Large Models

CSC2541H1 Topics in Machine Learning, Winter 2025, UToronto

Chris J. Maddison

Agenda

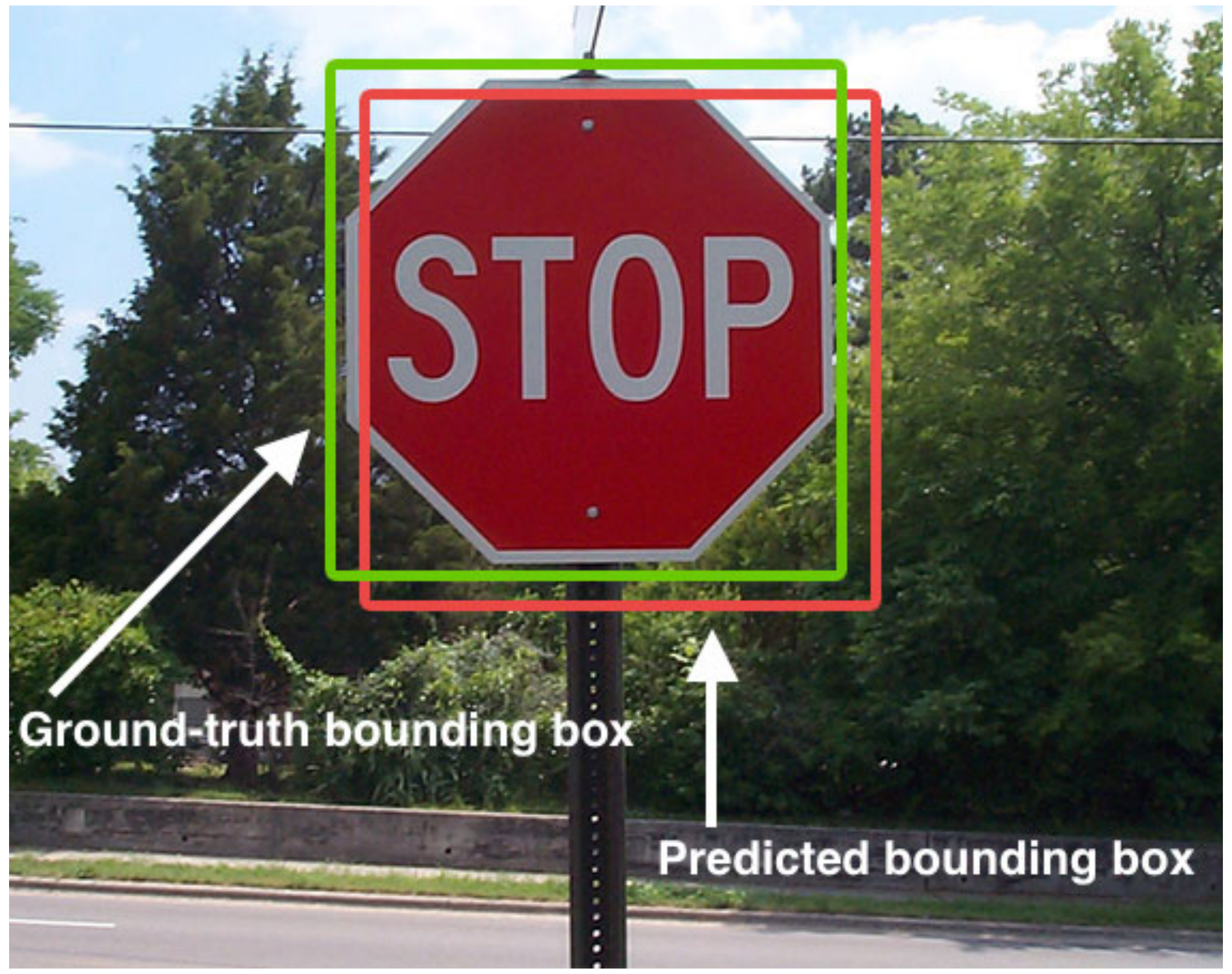
- Course Introduction
- The Story of A Single Bit
- Prediction, Learning, Conditional Prediction
- Large Models and Bitter Lessons

Course Introduction

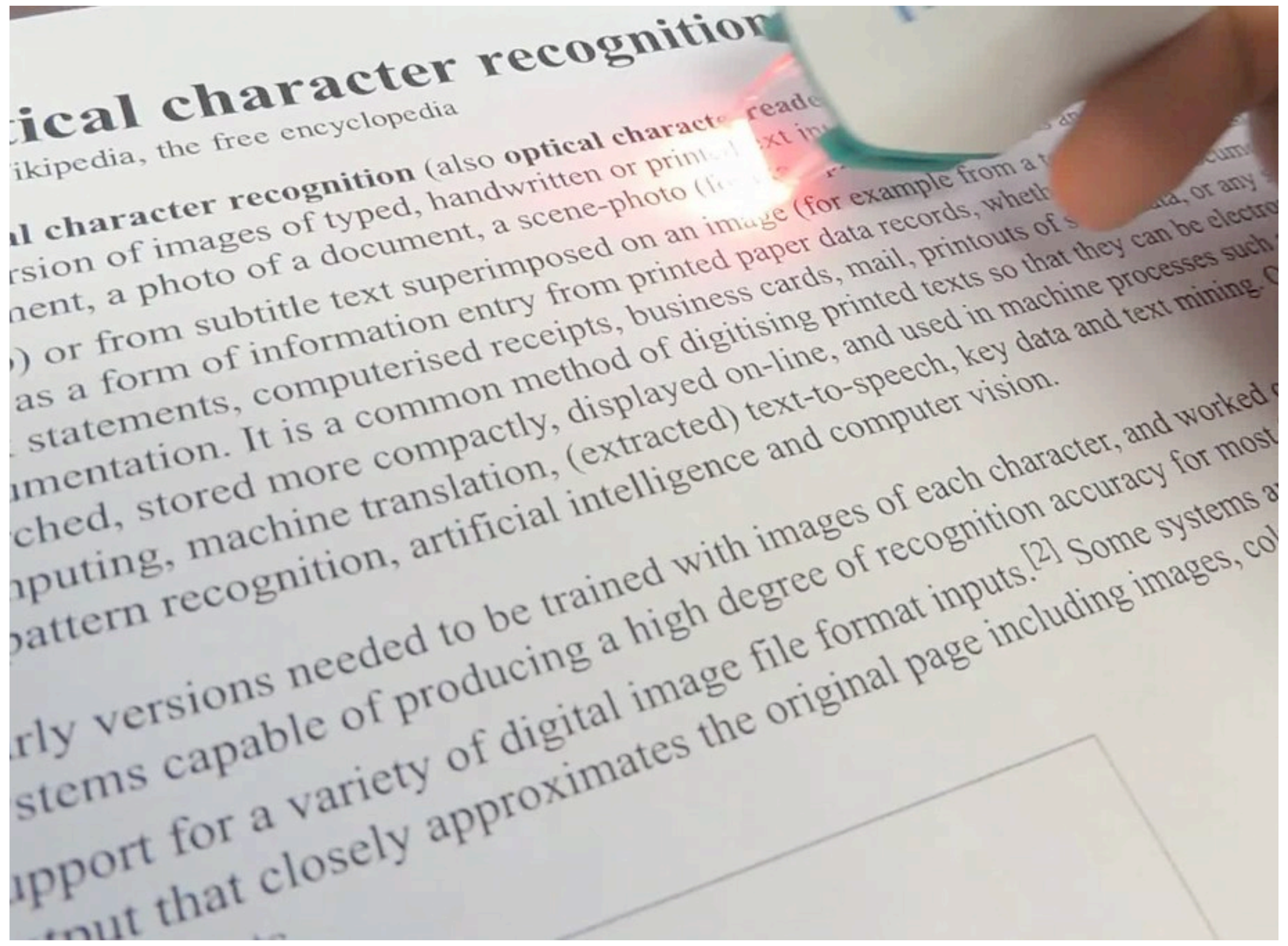
Machine learning: algorithms for prediction

For example,

sign identification



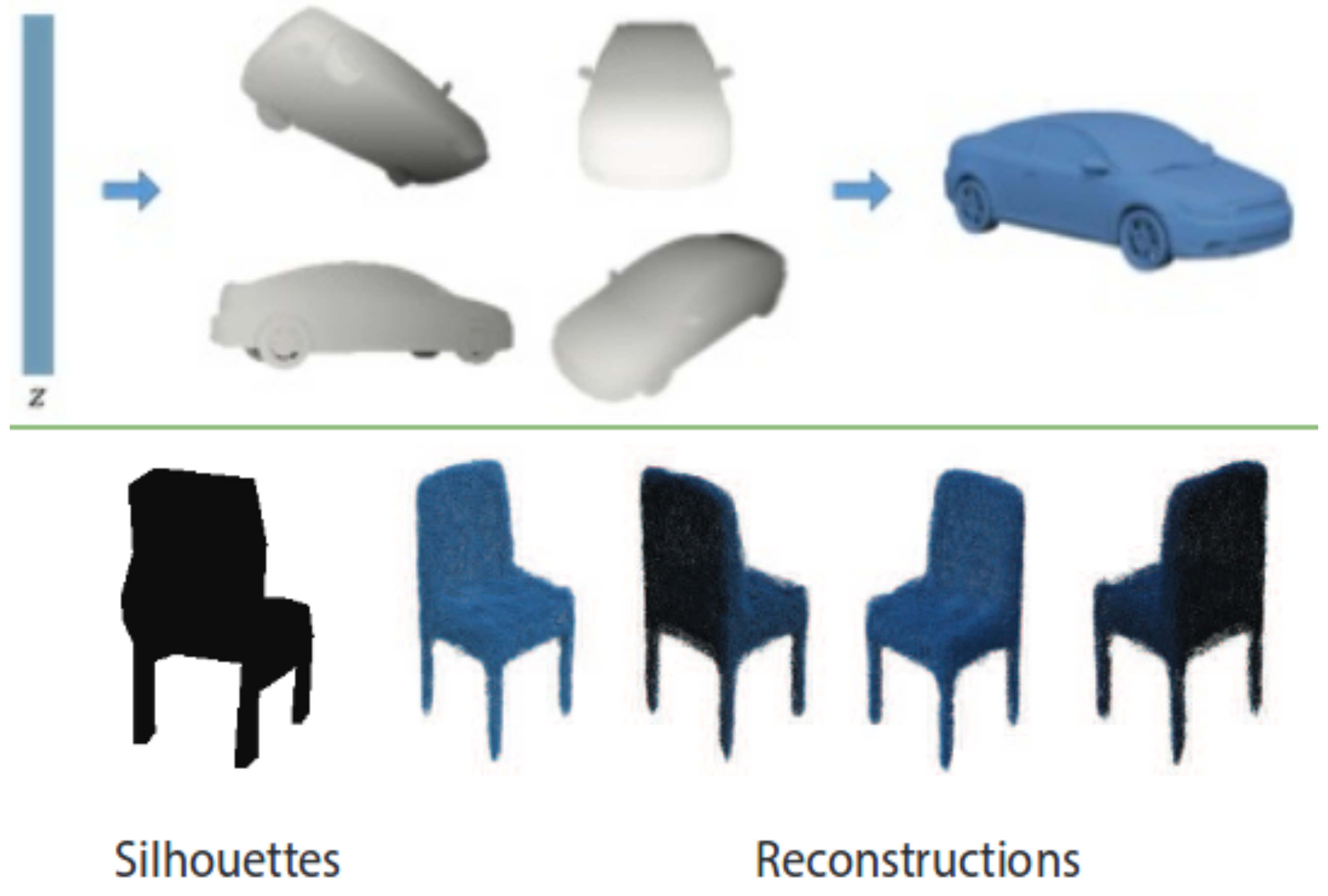
optical character recognition



ical character recognition
ikipedia, the free encyclopedia
l **character recognition** (also **optical charact** reade
rsion of images of typed, handwritten or print ext in
nent, a photo of a document, a scene-photo (f
) or from subtitle text superimposed on an image (for example from a t
as a form of information entry from printed paper data records, wheth ta, or any
statements, computerised receipts, business cards, mail, printouts of s
umentation. It is a common method of digitising printed texts so that they can be electro
ched, stored more compactly, displayed on-line, and used in machine processes such
puting, machine translation, (extracted) text-to-speech, key data and text mining. C
pattern recognition, artificial intelligence and computer vision.

rly versions needed to be trained with images of each character, and worked o
stems capable of producing a high degree of recognition accuracy for most
pport for a variety of digital image file format inputs.[2] Some systems a
tput that closely approximates the original page including images, co

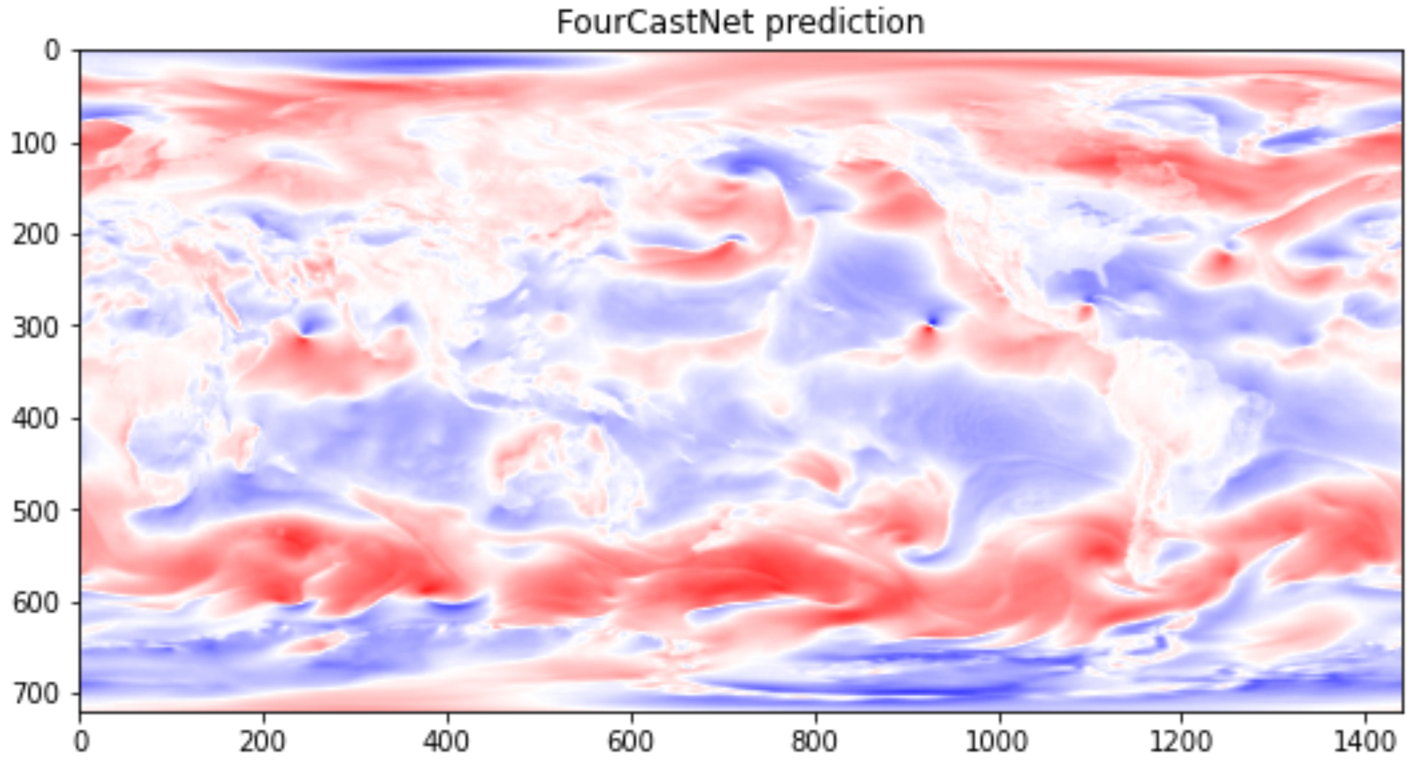
3d reconstructions



What is at stake?

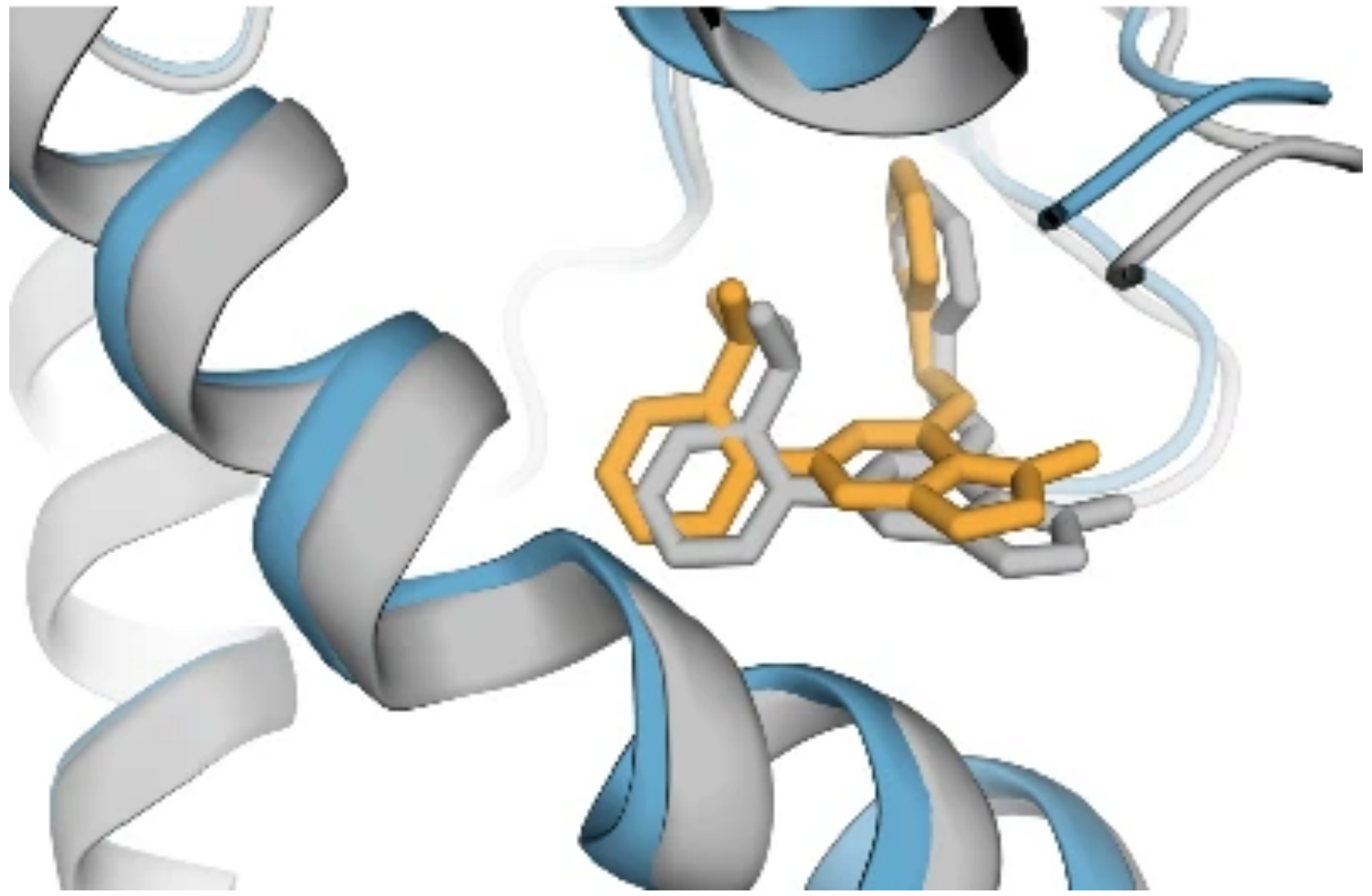
Good predictions help us make decisions that reduce suffering

climate prediction



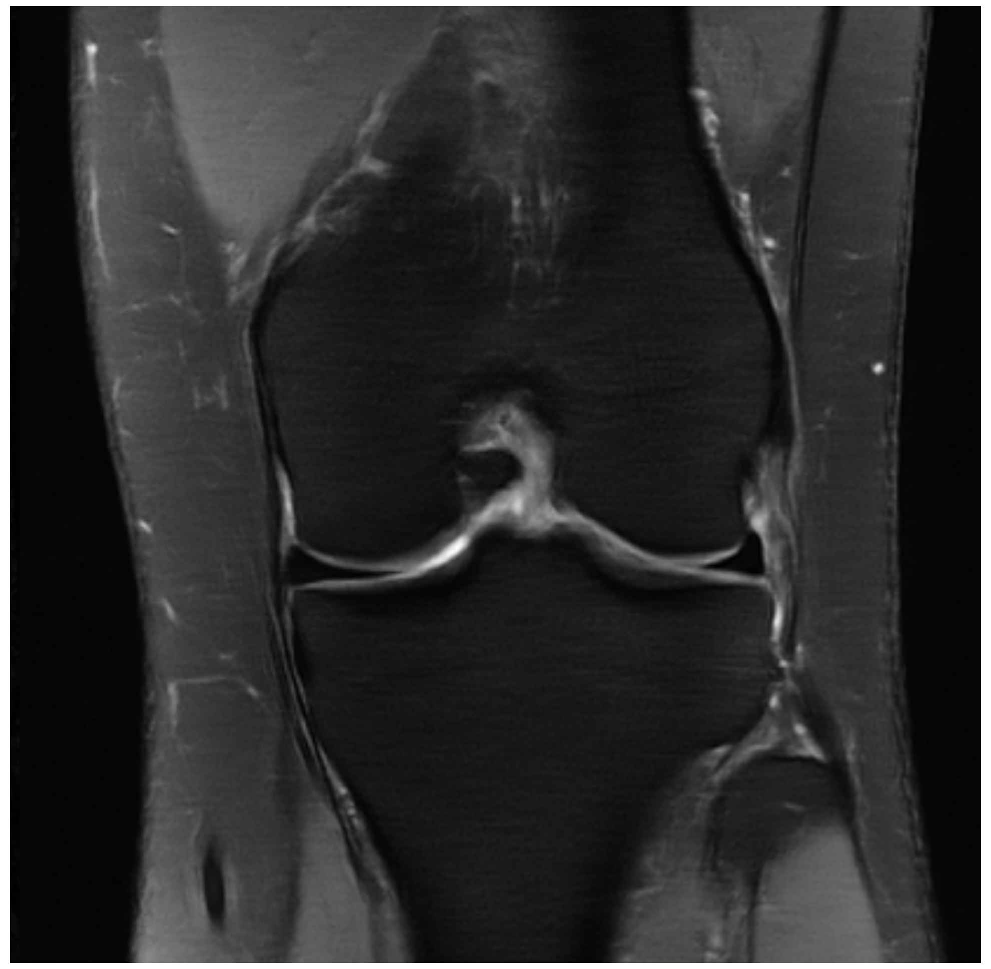
FourCastNet
(Pathak et al, 2022)

drug discovery



AlphaFold 3
(Abramson et al, 2024)

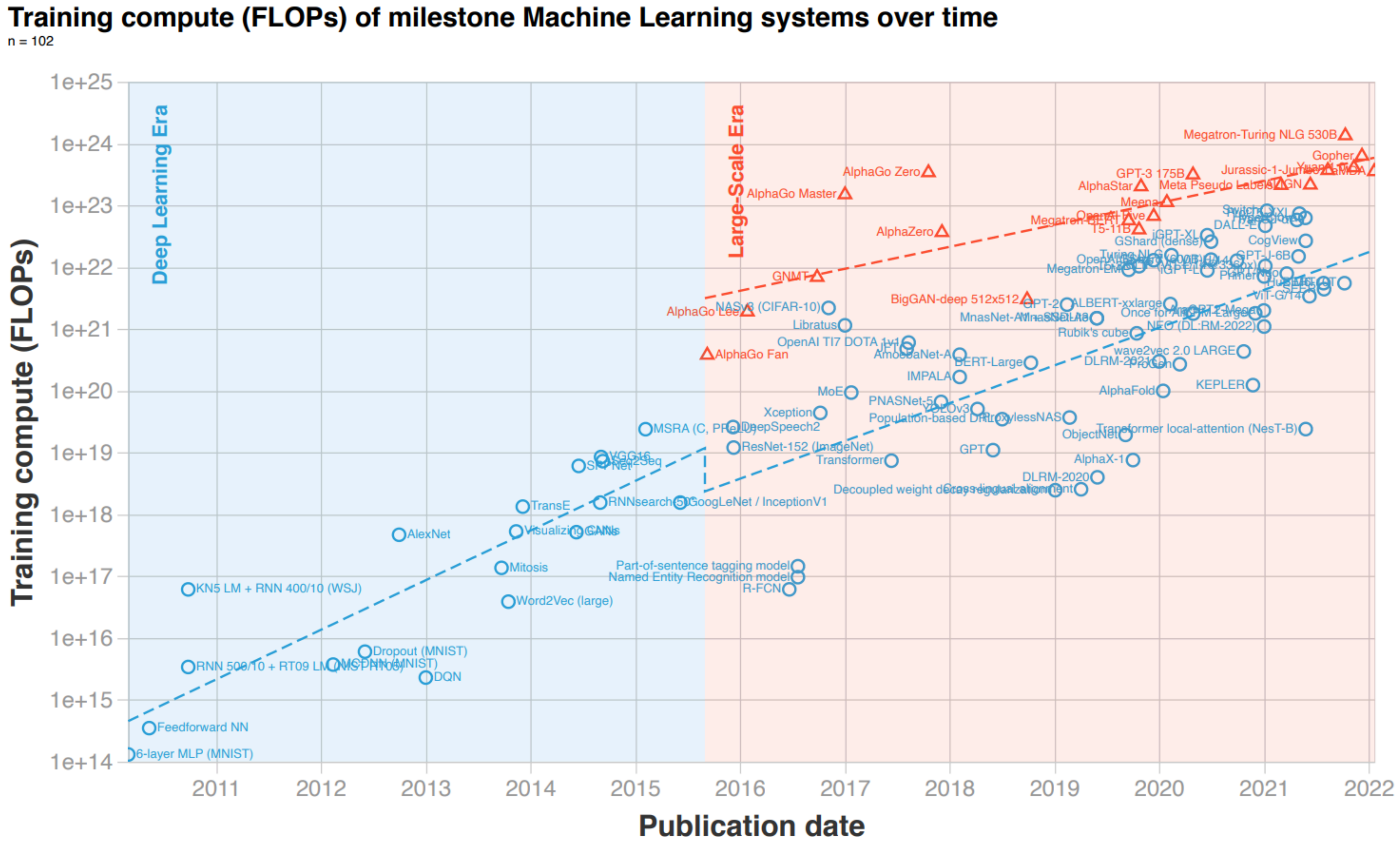
clinical decision-making



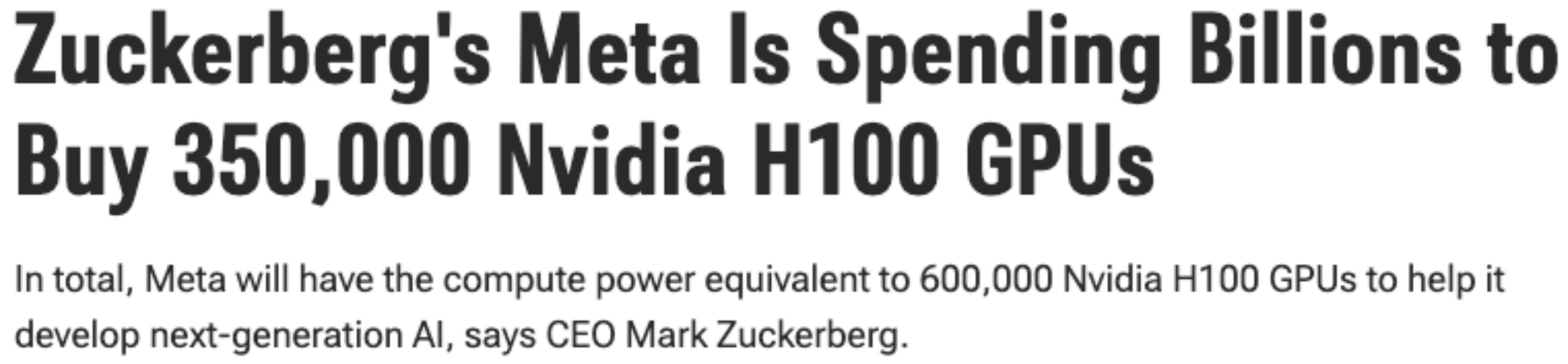
fastMRI
(Zbontar et al, 2019)

Compute, large models play a central role

Our algorithms are consuming increasing amounts of compute

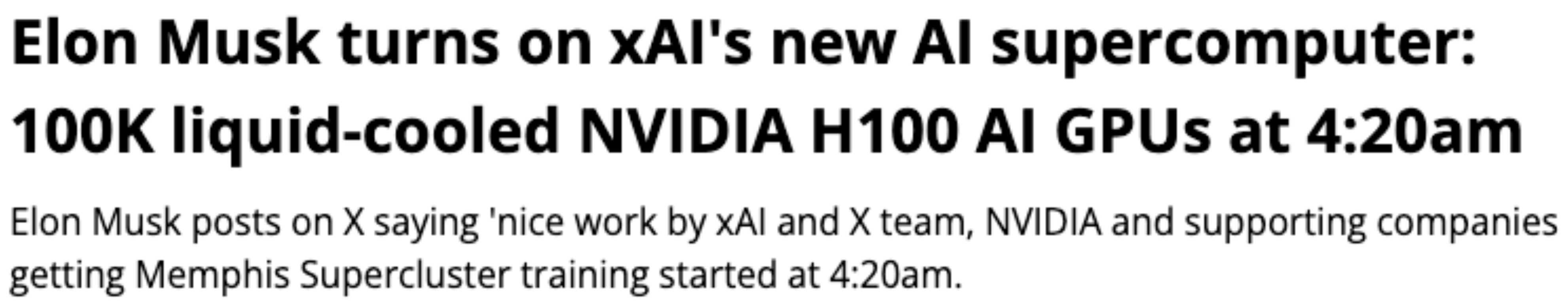


Sevilla et al., 2022. “Compute trends across three eras of machine learning”



Zuckerberg's Meta Is Spending Billions to Buy 350,000 Nvidia H100 GPUs

In total, Meta will have the compute power equivalent to 600,000 Nvidia H100 GPUs to help it develop next-generation AI, says CEO Mark Zuckerberg.



Elon Musk turns on xAI's new AI supercomputer: 100K liquid-cooled NVIDIA H100 AI GPUs at 4:20am

Elon Musk posts on X saying 'nice work by xAI and X team, NVIDIA and supporting companies getting Memphis Supercluster training started at 4:20am.

This course: the frontier of large models

- Deep dive into the development pipeline of one of the largest open-source models ever built.
    - We will read **The Llama 3 Herd of Models (arXiv:2407.21783)** over the course of 10 weeks. This will structure the course.
- A **graduate seminar course**:
    - Learning by reading the primary literature.
    - Most weeks will consist of **student presentations**.
    - Assessment largely focused on a **student project**.

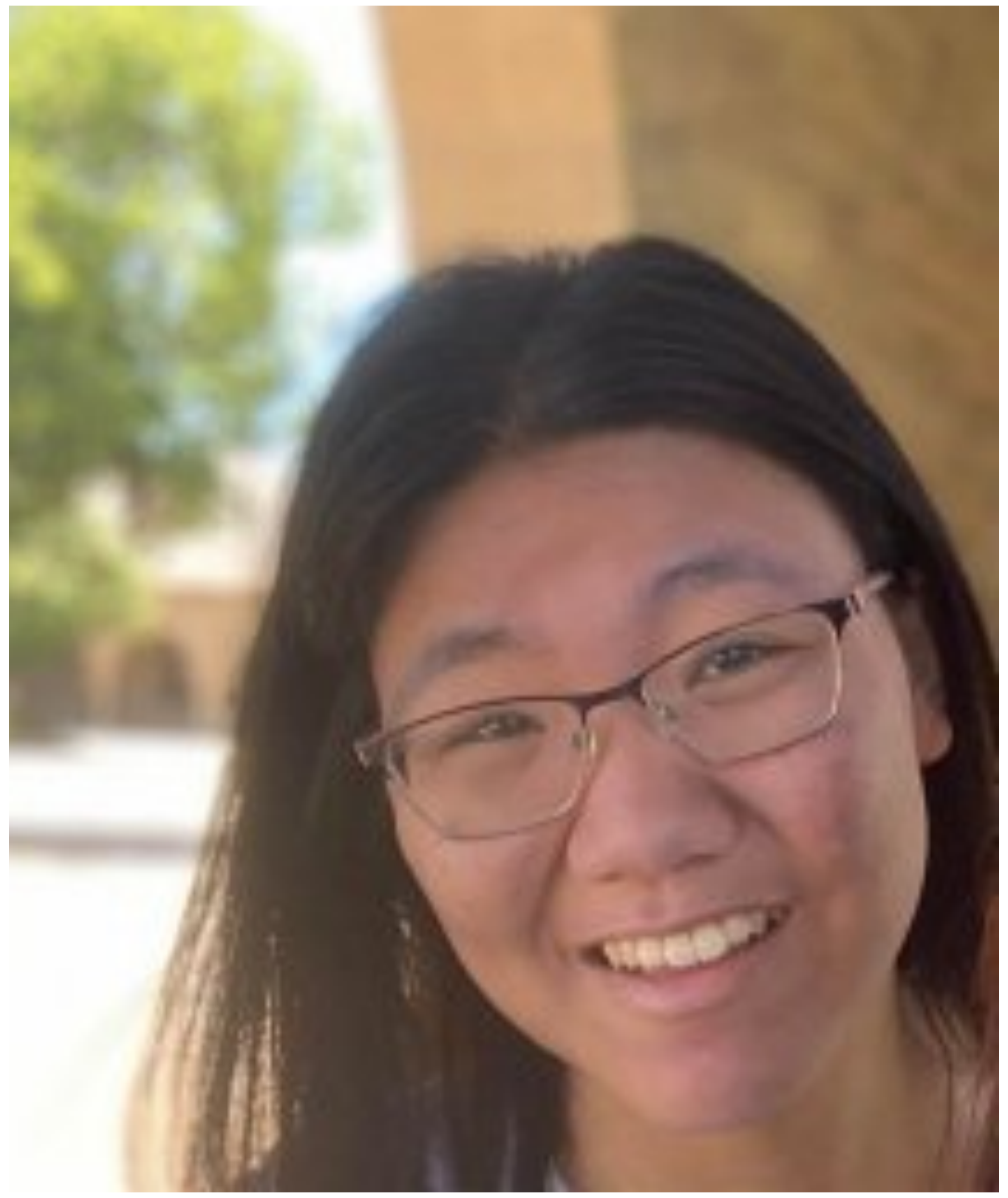
Course information

Course staff

csc2541-large-models@cs.toronto.edu

Chris Maddison
Instructor

Ayoub El Hanchi
TA



Frieda Rong
TA

Course information

Course website

https://www.cs.toronto.edu/~cmaddis/courses/csc2541_w25/

The course website will have the most up-to-date information and Quercus will be used for announcements.

Presentation sign-up

CSC2541 Winter 2025 Presentation Date Selection Form

https://forms.office.com/r/DpQ14sdAb3

Presentation timeline

- **At least two weeks before your presentation.**
    - Come to OH to finalize which paper you will present. Can come before two weeks.
- **A few days before your presentation.**
    - Meet with one of the TAs to practice your presentation and get feedback.
- **On the day of your presentation.**
    - Your slides and code notebook are due at the start of the class.
- **If you decide to drop the course (don’t feel bad), but please email me so that I can make sure we have enough presenters!**

Questions?

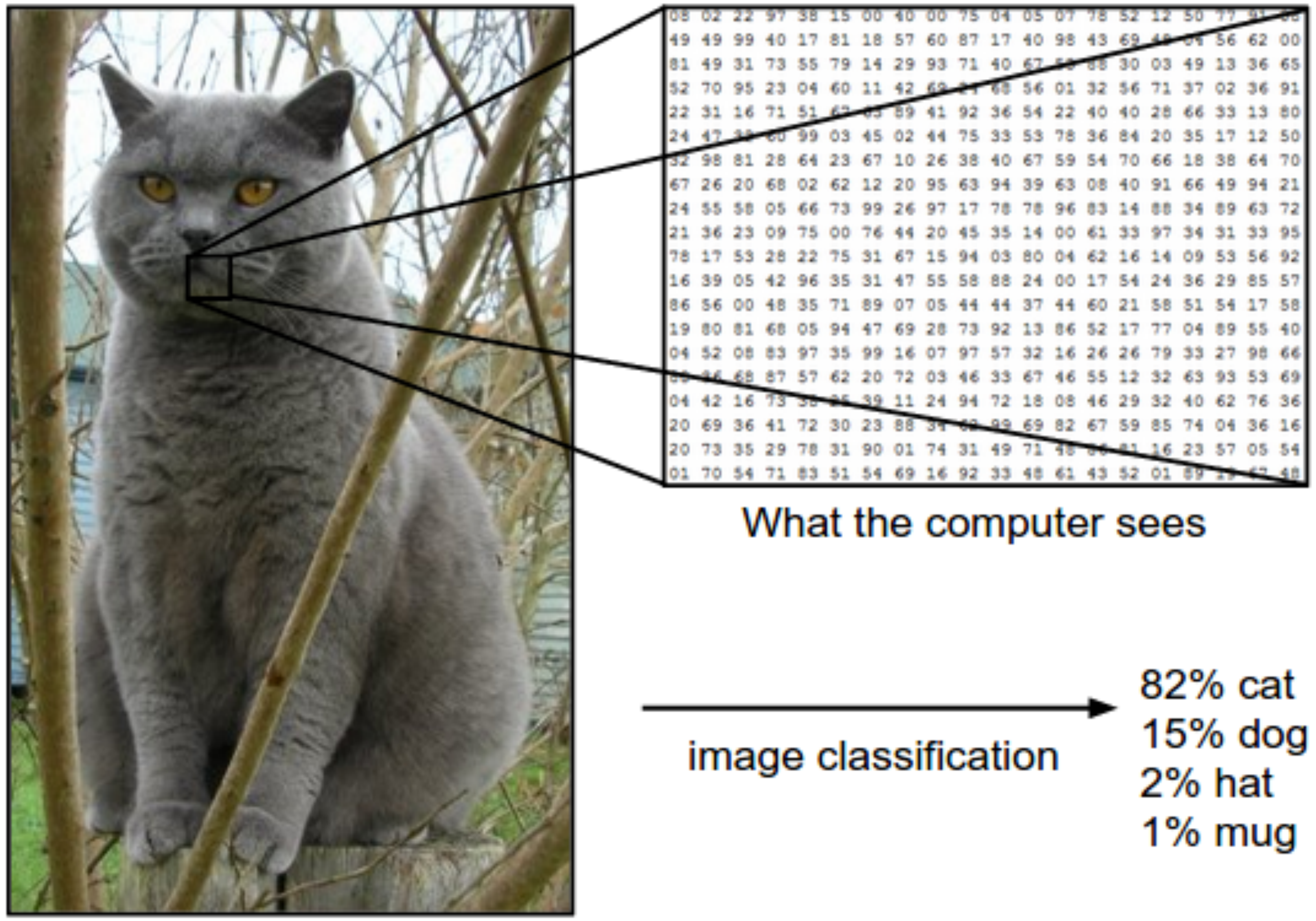
My mask

- I mask because I have had Long Covid since 2022.
- Long Covid is not rare, it can cause significant disability, and there is no approved treatment.
- I was so sick that I was off work for 2 years. This is my first time teaching since I got sick.
  - I am extremely grateful and excited to be here with you. Thanks for your patience.

The Story of A Single Bit

Data begins with a measurement

- A **measurement is an action that determines a property** of a system.
  - *E.g.*, silver halide crystals in film reducing to metallic silver determine light intensity.
- Stored measurements are data.
- In computers, we store data in a **digital representation.**
  - I.*e.*, a list of numbers.



Source: Andrej Karpathy

Bits

- More precisely, data is typically stored as sequences of 1s and 0s

$$(Y_1, Y_2, \ldots, Y_n) \in \{0,1\}^n$$

- I will tell you the story of a single bit of data, which is my story.
- What I am trying to highlight:
  - **Data is not abstract thing.**
  - The processes that produce it are complex and also very personal.

A Single Bit

- 1987, I was born in Boston.
- 2016, MSc from UofT.
- 2019, a novel coronavirus spreads across the world.
- 2020, PhD from Oxford.
- 2020, I joined the faculty at UofT.
- 2022, I decided to stop being as cautious about COVID.

A Single Bit

- Let Y be the outcome of a rapid antigen test, 1 if positive, 0 o.w.
- 7 February 2022, the single bit that changed my life:

$$Y = 1$$

- Something to think about: what is the provenance of a bit?

Prediction

Prediction

- **Could I have predicted whether I would test positive before the test?**
- Let’s study this abstractly. At a high level, the set up is as follows.
  - We specify a **prediction *before*** observing the outcome. A prediction a statement about a future event.
  - A **loss function** quantifies our prediction’s error upon seeing the outcome.
  - The expected loss function, or **risk, quantifies our error on average on random, unseen data.**

Bernoulli outcome with log-loss

A special case

- Represent the test outcome as a binary random variable $Y \in \{0,1\}$.
- We specify a prediction using a real number $q \in [0,1]$ to model $P(Y = 1)$.
  - Not the only choice! Could have predicted just 0 or 1.
- How do we score our prediction?

*The base of the logarithm does not change our discussion.

Bernoulli outcome with log-loss

A special case

- We score our prediction for each outcome using the **log-loss***,

$$\ell(Y, q) = -Y\log(q) - (1 - Y)\log(1 - q)\,.$$

- We score our prediction *on average* via the **risk (or cross-entropy in this case)**,

$$\mathbb{E}_Y[\ell(Y, q)] = -p\log(q) - (1 - p)\log(1 - q) = \ell(p, q).$$

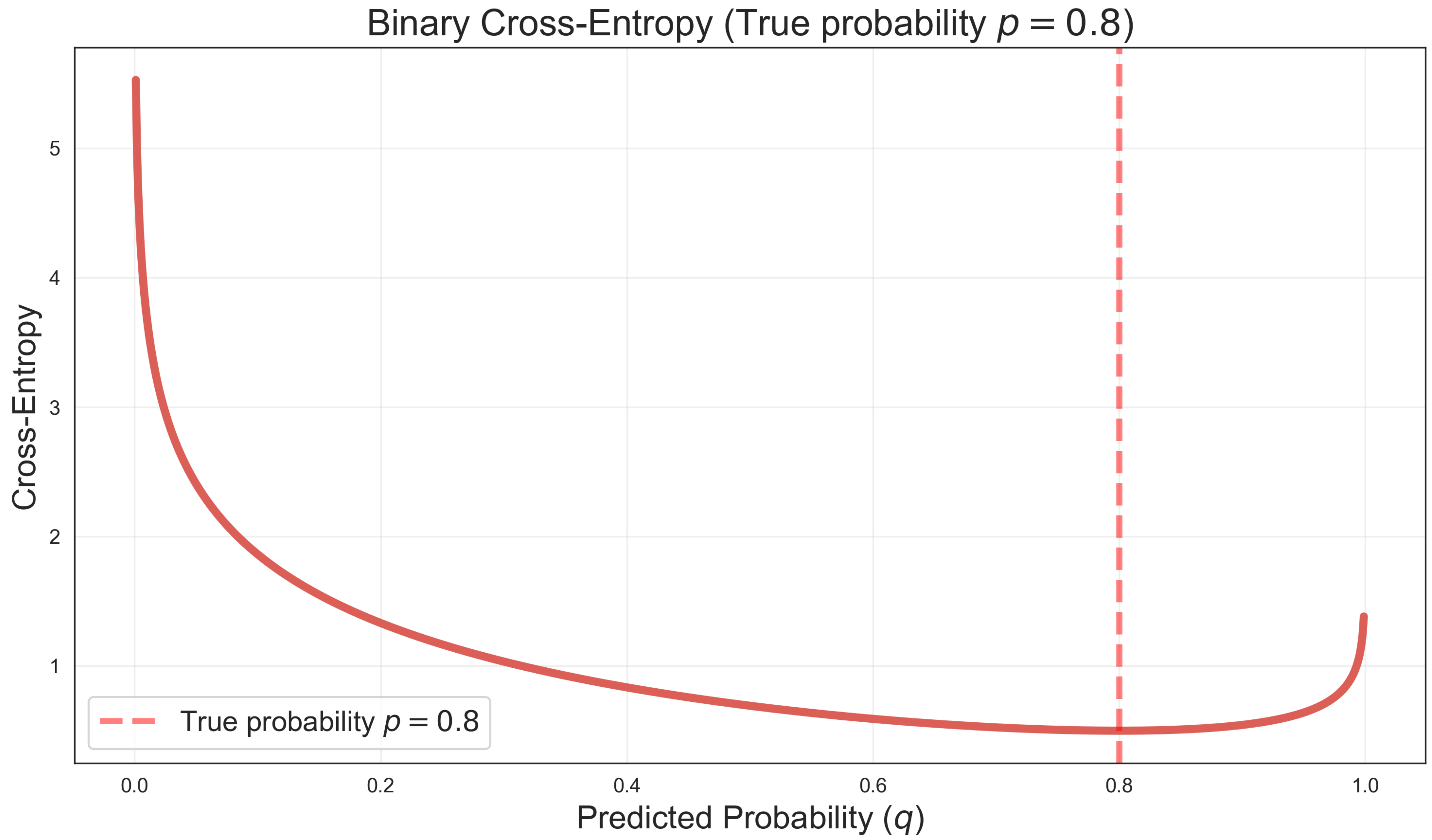
where $p = P(Y = 1)$.

Note: if $p \in \{0,1\}$, we drop the p term or $1 - p$ term, respectively.

*The base of the logarithm does not change our discussion.

Why is this a good choice for the risk?

p uniquely minimizes the cross-entropy



The prediction with the lowest risk is the true probability p.

Derivation

Cross-entropy minimizer

Prop 1. Let $p, q \in [0,1]$, then

$$\ell(p, q) \geq \ell(p, p)$$

with equality iff $q = p$.

Pf. First, consider the case $p, q \in (0,1)$. By **strict concavity:** $\log(x) \leq x - 1$ **with eq. iff** $x = 1$, and thus

$$\log(q/p) \leq q/p - 1 \text{ with equality iff } q = p.$$

Derivation

Cross-entropy minimizer

Applying $\log(q/p) \leq q/p - 1$ to q/p and $(1-q)/(1-p)$, we get

$$\begin{aligned} \ell(p,p) - \ell(p,q) &= p\log\left(\frac{q}{p}\right) + (1-p)\log\left(\frac{1-q}{1-p}\right) \\ &\leq p\left(\frac{q}{p} - 1\right) + (1-p)\left(\frac{1-q}{1-p} - 1\right) \\ &= (q-p) + (1-q-1+p) \\ &= 0 \end{aligned}$$

with equality iff $p = q$. If $p \in \{0,1\}$ or $q \in \{0,1\}$, the bound holds trivially. ∎

Uncertainty

Motivating the log-loss

- The log-loss can be motivated through a **statistical notion of uncertainty**.
- Goal: quantify the **“amount of uncertainty that is resolved when we observe an outcome Y”**, which measures how surprised we are to observe its value.
- To understand the goal consider the following scenario. **Can we quantify this?**
    - $Y = 1$ iff indep. events A and B happen.
    - After observing A, I would be less surprised to find out $Y = 1$ than I was before observing A, i.e., my uncertainty is reduced.

Uncertainty

Motivating the log-loss

- Let $h : (0,1] \to \mathbb{R}$ be the function that **represents my surprise upon observing an event that I think has probability q.**
  - If I believe $P(Y = 1) = q$, then my expected surprise is

$$ph(q) + (1 - p)h(1 - q).$$

- What properties should we expect h to satisfy?

Uncertainty

Motivating the log-loss

- Suppose I believe: $Y = 1$ iff indep. events A and B, $P(A) = q_1$, and $P(B) = q_2$.
  - My surprise at observing $Y = 1$ should be $h(q_1 q_2)$.
  - After observing A, the remaining surprise (after removing the surprise $h(q_1)$ at seeing A) should be exactly the *independent* surprise $h(q_2)$ of seeing B.
- To reflect this structure additively, **we can require that $h : (0,1] \to \mathbb{R}$ satisfy:**

$$h(q_1 q_2) - h(q_1) = h(q_2) \text{ for } q_1, q_2 \in (0,1]$$

Uncertainty

Motivating the log-loss

- If $h : (0,1] \to \mathbb{R}$ satisfies

  1. $h(q_1 q_2) = h(q_1) + h(q_2)$ for $q_1, q_2 \in (0,1]$,

  2. h is continuous and monotonically decreasing in q,

- then h must be

$$h(q) = -r \log_b(q) \text{ for some } r > 0, b > 1$$

- Follows from classical calculus arguments or see Robert Ash, *Information Theory.*

- We take $b = e$ and $r = 1$ by convention.

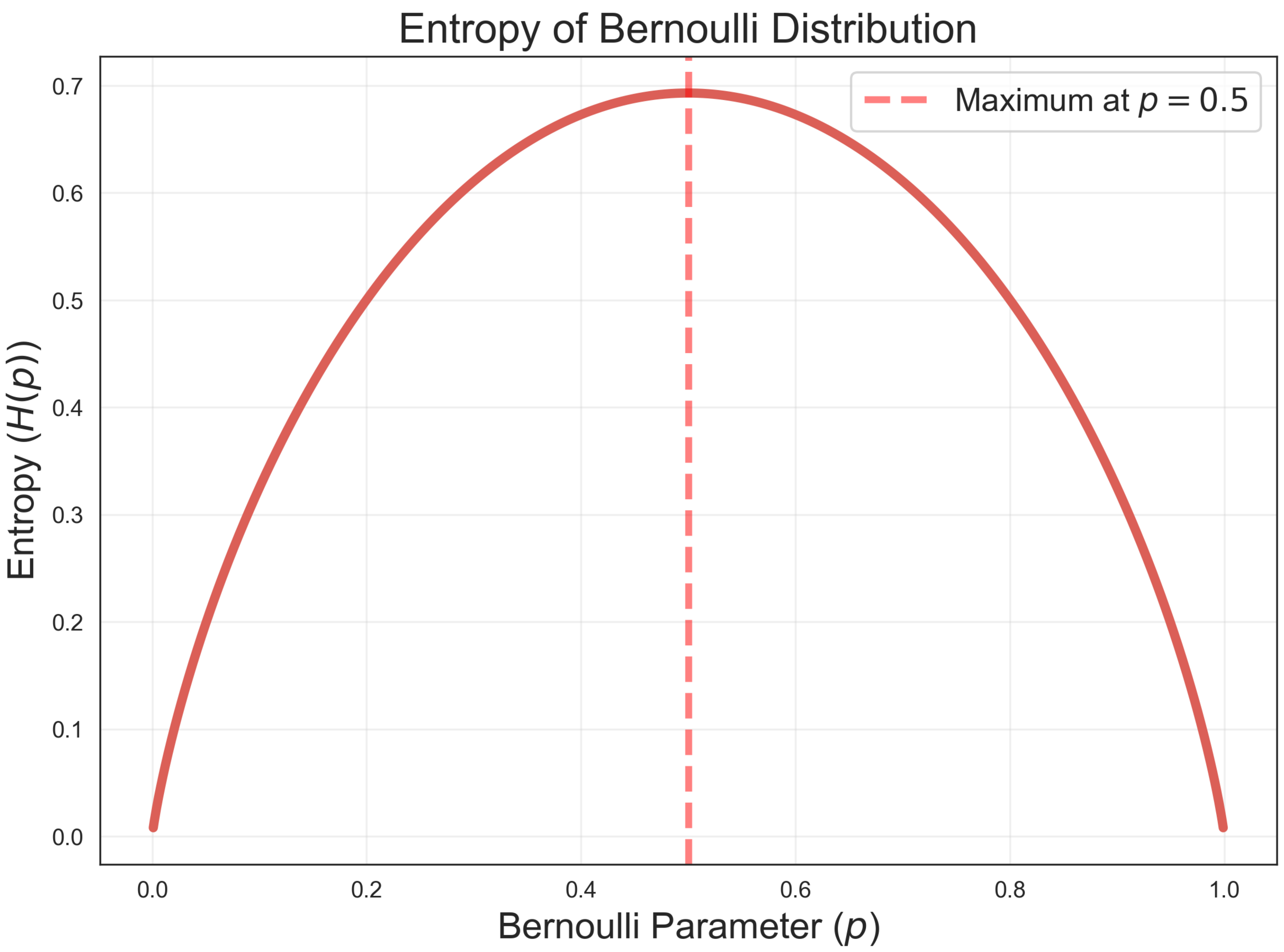
Entropy

the least surprised we could be

- $\ell(p, q)$ is minimized at $q = p$ and equals

$$H(p) = -p\log(p) - (1-p)\log(1-p)$$

- Called the **entropy, which is the least surprised we could be**.
  - Most surprised at outcomes if $p = 0.5$ and least surprised if $p \in \{0,1\}$.
- Claude Shannon: **cannot store data using fewer bits on average than the entropy.**



Recap

Prediction

- A prediction is a statement about a future event.
- We can predict random bits by specifying the probability of them being 1.
- The log-loss scores our surprise at observing the outcome.
  - Nice property: surprise at observing coincident, independent events is additive.
- The optimal prediction under the log-loss is the true probability, at which point the risk achieves the entropy.

Learning

Learning

- To predict well, we want $P(Y = 1)$. **But how can we get this in practice?**

- **Learning is the study of procedures that estimate predictors from data.**

- When we do a good job of learning, i.e., we found a good predictor from a set of observations, we say that we have achieved good **generalization**.

- Returning to our Bernoulli example, let’s study a simple learning algorithm: **empirical risk minimization**.

Empirical risk minimization

A special case

- We observe a data set $\{Y_i\}_{i=1}^{n}$ for outcomes Y_i that are i.i.d. Bern(p).
  - *E.g.*, a set of COVID test outcomes from Toronto during the pandemic.
- Ideally we would be able to solve the risk minimization problem to get p,

$$\arg\min_{q\in[0,1]} \ell(p, q)$$

- But we don’t have access to $\ell(p, q)$…

Empirical risk minimization

A special case

- Instead, we can approximate the risk with **the empirical risk**:

$$\frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, q).$$

- Notice that $\frac{1}{n}\sum_i \ell(Y_i, q) \to \ell(p, q)$ almost surely by the law of large numbers.
- This motivates **empirical risk minimization**, which is the estimation procedure that finds

$$\arg\min_{q\in[0,1]} \frac{1}{n}\sum_{i=1}^{n} \ell(Y_i, q)$$

Derivation

Bernoulli ERM

Prop 2. Given $\{Y_i\}_{i=1}^n$ i.i.d. Bern(p), the ERM w.r.t. the log-loss is

$$\hat{p}_n = \arg\min_{q \in [0,1]} \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, q) \text{ where } \hat{p}_n = \frac{1}{n} \sum_{i=1}^{n} Y_i$$

Pf. Note,

$$\frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, q) = \frac{1}{n} \sum_{i=1}^{n} - Y_i \log(q) - (1 - Y_i) \log(1 - q) = \ell(\hat{p}_n, q).$$

$\ell(\hat{p}_n, q)$ is a cross-entropy, so our result follows from Prop 1. ∎

ERM with log-loss

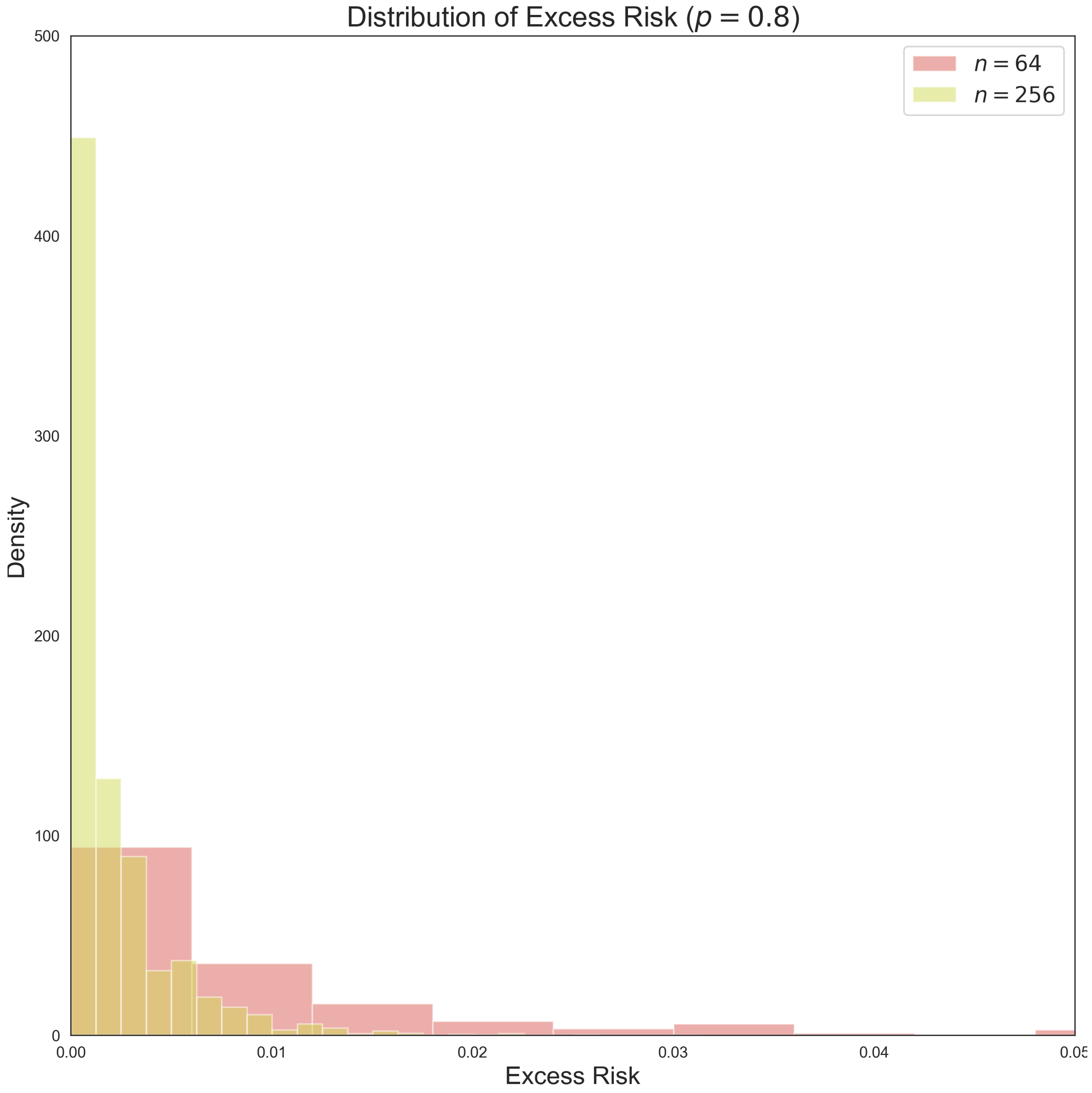
- Does $\hat{p}_n$ approach p as $n \to \infty$ in some sense?
- We can study this by studying the behaviour of **the excess risk**

$$\mathscr{E}(q) = \ell(p, q) - \ell(p, p)$$

- Note two things:
  - $\mathscr{E}(q) \geq \mathscr{E}(p) = 0$ for all $q \in [0,1]$.
  - **$\mathscr{E}(\hat{p}_n)$ is a non-negative real-valued *random* variable.**

ERM convergence

- **Does $\mathscr{E}(\hat{p}_n)$ converge?**
    - Recall, $\mathscr{E}(\hat{p}_n)$ is a random variable, so need to define "converge".
- Answer: it convergences in various senses under various conditions!
    - van der Vaart, *Asymptotic Statistics* or Ostrovskii and Bach (2020).
    - Very deep and theoretical area of inquiry - out of scope for this class.

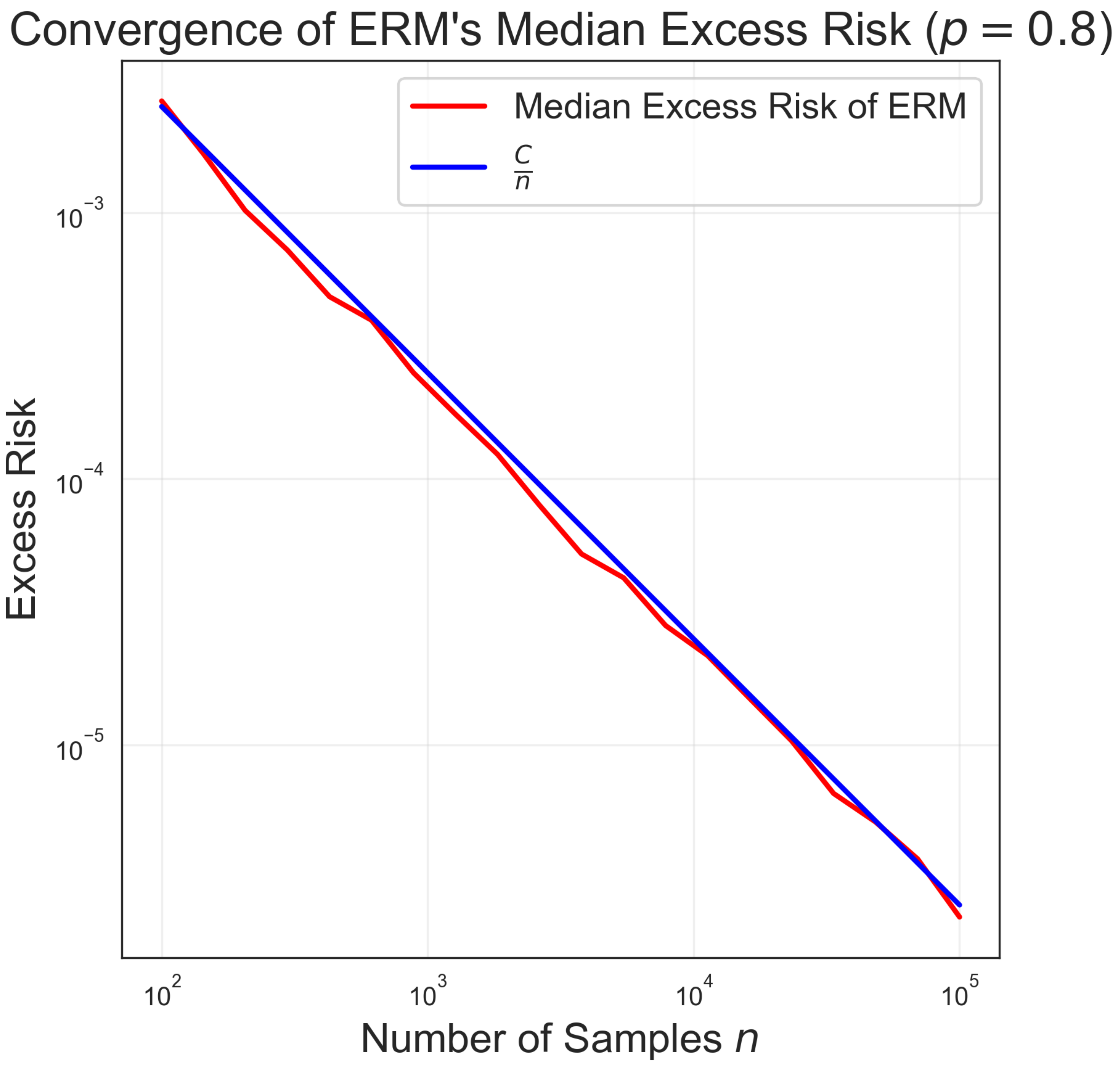


ERM convergence

- Let's study convergence in simulation (derivation in the Bernoulli case is a bit tricky).
- **Median converges like,**

$$\textbf{median}\left[\mathscr{E}(\hat{p}_n)\right] \to C/n$$

  - Typical rate for learning.
- **Key take-home: the more data we have, the better our predictions.**



Recap

Learning

- Learning is the study of procedures that estimate predictors from data.
- Empirical risk minimization tries to solve this by picking the predictor that minimizes the average loss on a data set.
- Predictors obtained from data are random because the data is random.
- We can study predictors by studying the excess risks, *i.e.*, the deviation of the expected loss from the best possible expected loss.
- When the excess risk is small, we have generalized.
- **The more data we have, the better we generalize.**

Conditional Prediction

Conditional prediction

- Could I have **predicted my test more accurately, if I had other measurements** about me?
  - Did anything in the last 37 years make $Y = 1$ more likely?
- Answer is **typically yes!**
- Let's study a special case of conditional prediction: **logistic regression**.

Logistic regression

A special case

- **Let $X \in \mathbb{R}^d$ be a random vector of other measurements called “features”.**
  - E.g., my age as a number, my location as coordinates, etc.
- Seeing $X = x$ **may inform us about** Y and make $Y = 1$ more predictable.
  - I.e., $P(Y = 1 \,|\, X = x)$ may have less entropy than $P(Y = 1)$.
- To take advantage of this, we can build **conditional predictions of** Y **given** X.
  - Logistic regression is a special case!

Logistic regression

A special case

- **Logistic regression:** predict $P(Y = 1 \mid X)$ with a sigmoid function that depends linearly on X:

$$q(X) = \sigma(w^\top X)$$

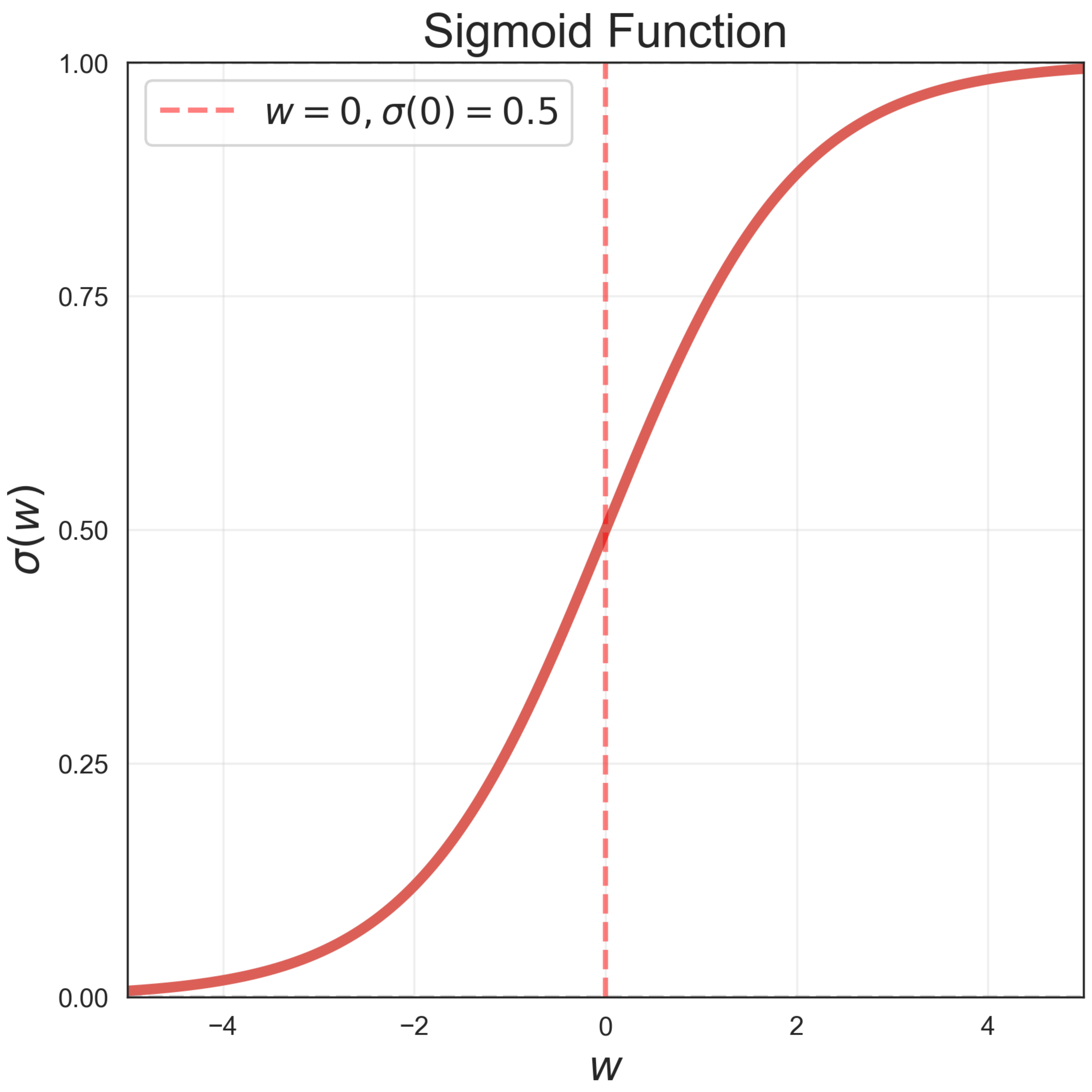
where $\sigma(t) = \dfrac{1}{1 + \exp(-t)}$ is the sigmoid and $w \in \mathbb{R}^d$ **are called parameters**.

- The params. plus the rule for computing the prediction is called **the model**.

Logistic regression

Intuition

- This is a generalization of the Bernoulli prediction case we considered.
    - Take $X \equiv 1$ to be constant.
    - Then $q(1) = \sigma(w)$ where $w \in \mathbb{R}$.
    - Can represent any $q \in (0,1)$ this way.



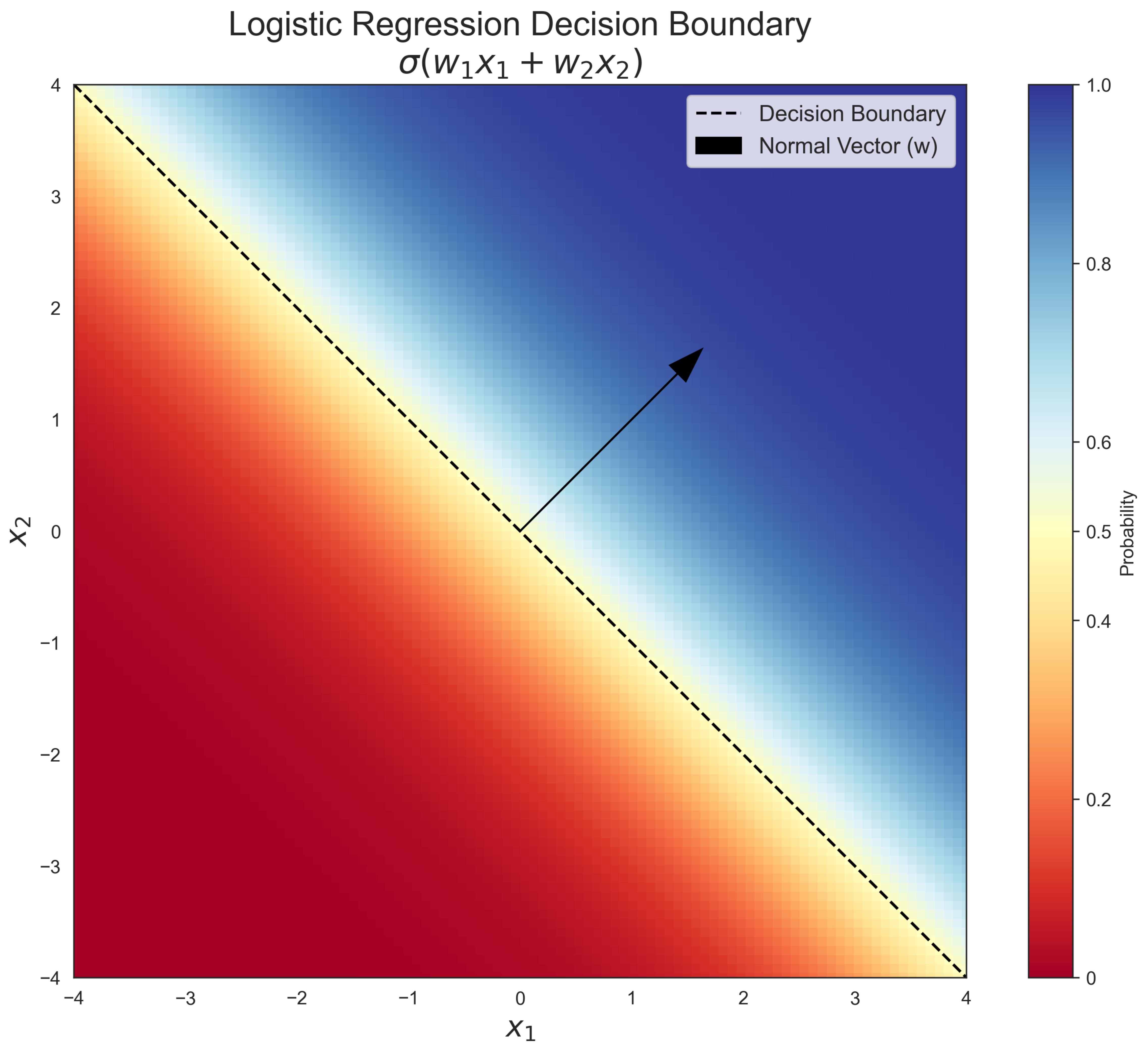
Logistic regression

Intuition

- The set of x where $\sigma(w^\top x) = 0.5$ is the hyperplane

$$\{x \in \mathbb{R}^d : w^\top x = 0\}$$

- ***w* separates our predictions.**
  - As x travels along w, our prediction that $Y = 1$ increases towards 1.
  - As x travels along $-w$, our prediction that $Y = 1$ decreases towards 0.



Conditional prediction

Which *w* should we pick?

- We can start by defining a notion of risk, similar to the Bernoulli case

$$R(w) = \mathbb{E}_{X,Y}\left[\ell(Y, q(X))\right]$$

- Interpretation: risk of our prediction of $P(Y = 1 \mid X)$ averaged over X.
- If the entropy of $P(Y = 1 \mid X = x)$ is much smaller than $P(Y = 1)$ for all x, then we can achieve much lower risk in principle with conditional prediction.

Logistic regression

Which *w* should we pick?

- The structure of R depends on X's distribution and its relationship to Y.
- **Realizable case:** let's assume, that there exists a unique w^* s.t.

$$w^* = \arg\min_{w \in \mathbb{R}^d} R(w) \text{ and that } P(Y = 1 \,|\, X = x) = \sigma(x^\top w^*)$$

- How do we get w^*? As before, it is common to **optimize the empirical risk:**

$$\hat{R}_n(w) = \frac{1}{n} \sum_{i=1}^{n} \ell(Y_i, \sigma(w^\top X_i)) \textbf{ where } (X_i, Y_i) \textbf{ are i.i.d. as } (X, Y)$$

Logistic regression

Which *w* should we pick?

- Optimizing is harder than the Bernoulli case: (i) sometimes there's no minimizer, (ii) when there is a minimizer, it's not always unique, and (iii) even if it's unique, there's often no closed form!
- Out of scope to study this in detail, let's assume there exists a unique ERM,

$$\hat{w}_n^* = \arg\min_{w \in \mathbb{R}^d} \hat{R}_n(w)$$

- How can we find the ERM? **Gradient descent is one choice!**

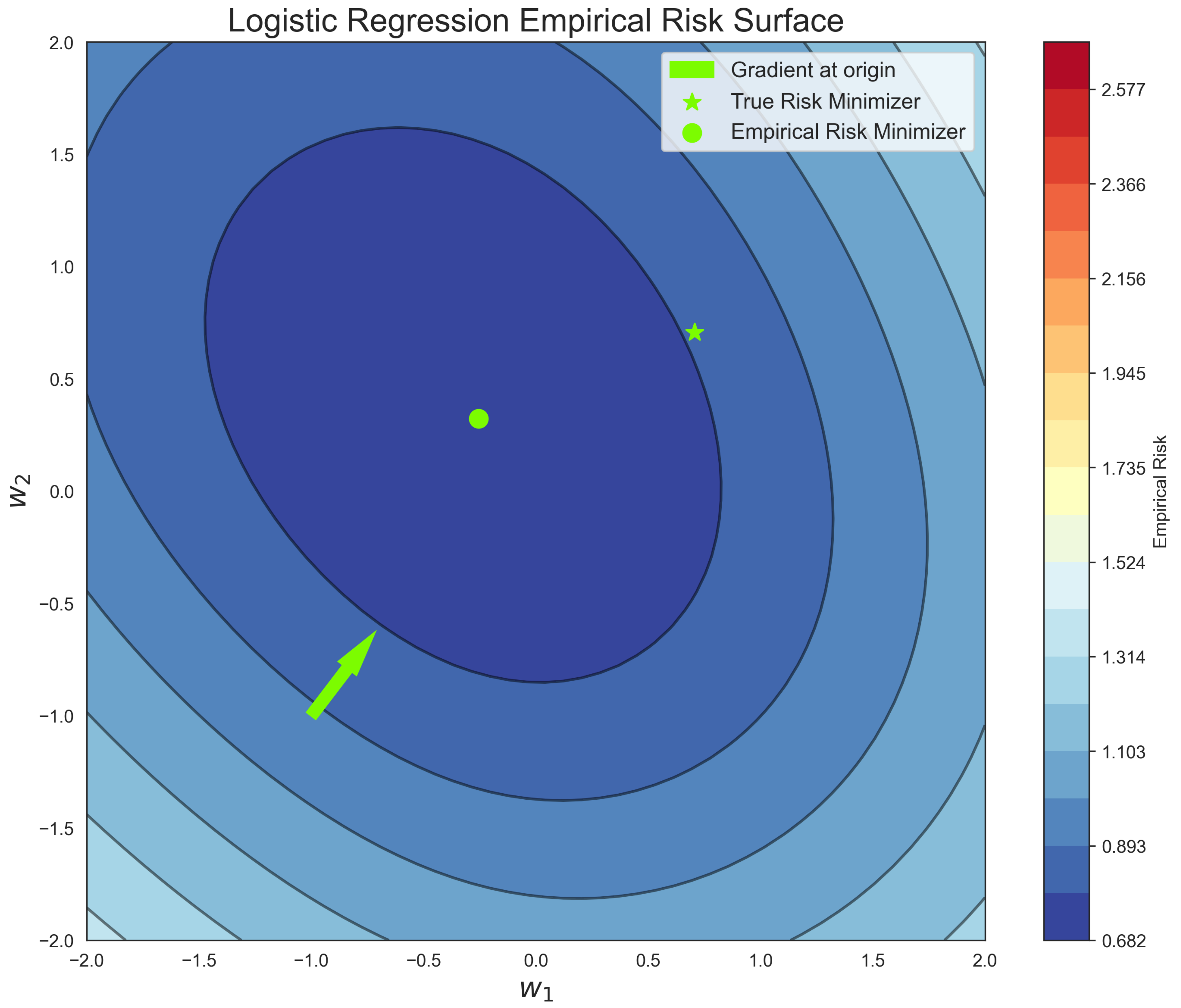
Gradient descent

Intuition

- The **gradient of the empirical risk** is the vector of partial derivatives,

$$\nabla \hat{R}_n(w) = \left(\frac{\partial \hat{R}_n}{\partial w_j} \right)_{j=1}^{d}$$

- **The negative gradient is the direction of greatest instantaneous descent on the surface of $\hat{R}_n(w)$.**



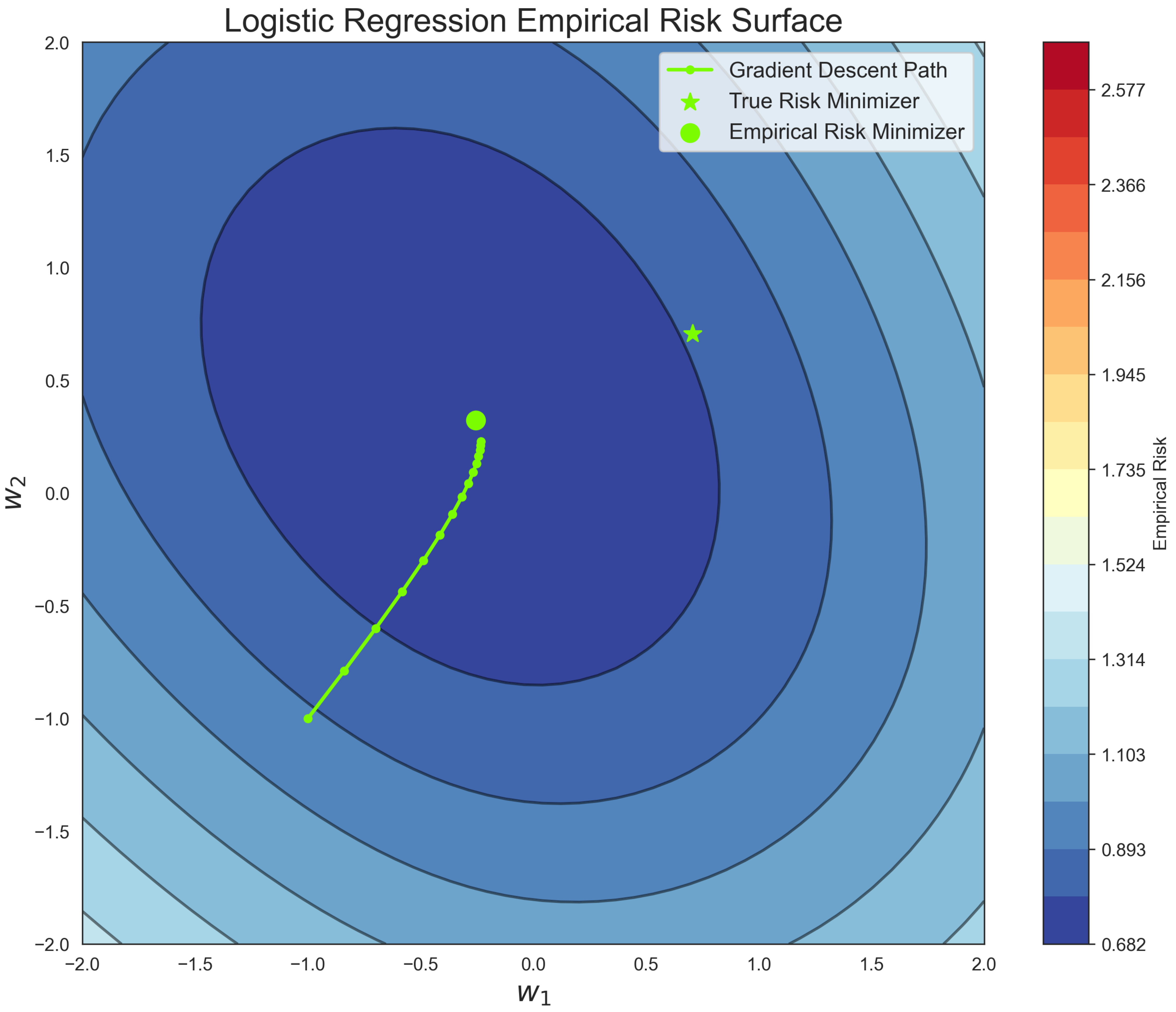
Gradient descent

Intuition

- The gradient descent algorithm iteratively **follows the gradient:**

$$w^{(t+1)} = w^{(t)} - \eta \nabla \hat{R}_n(w^{(t)})$$

for some step-size $\eta > 0$.



Gradient descent

Intuition

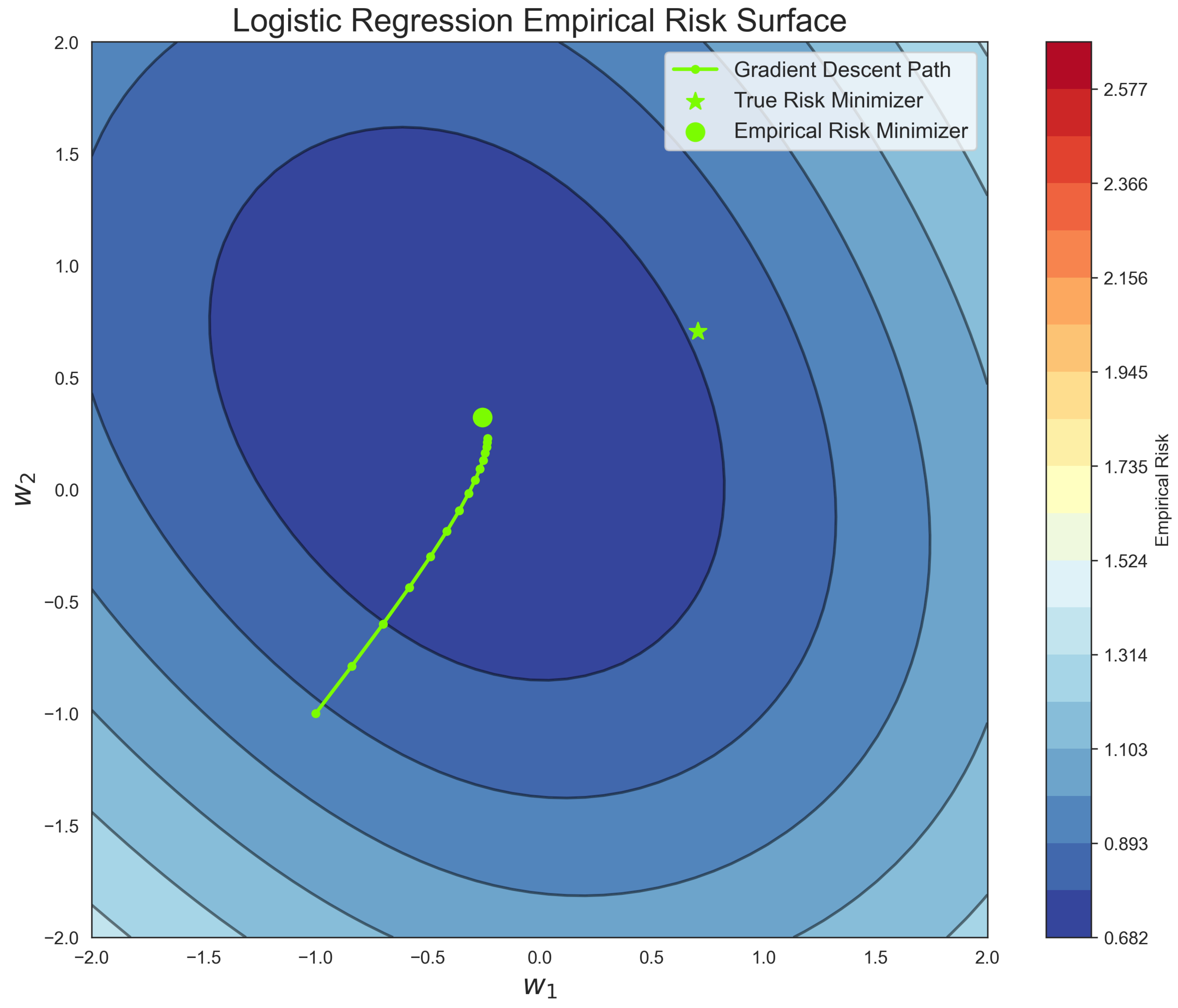
- In our setting, we have

$$w^{(t)} \to \hat{w}_n^*$$

for small η because the empirical risk is *smooth* and *convex*.

  - Technical terms that are out of scope for us.

- Optimization is another very large, very deep field (also out of scope).



Logistic regression

How well does ERM perform in this case?

- Does $\hat{w}_n^*$ approach w^* as $n \to \infty$ in some sense?
- Again, we can study generalization by studying the behaviour of **the excess risk**

$$\mathscr{E}(w) = R(w) - R(w^*)$$

- Classical result (see Ostrovskii and Bach, 2020): under mild smoothness conditions,

$$\mathbb{E}\left[\mathscr{E}(\hat{w}_n^*)\right] = \frac{d}{2n} + o(n^{-1}) \text{ as } n \to \infty$$

- **Key take-home: the more data we have, the better our predictions, BUT the more parameters, the worse our predictions.**

Recap

Conditional Prediction

- Observing more data can sometimes improve predictions.
- We can compute conditional predictions with parametric models like logistic regression.
- In general, finding the ERM of parametric models is challenging and we often resort to iterative optimization algorithms like gradient descent.
- **Rule of thumb: learning in parametric models improves with data and deteriorates with parameter count.**

Large Models and Bitter Lessons

From logistic regression to ChatGPT

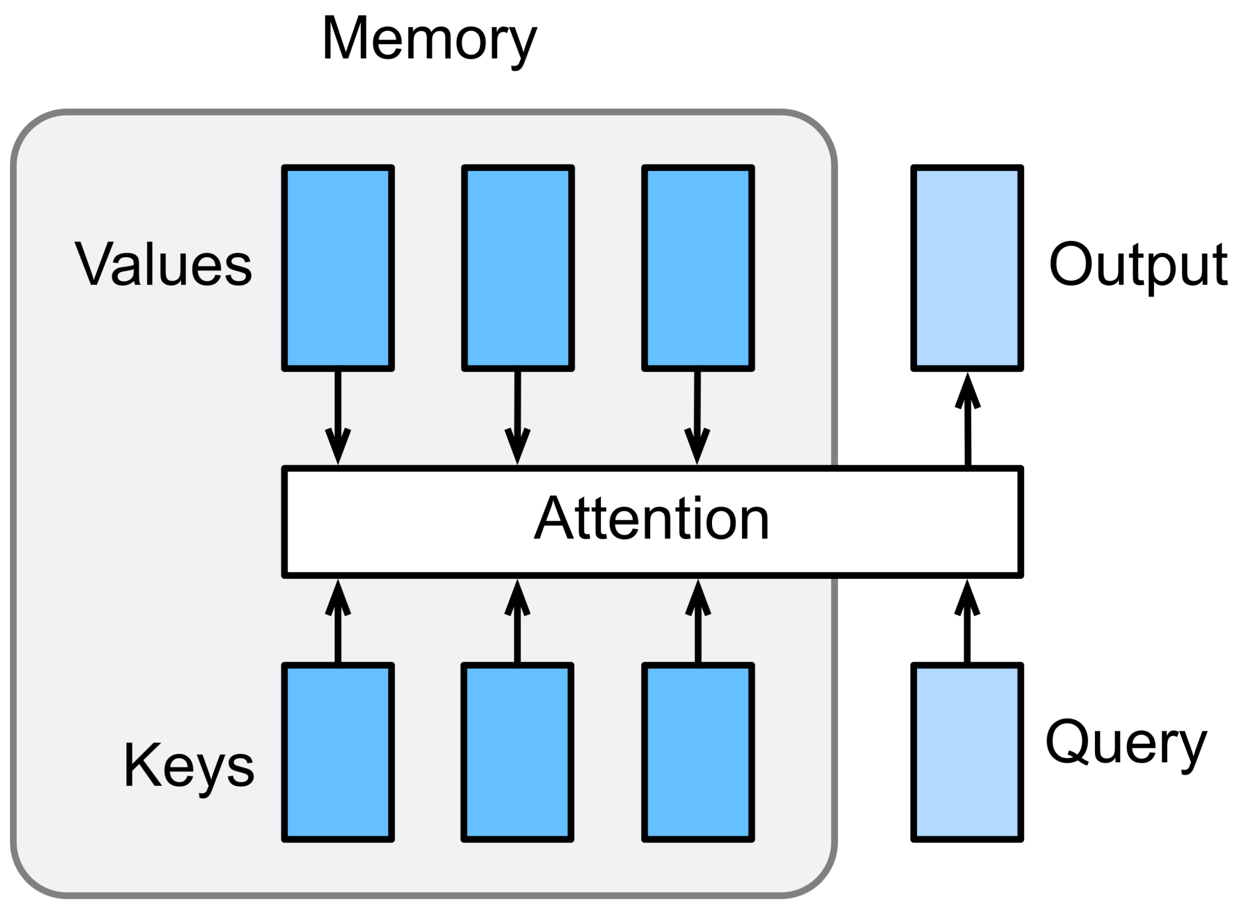
- So far the methods we’re looking at are simple and classical, taking us probably to the mid 20th century. You have probably already seen them.

- About 70 years of AI research brought us from what I presented to the start of the current revolution in AI.

- Just to give you a brief flavour of the kinds of changes to the paradigm that led to the large language models like ChatGPT and Claude…

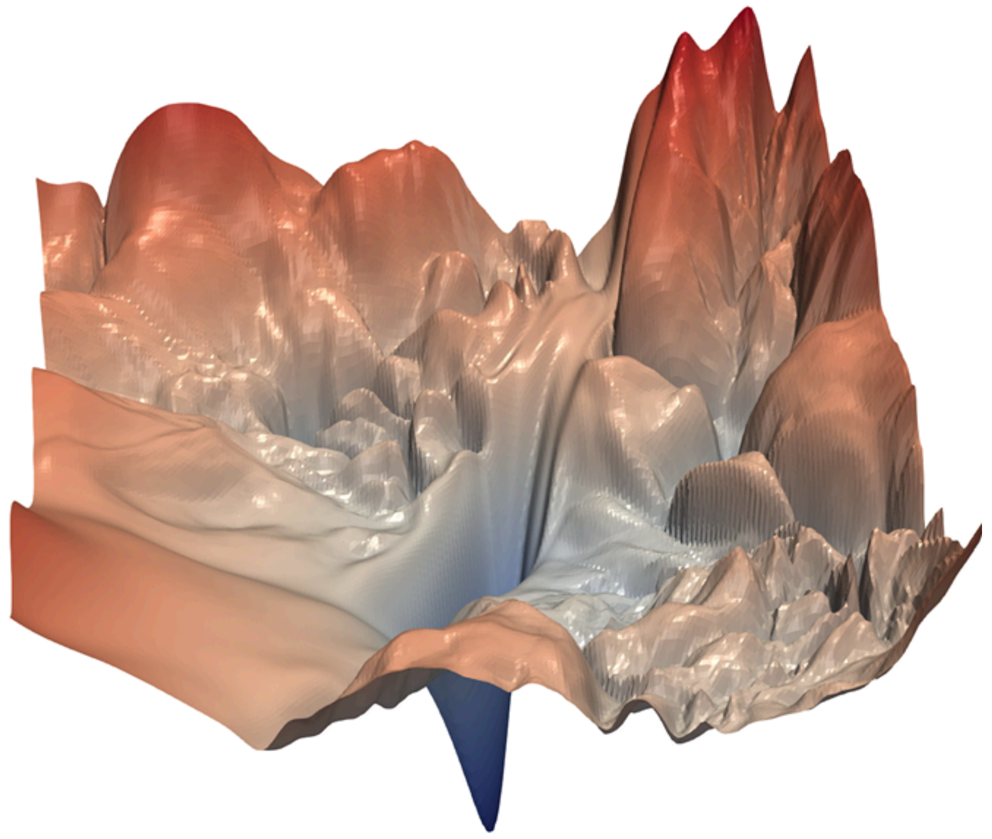
From logistic regression to ChatGPT

Predictions computed with **non-linear models**



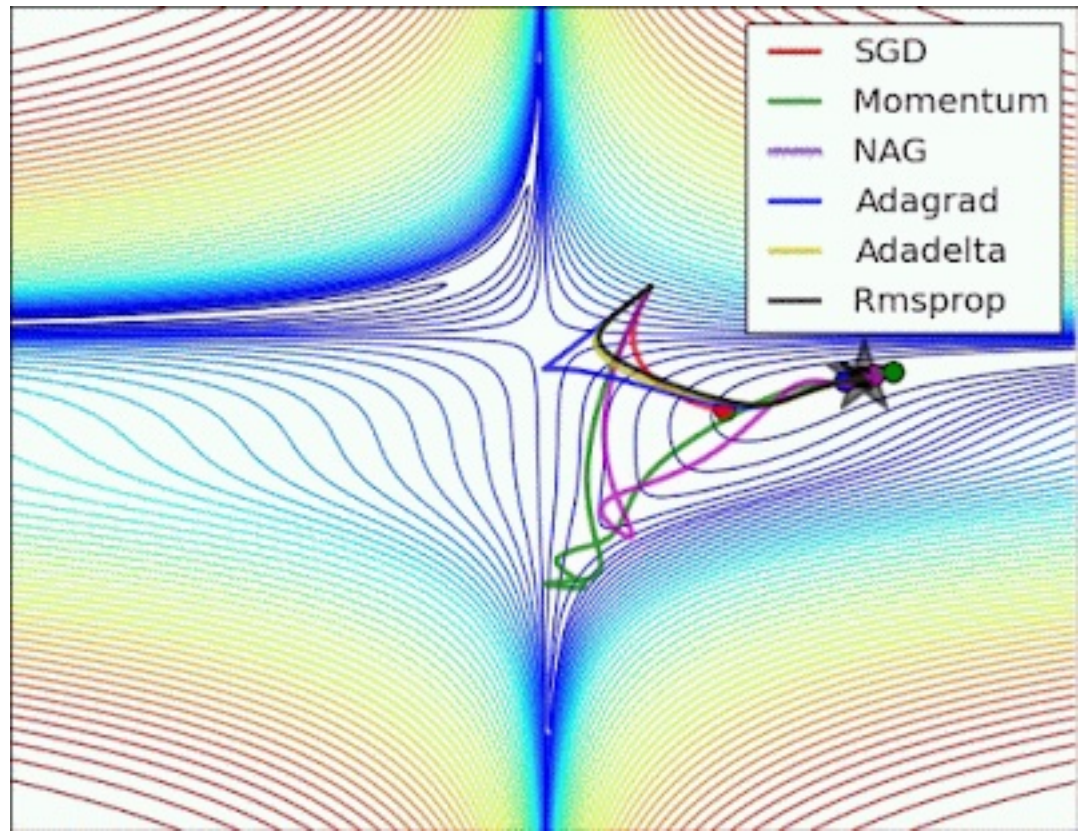
credit: Wikipedia

Non-convex empirical risk surfaces



Li et al. 2018. Visualizing the Loss Landscape of Neural Nets.

New, **better optimizers**



credit: Deniz Yuret

Still, AIs like ChatGPT are in this same basic paradigm as logistic regression:

They are parametric models.

From logistic regression to ChatGPT

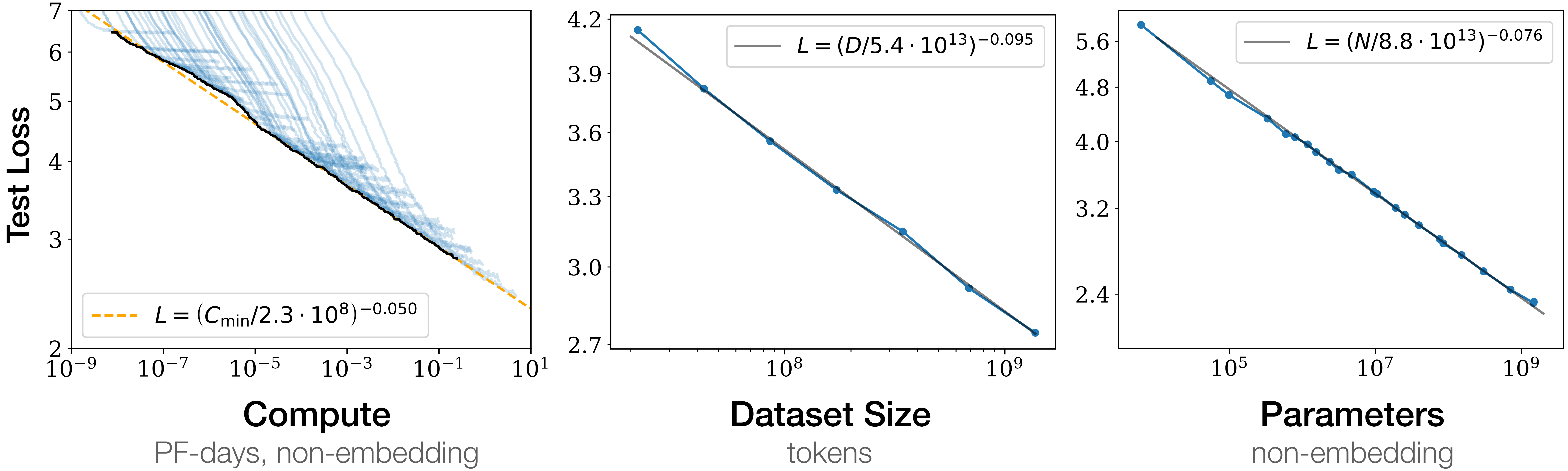
- Large language models (LLMs) are
  - predictors that are parameterized by real-valued vectors
  - whose parameters are set by minimizing empirical risks (ish)
  - with gradient-descent-type algorithms on data gathered from the internet.
- In essence, LLMs are **predictors of the text that is found on the internet**.
  - Massive oversimplification (**more next week!**)…

Despite this, LLM era is (apparently) distinguished by some key trends:

Risk seemingly improves with data **and parameter count**
Many capabilities emerge with scale.

Observation: risk shrinks smoothly with scale

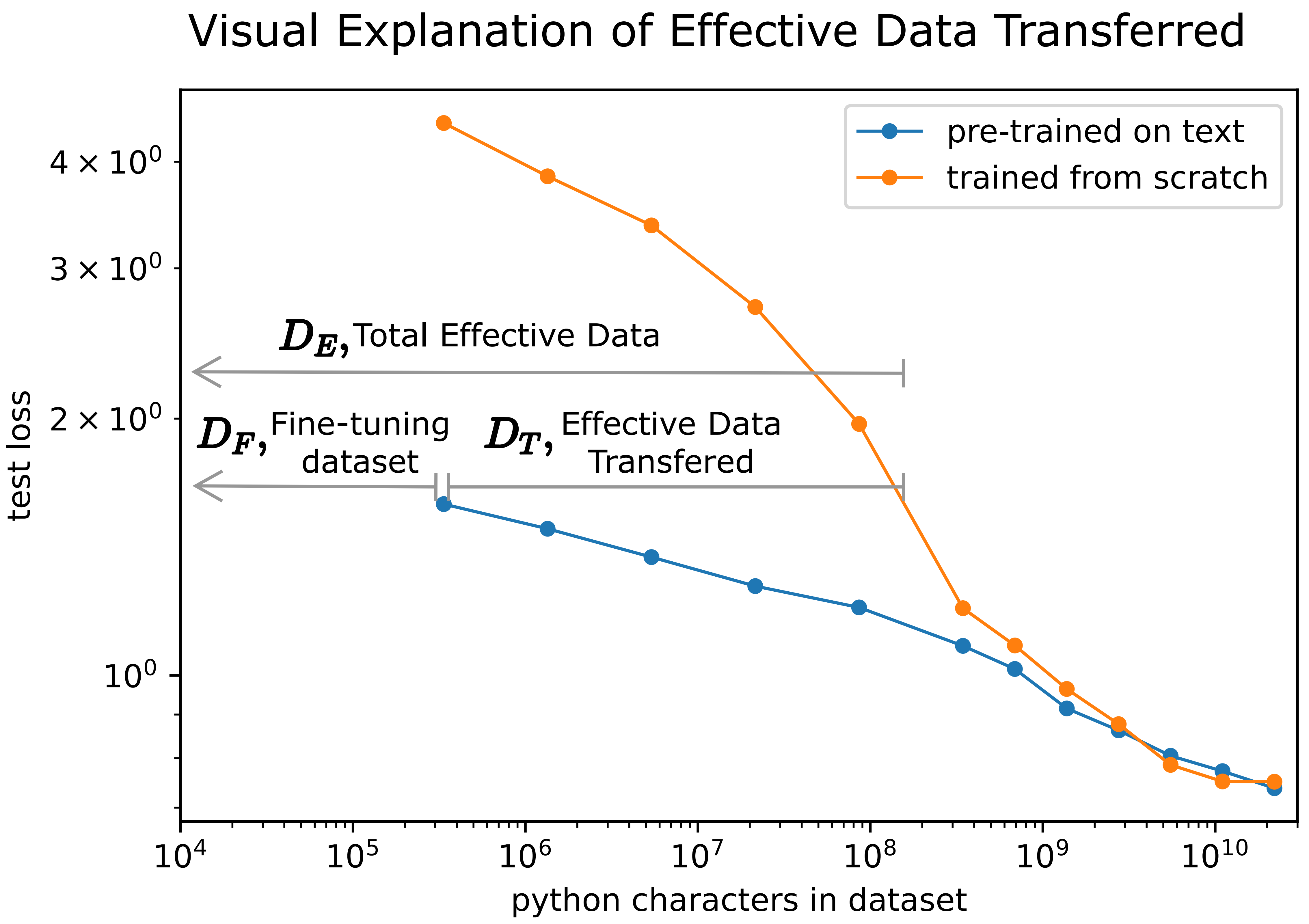
Increasing data or parameter count improves the risk



Kaplan et al. 2020. Scaling Laws for Neural Language Models.

Observation: information transfers between domains

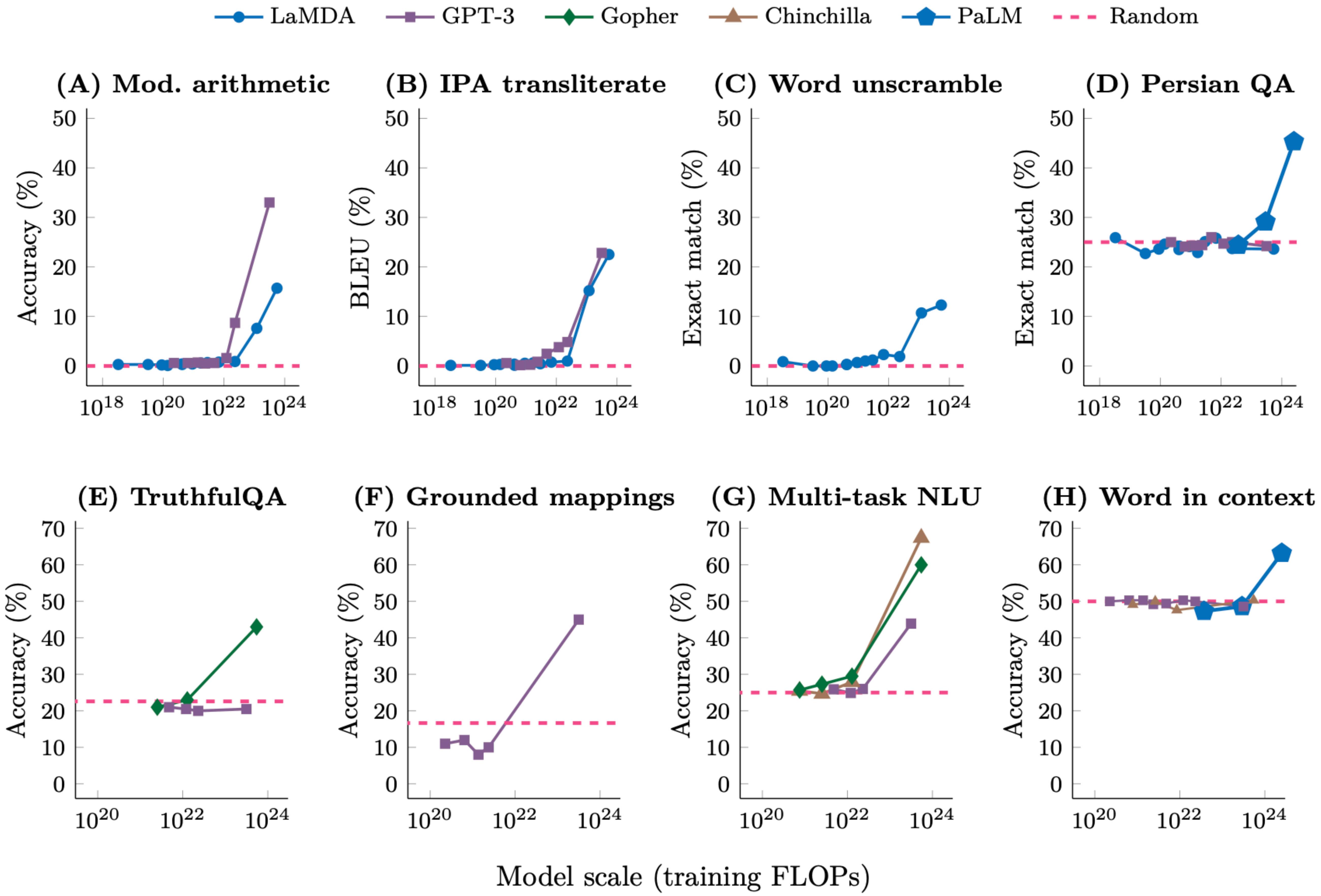
Pre-training on natural language transfers information to other domains



Kaplan et al. 2021. Scaling Laws for Transfer.

Observation: multi-task capabilities emerge with scale

As models scale on internet data, they improve on very diverse set of capabilities



Wei et al. 2022. Emergent Abilities of Large Language Models.

Paradigm shift

- Two things to note:
  - Many quantities (risk, optimal hyper parameters, etc.) have **smooth, predictable structure as you scale.**
  - **Diverse capabilities emerge** when training on internet-scale data
- These led a paradigm shift and a massive industrialization of our field.

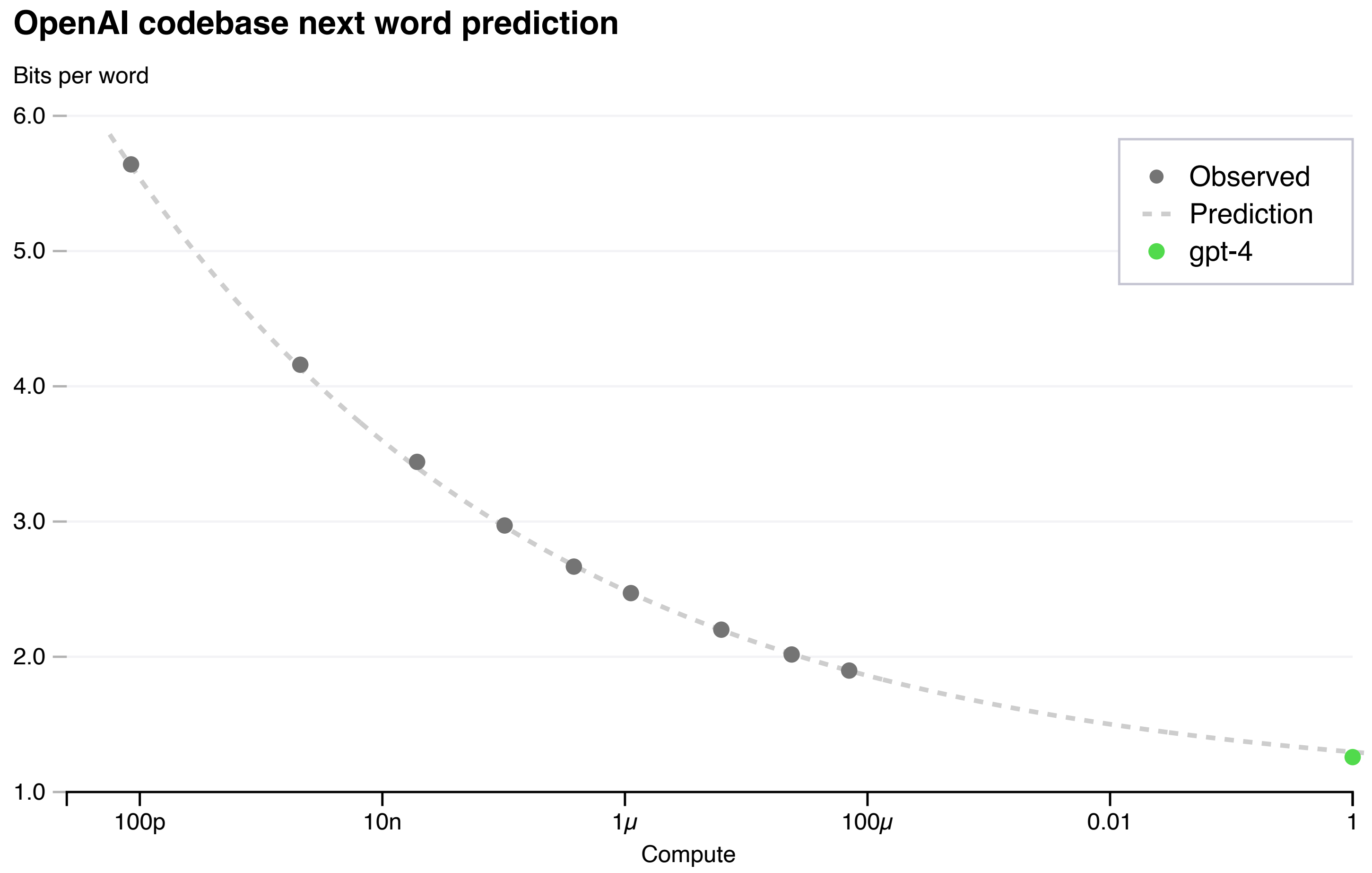
Paradigm shift

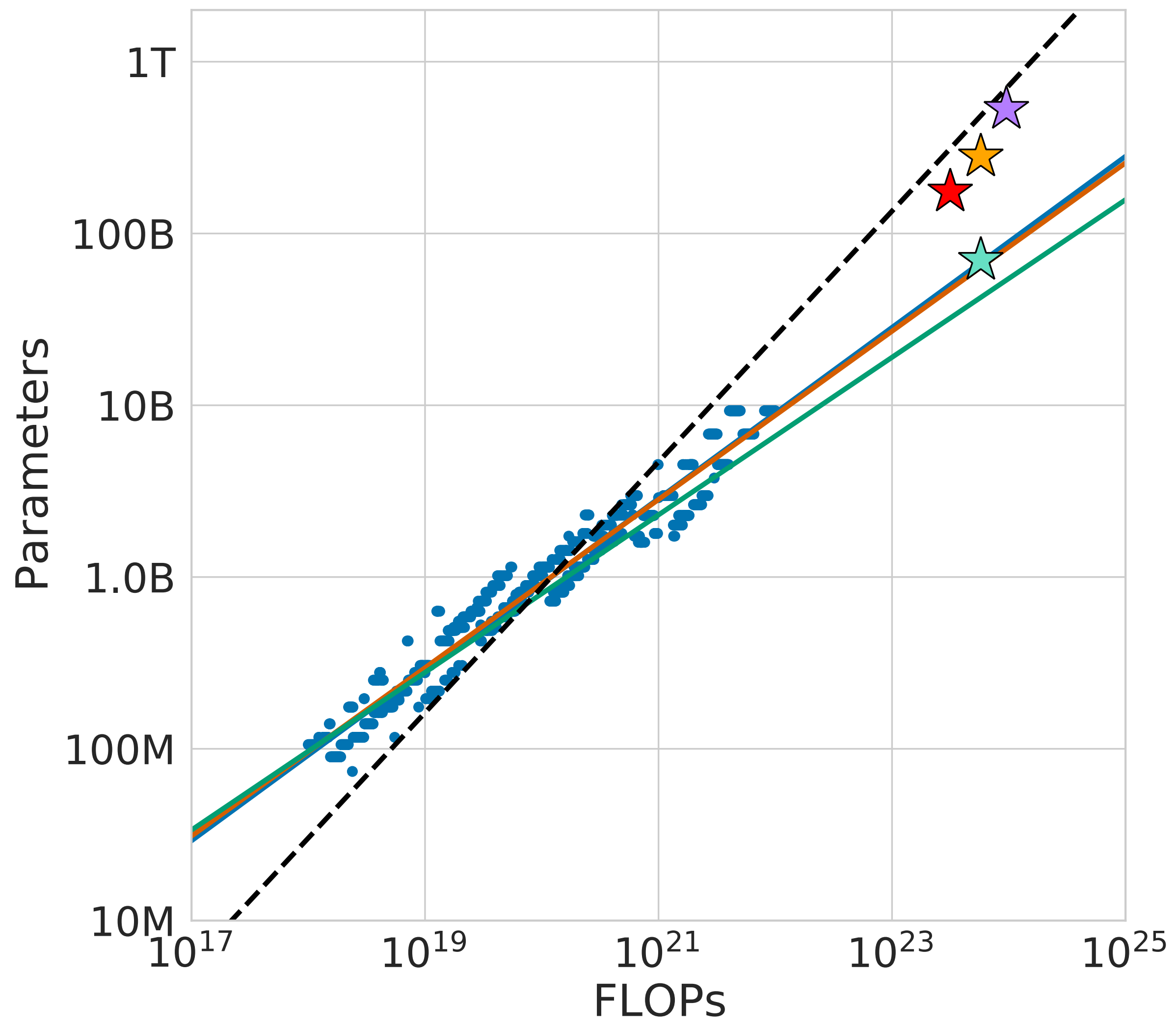
Predictable scaling motivated us to industrialize

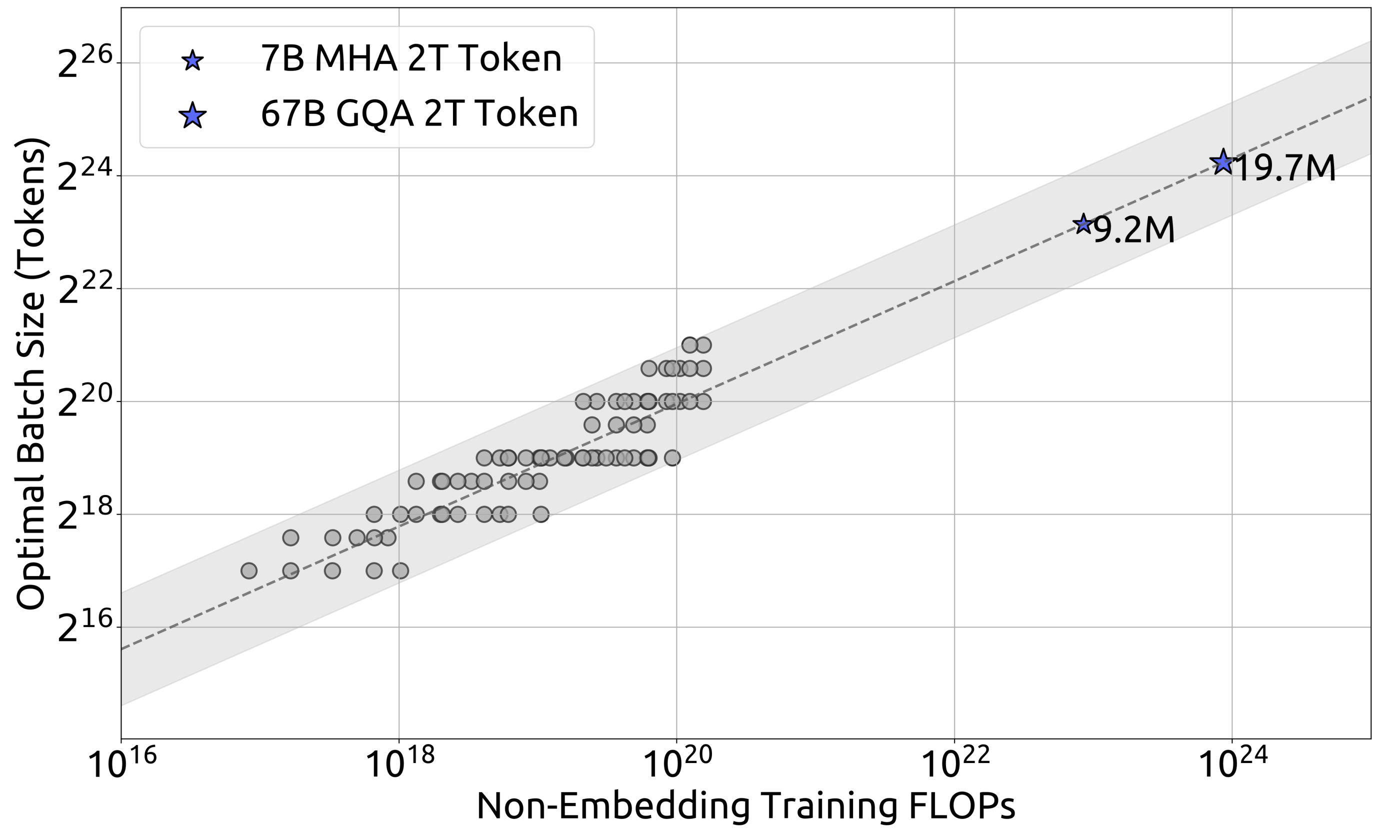
Capability prediction

Resource allocation

Hyperparameter selection







OpenAI, 2023. “GPT-4 Technical Report”

Hoffmann et al., 2022. “Training Compute-Optimal Large Language Models”

Bi et al., 2024. “DeepSeek LLM Scaling Open-Source Language Models with Longtermism”

Slide credit: Yangjun Ruan

Paradigm shift

Pre-scaling paradigm

- When I started in ML in 2011:
  - method-driven progress
  - single-task models
  - static datasets
- **Improving generalization was the job of the researcher** through clever models and methods that effectively reduced the dimension of the parameters.

Scaling paradigm

- In the scaling paradigm:
  - data-driven progress
  - massively multi-task
  - cookie-cutter methods
- **Generalization and capabilities are expanded by adding data and parameters**. The role of researchers is to find more data, like finding oil reserves.

The Bitter Lesson

by Rich Sutton

- Could compute be the key driver of progress in AI?
- Rich Sutton wrote about this in a 2019 essay titled “The Bitter Lesson”. He was comparing **two approaches to progress**:
  - researchers designing clever methods that capture knowledge of the data

vs.

  - compute invested into general-purpose algorithms
- **The “bitter lesson”, he argues, is that compute-driven approaches are winning over longer time scales.**

“One thing that should be learned from the bitter lesson is the great power of general purpose methods, of methods that continue to scale with increased computation even as the available computation becomes very great. The two methods that seem to scale arbitrarily in this way are *search* and *learning*.”

Rich Sutton

Thanks!