XV ERBD
Escola Regional de Banco de Dados
2019
Chapecó - SC

Data Analysis

CEFET/RJ

Eduardo Ogasawara
http://eic.cefet-rj.br/~eogasawara

Why Data Mining?

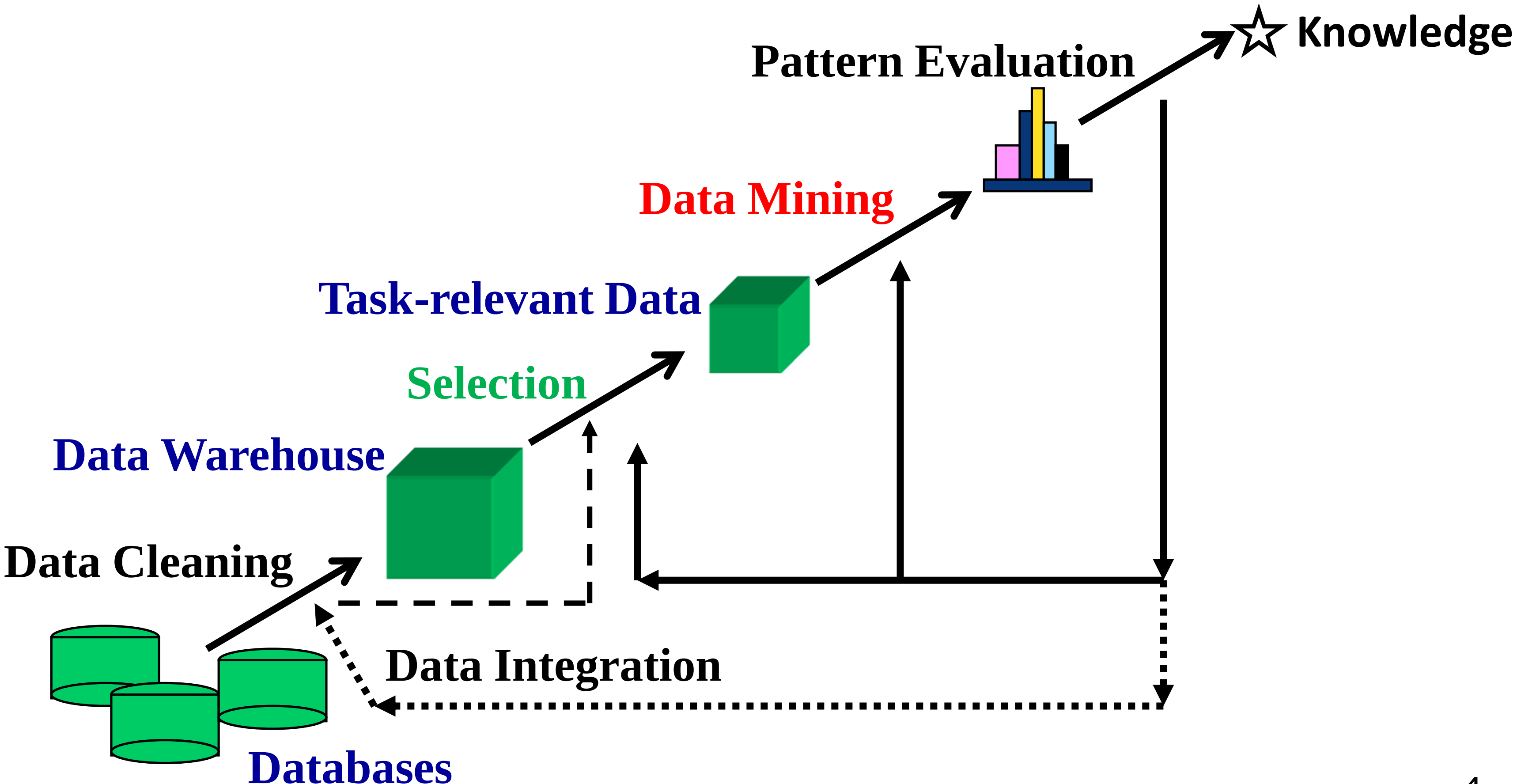
- The explosive growth of data: from terabytes to petabytes
  - Data collection and data availability
    - Automated data collection tools, database systems, Web
  - Major sources of abundant and diverse data (Big Data)
    - Business: Web, e-commerce, transactions, stocks
    - Science: sensors, astronomy, bioinformatics, simulation
    - Society and everyone: news, photos, videos, open data, IoT
- We are drowning in data, but starving for knowledge!
- “Need is the mother of invention”
  - Data mining - Automated analysis of massive data sets

What is Data Mining?

- Data mining (knowledge discovery from data)
  - Extraction of interesting (non-trivial, implicit, previously unknown and potentially useful) patterns or knowledge from a ~~massive~~ amount of data
- Alternative names
  - Knowledge discovery in databases (KDD)
  - knowledge extraction
  - business intelligence
  - data analysis
- Watch out: Is everything “data mining”?
  - Simple search and query processing ✘
  - (Deductive) expert systems ✘

Knowledge discovery from data (KDD) process

- This is a view from typical database systems
- Data mining plays an essential role in the KDD process



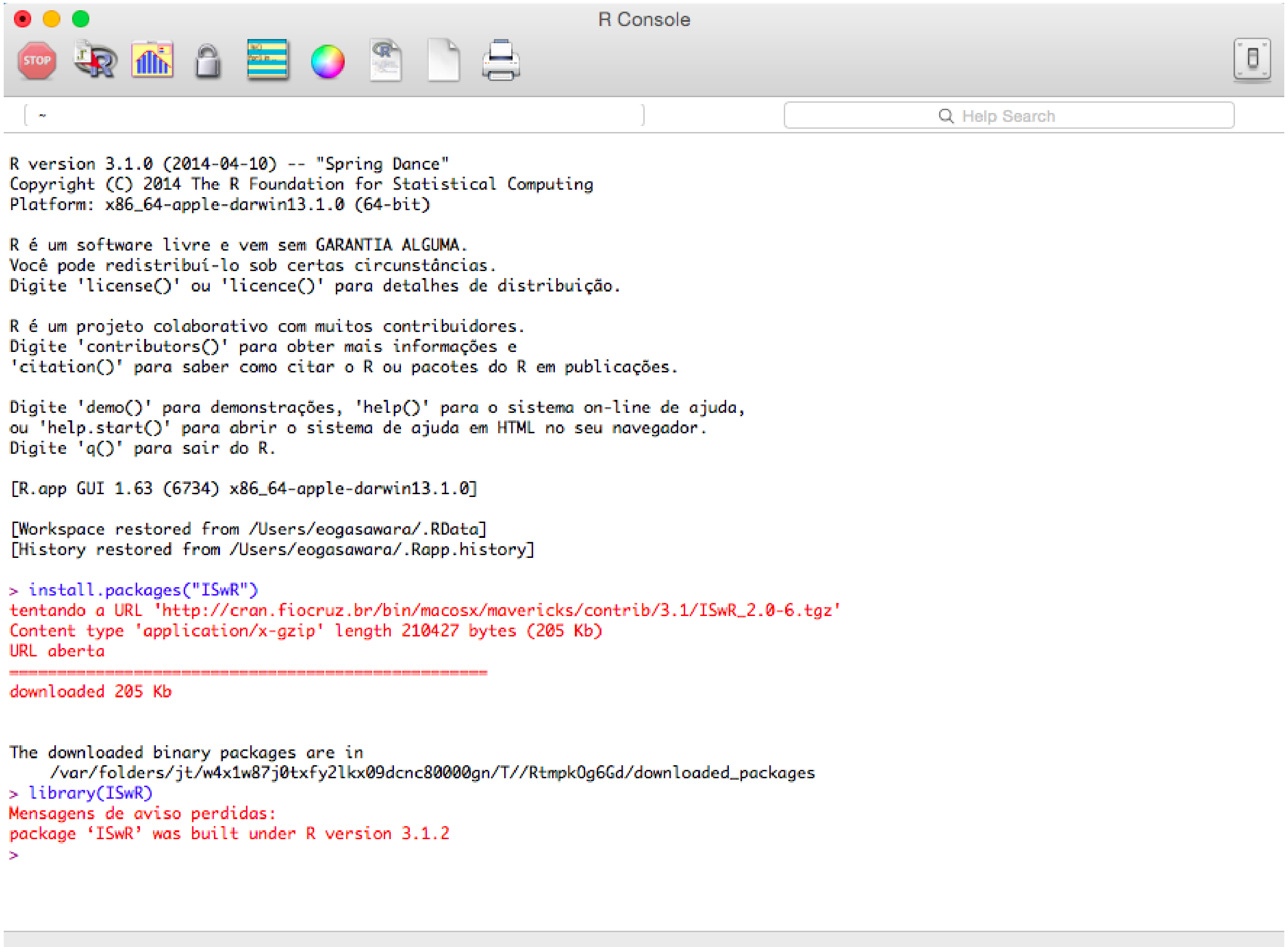
Data Analysis

- Data analysis is a process of inspecting, cleansing, transforming, and modeling data for KDD
- The process of data analysis
  - Data selection
  - Data processing
    - Cleaning, transforming
  - Exploratory data analysis
  - Communication

Basics of R

- R is a programming language and free software environment for statistical computing
  - Supported by the R Foundation for Statistical Computing
- Created by Ross Ihaka and Robert Gentleman at Auckland University, New Zealand
- R was derived by S (Bell Laboratories - AT&T)
- R is a language broadly used by statisticians, data miners, and data scientists

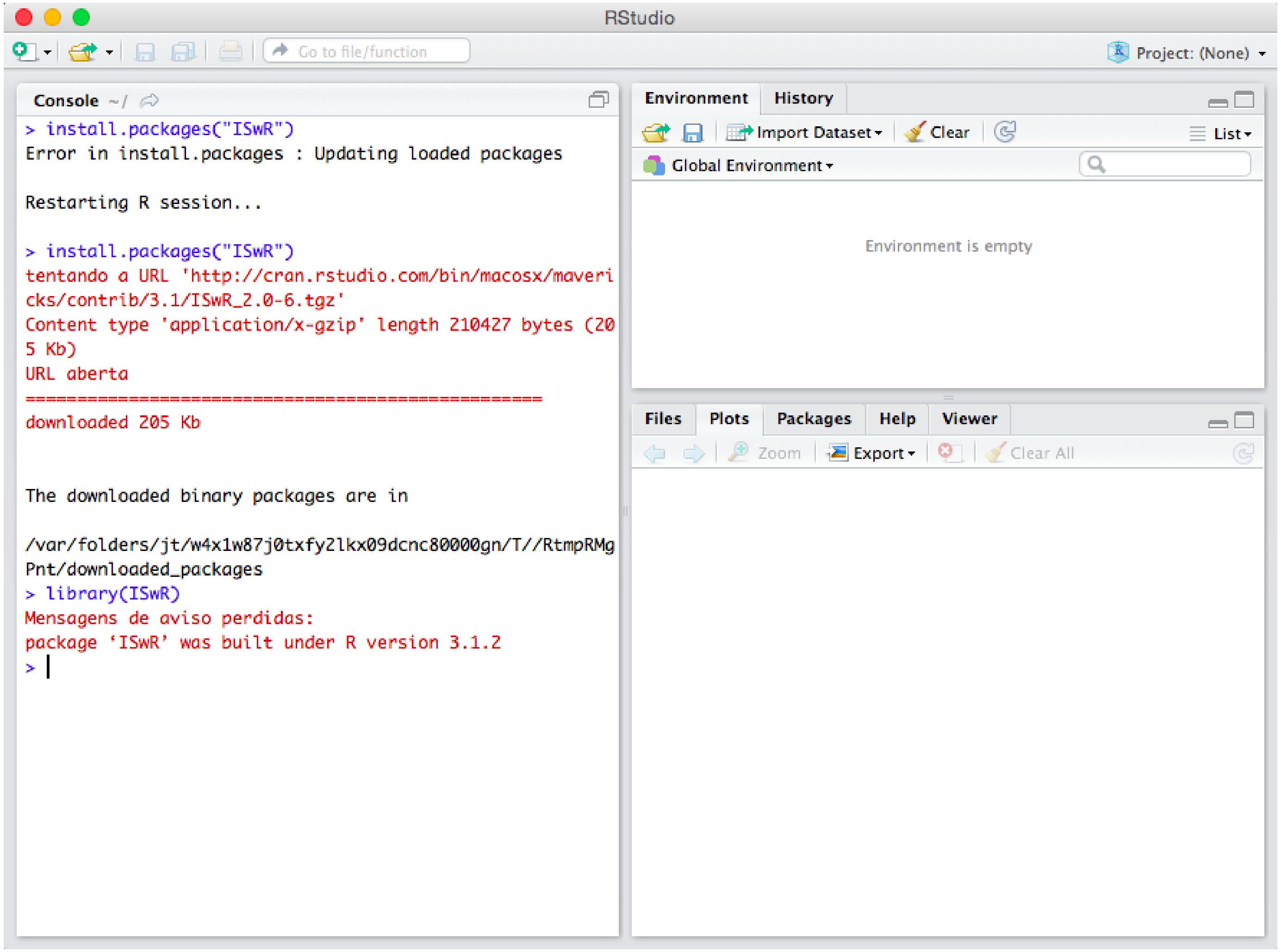
R Console




```
R version 3.1.0 (2014-04-10) -- "Spring Dance"
Copyright (C) 2014 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.1.0 (64-bit)

R é um software livre e vem sem GARANTIA ALGUMA.
Você pode redistribuí-lo sob certas circunstâncias.
Digite 'license()' ou 'licence()' para detalhes de distribuição.

R é um projeto colaborativo com muitos contribuidores.
Digite 'contributors()' para obter mais informações e
'citation()' para saber como citar o R ou pacotes do R em publicações.

Digite 'demo()' para demonstrações, 'help()' para o sistema on-line de ajuda,
ou 'help.start()' para abrir o sistema de ajuda em HTML no seu navegador.
Digite 'q()' para sair do R.

[R.app GUI 1.63 (6734) x86_64-apple-darwin13.1.0]

[Workspace restored from /Users/eogasawara/.RData]
[History restored from /Users/eogasawara/.Rapp.history]

> install.packages("ISwR")
tentando a URL 'http://cran.fiocruz.br/bin/macosx/mavericks/contrib/3.1/ISwR_2.0-6.tgz'
Content type 'application/x-gzip' length 210427 bytes (205 Kb)
URL aberta
==================================================
downloaded 205 Kb


The downloaded binary packages are in
    /var/folders/jt/w4x1w87j0txfy2lkx09dcnc80000gn/T//Rtmpk0g6Gd/downloaded_packages
> library(ISwR)
Mensagens de aviso perdidas:
package 'ISwR' was built under R version 3.1.2
>
```

Available for Windows, Mac, Linux

R Studio
http://www.rstudio.com



Great advantages: IDE with data visualization, debugging

CRAN Packages

- A broad number of packages (CRAN)
  - https://cran.r-project.org
- Strong Point of R
  - More than 14000 available packages (apr/2019)
  - http://cran.r-project.org/web/packages/
- Package installation
- Package loading

```
install.packages("TSPred")
install.packages("STMotif")
```

```
package 'TSPred' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\eduar\AppData\Local\Temp\RtmpMr5h0i\downloaded_packages
package 'STMotif' successfully unpacked and MD5 sums checked

The downloaded binary packages are in
        C:\Users\eduar\AppData\Local\Temp\RtmpMr5h0i\downloaded_packages
```

```
require(ggplot2)
require(TSPred)
require(STMotif)
```

```
Loading required package: ggplot2
Loading required package: TSPred
Warning message:
"package 'TSPred' was built under R version 3.5.3"Loading required package: STMotif
Warning message:
"package 'STMotif' was built under R version 3.5.3"
```

Basic concepts

- Assignment
- Value display
- Logical test
- Vector definition
    - Computing BMI
- Printing values

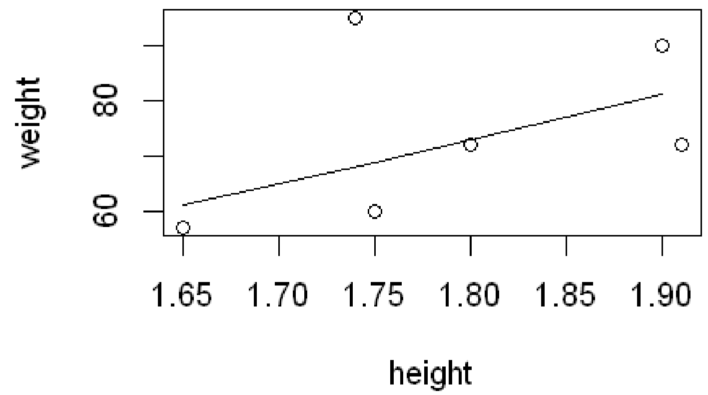
```
x <- 2 # variable assignment

x # variable evaluation

is.numeric(x) # variable

weight = c(60, 72, 57, 90, 95, 72) # vector with six obervations

height = c(1.75, 1.80, 1.65, 1.90, 1.74, 1.91)

bmi = weight/height^2

print(bmi)

print(sprintf("%.2f +/- %.2f", mean(bmi), sd(bmi)))
```

```
2

TRUE

[1] 19.59184 22.22222 20.93664 24.93075 31.37799 19.73630
[1] "23.13 +/- 4.49"
```

Plotting graphics & Statistical analysis

- Plotting a scatter graphics
  - Canvas is active until the next plot
- Test theoretical value of BMI equals to 22.5
  - Null hypothesis: no difference observed (p-value > 5%)
  - Alternative hypothesis: they are different

```
plot(height, weight)

hh = c(1.65, 1.70, 1.75, 1.80, 1.85, 1.90)
lines(hh, 22.5 * hh^2)
```




```
t.test(bmi, mu=22.5)
```

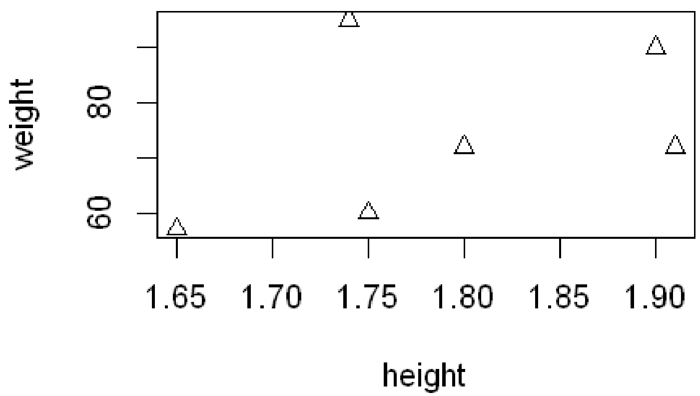
```
        One Sample t-test

data:  bmi
t = 0.34488, df = 5, p-value = 0.7442
alternative hypothesis: true mean is not equal to 22.5
95 percent confidence interval:
 18.41734 27.84791
sample estimates:
mean of x
 23.13262
```

Default arguments and help for functions

- Functions have default values
- View parameters of the function
- Use online help

```
plot(height, weight, pch=2)

args(plot.default)

?graphics::plot
```

```
function (x, y = NULL, type = "p", xlim = NULL, ylim = NULL,
    log = "", main = NULL, sub = NULL, xlab = NULL, ylab = NULL,
    ann = par("ann"), axes = TRUE, frame.plot = axes, panel.first = NULL,
    panel.last = NULL, asp = NA, ...)
NULL
```



More about vectors

- Operations with NA
- Name of observations
- Scalar multiplication

```
x <- c(A=1, B=NA, C=3)

mean(x)

mean(x, na.rm=TRUE)

names(x)

x["B"] <- 2

x["B"]*x
```

<NA>

2

'A' 'B' 'C'

A	2
B	4
C	6

Matrix

- Creation
- Creation by rows
- Names for rows and columns
- Transpose
- Determinant

```
m <- 1:9
dim(m) <- c(3,3)
m

mb <- matrix(1:9, nrow=3,byrow=TRUE)
rownames(mb) = LETTERS[1:3]
mb

t(m)

m*x

det(m)
```

1	4	7
2	5	8
3	6	9

A	1	2	3
B	4	5	6
C	7	8	9

1	2	3
4	5	6
7	8	9

1	4	7
4	10	16
9	18	27

0

Factors

- Factors are variables in R that refer to categorical data
- Factors in R are stored as a vector of integer values with a corresponding set of character values to use when the factor is displayed
- Both numeric and character variables can be made into factors, but a factor's levels are always character values

```
pain = c(0,3,2,2,1)
fpain = factor(pain,levels=0:3)
levels(fpain) = c("none","mild","medium","severe")

fpain

as.numeric(fpain)

levels(fpain)
```

```
none severe medium medium mild
▶ Levels:

1 4 3 3 2

'none' 'mild' 'medium' 'severe'
```

Lists

- Lists are the R objects which contain elements of different types, such as numbers, strings, vectors, matrix, data frame, and another list inside it.
- A list can also contain a matrix or a function as its elements
- A list is created using the list() function

```
x = c(5260,5470,5640,6180,6390,
      6515,6805,7515,7515,8230,8770)
y = c(3910,4220,3885,5160,5645,
      4680,5265,5975,6790,6900,7335)

lst <- list(A=x, B=y)

lst

lst$A
```

$A

5260 5470 5640 6180 6390 6515 6805 7515 7515 8230 8770

$B

3910 4220 3885 5160 5645 4680 5265 5975 6790 6900 7335

5260 5470 5640 6180 6390 6515 6805 7515 7515 8230 8770

Data frames

- A data frame is a table where each column corresponds to attributes, and each row corresponds to a tuple (object)

```
d <- data.frame(A=lst$A,B=lst$B)
d

df <- d[d$A > 7000 | d$A < 6000,]
df
```

A	B
5260	3910
5470	4220
5640	3885
6180	5160
6390	5645
6515	4680
6805	5265
7515	5975
7515	6790
8230	6900
8770	7335

	A	B
1	5260	3910
2	5470	4220
3	5640	3885
8	7515	5975
9	7515	6790
10	8230	6900
11	8770	7335

Implicitly Loops – sapply, lapply

- lapply, sapply executes a function for each column
  - The first character defines the return type
    - l – list, s – simple (vector or matrix)
  - The second parameter is the function to invoke
  - Following parameters are passed to the invoked function
- apply is the generic function
  - The second parameter defines if it calls the function for each row (1) or each column (2)

```
lapply(d, min, na.rm=TRUE)

sapply(d, min, na.rm=TRUE)

apply(d, 1, min)

apply(d, 2, min)
```

$A
5260
$B
3885

A	5260
B	3885

3910 4220 3885 5160 5645 4680 5265 5975 6790 6900 7335

A	5260
B	3885

Sort and order

```
sort(d$B)
o <- order(d$B)
o
ds <- d[o,]
ds
```

3885 3910 4220 4680 5160 5265 5645 5975 6790 6900 7335

3 1 2 6 4 7 5 8 9 10 11

	A	B
3	5640	3885
1	5260	3910
2	5470	4220
6	6515	4680
4	6180	5160
7	6805	5265
5	6390	5645
8	7515	5975
9	7515	6790
10	8230	6900
11	8770	7335

Loading and saving files

```r
wine = read.table("http://archive.ics.uci.edu/ml/machine-learning-databases/wine/wine.data",
    header = TRUE, sep = ",")
head(wine)
save(wine, file="wine.RData")

rm(wine)

load("wine.RData")
write.table(wine, file="wine.csv", row.names=FALSE, quote = FALSE)
```

X1	X14.23	X1.71	X2.43	X15.6	X127	X2.8	X3.06	X.28	X2.29	X5.64	X1.04	X3.92	X1065
1	13.20	1.78	2.14	11.2	100	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1050
1	13.16	2.36	2.67	18.6	101	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1185
1	14.37	1.95	2.50	16.8	113	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1480
1	13.24	2.59	2.87	21.0	118	2.80	2.69	0.39	1.82	4.32	1.04	2.93	735
1	14.20	1.76	2.45	15.2	112	3.27	3.39	0.34	1.97	6.75	1.05	2.85	1450
1	14.39	1.87	2.45	14.6	96	2.50	2.52	0.30	1.98	5.25	1.02	3.58	1290

Creating functions

```r
create_dataset <- function() {
  data <- read.table(text = "Year Months Flights Delays
                       2016 Jan-Mar 11 6
                       2016 Apr-Jun 12 5
                       2016 Jul-Sep 13 3
                       2016 Oct-Dec 12 5
                       2017 Jan-Mar 10 4
                       2017 Apr-Jun 9 3
                       2017 Jul-Sep 11 4
                       2017 Oct-Dec 25 15
                       2018 Jan-Mar 14 3
                       2018 Apr-Jun 12 5
                       2018 Jul-Sep 13 3
                       2018 Oct-Dec 15 4",
                       header = TRUE,sep = "")
  data$OnTime <- data$Flights - data$Delays
  data$Perc <- round(100 * data$Delays / data$Flights)
  return(data)
}

data <- create_dataset()
head(data)
```

Year	Months	Flights	Delays	OnTime	Perc
2016	Jan-Mar	11	6	5	55
2016	Apr-Jun	12	5	7	42
2016	Jul-Sep	13	3	10	23
2016	Oct-Dec	12	5	7	42
2017	Jan-Mar	10	4	6	40
2017	Apr-Jun	9	3	6	33

Pipelines

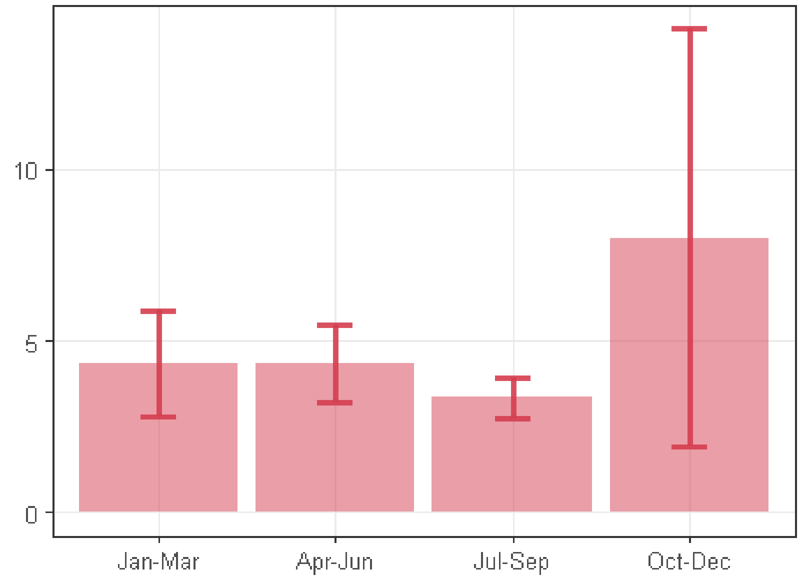
```r
loadlibrary("dplyr")

data_sd <- create_dataset() %>%
  select(variable=Months, value=Delays) %>%
  group_by(variable) %>%
  summarize(sd = sd(value), value = mean(value))

data_sd$variable <- factor(data_sd$variable,
    levels = c('Jan-Mar','Apr-Jun','Jul-Sep','Oct-Dec'))

head(data_sd)
```

variable	sd	value
Apr-Jun	1.1547005	4.333333
Jan-Mar	1.5275252	4.333333
Jul-Sep	0.5773503	3.333333
Oct-Dec	6.0827625	8.000000

The **dplyr** is an important package to know
Pipeline dataset %>% operators %>% first parameter of functions is implicit from the pipeline

The ggplot graphics

```
loadlibrary("RColorBrewer")

col_set <- brewer.pal(11, 'Spectral')

grf <- plot.bar(data_sd, colors=col_set[2], alpha=0.5)
grf <- grf + geom_errorbar(
    aes(x=variable, ymin=value-sd, ymax=value+sd),
    width=0.2, colour=col_set[2], alpha=0.9, size=1.1)

plot(grf)
```



RColorBrewer is a nice package to setup colors
GGPlot is a nice tool to plot graphics

The melt function

```
loadlibrary("reshape")
adjust_dataset <- function(data) {
  data <- melt(data[,c('Year', 'Months', 'Flights', 'Delays', 'OnTime', 'Perc')],
               id.vars = c(1,2))
  data$x <- sprintf("%d-%s", data$Year, data$Months)
  data$x <- factor(data$x,levels = data$x[1:12])
  return(data)
}
data <- create_dataset()
head(data)
data <- adjust_dataset(data)
head(data)
```

Year	Months	Flights	Delays	OnTime	Perc
2016	Jan-Mar	11	6	5	55
2016	Apr-Jun	12	5	7	42
2016	Jul-Sep	13	3	10	23
2016	Oct-Dec	12	5	7	42
2017	Jan-Mar	10	4	6	40
2017	Apr-Jun	9	3	6	33

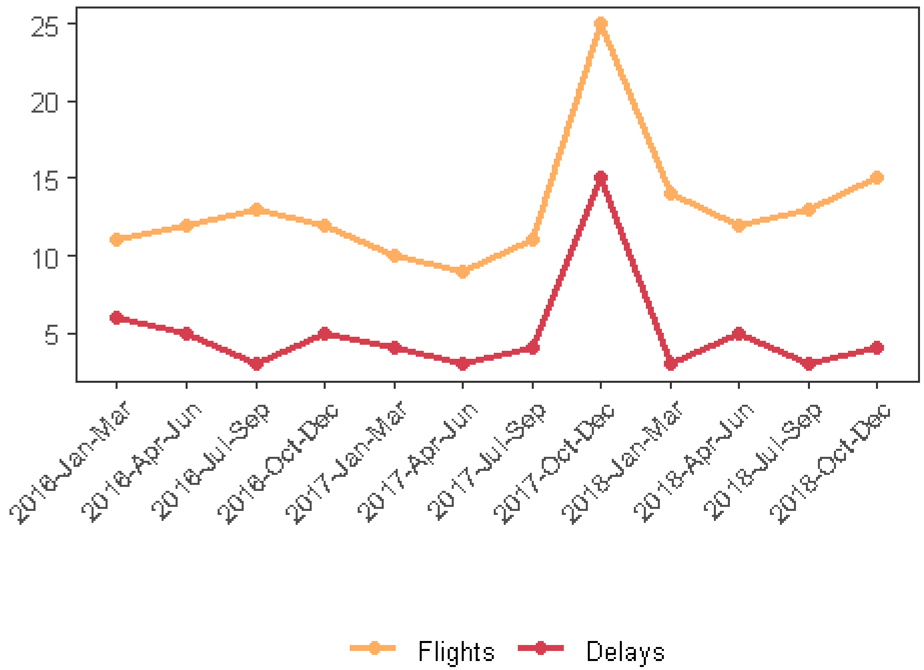
Year	Months	variable	value	x
2016	Jan-Mar	Flights	11	2016-Jan-Mar
2016	Apr-Jun	Flights	12	2016-Apr-Jun
2016	Jul-Sep	Flights	13	2016-Jul-Sep
2016	Oct-Dec	Flights	12	2016-Oct-Dec
2017	Jan-Mar	Flights	10	2017-Jan-Mar
2017	Apr-Jun	Flights	9	2017-Apr-Jun

The **melt** function transforms columns values into rows grouped by **id.vars**.

The name of columns is used to fill the **variable** attribute created during the **melt**.

Line graphics

```
grf <- plot.series(data %>% filter(variable %in% c('Flights', 'Delays')),
                   colors=col_set[c(4,2)])
grf <- grf + theme(axis.text.x = element_text(angle=45, hjust=1))

plot(grf)
```



Take some time studying myGraphics.ipynb

Joining data frames

```
stores <- data.frame(
    city = c("Rio de Janeiro", "Sao Paulo", "Paris", "New York", "Tokyo"),
    value = c(10, 12, 20, 25, 18))
head(stores)


divisions <- data.frame(
    city = c("Rio de Janeiro", "Sao Paulo", "Paris", "New York", "Tokyo"),
    country = c("Brazil", "Brazil", "France", "US", "Japan"))
head(divisions)

data <- merge(stores, divisions, by.x="city", by.y="city")
head(data)

result <- data %>% group_by(country) %>% summarize(count = n(), amount = sum(value))
head(result)
```

city	value
Rio de Janeiro	10
Sao Paulo	12
Paris	20
New York	25
Tokyo	18

city	country
Rio de Janeiro	Brazil
Sao Paulo	Brazil
Paris	France
New York	US
Tokyo	Japan

city	value	country
New York	25	US
Paris	20	France
Rio de Janeiro	10	Brazil
Sao Paulo	12	Brazil
Tokyo	18	Japan

country	count	amount
Brazil	2	22
France	1	20
Japan	1	18
US	1	25

Loops and Conditional

- R supports loops and conditionals in a similar way as in Java
- Loops should be used when strictly needed

```
for (i in 1:nrow(result)) {
  value <- result$amount[i]
  if (result$count[i] > 1) {
      value <- 0.8*value
  }
  print(sprintf("%6s - %.1f", result$country[i], value))
}
```

```
[1] "Brazil - 17.6"
[1] "France - 20.0"
[1] " Japan - 18.0"
[1] "    US - 25.0"
```

Practicing

- Take some time to practice the examples
  - https://nbviewer.jupyter.org/github/eogasawara/mylibrary/blob/master/myIntroduction.ipynb
- Take a look at how to prepare nice graphics using ggplot2
  - https://nbviewer.jupyter.org/github/eogasawara/mylibrary/blob/master/myGraphics.ipynb

Exploratory analysis

Types of Data Sets

- Record
  - Relational datasets
- Matrix
  - numerical matrix, crosstabs
- Documents
  - texts, term-frequency vector
- Transactions
- Graph and network
  - World Wide Web
  - Social or information networks
- Ordered
  - Temporal data: time-series
  - Sequential data: transaction sequences
- Spatial, image, and multimedia
  - Spatial data: maps
  - Images
  - Videos

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

Documents	team	coach	play	ball	score	game	win	lost	timeout	season
Document 1	3	0	5	0	2	6	0	2	0	2
Document 2	0	7	0	2	1	0	0	3	0	0
Document 3	0	1	0	0	1	2	2	0	3	0

TID	Items
1	Bread, Coke, Milk
2	Beer, Bread
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Coke, Diaper, Milk

Data	PIB - R$ (milhões)
1990.01	0.2
1990.02	0.4
1990.03	0.8
1990.04	0.7
1990.05	0.8

Important Characteristics of Structured Data

- Dimensionality
  - Curse of dimensionality
- Sparsity
  - Only presence counts
- Resolution
  - Patterns depend on the scale
- Distribution
  - Centrality and dispersion

Relational data

- Data sets are made up of data objects
- A data object represents an entity
  - sales database: customers, store items, sales
  - medical database: patients, treatments, illness
  - university database: students, professors, courses
- Attributes describe data objects
- Database
  - rows -> data objects (tuples)
  - columns -> attributes

Attributes

- Attribute (or dimensions, features, variables)
  - a data field, representing a characteristic or feature of a data object
  - E.g., customer _ID, name, address
- Types
  - Nominal
  - Binary
  - Ordinal
  - Numeric

Attribute Types

- Nominal: categories, states, or “names of things”
  - Hair_color = {auburn, black, blond, brown, grey, red, white}
  - marital status, occupation, ID numbers, zip codes
- Binary
  - Attribute with only two states (0 and 1)
  - Symmetric binary: both outcomes equally important
    - e.g., gender
  - Asymmetric binary: outcomes not equally important
    - e.g., medical test (positive vs. negative)
    - Convention: assign 1 to the most important outcome (e.g., HIV positive)
- Ordinal
  - Values have a meaningful order (ranking), but magnitude between successive values is not known
  - Size = {small, medium, large}, grades, army rankings

Numeric Attribute Types

- Quantity (integer or real-valued)
- Interval
  - Measured on a scale of equal-sized units
  - Values have order
    - E.g., the temperature in C˚or F˚, calendar dates
  - No true zero-point
- Ratio
  - Inherent zero-point
  - We can speak of values as being an order of magnitude larger than the unit of measurement (10 K˚ is twice as high as 5 K˚).
    - e.g., the temperature in Kelvin, length, counts, monetary quantities

Discrete vs. Continuous Attributes

- Discrete Attribute
  - Has only a finite or countably infinite set of values
  - Sometimes, represented as integer variables
- Continuous Attribute
  - Has real numbers as attribute values
    - E.g., temperature, height, or weight
  - Practically, real values can only be measured and represented using a finite number of digits
  - Continuous attributes are typically represented as floating-point variables

Iris Dataset

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
numeric	numeric	numeric	numeric	factor

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
51	7.0	3.2	4.7	1.4	versicolor
52	6.4	3.2	4.5	1.5	versicolor
53	6.9	3.1	4.9	1.5	versicolor
101	6.3	3.3	6.0	2.5	virginica
102	5.8	2.7	5.1	1.9	virginica
103	7.1	3.0	5.9	2.1	virginica

Basic Statistical Descriptions of Data

- Motivation
  - To better understand the data:
    - central tendency, variation and spread
- Data centrality and dispersion characteristics
  - median, max, min, quantiles, outliers, variance
- Numerical dimensions correspond to sorted intervals
  - Boxplot or quantile analysis on sorted intervals

Descriptive Measures

- Centrality
  - Mean (algebraic measure)
    - $\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n}$
  - Median
    - Middle value if an odd number of values, or weighted average of the middle two values otherwise
  - Mode
    - The value that occurs most frequently in the data
    - Unimodal, bimodal, trimodal
    - Empirical formula:
      - $mean - mode = 3 \cdot (mean - median)$
- Dispersion
  - Variance and standard deviation
    - Variance: (algebraic, scalable computation)
    - Standard deviation (σ): square root of the variance (σ^2)
      - $\sigma^2 = \frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n} = \frac{\sum_{i=1}^{n} {x_i}^2}{n} - \mu^2$

Measuring the Dispersion of Data

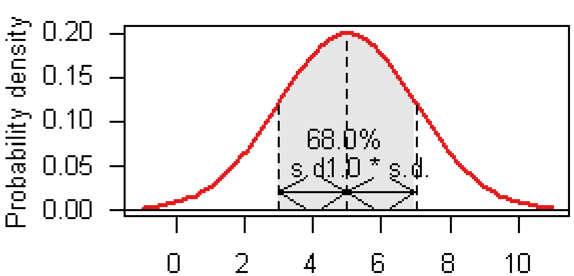
- Quartiles, outliers and boxplots
  - Quartiles: Q_1 (25th percentile), Q_3 (75th percentile)
  - Inter-quartile range: IQR = $Q_3 - Q_1$
  - Five number summary: min, Q_1, median, Q_3, max
  - Boxplot: ends of the box are the quartiles; median is marked; add whiskers, and plot outliers individually

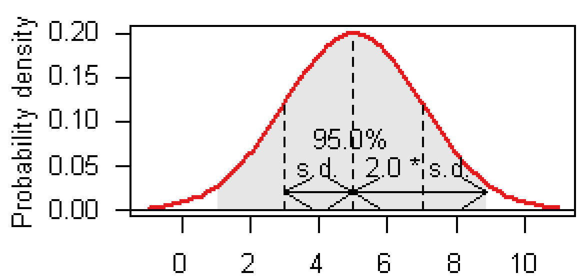
Statistics	Freq
Min.	4.300000
1st Qu.	5.100000
Median	5.800000
Mean	5.843333
3rd Qu.	6.400000
Max.	7.900000

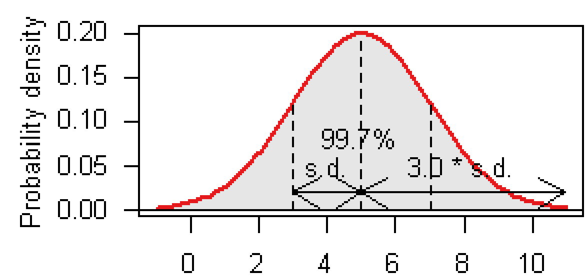
```
[1] "IQR=1.3"
```

Properties of Normal Distribution Curve

- The normal (distribution) curve
  - From μ–σ to μ+σ: contains about 68% of the measurements (μ: mean, σ: standard deviation)
  - From μ–2σ to μ+2σ: contains about 95% of it
  - From μ–3σ to μ+3σ: contains about 99.7% of it

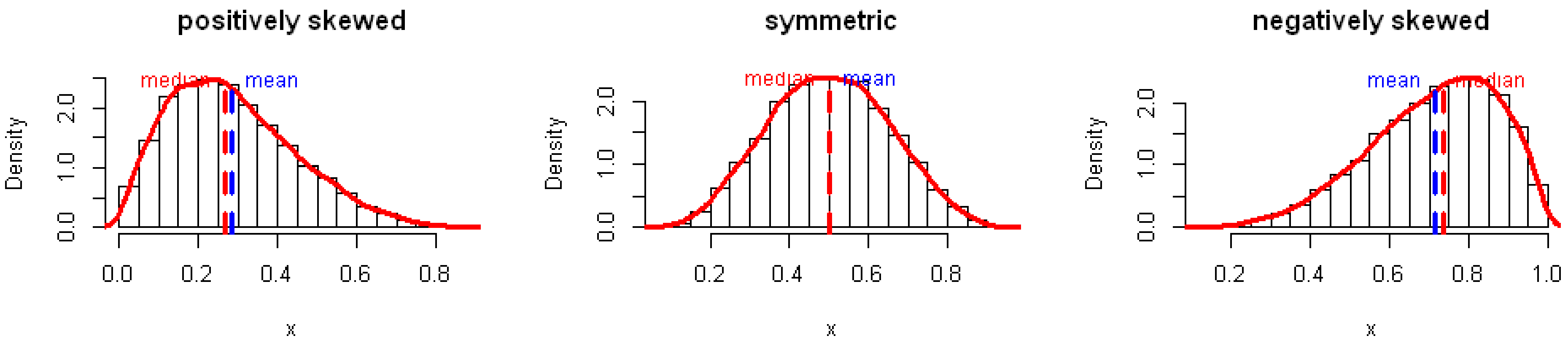




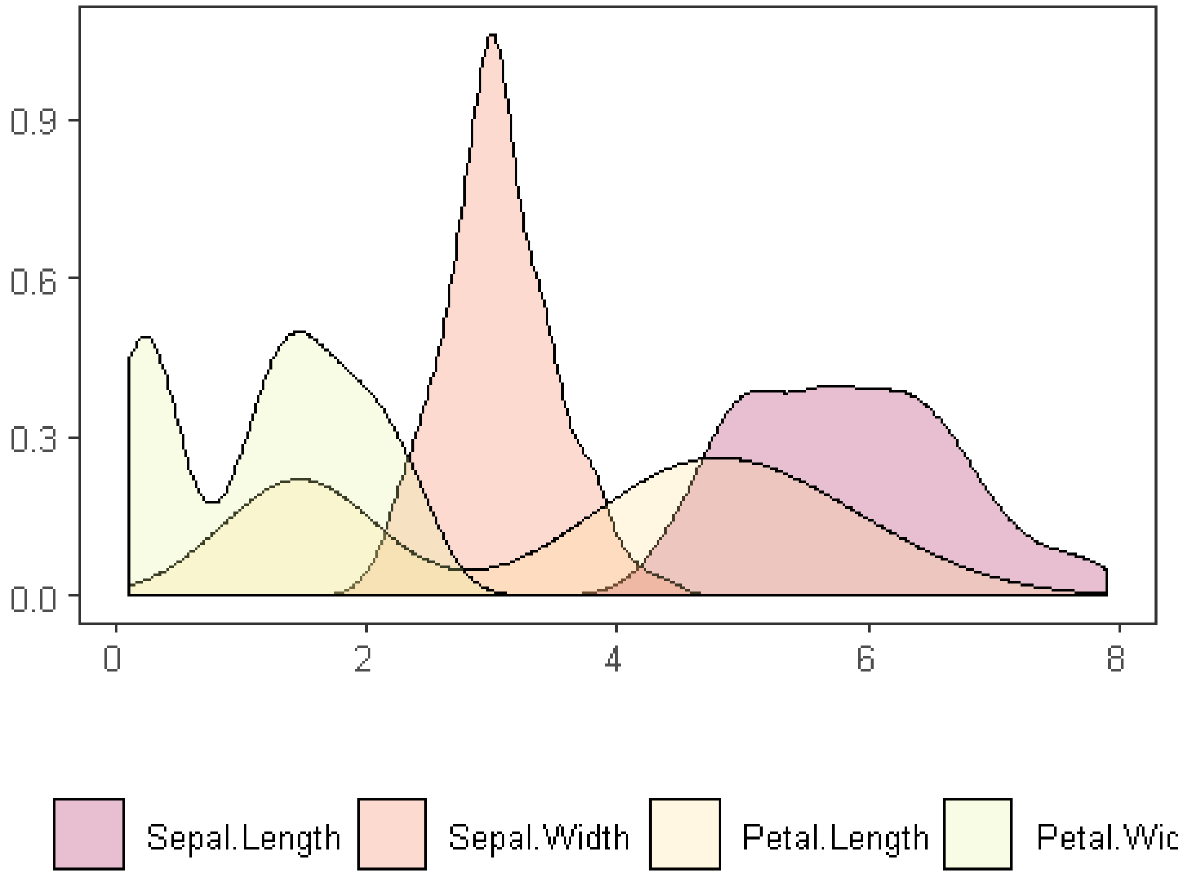


Symmetric vs. Skewed Data

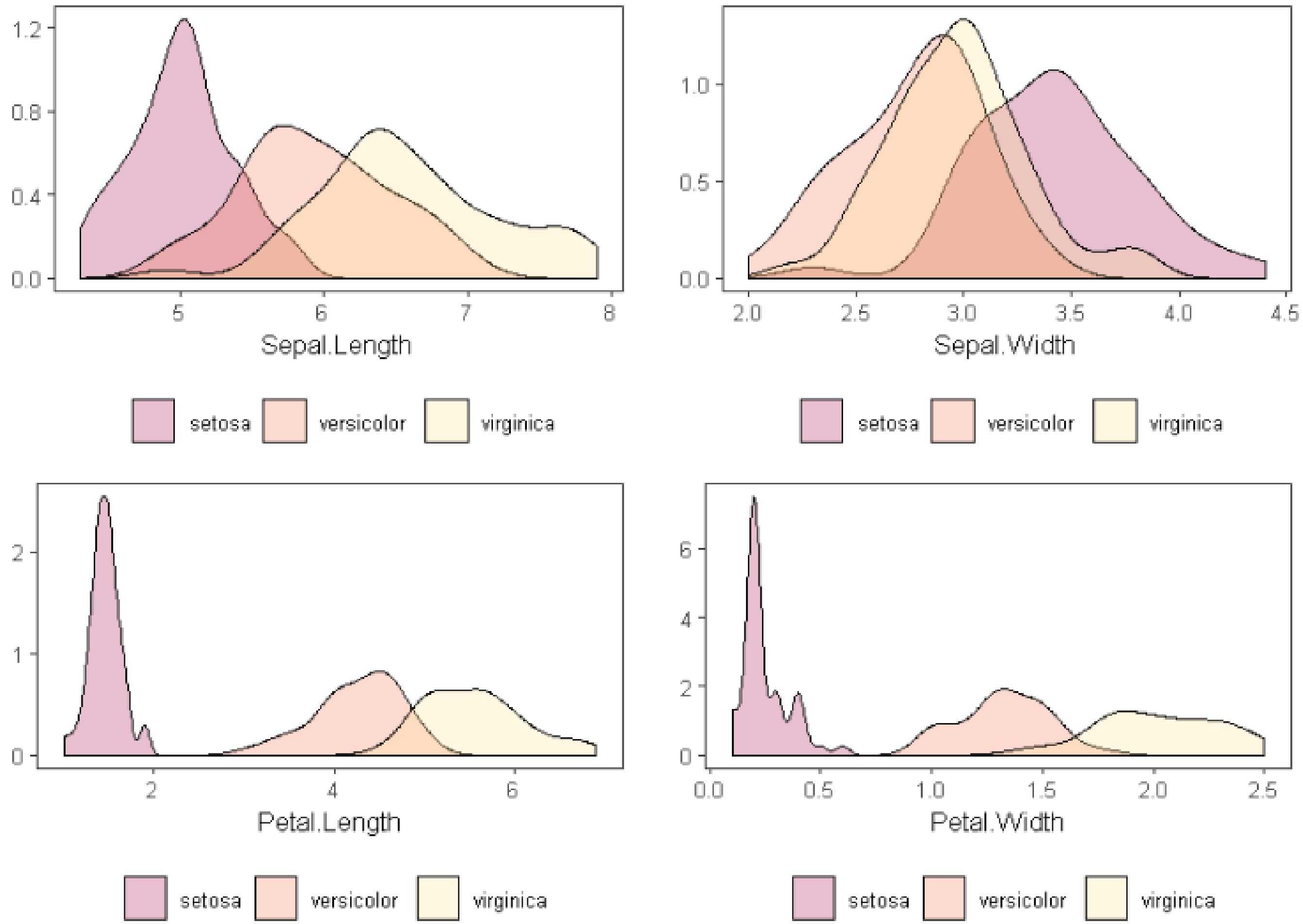
- Median and mean for:
  - positive, symmetric, and negatively skewed data



Probability density function



Density distributions per class label

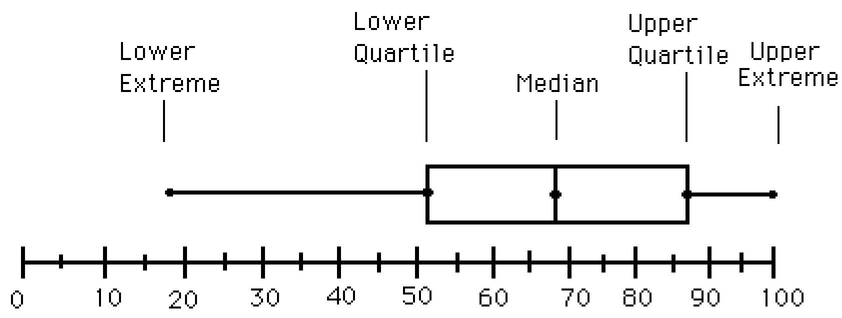


Graphic Displays of Basic Statistical Descriptions

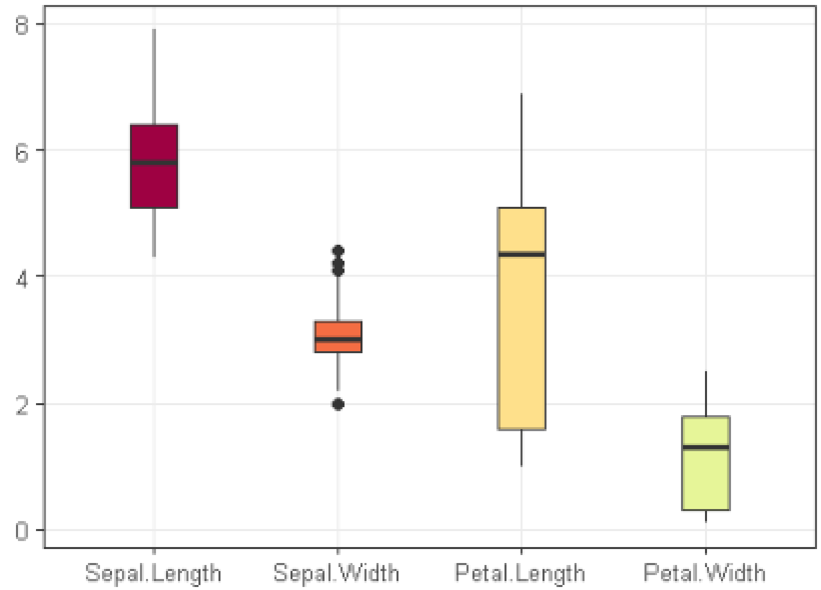
- Boxplot
- Histogram
- Quantile-quantile (q-q) plot
- Scatter plot

Boxplot Analysis

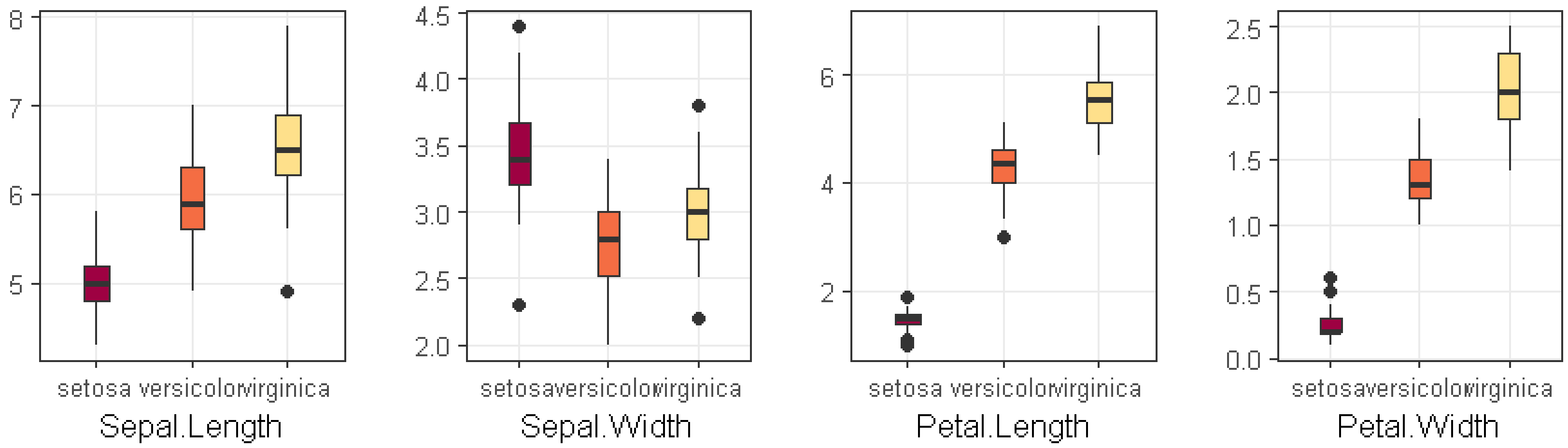
- Five-number summary of a distribution
  - Min., Q1, Median, Q3, Max.
- Boxplot
  - Data is represented with a box
  - The ends of the box are at the first and third quartiles, i.e., the height of the box is IQR
  - A line within the box marks the median
  - Whiskers: two lines outside the box extended to Minimum and Maximum
  - Outliers are values:
    - higher than Q3 + 1.5 x IQR
    - lower than Q1 - 1.5 x IQR



Boxplot for all variables

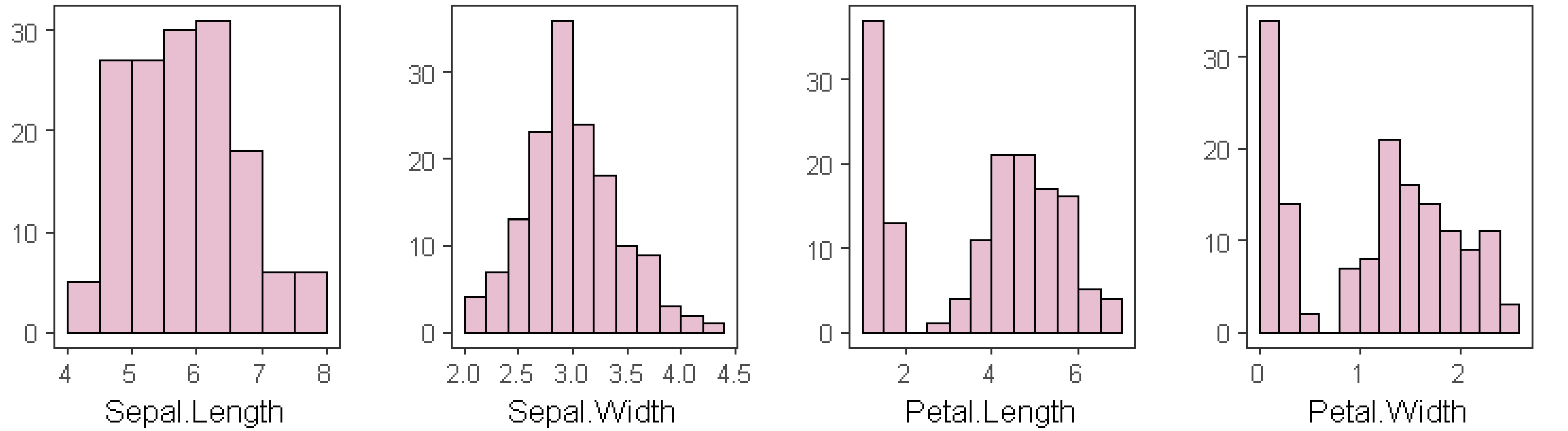


Boxplot per class label



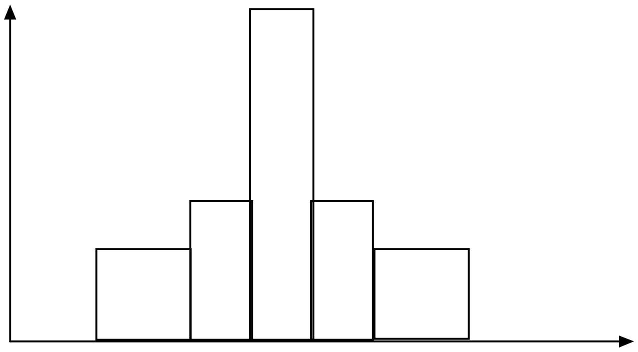
Histogram Analysis

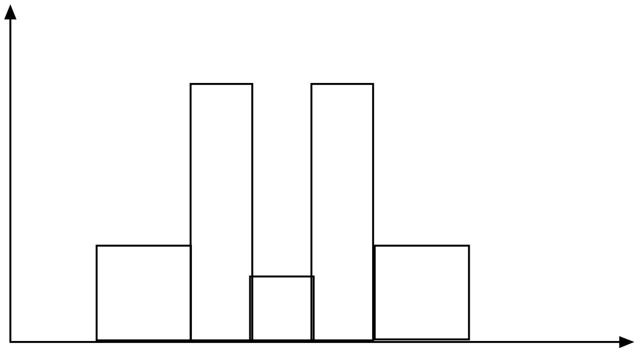
- The histogram displays values of tabulated frequencies
- It shows what proportion of cases into each category
- The area of the bar that denotes the value
  - It is a crucial property when the categories are not of uniform width
- The categories specify non-overlapping intervals of some variable
- The categories (bars) must be adjacent



Histograms may tell more than Boxplots

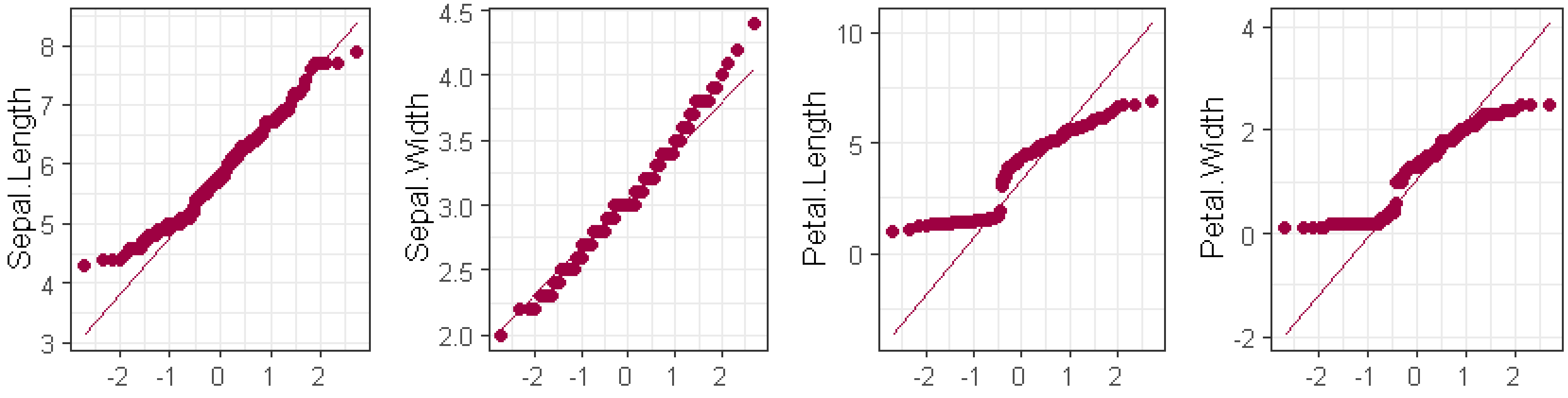
- The two histograms shown in the left may have the same boxplot representation
  - The same values for min, Q1, median, Q3, max
- However, they have rather different data distributions





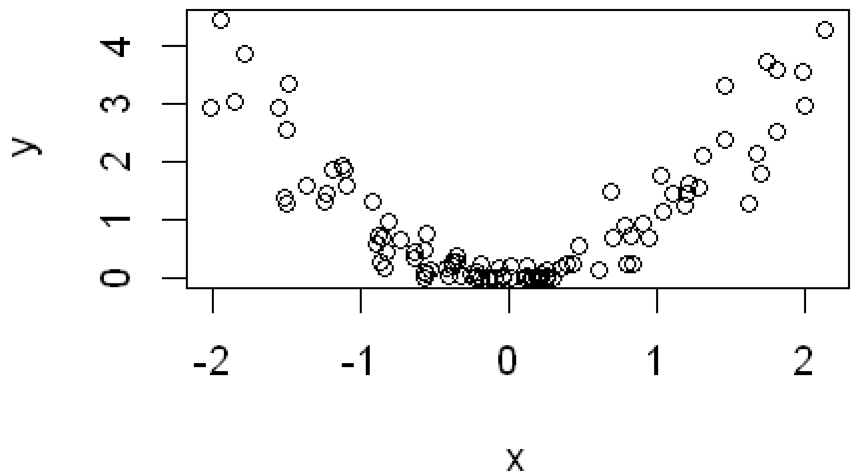
Quantile-Quantile (Q-Q) Plot

- Graphs the quantiles of one univariate distribution against the corresponding quantiles of another (theoretical distribution)
- A good approach to visual inspect if the distribution is similar to a standard normal

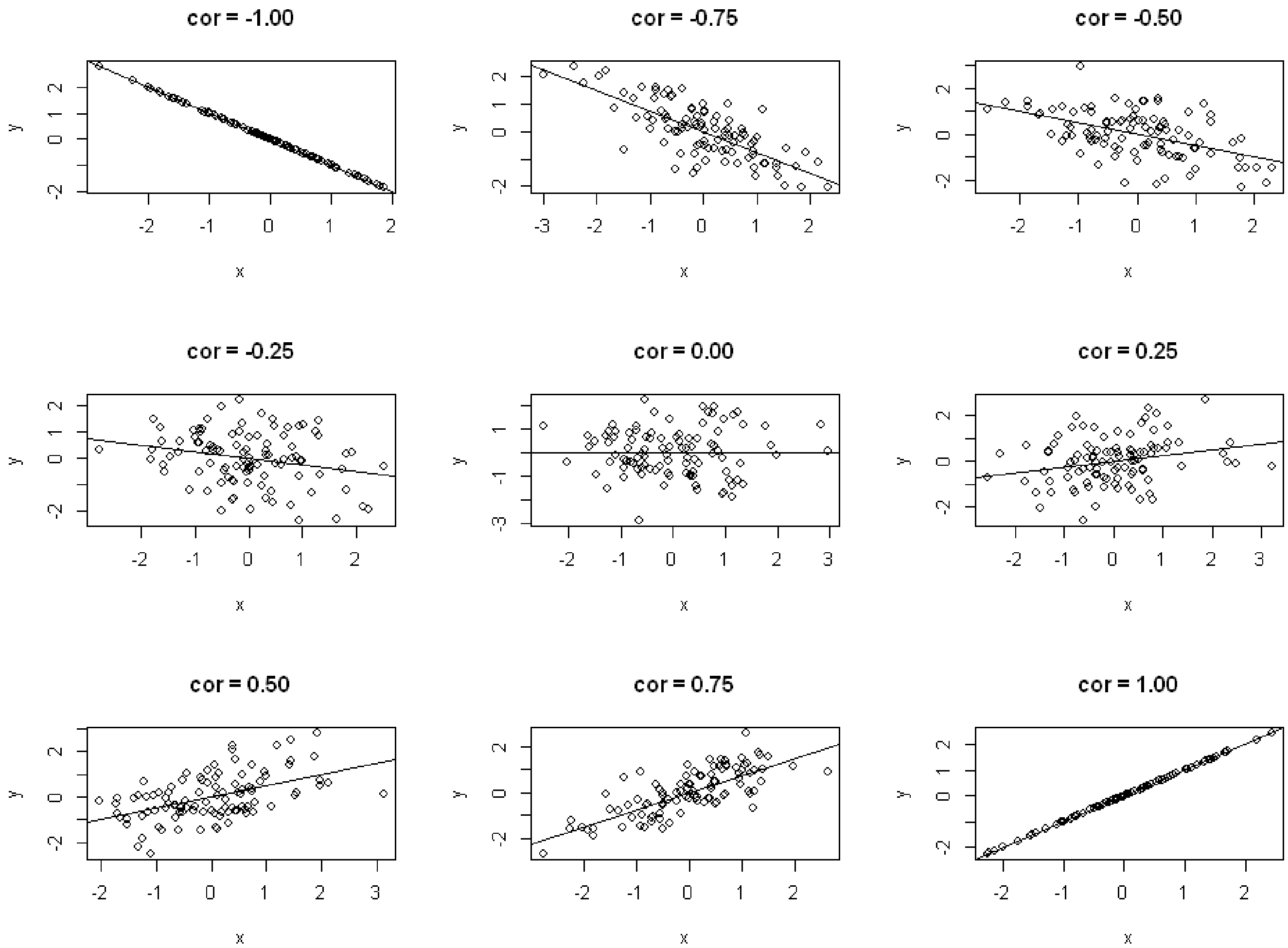


Scatter plot

- Provides the first look at bivariate data to see clusters of points, outliers
- Each pair of values is treated as a pair of coordinates and plotted as points in the plane



Data correlation



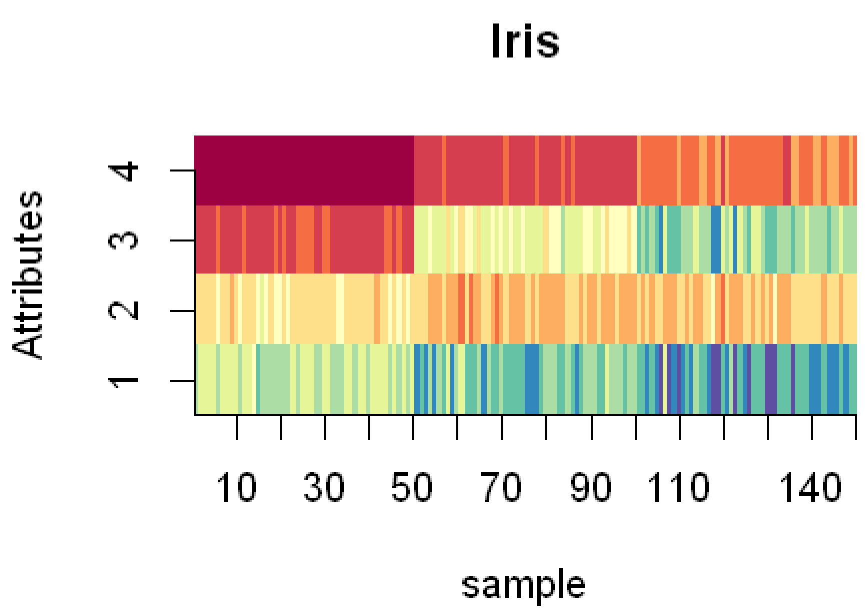
The first row presents negatively correlated data
The second row presents uncorrelated data
The third row presents positively correlated data

Data Visualization

- Why data visualization?
  - Gain insight into an information space by mapping data onto graphical primitives
  - Provide a qualitative overview of large data sets
  - Search for patterns, trends, structure, irregularities, relationships among data
  - Help find interesting regions and suitable parameters for further quantitative analysis
  - Provide visual proof of computer representations derived
- Categorization of visualization methods:
  - Pixel-oriented visualization techniques
  - Geometric projection visualization techniques
  - Icon-based visualization techniques
  - Hierarchical visualization techniques
  - Visualizing complex data and relations

Pixel-Oriented Visualization Techniques

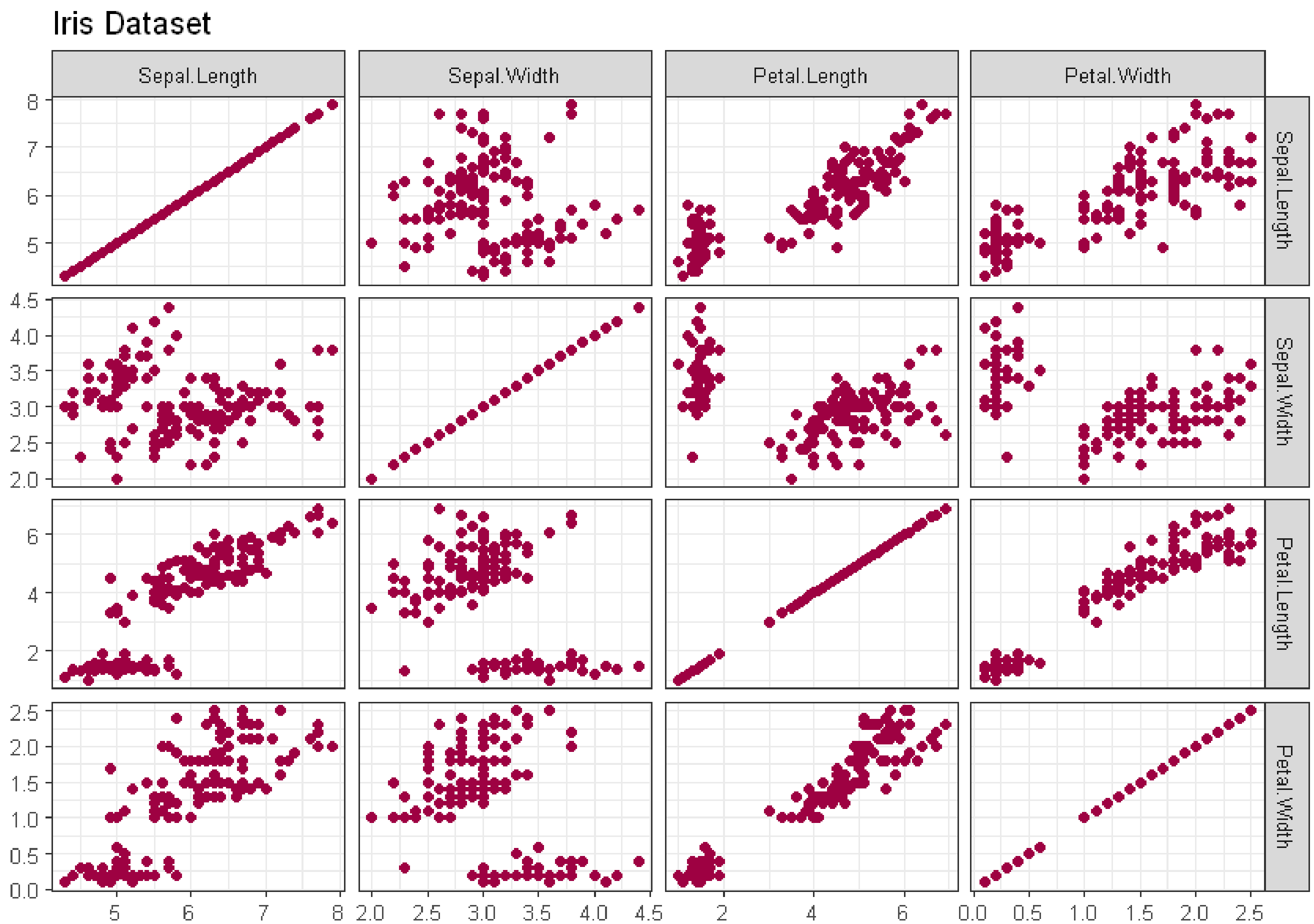
- For a data set of m dimensions, create m windows on the screen, one for each dimension
- The m dimension values of a record are mapped to m pixels at the corresponding positions in the windows
- The colors of the pixels reflect the corresponding values



Geometric Projection Visualization Techniques

- Visualization of geometric transformations and projections of the data
- Methods
  - Direct visualization
  - Scatterplot and scatterplot matrices
  - Landscapes
  - Parallel coordinates

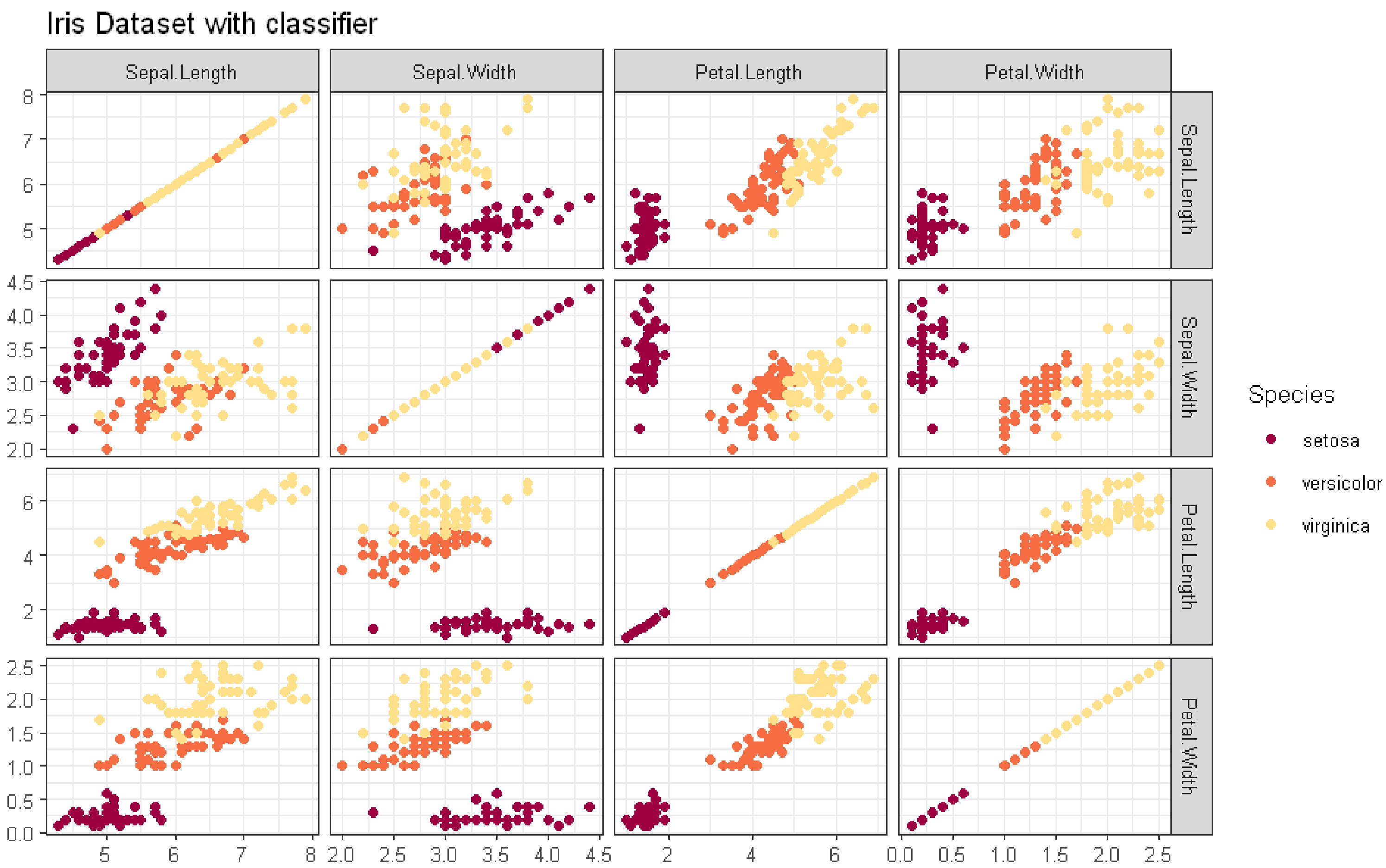
Scatterplot Matrices



A matrix of scatterplots (x-y-diagrams)

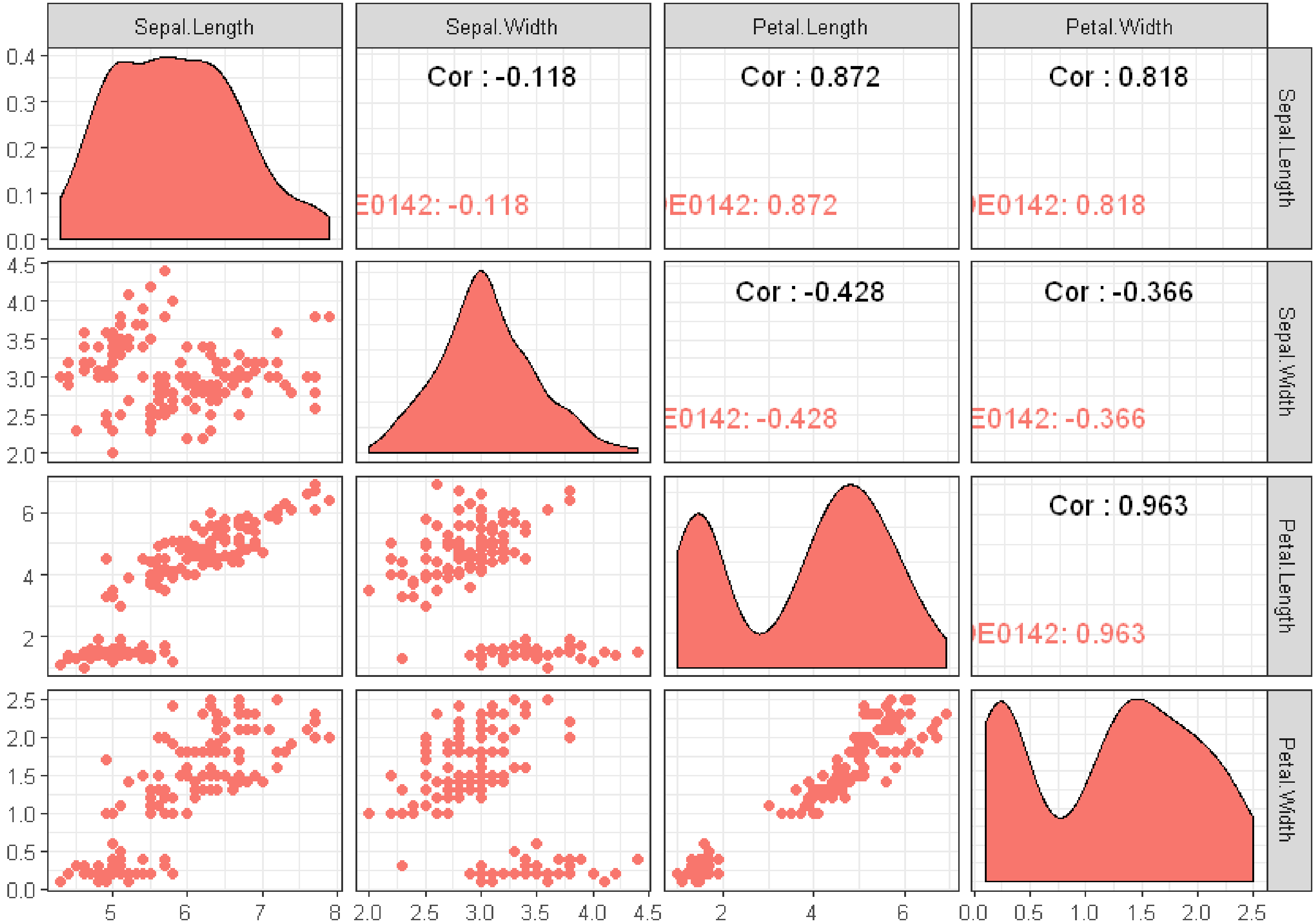
k-dimensional data: total of ($k^2/2$-k) scatterplots]

Scatterplot matrices with a class label

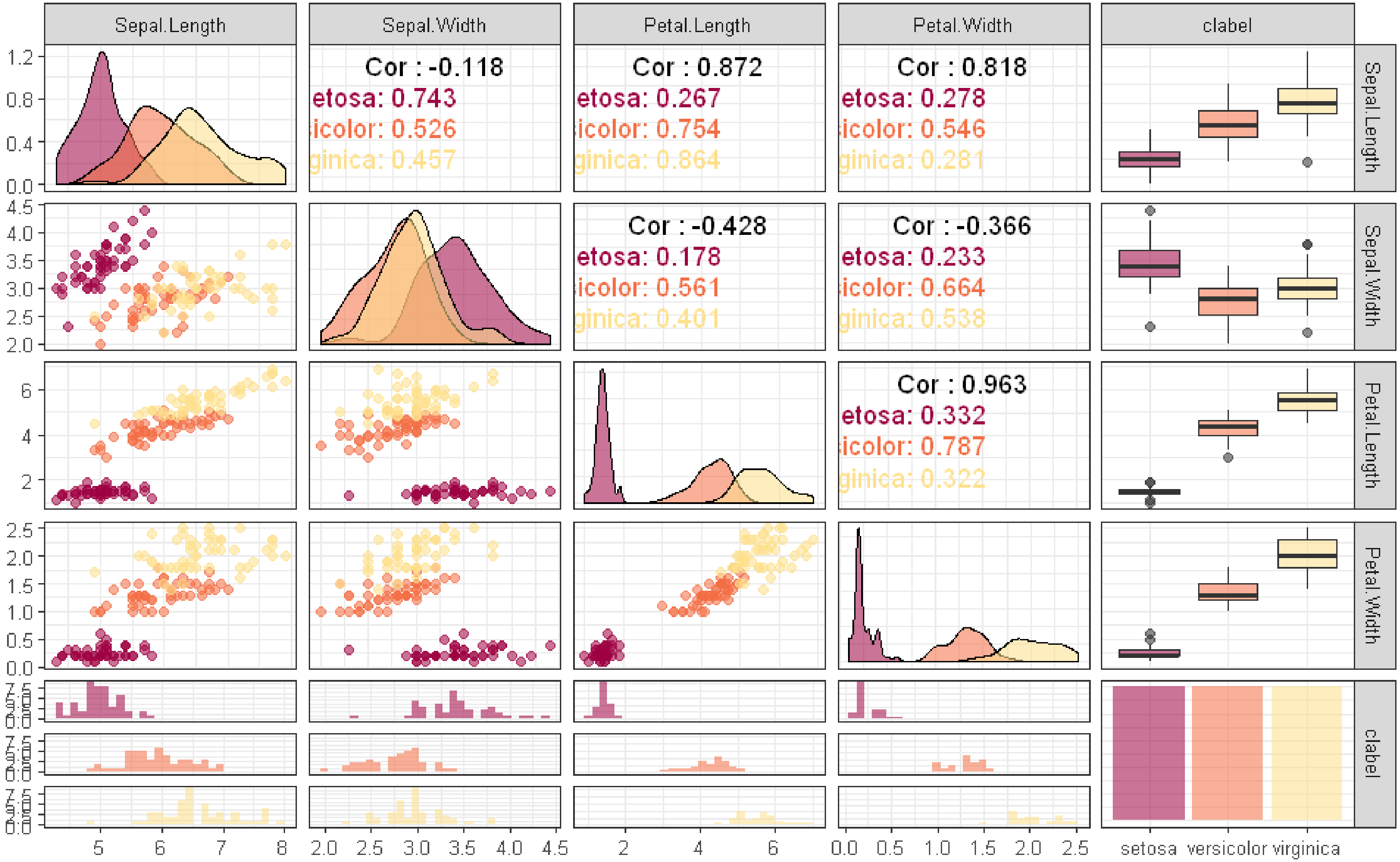


Advanced Matrices Plot

- The matrix of optimized plots of the k-dim. data



Advanced Matrices Plot with a class label



Landscapes

- Visualization of the data as perspective landscape
- The data needs to be transformed into a (possibly artificial) 2D spatial representation which preserves the characteristics of the data

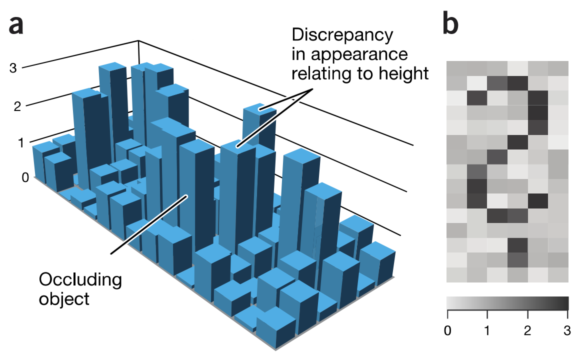


Figure 2 | Three-dimensional representation of abstract data. (**a**) Data occlusion and interference of visual encodings with depth cues can be problematic in three-dimensional space. (**b**) The same data as in **a** plotted as a two-dimensional heat map.

NATURE METHODS | VOL.9 NO.9 | SEPTEMBER 2012 | 851

Parallel Coordinates of a Data Set

Icon-Based Visualization Techniques

- Visualization of the data values as features of icons
- Typical visualization methods
  - Chernoff Faces
  - Salience
- General techniques
  - Shape coding: Use shape to represent certain information encoding
  - Color icons: Use color icons to encode more information
  - Tile bars: Use small icons to represent the relevant feature vectors in document retrieval

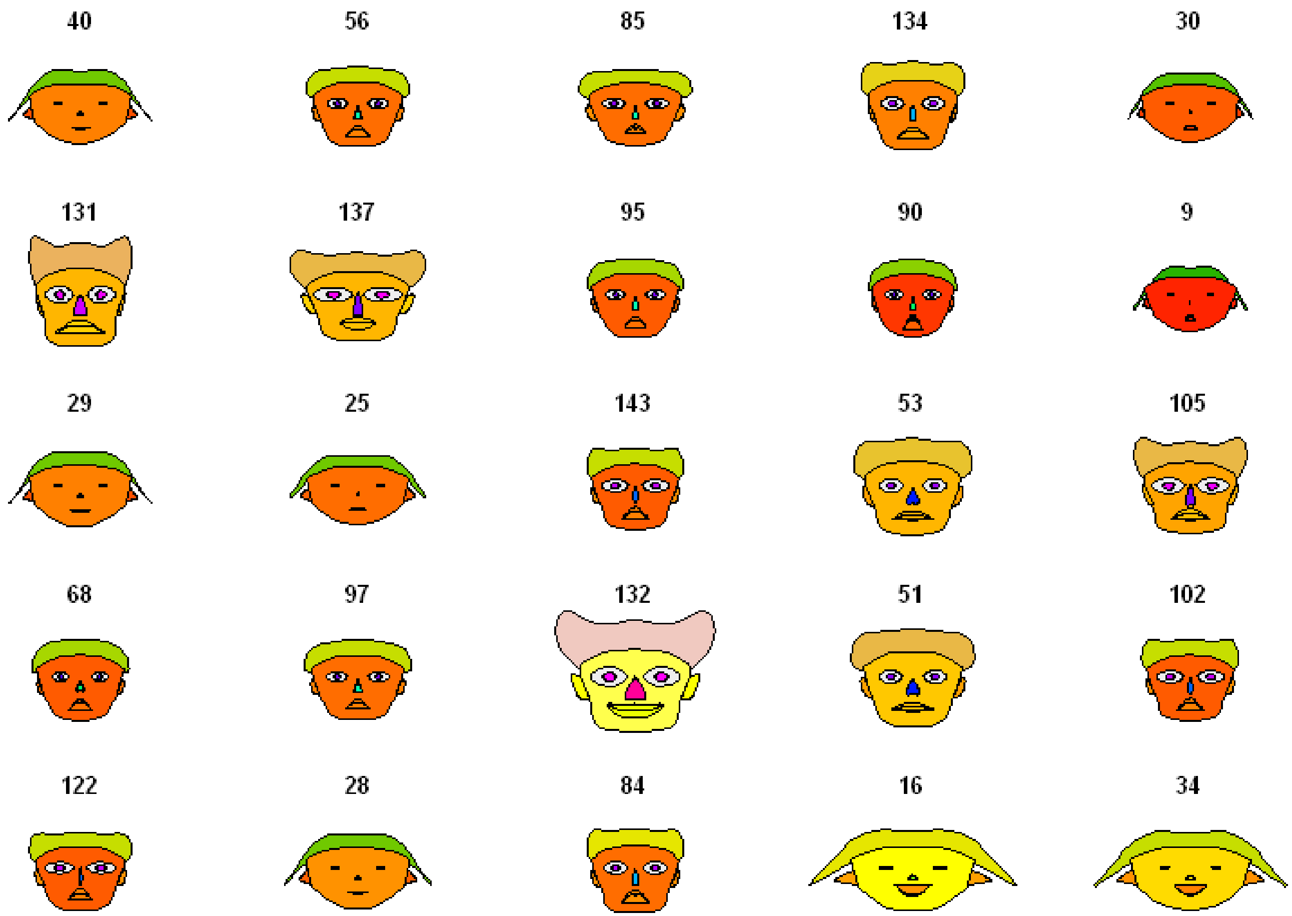
Chernoff Faces

- A way to display variables on a two-dimensional surface
  - Let x be eyebrow slant, y be eye size, z be nose length
- The figure shows faces produced using ten characteristics: head eccentricity, eye size, eye spacing, eye eccentricity, pupil size, eyebrow slant, nose size, mouth shape, mouth size, and mouth opening):
  - Each assigned one of 10 possible values

Gonick, L. and Smith, W. The Cartoon Guide to Statistics. New York: Harper Perennial, p. 212, 1993

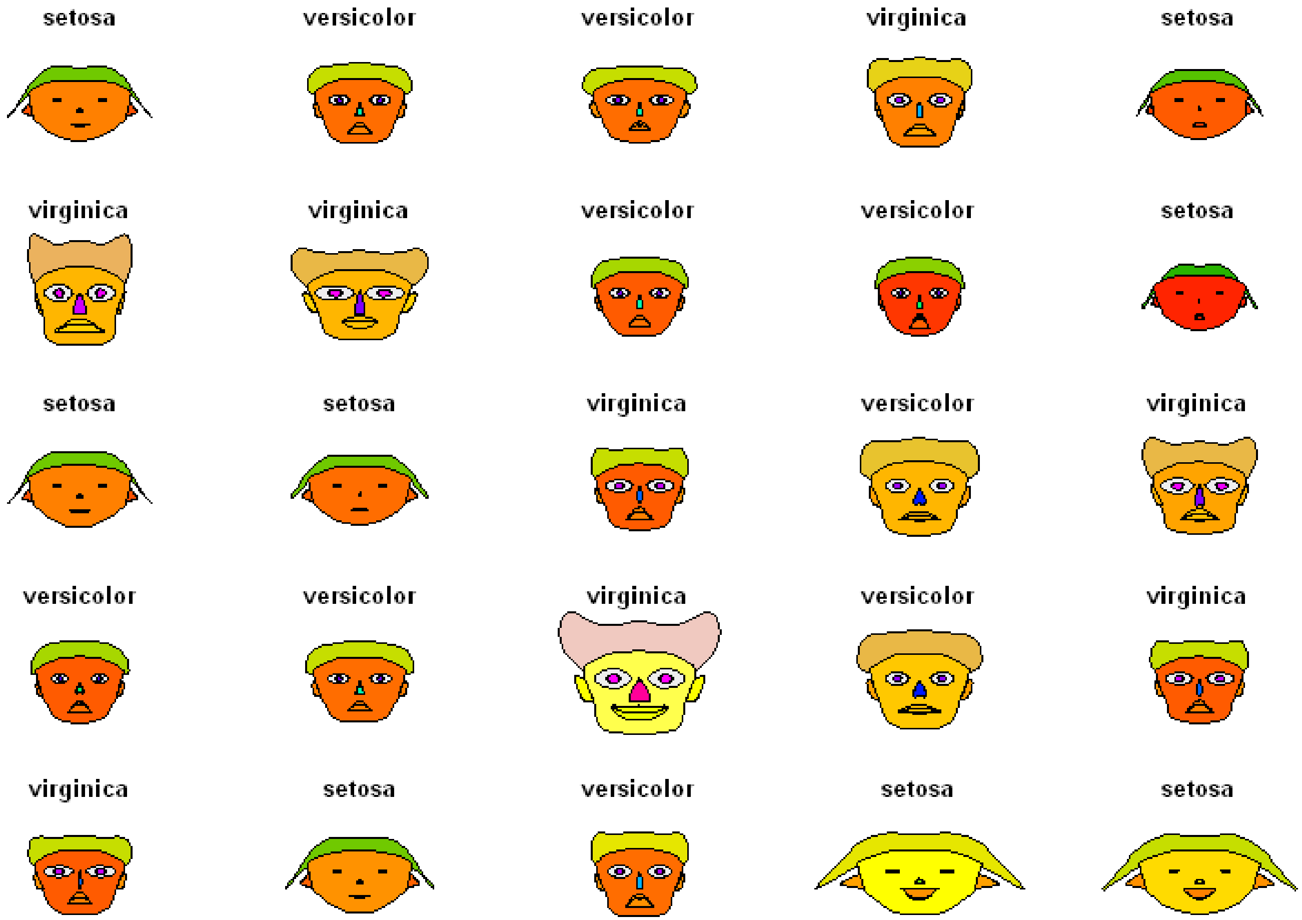
Weisstein, Eric W. "Chernoff Face." From MathWorld -A Wolfram Web Resource. mathworld.wolfram.com/ChernoffFace.html

Chernoff Faces example with the Iris dataset



Can you see any pattern?

Chernoff Faces example with the Iris dataset



Salience

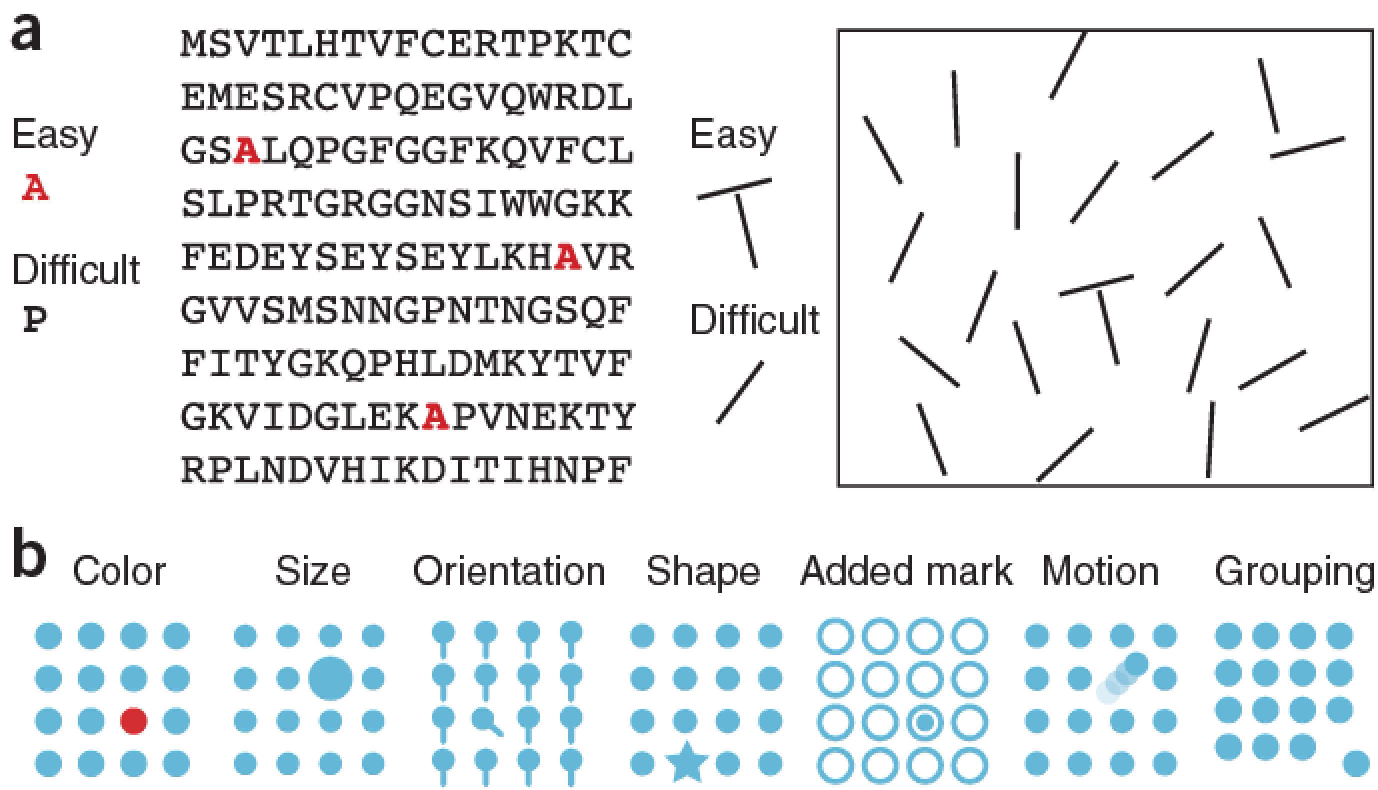


Figure 1 | Salience through visual features. (**a**) Certain elements can be seen in a single glance, whereas others are difficult to find. (**b**) Examples of visual features that make objects distinct.

NATURE METHODS | VOL.7 NO.10 | OCTOBER 2010 | **773**

Practicing

- Take some time to practice the examples
  - https://nbviewer.jupyter.org/github/eogasawara/mylibrary/blob/master/myExploratoryAnalysis.ipynb
- Learn to use Jupyter with R
  - http://jupyter.org

Data Preprocessing

Data Quality: Why Preprocess the Data?

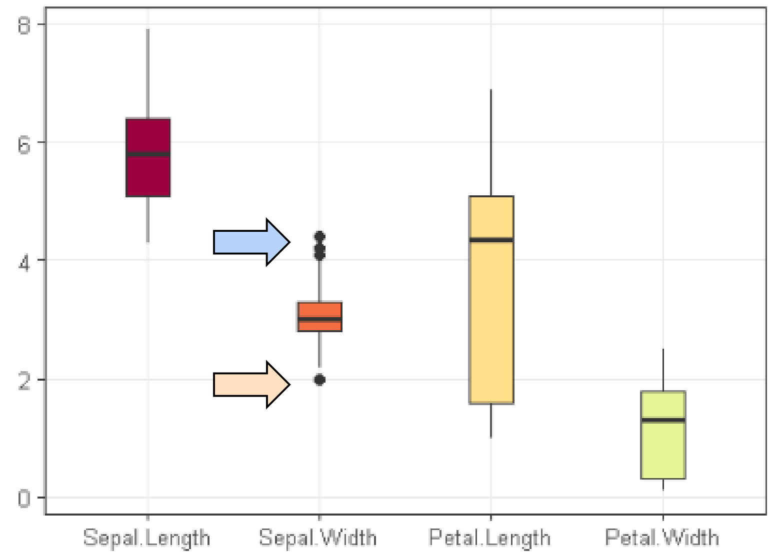
- Measures for data quality: A multidimensional view
  - Accuracy: correct or wrong, accurate or not
  - Completeness: not recorded, unavailable, …
  - Consistency: some modified but some not, dangling, …
  - Timeliness: timely update?
  - Believability: how trustable the data are correct?
  - Interpretability: how easily the data can be understood?

Major Tasks in Data Preprocessing

- Data cleaning
  - Fill in missing values, smooth noisy data, identify or remove outliers, and resolve inconsistencies
- Data integration
  - Integration of multiple databases, data cubes, or files
- Data reduction
  - Dimensionality reduction
  - Numerosity reduction
  - Data compression
- Data transformation and data discretization
  - Normalization
  - Concept hierarchy generation

Outlier removal based on boxplot

- Interval for regular data [Q_1-1.5·IQR, Q_3+1.5·IQR]
  - More conservative interval [Q_1-3·IQR, Q_3+3·IQR]



	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
16	5.7	4.4	1.5	0.4	setosa
33	5.2	4.1	1.5	0.1	setosa
34	5.5	4.2	1.4	0.2	setosa
61	5.0	2.0	3.5	1.0	versicolor

Handling Redundancy in Data Integration

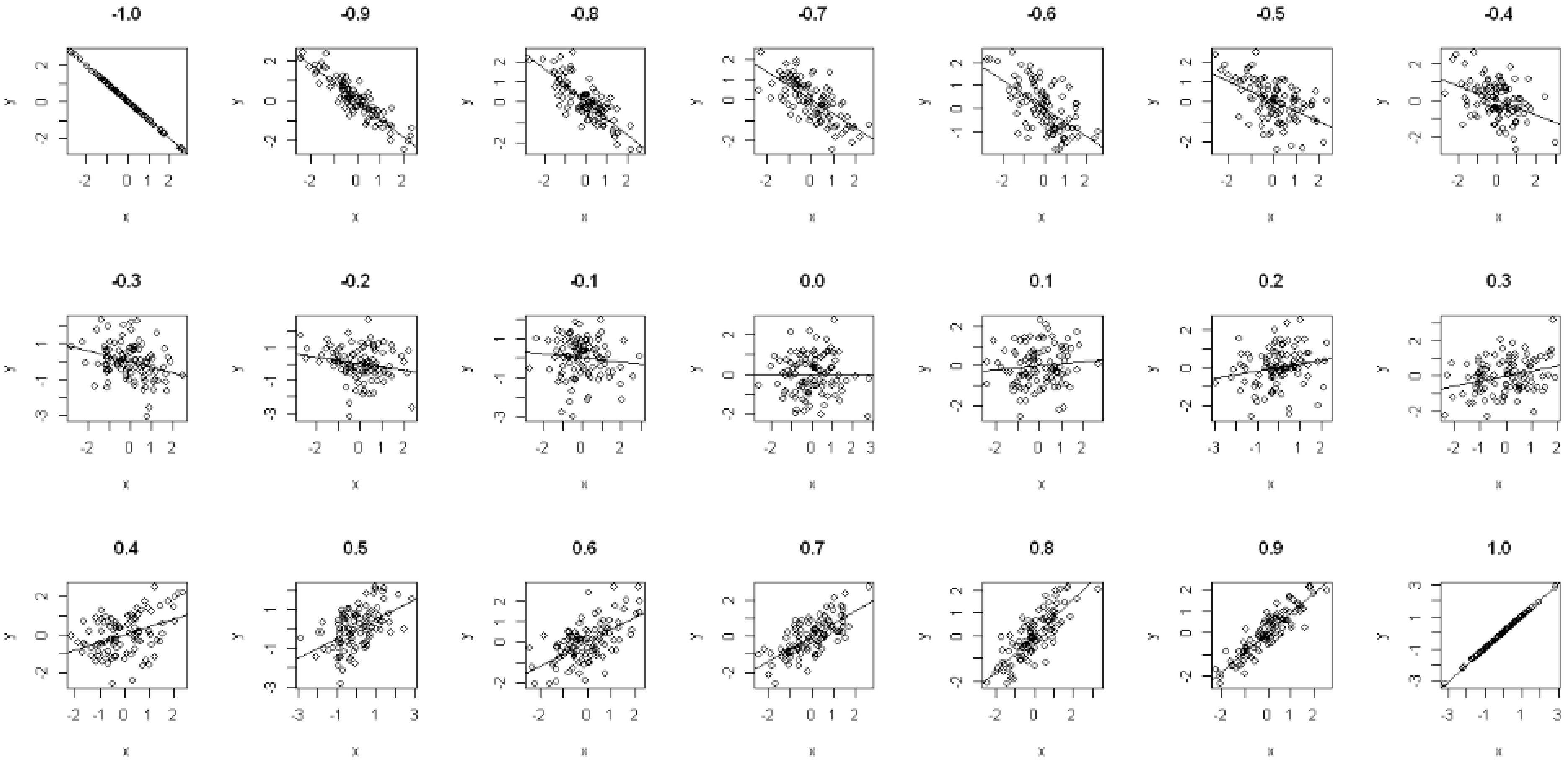
- Redundant data occur often when integration of multiple databases
  - Object identification: The same attribute or object may have different names in different databases
  - Derivable data: One attribute may be a “derived” attribute in another table, e.g., annual revenue
- Redundant attributes may be able to be detected by correlation analysis and covariance analysis
- Careful integration of the data from multiple sources may help reduce/avoid redundancies and inconsistencies and improve mining speed and quality

Correlation Analysis (Numeric Data)

- Correlation coefficient (Pearson's product moment coefficient)
  - $r_{A,B} = \frac{\sum_{i=1}^{n}(a_i - \bar{A})(b_i - \bar{B})}{(n-1)\sigma_A \sigma_B} = \frac{\sum_{i=1}^{n}(a_i b_i) - n\bar{A}\bar{B}}{(n-1)\sigma_A \sigma_B}$

  where n is the number of tuples, and are the respective means of A and B, σ_A and σ_B are the respective standard deviation of A and B, and $\Sigma(a_i b_i)$ is the sum of the AB cross-product.
- If $r_{A,B} > 0$, A and B are positively correlated (A's values increase as B's). The higher, the stronger correlation.
- $r_{A,B} = 0$: independent; $r_{AB} < 0$: negatively correlated

Visually Evaluating Correlation



Scatter plots showing the similarity from –1 to 1

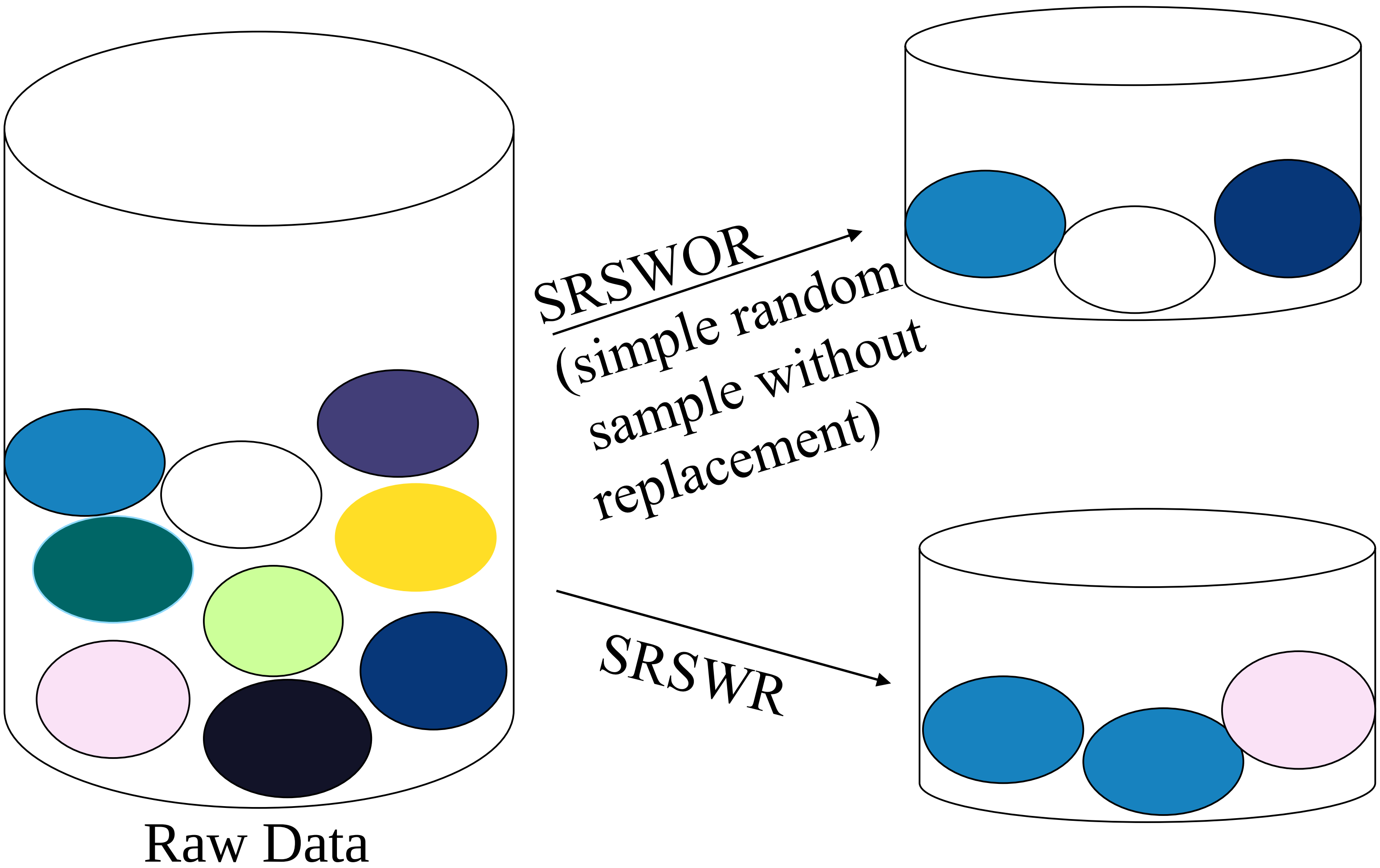
Sampling

- Sampling: obtaining a small sample s to represent the whole data set N
- Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data
- Key principle: Choose a representative subset of the data
  - Simple random sampling may have very poor performance in the presence of skew
  - Develop adaptive sampling methods, e.g., stratified sampling:
- Note: Sampling may not reduce database I/Os (page at a time)

Types of Sampling

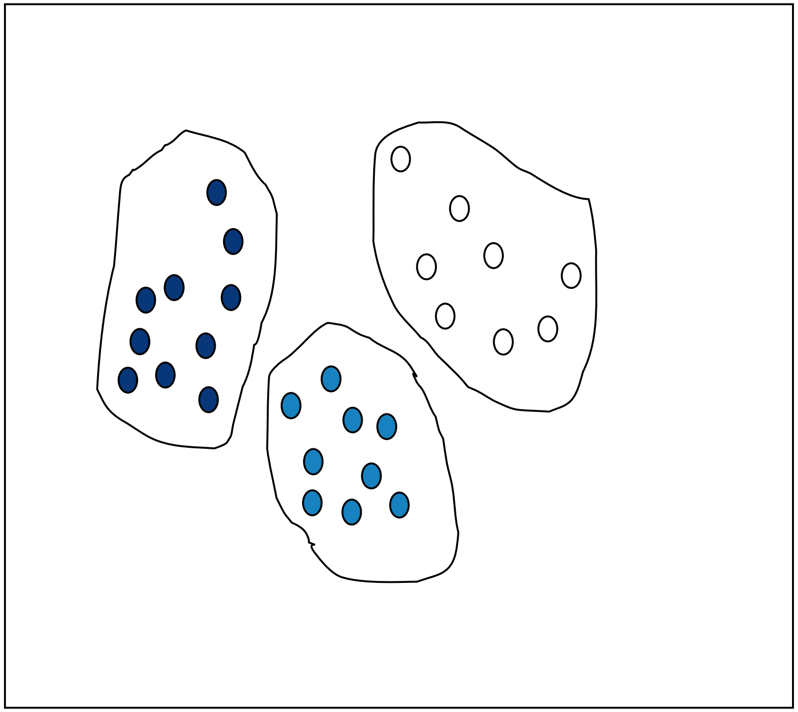
- Simple random sampling
  - There is an equal probability of selecting any particular item
- Sampling without replacement
  - Once an object is selected, it is removed from the population
- Sampling with replacement
  - A selected object is not removed from the population
- Stratified sampling:
  - Partition the data set, and draw samples from each partition (proportionally, i.e., approximately the same percentage of the data)
  - Used in conjunction with skewed data

Sampling: With or without Replacement

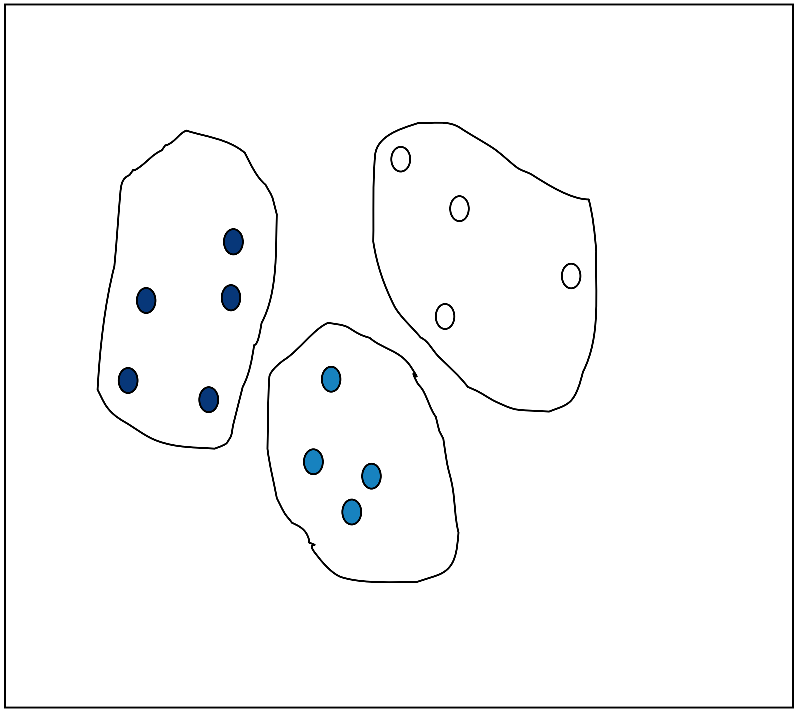


Sampling: Cluster or Stratified Sampling

Raw Data



Cluster/Stratified Sample



Sampling - Examples

80%

	setosa	versicolor	virginica
dataset	50	50	50
random sample	42	41	37
stratified sample	40	40	40

20%

	setosa	versicolor	virginica
random sample	8	11	11
stratified sample	10	10	10

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
34	5.5	4.2	1.4	0.2	setosa
107	4.9	2.5	4.5	1.7	virginica
76	6.6	3.0	4.4	1.4	versicolor
22	5.1	3.7	1.5	0.4	setosa
116	6.4	3.2	5.3	2.3	virginica
113	6.8	3.0	5.5	2.1	virginica

Data Transformation

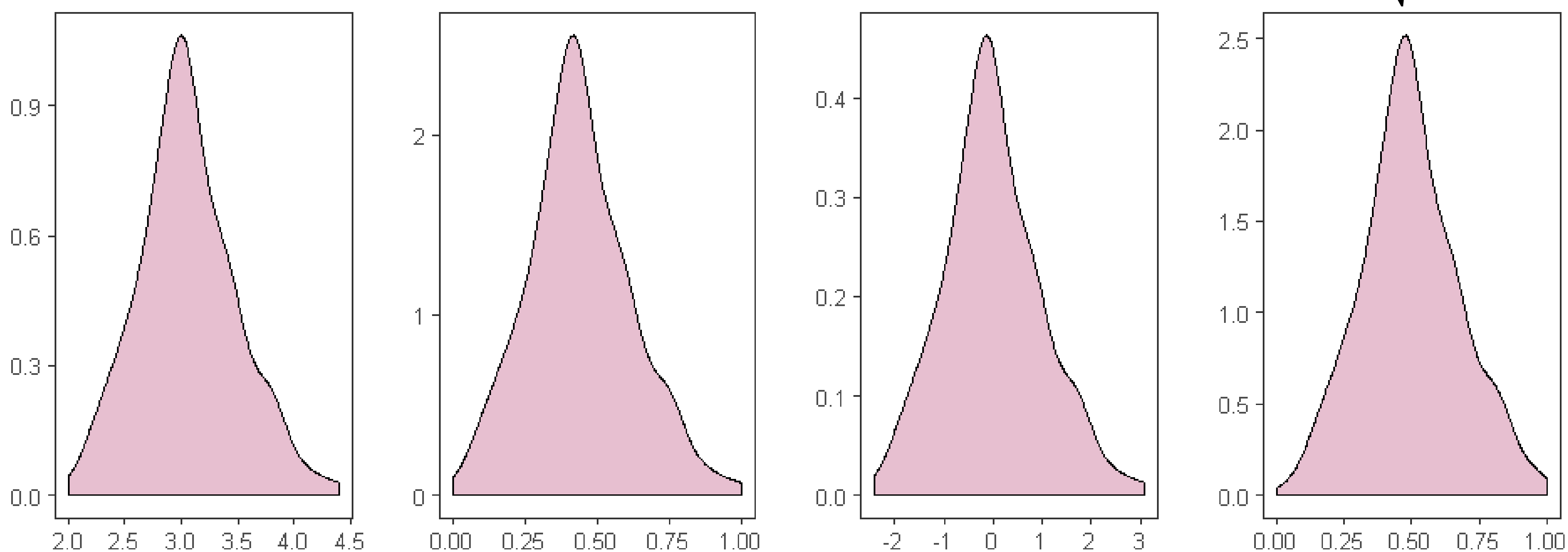
- A function that maps the entire set of values of a given attribute to a new set of replacement values s.t. each old value can be identified with one of the new values
- Methods
  - Attribute/feature construction
    - New attributes constructed from the given ones
    - Complex aggregation
  - Normalization: Scaled to fall within a smaller, specified range
  - Discretization / Smoothing
  - Concept hierarchy climbing
  - Categorical Mapping

Normalization

- **Min-max normalization**: to [$nmin_A$, $nmax_A$]
  - $nv = \frac{v - min_A}{max_A - min_A}(nmax_A - nmin_A) + nmin_A$
- **Z-score normalization** (μ: mean, σ: standard deviation):
  - $nv = \frac{v - \mu_A}{\sigma_A}$
- **Normalization by decimal scaling**
  - $nv = \frac{v}{10^j}$, where j is the smallest integer such that max(|nv|) < 1
- Let income range ($12,000,$98,000) with μ = 54,000, σ = 16,000, then $73,600
  - is mapped to $\frac{73600-12000}{98000-12000}(1-0)+0 = 0.716$ using min-max (0-1)
  - is mapped to $\frac{73600-54000}{16000} = 1.225$ using z-score
  - Is mapped to $\frac{v}{10^6} = 0.736$ using decimal scaling

Normalization

Data | Min-max [0-1] | Z-score/N(0,1) | N(0.5, $\sqrt{\frac{0.5}{2.698}}$)



Discretization & Smoothing

- Discretization is the process of transferring continuous functions, models, variables, and equations into discrete counterparts
- Smoothing is a technique that creates an approximating function that attempts to capture important patterns in the data while leaving out noise or other fine-scale structures/rapid phenomena
- A important part of the discretization/smoothing is to set up bins for proceeding the approximation

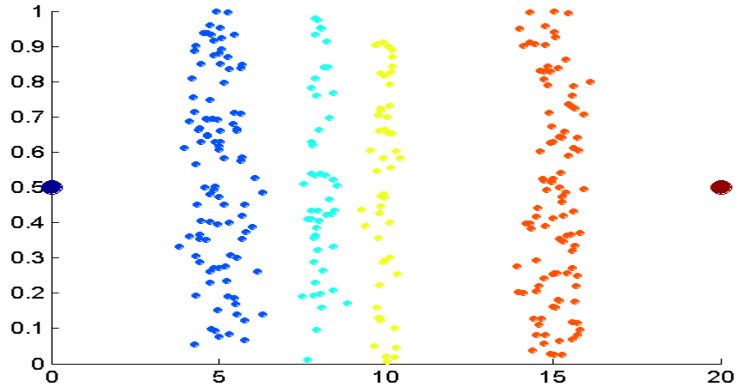
Binning methods for data smoothing

- Equal-width (distance) partitioning
  - Divides the range into N intervals of equal size: uniform grid
  - if A and B are the lowest and highest values of the attribute, the width of intervals will be: W = (B –A)/N
  - The most straightforward, but outliers may dominate presentation
  - Skewed data is not handled well
- Equal-depth (frequency) partitioning
  - Divides the range into N intervals, each containing approximately same number of samples
  - Good data scaling
  - Managing categorical attributes can be tricky

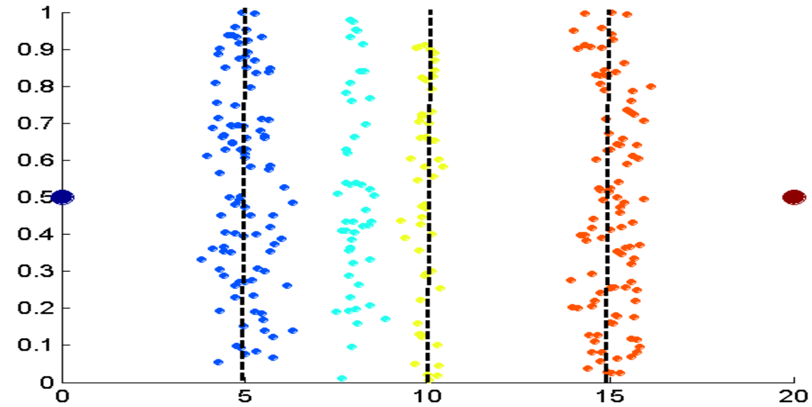
Binning methods for data smoothing

- Sorted data for price (in dollars):
  - 4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34
- Binning of size 3
  - Partition of equal-length: (34-4)/3
    - Bin 1 [4-13[: 4, 8, 9
    - Bin 2 [14-23[: 15, 21, 21
    - Bin 3 [23-34]: 24, 25, 26, 28, 29, 34
  - Partition into equal-frequency (equi-depth) bins:
    - Bin 1: 4, 8, 9, 15
    - Bin 2: 21, 21, 24, 25
    - Bin 3: 26, 28, 29, 34
    - Smoothing by bin means:
      - Bin 1: 9, 9, 9, 9
      - Bin 2: 23, 23, 23, 23
      - Bin 3: 29, 29, 29, 29
    - Smoothing by bin boundaries:
      - Bin 1: 4, 4, 4, 15
      - Bin 2: 21, 21, 25, 25
      - Bin 3: 26, 26, 26, 34

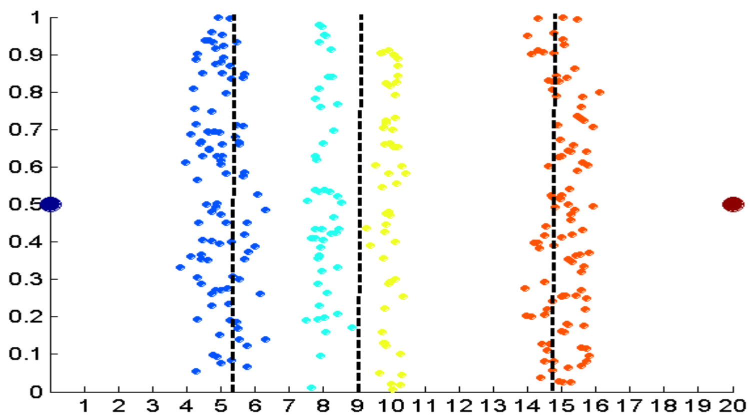
Influence on binning during data smoothing techniques



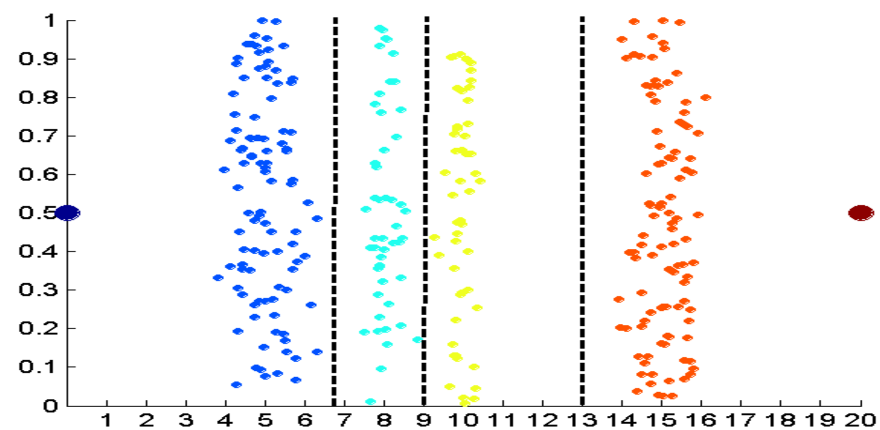
data



Equal interval width (binning)



Equal frequency (binning)



K-means clustering

Categorical Mapping

- n binary derived inputs: one for each value of the original attribute
  - This 1-to-N mapping is commonly applied when N is relatively small
- As N grows, the number of inputs to the model increases and consequently the number of parameters to be estimated increases
  - Thus, this method is not applicable to high-cardinality attributes with hundreds or thousands of distinct values

Example Categorical Mapping

iris

Filter

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
1	5.1	3.5	1.4	0.2	setosa
2	4.9	3.0	1.4	0.2	setosa
3	4.7	3.2	1.3	0.2	setosa
4	4.6	3.1	1.5	0.2	setosa
5	5.0	3.6	1.4	0.2	setosa

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Speciessetosa	Speciesversicolor	Speciesvirginica
1	5.1	3.5	1.4	0.2	1	0	0
2	4.9	3.0	1.4	0.2	1	0	0
3	4.7	3.2	1.3	0.2	1	0	0
4	4.6	3.1	1.5	0.2	1	0	0
5	5.0	3.6	1.4	0.2	1	0	0

Practicing

- Take some time to practice the examples
  - https://nbviewer.jupyter.org/github/eogasawara/mylibrary/blob/master/myPreprocessing.ipynb

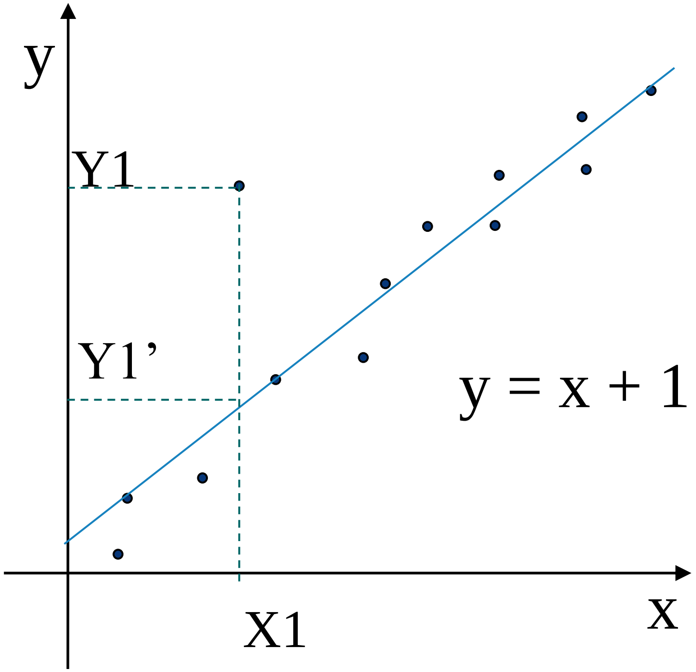
Regression

Regression Models

- Linear regression
  - Data modeled to fit a straight line
  - Often uses the least-square method to fit the line
- Multiple regression
  - Allows a response variable Y to be modeled as a linear function of the multidimensional feature vector

Regression Analysis

- A collective name for techniques for the modeling and analysis of numerical data consisting
  - values of a dependent variable (also called response variable or measurement)
  - one or more independent variables
- The parameters are estimated to give a "best fit" of the data
- Most commonly the best fit is evaluated by using the least squares method, but other criteria have also been used
- Used for prediction (including forecasting of time-series data), inference, hypothesis testing, and modeling of causal relationships



Types of regression models

- Linear regression: $Y = w X + b$
  - Two regression coefficients, w and b, specify the line and are to be estimated by using the data at hand
  - Using the least squares criterion to the known values of Y_1, Y_2, …, X_1, X_2, ….
- Multiple regression: $Y = b_0 + b_1 X_1 + b_2 X_2$
  - Many nonlinear functions can be approximated by the above
- Polynomial regression: $Y = b_0 + b_1 X_1 + b_2 X_1^2$

Boston dataset

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
numeric	numeric	numeric	integer	numeric	numeric	numeric	numeric	integer	numeric	numeric	numeric	numeric	numeric

crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	black	lstat	medv
0.00632	18	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4.98	24.0
0.02731	0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9.14	21.6
0.02729	0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4.03	34.7
0.03237	0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2.94	33.4
0.06905	0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5.33	36.2
0.02985	0	2.18	0	0.458	6.430	58.7	6.0622	3	222	18.7	394.12	5.21	28.7

Fitting a first model

- Explaining house price using lower status population variable
- *lm* builds the model
- *summary* describes the significance of the built model

```
lm.fit = lm(medv ~ lstat, data = Boston)

summary(lm.fit)
```

```
Call:
lm(formula = medv ~ lstat, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-15.168  -3.990  -1.318   2.034  24.500 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 34.55384    0.56263   61.41   <2e-16 ***
lstat       -0.95005    0.03873  -24.53   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.216 on 504 degrees of freedom
Multiple R-squared:  0.5441,	Adjusted R-squared:  0.5432 
F-statistic: 601.6 on 1 and 504 DF,  p-value: < 2.2e-16
```

Prediction

- The *predict* function makes predictions from the adjusted model
- The predictions can be presented with either confidence and prediction intervals
  - These intervals can be analyzed at https://statisticsbyjim.com/hypothesis-testing/confidence-prediction-tolerance-intervals/

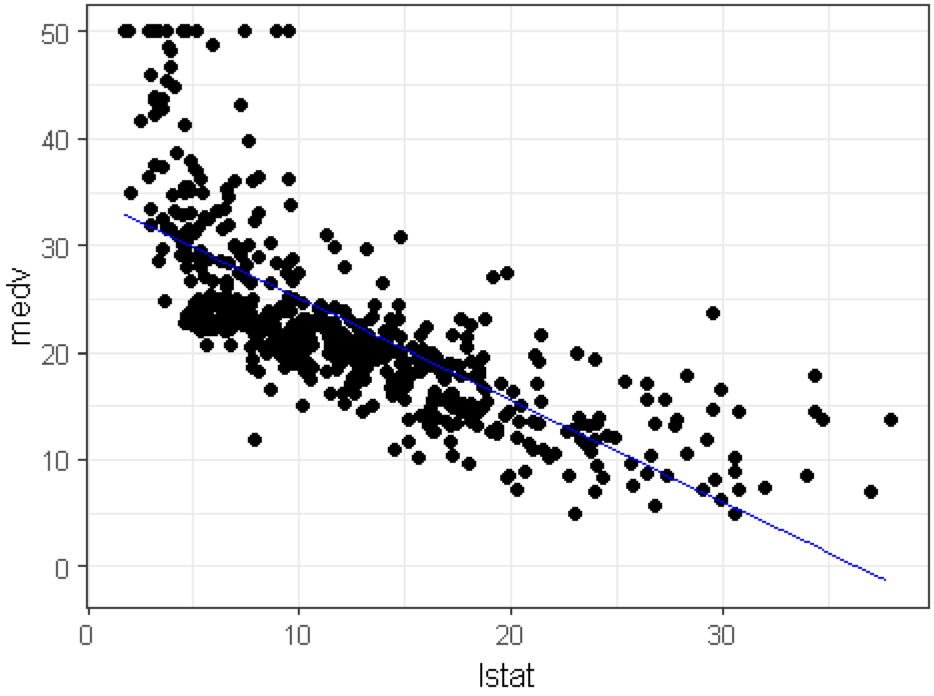
```
predict(lm.fit, data.frame(lstat =(c(5, 10, 15))), interval = "confidence")
predict(lm.fit, data.frame(lstat =(c(5, 10, 15))), interval = "prediction")
```

fit	lwr	upr
29.80359	29.00741	30.59978
25.05335	24.47413	25.63256
20.30310	19.73159	20.87461

fit	lwr	upr
29.80359	17.565675	42.04151
25.05335	12.827626	37.27907
20.30310	8.077742	32.52846

Plotting the regression model

- Good practice to plot the regression model
- Enables us to have a feeling of its quality



Polynomial regression

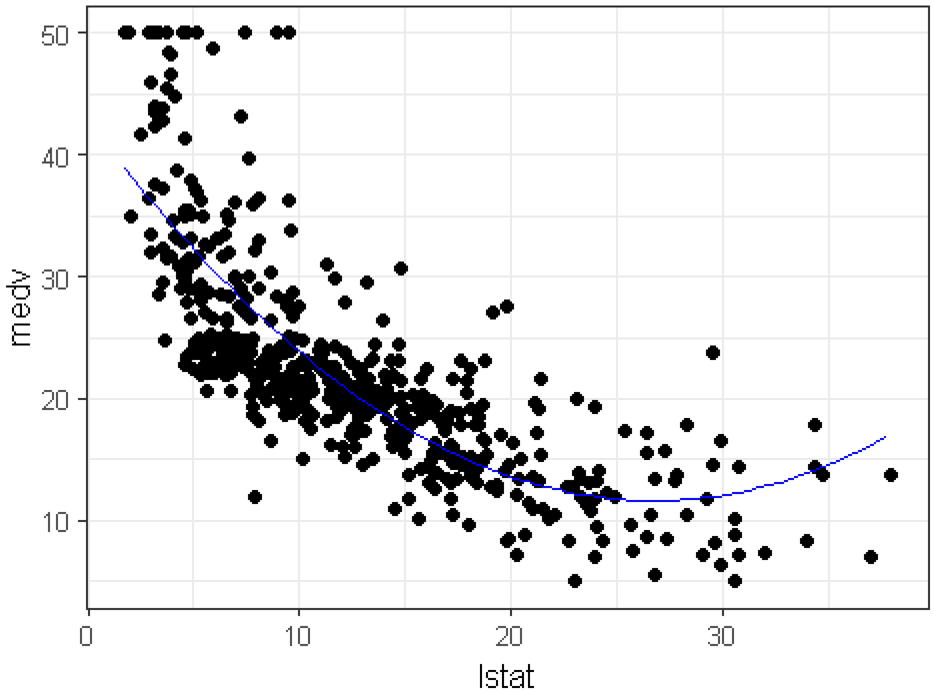
- It is possible to introduce polynomial dimensions of independent data
  - It is important to notice that it is still a linear model

```
lm.fit_p =lm(medv~lstat+I(lstat^2), data=Boston)
summary (lm.fit_p)
```

```
Call:
lm(formula = medv ~ lstat + I(lstat^2), data = Boston)

Residuals:
     Min       1Q   Median       3Q      Max 
-15.2834  -3.8313  -0.5295   2.3095  25.4148 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 42.862007   0.872084   49.15   <2e-16 ***
lstat       -2.332821   0.123803  -18.84   <2e-16 ***
I(lstat^2)   0.043547   0.003745   11.63   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 5.524 on 503 degrees of freedom
Multiple R-squared:  0.6407,	Adjusted R-squared:  0.6393 
F-statistic: 448.5 on 2 and 503 DF,  p-value: < 2.2e-16
```

Plotting the polynomial regression

- It is only necessary to present the basic dimension



Assessing the polynomial regression

- Using ANOVA
  - Null hypothesis: Both models are not different
    - p-value > 5%
  - Alternative hypothesis: They are different
    - p-value < 5%

```
anova(lm.fit, lm.fit_p)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
504	19472.38	NA	NA	NA	NA
503	15347.24	1	4125.138	135.1998	7.630116e-28

Multiple regression

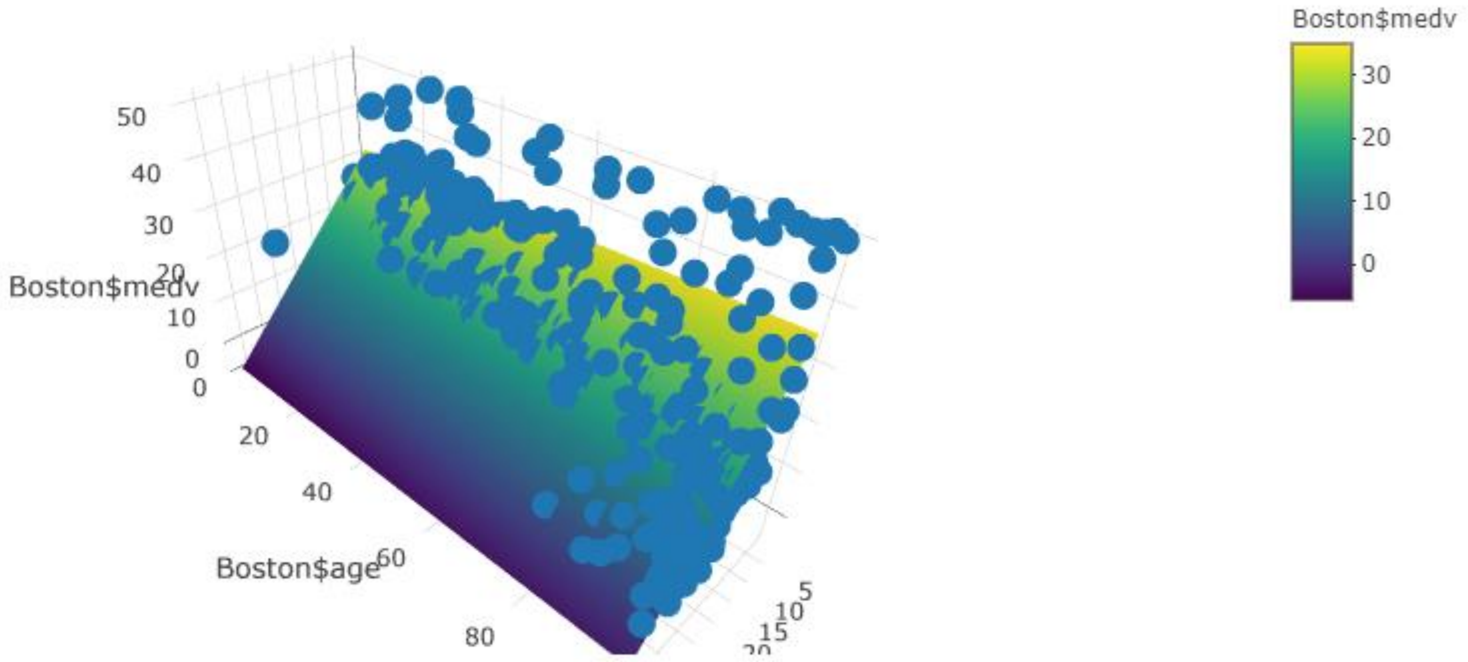
- It is possible to use more than one dimension for independent data

```
lm.fit2 =lm(medv~lstat+age, data=Boston)
summary (lm.fit2)
```

```
Call:
lm(formula = medv ~ lstat + age, data = Boston)

Residuals:
    Min      1Q  Median      3Q     Max 
-15.981  -3.978  -1.283   1.968  23.158 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 33.22276    0.73085  45.458  < 2e-16 ***
lstat       -1.03207    0.04819 -21.416  < 2e-16 ***
age          0.03454    0.01223   2.826  0.00491 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 6.173 on 503 degrees of freedom
Multiple R-squared:  0.5513,    Adjusted R-squared:  0.5495 
F-statistic:   309 on 2 and 503 DF,  p-value: < 2.2e-16
```

Plotting the surface of regression

- Explore from different angles …



Checking the significance of the model

- Using ANOVA

```
anova(lm.fit ,lm.fit2)
```

Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
504	19472.38	NA	NA	NA	NA
503	19168.13	1	304.2528	7.984043	0.004906776

Logistic Regression

- Classification

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
numeric	numeric	numeric	numeric	factor

Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa

Simplifying the problem

- Focus in one class prediction
  - Ex.: versicolor versus non-versicolor
    - 33% versus 67%

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species	versicolor
31	4.8	3.1	1.6	0.2	other	0
27	5.0	3.4	1.6	0.4	other	0
102	5.8	2.7	5.1	1.9	other	0
57	6.3	3.3	4.7	1.6	versicolor	1
113	6.8	3.0	5.5	2.1	other	0
73	6.3	2.5	4.9	1.5	versicolor	1

Building the model

- Uses logistic regression
  - Using all variables except Species (class label!)
- Measuring the adjustment of the model

Creation of logistic regression model using all independent variables.

```
pred <- glm(versicolor ~ .-Species, data=train, family = binomial)
```

Measuring the level of ajustment using training data.

```
res <- predict(pred, train, type="response")
res <- as.integer(res >= t)
table(res, train$versicolor)
```

```
res  0  1
  0 60  9
  1 20 31
```

Measuring the performance of the model

- Prediction

```
res <- predict(pred, test, type="response")
res <- res >= t
table(res, test$versicolor)
```

```
res      0  1
  FALSE 15  6
  TRUE   5  4
```

Building a simpler model

- Petal.Length and Petal.Width were more significant in the exploratory analysis
- During preprocessing, they also lead to lower entropy during discretization

Creation of logistic regression model using the independent variables with lower entropy during binning transformation.

```
pred <- glm(versicolor ~ Petal.Length + Petal.Width, data=train, family = binomial)
```

Measuring the level of ajustment using training data.

```
res <- predict(pred, train, type="response")
res <- as.integer(res >= t)
table(res, train$versicolor)
```

```
res  0  1
  0 62  9
  1 18 31
```

Measuring the performance of the model

- Prediction

```
res <- predict(pred, test, type="response")
res <- as.integer(res >= t)
table(res, test$versicolor)
```

```
res  0  1
  0 16  2
  1  4  8
```

Practicing

- Take some time to practice the examples
  - https://nbviewer.jupyter.org/github/eogasawara/mylibrary/blob/master/myRegression.ipynb

CEFET/RJ

Ph.D. and Master of Science

- Mechanical Engineering and Materials Technology
- Instrumentation and Applied Optics
- Production Engineering and Systems
- Science, Technology and Education

Master of Science

- Computer Science
- Electrical Engineering
- Ethnic and Racial Relations
- Philosophy and Teaching

PPCIC - Computer Science

Algorithms, Optimization, and Computational Modeling

- Algorithms, Combinatory, and Optimization
- Computational Modeling Applied to Science and Engineering
- Adaptive Networks
- Graph Theory and Applications

Data Management and Applications

- Data Integration, Management, and Workflows for Big Data
- Data Mining and Machine Learning
- Text Mining, Affective Computing and Behavior Analysis
- Systems and Applications

PPCIC - Computer Science

Course	Core	Credits
Computational Linear Algebra	Specific	3
Linear Algebra and Graphs	Specific	3
Graph Algorithms	Specific	3
Analysis and Design of Algorithms	Basic	3
Robotics Applications	Specific	3
Machine Learning	Specific	3
Computer Architecture	Basic	3
Database	Basic	3
Parallel and Distributed Computing	Basic	3
Fundamentals of Multimedia Systems	Specific	3
Large-scale Data Management	Specific	3
Scientific Methodology in computing	Basic	3
Statistical Methods	Basic	3
Data Mining	Specific	3
Process Mining	Specific	3
Text Mining	Specific	3
Optimization by Metaheurísticas	Specific	3
Operational Research	Specific	3

Main References