

Machine Learning for Time Series

(MLTS or MLTS-Deluxe Lectures)

Dr. Dario Zanca

Machine Learning and Data Analytics (MaD) Lab
Friedrich-Alexander-Universität Erlangen-Nürnberg
26.10.2023

Organisational Information

Machine Learning for time series

- 5 ECTS
- Lectures + Exercises

~~Machine Learning for Time Series (Deluxe)~~

- ~~7.5 ECTS~~
- ~~Lectures + Exercises + Project~~

- Time series fundamentals and definitions (2 lectures) ←
- Bayesian Inference (1 lecture)
- Gaussian processes (2 lectures)
- State space models (2 lectures)
- Autoregressive models (1 lecture)
- Data mining on time series (1 lecture)
- Deep learning on time series (4 lectures)
- Domain adaptation (1 lecture)

Lectures (online)

A new lecture recording is generally released every **Thursday** on FAU.TV

Consultation hours by appointment, write to dario.zanca@fau.de

Exercises (online)

Live Zoom Session starting on November 3rd

Recordings from previous editions are available at <https://www.fau.tv/course/id/3178>

StudOn 2023-2024:

<https://www.studon.fau.de/crs5276833.html>

Written Exam (5 ECTS)

- 70% from lectures, 30% from exercises
- **On-campus**

In this lecture...

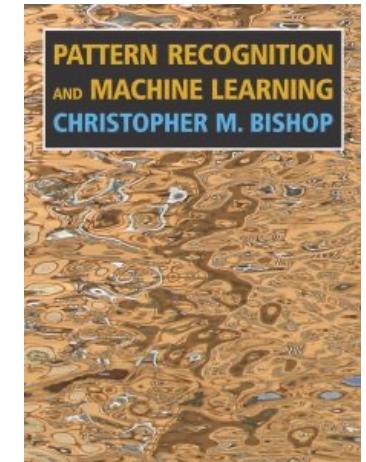
1. Types of Machine Learning
2. ML pipeline and good practise
3. ML tasks with time series
4. Example: linear regression

References

Pattern Recognition and Machine Learning

by Christopher Bishop

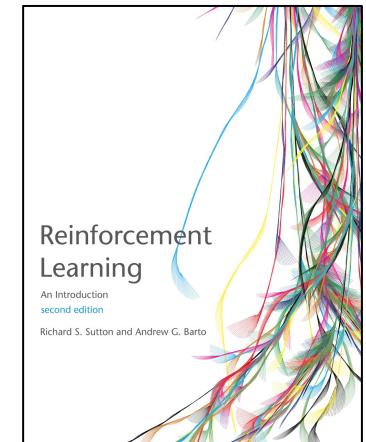
Available at: <https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/> (FREE)



Reinforcement Learning: An Introduction

by Richard S. Sutton and Andrew G. Barto

Available at: <http://incompleteideas.net/book/the-book.html>



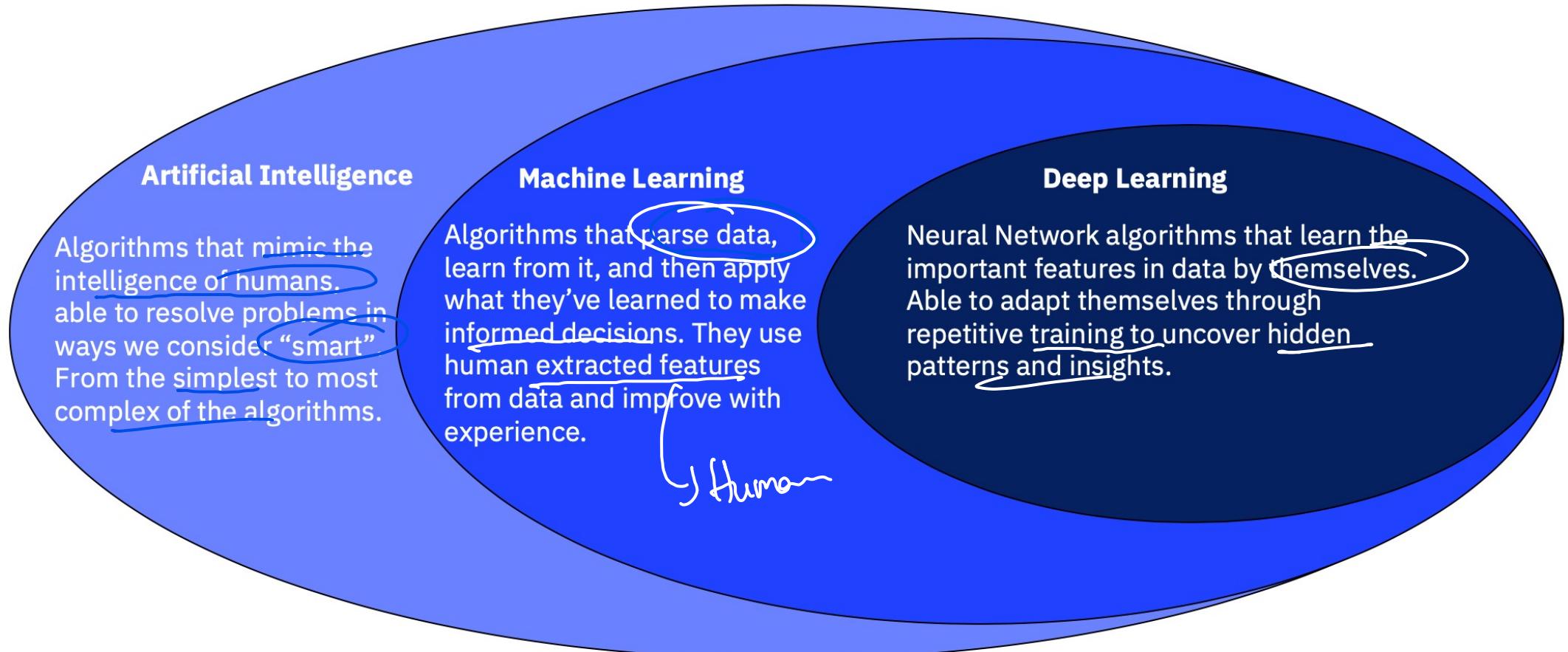


Machine learning basics

Types of machine learning



What is Machine Learning (ML)?



Supervised, unsupervised and reinforcement learning

Supervised Learning

- Learning using a teacher
- Makes machine learning explicitly
- Labeled data

Unsupervised Learning

- Learning using an “abstract” metric
- Machine understands the data (Identifies patterns/structures)
- Evaluation is qualitative or indirect

Reinforcement Learning

- Reward based learning
- Learning from positive and negative reinforcement
- Machine learns how to act in a certain environment to maximize rewards

Supervised, unsupervised and reinforcement learning

Supervised Learning

- Learning using a teacher
- Makes machine learning explicitly
- Labeled data

Unsupervised Learning

- Learning using an “abstract” metric
- Machine understands the data (Identifies patterns/structures)
- Evaluation is qualitative or indirect

Reinforcement Learning

- Reward based learning
- Learning from positive and negative reinforcement
- Machine learns how to act in a certain environment to maximize rewards

Supervised Learning

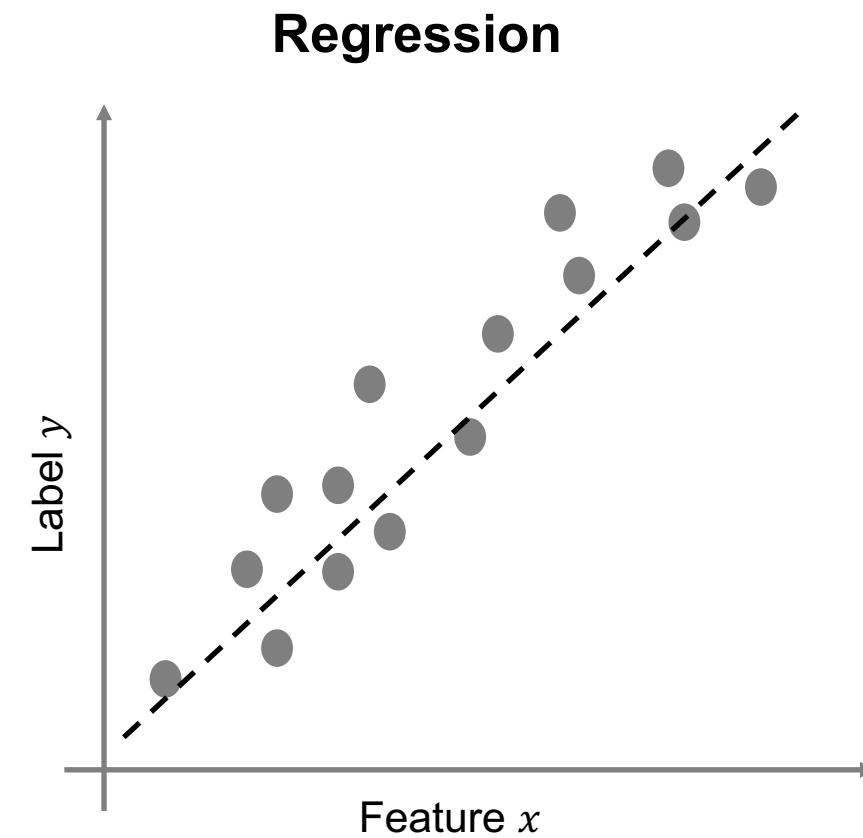
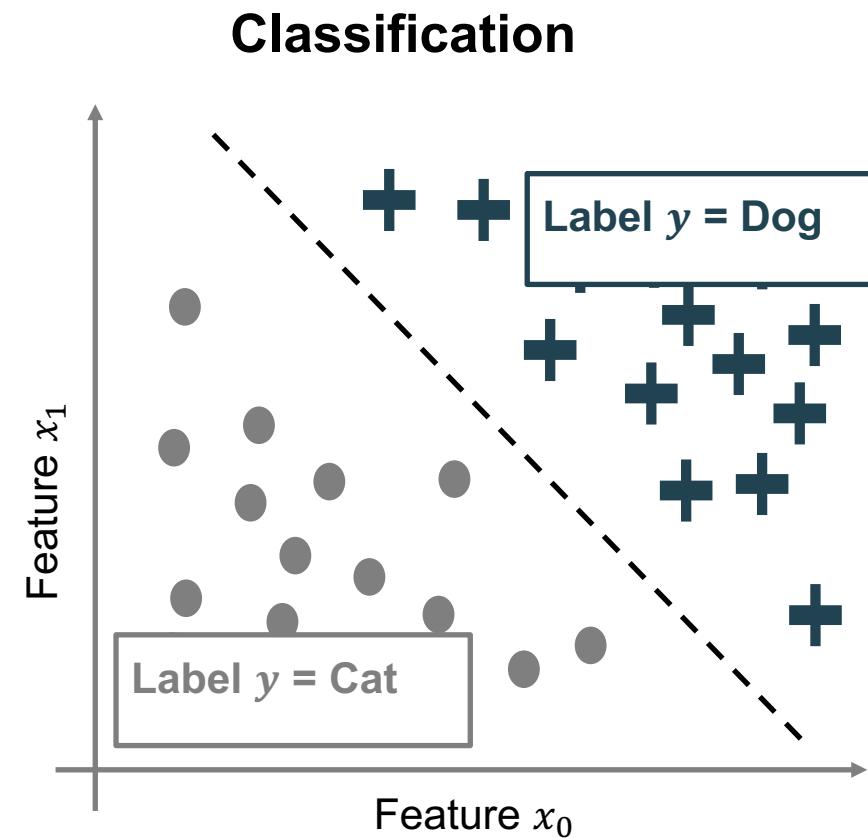
The agent observes some example **Input (Features)** and **Output (Label)** pairs and learns a function that maps input to output.

Key Aspects:

- Learning is explicit
- Learning using **direct feedback** ← direct feedback.
- Data with **labeled output**

→ Resolves classification and regression problems

Supervised Learning Problems



Regression

Regression is used to predict
a **continuous value**

Training is based on a set of
input – output pairs (**samples**)

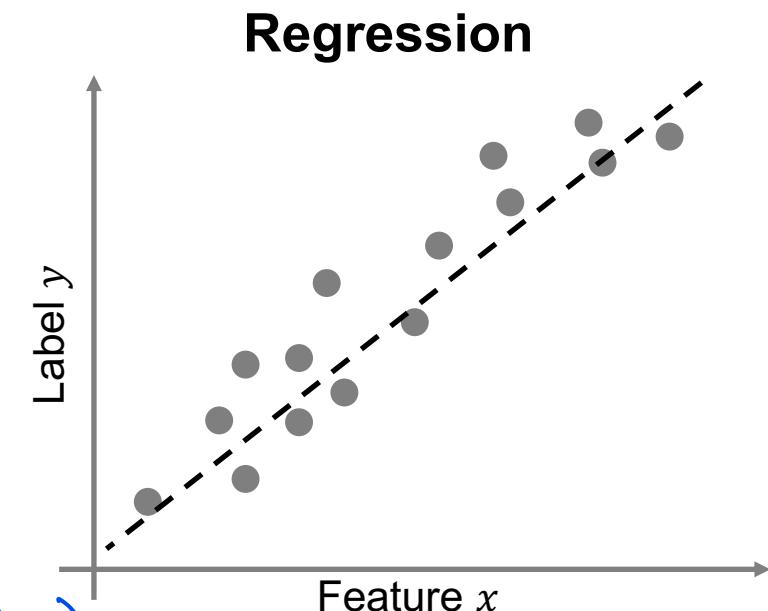
$$\mathcal{D} = \{(x_0, y_0), (x_1, y_1), \dots, (x_n, y_n)\}$$

Sample : (x_i, y_i)

$x_i \in \mathbb{R}^m$ is the **feature vector** of sample i

$y_i \in \mathbb{R}$ is the **label value** of sample i

(Feature vector,
Label value)



Regression

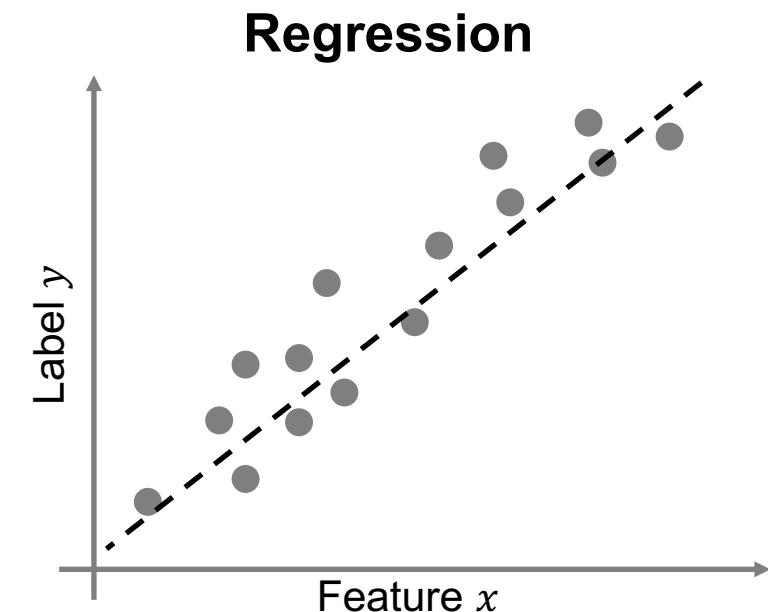
Goal:

Find a relationship (function), which expresses the input and output the best!

That means, we fit a regression model f to all samples:

$$f(x_i) = y_i \quad , \forall (x_i, y_i) \in \mathcal{D}$$

In this case f is a linear regression model!
(Black Line)



Classification

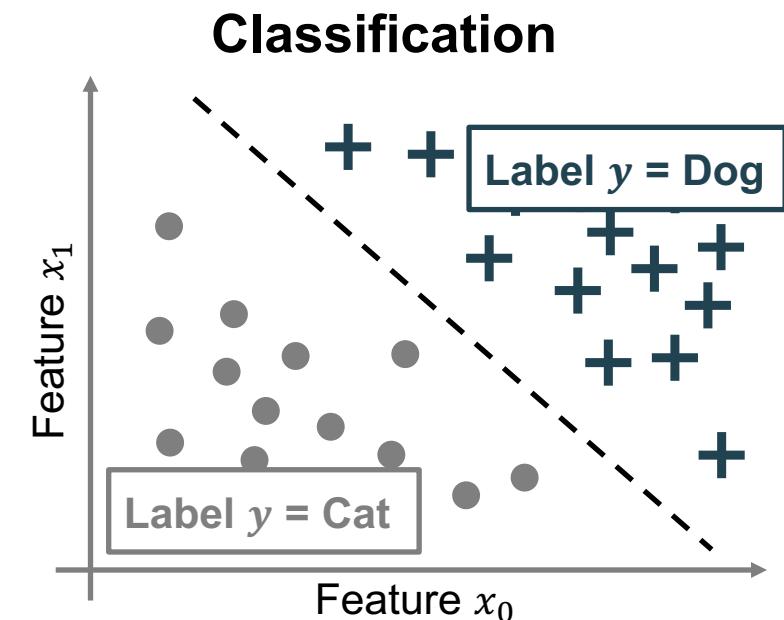
Classification is used to predict the class of the input

Sample : $s_i = (\vec{x}_i, \vec{y}_i)$

$\vec{y}_i \in L$ is the label of sample i

In this example:

- $L = \{Cat, Dog\}$
- Binary classification problem
- The output belongs to only one class!



Classification

Goal:

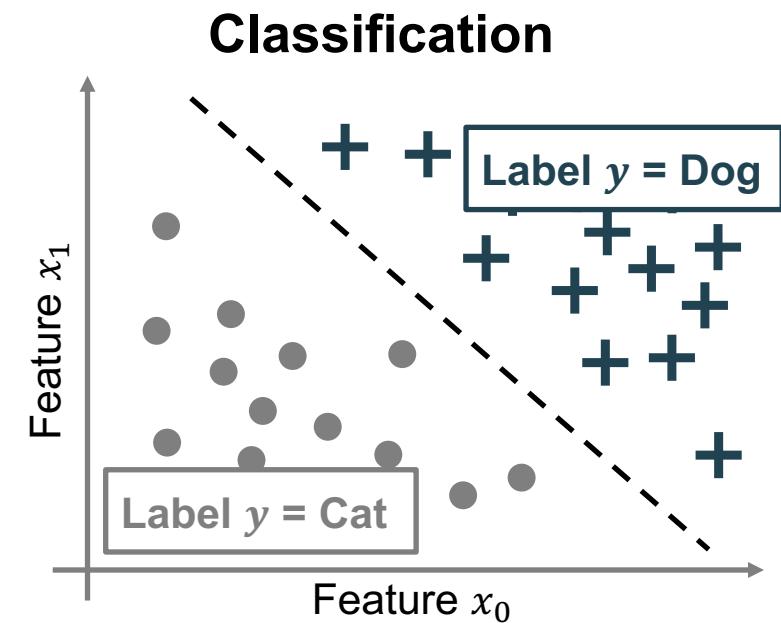
Find a way to divide the input into the output classes!

That means, we **find a decision boundary f** for all samples:

$$f(x_i) = y_i \quad , \forall (x_i, y_i) \in \mathcal{D}$$

linear decision boundary.

In this case f is a **linear decision boundary!**
(Black Line)



Supervised Learning Problems

fiting decision boundary.

Regression	Classification
<ul style="list-style-type: none">The output are <u>continuous or real values</u>We try to fit a <u>regression model</u>, which can predict <u>the output</u> more accuratelyRegression algorithms can be used to solve the <u>regression problems</u> such as Weather Prediction, House price prediction, Stock market prediction etc.	<ul style="list-style-type: none">The output variable must be a <u>discrete value (class)</u>We try to find a decision boundary, which can divide the dataset into different classes.Classification Algorithms can be used to solve <u>classification problems</u> such as Hand-written digits(MNIST), Identification of cancer cells, Defected or Undefected solar cells etc.

Supervised, unsupervised and reinforcement learning

Supervised Learning

- Learning using a teacher
- Makes machine learning explicitly
- Labeled data

Unsupervised Learning

- Learning using an “abstract” metric
- Machine understands the data (Identifies patterns/structures)
- Evaluation is qualitative or indirect

Indirect Evaluation.

Reinforcement Learning

- Reward based learning
- Learning from positive and negative reinforcement
- Machine learns how to act in a certain environment to maximize rewards

Unsupervised Learning

Unsupervised learning observes some example Input (Features) - No Labels - and finds patterns based on a metric

↳ Patterns based on metric (abstract)

Key Aspects:

- Learning is **implicit** *Implicit*
- Learning using **indirect feedback** - *Indirect fb*
- Methods are **self-organizing** *- Self org*

Resolves **clustering** and **dimensionality reduction** problems

Clustering

DR

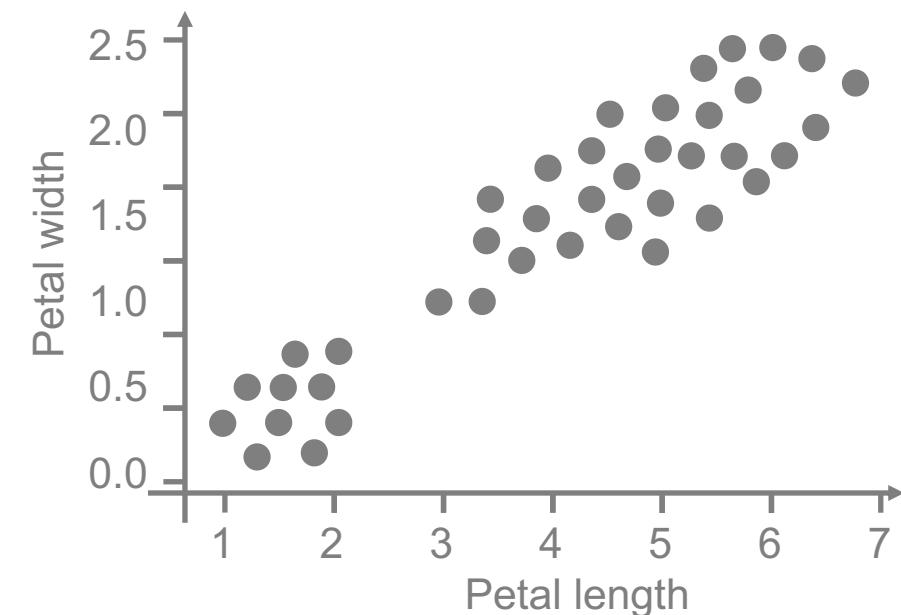
Clustering

Goal: Identify similar samples and assign them the same label

Mostly used for data analysis,
data exploration, and/or
data preprocessing

Clustering basic principles:

- Homogeneous data in the cluster
(Intra-cluster distance)
- Heterogenous data between the cluster
(Inter-cluster distance)



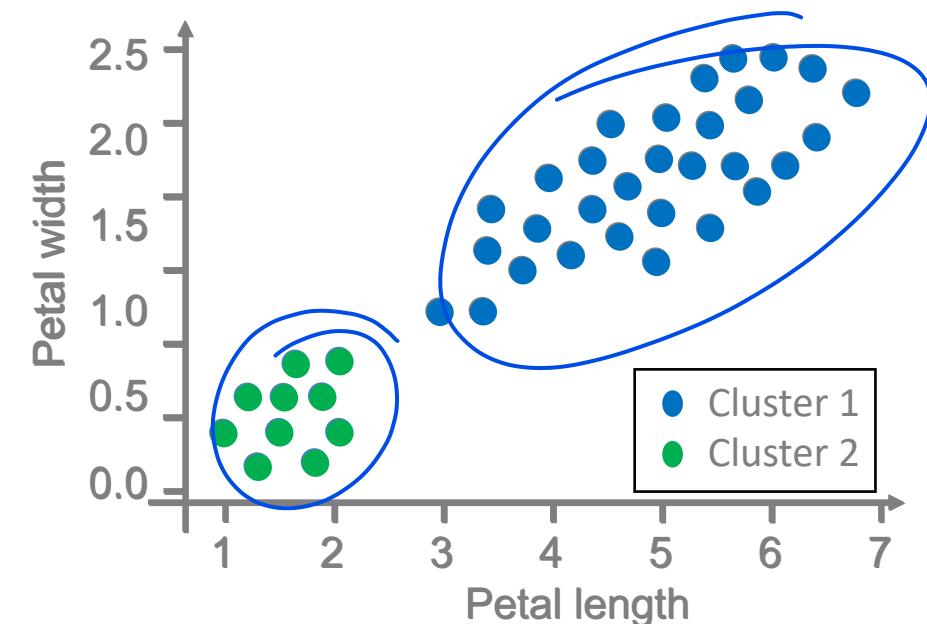
Clustering

Goal: Identify similar samples and assign them the same label

Mostly used for data analysis,
data exploration, and/or
data preprocessing

Clustering basic principles:

- Homogeneous data in the cluster
(Intra-cluster distance)
- Heterogenous data between the cluster
(Inter-cluster distance)



Curse of dimensionality

As the number of features or dimensions grows, the amount of data we need to generalize accurately grows exponentially.” – Charles Isbell

The intuition in lower dimensions does not hold in higher dimensions:

- Almost all samples are close to at least one boundary
- Distances (e.g., Euclidean) between all samples are similar
- Features might be wrongly correlated with outputs
- Finding decision boundaries becomes more complex

not true
in higher
dimensions.

→ much more features from
if samples is also
possible.

→ Problems become much more difficult to solve!

Example: Curse of dimensionality

The production system has N sensors attached with either the input set to “On” or “Off”

Question: How many samples do we need, to have all possible sensor states in the dataset?

$$N = 1 : |D| = 2^1 = 2$$

$$N = 10 : |D| = 2^{10} = 1024$$

$$N = 100 : |D| = 2^{100} = 1.2 \times 10^{30}$$

For $N = 100$, the number of points are even more than the number of atoms in the universe!

↑ 100 Sensors on/off

Dimensionality reduction

The goal:

Transform the samples from a high to a lower dimensional representation!

Ideally:

Find a representation, which solves your problem!

Typical Approaches:

- Feature Selection
- Feature Extraction

	S0	S1	S2	S3	S4	S5	S6	S7	S8
Sample0	0.2	0.1	11.1	2.2	Off	7	1.1	0	1.e-1
Sample1	1.2	-0.1	3.1	-0.1	On	9	2.3	-1	1.e-4
Sample2	2.7	1.1	0.1	0.1	Off	10	4.5	-1	1.e-9
Sample3	3.1	0.1	1.1	0.2	Off	1	6.6	-1	1.e-1

	T0	T1	T2	T3
Sample0	11.3	0.1	-1	7.8
Sample1	4.3	-0.1	1	6.8
Sample2	2.8	1.1	-1	7.1
Sample3	4.2	0.1	1	6.9

Dimensionality reduction

The goal:

Transform the samples from a high to a lower dimensional representation!

Ideally:

Find a representation, which solves your problem!

Typical Approaches:

- Feature Selection
- Feature Extraction

select some & discard the rest.

	S0	S1	S2	S3	S4	S5	S6	S7	S8
Sample0	0.2	0.1	11.1	2.2	Off	7	1.1	0	1.e-1
Sample1	1.2	-0.1	3.1	-0.1	On	9	2.3	-1	1.e-4
Sample2	2.7	1.1	0.1	0.1	Off	10	4.5	-1	1.e-9
Sample3	3.1	0.1	1.1	0.2	Off	1	6.6	-1	1.e-1
⋮									

Identical

	T0	T1	T2	T3
Sample0	11.3	0.1	-1	7.8
Sample1	4.3	-0.1	1	6.8
Sample2	2.8	1.1	-1	7.1
Sample3	4.2	0.1	1	6.9
⋮				

Dimensionality reduction

The goal:

Transform the samples from a high to a lower dimensional representation!

Ideally:

Find a representation, which solves your problem!

Typical Approaches:

- Feature Selection
- Feature Extraction

transform
or combine
features into single one.

	S0	S1	S2	S3	S4	S5	S6	S7	S8
Sample0	0.2	0.1	11.1	2.2	Off	7	1.1	0	1.e-1
Sample1	1.2	-0.1	3.1	-0.1	On	9	2.3	-1	1.e-4
Sample2	2.7	1.1	0.1	0.1	Off	10	4.5	-1	1.e-9
Sample3	3.1	0.1	1.1	0.2	Off	1	6.6	-1	1.e-1
⋮									

Applied a function f

	T0	T1	T2	T3
Sample0	11.3	0.1	-1	7.8
Sample1	4.3	-0.1	1	6.8
Sample2	2.8	1.1	-1	7.1
Sample3	4.2	0.1	1	6.9
⋮				

Supervised, unsupervised and reinforcement learning

Supervised Learning

- Learning using a teacher
- Makes machine learning explicitly
- Labeled data

Unsupervised Learning

- Learning using an “abstract” metric
- Machine understands the data (Identifies patterns/structures)
- Evaluation is qualitative or indirect

Abstract metric.

Reinforcement Learning

- Reward based learning
- Learning from positive and negative reinforcement
- Machine learns how to act in a certain environment to maximize rewards

Reinforcement Learning

Reinforcement learning observes some example Input (Features) – No Labels! - and finds the **optimal action** i.e., maximizes its future reward

Action → max. rewards.

Key Aspects:

- Learning is implicit *Implicit*
- Learning using indirect feedback based on trials and reward signals
- Actions are affecting future measurements (i.e., inputs)

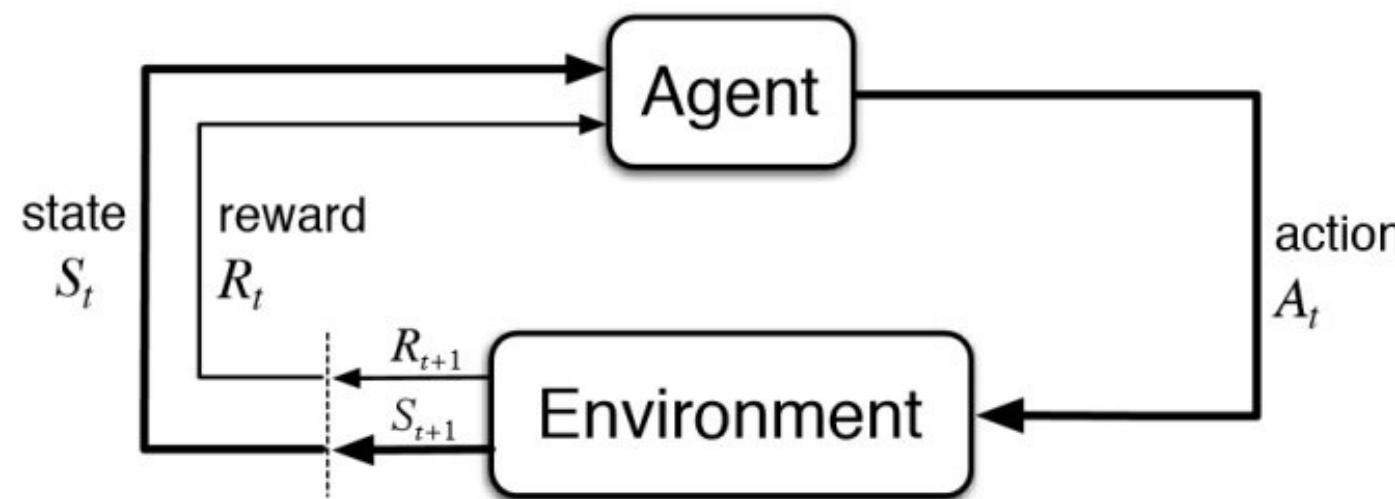
Resolves **control** and **decision** problems → *Control / decisions*

- i.e., controlling agents in games or robots

Reinforcement Learning

Goal: Agents should take actions in an environment which maximize the cumulative reward.

To achieve this RL uses reward and punishment signals based on the previous actions to optimize the model.



Reinforcement Learning vs Unsupervised Learning

Unsupervised Learning	Reinforcement Learning
<ul style="list-style-type: none">An indirect feedback is generated according to a metricFeedback is instantaneousLearning by using static data (no re-recording of data necessary)Prediction does not affect future measurements – The data is assumed Independent Identically Distributed (i.i.d)	<ul style="list-style-type: none">The feedback is given by a reward signalFeedback can be delayed (credit assignment problem)Training is based on trials i.e. interaction between environment and agent (re-recording necessary)The prediction (actions) affect future measurements i.e. the measurements are no necessarily i.i.d!
<i>metric</i>	<i>reward signal</i>
<i>No future impact.</i>	<i>delayed f/b.</i> <i>Not iid.</i>



Machine learning basics

ML pipeline and good practices



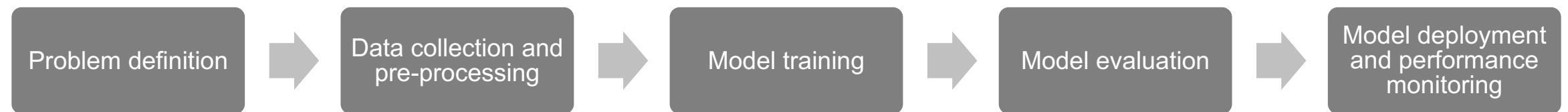
The ML pipeline

The concept of a pipeline guidance in a machine learning project.

- step-by-step process
- Each step has a **goal** and a **method**

There exist many pipelines proposals in the literature.

Here we propose a compact 5-steps pipeline:



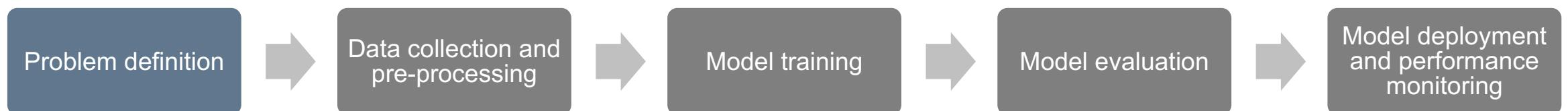
Step 1. Problem definition

In order to develop a satisfying solution, we need to define the problem.

- What goal (or **task**) we want to solve → Goal / task
- What kind of **data** we need → Data.

E.g., our goal is to monitor an industrial machine to predict failure and allow convenient scheduling of corrective maintenance.

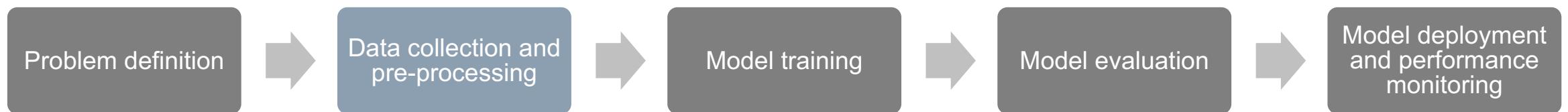
- Our goal is to predict failure one week in advance
 - Alternatively, predict the remaining useful life
- Prediction is based on data from the machine (sensors) and users (logs)



Step 2. Data collection and pre-processing

The phase of gathering the data and creating our dataset is called **data ingestion**.

- Data should **contain necessary information** to solve the task
- Data should **be enough** to describe all possible states



Step 2. Data collection and pre-processing

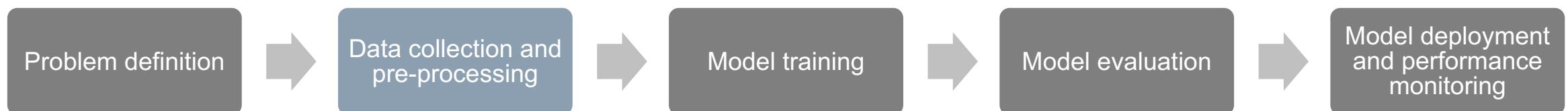
The phase of gathering the data and creating our dataset is called **data ingestion**.

- Data should contain necessary information to solve the task → necessary info.
- Data should be enough to describe all possible states – enough to describe all states.

Then, a **data preparation** phase follows, with the goal of making data usable for our machine learning solution.

- Remove missing values and outliers, apply dimensionality reduction, normalize, rescale, ...

preprocessing steps

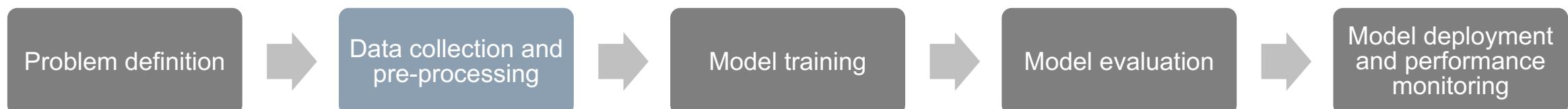


Step 2. Data collection and pre-processing

Finally, and before training any machine learning algorithm, we perform **data segregation**.

- We separate the target value from the input features
- We split the data collection into
 - Training set: used to fit our model parameters
 - Validation set: used to have an unbiased estimation of the model generalization capabilities during training, and tune hyperparameters of our model
 - Test set: used to evaluate performance of a final version of the model

Ingestion, preparation, segregation

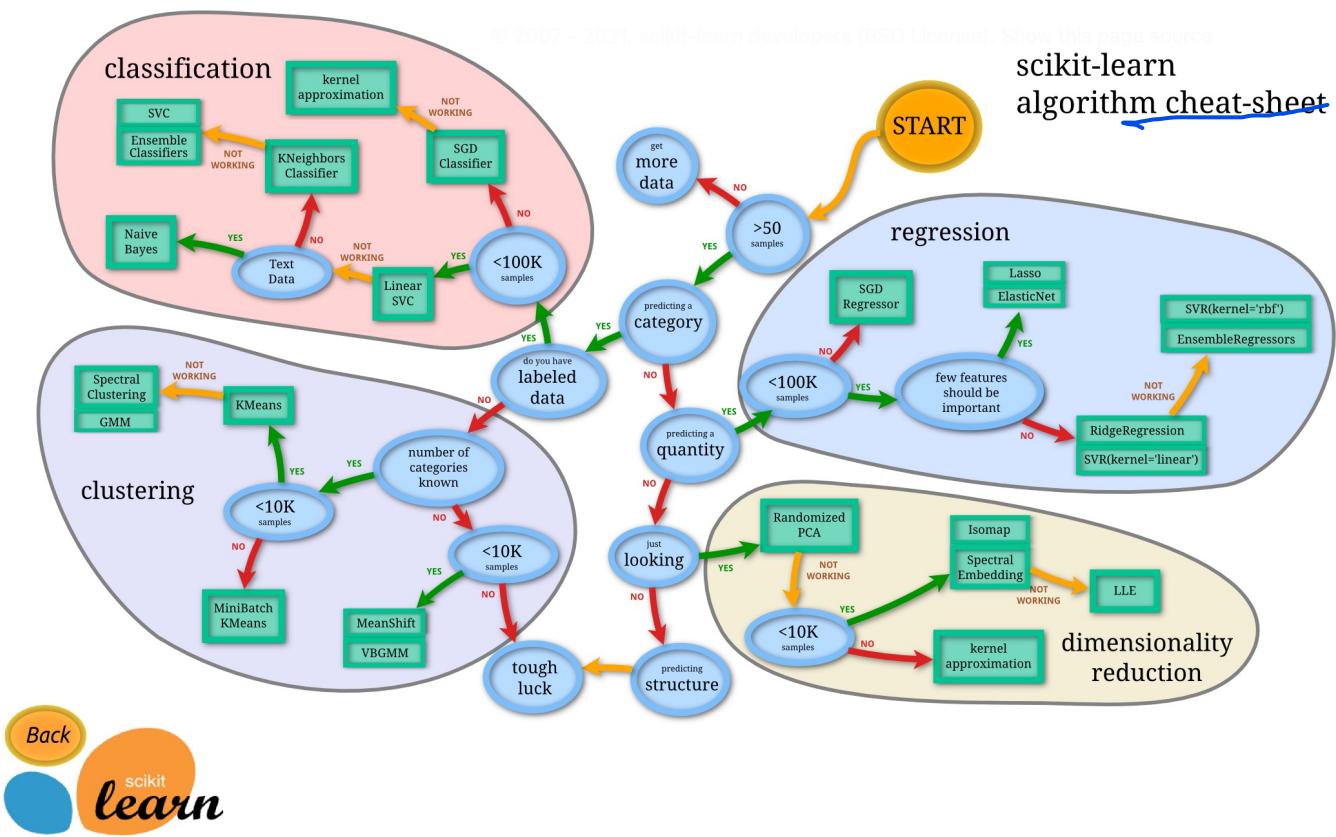


Step 3. Model training

We need to select an algorithm to be trained on our data.

- E.g., the prediction of a machine failure can be defined as a classification or a regression problem

To find out which algorithm is the best for our data set we have to test them.



https://scikit-learn.org/stable/tutorial/machine_learning_map/index.html

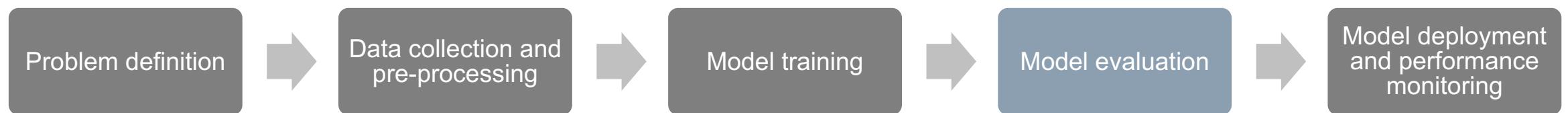


Step 4. Model evaluation

Training and evaluation of the model are iterative processes:

- First, we train our model with the training set
- Then evaluate its performance with validation set with evaluation metrics
- Based on this information, we tune our algorithm's hyperparameters

This iterative process continues unless we decide that we can't improve our algorithm anymore.

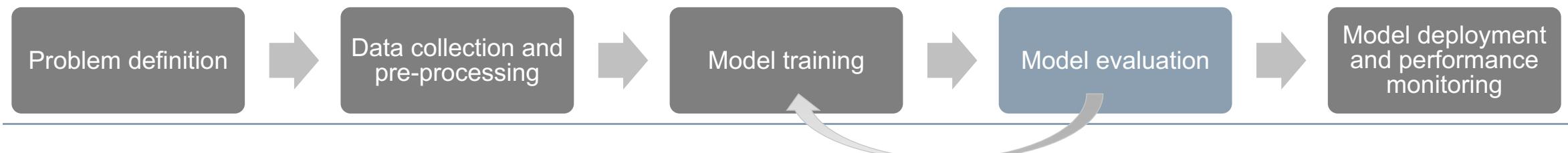


Step 4. Model evaluation

Training and evaluation of the model are iterative processes:

- First, we train our model with the training set
- Then evaluate its performance with validation set with evaluation metrics
- Based on this information, we **tune our algorithm's hyperparameters**

This iterative process continues unless we decide that we can't improve our algorithm anymore.

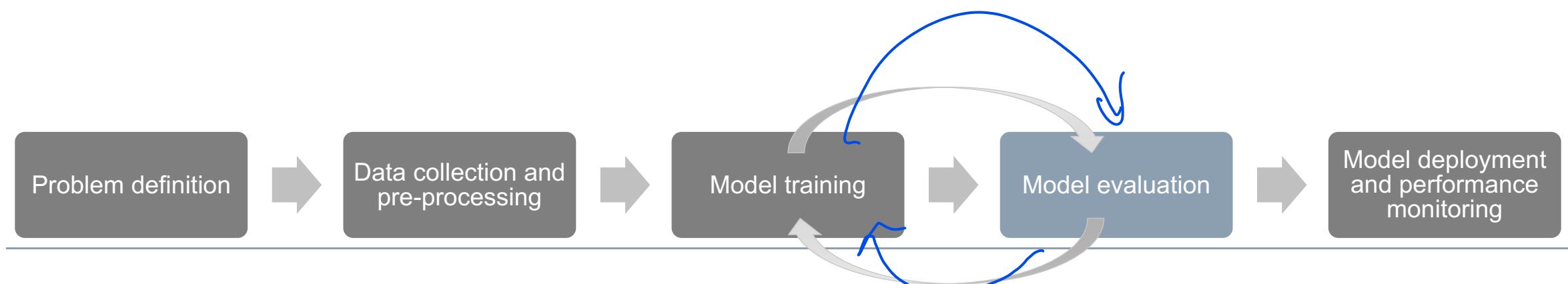


Step 4. Model evaluation

Training and evaluation of the model are iterative processes:

- First, we train our model with the training set
- Then evaluate its performance with validation set with evaluation metrics
- Based on this information, we **tune our algorithm's hyperparameters**

This iterative process continues unless we decide that we can't improve our algorithm anymore.



Step 4. Model evaluation

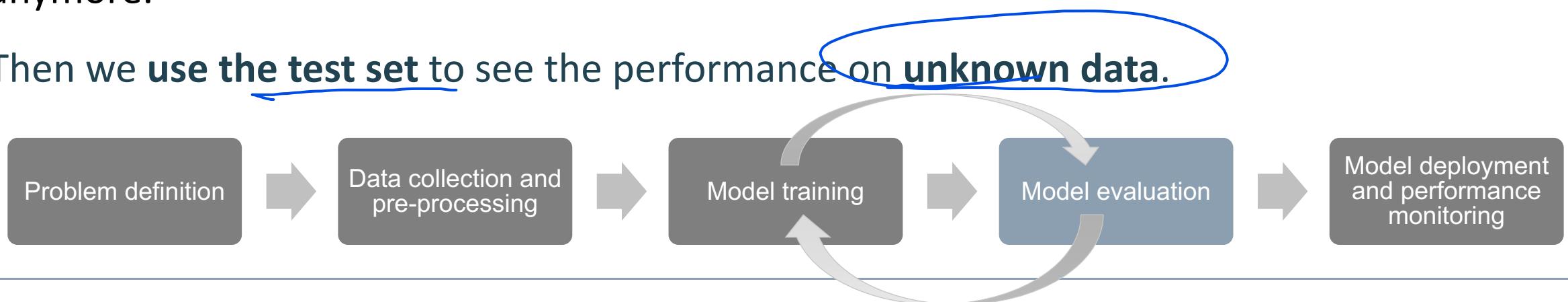
Training and evaluation of the model are iterative processes:

- First, we train our model with the training set
- Then evaluate its performance with validation set with evaluation metrics
- Based on this information, we **tune our algorithm's hyperparameters**

validation
↓
Hyper Parameters
tuning.

This iterative process continues unless we decide that we can't improve our algorithm anymore.

Then we **use the test set** to see the performance on **unknown data**.



Classification models evaluation – the confusion metrix

Confusion Matrix describes the performance of the model.

There are 4 important terms:

True Positives: The cases in which we predicted YES, and the actual output was also YES

True Negatives: The cases in which we predicted NO, and the actual output was NO

False Positives: The cases in which we predicted YES, and the actual output was NO

False Negatives: The cases in which we predicted NO, and the actual output was YES

Confusion Matrix

n=165	Predicted:NO	Predicted:YES
Actual: NO	50	10
Actual: YES	5	100

Classification Accuracy is what we usually mean, when we use the term accuracy. It is the ratio of number of correct predictions to the total number of input samples.

$$\text{Accuracy} = (150/165) = 0.909$$

Regression models evaluation – metrics

Mean Absolute Error (MAE):

- Average difference between the original and predicted values
- Measure how far predictions were from the actual output
- Does not give an idea about the direction of error.

↓
original ↑
predicted.

$$\text{Mean Absolute Error} = \frac{1}{N} \sum_{j=1}^N |y_j - \hat{y}_j|$$

↳ Where is the Error going?

Mean Squared Error (MSE):

- Similar to MAE
- It takes the average of the square of the difference between original and predicted value
- Larger errors become more pronounced, so that the model can focus on larger errors.

↳ Bigger Error more penalty?

$$\text{Mean Squared Error} = \frac{1}{N} \sum_{j=1}^N (y_j - \hat{y}_j)^2$$

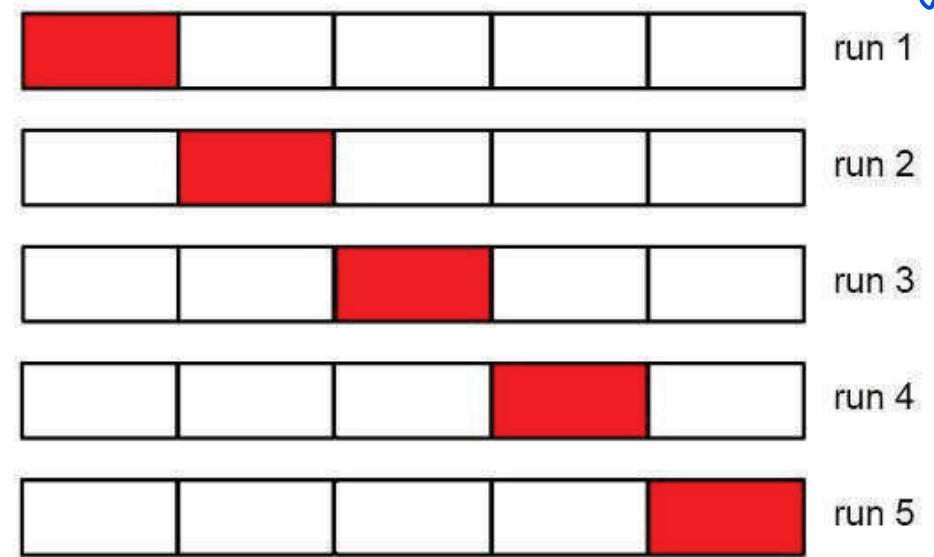
Cross validation

Cross-validation is a resampling method to iteratively use different portions of the dataset to train and validate a model among iterations.

- Particularly useful when the dataset size is small
- It can be also used for hyper-parameters selection

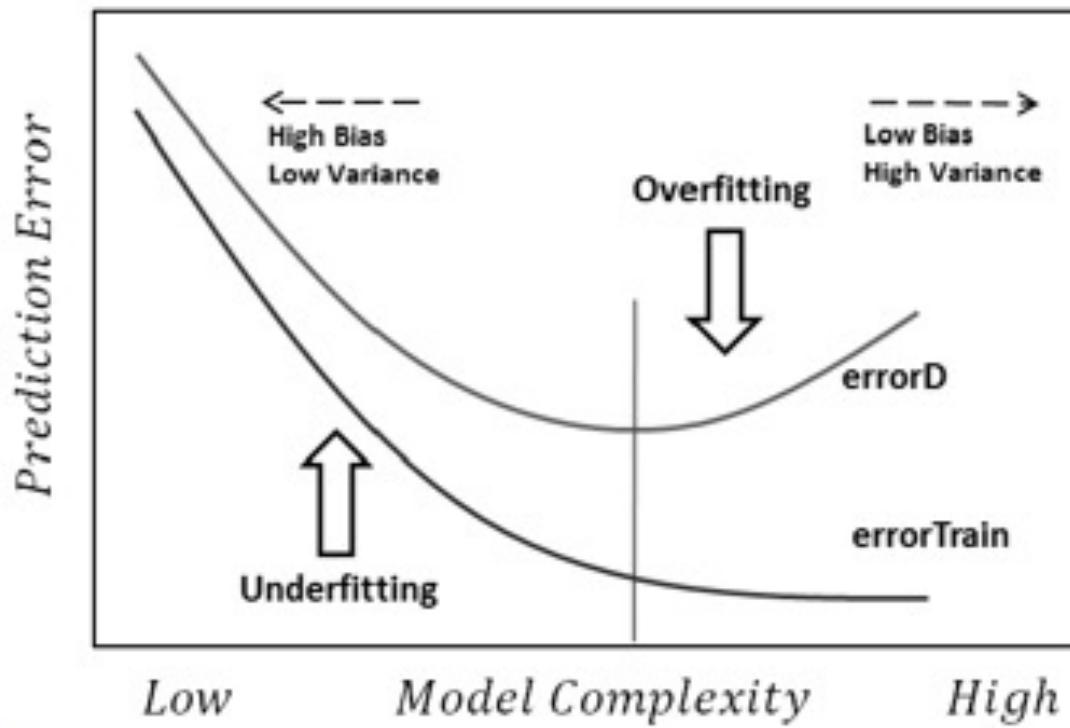
Use different set
for validation

If lucky? can lead to better results.
generic? - Go for cross validation

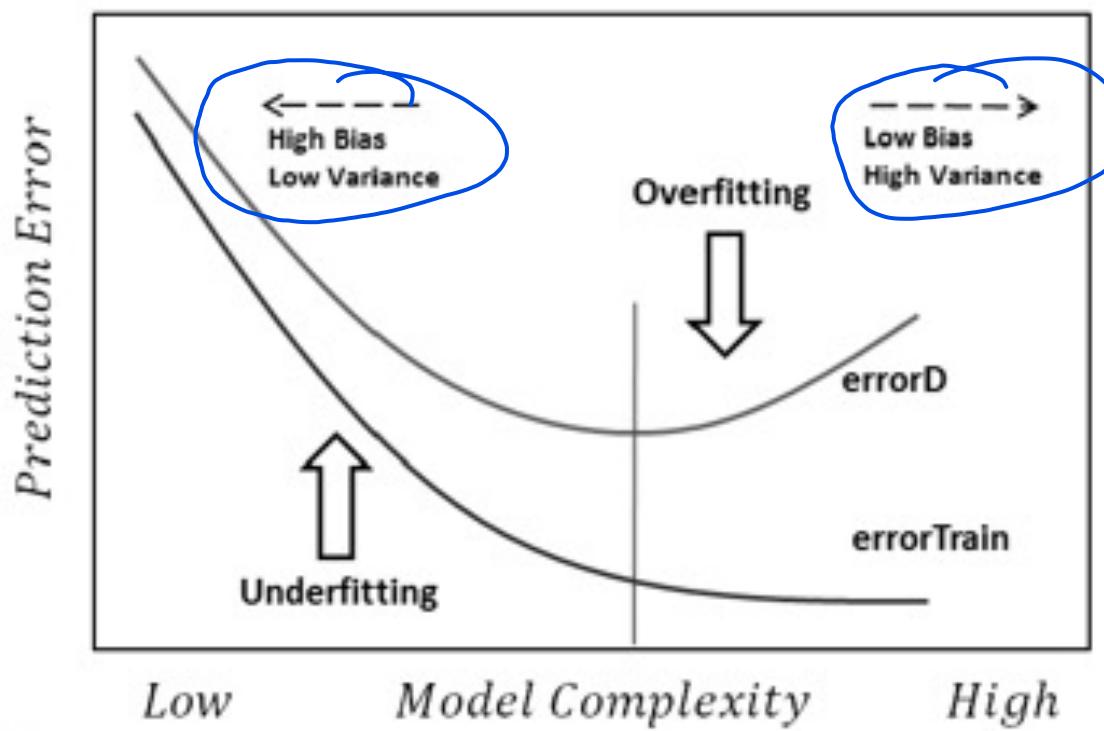


every time.
↳ Avg of accuracies.

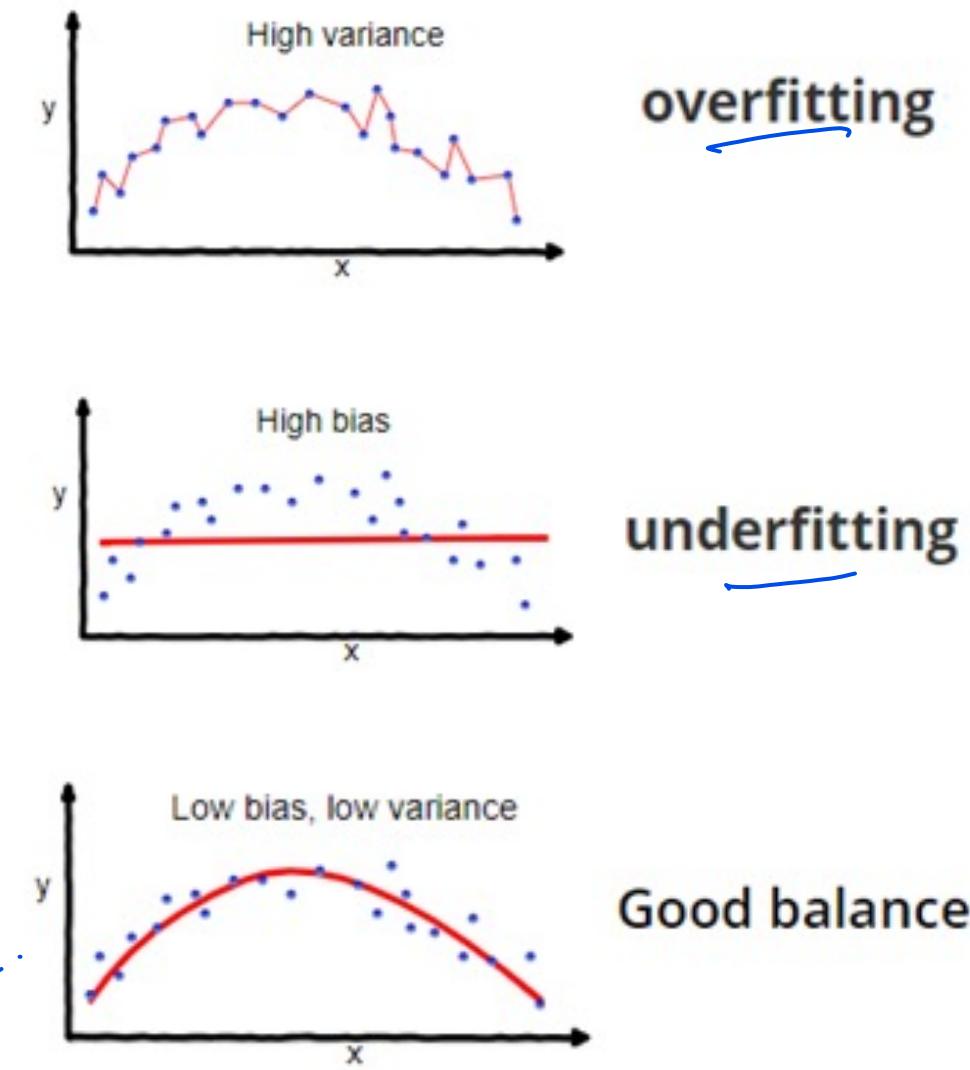
Underfitting and overfitting



Underfitting and overfitting



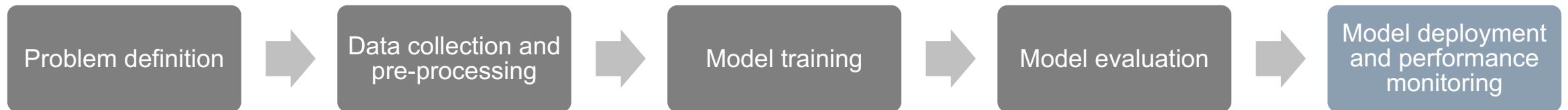
Model Comprehends α variance.
 $\epsilon_{\text{error}} \propto \frac{1}{2} \text{ bias}$.



Step 5. Model deployment and performance monitoring

After a successful training, we can **deploy the model**. Here we have often to consider the following issues:

- Real time requirements
- Robust hardware (sensors and processor)



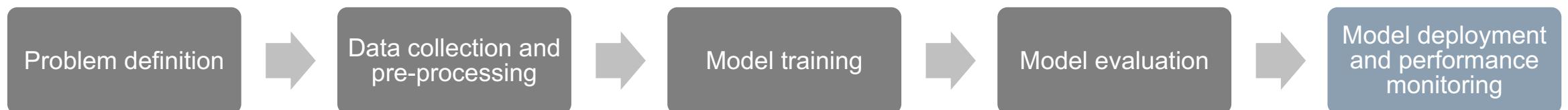
Step 5. Model deployment and performance monitoring

After a successful training, we can deploy the model. Here we have often to consider the following issues:

- Real time requirements
- Robust hardware (sensors and processor)

We must monitor the performance of our model and make sure it still produces satisfactory results.

- Sensors can malfunction and provide wrong data
- The data can be out of the trained range of the model





Time serie fundamentals

Common ML tasks with time series



The machine learning tasks

Machine learning (ML) can be regarded as a collection of methods that enables us to solve tasks which would be too difficult to be solved by a fixed written program designed by human beings.

From a phylosophycal point of view this is interesting because it can be seen as an attempt to formalize the concept of intelligence.

ML usually describes how machines should process examples.

- Examples are collections of features
- In the case of time series, features are the observations sorted in time.

observations in sorted time.

Time series classification

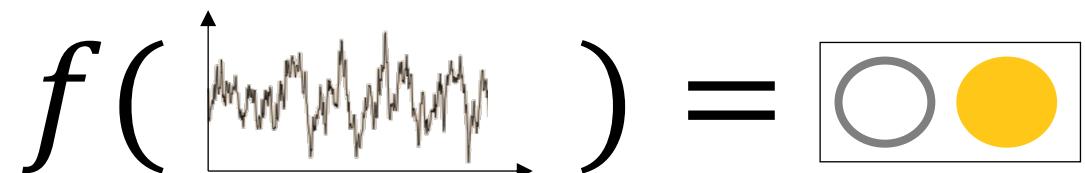
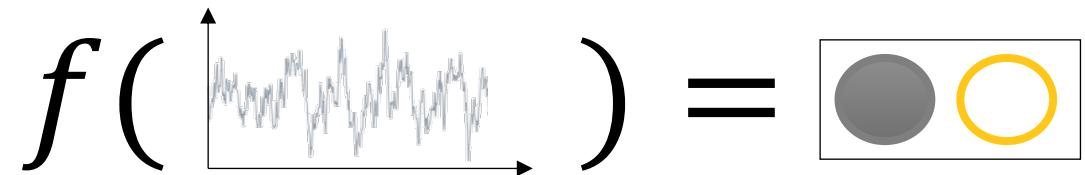
Let $\mathcal{D} = \{(S^{(1)}, c^{(1)}), \dots, (S^{(N)}, c^{(N)})\}$ be a dataset of pairs, where

- $S^{(i)}$ is a time series
- and $c^{(i)} \in \{0,1\}^K$ denotes the one-hot encoded class vector (also said, labels vector).

task

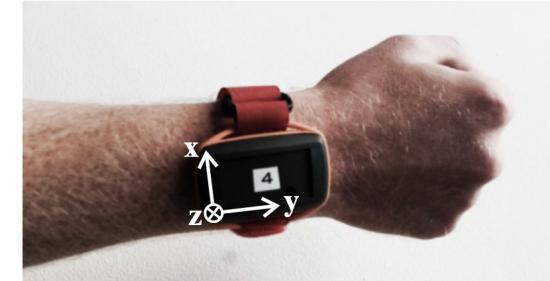
Then, a time series classification task is about learning a mapping function f , such that:

$$f(S^{(i)}) = c^{(i)} \quad \forall i \in \{1, \dots, N\}$$



Example of time series classification

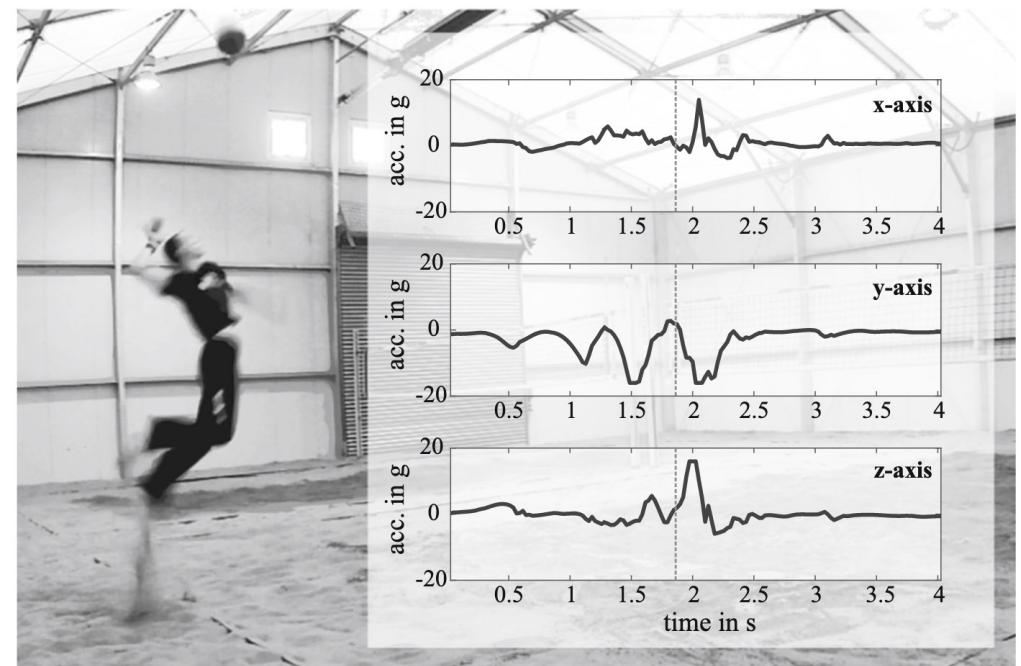
- Monitoring of player actions could help identifying and understanding risk factors and prevent such injuries.



Sensor attachment at the wrist of the dominant hand with a soft, thin wristband

Actions:

- Underhand serve
- Overhand serve
- Jump serve
- Underarm set
- Overhead set
- Shot attack
- Spike
- Block
- Dig
- Null class.



Time series forecasting

Let $S = \{s_1, \dots, s_{T_1}, s_{T_1+1}, \dots, s_{T_2}\}$ be a time series, with s_i being the i -th observation collected at time t_i , and $t_i < t_j, \forall j$.

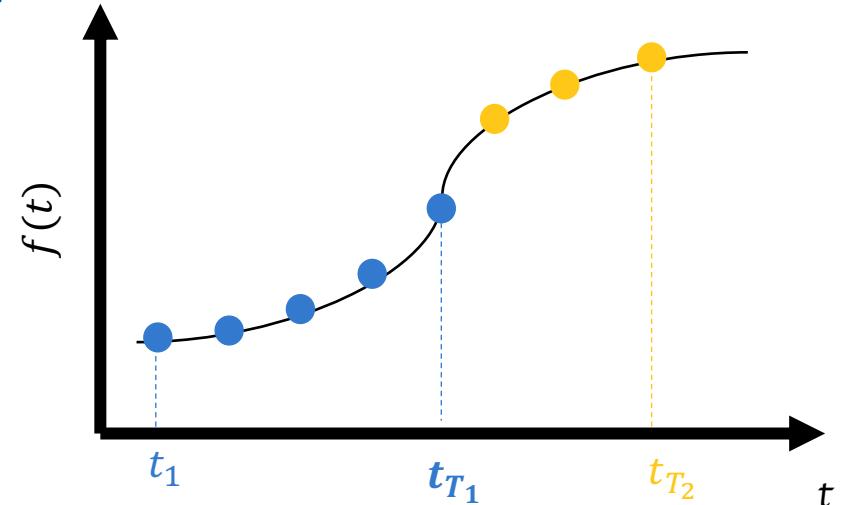
Then, a time series forecasting task is about predicting future values of a time series given some past data, i.e.,

$$f(s_1, \dots, s_{T_1}) = (s_{T_1+1}, \dots, s_{T_2})$$

$\uparrow_{\text{func}}^{\text{past}} : \text{(future values)}$

forecasting.

predict future given past data.



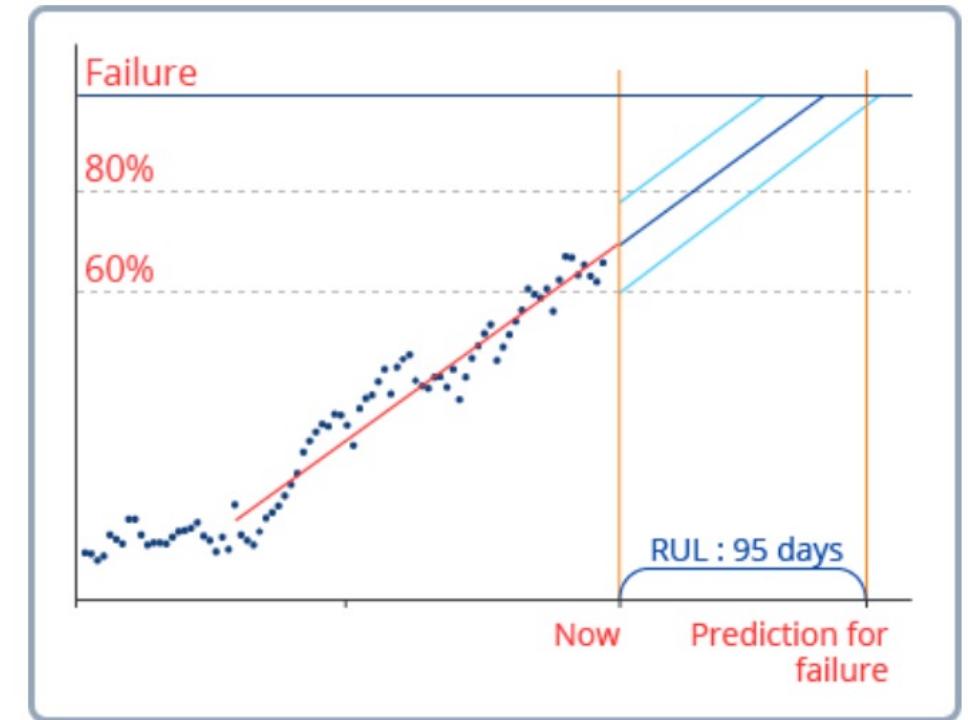
Example of time series forecasting: Prediction of Remaining Useful Life (RUL)

Objective:

- Machine components degrade over time
- This causes malfunctions in the production line
- Replace component before the malfunction!

Realization:

- Use data of malfunctioning systems
- Fit a regression model to the data and predict the RUL
- Use the model on active systems and apply necessary maintenance



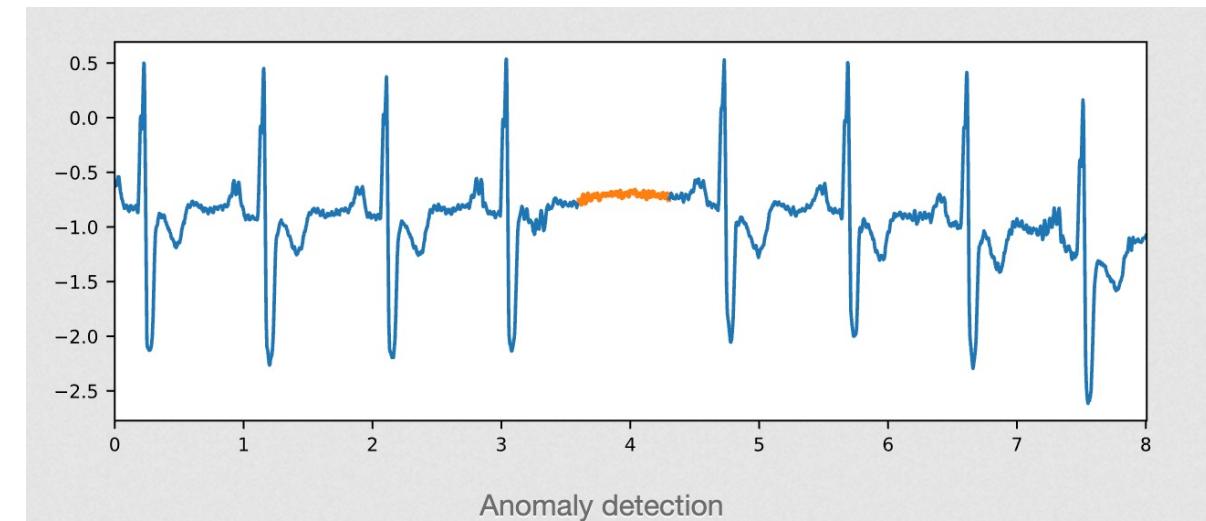
Anomaly detection on time series

Let $S = \{s_1, \dots, s_T\}$ be a time series, with s_i being the *i-th observation* collected at time t_i .

Then, an anomaly detection task is that of predicting the probability of a certain observation to be anomalous,

$$f(s_i) = p_i, \forall i \in \{1, \dots, T\}$$

with $p_i = 0$ for regular data and $p_i = 1$ for anomalous data.



(a) <https://siebert-julien.github.io/time-series-analysis-python/>

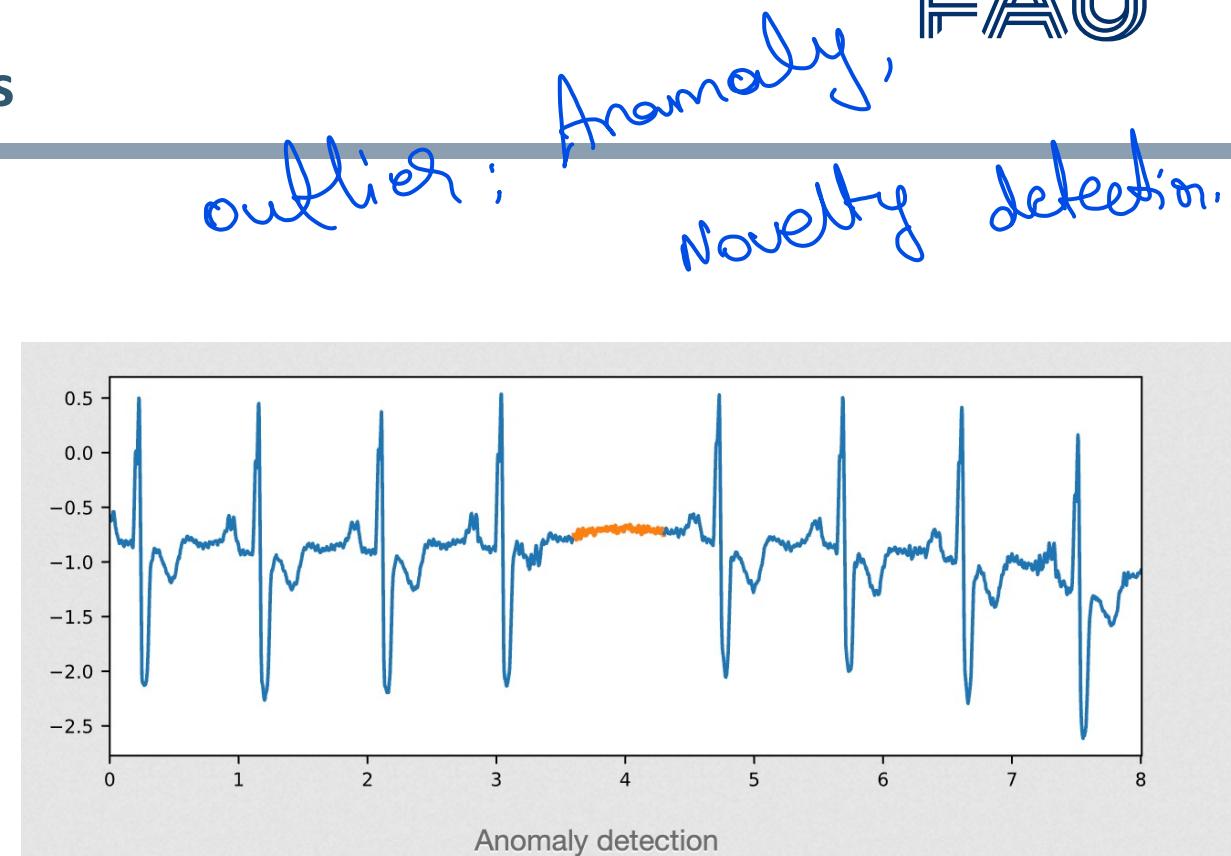
func (Sample) = $\begin{cases} \text{Anomalous.} \\ P_i = 1? \end{cases}$

Examples of anomaly detection on time series

Anomaly detection, sometimes also called outliers detection or novelty detection, is therefore the task of finding abnormal data points (equiv., outliers).^(a)

Examples of real world applications of anomaly detection on time series are:

- detecting fraud transactions
- fraudulent insurance claims
- cyber attacks to detecting abnormal equipment behaviors



^(a) <https://siebert-julien.github.io/time-series-analysis-python/>

Time series segmentation

Let $S = \{s_1, \dots, s_T\}$ be a time series, with s_i being the i -th observation collected at time t_i .

Time series segmentation is the task of splitting data points into segments, which reveal underlying properties of the generation process, which can formalized as the process of assigning every sample to its corresponding cluster, i.e., $f(s_i) = c_i$, with $c_i \in \{c_1, \dots, c_M\}$.

↑
Assign samples

Samples → clusters. $\{c_1, \dots, c_m\}$

↑ Earlier separation.

Example of time series segmentation

A typical example is that of online handwriting recognition.

A time series describes a list of coordinates, e.g., the point of a pen over a touchscreen surface, collected over time.

The task is to determine segments corresponding to a single letter.

(Segmentation)



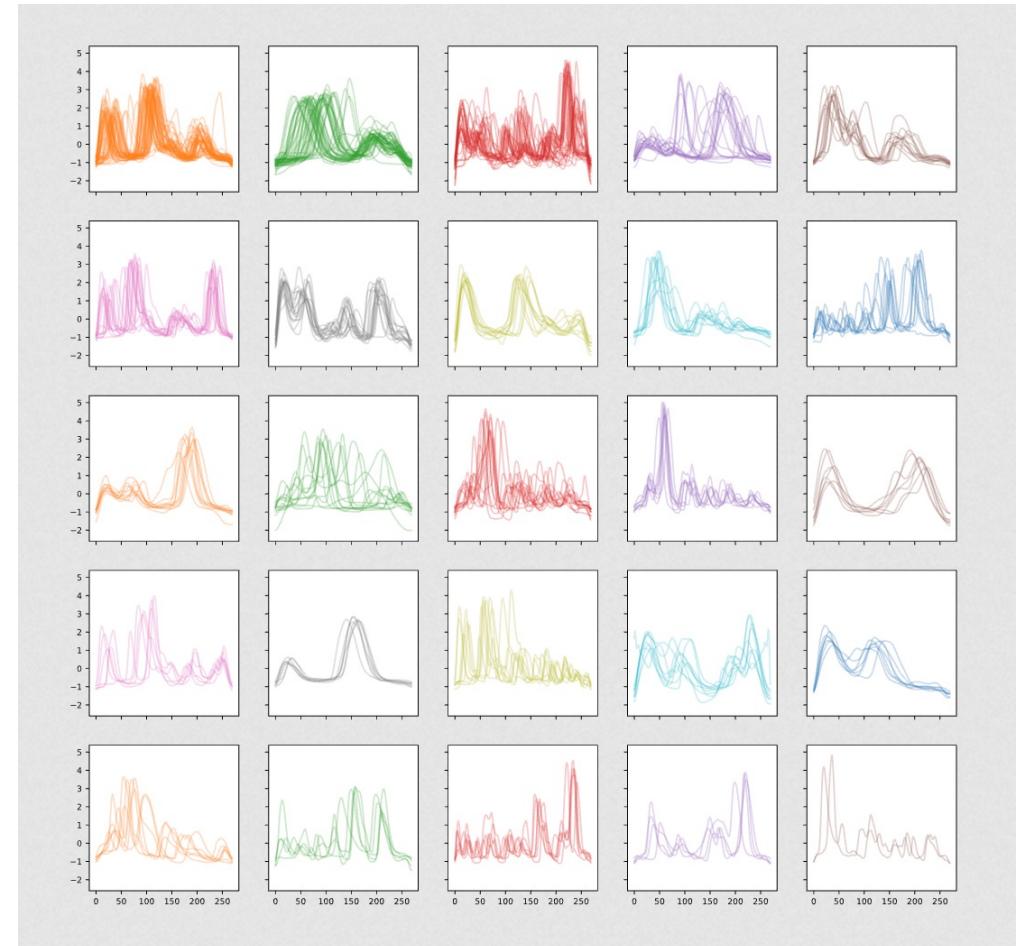
Time series clustering

→ Data Analysis
ε Preprocessing
Grouping.

Clustering can be applied to time series with the goal of grouping together similar sequences.

This finds important application in data analysis and pre-processing

- Find similar customers behaviours and exploit this information in recommender systems



(a) <https://siebert-julien.github.io/time-series-analysis-python/>



Time series fundamentals

Example: Linear regression for time series forecasting



Motivation

Linear regression is used in multiple scenarios each and every day!

- E.g., Trend Analysis in financial markets, sales and business



Motivation

Linear regression is used in multiple scenarios each and every day!

- E.g., Trend Analysis in financial markets, sales and business

The problem has to be simple:

- Dataset is small
- Linear model is enough, i.e., Trend Analysis
- Linear models are the basis for complex models, i.e., Deep Networks



Linear Regression is a method to fit linear models to our data!

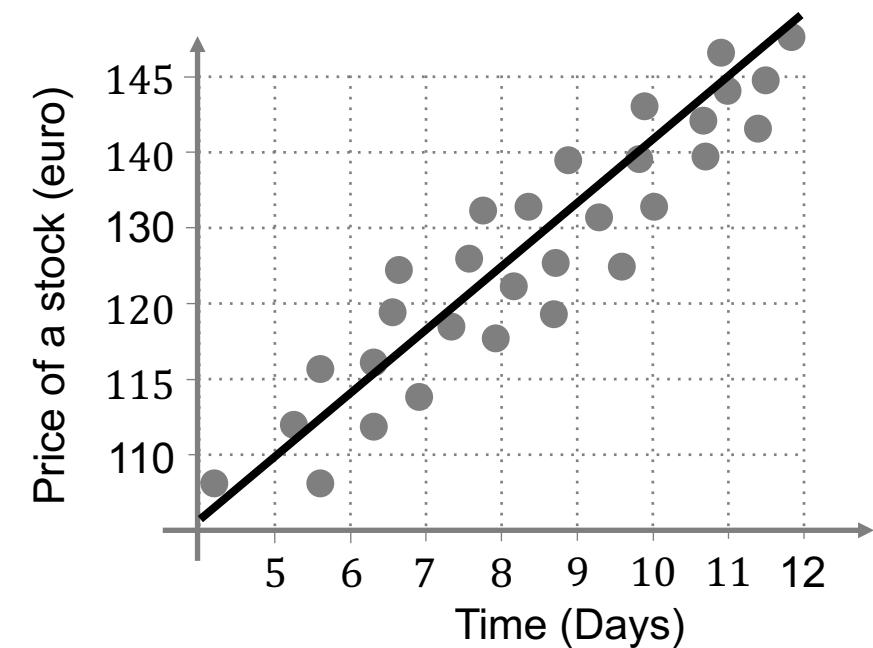
The linear model:

$$f(\mathbf{x}) = w_0 \cdot x_0 + w_1 \cdot x_1 = y$$

$$\mathbf{x} = \begin{pmatrix} x_0 \\ x_1 \end{pmatrix} = \begin{pmatrix} 1 \\ \text{Days} \end{pmatrix}$$

y = Stock price

w = Weights



Linear Regression is a method to fit **linear models** to our data!

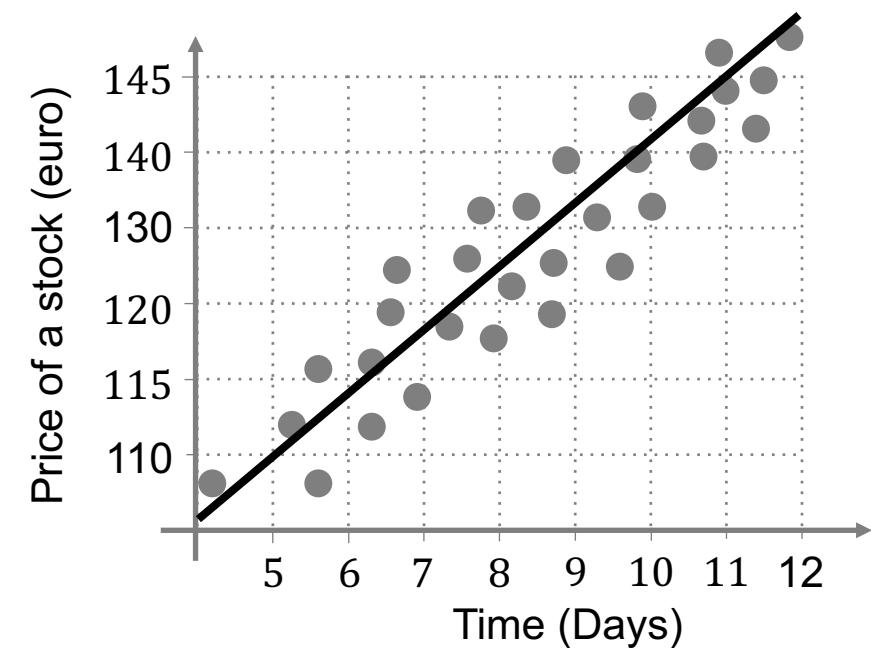
The linear model:

$$f(\mathbf{x}) = w_0 \cdot x_0 + w_1 \cdot x_1 = y$$

The linear model (in our example):

$$f(\mathbf{x}) = 70 \cdot x_0 + 6.5 \cdot x_1 = y$$

→ Finding a good w_0 and w_1 is
called **fitting!** → 



The previous description of the linear model
is not accurate!

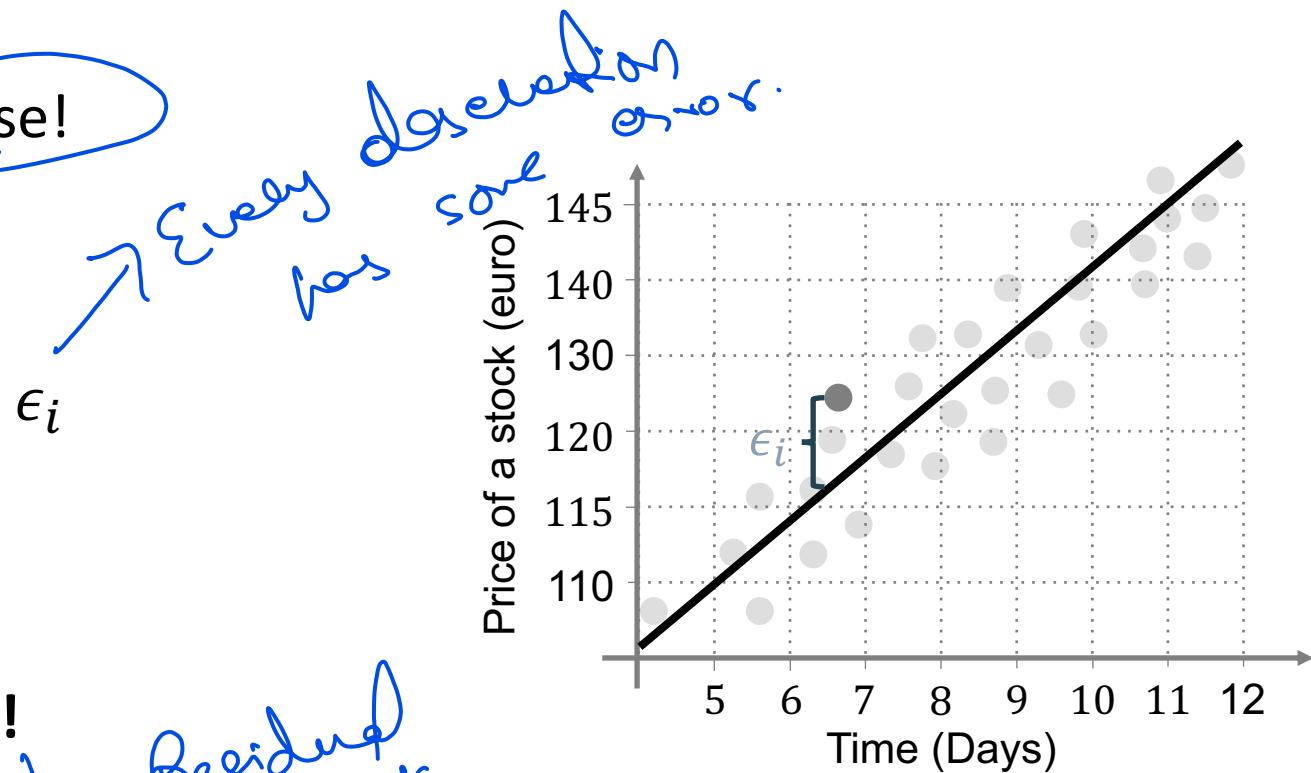
Reason: Real Systems produce noise!

Linear model with noise:

$$f(\mathbf{x}) = w_0 \cdot x_0 + w_1 \cdot x_1 + \epsilon_i$$

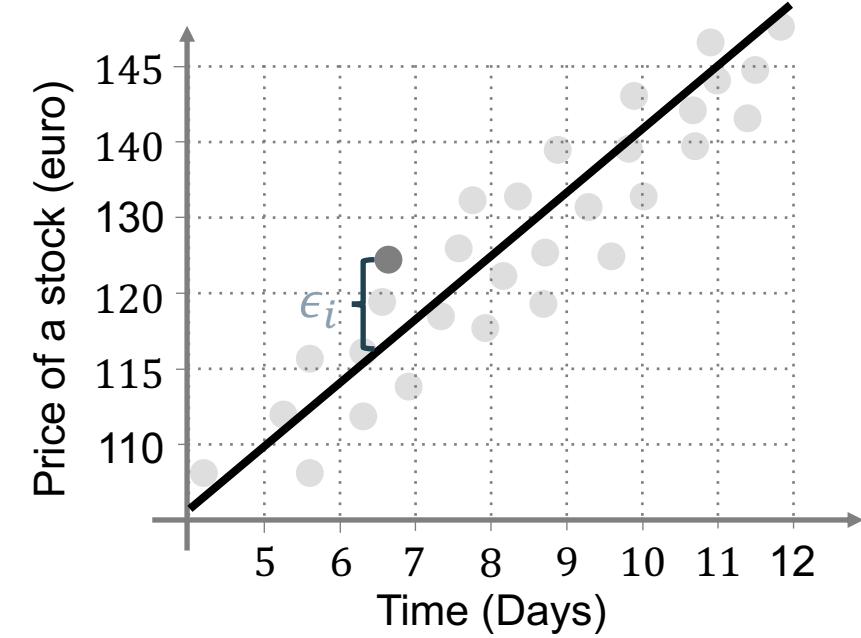
The ϵ_i and the summation ϵ of all samples is called **Residual Error**!

Typically, we assume ϵ is Gaussian
Distributed (i.e. Gaussian noise)



A more general formulation:

$$f(\mathbf{x}) = \sum_{j=1}^D w_j x_j + \epsilon = \mathbf{y}$$



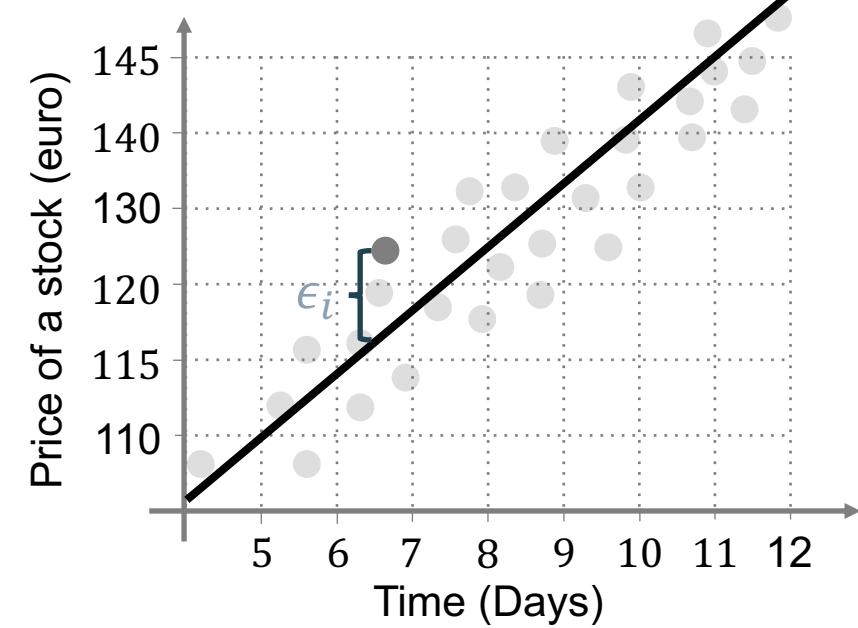
A more general formulation:

weighted sum + noise term

$$f(\mathbf{x}) = \sum_{j=1}^D w_j x_j + \epsilon = \mathbf{y}$$

The **model parameters** are:

$$\theta = \{w_0, \dots, w_n, \sigma\} = \{\mathbf{w}, \sigma\}$$



A more general formulation:

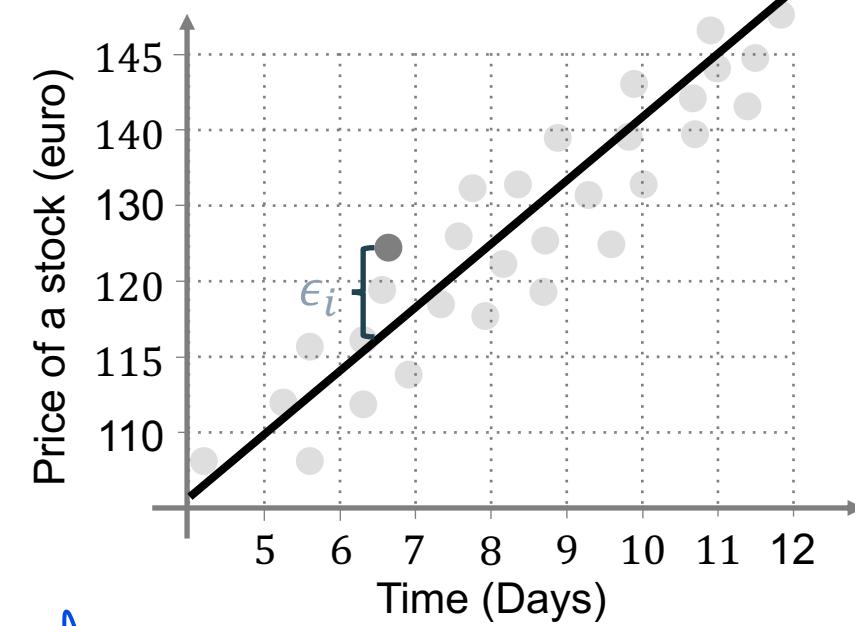
$$f(\mathbf{x}) = \sum_{j=1}^D w_j x_j + \epsilon = \mathbf{y}$$

The model parameters are:

$$\underline{\theta} = \{w_0, \dots, w_n, \sigma\} = \{\mathbf{w}, \sigma\}$$

↑ Noise
 ↓ Parameters
 ↓ weight vector

We define the optimal parameters as
 (Maximum likelihood estimation, MLE):



→ optimal parameters finding

A solution to this problem is given by the minimization of the negative log likelihood:

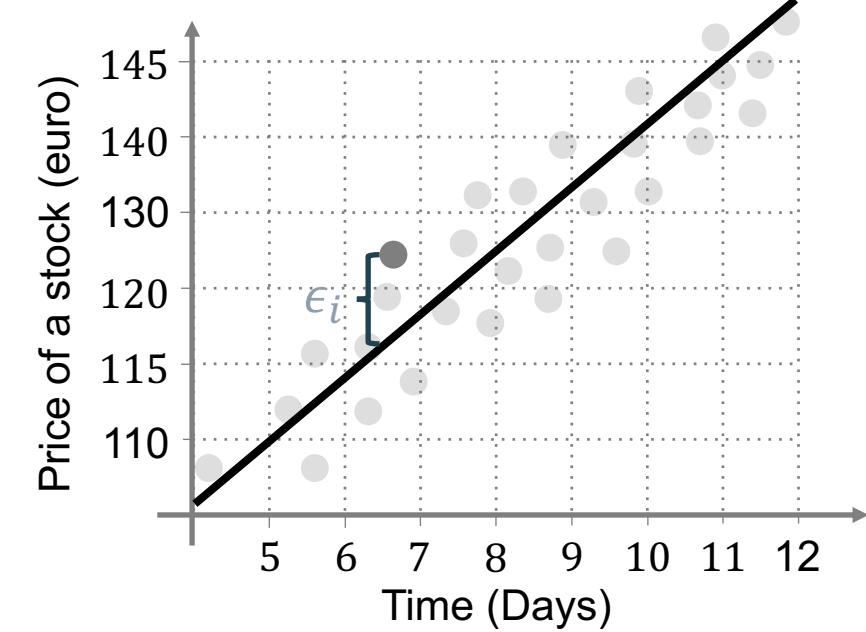
$$\text{NLL}(\theta) = \frac{1}{2} (y - xw)^T (y - xw)$$

We find the minimum conventionally by using the derivative:

$$\text{NLL}'(\theta) = x^T x w - x^T y$$

The solution, i.e., the minimum is:

$$w = (x^T x)^{-1} x^T y$$



CS?

Closed form Solution



Time series fundamentals

Recap



Recap

Types of machine learning

- Supervised learning
- Unsupervised learning
- Reinforcement learning

3 main
paradigms.

ML pipeline

- From problem definition to model deployment
- Training and Evaluation good practices

Common ML tasks with time series

- Time series classification
- Forecasting
- Anomaly detection
- Time series segmentation
- Time series clustering

Linear regression for time series forecasting

