Software Engineering for AI-Enabled Systems

SOFTWARE SYSTEME

UNIVERSITÄT LEIPZIG

Prof. Dr.-Ing. Norbert Siegmund
Software Systems

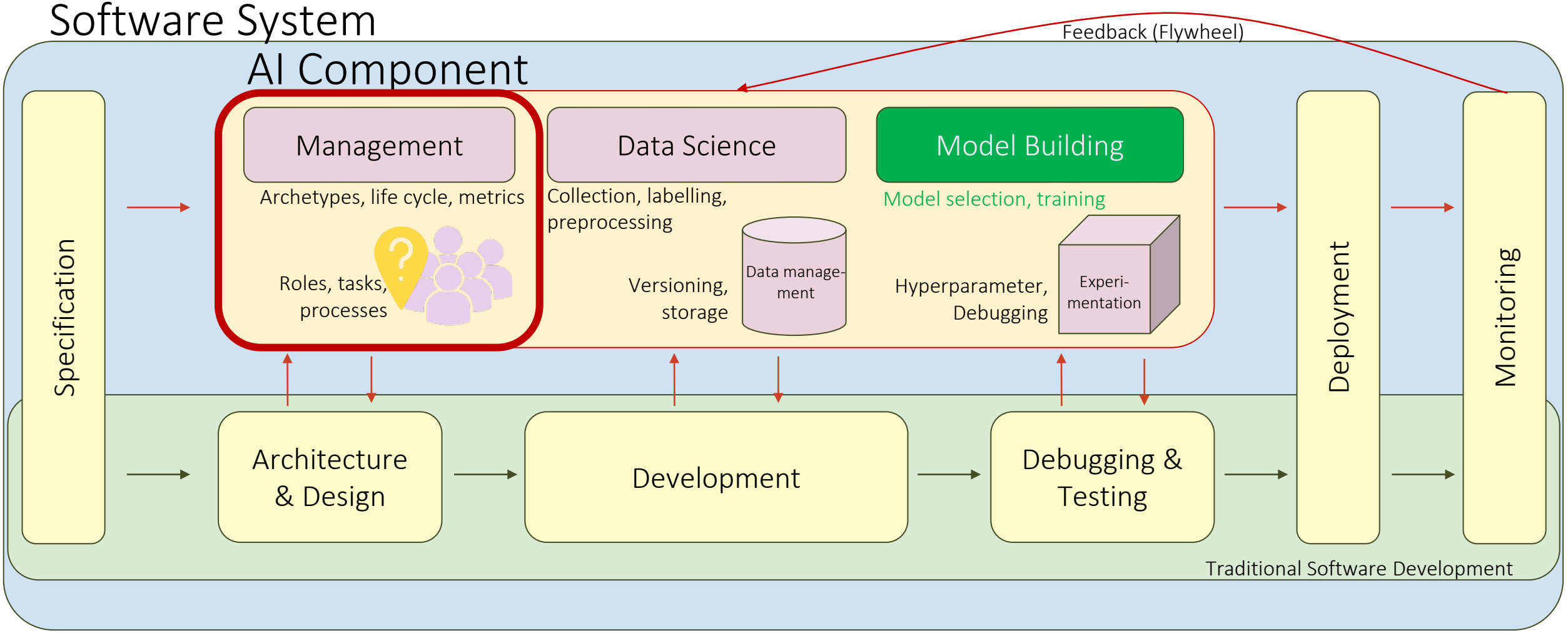
Topic I:
Why Project Management

Project Management of AI Systems

TL;DR:

- Scoping the project
- Setting the goal of the project, product, or system to be developed
- Roles & team management
- Project life cycle

Software System



Project management

- What AI project archetypes and ML model types exist to realize a business goal?
- How to start an AI/ML project based on requirements?
- How to select suitable metrics to optimize for?
- How to build teams and organize your company/project to productionize the AI/ML model?

Why Project Management?

Venturebeat reported: 87% of ML projects fail*1

Gartner reported: 53% of ML projects do not make it from prototype to production*2

Causes:

- Unclear project goal
- Misaligned ML and project goal (model metrics vs. product metrics)
- Separate and uncoordinated work between data scientists and software engineers
- Unclear interfaces in terms of team responsibilities, documentation, and technical level
- Cost is significant in terms of hardware and expert salaries
- Data management, collection, and quality ensurance requires specific management skills
- Higher risk of failing due to uncertain outcomes of the ML model
- Complexity of the project requires different skills (SE, DevOps, MLOps, DataScience, Data Engineers,…)

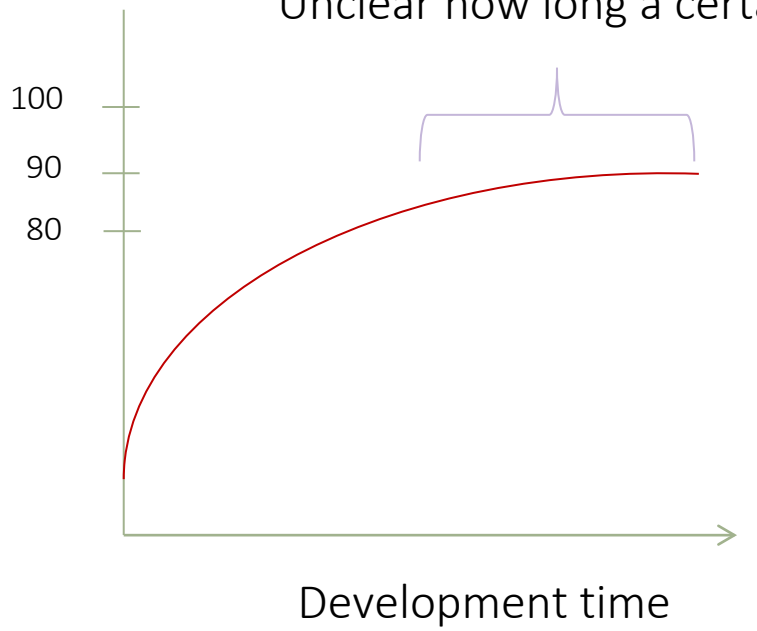
*1:https://venturebeat.com/2019/07/19/why-do-87-of-data-science-projects-never-make-it-into-production/

*2:https://www.gartner.de/de/newsroom/pressemitteilungen/2020-10-19-gartner-identifiziert-die-wichtigsten-strategischen-technologietrends-fuer-2021

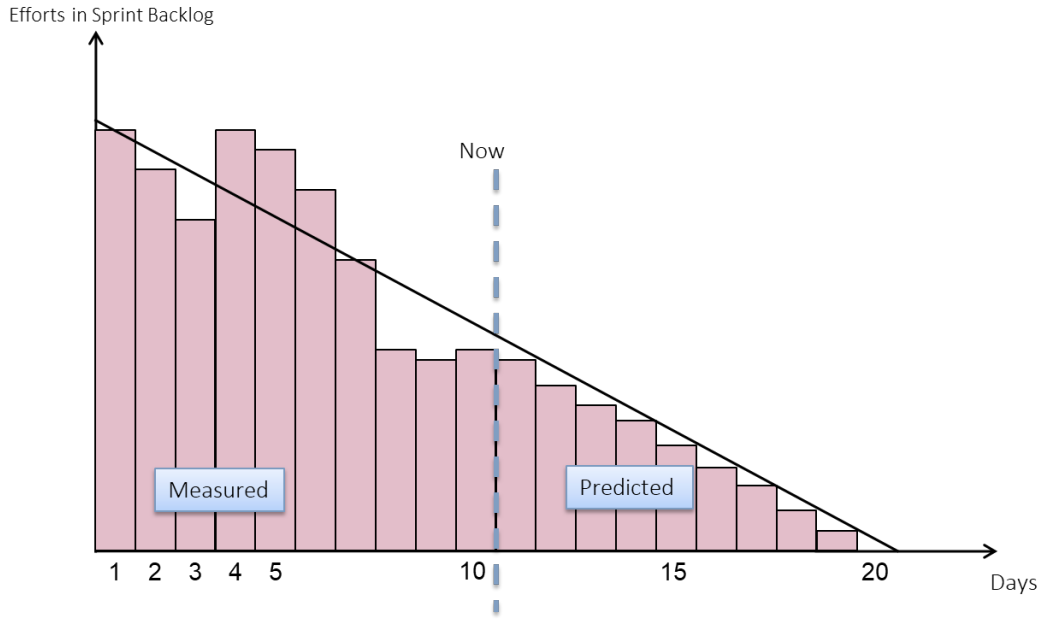
Uncertain Progress => Difficult Planning

Accuracy in %

Unclear how long a certain improve in a metric will take



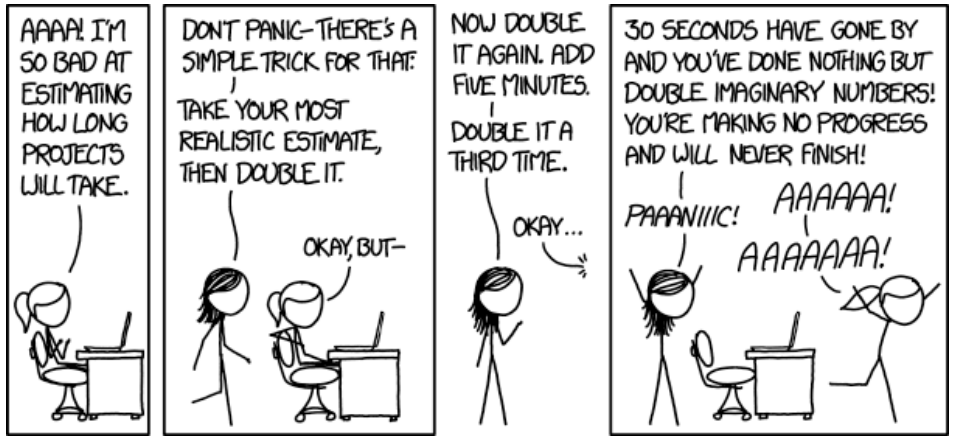
VS.



Missing Experience of ML Projects

Software systems have been developed in increasing complexity for 30 – 50 years.

ML projects exist in this extent for only 3-5 years.



ML projects need references from former projects to be able to make realistic resource predictions.

Hofstadter's law

From Wikipedia, the free encyclopedia

Hofstadter's law is a self-referential adage, coined by Douglas Hofstadter in his book *Gödel, Escher, Bach: An Eternal Golden Braid* (1979) to describe the widely experienced difficulty of accurately estimating the time it will take to complete tasks of substantial complexity:[1][2]

> Hofstadter's Law: It always takes longer than you expect, even when you take into account Hofstadter's Law.[2]

The law is often cited by programmers in discussions of techniques to improve productivity, such as *The Mythical Man-Month* or extreme programming.[3]

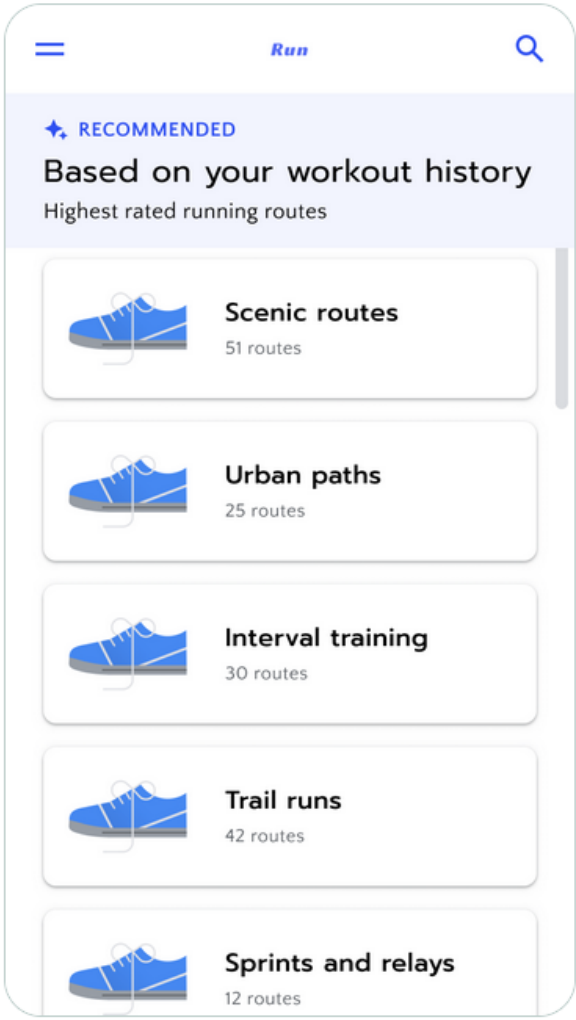
Do you need really AI?

AI scenarios:

- Recommend personalized suggestion
- Predict future events
- NLP, vision (recognition, labelling)
- Detection of low occurrence events that change of time
- Agent or bot mechanism (chat system, etc.)
- Dynamic content production

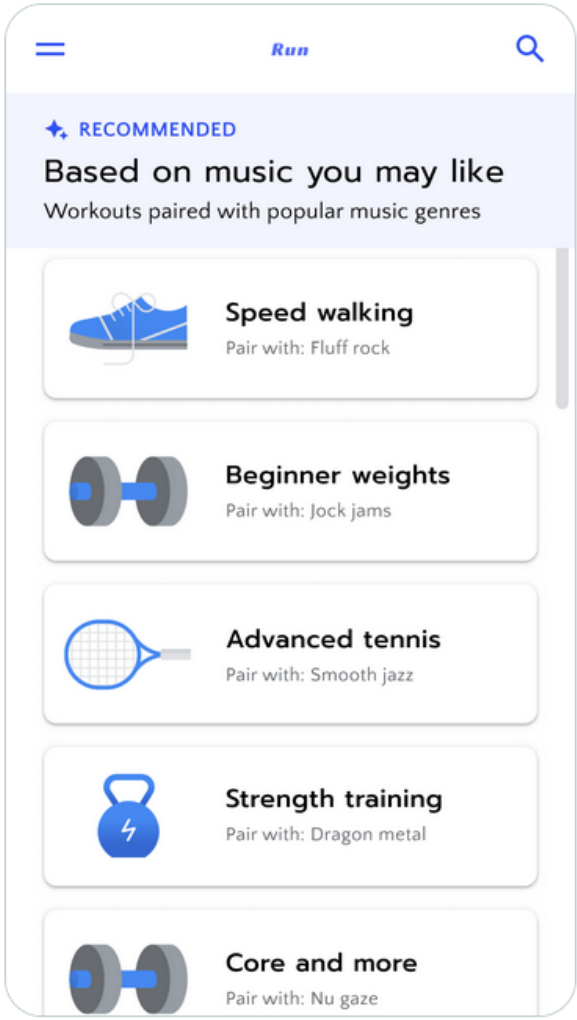
Non-AI scenarios:

- Predictability of functionality (in all contexts)
- Provide static or limited content
- Errors are substantially more costly than improvements
- Explainability of decisions
- Speed of development (better start with heuristics)
- People's desire to not automate a task



Aim for

Use AI when the predictive system can create a valuable personalized experience that couldn't exist without it.



Avoid

Don't use AI just because you can. Heuristics or manual control can often create better experiences. Here, using music preferences to suggest workouts will likely lead to a worse experience than letting people manually choose workouts.

To Automate or To Augment?

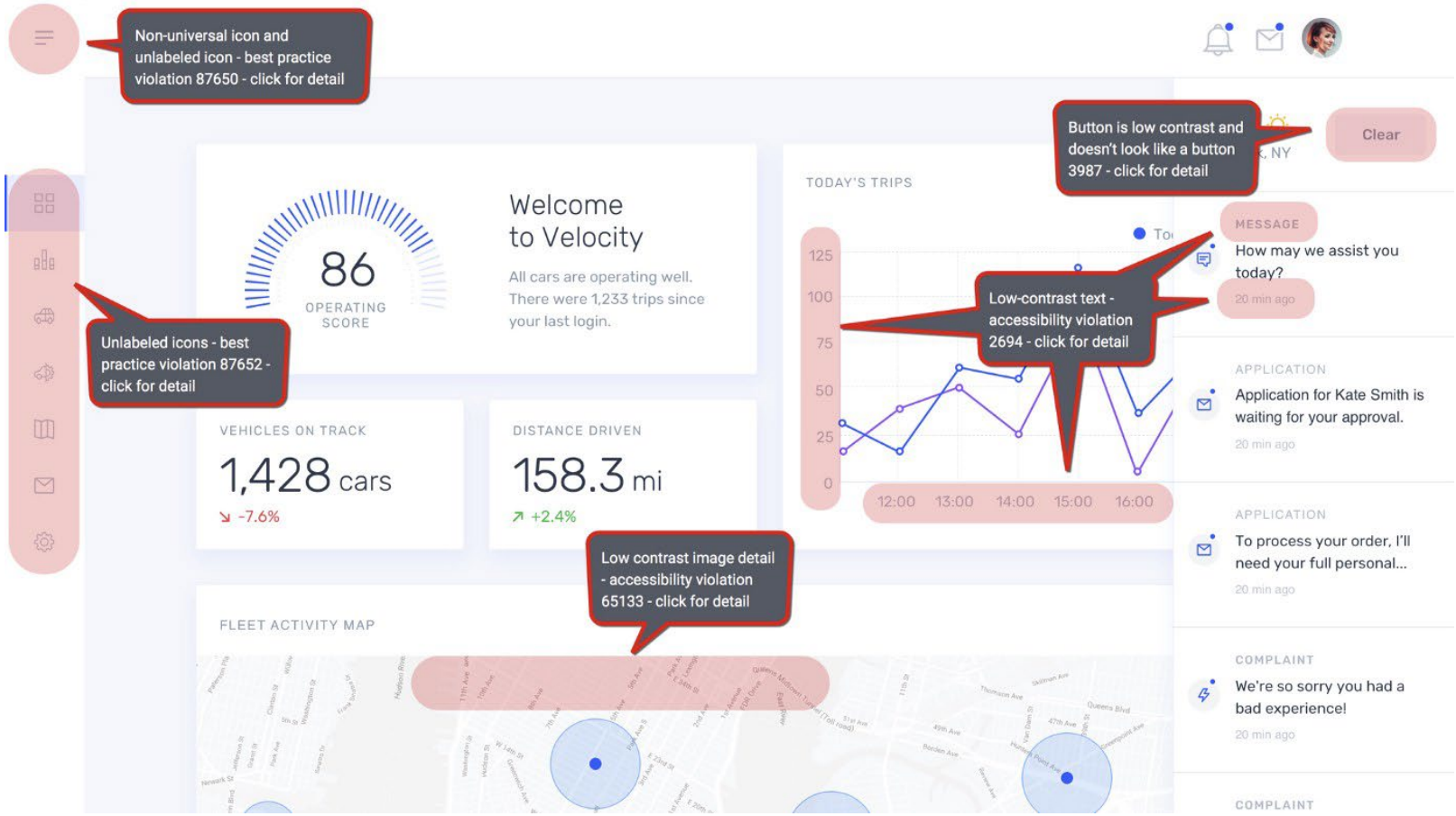
Automate if

- Increased efficiency
- Improved human safety
- Reduction of tedious tasks
- Enabling new experiences that weren’t possible without automation
- Enabling functionality that people lack the ability to do theirself

Augment if

- Increased user enjoyment of a task
- Higher levels of user control over automation
- Greater user responsibility and fulfillment
- Increased ability for the user to scale their efforts
- Increased creativity

Augmentation Example



Topic II:
Project Goal & Scope


Francesco Pochetti
@Fra_Pochetti
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.
7:08 PM · Mar 12, 2022 · Twitter Web App
291 Retweets 37 Quote Tweets 1,760 Likes

Life Cycle of ML Projects

Probably, the most important part!

Scoping & Planning Define project goals; specify requirements; allocate resources

Data unavailable; too costly or difficult to obtain; legal issues
Other (similar) tasks are easier to label
Too few or too much data requires different amount of resources

Data Define and collect data; preprocess, label, and organize data

Life Cycle of ML Projects

Scoping & Planning Define project goals; specify requirements; allocate resources

Infeasible task; inconsistent requirements; resource rebudgeting

Data Define and collect data; preprocess, label, and organize data

More data required; labelling unreliable; bias in data; validity threats

Model Model selection, training, experimentation, error analysis, debugging

Life Cycle of ML Projects

Scoping & Planning Define project goals; specify requirements; allocate resources

Model metric does not align with project goal; performance unsatisfiable/disappointing; requirements revisiting

Data Define and collect data; preprocess, label, and organize data

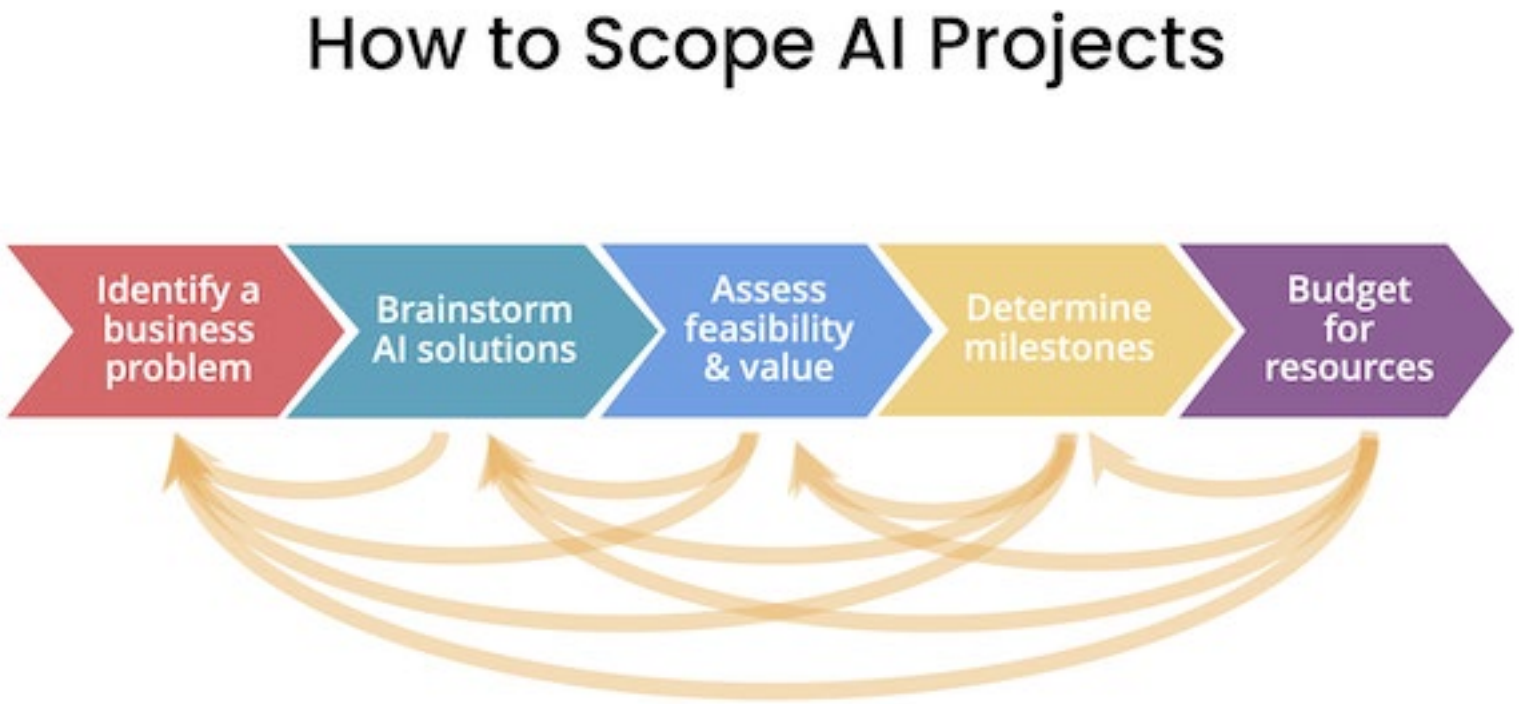
Data mismatch (different distribution in production); rare cases required; bias found

Model Model selection, training, experimentation, error analysis, debugging

Production performance != training performance; non-functional properties not met;

Deployment Deploy model, monitor and maintain system, feedback loop

Scoping & Goal definition



Source: DeepLearning.AI

Expectations project on goals may diverge

Economical:

- Make profit by selling more (e.g., via recommender system)
- Save expense (e.g., by automated flaw detection or human replacement)

Social:

- Help users in their tasks
- Increase satisfaction

Technological:

- New technological usage (e.g., be future proof)
- Accuracy of AI and joy of learning

Different objectives come from different expectations

Organisational: profit,revenue, trees saved, benefits to society, etc.

- Hard to relate objective to AI
- Affected by multiple aspects and not just AI
- Effects are not directly measurable

Futuristic: customer satisfaction (i.e., sentiment) and engagement (i.e., binding)

- May positively affect organisational objectives
- Important for AI in social networks
- But, hard to measure its success, hard to see effects of small changes, not timely

System/Product/Project Goal

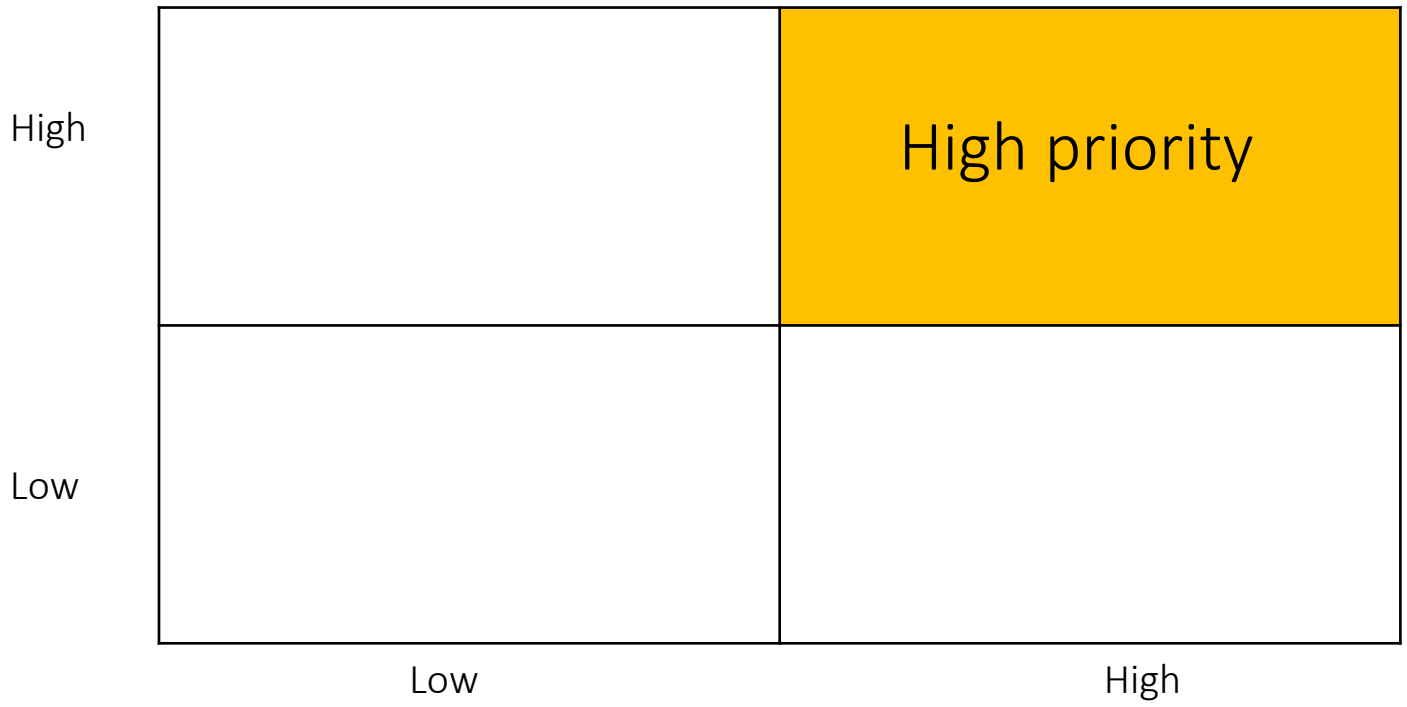
Criteria for success:

(1) Define desired outcomes: Communicate what is we want to achieve
(2) Achievable: Have a plan how to reach the goal
(3) Measurable: Know how to measure your success (evaluation criteria)

Example: Credit card fraud. Success the more unauthorized withdrawals we block

- Achievable? Yes
- Measurable? Yes
- Desired outcome? -> Let's discuss

Prioritize Projects



Impact
Value of AI solution
See Feasibility & Value

What Product Goal?

Goal: Make customers certain that a hair color looks good on them in order to sell more.

Goal: Improve engagement with app to spend more time with it in order to buy a monthly subscription.

Does it work though?: Engagement / personalization / recommendations / etc. can people make them to spend more money or solve their problems faster and longer so spend less money.

Buttom Line: Ultimate ML Goal

In a business context, the ultimate goal of ML is to **increase profits** directly (e.g., personalized ads, better recommendations, improved risk management for credits and insurances) or indirectly (e.g., improved customer satisfaction via automated help systems, improved engagement via better article selection for timeline, brand improvement via new technologies -> see Tesla)

In a non-profit context, the ultimate goal is to **improve society** either via climate protection, improvement of social factors, education, or public health.

Be certain to understand your business model and how AI/ML contributes to it. Have a clear vision not only *what* to reach for, but also how to *measure* the success of AI/ML!

Brainstorm solutions (no development, no coding, just thinking…)

Complexity, data needs, implementation effort

Do not start with an ML model!

Try the simplest thing that can satisfy the product needs

Simple things:

- Ask experts and users involved in the problem (maybe there is none?)
- Create a simple baseline via heuristics (maybe addressing 80% of the issues is enough)

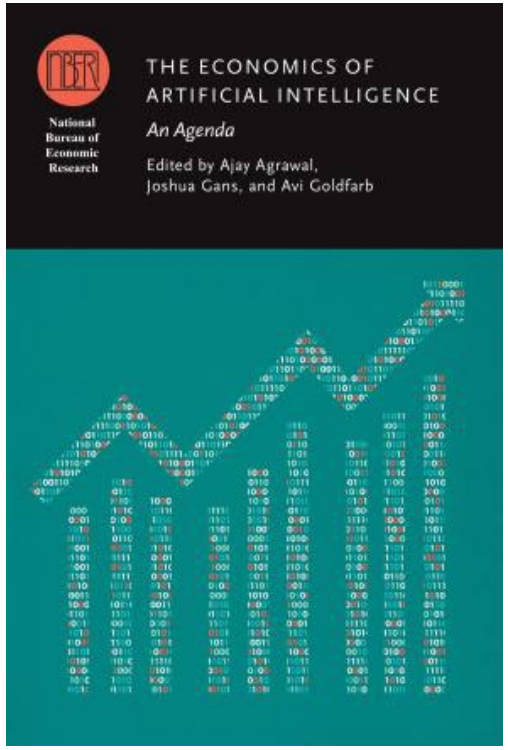
Consider more complex models step by step

- Use bullet proof simple models (linear regression, random forest, etc.) with high interpretability
- Review capabilities of existing more complex models (deep learning)

Feasibility and Value

Find problems where cheap predictions can have a large value / impact

Find complicated processes that can benefit from automation

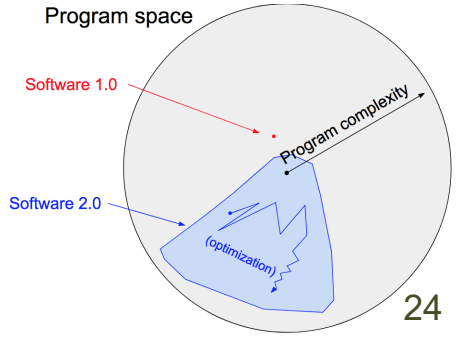


AI reduces cost of prediction for decision making (business analytics)

Cheap predictions enable

- Predictions everywhere
- For problems where predictions were too costly before

Domains with complicated rules / instructions to solve a problem may benefit from „Software 2.0“ (i.e., ML models figuring out the rules)



Feasibility & Value

Business value != model performance

Find a metric that can judge / measure the success of your product goal:

- Usually, independent of the model metric
- Collected via logging in the software (e.g., usages of a feature, click-through rate of recommendations)
- Only metric that matters; can be an aggregate of multiple metrics
- Guardrail metrics ensure that a certain threshold is reached or maintained

Model metrics measure success when the product is not yet deployed (target for optimization)

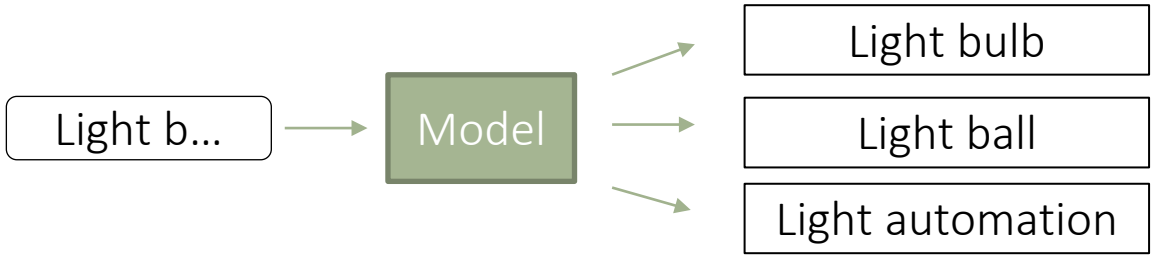
- Should be correlated to product metric & goal

Feasibility & Value

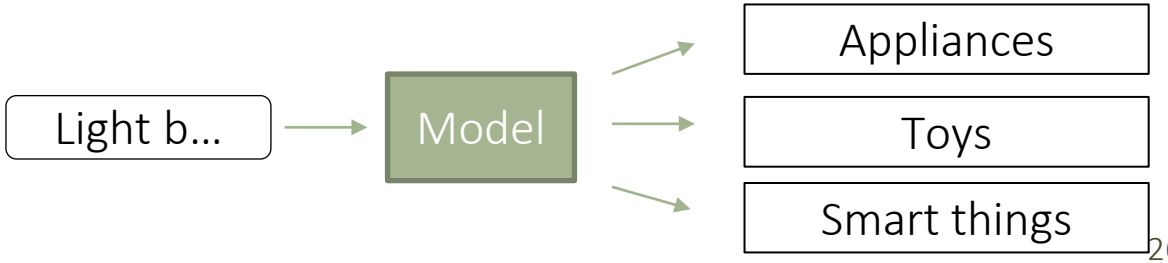
Choosing a feasible model and metric requires an understanding how the model interacts with the software

Example: Product recommendation

Variant 1: Search query completion
Idea: Guessing the chars and words of the user and complete the search string to support her search
Metric: Accuracy of all characters being generated compared to the actual user query
Feasibility: Model must be highly accurate as a single mistake produces non-sensible queries



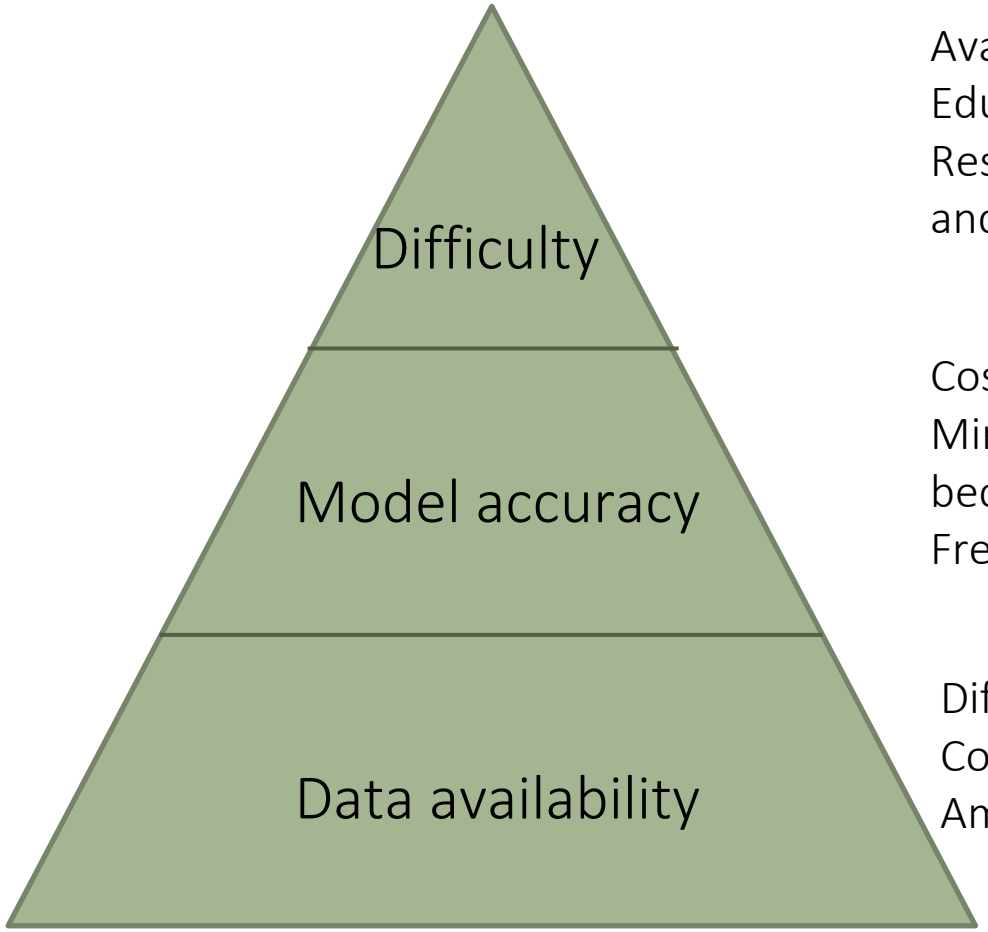
Variant 2: Topic / category suggestion
Idea: Classify user input into envisioned categories and suggest most likely category
Metric: Accuracy of suggesting the category the searched product is in
Feasibility: Model must predict only the category rather than all English words correctly. Click-through rate is easy to measure when user clicks on the category



Feasibility Pyramid for Cost Drivers

by FullStackDeepLearning

Cost drivers



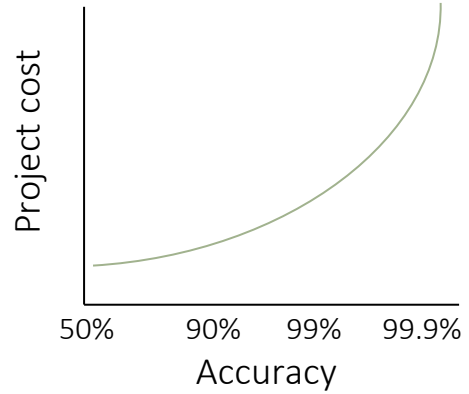
Availability of published (similar) solutions
Education requirements for the staff
Resource requirements for training (#GPUs) and deployment

Cost of wrong predictions
Minimum accuracy requirement to become useful
Frequency of required retraining

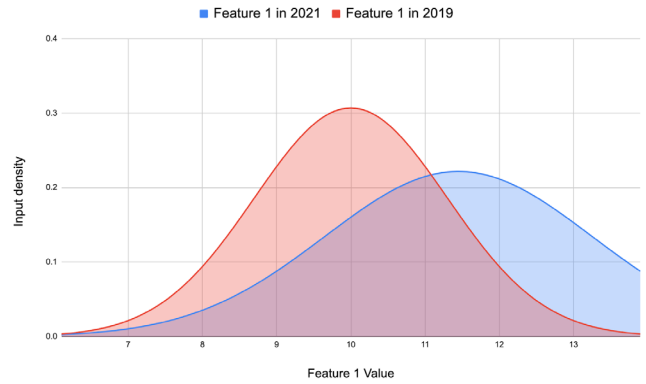
Difficulty of acquiring data
Costs of labelling data
Amount of required data

Difficult problems:

- Complex input space
- Multi-media data
- Ambiguous data
- Generalizations, causality
- World model problems
- Robustness & high precision tasks
- Explainability tasks



Feasibility & Value



Consider data drifts and the availability of labeled data when judging feasibility

- Assert the product domain about how fast data gets outdated; if correlation between input and output breaks then the model needs to be retrained once in a while (is this still feasible or economical?)
- Assert which data population is out there and whether you have access to all aspects, such as gender, ethics, dialects, product categories, opinions, weather (is this legally and ethnically correct and feasible?)

Consider the complexity and time it takes to produce an output

- How much does a single inference cost and how many do we expect ? (is the value higher than the cost?
- Is the inference time in line with our product goal (ms vs. hours for inference)?

Feasibility & Value

Techniques for improving feasibility:

- Consider safeguards in the interface with the model (e.g., show suggestions only when a certain confidence threshold is reached)
- Do not rely solely on one model (e.g., use suggestions from other sources, such as heuristics --- favorite product of today --- to increase value)
- Communicate to users of an experimental state or provide means for feedback (e.g., use up-/down votes, comments, etc.)
- Do not design a product around a model that needs to be 100% accurate all the time (e.g., consider the likelihood of being wrong and the consequences of errors to the product value and incorporate this in a proper metric)

Manage and Mitigate Risk

Risk Register

Project name: Common project risks

ID	Date raised	Risk description	Likelihood of the risk occuring	Impact if the risk occurs	Severity *Rating based on impact & likelihood.*	Owner *Person who will manage the risk.*	Mitigating action *Actions to mitigate the risk e.g. reduce the likelihood.*
1	[enter date]	Project purpose and need is not well-defined.	Medium	High	High	Project Sponsor	Complete a business case if not already provided and ensure purpose is well defined on Project Charter and PID.
2	[enter date]	Project design and deliverable definition is incomplete.	Low	High	High	Project Sponsor	Define the scope in detail via design workshops with input from subject matter experts.
3	[enter date]	Project schedule is not clearly defined or understood	Low	Medium	Medium	Project Manager	Hold scheduling workshops with the project team so they understand the plan and likelihood fo missed tasks is reduced.

Project management contains risk management:

1.Identify -> 2. Mitigate -> 3. Monitor

Scorecards are simple mechanism for identifying, managing, and reporting risk (suggested and used by Amazon AWS: Managing Machine Learning Projects)

At project start: discuss each line in the cards with the project managers and involved roles

- Issues not apply (not valid) must be reasoned for why (and documented)
- Issues that apply must contain an impact analysis together with possible actions for mitigations (e.g., differentiate between minor and major risk and how to solve it)

Significant risk must be propagated to higher authorities within the organization (e.g., legal or ethical issues), such that responsible groups can judge and handle them

Issues become tasks in the project plan and are handled accordingly

Scorecards: Project Context

Social, business, and regulatory environment of project

Always: Revise and update cards at project progress (e.g., review rounds)

Category	Requirement or Risk	Issue for project? (Y/N)	Status (Red/ Yellow/ Green)	Comments: Applicability, Status, Mitigations
Ethics				
Model makes *consequential decisions*				
Privacy				
Fairness, Bias				
Risk of bad press				
Need for transparency & auditability				
Applicability/success of ML for this application				
Closed-world development/testing vs open-world deployment				
Impact of ML model inferences				

Scorecards: Finance

Cost and benefits of the problem to be solved by AI/ML, including the surroundin software system

Category	Requirement or Risk	Issue for project? (Y/N)	Status (Red/ Yellow/ Green)	Comments: Applicability, Status, Mitigations
Financial model built				
Potential upside return				
Potential downside risk				
Worst-case downside				
Liability				
Cost of building model				
Cost of maintaining model				
Quality of model predictions vs expectations				
Uncertainty in model predictions				

Scorecards: Project Processes

Gives an overview over the whole process and avoids overlooking certain areas in the project

Category	Requirement or Risk	Issue for project? (Y/N)	Status (Red/ Yellow/ Green)	Comments: Applicability, Status, Mitigations
Research or development				
Project team skills & availability				
Project timelines				
Tests for bias applied				
Long tail analysis performed				
Statistical analysis validated				
Economic analysis of model metrics				
Team *temperature check*				
Verification and validation procedures completed				
Integration with existing processes & systems				
Instrumentation & monitoring in place				
Production model revisions				

Scorecards: Data Quality

Frequent problematic ML areas that can mislead the project into false assumptions (is there actually enough information?, false interpretation?, false assumptions?)

Category	Requirement or Risk	Issue for project? (Y/N)	Status (Red/ Yellow/ Green)	Comments: Applicability, Status, Mitigations
Input data precision				
Input data accuracy				
Data volumes & duration				
Data sources & pre-processing validated				
Production vs model data pipeline				
Data change over time: processes considered				

Scorecards: Summary

Scorecard	Status	Summary
1. Project context		
2. Financial		
3. Project processes		
4. Data quality assessment		

Summarizes key results of the individual scorecards

Scorecards can and should be customized to the organization, team, domain, and project at hand.

- Countries have different regulations and moral standards
- Business risks depend on the domain of the AI-enabled software system
- Personal and management risks depend on the people and processes involved

Example Scorecards from Amazon AWS

Project Context

Category	Example
Ethics	Weapons targeting systems Predictive policing[20] AI imitating humans[21]
Model makes *consequential decisions*	Denying people entry to country, or loans Criminal risk assessments for arrests, bail, sentencing
Privacy	HIPAA / GDPR applies
Fairness, Bias[22]	Race identified as loan risk factor Displaced jobs disproportionately held by minorities
Risk of bad press	Photo labeling app labels Self-driving car kills pedestrian[23]
Need for transparency & auditability	Recommendations must be independently verifiable[24]
Applicability/ success of ML for this application	Natural language assistant chat bots vs free-form conversational understanding
Closed-world development/ testing vs open-world deployment	Robot in lab vs open house environment (children, pets, stairs)
Impact of ML model inferences	Self-driving car *sees* pedestrian but ignores it as false-positive

Financial

Category	Example
Financial model built	Model with anticipated ROI available for review
Potential upside return	Increased customer retention of 5% Decreased cost per transaction of 5%
Potential downside risk	Decreased customer retention (10%) Increased cost per transaction (5%)
Worst-case downside	Automated trading algorithm causes Great Financial Crash
Liability	Self-driving car kills pedestrian
Cost of building model	6 months, team of 2[25]
Cost of maintaining model	Ongoing, 10 hours/month
Quality of model predictions vs expectations	Economic model assumes 100% correct predictions, but results 85% correct
Uncertainty in model predictions	Prediction might be accurate +/– 10% Extreme data points cause bad predictions

Data Quality

Category	Example
Input data precision[26]	Test & production data have same characteristics; outliers discarded for both model & production
Input data accuracy	Sensor values estimated to be +- 5% of actual
Data volumes & duration	Model data only available for 3 months (but business cycle is 1 year) 50% of source #3 data discarded
Data sources & pre-processing validated	Data source #1 now undergoing additional quality checks Data extract #2 discovered to be flawed; re-training required
Production vs model data pipeline	Prod inferences will use separate data source than model trained on
Data change over time: processes considered	Upstream system changes logic & meaning of its input to model[27]

Processes

Category	Example
Research or Development	Research, leading to development if successful
Project Team Skills & Availability	Team is missing production application development skills
Project Timelines	Committed timelines assume data is available, appropriate, and sufficient for model
Tests for bias applied	Recidivism rates the same across all populations
Long tail analysis performed	Minority groups disadvantaged because system is trained on majority
Statistical analysis validated[28]	Incorrect statistical analysis shows correlation where none exists, leading to incorrect inferences
Economic analysis of model metrics	Results of model are still in range of project economic value estimates
Team *temperature check*[29]	Team *gut check*: team willing to be a customer of the system
Verification and validation procedures completed	Testing completed: privacy assurances, A/B testing Security assessment completed

Summary

Category	Example
Integration with existing processes & systems	New interfaces or APIs required User process changes required
Instrumentation & monitoring in place	Model endpoint is instrumented to feed data back into training process Monitoring process that identifies model drift is defined
Production model revisions	Plan to retrain, relaunch models in place & funded

Is this Feasible and Valueable?

Milestones

Specify milestone to help keep track of the progress and focus on the pressure points in the development

1. Milestone: Domain expertise
2. Milestone: Reproduce other work with similar goals
3. Milestone: Automate the training and inference pipeline with a derived model from 2

1. Milestone: Domain Expertise

Start simple!

- Without machine learning
- With heuristics provided by domain experts
- Questions: How do people currently solve the problem? How would you solve the problem manually with the current data at hand?

Look at the data first!

- Apply exploratory data analysis (EDA) to get a feeling about the state of data (e.g., quality and availability of labels, missing data, false data, trends)
- Manually validate assumptions and confirm probably model choices
- Manually label data to build heuristics for solving the problem

2. Milestone: Reproduce Similar Work

Idea: Get a feeling of where are the limits of current approaches and how they tackle similar problems;
Bonus: Obtain examples and (open/licensed) source code to get quickly into the modelling phase

- Try to find similar implementations or similar data sets for your problem
- Reproduce the implementations / findings
- Convince your team that the task is feasible

How to find “similar” data sets?

- ML models map inputs to outputs. What are similar input and output **types**?
- Examples: Pet image labelling vs. generating HTML code from a Web page sketch; Or, classify food categories from text vs. classify music genres from chords

Question:

Which open data set might be helpful in your task of predicting the viewership of news articles?

Idea: Traffic statistics of Wikipedia

Data Sources

from Building ML Power Applications

- Internet archive
- Subreddit r/datasets
- Kaggle dataset page
- UCI Machine Learning repository
- Google dataset search
- Common crawl
- Wikipedia ML research data sets

2. Milestone: Reproduce Similar Work

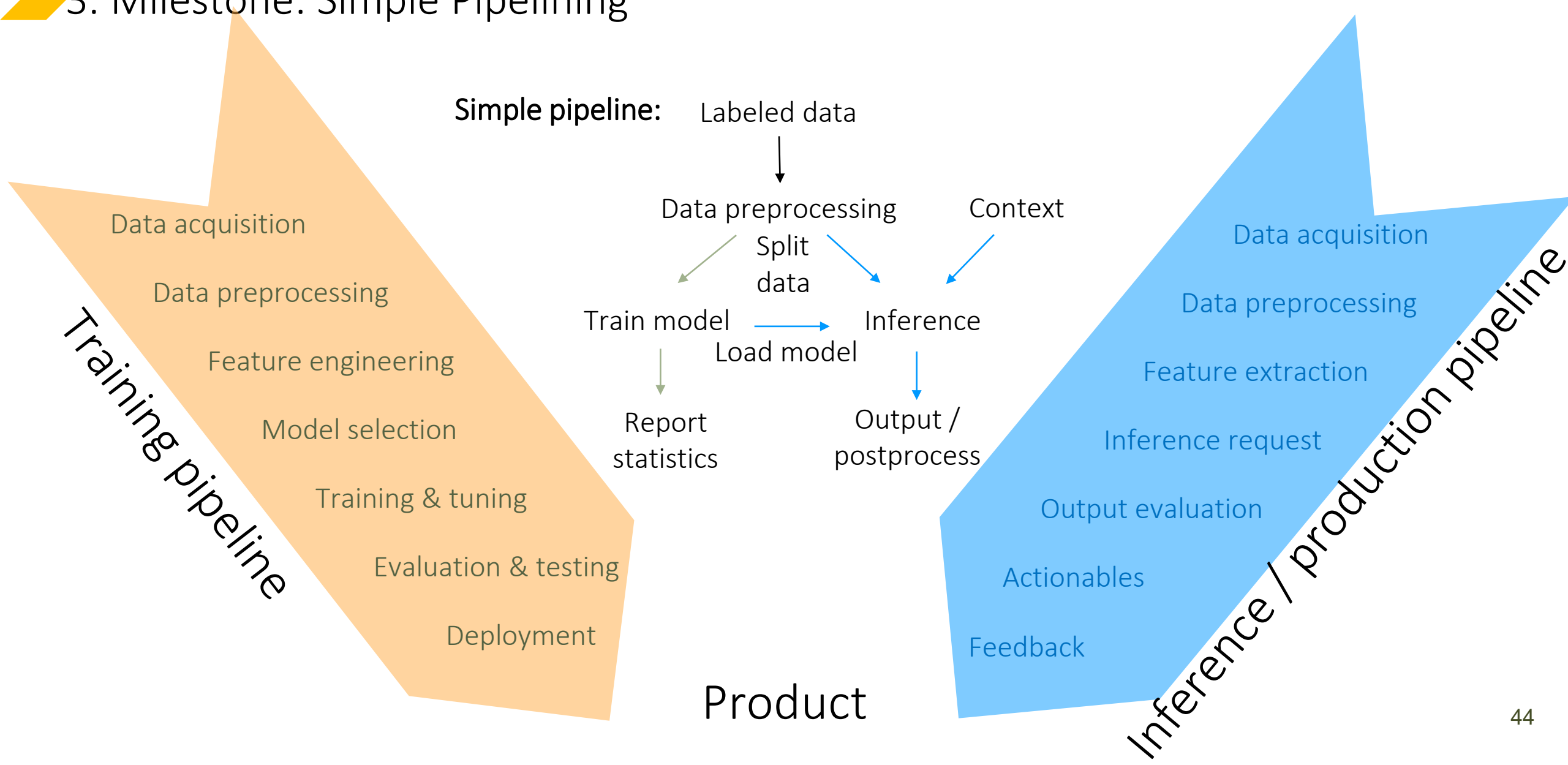
With these tangential data sets, we need to find similar implementations / models

Ways to find code:

- Papers with code repository (> 60,000 papers)
- Kaggle ML competitions (forums and tutorials often have code)
- GitHub search
- Youtube / MOOC / Tutorials from Web search
- Frameworks (TensorFlow, PyTorch, Scikit learn, Spark, etc.)

Try to find also pipeline code! (if you manage to do so, please update us! ☺)

3. Milestone: Simple Pipelining



Budget for Resources: Stage of Project

Determine what stage of project you are in:

- **Research project:** Exploratory in nature with questions such as “Is it possible to …”, “Can we use our data to … “, “surely we must be able to solve …” Since, the answer is unknown, we must invest substantial more resources to obtain a “yes.”, “maybe, if we find more data…”, “no, we do not believe it is possible”. Hard to budget as the time and effort for answering the questions are unpredictable. Roles: data scientists.

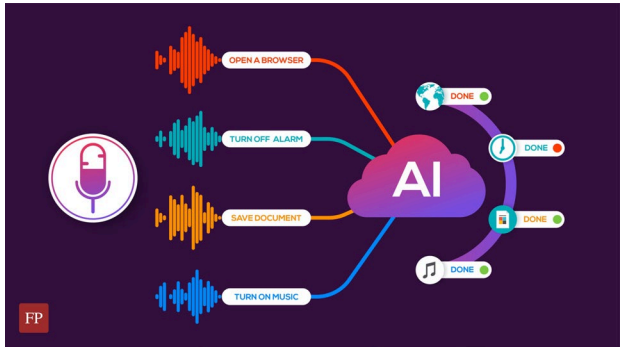
- **Development project:** Problem and feasibility is known as well as how to solve it. New questions arise: “How to scale the implementation?”, “How to pipeline end-to-end?”, “SLA requirements?”, “How to test it?”. These kinds of projects are easier to budget as we can calculate person moths and expected implementation duration. Roles: ML researcher, software engineers.

Budget for Resources: Development Project

Overview of expected costs for a small sized project with existing code:

- **Data collection and labelling**
  - Expected amount: 100K labelled data points (according to a study by Dimensional Research)
  - Manual labelling (5-10 labels with annotations per hour)
  - Outsourced activity (e.g., Mechanical Turk: 70,000$ for 100k data points),
  - Acquired data sets (e.g., Scaled 8,000$ – 80,0000$)
  - Quality check (ca. 5,000$)
  - **In total: 10,000$ - 85,000$**
- **AI research and implementation**
  - ML researcher: 5,000$ per month … about **15,000$** with 3 researchers (considering code can be reproduced)
- **Productionizing**
  - Infrastructure cost (cloud, data storage, training HW, data pipeline development, API development, etc.)
  - Varies a lot: can be between **30,000$ and 120,000$**
- **Total: 55,000$ - 220,000$**

Example: Speech Recognition*



Scope: Speech recognition for voice-based search

Metrics: Accuracy, latency, throughput

Project constraints: Timeline and resources

Data definition: Consistently labelled? How much silence before and after a speech clip? How to do volume normalization? -> Build conventions befor labelling

Model: Data-centric vs. model-centric: Fix code (model) instead of data and optimize on parameters and data

Deployment: Mobile phone (edge device) with local SW recording audio, sending API request to a prediction server and results transcript + search results

Project Archetypes

Improve existing process:

- Automate, speed up, reduce error, improve resolution of existing process
- **Validation:** Actual improvement?, performance improvement = higher business value?
- **Examples:** Code completion, customized recommender system, production line

Augment manual processes:

- Support manual tasks with suggestions, corrections, alternatives
- **Validation:** Performance threshold to be useful?, amount of data required?
- **Examples:** Slide designer, transcription engine, radiologist support

Automating manual processes:

- Replace manual work with an AI-based solution
- **Validation:** Acceptable failure rate?, guarantees for failure rate? Data labelling cost?
- **Examples:** Self-driving, Chat robot, customer support, website design

Define Degree of AI Automation in the System

Shadow deployment

Partial automation, AI co-worker

Human

Human in the loop

AI

Manual

AI assistant

Full automation

GitHub Copilot

Code recording plugin

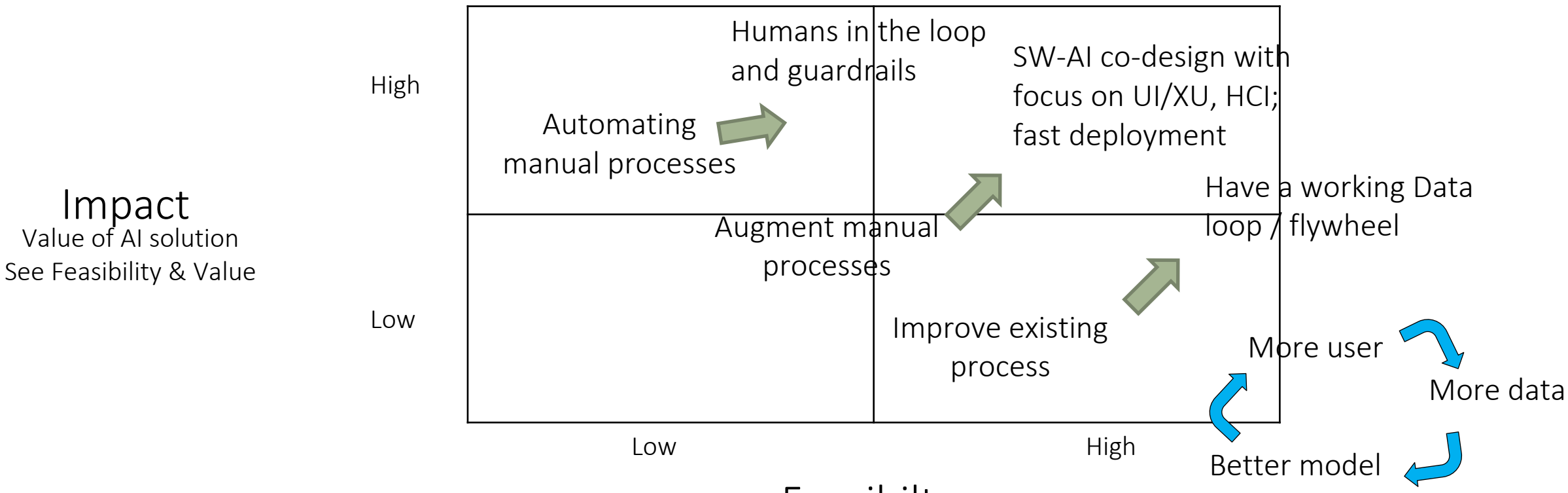
Human

AI

Developer

Code suggestions

Impact of Archetypes and Measures



Topic II:
Organization, Teams, and Roles

Learning Goals

- Know the expected responsibilities of roles in ML project teams
- Evaluate advantages and drawbacks of different team and organization compositions
- Apply best practices of team management

Organization: Teams & Roles

Challenges:

- Identifying and decomposing requirements (see requirements lecture)
- Negotiating training data quality and quantity (covered via multiple lectures)
- Integrating data science and software engineering teams (this lecture)

Views and Expectations on Roles: Variant 1

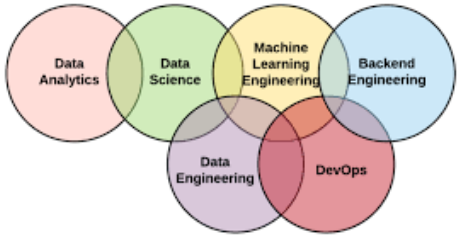
Roles	Tasks	Technologies
ML product engineer	Management part, business	Jira, Issue board, Kanban
ML engineer	Train & deploy ML models	TF, Docker
ML researcher / scientist	Build algorithms & train	TF, Pytorch, scikit, etc.
Data engineer	Data pipelines, storage, etc.	Airflow, Kafka, etc.
Data scientists	Wildcard for all other roles	Pandas, TF, SQL, etc.
DevOps (engineer)	Deploy & monitor production	AWS, cloud vendor, Kubeflow, etc.

Views and Expectations on Roles: Variant 2

Warning: Role names are not clearly defined and their responsibilities or considered working areas overlap. Excerpt:

- AI researcher:
  - Fundamental research in AI
  - Background in mathematics, statistics, ML, information theory, computer science
  - Located in academia and large companies (Google, Microsoft, Apple, Meta, etc.)
- AI specialist:
  - Refine and apply AI/ML techniques for concrete applications
  - Background in software engineering and AI/ML
- Specialist breakdown:
  - Data scientist: Background in math, statistics, analytics, ML
  - Data engineer: Background in databases, programming, big data, analytics
- Domain experts:
  - Know how data in a domain can be obtain and how to interpret them
  - Involved in pre-processing and validation (assumptions about the data)
- Engineers:
  - Infrastructure engineer: Setting up infrastructure for the pipeline, cloud provision
  - Application developers: Software engineers who productionize a model (deployment, monitoring, APIs,...)

Views and Expectations on Roles: Variant 3



A **machine learning engineer** should be able to:

- Assert that all tasks in production perform and are scheduled as expected
- Use ML libraries to their extent and possibly add new functionality
- Ensure that data science code is maintainable, scalable, debuggable
- Automate repeating tasks
- Knows and exercises SE best practices (SOLID, patterns, CI/CD, versioning, testing, error recovery, user acceptance testing, failover mechanisms, etc.)
- Chooses a suitable operational architecture together with the DevOps team
- Improves performance and make informed decisions about appropriate ML models

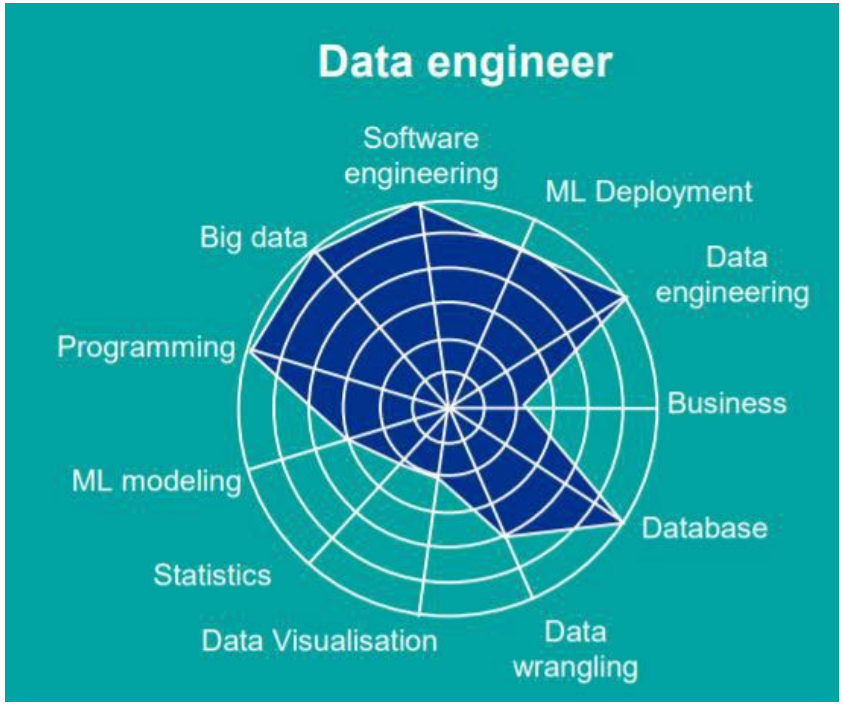
Views and Expectations on Roles: Variant 4

Data engineers should be able to:

- Create and maintain training pipelines
- Build a scalable ML system (including ML deployment)
- Build an infrastructure for data extraction, transfer, and load (ETL)
- Implement further utilities and tools along the way

Machine learning engineers should be able to:

- Design and implement ML systems
- Research and implement ML algorithms
- Select appropriate data sets
- Run and analyze ML experiments
- Conduct statistical analyses

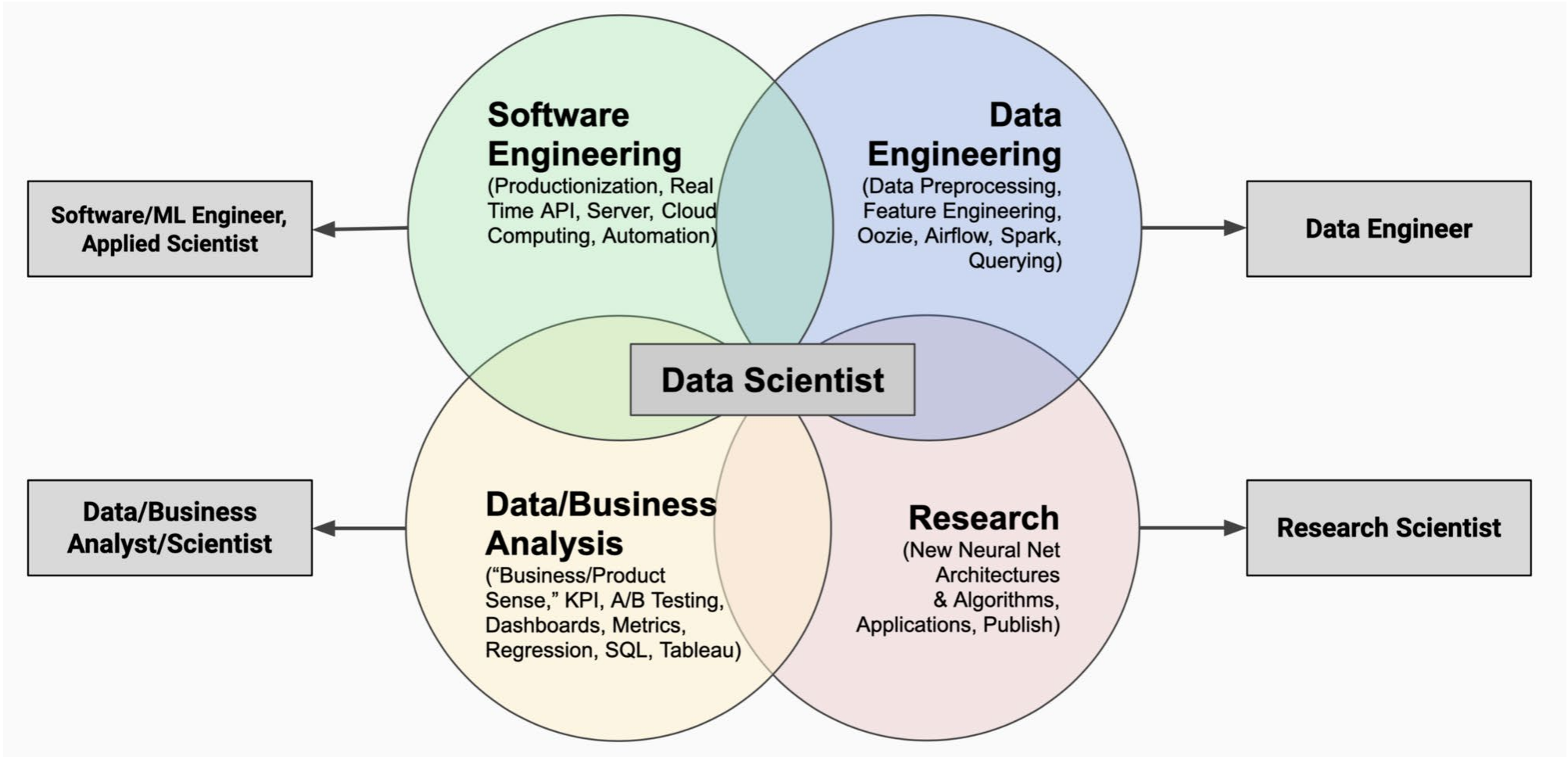


Views and Expectations on Roles: Variant 4

A data scientists is a combination of data engineering and ML engineer:

- Knows ML
- Knows programming
- Can analyze data and experiments
- Can visualize data
- Can automate things

Views and Expectations on Roles: Variant 4



Live Experiment

One group per role (data scientist, data engineer, ML engineer, ML Ops engineer) and 4 sources:

- https://www.jobvector.de/#
- https://www.stepstone.de
- https://de.indeed.com
- https://de.linkedin.com/jobs

Task: In 5 minutes, collect a list of traits and skills for each role such that we can compare them.

And it goes on and on… cloud solution architect, AI solution architect, ML Ops engineer, full stack ml engineer

Recap: Collaboration in SE

Teams build around the technical structure of a system (e.g., a team per microservice)

Teams negotiate and collaborate when their technical parts meet (e.g., API calls of one component to another)

Different development processes exist:

- Sequential: Teams per field of expertise with clear boundaries and responsibilities
- Spiral: Team to prototype the component having largest risk to realize first
- Agile: Teams are composed from different experts (UI/UX, backend, frontend, testing, etc.) iteratively refining prototypes to a shippable product

Responsibilities of AI Team Management

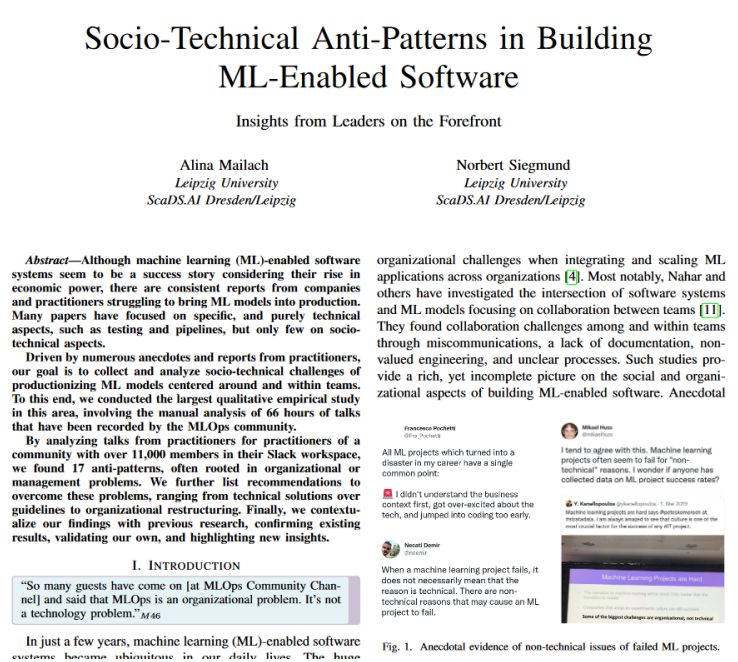
Responsibilities

- Hire the right people
- Manage and develop people
- Managing teams' output and align goals
- Good long-term decisions and technical debt reduction
- Manage expectations from leadership

Obstacles

- Education, composition, scarcity of people and limited budget
- Diverse roles, technology-hyped, CV-driven
- Timelines and uncertainty
- Technical debt (pipeline erosion), unclear technological trend
- Management level often have a limited or false understanding of „artificial intelligence“

Organizational Anti-Patterns



Socio-Technical Anti-Patterns in Building ML-Enabled Software

Insights from Leaders on the Forefront

Alina Mailach
Leipzig University
ScaDS.AI Dresden/Leipzig

Norbert Siegmund
Leipzig University
ScaDS.AI Dresden/Leipzig

***Abstract*—Although machine learning (ML)-enabled software systems seem to be a success story considering their rise in economic power, there are consistent reports from companies and practitioners struggling to bring ML models into production. Many papers have focused on specific, and purely technical aspects, such as testing and pipelines, but only few on socio-technical aspects.**

Driven by numerous anecdotes and reports from practitioners, our goal is to collect and analyze socio-technical challenges of productionizing ML models centered around and within teams. To this end, we conducted the largest qualitative empirical study in this area, involving the manual analysis of 66 hours of talks that have been recorded by the MLOps community.

By analyzing talks from practitioners for practitioners of a community with over 11,000 members in their Slack workspace, we found 17 anti-patterns, often rooted in organizational or management problems. We further list recommendations to overcome these problems, ranging from technical solutions over guidelines to organizational restructuring. Finally, we contextualize our findings with previous research, confirming existing results, validating our own, and highlighting new insights.

I. Introduction

"So many guests have come on [at MLOps Community Channel] and said that MLOps is an organizational problem. It's not a technology problem."[M46]

In just a few years, machine learning (ML)-enabled software systems became ubiquitous in our daily lives. The huge organizational challenges when integrating and scaling ML applications across organizations [4]. Most notably, Nahar and others have investigated the intersection of software systems and ML models focusing on collaboration between teams [11]. They found collaboration challenges among and within teams through miscommunications, a lack of documentation, non-valued engineering, and unclear processes. Such studies provide a rich, yet incomplete picture on the social and organizational aspects of building ML-enabled software. Anecdotal

Fig. 1. Anecdotal evidence of non-technical issues of failed ML projects.

Organizational Silos

> "How do I hand off this model to this other team? And there's a lot that goes into it, it's the actual version of the code to instantiate the model, it's the right set of weights. You probably want some way to track metadata associated with it, in case there are scores to be able to interact programmatically. And so at its core, this suddenly goes from what sounds like it should be a pretty easy problem to 'I'm spending weeks and meetings and emails trying to physically figure out and help this person instantiate my model and load my weights.' [..] And then, it gets worse because you try to build a new version of the model and instead of that person being able to grab the new version, they have to go through this whole stupid cycle again."$_{M36}$

(A) Model to production integration

Anti-Patterns (AP):

AP1: Long release cycles

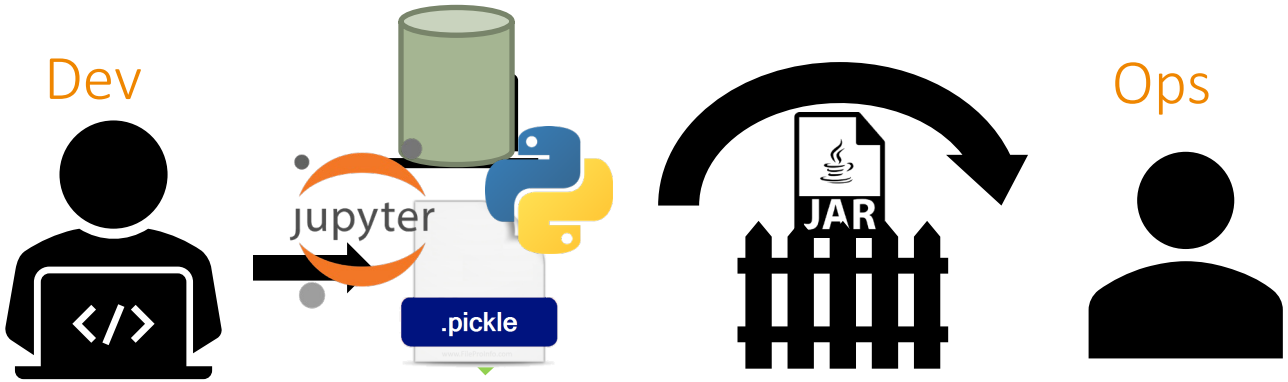
Long discussions over details due to lack of abstractions, documentations, and processes

AP2: Tension between teams

Complaints, often driven by a lack of understanding, cause frustration

AP3: Blocked data scientists

Data scientists cannot focus on developing new models, but are instead occupied with productionizing the model



Organizational Silos III

(A) Model to production integration

Causes (C):

C1: Clash of cultures and tools

Cultural differences between DS and SE (e.g., research driven DS vs. well-tested product-driven SE)

C2: Missing engineering competence

DS lack basic SE principles, such as code quality, design patterns, automation pipelines, virtualization

SE lack DS skills, such as ML basics, leading to often resorting back to the DS team

Organizational Silos III

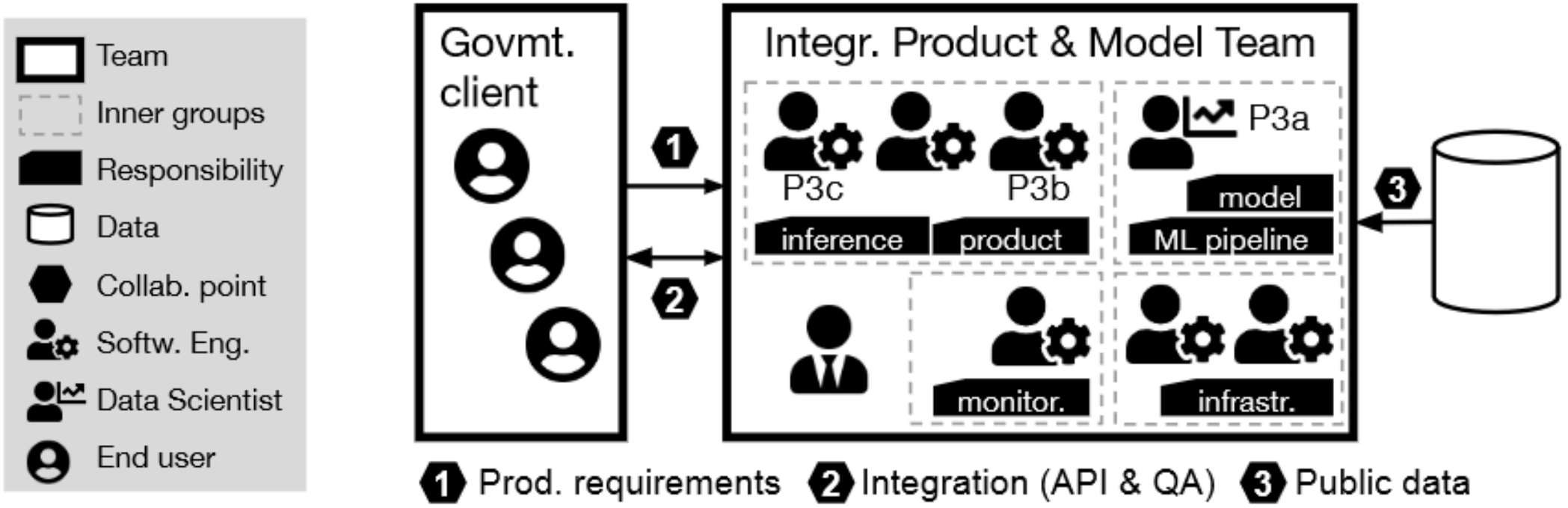
(A) Model to production integration

Recommendations (R):

- R1: Model registry and feature store
  - Idea: easing the symptoms by providing tools to have a cleaner interface between the two teams
  - Model registry are stores for model versions including their metadata and artifacts
  - Feature stores are databases of computed features for training and inference
- R2: Restructure teams to be cross-function
  - Cross-functional or integrated teams can help bridge the gap between DS and SE
- R3: Pair data science and software engineers
  - Pairing two developers with different background or using code reviews with changed teams
- R4: Translation work between the different roles and common understanding of the goal
  - Build a common language, share the respective goals of DS and SE, build shared responsibility

Example Team Compositions: Integrated Team

Challenges: Single data scientists feels separated in the team, knowledge silos, limited product feedback due to missing product owner



Organizational Silos IV

> "So, the backend handing over that data object, that has been a very painful process and primarily not so much because of technological restrictions, but much more because organizationally the data team has been quite separate from the backend team. Trying to get stories into their system has been quite difficult. And I think that's something that I still see today: where it's very hard to convince other teams to expose some kind of data that you would need or to get them to fetch some data that's not directly relevant for them or not relating to some kind of use case." $_{M46}$

(B) Data Producer vs. Data Consumer

Anti-Patterns (AP):

AP4: Requesting data is hard

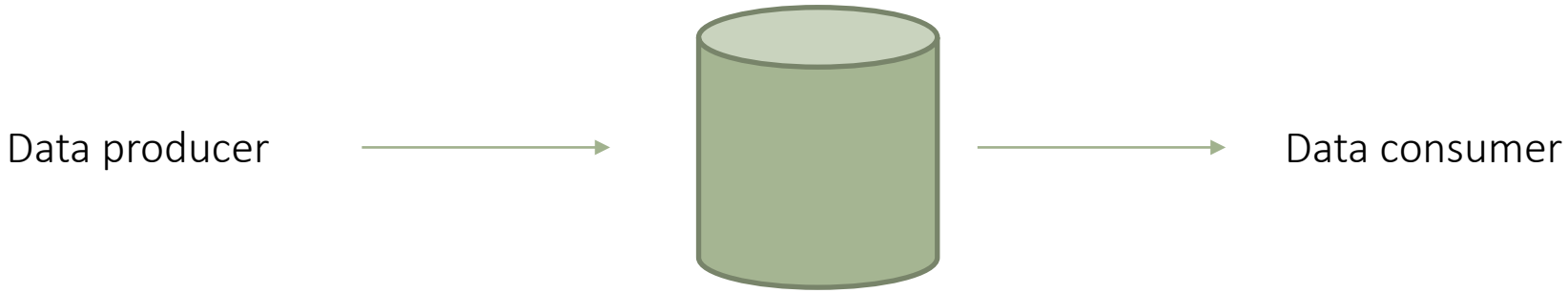
Hard to convince the producer to expose data; often no use case for producer involved (no benefit)

AP5: Tight technical coupling between producer and consumer team

Direct dependencies in code and data stores can lead to visible and silent failures at consumer side

AP6: Partial data availability

Incomplete data at the DS team can lead to degraded performance in production



Organizational Silos V

(B) Data Producer vs. Data Consumer

Causes (C):

C3: Lack of awareness and common goals

Organizational division results in limited awareness and concern about the data so value in spending time for fetching and maintaining data is no visible; Consumer requests can harm producers' success metrics.

C4: Missing documentation and unclear responsibilities

Intransparency about who produces data and who is responsible for maintaining it; definitions about data lacks clarity; data can hardly be trusted without documentation and metadata

Recommendations (R):

R5: Raising awareness of data producers

Demonstrate importance of data to producers; education and feedback towards producers

R6: Central platform as a contract between producer and consumer

Implement central platform for discovering and accessing data; platform should define quality standards, metrics, and documentation about ingested data

Communication I

> "Let's say you have two models which are quite similar. Both use the same feature. Both are developed in different teams. And then, those teams start to develop exactly the same feature in slightly different ways but semantically with the same meaning." $_{M29}$

(A) Redundant development

Anti-Patterns (AP)

AP7: Features are not discoverable, accessible, or reusable

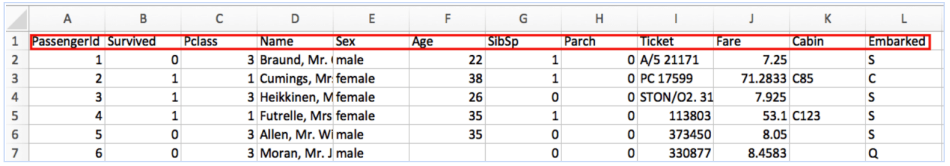
Semantically similar features may be developer multiple times since there is no awareness that this feature already exists; features with similar names, but different meanings can also degrade reusability

AP8: Redundant ML infrastructure and tools

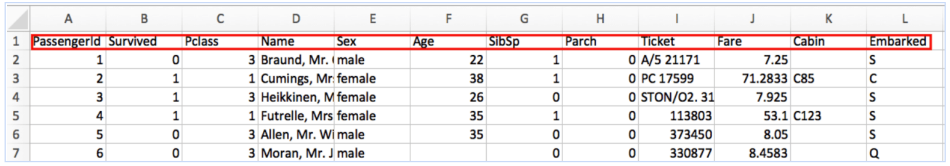
Every model development requires own raw data and preprocessing, possibly leading to redundant infrastructure for processing and tools for producing the data

AP9: Shadow IT

Teams may resort to self-hosted IT solutions with degraded security standards despite the availability of a dedicated central infrastructure provider



	A	B	C	D	E	F	G	H	I	J	K	L
1	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	1	0	3	Braund, Mr. (	male	22	1	0	A/5 21171	7.25		S
3	2	1	1	Cumings, Mr	female	38	1	0	PC 17599	71.2833	C85	C
4	3	1	3	Heikkinen, M	female	26	0	0	STON/O2. 31	7.925		S
5	4	1	1	Futrelle, Mrs	female	35	1	0	113803	53.1	C123	S
6	5	0	3	Allen, Mr. Wi	male	35	0	0	373450	8.05		S
7	6	0	3	Moran, Mr. J	male		0	0	330877	8.4583		Q



	A	B	C	D	E	F	G	H	I	J	K	L
1	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	1	0	3	Braund, Mr. (	male	22	1	0	A/5 21171	7.25		S
3	2	1	1	Cumings, Mr	female	38	1	0	PC 17599	71.2833	C85	C
4	3	1	3	Heikkinen, M	female	26	0	0	STON/O2. 31	7.925		S
5	4	1	1	Futrelle, Mrs	female	35	1	0	113803	53.1	C123	S
6	5	0	3	Allen, Mr. Wi	male	35	0	0	373450	8.05		S
7	6	0	3	Moran, Mr. J	male		0	0	330877	8.4583		Q

Communication II

(A) Redundant development

Causes (C)

C5: Missing intersections between teams, documentation, and trust

Teams work independent from each other, so there is no natural intersection among them; Teams have no communication and aren't aware about developments by others; no incentive for documentation and preparation for reusability of features and infrastructure

Recommendations (R)

R7: Unification and discovery through centralization

Enable discovery of features through central management of tools and infrastructure (unify tools); feature stores are a recommended solution

R8: Showcase meetings

Improve understanding of each other's work and progress via meetings in which a team presents its work, their use case, progress, and learnings

Communication III

> "All of the higher-ups were just looking at the data science team as a money suck. All they were doing was walking around. They weren't putting anything into production. They weren't doing much: looking at data, asking for more data. And, they weren't actually producing anything. And so the C-level executives: 'Hey what are we paying these guys for? What is going on here?' And they couldn't figure out the value." $_{M29}$

(B) Management vs. Data Science

Anti-Patterns (AP)

AP10: Data scientists struggle in their roles and get burned out

Unproductive feeling, not producing tangible results for the company; feeling of burnout due to never ending iterations without possible no valuable results

AP11: Tension between management and data science

Management expects DS to deliver value, but DS follow their research-oriented workflow; DS workflow may not directly align with the business value as metrics due no directly map and effects of model improvements are not immediately visible; managers get frustrated about DS that do not want to be managed or confronted with their own value to the company

their own value in the company: *"When you talk about data scientists: if I use the letters ROI, I get hate mails. And I've done that before! I've posted: 'You have to have ROI, you have to be connected to business problems.' I get hate mails. Data scientists hate it. [...] A lot of data scientists come straight out of academia and it's a different paradigm."* $_{M30}$

Communication IV

(B) Management vs. Data Science

Causes (C)

C6: Missing data science process

Missing standard development processes, such as SCRUM; research-oriented development contains too many uncertainties; management often unexperienced combined with a lack of documentation and reporting

C7: Communication mismatch between technical and non-technical people

Inability of technical people explaining their problems and achievements in an understandable way and relevant for the business causes tensions

Recommendations (R)

R9: Strong processes

New and clear development processes need to be implemented considering experiment-driven development

R10: Documentation and reporting

Documents enable retrospective analysis of the progress and supervision by the management; documentation is the basis for project reports and presentations

Leadership Vacuum I

> "Everyone kept thinking that the solution would be to hire more data scientists. So, for me it was going back to leadership and the head of engineering and saying: 'No, please stop! You can add [..] as many as you want, you're not going to solve the major issue that we have.'. [..] And this title conundrum came up with every single one of them. It was continuously 'Well, we can just hire a data scientist. Because we need ETL pipelines and we need somebody to make sure that they architect a very good relational database and then push all that reporting out to our multiple users', and I was like: '[..] Is that the right person that you should be hiring?' And they're like: 'It has data in the title'."$_{M37}$

(A) Headless-Chicking Hiring

Anti-Patterns (AP)

AP12: Staff with insufficient skills

Staff with ML background cannot build and maintain pipelines or applications

AP13: No product for data scientists

Hired DS for a specific task become obsolete after the task is completed; if use case is solvable without DS, the social structures and productivity can be negatively impacted

Leadership Vacuum II

(A) Headless-Chicking Hiring

Anti-Patterns (AP)

C8: Unclear roles and titles

Roles and titles are consistently undefined; the same job title can imply different tasks and required skills depending on the company

C9: Uneducated hiring

Hiring managers require more knowledge about the needed skills; missing communication about actually needed skills may lead to fallback hiring of general purpose “data scientist”

C10: Skill shortage on the market

Fast growing industry raises demand on specialized skills, possibly leading to hiring people with the wrong skills because people with the right skills are absent

Recommendations (R)

R11: Hiring for skill and potential rather than role

Providing a clear definition of the task and the relevant skills to solve it, enables hiring the right people

Leadership Vacuum III

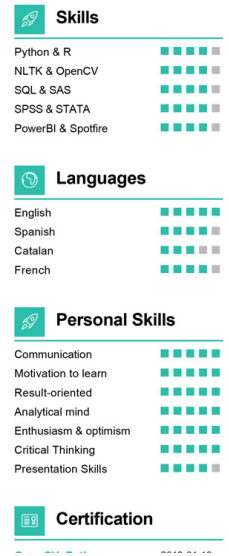
> "One of the developers, this guy, is very, very good in Scala. But, the problem is that the rest of the team used to work only [in] Python [..] This guy made the whole data processing pipeline in Scala and just delivered the mashed data for data scientists. And basically this guy [..] put in his résumé or his LinkedIn something like 'Yeah, I used to do tons of engineering in Scala'. But [..] the whole maintenance took days [..] or most of the time the code itself was not manageable at all."$_{M5}$

(B) Résumé-driven development

Anti-Patterns (AP)

AP14: Developed tools and models do not match the team or product goals

Decision process about which tools and libraries to use is highly driven by individuals; lack of authority and limited experience in the team may lead to decisions driven by personal benefits



Causes (C)

C11: Data scientists do not identify with the business value

Disconnection of technology-focused people with company outcome

C11: Missing decision maker

Unclear who makes the final decision, enables strong opinions on tools of people in a team

Recommendations (R)

R12: Rely on organization knowledge

Rely on existing technologies used by other teams as a fallback guideline if no authority is present

Leadership Vacuum IV

> "Monetizing data is different than just putting something into production slamming machine learning in. [..] No, you really shouldn't do everything [with ML]. Machine learning doesn't do! Stop! You can do a lot of what you want to do using traditional development methodologies or basic analytics."$_{M30}$

(C) Hype-driven Development

Anti-Patterns (AP)

AP15: Stuck with proof of concepts

Numerous proof of concepts without making it to production; product may not be relevant to the user

AP16: Perception of 'Everything can benefit from ML!'

No clear analysis about what aspects in the product can benefit from ML; no roadmap or strategy of how ML will help the organization achieving its goals

AP17: Product is not feasible due to missing data or talent

Organization may exactly what it wants, but does not have the right data or required staff to build it

Leadership Vacuum V

(C) Hype-driven Development

Causes (C)

C13: Missing organizational strategy, governance, and structure

Unmanaged data collection processes, missing data strategy cause problems for the first ML product

C14: Management lacks knowledge about ML

Management has no understanding what ML is and how it works, leading to underestimation of DS product development and software development; engineering effort for productionizing a model and ROI is underestimated

C15: Missing translation between different stakeholders

No management people with technological skills to guide the development of ML product resulting in poor translation of business goals and user problems to ML questions and metrics

C16: Missing production metrics

Missing production metrics causes DS not been able to evaluate their models on metrics that are meaningful for the product

Leadership Vacuum VI

(C) Hype-driven Development

Recommendations (R)

R13: Process to identify feasible use cases and product roadmap

Raising awareness of increased complexity due to ML; resort to traditional engineering methods and heuristics; identify use cases before starting to iterate on the problem; write proposals about feasibility estimates, data availability, and which organizational units should evaluate the product

R14: Understand customers and keep them close

DS should communicate with customers and understand their requirements

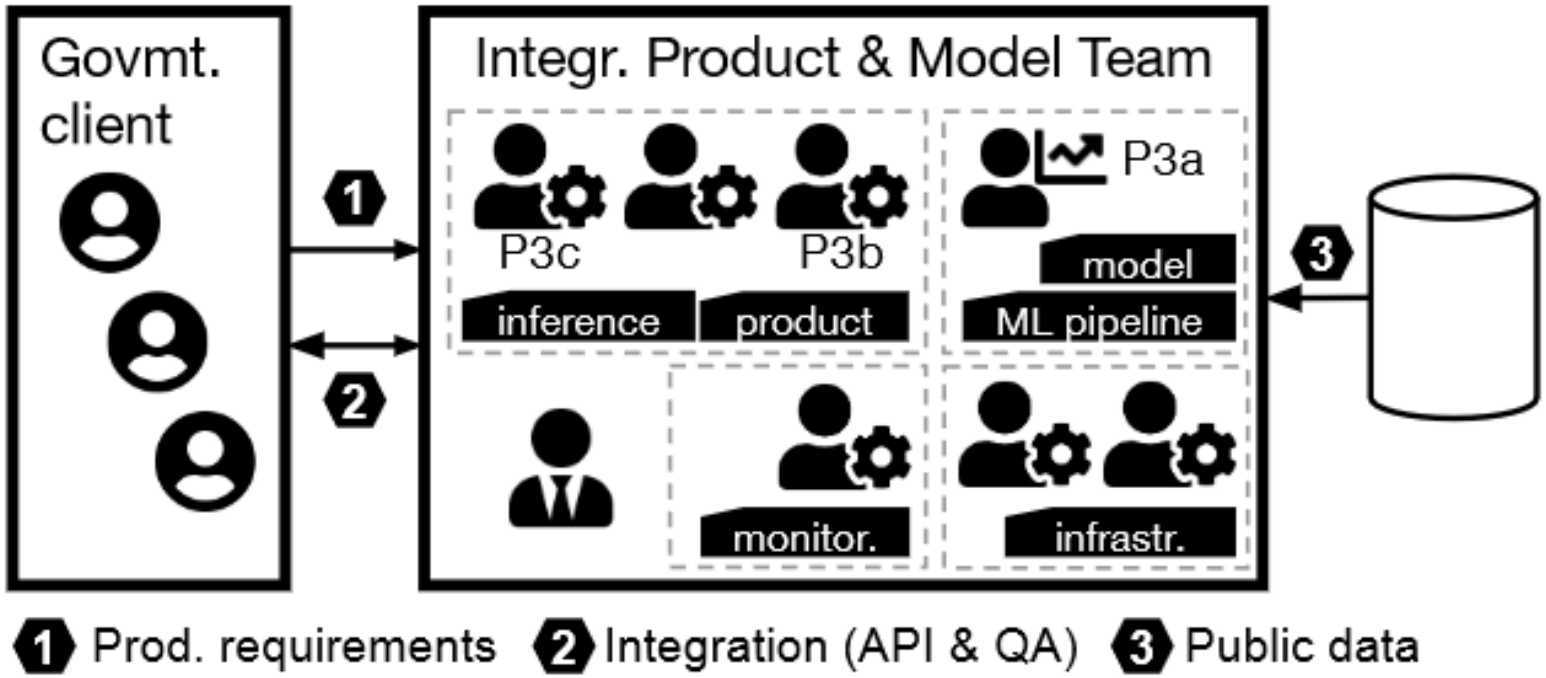
R15: Education

Education through workshops that emphasize data available

Organizational Best Practices

Integrated Team Building

If integration done right, different team members can learn from each other when directly having a shared physical environment. Integration efforts must be made explicit.



Process Ownership

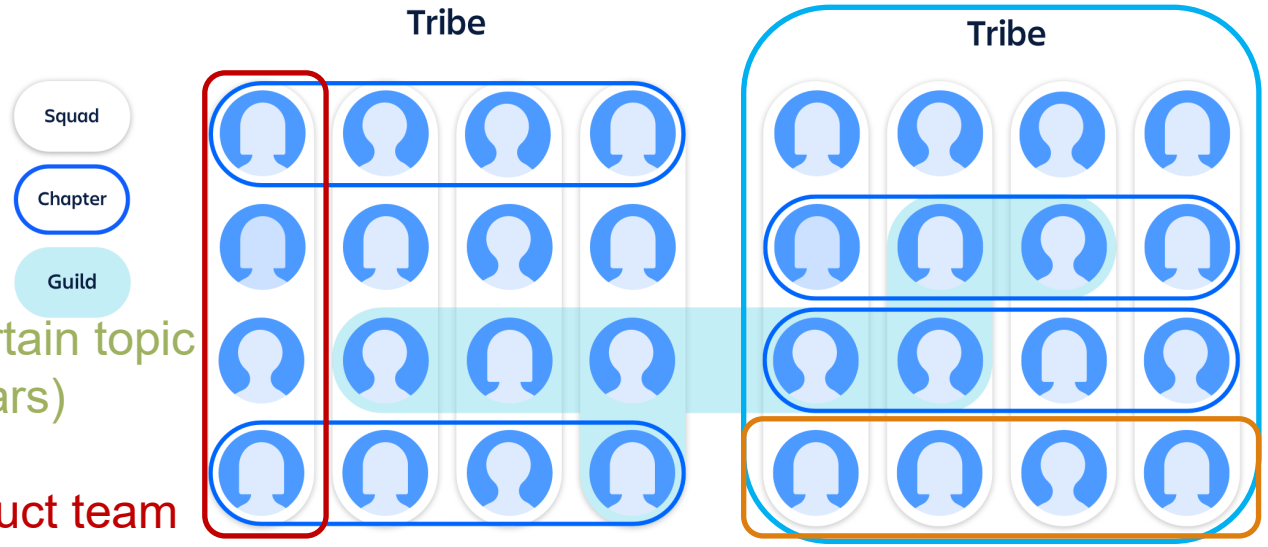
Owning the whole data science process from extraction to deployment as a single team allows for a centralized full stack view and breaks barriers across teams.

Requires clear definition of responsibilities and educated staff (software engineer requires ML knowledge and vice versa).

Centralization and Horizontal Teams

Having a central infrastructure with a dedicated infrastructure team helps unifying processes and tools across multiple ML projects within the same organization. Note that this pattern exists also in SE

Tribe ~ larger development department with possibly multiple simultaneous ML projects



Guild ~ interest group of a certain topic (e.g., speech recognition in cars)

Squad ~ agile product team

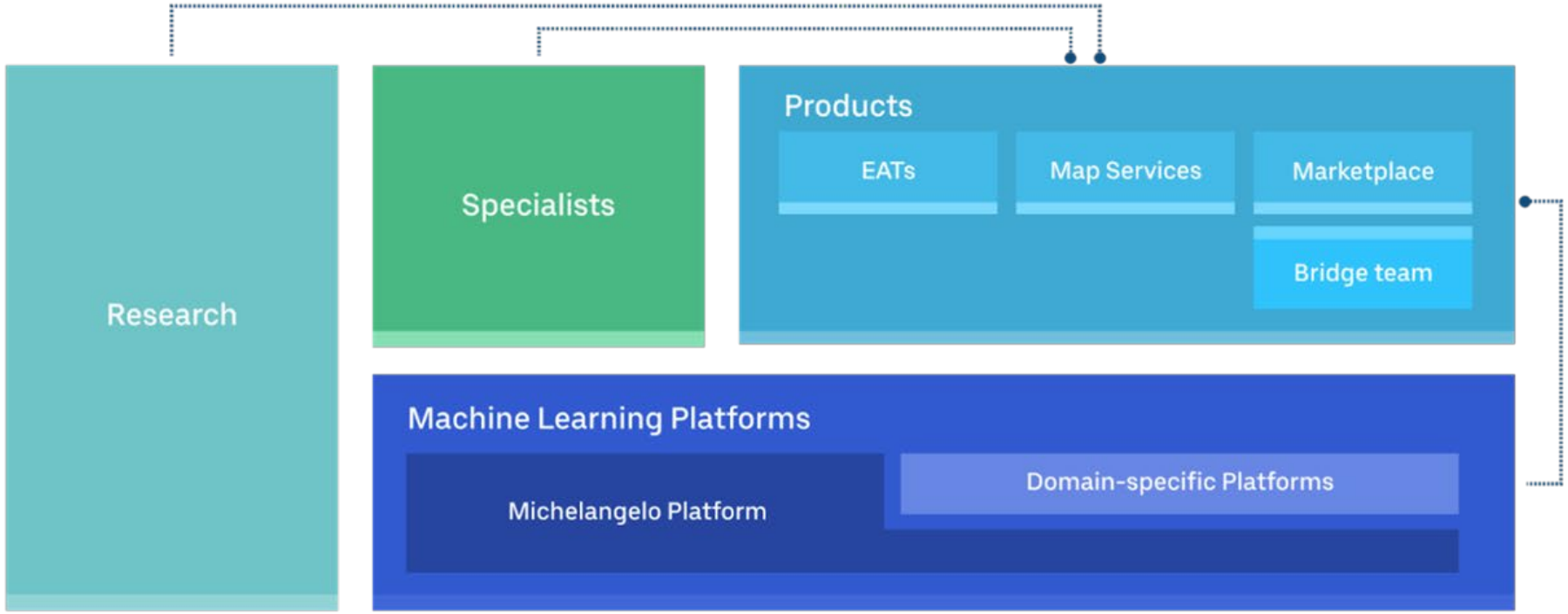
Chapter ~ group of specialist around a technology (e.g., MLOps, data engineers)

Example: Organization and Teams at Uber

ML specialists (e.g., NLP, vision, recommender, forecasting) build ML solutions, but are accompanied by product team member, such that they can maintain the model.

Product teams own the models they use and deploy. Composed of full stack devs. Get support by other teams for specific topics.

Research teams concentrate on novel technologies and tools, but do not own production code. Can be steered by specialistic and product teams.

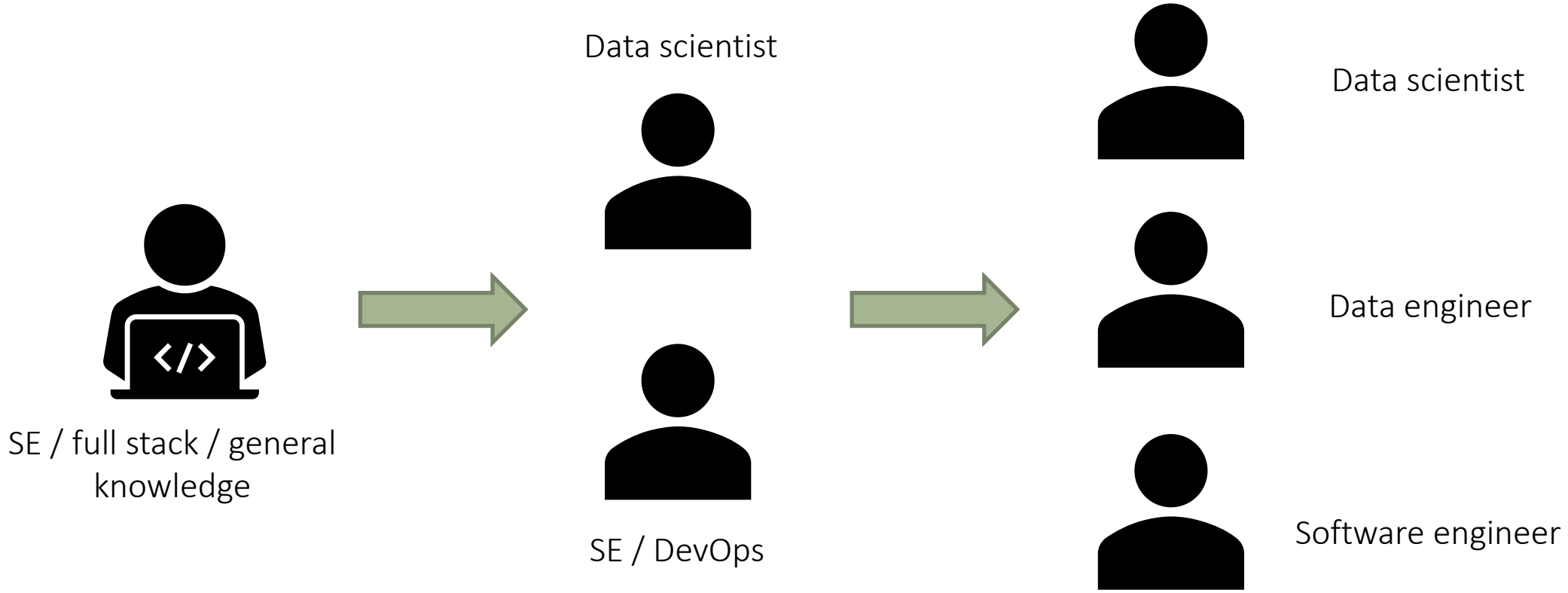


Uber Machine Learning

Platform team produces workflow tools for building, deploying, and operating ML solutions.

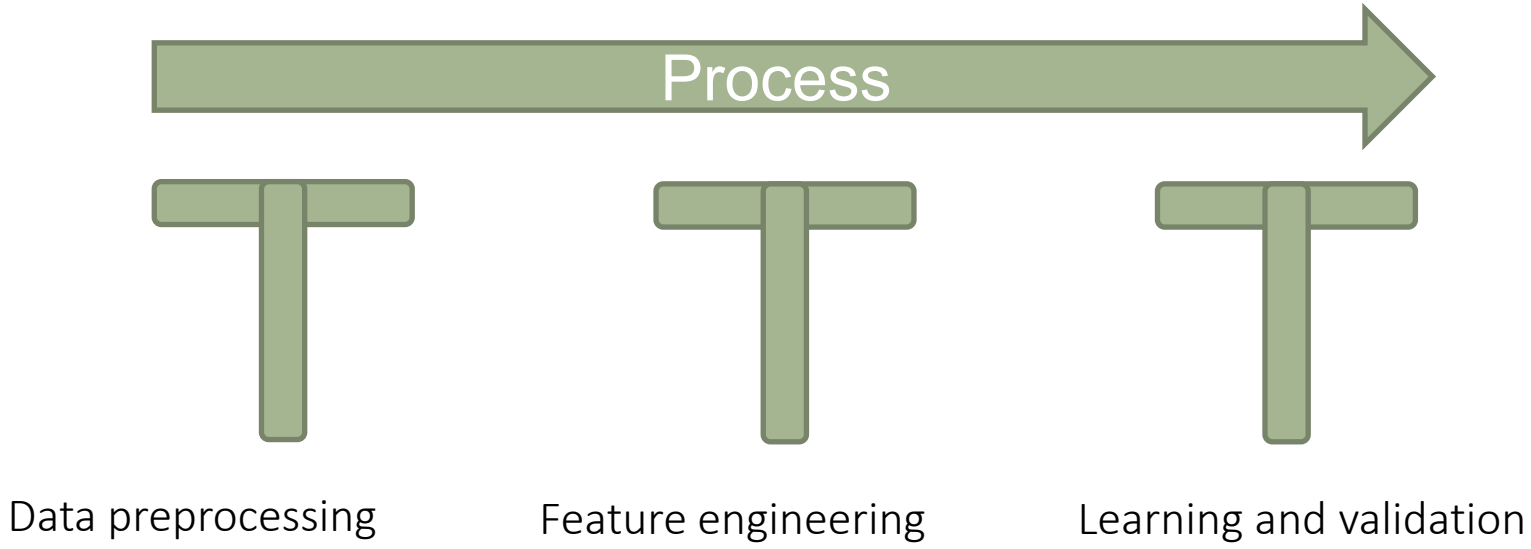
Hiring Process

For start-ups, focus hiring for generalists or software engineers that can keep up ML. The other way around is often too difficult. As the product grows, more specialized personell should be hired.



Separation of Data Science Process

Having clear boundaries within the data science process allows to develop deep knowledge in specific areas while maintaining a good overview. The goal is to build a T-shaped knowledge base.

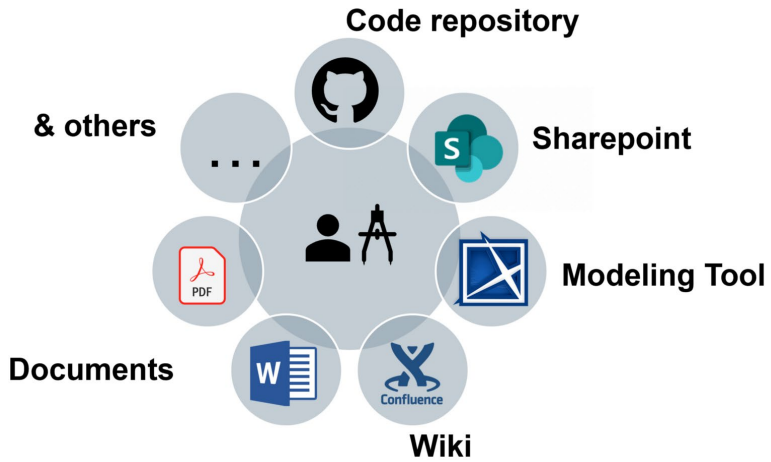


Documentation

Document artifacts, code, and processes to overcome communication problems.

Precondition for having a separated product and development team

Can be reviewed later and integrated with score / risk cards



Adapting Agile Processes

Work organization and iterative improvement is at the heart of ML projects and software engineering has developed an arsenal of techniques in the real of agile software development.

Unclear whether an entire adoption makes sense (time boxes may not work for unclear development stages), but individual methods, such as SCRUM are frequently used

Concrete ML Realization Concepts / Plans

To determine which problems to solve with AI / ML, make alternative plans that individually predict the potential value and the ability to realize and how it can be realized

Find broken items in production line

Image processing with cameras	No expert in vision available	Cost saving ~3Mio
Weighing items + user control	Easy to incorporate	Cost saving ~1Mio
Probability-based item inspection	Experts available, needs time for adoption	Cost saving unclear; it is probabilistic!

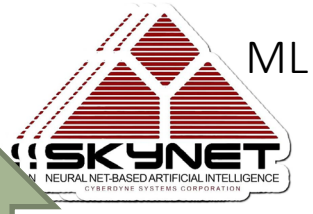
Solve Non-Technical Problems with Technical Solutions

Reducing communication effort among data, production, and ML teams via architectural solutions. Have centralized infrastructure and clear quality specifications about what comes in.

Examples:

- Model registry (having multiple models for different use cases and scenarios at a central station automatically updated by the ML team and queried / grabbed by the production team---similar to Docker hub),
- Feature store (having a server for storing features collected and created by different data teams and consumed by different development and production teams)

Organization Archetypes



ML first	CEO buys all-in into AI; ML divisions; long and short term goals; huge ML expertise in every business aspect; best data access; data and ML thinking in all stages; easy deployment; Hard to become; expensive; requires culture change (often impossible)
Independent ML team:	ML team reports to distinctive manger; ML team composed of different ML roles; Good access to data; easy to hire qualified personal; can invest into ML tooling; Model operation by other business units may be problematic; Slow feedback cycle
ML embedded in business / prodct:	Product teams have ML expertise; ML is involved in direct business decisions ML improvements impact business value; feedback cycle between idea and product Difficult to hire top talent; Access to resources lag; ML project cycle often not aligns with SE
ML in R&D:	ML takes part in the company‘s R&D processes; hire researchers Budgets for hiring qualified personel; work on long-term business goals Difficult to get data (no deployed system); Rarely productionized (small teams)
Novel or ad-hoc ML:	No or limited ML expertise in-house; small-medium businesses; Often high value and low technological risk with ML projects Limited support for ML project; hard to aquire good team

Organizational Decisions

Research vs. Productionizing:

- Do we have different teams for ML research and product development & deployment?
- How much SE knowledge is required, and do we require ML pipelines?

Data ownership:

- Who is responsible for collection, labelling, and preprocessing data?
- Who does the quality control, and do we have extra teams to organize data across multiple ML projects?

Model ownership:

- Does the ML team maintain their models in production? What about feedback loops?

Teams: Best Practices

Interdisciplinary teams: no knowledge silos, better education, better understanding

Collaboration points: Document responsibilities and interfaces

Value engineering work: Productionizing a product must be seen as a key task

Process and planning: Invest into clear processes and plan ahead instead of ad-hoc development activities

Favor Proactive Communication

Identify stakeholders and talk to them (pull requirements)

Schedule meetings with developers and data scientists and communicate decisions (architecture, infrastructure, ml, software, etc.)

Discuss with teams what they have to deliver and in what form at what time

Likely, no one will do it for you and present you the information you need.

Consider Cognitive Load of Teams

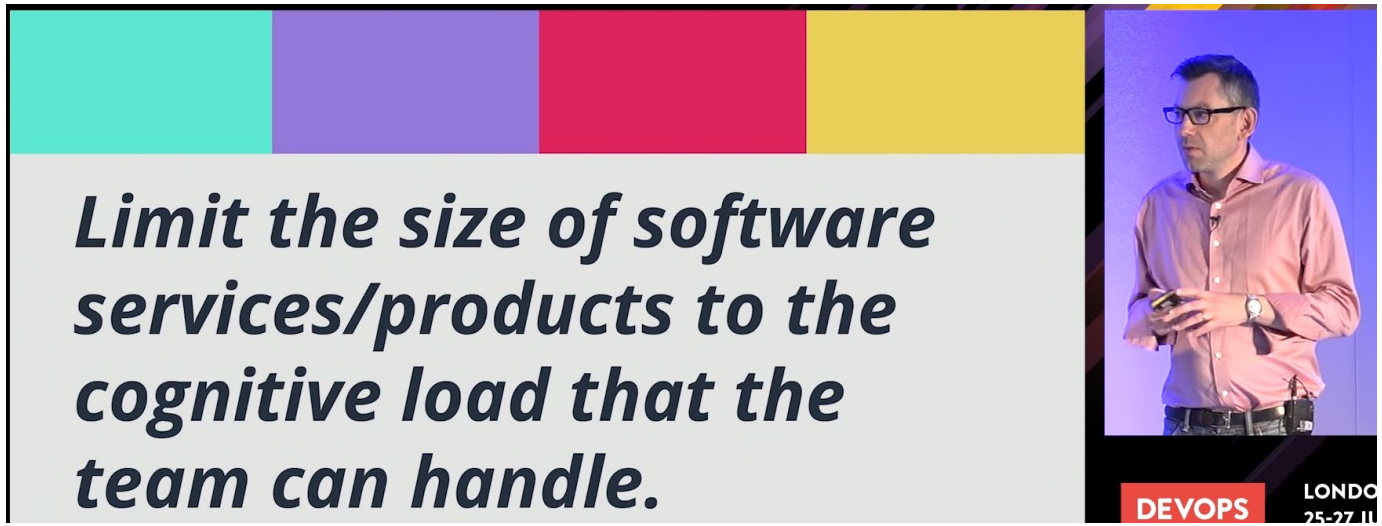
Matthew Skelton

COGNITIVE LOAD:
The total amount of mental effort being used in the working memory
- John Sweller

Intrinsic (i.e., skills)	How does regression work? List comprehension, etc.
Extraneous (i.e., mechanisms)	How do I deploy my ML model? Access to S3?
Germane (i.e., domain knowledge)	Which products do I sell? How does logistic mngt work?

Idea: Minimize load on extraneous aspects and maximize load on domain knowledge as this is linked to your business.

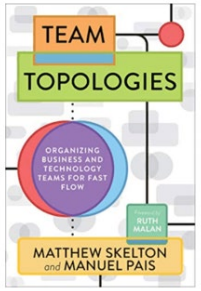
Cognitive Load: Adapt Software and not Your Team



Team-First approach:

- Ensemble programming for having a shared ownership of the service / product
- Domain-driven design (DDD) for having a well-defined domain boundary (team responsibilities are clearly defined)
- Assess and extended developer and operators expertise

Team Topologies



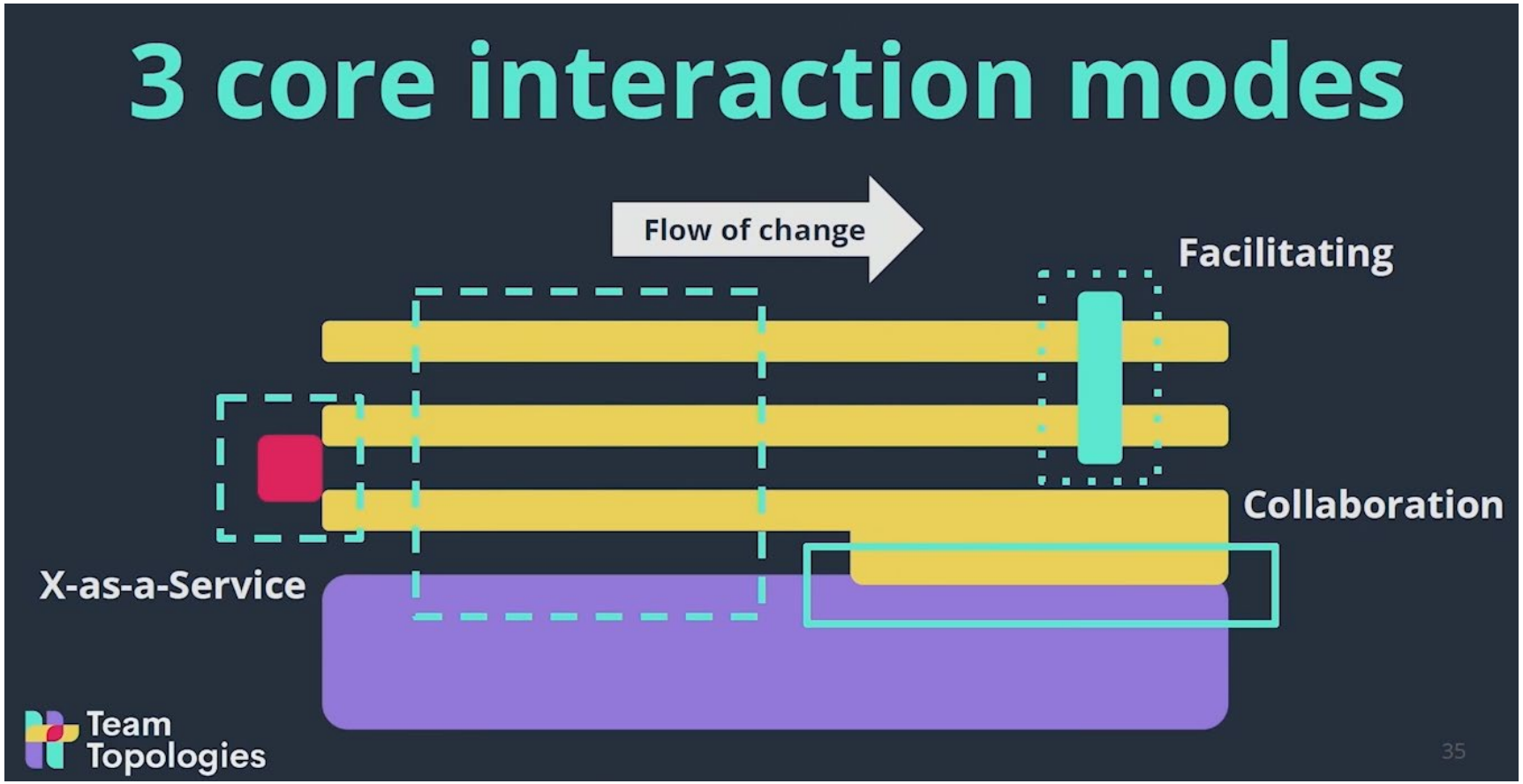
Four main topologies for building modularized software systems

- Streamlined team: End-to-end development; from features to deployment
- Enabling team: Build supporting services (e.g., transfer container technologies)
- Complicated subsystem: Concentration of a specific aspect (e.g., encryption, video streaming, etc.)
- Platform team: Offers reoccurring fundamental technologies (e.g., infrastructure access)

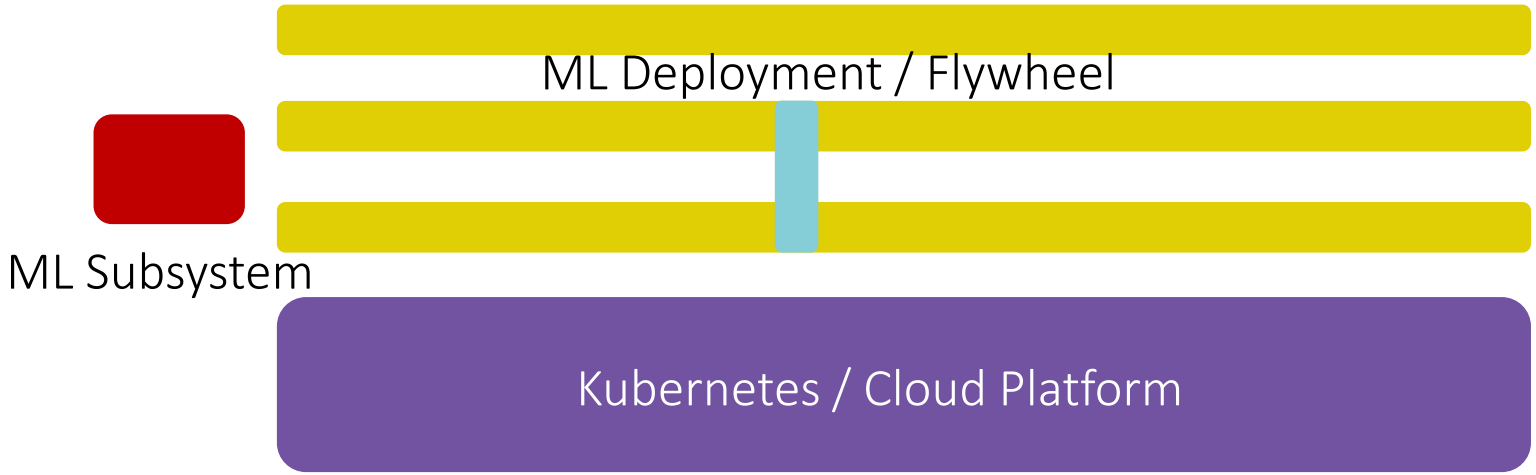
Enabling, subsystem, and platform team all reduce the cognitive load of the streamlined team, which builds the main value for the organization

Interaction Modes Between Teams

Goal: Reduce confusion between team responsibilities; reduce cognitive load of individual teams through clear interfaces



Team Topologies for AI/ML Systems



Microservices / components of my AI-based softwre system

Multiple ML projects contributing to a larger software system

Diversity

Source: Christian Kästner (**Process and Team Reflections)**

Lower the effects of group thinking (i.e., no dissenting views, limited alternatives, irrational decision making, limited outside influences) via introduction of diversity

"Men and women have different viewpoints, ideas, and market insights, which enables better **problem solving**. A gender-diverse workforce provides easier **access to resources**, such as various sources of credit, multiple sources of information, and wider industry knowledge. A gender-diverse workforce allows the company to serve an **increasingly diverse customer base**. Gender diversity helps companies **attract and retain talented women**."

"Cultural diversity leads to **process losses** through task conflict and decreased social integration, but to process gains through increased creativity and satisfaction."

Stahl, Günter K., et al. "Unraveling the effects of cultural diversity in teams: A meta-analysis of research on multicultural work groups." Journal of international business studies 41.4 (2010): 690-709.

Especially important for AI projects: Fairness, bias, ethics

Hints for team functioning

Trust them; strategic not tactical direction
Reduce bureaucracy, protect team
Physical colocation, time for interaction
Avoid in-team competition (bonuses etc.)
Time for quality assurance, cult of quality
Realistic deadlines
Peer coaching
Sense of elitism
Allow and encourage heterogeneity

DeMarco and Lister. Peopleware. Chapter 23

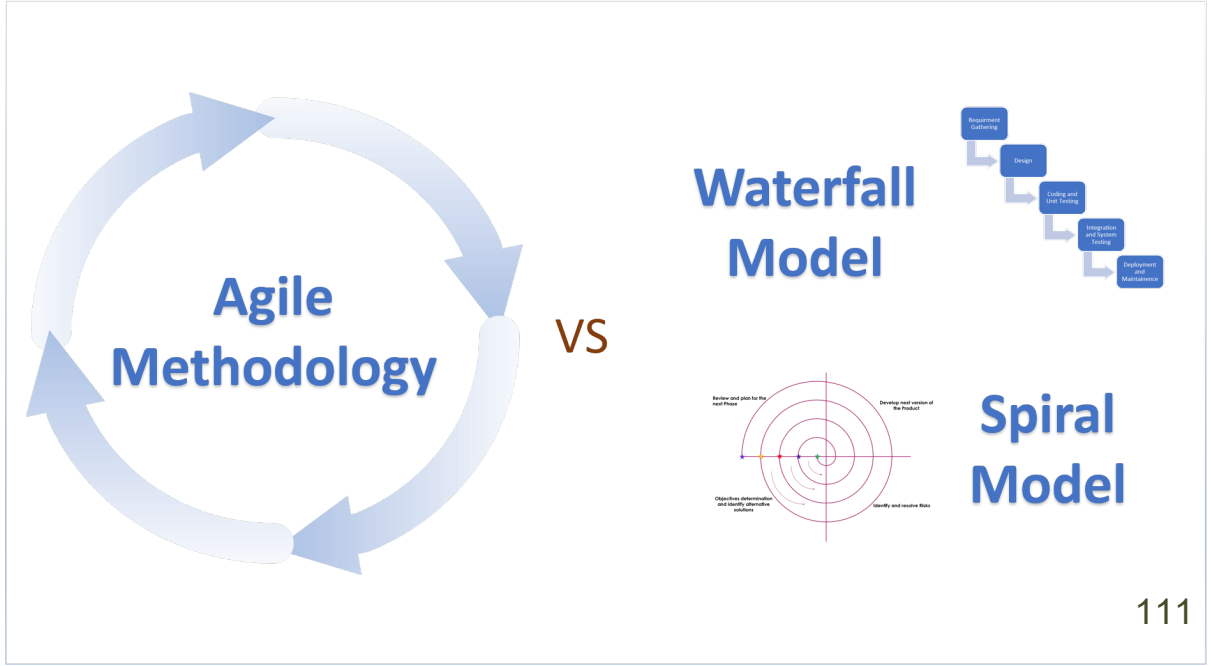
Topic III:
Development Processes

Recap: Software Development Processes

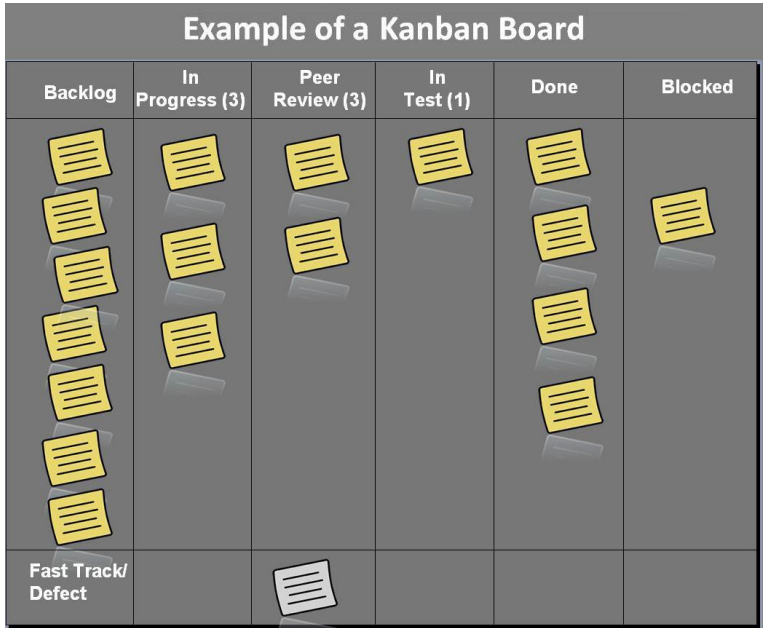
Sequential models: Waterfall, V-model (clear phases, heavy documentation, long feedback cycles)

Iterative models: Spiral model (risk-based focus), agile (short iteration cycles with small teams composed of different roles, quick feedback from customers)

Which process to apply for ML projects?



Process for Research Project



Kanban style development methodology suggested
by V.M. Megler, PhD (Amazon AWS):

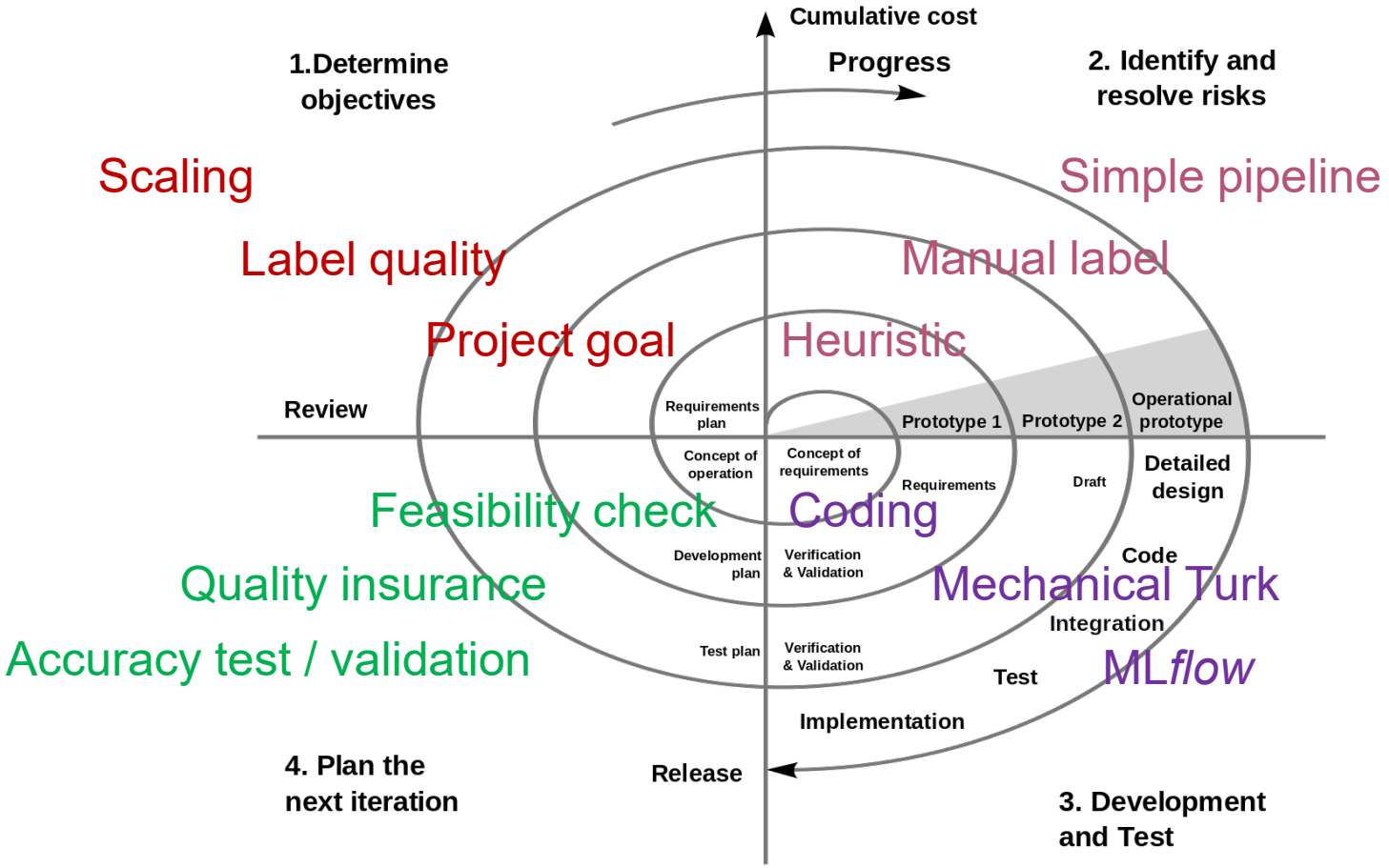
- Transparent project progress
- Tracking different stages of the pipeline on Kanban board
- No strict time slots and scheduled meeting as for Scrum-based development

Milestones for transitioning to a development project should be set.

Spiral model for transition is promising:

- Focus on parts of the pipeline with highest risk (e.g., Can we obtain labels in the right quality and quantity? Is the model powerful enough to solve the task?)

Spiral Model for Research to Dev Transition



Process for Development Project

If requirements are clear and tasks can be estimated in time and resource demands

- Agile / Scrum based development can be used.
- Integrated team

If requirements are not specified or goal is unclear or technical realization is not met

- Possibly not development project
- Separated teams: model team for technical realization and product team for specifying requirements