TEXT MINING

Lecture 02

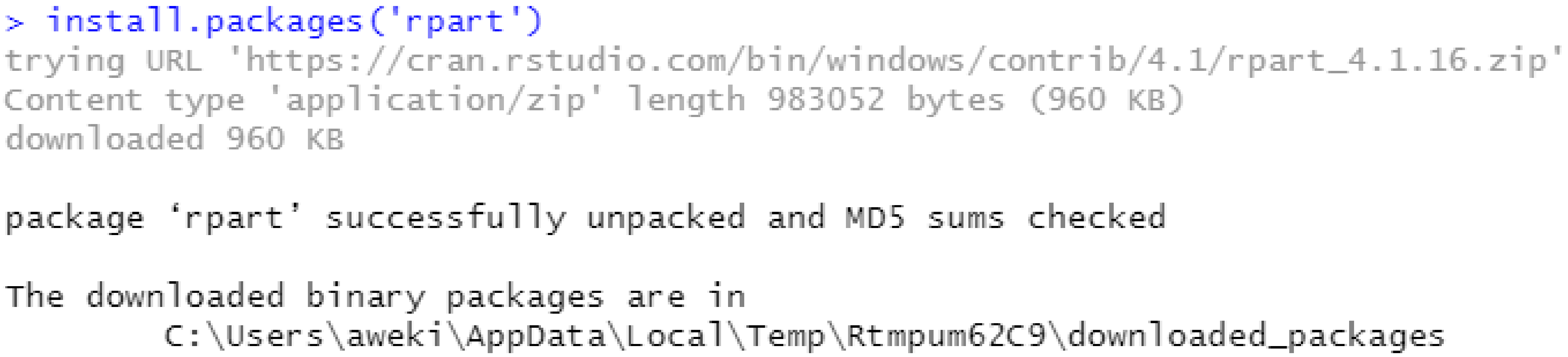
INTERMEDIATE R

KEUNGOUI KIM
awekim@handong.edu

R Basics

Package Installation and Loading

- install.packages(“package name”)

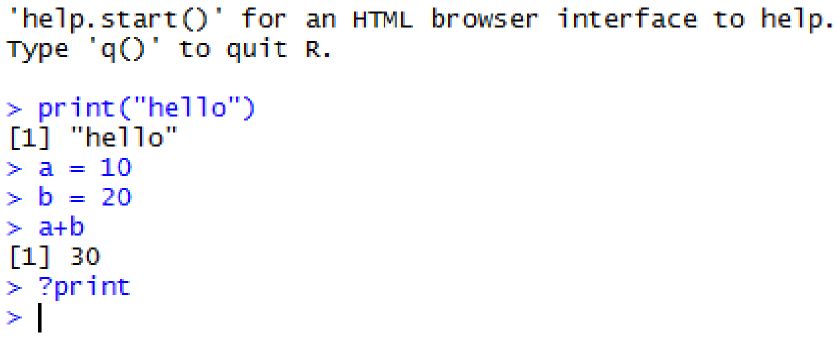


```
> install.packages('rpart')
trying URL 'https://cran.rstudio.com/bin/windows/contrib/4.1/rpart_4.1.16.zip'
Content type 'application/zip' length 983052 bytes (960 KB)
downloaded 960 KB

package ‘rpart’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\aweki\AppData\Local\Temp\Rtmpum62C9\downloaded_packages
```

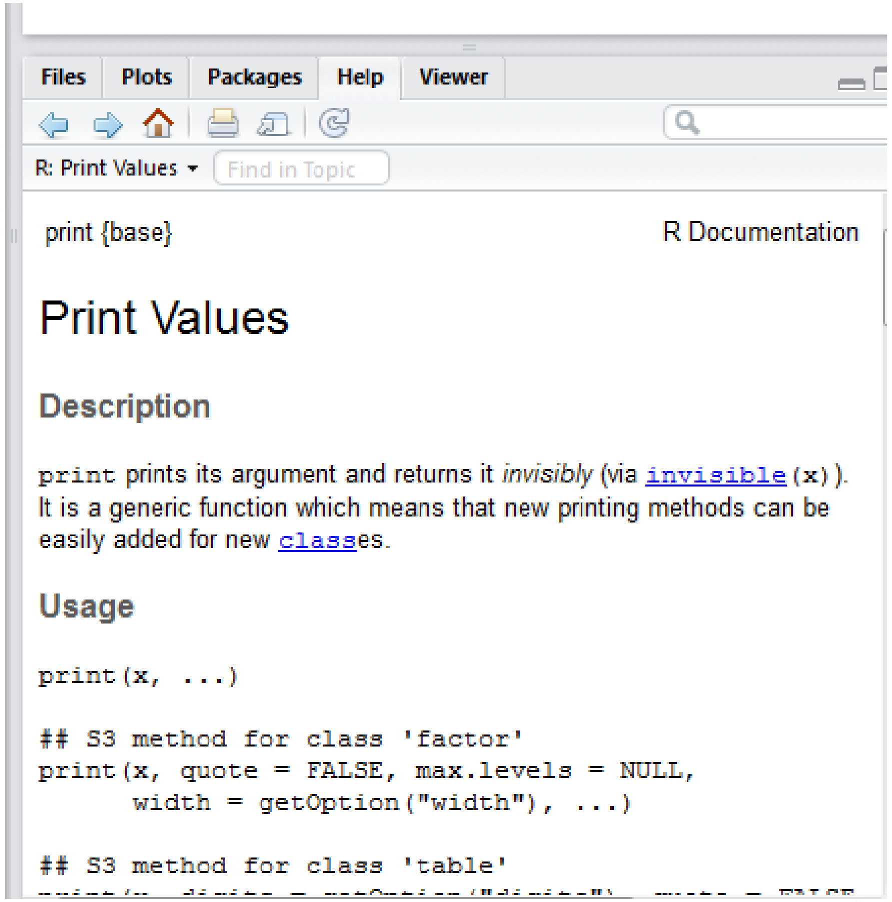
- to load package
  - library(“package name”)
  - or require(“package name”)

Help!!!

- Getting help
  - help(command) or ?command
  - example(command) to see examples



```
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> print("hello")
[1] "hello"
> a = 10
> b = 20
> a+b
[1] 30
> ?print
> 
```



print {base} R Documentation

Print Values

Description

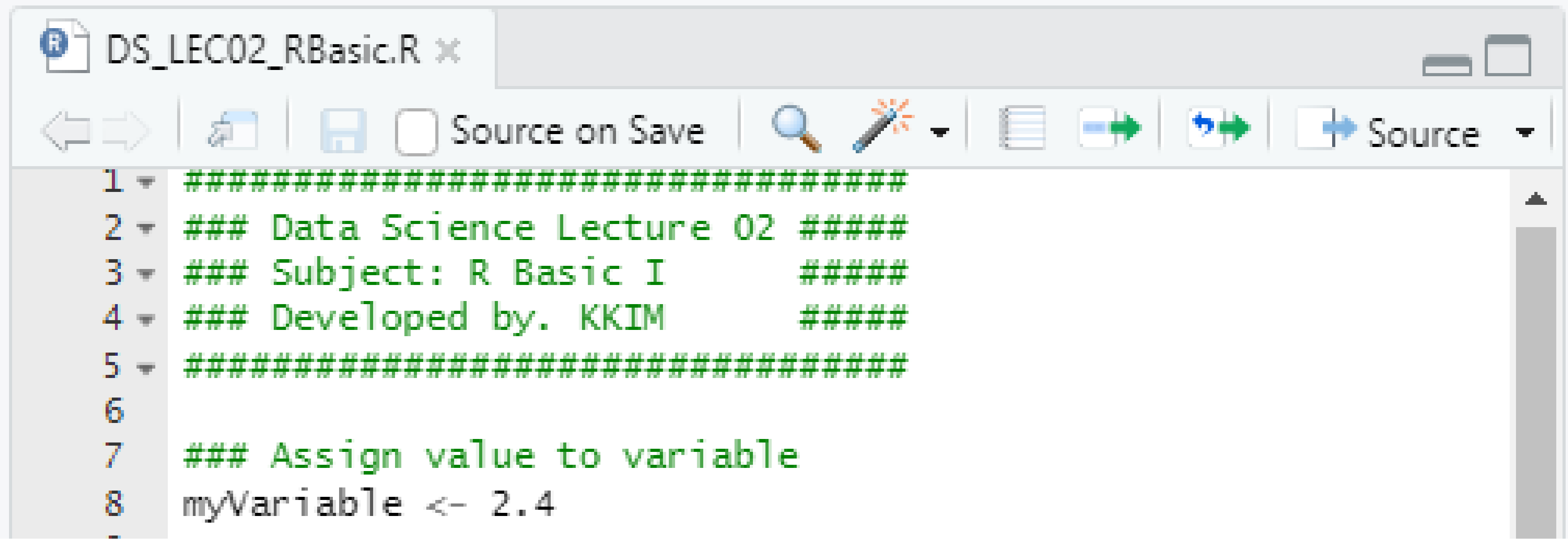
print prints its argument and returns it *invisibly* (via invisible(x)). It is a generic function which means that new printing methods can be easily added for new classes.

Usage

```
print(x, ...)

## S3 method for class 'factor'
print(x, quote = FALSE, max.levels = NULL,
      width = getOption("width"), ...)

## S3 method for class 'table'
```

Comments

- Comments are not operated in R
  - It is used to describe the program author, date, and nature of the code.
  - It is used to help programmers understand the code
- # or ctrl + shift + c



```
#################################
### Data Science Lecture 02 #####
### Subject: R Basic I      #####
### Developed by. KKIM      #####
#################################

### Assign value to variable
myVariable <- 2.4
```

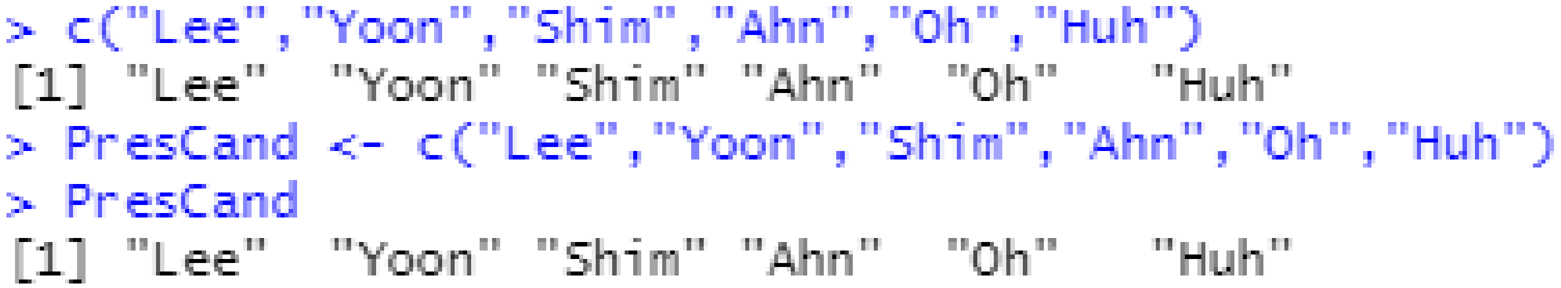
Vector

- A vector is a **sequence of data elements**.
  - A set with multiple elements
  - “sequence” meaning that it has an order, or it has an index number

  → Indexing and Slicing can be used

- c() function
  - A generic function that combines its arguments

  → fundamental function for creating a “vector”
  - Used for combining elements



```
> c("Lee","Yoon","Shim","Ahn","Oh","Huh")
[1] "Lee"  "Yoon" "Shim" "Ahn"  "Oh"   "Huh"
> PresCand <- c("Lee","Yoon","Shim","Ahn","Oh","Huh")
> PresCand
[1] "Lee"  "Yoon" "Shim" "Ahn"  "Oh"   "Huh"
```

DataFrame

- Conventional data set with rows and columns
  - e.g. student datasets may contain name(character), age(integer), major(factor), GPA(numeric, real number)…
  - Vectors and metrics can have values of the same data type
- A DataFrame has the variables of a data set as columns and the observations as rows.
  - “List of Vectors”
  - DataFrames can have variables(vectors) of the same length (and possibly different types)
- data.frame() function

List

- A list in R allows you to gather a variety of objects under one name (that is, the name of the list) in an ordered way.
    - List objects can be matrices, vectors, data frames, even other lists, etc.
    - It is not even required that these objects are related to each other in any way.
    - Fewer restrictions than the DataFrame

- list() function

R Indexing & Slicing

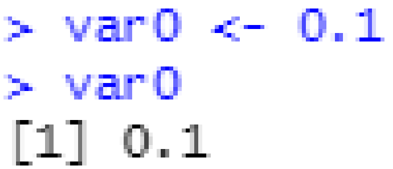
Indexing & Slicing

- Data structure
    - 0D: Scala
    - 1D: Vector
    - 2D: Matrix, DataFrame
    - 3D: List

- Indexing
    - Way of accessing specific value
- Slicing
    - Way of accessing specific values

Indexing & Slicing

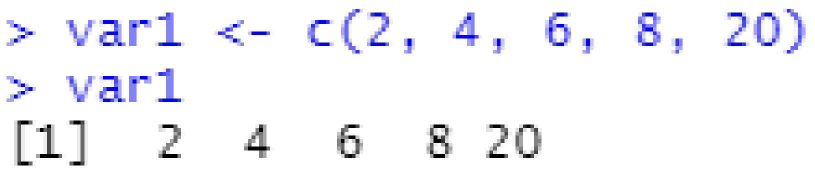
- 0D (scalar)
  - var0 <- 0.1
  - No need for indexing and slicing



```
> var0 <- 0.1
> var0
[1] 0.1
```

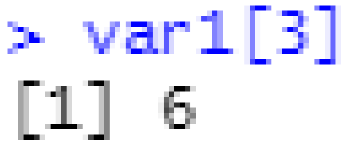
- 1D (vector)

var1 contains 5 elements.



```
> var1 <- c(2, 4, 6, 8, 20)
> var1
[1]  2  4  6  8 20
```

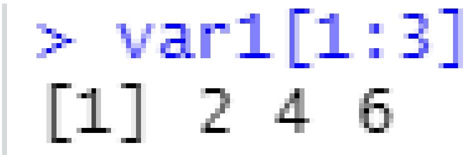
  - var1 <- c(2, 4, 6, 8, 20)
  - How can we access the 3rd element from var1?



```
> var1[3]
[1] 6
```

vec [*IndexNumber*]

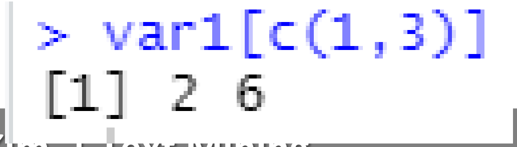
  - How can we access the 1st – 3rd element from var1?



```
> var1[1:3]
[1] 2 4 6
```

vec [*StartIndex* : *EndIndex*]

  - How can we access the 1st & 3rd element from var1?

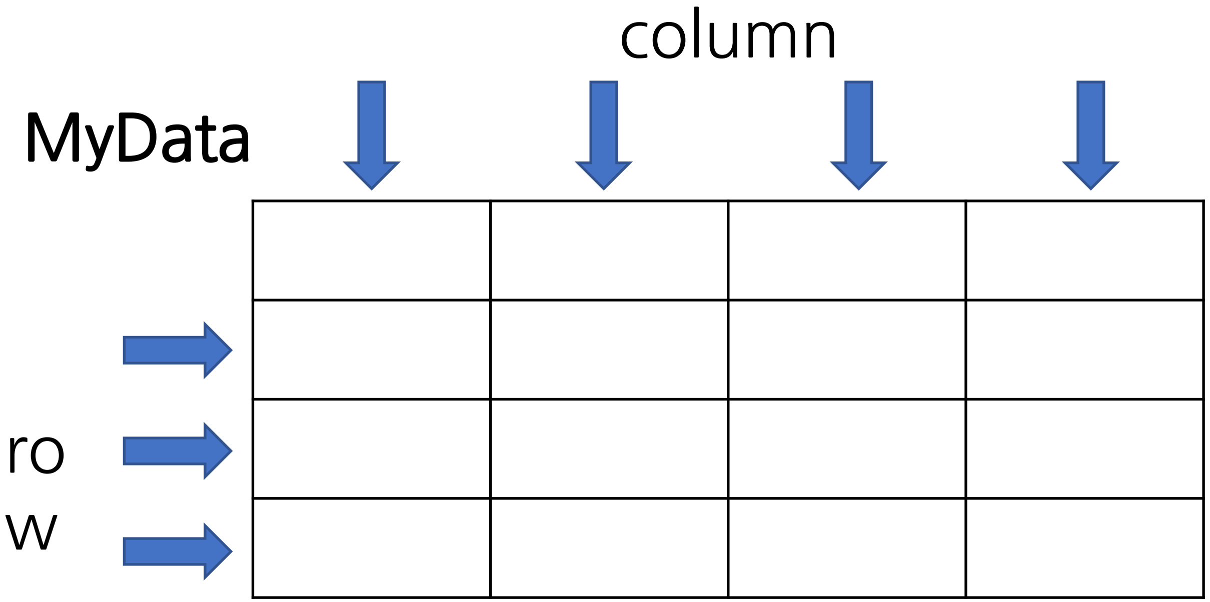


```
> var1[c(1,3)]
[1] 2 6
```

vec [c(*IndexNumber1*, *IndexNumber2*, ...)]

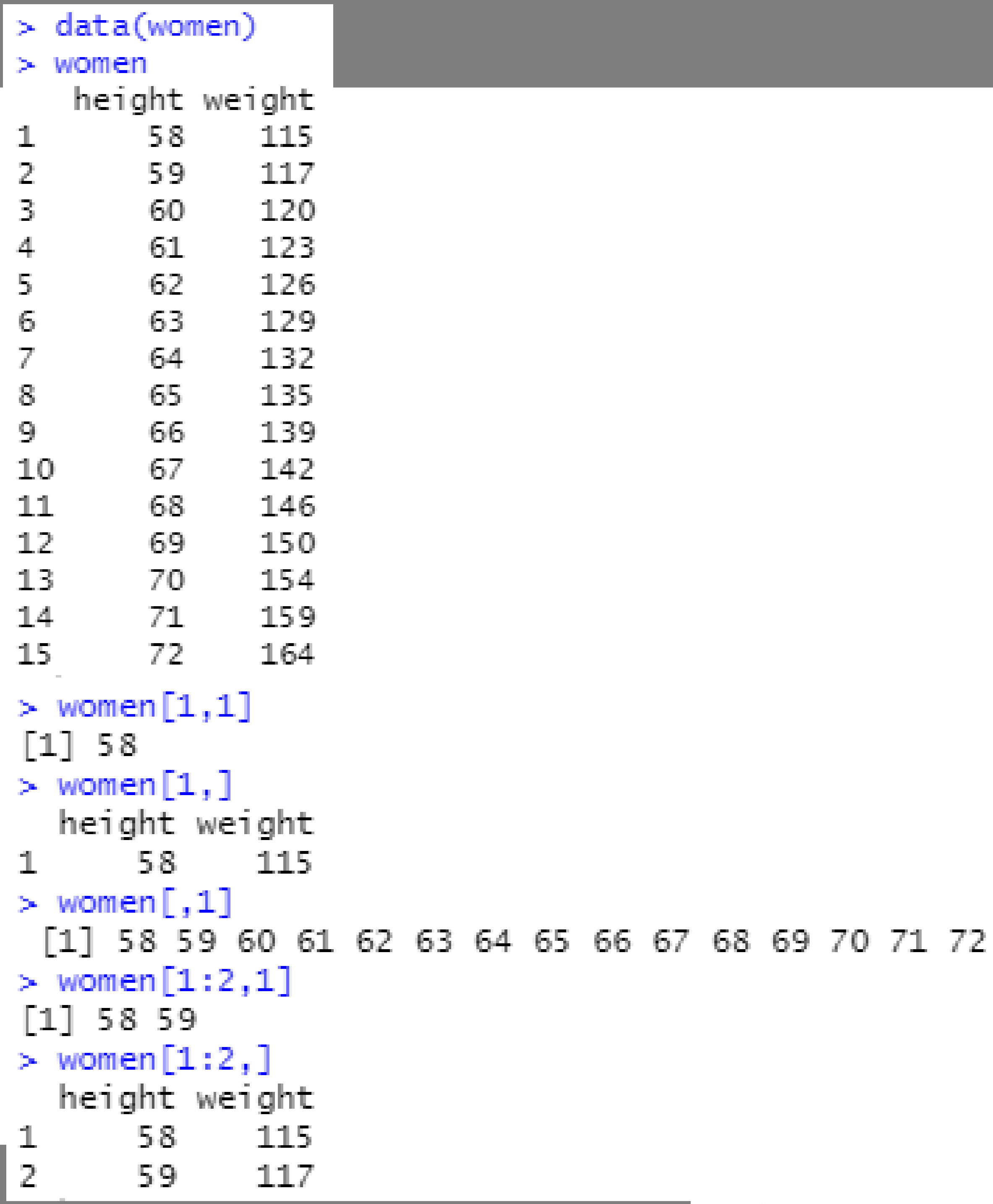
Indexing & Slicing

- 2D (matrix, DataFrame)
  - Method 1) Numbers



DF [*row-index*, *column-index*]

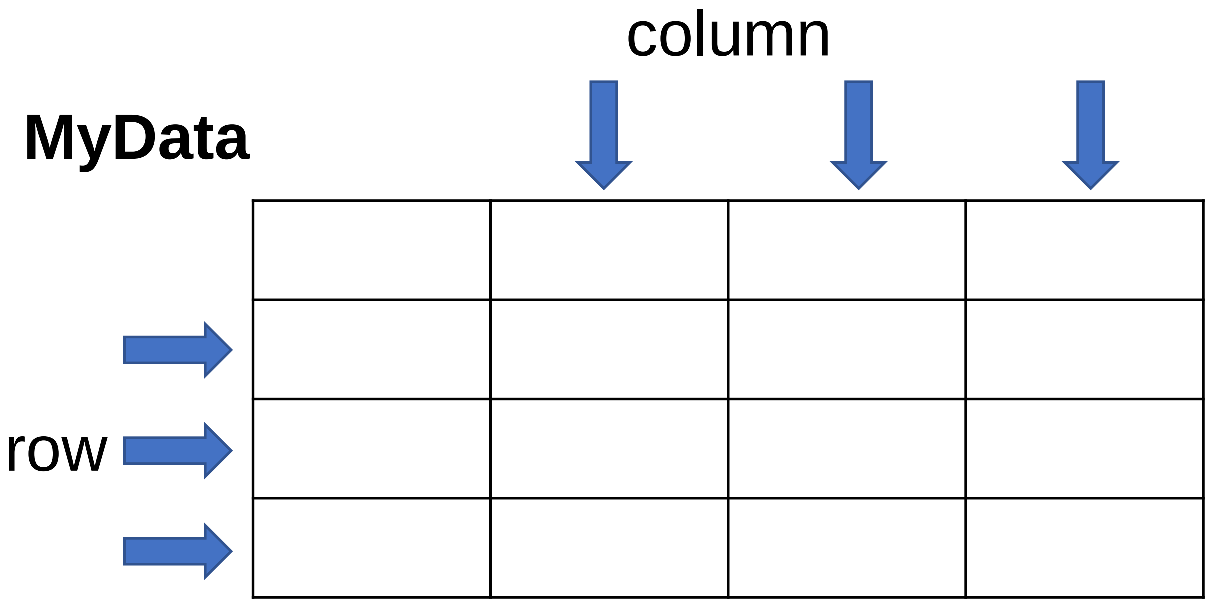
Either use
index number or slicing



```
> data(women)
> women
   height weight
1      58    115
2      59    117
3      60    120
4      61    123
5      62    126
6      63    129
7      64    132
8      65    135
9      66    139
10     67    142
11     68    146
12     69    150
13     70    154
14     71    159
15     72    164
> women[1,1]
[1] 58
> women[1,]
  height weight
1     58    115
> women[,1]
 [1] 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72
> women[1:2,1]
[1] 58 59
> women[1:2,]
  height weight
1     58    115
2     59    117
```

Indexing & Slicing

- 2D (matrix, DataFrame)
  - Method 1) Names (+Number)



column

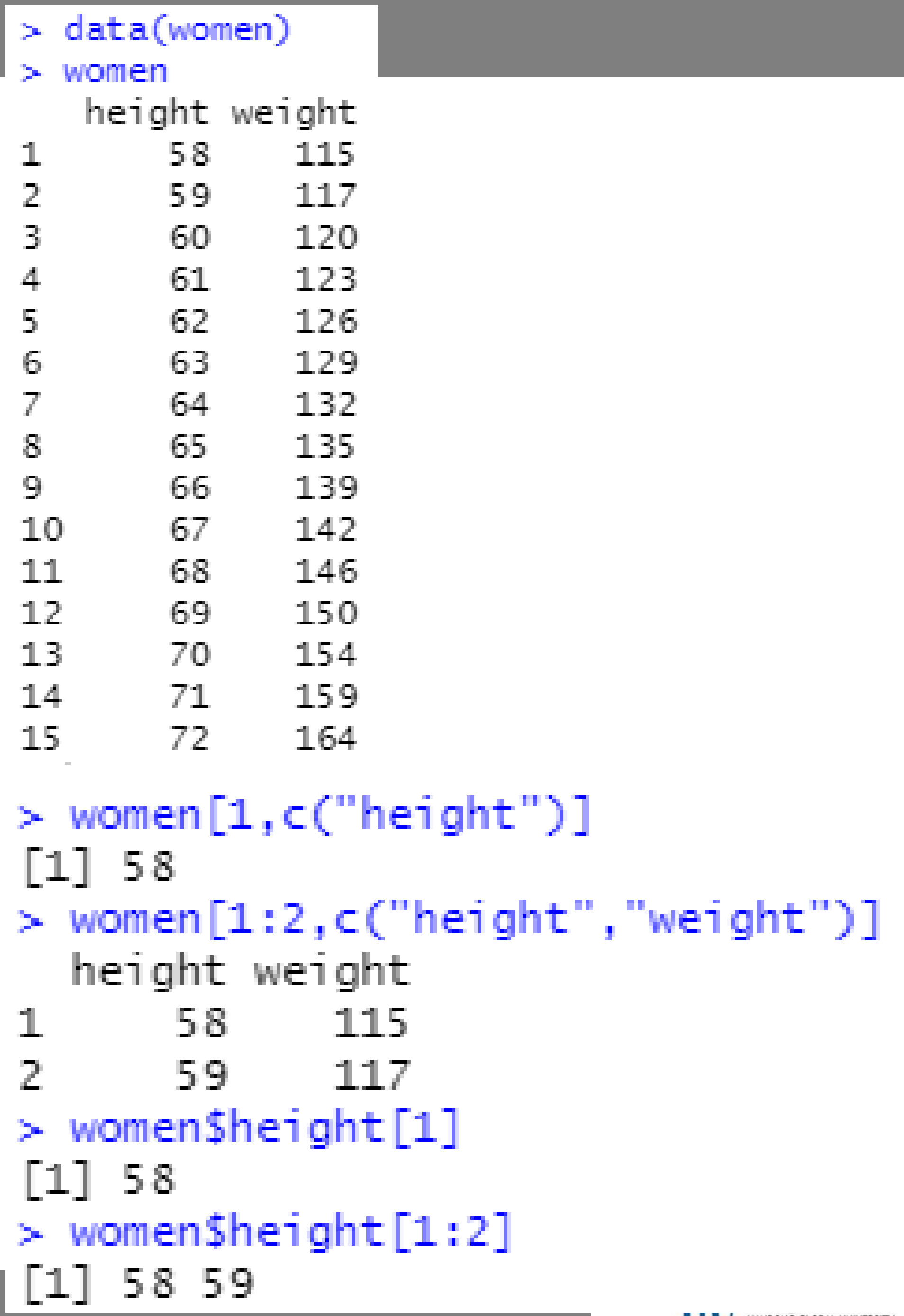
MyData

row

DF [*row-name, column-name*]

DF$column-name[column-number]

DF$column-name[column-slicing]

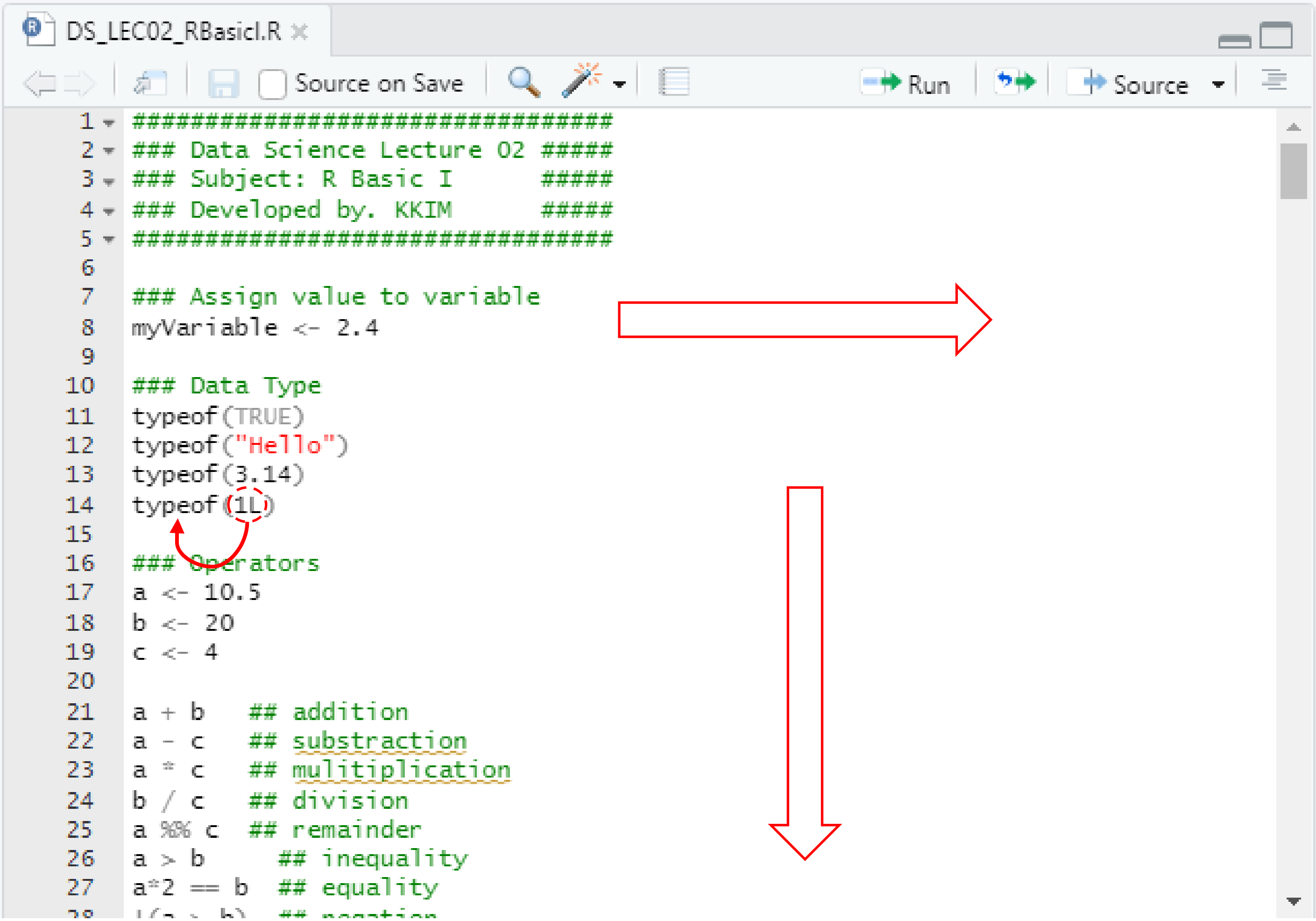


```
> data(women)
> women
   height weight
1      58    115
2      59    117
3      60    120
4      61    123
5      62    126
6      63    129
7      64    132
8      65    135
9      66    139
10     67    142
11     68    146
12     69    150
13     70    154
14     71    159
15     72    164

> women[1,c("height")]
[1] 58
> women[1:2,c("height","weight")]
  height weight
1     58    115
2     59    117
> women$height[1]
[1] 58
> women$height[1:2]
[1] 58 59
```

R Coding Overview

Code Writing

- When writing or reading programming codes, remember …
  - Left → Right
  - Top → Down
  - In → Out (when [ ] or () is used)



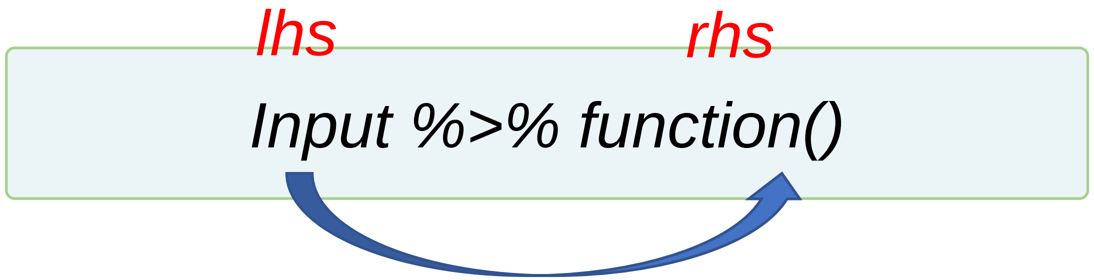
```
###################################
### Data Science Lecture 02 #####
### Subject: R Basic I      #####
### Developed by. KKIM      #####
###################################

### Assign value to variable
myVariable <- 2.4

### Data Type
typeof(TRUE)
typeof("Hello")
typeof(3.14)
typeof(1L)

### Operators
a <- 10.5
b <- 20
c <- 4

a + b   ## addition
a - c   ## substraction
a * c   ## mulitiplication
b / c   ## division
a %% c  ## remainder
a > b     ## inequality
a*2 == b  ## equality
!(a > b)  ## negation
```

head(AA[order(AA $Var, decreasing = TRUE),])

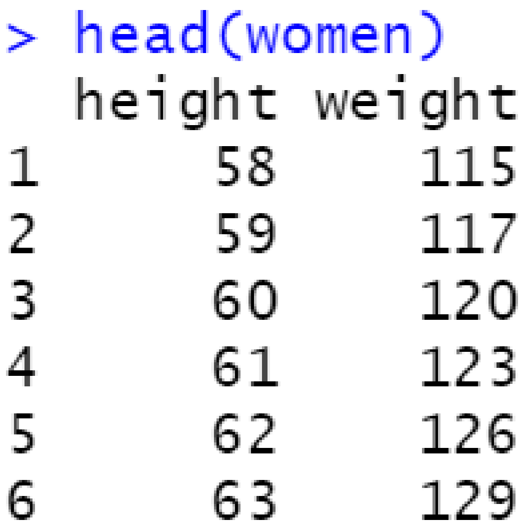
We can “understand” the code, but it will take some time…

Pipe Operator in R

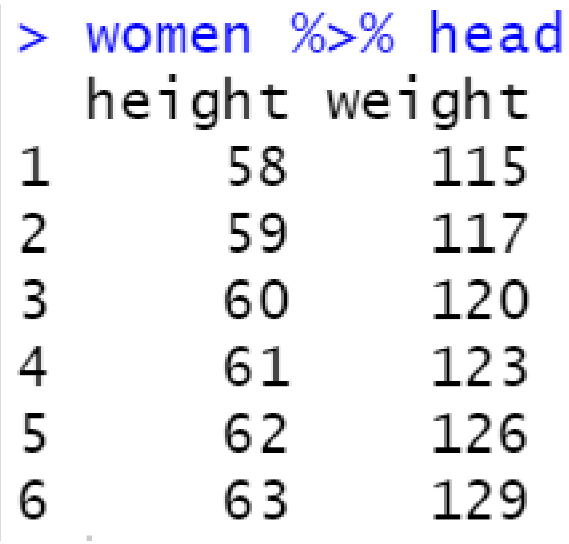
- Also known as “chain operator”


lhs
rhs
Input %>% function()

- Can be used with the ‘dplyr’ package
- Using the pipe operator, we can write and read codes from left to write. No more inside to outside



```
> head(women)
  height weight
1     58    115
2     59    117
3     60    120
4     61    123
5     62    126
6     63    129
```



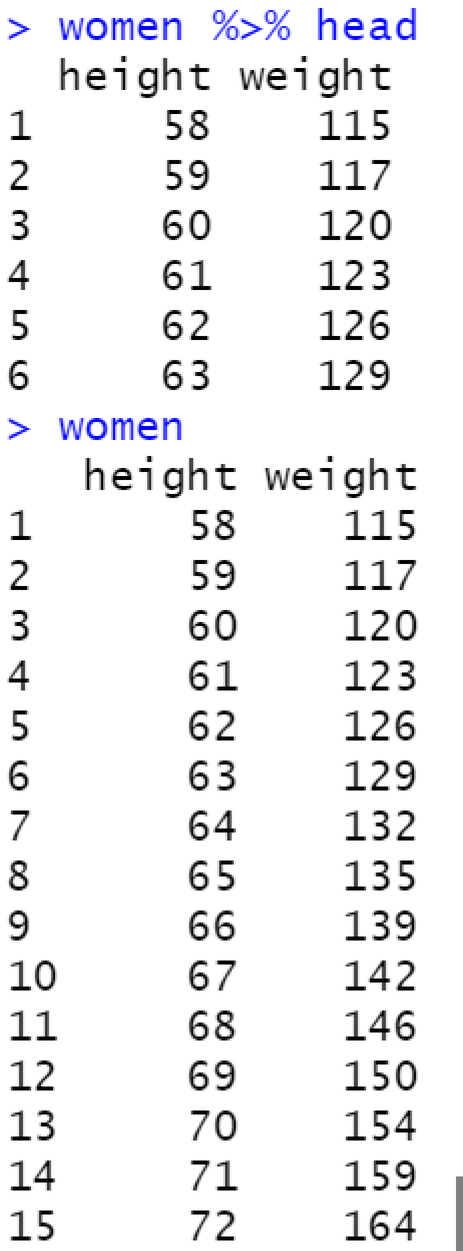
```
> women %>% head
  height weight
1     58    115
2     59    117
3     60    120
4     61    123
5     62    126
6     63    129
```

Women is used as an input for head() function

Pipe Operator in R

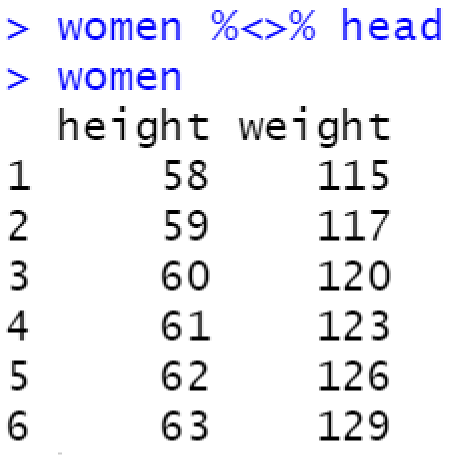
- Pipe operators
  - (dplyr package) %>%
  - (magrittr package) %<>%: Similar to %>%, but it “updates” the input

%>%



```
> women %>% head
  height weight
1     58    115
2     59    117
3     60    120
4     61    123
5     62    126
6     63    129
> women
   height weight
1      58    115
2      59    117
3      60    120
4      61    123
5      62    126
6      63    129
7      64    132
8      65    135
9      66    139
10     67    142
11     68    146
12     69    150
13     70    154
14     71    159
15     72    164
```

%<>%



```
> women %<>% head
> women
  height weight
1     58    115
2     59    117
3     60    120
4     61    123
5     62    126
6     63    129
```

Advantages of Using Pipe Operator

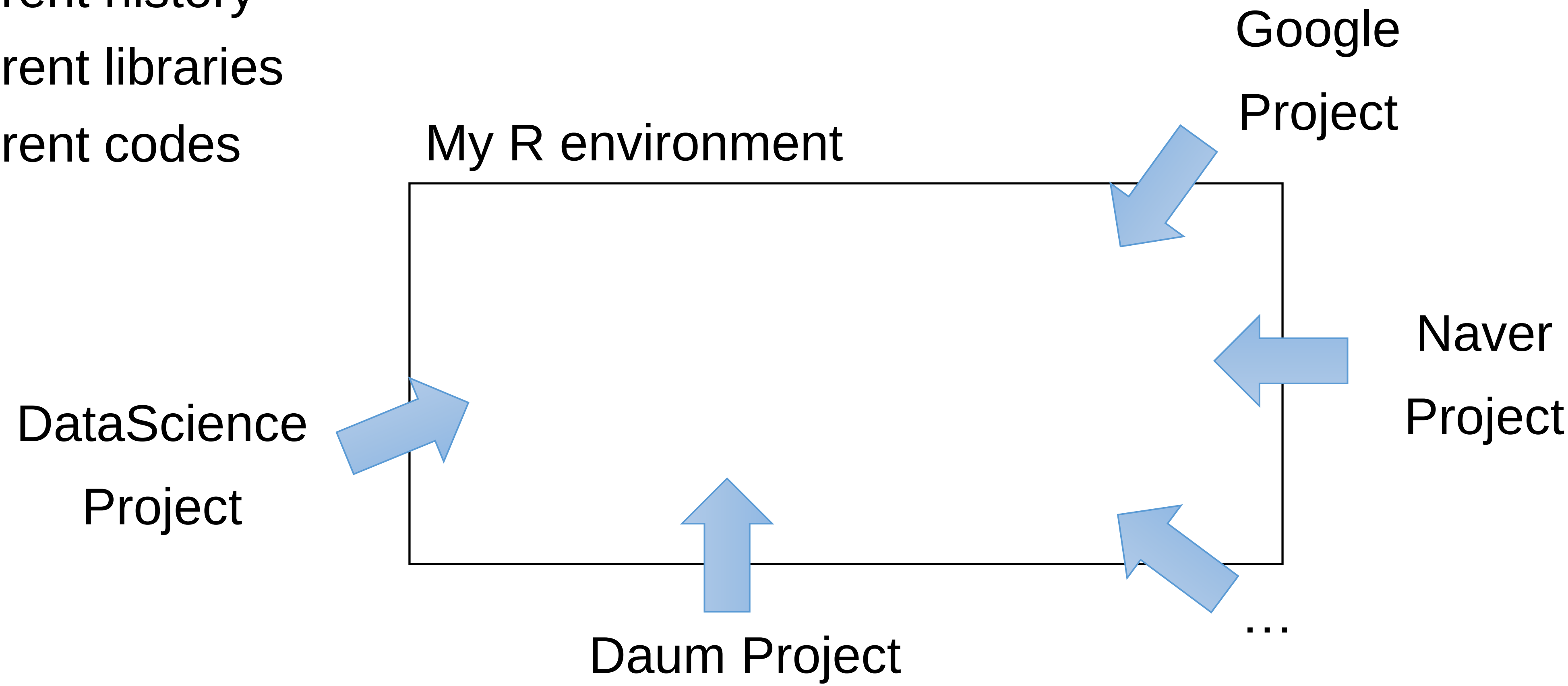
- Readability
  - Read from left to right
- Continuity
  - Write from left to right

- In Python, the pipe operator is not needed.
  - Object-oriented programming → class and instance
  - Also known as method chaining or flow programming

R Project &
Working Environment

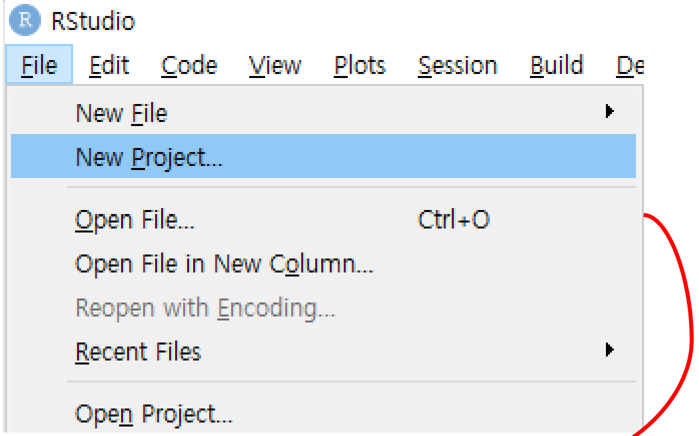
R Project

- To conduct various data analysis projects, it is necessary to "well” manage each project
  - Different data
  - Different history
  - Different libraries
  - Different codes

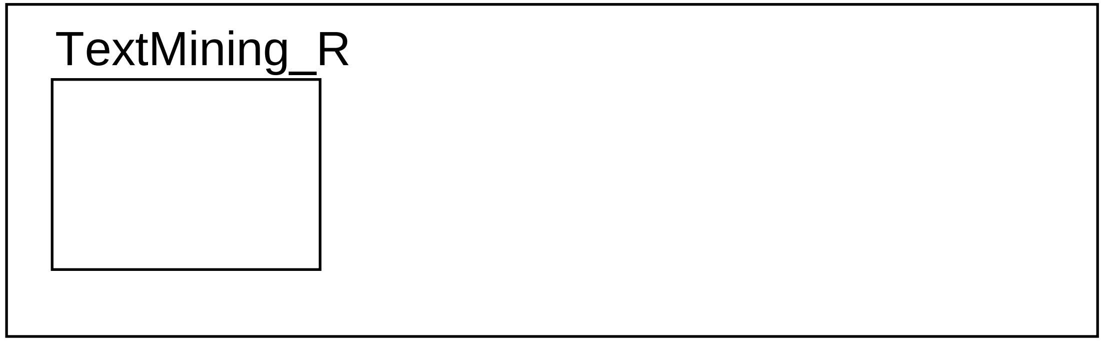


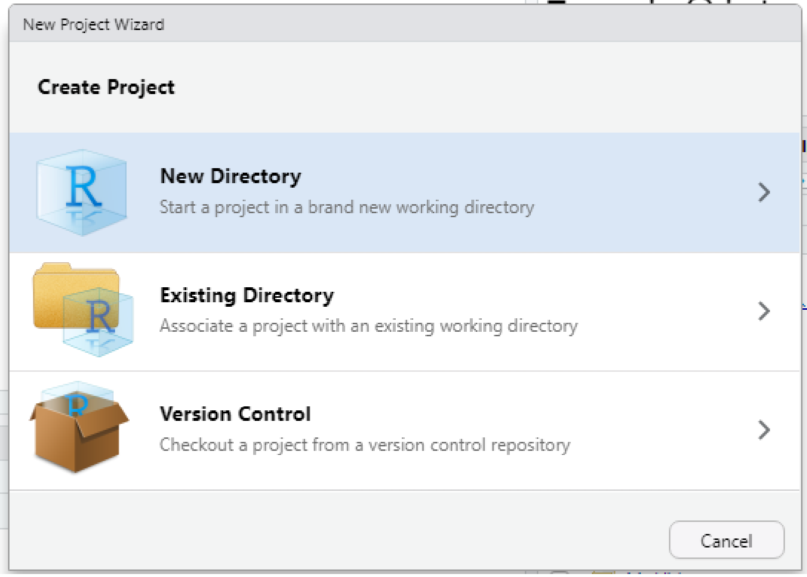
R Project

- R project



G:/내 드라이브/[Lecture]/TextMining



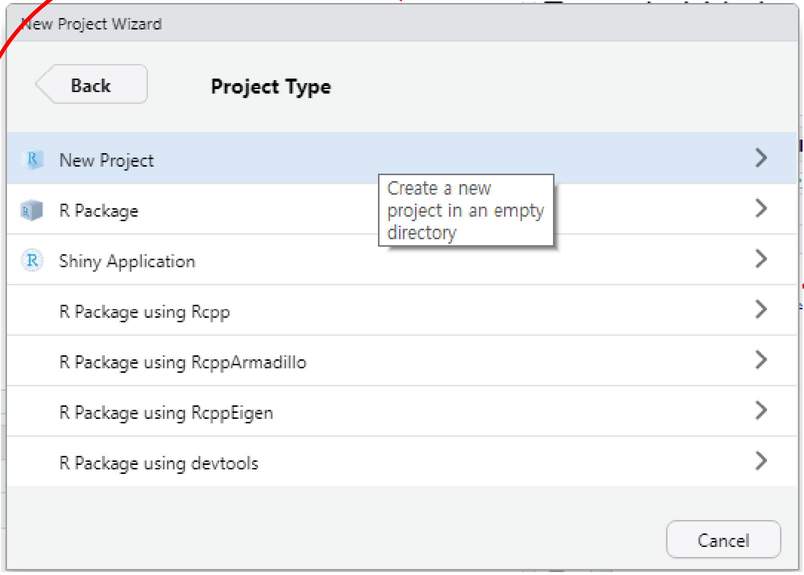

New Project Wizard

Create Project

New Directory
Start a project in a brand new working directory

Existing Directory
Associate a project with an existing working directory

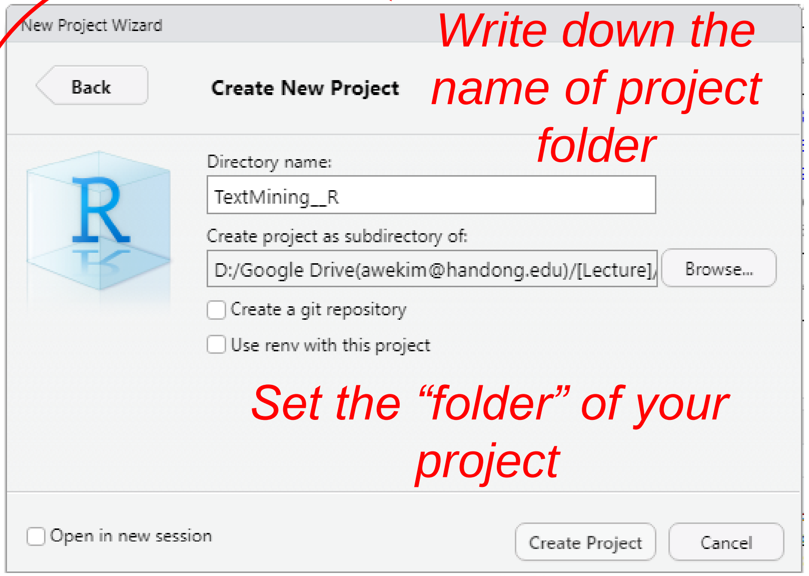
Version Control
Checkout a project from a version control repository


New Project Wizard

Project Type

New Project
R Package
Shiny Application
R Package using Rcpp
R Package using RcppArmadillo
R Package using RcppEigen
R Package using devtools

Create a new project in an empty directory


New Project Wizard

Create New Project

Directory name:
TextMining__R

Create project as subdirectory of:
D:/Google Drive(awekim@handong.edu)/[Lecture]/

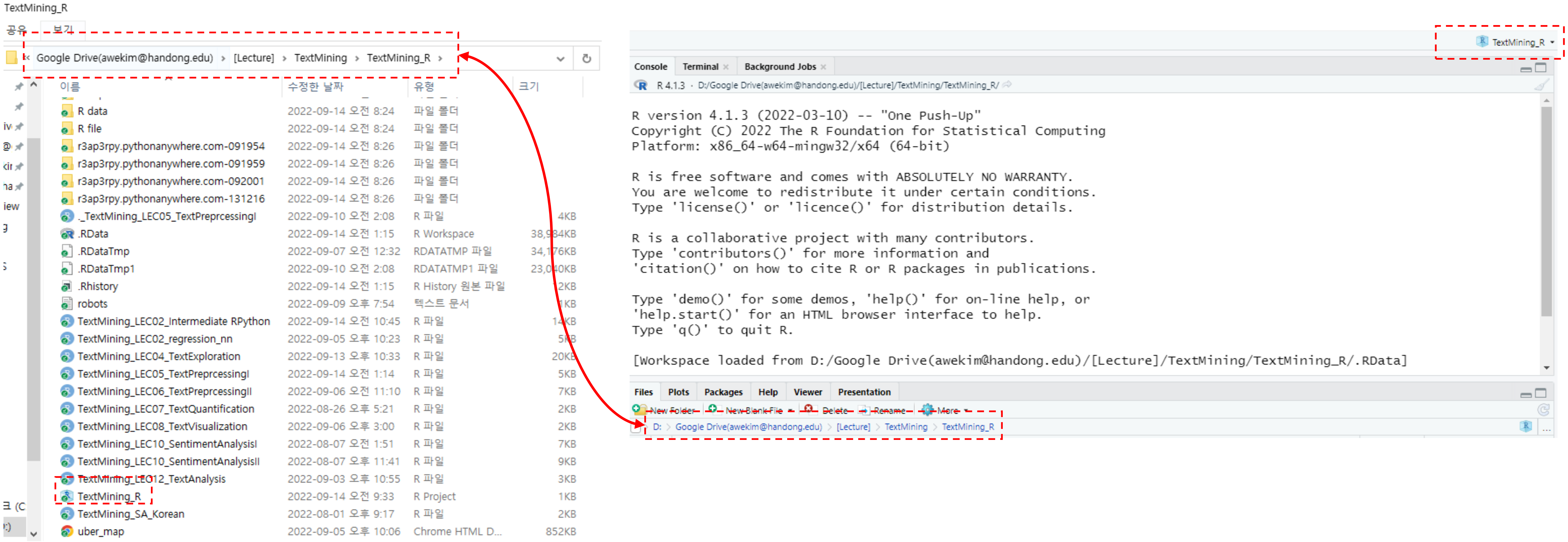
□ Create a git repository
□ Use renv with this project
□ Open in new session

Write down the name of project folder

Set the "folder" of your project

R Project

- R project
  - For each R project, the working directory is “fixed” to the folder that you created.
  - With this, we can use "relative path“ easily to import data set to the R environment.



TextMining_R

공유 보기

Google Drive(awekim@handong.edu) › [Lecture] › TextMining › TextMining_R ›

이름	수정한 날짜	유형	크기
R data	2022-09-14 오전 8:24	파일 폴더	
R file	2022-09-14 오전 8:24	파일 폴더	
r3ap3rpy.pythonanywhere.com-091954	2022-09-14 오전 8:26	파일 폴더	
r3ap3rpy.pythonanywhere.com-091959	2022-09-14 오전 8:26	파일 폴더	
r3ap3rpy.pythonanywhere.com-092001	2022-09-14 오전 8:26	파일 폴더	
r3ap3rpy.pythonanywhere.com-131216	2022-09-14 오전 8:26	파일 폴더	
._TextMining_LEC05_TextPreprcessingI	2022-09-10 오전 2:08	R 파일	4KB
.RData	2022-09-14 오전 1:15	R Workspace	38,984KB
.RDataTmp	2022-09-07 오전 12:32	RDATATMP 파일	34,176KB
.RDataTmp1	2022-09-10 오전 2:08	RDATATMP1 파일	23,040KB
.Rhistory	2022-09-14 오전 1:15	R History 원본 파일	12KB
robots	2022-09-09 오후 7:54	텍스트 문서	1KB
TextMining_LEC02_Intermediate RPython	2022-09-14 오전 10:45	R 파일	14KB
TextMining_LEC02_regression_nn	2022-09-05 오후 10:23	R 파일	5KB
TextMining_LEC04_TextExploration	2022-09-13 오후 10:33	R 파일	20KB
TextMining_LEC05_TextPreprcessingI	2022-09-14 오전 1:14	R 파일	5KB
TextMining_LEC06_TextPreprcessingII	2022-09-06 오전 11:10	R 파일	7KB
TextMining_LEC07_TextQuantification	2022-08-26 오후 5:21	R 파일	2KB
TextMining_LEC08_TextVisualization	2022-09-06 오후 3:00	R 파일	2KB
TextMining_LEC10_SentimentAnalysisI	2022-08-07 오전 1:51	R 파일	7KB
TextMining_LEC10_SentimentAnalysisII	2022-08-07 오후 11:41	R 파일	9KB
TextMining_LEC12_TextAnalysis	2022-09-03 오후 10:55	R 파일	3KB
TextMining_R	2022-09-14 오전 9:33	R Project	1KB
TextMining_SA_Korean	2022-08-01 오후 9:17	R 파일	2KB
uber_map	2022-09-05 오후 10:06	Chrome HTML D...	852KB

TextMining_R

Console | Terminal | Background Jobs

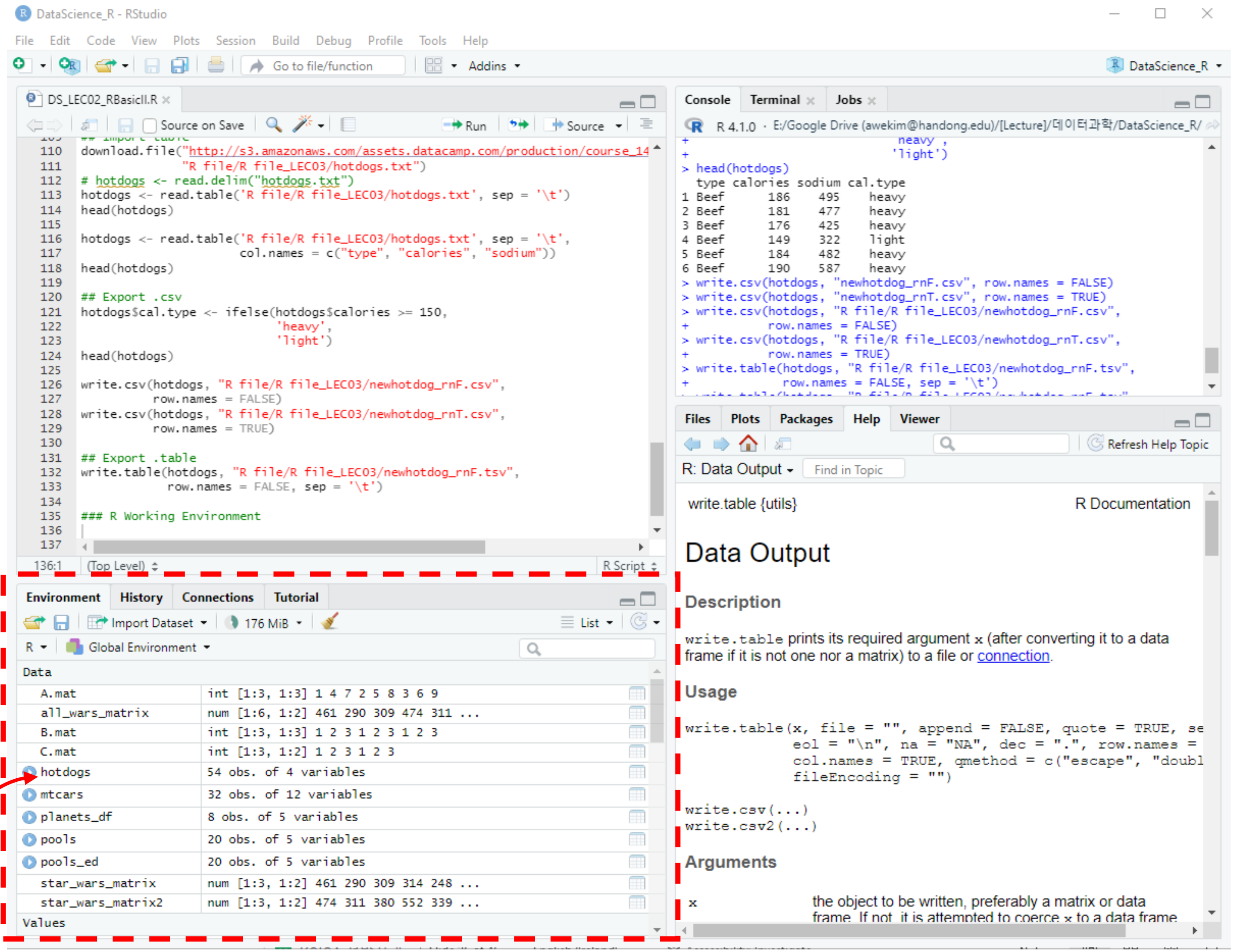
R 4.1.3 · D:/Google Drive(awekim@handong.edu)/[Lecture]/TextMining/TextMining_R/

```
R version 4.1.3 (2022-03-10) -- "One Push-Up"
Copyright (C) 2022 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

[Workspace loaded from D:/Google Drive(awekim@handong.edu)/[Lecture]/TextMining/TextMining_R/.RData]
```

D: › Google Drive(awekim@handong.edu) › [Lecture] › TextMining › TextMining_R

.RData file

- In R working environment, any type of R variables can be saved as .Rdata file
  - .RData file: data file used in the R working environment
- Advantages of using .RData files
  - Easy to track the changes
  - Easy to continue with the task



```
## Import table
download.file("http://s3.amazonaws.com/assets.datacamp.com/production/course_14
              "R file/R file_LEC03/hotdogs.txt")
# hotdogs <- read.delim("hotdogs.txt")
hotdogs <- read.table('R file/R file_LEC03/hotdogs.txt', sep = '\t')
head(hotdogs)

hotdogs <- read.table('R file/R file_LEC03/hotdogs.txt', sep = '\t',
                      col.names = c("type", "calories", "sodium"))
head(hotdogs)

## Export .csv
hotdogs$cal.type <- ifelse(hotdogs$calories >= 150,
                           'heavy',
                           'light')
head(hotdogs)

write.csv(hotdogs, "R file/R file_LEC03/newhotdog_rnF.csv",
          row.names = FALSE)
write.csv(hotdogs, "R file/R file_LEC03/newhotdog_rnT.csv",
          row.names = TRUE)

## Export .table
write.table(hotdogs, "R file/R file_LEC03/newhotdog_rnF.tsv",
            row.names = FALSE, sep = '\t')

### R Working Environment
```

R 4.1.0 · E:/Google Drive (awekim@handong.edu)/[Lecture]/데이터과학/DataScience_R/

```
+                            'heavy',
+                            'light')
> head(hotdogs)
  type calories sodium cal.type
1 Beef      186    495    heavy
2 Beef      181    477    heavy
3 Beef      176    425    heavy
4 Beef      149    322    light
5 Beef      184    482    heavy
6 Beef      190    587    heavy
> write.csv(hotdogs, "newhotdog_rnF.csv", row.names = FALSE)
> write.csv(hotdogs, "newhotdog_rnT.csv", row.names = TRUE)
> write.csv(hotdogs, "R file/R file_LEC03/newhotdog_rnF.csv",
+           row.names = FALSE)
> write.csv(hotdogs, "R file/R file_LEC03/newhotdog_rnT.csv",
+           row.names = TRUE)
> write.table(hotdogs, "R file/R file_LEC03/newhotdog_rnF.tsv",
+             row.names = FALSE, sep = '\t')
```

Data	
A.mat	int [1:3, 1:3] 1 4 7 2 5 8 3 6 9
all_wars_matrix	num [1:6, 1:2] 461 290 309 474 311 ...
B.mat	int [1:3, 1:3] 1 2 3 1 2 3 1 2 3
C.mat	int [1:3, 1:2] 1 2 3 1 2 3
hotdogs	54 obs. of 4 variables
mtcars	32 obs. of 12 variables
planets_df	8 obs. of 5 variables
pools	20 obs. of 5 variables
pools_ed	20 obs. of 5 variables
star_wars_matrix	num [1:3, 1:2] 461 290 309 314 248 ...
star_wars_matrix2	num [1:3, 1:2] 474 311 380 552 339 ...
Values	

write.table {utils} R Documentation

Data Output

Description

`write.table` prints its required argument `x` (after converting it to a data frame if it is not one nor a matrix) to a file or connection.

Usage

```
write.table(x, file = "", append = FALSE, quote = TRUE, se
            eol = "\n", na = "NA", dec = ".", row.names =
            col.names = TRUE, qmethod = c("escape", "doubl
            fileEncoding = "")

write.csv(...)
write.csv2(...)
```

Arguments

`x` the object to be written, preferably a matrix or data frame. If not, it is attempted to coerce `x` to a data frame

raw data

R Working Environment

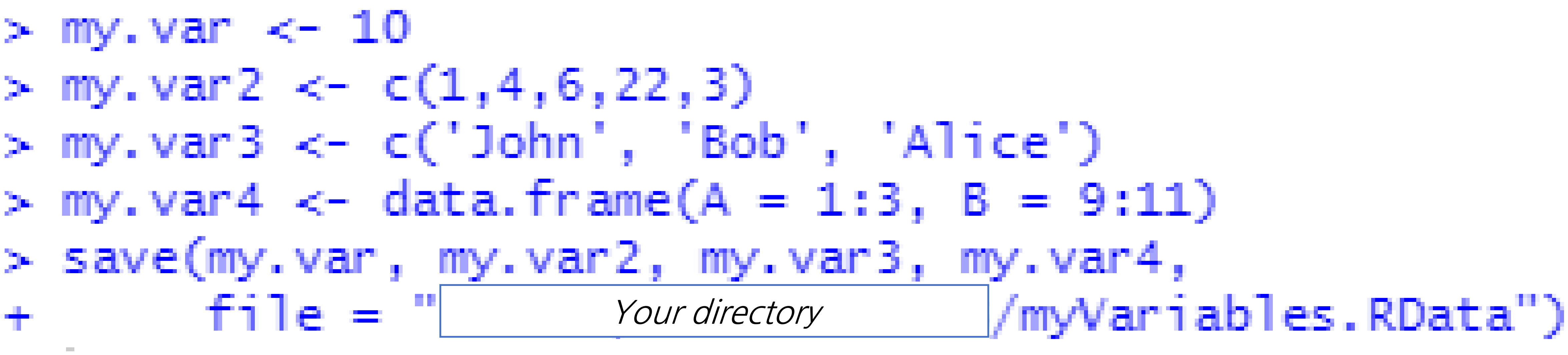
.RData file

- When to use .RData file?
  - Share your work in R with your colleagues
  - Continue to work on the same data on a different PC
  - Restore your work from some unexpected errors, etc.

Save .RData file – Data

- save() function
  - Used for managing variables.
  - Powerful when managing large data.

save(*Variables*, file=‘*FileDirectory/FileName.RData*’)



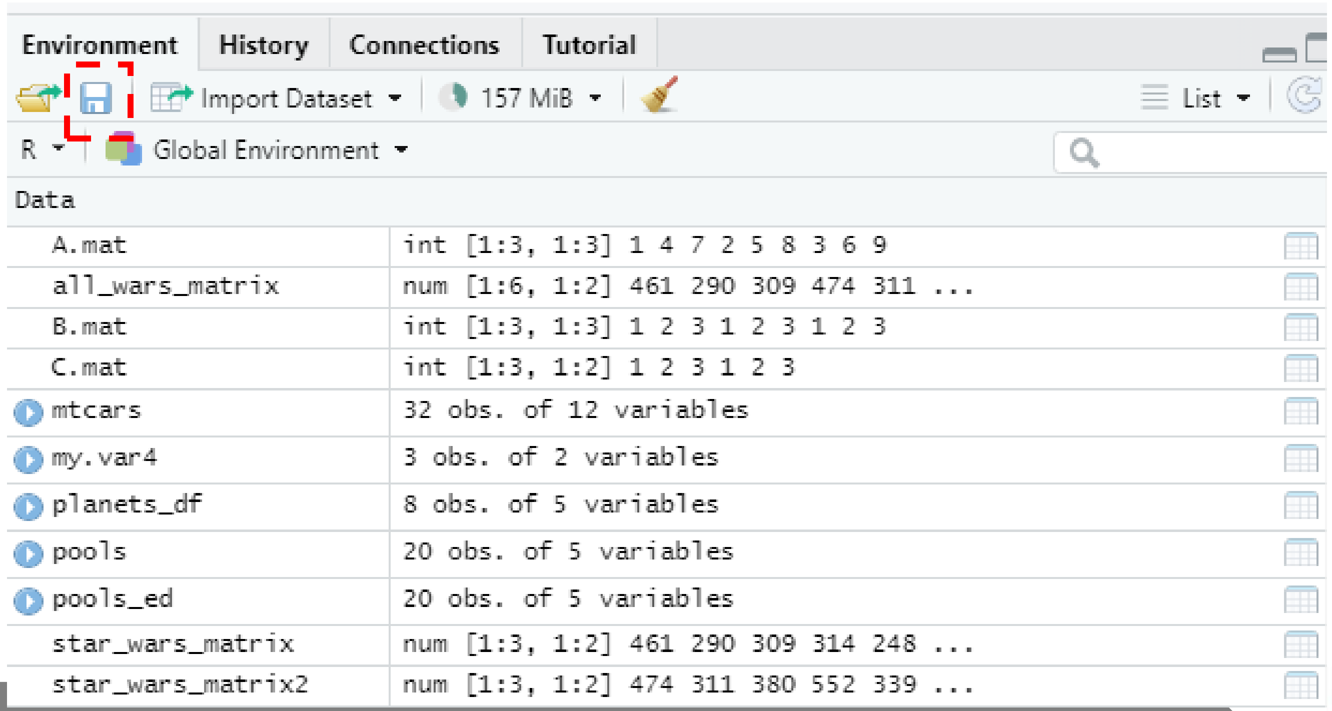
```
> my.var <- 10
> my.var2 <- c(1,4,6,22,3)
> my.var3 <- c('John', 'Bob', 'Alice')
> my.var4 <- data.frame(A = 1:3, B = 9:11)
> save(my.var, my.var2, my.var3, my.var4,
+      file = "Your directory/myVariables.RData")
```

Save .RData file – Working Environment

- Method 1) Save .Rdata of R working environment
  - Use save() function

```
save(list = ls(), file='FileDirectory/FileName.RData')
```

  - Press disk button

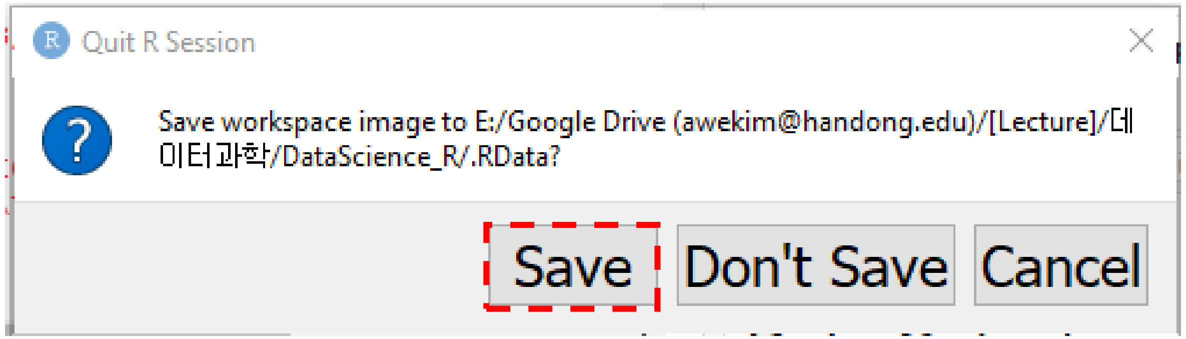


Data

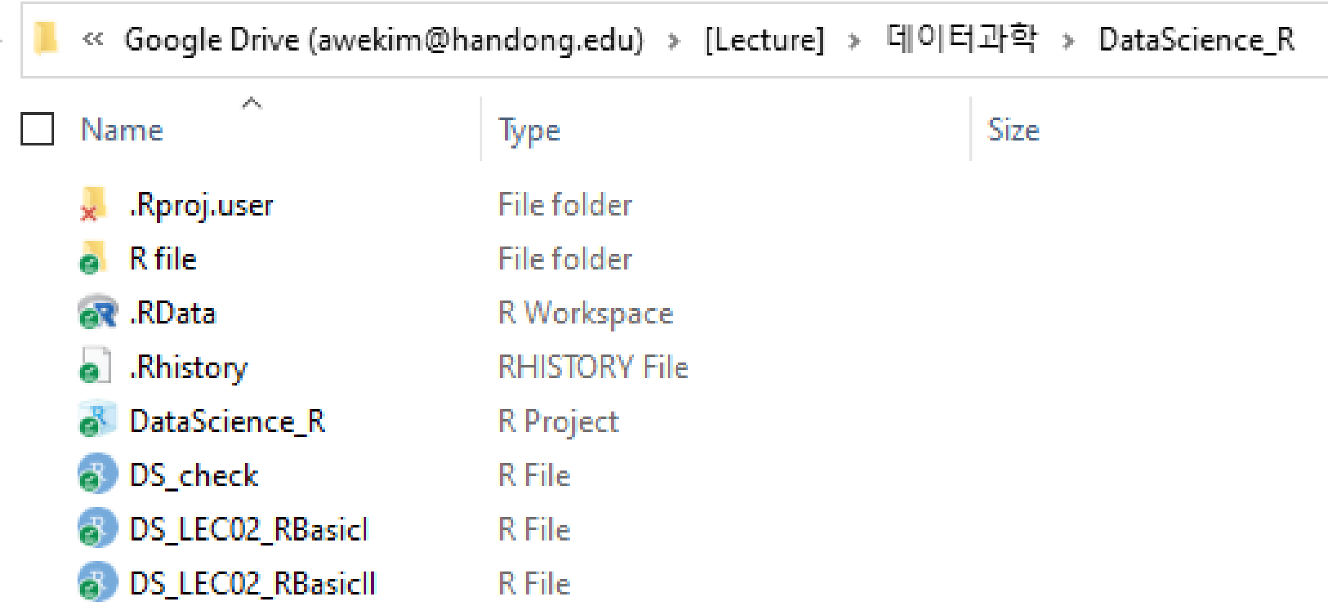
A.mat	int [1:3, 1:3] 1 4 7 2 5 8 3 6 9
all_wars_matrix	num [1:6, 1:2] 461 290 309 474 311 ...
B.mat	int [1:3, 1:3] 1 2 3 1 2 3 1 2 3
C.mat	int [1:3, 1:2] 1 2 3 1 2 3
mtcars	32 obs. of 12 variables
my.var4	3 obs. of 2 variables
planets_df	8 obs. of 5 variables
pools	20 obs. of 5 variables
pools_ed	20 obs. of 5 variables
star_wars_matrix	num [1:3, 1:2] 461 290 309 314 248 ...
star_wars_matrix2	num [1:3, 1:2] 474 311 380 552 339 ...

Save .RData file – Working Environment

- Method 2) Save .Rdata when closing R Session
  - By doing so, all of the tasks and variables in the R project can be saved.
  - Used for managing the whole R project



Save workspace image to E:/Google Drive (awekim@handong.edu)/[Lecture]/데이터과학/DataScience_R/.RData?


« Google Drive (awekim@handong.edu) › [Lecture] › 데이터과학 › DataScience_R

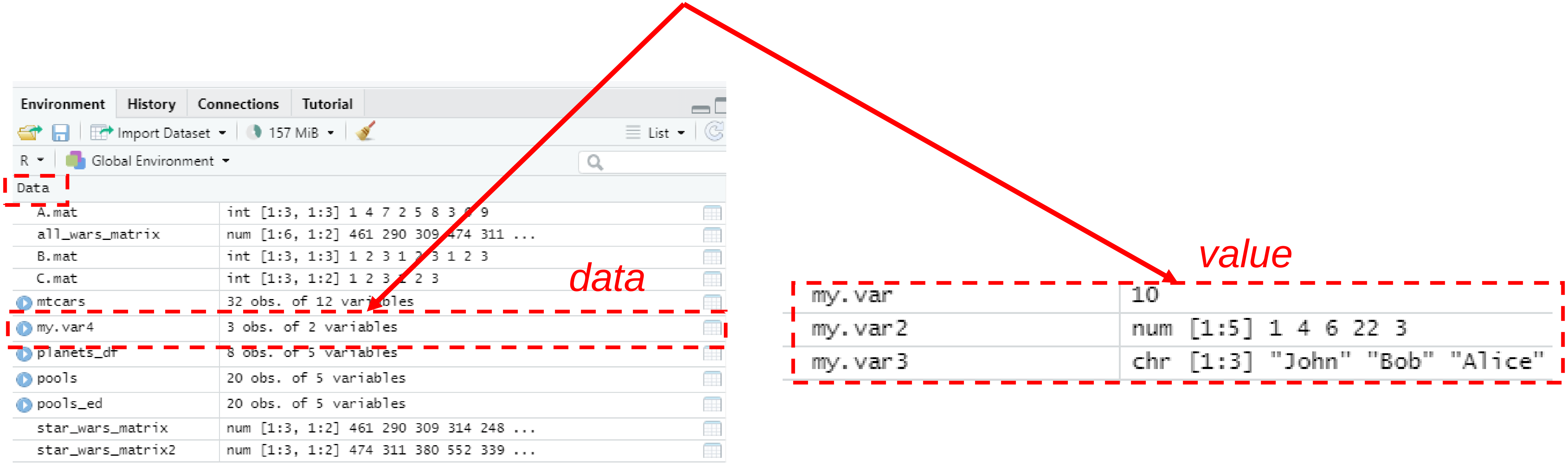
Name	Type	Size
.Rproj.user	File folder	
R file	File folder	
.RData	R Workspace	
.Rhistory	RHISTORY File	
DataScience_R	R Project	
DS_check	R File	
DS_LEC02_RBasicI	R File	
DS_LEC02_RBasicII	R File	

Load .RData file

- load() function

load(file=‘*FileDirectory/FileName.RData*’)

```
> load(file = "Your directory/myVariables.RData")
```



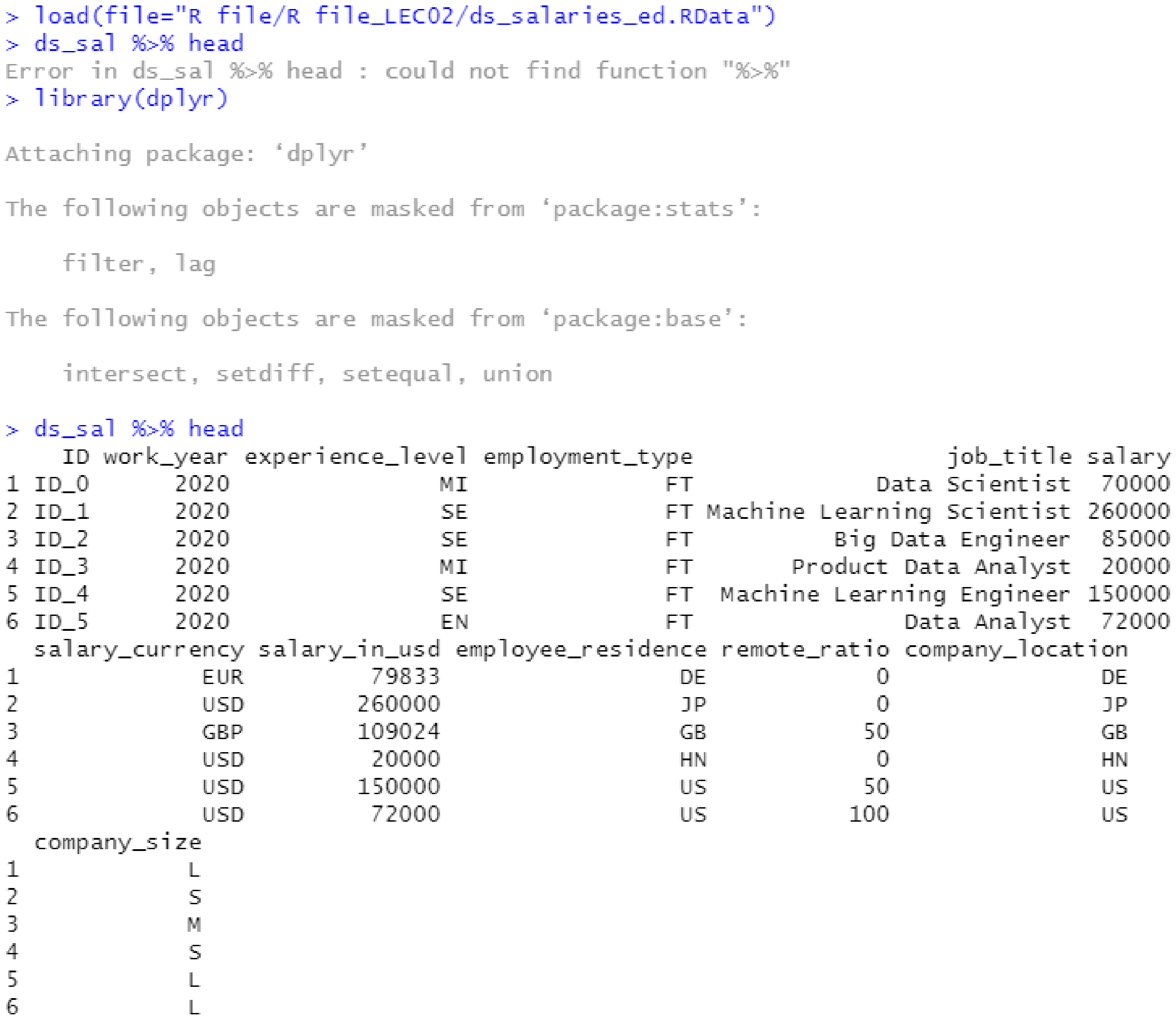
```
Environment   History   Connections   Tutorial
Import Dataset ▾   157 MiB ▾   List ▾
R ▾   Global Environment ▾
Data
  A.mat               int [1:3, 1:3] 1 4 7 2 5 8 3 6 9
  all_wars_matrix     num [1:6, 1:2] 461 290 309 474 311 ...
  B.mat               int [1:3, 1:3] 1 2 3 1 2 3 1 2 3
  C.mat               int [1:3, 1:2] 1 2 3 1 2 3
  mtcars              32 obs. of 12 variables
  my.var4             3 obs. of 2 variables
  planets_df          8 obs. of 5 variables
  pools               20 obs. of 5 variables
  pools_ed            20 obs. of 5 variables
  star_wars_matrix    num [1:3, 1:2] 461 290 309 314 248 ...
  star_wars_matrix2   num [1:3, 1:2] 474 311 380 552 339 ...
```

```
my.var    10
my.var2   num [1:5] 1 4 6 22 3
my.var3   chr [1:3] "John" "Bob" "Alice"
```

Data Manipulation with dplyr I

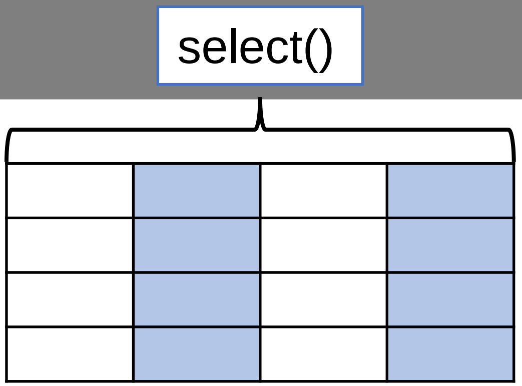
dplyr package

- dplyr package
  - a grammar of data manipulation, providing a consistent set of verbs that helps you solve the most common data manipulation challenges
  - select(), filter(), mutate(), arrange(), group_by(), summarise()



```
> load(file="R file/R file_LEC02/ds_salaries_ed.RData")
> ds_sal %>% head
Error in ds_sal %>% head : could not find function "%>%"
> 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

> ds_sal %>% head
    ID work_year experience_level employment_type                  job_title salary
1 ID_0      2020               MI              FT             Data Scientist  70000
2 ID_1      2020               SE              FT Machine Learning Scientist 260000
3 ID_2      2020               SE              FT          Big Data Engineer  85000
4 ID_3      2020               MI              FT       Product Data Analyst  20000
5 ID_4      2020               SE              FT  Machine Learning Engineer 150000
6 ID_5      2020               EN              FT               Data Analyst  72000
  salary_currency salary_in_usd employee_residence remote_ratio company_location
1             EUR         79833                 DE            0               DE
2             USD        260000                 JP            0               JP
3             GBP        109024                 GB           50               GB
4             USD         20000                 HN            0               HN
5             USD        150000                 US           50               US
6             USD         72000                 US          100               US
  company_size
1            L
2            S
3            M
4            S
5            L
6            L
```

dplyr::select()

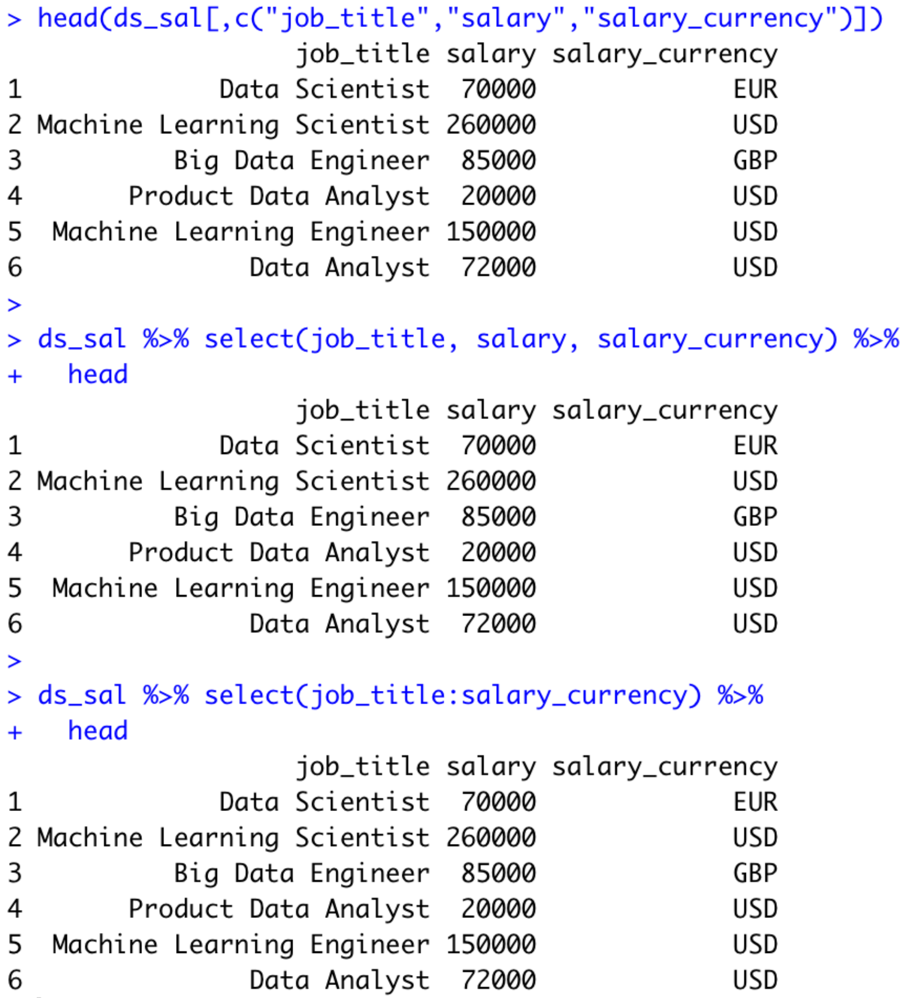


- select()
  - picks columns (variables) either with names or index
  - Alternatives: (columns) indexing + slicing, etc.

Select columns of job_title, salary, and salary_currency

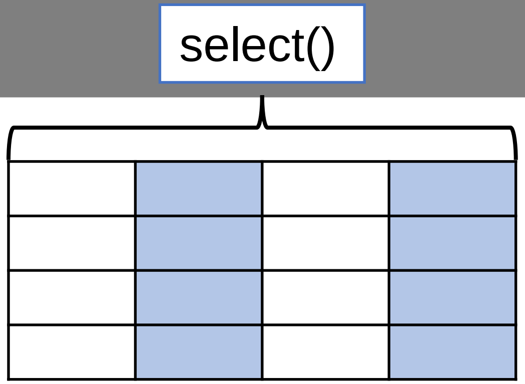
Select columns of job_title, salary, and salary_currency

Select columns from job_title to salary_currency

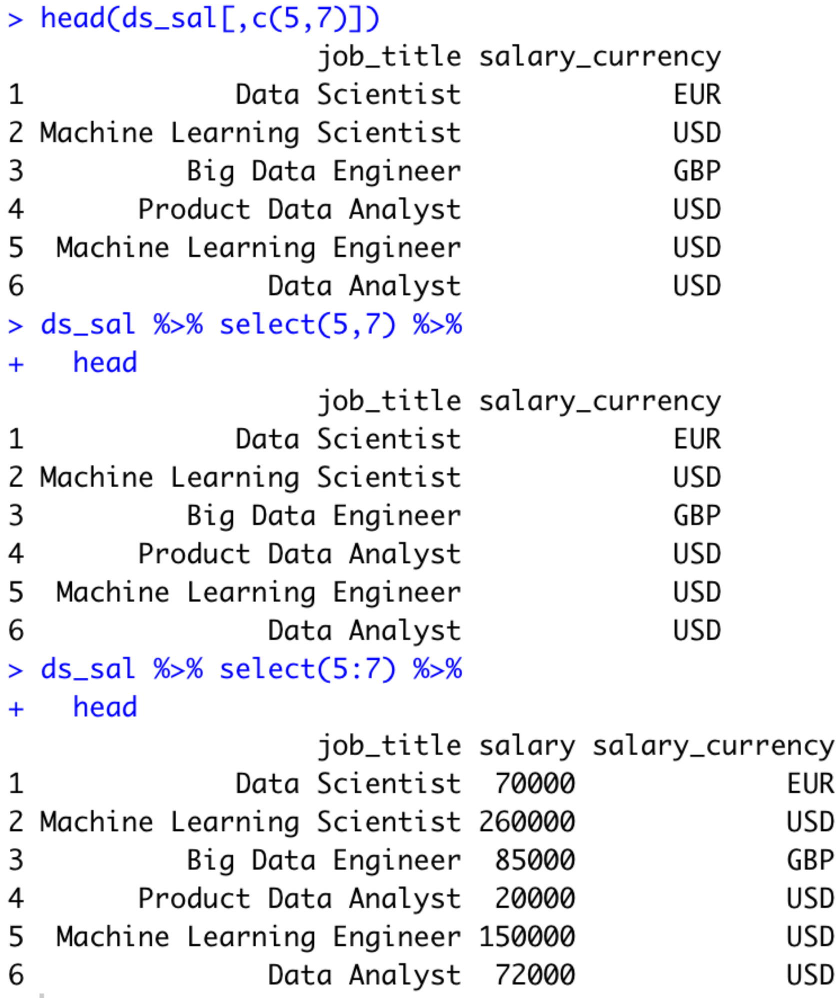


```
> head(ds_sal[,c("job_title","salary","salary_currency")])
                   job_title salary salary_currency
1             Data Scientist  70000             EUR
2 Machine Learning Scientist 260000             USD
3          Big Data Engineer  85000             GBP
4       Product Data Analyst  20000             USD
5  Machine Learning Engineer 150000             USD
6               Data Analyst  72000             USD
>
> ds_sal %>% select(job_title, salary, salary_currency) %>%
+   head
                   job_title salary salary_currency
1             Data Scientist  70000             EUR
2 Machine Learning Scientist 260000             USD
3          Big Data Engineer  85000             GBP
4       Product Data Analyst  20000             USD
5  Machine Learning Engineer 150000             USD
6               Data Analyst  72000             USD
>
> ds_sal %>% select(job_title:salary_currency) %>%
+   head
                   job_title salary salary_currency
1             Data Scientist  70000             EUR
2 Machine Learning Scientist 260000             USD
3          Big Data Engineer  85000             GBP
4       Product Data Analyst  20000             USD
5  Machine Learning Engineer 150000             USD
6               Data Analyst  72000             USD
```

dplyr::select()



- select()
  - picks columns (variables) either with names or index
  - Alternatives: (columns) indexing + slicing, etc.



Select column number 5 and 7

```
> head(ds_sal[,c(5,7)])
                  job_title salary_currency
1            Data Scientist             EUR
2 Machine Learning Scientist            USD
3         Big Data Engineer             GBP
4      Product Data Analyst             USD
5  Machine Learning Engineer            USD
6              Data Analyst             USD
```

Select column number 5 and 7

```
> ds_sal %>% select(5,7) %>%
+   head
                  job_title salary_currency
1            Data Scientist             EUR
2 Machine Learning Scientist            USD
3         Big Data Engineer             GBP
4      Product Data Analyst             USD
5  Machine Learning Engineer            USD
6              Data Analyst             USD
```

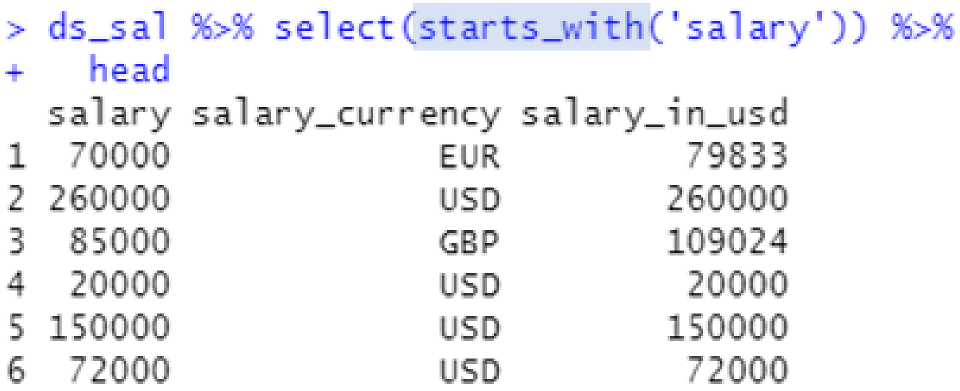
Select column number from 5 to 7

```
> ds_sal %>% select(5:7) %>%
+   head
                  job_title salary salary_currency
1            Data Scientist  70000             EUR
2 Machine Learning Scientist 260000            USD
3         Big Data Engineer  85000             GBP
4      Product Data Analyst  20000             USD
5  Machine Learning Engineer 150000            USD
6              Data Analyst  72000             USD
```

dplyr::select()

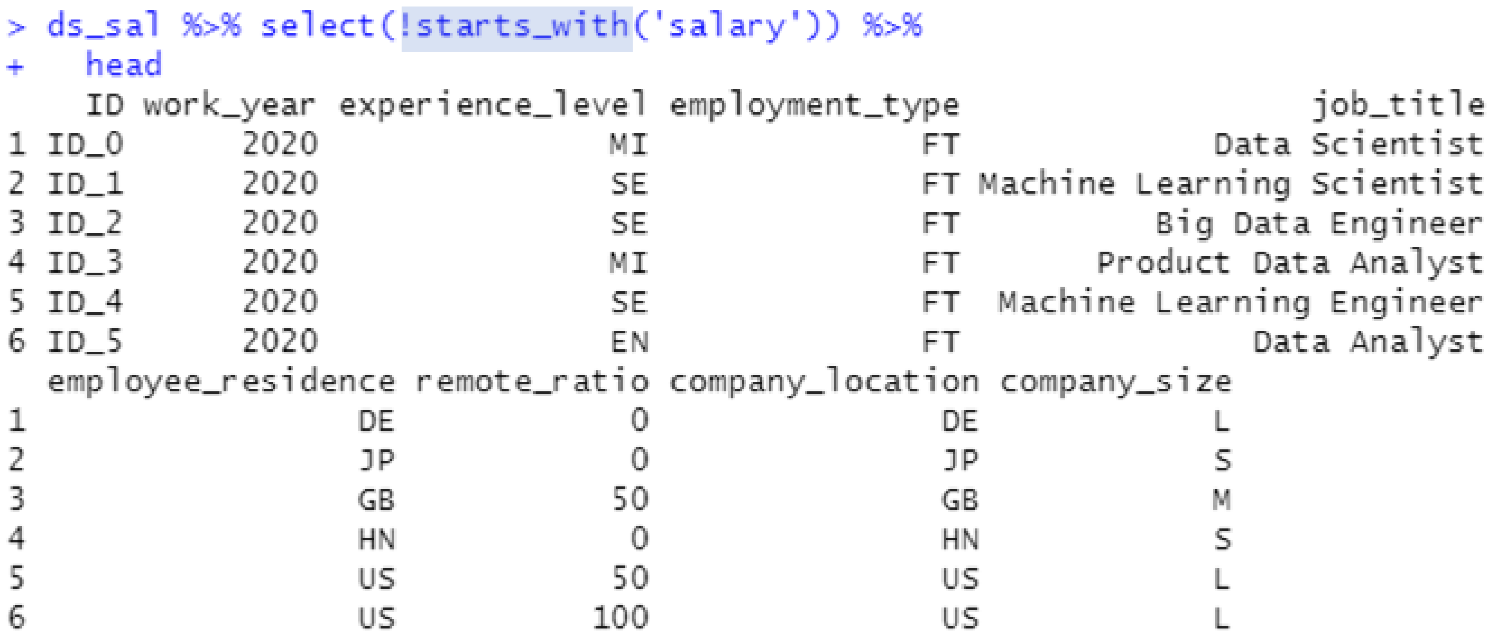
- select() + _with
  - Select columns by recognizing a certain “pattern”
  - starts_with & ends_with

Select columns that start with ‘salary’



```
> ds_sal %>% select(starts_with('salary')) %>%
+   head
  salary salary_currency salary_in_usd
1  70000             EUR         79833
2 260000             USD        260000
3  85000             GBP        109024
4  20000             USD         20000
5 150000             USD        150000
6  72000             USD         72000
```

Select columns that does not start with ‘salary’



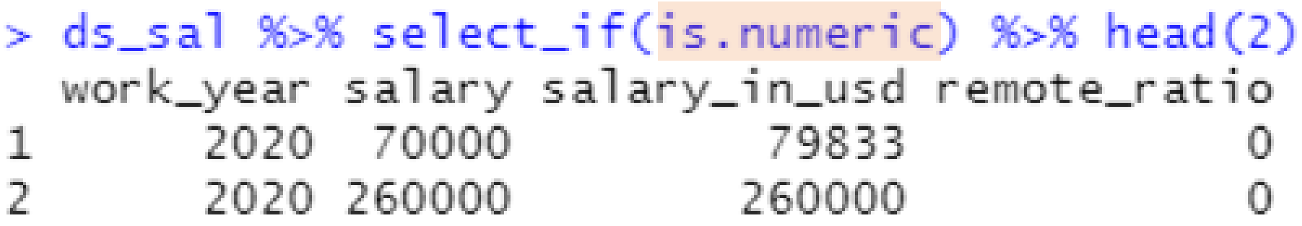
```
> ds_sal %>% select(!starts_with('salary')) %>%
+   head
   ID work_year experience_level employment_type                  job_title
1 ID_0      2020               MI              FT             Data Scientist
2 ID_1      2020               SE              FT Machine Learning Scientist
3 ID_2      2020               SE              FT          Big Data Engineer
4 ID_3      2020               MI              FT       Product Data Analyst
5 ID_4      2020               SE              FT  Machine Learning Engineer
6 ID_5      2020               EN              FT               Data Analyst
  employee_residence remote_ratio company_location company_size
1                 DE            0               DE            L
2                 JP            0               JP            S
3                 GB           50               GB            M
4                 HN            0               HN            S
5                 US           50               US            L
6                 US          100               US            L
```

Select columns that starts with “salary” or “company”?

dplyr::select()

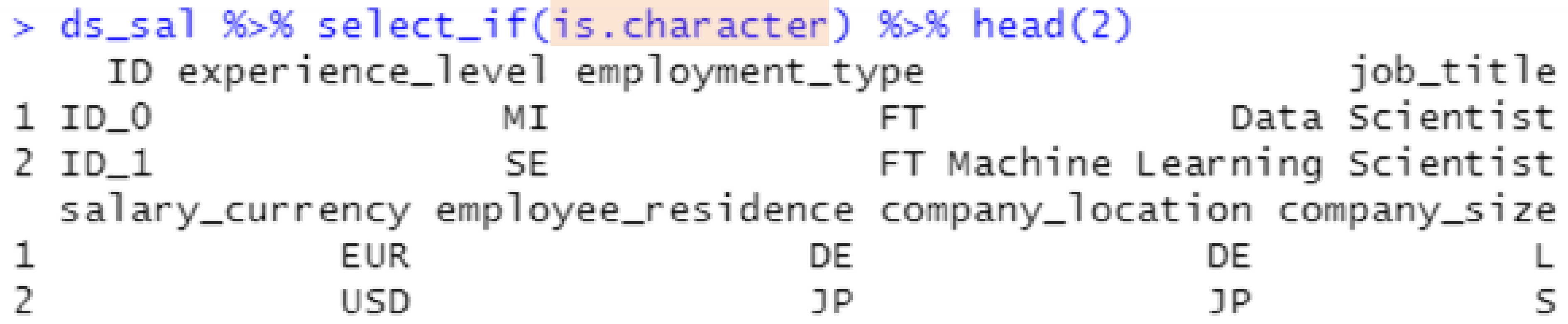
- select_if()
  - Select columns by recognizing a certain “pattern”
  - Useful when selecting specific types of columns

Select columns with numeric values



```
> ds_sal %>% select_if(is.numeric) %>% head(2)
  work_year salary salary_in_usd remote_ratio
1      2020  70000         79833            0
2      2020 260000        260000            0
```

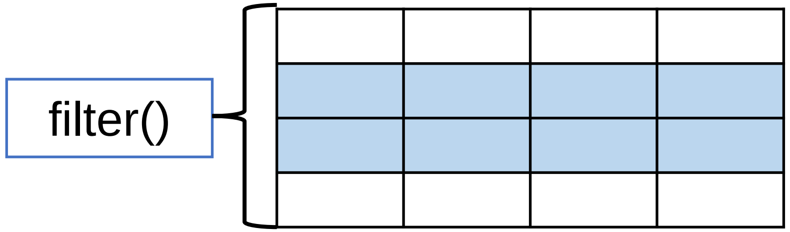
Select columns with character values



```
> ds_sal %>% select_if(is.character) %>% head(2)
    ID experience_level employment_type                  job_title
1 ID_0               MI              FT             Data Scientist
2 ID_1               SE              FT Machine Learning Scientist
  salary_currency employee_residence company_location company_size
1             EUR                 DE               DE            L
2             USD                 JP               JP            S
```

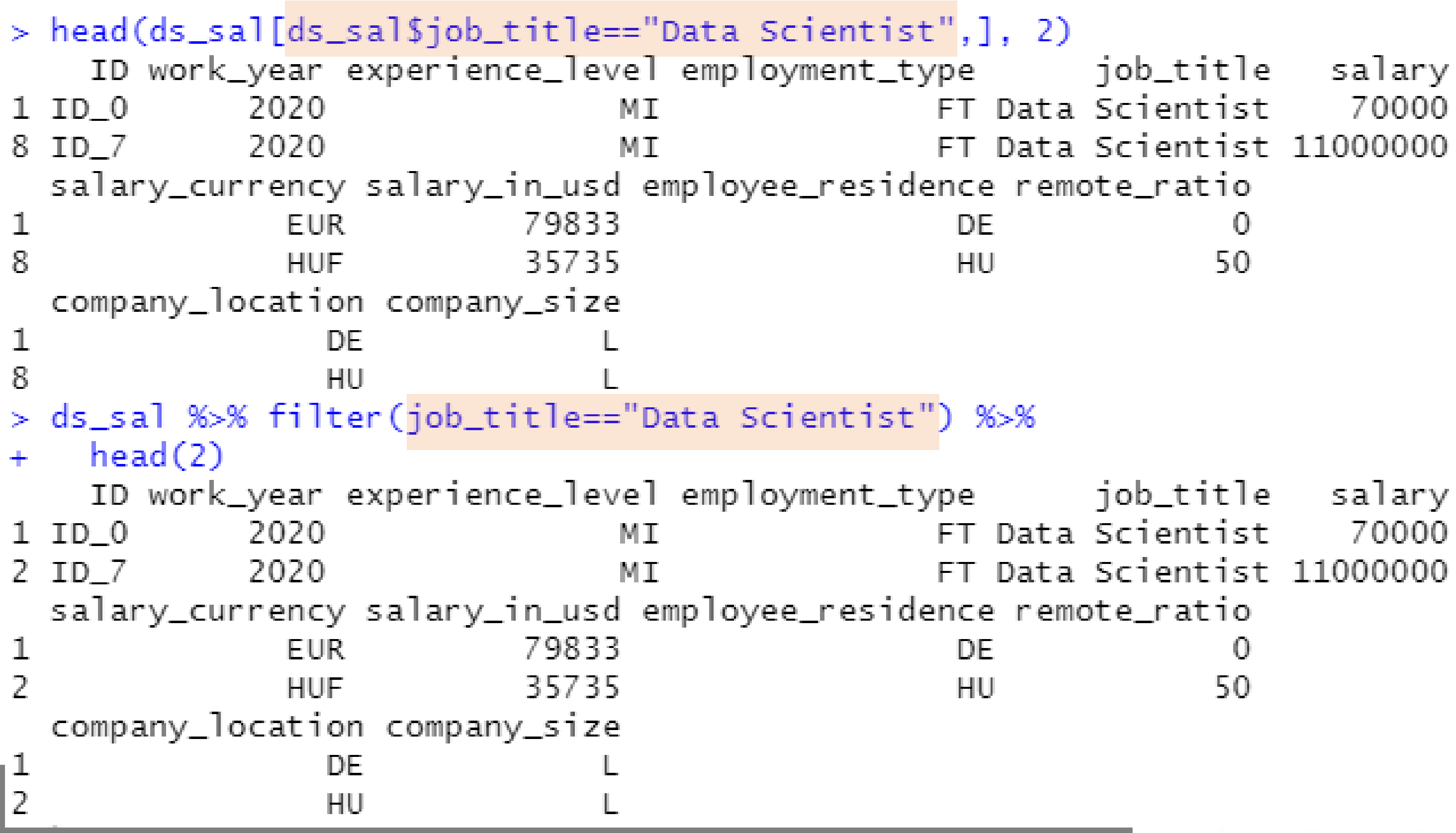
dplyr::filter()

- filter()
  - Subset a data frame, retaining all rows that satisfy your conditions. To be retained, the row must produce a value of ‘TRUE’ for all conditions.
  - Alternatives: (rows) indexing + slicing, etc.



Select rows that job title is ‘Data Scientist’

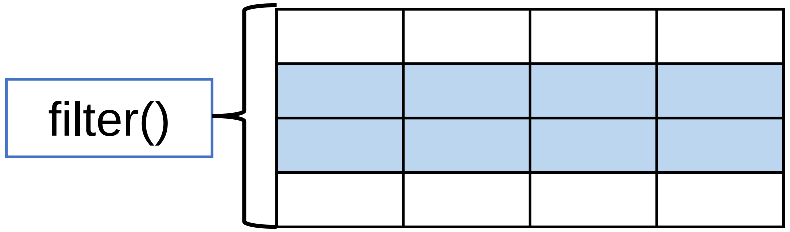
Select rows that job title is ‘Data Scientist’



```
> head(ds_sal[ds_sal$job_title=="Data Scientist",], 2)
    ID work_year experience_level employment_type      job_title   salary
1 ID_0      2020               MI              FT Data Scientist    70000
8 ID_7      2020               MI              FT Data Scientist 11000000
  salary_currency salary_in_usd employee_residence remote_ratio
1             EUR         79833                 DE            0
8             HUF         35735                 HU           50
  company_location company_size
1               DE            L
8               HU            L
> ds_sal %>% filter(job_title=="Data Scientist") %>%
+   head(2)
    ID work_year experience_level employment_type      job_title   salary
1 ID_0      2020               MI              FT Data Scientist    70000
2 ID_7      2020               MI              FT Data Scientist 11000000
  salary_currency salary_in_usd employee_residence remote_ratio
1             EUR         79833                 DE            0
2             HUF         35735                 HU           50
  company_location company_size
1               DE            L
2               HU            L
```

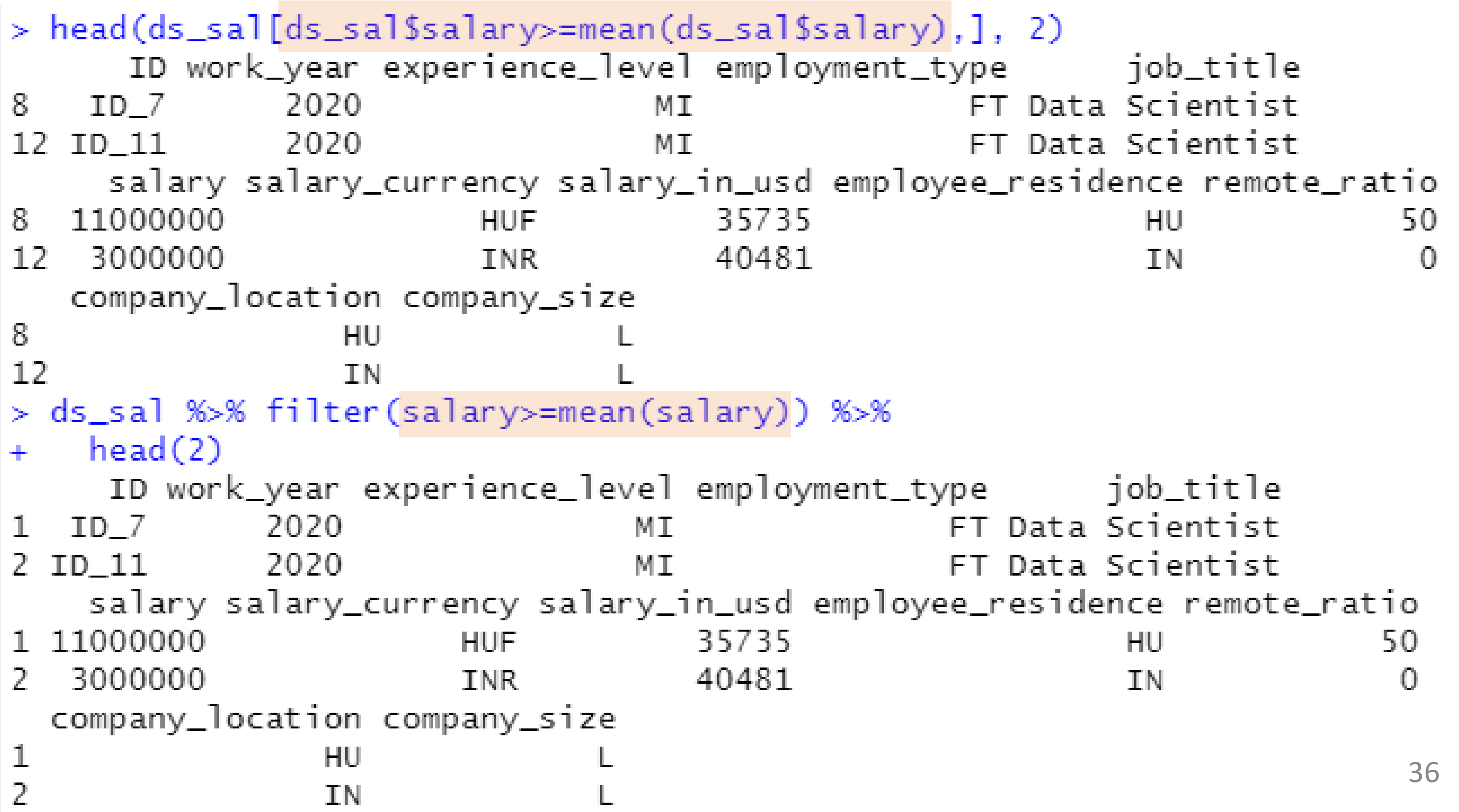
dplyr::filter()

- filter()
  - Subset a data frame, retaining all rows that satisfy your conditions. To be retained, the row must produce a value of ‘TRUE’ for all conditions.
  - Alternatives: (rows) indexing + slicing, etc.



Select rows that salary is above the average

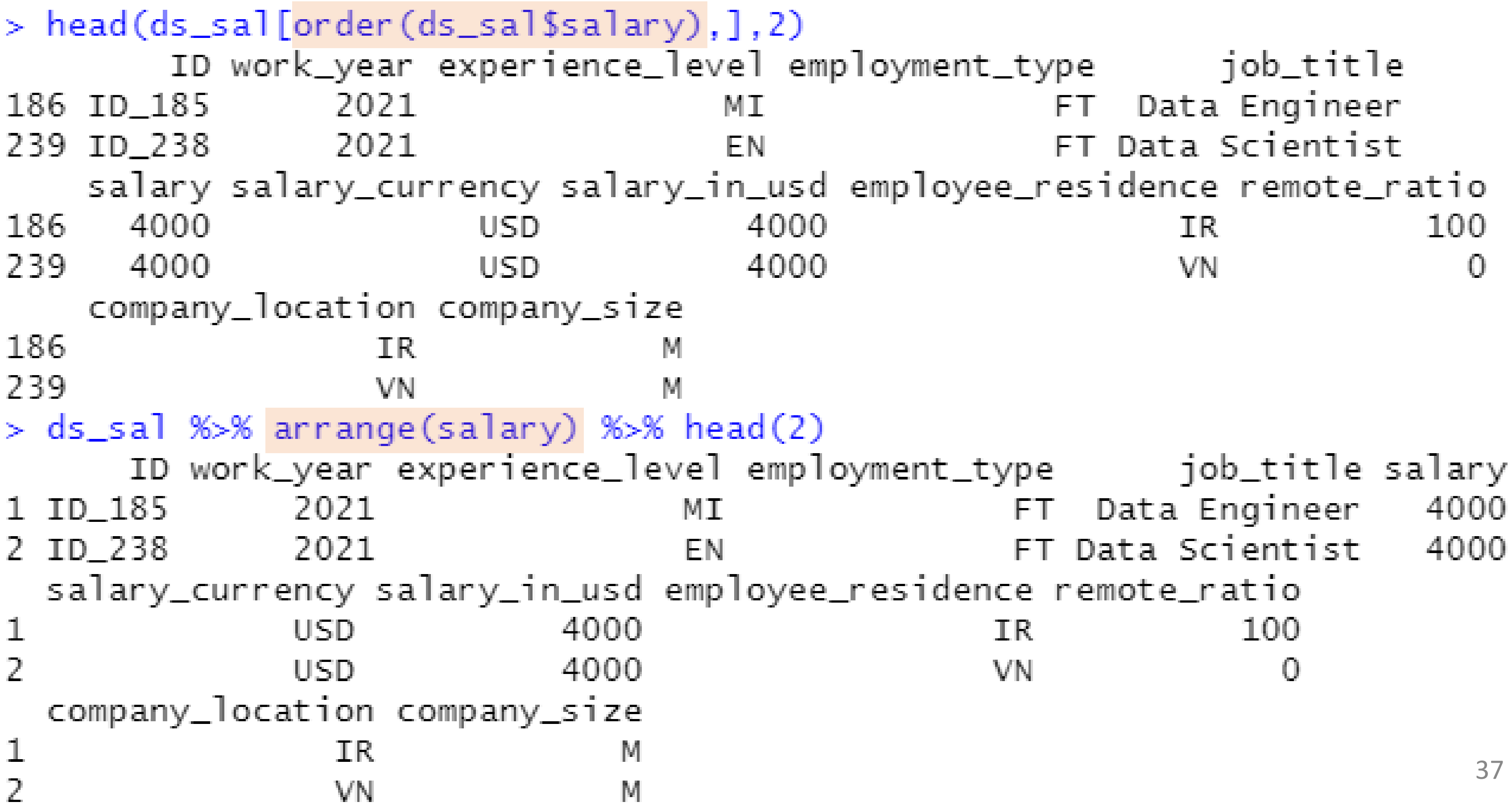
Select rows that salary is above the average



```
> head(ds_sal[ds_sal$salary>=mean(ds_sal$salary),], 2)
      ID work_year experience_level employment_type      job_title
8   ID_7      2020               MI              FT Data Scientist
12 ID_11      2020               MI              FT Data Scientist
     salary salary_currency salary_in_usd employee_residence remote_ratio
8  11000000             HUF         35735                 HU           50
12  3000000             INR         40481                 IN            0
   company_location company_size
8                HU            L
12               IN            L
> ds_sal %>% filter(salary>=mean(salary)) %>%
+   head(2)
     ID work_year experience_level employment_type      job_title
1  ID_7      2020               MI              FT Data Scientist
2 ID_11      2020               MI              FT Data Scientist
    salary salary_currency salary_in_usd employee_residence remote_ratio
1 11000000             HUF         35735                 HU           50
2  3000000             INR         40481                 IN            0
  company_location company_size
1               HU            L
2               IN            L
```

dplyr::arrange()

- arrange()
  - Orders the rows of a data frame by the values of selected columns
  - arrange(desc(*x*)) : arrange in a descending order with *x*
  - Alternatives: order



Arrange the data set in an ascending order with salary

```
> head(ds_sal[order(ds_sal$salary),],2)
        ID work_year experience_level employment_type      job_title
186 ID_185      2021               MI              FT  Data Engineer
239 ID_238      2021               EN              FT Data Scientist
    salary salary_currency salary_in_usd employee_residence remote_ratio
186   4000             USD          4000                 IR          100
239   4000             USD          4000                 VN            0
    company_location company_size
186               IR            M
239               VN            M
```

Arrange the data set in an ascending order with salary

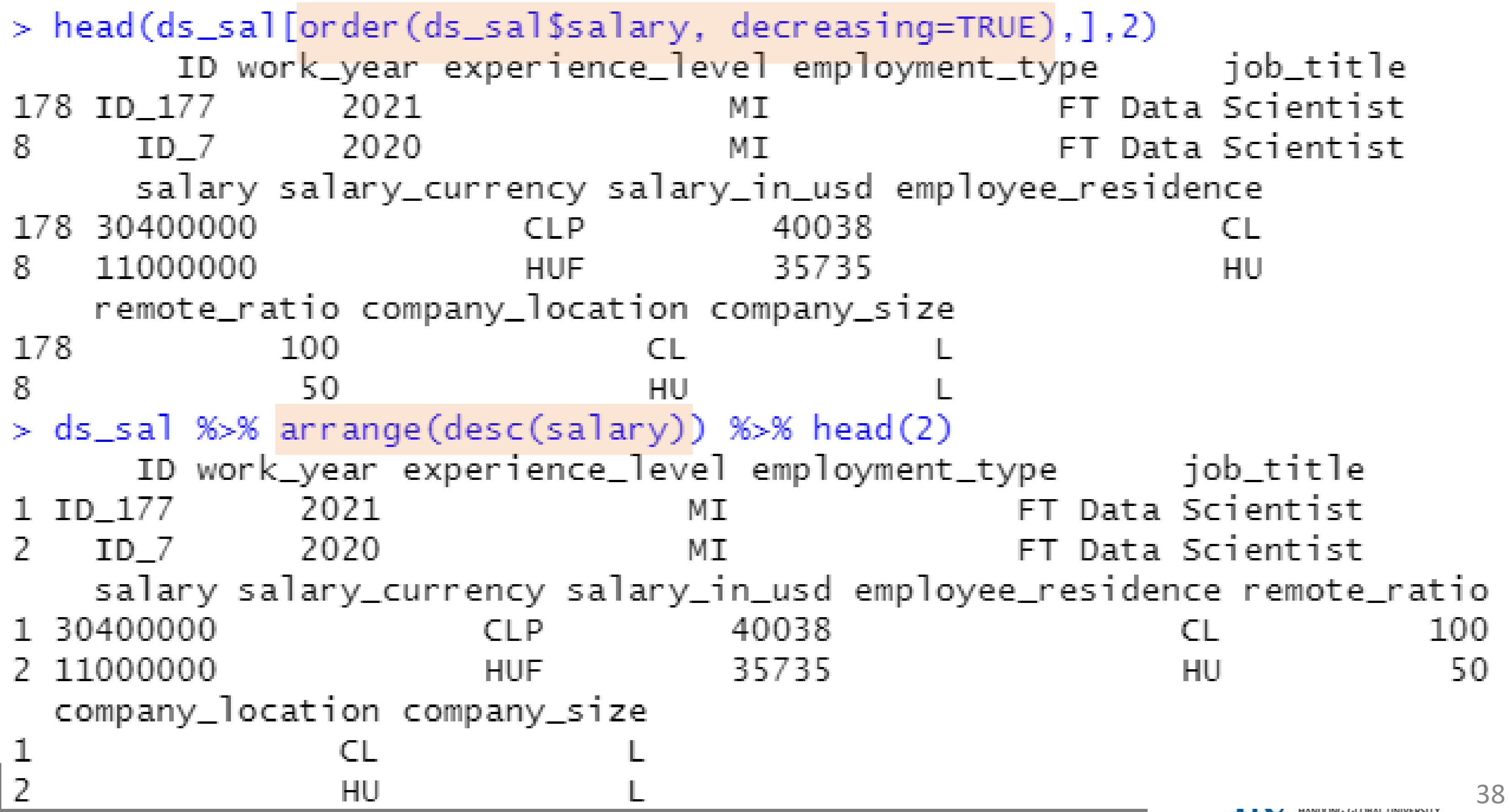
```
> ds_sal %>% arrange(salary) %>% head(2)
      ID work_year experience_level employment_type      job_title salary
1 ID_185      2021               MI              FT  Data Engineer   4000
2 ID_238      2021               EN              FT Data Scientist   4000
  salary_currency salary_in_usd employee_residence remote_ratio
1             USD          4000                 IR          100
2             USD          4000                 VN            0
  company_location company_size
1               IR            M
2               VN            M
```

dplyr::arrange()

- arrange()
  - Orders the rows of a data frame by the values of selected columns
  - arrange(desc(*x*)) : arrange in a descending order with *x*
  - Alternatives: order

Arrange the data set in a descending order with salary

Arrange the data set in a descending order with salary



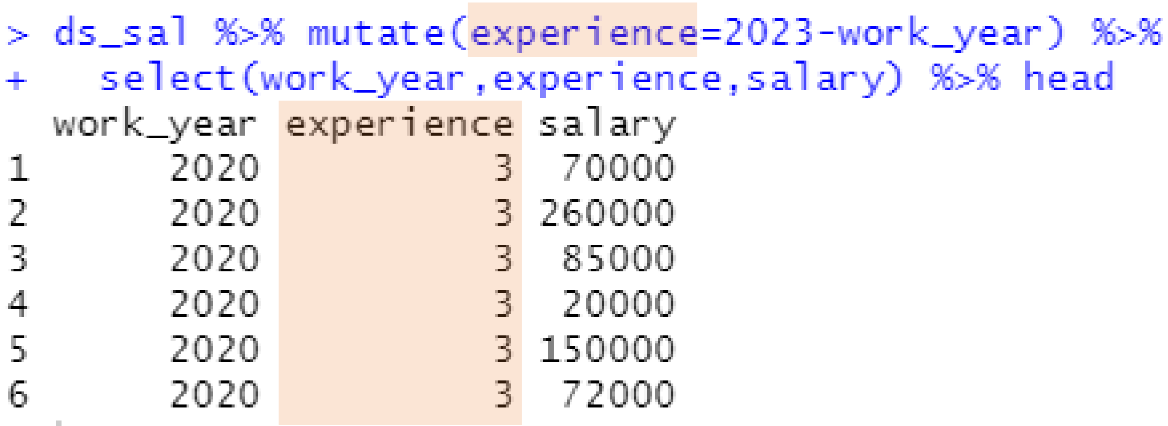
```
> head(ds_sal[order(ds_sal$salary, decreasing=TRUE),],2)
        ID work_year experience_level employment_type      job_title
178 ID_177      2021               MI              FT Data Scientist
8     ID_7      2020               MI              FT Data Scientist
      salary salary_currency salary_in_usd employee_residence
178 30400000             CLP         40038                 CL
8   11000000             HUF         35735                 HU
    remote_ratio company_location company_size
178          100               CL            L
8             50               HU            L
> ds_sal %>% arrange(desc(salary)) %>% head(2)
      ID work_year experience_level employment_type      job_title
1 ID_177      2021               MI              FT Data Scientist
2   ID_7      2020               MI              FT Data Scientist
    salary salary_currency salary_in_usd employee_residence remote_ratio
1 30400000             CLP         40038                 CL          100
2 11000000             HUF         35735                 HU           50
  company_location company_size
1               CL            L
2               HU            L
```

Data Manipulation with dplyr II

dplyr::mutate()

- mutate()
  - **<u>Adds new variables</u>** that are functions of existing variables
  - Use ifelse (if & else), case_when (if & elseif & else) with mutate()

Create experience variable referencing year 2022.

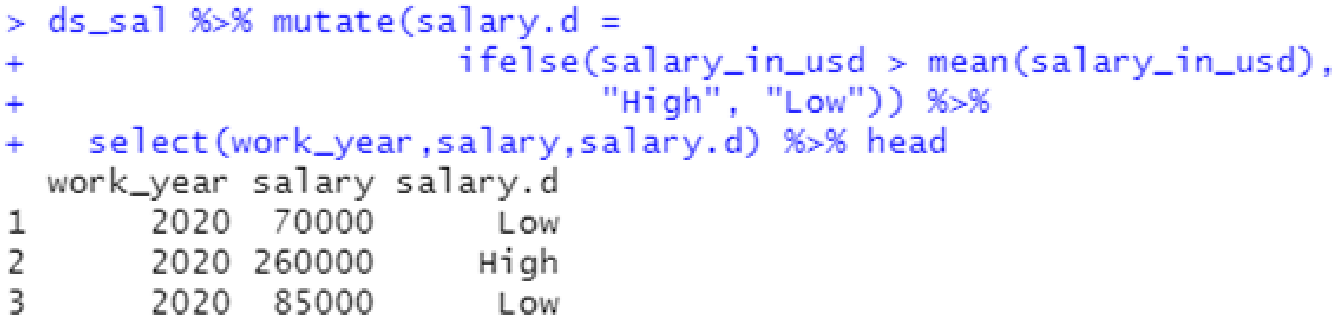


```
> ds_sal %>% mutate(experience=2023-work_year) %>%
+   select(work_year,experience,salary) %>% head
  work_year experience salary
1      2020          3  70000
2      2020          3 260000
3      2020          3  85000
4      2020          3  20000
5      2020          3 150000
6      2020          3  72000
```

ifelse(condition, TRUE-result, FALSE-result)

<Convert numeric variable to dummy variable>

Create salary.d variable, which returns High if salary_in_usd is higher than its average, and Low otherwise

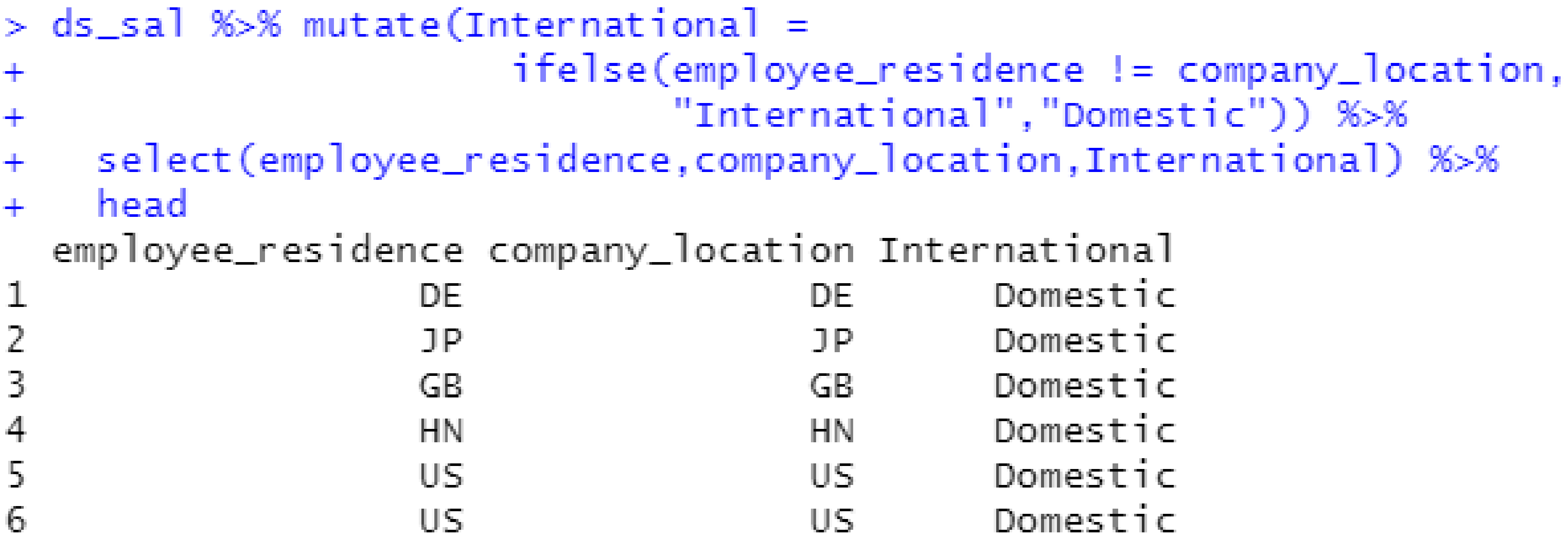


```
> ds_sal %>% mutate(salary.d =
+                     ifelse(salary_in_usd > mean(salary_in_usd),
+                            "High", "Low")) %>%
+   select(work_year,salary,salary.d) %>% head
  work_year salary salary.d
1      2020  70000      Low
2      2020 260000     High
3      2020  85000      Low
```

dplyr::mutate()

- mutate()
  - **<u>Adds new variables</u>** that are functions of existing variables
  - Use ifelse (if & else), case_when (if & elseif & else) with mutate()

Create International variable, which returns International if employee_residence and company_location are the different and Domestic otherwise

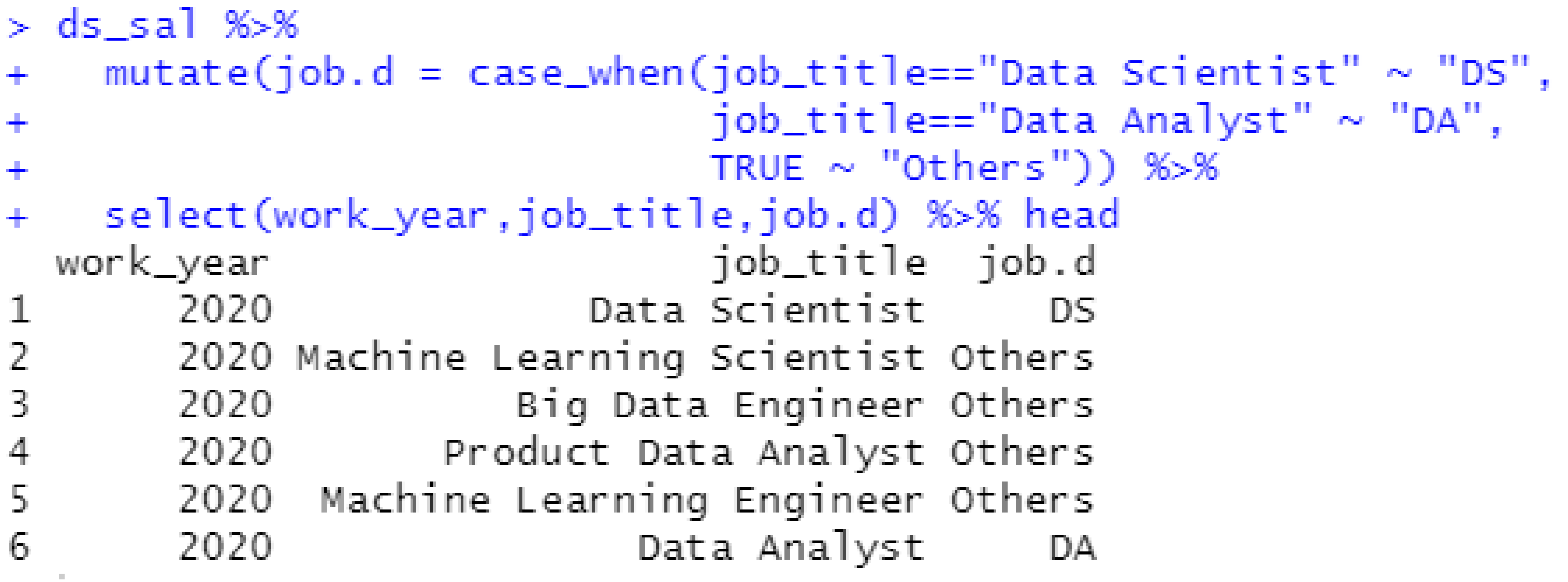


```
> ds_sal %>% mutate(International =
+                             ifelse(employee_residence != company_location,
+                                     "International","Domestic")) %>%
+   select(employee_residence,company_location,International) %>%
+   head
  employee_residence company_location International
1                 DE               DE      Domestic
2                 JP               JP      Domestic
3                 GB               GB      Domestic
4                 HN               HN      Domestic
5                 US               US      Domestic
6                 US               US      Domestic
```

dplyr::mutate()

- mutate()
  - **Adds new variables** that are functions of existing variables
  - Use ifelse (if & else), case_when (if & elseif & else) with mutate()

Create job.d variable, which returns DS for Data Scientist, DA for Data Analyst and others for Others



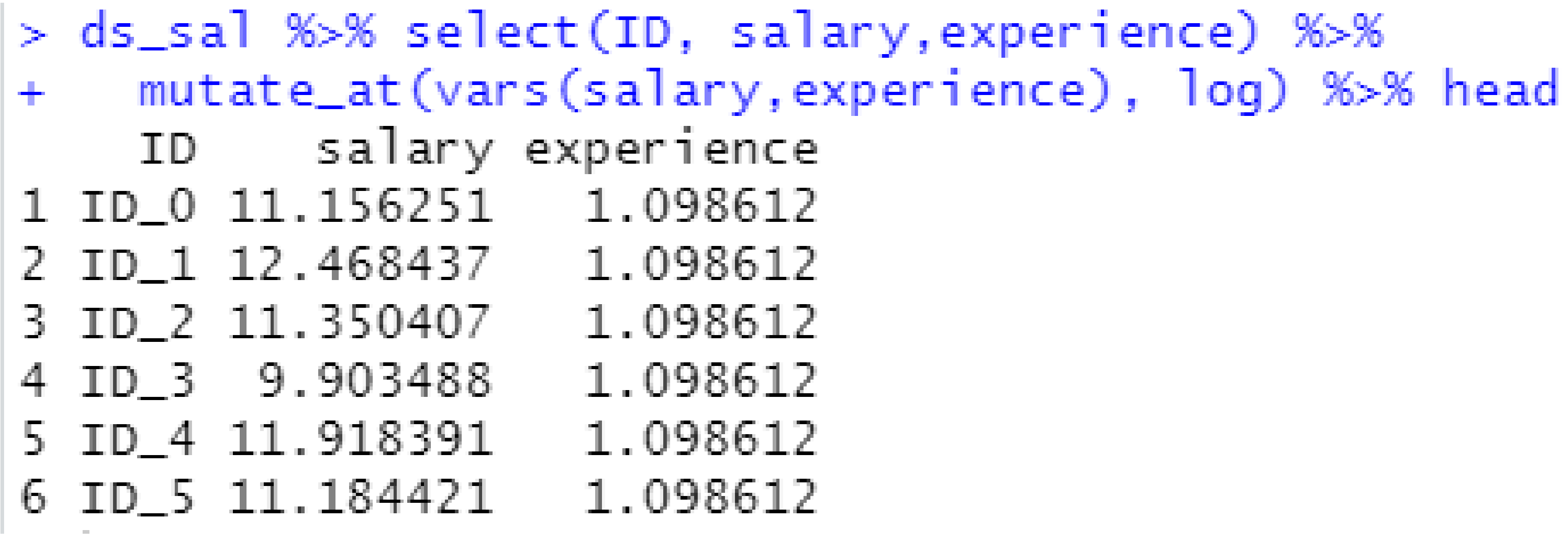
```
> ds_sal %>%
+   mutate(job.d = case_when(job_title=="Data Scientist" ~ "DS",
+                            job_title=="Data Analyst" ~ "DA",
+                            TRUE ~ "Others")) %>%
+   select(work_year,job_title,job.d) %>% head
  work_year                  job_title  job.d
1      2020             Data Scientist     DS
2      2020 Machine Learning Scientist Others
3      2020          Big Data Engineer Others
4      2020       Product Data Analyst Others
5      2020  Machine Learning Engineer Others
6      2020               Data Analyst     DA
```

dplyr::mutate()

- mutate_at()
  - **changes selected columns**

<Log Transformation>

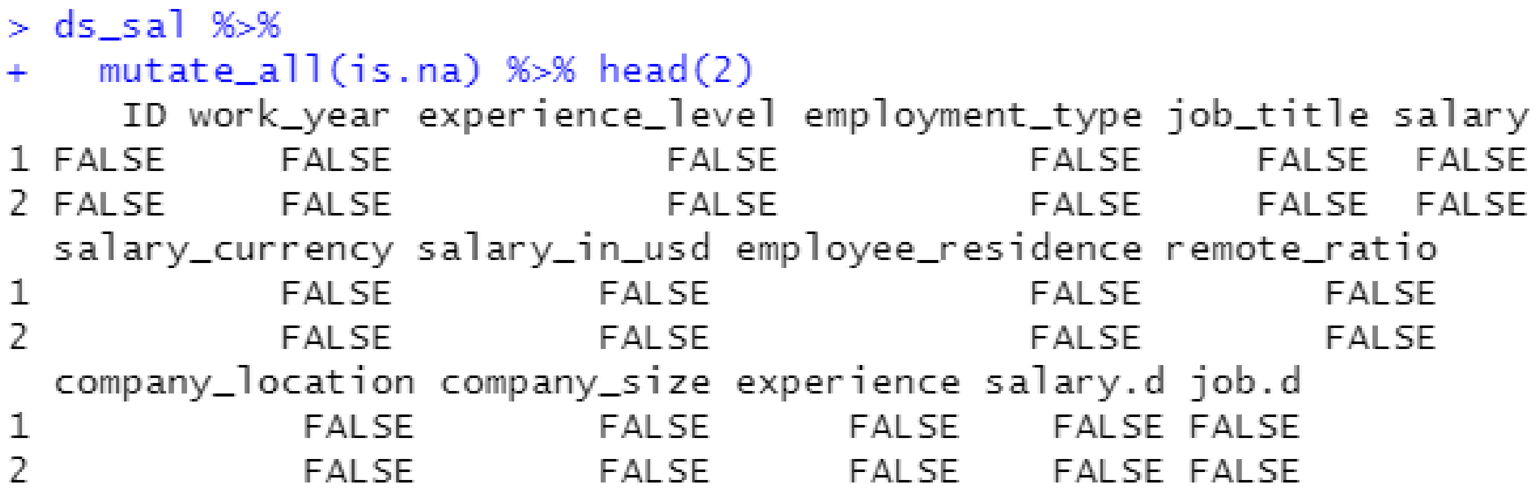
Convert all variables into a log scale



```
> ds_sal %>% select(ID, salary,experience) %>%
+   mutate_at(vars(salary,experience), log) %>% head
    ID    salary experience
1 ID_0 11.156251   1.098612
2 ID_1 12.468437   1.098612
3 ID_2 11.350407   1.098612
4 ID_3  9.903488   1.098612
5 ID_4 11.918391   1.098612
6 ID_5 11.184421   1.098612
```

dplyr::mutate()

- mutate_all()
  - **changes all columns**



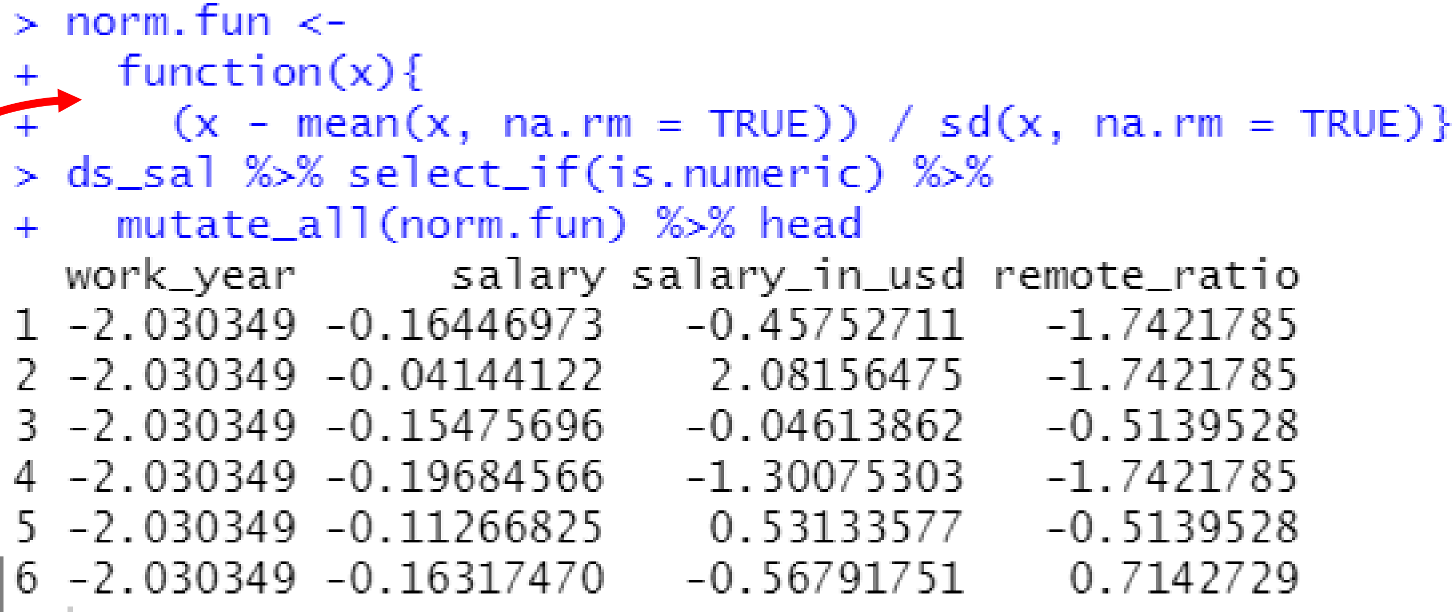
```
> ds_sal %>%
+   mutate_all(is.na) %>% head(2)
     ID work_year experience_level employment_type job_title salary
1 FALSE     FALSE            FALSE           FALSE     FALSE  FALSE
2 FALSE     FALSE            FALSE           FALSE     FALSE  FALSE
  salary_currency salary_in_usd employee_residence remote_ratio
1           FALSE         FALSE              FALSE        FALSE
2           FALSE         FALSE              FALSE        FALSE
  company_location company_size experience salary.d job.d
1            FALSE        FALSE      FALSE    FALSE FALSE
2            FALSE        FALSE      FALSE    FALSE FALSE
```

<Normalization>

Normalize numeric variables

Z-score Normalization

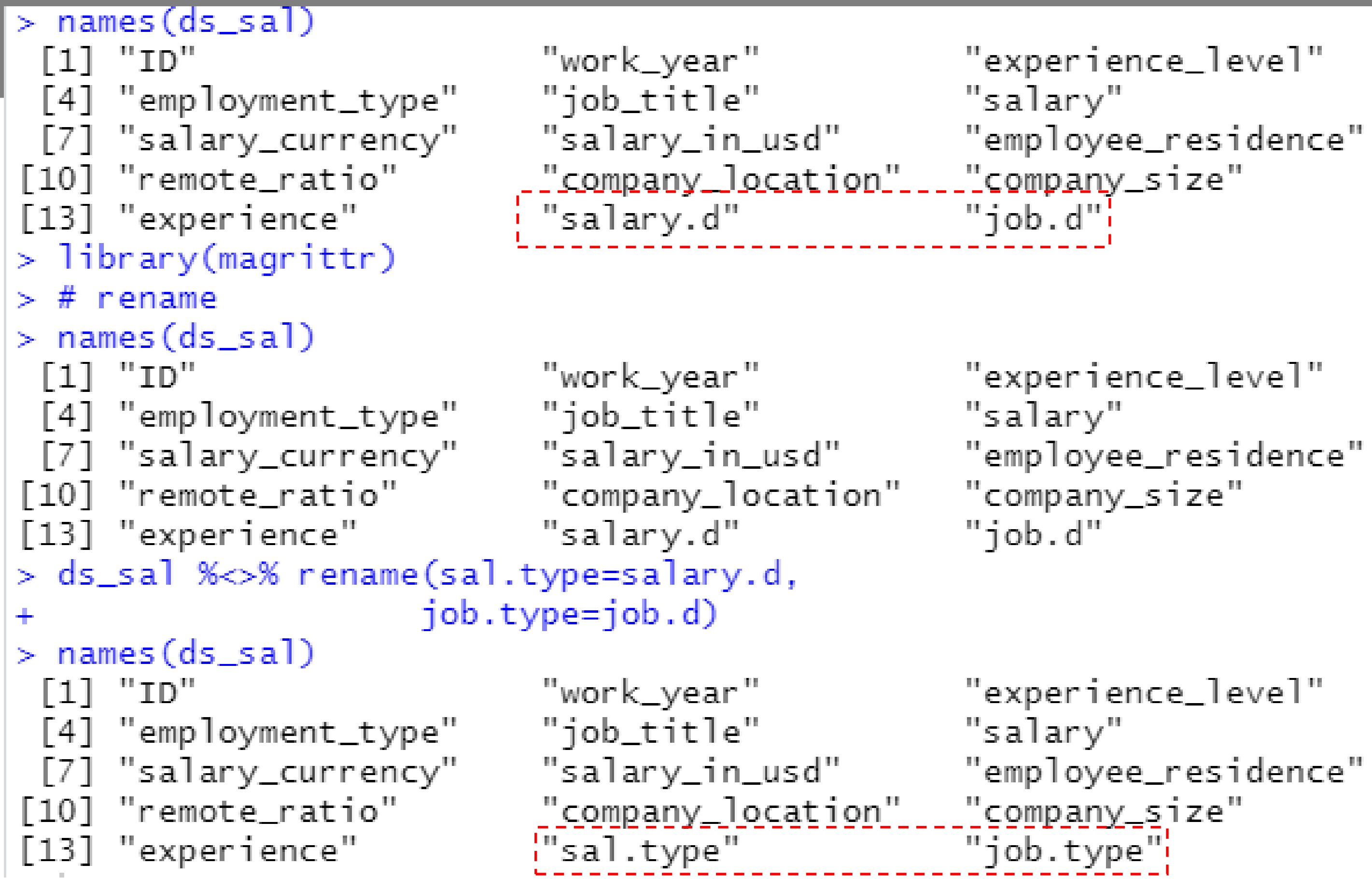
$$\frac{x - \text{mean(x)}}{std.dev(x)}$$



```
> norm.fun <-
+   function(x){
+     (x - mean(x, na.rm = TRUE)) / sd(x, na.rm = TRUE)}
> ds_sal %>% select_if(is.numeric) %>%
+   mutate_all(norm.fun) %>% head
  work_year      salary salary_in_usd remote_ratio
1 -2.030349 -0.16446973   -0.45752711   -1.7421785
2 -2.030349 -0.04144122    2.08156475   -1.7421785
3 -2.030349 -0.15475696   -0.04613862   -0.5139528
4 -2.030349 -0.19684566   -1.30075303   -1.7421785
5 -2.030349 -0.11266825    0.53133577   -0.5139528
6 -2.030349 -0.16317470   -0.56791751    0.7142729
```

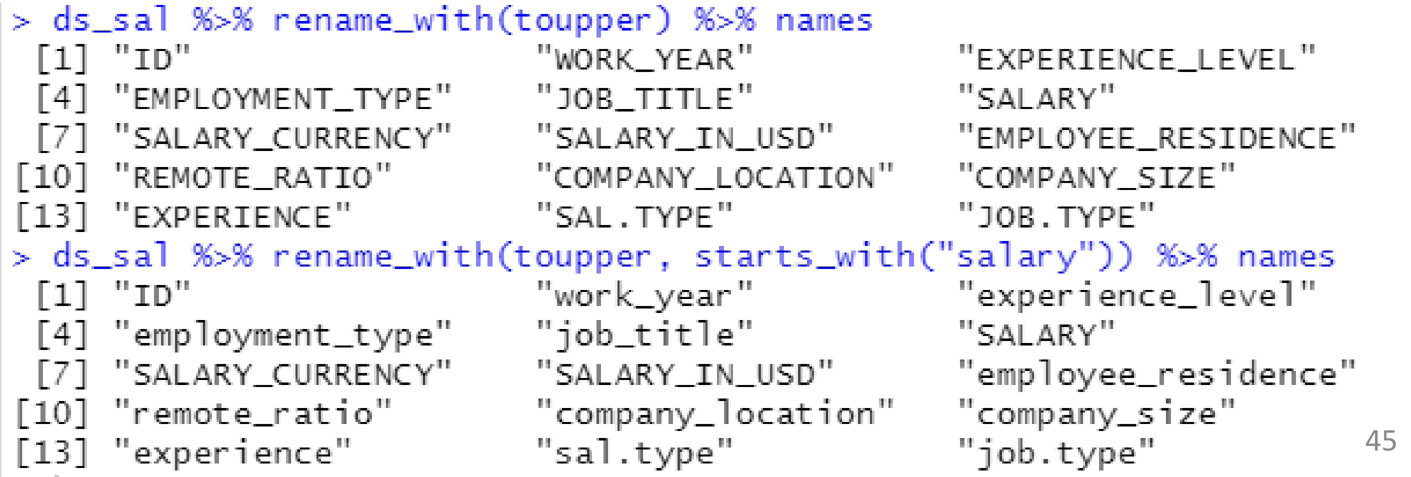
dplyr::rename()

- rename()
  - rename() changes the names of individual variables using *new_name = old_name*



```
> names(ds_sal)
 [1] "ID"                 "work_year"          "experience_level"
 [4] "employment_type"    "job_title"          "salary"
 [7] "salary_currency"    "salary_in_usd"      "employee_residence"
[10] "remote_ratio"       "company_location"   "company_size"
[13] "experience"         "salary.d"           "job.d"
> library(magrittr)
> # rename
> names(ds_sal)
 [1] "ID"                 "work_year"          "experience_level"
 [4] "employment_type"    "job_title"          "salary"
 [7] "salary_currency"    "salary_in_usd"      "employee_residence"
[10] "remote_ratio"       "company_location"   "company_size"
[13] "experience"         "salary.d"           "job.d"
> ds_sal %<>% rename(sal.type=salary.d,
+                    job.type=job.d)
> names(ds_sal)
 [1] "ID"                 "work_year"          "experience_level"
 [4] "employment_type"    "job_title"          "salary"
 [7] "salary_currency"    "salary_in_usd"      "employee_residence"
[10] "remote_ratio"       "company_location"   "company_size"
[13] "experience"         "sal.type"           "job.type"
```

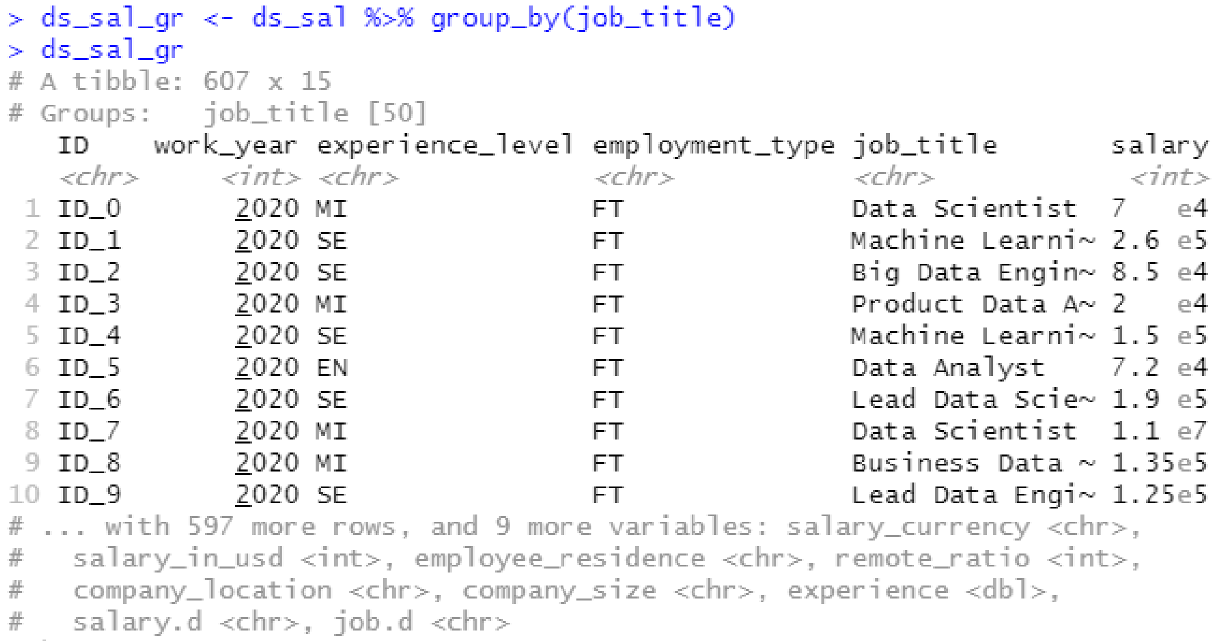
- rename_with()
  - rename columns with functions



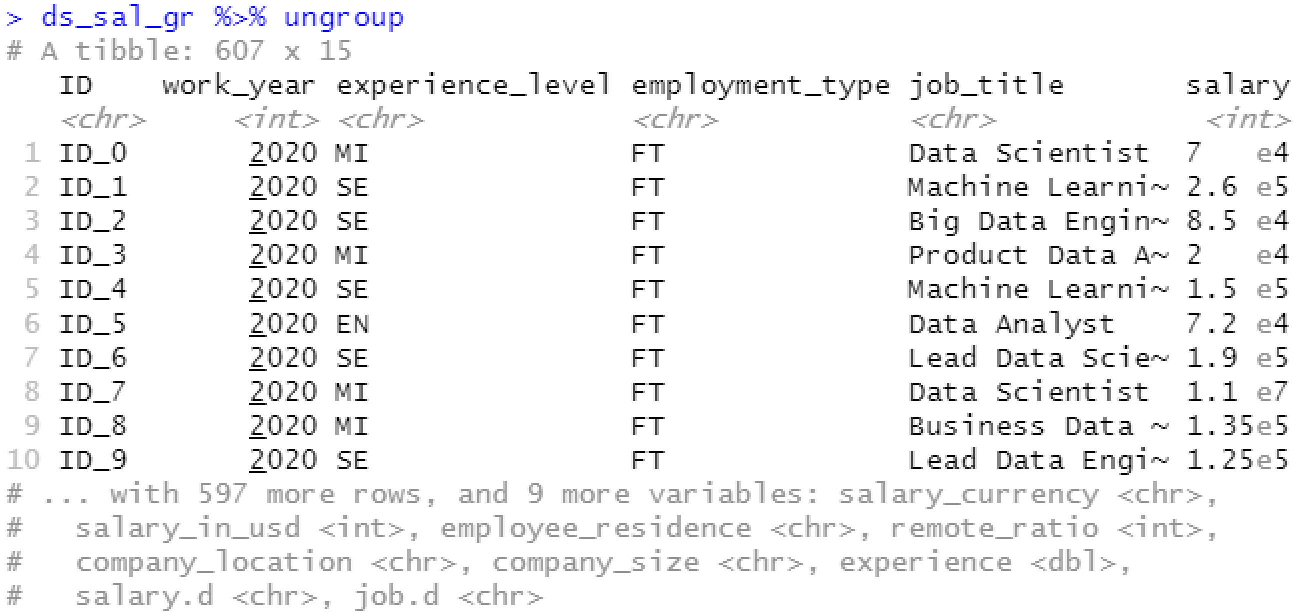
```
> ds_sal %>% rename_with(toupper) %>% names
 [1] "ID"                 "WORK_YEAR"          "EXPERIENCE_LEVEL"
 [4] "EMPLOYMENT_TYPE"    "JOB_TITLE"          "SALARY"
 [7] "SALARY_CURRENCY"    "SALARY_IN_USD"      "EMPLOYEE_RESIDENCE"
[10] "REMOTE_RATIO"       "COMPANY_LOCATION"   "COMPANY_SIZE"
[13] "EXPERIENCE"         "SAL.TYPE"           "JOB.TYPE"
> ds_sal %>% rename_with(toupper, starts_with("salary")) %>% names
 [1] "ID"                 "work_year"          "experience_level"
 [4] "employment_type"    "job_title"          "SALARY"
 [7] "SALARY_CURRENCY"    "SALARY_IN_USD"      "employee_residence"
[10] "remote_ratio"       "company_location"   "company_size"
[13] "experience"         "sal.type"           "job.type"
```

dplyr::group_by()

- group_by()
  - group_by() takes an existing tbl and converts it into a grouped tbl where operations are performed "by group".
  - ungroup() removes grouping



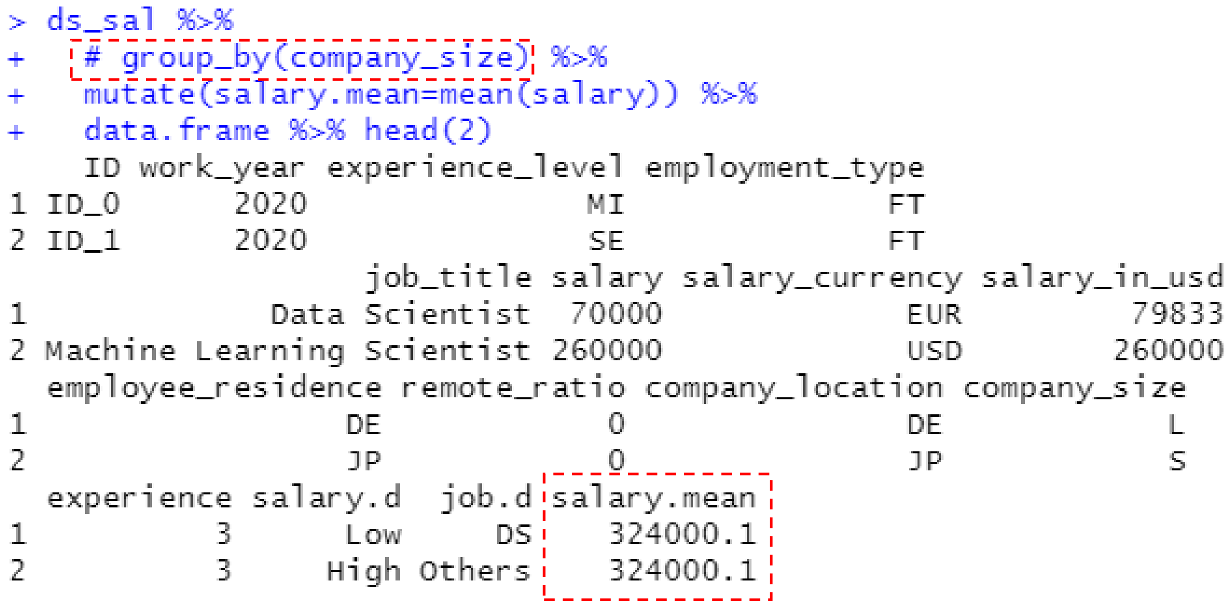
```
> ds_sal_gr <- ds_sal %>% group_by(job_title)
> ds_sal_gr
# A tibble: 607 x 15
# Groups:   job_title [50]
   ID    work_year experience_level employment_type job_title       salary
   <chr>     <int> <chr>            <chr>           <chr>            <int>
 1 ID_0       2020 MI               FT              Data Scientist  7    e4
 2 ID_1       2020 SE               FT              Machine Learni~ 2.6  e5
 3 ID_2       2020 SE               FT              Big Data Engin~ 8.5  e4
 4 ID_3       2020 MI               FT              Product Data A~ 2    e4
 5 ID_4       2020 SE               FT              Machine Learni~ 1.5  e5
 6 ID_5       2020 EN               FT              Data Analyst    7.2  e4
 7 ID_6       2020 SE               FT              Lead Data Scie~ 1.9  e5
 8 ID_7       2020 MI               FT              Data Scientist  1.1  e7
 9 ID_8       2020 MI               FT              Business Data ~ 1.35e5
10 ID_9       2020 SE               FT              Lead Data Engi~ 1.25e5
# ... with 597 more rows, and 9 more variables: salary_currency <chr>,
#   salary_in_usd <int>, employee_residence <chr>, remote_ratio <int>,
#   company_location <chr>, company_size <chr>, experience <dbl>,
#   salary.d <chr>, job.d <chr>
```



```
> ds_sal_gr %>% ungroup
# A tibble: 607 x 15
   ID    work_year experience_level employment_type job_title       salary
   <chr>     <int> <chr>            <chr>           <chr>            <int>
 1 ID_0       2020 MI               FT              Data Scientist  7    e4
 2 ID_1       2020 SE               FT              Machine Learni~ 2.6  e5
 3 ID_2       2020 SE               FT              Big Data Engin~ 8.5  e4
 4 ID_3       2020 MI               FT              Product Data A~ 2    e4
 5 ID_4       2020 SE               FT              Machine Learni~ 1.5  e5
 6 ID_5       2020 EN               FT              Data Analyst    7.2  e4
 7 ID_6       2020 SE               FT              Lead Data Scie~ 1.9  e5
 8 ID_7       2020 MI               FT              Data Scientist  1.1  e7
 9 ID_8       2020 MI               FT              Business Data ~ 1.35e5
10 ID_9       2020 SE               FT              Lead Data Engi~ 1.25e5
# ... with 597 more rows, and 9 more variables: salary_currency <chr>,
#   salary_in_usd <int>, employee_residence <chr>, remote_ratio <int>,
#   company_location <chr>, company_size <chr>, experience <dbl>,
#   salary.d <chr>, job.d <chr>
```

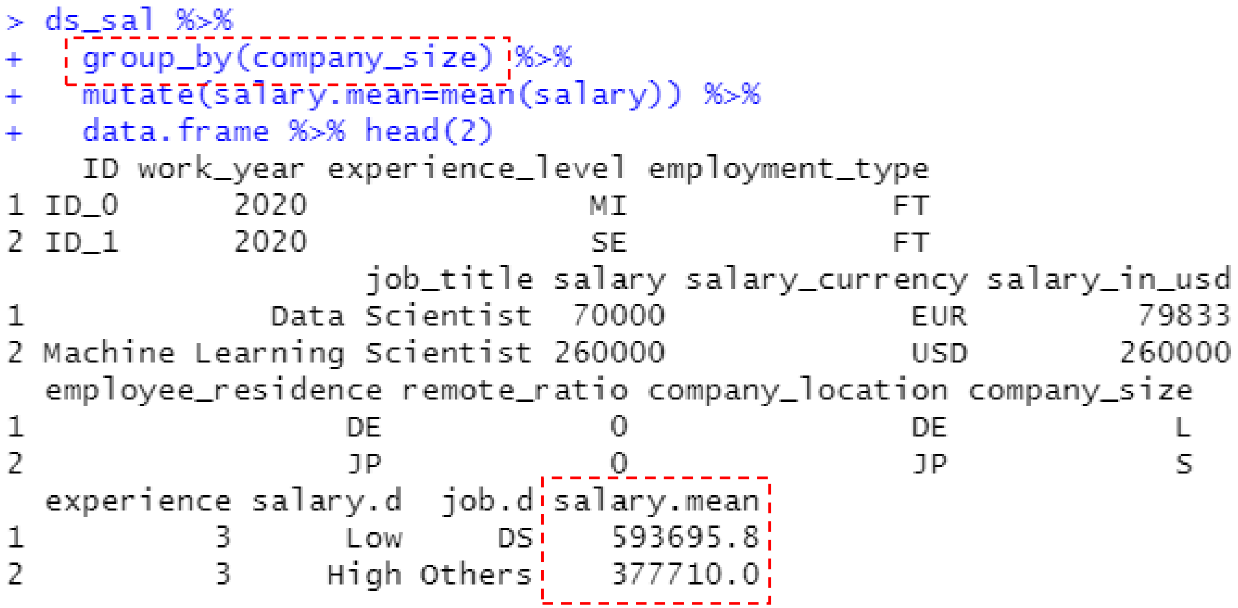
dplyr::group_by() + mutate()

- group_by() + mutate()
  - summarise the data based on the “group_by” target



```
> ds_sal %>%
+   # group_by(company_size) %>%
+   mutate(salary.mean=mean(salary)) %>%
+   data.frame %>% head(2)
    ID work_year experience_level employment_type
1 ID_0      2020               MI              FT
2 ID_1      2020               SE              FT
                   job_title salary salary_currency salary_in_usd
1             Data Scientist  70000             EUR         79833
2 Machine Learning Scientist 260000             USD        260000
  employee_residence remote_ratio company_location company_size
1                 DE            0               DE            L
2                 JP            0               JP            S
  experience salary.d  job.d salary.mean
1          3      Low     DS    324000.1
2          3     High Others    324000.1
```

average of all samples

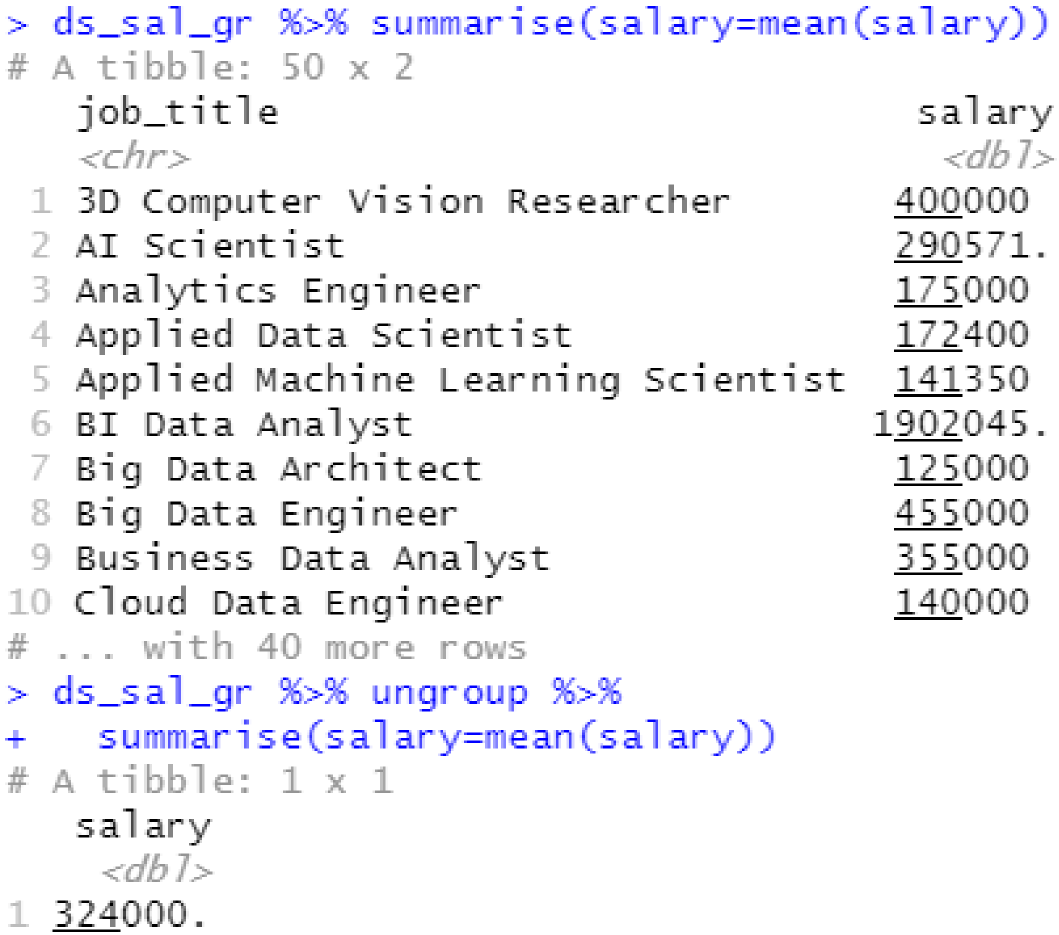


```
> ds_sal %>%
+   group_by(company_size) %>%
+   mutate(salary.mean=mean(salary)) %>%
+   data.frame %>% head(2)
    ID work_year experience_level employment_type
1 ID_0      2020               MI              FT
2 ID_1      2020               SE              FT
                   job_title salary salary_currency salary_in_usd
1             Data Scientist  70000             EUR         79833
2 Machine Learning Scientist 260000             USD        260000
  employee_residence remote_ratio company_location company_size
1                 DE            0               DE            L
2                 JP            0               JP            S
  experience salary.d  job.d salary.mean
1          3      Low     DS    593695.8
2          3     High Others    377710.0
```

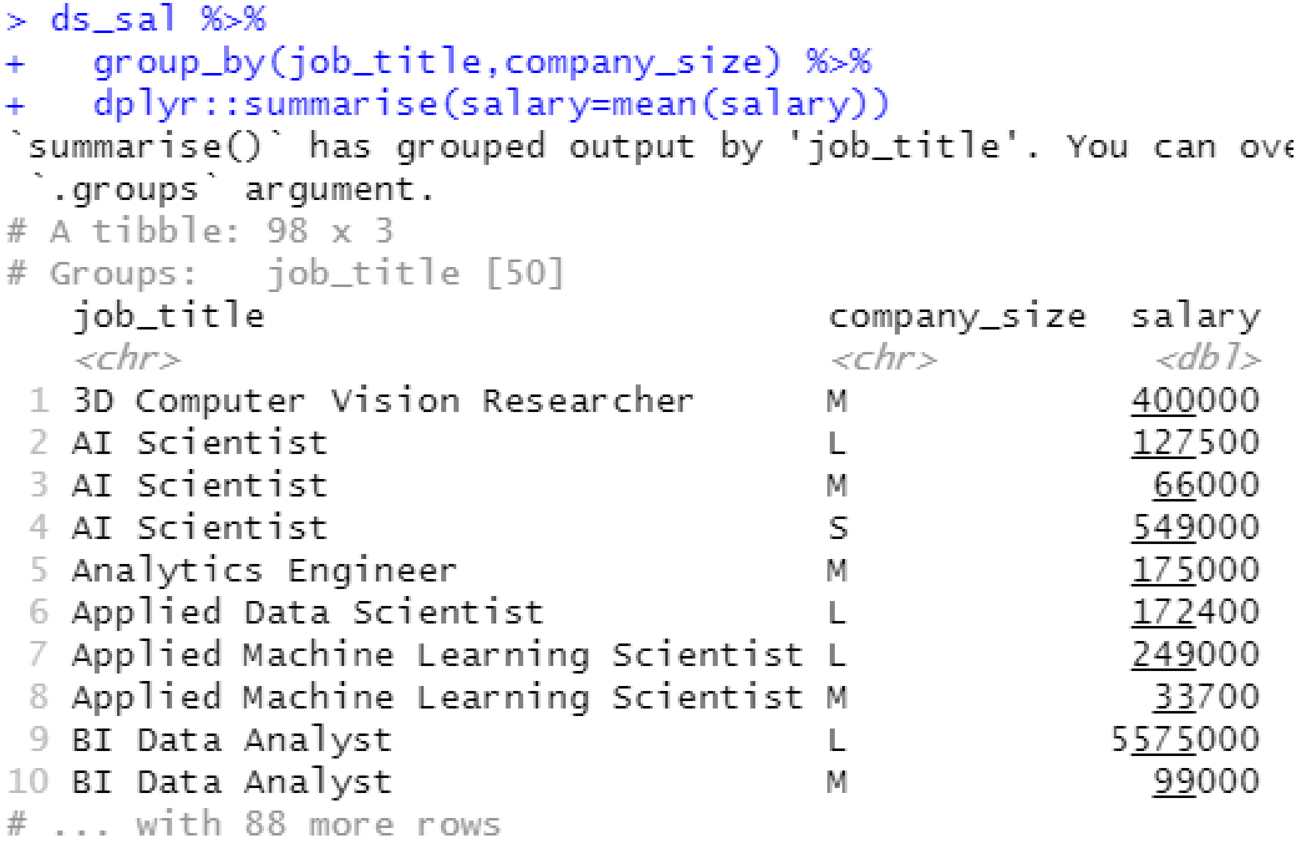
average of each group

dplyr::group_by() + summarise()

- group_by() + summarise()
  - summarise the data based on the “group_by” target



```
> ds_sal_gr %>% summarise(salary=mean(salary))
# A tibble: 50 x 2
   job_title                                salary
   <chr>                                     <dbl>
 1 3D Computer Vision Researcher            400000
 2 AI Scientist                             290571.
 3 Analytics Engineer                       175000
 4 Applied Data Scientist                   172400
 5 Applied Machine Learning Scientist       141350
 6 BI Data Analyst                         1902045.
 7 Big Data Architect                       125000
 8 Big Data Engineer                        455000
 9 Business Data Analyst                    355000
10 Cloud Data Engineer                      140000
# ... with 40 more rows
> ds_sal_gr %>% ungroup %>%
+   summarise(salary=mean(salary))
# A tibble: 1 x 1
   salary
    <dbl>
1 324000.
```

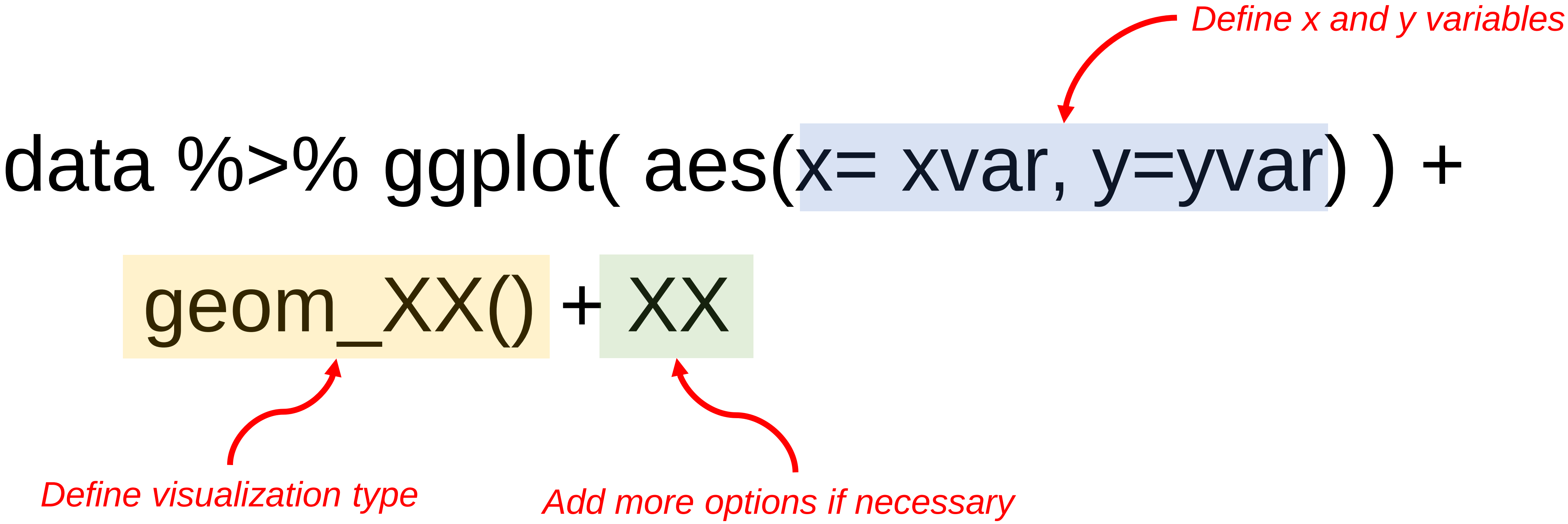


```
> ds_sal %>%
+   group_by(job_title,company_size) %>%
+   dplyr::summarise(salary=mean(salary))
`summarise()` has grouped output by 'job_title'. You can ove
  `.groups` argument.
# A tibble: 98 x 3
# Groups:   job_title [50]
   job_title                          company_size  salary
   <chr>                              <chr>          <dbl>
 1 3D Computer Vision Researcher      M             400000
 2 AI Scientist                       L             127500
 3 AI Scientist                       M              66000
 4 AI Scientist                       S             549000
 5 Analytics Engineer                 M             175000
 6 Applied Data Scientist             L             172400
 7 Applied Machine Learning Scientist L             249000
 8 Applied Machine Learning Scientist M              33700
 9 BI Data Analyst                    L            5575000
10 BI Data Analyst                    M              99000
# ... with 88 more rows
```

Data Visualization with ggplot

ggplot2

- ggplot2() package
  - A well-known package for creating data visualizations
  - Recommended to take “Data Visualization” to learn more details



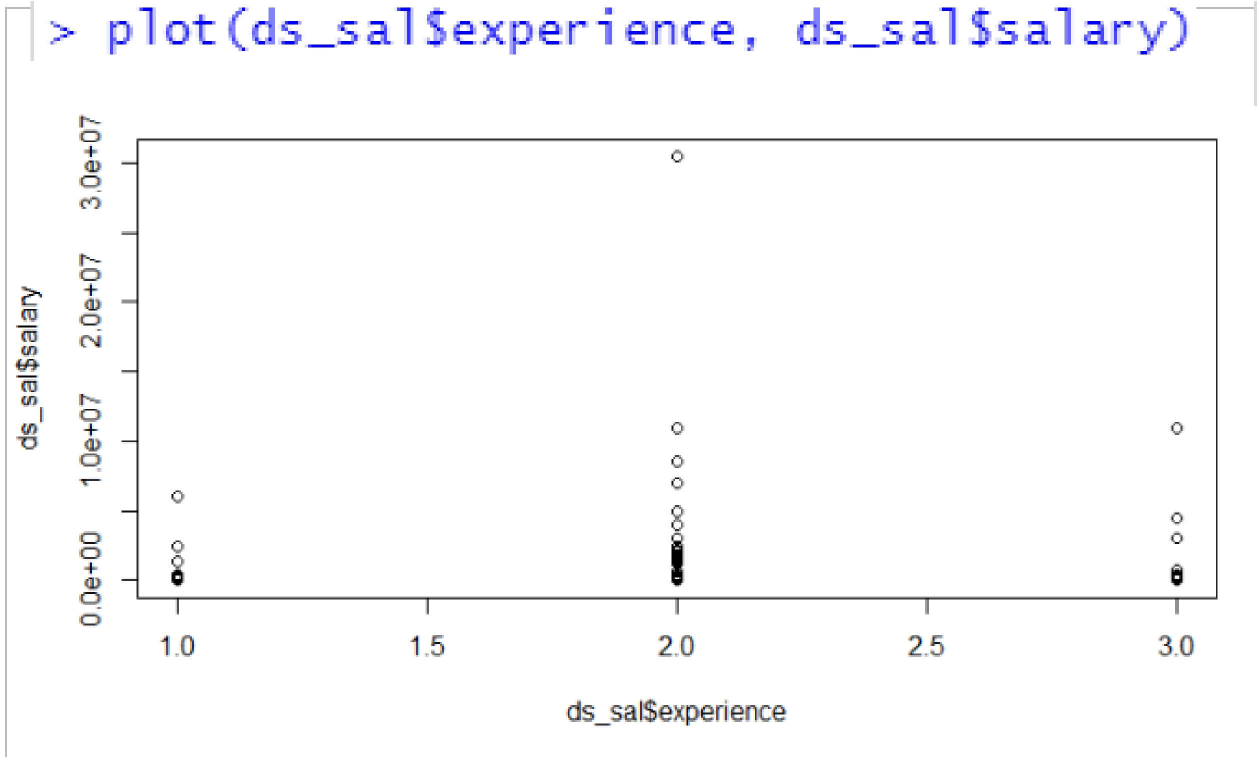
Define x and y variables

```
data %>% ggplot( aes(x= xvar, y=yvar) ) +
    geom_XX() + XX
```

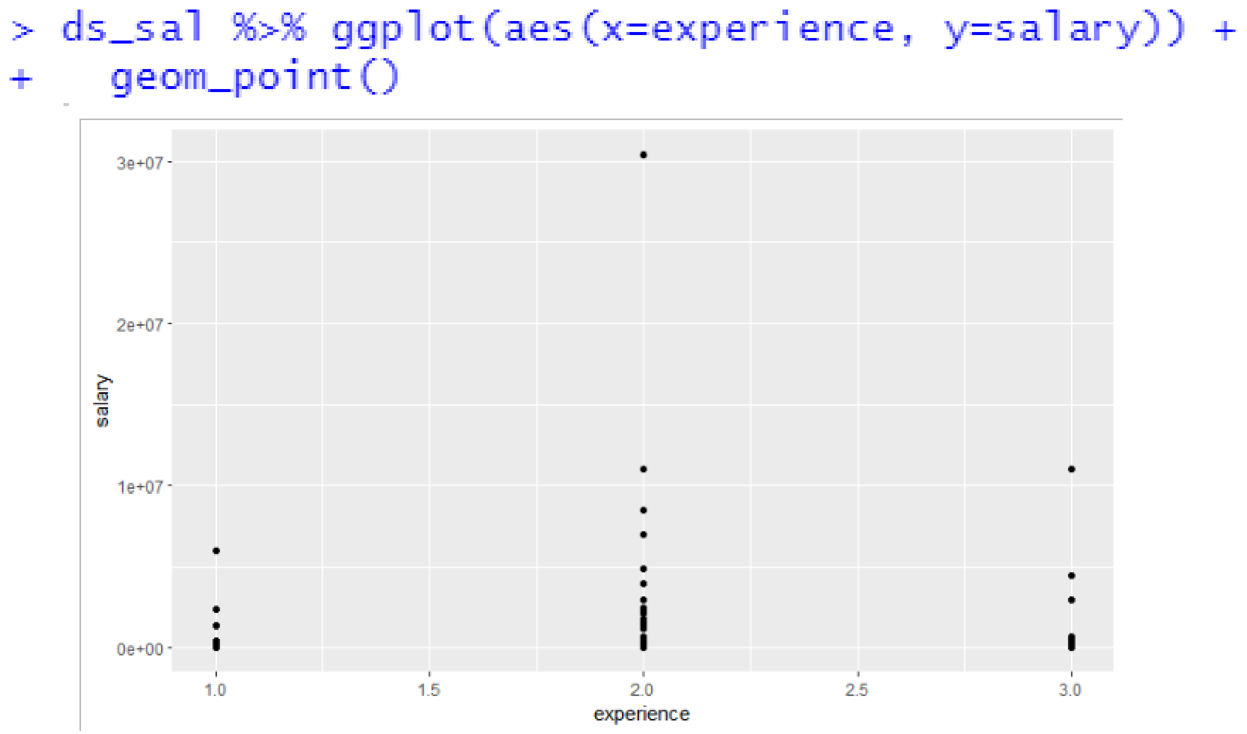
Define visualization type

Add more options if necessary

- Scatter plot
    - geom_point()



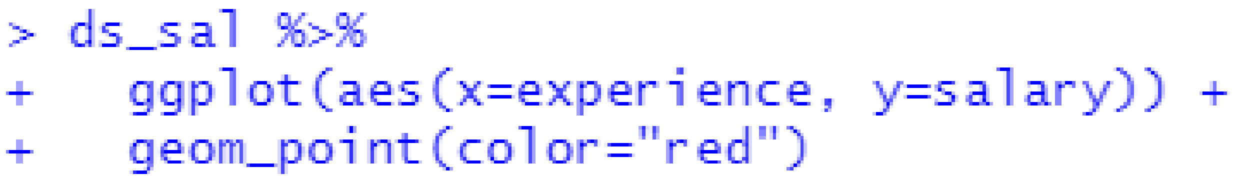
```
> plot(ds_sal$experience, ds_sal$salary)
```



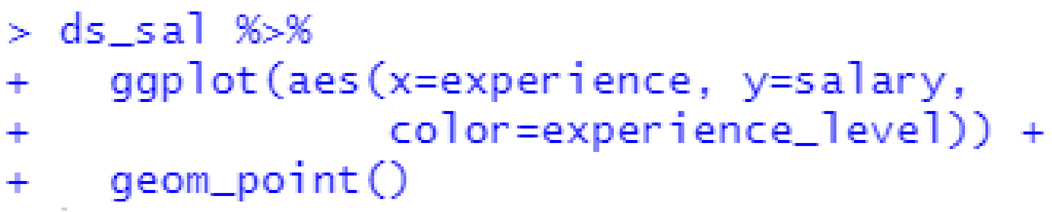
```
> ds_sal %>% ggplot(aes(x=experience, y=salary)) +
+   geom_point()
```

ggplot2

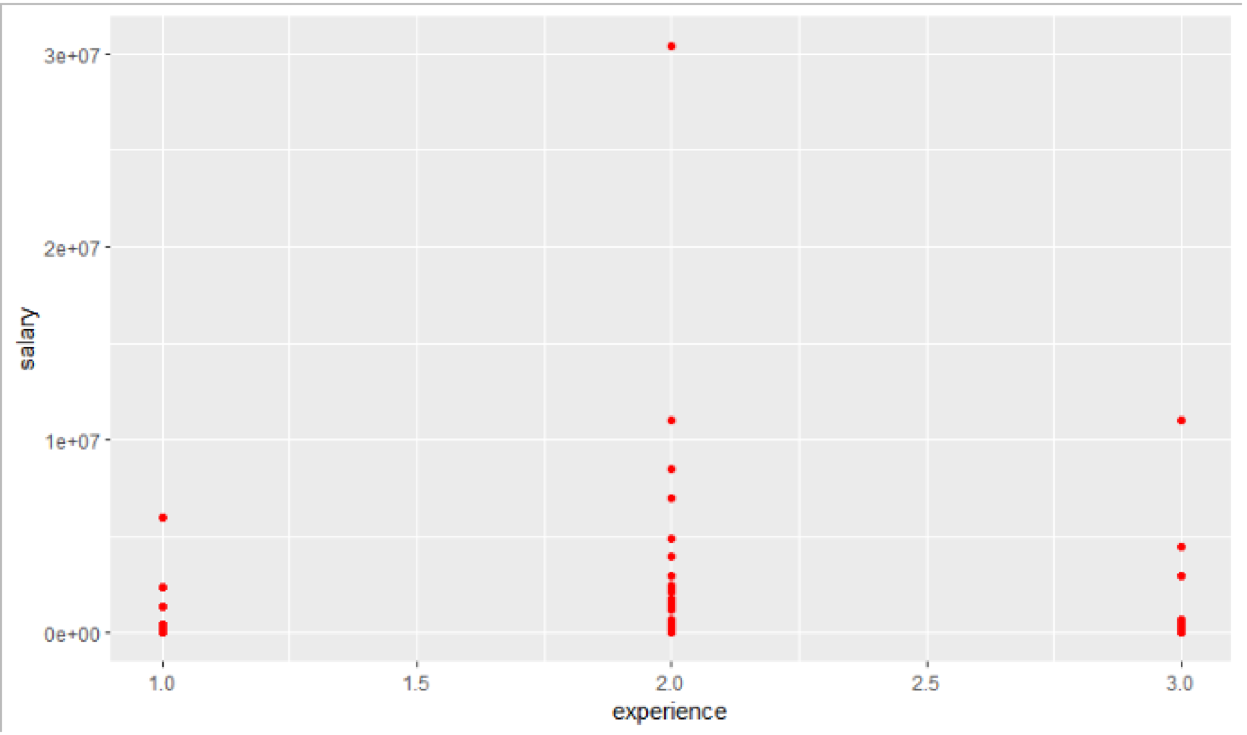
- Scatter plot
  - geom_point() + color

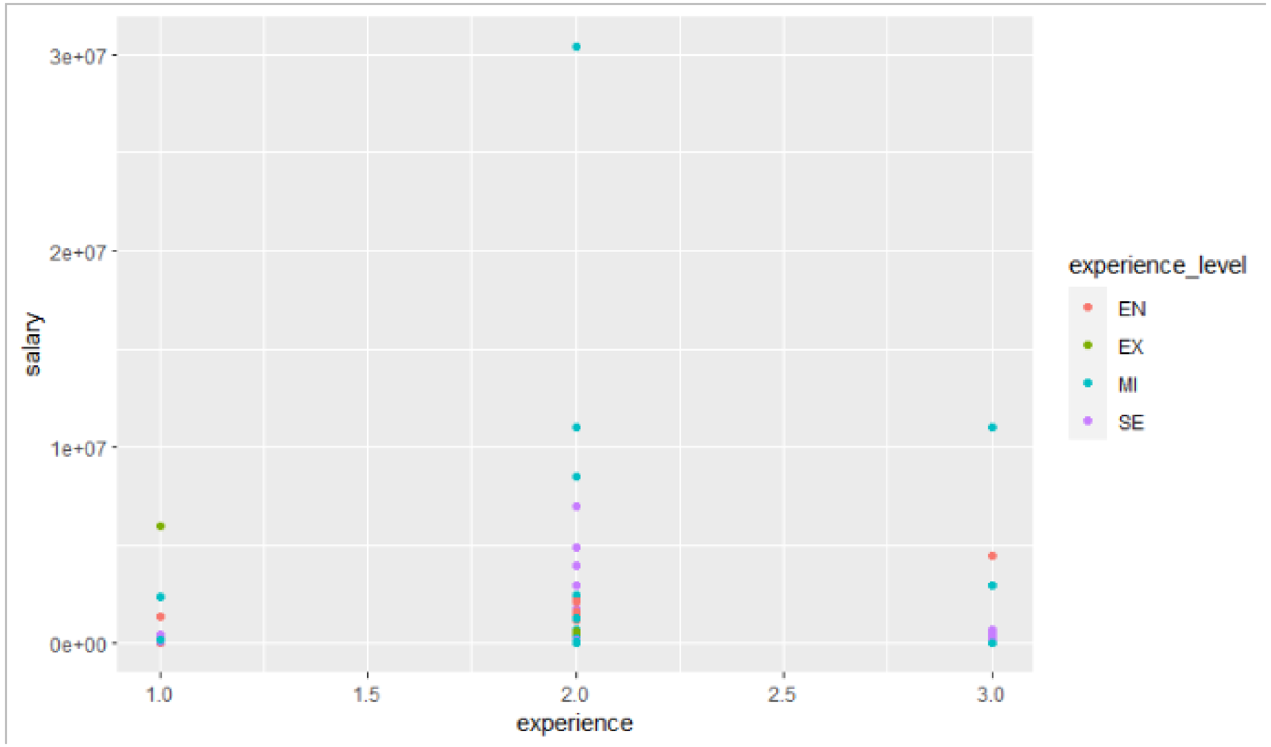


```
> ds_sal %>%
+   ggplot(aes(x=experience, y=salary)) +
+   geom_point(color="red")
```



```
> ds_sal %>%
+   ggplot(aes(x=experience, y=salary,
+               color=experience_level)) +
+   geom_point()
```

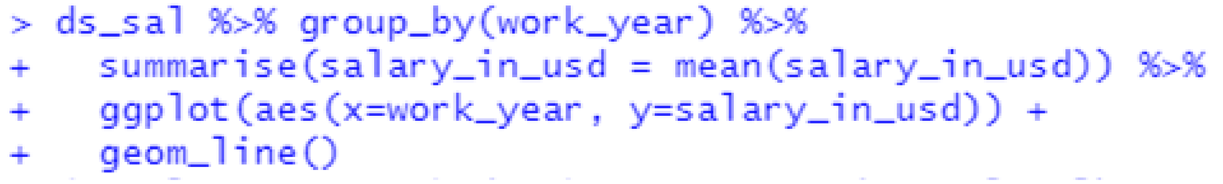




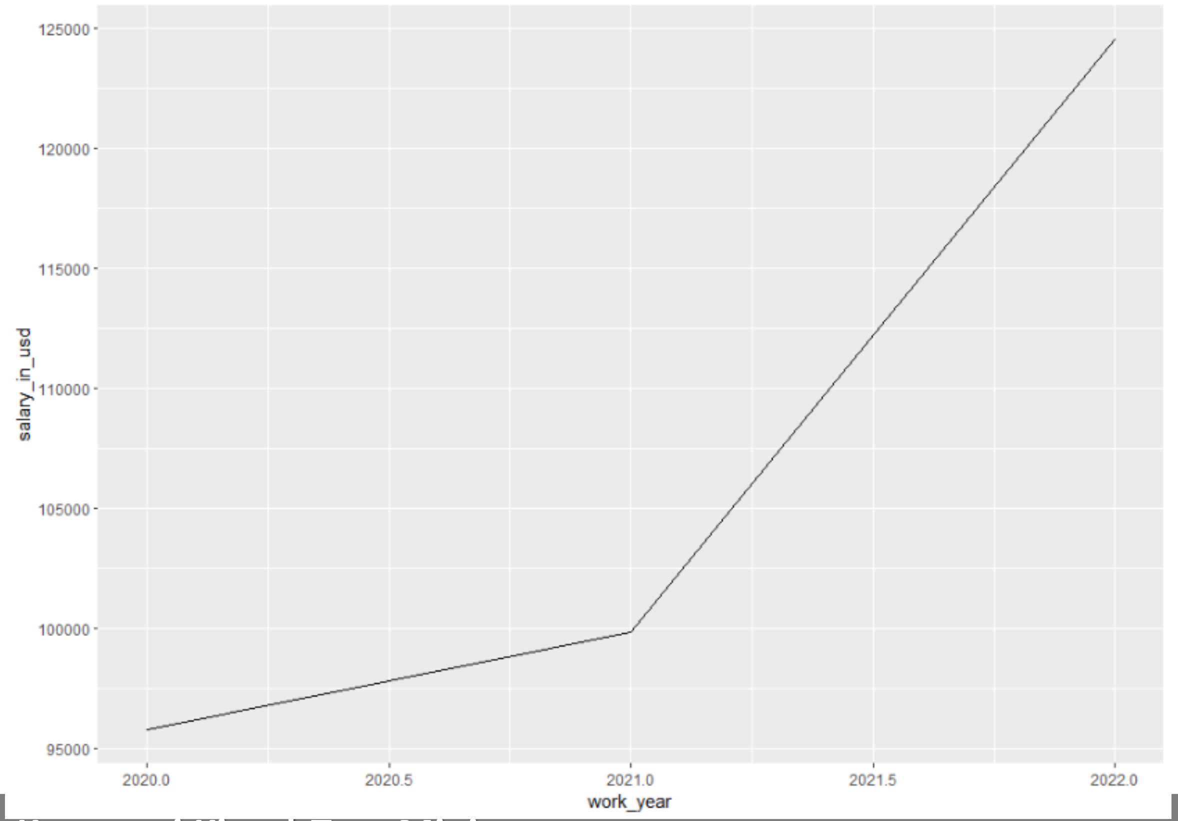
ggplot2

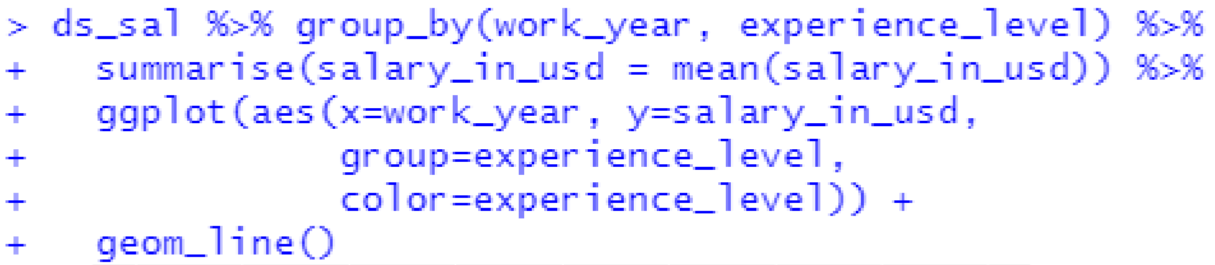
- Line plot
  - geom_line()

EN Entry-level / Junior MI Mid-level / Intermediate SE Senior-level / Expert EX Executive-level

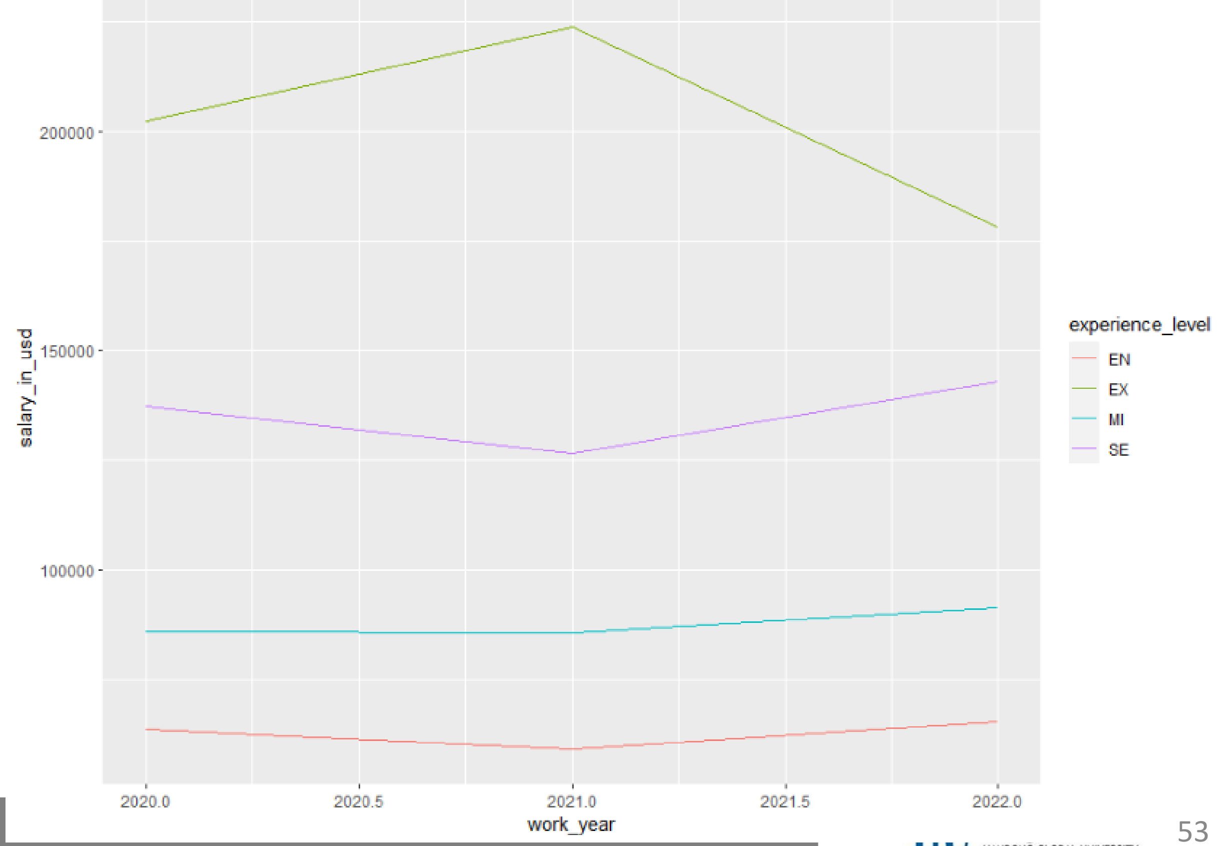


```
> ds_sal %>% group_by(work_year) %>%
+   summarise(salary_in_usd = mean(salary_in_usd)) %>%
+   ggplot(aes(x=work_year, y=salary_in_usd)) +
+   geom_line()
```



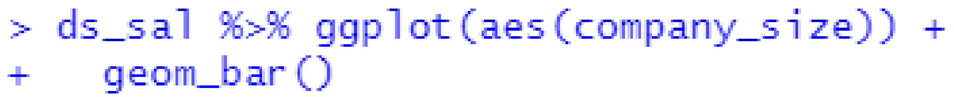


```
> ds_sal %>% group_by(work_year, experience_level) %>%
+   summarise(salary_in_usd = mean(salary_in_usd)) %>%
+   ggplot(aes(x=work_year, y=salary_in_usd,
+              group=experience_level,
+              color=experience_level)) +
+   geom_line()
```

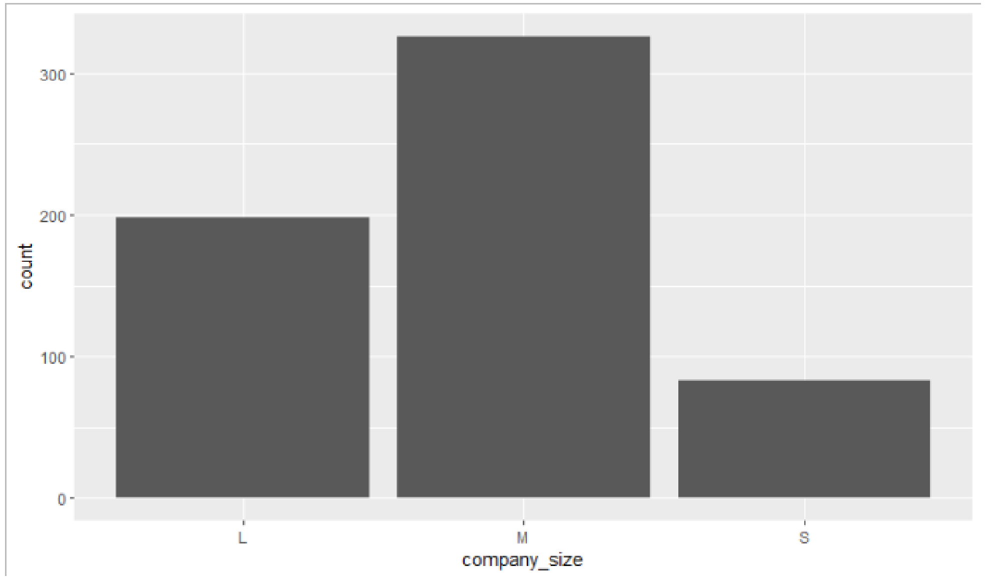


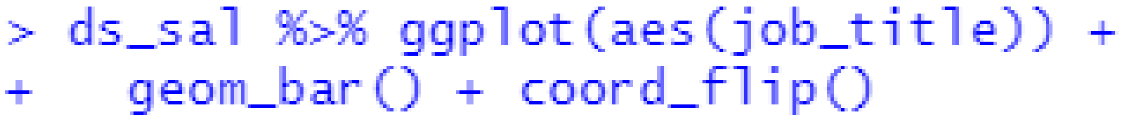
ggplot2

- Bar plot - frequency
  - geom_bar()

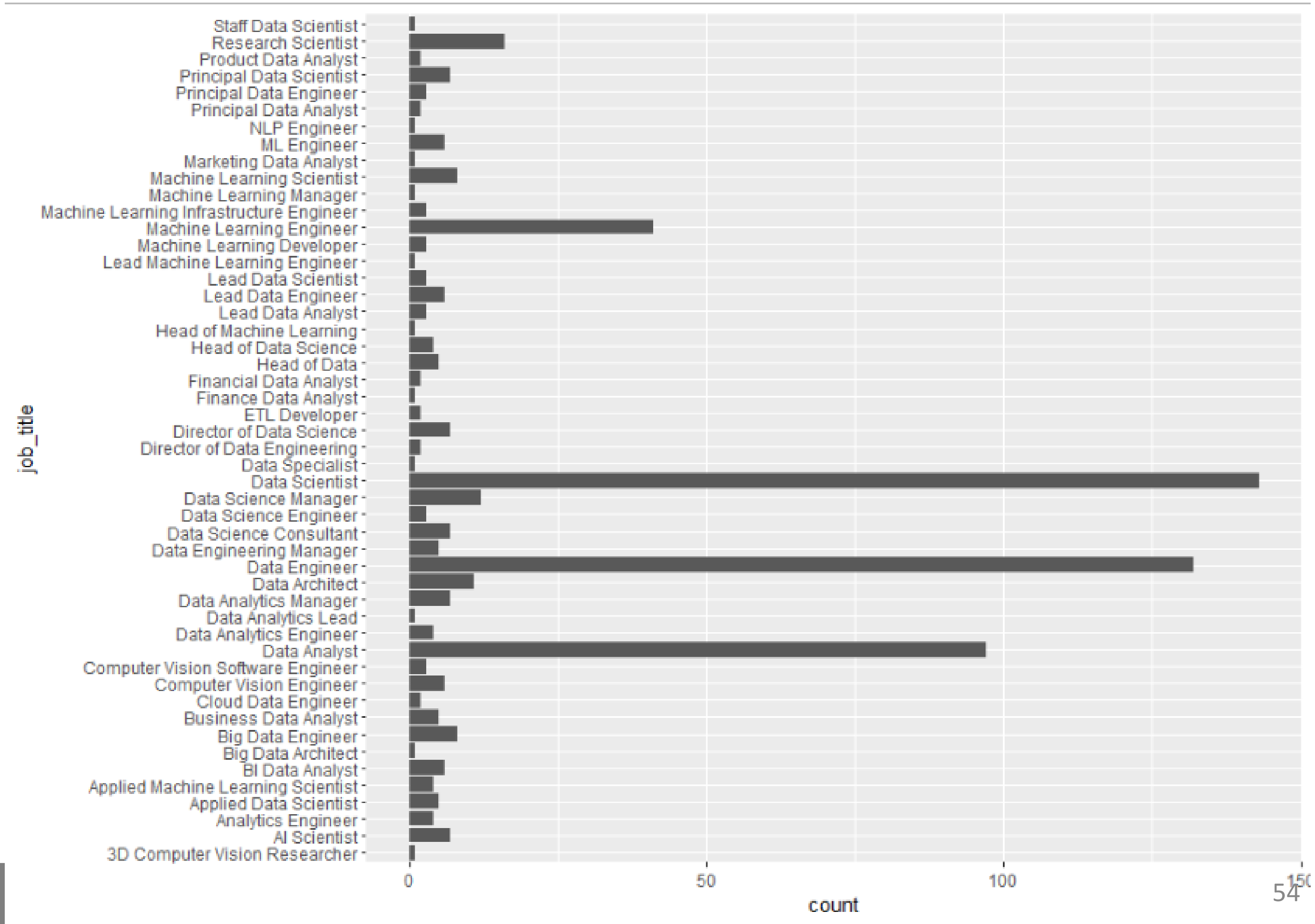


```
> ds_sal %>% ggplot(aes(company_size)) +
+   geom_bar()
```



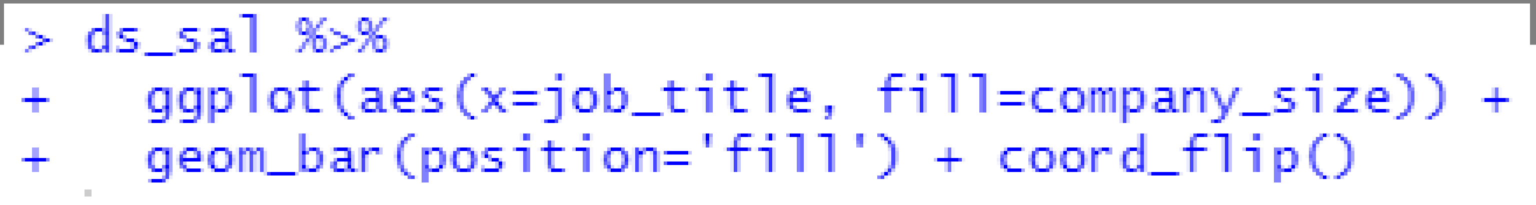


```
> ds_sal %>% ggplot(aes(job_title)) +
+   geom_bar() + coord_flip()
```

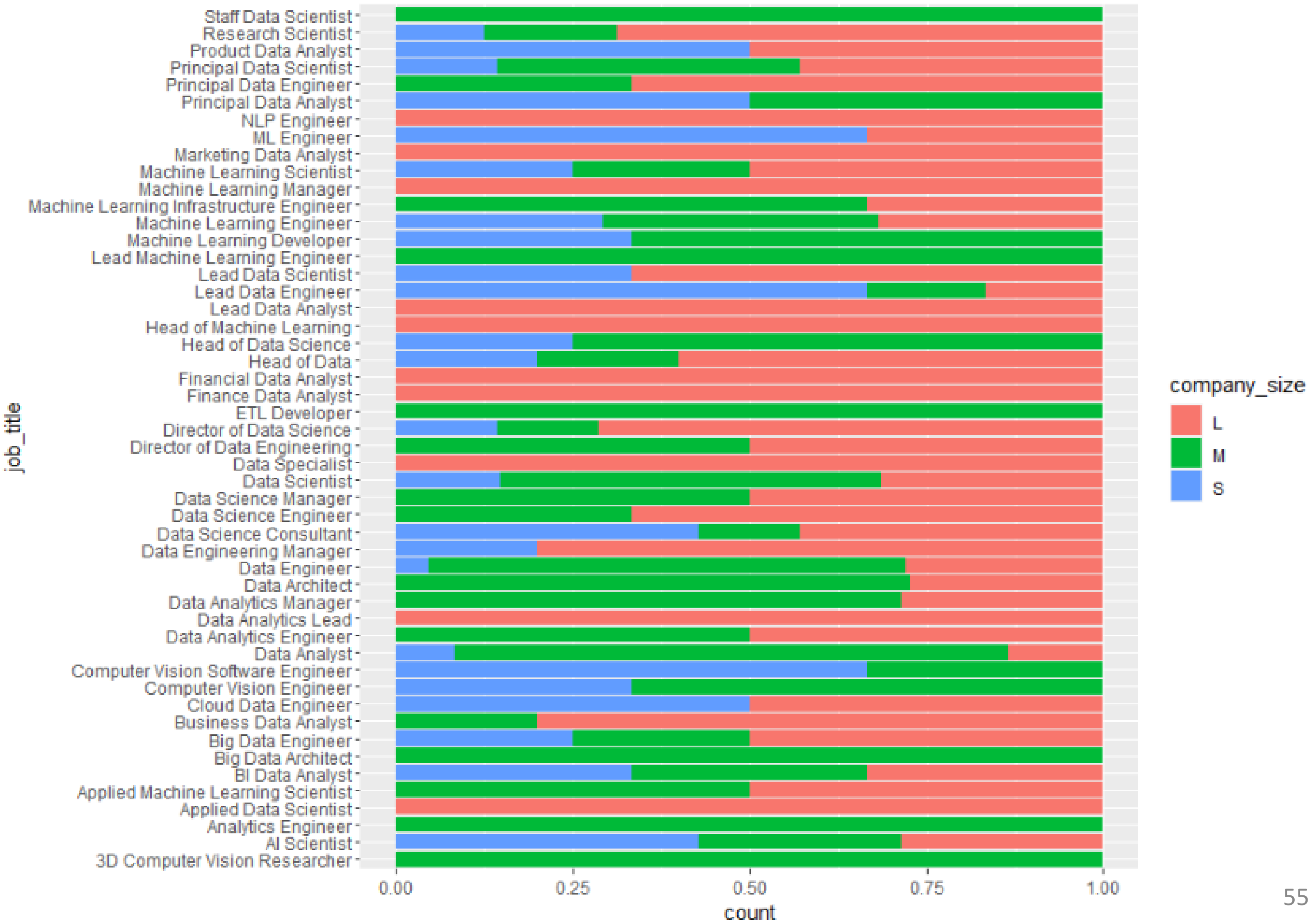


ggplot2

- Bar plot - frequency per group
  - geom_bar()

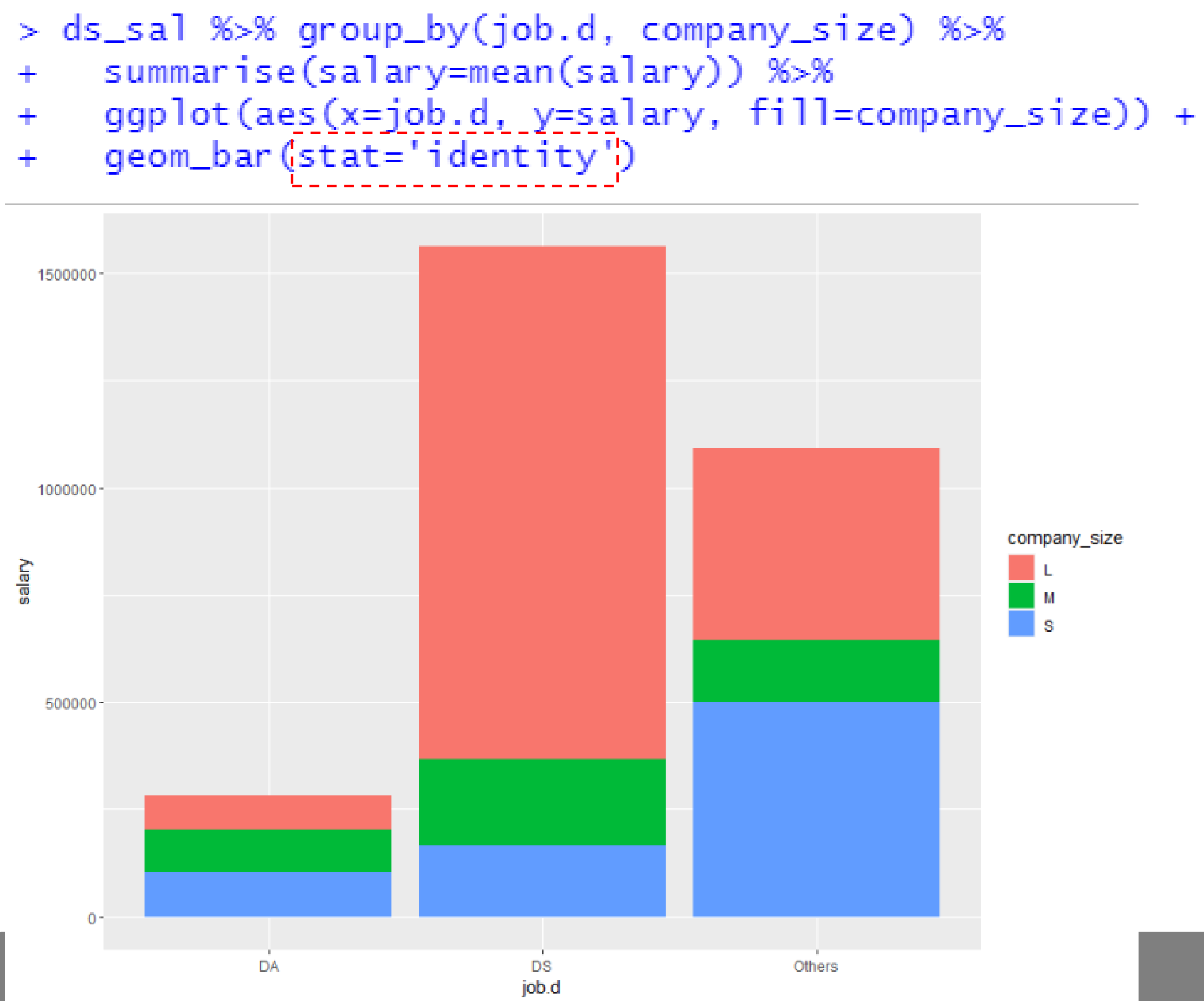


```
> ds_sal %>%
+   ggplot(aes(x=job_title, fill=company_size)) +
+   geom_bar(position='fill') + coord_flip()
```

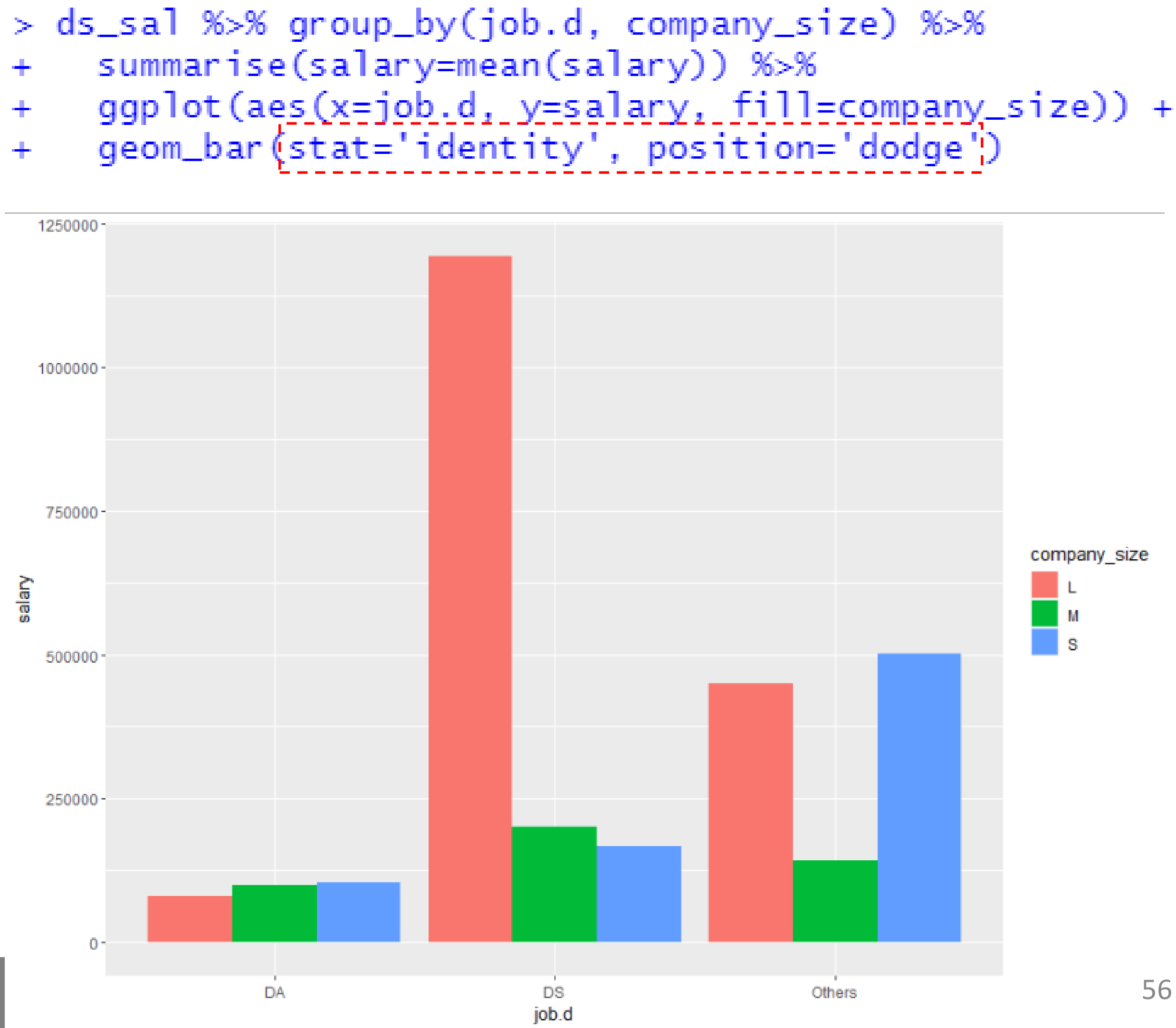


ggplot2

- Bar plot - frequency
  - geom_bar()



```
> ds_sal %>% group_by(job.d, company_size) %>%
+   summarise(salary=mean(salary)) %>%
+   ggplot(aes(x=job.d, y=salary, fill=company_size)) +
+   geom_bar(stat='identity')
```

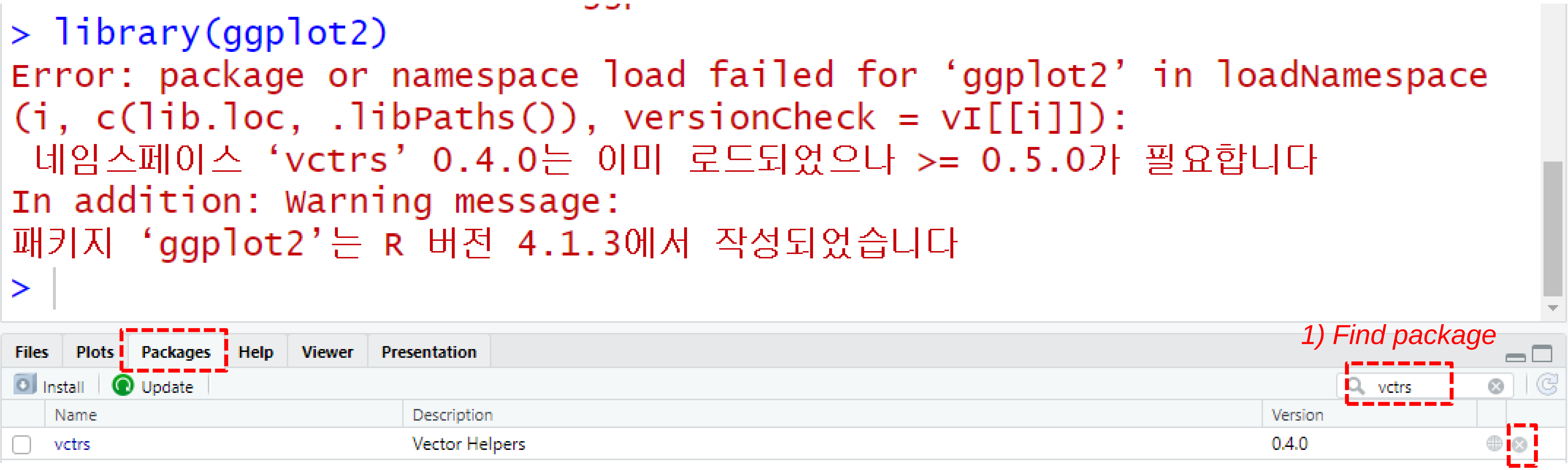


```
> ds_sal %>% group_by(job.d, company_size) %>%
+   summarise(salary=mean(salary)) %>%
+   ggplot(aes(x=job.d, y=salary, fill=company_size)) +
+   geom_bar(stat='identity', position='dodge')
```

Possible Errors

Unexpected errors

- Version issue



```
> library(ggplot2)
Error: package or namespace load failed for ‘ggplot2’ in loadNamespace
(i, c(lib.loc, .libPaths()), versionCheck = vI[[i]]):
 네임스페이스 ‘vctrs’ 0.4.0는 이미 로드되었으나 >= 0.5.0가 필요합니다
In addition: Warning message:
패키지 ‘ggplot2’는 R 버전 4.1.3에서 작성되었습니다
> 
```

	Name	Description	Version
□	vctrs	Vector Helpers	0.4.0

1) Find package

2) Click (remove)

3) install.packages(‘vctrs’)