QA: ML

Michael Hilton and Rohan Padhye

institute for SOFTWARE RESEARCH | Carnegie Mellon University School of Computer Science

Administrativia

- HW5 is out
  - HW5 Part A is based on static/dynamic analysis tools and CI
  - HW5 Part B is based on ML explainability (covered Thursday)
- HW6 is going to be the “Open Source Excursion”

Learning goals

- Understand challenges for QA of ML systems
- Be able to test assumptions about the data
- Understand and choose different fairness approaches
- Detect changes in the model over time


Disney+
Day
November 12

Disney+ Lessons learned

Disney says its new Disney+ streaming service is so popular you can't stream it

"The consumer demand for Disney+ has exceeded our highest expectations." Translation: Whoops.

By Peter Kafka | Nov 12, 2019, 12:30pm EST

SHARE

A scene from *The Mandalorian*, a new Star War series streaming on Disney+. | Lucasfilm

QA for ML

What challenges exist when trying to test ML system

What does it mean to do QA for a ML System?

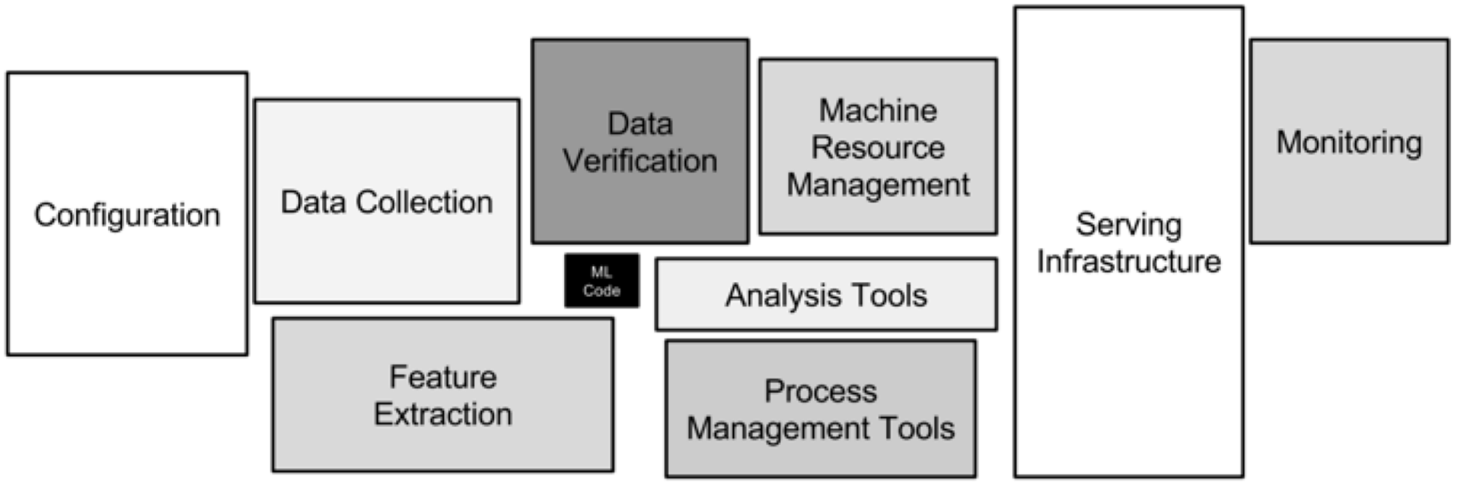


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

Broad considerations when testing ML

- Data debugging, validation, and testing
- Model debugging, validation, and testing
- Service debugging, validation, and testing
  - Traditionally testing, Design docs, already covered

Data Debugging

- Validate Input Data Using a Data Schema
  - For your feature data, understand the range and distribution. For categorical features, understand the set of possible values.
  - Encode your understanding into rules defined in the schema.
  - Test your data against the data schema.
- Test Engineered Data: For example:
  - All numeric features are scaled, for example, between 0 and 1.
  - One-hot encoded vectors only contain a single 1 and N-1 zeroes.
  - Missing data is replaced by mean or default values.
  - Data distributions after transformation conform to expectations.
  - Outliers are handled, such as by scaling or clipping.

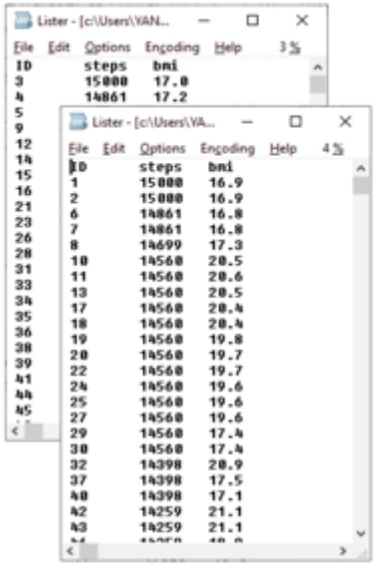
Data Debugging Cont…

- Is your data sampled in a way that represents your users (e.g., will be used for all ages, but you only have training data from senior citizens) and the real-world setting (e.g., will be used year-round, but you only have training data from the summer
- Training-serving skew—the difference between performance during training and performance during serving. During training, try to identify potential skews and work to address them. During evaluation, continue to try to get evaluation data that is as representative as possible of the deployed setting.
- Are any features in your model redundant or unnecessary? Use the simplest model possible.
- Data bias is another important consideration

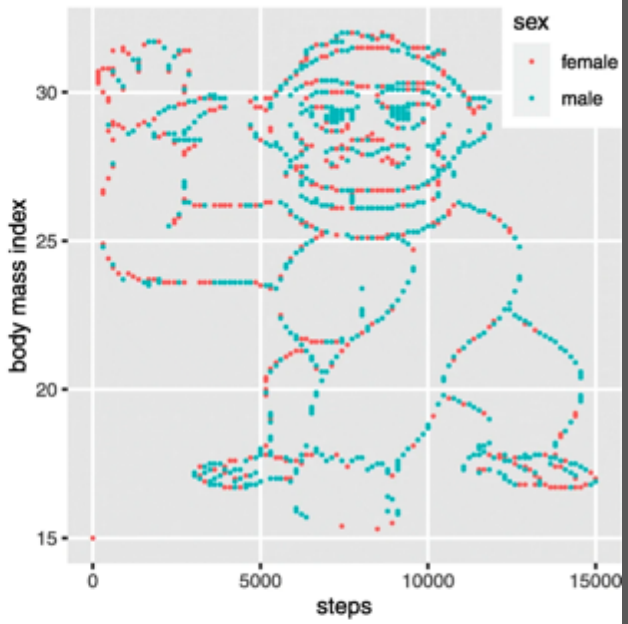
https://ai.google/responsibilities/responsible-ai-practices/

Examine your data…

a



b



c

	Gorilla not discovered	Gorilla discovered
Hypothesis-focused	14	5
Hypothesis-free	5	9

Model testing: Where do we get an oracle?

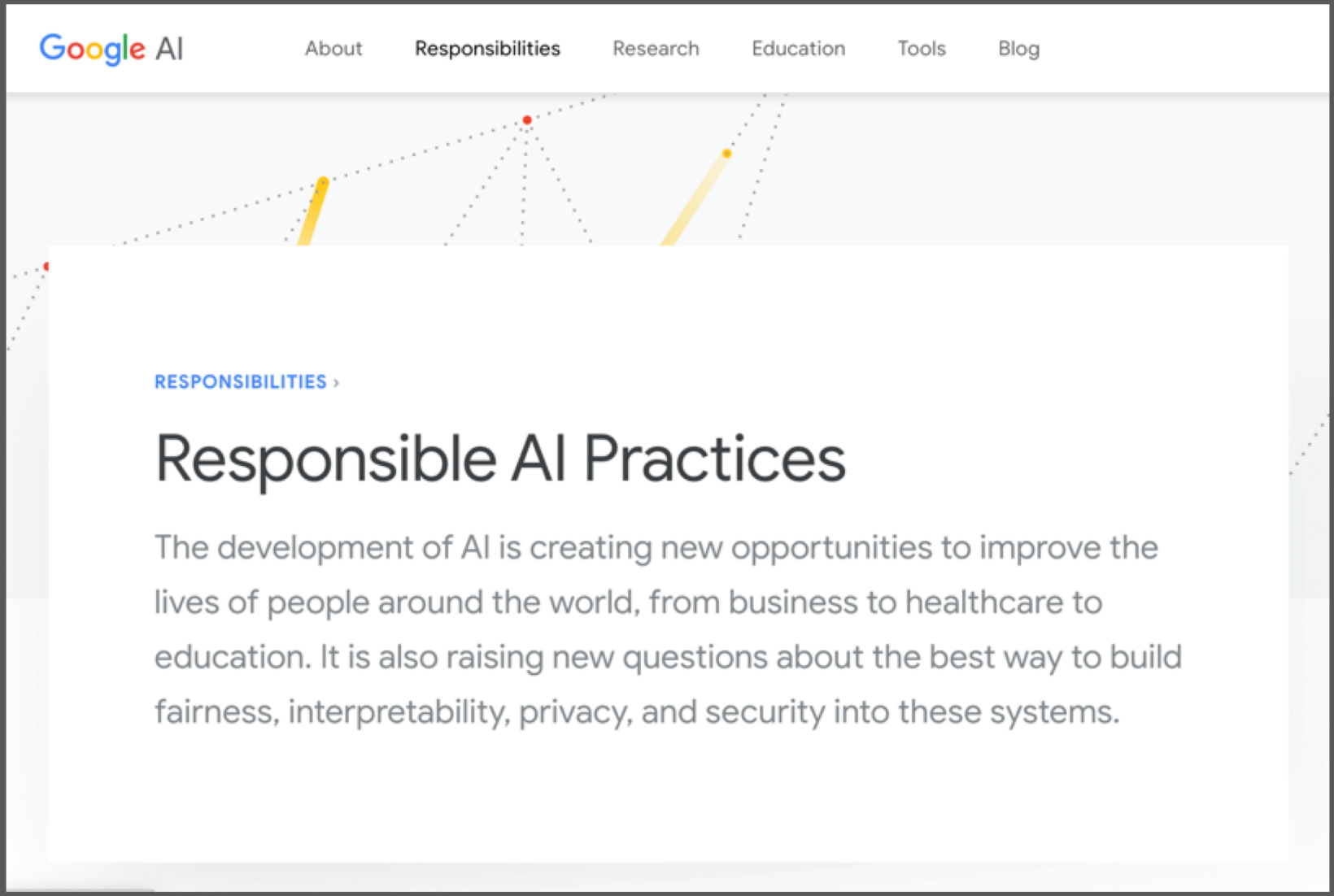
Model Debugging

- Check that the data can predict the labels.
  - Use 10 examples from your dataset that the model can easily learn from. Alternatively, use synthetic data.
- Establish a baseline
  - Use a linear model trained solely on most predictive feature
  - In classification, always predict the most common label
  - In regression, always predict the mean value

USE YOUR DATA TO TEST YOUR ML

Two approaches to consider

- Look at individual data points
  - We will look at LIME later today
- Statistically examine your entire data set
  - We will discuss next week



https://ai.google/responsibilities/responsible-ai-practices/

Recommended practices

- Use a human-centered design approach
- Identify multiple metrics to assess training and monitoring
- When possible, directly examine your raw data
- Understand the limitations of your dataset and model
- Test, Test, Test
- Continue to monitor and update the system after deployment

Use a human-centered design approach

- Design features with appropriate disclosures built-in: clarity and control is crucial to a good user experience.
- Consider augmentation and assistance: producing a single answer can be appropriate where there is a high probability that the answer satisfies a diversity of users and use cases. In other cases, it may be optimal for your system to suggest a few options to the user.
- Model potential adverse feedback early in the design process, followed by specific live testing and iteration for a small fraction of traffic before full deployment.
- Engage with a diverse set of users and use-case scenarios, and incorporate feedback before and throughout project development.

Identify multiple metrics to assess training and monitoring

- Does your data contain any mistakes (e.g., missing values, incorrect labels)?
- Is your data sampled in a way that represents your users, and the real-world setting? Is the data accurate?
- Training-serving skew—the difference between performance during training and performance during serving—is a persistent challenge.
- Are any features in your model redundant or unnecessary? Use the simplest model that meets your performance goals.
- Data bias is another important consideration; learn more in practices on AI and fairness.

Understand the limitations of your dataset and model

- A model trained to detect correlations should not be used to make causal inferences, or imply that it can. E.g., your model may learn that people who buy basketball shoes are taller on average, but this does not mean that a user who buys basketball shoes will become taller as a result.
- Machine learning models today are largely a reflection of the patterns of their training data. It is therefore important to communicate the scope and coverage of the training, hence clarifying the capability and limitations of the models. E.g., a shoe detector trained with stock photos can work best with stock photos but has limited capability when tested with user-generated cellphone photos.
- Communicate limitations to users where possible. For example, an app that uses ML to recognize specific bird species might communicate that the model was trained on a small set of images from a specific region of the world. By better educating the user, you may also improve the feedback provided from users about your feature or application.

Test, Test, Test

Learn from software engineering best test practices and quality engineering to make sure the AI system is working as intended and can be trusted.

- Conduct rigorous **unit tests** to test each component of the system in isolation.
- Conduct **integration tests** to understand how individual ML components interact with other parts of the overall system.
- Proactively detect **input drift** by testing the statistics of the inputs to the AI system to make sure they are not changing in unexpected ways.

Test, Test, Test - Continued

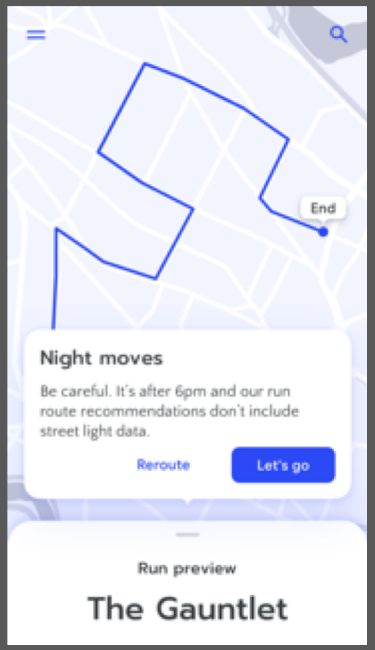
- Use a gold standard dataset to test the system and ensure that it **continues to behave as expected**. Update this test set regularly in line with changing users and use cases, and to reduce the likelihood of training on the test set.
- Conduct iterative user testing to incorporate a diverse set of users' needs in the development cycles.
- Apply the quality engineering principle of poka-yoke: build quality checks into a system, so that unintended failures either cannot happen or **trigger an immediate response** (e.g., if an important feature is unexpectedly missing, the AI system won't output a prediction).

Continue to monitor and update the system after deployment

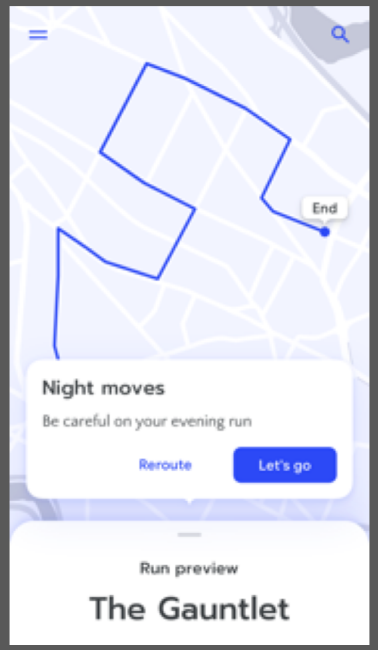
- Issues will occur: any model of the world is imperfect almost by definition. Build time into your product roadmap to allow you to address issues.
- Consider both short- and long-term solutions to issues. A simple fix may help to solve a problem quickly, but may not be the optimal solution in the long run.
- Before updating a deployed model, analyze how the candidate and deployed models differ, and how the update will affect the overall system quality and user experience.

Trust Calibration

- Trust calibration can help humans make better decision when they have a better understanding of how and why the decision was recommended.

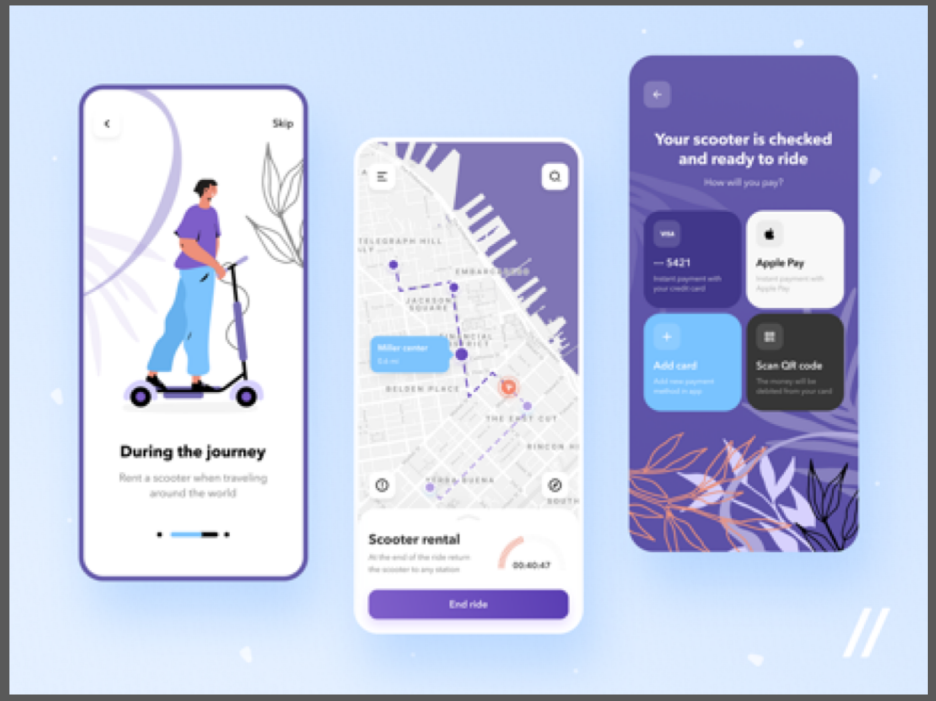


Vs



Activity

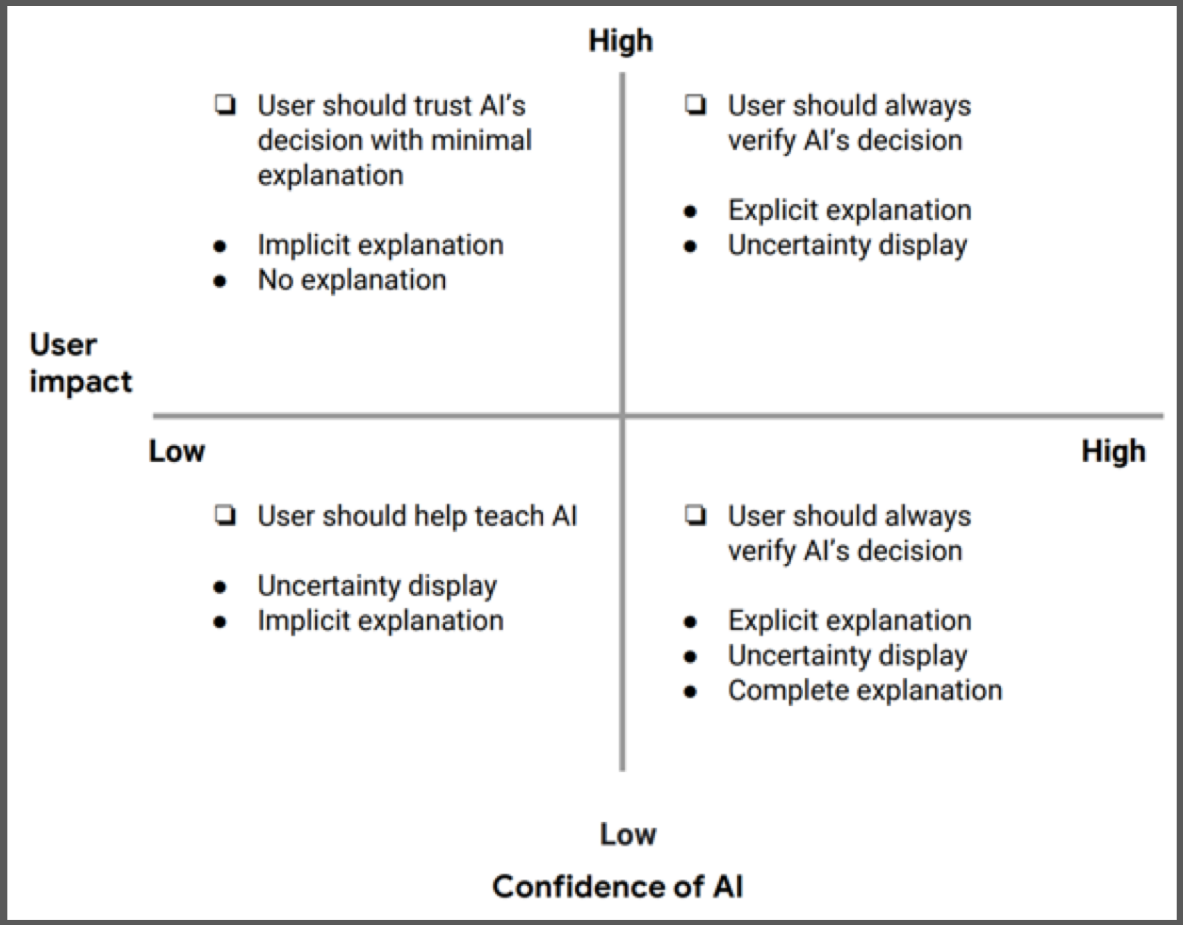
- Develop 3 candidate AI features of the scooter app
- Choose 1 feature, and describe a presentation that WOULD NOT help the user calibrate trust correctly, and a presentation that WOULD help the user calibrate trust correctly.



How to build trust

- Help users calibrate their trust
  - Articulate data sources
  - Tie explanations to user actions
  - Account for situational stakes
- Optimize for understanding
- Manage influence on user decisions

Explanation Strategy



USING DATA TO TEST ML

Activity:

- Inclass demo from: https://www.kaggle.com/vikumsw/explaining-random-forest-model-with-lime/notebook

- Working with your team, start trying to get LIME to run with your actual code (this will get you started on HW5)