Lecture 1: Introduction & Data

KI-Workshop
(HFT Stuttgart, 8-9 Nov 2023)

Michael Mommert
University of St. Gallen (soon-to-be HFT Stuttgart)

Today's lecture

What this course is about...

Who am I?

Course modalities

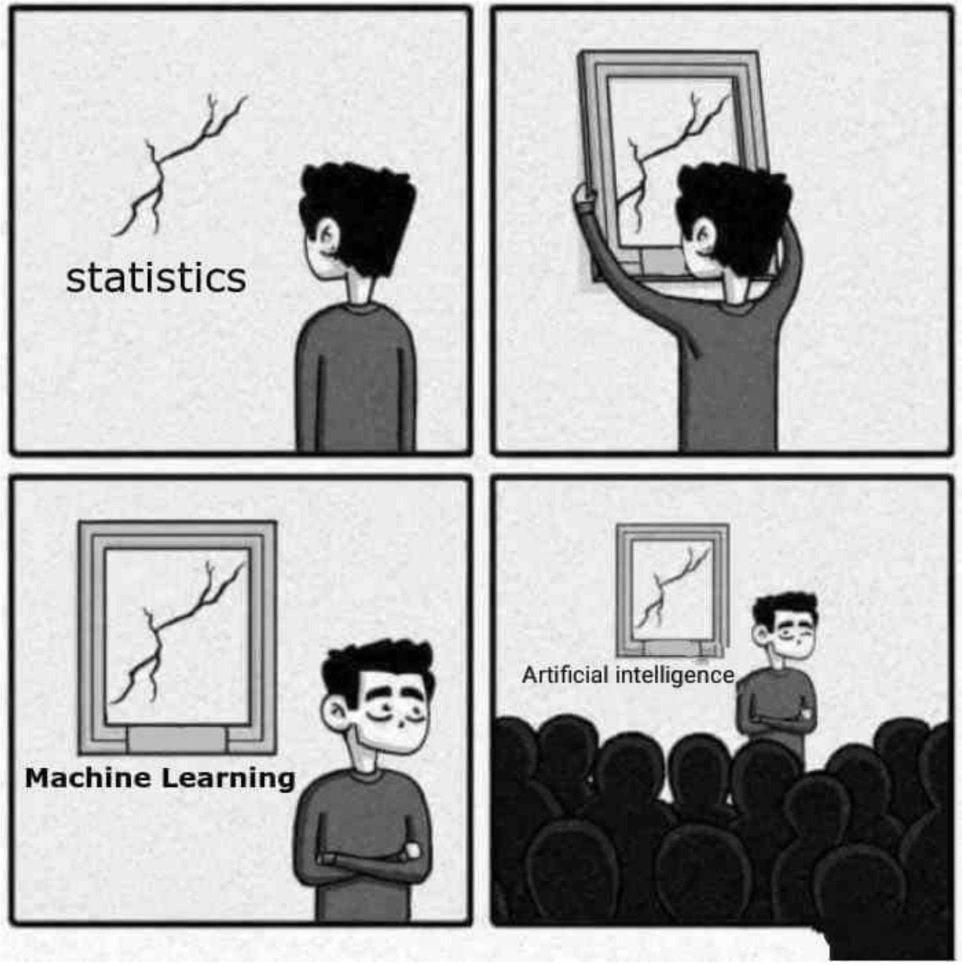
Course syllabus

Types of data

Features and feature engineering

Data scaling

What this course is about...



sandserif

Mapping terminology

Mapping terminology

Artificial Intelligence

Systems that perceive their environment and take actions to maximize their chances of achieving their goals.

Mapping terminology

Statistics

Data science

Mathematics

Artificial Intelligence

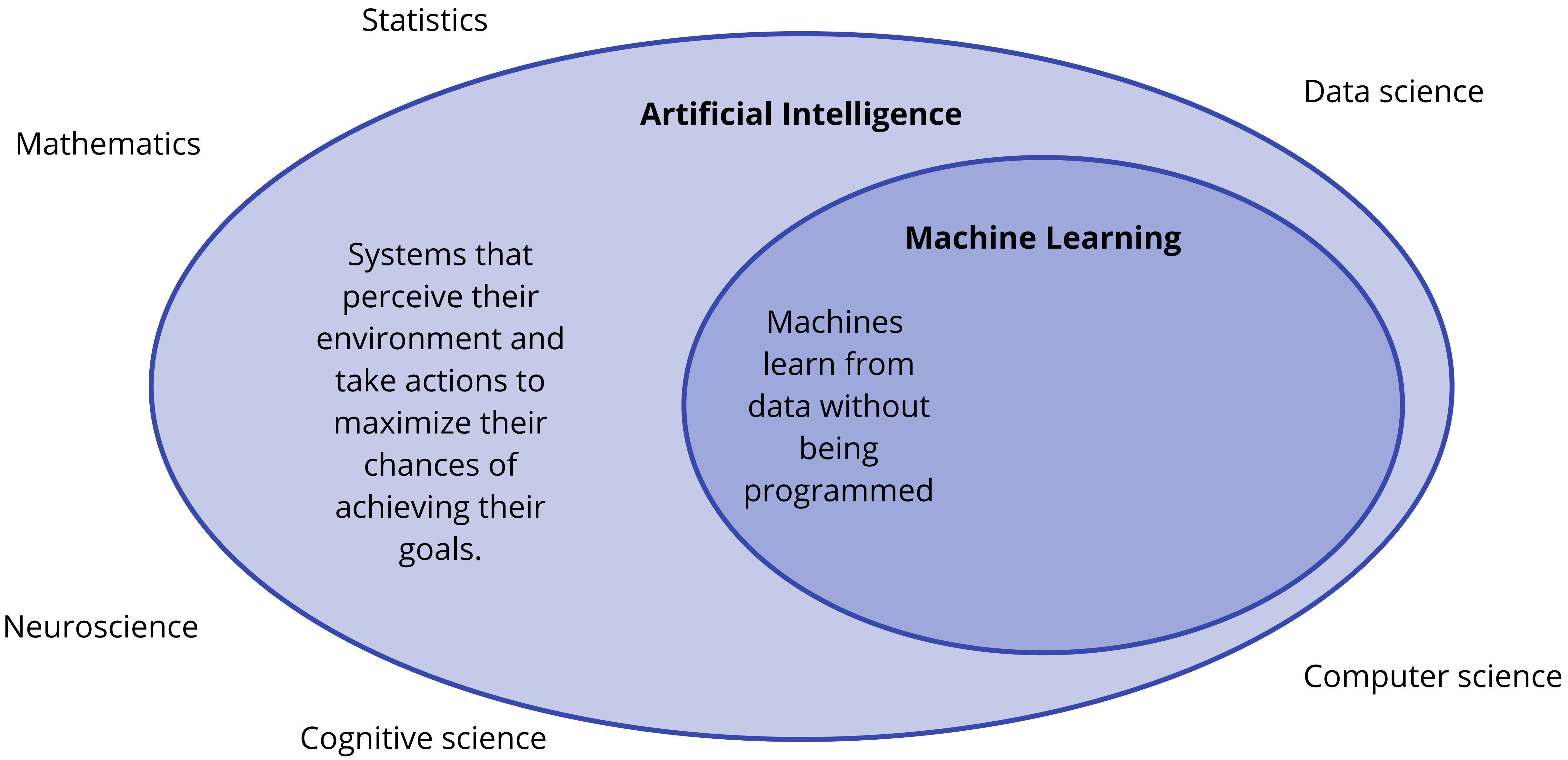
Systems that perceive their environment and take actions to maximize their chances of achieving their goals.

Neuroscience

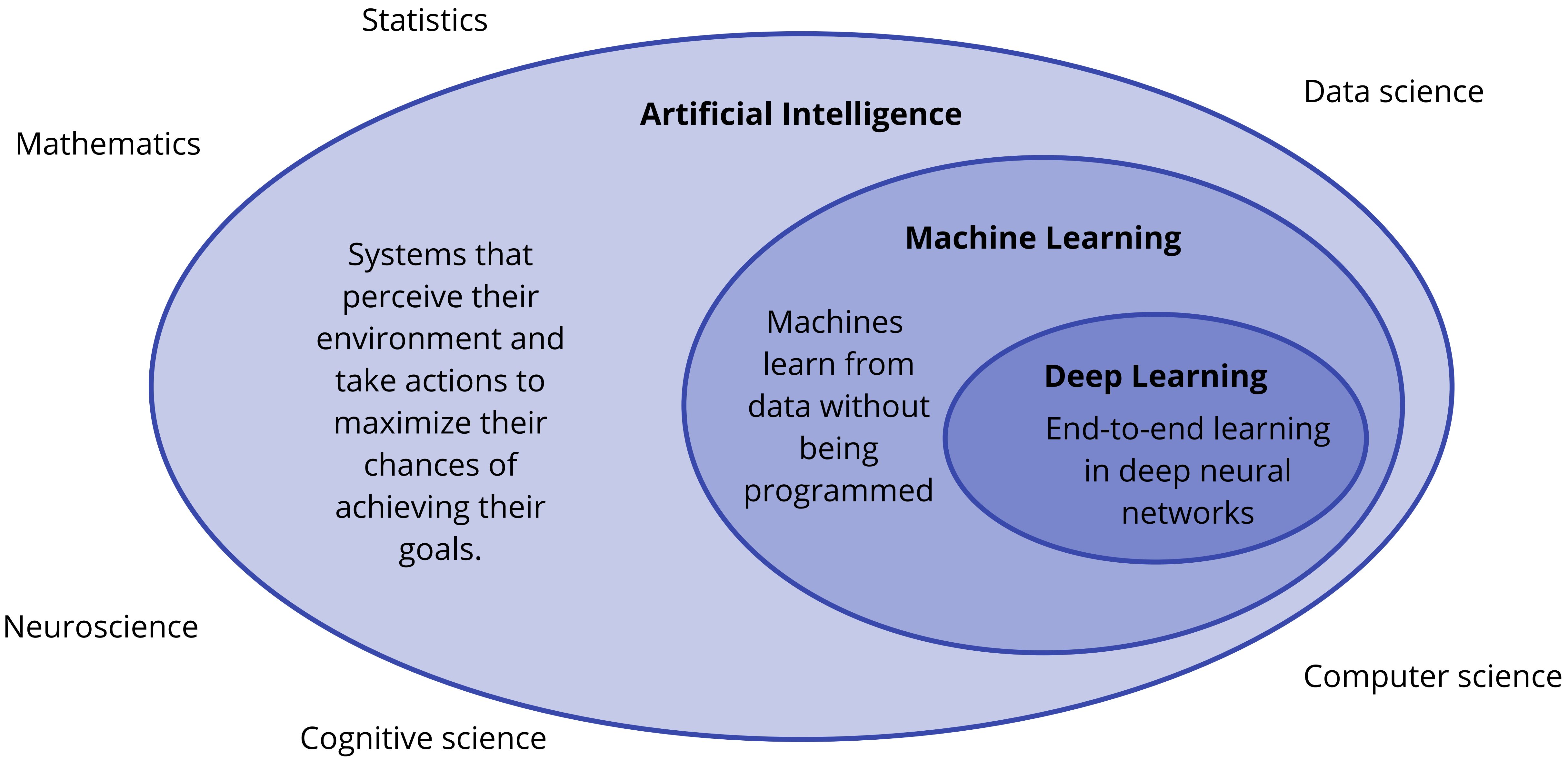
Computer science

Cognitive science

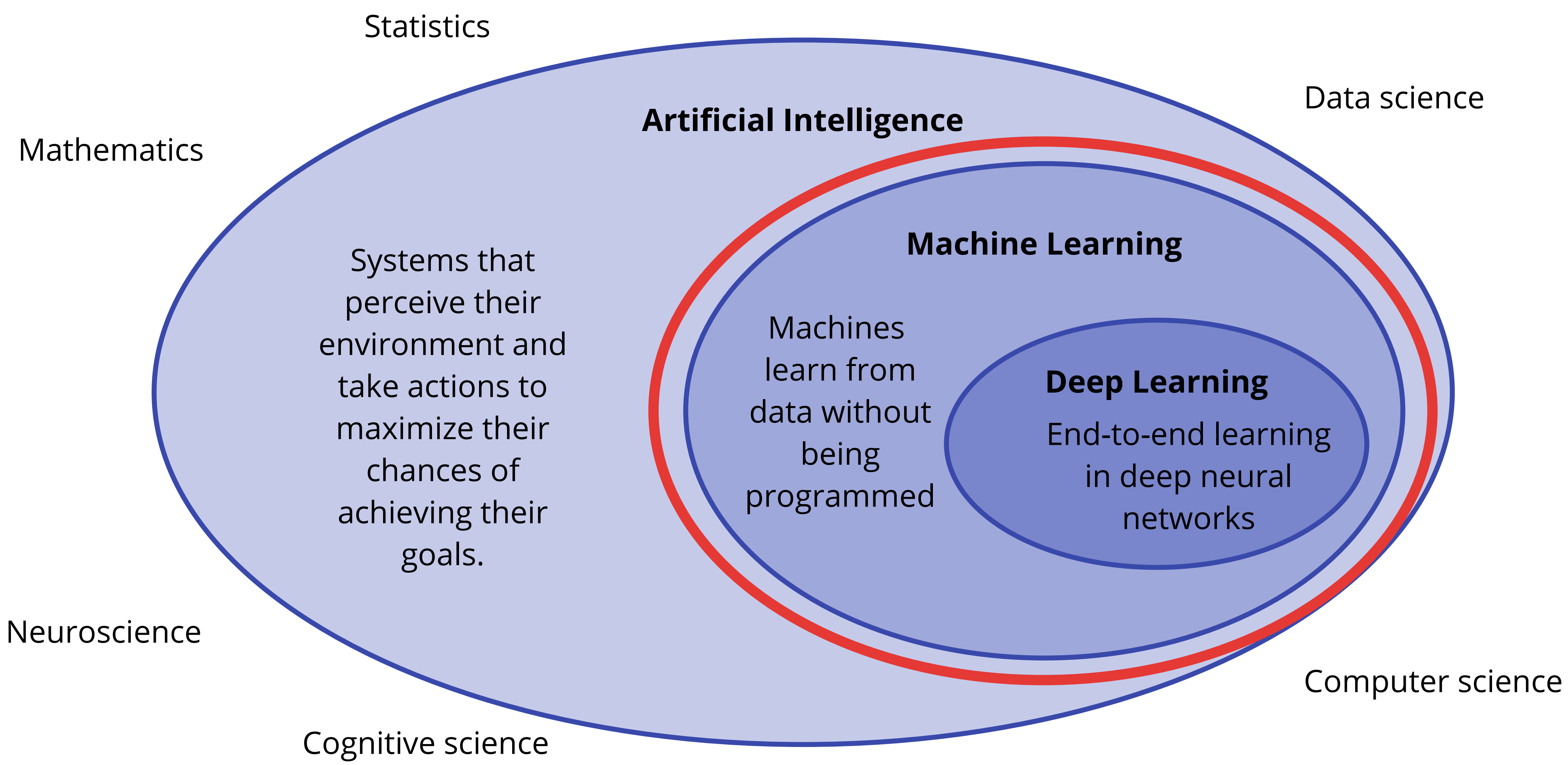
Mapping terminology



Mapping terminology



Mapping terminology



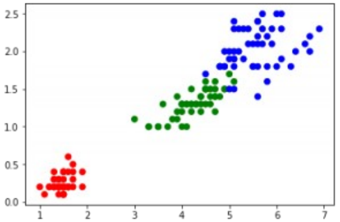
What is Machine Learning (ML)?

> *"The field of study that gives computers the ability to learn without being explicitly programmed."*
> - Arthur Samuel (1959)

Different approaches:

Iris Versicolor

- **Supervised learning**
  Find a function that relates input data to output data by learning a specific task.



- **Unsupervised learning**
  Find structure within a data set.

- **Reinforcement learning**
  Learn a task in a dynamic and responsive environment.

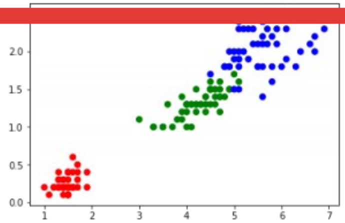
What is Machine Learning (ML)?

"The field of study that gives computers the ability to learn without being explicitly programmed."
- Arthur Samuel (1959)

Different approaches:

Iris Versicolor

- **Supervised learning**
  Find a function that relates input data to output data by learning a specific task.



- **Unsupervised learning**
  Find structure within a data set.

- **Reinforcement learning**
  Learn a task in a dynamic and responsive environment.

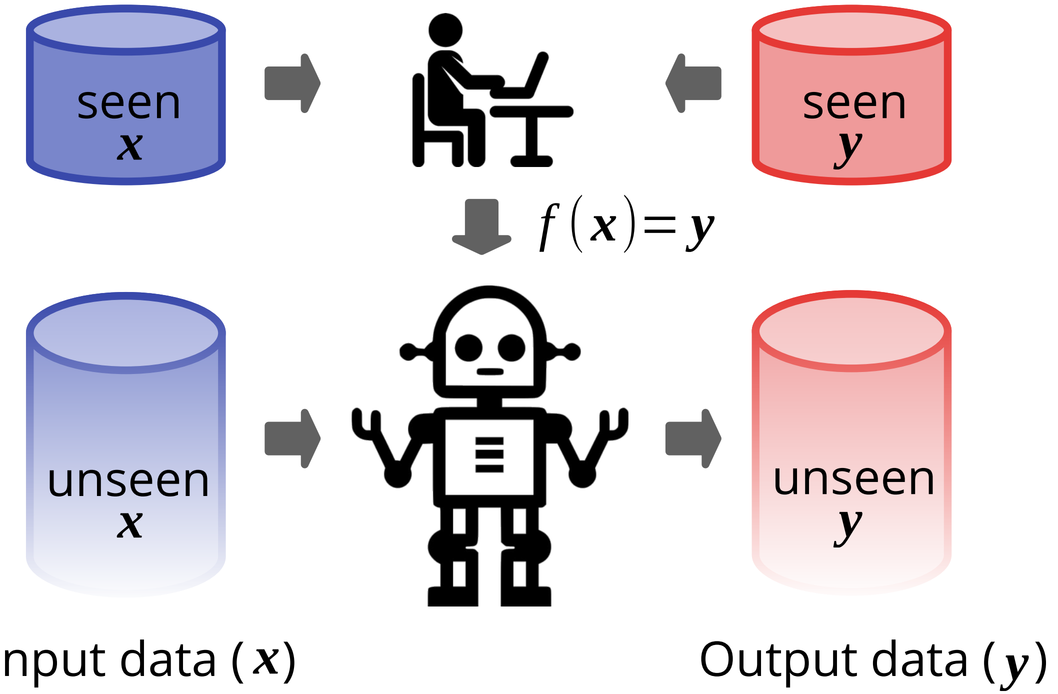
Supervised ML

Supervised ML

General goal for supervised problems:
Find a function (“task”) that relates input data ($\boldsymbol{x}$) to output data ($\boldsymbol{y}$) such that: $f(\boldsymbol{x}) = \boldsymbol{y}$

General goal for supervised problems:
Find a function (“task”) that relates input data ($\boldsymbol{x}$) to output data ($\boldsymbol{y}$) such that: $f(\boldsymbol{x}) = \boldsymbol{y}$

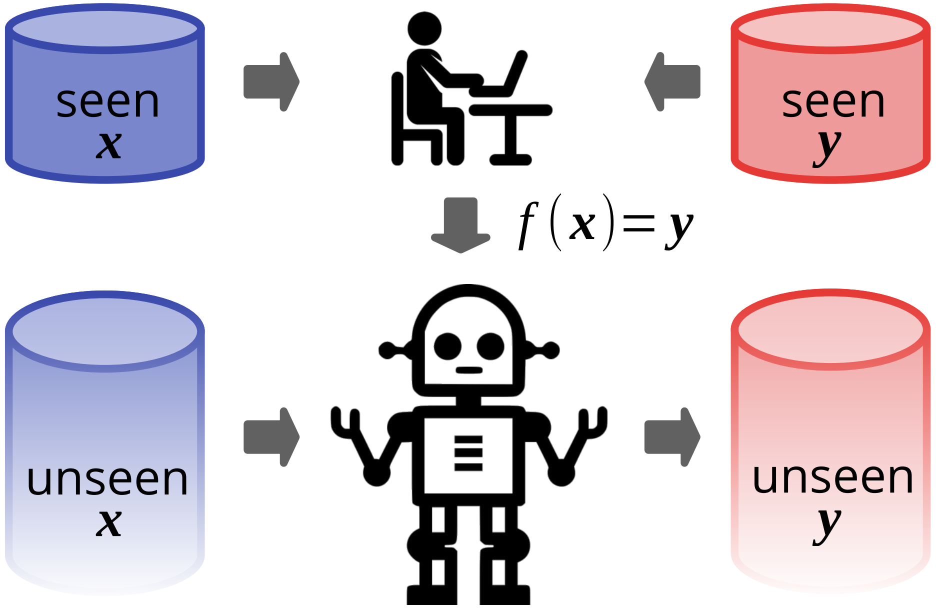
Traditional (Rule-based) Approach:



Supervised ML

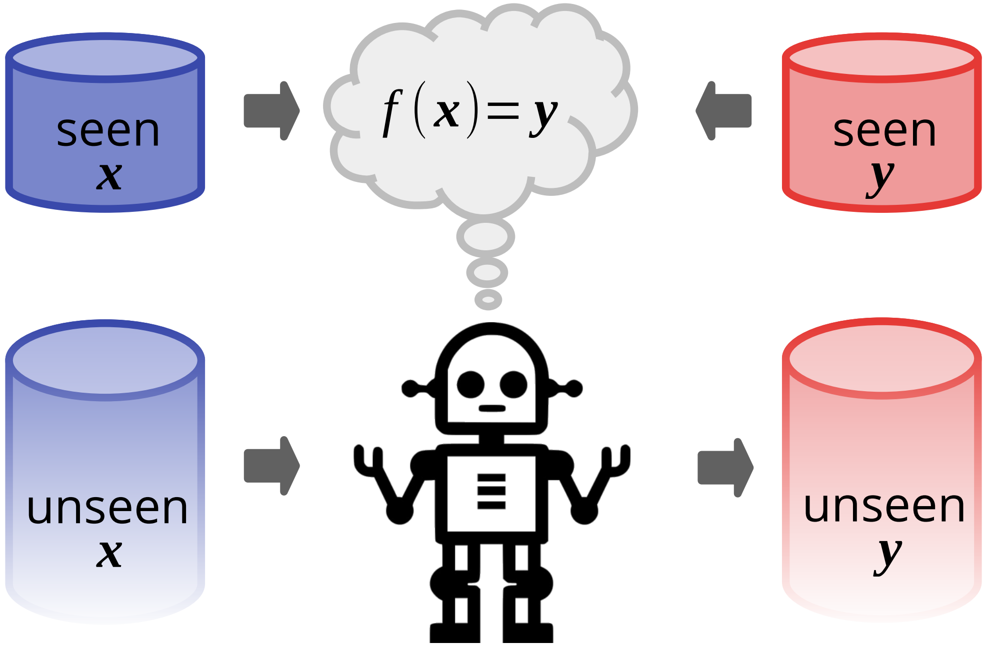
General goal for supervised problems:
Find a function ("task") that relates input data (x) to output data (y) such that: $f(x)=y$

Traditional (Rule-based) Approach:



Input data (x) Output data (y)

Machine-Learning Approach:



Input data (x) Output data (y)

Who am I?

About myself

About myself

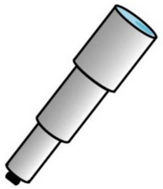
UNIVERSITÄT HEIDELBERG
ZUKUNFT SEIT 1386

Physics

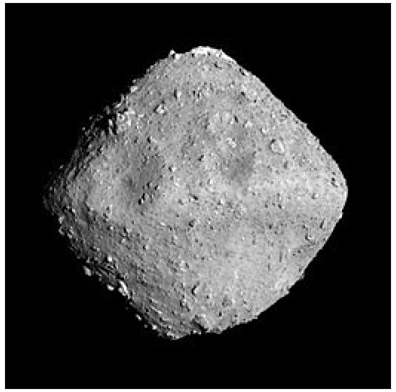
2009

About myself

About myself



ISAS/JAXA



Gerald Rhemann

Deutsches Zentrum für Luft- und Raumfahrt
German Aerospace Center

Northern Arizona University

Universität Heidelberg
Zukunft seit 1386

Freie Universität Berlin

Lowell Observatory
125 years | 1894 - 2019

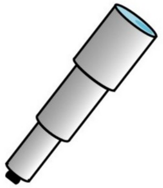
Physics

Dr. rer. nat.
(Earth Sciences)

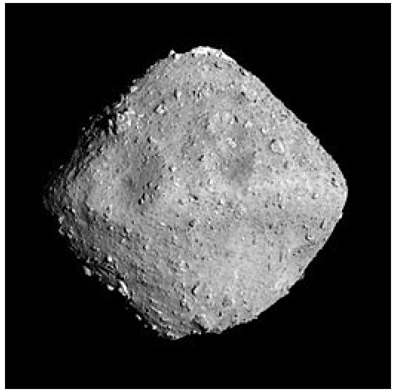
2009

2013

About myself



ISAS/JAXA



Gerald Rhemann

UNIVERSITÄT HEIDELBERG ZUKUNFT SEIT 1386

Deutsches Zentrum für Luft- und Raumfahrt German Aerospace Center

Freie Universität Berlin

NAU NORTHERN ARIZONA UNIVERSITY

LOWELL OBSERVATORY 125 YEARS | 1894 - 2019

Physics

Dr. rer. nat. (Earth Sciences)

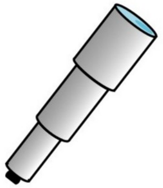
Postdoc @HSG-AIML

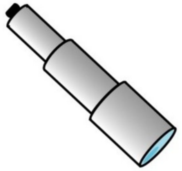
2009

2013

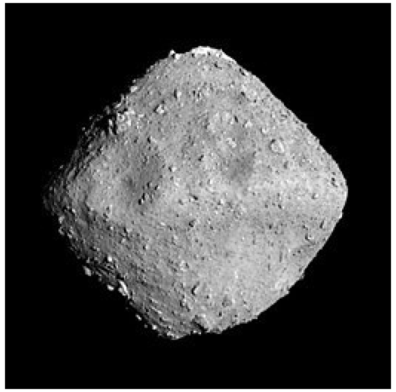
2020

About myself

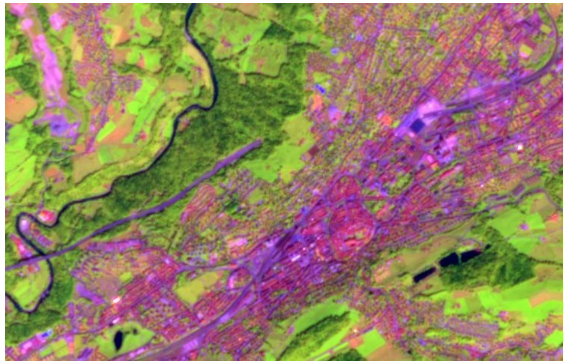


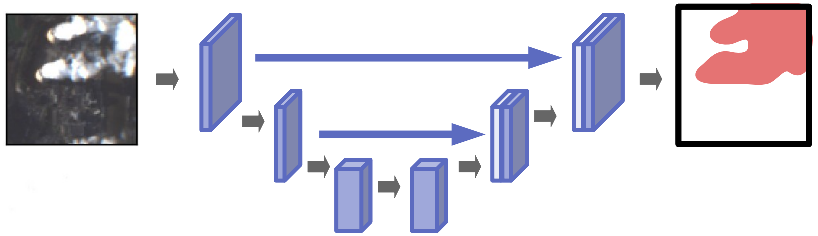


ISAS/JAXA



Gerald Rhemann





Universität Heidelberg – Zukunft seit 1386

Deutsches Zentrum für Luft- und Raumfahrt – German Aerospace Center (DLR)

Freie Universität Berlin

NAU – Northern Arizona University

Lowell Observatory – 125 years | 1894 - 2019

Universität St.Gallen

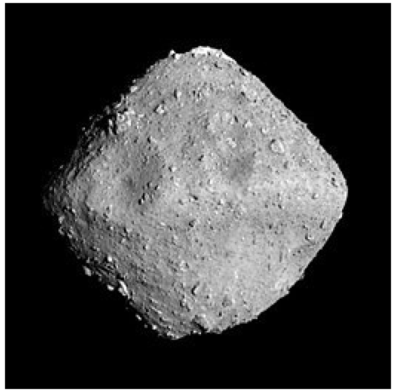
Physics – 2009

Dr. rer. nat. (Earth Sciences) – 2013

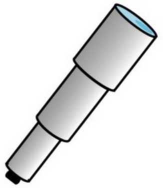
Postdoc @HSG-AIML – 2020

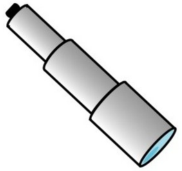
About myself

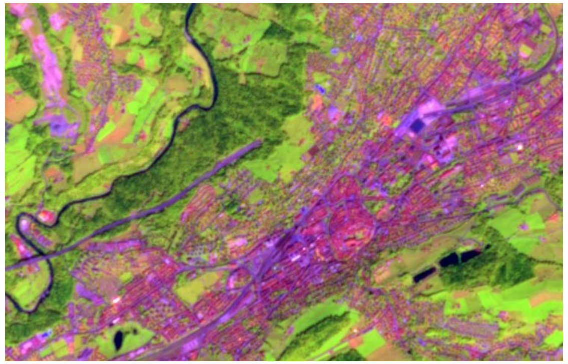
ISAS/JAXA

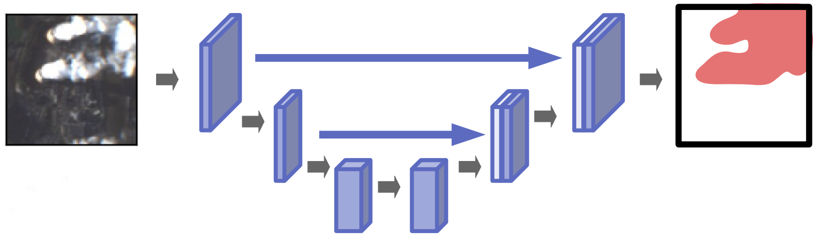


Gerald Rhemann









Universität Heidelberg – Zukunft seit 1386

DLR – Deutsches Zentrum für Luft- und Raumfahrt – German Aerospace Center

Freie Universität Berlin

NAU – Northern Arizona University

Lowell Observatory – 125 years | 1894 - 2019

Universität St.Gallen

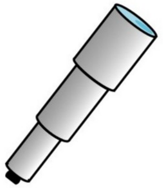
Physics – 2009

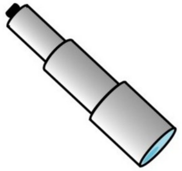
Dr. rer. nat. (Earth Sciences) – 2013

Postdoc @HSG-AIML – 2020

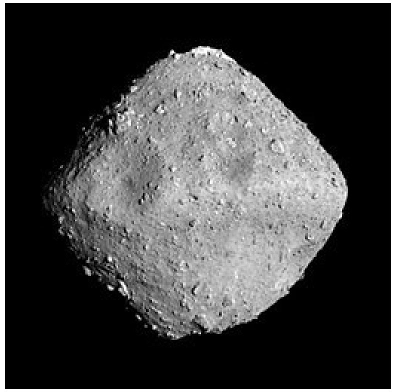
Asst. Prof. Computer Vision – 2022

About myself

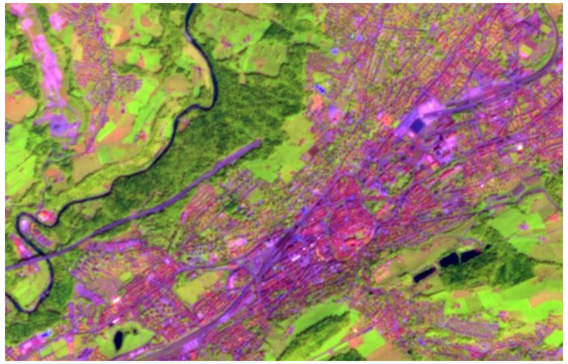


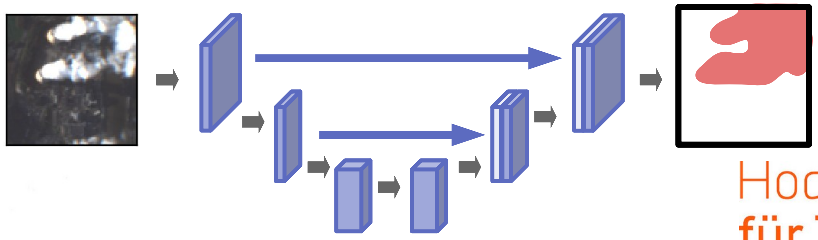


ISAS/JAXA



Gerald Rhemann





Universität Heidelberg – Zukunft seit 1386

Deutsches Zentrum für Luft- und Raumfahrt (DLR) – German Aerospace Center

Freie Universität Berlin

NAU – Northern Arizona University

Lowell Observatory – 125 Years | 1894 - 2019

Universität St.Gallen

Hochschule für Technik Stuttgart

- 2009: Physics
- 2013: Dr. rer. nat. (Earth Sciences)
- 2020: Postdoc @HSG-AIML
- 2022: Asst. Prof. Computer Vision
- 2024: Prof. AI in Remote Sensing

What I work on...

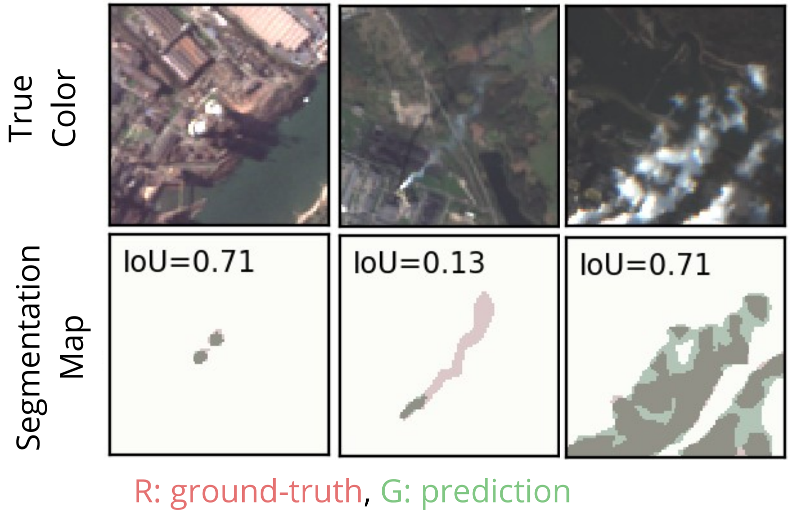
What I work on...

Commercial Vehicle Traffic Monitoring
(*Blattner et al. 2021*)

What I work on...

Commercial Vehicle Traffic Monitoring
(*Blattner et al. 2021*)

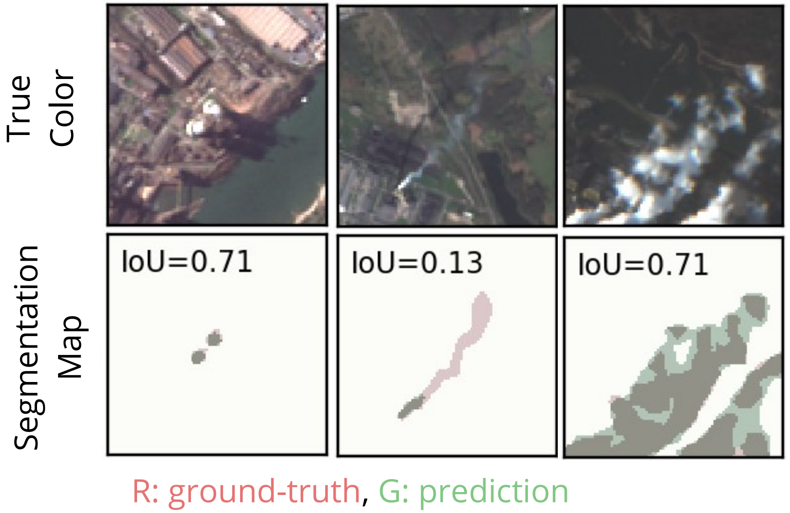
Characterization of Plumes and Estimation of Power Generation from Remote Sensing Data
(*Mommert et al. 2020, Hanna et al. 2023*)

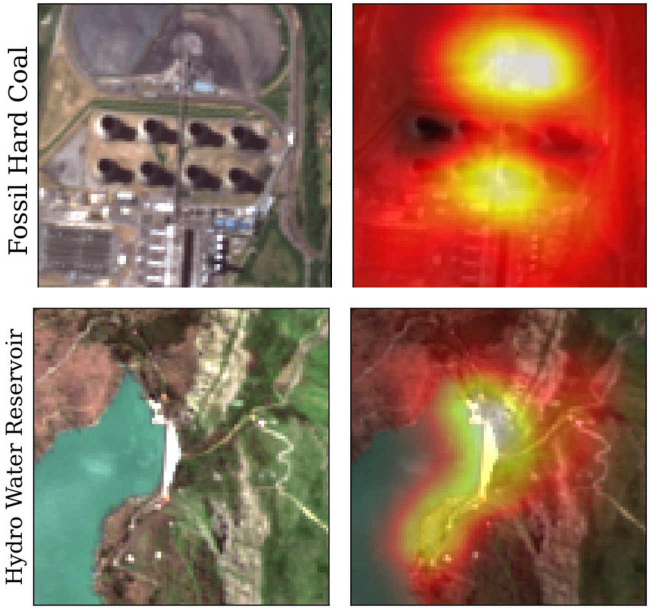


What I work on...

Commercial Vehicle Traffic Monitoring
(*Blattner et al. 2021*)

Characterization of Plumes and Estimation of Power Generation from Remote Sensing Data
(*Mommert et al. 2020, Hanna et al. 2023*)



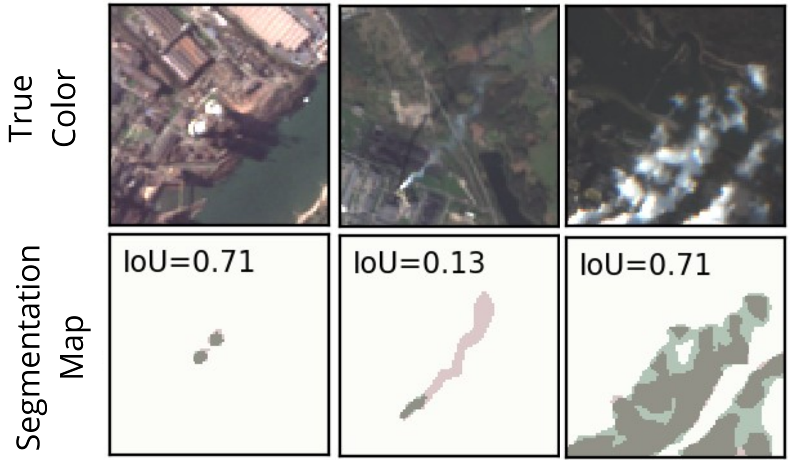


Power Plant Classification from Remote Imaging with Deep Learning
(*Mommert et al. 2021*)

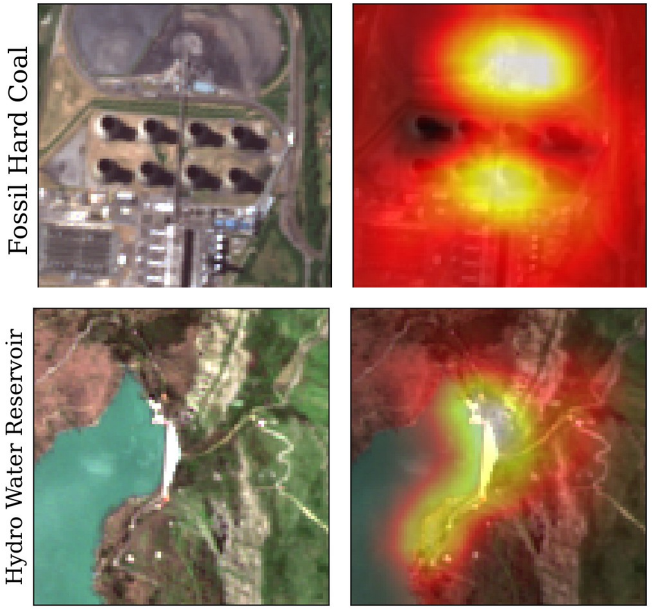
What I work on...

Commercial Vehicle Traffic Monitoring
(*Blattner et al. 2021*)

Characterization of Plumes and Estimation of Power Generation from Remote Sensing Data
(*Mommert et al. 2020, Hanna et al. 2023*)

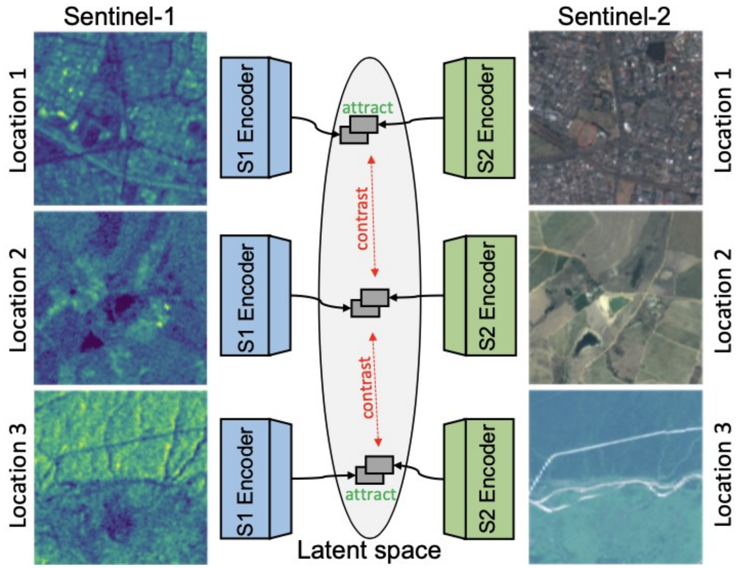


R: ground-truth, G: prediction

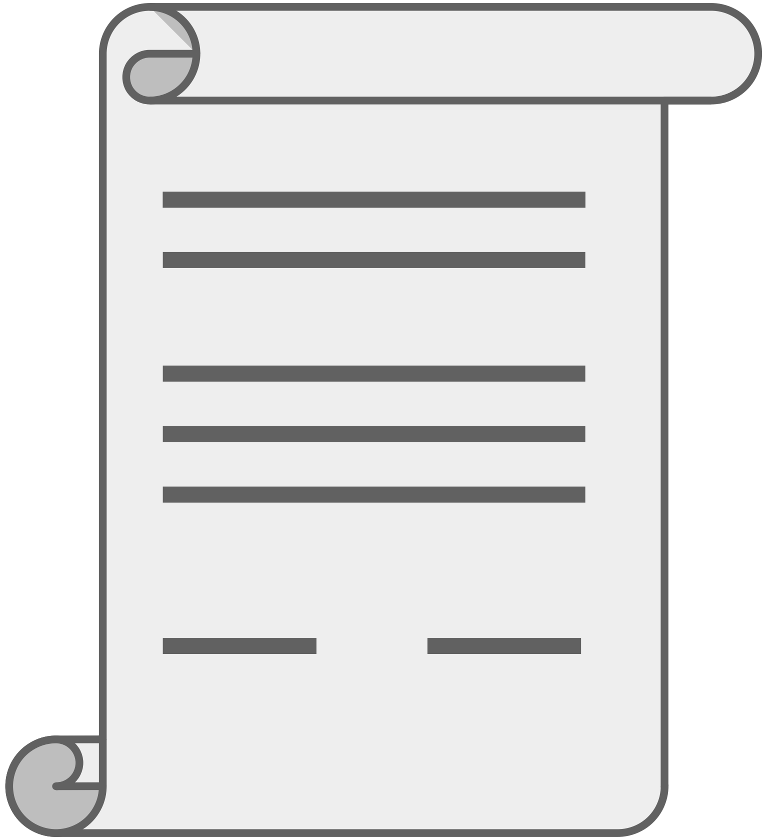


Power Plant Classification from Remote Imaging with Deep Learning
(*Mommert et al. 2021*)

Contrastive Self-supervised data fusion for Satellite Imagery
(Scheibenreif et al. 2022)



Course modalities

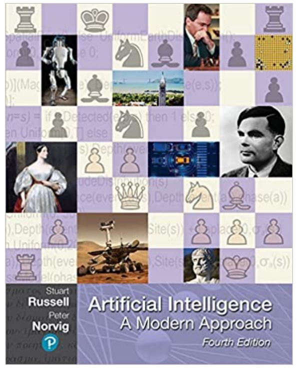


Course modalities

- **Goal** of this course:
  To understand and be able to implement and utilize supervised traditional Machine Learning and Deep Learning models.
- **Setup**: Combination of lectures and voluntary hands-on lab courses
- **Lecture mode**: This course is supposed to be bi-directional: let me know if anything is unclear, ask questions anytime!
- We will use **Google Colab** for running our Lab Notebooks (they offer free GPUs!). If you don't have a Google account, please let me know as soon as possible!

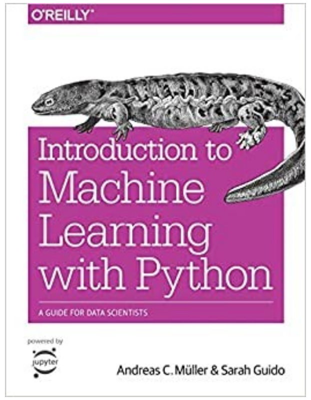
Literature resources

- Stuart Russell, Peter Norvig: **Artificial Intelligence: A Modern Approach** (2020 and earlier versions, MIT Press)
  Part V ("Learning") is especially relevant to this course and provides good introductions

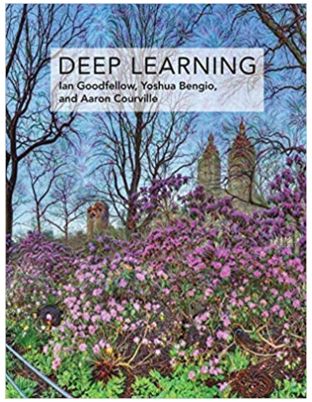


ebook@HSG

- Andreas Müller & Sarah Guido: **Introduction to Machine Learning with Python** (2017, O'Reilly)
  Easy-to-understand introduction to Python for ML, uses scikit-learn

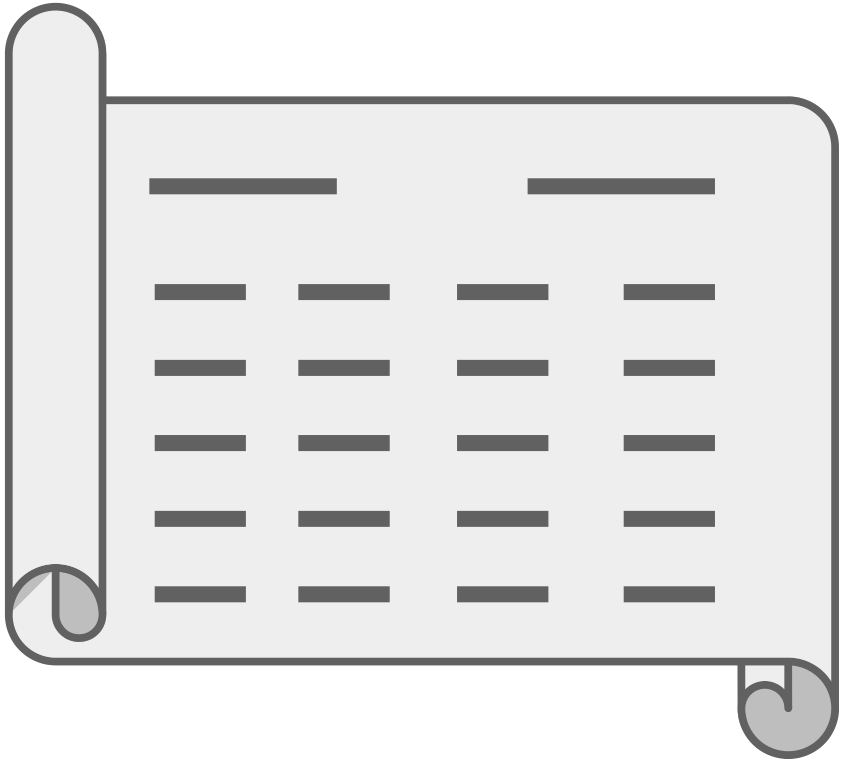


- Ian Goodfellow, Yoshua Bengio, Aaron Courville: **Deep Learning** (2016, MIT Press)
  All you need to know about Deep Learning



free online

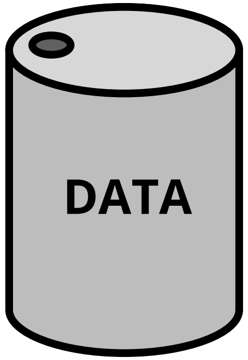
Course syllabus



Content

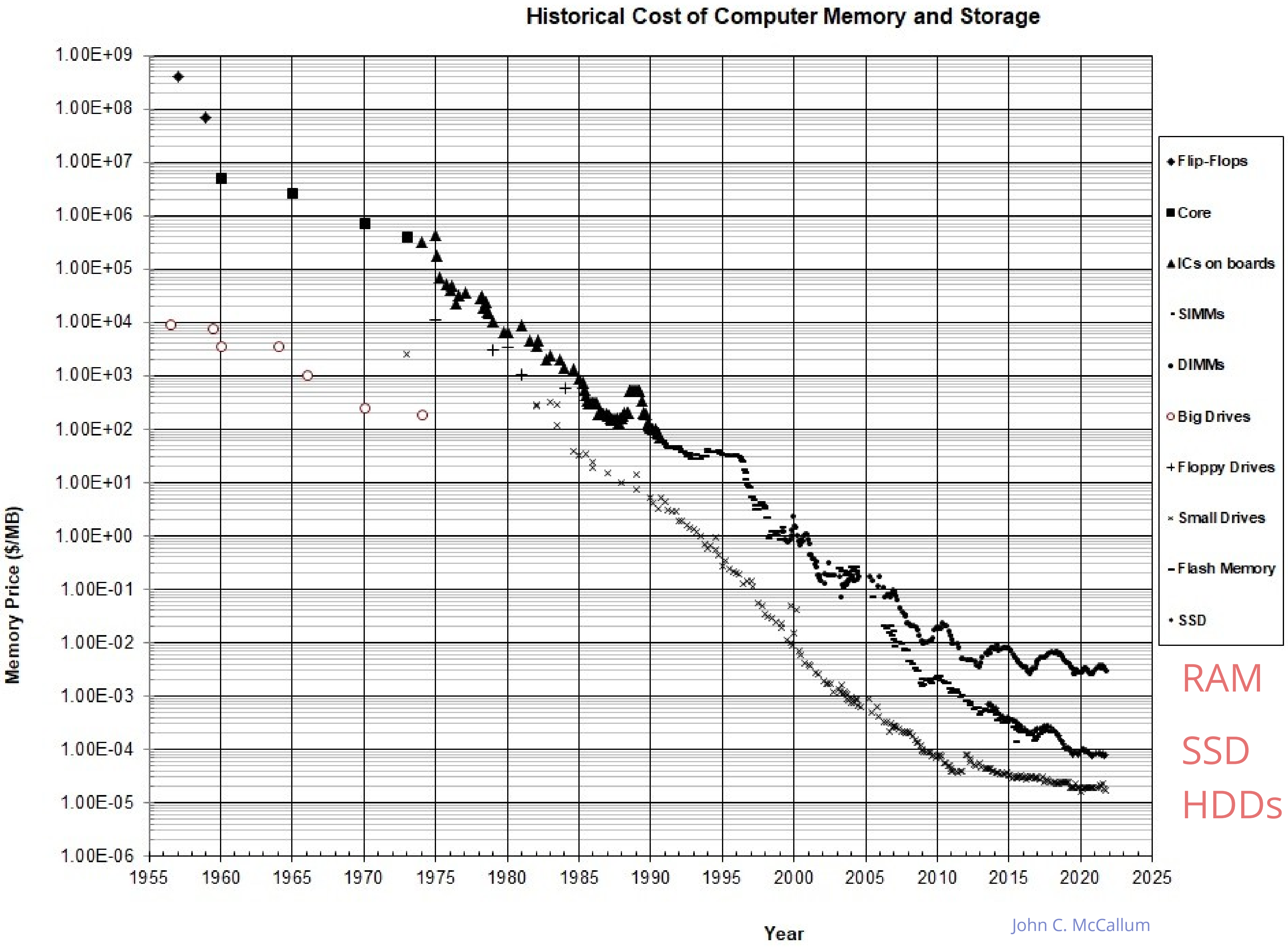
Slot	Wednesday	Thursday
09:00 - 10:30	**Intro & Data**	**Neural Networks**
10:30 – 10:45	Break	Break
10:45 – 12:15	**Supervised ML: Concepts**	**Convolutional Neural Networks & Computer Vision**
12:15 – 13:45	Lunch break	Lunch break
13:45 – 15:15	**Supervised ML: Methods**	**Lab: Neural Networks**
15:15 – 15:30	Break	Break
15:30 – 17:00	**Lab: Supervised ML**	**Advanced Deep Learning**

Data



Data storage

- Data storage used to be a bottleneck – not anymore!



John C. McCallum

Data storage

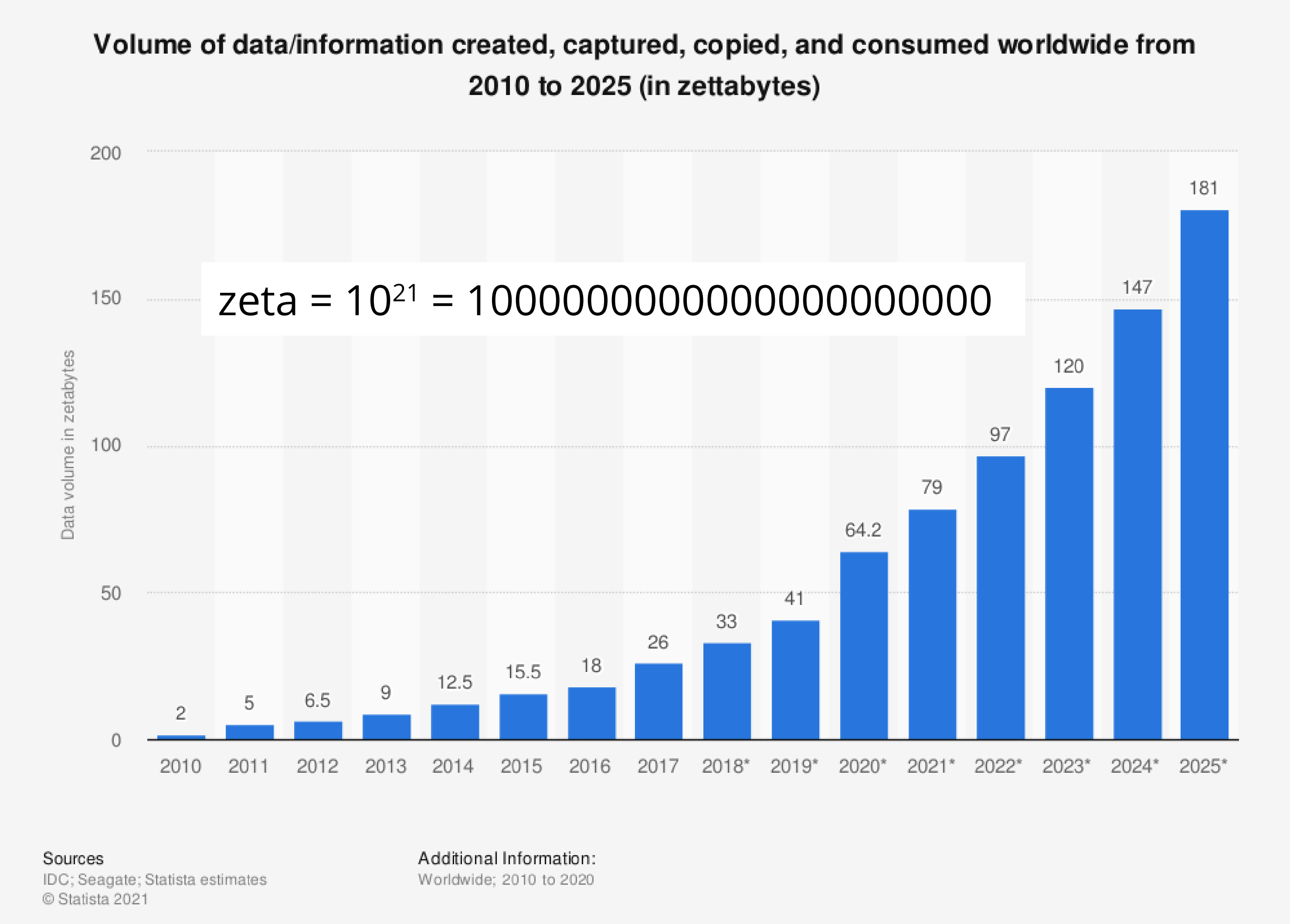
- Data storage used to be a bottleneck – not anymore!

Data storage

- Data storage used to be a bottleneck – not anymore!
- Vast amounts of data can now be stored easily

Data storage

- Data storage used to be a bottleneck – not anymore!
- Vast amounts of data can now be stored easily



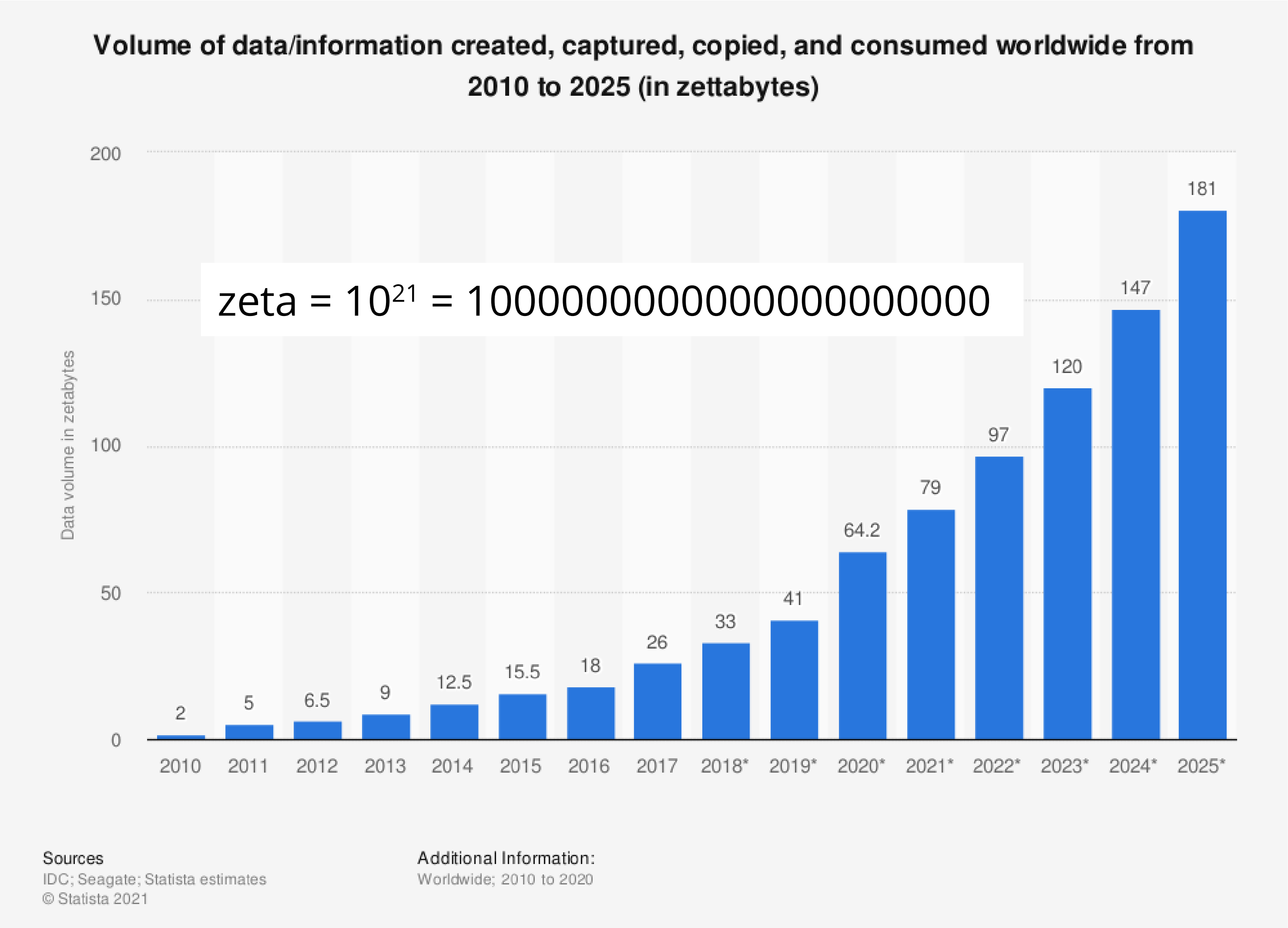
Statista

Data storage

- Data storage used to be a bottleneck – not anymore!

- Vast amounts of data can now be stored easily

- Is all this data technically accessible for analysis?
  (of course not, since most of it is privately owned, but...)



Statista

Structured vs unstructured data

Structured vs unstructured data

Structured data

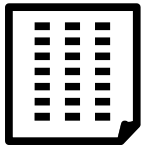
Preprocessed and formatted data that is easily queryable.

Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

Quantitative data

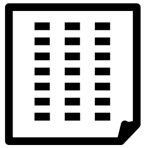


Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

Quantitative data



Unstructured data

Unprocessed and unformatted data is not easily queryable.

Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

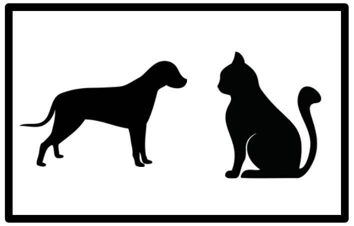
Quantitative data

Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

Image data



Video data

Textual data

Data stream

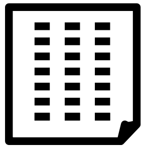
Audio data

Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

Quantitative data



Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

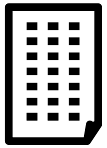
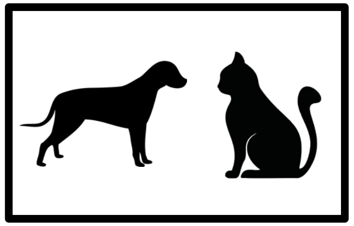
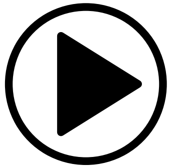


Image data

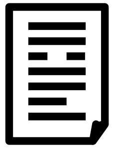


Video data



Data complexity

Textual data



Data stream

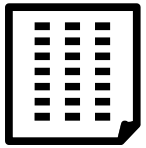
Audio data

Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

Quantitative data



Most data analysis techniques require data to be available in a structured form for easier processing.

Structured data can always be represented in a database **schema** (e.g., a table in 2 dimensions).

Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

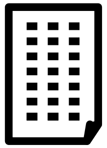
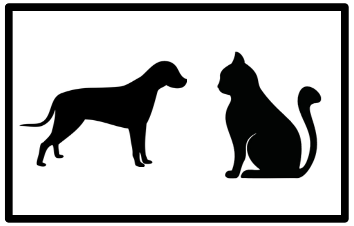
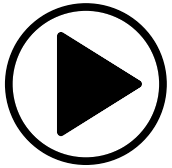


Image data

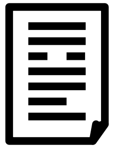


Video data



Data complexity

Textual data



Data stream

Audio data

Structured vs unstructured data

Structured data

Preprocessed and formatted data that is easily queryable.

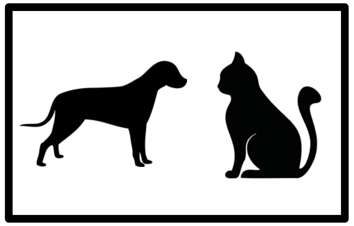
Quantitative data

Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

Image data



Video data

Data complexity

Textual data

Data stream

Audio data

Most data analysis techniques require data to be available in a structured form for easier processing.

Structured data can always be represented in a database **schema** (e.g., a table in 2 dimensions).

Quantitative and qualitative data

Quantitative data (can be measured; distances can be defined)	Qualitative (categorical) data (cannot be measured; distances not defined)

Quantitative and qualitative data

Quantitative data
(can be measured; distances can be defined)

Qualitative (categorical) data
(cannot be measured; distances not defined)

Continuous data

Real-valued numbers; potentially within a given range

Examples:
- Temperatures
- A person’s height
- Prices

Quantitative and qualitative data

Quantitative data
(can be measured; distances can be defined)

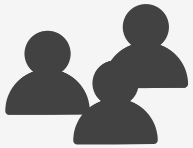
Continuous data

Real-valued numbers; potentially within a given range

Examples:
- Temperatures
- A person's height
- Prices

Discrete data

Discrete numbers; whole numbers or real numbers, potentially within a given range



Examples:
- Number of people in a room
- Inventory counts

Qualitative (categorical) data
(cannot be measured; distances not defined)

Quantitative and qualitative data

Quantitative data
(can be measured; distances can be defined)

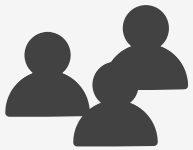
Continuous data

Real-valued numbers; potentially within a given range

Examples:
- Temperatures
- A person's height
- Prices

Discrete data

Discrete numbers; whole numbers or real numbers, potentially within a given range

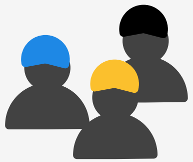


Examples:
- Number of people in a room
- Inventory counts

Qualitative (categorical) data
(cannot be measured; distances not defined)

Nominal data

Labels for different categories without ordering



Examples:
- Color of hair
- Names of persons
- Types of fruit

Quantitative and qualitative data

Quantitative data
(can be measured; distances can be defined)

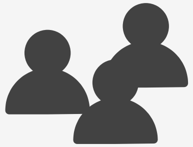
Continuous data

Real-valued numbers; potentially within a given range

Examples:
- Temperatures
- A person’s height
- Prices

Discrete data

Discrete numbers; whole numbers or real numbers, potentially within a given range

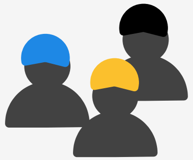


Examples:
- Number of people in a room
- Inventory counts

Qualitative (categorical) data
(cannot be measured; distances not defined)

Nominal data

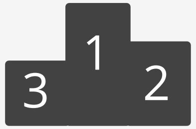
Labels for different categories without ordering



Examples:
- Color of hair
- Names of persons
- Types of fruit

Ordinal data

Labels for different categories following an inherent ranking scheme.



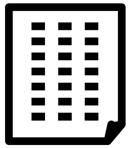
Examples:
- Rank in a competition
- Grades
- Day of the week

Turning unstructured data into structured data

Structured data

Preprocessed and formatted data that is easily queryable.

Quantitative



Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

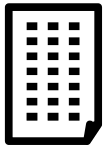
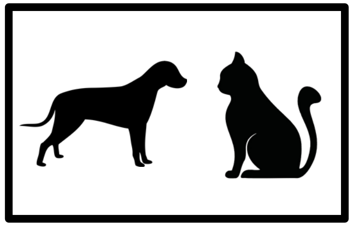
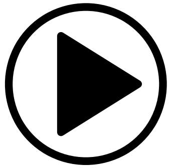


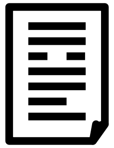
Image data



Video data



Textual data



Data stream

Audio data

Turning unstructured data into structured data

Structured data

Preprocessed and formatted data that is easily queryable.

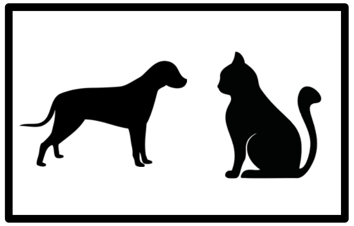
Quantitative

Unstructured data

Unprocessed and unformatted data is not easily queryable.

Qualitative data

Image data



Video data

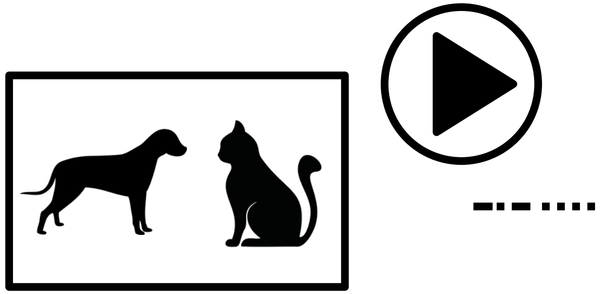
Textual data

Data stream

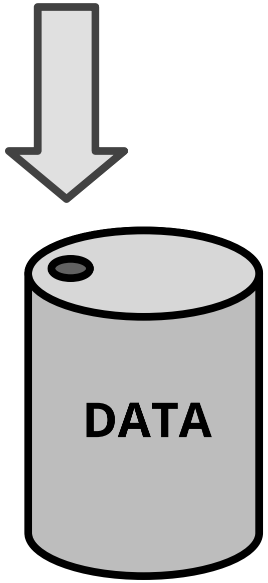
Audio data

Before ML methods can be applied to unstructured data, we have to process those and extract useful features from them.

This process is called **feature engineering**.



Features and Feature Engineering

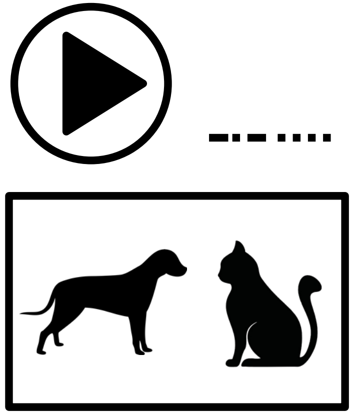


What are features?

Features are quantitative and independent variables based on which our ML models learn.

What are features?

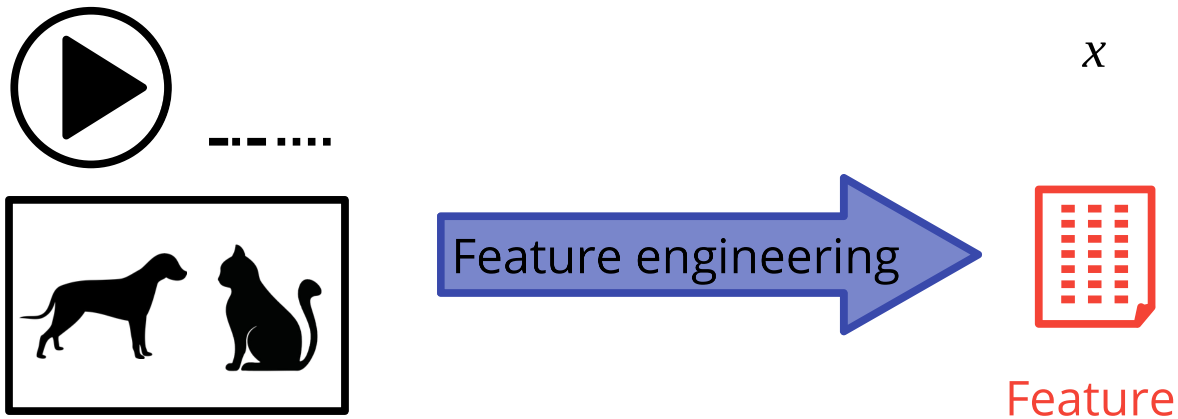
Features are quantitative and independent variables based on which our ML models learn.



Raw data
(qualitative)

What are features?

Features are quantitative and independent variables based on which our ML models learn.

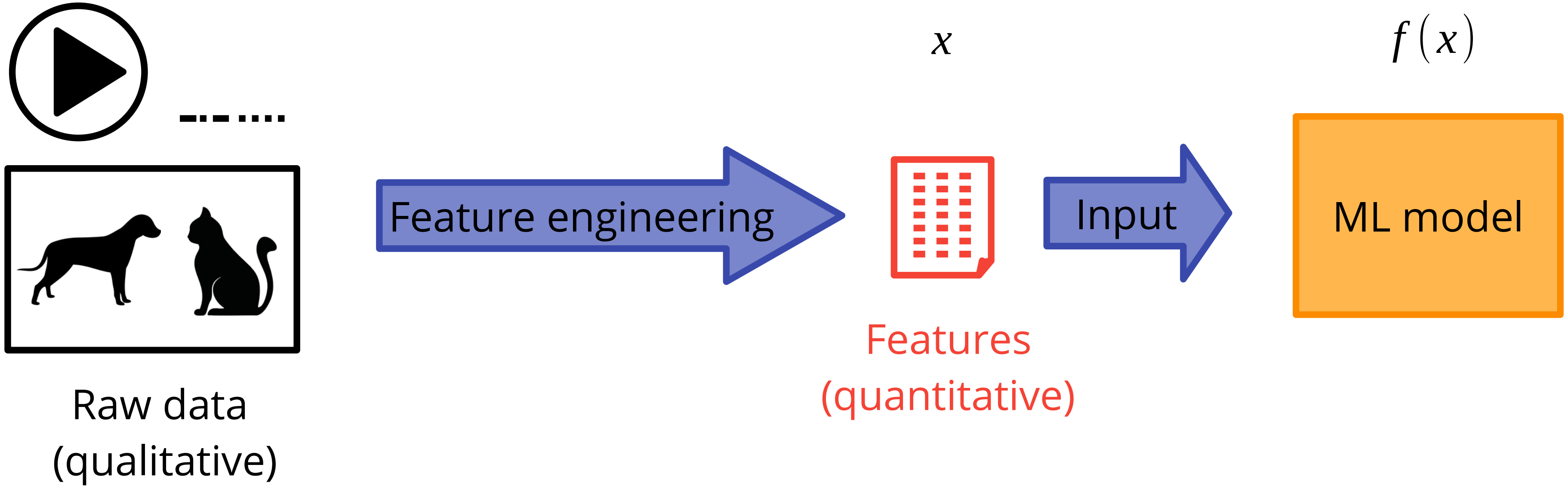


Raw data
(qualitative)

Features
(quantitative)

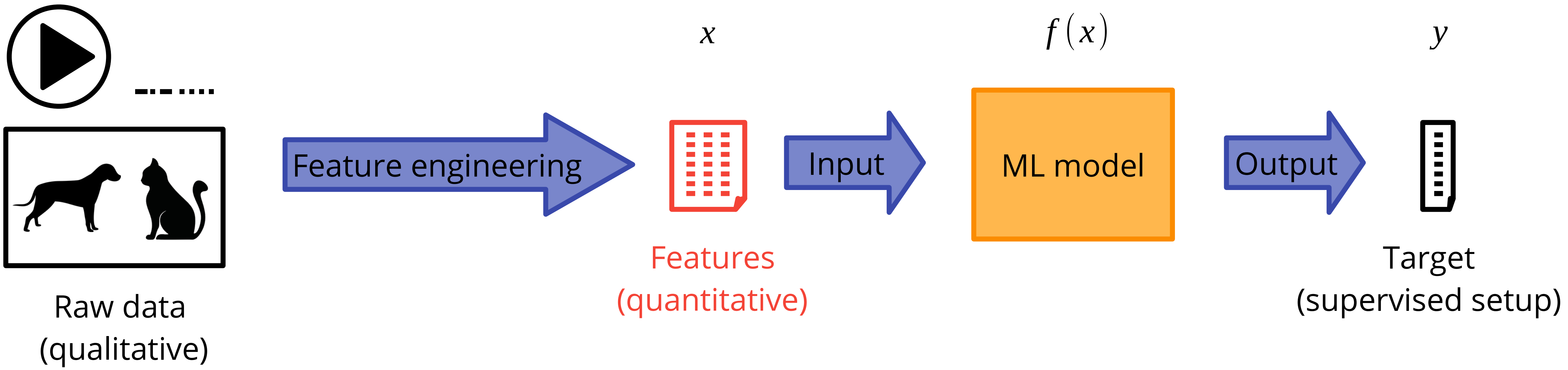
What are features?

Features are quantitative and independent variables based on which our ML models learn.



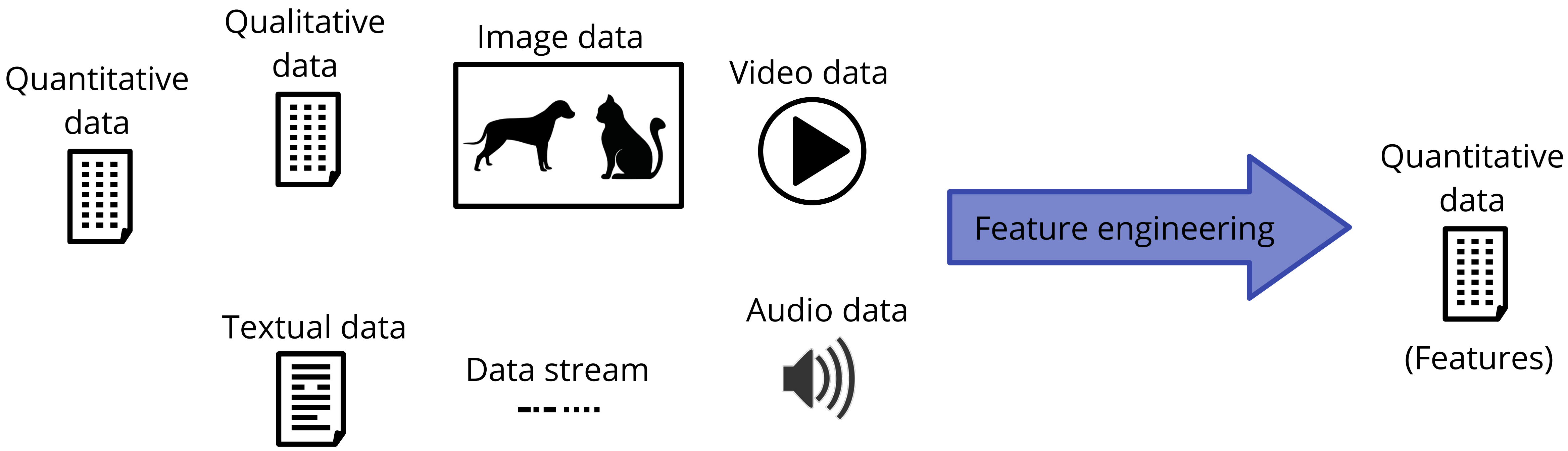
What are features?

Features are quantitative and independent variables based on which our ML models learn.



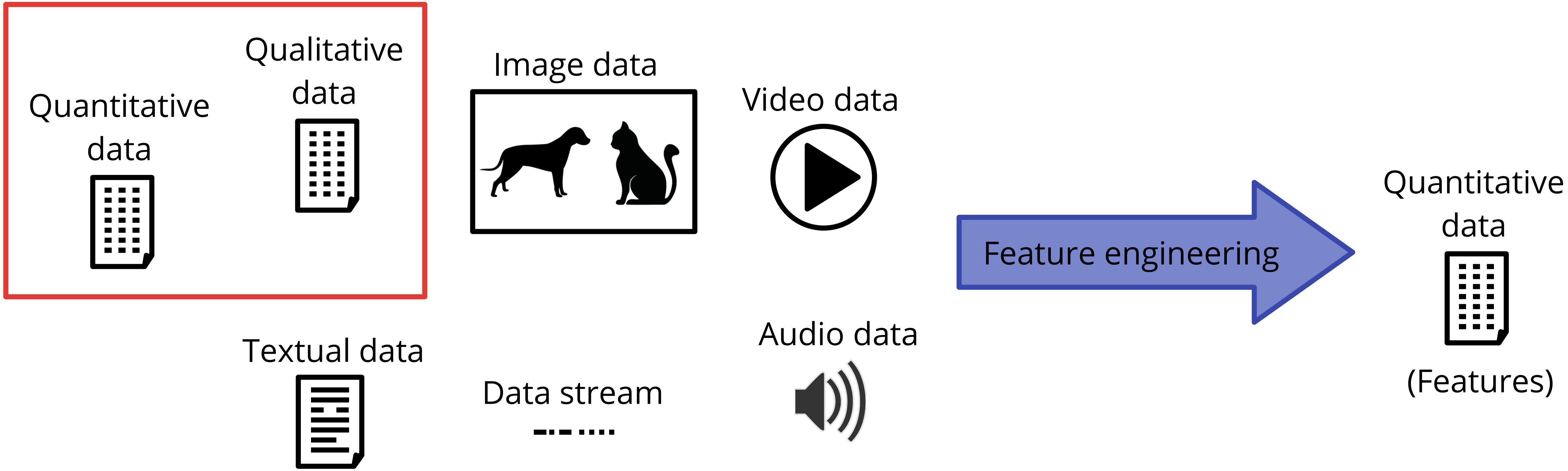
Feature engineering

Extract or create features that may provide a ML model with rich information on its task based on **domain knowledge**. Feature engineering can be applied to raw data, resulting in quantitative data that can be directly fed into the ML model (features).



Feature engineering

Extract or create features that may provide a ML model with rich information on its task based on **domain knowledge**. Feature engineering can be applied to raw data, resulting in quantitative data that can be directly fed into the ML model (features).



Feature engineering – quantitative data

Create meaningful features through mathematical transformations.

Examples:

Feature engineering – quantitative data

Create meaningful features through mathematical transformations.

Examples:

Arithmetic

Situation: You have two variables, x_1 and x_2, but you are more interested in their difference, δ.

Transformation:

$$\delta = x_1 - x_2$$

Feature engineering – quantitative data

Create meaningful features through mathematical transformations.

Examples:

Arithmetic

Situation: You have two variables, x_1 and x_2, but you are more interested in their difference, δ.

Transformation:

$$\delta = x_1 - x_2$$

Aggregation of Features

Situation: You have results from different business units, $\boldsymbol{x}_i$, but your ML model should not consider the results separately, but as an aggregated overall result, $\boldsymbol{x}$.

Transformation:

$$\boldsymbol{x} = \sum_i \boldsymbol{x}_i$$

Feature engineering – quantitative data

Create meaningful features through mathematical transformations.

Examples:

Arithmetic

Situation: You have two variables, x_1 and x_2, but you are more interested in their difference, δ.

Transformation:

$$\delta = x_1 - x_2$$

Aggregation of Features

Situation: You have results from different business units, $\boldsymbol{x_i}$, but your ML model should not consider the results separately, but as an aggregated overall result, $\boldsymbol{x}$.

Transformation:

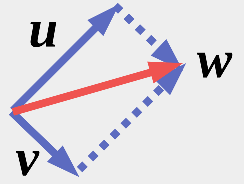
$$\boldsymbol{x} = \sum_i \boldsymbol{x_i}$$

Geometric Transformations

Situation: To identify common wind speed patterns, you have measurements of two orthogonal wind speed components, $\boldsymbol{u}$ and $\boldsymbol{v}$. Since only the magnitude of the resulting wind vector, $\boldsymbol{w}$, matters, you can utilize its magnitude, $|\boldsymbol{w}|$.

Transformation:

$$|\boldsymbol{w}| = \sqrt{\boldsymbol{u}^2 + \boldsymbol{v}^2}$$



Feature engineering – qualitative data

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: ordinal (ranked) data → discrete quantitative data
  The intuition is that the ranking/order of the classes is conserved in a discrete numerical schema and a "distance" can be defined.

  Examples:

  - Competition ranks: [1st, 2nd, 3rd, 4th, 5th] → [1, 2, 3, 4, 5]
  - Cloudiness scale: [clear, mostly clear, partly cloudy, mostly cloudy] → [0, 1, 2, 3]
  - Quality scale: [very good, good, satisfying, sufficient, insufficient] → [0, 1, 2, 3, 4]
  - Days of the week: [Mon, Tue, Wed, Thu, Fri, Sat, Sun] → [1, 2, 3, 4, 5, 6, 7]

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: ordinal (ranked) data → discrete quantitative data
  The intuition is that the ranking/order of the classes is conserved in a discrete numerical schema and a "distance" can be defined.

  Examples:

  - Competition ranks: [1st, 2nd, 3rd, 4th, 5th] → [1, 2, 3, 4, 5]
  - Cloudiness scale: [clear, mostly clear, partly cloudy, mostly cloudy] → [0, 1, 2, 3]
  - Quality scale: [very good, good, satisfying, sufficient, insufficient] → [0, 1, 2, 3, 4]
  - Days of the week: [Mon, Tue, Wed, Thu, Fri, Sat, Sun] → [1, 2, 3, 4, 5, 6, 7]

← be careful: day of week is cyclical!

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: ordinal (ranked) data → discrete quantitative data
  The intuition is that the ranking/order of the classes is conserved in a discrete numerical schema and a "distance" can be defined.

  Examples:

  - Competition ranks: [1st, 2nd, 3rd, 4th, 5th] → [1, 2, 3, 4, 5]
  - Cloudiness scale: [clear, mostly clear, partly cloudy, mostly cloudy] → [0, 1, 2, 3]
  - Quality scale: [very good, good, satisfying, sufficient, insufficient] → [0, 1, 2, 3, 4]
  - Days of the week: [Mon, Tue, Wed, Thu, Fri, Sat, Sun] → [1, 2, 3, 4, 5, 6, 7]

← be careful: day of week is cyclical!

(Caveat: Label encoding can also be used if a large number of classes is present)

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: *ordinal (ranked) data → discrete quantitative data*

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: *ordinal (ranked) data → discrete quantitative data*
- **One-hot encoding**: *nominal (unranked) data → binary coding of labels*
  For each possible class in a feature, a binary feature is introduced; for each sample, all one-hot features are zero, only those that match have a value of one.
  Examples:
  - House properties: [balcony, cellar, fireplace, jacuzzi] →
    samples: house 1: "balcony" →
    house 2: "fireplace" →
    house 3: "balcony and jacuzzi" →
    house 4: "cellar, fireplace and jacuzzi" →

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: *ordinal (ranked) data → discrete quantitative data*
- **One-hot encoding**: *nominal (unranked) data → binary coding of labels*
  For each possible class in a feature, a binary feature is introduced; for each sample, all one-hot features are zero, only those that match have a value of one.
  Examples:
  - House properties: [balcony, cellar, fireplace, jacuzzi] →
    samples: house 1: "balcony" →
    house 2: "fireplace" →
    house 3: "balcony and jacuzzi" →
    house 4: "cellar, fireplace and jacuzzi" →

balcony	cellar	fireplace	jacuzzi
1	0	0	0
0	0	1	0
1	0	0	1
0	1	1	1

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: *ordinal (ranked) data → discrete quantitative data*
- **One-hot encoding**: *nominal (unranked) data → binary coding of labels*
  For each possible class in a feature, a binary feature is introduced; for each sample, all one-hot features are zero, only those that match have a value of one.
  Examples:
  - House properties: [balcony, cellar, fireplace, jacuzzi] →
    samples: house 1: “balcony” →
    house 2: ”fireplace” →
    Multi-class feature: house 3: ”balcony and jacuzzi” →
    house 4: “cellar, fireplace and jacuzzi” →

balcony	cellar	fireplace	jacuzzi
1	0	0	0
0	0	1	0
1	0	0	1
0	1	1	1

Feature engineering – qualitative data

Qualitative (categorical) data cannot be fed into ML models directly, they have to be turned into quantitative data first. There are two common methods available, depending on the data type:

- **Label encoding**: *ordinal (ranked) data → discrete quantitative data*
- **One-hot encoding**: *nominal (unranked) data → binary coding of labels*
  For each possible class in a feature, a binary feature is introduced; for each sample, all one-hot features are zero, only those that match have a value of one.
  Examples:
  - House properties: [balcony, cellar, fireplace, jacuzzi] →
    samples: house 1: “balcony” →
    house 2: ”fireplace” →
    Multi-class feature: house 3: ”balcony and jacuzzi” →
    house 4: “cellar, fireplace and jacuzzi” →

balcony	cellar	fireplace	jacuzzi
1	0	0	0
0	0	1	0
1	0	0	1
0	1	1	1

(Caveat: if too many classes present, use label encoding instead; see *curse of dimensionality*)

Final data set nomenclature

Final data set nomenclature

Feature engineering results in a compilation of features that we can use to train our ML models.

Example:

Weight	Height	Wings	Legs	Cuteness
0.1	0.1	true	2	1
3.5	0.3	false	4	1
12.0	0.7	false	4	1
500	1.8	false	4	2
800	3.0	true	4	3
...	...	...	...	...

Pet	Type
true	bird
true	cat
true	dog
false	rhinoceros
false	chimera
...	...

Final data set nomenclature

Feature engineering results in a compilation of features that we can use to train our ML models.

Example:

Features/Attributes (input variables, $\boldsymbol{x}$) $f(\boldsymbol{x}) = \boldsymbol{y}$ **Targets**/Labels (output variables, $\boldsymbol{y}$)

Ground-Truth

Weight	Height	Wings	Legs	Cuteness
0.1	0.1	true	2	1
3.5	0.3	false	4	1
12.0	0.7	false	4	1
500	1.8	false	4	2
800	3.0	true	4	3
...	...	...	...	...

Pet	Type
true	bird
true	cat
true	dog
false	rhinoceros
false	chimera
...	...

Final data set nomenclature

Feature engineering results in a compilation of features that we can use to train our ML models.

Example:

Features/Attributes (input variables, $\boldsymbol{x}$)

$f(\boldsymbol{x}) = \boldsymbol{y}$

Targets/Labels (output variables, $\boldsymbol{y}$)

Ground-Truth

Samples/Instances

Weight	Height	Wings	Legs	Cuteness
0.1	0.1	true	2	1
3.5	0.3	false	4	1
12.0	0.7	false	4	1
500	1.8	false	4	2
800	3.0	true	4	3
...	...	...	...	...

Pet	Type
true	bird
true	cat
true	dog
false	rhinoceros
false	chimera
...	...

Final data set nomenclature

Feature engineering results in a compilation of features that we can use to train our ML models.

Example:

Features/Attributes (input variables, $\boldsymbol{x}$) $f(\boldsymbol{x}) = \boldsymbol{y}$ **Targets**/Labels (output variables, $\boldsymbol{y}$)

Samples/Instances

Weight	Height	Wings	Legs	Cuteness
0.1	0.1	true	2	1
3.5	0.3	false	4	1
12.0	0.7	false	4	1
500	1.8	false	4	2
800	3.0	true	4	3
...	...	...	...	...

Ground-Truth

Pet	Type
true	bird
true	cat
true	dog
false	rhinoceros
false	chimera
...	...

classes of label "Type"

Final data set nomenclature

Feature engineering results in a compilation of features that we can use to train our ML models.

Example:

Features/Attributes (input variables, $\boldsymbol{x}$) $f(\boldsymbol{x}) = \boldsymbol{y}$ **Targets**/Labels (output variables, $\boldsymbol{y}$)

Samples/Instances

Weight	Height	Wings	Legs	Cuteness
0.1	0.1	true	2	1
3.5	0.3	false	4	1
12.0	0.7	false	4	1
500	1.8	false	4	2
800	3.0	true	4	3
...	...	...	...	...

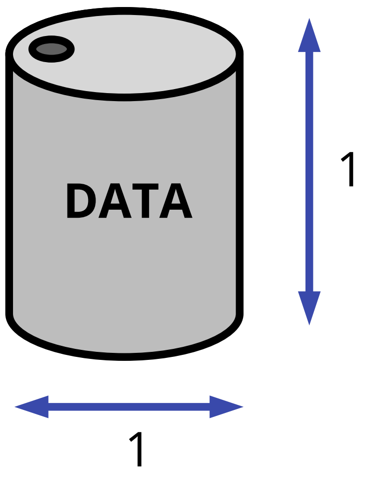
Ground-Truth

Pet	Type
true	bird
true	cat
true	dog
false	rhinoceros
false	chimera
...	...

classes of label "Type"

Data Types: continuous, continuous, binary, discrete, ordinal; binary, categorical (multi-class)

Data scaling



Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

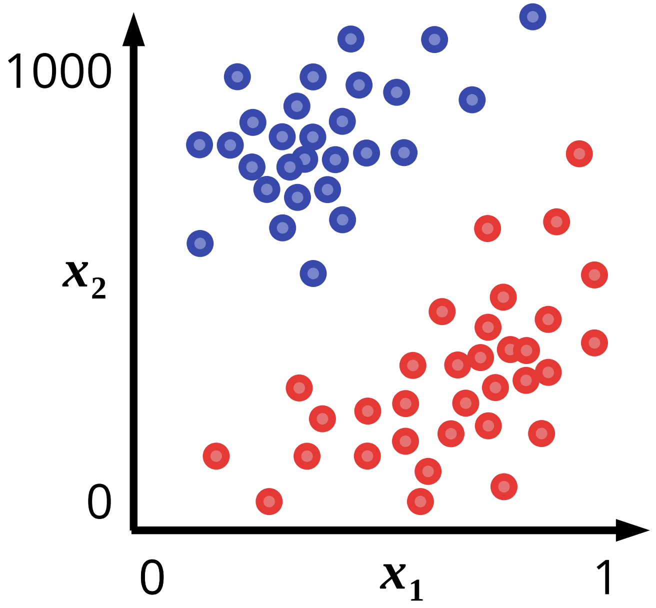
- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.

Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

- Many ML models are based on a notion of "distance" between samples; improperly scaled data may jeopardize the learning capability of such models.

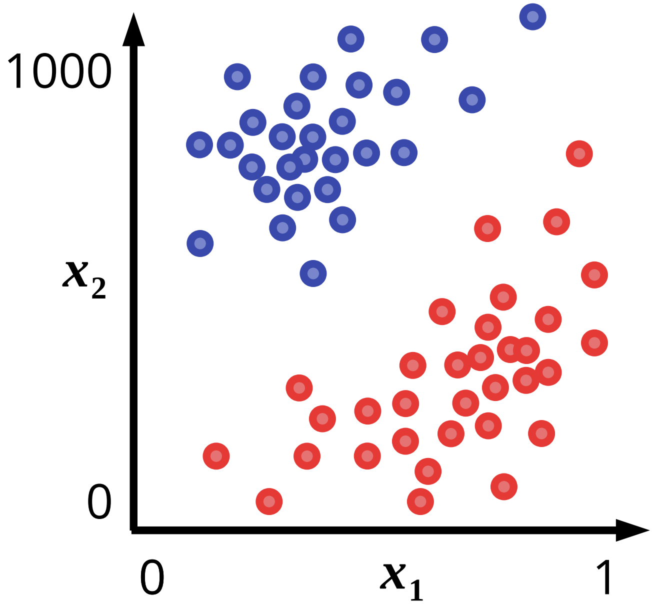


Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

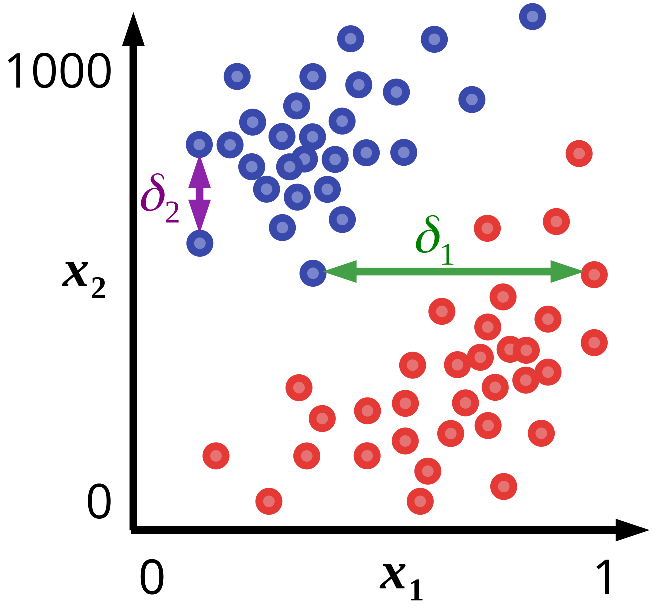
Euclidean distance metric

Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

Euclidean distance metric

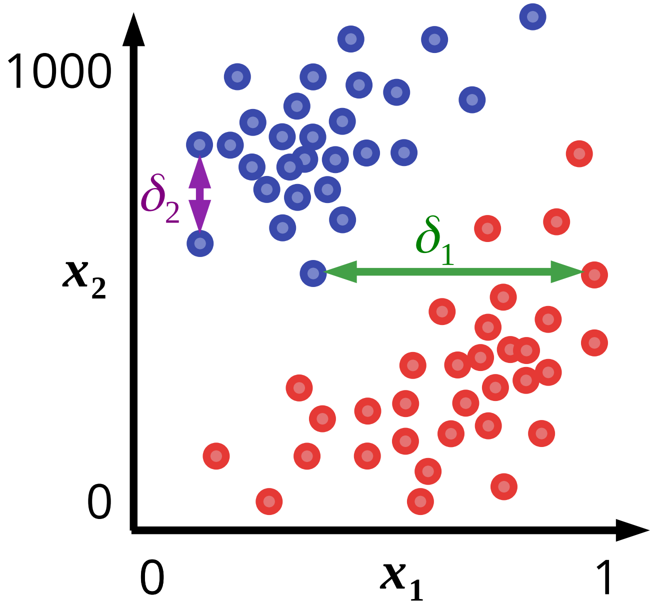
$$\delta_1 \ll \delta_2$$

Data scaling

Data scaling means to linearly transform your data in order to normalize them.

Why scale data?

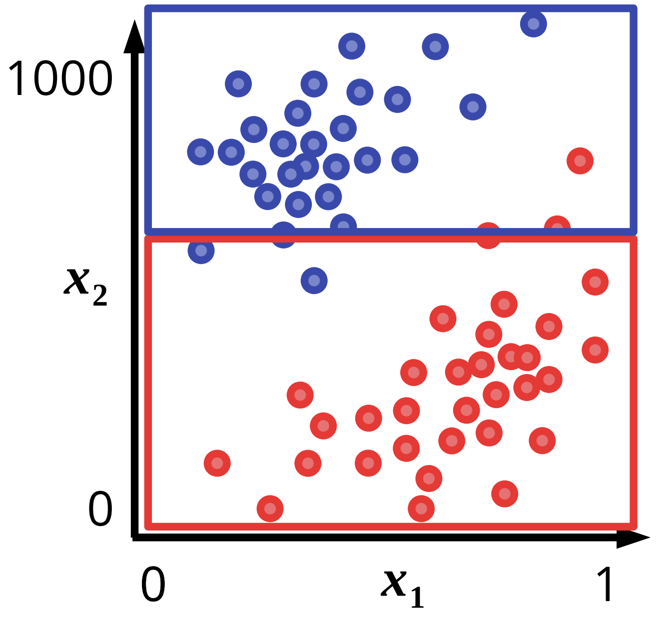
- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

Euclidean distance metric

$$\delta_1 \ll \delta_2$$



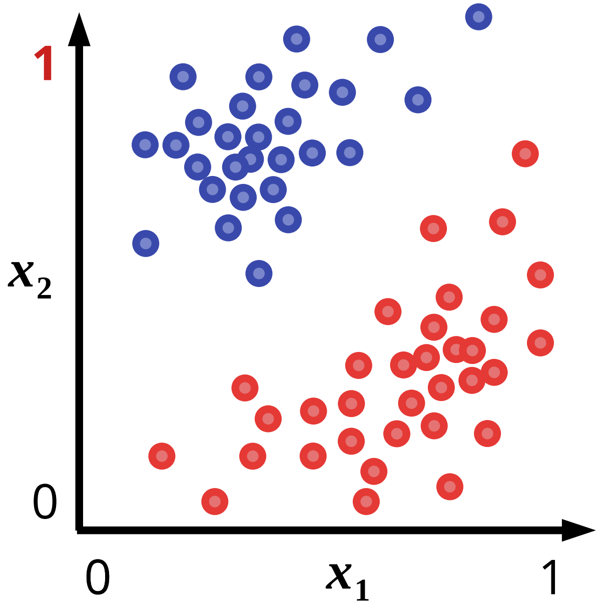
Decision regions of a hypothetical distance-based classifier.

Results are ok-ish, but could be much better…

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

- Many ML models are based on a notion of "distance" between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

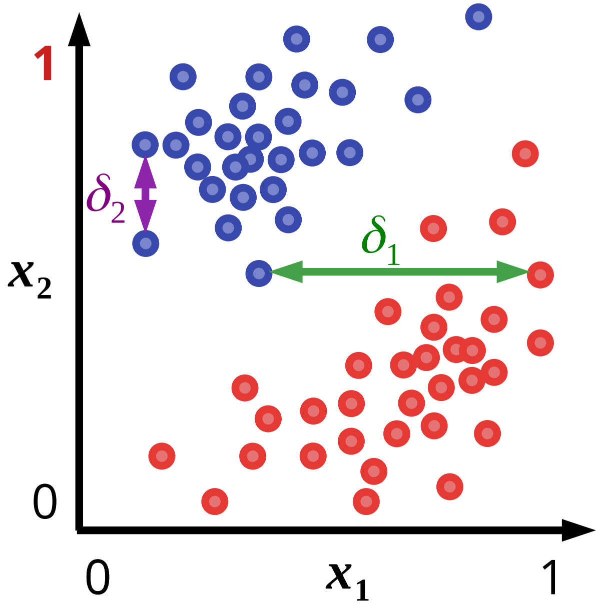
Euclidean distance metric

Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

- Many ML models are based on a notion of "distance" between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

Euclidean distance metric

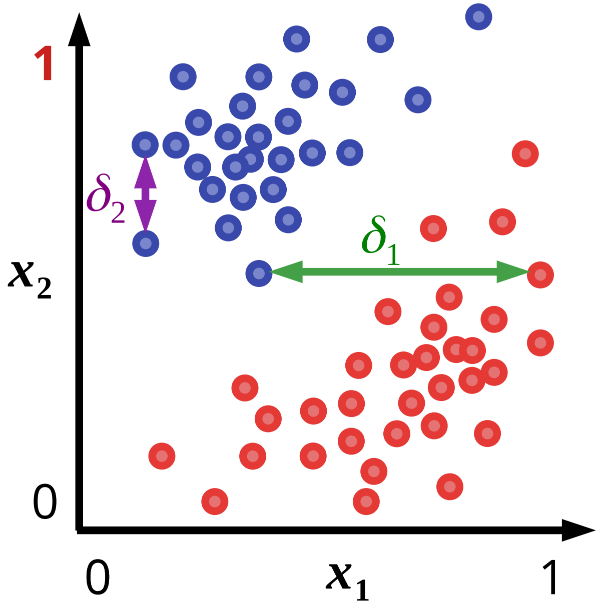
$\delta_1 > \delta_2$

Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

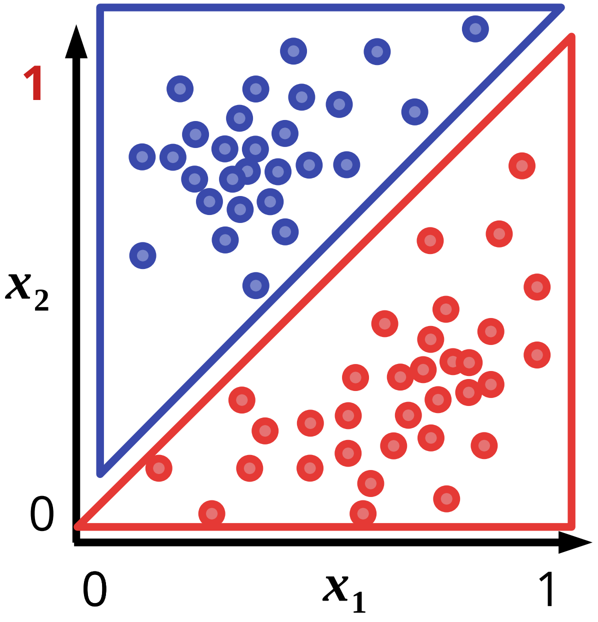
- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.



$$\delta = \sqrt{x_1^2 + x_2^2}$$

Euclidean distance metric

$\delta_1 > \delta_2$



Decision regions of a hypothetical distance-based classifier.

This is much better!

Data should be scaled!

Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

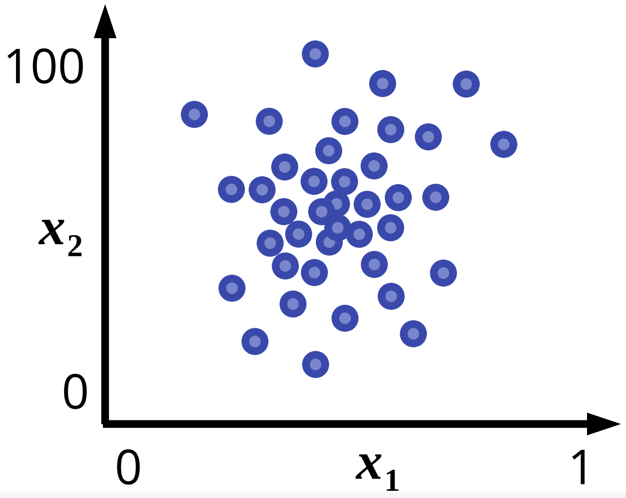
- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.
- Some ML models intrinsically presume that data are distributed following a Gaussian fashion with similar variances along all features; high variance along one feature leads to bias.

Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

- Many ML models are based on a notion of "distance" between samples; improperly scaled data may jeopardize the learning capability of such models.
- Some ML models intrinsically presume that data are distributed following a Gaussian fashion with similar variances along all features; high variance along one feature leads to bias.

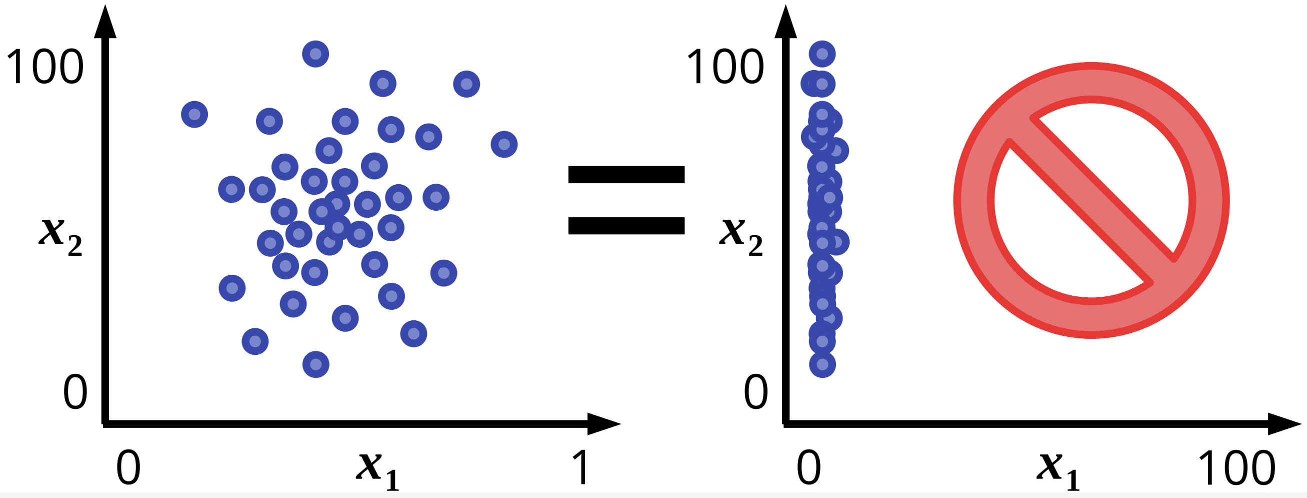


Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.
- Some ML models intrinsically presume that data are distributed following a Gaussian fashion with similar variances along all features; high variance along one feature leads to bias.

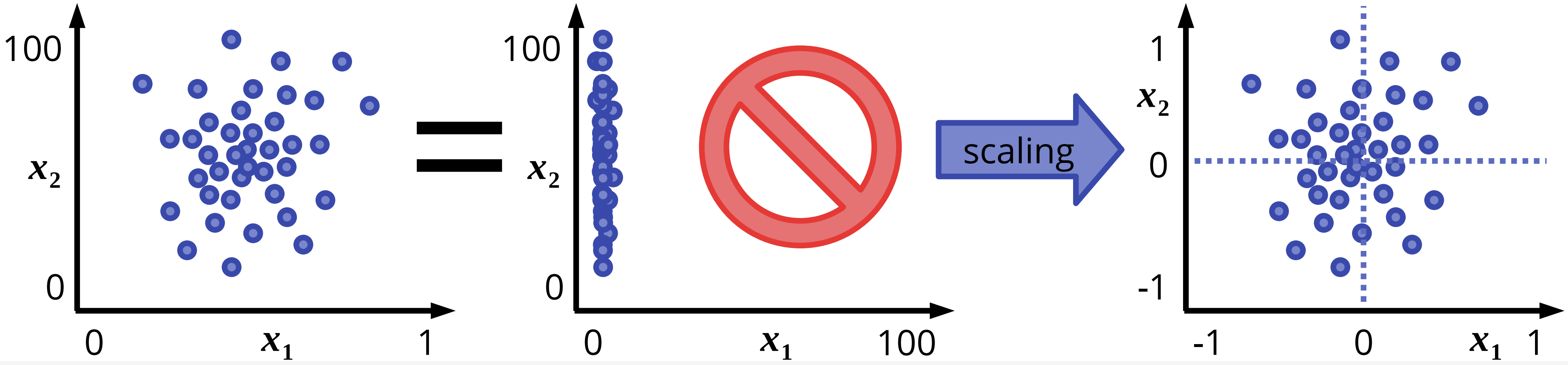


Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

- Many ML models are based on a notion of “distance” between samples; improperly scaled data may jeopardize the learning capability of such models.
- Some ML models intrinsically presume that data are distributed following a Gaussian fashion with similar variances along all features; high variance along one feature leads to bias.



Data scaling

Data scaling means to linearly transform your data in order to standardize them.

Why scale data?

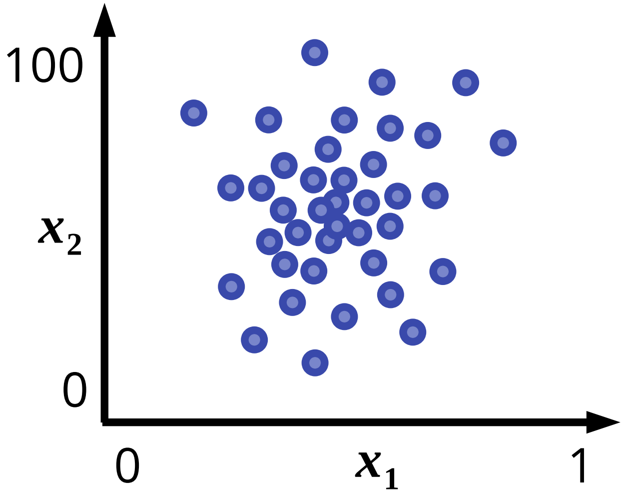
- Many ML models are based on a notion of "distance" between samples; improperly scaled data may jeopardize the learning capability of such models.
- Some ML models intrinsically presume that data are distributed following a Gaussian fashion with similar variances along all features; high variance along one feature leads to bias.

How to scale data?

- Normalize feature variances (to give similar weights to the different features)
- Normalize feature mean values (assumed by a number of ML models)

Data scaling – MinMax scaler

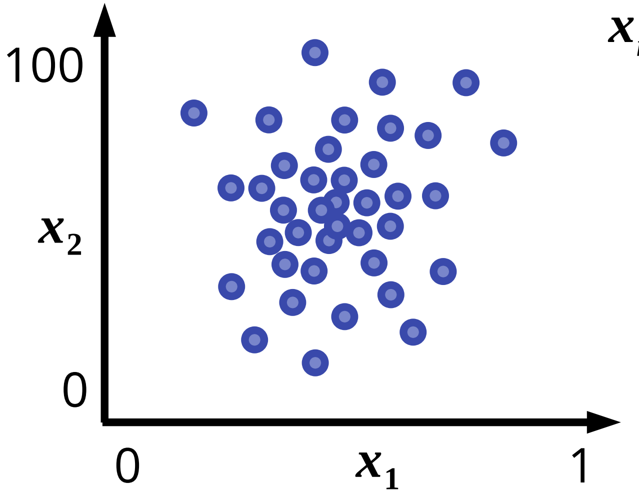
Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.



$x_i \in \boldsymbol{X}_i$

Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.

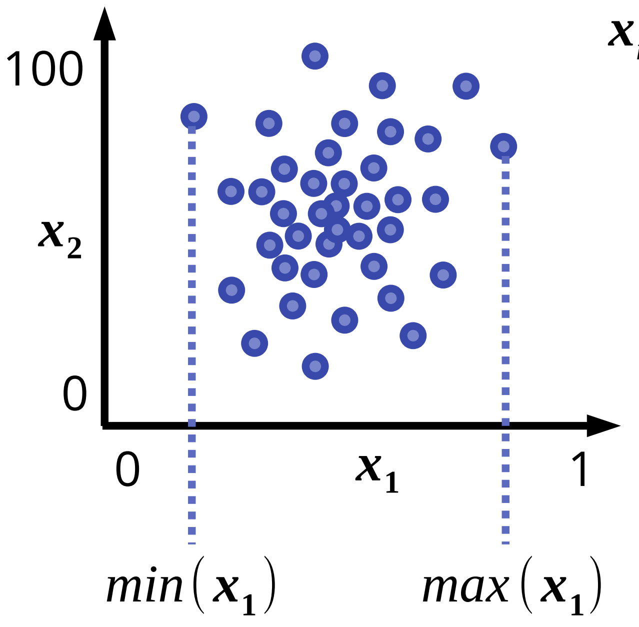


$$x_i' = \frac{x_i - min(X_i)}{max(X_i) - min(X_i)}$$

$$x_i \in X_i$$

Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.

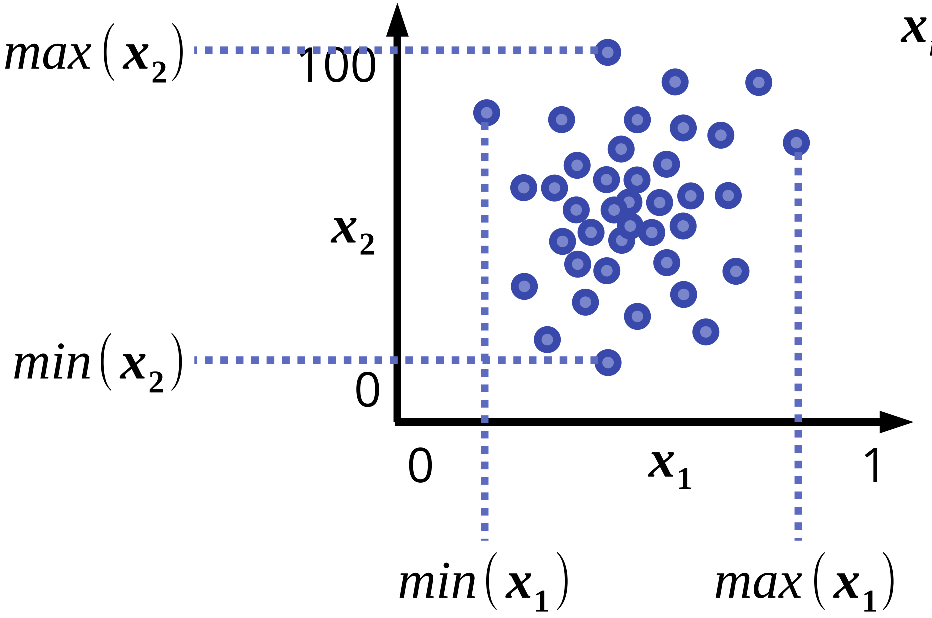


$$x_i' = \frac{x_i - min(X_i)}{max(X_i) - min(X_i)}$$

$$x_i \in X_i$$

Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.

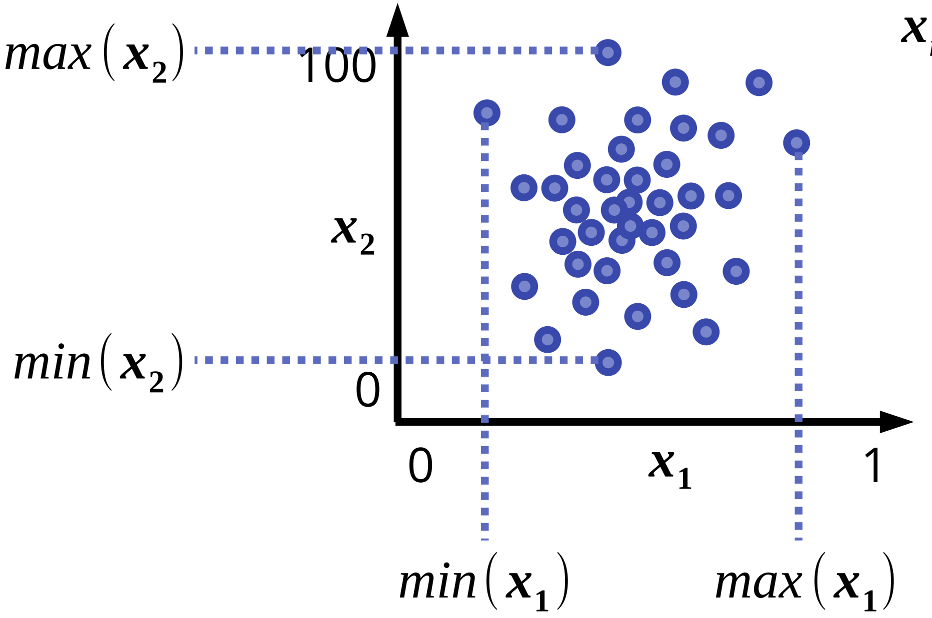


$$\boldsymbol{x}_i' = \frac{\boldsymbol{x}_i - min(\boldsymbol{X}_i)}{max(\boldsymbol{X}_i) - min(\boldsymbol{X}_i)}$$

$$x_i \in \boldsymbol{X}_i$$

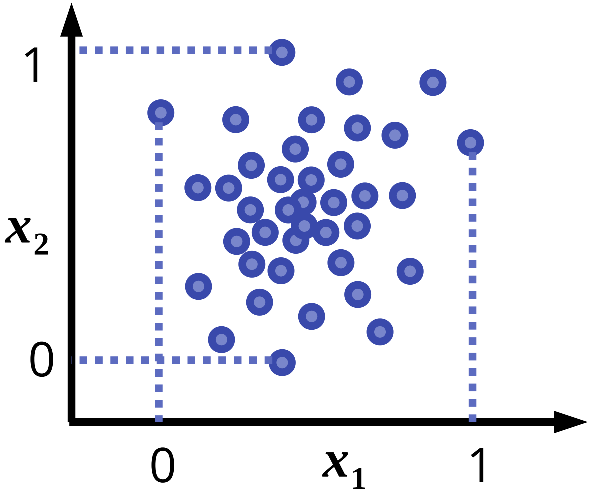
Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.



$$\boldsymbol{x}_i' = \frac{\boldsymbol{x}_i - min(\boldsymbol{X}_i)}{max(\boldsymbol{X}_i) - min(\boldsymbol{X}_i)}$$

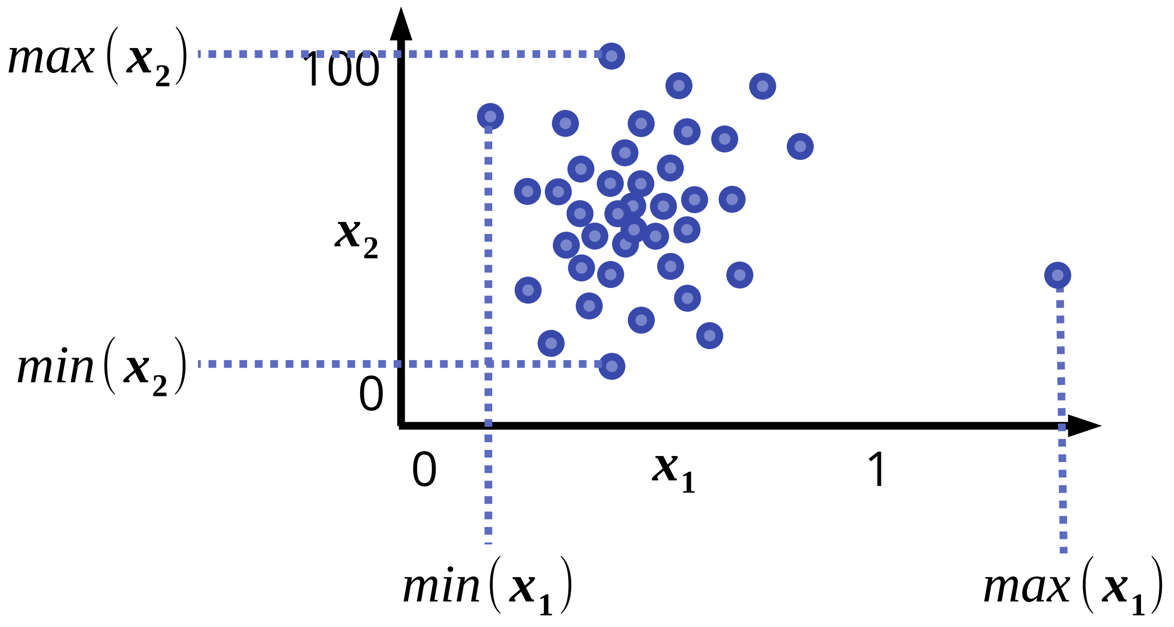
$$x_i \in \boldsymbol{X}_i$$



Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.

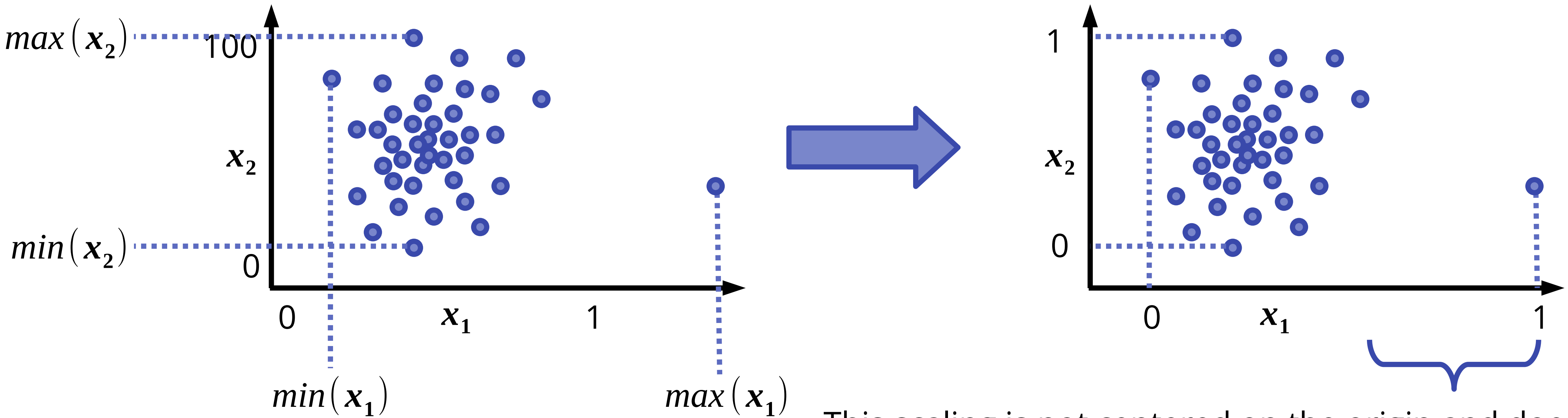
Disadvantage: the MinMax scaler is prone to outliers and does not center the distribution in the origin.



Data scaling – MinMax scaler

Scale every feature onto a range from 0 to 1 based on the minimum and maximum of the underlying distribution.

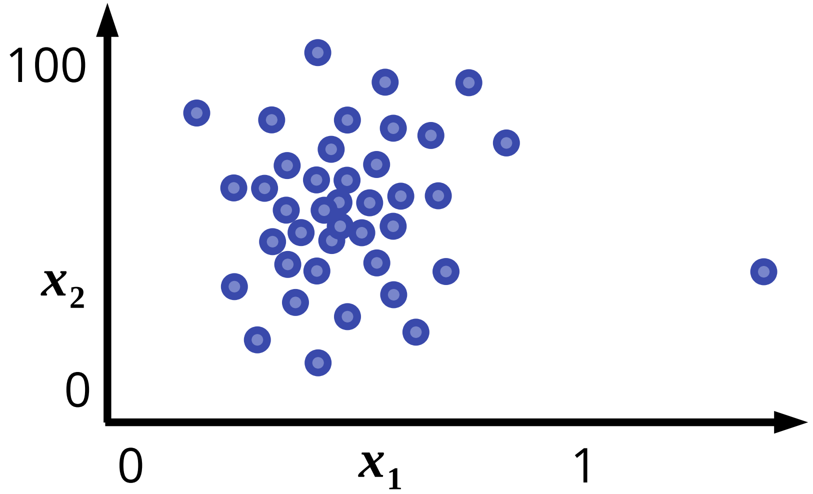
Disadvantage: the MinMax scaler is prone to outliers and does not center the distribution in the origin.



This scaling is not centered on the origin and does not describe the data distribution well.

Data scaling – Standard scaler

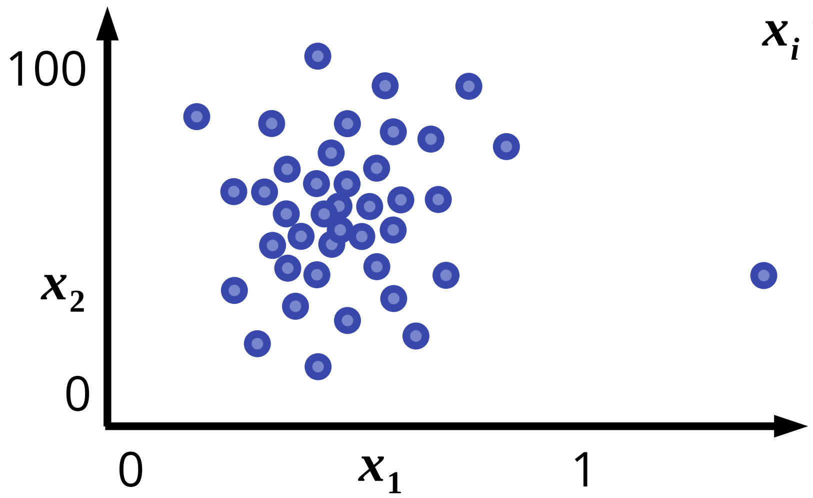
Scale every feature onto a range from -1 to 1 based on the mean and standard deviation of the underlying distribution.



$x_i \in \boldsymbol{X_i}$

Data scaling – Standard scaler

Scale every feature onto a range from -1 to 1 based on the mean and standard deviation of the underlying distribution.

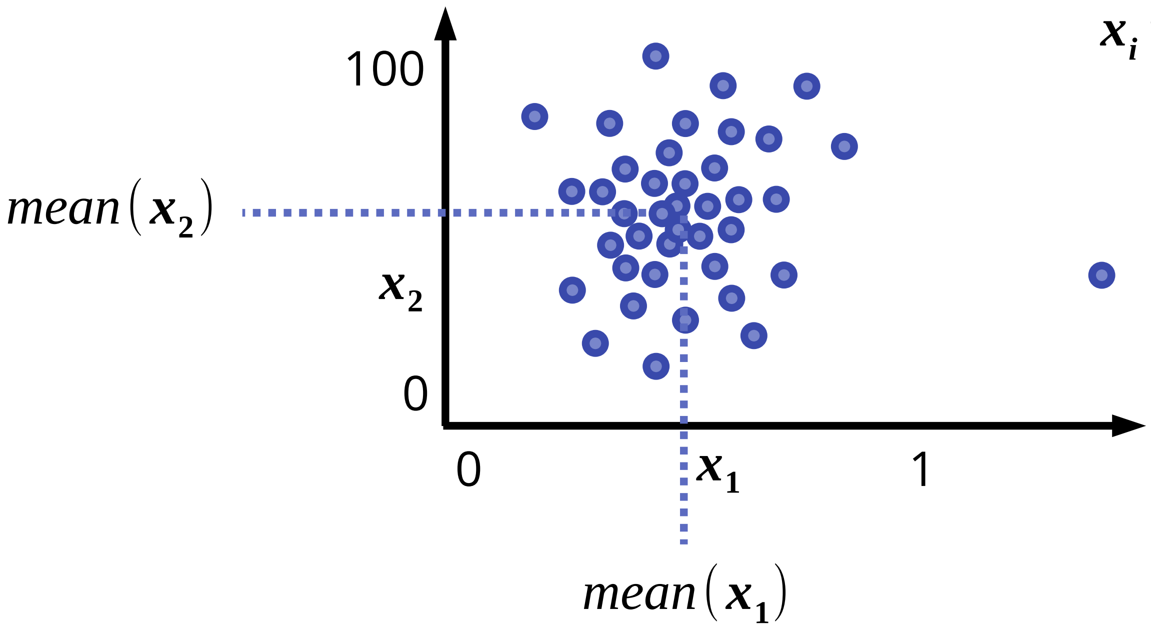


$$\boldsymbol{x_i}' = \frac{\boldsymbol{x_i} - mean(\boldsymbol{X_i})}{\sigma(\boldsymbol{X_i})}$$

$$x_i \in \boldsymbol{X_i}$$

Data scaling – Standard scaler

Scale every feature onto a range from -1 to 1 based on the mean and standard deviation of the underlying distribution.

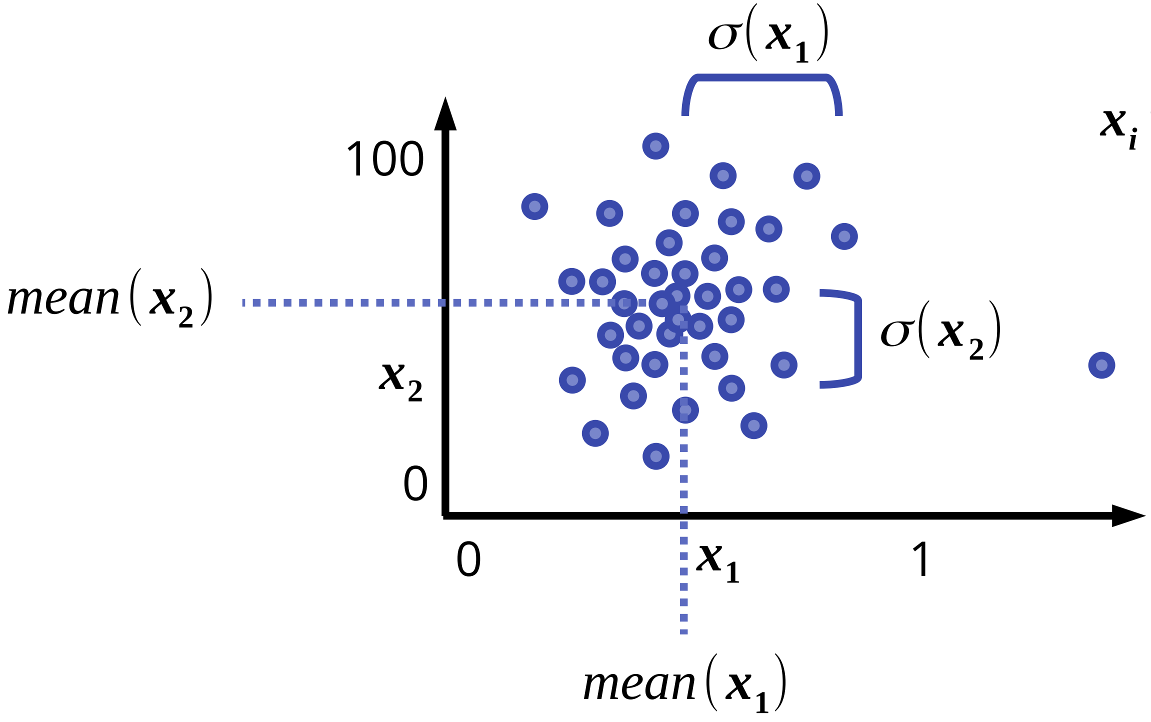


$$\boldsymbol{x}_i' = \frac{\boldsymbol{x}_i - mean(\boldsymbol{X}_i)}{\sigma(\boldsymbol{X}_i)}$$

$$x_i \in \boldsymbol{X}_i$$

Data scaling – Standard scaler

Scale every feature onto a range from -1 to 1 based on the mean and standard deviation of the underlying distribution.

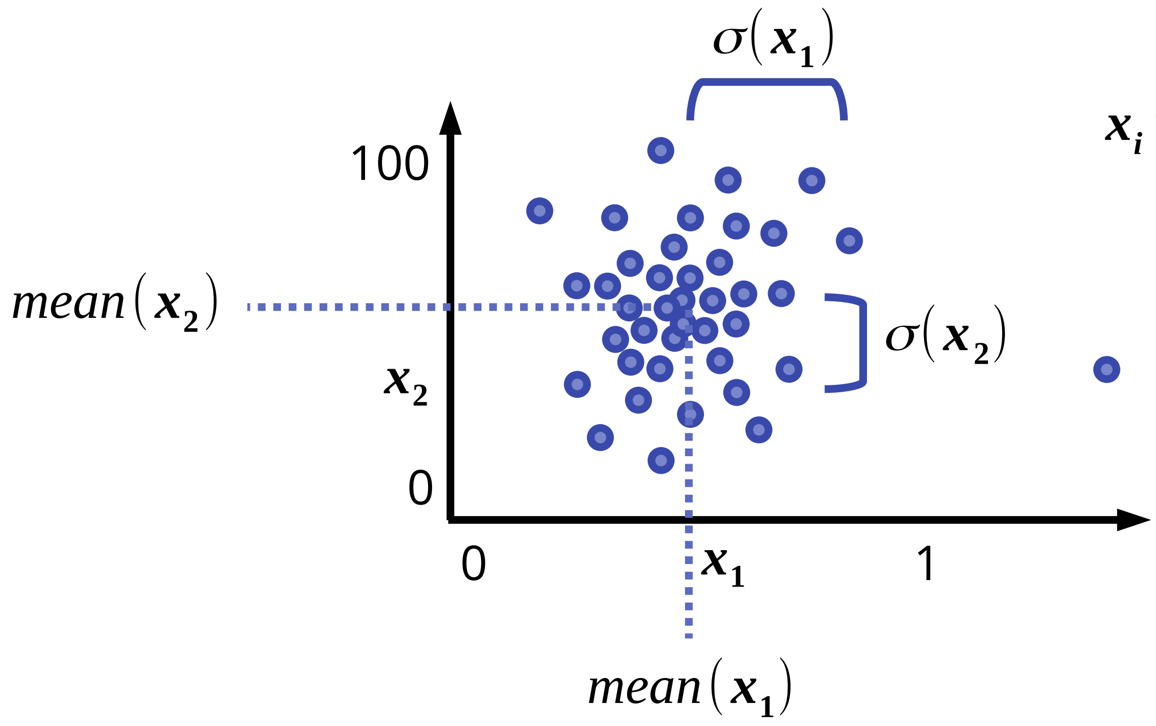


$$\boldsymbol{x}_i' = \frac{\boldsymbol{x}_i - mean(\boldsymbol{X}_i)}{\sigma(\boldsymbol{X}_i)}$$

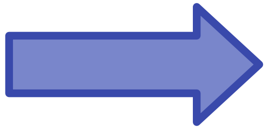
$$x_i \in \boldsymbol{X}_i$$

Data scaling – Standard scaler

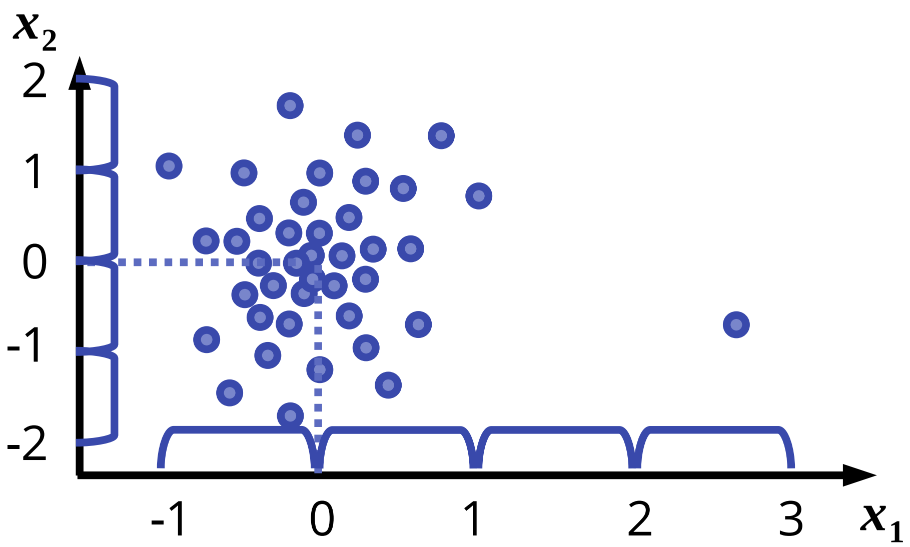
Scale every feature onto a range from -1 to 1 based on the mean and standard deviation of the underlying distribution.



$$\boldsymbol{x}_i{}' = \frac{\boldsymbol{x}_i - mean(\boldsymbol{X}_i)}{\sigma(\boldsymbol{X}_i)}$$



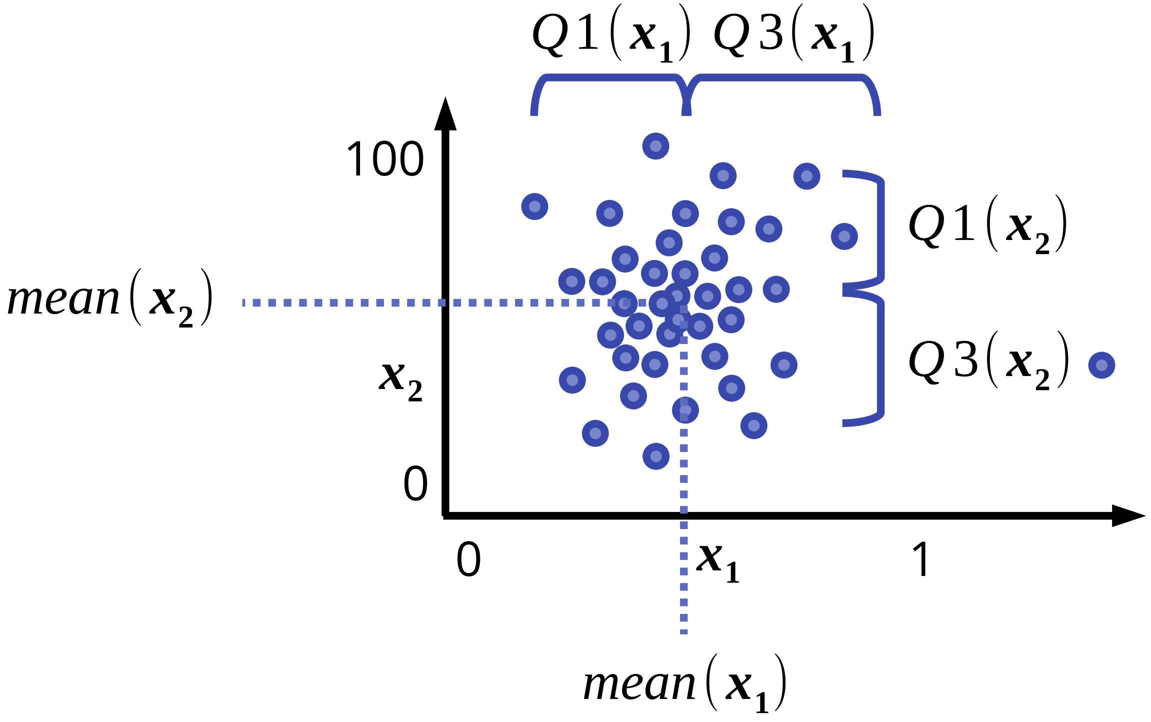
$$x_i \in \boldsymbol{X}_i$$



This scaling is centered on the origin but is still prone to outliers to some extent.

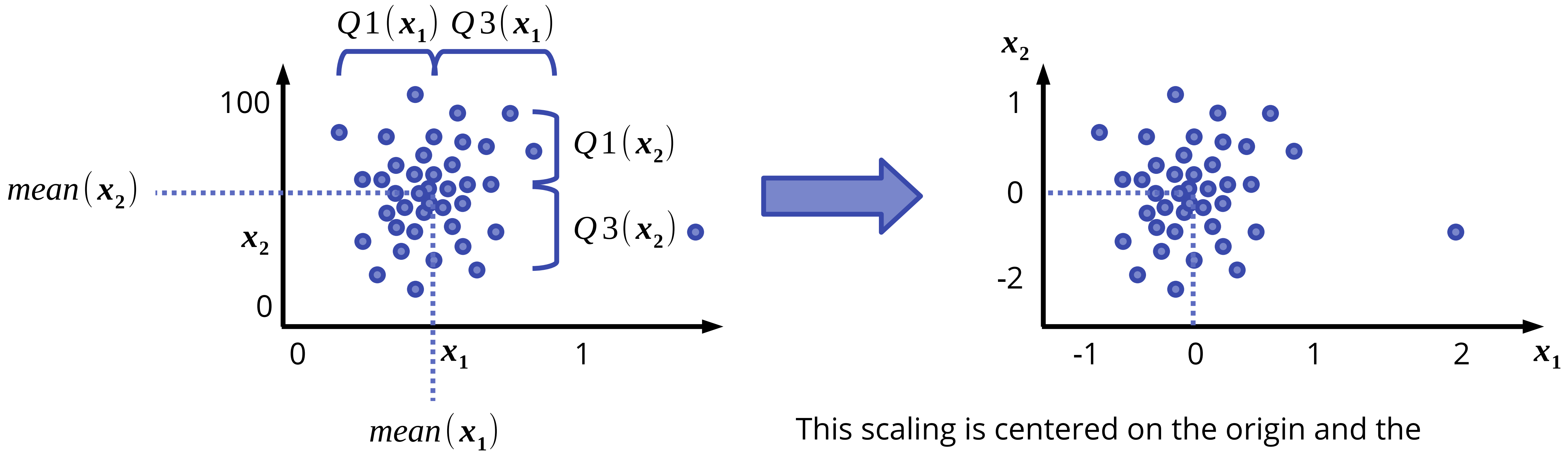
Data scaling – Robust scaler

Scale every feature onto a range from -1 to 1 based on the mean and the quantiles of the underlying distribution.



Data scaling – Robust scaler

Scale every feature onto a range from -1 to 1 based on the mean and the quantiles of the underlying distribution.



This scaling is centered on the origin and the resulting distribution is less affected by outliers

That’s all folks!