R Users Guide to Stat 201: Chapter 3

Michael Shyne, 2017

Chapter 3: Statistics for Describing, Exploring, and Combining Data

Chapter 3 covers summary or descriptive statistics. For the most part, these are very easy to generate in R. We will use the `mtcars` data set.

Measures of Center

Mean

We have already used the `mean()` function in the Chapter 1 guide.

```
mean(mtcars$mpg)
```

```
## [1] 20.09062
```

Median

The function for medians is simply `median()`

```
median(mtcars$mpg)
```

```
## [1] 19.2
```

Mode

There is not a built-in R function for mode. For many data sets it might be easiest to figure the mode by just examining the data or producing a frequency table. For other situations, though, it would be nice to have a function. We can write our own. Note: There are several new concepts being utilized here. I'm not going to explain them at the moment, but they will perhaps be covered in later chapters.

```
# Define a function to calculate mode, this can
#   now be used like any other R function.
get.mode <- function(x,...){
    # Make a table of x
    x.table <- table(x,...)

    # Return the names of the rows which have the max frequency
    return(rownames(x.table)[max(x.table)==x.table])
}

# We will have multiple modes
get.mode(mtcars$mpg)
```

```
## [1] "10.4" "15.2" "19.2" "21"   "21.4" "22.8" "30.4"
```

Remember, we can find the mode of a nominal or ordinal categorical variable.

```
cyl.fac <- as.factor(mtcars$cyl)

get.mode(cyl.fac)
```

```
## [1] "8"
```

Unfortunately, with our quickly written function, no mode, which occurs when all values have the same frequency, will result in a list of all values. If we were to write a "production-ready" function, we would need to handle this. We'll leave it be for now.

Midrange

R also lacks a function for midrange, but it is easily calculated. Remember, midrange is halfway between the minimum and maximum values, which we calculate as the mean of those values.

```
mean(c(min(mtcars$mpg), max(mtcars$mpg)))
```

```
## [1] 22.15
```

Measures of Variation

Range

The `range()` function does not provide the range as we expect it here. It returns instead the minimum and maximum values of the vector provided. Range, as a measure of variation, is the difference of those values.

```
diff(range(mtcars$mpg))
```

```
## [1] 23.5
```

Variance and standard deviation

R does have functions for these important and widely used statistics. They are `var()` and `sd()`.

```
var(mtcars$mpg)
```

```
## [1] 36.3241
```

and...

```
sd(mtcars$mpg)
```

```
## [1] 6.026948
```

Coefficient of variation

This is another one we will need to calculate "by hand". Coefficient of variation is standard deviation divided by mean.

```
cv <- sd(mtcars$mpg) / mean(mtcars$mpg)
cv
```

```
## [1] 0.2999881
```

Multiply by 100, if we want it as a percent.

```
cv * 100
```

```
## [1] 29.99881
```

Measures of Relative Standing and Boxplots

Z-scores

A z-score for a given value is value minus mean divided by standard deviation. We can always execute this calculation "by hand."

```
# What is the z-score for 30 mpg?
x <- 30

z <- (x - mean(mtcars$mpg)) / sd(mtcars$mpg)   # Notice the parentheses
z
```

```
## [1] 1.644178
```

There is an R function to do this more directly, but it is not much easier to use and may be harder to remember. I'll give it here for reference.

```
z.2 <- scale(x, mean(mtcars$mpg), sd(mtcars$mpg))
z.2[1,1]
```

```
## [1] 1.644178
```

The find the reverse, to find a value in the units of the data set corresponding to a given z-score, again the simplest method is to just do the calculation "by hand".

```
(z * sd(mtcars$mpg)) + mean(mtcars$mpg)
```

```
## [1] 30
```

Percentiles

To find values which correspond to a particular percentile (or multiple percentiles), we use the `quantile()` function.

```
# Find the 30th percentile of MPG
quantile(mtcars$mpg, .3)
```

```
##   30%
## 15.98
```

The reverse, finding the percentile of a value, is trickier. We need to employ a slightly advanced technique of calling an R function which will give us a function which we then can use to get our percentile.

```
# What is the percentile of 24.4 MPG
pc <- ecdf(mtcars$mpg)(24.4)

# To display as a percentile, as the book defines it,
#   multiply by 100 and round up
ceiling(pc*100)
```

```
## [1] 82
```

Quartiles and 5 number summary

Since quartiles are just particular percentiles, we already know how to find them. We just need to give the `quantile` function a vector of percentiles to find.

```
# Find Q1, Q2 and Q3
quantile(mtcars$mpg, c(.25, .50, .75))
```

```
##    25%    50%    75%
## 15.425 19.200 22.800
```

We can use the same method to produce a 5 number summary.

```
quantile(mtcars$mpg, c(0, .25, .50, .75, 1))
```

```
##     0%    25%    50%    75%   100%
## 10.400 15.425 19.200 22.800 33.900
```

Alternatively, we can use the `summary()` function. This is another general purpose function which will attempt to provide meaningful information on anything that is passed to it. Results are not always as useful as one would hope, but if we give it a vector of quantitative data, it will return the 5 number summary plus the mean.

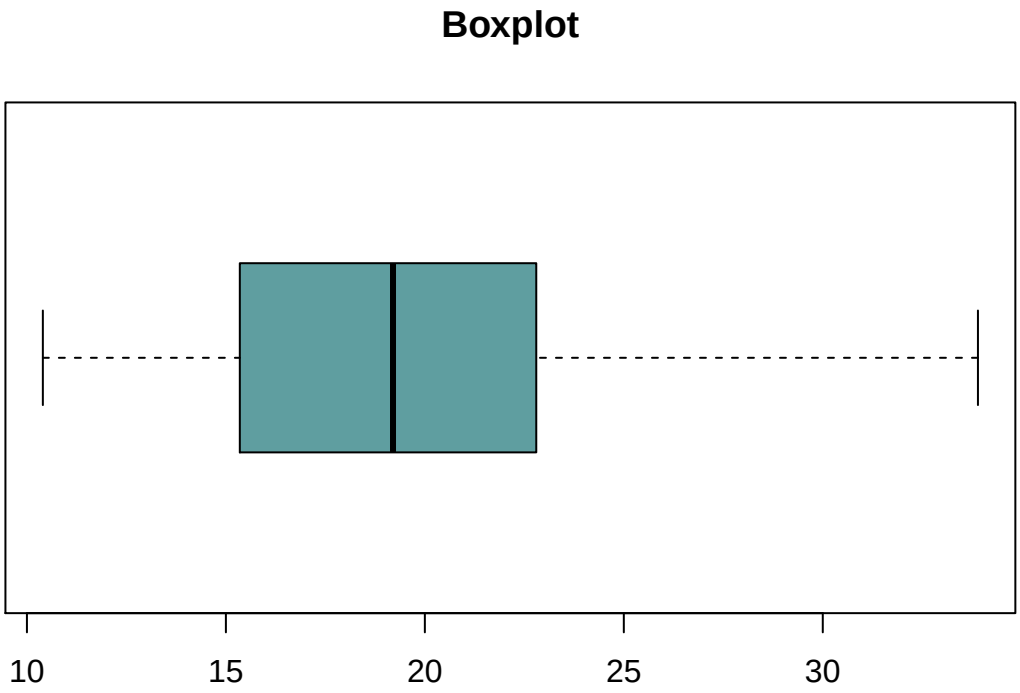
```
summary(mtcars$mpg)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   10.40   15.42   19.20   20.09   22.80   33.90
```

Boxplots

Boxplots are produced with the `boxplot()` function. By default, vertical plots are produced.

```
boxplot(mtcars$mpg,
        horizontal = TRUE,   # Draw a horizontal plot
        col = "cadetblue",
        main = "Boxplot")
```



License