Exercises Topics in Tidyverse

Exercises Week 12

08 January 2020

Exercises part 1: Grammar of Data

Optional: In case the lecture went a bit too fast...

When the lecture about the functions in `dplyr` went a bit to fast for you, then try out the Dataschool tutorial and watch the youtube video that is posted on that site.

Alternatively, take a look at Hadley Wickham's R for Data Science book, specifically chapters 9 to 13.

Practicing the `tidyverse`

Fot this task, you are a data analyst intern at the phone company BORANGE and just received two phony data sets. The first data set is called `Calls_minutes_pm` and the second is called `Traffic_MB`. Each data set is supposedly coming from a different department in the company. Your project leader requested you to obtain the information of 60 randomly sampled, anonymous phone users.

Your project leader wants you to gather the information of a stratified random sample of 20 enthusiastic phone users, 20 passive phone users, and 20 average users that are neither passive nor specifically enthusiastic. The information requested regarding these 60 anonymous phone users (ID = 1.... 60) concerns their calling time (in minutes), and data upload/download for each month of the year. This information is stored in the data sets `Calls_minutes_pm` and `Traffic_MB`, respectively.

You should be able to load the data as follows:

```r
load("./0_data/Calls_minutes_pm.RData")
load("./0_data/Traffic_MB.RData")
```

Merging / Joining Two Data Sets

Your project leader does not know much about statistics (and computing). Because it is very annoying to work with missing data, he requested that the departments should only send data of those months where no phone user has any missing (`NA`) values.

At first sight it looks like there are many `NA` values. However, when you merge the two data sets to fit the exact requirements of your project leader, your data would have no missings at all:

```r
load("./0_data/Phone_data.RData")
Phone_data
```

```
## # A tibble: 412 x 39
##       ID Group Month Incoming_Domest~ Incoming_Foreign Incoming_Mum
##    <int> <fct> <chr>            <int>            <int>        <int>
##  1     1 Pass~ April               21                4            5
##  2     1 Pass~ Augu~               24                6            4
##  3     1 Pass~ Febr~               22                3            4
##  4     1 Pass~ Janu~               12                4            3
##  5     1 Pass~ June                21                3            3
##  6     1 Pass~ March               27                2            4
##  7     1 Pass~ May                 26                4            2
##  8     1 Pass~ Sept~               26                4            4
##  9     2 Pass~ April               26                5             5
## 10     2 Pass~ Augu~               14                2             4
## # ... with 402 more rows, and 33 more variables: Outgoing_Domestic <int>,
## #   Outgoing_Foreign <int>, Outgoing_Mum <int>,
## #   Browsing_DownloadCellular <int>, Browsing_DownloadWifi <int>,
## #   Browsing_UploadCellular <int>, Browsing_UploadWifi <int>,
## #   Hotspot_DownloadCellular <int>, Hotspot_UploadCellular <int>,
## #   Maps_DownloadCellular <int>, Maps_DownloadWifi <int>,
## #   MBaggr_DownloadCellular <int>, MBaggr_DownloadForeign <int>,
## #   MBaggr_DownloadWifi <int>, MBaggr_UploadCellular <int>,
## #   MBaggr_UploadForeign <int>, MBaggr_UploadWifi <int>,
## #   Skype_DownloadCellular <int>, Skype_DownloadWifi <int>,
## #   Skype_UploadCellular <int>, Skype_UploadWifi <int>,
## #   Spotify_DownloadCellular <int>, Spotify_DownloadWifi <int>,
## #   Viber_DownloadCellular <int>, Viber_DownloadWifi <int>,
## #   Viber_UploadCellular <int>, Viber_UploadWifi <int>,
## #   WhatsApp_DownloadCellular <int>, WhatsApp_DownloadWifi <int>,
## #   WhatsApp_UploadCellular <int>, WhatsApp_UploadWifi <int>,
## #   Youtube_DownloadCellular <int>, Youtube_DownloadWifi <int>
```

1.1 The task

Merge the `Calls_minutes_pm` and `Traffic_MB` data such that the newly merged data set is equal to the wide format data set `Phone_data`.

Hint: You could do it in three steps: 1. Recreate `Phone_calltime` data (check the data folder) from the `Calls_minutes_pm.RData`. 2. Recreate `Phone_traffic` (check the data folder) from Traffic_MB. Note: You have to find a way to, in the end, only select those columns that have at least one non-missing value. 3. Merge the two resulting data frames.

Answer

```r
load("./0_data/Phone_calltime.RData")
call_time <- Calls_minutes_pm %>%
  unite(in_out_type, In_Out, Type) %>%
  gather(key = "Month", value = "Value", one_of(month.name), na.rm = TRUE) %>%
  spread(in_out_type, Value)
```

```r
load("./0_data/Phone_traffic.RData")
traffic <- Traffic_MB %>%
  unite(type_network, Up_Down, Network, sep = "") %>%
  gather(key = "Type", value = "Value", -ID, -Group, -Month, -type_network) %>%
  unite(combined_type, Type, type_network) %>%
  spread(combined_type, Value) %>%
  select_if(~ any(!is.na(.x))) %>%
  # `~any(!is.na(.x))` is equal to `function(x) any(!is.na(x))`
  mutate(Month = month.name[Month])
```

```r
phone_data_full <- inner_join(call_time, traffic, by = c("ID", "Group", "Month"))
# if not specified, the keys on which to join are selected automatically
# based on overlapping colnames. (i.e., here we could omit the `by` argument.)
phone_data_full %>% all_equal(Phone_data)
```

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

1.2 Data Description

For further analysis, a 60x38 `data.frame` is desired. The first two columns remain `ID` and `Group`, the remaing `36` columns represent the variables that represent the average calling time per month, or the average number of MB's used for up or downloading.

Your new `data.frame` should be similar to the `Phone_means` data (see the data folder).

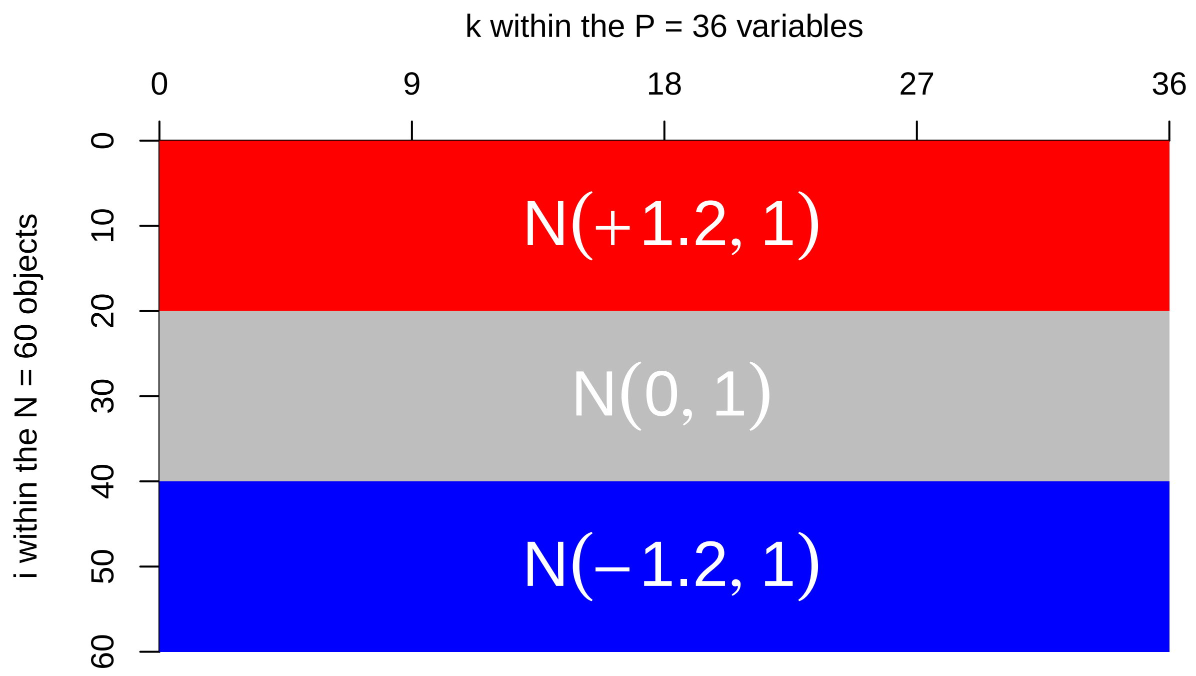
Answer

```r
load("./0_data/Phone_means.RData")
phone_means_aggr <- phone_data_full %>%
  group_by(Group, ID) %>%
  summarise_at(vars(-Month), mean)
  # vars allows you to use `select()` helpers&notation inside
```

Exercises part 2: Grammar of Graphics

2. Visualization of a Statistical Model

After some intractable and obscurious re-scaling of the variables, the `Phone_means` data set got transformed into the data set `Phone_transfmeans`. Your project leader states that a good statistical model of the `Phone_transfmeans` data can be visualized as follows:



The first 20 rows in the plot are the enthusastic phone users. Their re-scaled values of the average minutes and MB's (per month) can be seen as random draws from a normal distribution with the mean equal to 1.2 and a variance of 1. The values for the passive phone users (the blue group) can be seen as random draws from a normal distribution with a mean of -1.2 and variance of 1. The group in the middle (depicted in grey) have realizations that come frome a standard normal distribution.

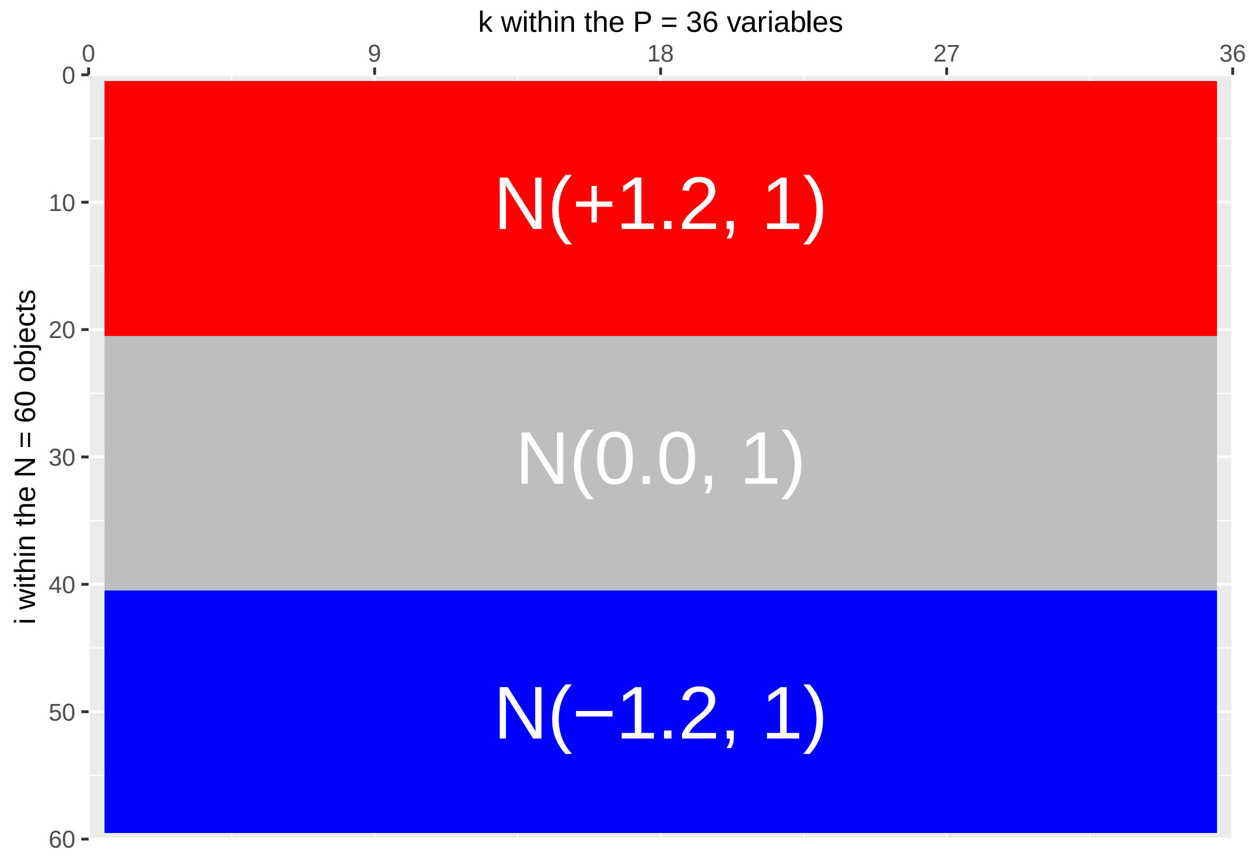
2.1a. Visualize the statistical model in `ggplot2`

Try to recreate the above visualization of the statistical model with `ggplot2`. You may use the data that we have specified below:

```r
transf_data <- data.frame(
  cbind(
    expand.grid(Objects = 1:60, Variables = 1:36),
    Value = rep(rep(c(+1.2, 0, -1.2), each = 20), 36)
))
```

Answer:

```r
ggplot(transf_data, aes(x = Variables, y = Objects, fill = Value)) +
  geom_tile(show.legend = FALSE) +
  scale_y_continuous(trans = "reverse", breaks = seq(0, 60, 10),
                     limits = c(60, 0), expand = c(0, 0)) +
  scale_x_continuous(position = "top", breaks = seq(0, 36, 9),
                     limits = c(0, 36), expand = c(0, 0)) +
  scale_fill_gradient2(low = "blue", mid = "grey", high = "red", midpoint = 0) +
  xlab("k within the P = 36 variables") +
  ylab("i within the N = 60 objects") +
  annotate("text", x = 18, y = c(10, 30, 50),
           label = c("N(+1.2, 1)", "N(0.0, 1)","N(-1.2, 1)"),
           color = "white", size = 10)
```



2.1b. Using Monte Carlo Data Sets

Create 1000 Monte Carlo data sets that come from the above statistical model. Then, show two plots next to each other. In the first plot visualize only one Monte Carlo data set in the same was as the statistical model is visualized. Do the the same for the second plot, but show the visualization of your estimate of the expectation of each value in the Monte Carlo data sets, i.e. your esimate of the expectation for the value of the first row and first variable (x_{ij}) is calculated as

$$E\left[x_{ij}\right] = \frac{1}{B}\sum_{b=1}^{B} x_{11}^{(b)},$$

where B is the number of Monte Carlo data sets, indexed over b.

Note that the second plot representing the expectation of the Monte Carlo data sets should very closely resemble your visualization of the statistical model

Answer

First, create the base plot without any data and geometries, to save some typing.

```r
base_plot <- ggplot() +
  scale_y_continuous(trans = "reverse", breaks = seq(0, 60, 10),
                     limits = c(60, 0), expand = c(0, 0)) +
  scale_x_continuous(position = "top", breaks = seq(0, 36, 9),
                     limits = c(0, 36), expand = c(0, 0)) +
  scale_fill_gradient2(low = "blue", mid = "grey", high = "red", midpoint = 0) +
  xlab("k within the P = 36 variables") +
  ylab("i within the N = 60 objects") +
  theme(legend.position = "none")
```

```r
P <- 36
N <- 60
B <- 1e3

base_df <- data.frame(cbind(expand.grid(Objects = 1:N, Variables = 1:P)))

simulate_mc <- function() {
  rep(rep(c(+1.2, 0, -1.2), each = 20), 36) + rnorm(P * N)
}

base_df <- base_df %>% mutate(
  one_obs = simulate_mc(),
  avg = rowMeans(replicate(B, simulate_mc()))
)

p1 <- base_plot +
  geom_tile(data = base_df, aes(x = Variables, y = Objects, fill = one_obs))

p2 <- base_plot +
  geom_tile(data = base_df, aes(x = Variables, y = Objects, fill = avg))

#install.packages("cowplot")
cowplot::plot_grid(p1, p2, ncol = 2)
```

2.2 Using the Real (Transformed Means) Data

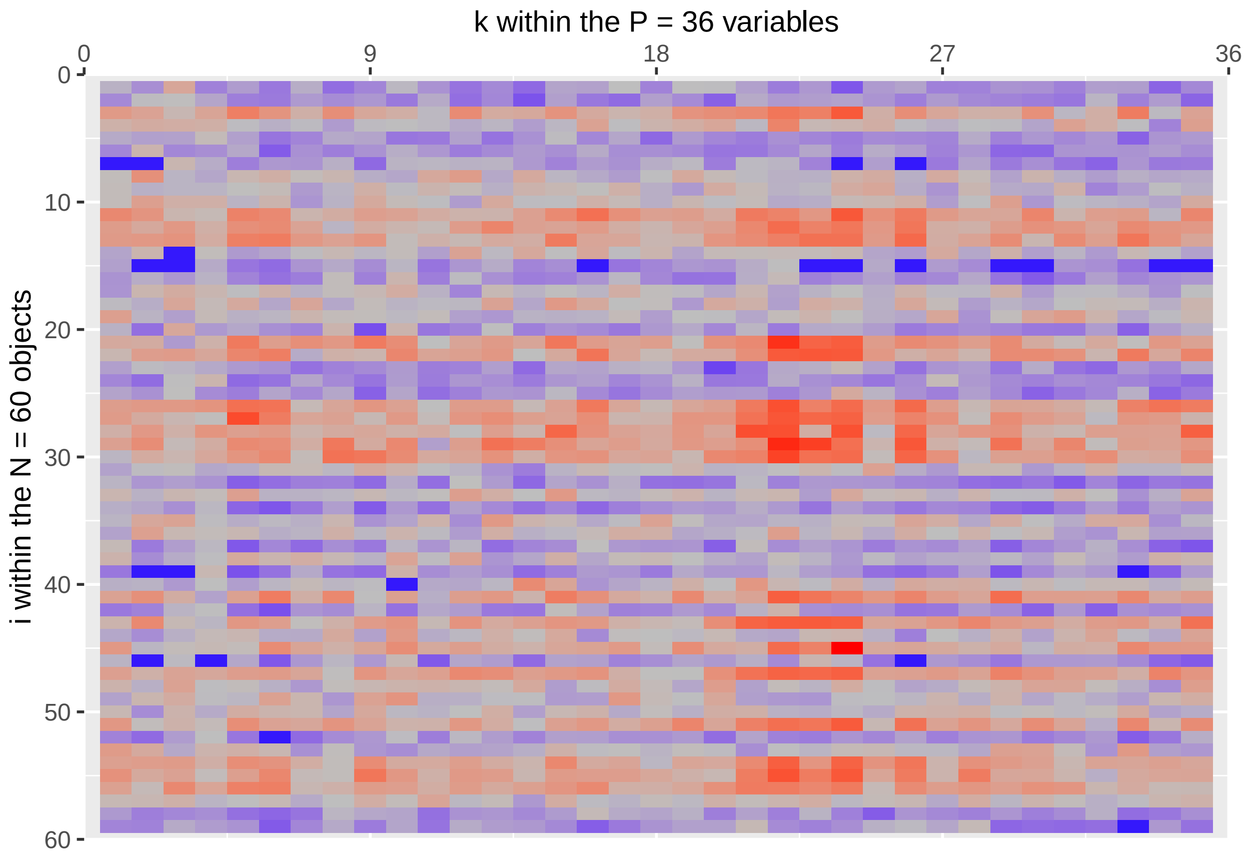
Also create the plot of **2.1** for the `Phone_transfmeans` data. This time keep the legend. Just by looking at the pictures only, do you think it would be fair to say that we can assume the statistical model for the `Phone_transfmeans` data?

You can load the `Phone_transfmeans` data as follows:

```
load("./0_data/Phone_transfmeans.RData")
```

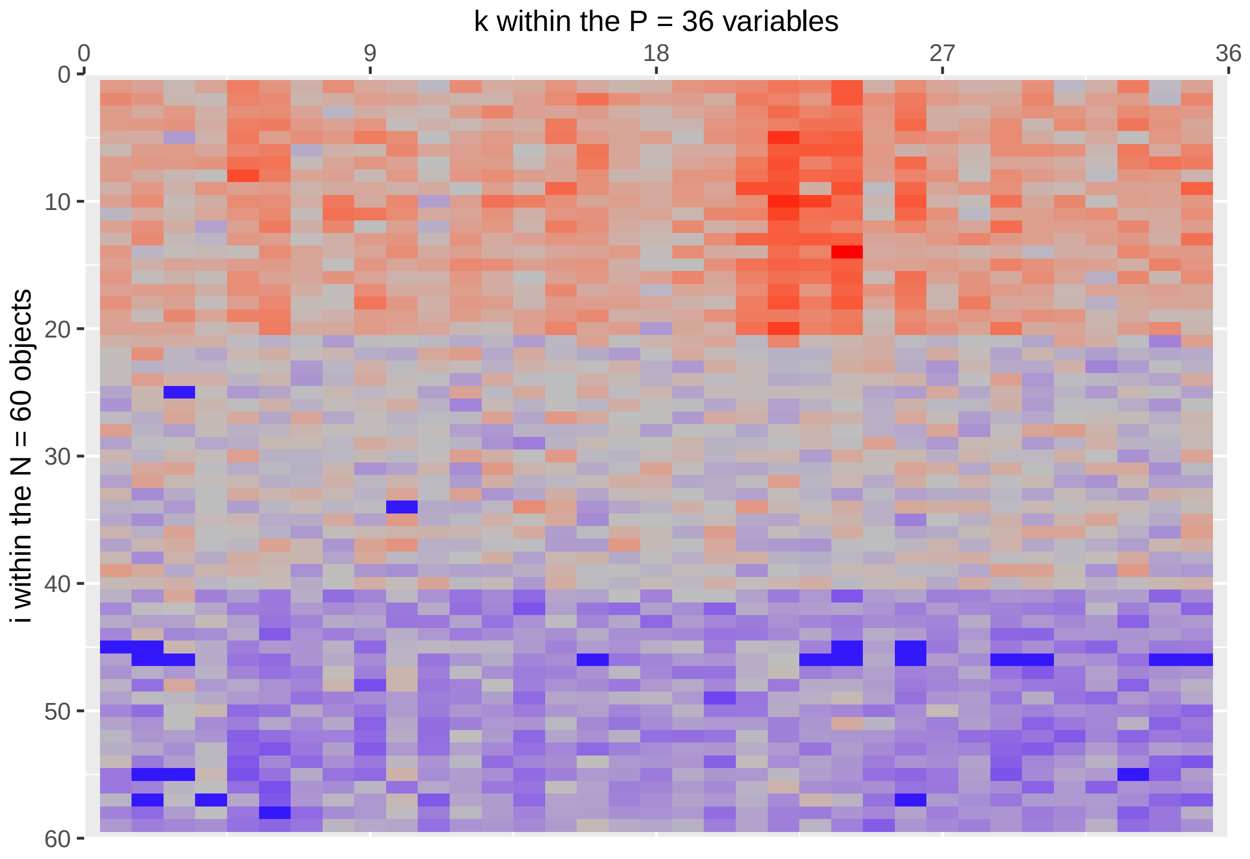
Answer

```
transfmeans_agg <- Phone_transfmeans %>%
  gather(Variables, Value, -ID, -Group) %>%
  arrange(Variables, ID) %>%
  mutate(VariableID = rep(1:36, each = 60))

base_plot +
  geom_tile(data = transfmeans_agg, aes(x = VariableID, y = ID, fill = Value))
```



It seems the model is the same (since the color does not seem to change per row). Only the rows are not ordered by groups anymore.

```
transfmeans_agg <- transfmeans_agg %>%
  arrange(VariableID, Group) %>%
  mutate(ID = rep(seq(60), 36))

base_plot +
  geom_tile(data = transfmeans_agg, aes(x = VariableID, y = ID, fill = Value))
```


k within the P = 36 variables
0
9
18
27
36
i within the N = 60 objects
0
10
20
30
40
50
60