




Business Intelligence

06 Data Understanding & Data Visualization

Prof. Dr. Bastian Amberg
(summer term 2024)

22.5.2024

Schedule

	Wed., 10:00-12:00			Fr., 14:00-16:00 (Start at 14:30)		Self-study	
Basics	W1	17.4.	(Meta-)Introduction	19.4.		Python-Basics	Chap. 1
	W2	24.4.	Data Warehouse – Overview & OLAP	26.4.	[Blockveranstaltung SE Prof. Gersch]		Chap. 2
	W3	1.5.		3.5.	Data Warehouse Modeling I 		Chap. 3
	W4	8.5.	Data Warehouse Modeling I & II	10.5.	Data Mining Introduction		
Main Part	W5	15.5.	CRISP-DM, Project understanding	17.5.	Python-Basics-Online Exercise	Python-Analytics	Chap. 1
	W6	22.5.	Data Understanding, Data Visualization	24.5.	No lectures, but bonus tasks 1.) Co-Create your exam 2.) Earn bonus points for the exam		Chap. 2
	W7	29.5.	Data Preparation	31.5.			
	W8	5.6.	Predictive Modeling I	7.6.	Predictive Modeling II (10:00 -12:00)	BI-Project	Start
	W9	12.6.	Fitting a Model I	14.6.	Python-Analytics-Online Exercise		
	W10	19.6.	Guest Lecture	21.6.	Fitting a Model II		
	W11	26.6.	How to avoid overfitting	28.6.	What is a good Model?		
Deepening	W12	3.7.	Project status update Evidence and Probabilities	5.7.	Similarity (and Clusters) From Machine to Deep Learning I		
	W13	10.7.		12.7.	From Machine to Deep Learning II		
	W14	17.7.	Project presentation	19.7.	Project presentation		End
Ref.					Klausur 1. Termin, 31.7.'24 Klausur 2. Termin, 2.10.'24	Projektbericht	

Note on Bonus tasks

Vom 24.5.'24 bis spätestens 7.6.'24

Bitte jeweils die genaue Aufgabenstellung inklusive Abgabeformalitäten beachten!

Diese ist ab Freitag, 24.5., 10 Uhr in Blackboard verfügbar.

1/6

1. Co-Create your exam

Einzelleistung (bzw. Leistung des gesamten Kurses)

- Zwei Aussagen im Kontext der Veranstaltungen 01 bis 04 formulieren (eine wahr, eine falsch)
- Freitextaufgabe zu Veranstaltungen 01 bis 04 formulieren, die mit einem Wort (keine Aussage über wahr oder falsch!) beantwortet werden kann
- Falls bis zum 7.6.'24 ≥ 84 (= 3 Fragen x 28 gemeldete Teilnehmer:innen) unterschiedliche, sinnvolle Fragen/Aussagen zusammen kommen, ist eine Teilmenge davon Bestandteil der Klausur

2. Earn bonus points for the exam

Gruppenarbeit (min. 2 bis max. 3 Personen pro Gruppe)

- Rechercheaufgabe zu Supervised, Unsupervised und Reinforcement Learning
- Grafische Darstellung auf einer Folie inklusive Beschreibung und kritischer Würdigung
- Dabei Einsatz von generativer KI möglich (z.B. ChatGPT, Bing AI),
sofern die verwendeten Anfragen/Befehle nachvollziehbar dokumentiert werden
- Maximal drei Bonuspunkte, die auf die in der Klausur erreichte Punktzahl addiert werden
(sofern die Klausur bestanden wurde)

- ✓ From business problems to data mining tasks
- ✓ Supervised vs. unsupervised methods vs. reinforcement learning

- [Demo Reinforcement Learning](#) (from the last lesson)
- [A.I. Learns to Drive From Scratch in Trackmania](#) (very well explained youtube video)
- Optimization tasks in stochastic, dynamic environments (project example)



Learning
Approach

Learn from....
(input? - output?)

Learn for....
(target?)

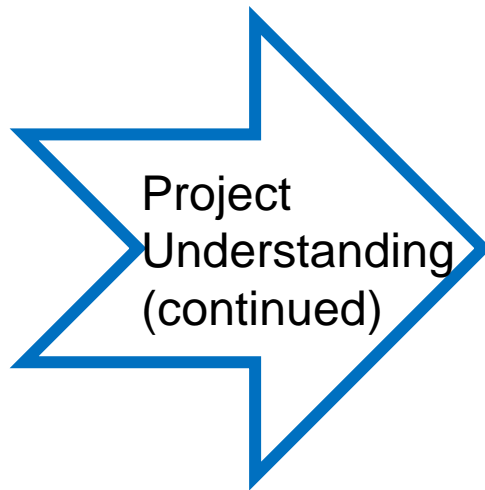
Example?

Kahoot-Fragen zu den Inhalten
www.kahoot.it
(über Smartphone oder Laptop)
PIN folgt

- ✓ The data mining process – CRISP-DM
 - Business / Project understanding

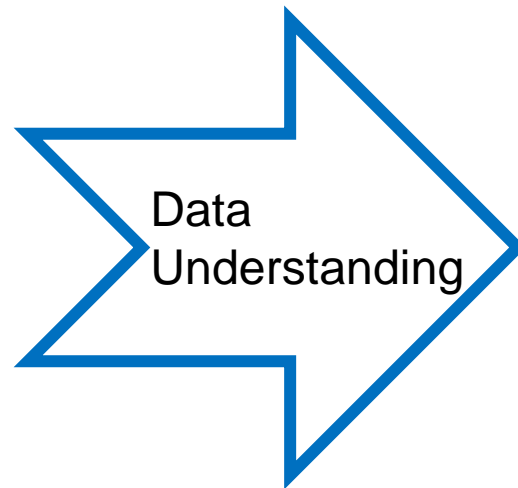
Today's Agenda

First Part

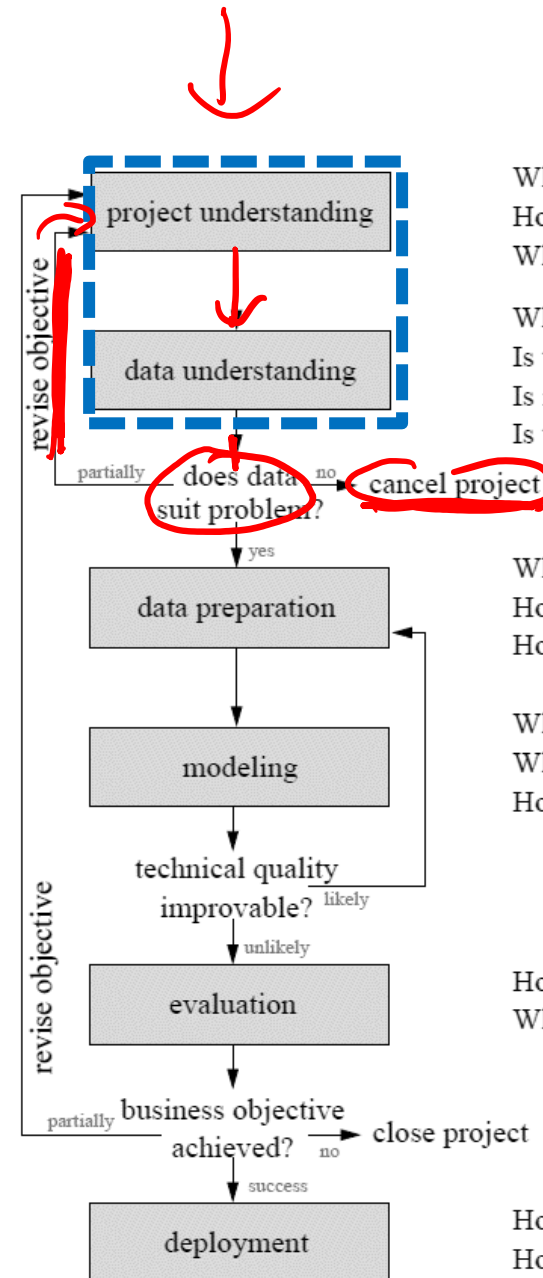


*See slides 5b
from last week*

Second Part



This set of slides



What exactly is the problem, the expected benefit?
How would a solution look like?
What is known about the domain?

What data do we have available?
Is the data relevant to the problem?
Is it valid? Does it reflect our expectations?
Is the data quality, quantity, recency sufficient?

Which data should we concentrate on?
How is the data best transformed for modeling?
How may we increase the data quality?

What kind of model architecture suits the problem best?
What is the best technique/method to get the model?
How good does the model perform technically?

How good is the model in terms of project requirements?
What have we learned from the project?

How is the model best deployed?
How do we know that the model is still valid?

First Part

(see slides 10 to 18 from slide set 5b)

✓ The data mining process – CRISP-DM

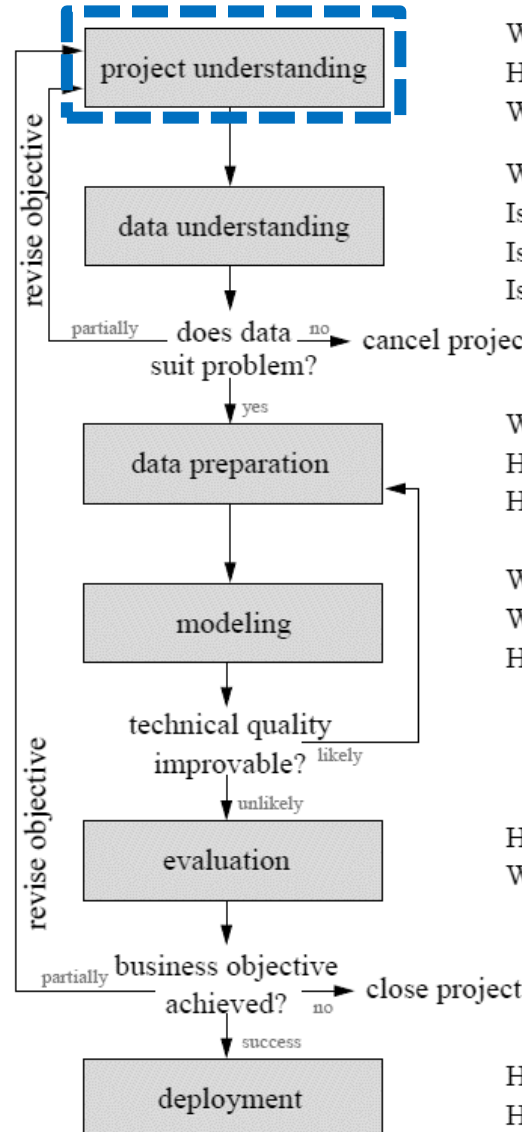
✓ Business / Project understanding



Image: "Moneyball"/Columbia Pictures, Video

- Assess the situation
- Determine analysis goals

Ref.



What exactly is the problem, the expected benefit?
How would a solution look like?
What is known about the domain?

What data do we have available?
Is the data relevant to the problem?
Is it valid? Does it reflect our expectations?
Is the data quality, quantity, recency sufficient?

Which data should we concentrate on?
How is the data best transformed for modeling?
How may we increase the data quality?

What kind of model architecture suits the problem best?
What is the best technique/method to get the model?
How good does the model perform technically?

How good is the model in terms of project requirements?
What have we learned from the project?

How is the model best deployed?
How do we know that the model is still valid?

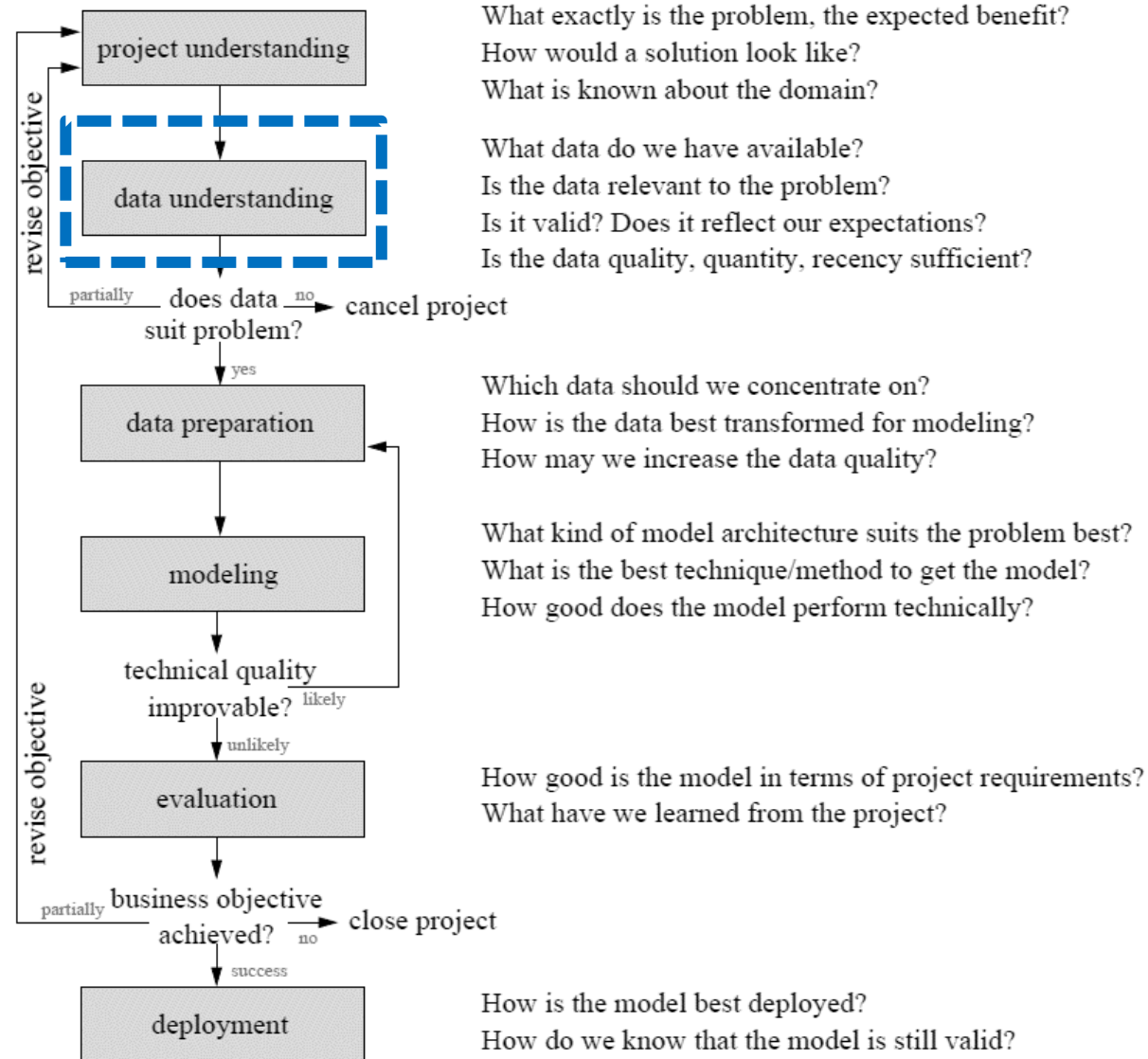
Second Part

Cross Industry Standard Process for Data Mining

Iteration as a rule

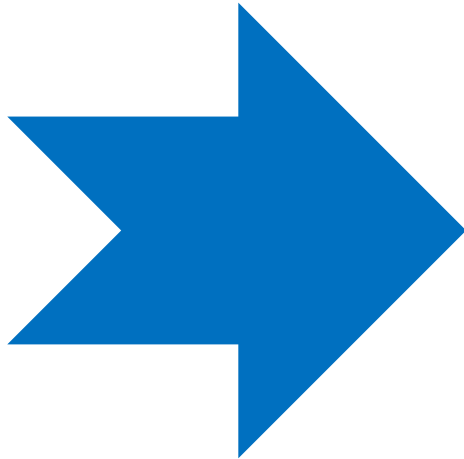
Process of data exploration

Implementation of the KDD Process



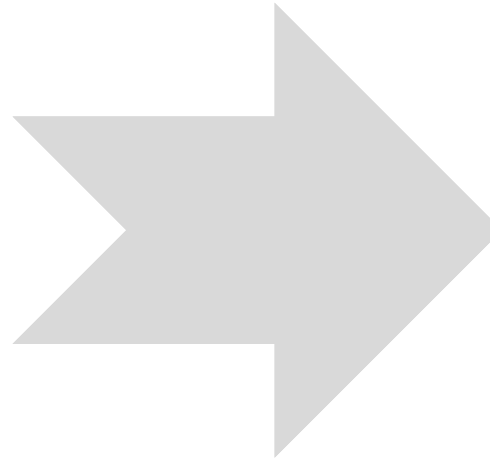
Agenda for Data Understanding

(1) Data Understanding



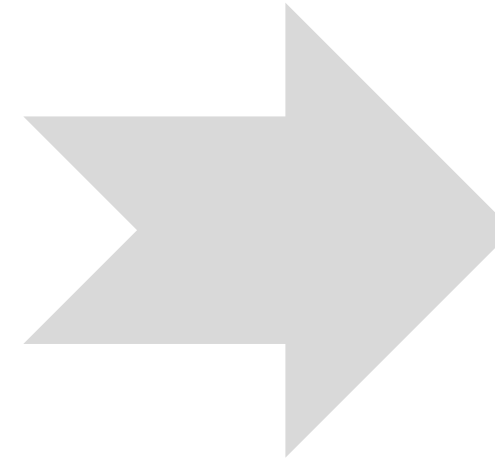
Attribute Understanding
Data Quality

(2) Data Visualization (= Data Understanding – Part 2)



Low-dimensional relationships

Univariate Analysis
Bivariate Analysis



Higher-dimensional relationships

Principal Component Analysis
Parallel Coordinates

Goals of data understanding

Gain **general insights** about the data (independent of the project goal)

Check the assumptions made during the project understanding phase (representativeness, informativeness, data quality, presence/absence of external factors, dependencies, ...)

Check the specified domain knowledge

Check suitability of the data for the project goals

Wertebereiche

↑ **Never trust any data** as long as you have not carried out some simple plausibility checks! ↓



Attribute understanding

And types of attributes

We often assume that the data set is provided in form of a simple table

The rows of the table are called instances, records or data objects

The columns of the table are called attributes, features or variables

Age ↓ Income ↓

	Attribute 1	...	Attribute M
→ Record 1			
...			
Record n			

Freie Universität Berlin

= ≠

Categorical (nominal): finite domain. The values of a categorical attribute are often called classes or categories

Examples: [female, male, diverse], [ordered, received]

0 → 1 → 2
female | male | diverse
011 | 011 | 011

≤, ≥

Ordinal: finite domain with a linear ordering on the domain.

Example: [B.Sc., M.Sc., Ph.D.], [Dawn, Noon, Afternoon, Evening, Night]

Numerical: values are numbers

Discrete: categorical attribute or numerical attribute whose domain is a subset of an integer number

Continuous: numerical attribute with values in the real numbers or in an interval (float)

Scales for numerical attributes

+ / -

Interval scale: the definition of the value 0 is arbitrary. Ratios are meaningless.

Examples: date (Unix standard time: time point zero is in the year 1970), temperature (°C or °F)



Ratio scale: 0 has a canonical meaning

(none of the measured quality exists)

Ratios make sense.

Examples: distance, duration

:/ *

Absolute scale: domain with a unique measurement unit.

Examples: any kind of counting process
(number of children, number of visits to the doctor)

Specific problems of categorical attributes

Levels of granularity; Dynamic domains

Different **levels of granularity** might be definable.

Examples:

product categories/types:

- └ General category: drinks, food, clothes, ...
- └ More refined categories for drinks: water, beer, wine, ...
- └ Further refinement for water based on the producer.
- └ Further refinement of the of each producer based on the bottle size (0.33 l, 0.5 l, 1 l, 1.5 l)

Dynamic domains: The possible values of the domain might change over time.

Example: certain product categories or products might not be sold anymore. New product categories or products are introduced.

⇒ The most refined level provides the **most detailed information**, but will not help to discover general associations like “Wine and cheese are often bought together”



⇒ The analysis of such data will be **biased** to values (example: products) that have been in the domain for a long time.



Data quality

Syntactic accuracy vs. Semantic accuracy



Syntactic accuracy is violated if an entry does not belong to the domain of the attribute

The entry female for the categorical attribute *gender*
Text entries for numerical attributes
Values out of range for numerical attributes (negative numbers for weight, distance, counting process, ...)

Syntactic accuracy can be checked quite easily

Semantic accuracy is violated if an entry is not correct although it belongs to the domain of the attribute

Name	Gender	
John Smith	Female	??
Lisa McIntosh	Female	
Rick Rickerton	Male	
Jane Smith	Male	??
John Doe	Male	

Semantic accuracy is more difficult to check than syntactic accuracy.

Can only be investigated based on “business rules” and plausibility checks.

Data quality

Completeness, Unbalanced data and timeliness

complete **attribute values**:

fraction of "null" entries for an attribute.

Note that missing values are not always marked explicitly as missing, for instance in the case of default entries!

e.g., Time: 12 o'clock
Birthday: 01.01.xxxx

complete **records**: complete records might be missing.

Example 1: Three years ago, a new system was introduced and not all customer data were transferred to the new system.

Example 2: The data set is biased, e.g., a bank might have rejected customers with no income, but they did not protocol it.

Unbalanced data:

the data set might be biased extremely to one type of records

Production line for goods including quality control → defective goods will be a very small fraction of all records!

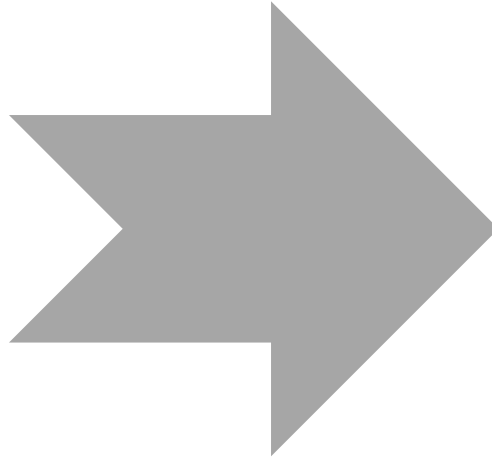
e.g., 99.9% (+)
0.01% (-)

Timeliness:

is the available data up to date to be considered to be representative?

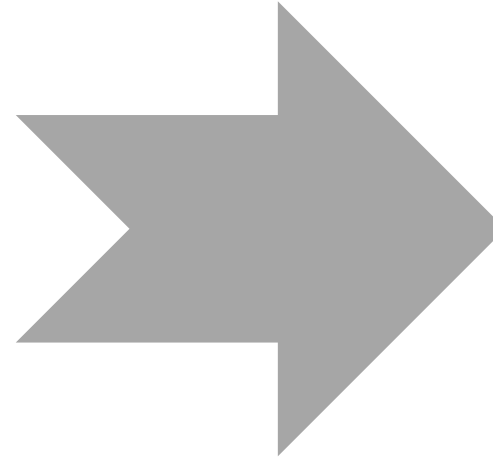


Data Visualization (= Data Understanding – Part 2)



Low-dimensional relationships

Univariate Analysis
Bivariate Analysis



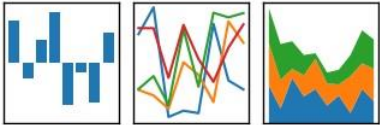
Higher-dimensional relationships

Principal Component Analysis
Parallel Coordinates

First: Python libraries for data visualization

used libraries...
(among others)

pandas
 $y_{it} = \beta' x_{it} + \mu_i + \epsilon_{it}$



home // about // **get pandas** // documentation // community // talks // donate

Python Data Analysis Library

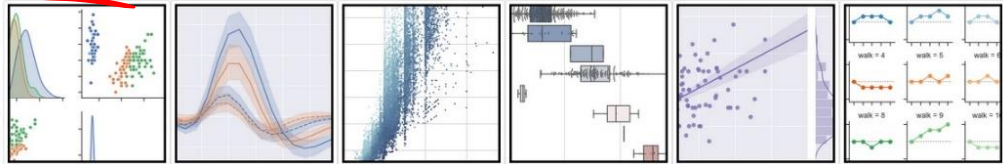
pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the [Python](#) programming language.

pandas is a [NumFOCUS](#) sponsored project. This will help ensure the success of development of *pandas* as a world-class open-source to [donate](#) to the project.

<https://pandas.pydata.org/>

VERSIONS
Version 1.4.2,
Release date Apr 02, 2022
[download](#) // [docs](#) // [pdf](#)

seaborn: statistical data visualization



Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

For a brief introduction to the ideas behind the library, you can read the [introductory notes](#). Visit the installation page to see how you can download the package. You can browse the [example gallery](#) to see what you can do with seaborn, and then check out the tutorial and API reference to find out how.

To see the code or report a bug, please visit the [github repository](#). General are most at home on [stackoverflow](#), where there is a seaborn tag.

<https://seaborn.pydata.org/>

Contents	Features
<ul style="list-style-type: none">• Introduction• Release notes• Installing• Example gallery	<ul style="list-style-type: none">• Relational: API Tutorial• Categorical: API Tutorial• Distributions: API Tutorial• Regressions: API Tutorial

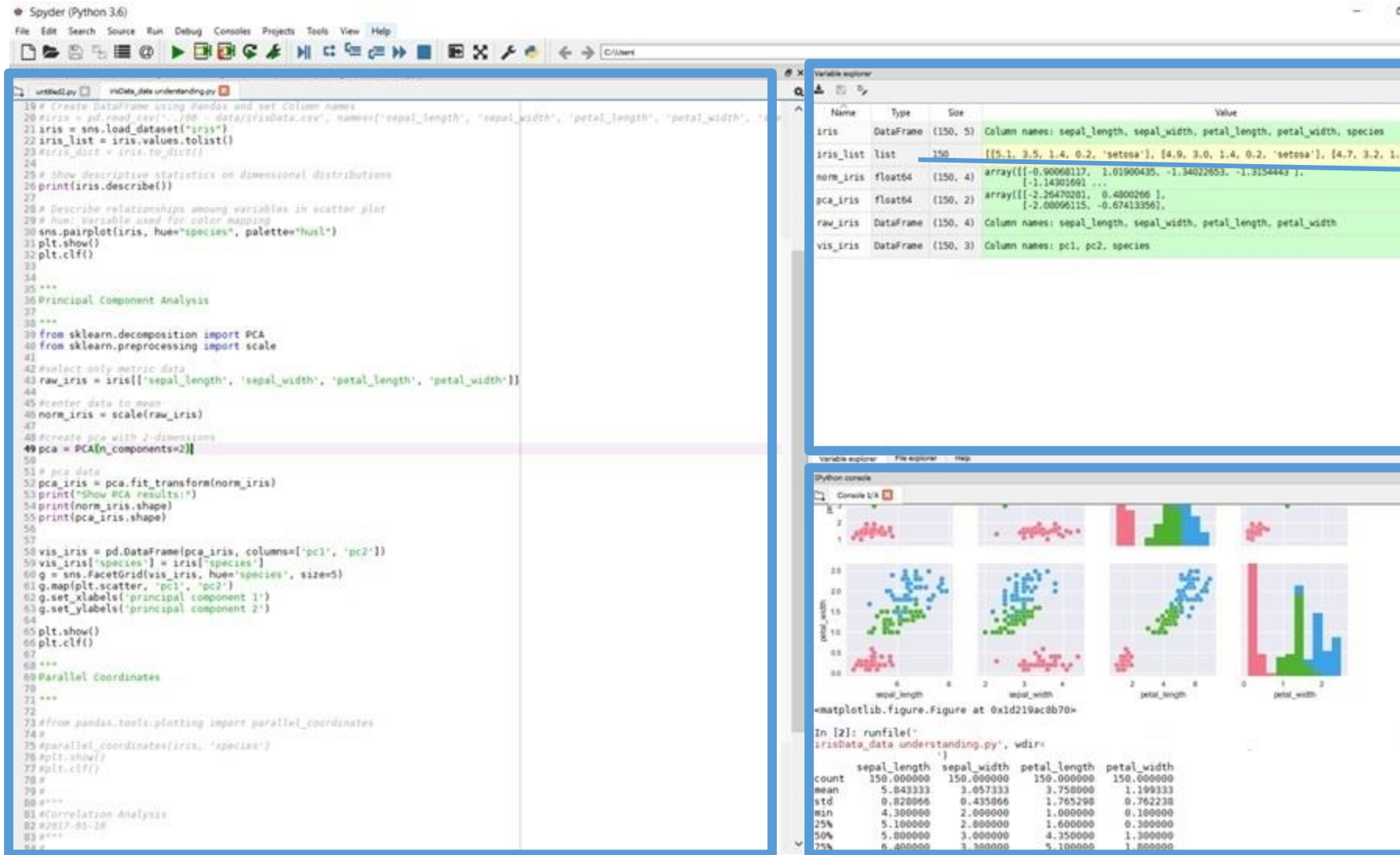
How to use?

>> See sample code on the slides.

First: Getting started with a Python-IDE

Anaconda Spyder

Code



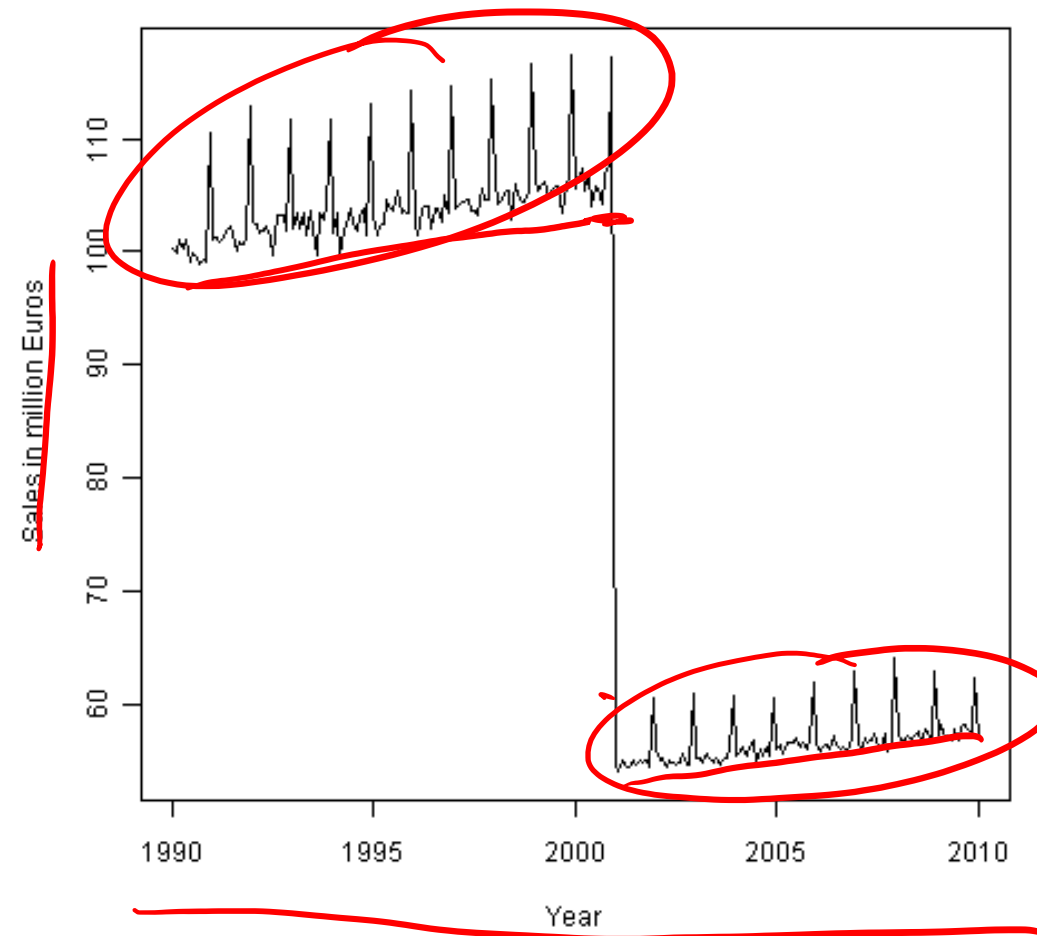
Variable Explorer

iPython Console
(input/output)

Ref.

“There is no excuse for failing to plot and look”.

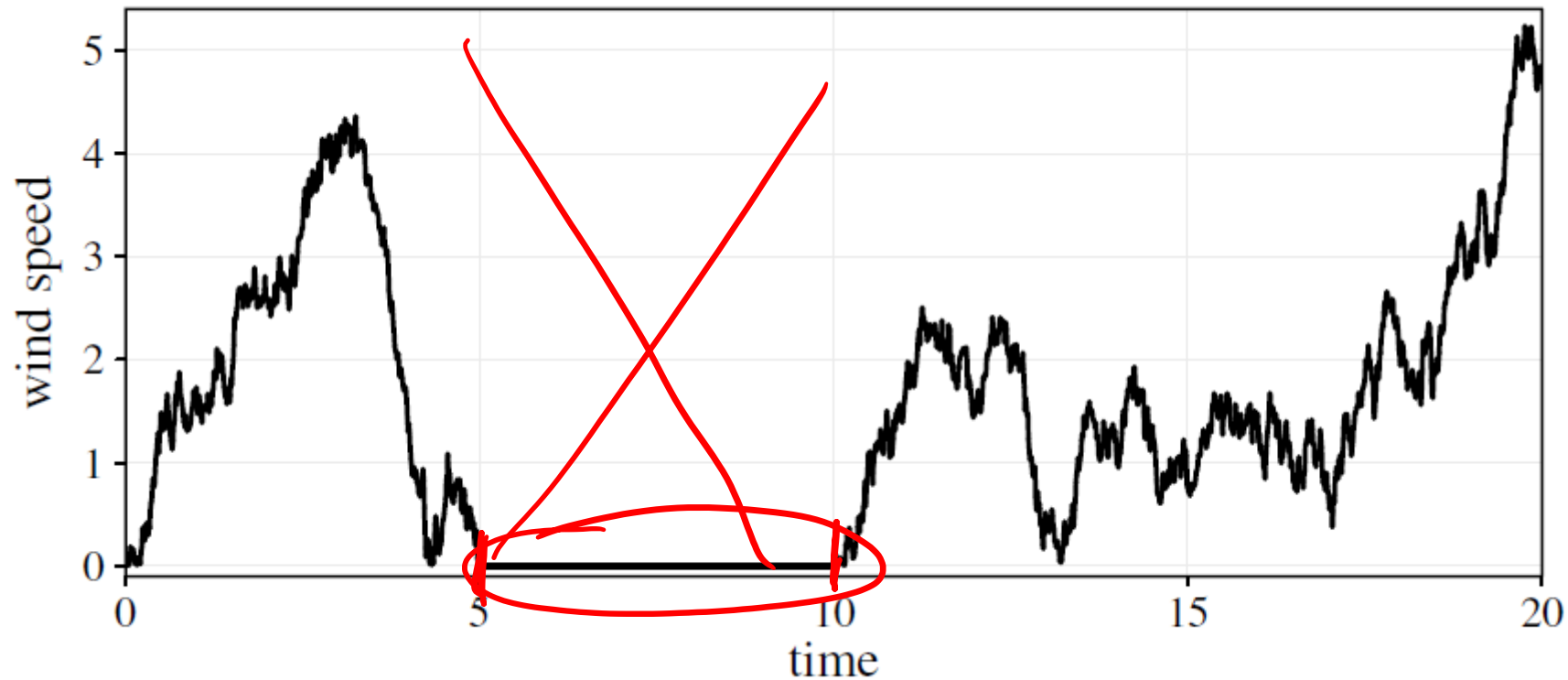
Tukey (1977)



1 €
= 1,95583 DM

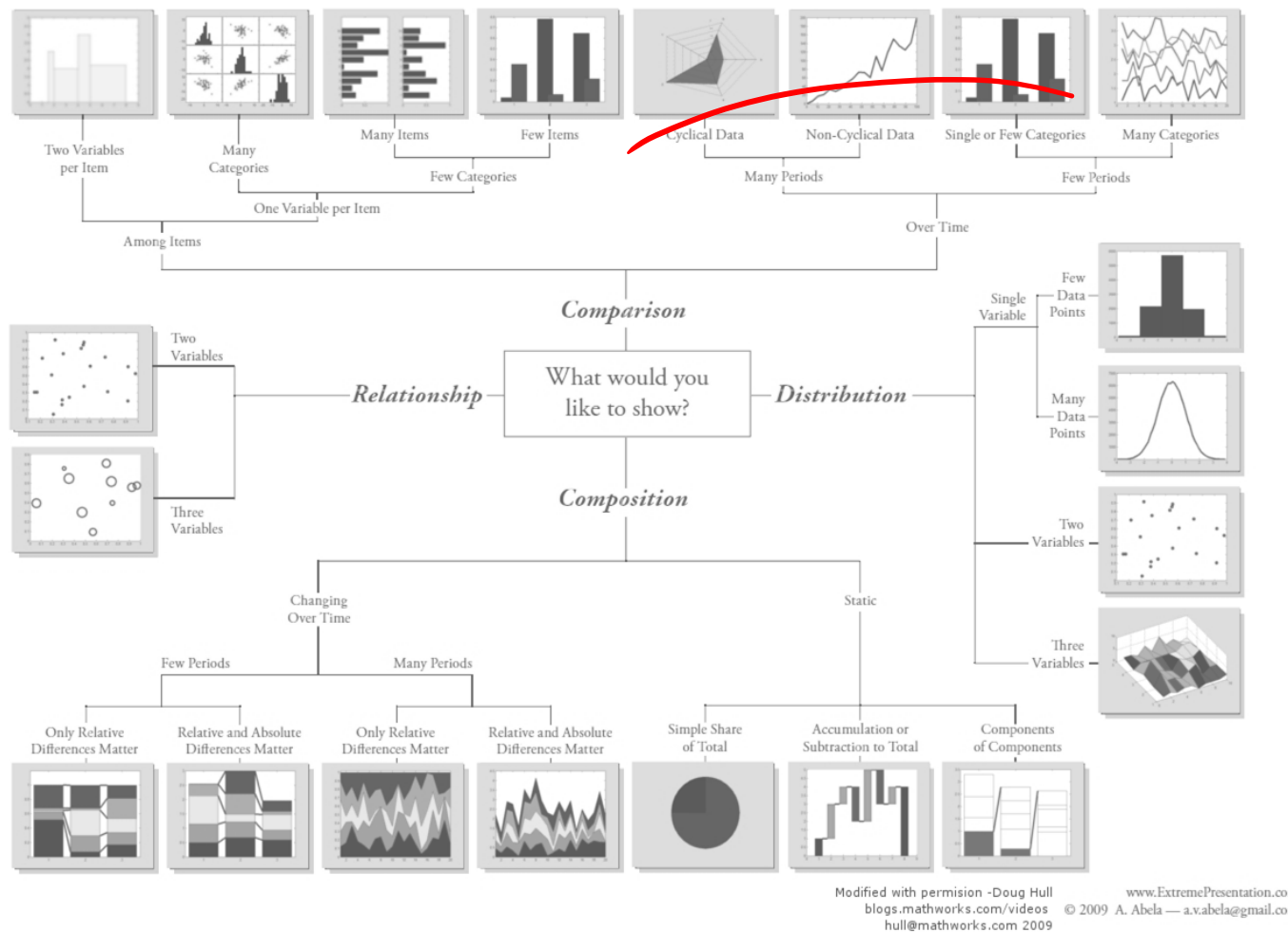
Hidden missing values

The zero values might result from a broken or blocked sensor and might be considered as missing values.



Visualization Types

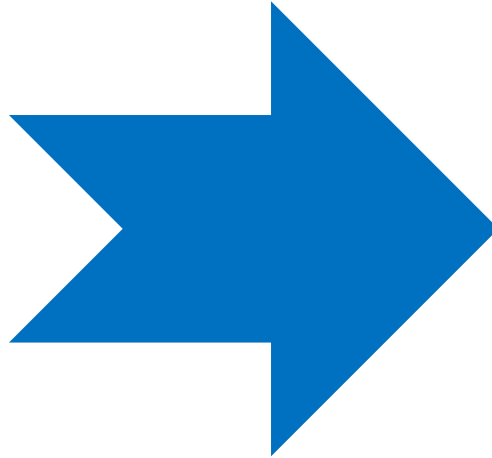
Selecting the 'right' visualization



Graphic representation constitutes one of the **basic sign-systems** conceived by the human mind for the purposes of **storing, understanding, and communicating essential information**. As a "language" for the eye, graphics benefits from the ubiquitous properties of visual perception. As a monosemic system, it forms the rational part of the world of images.
(Bertin, 1983)

*"They must **make sense to the user** and require a **visual language system** that uses colour, shape, line, hierarchy and composition to communicate clearly and appropriately, much like the alphabetic and character-based languages used worldwide between humans."*
(Woolman, 2002)

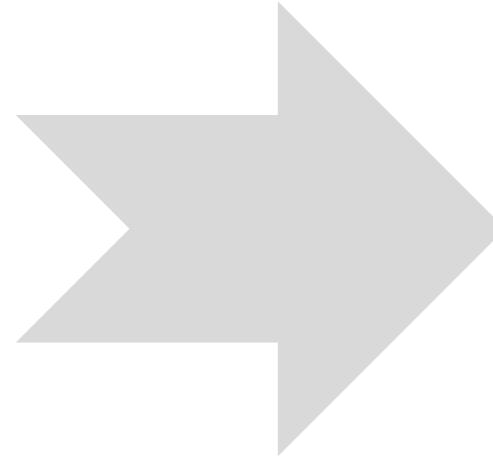
*A good infographic should be **functional** as a hammer, **multilayered** as an onion, and **beautiful** and **true** as an equation (or as a scientific theory). An information graphic must be precise, accurate, efficient, and deep before the designer can apply his or her own visual style or typographical and color preferences to the display.*
(Alberto Cairo, 2012)



Low-dimensional relationships

Univariate Analysis

Bivariate Analysis



Higher-dimensional relationships

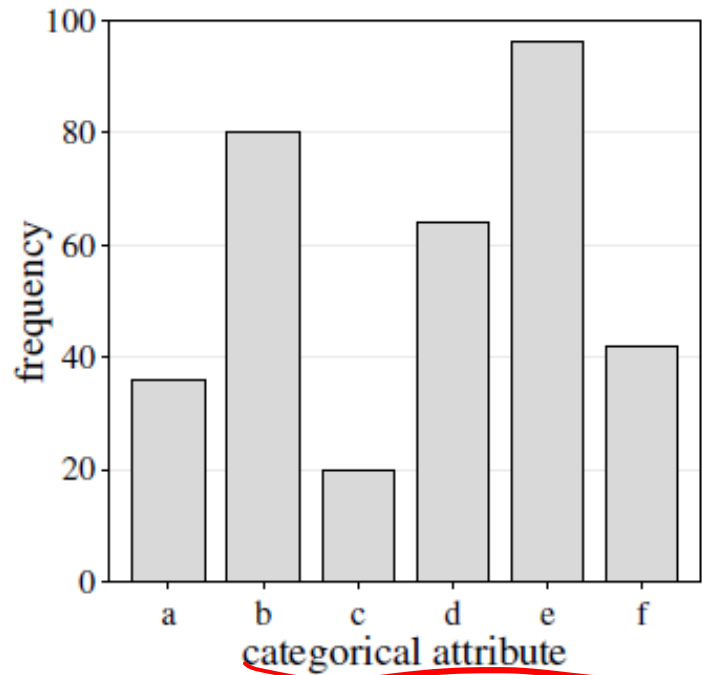
Principal Component Analysis

Parallel Coordinates

Common visualizations

Bar charts and Histograms

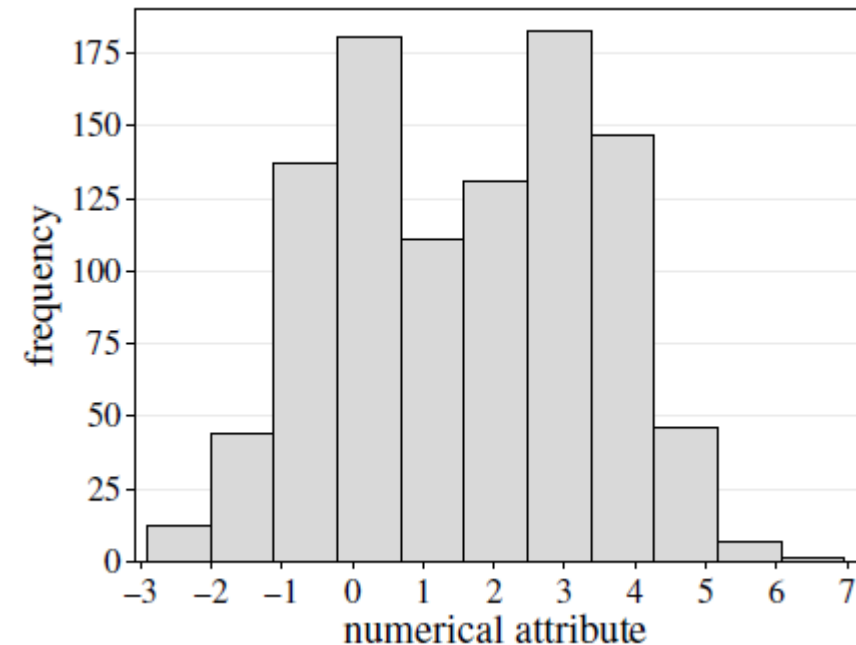
A **bar chart** is a simple way to depict the frequencies of the values of a categorical attribute.



A **histogram** shows the frequency distribution for a numerical attribute.

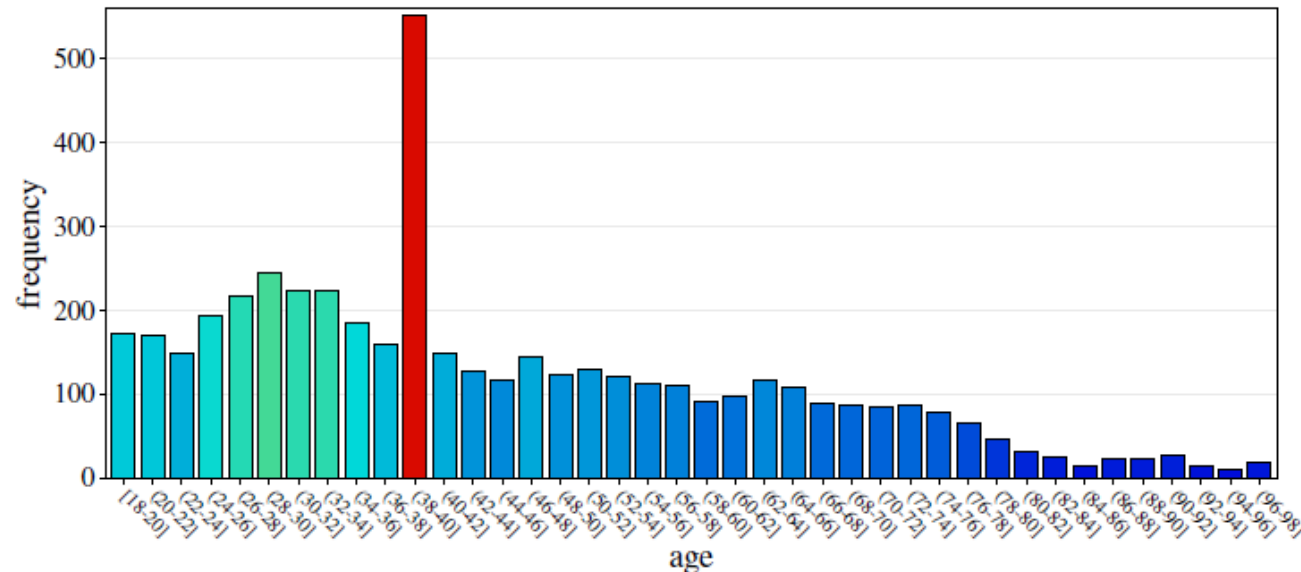
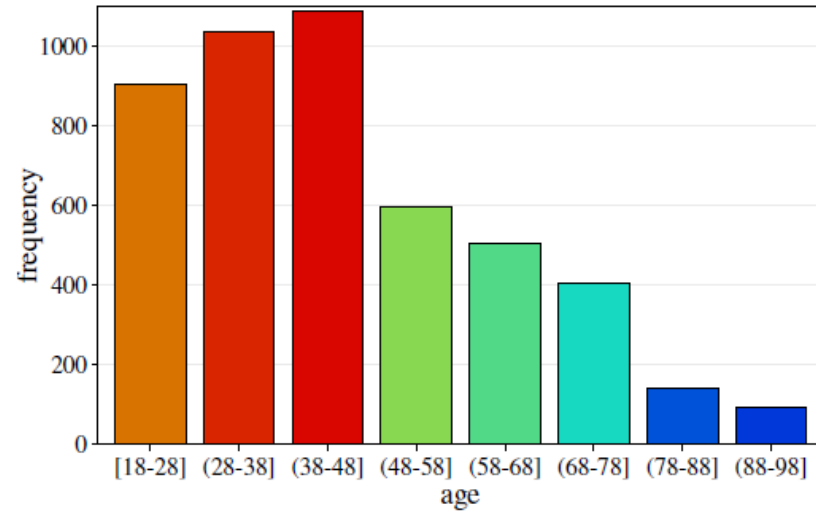
The range of numerical attribute is discretized into a fixed number of intervals (“bins”), usually of equal length.

For each interval, the (absolute) frequency of values falling into it is indicated by the height of a bar.

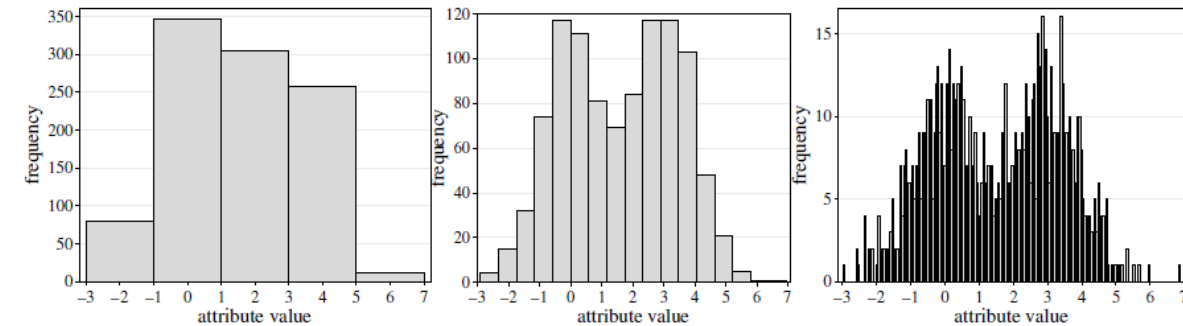
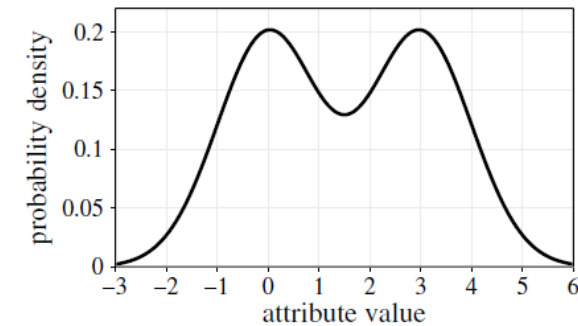


Common visualizations

Histograms: The number of bins is very important.



Three histograms with 5, 17 and 200 bins for a sample from the same bimodal distribution. Sample size is $n = 1000$.



Example data set

Iris data

Collected by E. Anderson in 1935

Contains measurements of four real-valued variables of 150 **iris flowers** of types Iris Setosa, Iris Versicolor, Iris Virginica

- Sepal length [Kelchblatt]
- Sepal widths
- Petal lengths [Blütenblatt]
- Petal widths

The fifth attribute is the name of the flower type

Sepal.Length Sepal.Width Petal.Length Petal.Width Species

```
5.1   3.5   1.4   0.2   Iris-setosa
...
...
5.0   3.3   1.4   0.2   Iris-setosa
7.0   3.2   4.7   1.4   Iris-versicolor
...
...
5.1   2.5   3.0   1.1   Iris-versicolor
5.7   2.8   4.1   1.3   Iris-virginica
...
...
5.9   3.0   5.1   1.8   Iris-virginica
```

Ref.

```
import pandas as pd
# Create DataFrame using Pandas and set Column names
iris = pd.read_csv('irisData.csv', names=['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'])
# Show descriptive statistics on dimensional distributions
print(iris.describe())
# Show histogram
iris.hist(column='sepal_length', bins = (4.0,4.5,5.0,5.5,6.0,6.5,7.0,7.5,8))
```



Iris data set: boxplots

Boxplots are a very compact way to visualize and summarize main characteristics of a sample from a numerical attribute

Line in the middle = median

Box = interquartile range

Whiskers = 1.5 x interquartile range

```
import pandas as pd
import seaborn as sns

iris = pd.read_csv('irisData.csv', names=['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'])
sns.boxplot(x="species", y="sepal_length", data=iris, notch=True)
```



Reminder:

Median:

the value in the middle (for the values given in increasing order)

q%-quantile ($0 < q < 100$):

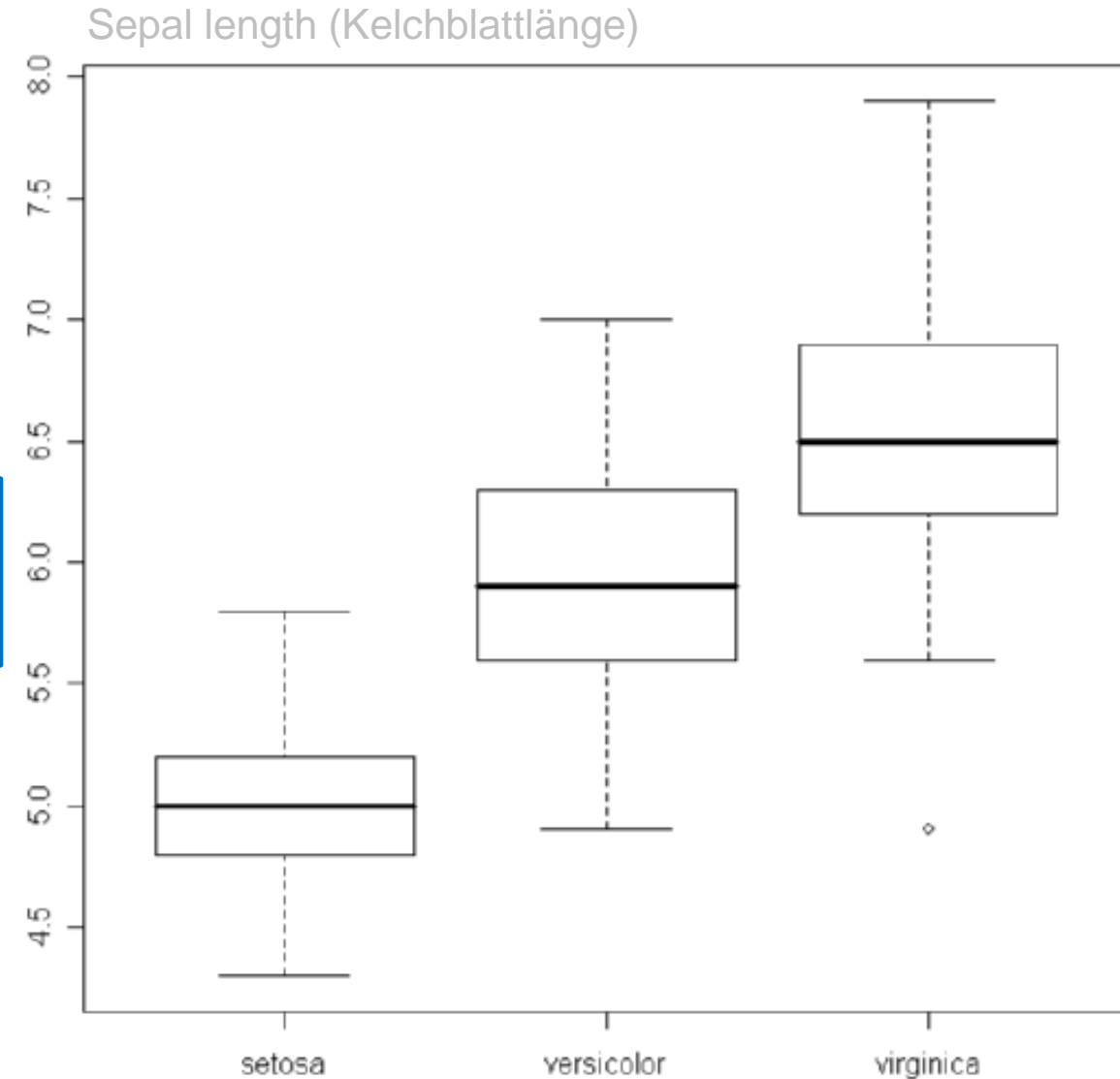
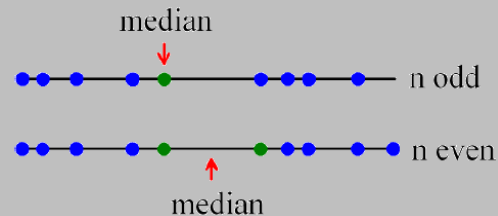
The value for which q% of the values are smaller and 100-q% are larger. The median is the 50%-quantile.

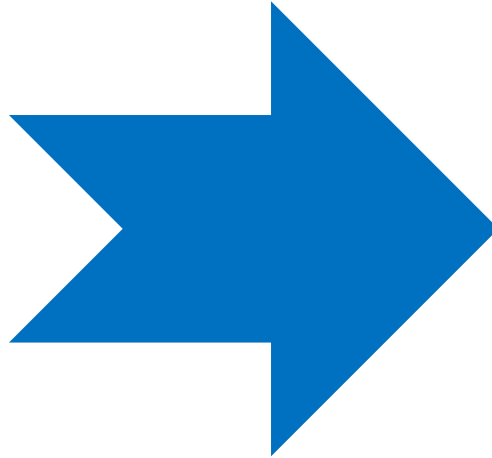
Quartiles:

25%-quantile (1st), median (2nd), 75%-quantile (3rd)

Interquartile range:

3rd quantile – 1st quantile

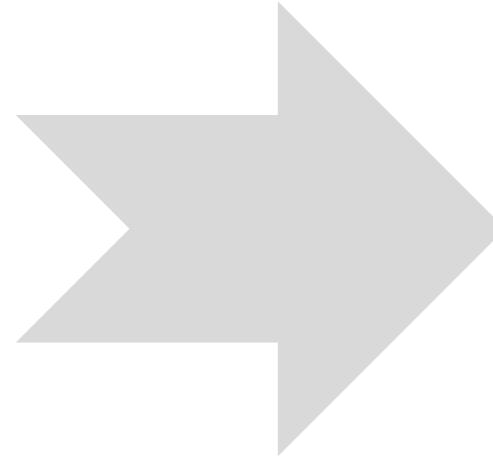




Low-dimensional relationships

Univariate Analysis

Bivariate Analysis



Higher-dimensional relationships

Principal Component Analysis

Parallel Coordinates

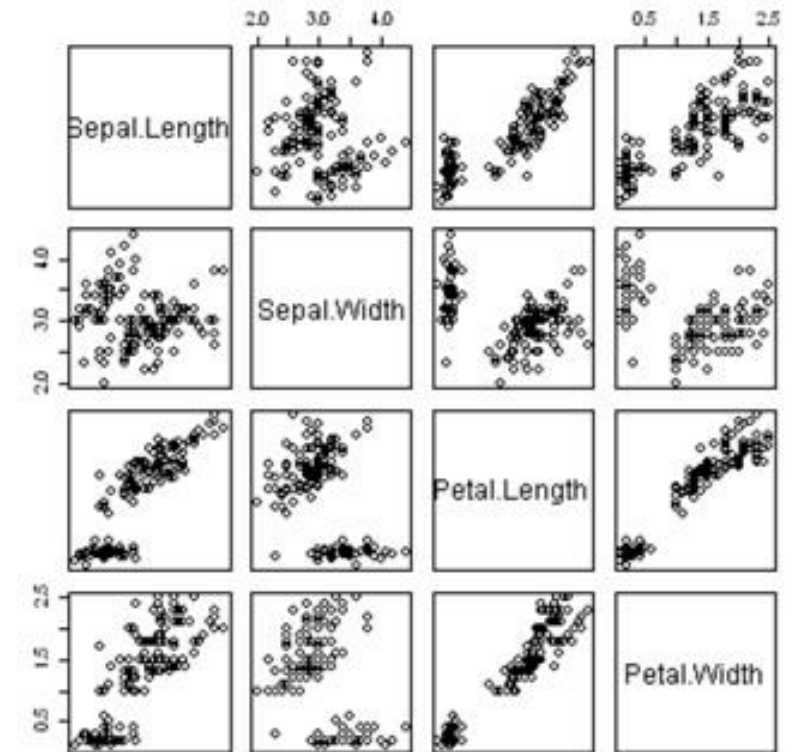
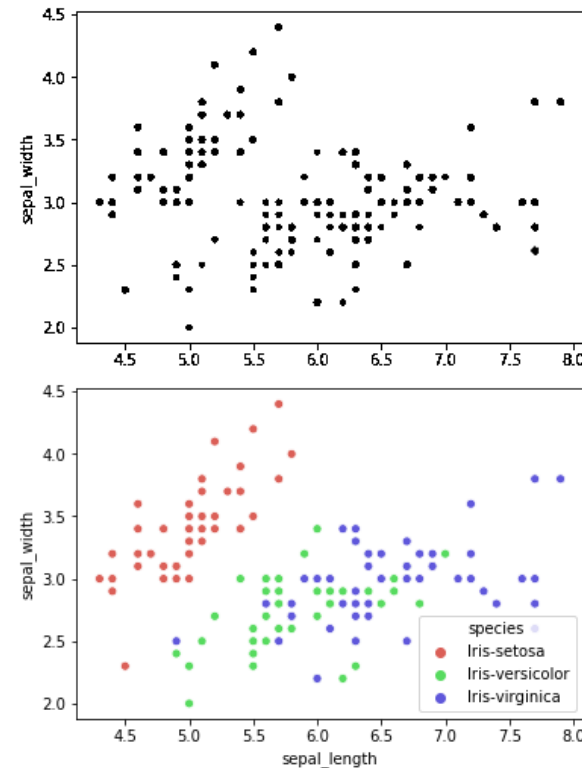
Common visualizations

Scatter plots

Scatter plots visualize two variables in a two-dimensional plot

Each axes corresponds to one variable

Not suited for larger data sets



```
import pandas as pd
import seaborn as sns

iris = pd.read_csv('irisData.csv', names=['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'])

# Describe relationships among variables in scatter plot
# hue: Variable used for color mapping
sns.scatterplot(data=iris, x="sepal_length", y="sepal_width", hue="species", palette="hls")

# Plot pairwise relationships in a dataset.
sns.pairplot(iris, hue="species", palette="hls")
# see https://seaborn.pydata.org/generated/seaborn.pairplot.html
```



Common visualizations

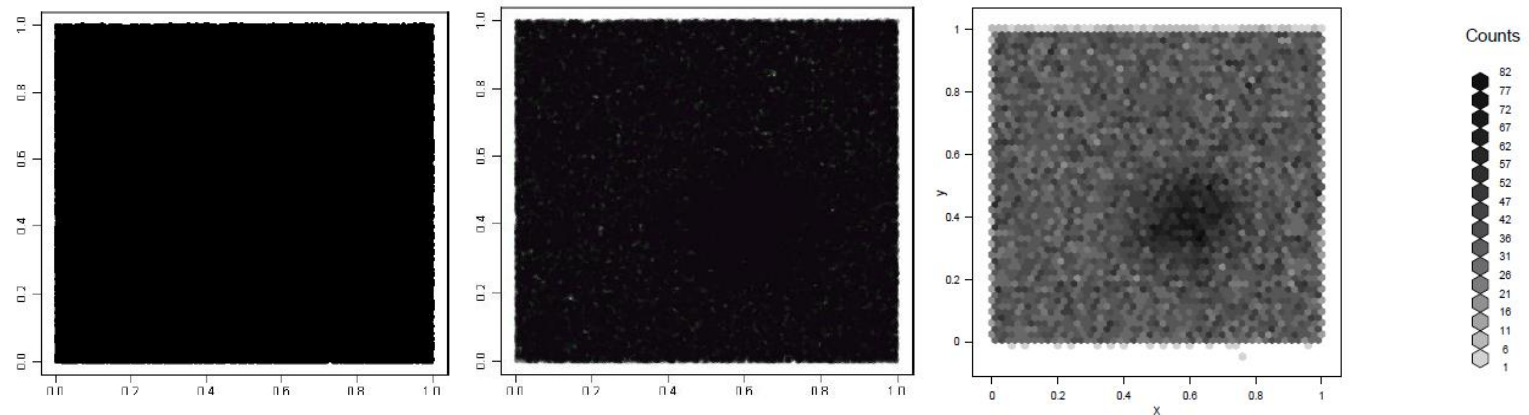
Scatter plots: density

For large data sets, points are plotted over each other and density information is lost.

Left:
1000000 objects

Middle:
Instead of solid points, semitransparent points are plotted

Right:
hexagonal binning. Grey intensity denotes number of points



Iris Data Set Example



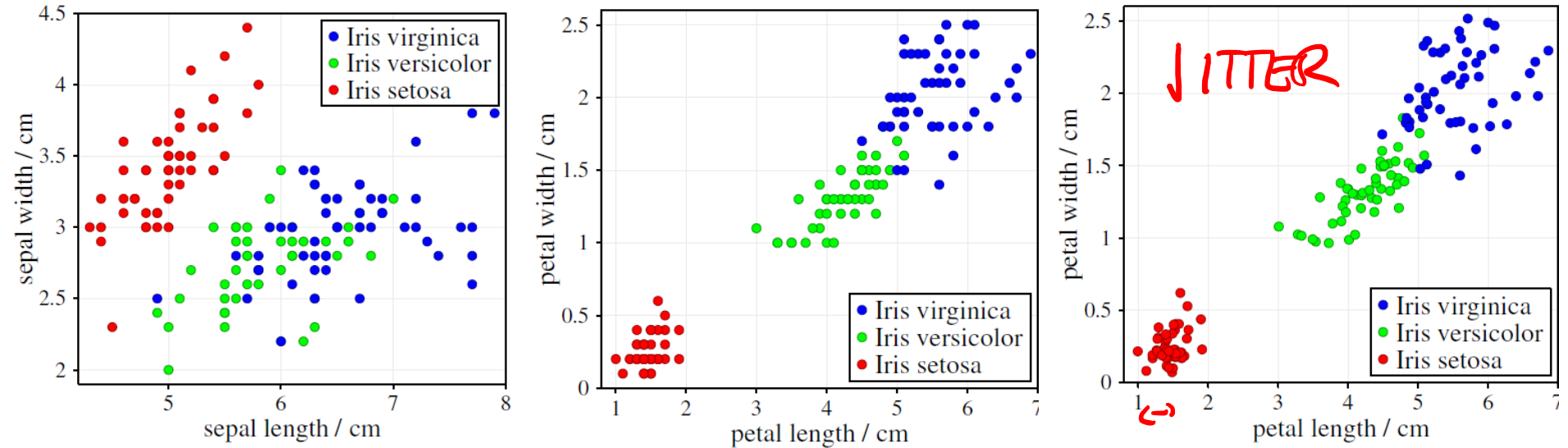
```
import pandas as pd
import seaborn as sns

iris = pd.read_csv('irisData.csv', names=['sepal_length', 'sepal_width',
'petal_length', 'petal_width', 'species'])

iris.plot.hexbin(x="sepal_length", y="sepal_width", gridsize=20)
sns.jointplot(data=iris, x="sepal_length", y="sepal_width", kind="hex",
color="k", joint_kws=dict(gridsize=20), marginal_kws=dict(bins=15, rug=True))
```

Common visualizations

Scatter plots: further elaboration



Scatter plots can be **enriched** with additional information:
color or different symbols incorporate **a third attribute** in the scatter plot.

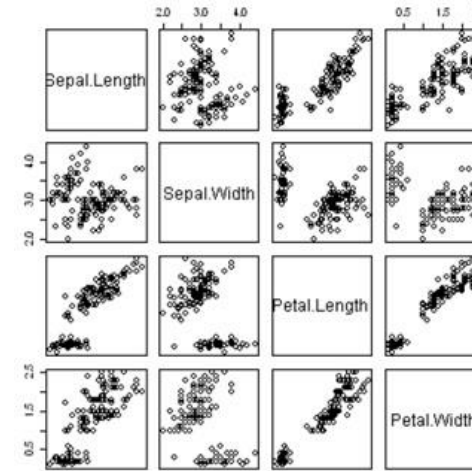
What differences does this reveal?

Data objects with the same values cannot be distinguished in a scatter plot → **jitter** (adding random noise)

Correlation analysis

Scatter plots can “visually” reveal correlations or dependencies between two attributes.

Statistical measures for correlation are a more formal approach to correlation analysis and can be carried out automatically.



```
import pandas as pd

iris = pd.read_csv('irisData.csv', names=...)

print("Show Pearson's correlation:")
print(iris.corr())
#
print()
print("Show Spearman's rho correlation:")
print(iris.corr('spearman'))
#
print()
print("Show Kendall's tau correlation:")
print(iris.corr('kendall'))
```



We briefly sketch...

Pearson's correlation coefficient

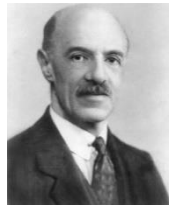
>> [video for explanation](#)

Rank correlation coefficients

>> [video for explanation](#)

Spearman's rho

Kendall's tau



Show Pearson's correlation:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.109369	0.871754	0.817954
sepal_width	-0.109369	1.000000	-0.420516	-0.356544
petal_length	0.871754	-0.420516	1.000000	0.962757
petal_width	0.817954	-0.356544	0.962757	1.000000

Show Spearman's rho correlation:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.159457	0.881386	0.834421
sepal_width	-0.159457	1.000000	-0.303421	-0.277511
petal_length	0.881386	-0.303421	1.000000	0.936003
petal_width	0.834421	-0.277511	0.936003	1.000000

Show Kendall's tau correlation:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.072112	0.717624	0.654960
sepal_width	-0.072112	1.000000	-0.182391	-0.146988
petal_length	0.717624	-0.182391	1.000000	0.803014
petal_width	0.654960	-0.146988	0.803014	1.000000

Please read on your own

$$r_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{(n-1)s_x s_y}$$

The larger the absolute value of the Pearson correlation coefficient, the stronger the **linear relationship** between the two attributes.

$$-1 \leq r_{xy} \leq 1$$

Applicable to **continuous** variables.

Ref.

Please read on your own

Rank correlation coefficients avoid this by ignoring the exact numerical values of the attributes and *considering only the ordering* of the values.

They intend to measure monotonous correlations between attributes, where the monotonous function does not have to be linear.

Example: Aggregate Single Sales (US)

Pos	Artist and Title	Sales estimate	This year
1	Mark Ronson - Uptown Funk	7,470,000	120,000
2	Pharrell Williams - Happy	7,280,000	40,000
3	Katy Perry - Dark Horse	6,230,000	20,000
4	Taylor Swift - Shake It Off	5,840,000	60,000
5	Meghan Trainor - All About That Bass	5,710,000	20,000

ordinal

continuous

Rank correlation coefficients

Spearman's rho

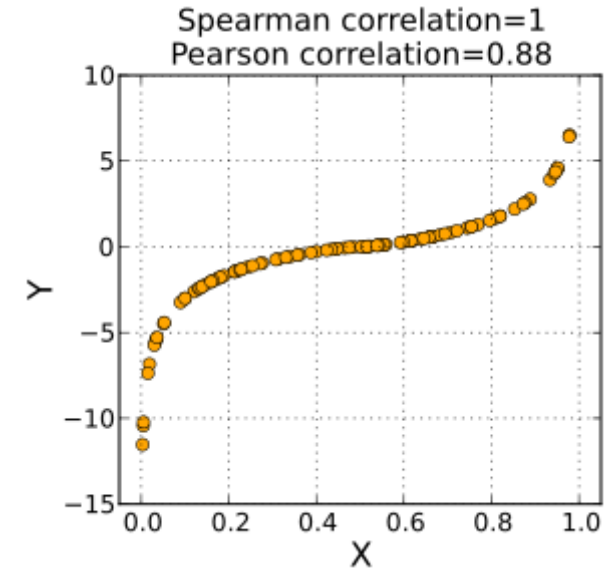
Spearman's rank correlation coefficient (**Spearman's rho**) is defined as

$$\rho = 1 - 6 \frac{\sum_{i=1}^n (r(x_i) - r(y_i))^2}{n(n^2 - 1)},$$

where we sum the deviations between $r(x_i)$ – the rank of value x_i when we sort the list (x_1, \dots, x_n) in increasing order – and $r(y_i)$.

When the rankings of the x - and y -values are exactly in the same order, Spearman's rho will yield the value 1.

If they are in reverse order, we will obtain the value -1.



Spearman's rho makes no assumption on the distribution and is applicable to **continuous** and **discrete** (ordinal) variables.

It is sensitive to large deviations.

Rank correlation coefficients

Kendall's tau

Kendall's tau rank correlation coefficient
(Kendall's tau) is defined as

$$\tau_a = \frac{C - D}{\frac{1}{2}n(n-1)}$$

where C and D denote the numbers of concordant (similar rank order) and discordant pairs with similar ranks, respectively.

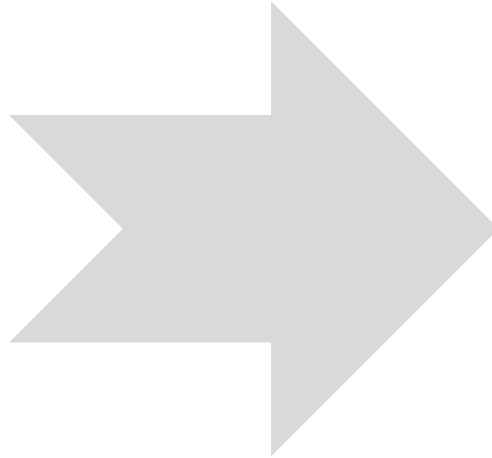
$$C = |\{(i, j) | x_i < x_j \text{ and } y_i < y_j\}|$$

$$D = |\{(i, j) | x_i < x_j \text{ and } y_i > y_j\}|$$

Kendall's tau makes no assumption on the distribution.

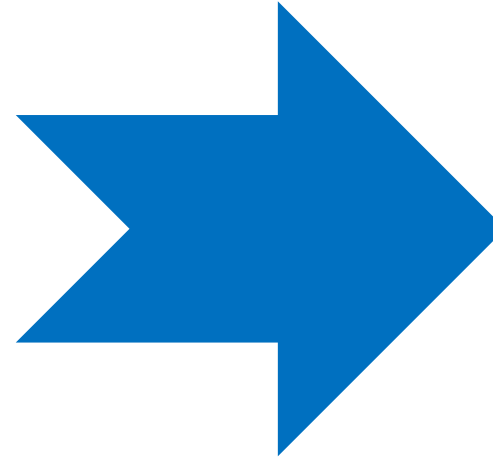
Kendall's tau_a is applicable to **continuous** and **discrete** (incl. ordinal) variables

Less sensitive to errors and discrepancies in the data as Spearman.



Low-dimensional relationships

Univariate Analysis
Bivariate Analysis



Higher-dimensional relationships

Principal Component Analysis
Parallel Coordinates

Outlook I:

Methods for higher-dimensional data

(and an introductory example about the main idea of **Principal Component Analysis**)

General approach for incorporating all attributes in a plot:

There is no unique measure for structure preservation.

Try to preserve as much of the “structure” of the high-dimensional data set when **representing (plotting) the data in two (or three) dimensions**

Define a measure that evaluates lower-dimensional representations (plots) of the data in terms of **how well a representation preserves the original “structure”** of the high-dimensional data set.

Find the representation (plot) that gives the best value for the defined measure.

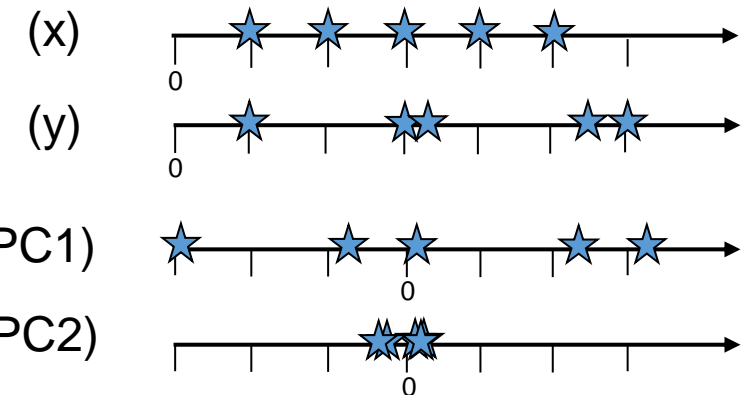
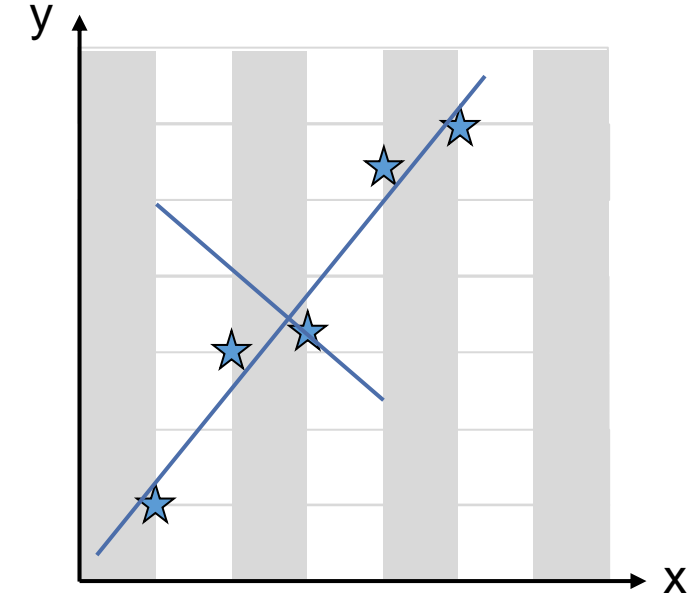
Next Lesson

Freie Universität



Berlin

From \mathbb{R}^2 to \mathbb{R}^1



Next Lesson: More details about PCA

Outlook II:

Data understanding vs. Data preparation

Example: Which attributes should be selected?

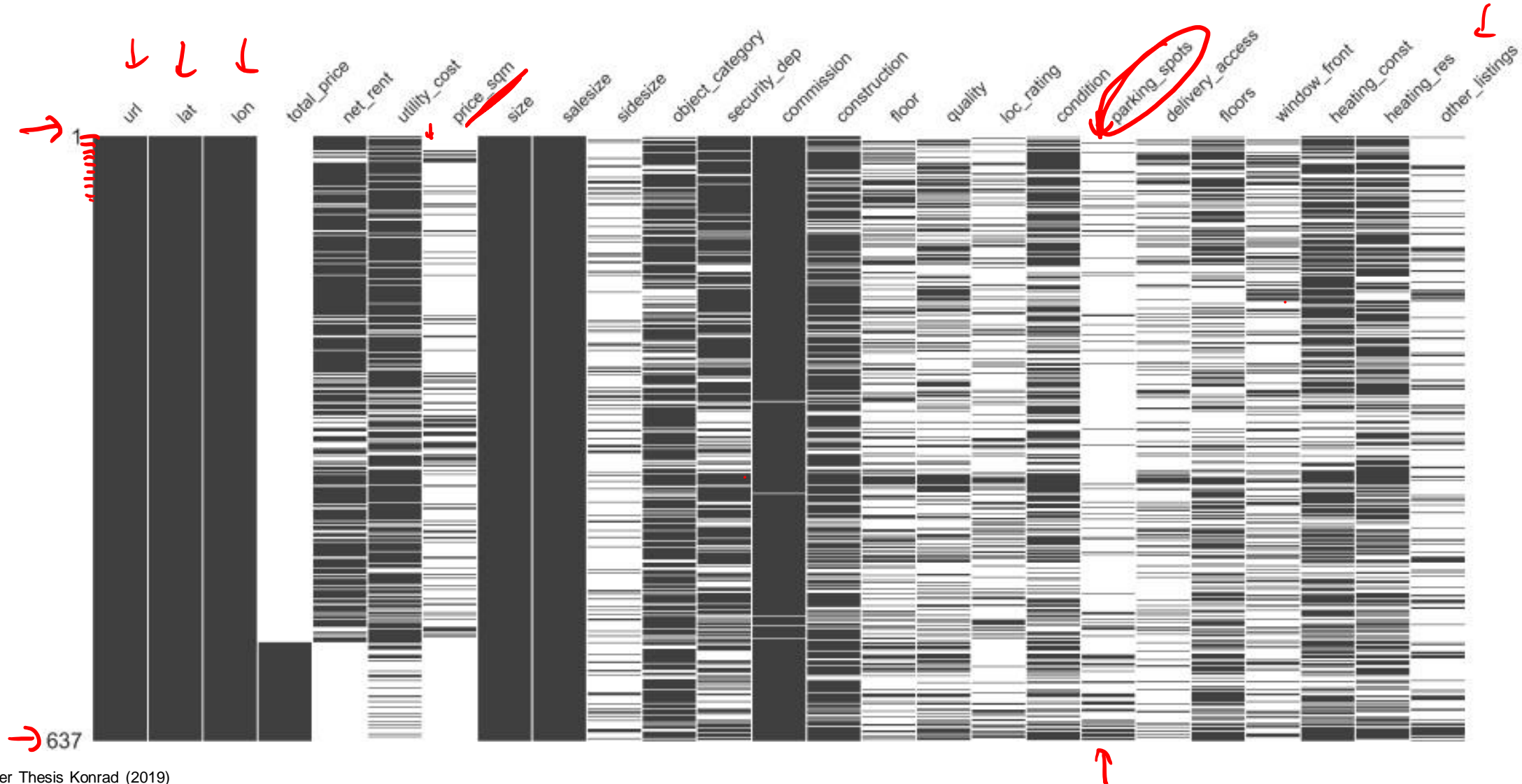
Next Lesson

Freie Universität



Berlin

"null"



Ref. Master Thesis Konrad (2019)

Fragen?

- ✓ Data understanding I
 - ✓ Attribute Understanding
 - ✓ Data Quality
- ✓ Data visualization, correlation analysis (Data understanding II)
- ✓ Low-dimensional relationships
 - ✓ Univariate Analysis
 - ✓ Bivariate Analysis
- Higher-dimensional relationships
 - Principal Component Analysis
 - Parallel Coordinates

Recommended reading

Berthold et al. Chapter 4

Han, J., Kamber, M., Pei, J.: Data Mining: Concepts and Techniques. Morgan Kaufmann, 2011

Todos for next Week

- Python-Analytics – Chapter 2
Kursmaterial > Readings/Übungen > Python Übungen – Jupyter
- Please try the sample code on the Iris dataset on your own.
- Please get familiar with the PCA (incl. excursus) on the following slides.

Principal Component Analysis (PCA)

Structure preservation through variance in data set

Next Lesson

Freie Universität



Berlin

PCA compresses a large data set to capture the *essence of the original data* through linear transformation

PCA uses the **variance in the data** set as the structure preservation criterion.

Assumption: Large variances describe interesting dynamics, smaller noise.

PCA constructs **a projection** from the high-dimensional space to a lower-dimensional space (plane or hyperplane) using only the most relevant dimensions

PCA preserves as much of the original variance of the data when projected to a lower-dimensional space

(Sample) variance for a numerical attribute:

$$s^2 = \frac{1}{n-1} \sum_{i=1}^n (x_i - \bar{x})^2 = \frac{n \sum_{i=1}^n x_i^2 - \left(\sum_{i=1}^n x_i \right)^2}{n(n-1)}$$

Principal Component Analysis

Procedure: Objective

The data points are first **centered around the origin** by subtracting the mean values

Objective:

find a projection in the form of a linear mapping given by $y = M(x - \bar{x})$, where M is a $q \times m$ matrix such that the **variance** of the projected data $y_i = M(x_i - \bar{x})$ is **maximized**

($2 \times m$ for projections to a plane)

PCA uses the **covariance matrix** which holds information on spread (variance) and orientation (covariance)

$$\Sigma = \begin{bmatrix} \sigma(x, x) & \sigma(x, y) \\ \sigma(y, x) & \sigma(y, y) \end{bmatrix}$$

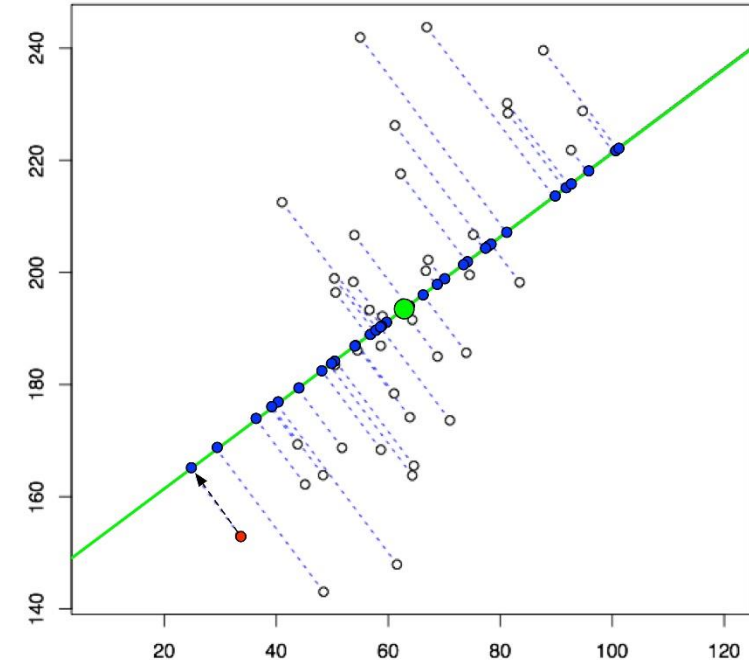
Next Lesson

Freie Universität



Berlin

Projecting 2 dimensions on 1



See excursus for in-depth information

Principal Component Analysis

Procedure: Problem

Problem:

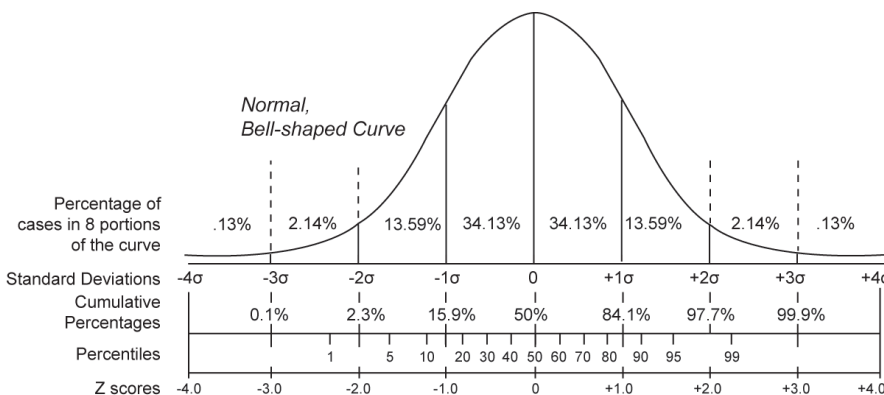
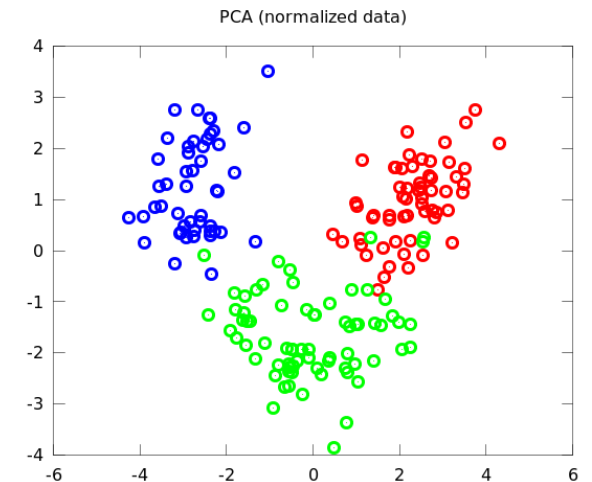
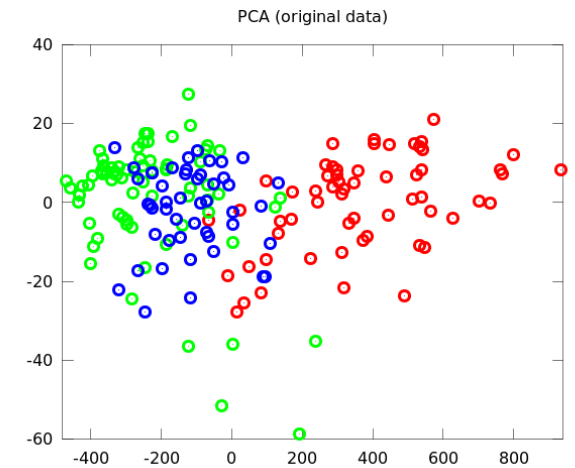
Without restriction for the matrix M , the entries in M can be chosen arbitrary large so that the data are not only projected, but also **scaled**, leading to an arbitrary large variance of the projected data.

We introduce **constraints** such that the matrix M is only a projection:

The row v_i of the matrix $M = (v_1, \dots, v_q)$ must be **normalized**, i.e., $\|v_i\| = 1$.

Next Lesson

Freie Universität



Usually, the data should be **zero-score standardized** ($x \rightarrow \frac{x - \hat{\mu}_x}{\hat{\sigma}_x}$) to ensure that all attributes contribute equally to the overall variance (with $\hat{\mu}_x$ being the mean value and $\hat{\sigma}_x$ the sample standard deviation of attribute X , z-score: numeric distance of x in standard deviations from mean)

Principal Component Analysis

Choosing principal components

Solution of the constraint optimization problem:

The projection matrix M is given by $M = (v_1, \dots, v_q)$,

where the **principal components** v_1, \dots, v_q are the *normalized eigenvectors of the covariance matrix* of the attributes in the data set

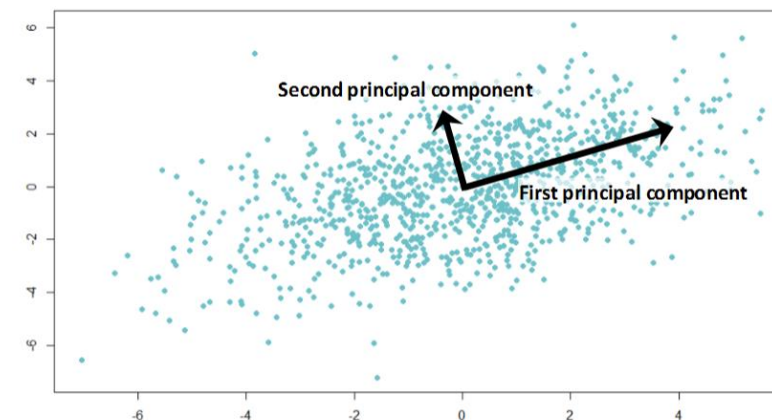
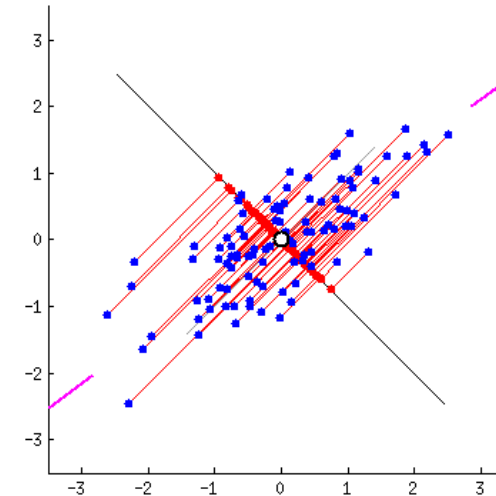
$$\sigma_{jk} = \frac{1}{n-1} \sum_{i=1}^n (x_{ij} - \bar{x}_j)(x_{ik} - \bar{x}_k)^T$$

for the q **largest eigenvalues** $\lambda_1 \geq \dots \geq \lambda_q$.

λ is called an eigenvalue of a matrix A , if there is a non-zero vector v such that $Av = \lambda v$ holds. The vector v is called eigenvector (direction of the data) to the eigenvalue λ (magnitude of its spread).

Next Lesson

Freie Universität



Principal Component Analysis

Dimension reduction

Let $\lambda_1 \geq \dots \geq \lambda_m$ be the eigenvalues of the covariance matrix.

When we project the data to the first q principal components v_1, \dots, v_q corresponding to the eigenvalues $\lambda_1, \dots, \lambda_q$, this projection will preserve a fraction of the variance of the original data.

$$\frac{\lambda_1 + \dots + \lambda_q}{\lambda_1 + \dots + \lambda_m}$$

Iris data set:

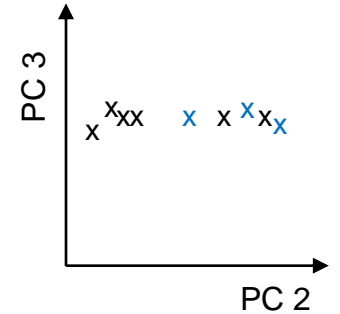
	PC1	PC2	PC3	PC4
Proportion of variance	0.73	0.229	0.0367	0.00518
Cum. proportion	0.73	0.958	0.9948	1.00000

Next Lesson

Freie Universität



Omid principal components which explain little variance in the data, like...



PCA – Iris data set example (1/2)

Next Lesson

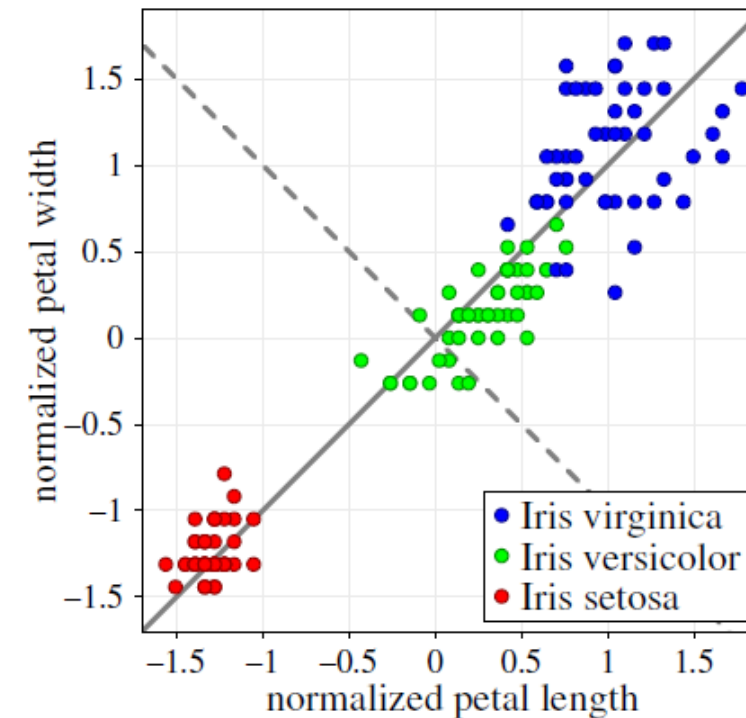
Freie Universität



Berlin

PCA applied to the **Iris data set** restricted to the (normalized) petal length and width

The principal components are always *orthogonal*



PCA – Iris data set example (2/2)

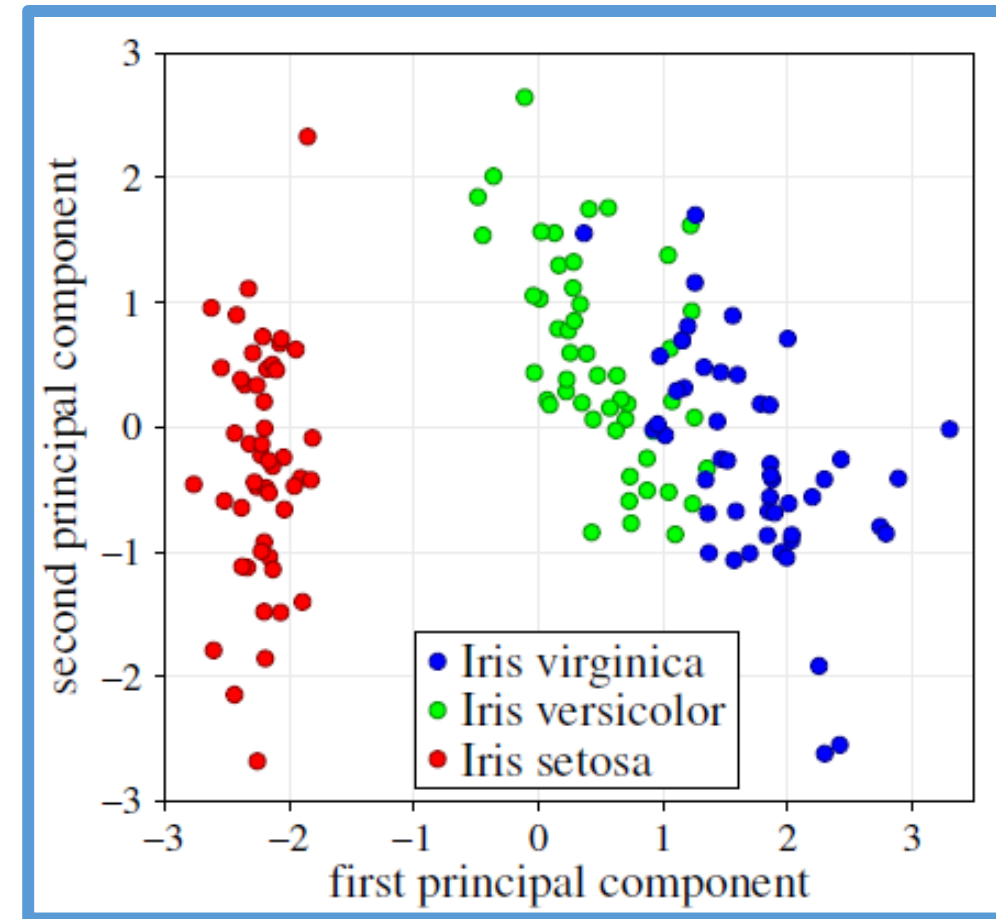
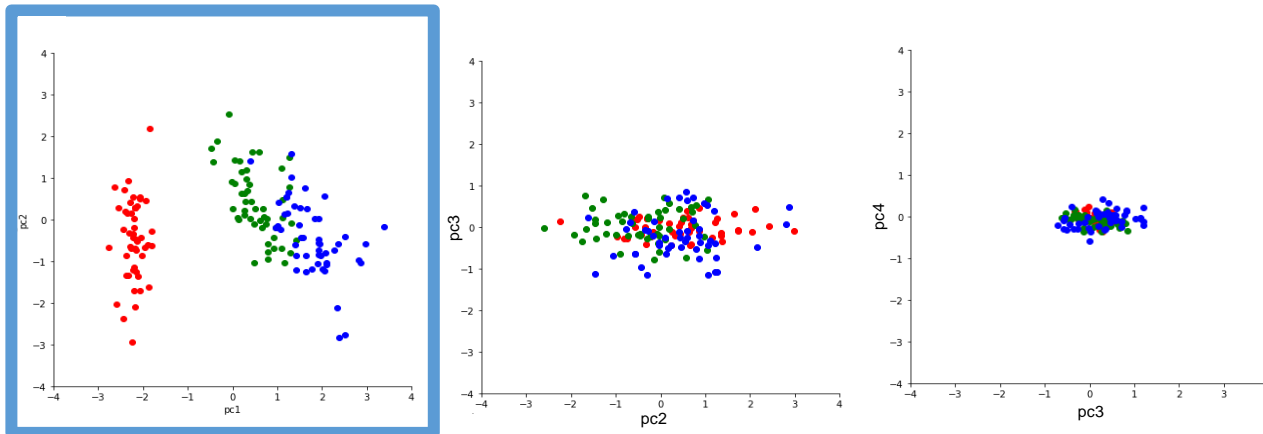
Next Lesson

Freie Universität



Berlin

Projection to the first two principal components of PCA taking all four numerical attributes into account



Original data is **reconstructable** from the principal components

Ref.

PCA – Chessboard example (1/2)

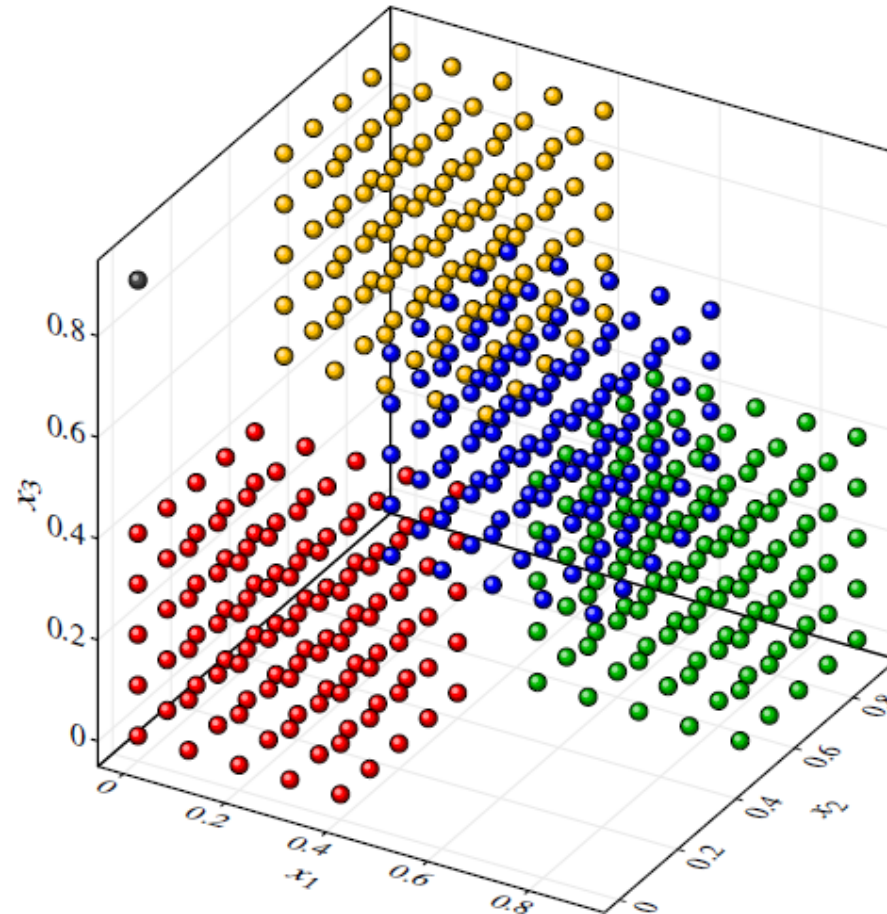
Next Lesson

Freie Universität



Berlin

An artificial data set filling a cube in a chessboard-like manner



Ref. Berthold et al. (2010)

PCA – Chessboard example (2/2)

Next Lesson

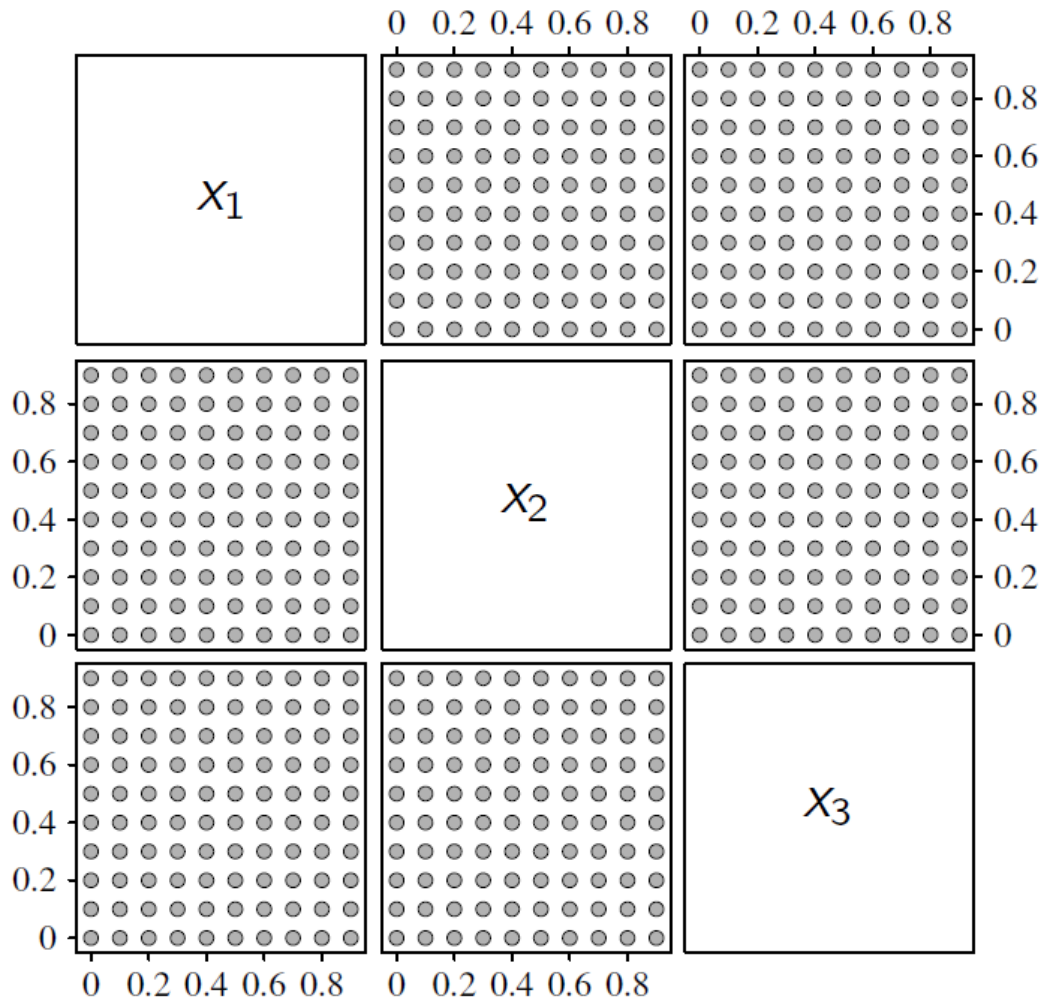
Freie Universität



Berlin

Scatter plots

Is data uniformly distributed over the grid?



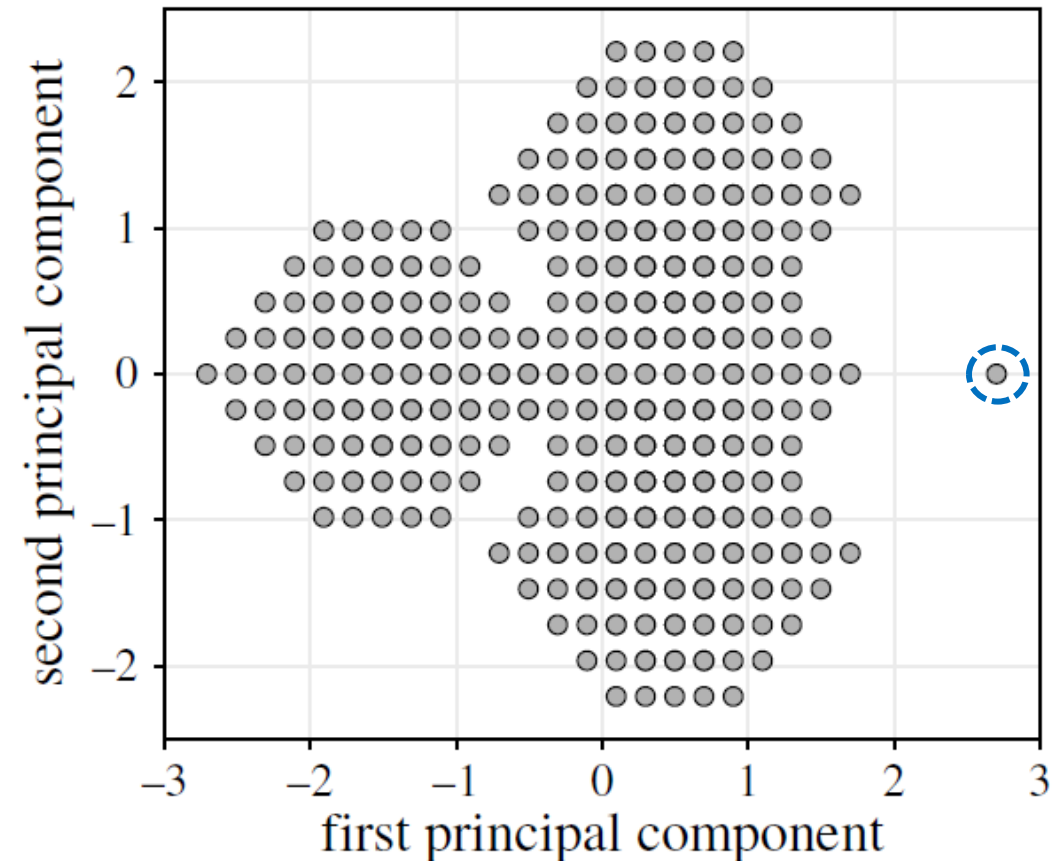
Ref. Berthold et al. (2010)

Projection to the first two **principal components**

Data is not uniformly distributed.

There is a pattern in the data set.

Data can be recreated from PCA.



Principal Component Analysis – Iris Data Set

Python Example

```
from sklearn.decomposition import PCA
from sklearn.preprocessing import scale

iris = pd.read_csv('irisData.csv', names=['sepal_length', 'sepal_width', 'petal_length', 'petal_width', 'species'])
#select only metric data
raw_iris = iris[['sepal_length', 'sepal_width', 'petal_length', 'petal_width']]

#center data to mean
norm_iris = scale(raw_iris)

#create pca with 4-dimensions
pca = PCA(n_components=4)
#pca data
pca_iris = pca.fit_transform(norm_iris)

print(pca.explained_variance_)
print(pca.explained_variance_ratio_)
print(pca.explained_variance_ratio_.cumsum())

#visualize pcas
vis_iris = pd.DataFrame(pca_iris, columns=['pc1', 'pc2', 'pc3', 'pc4'])
vis_iris['species'] = iris['species']
g = sns.FacetGrid(vis_iris, hue='species', palette=['r','g','b'], ylim=(-4,4), xlim=(-4,4), height = 6)
g.map(plt.scatter, 'pc1', 'pc2')
plt.show()

g = sns.FacetGrid(vis_iris, hue='species', palette=['r','g','b'], ylim=(-4,4), xlim=(-4,4), height = 6)
g.map(plt.scatter, 'pc2', 'pc3')
plt.show()

g = sns.FacetGrid(vis_iris, hue='species', palette=['r','g','b'], ylim=(-4,4), xlim=(-4,4), height = 6)
g.map(plt.scatter, 'pc3', 'pc4')
plt.show()
```

Ref.

Example in Python

