Neural Networks and Deep Learning for Scientists

38-616 & 09-616, Spring 2026

Olexandr (or Oles for short) Isayev

Department of Chemistry &
Computational Biology, SCS

olexandr@cmu.edu

Outline

- Course logistics
- Course objectives
- Course overview
- Anaconda Python& Jupyter notebook ecosystem
- Access to PSC resources & GPU computing

Class schedule

Mon & Wed 11:00- 12:30 AM

Mellon Institute 348

Canvas: https://canvas.cmu.edu/courses/52682

Slack or Discord channel ?

Course requirements

Designed for STEM master students & MCS recent graduates

Open to senior–year undergraduate students (with permission)

This is **practical**, application-oriented course, requiring skill in algorithmic problem solving.

We will use **Python & PyTorch** based tools and libraries. Prior programming experience with Python is needed.

Prerequisites: probability, linear algebra, statistical thermodynamics and intro ML or related quantitative courses

ML/DL experience

PyTorch or other DL framework experience

LLMs and Coding Assistants

M.S. in Data Analytics for Science (MS-DAS) program

- Pace of AI is crazy fast! Sorry for possible hiccups.
- Please gave your feedback. olexandr@cmu.edu
- Job market is tough, but we are here to help!

Learning objectives

- Be proficient in using modern computing technologies (Python, Jupyter notebooks and PyTorch, etc.)
- Know how to explore and classify large scientific data set using neural networks and other deep learning tools.
- Understand core components of a data analytics pipeline: EDA, classification, regressions, prediction, etc.
- Implement and analyze well-known existing ML and AI algorithms.
- Integrate multiple components of practical machine learning and deep neural network methods in a single system: data preprocessing, learning, regularization, model selection and be familiar with programming tools to accomplish it.
- Hands on experience with real-world cases on how neural networks and deep learning
- could address challenges in science.

Course Outline

- Basic concepts: Model accuracy, prediction accuracy, interpretability, supervised and unsupervised training, regularization.
- Artificial neural networks, feed-forward, activation functions, loss functions.
- Non-linear optimization, gradient descent, back-propagation
- Deep Learning tools: PyTorch, PyTorch Geometric, Hugging Face, etc
- Autoencoders, dense embedding, dimensionality reduction
- Convolutional networks, transfer learning, applications in image processing and sciences
- Recurrent networks, Transformers, GPT and their applications in NLP
- LLMs and AgenticAI
- Graph-based models, flow- and diffusion models
- Other topics: GANs, Reinforcement Learning, Multitask Learning, advanced applications of deep learning in chemical and biological sciences.

Course Structure

Monday

Lecture materials

Wednesday

- Recital/practice
- Tutorials
- Lab discussions

Reading

No textbook

Readings, **mostly DL papers** will be provided on Canvas portal and lectures.

The readings for this course are required.

We recommend you read them **before** the lecture.

Course Grades

5% for attendance

5% for class participation

50% for Lab assignments

40% for final open-ended class project

Bonus points for Top Kaggle leaderboard score

ChatGPT/Claude/AI Policy

- Embrace modern AI tools, if you find them useful for this class. Learn from it, but please clearly attribute e.g. ChatGPT, if used to comply with university policies on Ethics and Academic Integrity.

- **Academic Integrity Policy**
- Academic integrity refers to the implicit commitment that every member makes to all others in the community to practice those principles that underlie the mission of the university and define academic integrity. These are: honesty and good faith; clarity in the communication of core values; professional conduct of work; mutual trust and respect; and fairness and exemplary behavior. In this course, cheating will not be tolerated and could lead to expulsion from the university. Please remind yourself of the policy here: http://www.cmu.edu/academic-integrity/
-
- There is a zero tolerance policy on cheating. If you are found to be cheating on an exam, you will automatically receive a failing grade for the exam (which cannot be dropped) and you will be reported to the Dean of Students for further disciplinary action. This usually means an academic review board meeting and possible suspension/expulsion from the university.

Final Project

- You should work in teams of 2-3-4-5
- Open-ended project!
- Solve a science related problem with deep neural networks! Use your domain expertise
- Encourage to use your data. Try to pick a dataset of sufficiently large size (10K, 100K, 1M…)
- Jupyter notebook, which mixes together written markdown and code portions or python script and report.
  - ~1000 words (2 pages of text)
  - ~200-300 lines of code
- All text and code must be your own work, but you can adapt and built on existing models.

Final Project

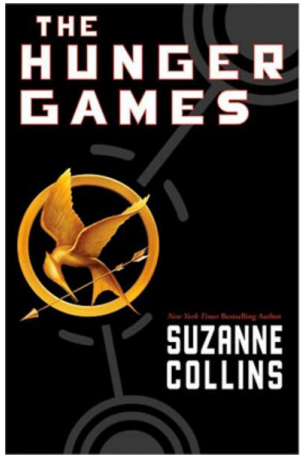
- Submit one paragraph project proposal (~February)
- Short project talk (Pitch! 1-2 slides) (~Early March)
- Presentation during last week and write short final report as a paper
- You will be graded by the course instructor and other students taking the course (peers)

Lab/Home Exercises

- Fully connected NN
- CNN
- Transformers/GPT model finetuning
- Graph-based NN

Lab auto-grading

Brought to you by



- Five Labs per semester
- Short solution discussions after each Lab
- Assignments will include 2 parts: *programmi*

  kaggle component (autoscoring)
- Sometimes you will compete

  with each other…

Dashboard | Public Leaderboard - Heritage Health Prize

This leaderboard is calculated on approximately 30% of the test data.
The final results will be based on the other 70%, so the final standings may be different.

#	Δ1w	Team Name * in the money	Score	Entries
1	–	EXL Analytics *	0.443793	555
2	–	POWERDOT	0.447651	671
3	–	Dolphin	0.450403	555
4	↑1	jack3	0.451425	455
5	↓1	Hopkins Biostat	0.451569	444
6	–	Xing Zhao	0.453081	161
7	–	Old Dogs With New Tricks	0.454096	370
8	–	Areté Associates	0.454424	112
9	–	Alice Sasandr	0.454670	376
10	↑9	J.A. Guerrero	0.454728	173

ChatGPT Policy

ChatGPT, the buzzy chatbot developed by OpenAI that is capable of writing cogent essays, solving science and math problems and producing working computer code.

This is (potentially) the future! Embrace ChatGPT, if you find it useful for this class.

Learn from it, but please clearly attribute ChatGPT (if used) to comply with university policies on Ethics and Academic Integrity.

January Class Plan

January 12	ML and neural networks history
January 14	Deep neural networks & their “anatomy”
January 21	Bridges2 introduction & GPU computing tutorial
January 26	Neural networks training
January 28	PyTorch tutorial

Questions?

Conda & Mamba Python

- For the class, we recommend you use Conda Python
- This distribution of Python, includes most libraries and tools

https://github.com/mamba-org/mamba

Installing additional packages

There are two general ways to install additional packages

```
conda (mamba) install <package name>

      conda search <package name>
      conda list


pip install <package name>
```

uv

- https://docs.astral.sh/uv/

```
Mac/Linux: curl -LsSf https://astral.sh/uv/install.sh | sh

Windows: powershell -ExecutionPolicy ByPass -c "irm
https://astral.sh/uv/install.ps1 | iex"
```

- Tutorial/intro: https://realpython.com/python-uv/

PyTorch

www.pytorch.org

Jupyter notebook

Launch jupyter via the command:

```
jupyter notebook
```

Open in browser: http://localhost:8888

New (alternative) environment: `jupyter lab`

Access to GPU resources

- Google Colab

Pittsburgh Supercomputing Center (www.psc.edu)

- PSC Bridges-2 GPU-AI -> Nvidia Tesla V100 and A100 (DGX boxes)

If you are NOT in MS-DAS program, but taking this course:

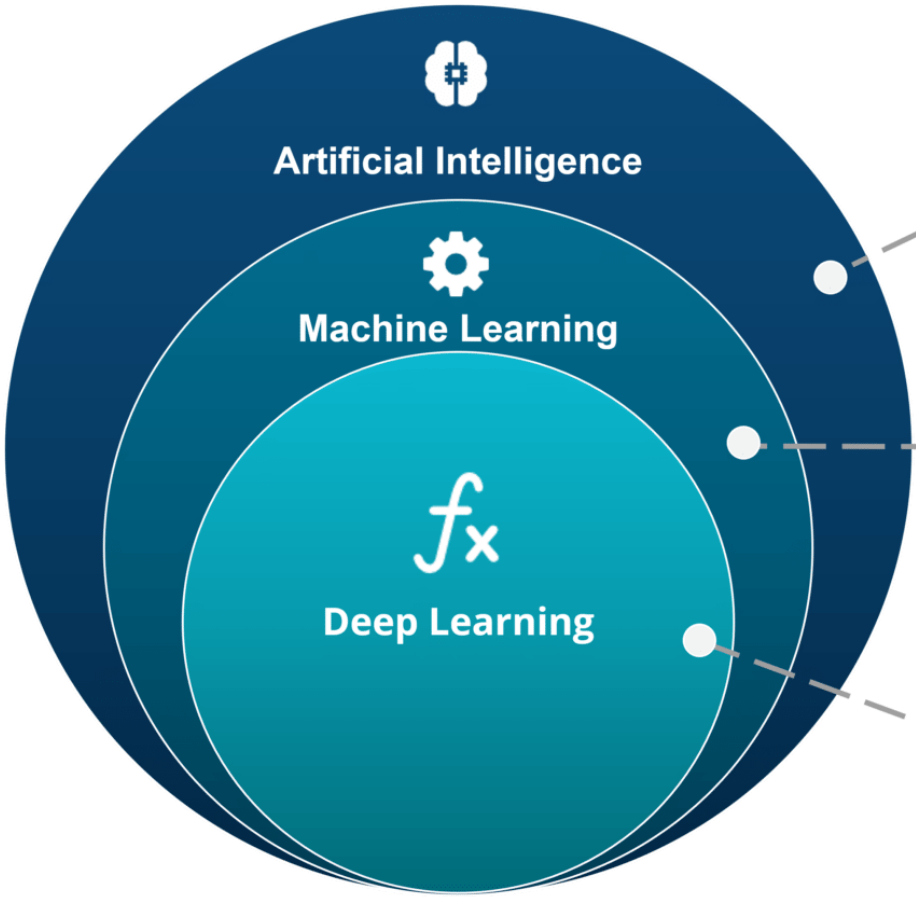
- Please go to https://identity.access-ci.org/new-user and register.
- **Register with an existing identity**: Using an existing University account when registering with ACCESS simplifies the sign-up process and enables you to log in to ACCESS using that existing account.
- Select an Identity Provider -> Carnegie Mellon University

Please send me your username. I will add you to the allocation.

ML Refresher and History of Deep Learning

38-616, Spring 2023

Olexandr Isayev
Department of Chemistry, CMU
olexandr@cmu.edu



ARTIFICIAL INTELLIGENCE

A technique which enables machines to mimic human behaviour

MACHINE LEARNING

Subset of AI technique which use statistical methods to enable machines to improve with experience

DEEP LEARNING

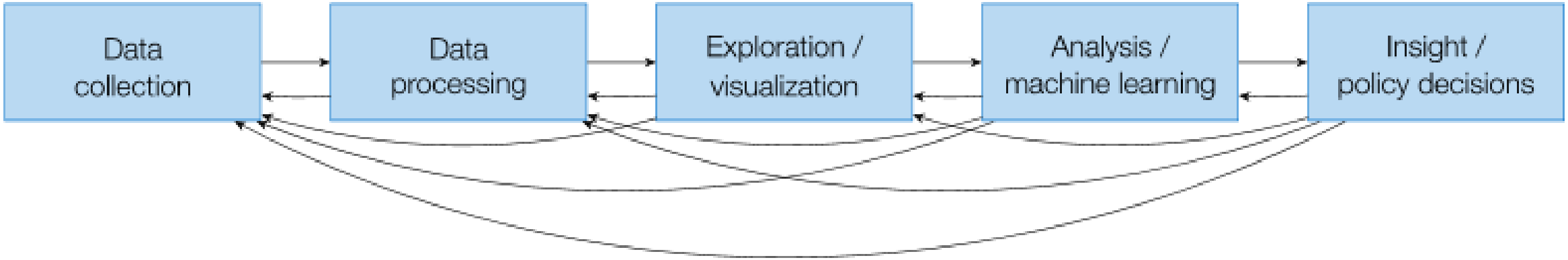
Subset of ML which make the computation of multi-layer neural network feasible

Artificial Intelligence (AI) the theory and development of computer systems able to perform tasks that normally require human intelligence, such as visual perception, speech recognition, decision-making, and translation between languages.

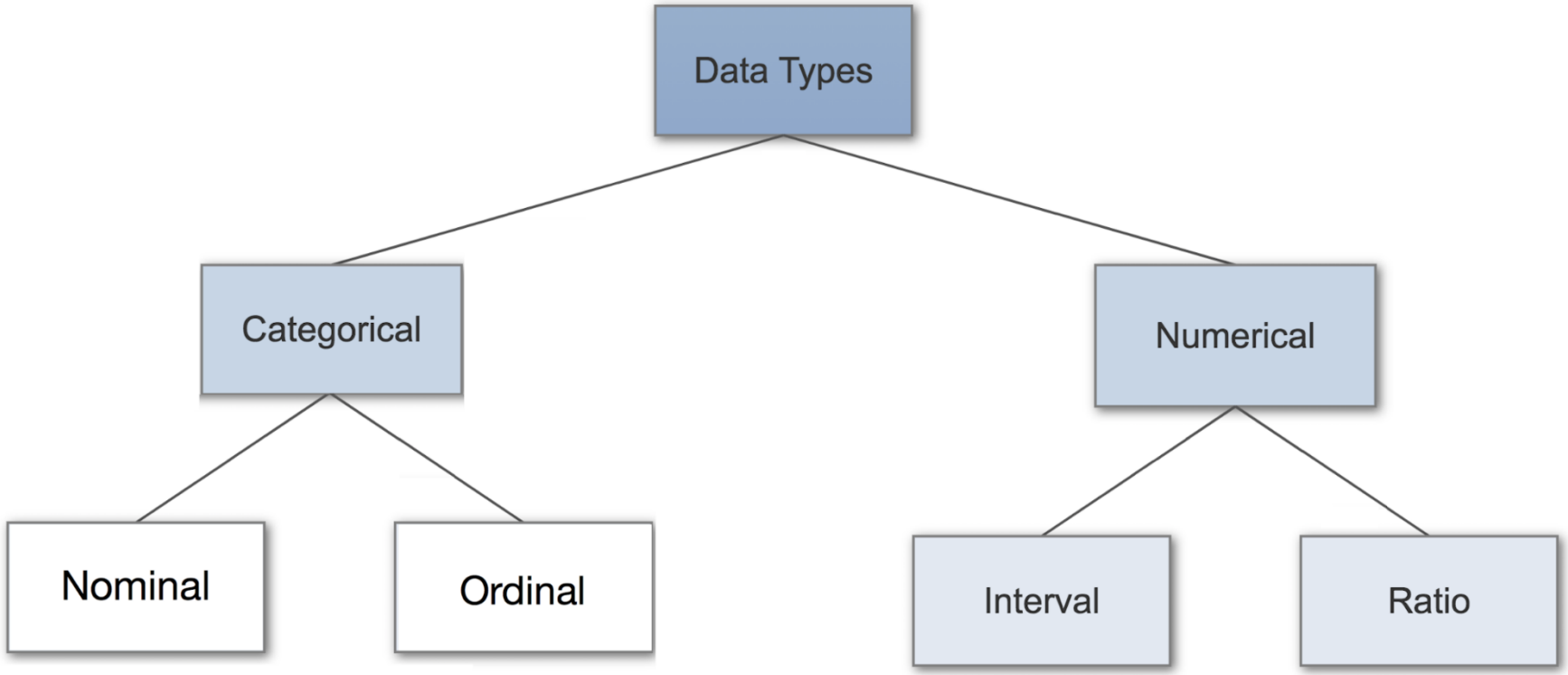
Machine learning (**ML**) is the study of algorithms and techniques that computer systems use to perform a specific task without using explicit instructions, relying on patterns and inference instead.

Machine learning (**ML**) is the collection of algorithms that learn from experiences and their outcomes and is able to predict the outcome of new experience

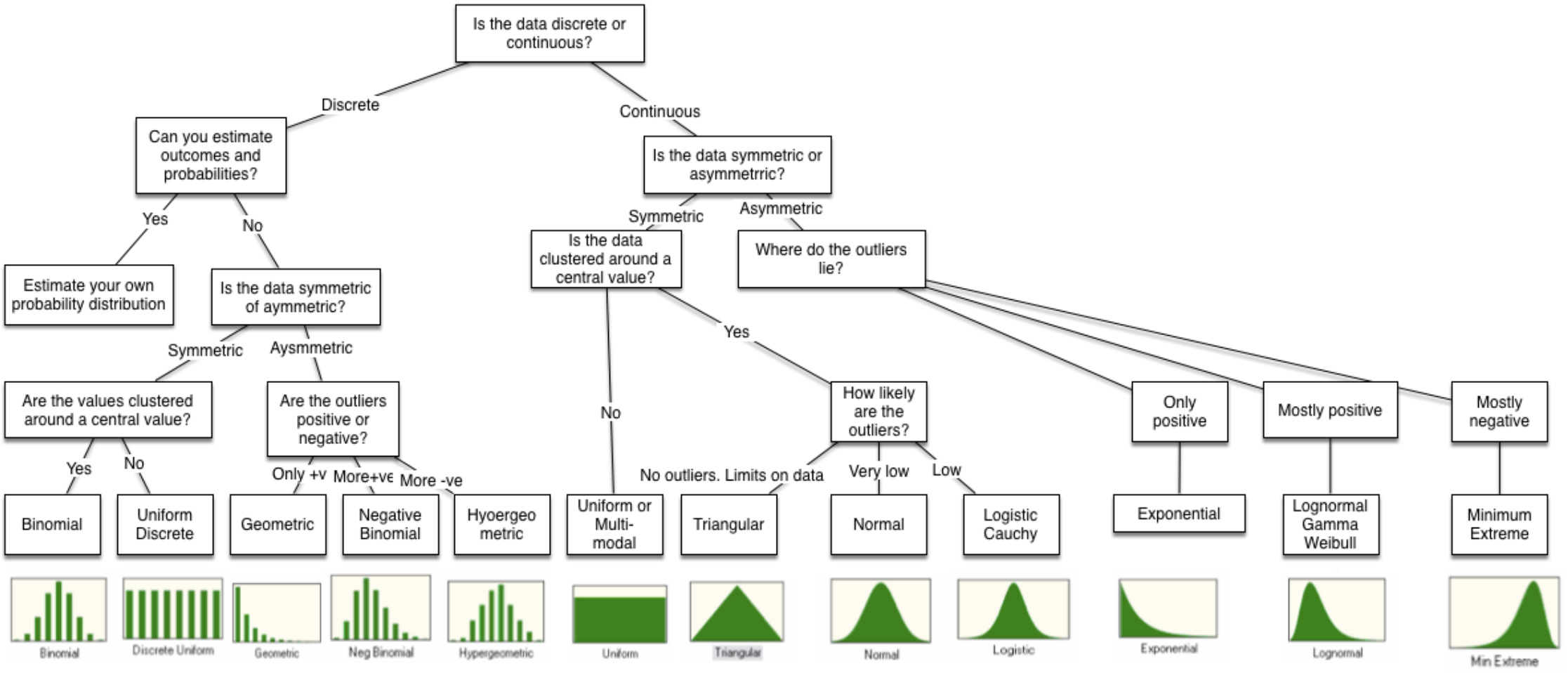
Course point of view: Data Pipeline



Basic Data Types in Statistics



Data and Distributions



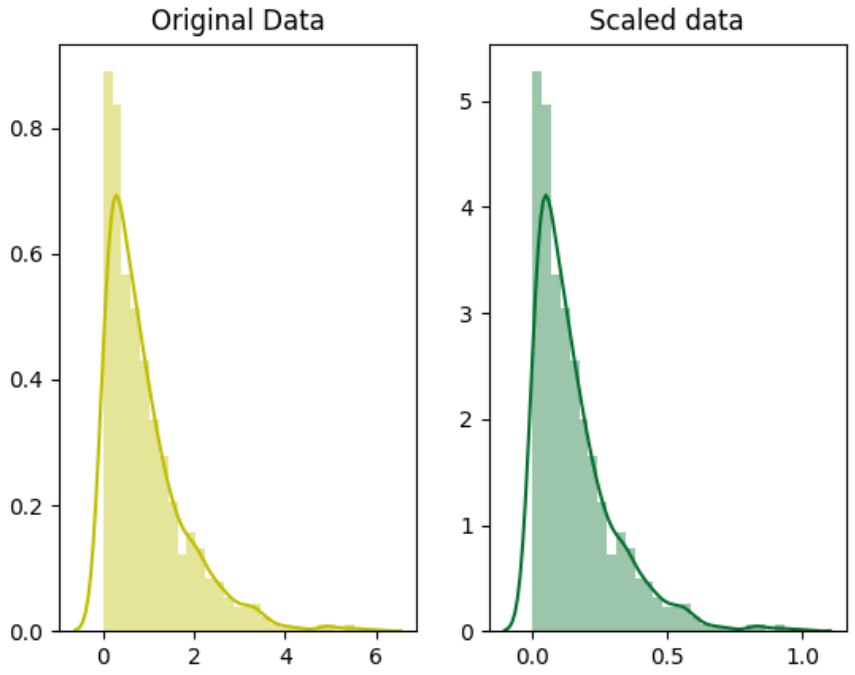
Summary Statistics

Provides:	Nominal	Ordinal	Interval	Ratio
The "order" of values is known		✔	✔	✔
"Counts," aka "Frequency of Distribution"	✔	✔	✔	✔
Mode	✔	✔	✔	✔
Median		✔	✔	✔
Mean			✔	✔
Can quantify the difference between each value			✔	✔
Can add or subtract values			✔	✔
Can multiple and divide values				✔
Has "true zero"				✔

Scaling

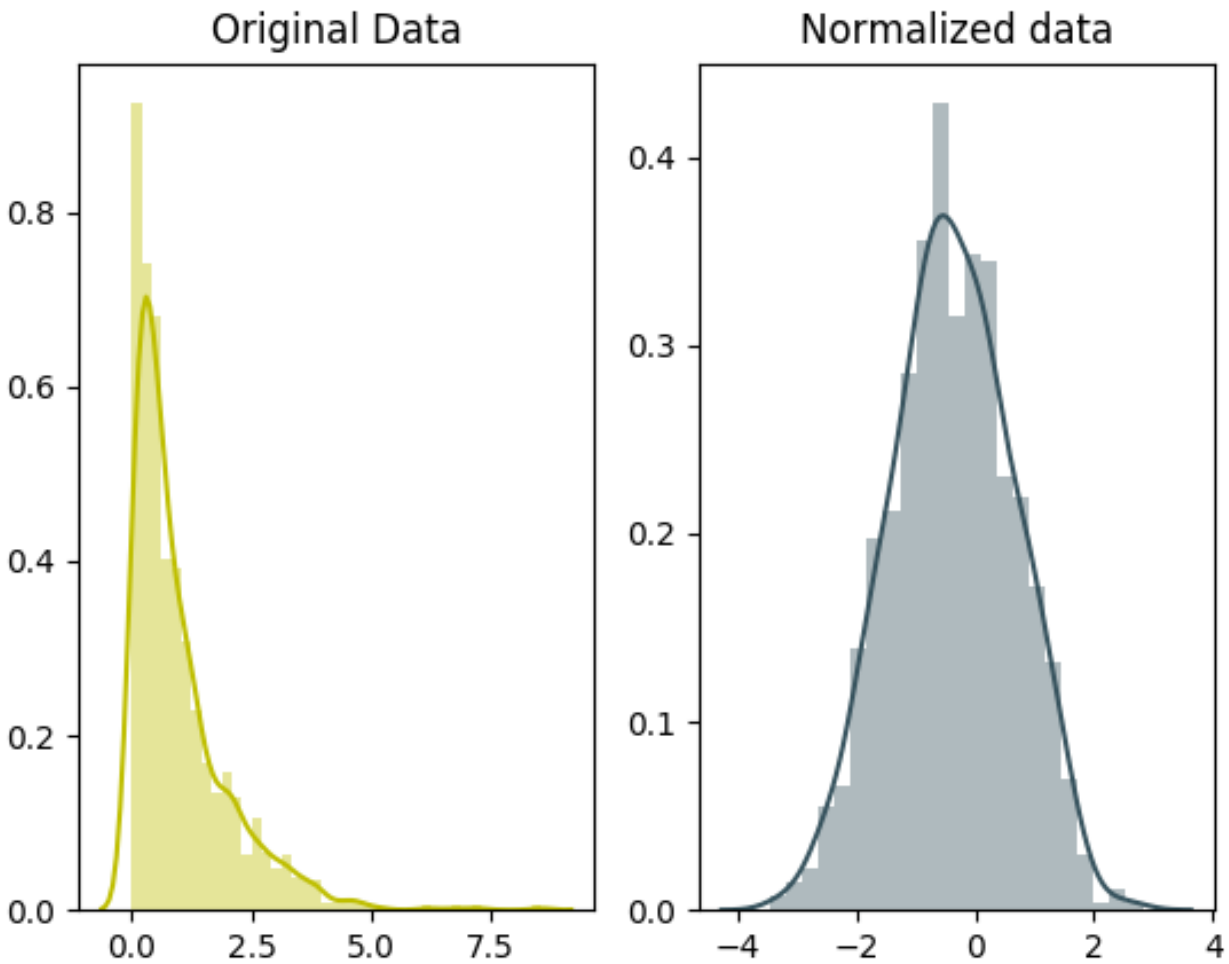
In scaling *(also called **min-max scaling**)*, you transform the data such that the features are within a specific range e.g. [0, 1].

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$



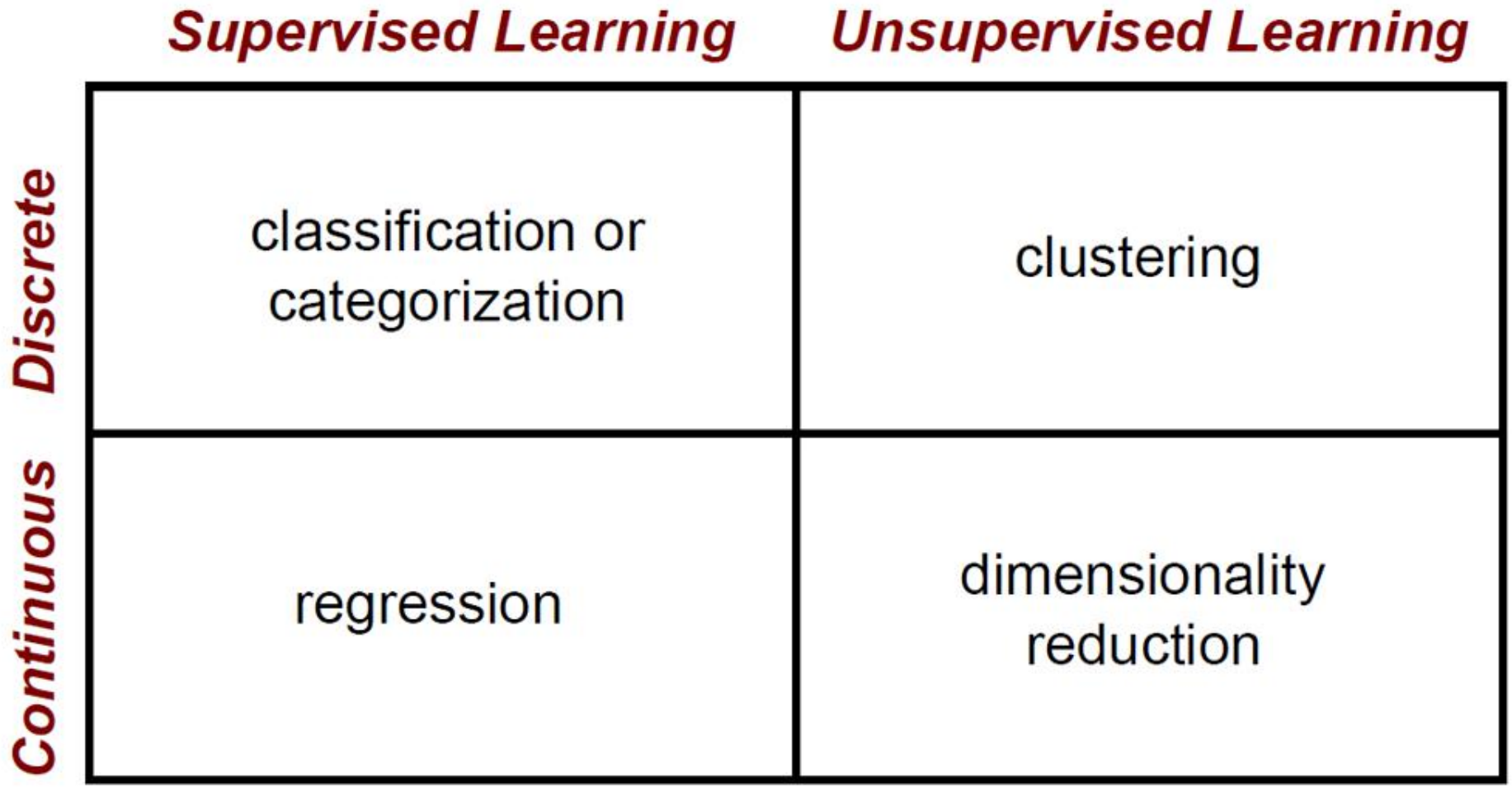
Good practice: always scale continuous data

Standardization



$$x' = \frac{x - x_{mean}}{\sigma}$$

Machine Learning Problems



Machine Learning

- **Supervised:** We are given input samples (X) and output samples (y) of a function y = f(X). We would like to "learn" f, and evaluate it on new data. Types:
  - **Classification:** y is discrete (class labels).
  - **Regression:** y is continuous, e.g. linear regression.

- **Unsupervised:** Given only samples X of the data, we compute a function f such that y = f(X) is "simpler".
  - **Clustering:** y is discrete
  - Y is continuous: **Matrix factorization, Kalman filtering, unsupervised neural networks**.

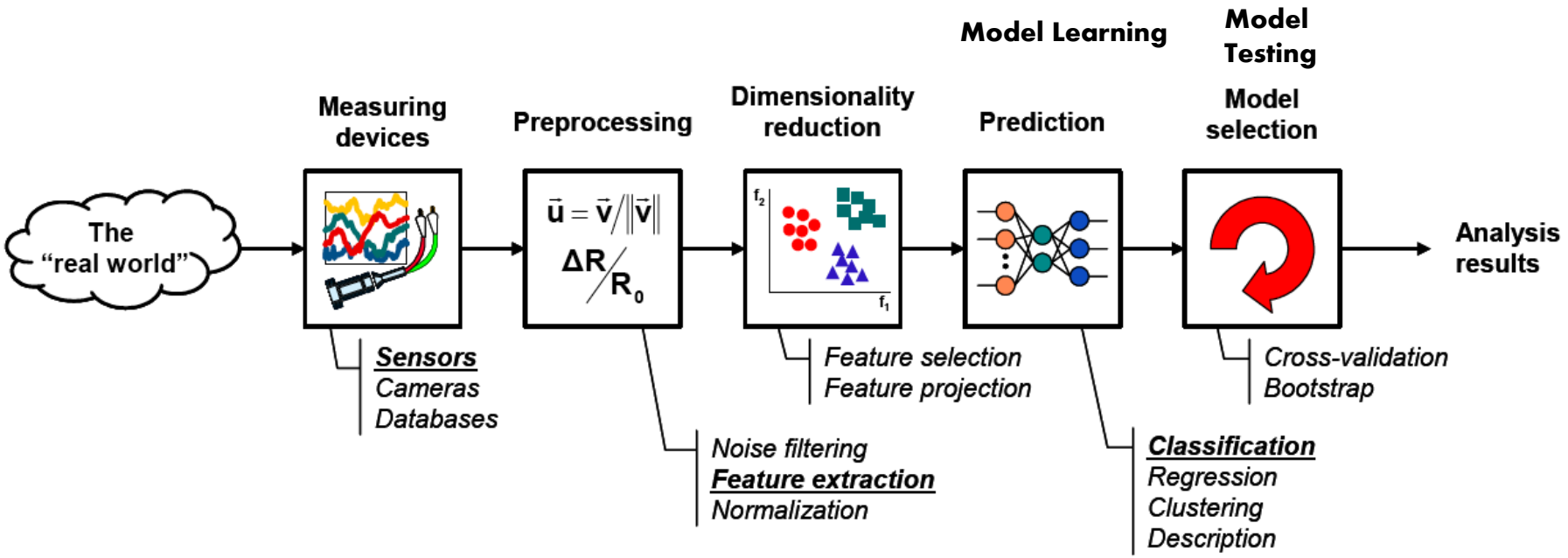
Supervised learning vs. unsupervised learning

- Supervised learning: discover patterns in the data that relate data attributes with a target (class) attribute.
  - These patterns are then utilized to predict the values of the target attribute in future data instances.
- Unsupervised learning: The data have no target attribute.
  - We want to explore the data to find some intrinsic structures in them.

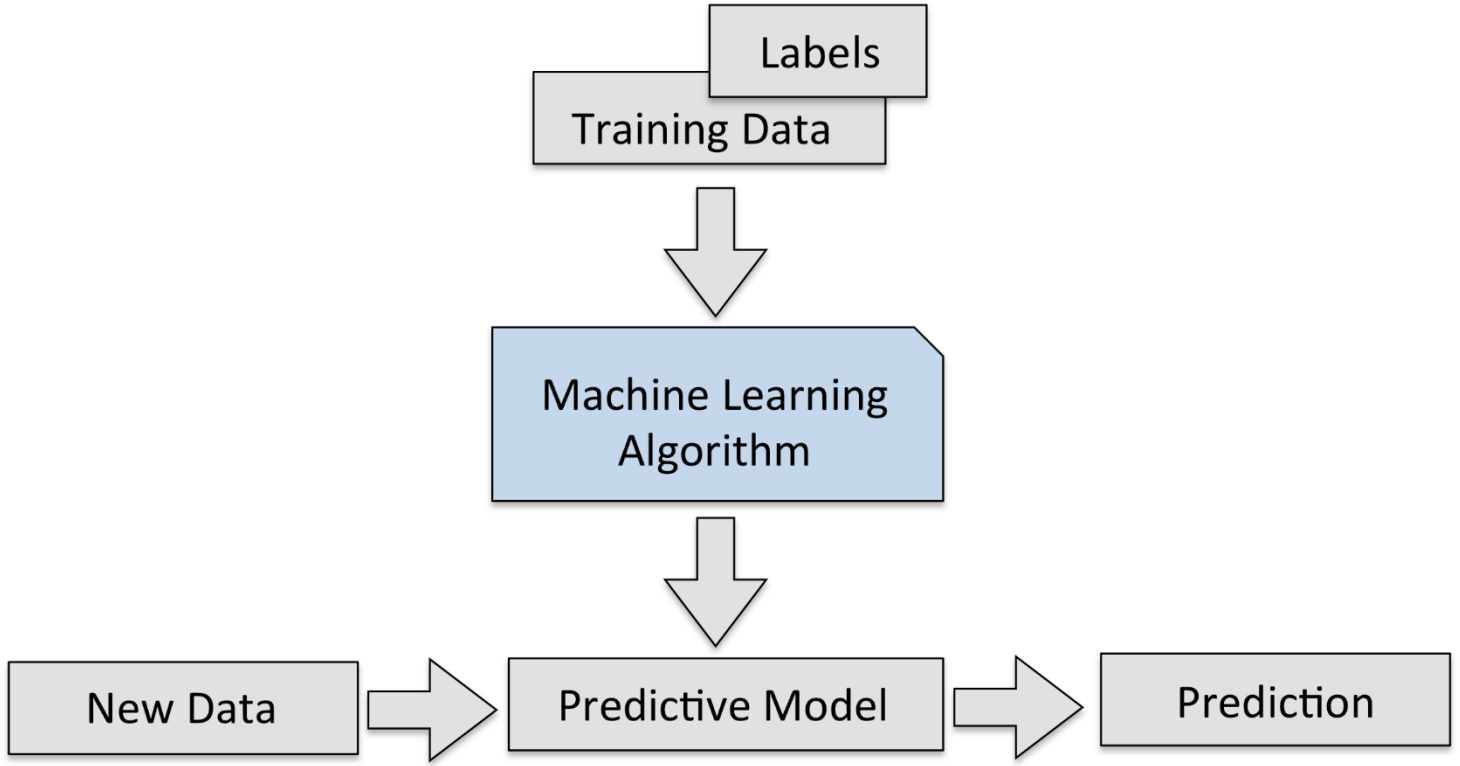
Clustering

- Clustering is a technique for finding similarity groups in data, called **clusters**. I.e.,
  - it groups data instances that are similar to (near) each other in one cluster and data instances that are very different (far away) from each other into different clusters.
- Clustering is often called an **unsupervised learning** task as no class values denoting an *a priori* grouping of the data instances are given, which is the case in supervised learning.
- Due to historical reasons, clustering is often considered synonymous with unsupervised learning.
  - In fact, association rule mining is also unsupervised
- This lecture focuses on clustering.

The Learning Process



Making predictions about the future with supervised learning



Predicting from Samples

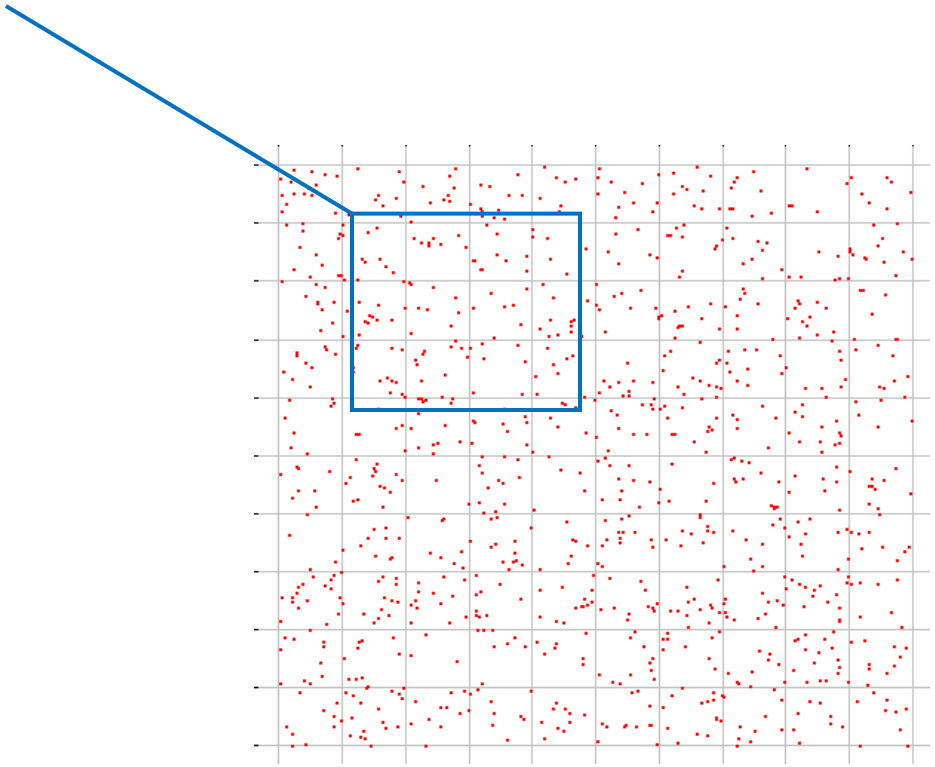
- Most datasets are **samples** from an **infinite population**.
- We are most interested in **models of the population**, but we have access only to a **sample** of it.

For datasets consisting of (X,y)

- features X + label y

a model is a prediction y = f(X)

We train on a training sample D
and we denote the model as $f_D(X)$



Bias and Variance Tradeoff

There is usually a bias-variance tradeoff caused by model complexity.

Complex models (many parameters) usually have lower bias, but higher variance.

Simple models (few parameters) have higher bias, but lower variance.

Bias and Variance Tradeoff

The total expected error is

$$Bias^2 + Variance$$

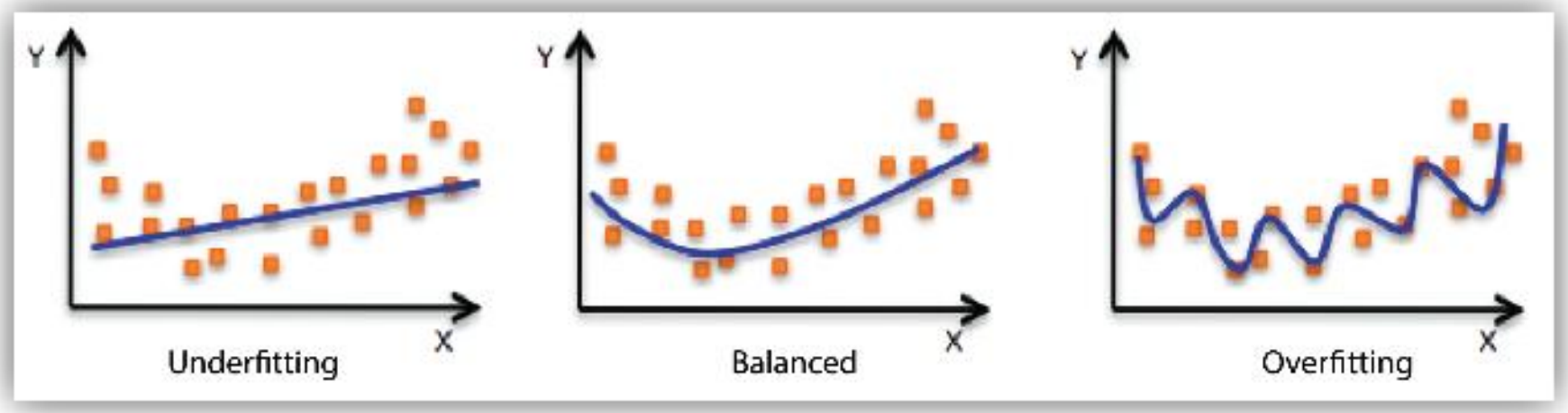
Because of the bias-variance trade-off, we want to **balance** these two contributions.

If $Variance$ strongly dominates, it means there is too much variation between models. This is called **over-fitting**.

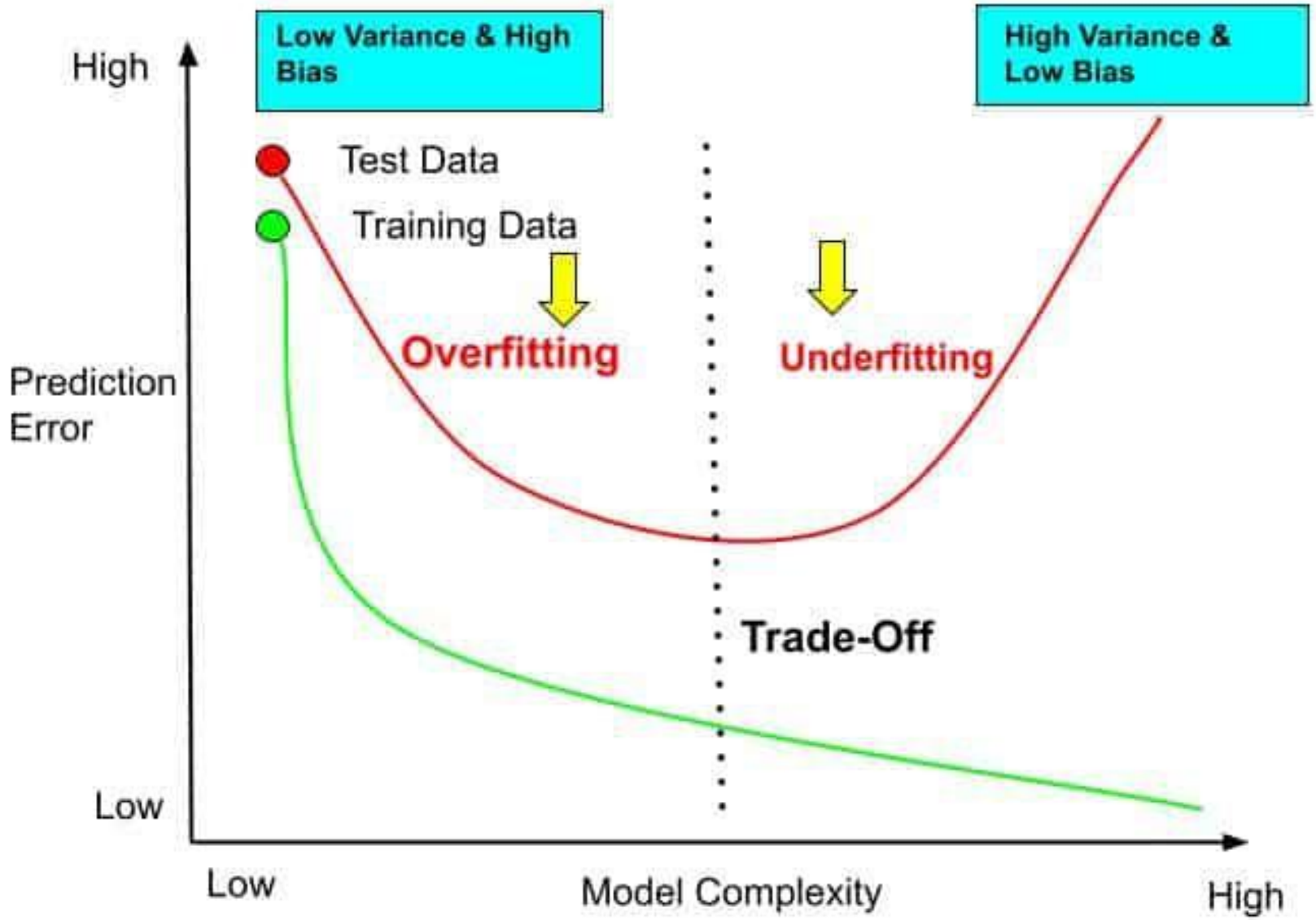
If $Bias$ strongly dominates, then the models are not fitting the data well enough. This is called **under-fitting**.

Underfitting vs. Overfitting

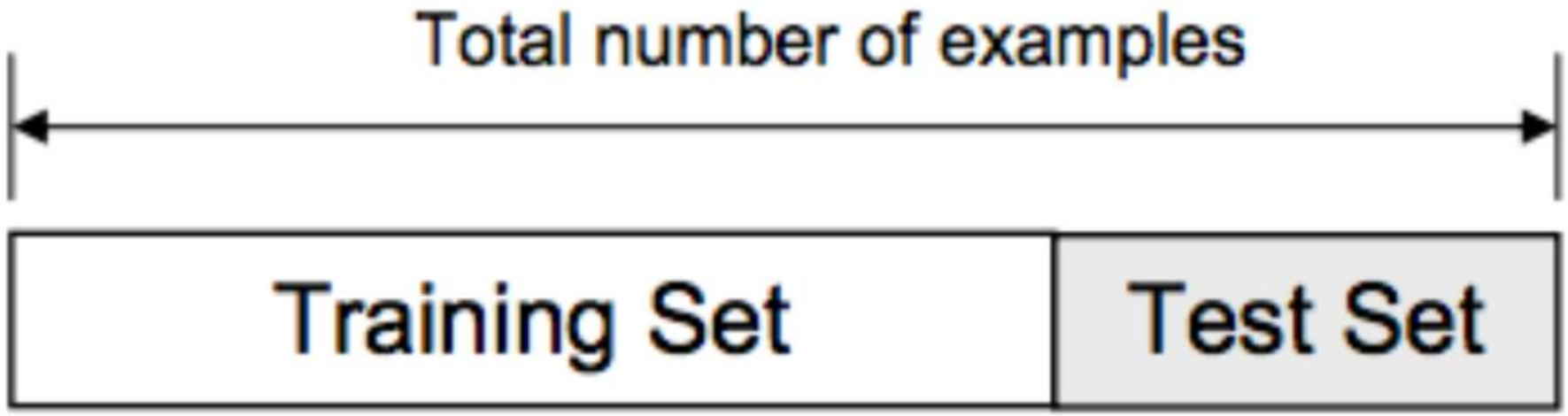
Understanding model fit is important for understanding the root cause for poor model accuracy. This understanding will guide you to take corrective steps. We can determine whether a predictive model is underfitting or overfitting the training data by looking at the prediction error on the training data and the evaluation data.



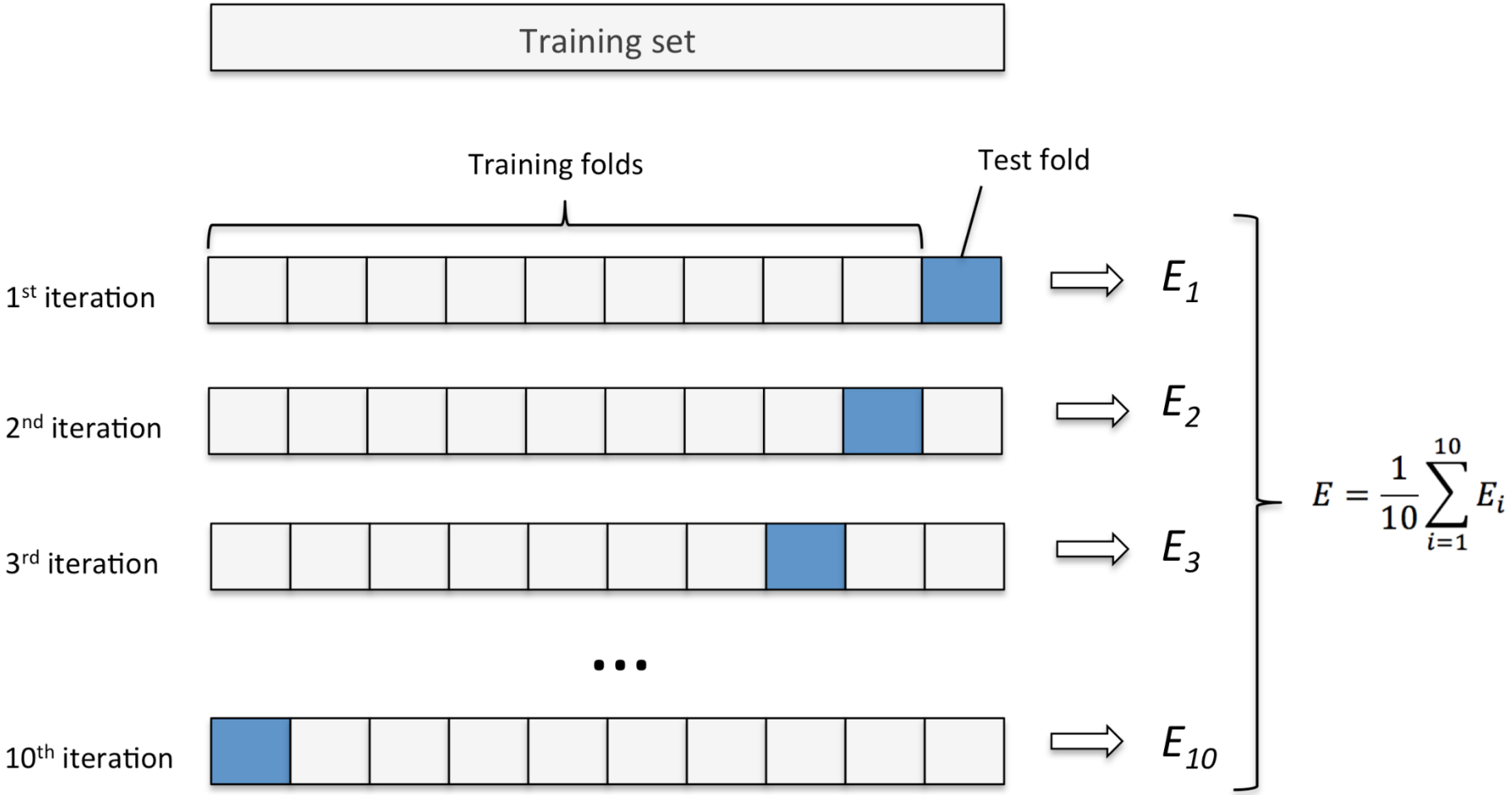
Your model is underfitting the training data when the model performs poorly on the training data. This is because the model is unable to capture the relationship between the input examples (often called X) and the target values (often called Y). Your model is overfitting your training data when you see that the model performs well on the training data but does not perform well on the evaluation data. This is because the model is memorizing the data it has seen and is unable to generalize to unseen examples.


Low Variance & High Bias
High Variance & Low Bias
High
Test Data
Training Data
Overfitting
Underfitting
Prediction Error
Trade-Off
Low
Low
Model Complexity
High

Training-Test Split



K-fold cross-validation



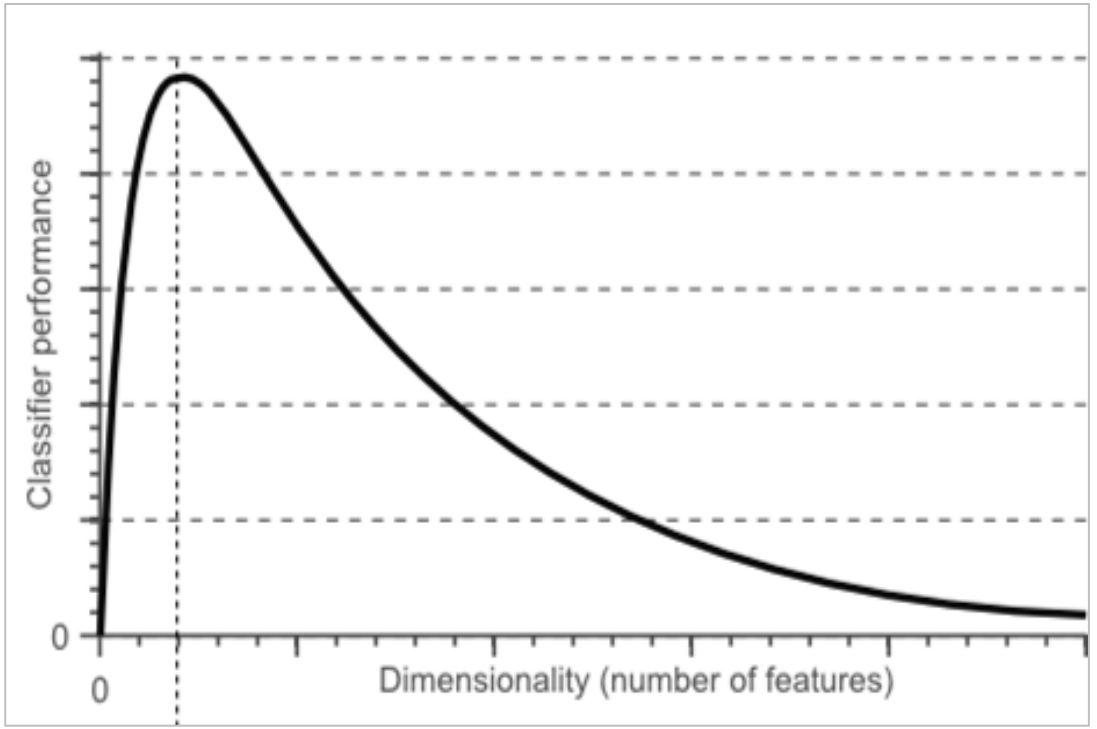
Huge number of dimensions causes problems

Data becomes very sparse, some algorithms become meaningless (e.g. density based clustering)

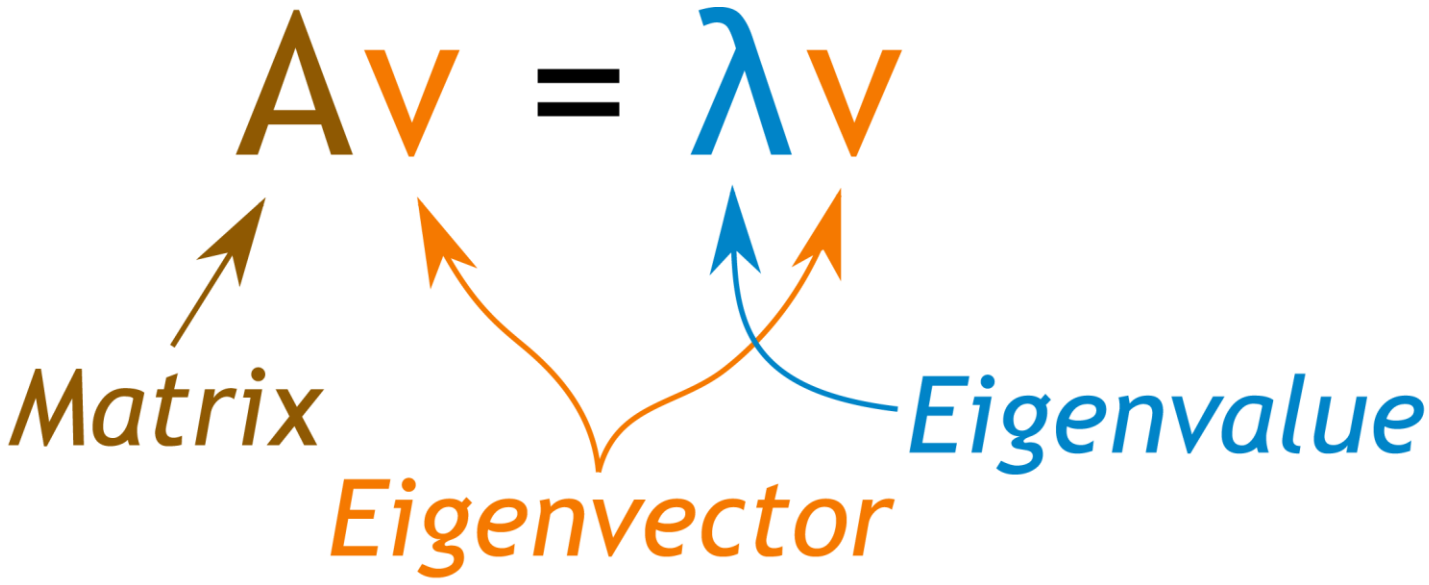
The complexity of several algorithms depends on the dimensionality and they become infeasible.

Curse of Dimensionality

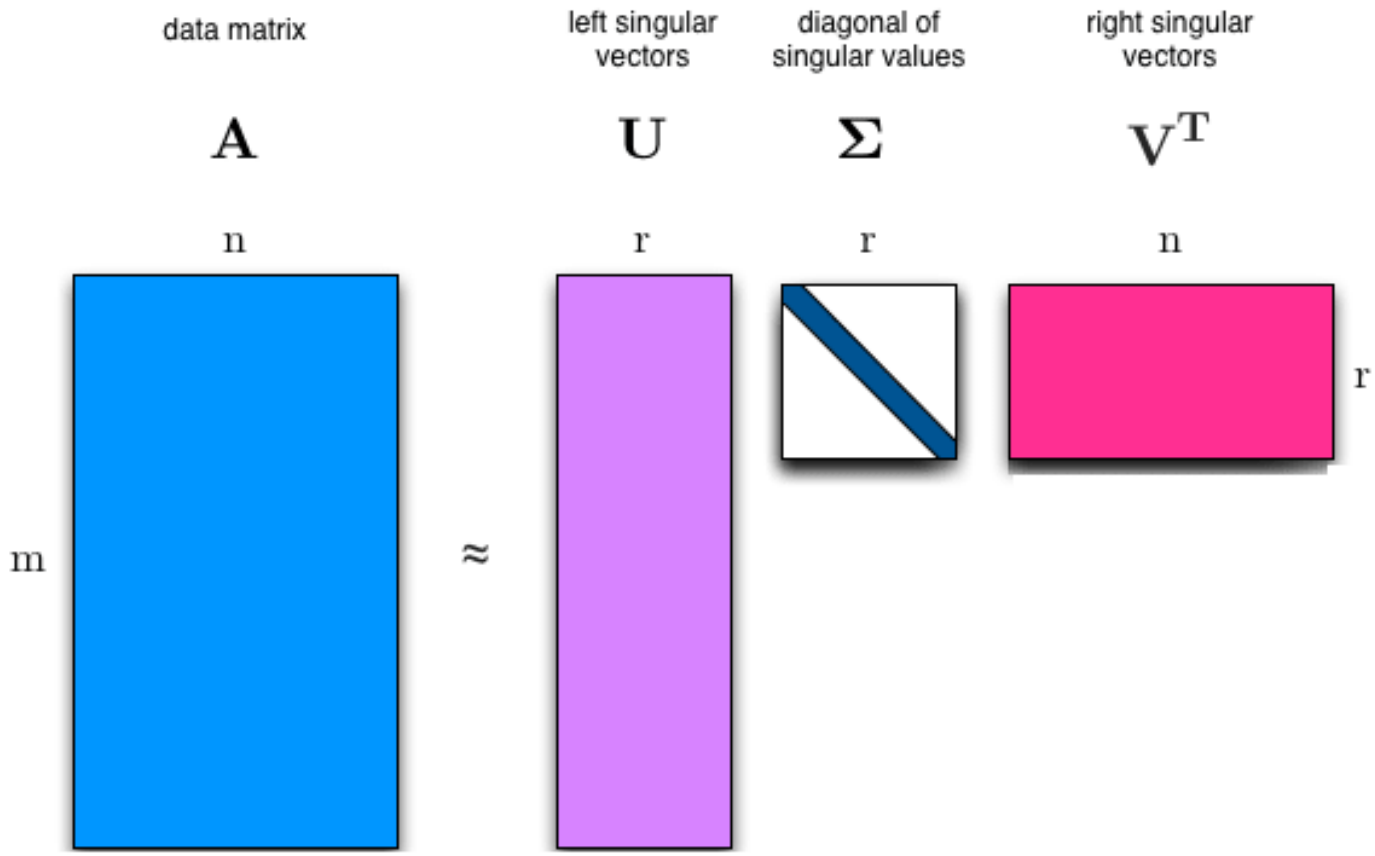
Machine learning algorithms tend to over-fit data when the sample has a lot of predictor variables. There is a number of features above which the performance of a ML will degrade rather than improve.



Eigenvalues and Eigenvectors

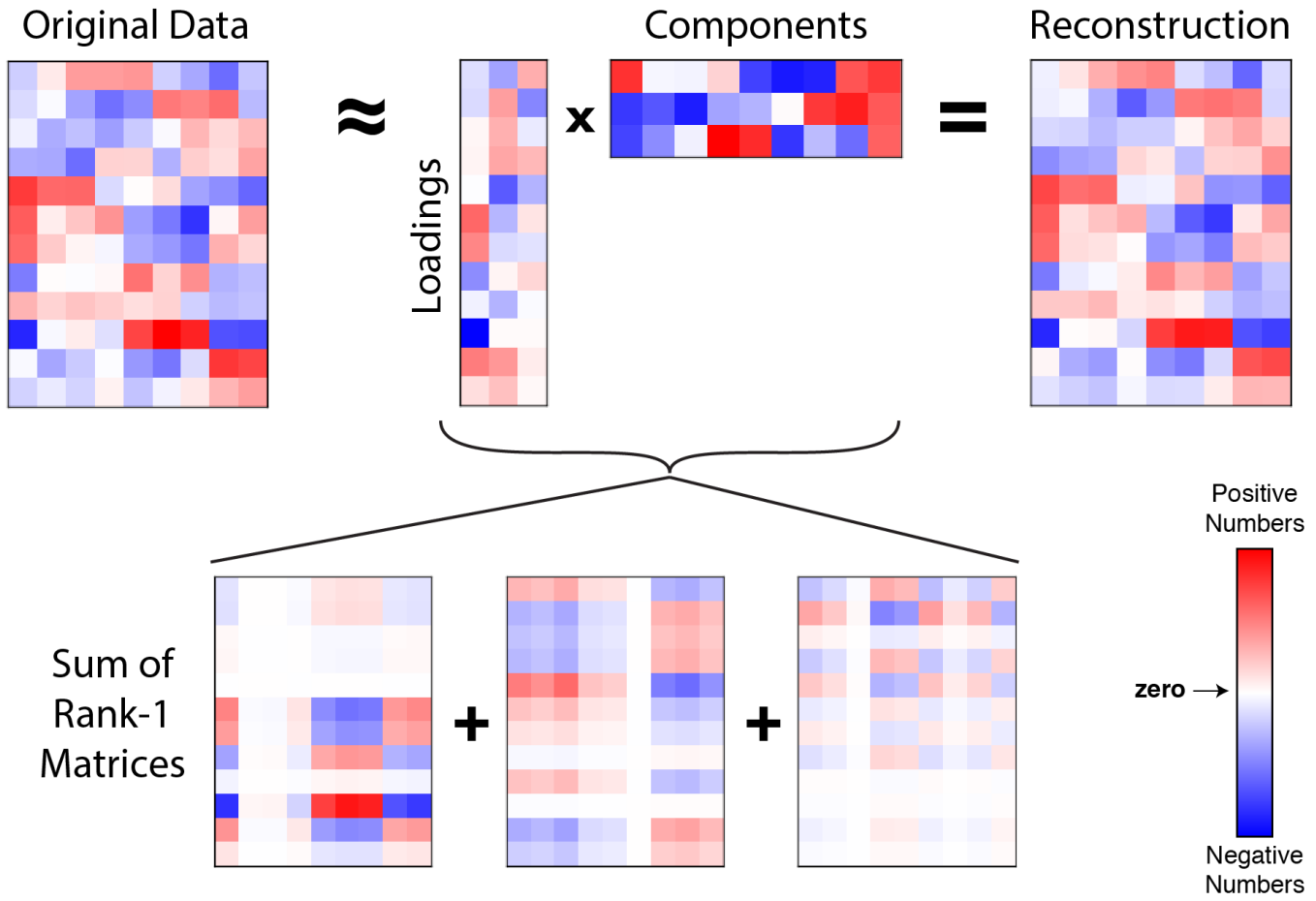


Singular Value Decomposition



The singular value decomposition (SVD) is a factorization of a real or complex matrix that generalizes the eigendecomposition of a square normal matrix to any m×n matrix.

SVD/PCA and Rank-k approximations



Milestones of deep learning

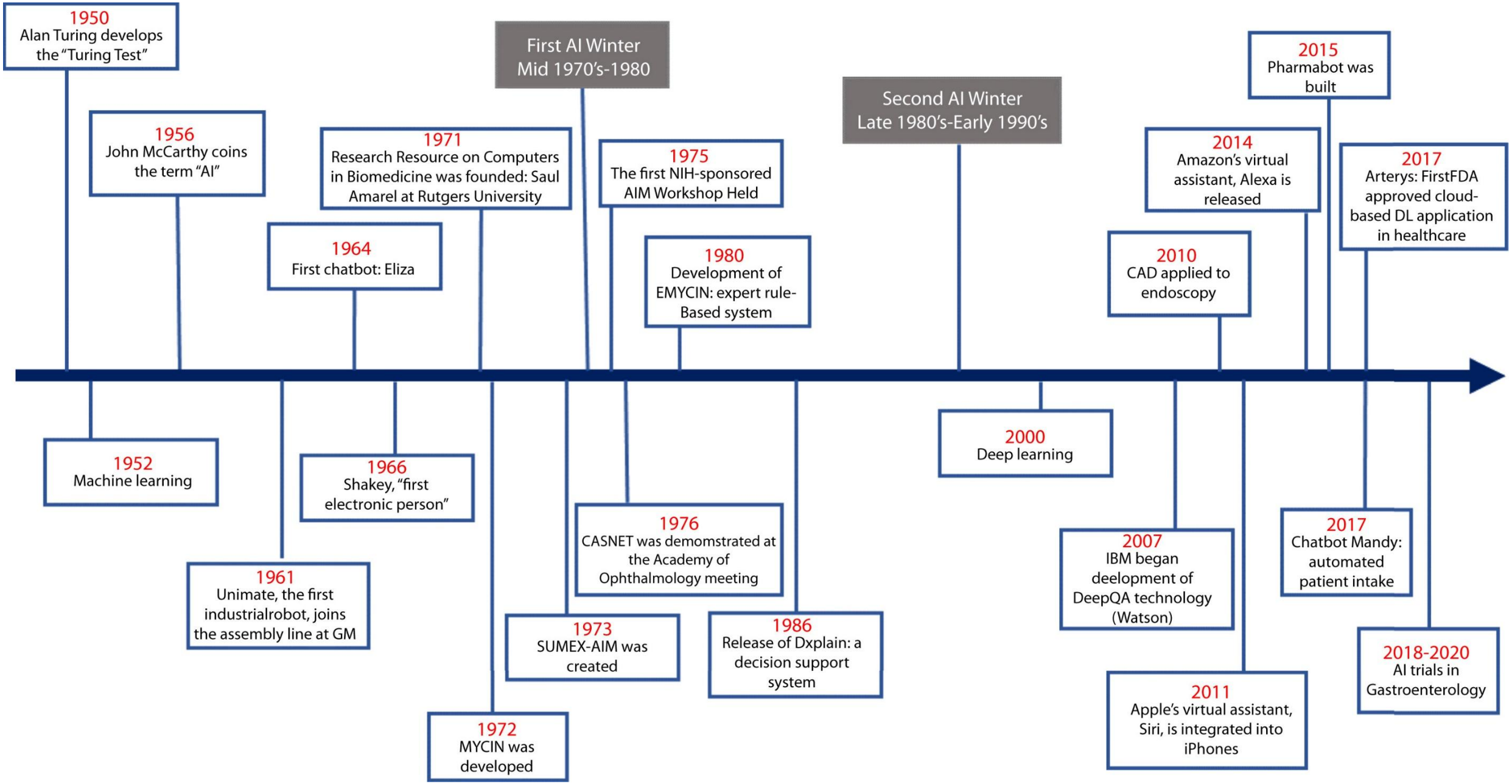
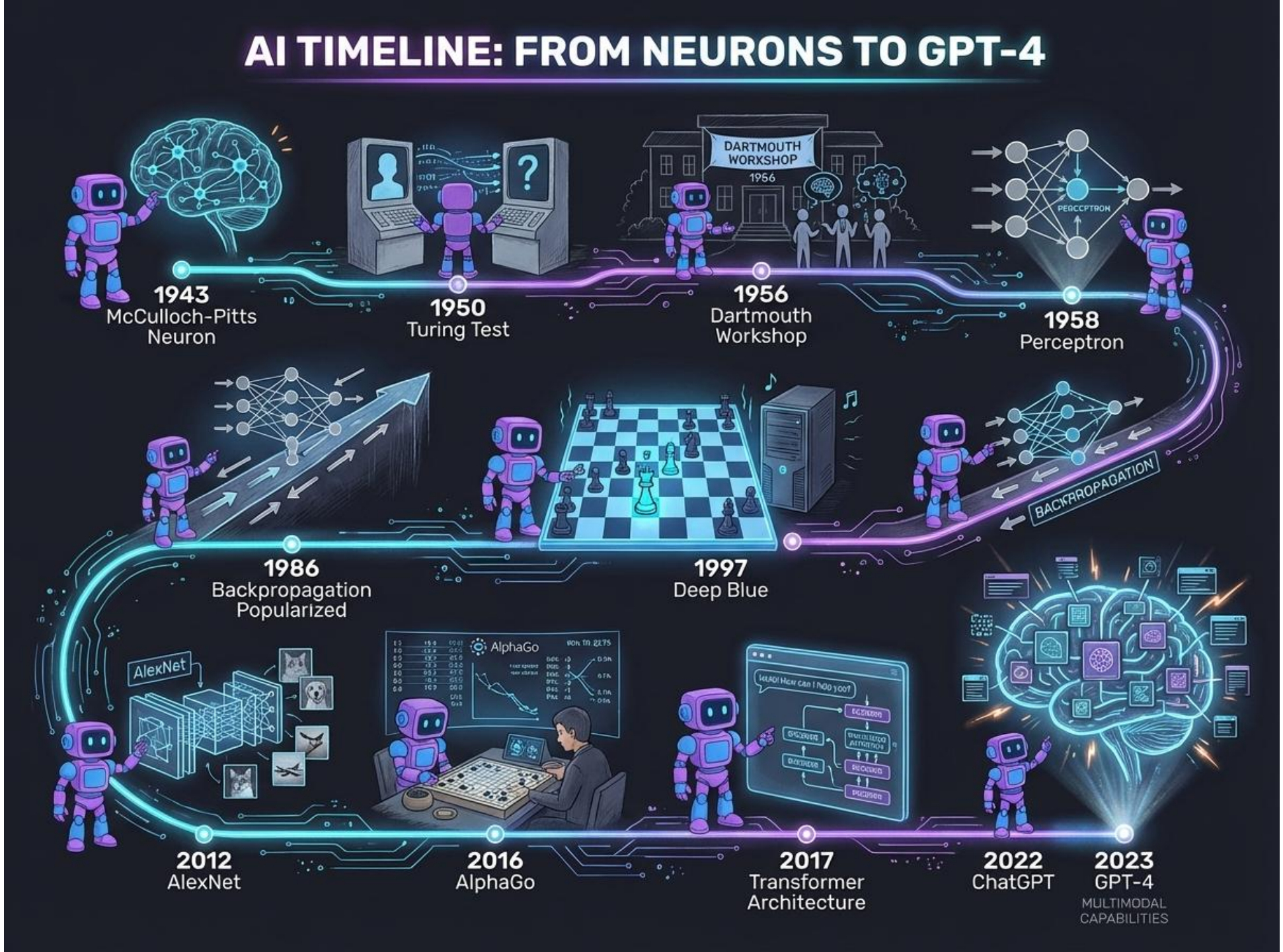


Image credit: https://doi.org/10.1016/j.gie.2020.06.040

AI TIMELINE: FROM NEURONS TO GPT-4



A few historical milestones

1958 New York Times…

NEW NAVY DEVICE LEARNS BY DOING

Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

WASHINGTON, July 7 (UPI) —The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The embryo—the Weather Bureau's $2,000,000 "704" computer—learned to differentiate between right and left after fifty attempts in the Navy's demonstration for newsmen.

The service said it would use this principle to build the first of its Perceptron thinking machines that will be able to read and write. It is expected to be finished in about a year at a cost of $100,000.

Dr. Frank Rosenblatt, designer of the Perceptron, conducted the demonstration. He said the machine would be the first device to think as the human brain. As do human beings, Perceptron will make mistakes at first, but will grow wiser as it gains experience, he said.

Dr. Rosenblatt, a research psychologist at the Cornell Aeronautical Laboratory, Buffalo, said Perceptrons might be fired to the planets as mechanical space explorers.

Without Human Controls

The Navy said the perceptron would be the first non-living mechanism "capable of receiving, recognizing and identifying its surroundings without any human training or control."

The "brain" is designed to remember images and information it has perceived itself. Ordinary computers remember only what is fed into them on punch cards or magnetic tape.

Later Perceptrons will be able to recognize people and call out their names and instantly translate speech in one language to speech or writing in another language, it was predicted.

Mr. Rosenblatt said in principle it would be possible to build brains that could reproduce themselves on an assembly line and which would be conscious of their existence.

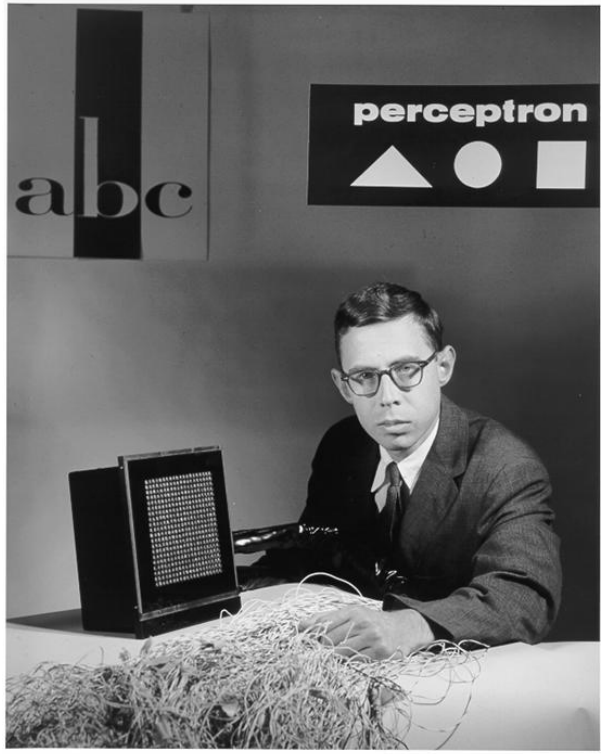
In today's demonstration, the "704" was fed two cards, one with squares marked on the left side and the other with squares on the right side.

Learns by Doing

In the first fifty trials, the machine made no distinction between them. It then started registering a "Q" for the left squares and "O" for the right squares.

Dr. Rosenblatt said he could explain why the machine learned only in highly technical terms. But he said the computer had undergone a "self-induced change in the wiring diagram."

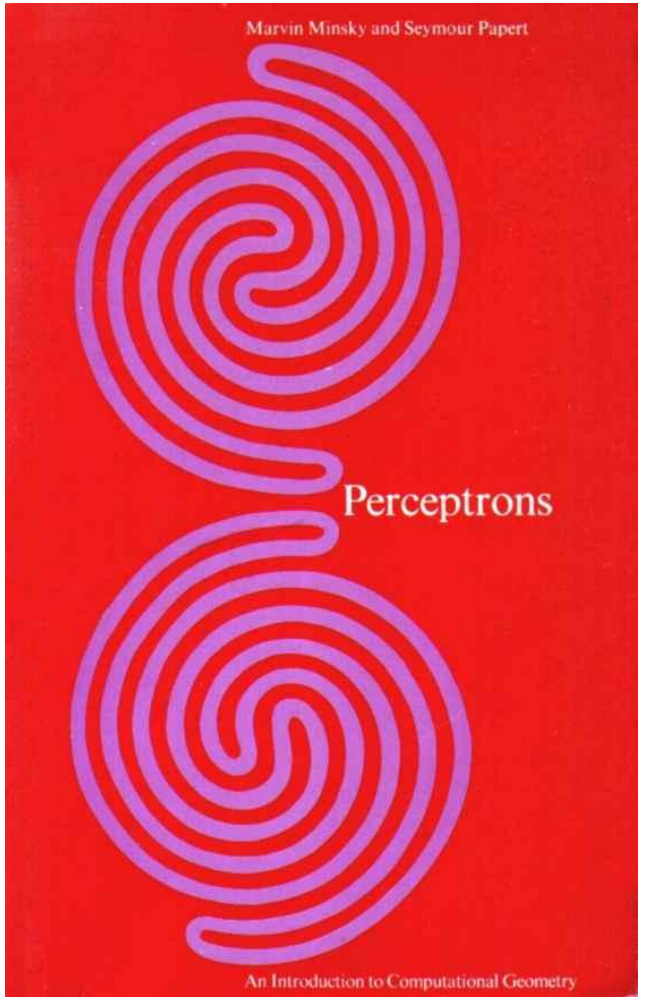
The first Perceptron will have about 1,000 electronic "association cells" receiving electrical impulses from an eye-like scanning device with 400 photo-cells. The human brain has 10,000,000,000 responsive cells, including 100,000,000 connections with the eyes.



Frank Rosenblatt (1928-1971)

A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
  - Fascinating reading: M. Olazaran, A Sociological Study of the Official History of the Perceptrons Controversy, *Social Studies of Science*, 1996

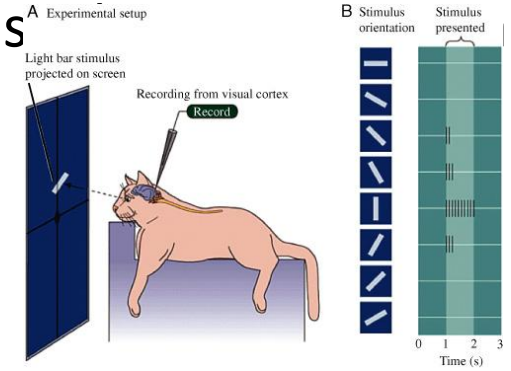


A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
  - Video (short version)
  - Inspired by the findings of Hubel & Wiesel about the hierarchical organization

Kunihiko Fukushima

is n 77)



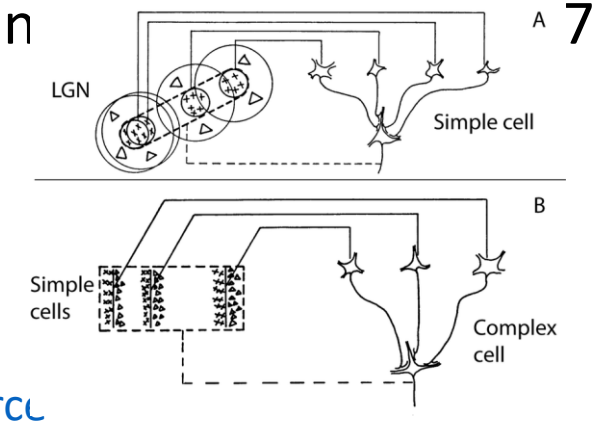
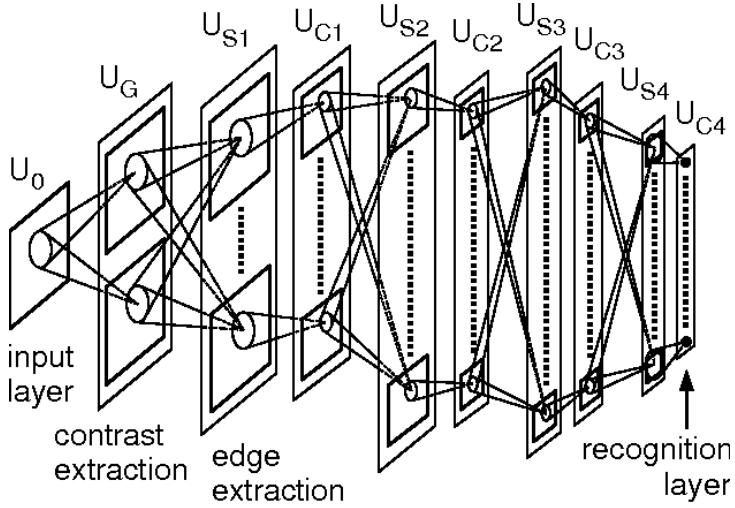


Image sourc



A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
- 1986: Back-propagation
  - Origins in control theory and optimization: Kelley (1960), Dreyfus (1962), Bryson & Ho (1969), Linnainmaa (1970)
  - Application to neural networks: Werbos (1974)
  - Popularized by Rumelhart, Hinton & Williams (1986)

A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
- 1986: Back-propagation
- 1989 – 1998: Convolutional neural networks
  - LeNet to LeNet-5

Yann LeCun
2018 ACM Turing Award winner
(with Hinton and Bengio)

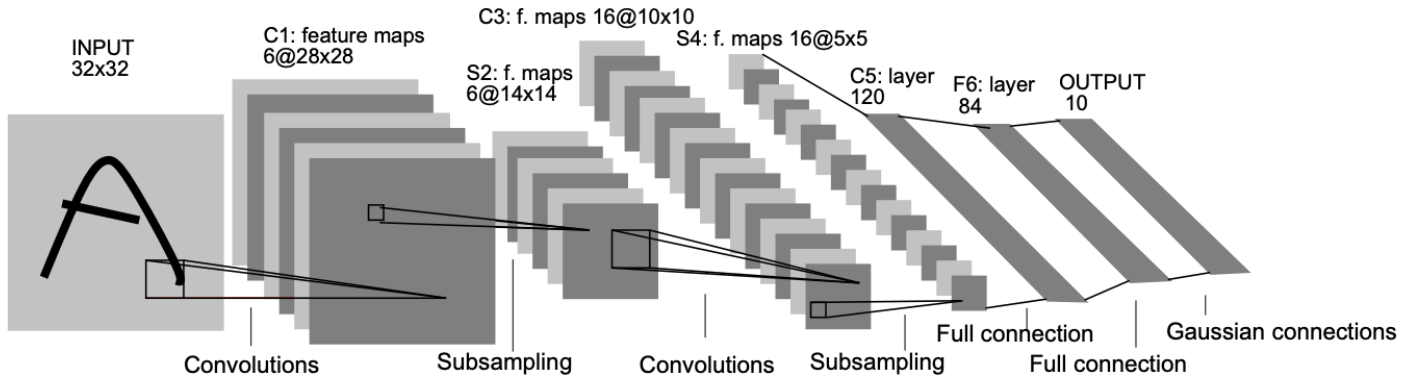


Fig. 2. Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Each plane is a feature map, i.e. a set of units whose weights are constrained to be identical.

A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
- 1986: Back-propagation
- 1989 – 1998: Convolutional neural networks
- 2012: AlexNet

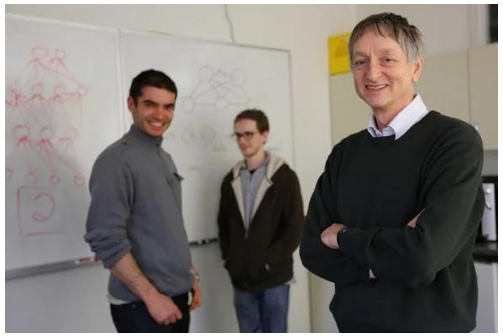
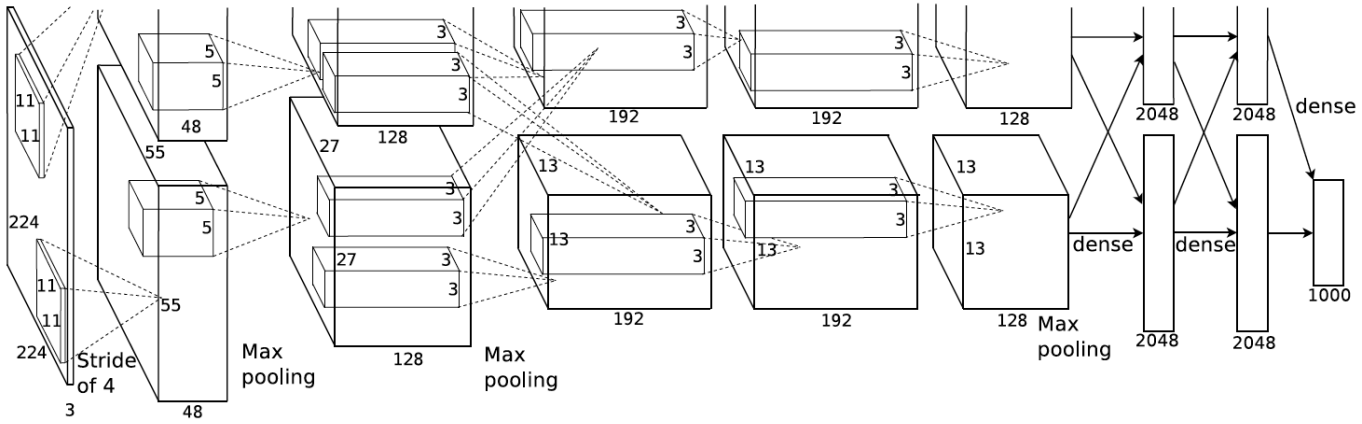
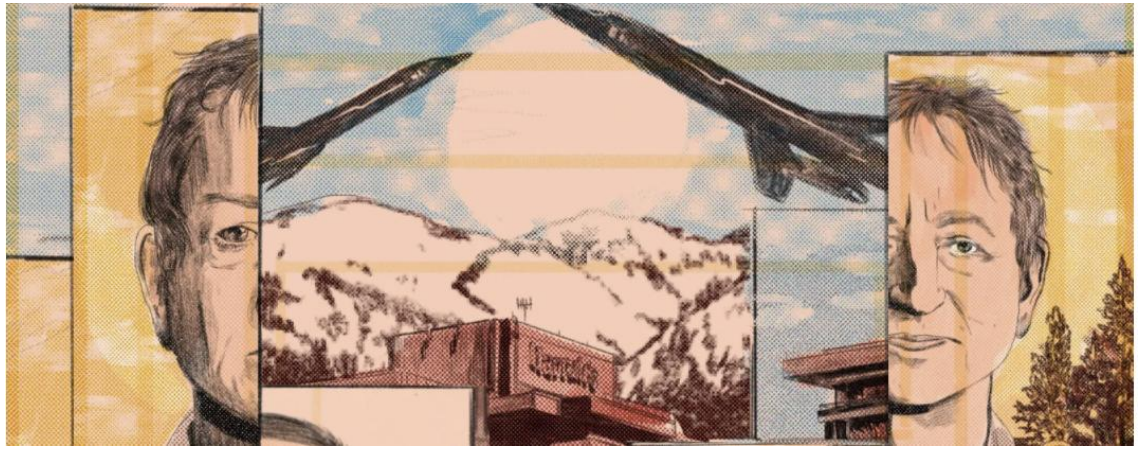


Photo source



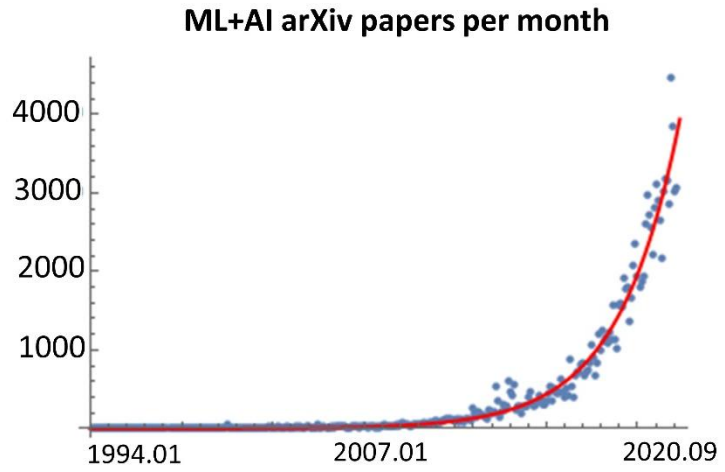
A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
- 1986: Back-propagation
- 1989 – 1998: Convolutional neural networks
- 2012: AlexNet
  - Fascinating reading: The secret auction that set off the race for AI supremacy, Wired, 3/16/2021



A few historical milestones

- 1958: Rosenblatt's perceptron
- 1969: Minsky and Papert Perceptrons book
- 1980: Fukushima's Neocognitron
- 1986: Back-propagation
- 1989 – 1998: Convolutional neural networks
- 2012: AlexNet
- 2012 – present: deep learning explosion



Source, via J. Johnson

The Architectural Evolution of Large Language Models

2017: The Genesis
The Transformer Architecture is Introduced
Merked a paradigm shift from sequential (RNNs, LE7Ms) to parvilal processing.

Key Innovation: Multi-Head Self-Attention
This mechanism allows the model to process all tokens in a sequence simultaneously and weigh the importance of different words, enabling faster training and a deeper understanding of context.

Core Components: Encoders and Decoders
Encoder maps input to a representation; Decoder generates output from it.

2018: The Great Divergence
The First Foundational LLMs Emerge
Two seminal models utilised different Transformer parts, setting distinct paths.

Path 1: BERT (Encoder-Only)
Used only the Transformer's encodes, excelling at language context for classification and question answering.

Path 2: GPT (Decoder-Only)
Used only the decoder, optimizing for generating cofierent, human-like text.

Auto-Regressive (Decoder-Only)
Decoder-contric models optimized for generative tasks. Benoretes output one token at a time, based only on preseding tokens (unidrnoctional).

Auto-Encoding (Encoder-Only)
Primarily encoder-based, toiluzed for contextual understanding and representation learning. Sees entire input at once (Indirectional).

Advantage: Good at learning from context, efficient at representation trarning.
Disadvantage: Not suitable for generating new, long sequences.
Examples: BEOT family, RoRERTa.

Sequence-to-Sequence (Encoder-Decoder)
Advantage: Excellent for conditional generation (e.g., translation, summarization).
Disadvantage: Higher parameter count and more complex training.
Example Models: Pangu family, T5.

Advantage: Suited for generatico facls: effective at language modeling.
Disadvantage: Locks context from future tokens during generation.
Examples: OPT-family, LxaI4G family.

2019-2024: An Era of Innovation and Scale

Encoder-Only Evolution: Refining Understanding

- **RoBERTa** (robust training)
- **ALBERT** (parameter efficiency)
- **ELECTRA** (GAN like training)

Recoder-Only Evolution: The Race to Scale and Capability

- **GPT-2** (scaling op)
- **GPT-3** (175B parameters, in-context learning)
- **PaLM** (efficient ssaling to 550B parameters)
- **LLaMA** (open-source, efficient)

Sequence-to-Sequence Evolution: Unifying Tasks

- **TS** (all NLP tasks as text-to-text)
- **BART** (combining BERT & GPT strengths)

The Rise of Multimodality
Later models like GPF4 and PaLM 6 broke the text-only border, integrating capabilities to process and understand multiple data types.