

Genowis
2019-09-23

Contents

1	5
2 R	7
2.1	7
2.2	8
2.3	9
2.4	9
2.5	10
2.6	11
2.7	11
2.8	11
2.9 base	11
3	15
3.1	15
3.2 json	15
3.3 html	15
4	17
4.1	17
4.2	17
4.3	17
4.4	17
4.5	17

4		<i>CONTENTS</i>
5		19
5.1	19
5.2	19
6		21
6.1	21
6.2	21
7		23
7.1	markdown	23
7.2	Rmarkdown	23
8		25

Chapter 1

flow R github Stack Over-

• R

•

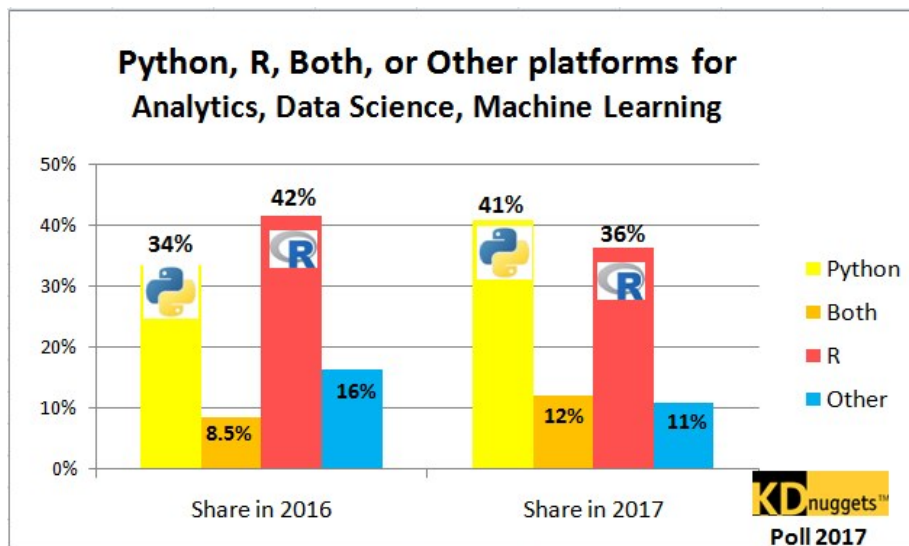
Chapter 2

R

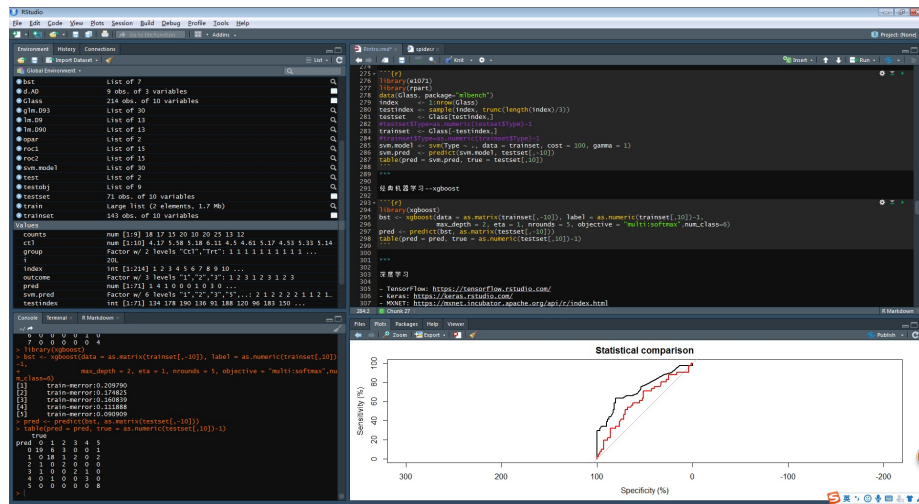
2.1

R is a language and environment for statistical computing and graphics. It is a GNU project which is similar to the S language and environment which was developed at Bell Laboratories (formerly AT&T, now Lucent Technologies) by John Chambers and colleagues.

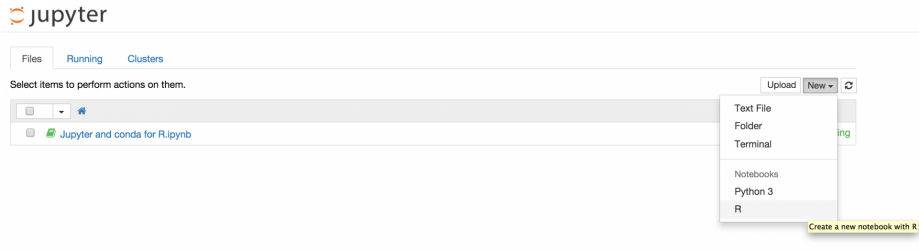
- 1980 S
-



- Rstudio




```
conda install -c r r-essentials
```



2.3

- -
 -
 -
 -
- character: "a", "1", "apple", "@"
- numeric: 1, 3.14, 100, 2e10
- integer: 1, 2, 500
- factor: 1 2
- logical: TRUE, FALSE
- date: 2018-01-19, 19/1/2018

2.4

-
-
-
-
-

```
- vector: c(1,2,3) c("a","b","c"), 1:10

- list: list(1,2,3) list(a="A",b="B",c="C",d=1)

- / matrix/array: matrix(c(1,2,3,4),ncol=2)

- data.frame: data.frame(ID=c(1,2,3), =c("A","B","A"))
```

2.5

- for

```
for ( i in 1:20) {

  if (i %% 2==0){
    print(paste(i," ",sep=""))
  }
  else {
    next
  }
}
```

```
## [1] "2 "
## [1] "4 "
## [1] "6 "
## [1] "8 "
## [1] "10 "
## [1] "12 "
## [1] "14 "
## [1] "16 "
## [1] "18 "
## [1] "20 "
```

- if... else
- while...
- repeat
- break
- next

2.6

```
- mean(), get_IHC()
```

```
- : body( )
```

```
- get_IHC(x, y) x y IHC
```

2.7

```
=function( ){  
}
```

```
func=function(x,y){  
  return(x/(y+1))  
}
```

```
func(1,1)
```

```
## [1] 0.5
```

2.8

2.9 base

2.9.1

• + * /

```
1:10+2
```

```
## [1] 3 4 5 6 7 8 9 10 11 12
```

```
9/(1:3)-0.5
```

```
## [1] 8.5 4.0 2.5
```

2.9.2

- `mean()` `max()` `min()` `quantile()` `sum()` `summary()`

```
summary(rnorm(100))
```

```
##      Min.   1st Qu.   Median     Mean   3rd Qu.    Max.
## -2.518890 -0.486208  0.191744  0.008009  0.783987  1.628505
```

2.9.3

- `&`, `|`, `!`

```
!(2>1 | 2>3)
```

```
## [1] FALSE
```

2.9.4

- `paste()`, `grep()` `grepl()`, `strsplit()` `strsub()`

```
tmp=paste(c(1:10),"163.com",sep="@")
unlist(strsplit(tmp,split="@"))[seq(1,20,2)]
```

```
## [1] "1" "2" "3" "4" "5" "6" "7" "8" "9" "10"
```

2.9.5

- `subset()`, `merge()`, `dim()`, `names()`

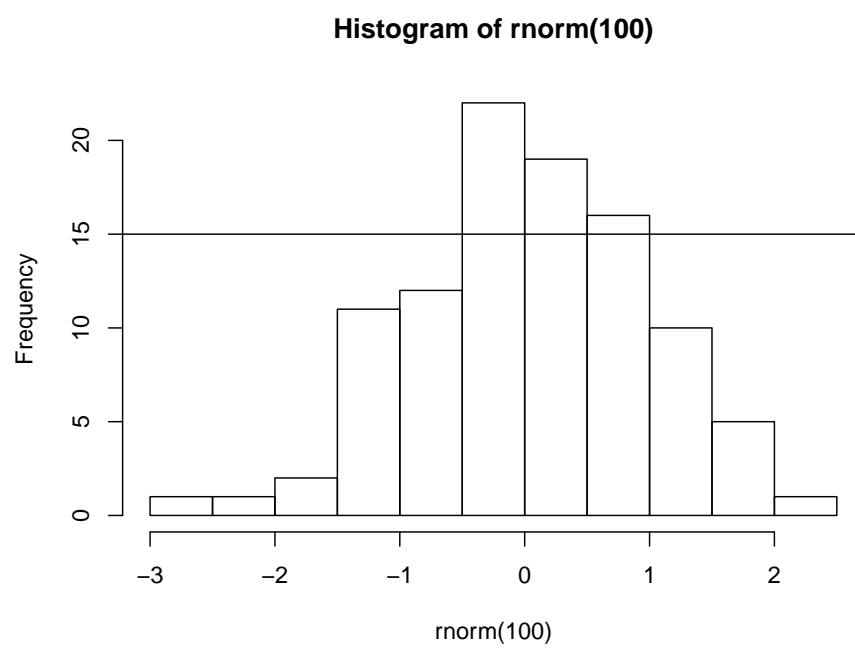
```
df=data.frame(x=1:10,y=rep("a",10),stringsAsFactors = F)
dim(subset(df,x>5))
```

```
## [1] 5 2
```

2.9.6

- `plot()` `adline()` `hist()`

```
hist(rnorm(100))  
abline(a=15,b=0)
```



Chapter 3

3.1

3.2 json

3.3 html

Here is a review of existing methods.

Chapter 4

4.1

4.1.1

4.1.2

4.1.3

4.2

4.2.1

4.2.2 χ^2

4.3

4.3.1 P

4.3.2

4.4

4.5

Chapter 5

5.1

5.1.1

5.1.2 /

5.1.3 json

5.1.4 xml

5.2

5.2.1 data.table

5.2.2 plyr/dplyr

5.2.3 magrittr

Chapter 6

6.1

6.2

6.2.1 `ggplot2`

6.2.2 `plotly`

6.2.3 `echarts4r`

Chapter 7

7.1 markdown

7.2 Rmarkdown

Chapter 8

Base R Cheat Sheet

Getting Help

Accessing the help files

?mean
Get help of a particular function.

help.search('weighted mean')
Search the help files for a word or phrase.

help(package = 'dplyr')
Find help for a package.

More about an object

str(iris)
Get a summary of an object's structure.

class(iris)
Find the class an object belongs to.

Using Packages

install.packages('dplyr')
Download and install a package from CRAN.

library(dplyr)
Load the package into the session, making all its functions available to use.

dplyr::select
Use a particular function from a package.

data(iris)
Load a built-in dataset into the environment.

Working Directory

getwd()
Find the current working directory (where inputs are found and outputs are sent).

setwd('C://file/path')
Change the current working directory.

Use projects in RStudio to set the working directory to the folder you are working in.

Vectors

Creating Vectors

c(2, 4, 6)	2 4 6	Join elements into a vector
2:6	2 3 4 5 6	An integer sequence
seq(2, 3, by=0.5)	2.0 2.5 3.0	A complex sequence
rep(1:2, times=3)	1 2 1 2 1 2	Repeat a vector
rep(1:2, each=3)	1 1 1 2 2 2	Repeat elements of a vector

Vector Functions

sort(x) Return x sorted. table(x) See counts of values.	rev(x) Return x reversed. unique(x) See unique values.
--	---

Selecting Vector Elements

By Position

x[4]	The fourth element.
x[-4]	All but the fourth.
x[2:4]	Elements two to four.
x[-(2:4)]	All elements except two to four.
x[c(1, 5)]	Elements one and five.

By Value

x[x == 10]	Elements which are equal to 10.
x[x < 0]	All elements less than zero.
x[x %in% c(1, 2, 5)]	Elements in the set 1, 2, 5.

Named Vectors

x['apple']	Element with name 'apple'.
-------------------	----------------------------

Programming

For Loop

```
for (variable in sequence){  
  Do something  
}
```

Example

```
for (i in 1:4){  
  j <- i + 10  
  print(j)  
}
```

While Loop

```
while (condition){  
  Do something  
}
```

Example

```
while (i < 5){  
  print(i)  
  i <- i + 1  
}
```

If Statements

```
if (condition){  
  Do something  
} else {  
  Do something different  
}
```

Example

```
if (i > 3){  
  print('Yes')  
} else {  
  print('No')  
}
```

Functions

```
function_name <- function(var){  
  Do something  
  return(new_variable)  
}
```

Example

```
square <- function(x){  
  squared <- x*x  
  return(squared)  
}
```

Reading and Writing Data

Also see the **readr** package.

Input	Output	Description
df <- read.table('file.txt')	write.table(df, 'file.txt')	Read and write a delimited text file.
df <- read.csv('file.csv')	write.csv(df, 'file.csv')	Read and write a comma separated value file. This is a special case of read table/ write table.
load('file.Rdata')	save(df, file = 'file.Rdata')	Read and write an R data file, a file type special for R.

Conditions	a == b	Are equal	a > b	Greater than	a >= b	Greater than or equal to	is.na(a)	Is missing
	a != b	Not equal	a < b	Less than	a <= b	Less than or equal to	is.null(a)	Is null

RStudio® is a trademark of RStudio, Inc. • CC-BY-Mhairi McNeill • mhairimcneill@gmail.com • 884-468-1212 • studio.com

Learn more at web page or vignette • package version • Updated: 3/15

Types

Converting between common data types in R. Can always go from a higher value in the table to a lower value.

as.logical	TRUE, FALSE, TRUE	Boolean values (TRUE or FALSE)
as.numeric	1, 0, 1	Integers or floating point numbers.
as.character	'1', '0', '1'	Character strings. Generally preferred to factors.
as.factor	'1', '0', '1', levels: '1', '0'	Character strings with preset levels. Needed for some statistical models.

Matrices

`m <- matrix(x, nrow = 3, ncol = 3)`
Create a matrix from x.

`m[2,]` - Select a row

`m[, 1]` - Select a column

`m[2, 3]` - Select an element

`t(m)`
Transpose
`m %*% n`
Matrix Multiplication
`solve(m, n)`
Find x in: $m \cdot x = n$

Lists

`l <- list(x = 1:5, y = c('a', 'b'))`
A list is a collection of elements which can be of different types.

`l[[2]]` - Second element of l

`l[1]` - New list with only the first element.

`l$x` - Element named x.

`l[["y"]]` - New list with only element named y.

Strings

Also see the **stringr** package.

`paste(x, y, sep = ' ')`
Join multiple vectors together.

`paste(x, collapse = ' ')`
Join elements of a vector together.

`grep(pattern, x)`
Find regular expression matches in x.

`gsub(pattern, replace, x)`
Replace matches in x with a string.

`toupper(x)`
Convert to uppercase.

`tolower(x)`
Convert to lowercase.

`nchar(x)`
Number of characters in a string.

Factors

factor(x)
Turn a vector into a factor. Can set the levels of the factor and the order.

cut(x, breaks = 4)
Turn a numeric vector into a factor by 'cutting' into sections.

Statistics

`lm(y ~ x, data=df)`
Linear model.

`glm(y ~ x, data=df)`
Generalised linear model.

summary
Get more detailed information out a model.

`t.test(x, y)`
Perform a t-test for difference between means.

`prop.test`
Test for a difference between proportions.

`pairwise.t.test`
Perform a t-test for paired data.

sdv
Analysis of variance.

Distributions

	Random Variates	Density Function	Cumulative Distribution	Quantile
Normal	<code>rnorm</code>	<code>dnorm</code>	<code>pnorm</code>	<code>qnorm</code>
Poisson	<code>rpois</code>	<code>dpois</code>	<code>ppois</code>	<code>qpois</code>
Binomial	<code>rbinom</code>	<code>dbinom</code>	<code>pbinom</code>	<code>qbinom</code>
Uniform	<code>runif</code>	<code>dunif</code>	<code>punif</code>	<code>qunif</code>

Plotting

Also see the **ggplot2** package.

`plot(x)`
Values of x in order.

`plot(x, y)`
Values of x against y.

`hist(x)`
Histogram of x.

Dates

See the **lubridate** package.

Variable Assignment

```
> a <- 'apple'
> a
[1] 'apple'
```

The Environment

`ls()`
List all variables in the environment.

`rm(x)`
Remove x from the environment.

`rm(list = ls())`
Remove all variables from the environment.

You can use the environment panel in RStudio to browse variables in your environment.

Data Frames

`df <- data.frame(x = 1:3, y = c('a', 'b', 'c'))`
A special case of a list where all elements are the same length.

List subsetting

`df$x`

`df[[2]]`

Understanding a data frame

`View(df)`
See the full data frame.

`head(df)`
See the first 6 rows.

Matrix subsetting

`df[, 2]`

`df[2, 1]`

`df[2, 2]`

`nrow(df)`
Number of rows.

`ncol(df)`
Number of columns.

`din(df)`
Number of columns and rows.

`cbind` - Bind columns.

`rbind` - Bind rows.