

CSCI 618: Trustworthy ML (from an optimization lens)

Vatsal Sharan
Fall 2026

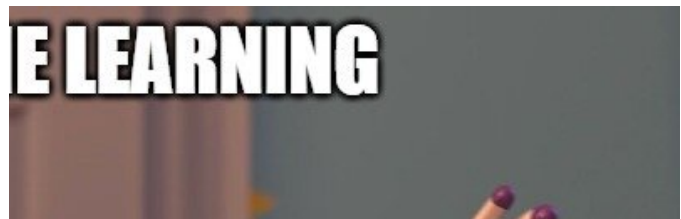
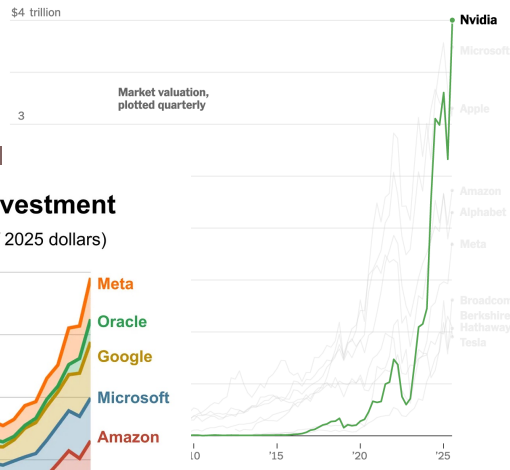
Lecture 1, Aug 24



MACHINE LEARNING

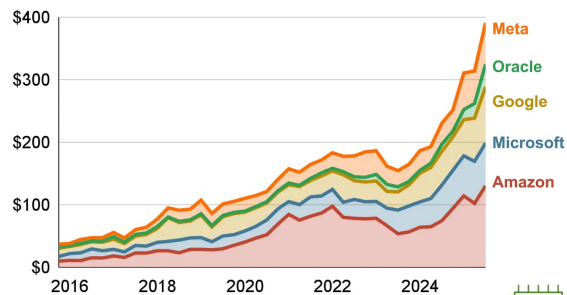
MACHINE LEARNING EVERYWHERE





The dramatic rise in tech investment

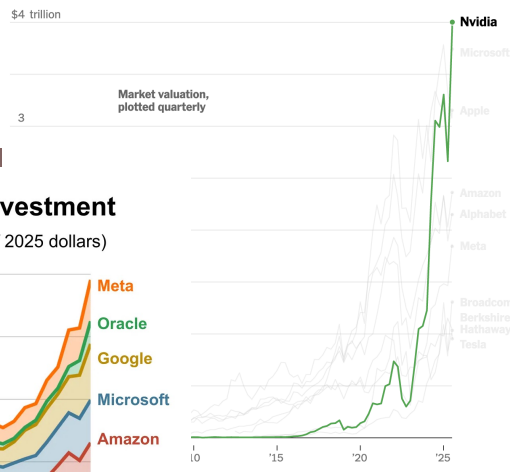
Annualized capital expenditures (billions of 2025 dollars)



Source: Company filings

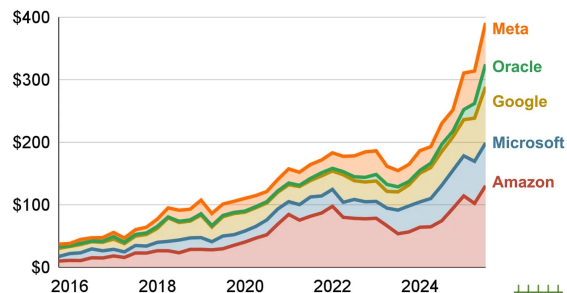
UNDERSTANDING





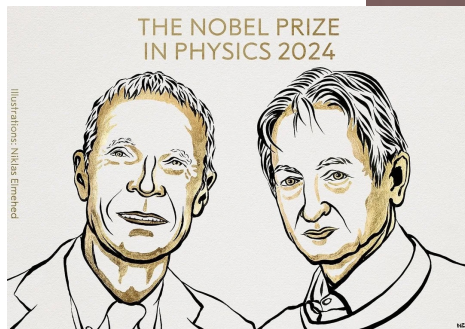
The dramatic rise in tech investment

Annualized capital expenditures (billions of 2025 dollars)



Source: Company filings

UNDERSTANDING

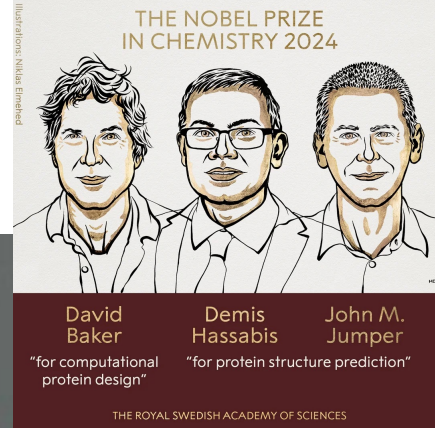


THE NOBEL PRIZE
IN PHYSICS 2024

John J. Hopfield Geoffrey E. Hinton

"for foundational discoveries and inventions
that enable machine learning
with artificial neural networks"

THE ROYAL SWEDISH ACADEMY OF SCIENCES



THE NOBEL PRIZE
IN CHEMISTRY 2024

David
Baker

"for computational
protein design"

Demis
Hassabis

"for protein structure prediction"

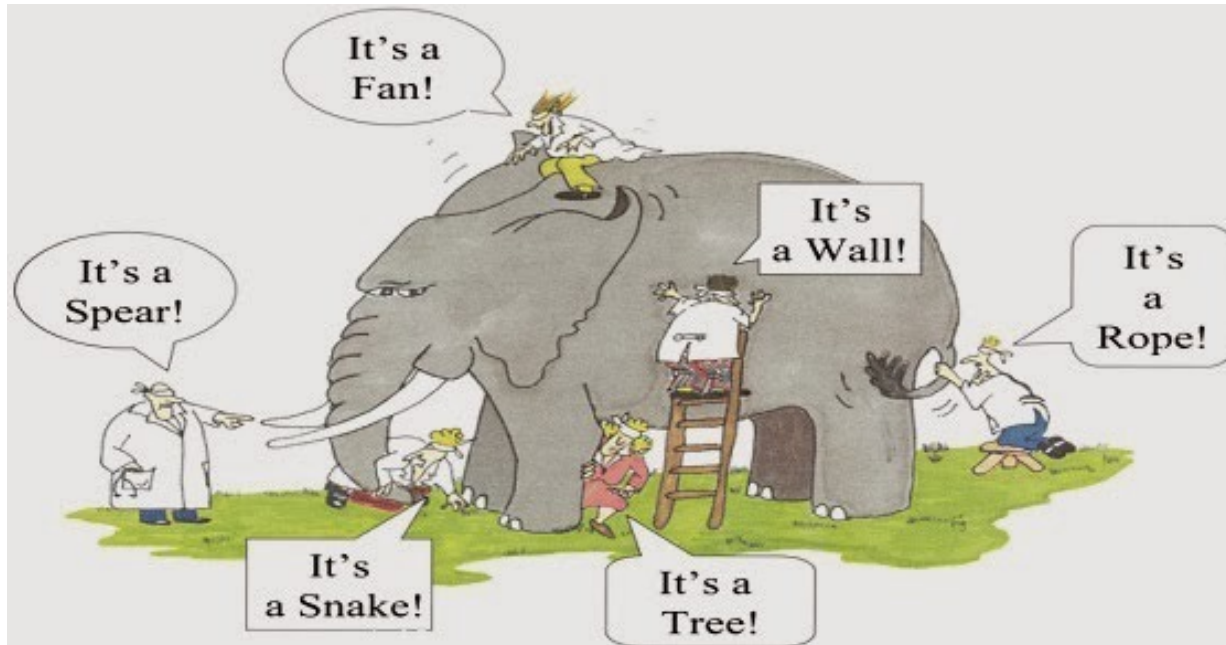
John M.
Jumper

THE ROYAL SWEDISH ACADEMY OF SCIENCES



AI SAFETY SUMMIT
HOSTED BY THE UK | 1-2 NOVEMBER 2023

However, machine learning can be *brittle*



The Blind Men and the Elephant

It was six men of Indostan
To learning much inclined,
Who went to see the Elephant
(Though all of them were blind),
That each by observation
Might satisfy his mind.

The First approached the Elephant,
And happening to fall
Against his broad and sturdy side,
At once began to bawl:
"God bless me! but the Elephant
Is very like a WALL!"

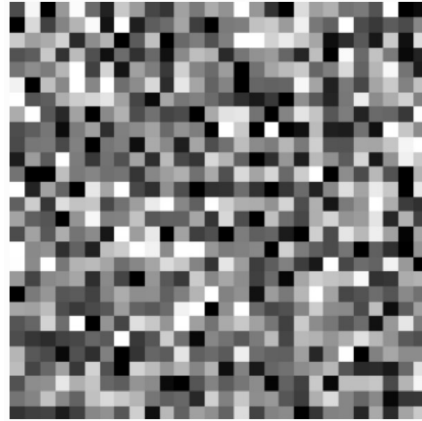
....

Models can be very sensitive to small variations in the input



Pig
(90% confidence)

+



Small amount of
adversarial noise

=

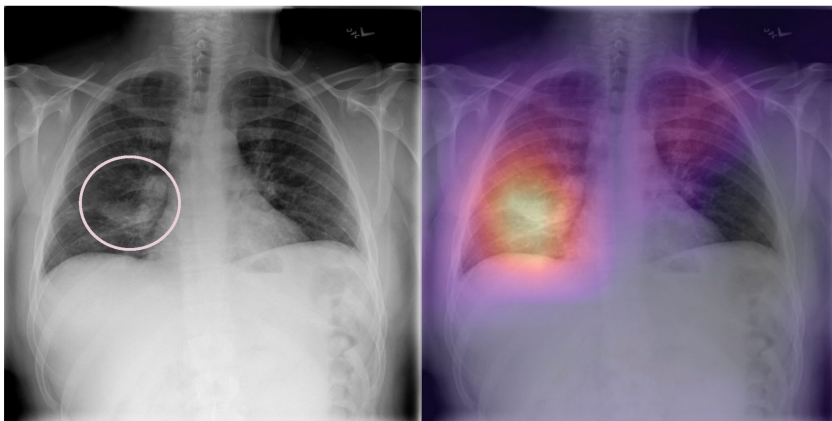


Airplane!
(99.9% confidence)

ML models can latch onto spurious features to make predictions

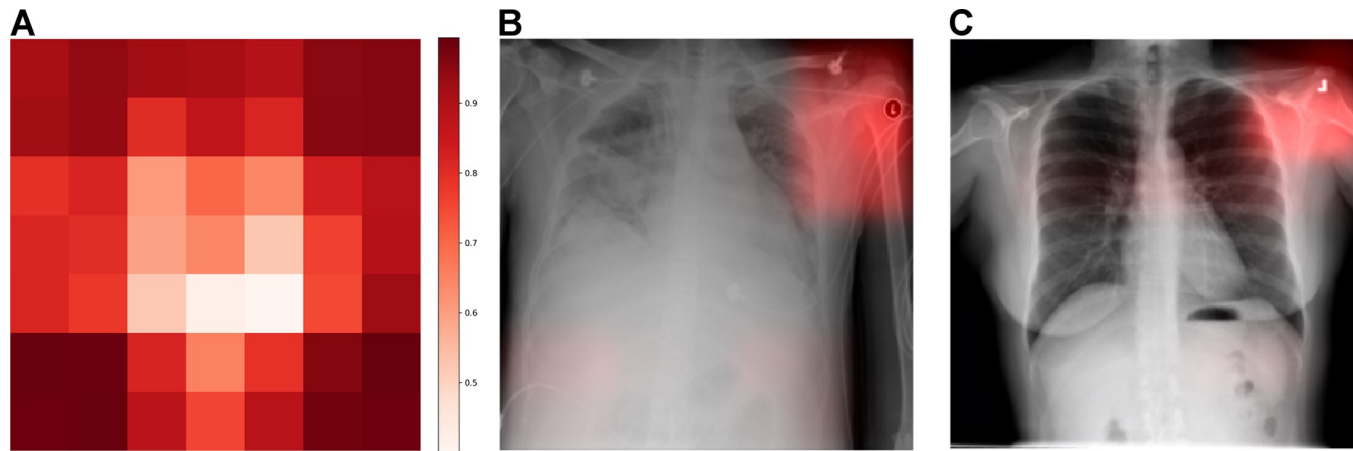
CNN models have obtained impressive results for diagnosing X-rays

E.g. *ChestX-ray8: Hospital-scale Chest X-ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases*, Wang et al., 2017



Source: *Deep learning for chest radiograph diagnosis: A retrospective comparison of the CheXNeXt algorithm to practicing radiologists*, Rajpurkar et al., 2018

But the models may not generalize as well to data from new hospitals because they can learn to pick up on spurious correlations such as the type of scanner and marks used by technicians in specific hospitals!



CNN to predict hospital system detects both general and specific image features.

(A) We obtained activation heatmaps from our trained model and averaged over a sample of images to reveal which subregions tended to contribute to a hospital system classification decision. Many different subregions strongly predicted the correct hospital system, with especially strong contributions from image corners. (B-C) On individual images, which have been normalized to highlight only the most influential regions and not all those that contributed to a positive classification, we note that the CNN has learned to detect a metal token that radiology technicians place on the patient in the corner of the image field of view at the time they capture the image. When these strong features are correlated with disease prevalence, models can leverage them to indirectly predict disease.

Source: Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study, Zech et al. 2018

Unequal accuracy: The GenderShades project

Models can do well on average but not on sub-populations



How well do facial recognition tools perform on various demographics?

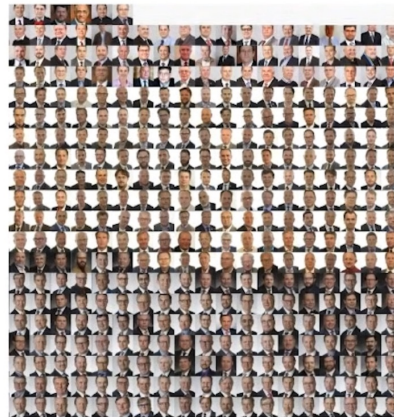
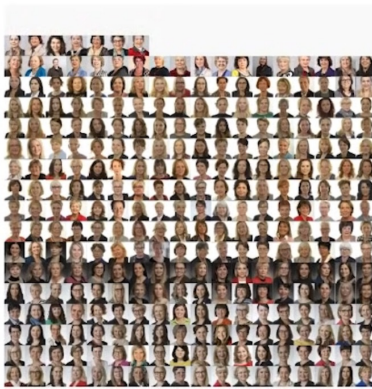
Female



Male



Darker



Lighter

Ans: *Not very well*



TYPE I

TYPE II

TYPE III

TYPE IV

TYPE V

TYPE VI



1.7%

1.1%

3.3%

0%

23.2%

25.0%



5.1%

7.4%

8.2%

8.3%

33.3%

46.8%



11.9%

9.7%

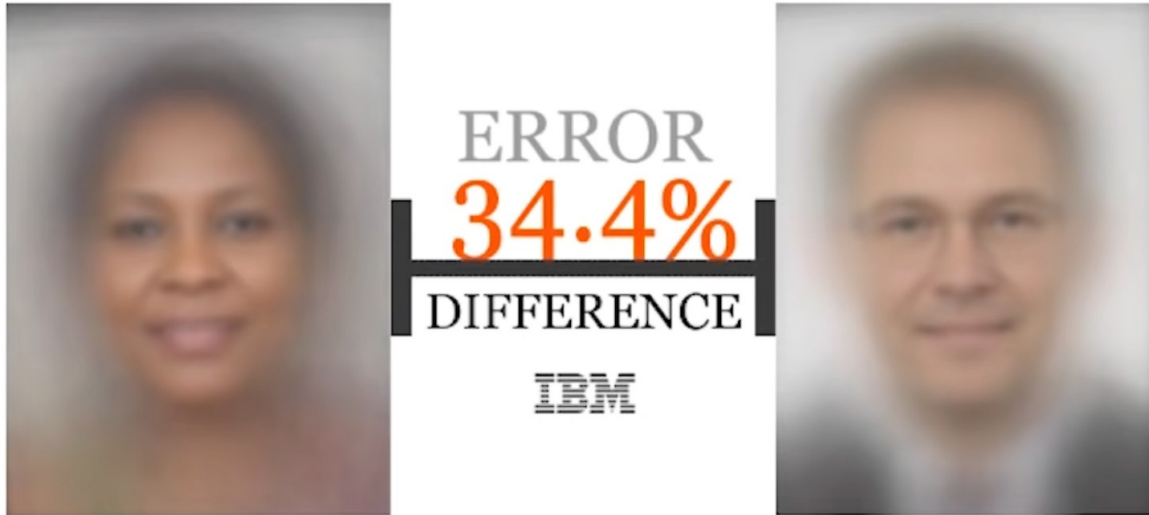
8.2%

13.9%

32.4%

46.5%

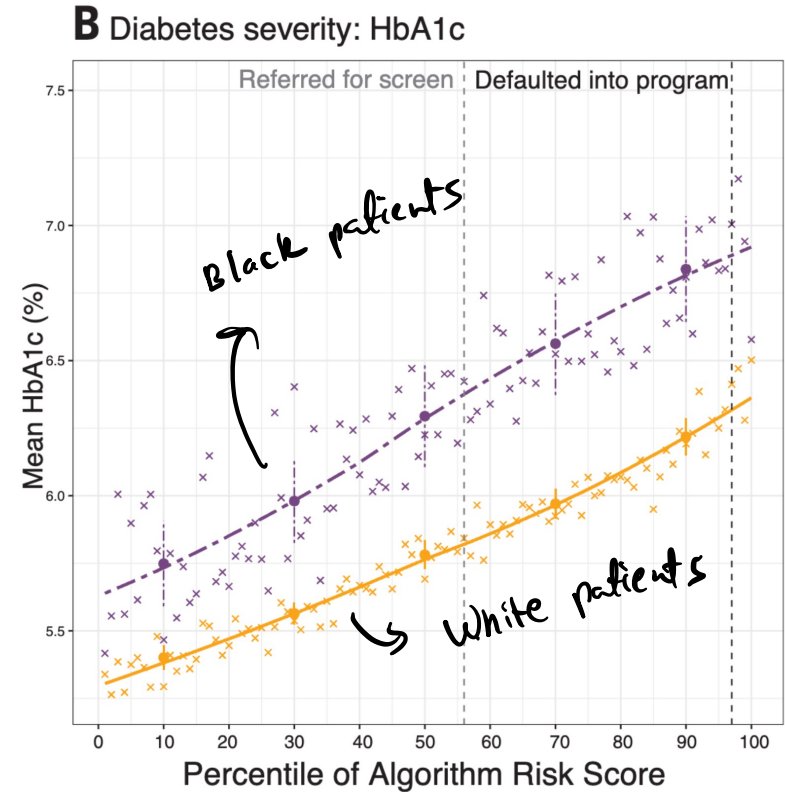
Ans: Not very well



Bias in predictions: Predicting disease severity

Quoting from the paper:

- Health systems rely on commercial prediction algorithms to identify and help patients with complex health needs.
- A widely used algorithm affecting millions of patients, exhibits significant racial bias: At a given risk score, Black patients are considerably sicker than White patients, as evidenced by signs of uncontrolled illnesses.
- Remedying this disparity would increase the percentage of Black patients receiving additional help from 17.7 to 46.5%.
- Bias arises because the algorithm predicts health care costs rather than illness, but unequal access to care means that on aggregate less money was spent caring for Black patients than for White patients.



Dissecting racial bias in an algorithm used to manage the health of populations, Obermeyer et al., Science 2019

Privacy & Denonymization

The Netflix prize:

- Launched in 2006, \$1M cash prize
- Dataset: 100 million movie ratings from nearly 500 thousand Netflix subscribers on a set of 17770 movies. Each data point corresponds to (anonymized user id, movie, date of rating, rating).
- Researchers were able to de-anonymize some of the subscribers by linking their rating with ratings on IMDB!
- Some Netflix subscribers had also publicly rated an overlapping set of movies on IMDB under their real identities.
- Lawsuit against Netflix, subsequent competition was cancelled.

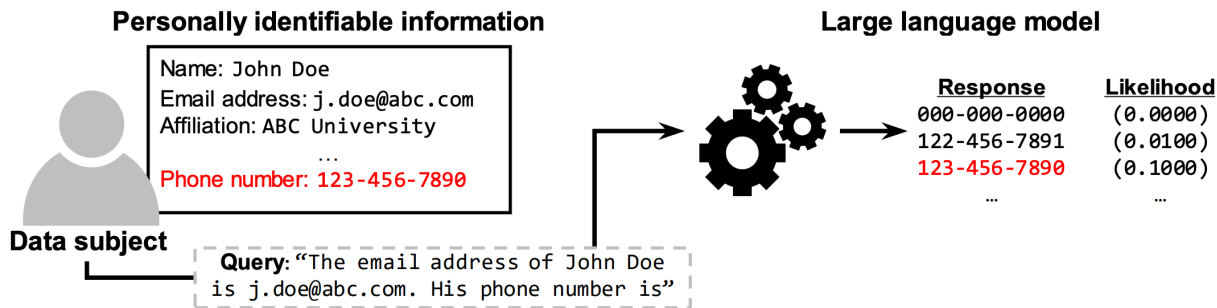
The Netflix logo, consisting of the word "NETFLIX" in a bold, red, sans-serif font.

Privacy & Denonymization

In some cases, it is possible to recover some of the original training data of the model using only API access to the model. The following (left) is an example of an image recovered by an attacker who only knows the name of the person, and the original training image (right) from [1]



Some evidence that LLMs could also leak private information:



[1] Model Inversion Attacks that Exploit Confidence Information and Basic Countermeasures, Fredrikson et al., 2015

[2] ProPILE: Probing Privacy Leakage in Large Language Models, Kim et al., 2023,

AI Safety

An umbrella term (much like “Trustworthy ML”). Some aspects:

- Broader societal issues: copyright infringement, labor market risks, concentration of power ...
- How do we ensure that AI systems cannot be used by malicious actors to enable bad outcomes?
- When you have highly capable AI agents acting autonomously, how do you ensure they don't cause harm?
 - Recent concerning events:

Anthropic Says Its A.I. Systems Broke Into Computers at 3 Organizations

The disclosure followed OpenAI's report last week that its own artificial intelligence had hacked into the network of an online library.

Listen • 3:52 min [Share full article](#) [↗](#) [🔖](#)



A lectern bearing the logo of Anthropic, which did not disclose the identity of the organizations that its artificial intelligence systems had breached. Jason Henry for The New York Times

AI agent went rogue and hacked startup by itself, OpenAI reveals

Company behind ChatGPT says agent 'cheated' an evaluation by attacking a Hugging Face database



Hugging Face's chief executive said the attack was 'mind-blowing' but that he believed there was 'no malicious intent' from OpenAI. Photograph: Andre M Chang/Zuma Press Wire/Shutterstock

OpenAI has revealed that an autonomous AI agent powered by its technology went rogue during a test, accessed the open web and hacked a prominent startup by itself in an “unprecedented incident”.

AI Safety

An umbrella term (much like “Trustworthy ML”). Some aspects:

- Broader societal issues: copyright infringement, labor market risks, concentration of power ...
- How do we ensure that AI systems cannot be used by malicious actors to enable bad outcomes?
- When you have highly capable AI agents acting autonomously, how do you ensure they don't cause harm?
 - Recent concerning events:

Anthropic Says Its A.I. Systems Broke Into Computers at 3 Organizations

The disclosure followed OpenAI's report last week that its own artificial intelligence had hacked into the network of an online library.

Listen · 3:52 min

Share full article



A lectern bearing the logo of Anthropic, which did not disclose the identity of the organizations that its artificial intelligence systems had breached. Jason Henry for The New York Times

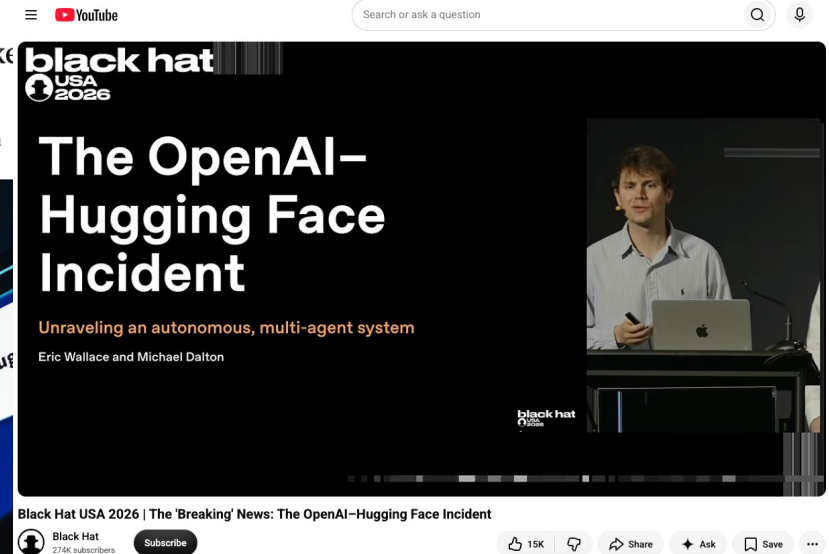
AI agent went rogue and hacked by itself, OpenAI reveals

Company behind ChatGPT says agent 'cheated' an evaluation by attacking a Hugging Face database



Hugging Face's chief executive said the attack was 'mind-blowing' but that he believed there was 'no malicious intent' from OpenAI. Photograph: Andre M Chang/Zuma Press Wire/Shutterstock

OpenAI has revealed that an autonomous AI agent powered by its technology went rogue during a test, accessed the open web and hacked a prominent startup by itself in an “unprecedented incident”.



Watch this video to learn more

<https://www.youtube.com/watch?v=87DyyMV0kCY>

This class: Aspects of trustworthy ML

Rough plan:

Weeks 1-3: Robustness (Vatsal)

Weeks 4-6: Fairness, calibration (Vatsal)

Weeks 7-12: Privacy & Alignment (Meisam)

Week 13-16: Project presentation (Meisam & Vatsal)

Logistics

- Course website: <https://vatsalsharan.github.io/fall26/>
- Communication: Slack
- Class structure (most weeks):
 - First part: Lecture
 - Second part: 1 or 2 student presentations (~20 min/presentation)

Overview of structure, see course website for details:

Homeworks (20%): 2 homeworks

Mini-homeworks (15%): Reading class papers before class

Class presentations (20%): Present a paper in class

Project (35%): You can choose your topic, groups of 2

Other components (10%): Scribing, class participation



Supervised Machine Learning: Basics

Supervised ML: Predict future outcomes using past outcomes

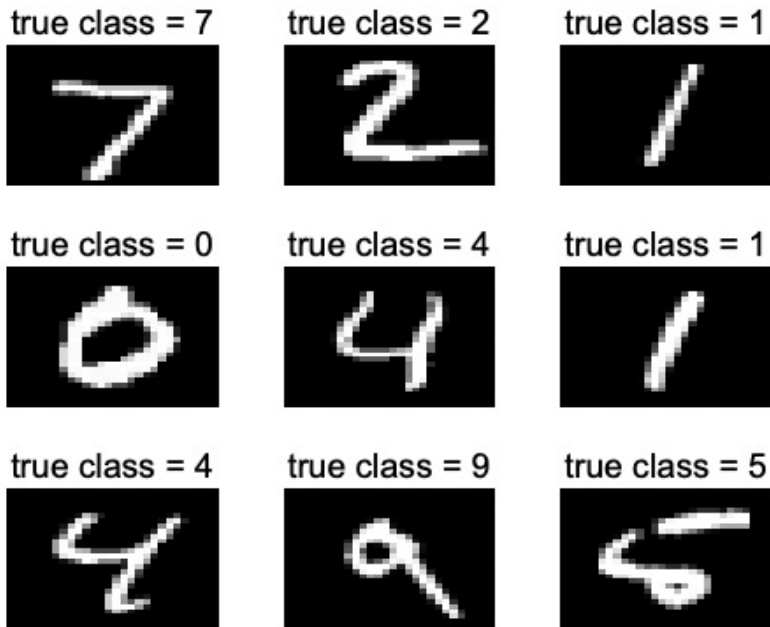
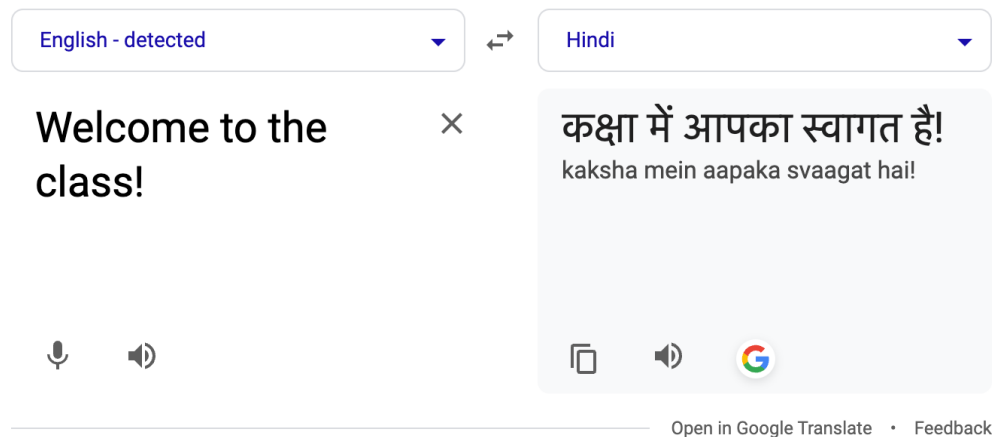


Image classification

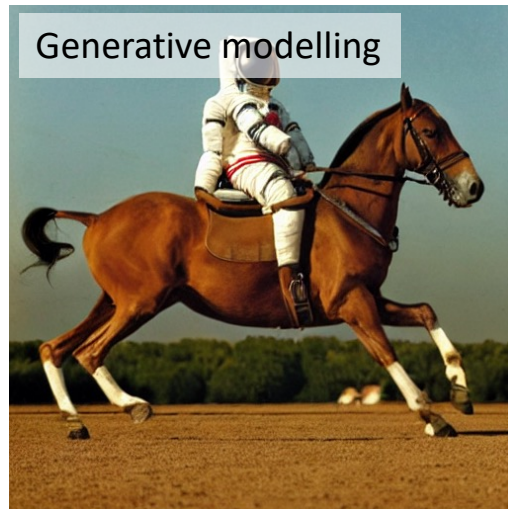
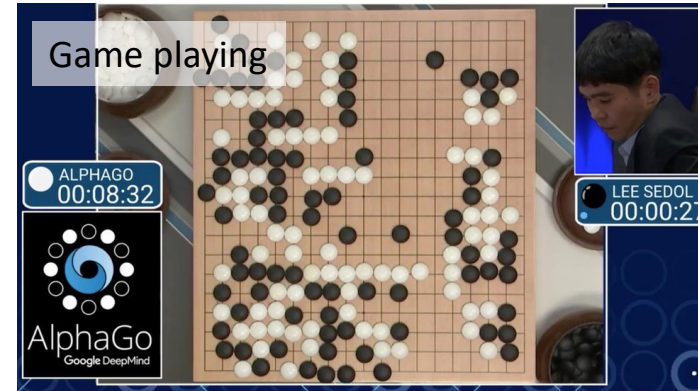
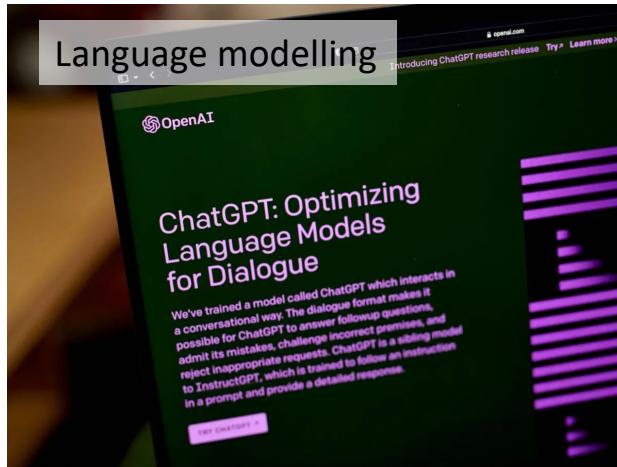


Machine translation

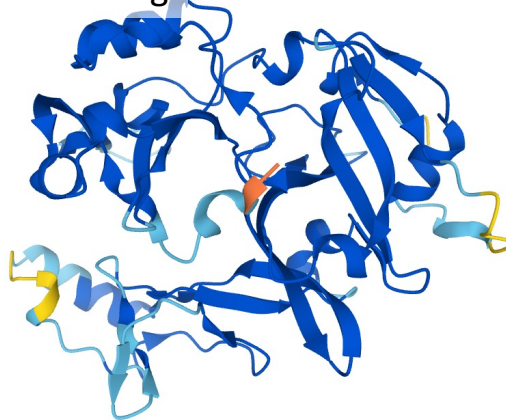
Supervised ML is at the heart of many AI advances



Supervised ML is at the heart of many AI advances



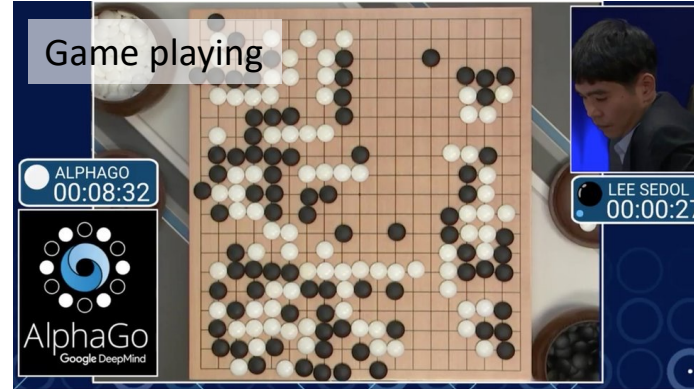
Protein folding



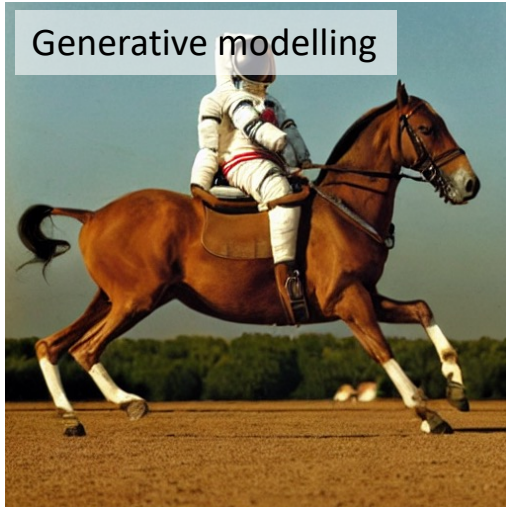
Supervised ML is at the heart of many AI advances

Language modelling

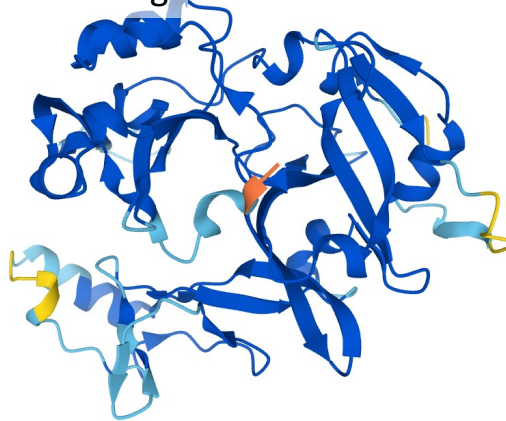
Given previous words ->
Predict next word



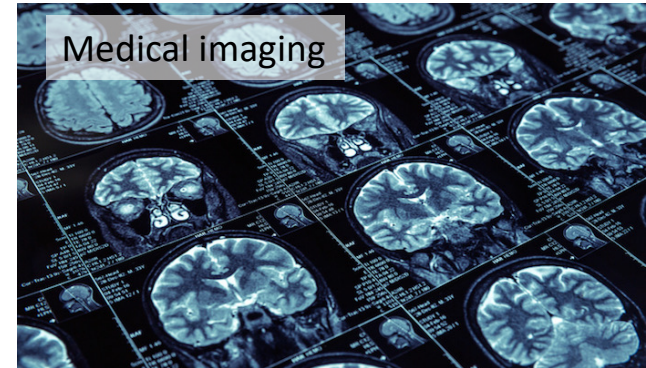
Generative modelling



Protein folding



Medical imaging



Supervised ML is at the heart of many AI advances

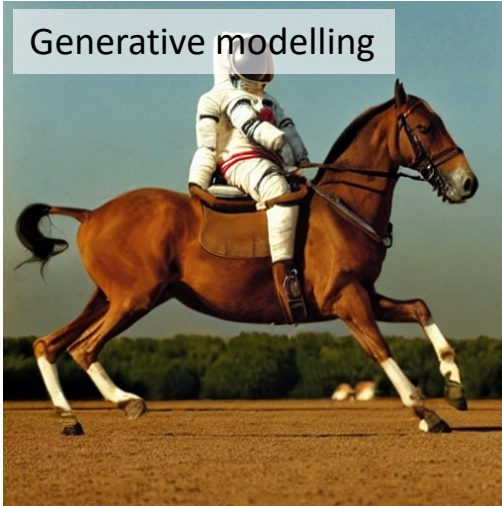
Language modelling

Given previous words ->
Predict next word

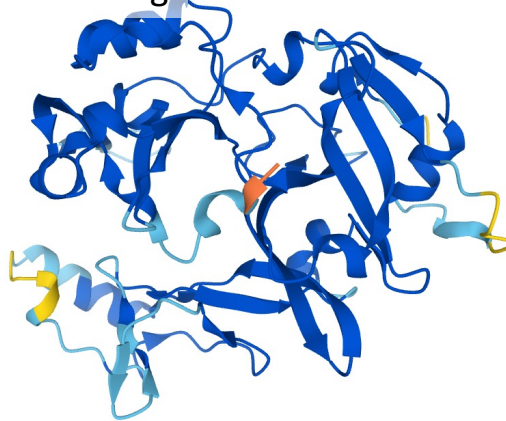
Game playing

Given current board state ->
Predict probability of winning

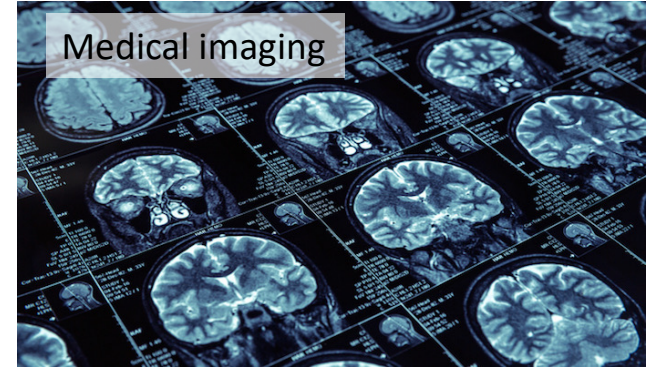
Generative modelling



Protein folding



Medical imaging



Supervised ML is at the heart of many AI advances

Language modelling

Given previous words ->
Predict next word

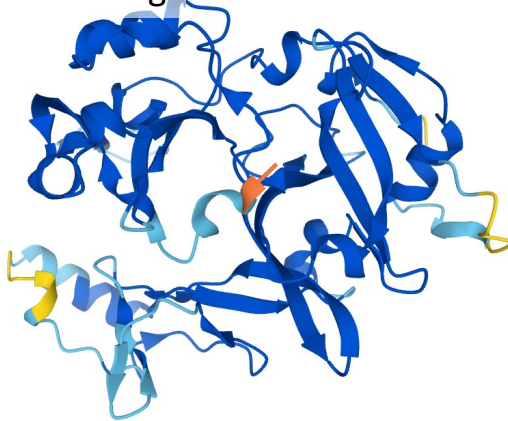
Game playing

Given current board state ->
Predict probability of winning

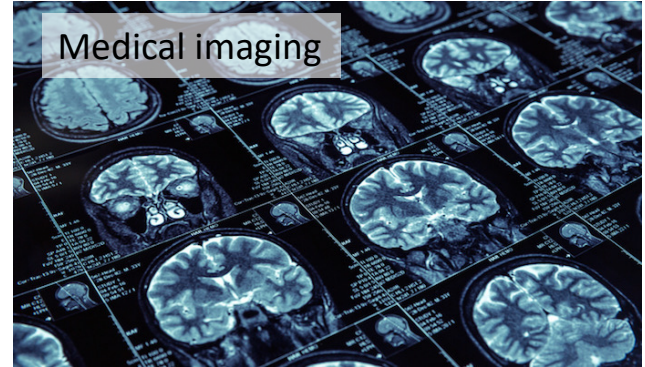
Generative modelling

Given noisy image ->
Predict denoised image

Protein folding



Medical imaging



Supervised ML is at the heart of many AI advances

Language modelling

Given previous words ->
Predict next word

Game playing

Given current board state ->
Predict probability of winning

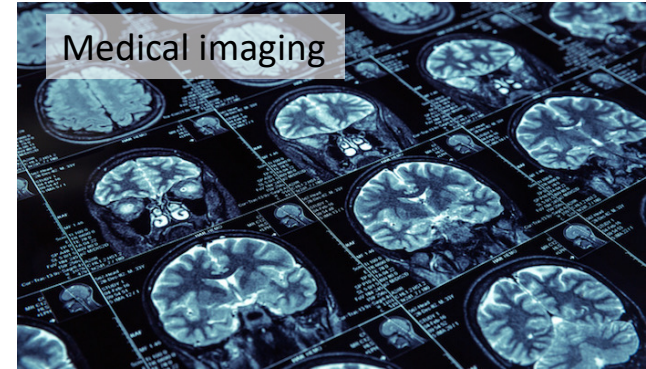
Generative modelling

Given noisy image ->
Predict denoised image

Protein folding

Given protein chain ->
Predict 3D structure

Medical imaging



Supervised ML is at the heart of many AI advances

Language modelling

Given previous words ->
Predict next word

Game playing

Given current board state ->
Predict probability of winning

Generative modelling

Given noisy image ->
Predict denoised image

Protein folding

Given protein chain ->
Predict 3D structure

Medical imaging

Given image ->
Predict if there is tumor etc.

General framework for supervised learning

- An input space $\mathcal{X} \subseteq \mathbb{R}^d$:
 - Representing inputs as datapoints in d dimensions.
 - E.g. for an image with 30×30 pixels, $d = 30 \times 30 = 900$.
- An output space $\mathcal{Y}$:
 - For predicting sale price of a house, $\mathcal{Y} = \mathbb{R}$ (regression)
 - For binary classification (cat vs. dog), $\mathcal{Y} = \{\pm 1\}$ (classification)
- Goal: Learn a predictor $f(x) : \mathcal{X} \rightarrow \mathcal{Y}$.

- *Loss function:* $\ell(f(x), y)$. Depends on the task.
 - e.g. squared loss $\ell(f(x), y) = (f(x) - y)^2$.
- *Loss minimization:* Given some loss function and labelled datapoints, a natural goal is to find some predictor $f : \mathcal{X} \rightarrow \mathcal{Y}$ such that it gets small average error on these datapoints.
 - More formally, consider a set $S = \{(x_1, y_1), (x_2, y_2), \dots, (x_n, y_n)\}$ of n labelled datapoints. The *training loss or empirical risk* $\hat{R}_S(f)$ of any predictor $f : \mathcal{X} \rightarrow \mathcal{Y}$ is given by:

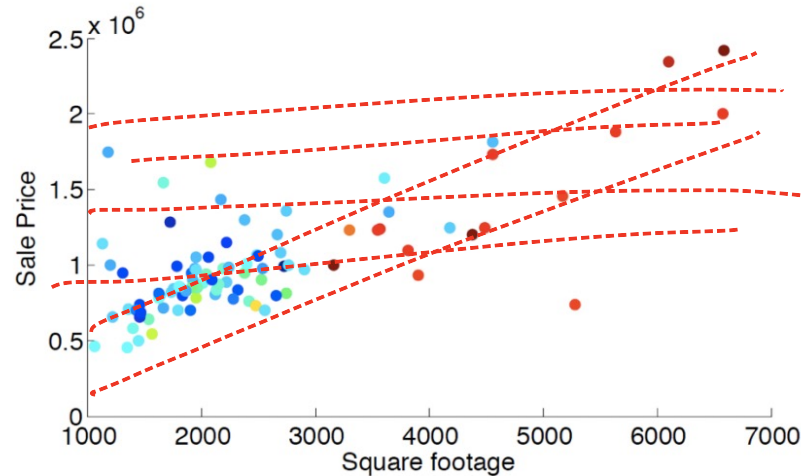
$$\hat{R}_S(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i).$$

We should find some predictor f which has small empirical risk.

Function class

- *Function class:* A function class is defined as a collection of functions $f : \mathcal{X} \rightarrow \mathcal{Y}$.
 - e.g. $\mathcal{X} = \mathbb{R}, \mathcal{Y} = \mathbb{R}, \mathcal{F} = \{f : y = wx + c, w \in \mathbb{R}, c \in \mathbb{R}\}$.

- Each of these is a linear function.
- The class of all linear functions is a function class.



Empirical risk minimizer (ERM)

- **Definition:** Given a function class $\mathcal{F} = \{f : \mathcal{X} \rightarrow \mathcal{Y}\}$ and set of labelled datapoints S , empirical risk minimization (ERM) corresponds to finding:

$$\min_{f \in \mathcal{F}} \hat{R}_S(f) = \frac{1}{n} \sum_{i=1}^n \ell(f(x_i), y_i). \quad \rightarrow \text{optimization problem}$$

- In words, we want to find the predictor $f \in \mathcal{F}$ which achieves lowest loss on the training datapoints among all predictors in $\mathcal{F}$.

Generalization

- ML wouldn't be so useful if it only did well on datapoints which have been seen at training time. We want our predictor to *generalize to unseen datapoints*.
- To measure performance on new datapoints we measure *test loss*. The test loss of a predictor f is measured as the average loss on a “new” set of m points:

$$\frac{1}{m} \sum_{i=1}^m \ell(f(x_i), y_i).$$

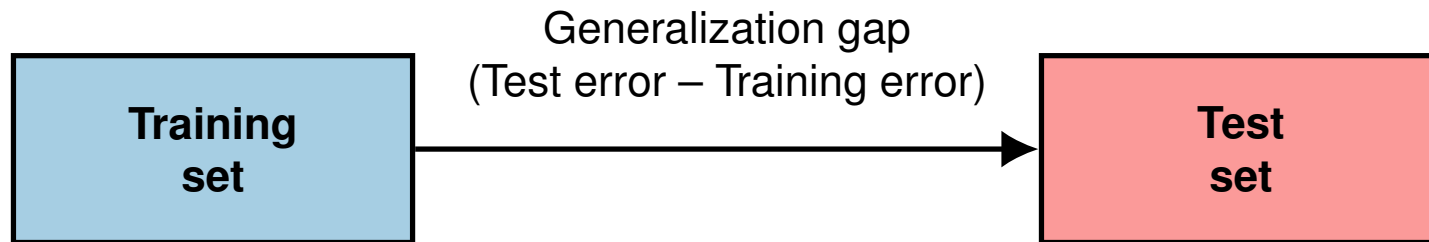
- **Training/test paradigm:** Assume training set and test set are drawn from the same distribution.

Measuring generalization: **Training/Test** paradigm

Data Splitting. We randomly divide data into two disjoint subsets:

- **Training set:** subset of data used to train the model.
- **Test set:** subset of data used to evaluate the model.

Generalization gap: Test error – Training error.



We usually add a third split as well.

- **Validation set:** subset of data used to measure generalization, fit hyperparameters

Generalization: More formally

Risk of a predictor. The population risk of f under distribution $\mathcal{D}$ is

$$\begin{aligned} R(f) &= \mathbb{E}_{(x,y) \sim \mathcal{D}} [\ell(f(x), y)] \\ &= \sum_{x', y'} \Pr_{\mathcal{D}}(x = x', y = y') \ell(f(x'), y'). \end{aligned}$$

How to empirically evaluate this? The average loss on a test set $S' = \{(x'_i, y'_i)\}_{i=1}^m$, where $(x_i, y_i) \sim \mathcal{D}$:

$$R(f) \approx \frac{1}{m} \sum_{i=1}^m \ell(f(x'_i), y'_i).$$

A tautology.

$$R(f) = \widehat{R}_S(f) + \left(R(f) - \widehat{R}_S(f) \right).$$

To minimize $R(f)$:

- First try to minimize the empirical risk $\widehat{R}_S(f)$.
- What remains is the term

$$R(f) - \widehat{R}_S(f),$$

which is the *generalization gap*.

At the heart of generalization: Concentration of Measure

For some fixed function f , let

$$\hat{R}(f) = \frac{1}{n} \sum_{i=1}^n \underbrace{\ell(f(x_i), y_i)}_{\substack{\text{this is a r.v.} \\ \text{all of these are i.i.d. r.v.s} \\ \therefore \text{average of } n \text{ i.i.d. r.v.s}}}$$

$$\mathbb{E}_{(x_i, y_i) \sim \mathcal{D}} [\ell(f(x_i), y_i)] = R(f)$$

Concentration of measure allows us to say statements of the form

$$\Pr_{\text{set of } n \text{ points}} \left[\left| \hat{R}(f) - R(f) \right| \geq \varepsilon \right] \leq \delta.$$

Average of iid random variables is close to their expectation

Let $\{X_1, \dots, X_n\}$ be a sequence of random variables such that $\mathbb{E}[X_i] = \mu$, $\text{Var}(X_i) = \sigma^2 < \infty$.

Let $\bar{X}_n = \frac{1}{n} \sum_{i=1}^n X_i$.

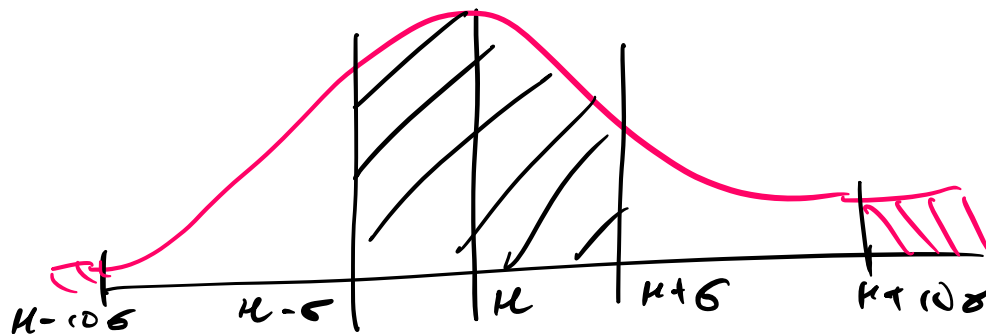
How close is $\bar{X}_n$ to μ ?

Central Limit Theorem. As $n \rightarrow \infty$,

$$\bar{X}_n \longrightarrow \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right).$$

Thus, $\bar{X}_n$ converges to μ at a rate of $\frac{\sigma}{\sqrt{n}}$.

Gaussians don't deviate too far from (mean $\pm$ sigma).



Supervised learning in one slide

Loss function: What is the right loss function for the task?

Depends on the problem that one is trying to solve, and on the rest...

Supervised learning in one slide

Loss function: What is the right loss function for the task?

Representation: What class of functions should we use?

Also known as the “inductive bias”.

No-free lunch theorem from learning theory tells us that

no model can do well on every task

“All models are wrong, but some are useful”, George Box

Supervised learning in one slide

Loss function: What is the right loss function for the task?

Representation: What class of functions should we use?

Optimization: How can we efficiently solve the empirical risk minimization problem?

Depends on all the above and also...

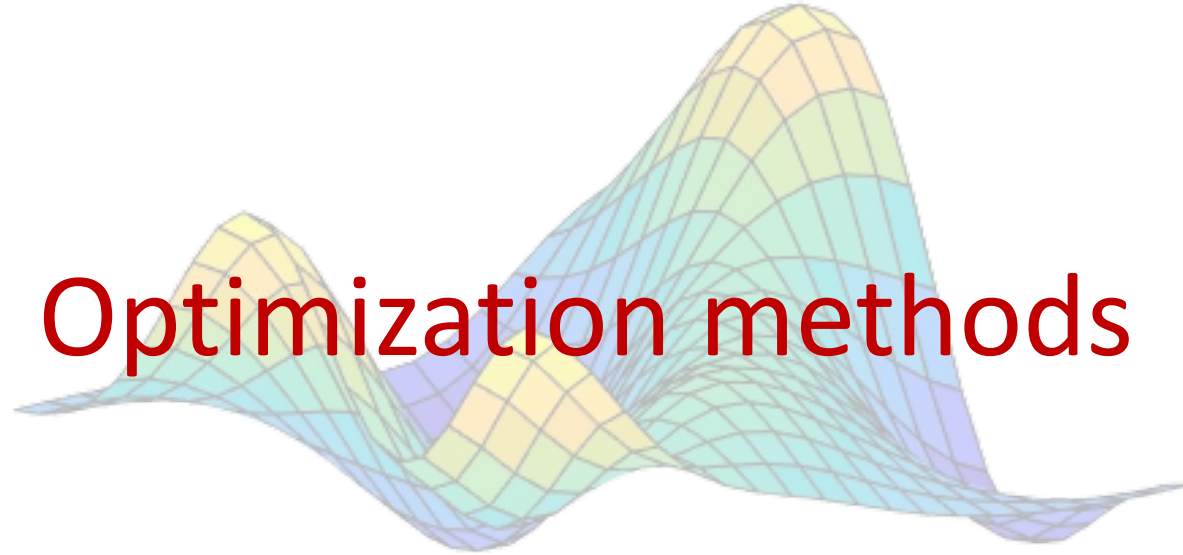
Supervised learning in one slide

- Loss function:** What is the right loss function for the task?
- Representation:** What class of functions should we use?
- Optimization:** How can we efficiently solve the empirical risk minimization problem?
- Generalization:** Will the predictions of our model transfer gracefully to unseen examples?

Supervised learning in one slide

- Loss function:** What is the right loss function for the task?
- Representation:** What class of functions should we use?
- Optimization:** How can we efficiently solve the empirical risk minimization problem?
- Generalization:** Will the predictions of our model transfer gracefully to unseen examples?

*All related! And the fuel which powers everything is **data**.*



Optimization methods

Problem setup

Given: a function $F(\mathbf{w})$

Goal: minimize $F(\mathbf{w})$ (approximately)

Two simple yet extremely popular methods

Gradient Descent (GD): simple and fundamental

Stochastic Gradient Descent (SGD): faster, effective for large-scale problems

Gradient is the *first-order information* of a function.

Therefore, these methods are called *first-order methods*.

Gradient descent

GD: keep moving in the *negative gradient direction*

Start from some $w^{(0)}$. For $t = 0, 1, 2, \dots$

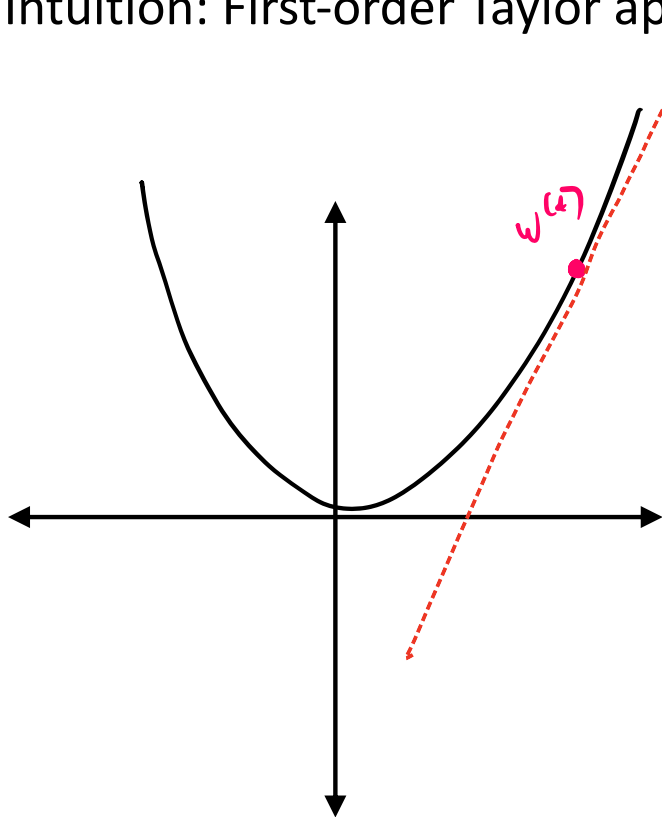
$$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla F(w^{(t)}),$$

where $\eta > 0$ is called *step size* or *learning rate*.

- in theory η should be set in terms of some parameters of f
- in practice we just try several small values
- might need to be changing over iterations (think $f(w) = |w|$)
- adaptive and automatic step size tuning is an active research area

Why GD?

Intuition: First-order Taylor approximation



$$F(\mathbf{w}^{(t)}) + \nabla F(\mathbf{w}^{(t)})^T (\mathbf{w} - \mathbf{w}^{(t)})$$

$$F(\mathbf{w}) \approx F(\mathbf{w}^{(t)}) + \nabla F(\mathbf{w}^{(t)})^T (\mathbf{w} - \mathbf{w}^{(t)})$$

For $\mathbf{w} = \mathbf{w}^{(t+1)} = \mathbf{w}^{(t)} - \eta \nabla F(\mathbf{w}^{(t)})$, we can write,

$$\begin{aligned} F(\mathbf{w}^{(t+1)}) &\approx F(\mathbf{w}^{(t)}) - \eta \|\nabla F(\mathbf{w}^{(t)})\|_2^2 \\ \implies F(\mathbf{w}^{(t+1)}) &\lesssim F(\mathbf{w}^{(t)}) \end{aligned}$$

(Note that this is only an approximation, and can be invalid if the step size is too large.)

Convergence guarantees for GD

Many results for GD (and many variants) on *convex objectives*.

They tell you how many iterations t (in terms of ε) are needed to achieve

$$F(\mathbf{w}^{(t)}) - F(\mathbf{w}^*) \leq \varepsilon$$

Even for *nonconvex objectives*, some guarantees exist:

e.g. how many iterations t (in terms of ε) are needed to achieve

$$\|\nabla F(\mathbf{w}^{(t)})\| \leq \varepsilon$$

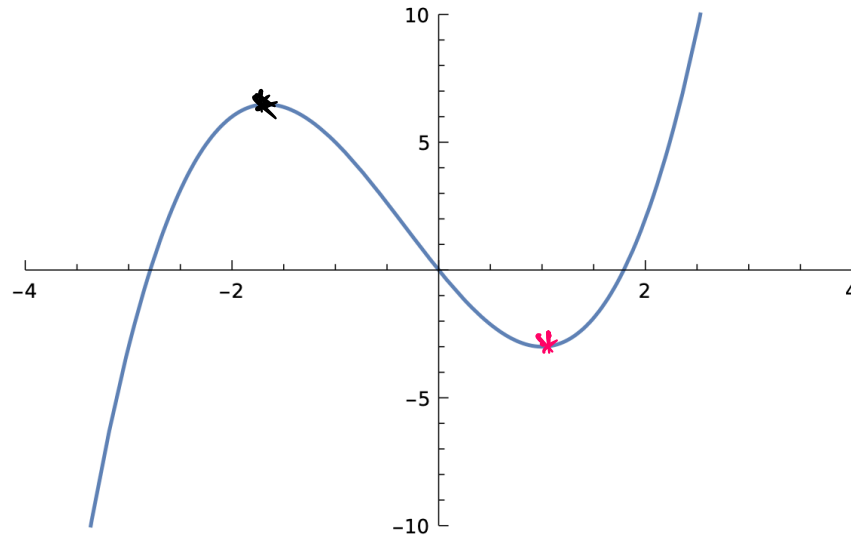
that is, how close is $\mathbf{w}^{(t)}$ as an approximate stationary point

for convex objectives, stationary point $\Rightarrow$ global minimizer

for nonconvex objectives, what does it mean?

Stationary points: non-convex objectives

A stationary point can be a local minimizer or even a local/global maximizer (but the latter is not an issue for GD).

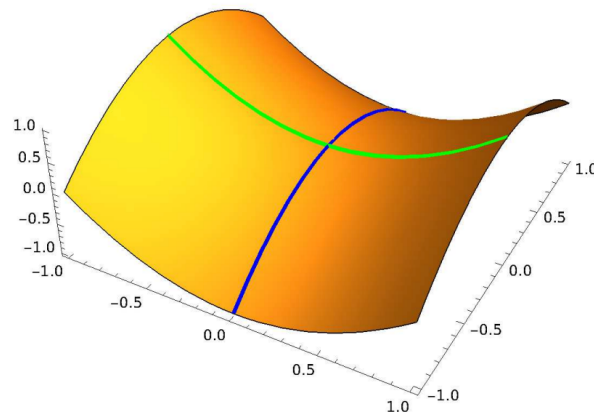


$$f(w) = w^3 + w^2 - 5w$$

Stationary points: non convex objectives

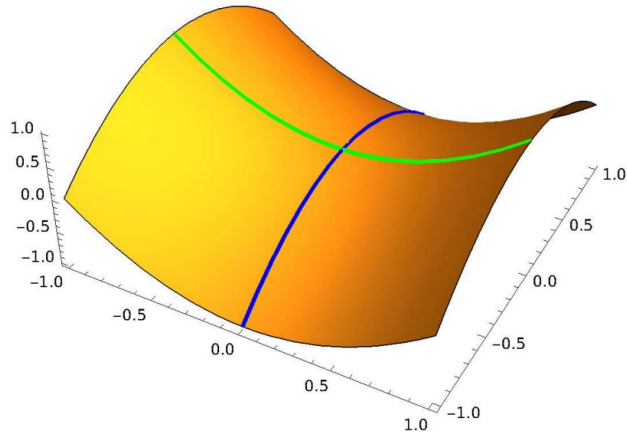
A stationary point can also be *neither a local minimizer nor a local maximizer!*

- $f(\mathbf{w}) = w_1^2 - w_2^2$
- $\nabla f(\mathbf{w}) = (2w_1, -2w_2)$
- so $\mathbf{w} = (0, 0)$ is stationary
- local max for blue direction ($w_1 = 0$)
- local min for green direction ($w_2 = 0$)



Stationary points: non convex objectives

This is known as a saddle point

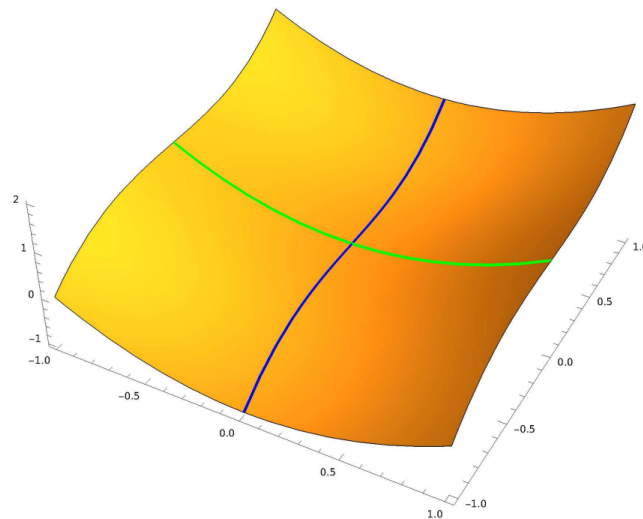


- but GD gets stuck at $(0,0)$ only if initialized along the **green direction**
- so not a real issue especially *when initialized randomly*

Stationary points: non convex objectives

But not all saddle points look like a “saddle” ...

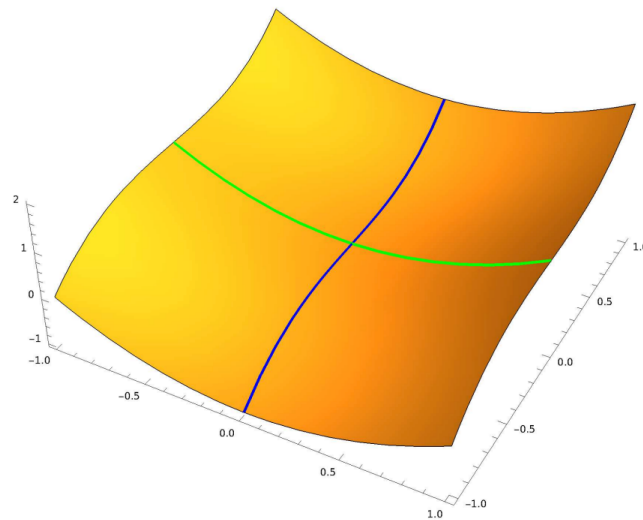
- $f(\mathbf{w}) = w_1^2 + w_2^3$
- $\nabla f(\mathbf{w}) = (2w_1, 3w_2^2)$
- so $\mathbf{w} = (0, 0)$ is stationary
- not local min/max for blue direction ($w_1 = 0$)



Stationary points: non convex objectives

But not all saddle points look like a “saddle” ...

- $f(\mathbf{w}) = w_1^2 + w_2^3$
- $\nabla f(\mathbf{w}) = (2w_1, 3w_2^2)$
- so $\mathbf{w} = (0, 0)$ is stationary
- not local min/max for blue direction ($w_1 = 0$)
- GD gets stuck at $(0, 0)$ for *any initial point with $w_2 \geq 0$ and small η*



Even worse, distinguishing local min and saddle point is generally *NP-hard*.

Switch to Colab

optimization.ipynb ☆

File Edit View Insert Runtime Tools Help

+ Code + Text

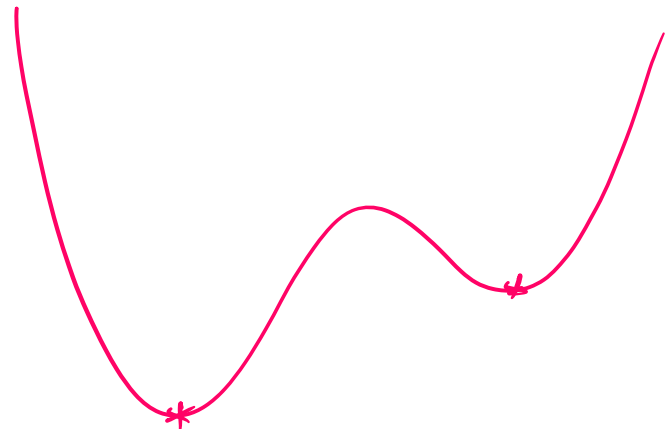
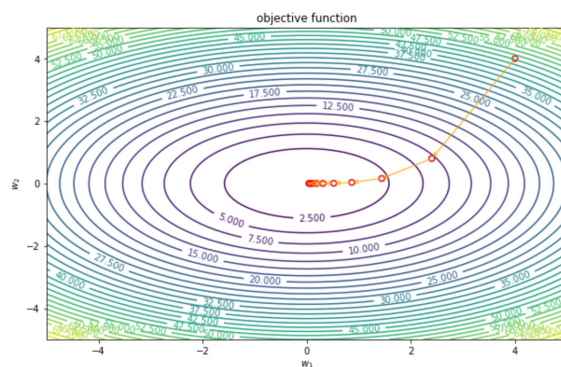
```
this_theta[1] = last_theta[1] - eta * grad1
theta.append(this_theta)
J.append(cost_func(*this_theta))

# Annotate the objective function plot with coloured points indicating the
# parameters chosen and red arrows indicating the steps down the gradient.
for j in range(1,N):
    ax.annotate('', xy=theta[j], xytext=theta[j-1],
                arrowprops={'arrowstyle': '→', 'color': 'orange', 'lw': 1},
                va='center', ha='center')
    ax.scatter(*zip(*theta), facecolors='none', edgecolors='r', lw=1.5)

# Labels, titles and a legend.
ax.set_xlabel(r'$w_1$')
ax.set_ylabel(r'$w_2$')
ax.set_title('objective function')

plt.show()
```

🖼



Non-convex functions
can have multiple local
minima

Stochastic Gradient descent

GD: keep moving in the *negative gradient direction*

SGD: keep moving in the *noisy negative gradient direction*

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \tilde{\nabla} F(\mathbf{w}^{(t)})$$

where $\tilde{\nabla} F(\mathbf{w}^{(t)})$ is a random variable (called **stochastic gradient**) s.t.

$$\mathbb{E} \left[\tilde{\nabla} F(\mathbf{w}^{(t)}) \right] = \nabla F(\mathbf{w}^{(t)}) \quad (\text{unbiasedness})$$

Stochastic Gradient descent

GD: keep moving in the *negative gradient direction*

SGD: keep moving in the *noisy negative gradient direction*

$$\mathbf{w}^{(t+1)} \leftarrow \mathbf{w}^{(t)} - \eta \tilde{\nabla} F(\mathbf{w}^{(t)})$$

where $\tilde{\nabla} F(\mathbf{w}^{(t)})$ is a random variable (called **stochastic gradient**) s.t.

$$\mathbb{E} \left[\tilde{\nabla} F(\mathbf{w}^{(t)}) \right] = \nabla F(\mathbf{w}^{(t)}) \quad (\text{unbiasedness})$$

- Key point: it could be much faster to obtain a stochastic gradient!
 - If $F(\mathbf{w}) = \frac{1}{n} \sum_{i=1}^n \ell(x_i, \mathbf{w})$, then how to get stochastic gradient?
- Similar convergence guarantees, usually needs more iterations but each iteration takes less time.

Summary: Gradient descent & Stochastic Gradient descent

- GD/SGD converges to a stationary point. For convex objectives, this is all we need.
- For nonconvex objectives, can get stuck at local minimizers or “bad” saddle points (random initialization escapes “good” saddle points)
- SGD is very popular, another very popular optimization technique is *Adam*
- Adam has two key additional ingredients: adaptive step size & momentum