Requirements and Risks
Overview

1. AI makes mistakes
2. Requirements
3. Risks
4. Risks mitigation ideas
   - Feature selection
   - Testing
   - Monitoring
5. Conclusion
AI makes mistakes

The mistakes intelligence makes:

- Aren’t necessarily intuitive.
- Aren’t the same from day to day.
- Aren’t easy to find ahead of time.
- Aren’t possible to fix with “just a little more work on intelligence.”
### Different types of mistakes

<table>
<thead>
<tr>
<th></th>
<th>The Intelligence says some is</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>There</td>
</tr>
<tr>
<td>Someone actually is</td>
<td>There</td>
</tr>
<tr>
<td></td>
<td>False positive</td>
</tr>
<tr>
<td>Not There</td>
<td>False positive</td>
</tr>
</tbody>
</table>
AI Changes

- If it learns something new, it maybe changes the result.
  - …but in which way?
- We control the way AI changes
Humanfactor

- Confusion
- Distrust
- Lack of confidence
- Fatigue
- Creep Factor
Intelligence Quality

- What makes an AI a good AI?
  - It depends on the use case
- A better Intelligence can support more forceful, frequent experiences
- Change comes from improved intelligence
Requirements Engineering

- The Process of defining the Specifications of the Software
- A lot of definitions and standards (e.g. IEEE 830)
Types of Requirements

- Functional requirements
  - What the system should do in terms of functionality
  - Input & output, response to events
- Quality (non-functional) requirements
  - How well the system delivers its functionality
  - Performance, reliability, security, safety, availability...
Specification in AI

Hard specifications are difficult

- Available Data is often small or partially describe the wanted AI model
- Accuracy is often not the requirement in an AI system (it’s quite okay if the model makes wrong predictions)
Importance of Context

Too many parameters, features, sensors and the human factor (complex problem)

• Slicing context to simple smaller problems
• Monitoring user activities
• Training AI to target smaller problems
• Continual verification
AI Models are specifications

- The AI Model is just a Specification
  - What should it do?
- If the implementation is correct but the result is wrong, it could be the wrong Model
Machine learning is like requirements engineering

Machine Learning as Requirements Engineering (Source [1])
Machine learning is like requirements engineering

Assuming AI models are specifications

- We need to identify relevant and representative data
- In both machine learning and requirements engineering, we may need to compromise user-desired functionality with laws governing privacy, fairness, or security
- When we identify that the specification does not fit, we have often gained valuable insights and in some cases learn something immediately actionable
Intelligent Experiences are Complex

- Understanding AI mistakes is not intuitive
- Unmitigated mistakes may result in system distrust
- False positives and false negatives may result in confusion

→ Lack of confidence, safety and reliability in the system
Risks Mitigation

- Complex AI systems require constant testing and monitoring

→ Large technical debt

- Traditional ways of paying debt
  1. Code refactoring
  2. Increasing coverage of unit tests
  3. Reduce dependencies
  4. Improving documentation

→ Code and system-level complexity for ML
Data and Feature Selection

1. Identify expected features from the training data in schemas with help of statistics for future analysis
2. Understanding the value of every feature
3. Reduce the feature set to the features that add the most predictive benefit
4. The training data has appropriate privacy controls
Testing

1. Version control for the model code and training data
2. Use A/B testing to understand online vs. offline result evaluation and model’s staleness
3. Use data slicing to understand model quality
4. Enforce ML fairness
5. Implement unit test for API usage
6. Implement assertions to verify the ML’s algorithmic sub-computations
Monitoring

It’s important verify that the ML system is working over time

→ In case of an incident that ML system code can be rolled back

How?
1. Notify on dependency changes
2. Log traffic and label data of new evolving features for future analysis
3. Measure statistical bias in predictions to monitor prediction quality
Conclusion

- AI makes different and unpredictable mistakes
- Understanding AI mistakes isn’t intuitive
- AI changes over time
- Understanding context is a complex task
- Requirements engineering produces large amount of technical debt
- Possible mitigations:
  1. Data and Feature Selection
  2. Testing
  3. Monitoring
Questions
THANK YOU FOR YOUR ATTENTION!
References


Acknowledgements & License

- The Images either come from the authors of these slides or the sources are linked.
- These slides are made available by the authors (Dani El-Soufi, Benjamin Fleischmann) under CC BY 4.0.