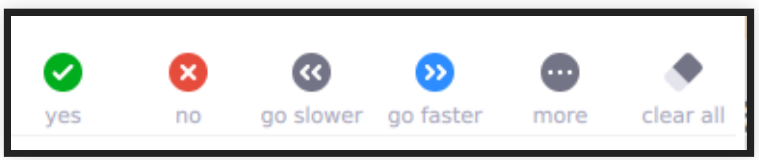
INTRODUCTION AND MOTIVATION

Eunsuk Kang & Christian Kaestner

LECTURE LOGISTICS DURING A PANDEMIC

If you can hear me, open the participant panel in Zoom and check "yes"



FIRST OF: THIS IS NOT NORMAL. WE UNDERSTAND.

Speaker notes

Picture AP Photo/Noah Berger, https://www.abc27.com/news/thats-2020-photographers-california-wildfire-image-a-sign-of-the-times/

FIRST OF: THIS IS NOT NORMAL. WE UNDERSTAND.

Expect:

- Internet and bandwidth issues
- Timezone issues
- Distractions -- parents, siblings, pets
- Feeling isolated, feeling overwhelmed
- Many additional sources of stress
- Hard time dealing with -gestures widely- *everything*...

Talk to us about accommodations of any kind

SIMULATING IN-CLASS EXPERIENCE

Discussions and interactions are important. We'll have regular in-class discussions and exercises

- Use chat, "raise hand" feature, or just speak
- If possible, keep camera on, muted by default
- Set preferred name in Zoom
- If possible, attend lecture and recitation live, recordings only as backup

- Suggestion: Have chat and participant list open, maybe separate window for gallery view for faces, second monitor highly recommended
- **Contact us for accommodations!**

PERSONAL CONNECTION

This is hard. We know.

- Talk inside and outside of class
- We are here at least 10 min before class and stay after class if you have questions, want to chat
- We encourage collaboration in all assignments, even "individual" assignments and reading quizzes

LEARNING GOALS

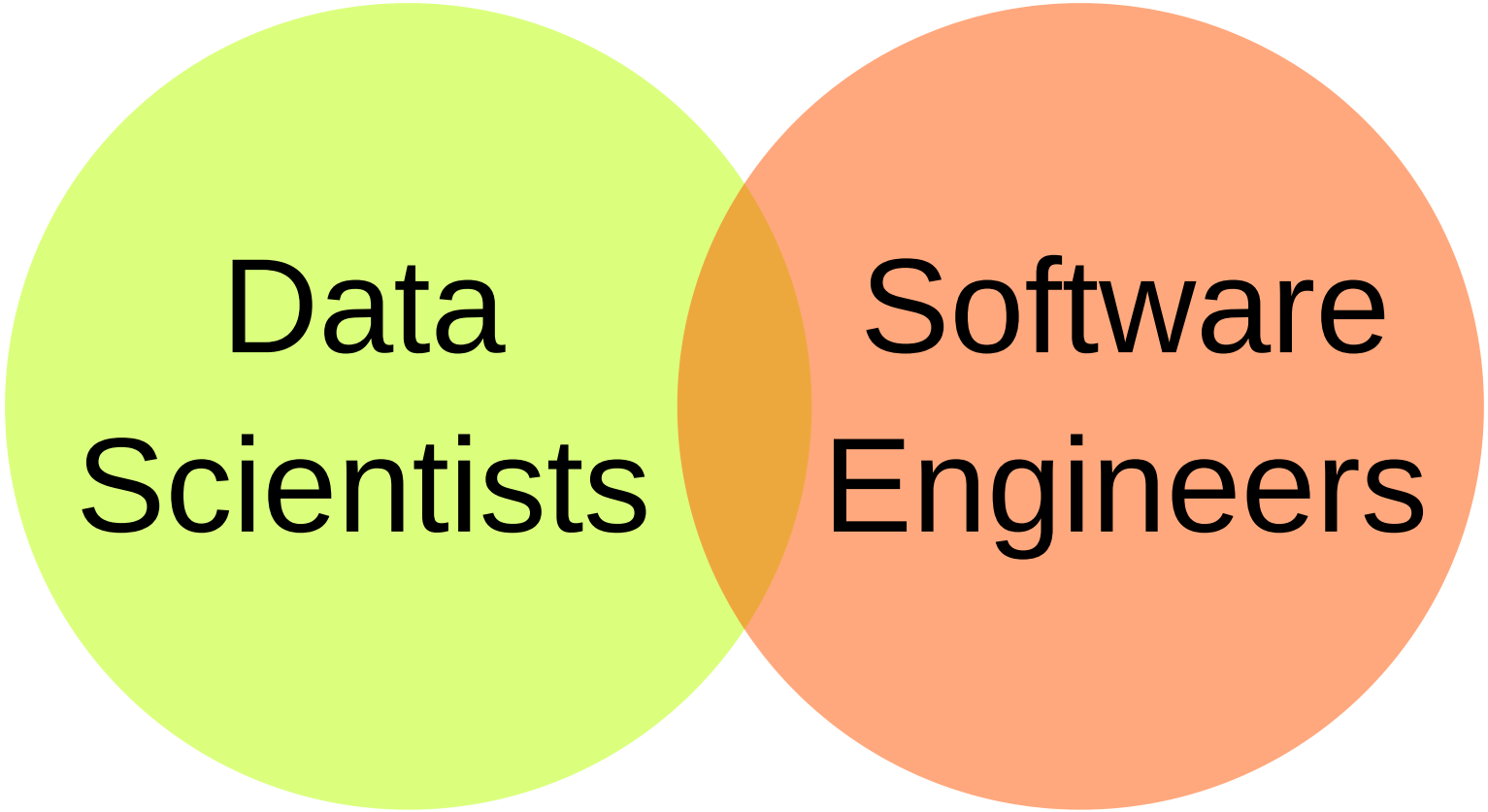
- Understand how AI components are parts of larger systems
- Illustrate the challenges in engineering an AI-enabled system beyond accuracy
- Explain the role of specifications and their lack in machine learning and the relationship to deductive and inductive reasoning
- Summarize the respective goals and challenges of software engineers vs data scientists

DISCLAIMERS

This is a fairly new class in a rapidly evolving field.

Many experiments for online teaching.

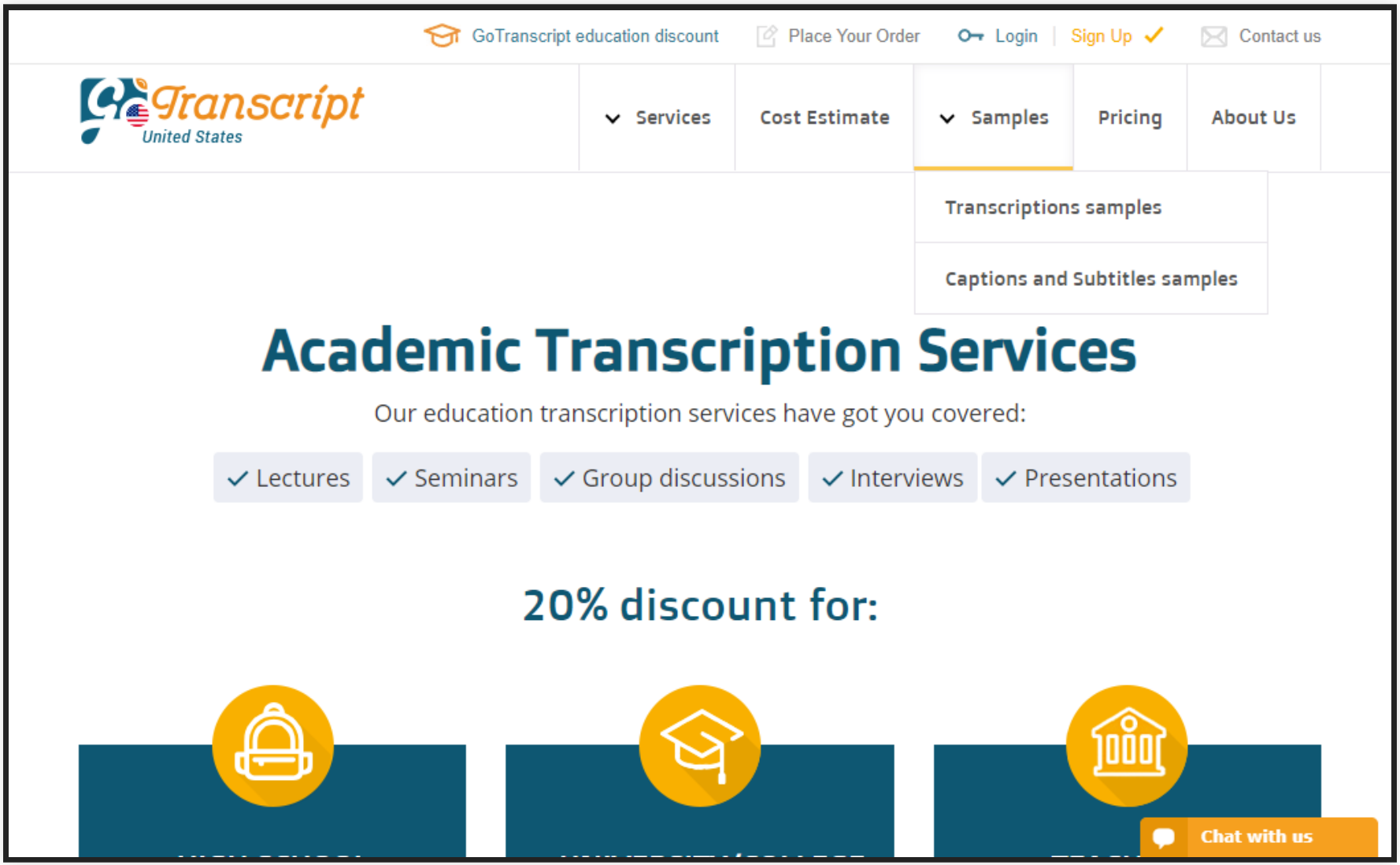
We are software engineers.


Data
Scientists
Software
Engineers

AGENDA


Preliminaries
Case Study
Syllabus
Introductions
Specifications

CASE STUDY: THE TRANSCRIPTION SERVICE STARTUP


GoTranscript education discount
Place Your Order
Login
Sign Up
Contact us
Transcript
United States
Services
Cost Estimate
Samples
Pricing
About Us
Transcriptions samples
Captions and Subtitles samples
Academic Transcription Services
Our education transcription services have got you covered:
✓ Lectures
✓ Seminars
✓ Group discussions
✓ Interviews
✓ Presentations
20% discount for:
Chat with us

TRANSCRIPTION SERVICES

- Take audio or video files and produce text.
    - Used by academics to analyze interview text
    - Podcast show notes
    - Subtitles for videos
- State of the art: Manual transcription, often mechanical turk (1.5 $/min)

THE STARTUP IDEA

PhD research on domain-specific speech recognition, that can detect technical jargon

DNN trained on public PBS interviews + transfer learning on smaller manually annotated domain-specific corpus

Research has shown amazing accuracy for talks in medicine, poverty and inequality research, and talks at Ruby programming conferences; published at top conferences

Idea: Let's commercialize the software and sell to academics and conference organizers

LIKELY CHALLENGES IN BUILDING COMMERCIAL PRODUCT?

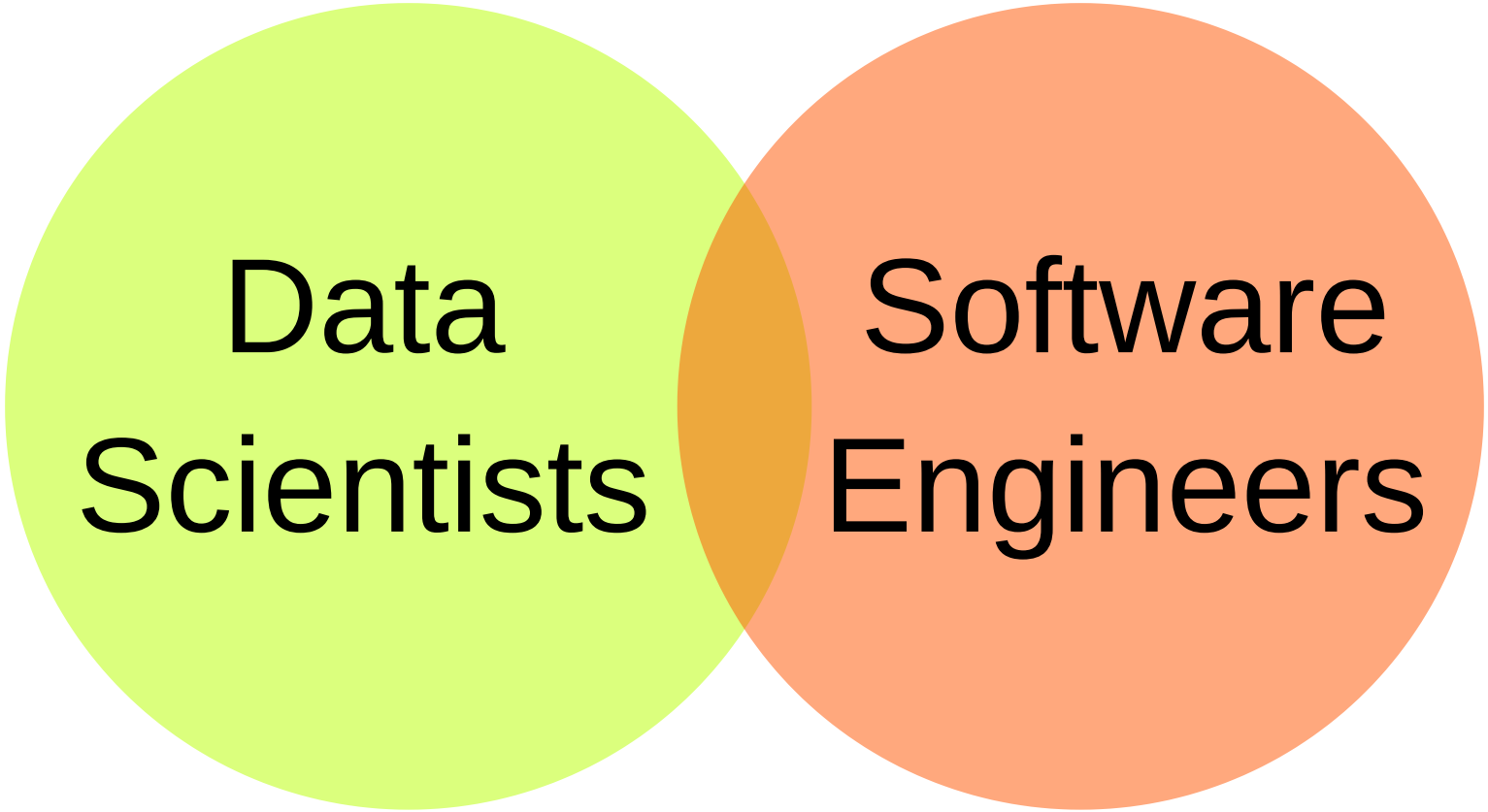
- Think about 2 challenges that the team will likely focus when turning their research into *a product* (business, development, deployment, operation)
- At least one challenge should not be about learning a good ML model, but about *building a product*
- Everybody, **type 2 likely challenges in the chat but *do not send them yet*. Vote "yes" when done.**

QUALITIES OF INTEREST ("ILITIES")

- Quality is about more than the absence of defects
- Quality in use (effectiveness, efficiency, satisfaction, freedom of risk, ...)
- Product quality (functional correctness and completeness, performance efficiency, compatibility, usability, dependability, scalability, security, maintainability, portability, ...)
- Process quality (manageability, evolvability, predictability, ...)

- "Quality is never an accident; it is always the result of high intention, sincere effort, intelligent direction and skillful execution; it represents the wise choice of many alternatives." (many attributions)

WHAT QUALITIES ARE IMPORTANT FOR A GOOD COMMERCIAL TRANSCRIPTION PRODUCT?


Data
Scientists
Software
Engineers

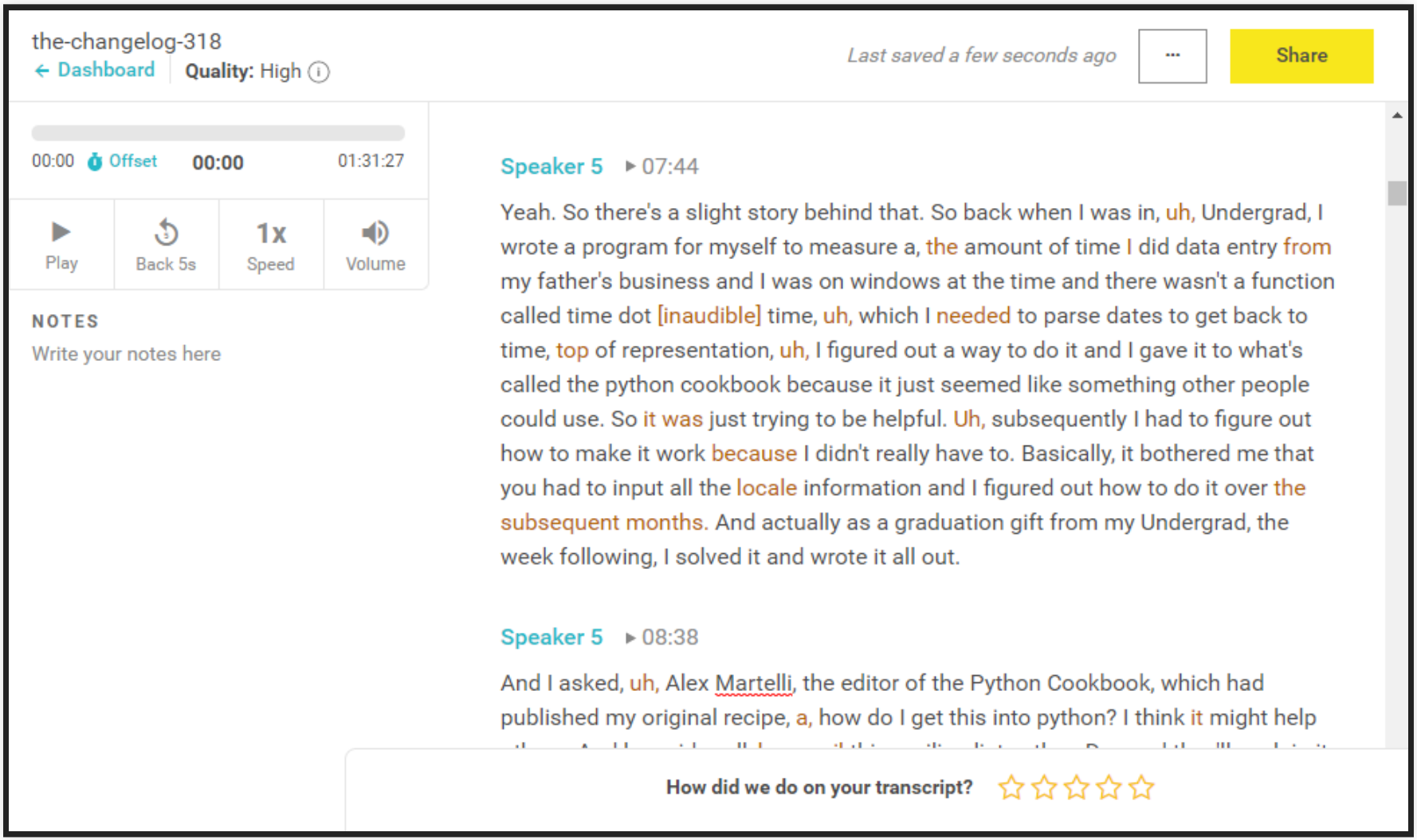
DATA SCIENTIST

- Often fixed dataset for training and evaluation (e.g., PBS interviews)
- Focused on accuracy
- Prototyping, often Jupyter notebooks or similar
- Expert in modeling techniques and feature engineering
- Model size, updateability, implementation stability typically does not matter

SOFTWARE ENGINEER

- Builds a product
- Concerned about cost, performance, stability, release time
- Identify quality through customer satisfaction
- Must scale solution, handle large amounts of data
- Detect and handle mistakes, preferably automatically
- Maintain, evolve, and extend the product over long periods
- Consider requirements for security, safety, fairness

LIKELY COLLABORATION CHALLENGES?


the-changelog-318
← Dashboard
Quality: High
Last saved a few seconds ago
...
Share
00:00
Offset
00:00
01:31:27
Play
Back 5s
1x
Speed
Volume
NOTES
Write your notes here
Speaker 5 ▸ 07:44
Yeah. So there's a slight story behind that. So back when I was in, uh, Undergrad, I wrote a program for myself to measure a, the amount of time I did data entry from my father's business and I was on windows at the time and there wasn't a function called time dot [inaudible] time, uh, which I needed to parse dates to get back to time, top of representation, uh, I figured out a way to do it and I gave it to what's called the python cookbook because it just seemed like something other people could use. So it was just trying to be helpful. Uh, subsequently I had to figure out how to make it work because I didn't really have to. Basically, it bothered me that you had to input all the locale information and I figured out how to do it over the subsequent months. And actually as a graduation gift from my Undergrad, the week following, I solved it and wrote it all out.
Speaker 5 ▸ 08:38
And I asked, uh, Alex Martelli, the editor of the Python Cookbook, which had published my original recipe, a, how do I get this into python? I think it might help
How did we do on your transcript?

Speaker notes

Highlights challenging fragments. Can see what users fix inplace to correct. Star rating for feedback.

EXAMPLES FOR DISCUSSION

- What does correctness or accuracy really mean? What accuracy do customers care about?
- How can we see how well we are doing in practice? How much feedback are customers going to give us before they leave?
- Can we estimate how good our transcriptions are? How are we doing for different customers or different topics?
- How to present results to the customers (including confidence)?
- When customers complain about poor transcriptions, how to prioritize and what to do?

- What are unacceptable mistakes and how can they be avoided? Is there a safety risk?
- Can we cope with an influx of customers?
- Will transcribing the same audio twice produce the same result? Does it matter?
- How can we debug and fix problems? How quickly?

EXAMPLES FOR DISCUSSION 2

- With more customers, transcriptions are taking longer and longer -- what can we do?
- Transcriptions sometimes crash. What to do?
- How do we achieve high availability?
- How can we see that everything is going fine and page somebody if it is not?
- We improve our entity detection model but somehow system behavior degrades... Why?
- Tensorflow update; does our infrastructure still work?
- Once somewhat successful, how to handle large amounts of data per day?
- Buy more machines or move to the cloud?

- Models are continuously improved. When to deploy? Can we roll back?
- Can we offer live transcription as an app? As a web service?
- Can we get better the longer a person talks? Should we then go back and reanalyze the beginning? Will this benefit the next upload as well?

EXAMPLES FOR DISCUSSION 3

- How many domains can be supported? Do we have the server capacity?
- How specific should domains be? Medical vs "International Conference on Allergy & Immunology"?
- How to make it easy to support new domains?

- Can we handle accents?
- Better recognition of male than female speakers?

- Can and should we learn from customer data?
- How can we debug problems on audio files we are not allowed to see?
- Any chance we might private leak customer data?
- Can competitors or bad actors attack our system?

SYLLABUS AND CLASS STRUCTURE

17-445/17-645, Fall 2020, 12 units

Monday/Wednesdays 1:30-2:50pm EDT, here on Zoom

Recitation Fridays 9:50-11:10am, EDT, on Zoom

INSTRUCTORS

Eunsuk Kang, Christian Kaestner, Vaithyanathan Narayanan

< brief introductions >

COMMUNICATION

Email to us, preferably se-ai@lists.andrew.cmu.edu

Announcements through canvas

No fixed office hours, but will stick around after lecture and recitation. Email us for extra meetings.

Welcome to ask questions publicly on Canvas.

Materials on GitHub. Pull requests encouraged!

SOFTWARE ENGINEERING CLASS

- Focused on engineering judgment
- Arguments, tradeoffs, and justification, rather than single correct answer
- "it depends..."
- Practical engagement, building systems, testing, automation
- Strong teamwork component
- Not focused on formal guarantees or machine learning fundamentals (modeling, statistics)

PREREQUISITES

Some machine-learning experience required

- Basic understanding of data science process, incl data cleaning, feature engineering, learning
- High level understand of machine-learning approaches
    - supervised learning
    - regression, decision trees, neural networks
    - accuracy, recall, precision, ROC curve
- Ideally some experience with notebooks and sklearn or other frameworks

No software-engineering knowledge required

- Basic programming skills will be useful
- Teamwork experience in product team is useful but not required
- No required exposure to requirements, software testing, software design, continuous integration, containers, process management, etc
    - if you are familiar with these, there will be some redundancy -- sorry

ACTIVE LECTURE

- Case study driven
- Discussion highly encouraged
- Contribute own experience
- Regular active in-class exercises
- In-class presentation
- Discussions over definitions

TEXTBOOK

Building Intelligent Systems: A Guide to Machine Learning Engineering

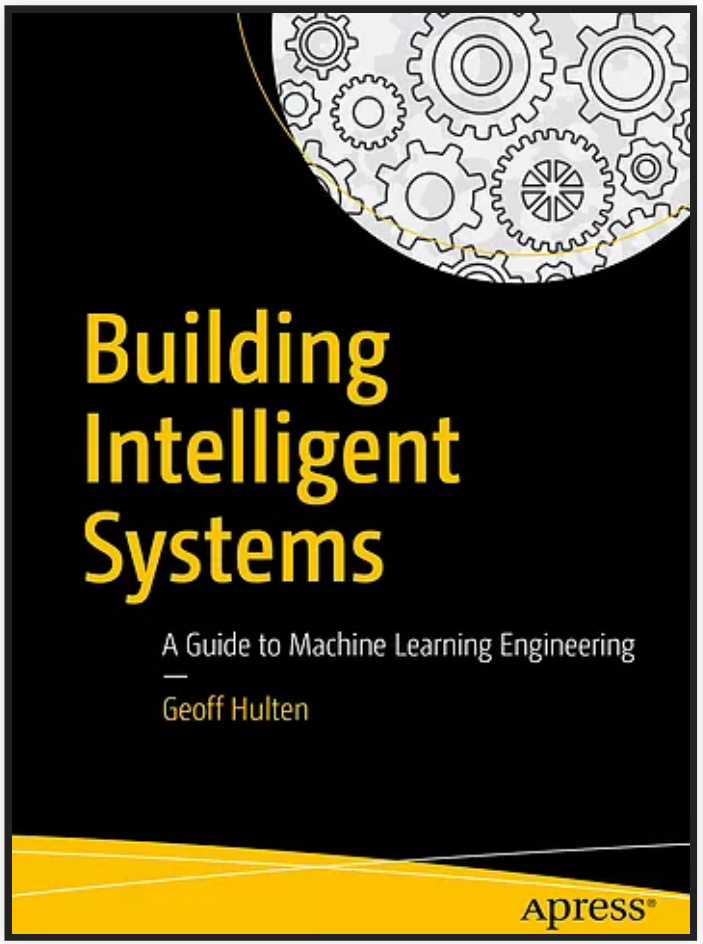
by Geoff Hulten

https://www.buildingintelligentsystems.com/

Most chapters assigned at some point in the semester

Supplemented with research articles, blog posts, videos, podcasts, ...

Electronic version in the library



READINGS AND QUIZZES

- Reading assignments for most lectures
  - Preparing in-class discussions
  - Background material, case descriptions, possibly also podcast, video, wikipedia
  - Complement with own research
- Short and easy online quizzes on readings, with partner, due before start of lecture
- Planned for: about 30-45 min for reading, 15 min for discussing and answering quiz

ASSIGNMENTS

- Series of 6 small to medium-sized individual assignments (mostly in first half)
    - engage with practical challenges
    - analyze risks, fairness
    - reason about tradeoffs and justify your decisions
    - mostly written reports, a little modeling, limited coding
    - Pandemic option: may be done with partner
- Large team project with 4 milestones (mostly in second half)
    - Build and deploy prediction service
    - Testing in production
    - Monitoring
    - Final presentation
- Due Wednesday night, see schedule.

RECITATIONS

Typically hands on exercises, use tools, analyze cases

Often designed to prepare for assignments

First recitation on Friday: remote work and collaboration + Git

GRADING

- 40% individual assignment
- 30% group project with final presentation
- 10% midterm
- 10% participation
- 10% reading quizzes
- no final exam

- expected grade cutoffs: 81-90% B, 91-100% A

GRADING PHILOSOPHY

- Specification grading, based in adult learning theory
- Giving you choices in what to work on or how to prioritize your work
- We are making every effort to be clear about expectations (specifications)
- Assignments broken down into expectations with point values, each graded pass/fail
- You should be able to tell what grade you will get for an assignment when you submit it, depending on what work you chose to do

[Example]

PARTICIPATION

- Participation is important
    - Participation in in-class discussions
    - Active participation in recitations
    - Alternative arrangements if you cannot attend classes live
- Participation != Attendance
- Grading:
    - 100%: Participates at least once in most lectures through chat or audio, or
    - 100%: Participates in 25% of lectures and actively contributes to discussions in most recitations
    - 90%: Participates at least once in over half of the lectures
    - 70%: Participates at least once in 25% of the lectures
    - 40%: Participates at least once in at least 3 lectures or recitations.
    - 0%: No participation in the entire semester.

FLEXIBILITY AND ACCOMMODATIONS

(details in syllabus)

- 7 tokens per student:
    - Submit individual assignment 1 day late for 1 token (after running out of tokens 15% penalty per late day)
    - Redo individual assignment for 3 token
    - Resubmit or submit reading quiz late for 1 token
    - Remaining tokens count toward participation

- 7 tokens per team:
    - Submit milestone 1 day late for 1 token (no late submissions accepted when out of tokens)
    - Redo milestone for 3 token
- Exceptions and accommodations on request, email us.

TEAMWORK

- Teams stay together for project throughout semester, starting in next week
- Please fill out survey after class
- Some advice in lecture + we'll help with debugging team issues
- Peer grading on all milestones (based on citizenship on team)

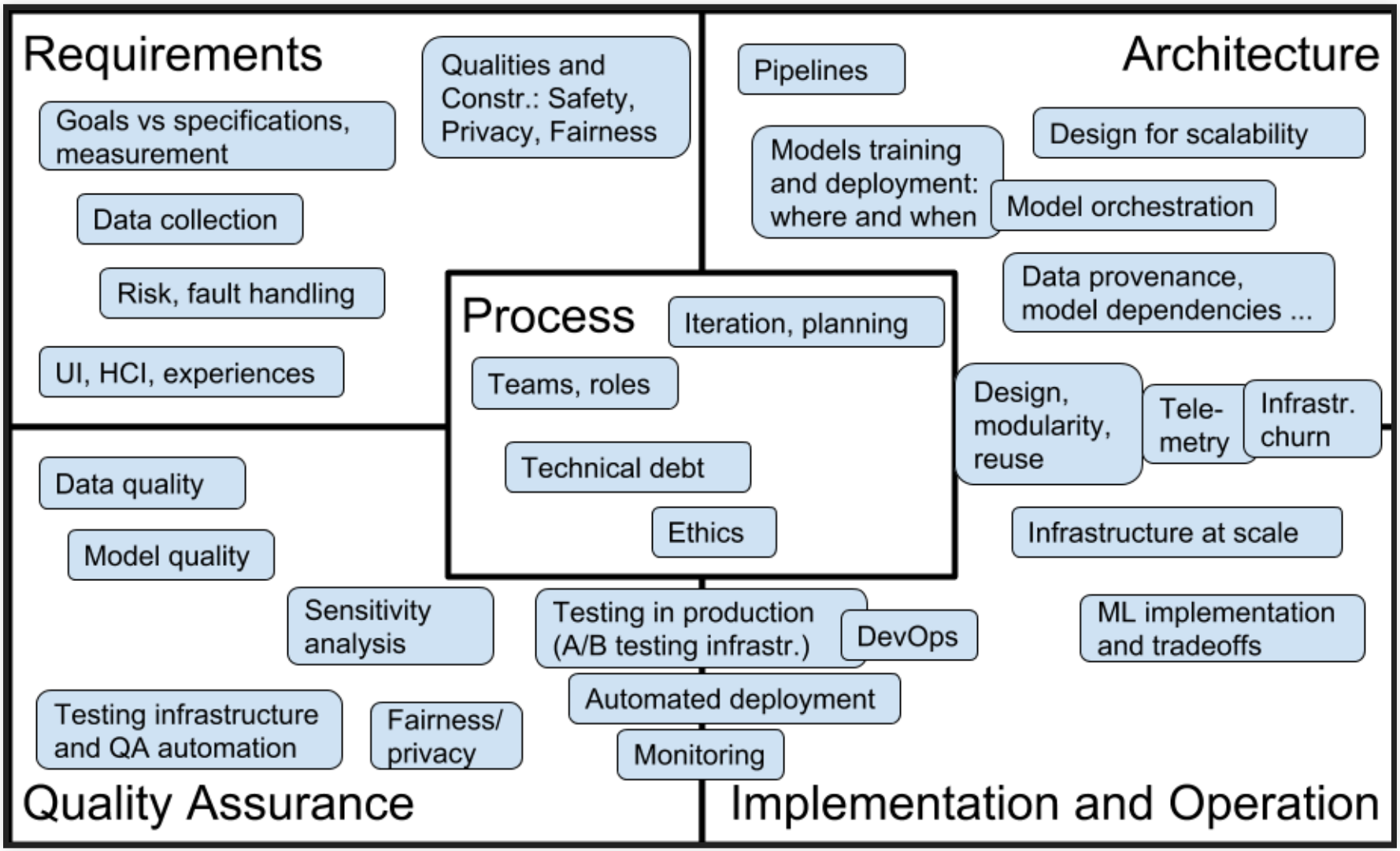
ACADEMIC HONESTY

See web page

In a nutshell: do not copy, do not lie, do not share or publicly release your solutions

In group work, be honest about contributions of team members, do not cover for others

If you feel overwhelmed or stressed, please come and talk to us (see syllabus for other support opportunities)


Requirements
Goals vs specifications, measurement
Qualities and Constr.: Safety, Privacy, Fairness
Data collection
Risk, fault handling
UI, HCI, experiences
Architecture
Pipelines
Design for scalability
Models training and deployment: where and when
Model orchestration
Data provenance, model dependencies ...
Process
Iteration, planning
Teams, roles
Technical debt
Ethics
Design, modularity, reuse
Tele-metry
Infrastr. churn
Infrastructure at scale
Data quality
Model quality
Sensitivity analysis
Testing in production (A/B testing infrastr.)
DevOps
ML implementation and tradeoffs
Automated deployment
Testing infrastructure and QA automation
Fairness/ privacy
Monitoring
Quality Assurance
Implementation and Operation

ASIDE: AI VS ML

- Artificial intelligence is an umbrella term covering symbolic AI (problem solving, reasoning) as well as machine learning (statistical learning from data)
- This course focuses mostly on *statistical machine learning* and supervised learning (extrapolating from data, inductive reasoning)
- We will cover *symbolic AI* (expert systems, probabilistic reasoning, ...) selectively, often for contrast

INTRODUCTIONS

Let's go around the "room" for introductions:

- Your (preferred name)
- In two sentences, your data science background and goals
- In two sentences, your software engineering background, if any, and goals
- One topic you are particularly interested in, if any?

CORRECTNESS AND SPECIFICATIONS

DEDUCTIVE VS. INDUCTIVE REASONING

WHO IS TO BLAME?

```
Algorithms.shortestDistance(g, "Tom", "Anne");

> ArrayOutOfBoundsException
```

```
Algorithms.shortestDistance(g, "Tom", "Anne");

> -1
```

WHO IS TO BLAME?

```
class Algorithms {
    /**
     * This method finds the shortest distance between to
     * verticies. It returns -1 if the two nodes are not
     * connected.
    */
    int shortestDistance(…) {…}
}
```

```
class Algorithms {
    /**
     * This method finds the shortest distance between to
     * verticies. Method is only supported
     * for connected verticies.
    */
    int shortestDistance(…) {…}
}
```

SYSTEM DECOMPOSITION WITH INTERFACES

```
/*@ requires amount >= 0;
    ensures balance == \old(balance)-amount &&
            \result == balance;
@*/
public int debit(int amount) {
    ...
}
```

(JML specification in Java, pre- and postconditions)

```
/**
  * Calls the <code>read(byte[], int, int)</code> overloaded [..
  * @param buf The buffer to read bytes into
  * @return The value retured from <code>in.read(byte[], int, in
  * @exception IOException If an error occurs
  */
public int read(byte[] buf) throws IOException
{
    return read(buf, 0, buf.length);
}
```

(textual specification with JavaDoc)

CONTRACTS/SPECIFICATIONS

- Contracts describe expected behavior for methods, while hiding the implementation behind
- States method's and caller's responsibilities
- Analogy: legal contract
    - If you pay me this amount on this schedule...
    - I will build the following...
    - Some contracts have remedies for nonperformance
- Invariants must hold before and after loop/method execution
- Defines what it means for implementation to be correct, including exceptional behavior

WHO IS TO BLAME?

```
Math.sqrt(-5);

> 0
```

BENEFITS OF SPECIFICATIONS

- Exact specification of what should be implemented
- Decompose a system into its parts, develop and test parts independently
- Accurate blame assignments and identification of buggy behavior
- Useful for test generation and as test oracle

SPECIFICATIONS IN MACHINE LEARNING?

```
/**
  ????
*/
String transcribe(File audioFile);
```

SPECIFICATIONS IN MACHINE LEARNING?

```
/**
  ????
*/
List<Product> suggestedPurchases(List<Product> pastPurchases);
```

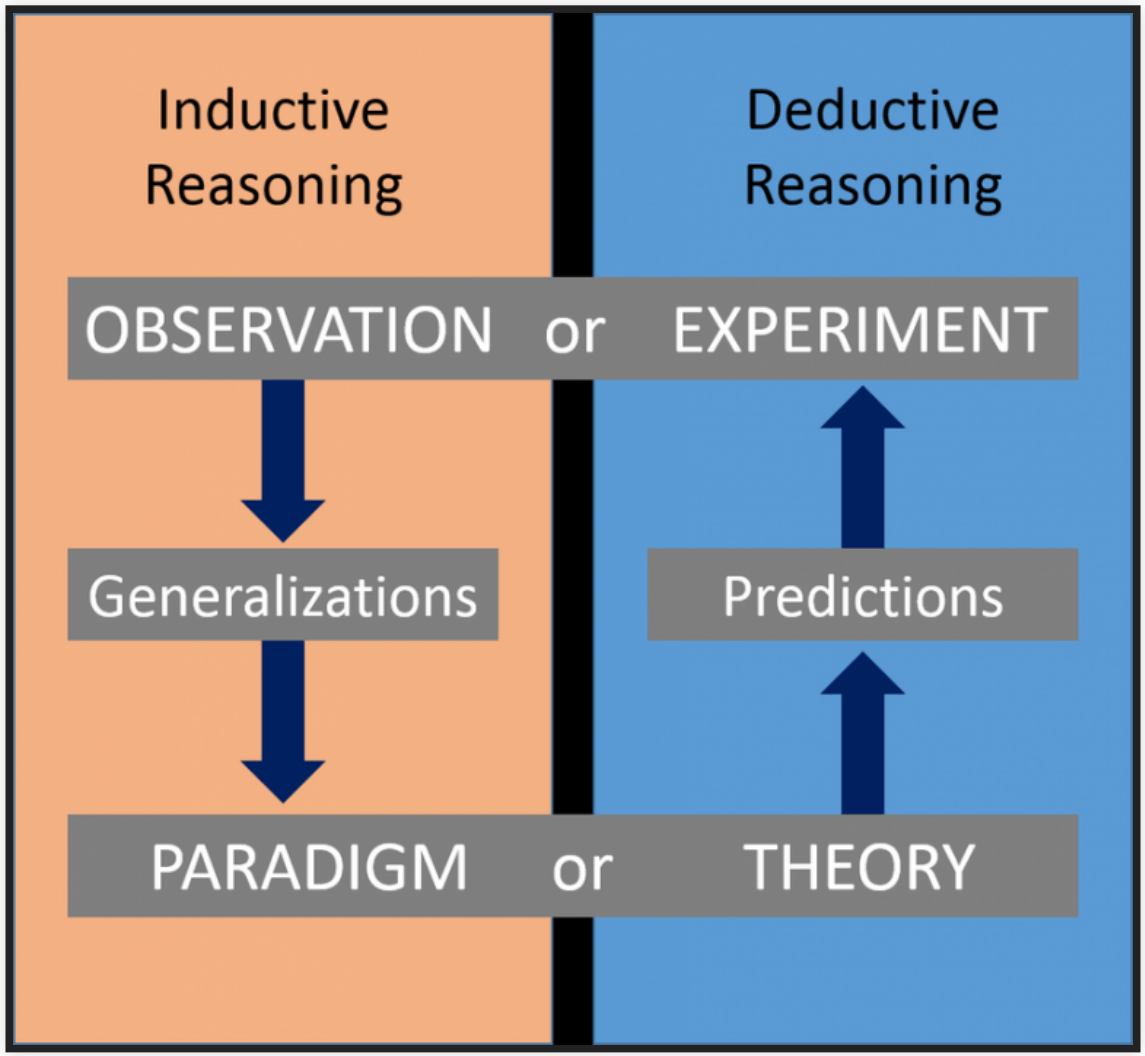
SPECIFICATIONS IN MACHINE LEARNING?

```
/**
  ????
*/
Boolean predictRecidivism(int age,
                          List<Crime> priors,
                          Gender gender,
                          int timeServed,
                          ...);
```

SPECIFICATIONS IN MACHINE LEARNING?

- Usually clear specifications do not exist -- we use machine learning exactly because we do not know the specifications
- Can define correctness for some data, but not general rules; sometimes can only determine correctness after the fact
- Learning for tasks for which we cannot write specifications
    - Too complex
    - Rules unknown
- AI will learn rules/specifications, often not in a human-readable form, but are those the right ones?

- Often *goals* used instead --> maximize a specific objective



(Daniel Miessler, CC SA 2.0)

DEDUCTIVE REASONING

- Combining logical statements following agreed upon rules to form new statements
- Proving theorems from axioms
- From general to the particular
- *mathy reasoning, eg. proof that π is irrational*

- Formal methods, classic rule-based AI systems, expert systems

INDUCTIVE REASONING

- Constructing axioms from observations
- Strong evidence suggests a rule
- From particular to the general
- *sciency reasoning, eg. finding laws of nature*

- Most modern machine learning systems, statistical learning

RESULTING SHIFT IN DESIGN THINKING?

From deductive reasoning to inductive reasoning...

From clear specifications to goals...

From guarantees to best effort...

What does this mean for software engineering?

For decomposing software systems?

For correctness of AI-enabled systems?

For safety?

For design, implementation, testing, deployment, operations?

A TOUCH OF REALISM

While it is possible to formally specify programs and prove them correct, this is rarely ever done.

In practice, specifications are often textual, local, weak, vague, or ambiguous, if they exist at all. Some informal requirements and some tests might be the only specifications available.

Software engineers have long development methods to deal with uncertainty, missing specifications, and unreliable components.

AI may raise the stakes, but the problem and solutions are not entirely new.

SURVEY TIME

Survey helps us to form teams (link on Canvas)

SUMMARY

- *Data scientists* and *software engineers* have different goals and focuses
  - Building systems requires both
  - Various qualities are relevant, beyond just accuracy
- Inductive reasoning and lack of specifications