CSCE 585: Machine Learning Systems

ML Systems in Production

Pooyan Jamshidi

Fall 2023 | Course Website: https://pooyanjamshidi.github.io/mls/

UNIVERSITY OF South Carolina

ML in research vs. production

This part of lecture is mainly adopted from CS 329S: Machine Learning Systems Design at Stanford

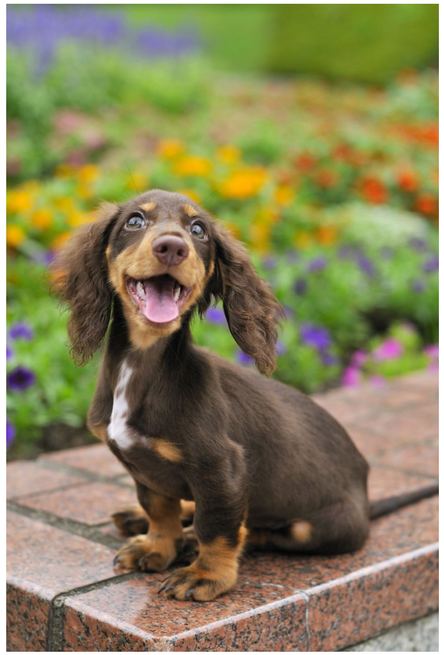
ML in research vs. in production

	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)

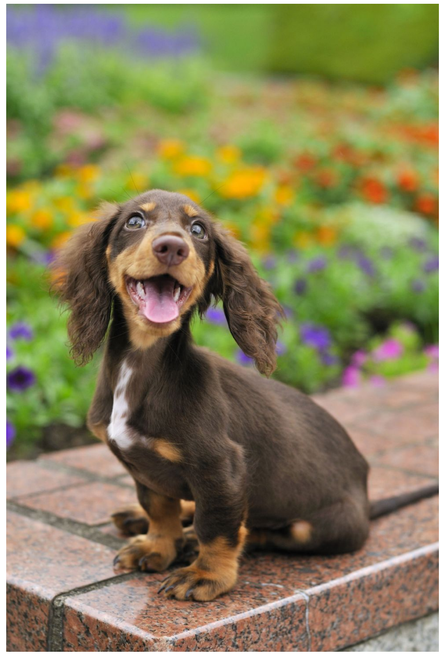
Stakeholder objectives

ML team
highest accuracy



Stakeholder objectives

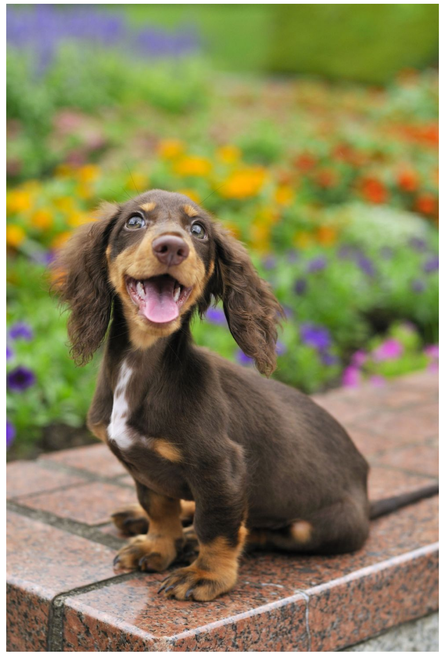
ML team
highest accuracy



Sales
sells more ads

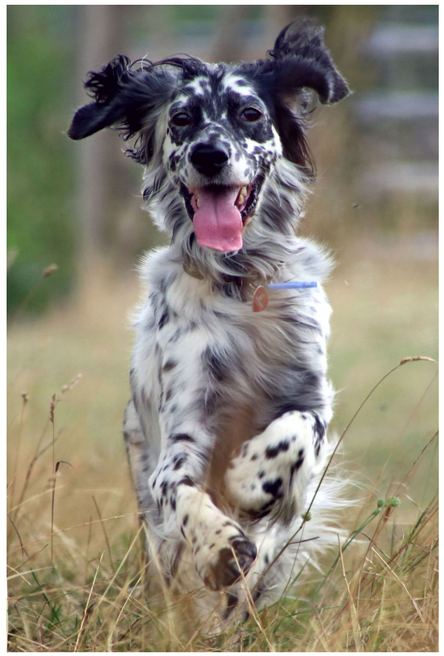
Stakeholder objectives

ML team
highest accuracy



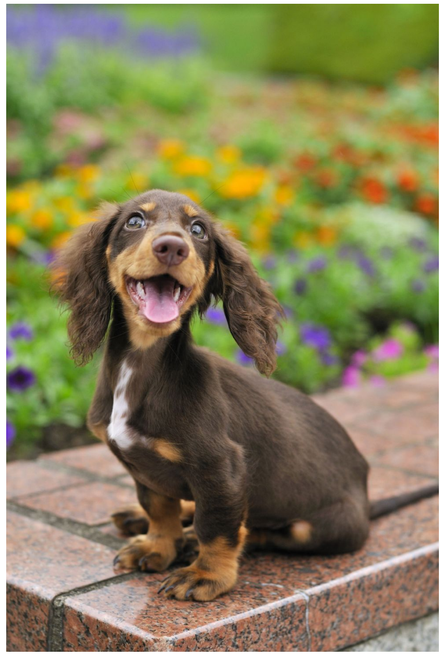
Sales
sells more ads

Product
fastest inference



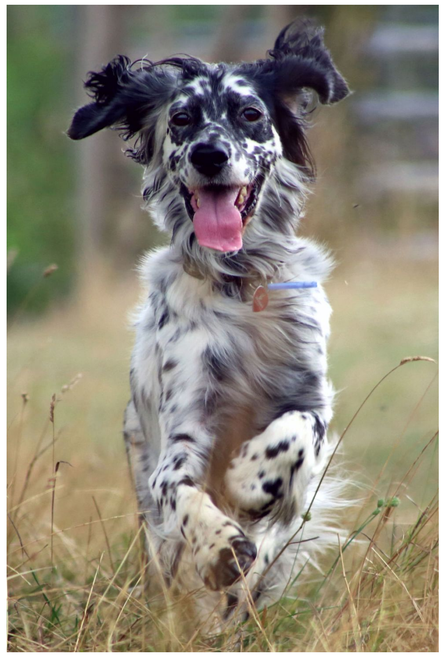
Stakeholder objectives

ML team
highest accuracy

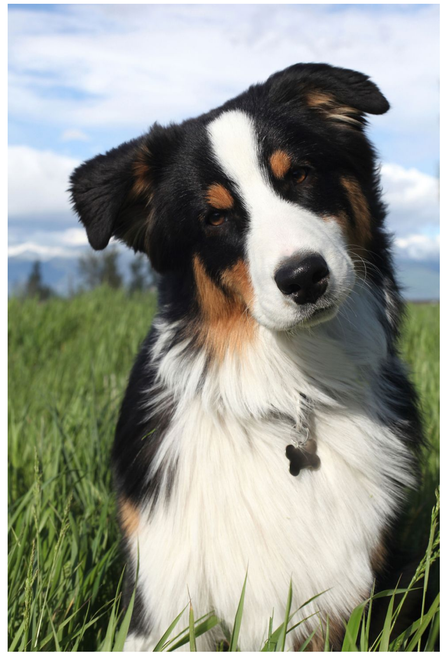


Sales
sells more ads

Product
fastest inference

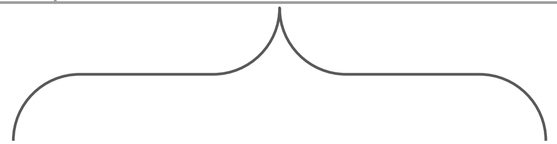


Manager
maximizes profit
= laying off ML teams



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 matters

Latency 100 -> 400 ms reduces searches 0.2% - 0.6% (2009)

30% increase in latency costs 0.5% conversion rate (2019)

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

- Real-time: low latency = high throughput
- Batched: high latency, high throughput

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

THE COGNITIVE CODER

By Armand Ruiz, Contributor, InfoWorld | SEP 26, 2017 7:22 AM PDT

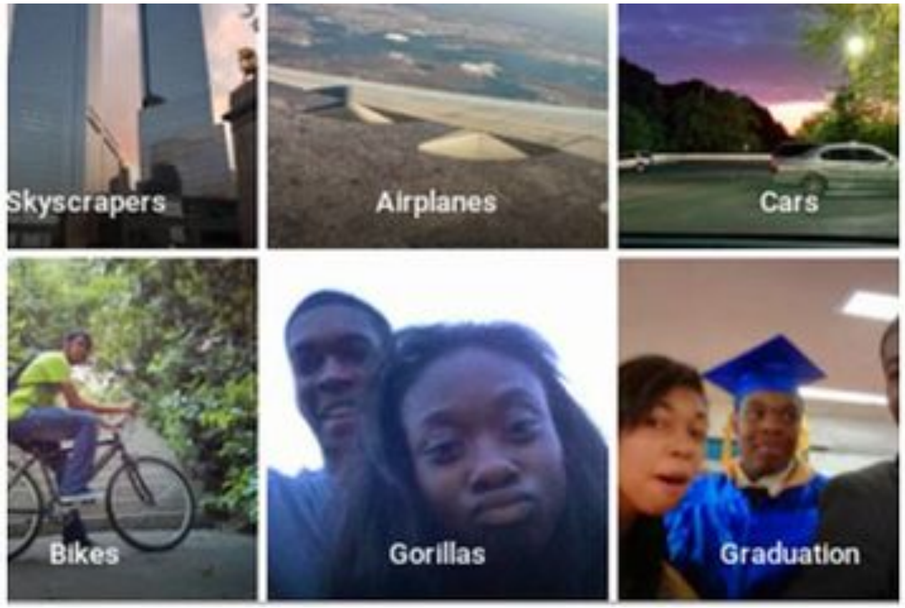
The 80/20 data science dilemma

Most data scientists spend only 20 percent of their time on actual data analysis and 80 percent of their time finding, cleaning, and reorganizing huge amounts of data, which is an inefficient data strategy

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 (sadly)	Important

Fairness

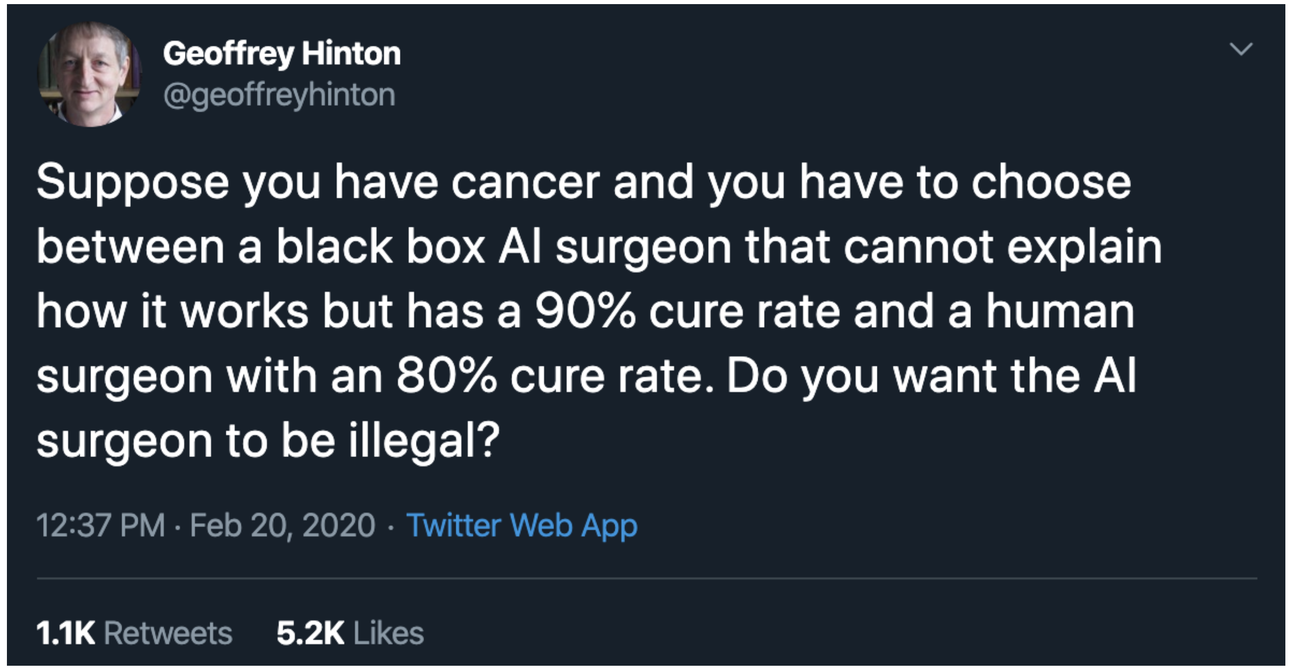


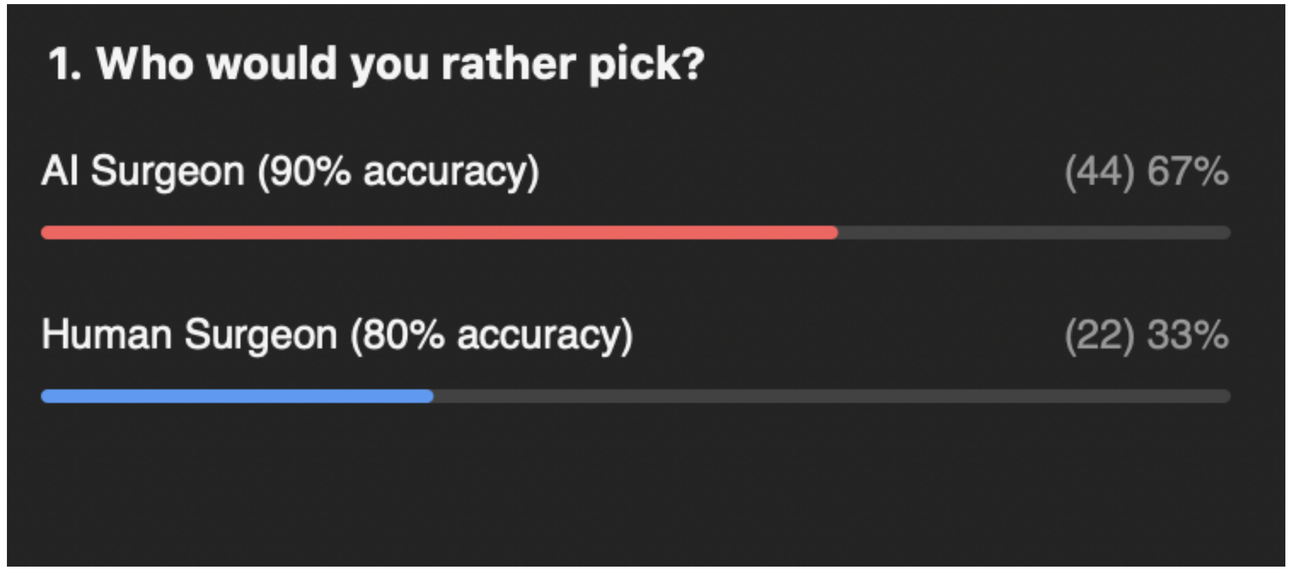
The Berkeley study found that both face-to-face and online lenders rejected a total of 1.3 million creditworthy black and Latino applicants between 2008 and 2015. Researchers said they believe the applicants "would have been accepted had the applicant not been in these minority groups." That's because 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 (sadly)	Important
Interpretability*	Good to have	Important

Interpretability





Result from the Zoom poll

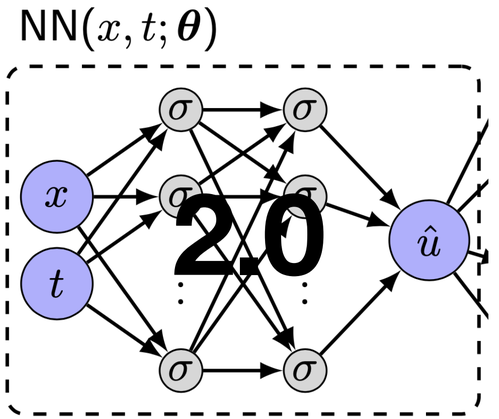
ML in research vs. in production

	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 (sadly)	Important
Interpretability	Good to have	Important

ML systems vs. traditional software

Software 1.0 vs Software 2.0

Software 1.0 vs Software 2.0

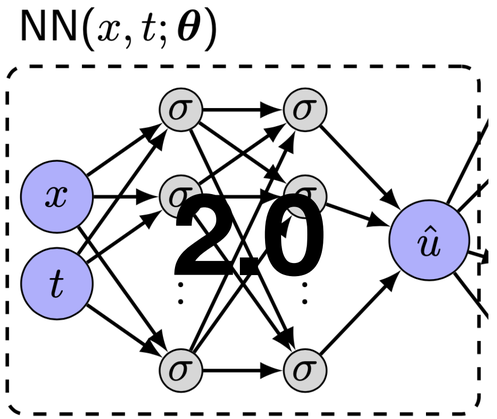


- Written in code (C++, ...)
- Requires domain expertise
    1. Decompose the problem
    2. Design algorithms
    3. Compose into a system

- Written in terms of a neural network model with
    - A model architecture
    - Weights that are determined using optimization

https://medium.com/@karpathy/software-2-0-a64152b37c35

Software 1.0 vs Software 2.0



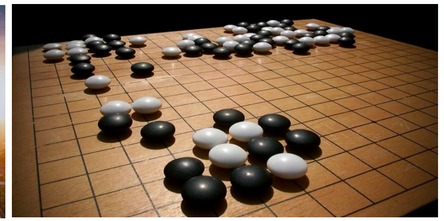
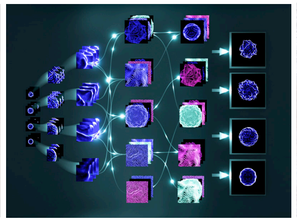
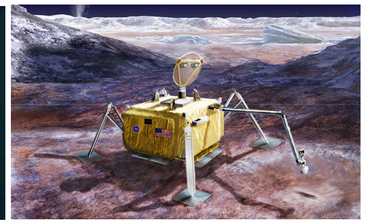
- **Input**: Algorithms in code
- **Compiled to**: Machine instructions

- **Input**: Training data
- **Compiled to**: Learned parameters

https://medium.com/@karpathy/software-2-0-a64152b37c35

Software 1.0 vs Software 2.0




AI for Social Good

- **Easier to build and deploy**
  - Build products faster
  - Predictable runtimes and memory use: easier qualification
- A **wide range of applications** from self-driving cars, to game, healthcare, robotics, space, and social good.
- **1000x Productivity**: Google shrinks language translation code from 500k LoC to 500

https://jack-clark.net/2017/10/09/import-ai-63-google-shrinks-language-translation-code-from-500000-to-500-lines-with-ai-only-25-of-surveyed-people-believe-automationbetter-jobs/

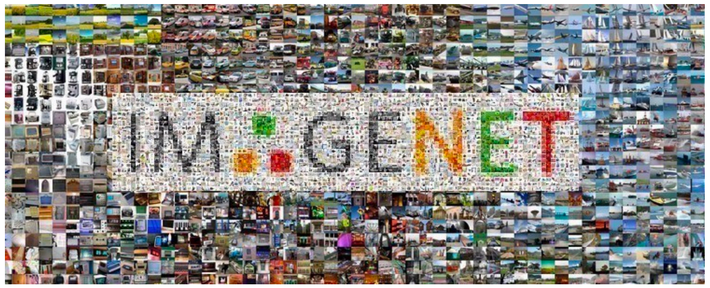
https://ai.google/social-good/

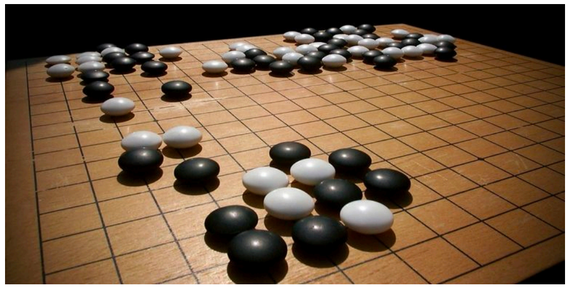
What is going on in this mad era of AI/ML!

It’s incredible, isn’t it?

Incredible advances in:

1. Image Recognition (ImageNet + Deep Learning)
2. Reinforcement Learning (DeepMind AlphaGo Zero)
3. Natural Language Processing (GPT-3)





Traditional software

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

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

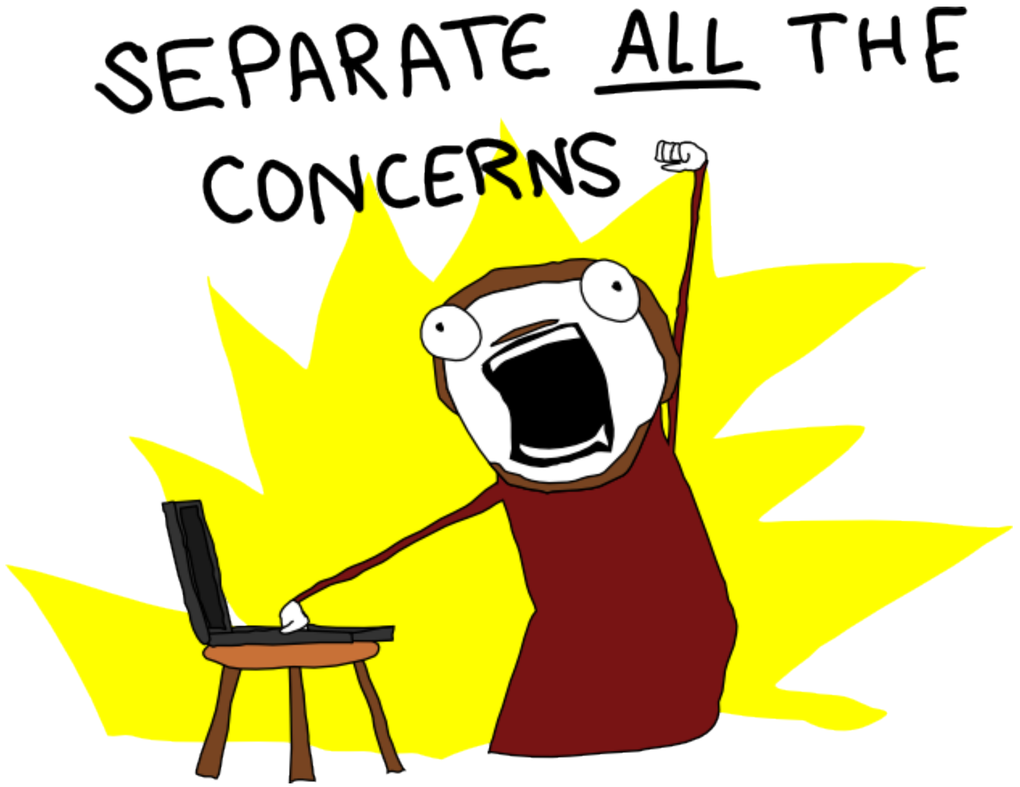


Image by Arda Cetinkaya

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

ML System: version data

- Line-by-line diffs like Git doesn’t work with datasets
- Can’t naively create multiple copies of large datasets
- How to merge changes?

ML System: test data

- How to test data correctness/usefulness?
- How to know if data meets model assumptions?
- How to know when the underlying data distribution has changed? How to measure the changes?
- How to know if a data sample is good or bad for your systems?
  - Not all data points are equal (e.g. images of road surfaces with cyclists are more important for autonomous vehicles)
  - Bad data might harm your model and/or make it susceptible to attacks like data poisoning 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
- How to run CI/CD tests if a test takes hours/days?

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

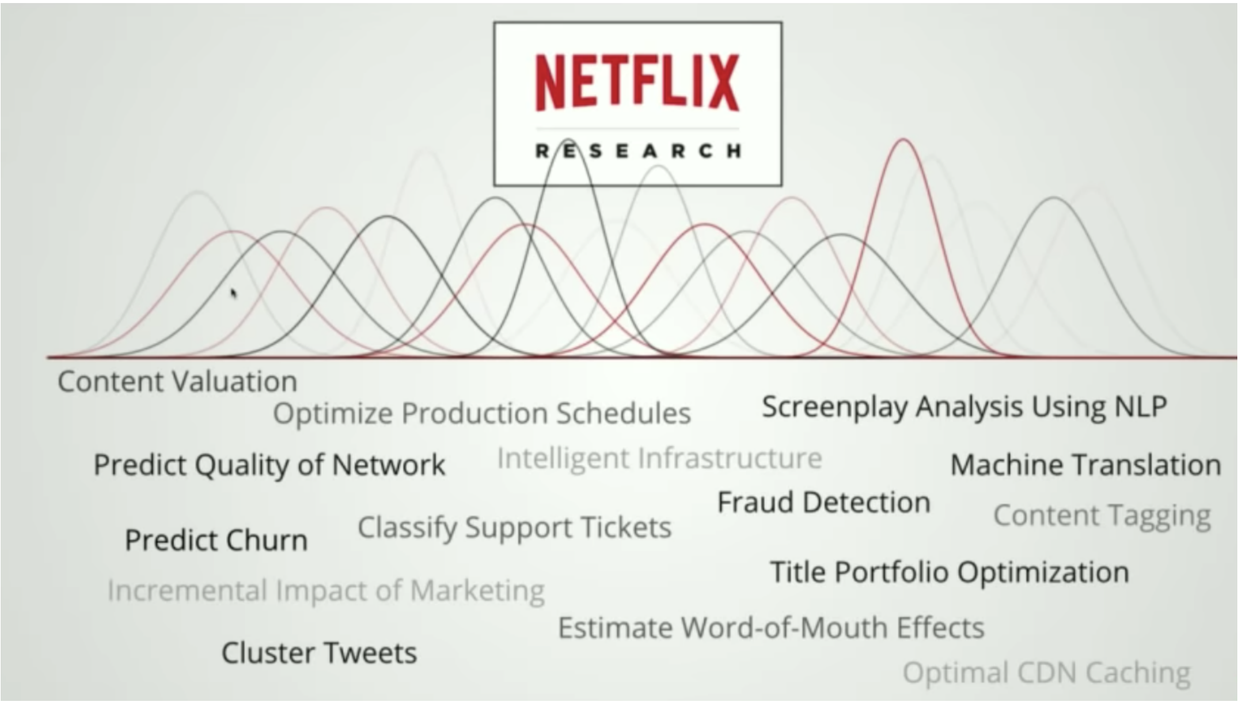


Image from Ville Tuulos (Netflix, Outerbounds)

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

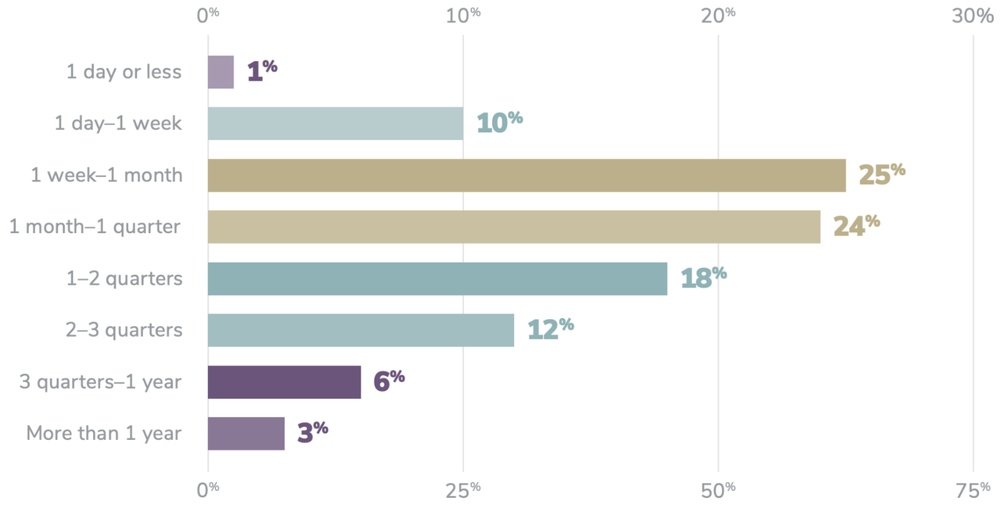
DevOps: Pace of software delivery is accelerating

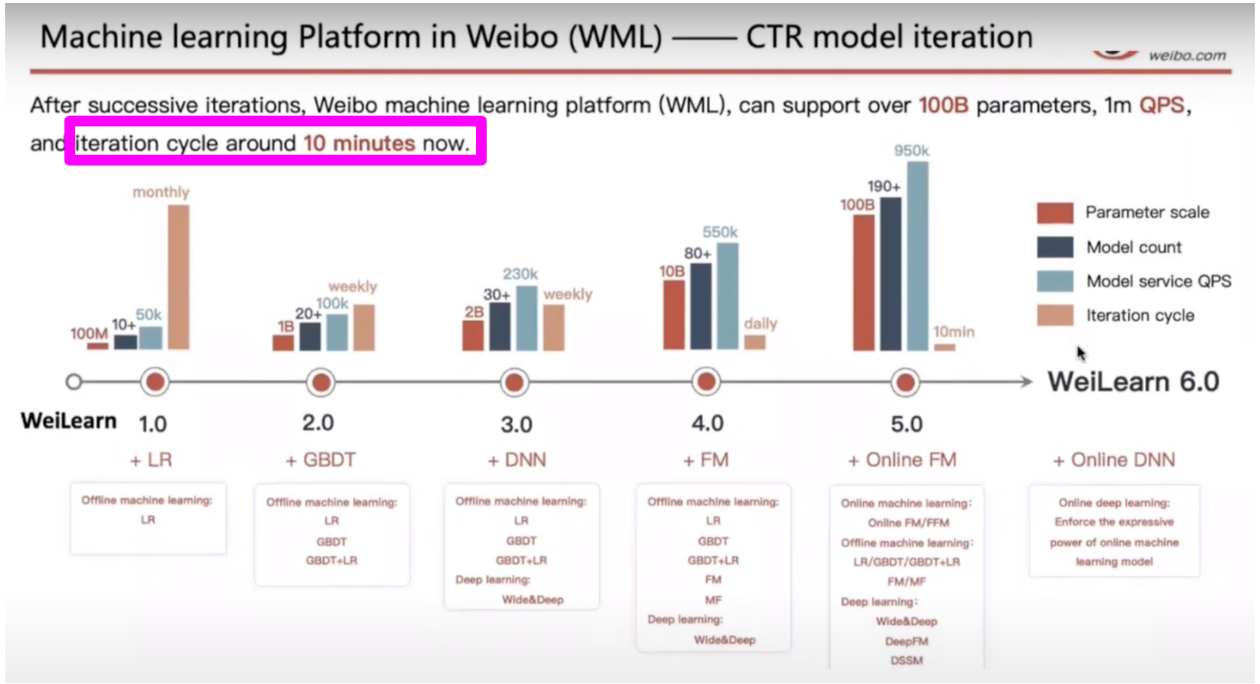
- Elite performers deploy **973x** more frequently with **6570x** faster lead time to deploy (Google DevOps Report, 2021)
- DevOps standard (2015)
  - Etsy deployed 50 times/day
  - Netflix 1000s times/day
  - AWS every 11.7 seconds

DevOps to MLOps: Slow vs. Fast

We'll learn how to do minute-iteration cycle!

Only 11% of organizations can put a model into production within a week, and 64% take a month or longer





Left image from Algorithmia | Right image: Machine learning with Flink in Weibo (Qian Yu, QCon 2019)

Accelerating ML Delivery

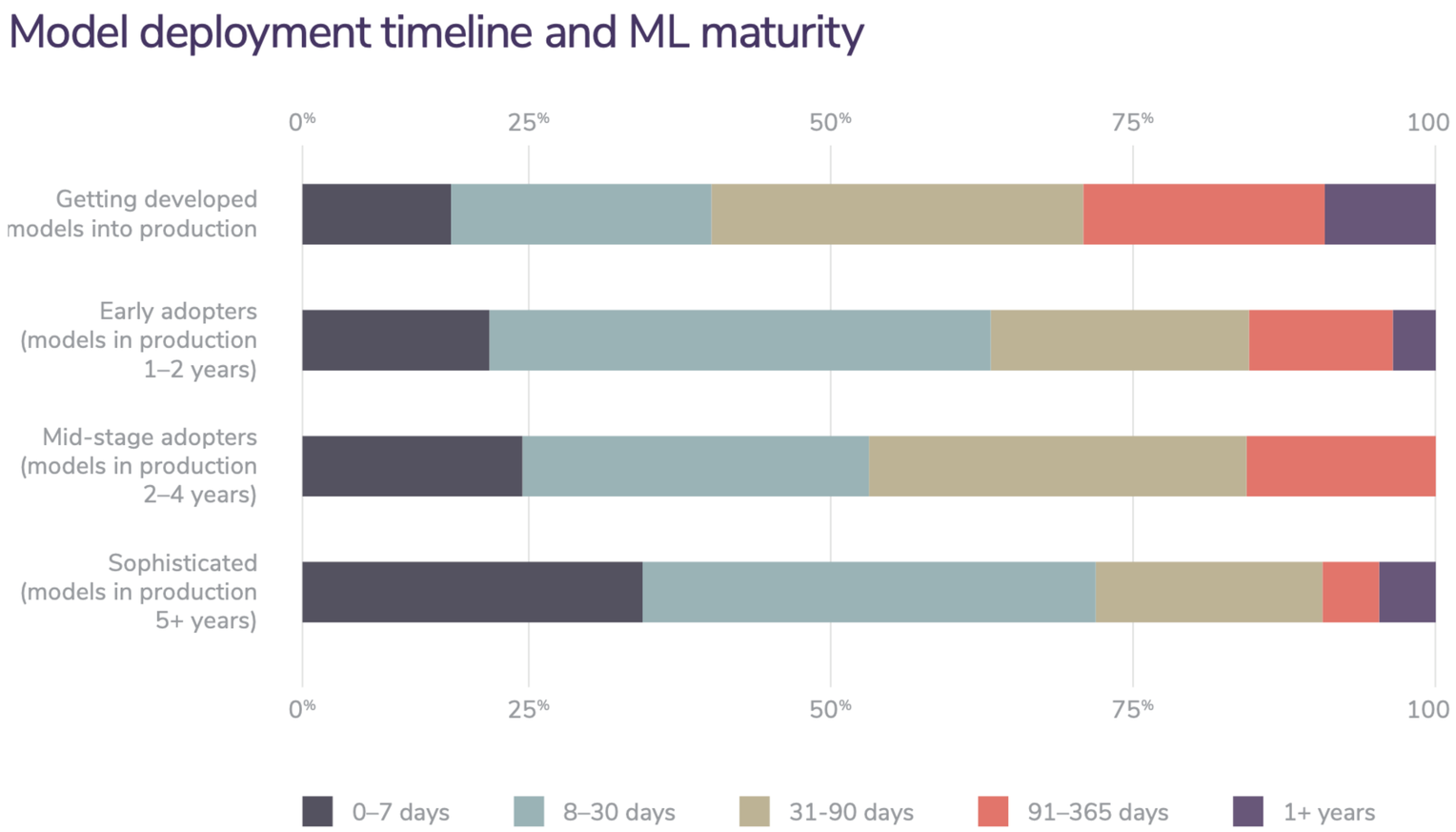
ML + DevOps = 

Myth #4: ML can magically transform your business overnight

Myth #4: ML can magically transform your business overnight

Magically: possible
Overnight: no

Efficiency improves with maturity



2020 state of enterprise machine learning (Algorithmia, 2020)

ML engineering is more engineering than ML

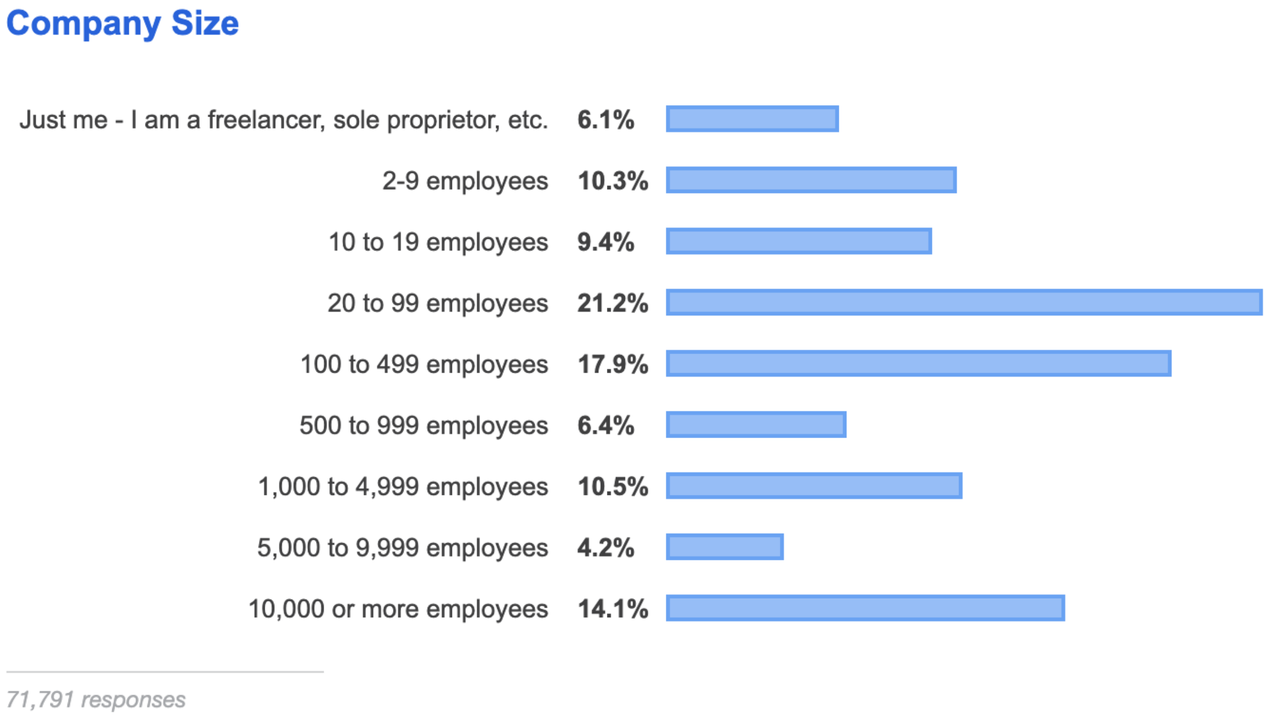
MLEs might spend most of their time:

- 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

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



[StackOverflow Developer Survey 2019]