Descriptive Statistics and Associations

Blake Pappas

8/28/2021

Writing Functions in R

R has built-in functions for most standard statistical methods. However, you may at times find it useful to write your own functions.

Functions are defined using `function`. When you define a function, you decide what its arguments (inputs) should be. The code to be implemented by the function is written inside curly braces. The code inside braces should depend only on the function's arguments.

Example: A function called `my.product.min` is defined below. It has one argument, `x`, which is a numeric vector. The function multiplies all elements of a vector `x` by the smallest element of `x`.

```
my.product.min <- function(x)
{
  min <- min(x)
  return(x * min)
}
```

Once the function is defined, it can be called in your workspace just like any other function.

```
x1 <- rnorm(10)  # Generates 10 random numbers stored in a vector "x1"
my.product.min(x1) # Apply my.product.min function to x1
```

```
##  [1]  0.9898860  0.2455262  0.7336690 -0.5363483 -1.6260934 -0.3131638
##  [7] -0.6525532  0.7960534  2.6566182 -3.0412120
```

```
x2 <- rnorm(14)  # Generates 14 random numbers and apply function
my.product.min(x2)
```

```
##  [1] -3.6672174 -2.8239707 -0.7833182 -1.9953147  1.4627350  1.6032166
##  [7]  0.8416402 -1.9644506 -2.1040884 -3.2490997 -1.1038645 -1.2705109
## [13]  3.0074298 -1.4814220
```

Example 2: Now, define a function with two arguments. The function `sum.squares.scalar1` will require a vector, `y.vec`, and a single number (scalar), called `a`. It will compute the sum of squared deviations of the elements of `y.vec` from `a`. Using notation, the function will return the following quantity:

$$\sum_{i=1}^{n}(y_i - a)^2$$

Here is the code to define the function.

```r
sum.squares.scalar1 <- function(y.vec, a)
{
  return(sum((y.vec - a)^2))
}
```

This code chunk tests the function on random numbers.

```r
y1 <- rnorm(100) # Generates 100 random numbers
a1 <- 1.5

sum.squares.scalar1(y1, a1)
```

```
## [1] 346.9067
```

Example 3: If you write functions often, you might want to insert a few quality checks into the function. For example, you might want to make sure the user provides appropriate inputs so that the function behaves as expected. The `stop` function allows you to produce an error message when something goes wrong.

Here is an example using the `sum.squares.scalar2` function. It stop the function and produces an error message if "a" has more than one element; that is, if it is not a scalar.

```r
sum.squares.scalar2 <- function(y.vec, a)
{
  if(length(a) > 1){stop("Error: a should be a scalar.")}
  return(sum((y.vec - a)^2))
}
```

The code chunk below tests it out. Uncomment line 76 (remove the # sign) temporarily to see the error message. Recomment (add the # sign) afterwards to ensure the .Rmd file runs without errors.

```r
y1 <- rnorm(10) # Vector of 10 random numbers
a2 <- c(10, 11, -2) # Vector of 3 numbers
a3 <- 5 # Scalar

# sum.squares.scalar2(y1, a2)
```

```r
sum.squares.scalar2(y1, a3)
```

```
## [1] 293.3061
```

Exercises

The file `Airbnb_NOLA.csv` contains data from a random sample of 250 Airbnb listings in New Orleans. These listings were scraped from the Airbnb website in August 2018. The data set contains several variables, including the `Number_of_reviews` at the time of data collection and `Price`, the average price per night.

Exercise 1: Explore Marginal Distributions

A *marginal distribution* refers to the behavior of one variable at a time. Univariate exploratory analyses give insight into the marginal distributions of variables.

Answer the following questions. Include code chunks in your response where appropriate, but make sure to include text answering the questions.

1. Calculate the mean, median, standard deviation, and range of the variable `Price`.

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

mean(NOLA$Price)
```

```
## [1] 196.7763
```

```
median(NOLA$Price)
```

```
## [1] 130
```

```
sd(NOLA$Price)
```

```
## [1] 254.1159
```

```
range.Price <- max(NOLA$Price) - min(NOLA$Price)
range.Price
```

```
## [1] 9986
```

Answer: The mean of `Price` is 196.7763. The median of `Price` is 130. The standard deviation is 254.1159. The range of `Price` is 9,986.

2. List the possible values of the variable `Room_Type` and give the proportion of observations that take each value.

```
unique(NOLA$Room_Type)
```

```
## [1] "Entire home/apt" "Private room"    "Shared room"
```

```
Count_Room_Type <- data.frame(c(NOLA$Room_Type))
nrow(Count_Room_Type)
```

```
## [1] 5878
```

```
Room_Proportion <- table(NOLA$Room_Type)
Room_Proportion
```

```
##
## Entire home/apt    Private room     Shared room
##            4920             920              38
```

```
(Room_Proportion / nrow(Count_Room_Type)) * 100
```

```
##
## Entire home/apt    Private room     Shared room
##      83.7019394      15.6515822       0.6464784
```

Answer: The possible values of the variable `Room_Type` include "Entire home/apt", "Private room", and "Shared room." The proportion of observations for "Entire home/apt", "Private room", and "Shared room" are 4,920 (83.7%), 920 (15.7%), and 38 (0.6%), respectively.

3. 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'.

Based on these statistics, does there appear to be an association between the `Room_Type` and the `Price`?

```
Price_RT <- c(subset(NOLA, NOLA$Room_Type == 'Entire home/apt'))
mean(Price_RT$Price)
```

```
## [1] 216.3012
```

```
median(Price_RT$Price)
```

```
## [1] 150
```

```
sd(Price_RT$Price)
```

```
## [1] 269.5016
```

```
Price_LT <- c(subset(NOLA, NOLA$Room_Type == 'Private room'))
mean(Price_LT$Price)
```

```
## [1] 98.81739
```

```
median(Price_LT$Price)
```

```
## [1] 75
```

```
sd(Price_LT$Price)
```

```
## [1] 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.

4. What proportion of listings have more than 50 reviews?

```r
more_than_50 <- subset(NOLA, NOLA$Number_of_Reviews > 50)
nrow(more_than_50)
```

```
## [1] 1486
```

```r
nrow(NOLA)
```

```
## [1] 5878
```

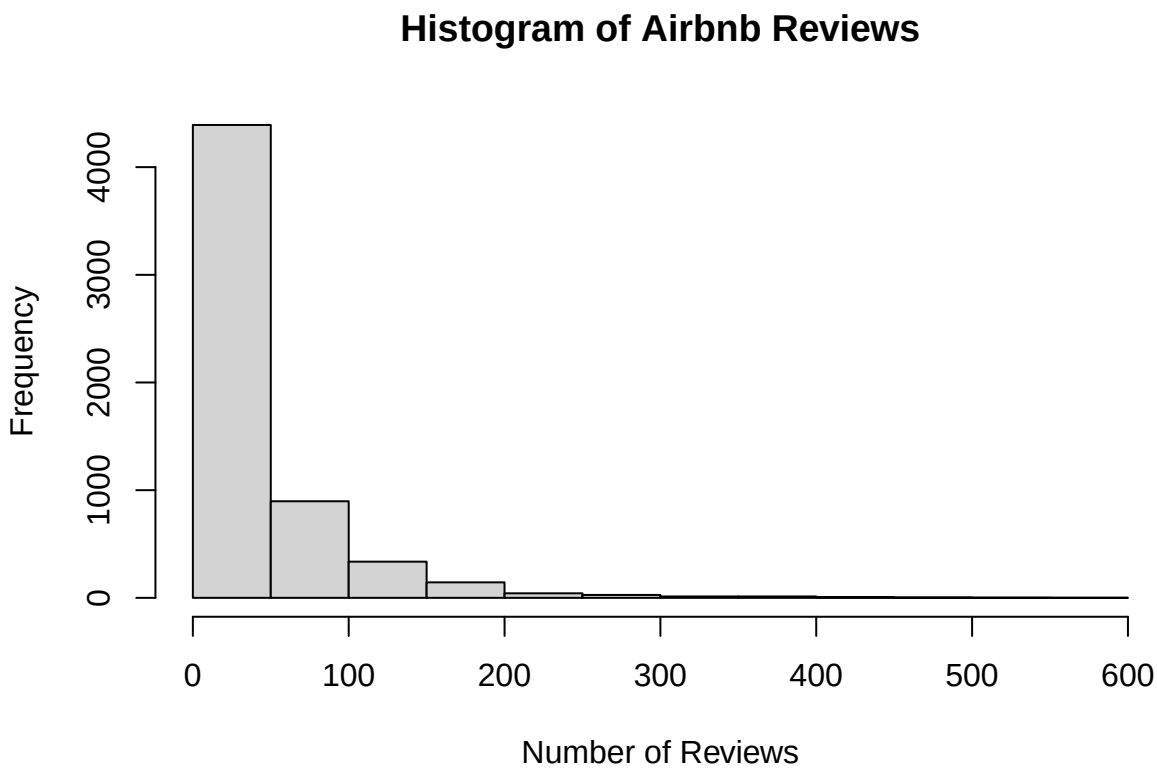
```r
Proportion_More_Than_50 <- (nrow(more_than_50) / nrow(NOLA)) * 100
Proportion_More_Than_50
```

```
## [1] 25.28071
```

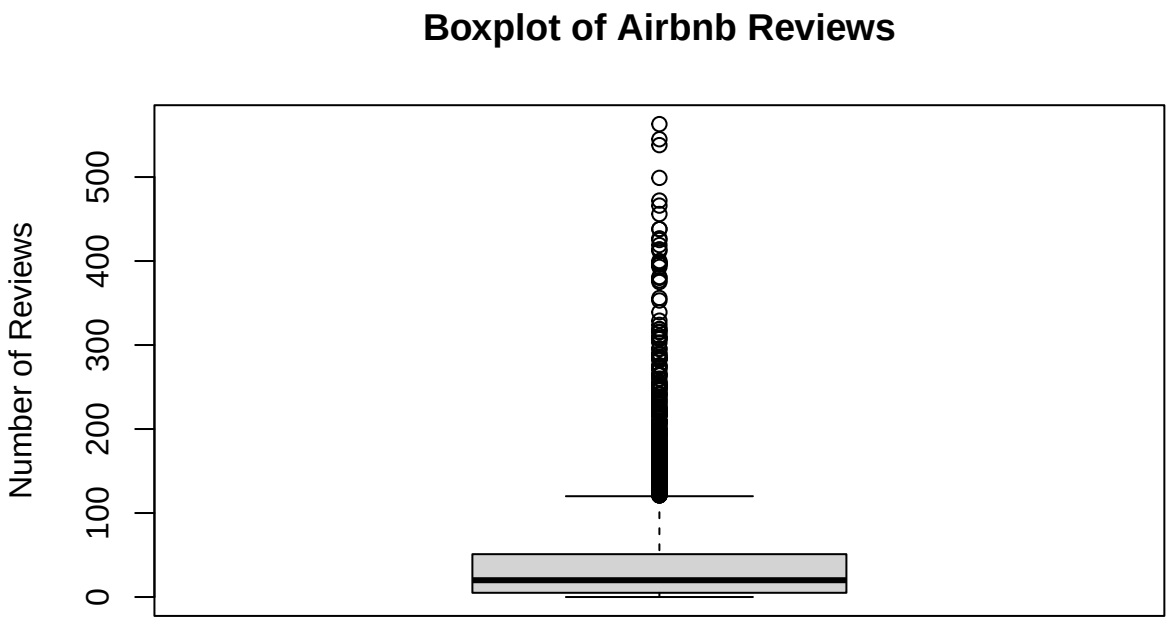
Answer: 1,486 (25.28% of) listings have more than 50 reviews.

5. Create a histogram and boxplot of the variable `Number_of_reviews`. Give the plots informative titles and axis labels. Describe the shape of the distribution.

```r
hist(NOLA$Number_of_Reviews, main = "Histogram of Airbnb Reviews", xlab = "Number of Reviews")
```



```r
boxplot(NOLA$Number_of_Reviews, main = 'Boxplot of Airbnb Reviews', ylab = 'Number of Reviews')
```



Answer: Based on the histogram and box plot, one can infer that the shape of the distribution is right (positively) skewed.

6. What population does this data set represent? Do you think the statistics you just calculated would be representative of Airbnb prices in another city? What about in New Orleans during a different time of year?

Answer: This data set represents Airbnb property listings in New Orleans, LA from 2013-2018. I do think the statistics would be representative of Airbnb prices in another city because of the large size of the sample which I had been provided (5,878 properties). Typically, the larger a sample size is, the closer that the sample mean and standard deviation approach the population mean and standard deviation. However, I do not think that these statistics would be as representative during a different time of year. For example, I would expect prices paid (on average) for Airbnbs in New Orleans to be positively skewed during the month of February because of Mardi Gras. This event draws thousands to the city, and rental prices increase as a result of increased demand for housing rentals, but not enough supply. If you compared the average Airbnb rental price in New Orleans to that of the national average for the month of February, you would likely see that it is statistically, significantly higher.

Exercise 2 : Writing Functions

Write a function to calculate the standard deviation of a numeric vector.

Answer: See below for my custom standard deviation function.

```
my.sd <- function(x)
{
 standard_deviation <- sd(x)
 return(standard_deviation)
}
```

Use the code chunk below to compare your function's results to those of R's built-in `sd` function. (They should give the same answer!)

```
x <- rnorm(20)
sd(x)
```

```
## [1] 1.253998
```

```
my.sd(x)
```

```
## [1] 1.253998
```

Exercise 3: Writing Functions

Repeat exercise 3, but this time require the input vector to have a length of at least 2. Return an error message if it has only one element. (The standard deviation is not defined for a single number!) See example 3 as a reference and use the `stop` function.

Answer:

```
my.sdev <- function(x)
{
  if(length(x) < 2){stop("Error: Standard deviation requires more than one element.")}
  return(sd(x))
}
```

Use this chunk to test your function. Uncomment line 207 temporarily to see the error message. Recomment afterwards to ensure the .Rmd file runs without errors.

```
y1 <- 3 # Vector with one element -- it shouldn't work.
y2 <- rnorm(3) # Vector with three elements - should work!

# Apply Your Function to y1 and y2 in This Space

# my.sdev(y1)
my.sdev(y2)
```

```
## [1] 0.5272262
```