Mt1 2017

Irlanda Ayon-Moreno

5/4/2018

```
library(dplyr)
```

```
## Warning: package 'dplyr' was built under R version 3.4.2
```

```
library(ggplot2)
```

Command Outputs

```
x <- 1:5
y <- letters[1:5]
names(y) <- 1:5
```

What is the output of `y[x <= 2]`?

```
y[x <= 2]
```

```
##   1   2
## "a" "b"
```

What is the output of `y[x[5]]`?

```
y[x[5]]
```

```
##   5
## "e"
```

What is the output of `y[!(x < 3)]`?

```
y[!(x < 3)]
```

```
##   3   4   5
## "c" "d" "e"
```

What is the output of `y[x/x]`?

```
y[x/x]
```

```
##   1   1   1   1   1
## "a" "a" "a" "a" "a"
```

What is the output of `y[x[-2][3]]`?

```
y[x[-2][3]]
```

```
##   4
## "d"
```

What is the output of `x[(y != "c") & (y != "a")]`?

```
x[(y != "c") & (y != "a")]
```

```
## [1] 2 4 5
```

What is the output of `x[(y == "c") | (y == "a")]`?

```
x[(y == "c") | (y == "a")]
```

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

Do these commands return the numeric vector given by: 1 2 3 4 5?

- `c(1 2 3 4 5)`
- `1:5`
- `list(1, 2, 3, 4, 5)`
- `seq(from = 1, to = 5, by = 1)`
- `[1, 2, 3, 4, 5]`
- `c("1", "2", "3", "4", "5")`
- `1 + c(0, 1, 2, 3, 4)`
- `sqrt(1, 4, 9, 16, 25)`

```
c(1 2 3 4 5) # no
1:5 # yes
list(1, 2, 3, 4, 5) # no
seq(from = 1, to = 5, by = 1) # yes
[1, 2, 3, 4, 5] # no
c("1", "2", "3", "4", "5") # no
1 + c(0, 1, 2, 3, 4) # yes
sqrt(1, 4, 9, 16, 25) # no
```

```
## Error: <text>:1:5: unexpected numeric constant
## 1: c(1 2
##         ^
```

Data Types

```
x <- c(1.1, 2.2, 3.3, 4.4)
x
```

```
## [1] 1.1 2.2 3.3 4.4
```

```
typeof(x)
```

```
## [1] "double"
```

```
y <- (x == 1.1)
y
```

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

```
typeof(y)
```

```
## [1] "logical"
```

```
z <- y + 0
z
```

```
## [1] 1 0 0 0
```

```
typeof(z)
```

```
## [1] "double"
```

```
w <- c(x, "5.5")
w
```

```
## [1] "1.1" "2.2" "3.3" "4.4" "5.5"
```

```
typeof(w)
```

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

```
xyz1 <- c(x, y, z, w[1])
xyz1
```

```
##  [1] "1.1"   "2.2"   "3.3"   "4.4"   "TRUE"  "FALSE" "FALSE" "FALSE"
##  [9] "1"     "0"     "0"     "0"     "1.1"
```

```
typeof(xyz1)
```

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

Files

True or False?

“”, **“;”**, ** " **“, and** ”,“** are charcters commonly used as field-delimiters

A file format is a way of interpreting the bytes in a file

The CSV format name stands for Common Standard Value

A CSV format is a special case of a spreadsheet format

In a CSV file, there must be a single header line containing the names of the fields

Binary formats tend to provide smaller files and faster access speeds

One disadvantage of plain text files is their lack of standard structure

Data stored in a binary format can be accessed using a text editor

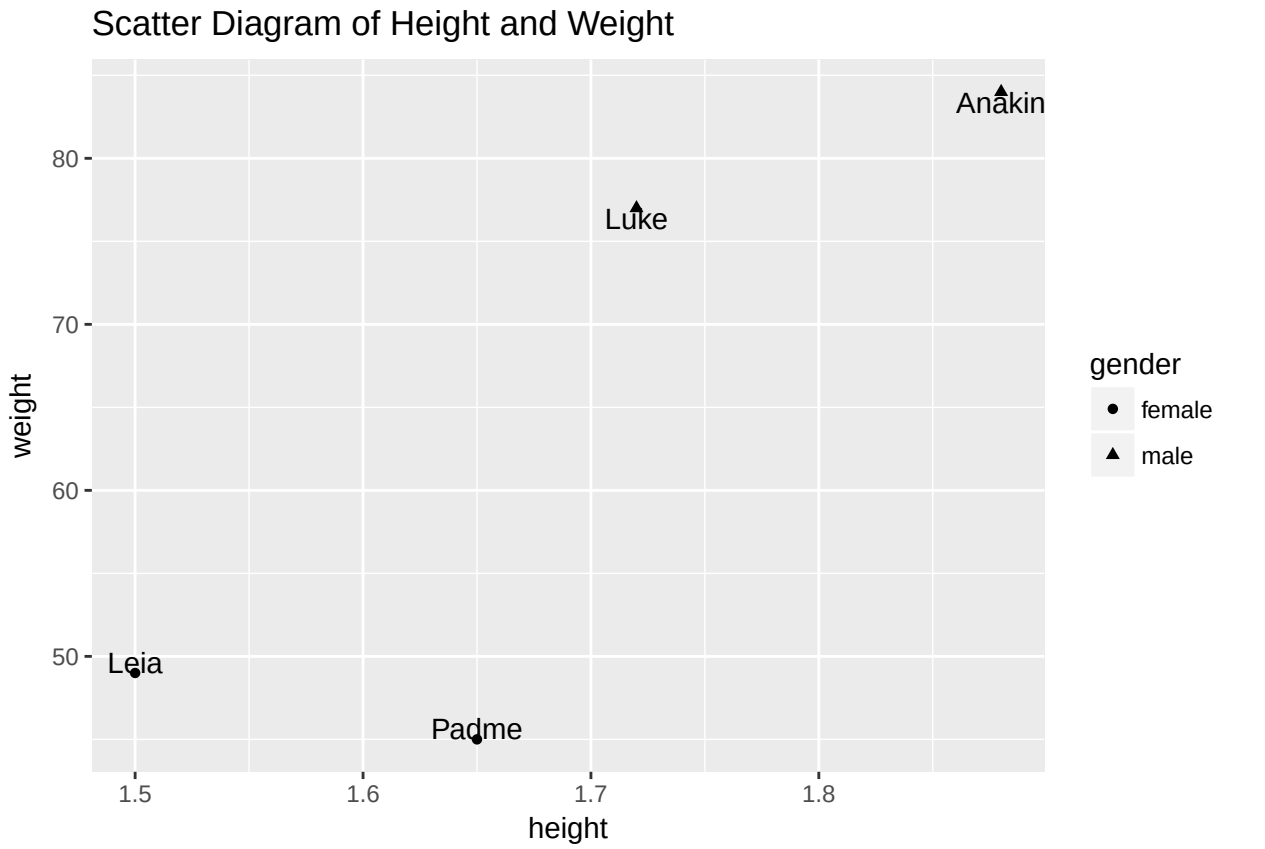
ggplot2

```
sw
```

```
##     name gender height weight
## 1 Anakin   male   1.88     84
## 2  Padme female   1.65     45
## 3   Luke   male   1.72     77
## 4   Leia female   1.50     49
```

Create a scatter plot of `height` and `weight`. Distinguish the points using a person's `gender` and labeling each point with their `name`. Finally, add a title.

```
ggplot(data = sw, mapping = aes(x = height, y = weight)) +
  geom_point(aes(shape = gender)) +
  geom_text(aes(label = name), vjust = "inward") +
  ggtitle("Scatter Diagram of Height and Weight")
```

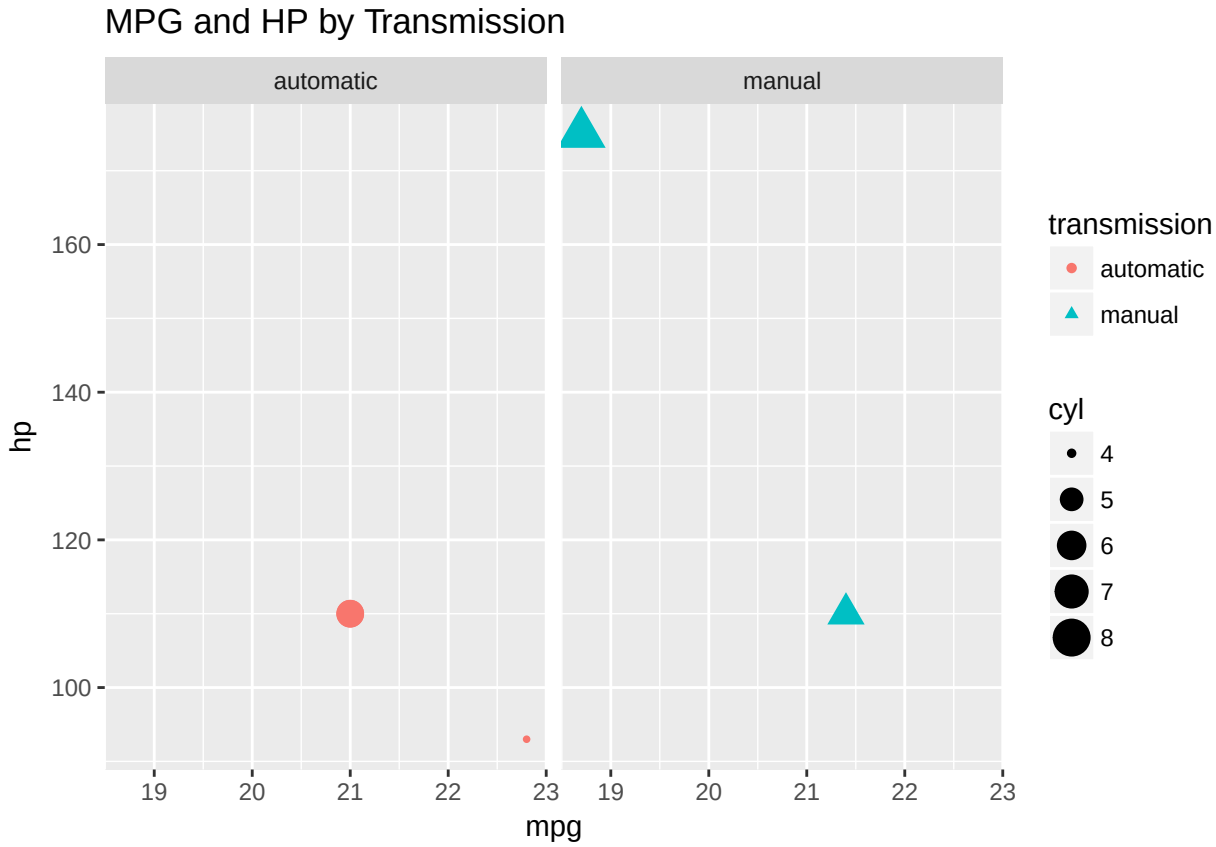


Create a scatterplot of `mpg` and `hp`, with the size of the dots depending on `cyl`, the shape of the dots depending on `transmission`, and the color of the dots depending on `transmission`. Facet by `transmissionm` and include a title.

```
coches
```

```
##                    mpg cyl  hp transmission
## Mazda RX4         21.0   6 110    automatic
## Mazda RX4 Wag     21.0   6 110    automatic
## Dotsun 710        22.8   4  93    automatic
## Hornest 4 Drive   21.4   6 110       manual
## Hornest Sportabout 18.7   8 175       manual
```

```
ggplot(data = coches, mapping = aes(x = mpg, y = hp)) +
  geom_point(aes(size = cyl, shape = transmission, color = transmission)) +
  facet_grid(.~transmission) +
  ggtitle("MPG and HP by Transmission")
```



dplyr

```
sw
```

```
##     name gender height weight
## 1 Anakin   male   1.88     84
## 2  Padme female   1.65     45
## 3   Luke   male   1.72     77
```

```
## 4   Leia female   1.50     49
```

Which of the following commands returns the number of females and males?

- `sw %>% select(gender) %>% count()`
- `sw %>% count(gender)`
- `sw %>% group_by(gender) %>% count()`
- `sw %>% group_by(gender) %>% summarise()`

```
sw %>% select(gender) %>% count() # no
```

```
## # A tibble: 1 x 1
##       n
##   <int>
## 1     4
```

```
sw %>% count(gender) # yes
```

```
## # A tibble: 2 x 2
##   gender     n
##   <fct>  <int>
## 1 female     2
## 2 male       2
```

```
sw %>% group_by(gender) %>% count() # yes
```

```
## # A tibble: 2 x 2
## # Groups:   gender [2]
##   gender     n
##   <fct>  <int>
## 1 female     2
## 2 male       2
```

```
sw %>% group_by(gender) %>% summarise() # no
```

```
## # A tibble: 2 x 1
##   gender
##   <fct>
## 1 female
## 2 male
```

Which of the following commands returns the data (rows) of male individuals?

- `sw %>% select(gender == "male")`
- `sw %>% filter(gender == "male")`
- `sw %>% group_by(gender == "male")`
- `sw %>% filter(by == "male")`

```
sw %>% select(gender == "male") # no
```

```
## Error in FUN(X[[i]], ...): object 'gender' not found
```

```
sw %>% filter(gender == "male") # yes
```

```
##     name gender height weight
## 1 Anakin   male   1.88     84
## 2   Luke   male   1.72     77
```

```
sw %>% group_by(gender == "male") # no
```

```
## # A tibble: 4 x 5
## # Groups:   gender == "male" [2]
##   name   gender height weight `gender == "male"`
##   <fct>  <fct>   <dbl>  <dbl> <lgl>
## 1 Anakin male     1.88    84. TRUE
## 2 Padme  female   1.65    45. FALSE
## 3 Luke   male     1.72    77. TRUE
## 4 Leia   female   1.50    49. FALSE
```

```
sw %>% filter(by == "male") # no
```

```
## Error in filter_impl(.data, quo): Evaluation error: comparison (1) is possible only for atomic and li
```