Elements of Machine Learning & Data Science

Winter semester 2025/26

Lecture 18 – Data Quality and Preprocessing

12.01.2026

Prof. Bastian Leibe

slides by Prof. Wil van der Aalst

Elements of Machine Learning & Data Science

Winter semester 2025/26

Part 3: Empirical Analysis and Performance Optimization

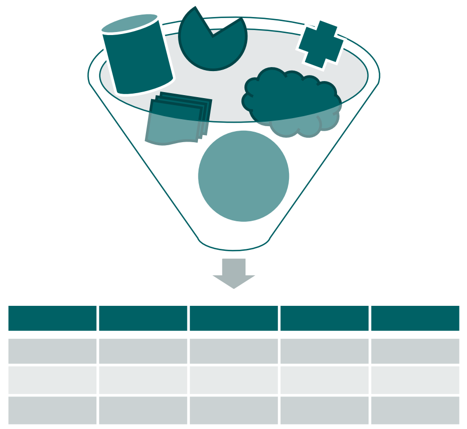
Content by Prof. Holger Hoos
Chair for AI Methodology (AIM)

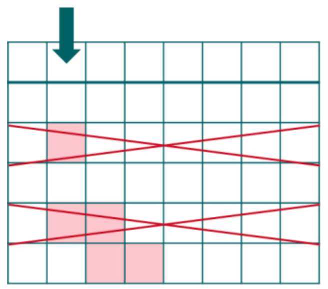
Content by Prof. Wil van der Aalst
Chair of Process and Data Science (PADS)

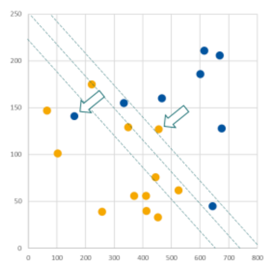
Empirical Analysis and Performance Evaluation Topics

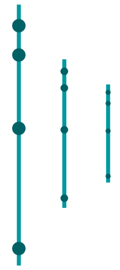
15. **Data Quality and Preprocessing**
16. Responsible Data Science
17. Evaluation
18. Performance Optimization

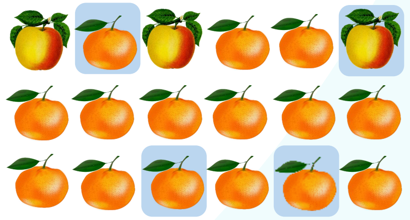
Preprocessing


Data Extraction


Missing Values


Outliers


Normalization

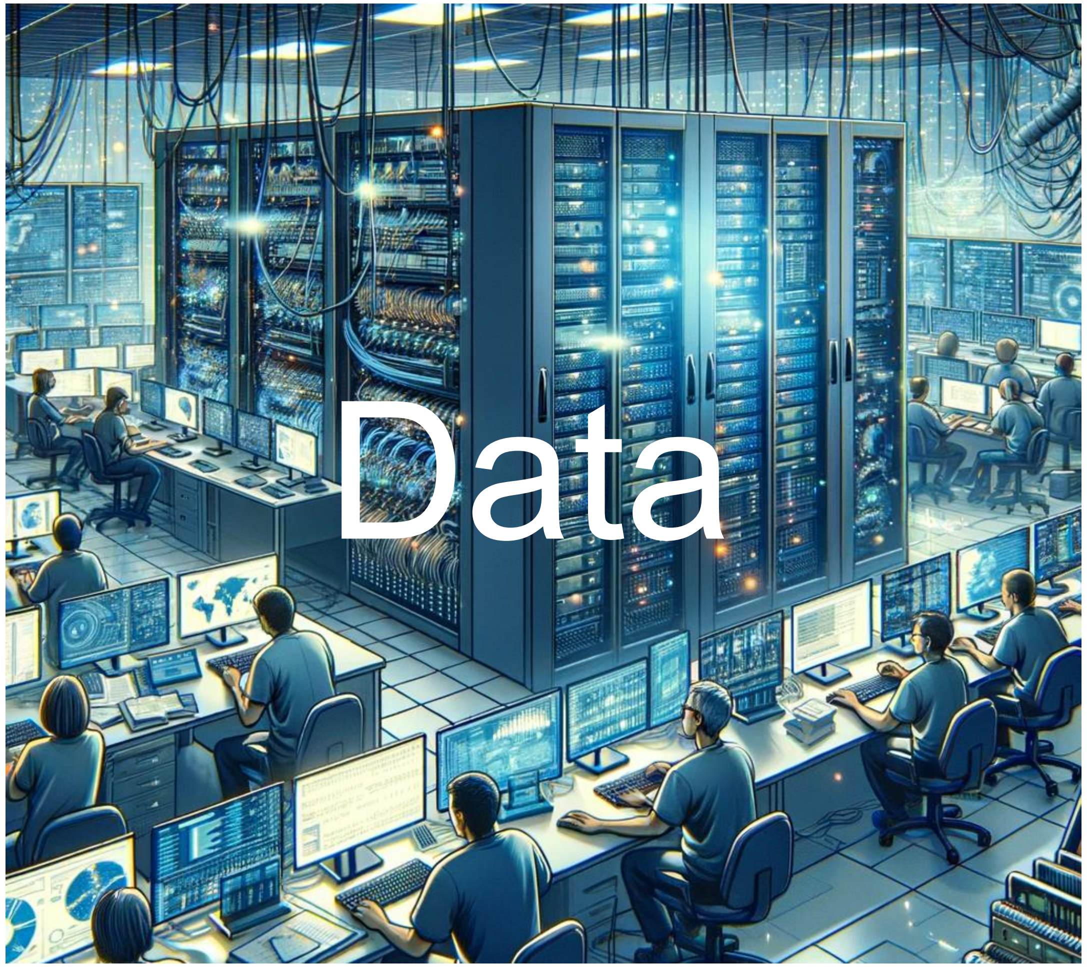

Subset Selection

Let’s Take A Step Back: How to Get the Data?

80/20

It is not uncommon that 80% of the effort/time in a data science project is devoted to finding, extracting, cleaning, and transforming the data. Only 20% is concerned with analysis.

The Two Biggest Hurdles in Practice:
Getting the Data and Implementing Changes



Generated using DALL E3

Real-Life Examples

CIO of a US bank: “We reduced the number of applications from 12.000 to 8.000” : -)

Tables may have hundreds of columns (e.g. EKPO has > 300 fields).

An SAP installation has hundreds of thousands of tables.

EKPO – Purchasing Document Item
#1 MANDT – Client
#2 EBELN – Purchasing Document Number
#3 EBELP – Item Number of Purchasing Document
#4 LOEKZ – Deletion indicator in purchasing document
#5 STATU – RFQ status
…
#299 POL_ID – Order List Item Number
#300 CONS_ORDER – Purchase Order for Consignment

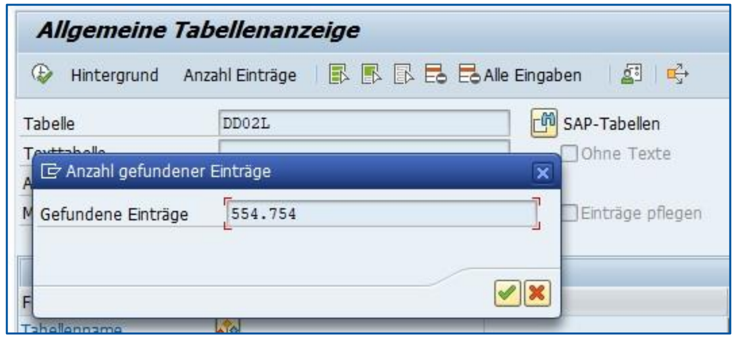

Allgemeine Tabellenanzeige

Tabelle: DD02L

Anzahl gefundener Einträge

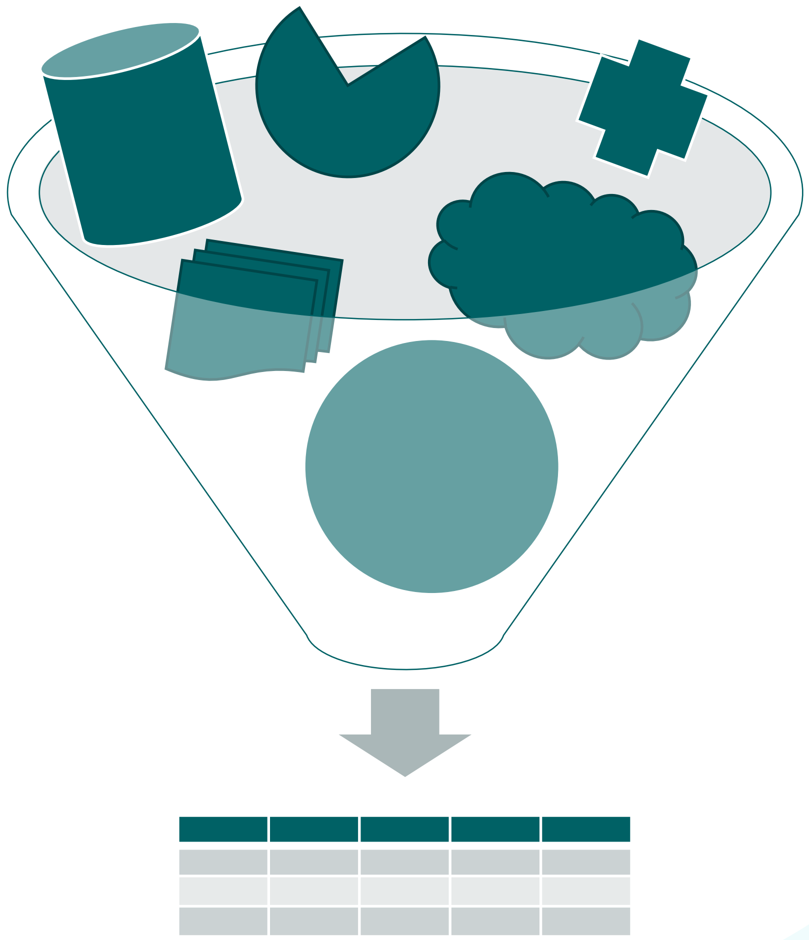
Gefundene Einträge: 554.754

DD02L is the SAP table for SAP Tables.

Organizations such as Siemens have 70 SAP installations.

Data Quality & Preprocessing

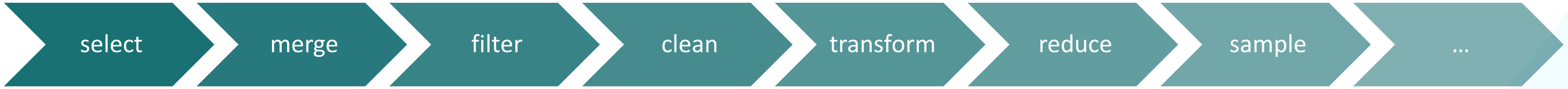
1. **Introduction**
2. Missing Values
3. Outliers
4. Transformation & Normalization
5. Reduction



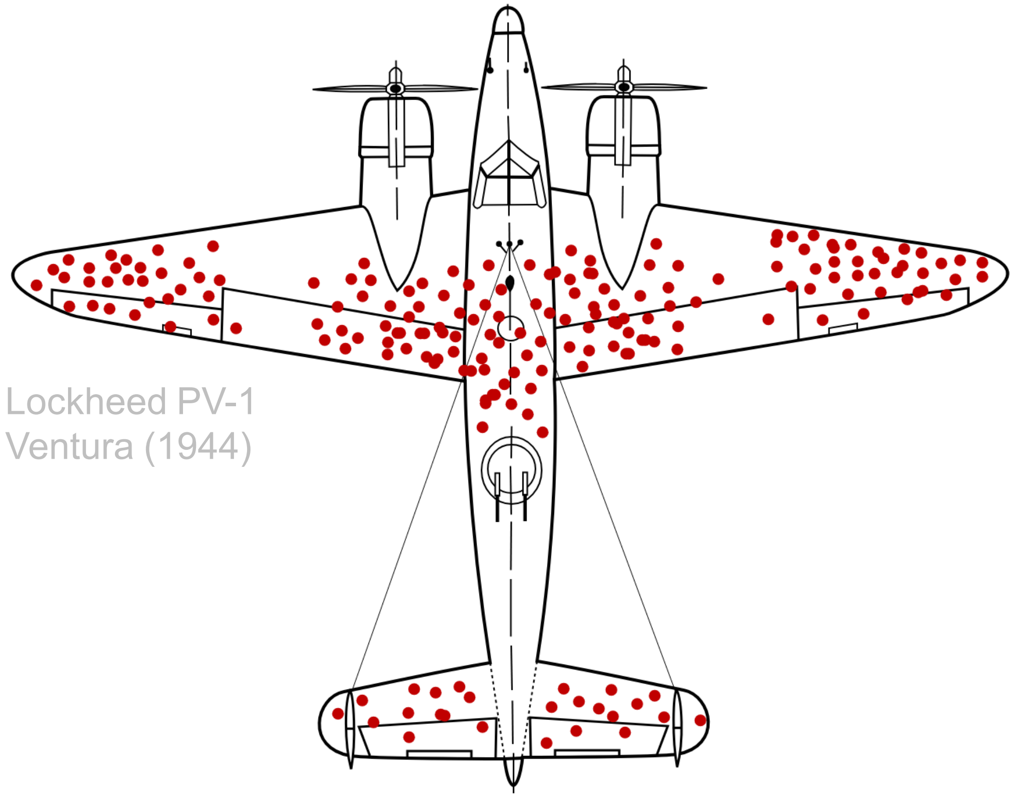
Data Science Pipeline

- Garbage in, garbage out
- Possible problems (big data, security), errors (data quality), biases (e.g., survivorship bias) everywhere
- Problems, errors and biases propagate

Goal: increase data quality and modify the data to suit the analysis question and applied techniques



Example: Survivorship Bias



Lockheed PV-1
Ventura (1944)

https://en.wikipedia.org/wiki/Survivorship_bias

A Canadian study in 2011 revealed that 97.4 percent of Porsches from the last 25 years are still on the road.

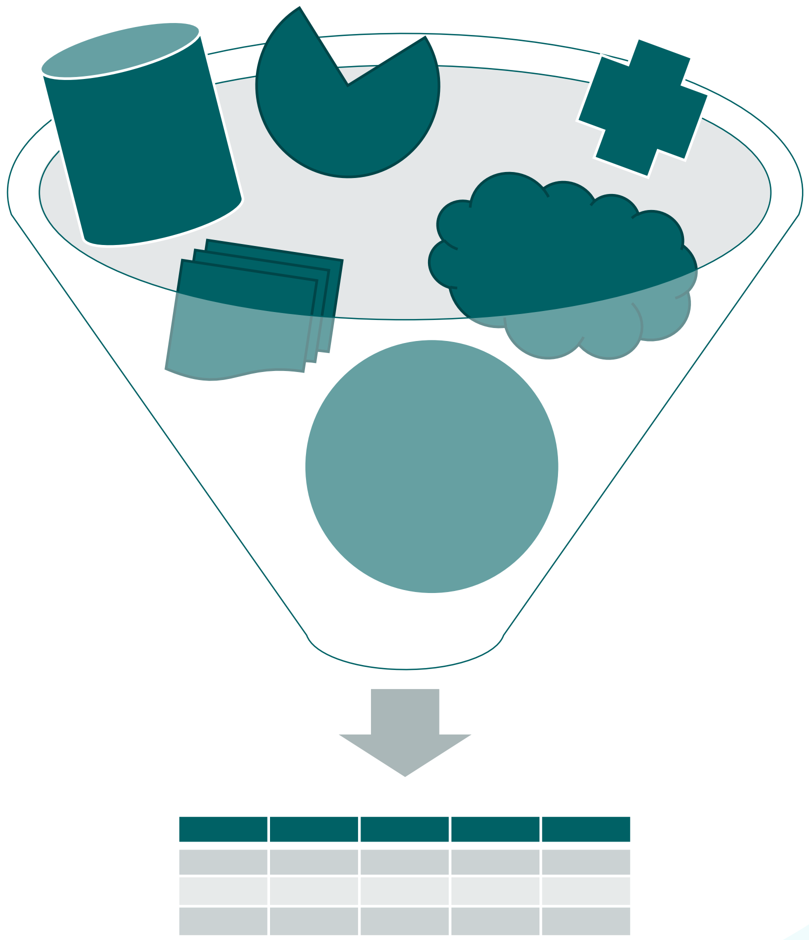
Data Quality Aspects

- **Accuracy** (wrong value)
- **Completeness** (missing value)
- **Consistency** (different conventions/formats)
- **Timeliness** (outdated values)

Name	Age	Siblings	Date of Admission
Sara Johnson	55	0	30.09.2022
NAME	17		23-11-22
Smith	28	2	8/24/22
Emma Miller	2	56	May 10th, 22
Jones	187	3	220701
…	…	…	

Data Quality & Preprocessing

1. Introduction
2. **Missing Values**
3. Outliers
4. Transformation & Normalization
5. Reduction



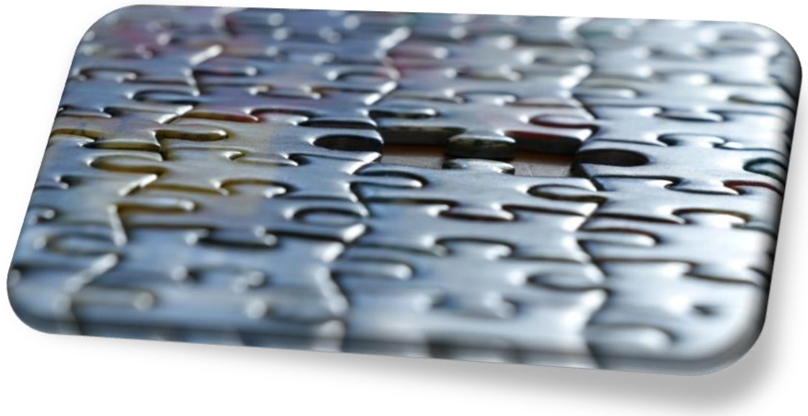
Detecting Missing Values

Missing values may be obvious…

- Empty value
- NaN / NA

… or may be disguised!

- Default value
- Invalid value



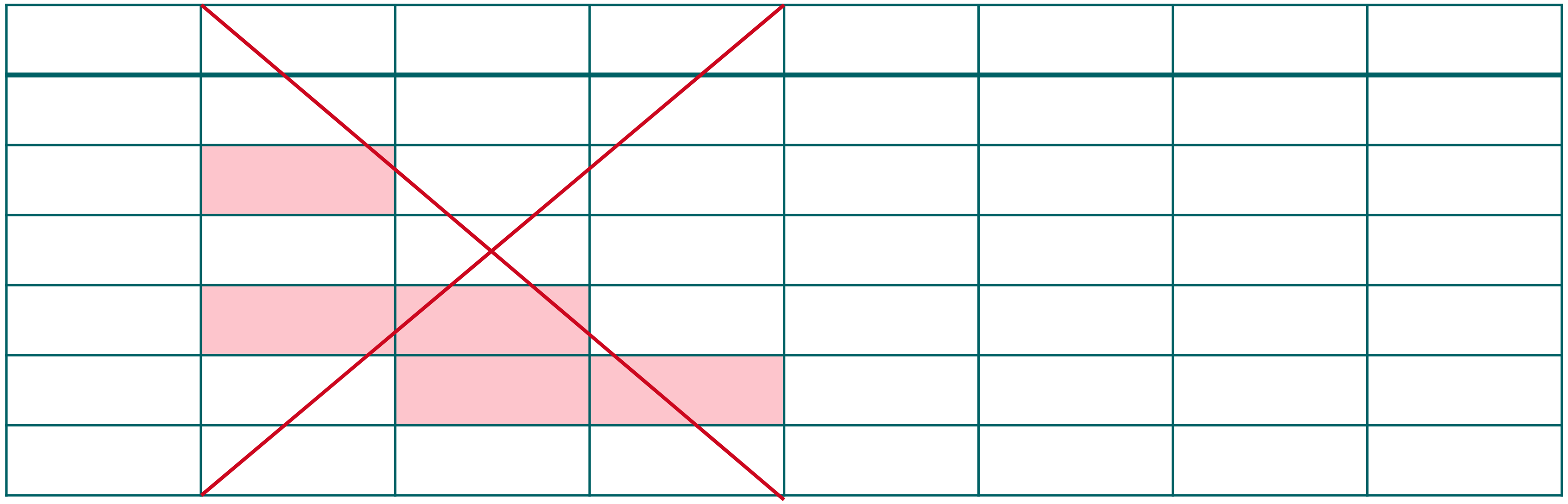
Handling Missing Values

1) Fill in manually
2) Ignore
3) Fill in a derived value

Handling Missing Values: Ignore

Discard the feature

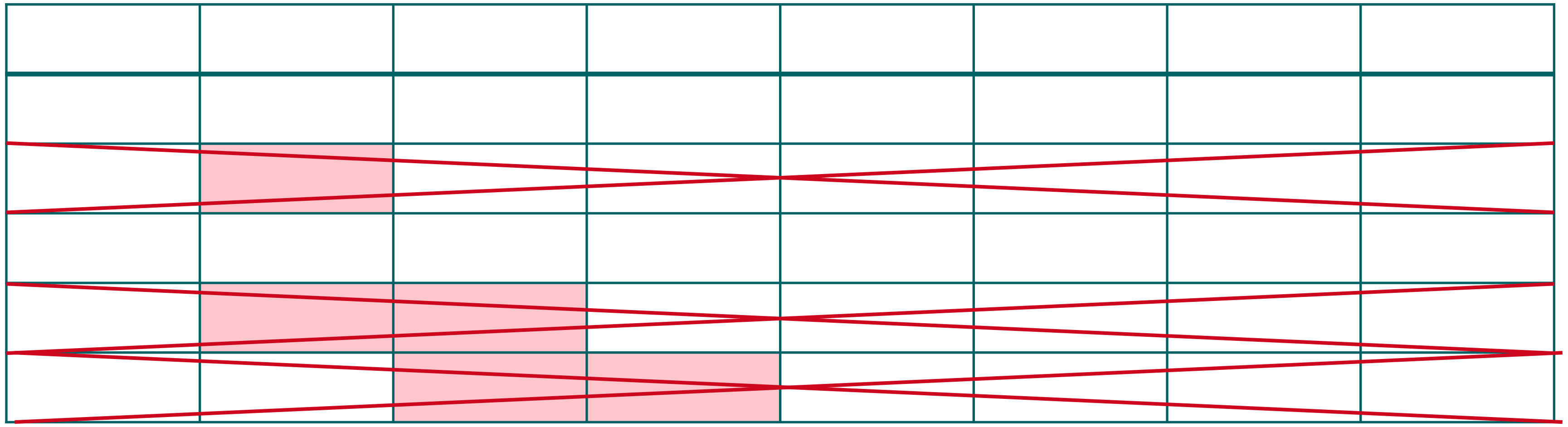
- The whole feature is removed from the data
- Usually done if the number of missing values is too large to allow meaningful analysis (as a rule of thumb, if more than 60 % of the feature values are missing)



Handling Missing Values: Ignore

Discard the instance

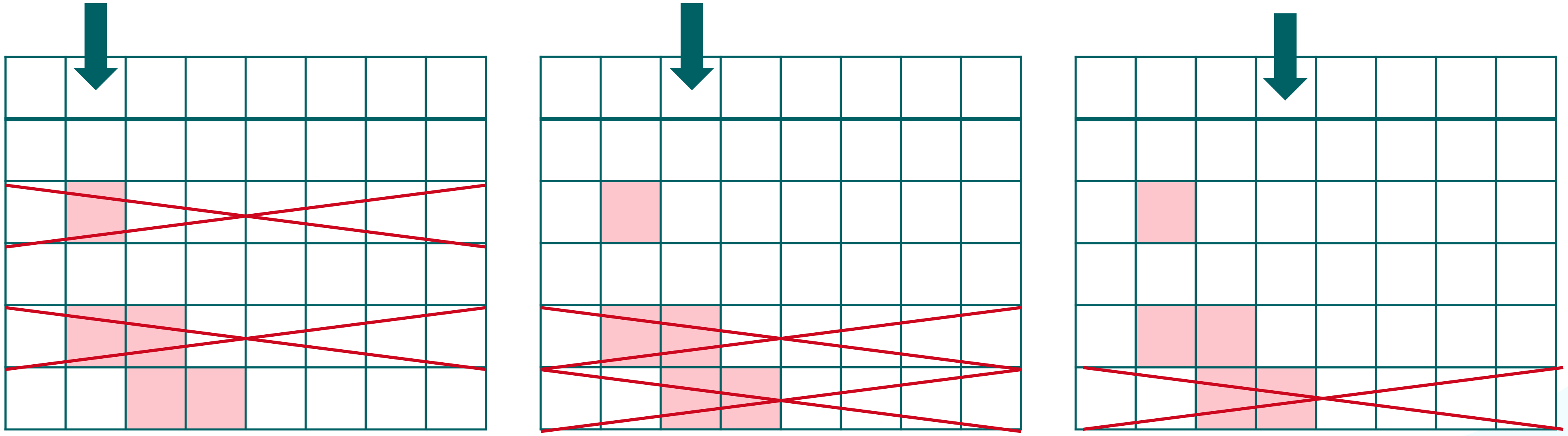
- The entire instance is simply discarded
- Usually done when the whole instance becomes unusable (e.g., labeling attribute for classification is missing)
- If the data set misses a lot of values, this technique may make the whole data set unusable or introduce a bias



Handling Missing Values: Ignore

Ignore the instance only for features where the value is missing

- The instance is ignored when analyzing features where it misses a value
- Information for other features remains usable



Handling Missing Values: Create

Mean/median/mode of the whole feature

- Compute mean/median/mode and fill the gaps accordingly
- Example: compute yearly income

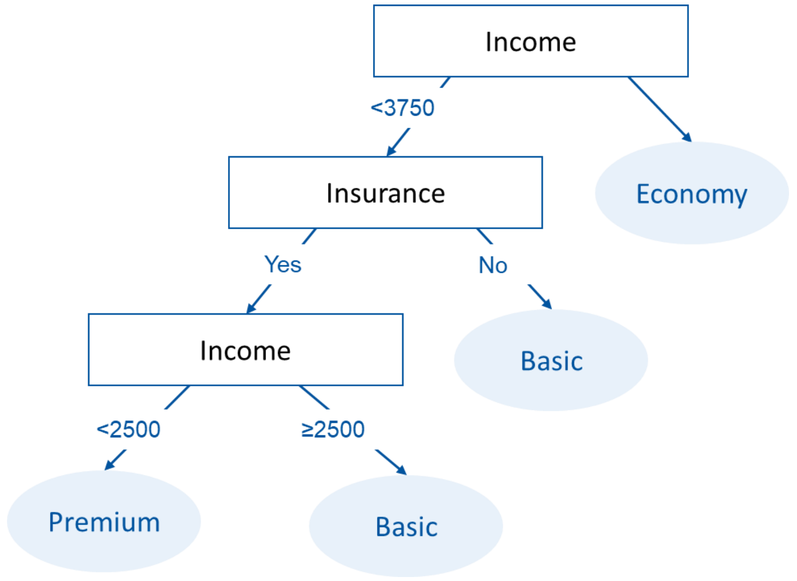
Mean/median/mode of all instances belonging to the same class

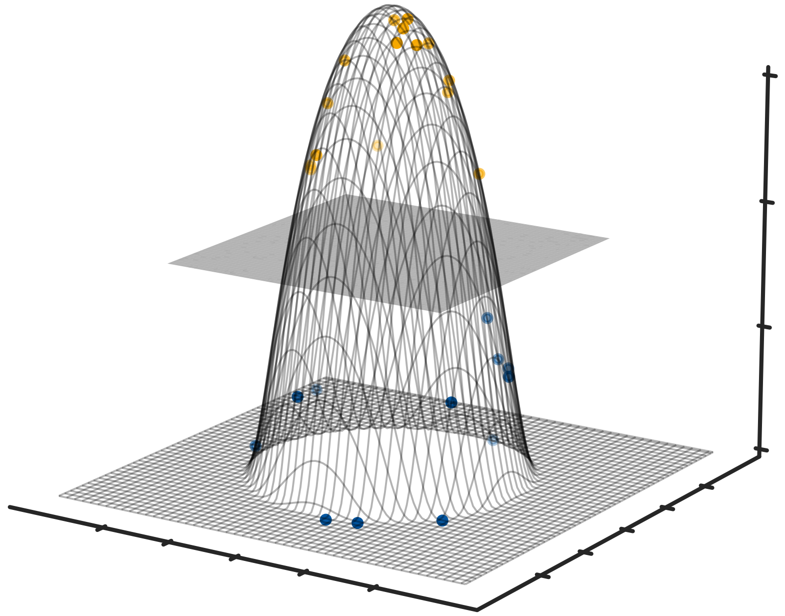
- Compute mean/median/mode only based on instances with the same class label
- Higher chances to be accurate compared to the overall mean/median/mode
- Example: compute income for a 20-year-old Student living in Aachen, Germany

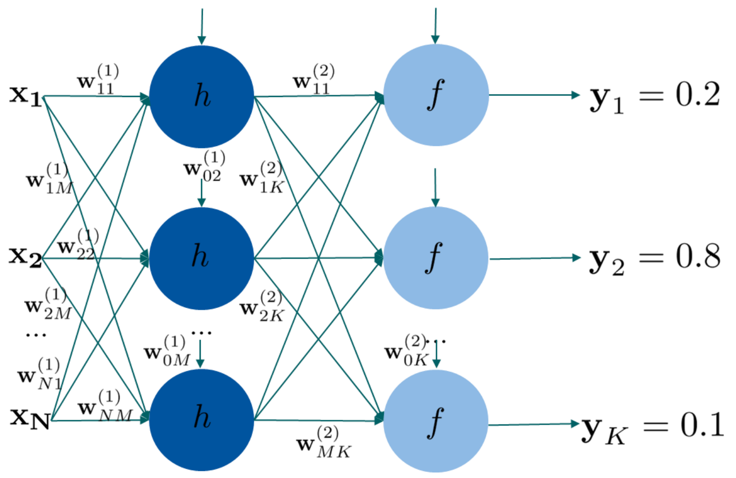
Handling Missing Values: Create

Complex derived value (use a predictor model)

- Fill in the value given by a suitable prediction model
- E.g., decision trees, regression, NNs, SVMs…

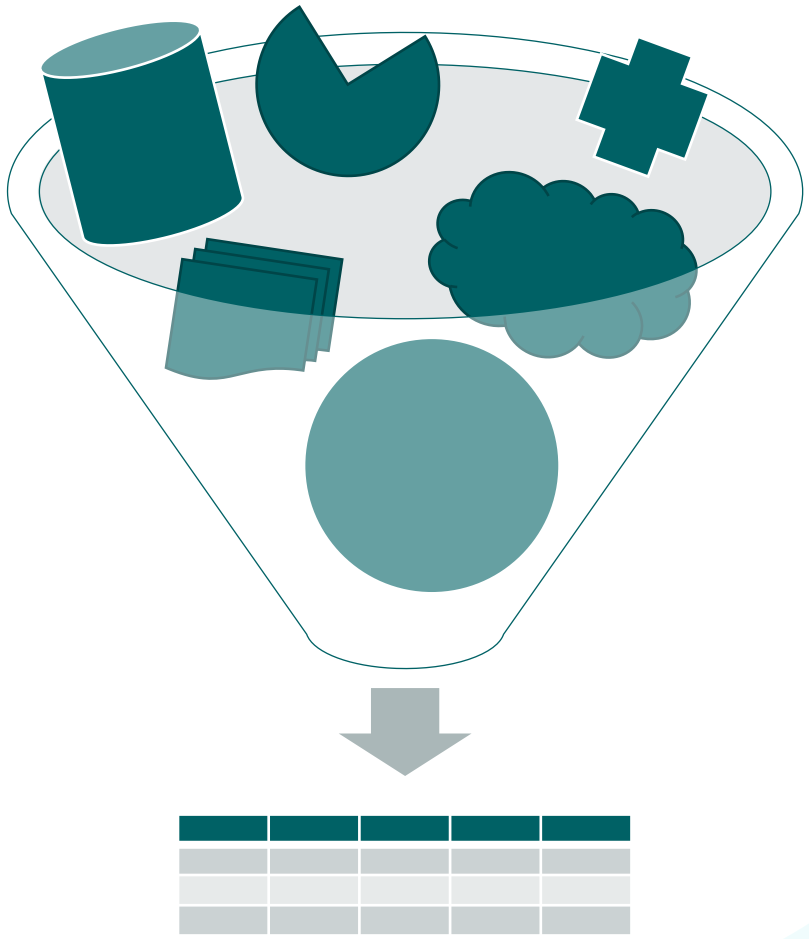




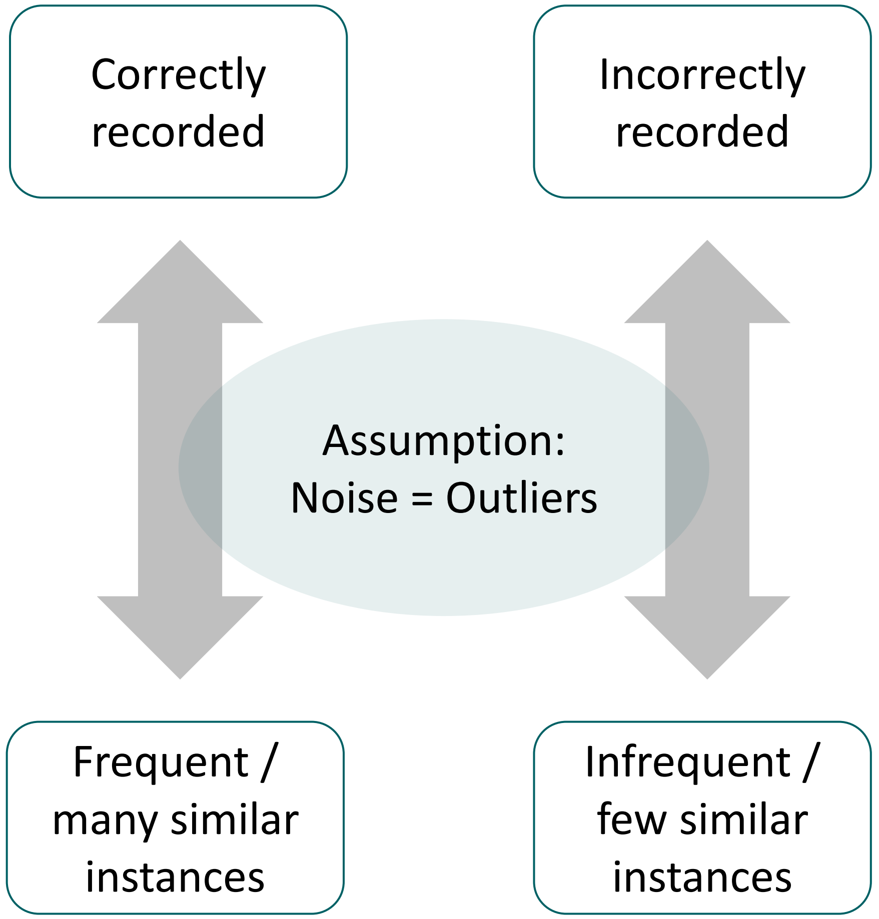


Data Quality & Preprocessing

1. Introduction
2. Missing Values
3. **Outliers**
4. Transformation & Normalization
5. Reduction



Introduction

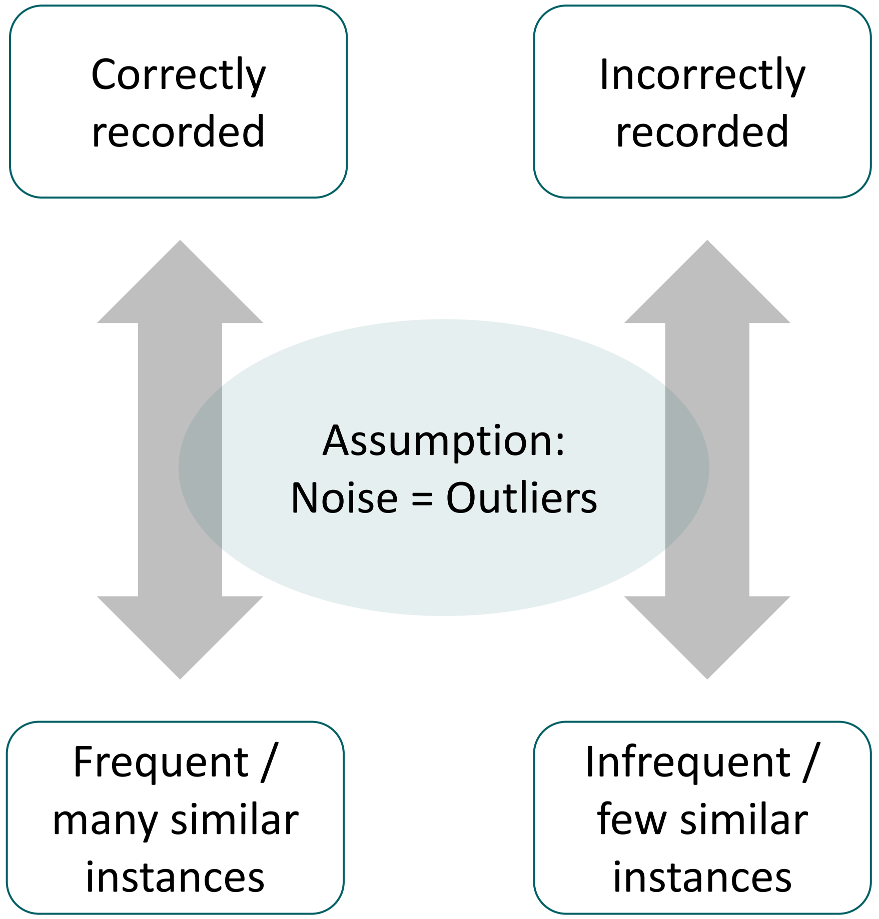

Correctly recorded
Incorrectly recorded
Assumption:
Noise = Outliers
Frequent / many similar instances
Infrequent / few similar instances

e.g., 25.5 centimeters of snow in Rome

What is noise?

- We assume that noise causes outliers
- Thus, outliers indicate noise

Outlier Detection

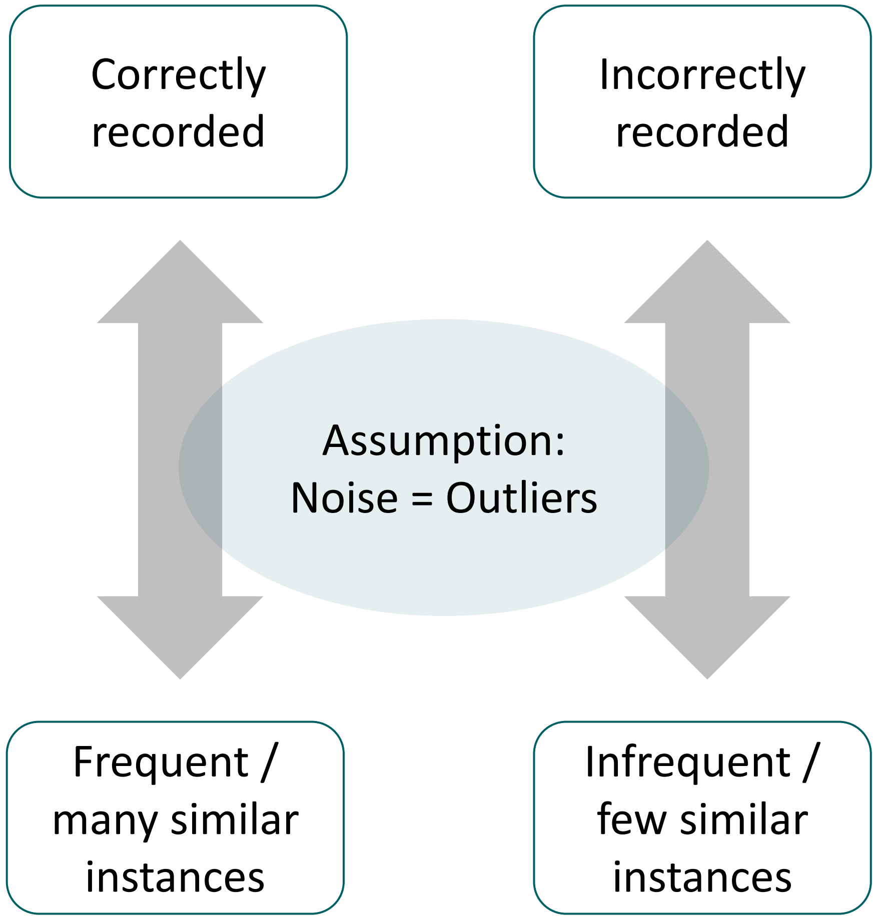

Correctly recorded
Incorrectly recorded
Assumption: Noise = Outliers
Frequent / many similar instances
Infrequent / few similar instances

How to detect outliers?

- Boxplots
- Decision trees
- Regression
- SVMs
- Clustering
- …

→ Predictor models can be used to **define** outliers

Outlier Handling


Correctly recorded
Incorrectly recorded
Assumption: Noise = Outliers
Frequent / many similar instances
Infrequent / few similar instances

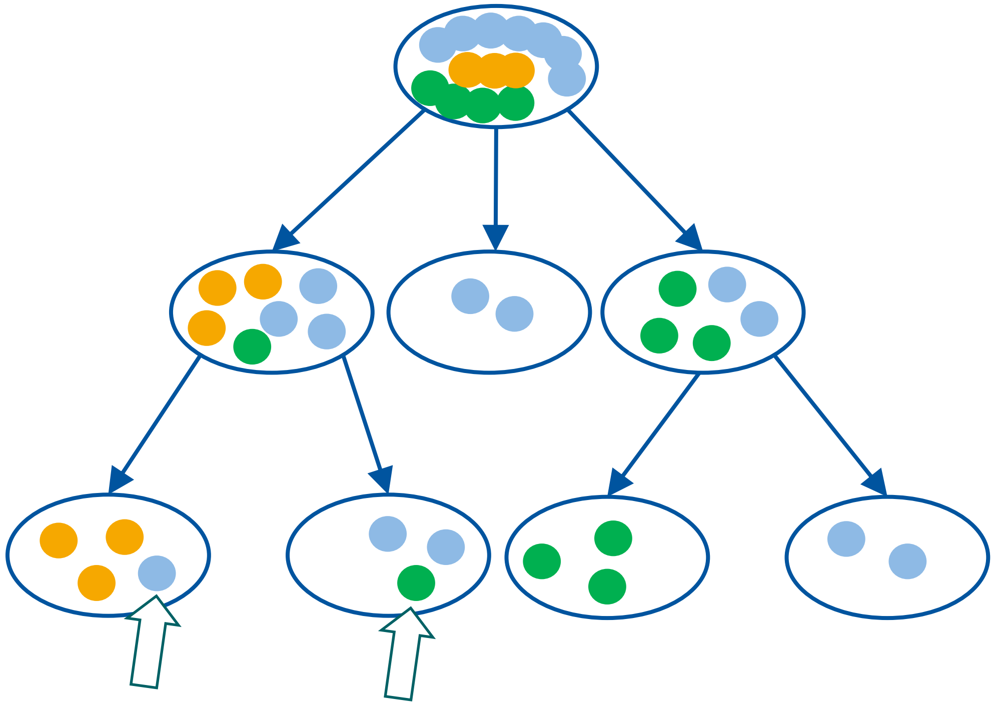
How to handle outliers?

Outliers can be handled as missing values:

- Fill in a correct value manually
- Ignore the feature/instance
- Replace with a derived value

→ Predictor models can be used to **replace** outliers

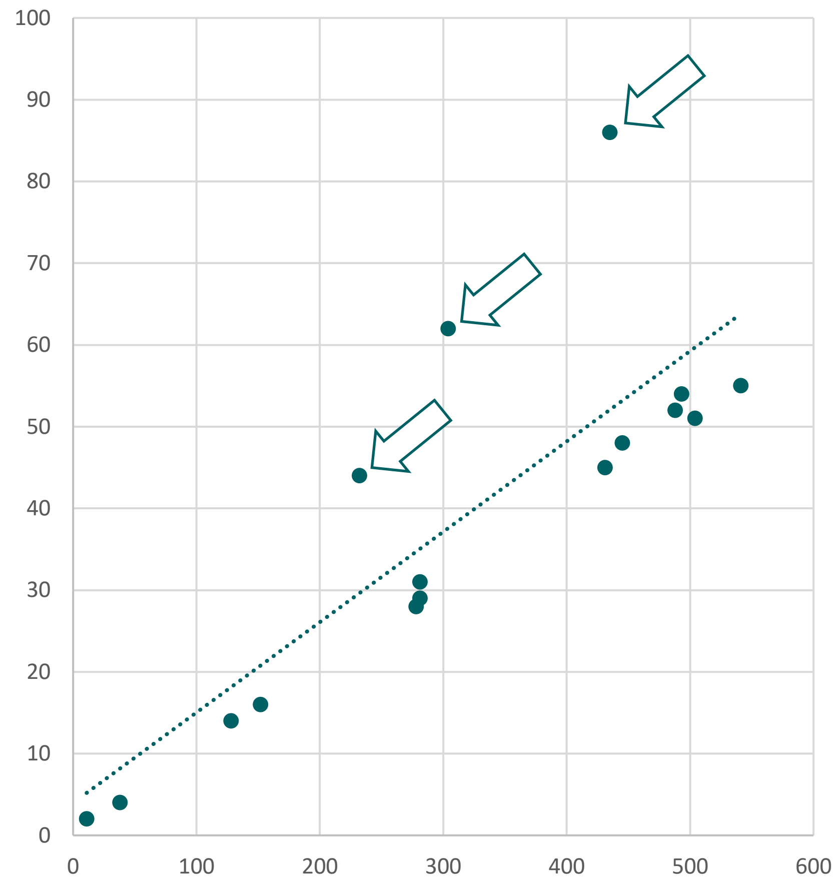
Outlier Detection - Decision Trees



How to detect outliers?

- Every leaf node is assigned a class label
- Instances in that leaf node with a non-matching class label can be considered outliers

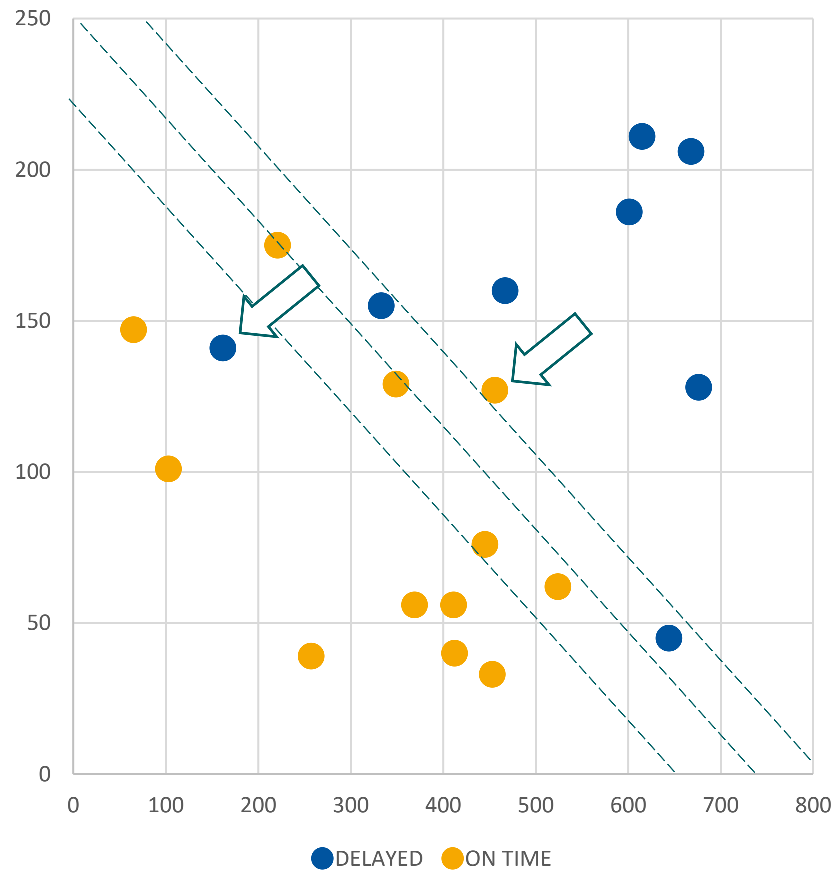
Outlier Detection - Regression



How to detect outliers?

- Instances which are far away from the predicted value are considered outliers
- The definition of ‘far away’ depends on an error function and threshold

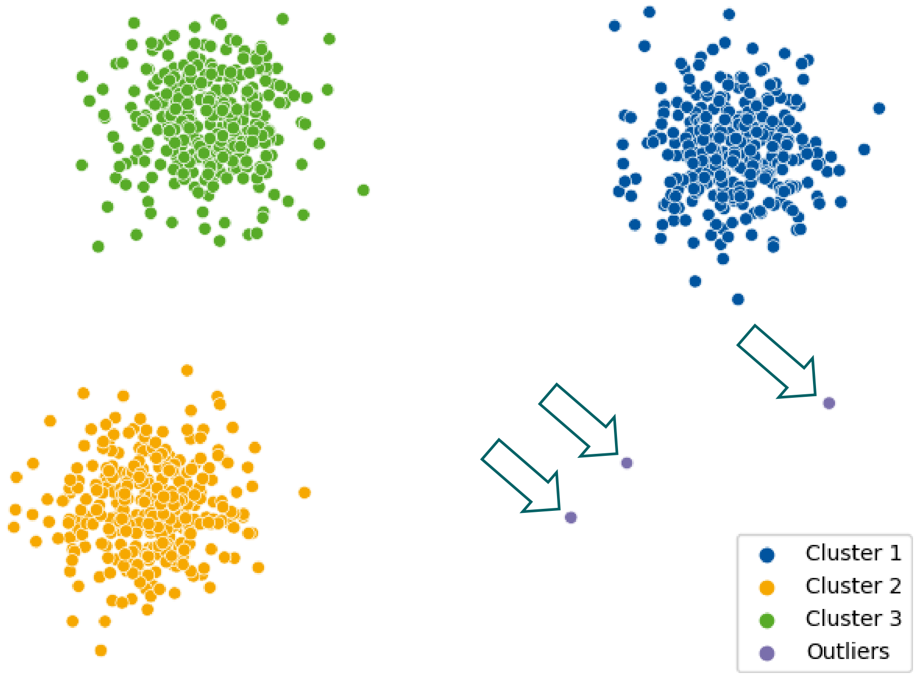
Outlier Detection – SVM



How to detect outliers?

- Instances which are (too far) on the wrong side of the hyperplane are considered outliers
- Soft margin may be used to define how far

Outlier Detection - Clustering

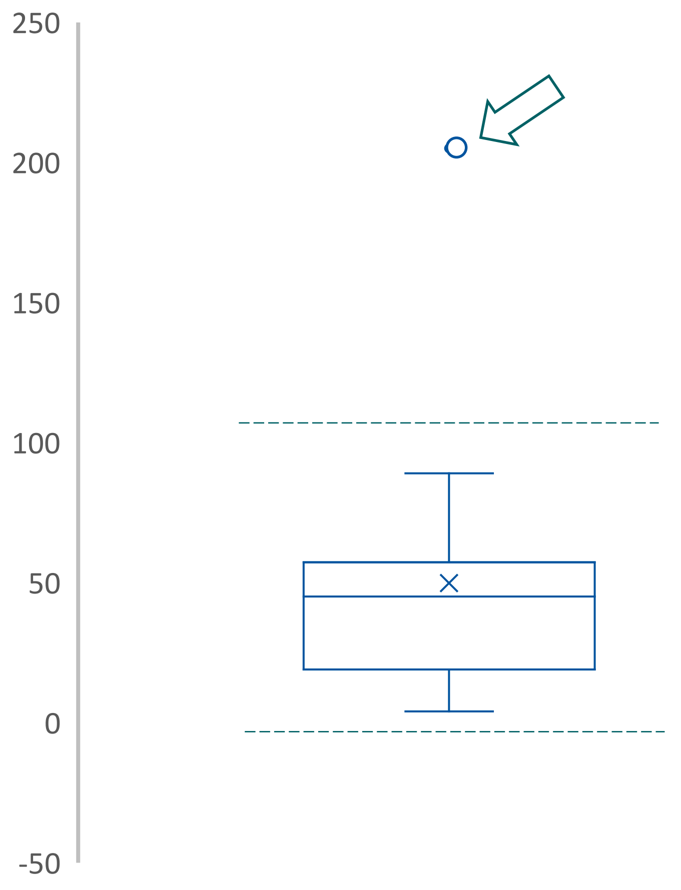


How to detect outliers?

- Instances outside of any cluster can be considered outliers

(Of course, a prerequisite for this is that the clustering algorithm itself can handle outliers…)

Outlier Detection - Boxplots



How to detect outliers?

- Instances above the upper fence
- Instances below the lower fence

→ Outlier handling option:
Clamp values to the nearest fence or take median value

Outlier Handling

[1]

How to handle outliers?

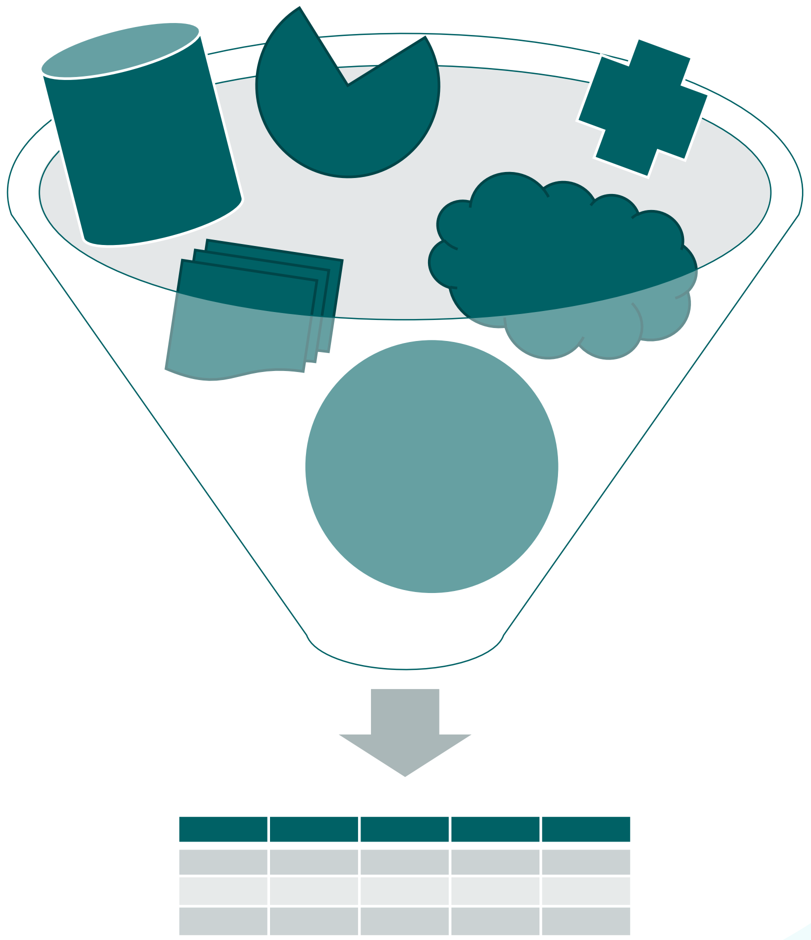
Outliers can be handled as missing values:

1) Fill in a correct value manually
2) Ignore the feature/instance
3) Replace with a derived value

Again: the appropriate method depends on the data and purpose

Data Quality & Preprocessing

1. Introduction
2. Missing Values
3. Outliers
4. **Transformation & Normalization**
5. Reduction



Preprocessing – Preparing the Data for Analysis

- Transformation: change the data to the right data type
- Normalization: adjust the influence of features
- Reduction: make the data smaller for analysis

[1]

Preprocessing – Transformation

- One-hot encoding: categorical to numerical
- Binning: numerical to categorical

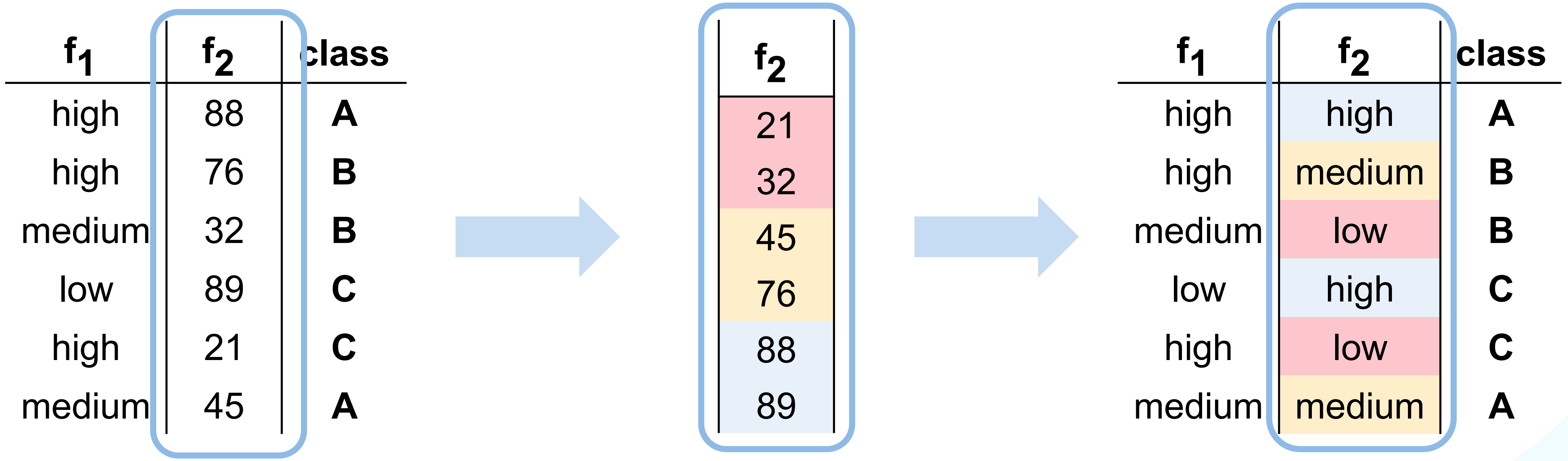
f_1	f_2	class
high	88	**A**
high	76	**B**
medium	32	**B**
low	89	**C**
high	21	**C**
medium	45	**A**

$\Rightarrow$

f_1 - high	f_1 - medium	f_1 - low	f_2	class
1	0	0	88	**A**
1	0	0	76	**B**
0	1	0	32	**B**
0	0	1	89	**C**
1	0	0	21	**C**
0	1	0	45	**A**

Preprocessing – Transformation

- One-hot encoding: categorical to numerical
- Binning: numerical to categorical



f_1	f_2	class
high	88	A
high	76	B
medium	32	B
low	89	C
high	21	C
medium	45	A

f_2
21
32
45
76
88
89

f_1	f_2	class
high	high	A
high	medium	B
medium	low	B
low	high	C
high	low	C
medium	medium	A

Preprocessing – Normalization

Adjusting the influence of features

- Feature weight and range often depends on the chosen unit (km, mm, miles, …)
- Algorithms tend to give more weight to features with a large range

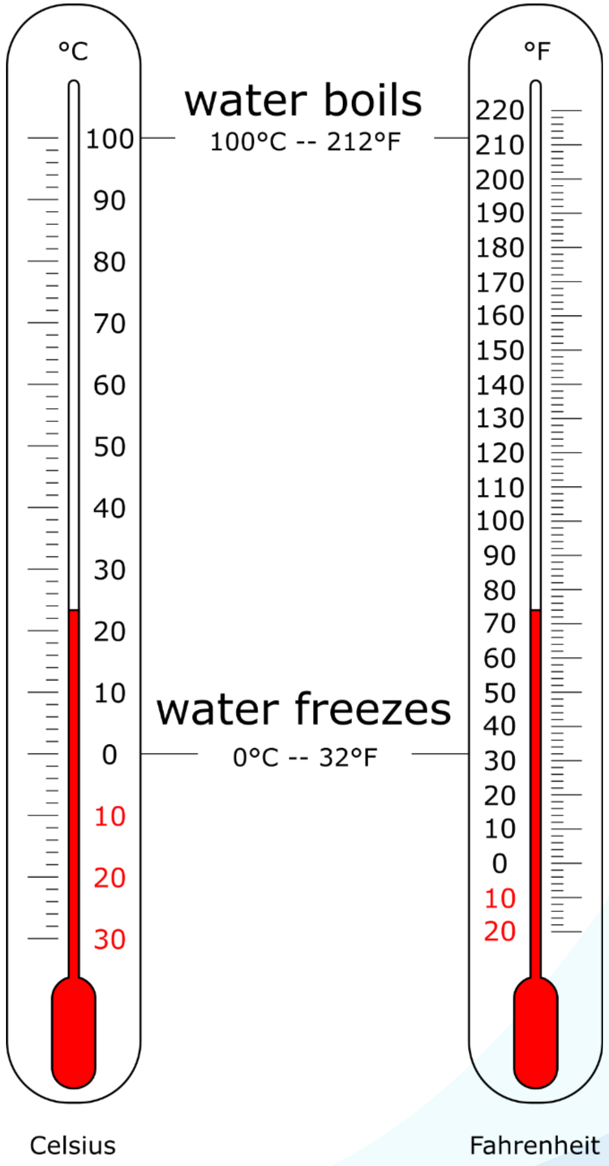
→ May introduce an unwanted bias

→ May hinder interpretability

- Scales may be non-linear
 (e.g. logarithmic)

Sum of squared errors:

$$\frac{1}{2}\sum_{i=1}^{N}(t_i - \mathbb{M}(\mathbf{x_i}))^2$$



Preprocessing – Normalization

Min-max normalization

- Maps the values onto a predefined range [low, high]
- Preserves relative differences, i.e., relations between the data values

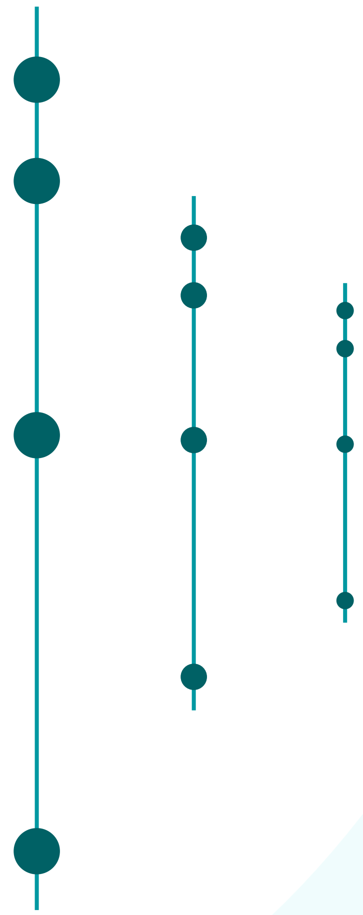
We normalize feature *d* by replacing its value for each instance *i* as follows:

value of feature *d* in the *i*th instance

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - d_{\min}}{d_{\max} - d_{\min}} \cdot (\text{high} - \text{low}) + \text{low}$$

maximal value of feature *d*

minimal value of feature *d*



Preprocessing – Normalization

Min-max normalization

d
11
82
33
12
76

$d_{min} = 11$
$d_{max} = 82$

Consider
$\text{high} = 100$
$\text{low} = 5$

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - d_{\min}}{d_{\max} - d_{\min}} \cdot (\text{high} - \text{low}) + \text{low}$$

Preprocessing – Normalization

Min-max normalization

d
11
82
33
12
76

$d_{min} = 11$
$d_{max} = 82$

Consider
$\text{high} = 100$
$\text{low} = 5$

norm(d)
$(11 - 11)/(82 - 11) \cdot (100 - 5) + 5$
$(82 - 11)/(82 - 11) \cdot (100 - 5) + 5$
$(33 - 11)/(82 - 11) \cdot (100 - 5) + 5$
$(12 - 11)/(82 - 11) \cdot (100 - 5) + 5$
$(76 - 11)/(82 - 11) \cdot (100 - 5) + 5$

$\approx$

norm(d)
5
100
34.44
6.34
91.97

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - d_{\text{min}}}{d_{\text{max}} - d_{\text{min}}} \cdot (\text{high} - \text{low}) + \text{low}$$

Preprocessing – Normalization

Standard score (Z-score) normalization

- Uses the standard deviation to quantify the significance of the difference between a value and the overall mean
- Range is $[-\infty, \infty]$, but 0 has a clear meaning
- Useful when actual minimum and maximum of the attribute are unknown
- Useful when outliers may impact min-max normalization

For each *i*:

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - \bar{d}}{\text{sd}(d)}$$

$\bar{d}$ is the mean of all values of feature d: $\frac{1}{N}\sum_{i=1}^{N} \mathbf{x_i}[d]$

$\text{sd}(d)$ is the standard deviation of feature d: $\sqrt{\left(\frac{\sum_{i=1}^{N}(\mathbf{x_i}[d]-\bar{d})^2}{N-1}\right)}$

Preprocessing – Normalization

Standard score (Z-score) normalization

d
11
82
33
12
76

$\bar{d} = 42.8$

$\text{sd}(d) = 34.259$

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - \bar{d}}{\text{sd}(d)}$$

Preprocessing – Normalization

Standard score (Z-score) normalization

d
11
82
33
12
76

$\bar{d} = 42.8$
$\text{sd}(d) = 34.259$

norm(d)
$(11 - 42.8)/34.259$
$(82 - 42.8)/34.259$
$(33 - 42.8)/34.259$
$(12 - 42.8)/34.259$
$(76 - 42.8)/34.259$

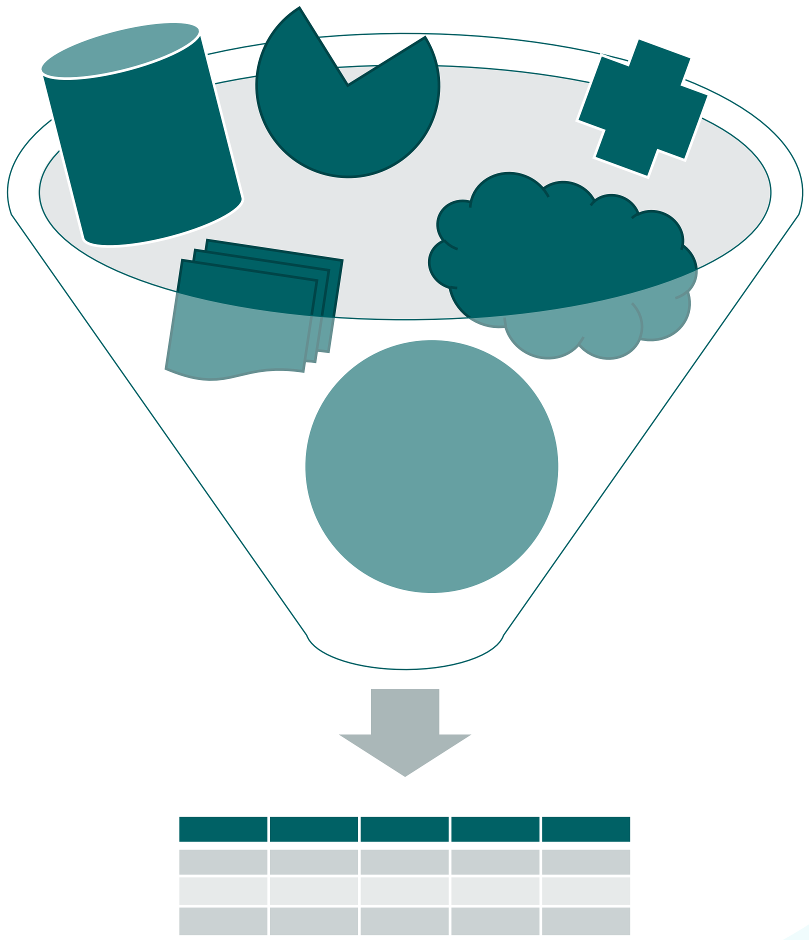
$\approx$

norm(d)
-0.93
1.14
-0.29
-0.90
0.97

$$\text{norm}(\mathbf{x_i}[d]) = \frac{\mathbf{x_i}[d] - \bar{d}}{\text{sd}(d)}$$

Data Quality & Preprocessing

1. Introduction
2. Missing Values
3. Outliers
4. Transformation & Normalization
5. **Reduction**



Preprocessing – Data Reduction

- Analysis may become unfeasible due to size of data
- Goal: reduce the data size but maintain same (or similar) analysis results

- Feature reduction: remove or replace some features
- Instance reduction: remove, replace or aggregate some instances

Feature reduction

Instance reduction

ID	f_1	f_2	…	f_D
1				
2				
…				
N				

[2]

Preprocessing – Feature Reduction

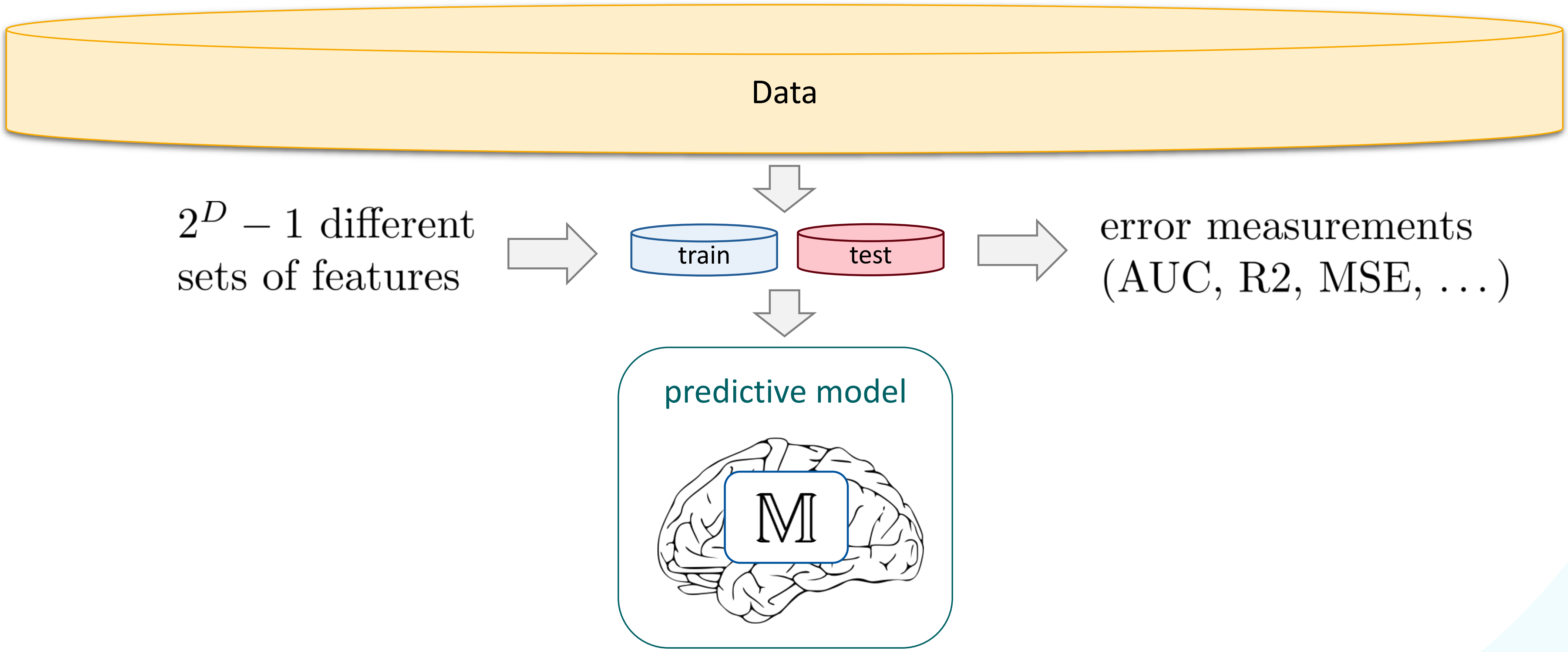
Projecting data on fewer dimensions

- Autoencoders (compare text mining): a special type of NN that transforms the input data into a representation with fewer dimensions (encoding). Learning a good representation is key!
- Principal Component Analysis (PCA): represent the original features by a few orthogonal (uncorrelated) variables that capture most of the variability

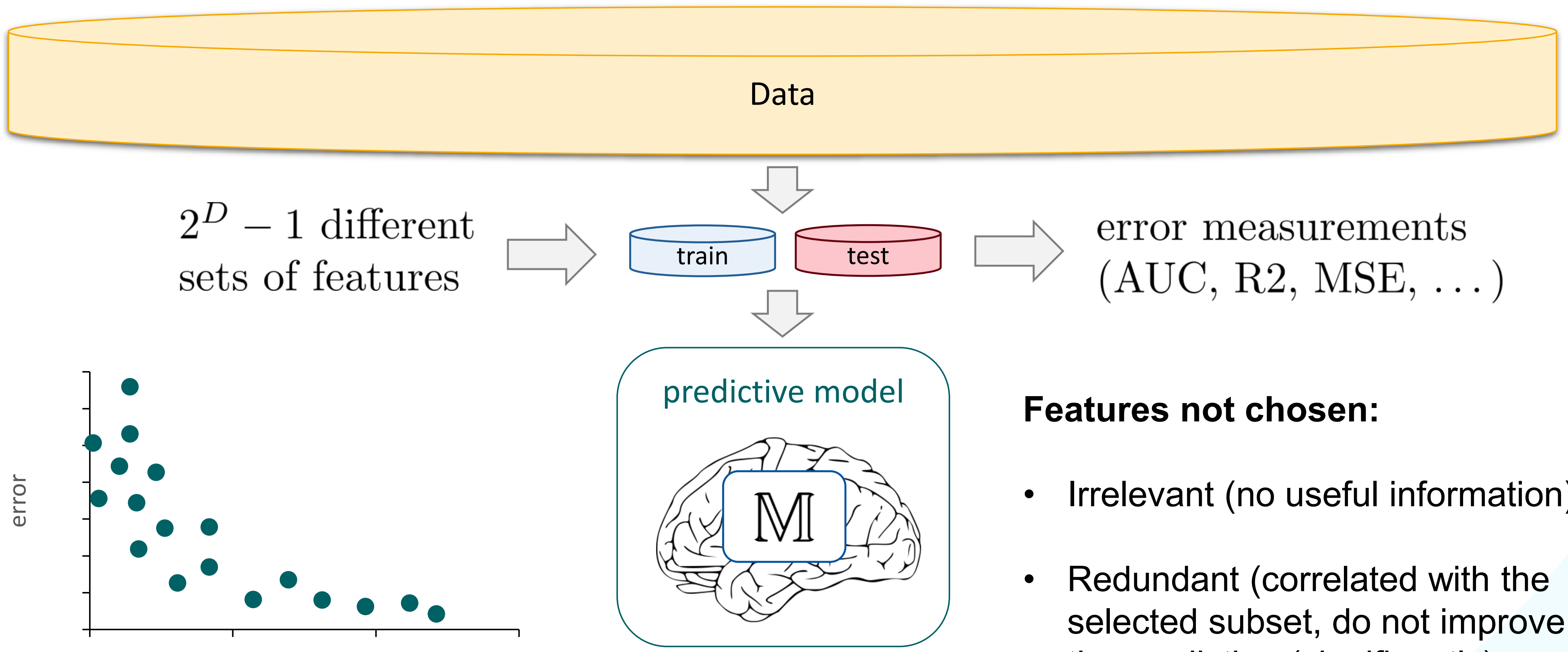
Feature subset selection: detect and remove irrelevant/redundant features

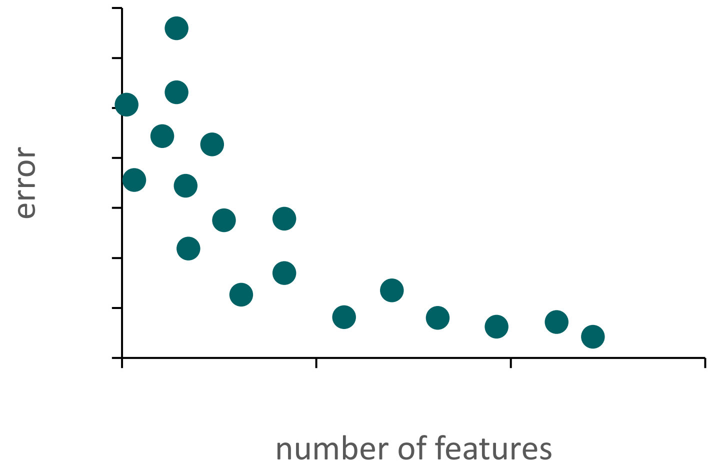
- Use domain knowledge (e.g., remove identifiers)
- Exploit dependencies (e.g., delete features that can be estimated from others using regression)
- Model-driven (e.g. delete features that are not used in a constructed decision tree or, more general, features that can be left out without reducing the quality of the model much)

Preprocessing – Feature Subset Selection


Data
$2^D - 1$ different sets of features
train
test
error measurements (AUC, R2, MSE, . . .)
predictive model
M

Preprocessing – Feature Subset Selection


Data
$2^D - 1$ different sets of features
train
test
error measurements (AUC, R2, MSE, ...)
predictive model
M

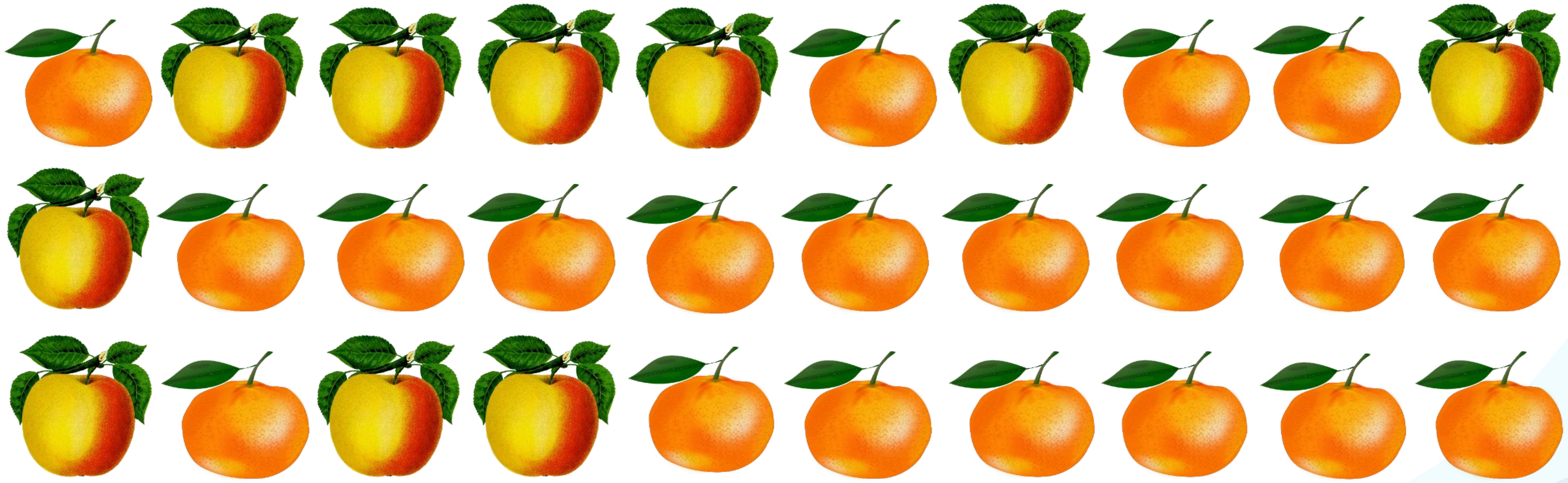


Features not chosen:

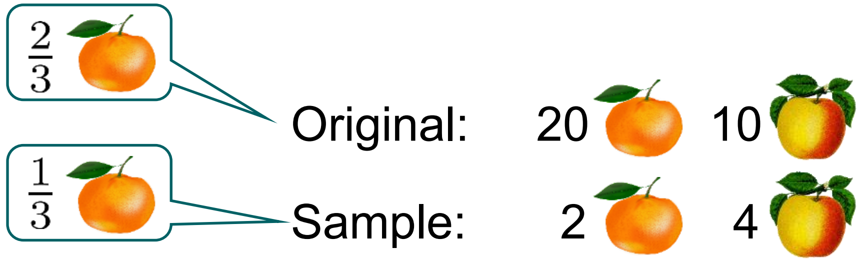
- Irrelevant (no useful information)
- Redundant (correlated with the selected subset, do not improve the prediction (significantly)

Preprocessing – Sampling

Goals: make the data smaller, remove or introduce biases

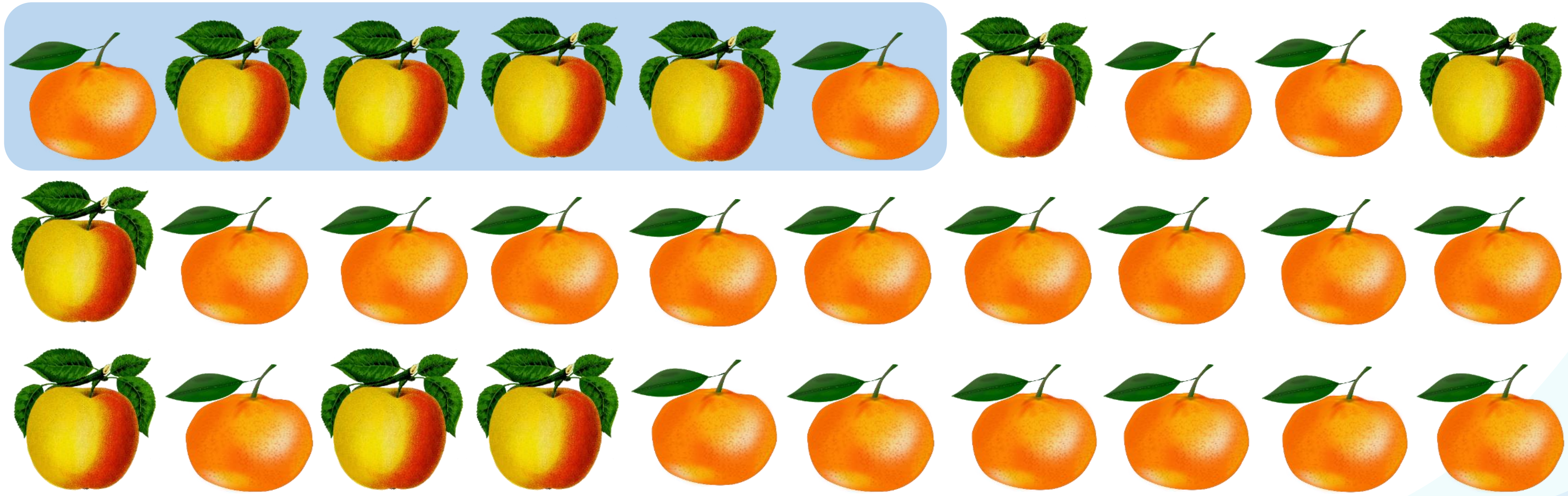


Preprocessing – Sampling

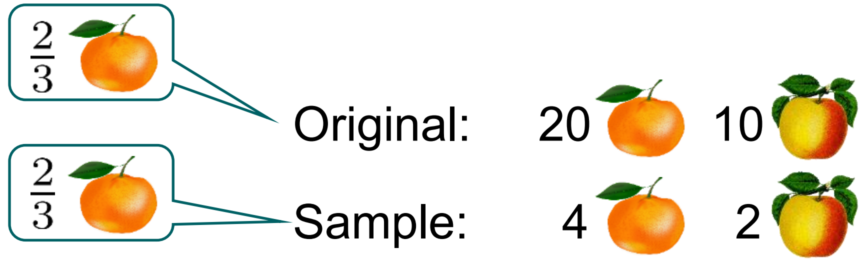


Top sampling:

take the first *N* instances

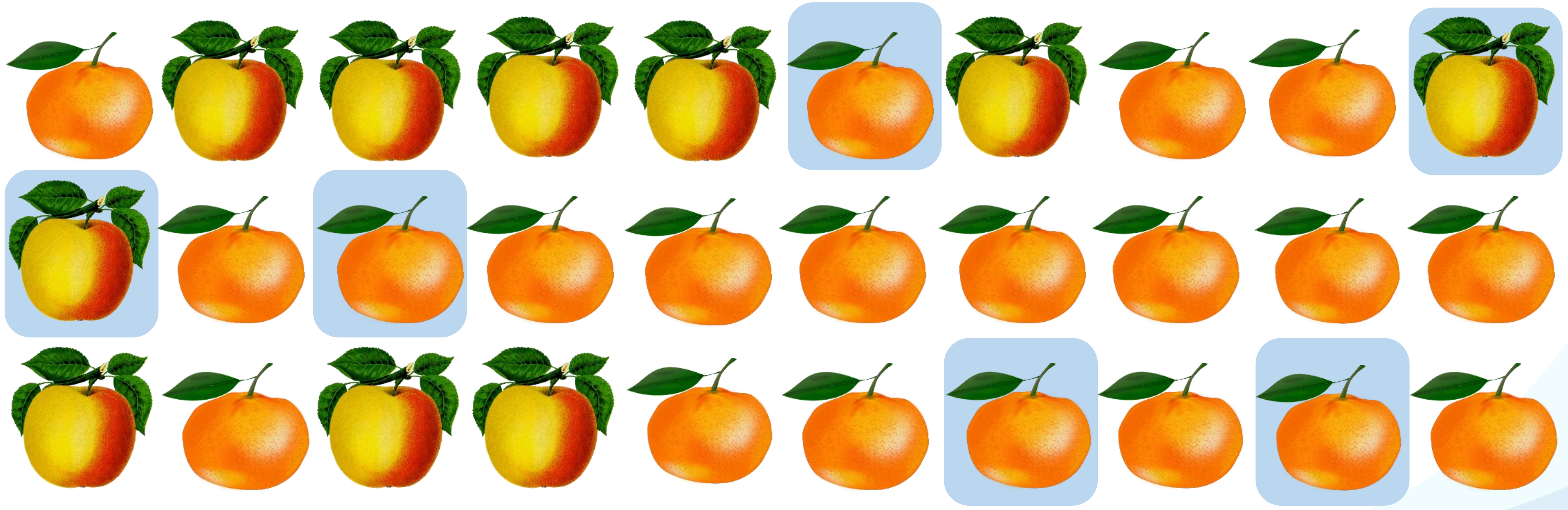


Preprocessing – Sampling

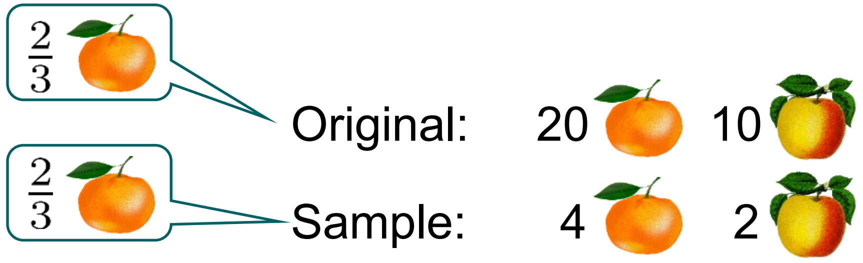


Random sampling:

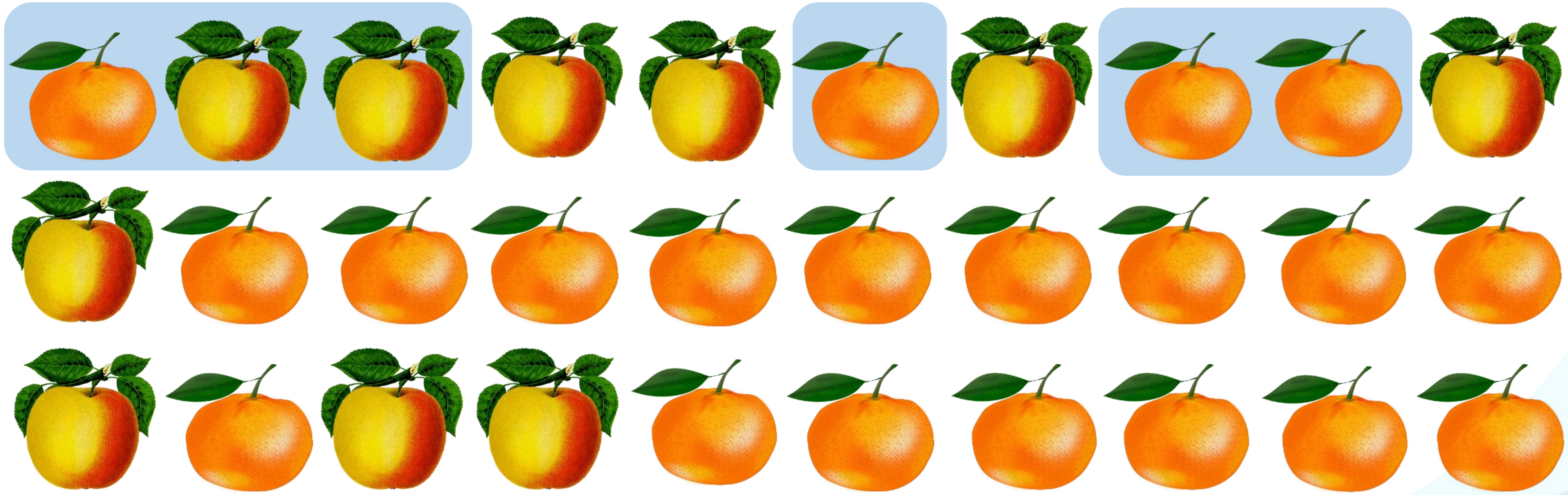
take *N* arbitrary instances (based on random generator)



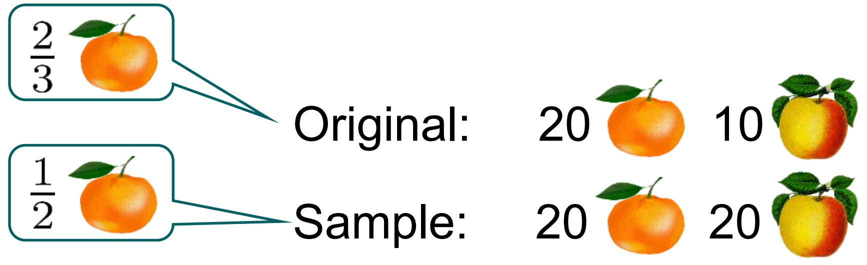
Preprocessing – Sampling



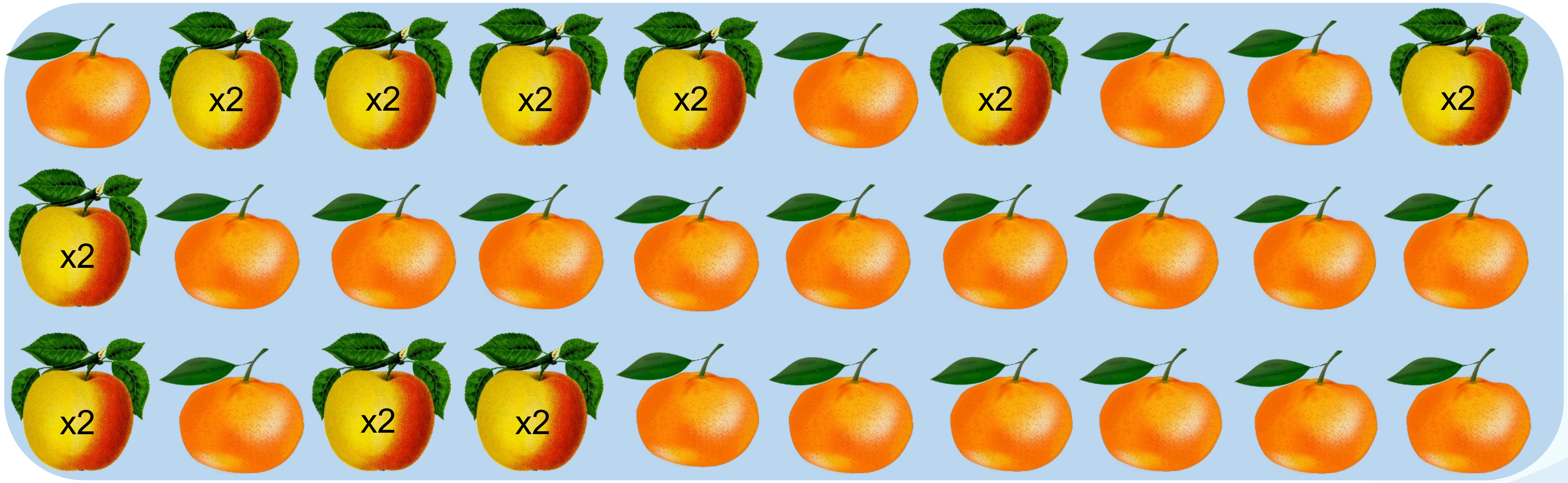
Stratified sampling: ensure that relative frequencies are maintained (e.g., take the same percentage from every group)



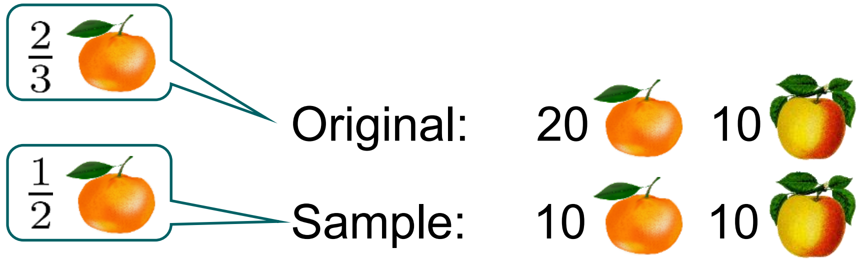
Preprocessing – Sampling



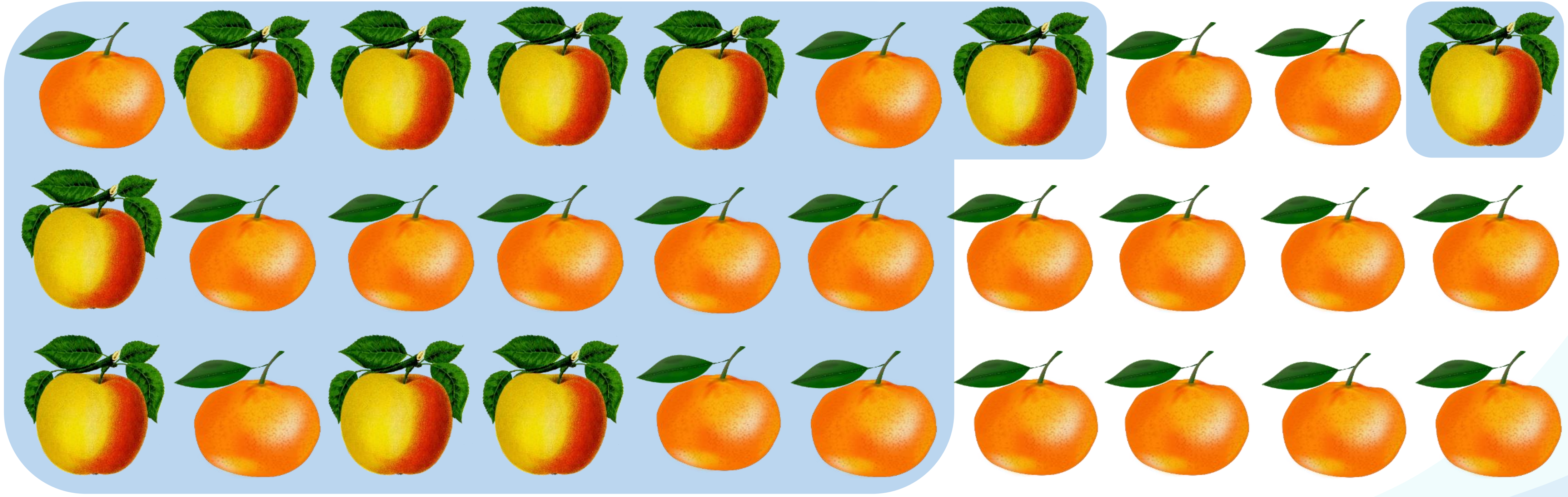
Over-sampling: ensure a certain distribution
(e.g. equal frequency for each group) by duplicating under-represented instances



Preprocessing – Sampling



Under-sampling: ensure a certain distribution
(e.g. equal frequency for each group) by leaving out over-represented instances



To Conclude

select → merge → filter → clean → transform → reduce → sample → …

Goal: increase data quality and modify the data to suit the analysis question and applied techniques

Best strategy/solution: depends on the data, context and goal of the analysis

Data quality aspects

- Missing data
- Noise/outliers
- Semantic problems

Data preprocessing

- Transformation
- Normalization
- Data reduction

Garbage in, Garbage out (GIGO)

80/20