Statistical Learning II

Lecture 2 - Introduction & supervised learning

Bruno Loureiro
@ CSD, DI-ENS & CNRS

brloureiro@gmail.com

DL3 IASO, Université Paris Dauphine-PSL
17.09.2025

Course Organisation

- 12 classes of 3h, divided in:

  - 1h30 lectures

  - 1h30 exercises & lab (Python)
  (with Leonardo De Filippis)

Course Organisation

- 12 classes of 3h, divided in:

  - 1h30 lectures

  - 1h30 exercises & lab (Python)
  (with Leonardo De Filippis)

- Contact us: via Microsoft Teams or e-mail:

  bruno.loureiro@di.ens.fr and leonardo.defilippis99@gmail.com

Course Organisation

- 12 classes of 3h, divided in:

  - 1h30 lectures

  - 1h30 exercises & lab (Python) (with Leonardo De Filippis)

- Contact us: via Microsoft Teams or e-mail:

  bruno.loureiro@di.ens.fr and leonardo.defilippis99@gmail.com

- Evaluation: Partiel (25%) + Final (75%)

Menu for the semester

Goal: Develop a *mathematical* understanding of *classical* and *modern* machine learning models

Menu for the semester

Goal: Develop a *mathematical* understanding of *classical* and *modern* machine learning models

- Classical methods:
    - Ridge regression
    - LASSO
    - Generalised linear models
    - Kernel methods
    - Principal component analysis (PCA)

Menu for the semester

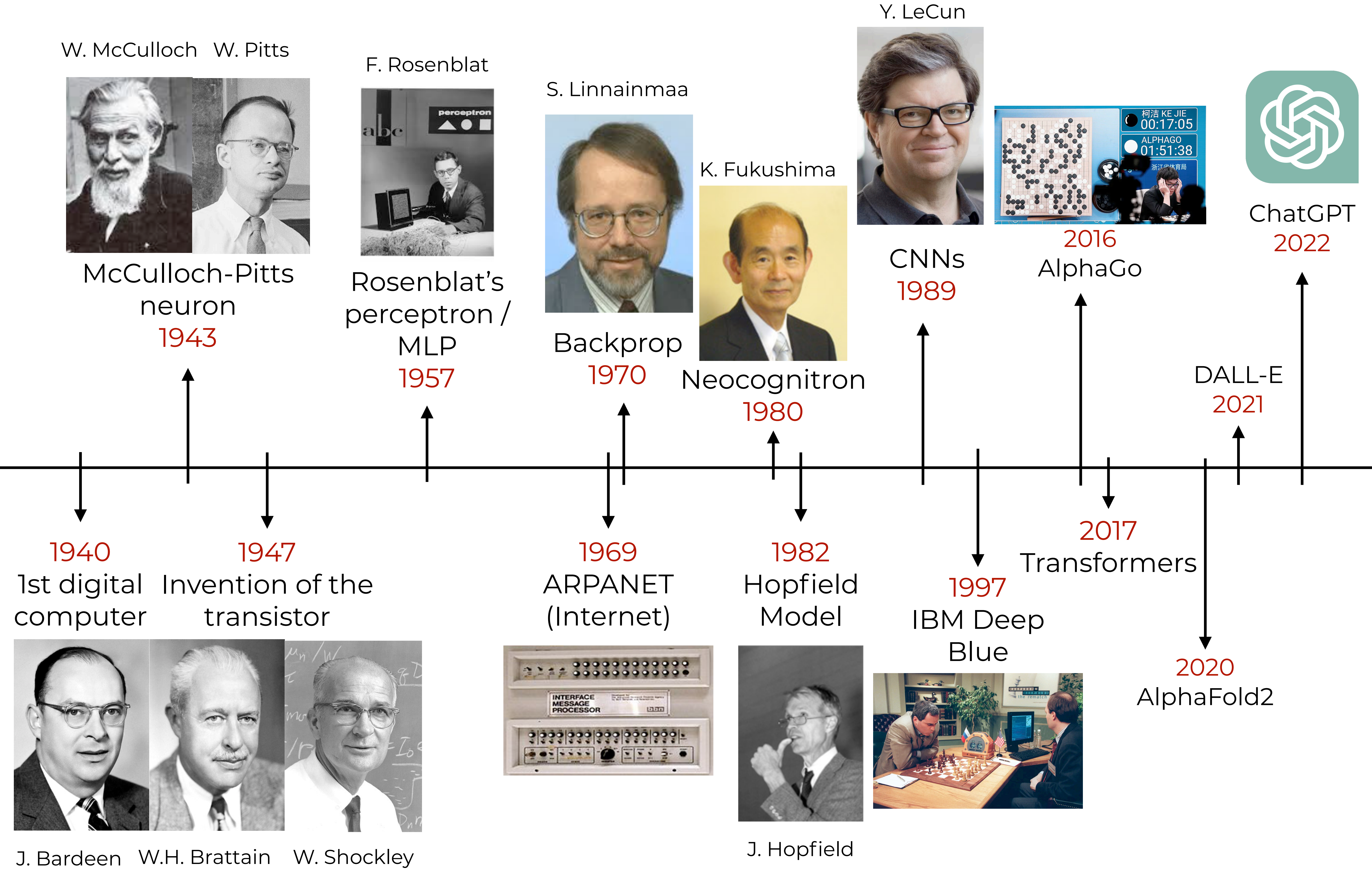
Goal: Develop a *mathematical* understanding of *classical* and *modern* machine learning models

- Classical methods:
    - Ridge regression
    - LASSO
    - Generalised linear models
    - Kernel methods
    - Principal component analysis (PCA)
- Modern methods:
    - Neural networks
    - Diffusion models
    - Your suggestions?

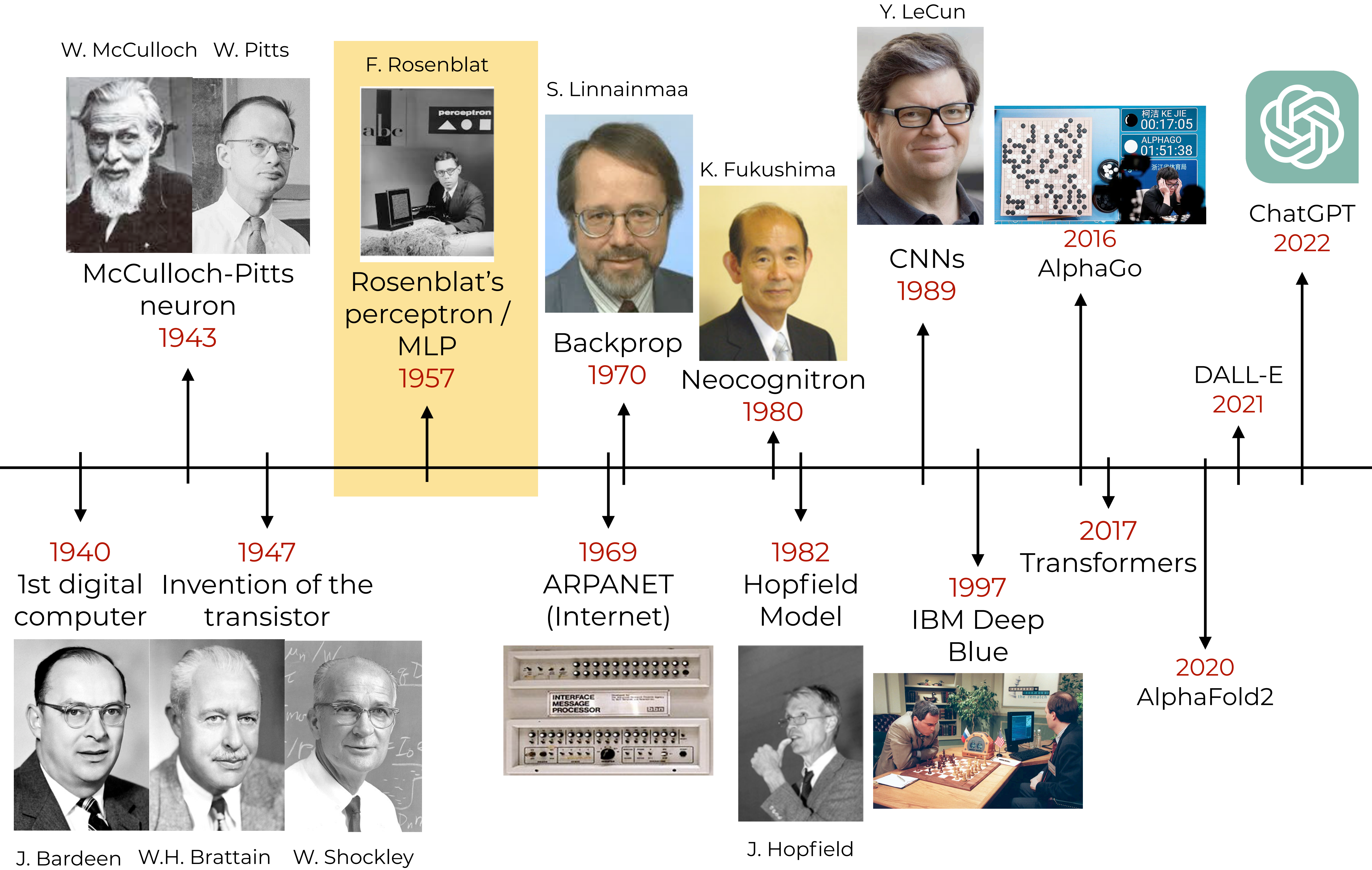
Introduction & Motivation

Or: *why should I care about Statistical Learning?*

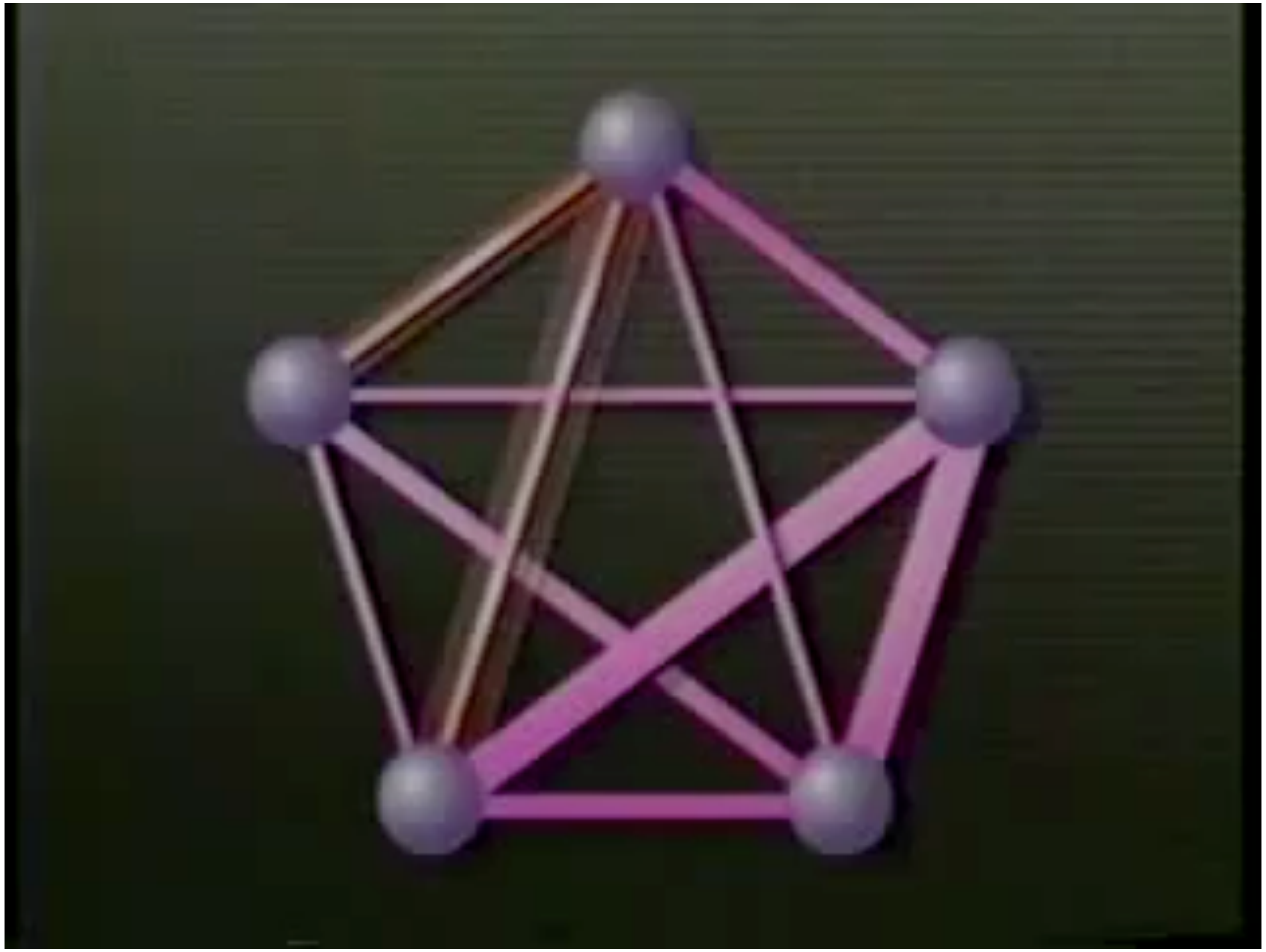
A brief history of ML



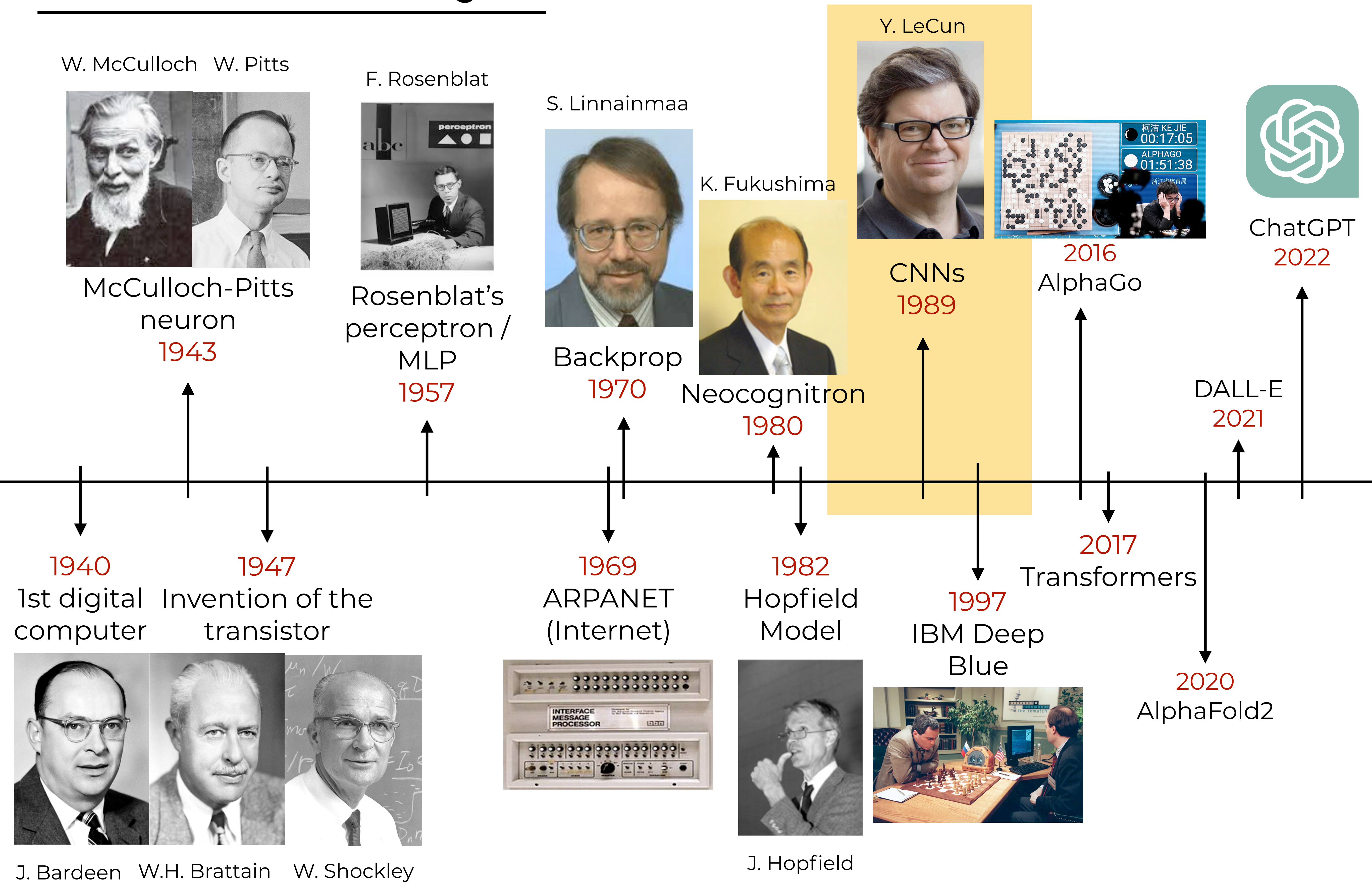
A brief history of ML



A brief history of ML

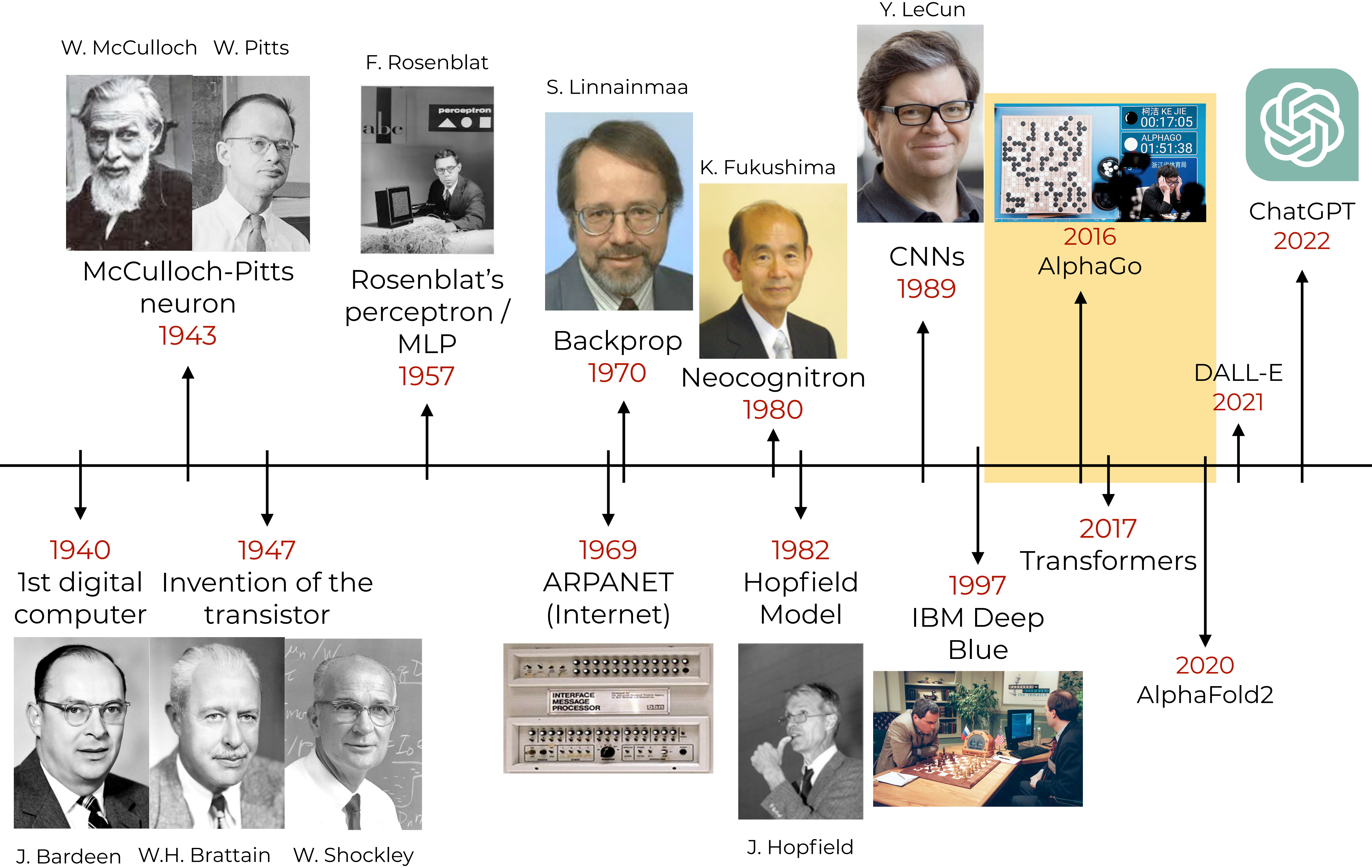


A brief history of ML



A brief history of ML

A brief history of ML



A brief history of ML

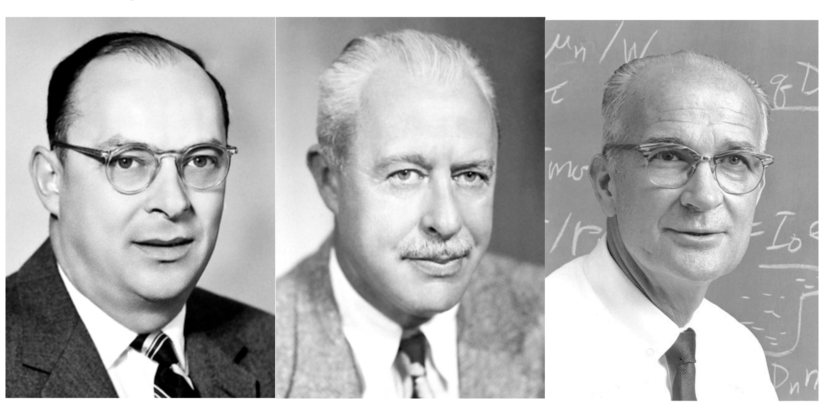
A brief history of ML

1940
1st digital computer

1943
W. McCulloch W. Pitts

McCulloch-Pitts neuron
1943

1947
Invention of the transistor



J. Bardeen W.H. Brattain W. Shockley

F. Rosenblat

Rosenblat's perceptron / MLP
1957

1969
ARPANET (Internet)

S. Linnainmaa

Backprop
1970

K. Fukushima

Neocognitron
1980

1982
Hopfield Model

J. Hopfield

Y. LeCun

CNNs
1989

1997
IBM Deep Blue

2016
AlphaGo

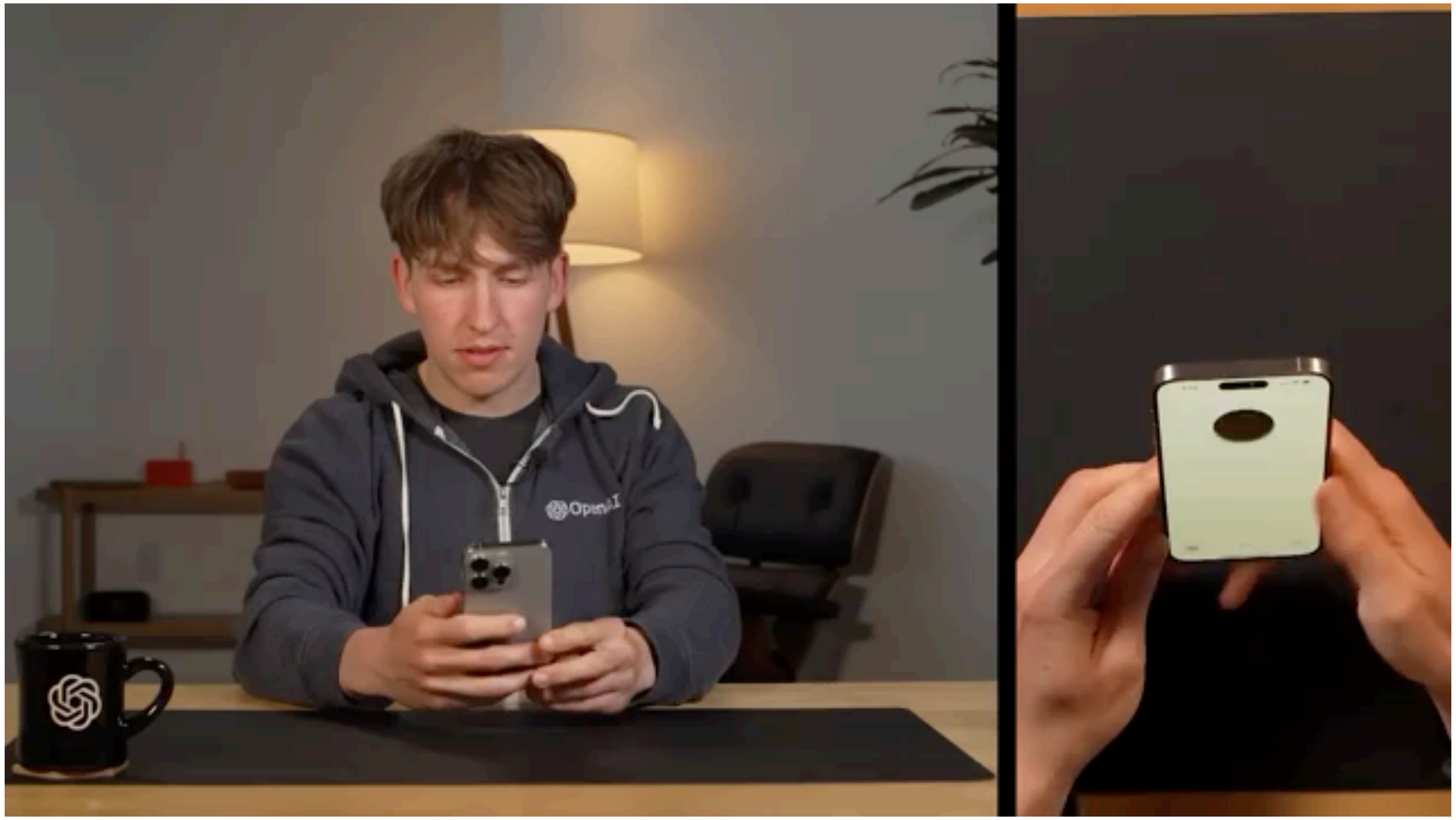
2017
Transformers

2020
AlphaFold2

DALL-E
2021

ChatGPT
2022

A brief history of ML



What about the maths?

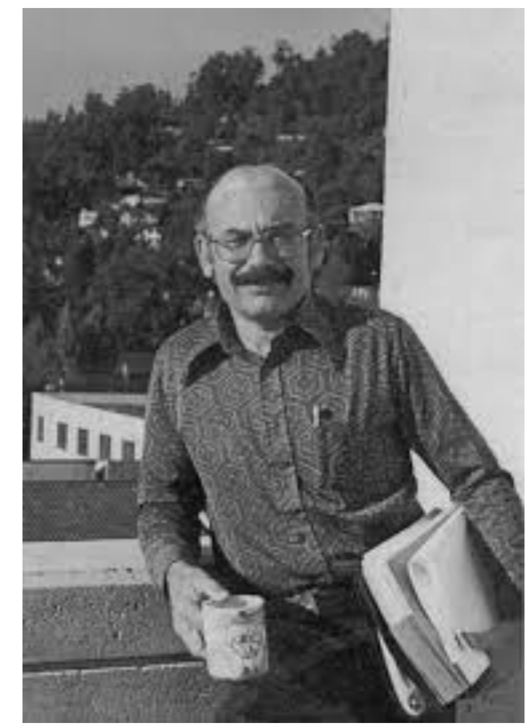
Yet, on the mathematical side...

Leo Breiman
Statistics Department, University of California, Berkeley, CA 94305;
e-mail: leo@stat.berkeley.edu

Reflections After Refereeing Papers for NIPS

For instance, there are many important questions regarding neural networks which are largely unanswered. There seem to be conflicting stories regarding the following issues:

- Why don't heavily parameterized neural networks overfit the data?
- What is the effective number of parameters?
- Why doesn't backpropagation head for a poor local minima?
- When should one stop the backpropagation and use the current parameters?



Leo Breiman
1928

What about the maths?

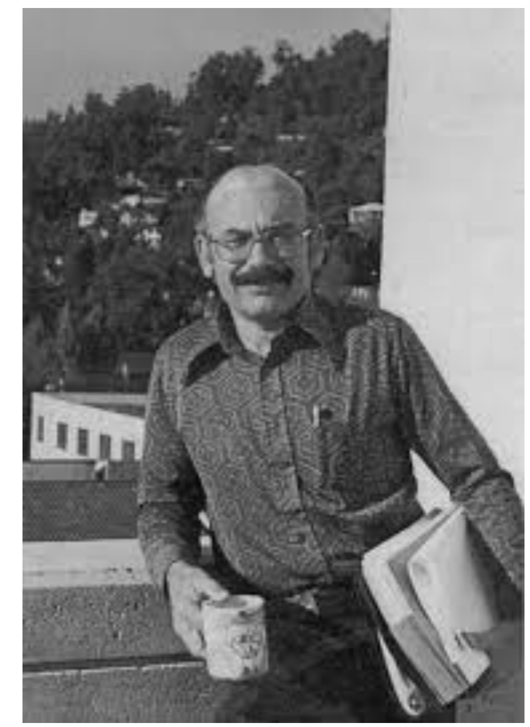
Yet, on the mathematical side...

Leo Breiman
Statistics Department, University of California, Berkeley, CA 94305;
e-mail: leo@stat.berkeley.edu

Reflections After Refereeing Papers for NIPS

For instance, there are many important questions regarding neural networks which are largely unanswered. There seem to be conflicting stories regarding the following issues:

- Why don't heavily parameterized neural networks overfit the data?
- What is the effective number of parameters?
- Why doesn't backpropagation head for a poor local minima?
- When should one stop the backpropagation and use the current parameters?



Leo Breiman
1928

This was written in 1995!!!

What about the maths?

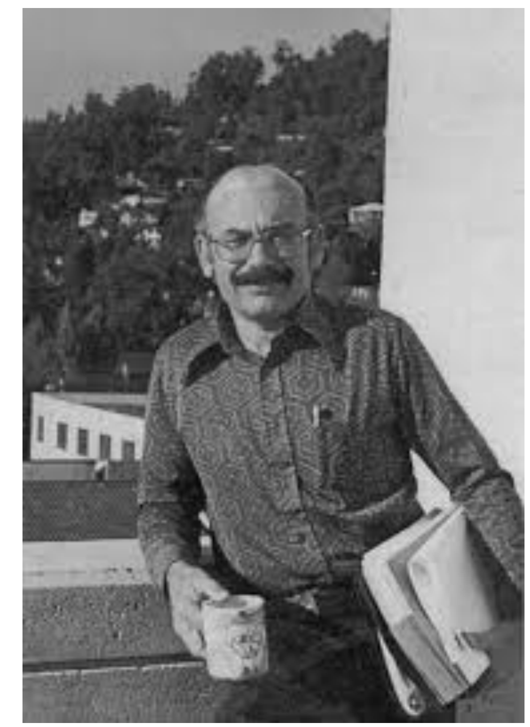
Yet, on the mathematical side...

Leo Breiman
Statistics Department, University of California, Berkeley, CA 94305;
e-mail: leo@stat.berkeley.edu

Reflections After Refereeing Papers for NIPS

For instance, there are many important questions regarding neural networks which are largely unanswered. There seem to be conflicting stories regarding the following issues:

- Why don't heavily parameterized neural networks overfit the data?
- What is the effective number of parameters?
- Why doesn't backpropagation head for a poor local minima?
- When should one stop the backpropagation and use the current parameters?



Leo Breiman
1928

This was written in 1995!!!

But why should I care?

Some good reasons to care

- Reliability and Liability

  If a model does something unexpected, who is responsible?

  Crucial in sensitive applications, e.g. medicine, law, self-driving cars/planes…

Some good reasons to care

- Reliability and Liability

  If a model does something unexpected, who is responsible?

  Crucial in sensitive applications, e.g. medicine, law, self-driving cars/planes…

- Efficient design

  Data centres are responsible for 4% of the energy consumption in the US.

  Can we design models and algorithms that learn more efficiently from data?

Some good reasons to care

- Reliability and Liability

  If a model does something unexpected, who is responsible?

  Crucial in sensitive applications, e.g. medicine, law, self-driving cars/planes...

- Efficient design

  Data centres are responsible for 4% of the energy consumption in the US.

  Can we design models and algorithms that learn more efficiently from data?

- Scientific curiosity

Some good reasons to care

- Reliability and Liability

  If a model does something unexpected, who is responsible?

  Crucial in sensitive applications, e.g. medicine, law, self-driving cars/planes…

- Efficient design

  Data centres are responsible for 4% of the energy consumption in the US.

  Can we design models and algorithms that learn more efficiently from data?

- Scientific curiosity

- And in the worst case, understanding the maths will make you a better engineer / data scientist.

Our expectations

My expectations: By the end of the term, I expect you to:

- Get acquainted with the most popular machine learning algorithms

Our expectations

My expectations: By the end of the term, I expect you to:

- Get acquainted with the most popular machine learning algorithms

- Appreciate (some) of the mathematics behind the methods.

Our expectations

My expectations: By the end of the term, I expect you to:

- Get acquainted with the most popular machine learning algorithms

- Appreciate (some) of the mathematics behind the methods.

- Be able to implement these methods from scratch.

Our expectations

<u>My expectations</u>: By the end of the term, I expect you to:

- Get acquainted with the most popular machine learning algorithms

- Appreciate (some) of the mathematics behind the methods.

- Be able to implement these methods from scratch.

<u>Your expectations:</u> Let’s make a quick survey!

https://forms.gle/nhTyc3pHHiKgTJg9A

99% of books in statistical learning:

Let $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1, \ldots, n$,
denote independently drawn samples from a probability distribution….

99% of books in statistical learning:

Let $(\boldsymbol{x}_i, y_i) \in \mathbb{R}^d \times \mathbb{R}$, $i = 1,\ldots,n$,
denote independently drawn samples from a probability distribution....

Supervised learning

Supervised Learning

Not grumpy

Grumpy

Supervised Learning

Not grumpy

Grumpy

???

Supervised Learning

Inputs / covariates $x \in \mathcal{X}$

x_1

x_2

x_3

y_1
Not grumpy

y_2
Grumpy

y_3
???

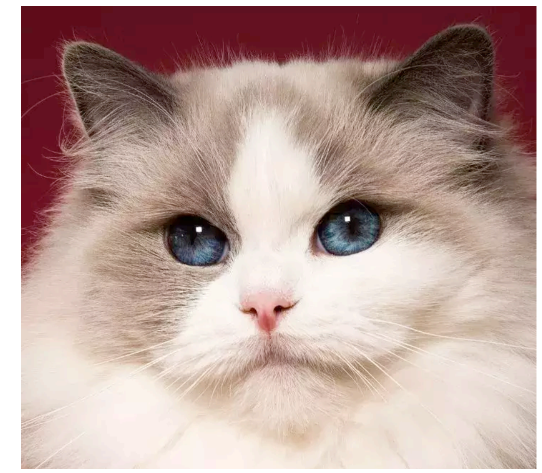
Outputs / Labels / response $y \in \mathcal{Y}$

Supervised Learning

Inputs / covariates $x \in \mathcal{X}$

x_1

x_2



x_3

y_1
Not grumpy

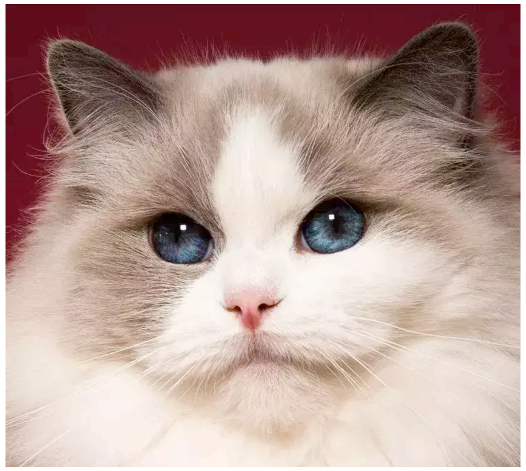
y_2
Grumpy

y_3
???

Outputs / Labels / response $y \in \mathcal{Y}$

Supervised Learning

Inputs / covariates $x \in \mathscr{X}$



x_1 x_2 x_3

$f : \mathscr{X} \to \mathscr{Y}$

y_1 Not grumpy

y_2 Grumpy

y_3 ???

Outputs / Labels / response $y \in \mathscr{Y}$

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

- The input space $\mathscr{X}$ is often assumed to be a vector space $\mathscr{X} \subset \mathbb{R}^d$. But keep in mind in real life it can be any data structure (e.g. a `pandas.DataFrame`)

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

- The input space $\mathscr{X}$ is often assumed to be a vector space $\mathscr{X} \subset \mathbb{R}^d$. But keep in mind in real life it can be any data structure (e.g. a pandas.DataFrame)

- The output space $\mathscr{Y}$ is often assumed to be a subset $\mathscr{Y} \subset \mathbb{R}$.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

- The input space $\mathscr{X}$ is often assumed to be a vector space $\mathscr{X} \subset \mathbb{R}^d$. But keep in mind in real life it can be any data structure (e.g. a pandas.DataFrame)
- The output space $\mathscr{Y}$ is often assumed to be a subset $\mathscr{Y} \subset \mathbb{R}$.
- In particular, if $|\mathscr{Y}| = k$ is a discrete set, we say we have a classification problem.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

- The input space $\mathscr{X}$ is often assumed to be a vector space $\mathscr{X} \subset \mathbb{R}^d$. But keep in mind in real life it can be any data structure (e.g. a pandas.DataFrame)
- The output space $\mathscr{Y}$ is often assumed to be a subset $\mathscr{Y} \subset \mathbb{R}$.
- In particular, if $|\mathscr{Y}| = k$ is a discrete set, we say we have a classification problem.
- If $\mathscr{Y}$ is a continuous set, we say we have a regression problem

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

- The input space $\mathscr{X}$ is often assumed to be a vector space $\mathscr{X} \subset \mathbb{R}^d$. But keep in mind in real life it can be any data structure (e.g. a pandas.DataFrame)

- The output space $\mathscr{Y}$ is often assumed to be a subset $\mathscr{Y} \subset \mathbb{R}$.

- In particular, if $|\mathscr{Y}| = k$ is a discrete set, we say we have a classification problem.

- If $\mathscr{Y}$ is a continuous set, we say we have a regression problem

It is very common to consider a one-hot encoding $\mathscr{Y} = \{0,1\}^k$ in classification.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

Examples of classification:

- Grumpy vs. Non-grumpy cats

$\mathscr{X} = \{\text{photos of cats}\}$, $\mathscr{Y} = \{\text{grumpy}, \text{not grumpy}\}$

- E-mail spam detection

$\mathscr{X} = \{\text{your inbox}\}$, $\mathscr{Y} = \{\text{spam}, \text{not spam}\}$

- Medical diagnosis

$\mathscr{X} = \{\text{medical data}\}$, $\mathscr{Y} = \{\text{diseases}\}$

- Sentiment analysis

$\mathscr{X} = \{\text{text}\}$, $\mathscr{Y} = \{\text{positive}, \text{negative}, \text{neutral}\}$

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

Examples of regression:

- Temperature prediction
$\mathscr{X} = \mathbb{R}^3, \mathscr{Y} = \mathbb{R}$

- Stock price prediction
$\mathscr{X} = \{\text{list of stocks}\}, \mathscr{Y} = \mathbb{R}_+$

- Life expectancy
$\mathscr{X} = \{\text{medical data}\}, \mathscr{Y} = \mathbb{R}_+$

- Any price, cost, income, etc. prediction.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

In supervised learning, our goal is to use the data to learn a function that correctly assigns the labels to the responses.

$$\boxed{f : \mathscr{X} \to \mathscr{Y}}$$

Supervised Learning

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

In supervised learning, our goal is to use the data to learn a function that correctly assigns the labels to the responses.

$$f : \mathscr{X} \to \mathscr{Y}$$

For classification, it is common to define instead:

$$f : \mathscr{X} \to [0,1]^{|\mathscr{Y}|}$$

Where $f(x)$ is a vector of class probabilities. In this case, final prediction is given by:

$$\hat{y} = \underset{k \in |\mathscr{Y}|}{\operatorname{argmax}} f(x)$$

Supervised Learning

Let $\mathcal{D} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y} : i = 1,\ldots,n\}$ denote the training data.

In supervised learning, our goal is to use the data to learn a function that correctly assigns the labels to the responses.

$$\boxed{f : \mathcal{X} \to \mathcal{Y}}$$

Two key words: correctly and learn.

Loss function

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

In supervised learning, our goal is to use the data to learn a function that correctly assigns the labels to the responses.

$$f : \mathscr{X} \to \mathscr{Y}$$

Two key words: correctly and learn. To quantify the first, it is common to introduce a loss function:

$$\ell : \mathscr{Y} \times \mathscr{Y} \to \mathbb{R}_+$$

Loss function

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1, \ldots, n\}$ denote the training data.

In supervised learning, our goal is to use the data to learn a function that correctly assigns the labels to the responses.

$$f : \mathscr{X} \to \mathscr{Y}$$

Two key words: correctly and learn. To quantify the first, it is common to introduce a loss function:

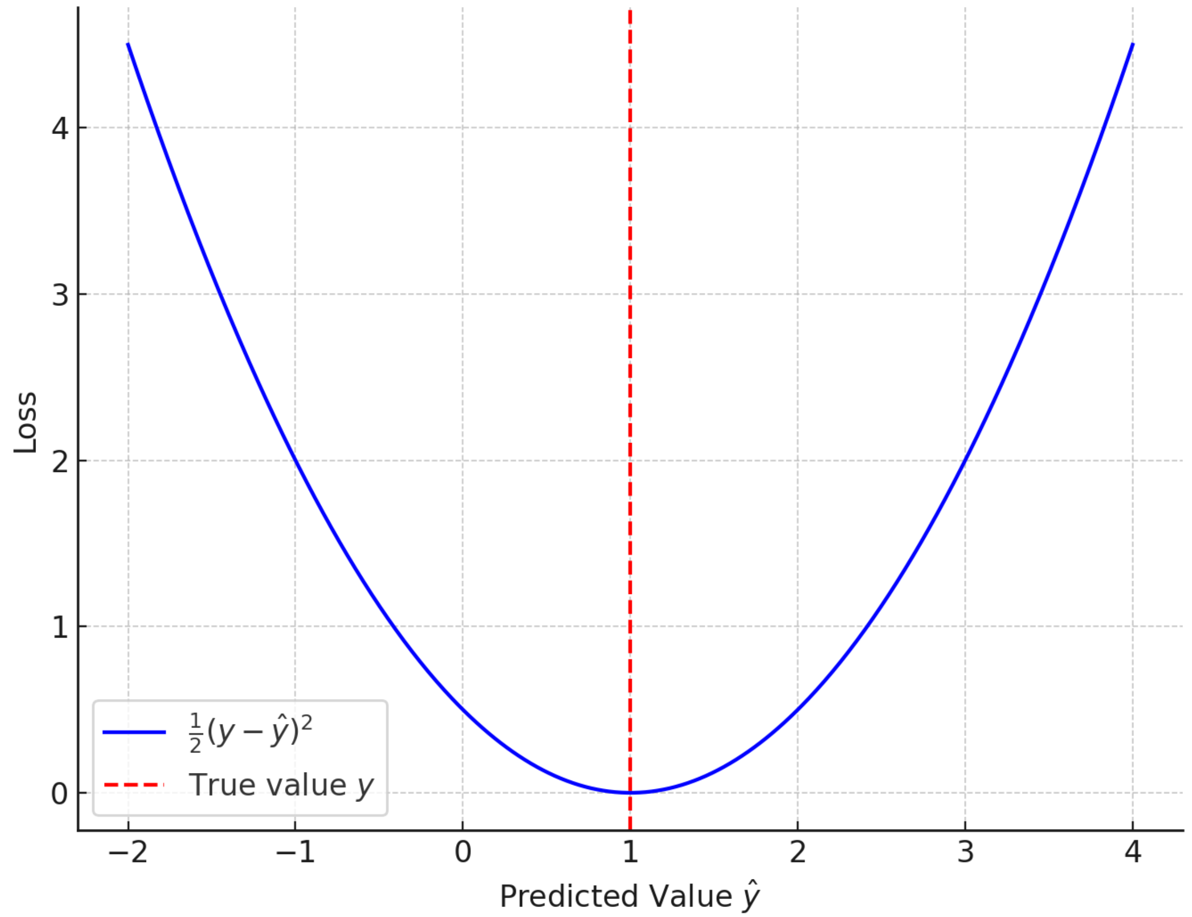
$$\ell : \mathscr{Y} \times \mathscr{Y} \to \mathbb{R}_+$$

For classification this will also depend on the encoding.

Regression losses

Examples in regression:

- Square loss: $\ell(y, z) = \frac{1}{2}(y - z)^2$



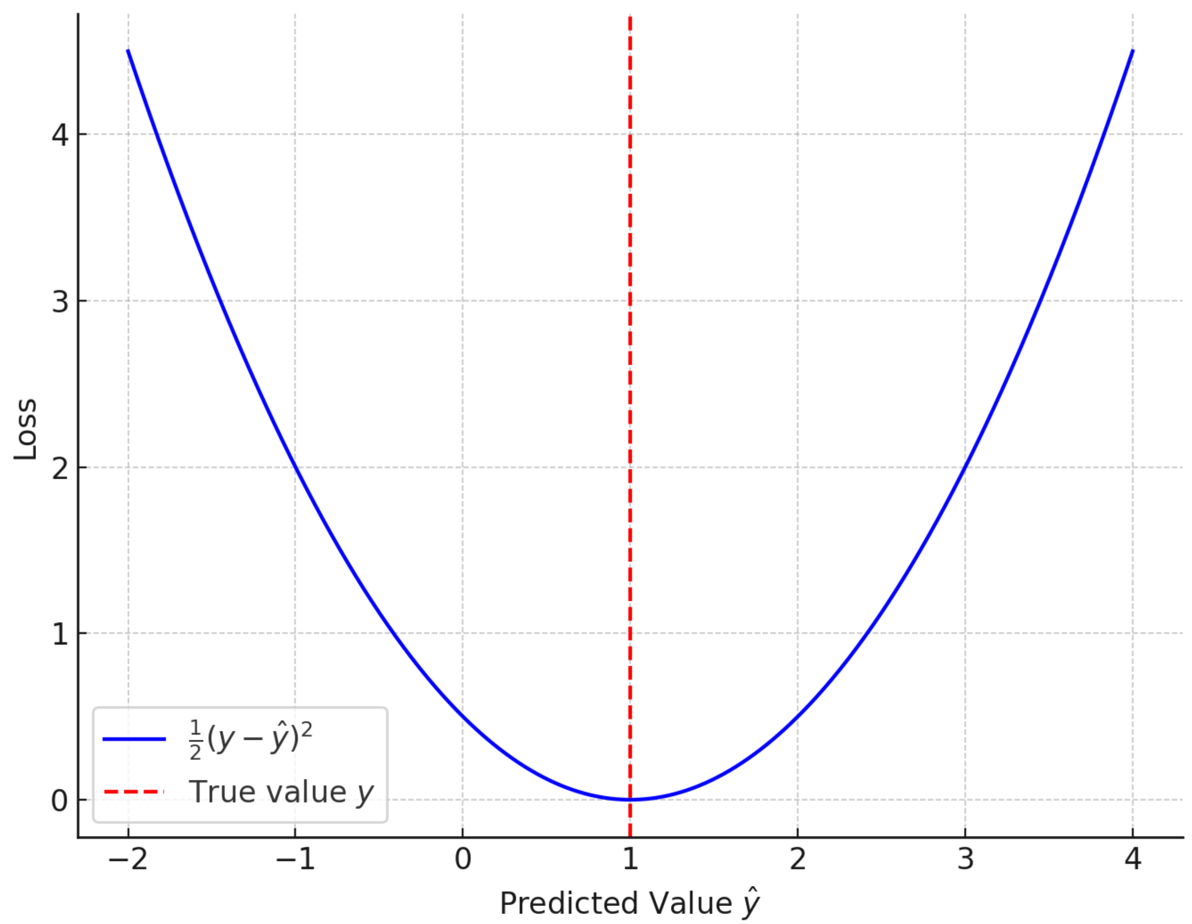
Regression losses

Examples in regression:

- Square loss: $\ell(y, z) = \dfrac{1}{2}(y - z)^2$

The square loss is sensitive to outliers

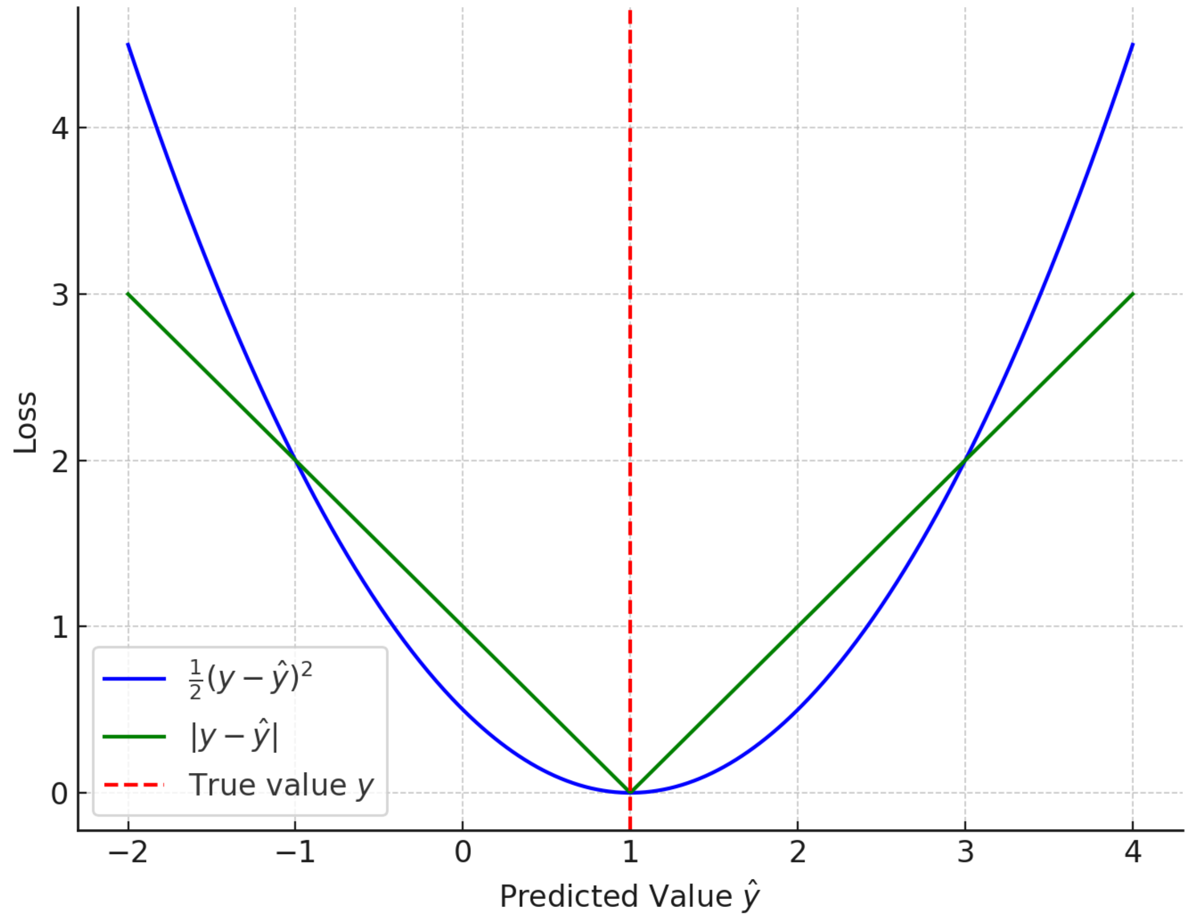
Exercise: show this.



Regression losses

Examples in regression:

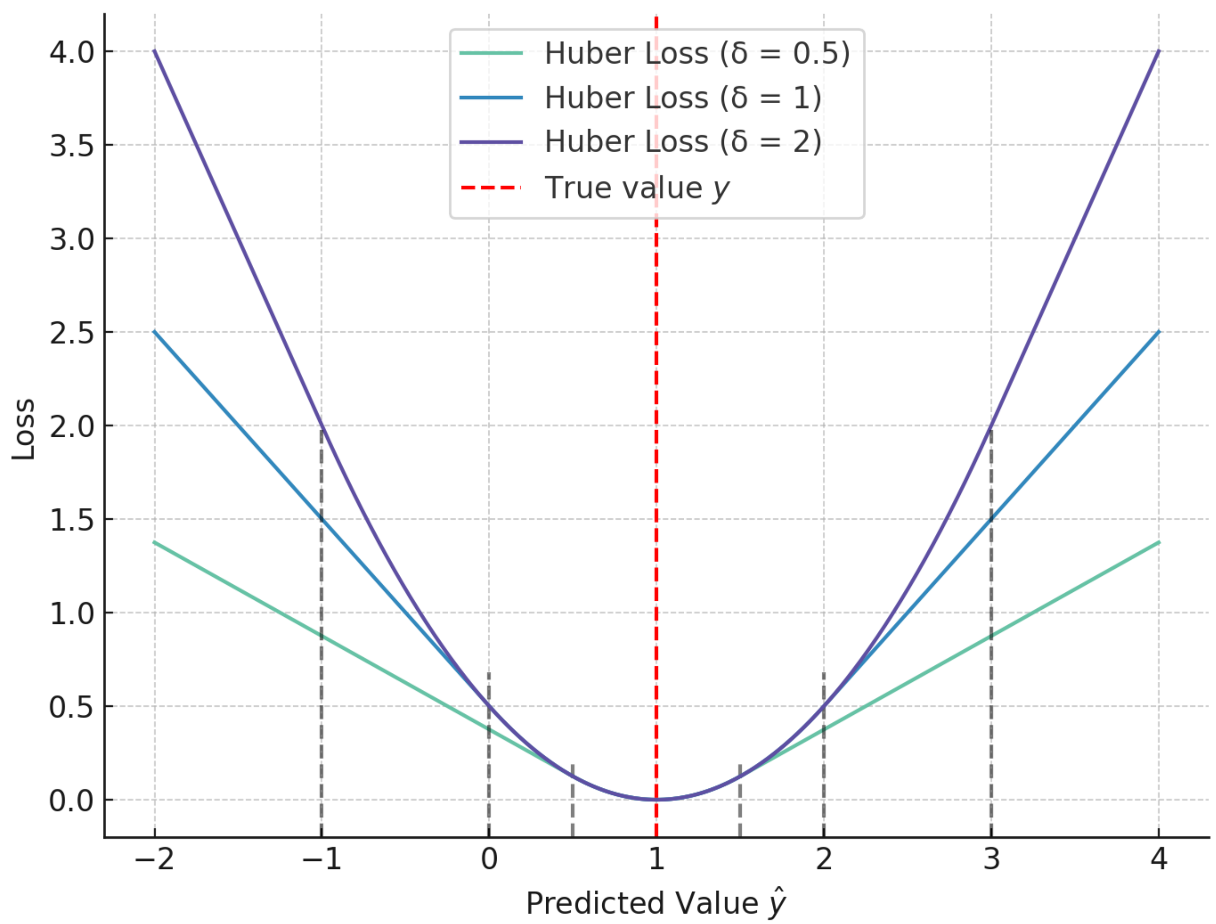
- Square loss: $\ell(y, z) = \dfrac{1}{2}(y - z)^2$
- Absolute loss: $\ell(y, z) = |y - z|$



Regression losses

Examples in regression:

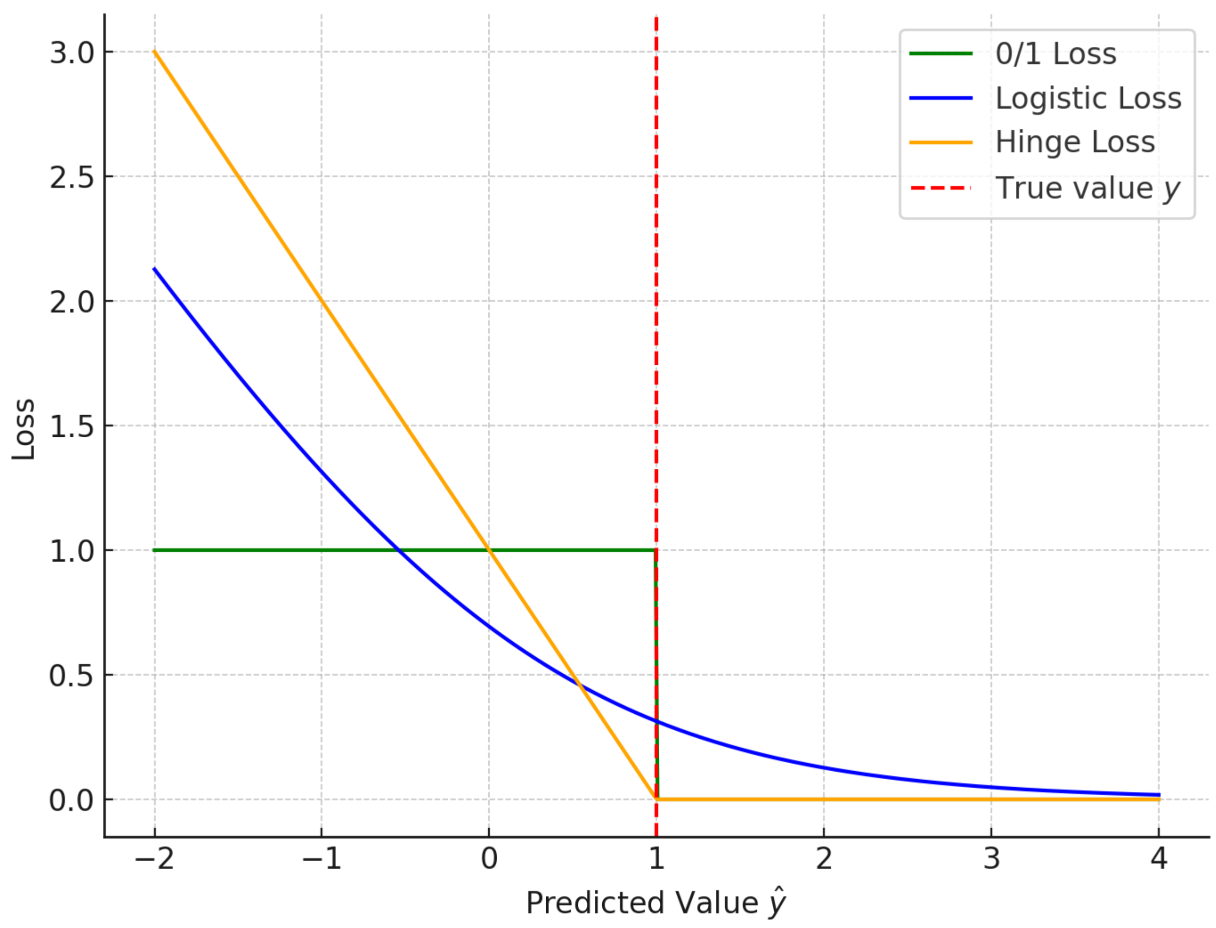
- Huber loss: $\ell_\delta(y,z) = \begin{cases} \frac{1}{2}(y-z)^2 & \text{if } ||y-z| \leq \delta \\ \delta(|y-z| - \frac{1}{2}\delta) & \text{if } |y-z| > \delta \end{cases}$



Classification losses

Examples in binary classification $\mathcal{Y} = \{-1, +1\}$:

- 0/1 loss: $\ell(y, z) = \delta_{yz}$ (or $\ell(y,z) = \theta(y - z) = \begin{cases} 1 & \text{if } y - z \le 0 \\ 0 & \text{otherwise} \end{cases}$)
- Logistic loss: $\ell(y, z) = \log(1 + e^{-yz})$
- Hinge loss: $\ell(y, z) = \max(0, 1 - yz)$



Empirical risk

Let $\mathscr{D} = \{(x_i, y_i) \in \mathscr{X} \times \mathscr{Y} : i = 1,\ldots,n\}$ denote the training data.

Given a loss function $\ell : \mathscr{Y} \times \mathscr{Y} \to \mathbb{R}_+$, and a predictor $f : \mathscr{X} \to \mathscr{Y}$ define the empirical risk:

$$\hat{\mathscr{R}}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i))$$

Also known as the training loss.

Empirical risk

Let $\mathcal{D} = \{(x_i, y_i) \in \mathcal{X} \times \mathcal{Y} : i = 1,\ldots,n\}$ denote the training data.

Given a loss function $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$, and a predictor $f : \mathcal{X} \to \mathcal{Y}$ define the empirical risk:

$$\hat{\mathcal{R}}_n(f) = \frac{1}{n}\sum_{i=1}^{n} \ell(y_i, f(x_i))$$

Also known as the training loss. This quantifies how well we fit the data. But is this a good notion of learning?

$$f(x) = \begin{cases} y_i & \text{if } x \in \mathcal{D} \\ 0 & \text{otherwise} \end{cases} \quad \Rightarrow \quad \hat{\mathcal{R}}_n = 0$$

Probabilistic framework

Instead, we would like predictors that do well on unseen data.

Probabilistic framework

Instead, we would like predictors that do well on unseen data.

Assume there is an underlying data distribution p over $\mathcal{X} \times \mathcal{Y}$:

$$(x_i, y_i) \sim p \qquad \text{i.i.d.}$$

Probabilistic framework

Instead, we would like predictors that do well on unseen data.

Assume there is an underlying data distribution p over $\mathscr{X} \times \mathscr{Y}$:

$$(x_i, y_i) \sim p \qquad \text{i.i.d.}$$

- The "i.i.d." assumption might not always hold. (Sampling bias, distribution shift, etc.)

- Under this assumption, $\hat{\mathscr{R}}_n$ is a random function.

Population risk

Instead, we would like predictors that do well on unseen data.

Assume there is an underlying data distribution p over $\mathcal{X} \times \mathcal{Y}$:

$$(x_i, y_i) \sim p \qquad \text{i.i.d.}$$

Define the notion of population risk of a predictor $f : \mathcal{X} \to \mathcal{Y}$:

$$\mathcal{R}(f) = \mathbb{E}\left[\ell(y, f(x))\right]$$

Also known as the generalisation or test error.

Population risk

Instead, we would like predictors that do well on unseen data.

Assume there is an underlying data distribution p over $\mathcal{X} \times \mathcal{Y}$:

$$(x_i, y_i) \sim p \qquad \text{i.i.d.}$$

Define the notion of population risk of a predictor $f : \mathcal{X} \to \mathcal{Y}$:

$$\mathcal{R}(f) = \mathbb{E}\left[\ell(y, f(x))\right]$$

Also known as the generalisation or test error.

$\mathcal{R}$ is a deterministic function of the predictor f

Validation set

In practice, the statistician almost never has access to the data distribution.

A common procedure to estimate $\mathscr{R}$ consists of splitting the training data in training and validation set $\mathscr{D} = \mathscr{D}_{\mathrm{T}} \cup \mathscr{D}_{\mathrm{V}}$.

Train on $\mathscr{D}_{\mathrm{T}}$, test on $\mathscr{D}_{\mathrm{V}}$.

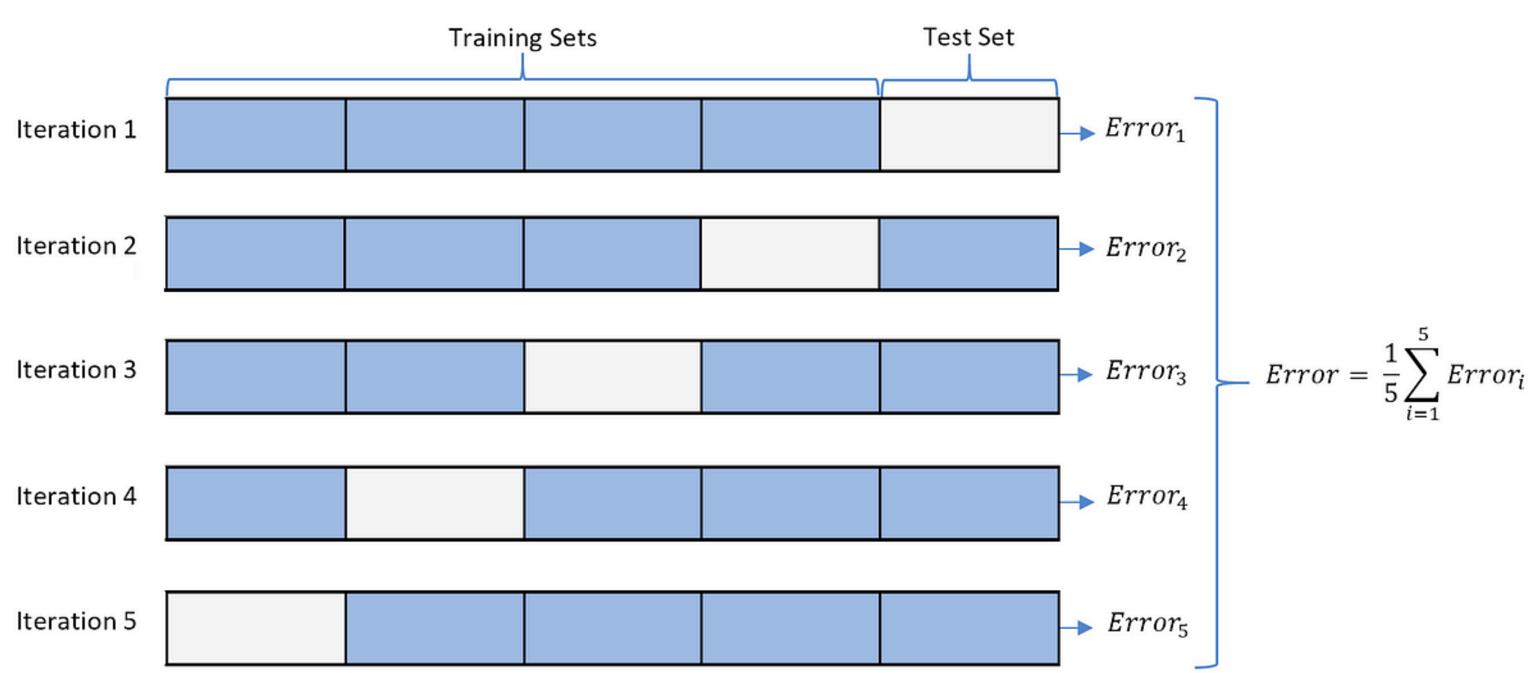
Validation set

In practice, the statistician almost never has access to the data distribution.

A common procedure to estimate $\mathscr{R}$ consists of splitting the training data in training and validation set $\mathscr{D} = \mathscr{D}_{\mathrm{T}} \cup \mathscr{D}_{\mathrm{V}}$.

Train on $\mathscr{D}_{\mathrm{T}}$, test on $\mathscr{D}_{\mathrm{V}}$.

To reduce error, often one repeats this procedure k times, averaging over the result. This is known as k fold cross-validation.



Conditional risk

Given the data distribution p and a loss function ℓ, we can decompose the risk:

$$\mathscr{R}(f) = \mathbb{E}_{(X,Y)\sim p}[\ell(Y, f(X))]$$

Conditional risk

Given the data distribution p and a loss function ℓ, we can decompose the risk:

$$\begin{aligned}\mathscr{R}(f) &= \mathbb{E}_{(X,Y)\sim p}[\ell(Y, f(X))] \\ &= \mathbb{E}_{X\sim p_x}\left[\mathbb{E}[\ell(Y, f(x)) \,|\, X = x]\right]\end{aligned}$$

The internal expectation is over the conditional distribution $Y | X = x$

Conditional risk

Given the data distribution p and a loss function ℓ, we can decompose the risk:

$$\mathscr{R}(f) = \mathbb{E}_{(X,Y)\sim p}[\ell(Y, f(X))]$$

$$= \mathbb{E}_{X\sim p_x}\left[\mathbb{E}[\ell(Y, f(x)) \,|\, X = x]\right]$$

"Conditional risk"

Conditional risk

Given the data distribution p and a loss function ℓ, we can decompose the risk:

$$\mathcal{R}(f) = \mathbb{E}_{(X,Y)\sim p}[\ell(Y, f(X))]$$

$$= \mathbb{E}_{X\sim p_x}\left[r(z \mid X)\right] \qquad z = f(x) \in \mathcal{Y}$$

Where we have defined:

$$\boxed{r(z \mid x) = \mathbb{E}[\ell(Y, z) \mid X = x]}$$

"Conditional risk"

Bayes risk

The Bayes predictor is the best achievable predictor:

$$f_\star(x) \in \operatorname*{argmin}_{z \in \mathcal{Y}} r(z \mid x)$$

And is also known as the target function.

Bayes risk

The Bayes predictor is the best achievable predictor:

$$f_\star(x) \in \underset{z \in \mathcal{Y}}{\operatorname{argmin}} \; r(z \,|\, x)$$

And is also known as the target function.

Similarly, the Bayes risk is the best achievable risk:

$$\mathcal{R}_\star = \mathbb{E}_{X \sim p_x} \left[\inf_{z \in \mathcal{Y}} \; r(z \,|\, X) \right]$$

Bayes risk

The Bayes predictor is the best achievable predictor:

$$f_\star(x) \in \underset{z \in \mathcal{Y}}{\operatorname{argmin}} \; r(z \,|\, x)$$

And is also known as the target function.

Similarly, the Bayes risk is the best achievable risk:

$$\mathscr{R}_\star = \mathbb{E}_{X \sim p_x}\left[\inf_{z \in \mathcal{Y}} \; r(z \,|\, X)\right]$$

- The Bayes predictor $f_\star$ might not be unique.
- Typically we have $\mathscr{R}_\star \neq 0$.   Examples in the TD