Deep Learning:

Introduction & Background

Robert Legenstein & Ozan Özdenizci
Institute of Machine Learning and Neural Computation
robert.legenstein@tugraz.at
oezdenizci@tugraz.at

Deep Learning VO - WS 25/26
Lecture 1 - October 6th, 2025

Course Organization

- **Lecturers:** Robert Legenstein (VO), Ozan Özdenizci (VO), Simon Hitzginger (KU)
  - **VO** (DAT.C301UF) & **KU** (DAT.C302UF)
  - **KU:** theoretical & practical examples in sync with VO - attend both classes if possible
- **Course Location & Time**: HS i12 on Mondays @ 16:15 - 17:45
- **KU Start:** 22.10.2025
- **TeachCenter**: https://tc.tugraz.at/main/course/view.php?id=1740

  Lectures will not be recorded. Slides will be on TeachCenter.
- **Literature**: I. Goodfellow, Y. Bengio, A. Courville: “Deep Learning”, MIT Press, 2016.

  https://www.deeplearningbook.org

More literature & resources

- C. Bishop: “Pattern Recognition and Machine Learning”, Springer Verlag, 2006.
  https://www.microsoft.com/en-us/research/people/cmbishop/prml-book/

- F. Chollet: “Deep Learning with Python”, Manning Publications, 2017.
  https://www.manning.com/books/deep-learning-with-python/

- A. Zhang, Z. Lipton, M. Li, A. Smola: “Dive into Deep Learning”, Cambridge University Press, 2023.
  https://d2l.ai/

- Simon J. D. Prince: “Understanding Deep Learning”, MIT Press, 2023.
  https://udlbook.github.io/udlbook/

Goals of the lecture & Course content

- To acquire a deep understanding of neural networks.
- Ability to apply neural networks to pattern recognition problems.
- Become familiar with main challenges in deep learning and learn practical tricks.
- Practical work with deep neural networks (covered in KU).

Topics:

- Theoretical background on machine learning
- Deep neural networks
  - Training methods, optimization
  - Regularization, generalization, …
- Convolutional neural networks
- Recurrent neural networks
- Unsupervised learning
  - Variational autoencoders
  - Generative adversarial networks
  - Denoising diffusion models, …
- Transformers

Grading in VO

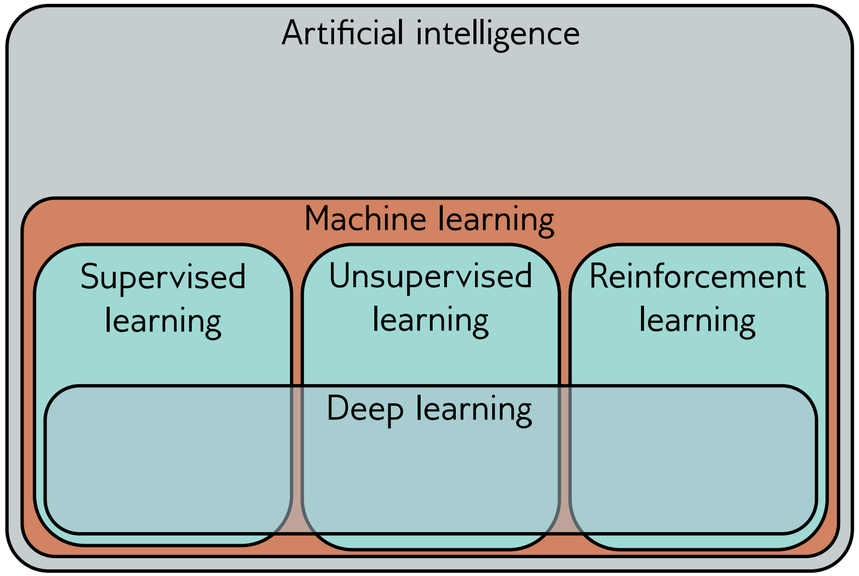
- Written exams following the semester.
  - [87.5, 100] % : 1
  - [75, 87.5) % : 2
  - [62.5, 75) % : 3
  - [50, 62.5) % : 4
  - [0, 50) % : 5
- Grading is independent from KU.

Today

- [ ] Motivation
- [ ] Pattern Recognition
- [ ] Central Concepts in Machine Learning
- [ ] Light Introduction to Neural Networks

Building Intelligent Machines

- **Artificial intelligence** (AI) is concerned with building systems that simulate intelligent behavior and perform tasks typically associated with human intelligence (e.g., reasoning, problem-solving, understanding natural language…).
- **Machine learning** (ML) is a subset of AI that learns to make decisions by fitting mathematical models to observed data.
- A deep neural network is a type of ML model, and when it is fitted to data, this is referred to as ***deep learning***.



Neural Networks

The first inspiration came from the human brain:

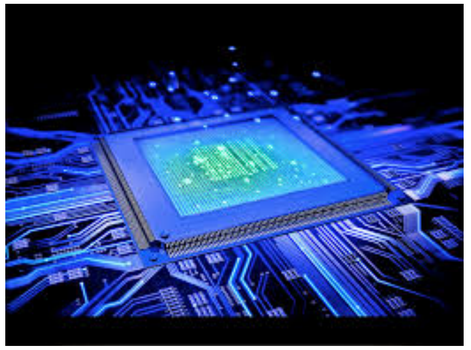
Brain

$\approx 10^{12}$ neurons

$\approx 10^{15}$ connections (synapses)

$\approx$ 4 km axons per mm^3,
parallel and distributed

Computer

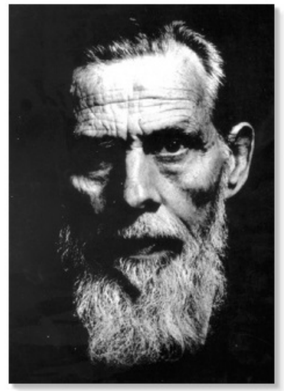


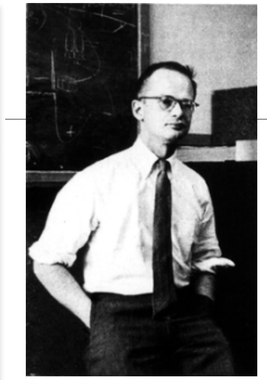
$\approx 10^{11}$ transistors

serial system with
local computations

The brain is learning from experience by changing the strength of synaptic connections between neurons.

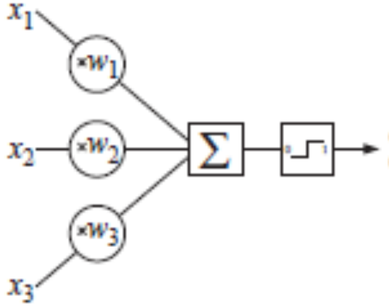
Neural Networks: A Brief History





- 1943 - McCulloch-Pitts neuron: the first model of a neuron ("Perceptron")

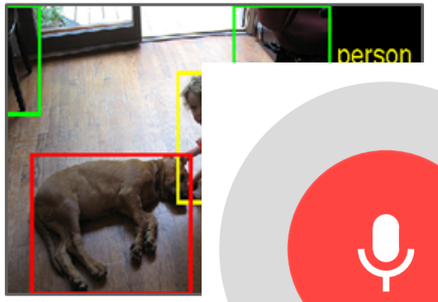
$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b > 0, \\ 0 & \text{otherwise} \end{cases}$$



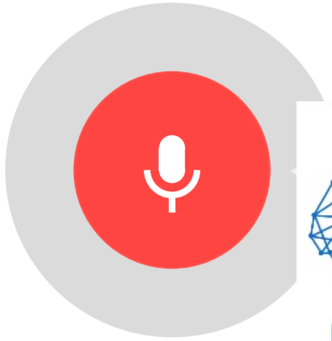
- 1958 - Rosenblatt's perceptron: The first hardware implementation of a single perceptron that can be trained.
  - *"[…] walk, talk, see, write, reproduce themselves and are conscious of their own existence."*
- 1969 - Minsky and Papert: The book "Perceptrons" calms down the euphoria.
  - *Single-layer perceptrons are not that expressive, can not implement the XOR function.*

- Early 80s - Backpropagation learning: Training a multilayer neural network.
- 90s - Kernel-based methods (support vector machines, …), graphical models, etc.
- 2006 - Deep belief networks (Hinton et al.): Successful training of deep neural networks on high-dimensional data.

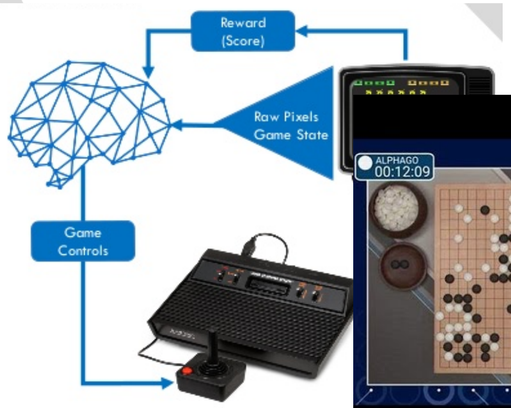
Frontiers of Machine Learning and AI



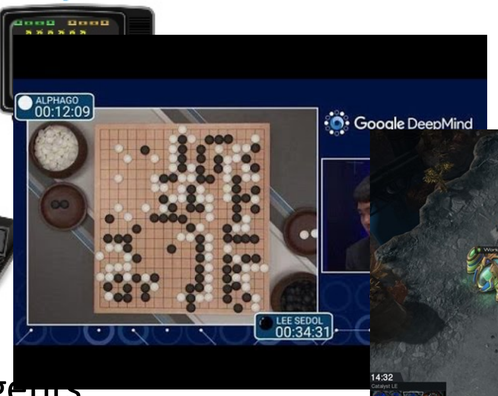
Object r
2012:



Speech-r
2012:



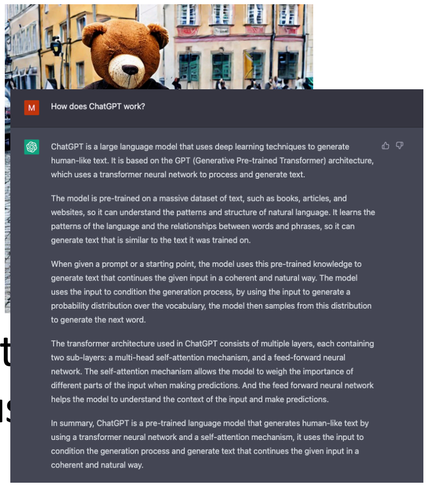
Autonomous agents
2015: Atari-games



Board games
2015/17: AlphaGo

Strategy
2019: Sta

Deep generat
2020s: Diffu



Large language models
2023: GPT-4

Why Now?

- **Larger amount of data**
  - Huge datasets & easier collection

- **Hardware**
  - Massively parallelizable Graphics Processing Units (GPUs)

NVIDIA

- **Software**
  - Improved & tailored techniques - Toolboxes, libraries

PyTorch

TensorFlow

Impact

- Ranked among the highest impact factor venues in **all of science**.

- Heavily invested by the industry.

Google Research

OpenAI

Google DeepMind

Meta AI

Microsoft

amazon | science

IBM Research

Google Scholar

Top publications

Categories

English

	Publication	h5-index	h5-median
1.	Nature	467	707
2.	The New England Journal of Medicine	439	876
3.	Science	424	665
4.	IEEE/CVF Conference on Computer Vision and Pattern Recognition	422	681
5.	The Lancet	368	688
6.	Nature Communications	349	456
7.	Advanced Materials	326	415
8.	Cell	316	503
9.	Neural Information Processing Systems	309	503
10.	International Conference on Learning Representations	303	563
11.	JAMA	286	476
12.	Science of The Total Environment	273	375
13.	Nature Medicine	268	459
14.	Proceedings of the National Academy of Sciences	268	394
15.	Angewandte Chemie International Edition	266	362
16.	Chemical Reviews	264	459
17.	International Conference on Machine Learning	254	463
18.	Chemical Society Reviews	248	390
19.	Journal of Cleaner Production	246	321
20.	Nucleic Acids Research	238	539

Today

- [x] Motivation
- [ ] Pattern Recognition
- [ ] Central Concepts in Machine Learning
- [ ] Light Introduction to Neural Networks

Pattern Recognition

*The field of **pattern recognition** is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.*

[C. Bishop, “Pattern Recognition and Machine Learning”]

- Speech recognition
- Handwritten character recognition
- Medical diagnosis, etc.

Humans can easily do these tasks but it is hard to write programs for them.

Learning Algorithms

*A computer program is said to **learn** from experience E with respect to some task T and performance measure P, if its performance at task T, as measured by P, improves with experience E.*

[T. Mitchell, "Machine Learning", McGraw-Hill (1997)]

- "**Example"**: a vector $\mathbf{x} \in \mathbb{R}^D$ of **features** x_i that have been measured from some object or event that we want the machine learning system to process.
- "**Experience"**: A collection of examples (dataset) $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(N)}\}$
- **Unsupervised learning**: Learn useful properties of the dataset.
  - Clustering, dimensionality reduction (PCA, ICA, …), generative modeling (representing distribution of examples), …
- **Supervised learning**: Learn to predict the **label** or **target** $t^{(i)}$ associated with each **input** $\mathbf{x}^{(i)}$.

Classification Regression

Classification Tasks

- Compute, to which of K categories (classes) some example $\boldsymbol{x}$ belongs to.
- The learning algorithm should produce a function $f : \mathbb{R}^D \rightarrow \{0, \ldots, K-1\}$.
- Usually a parametrized function $y = f_W(x)$ with parameters W.

- Learning is then the process of finding parameters W that improve performance.
- Often, the learning algorithm produces a distribution over outputs.

"dog" vs "cat"

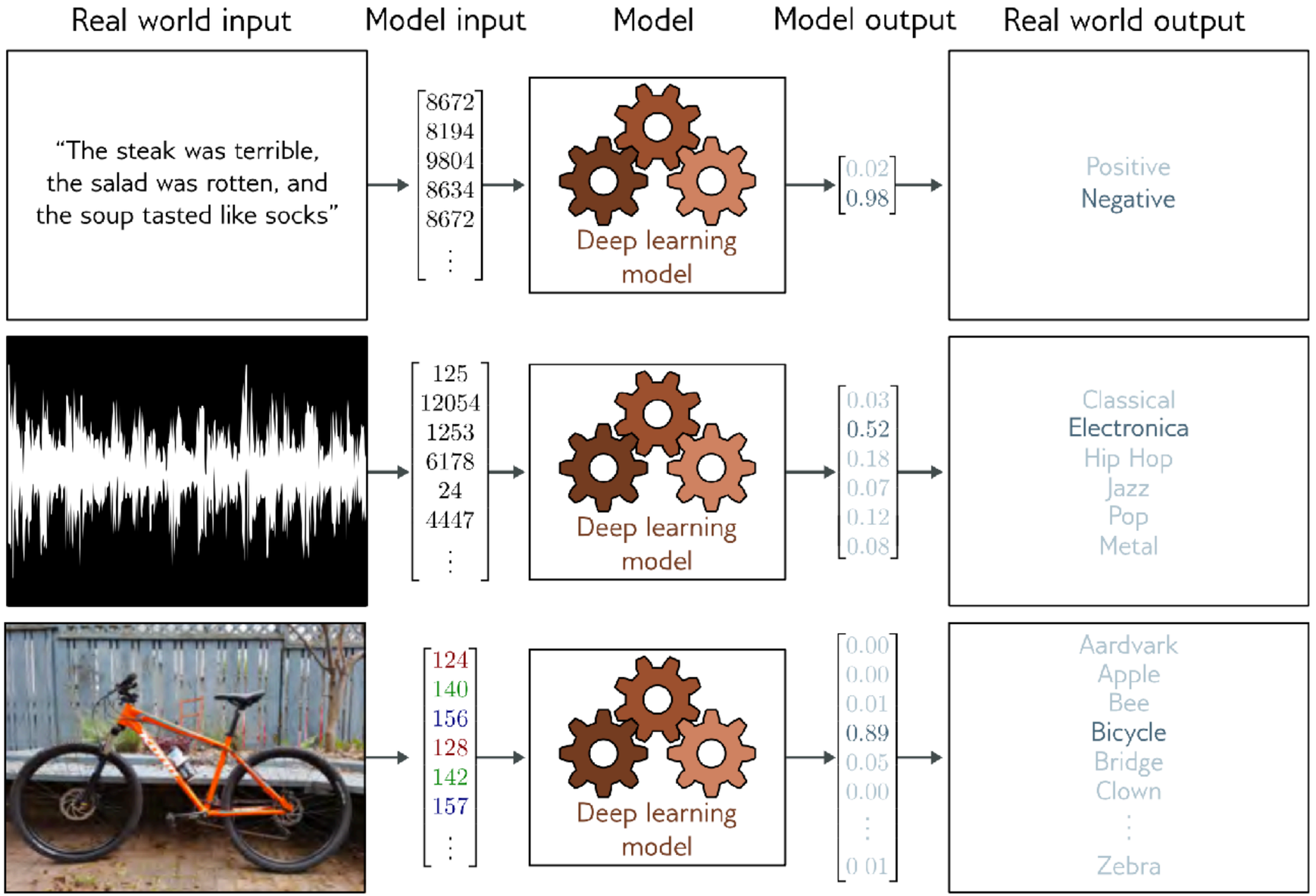
C_1

C_2

$P(cat \mid \mathbf{x}^{(i)}) = 0.8$

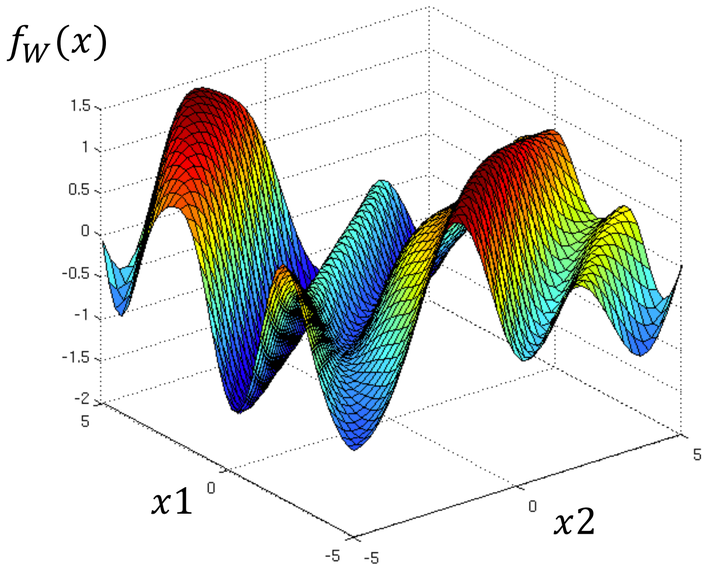
$P(dog \mid \mathbf{x}^{(i)}) = 0.2$

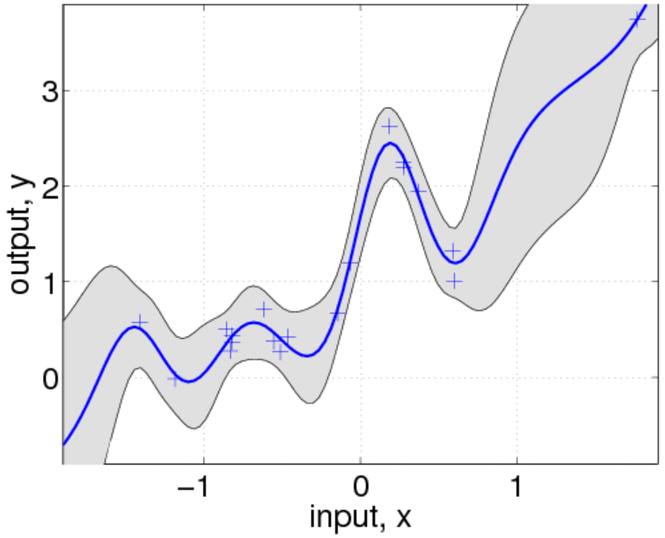
Classification Tasks - Examples



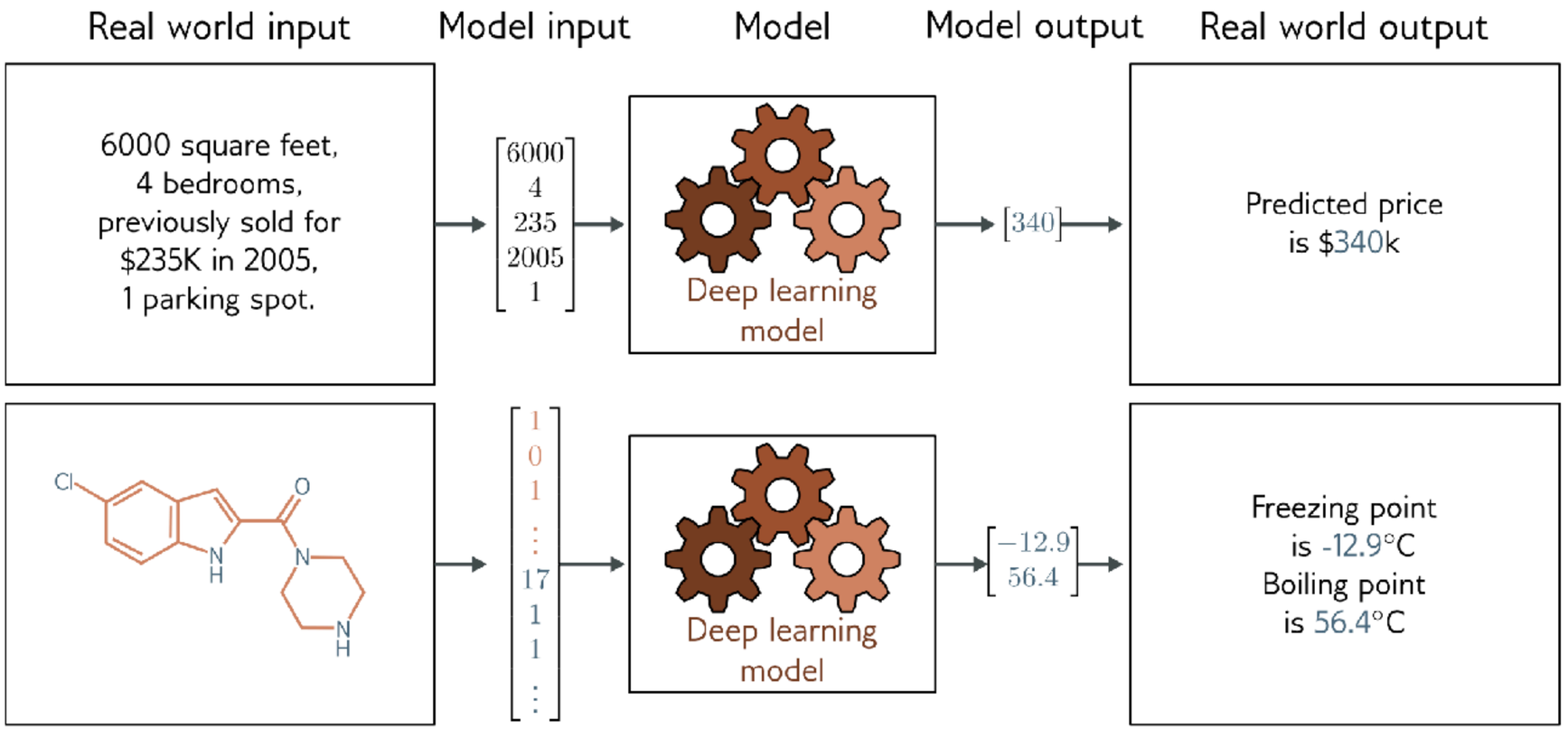
Regression Tasks

- Predict a real value given the input $\mathbf{x}$.
- The learning algorithm should produce a function $f_W : \mathbb{R}^D \rightarrow \mathbb{R}$.
- Learning is then the process of finding parameters W that improve performance.
- Often, the learning algorithm produces a distribution over outputs.



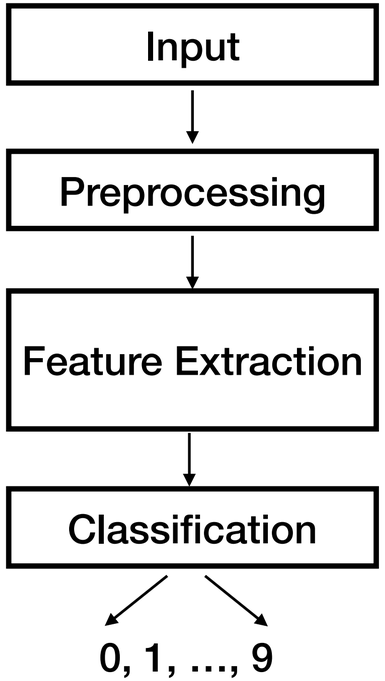


Regression Tasks - Examples

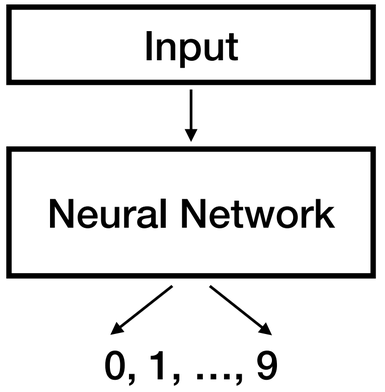


Training Philosophy

- **Classical approach**:
  - **Preprocessing** of original data (e.g., scaling and normalizing images w.r.t. luminance). Reduces the within-class variability.
  - **Feature extraction**. Reduces the dimensionality of data.



- **Modern approach**:
  - **End-to-end training** (minimal preprocessing).
    - No hand-crafted feature extraction.
    - Neural network does everything.
    - Needs large and deep networks.
    - Needs very large training data sets.



Today

- [x] Motivation
- [x] Pattern Recognition
- [ ] Central Concepts in Machine Learning
- [ ] Light Introduction to Neural Networks

Central Concepts in Machine Learning

- Generalization
- Capacity, Overfitting, Underfitting
- Regularization
- Hyperparameters

Generalization

observe: 1, 2, 4, 7, …

What is next?

Which continuation is correct ("generalizes")?

- 1, 2, 4, 7, **11**, 16, … : $a_{n+1} = a_n + n$ (Lazy Caterer's sequence)
- 1, 2, 4, 7, **12**, 20, … : $a_{n+2} = a_{n+1} + a_n + 1$
- 1, 2, 4, 7, **13**, 24, … : $a_{n+1} = a_n + a_{n-1} + a_{n-2}$ (Tribonacci sequence)
- 1, 2, 4, 7, **14**, 28 : divisors of 28.
- 1, 2, 4, 7, **1**, 1, 5, … : decimal expansions of $\pi = 3{,}1415\ldots$ and $e = 2.7182\ldots$ interleaved

The On-Line Encyclopedia of Integer Sequences

In the meantime:

observe: 1, 2, 4, 7, …

What is next?

A side note: This particular case is not a problem of "generalization" only (more on that later)…

Default (GPT-3.5)

OE Observe the integer sequence: 1, 2, 4, 7, ...
What can be the next integer?

The given sequence appears to be an arithmetic sequence. In an arithmetic sequence, each term is obtained by adding a constant difference to the previous term. Let's calculate the differences between consecutive terms:

2 - 1 = 1
4 - 2 = 2
7 - 4 = 3

The differences between consecutive terms are increasing by 1 each time. This suggests that the common difference between terms is 3. So, to find the next term in the sequence, we can add 3 to the last term given in the sequence:

7 + 3 = 10

Therefore, the next integer in the sequence is 10.

Generalization

The central challenge in machine learning:

- To perform well on *new, unseen examples*.
- The ability to perform well on previously unobserved examples is called **generalization**.

- We consider a **training set** X, t and typically reduce some "error" on it (**training error**).
- **Generalization error**: The expected value of the error on a new input.
- We *estimate* the generalization error using a separate **test set** (**test error, empirical error**).

Generalization

Statistical Learning Theory

The theoretical framework is based on the following concept:

- **Data generating process:** Training and test data are drawn from a probability distribution (**data generating distribution** p_{data}) over datasets.

- **i.i.d. assumption:** Each example is independently drawn from the same distribution.

- The test set is independently drawn from the training set, but from the same distribution.

The factors determining how well a machine learning algorithm performs are its ability to:

1. Make the training error small.
2. Make the gap between the training error and the generalization error small (difference to optimization).

Generalization

Statistical Learning Theory

The theoretical framework is bas

- **Data generating process:** Trai

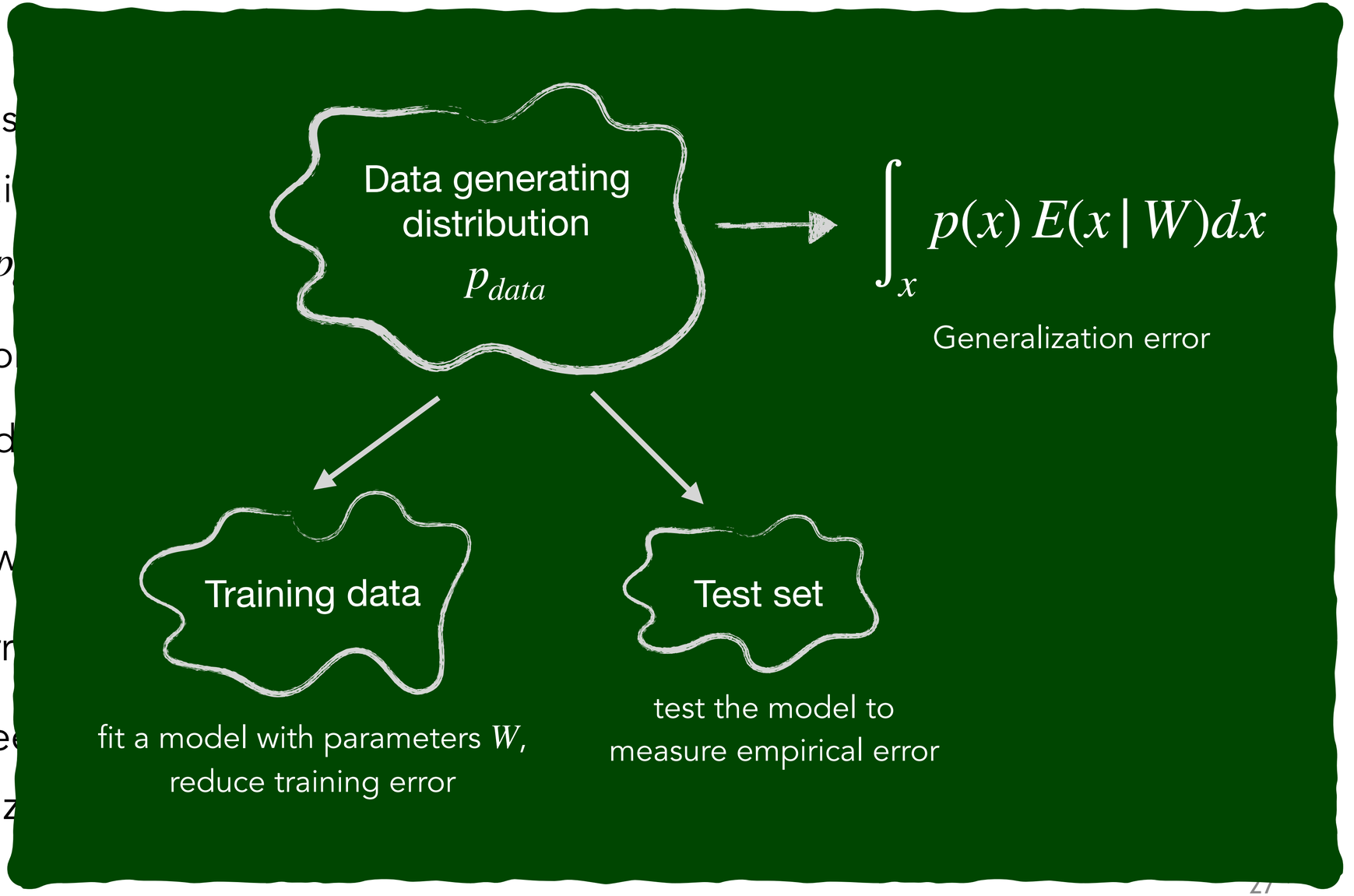
(**data generating distribution** p

- **i.i.d. assumption:** Each examp

- The test set is independently d

The factors determining how

1. Make the training err
2. Make the gap betwe
   (difference to optimiz



Capacity, Overfitting, Underfitting

- A learning algorithm chooses from a set of hypotheses (**hypothesis space**) $\mathscr{H}$.
- $\mathscr{H}$ may be the set of functions $\{f_W \mid W \in \mathscr{W}\}$.
  - W is a parameter vector from the set of possible parameter vectors $\mathscr{W}$.
  - By choosing W, we choose *one* possible model that can be represented by f_W.

Model capacity (complexity): The ability of a model to fit a wide variety of functions.

Underfitting: The model is not able to obtain low error on the training set (capacity too low).

Overfitting: The gap between the training error and test error is too large.

Capacity, Overfitting, Underfitting

The most important result in statistical learning theory:

The discrepancy between training error and generalization error is bounded from above by a quantity that grows as the model capacity grows but shrinks as the number of training examples increase.

[V. Vapnik (1995) "The Nature of Statistical Learning Theory"]

(Try to fit the simplest model that can capture the data and use as much data as possible for training.)

Occam's razor: Among competing hypotheses that explain known observations equally well, choose the *simplest* one.

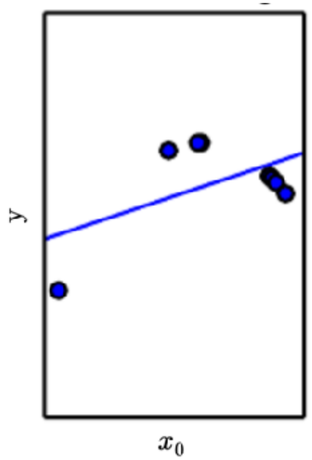
However, it is sometimes easier to optimize complex models,

… and complex models can be fit surprisingly well with large data sets and regularization.

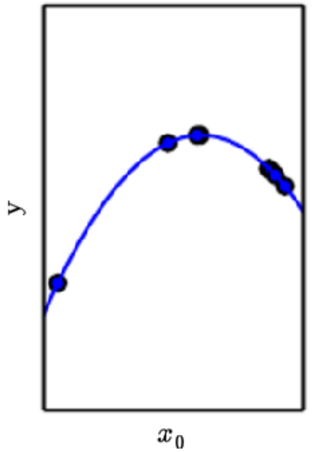
Model Complexity (Example: regression task)

$$\hat{y}(x) = w_0 + w_1 x \qquad \hat{y}(x) = w_0 + w_1 x + w_2 x^2 \qquad \hat{y}(x) = w_0 + \sum_{i=1}^{9} w_i x^i$$

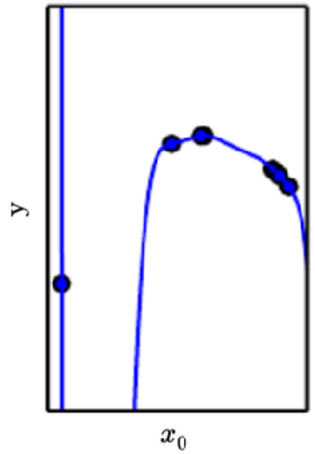
Underfitting



Appropriate capacity



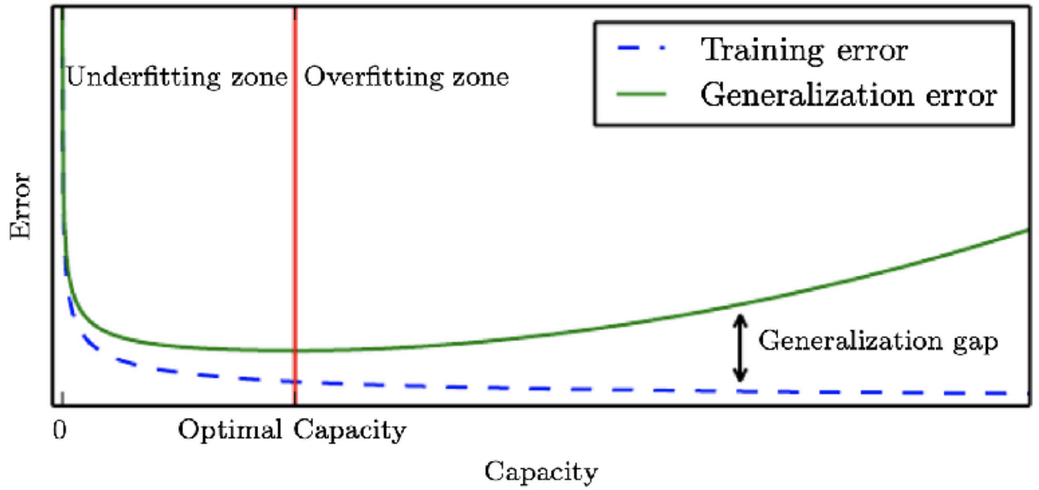
Overfitting



Training error: Error on training set

Generalization error: Expected error on data distribution

Test error / Empirical error: Error on our test set (a quantitative estimate of the generalization).



Quick recap: Error measures for regression

Sum squared error: $$E_{SSE} = \sum_{n=1}^{N} \{y(x^{(n)}) - t^{(n)}\}^2$$

Mean squared error: $$E_{MSE} = \frac{1}{N} \sum_{n=1}^{N} \{y(x^{(n)}) - t^{(n)}\}^2$$

Root mean squared error: $$E_{RMS} = \sqrt{\frac{1}{N} \sum_{n=1}^{N} \{y(x^{(n)}) - t^{(n)}\}^2}$$

Mean absolute error: $$E_{MAE} = \frac{1}{N} \sum_{n=1}^{N} |y(x^{(n)}) - t^{(n)}|$$

Regularization

- So far, we varied the model complexity to reduce generalization error (excluding hypotheses).
- We can also give a learning algorithm a preference for “simple” hypotheses.
- An unpreferred hypothesis will be chosen only if it fits the training data significantly better.

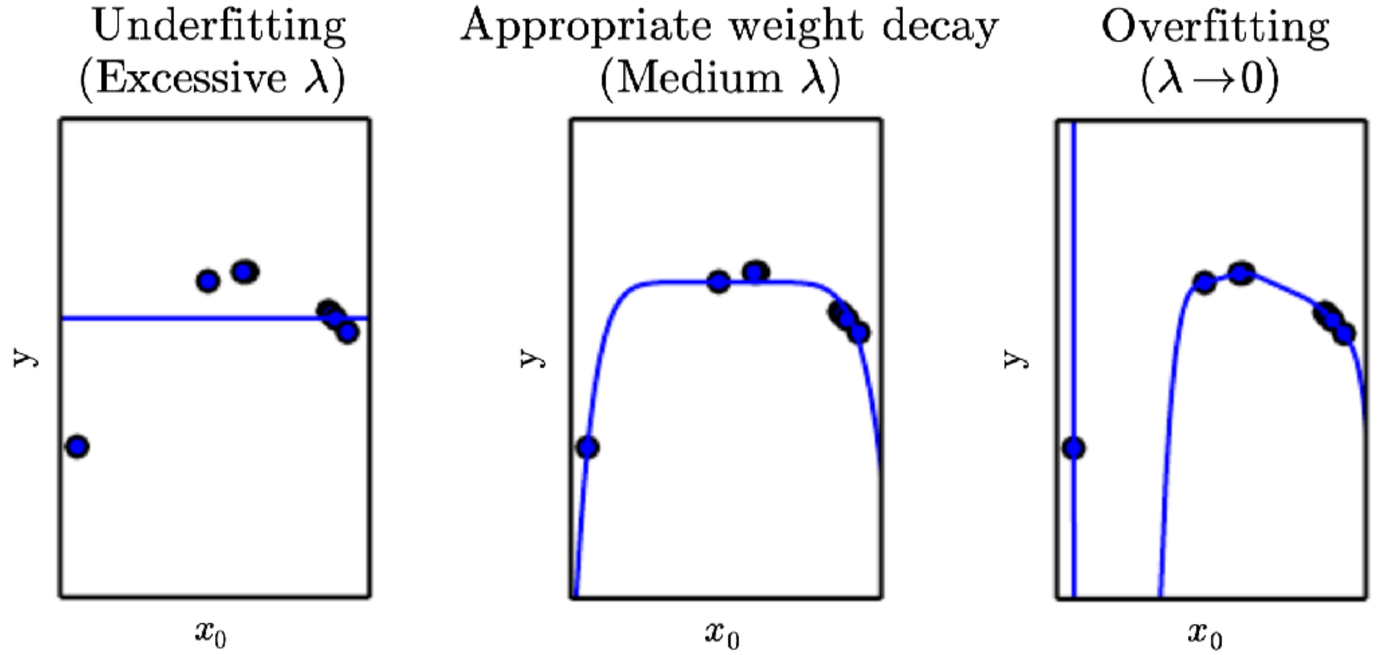
> ***Regularization*** *is any modification we make to a learning algorithm that is intended to reduce its generalization error but not its training error.*

Regularization (Example: weight decay)

$$\hat{y}(x) = w_0 + \sum_{i=1}^{9} w_i x^i$$

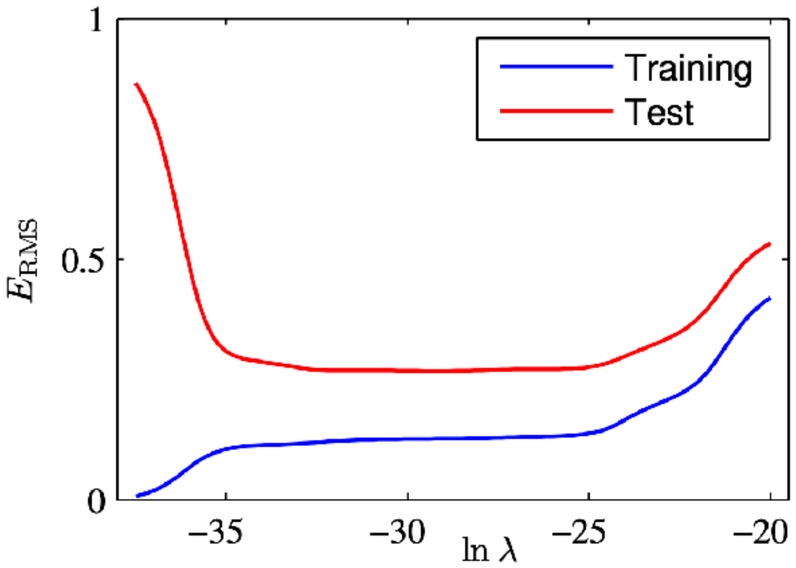
Minimize an error function that favors small parameters:

$$E(\mathbf{w}) = \sum_{n=1}^{N} \{\hat{y}(x^{(n)}) - t^{(n)}\}^2 + \lambda ||\mathbf{w}||^2$$



Hyperparameters

Hyperparameter λ controls the effective capacity of the model.



How to determine the hyperparameter value?

Practical method: Hold back data, called validation set, to optimize model complexity.

Training set: To optimize model parameters.

Validation set: To optimize hyperparameters, find model complexity.

Test set: To estimate the generalization error.

Summary

- **Generalization** is one of the central challenges in machine learning.
  - Statistical learning theory provides a theoretical grounding of the problem.
- We assume a data generating distribution in the background which produces training and test data.
  - Test data is used to compute the empirical error which approximates the generalization error.
- A good ML algorithm should:
  - *(i)* make the training error small, *(ii)* reduce the gap between training and generalization error.
- Avoid **overfitting** and **underfitting**.
- Generalization gap can be reduced by:
  - *(i)* simplifying the model, *(ii)* increasing the number of training data.
- **Regularization** is a convenient way to reduce model complexity implicitly.

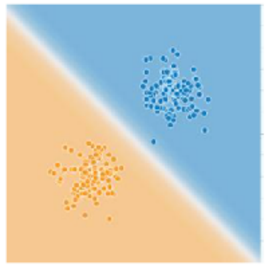
Today

- [x] Motivation
- [x] Pattern Recognition
- [x] Central Concepts in Machine Learning
- [ ] Light Introduction to Neural Networks

Artificial Neurons

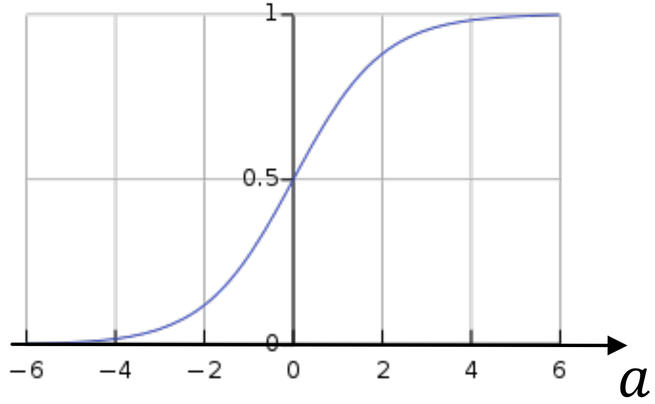
The simplest neural network: a neuron

$\mathrm{z}(\boldsymbol{x}) < 0.5 \Longrightarrow \text{Class } 0$
$z(\boldsymbol{x}) \geq 0.5 \Longrightarrow \text{Class } 1$



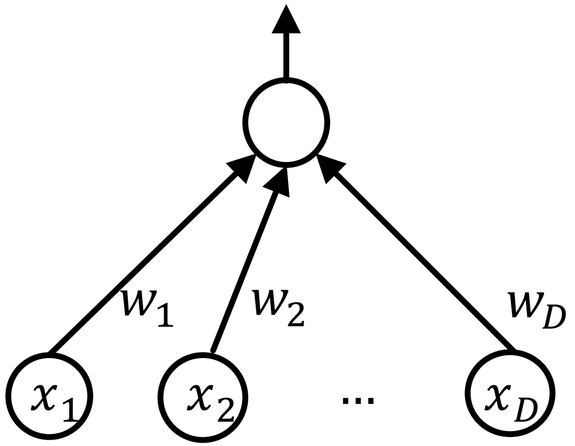
Activation function: $h\colon \mathbb{R} \to \mathbb{R}$

$$\sigma(a) = \frac{1}{1 + e^{-a}}$$



$$z(\boldsymbol{x}) = h\left(\sum_{i=1}^{D} w_i x_i + w_0\right) = h(\boldsymbol{w}^T \boldsymbol{x} + w_0)$$

activation a



Input vector: $\boldsymbol{x} = \begin{pmatrix} x_1 \\ \vdots \\ x_D \end{pmatrix} \in \mathbb{R}^D$

Weight vector: $\boldsymbol{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_D \end{pmatrix} \in \mathbb{R}^D$

Bias: $w_0 \in \mathbb{R}$

parameters

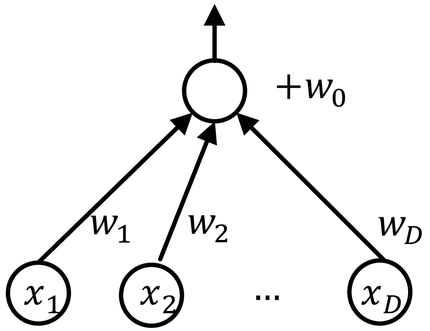
Some mathematical conventions

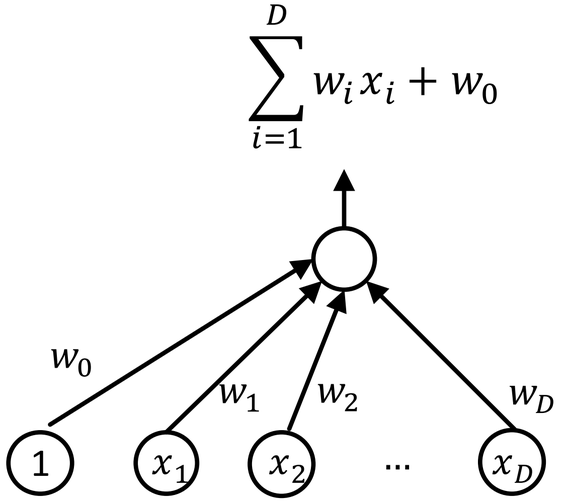
- Vectors are written bold-face.
- We always assume column vectors: $\boldsymbol{w} = \begin{pmatrix} w_1 \\ \vdots \\ w_D \end{pmatrix} \in \mathbb{R}^D$
- The transpose is then a row vector: $\boldsymbol{w}^T = (w_1, \ldots, w_D)$
- Getting rid of the bias:

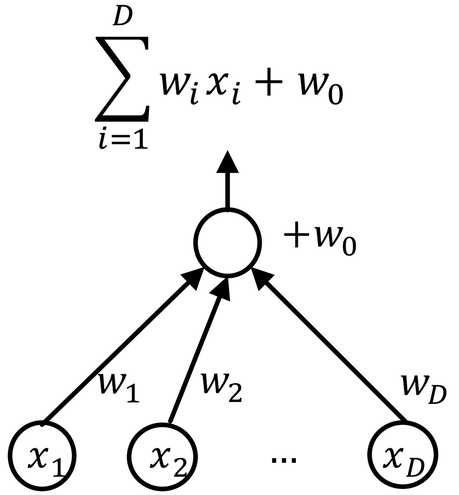
$$\boldsymbol{x} = (1, x_1, \ldots, x_D)^T$$
$$\boldsymbol{w} = (w_0, w_1, \ldots, w_D)^T$$

$$\boldsymbol{w}^T\boldsymbol{x} = \sum_{i=1}^{D} w_i x_i + w_0$$

$$z(\boldsymbol{x}) = h\left(\sum_{i=1}^{D} w_i x_i + w_0\right) = h(\boldsymbol{w}^T\boldsymbol{x} + w_0)$$

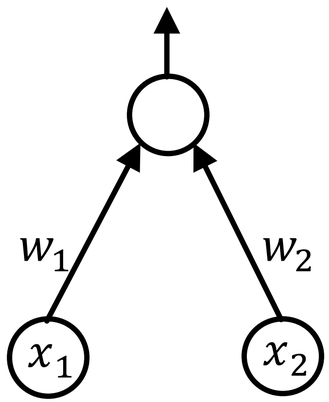




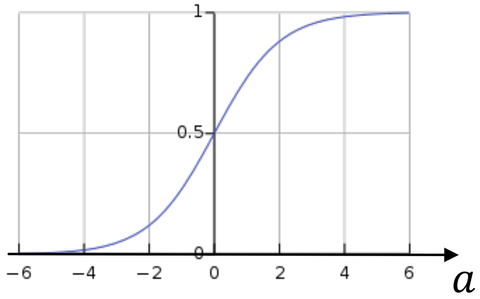


Artificial Neurons

$$z(\boldsymbol{x}) = h(\boldsymbol{w}^T\boldsymbol{x} + w_0)$$



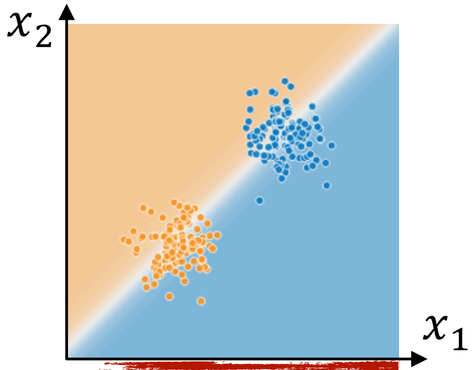
Activation function: $h\colon \mathbb{R} \to \mathbb{R}$



Error measure: Estimates "the quality of network approximation". For instance:

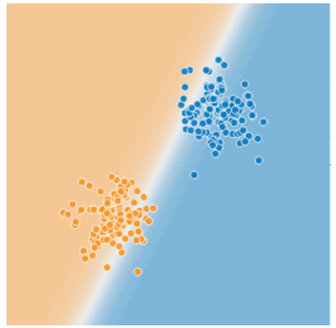
- **Misclassification rate:** Number of incorrectly classified training examples
- **Cross-Entropy-Error**: Similar, but better suited for network optimization.

Parameters:
$w_0 = 0$
$w_1 = 1$
$w_2 = -1$



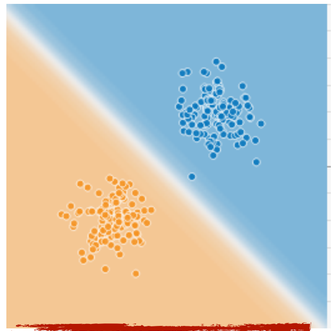
$E(\boldsymbol{w}) = 49.5$

Parameters:
$w_0 = 0$
$w_1 = 1$
$w_2 = -0.5$

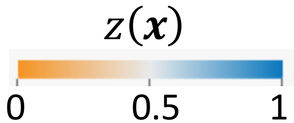


$E(\boldsymbol{w}) = 10.2$

Parameters:
$w_0 = 0$
$w_1 = 1$
$w_2 = 1$



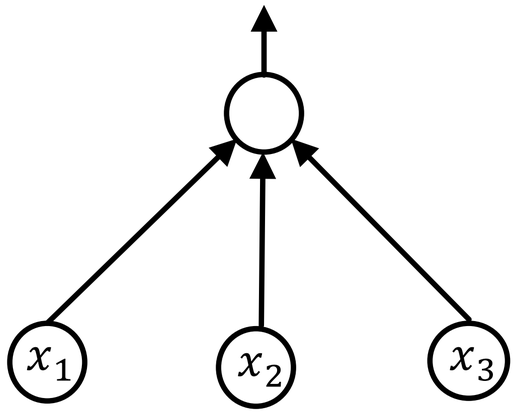
$E(\boldsymbol{w}) = 0.01$

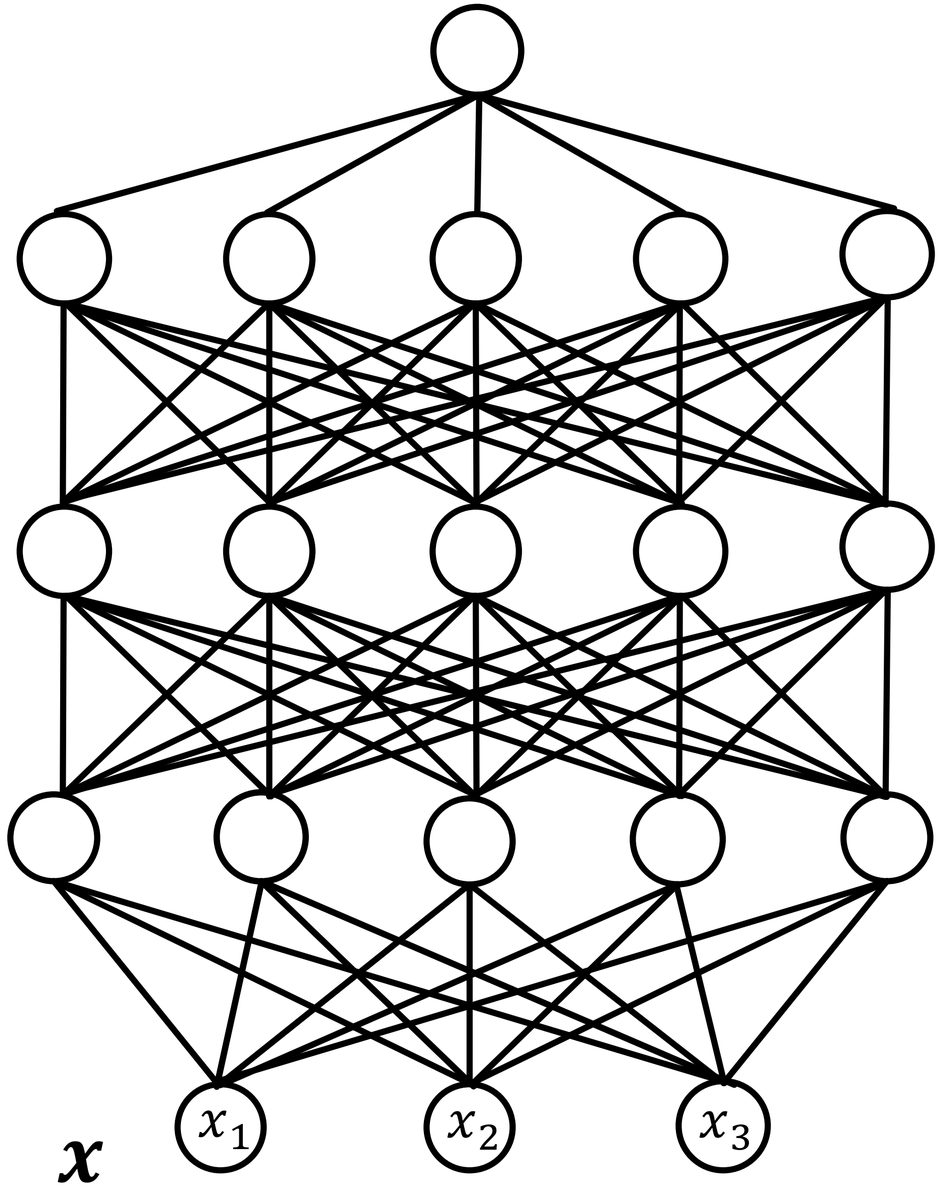


Multi-Layer Perceptrons (MLP)

$$z(\boldsymbol{x}) = h(\boldsymbol{w}^T[\text{output of the previous layer}] + w_0)$$

$$z(\boldsymbol{x}) = h(\boldsymbol{w}^T\boldsymbol{x} + w_0)$$





Multi-Layer Perceptrons (MLP)

$$y(\boldsymbol{x}) = z^{(L)}(\boldsymbol{x}) = h^{(L)}\left(W^{(L)}\boldsymbol{z}^{(L-1)}\right)$$

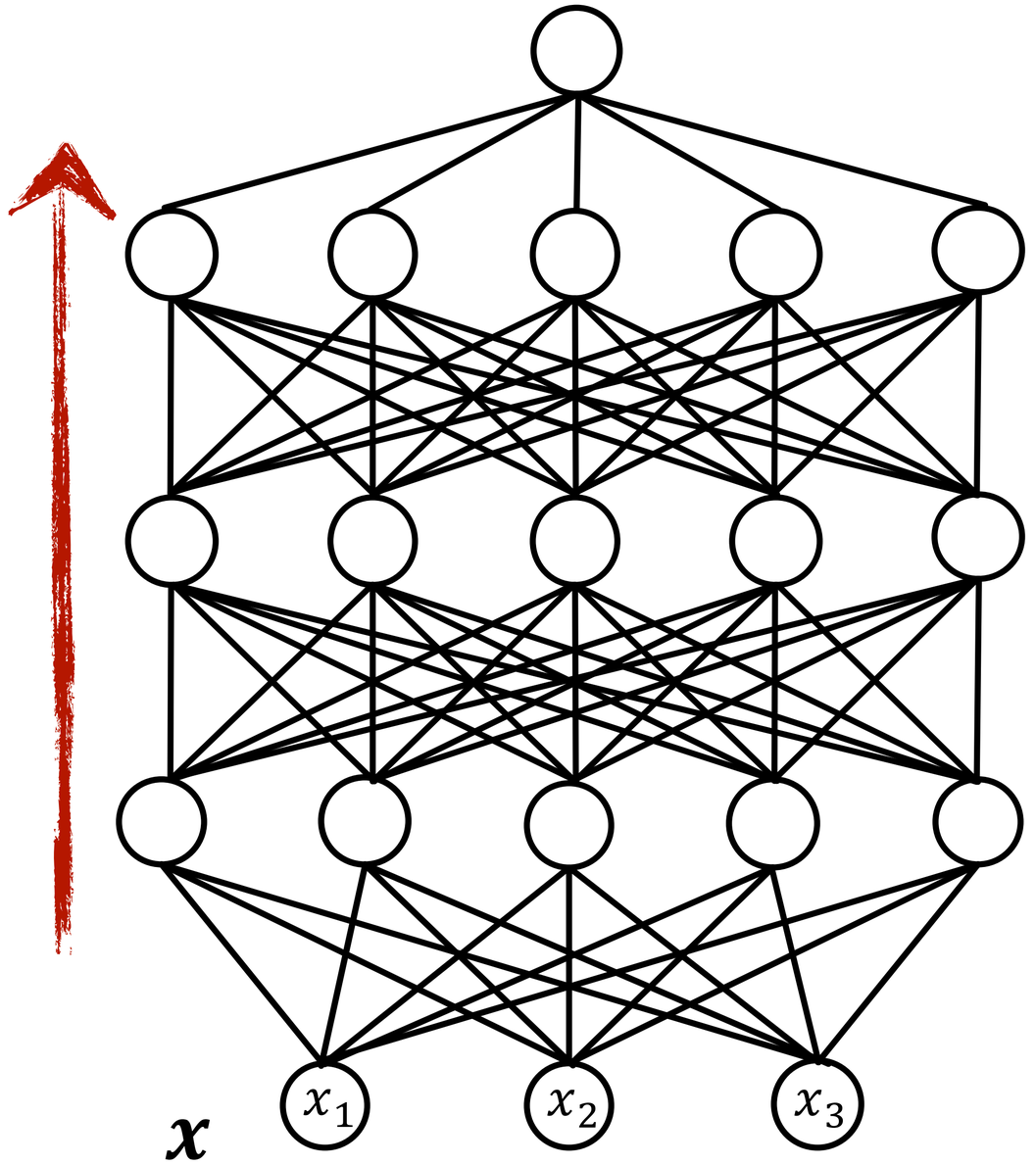
$$\boldsymbol{z}^{(l)}(\boldsymbol{x}) = h^{(l)}\left(W^{(l)}\boldsymbol{z}^{(l-1)}\right)$$

$$\boldsymbol{z}^{(2)}(\boldsymbol{x}) = h^{(2)}\left(W^{(2)}\boldsymbol{z}^{(1)}\right)$$

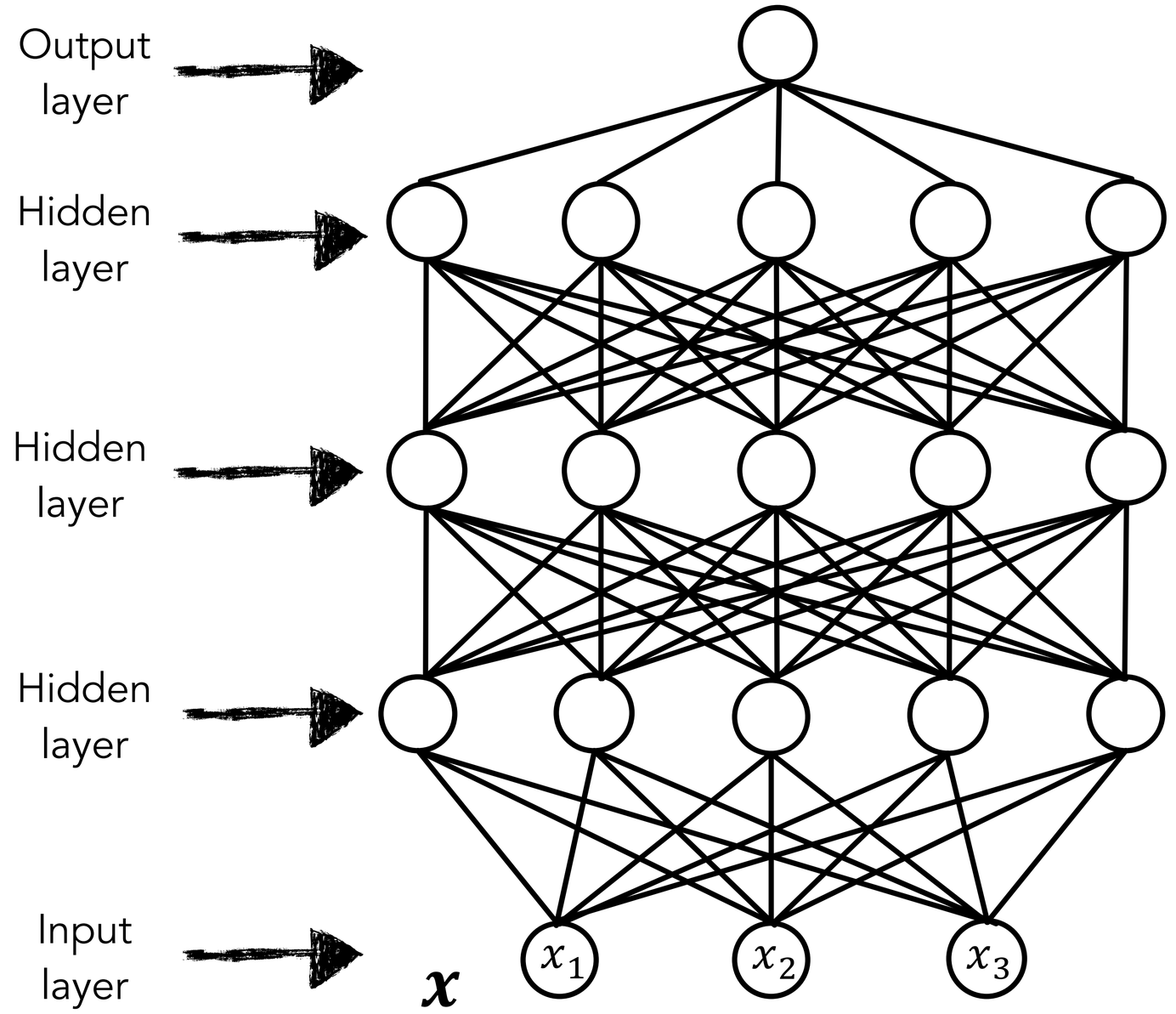
$$\boldsymbol{z}^{(1)}(\boldsymbol{x}) = h^{(1)}\left(W^{(1)}\boldsymbol{x}\right)$$

Note:

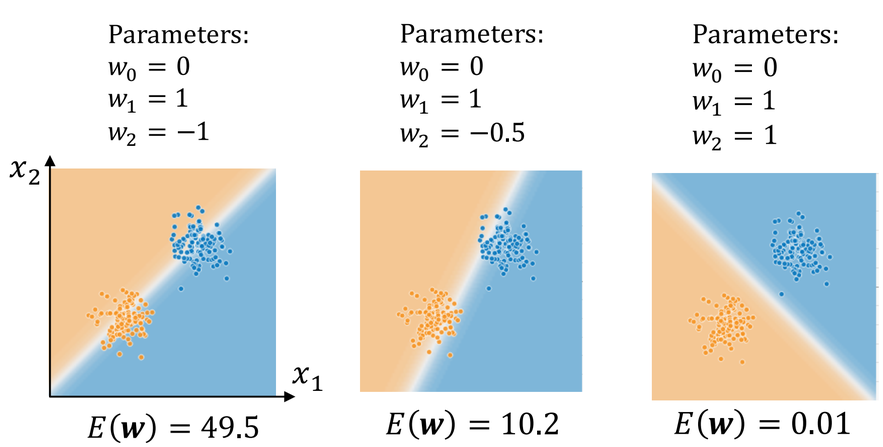
$$z_j^{(1)} = h^{(1)}\left(W_{j,:}^{(1)}\boldsymbol{x}\right)$$



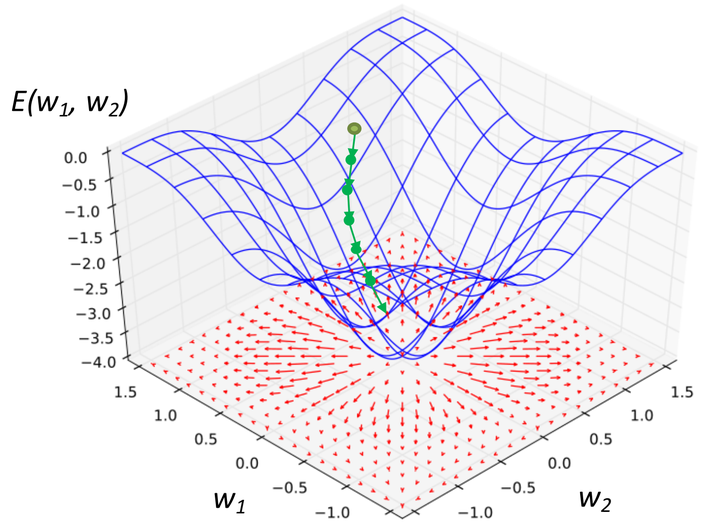
Multi-Layer Perceptrons (MLP)



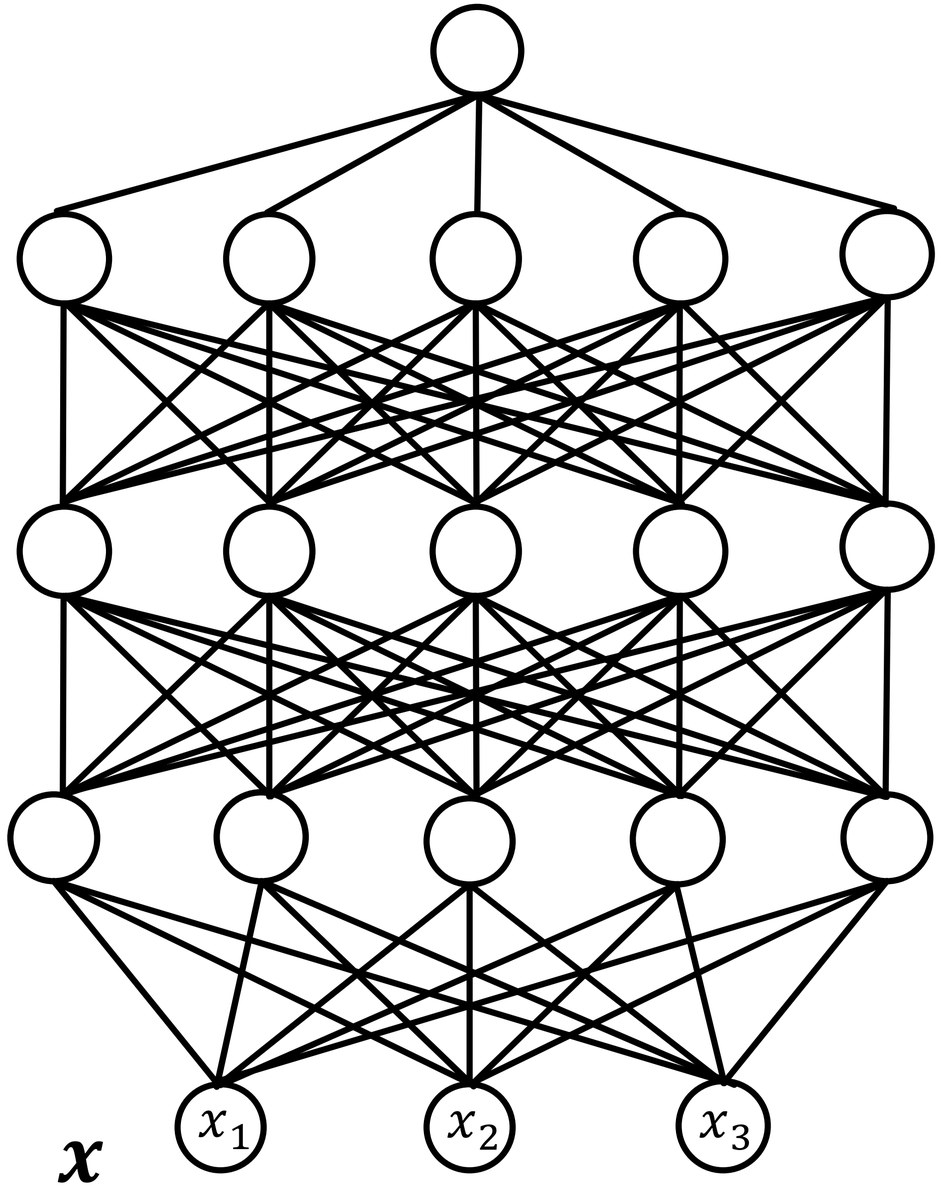
Multi-Layer Perceptrons (MLP)



Training neural networks to have lesser "error measure":



playground



Today

- ☑ Motivation
- ☑ Pattern Recognition
- ☑ Central Concepts in Machine Learning
- ☑ Light Introduction to Neural Networks

Questions?