Machine Learning - the basics in R

Jan-Philipp Kolb

22 Mai, 2019

Introduction round

Please tell us shortly...

- Where are you from? What are you studying/working?
- What is your experience level in R/other programming languages?
- What are your expectations of this course?
- Where do you think you can use Machine Learning in the future?

Preliminaries

- This topic is huge - we concentrate on presenting the applications in R
- Usually we have big differences in knowledge and abilities of the participants - please tell, if it is too fast or slow.
- We have many **exercises** because at the end you can only learn on your own
- We have many **examples** - try them!
- If there are questions - always ask
- R is more fun together - ask your neighbor

Content of this section

- The first section is about laying the foundations in R. We will need all things covered later on.

Topics section:

- Why R is a good choice
- Constraints of R-usage
- R is modular
- Import and export of data

Why R is a good choice ...

- ... because it is an **open source language**
- ... outstanding graphs - **graphics**, **graphics**, **graphics**
- ... relates to other languages - **R can be used in combination with other programs** - e.g. **data linking**
- ...R can be used **for automation**
- ... Vast Community - **you can use the intelligence of other people ;-)** and new statistical methodologies are implemented quite fast
- Because R can be combined with other programs like `PostgreSQL` or `Python`

Constraints

Newer modules in Python

- Machine learning is a field that changes rapidly.
- Some new tools are first developed in Python.
- The package `reticulate` offers the possibility to use these modules from an R environment.
- Good news - Python is also Open Source

Big Data

- Especially if you work with web data, you quickly have to deal with large amounts of data.
- Therefore one must fall back on databases and parallelization strategies, which can be used in R.

R is modular

Install packages from CRAN Server

```
install.packages("lme4")
```

Install packages from Bioconductor Server

```
source("https://bioconductor.org/biocLite.R")
biocLite(c("GenomicFeatures", "AnnotationDbi"))
```

Install packages from Github

```
install.packages("devtools")
library(devtools)

devtools::install_github("koalaverse/vip")
```

Preparation - packages

```
library(dplyr)
library(magrittr)
```

Import .csv data

The read.csv command

- Use read.csv2 for German data

```
?read.csv
?read.csv2
```

Using a path to import data

```
path1<-"https://raw.githubusercontent.com/"
path2<- "thomaspernet/data_csv_r/master/data/"
dname <- "titanic_csv.csv"
titanic <- read.csv(paste0(path1,path2,dname))
```

Save the dataset

```
save(titanic,file="../data/titanic.RData")
```

The titanic dataset

pclass	survived	name	sex	age	sibsp	parch	ticket	fare	cabin	embarked	home.dest
1	1	Allen, Miss. Elisab...	female	29.0000	0	0	24160	211.3375	B5	S	St Louis, MO
1	1	Allison, Master. H...	male	0.9167	1	2	113781	151.5500	C22 C26	S	Montreal, PQ / Chesterville, ON
1	0	Allison, Miss. Hele...	female	2.0000	1	2	113781	151.5500	C22 C26	S	Montreal, PQ / Chesterville, ON
1	0	Allison, Mr. Hudso...	male	30.0000	1	2	113781	151.5500	C22 C26	S	Montreal, PQ / Chesterville, ON
1	0	Allison, Mrs. Huds...	female	25.0000	1	2	113781	151.5500	C22 C26	S	Montreal, PQ / Chesterville, ON
1	1	Anderson, Mr. Harry	male	48.0000	0	0	19952	26.5500	E12	S	New York, NY
1	1	Andrews, Miss. Ko...	female	63.0000	1	0	13502	77.9583	D7	S	Hudson, NY
1	0	Andrews, Mr. Tho...	male	39.0000	0	0	112050	0.0000	A36	S	Belfast, NI
1	1	Appleton, Mrs. Ed...	female	53.0000	2	0	11769	51.4792	C101	S	Bayside, Queens, NY

The function scan to import data

- scan has an easy way to distinguish comments from data

```
?scan
```

Example dataset

```
cat("TITLE extra line", "# a comment","2 3 5 7", "11 13 17",
    file = "../data/ex.data", sep = "\n")
```

Import data and skip the first line

```
pp<-scan("../data/ex.data",skip=1,quiet=TRUE)
```

```
pp <- scan("../data/ex.data",comment.char="#", skip = 1,quiet =
```

The download the data from UCI.

```
path1 <- "http://archive.ics.uci.edu/ml/"
path2 <- "machine-learning-databases/00243/"
dname <- 'yacht_hydrodynamics.data'

url<- paste0(path1,path2,dname)
Yacht_Data <- readr::read_table(file = url)
```

Built in datasets

- A sample dataset is often provided to demonstrate the functionality of a package.
- These records can be loaded using the `data` command.

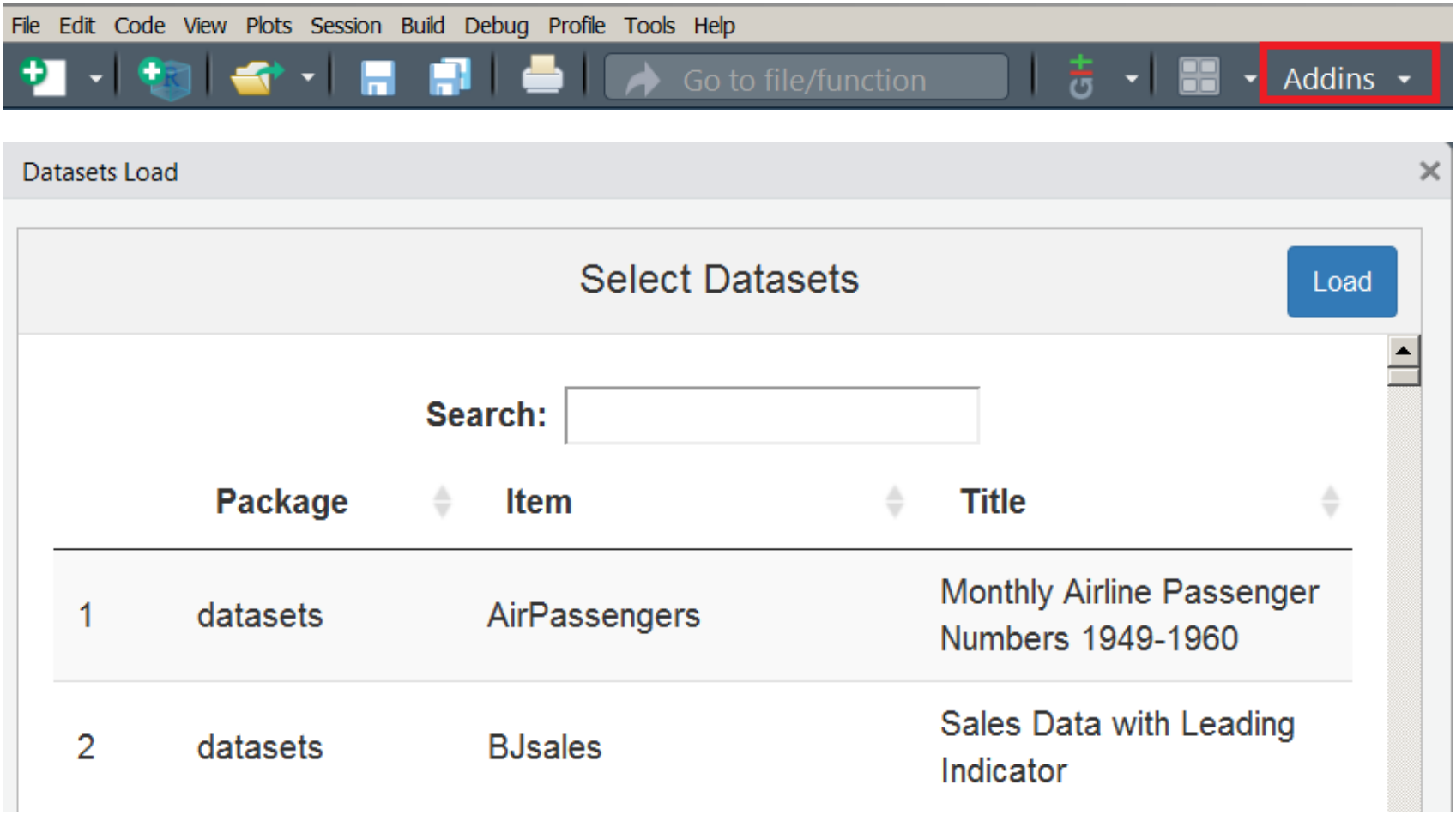
```
data(iris)
```

- There is also a **RStudio Add-In** that helps to find a built-in dataset.

```
install.packages("datasets.load")
```

Exkurs RStudio Addins

- Oben rechts befindet sich ein Button Addins



Exercise

Load the the built-in dataset `swiss` and answer the following questions:

- How many observations and variables are available?
- What is the scale level of the variables?

Create an interactive data table

The R-package data.table

Get an overview

```
data(airquality)
head(airquality)
```

```
##   Ozone Solar.R Wind Temp Month Day
## 1    41     190  7.4   67     5   1
## 2    36     118  8.0   72     5   2
## 3    12     149 12.6   74     5   3
## 4    18     313 11.5   62     5   4
## 5    NA      NA 14.3   56     5   5
## 6    28      NA 14.9   66     5   6
```

Overview with data.table

```
library(data.table)
(airq <- data.table(airquality))
```

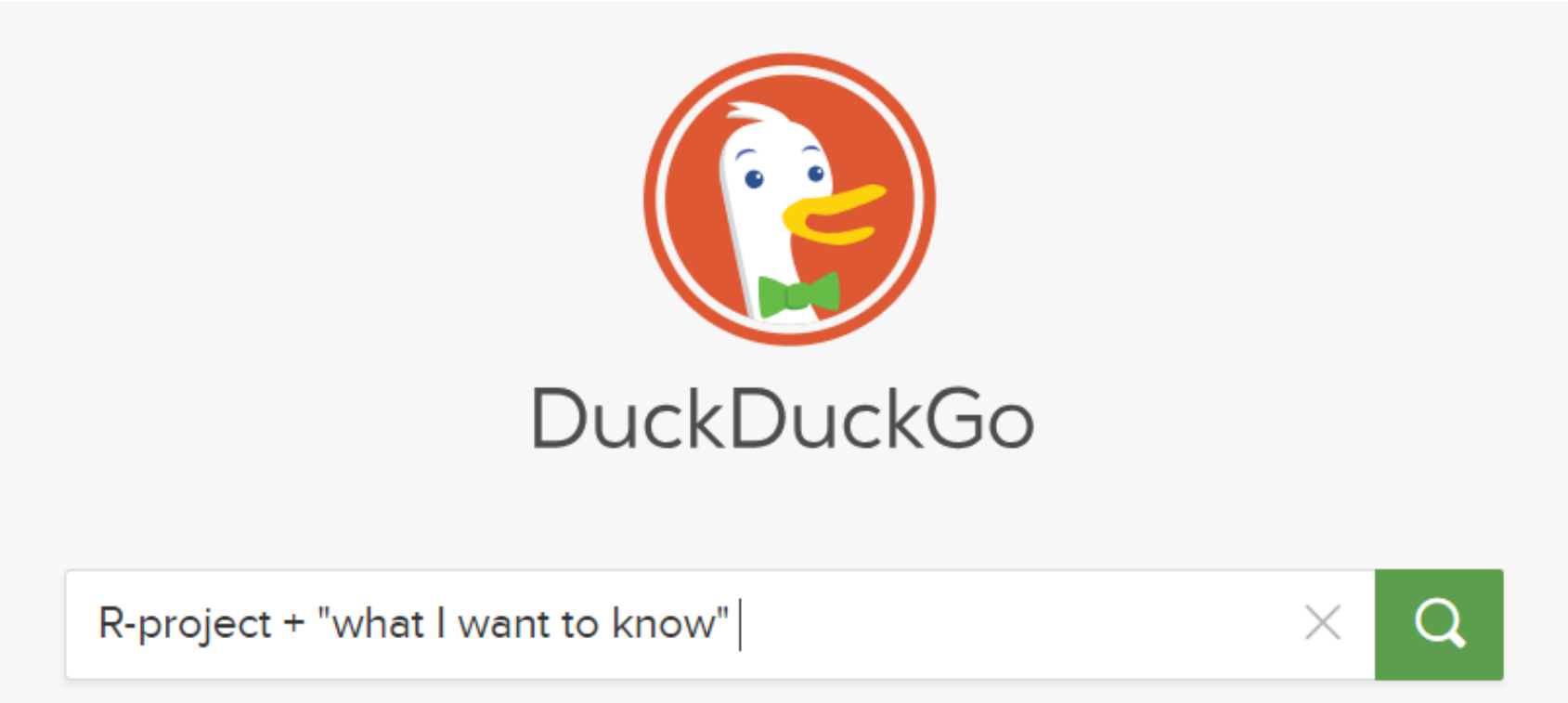
```
##       Ozone Solar.R Wind Temp Month Day
##   1:    41     190  7.4   67     5   1
##   2:    36     118  8.0   72     5   2
##   3:    12     149 12.6   74     5   3
##   4:    18     313 11.5   62     5   4
##   5:    NA      NA 14.3   56     5   5
##  ---
## 149:    30     193  6.9   70     9  26
## 150:    NA     145 13.2   77     9  27
## 151:    14     191 14.3   75     9  28
## 152:    18     131  8.0   76     9  29
## 153:    20     223 11.5   68     9  30
```

How to get help

- I use **duckduckgo:**

```
R-project + "what I want to know"
```

- this works of course for all search engines!



Exercise

- Draw 8 random numbers from the uniform distribution and save them in a vector `x`
- Compute the logarithm of `x`, return suitably lagged and iterated differences,
- compute the exponential function and round the result

```
## [1] 0.9 0.9 1.6 0.2 3.2 0.8 0.2
```

The pipe operator

```
library(magrittr)

# Perform the same computations on `x` as above
x %>% log() %>%
    diff() %>%
    exp() %>%
    round(1)
```

```
## [1] 0.9 0.9 1.6 0.2 3.2 0.8 0.2
```

How to deal with missing values

```
?na.omit
```

```
airq
```

```
##       Ozone Solar.R Wind Temp Month Day
##   1:     41     190  7.4   67     5   1
##   2:     36     118  8.0   72     5   2
##   3:     12     149 12.6   74     5   3
##   4:     18     313 11.5   62     5   4
##   5:     NA      NA 14.3   56     5   5
##  ---
## 149:     30     193  6.9   70     9  26
## 150:     NA     145 13.2   77     9  27
## 151:     14     191 14.3   75     9  28
## 152:     18     131  8.0   76     9  29
## 153:     20     223 11.5   68     9  30
```

The command na.omit

```
na.omit(airq)
```

```
##       Ozone Solar.R Wind Temp Month Day
##   1:     41     190  7.4   67     5   1
##   2:     36     118  8.0   72     5   2
##   3:     12     149 12.6   74     5   3
##   4:     18     313 11.5   62     5   4
##   5:     23     299  8.6   65     5   7
##  ---
## 107:     14      20 16.6   63     9  25
## 108:     30     193  6.9   70     9  26
## 109:     14     191 14.3   75     9  28
## 110:     18     131  8.0   76     9  29
## 111:     20     223 11.5   68     9  30
```

Clean the titanic data set

```
clean_titanic <- titanic %>%
  mutate(pclass=factor(pclass,levels = c(1, 2, 3),
                       labels=c('Upper','Middle','Lower')),
    survived = factor(survived,levels = c(0, 1),
                      labels=c('No', 'Yes'))) %>%
na.omit()
```

mutate(pclass = factor(...:

- Add label to the variable pclass.
- 1 becomes Upper, 2 becomes MIddle and 3 becomes lower

factor(survived,...:

- Add label to the variable survived.
- 1 Becomes No and 2 becomes Yes
- `na.omit()`: Remove the NA observations

Get an overview of the data

```
glimpse(clean_titanic)
```

```
## Observations: 1,045
## Variables: 13
## $ X          <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13,
## $ pclass     <fct> Upper, Upper, Upper, Upper, Upper, Upper, U
## $ survived   <fct> Yes, Yes, No, No, No, Yes, Yes, No, Yes, No
## $ name       <fct> "Allen, Miss. Elisabeth Walton", "Allison,
## $ sex        <fct> female, male, female, male, female, male, f
## $ age        <dbl> 29.0000, 0.9167, 2.0000, 30.0000, 25.0000,
## $ sibsp      <int> 0, 1, 1, 1, 1, 0, 1, 0, 2, 0, 1, 1, 0, 0, 0
## $ parch      <int> 0, 2, 2, 2, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
## $ ticket     <fct> 24160, 113781, 113781, 113781, 113781, 1995
## $ fare       <dbl> 211.3375, 151.5500, 151.5500, 151.5500, 151
## $ cabin      <fct> B5, C22 C26, C22 C26, C22 C26, C22 C26, E12
## $ embarked   <fct> S, S, S, S, S, S, S, S, S, C, C, C, C, S, S
## $ home.dest <fct> "St Louis, MO", "Montreal, PQ / Chestervill
```

Example Data - Housing Values in Suburbs of Boston

```
library(MASS)
bdat <- Boston
```

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
1	0.00632	18.0	2.31	0	0.5380	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
2	0.02731	0.0	7.07	0	0.4690	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
3	0.02729	0.0	7.07	0	0.4690	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
4	0.03237	0.0	2.18	0	0.4580	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
5	0.06905	0.0	2.18	0	0.4580	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
6	0.02985	0.0	2.18	0	0.4580	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.7
7	0.08829	12.5	7.87	0	0.5240	6.012	66.6	5.5605	5	311	15.2	395.60	12.43	22.9
8	0.14455	12.5	7.87	0	0.5240	6.172	96.1	5.9505	5	311	15.2	396.90	19.15	27.1
9	0.21124	12.5	7.87	0	0.5240	5.631	100.0	6.0821	5	311	15.2	386.63	29.93	16.5
10	0.17004	12.5	7.87	0	0.5240	6.004	85.9	6.5921	5	311	15.2	386.71	17.10	18.9
11	0.22489	12.5	7.87	0	0.5240	6.377	94.3	6.3467	5	311	15.2	392.52	20.45	15.0
12	0.11747	12.5	7.87	0	0.5240	6.009	82.9	6.2267	5	311	15.2	396.90	13.27	18.9
13	0.09378	12.5	7.87	0	0.5240	5.889	39.0	5.4509	5	311	15.2	390.50	15.71	21.7
14	0.62976	0.0	8.14	0	0.5380	5.949	61.8	4.7075	4	307	21.0	396.90	8.26	20.4

Normalize your data

Compute maximum and minimum per column

```
maxs <- apply(bdat, 2, max)
mins <- apply(bdat, 2, min)
```

scale - Scaling and Centering of Matrix-like Objects

```
scaled <- as.data.frame(scale(bdat, center = mins,
                              scale = maxs - mins))
```

The scaled data

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black
1	0.000000e+00	0.180	0.06781525	0	0.31481481	0.5775053	0.64160659	0.26920314	0.00000000	0.208015267	0.2872340	1.0000000
2	2.359225e-04	0.000	0.24230205	0	0.17283951	0.5479977	0.78269825	0.34896198	0.04347826	0.104961832	0.5531915	1.0000000
3	2.356977e-04	0.000	0.24230205	0	0.17283951	0.6943859	0.59938208	0.34896198	0.04347826	0.104961832	0.5531915	0.9897373
4	2.927957e-04	0.000	0.06304985	0	0.15020576	0.6585553	0.44181256	0.44854459	0.08695652	0.066793893	0.6489362	0.9942761
5	7.050701e-04	0.000	0.06304985	0	0.15020576	0.6871048	0.52832132	0.44854459	0.08695652	0.066793893	0.6489362	1.0000000
6	2.644715e-04	0.000	0.06304985	0	0.15020576	0.5497222	0.57466529	0.44854459	0.08695652	0.066793893	0.6489362	0.9929901
7	9.213230e-04	0.125	0.27162757	0	0.28600823	0.4696302	0.65602472	0.40292264	0.17391304	0.236641221	0.2765957	0.9967220
8	1.553672e-03	0.125	0.27162757	0	0.28600823	0.5002874	0.95983522	0.43838718	0.17391304	0.236641221	0.2765957	1.0000000
9	2.303251e-03	0.125	0.27162757	0	0.28600823	0.3966277	1.00000000	0.45035419	0.17391304	0.236641221	0.2765957	0.9741036
10	1.840173e-03	0.125	0.27162757	0	0.28600823	0.4680973	0.85478888	0.49673090	0.17391304	0.236641221	0.2765957	0.9743053
11	2.456674e-03	0.125	0.27162757	0	0.28600823	0.5395670	0.94129763	0.47441552	0.17391304	0.236641221	0.2765957	0.9889556
12	1.249299e-03	0.125	0.27162757	0	0.28600823	0.4690554	0.82389289	0.46350335	0.17391304	0.236641221	0.2765957	1.0000000
13	9.830293e-04	0.125	0.27162757	0	0.28600823	0.4460625	0.37178167	0.39295620	0.17391304	0.236641221	0.2765957	0.9838620

The command sample

- We can use this command to draw a sample.
- We need the command later to split our dataset into a test and a training dataset.

```
sample(1:10,3,replace=T)
```

```
## [1] 2 9 1
```

```
sample(1:10,3,replace=T)
```

```
## [1] 7 5 6
```

Set a seed

- `set.seed` is the recommended way to specify seeds.
- If we set a seed, we get the same result for random events.
- This function is mainly required for simulations.

```
set.seed(234)
sample(1:10,3,replace=T)
```

```
## [1] 8 8 1
```

```
set.seed(234)
sample(1:10,3,replace=T)
```

```
## [1] 8 8 1
```

Time measurement

```
start_time <- Sys.time()
ab <- runif(10000000)
end_time <- Sys.time()

end_time - start_time

## Time difference of 0.384038 secs
```

How many cores are available

```
library(doParallel)
detectCores()
```

```
## [1] 4
```

Make cluster

```
cl <- makeCluster(detectCores())
registerDoParallel(cl)

start_time <- Sys.time()
ab <- runif(10000000)
end_time <- Sys.time()

end_time - start_time


## Time difference of 0.3160312 secs

stopCluster(cl)

?parallel::makeCluster
```

Resources

- **Course materials for the Data Science Specialization**
- Data wrangling - ** vignette `dplyr`** -
- The usage of pipes - **`magrittr` vignette**