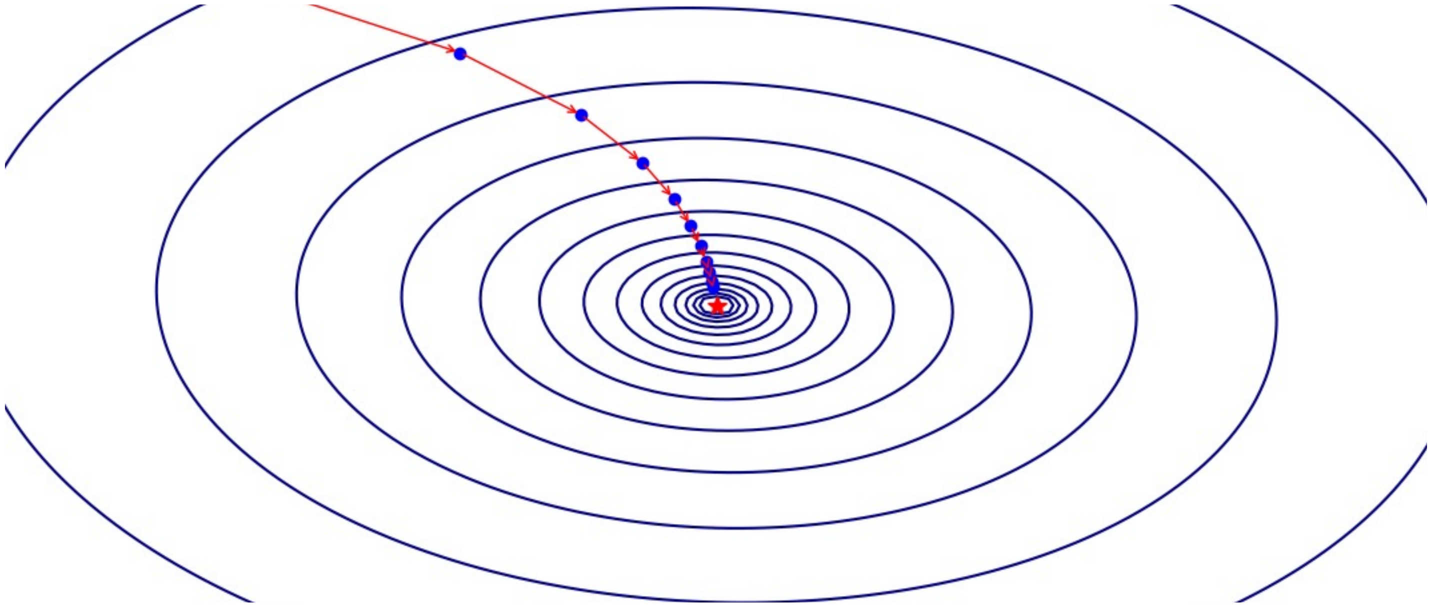


# Machine Learning in Signal Processing

Winter Semester 2023/24

2. Basics and Terminology

24.10.2023

Prof. Dr. Vasileios Belagiannis

Chair of Multimedia Communications and Signal Processing

Friedrich-Alexander-Universität
Technische Fakultät

# Course Topics

1. Introduction.
2. **Basics and terminology.**
3. Linear regression.
4. Linear classification.
5. Performance evaluation.
6. Neural networks.
7. Deep neural networks.
8. Decision trees.
9. Ensemble models.
10. Random forests.
11. Clustering / Unsupervised learning.
12. Dimensionality reduction.
13. Support vector machines.
14. Recap and Q&A.

- The exam will be written.
- We will have an exam preparation test before the end of the year.

Acknowledgements

Ideas and inspiration from:

- CSC311 Introduction to Machine Learning, University of Toronto.
- Introduction to Machine Learning: LMU Munich.
- Introduction to Machine Learning, CSAIL, MIT.
- CSE 574 Introduction to Machine Learning, University of Buffalo.
- Special thanks Arij Bouazizi, Julia Hornauer, Julian Wiederer, Adrian Holzbock and Youssef Dawoud for contributing to the lecture preparation.

## Last Lecture Recap

- Machine Learning Milestones.
  - Least Squares ( Adrien-Marie Legendre, 1805).
  - Bayes' Theorem (1812).
  - Graphical Models (1921).
  - McCulloch-Pitts Neuron (1943).
  - Perceptron (1958).
  - Reinforcement Learning (1989).
  - Random Forest Algorithm (1995).
  - Support Vector Machines (1995).
  - Deep Learning Revolution (2012).

## Today’s Agenda and Objectives

- Define machine learning.
- Understand the K-Nearest Neighbours algorithm.
- Study different types of supervision.
- Discuss the optimisation in machine learning.
- Gradient descent variants.
- Model design.

# Machine Learning

- Definition by Thomas Mitchell[*]:
  - “A computer program is said to learn from experience $E$ with respect to some class of tasks $\mathrm{T}$ and performance measure $\mathrm{P}$, if its performance at tasks in $\mathrm{T}$, as measured by $P$, improves with experience $E$.”
- What does the computer program represent?
- What is the experience?
- What is the task?
- What is the performance measure?
- How does the computer program learn?

*Thomas Mitchell. Machine learning. McGraw-Hill Education, 1997.

## Common Tasks (T)

- Classification
  - A function $f: R^n \rightarrow 1, \dots, K$ that maps a sample to $K$ categories, represented by a  set of n-dimensional features, to a categorical output.
  - The numeric code $\mathrm{y} = f(x)$ is the category prediction for the input $x$.
  - The output can be described on different ways, e.g. probabilistic.
- Regression
  - A function $f: R^n \rightarrow R$ that predicts a numerical value.
  - *Can we predict more than one numerical values?*
- More tasks: Clustering, Machine translation (e.g. English to German), structured prediction  (e.g. image captioning), denoising (i.e. recover the signal from a set  of corrupted measurements).

*Thomas Mitchell. Machine learning. McGraw-Hill Education, 1997.

## Performance Measure (P)

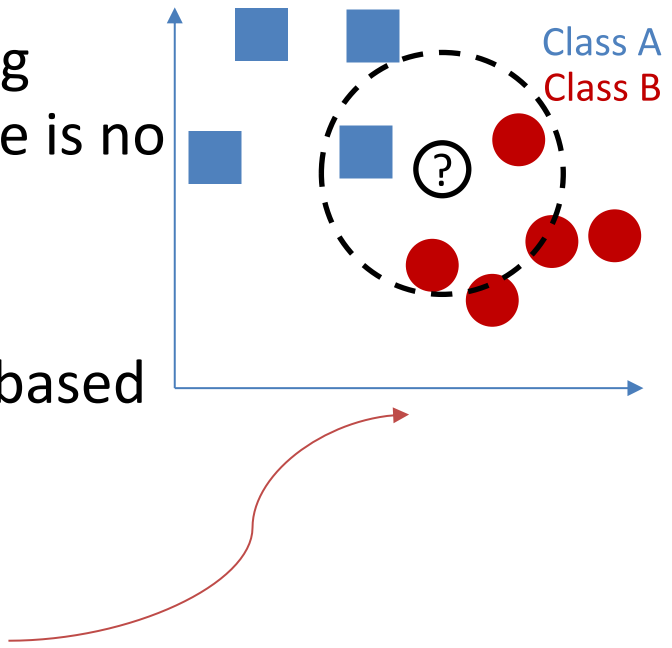
- It refers to the evaluation of machine learning algorithms.
- Requirement: a quantitative measure of the ability to perform a task.
- Same metrics among different machine  learning algorithms. It helps to compare algorithms.
- Task-related performance measure.
  - Classification: Accuracy, precision,  recall, sensitivity, specificity, F1-score.
  - Regression: mean absolute error  (MAE), root mean square Error (RMSE), MSE, p-norm.
  - For some tasks it is not clear what to  measure (e.g. image or audio generation).

## Experience (E)

- Due to the complexity, it is currently unlikely to experience the whole world using a computer program. For that reason, machine learning algorithms experience datasets.
- Dataset :
  - A dataset is a collection of samples. It is divided into training and test sets. We typically assume that all samples are independent from each other; and the training and test set are identically distributed (i.i.d assumption). Additionally, all data are drawn from the same probability distribution.

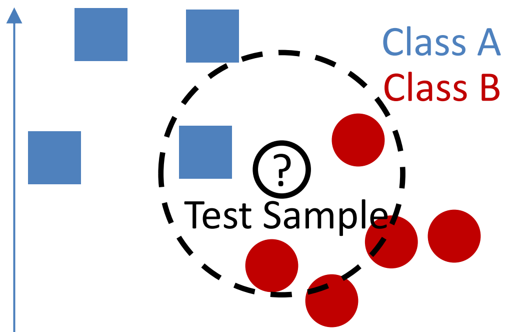
# K-Nearest Neighbours Algorithm

- K-NN is a simple machine learning algorithm that we use for classification or regression.
- K-NN relies exclusively on the training samples for prediction and thus there is no training process.
- Consider the problem of classifying whether a property worth investing based on the construction year and square meters of the property.
  - 2-class problem with features 2.
  - Access to a training set with N samples.



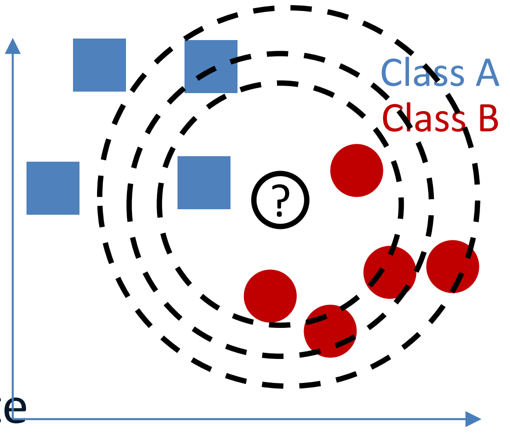
# K-Nearest Neighbours Algorithm

- The algorithm can be summarised as:
  1. Calculate the <u>distance</u> between the reference test sample and the training set.
  2. <u>Sort</u> the training set based on the distance.
  3. <u>Pick</u> the first k samples from the training set.
  - If regression, <u>average</u> the labels of the k samples.
  - If classification vote for each class and get the <u>mode</u> of the k labels.
- *What is the performance evaluation metric for our 2-class example?*
- *How does the prediction scale w.r.t the training set?*



## Algorithm Properties

- K is often an odd number.
  - There is no optimal way to choose it.
- Common distance measures:
  - Euclidean distance, Hamming distance, Manhattan distance or Minkowski distance.
- K-Nearest Neighbours is a lazy learning approach.
  - There is no necessary to learn/train the model. It will wait for a query sample and then compute its output based on the whole training.
  - On the other hand a eager learning approach would learn to generalise from the training set. It would then construct a model that is not depended on the training set for making a decision.



Python example and reference: https://www.datacamp.com/tutorial/k-nearest-neighbor-classification-scikit-learn

# Algorithm Properties (Cont.)

- Course of dimensionality.
  - The algorithm works reasonably well for features of low dimensionality.
  - High dimensional data may lead to overfit (will be later discussed) on the training set unless there is a large number of sample.
- To prevent the problem, one can reduce the feature dimensions with principal component analysis (will be later discussed).
- Our goal is learn to learn and not store all training samples / features.

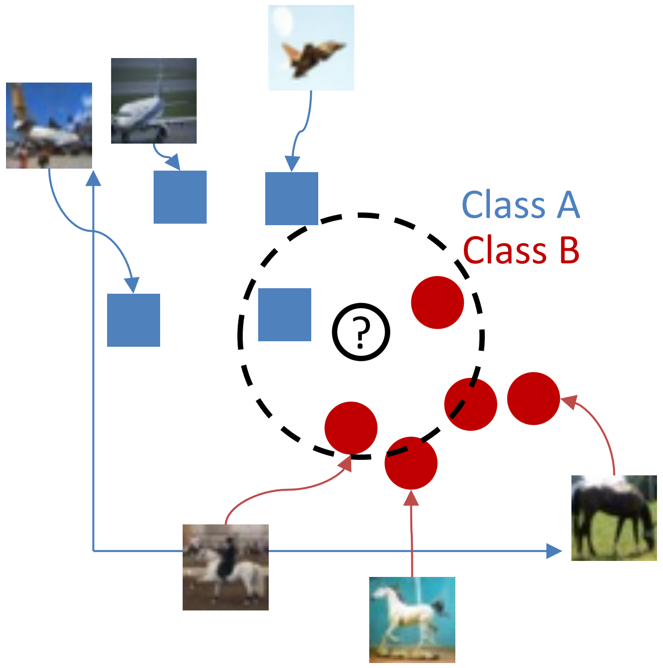


Image Source: https://www.cs.toronto.edu/~kriz/cifar.html (CIFAR* dataset).

*Krizhevsky, Alex, and Geoffrey Hinton. "Learning multiple layers of features from tiny images." (2009): 7.

## Learning

- The way of experiencing a dataset is connected to the type of learning.
- Common learning categories: supervised learning, unsupervised learning, weakly-supervised learning, semi-supervised learning, self-supervised learning, reinforcement learning.
- Supervised learning $\rightarrow D = \{\{\boldsymbol{x_1}, \boldsymbol{y_1}\}, \dots, \{\boldsymbol{x_n}, \boldsymbol{y_n}\}\}$, where we have access to pairs of input features $\boldsymbol{x_i} \in \mathbb{R}$ and targets $\boldsymbol{y_i}$, indexed by $i$.
  - The input features can be the image pixels, while the target or label $\boldsymbol{y}$ can vary from class category to scalar, vector and tensor output.
  - In general, the input data space $\mathcal{X}$ for $\boldsymbol{x_i} \in \mathcal{X}$ and output target space $\mathcal{Y}$ for $\boldsymbol{y_i} \in \mathcal{Y}$ varies with respect to the application.
  - During training, the algorithm learns to predict $\boldsymbol{y}$ from $\mathbf{x}$, as $p(\boldsymbol{y}|\boldsymbol{x})$.
  - Supervised learning is the most frequent learning approach.

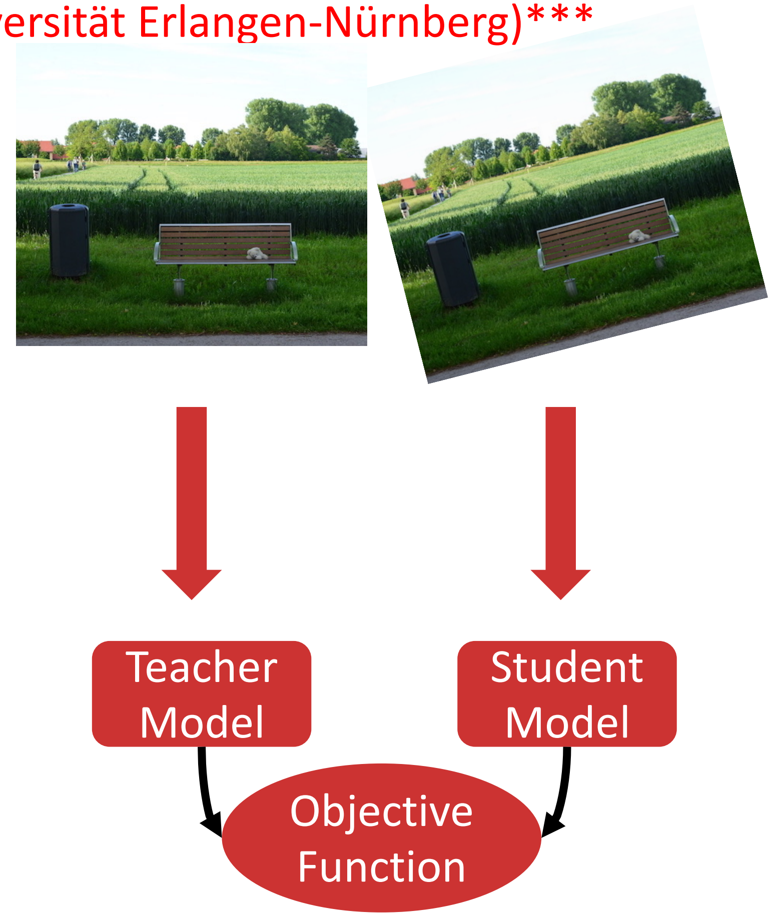
## Learning (Cont.)

- Unsupervised learning $\rightarrow$ learn the data probability distribution $p(\boldsymbol{x})$ on implicit or explicit way.
  - There is not any label $\boldsymbol{y_i}$ available.
  - Density estimation: construct an estimate of the probability density function from the available training data $D$.
  - The algorithm has to understand the data without guidance (supervision).
- Weakly-supervised learning $\rightarrow$ The labels of the training set $\boldsymbol{y_i} \in \mathcal{Y}$ are not available for the training set due to the annotation complexity. Instead the training set has "cheaper" (i.e. easier produced) labels $\boldsymbol{y_i} \in \mathcal{Y}$ for each input for $\boldsymbol{x_i} \in \mathcal{X}$.
  - Example: image-based object detection based on training samples that provide only the object category annotation, not the location within the image*.

*Hakan Bilen and Andrea Vedaldi. Weakly supervised deep detection networks. CVPR 2016.

# Learning (Cont.)

- Semi-supervised learning → $D_1 = \{\{\boldsymbol{x_1}, \boldsymbol{y_1}\}, \dots, \{\boldsymbol{x_k}, \boldsymbol{y_k}\}\}$ where $k \ll n$. There is a handful amount of labels data. In addition, there is unlabelled data → $D_2 = \{\{\boldsymbol{x_1}\}, \dots, \{\boldsymbol{x_n}\}\}$.
  - Object detection with limited annotated bounding boxes*.
  - Book to study: *Chapelle, Olivier, Bernhard Schölkopf, and Alexander Zien. "Semi-supervised learning. Adaptive computation and machine learning series." (2006).*



*Xu, Mengde, et al. "End-to-end semi-supervised object detection with soft teacher." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021.

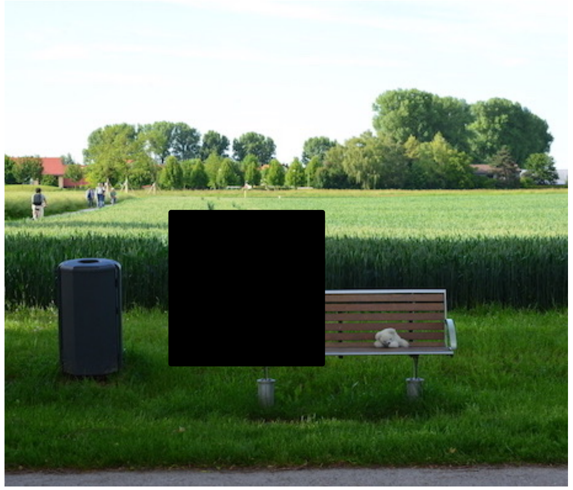
## Learning (Cont.)

- Self-supervised learning → $D = \{\{\boldsymbol{x}_1\}, \ldots, \{\boldsymbol{x}_n\}\}$. The data acts as supervision and proxy tasks are used for representation learning.
  - Common in the language models, e.g. provide part of a sentence and predict the rest.
  - It is usually combined with supervised learning. First, we learn an intermediate representation with self-supervision and then further train the model with supervised learning (current state of the art recipe in computer vision tasks).
  - Examples: Predict relative position of images patches[*]. Predict image rotations[**].
- *What other image operations can be used for self-supervision?*

*Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. ICCV 2015.
**Gidaris, Spyros, Praveer Singh, and Nikos Komodakis. "Unsupervised representation learning by predicting image rotations.", ICLR 2018
Online reference: https://lilianweng.github.io/posts/2019-11-10-self-supervised/

# Learning (Cont.)

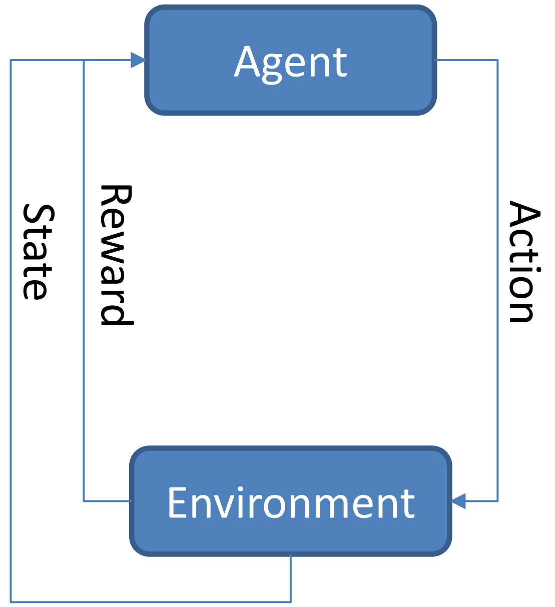
- Self-supervised learning $\rightarrow D = \{\{\boldsymbol{x_1}\}, \ldots, \{\boldsymbol{x_n}\}\}$. The data acts as supervision and proxy tasks are used for representation learning.
  - It is developed for deep neural networks.
  - Image[*] and text[**] masking is a common approach of self-supervision.



*He, Kaiming, et al. "Masked autoencoders are scalable vision learners." Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. 2022.
**Devlin, J., et al. "Bert: Pre-training of deep bidirectional transformers for language understanding In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, 4171–4186.. ACL." ACL 2019

## Learning (Cont.)

- <u>Reinforcement Learning</u>: More than experiencing a fixed dataset $D$.
  - There is an environment where the interaction with it creates experience.
  - A simulator is necessary for the learning process, e.g. CARLA* simulator. The learning dataset $D$ is synthesized at every episode of training.
  - Policy Learning: mapping from a state to an action.
  - Common for planning problems, e.g. automated driving trajectory planning,



*Dosovitskiy, Alexey, et al. "CARLA: An open urban driving simulator." *Conference on robot learning*. PMLR, 2017.

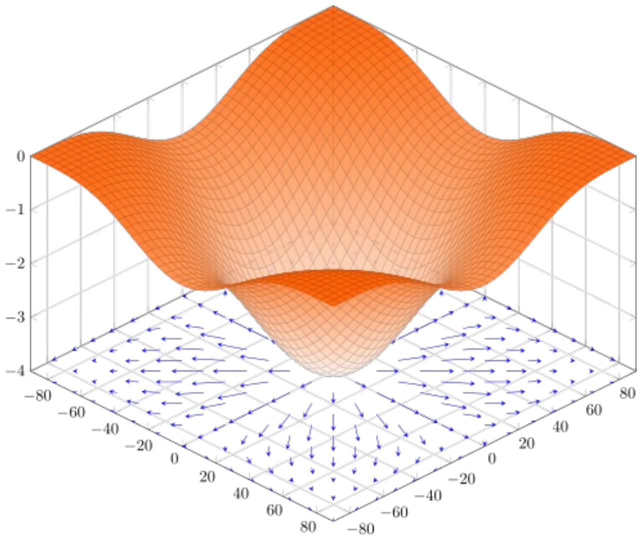
# Optimization

- Objective function: Our machine learning algorithm has several unknown parameters. To estimate them, we define an objective function that its output is minimised (or maximised). This is the optimisation procedure.
  - Most of the time, we will have a loss function to minimise.
  - We look for the parameter values that minimise the loss function.
  - We mostly focus on continuous function minimisation (optimization), i.e., real-valued model parameters and real-valued loss function.
  - In case of discrete values for our parameters, we refer to combinatorial optimization.
- *How do we group the continuous function minimisation methods?*

Online reference https://machinelearningmastery.com/tour-of-optimization-algorithms/

# Optimization (Cont.)

- A common grouping of the optimisation algorithm is based on whether the objective function is differentiable*.
- Differentiable objective function.
  - Gradient-based optimisation: first- and second-order methods.
  - First order is gradient descent, while second order is Newton's method or a Quasi-Newton method, e.g. Broyden-Fletcher-Goldfarb-Shanno (BFGS).
- Non-differential objective function.
  - Nelder–Mead method, cyclic coordinate descent, evolution strategy, genetic algorithm and particle swarm optimization.

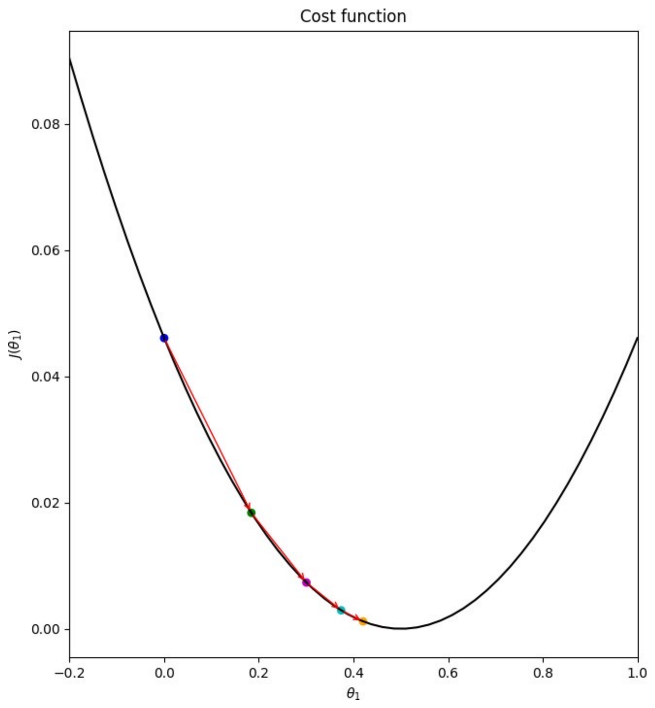


By MartinThoma - Own work, CC0, https://commons.wikimedia.org/w/index.php?curid=71375503

*Kochenderfer, Mykel J., and Tim A. Wheeler. Algorithms for optimization. Mit Press, 2019.

# Gradient-Based Optimization

- Gradient-based optimization is a common way of looking for the minimum or maximum of a function.
- We will mostly rely on gradient-based optimization for finding the values of the parameter that minimize the loss function of our machine learning program.
- The cost function $f(\theta)$ has a <u>global minimum</u>. Given an initial search point, i.e. an initial set of values for $\theta$ the minimum can be found using gradient-based optimization.
- *What is the difference between global and local minimum / maximum?*



# Gradient-Based Optimization: Gradient Descent

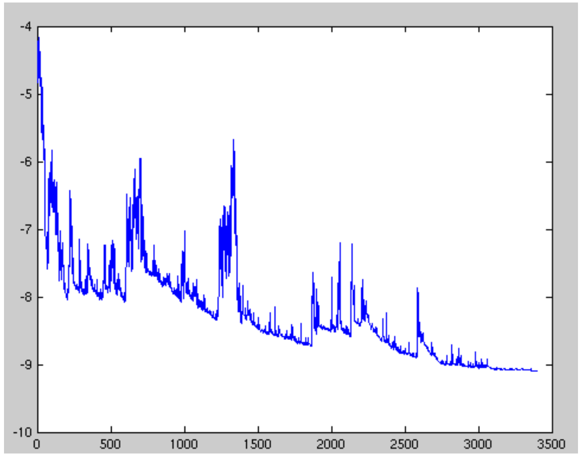
- It is an <u>iterative optimization algorithm to find a function's minimum</u>. The function can represent, for instance, a regression or classification problem.
  - Given the loss function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ that is parametrized by $\boldsymbol{w}$, the algorithm iteratively updates the parameters in the opposite direction of the gradient of the loss function $\nabla\boldsymbol{w}\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ .
  - The learning rate $\eta$ influences how much an update will modify the parameters.
  - The goal of the algorithm is to reach the global minimum of the loss function, or the local minimum when the loss is non-convex.
- *What are the two common ways to compute gradients?*
- There are <u>three</u> main variants of gradient descent based on the way we handle the data.

## Gradient Descent: Batch Gradient Descent

- This the standard (vanilla) gradient descent where the gradient of the loss $\mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$ w.r.t. the parameters $\boldsymbol{w}$ is computed on the whole training set.
- It is defined as: $\boldsymbol{w} = \boldsymbol{w} - \eta \nabla \boldsymbol{w} \mathcal{L}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{w})$.
- When the dataset is large, an update can be slow.
- For datasets that do not fit into the memory, it will not be possible to perform an update at all.
- The batch gradient descent will converge to the global minimum after a sufficient number of iterations.
- For non-convex, it will reach a local minimum.
- *Which machine learning algorithm is highly non-convex but works well with local minimum solutions?*

# Gradient Descent: Stochastic Gradient Descent

- The parameter update is performed for <u>each training sample separately</u>.
- It can be written as:
  - $\boldsymbol{w} = \boldsymbol{w} - \eta \nabla \boldsymbol{w} \mathcal{L}(\boldsymbol{x}_i, \boldsymbol{y}_i, \boldsymbol{w})$.
- The update is fast, but with high variance. It can also be used for on-line training.
- The main limitation is the <u>fluctuations</u>, which are caused due to the lack of gradient averaging during the update step.
- Stochastic gradient descent (SGD) can reach a global or local minimum when lowering accordingly the learning rate and iterating <u>longer</u> than with the batch gradient descent.



By Joe pharos at the English-language Wikipedia, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=42498187

## Gradient Descent: Mini-batch Gradient Descent

- The third variant of gradient descent is the best compromise between efficient and stable training.
- It performs a gradient update for every mini-batch of $n$ samples:
  - $\boldsymbol{w} = \boldsymbol{w} - \eta \nabla_{\boldsymbol{w}} \mathcal{L}(\boldsymbol{x}_{i:i+n}, \boldsymbol{y}_{i:i+n}, \boldsymbol{w})$.
- Mini-batch gradient descent offers:
  - Smaller variance of the parameter updates compared to SGD and thus more stable convergence.
  - Mini-batch gradients can be efficiently computed. As a result, it comparably fast to SGD.
- When one refers to SGD, it usually implies the mini-batch version.

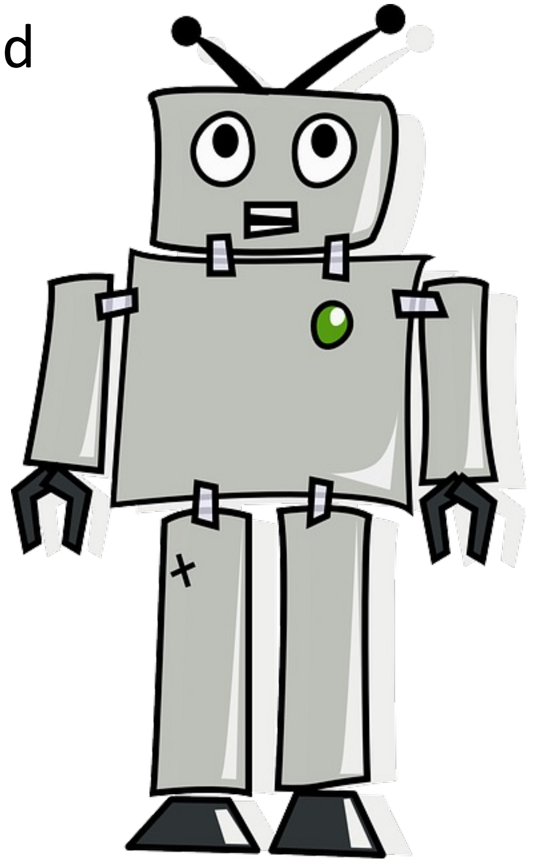
## Gradient Descent Challenges

- The gradient descent may result in bad convergence due to:
  - Fixed learning rate → A low learning rate results in slow convergence, while a large one will result in fluctuations around the minimum and lack of convergence.
  - Adaptive learning rate → It can decrease based on some rule. However, it's not straight forward which technique is the right one[*].
  - Saddle points → A point where the slopes (gradients) in orthogonal directions are all close to or zero, but a local minimum. The gradient descent can get stuck in these kinds of critical points[**].
  - Multiple learning rates → All parameters do not adapt in the same way. However, the learning is the same for all of them. Using different learning rates per parameter can improve the convergence but it increases the problem complexity.

*Christian Darken, Joseph Chang, and John Moody. Learning rate schedules for faster stochastic gradient search. In Neural Networks for Signal Processing II Proceedings of the 1992 IEEE Workshop, pages 3–12. IEEE, 1992.
**Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. Neurips 2014.

# Building a Machine Learning Model

- To design an algorithm, we normally need to consider the following parts:
  - Dataset, data pre-processing and representation.
  - Objective function / Type of supervision.
  - Machine Learning Algorithm.
  - Optimisation.
- We will study different approaches to address each step.
- At the moment deep neural networks dominate within the field.
- *What is the a common data pre-processing technique?*



Source: https://pixabay.com/de/vectors/roboter-android-148989/

## Study Material

- *Chapter 1, Shalev-Shwartz, Shai, and Shai Ben-David. Understanding machine learning: From theory to algorithms. Cambridge university press, 2014.*

# Next Lecture

Linear Regression