Lab Week 1

STAT5003

Contents

1 Data structure

Before you start the tutorial, *create a new RStudio project* in a new directory. Then, start your tutorial by writing your answers in a new *R Markdown file*.

2 File I/O

2.1 Read

a) Download the `Cereal.csv` file from the Canvas page and use the `read.csv` command to read in the csv file into `R` and assign it to the object called `cereal`.

Solution

```
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
```

```
##
##     intersect, setdiff, setequal, union
```

```r
library(ggplot2) #tidyverse packages
cereal <- read.csv("Cereal.csv", header = TRUE)
#cereal_df #in HTML file, prints out full data frame - Not ideal for a report
cereal_tbl<- read_csv("Cereal.csv")
```

```
## Rows: 77 Columns: 16

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): name, mfr, type
## dbl (13): calories, protein, fat, sodium, fiber, carbo, sugars, potass, vita...
##
## 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
cereal_tbl #in HTML file, only prints out first 10 rows - Gives a snapshot
```

```
## # A tibble: 77 x 16
##    name       mfr   type  calor~1 protein   fat sodium fiber carbo sugars potass
##    <chr>      <chr> <chr>   <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>
##  1 100%_Bran  N     C          70       4     1    130  10     5        6    280
##  2 100%_Natu~ Q     C         120       3     5     15   2     8        8    135
##  3 All-Bran   K     C          70       4     1    260   9     7        5    320
##  4 All-Bran_~ K     C          50       4     0    140  14     8        0    330
##  5 Almond_De~ R     C         110       2     2    200   1    14        8     -1
##  6 Apple_Cin~ G     C         110       2     2    180   1.5  10.5     10     70
##  7 Apple_Jac~ K     C         110       2     0    125   1    11       14     30
##  8 Basic_4    G     C         130       3     2    210   2    18        8    100
##  9 Bran_Chex  R     C          90       2     1    200   4    15        6    125
## 10 Bran_Flak~ P     C          90       3     0    210   5    13        5    190
## # ... with 67 more rows, 5 more variables: vitamins <dbl>, shelf <dbl>,
## #   weight <dbl>, cups <dbl>, rating <dbl>, and abbreviated variable name
## #   1: calories
```

2.2 Data frames

a) There should be a default dataset in R called `cereal`. Use the `head` function to inspect the first few lines of the data frame and use `class` to check that `cereal` is in fact a data frame.

Solution

```r
# Base R
head(cereal)
```

```
##                         name mfr type calories protein fat sodium fiber carbo
## 1                 100%_Bran   N    C       70       4   1    130  10.0   5.0
## 2         100%_Natural_Bran   Q    C      120       3   5     15   2.0   8.0
## 3                  All-Bran   K    C       70       4   1    260   9.0   7.0
## 4 All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## 5            Almond_Delight   R    C      110       2   2    200   1.0  14.0
## 6   Apple_Cinnamon_Cheerios   G    C      110       2   2    180   1.5  10.5
##   sugars potass vitamins shelf weight cups   rating
## 1      6    280       25     3      1 0.33 68.40297
## 2      8    135        0     3      1 1.00 33.98368
## 3      5    320       25     3      1 0.33 59.42551
## 4      0    330       25     3      1 0.50 93.70491
## 5      8     -1       25     3      1 0.75 34.38484
## 6     10     70       25     1      1 0.75 29.50954
```

```
cereal_tbl %>% head(7) # using pipe
```

```
## # A tibble: 7 x 16
##   name        mfr   type  calor~1 protein   fat sodium fiber carbo sugars potass
##   <chr>       <chr> <chr>   <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>
## 1 100%_Bran    N     C          70       4     1    130  10      5        6    280
## 2 100%_Natur~ Q     C          120       3     5     15   2      8        8    135
## 3 All-Bran    K     C           70       4     1    260   9      7        5    320
## 4 All-Bran_w~ K     C           50       4     0    140  14      8        0    330
## 5 Almond_Del~ R     C          110       2     2    200   1     14        8     -1
## 6 Apple_Cinn~ G     C          110       2     2    180   1.5  10.5     10     70
## 7 Apple_Jacks K     C          110       2     0    125   1     11       14     30
## # ... with 5 more variables: vitamins <dbl>, shelf <dbl>, weight <dbl>,
## #   cups <dbl>, rating <dbl>, and abbreviated variable name 1: calories
```

```
cereal_tbl %>% class()# tells us this is a tibble
```

```
## [1] "spec_tbl_df" "tbl_df"      "tbl"         "data.frame"
```

```
#class(cereal_tbl)
```

b) What are the column names of the cereal data frame? How many rows are there? (`dim` and `nrow`)

Solution

```
cereal_tbl %>% colnames()
```

```
##  [1] "name"     "mfr"      "type"     "calories" "protein"  "fat"
##  [7] "sodium"   "fiber"    "carbo"    "sugars"   "potass"   "vitamins"
## [13] "shelf"    "weight"   "cups"     "rating"
```

```
cereal_tbl %>% dim()
```

```
## [1] 77 16
```

```
cereal_tbl %>% nrow()
```

```
## [1] 77
```

```
dim(cereal)
```

```
## [1] 77 16
```

```
nrow(cereal)
```

```
## [1] 77
```

c) Extract the `calories` column using the `$` operator and using the `[[` operator.

Solution

```
# Some newer ways
Cal<-cereal_tbl %>% select(calories) #tibble with one column
Cal<-cereal_tbl %>% pull(calories) #pull out the column as vector.

# Base R
AlternativeCal<-cereal[["calories"]]
class(cereal["calories"])
```

```
## [1] "data.frame"
```

```
class(cereal[["calories"]])
```

```
## [1] "integer"
```

d) Extract rows 1 to 10 from the `cereal` data frame.

Solution

```
# Base R
cereal[1:10,]
```

```
##                         name mfr type calories protein fat sodium fiber carbo
## 1                  100%_Bran   N    C       70       4   1    130  10.0   5.0
## 2          100%_Natural_Bran   Q    C      120       3   5     15   2.0   8.0
## 3                   All-Bran   K    C       70       4   1    260   9.0   7.0
## 4  All-Bran_with_Extra_Fiber   K    C       50       4   0    140  14.0   8.0
## 5             Almond_Delight   R    C      110       2   2    200   1.0  14.0
## 6    Apple_Cinnamon_Cheerios   G    C      110       2   2    180   1.5  10.5
## 7                Apple_Jacks   K    C      110       2   0    125   1.0  11.0
## 8                    Basic_4   G    C      130       3   2    210   2.0  18.0
## 9                  Bran_Chex   R    C       90       2   1    200   4.0  15.0
## 10               Bran_Flakes   P    C       90       3   0    210   5.0  13.0
##    sugars potass vitamins shelf weight cups   rating
## 1       6    280       25     3   1.00 0.33 68.40297
## 2       8    135        0     3   1.00 1.00 33.98368
## 3       5    320       25     3   1.00 0.33 59.42551
## 4       0    330       25     3   1.00 0.50 93.70491
## 5       8     -1       25     3   1.00 0.75 34.38484
## 6      10     70       25     1   1.00 0.75 29.50954
## 7      14     30       25     2   1.00 1.00 33.17409
## 8       8    100       25     3   1.33 0.75 37.03856
## 9       6    125       25     1   1.00 0.67 49.12025
## 10      5    190       25     3   1.00 0.67 53.31381
```

```
# Tidyverse
cereal_tbl %>% slice(1:10)
```

```
## # A tibble: 10 x 16
##    name       mfr   type  calor~1 protein   fat sodium fiber carbo sugars potass
##    <chr>      <chr> <chr>   <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>
##  1 100%_Bran  N     C          70       4     1    130  10     5        6    280
##  2 100%_Natu~ Q     C         120       3     5     15   2     8        8    135
##  3 All-Bran   K     C          70       4     1    260   9     7        5    320
##  4 All-Bran_~ K     C          50       4     0    140  14     8        0    330
##  5 Almond_De~ R     C         110       2     2    200   1    14        8     -1
##  6 Apple_Cin~ G     C         110       2     2    180   1.5  10.5       10     70
##  7 Apple_Jac~ K     C         110       2     0    125   1    11       14     30
##  8 Basic_4    G     C         130       3     2    210   2    18        8    100
##  9 Bran_Chex  R     C          90       2     1    200   4    15        6    125
## 10 Bran_Flak~ P     C          90       3     0    210   5    13        5    190
## # ... with 5 more variables: vitamins <dbl>, shelf <dbl>, weight <dbl>,
## #   cups <dbl>, rating <dbl>, and abbreviated variable name 1: calories
```

e) Make a new data frame called `Kelloggs` that only contains rows that belongs to manufacturer, Kelloggs (when `mfr` takes the value `"K"`).

Solution

```
# Base R
Kelloggs <- subset(cereal, mfr == "K")
head(Kelloggs)
```

```
##                          name mfr type calories protein fat sodium fiber carbo
## 3                    All-Bran   K    C       70       4   1    260     9     7
## 4   All-Bran_with_Extra_Fiber   K    C       50       4   0    140    14     8
## 7                 Apple_Jacks   K    C      110       2   0    125     1    11
## 17                Corn_Flakes   K    C      100       2   0    290     1    21
## 18                  Corn_Pops   K    C      110       1   0     90     1    13
## 20         Cracklin'_Oat_Bran   K    C      110       3   3    140     4    10
##    sugars potass vitamins shelf weight cups   rating
## 3       5    320       25     3      1 0.33 59.42551
## 4       0    330       25     3      1 0.50 93.70491
## 7      14     30       25     2      1 1.00 33.17409
## 17      2     35       25     1      1 1.00 45.86332
## 18     12     20       25     2      1 1.00 35.78279
## 20      7    160       25     3      1 0.50 40.44877
```

```
Kelloggs.2 <- cereal[cereal$mfr == "K", -2] #removes 2nd column
head(Kelloggs.2)
```

```
##                          name type calories protein fat sodium fiber carbo
## 3                    All-Bran    C       70       4   1    260     9     7
## 4   All-Bran_with_Extra_Fiber    C       50       4   0    140    14     8
## 7                 Apple_Jacks    C      110       2   0    125     1    11
## 17                Corn_Flakes    C      100       2   0    290     1    21
## 18                  Corn_Pops    C      110       1   0     90     1    13
## 20         Cracklin'_Oat_Bran    C      110       3   3    140     4    10
##    sugars potass vitamins shelf weight cups   rating
## 3       5    320       25     3      1 0.33 59.42551
## 4       0    330       25     3      1 0.50 93.70491
## 7      14     30       25     2      1 1.00 33.17409
## 17      2     35       25     1      1 1.00 45.86332
## 18     12     20       25     2      1 1.00 35.78279
## 20      7    160       25     3      1 0.50 40.44877
```

```
# Base R splitting
cereal.splitted <- split(cereal, cereal$mfr) #list of 7 items based on mfr
Kelloggs.3 <- cereal.splitted["K"]
Kelloggs.4 <- cereal.splitted[["K"]]
identical(Kelloggs, Kelloggs.4)
```

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

```
# Tidyverse way
kelloggs_tbl <- cereal_tbl %>% filter(mfr == "K") #tidy way :)
kelloggs_tbl <- cereal_tbl %>% filter(mfr == "K") %>% select(-mfr) #drop mfr
```

2.3 Factors

a) Load the `Cereal` data again with the `read.csv` command again. This time, use the optional argument, `stringsAsFactors = TRUE`.
b) The `mfr` and `type` columns are now factors. Check that this is true.

Solution

```
cereal <- read_csv("Cereal.csv")
```

```
## Rows: 77 Columns: 16
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (3): name, mfr, type
## dbl (13): calories, protein, fat, sodium, fiber, carbo, sugars, potass, vita...
##
## 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.
```

```
cereal.with.factors <- read.csv("Cereal.csv", stringsAsFactors = TRUE)
cereal.with.factors
```

```
##                                          name mfr type calories protein fat sodium
## 1                                   100%_Bran   N    C       70       4   1    130
## 2                           100%_Natural_Bran   Q    C      120       3   5     15
## 3                                    All-Bran   K    C       70       4   1    260
## 4                   All-Bran_with_Extra_Fiber   K    C       50       4   0    140
## 5                              Almond_Delight   R    C      110       2   2    200
## 6                     Apple_Cinnamon_Cheerios   G    C      110       2   2    180
## 7                                 Apple_Jacks   K    C      110       2   0    125
## 8                                     Basic_4   G    C      130       3   2    210
## 9                                   Bran_Chex   R    C       90       2   1    200
## 10                                Bran_Flakes   P    C       90       3   0    210
## 11                               Cap'n'Crunch   Q    C      120       1   2    220
## 12                                   Cheerios   G    C      110       6   2    290
## 13                      Cinnamon_Toast_Crunch   G    C      120       1   3    210
## 14                                   Clusters   G    C      110       3   2    140
## 15                                Cocoa_Puffs   G    C      110       1   1    180
## 16                                  Corn_Chex   R    C      110       2   0    280
## 17                                Corn_Flakes   K    C      100       2   0    290
## 18                                  Corn_Pops   K    C      110       1   0     90
## 19                              Count_Chocula   G    C      110       1   1    180
## 20                         Cracklin'_Oat_Bran   K    C      110       3   3    140
## 21                     Cream_of_Wheat_(Quick)   N    H      100       3   0     80
## 22                                    Crispix   K    C      110       2   0    220
## 23                     Crispy_Wheat_&_Raisins   G    C      100       2   1    140
## 24                                Double_Chex   R    C      100       2   0    190
## 25                                Froot_Loops   K    C      110       2   1    125
## 26                             Frosted_Flakes   K    C      110       1   0    200
## 27                        Frosted_Mini-Wheats   K    C      100       3   0      0
## 28 Fruit_&_Fibre_Dates,_Walnuts,_and_Oats   P    C      120       3   2    160
## 29                              Fruitful_Bran   K    C      120       3   0    240
## 30                             Fruity_Pebbles   P    C      110       1   1    135
## 31                               Golden_Crisp   P    C      100       2   0     45
## 32                             Golden_Grahams   G    C      110       1   1    280
## 33                          Grape_Nuts_Flakes   P    C      100       3   1    140
## 34                                 Grape-Nuts   P    C      110       3   0    170
## 35                         Great_Grains_Pecan   P    C      120       3   3     75
## 36                           Honey_Graham_Ohs   Q    C      120       1   2    220
## 37                         Honey_Nut_Cheerios   G    C      110       3   1    250
## 38                                 Honey-comb   P    C      110       1   0    180
## 39                Just_Right_Crunchy__Nuggets   K    C      110       2   1    170
## 40                 Just_Right_Fruit_&_Nut   K    C      140       3   1    170
## 41                                    Kix   G    C      110       2   1    260
## 42                                   Life   Q    C      100       4   2    150
## 43                           Lucky_Charms   G    C      110       2   1    180
## 44                                  Maypo   A    H      100       4   1      0
## 45       Muesli_Raisins,_Dates,_&_Almonds   R    C      150       4   3     95
## 46      Muesli_Raisins,_Peaches,_&_Pecans   R    C      150       4   3    150
## 47                   Mueslix_Crispy_Blend   K    C      160       3   2    150
## 48                   Multi-Grain_Cheerios   G    C      100       2   1    220
## 49                       Nut&Honey_Crunch   K    C      120       2   1    190
## 50              Nutri-Grain_Almond-Raisin   K    C      140       3   2    220
## 51                      Nutri-grain_Wheat   K    C       90       3   0    170
## 52                   Oatmeal_Raisin_Crisp   G    C      130       3   2    170
## 53                  Post_Nat._Raisin_Bran   P    C      120       3   1    200
## 54                             Product_19   K    C      100       3   0    320
## 55                            Puffed_Rice   Q    C       50       1   0      0
## 56                           Puffed_Wheat   Q    C       50       2   0      0
## 57                     Quaker_Oat_Squares   Q    C      100       4   1    135
## 58                         Quaker_Oatmeal   Q    H      100       5   2      0
## 59                            Raisin_Bran   K    C      120       3   1    210
## 60                        Raisin_Nut_Bran   G    C      100       3   2    140
## 61                         Raisin_Squares   K    C       90       2   0      0
## 62                              Rice_Chex   R    C      110       1   0    240
## 63                          Rice_Krispies   K    C      110       2   0    290
## 64                         Shredded_Wheat   N    C       80       2   0      0
## 65                 Shredded_Wheat_'n'Bran   N    C       90       3   0      0
## 66              Shredded_Wheat_spoon_size   N    C       90       3   0      0
## 67                                 Smacks   K    C      110       2   1     70
## 68                              Special_K   K    C      110       6   0    230
## 69                Strawberry_Fruit_Wheats   N    C       90       2   0     15
## 70                      Total_Corn_Flakes   G    C      110       2   1    200
## 71                      Total_Raisin_Bran   G    C      140       3   1    190
## 72                      Total_Whole_Grain   G    C      100       3   1    200
## 73                                Triples   G    C      110       2   1    250
## 74                                   Trix   G    C      110       1   1    140
## 75                             Wheat_Chex   R    C      100       3   1    230
## 76                               Wheaties   G    C      100       3   1    200
## 77                    Wheaties_Honey_Gold   G    C      110       2   1    200
##    fiber carbo sugars potass vitamins shelf weight cups   rating
## 1   10.0   5.0      6    280       25     3   1.00 0.33 68.40297
## 2    2.0   8.0      8    135        0     3   1.00 1.00 33.98368
## 3    9.0   7.0      5    320       25     3   1.00 0.33 59.42551
## 4   14.0   8.0      0    330       25     3   1.00 0.50 93.70491
## 5    1.0  14.0      8     -1       25     3   1.00 0.75 34.38484
## 6    1.5  10.5     10     70       25     1   1.00 0.75 29.50954
## 7    1.0  11.0     14     30       25     2   1.00 1.00 33.17409
## 8    2.0  18.0      8    100       25     3   1.33 0.75 37.03856
## 9    4.0  15.0      6    125       25     1   1.00 0.67 49.12025
## 10   5.0  13.0      5    190       25     3   1.00 0.67 53.31381
## 11   0.0  12.0     12     35       25     2   1.00 0.75 18.04285
## 12   2.0  17.0      1    105       25     1   1.00 1.25 50.76500
## 13   0.0  13.0      9     45       25     2   1.00 0.75 19.82357
## 14   2.0  13.0      7    105       25     3   1.00 0.50 40.40021
## 15   0.0  12.0     13     55       25     2   1.00 1.00 22.73645
## 16   0.0  22.0       3     25       25      1   1.00 1.00 41.44502
## 17   1.0  21.0       2     35       25      1   1.00 1.00 45.86332
## 18   1.0  13.0      12     20       25      2   1.00 1.00 35.78279
## 19   0.0  12.0      13     65       25      2   1.00 1.00 22.39651
## 20   4.0  10.0       7    160       25      3   1.00 0.50 40.44877
## 21   1.0  21.0       0     -1        0      2   1.00 1.00 64.53382
## 22   1.0  21.0       3     30       25      3   1.00 1.00 46.89564
## 23   2.0  11.0      10    120       25      3   1.00 0.75 36.17620
## 24   1.0  18.0       5     80       25      3   1.00 0.75 44.33086
## 25   1.0  11.0      13     30       25      2   1.00 1.00 32.20758
## 26   1.0  14.0      11     25       25      1   1.00 0.75 31.43597
## 27   3.0  14.0       7    100       25      2   1.00 0.80 58.34514
## 28   5.0  12.0      10    200       25      3   1.25 0.67 40.91705
## 29   5.0  14.0      12    190       25      3   1.33 0.67 41.01549
## 30   0.0  13.0      12     25       25      2   1.00 0.75 28.02576
## 31   0.0  11.0      15     40       25      1   1.00 0.88 35.25244
## 32   0.0  15.0       9     45       25      2   1.00 0.75 23.80404
## 33   3.0  15.0       5     85       25      3   1.00 0.88 52.07690
## 34   3.0  17.0       3     90       25      3   1.00 0.25 53.37101
## 35   3.0  13.0       4    100       25      3   1.00 0.33 45.81172
## 36   1.0  12.0      11     45       25      2   1.00 1.00 21.87129
## 37   1.5  11.5      10     90       25      1   1.00 0.75 31.07222
## 38   0.0  14.0      11     35       25      1   1.00 1.33 28.74241
## 39   1.0  17.0       6     60      100      3   1.00 1.00 36.52368
## 40   2.0  20.0       9     95      100      3   1.30 0.75 36.47151
## 41   0.0  21.0       3     40       25      2   1.00 1.50 39.24111
## 42   2.0  12.0       6     95       25      2   1.00 0.67 45.32807
## 43   0.0  12.0      12     55       25      2   1.00 1.00 26.73451
## 44   0.0  16.0       3     95       25      2   1.00 1.00 54.85092
## 45   3.0  16.0      11    170       25      3   1.00 1.00 37.13686
## 46   3.0  16.0      11    170       25      3   1.00 1.00 34.13976
## 47   3.0  17.0      13    160       25      3   1.50 0.67 30.31335
## 48   2.0  15.0       6     90       25      1   1.00 1.00 40.10596
## 49   0.0  15.0       9     40       25      2   1.00 0.67 29.92429
## 50   3.0  21.0       7    130       25      3   1.33 0.67 40.69232
## 51   3.0  18.0       2     90       25      3   1.00 1.00 59.64284
## 52   1.5  13.5      10    120       25      3   1.25 0.50 30.45084
## 53   6.0  11.0      14    260       25      3   1.33 0.67 37.84059
## 54   1.0  20.0       3     45      100      3   1.00 1.00 41.50354
## 55   0.0  13.0       0     15        0      3   0.50 1.00 60.75611
## 56   1.0  10.0       0     50        0      3   0.50 1.00 63.00565
## 57   2.0  14.0       6    110       25      3   1.00 0.50 49.51187
## 58   2.7  -1.0      -1    110        0      1   1.00 0.67 50.82839
## 59   5.0  14.0      12    240       25      2   1.33 0.75 39.25920
## 60   2.5  10.5       8    140       25      3   1.00 0.50 39.70340
## 61   2.0  15.0       6    110       25      3   1.00 0.50 55.33314
## 62   0.0  23.0       2     30       25      1   1.00 1.13 41.99893
## 63   0.0  22.0       3     35       25      1   1.00 1.00 40.56016
## 64   3.0  16.0       0     95        0      1   0.83 1.00 68.23588
## 65   4.0  19.0       0    140        0      1   1.00 0.67 74.47295
## 66   3.0  20.0       0    120        0      1   1.00 0.67 72.80179
## 67   1.0   9.0      15     40       25      2   1.00 0.75 31.23005
## 68   1.0  16.0       3     55       25      1   1.00 1.00 53.13132
## 69   3.0  15.0       5     90       25      2   1.00 1.00 59.36399
## 70   0.0  21.0      3     35       100     3   1.00 1.00 38.83975
## 71   4.0  15.0     14    230       100     3   1.50 1.00 28.59278
## 72   3.0  16.0      3    110       100     3   1.00 1.00 46.65884
## 73   0.0  21.0      3     60        25     3   1.00 0.75 39.10617
## 74   0.0  13.0     12     25        25     2   1.00 1.00 27.75330
## 75   3.0  17.0      3    115        25     1   1.00 0.67 49.78744
## 76   3.0  17.0      3    110        25     1   1.00 1.00 51.59219
## 77   1.0  16.0      8     60        25     1   1.00 0.75 36.18756
```

```
levels(cereal.with.factors$mfr)
```

```
## [1] "A" "G" "K" "N" "P" "Q" "R"
```

```
class(cereal.with.factors$mfr)
```

```
## [1] "factor"
```

```
class(cereal$mfr)
```

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

```
## or
class(cereal.with.factors$carbo)
```

```
## [1] "numeric"
```

```
class(cereal$carbo)
```

```
## [1] "numeric"
```

```
## or
str(cereal.with.factors) #only characters become factors
```

```
## 'data.frame':    77 obs. of  16 variables:
##  $ name    : Factor w/ 77 levels "100%_Bran","100%_Natural_Bran",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ mfr     : Factor w/ 7 levels "A","G","K","N",..: 4 6 3 3 7 2 3 2 7 5 ...
##  $ type    : Factor w/ 2 levels "C","H": 1 1 1 1 1 1 1 1 1 1 ...
##  $ calories: int  70 120 70 50 110 110 110 130 90 90 ...
##  $ protein : int  4 3 4 4 2 2 2 3 2 3 ...
##  $ fat     : int  1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium  : int  130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber   : num  10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo   : num  5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars  : int  6 8 5 0 8 10 14 8 6 5 ...
##  $ potass  : int  280 135 320 330 -1 70 30 100 125 190 ...
##  $ vitamins: int  25 0 25 25 25 25 25 25 25 25 ...
##  $ shelf   : int  3 3 3 3 3 1 2 3 1 3 ...
##  $ weight  : num  1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups    : num  0.33 1 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ rating  : num  68.4 34 59.4 93.7 34.4 ...
```

```
str(cereal)
```

```
## spc_tbl_ [77 x 16] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ name    : chr [1:77] "100%_Bran" "100%_Natural_Bran" "All-Bran" "All-Bran_with_Extra_Fiber" ...
##  $ mfr     : chr [1:77] "N" "Q" "K" "K" ...
##  $ type    : chr [1:77] "C" "C" "C" "C" ...
##  $ calories: num [1:77] 70 120 70 50 110 110 110 130 90 90 ...
##  $ protein : num [1:77] 4 3 4 4 2 2 2 3 2 3 ...
##  $ fat     : num [1:77] 1 5 1 0 2 2 0 2 1 0 ...
##  $ sodium  : num [1:77] 130 15 260 140 200 180 125 210 200 210 ...
##  $ fiber   : num [1:77] 10 2 9 14 1 1.5 1 2 4 5 ...
##  $ carbo   : num [1:77] 5 8 7 8 14 10.5 11 18 15 13 ...
##  $ sugars  : num [1:77] 6 8 5 0 8 10 14 8 6 5 ...
##  $ potass  : num [1:77] 280 135 320 330 -1 70 30 100 125 190 ...
##  $ vitamins: num [1:77] 25 0 25 25 25 25 25 25 25 25 ...
##  $ shelf   : num [1:77] 3 3 3 3 3 1 2 3 1 3 ...
##  $ weight  : num [1:77] 1 1 1 1 1 1 1 1.33 1 1 ...
##  $ cups    : num [1:77] 0.33 1 0.33 0.5 0.75 0.75 1 0.75 0.67 0.67 ...
##  $ rating  : num [1:77] 68.4 34 59.4 93.7 34.4 ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   name = col_character(),
##   ..   mfr = col_character(),
##   ..   type = col_character(),
##   ..   calories = col_double(),
##   ..   protein = col_double(),
##   ..   fat = col_double(),
##   ..   sodium = col_double(),
##   ..   fiber = col_double(),
##   ..   carbo = col_double(),
##   ..   sugars = col_double(),
##   ..   potass = col_double(),
##   ..   vitamins = col_double(),
##   ..   shelf = col_double(),
##   ..   weight = col_double(),
##   ..   cups = col_double(),
##   ..   rating = col_double()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

b) How many levels are there in `mfr` and `type`? (use the functions `levels` or `nlevels`) - levels will only count FACTORS not CHARACTER strings

Solution

```
levels(cereal.with.factors$mfr)
```

```
## [1] "A" "G" "K" "N" "P" "Q" "R"
```

```
## or
nlevels(cereal.with.factors$mfr)
```

```
## [1] 7
```

```
## or
str(cereal.with.factors$mfr)
```

```
##  Factor w/ 7 levels "A","G","K","N",..: 4 6 3 3 7 2 3 2 7 5 ...
```

```
#class() typeof()
```

2.4 Vectors

a) Extract the `calories` into a new vector called `cereal.calories`.

Solution

```
cereal.calories <- cereal$calories
cereal.calories <- cereal[["calories"]]
cereal_calories <- cereal_tbl %>% pull(calories)
```

b) How many elements are there in `cereal.calories`? (`length`)

Solution

```
length(cereal.calories)
```

```
## [1] 77
```

```
cereal_calories %>% length()
```

```
## [1] 77
```

c) Extract the 5th to the 10th element from `cereal.calories`.

Solution

```
#cereal.calories[5:10] # most code works this way
cereal_calories[5:10]
```

```
## [1] 110 110 110 130  90  90
```

d) Add one more element to `cereal.calories` using `c()`.

Solution

```
cereal_calories <- c(cereal_calories, 1.0) #c for concatenate
length(cereal.calories)
```

```
## [1] 77
```

2.5 Matrix

a) Can you force the cereal data frame to be a Matrix? (`as.matrix(cereal)`). Check that the elements have been forced into the character type.

Solution

```
cereal.matrix <- as.matrix(cereal)
str(cereal.matrix)
```

```
##  chr [1:77, 1:16] "100%_Bran" "100%_Natural_Bran" "All-Bran" ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:16] "name" "mfr" "type" "calories" ...
```

b) Now do this again, but this time leave out the `mfr`, `name` and `type` columns. Check that the elements are now numeric.

Solution

```
cereal.removed <- cereal[, -(1:3)]
cereal.removed
```

```
## # A tibble: 77 x 13
##    calories protein   fat sodium fiber carbo sugars potass vitamins shelf weight
##       <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>    <dbl> <dbl>  <dbl>
##  1       70       4     1    130    10     5      6    280       25     3    1
##  2      120       3     5     15     2     8      8    135        0     3    1
##  3       70       4     1    260     9     7      5    320       25     3    1
## 4       50       4     0    140  14     8        0    330        25     3   1
## 5      110       2     2    200   1    14        8     -1        25     3   1
## 6      110       2     2    180   1.5  10.5     10     70        25     1   1
## 7      110       2     0    125   1    11       14     30        25     2   1
## 8      130       3     2    210   2    18        8    100        25     3   1.33
## 9       90       2     1    200   4    15        6    125        25     1   1
## 10      90       3     0    210   5    13        5    190        25     3   1
## # ... with 67 more rows, and 2 more variables: cups <dbl>, rating <dbl>
```

```
cereal.numeric.matrix <- as.matrix(cereal.removed)
str(cereal.numeric.matrix)
```

```
##  num [1:77, 1:13] 70 120 70 50 110 110 110 130 90 90 ...
##  - attr(*, "dimnames")=List of 2
##   ..$ : NULL
##   ..$ : chr [1:13] "calories" "protein" "fat" "sodium" ...
```

3 Numerical summary

3.1 Summary

a) Use the summary function to extract the median, 1st quartile and 3rd quartile data from the sodium column.

Solution

```
summary(cereal$sodium)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   130.0   180.0   159.7   210.0   320.0
```

3.2 Basic statistics

b) Find the max, min, standard deviation and mean of the sodium (max(), min(), sd(), mean())

Solution

```
cereal_tbl %>% pull(sodium) %>% summary()
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   130.0   180.0   159.7   210.0   320.0
```

```
sodium<-cereal$sodium
max(sodium)
```

```
## [1] 320
```

```
min(cereal$sodium)
```

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

```
sd(cereal$sodium)
```

```
## [1] 83.8323
```

```
mean(cereal$sodium)
```

```
## [1] 159.6753
```

```
summary(sodium)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##     0.0   130.0   180.0   159.7   210.0   320.0
```

c) Find the mean sodium of each mfr.

Solution

```
# Can be done by repeated subsetting, this is tedious
kelloggs.cereals <- subset(cereal, mfr == "K")
mean(kelloggs.cereals$sodium)
```

```
## [1] 174.7826
```

```
# Can use a formula and the aggregate function

mean.sodiums <- aggregate(sodium ~ mfr, data = cereal, FUN = mean)
# Can split vector (or data.frame if you wanted) by
# another vector. In this case, split by species.


split.sodium <- split(cereal$sodium, cereal$mfr)
# Apply a function over a list and return a list (_l_apply for list apply)
lapply(split.sodium, mean)
```

```
## $A
## [1] 0
##
## $G
## [1] 200.4545
##
## $K
## [1] 174.7826
##
## $N
## [1] 37.5
##
## $P
## [1] 146.1111
##
## $Q
## [1] 92.5
##
## $R
## [1] 198.125
```

```
# Apply a function over a list and return a _s_implified format (_s_apply for simplify apply)
sapply(split.sodium, mean)
```

```
##        A        G        K        N        P        Q        R
##   0.0000 200.4545 174.7826  37.5000 146.1111  92.5000 198.1250
```

```
# Also could use by and tapply, vapply, for the interested students, omitted here.

cereal_tbl %>%
    select(sodium, mfr) %>%
    group_by(mfr) %>%
    summarise(mean_sodium = mean(sodium))
```

```
## # A tibble: 7 x 2
##   mfr   mean_sodium
##   <chr>       <dbl>
## 1 A             0
## 2 G           200.
## 3 K           175.
## 4 N            37.5
## 5 P           146.
## 6 Q            92.5
## 7 R           198.
```

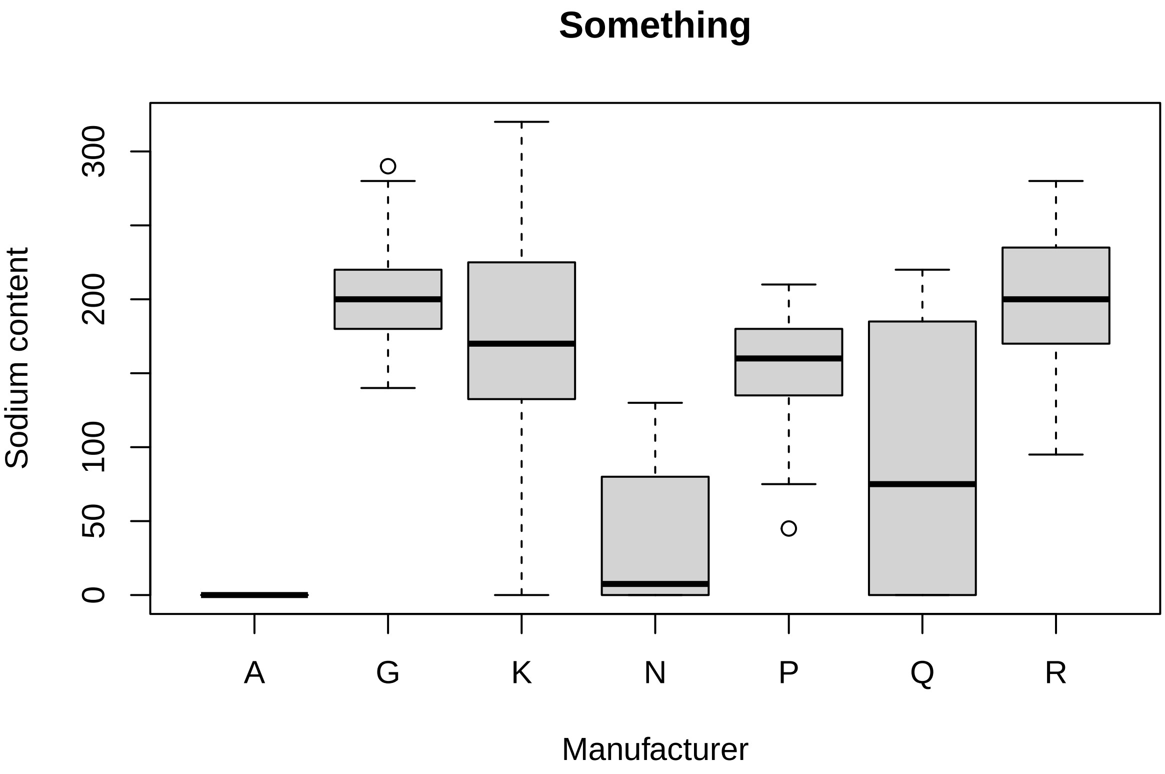
4 Graphical summary

4.1 Boxplot

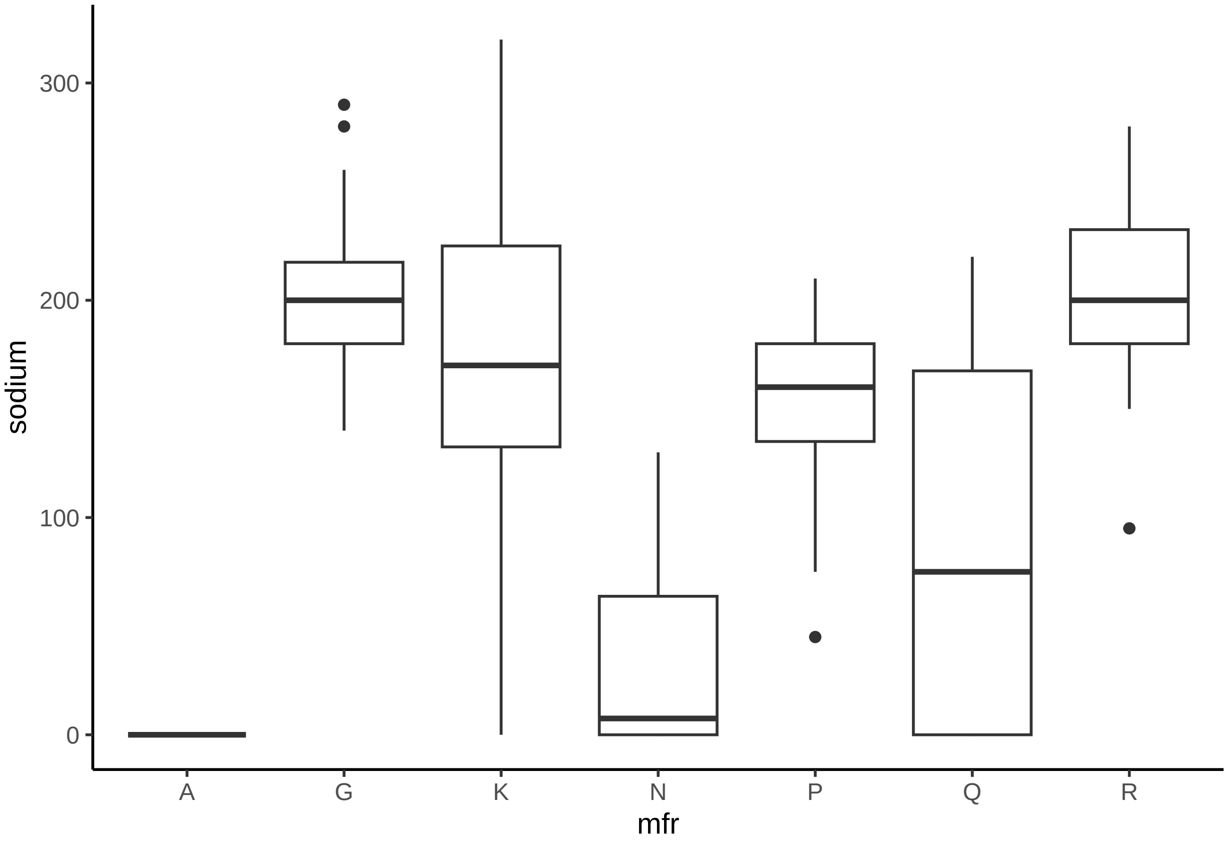
a) Make a boxplot of the `sodium` against `mfr` using `boxplot()`.

Solution

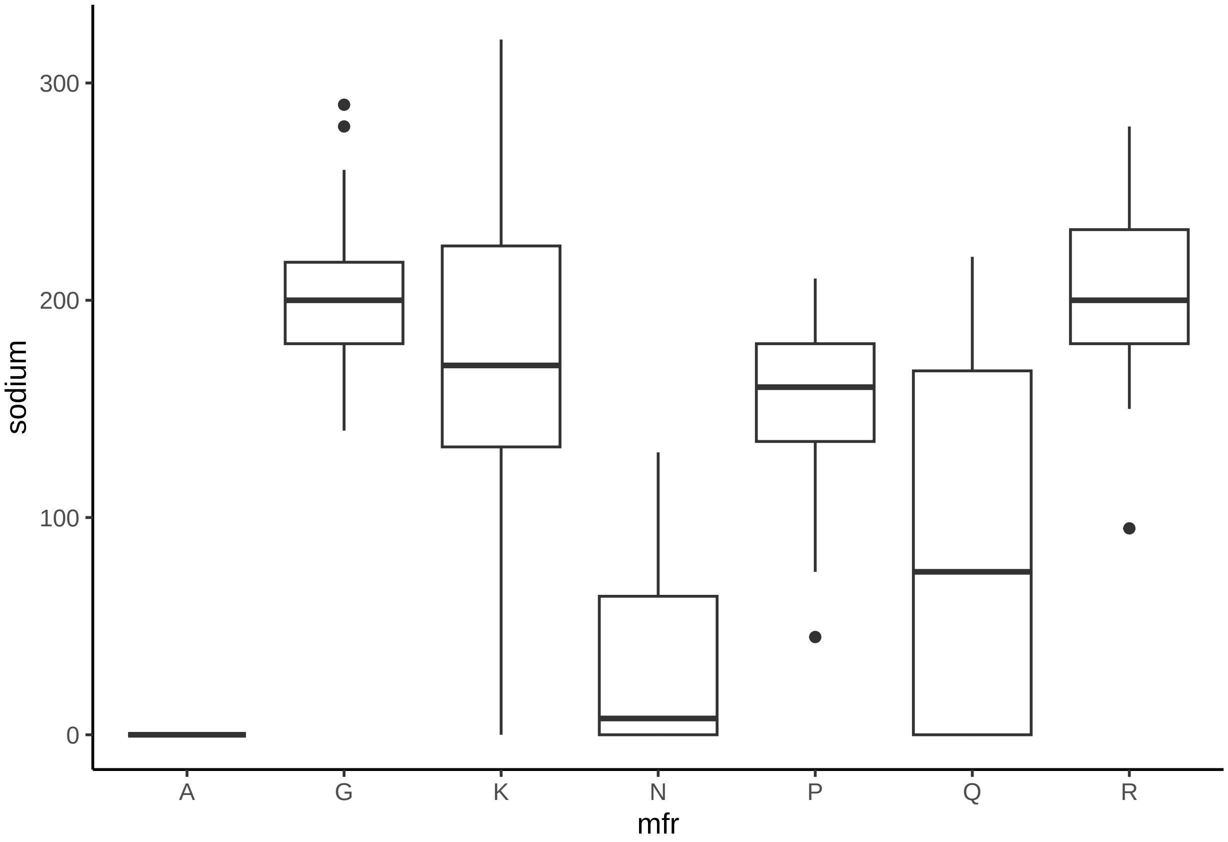
```
boxplot(sodium ~ mfr, data = cereal,
        xlab = "Manufacturer", ylab = "Sodium content", main = "Something")
```



```
ggplot(cereal_tbl, aes(x = mfr, y = sodium)) + geom_boxplot() + theme_classic()
```



```
cereal_tbl %>% ggplot(aes(x = mfr, y = sodium)) + geom_boxplot() + theme_classic()
```

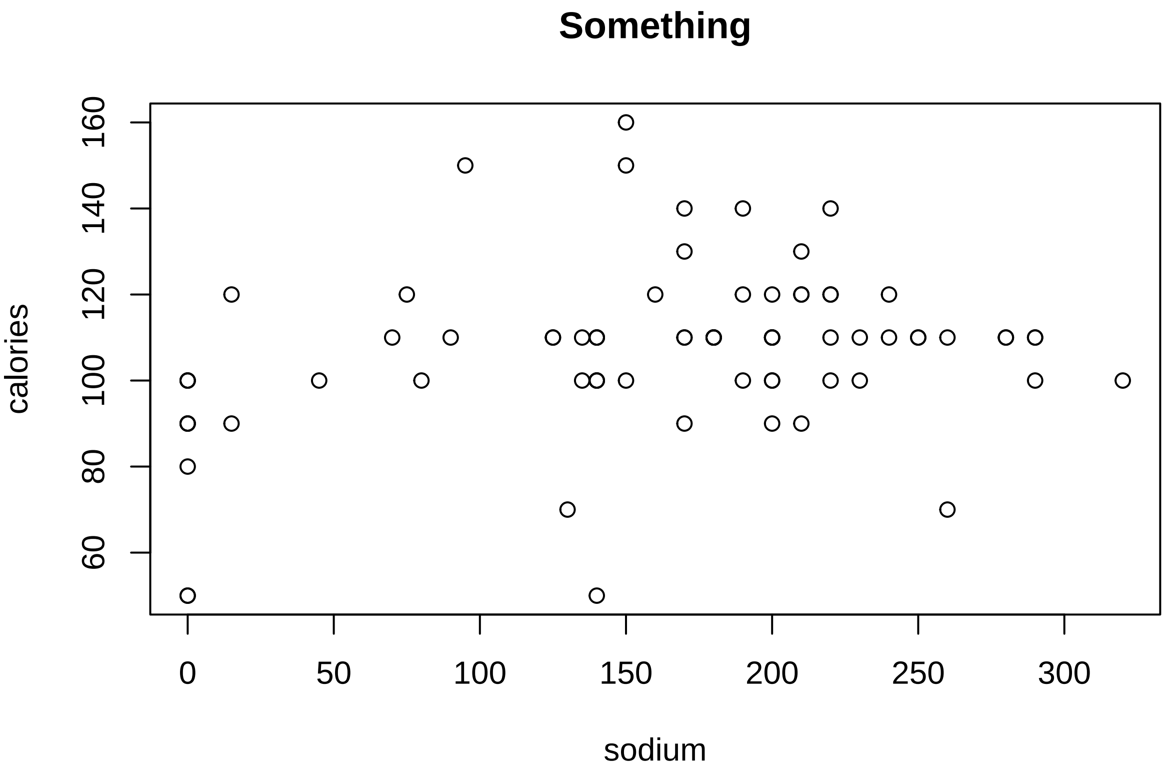


4.2 plot?

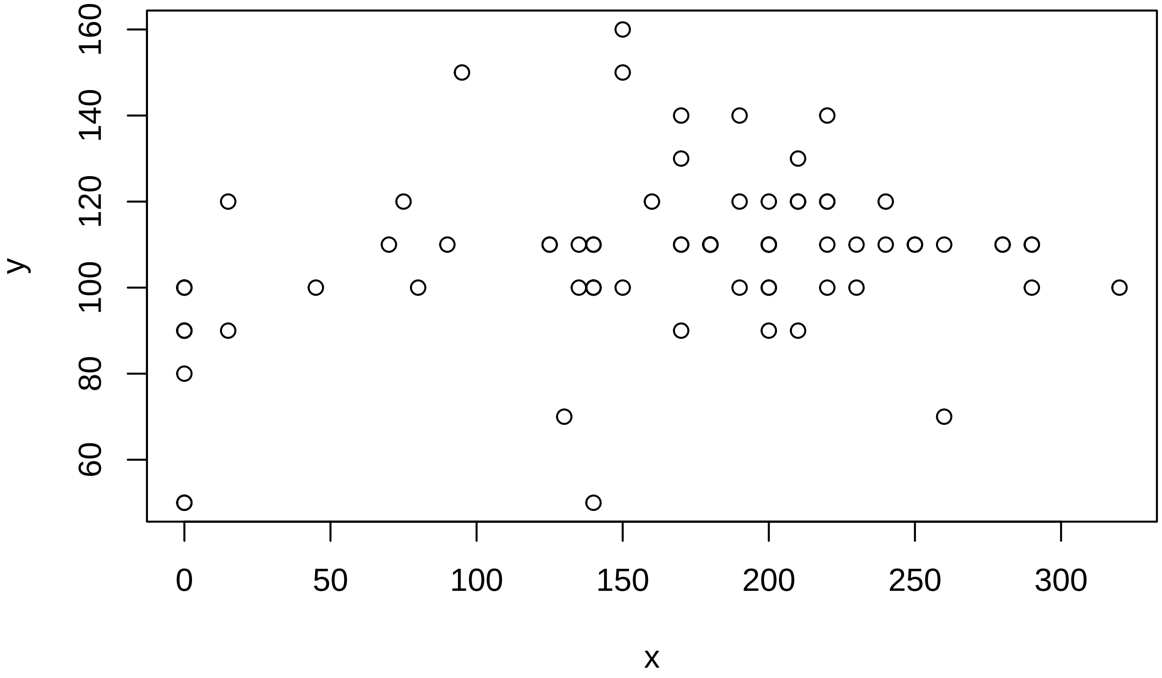
b) Plot calories against sodium using plot().

Solution

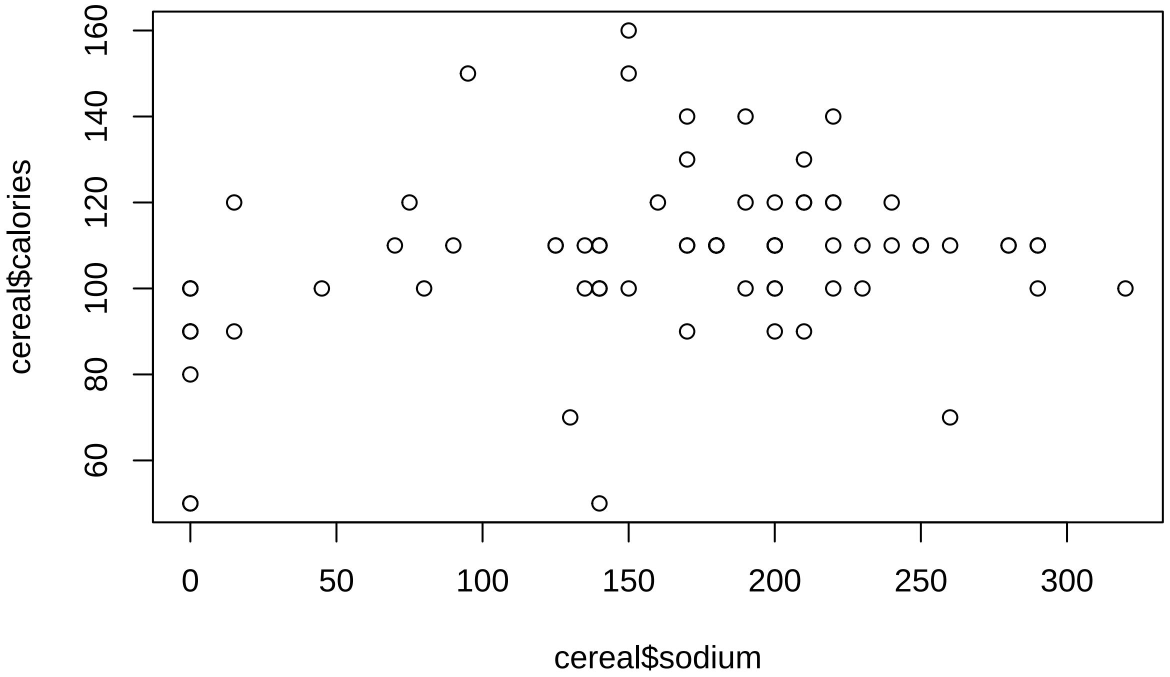
```
# Using formula
plot(calories ~ sodium, data = cereal, main = "Something")
```



```
# Another way, define x and y
x <- cereal$sodium
y <- cereal$calories
plot(x, y)
```



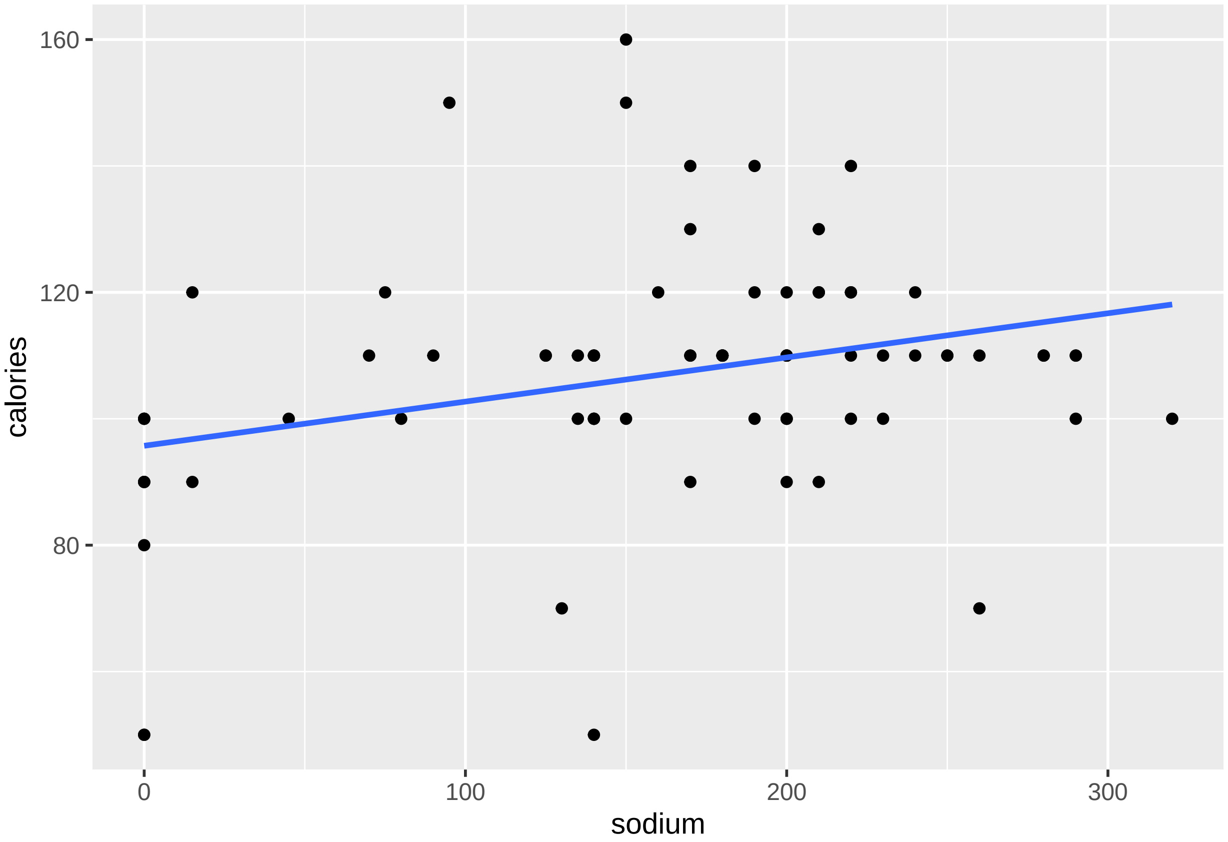
```
plot(cereal$sodium, cereal$calories)
```



```
# Alternatively, use with to help R find the vectors
#with(cereal, plot(sodium, calories))

ggplot(cereal_tbl, aes(x = sodium, y = calories)) +
     geom_point() + geom_smooth(method = "lm", se = FALSE)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



5 Write Data to File

b) Write data frame with only the Kellogg's observations to a file called `kelloggs.csv`. Use the `write.csv` command.

Solution

```
write.csv(kelloggs_tbl, file = "kelloggs.csv")


head(cereal)
```

```
## # A tibble: 6 x 16
##   name         mfr   type  calor~1 protein   fat sodium fiber carbo sugars potass
##   <chr>        <chr> <chr>   <dbl>   <dbl> <dbl>  <dbl> <dbl> <dbl>  <dbl>  <dbl>
## 1 100%_Bran    N     C          70       4     1    130  10     5        6    280
## 2 100%_Natur~ Q     C         120       3     5     15   2     8        8    135
## 3 All-Bran     K     C          70       4     1    260   9     7        5    320
## 4 All-Bran_w~ K     C          50       4     0    140  14     8        0    330
## 5 Almond_Del~ R     C         110       2     2    200   1    14        8     -1
## 6 Apple_Cinn~ G     C         110       2     2    180   1.5  10.5     10     70
## # ... with 5 more variables: vitamins <dbl>, shelf <dbl>, weight <dbl>,
## #   cups <dbl>, rating <dbl>, and abbreviated variable name 1: calories
```