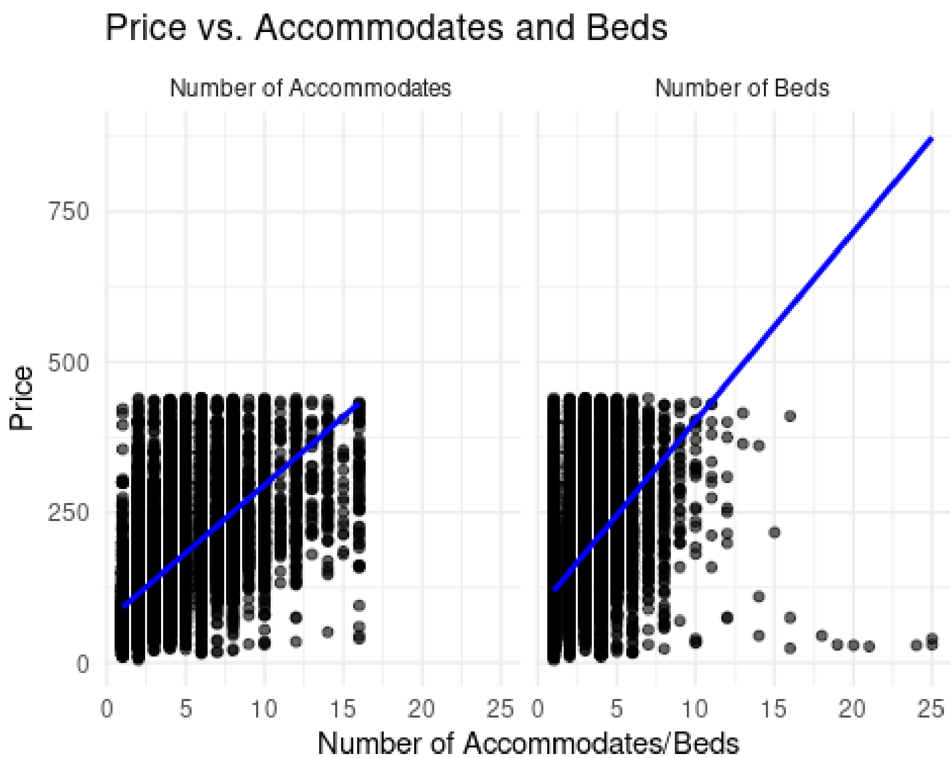
```
# Group the data by neighborhood and calculate average price
df_grouped <- airbnb_filtered %>%
  group_by(neighbourhood) %>%
  summarise(
    Avg_Price = mean(price, na.rm = TRUE),
    latitude = first(latitude),
    longitude = first(longitude),
    .groups = 'drop'
  )

print(df_grouped)

## # A tibble: 265 × 4
##    neighbourhood   Avg_Price latitude longitude
##    <chr>               <dbl>    <dbl>     <dbl>
##  1 Acton               171.      34.5     -118.
##  2 Adams-Normandie      90.5     34.0     -118.
##  3 Agoura Hills        174.      34.2     -119.
##  4 Agua Dulce          158.      34.5     -118.
##  5 Alhambra            128.      34.1     -118.
##  6 Alondra Park        178.      33.9     -118.
##  7 Altadena            155.      34.2     -118.
##  8 Angeles Crest       168.      34.4     -118.
##  9 Arcadia             123.      34.1     -118.
```

```
## 10 Arleta                100        34.2      -118.
## # i 255 more rows

# Create a color palette based on average prices
pal <- colorNumeric(palette = "viridis", domain = df_grouped$Avg_Price)

# Create the interactive map with color tones
leaflet(df_grouped) %>%
  addTiles() %>%
  addCircleMarkers(
    lng = ~longitude,
    lat = ~latitude,
    radius = ~Avg_Price / 50,
    popup = ~paste(neighbourhood, ": $", round(Avg_Price, 2)),
    color = ~pal(Avg_Price),
    fillOpacity = 0.7
  ) %>%
  setView(lng = mean(df_grouped$longitude), lat = mean(df_grouped$latitude),
zoom = 11) %>%
  addLegend("bottomright", pal = pal, values = ~Avg_Price,
            title = "Average Price",
            opacity = 0.7)
```

Data Modeling Visulaization

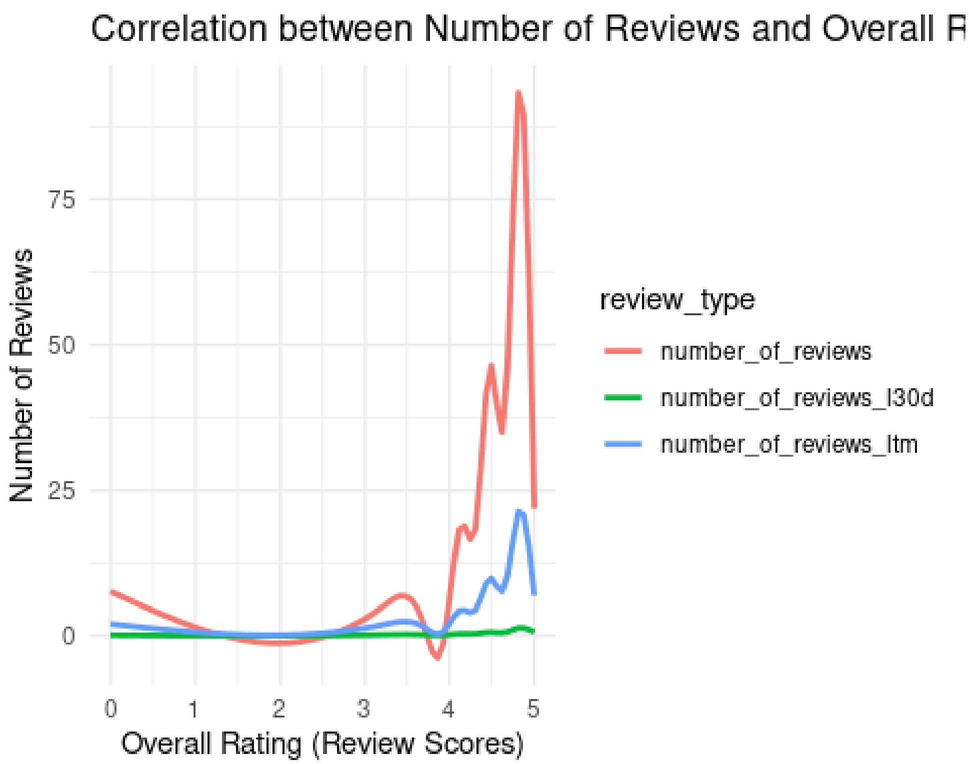
```
# Impact of Beds and Accommodates on Price of Room Types
# Checking which has more impact: Accommodates or Beds
correlation_matrix <- airbnb_filtered %>%
  select(price, accommodates, beds) %>%
  cor()
print(correlation_matrix)

##                  price accommodates      beds
## price        1.0000000    0.6022625 0.4964012
## accommodates 0.6022625    1.0000000 0.8219663
## beds         0.4964012    0.8219663 1.0000000

# Reshape the data for combined plotting
airbnb_long <- airbnb_filtered %>%
  pivot_longer(cols = c(beds, accommodates), names_to = "Type", values_to =
"Value")

# Create a single graph for Price vs. Beds and Price vs. Accommodates
ggplot(airbnb_long, aes(x = Value, y = price)) +
  geom_point(alpha = 0.6) +  # Adjust transparency for better visibility
  geom_smooth(method = "lm", se = FALSE, color = "blue") +  # Add linear
regression line
  labs(title = "Price vs. Accommodates and Beds",
       x = "Number of Accommodates/Beds",
       y = "Price") +
  facet_wrap(~ Type, labeller = as_labeller(c(beds = "Number of Beds",
```

```
accommodates = "Number of Accommodates"))) +
  theme_minimal()
```



Question 6: Is there a correlation between the number of reviews and overall ratings? Do hosts with more reviews tend to have better ratings?

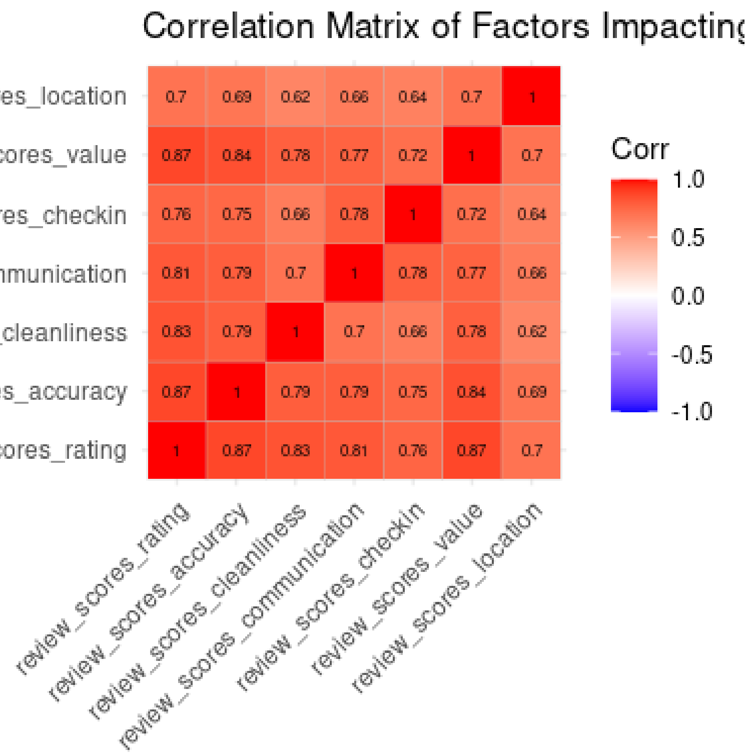
```
# Transform data for faceting
airbnb_long <- airbnb_cleaned %>%
  pivot_longer(cols = c(number_of_reviews, number_of_reviews_l30d,
number_of_reviews_ltm),
               names_to = "review_type",
               values_to = "review_count")

# Creating a scatter plot with a trend line
ggplot(airbnb_long, aes(x =review_scores_rating , y = review_count)) +
  geom_smooth(aes(color = review_type), se=FALSE) +
  labs(title = "Correlation between Number of Reviews and Overall Rating",
       x = "Overall Rating (Review Scores)",
       y = "Number of Reviews") +
  theme_minimal()
```



Question 7: Which factors, such as the check-in process, cleanliness, accuracy of listing descriptions, etc., most significantly impact review ratings?

```
cor_matrix <- cor(airbnb_cleaned[, c("review_scores_rating",
"review_scores_accuracy",
                                     "review_scores_cleanliness",
"review_scores_communication",
                                     "review_scores_checkin",
"review_scores_value",
                                     "review_scores_location")],
                  use = "complete.obs")

print("Correlation matrix of factors impacting review ratings:")

## [1] "Correlation matrix of factors impacting review ratings:"

print(cor_matrix)

##                             review_scores_rating review_scores_accuracy
## review_scores_rating                   1.0000000              0.8731444
## review_scores_accuracy                 0.8731444              1.0000000
## review_scores_cleanliness              0.8281561              0.7925776
## review_scores_communication            0.8103266              0.7936822
## review_scores_checkin                  0.7569332              0.7532662
## review_scores_value                    0.8691172              0.8431411
## review_scores_location                 0.6952753              0.6936656
##                             review_scores_cleanliness
```

```
## review_scores_rating                              0.8281561
## review_scores_accuracy                            0.7925776
## review_scores_cleanliness                         1.0000000
## review_scores_communication                       0.6999623
## review_scores_checkin                             0.6647535
## review_scores_value                               0.7751937
## review_scores_location                            0.6201472
##                             review_scores_communication
review_scores_checkin
## review_scores_rating                                0.8103266
0.7569332
## review_scores_accuracy                              0.7936822
0.7532662
## review_scores_cleanliness                           0.6999623
0.6647535
## review_scores_communication                         1.0000000
0.7836596
## review_scores_checkin                               0.7836596
1.0000000
## review_scores_value                                 0.7681240
0.7180183
## review_scores_location                              0.6556051
0.6442605
##                             review_scores_value review_scores_location
## review_scores_rating                  0.8691172              0.6952753
## review_scores_accuracy                0.8431411              0.6936656
## review_scores_cleanliness             0.7751937              0.6201472
## review_scores_communication           0.7681240              0.6556051
## review_scores_checkin                 0.7180183              0.6442605
## review_scores_value                   1.0000000              0.6993991
## review_scores_location                0.6993991              1.0000000

# Visualize correlations
ggcorrplot(cor_matrix, lab = TRUE, lab_size = 2, title = "Correlation Matrix
of Factors Impacting Review Ratings") +
  theme(plot.title = element_text(size = 13), axis.text.x = element_text(size
= 9), axis.text.y = element_text(size = 9))
```



Summary Statistics

```
#Summary Statistics for Listing Capacity across Room Type

get_mode <- function(x) {
  uniqx <- unique(x)
  uniqx[which.max(tabulate(match(x, uniqx)))]
}

summary_stats_accommodates <- airbnb_cleaned %>%
  group_by(room_type) %>%
  summarize(
    mean = mean(accommodates, na.rm = TRUE),
    median = median(accommodates, na.rm = TRUE),
    min = min(accommodates, na.rm = TRUE),
    max = max(accommodates, na.rm = TRUE),
    sd = sd(accommodates, na.rm = TRUE),
    mode = get_mode(accommodates),
    Inter_quertile = IQR(accommodates, na.rm=TRUE),
    count = n())

summary_stats_accommodates

## # A tibble: 4 × 9
##   room_type        mean median   min   max    sd  mode Inter_quertile
count
##   <chr>           <dbl>  <dbl> <dbl> <dbl> <dbl> <dbl>          <dbl>
```

```
<int>
## 1 Entire home/apt  4.66      4     1     16 2.81      2               4
24493
## 2 Hotel room       2.34      2     1      6 0.922     2               0
62
## 3 Private room     1.99      2     1     16 1.11      2               1
7530
## 4 Shared room      2.24      1     1     16 2.33      1               1
364
```

Build and Evaluate Price Prediction Model

```
#Load the libraries

library(rpart) # for creating decision tree model
library(rattle) # for plotting decision tree model
library(caret) # for evaluating decision tree model
library(randomForest) # for creating random forest model
library(rpart.plot) #for creating rpart plot
library(ISLR)

# Data Preparation and Feature Engineering

#Select relevant features from the original dataset

airbnb_model <- airbnb_filtered %>%
  select(
    host_is_superhost,
    room_type,
    accommodates,
    beds,
    price,
    instant_bookable,
    minimum_nights,
    number_of_reviews,
    review_scores_rating,
    reviews_per_month,
    neighbourhood_group
  )



# One-Hot Encoding for categorical variables
dummy <- dummyVars(price ~ host_is_superhost + neighbourhood_group +
room_type + instant_bookable, data = airbnb_model)
one_hot_encoded <- predict(dummy, newdata = airbnb_model)
one_hot_encoded_df <- as.data.frame(one_hot_encoded)




# Update dataset with one-hot encoded variables
```

```r
airbnb_model <- cbind(
  airbnb_model %>% select(-host_is_superhost, -neighbourhood_group, -
room_type, -instant_bookable),
  one_hot_encoded_df
)

# Standardize numeric features for Linear Regression
scale_features <- function(x) {
  (x - mean(x, na.rm = TRUE)) / sd(x, na.rm = TRUE)
}
num_cols <- c("accommodates", "beds", "minimum_nights", "number_of_reviews",
"review_scores_rating", "reviews_per_month")
airbnb_model[num_cols] <- lapply(airbnb_model[num_cols], scale_features)

# Apply log transformation to the target variable 'price' to stabilize
variance and reduce the effect of extreme values, making the data more
suitable for modeling.
airbnb_model$price <- log(airbnb_model$price)

# Create new features to improve the model
airbnb_model$accommodates_beds <- airbnb_model$accommodates *
airbnb_model$beds  # Interaction feature: accommodates × beds


# Clean column names
colnames(airbnb_model) <- colnames(airbnb_model) %>%
  gsub(" ", "_", .) %>%
  gsub("[^A-Za-z0-9_]", "", .) %>%
  tolower()
```

Split the Data

```r
set.seed(123)

# random sampling 70% of the rows based on the row number
training_index <- sample(c(1:nrow(airbnb_model)), 0.7*nrow(airbnb_model))

# use the index to select rows for train data set
train <- airbnb_model[training_index, ]
test <- airbnb_model[-training_index, ]

# check the dimensions of the training and test data set
dim(train)

## [1] 20695    19

dim(test)

## [1] 8870   19
```

Linear Regression Model

```
#Train the Linear Regression Model

lm_model <- lm(price ~ .+I(accommodates^2), data = train)

#Predictions and Evaluation
predictions_lm <- predict(lm_model, newdata = test)

#Extract the true label
true_label <- test$price

# Evaluation Metrics
mse_lm <- mean((true_label - predictions_lm)^2)
rmse_lm <- sqrt(mse_lm)
SS_res_lm <- sum((true_label - predictions_lm)^2)
SS_tot_lm <- sum((true_label - mean(true_label))^2)
r_squared_lm <- 1 - (SS_res_lm / SS_tot_lm)

# Print Metrics
cat("Linear Regression Model - Mean Squared Error (MSE):", mse_lm, "\n")

## Linear Regression Model - Mean Squared Error (MSE): 0.1690508

cat("Linear Regression Model - Root Mean Squared Error (RMSE):", rmse_lm,
"\n")

## Linear Regression Model - Root Mean Squared Error (RMSE): 0.4111579

cat("Linear Regression Model - R-squared (R²):", r_squared_lm, "\n")

## Linear Regression Model - R-squared (R²): 0.5313869
```

Decision Tree

```
# Set stopping parameters
control_params <- rpart.control(
  minsplit = 30,   # Minimum observations to split a node
  minbucket = 3,   # Minimum observations in a leaf node
  cp = 0.001,       # Complexity parameter
  maxdepth = 30    # Maximum depth of the tree
)




#Build the Decision tree model
dt_model <- rpart(price ~ ., data = train, method =  "anova", control =
control_params)

#Predictions and Evaluation
predictions_dt <- predict(dt_model, newdata = test)
```

```r
#Extract the true label
true_label <- test$price

# Evaluation Metrics
mse_dt <- mean((true_label - predictions_dt)^2)
rmse_dt <- sqrt(mse_dt)
SS_res_dt <- sum((true_label - predictions_dt)^2)
SS_tot_dt <- sum((true_label - mean(true_label))^2)
r_squared_dt <- 1 - (SS_res_dt / SS_tot_dt)

# Print Metrics
cat("Decision Tree Regression Model - Mean Squared Error (MSE):", mse_dt,
"\n")

## Decision Tree Regression Model - Mean Squared Error (MSE): 0.1612482

cat("Decision Tree Regression Model - Root Mean Squared Error (RMSE):",
rmse_dt, "\n")

## Decision Tree Regression Model - Root Mean Squared Error (RMSE): 0.4015572

cat("Decision Tree Regression Model - R-squared (R²):", r_squared_dt, "\n")

## Decision Tree Regression Model - R-squared (R²): 0.553016
```

Check if Decision Tree model is overfitted or not.

```r
#Predictions for Decision Tree on Training Data
predictions_dt_train <- predict(dt_model, newdata = train)

#Extract the train label
train_label <- train$price

mse_dt_train <- mean((train_label - predictions_dt_train)^2)
rmse_dt_train <- sqrt(mse_dt_train)
SS_res_dt_train <- sum((train_label - predictions_dt_train)^2)
SS_tot_dt_train <- sum((train_label - mean(train_label))^2)

# R-squared (proportion of variance explained)
r_squared_dt_train <- 1 - (SS_res_dt_train / SS_tot_dt_train)

# Print the Training Metrics
cat("Training Metrics for Decision Tree Model:\n")

## Training Metrics for Decision Tree Model:

cat("DT MSE (Train):", mse_dt_train, "\n")

## DT MSE (Train): 0.1619296

cat("DT RMSE (Train):", rmse_dt_train, "\n")
```

```
## DT RMSE (Train): 0.4024048

cat("DT R-squared (Train):", r_squared_dt_train, "\n")

## DT R-squared (Train): 0.5591968
```

Random Forest

```
#Build the Random Forest Regression model
rf_model <- randomForest(price ~ ., data = train, ntree = 100, mtry = 3)

#Predictions and Evaluation
predictions_rf <- predict(rf_model, newdata = test)

#Extract Actual label
true_label <- test$price

# Evaluation Metrics
mse_rf <- mean((true_label - predictions_rf)^2)
rmse_rf <- sqrt(mse_rf)
SS_res_rf <- sum((true_label - predictions_rf)^2)
SS_tot_rf <- sum((true_label - mean(true_label))^2)
r_squared_rf <- 1 - (SS_res_rf / SS_tot_rf)

# Print Metrics
cat("Random Forest Regression Model - Mean Squared Error (MSE):", mse_rf,
"\n")

## Random Forest Regression Model - Mean Squared Error (MSE): 0.1477122

cat("Random Forest Regression Model - Root Mean Squared Error (RMSE):",
rmse_rf, "\n")

## Random Forest Regression Model - Root Mean Squared Error (RMSE): 0.3843334

cat("Random Forest Regression Model - R-squared (R²):", r_squared_rf, "\n")

## Random Forest Regression Model - R-squared (R²): 0.5905381
```

Check if the Random Forest model is overfitted or not.

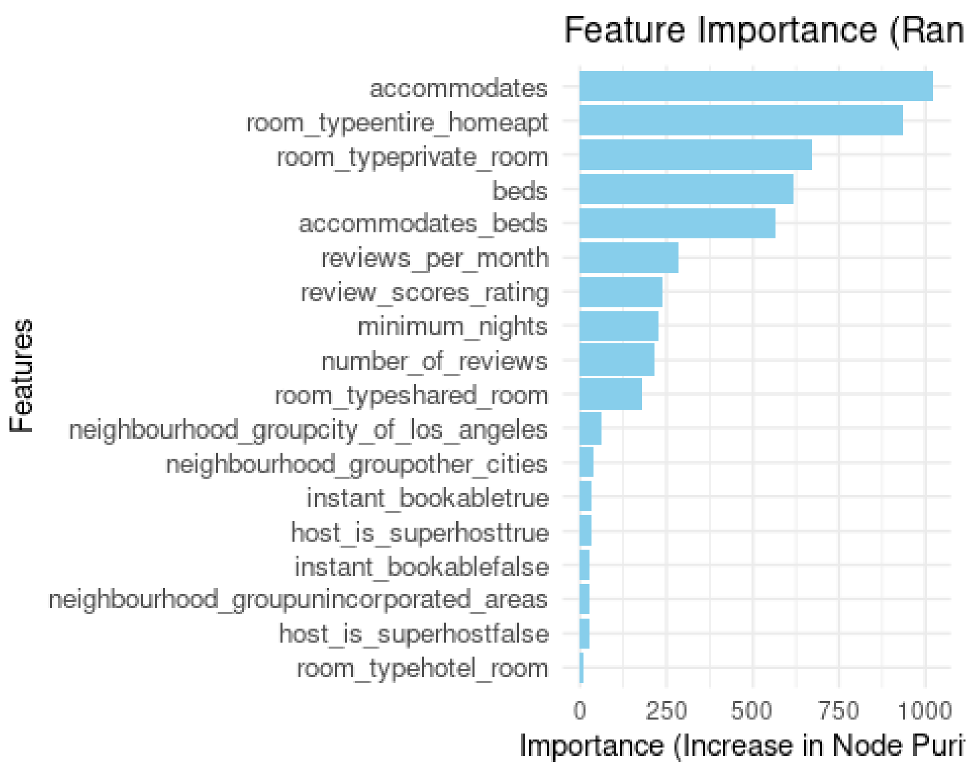
```
# Predictions for Random Forest on Training Data
predictions_rf_train <- predict(rf_model, newdata = train)

#Extract the train label
train_label <- train$price



# Evaluate the Random Forest Model on Training Data

mse_rf_train <- mean((train_label - predictions_rf_train)^2)
rmse_rf_train <- sqrt(mse_rf_train)
```

```r
SS_res_rf_train <- sum((train_label - predictions_rf_train)^2)
SS_tot_rf_train <- sum((train_label - mean(train_label))^2)

# R-squared (proportion of variance explained)
r_squared_rf_train <- 1 - (SS_res_rf_train / SS_tot_rf_train)

# Print the Training Metrics
cat("Training Metrics for Random Forest Model:\n")

## Training Metrics for Random Forest Model:

cat("RF MSE (Train):", mse_rf_train, "\n")

## RF MSE (Train): 0.1067893

cat("RF RMSE (Train):", rmse_rf_train, "\n")

## RF RMSE (Train): 0.3267864

cat("RF R-squared (Train):", r_squared_rf_train, "\n")

## RF R-squared (Train): 0.7092991
```

Feature Importance

```r
# Extract feature importance from the Random Forest model
feature_importance <- importance(rf_model)

# Print the feature importance to inspect its structure
print(feature_importance)

##                                          IncNodePurity
## accommodates                               1021.910574
## beds                                        616.581812
## minimum_nights                              227.896048
## number_of_reviews                           213.483402
## review_scores_rating                        240.558656
## reviews_per_month                           283.726208
## host_is_superhostfalse                       28.565936
## host_is_superhosttrue                        32.535682
## neighbourhood_groupcity_of_los_angeles       61.949116
## neighbourhood_groupother_cities              37.738789
## neighbourhood_groupunincorporated_areas      28.830904
## room_typeentire_homeapt                     932.498810
## room_typehotel_room                           8.068858
## room_typeprivate_room                       671.938825
## room_typeshared_room                        182.637746
## instant_bookablefalse                        30.476396
## instant_bookabletrue                         32.695634
## accommodates_beds                           565.078126
```

```r
# Convert importance values into a data frame for visualization
feature_importance_df <- data.frame(
  Feature = rownames(feature_importance),
  Importance = feature_importance[, "IncNodePurity"]
) %>%
  arrange(desc(Importance))  # Sort features by importance

# Plot the feature importance
ggplot(feature_importance_df, aes(x = Importance, y = reorder(Feature,
Importance))) +
  geom_bar(stat = "identity", fill = "skyblue") +  # Horizontal bar chart
  theme_minimal() +
  labs(
    title = "Feature Importance (Random Forest)",  # Chart title
    x = "Importance (Increase in Node Purity)",    # X-axis label
    y = "Features"                                # Y-axis label
  ) +
  theme(axis.text.y = element_text(size = 10))  # Adjust font size for
readability
```



Note that the echo = FALSE parameter was added to the code chunk to prevent printing of the R code that generated the plot.