FinalProject_IS457_107.R

[redacted]

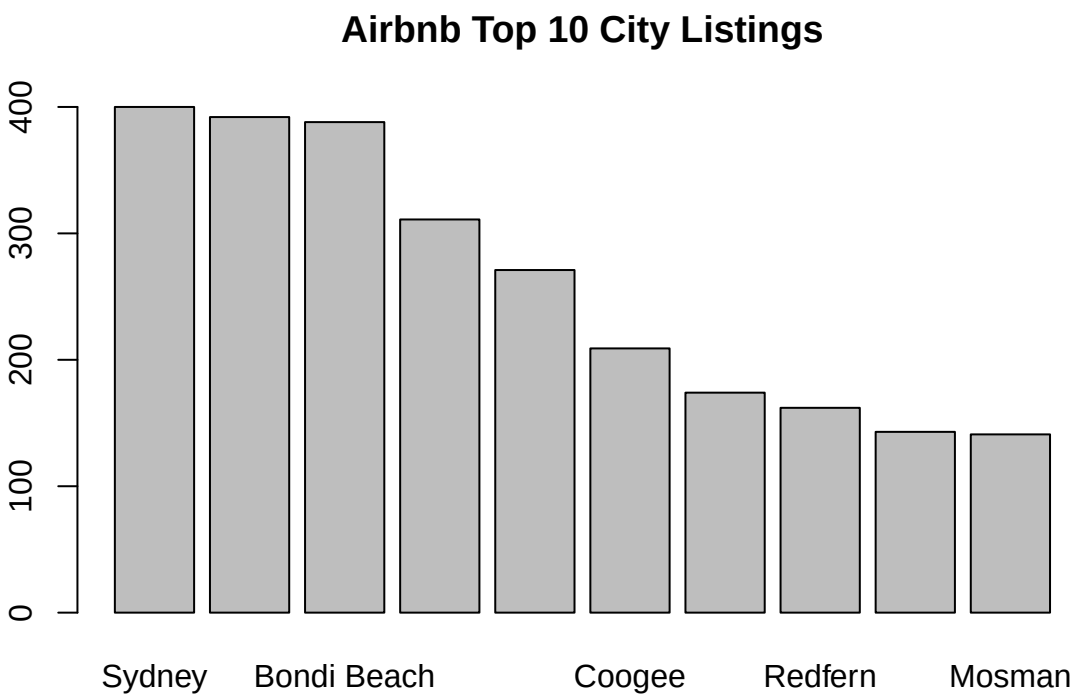
Mon Apr 29 02:50:46 2019

```
# QUESTION 1

Airbnb_dataset <- read.csv(file="Airbnb-Sydney.csv", header=TRUE, sep=",")

# nrow(Airbnb_dataset[Airbnb_dataset$host_response_rate == "N/A", ])  prints 2483
# nrow(Airbnb_dataset)

# What variables have missing values?
# Neighborhood_overview, house_rules, host_response_time,
# host_response_rate have missing values.
# What types/forms of missing values are they? (e.g blank,
# NA, N/A, -, etc.)
# Neighborhood_overview and house_rules are missing
# character descriptions. Their NAs are blank cells.
# Host_response_times and host_response_rate display N/A instead of blank
# cells, both are missing character descriptions. With host_response_rate
# being a percentage. Whenever either Host_response_times or host_response_rate
# is N/A the other is also N/A.

# Please briefly describe how you deal will with these missing values and justify
# why you chose these methods
# I would drop the observations with N/As. This is because there
# are 2483 values with N/As for host_response_rate
# and Host_response_times, it would skew the data too much if we were to
# replace all N/As with mean, median, or mode
# data. The data would lose accuracy. Hence we should drop the observations
# with NAs.

# Describe how your choice of method to deal with missing values
# will affect your later analysis.
# The dataset contains 10815 rows, dropping 2483 rows would mean we
# lose roughly 23% of our data.
# This would mean we would lose all data associated with those rows.
# This will limit our dataset
# and observations we can make.

row.has.na <- apply(Airbnb_dataset, 1, function(x){any(is.na(x))  || any(x == "N/A")} )
Airbnb_dataset <- Airbnb_dataset[!row.has.na,]
dim(Airbnb_dataset)
```

```
## [1] 8327   36
```

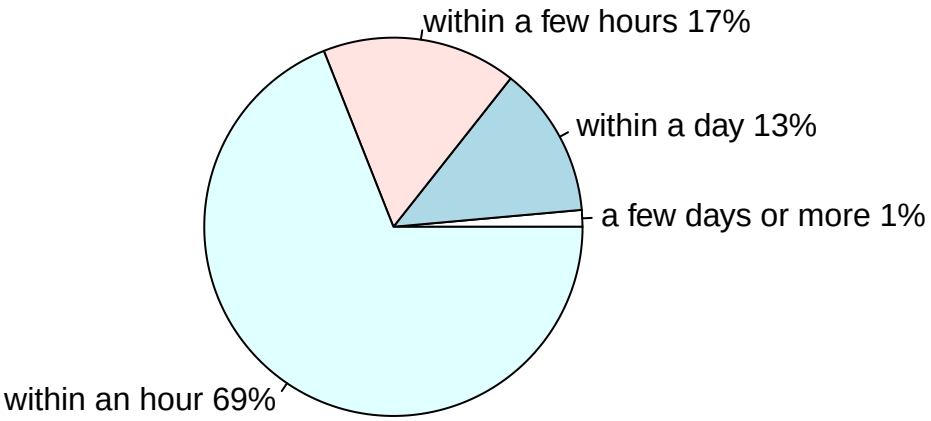
```
# Comment on and explain any other data cleaning or preparation steps you think would be
# necessary from your inspection of the data (you do not need to carry them out).
# It is possible we can remove a row if it contains a blank value for a description.
# I think at this stage, however that would be uneccessary, and would result in us losing
# more data than we should.
```

```r
# QUESTIONS 2 & 3

# Since it is hard to get an overview for large data sets, conduct a
# preliminary exploration to explore the variables of the Sydney Airbnb data by
# calculating some
# descriptive and distributional statistics. Describe what you find that is
# unexpected or interesting.

# summary(Airbnb_dataset)

cities = as.character(Airbnb_dataset$city)
top_cities = head(sort(table(cities),decreasing=TRUE), n = 10)
barplot(top_cities, main = "Airbnb Top 10 City Listings")
```

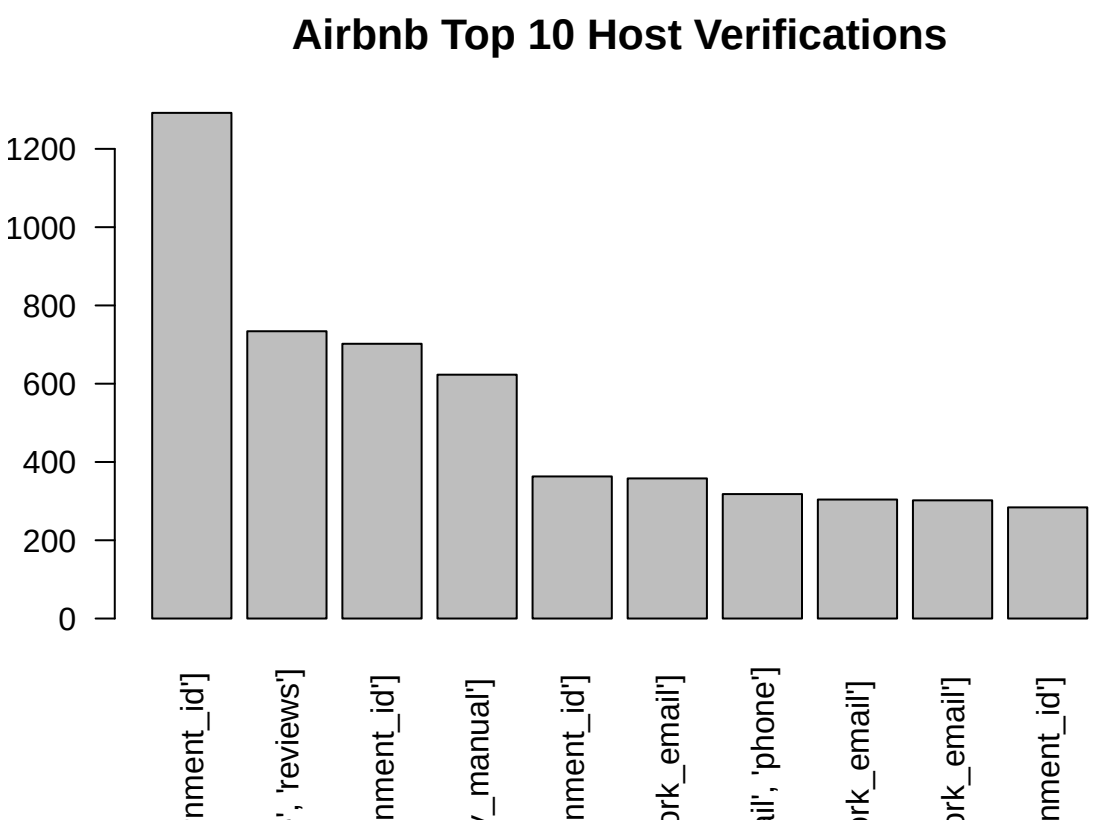



```r
# The top Airbnb listings for cities are Sydney, Surry Hills, Bondi Beach etc.
# As one of the most popular cities for tourists and the basis of our dataset it is not
# suprising Syndey has the most Airbnb listings. It is interesting to see
# where around Syndey Airbnbs are listed for.
# Such as Surry Hills and Bondi Beach, both are very
# touristy areas with great scenery around Syndey.

response_times = as.character(Airbnb_dataset$host_response_time)
response_times_df = as.data.frame(table(response_times))

pct <- round(response_times_df$Freq/sum(response_times_df$Freq)*100)
lbls <- paste(response_times_df$response_times, pct)
lbls <- paste(lbls,"%",sep="")

pie(table(response_times), main = "Frequency of Host Reponse Times", labels = lbls)
```

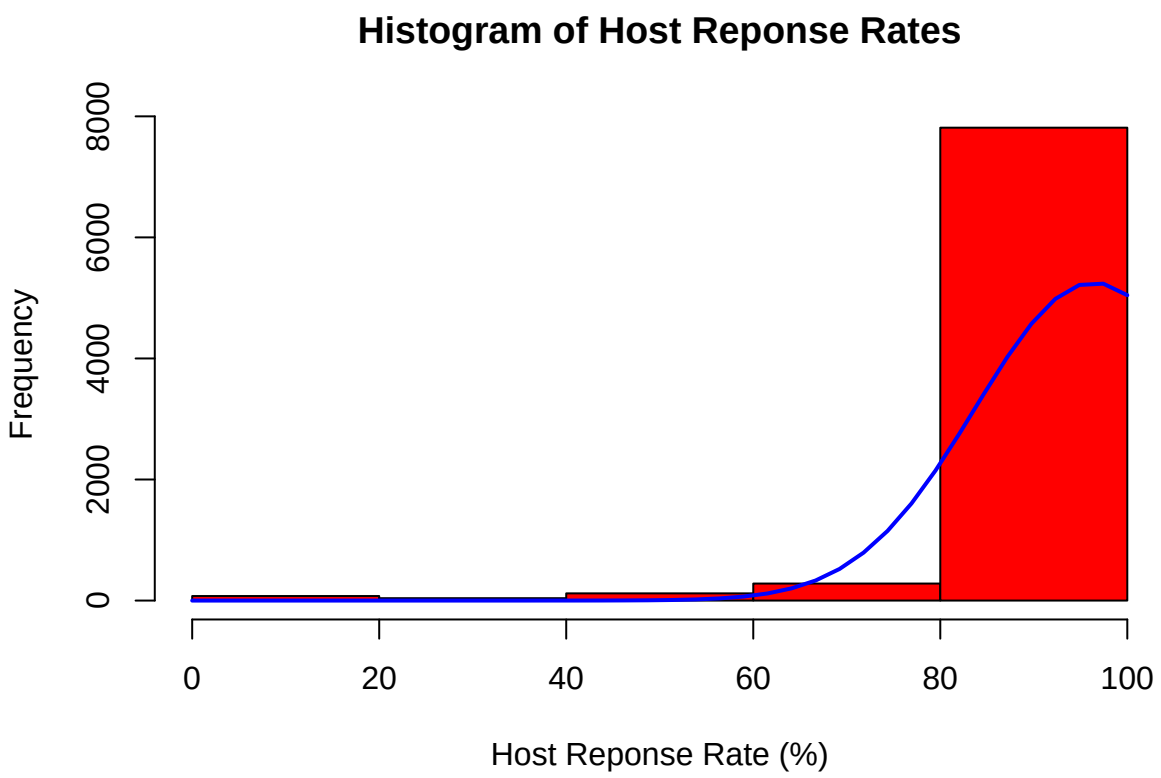
Frequency of Host Reponse Times




```
# Based on the data most hosts respond within an hour 69%.
# Significantly less hosts require more than an hour
# to respond. For example, hosts responding within a
# couple hours is 17%.


host_verifications = as.character(Airbnb_dataset$host_verifications)
top_host_verifications = head(sort(table(host_verifications),decreasing=TRUE), n = 10)

barplot(top_host_verifications, main = "Airbnb Top 10 Host Verifications", las= 2, cex.main=1.3)
```

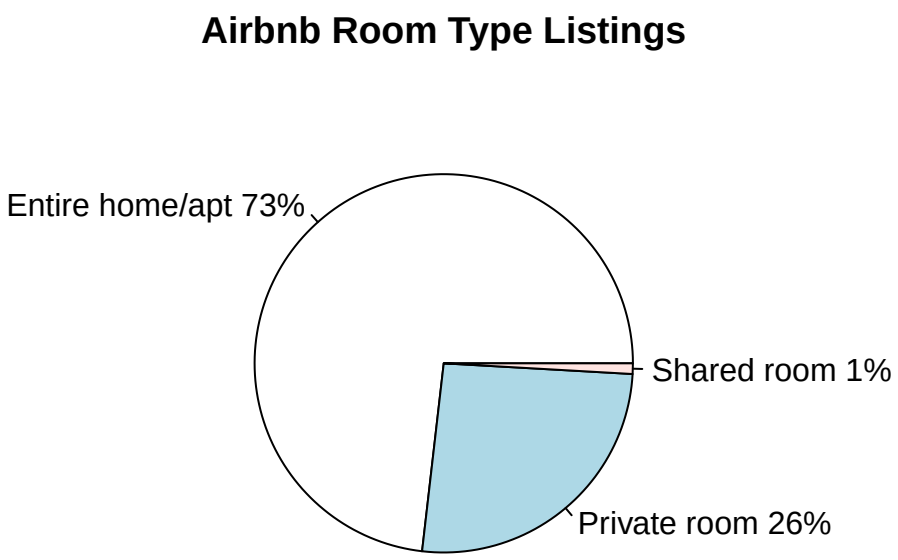


```
# Most hosts provide email, phone, reviews, jumio, and government_id verifications
# The only common verification provided by all of the top 10 host verifications
# is email and phone #.



host_resp_rates_percent = as.character(Airbnb_dataset$host_response_rate)
host_resp_rates_numeric  = as.numeric(sub("%", "", host_resp_rates_percent))
x <- host_resp_rates_numeric
h<-hist(x, breaks=5, col="red", xlab="Host Reponse Rate (%)",
        main="Histogram of Host Reponse Rates")
xfit<-seq(min(x), max(x), length=40)
yfit<-dnorm(xfit, mean=mean(x), sd=sd(x))
yfit <- yfit * diff(h$mids[1:2]) * length(x)
lines(xfit, yfit, col="blue", lwd=2)
```




```
# The data here is skewed left. The data is not symmetric, not uniformly distributed.
# The data is unimodal (has one peak) at 100% host reponse rate.
# Hosts are most likely to respond
# 100% of the time.



room_type = as.character(Airbnb_dataset$room_type)
room_type_df = as.data.frame(table(room_type))

pct <- round(room_type_df$Freq/sum(room_type_df$Freq)*100)
lbls <- paste(room_type_df$room_type, pct)
lbls <- paste(lbls,"%",sep="")
```

```r
pie(table(room_type), main = "Airbnb Room Type Listings", labels = lbls)
```




```r
# 73% of room types listed are entire home/apt. This is the most popular
# room type listing

# Question 4
# Now look at the relationships among several variables.

# For example, look at "review per month" and "number of reviews."
# They are both indicators of number of guests, but convey different information.
# What different information do they convey and what similar information?
# How are the two variables related? Answer this quantitatively.


# Find the top number of reviews per month and the top number of
# reviews total
top_rev_p_month_rows = head(Airbnb_dataset[order(Airbnb_dataset$reviews_per_month, decreasing= T),], n =
top_num_rev = head(Airbnb_dataset[order(Airbnb_dataset$number_of_reviews, decreasing= T),], n = 100)

combined_ids = as.vector(top_rev_p_month_rows$id)
combined_ids = append(combined_ids, as.vector(top_num_rev$id), after = length(combined_ids))

indivListings = length(unique(combined_ids))/length(combined_ids)
paste0(formatC(100 - indivListings *100, format = "f", digits = 2), "%")
```

```
## [1] "10.00%"
```

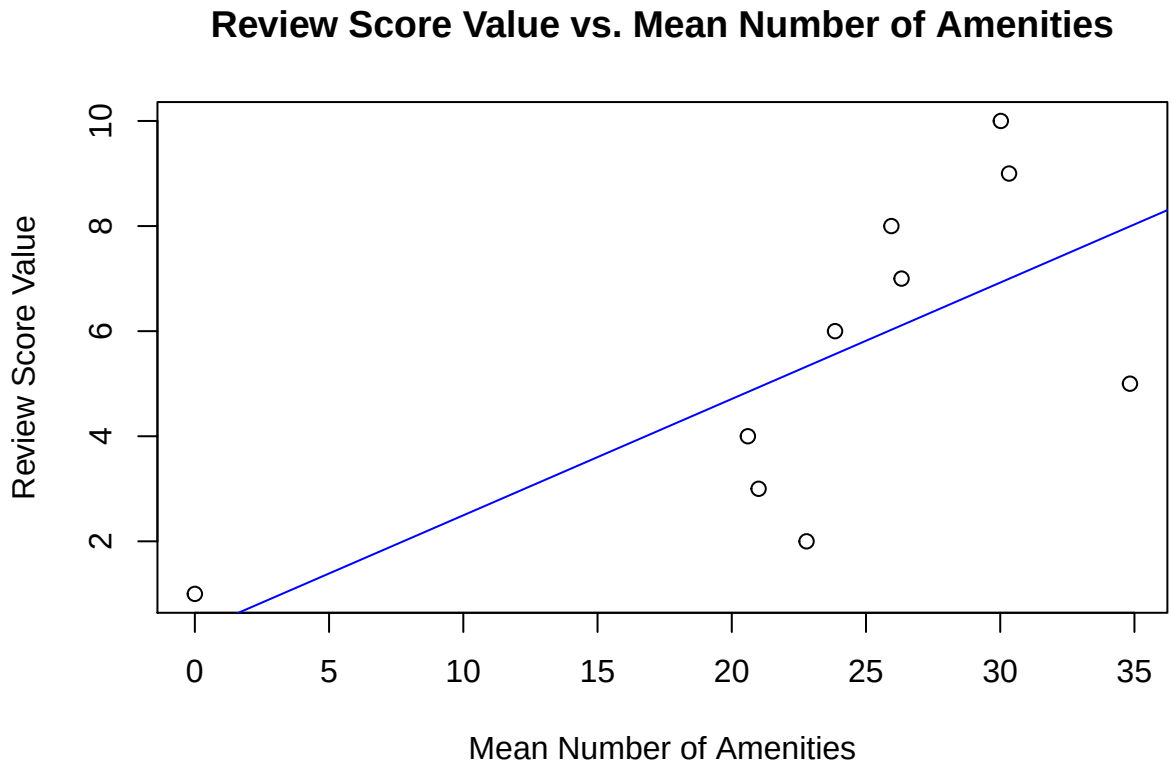
```r
# Out of each of the top 100 listings for reviews per month and total number of reviews,
# (that is 200 listings), 180 of the listings are unique listings. Meaning only 10% (20/200) of
# listings are both in the top 100 number of reviews and top 100 number of reviews per month.


# Find the overlapping host ids
overlapping_host_ids <- intersect(top_rev_p_month_rows$host_id, top_num_rev$host_id)

# Get the rows corresponding to the overlapping host ids
overlapping_rows_top_month = top_rev_p_month_rows[
```

```
  match(overlapping_host_ids,top_rev_p_month_rows$host_id),]
overlapping_rows_top_num = top_num_rev[match(overlapping_host_ids,top_num_rev$host_id),]

percentDifferentListings = length(intersect(
  overlapping_rows_top_month$id, overlapping_rows_top_num$id))/nrow(overlapping_rows_top_month)
percentDifferentListings = paste0(formatC(100 - (percentDifferentListings *100),
                                          format = "f", digits = 2), "%")
print(percentDifferentListings)
```

```
## [1] "11.11%"
```

```
# So 11.11% of the hosts with the top total number of reviews and top reviews per
# month have different Airbnb listings that have made the top in each category.
# Whereas 88.89% of hosts have the same listing that is both in the top total number
# of reviews and top reviews per month. So we can conclude as a host it is most likely
# to have a single one of your listings top in both categories, as opposed to one listing
# top in one category and a different listing top in the other.



# I have a hypothesis that the more amentities offered the higher
# people will rate review scores value because they feel the listing
# provided good value for the price.


listing_amenities = as.vector(Airbnb_dataset$amenities)
listing_amenities = strsplit(listing_amenities, ",")

num_amenities = c()
for(val in listing_amenities){
  num_amenities = c(num_amenities, length(val))
}


xy_amenities_scores = as.data.frame(cbind(num_amenities, Airbnb_dataset$review_scores_value))
colnames(xy_amenities_scores) = c("x", "y")


# Find the mean number of ammentities for each rating score value.
all_rating_amenities_mean = c()
for(i in 1:10){
  rating_amenities_mean =
    as.data.frame(xy_amenities_scores[xy_amenities_scores$y == i,])

  all_rating_amenities_mean = c(all_rating_amenities_mean,
                                mean(rating_amenities_mean$x))
}

# Replace NaNs with 0
all_rating_amenities_mean[is.nan(all_rating_amenities_mean)] <- 0
xy_coords = as.data.frame(cbind(all_rating_amenities_mean, 1:10))
colnames(xy_coords) = c("x", "y")

plot(xy_coords,
```

```
     main ="Review Score Value vs. Mean Number of Amenities",
     xlab="Mean Number of Amenities", ylab = "Review Score Value")

abline(lm(xy_coords$y~xy_coords$x), col = "blue")
```

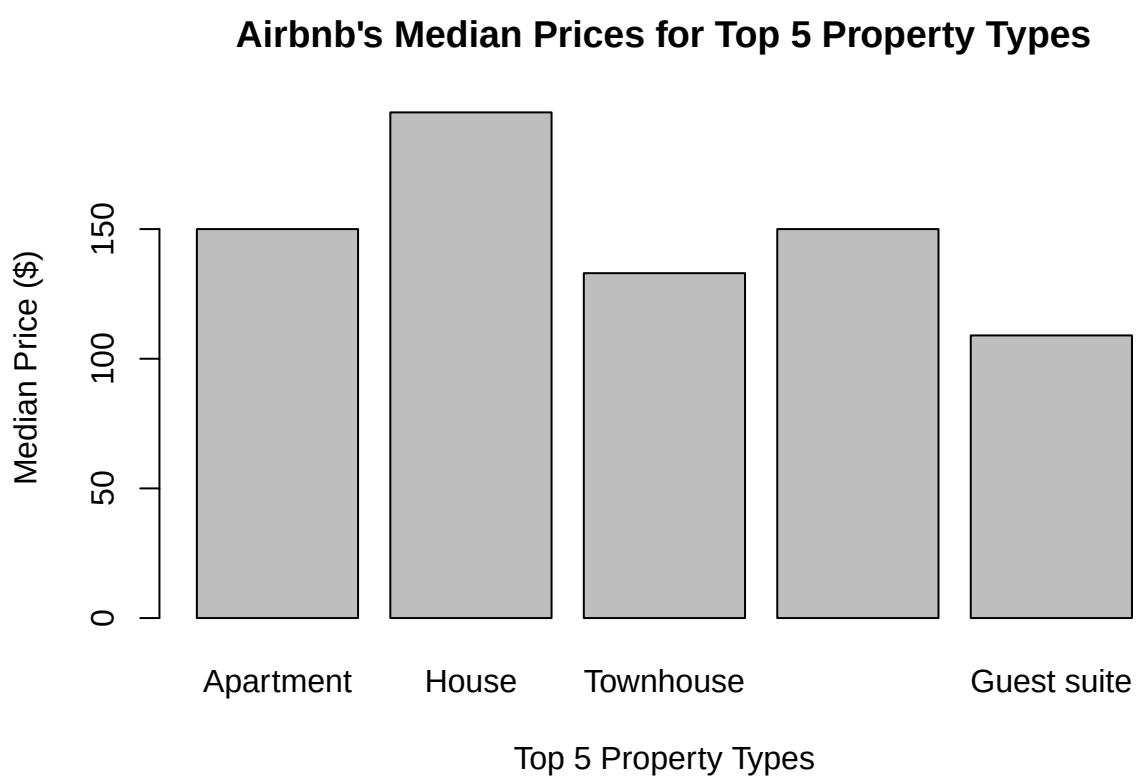



```
# Correlation Coefficient
print(cor(xy_coords$y, xy_coords$x))
```

```
## [1] 0.6892049
```

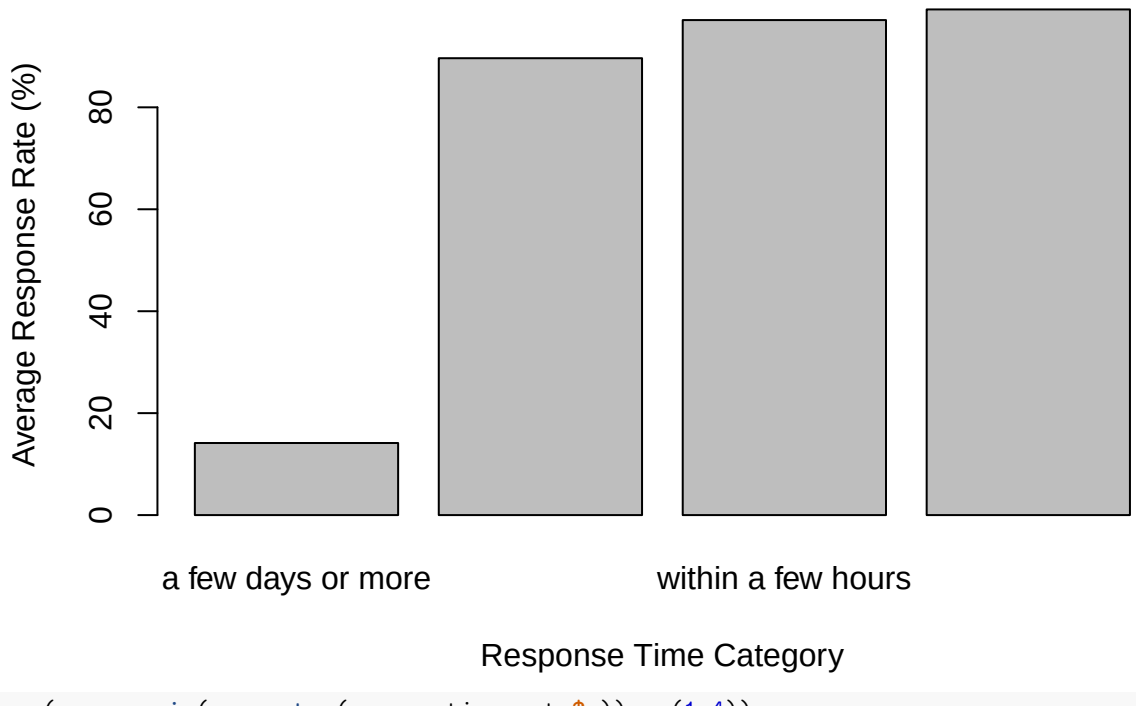
```
# As we can see from the graph above. There is postive
# correlation between offering more amenities and
# higher review score value. This is inherently because
# people feel they are getting their money's worth.
# Airbnb should encourage its hosts to list all their
# amenities and prioritize listings that have more
# amenities than others. It results in happier customers.



# I have a hypothesis that the different propertes types
# will have different pricing. The airbnb dataset
# lists 31 different property types I would like to compare
# the top 5 property types and their pricing.


# Filter Airbnb dataset to not include rows with empty cells for
# property type or pricing
filtered_airbnb_data = Airbnb_dataset[
  (Airbnb_dataset$property_type != "") & (Airbnb_dataset$price != ""),   ]
```

```r
# Top 5 kinds of properties
top_prop_types = as.data.frame(head(sort(
  table(filtered_airbnb_data$property_type),
  decreasing=TRUE), n = 5))
colnames(top_prop_types) = c("PropertyType", "Frequency")

# Convert top 5 property types to vector
prop_types_vec = as.vector(top_prop_types$PropertyType)

median_price_per_type = c()

# Median price corresponding to each property type
for(type in prop_types_vec){
  typ_filter = filtered_airbnb_data[filtered_airbnb_data$property_type == type, ]
  median_price_per_type = c(median_price_per_type,
                            median(as.numeric(gsub('\\$|,', '', typ_filter$price))))
}

xy_prop_median_price = as.data.frame(cbind(prop_types_vec, median_price_per_type))
colnames(xy_prop_median_price) = c("x", "y")

barplot(as.numeric(as.vector(xy_prop_median_price$y)),
        names.arg=as.vector(xy_prop_median_price$x) ,
        main = "Airbnb's Median Prices for Top 5 Property Types",
        xlab = "Top 5 Property Types", ylab = "Median Price ($)")
```




```r
# Based on the results the rankings from highest to lowest price
# listing is House > Condominium = Apartment > Townhouse > Guest Suite
```

```r
# Generally townhouses include more sqft and amenities,
# it is suprising that from the listings their median price is $17 less
# than an apartment! Why is that? Is it because there is more
# demand for apartments than townhouses?


# I wonder what the relationship is between host response rate and host response
# time. I predict hosts with quicker response times will also have higher
# response rates as well.


host_response_times_vec = as.vector(Airbnb_dataset$host_response_time)
host_reponse_rate_vec =  as.vector(Airbnb_dataset$host_response_rate)
host_reponse_time_types = as.vector(
  as.data.frame(table(host_response_times_vec))$host_response_times_vec)

avg_reponse_rate_by_time = c()

for(type in host_reponse_time_types){
  typ_filter = Airbnb_dataset[Airbnb_dataset$host_response_time == type, ]
  avg_reponse_rate_by_time = c(avg_reponse_rate_by_time,
                               mean(as.numeric(sub("%", "",
                                                   typ_filter$host_response_rate))))
}


xy_rep_time_rate = as.data.frame(cbind(host_reponse_time_types,
                                       avg_reponse_rate_by_time))
colnames(xy_rep_time_rate) = c("x", "y")

barplot(as.numeric(as.vector(xy_rep_time_rate$y)),
        names.arg=as.vector(xy_rep_time_rate$x) ,
        main = "Airbnb Host's Average Reponse Rate by Response Time",
        xlab = "Response Time Category", ylab = "Average Response Rate (%)")
```




```
cor(as.numeric(as.vector(xy_rep_time_rate$y)), c(1:4))
```

```
## [1] 0.8312174
```

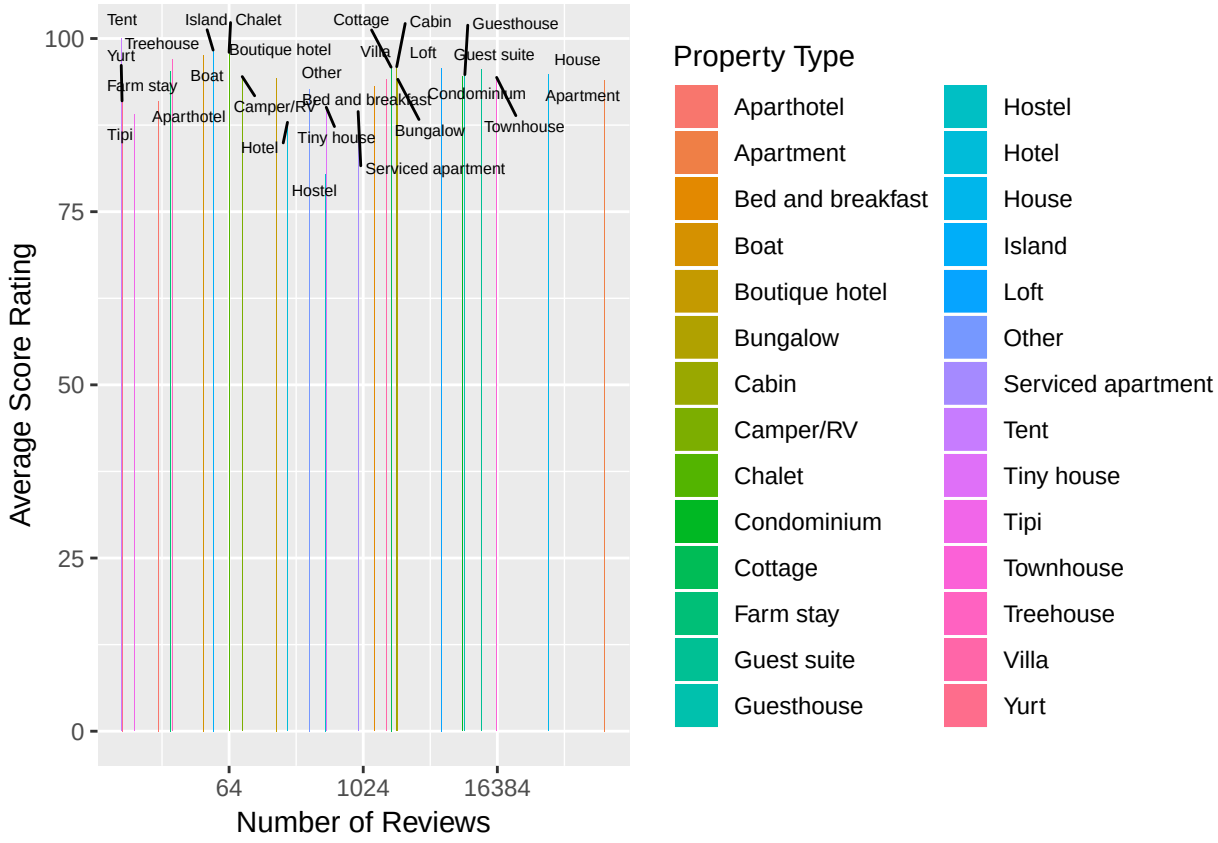
```
# According to the results hosts with the quickest response times
# also have the highest reponse rate. This proves my hypothesis is
# correct.


# QUESTION 5
```

```
# Business Hypotheses
```

```
# I have a business hypothesis that the greater number of amentities a
# host offers the higher a customer will rate their review_score_value.
# Because then customers feel like they are getting the most
# value for what they are paying for. We can also look at pricing to see
# if a greater number of amentites combined with a lower price results
# in the highest review_score value. Relevant variables:
# amentities, review_score_value, price

# I have a business hypothesis that certain cities and property types
# will have higher pricings than others I would like to isolate the top
# property types listed and the top cities listed, and analyze
# prices for the top cities property types. Uses these results
# Airbnb could the property types and cities than are the most pricey
# in the Syndey area. Relevant variables: cities, property_type, price

# I have a business hypothesis that certain hosts with a quick response time
# will also have a high reponse rate and have a higher overall review. I
# also predict they are more likely to be a superhost. Relevant variables:
```

```r
# response_time, response_rate, review_scores_rating, superhost




#install.packages("ggplot2")
library(ggplot2)
#install.packages("lattice")
library(lattice)

#install.packages("ggrepel")
library(ggrepel)

# QUESTION 6

# Reviews are an important factor when users choose listings. Users have
# different expectations for different types of property.
#  For example, if a user chooses a house, he might expect the house
# to be more spacious than an apartment. Therefore, we would like to see
# if property type affects how users give their reviews.
# Make ONE plot to visualize the relationship between "review scores rating" and
# "number of reviews" for all categories of "property type." Explain what you
# find from your graph.

# 6.1
plot_data = as.data.frame(cbind(as.character(Airbnb_dataset$property_type),
                                Airbnb_dataset$review_scores_rating,
                                Airbnb_dataset$number_of_reviews))
colnames(plot_data) = c("PropType", "ScoreRating", "NumReviews")


property_types_vec = as.vector(as.data.frame(table(plot_data$PropType))$Var1)

total_num_revs = c()
avg_score_rating = c()

for(type in property_types_vec){
  typ_filter = plot_data[plot_data$PropType == type, ]
  total_num_revs = c(total_num_revs, sum(as.numeric(
    as.vector(typ_filter$NumReviews))))
  avg_score_rating = c(avg_score_rating, mean(as.numeric(
    as.vector(typ_filter$ScoreRating))))
}


filtered_plot_data = as.data.frame(cbind(property_types_vec, total_num_revs,
                                         avg_score_rating))
colnames(filtered_plot_data) = c("PropType", "NumReviews", "AvgScoreRating")

filtered_plot_data$AvgScoreRating =
  (as.numeric(as.character(filtered_plot_data$AvgScoreRating)))
filtered_plot_data$NumReviews =
  (as.numeric(as.character(filtered_plot_data$NumReviews)))
```

```
ggplot(data = filtered_plot_data,
       aes(fill=filtered_plot_data$PropType, y=AvgScoreRating,
           x=NumReviews)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Number of Reviews", y = "Average Score Rating",
       fill = "Property Type") +
  geom_text_repel(aes(label = filtered_plot_data$PropType), size = 2) +
  scale_x_continuous(trans = "log2")+
  scale_y_continuous(limits = c(0, 100))
```

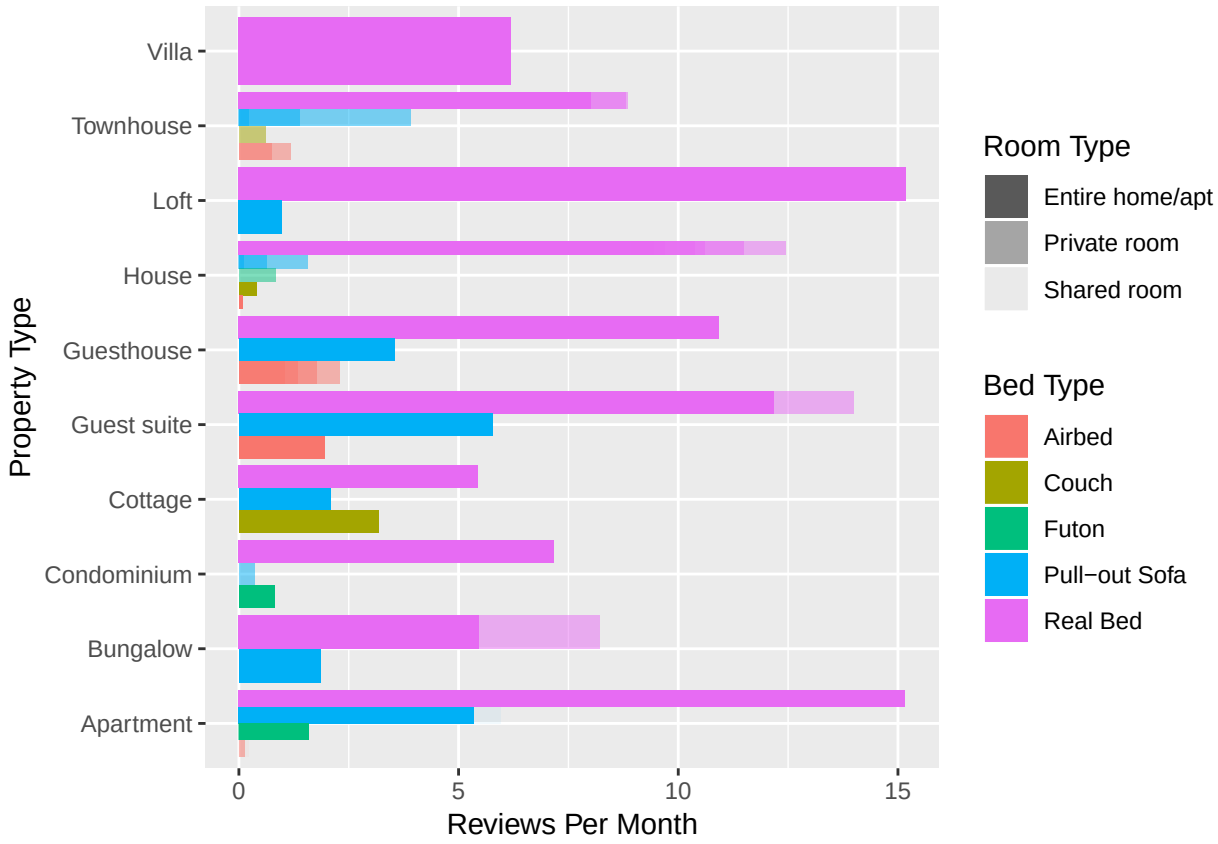



```
# From the data we find that Apartments > Houses > Townhouses
# had the greatest number of reviews. But just because a property
# has a greater number of reviews does not mean its average score
# is higher. Take the Tent property, for example, it only has 7
# reviews however it is the only property to have an average score
# rating of 100.

# 6.2

# Find top 10 property types
top_prop_types = as.data.frame(head(sort(
  table(Airbnb_dataset$property_type),
  decreasing=TRUE), n = 10))
```

```r
# Extract the top property rows
top_prop_data = filtered_plot_data[
  match(top_prop_types$Var1, filtered_plot_data$PropType),]

# Sort the property rows by number of reviews (ascending).
top_prop_data = top_prop_data[
  order(as.numeric(as.vector(top_prop_data$NumReviews))), ]

# Print top number of reviews for top property types
tail(top_prop_data)
```

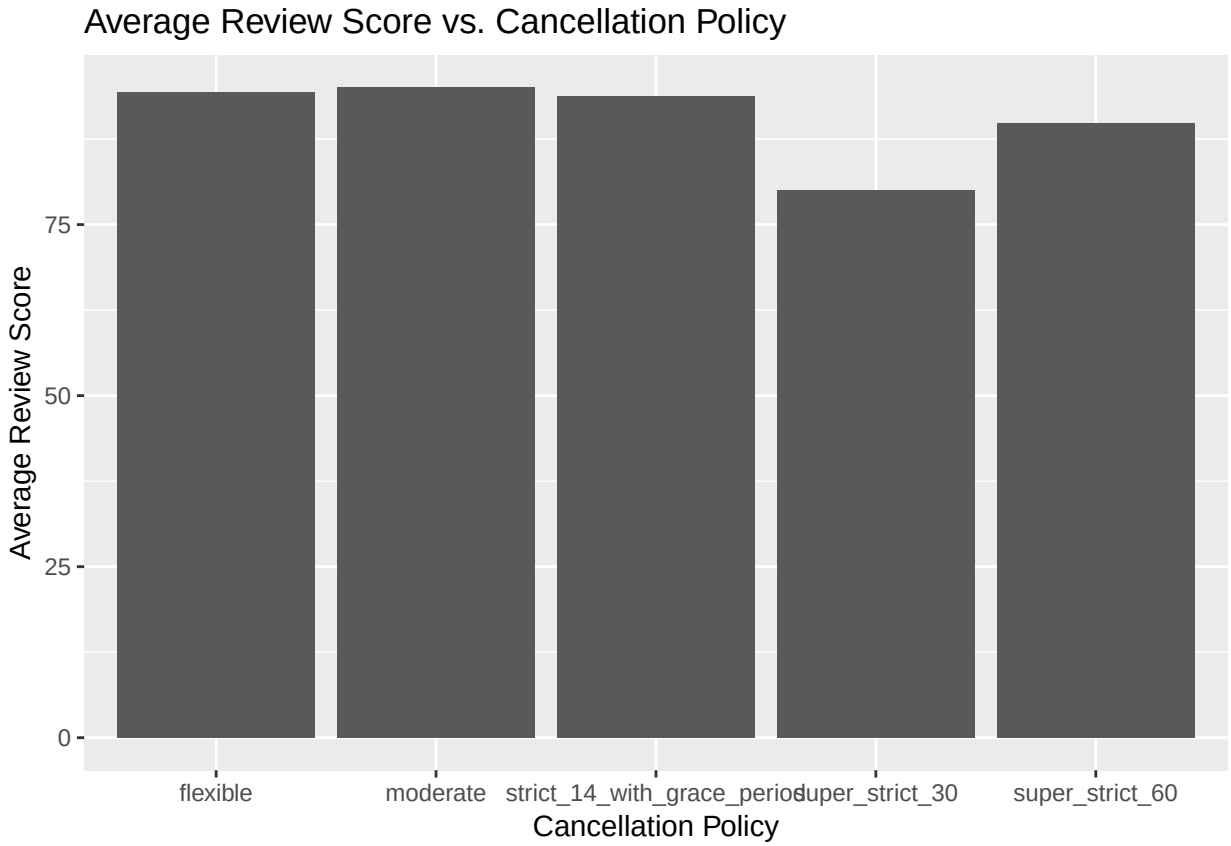
```
##       PropType NumReviews AvgScoreRating
## 10 Condominium       7925       94.55133
## 14  Guesthouse       8370       94.79096
## 13 Guest suite      11817       95.54260
## 25   Townhouse      16147       94.38280
## 17       House      46707       94.77154
## 2    Apartment     151609       94.00258
```

```r
dataset_variables = as.data.frame(cbind(as.character(Airbnb_dataset$property_type),
                                        as.character(Airbnb_dataset$room_type),
                                        as.character(Airbnb_dataset$bed_type),
                                        Airbnb_dataset$reviews_per_month))

colnames(dataset_variables) = c("PropTyp", "RoomTyp", "BedTyp", "RevPMonth")

top_prop_types_vec = as.character(as.vector(top_prop_data$PropType))



dataset_variables = dataset_variables[
  dataset_variables$PropTyp %in% top_prop_types_vec, ]

dataset_variables$RoomTyp = as.vector(as.character(dataset_variables$RoomTyp))



# This took me hours. But it looks awesome.
ggplot(dataset_variables, aes(x=dataset_variables$PropTyp,
                              y=as.numeric(
                                as.vector(dataset_variables$RevPMonth)),
                              group=dataset_variables$BedTyp,
                              fill=dataset_variables$BedTyp,
                              alpha = dataset_variables$RoomTyp)) +
  geom_bar(stat="identity",position="dodge") +
  # scale_fill_manual(values = c("red", "green", "blue", "white") didn't look nice
  scale_alpha_manual(values=c(1, 0.5, 0.05)) +
  labs(x = "Property Type", y = "Reviews Per Month",
       fill = "Bed Type", alpha = "Room Type") +
  coord_flip()
```



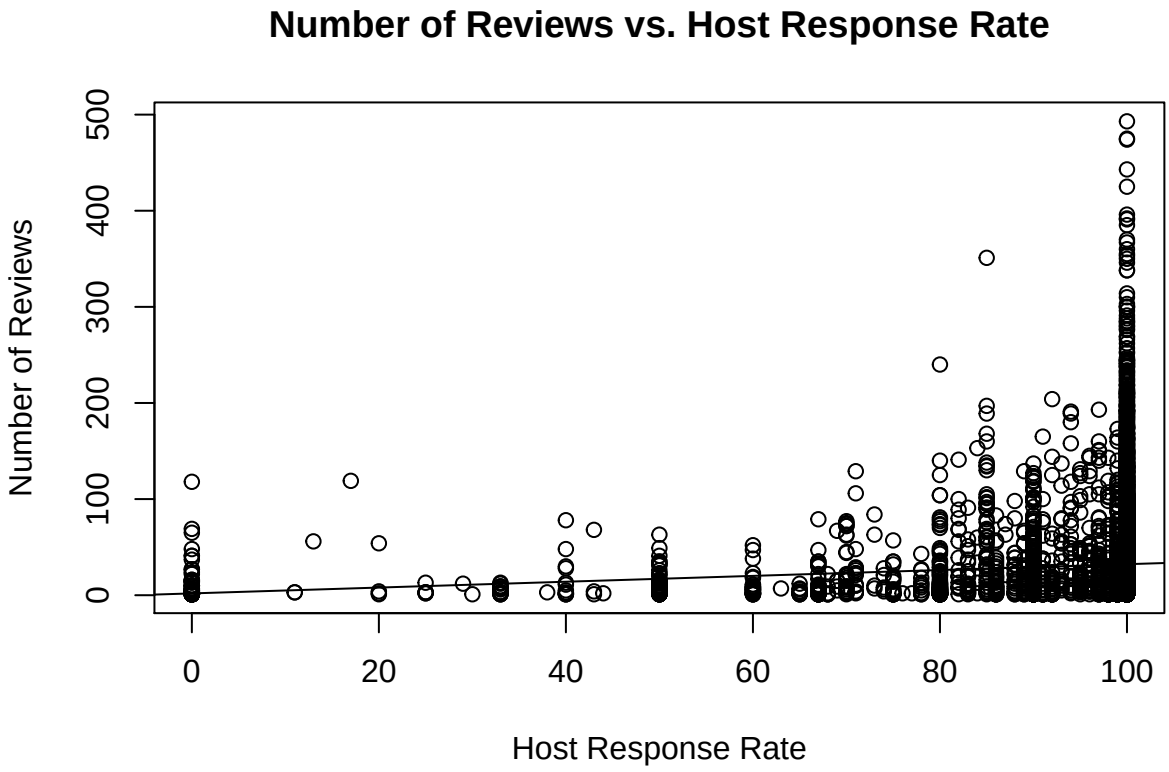

```
# There are a number of interesting conclusions to
# draw from this graph. First to explain I used
# different fill colors for room types and different alphas to represent
# different bed types. Also if a room type or bed type doesn't exist
# for a certain property type a bar is not created for it
# or alpha value not displayed for it.

# Once conlusion to draw is that for all top 10 properties
# types the most reviews written are for Real Bed room types combined
# with Entire home/apt bed types. The most reviews per month are given to
# property type:Loft, Room Type: Real Bed, Bed Type: Entire home/apt.
# For 7/10 top property types pull-out-sofas are the room type
# to get the second most amount of reviews per month (after real beds).



# 6.3 Completed in Question 3. Refer back to Question 3.


# Question 7

# 7.1
# Clean price data. Convert to numeric.
Airbnb_dataset$price = as.numeric(gsub('\\$|,', '', Airbnb_dataset$price))

# 7.2
```

```r
#install.packages("tibble")
library(tibble)

listing_amenities = as.vector(Airbnb_dataset$amenities)
listing_amenities = strsplit(listing_amenities, ",")

num_amenities = c()
for(val in listing_amenities){
  num_amenities = c(num_amenities, length(val))
}

# Tibble used to insert column after amenities
Airbnb_dataset = add_column(Airbnb_dataset, num_amenities, .after = "amenities")


# 7.3
cancellation_types =
  as.character(as.data.frame(table(Airbnb_dataset$cancellation_policy))$Var1)


mean_rev_score_rat = c()
for(type in cancellation_types){
  type_filter = Airbnb_dataset[Airbnb_dataset$cancellation_policy == type, ]
  mean_rev_score_rat = c(mean_rev_score_rat,
                         mean(as.vector(type_filter$review_scores_rating)))
}

cancellation_type_rev = as.data.frame(
  cbind(cancellation_types, mean_rev_score_rat))
colnames(cancellation_type_rev) = c("CancelTyp", "AvgRevScore")

ggplot(data = cancellation_type_rev,
       aes(y=as.numeric(as.vector(AvgRevScore)), x=CancelTyp)) +
  geom_bar(stat = "identity", position = "dodge") +
  labs(x = "Cancellation Policy", y = "Average Review Score",
       title = "Average Review Score vs. Cancellation Policy")
```




```
# It is interesting to see that the flexible and moderate cancellation
# policies have the two highest average review score. I am guessing
# that is because the cancellation policies are the most
# lenient for customers. Hence it recieves the higher avg rev scores.

# Convert extra people and cleaning prices to numeric
Airbnb_dataset$extra_people = as.numeric(gsub('\\$|,', '',
                                              Airbnb_dataset$extra_people))

# If cleaning fee isn't present, NA will be produced, rightfully so
Airbnb_dataset$cleaning_fee = as.numeric(gsub('\\$|,', '',
                                              Airbnb_dataset$cleaning_fee))

# Convert host_response_rate percent to numeric
Airbnb_dataset$host_response_rate = as.numeric(sub("%", "",
                                              Airbnb_dataset$host_response_rate))



# QUESTION 8

# I chose number of reviews as the dependent variable.
# The number of reviews given is a more long-term metric
# as compared to the reviews per month which can vary drastically
# from month to month. Whereas number of reviews given overall is
# cummulative.
```

```r
# 10 variables potentially affecting number of reviews
#
# Host response time.  I think host response time affects
# number of reviews because hosts responded quicker might have
# more reviews because they are more popular.
#
# SuperHost. If superhost it could mean more reviews
# because more guests like their host/find listing more popular.
#
# Host identity verified. If host identity verified guests
# could feel more personable with host and write a review.
#
# Host verifications. If host has more verifications guests could
# feel more personable with host and write a review.
#
# Property types that are more popular will have more reviews.
#
# Accommodates. If a listing can accommodate more people. The listing
# could receive more reviews because more people spend time at the property.
#
# Amenities. If a listing has more amenities. More people might like the
# listing and give it a positive review.
#
# Price. If a price is high people may feel inclined to leave a review to
# describe whether they felt they got their money's worth.
#
#
# Description. Certain keywords in the description could  render a property
# more popular, hence more likely for a user to leave a review.
#
# Host response rate. If hosts follows up with a customer, the customer
# knows whether they can stay at the property and then can leave a review



plot(Airbnb_dataset$host_response_rate, Airbnb_dataset$number_of_reviews,
     xlab = "Host Response Rate", ylab = "Number of Reviews",
     main = "Number of Reviews vs. Host Response Rate")
abline(lm(Airbnb_dataset$number_of_reviews ~ Airbnb_dataset$host_response_rate))
```

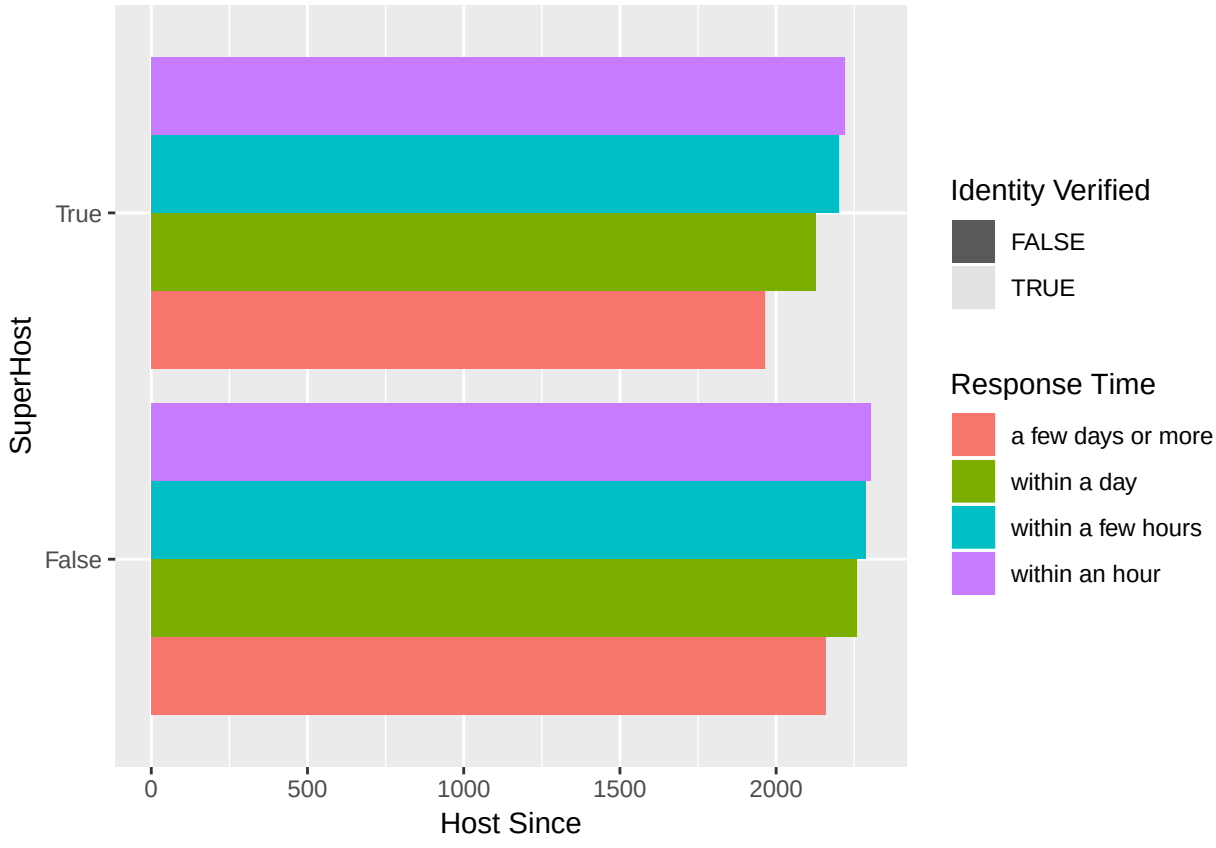



```
cor(Airbnb_dataset$number_of_reviews, Airbnb_dataset$host_response_rate)
```

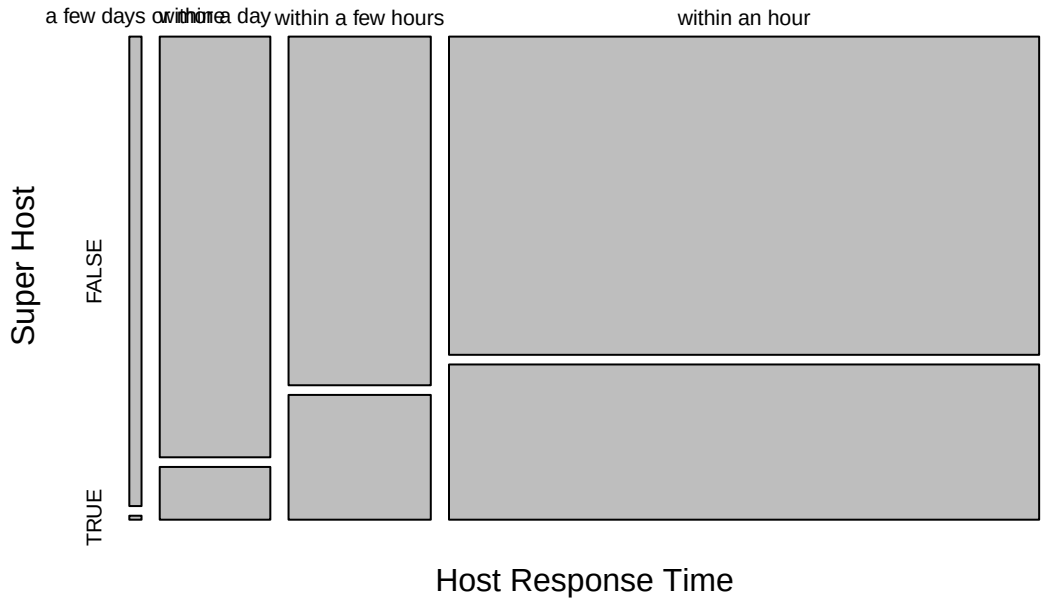
```
## [1] 0.08685835
```

```
# The plot is not statistically significant, the correlation coeffiecient
# does not differ greatly from 0 (very weak positive correlation: 0.087).
# p > 0.05.



# QUESTION 9

# 9.1
# Convert host_since to numeric and add as seperate column to dataset
host_since_dates_vec = as.vector(as.character(Airbnb_dataset$host_since))

host_since_numeric = c()
for(date in host_since_dates_vec){
  # Add 20 to year
  date =
    paste(substr(date, 1, nchar(date) - 2),
          "20", substr(date, nchar(date)-1, nchar(date)), sep = "")
  host_since_date = as.POSIXct(date, format="%m/%d/%Y")
  host_since_numeric = c(host_since_numeric,
                                as.numeric(host_since_date))
}

Airbnb_dataset = add_column(Airbnb_dataset,
                            host_since_numeric, .after = "host_since")
```

```
condense_data_set = as.data.frame(cbind(
      as.character(Airbnb_dataset$host_is_superhost),
      Airbnb_dataset$host_since_numeric,
      as.character(Airbnb_dataset$host_response_time),
      Airbnb_dataset$host_response_rate,
      Airbnb_dataset$host_verifications,
      Airbnb_dataset$host_identity_verified))


colnames(condense_data_set) = c("Superhost", "HostSince",
                                "ResTime", "ResRate", "Verifications",
                                "IdentityVer")

ggplot(condense_data_set, aes(x=condense_data_set$Superhost,
                              y=as.numeric(condense_data_set$HostSince),
                              group=condense_data_set$ResTime,
                              fill=condense_data_set$ResTime,
                              alpha = as.logical(
                                as.numeric(condense_data_set$IdentityVer)-1))) +
  geom_bar(stat="identity",position="dodge") +
  scale_x_discrete(labels = c("False", "True")) +
  scale_alpha_manual(values = c(1, 0.1)) +
  labs(x = "SuperHost", y = "Host Since",
       fill = "Response Time", alpha = "Identity Verified") +
  coord_flip()
```



```r
# According to the data, out of hosts that are
# superhosts, hosts that respond in an hour
# and have been a host the longest and have their
# identity verified are the most likely to be
# a superhost. However the majority of individuals
# who aren't superhosts have had their identity verified
# have been hosts the longest, and have a quick response times.
# Yet they are still not superhosts.

# This leads me to believe whether someone is a super host
# has more to do with their with their ratings. As opposed
# to how long they've been a host or their reponse times,
# or whether their identity is verified.

# 9.2
mosaic_data = as.data.frame(cbind(as.character(Airbnb_dataset$host_response_time),
      as.logical(as.numeric(Airbnb_dataset$host_is_superhost) - 1)))
colnames(mosaic_data) = c("HostRespTime", "SuperHost")

mosaicplot(table(mosaic_data), xlab = "Host Response Time",
           ylab = "Super Host", main = "Super Host vs. Host Response Time")
```




```r
# This mosaic plot confirms a conclusion I had
# come across in the previous question (9.1).

# Like before, we see from this mosaic plot
# that the greatest number of superhosts out of all the
# host reponse time categories are the ones that respond within an hour.
# However we also see that the greatest number of non-superhosts
# out of all the host reponse time categories
# are the ones that respond within an hour.  Percentage
# wise the greatest % of superhosts respond with an hour
```

```
# (48.7%). Second most are those who reponse within a few hours,
# then within a day, and lastly within a few days. s


# Question 10 (1)
# install.packages("stringr")

# 10.1
library(stringr)

stop_words =
  c("a",          "able",     "about",    "across", "after",
    "all",      "almost",   "also",     "am",      "among",
    "an" ,      "and"   ,   "any"   ,   "are"  ,  "as"  ,
    "at" ,      "be"  ,     "because" , "been" ,    "but"  ,
    "by"  ,     "can"  ,    "cannot" ,  "could" ,   "dear" ,
    "did" ,     "do"  ,     "does",     "either" ,  "else"   ,
    "ever",     "every" ,   "for"  ,    "from"  ,   "get"   ,
    "got",    "had",      "has" ,     "have",     "he" ,
    "her" ,     "hers",     "him" ,     "his" ,     "how"  ,
    "however",  "i"     ,   "if",     "in"   ,    "into"  ,
    "is"   ,    "it" ,      "its" ,     "just" ,    "least" ,
    "let"  ,    "like" ,    "likely",   "may"  ,    "me"      ,
    "might",  "most" ,    "must"  ,   "my"   ,    "neither" ,
    "no"   ,  "nor"  ,    "not"   ,   "of"   ,    "off"    ,
    "often",    "on"   ,    "only"  ,   "or"  ,   "other"  ,
    "our" ,     "own"  ,    "rather",   "said"  ,   "say"    ,
    "says" ,    "she"   ,   "should",   "since"  ,  "so"     ,
    "some" ,  "than"  ,   "that"  ,   "the"   ,   "their"  ,
    "them" ,    "then" ,    "there" ,   "these" ,   "they"   ,
    "this" ,    "is" ,    "to"    ,   "too"   ,   "was"    ,
    "us"   ,    "wants" ,   "was"  ,    "we"   ,    "were"  ,
    "what" ,    "when" ,    "where" ,   "which" ,   "while" ,
    "who"  ,    "whom" ,    "why"  ,    "will"  ,   "with"  ,
    "would" ,   "yet",      "you"  ,    "your")


removeStopWords = function(str, stop_words){
  str = tolower(str)
  str = gsub('[[:punct:]]', '', str)
  extract_words = unlist(strsplit(str, " "))
  return (paste(extract_words[!extract_words %in% stop_words], collapse = " "))
}


# Remove stop words in airbnb descriptions
removed_stop_words_desc = c()
for(desc in as.vector(Airbnb_dataset$description)){
  removed_stop_words_desc = c(removed_stop_words_desc,
                          removeStopWords(desc, stop_words))
}
```

```
Airbnb_dataset = add_column(Airbnb_dataset, removed_stop_words_desc,
                            .after= "description")

# Combine strings from removed_stop_words_desc vector into one string
combined_removed_words = paste(removed_stop_words_desc, sep="", collapse="")

# Remove punctuation from combined_removed_words
combined_removed_words = gsub('[[:punct:]]', '', combined_removed_words)
combined_removed_words = tolower(combined_removed_words)

# Top 10 words found in descriptions
desc_words_freq = sort(table(strsplit(combined_removed_words, " ")),
                       decreasing=TRUE)
top_desc_words = head(desc_words_freq, n = 10)
print(top_desc_words)
```

```
##
##           apartment      walk   bedroom    sydney      room   kitchen
##     45548     10742      7934      7487      7370      7197      6872
##     beach       bed     house
##      6629      6607      5394
```

```
# 10.2
beaches_data_frame =
  as.data.frame( Airbnb_dataset[
    (str_detect(Airbnb_dataset$removed_stop_words_desc, "beach") |
     str_detect(Airbnb_dataset$removed_stop_words_desc, "beaches")), ] )

# Beach keyword average price
avg_beach_price =
  sum(as.vector(beaches_data_frame$price))/nrow(beaches_data_frame)


no_beach_data_frame =
  as.data.frame( Airbnb_dataset[!(str_detect(
    Airbnb_dataset$removed_stop_words_desc, "beach") |
    str_detect(Airbnb_dataset$removed_stop_words_desc, "beaches")), ] )

# No beach keyword average price
avg_price_no_beach =
  sum(as.vector(no_beach_data_frame$price))/nrow(no_beach_data_frame)

# Round prices
avg_beach_price = round(avg_beach_price, 2)
avg_price_no_beach = round(avg_price_no_beach, 2)

print(paste("Average Beach Keyword Price: ", "$",
            avg_beach_price, sep = ""))
```

```
## [1] "Average Beach Keyword Price: $239.33"
```

```
print(paste("Average Non-beach Keyword Price: ", "$",
            avg_price_no_beach, sep = ""))
```

```
## [1] "Average Non-beach Keyword Price: $178.23"
```

```r
# As we can see a description with keyword beach or "beaches" has an
# average price that is $61.10 more.

word_frequency = function(str, search_word){
  str = tolower(str)
  str = gsub('[[:punct:]]', '', str)
  table_string_freq = as.data.frame(table(strsplit(str, " ")))
  return(table_string_freq[table_string_freq$Var1 == search_word, ]$Freq)
}

# Example. Should print 3.
word_frequency("small, SMALL, cute two bedroom! !SMaLL!", "small")
```

```
## [1] 3
```

```r
# 10.3

# Function for calculating average price with and without
# keyword in Airbnb dataset
avg_price_keyword = function(df, keyword){
  keyword = tolower(keyword)
  keyword = gsub('[[:punct:]]', '', keyword)
  word_data_frame =
    as.data.frame( df[str_detect(df$removed_stop_words_desc, keyword), ] )
  no_word_data_frame =
    as.data.frame( df[!str_detect(df$removed_stop_words_desc, keyword), ] )

  avg_word_price = sum(as.vector(word_data_frame$price))/
    length(word_data_frame$price)

  avg_no_word_price = sum(as.vector(no_word_data_frame$price))/
    length(no_word_data_frame$price)

  avg_word_price = paste("$", round(avg_word_price, 2), sep = "")
  avg_no_word_price = paste("$", round(avg_no_word_price, 2), sep = "")

  result_df = as.data.frame(cbind(avg_word_price, avg_no_word_price))
  colnames(result_df) = c("With Keyword", "Without Keyword")
  return (result_df)
}

# I predict descriptions with the word "penthouse" will have higher prices.
print("Keyword: penthouse")
```

```
## [1] "Keyword: penthouse"
```

```r
print(avg_price_keyword(Airbnb_dataset, "penthouse"))
```

```
##   With Keyword Without Keyword
## 1      $297.94         $202.65
```

```r
# That is confirmed true based on printed results.

# I predict descriptions with the word "bondi" will have higher prices
# because that area is by the beach and could be more expensive to live near.
print("Keyword: bondi")
```

```
## [1] "Keyword: bondi"
```

```
print(avg_price_keyword(Airbnb_dataset, "bondi"))
```

```
##   With Keyword Without Keyword
## 1      $209.16         $203.09
```

```
# That is confirmed true based on printed results.

# I predict descriptions with the word "AC" will have higher prices
# because an AC is an important, Sydney can get hot.
print("Keyword: AC")
```

```
## [1] "Keyword: AC"
```

```
print(avg_price_keyword(Airbnb_dataset, "AC"))
```

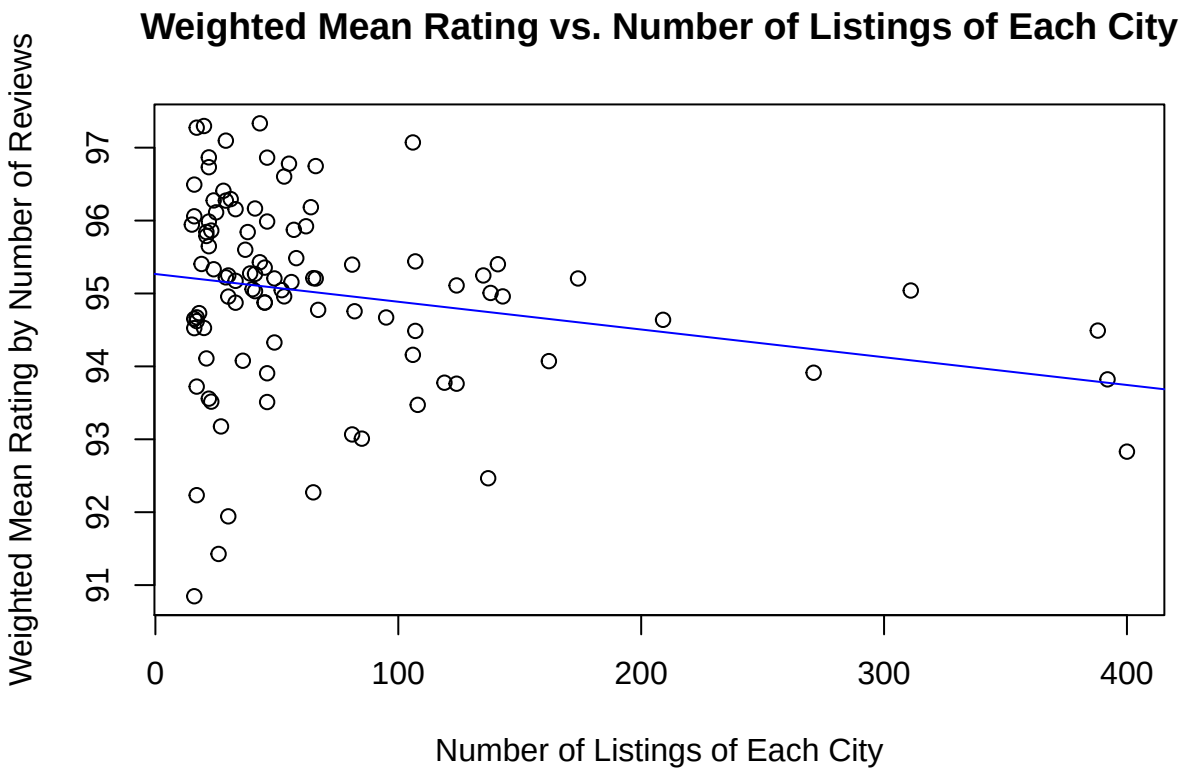
```
##   With Keyword Without Keyword
## 1         $205         $178.38
```

```
# That is confirmed true based on printed results.


# Question 10 (2)

# 10.2.1
# I will explore listings by city. Because I think most people
# looking at my data won't be familiar with all the
# zipcodes for a region, cities would be easier identifiable places
# for them as opposed to zipcodes.

cities = as.character(Airbnb_dataset$city)
top_cities = sort(table(cities),decreasing=TRUE)

# Number of listings for each city. Ordered decreasing.
# print(top_cities)

# Top 100 city listings
top_100_cities = head(top_cities, n = 100)
print(top_100_cities)
```

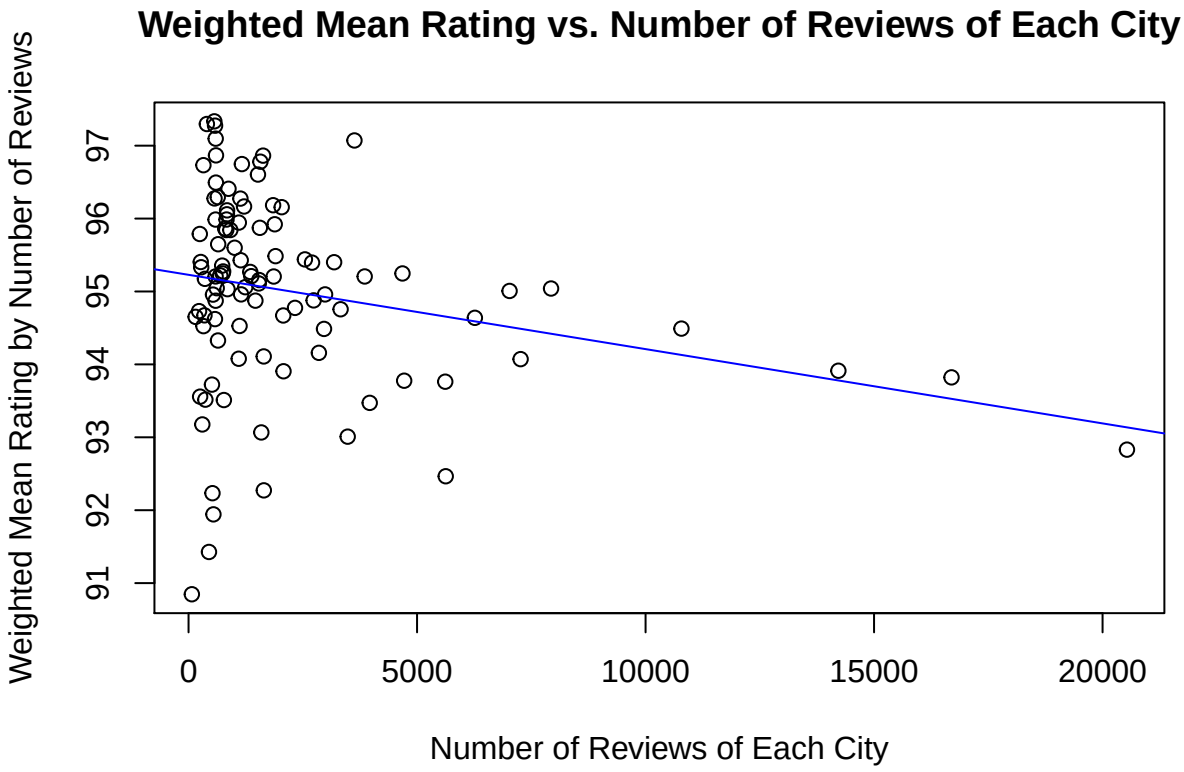
```
## cities
##              Sydney         Surry Hills         Bondi Beach
##                 400                 392                 388
##               Manly        Darlinghurst             Coogee
##                 311                 271                 209
##            Randwick             Redfern              Bondi
##                 174                 162                 143
##              Mosman             Newtown         Chippendale
##                 141                 138                 137
##          Paddington         North Bondi             Pyrmont
##                 135                 124                 124
##         Potts Point              Ultimo              Bronte
##                 119                 108                 107
##            Waterloo        Avalon Beach      Bondi Junction
##                 107                 106                 106
##            Maroubra           Haymarket Sydney Olympic Park
```

```
##                 95                 85                 82
##             Mascot            Zetland              Glebe
##                 81                 81                 67
##       Elizabeth Bay          Fairlight          Camperdown
##                 66                 66                 65
##        Marrickville         Alexandria            Balmain
##                 65                 64                 62
##          Leichhardt          Annandale          Tamarama
##                 58                 57                 56
##          Palm Beach        Erskineville          Rosebery
##                 55                 53                 53
##          Freshwater           Clovelly          Rose Bay
##                 52                 49                 49
##           Chatswood         Neutral Bay            Newport
##                 46                 46                 46
##     Rushcutters Bay       Bellevue Hill       Forest Lodge
##                 46                 45                 45
##       Woolloomooloo          Balgowlah          Cronulla
##                 45                 43                 43
##            Cremorne            Dee Why            Rozelle
##                 41                 41                 41
##         Wolli Creek           Ashfield           Waverley
##                 40                 39                 38
##           Woollahra       North Sydney          Arncliffe
##                 37                 36                 33
##           Kingsford          Vaucluse         Queenscliff
##                 33                 33                 31
##          Birchgrove            Rhodes         Strathfield
##                 30                 30                 30
##        Dulwich Hill          Mona Vale          Parramatta
##                 29                 29                 29
##            Stanmore         Crows Nest             Burwood
##                 28                 27                 26
##       Darling Point          Cammeray          Double Bay
##                 25                 24                 24
##   Brighton-Le-Sands          Narrabeen            Bundeena
##                 23                 23                 22
##              Enmore         Manly Vale       Millers Point
##                 22                 22                 22
##                Ryde         Darlington          Drummoyne
##                 22                 21                 21
##           Lilyfield         Kirribilli     Wollstonecraft
##                 21                 20                 20
##     North Balgowlah    Wentworth Point          Edgecliff
##                 19                 18                 17
##           Greenwich           Lewisham          Petersham
##                 17                 17                 17
##           Roseville          Glenfield          Kensington
##                 17                 16                 16
##           Lane Cove    North Narrabeen          Sans Souci
##                 16                 16                 16
##              Avalon
##                 15
```

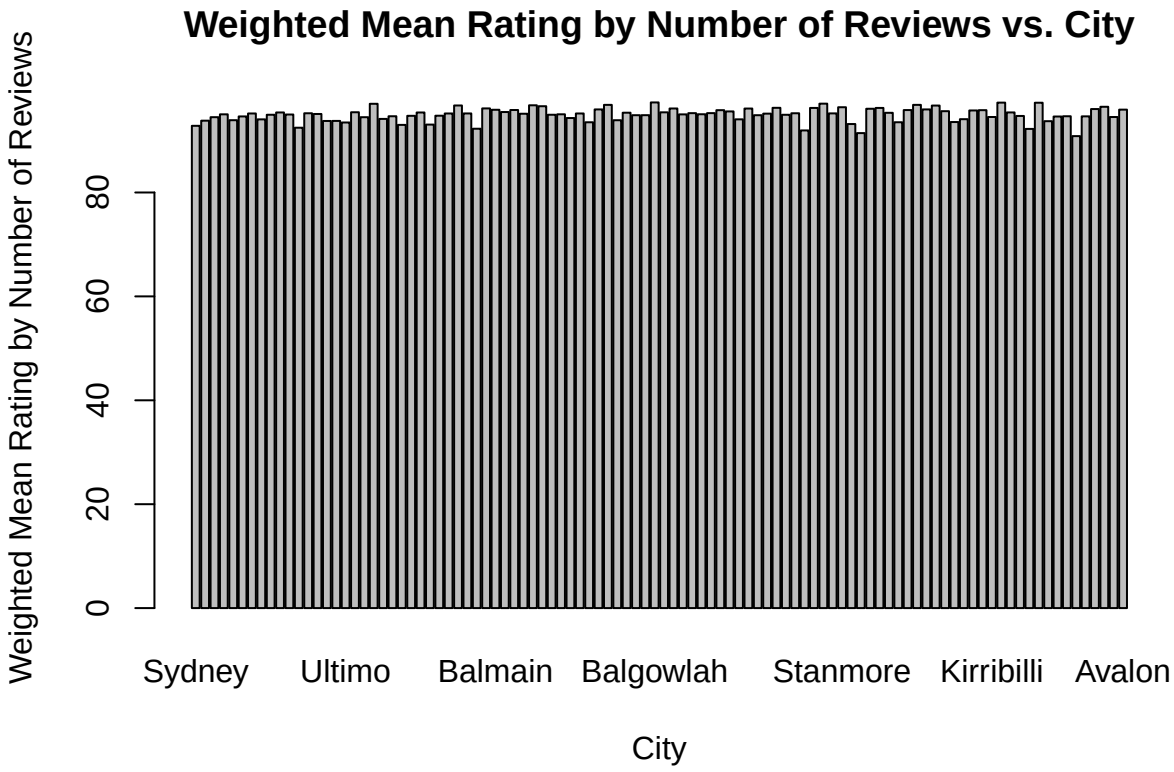
```r
# Bottom city listings
# bottom_cities = tail(top_cities, n = length(top_cities) - 100)

top_100_cities_vec = as.vector(as.data.frame(top_100_cities)$cities)

weighted_rating_cities = c()
num_reviews = c()
for(city in top_100_cities_vec){
  city_filter =
    Airbnb_dataset[Airbnb_dataset$city == city, ]

  # Calculate weighted mean rating of each city
  weighted_rating_cities = c(weighted_rating_cities,
                       sum(city_filter$review_scores_rating *
                             city_filter$number_of_reviews/
                             sum(city_filter$number_of_reviews)))
  num_reviews = c(num_reviews, sum(city_filter$number_of_reviews))
}


top_city_rating_df = as.data.frame(
  cbind(as.data.frame(top_100_cities),
        num_reviews, as.numeric(weighted_rating_cities)))
colnames(top_city_rating_df) = c("Cities", "NumListings",
                                 "NumReviews", "WeightedRating")


# Is there correlation between a cities number of listing occurrences and its
# weighted rating?
plot(top_city_rating_df$NumListings, top_city_rating_df$WeightedRating,
     main = "Weighted Mean Rating vs. Number of Listings of Each City",
     xlab = "Number of Listings of Each City", ylab =
       "Weighted Mean Rating by Number of Reviews")
abline(lm(top_city_rating_df$WeightedRating ~ top_city_rating_df$NumListings),
       col = "blue")
```




```
# There is weak negative correlation between the number of weighted
# mean rating and the number of listing occurences.


# Is there correlation between a cities number of reviews given and its
# weighted rating?
plot(top_city_rating_df$NumReviews, top_city_rating_df$WeightedRating,
     main = "Weighted Mean Rating vs. Number of Reviews of Each City",
     xlab = "Number of Reviews of Each City",
     ylab = "Weighted Mean Rating by Number of Reviews")
abline(lm(top_city_rating_df$WeightedRating ~ top_city_rating_df$NumReviews),
       col = "blue")
```




```
# There is weak negative correlation between the number of weighted
# mean rating and the number of reviews given.



barplot(as.numeric(as.vector(top_city_rating_df$WeightedRating)),
        names.arg=as.vector(top_city_rating_df$Cities) ,
        main = "Weighted Mean Rating by Number of Reviews vs. City",
        xlab = "City", ylab = "Weighted Mean Rating by Number of Reviews")
```




```
range(top_city_rating_df$WeightedRating)
```

```
## [1] 90.84722 97.33333
```

```
# I can see from the plot that all top 100 cities have weighted mean
# ratings within 6.49% of each other. I conclude this means
# that the top city listings also have the topmost ratings. It
# is clear that since the range is small the top cities have ratings close
# to each other. All of them over 90. I explored other correlations above.


# 10.2.2
avg_rating_keyword = function(df, keyword){
  keyword = tolower(keyword)
  keyword = gsub('[[:punct:]]', '', keyword)
  word_data_frame =
    as.data.frame( df[str_detect(df$removed_stop_words_desc, keyword), ] )
  no_word_data_frame =
    as.data.frame( df[!str_detect(df$removed_stop_words_desc, keyword), ] )

  avg_word_rating = sum(as.vector(word_data_frame$review_scores_rating))/
    length(word_data_frame$review_scores_rating)

  avg_no_word_rating = sum(as.vector(no_word_data_frame$review_scores_rating))/
    length(no_word_data_frame$review_scores_rating)

  avg_word_rating = round(avg_word_rating, 2)
  avg_no_word_rating = round(avg_no_word_rating, 2)

  result_df = as.data.frame(cbind(avg_word_rating, avg_no_word_rating))
```

```
  colnames(result_df) = c("With Keyword", "Without Keyword")
  return (result_df)
}


# I predict descriptions with the word "beach" will have higher ratings.
print("Keyword: beach")
```

```
## [1] "Keyword: beach"
```

```
print(avg_rating_keyword(Airbnb_dataset, "beach"))
```

```
##   With Keyword Without Keyword
## 1        94.75           93.83
```

```
# Cool! That is confirmed true based on printed results.

# I predict descriptions with the word "comfortable" will have higher ratings.
print("Keyword: comfortable")
```

```
## [1] "Keyword: comfortable"
```

```
print(avg_rating_keyword(Airbnb_dataset, "comfortable"))
```

```
##   With Keyword Without Keyword
## 1        94.31           94.19
```

```
# Descriptions with keyword comfortable are slightly higher.
# So it is confirmed true.



#  Part IV: Your turn


# I predict descriptions with the word "backpackers" will have lower ratings.
print("Keyword: backpackers")
```

```
## [1] "Keyword: backpackers"
```

```
print(avg_rating_keyword(Airbnb_dataset, "backpackers"))
```

```
##   With Keyword Without Keyword
## 1        81.18           94.28
```

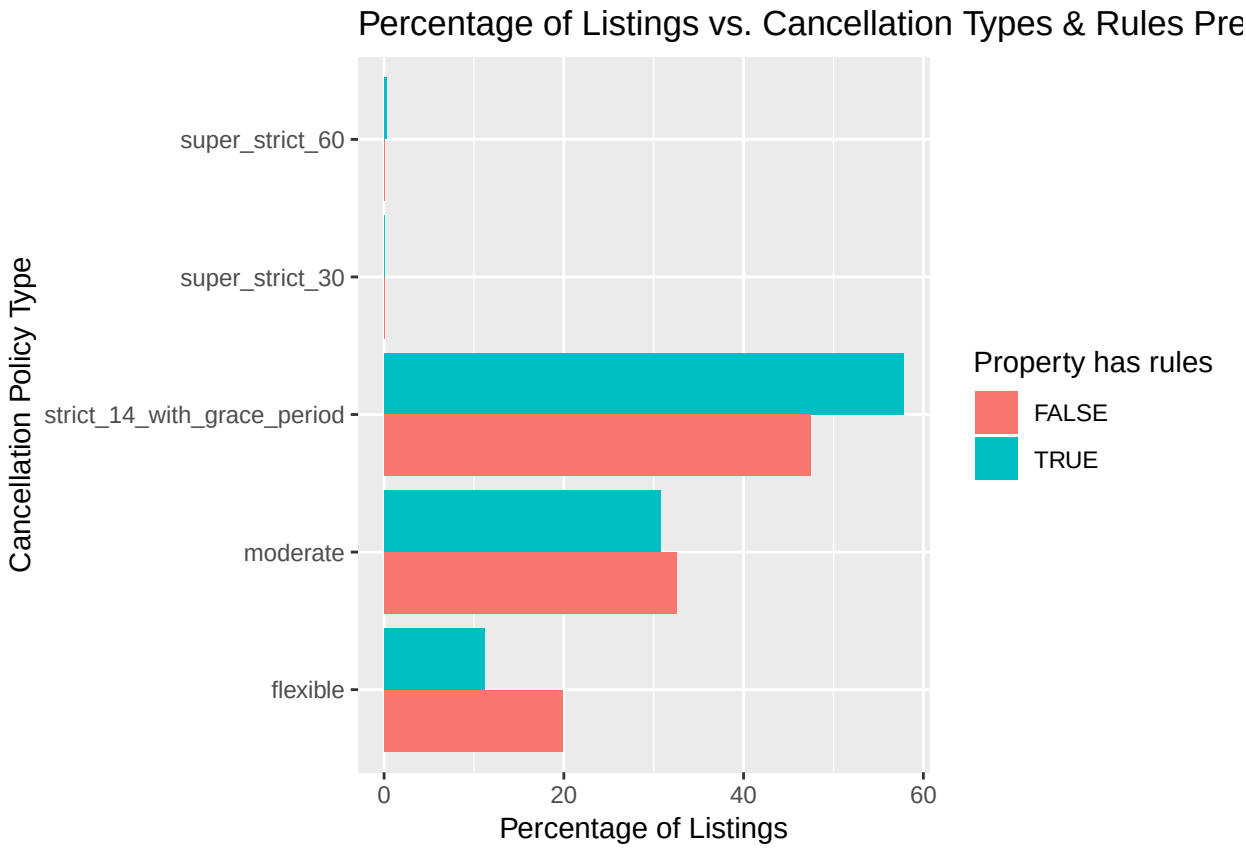
```
# Descriptions with keyword backpackers are significantly lower.
# So it is confirmed true.

# Does strict rules affect user reviews?
rules_df = Airbnb_dataset[Airbnb_dataset$house_rules != "", ]
no_rules_df = Airbnb_dataset[Airbnb_dataset$house_rules == "", ]

print(mean(rules_df$review_scores_rating))
```

```
## [1] 94.11433
```

```
print(mean(no_rules_df$review_scores_rating))
```

```
## [1] 94.72279
```

```r
# Hosts with no rules have slightly higher review scores

# I hypothesize hosts with rules have stricter cancellation policies.

perc_cancel_rule =  (as.data.frame(table(rules_df$cancellation_policy))$Freq
                     / nrow(rules_df)) * 100
perc_cancel_no_rule = (as.data.frame(table(no_rules_df$cancellation_policy))$Freq
                       / nrow(no_rules_df)) * 100

has_rule = c(rep(T, 5), rep(F, 5))
cancel_policy_typ = c(as.character(as.data.frame(
  table(rules_df$cancellation_policy))$Var1))
cancel_policy_typ = c(cancel_policy_typ, cancel_policy_typ)

percent_cancels = c(perc_cancel_rule, perc_cancel_no_rule)

data_cancel_rule = as.data.frame(cbind(cancel_policy_typ, has_rule, percent_cancels))
colnames(data_cancel_rule) = c("CancelPolicy", "HasRule", "PercentCancels")


ggplot(data_cancel_rule, aes(x=data_cancel_rule$CancelPolicy,
                             y=as.numeric(
                               as.vector(data_cancel_rule$PercentCancels)),
                             group=data_cancel_rule$HasRule,
                             fill=data_cancel_rule$HasRule
                             )) +
  geom_bar(stat="identity",position="dodge") +
  labs(title = "Percentage of Listings vs. Cancellation Types & Rules Present" ,
       x = "Cancellation Policy Type", y = "Percentage of Listings",
       fill = "Property has rules") +
  coord_flip()
```




```
# My hypothesis is correct!
# Properties with rules listed have the highest percentage of super_strict_60,
# super_strict_30
# and strict_14_with_grace_period cancellation policies. In addition they also have
# a lower percentage of moderate and flexible cancellation policies as compared
# to the properties listed without rules.


# Part V: Conclusion

# Here are some concluding takeaways:

# The correlation coefficient between number of amenities a host lists having and
# their average
# review is 0.7. There is strong positive correlation between the variables.
# Hosts that provide more
# amenities are more likely to be reviewed higher.
#
# Houses are the second most popular property on the market compared to Apartments
# which are first. But unlike apartments houses have a median price that is
# $45 dollars  per night higher. Airbnb can make significantly more money
# encouraging home owners to airbnb their homes.
#
# The correlation coefficient between how quickly a host responds and their
# response rate, is 0.83. This strong positive correlation indicates that hosts
# that respond the quickest are also most likely to have higher average
# response rates.
```

```
#
# When looking at all properties, properties with a greater number of reviews has
# no correlation with that respective properties average score rating.
#
# Apartments with real beds receive the most reviews per month (~15).
# This customer segment tends to be more vocal. It is likely they are a younger
# demographic because they are moving into individual smaller living spaces and
# not into larger homes like a family would.
#
# Flexible and moderate cancellation types have the highest average review scores.
# It could be because the cancellation policies are the most lenient for customers.
# Hence Airbnb should encourage hosts, to have less strict cancellation types where
# appropriate. It results in happier customers.
#
# Superhosts are most likely the ones that respond the quickest, have the
# highest response rates and most importantly receive the highest reviews.
#
# Certain keywords like "beach", "penthouse", "bondi", "AC" if included in listing
# descriptions result in higher list pricing. The underlying question is
# what keywords
# derive value? Why do hosts equate certain keywords with listing higher prices?
# This is more of a business psychological to explore.
#
# Certain keywords like "comfortable", and "backpackers" result in different average
# ratings. I would suggest Airbnb identify certain keywords in listing descriptions
# and filter the results a user sees so that they are more likely to choose a listing
# that they would rate highly. This will automatically filter out listings that a user
# would rate lower out of users sight.



# Part VI:
# Clearly the explain and describe the Lifecycle of Data Science for this project.

# We started this project by acquiring our data. In this case the data was made publicly
# available by  and gathered by Airbnb.
#
# I then cleaned the data. I found that descriptions were left blank and that certain
# host information had N/As. I decided to drop rows that has N/As present in any column.
# Although we lost this data, the data we now had did not constantly need to be check for
# NAs every step moving forward. I significantly reduced the amount of time needed to clean the
# data moving forward by simply dropping the rows.
#
# I then analyzed general quantitate measurements. I analyzed the frequency of single variables
# in the dataset. I modeled the frequencies found of single variables with pie charts, bar graphs,
# and histograms. I then analyzed the relationship found between multiple variables (e.g., host
# response time vs. host response rate, variables that qualify a host as a "superhost",  number
# of amenities vs. review score value....)
#
# Throughout the process of writing code and creating graphs, I was published new findings in
# my comments. I was sharing information collected and making business judgements throughout the
# process.
#
```

```
# To produce effective graphs I had to subset into specific portions of data I wanted to analyze
# and ignore the rest. I had to destroy data that included NAs and in some cases blank cells.
#
# Other times I had to modify existing data. For example pricing had the $ symbol, I had to curate
# data and convert it from a character to numeric type so it could be analyzed further.
# Preserving, destroying, and modifying data was essential to extracting the data needed to make
# appropriate conclusions.



# WholeTale

# WholeTale Link: https://dashboard.wholetale.org/tale/view/5cc6a1682744a50001c611b8
# If that link doesn't work: https://dashboard.wholetale.org/run/5cc6a1692744a50001c611bb

# Jupyter Link: https://tmp-w9mqmemqnu8y.wholetale.org/tree
```