Executive Briefing: The black box – interpretability, reproducibility, and data management

O'Reilly Artificial Intelligence conference
London
October, 2019

Mark Madsen

What does management care about?

The perspective and problems of the person responsible for oversight of the results is spread across the organization and across multiple projects

- The owner of a decision or the outcome of the decision. Aka the process owner
- The CAO, CDO, VP of analytics, aka “your boss” if you’re a data scientist.

Repeatability

Operational predictability

Reproducibility

Can you support the answer at a later date?

Can you replicate the process used to get the answer?

Can you get similar results on the same or similar input?

Scientific reproducibility

Can you get the same results given the same starting conditions?

- Duplicate as much as possible the experiment.
- Results of experimental replication <u>may not be the same</u> for many reasons
- Detailed statistical analysis is needed
- And critical assessment of experiments

Reproducibility in data science

Can you get the same results on the same data?

- Direct replication is expected
- Assumption is that same input = same output
- You ***can*** have this with an unexplainable box
- But there are also confounding factors

Moving from predictable rule-based systems to complex mathematical systems, and from there to systems that exhibit *stochasticity,* makes the task harder.

One thing worse than a black box is a random black box.

If you trust the box then it’s ok not to understand it.

Without reproducibility there cannot be trust.

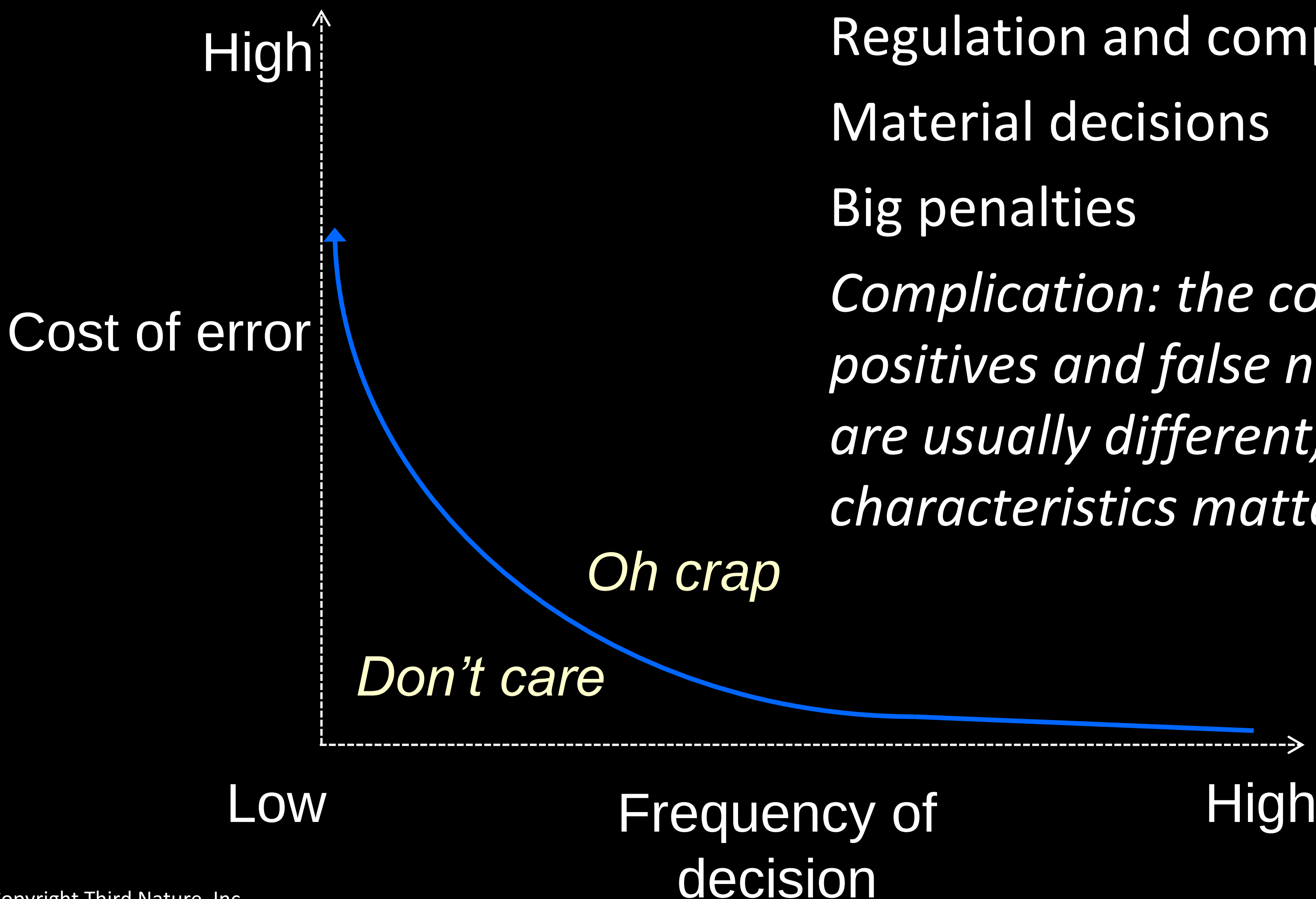
Trust comes from it working

It works = reliability, working over time

Reliability of your solution is a *systems* problem, not a just a model problem (in the GST sense of system)

Therefore, if there is trust, or there is low risk, you may not need to worry about reproducibility

Interpretability and reproducibility are driven by trust, which only matters when there is enough risk



Regulation and compliance

Material decisions

Big penalties

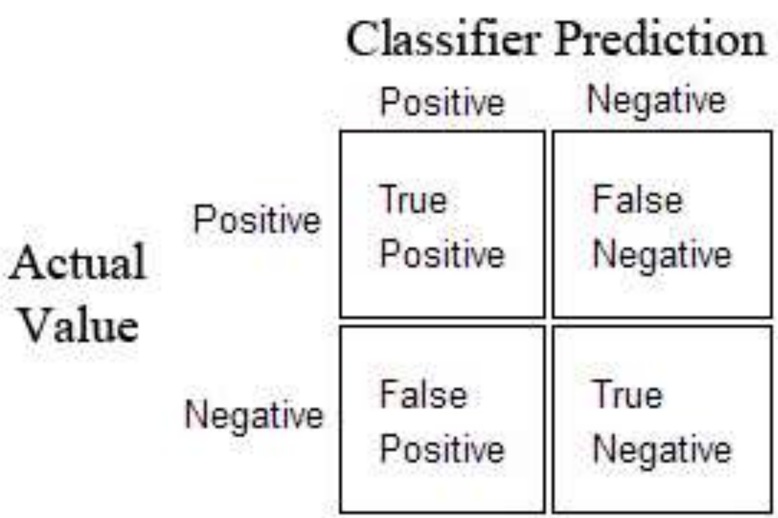
Complication: the cost of false positives and false negatives are usually different, so model characteristics matter here.

Beyond toy examples, this problem matters

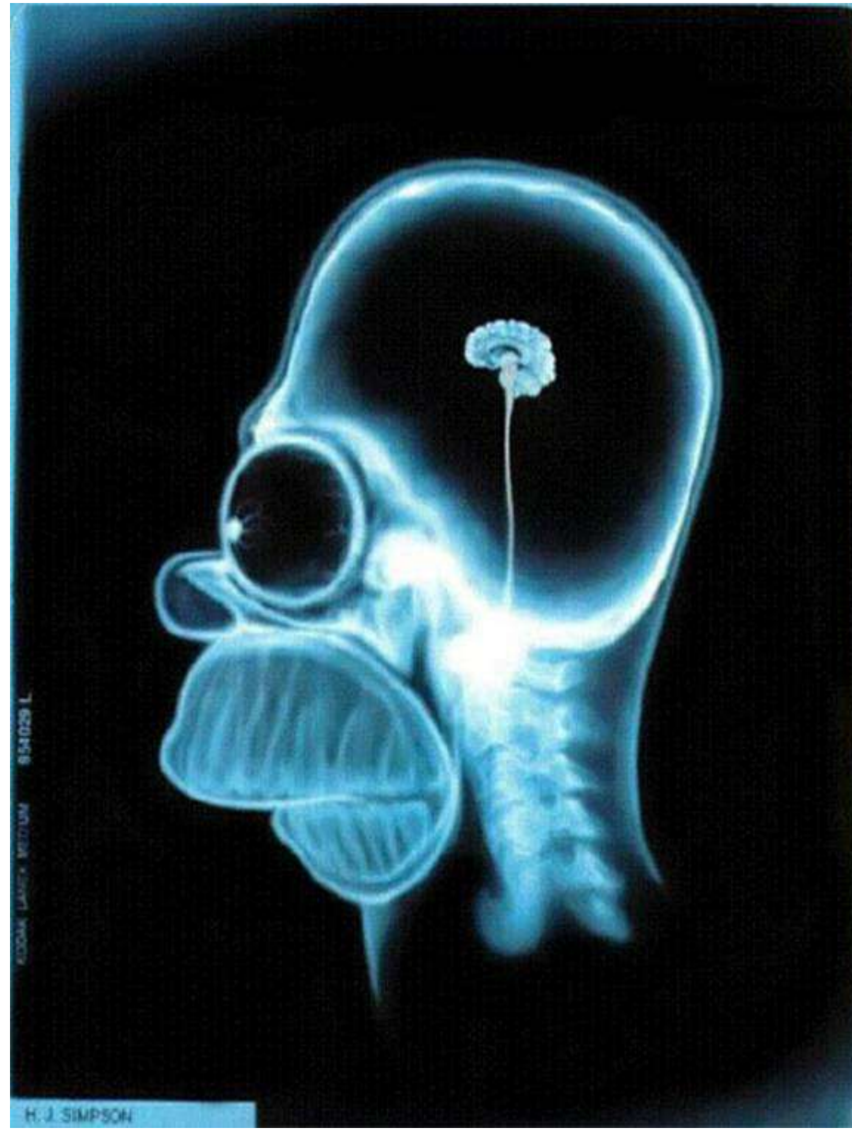
Edge case error problems will limit AI applicability until they are solved (don’t hold your breath).

The uncertainty challenge means:

- Calculate error costs and apply them to your model before (and after)
- The more risk averse, the more predictability is desire, the less variance one can tolerate



		Classifier Prediction	
		Positive	Negative
Actual Value	Positive	True Positive	False Negative
	Negative	False Positive	True Negative



The real questions

Can I support this result at a later date?

- Do I care?
  - Is the risk (cost) worth worrying about?
  - Is the cost of reproducibility less than the risk and cost?
- Do I only need to justify the decision?
  - interpretability and trust may be enough
  - Unless there are audits or legal discovery involved
- Do I need to reproduce the results?
  - May not need interpretability
  - Need a lot of other things

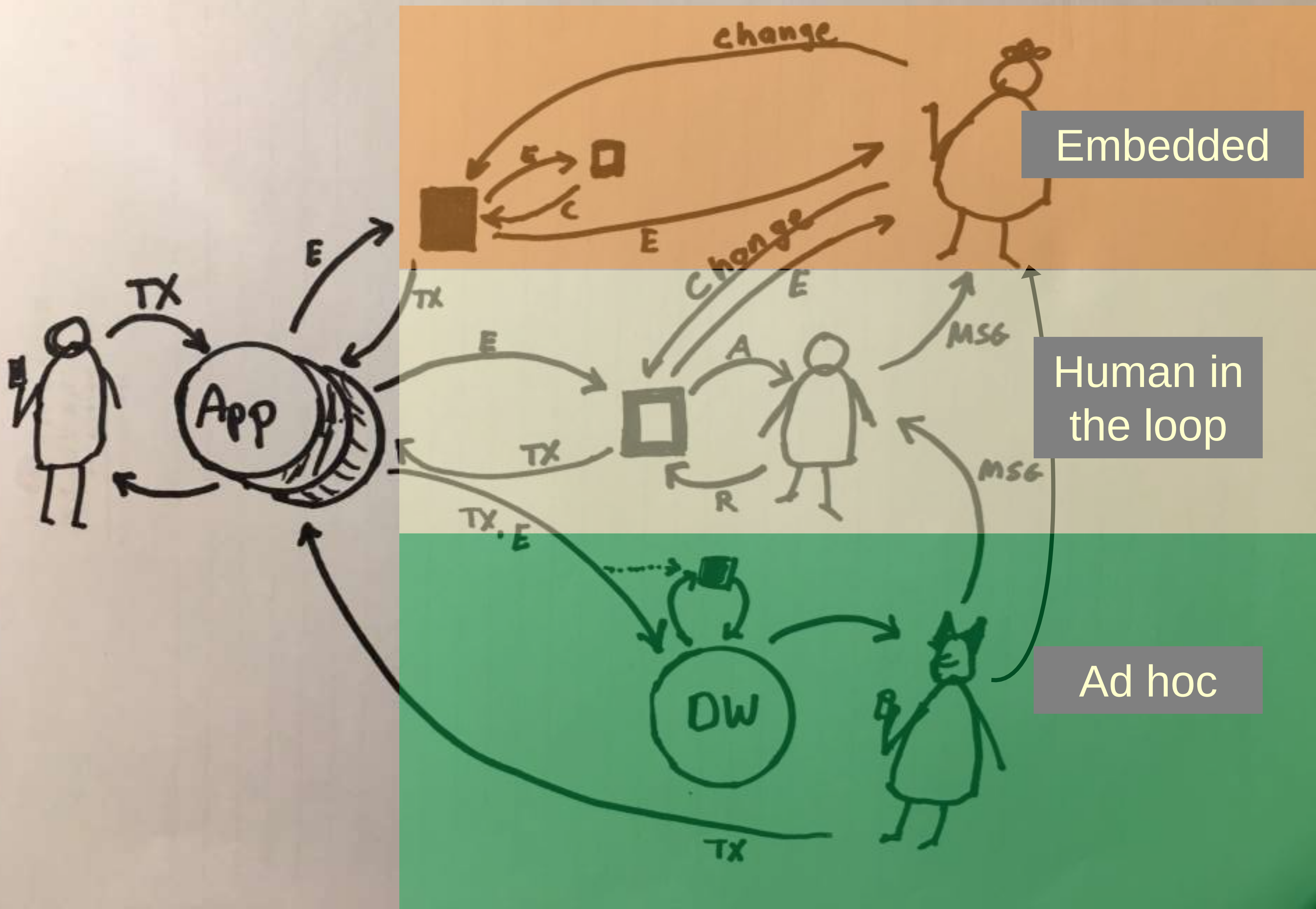
The real need is trust. Our trust is based on all the elements that are involved, not just the model.

The higher the stakes the more you must think about all the ways it could go wrong.

Reliability and robustness of the technology environment is as important as the model.

Reproducibility is not just the data scientist's problem – it is an operational concern.

Three categories of use, with differing trust levels

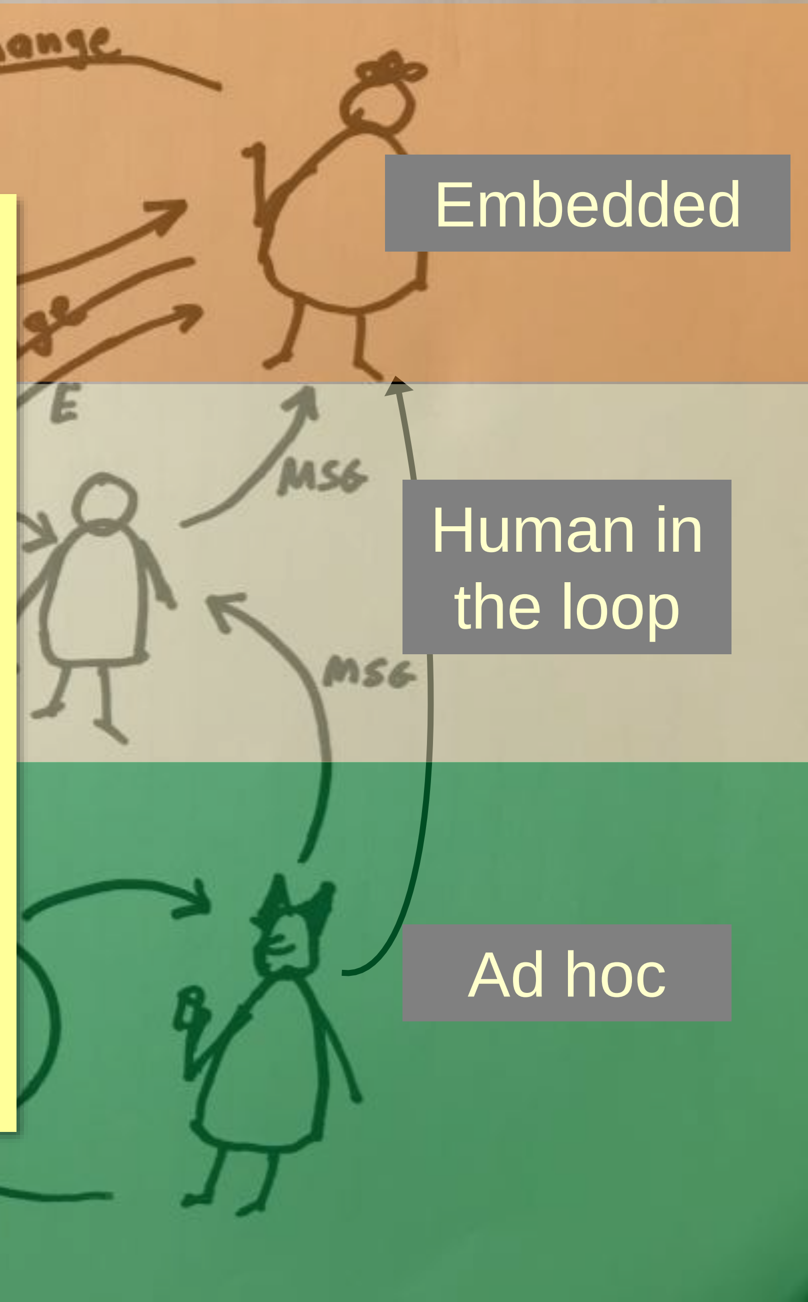


Explainability only addresses the *functioning* of the model

Three categories of use, with differing trust levels

In general, explainability is more useful when there is human agency and control in decisions.

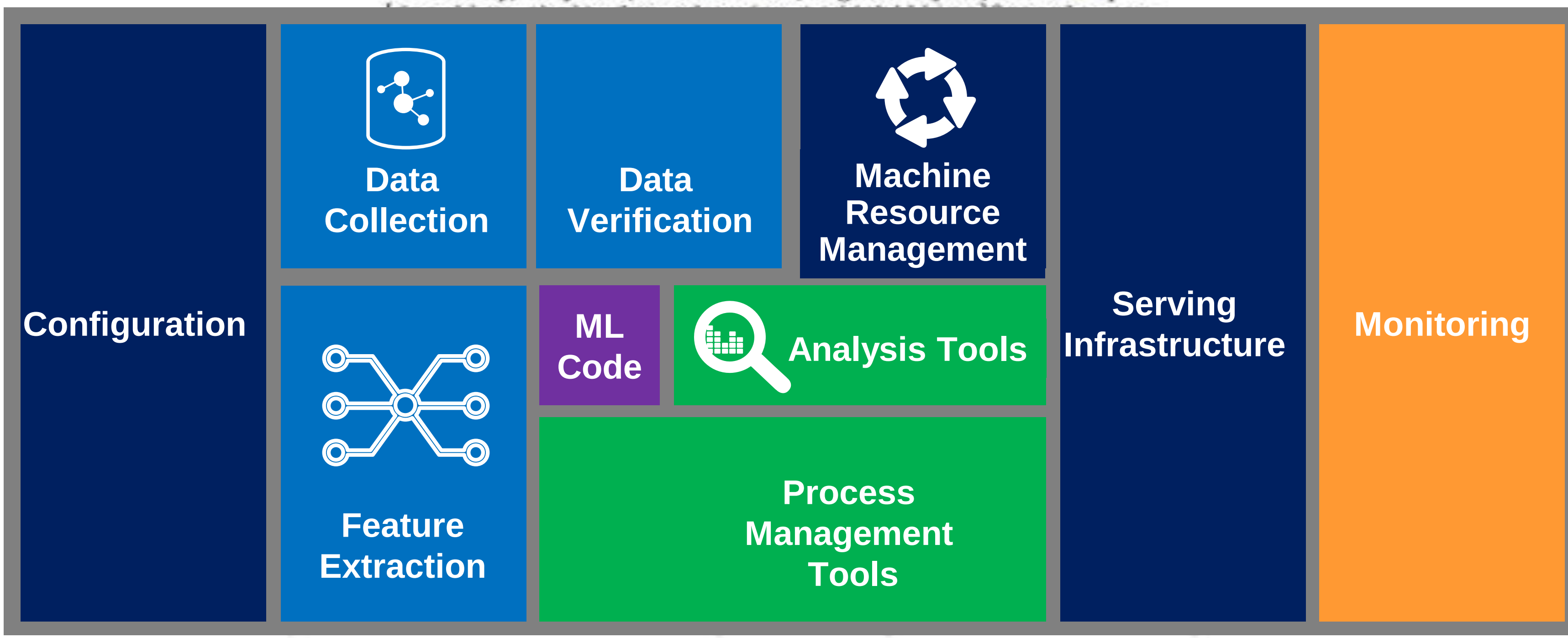
Reproducibility is increasingly important where the machine has agency because the impact of automation expands the scope and increases the complexity beyond the model.



Machine learning is the smallest part of the environment

Hidden Technical Debt in Machine Learning Systems

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips

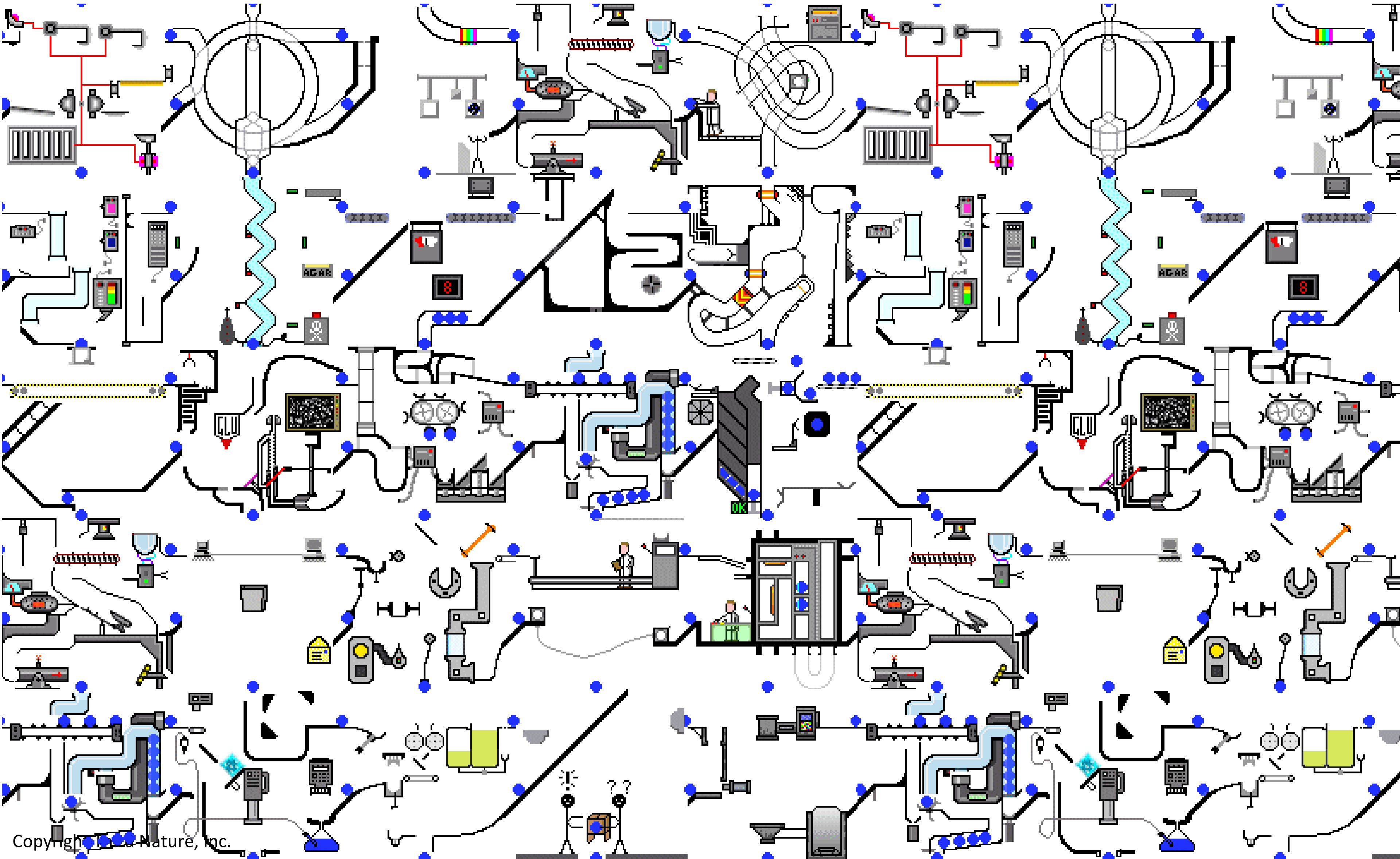


Static Analysis of Data Dependencies. In traditional code, compilers and build systems perform static analysis of dependency graphs. Tools for static analysis of data dependencies are far less common, but are essential for error checking, tracking down consumers, and enforcing migration and updates. One such tool is the automated feature management system described in [12], which enables data sources and features to be annotated. Automated checks can then be run to ensure that all dependencies have the appropriate annotations, and dependency trees can be fully resolved. This kind of tooling can make migration and deletion much safer in practice.

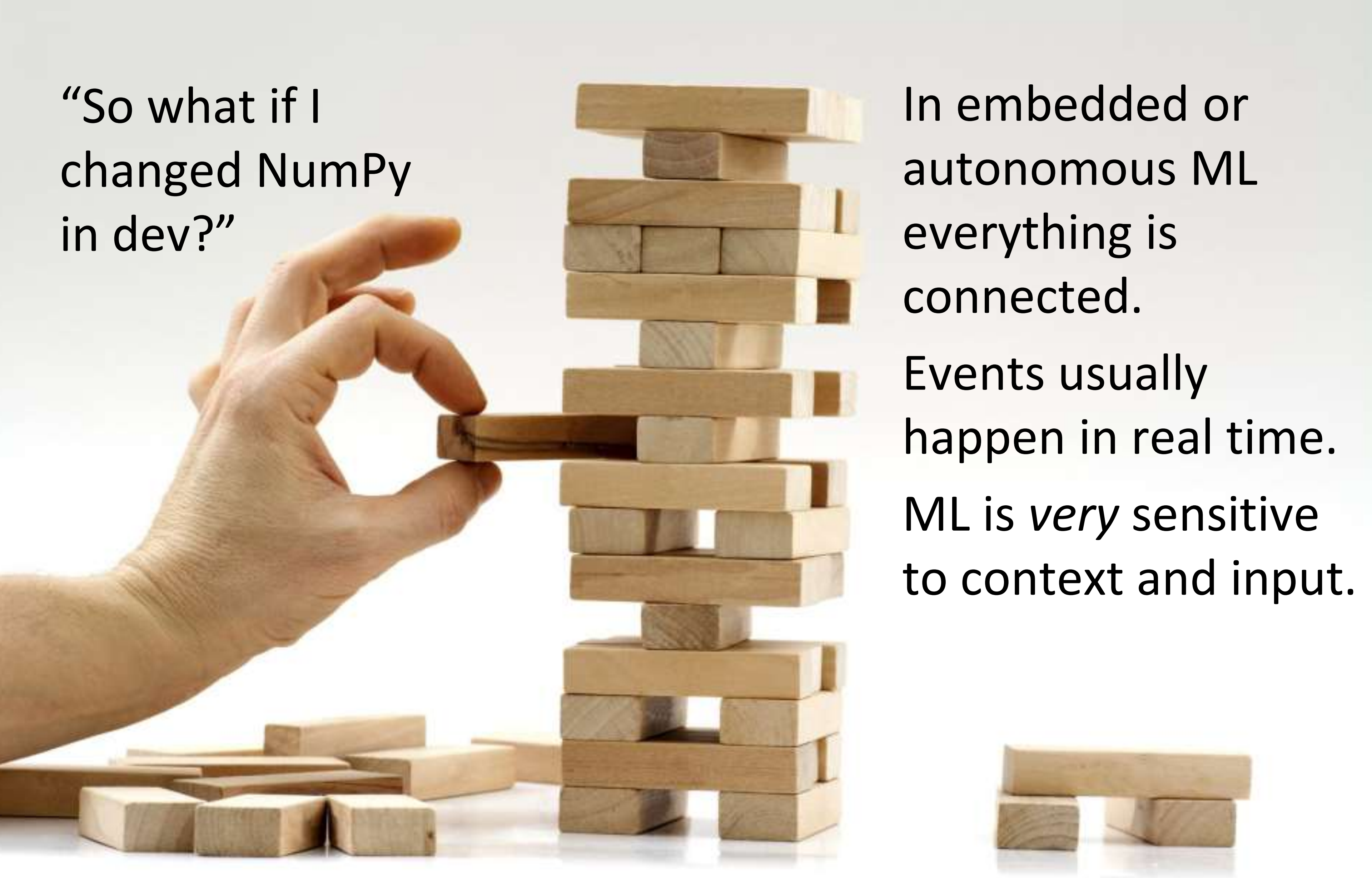
https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems

The end result in the data science landscape is complexity

You can explain the model, but can you explain this?



ML Principle: CACE, Change Anything Change Everything



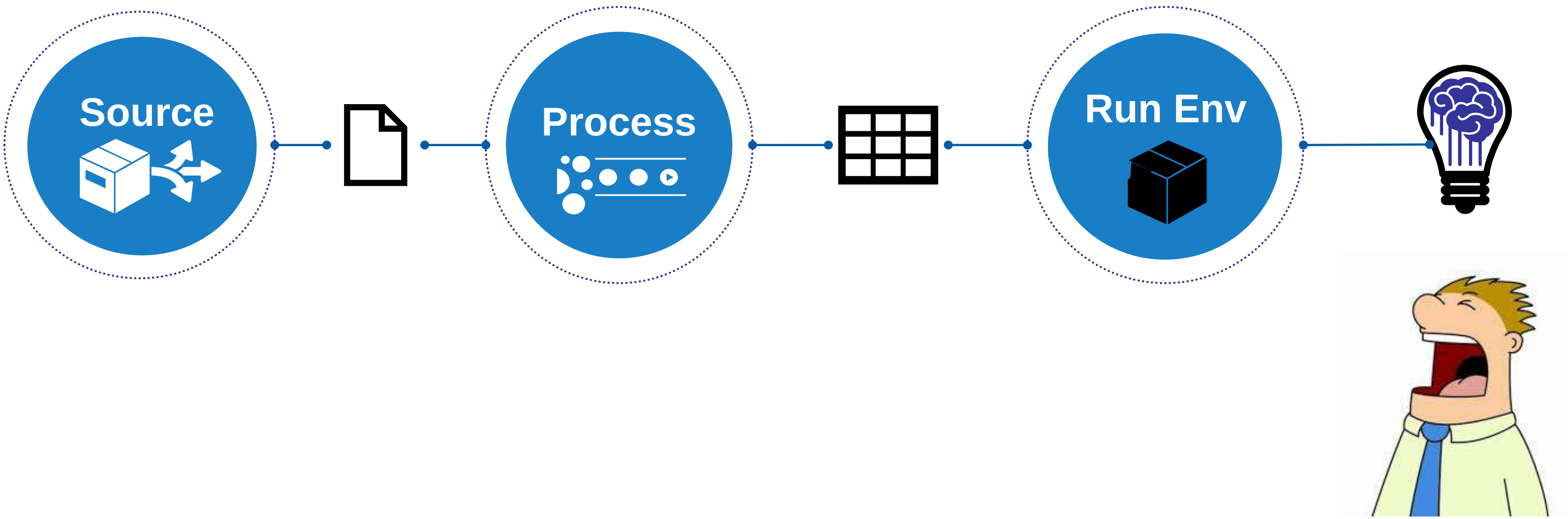
"So what if I changed NumPy in dev?"

In embedded or autonomous ML everything is connected.

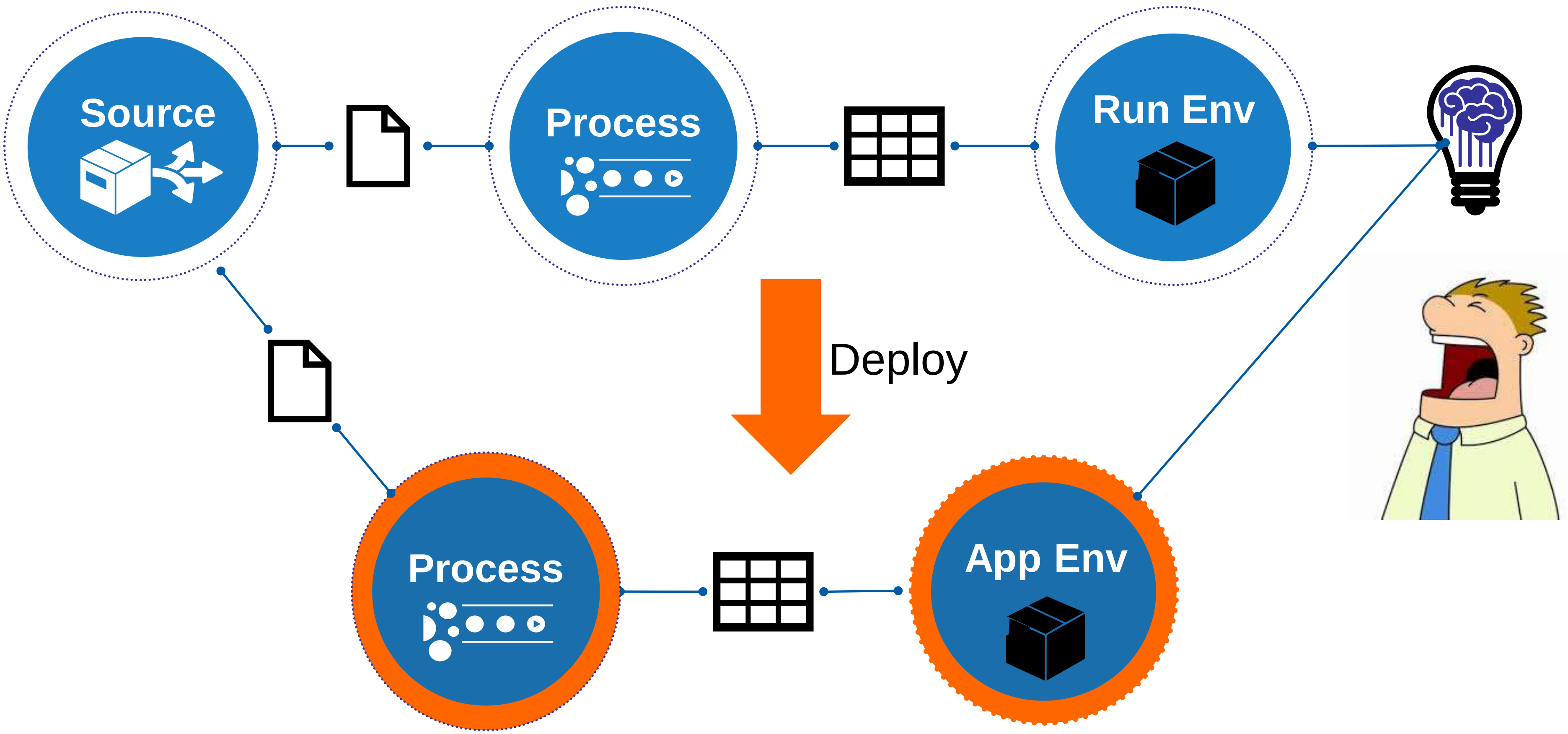
Events usually happen in real time.

ML is *very* sensitive to context and input.

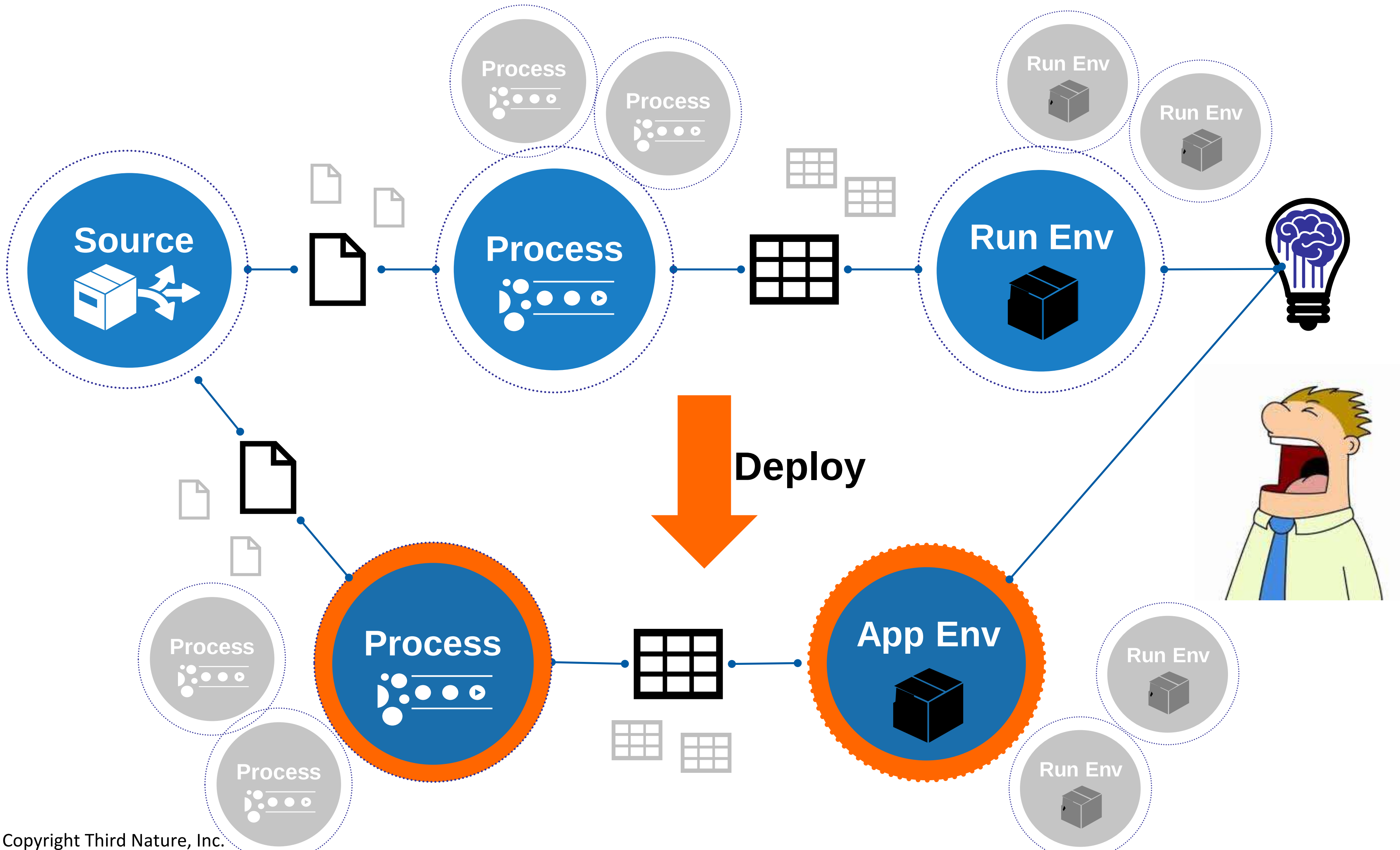
Working backwards from the decision or answer, what do you need to have reproducibiity?



Fine for a single-run model. What if it's embedded in an application somewhere?



If a model is used over time, you will likely change it. Or sources, or processes will change.



We’re so focused on the light switch that we’re not talking about the light.

Managing the code without managing the data?

Most emphasis in the industry is on code and code artifacts:

- Model repositories
- Model management
- Pipeline frameworks
- Packaging
- Versioning
- Tools

Why? Because vendors want to sell you products for the problem they helped create.

What do the experts say?

TIDY DATA: Hadley Wickham makes the case for Tidy data sets, that have specific structure, are easy to work with, that free analysts from mundane data manipulation chores – there's no need to start from scratch and reinvent new methods for data cleaning

Source: Tidy Data by Hadley Wickham, Journal of Statistical Software, Vol 59, issue 10 (2014)

https://vita.had.co.nz/papers/tidy-data.html

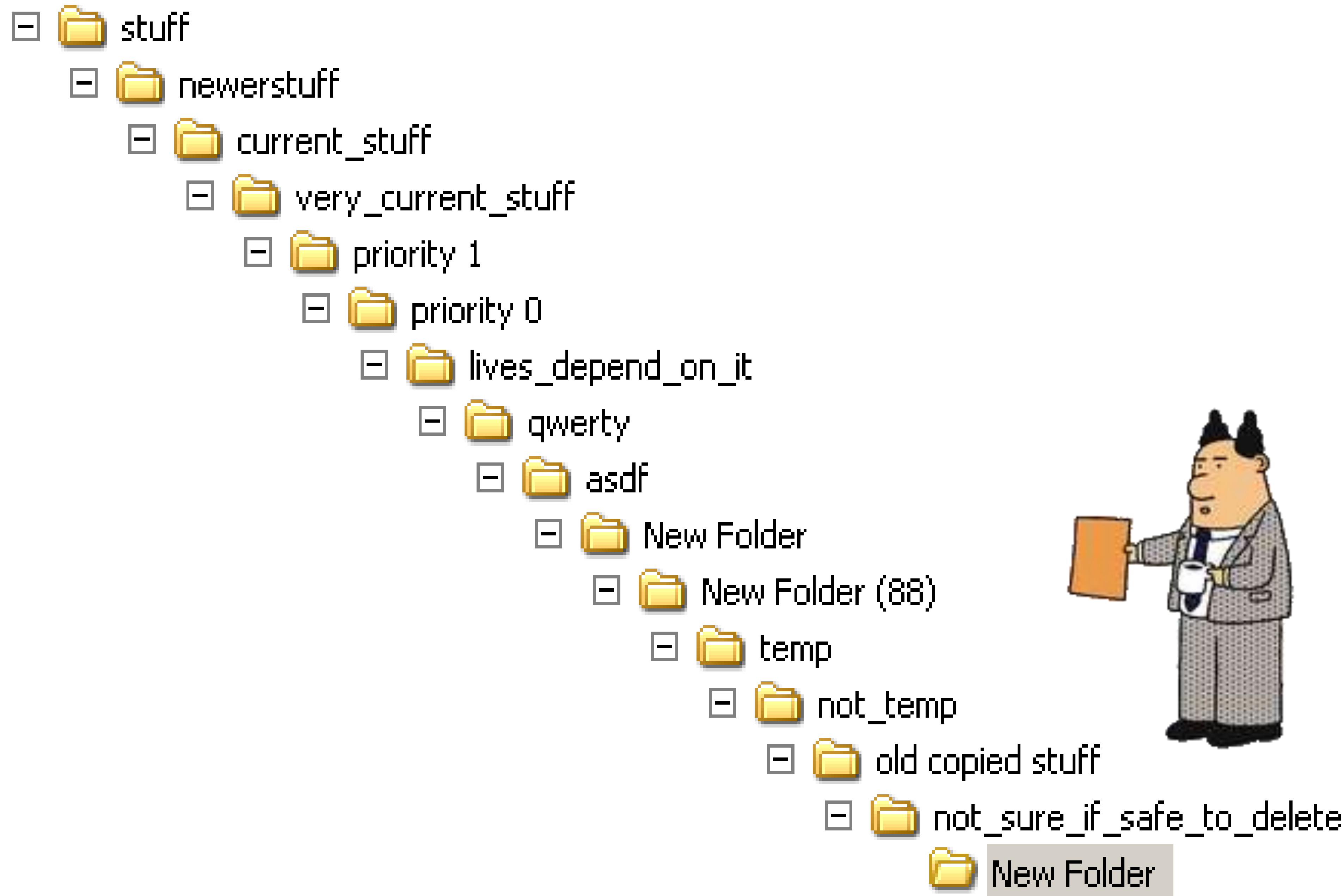
What about AI? GIGO!

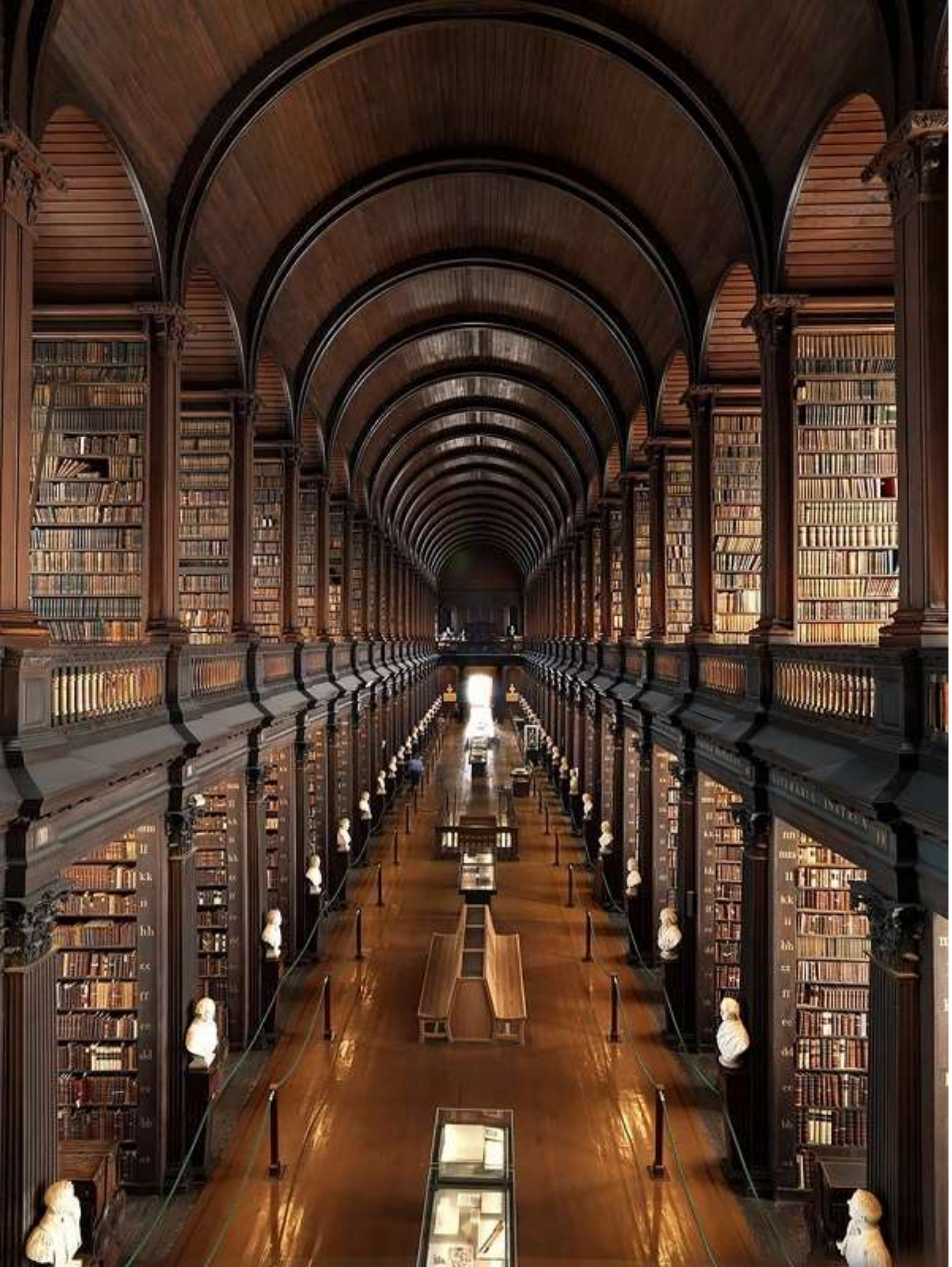
"If your boss asks you, tell them that I said build a unified data warehouse"

Andrew Ng
Leading AI Researcher

Everyone wants shortcuts. There's aren't any shortcuts

Reproducibility? No problem. Just save versions of data





Today’s market solution:

the Data Lake*

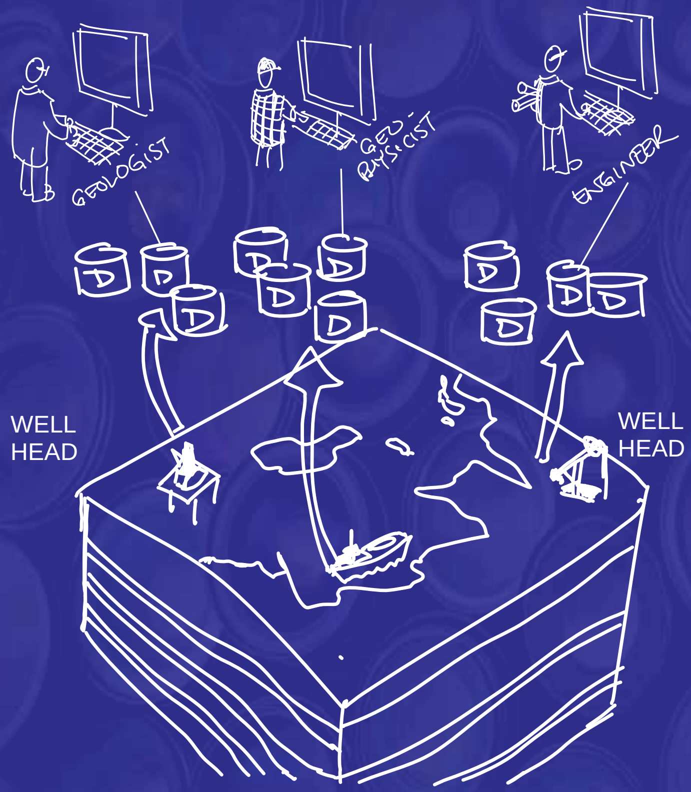
**Depiction for illustrative purposes only, no warranty express or implied, actual system may vary.*
By a lot.

Today's market solution:

the Data Lake + DIY

Data hoarding is not a data management strategy

A standard data science approach (to avoid)



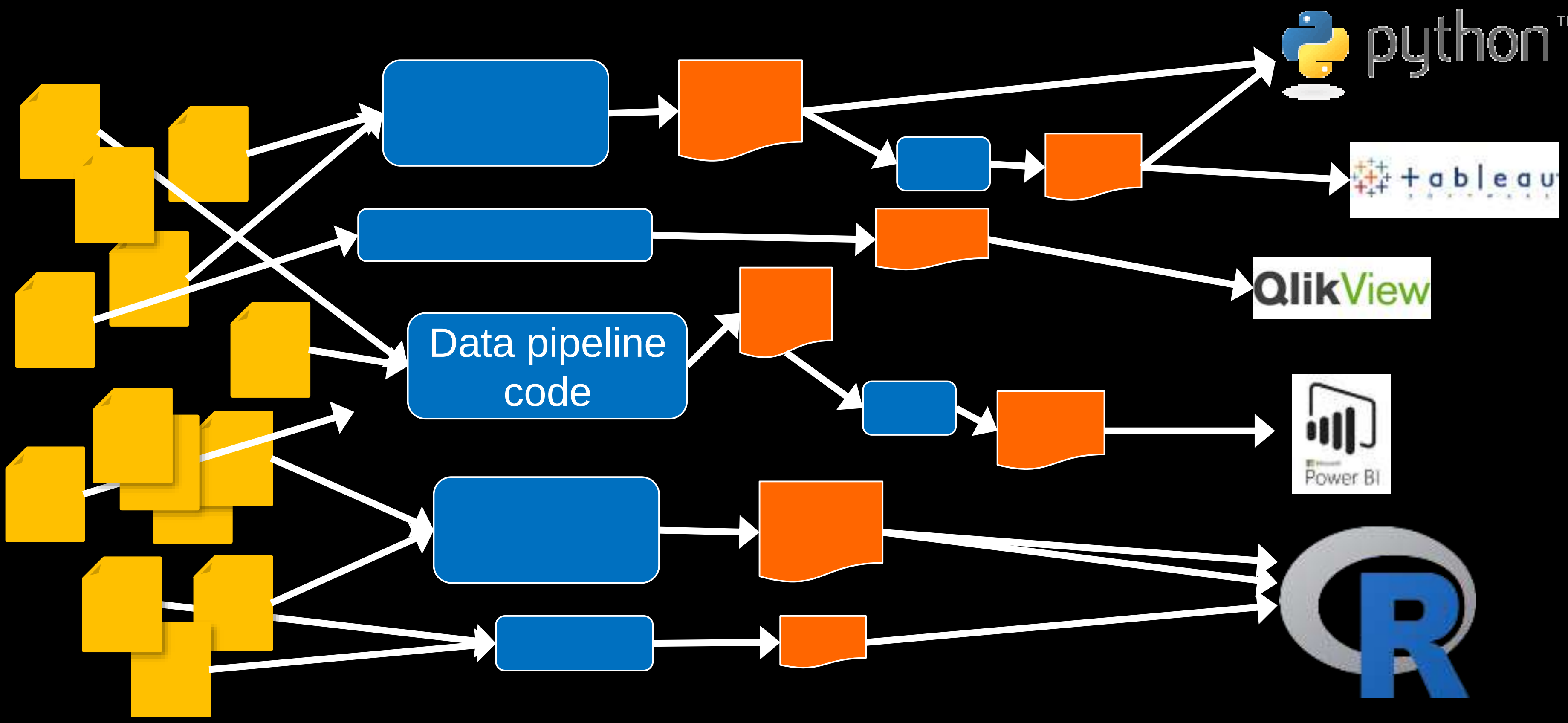
Example use cases

One-Pipeline-
Per-Process

Redundant Effort /
Cost / Complexity /
etc.

A dystopian model: Lake + data engineers + pipelines

The Lake with pipelines to files or tables for each model is exactly the same pattern as mainframe COBOL batch



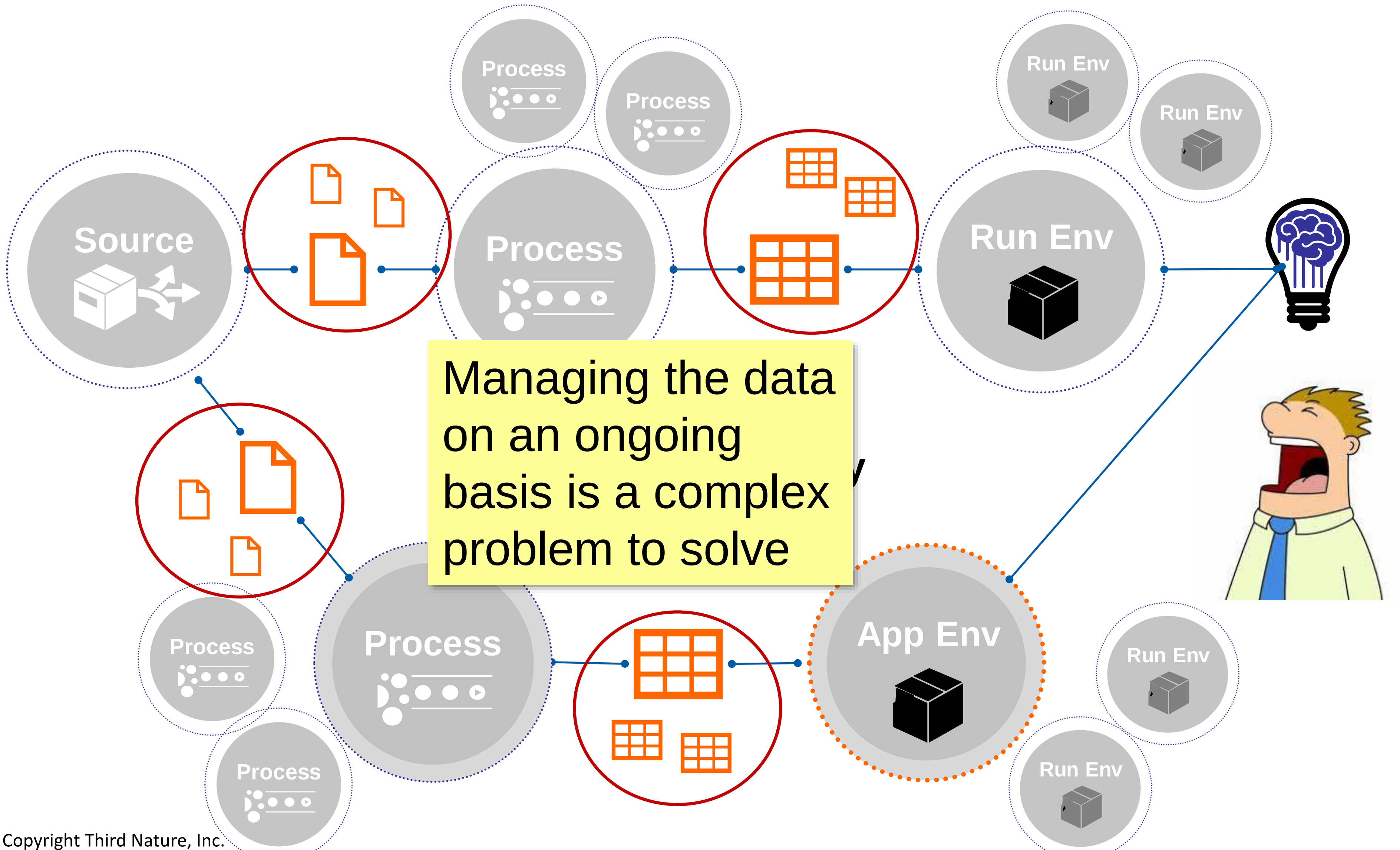
We already know that people don't scale. Don't do this

Self-service tradeoffs

Pay now or pay later, but you will always pay. The question is which payment will be less.

Self-service gives flexibility and agility, but can reduce repeatability and add duplicated effort and conflicts, and an increase in risk.

But that self-serve bit only applies to building models. Embedding them in the enterprise is more involved



Managing the data on an ongoing basis is a complex problem to solve

The organization-wide focus needs to be on repeatability – where it can be supported

Separate the uses from the infrastructure – focus on the city, not the buildings

Buildings above: flexibility, repurposing, faster change above, funded independently

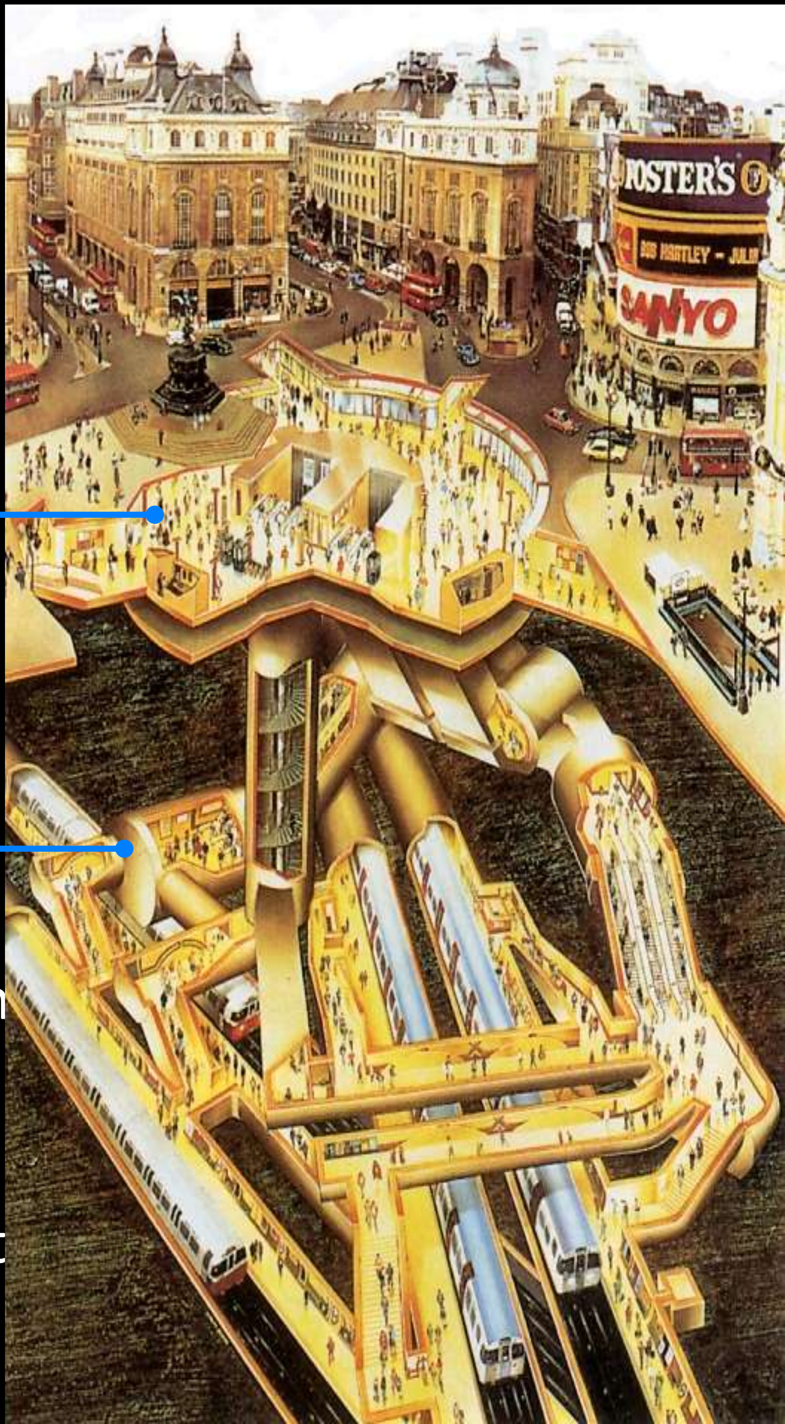
Applications

Utilities below: stability, reuse, slow predictable change below, funded centrally

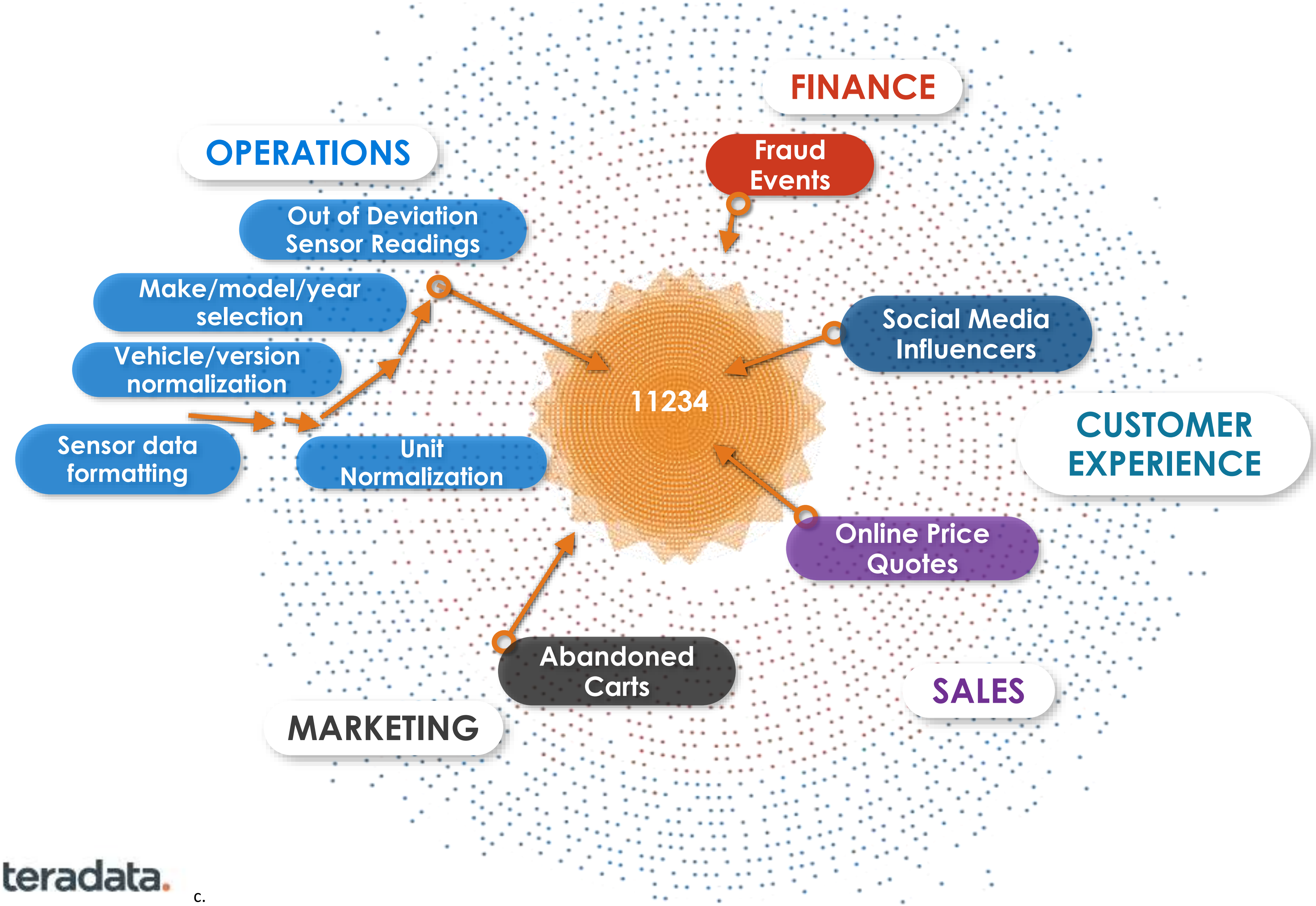
Infrastructure

The infrastructure is a hidden combination of technology, process, and methods.

There is a need to draw boundaries for responsibilities in the organization. It can't be all business or all IT.



Data pipelines, data architecture, and curation



teradata.

c.

Data curation is not data modeling

The problem with so many sources, types, formats and latencies of data is that it is now impossible to create in advance one model for all of it.

Data modeling is about the *inside* of a dataset. **Curation is about the entire dataset**.

It's about: creating, labeling, organizing, finding, navigating, retiring.

Data curation, rather than data modeling, is becoming the most important data management practice.

Data can be maintained at multiple levels: not raw or DW

Ingredients

Goal: available

User needs a recipe in order to make use of the data.

Pre-mixed

Goal: discoverable and integrateable

User needs a menu to choose from the data available

Meals

Goal: usable

User needs utensils but is given a finished meal

Data architecture dictates technology and process

Collection

Capture metadata (including request)
Record the structure
Apply keys
PII masking, restrict
Start lineage

Distribution

Common structures

RDM and MDM

Subject models

Make data findable and linkable

Data provisioning

Consumption

Model for target use

Quality rules

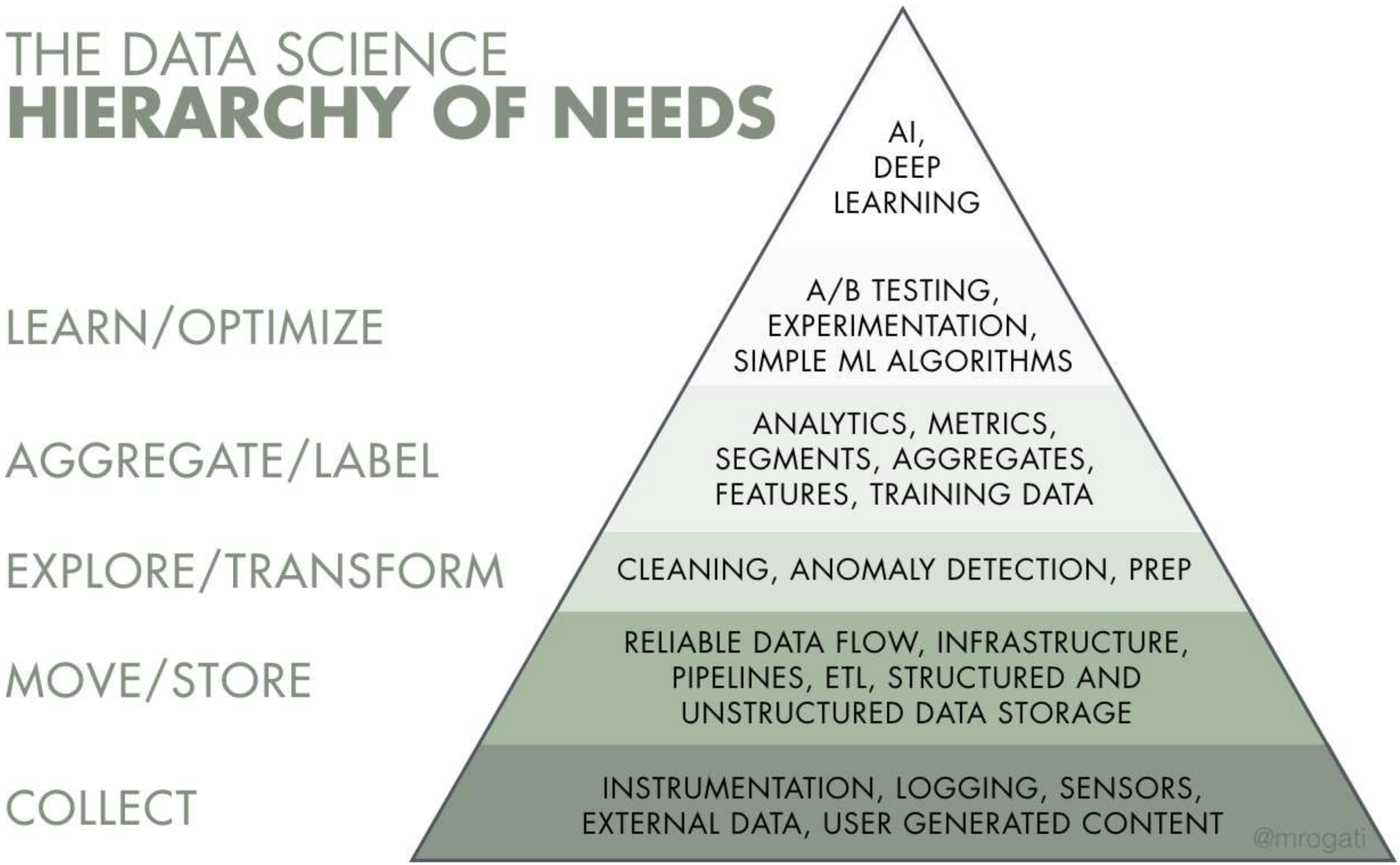
Track provenance

Apply SLAs

As few engines as possible – not fewer

Decide on policies for when to place data in which area

ML has a lot of data requirements: no shortcuts



https://hackernoon.com/the-ai-hierarchy-of-needs-18f111fcc007

Manage your data (or it will manage you)

Data is the foundation of the models, and the results. Focus on managing data as well as the code and tool artifacts.

Reproducibility, and data science in general, requires that you treat the operating and the build environments as a whole.

Data science in the long term is not about individual projects, but organizational capability. This means infrastructure, process, governance.

Things you can do

1. Determine what level of risk is acceptable based on the costs of false positives and false negatives, or poor accuracy.
2. Manage the data history such that you can always reconstruct it.
3. Think through the dependencies outside the model that affect reproducibility.
4. Track the configuration of *everything* in the chain of dependencies.
5. Do not let a thousand flowers bloom with BYOT. You will have to cut many of them due to complexity.
6. Do not let IT overcomplicate your environment with technology because of belief things must be big. Brittle infrastructure can’t be tolerated with embedded machine learning.
7. Make a rational decision about how much effort to expend.

Mark Madsen

Mark Madsen is a Fellow at Teradata in the Technology and Innovation Office. he focused on the data and analytics ecosystem, problems of large-scale design, and R&D.

Prior to that he was president of Third Nature, a research and consulting firm focused on analytics, integration and data management.

Mark is an award-winning author, architect and CTO whose work has been featured in numerous industry publications. He received awards for his work from the American Productivity & Quality Center and the Smithsonian Institute. He is an international speaker, chairs several conferences and program committees.