Machine Learning for social sciences

Introduction to Machine Learning

William Aboucaya

Who am I?

William Aboucaya, data scientist at the ACSS institute

Ph. D. in computer science focused on **online citizen participation**

Expertise in **data science** and **natural language processing**

Currently working on

- Discourses in the European parliament
- (Gender) dynamics of research in management
- NLP for requirements engineering

Why are we here?

Who are you?

- Who has prior experience with coding?

Who are you?

- Who has prior experience with coding?
- Who has prior experience with data science? machine learning?

Who are you?

- Who has prior experience with coding?
- Who has prior experience with data science? machine learning?
- Do some of you have an idea for their masters project relying on machine learning / data science?

What is machine learning?

What is machine learning?

- A part of the field of artificial intelligence?

What is machine learning?

- A part of the field of artificial intelligence?
- A field of methods aiming at making sense of large data?

What is machine learning?

- A part of the field of artificial intelligence?
- A field of methods aiming at making sense of large data?
- Extrapolation on new data based on previous trends?

What is machine learning?

- A part of the field of artificial intelligence?
- A field of methods aiming at making sense of large data?
- Extrapolation on new data based on previous trends?

All of it and more!

Definition

Intuitive definition:

Machine learning is the study and practice of algorithms that *automatically* extract patterns from data and improve their performance on tasks through experience.

Examples:

- Supervised: image classification, house price regression
- Unsupervised: clustering, dimensionality reduction
- Reinforcement: game playing, robotics, LLM alignment

Definition

Formal (concise) definition:

An algorithm is said to *learn* from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.

Key ingredients

- **Data / Experience (E):** labeled or unlabeled examples, interactions, sensor readings
- **Task (T):** classification, regression, clustering, control, etc.
- **Performance (P):** accuracy, MSE, reward, F1 score, etc.
- **Model / Hypothesis:** $f_\theta(x)$ with parameters θ learned from data

Outline of the module

1. Introduction to Machine Learning $\Leftarrow$ **You are here**
2. Regressions
3. Text classification and generation
4. Clustering and topic modeling
5. NLI and Zero-Shot classification
6. Wildcard session based on what would be interesting to you
7. Project topic selection + Exam

Outline of the session

1. Why do we train models?

2. Some general concepts for next classes

3. A short history of machine learning[1]

[1] Umberto Michelucci. "Machine Learning: History and Terminology". In: *Fundamental Mathematical Concepts for Machine Learning in Science*. Cham: Springer International Publishing, 2024, pp. 9–20. DOI: 10.1007/978-3-031-56431-4_2.

Why do we train models?

Expert systems vs. Trained (ML) models

Two Paradigms in AI

Expert Systems (Symbolic AI)

Knowledge is **explicitly encoded** as rules provided by human experts.

Trained Models (Statistical / Machine Learning)

Knowledge is **implicitly learned** from data by optimizing model parameters

Expert systems vs. Trained (ML) models

Two Paradigms in AI

Expert Systems (Symbolic AI)

Knowledge is **explicitly encoded** as rules provided by human experts.

Trained Models (Statistical / Machine Learning)

Knowledge is **implicitly learned** from data by optimizing model parameters

- Expert systems: "top-down" reasoning
- Trained models: "bottom-up" learning

Expert Systems

- Core idea: emulate human decision-making via a rule base and inference engine
- Knowledge representation: **IF–THEN** rules
- Advantages:
  - Transparent reasoning
  - Easy to audit and modify
- Limitations:
  - Hard to scale and maintain
  - Fragile in noisy or uncertain domains

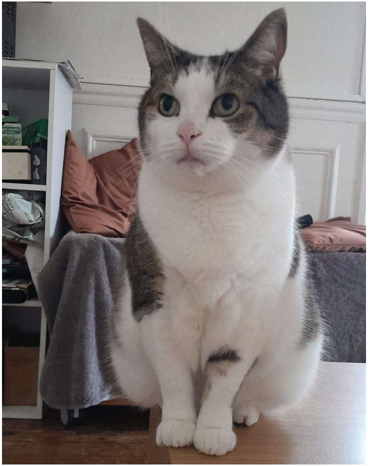
Trained Models (Machine Learning)

- Core idea: learn patterns from large datasets
- Examples: neural networks, decision trees, transformers
- Knowledge representation: **parameters and weights**
- Advantages:
  - Adapt well to complex and unstructured data
  - Automatically improve with more data
- Limitations:
  - Lack interpretability (“black box”)
  - Require large amounts of data and computing power

Say we want to identify cat pictures

Using an expert system:

if *[Entity] number_of_limbs 4*
and
[Entity] has [Fur]
and
[Entity] has [Whiskers]
then
[Entity] is [Cat]
else
[Entity] is_not [Cat]



Example: Identifying cat pictures

Say we want to identify cat pictures

Using an expert system:

if *[Entity] number_of_limbs 4*
and
[Entity] has [Fur]
and
[Entity] has [Whiskers]
then
[Entity] is [Cat]
else
[Entity] is_not [Cat]

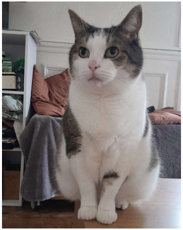
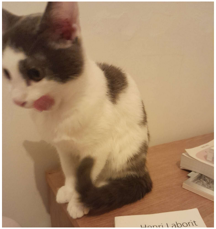
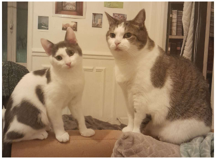
But outliers can be hard to anticipate!

Example: Identifying cat pictures

Say we want to identify cat pictures

Using machine learning:

Cats


Non-cats

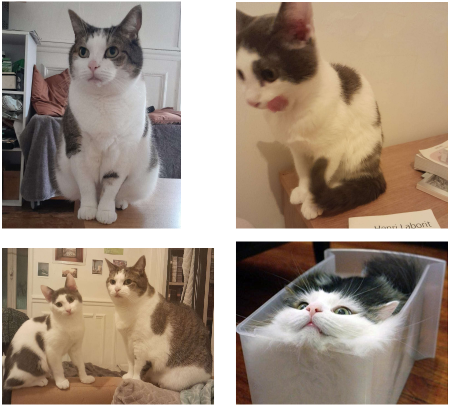



Example: Identifying cat pictures

Say we want to identify cat pictures

Using machine learning:

Cats



Non-cats

Generalize from hundreds of pictures of cats and non-cats!

Summary

Expert Systems

- Rule-based
- Transparent
- Brittle

Trained Models

- Data-driven
- Opaque
- Robust

Summary

Expert Systems

- Rule-based
- Transparent
- Brittle

Trained Models

- Data-driven
- Opaque
- Robust

Modern Trend: Neuro-Symbolic AI

Combining explicit reasoning (rules, logic, knowledge bases) with learned representations (trained neural networks) to get the best of both worlds.

Some general concepts for next classes

Supervized and Unsupervized learning

Supervised Learning

- Learns from **labeled data** — each training example has input-output pairs.
- Goal: learn a mapping from inputs X to outputs Y.
- Common tasks:
  - Classification (e.g., spam vs. non-spam emails)
  - Regression (e.g., predicting house prices)
- **Example:** Email Classification
  - Input: email text
  - Output: `spam / not spam`

Supervized and Unsupervized learning

Supervised Learning

- Learns from **labeled data** — each training example has input-output pairs.
- Goal: learn a mapping from inputs X to outputs Y.
- Common tasks:
  - Classification (e.g., spam vs. non-spam emails)
  - Regression (e.g., predicting house prices)
- **Example:** Email Classification
  - Input: email text
  - Output: `spam / not spam`

Unsupervised Learning

- Learns from **unlabeled data** — only inputs, no explicit outputs.
- Goal: discover hidden patterns or groupings in the data.
- Common tasks:
  - Clustering
  - Dimensionality reduction
- **Example:** Customer Segmentation
  - Input: customer purchase history
  - Output: group labels found automatically

Supervized and Unsupervized learning

Supervised Learning

- Learns from **labeled data** — each training example has input-output pairs.
- Goal: learn a mapping from inputs X to outputs Y.
- Common tasks:
  - Classification (e.g., spam vs. non-spam emails)
  - Regression (e.g., predicting house prices)
- **Example:** Email Classification
  - Input: email text
  - Output: `spam / not spam`

Unsupervised Learning

- Learns from **unlabeled data** — only inputs, no explicit outputs.
- Goal: discover hidden patterns or groupings in the data.
- Common tasks:
  - Clustering
  - Dimensionality reduction
- **Example:** Customer Segmentation
  - Input: customer purchase history
  - Output: group labels found automatically

Key distinction: Supervised learning uses labeled data, while unsupervised learning finds structure in unlabeled data.

Regression problem

Definition

Regression is a type of supervised learning used to model the relationship between a dependent (target) variable y and one or more independent (input) variables x.

Goal

To predict continuous outcomes, such as prices, temperatures, or sales, by fitting a function:

$$\hat{y} = f_\theta(x)$$

where θ are model parameters learned from data.

Classification problem

Definition

Classification is another type of supervised learning used to assign a class $\hat{y}$ to an input x among a series of pre-defined labels $\{1, 2, ..., K\}$.

Goal

To predict the probability of each label to be the relevant label by fitting a function and applying an algorithm:

$$\hat{y} = \arg\max_{i=1}^{K} P_\theta(y = i|x)$$

where θ are model parameters learned from data.

What is a Neural Network?

- Inspired by the structure of the human brain.
- Composed of interconnected units called **neurons**.
- Each neuron processes inputs and passes the output to the next layer.
- Used for tasks like classification, regression, image and speech recognition.

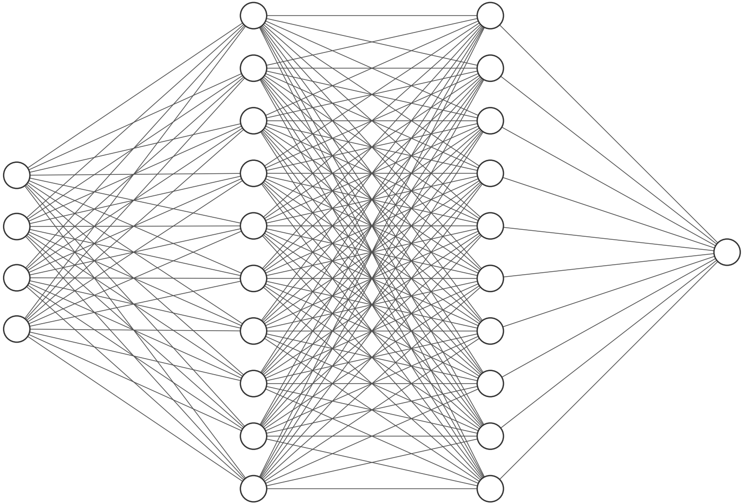
What is a Neural Network?

- Inspired by the structure of the human brain.
- Composed of interconnected units called **neurons**.
- Each neuron processes inputs and passes the output to the next layer.
- Used for tasks like classification, regression, image and speech recognition.

Structure of a Neural Network

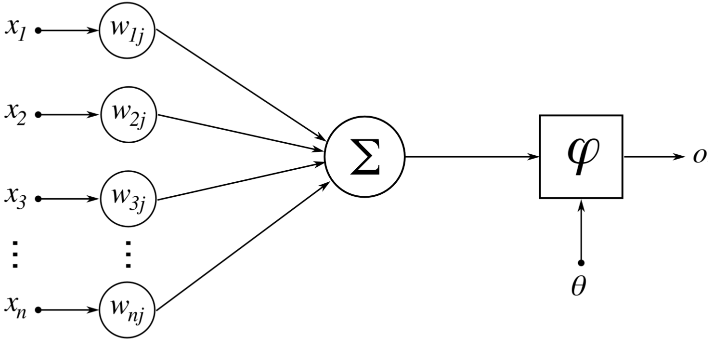
- Input layer - receives data.
- Hidden layers - extract features and patterns.
- Output layer - produces final prediction.

Structure of a basic Feed-Forward Neural Network



Input Layer $\in \mathbb{R}^4$ Hidden Layer $\in \mathbb{R}^{10}$ Hidden Layer $\in \mathbb{R}^{10}$ Output Layer $\in \mathbb{R}^1$

Mathematical Model



- x_i: input features
- w_i: weights
- θ: bias term
- Σ: aggregation function (e.g. sum, norm)
- φ: activation function (e.g. ReLU, sigmoid)
- o: output

Training Neural Networks

1. Initialize weights randomly.
2. Compute output through **forward propagation**.
3. Calculate error with a **loss function**.
4. Update weights using **backpropagation** and gradient descent.

Goal

Minimize the loss function to improve the quality of model predictions.

Assessing a model

Goal: Assess how well a trained model generalizes to unseen data.

Key idea: Estimate the model's predictive performance, not just its ability to fit training data.

Process:

1. Split data into *train*, *validation*, and *test* sets.
2. Train model on the training set.
3. Tune hyperparameters based on results for validation set if necessary.
4. Evaluate final performance on the test set.

Common concerns:

- Overfitting vs. underfitting
- Data leakage
- Model bias and variance

Evaluation for Continuous Outputs

Objective: Measure the closeness between predicted and actual values.

Common metrics:

- Mean Absolute Error (MAE): $\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$
- Mean Squared Error (MSE): $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$
- R^2 (Coefficient of Determination): $1 - \frac{\sum(y_i - \hat{y}_i)^2}{\sum(y_i - \bar{y})^2}$

Interpretation:

- Lower error $\Rightarrow$ better performance.
- R^2 closer to 1 $\Rightarrow$ better fit.

Considerations:

- Scale sensitivity of errors.
- Outlier impact (MSE vs. MAE).

Evaluation for Discrete Outputs

Objective: Compare predicted class labels with true labels.

Confusion Matrix:

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

Common metrics:

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1-score: $2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$

Evaluation for Discrete Outputs

Objective: Compare predicted class labels with true labels.

Confusion Matrix:

$$\begin{bmatrix} TP & FP \\ FN & TN \end{bmatrix}$$

Common metrics:

- Accuracy: $\frac{TP+TN}{TP+TN+FP+FN}$
- Precision: $\frac{TP}{TP+FP}$
- Recall: $\frac{TP}{TP+FN}$
- F1-score: $2 \times \frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$

Note: Metric choice depends on class balance and application context.
For multi-class models, these metrics can be aggregated by either using:

- Micro-averaging: compute the metric globally by changing the “true” class according to the true label of each datapoint.
- Macro-averaging: compute the metric for each class individually and then produce the unweighted average of all scores.

Underfitting and Overfitting

Goal: Find the right model complexity to capture patterns in data without memorizing noise.

Underfitting:

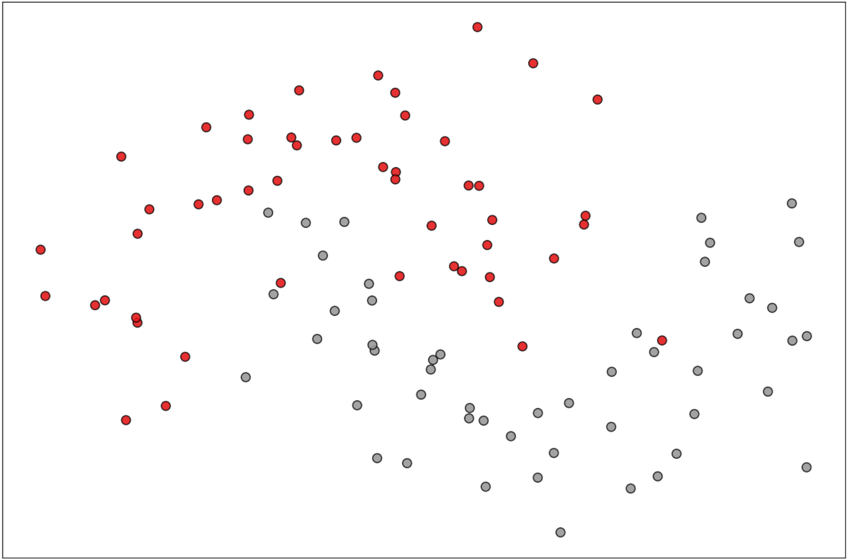
- Model is too simple to capture underlying patterns.
- Both training and test errors are high.
- Example: Using linear regression for highly nonlinear data.

Overfitting:

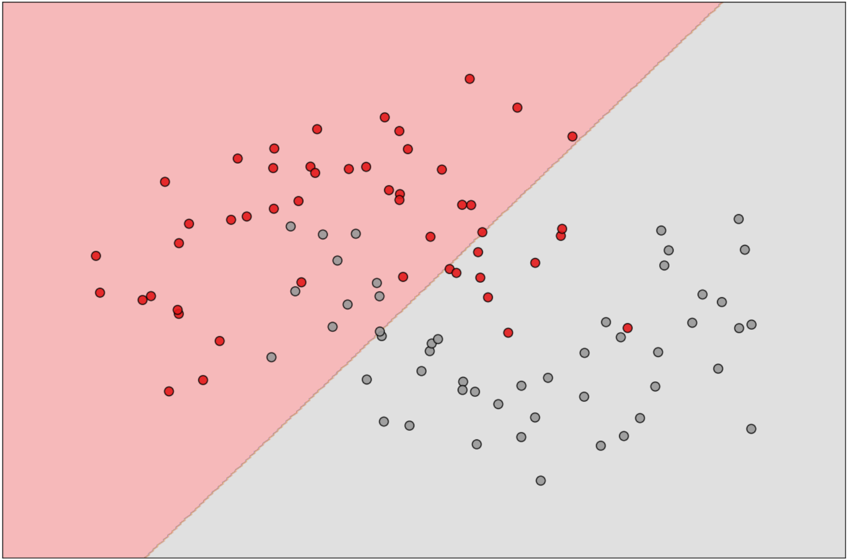
- Model is too complex and fits noise in training data.
- Training error is very low, but test error is high.
- Example: High-degree polynomials.

*Optimal model complexity minimizes **validation** error.*

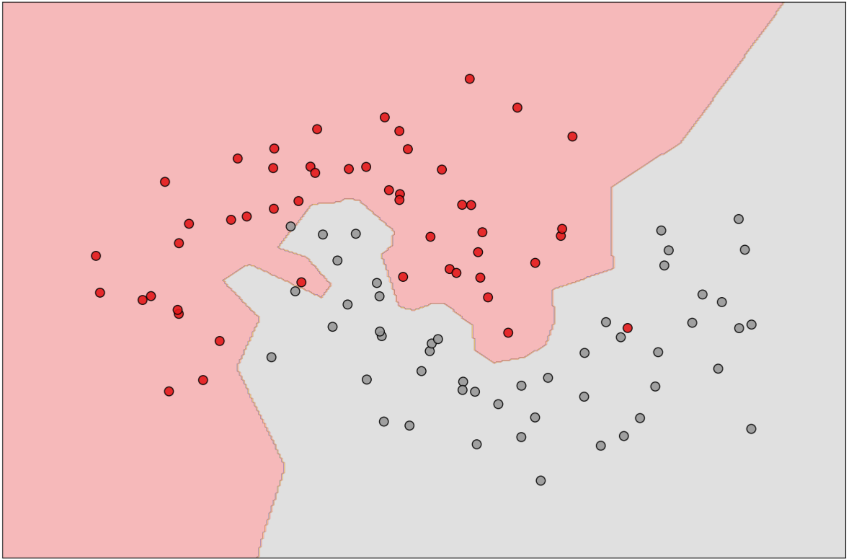
A simple example: initial data



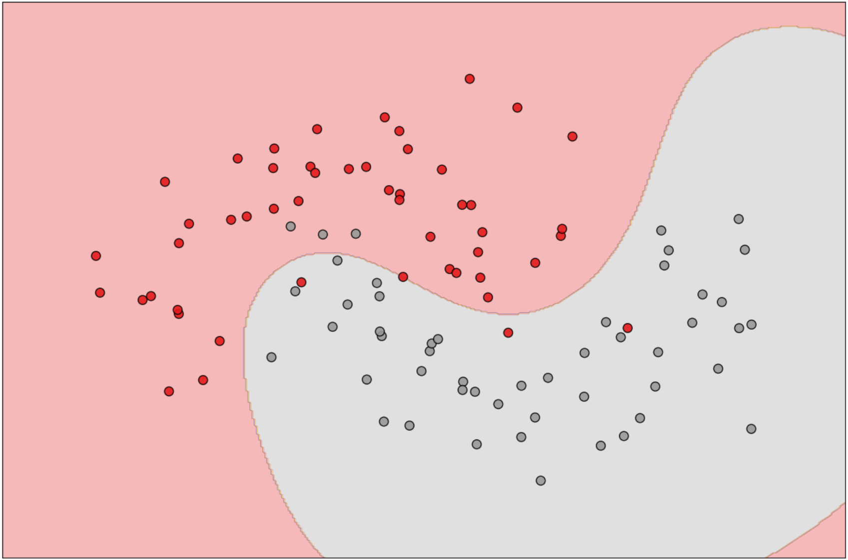
A simple example: Underfitting



A simple example: Overfitting



A simple example: Good fitting



A short history of machine learning[2]

[2] Umberto Michelucci. "Machine Learning: History and Terminology". In: *Fundamental Mathematical Concepts for Machine Learning in Science*. Cham: Springer International Publishing, 2024, pp. 9–20. DOI: 10.1007/978-3-031-56431-4_2.

Alan Turing and the Birth of Machine Intelligence (1950)

- Seminal paper[3] posing the question: **Can machines think?**
- Introduced the **Turing Test** — assessing if a machine's responses are indistinguishable from a human's.
- Speculated that machines could **learn from experience** rather than being pre-programmed with knowledge.
- Set the philosophical and conceptual foundation for Artificial Intelligence and Machine Learning.

[3] A. M. Turing. "Computing Machinery and Intelligence". In: *Mind* 59.236 (1950), pp. 433–460. URL: http://www.jstor.org/stable/2251299.

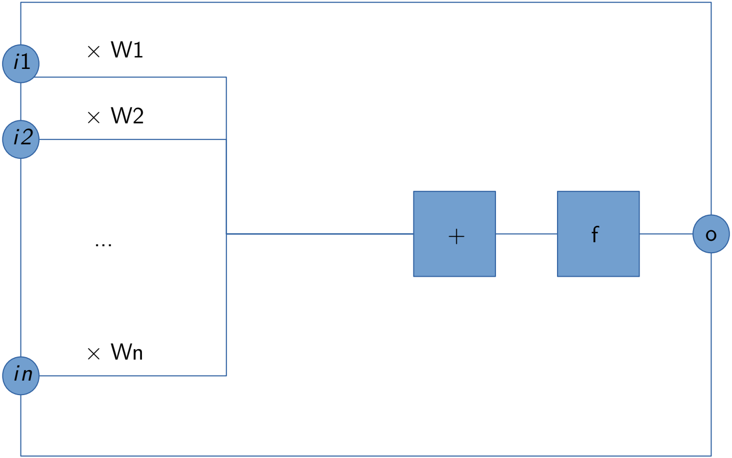
Checkers, Perceptron and Early Neural Networks (1950s)

- **Arthur Samuel**[4] coins the term "machine learning" and proposes a program to compute odds of victory in a game of checkers
- **Frank Rosenblatt**[5] develops the **Perceptron**, inspired by neurons.
  - Represented the first model capable of **learning from data** through weight adjustment.
  - Could only handle **linearly separable data**.
  - Introduced the notion of **supervised learning**.

[4] A. L. Samuel. "Some Studies in Machine Learning Using the Game of Checkers". In: *IBM Journal of Research and Development* 3.3 (1959). DOI: 10.1147/rd.33.0210.

[5] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain.". In: *Psychological Review* 65.6 (1958). DOI: 10.1037/h0042519.

Perceptron: the most simple example of a neural network



$$o = f\left(\sum_{k=1}^{n} i_k \cdot W_k\right)$$

First AI Winter

- **Minsky & Papert's** work[6] analyzed mathematical limitations of the perceptron.
- Proved it couldn't solve problems like XOR.
- Led to reduced funding and interest in neural networks — known as the **First AI Winter**.
- Their work later guided development of **multi-layer networks**.

[6] Marvin Minsky and Seymour Papert. *Perceptrons.* Perceptrons. Oxford, England: M.I.T. Press, 1969.

The 1980s Renaissance: Algorithms and Neural Networks

- **Backpropagation**[7]: efficient training of multi-layer neural networks.
- **Decision Trees**[8]: used information entropy to classify data.
- Practical applications appeared — e.g., **NetTalk**[9] for speech synthesis.
- Renewed optimism marked the **Second Wave of Neural Networks**.

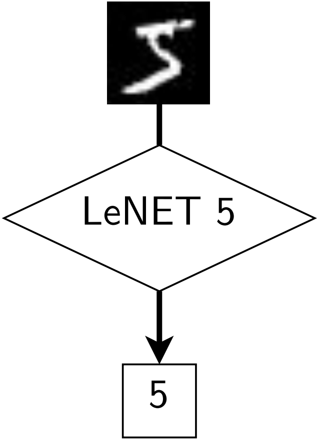
[7] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. "Learning representations by back-propagating errors". In: *Nature* 323.6088 (Oct. 1986), pp. 533–536. DOI: 10.1038/323533a0.

[8] J. R. Quinlan. "Induction of decision trees". In: *Machine Learning* 1.1 (Mar. 1986), pp. 81–106. DOI: 10.1007/BF00116251.

[9] Terrence J. Sejnowski and Charles R. Rosenberg. "Parallel Networks that Learn to Pronounce English Text". In: *Complex Systems* 1 (1987).

1990s: Toward Practical Applications

- **Convolutional Neural Networks (CNNs)** applied on hand-written digits and latin characters recognition[10].
- Emergence of **Support Vector Machines (SVMs)**[11] for robust classification.
- Increasing computational power and data availability bridged theory with real-world applications.



[10] Y. Lecun et al. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: 10.1109/5.726791.

[11] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

2000s: Reinforcement Learning and Big Data

- Major advances in **Reinforcement Learning (RL)** and formalization of RL principles[12].
- Emergence of very large datasets
- Open-source tools facilitate ML implementation[13].

[12] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998. ISBN: 0262193981.

[13] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

2010s: The Deep Learning Revolution

- **ImageNet**[14]: deep CNNs surpass traditional methods in image recognition.
- **Transformers**[15]: attention-based architectures revolutionized NLP.
- **BERT** and **T5** models advanced understanding and generation of natural language.
- Paradigm shift with **foundation models** fine-tuned on specific tasks.
- Transition toward **Large Language Models (LLMs)** with billions of parameters.

[14] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems*. Vol. 25. 2012. DOI: 10.1145/3065386.

[15] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. 2017.

Generative AI (2010s–Present)

- Enabled creation of synthetic images, data, and texts.
- LLM-based chatbots have been adopted by the general public.
- Development of large-scale infrastructures for cloud-computing.
- Generative AI now is implemented in an increasing number of sectors.