ML Ops
CSP7040

Timings

- Slot: S
  - Fri: 6-7:30* PM
  - Sat: 5-6* PM
- Attendance will be taken using google meet.

Prerequisites

- Knowledge of CS principles and skills
- Understanding of ML algorithms (from “Machine Learning” course)
- Familiarity with at least one ML framework
  - We will be using Python/PyTorch.
- Familiarity with basic maths

Use of Google Classroom

- Use google classroom for all discussions related to the course.

- Post general questions as public comments and specific questions as private comments under appropriate heading.

Maintain

- Discipline
- Academic integrity
- Class decorum

Academic Code of Honour

- Cite all the resources you refer.
  - E.g. if you read it in a paper or webpage, cite it.
- NOT OK to ask someone to do assignments/projects for you.
- OK to discuss questions with classmates. Disclose your discussion partners.
- NOT OK to copy solutions from classmates.
- OK to use existing solutions as part of your projects/assignments. Clarify your contributions.
- NOT OK to pretend that someone's solution is yours.
- NOT OK to post your assignment solutions online.
- **ASK in google classroom if unsure!**

Marks distribution

- Assignments (Code + Report) : 3 or 4

- Each assignment will be of nearly the same weightage (within 20%)

- Class Performance: 5, with -ve marking

Any Comments?

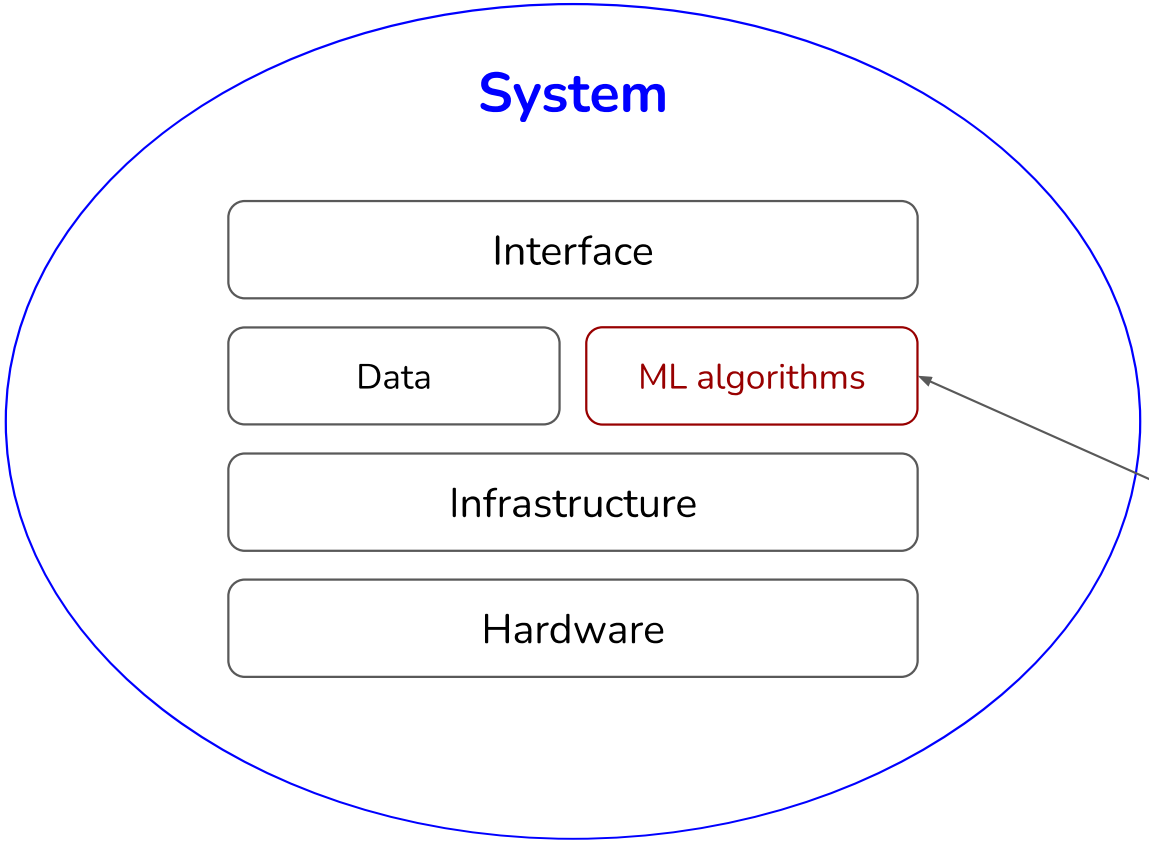
Agenda

1. ML research vs. ML production
2. ML systems vs. traditional software
3. ML production myths

Adapted from the slides shared by Dr. Chip Huyen of Stanford University for the course CS 329S at https://stanford-cs329s.github.io/

Why this course?

- ML algorithms is the less problematic part.
- The hard part is to **how to make algorithms work with other parts to solve real-world problems**.

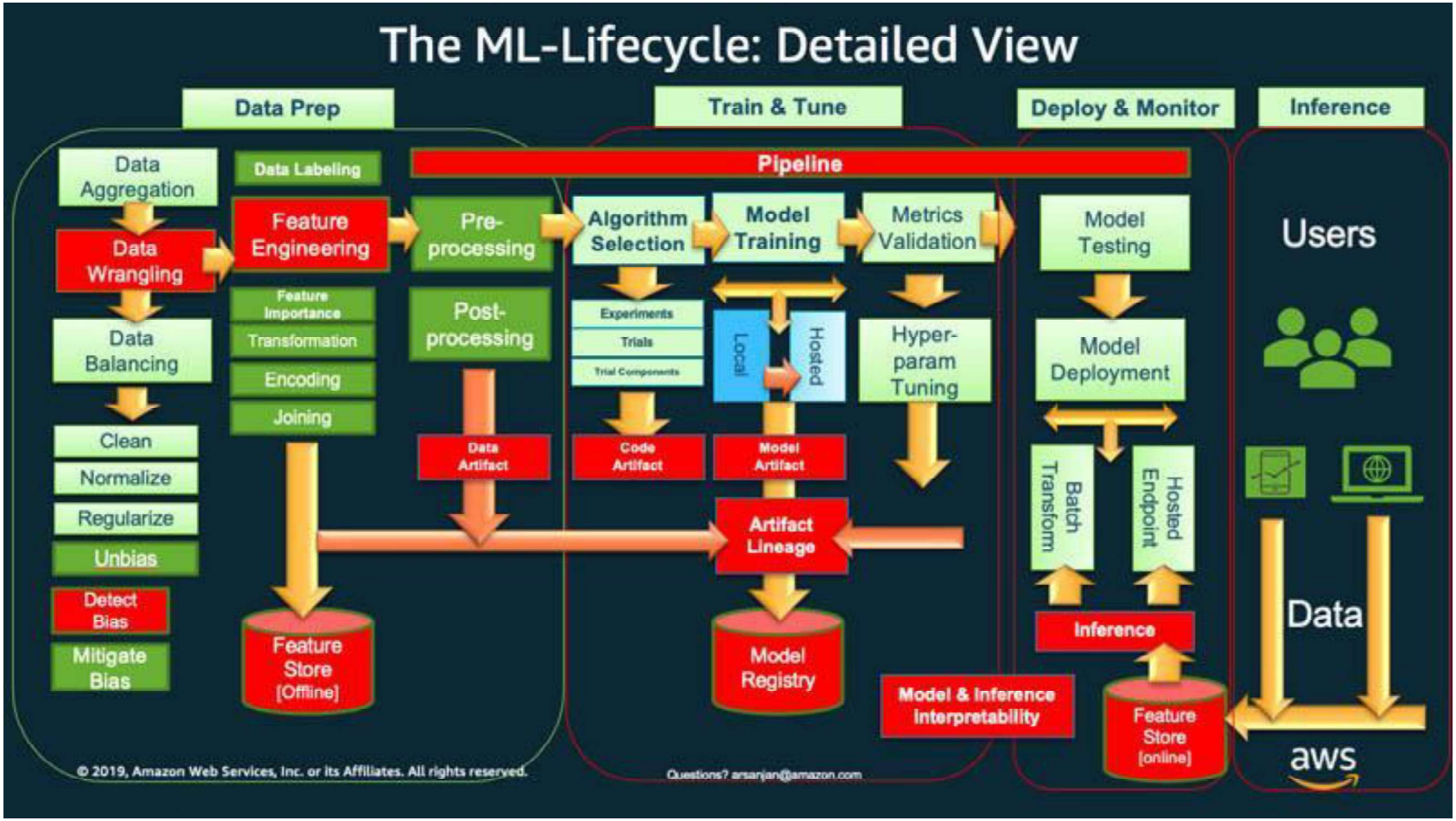

System
Interface
Data
ML algorithms
Infrastructure
Hardware
Most ML courses/books

What is ML Ops

“An ML culture and practice that unifies ML application development (Dev) with ML system deployment and operations (Ops).”

Source: https://aws.amazon.com/what-is/mlops/

Pipeline



Source: https://aws.amazon.com/what-is/mlops/

What's machine learning systems design?

The process of defining the **interface, algorithms, data**, **infrastructure**, and **hardware** for a machine learning system to satisfy **specified requirements**.

What's machine learning systems design?

The process of defining the **interface, algorithms, data**, **infrastructure**, and **hardware** for a machine learning system to satisfy **specified requirements**.

reliable, scalable, maintainable, adaptable

The questions to think about …

- You've trained a model, now what?
- What are different components of an ML system?
- How to do data engineering?
- How to evaluate your models, both offline and online?
- What's the difference between online prediction and batch prediction?
- How to serve a model on the cloud? On the edge?
- How to continually monitor and deploy changes to ML systems?
- …

ML research vs. ML production

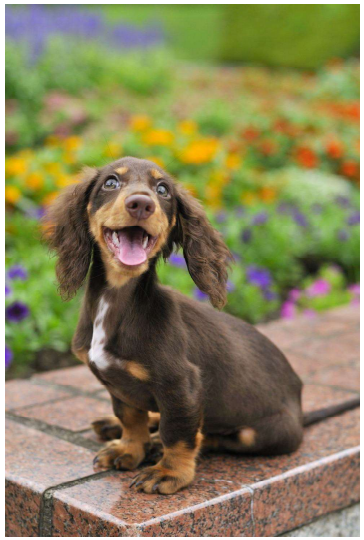
ML research vs. ML production

	Research	**Production**
Objectives	Model performance*	Different stakeholders have different objectives

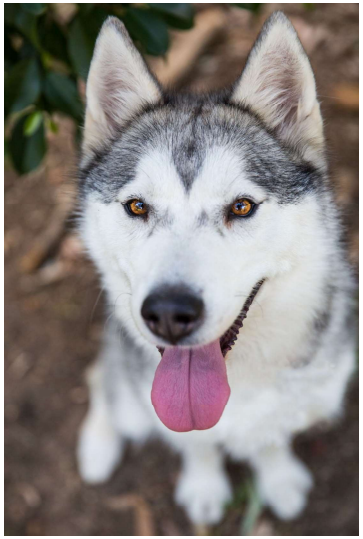
"*" It's actively being worked. See Utility is in the Eye of the User: A Critique of NLP Leaderboards (Ethayarajh and Jurafsky, EMNLP 2020)

Stakeholder objectives

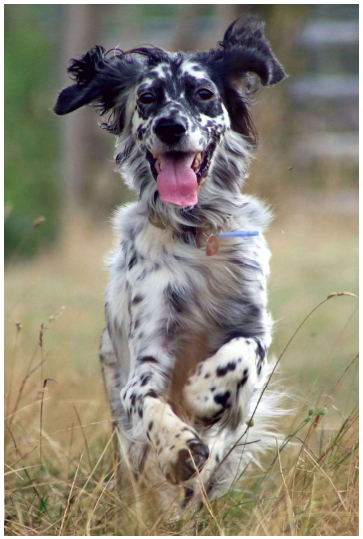
ML team
highest accuracy



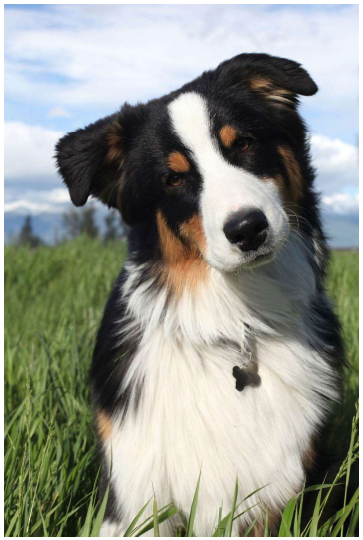
Sales
sells more ads



Product
fastest inference



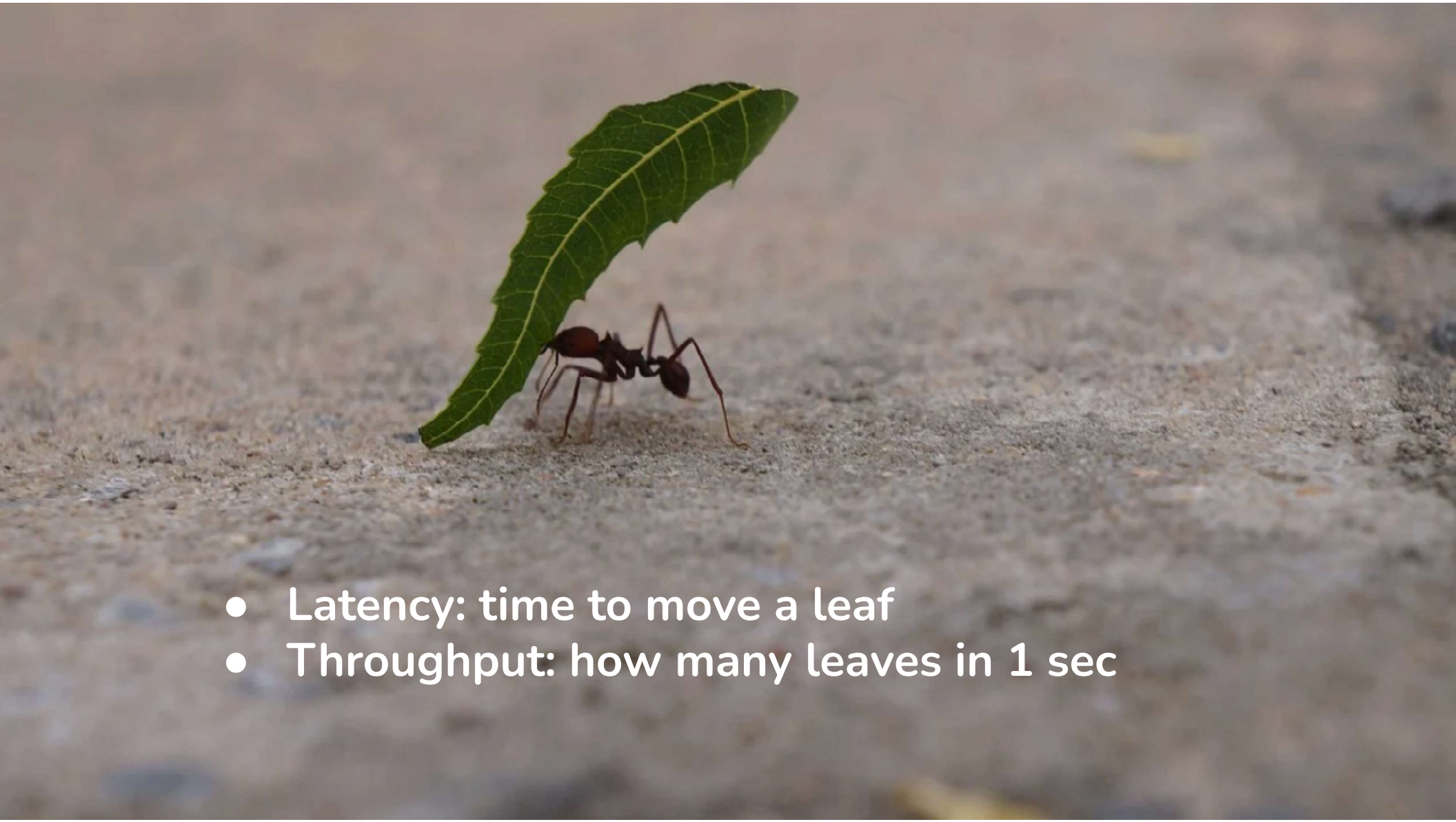
Manager
maximizes profit



Computational priority

	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency

generating predictions


• Latency: time to move a leaf
• Throughput: how many leaves in 1 sec

ML in research vs. in production

	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting

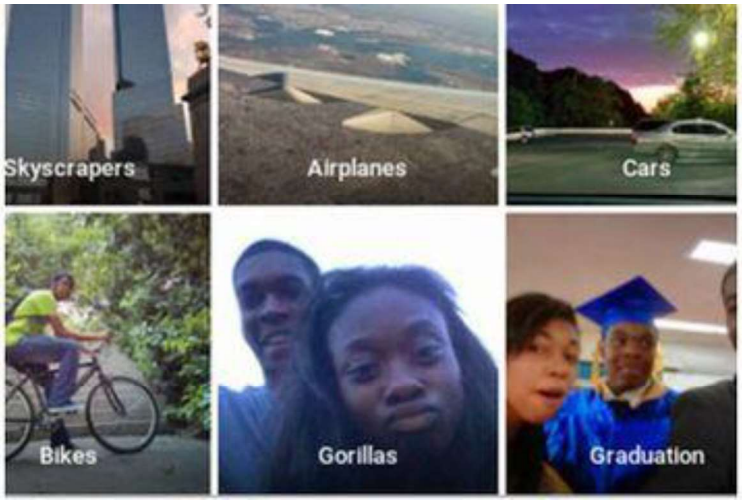
Data

Research	Production
• Clean • Static • Mostly historical data	• Messy • Constantly shifting • Historical + streaming data • Biased, and you don't know how biased • Privacy + regulatory concerns

ML in research vs. in production

	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting
Fairness	Good to have	Important

Fairness

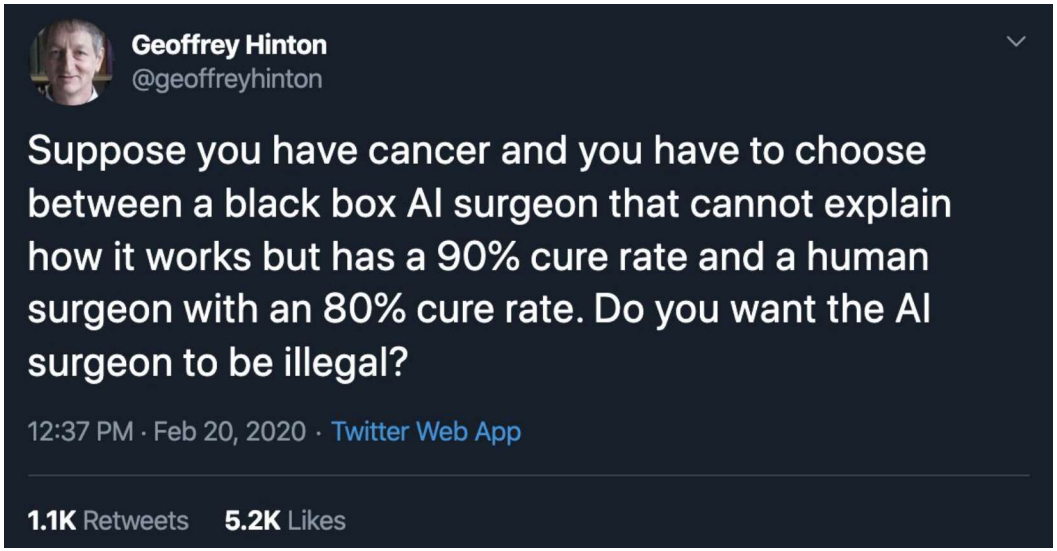


The Berkeley study found that both face-to-face and online lenders rejected a total of 1.3 million creditworthy black and Latino applicants between 2008 and 2015. Researchers said they believe the applicants "would have been accepted had the applicant not been in these minority groups." That's because when they used the income and credit scores of the rejected applications but deleted the race identifiers, the mortgage application was accepted.

ML in research vs. in production

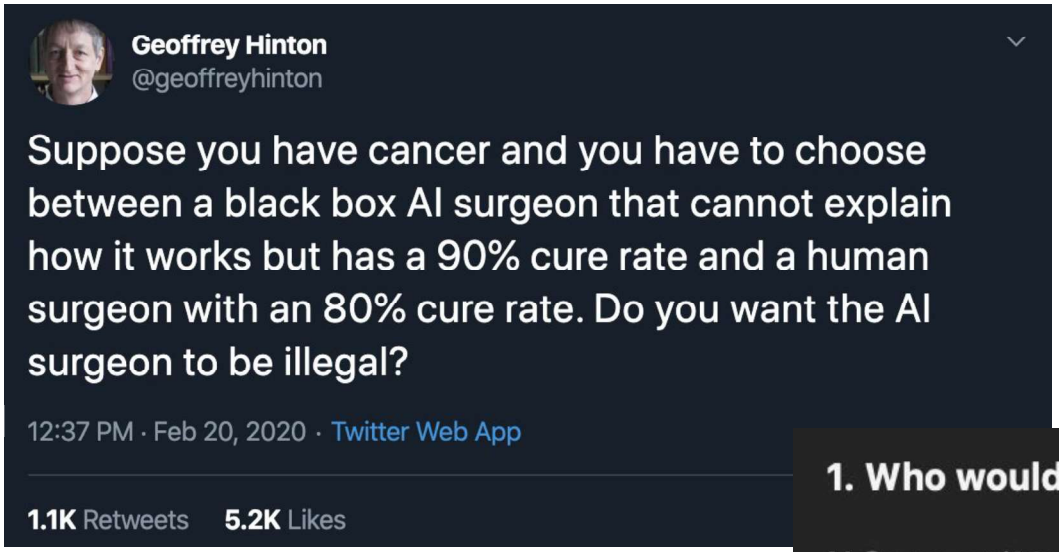
	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting
Fairness	Good to have	Important
Interpretability	Good to have	Important

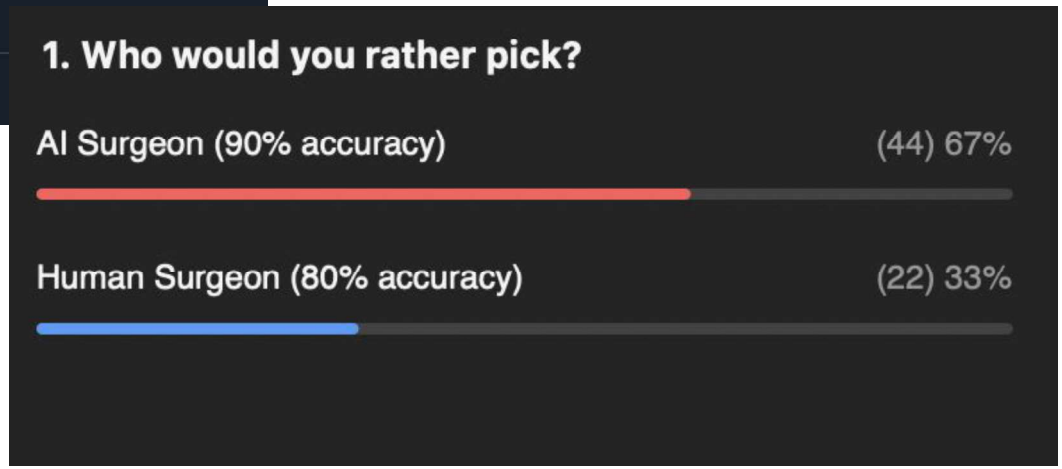
Interpretability



Zoom poll: which one would you want as your surgeon?

Interpretability





Result from the Zoom poll last year

ML in research vs. in production

	Research	**Production**
Objectives	Model performance	Different stakeholders have different objectives
Computational priority	Fast training, high throughput	Fast inference, low latency
Data	Static	Constantly shifting
Fairness	Good to have	Important
Interpretability	Good to have	Important

ML systems vs. traditional software

Traditional software

Separation of Concerns is a design principle for separating a computer program into distinct sections such that each section addresses a separate concern

- Code and data are separate
  - Inputs into the system shouldn't change the underlying code

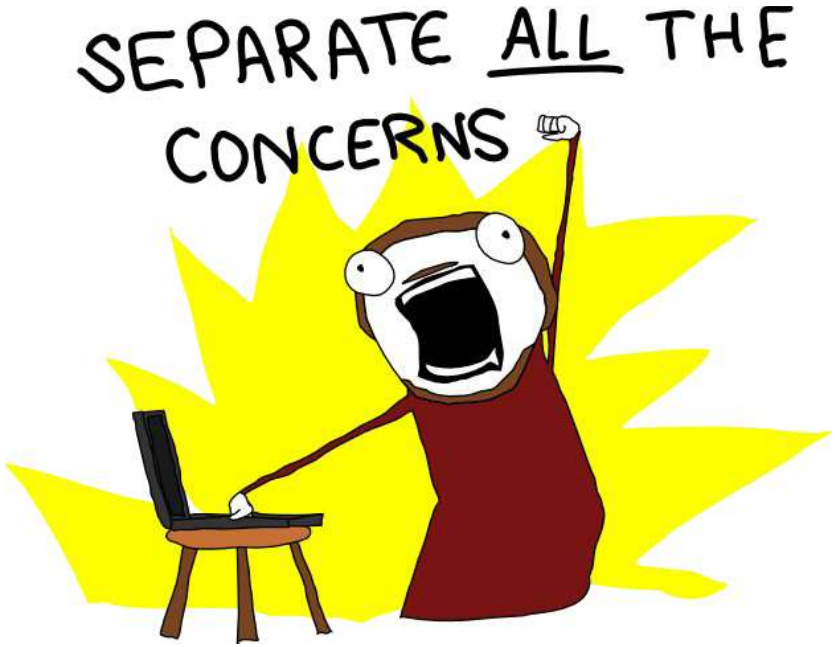


Image by Arda Cetinkaya

ML systems

- Code and data are tightly coupled
  - ML systems are part code, part data
- Not only test and version code, need to test and version data too

the hard part

How to ...

- Validate data correctness?
- Test features' usefulness?
- Detect when the underlying data distribution has changed?
- Know if the changes are bad for models without ground truth labels?
- Detect malicious data?
  - Not all data points are equal (e.g. scans of cancerous lungs are more valuable)
  - Bad data might harm your model and/or make it susceptible to attacks

Engineering challenges with large ML models

- Too big to fit on-device
- Consume too much energy to work on-device
- Too slow to be useful
  - Autocompletion is useless if it takes longer to make a prediction than to type

ML production myths

Myth #1: Deploying is hard

Myth #1: Deploying is hard

Deploying is easy. Deploying reliably is hard

Myth #2: You only deploy one or two ML models at a time

Myth #2: You only deploy one or two ML models at a time

Booking.com: 150+ models, Uber: thousands

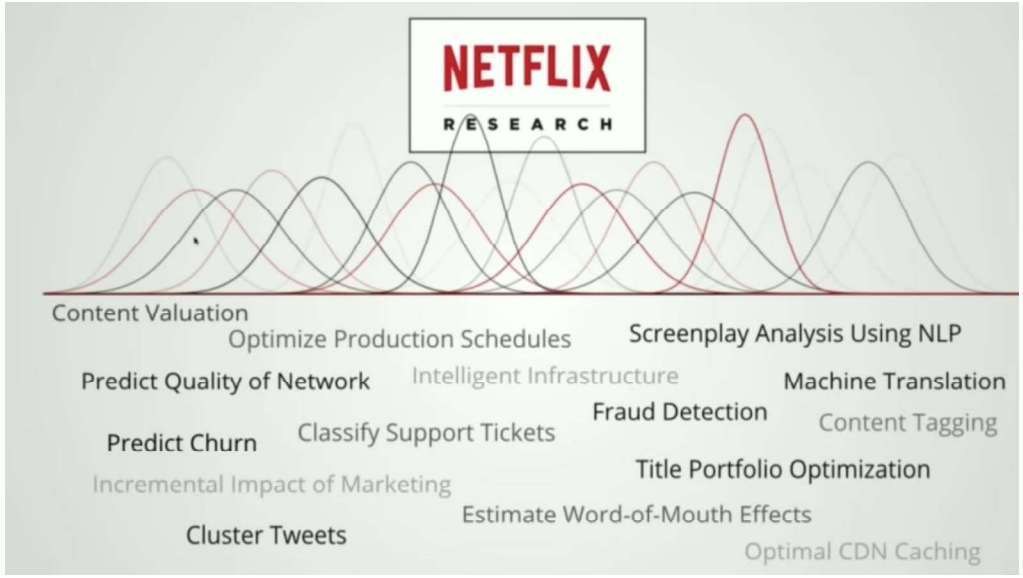


Image from Ville Tuulos (Netflix, Outerbounds)

Myth #3: You won’t need to update your models as much

DevOps: Pace of software delivery is accelerating

- Elite performers deploy **973x** more frequently with **6570x** faster lead time to deploy (Google DevOps Report, 2021)
- DevOps standard (2015)
  - Etsy deployed 50 times/day
  - Netflix 1000s times/day
  - AWS every 11.7 seconds

Myth #4: ML can magically transform your business overnight

Myth #4: ML can magically transform your business overnight

Magically: possible
Overnight: no

ML engineering is more engineering than ML

MLEs might spend most of their time:

- wrangling data
- understanding data
- setting up infrastructure
- deploying models

instead of training ML models

Myth #5: Most ML engineers don’t need to worry about scale