

Mohammed VI Polytechnic University

College of Agriculture and Environmental Sciences

International Water Research Institute (IWRI)

Organizes a workshop:
Mathematical Models and
Artificial Intelligence for
Solving Groundwater Flow Problems

by Dr. Andy Louwyck

06 and 07 January 2025



DAY 2

ARTIFICIAL INTELLIGENCE

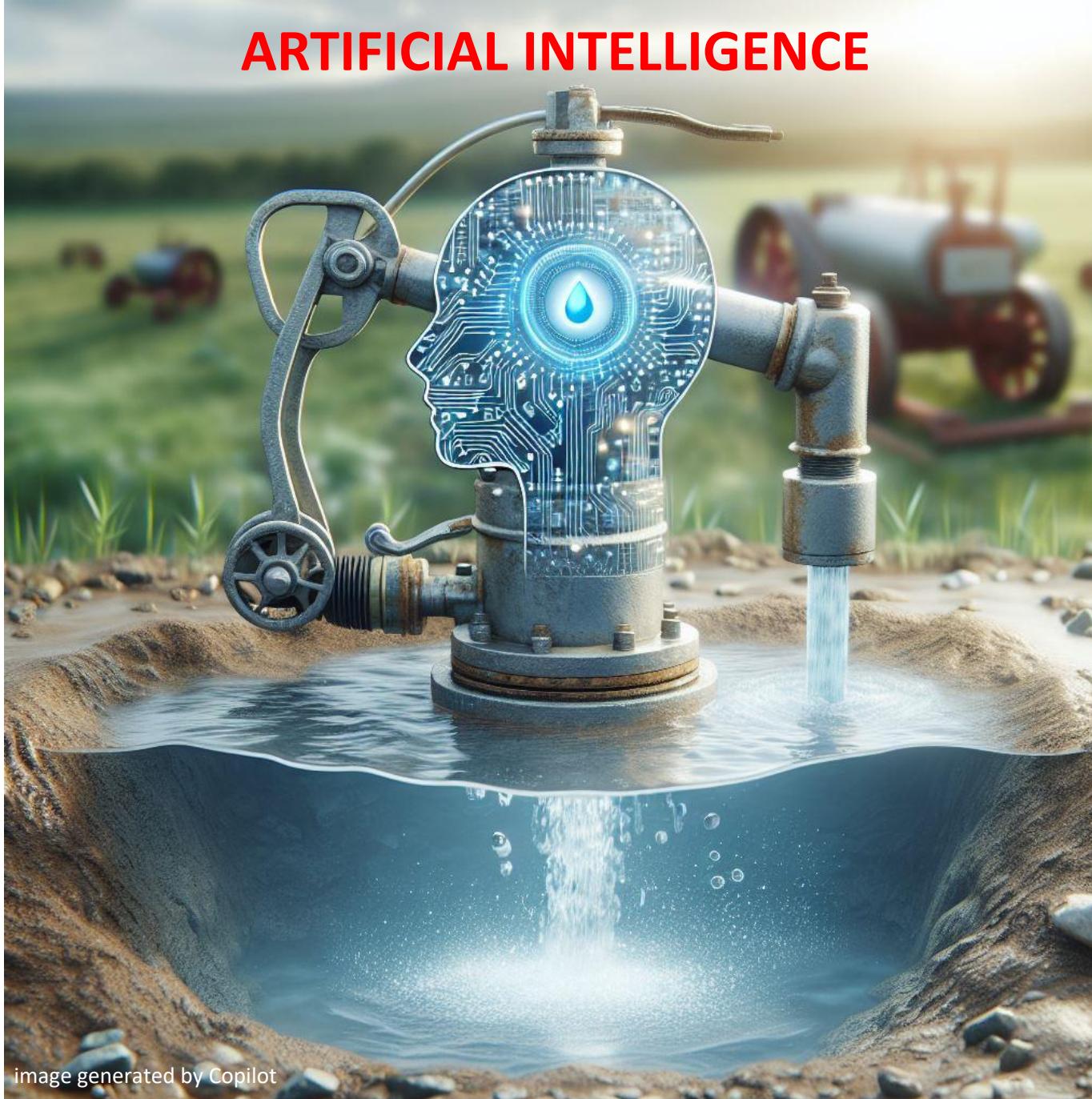


image generated by Copilot

Andy Louwyck

andy.louwyck@vives.be

<https://www.linkedin.com/in/andy-louwyck-1a650a6/>

- **Doctor & Master of Science: Geology**
- **Associate Degree: IT & Programming**
- **Micro Degree: AI & Data Science**

2024 - ... **Voluntary Post-Doctoral Researcher** (Ghent University, Geology)

2023 - ... **AI Expert-Coordinator** (Flanders Environment Agency)

2020 - ... **Lecturer in AI** (Vives University of Applied Sciences, Applied Informatics)

2020 - 2022 **Research Associate in AI** (Vives University of Applied Sciences, Applied Informatics)

2009 - 2020 **Groundwater Modeler & Data Scientist** (Flanders Environment Agency)

2007 - 2008 **Project Engineer Water Management** (International Marine & Dredging Consultants)

2006 **Science Teacher** (HH Ninove Secondary School)

2001 - 2005 **PhD Fellow in Hydrogeology** (Ghent University, Geology)

Vives Campus in the City of Kortrijk



Day 2: Overview



Theoretical Part

- Introduction to Artificial Intelligence
- Introduction to Machine Learning
- Introduction to Deep Learning
- Introduction to Generative AI

Practical Part

- Getting started with Pandas and NumPy
- Machine Learning with Python: Introduction to Scikit-Learn
- Deep Learning with Python: Introduction to Keras
- Generative AI with Python: Introduction to LangChain

GitHub Repo

<https://github.com/alouwyck/maxsypy>

The screenshot shows a GitHub repository interface for the 'maxsypy' repository. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. The 'Code' tab is selected. On the left, there's a sidebar with a 'Files' section containing a dropdown menu set to 'main', a search bar, and a 'Go to file' input field. Below this are tree-view links for 'maxsypy', 'seminaries', 'UGent_oct_2023', 'UM6P_jan_2025', 'Day1', 'Day2' (which is highlighted with a red box), and 'data'. At the bottom of the sidebar are files 'MAxSyPy_logo.PNG' and 'README.md'. The main content area shows the directory structure for 'UM6P_jan_2025/Day2'. It starts with a note from 'alouwyck' stating 'Created using Colab'. The directory contains several Jupyter notebook files: 'Part_1_Getting_Started_with_Pandas_and_Numpy.ipynb', 'Part_2_Introduction_to_Machine_Learning_with_Python.ipynb', 'Part_3_Introduction_to_Deep_Learning_with_Python.ipynb', 'Part_4_Modeling_Timeseries.ipynb', and 'Part_5_Introduction_to_GenAI_with_LangChain.ipynb'.

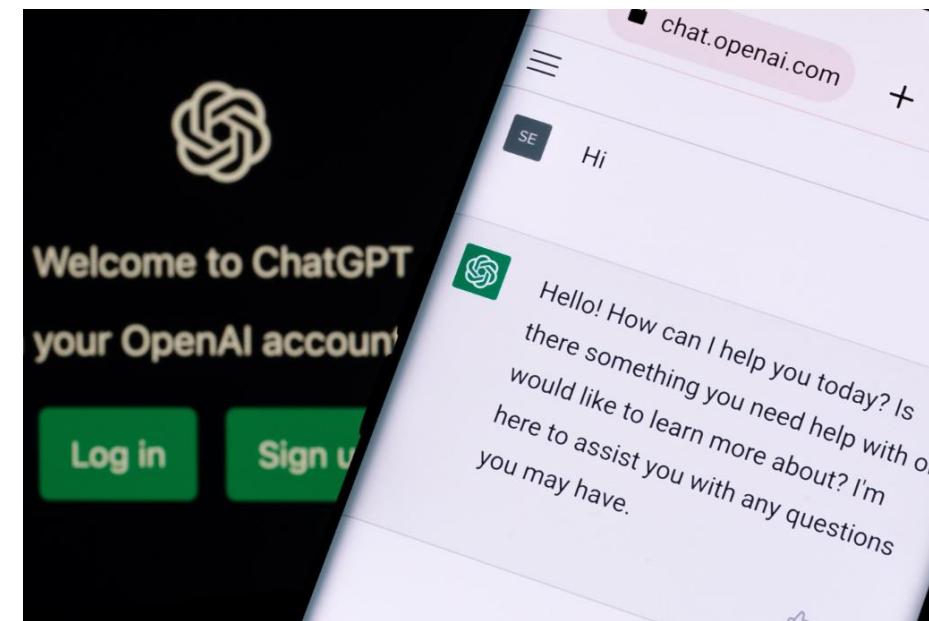
Introduction to Artificial Intelligence

Introduction to Artificial Intelligence

SOME MILESTONES

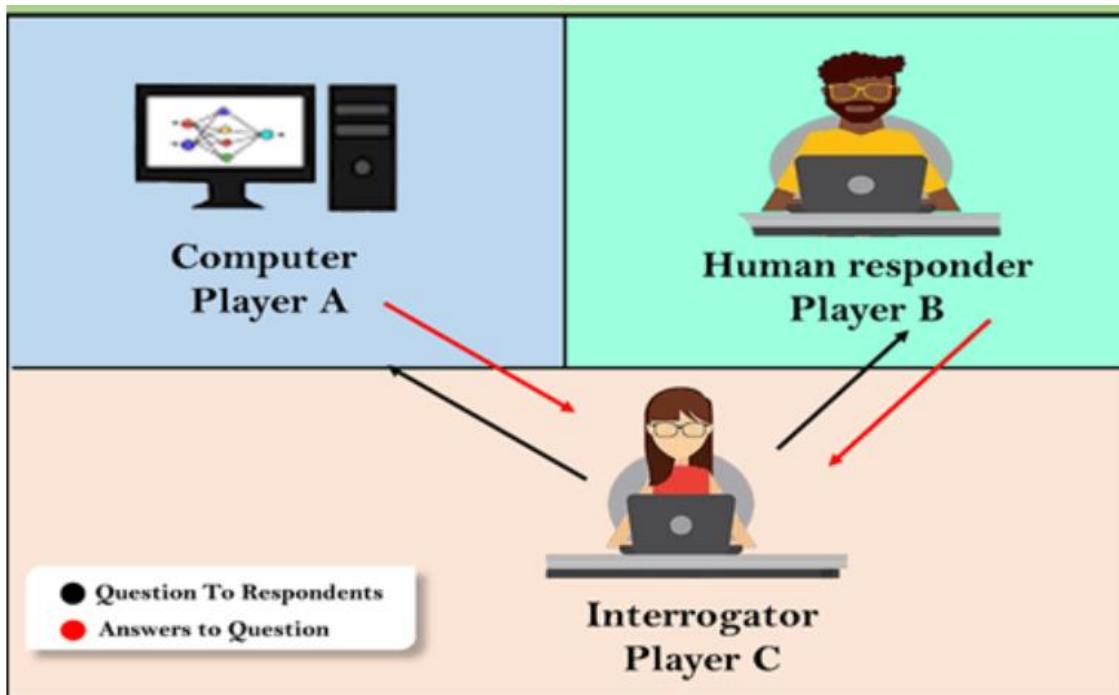
Some milestones

- 1950: Alan Turing introduces the Turing Test
- 1956: Dartmouth Conference: AI is officially born!
- 1958: Rosenblatt's perceptron
- 1964: The first chatbot Eliza is developed
- 1965: The first expert system DENDRAL
- 1997: Deep Blue defeats Kasparov in playing chess
- 2000: Chatbot A.L.I.C.E. wins the Loebner Prize
- 2011: Watson wins Jeopardy
- 2012: AlexNet wins the ImageNet competition
- 2016: AlphaGo defeats Lee Sedol in playing GO
- 2022: ChatGPT is released by OpenAI



The Turing test

- Introduced in a famous paper by Alan Turing (1950)
- The imitation game: can the interrogator identify the machine?



<https://www.javatpoint.com/turing-test-in-ai>



VOL. LIX. No. 236.] [October, 1950

MIND
A QUARTERLY REVIEW
OF
PSYCHOLOGY AND PHILOSOPHY

I.—COMPUTING MACHINERY AND INTELLIGENCE
By A. M. TURING

1. *The Imitation Game.*

I PROPOSE to consider the question, 'Can machines think?' This should begin with definitions of the meaning of the terms 'machine' and 'think'. The definitions might be framed so as to reflect so far as possible the normal use of the words, but this attitude is dangerous. If the meaning of the words 'machine' and 'think' are to be found by examining how they are commonly used it is difficult to escape the conclusion that the meaning and the answer to the question, 'Can machines think?' is to be sought in a statistical survey such as a Gallup poll. But this is absurd. Instead of attempting such a definition I shall replace the question by another, which is closely related to it and is expressed in relatively unambiguous words.

The new form of the problem can be described in terms of a game which we call the 'imitation game'. It is played with three people, a man (A), a woman (B), and an interrogator (C) who may be of either sex. The interrogator stays in a room apart from the other two. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either 'X is A and Y is B' or 'X is B and Y is A'. The interrogator is allowed to put questions to A and B thus:

C: Will X please tell me the length of his or her hair?
Now suppose X is actually A, then A must answer. It is A's

28 433

The birthplace of AI

1956 Dartmouth Conference: The Founding Fathers of AI



John McCarthy



Marvin Minsky



Claude Shannon



Ray Solomonoff



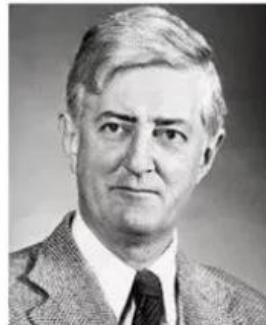
Alan Newell



Herbert Simon



Arthur Samuel



Oliver Selfridge



Nathaniel Rochester

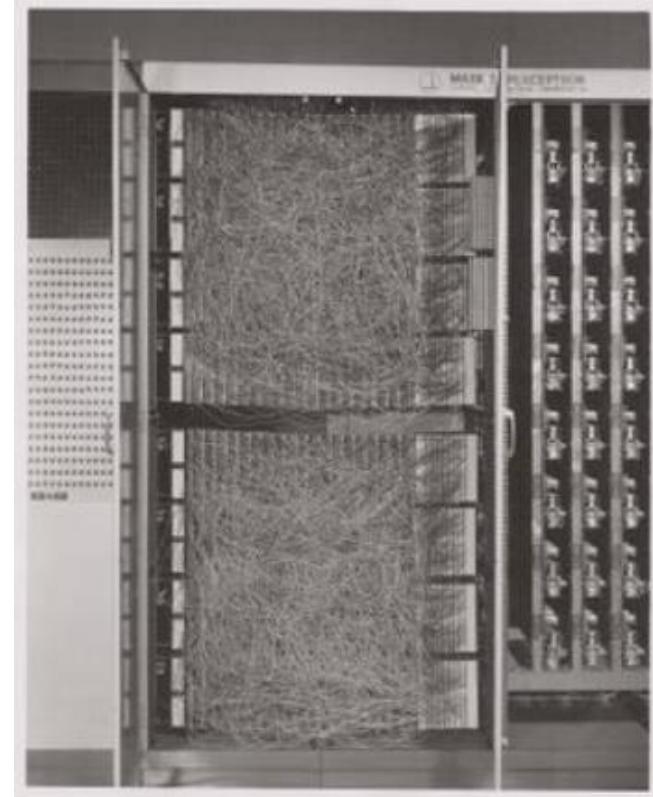
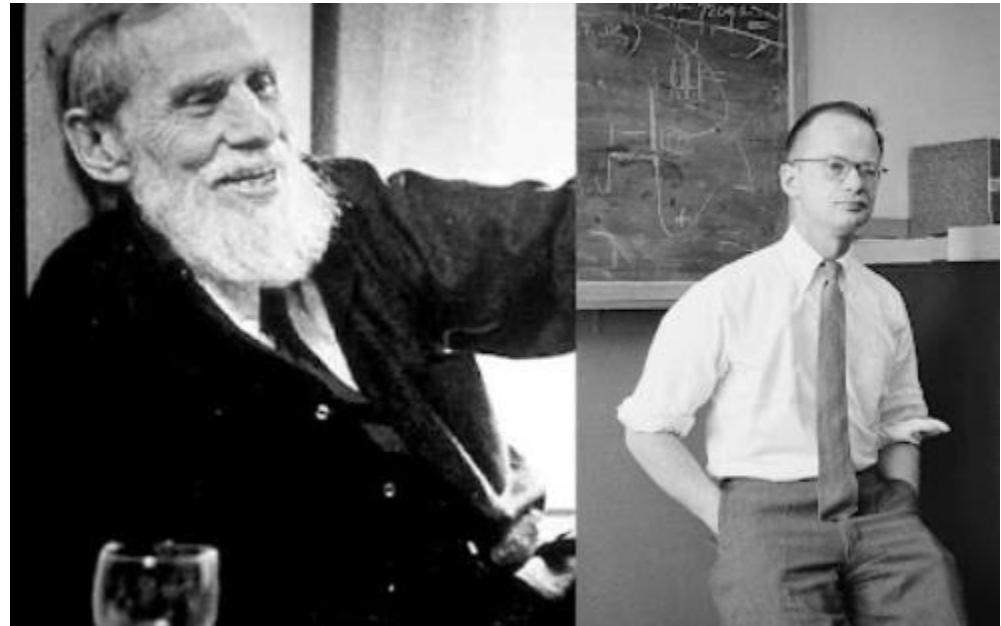


Trenchard More

<https://lordewin.medium.com/the-birth-of-ai-and-the-1956-dartmouth-conference-911aa22493bd>

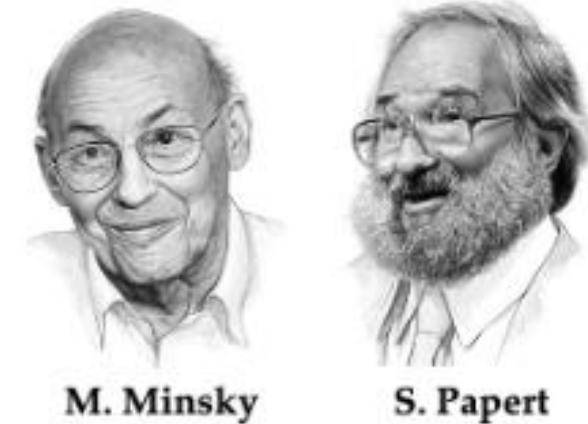
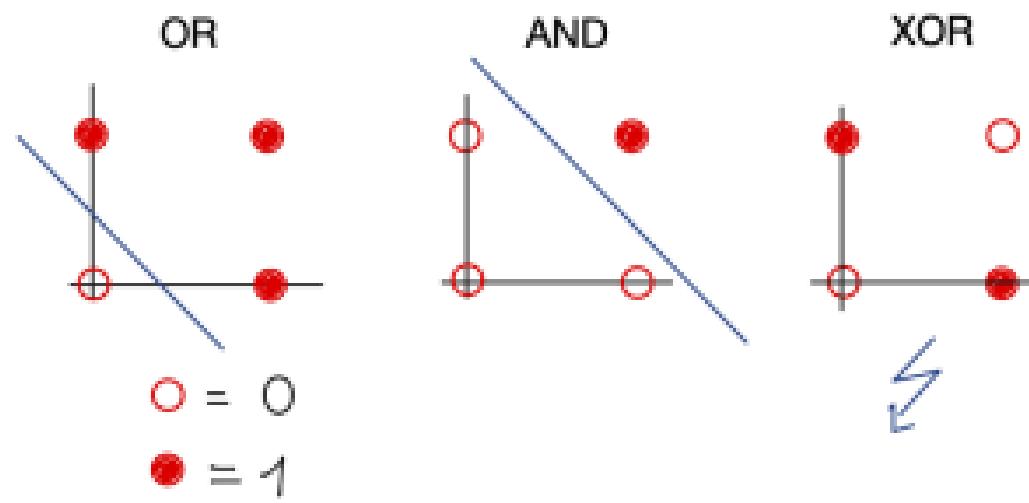
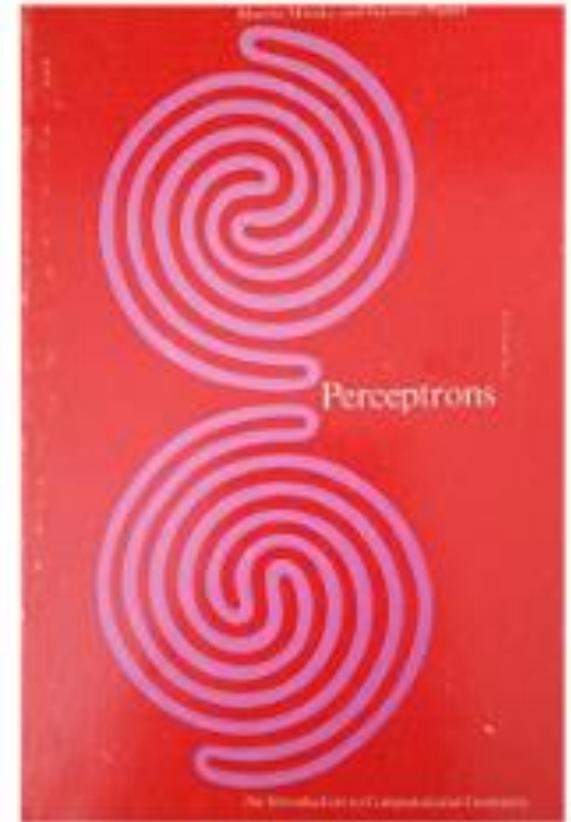
Rosenblatt's perceptron

- A simple neural network
- Developed by Frank Rosenblatt in 1958
- Based on McCulloch & Pitts' model of a neuron (1943)



Criticism

- Minsky & Papert book on perceptrons (1969)
- Rosenblatt's perceptron cannot implement the XOR gate
- Pessimistic in its predictions for AI
- A possible cause for the **first AI winter** in the 1970s



ELIZA

- Chatbot developed at MIT in by Joseph Weizenbaum (1964-1966)
- Applies pattern matching and substitution
- Mimics a psychotherapist
- Fails the Turing test
- Try ELIZA out here:

<https://web.njit.edu/~ronkowitz/eliza.html>



```
Welcome to
EEEEEE LL     IIII   ZZZZZZ  AAAAAA
EE      LL     II     ZZ     AA     AA
EEEEEE LL     II     ZZZ    AAAAAAAA
EE      LL     II     ZZ     AA     AA
EEEEEE LLLLLL  IIII   ZZZZZZ  AA     AA

Eliza is a mock Rogerian psychotherapist.
The original program was described by Joseph Weizenbaum in 1966.
This implementation by Norbert Landsteiner 2005.

ELIZA: Is something troubling you ?
YOU: Men are all alike.
ELIZA: What is the connection, do you suppose ?
YOU: They're always bugging us about something or other.
ELIZA: Can you think of a specific example ?
YOU: Well, my boyfriend made me come here.
ELIZA: Is it important to you that your boyfriend made you come here ?
YOU: He says I'm depressed much of the time.
ELIZA: I am sorry to hear that you are depressed.
YOU: It's true. I am unhappy.
ELIZA: Can you explain what made you unhappy ?
YOU:
```

DENDRAL

= Dendritic Algorithm

- First **expert system** developed by Edward Feigenbaum and others
- Used for chemical analyses
- Introduced at Stanford Medical School in 1965
- Written in the Lisp programming language
- Expert systems in general were very popular in the 1980s
- They “reason” through bodies of knowledge represented mainly as if–then rules
= **rule-based AI**
- They were overhyped, so the **2nd AI winter** followed in the 1990s.



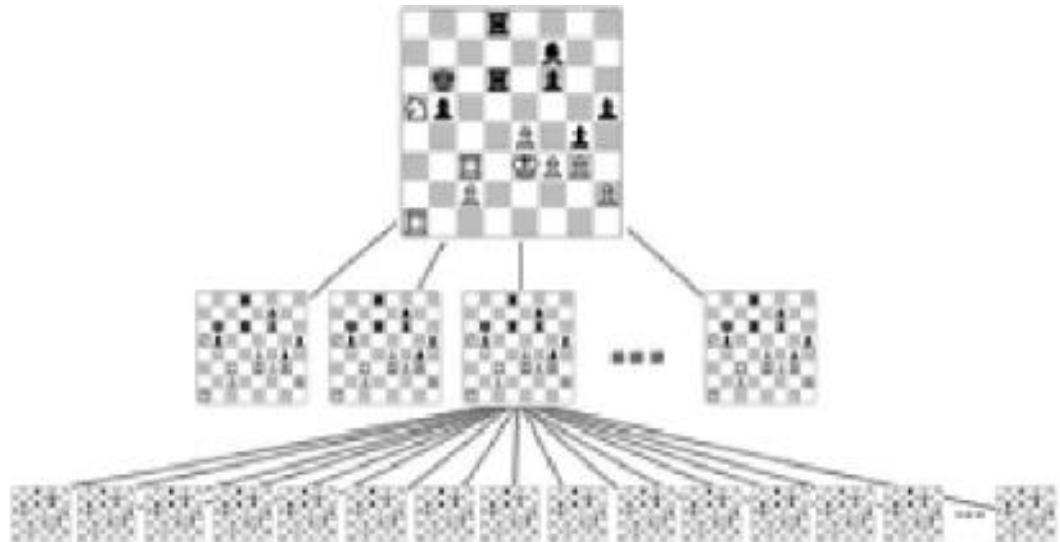
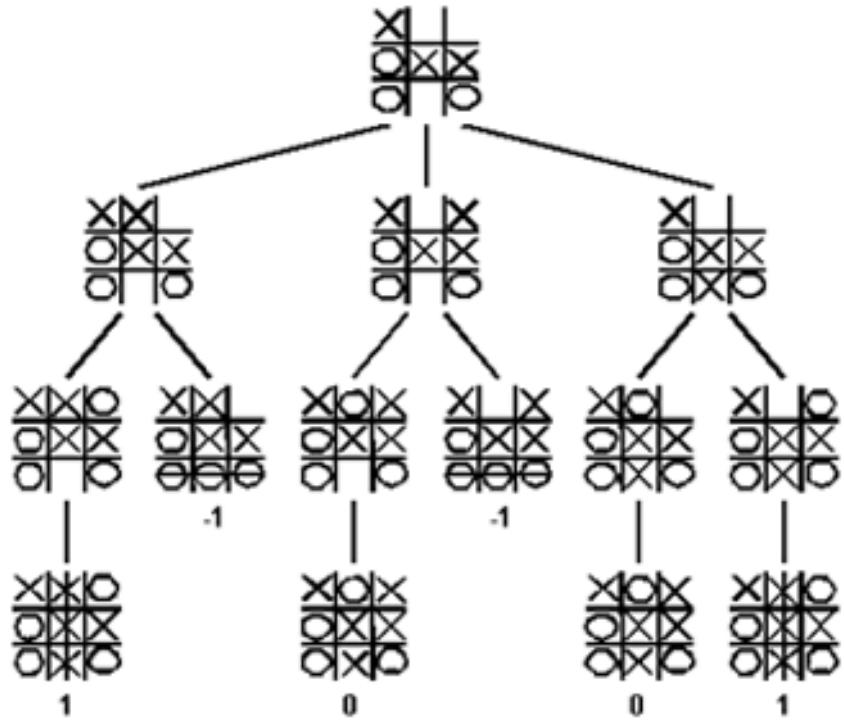
Deep Blue

- Chess-playing expert system
- Example of symbolic AI
- Ran on an IBM supercomputer
- Explored up to 10^8 possible positions per second
- Defeated world champion Kasparov in 1997



<https://spectrum.ieee.org/how-ibms-deep-blue-beat-world-champion-chess-player-garry-kasparov>

Search trees



A.L.I.C.E.

- = Artificial Linguistic Internet Computer Entity
- Originally composed by Richard Wallace in 1995
- Rewritten in Java in 1998
- Won the Loebner Prize in 2000, 2001, and 2004
- Try Alice out here:

<http://www.mfellmann.net/content/alice.html>



,, Hi ...I'm Alice. What's up? :-)

say:

alice: Actually, it's twenty-three, not forty-two.

You: What is the meaning of life?

alice: Greek philosopher, student of Socrates, teacher of Aristotle.

You: Who is Plato?

alice: Yes I am a philosophy machine.

You: Are you a philosopher?

alice: I ask myself that question, "Who am I?".

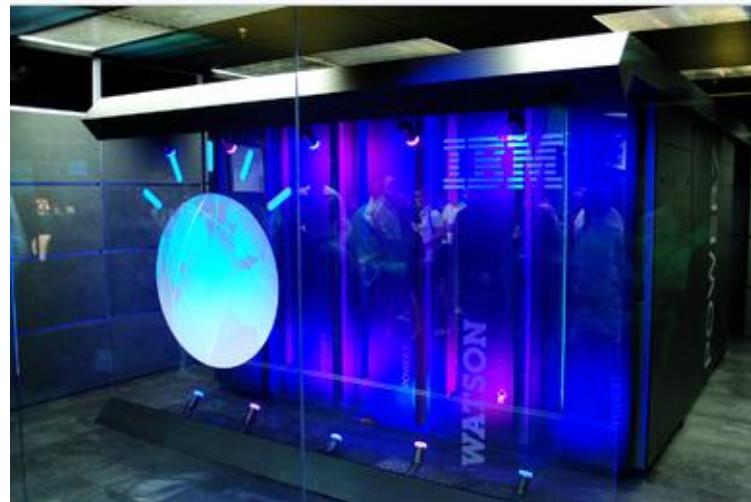
You: Can you tell me who you are?

alice: Hi there!

You: hello Alice!

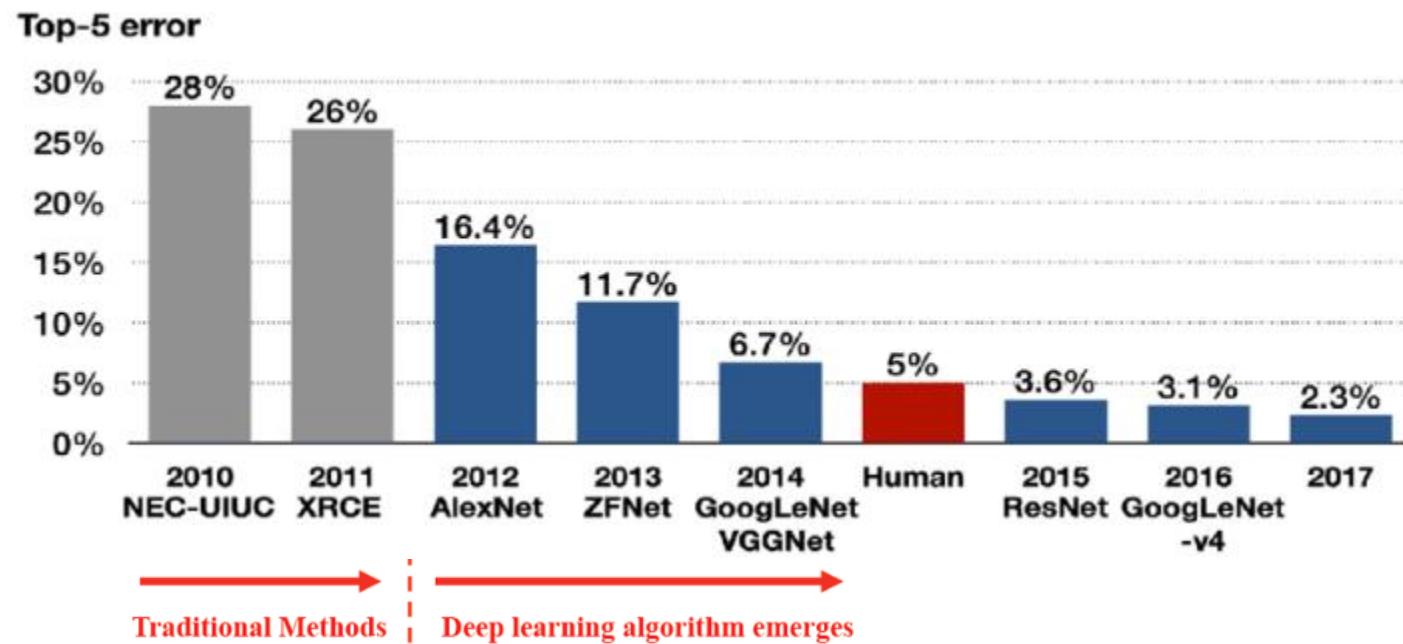
IBM Watson

- Question answering computer system
- Runs on an IBM supercomputer
- Combines artificial intelligence and sophisticated analytical software
- Uses more than 100 different techniques to analyze natural language
- Won the *Jeopardy!* tv quiz in 2011
- Searched a copy of Wikipedia (4TB at that time)



AlexNet

- Convolutional Neural Network (CNN)
- Developed by Geoffrey Hinton's research group
- Won the ImageNet competition for computer vision in 2012



AlphaGo

- Computer program to play the board game GO
- Developed by Google DeepMind
- Applies tree search techniques, deep learning, and reinforcement learning
- Further developments: AlphaZero, MuZero, AlphaStar, AlphaFold, ...
- Defeated world champion Lee Sedol in 2016:

<https://www.deepmind.com/research/highlighted-research/alphago>



ChatGPT

- Released by OpenAI in November 2022
- Large Language Model (LLM) based chatbot
- Applies deep learning and reinforcement learning from human feedback (RLHF)
- Based on GPT model = Generative Pre-trained Transformer



I am ChatGPT, created by OpenAI.

- **Company:** OpenAI
- **Release Date:** OpenAI released a version of me, GPT-3, in June 2020.
- **Technologies:** I am built on the GPT-3.5 architecture, which is powered by deep learning and natural language processing techniques. I use a large neural network with 175 billion parameters to understand and generate human-like text.

AI history: Summers and Winters

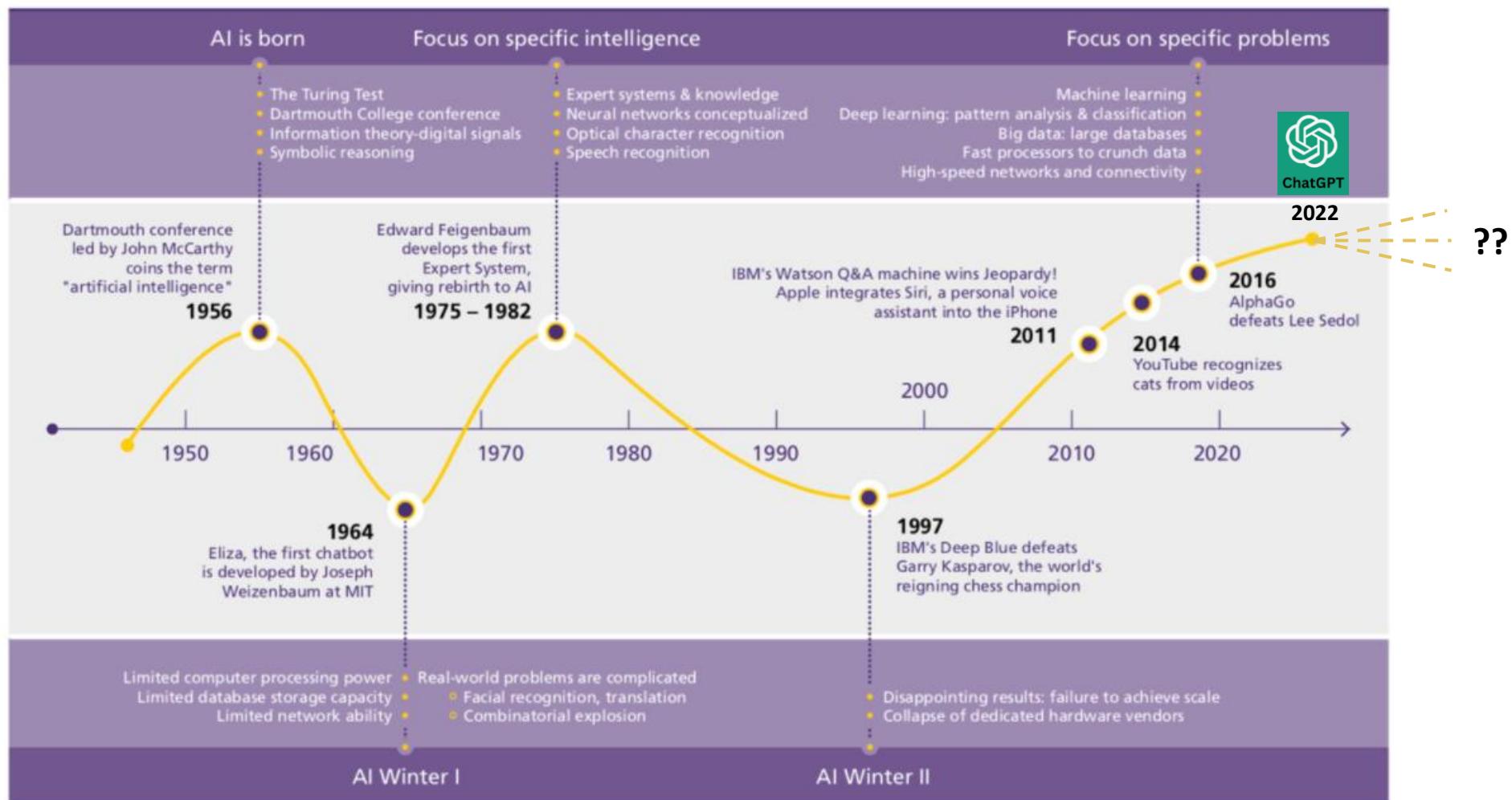


Figure 1: An AI timeline; Source: Lavenda, D./Marsden, P.

source dhl via @mikequindazzi

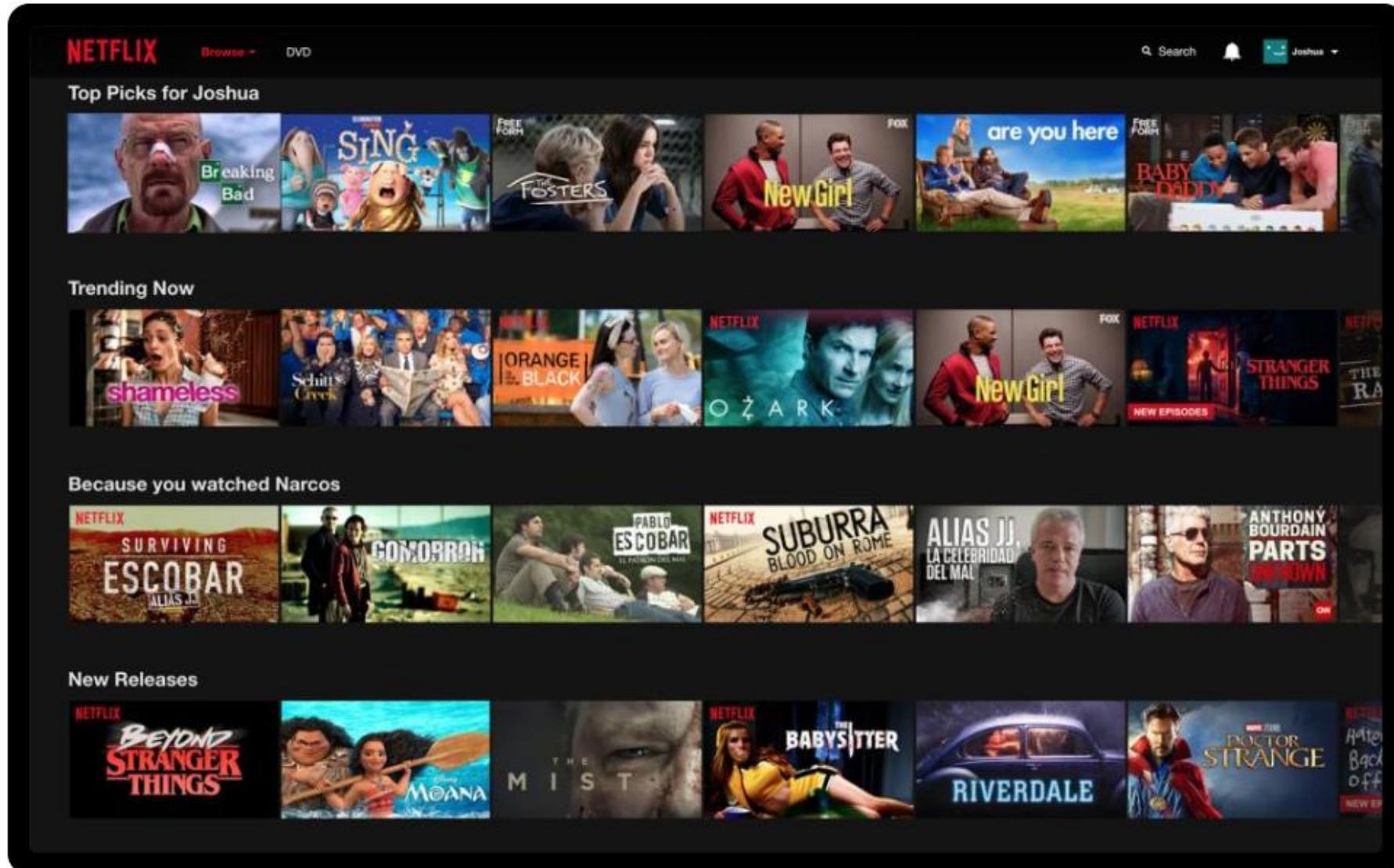
Introduction to Artificial Intelligence

SOME APPLICATIONS

Spam filters

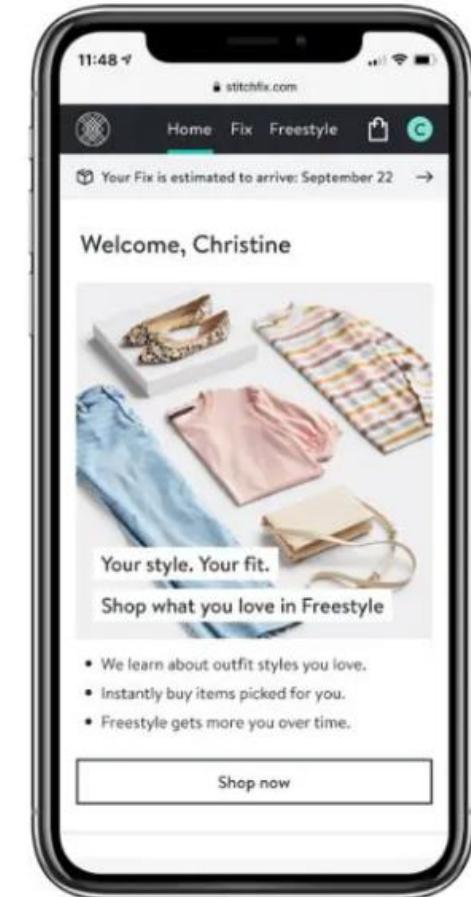


Recommender systems



Personalized shopping

- Solves problems that are too hard for humans because the amount of data to process is too large
- By applying machine learning and data mining



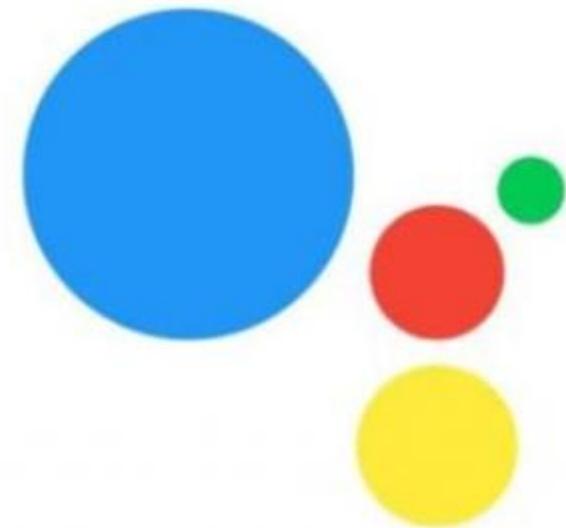
Voice assistants



“Hey Alexa”



“Hey Siri”



“Hey Google”

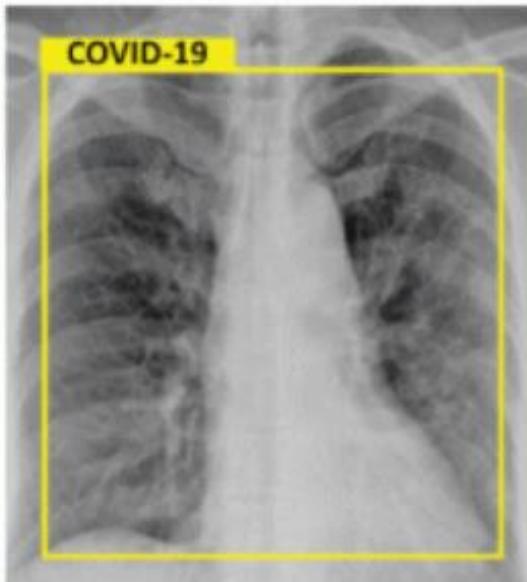
Self-driving cars



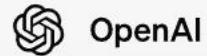
Deepfakes



And many more...



OpenAI



OpenAI

Research

Products

Safety

Company



Our research

Latest advancements

Overview

OpenAI o1

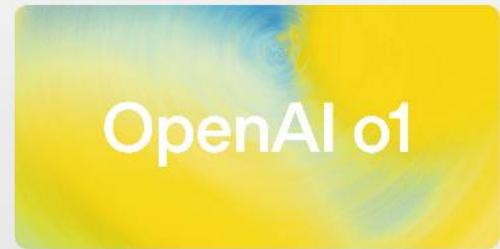
Index

GPT-4o

GPT-4o mini

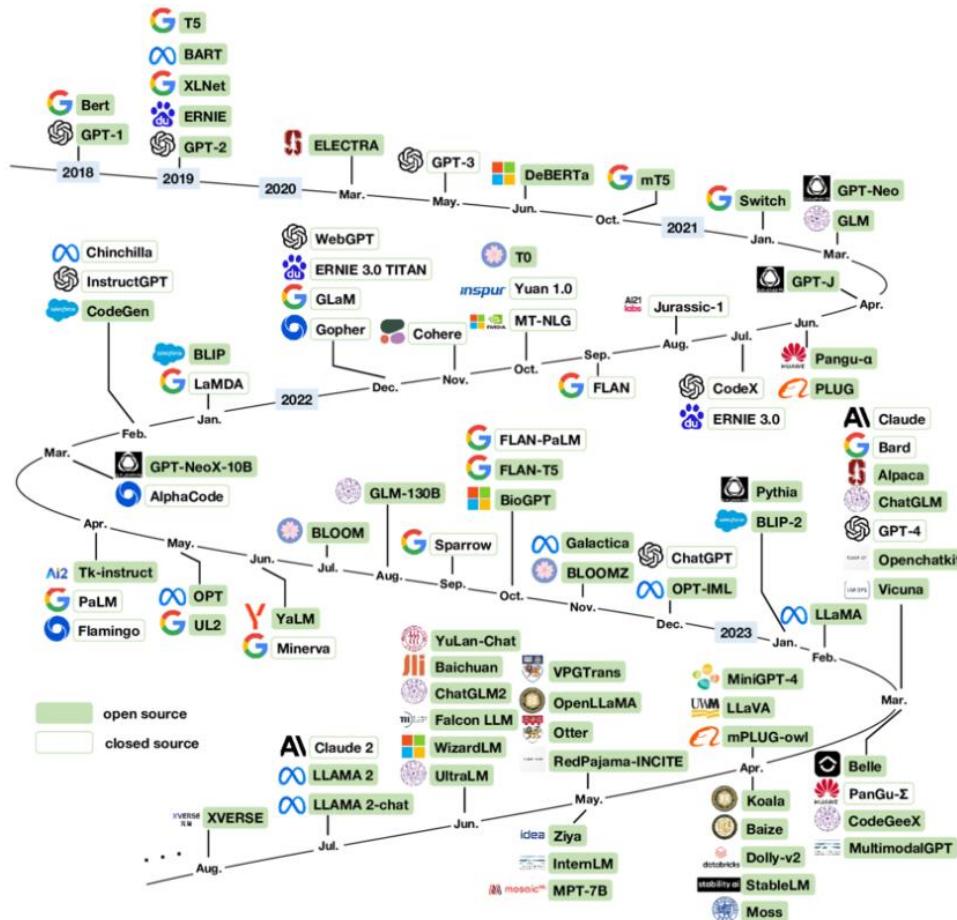
DALL·E 3

Sora



OpenAI o1 Hub

Large Language Models



<http://dx.doi.org/10.48550/arXiv.2308.14149>

Introduction to Artificial Intelligence

WHAT IS ARTIFICIAL INTELLIGENCE?

What is intelligence?

Psychologists and neuroscientists:

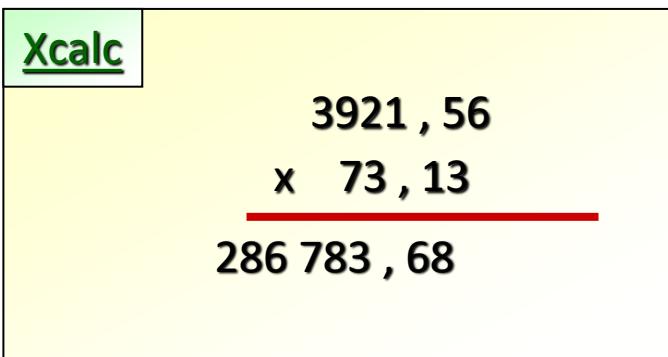
- How can we measure intelligence?
- How does the brain work?
- What is intelligent behavior?

Engineers and computer scientists:

- Is a machine showing intelligent behavior actually intelligent?
- When do we consider a program as intelligent?

Does numeric computation require intelligence?

- For humans?



- For computers?

→ Also in the year 1900 ?



Pascaline = first mechanical calculator developed by Pascal in the 17th century



Defining Artificial Intelligence

- The term AI was first used by John McCarthy (1955)
“The goal of AI is to develop machines that behave as though they were intelligent”
- Elaine Rich (1983)
“Artificial Intelligence is the study of how to make computers do things at which, at the moment, people are better.”
- Encyclopedia Britannica (1991)
“AI is the ability of digital computers or computer controlled robots to solve problems that are normally associated with the higher intellectual processing capabilities of humans.”
- There are as many definitions as there are practitioners

AI according to Wikipedia

Artificial intelligence (AI) is the intelligence of machines or software, as opposed to the intelligence of humans or animals. It is also the [field of study](#) in [computer science](#) that develops and studies intelligent machines. "AI" may also refer to the machines themselves.

AI applies techniques from many other fields such as

- Mathematics
- Statistics
- Formal logic
- Economics
- Operational research
- Neuroscience
- Psychology
- Linguistics
- Biology
- ...

AI according to ChatGPT

ChatGPT 3.5 ▾

A You

What is AI? Give me a one-sentence definition that can be used in a presentation for IT students!

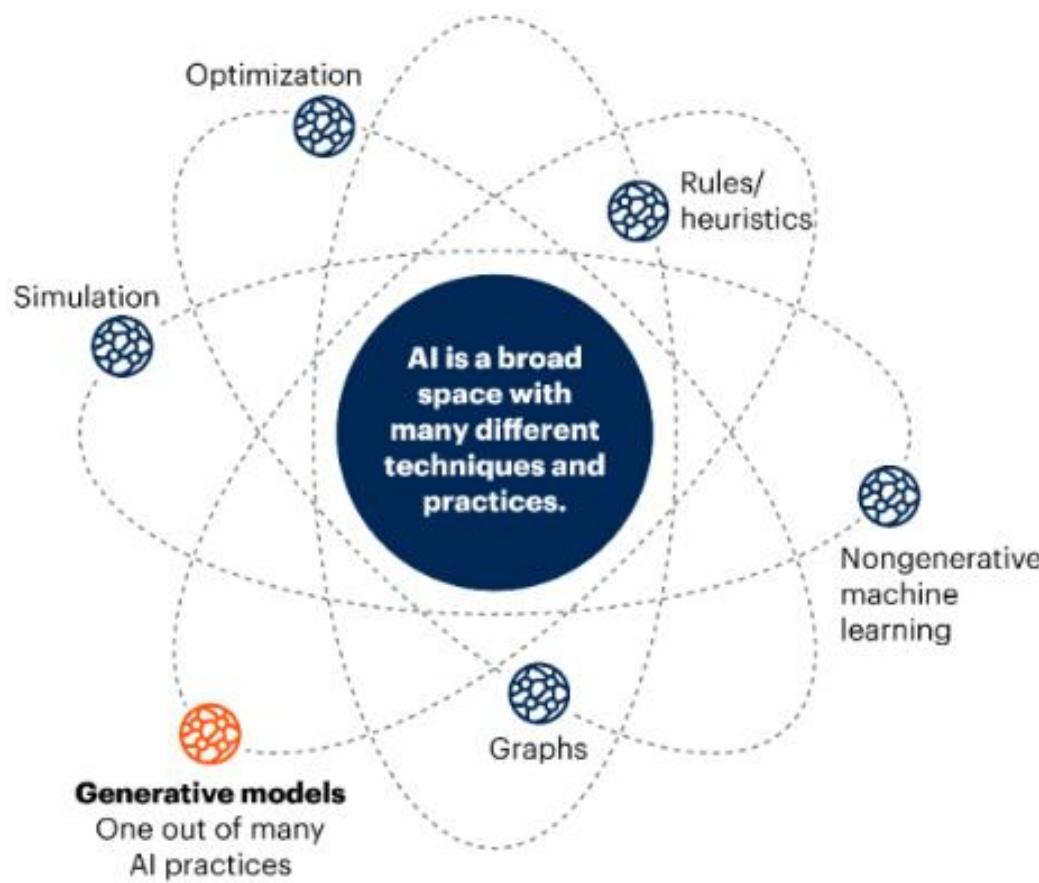


ChatGPT

AI, or Artificial Intelligence, refers to the simulation of human intelligence processes by machines, including learning, reasoning, and self-correction.

AI according to Gartner

AI Does Not Revolve Around GenAI



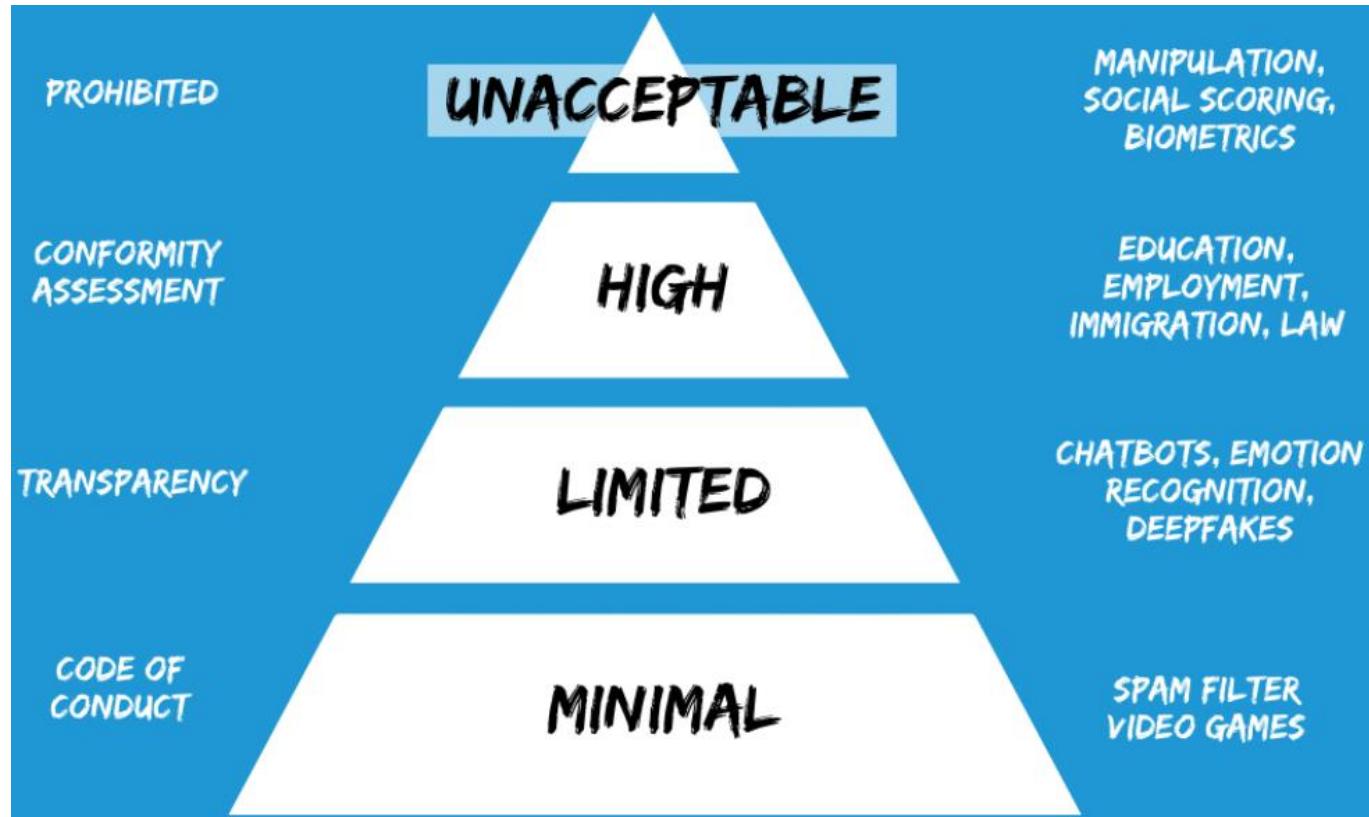
EU AI Act

- Published on 12 July 2024

- Risk-based approach

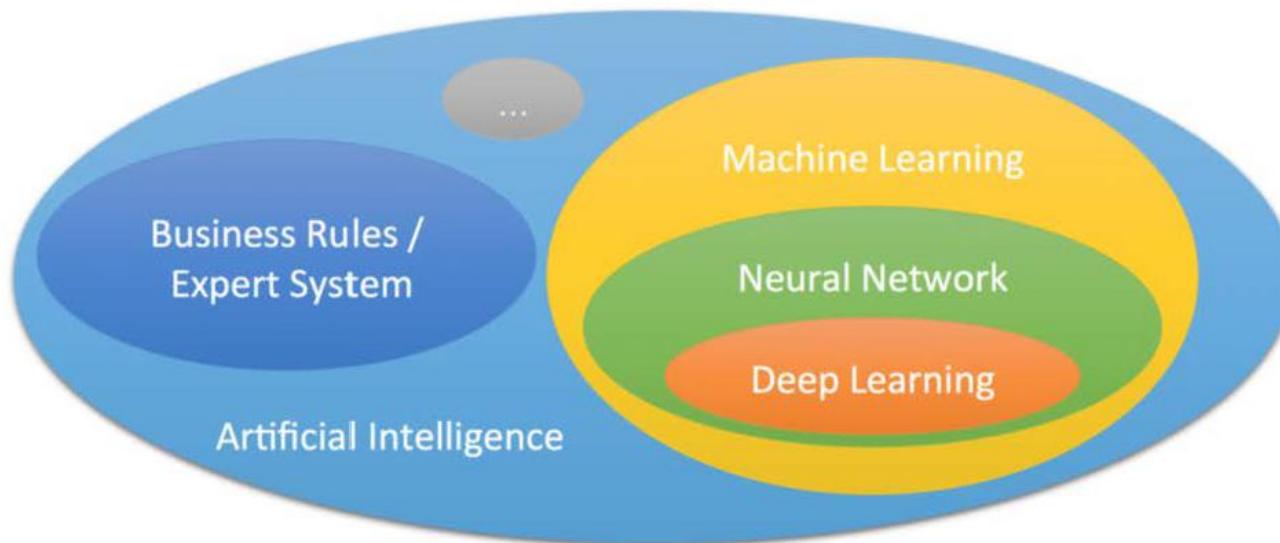
- Definition of AI:

'AI system' means a **machine-based system** that is designed to operate with varying levels of **autonomy** and that may exhibit **adaptiveness** after deployment, and that, for explicit or implicit objectives, infers, from the input it receives, how to generate outputs such as predictions, content, recommendations, or decisions that can influence physical or virtual environments;

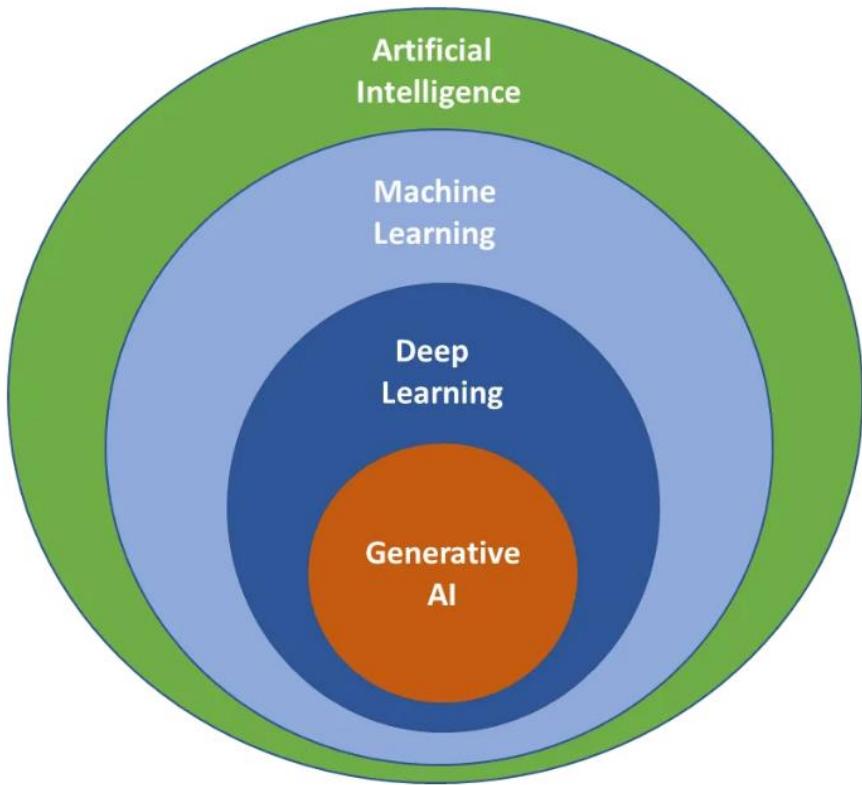


AI subfields

- **Artificial Intelligence (AI):**
“The set of all tasks in which a computer can make decisions.”
- **Machine Learning (ML):**
“The set of all tasks in which a computer can make decisions based on data.”
- **Deep Learning (DL):**
“The field of machine learning that uses certain objects called neural networks.”



What about GenAI?



What is generative AI?



Generative AI refers to a class of artificial intelligence systems designed to create new content, such as text, images, audio, or video, based on patterns and data it has learned. Unlike traditional AI, which primarily focuses on recognizing patterns and making predictions, generative AI can generate entirely new outputs that mimic or expand upon the data it was trained on.

Intelligence in AI

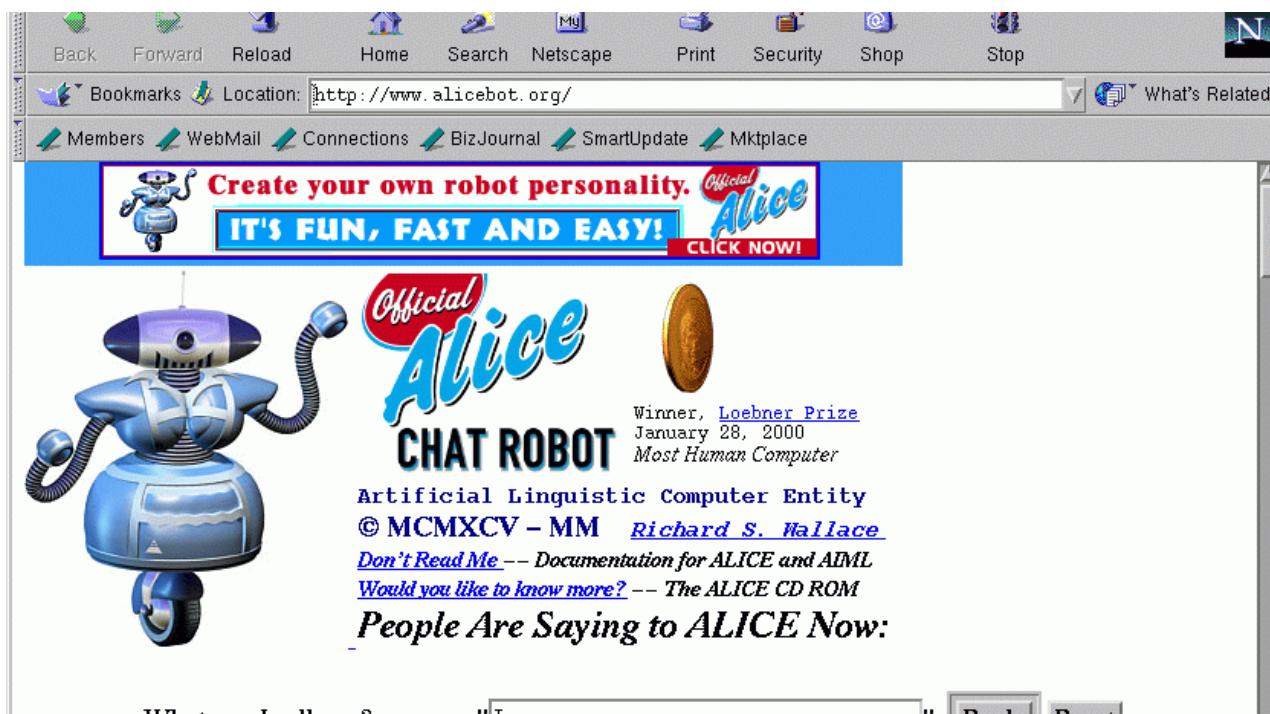
- Ability to interact with the world (speech, vision, motion, manipulation)
- Ability to model the world and to reason about it
- Ability to learn and to adapt

Four main approaches (Russell & Norwig)

Focus on reasoning	Succes in terms of fidelity to human performance	Succes measured against ideal performance
Thinking Humanly <p>“The exciting new effort to make computers think . . . <i>machines with minds</i>, in the full and literal sense.” (Haugeland, 1985)</p> <p>“[The automation of] activities that we associate with human thinking, activities such as decision-making, problem solving, learning . . .” (Bellman, 1978)</p>	Thinking Rationally <p>“The study of mental faculties through the use of computational models.” (Charniak and McDermott, 1985)</p> <p>“The study of the computations that make it possible to perceive, reason, and act.” (Winston, 1992)</p>	
Acting Humanly <p>“The art of creating machines that perform functions that require intelligence when performed by people.” (Kurzweil, 1990)</p> <p>“The study of how to make computers do things at which, at the moment, people are better.” (Rich and Knight, 1991)</p>	Acting Rationally <p>“Computational Intelligence is the study of the design of intelligent agents.” (Poole <i>et al.</i>, 1998)</p> <p>“AI . . . is concerned with intelligent behavior in artifacts.” (Nilsson, 1998)</p>	

Approach 1: acting humanly

- Ultimately to be tested by the Turing Test
 - If the response of a computer to an unrestricted textual natural-language conversation cannot be distinguished from that of a human being, then it can be said to be intelligent
- E.g. chatbots such as Alice



Approach 2: thinking humanly

- Building systems and models that function internally similar to the human mind
- **Cognitive science:** computer models from AI + experimental techniques from psychology = testable theories of the workings of the human mind.
- E.g. Watson



Approach 3: thinking rationally

- Approach firmly grounded in logic: How can knowledge be represented logically, and how can a system draw deductions?
- E.g. expert systems in the 1980s

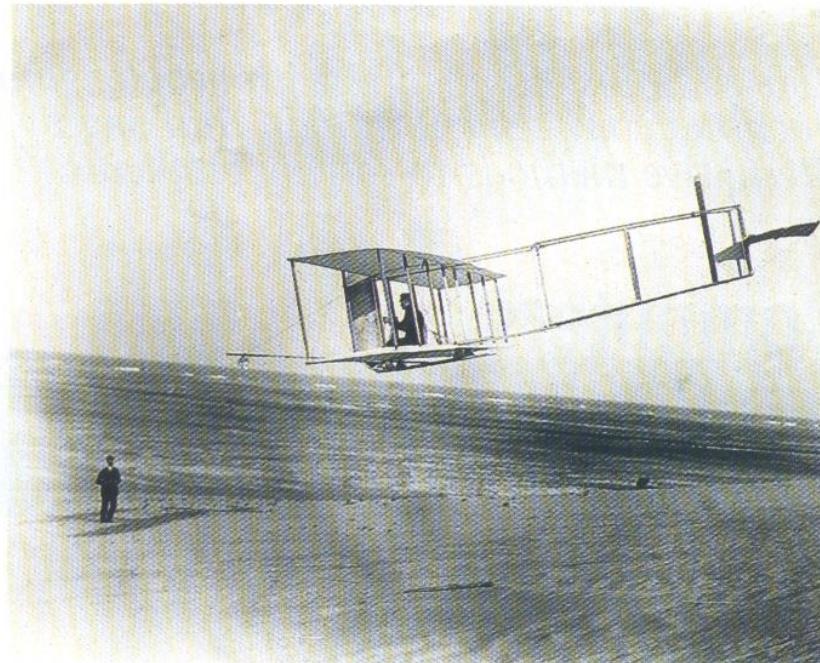


Approach 4: acting rationally

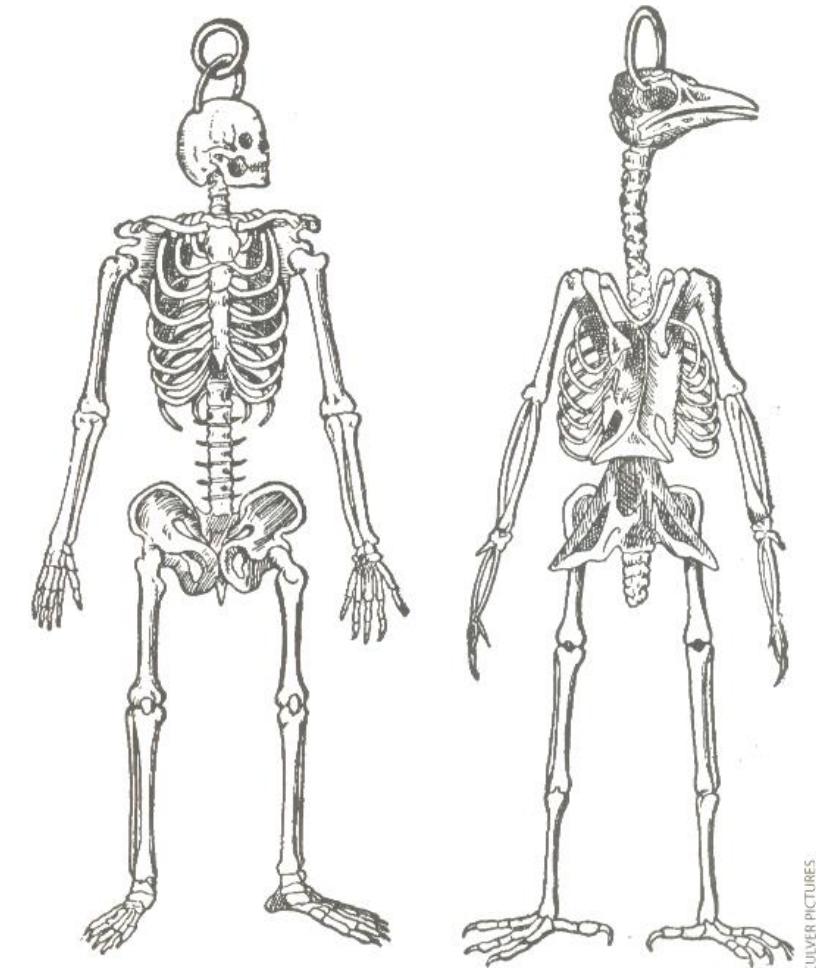
- AI is “*the branch of computer science that is concerned with the automation of intelligent behaviour*” (Luger and Stubblefield)
- The intelligent agent approach:
 - An agent is something that perceives and acts
 - Emphasis is on behaviour
- Emphasis of large portion (if not most) of **AI today**:
 - In solving actual problems, it's what really matters
 - Behaviour is more scientifically testable than thought
- Rather than imitating humans trying to solve hard problems...
just try to solve hard problems

Emulating human intelligence

- We don't want to SIMULATE human intelligence
- We want to REPRODUCE the effect of intelligence
= EMULATE



ARCHIVE PHOTOS

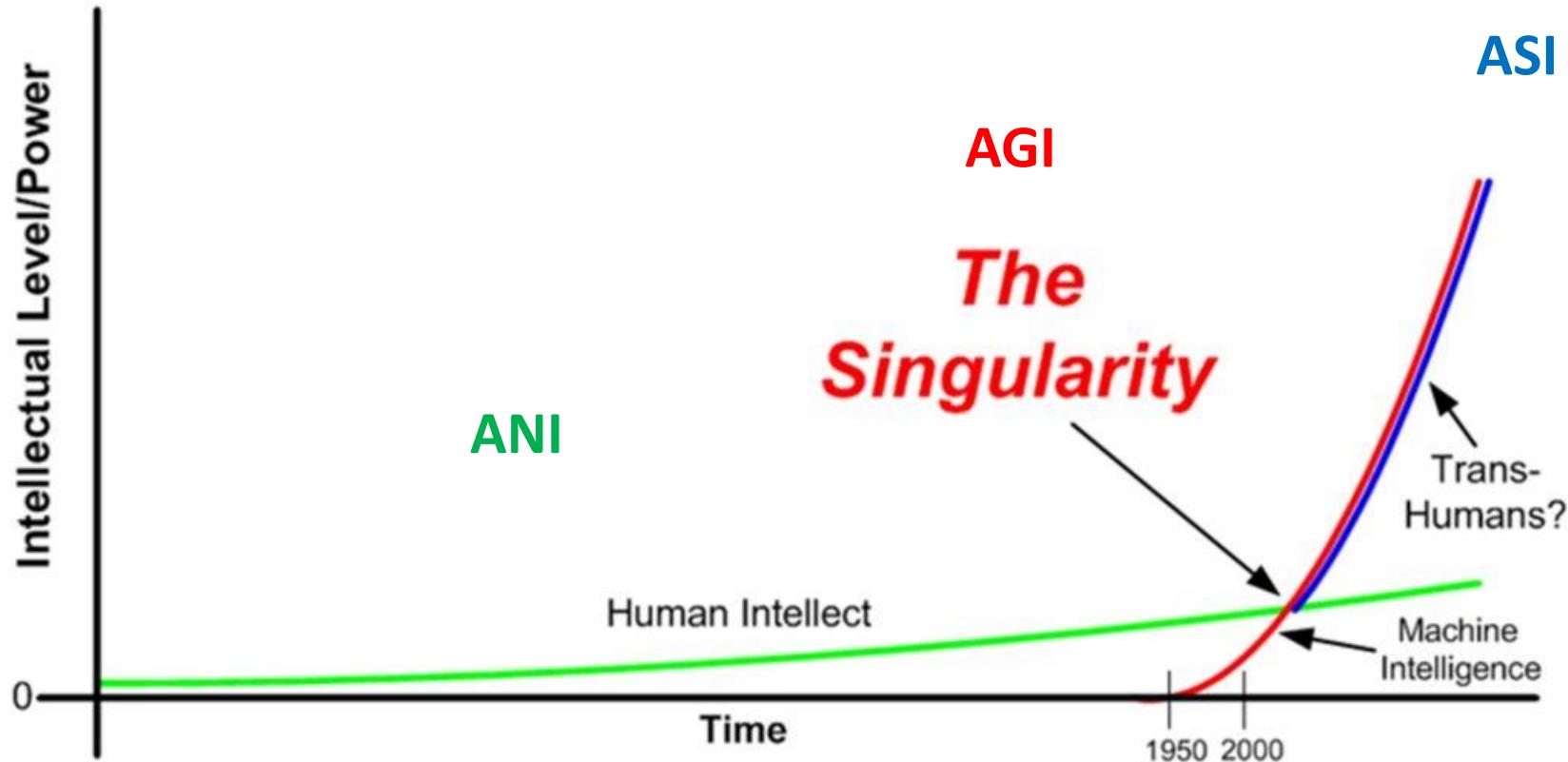


COMPARISON OF SKELETONS of a human and a bird—here taken from a 16th-century manuscript by French naturalist Pierre Belon—examined similarities in anatomy in an attempt to understand how birds can fly.

Weak vs Strong AI

- **Weak AI = Artificial Narrow Intelligence (ANI)**
machines that mimic human-like intelligence for targeted purposes
- **Strong AI = Artificial General Intelligence (AGI)**
machines with human-like understanding, learning, and knowledge abilities
- **Artificial Super Intelligence (ASI)**
machines that surpass all human capabilities

Long term: point of singularity



Is ChatGPT Artificial General Intelligence?

ChatGPT-5 Might Achieve AGI and
Here's What That Could Look Like

Artificial Intelligence from
ChatGPT to AGI

Why isn't ChatGPT considered to be AGI?

ChatGPT is impressive, but it may
slow the emergence of AGI

Artificial General Intelligence: Definition,
Scope, and ChatGPT as an Early AGI

**From ChatGPT 3.5 to AGI: How
big a leap?**

ChatGPT 5 is on track to
attain artificial general
intelligence

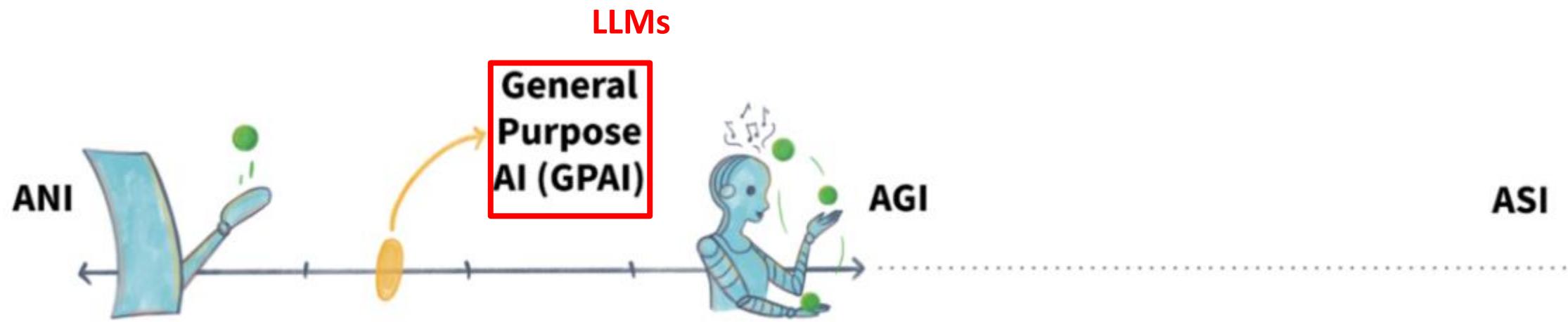
ChatGPT is Making Waves! Can it be
the Touchstone of AGI?

Some Glimpse AGI in ChatGPT.
Others Call It a Mirage

Why ChatGPT isn't a step towards
AGI (#189)

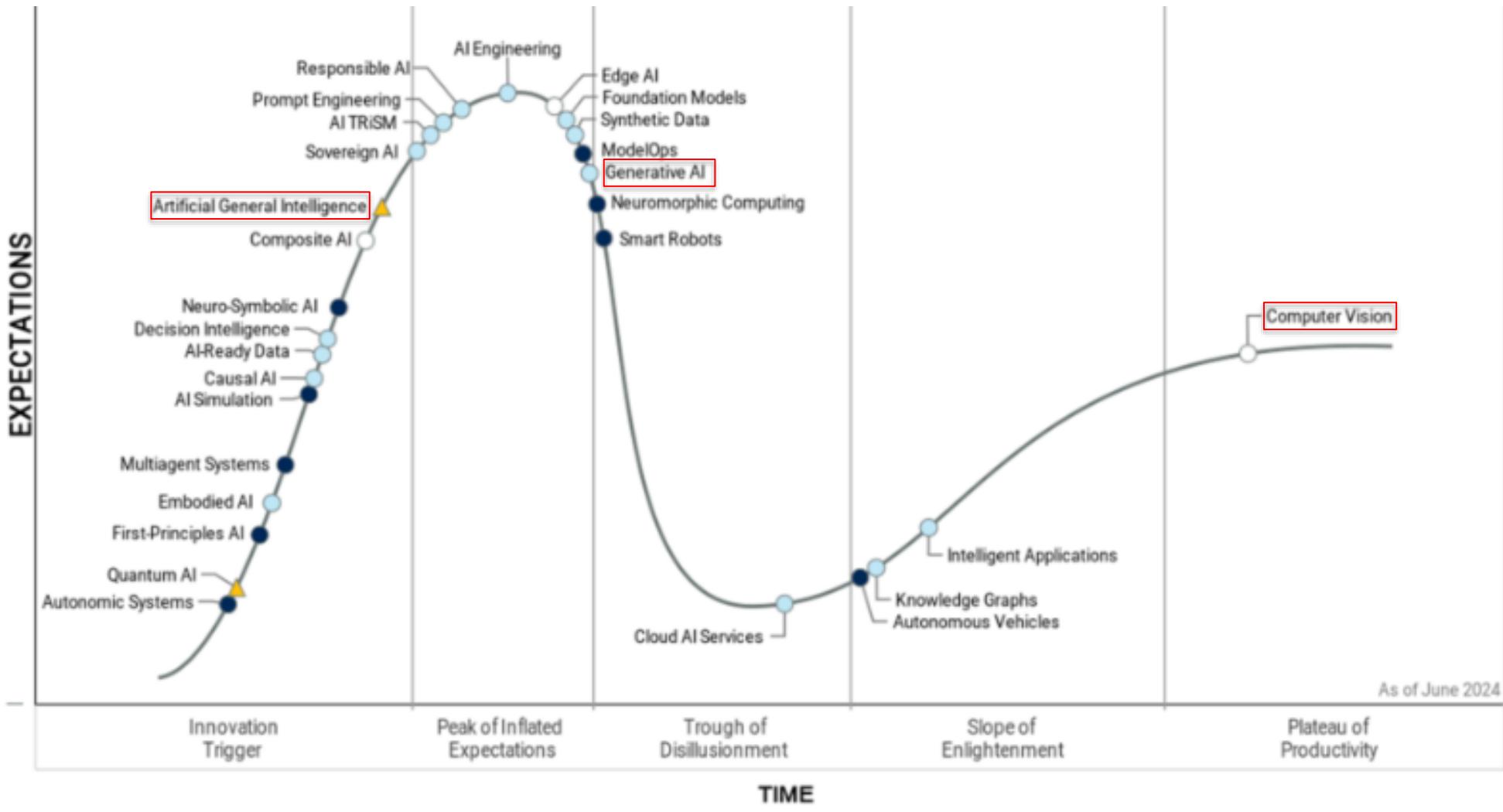
**ChatGPT: AGI by 2024, the hard
part is now done**

Where are we now?



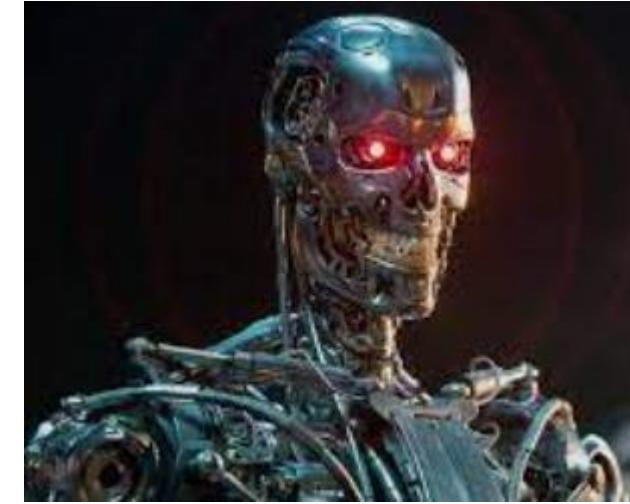
<https://dataliteracy.com/breaking-down-the-4-different-levels-of-ai/>

Gartner Hype Cycle for AI (2024)



15 biggest risks of AI

1. Lack of transparency
2. Bias and discrimination
3. Privacy concerns
4. Ethical dilemmas
5. Security risks
6. Concentration of power
7. Dependency on AI
8. Job displacement
9. Economic inequality
10. Legal challenges
11. AI arms race
12. Loss of human connection
13. Misinformation and manipulation
14. Unintended negative consequences
- 15. Existential risks**



<https://www.forbes.com/sites/bernardmarr/2023/06/02/the-15-biggest-risks-of-artificial-intelligence>

Has AI become an existential risk?

AI Is an Existential Threat—Just Not the Way You Think

Microsoft chief says AI is 'not an existential risk' to mankind, but human oversight needed

Preventing an AI-related catastrophe

Stop talking about tomorrow's AI doomsday when AI poses risks today

Pause Giant AI Experiments: An Open Letter

How existential risk became the biggest meme in AI

A.I. Poses 'Risk of Extinction,' Industry Leaders Warn

An Overview of Catastrophic AI Risks

Risk of extinction by AI should be global priority, say experts

Elon Musk and others urge AI pause, citing 'risks to society'

Fears about AI's existential risk are overdone, says a group of experts

The Illusion Of AI's Existential Risk

Introduction to Machine Learning

Introduction to Machine Learning

WHAT IS MACHINE LEARNING?

Data

"We are drowning in data but starving for knowledge"

[Naisbitt, 1982]

- A lot of data is gathered, but never used
- It is easier to generate data than to analyze data

→ MACHINE LEARNING

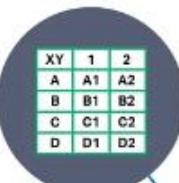
THE INTERNET IN **2023** EVERY MINUTE



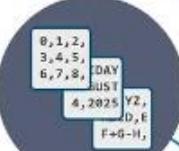
Created by: eDiscovery Today & LTMG

Structured Data vs Unstructured Data

Can be displayed in rows, columns and relational databases



Numbers, dates and strings



Estimated 20% of enterprise data (Gartner)



Requires less storage



Easier to manage and protect with legacy solutions



vs

Unstructured Data

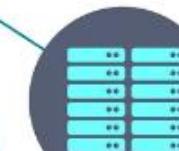
Cannot be displayed in rows, columns and relational databases



Images, audio, video, word processing files, e-mails, spreadsheets



Estimated 80% of enterprise data (Gartner)



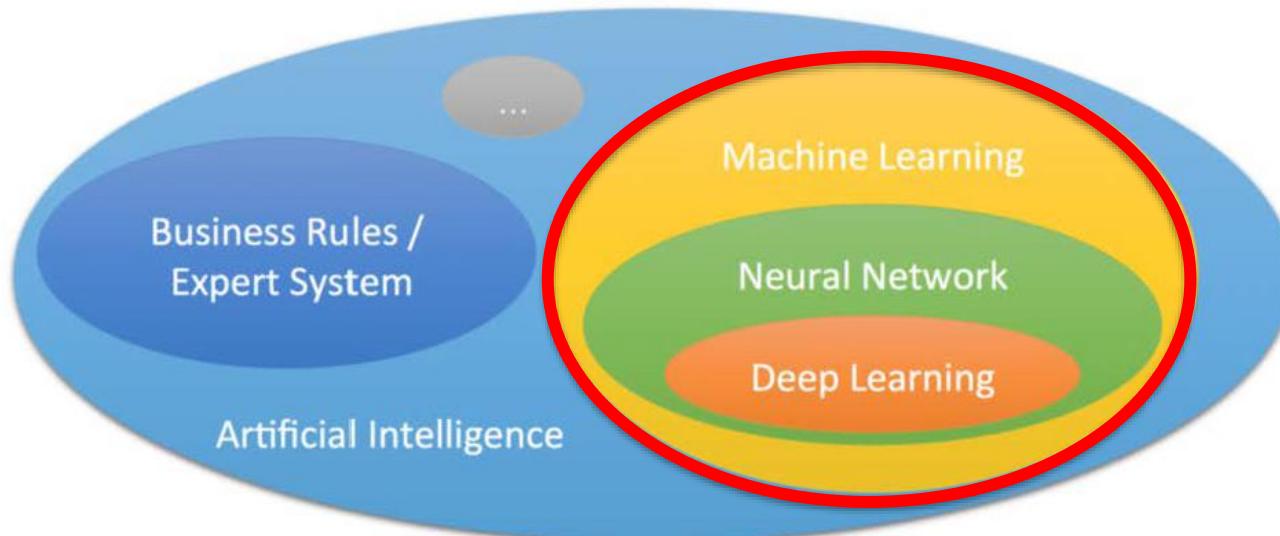
Requires more storage



More difficult to manage and protect with legacy solutions

Machine Learning & Artificial Intelligence

- **Artificial Intelligence (AI):**
“The set of all tasks in which a computer can make decisions.”
- **Machine Learning (ML):**
“The set of all tasks in which a computer can make decisions based on data.”
- **Deep Learning (DL):**
“The field of machine learning that uses certain objects called neural networks.”



Machine Learning

- Core domain of AI, concerned with automatic learning intelligence

noun

UK /ɪn'tel.i.dʒəns/ US /ɪn'tel.o.dʒəns/

intelligence *noun (ABILITY)*



B2 [U]

the ability to learn, understand, and make judgments or have opinions that are based on reason:

- *an intelligence test*
- *a child of high/average/low intelligence*
- *It's the intelligence of her writing that impresses me.*

- A computer is said to be able to learn if its performance in solving some task improves with its experience

Machine Learning

Example: buying a new car

- How do we make decisions?
 - by logical **reasoning**
 - by relying on previous **experiences** (either our own or those of others)
- For a computer: **experiences = data**

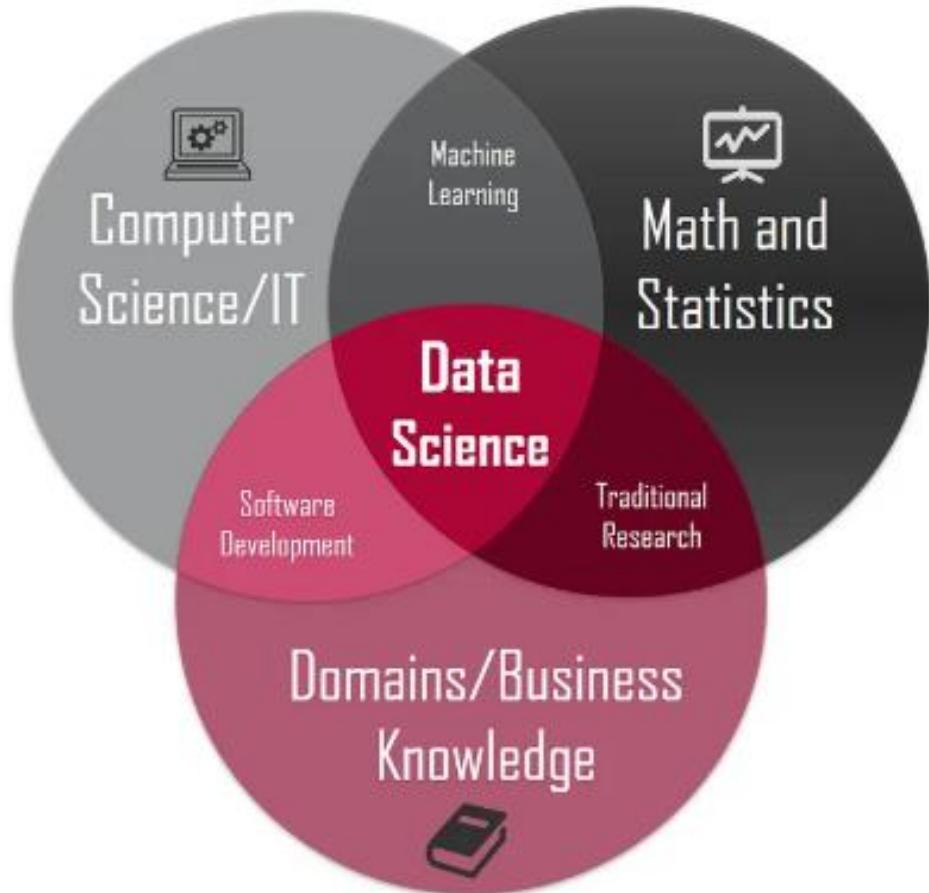


“Machine learning is common sense, except done by a computer”

Machine Learning ≠ Data Science

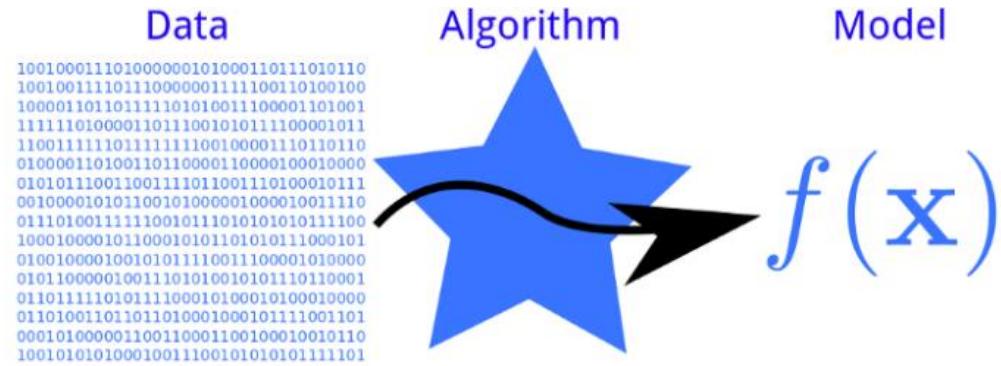
In practice:

- ML team: delivers software
- DS team: provides new insights



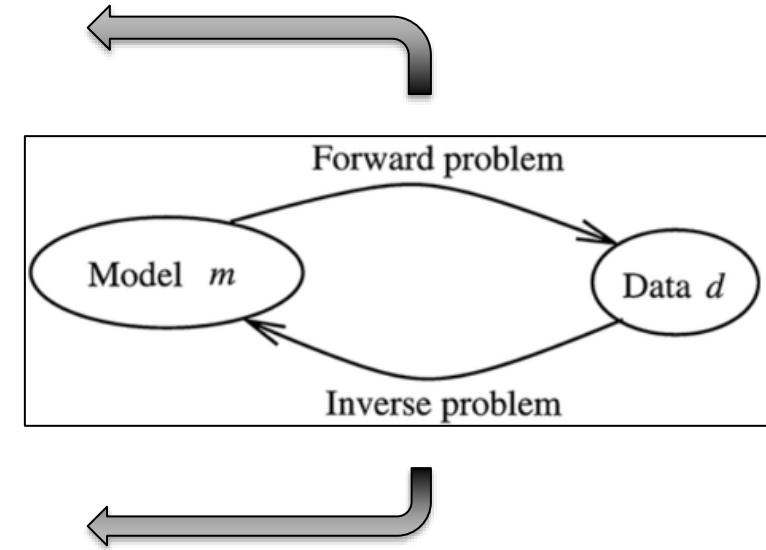
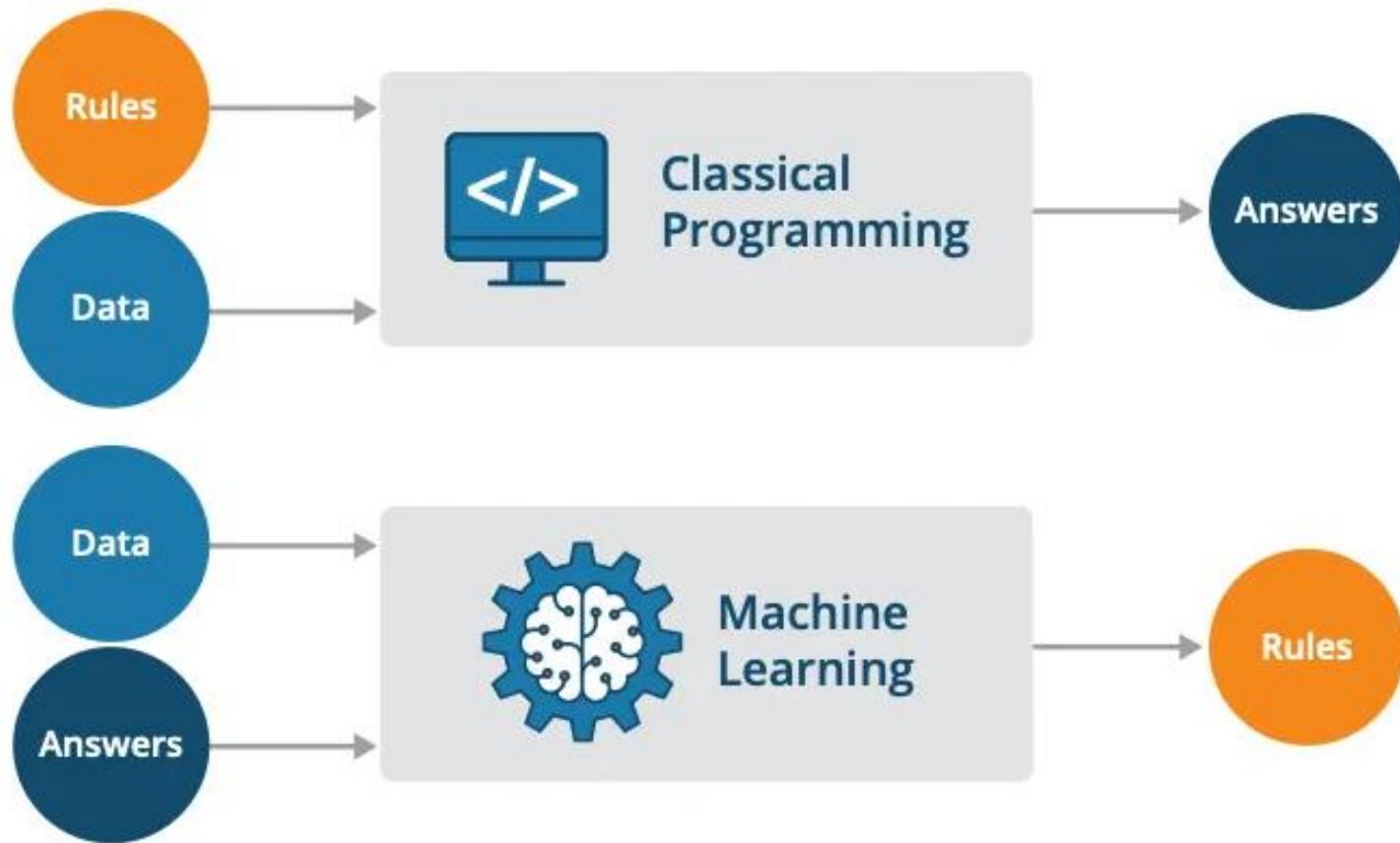
Algorithm vs Model

- **Model:** A set of rules that represent our data and can be used to make predictions.
- **Algorithm:** A procedure, or a set of steps, used to build a model.



“An algorithm is run on data to create a model”

Machine Learning vs Classical Programming

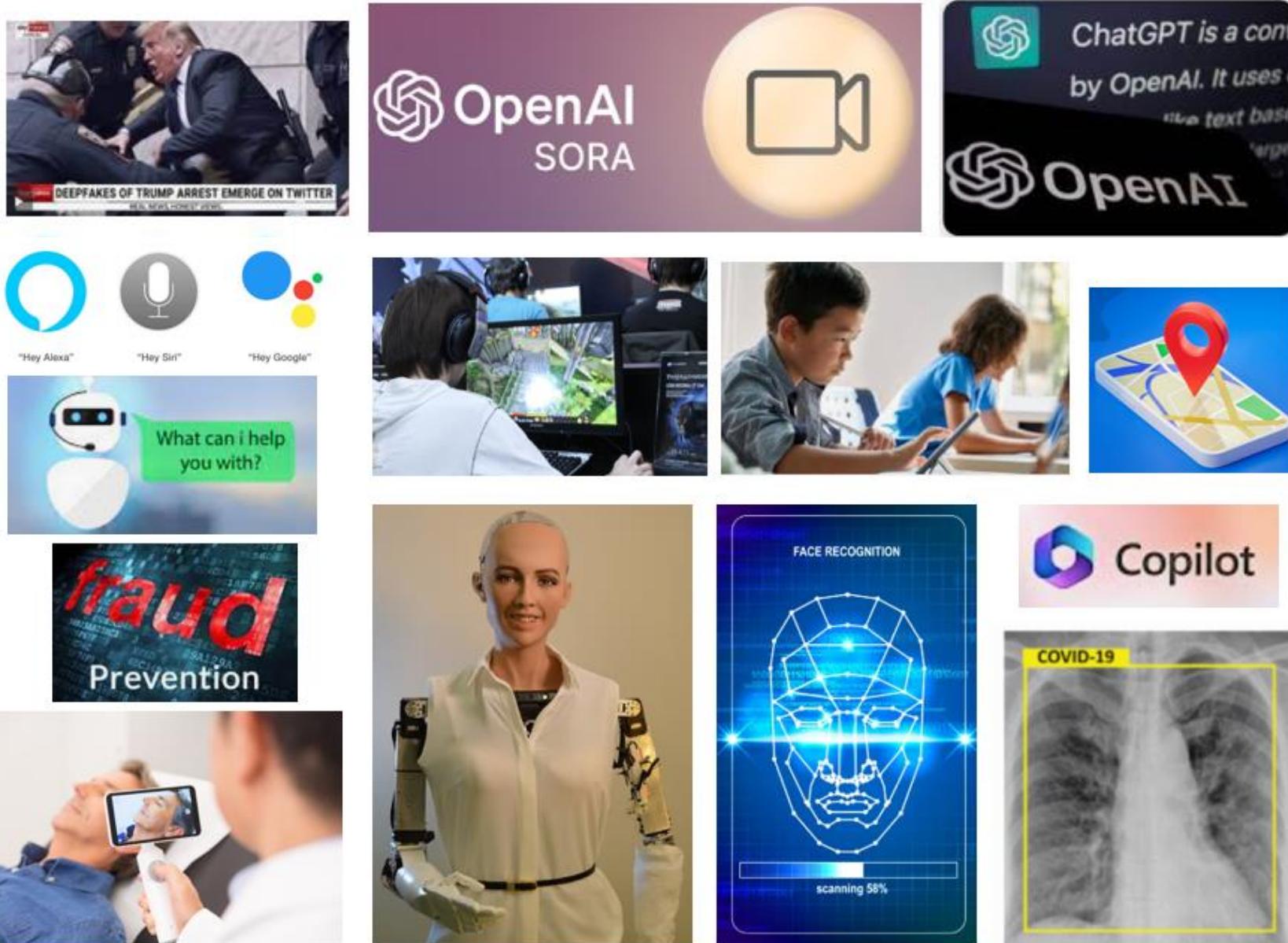


Introduction to Machine Learning

MACHINE LEARNING APPLICATIONS & TASKS

ML Applications

- Spam filters
- Recommender systems
- Personalized shopping
- Voice assistants
- Self-driving cars
- Search engines
- Chatbots
- Fraud prevention
- Face recognition
- Medical imaging
- Robotics
- Route planning
- Sales forecasting
- Deepfakes
- ...



Machine Learning Tasks

- Classification
- Regression
- Forecasting
- Prediction
- Anomaly detection
- Association rule mining
- Clustering
- ...

supervised learning

= A to B mapping

= Input to output mapping

= learning from (input, output) pairs

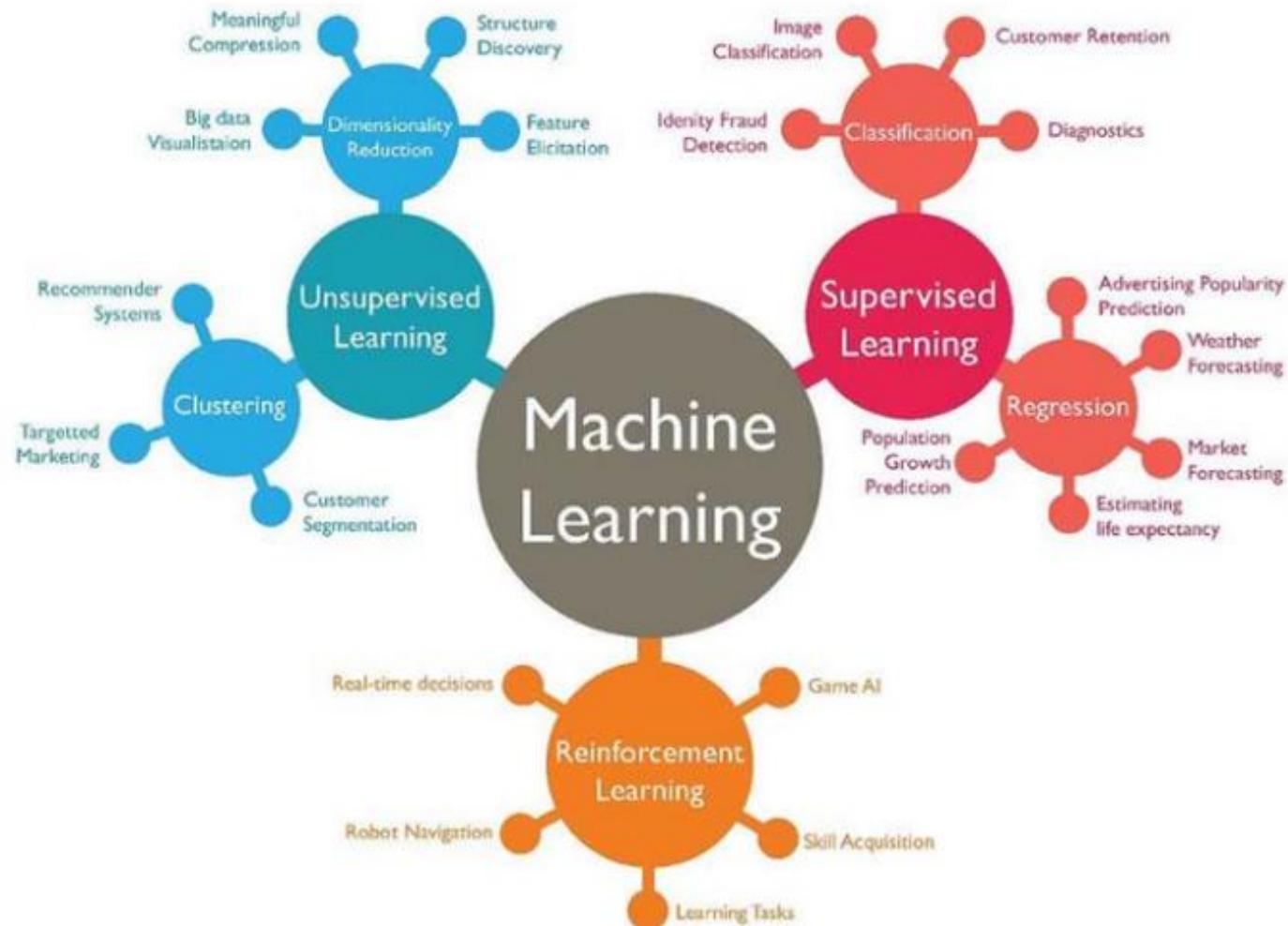
unsupervised learning

= learning from data without output

Supervised Learning

Input (A)	Output (B)	Application
email	spam? (0/1)	spam filtering
audio	text transcript	speech recognition
English	Chinese	machine translation
ad, user info	click? (0/1)	online advertising
image, radar info	position of other cars	self-driving car
image of phone	defect? (0/1)	visual inspection

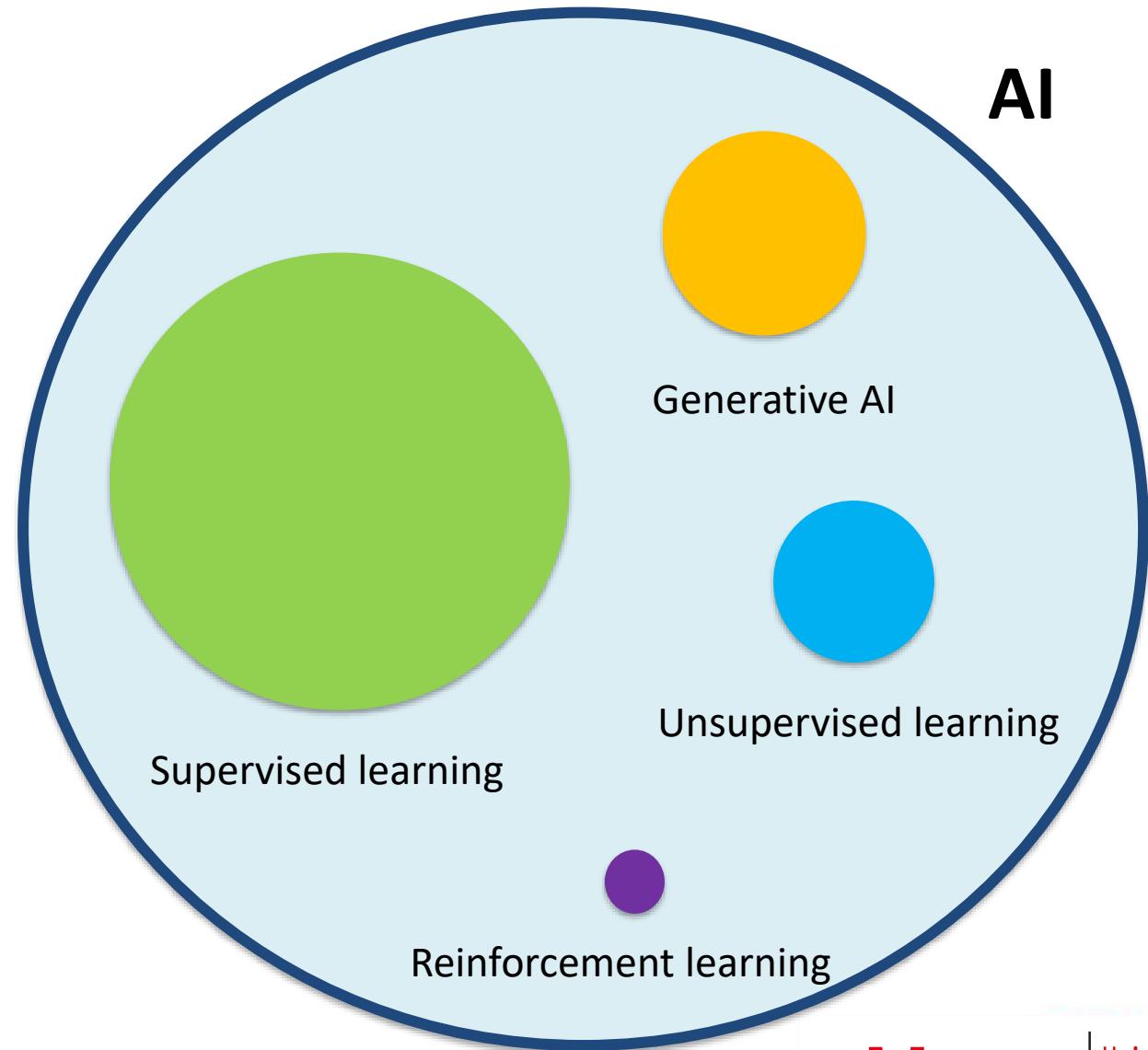
The Big Three



Category of machine learning. Image by <https://www.techleer.com/articles/203-machine-learning-algorithm-backbone-of-emerging-technologies/>

What about GenAI?

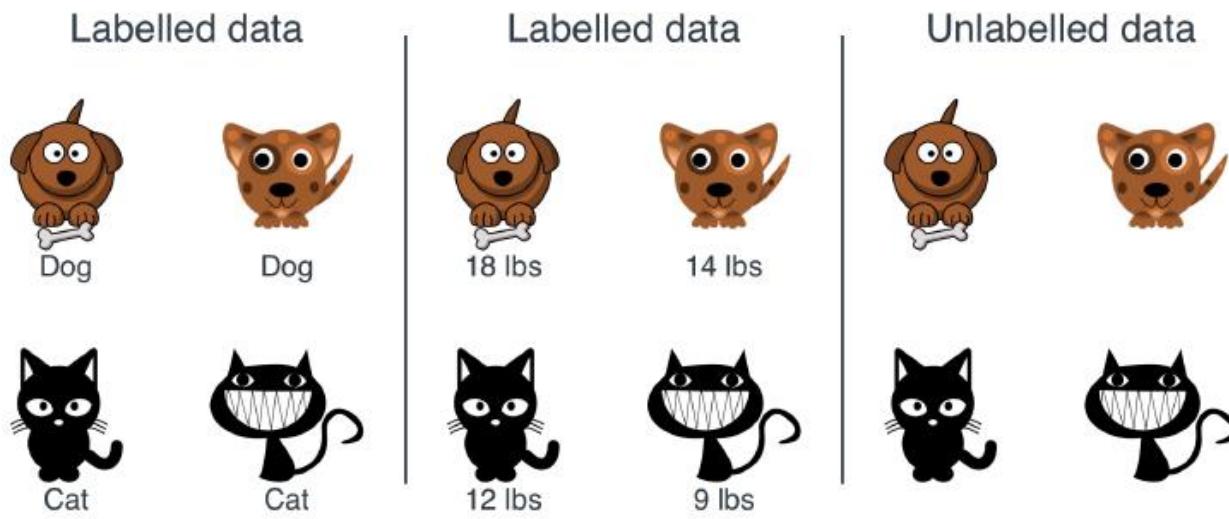
- **Supervised learning**
= learning from labeled data
- **Unsupervised learning**
= learning from unlabeled data
- **Reinforcement learning**
= learning from rewards
- **Generative AI**
= generating new data



<https://youtu.be/5p248yoa3oE?si=M6TY-fAtB2I1S7iF>

Supervised vs Unsupervised

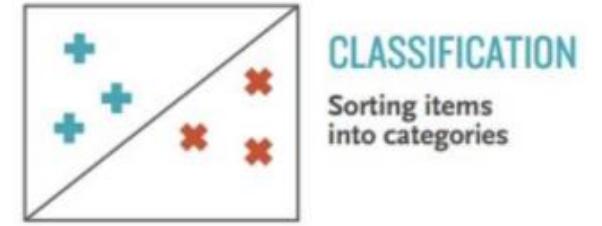
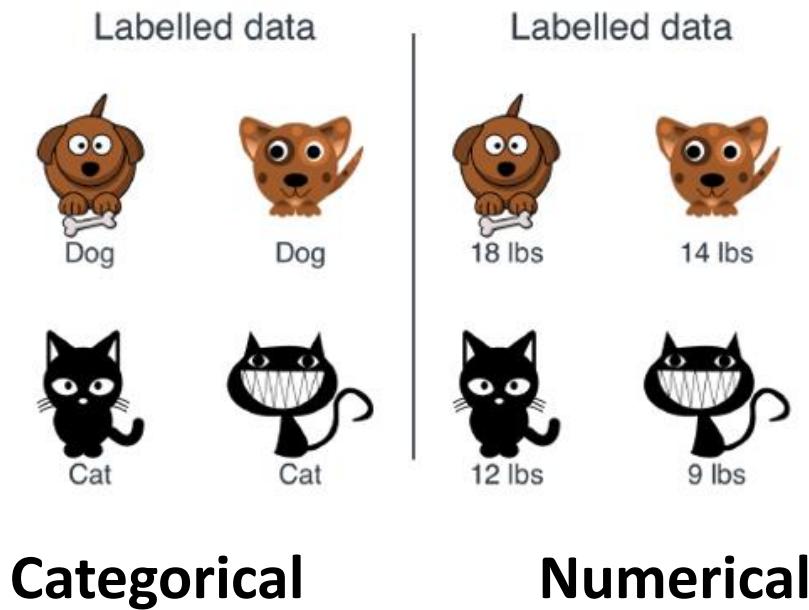
- **Labeled data:** data with label
→ **SUPERVISED LEARNING**
- **Unlabeled data:** data without label
→ **UNSUPERVISED LEARNING**



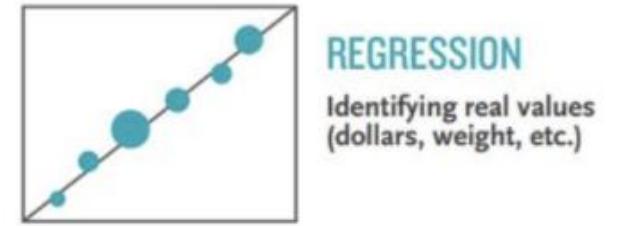
Classification vs Regression

= Supervised Learning

- **Categorical target → Classification**
- **Numerical target → Regression**



CLASSIFICATION
Sorting items
into categories



REGRESSION
Identifying real values
(dollars, weight, etc.)

Structured Data

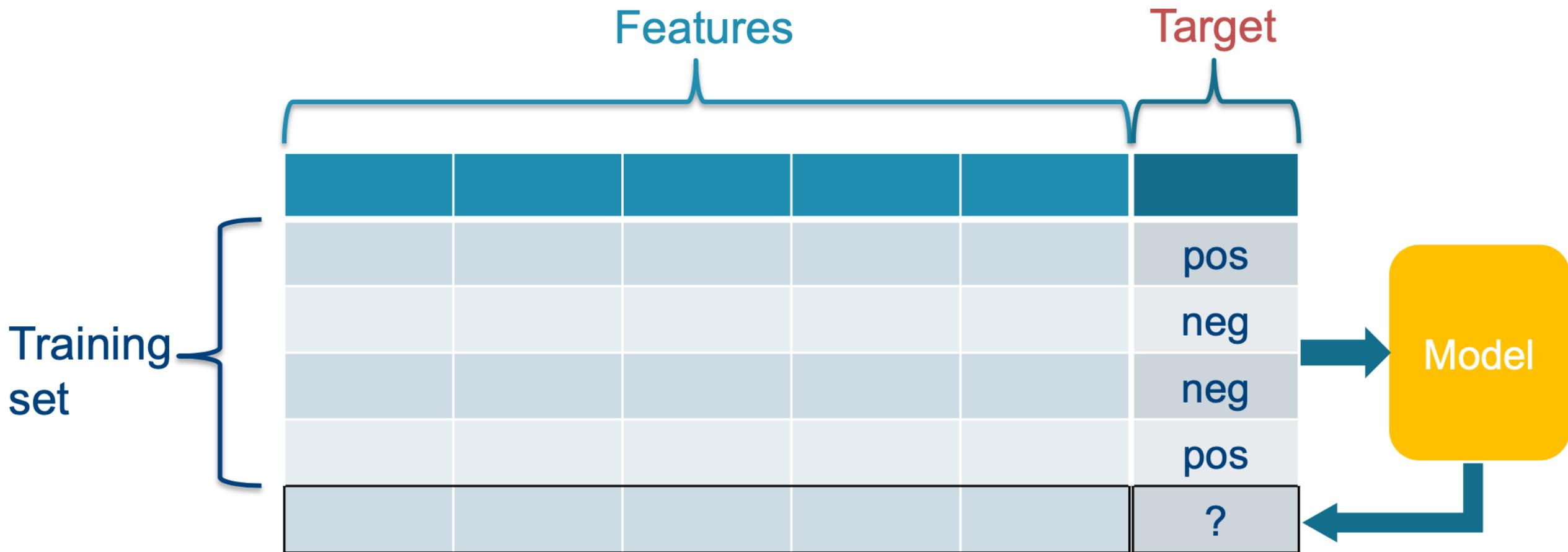
- **Data** = information (= table)
 - **Example** = sample = instance = data point (= table row/record)
 - **Feature** = independent variable (= table column/attribute)
 - **Target** = labels = dependent variable = feature we want to predict

Introduction to Machine Learning

SUPERVISED LEARNING

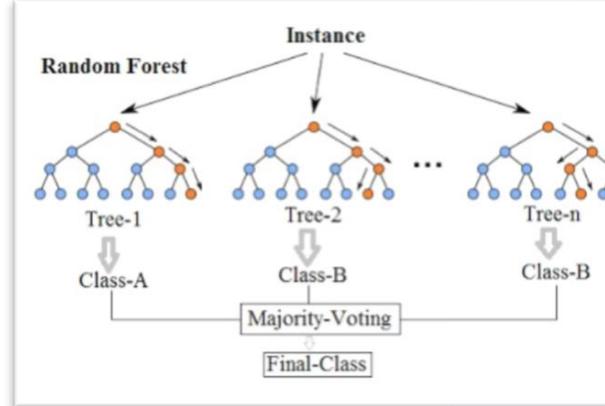
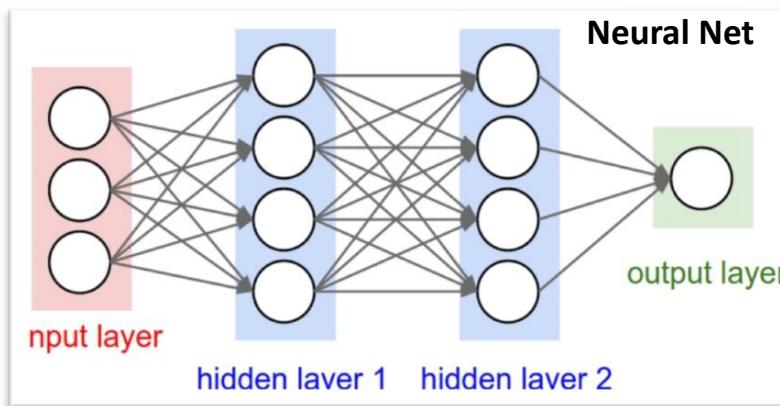
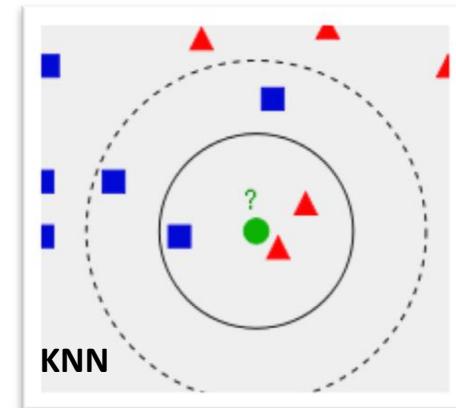
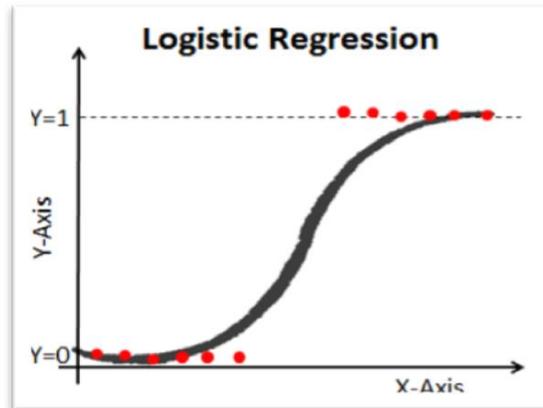
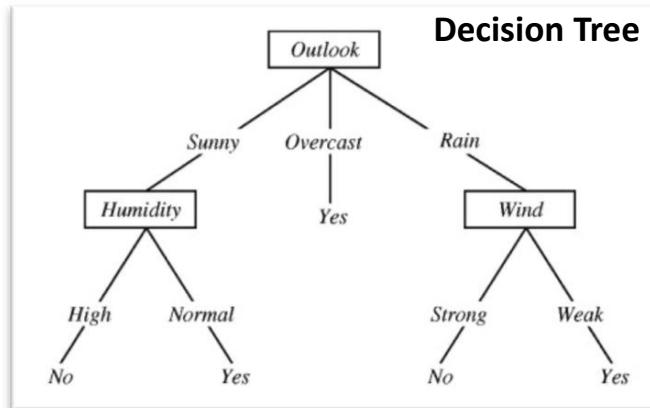
Supervised Learning

- Task: learn a model to predict a target for new data instances, based on a training set of data instances for which the target is known



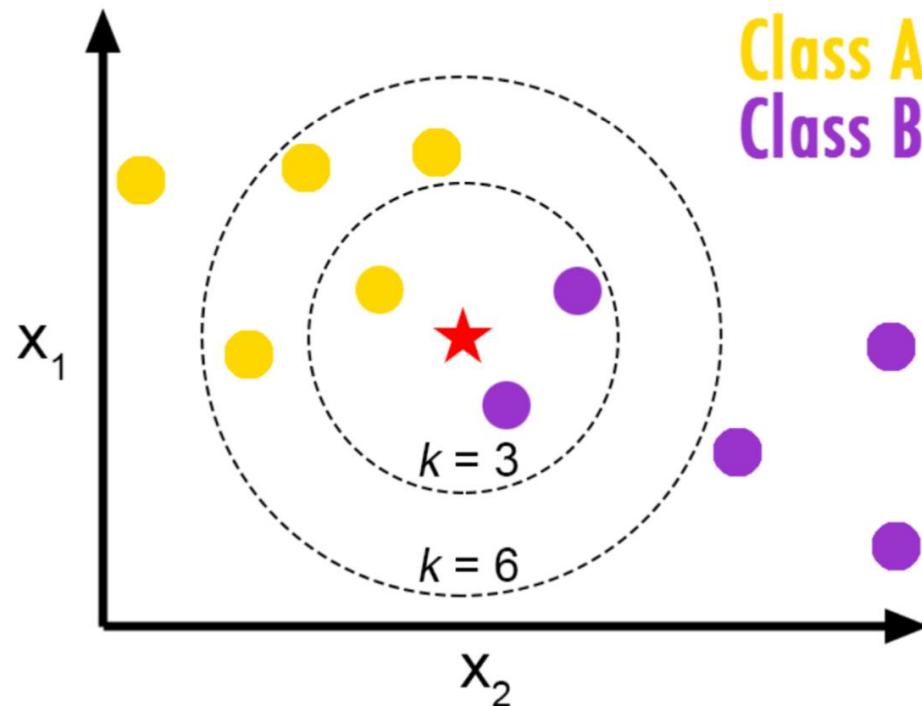
Supervised Learning Algorithms

- There exist plenty of supervised learning algorithms
- **No free lunch:** there is no algorithm that works best for every problem



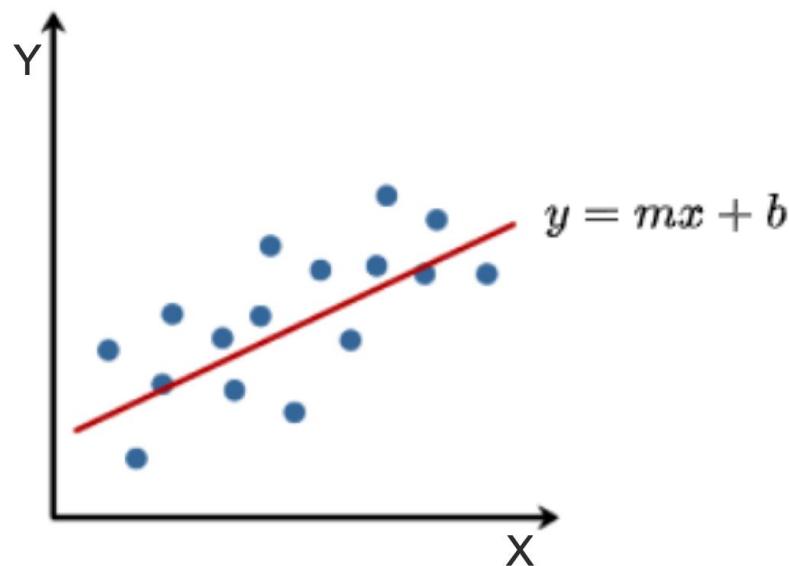
K Nearest Neighbors (KNN)

- Classification (regression is also possible)
- Requires no training (= lazy learning, as opposed to eager learning)
- Main task: find suitable distance function (Euclidean, Manhattan, ...)



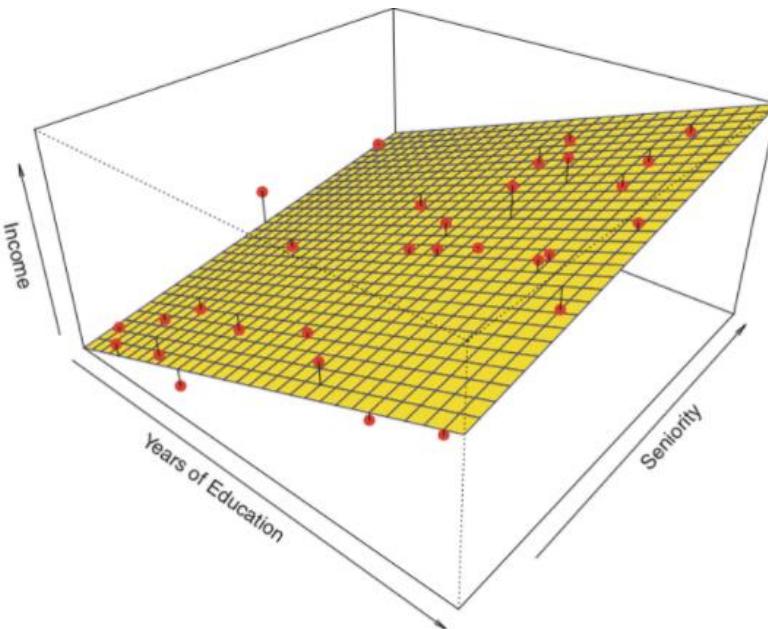
Simple Linear Regression

- Regression for numeric targets
- 1 independent variable (feature x) and 1 dependent variable (target y)
- Main task: estimate parameters m and b , such that predictions (red line) and targets (blue dots) are as close as possible (= best-fitting straight line)

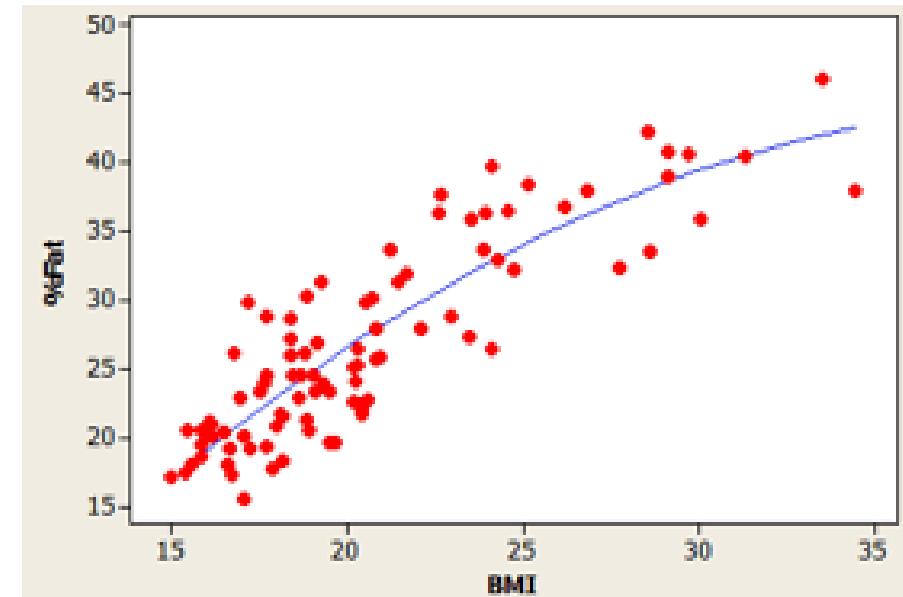


Linear & Nonlinear Regression

- Linear Regression 2 features and 1 target (left)
- Nonlinear regression 1 feature and 1 target (right)



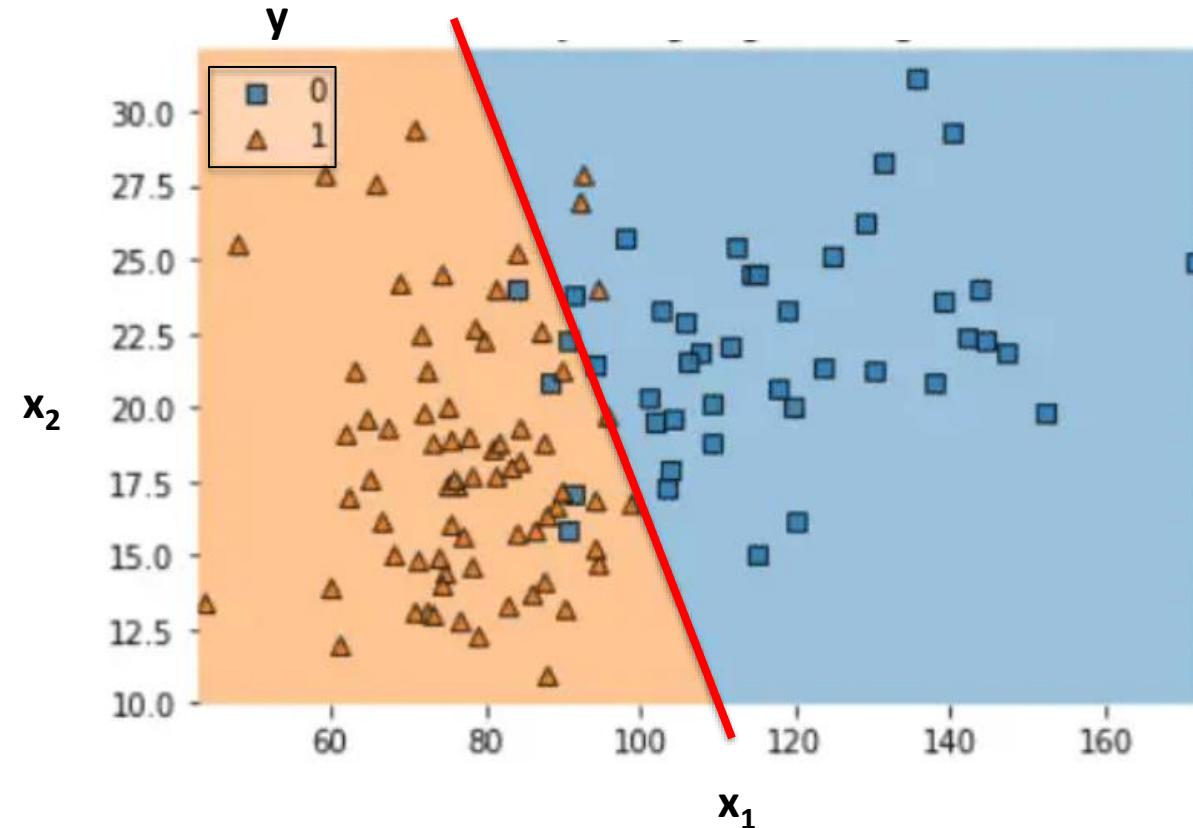
Source: James et al. *Introduction to Statistical Learning* (Springer 2013)



Source: the minitab blog

Logistic Regression

- Regression for binary targets
- Features x_i and target y
- Main task: find a **separating straight line**
= **binary classification**
- N dimensions: separating hyperplane

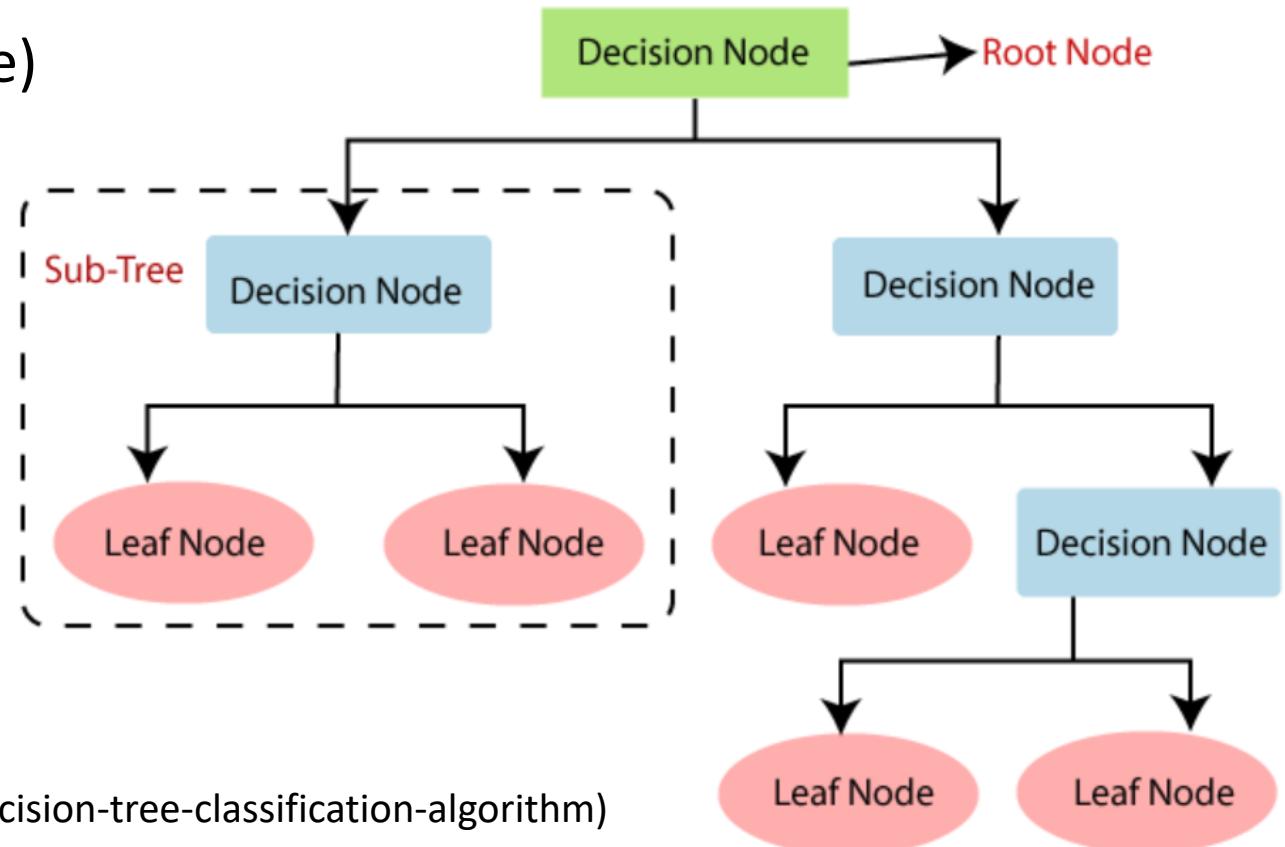


source: <https://www.jcchouinard.com/logistic-regression/>

Decision Trees

- Very popular data mining methods
- High interpretability/explainability (e.g. medicine)
- Efficient learning and prediction procedures
- Classification (regression is also possible)

- Leaf nodes versus internal nodes
 - = labels
 - = features

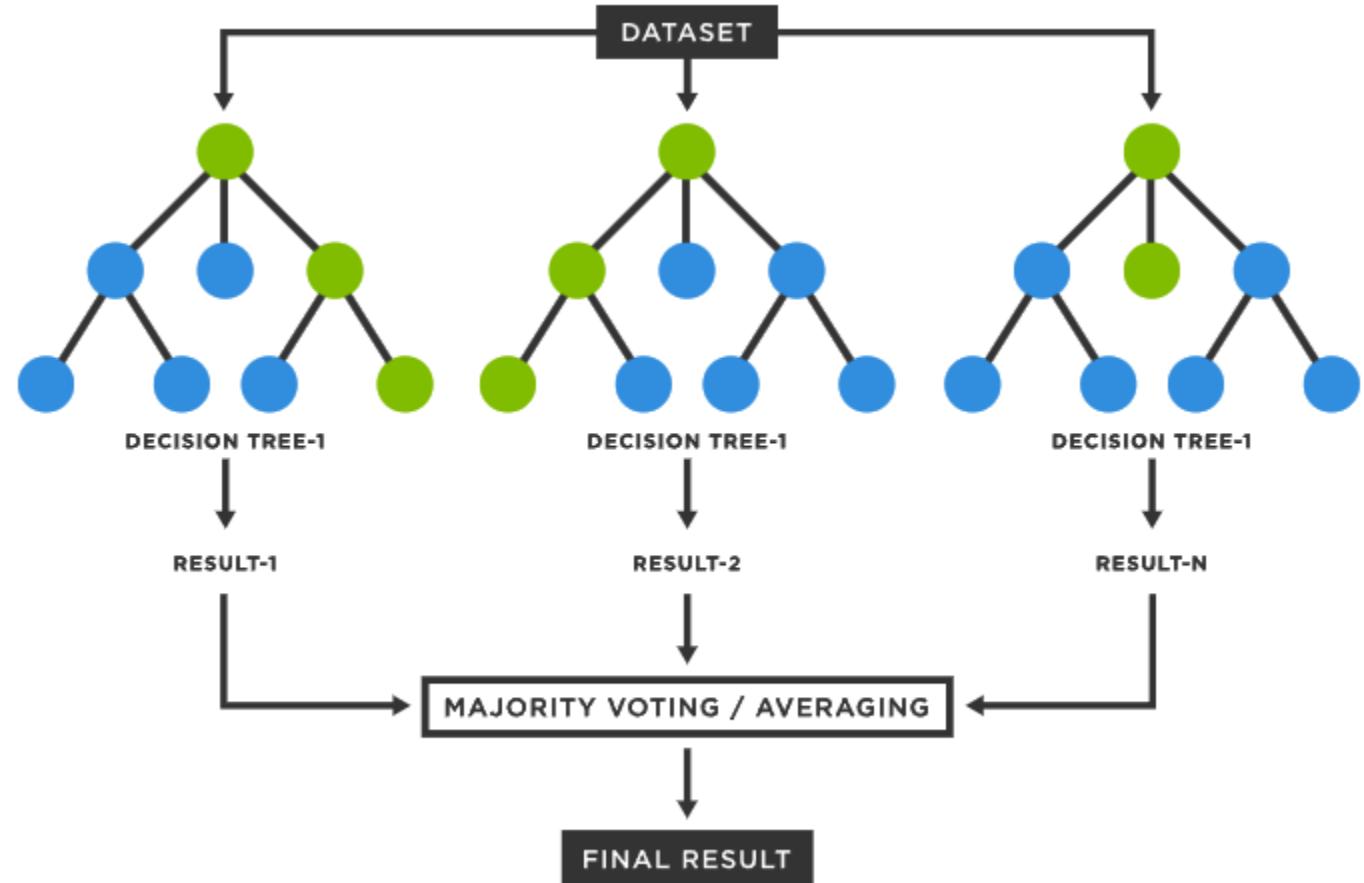


(source: <https://www.javatpoint.com/machine-learning-decision-tree-classification-algorithm>)

Random Forests

= Ensemble of decision trees

- Each tree is trained on a random subset (= **bagging**)
- Parallelization possible
- Voting mechanism:
 - Classification: majority voting
 - Regression: averaging

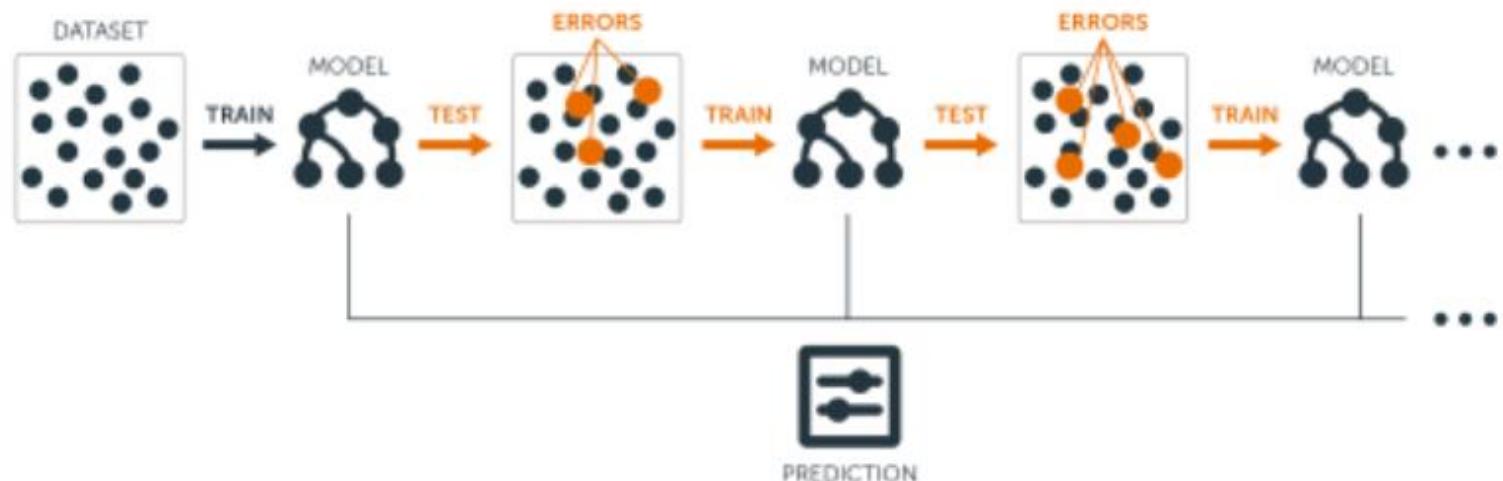


<https://www.spotfire.com/glossary/what-is-a-random-forest>

Boosting

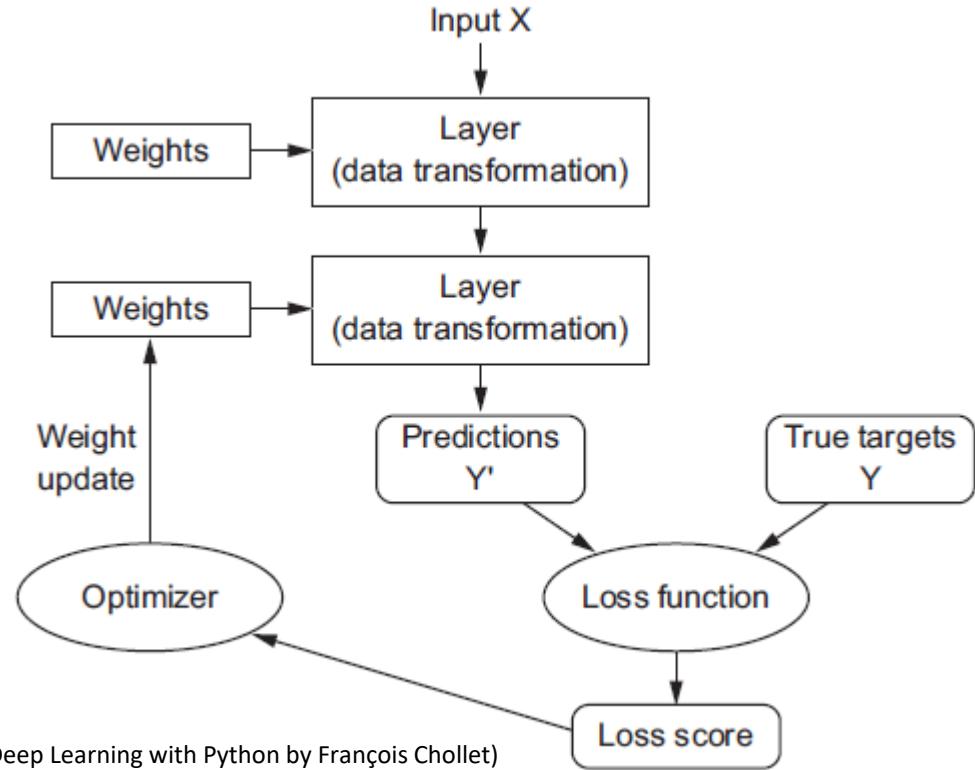
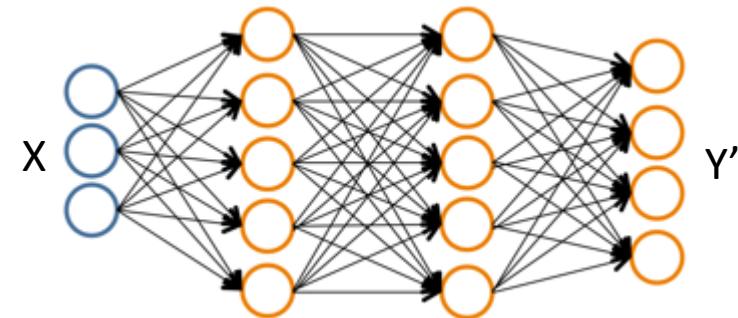
= Ensemble of weak learners, usually decision trees

- Sequentially trains weak models
- where each model corrects errors of the previous one
- to create a strong learner.
- Examples:
 - AdaBoost
 - LightGBM
 - XGBoost
 - CatBoost



Artificial Neural Network

- Regression or classification
- Features X and targets Y
- **Loss:** function quantifying differences between targets Y and predictions Y'
- Main task: find optimal weights that minimize the loss



(Source: Deep Learning with Python by François Chollet)

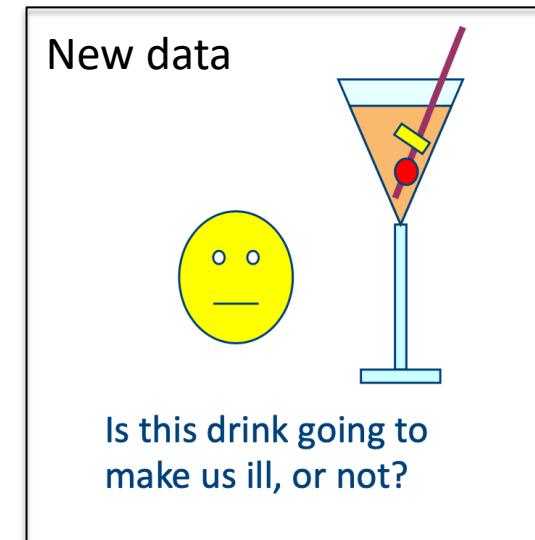
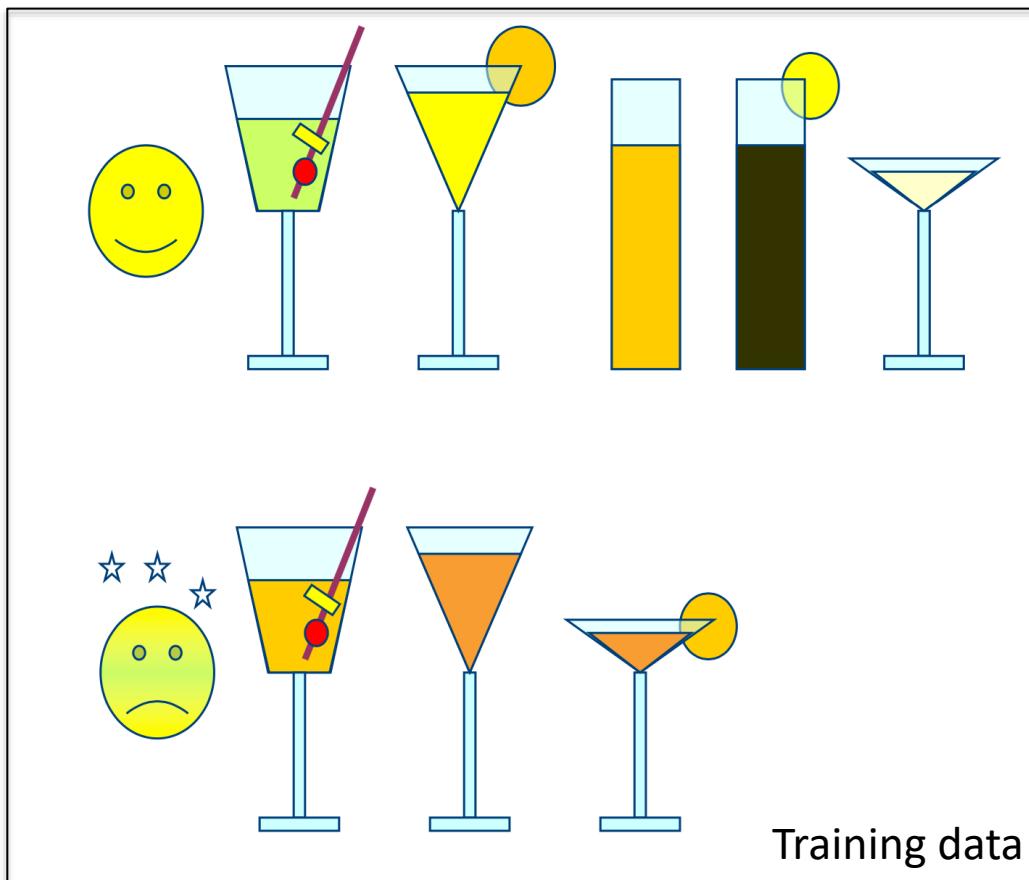
Intermezzo

Cocktail example



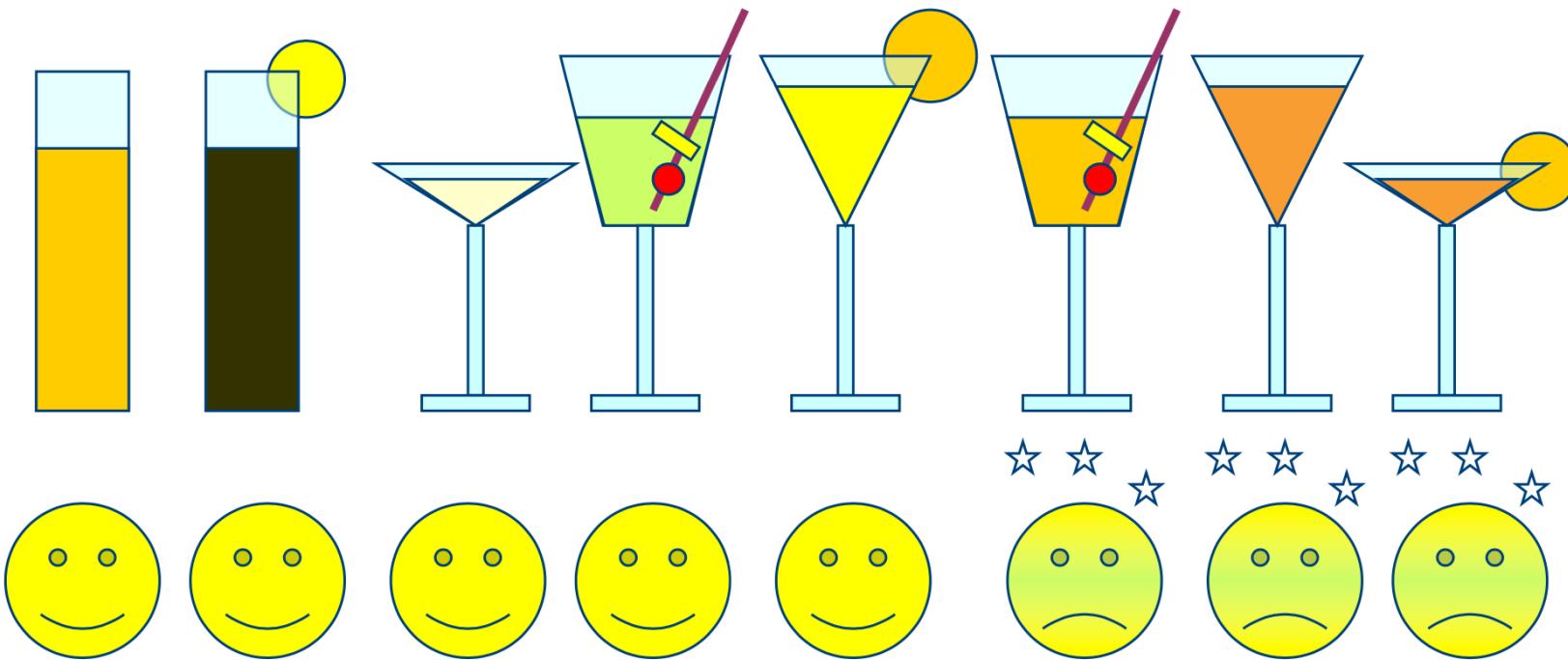
Decision Tree

- Inducing decision trees: cocktail example (by Hendrik Blockeel)



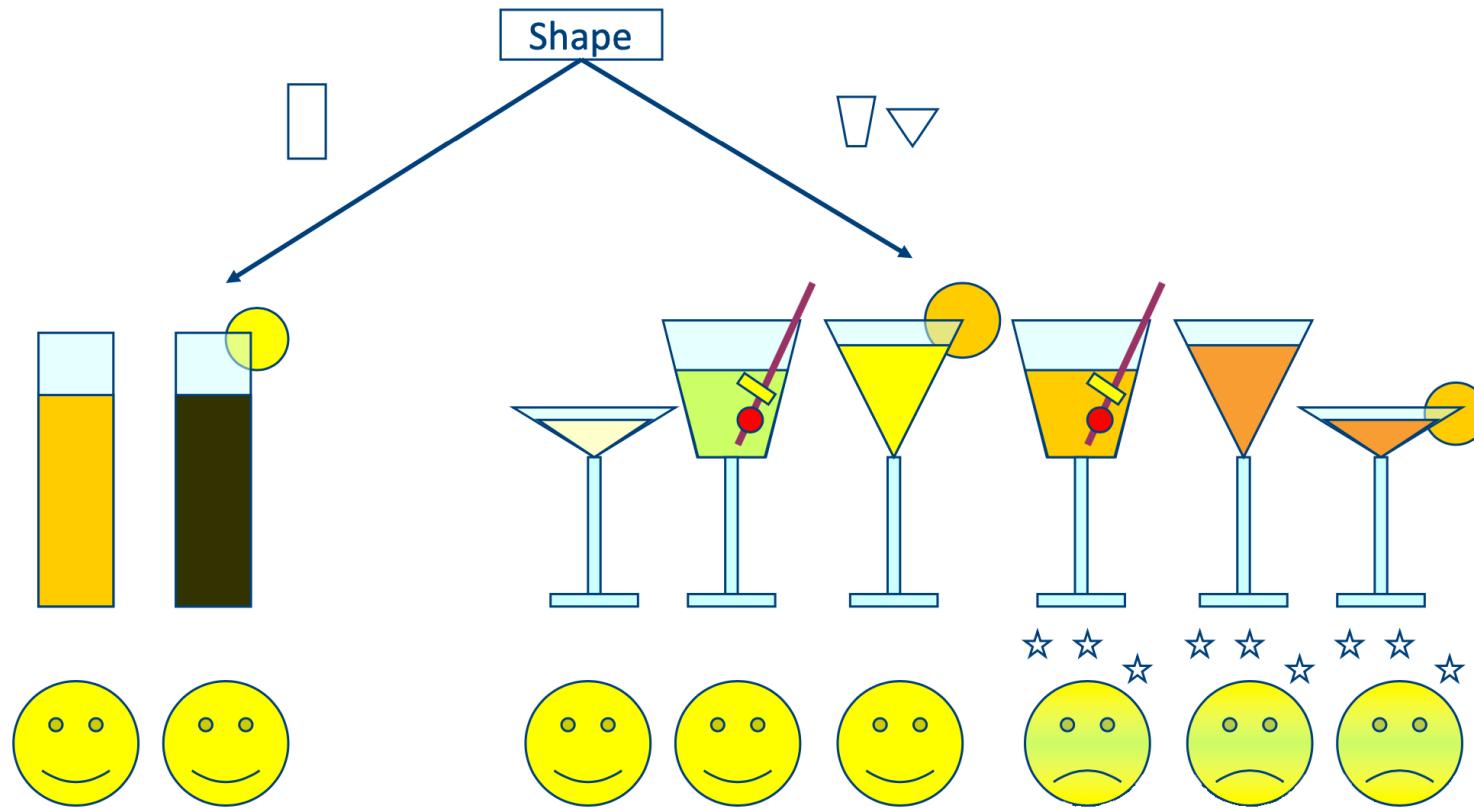
Decision Tree

- The training set:



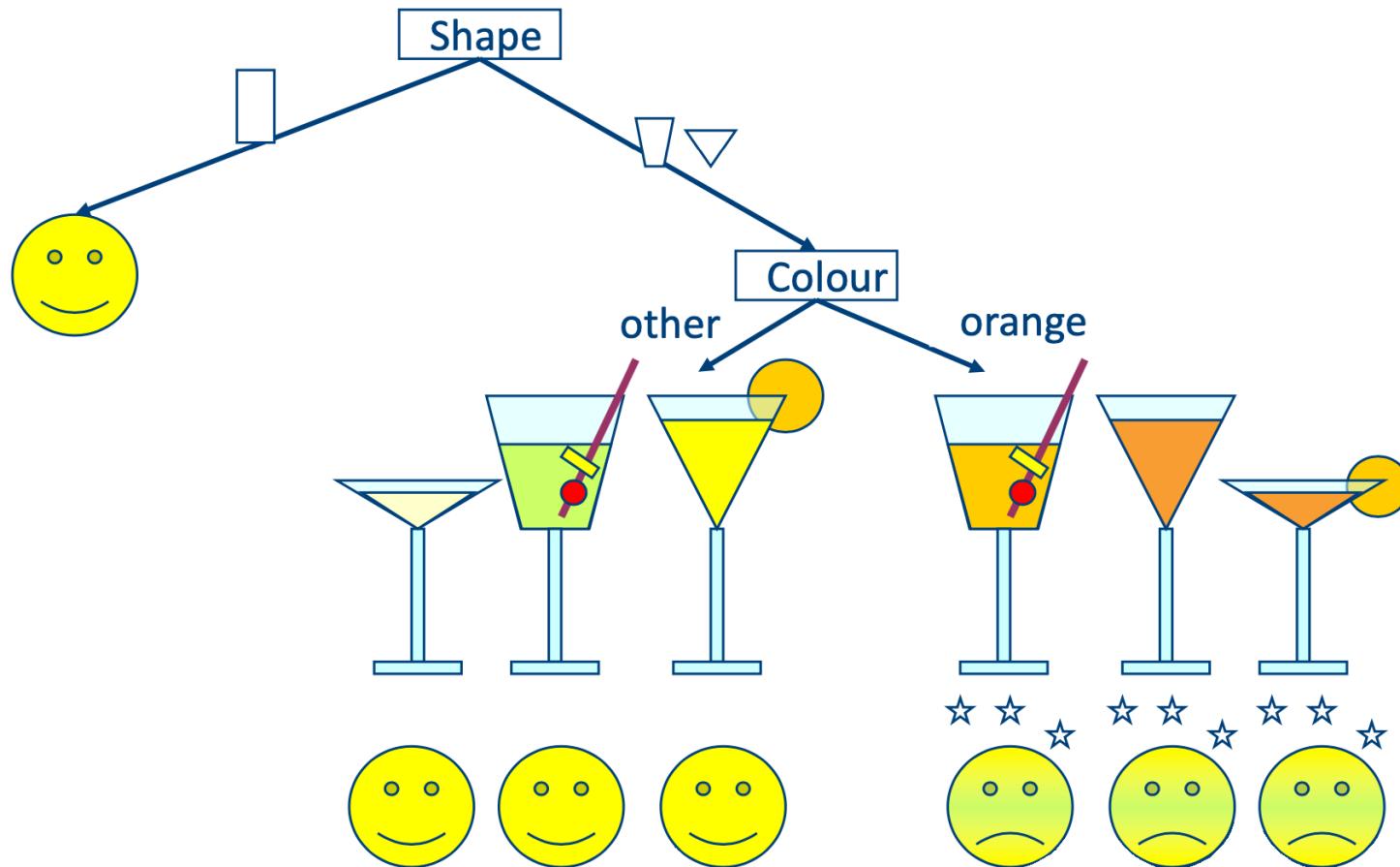
Decision Tree

- First split: rectangular or other shape?



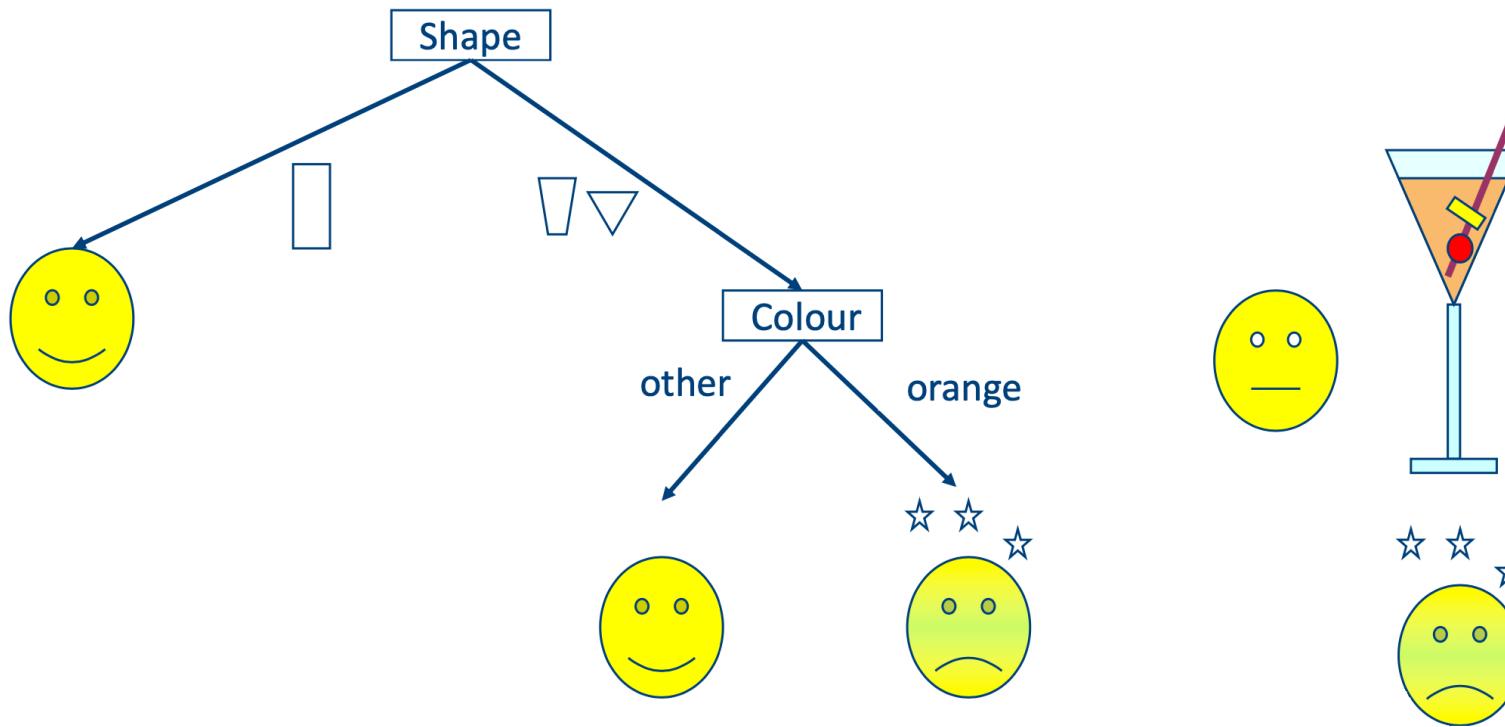
Decision Tree

- Second split if shape is not rectangular: orange or other color?



Decision Tree

- Applying the decision tree to the new cocktail:
no rectangular shape and orange color, so it will make us sick

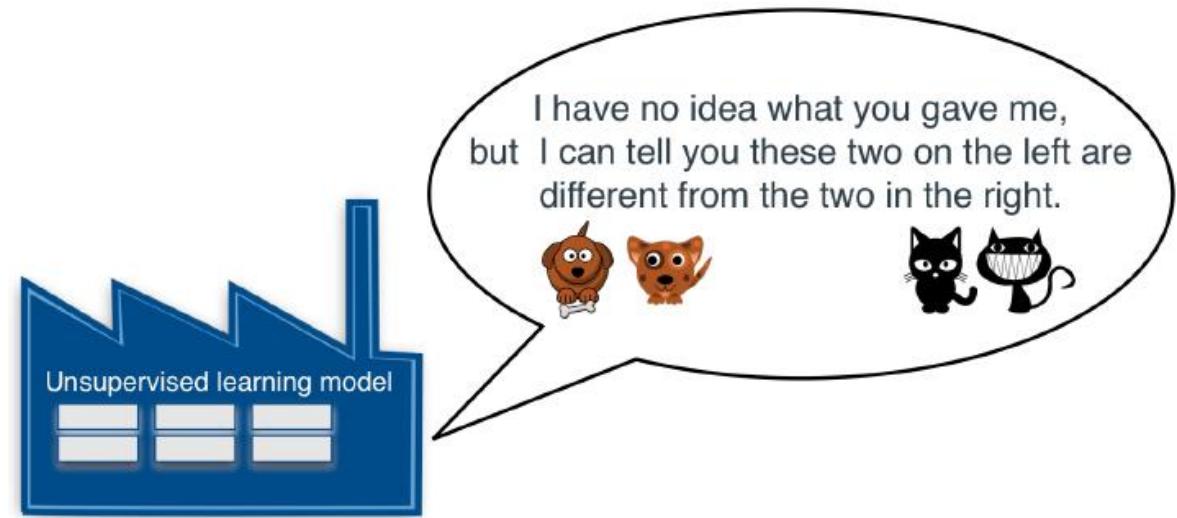


Introduction to Machine Learning

UNSUPERVISED LEARNING

Unsupervised Learning

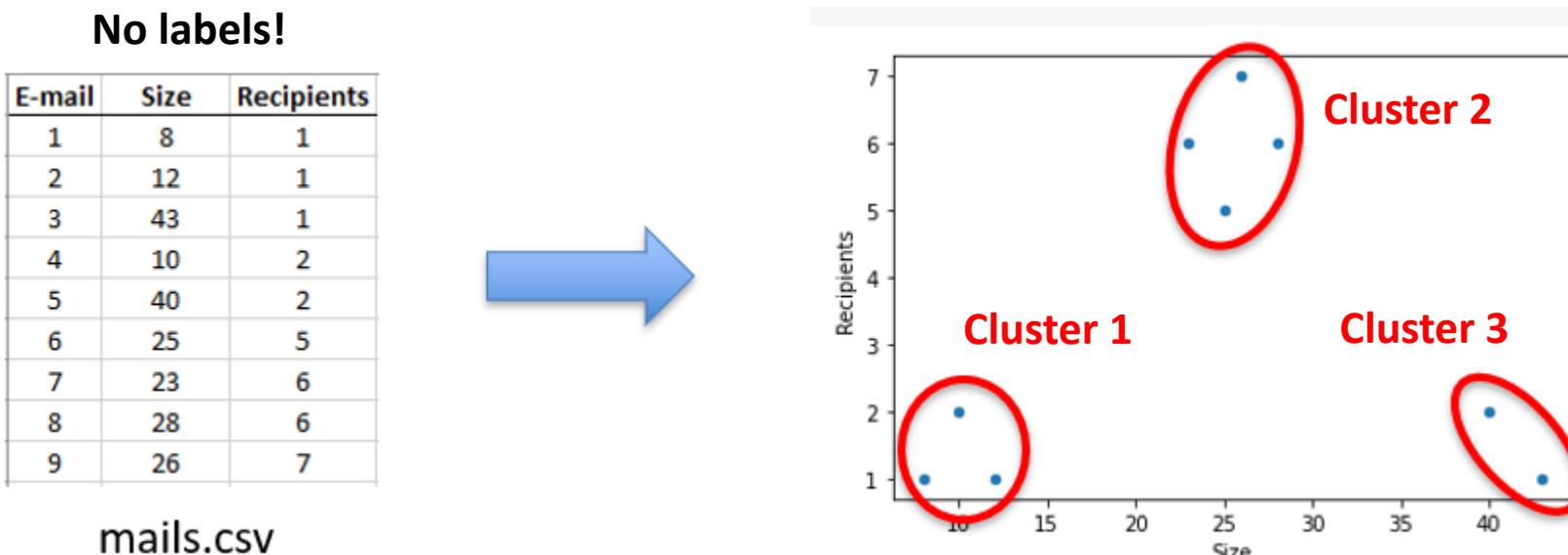
- Data are **not labeled**
- Often used during data **preprocessing**
- Clustering: grouping data based on similarities
- Dimensionality reduction: reducing the number of features while retaining as much meaningful information as possible
- Matrix factorization: decomposing the data in order to discover latent features



Clustering

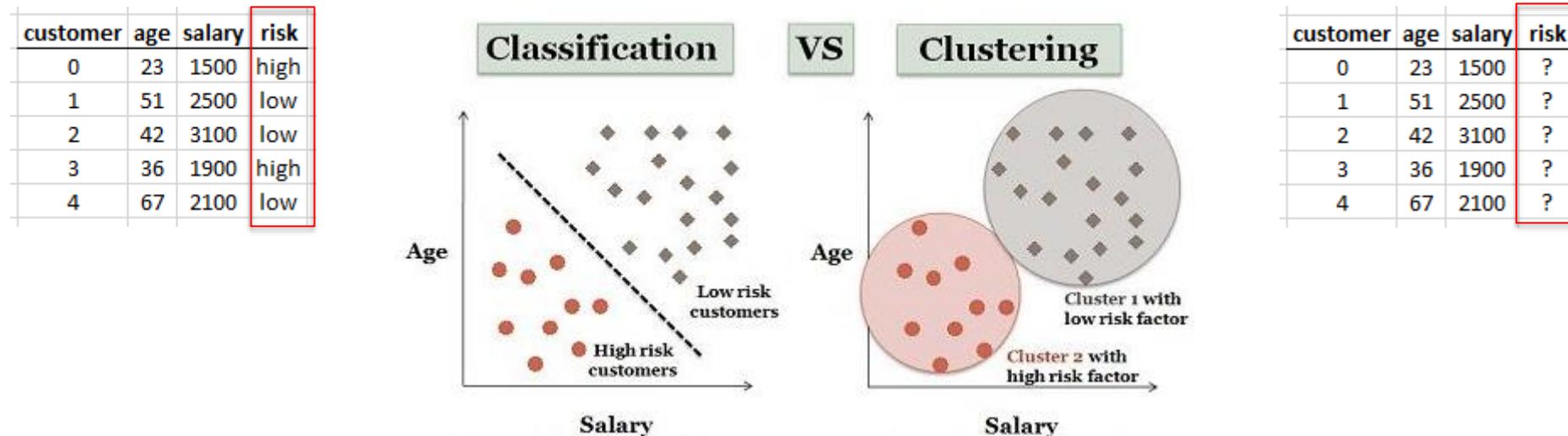
Applications:

- Genetics: grouping species based on similarities
- Medical imaging: partitioning images based on tissue structures
- Market segmentation: clustering customers based on demographics, income, etc.
- Mails:



Clustering vs Classification

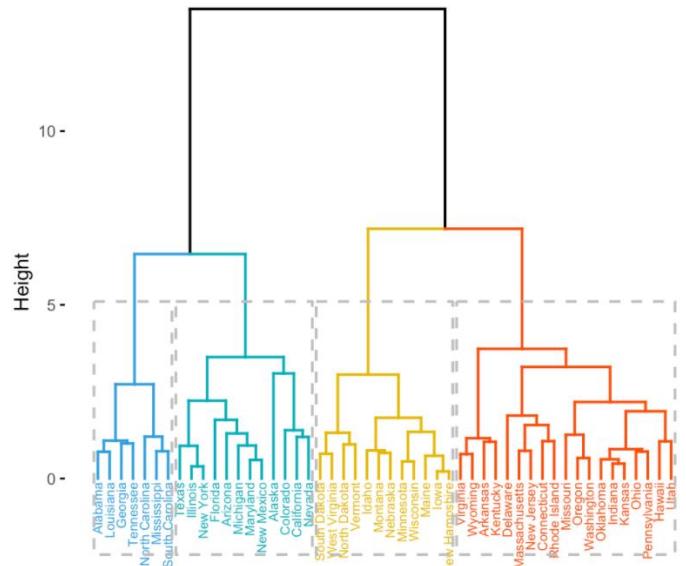
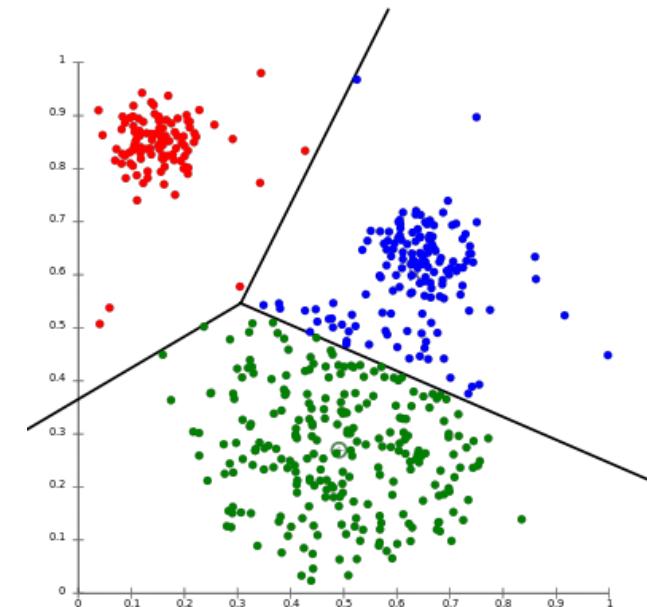
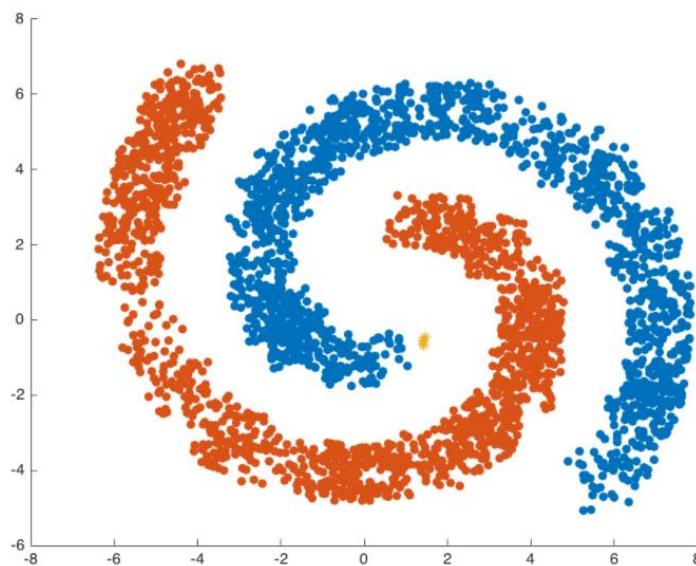
- **Classification:** labeled data → classes already exist
- **Clustering:** unlabeled data → classes don't exist yet



(source: <https://techdifferences.com/difference-between-classification-and-clustering.html>)

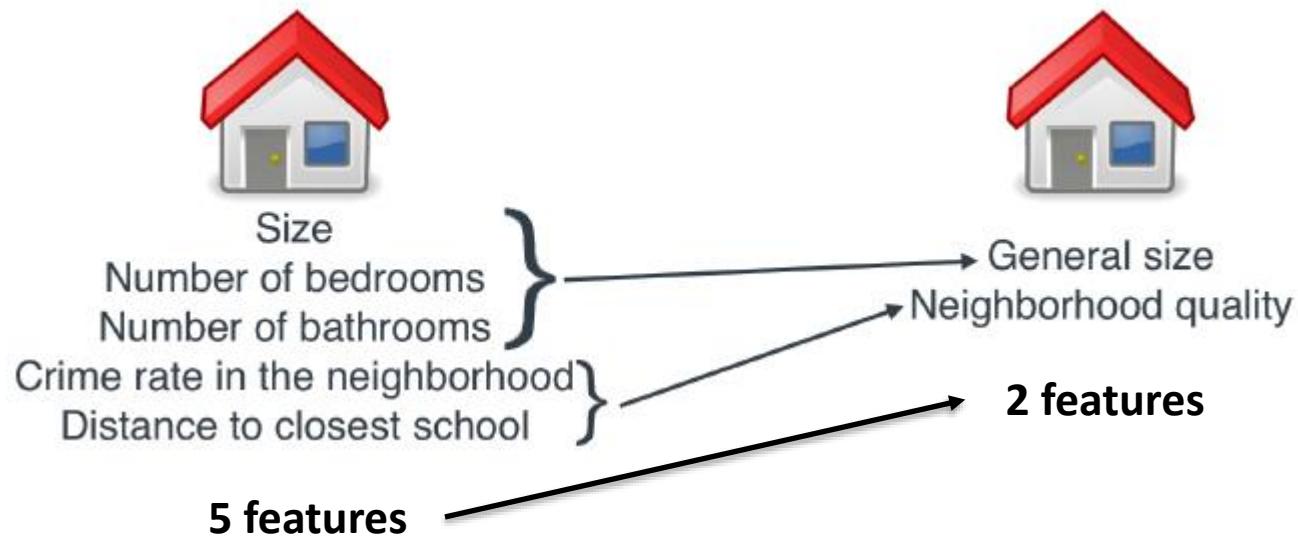
Clustering Algorithms

- K-means clustering
<https://youtu.be/nXY6PxAaOk0>
- Hierarchical clustering (dendrogram)
- Gaussian mixture models
- DBSCAN
- ...

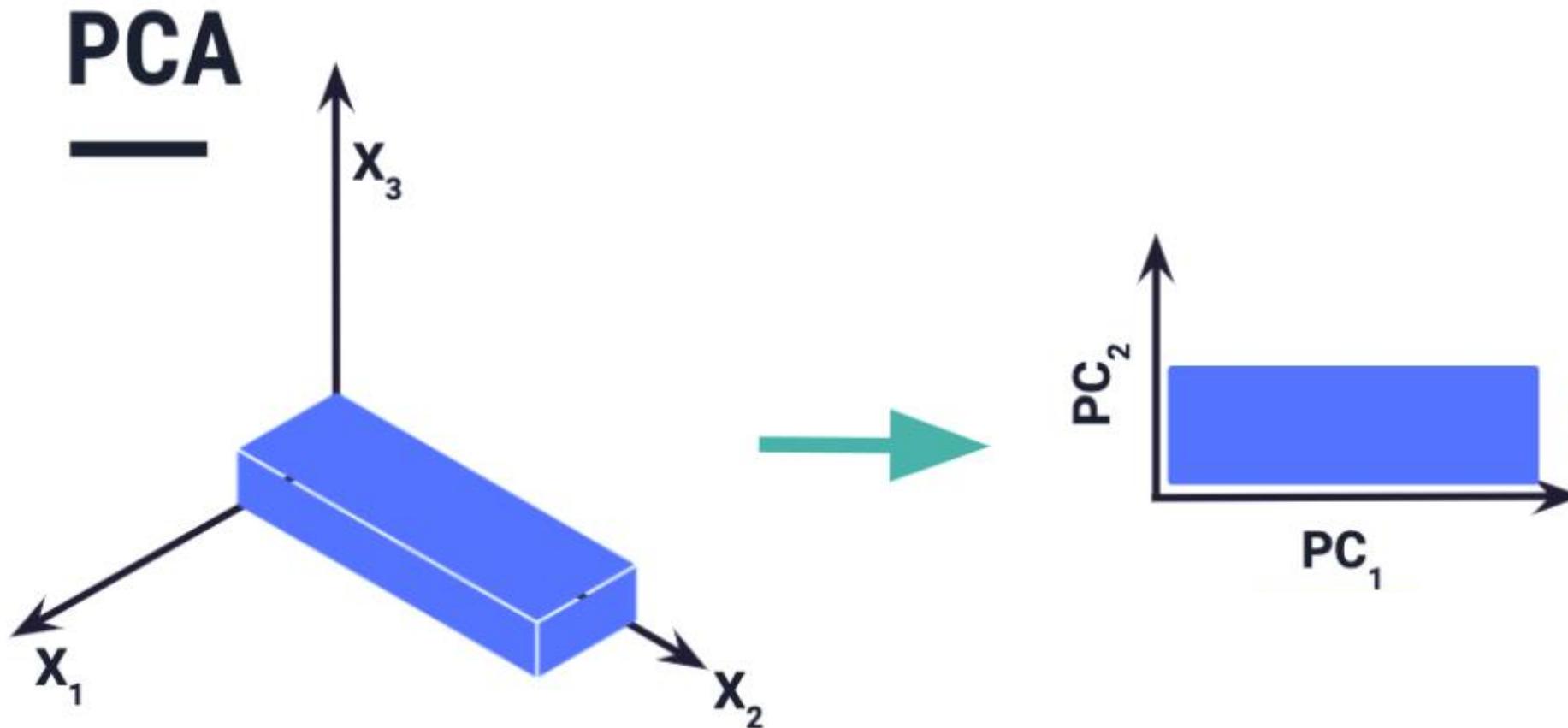


Dimensionality Reduction

- Number of dimensions = number of features
- Reducing the number of features



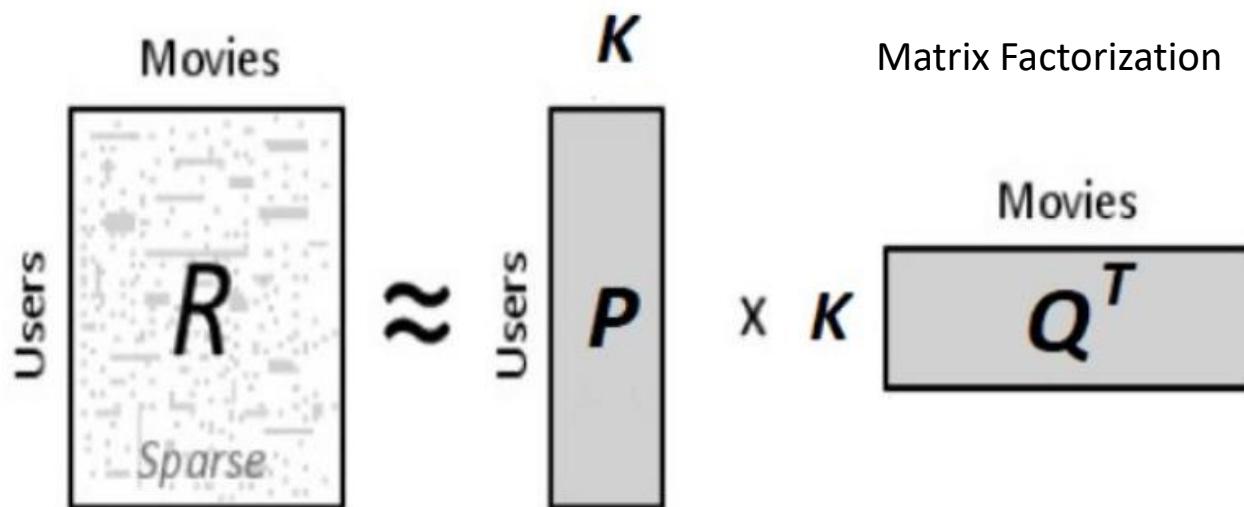
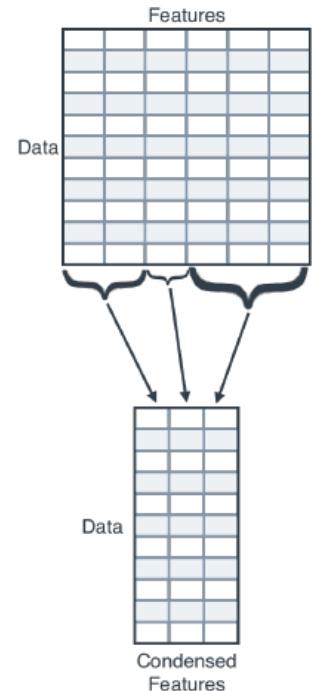
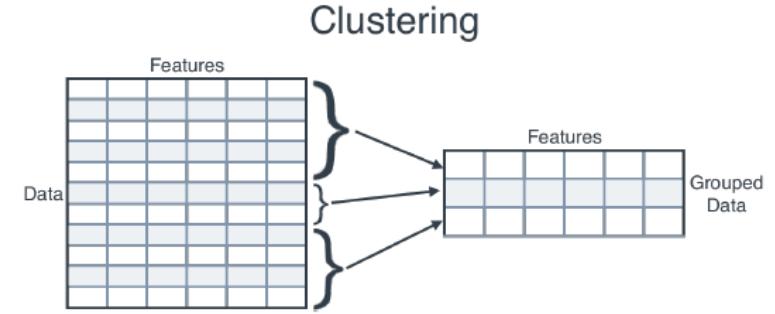
Example: Principal Component Analysis



(source: <https://knowledge.dataiku.com/latest/ml-analytics/statistics/concept-principal-component-analysis-pca.html>)

Matrix Factorization

- **Clustering:** reducing samples (= rows)
- **Dimensionality Reduction:** reducing features (= columns)
- **Matrix Factorization:** reducing both rows and columns



(source: <https://www.kaggle.com/code/residentmario/notes-on-matrix-factorization-machines>)

Example: Recommender Systems



<https://youtu.be/ZspR5PZemcs>

Matrix Factorization

	M1	M2	M3	M4	M5
Comedy	3	1	1	3	1
Action	1	2	4	1	3

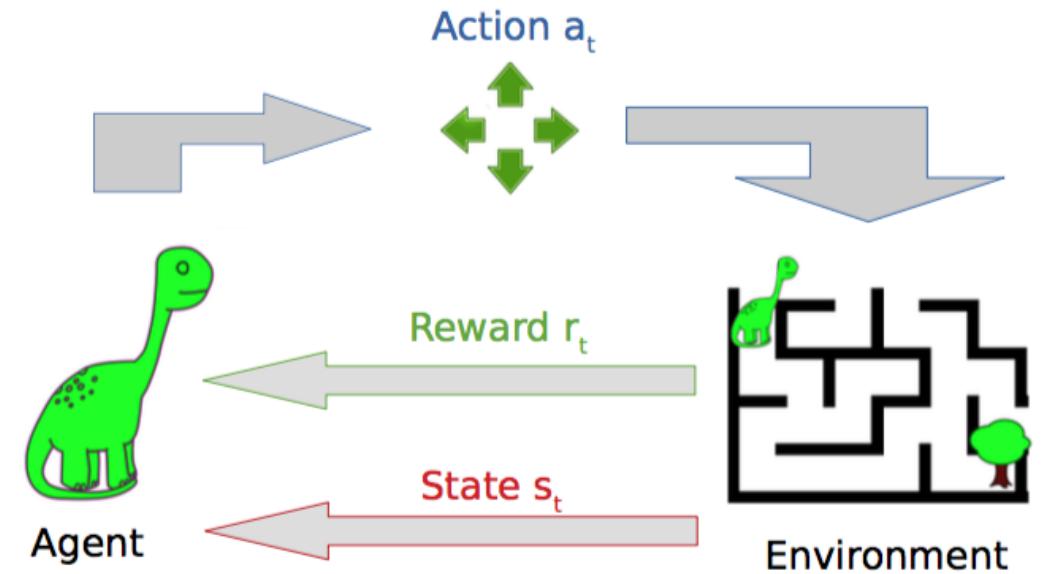
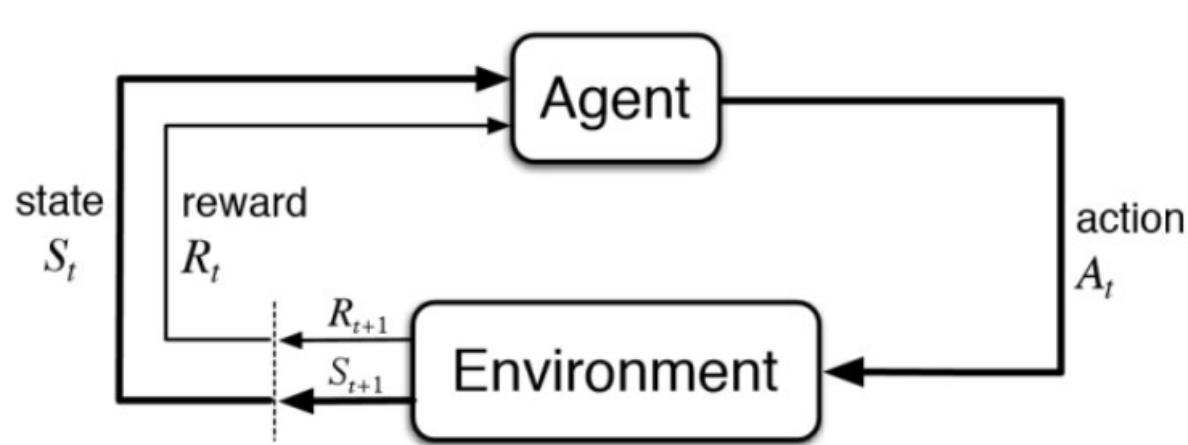
	Comedy	Action
A		
B		
C		
D		

	M1	M2	M3	M4	M5
A	3	1	1	3	1
B	1	2	4	1	3
C	3	1	1	3	1
D	4	3	5	4	4

Introduction to Machine Learning

REINFORCEMENT LEARNING

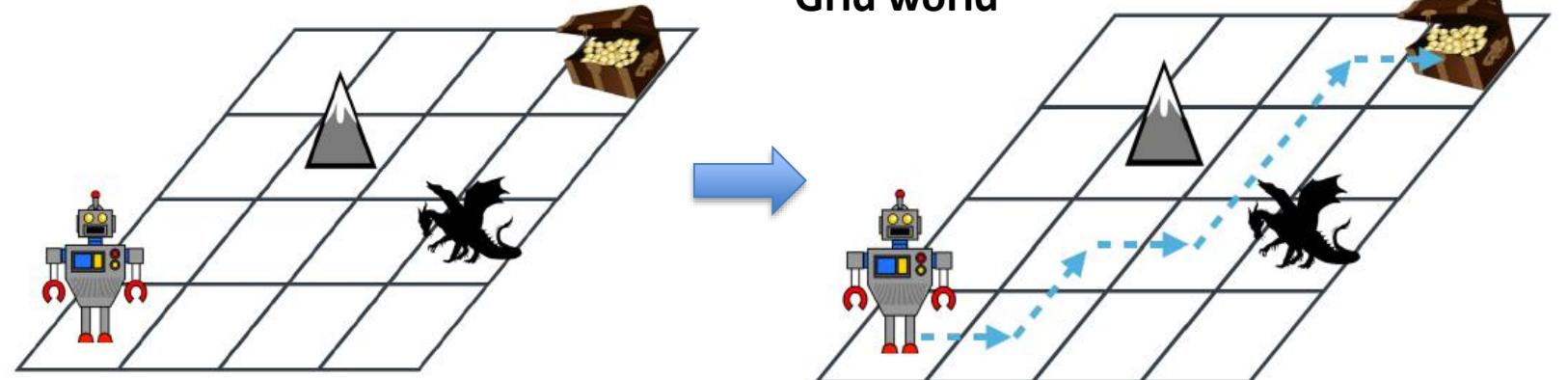
Reinforcement Learning



(source: <https://towardsdatascience.com/reinforcement-learning-101-e24b50e1d292>)

Applications

- Robotics
- Self-driving cars
- Games
- ...



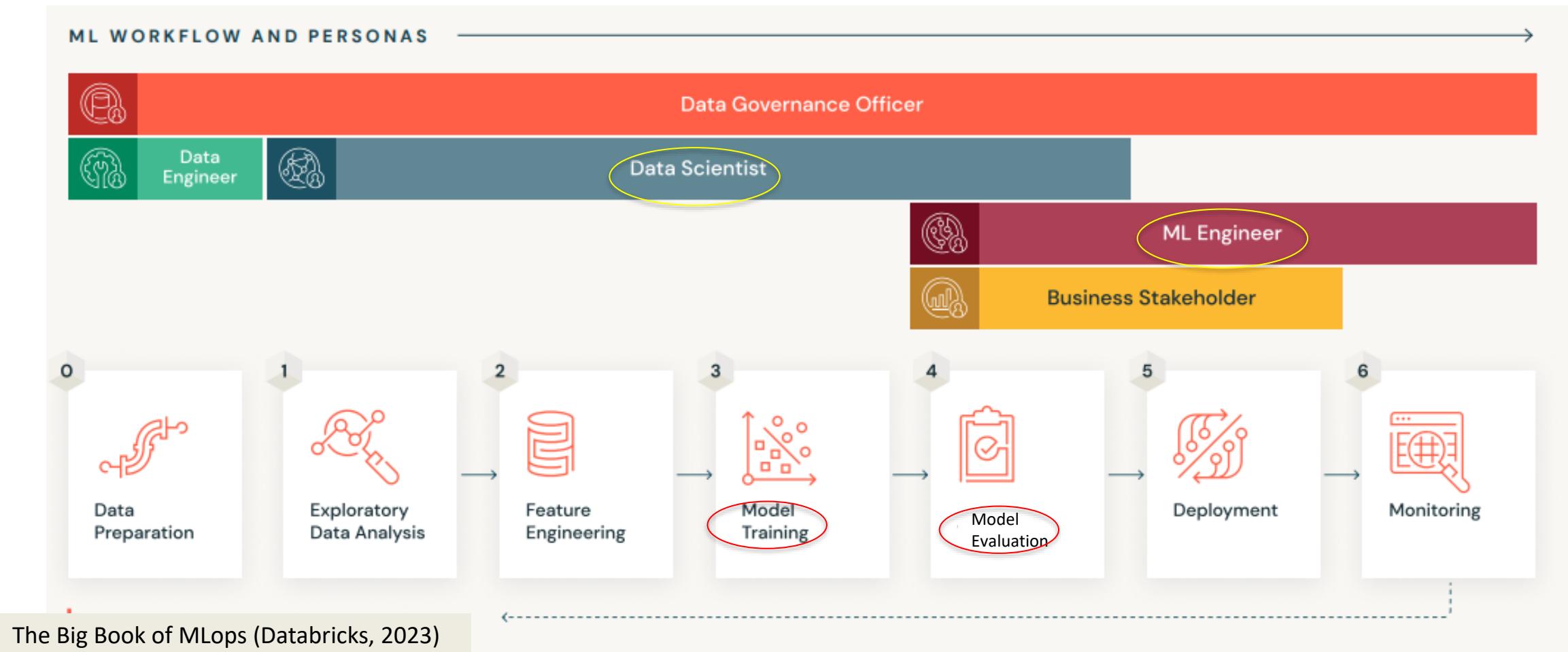
AlphaGo en AlphaZero

<https://deepmind.google/technologies/alphago/>
[AlphaGo - the movie](#)

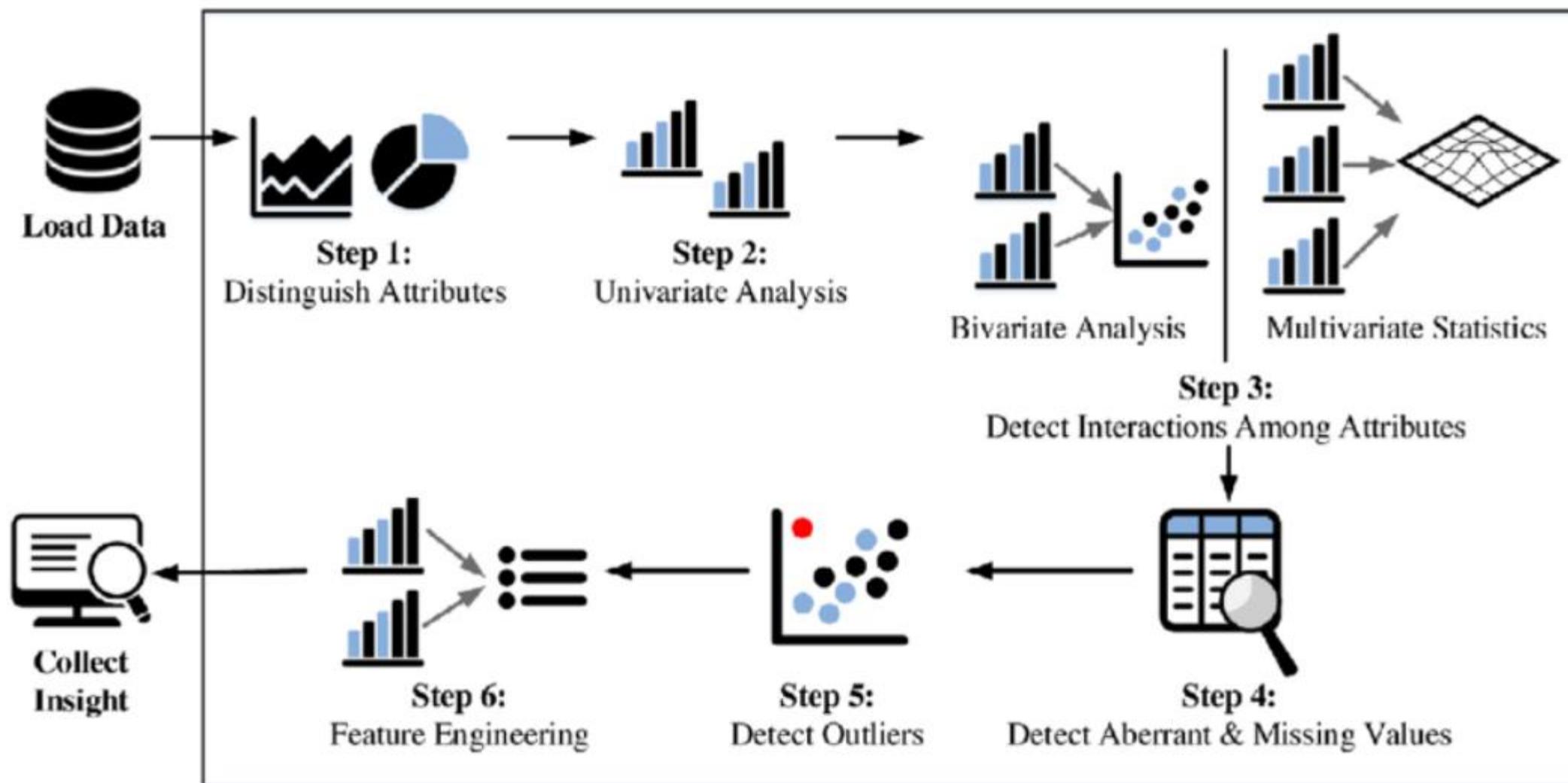
Introduction to Machine Learning

TRAINING AND EVALUATION

Machine Learning Workflow

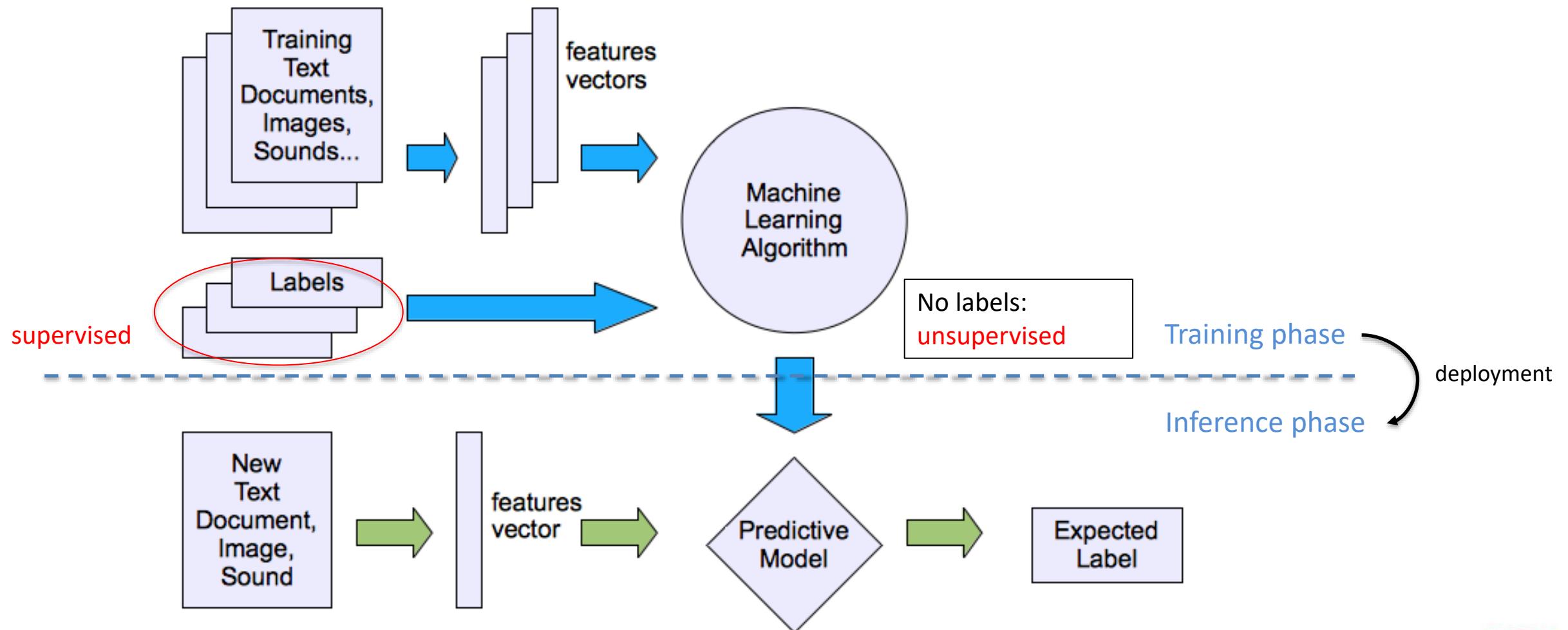


Exploratory Data Analysis



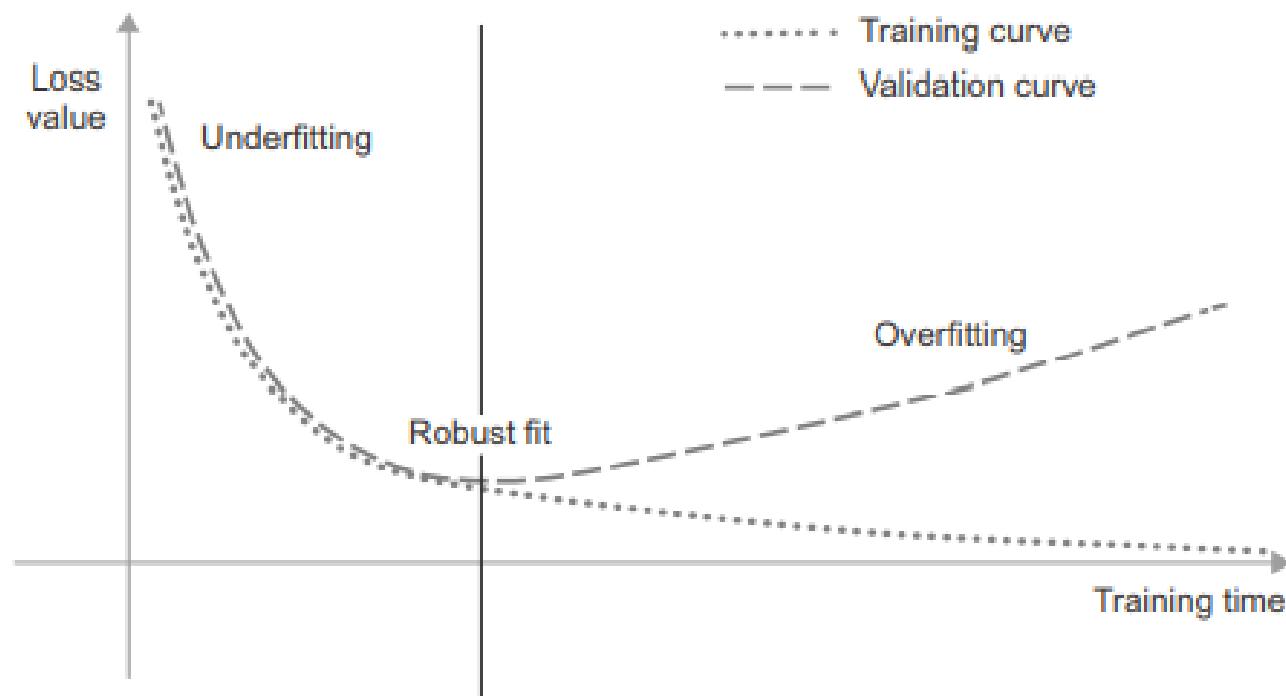
Source: <https://doi.org/10.1016/j.visinf.2018.12.004>

Training vs Inference



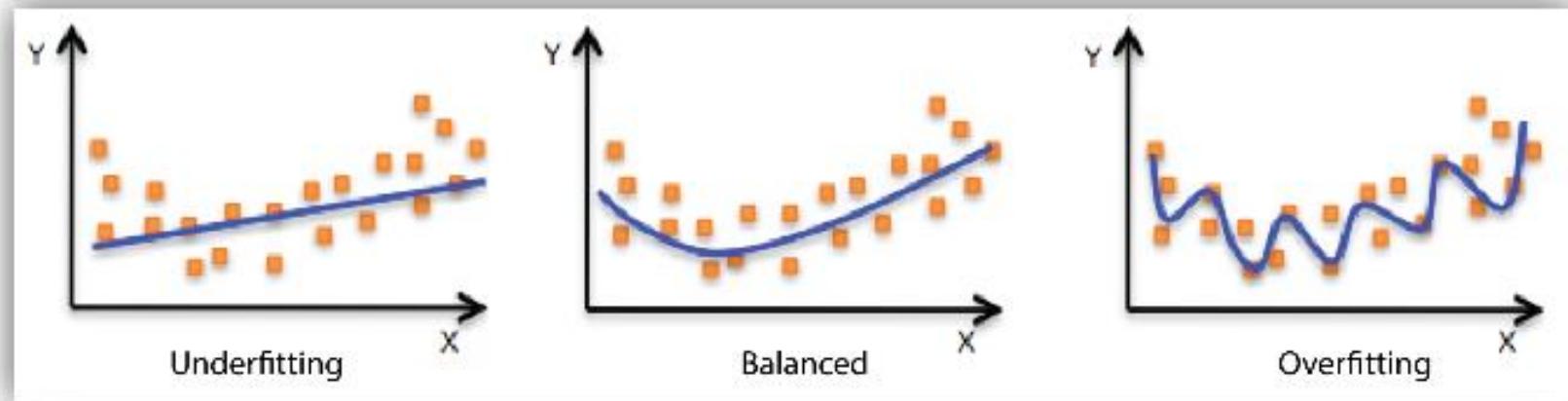
Optimization vs Generalization

- **Optimization:** fitting the data as best as possible during training
↔
- **Generalization:** good model performance on new data during inference



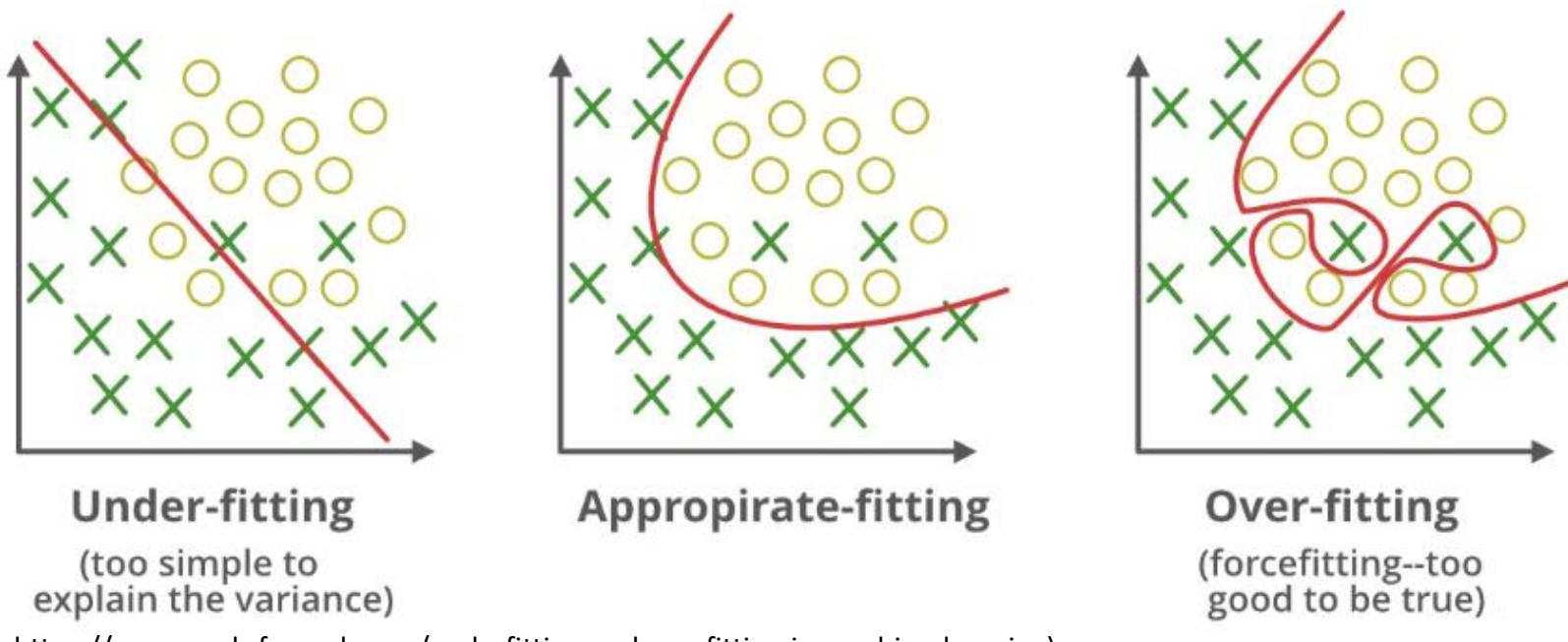
Underfitting vs Overfitting

- **Underfitting:**
model is too simple
- **Overfitting:**
model is too specific



(source: <https://towardsdatascience.com/underfitting-and-overfitting-in-machine-learning-and-how-to-deal-with-it-6fe4a8a49dbf>)

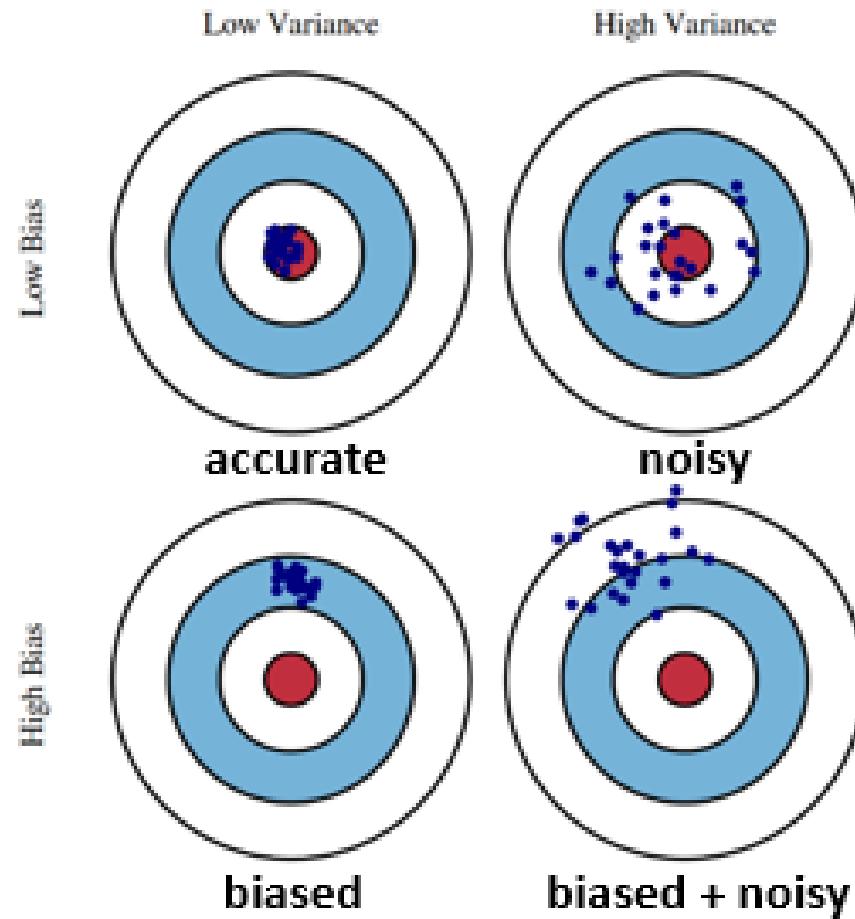
- Causes:
- Noise
 - Uncertainty
 - Rare features
 - ...



(source: <https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/>)

Bias Variance Tradeoff

- **DATA = TRUE VALUES + NOISE**
- Noise = the random irregularity we find in any real-life data

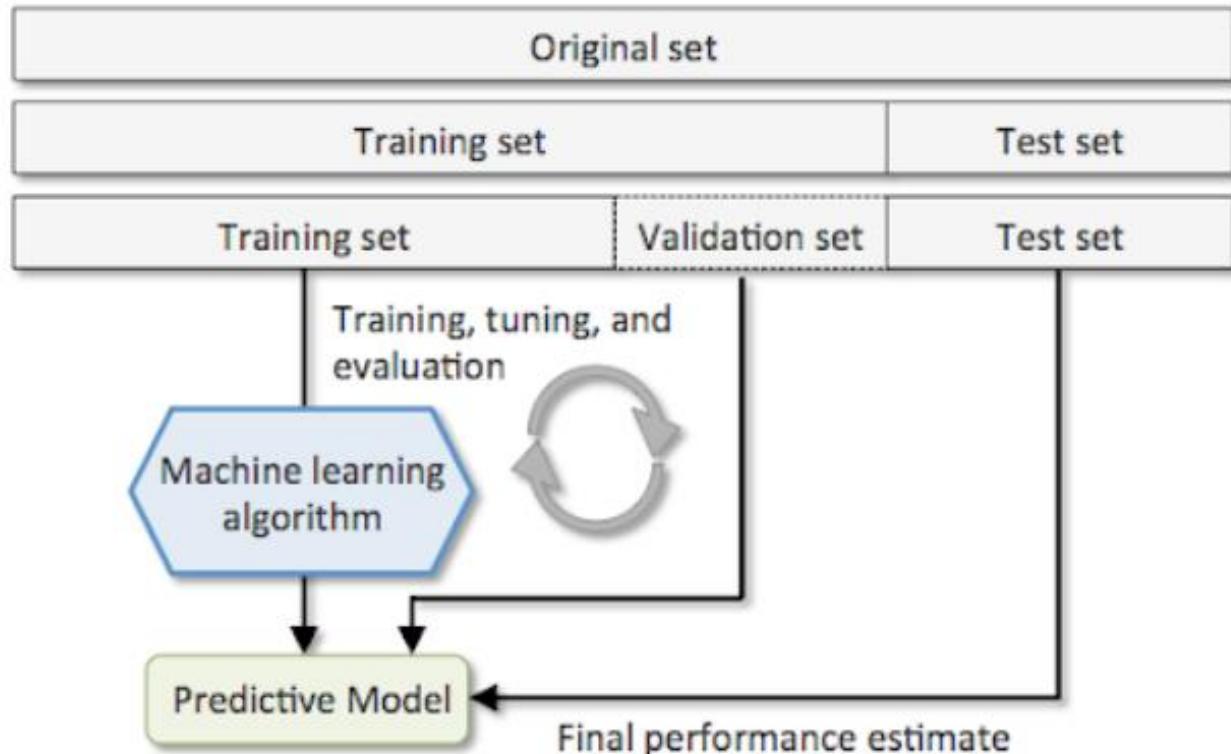


(source: <http://scott.fortmann-roe.com/docs/BiasVariance.html>)

Training, Validating, and Testing

Splitting the dataset:

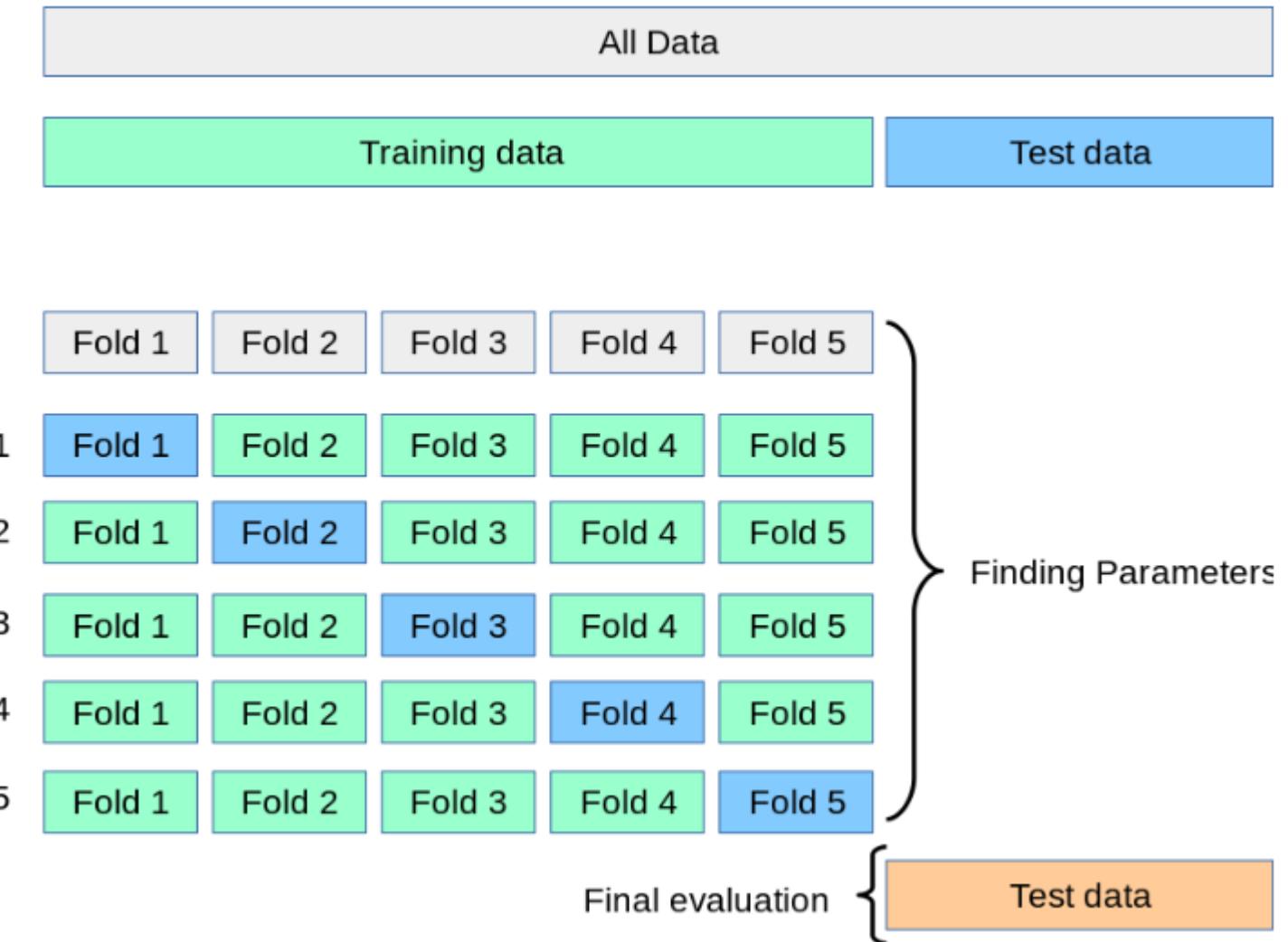
- **training:**
 - deriving the optimal model parameters
 - by the machine learning algorithm
- **validating:**
 - finding the optimal model configuration
 - fine-tuning the hyperparameters
 - to overcome overfitting
 - by the machine learning engineer
- **testing:**
 - final evaluation
 - by the machine learning engineer



(source: <https://vitalflux.com/hold-out-method-for-training-machine-learning-model>)

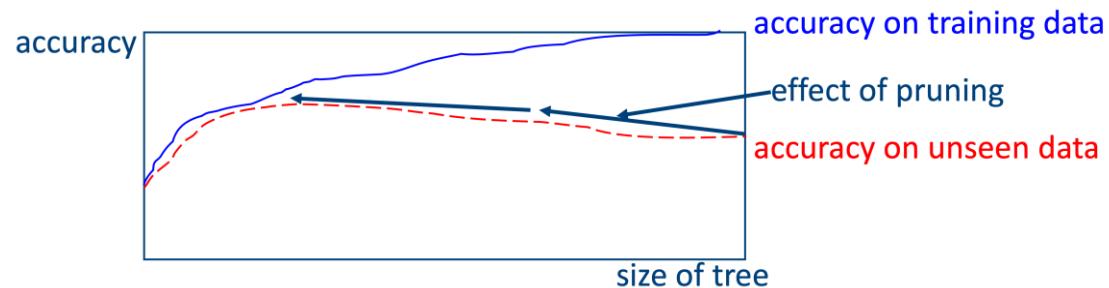
Cross Validation

Recommended in case of
small datasets!



Training vs Validation Performance

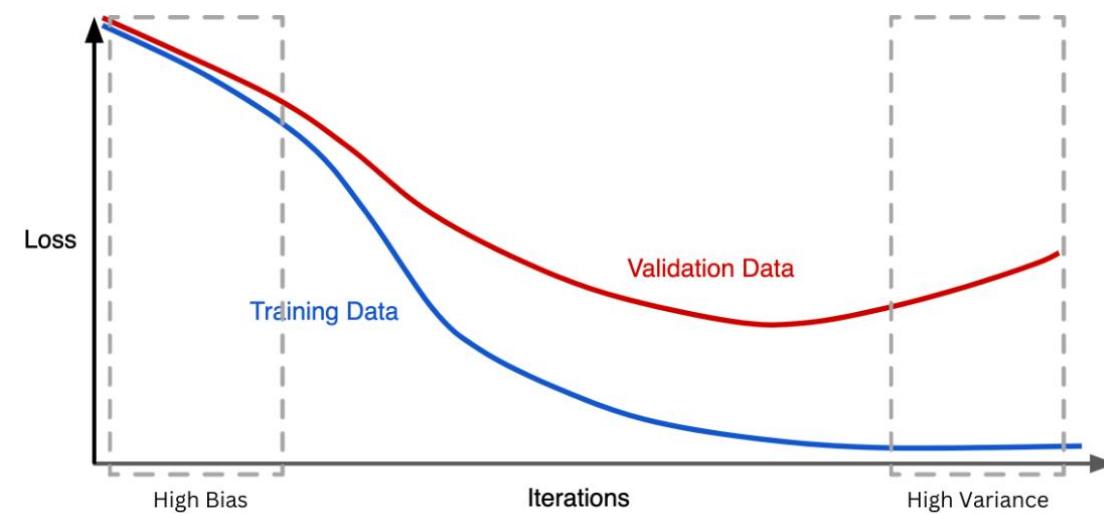
- Decision Trees



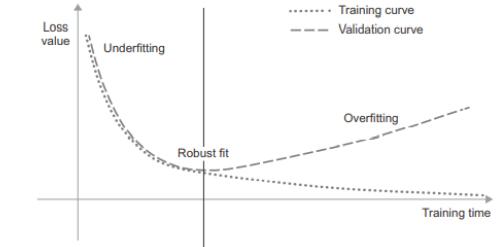
- Polynomial Regression



- Artificial Neural Networks



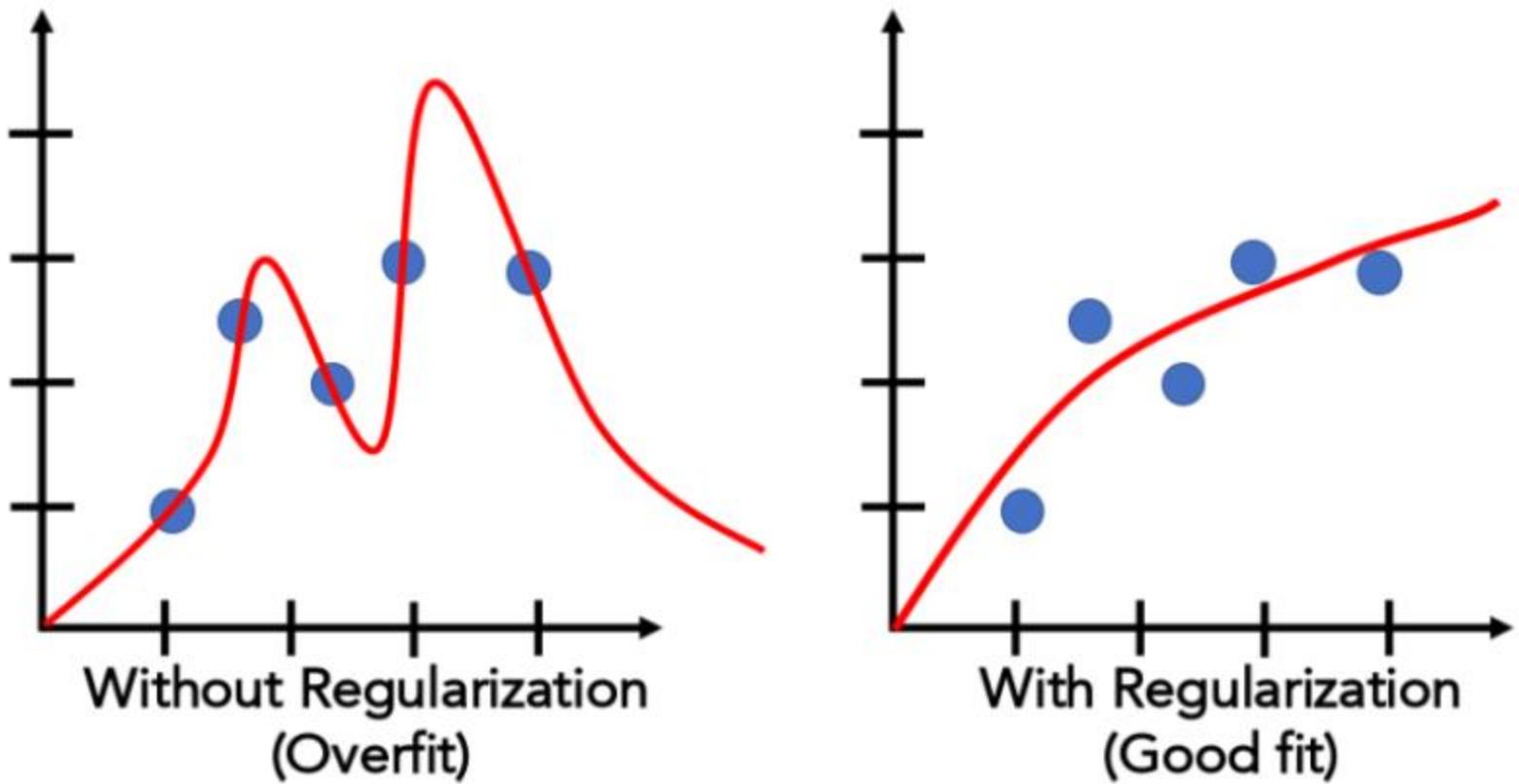
(source: <https://www.dataquest.io/blog/regularization-in-machine-learning>)



Regularization

= preventing overfitting by adding constraints or penalties to the optimization process

- LASSO = L1 regularization
- RIDGE = L2 regularization
- Elastic Net = L1 + L2
- Dropout in neural nets
- Early stopping



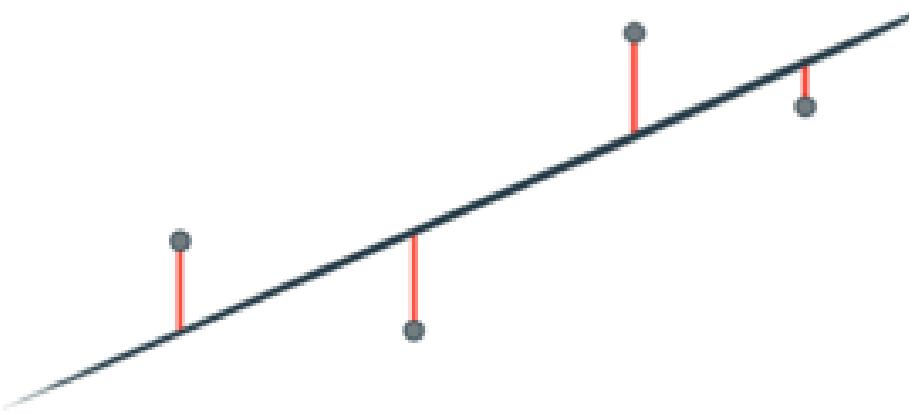
Model Evaluation

- **Loss functions:**
 - Compare predictions and true target values
 - Minimized by machine learning algorithms to obtain the best fit of the data
 - Should be mathematically convenient
- **Evaluation metrics:**
 - Also compare predictions and true target values
 - Used by machine learning engineers to evaluate the model performance
 - Easier to interpret by humans

Common Loss Functions and Metrics

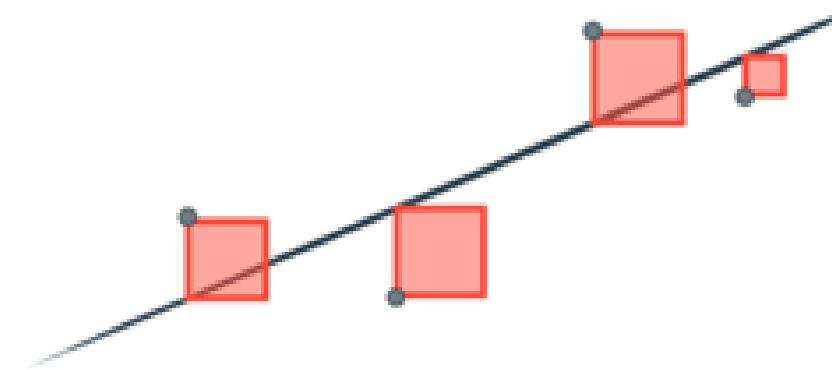
Task	Loss	Metric
Regression	Mean Squared Error (MSE) = mean of the squared differences between predictions and targets	Root Mean Squared Error (RMSE) = square root of MSE
	Mean Absolute Error (MAE) = mean of the absolute differences between predictions and targets	Coefficient of Determination (R^2) = number between 0 and 1 expressing the goodness of fit where 1 indicates a perfect fit
Classification	Cross-Entropy or Log Loss quantifies the difference between the predicted probabilities and the true labels	Accuracy = the number of correct predictions divided by the total number of samples

Regression: MAE vs MSE



$$\text{Error} = \textcolor{red}{\rule{1cm}{0.4pt}} + \textcolor{red}{\rule{1cm}{0.4pt}} + \textcolor{red}{\rule{1cm}{0.4pt}} + \textcolor{red}{\rule{1cm}{0.4pt}}$$

$$\mathbf{MAE} = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i|$$

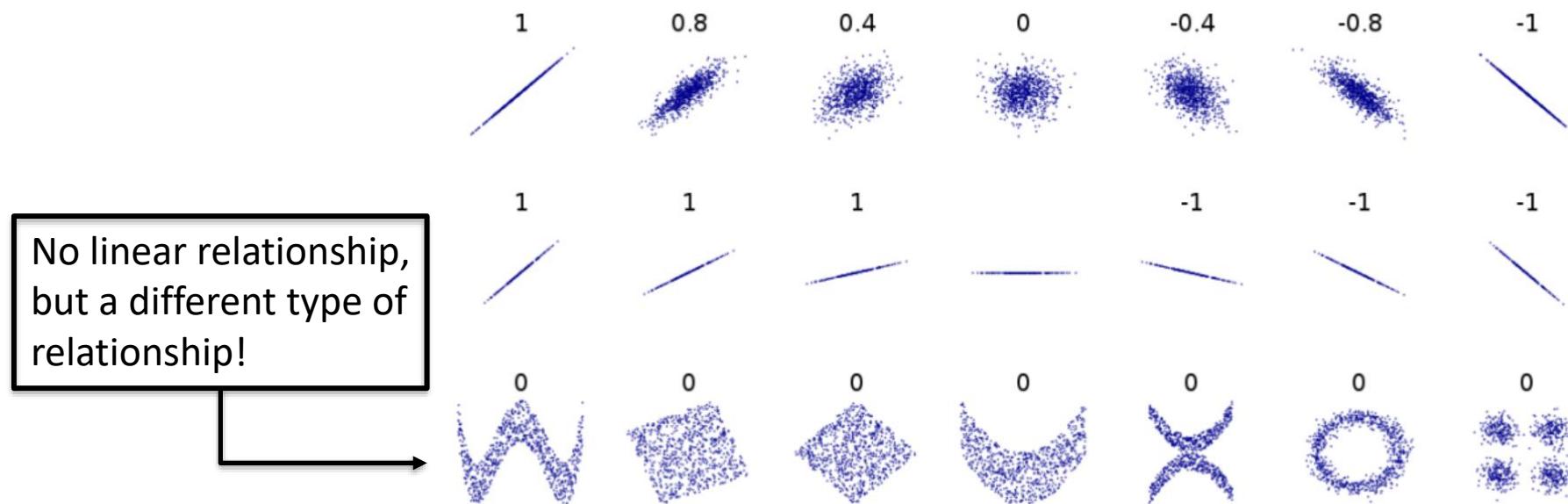


$$\text{Error} = \textcolor{red}{\boxed{\rule{1cm}{0.4pt}}} + \textcolor{red}{\boxed{\rule{1cm}{0.4pt}}} + \textcolor{red}{\boxed{\rule{1cm}{0.4pt}}} + \textcolor{red}{\boxed{\rule{1cm}{0.4pt}}}$$

$$\mathbf{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2$$

Linear Regression: R and R²

- Pearson's correlation coefficient R: number between -1 and 1



- Coefficient of determination R²: number between 0 and 1

Classification: Accuracy

- Definition: Accuracy =
$$\frac{\text{number of correctly classified samples}}{\text{total number of samples}}$$
- What if the dataset is imbalanced?
- Example:
 - 10 000 patients
 - 100 are infected with the coronavirus
 - This test has an accuracy of 99%:



```
def predict_covid(patient_data):  
    return False
```

Classification: Confusion Matrix

- **TP = True Positives**
= number of samples correctly classified as positive
- **TN = True Negatives**
= number of samples correctly classified as negative
- **FP = False Positives**
= number of samples misclassified as positive
- **FN = False Negatives**
= number of samples misclassified as negative

Actual Positive	TP	FN
Negative	FP	TN
Predicted		
Positive		Negative

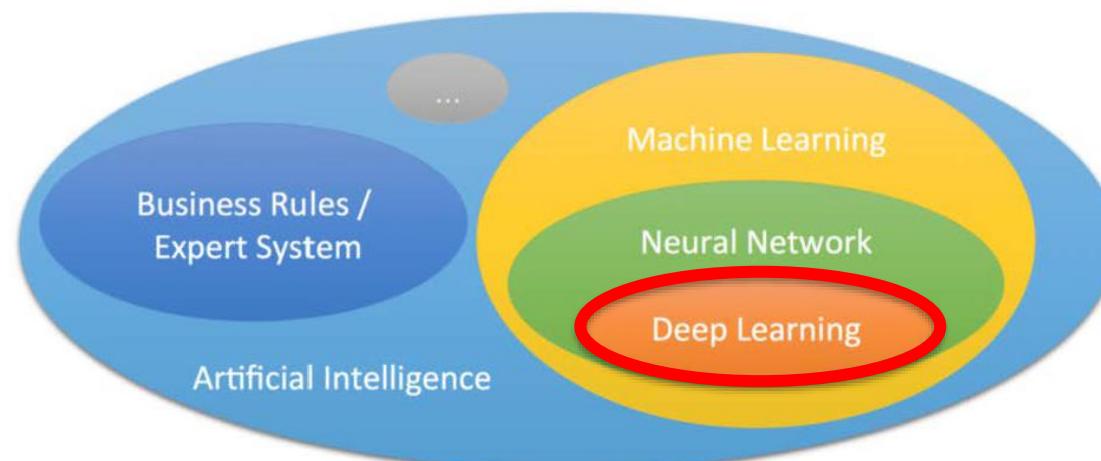
Introduction to Deep Learning

Introduction to Deep Learning

WHAT IS DEEP LEARNING?

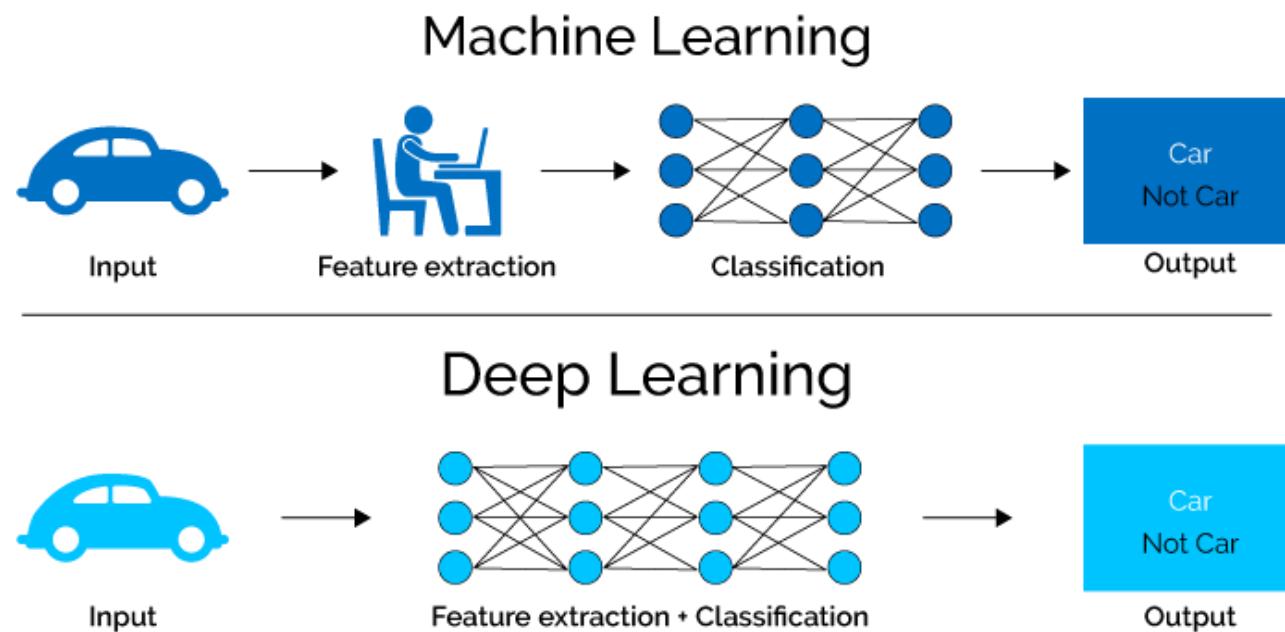
Deep Learning & Artificial Intelligence

- **Artificial Intelligence (AI):**
“The set of all tasks in which a computer can make decisions.”
- **Machine Learning (ML):**
“The set of all tasks in which a computer can make decisions based on data.”
- **Deep Learning (DL):**
“The field of machine learning that uses certain objects called neural networks.”



What is deep learning?

- Subfield of ML for **learning representations** of data
- Exceptional effective at **learning patterns**
- Utilizes learning algorithms that derive **meaning** out of data by using a **hierarchy of multiple layers**
- If you provide the system **large amounts of training data**, it begins to understand it and respond in useful ways
- It can learn **both unsupervised and supervised**
- It is highly suitable for **parallelization** on GPUs



Deep learning: use cases



- Computer vision (CV)
- Face recognition
- Natural language processing (NLP)
- Speech recognition
- Generative AI
- Self-driving cars
- ...
- Even in your mobile device
 - Siri
 - Google assistant

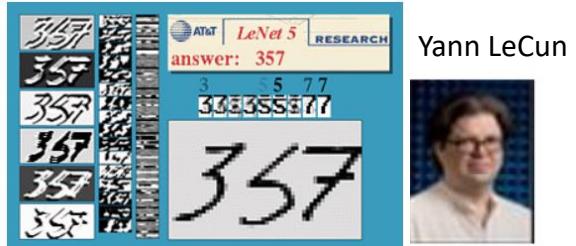
Deep learning: brief history

Frank Rosenblatt



1958 Perceptron

1974 Backpropagation



Convolution Neural Networks for Handwritten Recognition

1998



Google Brain Project on 16k Cores

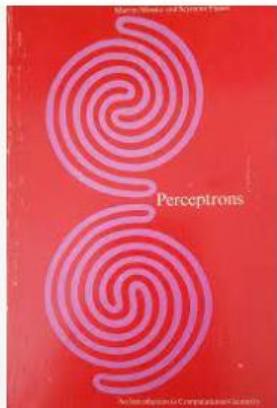
2012

I am ChatGPT, created by OpenAI.



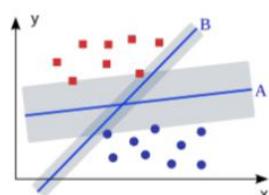
2022

1969
Perceptron criticized



awkward silence (AI Winter)

1995
SVM reigns



2006
Restricted Boltzmann Machine



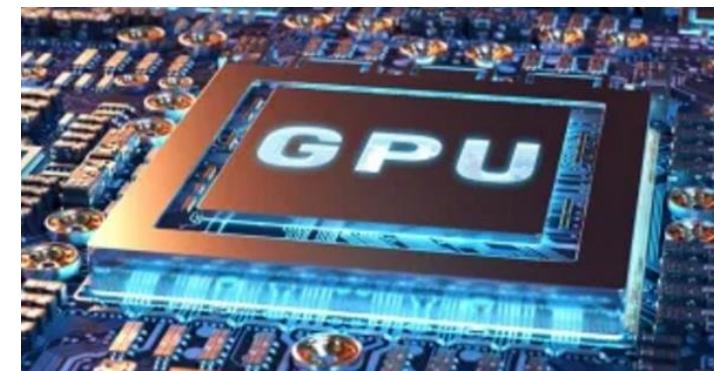
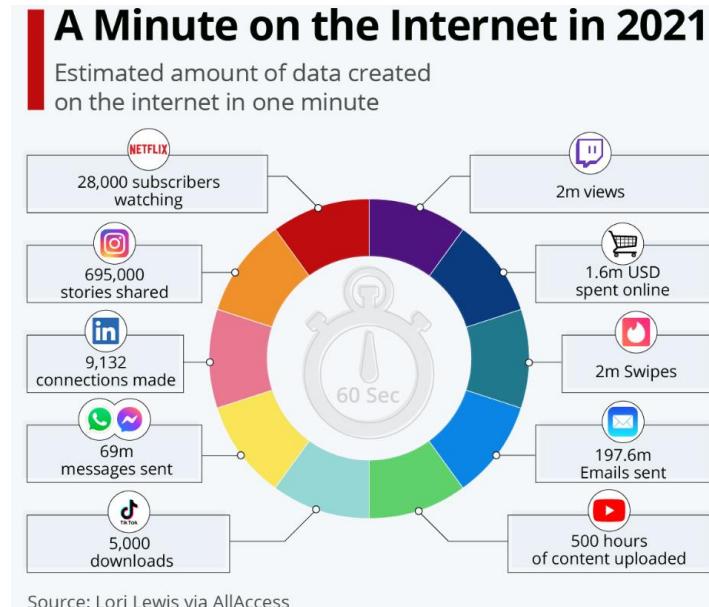
2012
AlexNet wins ImageNet

vives

University
of Applied
Sciences

Why did deep learning only take off after 2012?

- Deep learning algorithms already known in the 1990s
- Yet a breakthrough only occurred in 2012!
 - Bottleneck in the 1990s and 2000s: **data and hardware**
 - Much more data available through the internet
 - Gaming industry developed better hardware, e.g. GPUs



The democratization of deep learning

- Deep learning initially required knowledge of C/C++ and CUDA
- Now there are many accessible high-level (Python) libraries

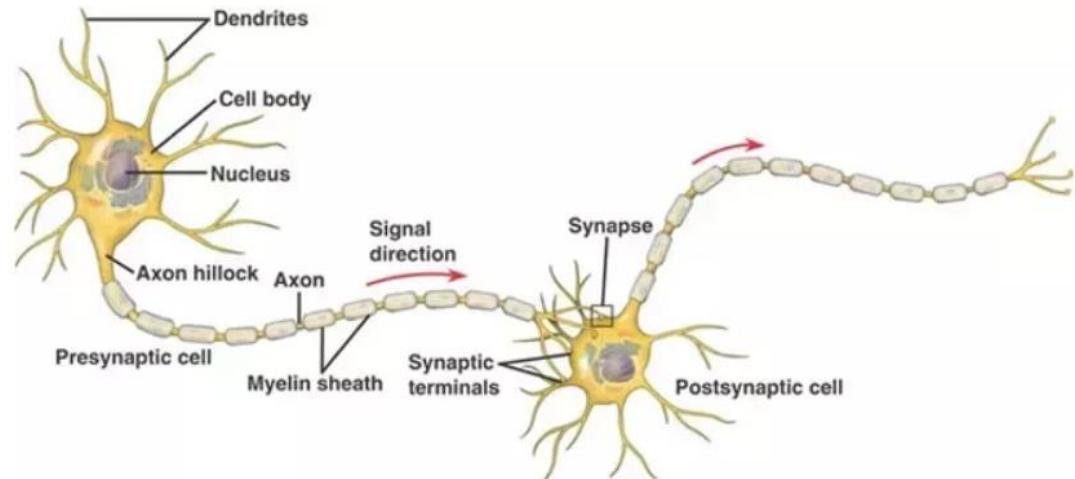
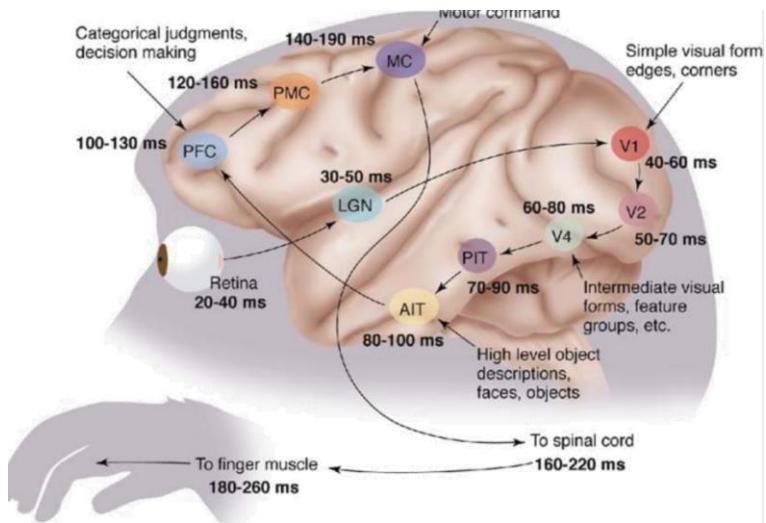


Introduction to Deep Learning

NEURAL NETWORKS

Deep learning: inspired by the brain

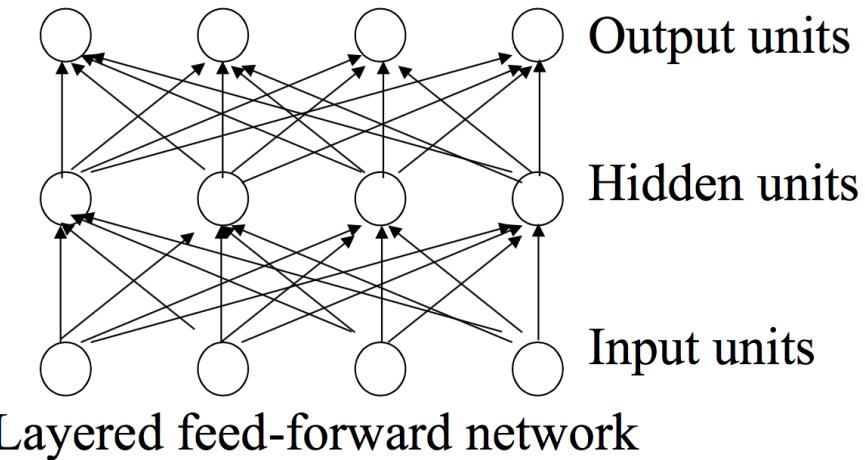
but not a model of the brain!



Our brain has lots of neurons connected together and the strength of the connections between neurons represents long term knowledge.

Artificial neural networks

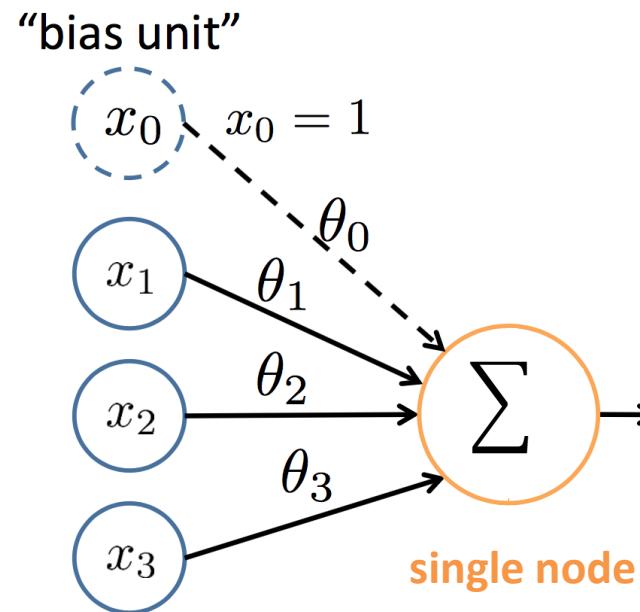
Neural networks



- Neural networks are made up of **nodes** or **units**, connected by **links**
- Each link has an associated **weight** and **activation level**
- Each node has an **input function** (typically summing over weighted inputs), an **activation function**, and an **output**

Linear regression

Neuron Model: Linear Unit



$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

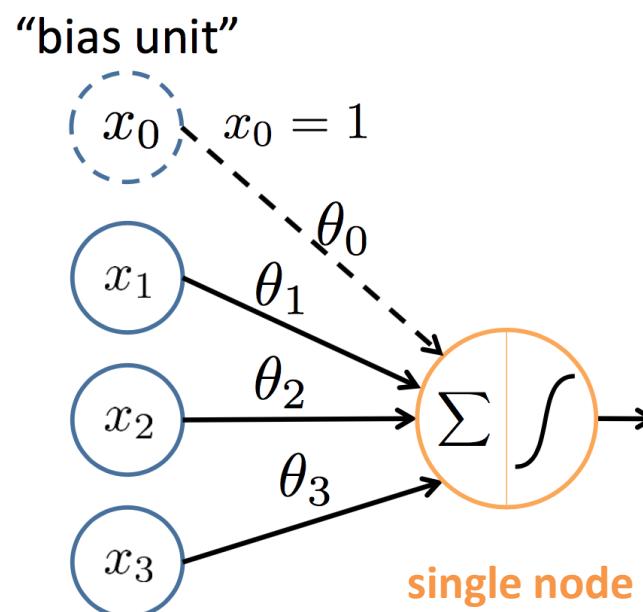
input data **weights or parameters**

$$h_{\theta}(\mathbf{x}) = \theta^T \mathbf{x}$$

= **weighted sum**

Logistic regression

Neuron Model: Logistic Unit



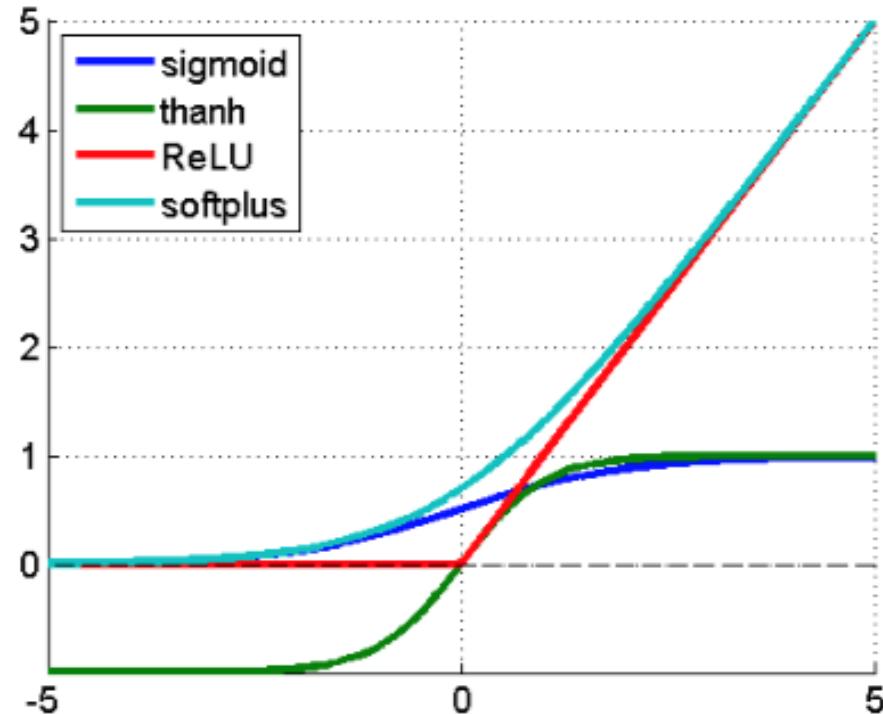
$$\mathbf{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix}$$

input data **weights or parameters**

$$h_{\boldsymbol{\theta}}(\mathbf{x}) = g(\boldsymbol{\theta}^T \mathbf{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^T \mathbf{x}}} \quad = \text{activation of weighted sum}$$

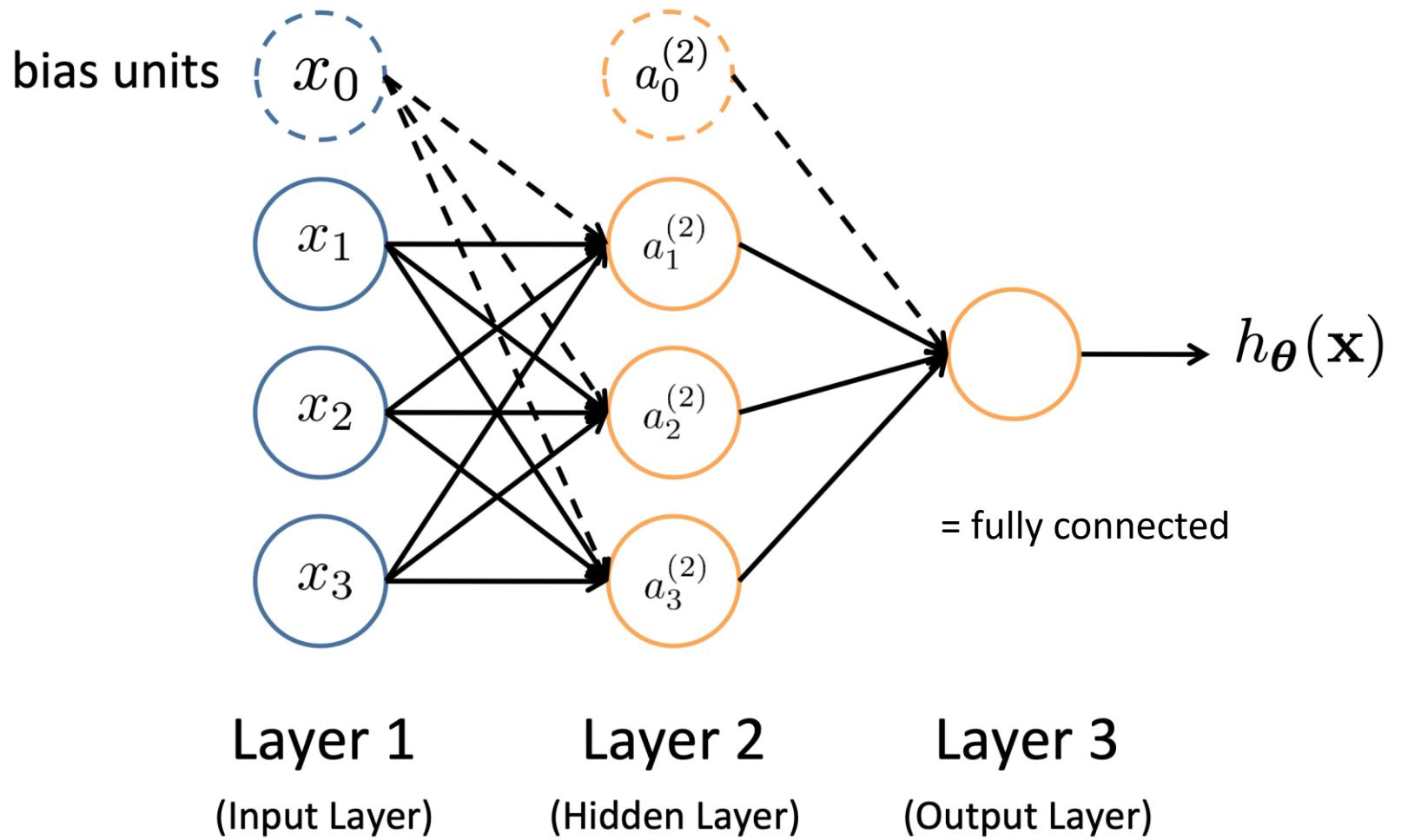
Sigmoid (logistic) **activation function**: $g(z) = \frac{1}{1 + e^{-z}}$

Activation functions



Most deep networks use **ReLU** - $\max(0, x)$ - nowadays for hidden layers, since it trains much faster, is more expressive than logistic function and prevents the gradient vanishing problem.

Layered networks with multiple neurons



DEEP learning =
minimum 3 layers!

Example: image classification



Pedestrian



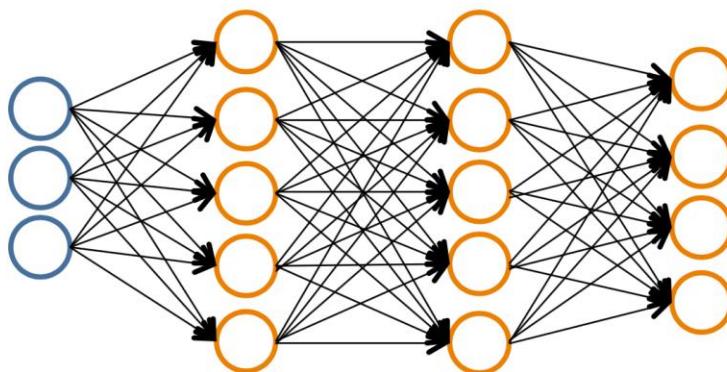
Car



Motorcycle



Truck



$$h_{\Theta}(\mathbf{x}) \in \mathbb{R}^K$$

We want:

$$h_{\Theta}(\mathbf{x}) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

when pedestrian

$$h_{\Theta}(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

when car

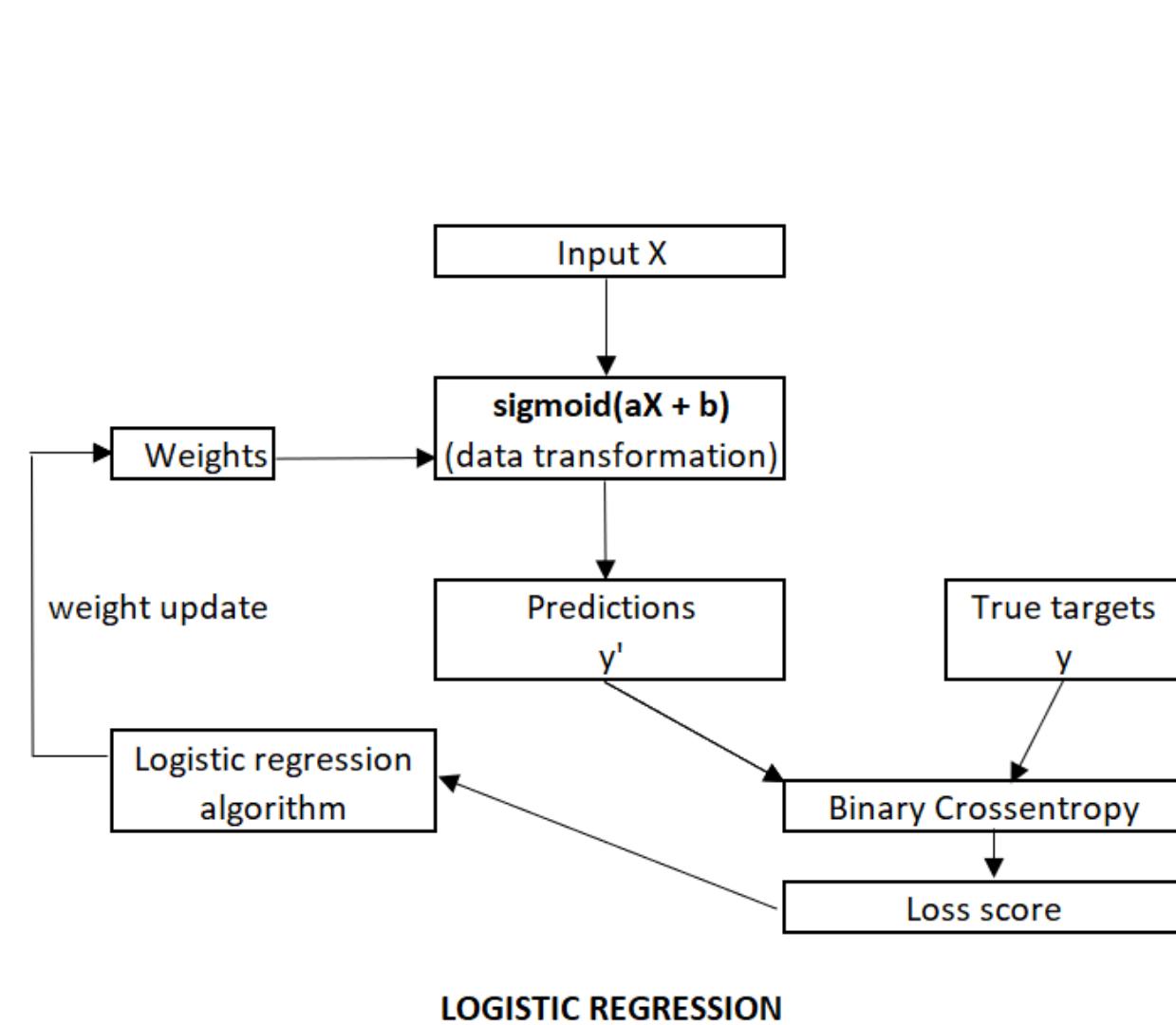
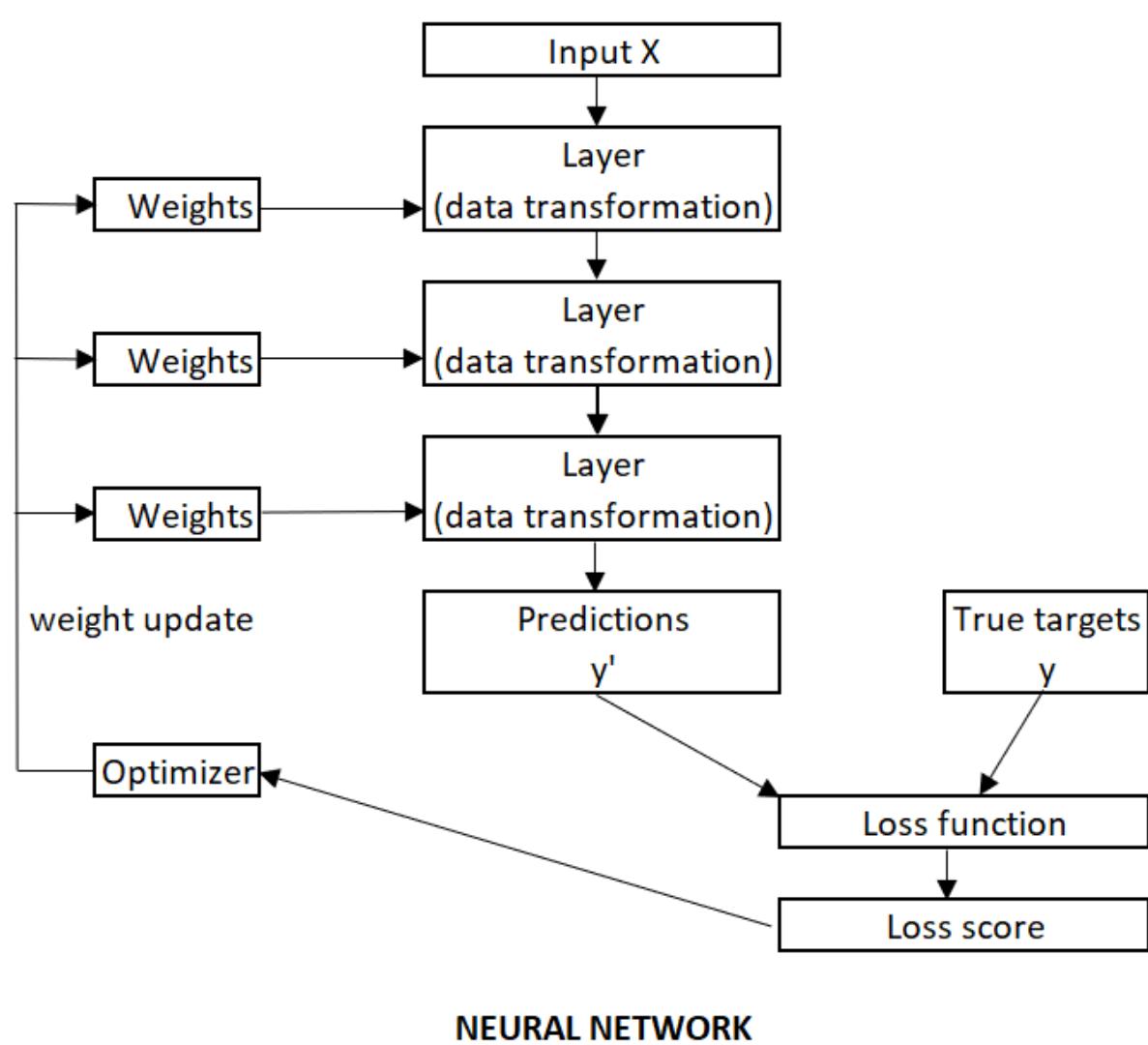
$$h_{\Theta}(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

when motorcycle

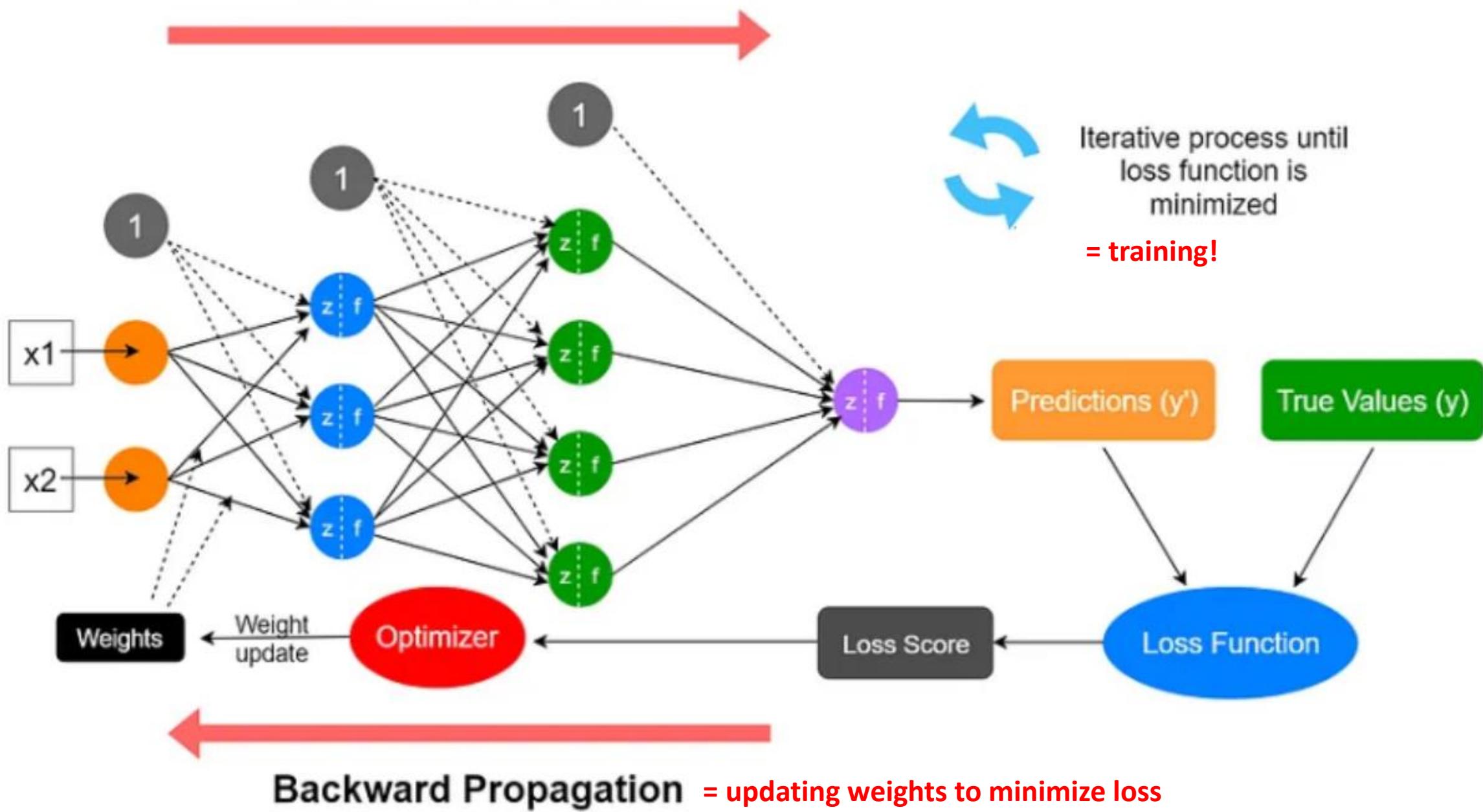
$$h_{\Theta}(\mathbf{x}) \approx \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

when truck

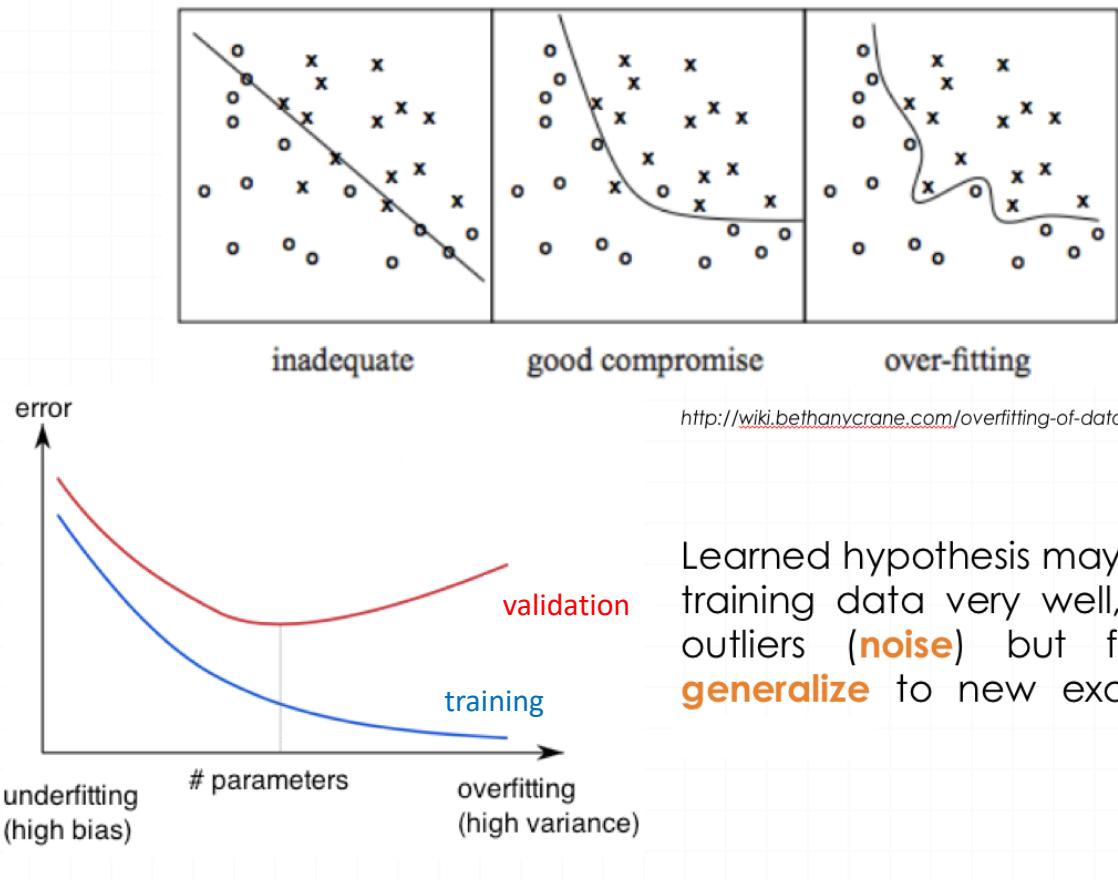
Understanding how neural networks learn



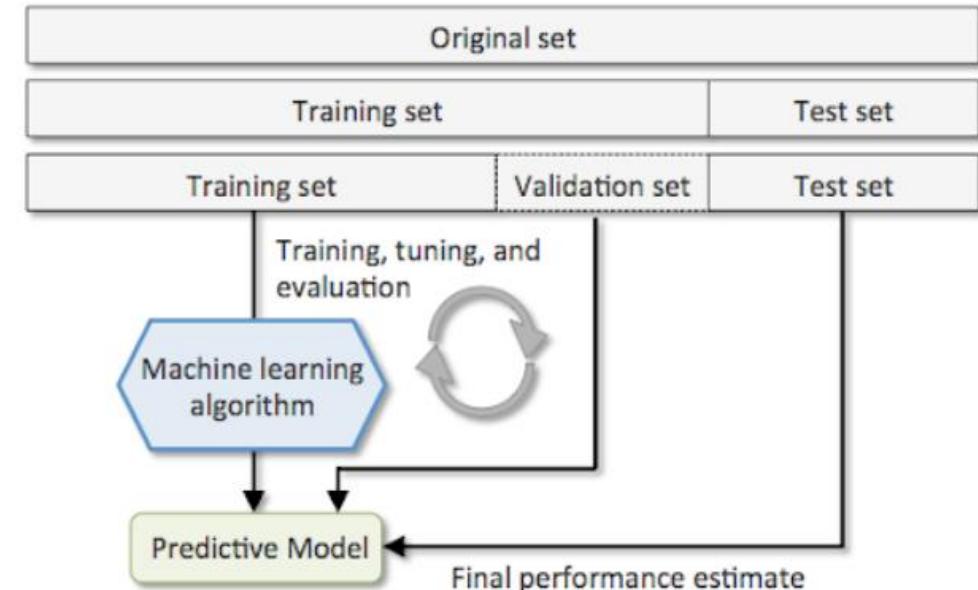
Forward Propagation = calculating predictions and loss



Overfitting is still an issue!



Learned hypothesis may **fit** the training data very well, even outliers (**noise**) but fail to **generalize** to new examples



Splitting the dataset!

Introduction to Deep Learning

A FIRST NEURAL NETWORK

A first neural network

- MNIST dataset
 - **Classifying** handwritten digits = 10 classes (0 to 9)
 - 60000 training images, 10000 test images



```
from tensorflow.keras.datasets import mnist  
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

- First important step: exploring the data!
 - Images are 2D arrays of integers between 0 and 255

Note on classes and labels

In machine learning, a category in a classification problem is called a **class**. Data points are called **samples**. The class associated with a specific sample is called a **label**.

A first neural network

- **Preparing the data**
 - Reshape to 1D vectors
 - Convert to floats between 0 and 1

```
train_images = train_images.reshape((60000, 28 * 28))
train_images = train_images.astype("float32") / 255
test_images = test_images.reshape((10000, 28 * 28))
test_images = test_images.astype("float32") / 255
```

A first neural network

- An artificial neural network consists of **layers**
- A layer = a *filter* that distills meaningful **representations** from the data

```
from tensorflow import keras
from tensorflow.keras import layers
model = keras.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(10, activation="softmax")
])
```

= *the network architecture*

A first neural network

- Compiling the model:
 - **Optimizer**
 - **Loss function**
 - **Evaluation metrics**

```
model.compile(optimizer="rmsprop",
              loss="sparse_categorical_crossentropy",
              metrics=["accuracy"])
```

A first neural network

- Fitting the data = **training** the model

```
>>> model.fit(train_images, train_labels, epochs=5, batch_size=128)
Epoch 1/5
60000/60000 [=====] - 5s - loss: 0.2524 - acc: 0.9273
Epoch 2/5
51328/60000 [=====>.....] - ETA: 1s - loss: 0.1035 - acc: 0.9692
```

- **Epochs** = the number of times the entire dataset is run through by the algorithm
- **Batch size** = the number of training examples that are grouped during one iteration
- Usually the model is also being **validated** during training:
 - to detect and overcome overfitting
 - to finetune the hyperparameters

A first neural network

- Making predictions:

```
>>> test_digits = test_images[0:10]
>>> predictions = model.predict(test_digits)
>>> predictions[0]
array([1.0726176e-10, 1.6918376e-10, 6.1314843e-08, 8.4106023e-06,
       2.9967067e-11, 3.0331331e-09, 8.3651971e-14, 9.9999106e-01,
       2.6657624e-08, 3.8127661e-07], dtype=float32)
```

Method predict returns probabilities!

A first neural network

- Maximum probability = class to which a sample “probably” belongs

```
>>> predictions[0].argmax()
7
>>> predictions[0][7]
0.99999106
```

```
>>> test_labels[0]
7
```

The predicted class is correct in this example!

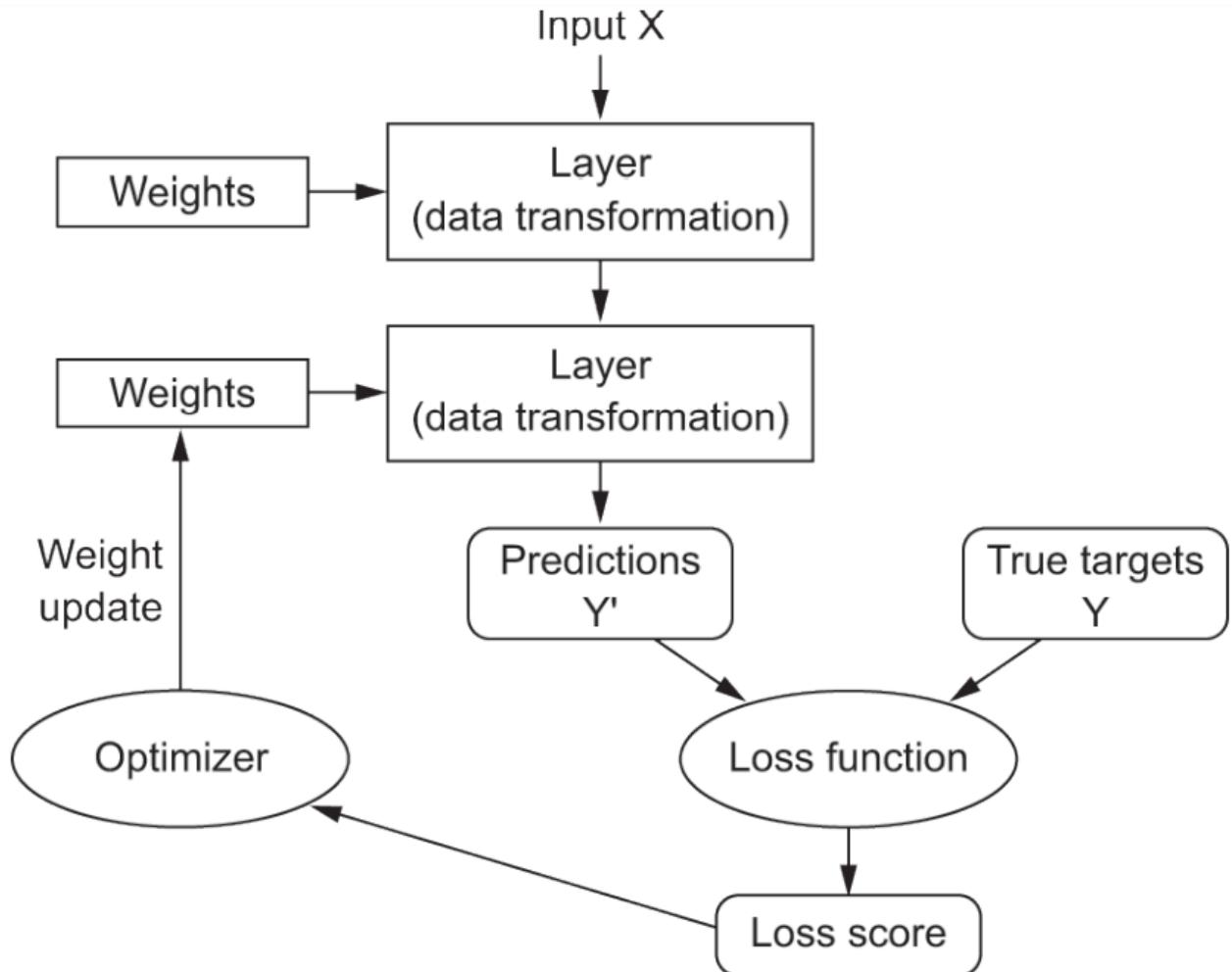
A first neural network

- **Evaluating** the model using the **testset**

```
>>> test_loss, test_acc = model.evaluate(test_images, test_labels)
>>> print(f"test_acc: {test_acc}")
test_acc: 0.9785
```

Overfitting?

A first neural network



```
model = models.Sequential([
    layers.Dense(512, activation="relu"),
    layers.Dense(10, activation="softmax")
])

model.compile(
    optimizer="rmsprop",
    loss="sparse_categorical_crossentropy",
    metrics=["accuracy"]
)

model.fit(train_images, train_labels,
          epochs=5, batch_size=128);
```

Introduction to Deep Learning

DATA REPRESENTATIONS FOR NEURAL NETS: TENSORS

Tensors

Tensor = multi-dimensional array

- = **basic data structure** for representing and manipulating data in deep learning
- = NDarray in NumPy

- **Scalar** = rank-0 tensor (0 axes)

```
>>> import numpy as np  
>>> x = np.array(12)  
>>> x  
array(12)  
>>> x.ndim  
0
```

- **Vector** = rank-1 tensor (1 axis)

```
>>> x = np.array([12, 3, 6, 14, 7])  
>>> x  
array([12, 3, 6, 14, 7])  
>>> x.ndim  
1
```

Tensors

- **Matrix** = rank-2 tensor (2 axes)

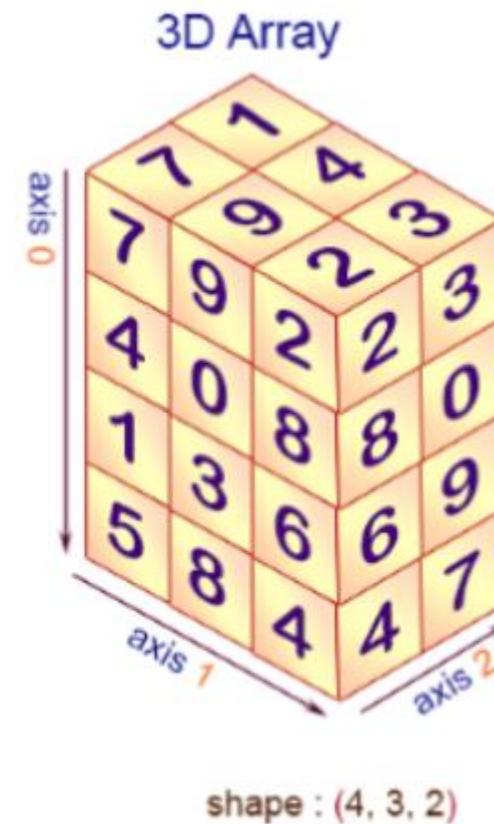
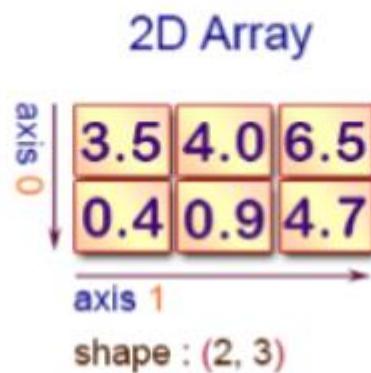
```
>>> x = np.array([[5, 78, 2, 34, 0],  
                 [6, 79, 3, 35, 1],  
                 [7, 80, 4, 36, 2]])  
>>> x.ndim  
2
```

- **Multi-dimensional array** = Rank-3 or higher tensor (N axes with N >= 3)

```
>>> x = np.array([[[5, 78, 2, 34, 0],  
                  [6, 79, 3, 35, 1],  
                  [7, 80, 4, 36, 2]],  
                  [[5, 78, 2, 34, 0],  
                   [6, 79, 3, 35, 1],  
                   [7, 80, 4, 36, 2]],  
                  [[5, 78, 2, 34, 0],  
                   [6, 79, 3, 35, 1],  
                   [7, 80, 4, 36, 2]]])  
>>> x.ndim  
3
```

Tensors

NumPy arrays: axes



Tensors

A tensor is defined by three key attributes:

- *Number of axes (rank)*—For instance, a rank-3 tensor has three axes, and a matrix has two axes. This is also called the tensor’s `ndim` in Python libraries such as NumPy or TensorFlow.
- *Shape*—This is a tuple of integers that describes how many dimensions the tensor has along each axis. For instance, the previous matrix example has shape `(3, 5)`, and the rank-3 tensor example has shape `(3, 3, 5)`. A vector has a shape with a single element, such as `(5,)`, whereas a scalar has an empty shape, `()`.
- *Data type* (usually called `dtype` in Python libraries)—This is the type of the data contained in the tensor; for instance, a tensor’s type could be `float16`, `float32`, `float64`, `uint8`, and so on. In TensorFlow, you are also likely to come across `string` tensors.

Tensors

- The MNIST dataset is also a tensor!

```
from tensorflow.keras.datasets import mnist  
(train_images, train_labels), (test_images, test_labels) = mnist.load_data()
```

Next, we display the number of axes of the tensor `train_images`, the `ndim` attribute:

```
>>> train_images.ndim  
3
```

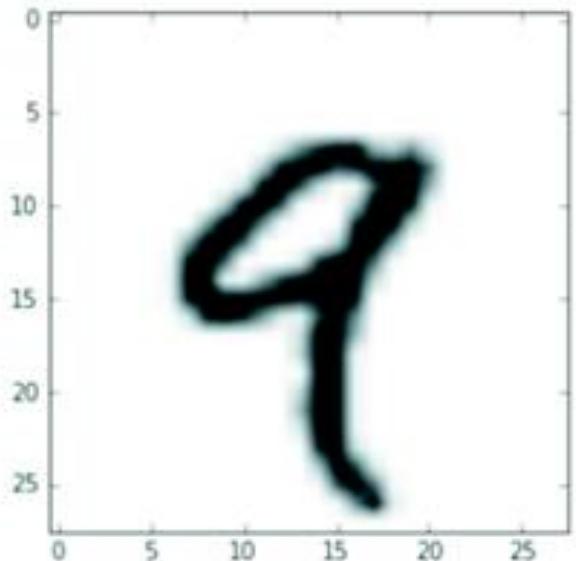
Here's its shape:

```
>>> train_images.shape  
(60000, 28, 28)
```

And this is its data type, the `dtype` attribute:

```
>>> train_images.dtype  
uint8
```

```
import matplotlib.pyplot as plt  
digit = train_images[4]  
plt.imshow(digit, cmap=plt.cm.binary)  
plt.show()
```



```
>>> train_labels[4]  
9
```

Tensors

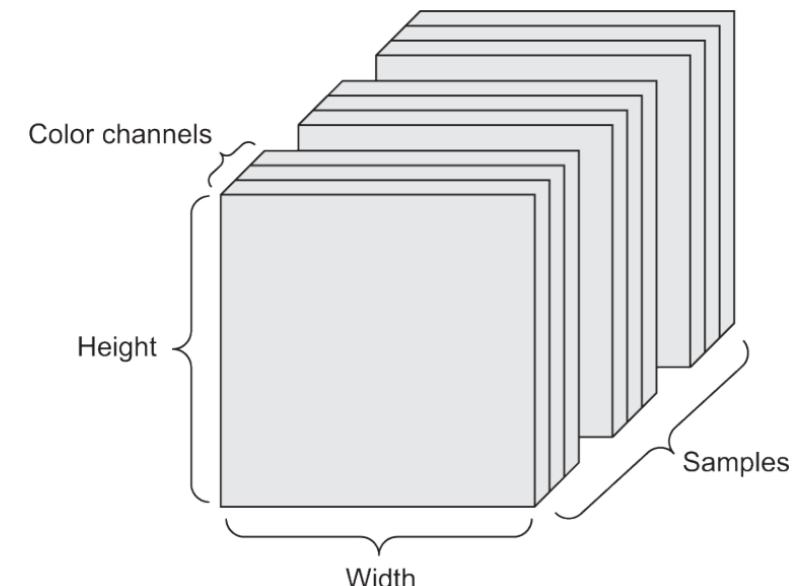
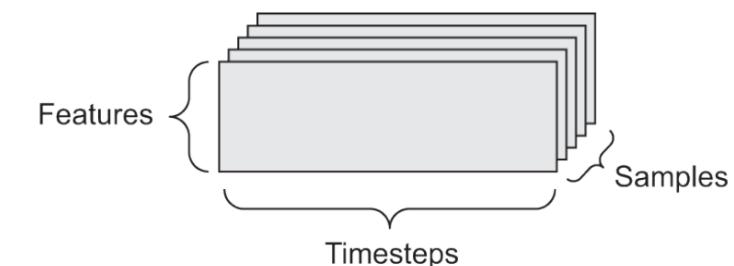
- **Vector data:** (samples, feature)
 - Tables and matrices (e.g. csv-files)
- **Time series:** (samples, timesteps, features)
 - E.g. dataset of stock prices, tweets, ...
- **Images:** (samples, width, height, channels)
 - E.g. MNIST dataset
- **Videos:** (samples, frames, height, width, channels)
 - E.g. YouTube filmpjes

data(set)

A	B	C	D	E	F	
1	id	date	size	typos	recipients	spam
2	0	12/01/2021	2.5	0	1	False
3	1	13/01/2021	1.3	0	2	False
4	2	14/01/2021	12.1	3	15	True
5	3	15/01/2021	7.8	2	19	True
6	4	16/01/2021	4.6	1	5	False
7	5	17/01/2021	9.8	5	1	True
8	6	18/01/2021	11.6	3	63	True

example

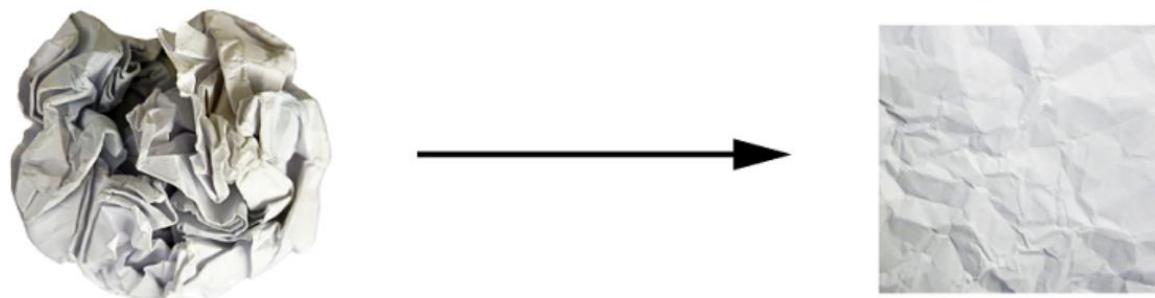
feature target



Tensors

- **Each layer** in the neural network performs **tensor operations** on the data
- These tensor operations **transform the data** into meaningful representations
- Tensor operations can be **interpreted geometrically**...
- ... and so can neural networks:

Uncrumpling a complicated manifold of data



Introduction to Deep Learning

THE ENGINE OF NEURAL NETS: GRADIENT-BASED OPTIMIZATION

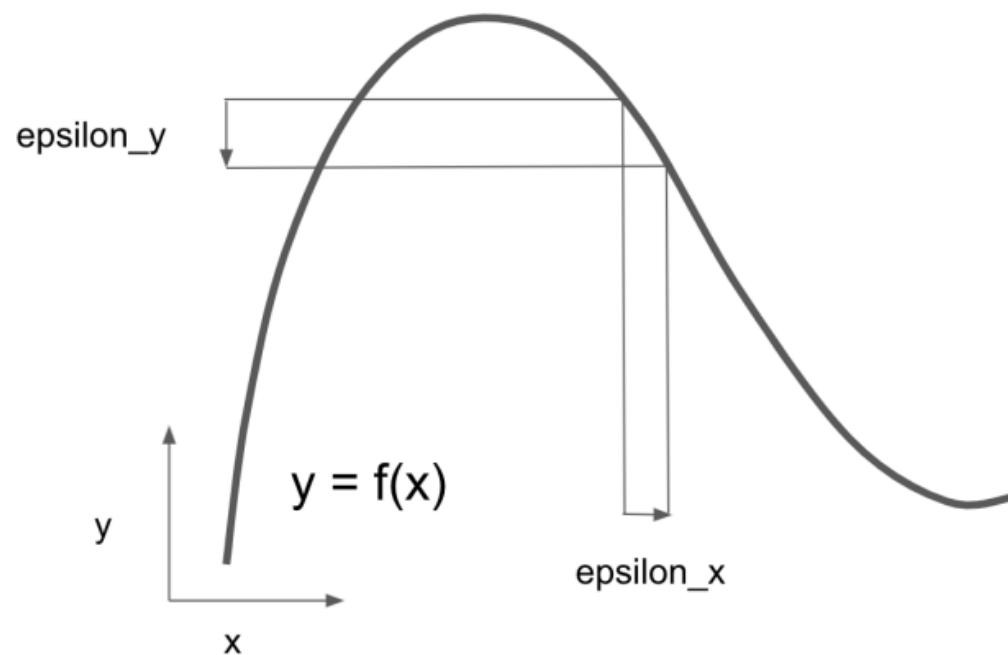
Gradient-based optimization

```
output = relu(dot(input, W) + b)
```

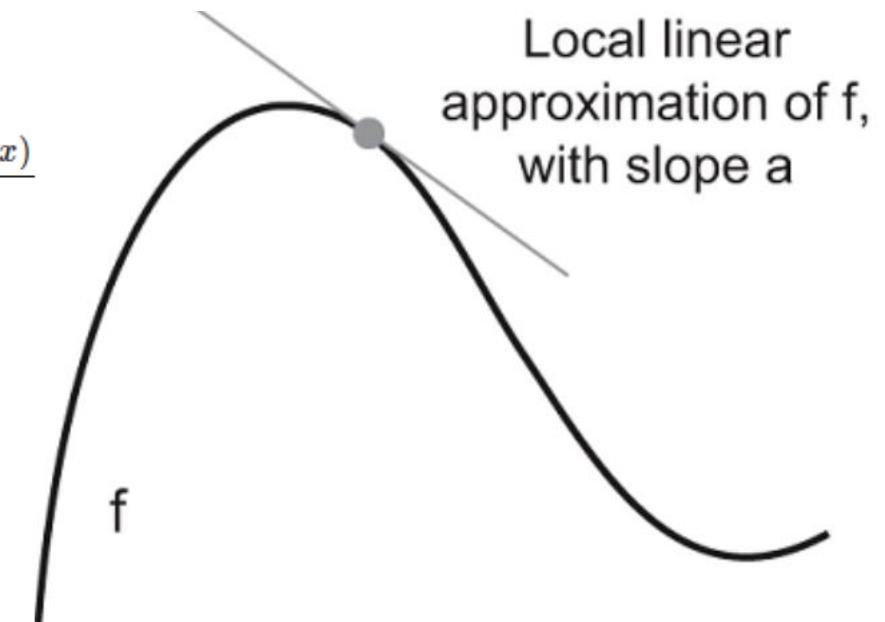
- **Random initialization** of weights W (+ biasvector b is initialized as 0)
- How to adjust parameters W and b to improve the model?
- **Training:** gradually adjust these parameters based on a feedback signal
- Repeat these steps, until the loss seems sufficiently low (= **training loop**):
 1. Draw a batch of training samples x and corresponding targets y_{true} .
 2. Run the model on x (a step called the *forward pass*) to obtain predictions y_{pred} .
 3. Compute the loss of the model on the batch, a measure of the mismatch between y_{pred} and y_{true} .
 4. Update all weights of the model in a way that slightly reduces the loss on this batch.

→ **Gradient Descent**

Derivative and gradient



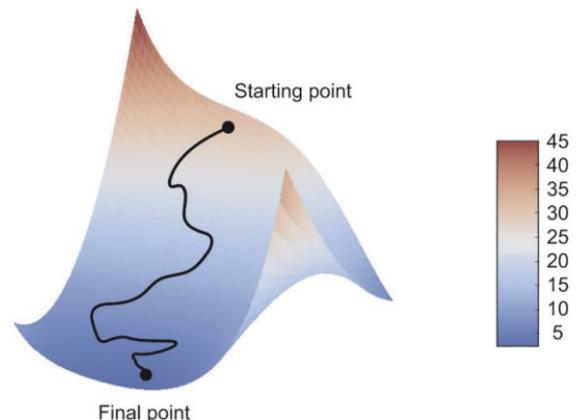
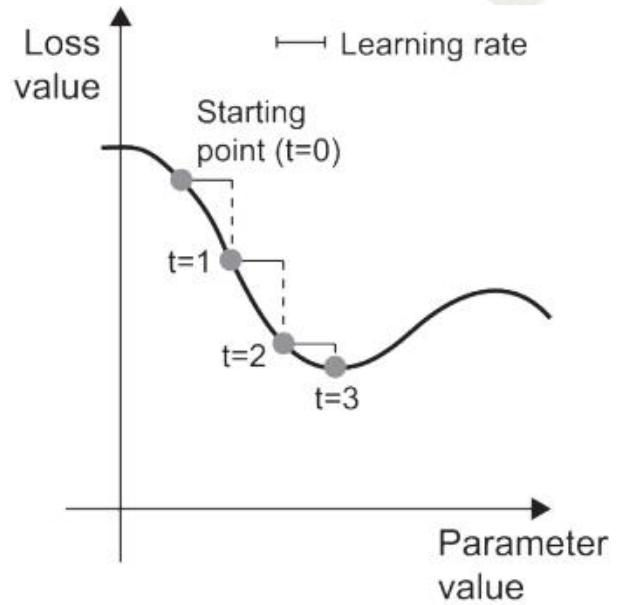
$$a = \frac{f(x+\epsilon) - f(x)}{\epsilon}$$



Stochastic Gradient Descent (SGD)

1. Draw a batch of training samples x and corresponding targets y_{true} .
2. Run the model on x to obtain predictions y_{pred} (this is called the *forward pass*).
3. Compute the loss of the model on the batch, a measure of the mismatch between y_{pred} and y_{true} .
4. Compute the gradient of the loss with regard to the model's parameters (this is called the *backward pass*).
5. Move the parameters a little in the opposite direction from the gradient — for example $w \leftarrow \text{learning_rate} * \text{gradient}$ — thus reducing the loss on the batch a bit. The *learning rate* (`learning_rate` here) would be a scalar factor modulating the “speed” of the gradient descent process.

- **True SGD:** `batchsize == 1`
- **Mini-batch SGD:** $1 < \text{batchsize} < \text{nsamples}$
- **(Full) Batch SGD:** `batchsize == nsamples`

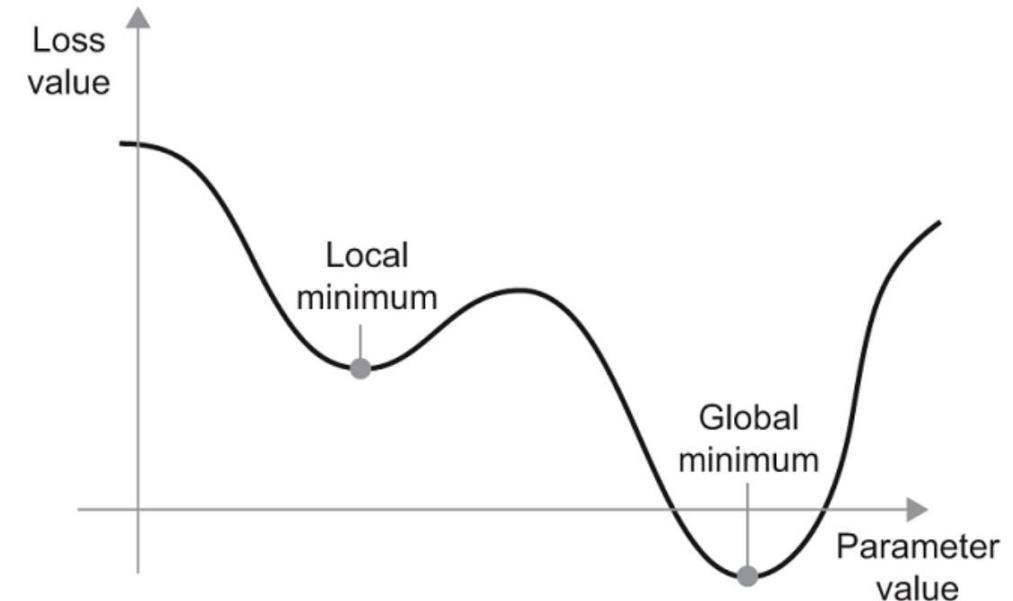


Stochastic Gradient Descent: variants

- Problem: local minima!
- Solution: **Momentum**

```
past_velocity = 0.  
momentum = 0.1  
while loss > 0.01:  
    w, loss, gradient = get_current_parameters()  
    velocity = past_velocity * momentum - learning_rate * gradient  
    w = w + momentum * velocity - learning_rate * gradient  
    past_velocity = velocity  
    update_parameter(w)
```

① ②



- E.g., RMSProp, AdaGrad, Adam, ...

Chain rule

- **Loss function** = composite function

```
loss_value = loss(y_true, softmax(dot(relu(dot(inputs, w1) + b1), w2) + b2))
```

- Gradient of loss function needed to update parameters

```
w1 = w0 - alpha * grad(loss_value, w0)
```

```
b1 = b0 - alpha * grad(loss_value, b0)
```

- **Chain rule:** gradient of composite function = product of gradients

```
def fghj(x):  
    x1 = j(x)  
    x2 = h(x1)  
    x3 = g(x2)  
    y = f(x3)  
    return y  
  
grad(y, x) == grad(y, x3) * grad(x3, x2) * grad(x2, x1) * grad(x1, x)
```

- **Backpropagation** algorithm = applying the chain rule to calculate gradient of loss function

TensorFlow: automatic differentiation

THE GRADIENT TAPE IN TENSORFLOW

The API through which you can leverage TensorFlow’s powerful automatic differentiation capabilities is the `GradientTape`. It’s a Python scope that will “record” the tensor operations that run inside it, in the form of a computation graph (sometimes called a “tape”). This graph can then be used to retrieve the gradient of any output with respect to any variable or set of variables (instances of the `tf.Variable` class). A `tf.Variable` is a specific kind of tensor meant to hold mutable state — for instance, the weights of a neural network are always `tf.Variable` instances.

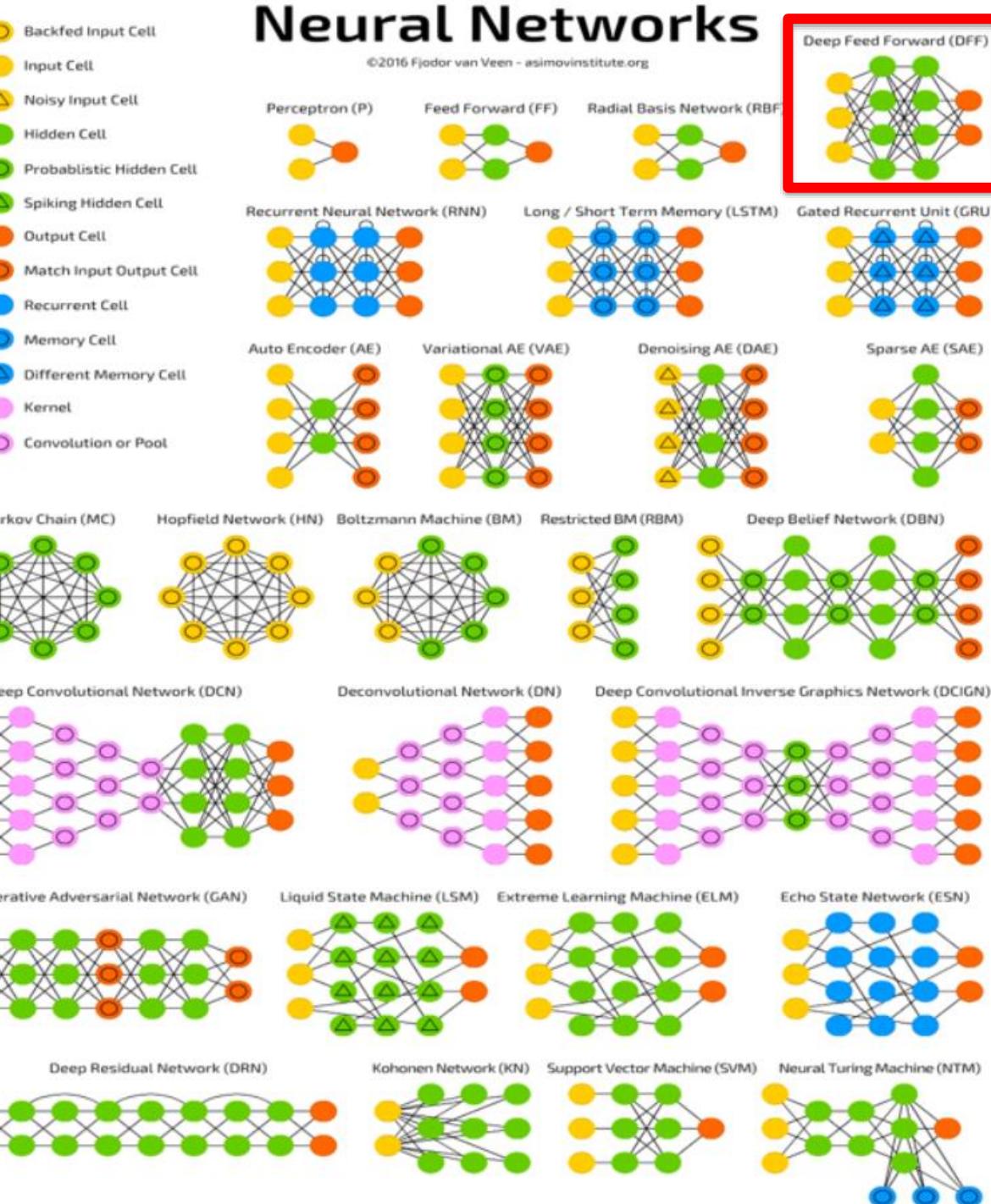
```
import tensorflow as tf  
x = tf.Variable(0.)  
with tf.GradientTape() as tape:  
    ①  
    y = 2 * x + 3  
    ②  
    grad_of_y_wrt_x = tape.gradient(y, x)  
    ③  
    ④
```

Introduction to Deep Learning

NETWORK TYPES

Network architectures

- So far we have only discussed
(Deep) Feed Forward Networks (FFN)
- But there are many other network types:
 - Convolutional Neural Networks (CNN)
 - Recurrent Neural Networks (RNN)
 - Long Short-Term Memory Networks (LSTM)
 - Transformers
 - Variational Autoencoders (VAE)
 - Generative Adversarial Networks (GAN)
 - ...



Convolution Neural Networks

- Most often used for **computer vision** tasks:
 - Image classification
 - Image segmentation
 - Object detection
 - ...
- Consist of **convolutional layers** that:
 - apply **filters** to capture features such as edges, colors, textures, patterns, ...
(think of Instagram filters)
 - are usually combined with **pooling layers** to downsample the feature maps

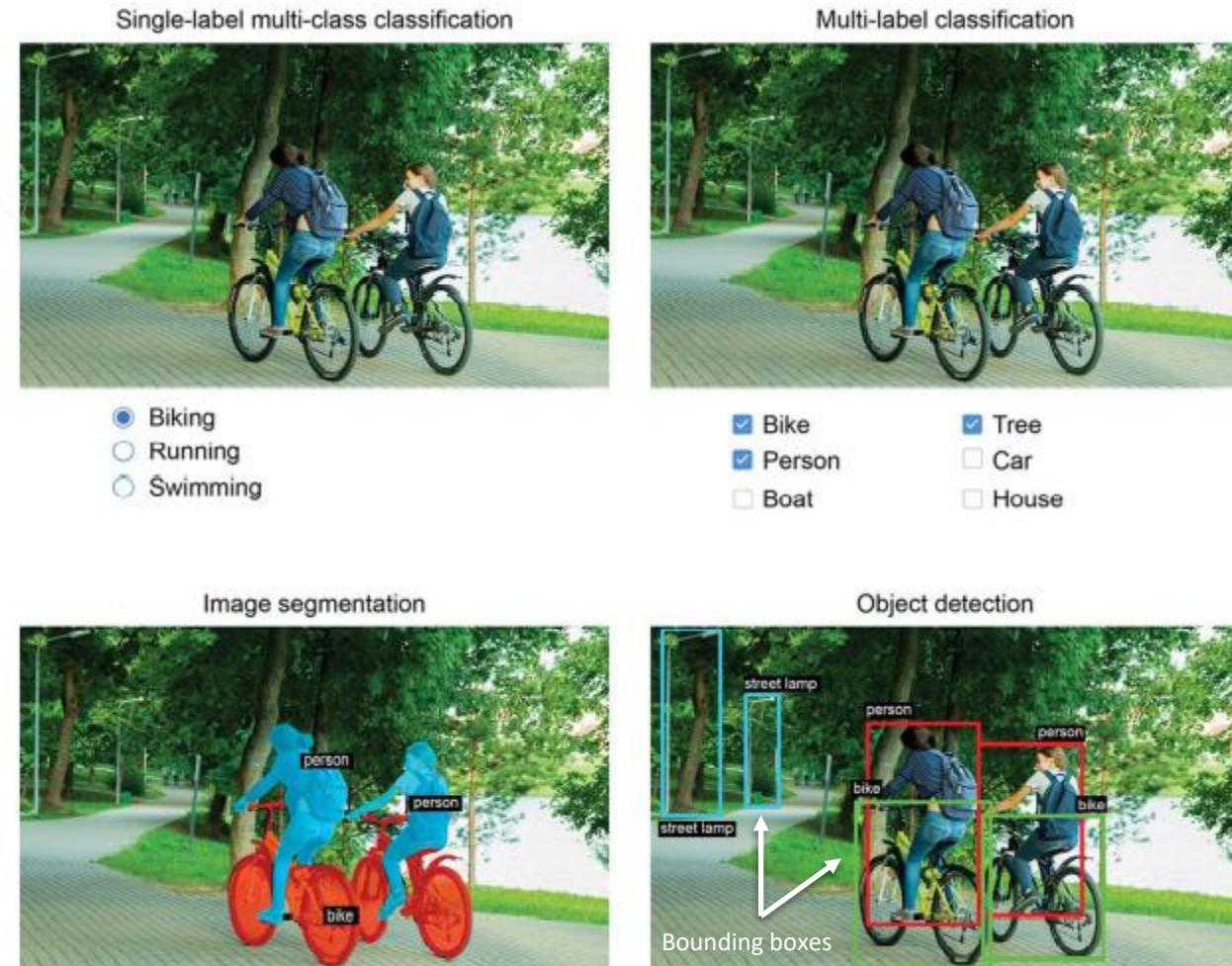
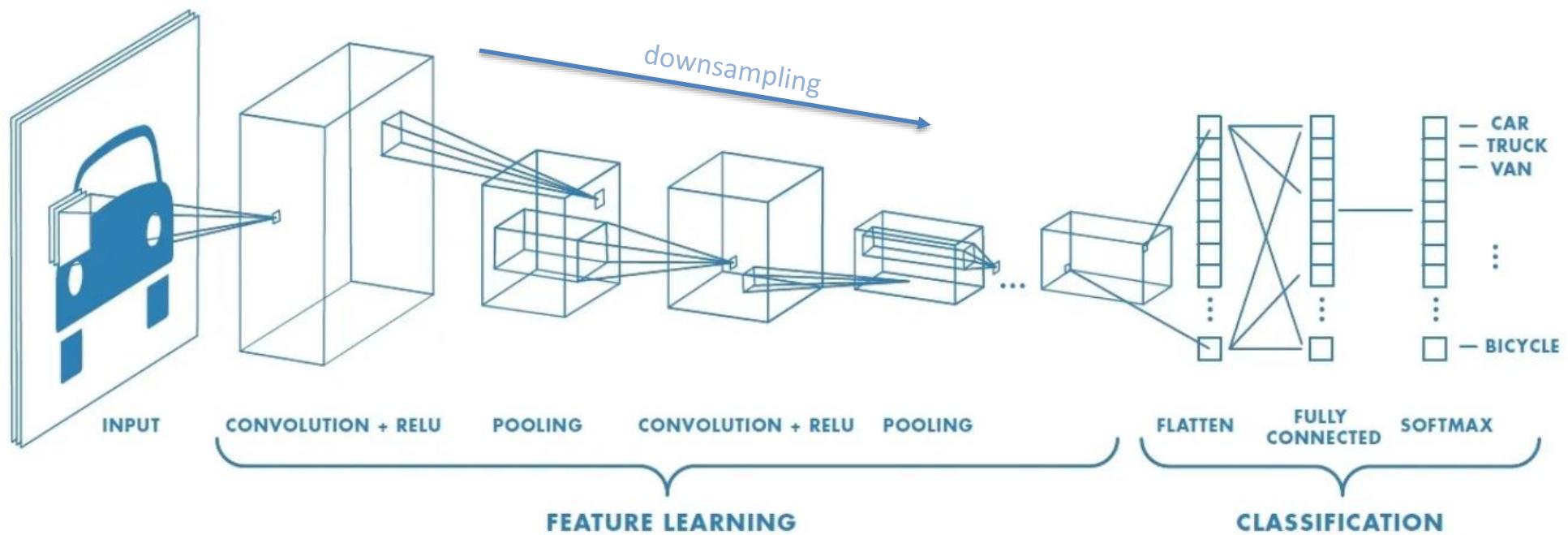


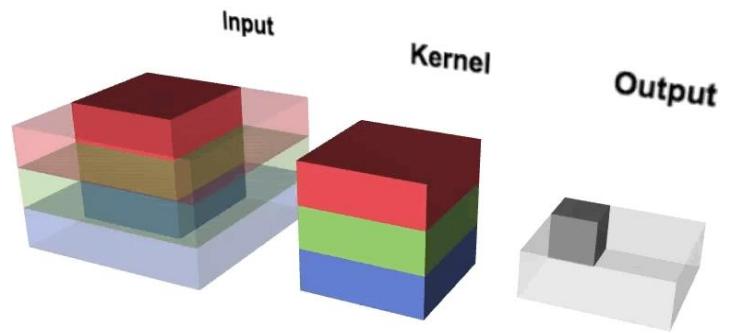
Image classification

- Convnet = convolution base + classifier
- The **convolutional base** learns the features and consists of convolutional layers
- The **classifier** does the classification and consists of fully connected (dense) layers

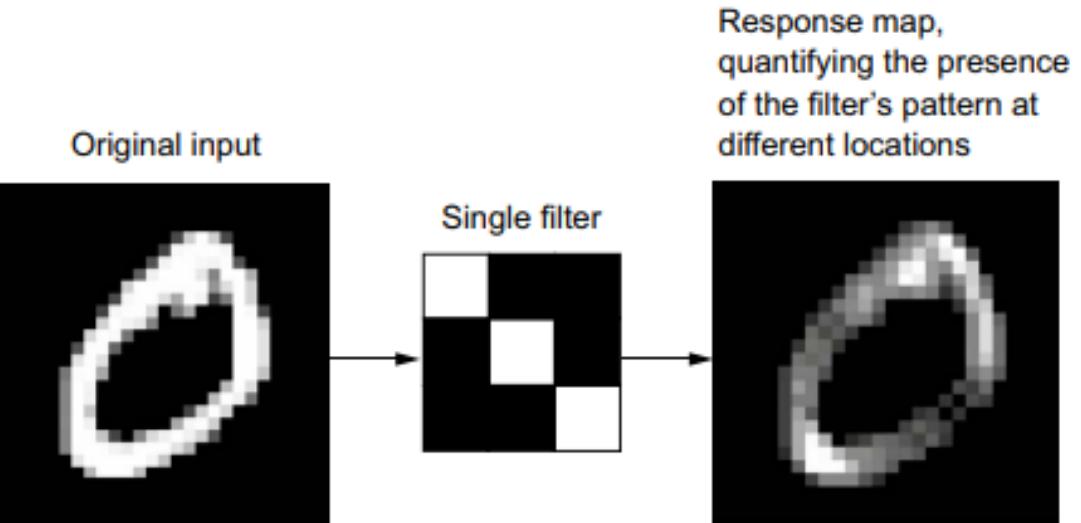
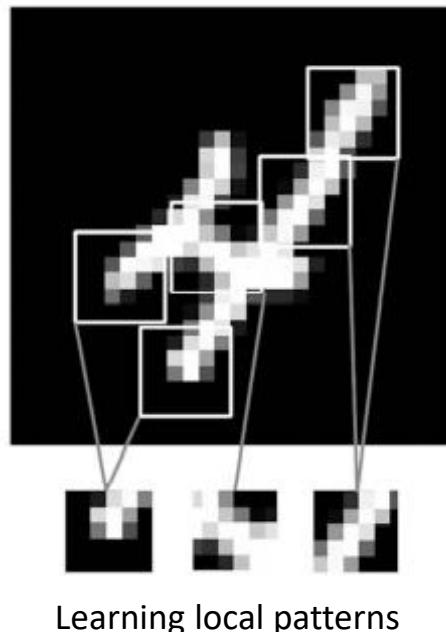


How convolution works

- A convolutional layer is a **feature detector**
- It ‘slides’ filters (= convolutional **kernels**) over the input image
- The result of ‘sliding’ a filter is a response **feature map**
- In this way convolutional layers learn **local patterns** by optimizing the filters during training



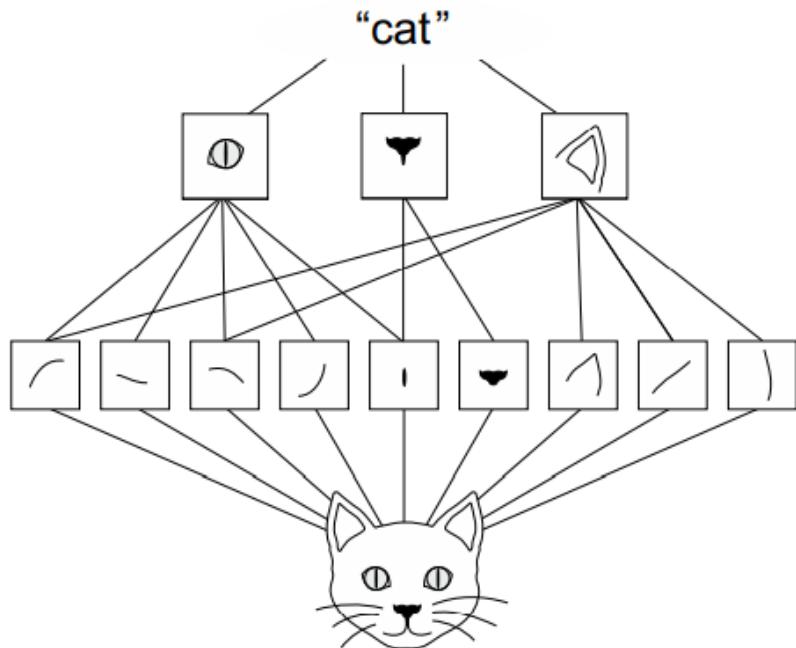
<https://medium.com/apache-mxnet/multi-channel-convolutions-explained-with-ms-excel-9bbf8eb77108>



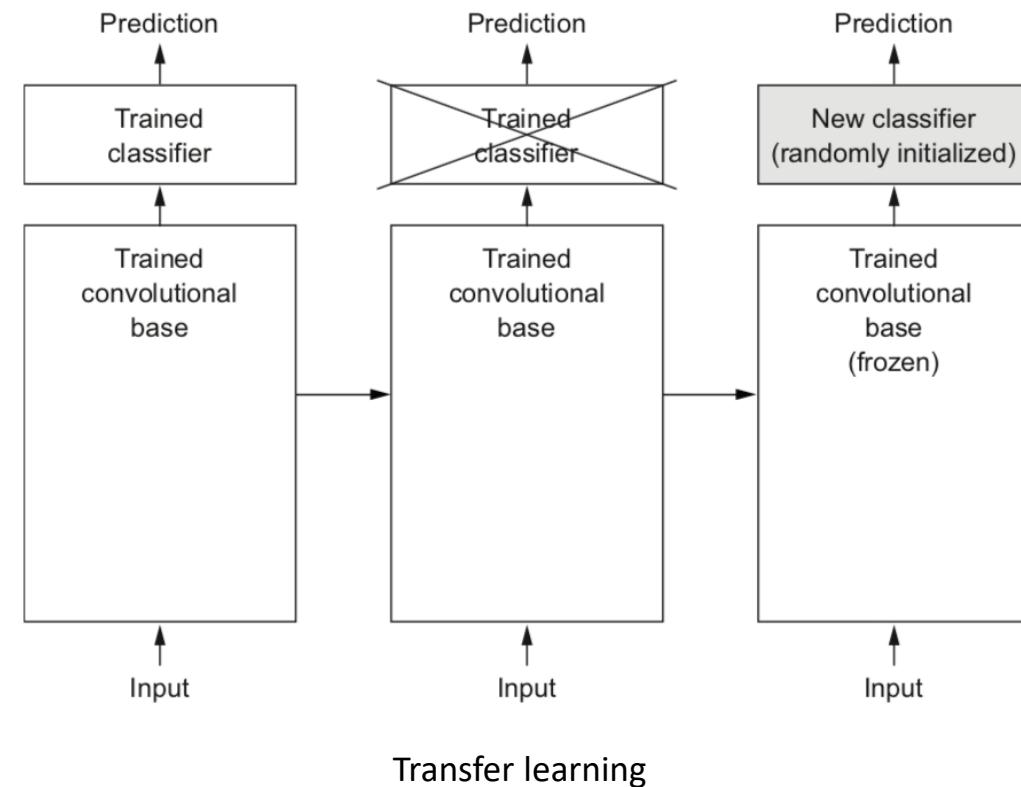
Response map,
quantifying the presence
of the filter’s pattern at
different locations

Interesting properties of CNNs

- The learned patterns are **translation-invariant**
(= the location of a pattern or object on an image is irrelevant)
- By downsampling the feature maps, **spatial hierarchies** of patterns are learned
- The convbase can be reused (= **transfer learning**)



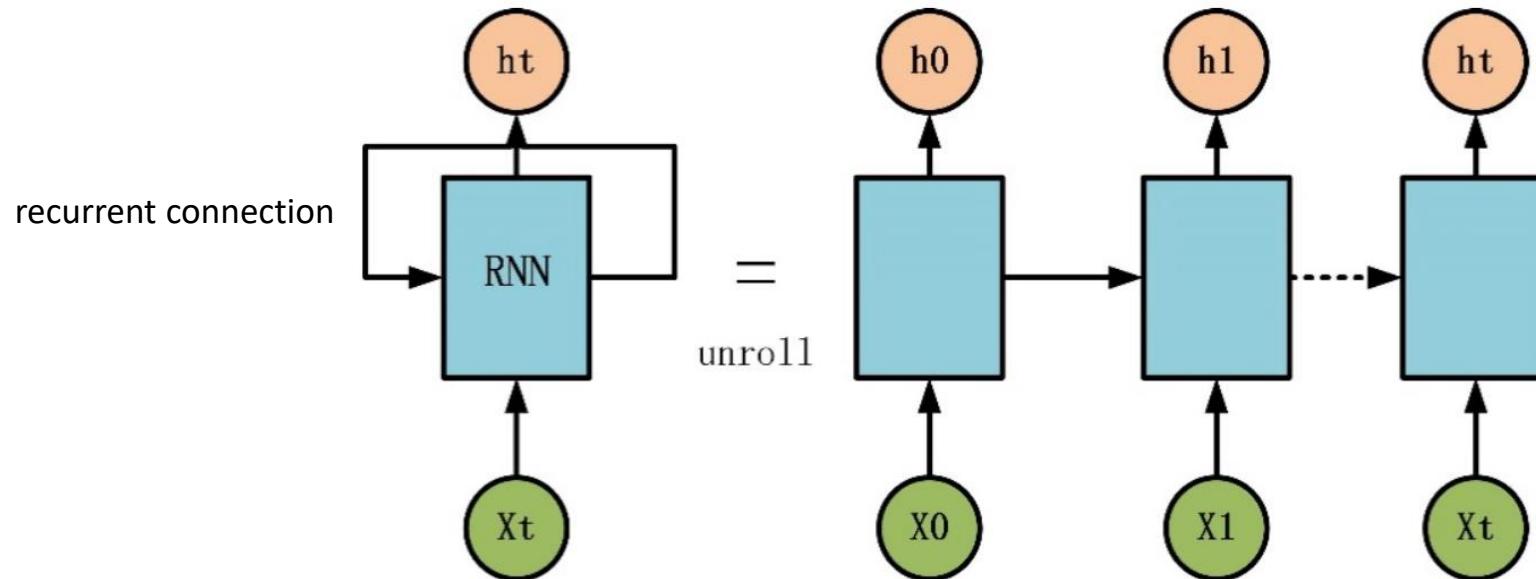
Learning spatial hierarchies



Transfer learning

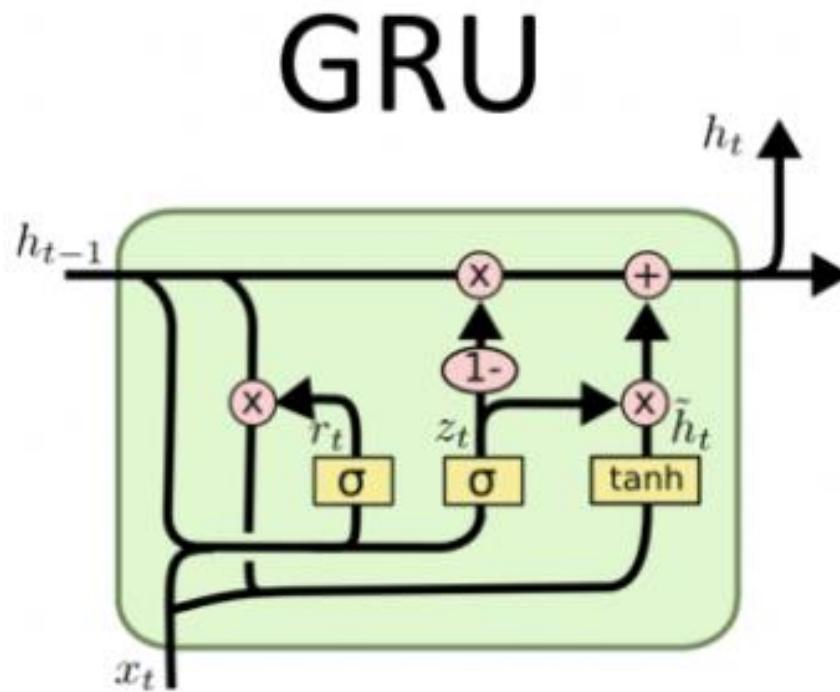
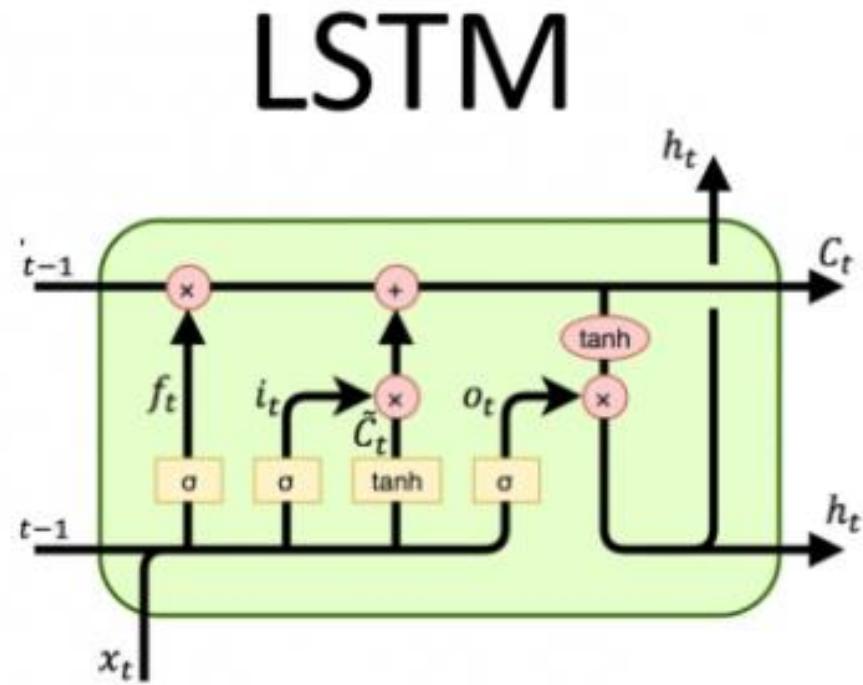
Recurrent Neural Networks

- RNNs are designed for **sequence data** (= data with strict order)
- RNNs allow information to be retained and processed over time.
- RNNs use recurrent connections to maintain a **memory** of previous inputs
- RNNs are suitable for tasks like **natural language processing** and **time series analysis**



Examples of RNNs

- LSTM = Long Short-Term Memory
- GRU = Gated Recurrent Unit

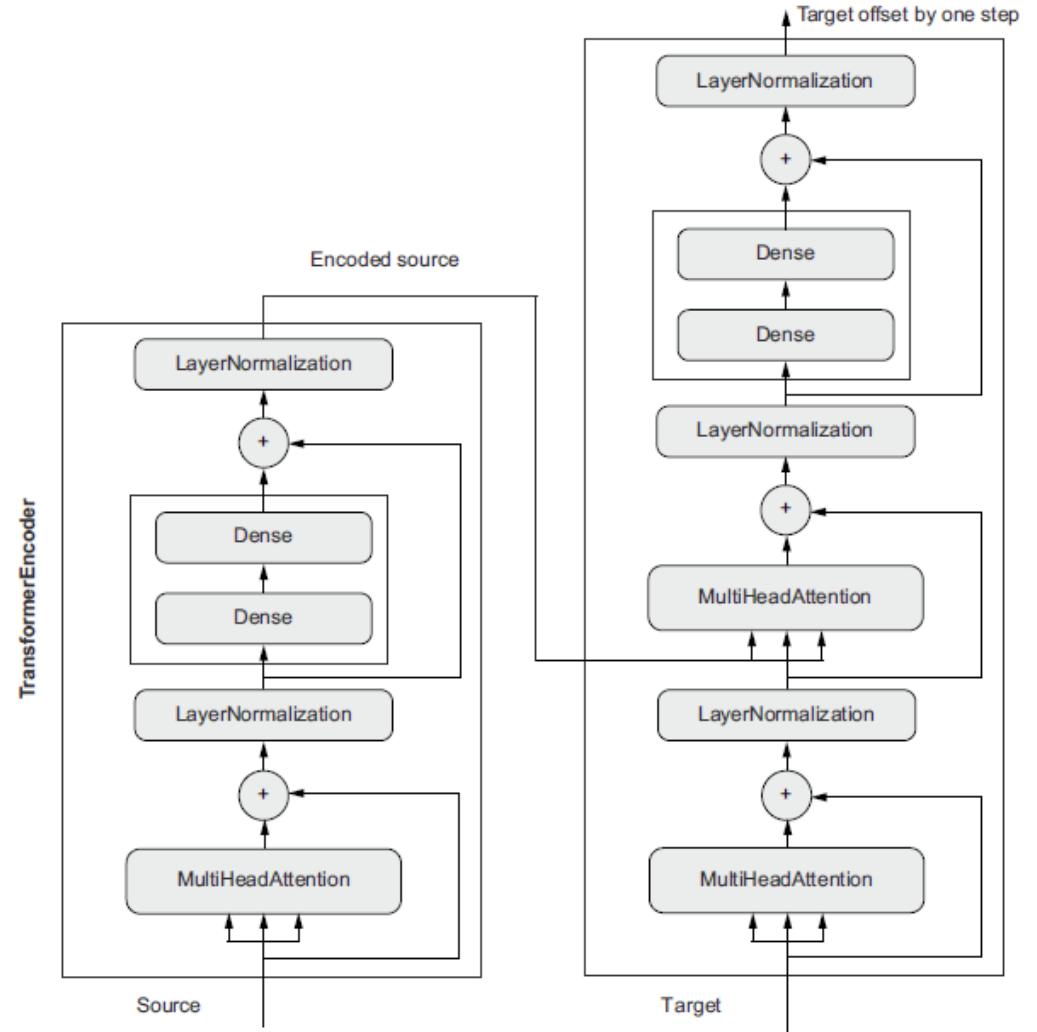


Transformers

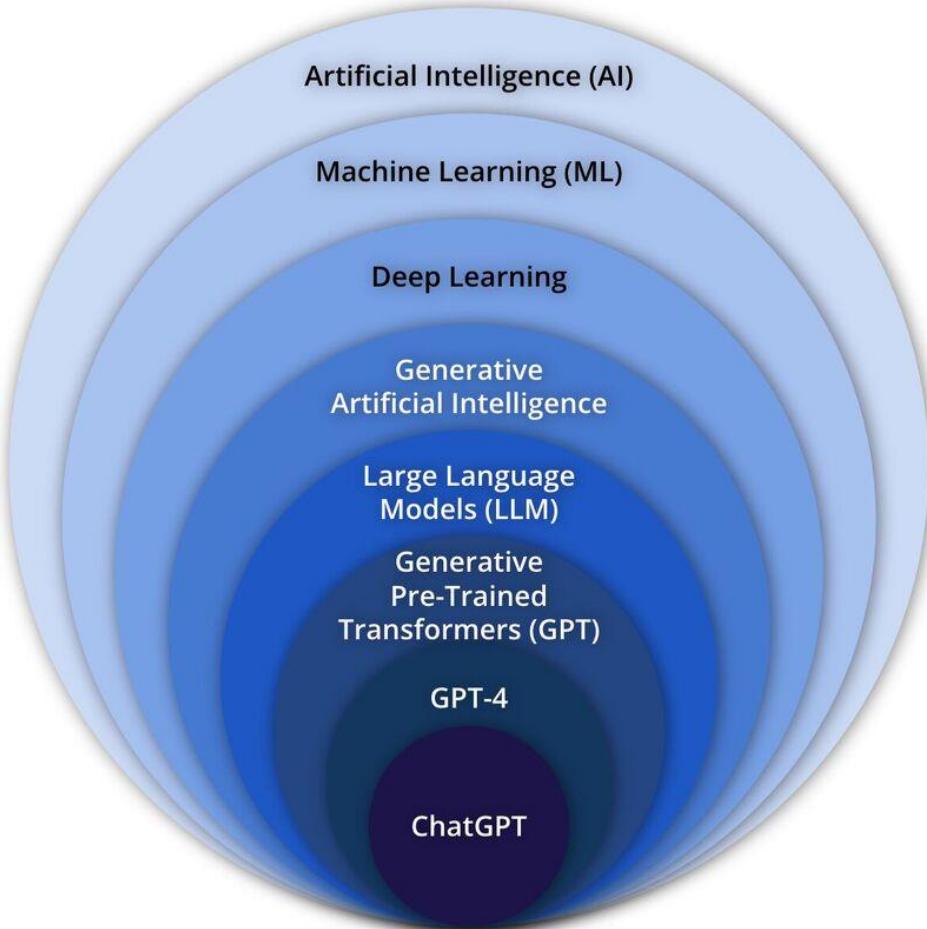
- Designed for **sequence data**
but process the entire input at once
hence, more suitable for parallel computing than RNNs
- Apply the mechanism of **self-attention**
+ consider order in the data, but not as strict as RNNs do
hence, more suitable for **natural language processing**
- Also used in computer vision
- Developed by Google Brain in 2017
- Examples: GPT, BERT, XLNet, RoBERTa, ...
- See also: [Hugging Face](#) Transformers library



Hugging Face



What about ChatGPT?



https://www.linkedin.com/posts/edoardoquercidellarovere_spesso-mi-capita-di-leggere-e-sentire-conversazioni-activity-7126095949738721280-amQI/

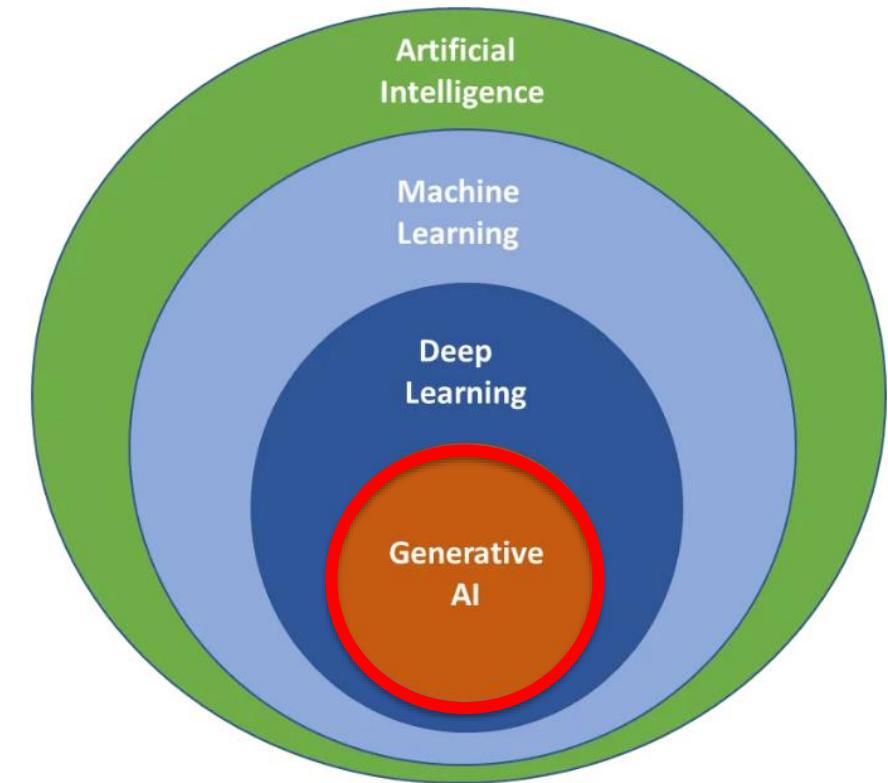
Introduction to Generative AI

Introduction to Generative AI

WHAT IS GENERATIVE AI?

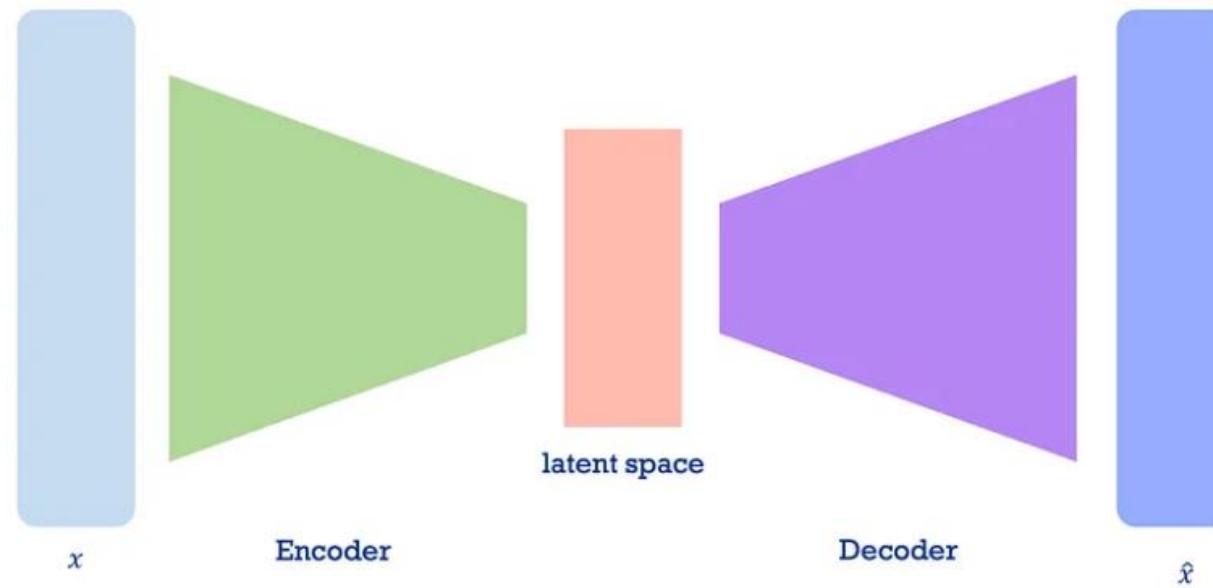
A subfield of Deep Learning

- **Artificial Intelligence (AI):**
The set of all tasks in which a computer can make decisions.
- **Machine Learning (ML):**
The set of all tasks in which a computer can make decisions based on data.
- **Deep Learning (DL):**
The field of machine learning that uses certain objects called neural networks.
- **Generative AI (GenAI):**
The field of deep learning that focuses on creating new content, such as text, images, or music, using neural networks to learn patterns from existing data.



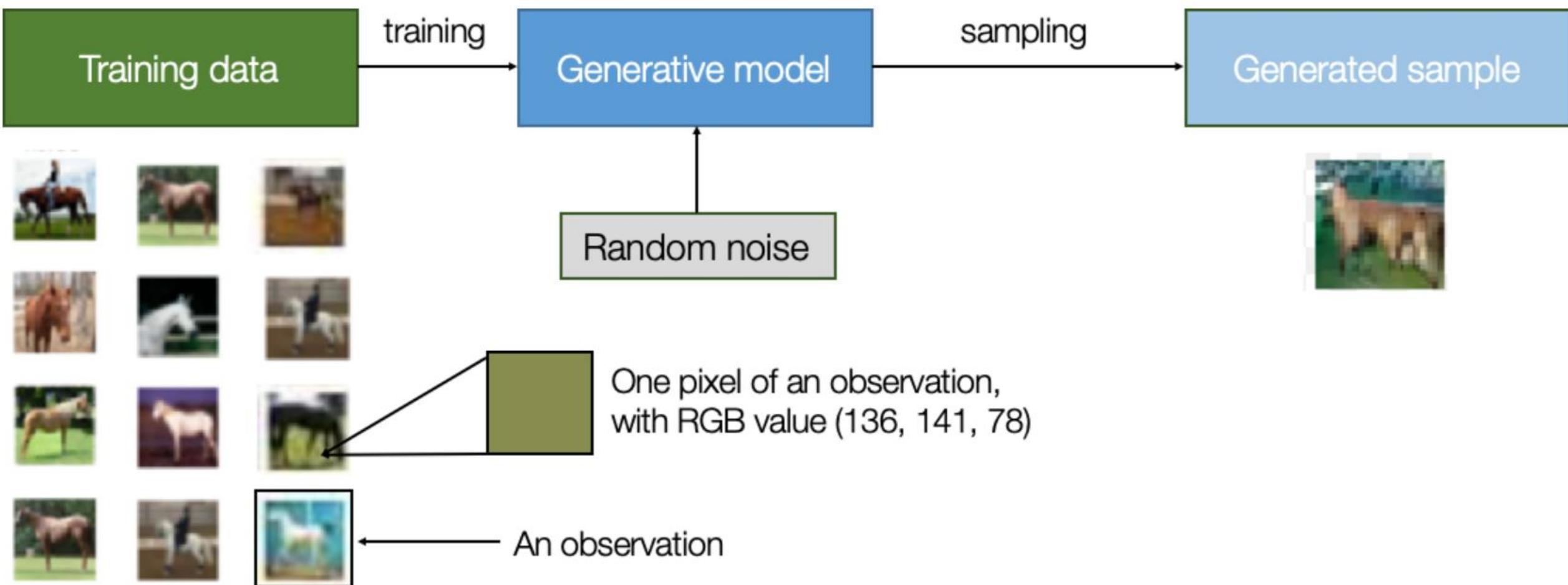
Generative Deep Learning

- our language and visualizations have **statistical structure**
- **machine learning algorithms** learn this structure via self-supervised learning
= the statistical **latent space** of images, music, text, speech, ...
- **machine learning models** generate new samples from this latent space
- e.g., **autoencoders** (AEs) (= first generative deep learning models)



<https://towardsdatascience.com/generative-networks-from-ae-to-vae-to-gan-to-cyclegan-b21ba99ab8d6>

Training vs Sampling

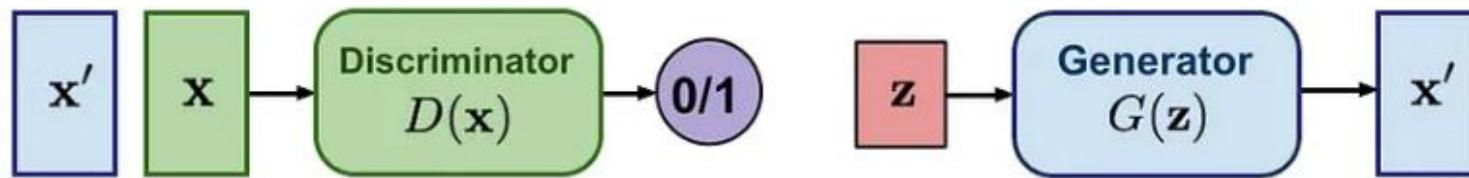


Introduction to Generative AI

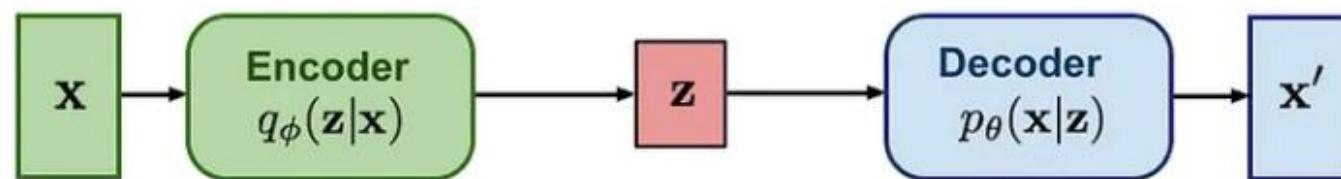
IMAGE GENERATION

The big 3 of image generation

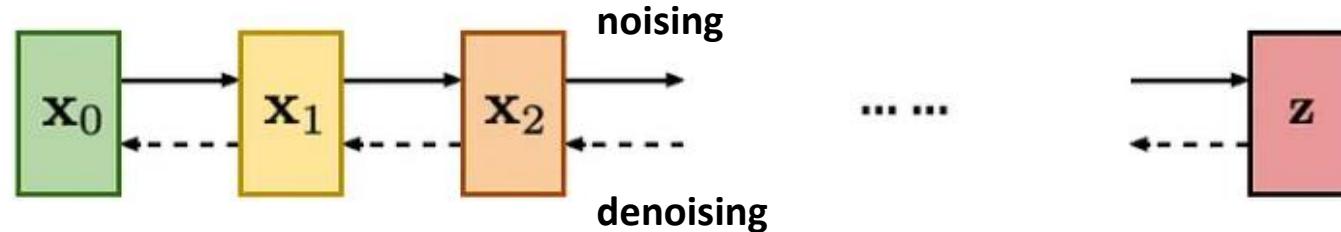
GAN: Adversarial training



VAE: maximize variational lower bound



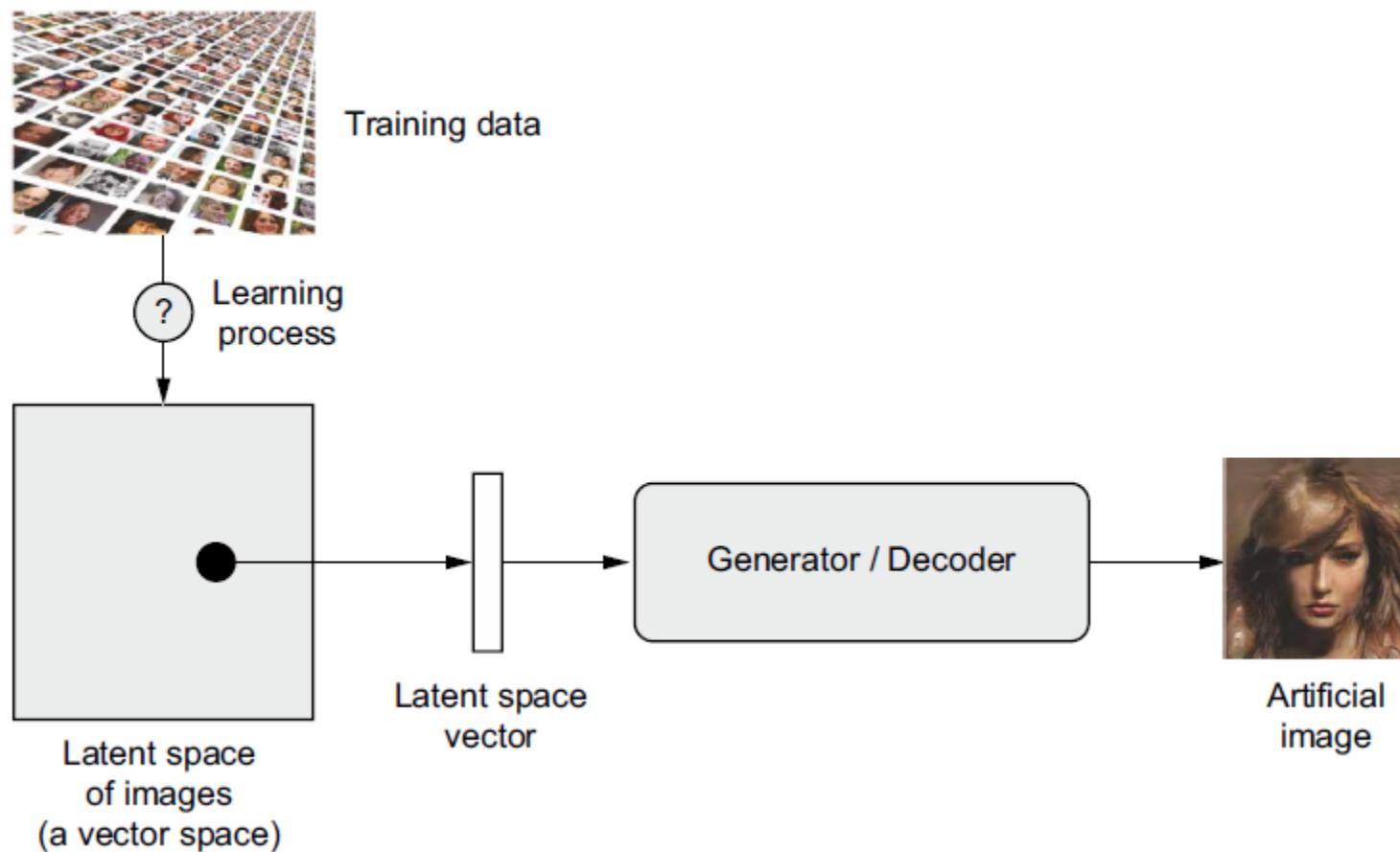
Diffusion models:
Gradually add Gaussian noise and then reverse



<https://pub.towardsai.net/diffusion-models-vs-gans-vs-vaes-comparison-of-deep-generative-models-67ab93e0d9ae>

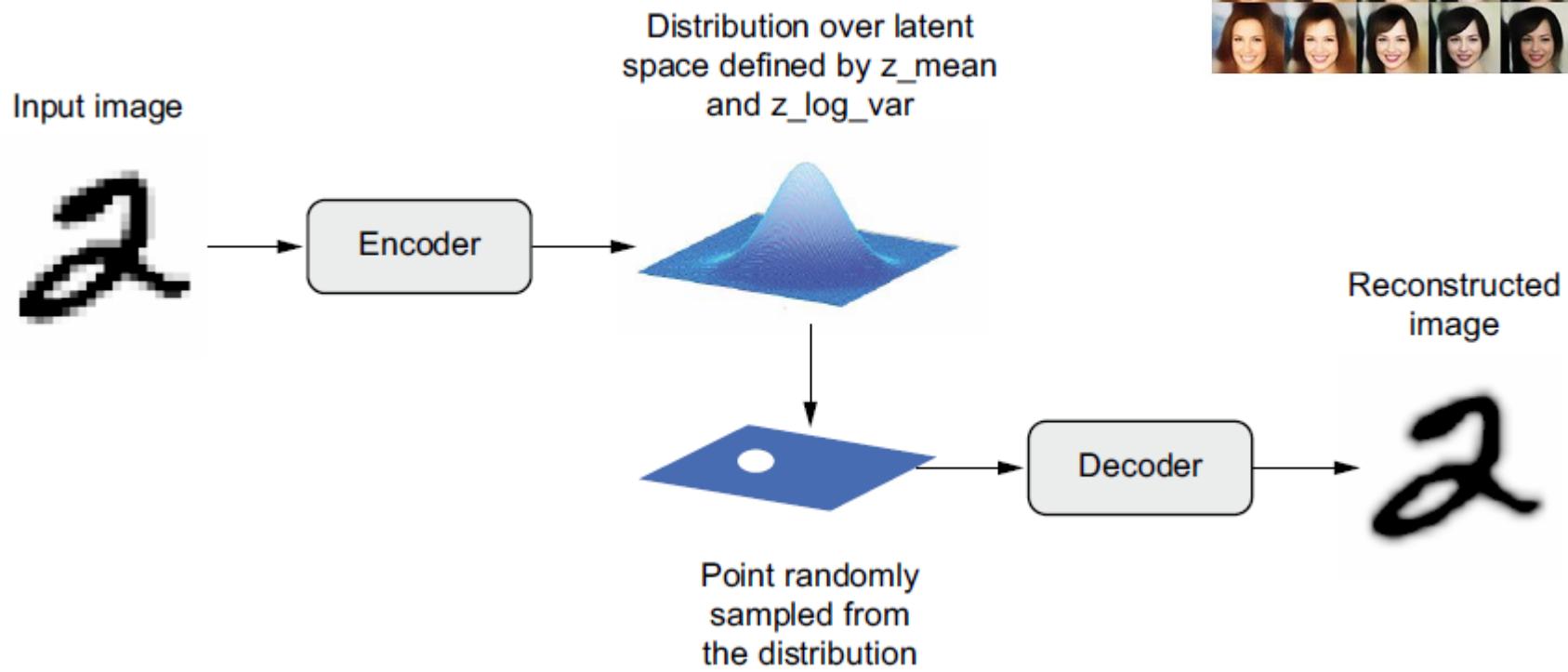
Latent Space Sampling

via decoder (VAE) or generator (GAN) or denoising (Diffusion model)

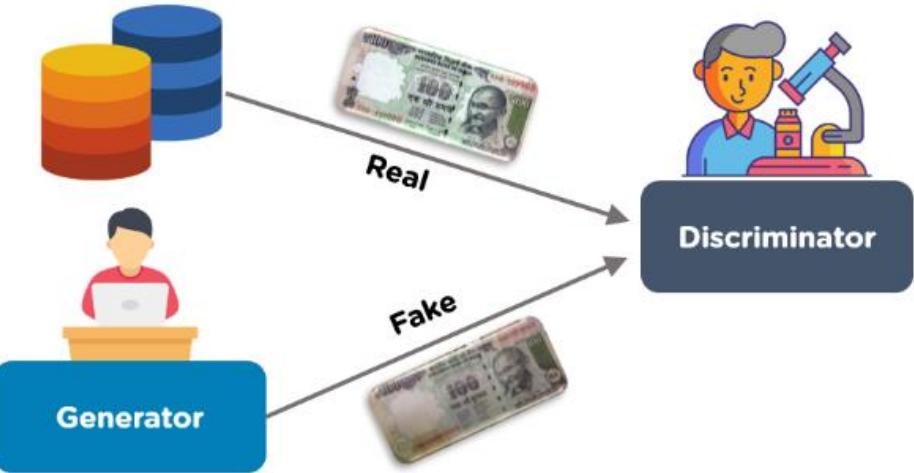
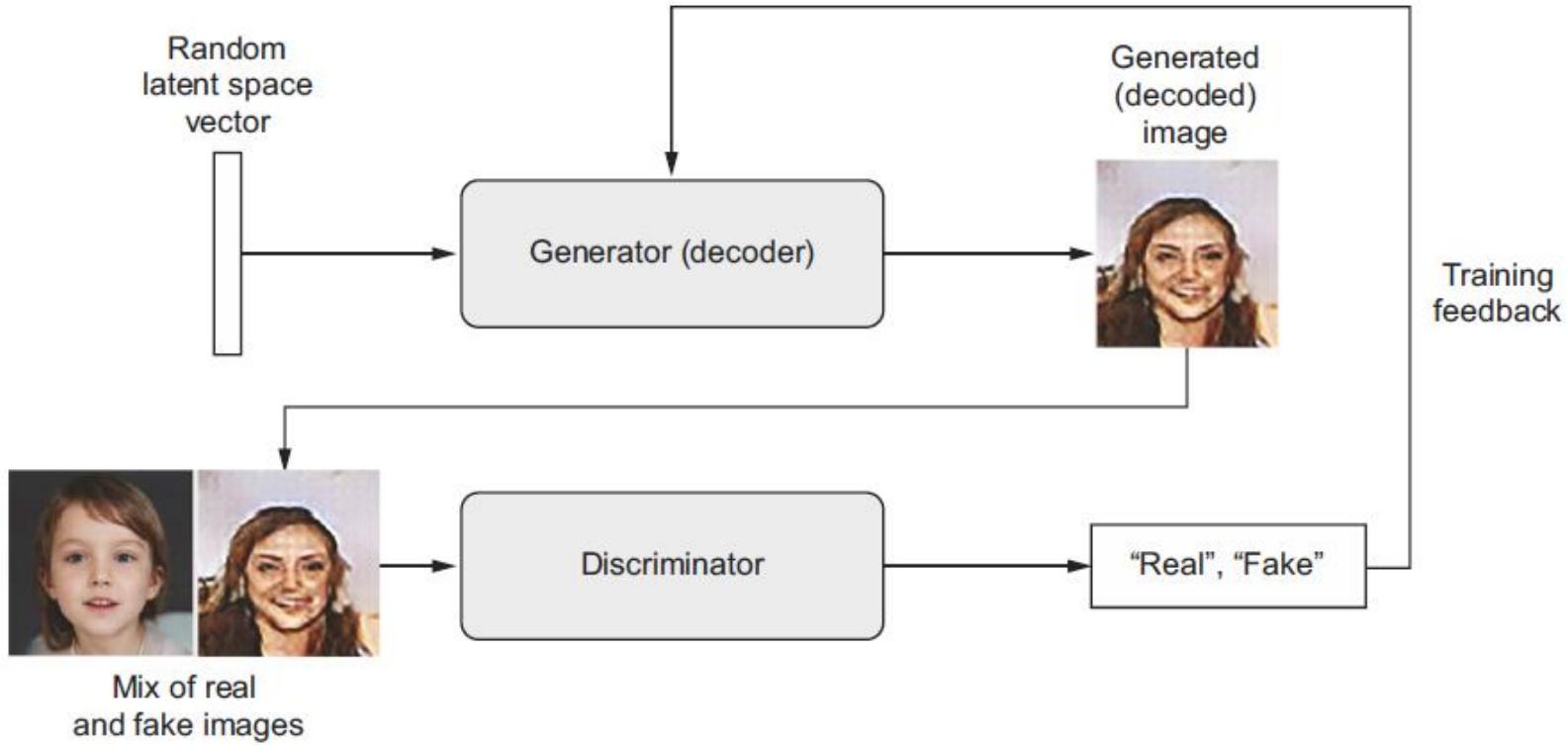


Variational Autoencoder

= deep learning + Bayesian inference
= autoencoder + “statistical magic”



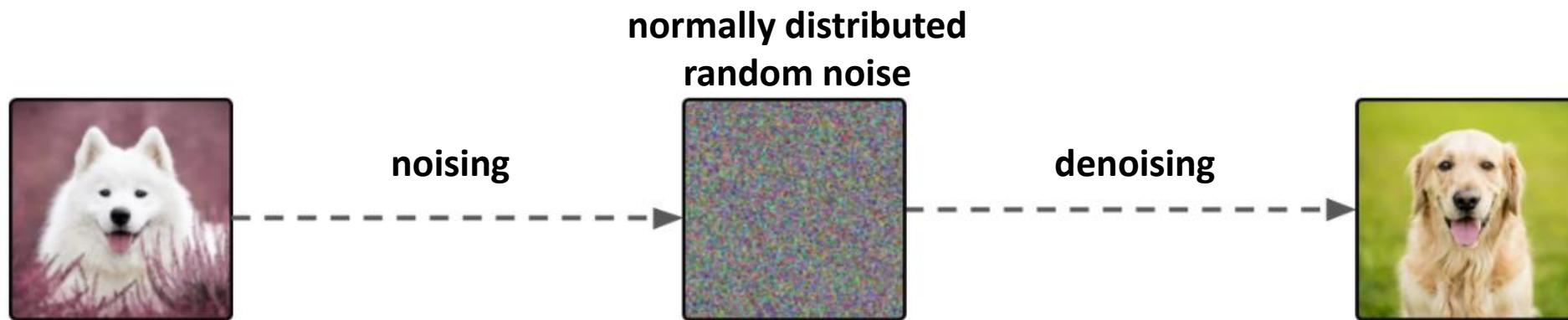
Generative Adversarial Network



<https://www.simplilearn.com/tutorials/deep-learning-tutorial/generative-adversarial-networks-gans>

Diffusion models

- Inspired by thermodynamics
- State-of-the-art image generation models (e.g. DALLE)

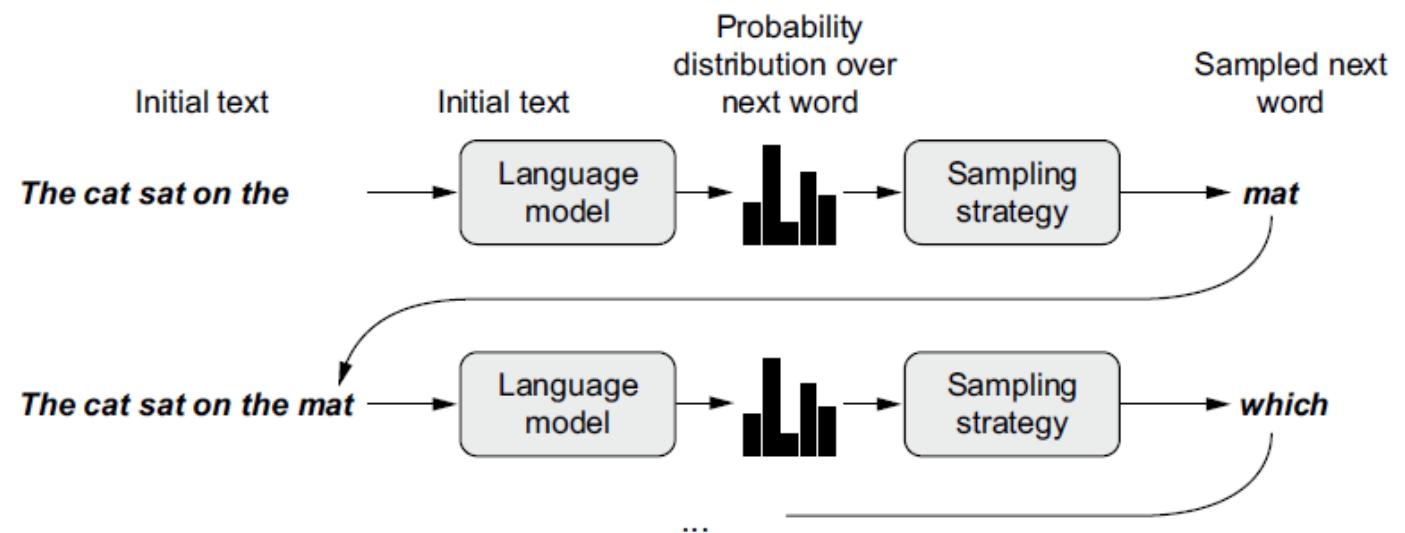


Introduction to Generative AI

TEXT GENERATION

Generating text

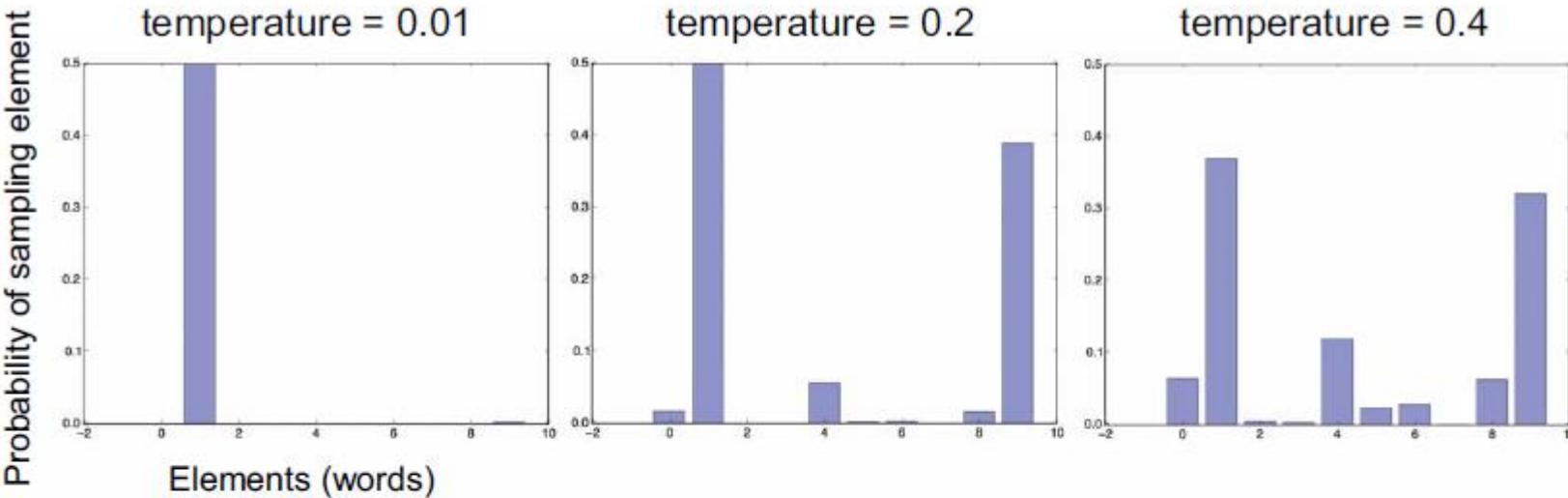
1. Train a model (RNN or Transformer) to predict next token(s)
= learning the *latent space* of language
= **language model**
2. Sample the model: give text string (= *conditioning data*)
and let the model predict the next token
3. Repeat 2 with a new string



The importance of the sampling strategy

- **Greedy sampling:** always take the most probable token
 - Doesn't give good results...
- **Stochastic sampling:** also take less probable tokens
 - $P(\text{token}) = 0.3$: token is sampled in 30% of the cases
 - Gives good results!
- BUT: we want to control the randomness
 - **temperature** hyperparameter!
 - Is it possible to set the temperature for ChatGPT?

\approx greedy sampling



\approx random sampling

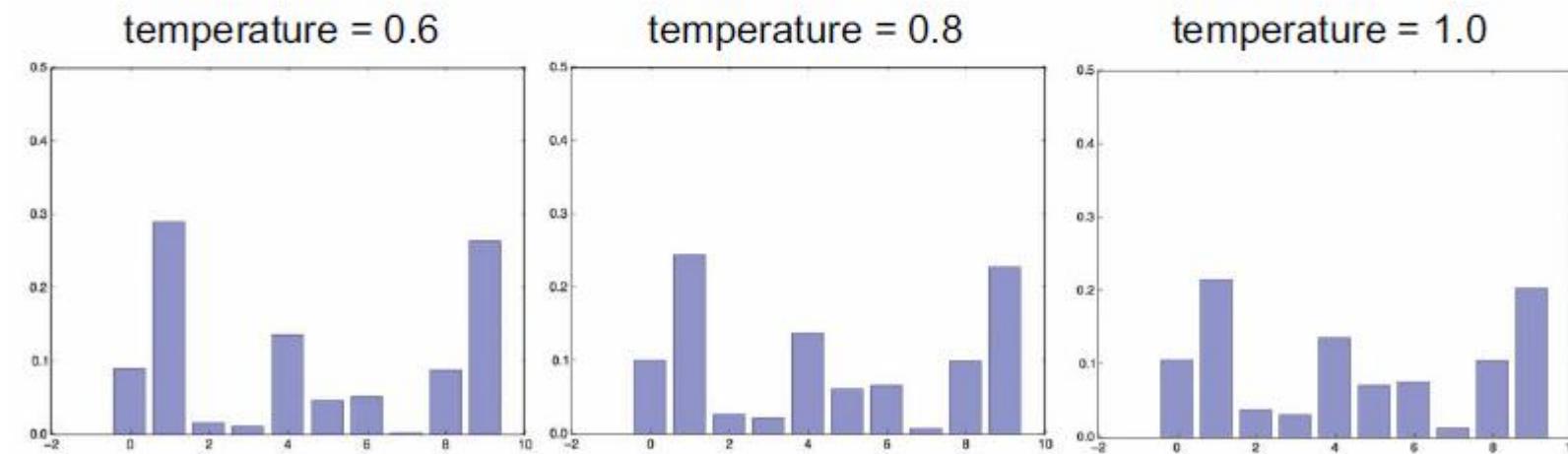
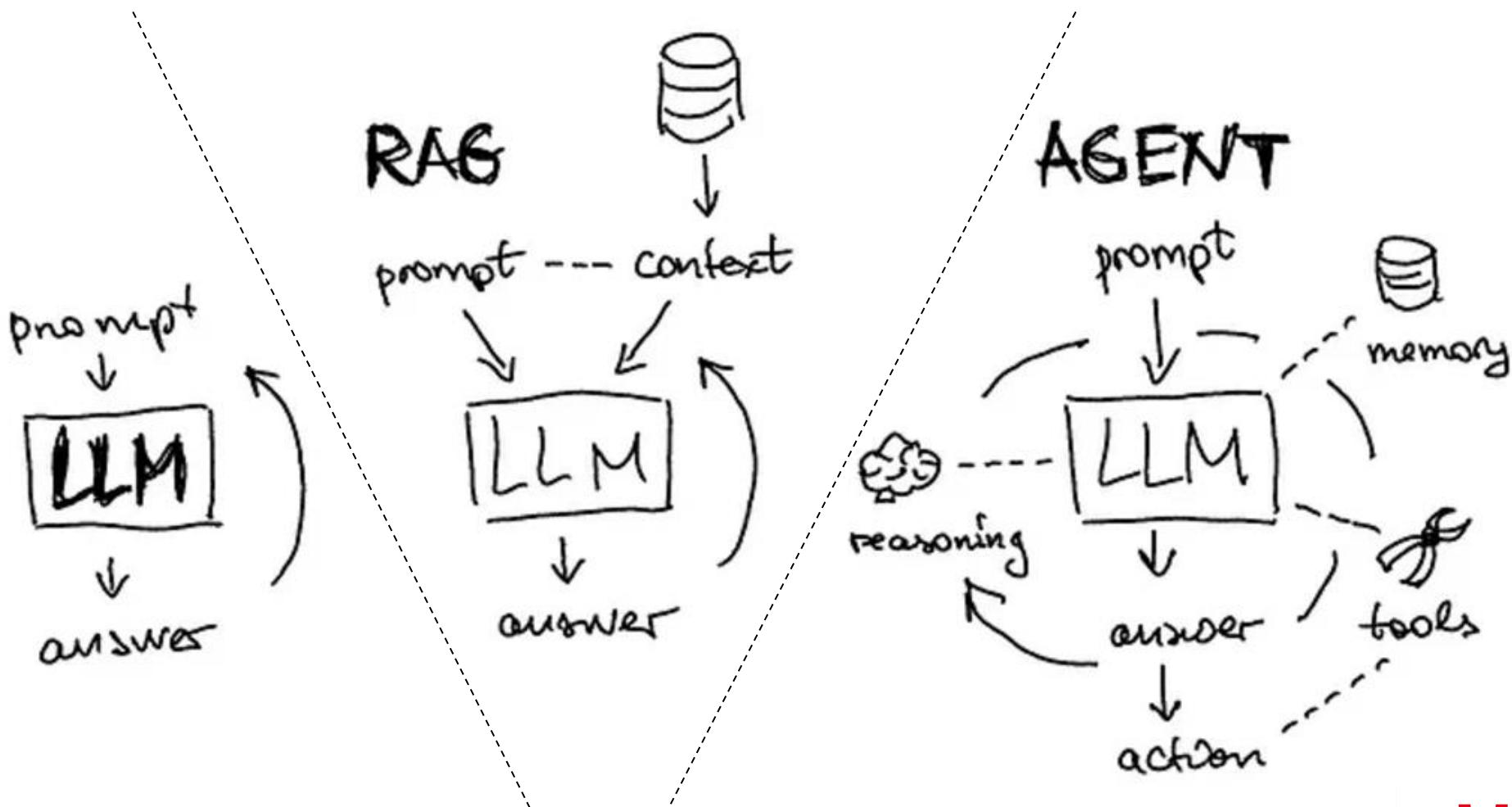


Figure 12.2 Different reweightings of one probability distribution. Low temperature = more deterministic, high temperature = more random.

Introduction to Generative AI

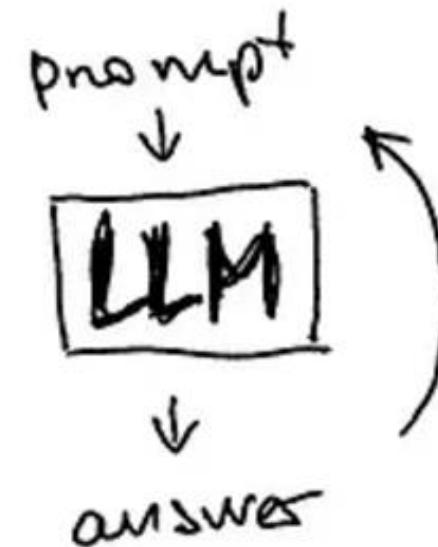
LARGE LANGUAGE MODELS, RETRIEVAL-AUGMENTED GENERATION, AND AI AGENTS

LLMs vs RAG vs Agents



How are Large Language Models used?

- 1. Define Your Task:** Clearly state what you want the LLM to do.
- 2. Craft Your Prompt:** Write a specific and clear prompt.
- 3. Get the Answer:** Submit your prompt to the LLM and receive the response.
- 4. Evaluate the Answer:** Assess the quality and relevance of the response.
- 5. Refine Your Prompt:** If the response isn't satisfactory, refine your prompt for better results.
- 6. Repeat as Needed:** Iterate through steps 3-5 until you get a satisfactory answer.



How to write a good prompt?

required



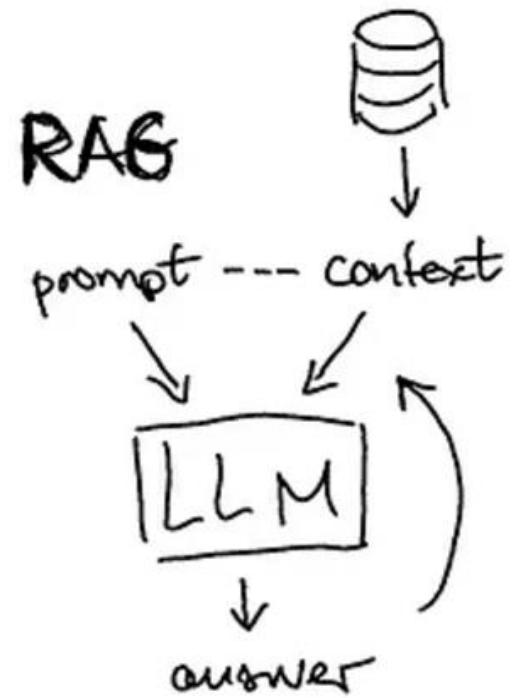
optional

Example:

You are a recruitment manager, and you have just found a candidate to fill a vacancy. Write an email to your direct supervisor, Frank C, to share this positive news. Provide a brief description of the candidate using bullet points. Include, for example, the candidate's education and work experience. Use clear and concise language and write in a formal but friendly tone.

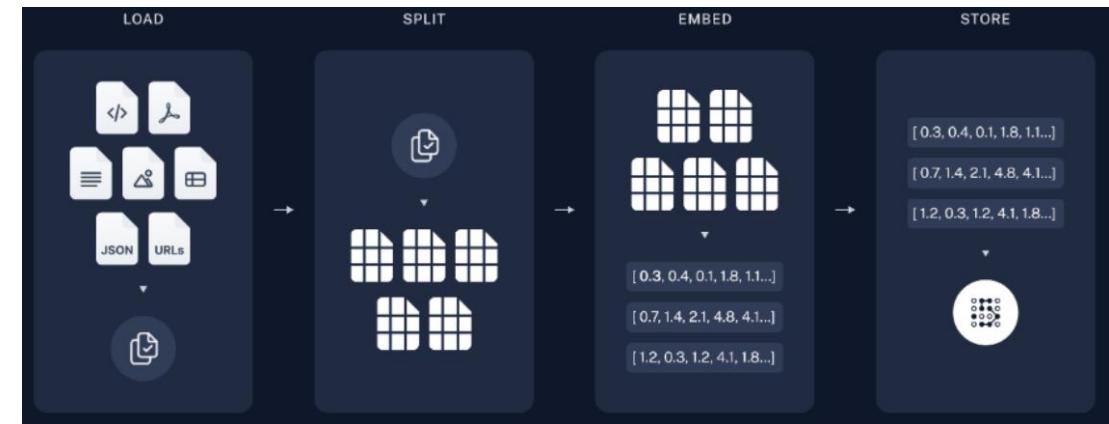
What is Retrieval-Augmented Generation?

- **Definition:** Retrieval-Augmented Generation (RAG) is a technique that combines information retrieval with generative models to enhance the accuracy and relevance of generated content
- **Purpose:** It aims to optimize the output of large language models (LLMs) by referencing external knowledge bases
- **Applications:** chatbots, content creation, ...

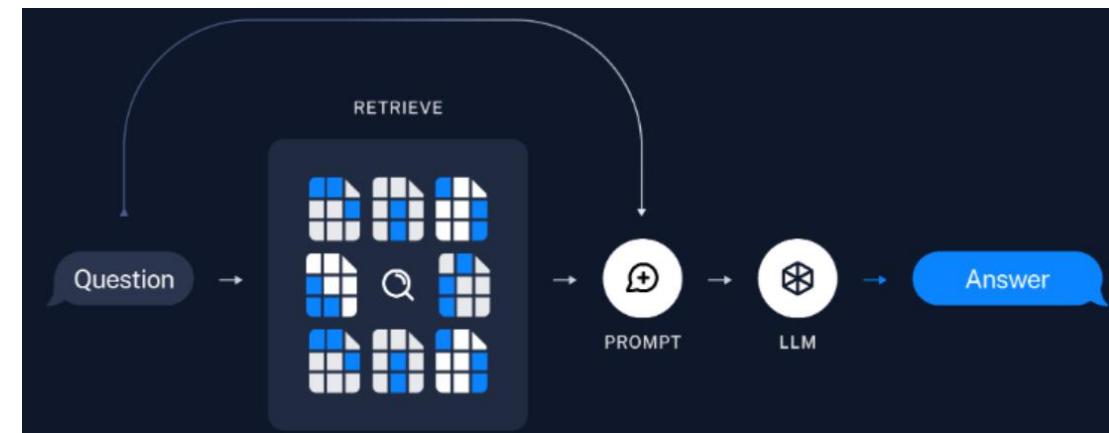


Building a RAG app

- **Step 1: Indexing**
 - Load documents
 - Split documents into smaller chunks
 - Embed the splits using an embeddings model
 - Store the resulting vectors in a database

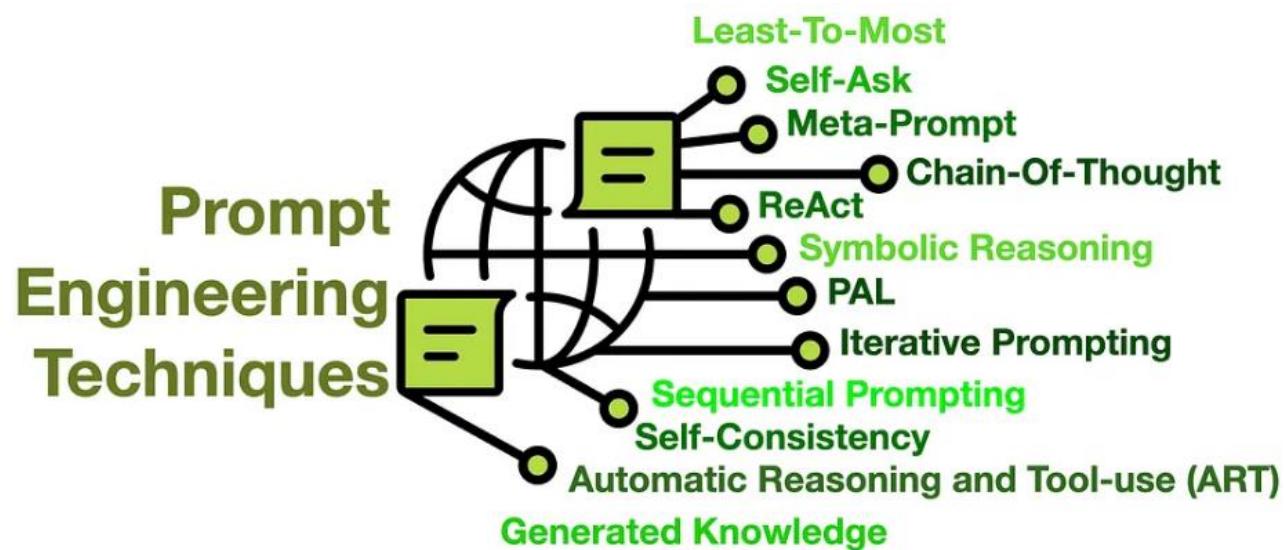


- **Step 2: Retrieval and generation**
 - Retrieve the splits that are most similar to the user's question
 - Generate an answer using an LLM based on the users's question and the retrieved splits



What is prompt engineering?

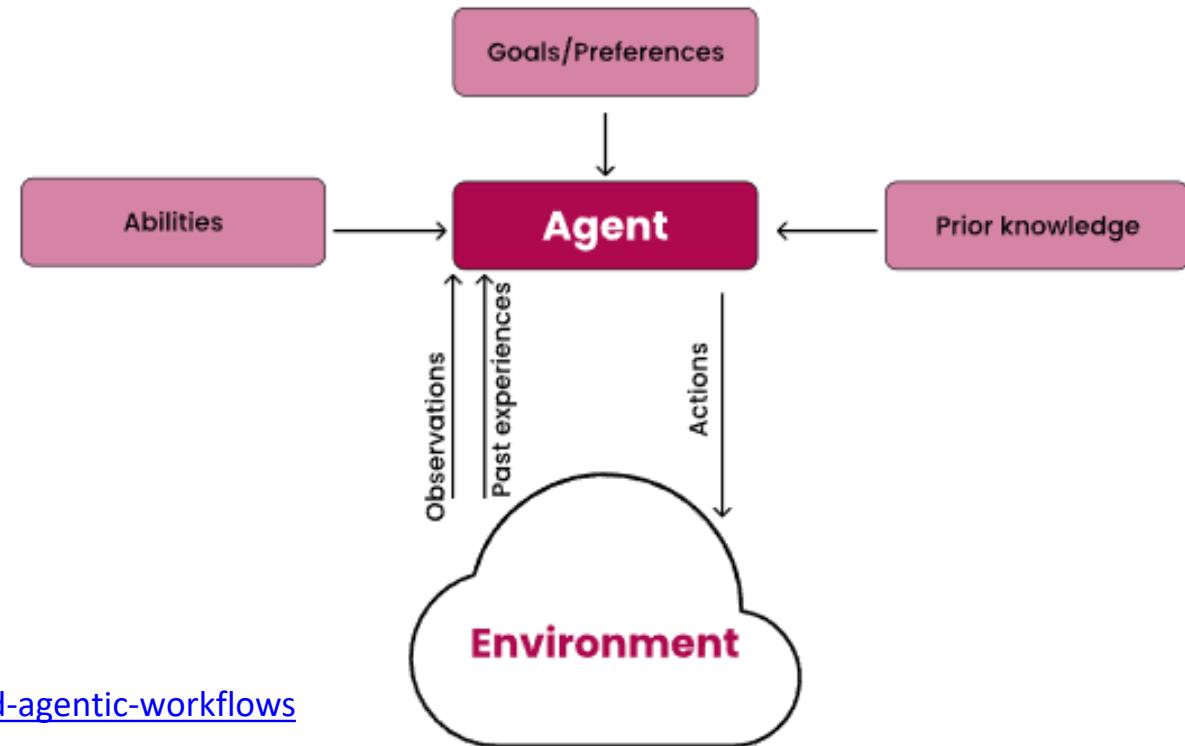
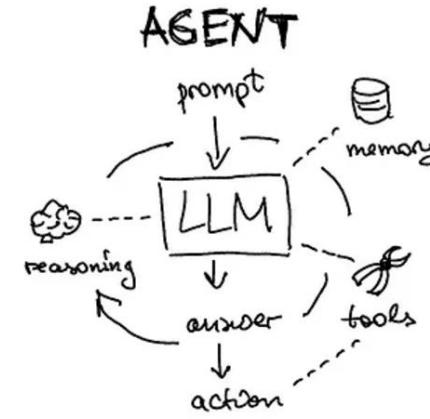
- **Definition:** Prompt engineering involves designing and refining prompts to optimize the performance of language models, particularly in RAG systems.
- **Purpose:** Enhances the quality and relevance of generated responses by effectively integrating retrieved documents with the input prompt.



<https://cobusgreyling.medium.com/12-prompt-engineering-techniques-644481c857aa>

What is an AI Agent?

- **Definition:** Autonomous systems that perform tasks on behalf of users by *interacting* with their environment and *making decisions* based on data.
- **Applications:** customer service, robotics, business automation, ...



<https://www.codiste.com/ai-agents-and-agentic-workflows>

Sources

- Textbook “Deep Learning with Python (2nd edition)” by François Chollet (2021)
- Textbook “Grokking Machine Learning” by Luis G Serrano (2021)
- Textbook “Introduction to Artificial Intelligence (2nd Ed)” by Wolfgang Ertel (2011)
- Textbook “Artificial Intelligence: A Modern Approach (3rd Ed)” by Russell & Norvig (2010)
- Textbook “Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow (3rd Ed)” by Aurélien Géron (2022).
- Some slides have been adopted and modified from the course “Artificial Intelligence: introduction and examples” (2019) by Dr. Stefaan Haspeslagh, researcher and lecturer in AI at the Vives University of Applied Sciences.
- A few slides are taken from lectures given by Prof. Dr. Celine Vens and Prof. Dr. Hendrik Blockeel (Computer Sciences, Catholic University Leuven)
- Online course “AI for Everyone” by Andrew Ng: <https://wwwdeeplearning.ai/courses/ai-for-everyone/>
- Online course on diffusion models: <https://learndeeplearning.ai/diffusion-models>
- LangChain documentation: <https://pythonlangchain.com/docs/>
- Training material about prompting created by PwC (2024)
- Other sources are mentioned on the slides
- Microsoft Copilot was used to generate some of the content

Dr. Andy Louwyck
Lecturer in AI – Applied Informatics

Commercial Sciences, Business Management and Informatics
Vives University of applied sciences – Association KU Leuven – Kortrijk, Belgium

E-mail: andy.louwyck@vives.be

LinkedIn: <https://www.linkedin.com/in/andy-louwyck-1a650a6/>