TUT 7

Uzair Mirza

07/03/2022

Introduction

This week's lecture we will be discussing topics from Ch:17. Chapter 17 is about **Tidying Data**. All the problems being discussed can be found on the PASIAS here

Q.17.22 Feeling the heat

There were heat alerts in Toronto and the city decided to open the pools and cooling stuff for extra hours. The dataset we have has all the heat alerts from 2001 to 2016.

a. Read the data and observe the variables

```r
my_url <- "http://ritsokiguess.site/datafiles/heat.csv"
heat <- read_csv(my_url)
```

```
## Rows: 200 Columns: 4-- Column specification ---------------------------------------------------------
## Delimiter: ","
## chr  (2): code, text
## dbl  (1): id
## date (1): date
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
view(heat)
```

Specefically in this dataset the new types of data is `date`.
Note the data file contains the dates in **year-month-day** order hence read_csv is able to recognise this.
Having dates in `date` format allows us to work with them in an easier way(we can specify ranges and etc.)

(if the source file did not follow the same format(**year-month-day**) `lubridate` could be used to read in the dates as dates)

b. Get the count for each `code`

```
heat %>% group_by(code) %>% summarise(n = n()) -> a
a
```

```
## # A tibble: 6 x 2
##   code      n
##   <chr> <int>
## 1 EHA      59
## 2 EHAD      1
## 3 EHAE     18
## 4 HA       93
## 5 HAE      16
## 6 HAU      13
```

What each code means:

```
code <- a$code
meaning <- c("(Start of) Extended Heat Alert", "Extreme Heat Alert downgraded to Heat Alert", "Extended

codeXmeaning <-tibble(code, meaning)
codeXmeaning
```

```
## # A tibble: 6 x 2
##   code  meaning
##   <chr> <chr>
## 1 EHA   (Start of) Extended Heat Alert
## 2 EHAD  Extreme Heat Alert downgraded to Heat Alert
## 3 EHAE  Extended Heat Alert continues
## 4 HA    (Start of) Heat Alert
## 5 HAE   Heat Alert continues
## 6 HAU   Heat Alert upgraded to Extended Heat Alert
```

c. How many (regular and extended) heat alert events are there altogether? A heat alert event is a stretch of consecutive days, on all of which there is a heat alert or extended heat alert

Essentially we are trying to find when there was a change in `code` status for our recorded dates ie Event starts specefic generated code is issued, event ends the specefic generated code is generated.
So we are trying to find the case when the diffrence between these code changes was more than 1.

`as.numeric(date)` -> counts the number of days it has been since a specefic date(some date in 1970)
`diff()` -> takes a diffrence st; given tupple x our result y is generated such as. y[i] = x[i+1] - x[i].
For the last value we compare it with itself, hence diff is always 0.

```
heat %>%
  select(-text) %>%
  mutate(daycount = as.numeric(date)) %>%
  mutate(daydiff = abs(c(diff(daycount), 0))) %>%
  count(daydiff != 1)
```

```
## # A tibble: 2 x 2
##   `daydiff != 1`     n
##   <lgl>           <int>
## 1 FALSE             121
## 2 TRUE               79
```

d. We are going to investigate how many heat alert days there were in each year.

To do that, we have to extract the year from each of our dates and work with the data

Library `lubridate` is used to work with the dates here.
Function `year()` extracts the year from the date and stores it an `dbl`

```
# the library needed for the function call year()
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
# this gets us the date
heat %>% select(-text) %>% mutate(year = year(date))
```

```
## # A tibble: 200 x 4
##       id date       code   year
##    <dbl> <date>     <chr> <dbl>
##  1   232 2016-09-08 HAU    2016
##  2   231 2016-09-07 HAE    2016
##  3   230 2016-09-06 HA     2016
##  4   228 2016-08-13 EHAE   2016
##  5   227 2016-08-12 EHAE   2016
##  6   226 2016-08-11 HAU    2016
##  7   225 2016-08-10 HAE    2016
##  8   224 2016-08-09 HA     2016
##  9   222 2016-08-05 HAE    2016
## 10   221 2016-08-04 HA     2016
## # ... with 190 more rows
```

```
# now we count each of these Unique years
heat %>% select(-text) %>% mutate(year = year(date)) %>% count(year)
```

```
## # A tibble: 16 x 2
##     year     n
##    <dbl> <int>
##  1  2001     9
##  2  2002    16
##  3  2003     6
##  4  2004     2
## 5  2005    26
## 6  2006    17
## 7  2007    15
## 8  2008     9
## 9  2009     3
## 10 2010    16
## 11 2011    12
## 12 2012    21
## 13 2013    13
## 14 2014     1
## 15 2015    12
## 16 2016    22
```

17.15 Ethanol and sleep time in rats

Effect of Ethanol on sleep time in rats. So we have 20 rats, 5 for each group of ethanol; we have data for 3 different ethanol levels and 1 control group.

a. Read and observe the format of the data

```
my_url <- "http://ritsokiguess.site/datafiles/ratsleep.txt"
sleep1 <- read_delim(my_url, " ")
```

```
## Rows: 4 Columns: 6-- Column specification -------------------------------------------------------
## Delimiter: " "
## chr (1): treatment
## dbl (5): obs1, obs2, obs3, obs4, obs5
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
view(sleep1)
```

So here each observation in the table has the value of the sleep time of the mice.
Note it is not really useful in terms of testing when working with data in this format. Instead let us try to improve on it to get in in a format to carry out tests and make plots.

b. Transform this data frame into one that you could use for modelling or making graphs.

GOAL: get a col so we have the sleep time for each observation under a specefic variable `sleeptime` .
`pivot_longer()` is the function to use. ‘it takes in a WIDE data frame and makes it LONGER; reduces the number of col and increases the number of rows

```
(sleep1 %>%
  pivot_longer(!treatment, names_to="rep", values_to="sleeptime") -> sleep)
```

```
## # A tibble: 20 x 3
##    treatment rep   sleeptime
##    <chr>     <chr>     <dbl>
##  1 e0        obs1       88.6
##  2 e0        obs2       73.2
##  3 e0        obs3       91.4
##  4 e0        obs4       68
##  5 e0        obs5       75.2
##  6 e1        obs1       63
##  7 e1        obs2       53.9
##  8 e1        obs3       69.2
##  9 e1        obs4       50.1
## 10 e1        obs5       71.5
## 11 e2        obs1       44.9
## 12 e2        obs2       59.5
## 13 e2        obs3       40.2
## 14 e2        obs4       56.3
## 15 e2        obs5       38.7
## 16 e4        obs1       31
## 17 e4        obs2       39.6
## 18 e4        obs3       45.3
## 19 e4        obs4       25.2
## 20 e4        obs5       22.7
```

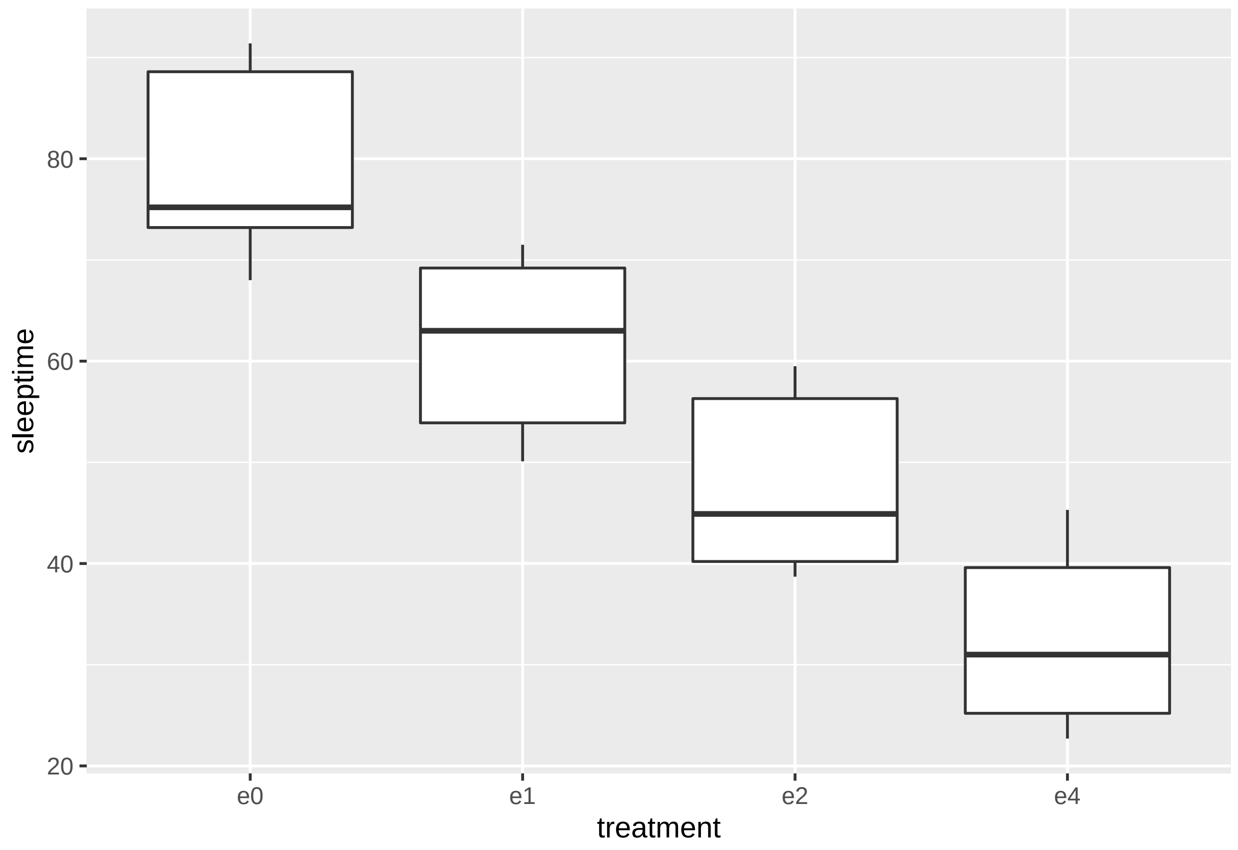
```
(sleep1 %>%
  pivot_longer(-treatment, names_to="rep", values_to="sleeptime") -> sleep)
```

```
## # A tibble: 20 x 3
##    treatment rep   sleeptime
##    <chr>     <chr>     <dbl>
##  1 e0        obs1       88.6
##  2 e0        obs2       73.2
##  3 e0        obs3       91.4
##  4 e0        obs4       68
##  5 e0        obs5       75.2
##  6 e1        obs1       63
##  7 e1        obs2       53.9
##  8 e1        obs3       69.2
##  9 e1        obs4       50.1
## 10 e1        obs5       71.5
## 11 e2        obs1       44.9
## 12 e2        obs2       59.5
## 13 e2        obs3       40.2
## 14 e2        obs4       56.3
## 15 e2        obs5       38.7
## 16 e4        obs1       31
## 17 e4        obs2       39.6
## 18 e4        obs3       45.3
## 19 e4        obs4       25.2
## 20 e4        obs5       22.7
```

`names_to=` -> the col being concacinated into the new varible
`value_to=` -> assigning to the new variable.

c. Using your new data frame, make side-by-side boxplots of sleep time by treatment group.

```
ggplot(sleep, aes(x = treatment, y = sleeptime)) + geom_boxplot()
```



d. In your boxplots, how does the median sleep time appear to depend on treatment group?

Differs between the groups.

17.18 Location, species and disease in plants

Reading the table and what the variables are.
2 Species, A and B
Plausiable location for the disease X and Y
Disease present or absent P and A.

```
my_url <- "http://ritsokiguess.site/datafiles/disease.txt"
tbl <- read_table(my_url)
```

```
##
## -- Column specification --------------------------------------------------------
## cols(
##   Species = col_character(),
##   px = col_double(),
##   py = col_double(),
##   ax = col_double(),
##   ay = col_double()
## )
```

```
tbl
```

```
## # A tibble: 2 x 5
##   Species    px    py    ax    ay
##   <chr>   <dbl> <dbl> <dbl> <dbl>
## 1 A          44    12    38    10
## 2 B          28    22    20    18
```

Lets make it tidy with the goal to make sense of the data.
GOAL: We want to observe the frequencies in each catogery. So lets start with creating that variable.

```
tbl %>% pivot_longer(-Species, names_to="disloc", values_to = "frequency") -> tbl.2
tbl.2
```

```
## # A tibble: 8 x 3
##   Species disloc frequency
##   <chr>   <chr>      <dbl>
## 1 A       px            44
## 2 A       py            12
## 3 A       ax            38
## 4 A       ay            10
## 5 B       px            28
## 6 B       py            22
## 7 B       ax            20
## 8 B       ay            18
```

Lets improve on it.
GOAL: split `disloc` into 2 variables which can account for whether the disease is present or absent and the location of the disease.
We notice it is a chracter of len 2, hence seperation based on the first chracter will do the job.
`seprate(variable, c("new_a","new_b"), c)`. Here the `variable` is the variable in the df we are splitting, `c("new_a","new_b")` are the new variables it is being split into. `c` is the length of the spliting point for the variable.

```
(tbl.2 %>% separate(disloc, c("disease", "location"), 1) -> tbl.3)
```

```
## # A tibble: 8 x 4
##   Species disease location frequency
##   <chr>   <chr>   <chr>        <dbl>
## 1 A       p       x               44
## 2 A       p       y               12
## 3 A       a       x               38
## 4 A       a       y               10
## 5 B        p        x                 28
## 6 B        p        y                 22
## 7 B        a        x                 20
## 8 B        a        y                 18
```