Friedrich-Alexander-Universität
Technische Fakultät

Machine Learning
Data Analytics

FAU

Machine Learning for Time Series (MLTS)

Lecture 2: Time Series Fundamentals and Definitions (Part 2)

Dr. Dario Zanca

Machine Learning and Data Analytics (MaD) Lab
Friedrich-Alexander-Universität Erlangen-Nürnberg
24.10.2024

Topics overview

1. Time series fundamentals and definitions (Part 1)
2. Time series fundamentals and definitions (Part 2)
3. Bayesian Inference and Gaussian Processes
4. State space models (Kalman Filters)
5. State space models (Particle Filters)
6. Autoregressive models
7. Data mining on time series
8. Deep Learning (DL) for Time Series (Introduction to DL)
9. DL – Convolutional models (CNNs)
10. DL – Recurrent models (RNNs and LSTMs)
11. DL – Attention-based models (Transformers)
12. DL – From BERT to ChatGPT
13. DL – New Trends in Time Series processing
14. Time series in the real world

Topics overview

1. Time series fundamentals and definitions (Part 1)
2. Time series fundamentals and definitions (Part 2)
3. Bayesian Inference and Gaussian Processes
4. State space models (Kalman Filters)
5. State space models (Particle Filters)
6. Autoregressive models
7. Data mining on time series
8. Deep Learning (DL) for Time Series (Introduction to DL)
9. DL – Convolutional models (CNNs)
10. DL – Recurrent models (RNNs and LSTMs)
11. DL – Attention-based models (Transformers)
12. DL – From BERT to ChatGPT
13. DL – New Trends in Time Series processing
14. Time series in the real world

Organisational Information

Machine Learning for Time Series (MLTS)

- 5 ECTS
- Lectures + Exercises

Course times

Lectures (on campus) - *Dr. Dario Zanca and Dr. Emmanuelle Salin*

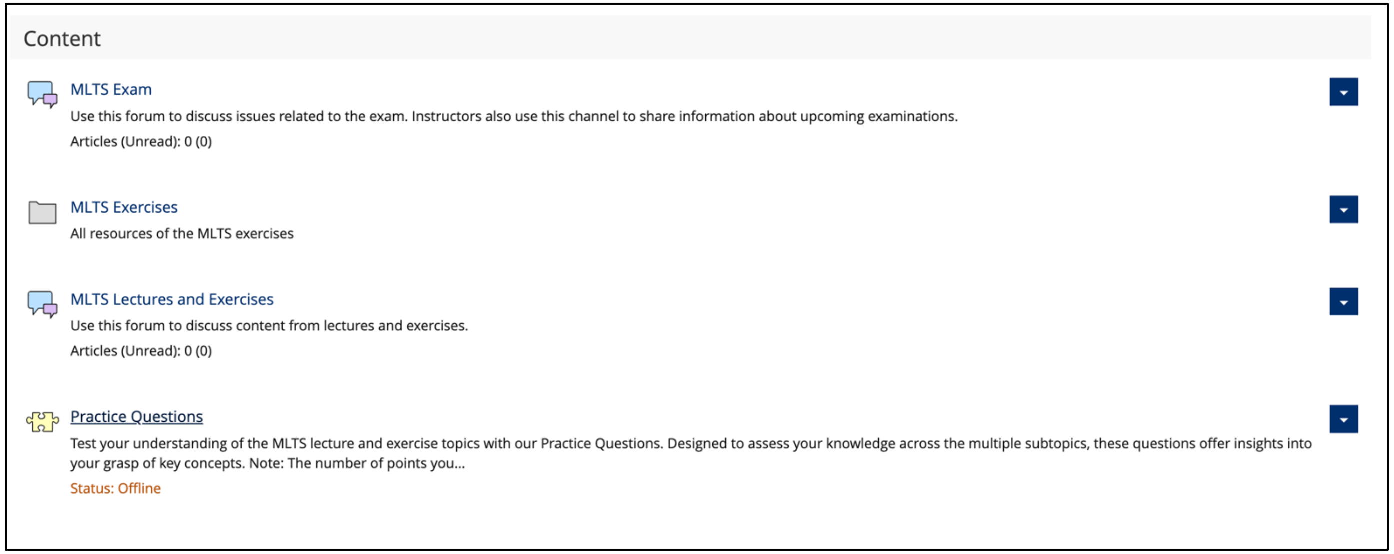
- Lectures on Thursdays, h. 14.15 - 15.45 (90 mins)
- Consultation hours by appointment
- Recordings from past years available (only **partial** topics overlap)

Exercises (on campus) – *Richard Dirauf, M.Sc.*

- Exercises on Thursdays, h. 12.15 – 13.45
  - starting on October 31st
- Recordings from past years available (only **partial** topics overlap)

Course StudON

StudOn 2024-2025: https://www.studon.fau.de/crs5911979.html



If you are entitled to take or re-take the exam, you can request access to this year's material at this "MATERIALS ONLY" StudON group:

https://www.studon.fau.de/crs6083795_join.html

Written Exam (5 ECTS)

- Written examination
- 60% content from lectures, 40% content form exercises
- The exam will be in person and it will be a closed-book exam

Course organizers
Lecturers

Machine Learning and Data Analytics (MaD) Lab

- Dr. Dario Zanca, dario.zanca@fau.de *
- Dr. Emmanuelle Salin, emmanuelle.salin@fau.de *
- Prof. Dr. Björn Eskofier, bjoern.eskofier@fau.de

* Please, address all your correspondence about the course to Dr. Dario Zanca and Dr. Emmanuelle Salin

Course organizers
Teaching assistants

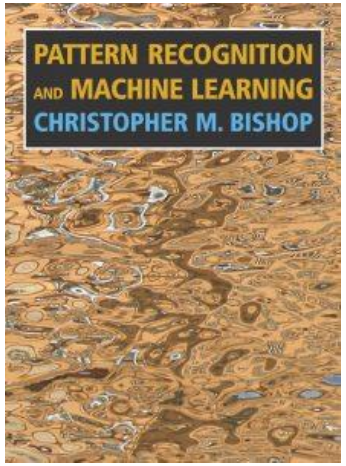
Machine Learning and Data Analytics (MaD) Lab

- Richard Dirauf (M.Sc.), richard.dirauf@fau.de

References

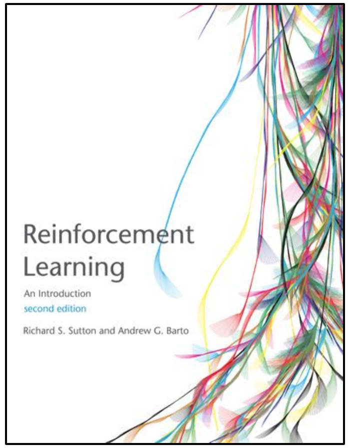
Pattern Recognition and Machine Learning

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



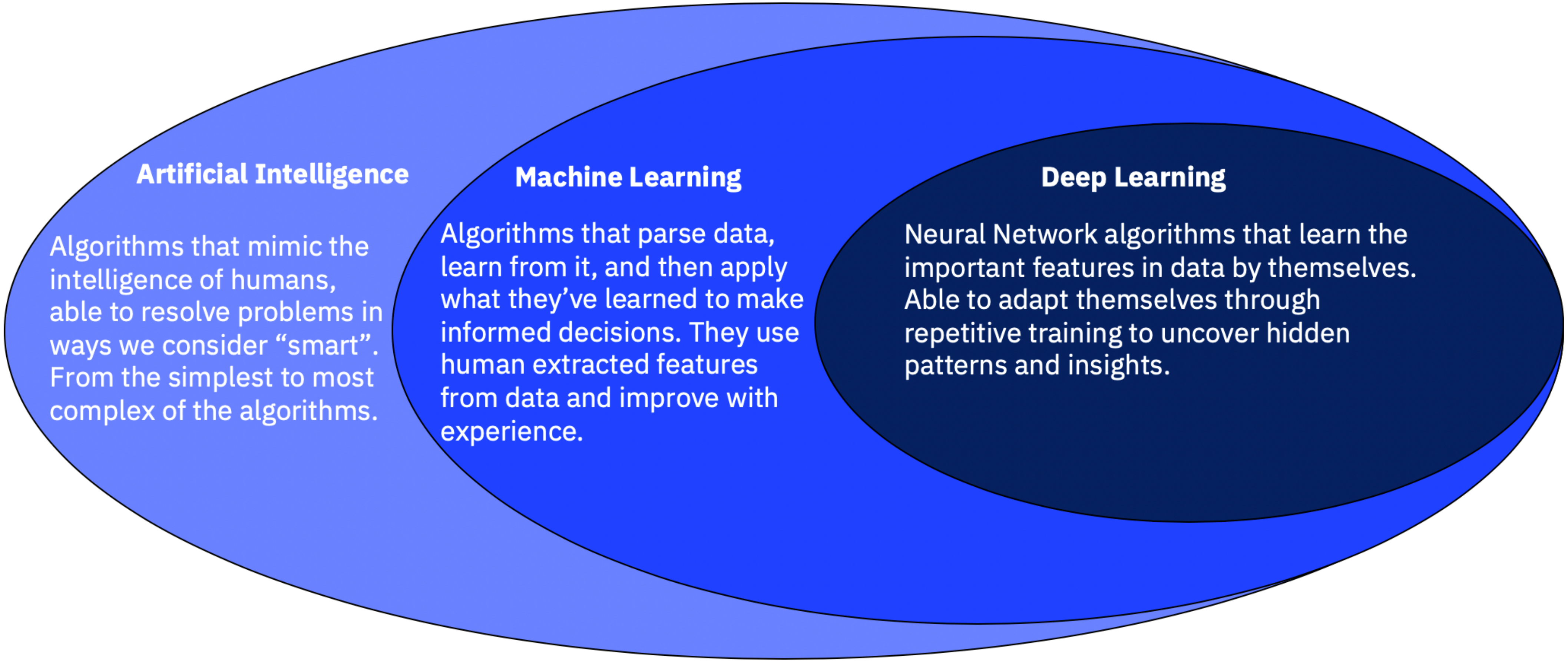
In this lecture...

1. *Types of Machine Learning*
2. *ML Pipeline and Good Practices*
3. *Common ML Tasks with Time Series*
4. *Example: Linear Regression for Time Series Forecasting*
5. *Recap*

Machine learning basics

Types of machine learning

What is Machine Learning (ML)?



Source: https://www.ibm.com/blogs

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

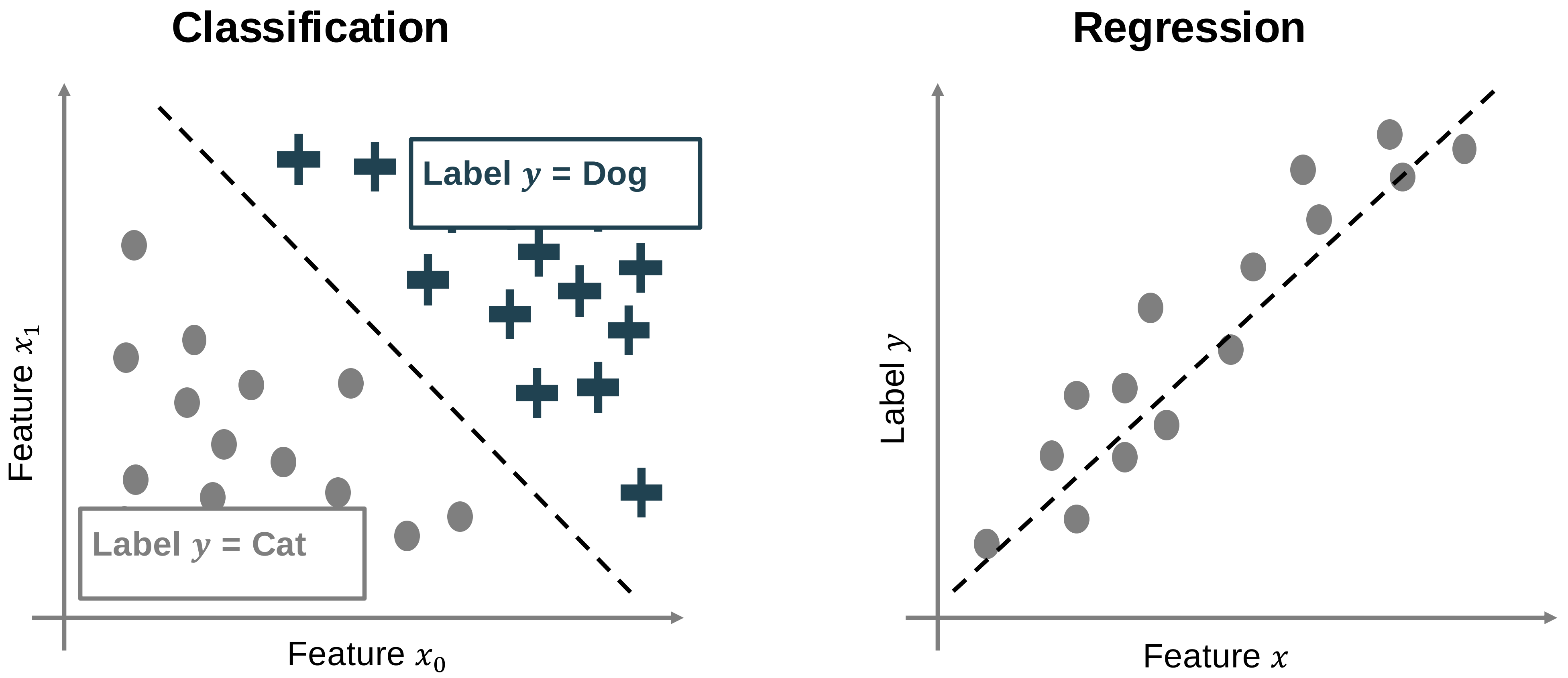
The agent observes some example **Input (Features)** and **Output (Label) pairs** and learns a **function that maps input to output**.

Key Aspects:

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

→ Resolves classification and regression problems

Supervised Learning Problems



Source: https://www.kdnuggets.com

Regression

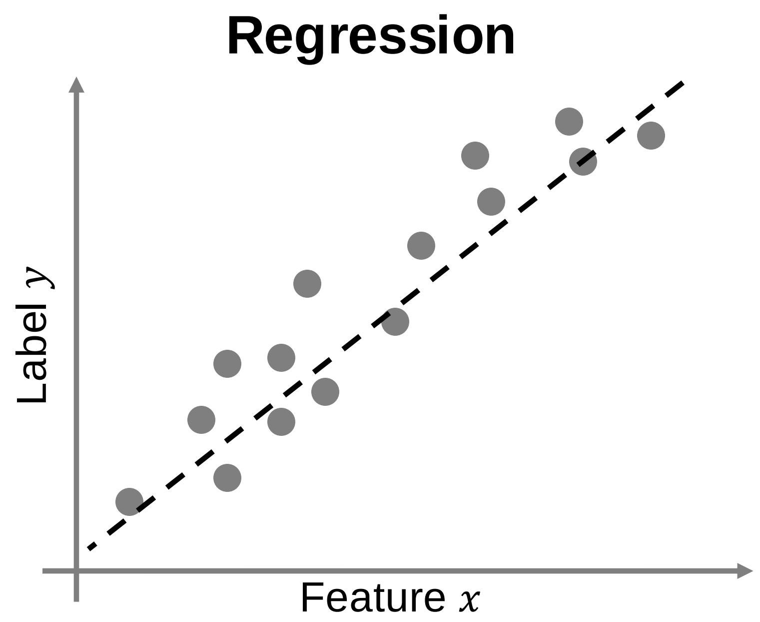
Regression is used to predict a **continuous value**

Training is based on a set of input – output pairs (**samples**)
$\mathcal{D} = \{(\boldsymbol{x}_0, \boldsymbol{y}_0), (\mathbf{x}_1, \boldsymbol{y}_1), \dots, (\mathbf{x}_\mathrm{n}, \boldsymbol{y}_n)\}$

Sample : $(\boldsymbol{x}_i, \boldsymbol{y}_i)$

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

$\boldsymbol{y}_i \in \mathbb{R}$ is the **label value** of sample i



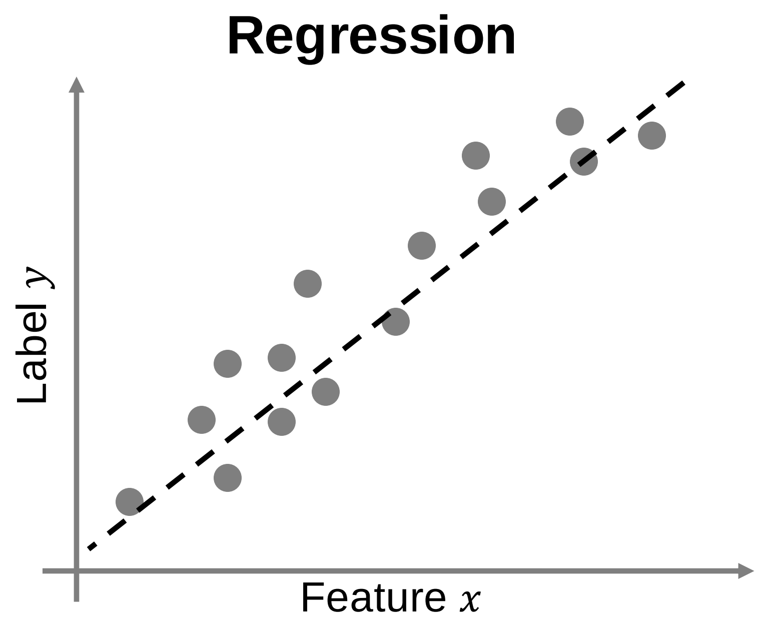
Regression

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

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

$$f(\boldsymbol{x}_i) = \boldsymbol{y}_i \quad , \forall (\boldsymbol{x}_i, \boldsymbol{y}_i) \in \mathcal{D}$$

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



Classification

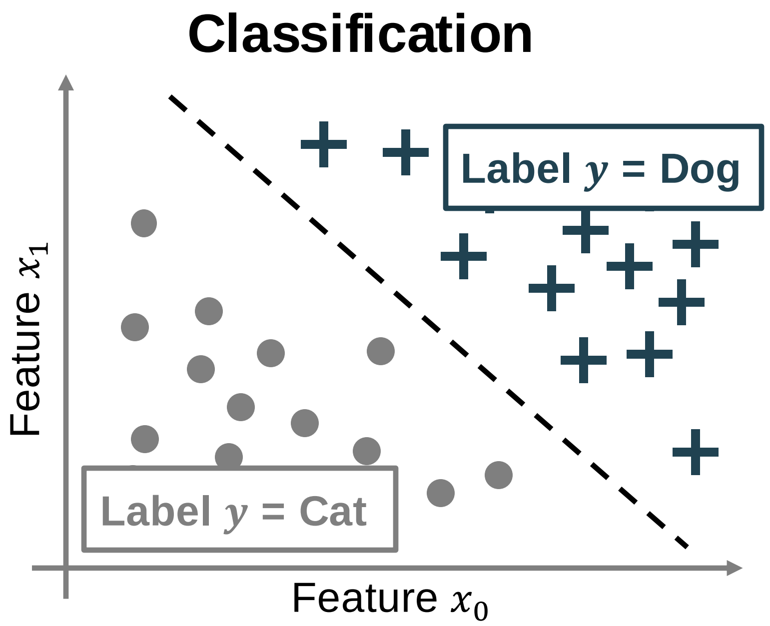
Classification is used to predict the **class** of the input

Sample : $\boldsymbol{s}_{\mathbf{i}} = (\vec{x_i}, \vec{y_i})$

$\vec{y_i} \in L$ is the **label** of sample i

In this example:

- $L = \{„Cat“ \coloneqq -1, „Dog“ \coloneqq 1\}$
- Binary classification problem
- The output belongs to only one class!



Classification

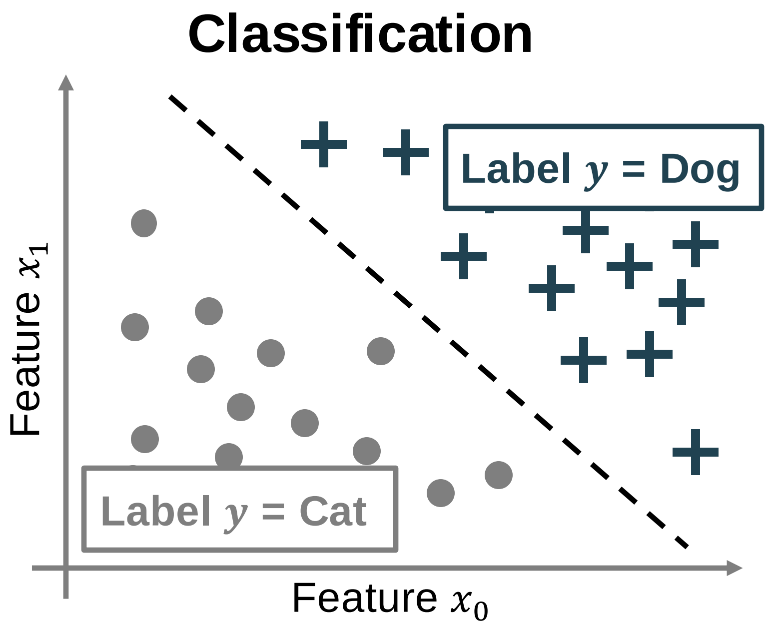
Goal:

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

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

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

In this case $f(x) = 0$ is a **decision boundary!** (Black Line)



Supervised Learning Problems

Regression	Classification
• The output are **continuous or real values**	• The output variable must be a **discrete value (class)**
• We try to fit a **regression model**, which can predict the output more accurately	• We try to find a **decision boundary**, which can divide the dataset into different classes.
• Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, Stock market prediction etc.	• Classification Algorithms can be used to solve classification problems such as Hand-written digits(MNIST), Identification of cancer cells, Defected or Undefected solar cells etc.

Source: https://scikit-learn.org/

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

Unsupervised Learning

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

Key Aspects:

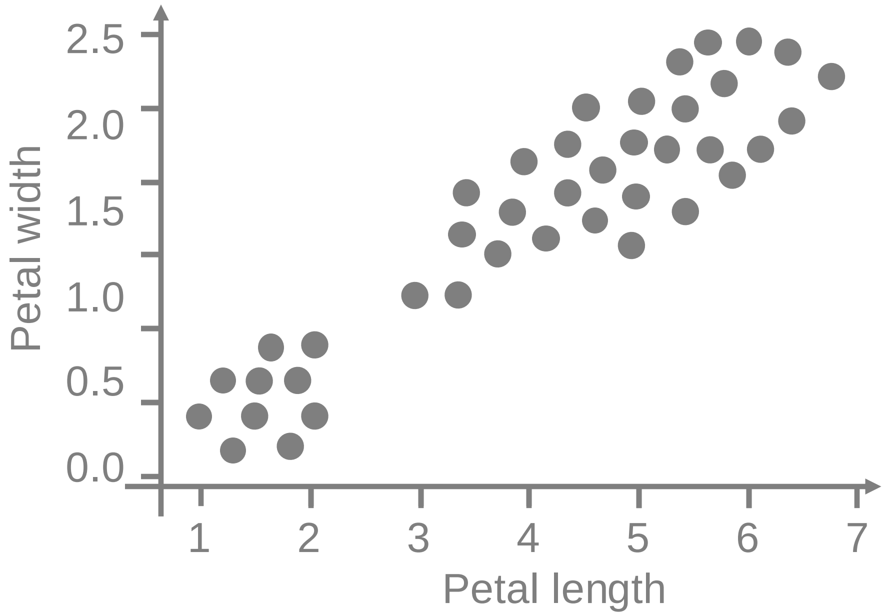
- Learning is **implicit**
- Learning using **indirect feedback**
- Methods are **self-organizing**

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

Clustering

Goal: Identify similar samples and assign them the same label

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



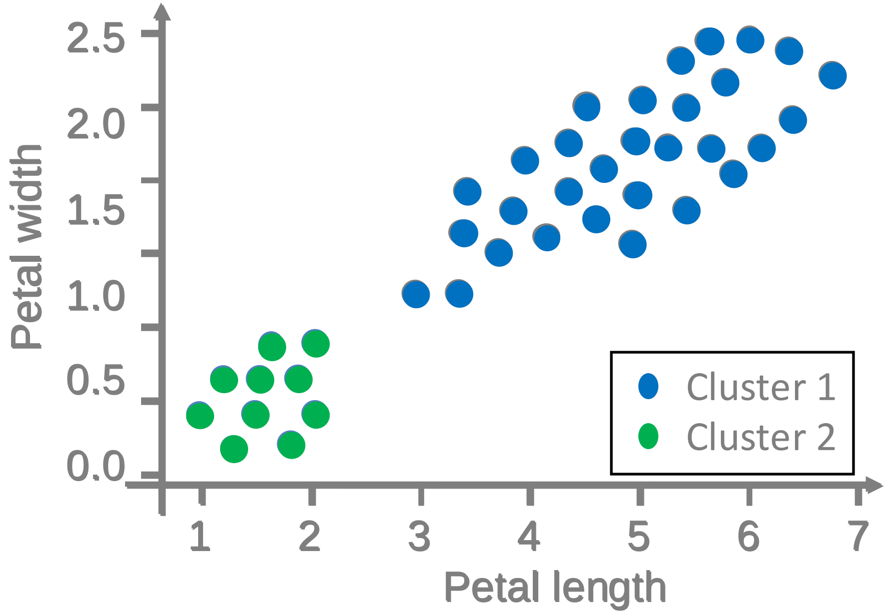
Clustering basic principles:

- 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

→ **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?

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

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

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

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

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

	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

	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

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

Key Aspects:

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

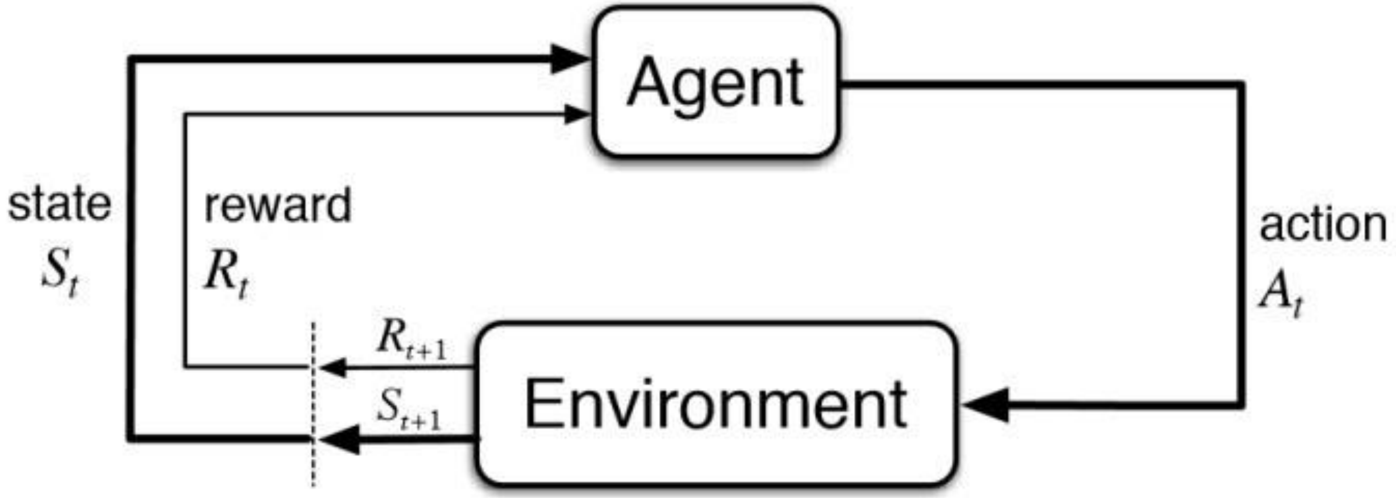
Resolves **control** and **decision** problems

- 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.



Source: Reinforcement Learning: An Introduction, Richard S. Sutton, Andrew G. Barto, 2018

Reinforcement Learning vs Unsupervised Learning

Unsupervised Learning	Reinforcement Learning
• An indirect feedback is generated according to **a metric**	• The feedback is given by a **reward signal**
• Feedback is **instantaneous**	• Feedback can be **delayed** (credit assignment problem)
• Learning by using **static data** (no re-recording of data necessary)	• Training is **based on trials** i.e. interaction between environment and agent (re-recording necessary)
• Prediction does **not affect future measurements** – The data is assumed Independent Identically Distributed (i.i.d)	• The prediction (actions) **affect future measurements** i.e. the measurements are no necessarily i.i.d!

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:

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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
- What kind of **data** we need

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)

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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

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

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

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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

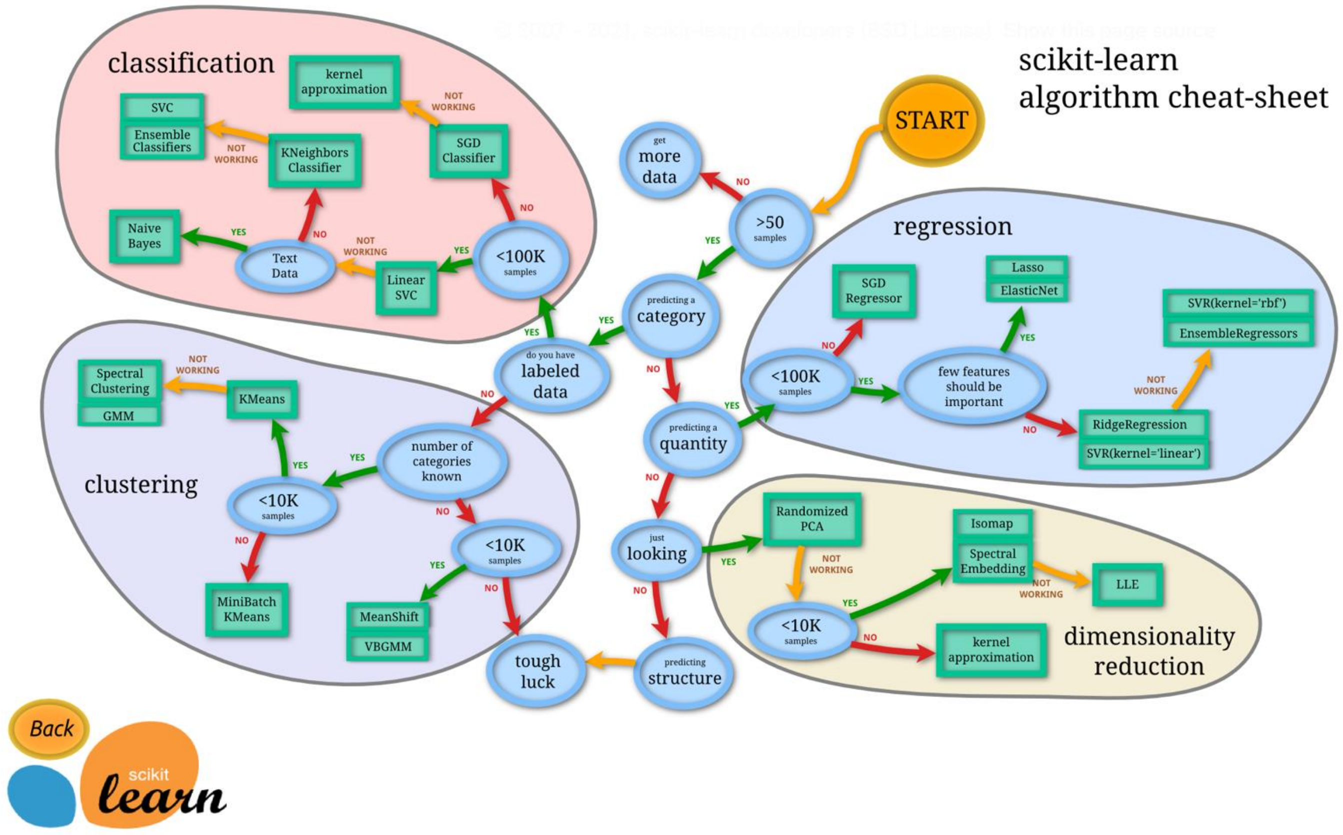
Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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.

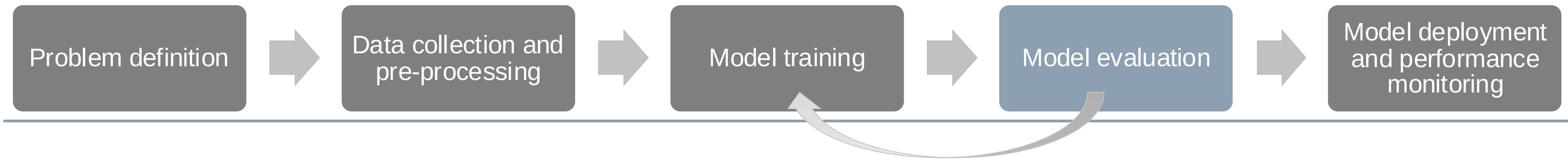
Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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.

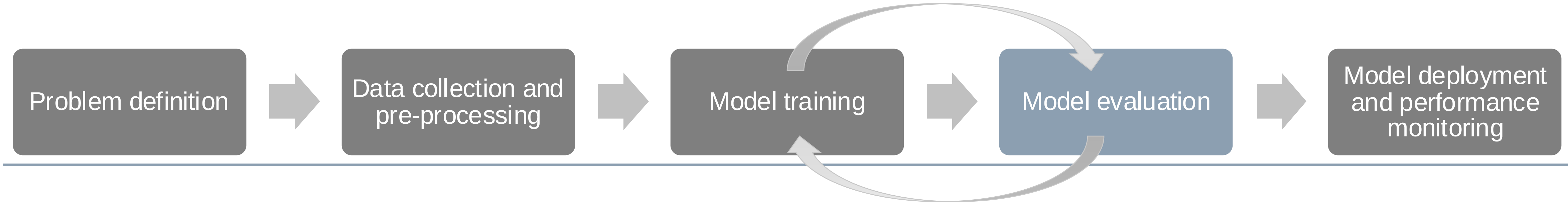


Step 4. Model evaluation

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.



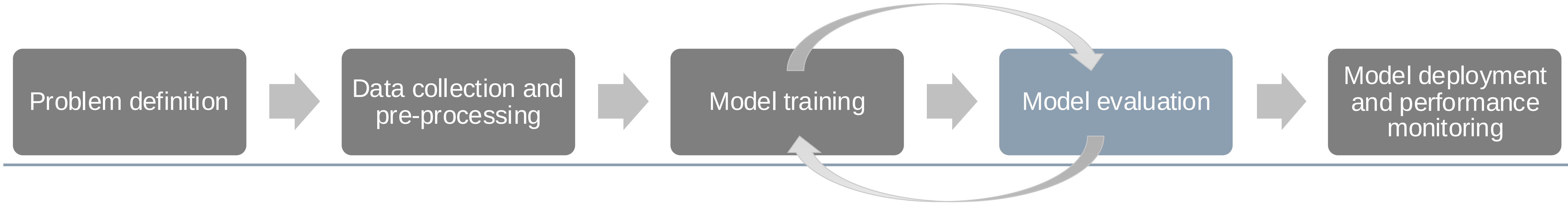
Step 4. Model evaluation

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.

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



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.

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 and idea about the direction of error.

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

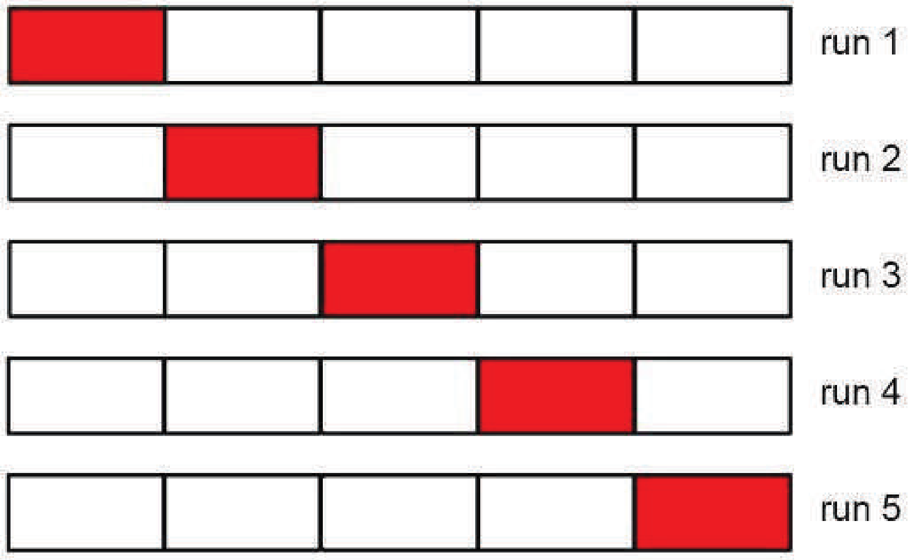
Mean Squared Error (MSE):

- 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.

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

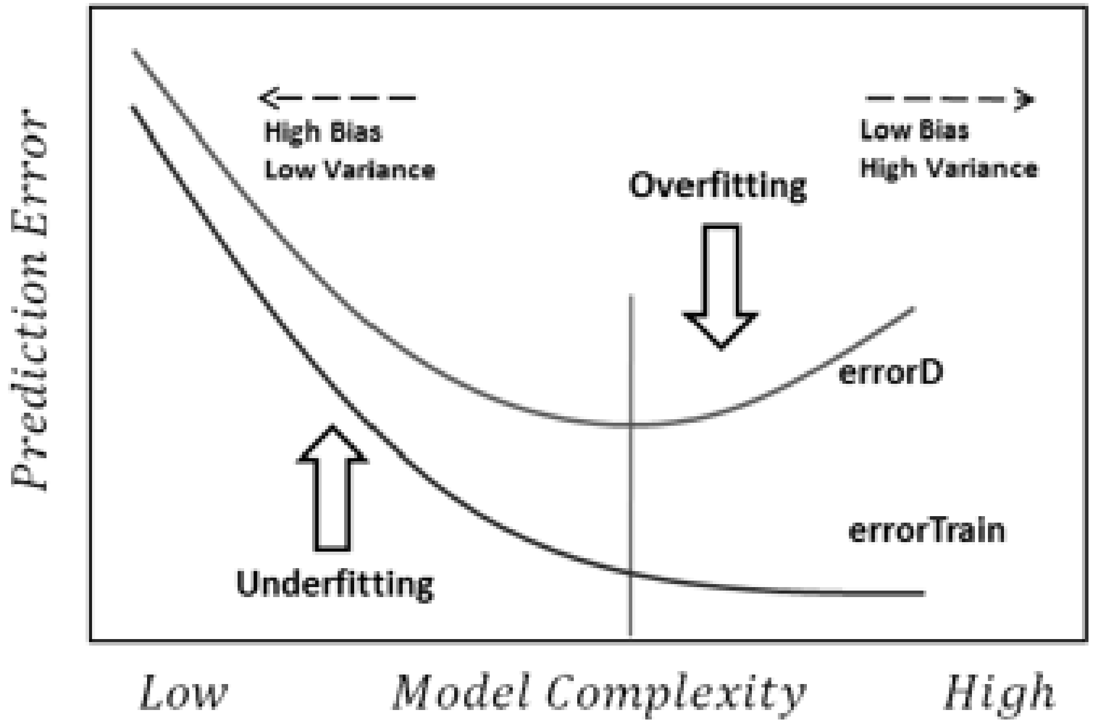
Cross validation

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

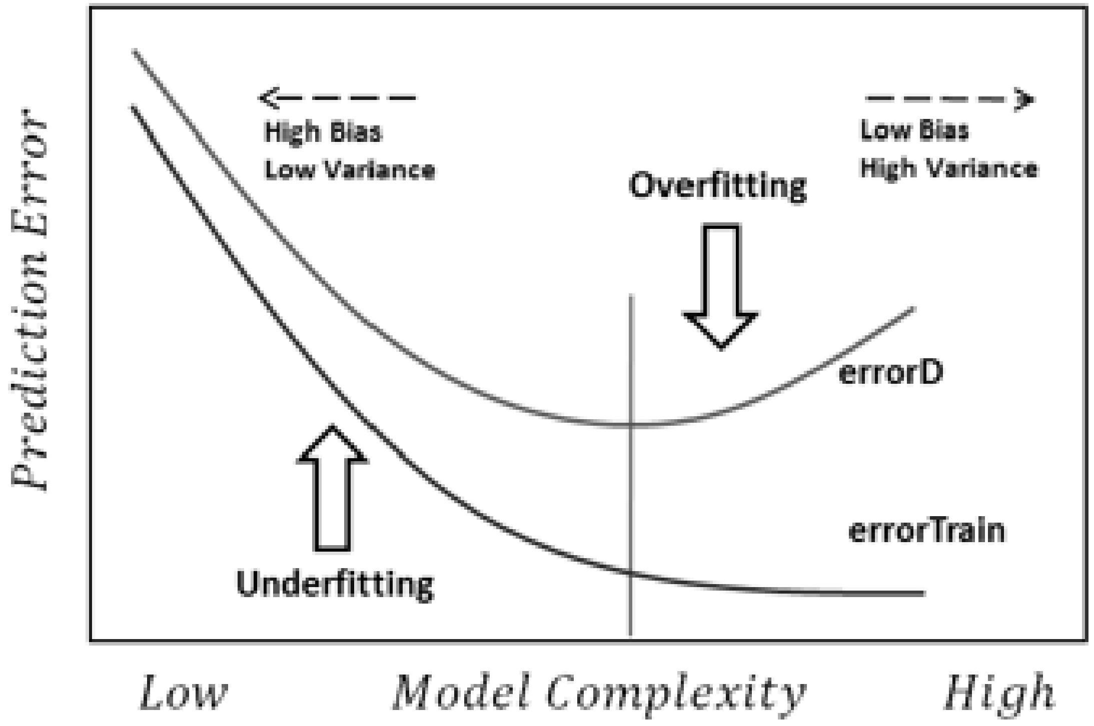


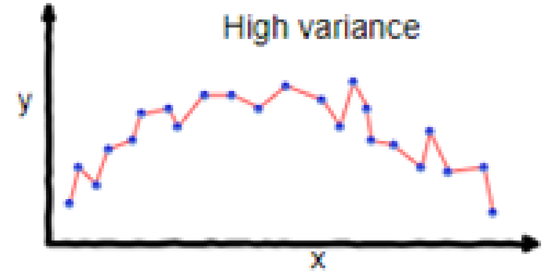
- Particularly useful when the dataset size is small
- It can be also used for hyper-parameters selection

Underfitting and overfitting

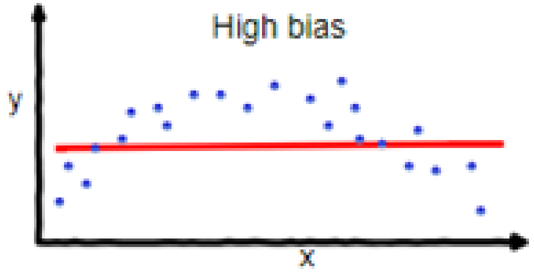


Underfitting and overfitting

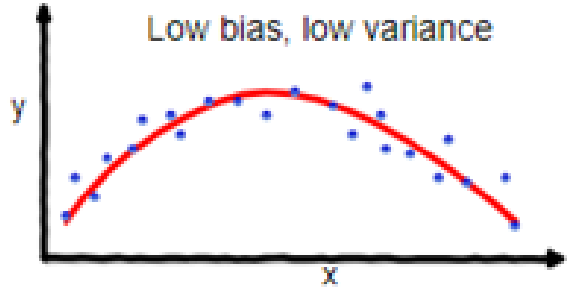




overfitting



underfitting



Good balance

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

Problem definition → Data collection and pre-processing → Model training → Model evaluation → Model deployment and performance monitoring

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 thisis interesting because it can be seen as an attempt to formalize the concept of **intelligence**.

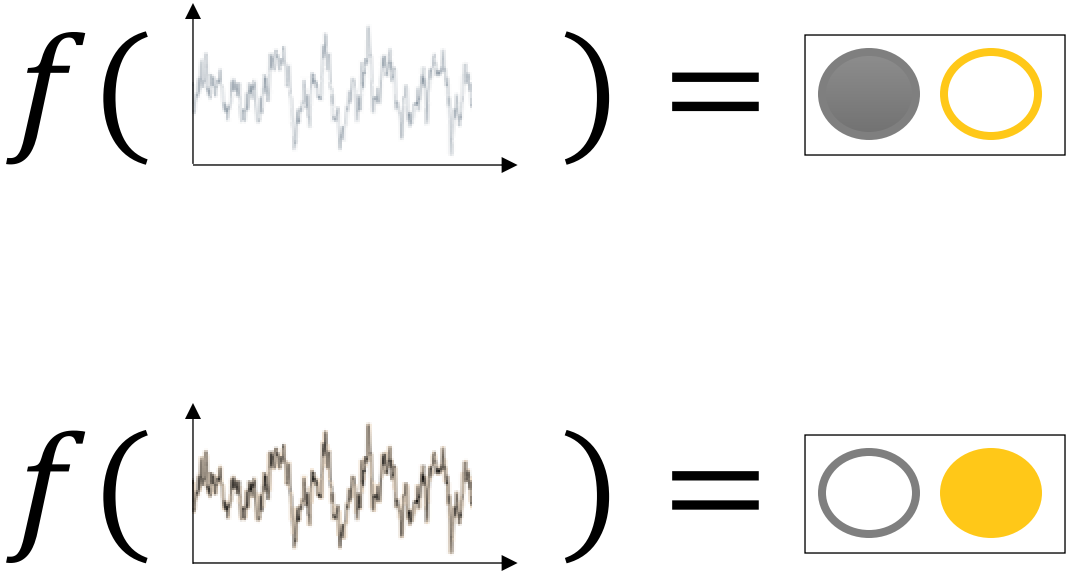
ML usually describes how machines should process **examples.**

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

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).



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

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

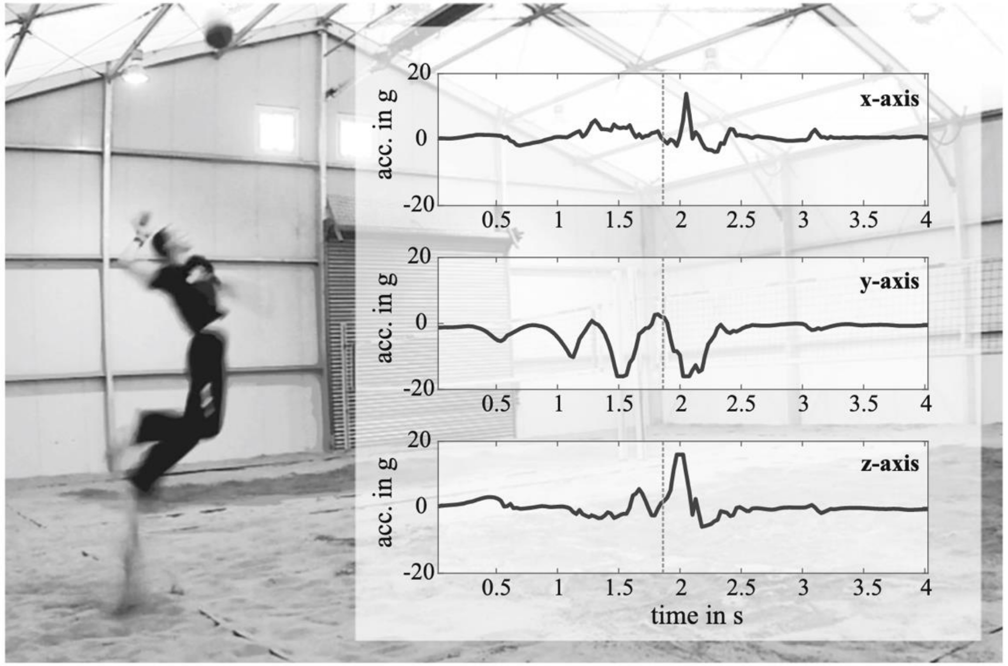
Example of time series classification

➢ 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.

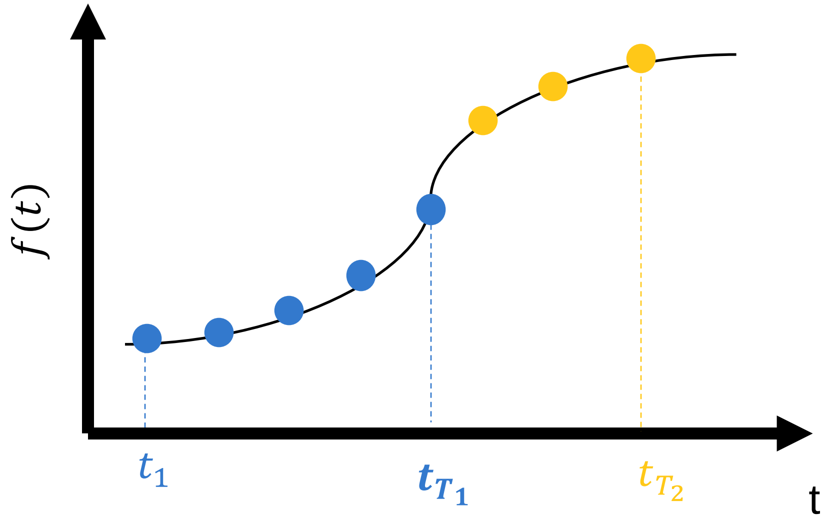


Time series forecasting

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.,

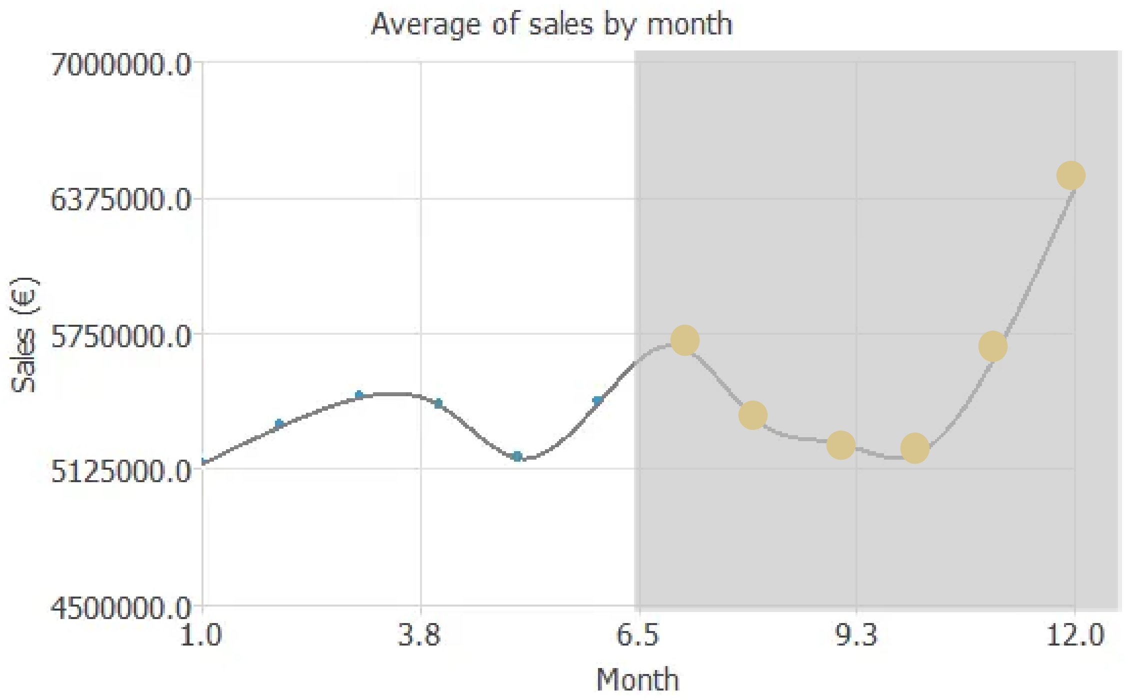
$$f(s_1, \dots, s_{T_1}) = (s_{T_1+1}, \dots, s_{T_2})$$



Example of time series forecasting:
Forecasting Monthly Sales for a Retail Store

Scenario: A retail store wants to predict future sales to optimize inventory management, staffing, and promotions. The data available consists of monthly sales figures for the last 3 years.

Objective: Predict sales for the next 6 months.



● = Collected data

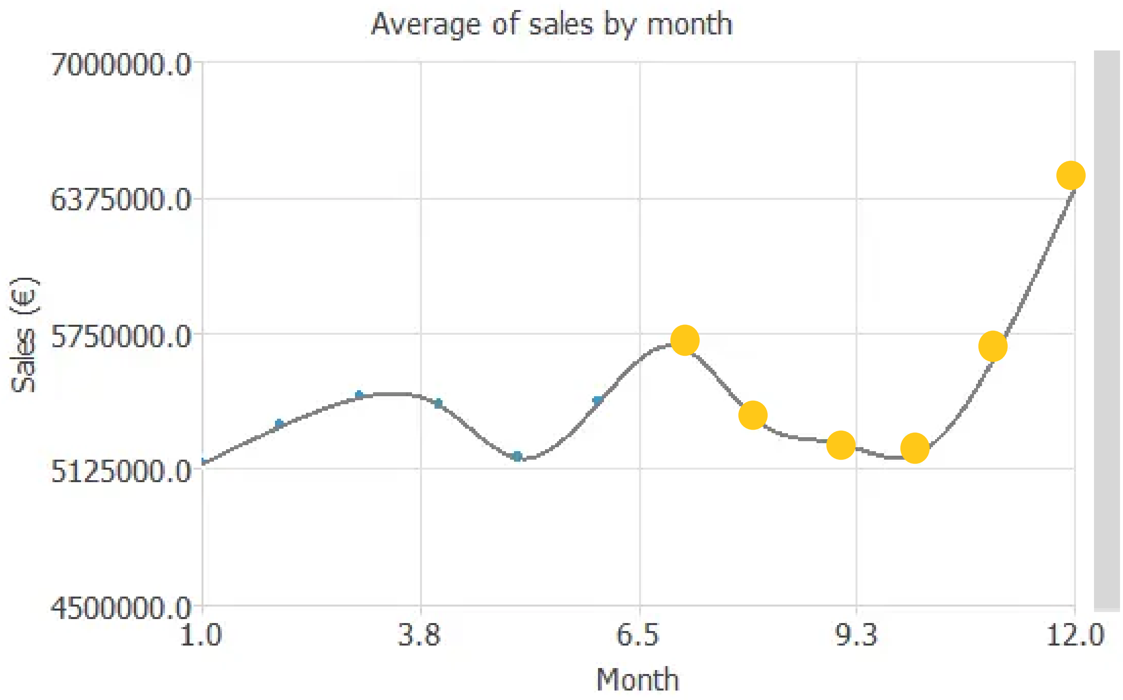
● = Predicted by a model

Full example here: https://www.neuraldesigner.com/blog/retail-store-sales-forecasting/

Example of time series forecasting:
Forecasting Monthly Sales for a Retail Store

Scenario: A retail store wants to predict future sales to optimize inventory management, staffing, and promotions. The data available consists of monthly sales figures for the last 3 years.

Objective: Predict sales for the next 6 months.



● = Collected data

● = Predicted by a model

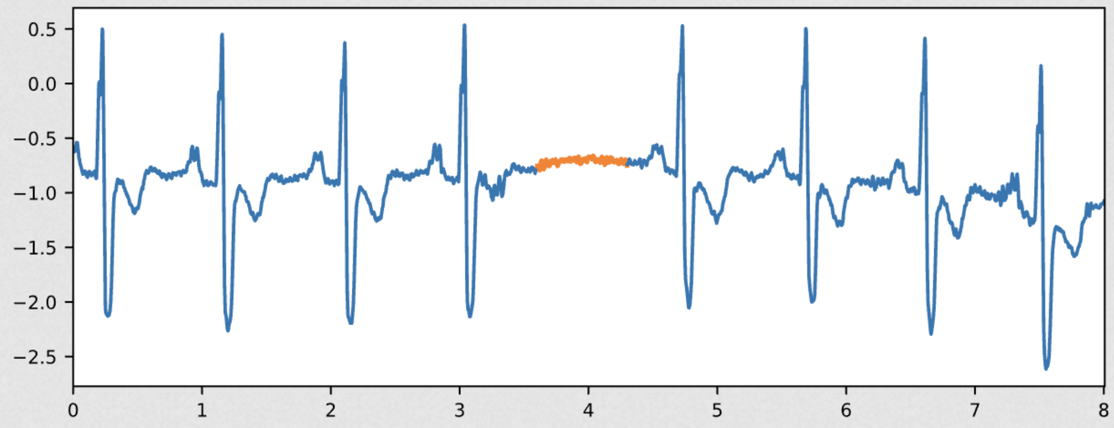
Anomaly detection on time series

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.

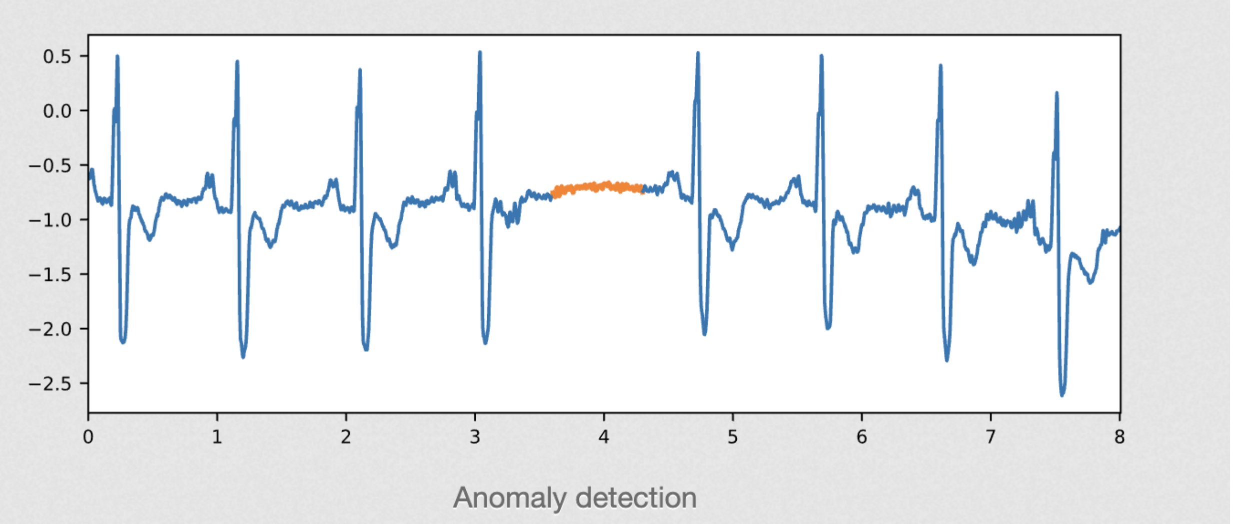


Anomaly detection

[a] https://siebert-julien.github.io/time-series-analysis-python/

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]



[a] https://siebert-julien.github.io/time-series-analysis-python/

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

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 $c_i \in \{c_1, \dots, c_M\}$.

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.

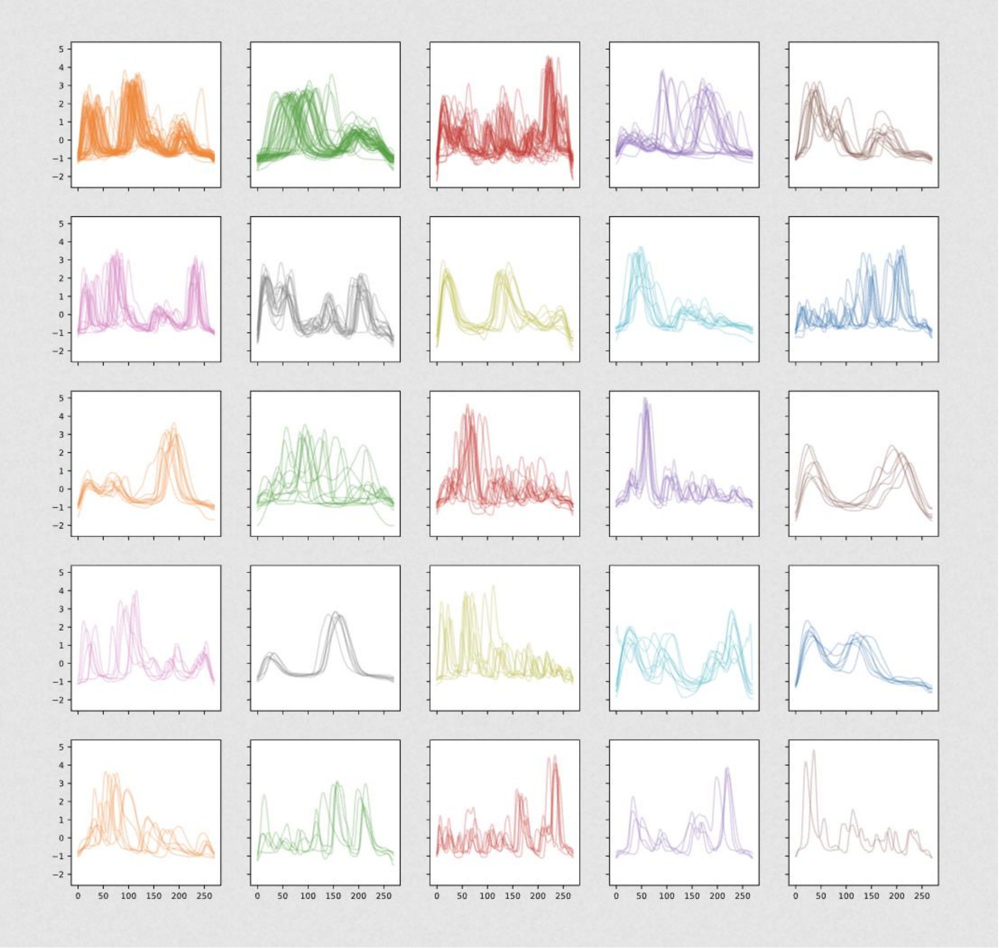
https://link.springer.com/chapter/10.1007/978-981-13-9361-7_9

Time series clustering

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!

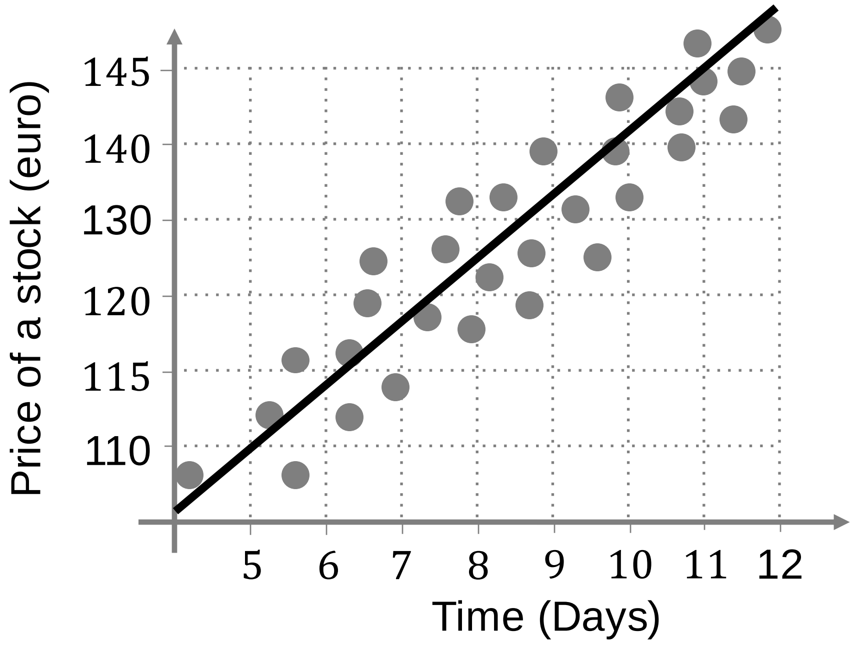
The linear model:

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

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

$\mathbf{y} = \text{Stock price}$

$\mathbf{w} = \text{Weights}$



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

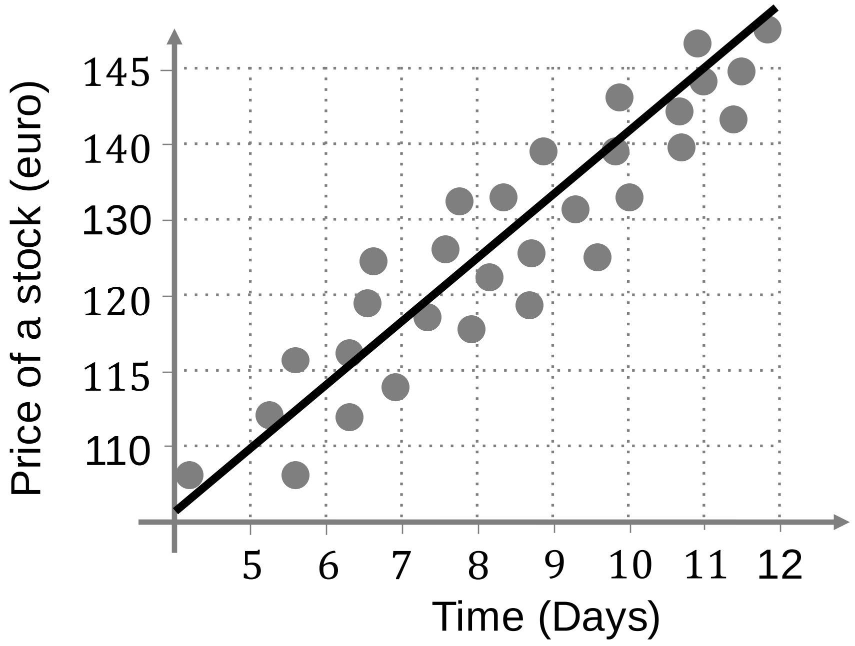
The linear model:

$$\mathrm{f}(\mathbf{x}) = \mathrm{w}_0 \cdot x_0 + \mathrm{w}_1 \cdot x_1 = \mathbf{y}$$

The linear model (in our example):

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

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



The previous description of the linear model is not accurate!

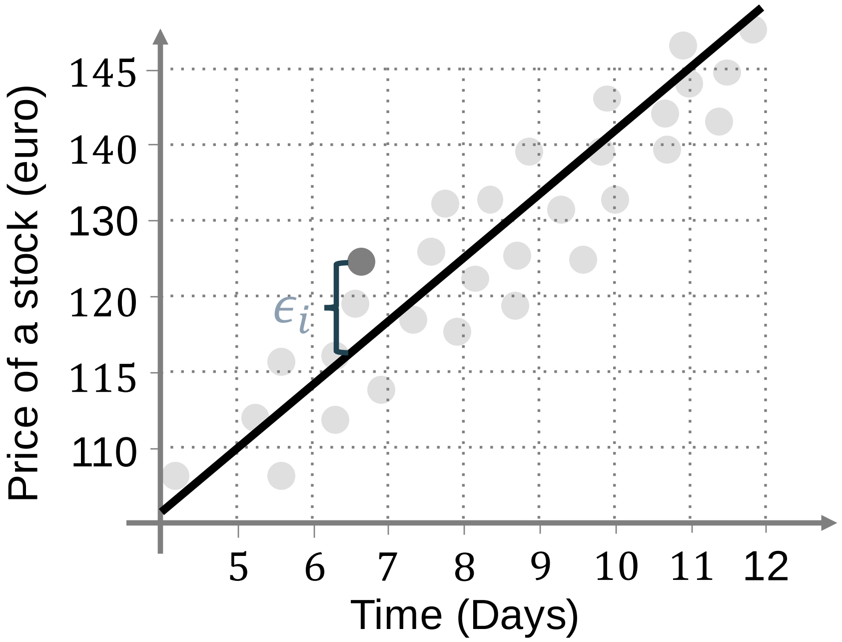
Reason: Real Systems produce noise!

Linear model with noise:

$$f(\mathbf{x}) = \mathrm{w}_0 \cdot x_0 + \mathrm{w}_1 \cdot x_1 + \epsilon_i$$

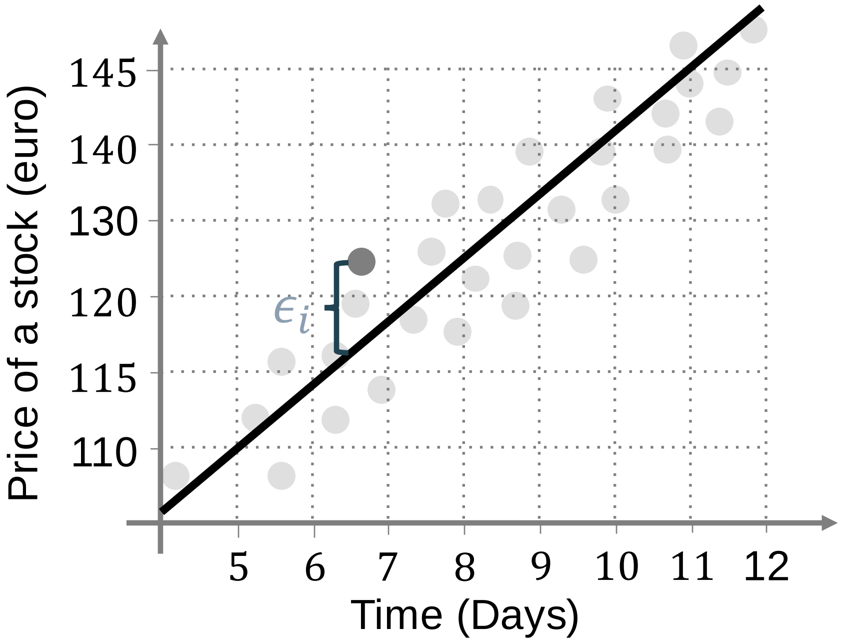
The ϵ_i and the summation $\boldsymbol{\epsilon}$ of all samples is called **Residual Error!**

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



A more general formulation:

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

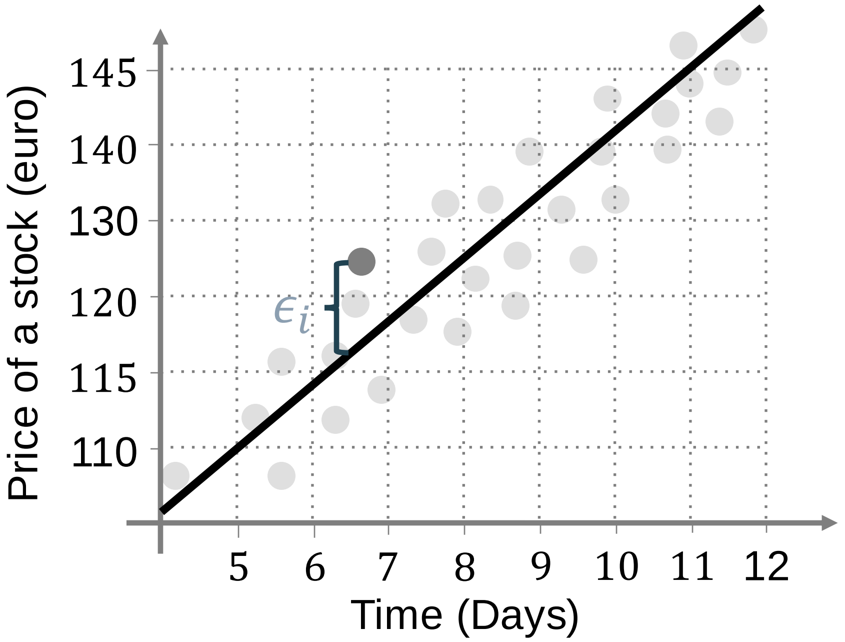


A more general formulation:

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

The **model parameters** are:

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



A more general formulation:

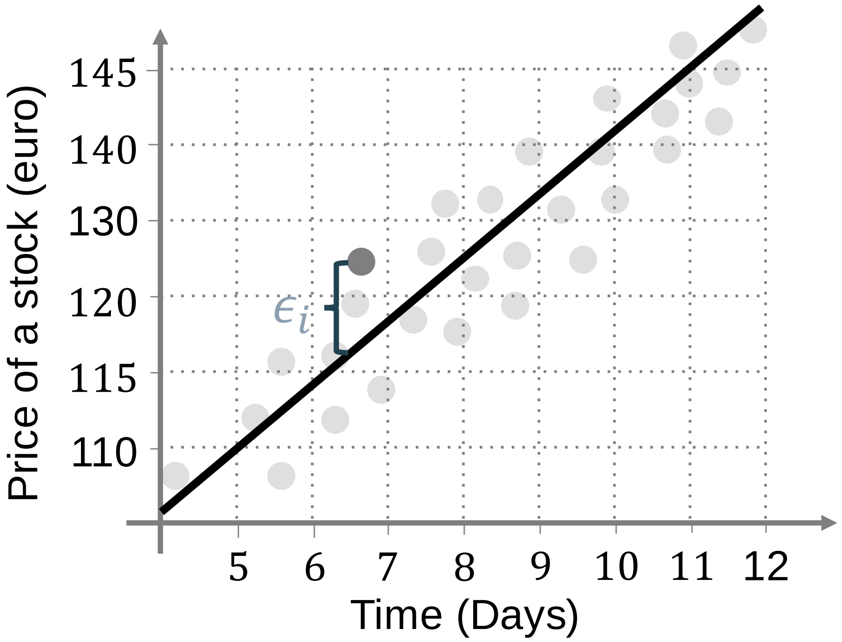
$$f(\mathbf{x}) = \sum_{j=1}^{D} w_j x_j + \epsilon = \mathbf{y}$$

The **model parameters** are:

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

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

$$\boldsymbol{\theta}^* = \underset{\boldsymbol{\theta}}{\operatorname{argmax}}\ p(\mathcal{D}|\boldsymbol{\theta})$$



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

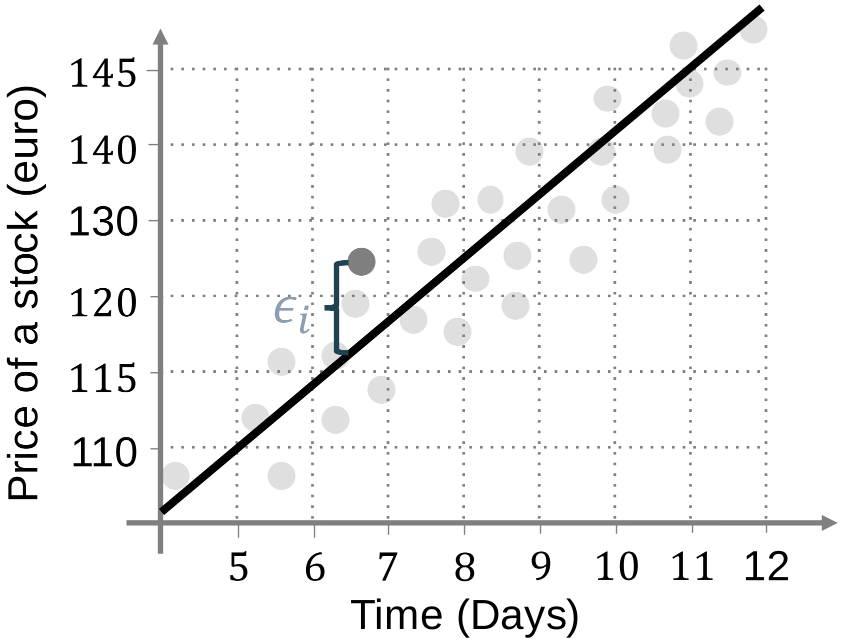
$$\mathrm{NLL}(\boldsymbol{\theta}) = \frac{\mathbf{1}}{\mathbf{2}}(\mathbf{y} - \mathbf{xw})^{\mathbf{T}}(\mathbf{y} - \mathbf{xw})$$

We find the minimum conventionally by using **the derivative**:

$$\mathrm{NLL}'(\boldsymbol{\theta}) = \mathbf{x}^{\mathrm{T}}\mathbf{xw} - \mathbf{x}^{\mathrm{T}}\mathbf{y}$$

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

$$\mathbf{w} = \left(\mathbf{x}^{\mathrm{T}}\mathbf{x}\right)^{-1}\mathbf{x}^{\mathrm{T}}\mathbf{y}$$



Time series fundamentals

Recap

Recap

Types of machine learning

- Supervised learning
- Unsupervised learning
- Reinforcement learning

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