Exploratory Data Analysis

Blake Pappas

9/5/2021

Packages and Libraries

When you open an R session, a few libraries (base, stats, graphics) are already loaded. There are many other libraries available that contain functions for statistical analysis and datasets. CRAN is a repository from which these packages can be installed easily.

To load a library that is already installed, use the `library()` function. You need to do this each time you begin an R session (or at the start of an RMarkdown document that requires the library).

Here's an example: the dataset called `galaxies` is contained in the MASS package. It will not load without loading the library. Run the code below to see the error message.

```
data(galaxies)
```

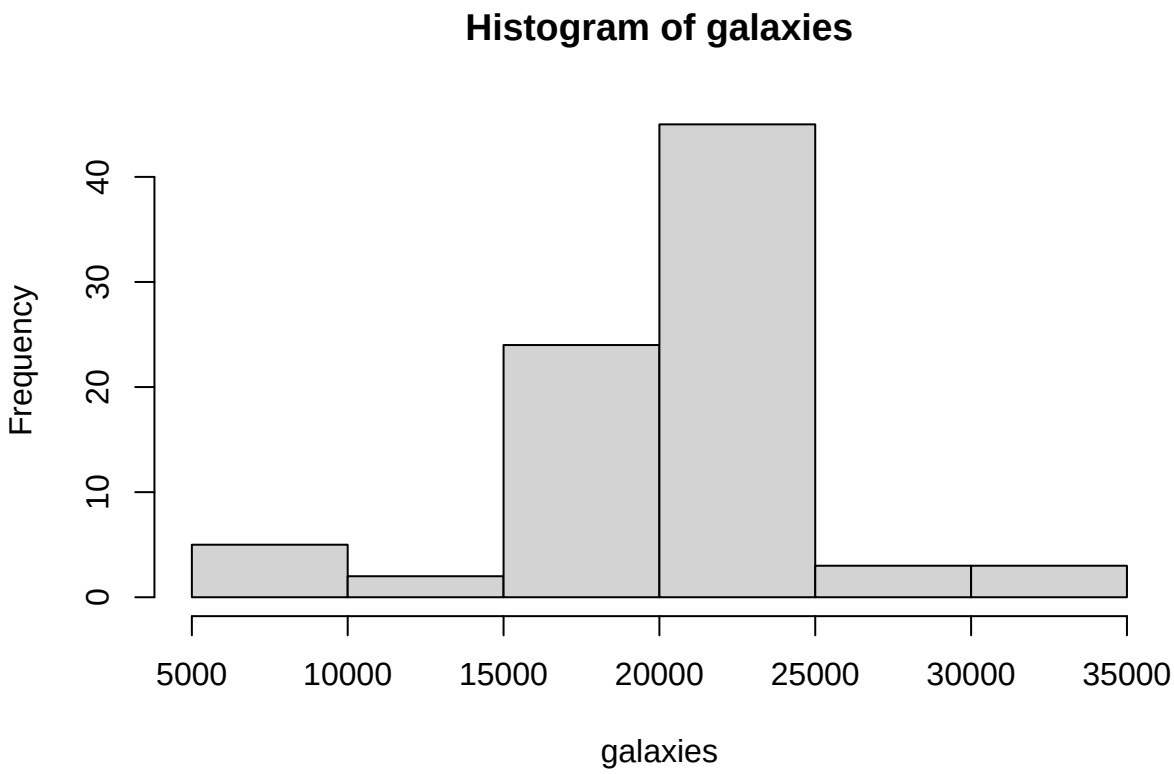
```
## Warning in data(galaxies): data set 'galaxies' not found
```

After you load the MASS library, it will work.

```
library(MASS)
data(galaxies)  # This "attaches" the data set to your workspace.
head(galaxies)
```

```
## [1]  9172  9350  9483  9558  9775 10227
```

```
hist(galaxies)  # Remember the word for the shape of this distribution?
```



MASS is one of a small number of packages that are installed when R is installed. For most other packages, first install them using the function `install.packages()`. When R asks you to select a CRAN mirror, choose one that is geographically close to you. After you install the package once, you can then use the package by loading the library with the `library()` function at the beginning of each session.

Anyone can publish an R package on CRAN. If you are using a package to perform a statistical analysis, look at the documentation and make sure the package seems to come from a reliable source.

The code below would install and load the package "abind", which is often used to store data or simulation results in multidimensional arrays. (Optional:) Try installing and loading a package of your choosing here. Some popular packages include "scales," which lets you control scale and color transparency in a plot, or "reshape2," which can turn wide data sets into tall ones.

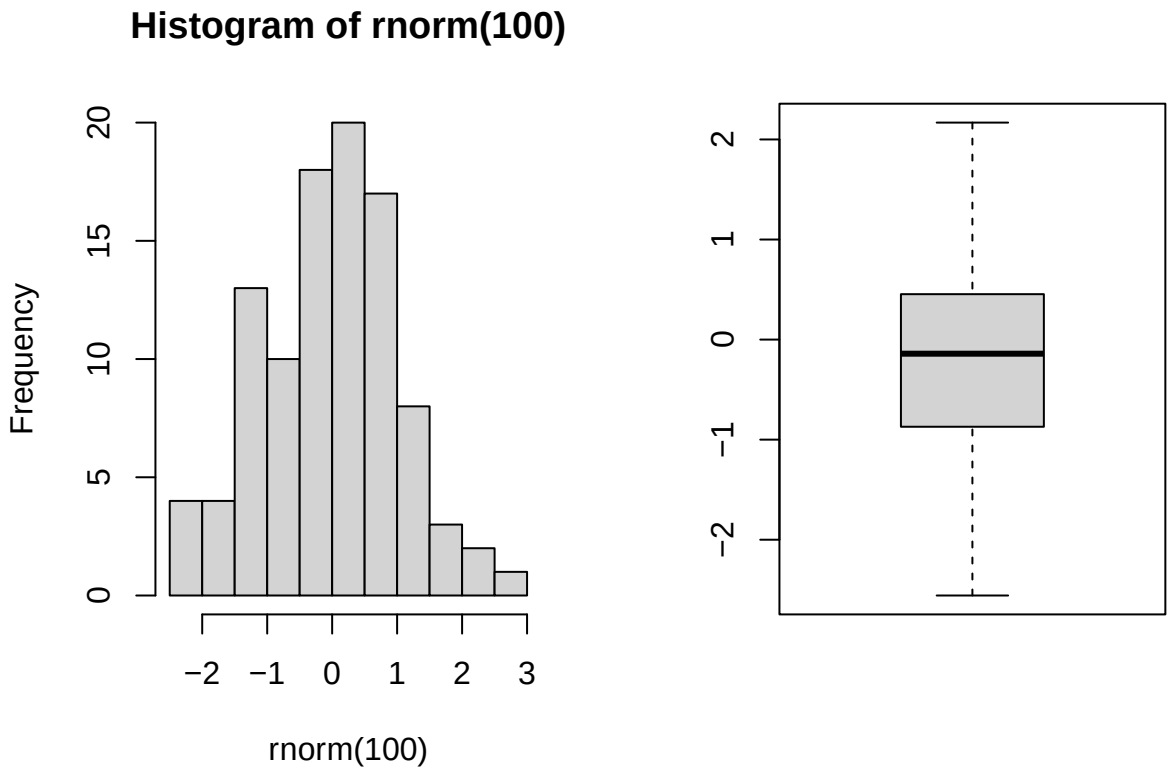
```
# Optional: replace "abind" in the code below with any package of your choosing and uncomment.

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

# install.packages("dplyr")
# library(dplyr)
```

You can also install a package with a point-and-click GUI by choosing "Tools" in the RStudio window and clicking "Install packages".

Displaying Multiple Panels in One Plot

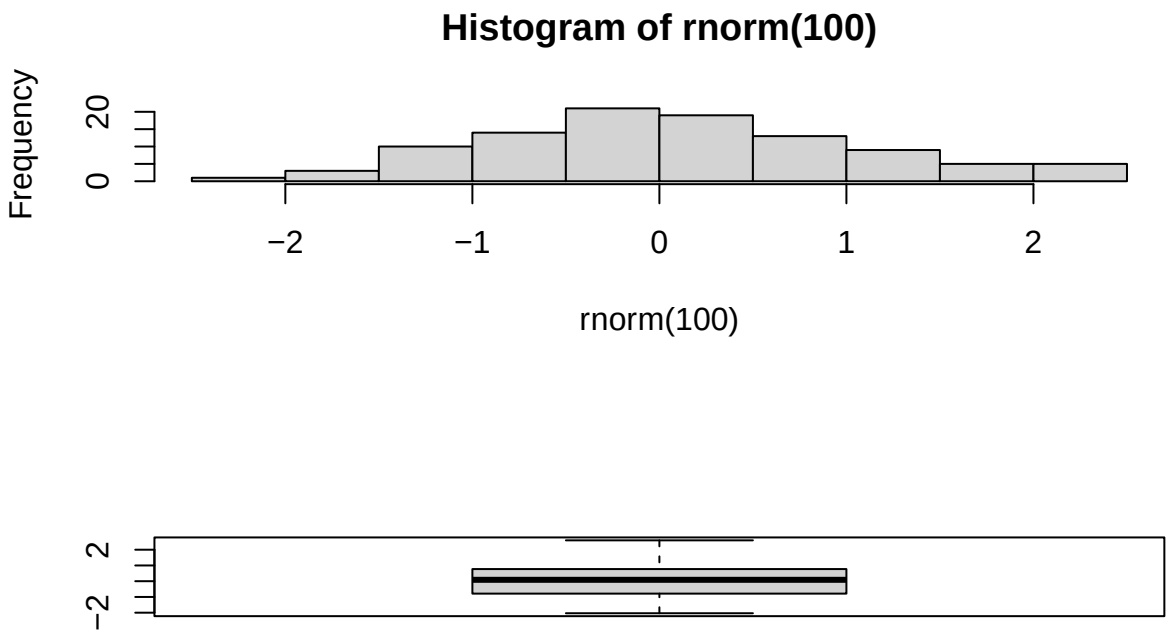
The `par` function lets you control the graphics created in R. It has many options that control the margins, scale, and titles of plots. The `mfrow` option allows you to arrange panels of plots in a grid. Set `mfrow = c(a, b)` to use a grid with a rows and b columns.

```
par(mfrow = c(1, 2))
hist(rnorm(100))
boxplot(rnorm(100))
```



The following code switches the arrangement to have two rows.

```
par(mfrow = c(2, 1))
hist(rnorm(100))
boxplot(rnorm(100))
```



Note: The methods for displaying multiple plots are quite different if you use ggplot2.

R with Missing Values

Some functions in R will not work in the presence of missing values. Most plotting functions will just ignore missing values, but many calculations will return the value NA.

```
# Create a Vector with Missing Values
x <- c(4, 18, 9, NA, 0)
```

The vector `x` has an element whose value is NA, which is a special value in R representing a missing value. If you try to use the mean function on `x`, the answer is NA. The option `na.rm = T` can be used to ignore the missing values.

```
mean(x)
```

```
## [1] NA
```

```
mean(x, na.rm = T)  # Use "na.rm = T" to ignore NAs
```

```
## [1] 7.75
```

You might get data in which the missing values are not coded as NA values. Here, `y` has a missing value but it is not treated as NA. Further, R thinks that `y` is a vector of character data because of the "no response" value.

```r
y <- c('no response', 3, 4, 1, 12)
is.na(y[1])  # Returns TRUE if y[1] is NA
```

```
## [1] FALSE
```

```r
class(y)
```

```
## [1] "character"
```

The mean of y cannot be calculated, even with `na.rm = T`, because `y` is a character vector.

```r
mean(y, na.rm = T)
```

```
## Warning in mean.default(y, na.rm = T): argument is not numeric or logical:
## returning NA
```

```
## [1] NA
```

The best way to prevent this is to ensure that R recognizes NAs when you first read in the data using the `na.strings` option in `read.csv`. Below is one quick way to fix it. The `as.numeric` function will turn `y` into a numeric vector. Any elements than cannot be turned into numeric values will become NAs. *Be careful with this approach:* it will not work as you expect on a variable whose class is `factor`, for example.

```r
y <- as.numeric(y)
```

```
## Warning: NAs introduced by coercion
```

```r
y
```

```
## [1] NA  3  4  1 12
```

```r
class(y)
```

```
## [1] "numeric"
```

Missing values are addressed differently in functions of two variables, such as the `cor()` function. The argument `use = 'complete.obs'` will omit missing values. See the R documentation for different options for the "use" argument.

```r
cor(x, y, use = 'complete.obs')
```

```
## [1] -0.9122455
```

Exercises

Exercise 1: Explore Associations in Airbnb Data

Use the Airbnb data, found in the file `Airbnb_Listings_NOLA.csv`.

The following R functions will help in answering the questions below: mean, median, quantile, boxplot, subset, table, prop.table.

a. Calculate the mean, median, and standard deviation of the variable `Price` among only listings for which `Room_Type` is `Entire home/apt`. Then calculate the mean, median, and standard deviation of the variable `Price` among only listings for which `Room_Type` is `Private room`.

```
NOLA <- read.csv("Airbnb_Listings_NOLA.csv")

Price_RT <- c(subset(NOLA, NOLA$Room_Type == 'Entire home/apt'))
tapply(Price_RT$Price, Price_RT$Room_Type, mean, na.rm = T)
```

```
## Entire home/apt
##        216.3012
```

```
tapply(Price_RT$Price, Price_RT$Room_Type, median, na.rm = T)
```

```
## Entire home/apt
##             150
```

```
tapply(Price_RT$Price, Price_RT$Room_Type, sd, na.rm = T)
```

```
## Entire home/apt
##        269.5016
```

```
Price_LT <- c(subset(NOLA, NOLA$Room_Type == 'Private room'))
tapply(Price_LT$Price, Price_LT$Room_Type, mean, na.rm = T)
```

```
## Private room
##     98.81739
```

```
tapply(Price_LT$Price, Price_LT$Room_Type, median, na.rm = T)
```

```
## Private room
##           75
```

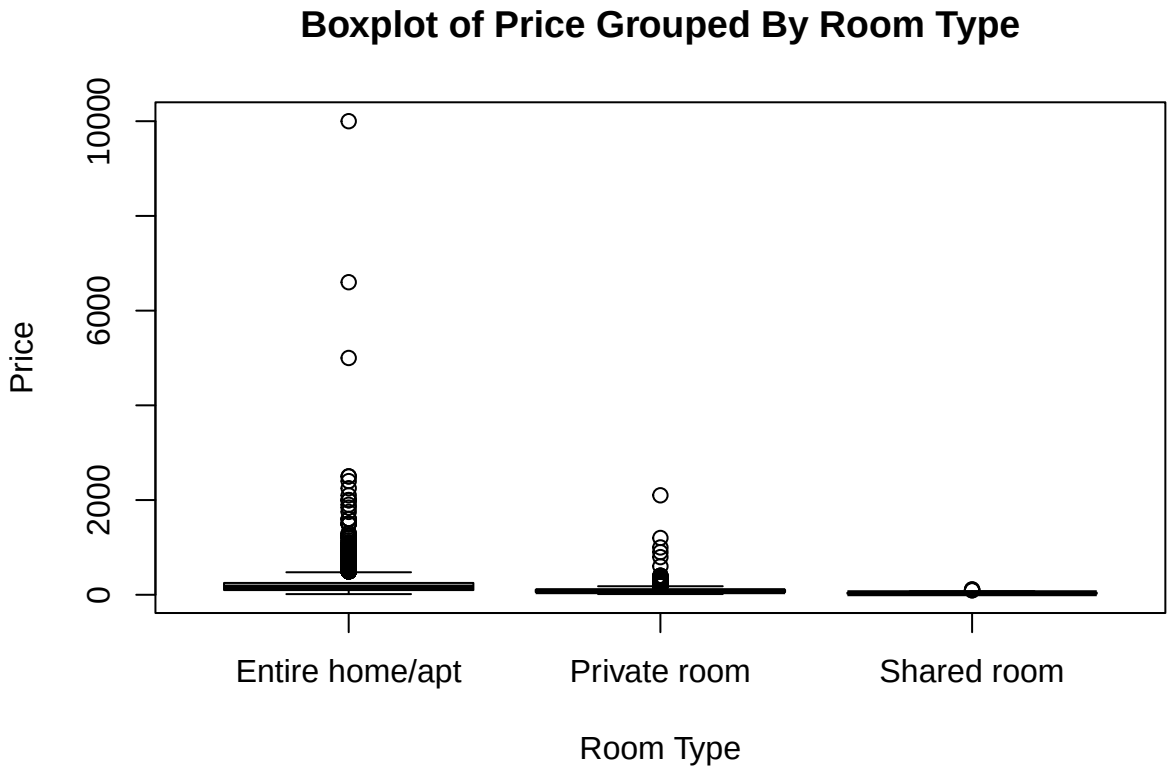
```
tapply(Price_LT$Price, Price_LT$Room_Type, sd, na.rm = T)
```

```
## Private room
##     107.2802
```

Answer: When `Room_Type` is 'Entire home/apt', the mean, median, and standard deviation of the variable `Price` are 216.3012, 150, and 269.5016, respectively. When `Room_Type` is 'Private room', the mean, median, and standard deviation of the variable `Price` are 98.81739, 75, and 107.2802, respectively. Based on these statistics, there does appear to be an association between `Room_Type` and `Price`. The larger the Airbnb, the higher the mean, median, and standard deviation tend to be.

b. Use boxplots or histograms to display the distribution of `Price` grouped by `Room_Type`. Based on the results in parts a and b, is price associated with room type?

```
# Create a Data Frame of Random Numbers
randoms <- data.frame(size = rnorm(100, 20, 3), species = as.factor(sample(1:4, 100, replace = TRUE)))

# The Syntax a ~ b Will Make Boxplots of Variable "a"
# Grouped by Variable "b".
# In the "data=", Put the Name of the Data Frame
boxplot(NOLA$Price ~ NOLA$Room_Type, data = randoms, main = "Boxplot of Price Grouped By Room Type", xla
```



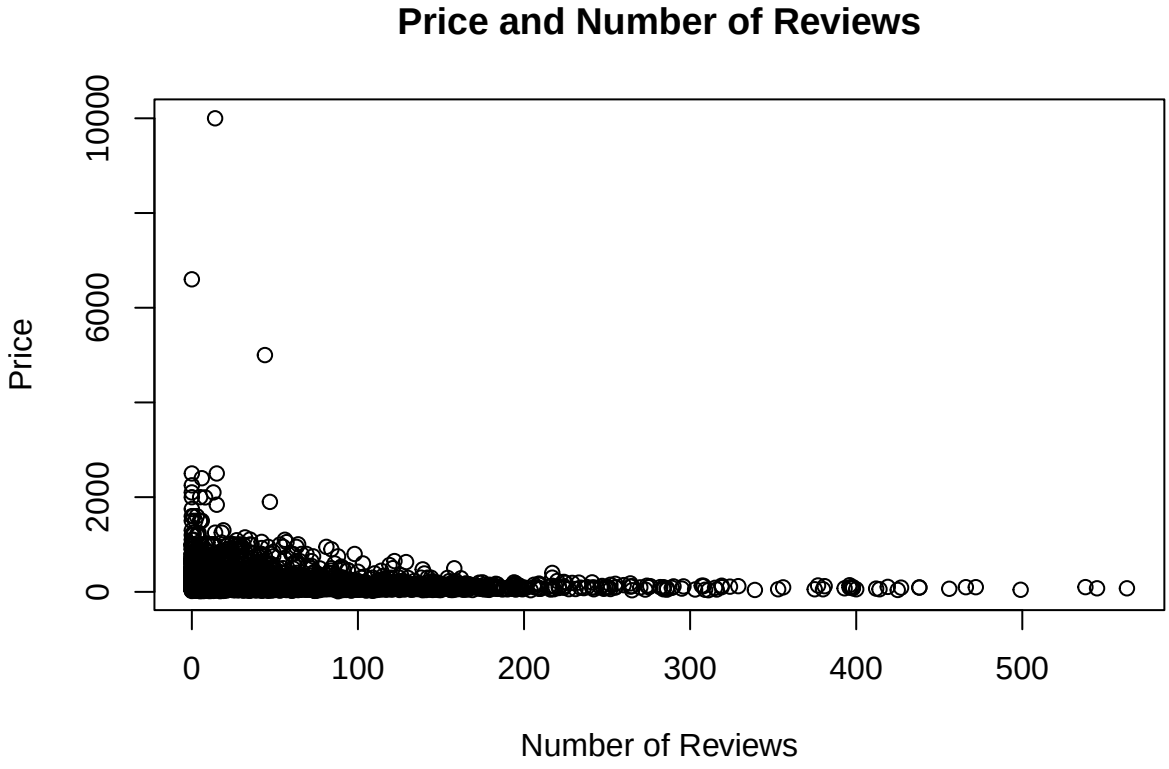
Answer: Based on the results from parts a and b, there definitely is an association between `Price` and `Room Type`. The boxplot demonstrates how the larger the room that is rented, the higher the price that one pays for that room. The results support the notion that the greater the size of the Airbnb, the higher the price that is typically charged.

b. Calculate Pearson's correlation between `Price` and `Number_of_Reviews`. Use the `cor()` function. What does this suggest about the relationship between the variables. Make a scatterplot of `Number_of_Reviews` and `Price`. Does their relationship look linear?

```r
cor(NOLA$Price, NOLA$Number_of_Reviews, use = 'complete.obs')
```

```
## [1] -0.1363863
```

```r
plot(NOLA$Number_of_Review, NOLA$Price, main = "Price and Number of Reviews", xlab = "Number of Reviews
```



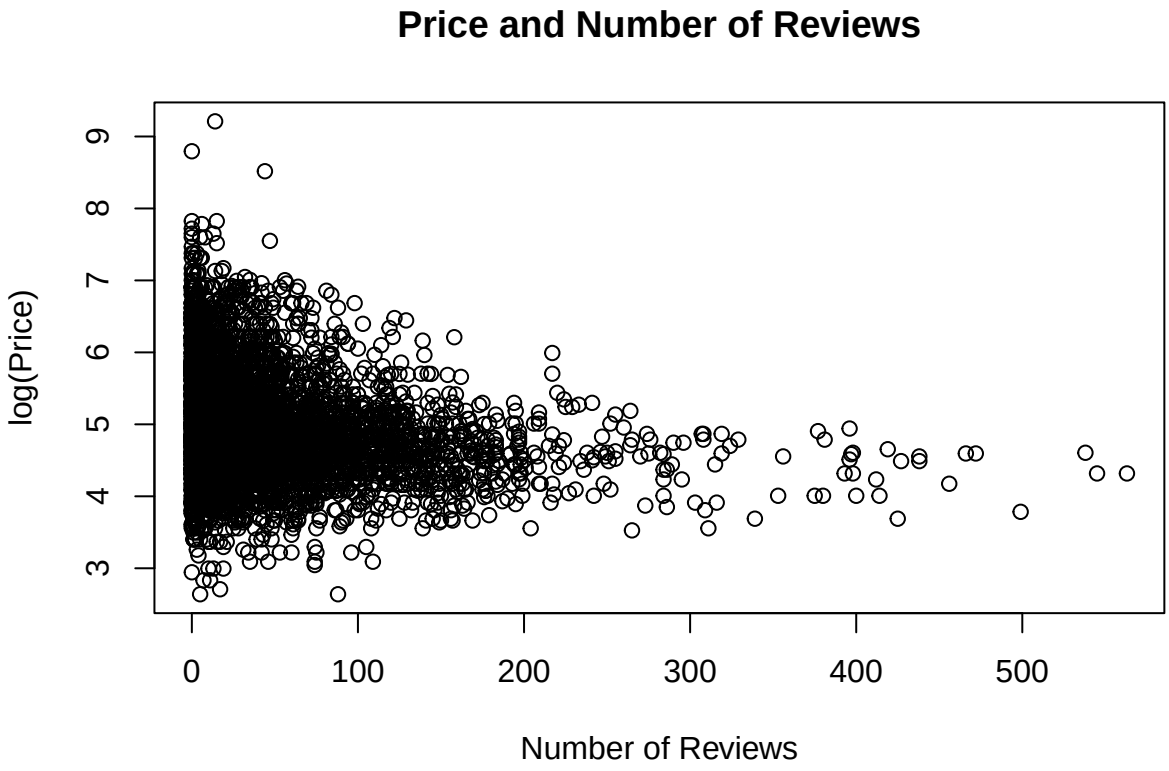
Answer: The Pearson's correlation between `Price` and `Number of Reviews` is approximately -0.1363863. This suggests there is a weak, negative relationship between `Price` and `Number of Reviews`. Looking at the scatterplot, there does not seem to be a linear relationship between the two variables. If anything, it's more curved than linear.

c. Now apply a log transformation to `Price` using the `log()` function. The log transformation is often used for highly skewed variables such as Price. Again, make a scatterplot of `Number_of_Reviews` and log(`Price`) and calculate Pearson's correlation between the two variables. What do these two calculations suggest about the association between price and the number of reviews?

```r
cor(log(NOLA$Price), NOLA$Number_of_Reviews, use = 'complete.obs')
```

```
## [1] -0.2125922
```

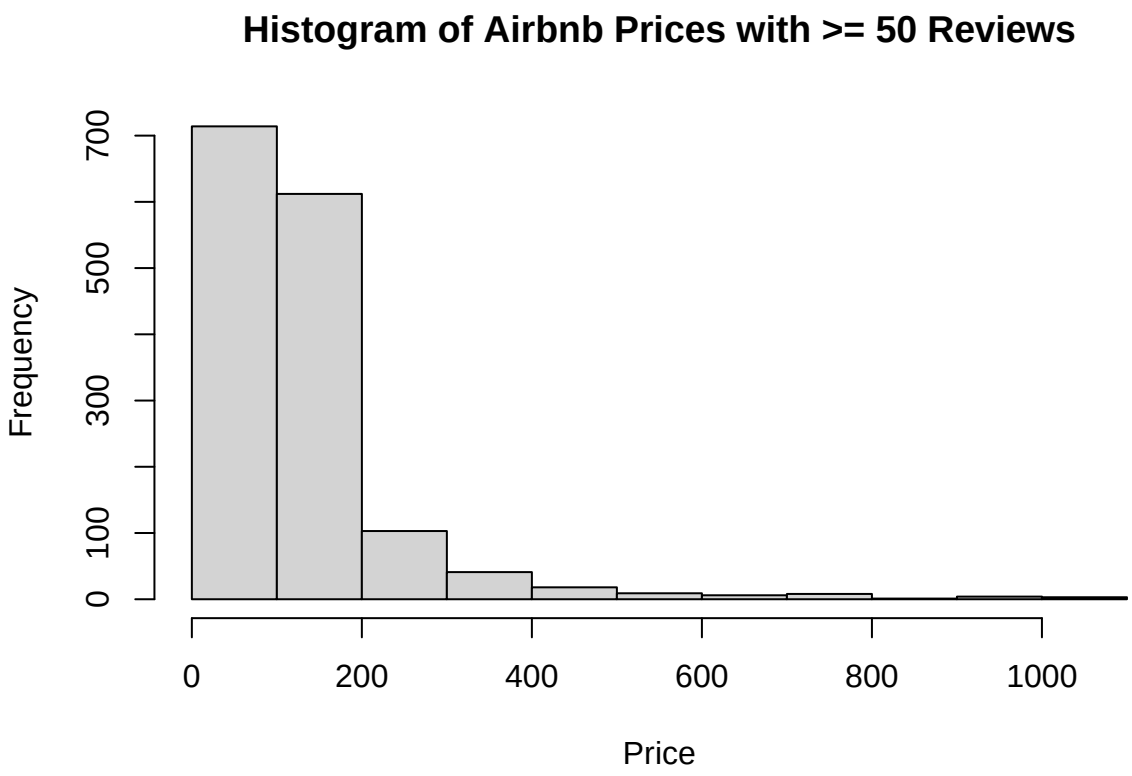
```r
plot(NOLA$Number_of_Review, log(NOLA$Price), main = "Price and Number of Reviews", xlab = "Number of Re
```



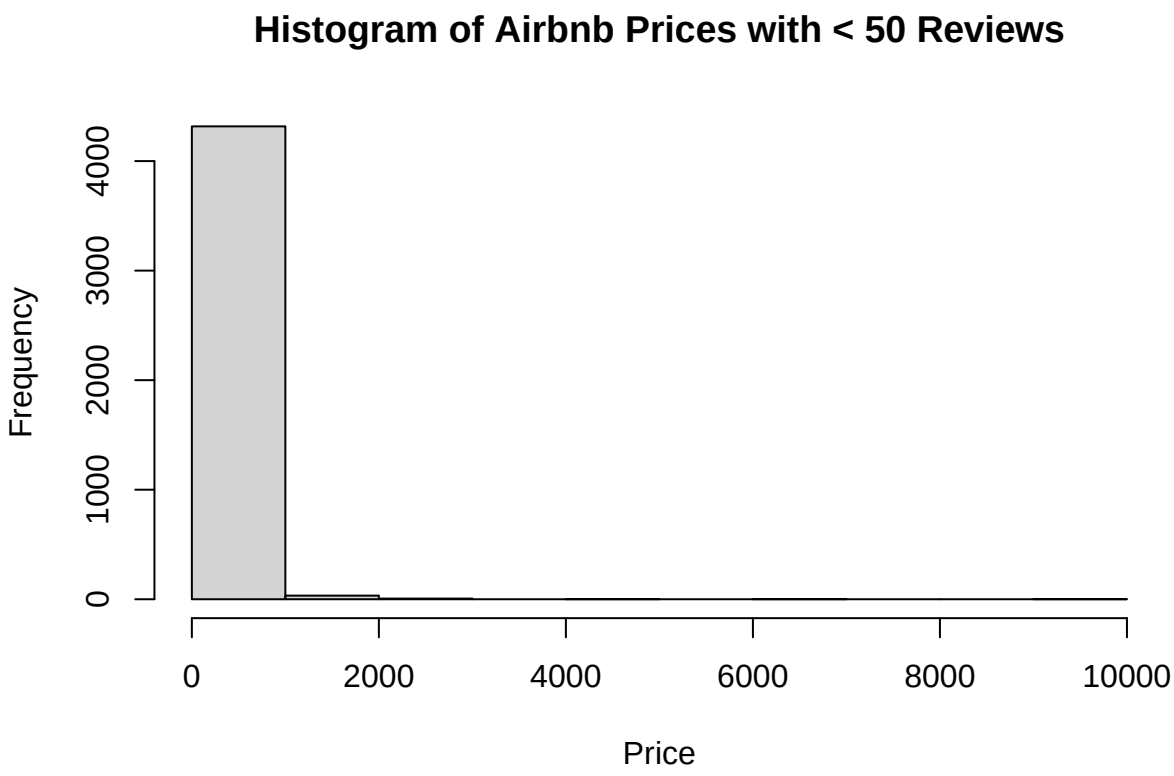
Answer: These two new calculations uphold the notion of a weak, negative relationship between `Price` and `Number of Reviews`. The scatterplot makes it even more evident how the fewer the amount of reviews that a property has, the higher the Price that tends to be charged at that property.

d. Subset the data into two groups: those listings with 50 or more reviews and those with less than 50 reviews. Use boxplots or histograms to compare the distribution of `Price` among the two subsets. What do these plots suggest about the association between price and the number of reviews?

```
more_than_50 <- subset(NOLA, NOLA$Number_of_Reviews >= 50)
less_than_50 <- subset(NOLA, NOLA$Number_of_Reviews < 50)

hist(more_than_50$Price, main = "Histogram of Airbnb Prices with >= 50 Reviews", xlab = "Price")
```




```
hist(less_than_50$Price, main = "Histogram of Airbnb Prices with < 50 Reviews", xlab = "Price")
```



Answer: These two plots suggest a negative relationship between `Price` and the `Number of Reviews`. That is, the more reviews that a property has, the lower the price (or, the less reviews a property has, the higher the price).

e. Create a new categorical variable in your data set whose value is 1 if the listing has 50+ reviews and 0 if the listing has less than 50 reviews. Call the new variable `high_listings`. Convert the `high_listings` variable to a factor using the `as.factor` function.

Then create a frequency table that shows the counts for your new variable and the `Room_Type`. Use row or column proportions to investigate whether `Room_Type` is associated with `high_listings`. Give your answer about whether the variables are associated and support it with the relevant proportions that you calculated.

```
high_listings <- as.factor(NOLA$Number_of_Reviews >= 50)

# Use the Table Function to Create a Two-Way Table
data_table <- table(NOLA$Room_Type, high_listings)
data_table # A Boolean of TRUE equates to 1, while a Boolean of FALSE equates to 0
```

```
##                  high_listings
##                   FALSE TRUE
##   Entire home/apt  3709 1211
##   Private room      622  298
##   Shared room        28   10
```

```r
# prop.table Can be Used to Find Row and Column Proportions
# The Margin Option Controls Whether You Get Row or Column Proportions
prop.table(data_table, margin = 1) # Row proportions
```

```
##                  high_listings
##                      FALSE      TRUE
##   Entire home/apt 0.7538618 0.2461382
##   Private room    0.6760870 0.3239130
##   Shared room     0.7368421 0.2631579
```

```r
prop.table(data_table, margin = 2)  # Column proportions
```

```
##                  high_listings
##                        FALSE        TRUE
##   Entire home/apt 0.850883230 0.797235023
##   Private room    0.142693278 0.196181698
##   Shared room     0.006423492 0.006583278
```

Answer: The results from the calculated frequency/proportion tables indicate there is an association between the `Room_Type` and `high_listings` variables. Across all Room Types, there is a higher proportion of properties having less than 50 reviews, compared to that of properties having 50 or more reviews (75.39% for Entire home/apt, 67.61% for Private room, and 73.68% for Shared room). Similarly, there is also (by far) a higher proportion of listings having a `Room_Type` of "Entire home/apt" with either, regardless of the `Number of Reviews` (85.09% for Entire home/apt with less than 50 reviews, 79.72% for Entire home/apt with 50 or more reviews).

Exercise 3

There is a data set called `iris` in the MASS R package. Load the MASS library and the iris data. Use `?iris` to view the documentation for the data set.

```r
library(MASS)
data(iris)
head(iris)
```

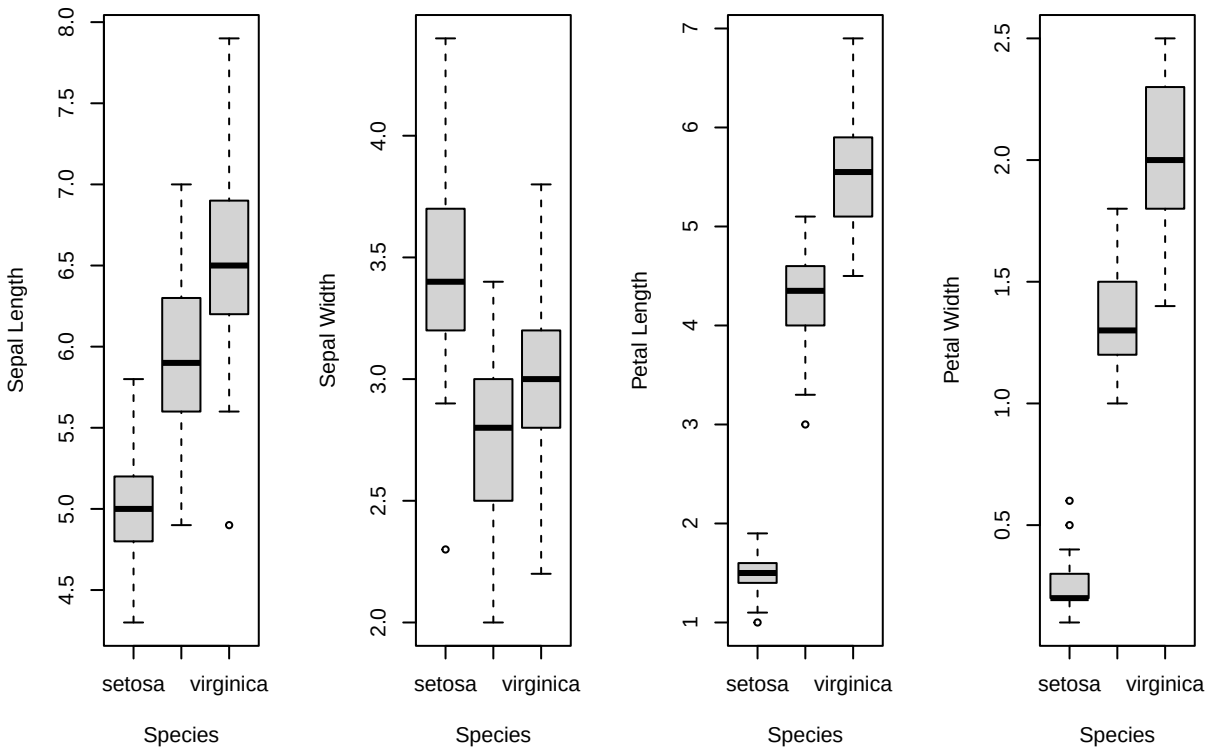
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1          5.1         3.5          1.4         0.2  setosa
## 2          4.9         3.0          1.4         0.2  setosa
## 3          4.7         3.2          1.3         0.2  setosa
## 4          4.6         3.1          1.5         0.2  setosa
## 5          5.0         3.6          1.4         0.2  setosa
## 6          5.4         3.9          1.7         0.4  setosa
```

```r
#?iris
```

a. Create four boxplots showing `Sepal.Length`, `Sepal.Width`, `Petal.Length`, `Petal.Width`grouped by `Species`. Use `par(mfrow)` to display them with a single plot with four panels.

Answer: See below for boxplots.

```r
par(mfrow = c(1, 4))
boxplot(iris$Sepal.Length ~ iris$Species, xlab = "Species", ylab = "Sepal Length")
boxplot(iris$Sepal.Width ~ iris$Species, xlab = "Species", ylab = "Sepal Width")
boxplot(iris$Petal.Length ~ iris$Species, xlab = "Species", ylab = "Petal Length")
boxplot(iris$Petal.Width ~ iris$Species, xlab = "Species", ylab = "Petal Width")
```



b. Suppose you were given these measurements of an iris at random: `Sepal.Length` = 6, `Sepal.Width` = 2.7, `Petal.Length` = 6, `Petal.Width` = 1.6. Which species do you think it would belong to? Are any of its measurements surprising?

Answer: I think this iris would likely belong to the species versicolor. Based on the iris data, the species versicolor has a median `Sepal.Length` of 5.9, a median `Sepal.Width` of 2.8, a median `Petal.Length` of 4.2, and a median `Petal.Width` of 1.3. Compared to the other two species of iris, the measurements of this randomly selected iris adhere closer with the medians, 1st quartiles, and 3rd quartiles of the for iris versicolor. The only measurement that seems surprising is the `Petal.Length` of 6, which actually falls out of the range altogether for iris veriscolor. Perhaps the `Petal.Length` for this randomly sampled iris could be an outlier.

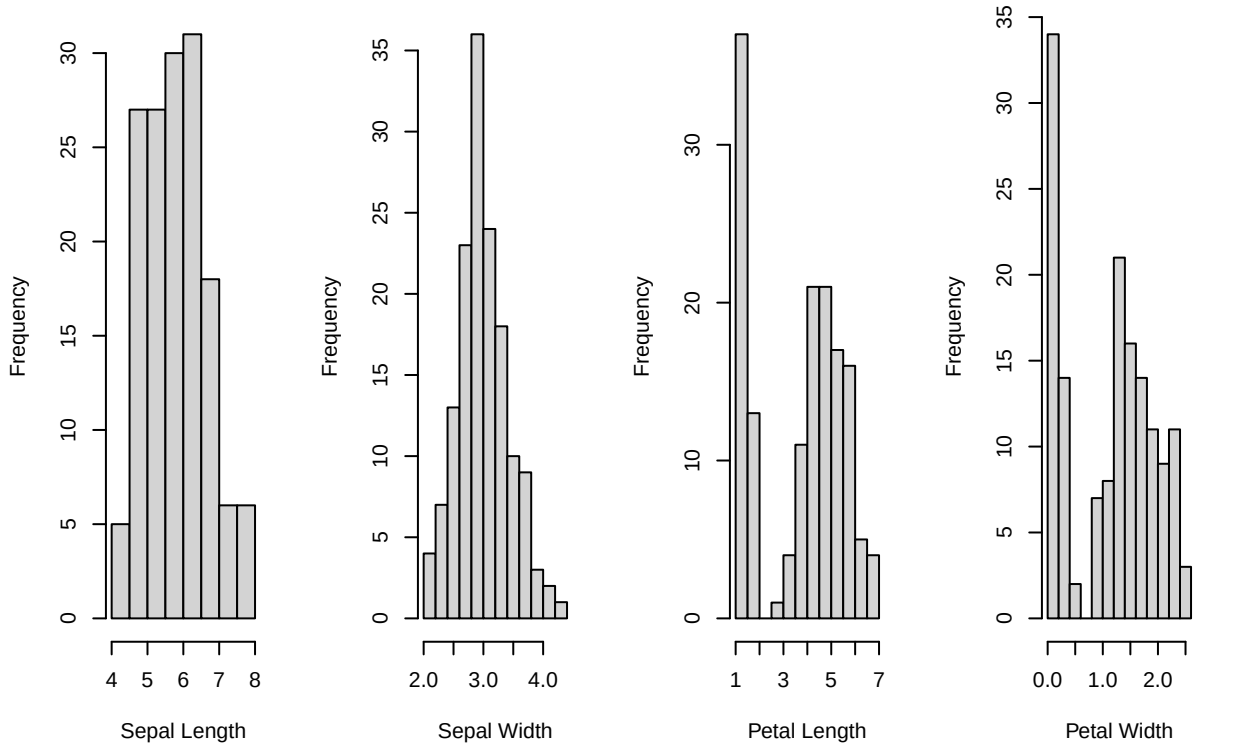
c. Use some plots and summary statistics to continue exploring the data. Propose a simple rule for classifying iris species based on their petal and sepal measurements. (Your rule might look something like this: if petal.width > 5 and $2 <$ sepal.width < 4, classify as "virginica")

Answer: See below for plots, summary statistics, and species classification rule.

```r
# Summary Statistics
library(MASS)
data(iris)
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

```r
# Plots
par(mfrow = c(1, 4))
hist(iris$Sepal.Length, main = "Histogram of Iris Sepal Lengths", xlab = "Sepal Length")
hist(iris$Sepal.Width, main = "Histogram of Iris Sepal Widths", xlab = "Sepal Width")
hist(iris$Petal.Length, main = "Histogram of Iris Petal Lenths", xlab = "Petal Length")
hist(iris$Petal.Width, main = "Histogram of Iris Petal Widths", xlab = "Petal Width")
```




```
# Rule for Classifying Iris Species (Doesn't Always Work...)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following object is masked from 'package:MASS':
##
##     select

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
x <- 1:150
case_when(
  iris$Sepal.Length < 5 & iris$Sepal.Width > 3.5 & iris$Petal.Length < 2 & iris$Petal.Width < 0.5 ~ "se
  (iris$Sepal.Length > 5 & iris$Sepal.Length < 6.5) & iris$Sepal.Width < 2.8 & (iris$Petal.Length > 2 &
  iris$Sepal.Length > 6.5 & (iris$Sepal.Width > 2.8 & iris$Sepal.Width < 3.5) & iris$Petal.Length > 5 &
  TRUE ~ as.character(x)
)
```

```
##   [1] "1"          "2"          "3"          "4"          "5"
##   [6] "6"          "7"          "8"          "9"          "10"
##  [11] "11"         "12"         "13"         "14"         "15"
##  [16] "16"         "17"         "18"         "19"         "20"
##  [21] "21"         "22"         "setosa"     "24"         "25"
##  [26] "26"         "27"         "28"         "29"         "30"
##  [31] "31"         "32"         "33"         "34"         "35"
##  [36] "36"         "37"         "setosa"     "39"         "40"
##  [41] "41"         "42"         "43"         "44"         "45"
##  [46] "46"         "47"         "48"         "49"         "50"
##  [51] "51"         "52"         "53"         "veriscolor" "55"
##  [56] "56"         "57"         "58"         "59"         "veriscolor"
##  [61] "61"         "62"         "veriscolor" "64"         "65"
##  [66] "66"         "67"         "veriscolor" "69"         "veriscolor"
##  [71] "71"         "72"         "73"         "74"         "75"
##  [76] "76"         "77"         "78"         "79"         "veriscolor"
##  [81] "veriscolor" "veriscolor" "veriscolor" "84"         "85"
##  [86] "86"         "87"         "veriscolor" "89"         "veriscolor"
##  [91] "veriscolor" "92"         "veriscolor" "94"         "veriscolor"
##  [96] "96"         "97"         "98"         "veriscolor" "100"
## [101] "101"        "102"        "virginica"  "104"        "105"
## [106] "virginica"  "107"        "virginica"  "109"        "110"
## [111] "111"        "112"        "virginica"  "114"        "115"
## [116] "116"        "117"        "118"        "119"        "120"
## [121] "virginica"  "122"        "123"        "124"        "virginica"
## [126] "virginica"  "127"        "128"        "129"        "virginica"
## [131] "131"        "132"        "133"        "134"        "135"
## [136] "virginica"  "137"        "138"        "139"        "virginica"
## [141] "virginica"  "virginica"  "143"        "virginica"  "virginica"
## [146] "virginica"  "147"        "148"        "149"        "150"
```