# Software Engineering for Machine Learning (SE4ML)

17313 - Foundations of Software Engineering

# Nadia Nahar

Software Engineering Ph.D. Student, Carnegie Mellon University

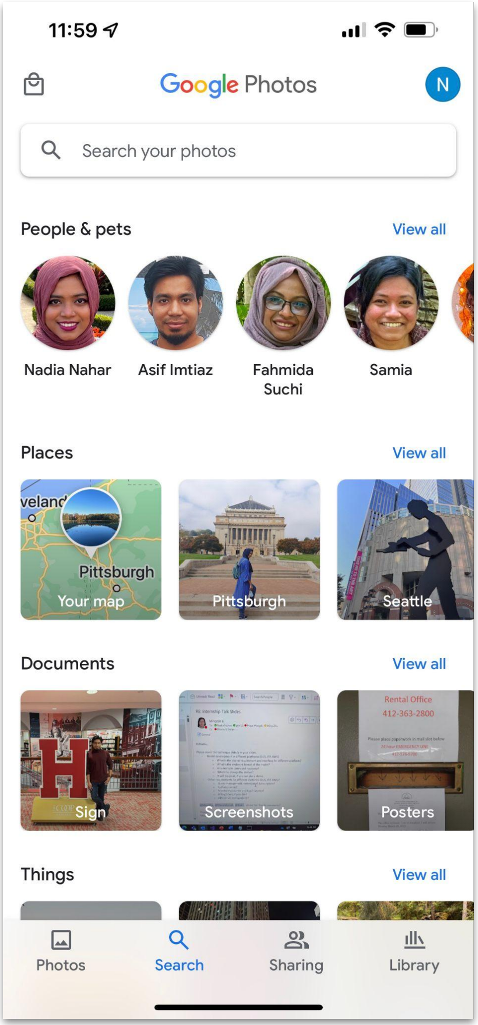
Research on Software Engineering for Machine Learning (SE4ML)

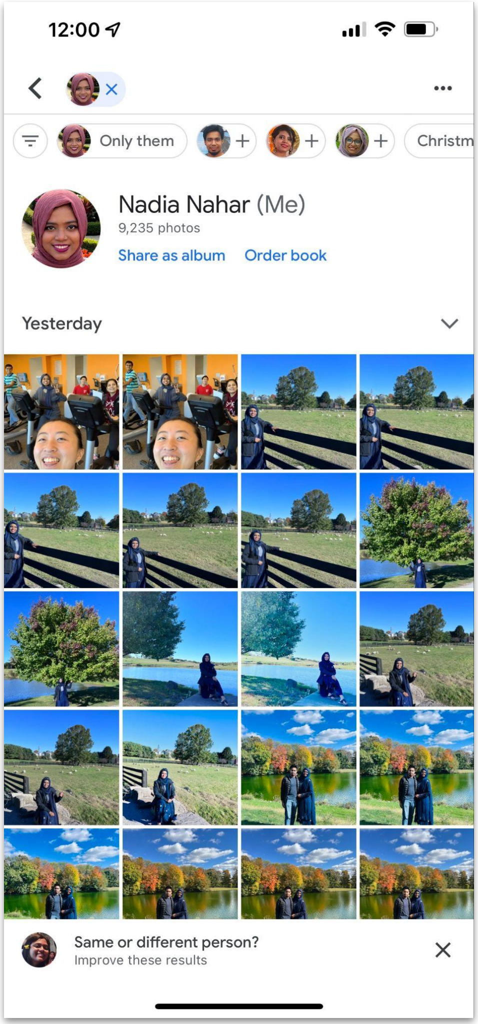
Worked on Deep Learning Inference Service (DLIS) at Microsoft

# Machine Learning in Software Products

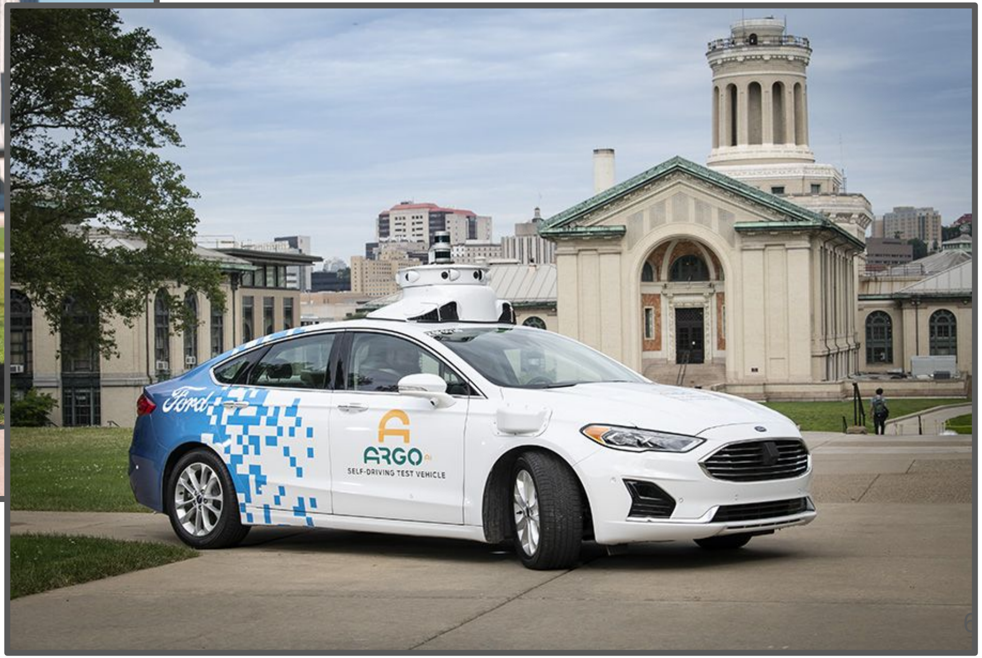
Can you think of a product you use,
that has one/more ML component(s)?

Google Photos



# Autonomous Car

# DALL-E

TEXT DESCRIPTION

An astronaut Teddy bears A bowl of soup

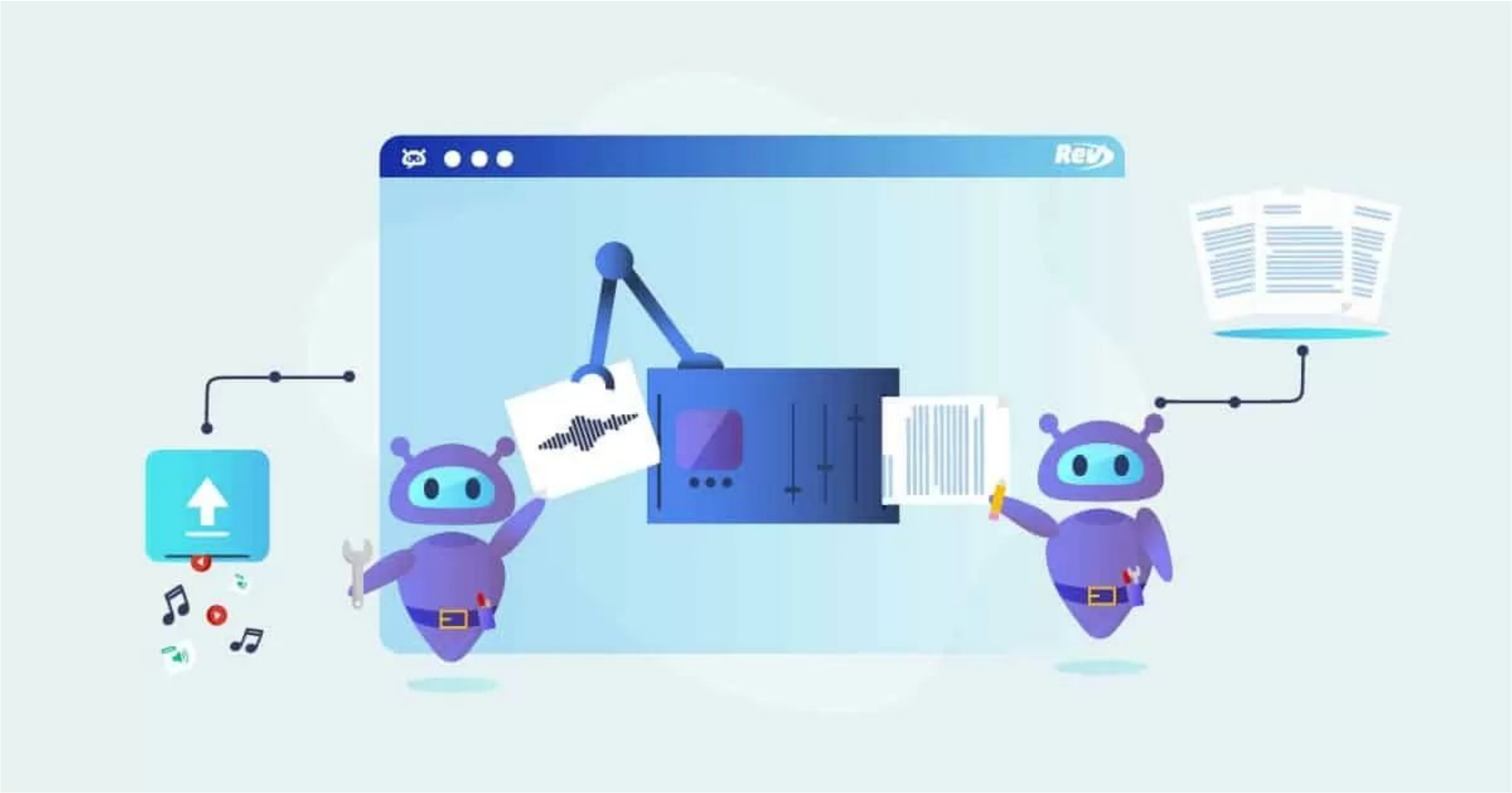
riding a horse lounging in a tropical resort in space playing basketball with cats in space

in a photorealistic style in the style of Andy Warhol as a pencil drawing

→

DALL·E 2

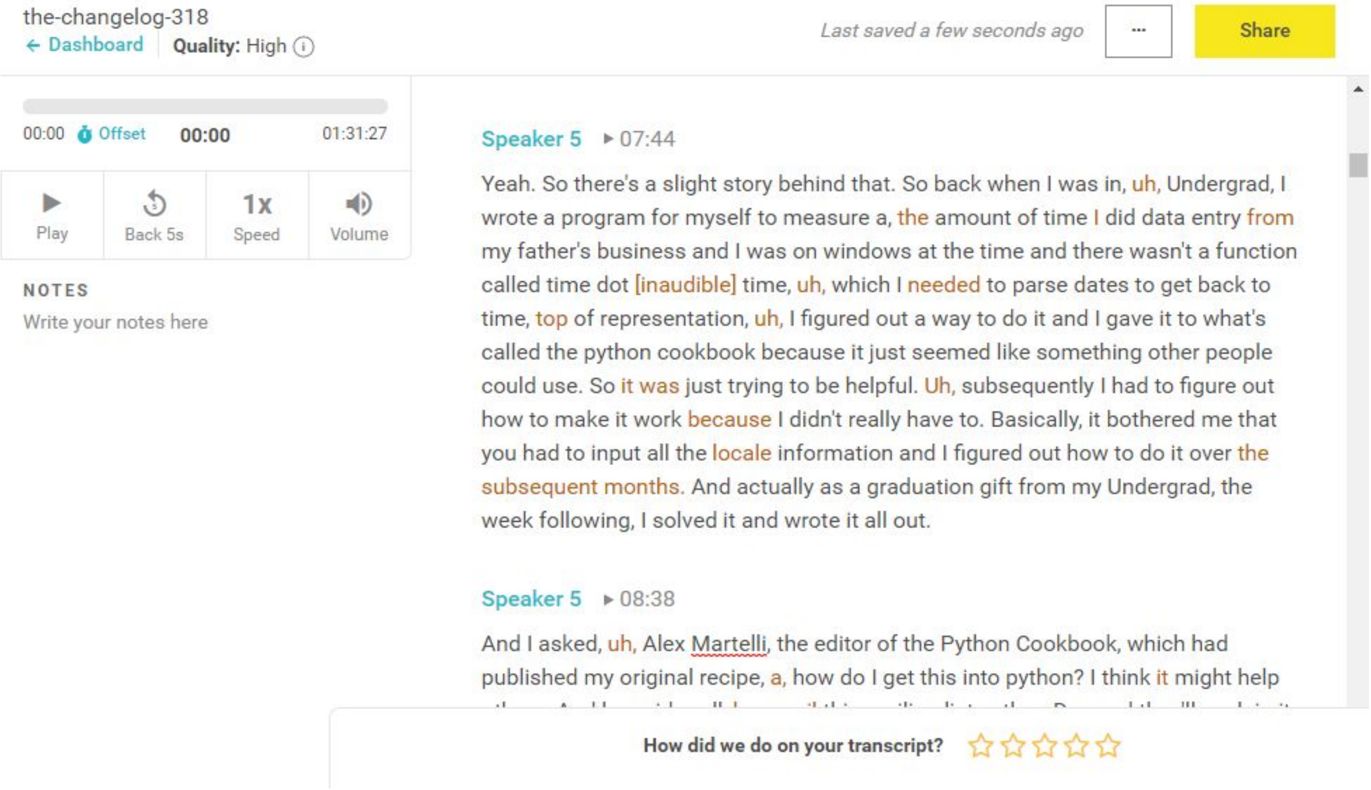
# Case Study: Transcription Service



# Participation Activity

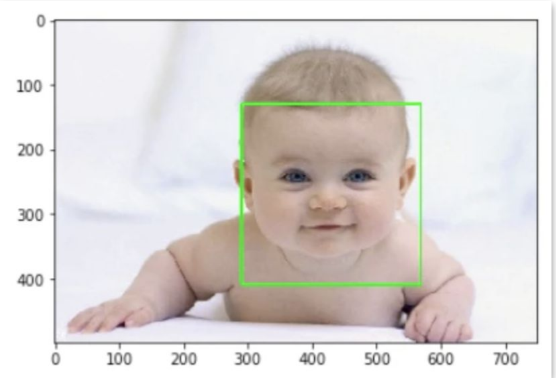
What functionalities do you need to provide, to sell a model for transcription as a product?

# Case Study: Transcription Service

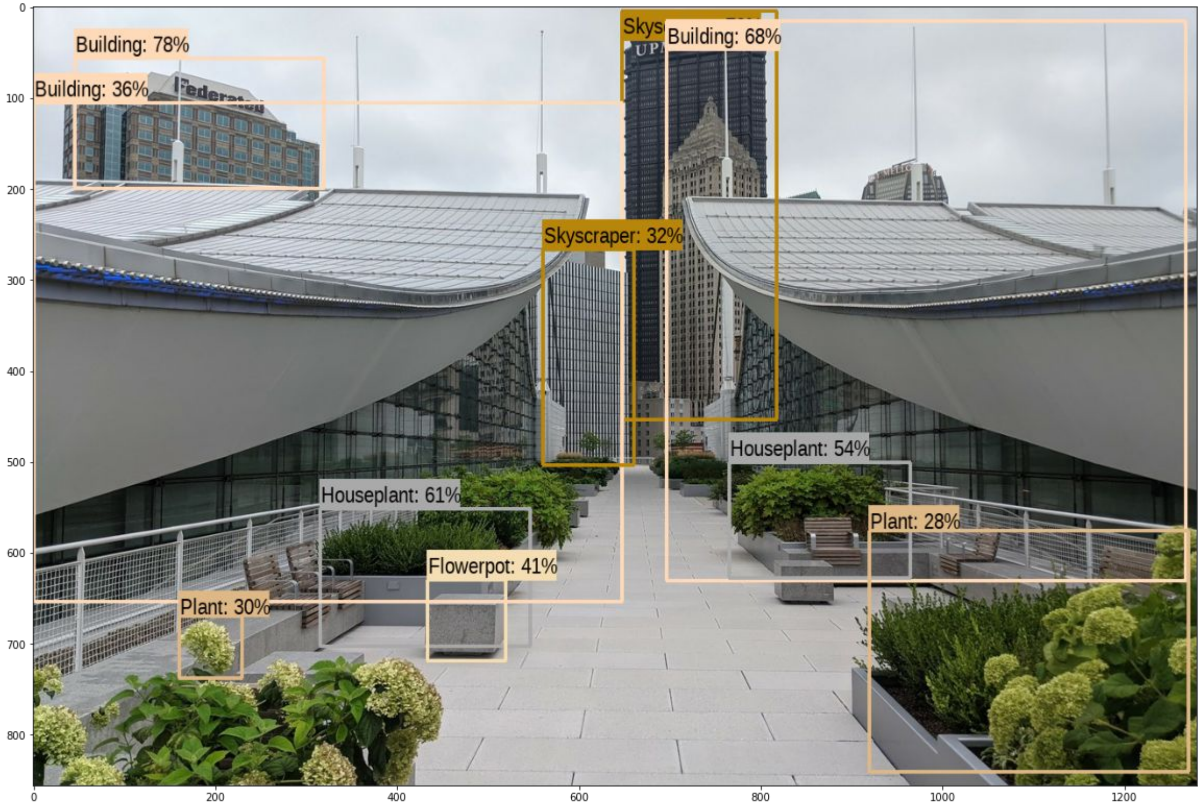


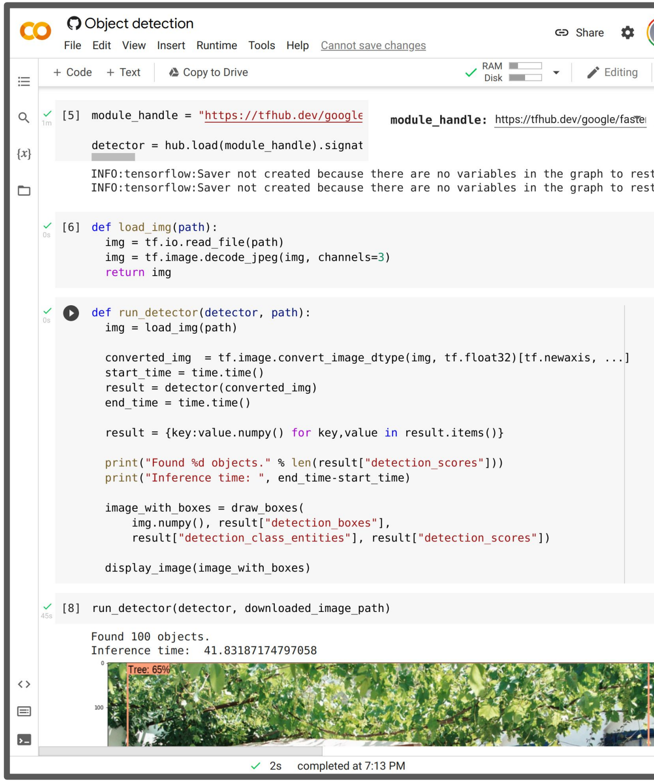
# From Models to Systems

# Data Science is Model Centric

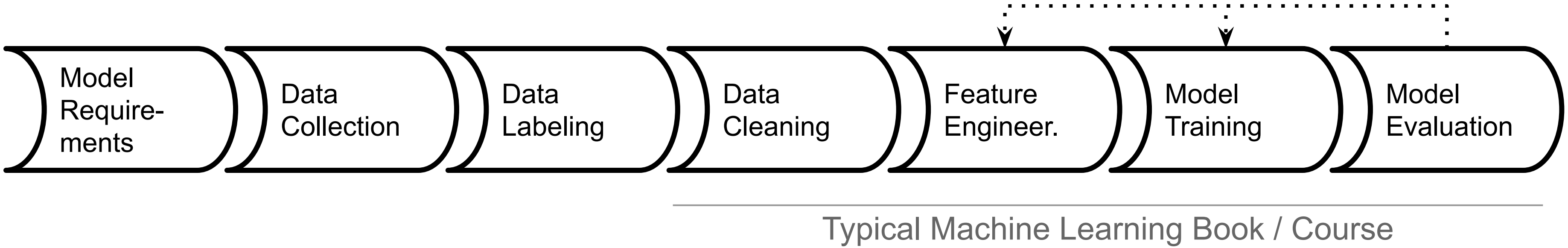



# Data Science is Model Centric



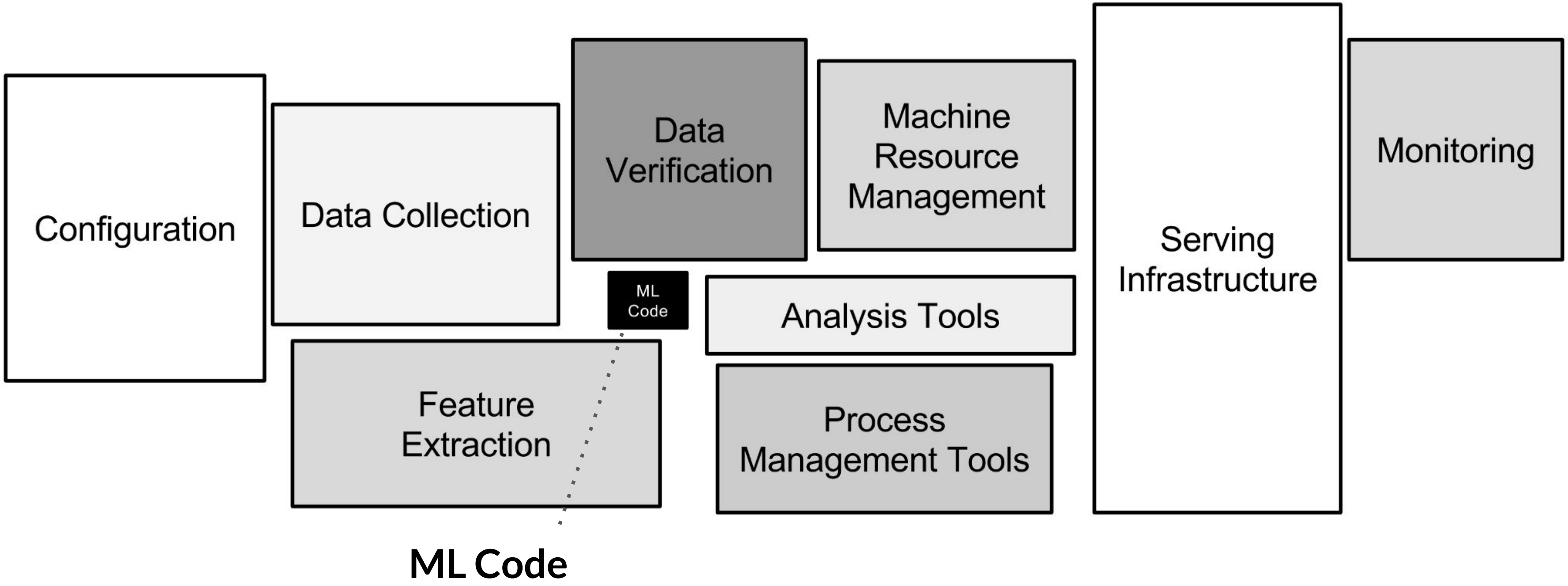



# Data Science Pipelines



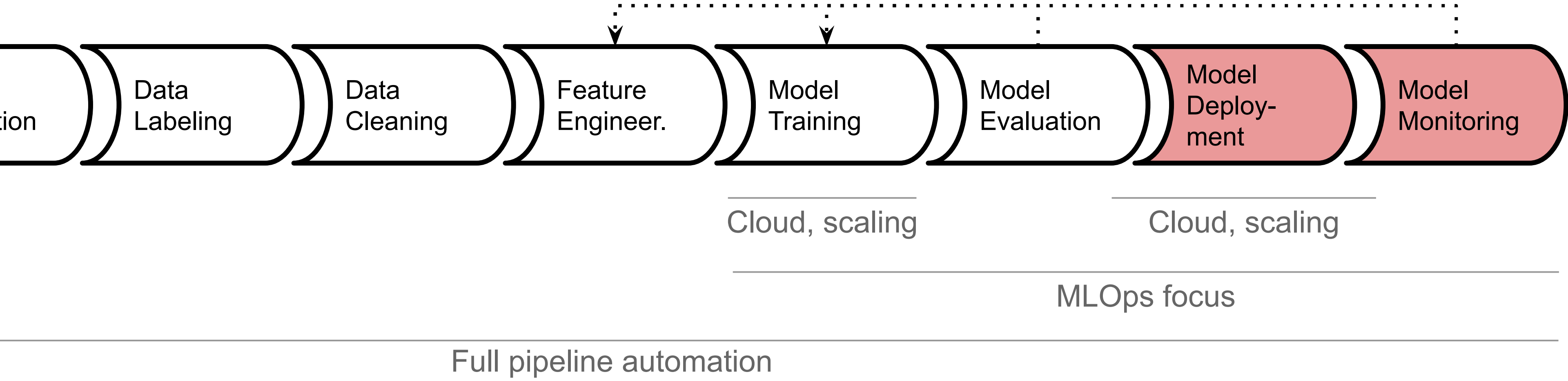

Focus: building models from given data, evaluating accuracy

# Model Deployment is Complex



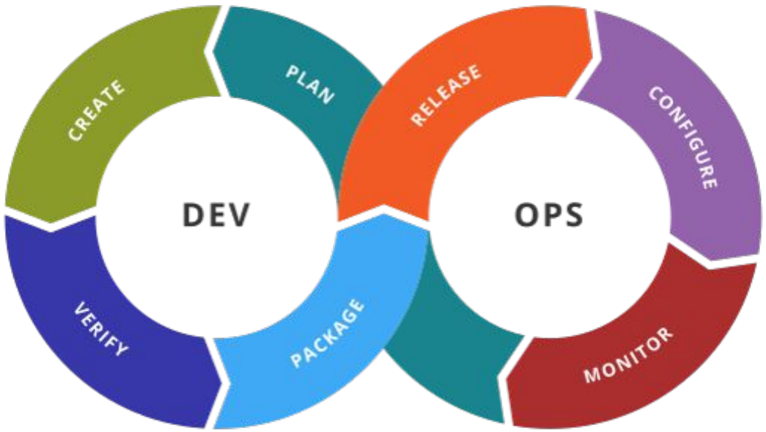

Sculley, et al. *"Hidden technical debt in machine learning systems."* NeurIPS 28 (2015): 2503-2511.

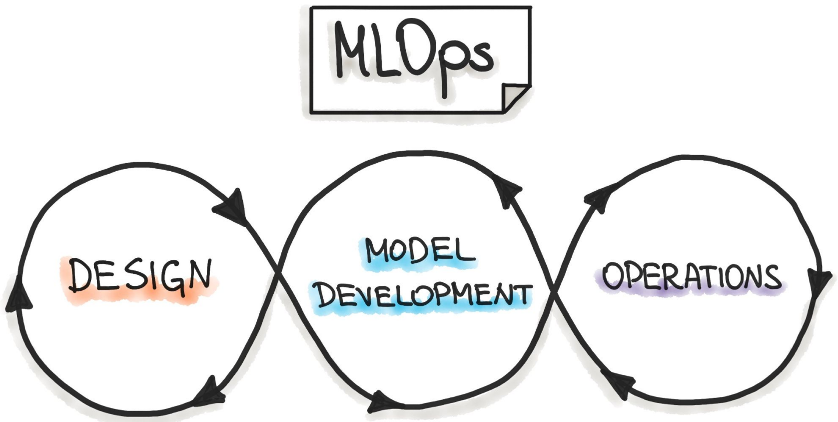
# Pipeline Automation and MLOps




Focus: experimenting, deploying, scaling training and serving, model monitoring and updating

# DevOps and MLOps



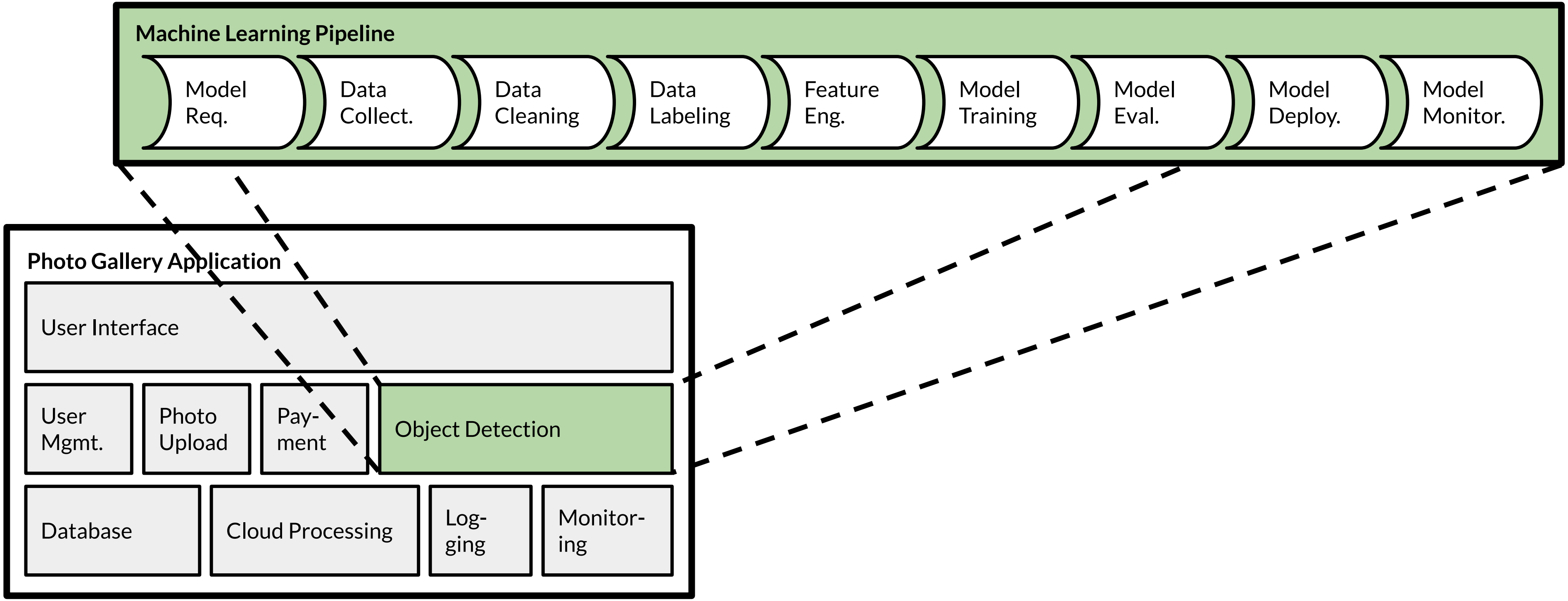




Set of practices for continuous delivery; relies on heavy automation, e.g., continuous delivery, monitoring

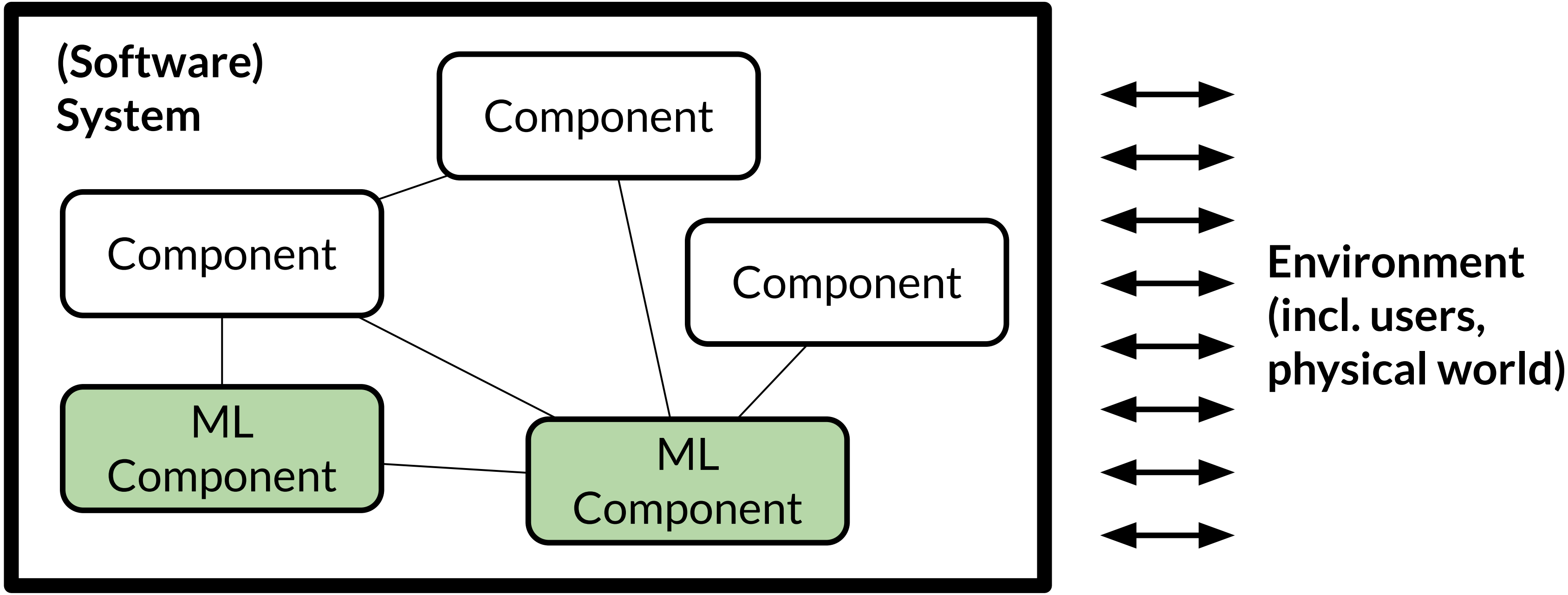
Automation around Machine Learning pipeline, including training, evaluation, versioning, and deployment

**Think about MLOps as a specialized subset of DevOps for machine learning applications**

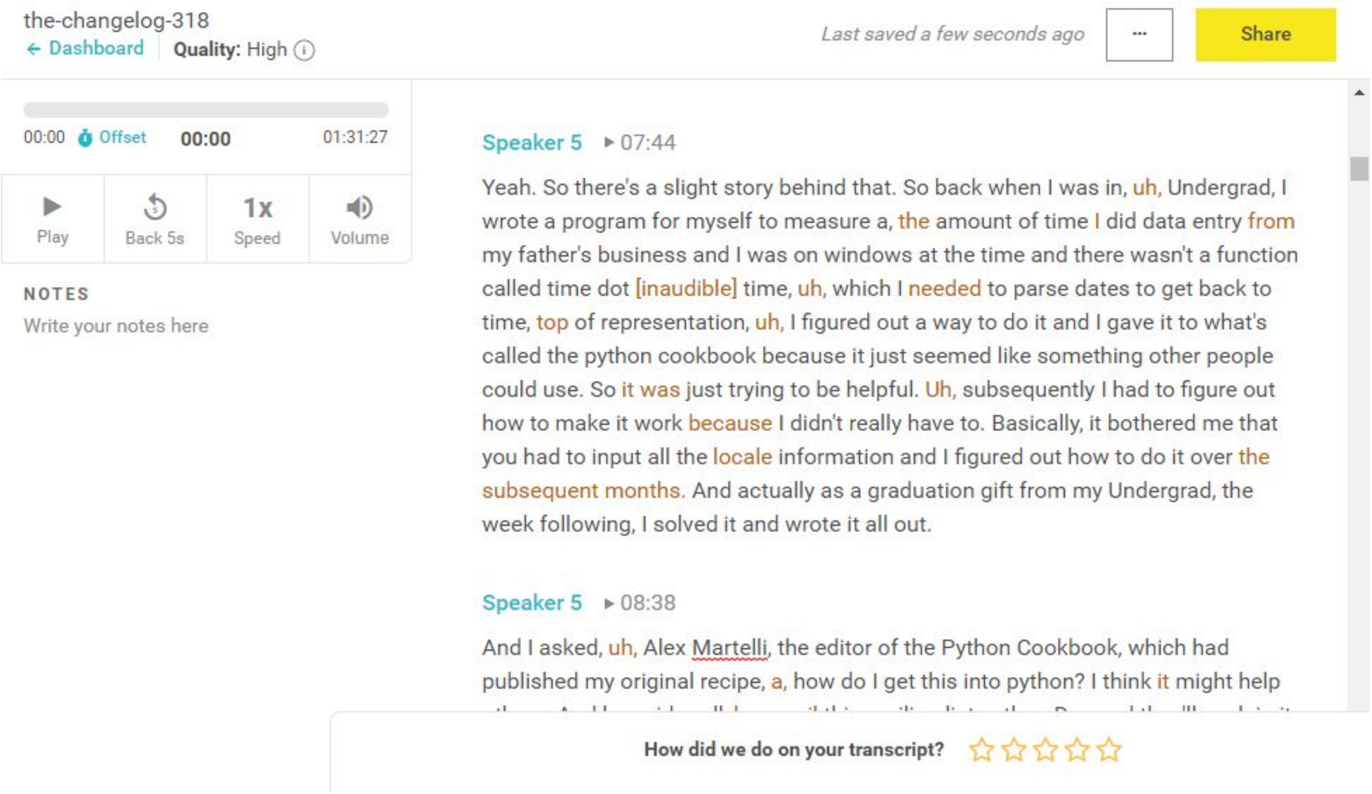
# ML is a Component in a System



# Systems Thinking



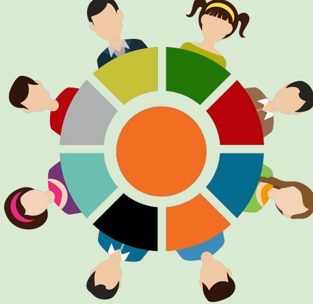
# Case Study: Transcription Service



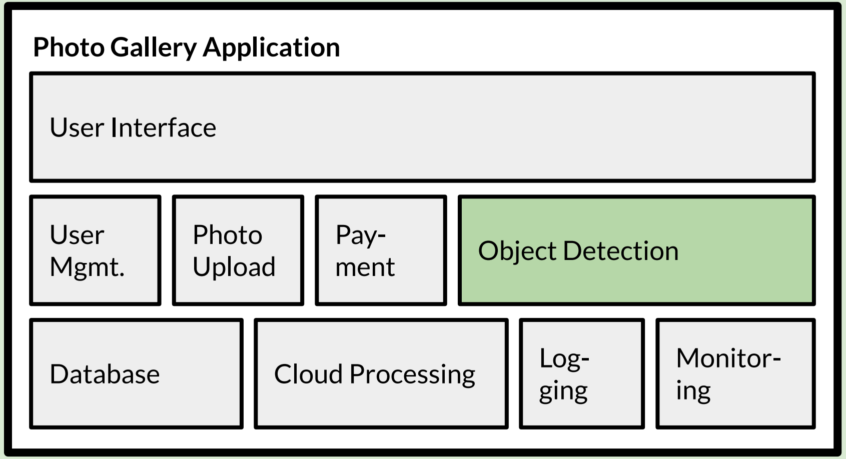
# Can you point out some ML vs non-ML components in the transcription product?

# Can you point out some ML vs non-ML components in the apps, you mentioned?

# Team Activity



Draw a diagram like this, for the app you mentioned before.



# What Changes with Machine Learning?

# Specifications & Testing in SE

```
/**
 * Return the sum of all values
 * @ensures \result = \sum int i; 0 <= i < …
 */
int sum(int[] values);
```

```
@Test
void testSentence1() {
  assertEquals(9, sum({2, 3, 4}));
}
```

# Lack of Specifications in ML

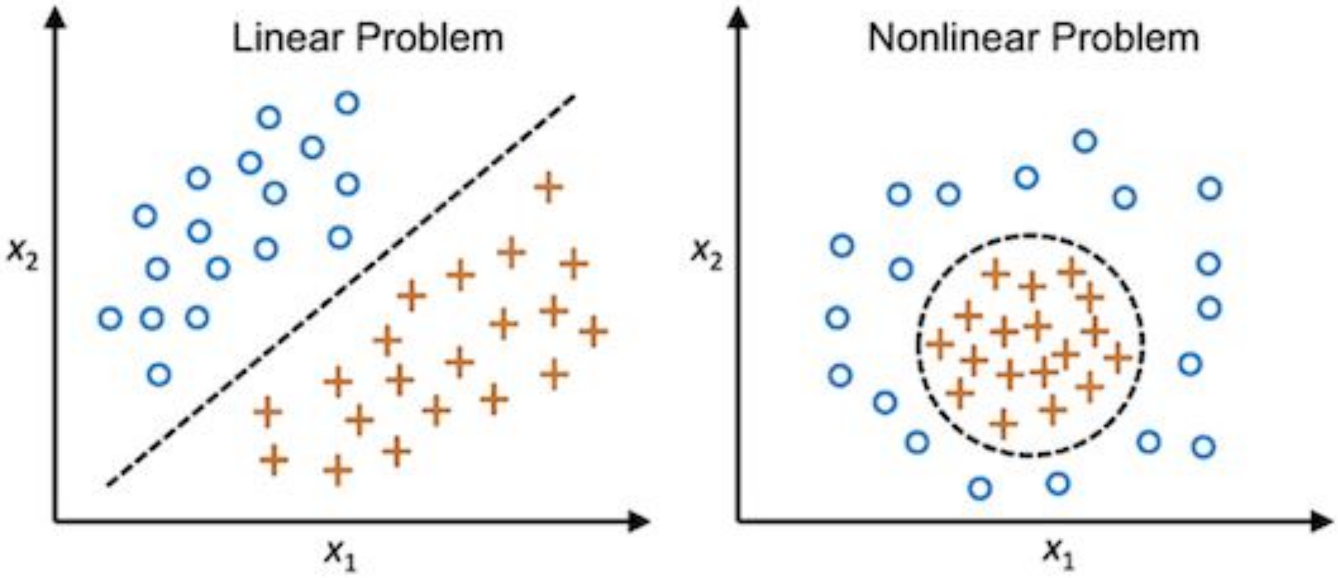
```
/**
 * Detect objects visible in image
 * ????
 */
ObjectId[] detectObjects(File image);
```

# Lack of Specifications in ML

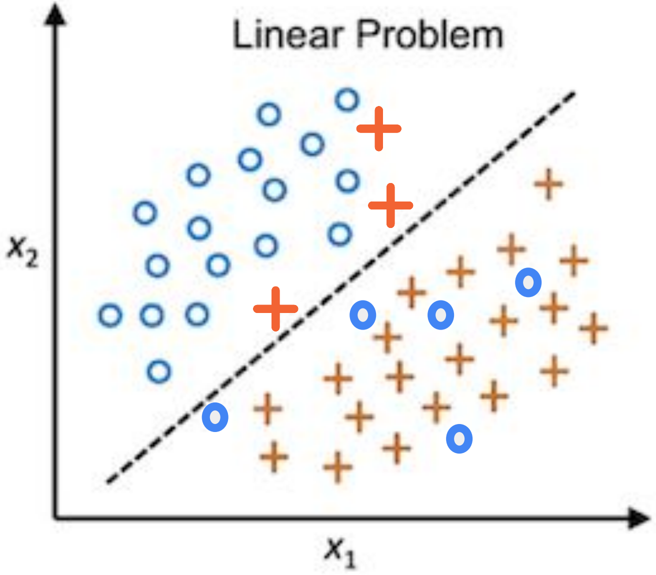
```
@Test
void testHomePhoto() {
  assertEquals({HOUSE, PLANT},
               detectObjects("img1.jpg"));
}
@Test
void testStreetPhoto() {
  assertEquals({PERSON, DOG, BICYCLE},
               detectObjects("img2.jpg"));
}
```

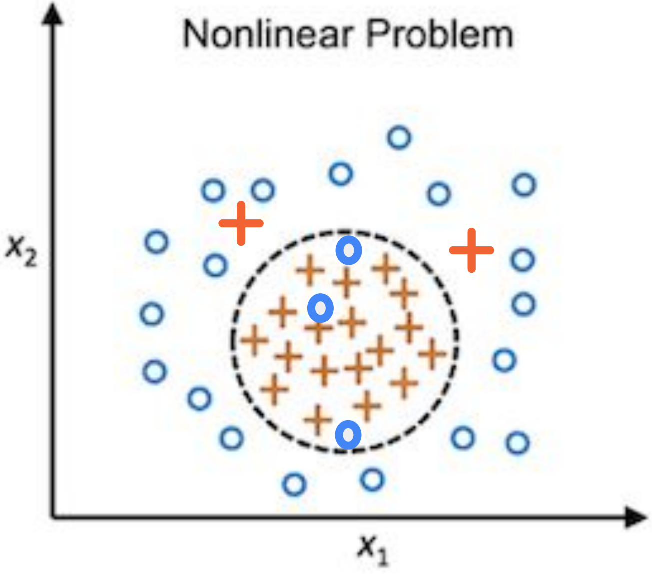
# We Cannot Define Rules for Machine Learning Models!

# ML Models Learn from Data



# Real World Data is not Ideal



# ML Model = Unreliable Function

**Object Detection Model**

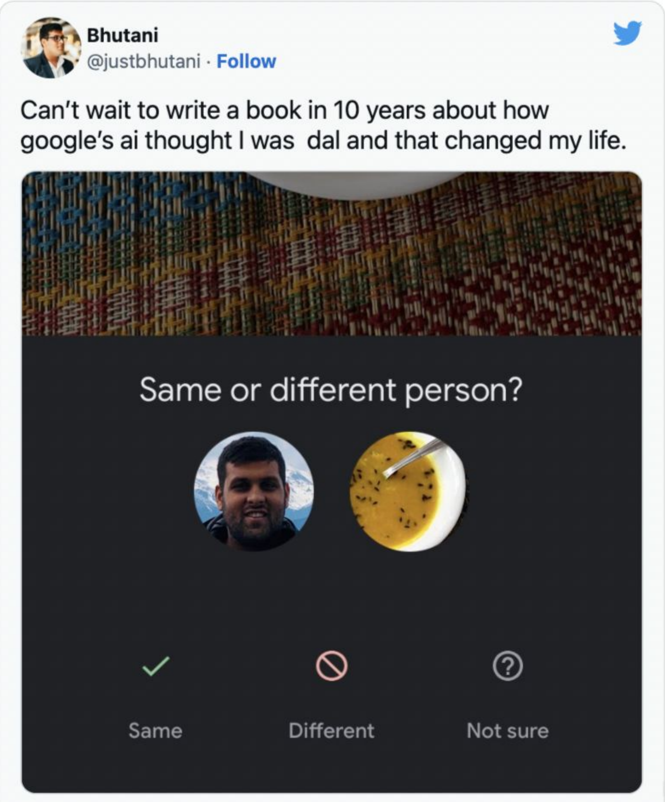
Building 99%
Path 97%
Plants 98%
Flowerpot 41%
Tree 4%

No guarantees, may make mistakes, confidence unreliable

Model often inscrutable, opaque

Evaluated in terms of accuracy, not correctness

# Model Makes Mistake




# Mistakes Cause Harms


stop hoarding and work with your ...
@jackyalcine
Follow

Google Photos, y'all fucked up. My friend's not a gorilla.



6:22 PM - 28 Jun 2015

3,352 Retweets 2,767 Likes

232 3.4K 2.8K

# All Models are Wrong!

*All models are approximations. Assumptions, whether implied or clearly stated, are never exactly true.*

***All models are wrong, but some models are useful.***

*So the question you need to ask is not "Is the model true?" (it never is) but "Is the model good enough for this particular application?"*

*George Box*

# Lack of Specifications…

… breaks modular reasoning

… challenges quality assurance

… inhibits safety and fairness reasoning

… hinders coordination across teams

(though, we didn't need ML to build low quality, harmful, and unethical software)

# Building ML-Enabled Systems

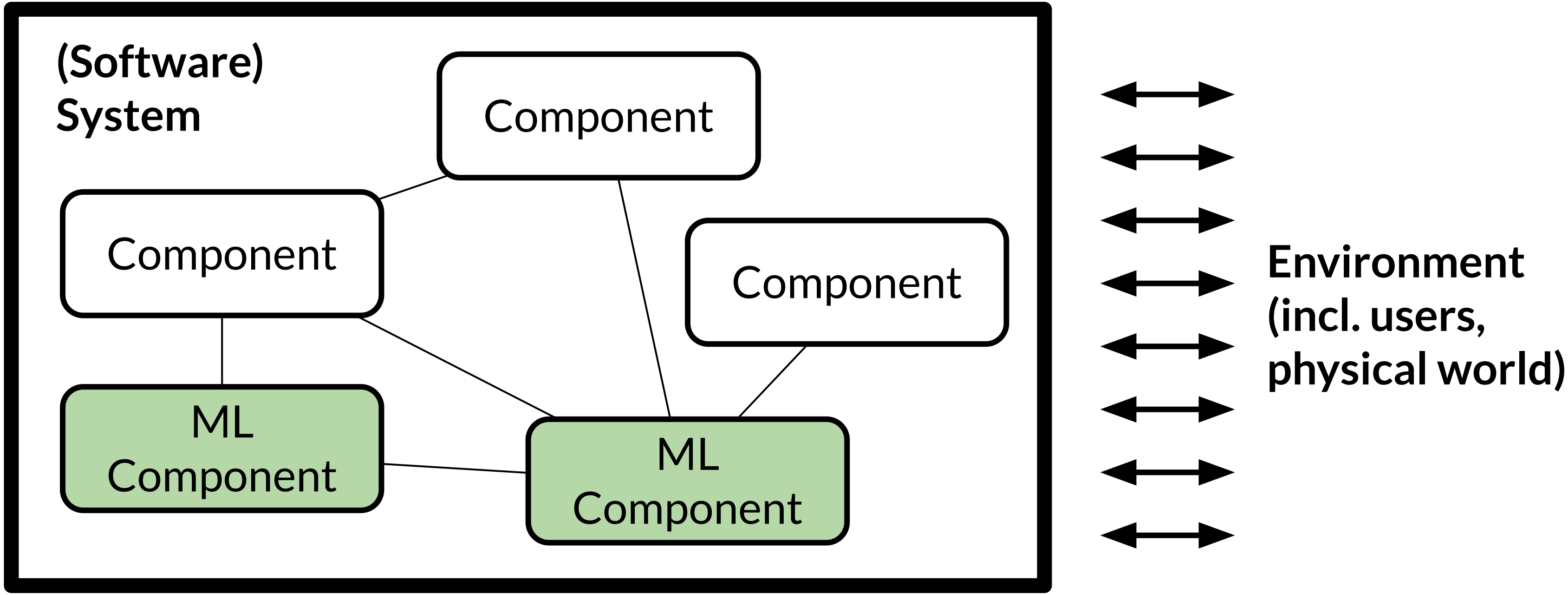
# Building ML-Enabled Systems

Understand *system* needs and goals and interactions with environment

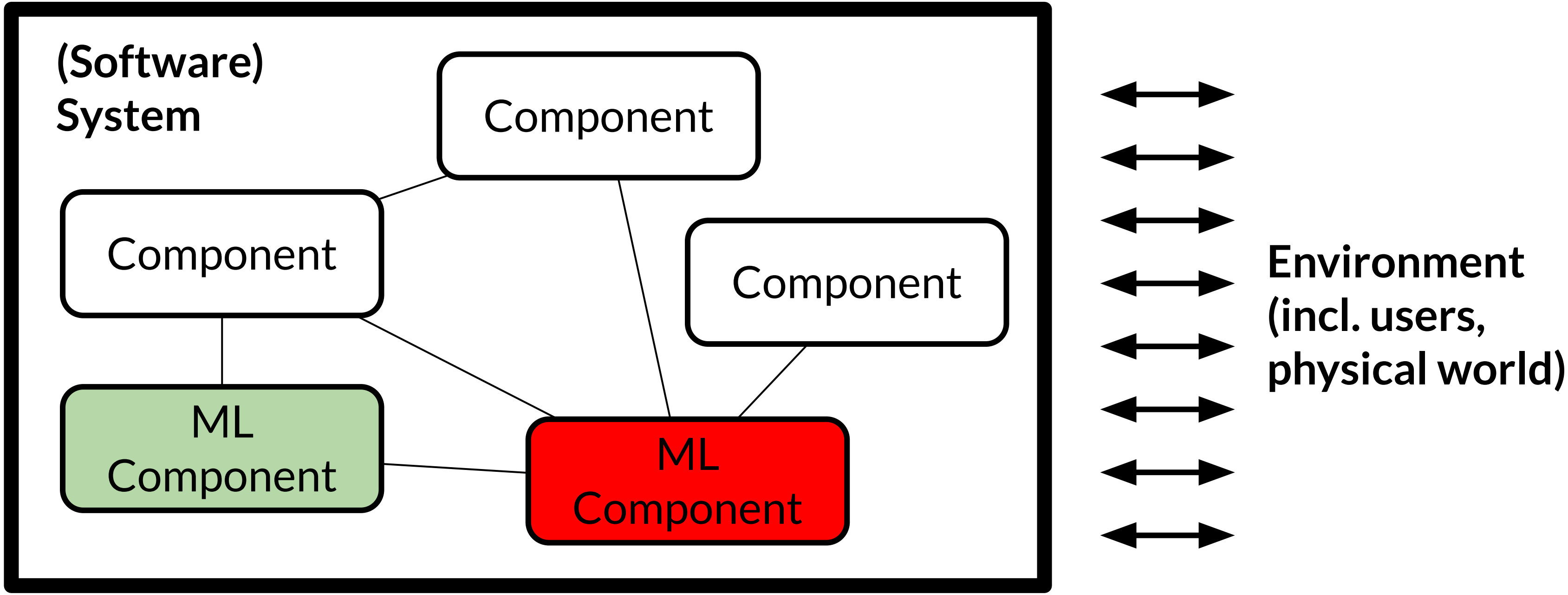
Designing components and integrating ML and non-ML parts into a *system*

Many roles and stakeholders, interdisciplinary endeavour

# Systems Thinking



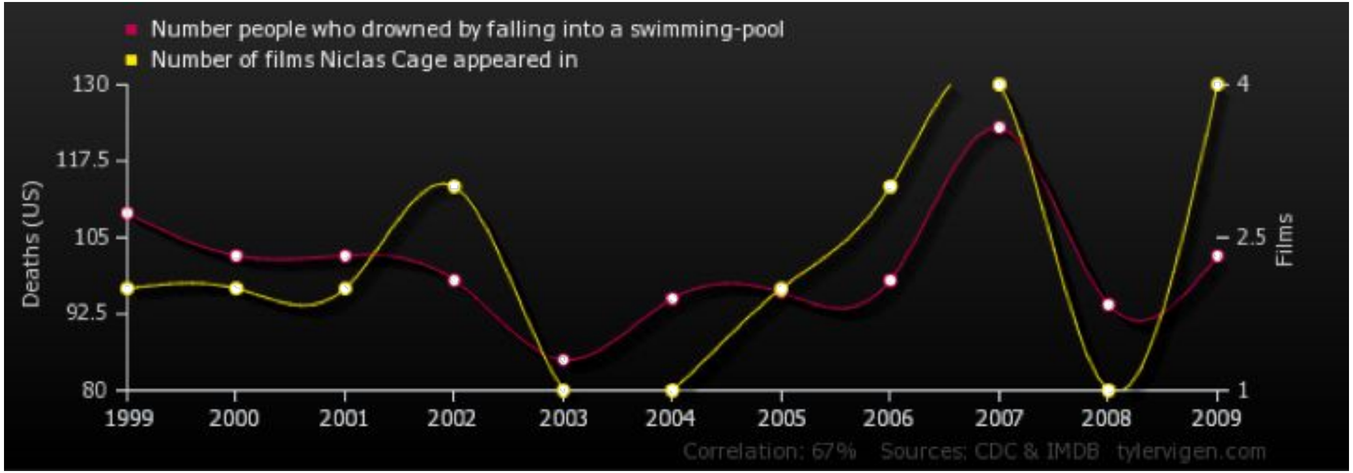
# What to do when the ML component makes mistake?

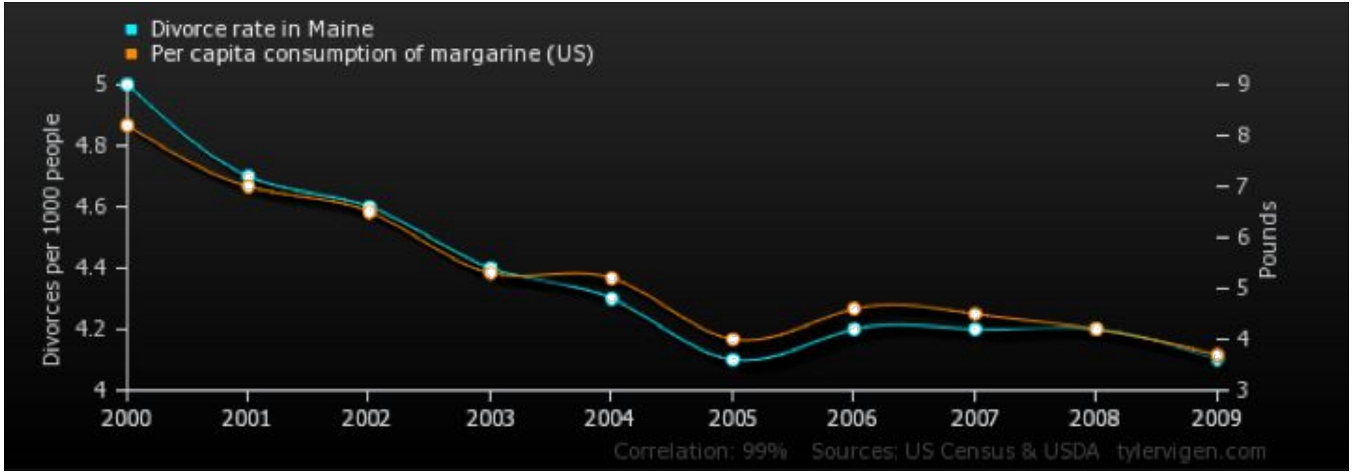


# Commons Sources of Wrong Prediction

- Insufficient training data
- Noisy training data
- Biased training data
- Overfitting
- Poor model fit, poor model selection, poor hyperparameters
- Missing context, missing important features
- Noisy inputs
- "Out of distribution" inputs

# Correlation vs Causation



# Reasons Barely Matter

- No model is always "correct". Some mistakes are unavoidable
- Anticipate the eventual mistake
- Make the system safe despite mistakes

Consider the rest of the system…

# Example: Smart Toaster

# Safety is a System Property

Code/models are not unsafe, cannot harm people

Systems can interact with the environment in ways that are unsafe

How can you ensure that smart toaster does not burn the kitchen?

# Safety Assurance in/outside the Model

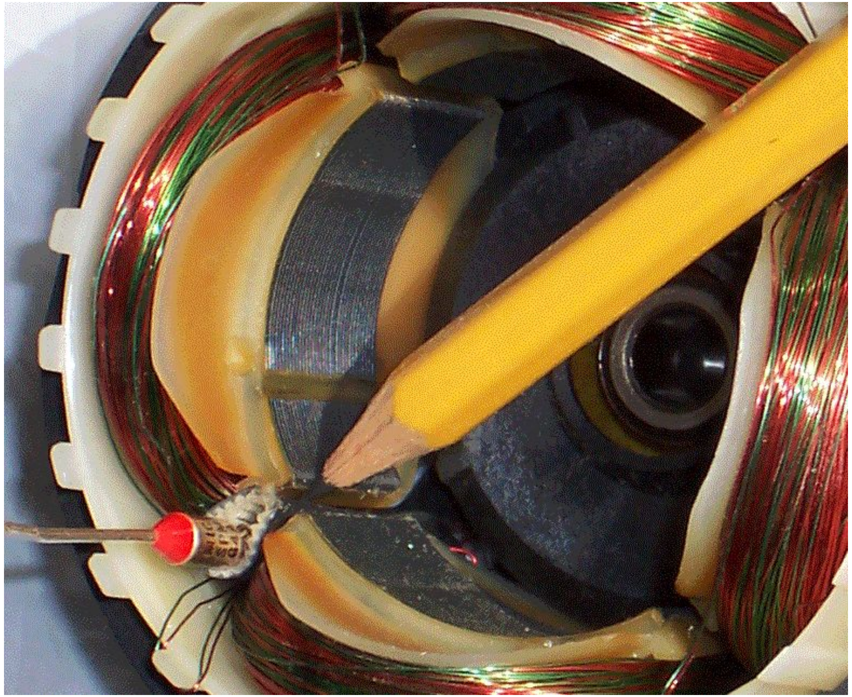
**In the model**

- Ensure maximum toasting time
- Use heat sensor and past outputs for prediction

Hard to make guarantees

**Outside the model**

- Simple code check for max toasting time
- Non-ML rule to shut down if too hot
- Hardware solution: thermal fuse



# Human in the Loop

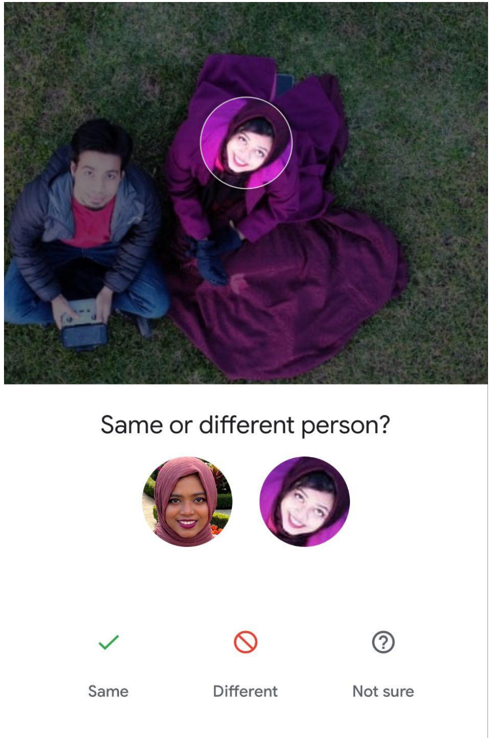
to me

Hey Nadia,

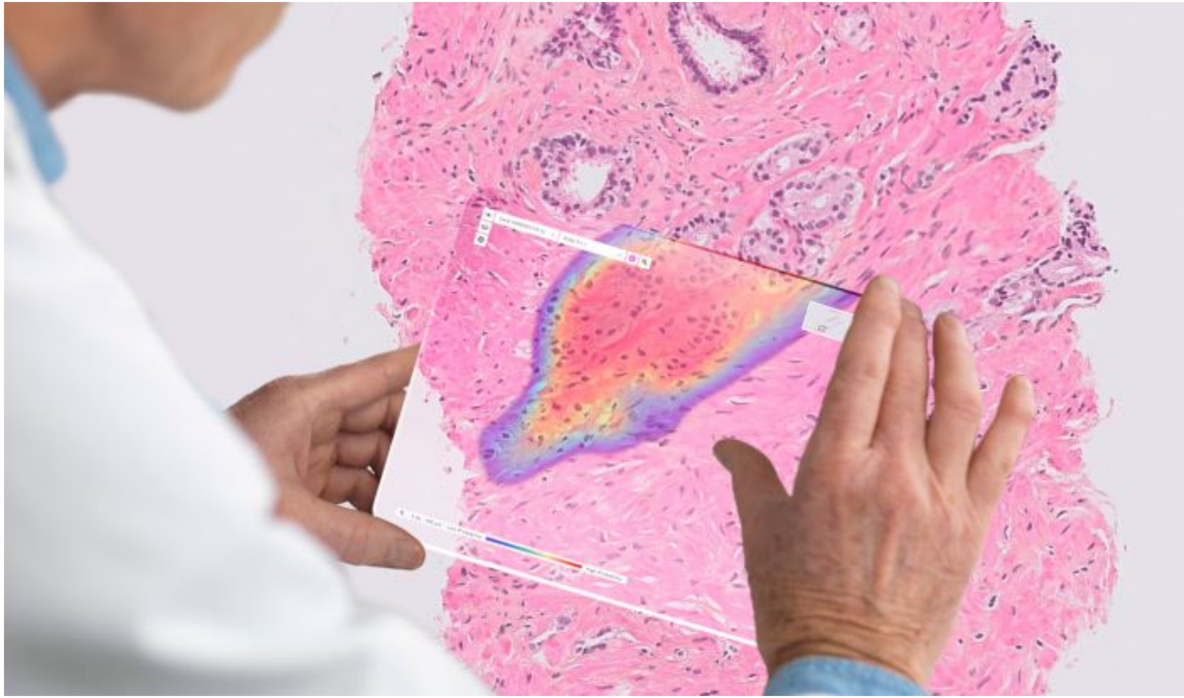
Does Wednesday work for you?

Sure, what time? | Yes, what time? | No, it doesn't.

Reply | Forward



# Human in the Loop



AI powered diagnostic systems for cancer does not replace pathologists

# Human in the Loop

## Food delivery robot pauses operations after Monday incident

Emily Ackerman relies on a wheelchair for mobility and was trapped on Forbes Avenue when robot wouldn't move

# Many different strategies

Based on fault-tolerant design, assuming that there will be software/ML mistakes or environment changes violating assumptions

- Human in the loop
- Undoable actions
- Guardrails
- Mistake detection and recovery (monitoring, doer-checker, fail-over, redundancy)
- Containment and isolation

# Actions to Consider While Presenting Intelligence

**Automate**: Take action on user's behalf

**Prompt**: Ask the user if an action should be taken

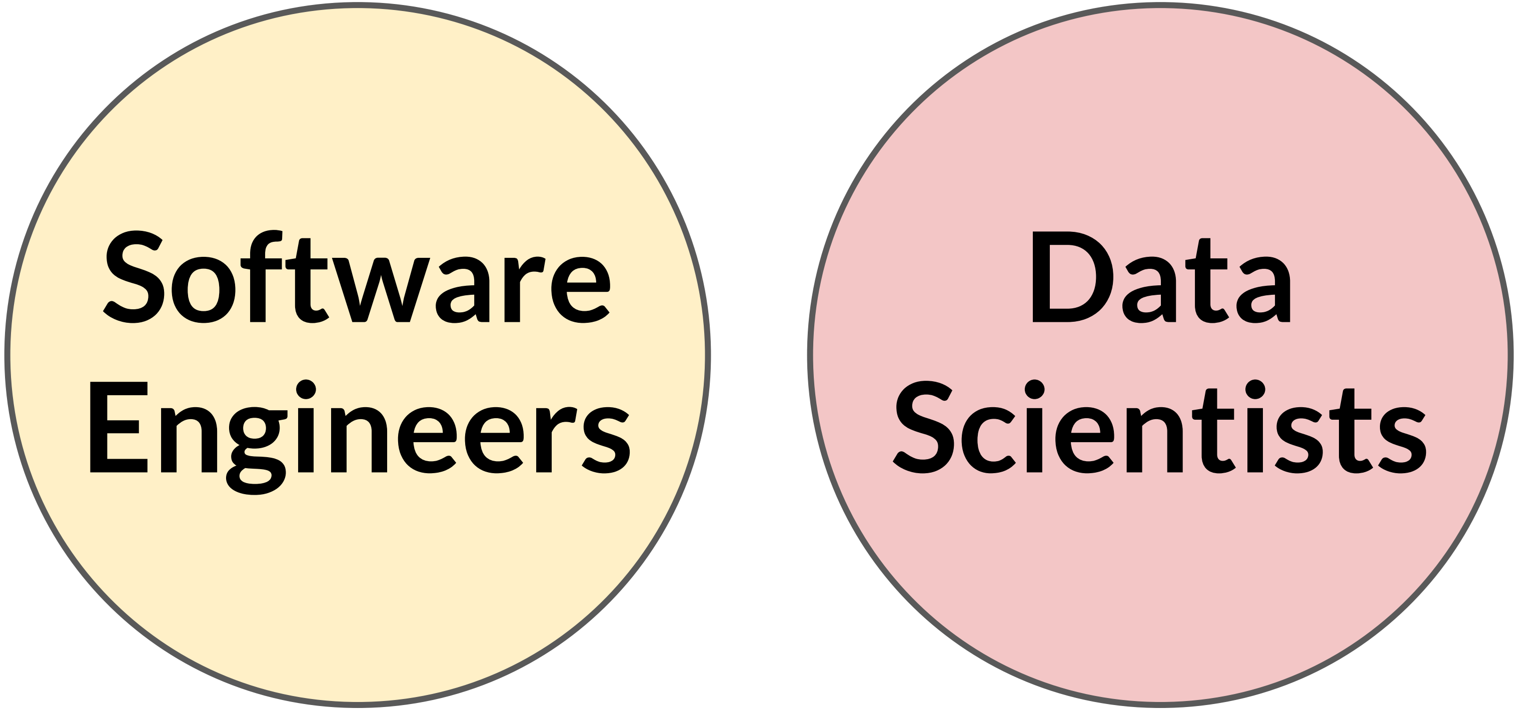
**Organize/Annotate/Augment**: Add information to a display

**Hybrids of these**

For your mentioned apps, which of the **Automate**, **Prompt**, or **Augment** would you use, and how?

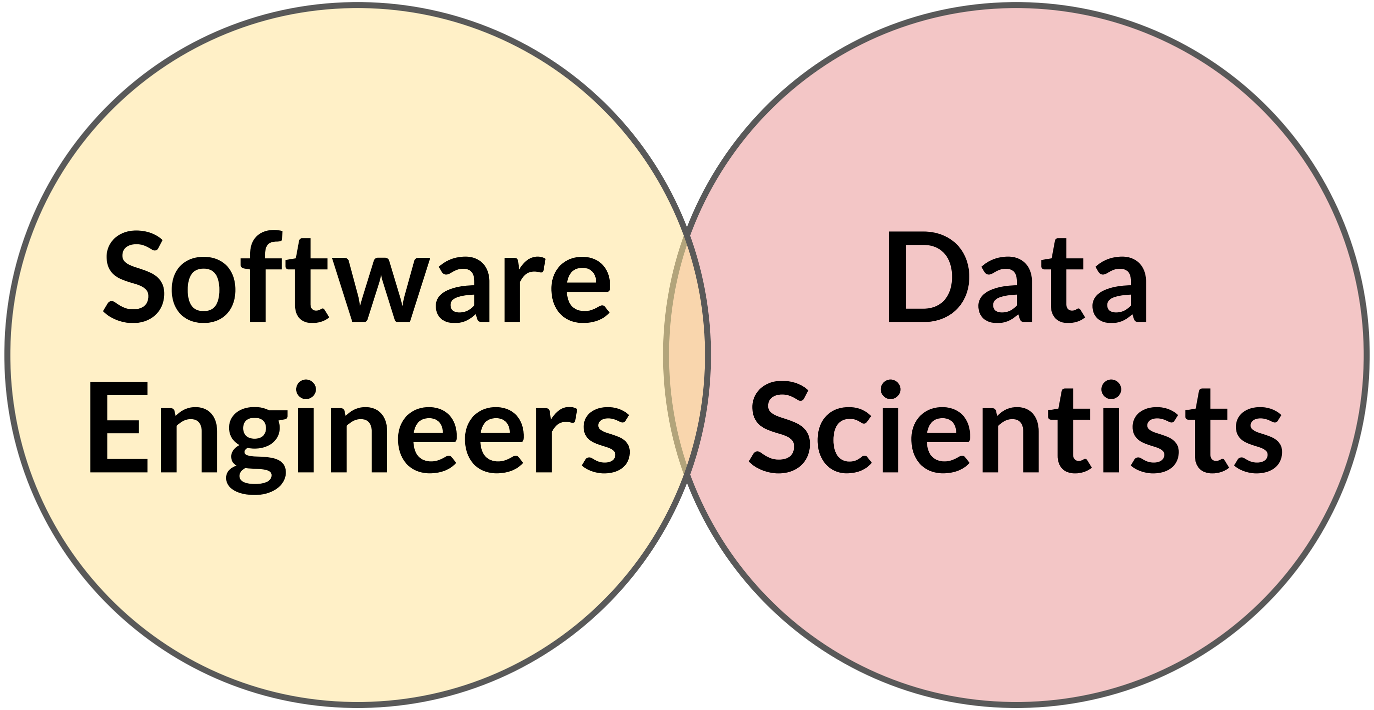
# Building ML-Enabled Systems Need Team Effort

# We cannot do it alone

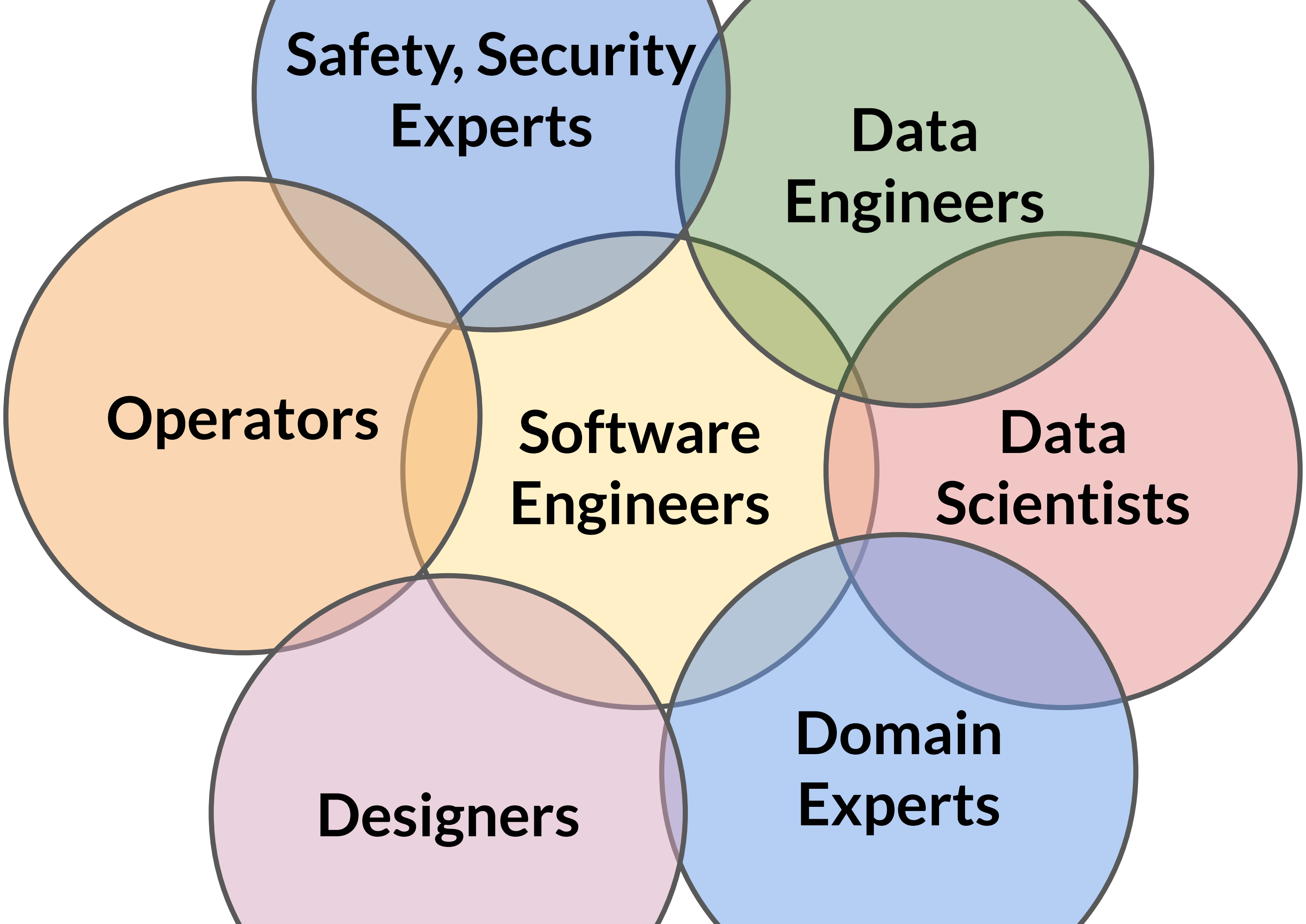



and data engineers + domain specialists + operators + business team +
project managers + designers, UI experts + safety, security specialists + lawyers + ...

# Interdisciplinary Teams




and data engineers + domain specialists + operators + business team +
project managers + designers, UI experts + safety, security specialists + lawyers + ...


Safety, Security Experts
Data Engineers
Operators
Software Engineers
Data Scientists
Designers
Domain Experts

# T-Shaped Professionals

**I-Shaped**
Deep expertise in one topic

**Generalist**
Broad knowledge of many topics, but not expert in any

**T-Shaped**
Expert in one topic and broad knowledge of other topics

# Why do 87% of data science projects never make it into production?

VB Staff

July 19, 2019 4:

Collaboration Problems

And the third issue, intimately connected to those silos, is the lack of collaboration. Data scientists have been around since the 1950s — and they were individuals sitting in a basement working behind a terminal. But now that it's a team sport, and the importance of that work is now being embedded into the fabric of the company, it's essential that every person on the team is able to collaborate with everyone else: the data engineers, the data stewards, people that understand the data science, or analytics, or BI specialists, all the way up to DevOps and engineering.

"This is a big place that holds companies back because they're not used to collaborating in this way," Leff says. "Because when they take those insights, and they flip them over the wall, now you're asking an engineer to rewrite a data science model created by a data scientist, how's that work out, usually?"

https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

# WHY DO MACHINE LEARNING PROJECTS FAIL?

Think ahead to production so that you don't let your machine learning project collapse before it even gets started.

Rahul Agarwal

| Expert Columnist

Agarwal is a senior data scientist currently working with Walm

## 4. YOUR MODEL MIGHT NOT EVEN GO TO PRODUCTION

Let's imagine that you've created this impressive machine learning model. It gives 90 percent accuracy, but it takes around 10 seconds to fetch a prediction. Or maybe it takes a lot of resour to predict.

Is that ac

most likely no.

**Mismatch in Assumptions**

https://builtin.com/machine-learning/why-do-machine-learning-projects-fail

# Top 10 Reasons Why 87% of Machine Learning Projects Fail

In this article, find out why 87% of machine learning projects fail.

by Prajeen MV · Oct. 13, 20 · AI Zone · Opinion

## A Disconnect Between Data Science and Traditional Software Development

A disconnect between Data Science and traditional Software development is another major factor. Traditional software development tends to be more predictable and measurable.

**However, Data science is still part-research and part-engineering.**

**Different Ways of Working**

https://dzone.com/articles/top-10-reasons-why-87-of-the-machine-learning-proj

# Frustrations shared in Twitter…

All ML projects which turned into a disaster in my career have a single common point:

🚨 I didn't understand the business context first, got over-excited about the tech, and jumped into coding too early.

1:08 PM · Mar 12, 2022 · Twitter Web App

**297** Retweets **39** Quote Tweets **1,786** Likes

Machine Learning lives in an uncanny valley btw Science and Engineering.

It's the worst of both worlds.

We don't care about understanding, just making things "work" (bad science).

We don't care if things work in the real world, just on contrived benchmarks (bad engineering).

6:45 AM · Jan 29, 2022 · Twitter Web App

**202** Retweets **37** Quote Tweets **1,451** Likes

We need better collaboration practices, learnings for SE itself

# Decades of SE Experience

Development lifecycles

Requirements engineering

Safety engineering

Big-data architectures

Integration & system testing, testing in production

# Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process

Nadia Nahar
nadian@andrew.cmu.edu
Carnegie Mellon University
Pittsburgh, PA, USA

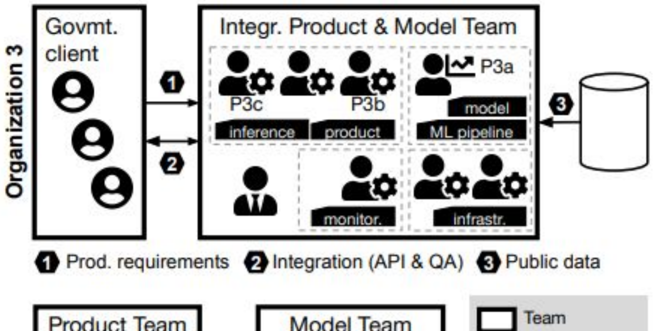
Shurui Zhou
University of Toronto
Toronto, Ontario, Canada

Grace Lewis
Carnegie Mellon Software Engineering Institute
Pittsburgh, PA, USA

Christian Kästner
Carnegie Mellon University
Pittsburgh, PA, USA

## ABSTRACT

The introduction of machine learning (ML) components in software projects has created the need for software engineers to collaborate with data scientists and other specialists. While collaboration can always be challenging, ML introduces additional challenges with its exploratory model development process, additional skills and knowledge needed, difficulties testing ML systems, need for continuous evolution and monitoring, and non-traditional quality requirements such as fairness and explainability. Through inter-

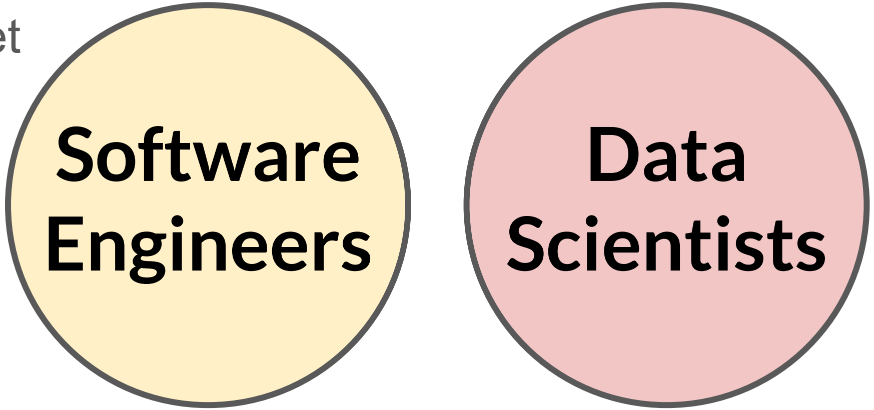


# Collaboration Challenges at Interfaces between Roles & Teams

Business vs. engineering vs. science mindset

Inconsistent vocabulary

Different priorities, conflicting goals



*Collaboration Points* | *Themes*

| Collaboration Points | Communication | Documentation | Engineering | Process |
| --- | --- | --- | --- | --- |
| **Requirements and Planning** | | | | |
| - Product and Model Requirements | ✓ | ✓ | ✓ | ✓ |
| - Project Planning | ✓ | – | – | ✓ |
| **Training Data** | | | | |
| - Negotiating Data Quality and Quantity | ✓ | ✓ | ✓ | ✓ |
| **Product-Model Integration** | | | | |
| - Responsibility and Cultural Clashes | ✓ | – | ✓ | ✓ |
| - Quality Assurance for Model and Product | ✓ | ✓ | ✓ | ✓ |

Communication   Documentation   Engineering   Process

# Summary

- Consider ML as an unreliable component of the System
- All ML models make mistakes
- Safeguard ML models considering the system view
- Building ML systems need team efforts
- Collaborative culture among software engineers, data scientists, and other stakeholders are necessary
- The role of Software Engineering is important in ML

# CMU 17-645: Machine Learning in Production

## Fundamentals of Engineering AI-Enabled Systems

**Holistic system view:** AI and non-AI components, pipelines, stakeholders, environment interactions, feedback loops

**Requirements:**
- System and model goals
- User requirements
- Environment assumptions
- Quality beyond accuracy
- Measurement
- Risk analysis
- Planning for mistakes

**Architecture + design:**
- Modeling tradeoffs
- Deployment architecture
- Data science pipelines
- Telemetry, monitoring
- Anticipating evolution
- Big data processing
- Human-AI design

**Quality assurance:**
- Model testing
- Data quality
- QA automation
- Testing in production
- Infrastructure quality
- Debugging

**Operations:**
- Continuous deployment
- Contin. experimentation
- Configuration mgmt.
- Monitoring
- Versioning
- Big data
- DevOps, MLOps

**Teams and process:** Data science vs software eng. workflows, interdisciplinary teams, collaboration points, technical debt

## Responsible AI Engineering

- Provenance, versioning, reproducibility
- Safety
- Security and privacy
- Fairness
- Interpretability and explainability
- Transparency and trust

Ethics, governance, regulation, compliance, organizational culture



https://ckaestne.github.io/seai/

# Further Readings


- Hulten, Geoff. "Building Intelligent Systems: A Guide to Machine Learning Engineering." Apress, 2018.
- Nahar, Nadia, et al. “Collaboration Challenges in Building ML-Enabled Systems: Communication, Documentation, Engineering, and Process.” In Proceedings of the 44th International Conference on Software Engineering (ICSE), May 2022.
- Amershi, Saleema, et al. "Software Engineering for Machine Learning: A Case Study." In 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), pp. 291-300. IEEE, 2019.
- Giray, Görkem. "A software engineering perspective on engineering machine learning systems: State of the art and challenges." Journal of Systems and Software 180 (2021): 111031.
- Ozkaya, Ipek. "What Is Really Different in Engineering AI-Enabled Systems?" IEEE Software 37, no. 4 (2020): 3-6.
- Passi, Samir, and Phoebe Sengers. "Making data science systems work." Big Data & Society 7, no. 2 (2020).