

Bill's Drills Book

William McDonald

2020-04-19

Contents

1 Drills: Part of Every Healthy Intellectual Diet	5
2 bookdown Tips for This Document	7
2.1 Basic conventions	7
2.2 Referencing other parts of the document	7
2.3 Inserting pictures	8
2.4 Referencing citations:	8
3 Data Exploration	11
3.1 Counting things. The naming of parts.	11
3.2 Summarize is another very useful function:	14
4 Sampling	17
4.1 Think about throwing a bunch of dice.	17
4.2 A keen way to divide up a dataset into testing and training components.	17
5 Factor Practice	21
6 Crossing Trial	23
7 By Any Other Name	27
8 Correlation Plots	31
9 if_else() and case_when(): Comparison	39
9.1 case_when()	39
9.2 Compare this with if_else()	41
10 Subsetting	43
10.1 Subsetting using brackets	43
10.2 Subset using brackets by omitting the rows and columns we don't want	44
10.3 Subset using brackets in combination with the which() function and the %in% operator	44

10.4 Subset using the <code>subset()</code> function	45
10.5 Subset using dplyr's <code>filter()</code> and <code>select()</code>	45

Chapter 1

Drills: Part of Every Healthy Intellectual Diet

The goal of this book is to organize my R drills into reasonable chunks, the better to understand my strengths and weaknesses, and to plan new forays into data science.

Chapter 2

bookdown Tips for This Document

2.1 Basic conventions

- the `_bookdown.yml` file contains a snippet that is important to inserting the word “Chapter” before the chapter number in each of the Rmd files.
- packages are indicated in bold, like **dplyr**
- inline code and filenames are indicated in typewriter face using backticks, like `_bookdown.yml`
- `_output.yml` is modified from that used by Xie in his `bookdown-demo` (Xie, 2020); it evokes `style.css`, `toc.css`, `preamble.tex`, which are also borrowed from Xie.
- chapters are set in order by using adding 01, 02, 03, ... before the name of their Rmd, like `01chapter.Rmd`. Note that they can have short descriptive phrases, since the actual chapter titles are determined by the hashtag.

2.2 Referencing other parts of the document

This is a good place to practice referencing figures. Say that I want to refer the reader back to my first starwars figure. See Figure 3.1.

I can reference other pages in a similar fashion. See Chapter 10. Note that this works by referencing a `{#label}` placed in the chapter title.

See Chapter 1

See Chapter 3

Note that the `{#label}` uses a single run-together word. It does not tolerate spaces and this cannot be overcome by ‘quoting’ it.

2.3 Inserting pictures

Pictures can be included in the `test-book_files` subdirectories and referenced like this:

Some of the subdirectories throw an error in building the book, so I settled on `_bookdown_files/pathologyImages` as the location.

Also, I note that the build does not generate the caption unless the reference is on its own line.

Also note that some controls on image size are available. For instance, the same image can be displayed at 50% size:

2.4 Referencing citations:

In order to insert citations, one needs a `.bib` file in the project. I’ve included one in this project as `book.bib`. The yml header in Chapter 1 needs to have a `bibliography :` and `biblio — style :` line added.

To insert a citation, use the **citr** Addin from RStudio. **bookdown**, for instance, is cited thusly (Xie, 2020). Note that I need to figure out an adequate workflow of references. The convenience of Endnote in MS Word will not be available. Nonetheless, if I populate the `book.bib` and `packages.bib` files carefully, with `.txt` files generated in Endnote, I should be OK.

For instance, a recent dump of my Endnote library is in `bookFromEndnote.txt`. This can be opened in RStudio, and I can copy-and-paste references from the `.txt` file to my `book.bib`. For instance, if I have a breast paper that I want to cite here (Stevens and Parekh, 2016), I’d copy-and-paste the reference from `bookFromEndnot.txt` to `book.bib`.

Of note, Yihui Xie includes a nifty bit of code to automatically generate a bib database for R packages:

```
knitr::write_bib(c(.packages(), 'bookdown', 'knitr', 'rmarkdown', 'tidyverse', 'Comple
```

References appear automatically at the end of a chapter.

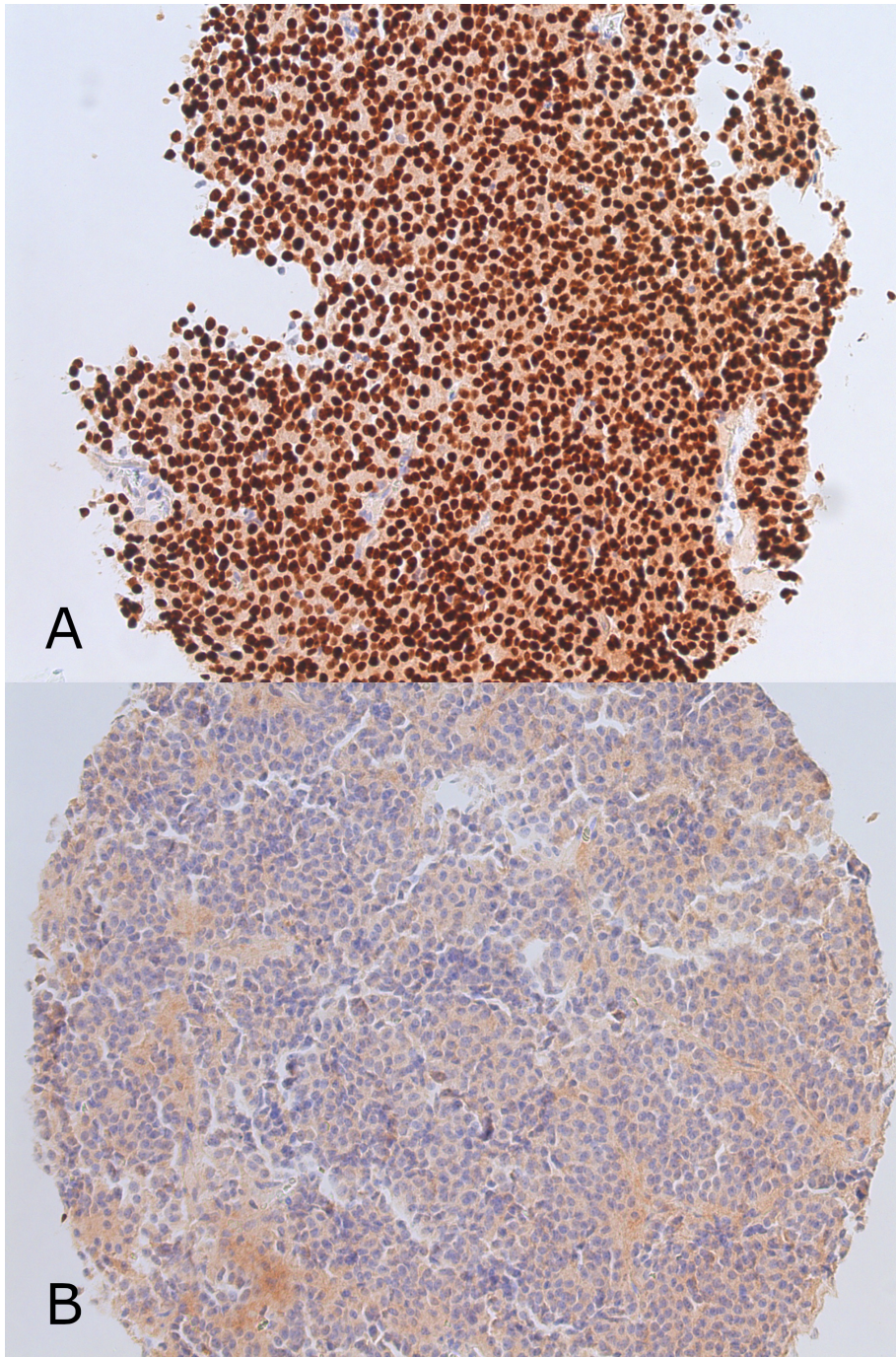


Figure 2.1: Tpit immunohistochemical stain. Figure A silent corticotroph. Figure B gonadotroph

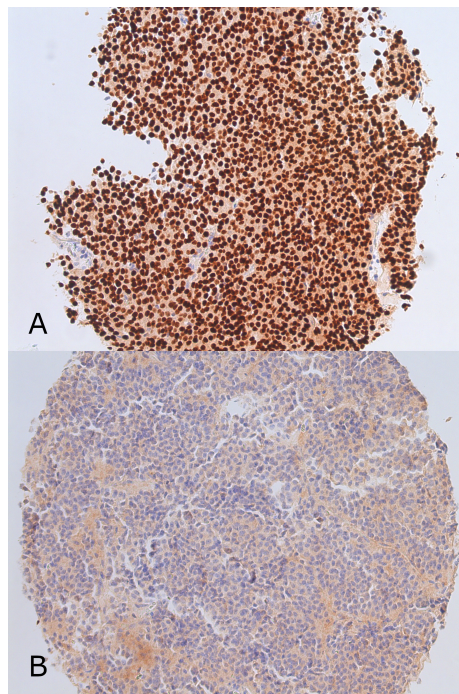


Figure 2.2: Tpit immunohistochemical stain. Figure A silent corticotroph. Figure B gonadotroph

Chapter 3

Data Exploration

Data exploration is one of the most important aspects of data science and forms the cornerstone of my drills. Nonetheless, I have lots of room for improvement.

I like Hadely Wickham's writing and find his approach exceptionally clear. Therefore, I'll use the **tidyverse**.

```
library(tidyverse)
```

3.1 Counting things. The naming of parts.

```
starwars %>%  
  filter(!is.na(species)) %>%  
  count(species = fct_lump(species, 5), sort = TRUE) %>%  
  mutate(species = fct_reorder(species, n)) %>%  
  ggplot(aes(species, n)) +  
  geom_col() + coord_flip()
```

I like stacked bars for their economy, but it's easy to over do it. Supperimposing gender onto the columns seems easy...

```
starwars %>%  
  filter(!is.na(species)) %>%  
  count(species = fct_lump(species, 5), gender = fct_lump(gender, 2), sort = TRUE) %>%  
  mutate(species = fct_reorder(species, n)) %>%  
  ggplot(aes(species, n, fill = gender)) +  
  geom_col() + coord_flip()
```

```
## Warning: Factor `gender` contains implicit NA, consider using  
## `forcats::fct_explicit_na`
```

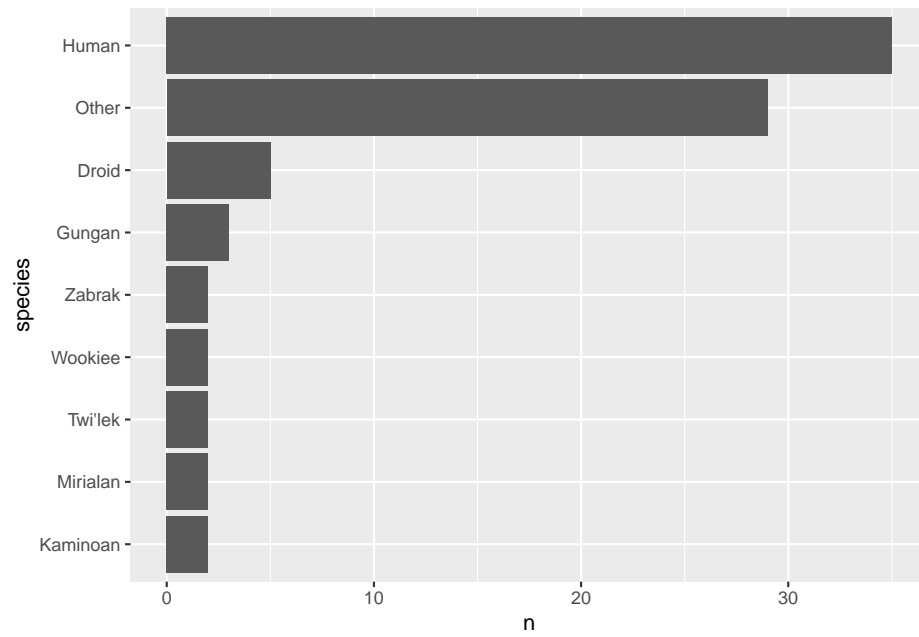


Figure 3.1: Starwars Figure 1

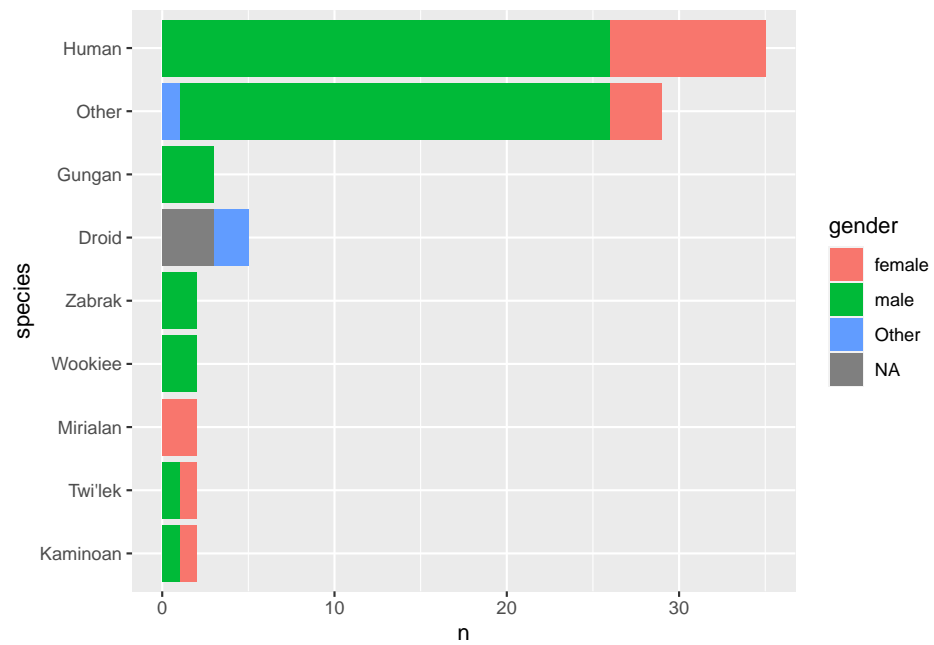


Figure 3.2: Starwars Figure 2

But note that I've got a problem: the Droids, which outnumber the Gungans, are now reordered to *after* the Gungans. This happens because the n that we're counting comprises subcategories of species *and* gender. Only three Gungan males exist (and no females), but that is enough to tie the Droid NA category. The Droid NA category come after the Gungan category, presumably because *male* comes before *NA*, or because *NA* comes last (more likely).

Exploring this, I see that I'm getting warning messages about the implicit NA's in gender. Note that the following renders a slightly different plot. I *still* have not fixed the order of the species.

```
starwars %>%
  filter(!is.na(species)) %>%
  count(species = fct_lump(species, 5), gender = fct_lump(gender, 2), sort = TRUE) %>%
  mutate(gender = fct_explicit_na(gender),
         species = fct_reorder(species, n)) %>%
  ggplot(aes(species, n, fill = gender)) +
  geom_col() + coord_flip()

## Warning: Factor `gender` contains implicit NA, consider using
## `forcats::fct_explicit_na`
```

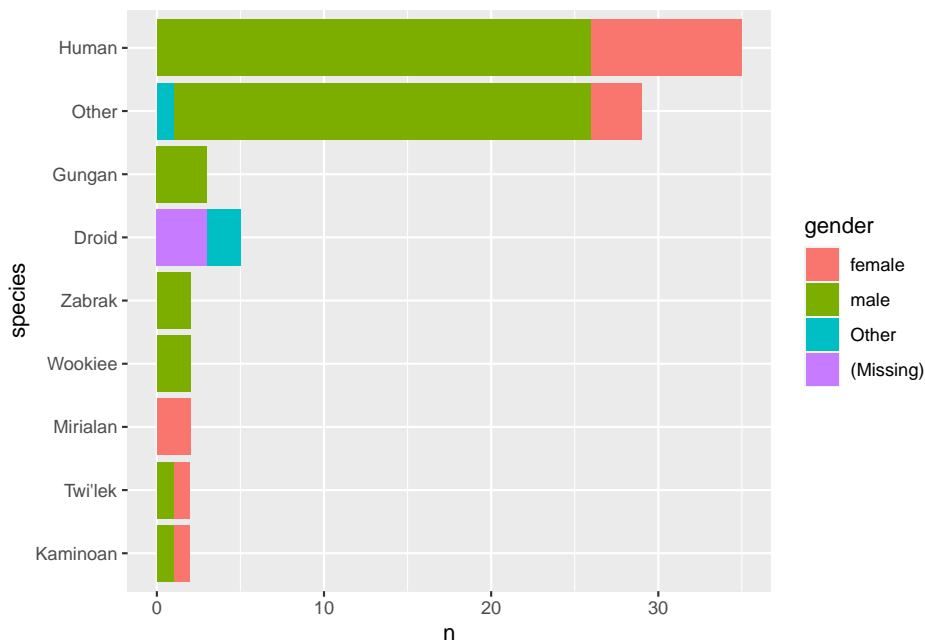


Figure 3.3: Starwars Figure 3

The trick here is to use `group_by()` and `ungroup()` wisely.

```
starwars %>% filter(!is.na(species)) %>%
  mutate(species = fct_lump(species, 5)) %>%
  group_by(species) %>%
  mutate(typeCount = n()) %>%
  ungroup() %>%
  mutate(species = fct_reorder(species, typeCount)) %>%
  ggplot()+
  geom_bar(aes(species, fill = gender))+
  coord_flip()
```

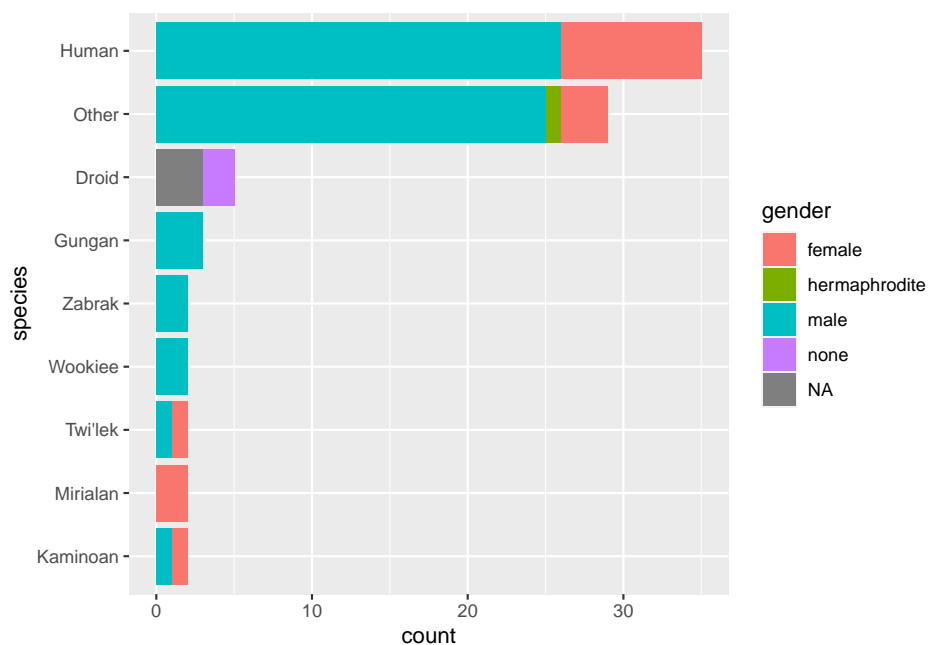


Figure 3.4: Starwars Figure 4

As opposed to using `count()`, which progressively narrows the information available to be used, by using `group_by()/mutate()/ungroup()` with `geom_bar()` we have all of the variables still available for plotting.

3.2 Summarize is another very useful function:

```
starwars %>%
  filter(!is.na(species)) %>%
  group_by(species) %>%
```



```
summarize(n=n(), mean = mean(height, na.rm = TRUE)) %>%  
  arrange(desc(n))
```

```
## # A tibble: 37 x 3  
##   species      n mean  
##   <chr>    <int> <dbl>  
## 1 Human      35  177.  
## 2 Droid       5  140  
## 3 Gungan      3  209.  
## 4 Kaminoan    2  221  
## 5 Mirialan     2  168  
## 6 Twi'lek      2  179  
## 7 Wookiee      2  231  
## 8 Zabrak       2  173  
## 9 Aleena       1   79  
## 10 Besalisk    1  198  
## # ... with 27 more rows
```


Chapter 4

Sampling

4.1 Think about throwing a bunch of dice.

```
sample(1:6, size=100, replace=TRUE)
```

```
## [1] 4 3 6 3 4 2 1 4 1 4 4 3 4 2 2 1 3 3 3 2 6 4 4 3 3 4 1 5 3 6 3 1 1 2 2 5 2
## [38] 2 6 3 5 3 5 3 4 6 4 3 2 3 2 6 1 5 6 1 4 4 4 2 1 5 1 6 3 1 5 2 6 1 2 3 6 3
## [75] 1 4 5 5 4 6 6 4 5 2 6 2 6 6 1 6 2 2 3 6 4 2 6 4 6 6
```

```
sample(1:6, size=100, replace=TRUE) %>% table()
```

```
## .
## 1 2 3 4 5 6
## 22 18 17 14 18 11
```

```
sample(1:6, size=100, replace=TRUE) %>% table() %>% prop.table()
```

```
## .
## 1 2 3 4 5 6
## 0.23 0.15 0.12 0.18 0.21 0.11
```

4.2 A keen way to divide up a dataset into testing and training components.

```
x <- 1:10
y <- 11:30
```

```
df <- data.frame(x,y)
df

##      x  y
## 1   1 11
## 2   2 12
## 3   3 13
## 4   4 14
## 5   5 15
## 6   6 16
## 7   7 17
## 8   8 18
## 9   9 19
## 10 10 20
## 11  1 21
## 12  2 22
## 13  3 23
## 14  4 24
## 15  5 25
## 16  6 26
## 17  7 27
## 18  8 28
## 19  9 29
## 20 10 30

set.seed(0)
train_indexes = sample(1:nrow(df), .7 * nrow(df))

train_set <- df[train_indexes,]
test_set <- df[-train_indexes,]

train_set

##      x  y
## 14  4 24
##  4  4 14
##  7  7 17
##  1  1 11
##  2  2 12
## 13  3 23
## 18  8 28
## 11  1 21
## 16  6 26
## 15  5 25
##  3  3 13
## 17  7 27
```

4.2. A KEEN WAY TO DIVIDE UP A DATASET INTO TESTING AND TRAINING COMPONENTS.19

```
## 5  5 15
```

```
## 8  8 18
```

```
test_set
```

```
##      x  y
```

```
## 6    6 16
```

```
## 9    9 19
```

```
## 10   10 20
```

```
## 12    2 22
```

```
## 19    9 29
```

```
## 20   10 30
```


Chapter 5

Factor Practice

```
cups <- c("small", "medium", "large")
manyCups <- sample(cups, size = 100, replace = TRUE)
sizesCups <- factor(manyCups, levels = c("small", "medium", "large"))
sizesCups
```

```
## [1] medium medium medium medium large small large small small small
## [11] small medium small small medium medium medium small large small
## [21] large medium medium medium medium large medium small large medium
## [31] small small large medium medium large large medium medium medium
## [41] medium small medium medium medium medium small large large medium
## [51] large large medium large large small small small small large
## [61] medium large small small medium small small small small large
## [71] medium small small large large large medium medium medium large
## [81] medium medium large large large small medium medium small large
## [91] large medium large medium small medium small large large small
## Levels: small medium large
```

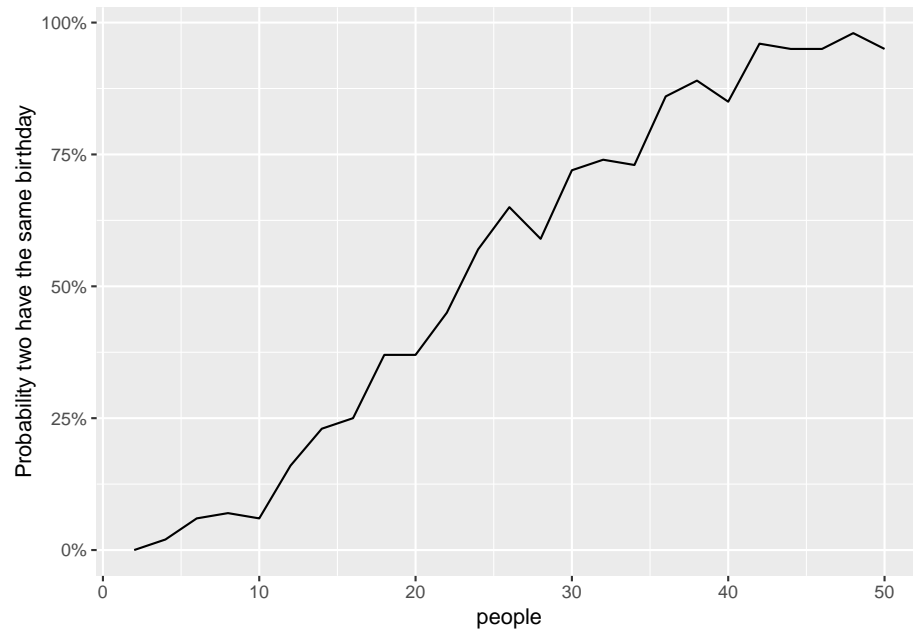

Chapter 6

Crossing Trial

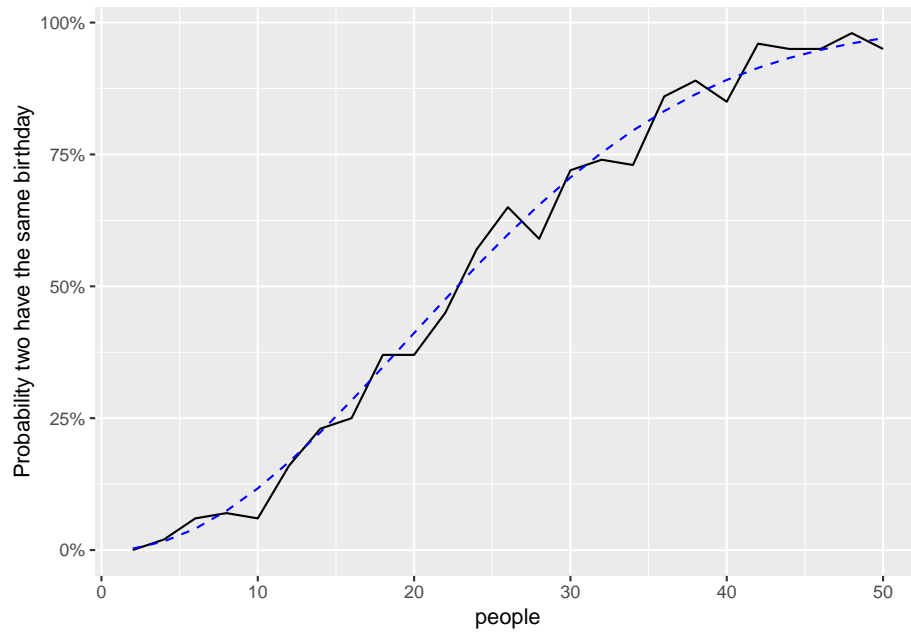
From David Robinson birthday paradox Rblogger at <https://www.r-bloggers.com/the-birthday-paradox-puzzle-tidy-simulation-in-r/>

```
summarized <- crossing(people = seq(2, 50, 2),
                      trial = 1:100) %>%
  mutate(birthday = map(people, ~ sample(365, .x, replace = TRUE)),
         multiple = map_lgl(birthday, ~ any(duplicated(.x)))) %>%
  group_by(people) %>%
  summarize(chance = mean(multiple))

ggplot(summarized, aes(people, chance)) +
  geom_line() +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(y = "Probability two have the same birthday")
```



```
# Checking the work with pbirthday function
summarized %>%
  mutate(exact = map_dbl(people, pbirthday)) %>%
  ggplot(aes(people, chance)) +
  geom_line() +
  geom_line(aes(y = exact), lty = 2, color = "blue") +
  scale_y_continuous(labels = scales::percent_format()) +
  labs(y = "Probability two have the same birthday")
```

Chapter 7

By Any Other Name

This deceptively simple-seeming idea gets complex quickly. The following YouTube was a nice description of the process: <https://www.youtube.com/watch?v=Okc0IL5uTnA>

```
my.data <- data.frame(colOne=1:3, column2=4:6, column_3=7:9)
rownames(my.data) <- c("ant", "bee", "cat")
names(my.data)
```

```
## [1] "colOne" "column2" "column_3"
```

```
colnames(my.data)
```

```
## [1] "colOne" "column2" "column_3"
```

```
#make some changes
```

```
names(my.data) <- c("col_1", "col_2", "col_3")
my.data
```

```
##      col_1 col_2 col_3
## ant      1     4     7
## bee      2     5     8
## cat      3     6     9
```

```
names(my.data)[3] <- "col.3"
my.data
```

```
##      col_1 col_2 col.3
## ant      1     4     7
## bee      2     5     8
## cat      3     6     9
```

```
names(my.data)[names(my.data)=="col_2"]
```

```
## [1] "col_2"
my.data["col_2"]

##      col_2
## ant      4
## bee      5
## cat      6
my.data$col_2

## [1] 4 5 6
my.data[,2]

## [1] 4 5 6
names(my.data)[names(my.data)=="col_2"] <- "col.2"
my.data

##      col_1 col.2 col.3
## ant      1      4      7
## bee      2      5      8
## cat      3      6      9
names(my.data) <- gsub("_", ".", names(my.data))
my.data

##      col.1 col.2 col.3
## ant      1      4      7
## bee      2      5      8
## cat      3      6      9
rownames(my.data)

## [1] "ant" "bee" "cat"
my.data$species <- rownames(my.data)
my.data

##      col.1 col.2 col.3 species
## ant      1      4      7      ant
## bee      2      5      8      bee
## cat      3      6      9      cat
rownames(my.data) <- NULL
my.data

##      col.1 col.2 col.3 species
## 1      1      4      7      ant
## 2      2      5      8      bee
```

```
## 3      3      6      9      cat
```

```
colnames(my.data) <- c("good", "better", "best", "species")
my.data
```

```
##   good better best species
## 1     1      4    7      ant
## 2     2      5    8      bee
## 3     3      6    9      cat
```

```
keep <- 2:ncol(my.data)
my.data[,keep]
```

```
##   better best species
## 1      4    7      ant
## 2      5    8      bee
## 3      6    9      cat
```


Chapter 8

Correlation Plots

```
head(iris)
```

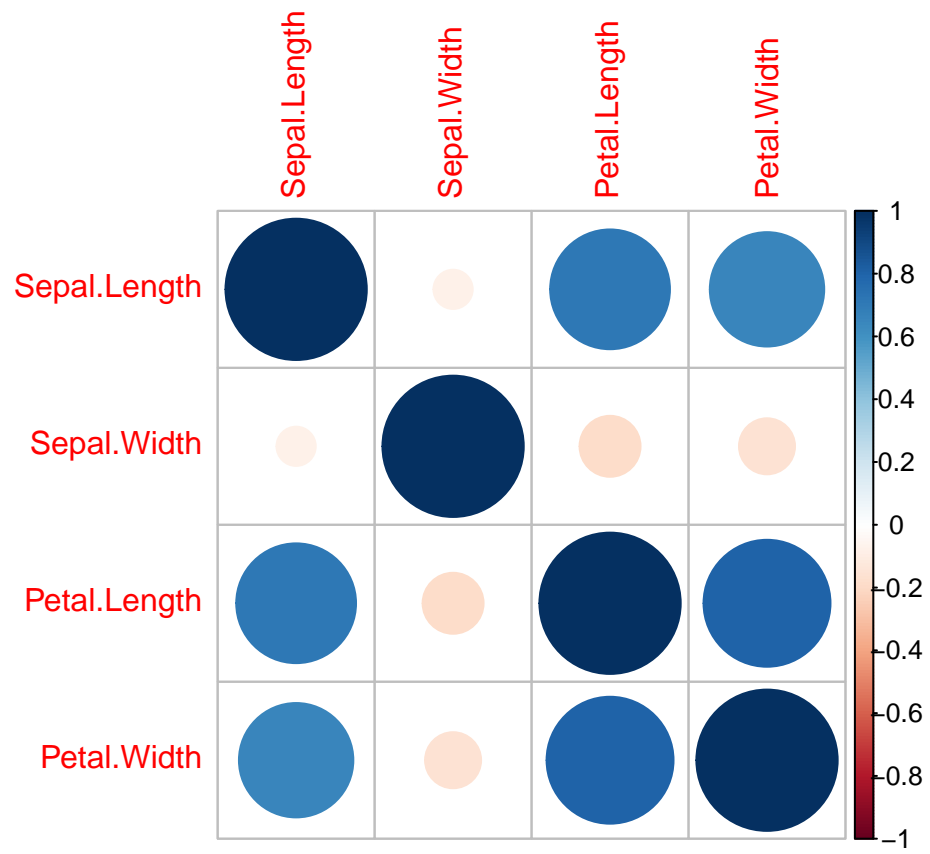
```
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1         5.1         3.5         1.4         0.2   setosa
## 2         4.9         3.0         1.4         0.2   setosa
## 3         4.7         3.2         1.3         0.2   setosa
## 4         4.6         3.1         1.5       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

```
iris %>% select(-Species) %>% cor()
```

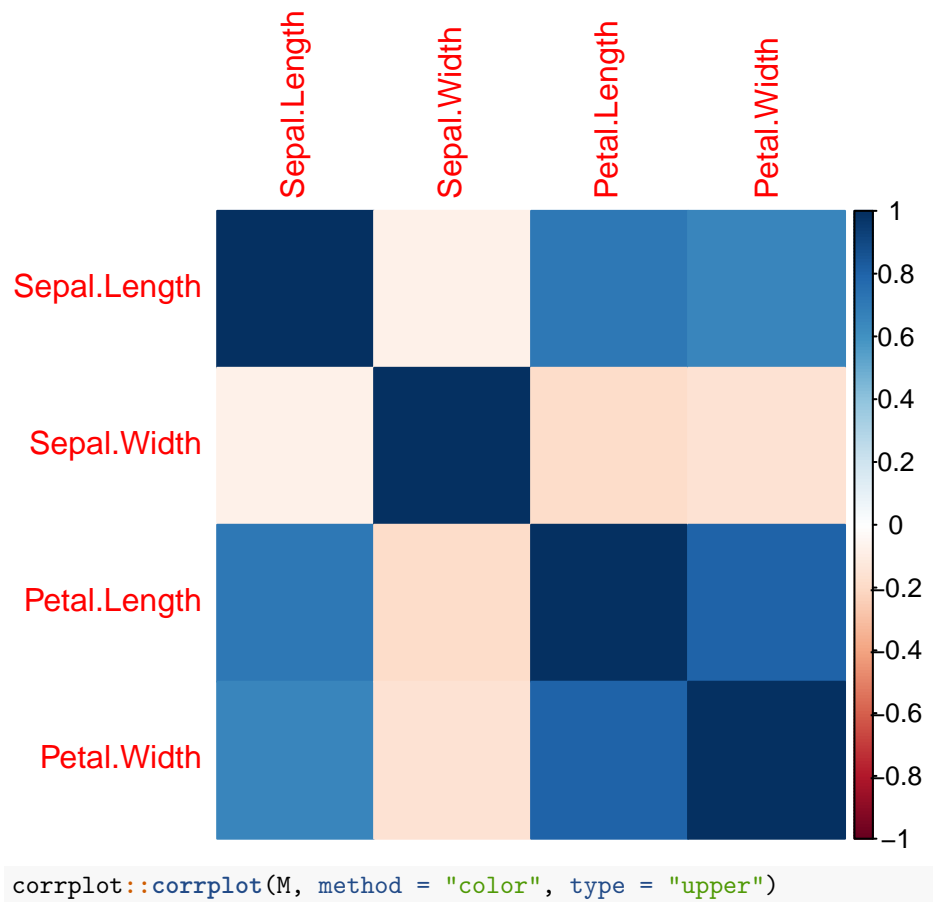
```
##           Sepal.Length Sepal.Width Petal.Length Petal.Width
## Sepal.Length    1.0000000 -0.1175698  0.8717538  0.8179411
## Sepal.Width     -0.1175698  1.0000000 -0.4284401 -0.3661259
## Petal.Length     0.8717538 -0.4284401  1.0000000  0.9628654
## Petal.Width      0.8179411 -0.3661259  0.9628654  1.0000000
```

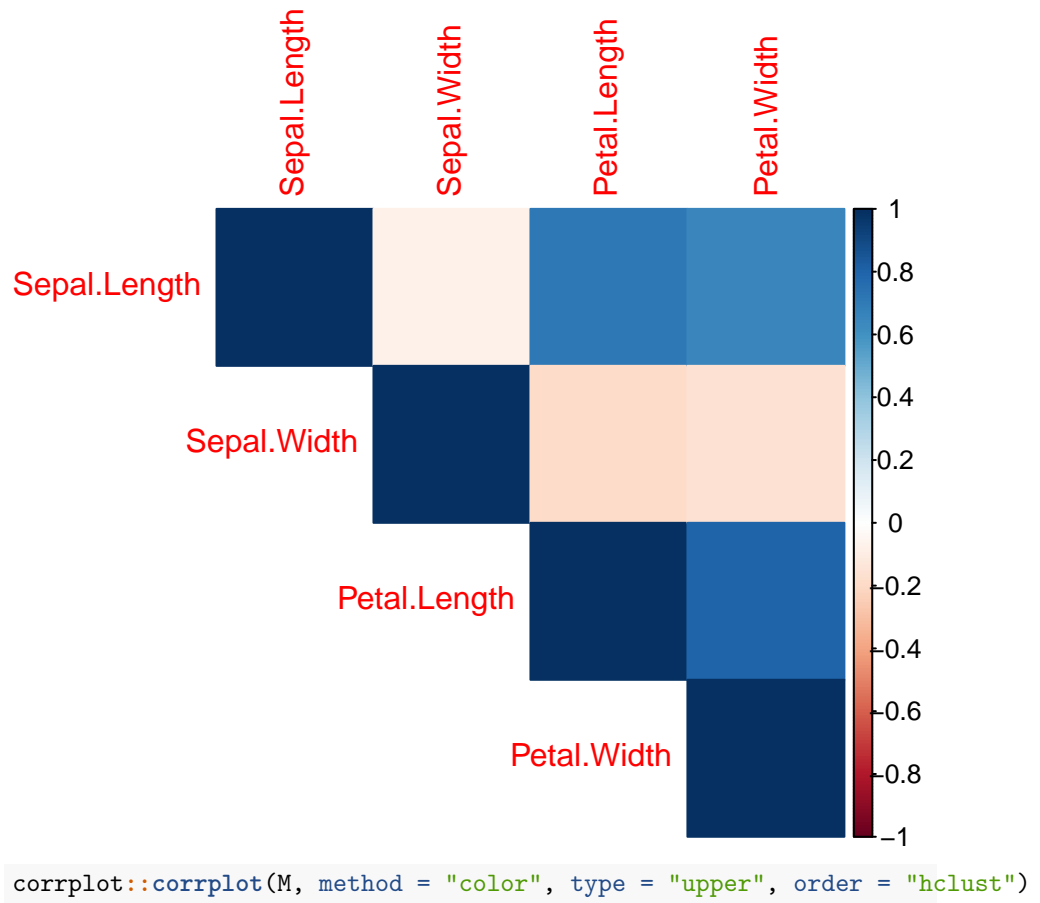
```
M <- iris %>% select(-Species) %>% cor(method = "kendall")
```

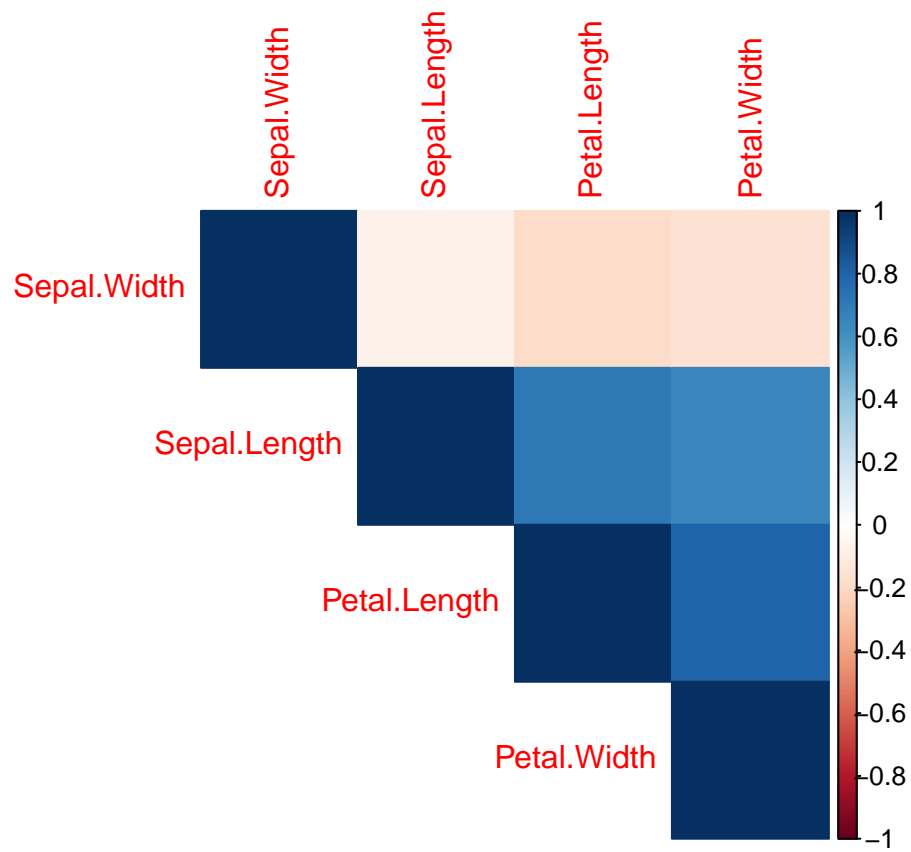
```
corrplot::corrplot(M)
```



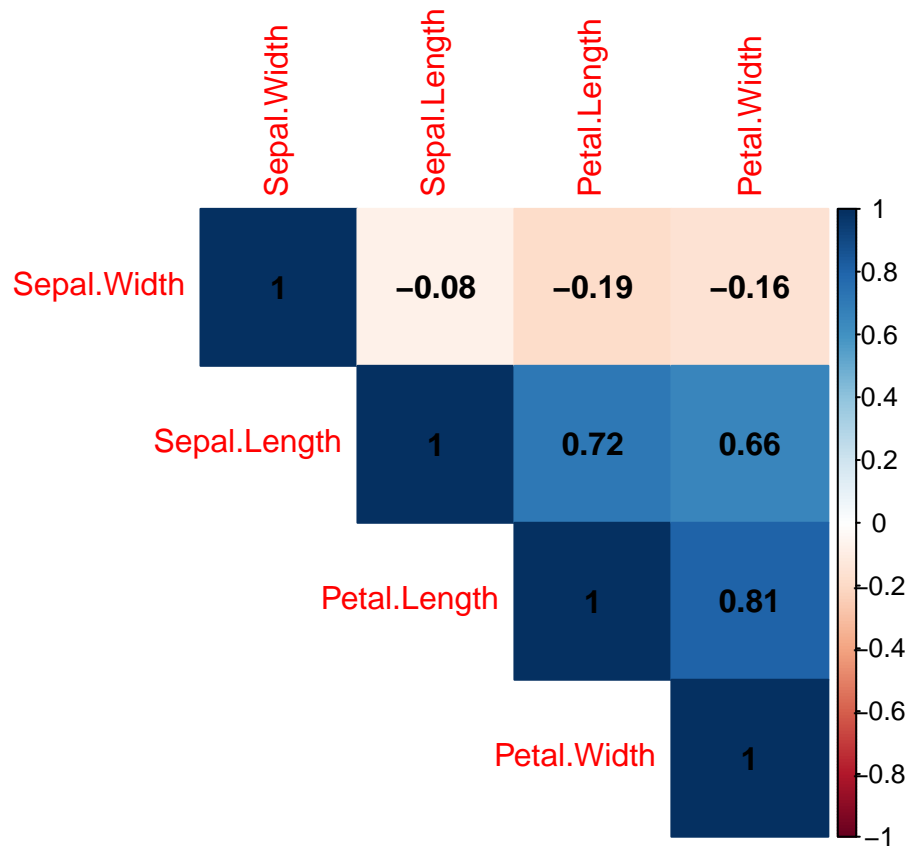
```
corrplot::corrplot(M, method = "color")
```

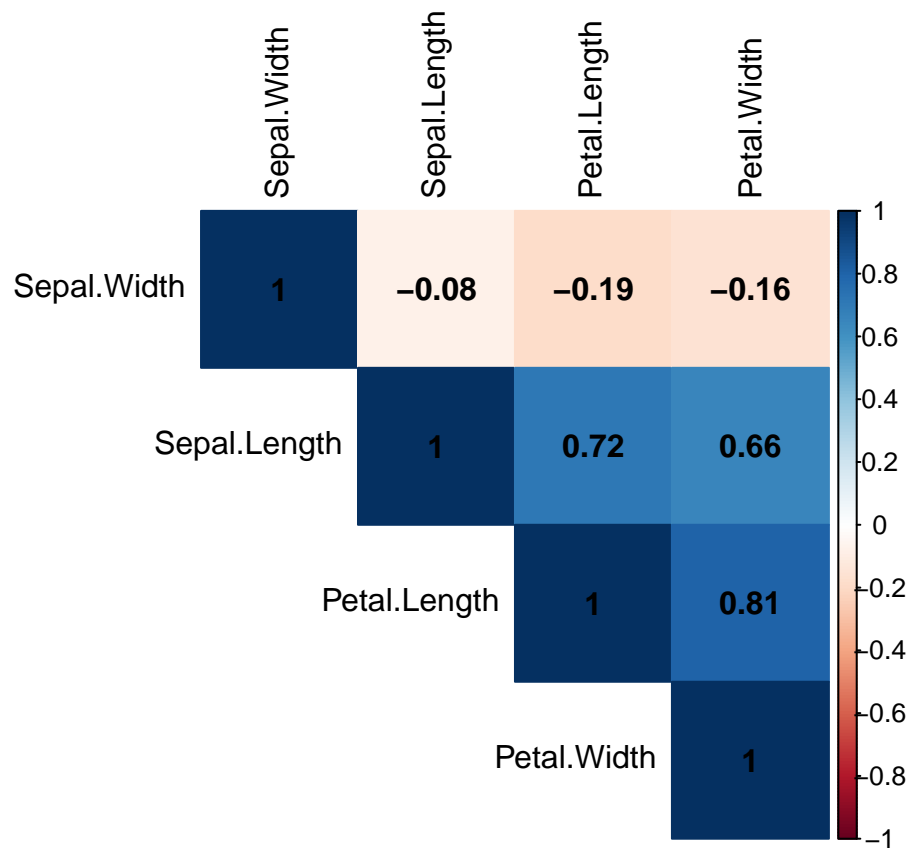




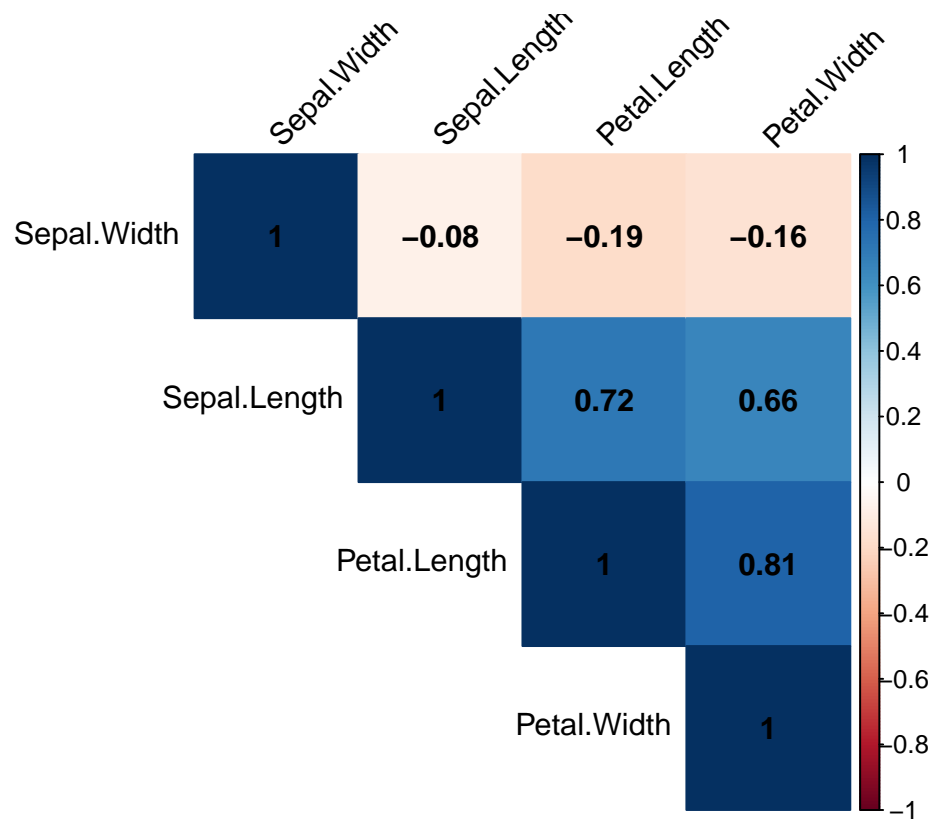
```
corrplot::corrplot(M, method = "color", type = "upper", order = "hclust", addCoef.col = "black")
```



```
corrplot::corrplot(M, method = "color", type = "upper", order = "hclust", addCoef.col = TRUE)
```



```
corrplot::corrplot(M, method = "color", type = "upper", order = "hclust", addCoef.col = "black",
```



Chapter 9

if__else() and case__when(): Comparison

9.1 case__when()

case_when() from https://www.rdocumentation.org/packages/dplyr/versions/0.7.8/topics/case_when

```
x <- 1:50  
y <- 51:100
```

```
df <- data.frame(x,y)  
df
```

```
##      x  y  
## 1    1 51  
## 2    2 52  
## 3    3 53  
## 4    4 54  
## 5    5 55  
## 6    6 56  
## 7    7 57  
## 8    8 58  
## 9    9 59  
## 10  10 60  
## 11  11 61  
## 12  12 62  
## 13  13 63  
## 14  14 64  
## 15  15 65
```

```
## 16 16 66
## 17 17 67
## 18 18 68
## 19 19 69
## 20 20 70
## 21 21 71
## 22 22 72
## 23 23 73
## 24 24 74
## 25 25 75
## 26 26 76
## 27 27 77
## 28 28 78
## 29 29 79
## 30 30 80
## 31 31 81
## 32 32 82
## 33 33 83
## 34 34 84
## 35 35 85
## 36 36 86
## 37 37 87
## 38 38 88
## 39 39 89
## 40 40 90
## 41 41 91
## 42 42 92
## 43 43 93
## 44 44 94
## 45 45 95
## 46 46 96
## 47 47 97
## 48 48 98
## 49 49 99
## 50 50 100
```

```
case_when(
  x %% 35 == 0 ~ "fizz buzz",
  x %% 5 == 0 ~ "fizz",
  x %% 7 == 0 ~ "buzz",
  TRUE ~ as.character(x)
)
```

```
## [1] "1"      "2"      "3"      "4"      "fizz"   "6"
## [7] "buzz"   "8"      "9"      "fizz"   "11"     "12"
## [13] "13"     "buzz"   "fizz"   "16"     "17"     "18"
```



```
## [19] "19"      "fizz"      "buzz"      "22"      "23"      "24"
## [25] "fizz"     "26"      "27"      "buzz"     "29"      "fizz"
## [31] "31"      "32"      "33"      "34"      "fizz buzz" "36"
## [37] "37"      "38"      "39"      "fizz"     "41"      "buzz"
## [43] "43"      "44"      "fizz"     "46"      "47"      "48"
## [49] "buzz"     "fizz"
```

9.2 Compare this with if_else()

```
if_else(x %% 2 == 0, "even", "odd")
```

```
## [1] "odd" "even" "odd" "even" "odd" "even" "odd" "even" "odd" "even"
## [11] "odd" "even" "odd" "even" "odd" "even" "odd" "even" "odd" "even"
## [21] "odd" "even" "odd" "even" "odd" "even" "odd" "even" "odd" "even"
## [31] "odd" "even" "odd" "even" "odd" "even" "odd" "even" "odd" "even"
## [41] "odd" "even" "odd" "even" "odd" "even" "odd" "even" "odd" "even"
```


Chapter 10

Subsetting

From <https://www.r-bloggers.com/5-ways-to-subset-a-data-frame-in-r/>

Note: since this is down for maintenance, I will turn off evaluation on these chunks:

```
education <- read.csv("https://vincentarelbundock.github.io/Rdatasets/csv/robustbase/education.csv")
colnames(education) <- c("X", "State", "Region", "Urban.Population", "Per.Capita.Income", "Minor.Population", "Education.Expenditures")
glimpse(education)
```

```
## Rows: 50
## Columns: 7
## $ X               <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 1...
## $ State           <chr> "ME", "NH", "VT", "MA", "RI", "CT", "NY", "N...
## $ Region          <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2,...
## $ Urban.Population <int> 508, 564, 322, 846, 871, 774, 856, 889, 715,...
## $ Per.Capita.Income <int> 3944, 4578, 4011, 5233, 4780, 5889, 5663, 57...
## $ Minor.Population <int> 325, 323, 328, 305, 303, 307, 301, 310, 300,...
## $ Education.Expenditures <int> 235, 231, 270, 261, 300, 317, 387, 285, 300,...
```

10.1 Subsetting using brackets

```
education[c(10:21), c(2, 6:7)]
```

```
##      State Minor.Population Education.Expenditures
## 10    OH                324                    221
## 11    IN                329                    264
## 12    IL                320                    308
```

```
## 13    MI                337                379
## 14    WI                328                342
## 15    MN                330                378
## 16    IA                318                232
## 17    MO                309                231
## 18    ND                333                246
## 19    SD                330                230
## 20    NB                318                268
## 21    KS                304                337
```

10.2 Subset using brackets by omitting the rows and columns we don't want

```
education[-c(1:9,22:50),-c(1,3:5)]
```

```
##      State Minor.Population Education.Expenditures
## 10     OH                324                221
## 11     IN                329                264
## 12     IL                320                308
## 13     MI                337                379
## 14     WI                328                342
## 15     MN                330                378
## 16     IA                318                232
## 17     MO                309                231
## 18     ND                333                246
## 19     SD                330                230
## 20     NB                318                268
## 21     KS                304                337
```

10.3 Subset using brackets in combination with the which() function and the %in% operator

```
education[which(education$Region == 2),names(education) %in% c("State","Minor.Population")]
```

```
##      State Minor.Population Education.Expenditures
## 10     OH                324                221
## 11     IN                329                264
## 12     IL                320                308
## 13     MI                337                379
```

## 14	WI	328	342
## 15	MN	330	378
## 16	IA	318	232
## 17	MO	309	231
## 18	ND	333	246
## 19	SD	330	230
## 20	NB	318	268
## 21	KS	304	337

10.4 Subset using the subset() function

```
subset(education, Region == 2, select = c("State", "Minor.Population", "Education.Expenditures"))
```

##	State	Minor.Population	Education.Expenditures
## 10	OH	324	221
## 11	IN	329	264
## 12	IL	320	308
## 13	MI	337	379
## 14	WI	328	342
## 15	MN	330	378
## 16	IA	318	232
## 17	MO	309	231
## 18	ND	333	246
## 19	SD	330	230
## 20	NB	318	268
## 21	KS	304	337

10.5 Subset using dplyr's filter() and select()

```
select(filter(education, Region == 2), c(State, Minor.Population, Education.Expenditures))
```

##	State	Minor.Population	Education.Expenditures
## 1	OH	324	221
## 2	IN	329	264
## 3	IL	320	308
## 4	MI	337	379
## 5	WI	328	342
## 6	MN	330	378
## 7	IA	318	232
## 8	MO	309	231
## 9	ND	333	246
## 10	SD	330	230

## 11	NB	318	268
## 12	KS	304	337

Bibliography

Stevens, T. M. and Parekh, V. (2016). Mammary analogue secretory carcinoma. *Arch Pathol Lab Med.*, 140(9):997–1001. doi: 10.5858/arpa.2015-0075-RS.

Xie, Y. (2020). *bookdown: Authoring Books and Technical Documents with R Markdown*. R package version 0.18.