DATA, Exploratory Data Analysis, and R

Introdution to Data Visualization

Marcus Birkenkrahe

September 23, 2024

Contents

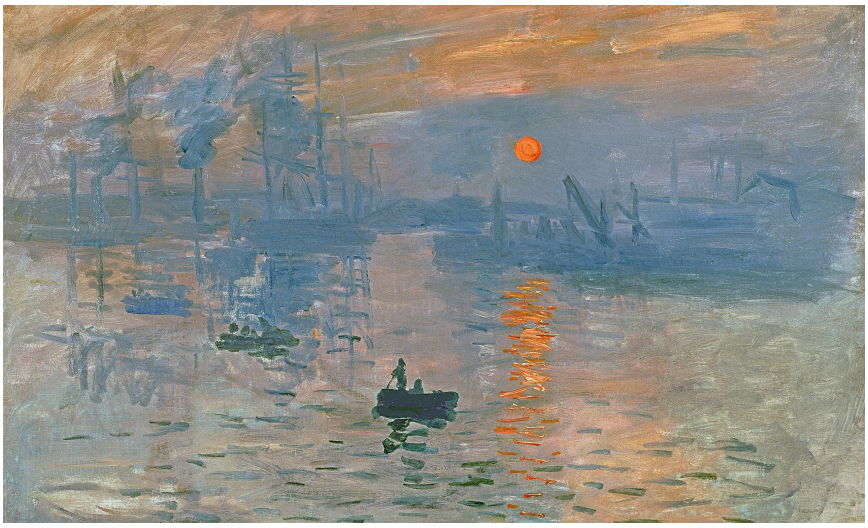


Figure 1: Impression, soleil levant (sunrise) by Claude Monet (1872)

1 README

- Why do we analyze data?
- Data as a concept and as a practice
- The importance of metadata
- Exploratory Data Analysis
- Numbers vs. graphs
- Data analysis workflow
- Why R?
- R package management
- Download R practice file

2 Why do we analyze data?

- Well, **what do you think**?

  Reframe the question: Tip: reframe WHY questions as WHAT questions -

  - What data?
  - What analysis?
  - What benefits?

  Answers:

  1. to **understand** what has happened or what is happening;
  2. to **predict** what is likely to happen, either in the future or in other circumstances we haven't seen;
  3. to **guide** us in making decisions.

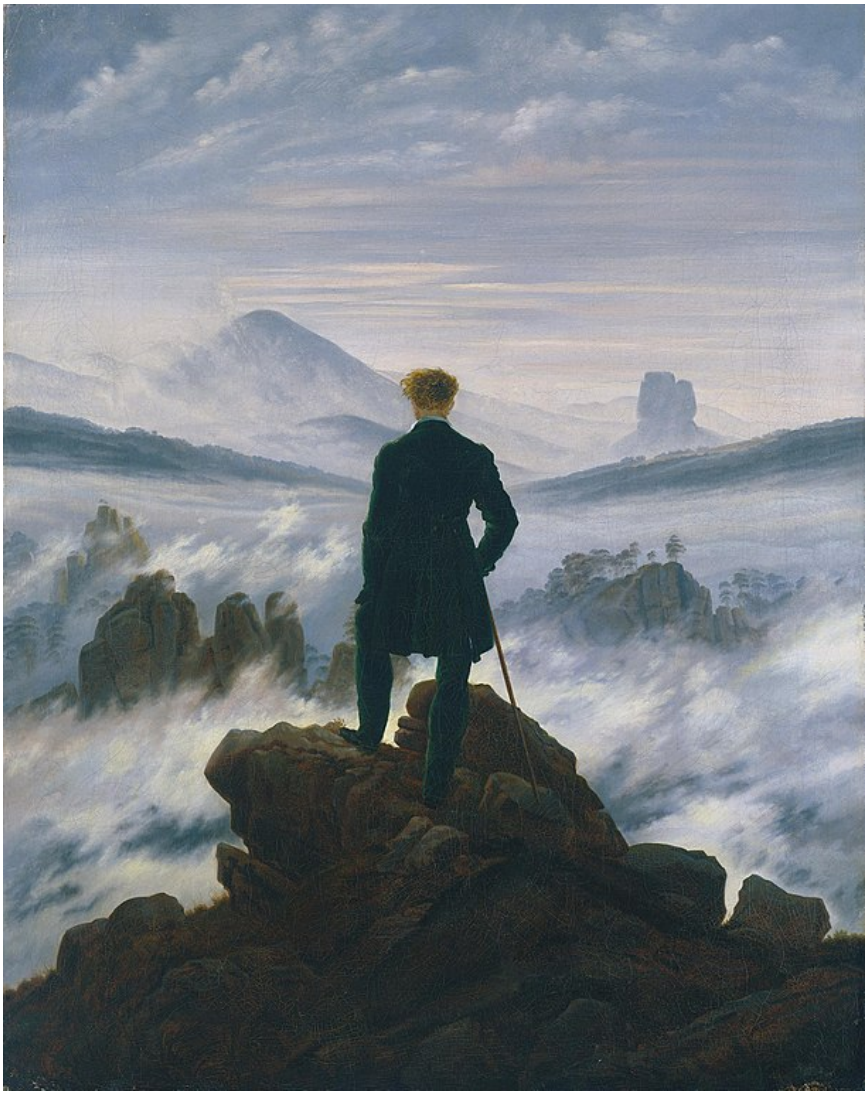


Figure 2: Caspar David Friedrich, Wanderer above a sea of fog (1818)

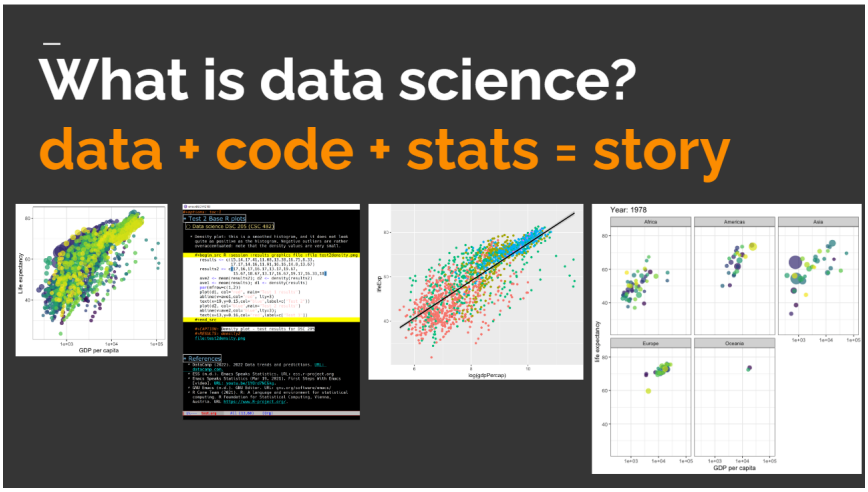


Figure 3: collapsed data science pipeline

3 Data - concept

- An **entity**, e.g.
  - family history of patient in medical study
  - competing company characteristics in marketing analysis
- An **event**, e.g.
  - demographic characteristics of voters
  - locations visited during a shopping visit
- A **process**, e.g.
  - manufacturing data from a production line
  - application information from a hiring process

Each of these terms emphasizes different objects and favors different visualization tools:

- **Entities** are stored & queried as SQL tables in relational database systems (and the corresponding algebras)
- **Events** drive processes, and event logs visualize processes and make them accessible to analysis (and can be visualized as graphs).

- **Processes** are event-driven structures that enable automation.

All of these can be visualized as Petri nets - directed, bipartite graphs that constitute discrete dynamical systems. The two parts are places and transitions. Realtime visualization with process mining.

4 Data - practice

- Data structures = rectangular array of observed values
- Rows = observation of entity, event, or process
- Columns = recorded characteristic or attribute

```
## extract first six records from the mtcars data frame
head(mtcars)
```

```
mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

- Data frame rows and columns have *names*
- Complete description with help(mtcars)
- Meta data supplement data frame content
- Which other data structures can you name?
  - **Arrays**
  - **Vectors**
  - **Matrices**
  - **Lists**
  - **Factor vectors**
  - **Graphs**
  - Classes
  - Queues
  - Heaps
  - Stacks

5 Meta data

- "Data about data" (Greek 'meta'= 'after', 'beyond')
- **What are the meta data for the `mtcars` dataset?**
  1. Data types of each column - `sapply(mtcars,class)`
  2. Summary structure of the data - `str(mtcars)`
  3. Summary statistics of the data - `summary(mtcars)`
  4. Dimension of the data frame - `dim(mtcars)`
  5. Row names - `rownames(mtcars)`
  6. Column names - `colnames(mtcars)`
  7. Byte size of the data - `object.size(mtcars)`
  8. Original source of the data - `help(mtcars)`
  9. Scientific paper analyzing the data `?mtcars`
  10. Which package the data belong to (if any) - `find("mtcars")`
  11. Help files for data - `??mtcars`
- **What could be issues with metadata?** Same as with data except often less reliable because has to be maintained:
  - **Completeness** - origin
  - **Consistency** - logic, values, (time) dependency
  - **Correctness** - accuracy of origin and validity

  With the rising importance of LLMs meta-data become even more important because they provide context, improve user interaction (cp. ChatGPT memory of you), allow for model updates, and bias mitigation.

  > "As potentially valuable as metadata is, we cannot afford to accept it uncritically: we should always cross-check the metadata with the actual data values, with our intuition and prior understanding of the subject matter, and with other sources of information that may be available." (Pearson, 2018)

- **What is the value of meta data:**

Figure 4: Greek goddess of peace and spring "Eirene"

1. Data discovery and identification (e.g. data types, structure)
2. Context and provenance (e.g. where and by whom collected and used)
3. Data understanding and meaning
4. Data quality and constraints (e.g. acceptable value ranges, size in memory, time of collection)
5. Data lineage (e.g. stuff done to the data after collection)
6. Licensing and usage restrictions (e.g. legal or ethical constraints)
7. Versioning
8. Data quality assessment (e.g. how accuracy of data was validated)

All of these point to the usefulness, meaning and truthfulness of the data - without available meta data, the underlying data mean little.

6 Practice: meta or not meta?

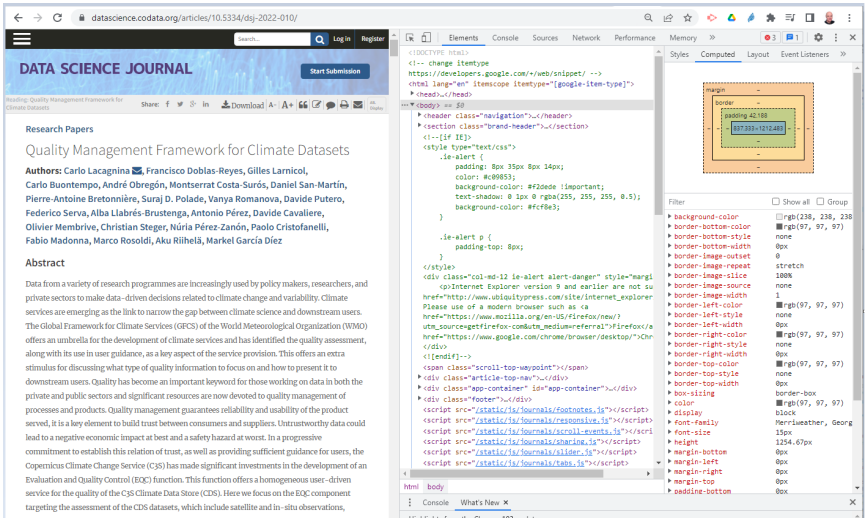


Figure 5: datascience.codata.org/articles/10.5334/dsj-2022-010/

Pair exercise: Identify the different types of data and metadata in the screenshot of an online journal article.

1. **Article content meta data:** Journal title, "Research paper", title, authors, length, date, keywords, publication place.

2. **Layout meta data:** HTML/CSS elements

3. **Browser meta data:** browser data (buttons for: download, font size, print, login, register, menu options; browser console; URL)

4. **Article content data:** abstract + paper text, tables and figures

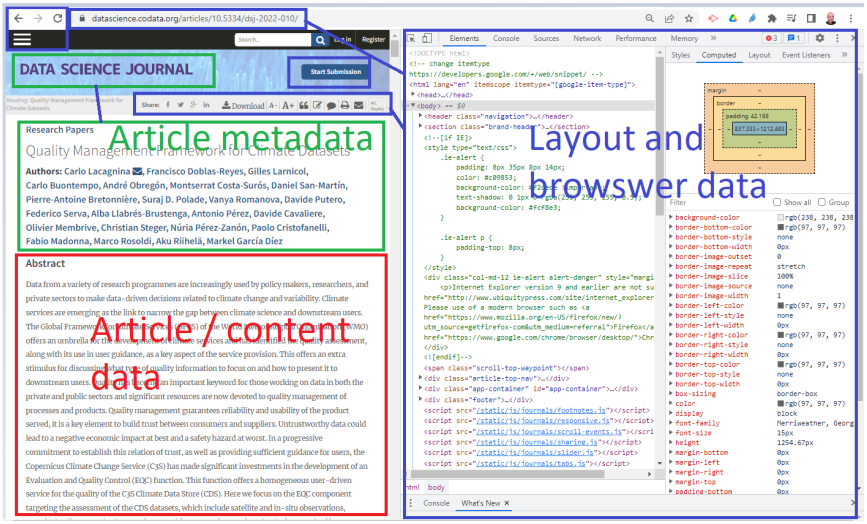


Figure 6: Solution

7 Installing older versions of packages

- For example for the `MASS` package: check your R `version` and then pick an earlier package version using the CRAN archive.

- If you have R version 4.0.4 (2021-02-15), then version 7.3.54 from 2021-05-03 is a safe bet:

```
install.packages("remotes")
require(remotes)
install_version("MASS", version="7.3.54")
library(MASS)
search()  # MASS appears in environment list
```

- To list functions in a package, use `lsf.str` for lots of detail, or `ls` for an overview:

```
lsf.str("package:MASS")
ls("package:MASS")
```

8 Problem: missing values

Open an Org-mode file to code along.

- The `Pima` datasets contained in `MASS` represent an interesting show case.
- Load the `MASS` dataset and filter all data sets contained therein:

```
library(MASS)
data(package="MASS")$results[,"Item"] -> datasets  # filter "Items" attribute
datasets # a character vector of package names
```

```
 [1] "abbey"     "accdeaths" "Aids2"     "Animals"   "anorexia"  "bacteria"  "beav1
 [8] "beav2"     "biopsy"    "birthwt"   "Boston"    "cabbages"  "caith"     "Cars
[15] "cats"      "cement"    "chem"      "coop"      "cpus"      "crabs"     "Cush
[22] "DDT"       "deaths"    "drivers"   "eagles"    "epil"      "farms"     "fgl"
[29] "forbes"    "GAGurine"  "galaxies"  "gehan"     "genotype"  "geyser"    "gilg
[36] "hills"     "housing"   "immer"     "Insurance" "leuk"      "mammals"   "mcyc
[43] "Melanoma"  "menarche"  "michelson" "minn38"    "motors"    "muscle"    "newc
[50] "nlschools" "npk"       "npr1"      "oats"      "OME"       "painters"  "petr
[57] "phones"    "Pima.te"   "Pima.tr"   "Pima.tr2"  "quine"     "Rabbit"    "roac
[64] "rotifer"   "Rubber"    "ships"     "shoes"     "shrimp"    "shuttle"   "Sitk
[71] "Sitka89"   "Skye"      "snails"    "SP500"     "steam"     "stormer"   "surv
[78] "synth.te"  "synth.tr"  "topo"      "Traffic"   "UScereal"  "UScrime"   "VA"
[85] "waders"    "whiteside" "wtloss"
```

```
data(package="MASS")$results[,"Item"] -> datasets
datasets
```

```
[1] "abbey"     "accdeaths" "Aids2"     "Animals"   "anorexia"  "bacteria"  "beav1"
 [8] "beav2"     "biopsy"    "birthwt"   "Boston"    "cabbages"  "caith"     "Cars93"
[15] "cats"      "cement"    "chem"      "coop"      "cpus"      "crabs"     "Cushings
[22] "DDT"       "deaths"    "drivers"   "eagles"    "epil"      "farms"     "fgl"
[29] "forbes"    "GAGurine"  "galaxies"  "gehan"     "genotype"  "geyser"    "gilgais"
```

```
[36] "hills"     "housing"   "immer"     "Insurance" "leuk"      "mammals"   "mcycle"
[43] "Melanoma"  "menarche"  "michelson" "minn38"    "motors"    "muscle"    "newcomb"
[50] "nlschools" "npk"       "npr1"      "oats"      "OME"       "painters"  "petrol"
[57] "phones"    "Pima.te"   "Pima.tr"   "Pima.tr2"  "quine"     "Rabbit"    "road"
[64] "rotifer"   "Rubber"    "ships"     "shoes"     "shrimp"    "shuttle"   "Sitka"
[71] "Sitka89"   "Skye"      "snails"    "SP500"     "steam"     "stormer"   "survey"
[78] "synth.te"  "synth.tr"  "topo"      "Traffic"   "UScereal"  "UScrime"   "VA"
[85] "waders"    "whiteside" "wtloss"
```

```
ls()
```

```
[1] "datasets" "r"        "x"        "y"
```

- Filter for datasets that contain "Pima":

  ```
  grep("Pima",datasets) # character vector indices whose values have "Pima" in them
  datasets[grep("Pima",datasets)] # return corresponding vector elements
  datasets[c(11,12,13)]
  ```

  ```
  [1] 58 59 60
  [1] "Pima.te"  "Pima.tr"  "Pima.tr2"
  [1] "Boston"   "cabbages" "caith"
  ```

- For funnsies: in one command

  ```
  data(package = "MASS")$results[grep("Pima", data(package = "MASS")$results[, "Iter
  ```

  ```
  [1] "Pima.te"  "Pima.tr"  "Pima.tr2"
  ```

- Check out structure of Pima datasets:

  ```
  library(MASS)
  str(Pima.te)
  str(Pima.tr)
  str(Pima.tr2)
  ```

```
'data.frame': 332 obs. of  8 variables:
 $ npreg: int  6 1 1 3 2 5 0 1 3 9 ...
 $ glu  : int  148 85 89 78 197 166 118 103 126 119 ...
 $ bp   : int  72 66 66 50 70 72 84 30 88 80 ...
 $ skin : int  35 29 23 32 45 19 47 38 41 35 ...
 $ bmi  : num  33.6 26.6 28.1 31 30.5 25.8 45.8 43.3 39.3 29 ...
 $ ped  : num  0.627 0.351 0.167 0.248 0.158 0.587 0.551 0.183 0.704 0.263 ...
 $ age  : int  50 31 21 26 53 51 31 33 27 29 ...
 $ type : Factor w/ 2 levels "No","Yes": 2 1 1 2 2 2 2 1 1 2 ...
'data.frame': 200 obs. of  8 variables:
 $ npreg: int  5 7 5 0 0 5 3 1 3 2 ...
 $ glu  : int  86 195 77 165 107 97 83 193 142 128 ...
 $ bp   : int  68 70 82 76 60 76 58 50 80 78 ...
 $ skin : int  28 33 41 43 25 27 31 16 15 37 ...
 $ bmi  : num  30.2 25.1 35.8 47.9 26.4 35.6 34.3 25.9 32.4 43.3 ...
 $ ped  : num  0.364 0.163 0.156 0.259 0.133 ...
 $ age  : int  24 55 35 26 23 52 25 24 63 31 ...
 $ type : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 2 1 1 1 2 ...
'data.frame': 300 obs. of  8 variables:
 $ npreg: int  5 7 5 0 0 5 3 1 3 2 ...
 $ glu  : int  86 195 77 165 107 97 83 193 142 128 ...
 $ bp   : int  68 70 82 76 60 76 58 50 80 78 ...
 $ skin : int  28 33 41 43 25 27 31 16 15 37 ...
 $ bmi  : num  30.2 25.1 35.8 47.9 26.4 35.6 34.3 25.9 32.4 43.3 ...
 $ ped  : num  0.364 0.163 0.156 0.259 0.133 ...
 $ age  : int  24 55 35 26 23 52 25 24 63 31 ...
 $ type : Factor w/ 2 levels "No","Yes": 1 2 1 1 1 2 1 1 1 2 ...
```

- The MASS package contains three different versions of the Pima indians data set (diabetes in women of the Pima tribe)
- MASS metadata comments (from the documentation):

  > "The training set `Pima.tr` contains a randomly selected set of 200 subjects, and `Pima.te` contains the remaining 332 subjects. `Pima.tr2` contains `Pima.tr` plus 100 subjects with missing values in the explanatory variables."

- The kaggle.com database is yet another version: more records, one more variable - the "Metadata" information is missing

- Missing data are often coded as `0` instead of `NA` leading to errors:

  > "A number of studies characterizing *binary classifiers* have been published using [the Pima] dataset as a benchmark where the authors were not aware that data values were missing." (Pearson, 2018)

- How many missing values `NA` do these different datasets have?

```
any(is.na(Pima.te))  # no missing values (NA)
any(is.na(Pima.tr))  # no missing values (NA)
any(is.na(Pima.tr2)) # has missing values (NA)
```

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

- The summary function is also useful in this regard:

```
summary(Pima.tr2)
```

```
    npreg              glu              bp              skin      
Min.   : 0.000000   Min.   : 56.0000   Min.   : 38.00000   Min.   : 7.00000  
1st Qu.: 1.000000   1st Qu.:101.0000   1st Qu.: 64.00000   1st Qu.:21.00000  
Median : 3.000000   Median :121.0000   Median : 72.00000   Median :29.00000  
Mean   : 3.786667   Mean   :123.7433   Mean   : 72.32056   Mean   :29.15347  
3rd Qu.: 6.000000   3rd Qu.:142.0000   3rd Qu.: 80.00000   3rd Qu.:36.00000  
Max.   :14.000000   Max.   :199.0000   Max.   :114.00000   Max.   :99.00000  
  NA's   :13          NA's   :98                                            
bmi               ped                age             type    
Min.   :18.20000   Min.   :0.0780000   Min.   :21.00000   No :194  
1st Qu.:27.10000   1st Qu.:0.2367500   1st Qu.:24.00000   Yes:106  
Median :32.00000   Median :0.3360000   Median :29.00000            
Mean   :32.05286   Mean   :0.4356567   Mean   :33.09667            
3rd Qu.:36.50000   3rd Qu.:0.5867500   3rd Qu.:40.00000            
Max.   :52.90000   Max.   :2.2880000   Max.   :72.00000            
NA's   :3                                                          
```

9 Excursion: `NA` values in standard functions

- Compute the average of three number 1,2,3 using the `mean` function.

```
mean(c(1,2,3))  # c(1,2,3) is the vector of elements 1,2,3
```

```
2
```

- Now add an `NA` to the vector and compute the average again:

```
mean(c(1,2,3,NA)) # vector of elements 1,2,3 and one missing element
```

```
[1] NA
```

- How can we fix this?

```
mean(c(1,2,3,NA), na.rm=TRUE)
```

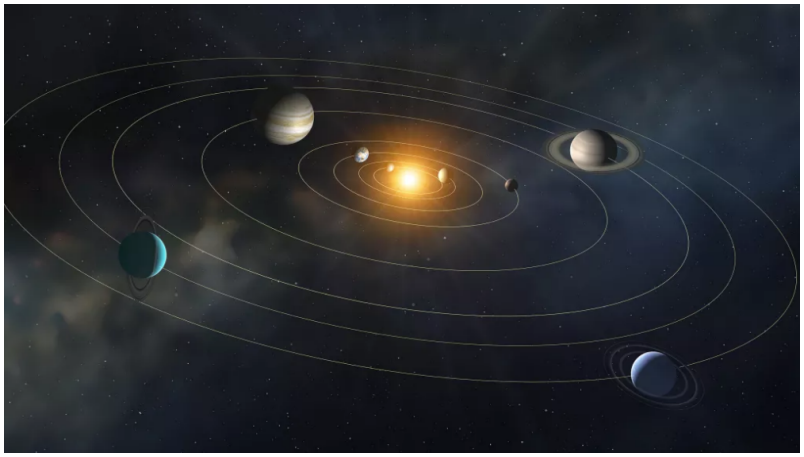
```
[1] 2
```

- How could you have found out that this is the way it works?

```
?mean
```

10 Problem: variable definitions

- How many planets are there orbiting the sun?



- Definitions count: e.g. *planethood* (Weintraub, 2007)
    1. the object is too small to generate nuclear fusion energy
    2. the object is big enough to be spherical
    3. the object must have a primary orbit around a star
- Unrecognized disagreements in the definition of a variable are possible between those who *measure and record* it, and those who use data in *analysis*.
- Prominent contemporary examples:
    1. **Epidemic data**: When does a patient die of COVID-19? What is the cause of death? When do two patients have the same disease? How reliable are data when a disease has not been studied?
    2. **Political data:** Who qualifies as "right-wing" or "left-wing"? Are these defined the same or even similar in 1850, 1920, 2024? Is it easy to get authentic data from people surveys?
    3. **Economic data**: What qualifies as "poverty" in different countries or even within the same country over time? Should one focus on income only, or also on housing, healthcare, education?

11 Exploratory data analysis (eda)

> "We look at *numbers* or *graphs* and try to find *patterns*. We pursue leads suggested by background information, imagination, patterns perceived, and experience with other data analyses." (Diaconis, 1985)

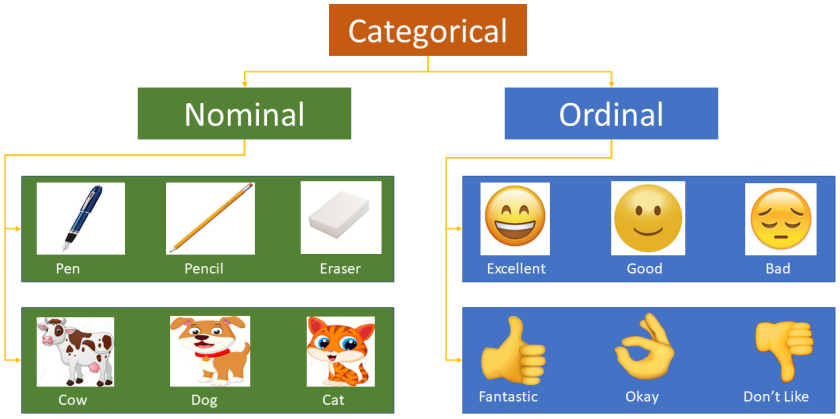
- Analysis is always based on exploring *numbers* (quantification)
- *Non-numerical* data are converted to numbers: e.g. *categorical* variables are converted from *discrete* named values ("political party", "city") into counts or relative *frequencies*
- In R, each discrete value or category is also called a *level*.

```
fv <- factor(c("male","female","female","female","non-disclosed"))
fv
```

```
[1] male          female        female        female        non-disclosed
Levels: female male non-disclosed
```

- **Further study:** If you have not completed DSC 105 (introduction to data science), check out chapter 4 ("Factors") of the DataCamp course "Introduction to R".

12 Types of categorical variables



- Few levels (e.g. "Firm", "Party", "City")
- Many levels (e.g. US ZIP code with 40,000 levels)
- Exploitable sub-structure (e.g. text data[1])

```
str(ToothGrowth)
```

```
'data.frame': 60 obs. of  3 variables:
 $ len : num  4.2 11.5 7.3 5.8 6.4 10 11.2 11.2 5.2 7 ...
 $ supp: Factor w/ 2 levels "OJ","VC": 2 2 2 2 2 2 2 2 2 2 ...
 $ dose: num  0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 ...
```

13 Some issues with graphs

- Humans are better at seeing patterns in graphs than numbers[2]

[1] Text data can be normalized (reduced - e.g. parsed into words, eliminating common words like "and", "of" and punctuation marks), and converted to numbers. The numbers are analyzed mathematically, and the result is transformed back to allow interpretation of the original text data. This technique leads to impressive NLP feats (so-called transformer ML models based on massive mined data sets, like OpenAI's line of GPT models).

[2] The plots show *Anscombe's quartet* - four scatterplots which despite having different numerical values all have identical mean, variance, and standard correlation (Source: revolutionanalytics.com).

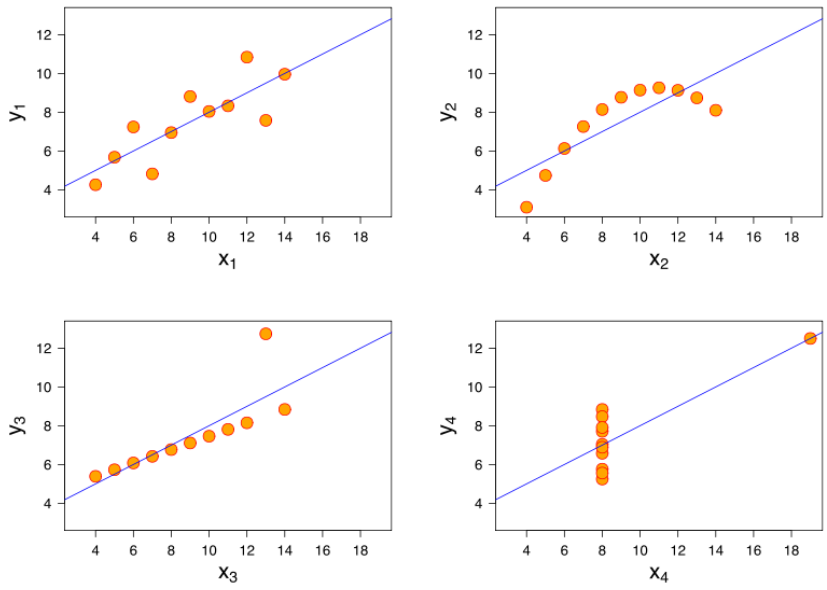


Figure 7: Anscombe dataset

- Use different graphs to explore and to explain - data mining is *exploratory*, data story telling is *explanatory*[3]
- Usefulness of a graph depends on **how data** are displayed, and strongly on **which data** are chosen to be displayed

14 Practice: Plot the Anscombe quartet

- We already looked at the `summary` data of this built-in dataset. Now let's print all of the data.
- Search for the `Anscombe` dataset in R.

```
??anscombe  # run help commands on the R shell
```

- Print the dataset:

[3]This difference goes deeper than data science: explanatory research is usually confirmatory (of some theory), while exploratory research is used to construct, or build, theory. Personal note: All of my own research has been exploratory rather than confirmatory.

```
anscombe
```

```
   x1 x2 x3 x4    y1   y2    y3    y4
1  10 10 10  8  8.04 9.14  7.46  6.58
2   8  8  8  8  6.95 8.14  6.77  5.76
3  13 13 13  8  7.58 8.74 12.74  7.71
4   9  9  9  8  8.81 8.77  7.11  8.84
5  11 11 11  8  8.33 9.26  7.81  8.47
6  14 14 14  8  9.96 8.10  8.84  7.04
7   6  6  6  8  7.24 6.13  6.08  5.25
8   4  4  4 19  4.26 3.10  5.39 12.50
9  12 12 12  8 10.84 9.13  8.15  5.56
10  7  7  7  8  4.82 7.26  6.42  7.91
11  5  5  5  8  5.68 4.74  5.73  6.89
```

- To get statistical summary information:

```
summary(anscombe)
```

```
 x1             x2             x3             x4           y1
Min.   : 4.0   Min.   : 4.0   Min.   : 4.0   Min.   : 8   Min.   : 4.260000
1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 6.5   1st Qu.: 8   1st Qu.: 6.315000
Median : 9.0   Median : 9.0   Median : 9.0   Median : 8   Median : 7.580000
Mean   : 9.0   Mean   : 9.0   Mean   : 9.0   Mean   : 9   Mean   : 7.500909
3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.:11.5   3rd Qu.: 8   3rd Qu.: 8.570000
Max.   :14.0   Max.   :14.0   Max.   :14.0   Max.   :19   Max.   :10.840000
 y2              y3             y4
Min.   :3.100000   Min.   : 5.39   Min.   : 5.250000
1st Qu.:6.695000   1st Qu.: 6.25   1st Qu.: 6.170000
Median :8.140000   Median : 7.11   Median : 7.040000
Mean   :7.500909   Mean   : 7.50   Mean   : 7.500909
3rd Qu.:8.950000   3rd Qu.: 7.98   3rd Qu.: 8.190000
Max.   :9.260000   Max.   :12.74   Max.   :12.500000
```

- `summary` is a generic function:

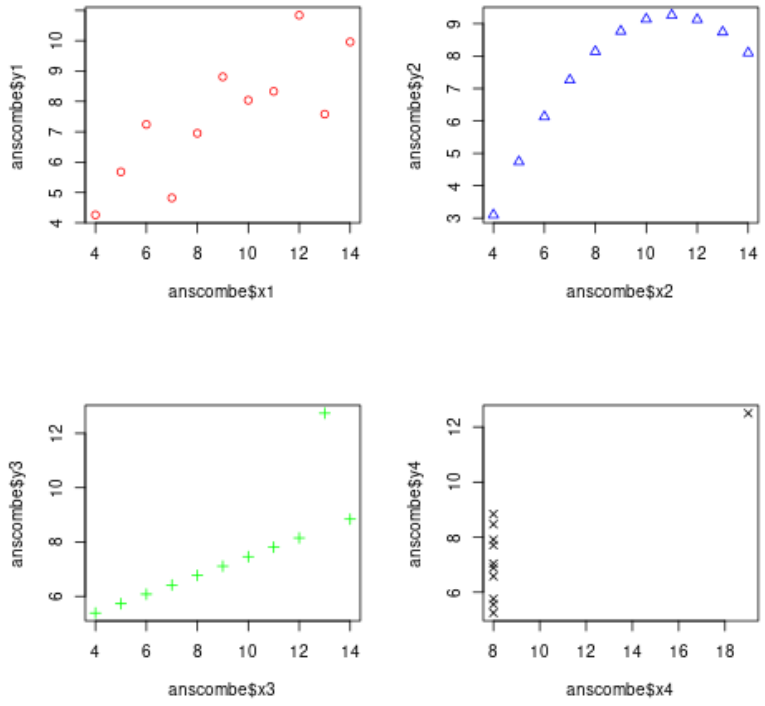
```
methods(summary) # data structures that summary works with
mean(anscombe)  # not a generic function
```

```
 [1] summary.aov                    summary.aovlist*
 [3] summary.aspell*                summary.check_packages_in_dir*
 [5] summary.connection             summary.data.frame
 [7] summary.Date                   summary.default
 [9] summary.ecdf*                  summary.factor
[11] summary.glm                    summary.infl*
[13] summary.lm                     summary.loess*
[15] summary.loglm*                 summary.manova
[17] summary.matrix                 summary.mlm*
[19] summary.negbin*                summary.nls*
[21] summary.packageStatus*         summary.polr*
[23] summary.POSIXct                summary.POSIXlt
[25] summary.ppr*                   summary.prcomp*
[27] summary.princomp*              summary.proc_time
[29] summary.rlm*                   summary.srcfile
[31] summary.srcref                 summary.stepfun
[33] summary.stl*                   summary.table
[35] summary.tukeysmooth*           summary.warnings
see '?methods' for accessing help and source code
[1] NA
Warning message:
In mean.default(anscombe) :
  argument is not numeric or logical: returning NA
```

- Now let's create a 2x2 facet plot with the Anscombe Quartet as shown in the figure, for the vector pairs (x1,y1), (x2,y2), (x3,y3), and (x4,y4). We'll make a basic plot only:

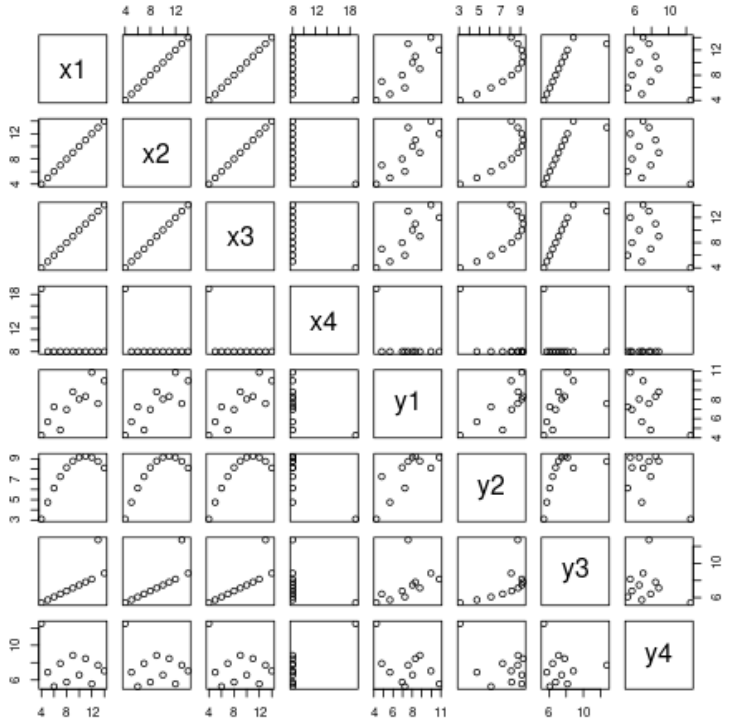
  1. Divide the plotting plane in 2 x 2 plots.
  2. Plot the Anscombe plots (four plots, different x,y vectors).
  3. Distinguish each plot by color (`col`) and point character (`pch`).

```
par(mfrow=c(2,2))
plot(anscombe$x1,anscombe$y1, col="red", pch=1)
plot(anscombe$x2,anscombe$y2, col="blue", pch=2)
plot(anscombe$x3,anscombe$y3, col="green", pch=3)
plot(anscombe$x4,anscombe$y4, col="black", pch=4)
```



The default for `plot` is a *pair plot*:

```
plot(anscombe)
```

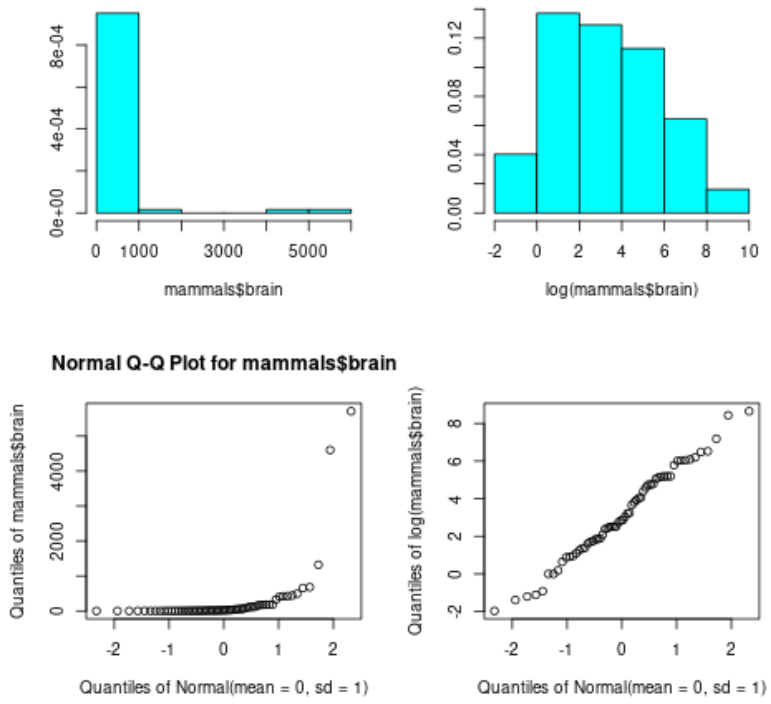


15 Practice: raw vs. transformed graph data

- The following two sets of plots are constructed from the `brain` element of the `mammals` dataset from the `MASS` package (doc) that lists body and brain weights for 62 different animals. The `qqplot` function is part of the `EnvStats` package (doc).
- **What do you think which graphs are more meaningful and why?**

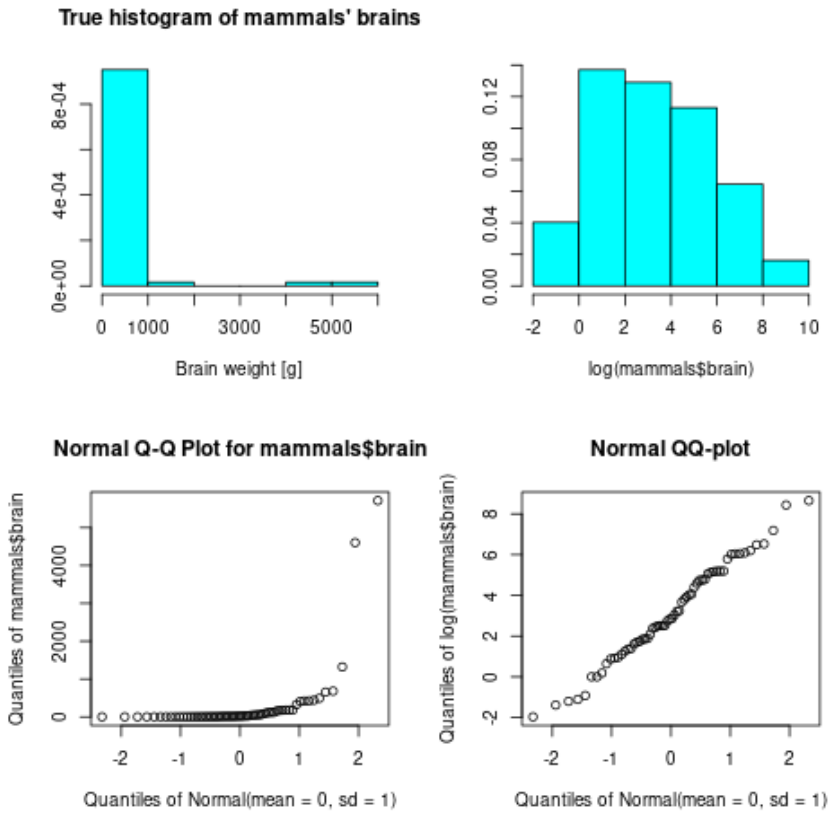
```
## load libraries
library(MASS)
library(EnvStats)
par(mfrow=c(2,2))

truehist(mammals$brain)
truehist(log(mammals$brain))
```

```
qqPlot(mammals$brain)
qqPlot(log(mammals$brain),main="")
```



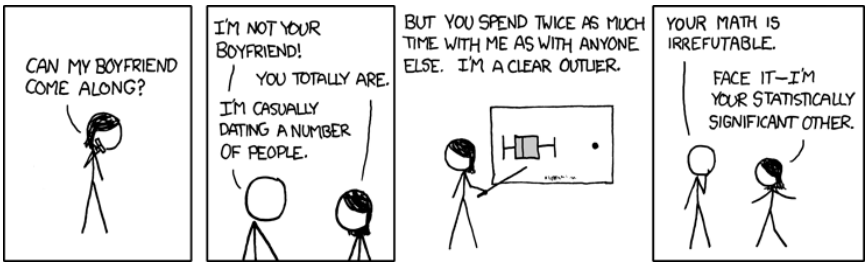

```
## load libraries
library(MASS)
library(EnvStats)

## set up 2 x 2 plot panel
par(mfrow=c(2,2))

## plot true histograms of the brain size data
truehist(mammals$brain,
   main="True histogram of mammals' brains",
   xlab="Brain weight [g]")
truehist(log(mammals$brain))
```

```
## make a quantile-quantile plot of the brain data
qqPlot(mammals$brain)
qqPlot(log(mammals$brain),main="")
title("Normal QQ-plot")
```



- The plots tell us something about the *distribution* of data values.
- The left-hand pair were generated from *raw data* values, the right-hand pair were generated from *log-transformed* data.
- The right-hand pair suggests that the data exhibit a *Gaussian* (normal) distribution (QQ-plots compare two distributions).

16 R for exploratory analysis



- Exploratory analysis has more use for graphical tools
- R supports many different graphical displays and plot types
- Important focus: searching for anomalies and outliers in the data

17 Data analysis workflow

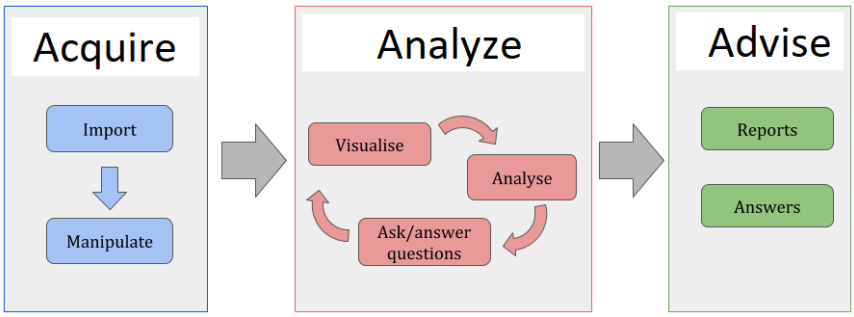


Figure 8: Data analysis workflow (emanuelaf.github.io - modified)

1. **Acquire**: make data available to the software
2. **Analyse**: perform the analysis
3. **Advise**: make analysis results available to those who need them

- In **training**, the emphasis is often on (2) analysis, and pre-loaded, small, clean datasets and well-tested packages are used.
- On the **job**, the emphasis is on (1) acquisition, and much time is spent importing and readying the data for analysis

- In **business**, the main interest is (3) advice for decision-making support, hence the shift to storytelling and interpretation

18 Computers

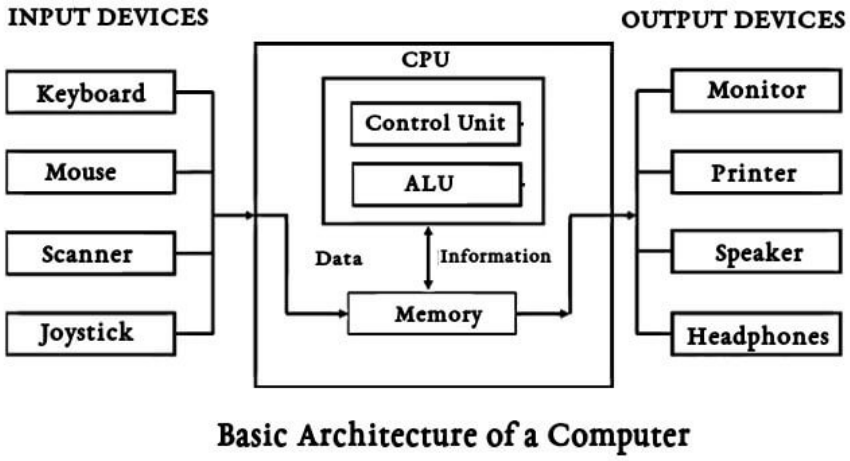


Figure 9: Von Neumann computer architecture (PSC Arivukal, 2020)

- RAM is several orders of magnitude faster than NVM
- Most R functions require raw data and results to fit in RAM
- OS and Internet impose severe infrastructure constraints [4]

[4] Though they can also be enablers of education: e.g. Linux and the command line shell as a data science tool, and online REPL installations (usually Docker containers) as training grounds.

19 Why R?

- R is FOSS (Free Open Source Software) available for all OS
- Supported range of analysis methods ready for use
- Unix-style package and version control system
- Diverse, active community of users and developers

20 The structure of R

1. Set of *base R packages* for basic statistics, data analysis, graphics
2. Set of *recommended packages* included in installations (like `MASS`)
3. Set of *optional add-on packages* for special purposes

Example: The optional, popular `ggplot2` graphics package was downloaded more than 272 mio. times between 2012 and 2020, with a monthly average of $>$ 800k downloads (Source: CRAN, 2021).

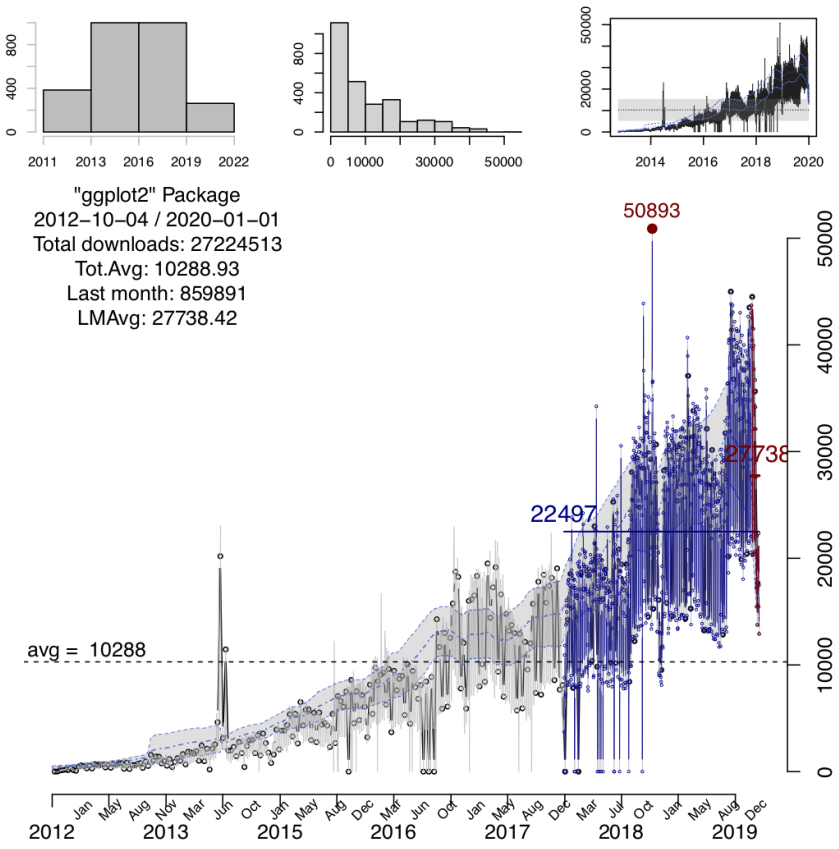


Figure 10: ggplot2 downloads from CRAN 2012-2020

21 Installation and loading R packages

- We'll do this directly on the command line (for more, see here):
- Installation = download and unpacking of binary or compilation (on Windows, when you're asked, do not compile from source):

```
install.packages("MASS")
install.packages("EnvStats")
```

- Loading = load package (functions + datasets) into current R session:

```
library(MASS)
library(EnvStats)
```

- Alternatively, you can use the Rgui program, or the RStudio IDE

22 Optional installation in the Rgui (Windows)

- Start the Rgui from the CMD line terminal
- The Rgui includes a command line and graphics
- The RTerm or R program is a console only
- In the R GUI, find the tab "Packages"
- Set CRAN mirror site (closest to you)
- Install or update package from list

23 Questions to ask from data

1. Where does the dataset come from, and how is it documented?
2. How many records (rows) does this dataset contain?
3. How many fields (variables, columns) are included in each record?
4. What kinds of variables are these (e.g. numerical, categorical)
5. Are there missing values? (`NA`)

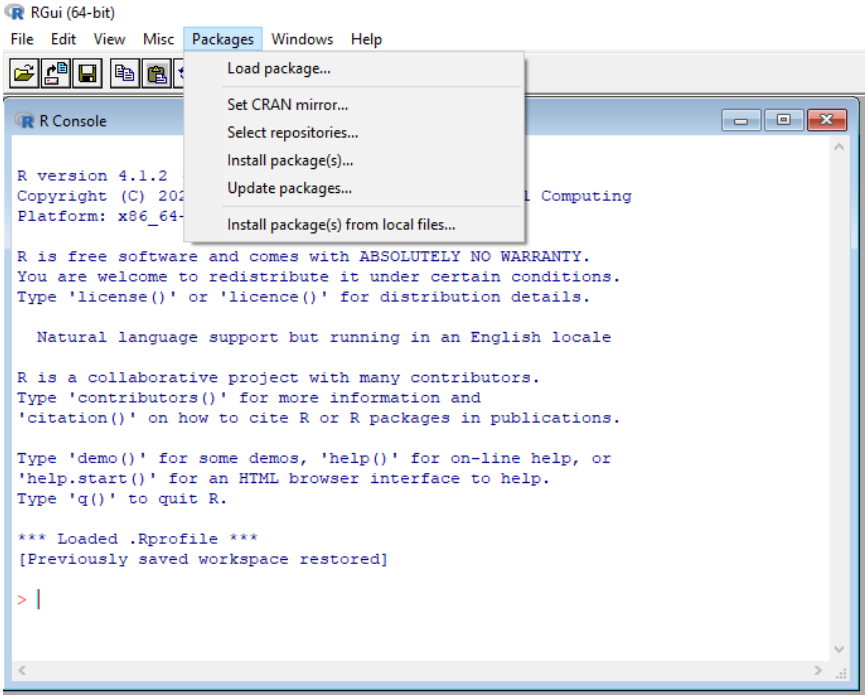


Figure 11: Package management in the Rgui program

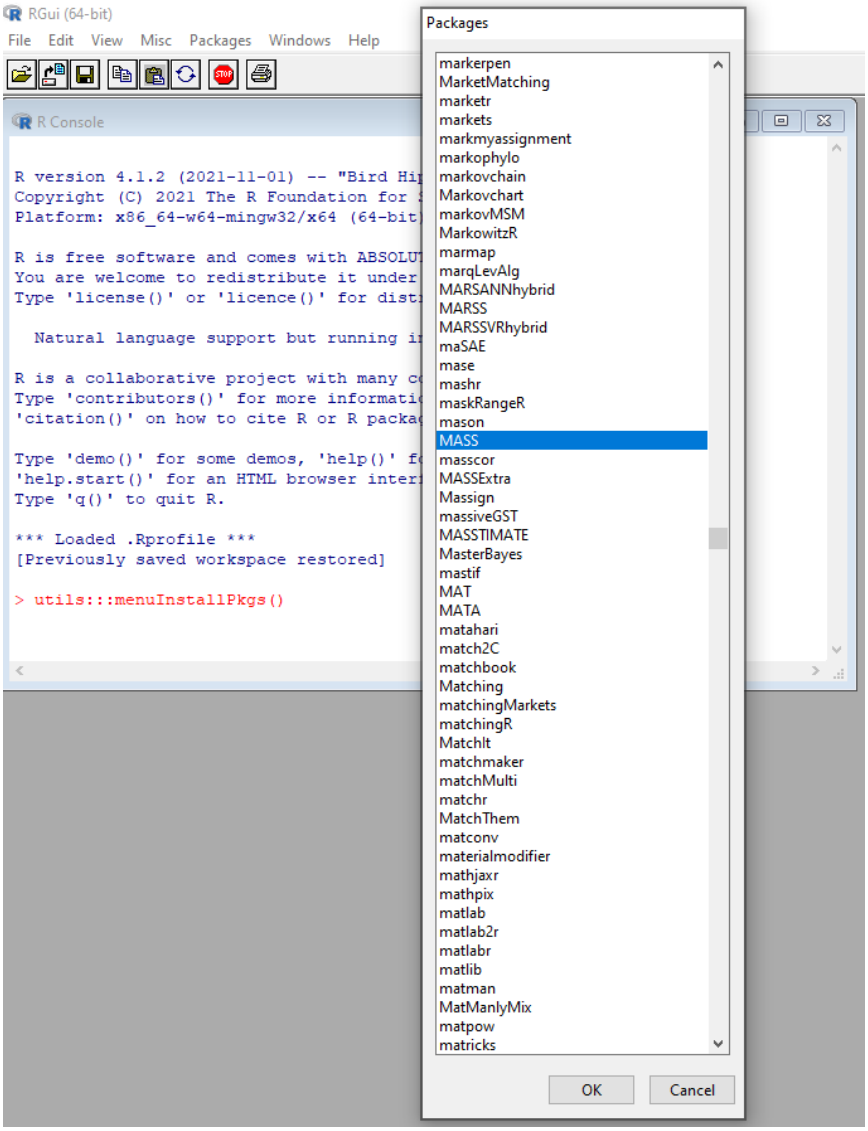


Figure 12: Package management in the Rgui program

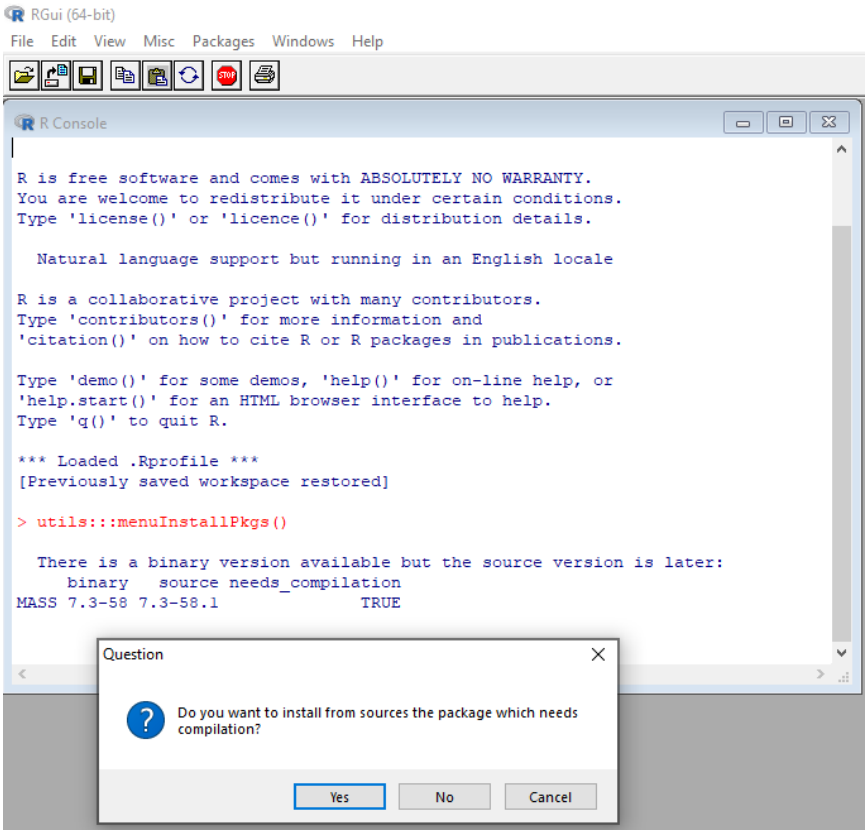


Figure 13: Package management in the Rgui program

6. If there are missing values: are these variables always observed?
7. If there are missing values: how are they represented?
8. Are the variables included in the dataset the ones we expect?
9. Are the variable values consistent with what we expect?
10. Do the variables exhibit the relationships we expect?

24 Practice: a representative R session

- To download the practice file on Linux:

  ```
  wget -O eda.org https://tinyurl.com/eda-practice-org
  ```

- Open the file in Emacs to work through it.
- Summary:

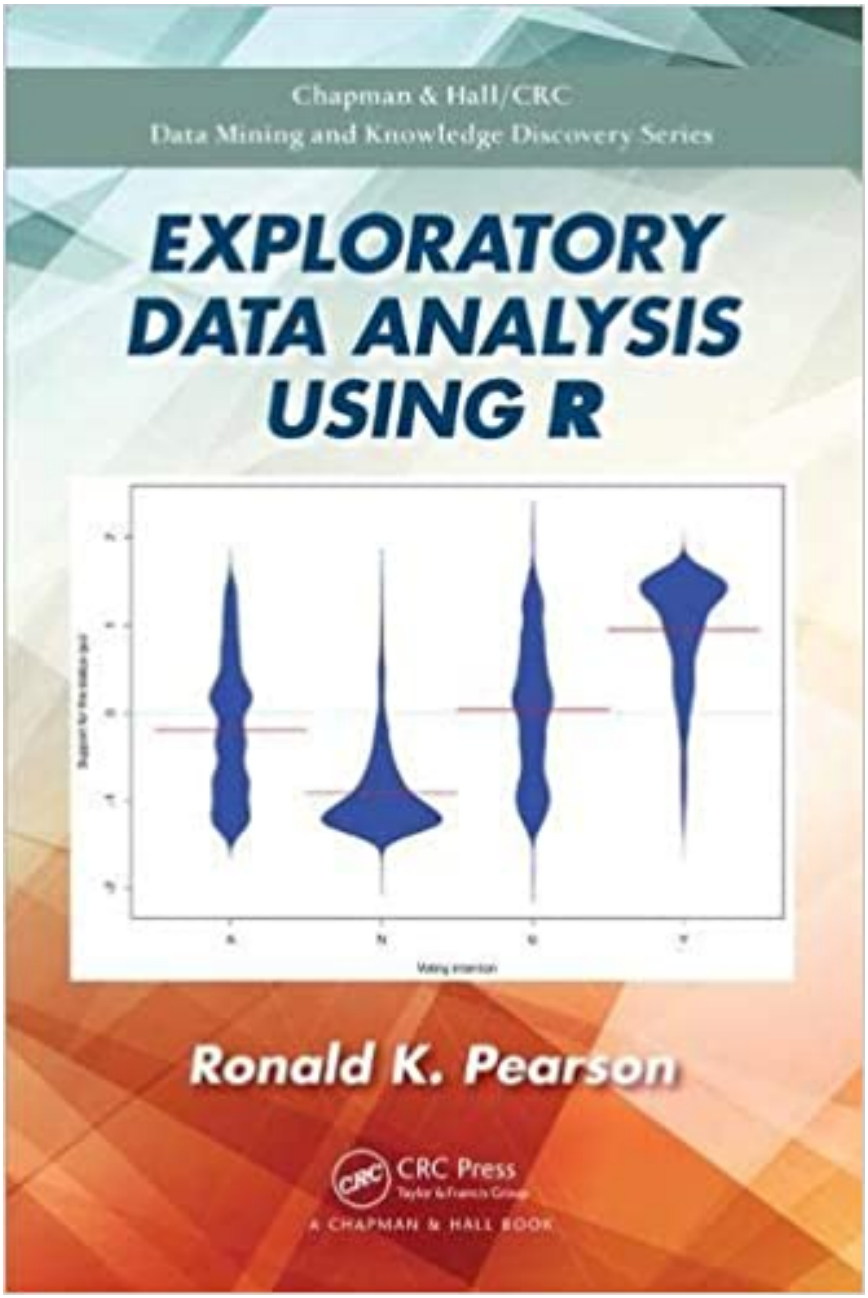
25 Concept summary

- Data are analysed to understand, predict, or guide decisions
- Data are entities, events or processes
- Meta data contain critical information for validation
- The data analysis workflow: acquire, analyze, advise
- R is FOSS, specialized on stats, and popular
- CRAN is the central hub for R package management

26 GLOSSARY

TERM	MEANING
Data frame	Rectangular array
Observation	Recorded event
Attribute	Characteristic
Meta data	Data about data
Data	Entity, event, process
Binary classifier	Attribute with 2 values
Missing value (`NA`)	Values that were not recorded
Categorical variable	Non-numerical, discrete
Level	Category, discrete value
Anomaly, outlier	Unusual data
CRAN	Comprehensive R Archive Network
Rgui	R console pgm with graphics
Rterm	R console (terminal) pgm only

27 References



- CRAN (27 April 2021). Visualize downloads from CRAN Packages.

Online: cran.r-project.org.

- OpenAI (2022). Example: Generate an outline for a research topic. Online: beta.openai.com.
- Pearson, R.K. (2018). Exploratory Data Analysis Using R. CRC Press.
- PSC Arivukal (July 26, 2020). Basic Computer Architecture. Online: pscarivukal.com.
- Revolutionanalytics (May 2, 2017). The Datasaurus Dozen. Online: blog.revolutionanalytics.com.