Data science with R: tidyverse

VI Functional programming: purrr

Assignment

In this assignment we will test our **purrr** skills! Create *R* script called *assignment_6.R*. From the course sources download zip file: "**assignment_06.zip**", and extract the files from the zip file in the project **data** folder.

Exercise 1

In the first exercise we will try to simulate some data using *R*'s built in random numbers generators. Numbers will be generated using different type of distributions (each distribution has its own function inside *R*).

Inside "**assignment_06.zip**" there is a file called "**simulations_blue_print.txt**", which holds the instruction for our simulations. Instructions are given as a text input, each line includes information for given group of simulations:

- first part tells the function used in the simulation (for example "rnorm" - normal distribution random numbers generator)
- and all arguments used for given function (for example "n=10;mean=0;sd=3")
- arguments and function name are separated with character ";"

Your job is to:

- import "**simulations_blue_print.txt**" (try importing it a s a tibble)
- try to prepare imported data, so it can be used inside **map_invoke()** function, that will do the simulations
- execute the simulations and store it inside a tibble

Exercise 2

In the second exercise use simulated data from *Exercise 1* and:

- visualize distribution for each group of simulated data

- for the visualization use **density plot**
- create sub plots with **cowplot** package for each distribution type (sub plots for: **rnorm**, **runif**, **rexp**)
- **fill** of each density plot must be determined by function call for selected group (for example one group is: "rnorm;n=10;mean=0;sd=3", and so on)
- **HINT**: try to combine **group_by()**, **nest()**, **map()** and **ggplot()** to create nested plots as a list column inside a tibble
- **HINT**: use **plot_grid()** function on your list column to draw the sub plots

Exercise 3

In this exercise you will use so called "**gapminder**" data. The **gapminder** data is stored inside multiple **.csv** files (one file per each country). Your job is to import all the files at once, and store it into a single tibble using **map()** function logic. (The files an be found inside folder *gapminder*).

Exercise 4

In the last exercise you will be using imported **gapminder** data. We would like to:

- draw a **line chart** where **life expectancy** is shown on **y axis** and **year** is shown on **x axis**
- each country has its own line
- each continent has its own sub plot
- apply a similar recipe as for *Exercise 2* to draw these sub plots