Introduction to R Workshop (Session 1)

Dr Carlos Moreno-Garcia

School of Computing

Robert Gordon University

What is R?

- R is an open-source programming language for statistical analysis and graphics.
- Command-line based, however, complementary tools provide a friendly user interface.
- Will require you to learn both syntax and semantic of R.

What do we use R for?

- Exploratory and statistical data analysis.

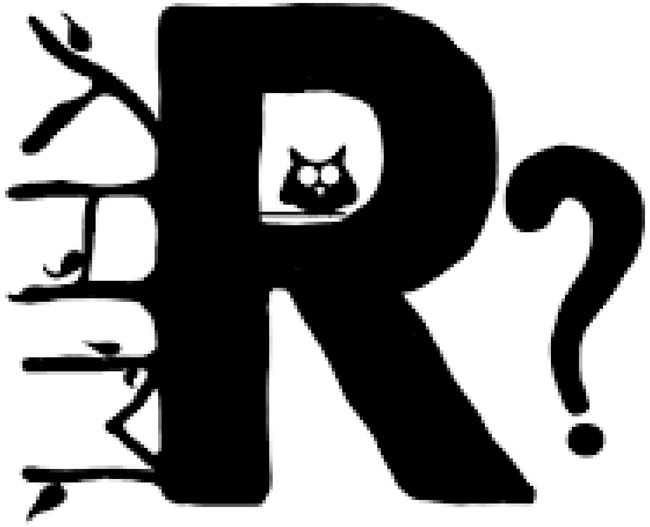
- Visualisation and graphics.

- Data preparation (data wrangling).

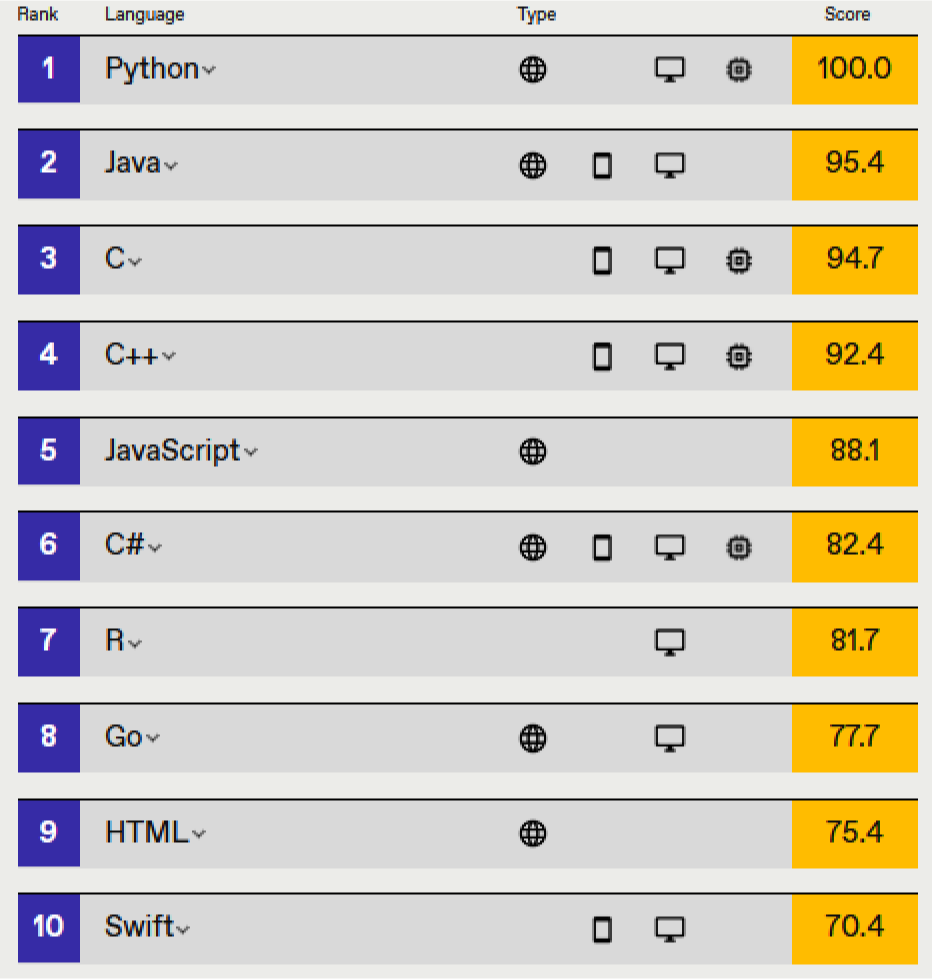
- Machine learning and modelling.

Why R?

- Free.
- Easy to use.
- Has a package for everything.
- Has a great online support community.
- Is a statistical tool AND a programming language.
- Available across platforms.
- Similar to Python and MATLAB.
- Robust for visualisations.
- You can produce reports of your work easily.

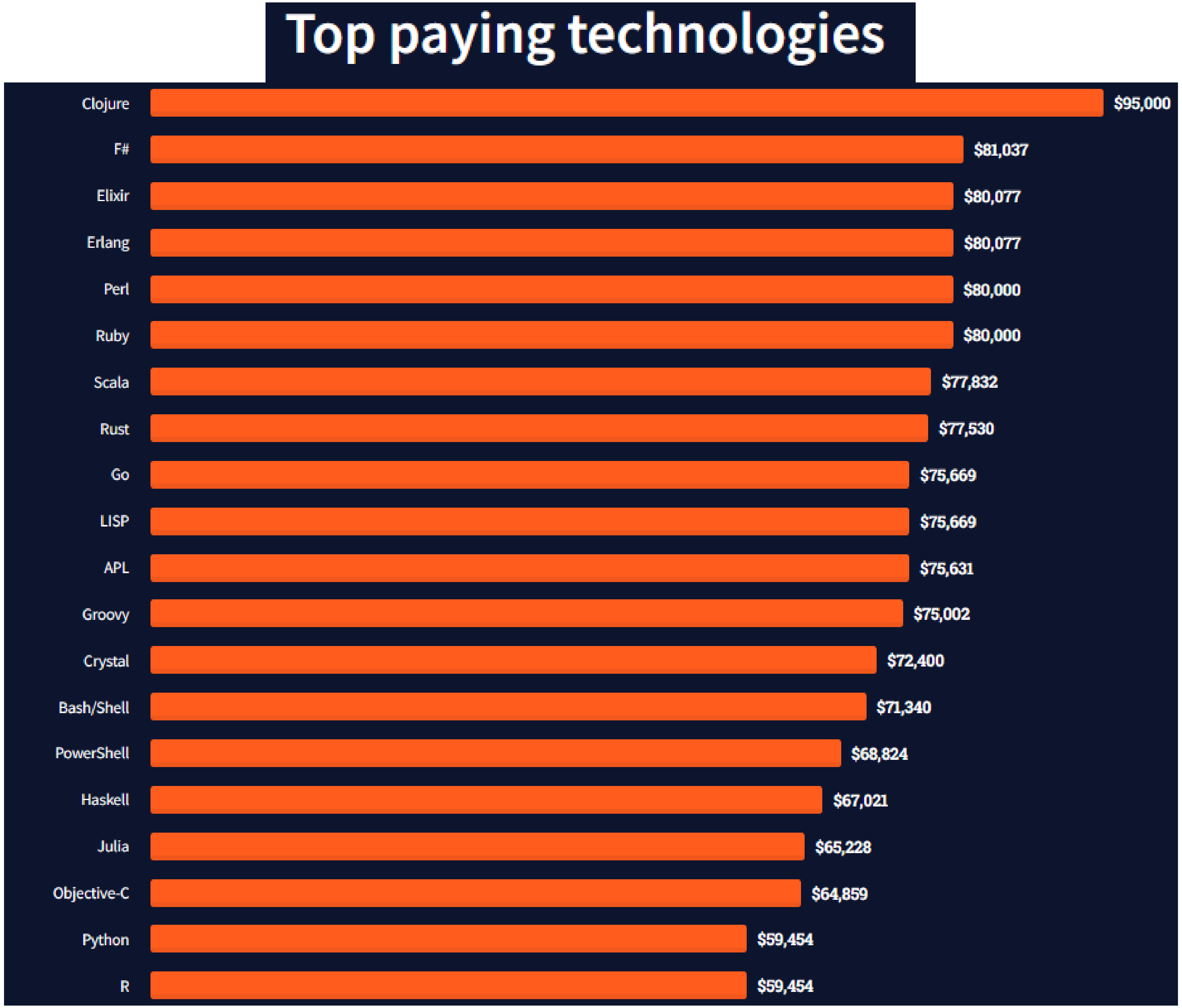


Some stats



Rank	Language	Type				Score
1	Python	🌐		🖥	▣	100.0
2	Java	🌐	📱	🖥		95.4
3	C		📱	🖥	▣	94.7
4	C++		📱	🖥	▣	92.4
5	JavaScript	🌐				88.1
6	C#	🌐	📱	🖥	▣	82.4
7	R			🖥		81.7
8	Go	🌐		🖥		77.7
9	HTML	🌐				75.4
10	Swift		📱	🖥		70.4

https://spectrum.ieee.org/top-programming-languages/



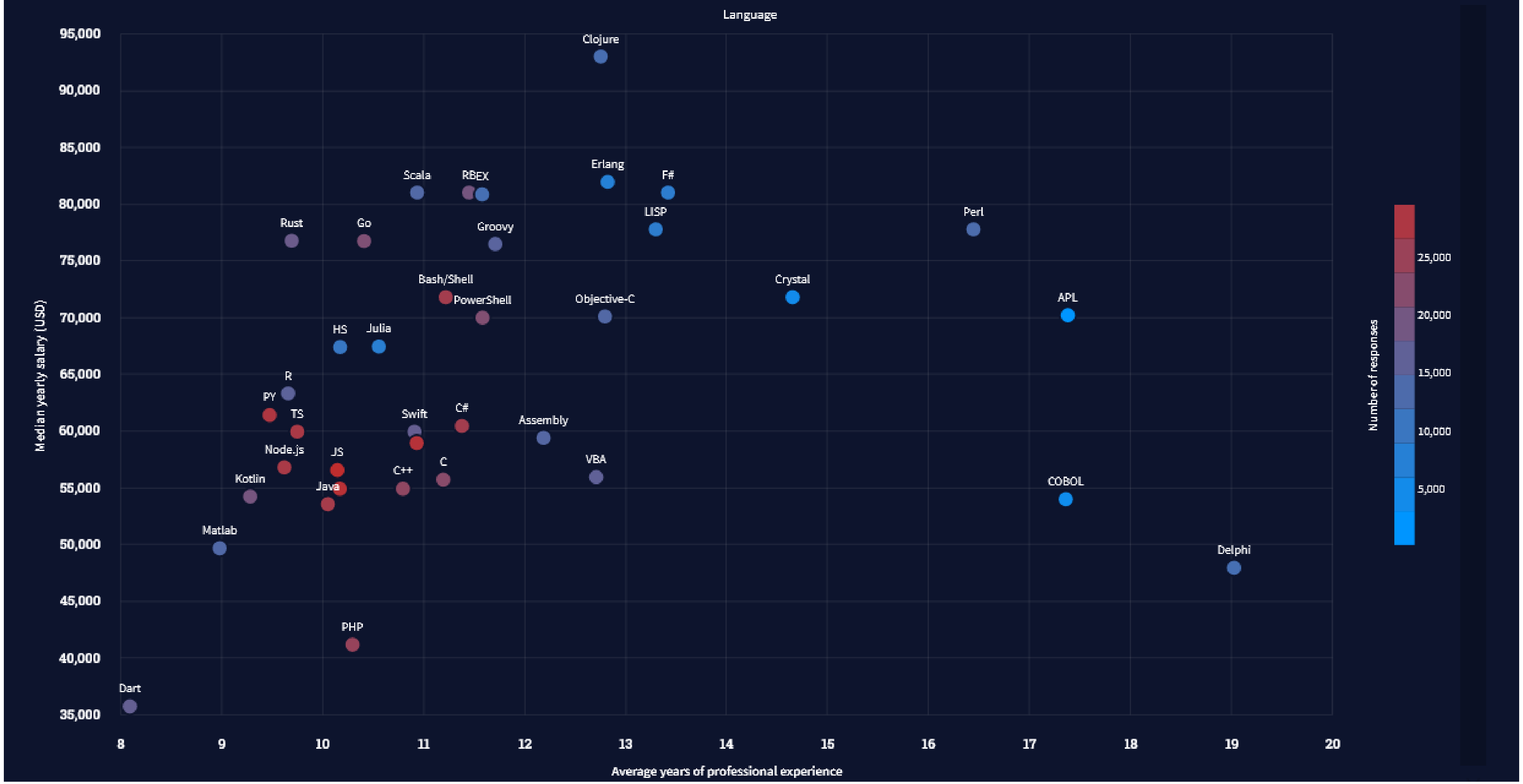
Top paying technologies

https://insights.stackoverflow.com/survey/2021

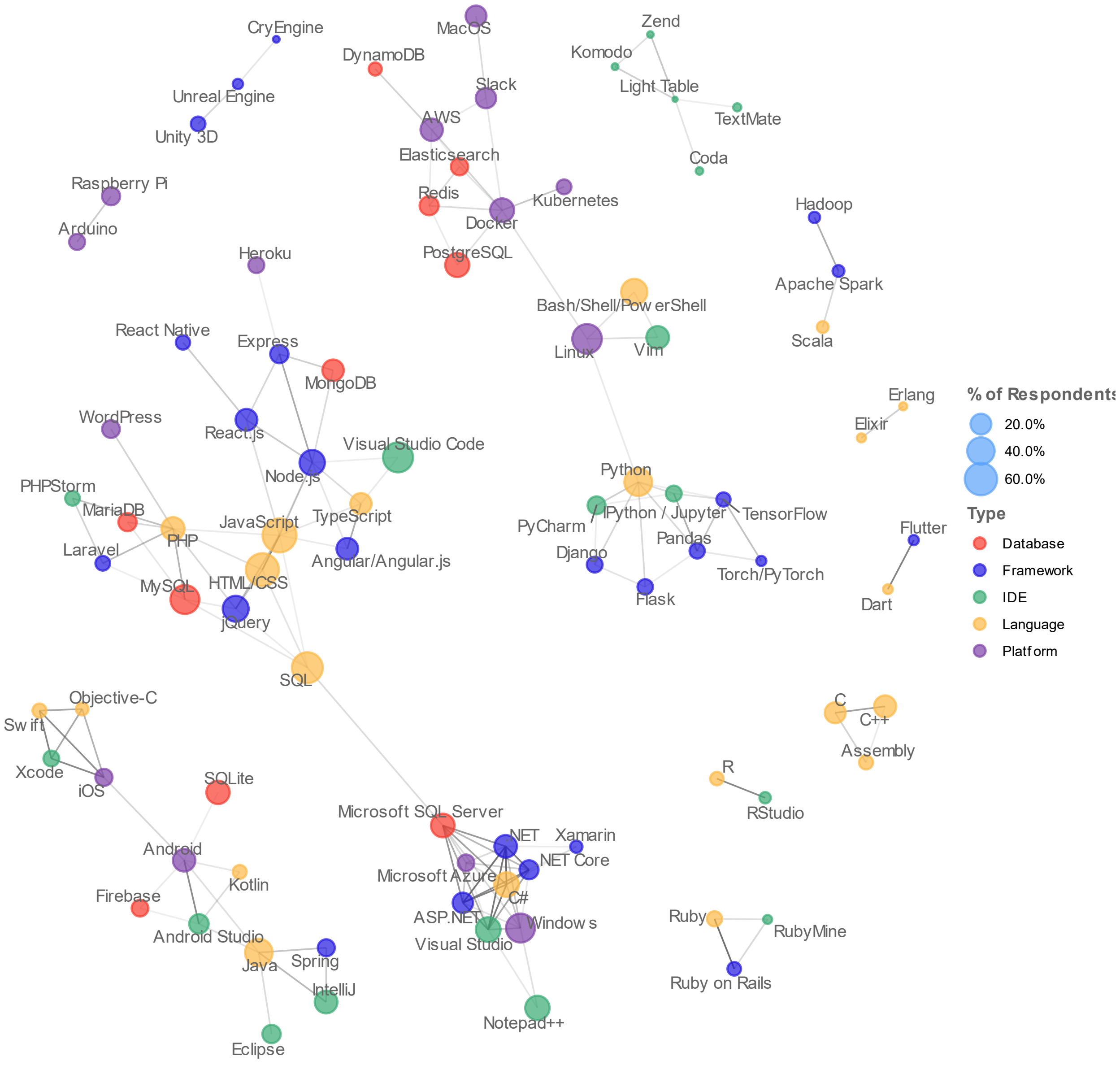
Salary and experience by language

PHP developers are disproportionately underpaid compared to other languages with the same experience.

43,676 responses



How Technologies Are Connected



https://insights.stackoverflow.com/survey/2019

Getting and installing R

1. Download R

   http://cran.r-project.org/

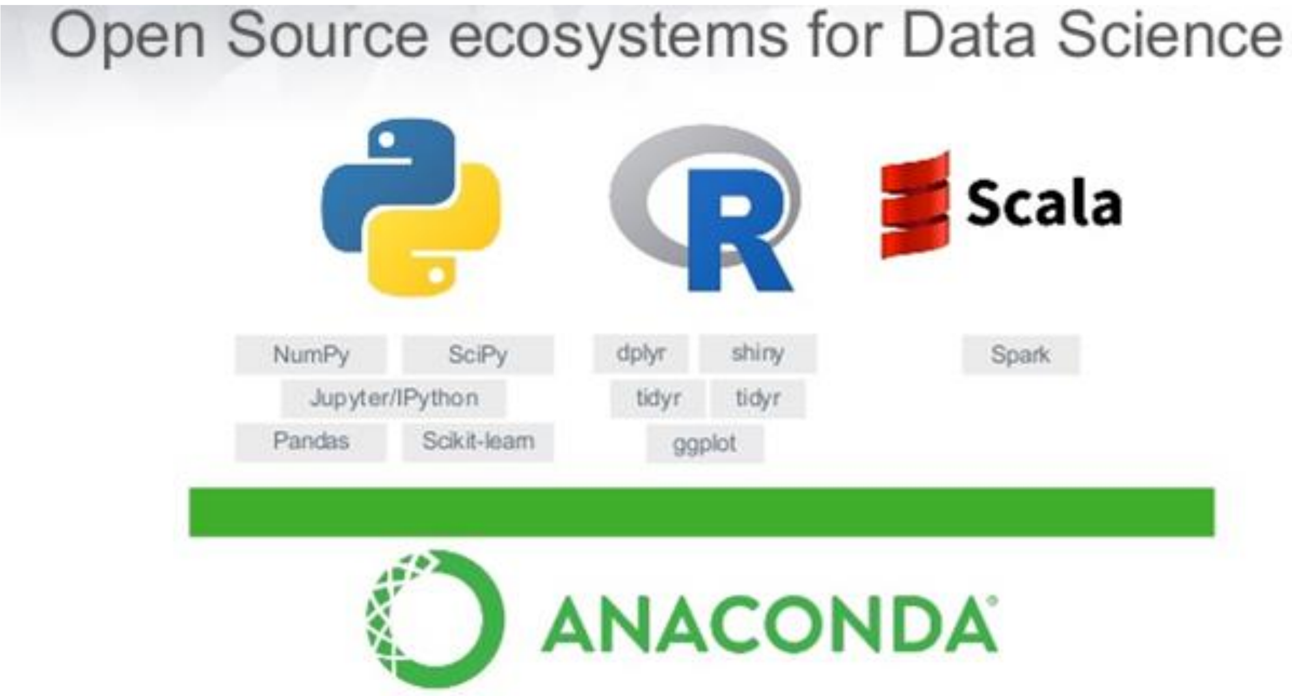
2. Download RStudio

   http://www.rstudio.com

OR

1. Download Anaconda

   https://www.anaconda.com/



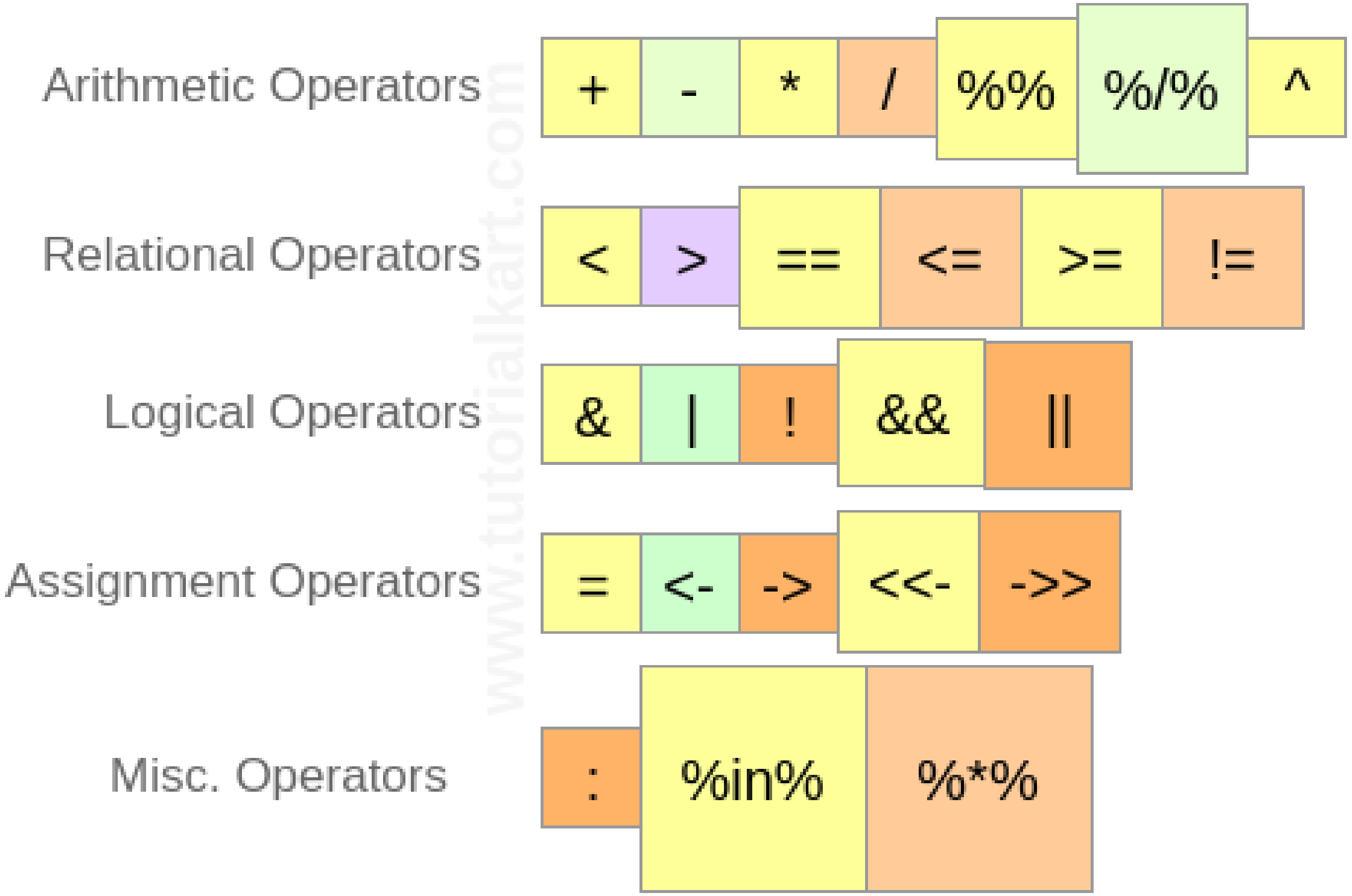
Introduction to R: operators & data types

- R in principle is like a calculator, you execute a command and R responds.

- These commands are mathematical operations.

- Therefore, we need to first know the operators and the data types.

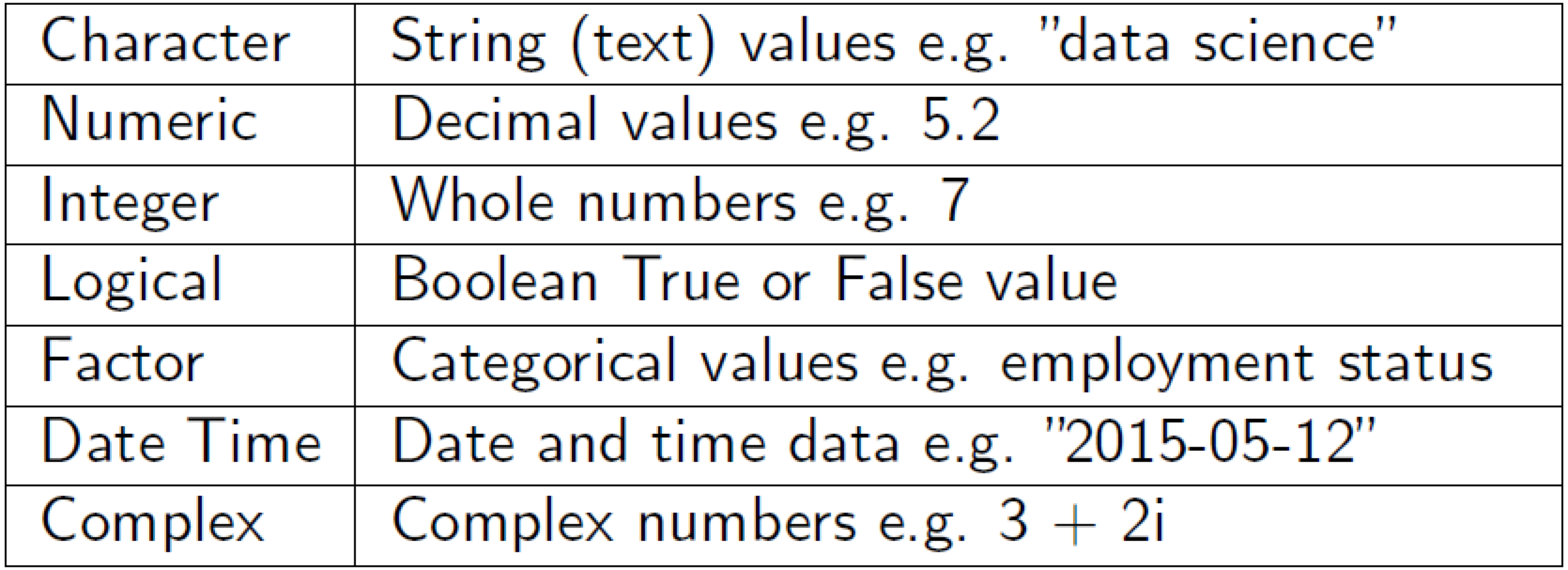
Operators



Arithmetic Operators	+	-	*	/	%%	%/%	^
Relational Operators	<	>	==	<=	>=	!=	
Logical Operators	&	|	!	&&	||		
Assignment Operators	=	<-	->	<<-	->>		
Misc. Operators	:	%in%	%*%				

https://www.tutorialkart.com/r-tutorial/r-operators/

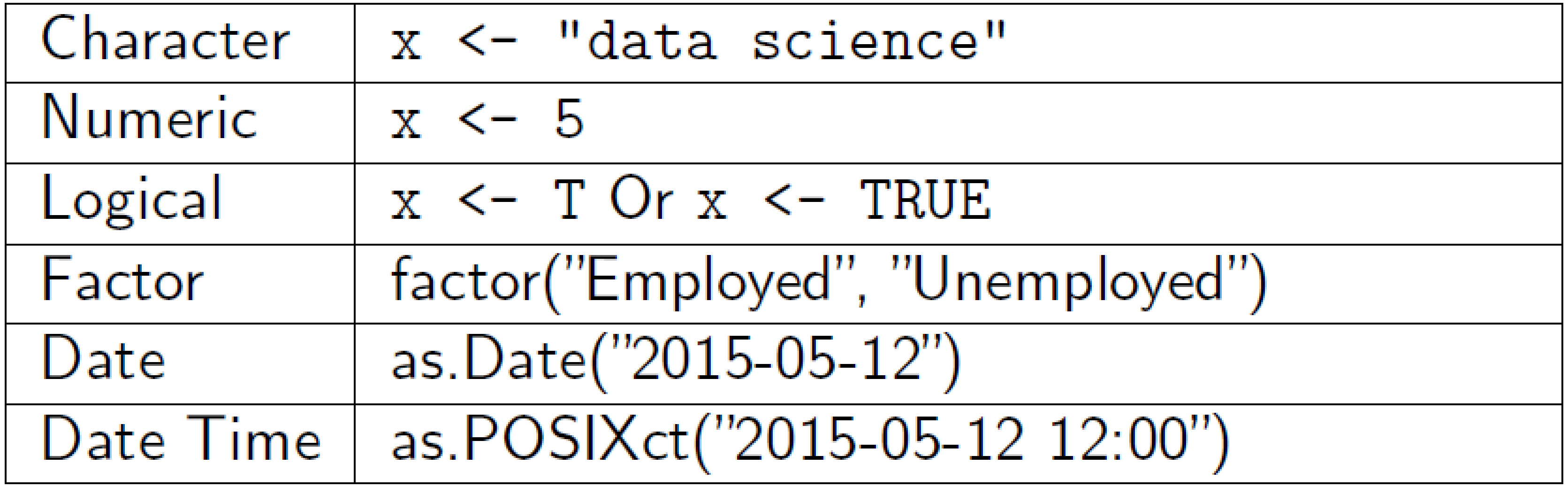
Data types



Character	String (text) values e.g. "data science"
Numeric	Decimal values e.g. 5.2
Integer	Whole numbers e.g. 7
Logical	Boolean True or False value
Factor	Categorical values e.g. employment status
Date Time	Date and time data e.g. "2015-05-12"
Complex	Complex numbers e.g. $3 + 2i$

Variables

- A variable is a named placeholder for data.
- Initialised using the assignment operator "<-" or "=“.

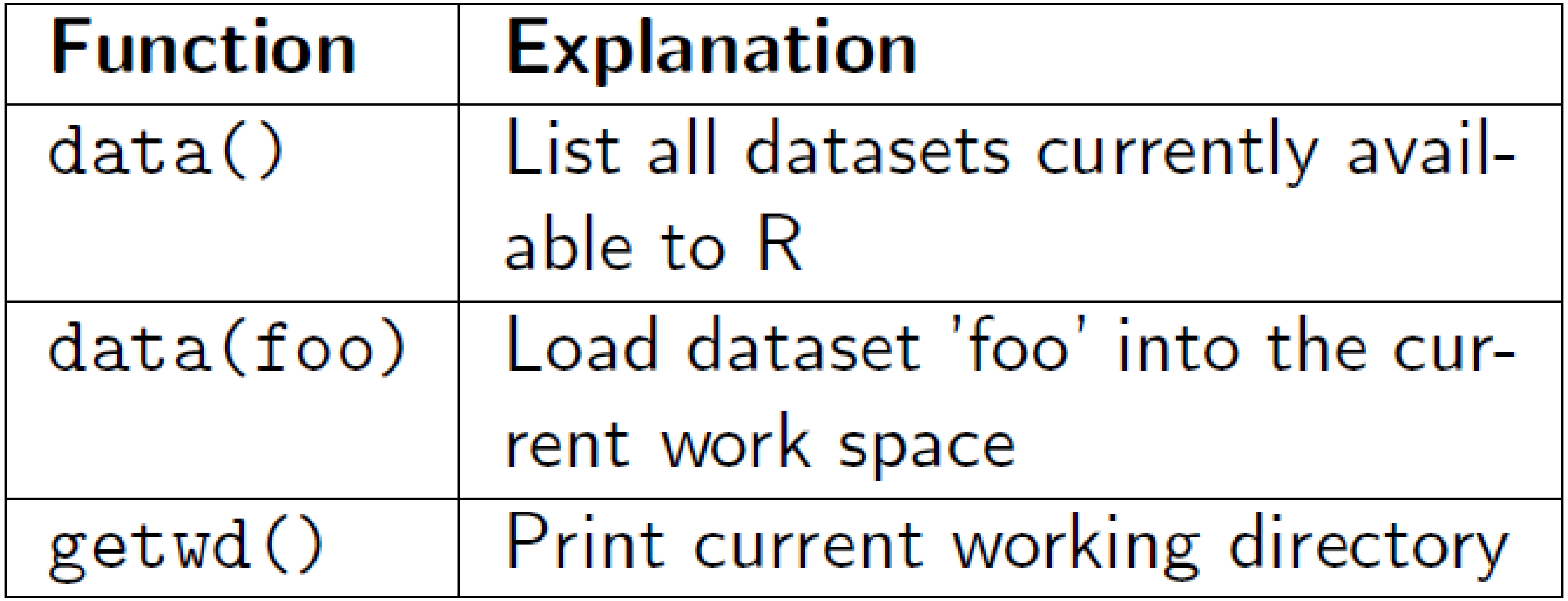


Character	`x <- "data science"`
Numeric	`x <- 5`
Logical	`x <- T` Or `x <- TRUE`
Factor	factor(”Employed”, ”Unemployed”)
Date	as.Date(”2015-05-12”)
Date Time	as.POSIXct(”2015-05-12 12:00”)

Functions

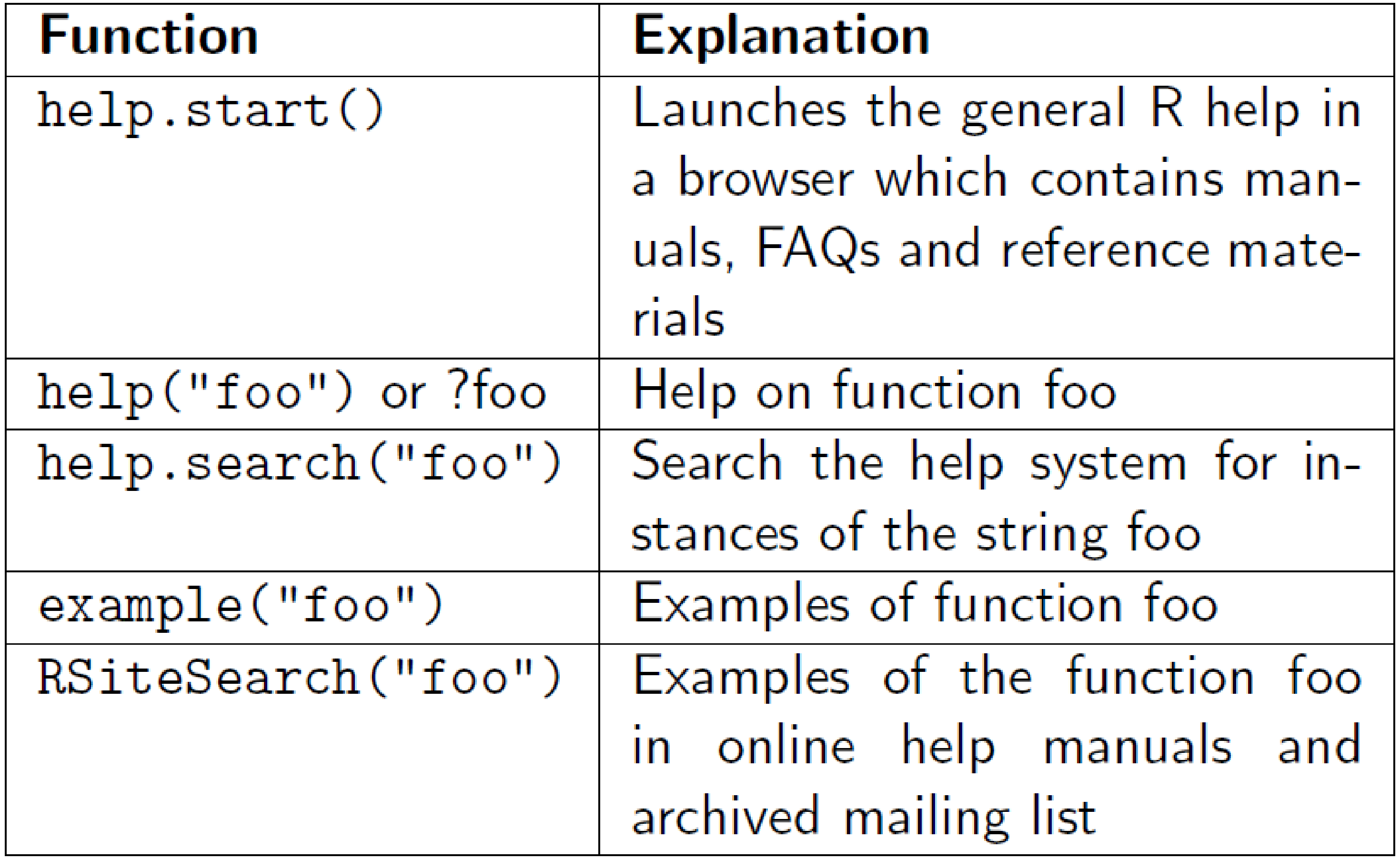
- A function is a named group of code that is used to give instructions to R.
- May or may not accept input parameters.
- May or may not return a value.
- R comes with an extensive collection of in-built functions.
- General syntax of a function:
  - A function can be identified by the “()” after the name.
  - Example: **function.name(parameters)**.
  - A “.” in a function’s name is just there to split words!

Examples of functions



Function	Explanation
`data()`	List all datasets currently available to R
`data(foo)`	Load dataset 'foo' into the current work space
`getwd()`	Print current working directory

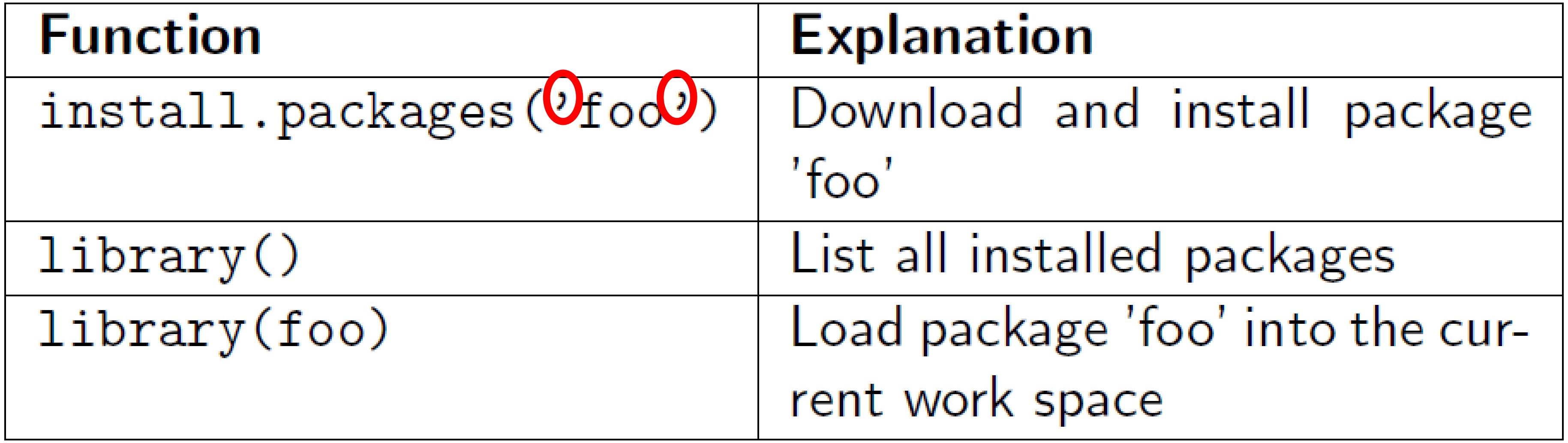
Getting help



Function	Explanation
`help.start()`	Launches the general R help in a browser which contains manuals, FAQs and reference materials
`help("foo")` or ?foo	Help on function foo
`help.search("foo")`	Search the help system for instances of the string foo
`example("foo")`	Examples of function foo
`RSiteSearch("foo")`	Examples of the function foo in online help manuals and archived mailing list

Packages

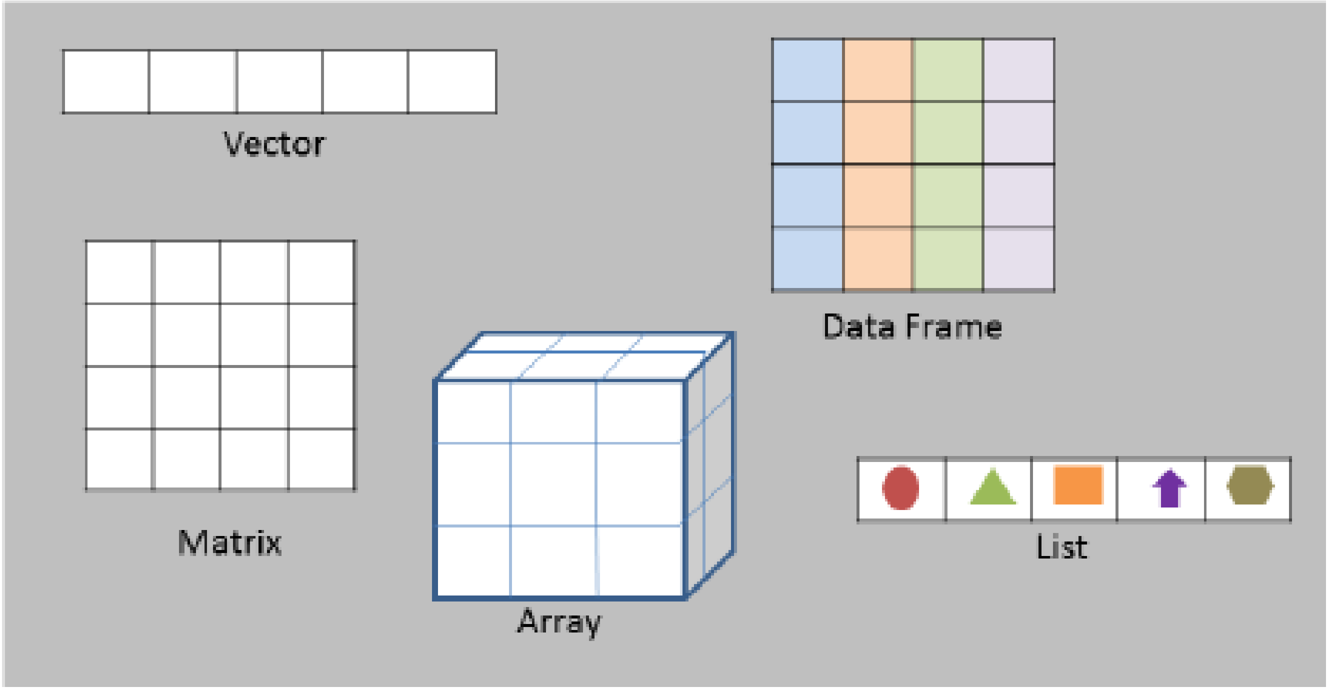
- Packages are named collections of (related) functions and data that are encapsulated and distributed together.



Function	Explanation
`install.packages('foo')`	Download and install package 'foo'
`library()`	List all installed packages
`library(foo)`	Load package 'foo' into the current work space

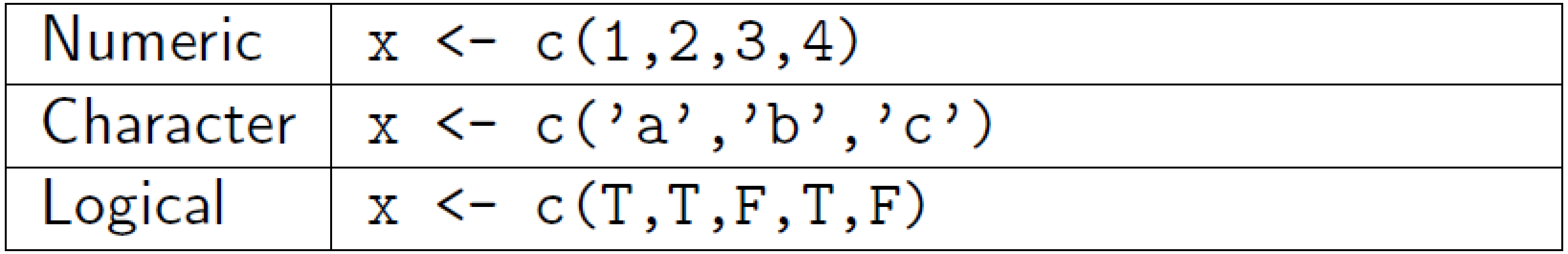
Data structures

- R provides the following structures to collect data types/variables:



Vectors

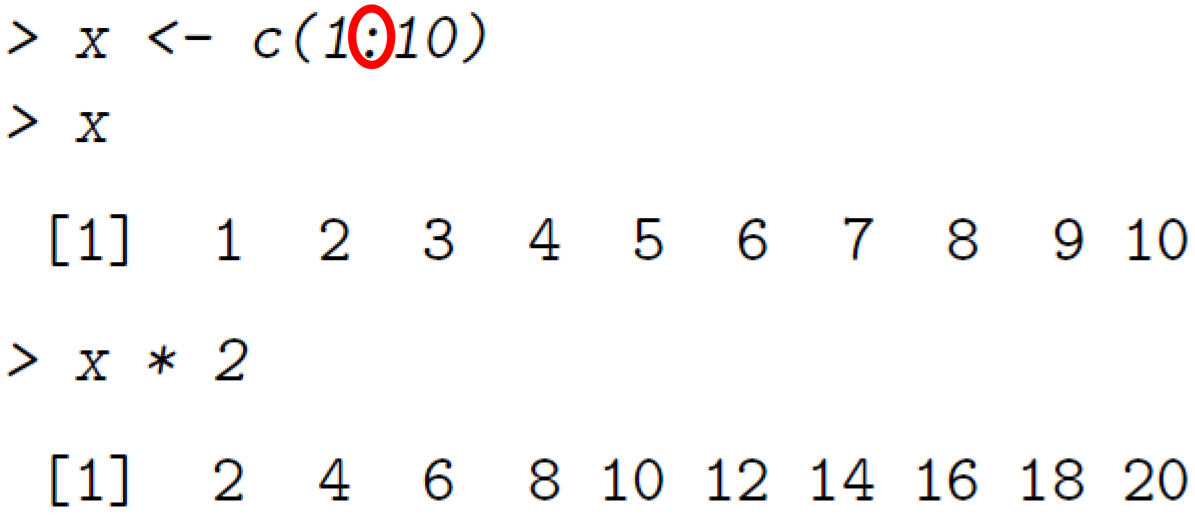
- Vectors are created using the c() function.
- general syntax is c(val_1, val_2, ..., val_n).
- *val* can be any valid data type AS LONG AS IT REMAINS THE SAME.



Numeric	`x <- c(1,2,3,4)`
Character	`x <- c(’a’,’b’,’c’)`
Logical	`x <- c(T,T,F,T,F)`

Vectorised operations

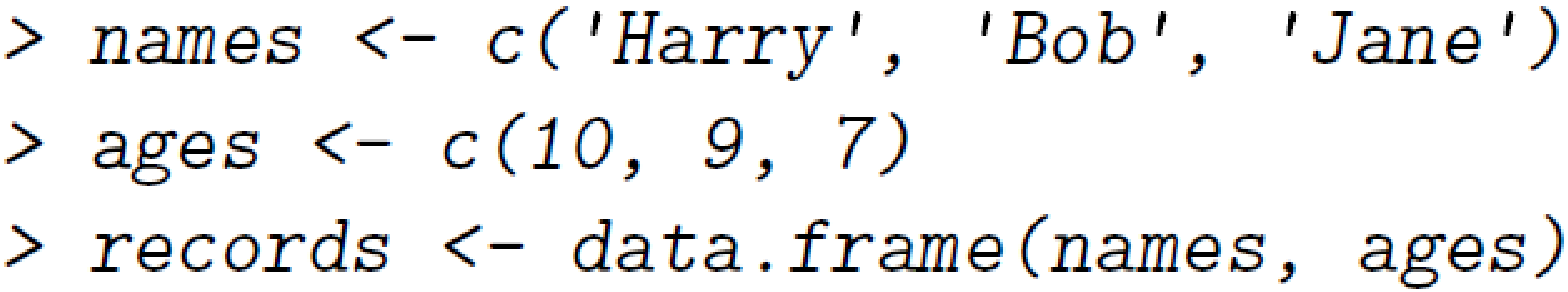
- Operations in R are applied to entire vectors, instead of individual data elements within the vectors.



```
> x <- c(1:10)
> x
 [1]  1  2  3  4  5  6  7  8  9 10
> x * 2
 [1]  2  4  6  8 10 12 14 16 18 20
```

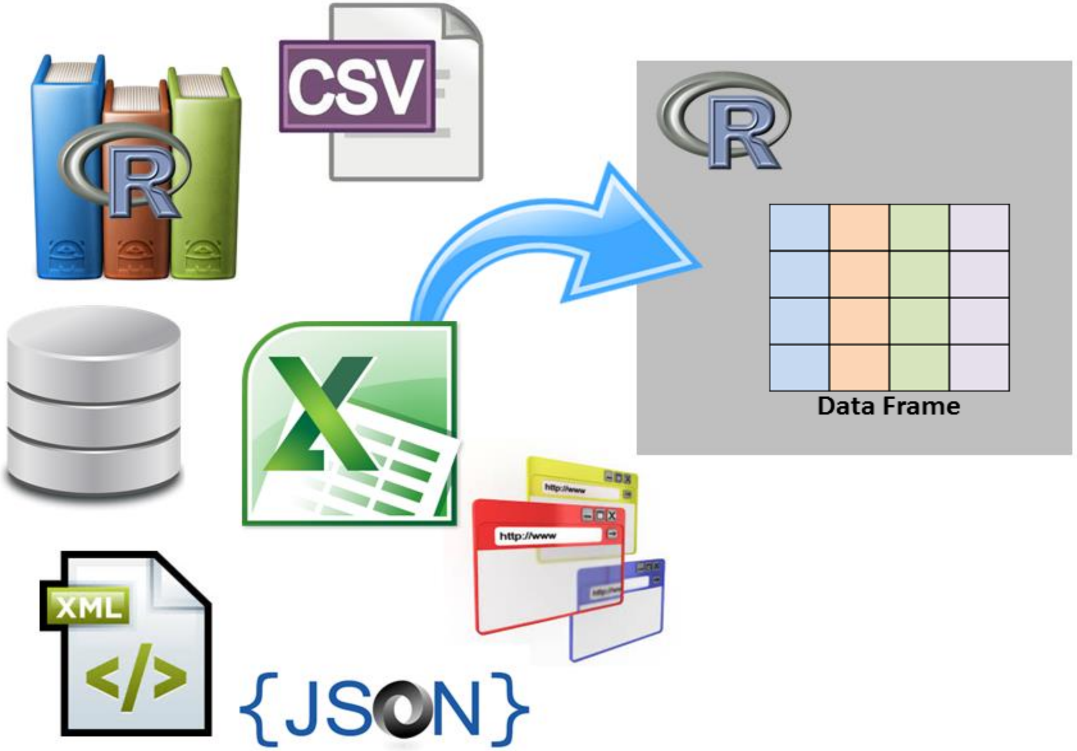
Data frames

- Data Frames are created using the data.frame() function
- general syntax is data.frame(vect1, vect2, ..., vectn).



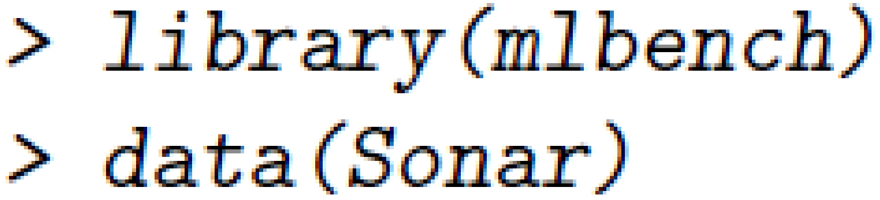
```
> names <- c('Harry', 'Bob', 'Jane')
> ages <- c(10, 9, 7)
> records <- data.frame(names, ages)
```

Loading data into R



Loading data from packages

- R comes pre-installed with a number of datasets.
- Third-party packages also come with more datasets.
- Use the function data() to see a list of installed datasets.
- Load the 'Sonar' dataset from the ‘mlbench’ library:



```
> library(mlbench)
> data(Sonar)
```

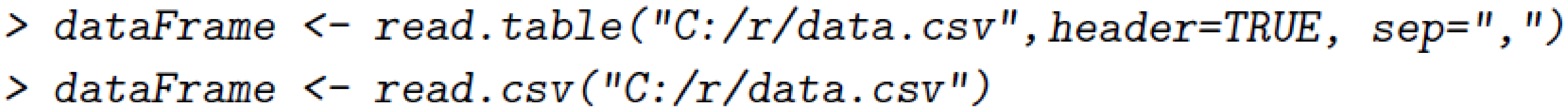
- This dataset contains 208 observations of the classification of sonar signals (Mines vs Rocks).

https://www.rdocumentation.org/packages/mlbench/versions/2.1-1/topics/Sonar

Loading Data from .csv files

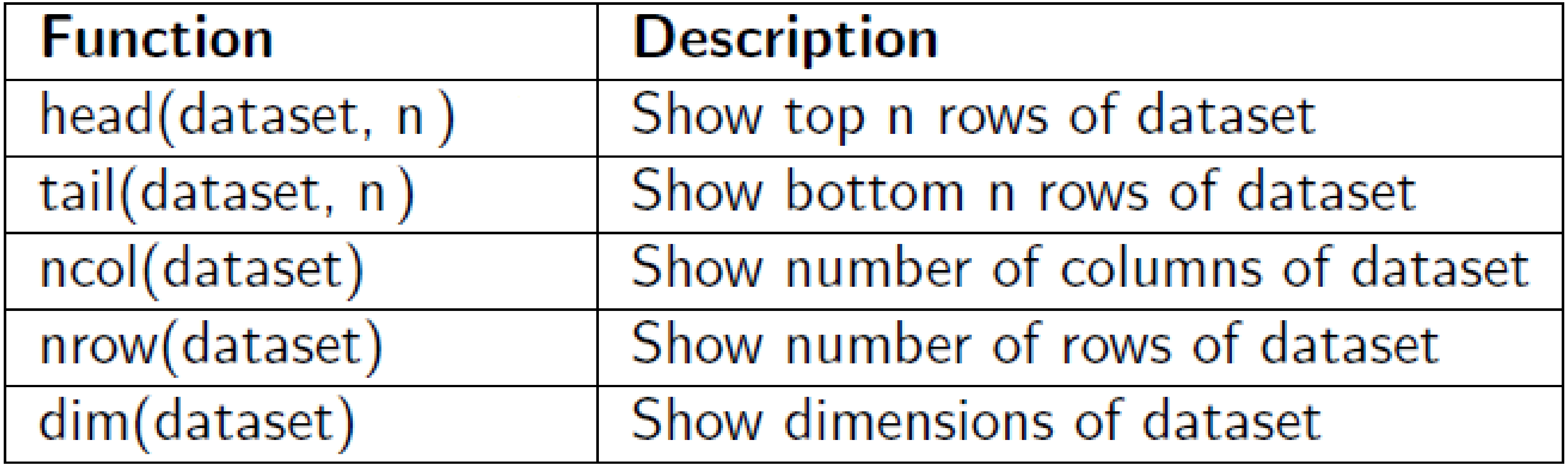
- Delimited files can be read in R using the read.table() function.

- Comma separated value (CSV) files can also be read using the read.csv() function.



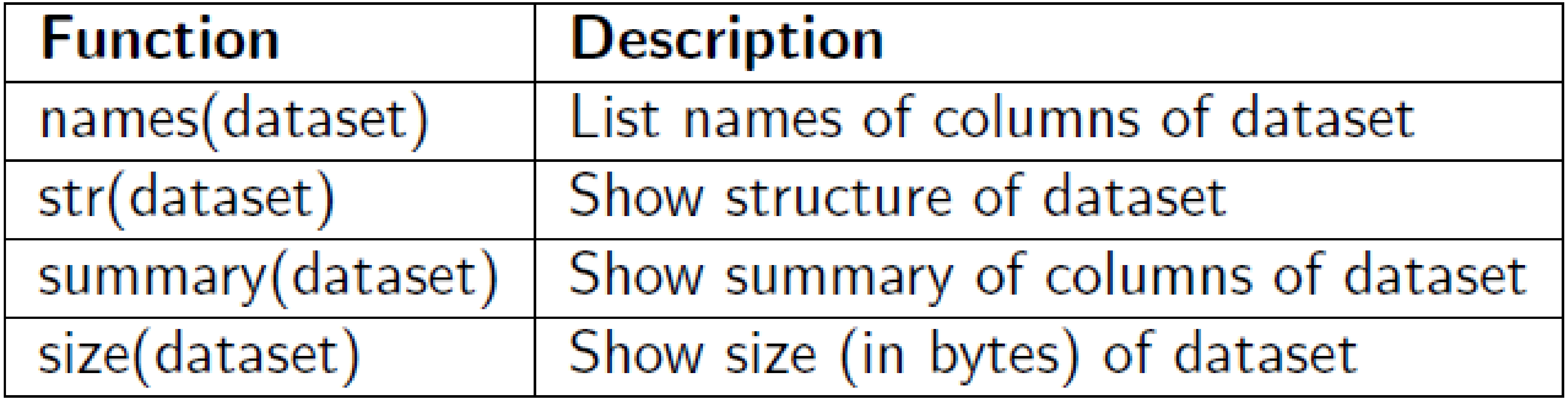
```
> dataFrame <- read.table("C:/r/data.csv",header=TRUE, sep=",")
> dataFrame <- read.csv("C:/r/data.csv")
```

Exploring data



Function	Description
head(dataset, n)	Show top n rows of dataset
tail(dataset, n)	Show bottom n rows of dataset
ncol(dataset)	Show number of columns of dataset
nrow(dataset)	Show number of rows of dataset
dim(dataset)	Show dimensions of dataset

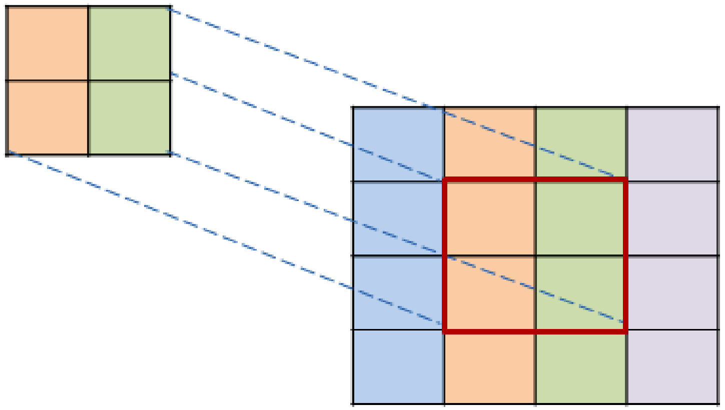
Exploring data



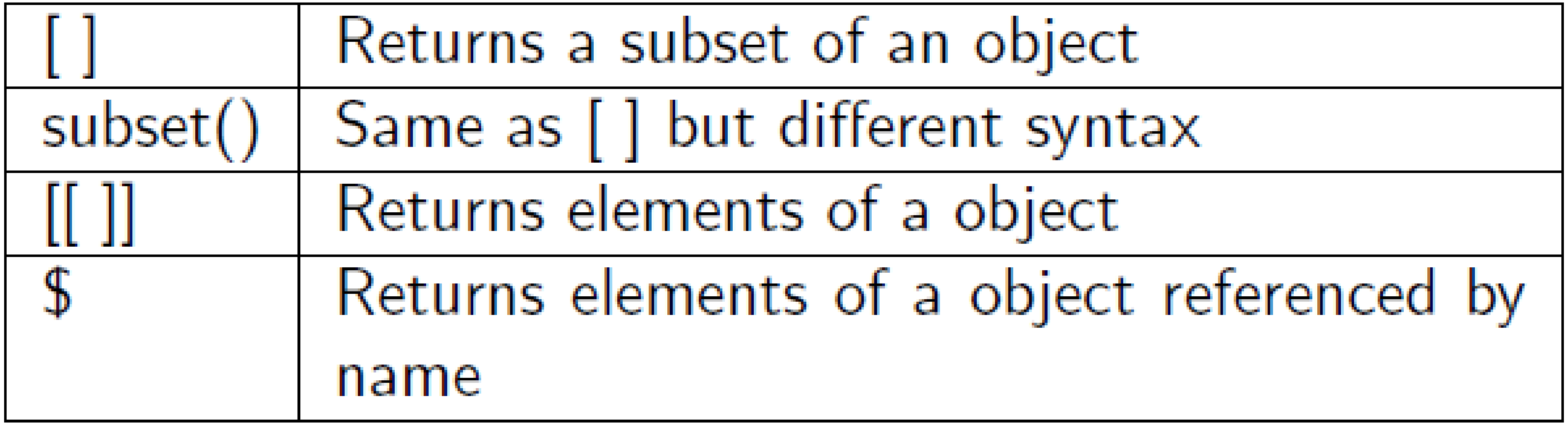
Function	Description
names(dataset)	List names of columns of dataset
str(dataset)	Show structure of dataset
summary(dataset)	Show summary of columns of dataset
size(dataset)	Show size (in bytes) of dataset

Sub-setting data frames

- R provides a number of ways for accessing parts or elements of a data frame.
- New data frames can be generated from these subsections for:
  - Better analysis
  - Machine learning



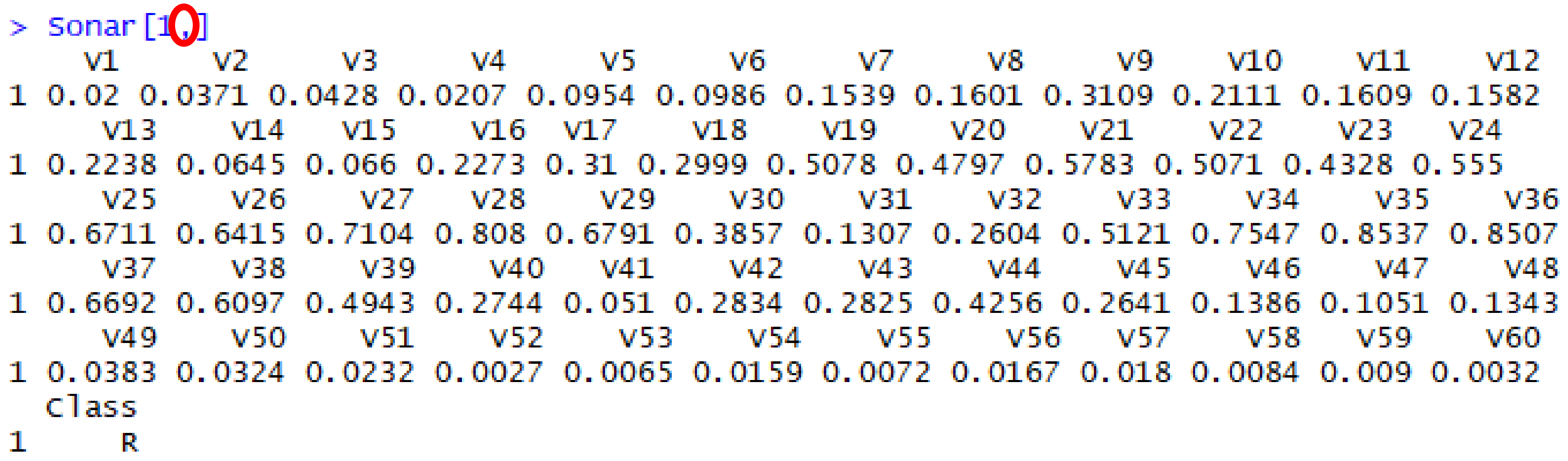
Four main ways of sub-setting



[ ]	Returns a subset of an object
subset()	Same as [ ] but different syntax
[[ ]]	Returns elements of a object
$	Returns elements of a object referenced by name

Selecting an object

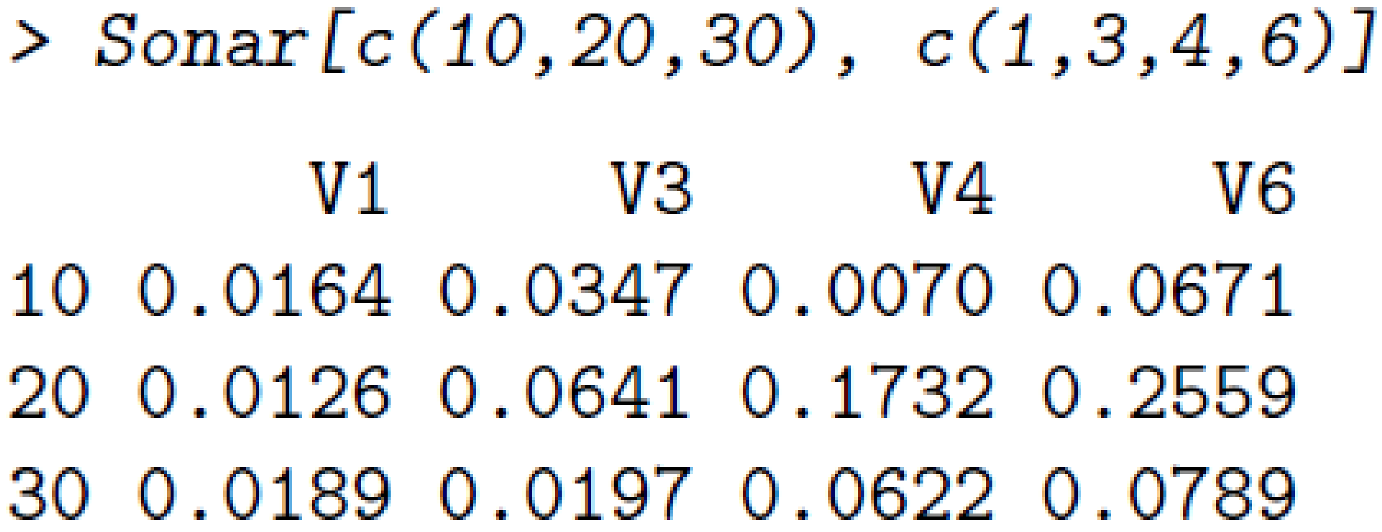
- Select the first reading (i.e. row) of the Sonar dataset.



```
> Sonar[1,]
    V1     V2     V3     V4     V5     V6     V7     V8     V9    V10    V11    V12
1 0.02 0.0371 0.0428 0.0207 0.0954 0.0986 0.1539 0.1601 0.3109 0.2111 0.1609 0.1582
     V13    V14   V15    V16  V17    V18    V19    V20    V21    V22    V23   V24
1 0.2238 0.0645 0.066 0.2273 0.31 0.2999 0.5078 0.4797 0.5783 0.5071 0.4328 0.555
     V25    V26    V27   V28    V29    V30    V31    V32    V33    V34    V35    V36
1 0.6711 0.6415 0.7104 0.808 0.6791 0.3857 0.1307 0.2604 0.5121 0.7547 0.8537 0.8507
     V37    V38    V39    V40   V41    V42    V43    V44    V45    V46    V47    V48
1 0.6692 0.6097 0.4943 0.2744 0.051 0.2834 0.2825 0.4256 0.2641 0.1386 0.1051 0.1343
     V49    V50    V51    V52    V53    V54    V55    V56   V57    V58   V59    V60
1 0.0383 0.0324 0.0232 0.0027 0.0065 0.0159 0.0072 0.0167 0.018 0.0084 0.009 0.0032
  Class
1     R
```

Selecting specific rows/columns

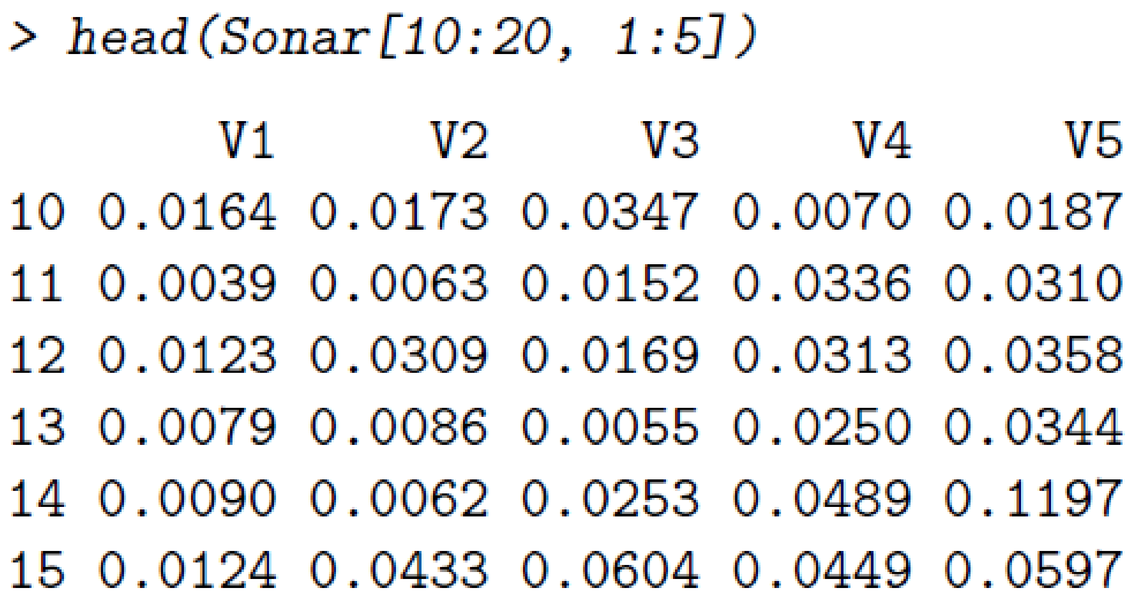
- Select rows 10, 20, 30 and columns 1, 3, 4, 6.



```
> Sonar[c(10,20,30), c(1,3,4,6)]
       V1     V3     V4     V6
10 0.0164 0.0347 0.0070 0.0671
20 0.0126 0.0641 0.1732 0.2559
30 0.0189 0.0197 0.0622 0.0789
```

Sub-setting by range of rows/columns

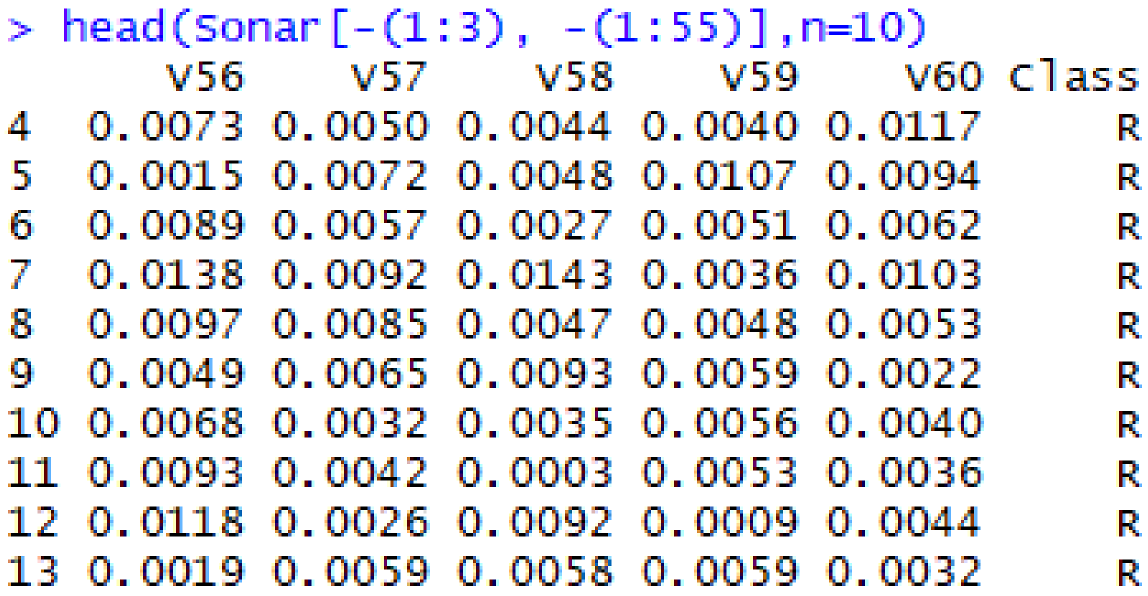
- Select rows 10-30 and columns 1-5.



```
> head(Sonar[10:20, 1:5])
       V1     V2     V3     V4     V5
10 0.0164 0.0173 0.0347 0.0070 0.0187
11 0.0039 0.0063 0.0152 0.0336 0.0310
12 0.0123 0.0309 0.0169 0.0313 0.0358
13 0.0079 0.0086 0.0055 0.0250 0.0344
14 0.0090 0.0062 0.0253 0.0489 0.1197
15 0.0124 0.0433 0.0604 0.0449 0.0597
```

Excluding specific rows/columns

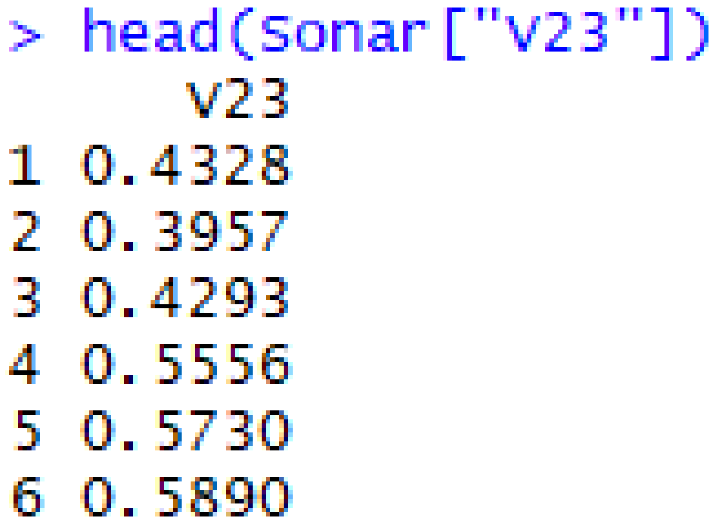
- Select rows 4 to 13 and the last 5 columns.



```
> head(Sonar[-(1:3), -(1:55)],n=10)
      V56    V57    V58    V59    V60 Class
4  0.0073 0.0050 0.0044 0.0040 0.0117     R
5  0.0015 0.0072 0.0048 0.0107 0.0094     R
6  0.0089 0.0057 0.0027 0.0051 0.0062     R
7  0.0138 0.0092 0.0143 0.0036 0.0103     R
8  0.0097 0.0085 0.0047 0.0048 0.0053     R
9  0.0049 0.0065 0.0093 0.0059 0.0022     R
10 0.0068 0.0032 0.0035 0.0056 0.0040     R
11 0.0093 0.0042 0.0003 0.0053 0.0036     R
12 0.0118 0.0026 0.0092 0.0009 0.0044     R
13 0.0019 0.0059 0.0058 0.0059 0.0032     R
```

Sub-setting by column name

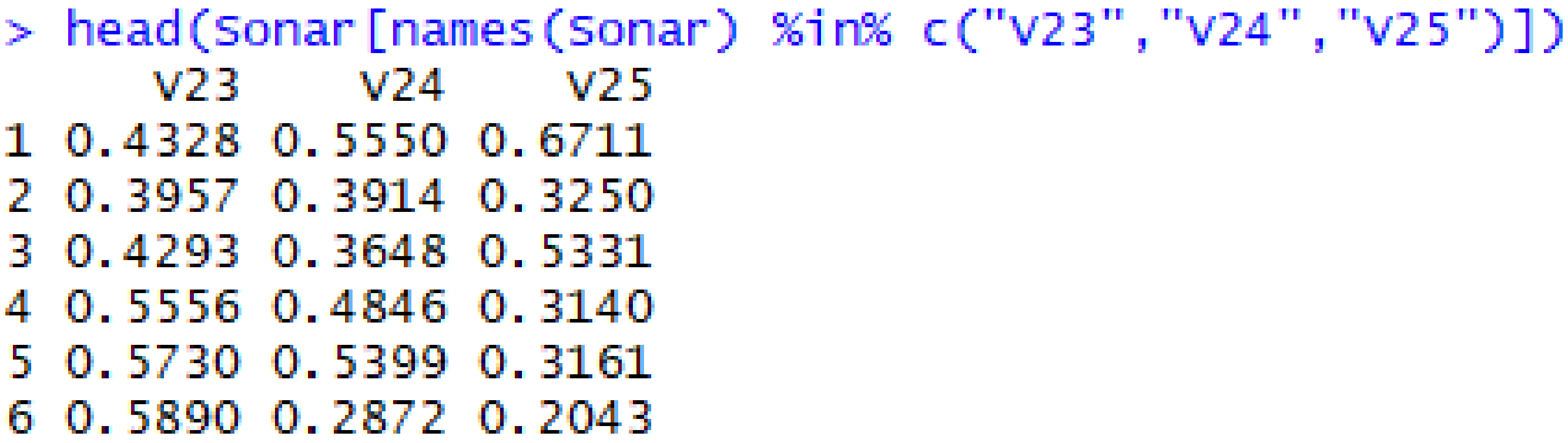
- Select “the first” rows and the column labelled “V23“.



```
> head(sonar["V23"])
     V23
1 0.4328
2 0.3957
3 0.4293
4 0.5556
5 0.5730
6 0.5890
```

Sub-setting by column name

- Select “the first” rows and the columns labelled “V23“, “V24” and “V25”.

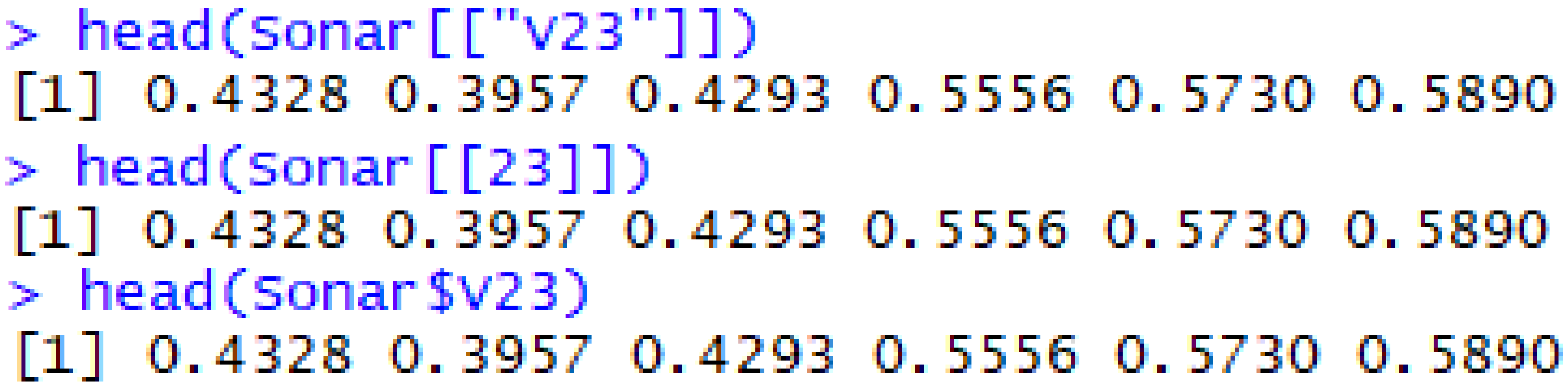


```
> head(Sonar[names(Sonar) %in% c("V23","V24","V25")])
     V23    V24    V25
1 0.4328 0.5550 0.6711
2 0.3957 0.3914 0.3250
3 0.4293 0.3648 0.5331
4 0.5556 0.4846 0.3140
5 0.5730 0.5399 0.3161
6 0.5890 0.2872 0.2043
```

- You can exclude columns by using !names() instead.

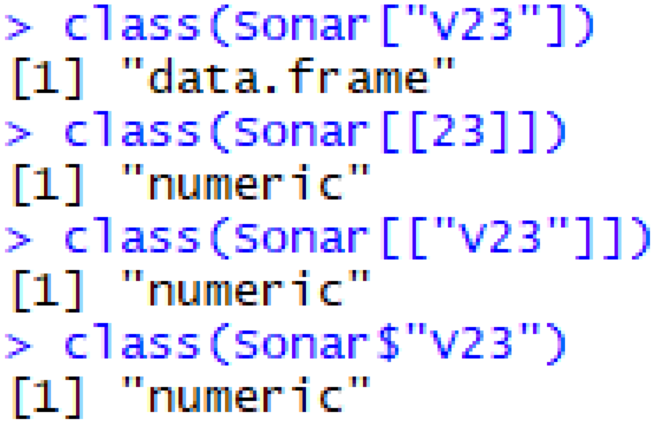
Accessing the content of a column

- Select “the first” columns labelled “V23”.



```
> head(Sonar[["V23"]])
[1] 0.4328 0.3957 0.4293 0.5556 0.5730 0.5890
> head(Sonar[[23]])
[1] 0.4328 0.3957 0.4293 0.5556 0.5730 0.5890
> head(Sonar$V23)
[1] 0.4328 0.3957 0.4293 0.5556 0.5730 0.5890
```

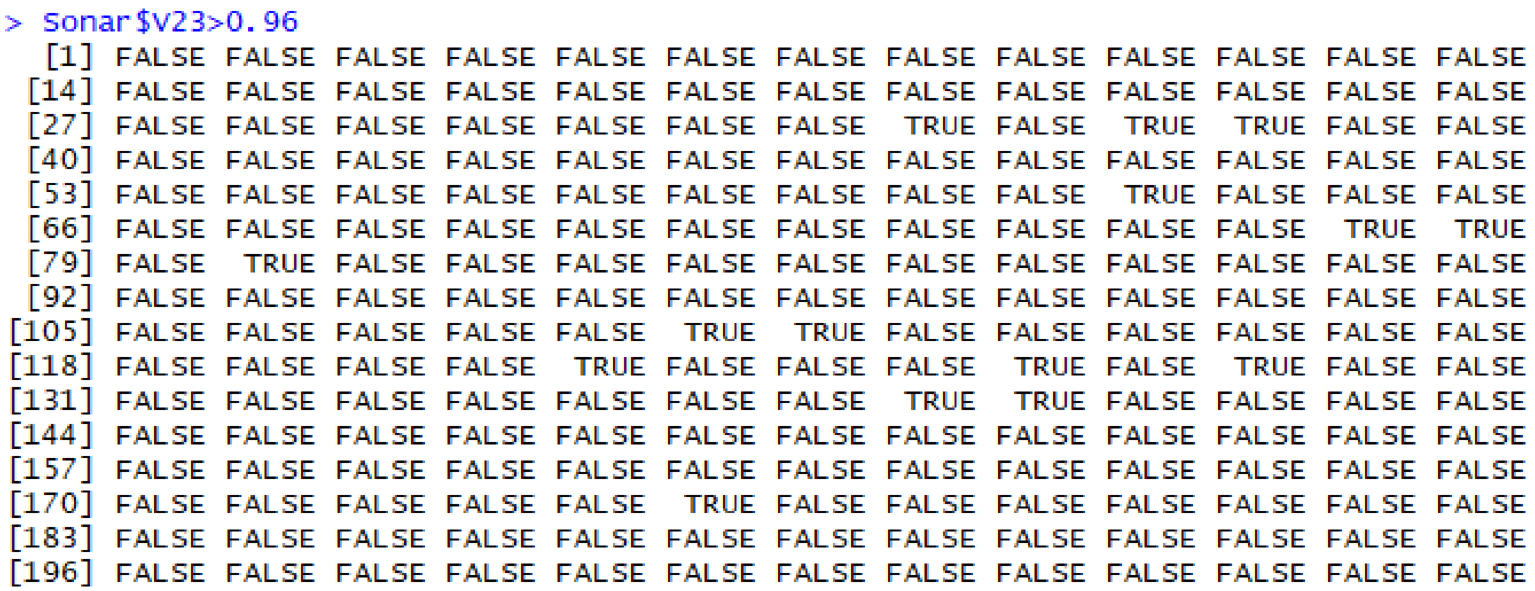
- What is the difference between [], $ and [[]]?



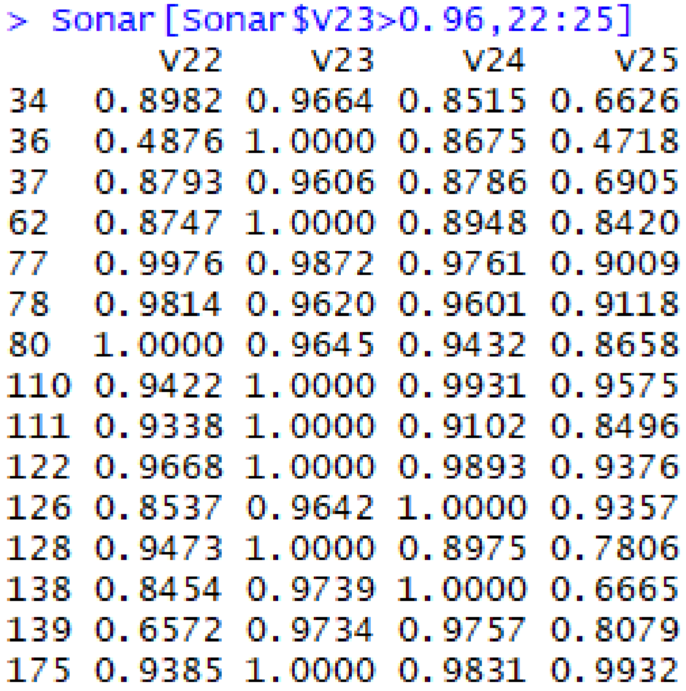
```
> class(Sonar["V23"])
[1] "data.frame"
> class(Sonar[[23]])
[1] "numeric"
> class(Sonar[["V23"]])
[1] "numeric"
> class(Sonar$"V23")
[1] "numeric"
```

Select rows that meet a condition

- Select rows where “V23" is greater than 0.96.



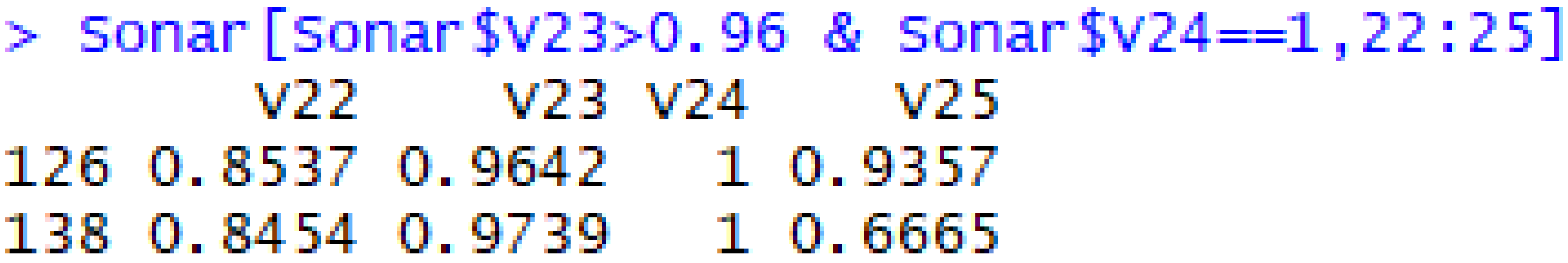
```
> Sonar$V23>0.96
  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [14] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [27] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE  TRUE  TRUE FALSE FALSE
 [40] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [53] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
 [66] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
 [79] FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
 [92] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[105] FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
[118] FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE  TRUE FALSE  TRUE FALSE FALSE
[131] FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE FALSE FALSE FALSE FALSE
[144] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[157] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[170] FALSE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[183] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
[196] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```



```
> Sonar[Sonar$V23>0.96,22:25]
       V22    V23    V24    V25
34  0.8982 0.9664 0.8515 0.6626
36  0.4876 1.0000 0.8675 0.4718
37  0.8793 0.9606 0.8786 0.6905
62  0.8747 1.0000 0.8948 0.8420
77  0.9976 0.9872 0.9761 0.9009
78  0.9814 0.9620 0.9601 0.9118
80  1.0000 0.9645 0.9432 0.8658
110 0.9422 1.0000 0.9931 0.9575
111 0.9338 1.0000 0.9102 0.8496
122 0.9668 1.0000 0.9893 0.9376
126 0.8537 0.9642 1.0000 0.9357
128 0.9473 1.0000 0.8975 0.7806
138 0.8454 0.9739 1.0000 0.6665
139 0.6572 0.9734 0.9757 0.8079
175 0.9385 1.0000 0.9831 0.9932
```

Select rows that meet two conditions

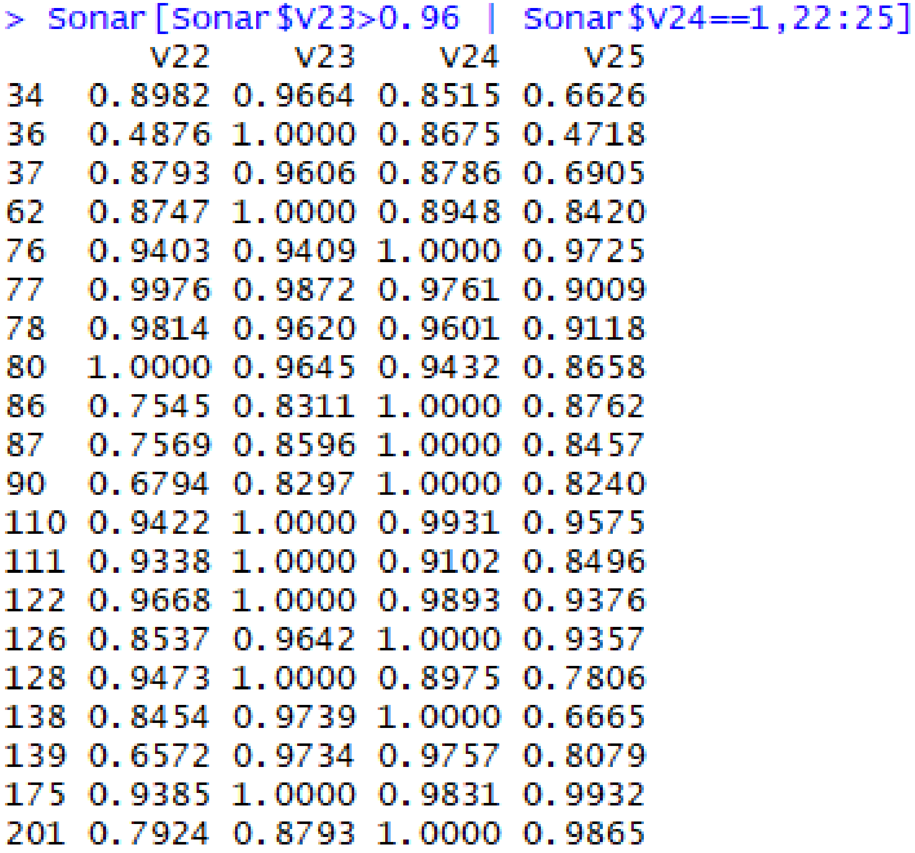
- Select rows where “V23" is greater than 0.96 AND V24 equal to 1.



```
> Sonar[Sonar$V23>0.96 & Sonar$V24==1,22:25]
       V22    V23 V24    V25
126 0.8537 0.9642   1 0.9357
138 0.8454 0.9739   1 0.6665
```

Select rows that meet two conditions

- Select rows where “V23" is greater than 0.96 OR V24 equal to 1.



```
> Sonar[Sonar$V23>0.96 | Sonar$V24==1,22:25]
       V22    V23    V24    V25
34  0.8982 0.9664 0.8515 0.6626
36  0.4876 1.0000 0.8675 0.4718
37  0.8793 0.9606 0.8786 0.6905
62  0.8747 1.0000 0.8948 0.8420
76  0.9403 0.9409 1.0000 0.9725
77  0.9976 0.9872 0.9761 0.9009
78  0.9814 0.9620 0.9601 0.9118
80  1.0000 0.9645 0.9432 0.8658
86  0.7545 0.8311 1.0000 0.8762
87  0.7569 0.8596 1.0000 0.8457
90  0.6794 0.8297 1.0000 0.8240
110 0.9422 1.0000 0.9931 0.9575
111 0.9338 1.0000 0.9102 0.8496
122 0.9668 1.0000 0.9893 0.9376
126 0.8537 0.9642 1.0000 0.9357
128 0.9473 1.0000 0.8975 0.7806
138 0.8454 0.9739 1.0000 0.6665
139 0.6572 0.9734 0.9757 0.8079
175 0.9385 1.0000 0.9831 0.9932
201 0.7924 0.8793 1.0000 0.9865
```

Lab Activity Session 1

- Install Rstudio!

- Open RStudio and load “GSM0008_S1_Lab.R”

- Follow the script to practice what we have learnt so far.