Introduction to R

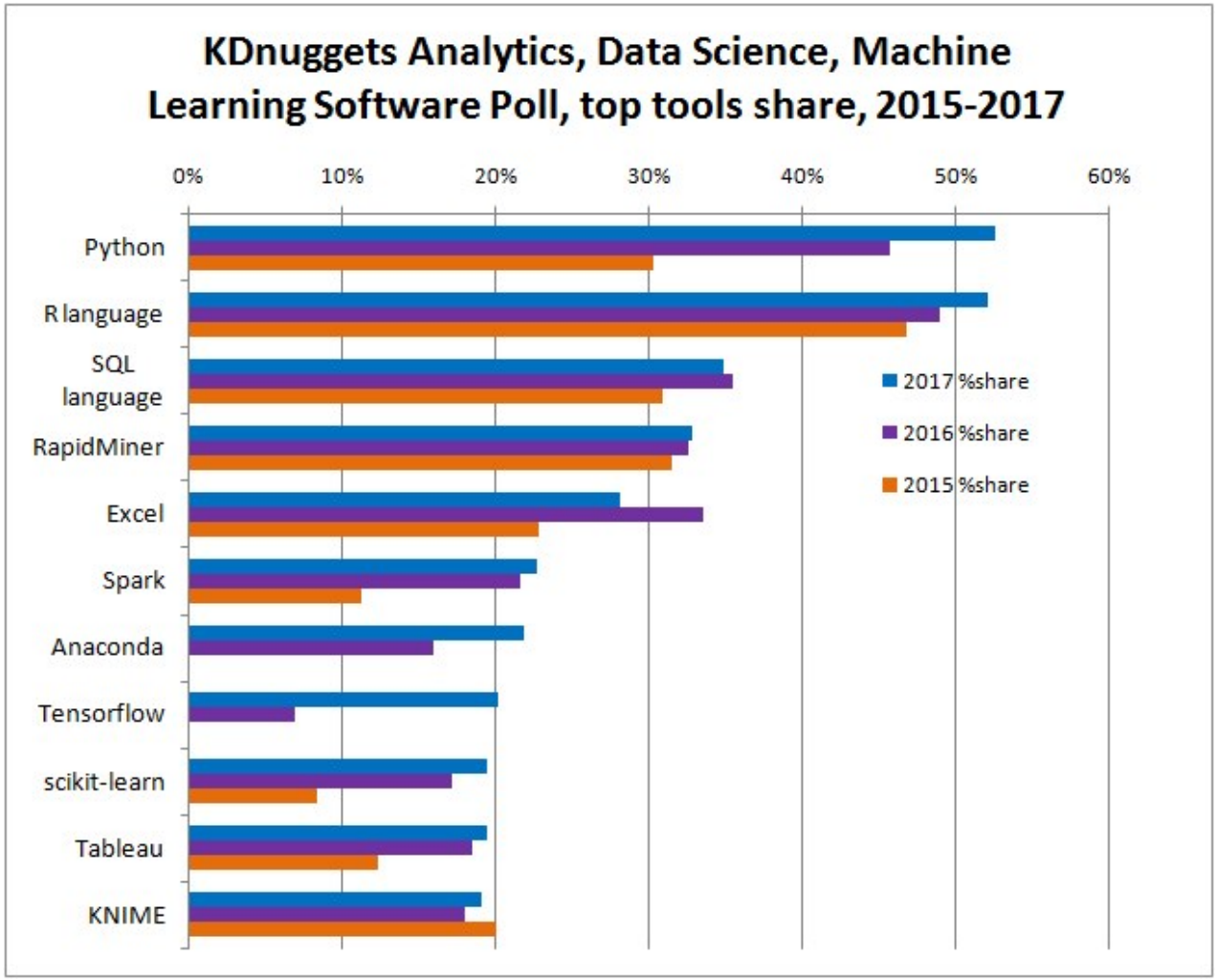
S. Trahasch, S. Niro

October 8, 2017

Goals

- Know basic concepts of R
- Be able to use essential data structures and commands
- Create simple data visualizations
- Be able to install and use R packages

Introduction (kdnuggets.com Survey 2015-2017)



Introduction (kdnuggets.com Survey 2015-2017)

Table 1: Top Analytics/Data Science Tools in 2017 KDnuggets Poll

Tool	2017 % Usage	% change 2017 vs 2016	% alone
Python	52.6%	15%	0.2%
R language	52.1%	6.4%	3.3%
SQL language	34.9%	-1.8%	0%
RapidMiner	32.8%	0.7%	13.6%
Excel	28.1%	-16%	0.1%
Spark	22.7%	5.3%	0.2%
Anaconda	21.8%	37%	0.8%
Tensorflow	20.2%	195%	0%
scikit-learn	19.5%	13%	0%
Tableau	19.4%	5.0%	0.4%
KNIME	19.1%	6.3%	2.4%

Introduction

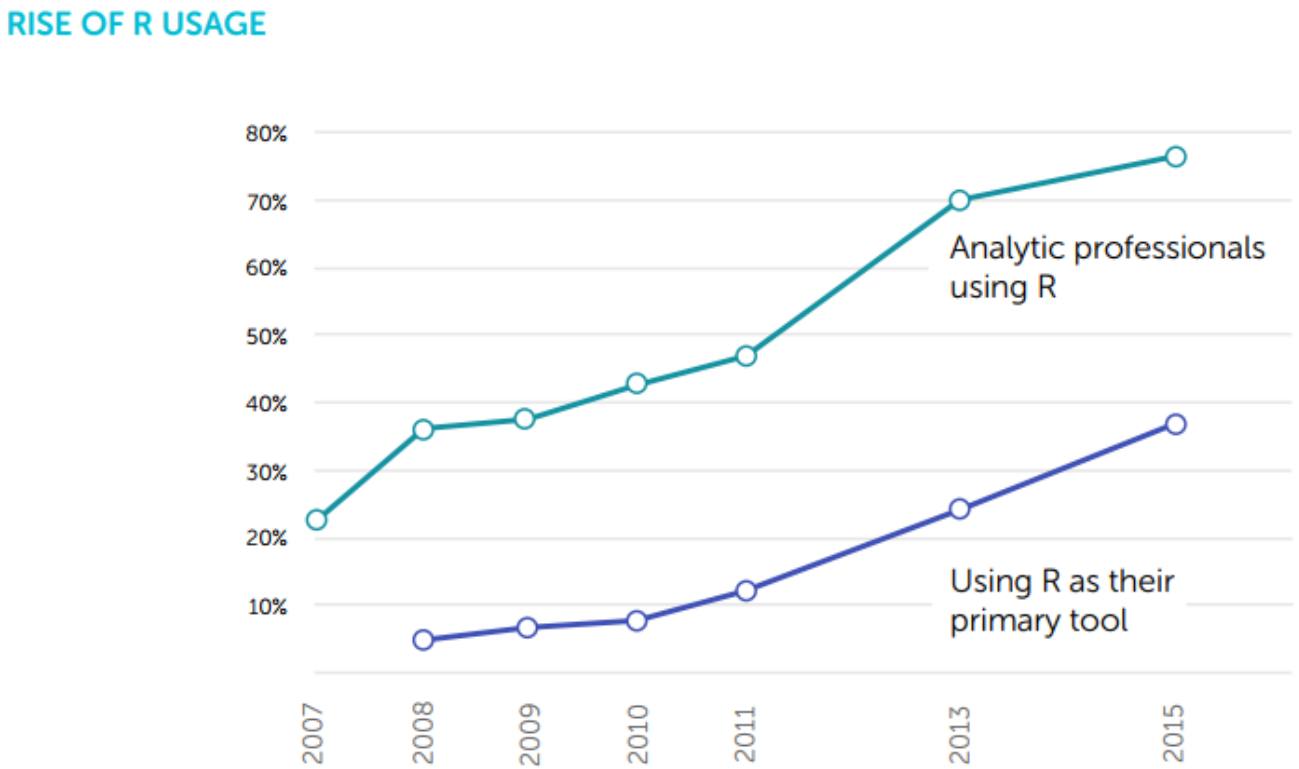


Figure: Rexer Analytics Data Miner Survey 2015

Introduction

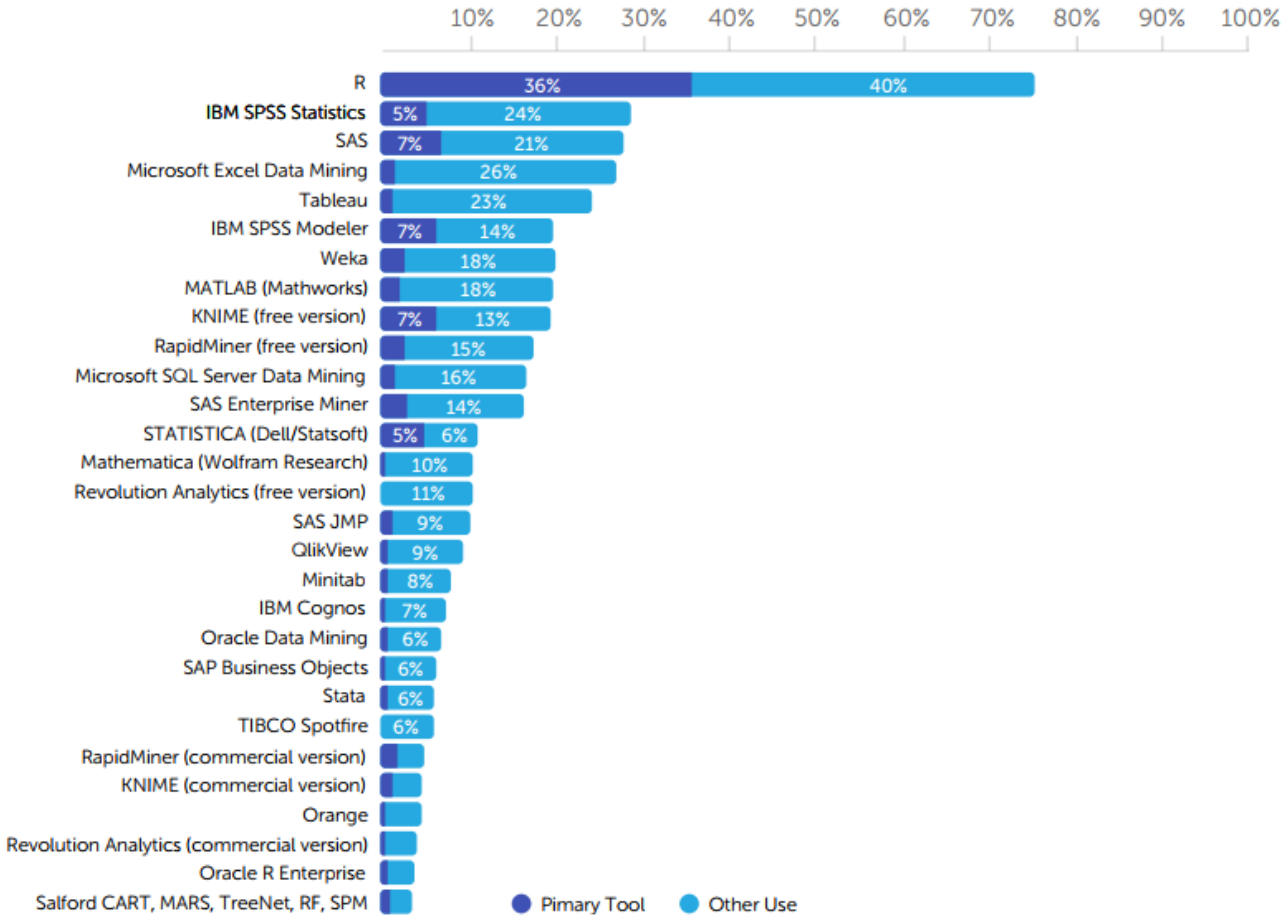


Figure: Rexer Analytics Data Miner Survey 2015

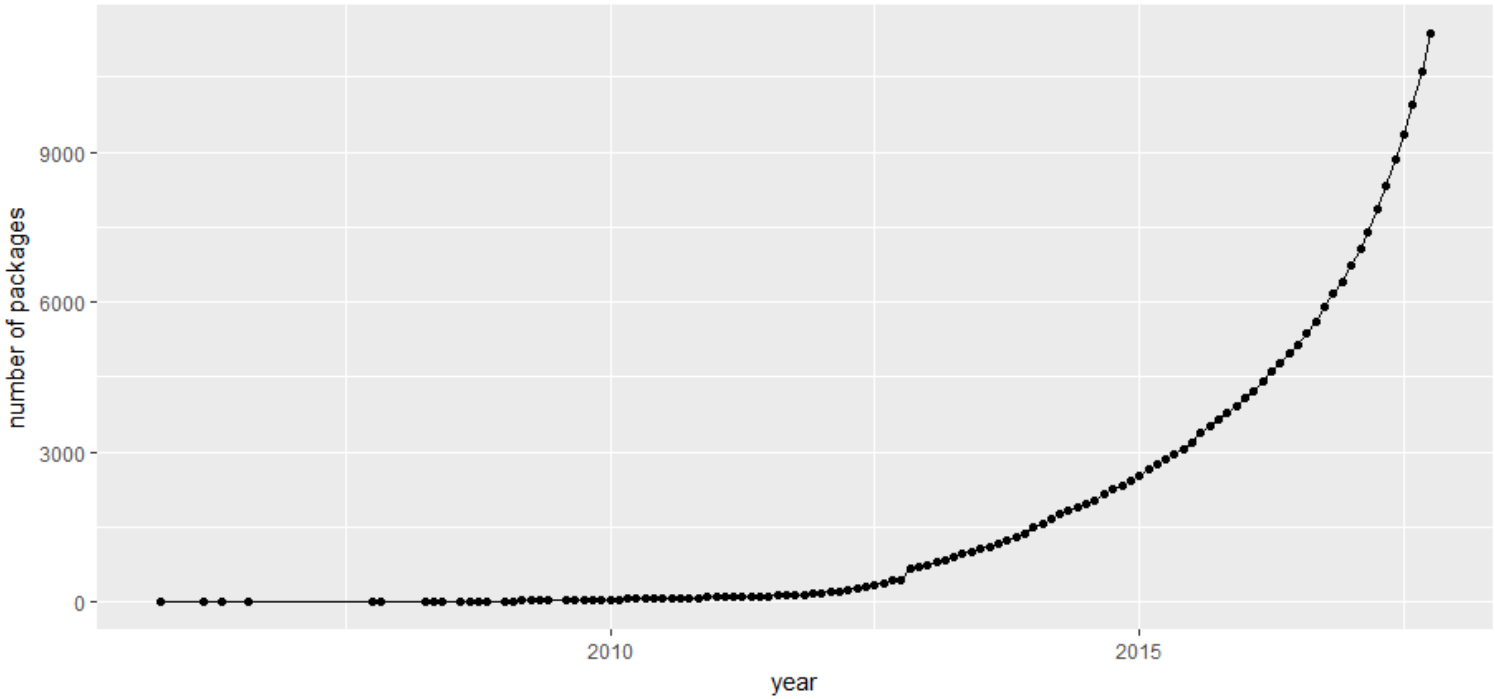
History: S, S-Plus, R

- Becker, R. A. und Chambers, J. M. publish **S** in 1984, a language for **data analysis (statistics)** and **graphics**
- **S-PLUS** is a commercial implementation of S
- **R** is an open source implementation of S developed in 1992 by **R**oss Ihaka and **R**obert Gentleman

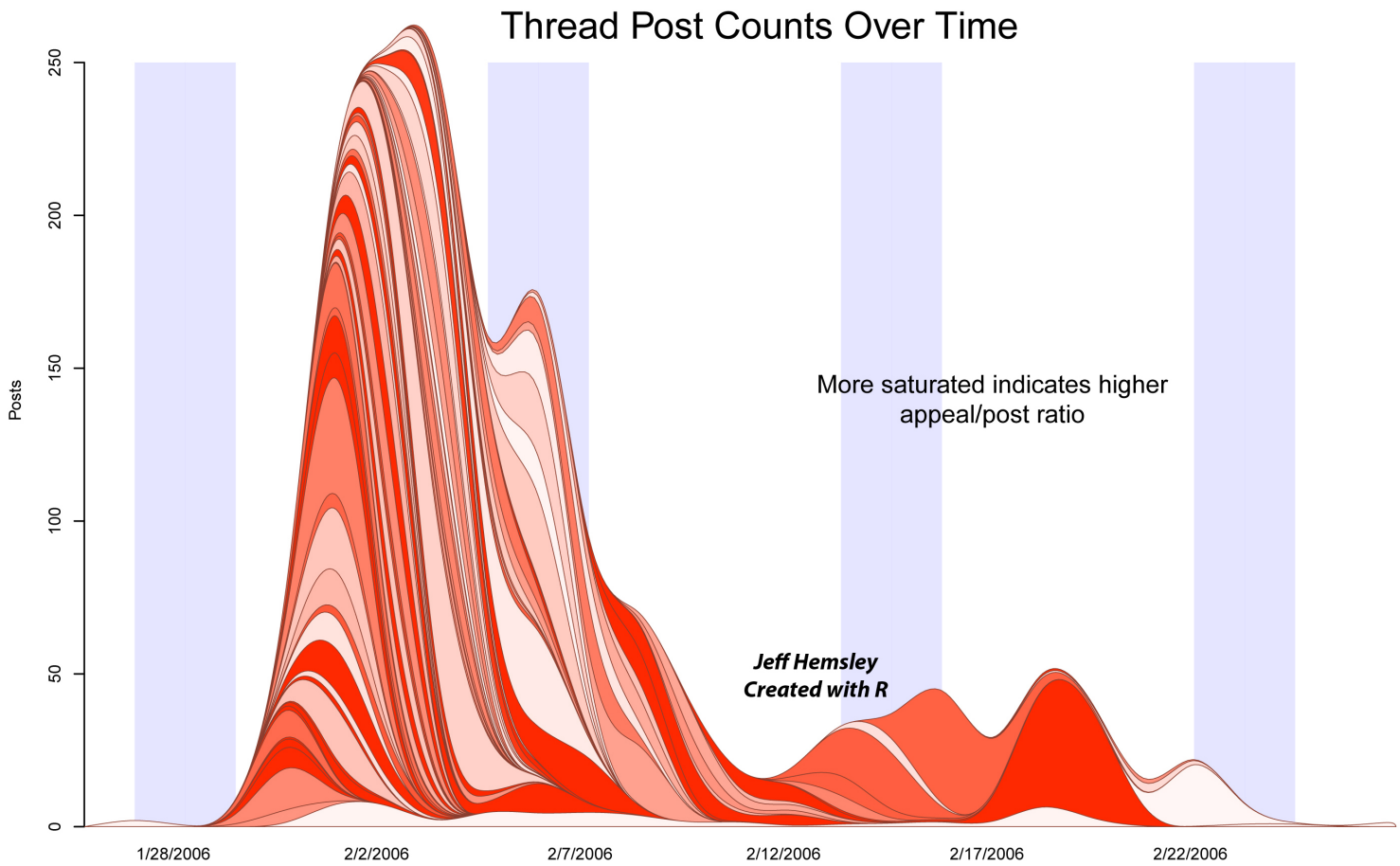
Advantages of R

- Domain specific language for data analysis and visualization
- Open Source, no license costs (GNU GPL)
- Huge active community
- Available for all platforms: Windows, Linux, Solaris, ..
- Huge number of packages ($>$ 10000). New methods are often implemented and provided as (free) R-Packages
- Faster than S-Plus
- Bindings/Interfaces for several programming languages available (Java, Python, ...)
- Integration of R into other data analysis software (Rapidminer, SAP HANA, SPSS, SAS, ...)

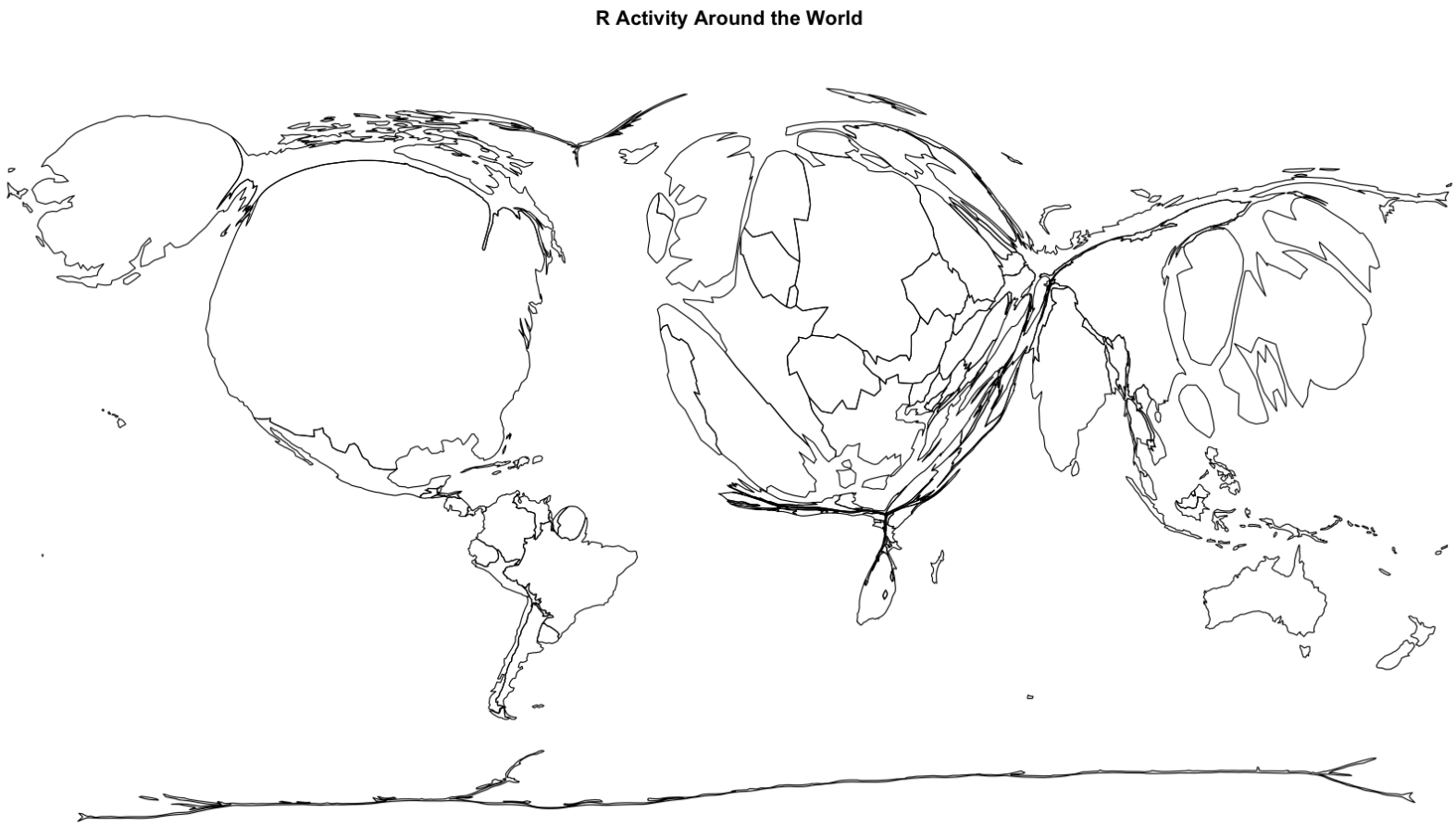
Graphics with R - Examples (Number of R-Packages)



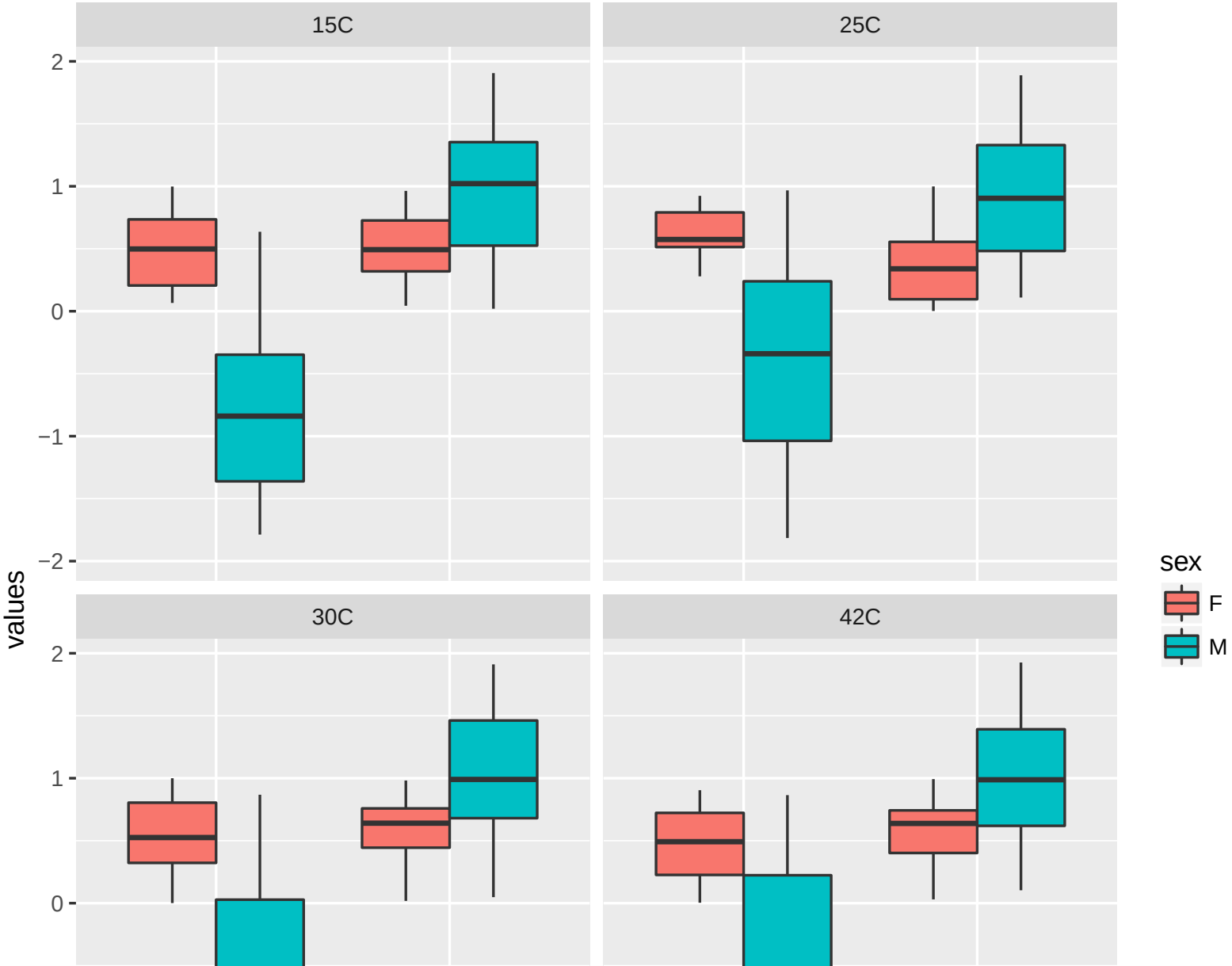
Graphics with R - Examples



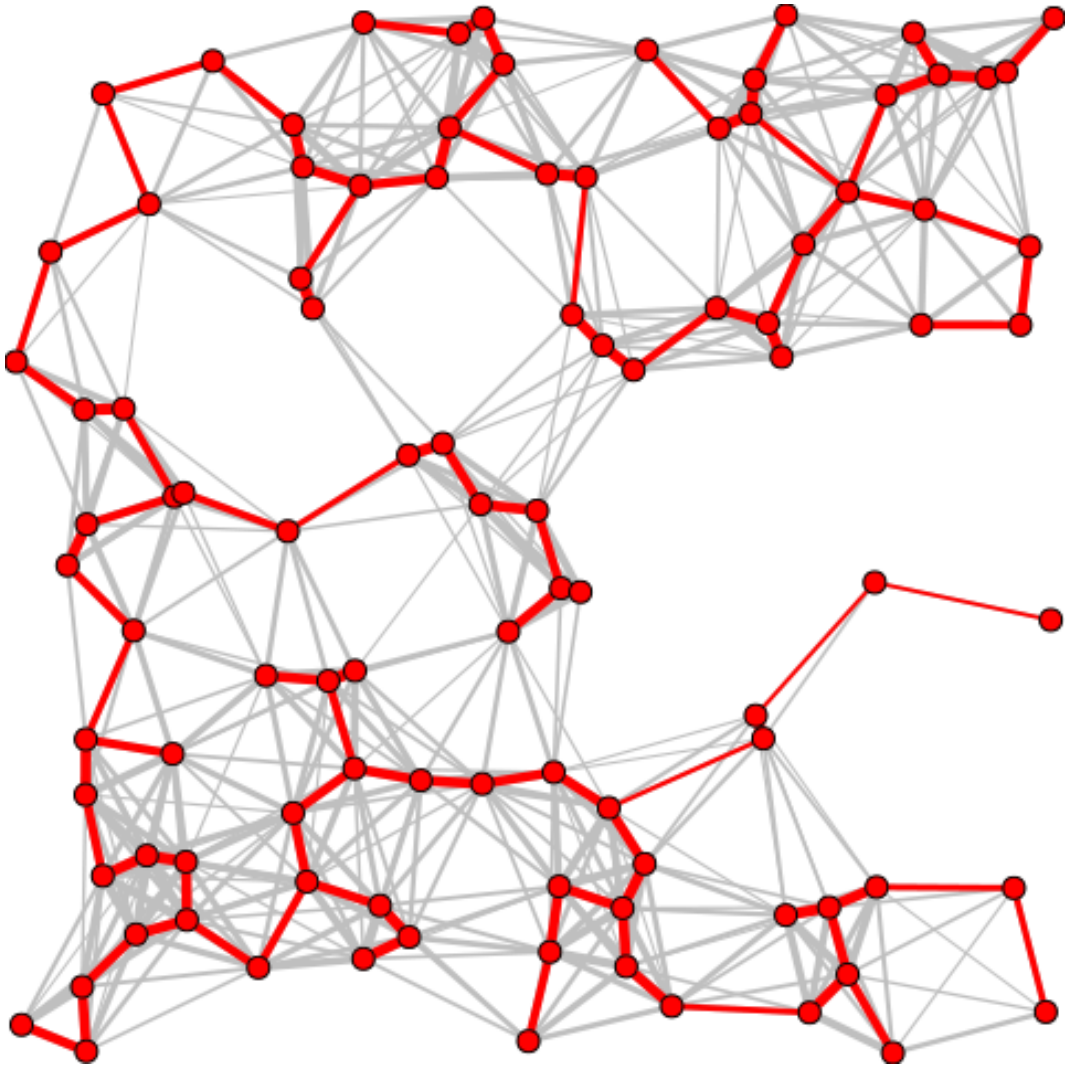
Graphics with R - Examples



Graphics with R - Examples

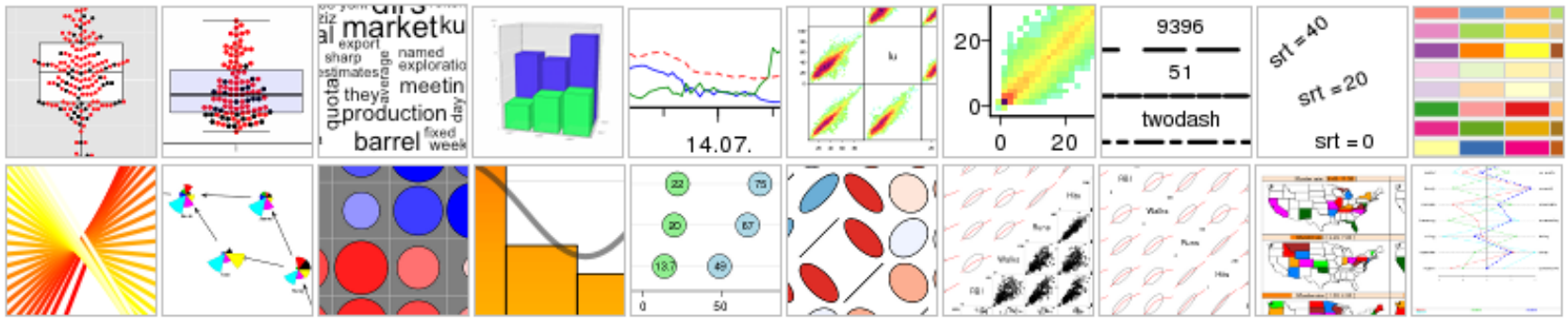


Graphics with R - Examples

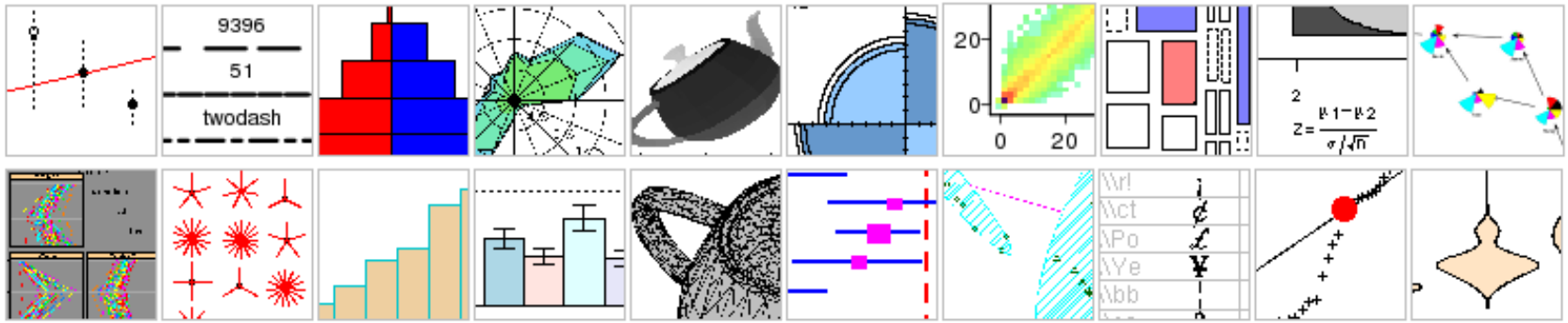


Graphics with R - Examples

» Last entries ...

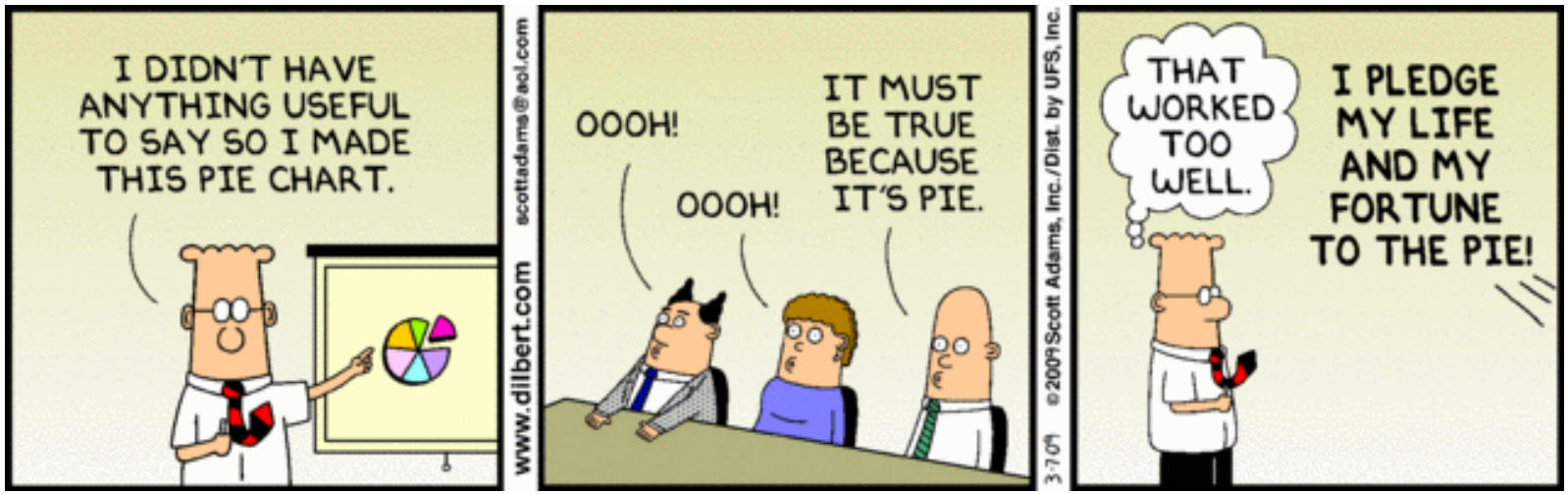


» Random entries



More: http://www.sr.bham.ac.uk/ ajrs/R/r-gallery.html
http://addictedtor.free.fr/graphiques/

Grpahics (Dilbert)



Disadvantage of R

- No graphical user interface
- Flat learning curve compared to other data analysis software
- Quality of the packages depends on the number of users
- Error messages sometimes hard to interpret

R as Calculator


```
3.5 + 1.5

[1] 5
```

```
x <- 6 * (1/3)   # Assignment
                 # (recommended)
x

[1] 2
```

```
x = 2^2          # Assignment
print(x)

[1] 4
```

Operator	
+	Addition
−	Subtraction
*	Multiplication
/	Division
^	Power
%%	Modulo

More math. functions:
sin(x), sqrt(x), exp(x), ...

Vectors

Ordered set of elements of the same type

```
a <- c(4, 5, 6) # combine

a

[1] 4 5 6
```

```
length(a) # length of a

[1] 3
```

```
a[2]        # second element of a

[1] 5
```

Vectors: Arithmetic


```
a <- seq(from = 1, to = 3, by = 1) # equals c(1,2,3)
b <- 9:7 # equals c(9, 8, 7)
a

[1] 1 2 3

b

[1] 9 8 7
```

manual

```
  c <- c(0,0,0)
  for(i in 1:length(a))
  {
c[i] <- a[i] + b[i]
  }
  c
```

vectorized (recommended)

```
c <- a + b
c

[1] 10 10 10
```

Vectors: Recycling


```
a <- 1:6
a

[1] 1 2 3 4 5 6

a + c(1,2) # ???

[1] 2 4 4 6 6 8
```

$$\begin{pmatrix}1\\2\\3\\4\\5\\6\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix} \longrightarrow^{recycling} \begin{pmatrix}1\\2\\3\\4\\5\\6\end{pmatrix} + \begin{pmatrix}1\\2\\1\\2\\1\\2\end{pmatrix} = \begin{pmatrix}2\\4\\4\\6\\6\\8\end{pmatrix}$$

Vectors and functions


```
a <- 1:4
```

Functions are applied to every element of the vector. The result is a new vector.

```
sqrt(a) # square root
```

```
[1] 1.000000 1.414214 1.732051 2.000000
```

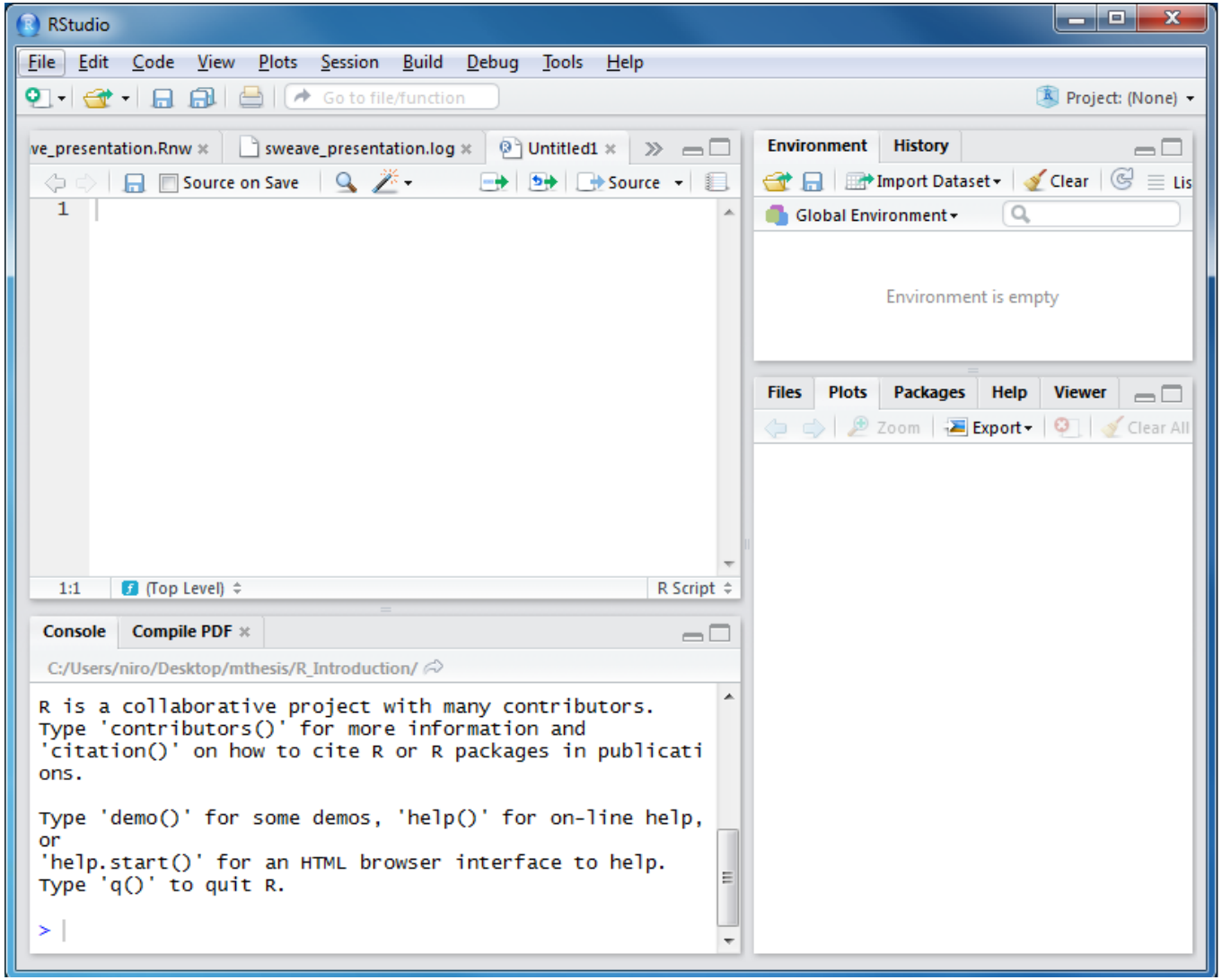
```
max(a^2) # biggest element
```

```
[1] 16
```

```
sum(a^2) # sum of all elements
```

```
[1] 30
```

R-Studio



Exercise I

- Create a vector x of integers in the interval $[-10; 10]$
- How many elements are in x (length function)?
- Which value has the 10th and 22th element?
- Calculate $y(x) = -x^2 + 20$
- What is the smalles/biggest value of $y(x)$ (min/max)?
- Plot the function $y(x)$ using plot(x, y)
- Add the argument

```
type = "l"
```

to the plot function call. How does the plot change for

```
type = "b"
```

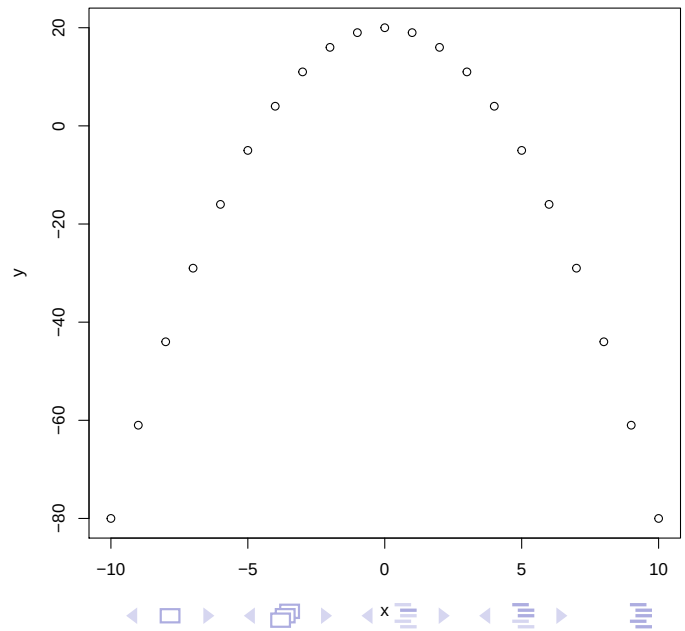
```
type = "p"
```

- Optional: Calculate $\bar{y} = \frac{1}{N} \cdot \sum_{i=1}^{N}(y_i)$

Exercise I


```
x <- -10:10
length(x)

[1] 21

x[10]

[1] -1

y <- -x^2 + 20
min(y)

[1] -80

max(y)

[1] 20
```

```
plot(x,y)
1/length(y) * sum(y)
mean(y)

[1] -16.66667
[1] -16.66667
```

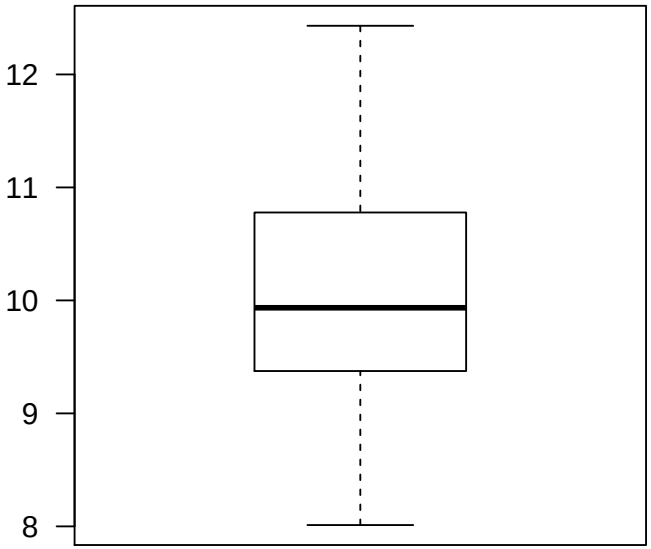


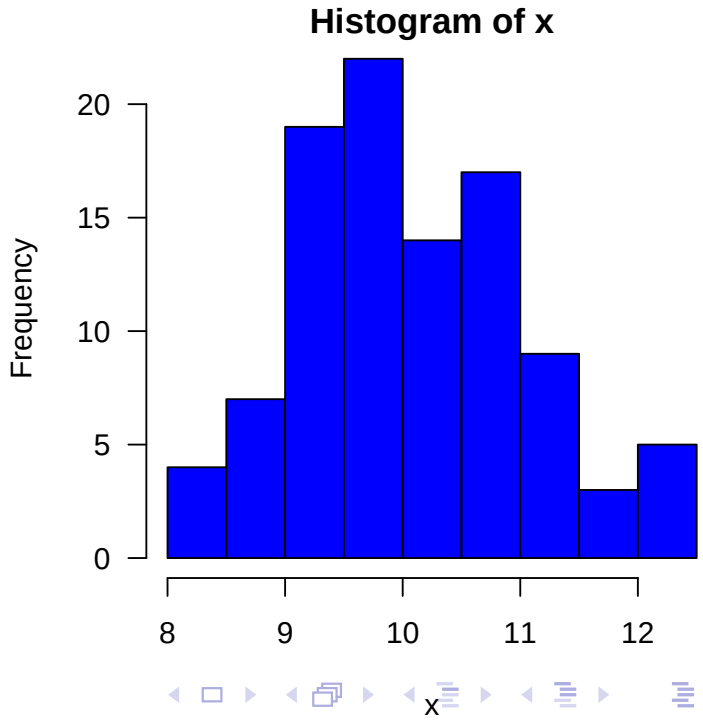
Exercise II

- Create $n = 100$ normal distributed random values with a mean of 10 and a standard deviation of 1 using the **rnorm** function (Get help using **?rnorm** or help(rnorm))
- Calculate the mean (**mean**) and the standard deviation (**sd**) of the generated values
- Create a boxplot (**boxplot**) and a histogram (**hist**)
- Repeat everything with $n = 10000$. What changes?
- Optional: Repeat the experiment with uniform distributed random values (**runif**)

Exercise II (Solution)

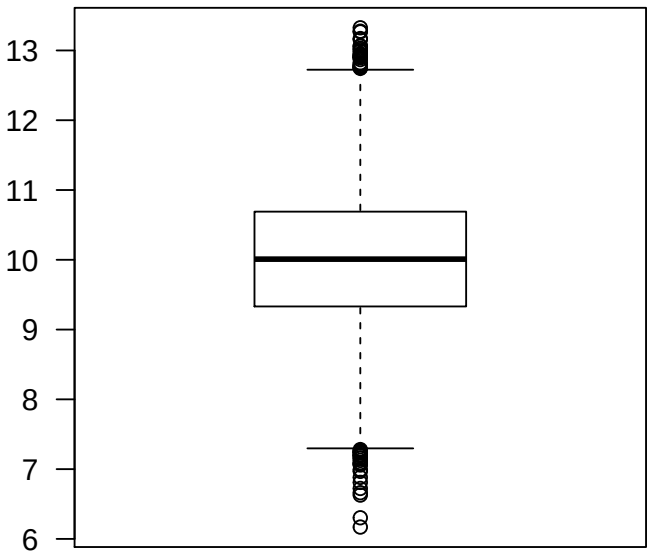

```
x <- rnorm(100, mean=10, sd=1)
mean(x); sd(x)
par(las =1, mar=c(4,4,1,.1))
boxplot(x); hist(x, col="blue")

[1] 10.05844
[1] 0.9868056
```

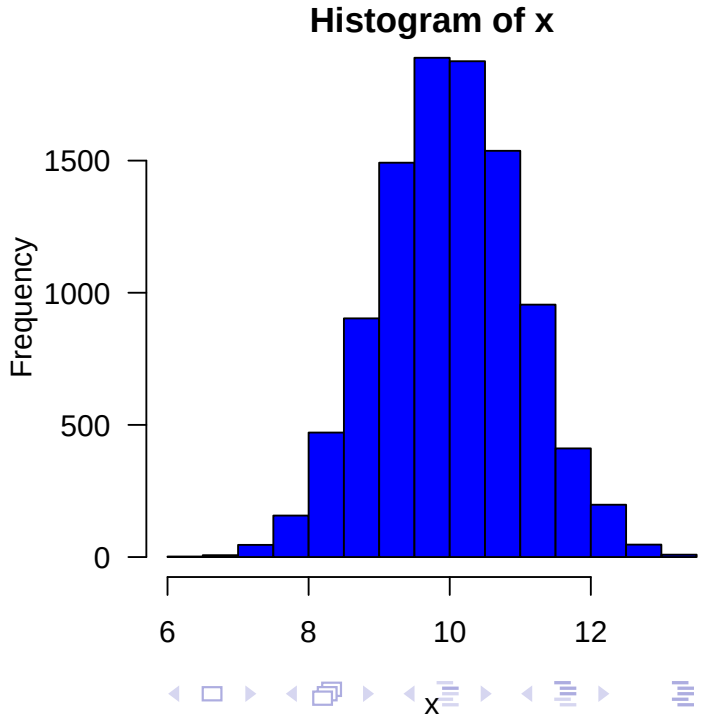


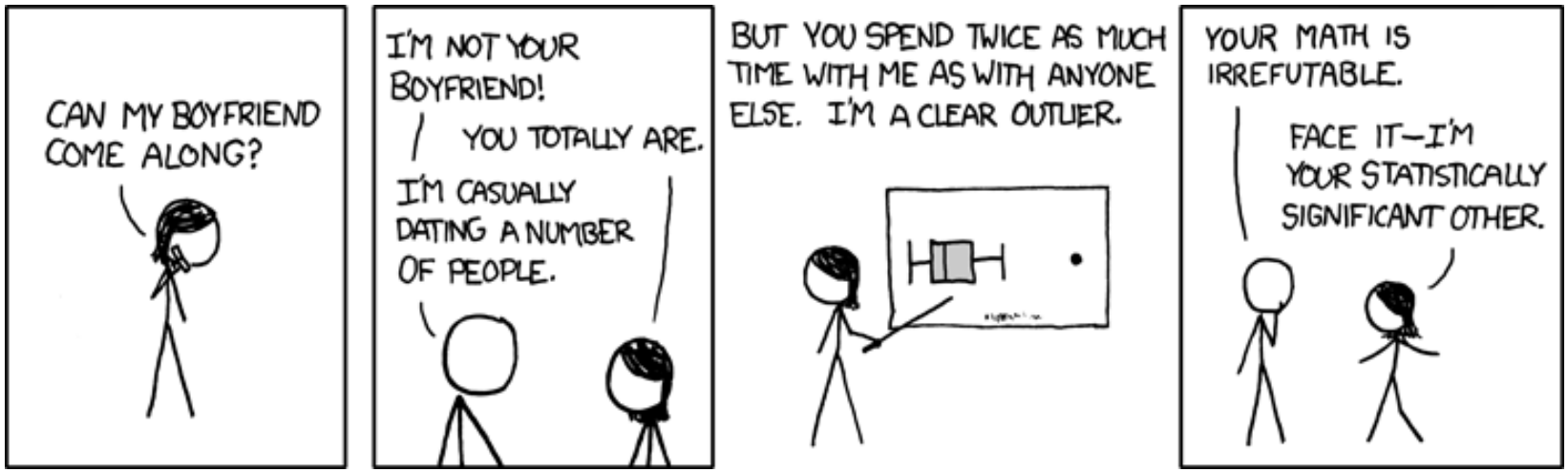


Exercise II (Solution)


```
x <- rnorm(10000, mean=10, sd=1)
mean(x); sd(x)
par(las =1, mar=c(4,4,1,.1))
boxplot(x); hist(x, col="blue")

[1] 10.00832
[1] 1.003438
```







Other types (*mode*)

Logical (Boolean value)

```
married <- c(TRUE, FALSE, T, F, T)
print(married)

[1]  TRUE FALSE  TRUE FALSE  TRUE
```

Character (String)

```
name <- c("Max", "Fritz")
print(name)

[1] "Max"   "Fritz"
```

Factor (Categorical values):

```
sex <- factor(c("m","m","w","m","w","w"))
print(sex)

[1] m m w m w w
Levels: m w
```

Logical and relational operators


```
10 == (2 + 8)
```

```
[1] TRUE
```

```
(10 %% 3 != 0) && (4 < 5)
```

```
[1] TRUE
```

```
!FALSE
```

```
[1] TRUE
```

Operator	Meaning
==	Equality
!=	Inequality
>	Greater than
<=	Less than or equal
Logical:	
!	NOT
&&	AND
||	OR

```
c(5,8,10) > 5
```

```
[1] FALSE  TRUE  TRUE
```

Conditional Execution


```
if(2+2==5)
{
  print("2+2 equals 5")
}else
{
  print("2+2 not equals 5")
}

[1] "2+2 not equals 5"
```

short:

```
ifelse(2+2==5, "equal", "not equal")

[1] "not equal"

ifelse(1:10==5, "equal", "not equal") # vectorized

 [1] "not equal" "not equal" "not equal" "not equal"
 [5] "equal"     "not equal" "not equal" "not equal"
```

Conditional selection


```
a <- c(2,4,6,8,10)
```

```
a[1:3]  # Index based selection

[1] 2 4 6
```

```
a[c(T,T,T,F,F)] # Conditional selection

[1] 2 4 6
```

```
a[a < 7] # Conditional selection

[1] 2 4 6
```

Data Frame

- List of vectors of the same length (columns) that are named
- Important data structure
- Example

Name	Division	Shoesize
Dennis	APC	42
Ralf	SIB	43
Stefan	IS	42
Susanne	APC	39
Swen	SIB	42
Werner	SIB	43

Table: Participants as CSV-File

- Two Indices: df[Row(s) , Column(s)]

Data Frame


```
df <- read.csv("participants.csv", sep=";")
df

     Name Gruppe Shoesize
1  Dennis    APC       42
2    Ralf    SIB       43
3  Stefan     IS       42
4 Susanne    APC       39
5    Swen    SIB       42
6  Werner    SIB       43

names(df)  # Column names

[1] "Name"     "Gruppe"   "Shoesize"

dim(df)    # Dimensions (rows, columns)

[1] 6 3
```

Data Frame: Access the contents


```
df[1, ] # first row, all columns
```

```
    Name Gruppe Shoesize
1 Dennis    APC       42
```

```
df[1,3] # first row, third column
```

```
[1] 42
```

```
df[,2] # all rows, second column
```

```
[1] APC SIB IS  APC SIB SIB
Levels: APC IS SIB
```

```
df[,"Shoesize"] # Column by name I
```

```
[1] 42 43 42 39 42 43
```

Data Frame: Access the contents II


```
df$Name # Column by name II

[1] Dennis  Ralf    Stefan  Susanne Swen    Werner
Levels: Dennis Ralf Stefan Susanne Swen Werner

df[df$Shoesize < 41,]

     Name Gruppe Shoesize
4 Susanne    APC       39

df[df$Group == "APC", "Name"]

factor(0)
Levels: Dennis Ralf Stefan Susanne Swen Werner
```

```
str(df)

'data.frame': 6 obs. of  3 variables:
 $ Name    : Factor w/ 6 levels "Dennis","Ralf",..: 1 2 3 4 5 6
 $ Gruppe  : Factor w/ 3 levels "APC","IS","SIB": 1 3 2 1 3 3
 $ Shoesize: int  42 43 42 39 42 43
```

```
summary(df)

      Name   Gruppe     Shoesize
 Dennis :1   APC:2   Min.   :39.00
 Ralf   :1   IS :1   1st Qu.:42.00
 Stefan :1   SIB:3   Median :42.00
 Susanne:1           Mean   :41.83
 Swen   :1           3rd Qu.:42.75
 Werner :1           Max.   :43.00
```

Exercise III (Part 1): Packages and Data Frames

- Install the two packages **rpart** and **rpart.plot** (*Tools/Install Packages ...* in RStudio or via Console with *install.packages("PACKAGE_NAME")*)
- Load both Packages with *library(PACKAGE_NAME)*
- Load the example data set **ptitanic** with *data(ptitanic)*
- Use the functions **summary** and **str** on the ptitanic data frame
- Create a scatterplot **plot** (Color the data points with the argument

```
col = ifelse(ptitanic$survived==" survived", "green", "red")
```

Exercise III (Part 1 Solution)

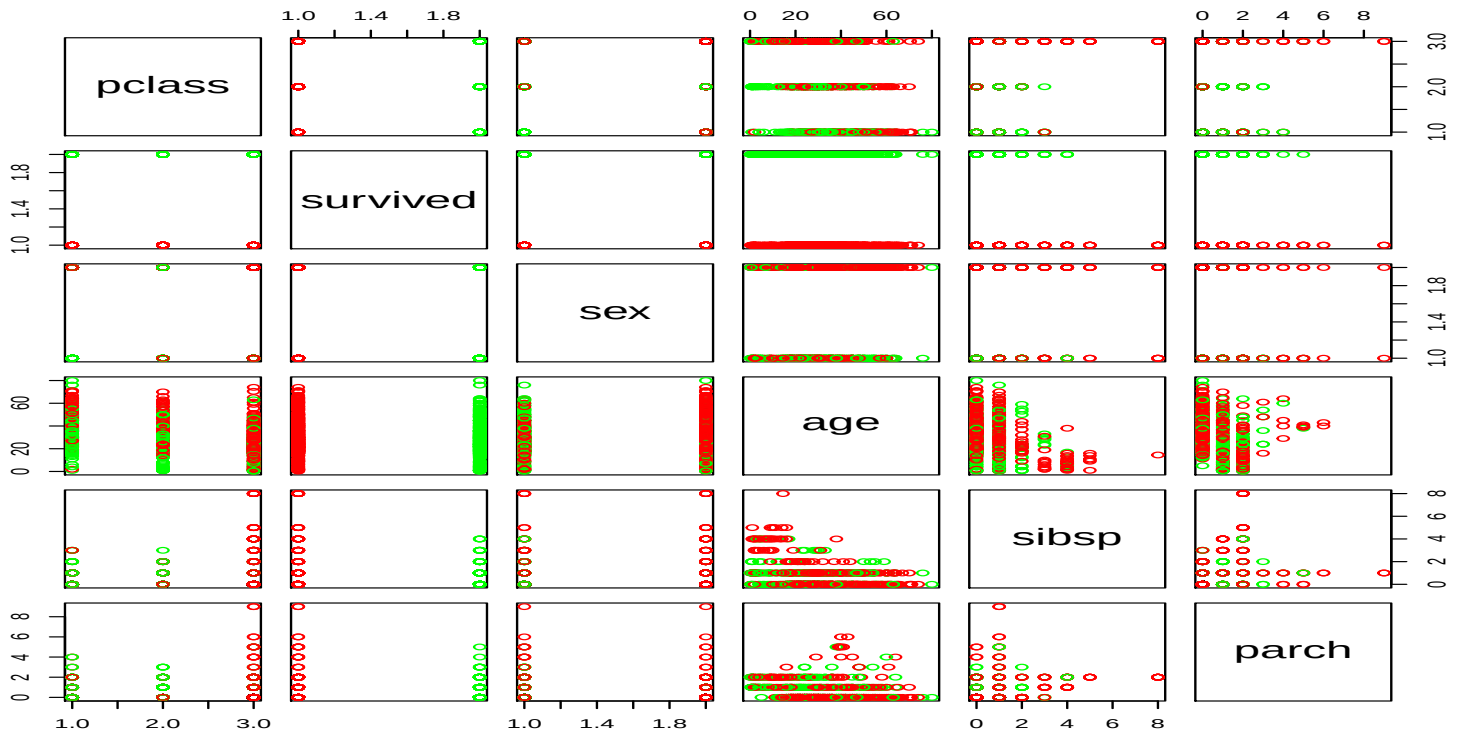

```
library(rpart);library(rpart.plot);data(ptitanic); options(width=
str(ptitanic)

'data.frame': 1309 obs. of  6 variables:
 $ pclass  : Factor w/ 3 levels "1st","2nd","3rd": 1 1 1 1 1 1 1
 $ survived: Factor w/ 2 levels "died","survived": 2 2 1 1 1 2 2
 $ sex     : Factor w/ 2 levels "female","male": 1 2 1 2 1 2 1 2
 $ age     :Class 'labelled'  atomic [1:1309] 29 0.917 2 30 25 ..
  .. ..- attr(*, "units")= chr "Year"
  .. ..- attr(*, "label")= chr "Age"
 $ sibsp   :Class 'labelled'  atomic [1:1309] 0 1 1 1 1 0 1 0 2 0
  .. ..- attr(*, "label")= chr "Number of Siblings/Spouses Aboard
 $ parch   :Class 'labelled'  atomic [1:1309] 0 2 2 2 2 0 0 0 0 0
  .. ..- attr(*, "label")= chr "Number of Parents/Children Aboard
```

Exercise III (Part 1 Solution)


```
summary(ptitanic)

 pclass        survived       sex           age
 1st:323   died    :809   female:466   Min.    : 0.1667
 2nd:277   survived:500   male  :843   1st Qu.:21.0000
 3rd:709                               Median :28.0000
                                       Mean   :29.8811
                                       3rd Qu.:39.0000
                                       Max.   :80.0000
                                       NA's   :263

     sibsp           parch
 Min.   :0.0000   Min.   :0.000
 1st Qu.:0.0000   1st Qu.:0.000
 Median :0.0000   Median :0.000
 Mean   :0.4989   Mean   :0.385
 3rd Qu.:1.0000   3rd Qu.:0.000
 Max.   :8.0000   Max.   :9.000
```

Exercise III (Part 1 Solution)


```
plot(ptitanic, col = ifelse(ptitanic$survived == "survived",
    "green", "red"))
```



Excursion: Data-Mining I

- Goal: Discover unknown relationships in the data using algorithms
- Here: Find out what properties (attributes) determined whether a passenger survived the catastrophe or not

x_1	...	x_i	y
1	..	1	survived
1	..	1	died
2	..	1	survived
1	..	1	?
1	..	1	?
2	..	5	?

Excursion: Data-Mining II

- We have a data frame
- We need to show the algorithm what the independent variables x are and what the dependent variable y is
- Two possibilities:
  - Split the data frame in x und y
  - Use R formula notation:
    Formula (Principle):
    $y \sim x_1 + ... + x_i$
    y and x are the names of the columns in the data frame

Exercise III (Part 2): Machine Learning from Disaster

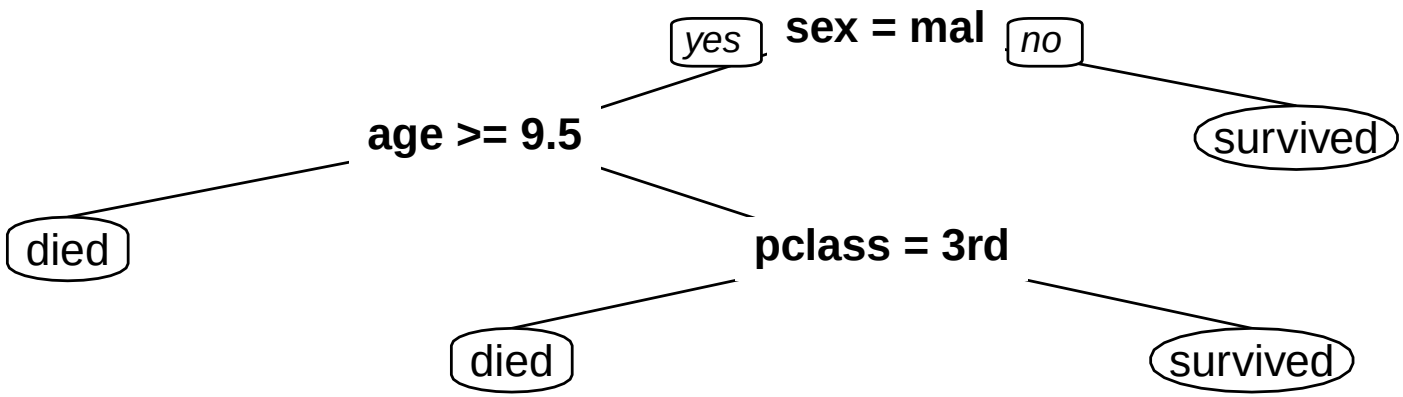
- Goal: Find out what properties (attributes) determined whether a passenger survived the catastrophe or not
- Therefore we create a decision tree using *rpart*. We only consider the attributes *sex*, *age*, und *pclass*:

```
rtree <- rpart(survived ~ sex + age + pclass
                # equivalent to  y ~ x1 + .. + x2

                , data = ptitanic) # data frame
```

- Draw the decision tree with **prp**
- Would you have survived?

Exercise III (Part 2 Solution)


```
prp(rtree)
```



Ende