Week 1: Practice R

9/17/2020

1. Suppose you want to create a vector of the numbers 100 to 1, where 100 is written 100 times, 99 is written 99 times and so on till 1 is written 1 time, namely 100,100,...,3,3,3,2,2,1. Provide an R command that can help you do this.
   Hint: Check the `rep` command in R.

```
# for (i in 100:1) {
#   print(rep(i, times = i))
# }

# alternative
# rep(value, times)
rep(x = c(10:1), times = c(10:1))
```

```
##  [1] 10 10 10 10 10 10 10 10 10 10  9  9  9  9  9  9  9  9  8  8  8  8  8  8
## [26]  8  8  7  7  7  7  7  7  7  6  6  6  6  6  6  5  5  5  5  5  4  4  4  4  3
## [51]  3  3  2  2  1
```

```
# alternative
# rep(10:1, c(10:1))
# rep(100:1, c(100:1))
```

2. Suppose you have a vector `A <- c(1,2,0,4)` and a vector `B <- c(3,6)`, then what is the result of `A*B` in R?

```
A <- c(1, 2, 0, 4)
B <- c(3, 6)
A*B
```

```
## [1]  3 12  0 24
```

```
# the shorter vector is recycled
# longer vector must be of a multiple of shorter vector length
# 1*3  2*6  0*3  4*6
# 3 12 0 24
```

3. Run the following R commands and explain the numbers that appear, each time we run table(gender).

```
gender <- factor(c(rep("female", 91), rep("male", 92)))
table(gender)
```

```
## gender
## female   male
##     91     92
```

```r
# number of times each level appeared in the vector
# female   male
#     91     92

gender1 <- factor(gender, levels=c("male", "female"))
table(gender1) # will give the same result as gender. because the levels are defined correctly
```

```
## gender1
##   male female
##     92     91
```

```r
gender2 <- factor(gender, levels=c("Male", "female"))
table(gender2)
```

```
## gender2
##   Male female
##      0     91
```

```r
# "Male" level will be of 0 value because the spelling is different (because M is capitalised)
```

4. Suppose we want to convert a factor variable to a numeric variable in, how do we do so? For example, convert the factor variable below to numeric.

```r
X <- factor(c(4, 5, 6, 6, 4))
X
```

```
## [1] 4 5 6 6 4
## Levels: 4 5 6
```

```r
# as.numeric(X) returns 1 2 3 3 1
# each factor is labelled a number, return value indicates which factor the element in the vector belon

x_numeric <- as.numeric(as.character(X))
x_numeric
```

```
## [1] 4 5 6 6 4
```

5. Load the dataframe mtcars available with the base R installation in data. The data was extracted from the 1974 Motor Trend US magazine, and comprises fuel consumption and 10 aspects of automobile design and performance for 32 automobiles (1973-74 models).

(a) Use `ggplot()` to plot the weight versus miles per gallon and comment on the relationship.

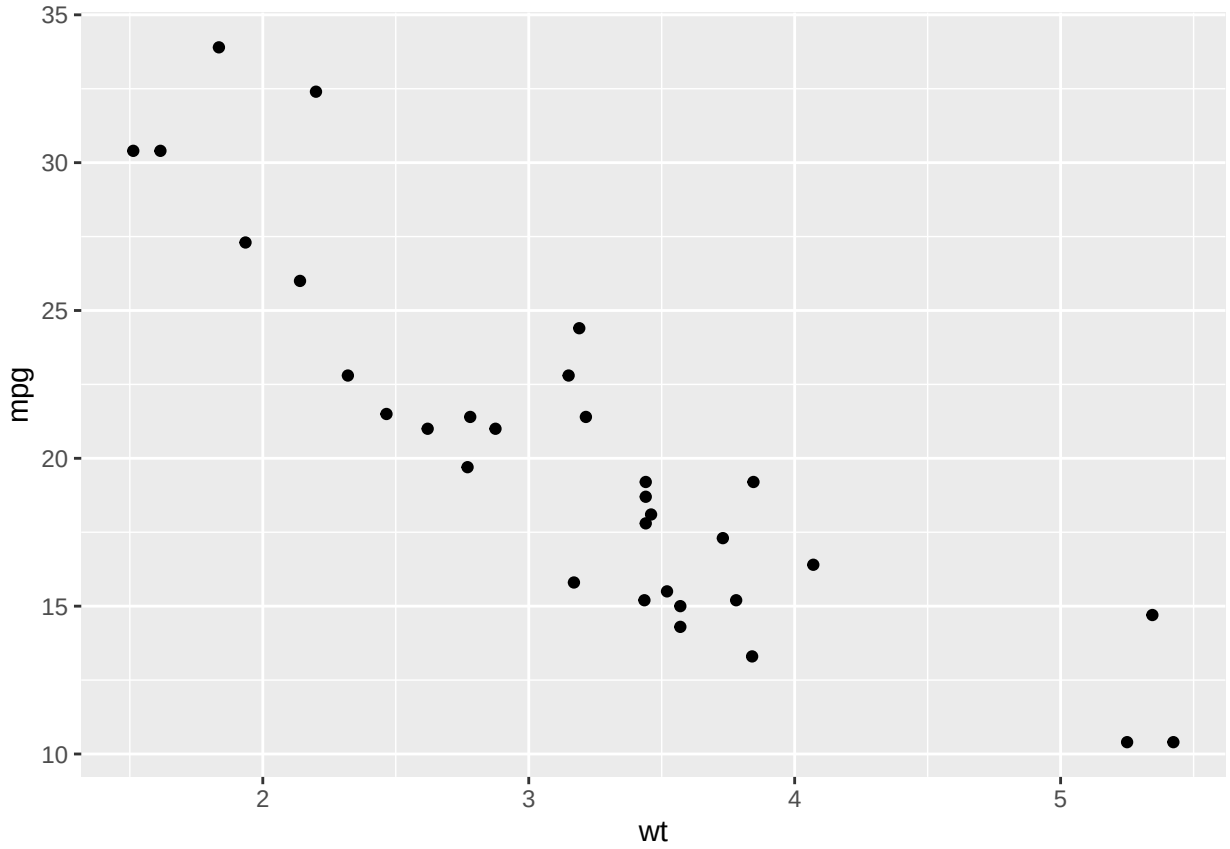
(b) Add to this plot, a coloring of the points based on the number of cylinders, and the sizing of the points based on the displacement (volume) of the car.

(c) Use the `tapply()` function to compute the standard deviation of the mpg for groups with the same number of cylinders.

```r
library(ggplot2)
mtcars
```

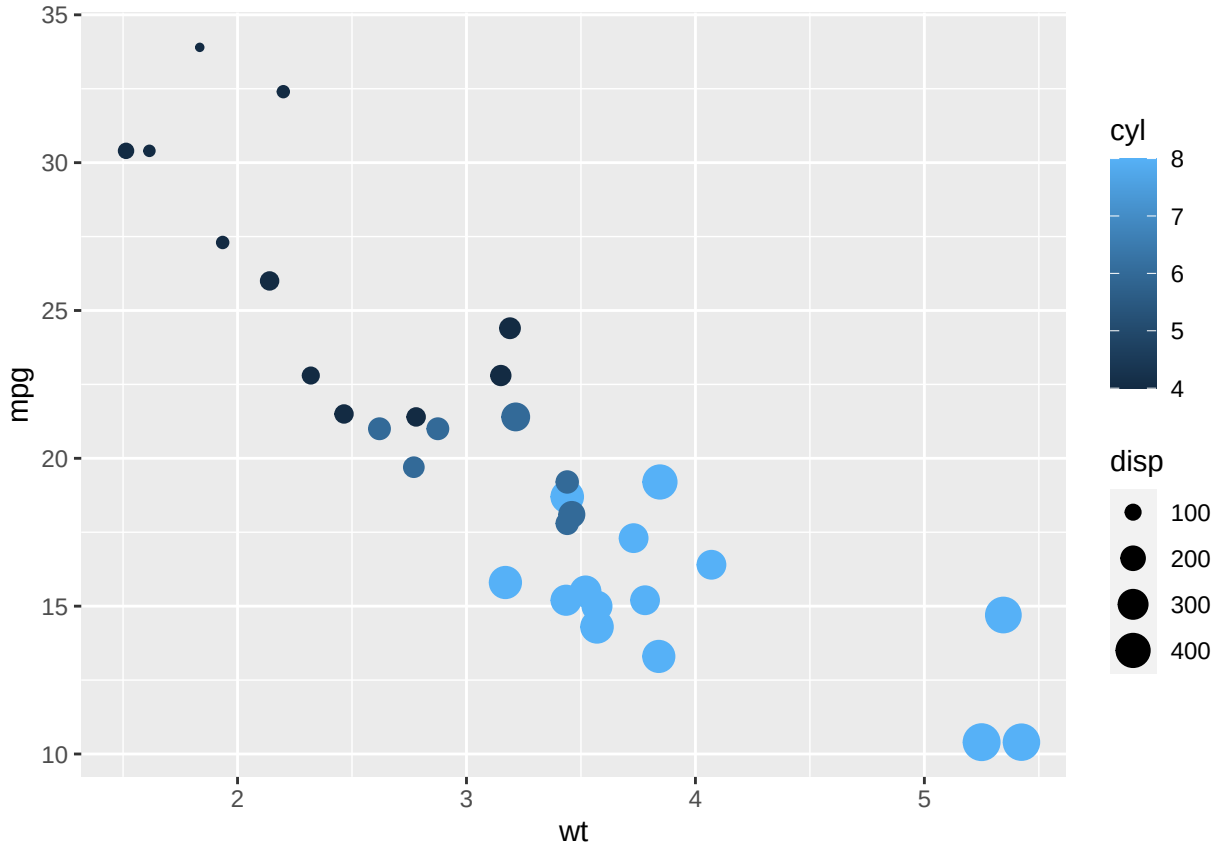
```
##                     mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4          21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag      21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710         22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive     21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout  18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant            18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360         14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D          24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230           22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280           19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C          17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE         16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL         17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC        15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood 10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4  8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial  14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128           32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic        30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla     33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona      21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger   15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin        15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28         13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird   19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9          27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
## Porsche 914-2      26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa       30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L     15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino       19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora      15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E         21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

```r
# part (a)
ggplot(data = mtcars, aes(x = wt, y = mpg)) + geom_point()
```




```
# weight and miles per gallon have an inverse relationship

# part (b)
ggplot(mtcars, aes(x = wt, y = mpg, color = cyl, size = disp)) + geom_point()
```




```
# part (c)
tapply(mtcars$mpg, mtcars$cyl, sd)
```

```
##        4        6        8
## 4.509828 1.453567 2.560048
```