Hewlett Packard Enterprise

IT'S NOT ABOUT THE ALGORITHM, IT'S ABOUT THE LOGISTICS.

Ted Dunning

March 4, 2020

CONTACT INFORMATION

Ted Dunning, PhD

Chief Technologist, HPE
Committer, PMC member, board member, Apache Software Foundation
O'Reilly author

Email tdunning@mapr.com tdunning@apache.org
Twitter @Ted_Dunning

SOME COOL PROJECTS I HAVE WORKED ON

- Dynamic pricing, media recommendations
  - Resulted in 3x revenue enhancement
- Autonomous car machine learning systems
  - System now used by automakers around the world
- Unique identity system for India (Aadhaar)
  - Increased the population in India who can participate economically by 50%
- Fraud detection and prevention systems
  - Single model saved 70 million dollars first year, others have saved billions

The biggest thing I have learned:

The biggest thing I have learned:

Doing more smart stuff

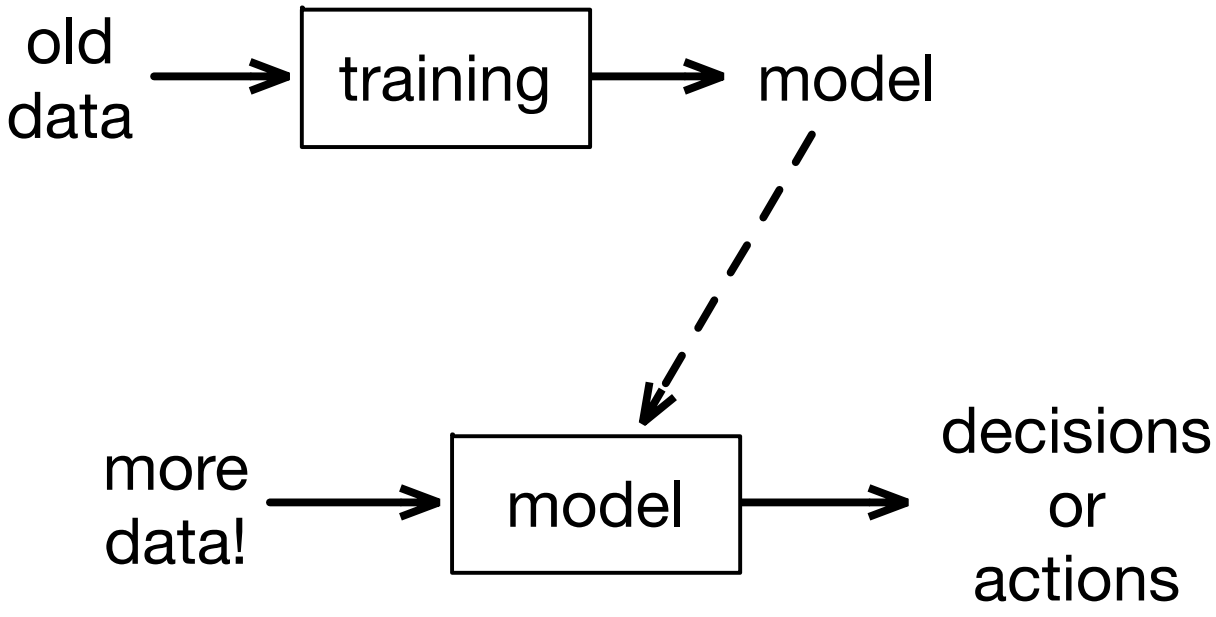
The biggest thing I have learned:

Doing more smart stuff
isn’t as useful
as doing less stupid stuff

Let’s see why and how

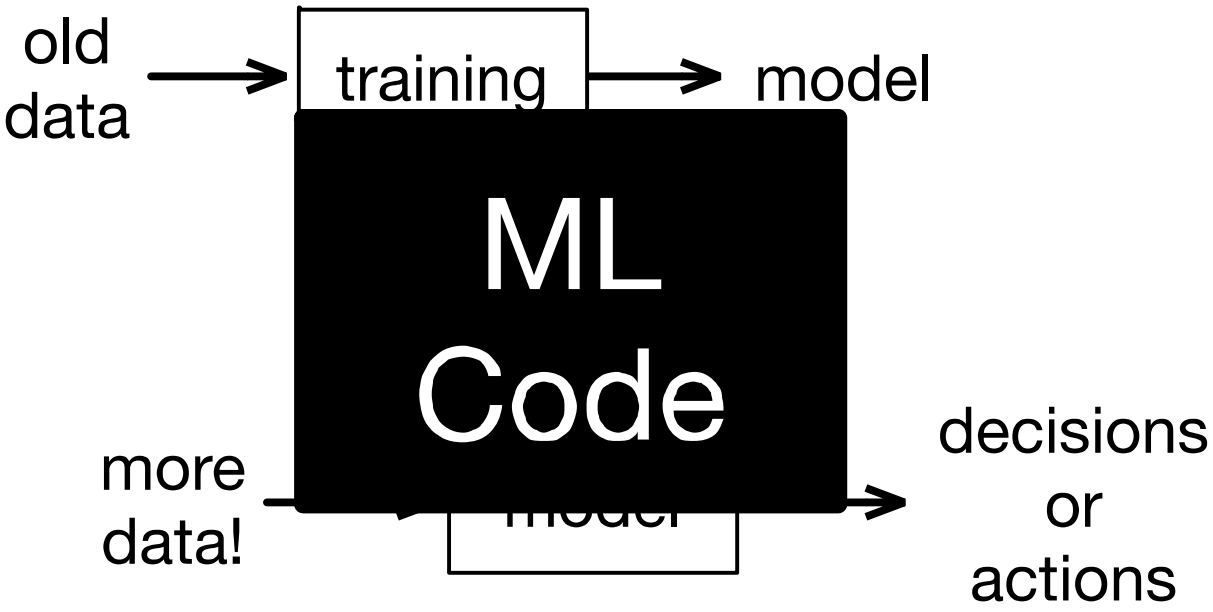
TRADITIONAL VIEW

TRADITIONAL VIEW: THIS ISN'T THE WHOLE STORY

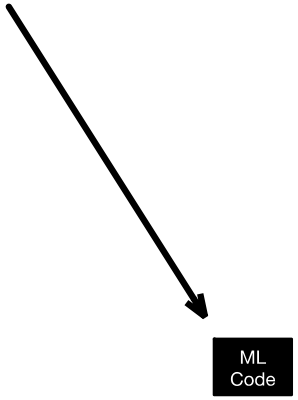


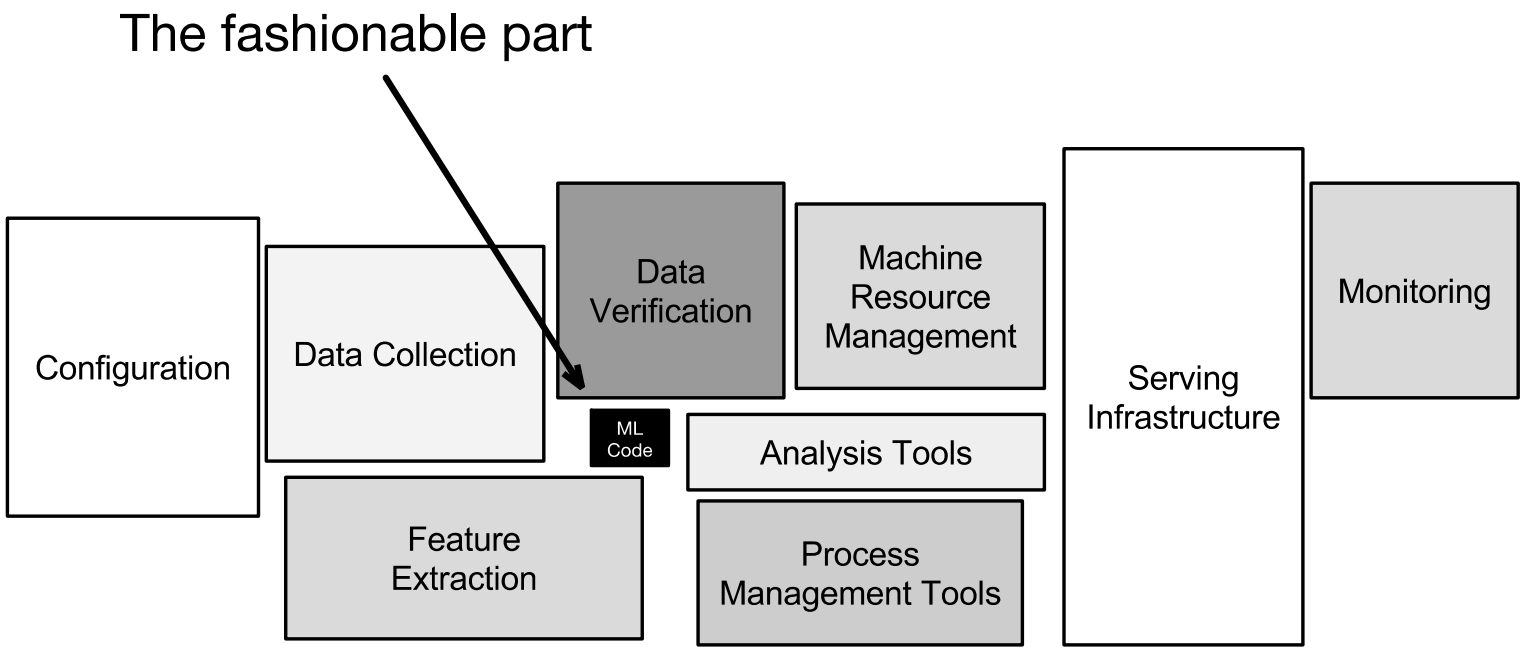
90% of the effort in successful machine learning isn’t the learning ...

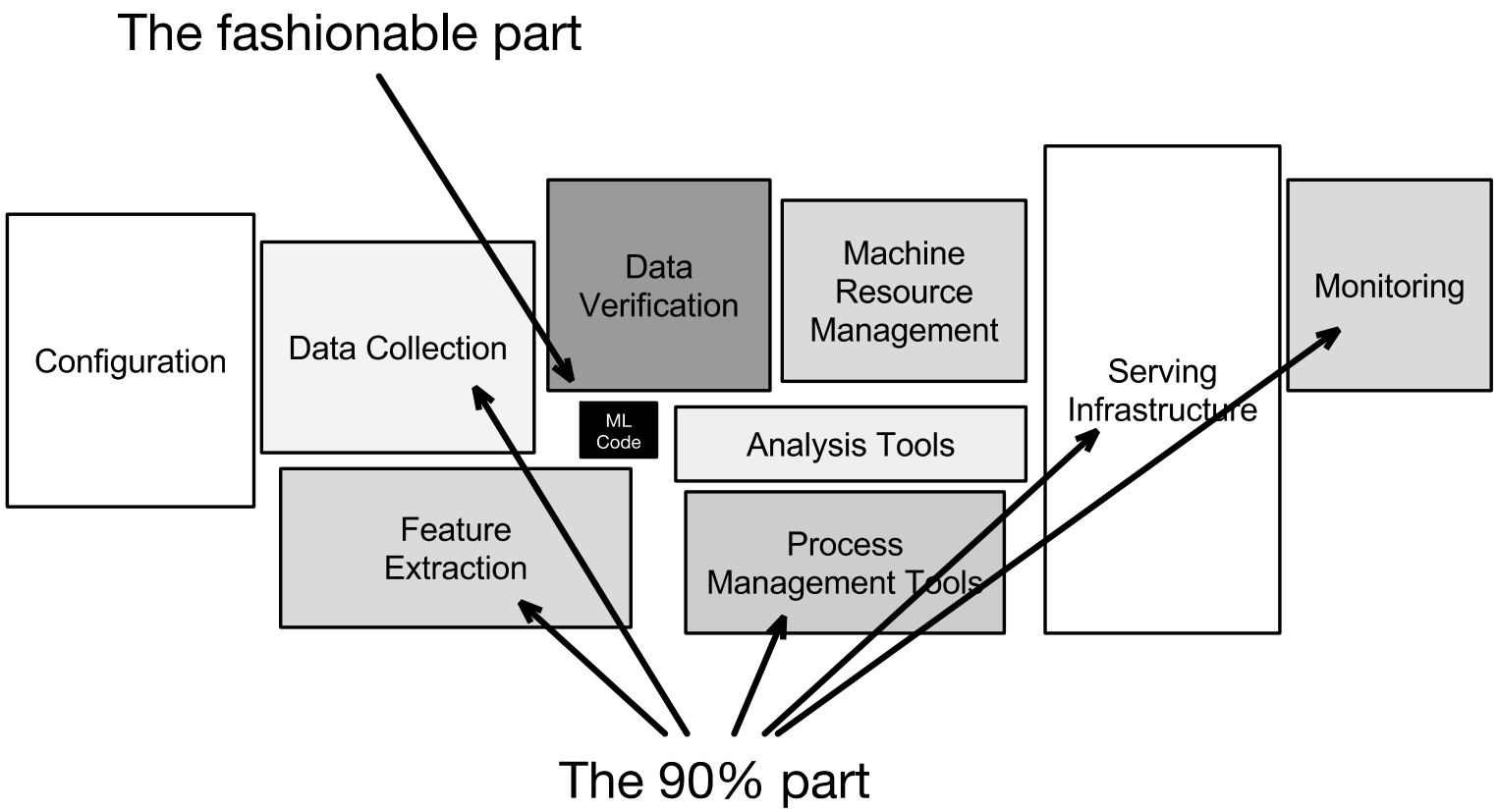
It’s the logistics

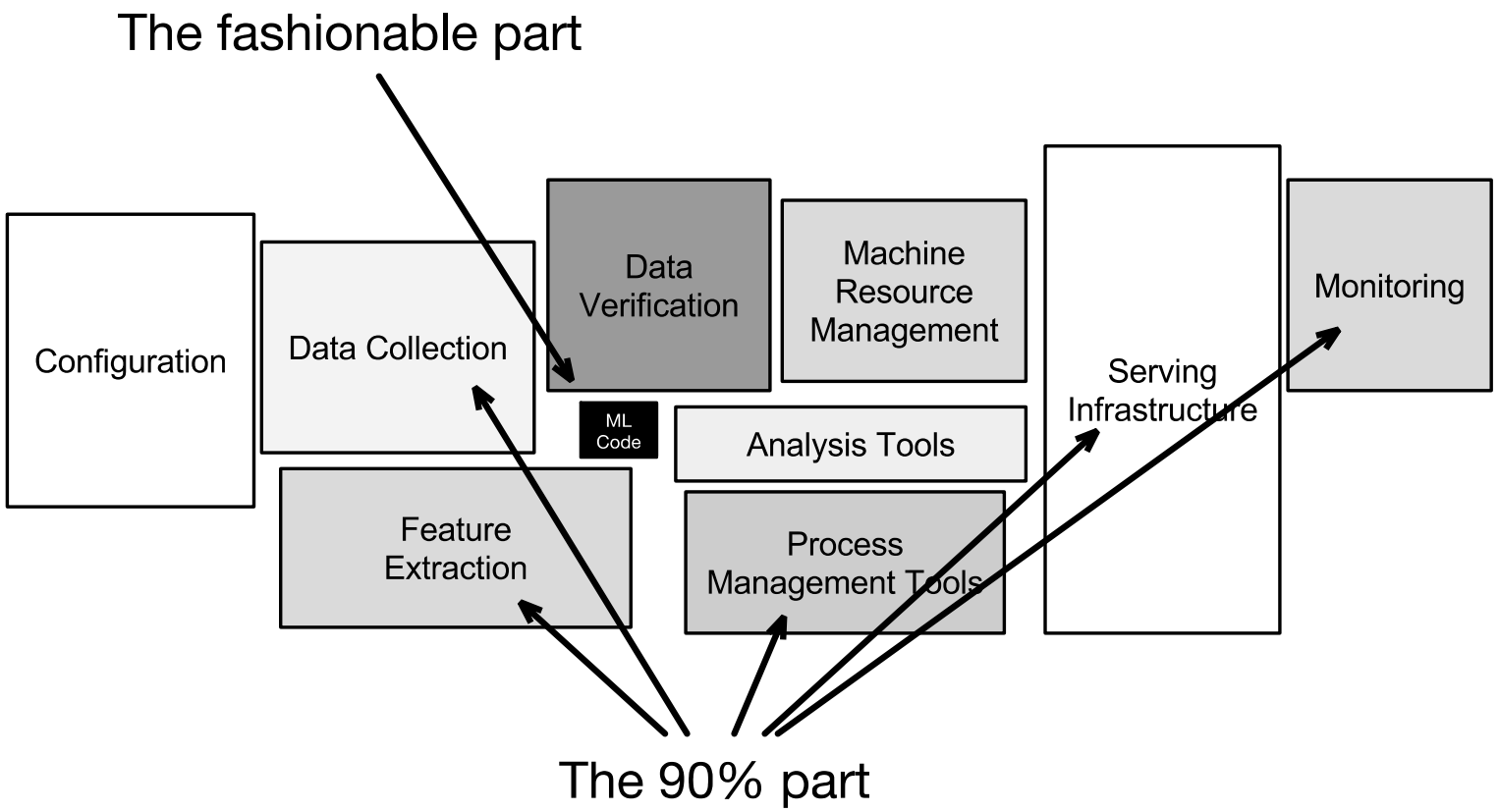

old
data
training
model
ML
Code
more
data!
decisions
or
actions

The fashionable part

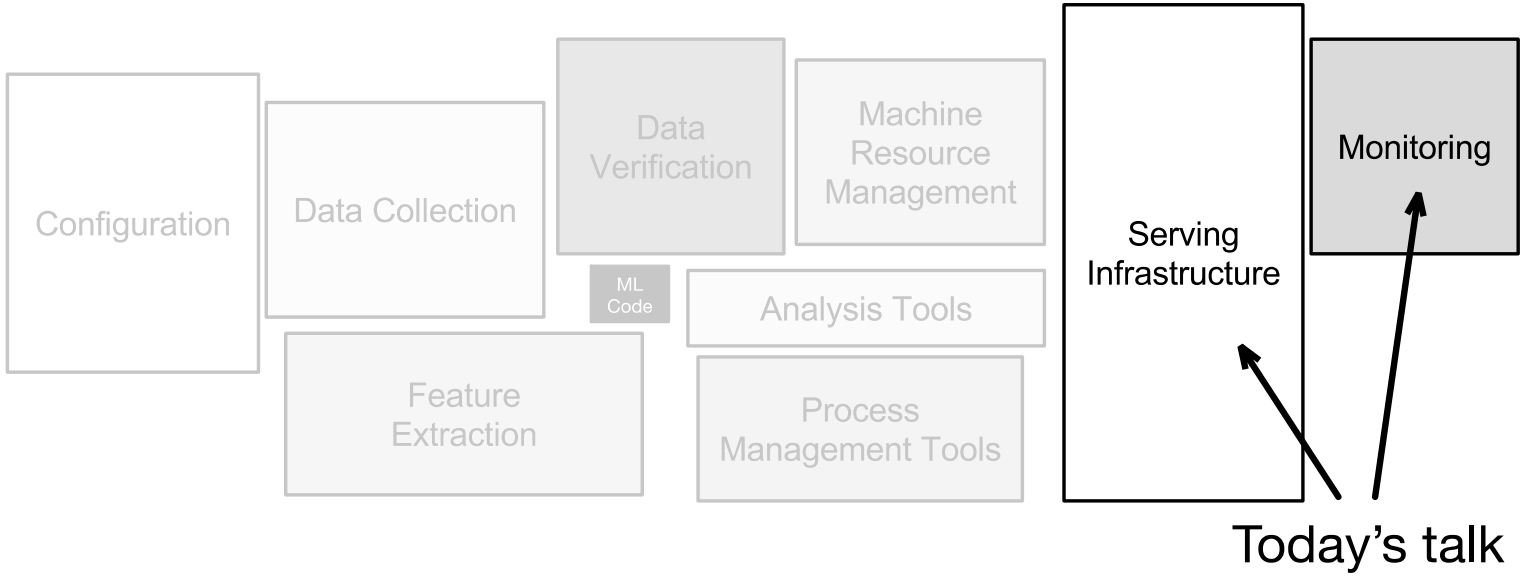



The fashionable part
Configuration
Data Collection
Data Verification
Machine Resource Management
Serving Infrastructure
Monitoring
ML Code
Analysis Tools
Feature Extraction
Process Management Tools


The fashionable part
Configuration
Data Collection
Data Verification
Machine Resource Management
Serving Infrastructure
Monitoring
ML Code
Analysis Tools
Feature Extraction
Process Management Tools
The 90% part

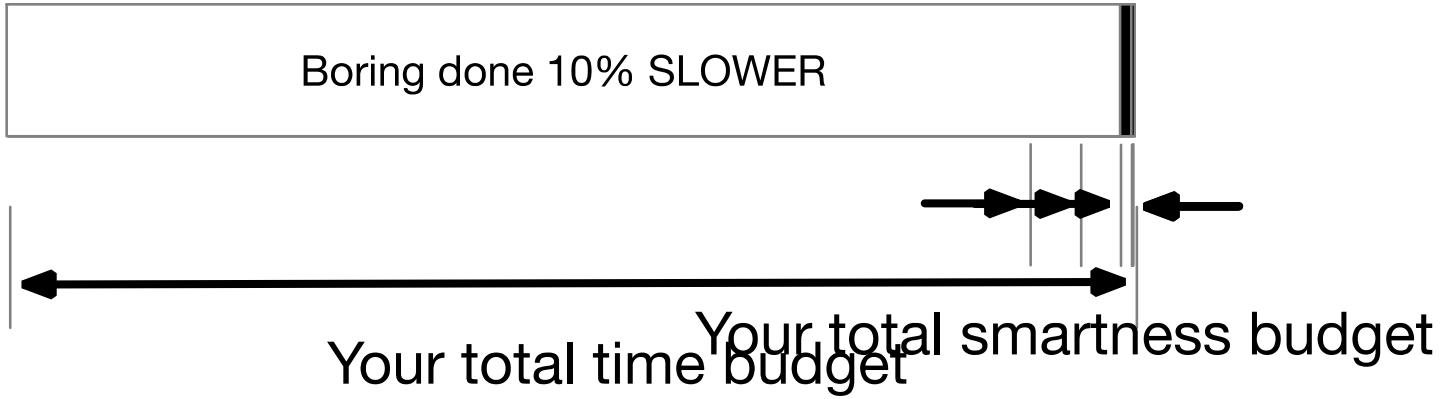


Hidden Technical Debt in Machine Learning Systems
https://papers.nips.cc/paper/5656-hidden-technical-debt-in-machine-learning-systems.pdf


Configuration
Data Collection
Data Verification
Machine Resource Management
ML Code
Analysis Tools
Feature Extraction
Process Management Tools
Serving Infrastructure
Monitoring
Today's talk

So why focus on the boring bits?

So why focus on the boring bits?



So why focus on the boring bits?

So why focus on the boring bits?

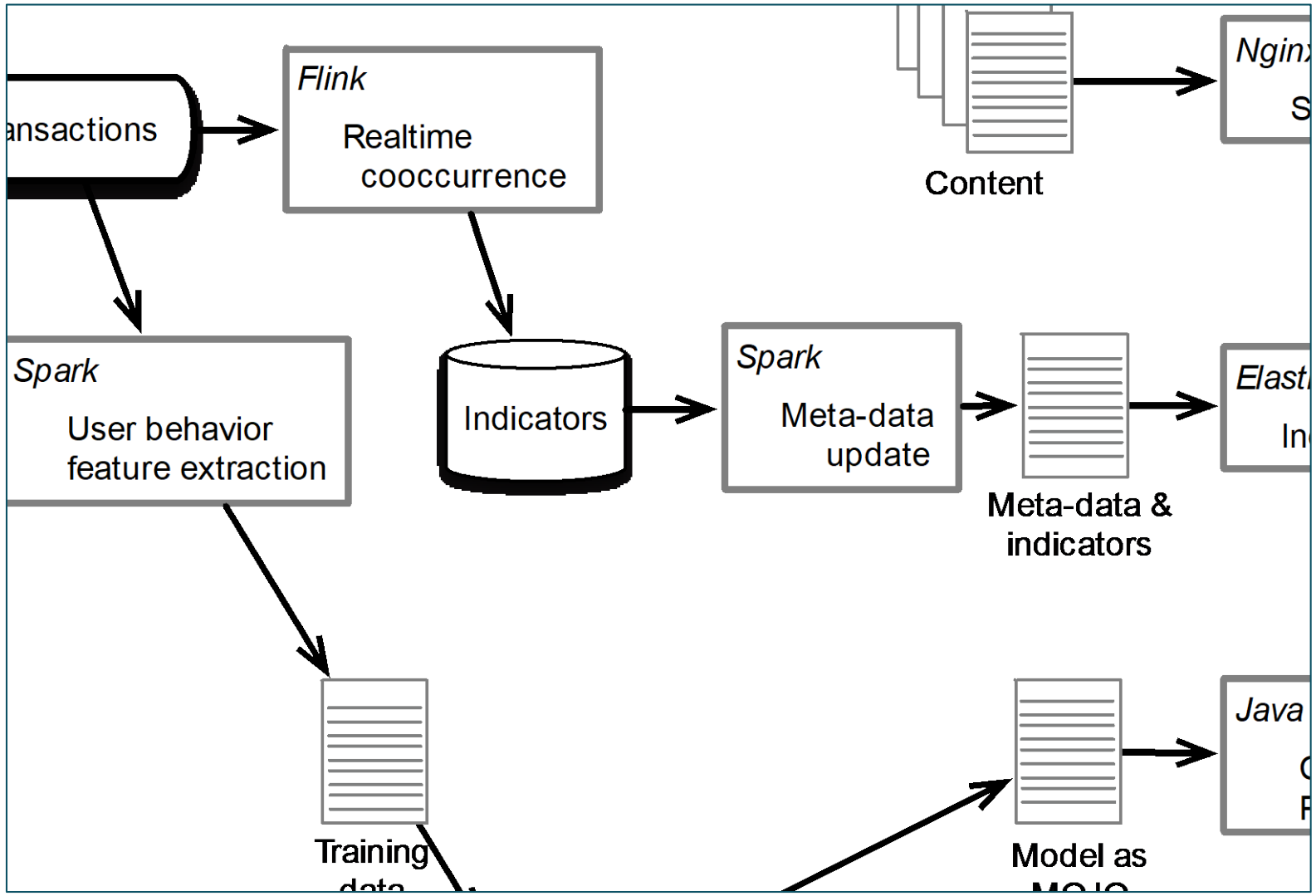
Because they will eat the clever bits!

So why focus on the boring bits?

Because they will eat the clever bits!

The logistics are *harder* with machine learning

MACHINE LEARNING
IS (JUST)
A NEW WAY TO CODE



Lots of modern software is based on data flows

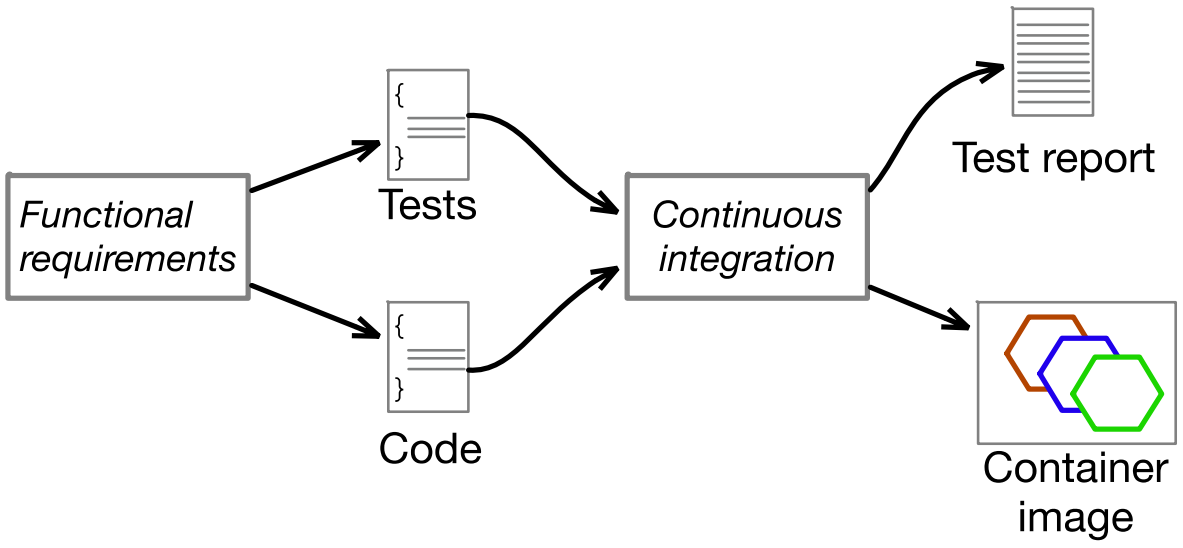
Data comes in

It gets munged and transformed

Maybe persisted

And then combined, munged and transformed again

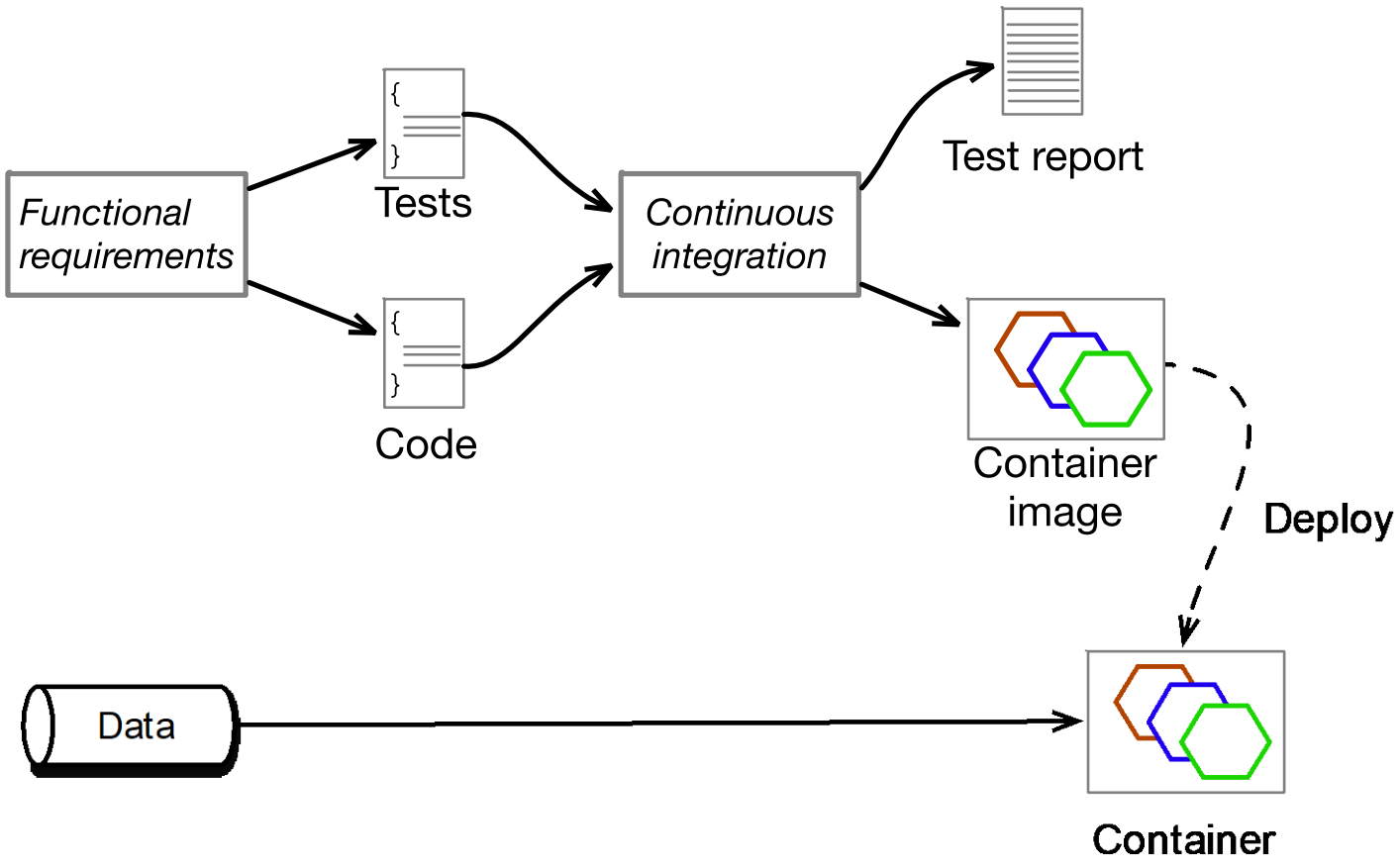
(rinse, repeat)



Software development, continuous integration and deployment are really just data flows

Requirements are turned into tests and code

Tests are run, code is packaged



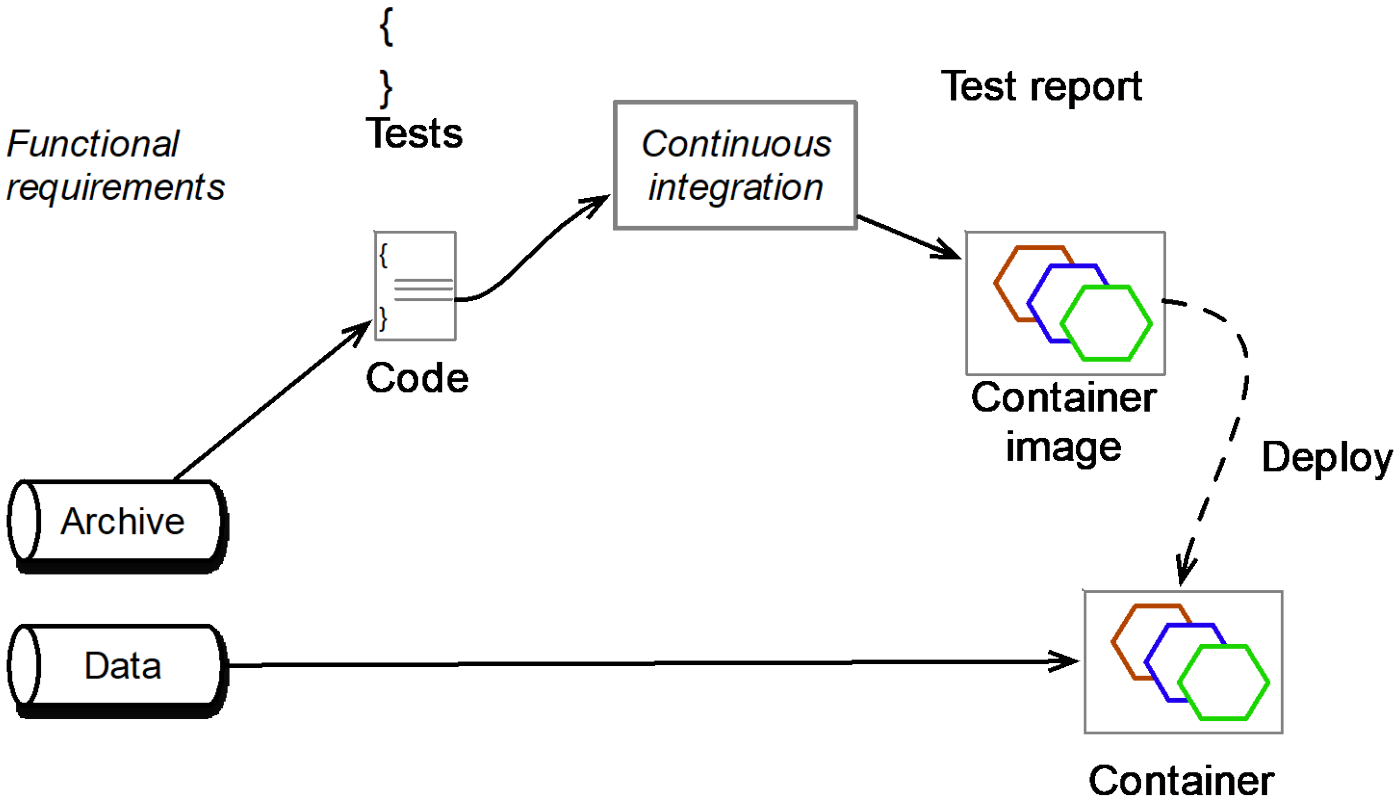
Software development, continuous integration and deployment are really just data flows

Requirements are turned into tests and code

Tests are run, code is packaged

And services deployed

To process data



With machine learning, we don't have a spec

We don't know what the software should do. That makes tests impossible

So we learn the function from historical data

This is fundamentally different from normal code development

But also much the same

MACHINE LEARNING IN A NUTSHELL

Machine Learning is just estimating functions from data

These functions can

- act,
- predict,
- estimate, or
- introspect

MACHINE LEARNING IN A NUTSHELL

Machine Learning is just estimating functions from data

These functions can

- stop a fraudulent purchase,
- predict machine life,
- nowcast current weather, or
- show possible fault causes

MACHINE LEARNING
IS (FINALLY)
GOOD ENOUGH TO BE SIMPLE

MACHINE LEARNING
IS (FINALLY)
GOOD ENOUGH TO BE SIMPLE

BUT THE LOGISTICS HAVEN'T CAUGHT UP

MACHINE LEARNING
IS (FINALLY)
GOOD ENOUGH TO BE SIMPLE

BUT THE LOGISTICS HAVEN'T CAUGHT UP
AND SOME PARTS ARE REALLY HARD

WHY?

- Just getting the training data is hard
  - Which data? How to make it accessible? Multiple sources!
  - New kinds of observations force restarts
  - Requires a ton of domain knowledge

- The myth of the unitary model
  - You can't train just one
  - You will have dozens of models, likely hundreds or more
  - Handoff to new versions is tricky
  - You have to get run-time to be sure about which is better

WHAT MACHINE LEARNING TOOL IS BEST?

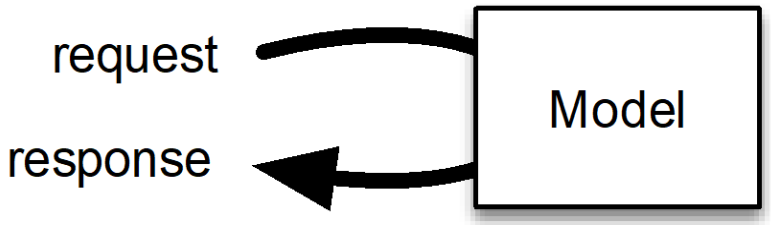
- Most successful groups keep several "favorite" machine learning tools at hand
  - No single tool is best in every situation
- The most important tool is a platform that supports logistics well
  - Don't have to do everything at the application level
  - Lots of what matters can be handled at the platform level
- A good design *for the logistics* can make a big difference
- Different horses for different courses

SOME GOTCHAS

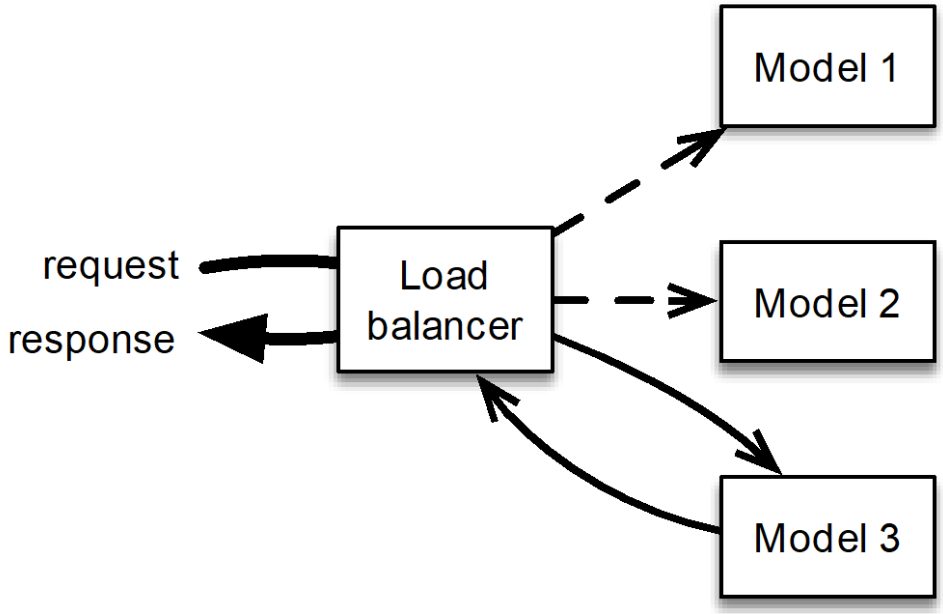
- Ops-oriented people will not “get it” regarding modeling subtleties
- Data scientists will not “get it” regarding operational realities
- Therefore, modelers have to deliver self-contained models
- And, ops has to provide pre-wired structure

So we need to plan for lots of iterations,
lots of interacting models

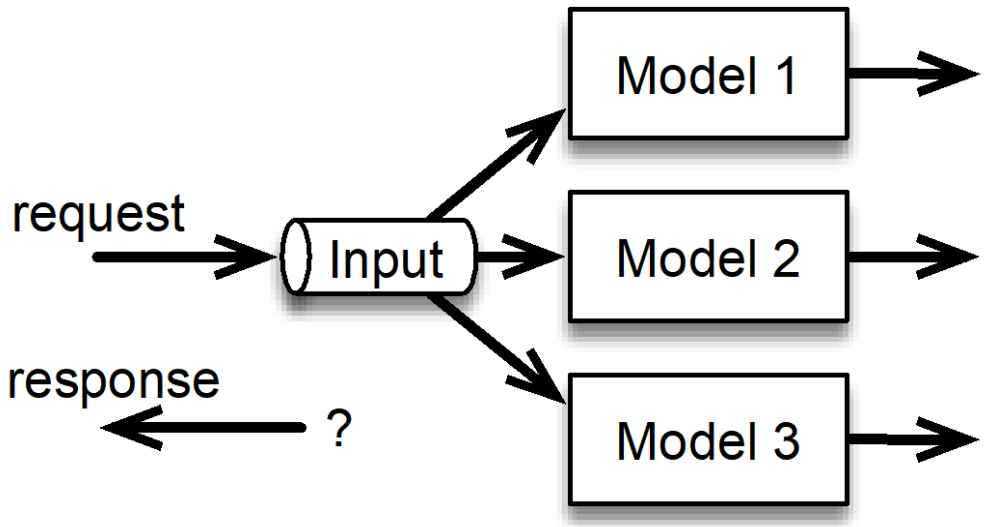
WHAT WE ULTIMATELY WANT



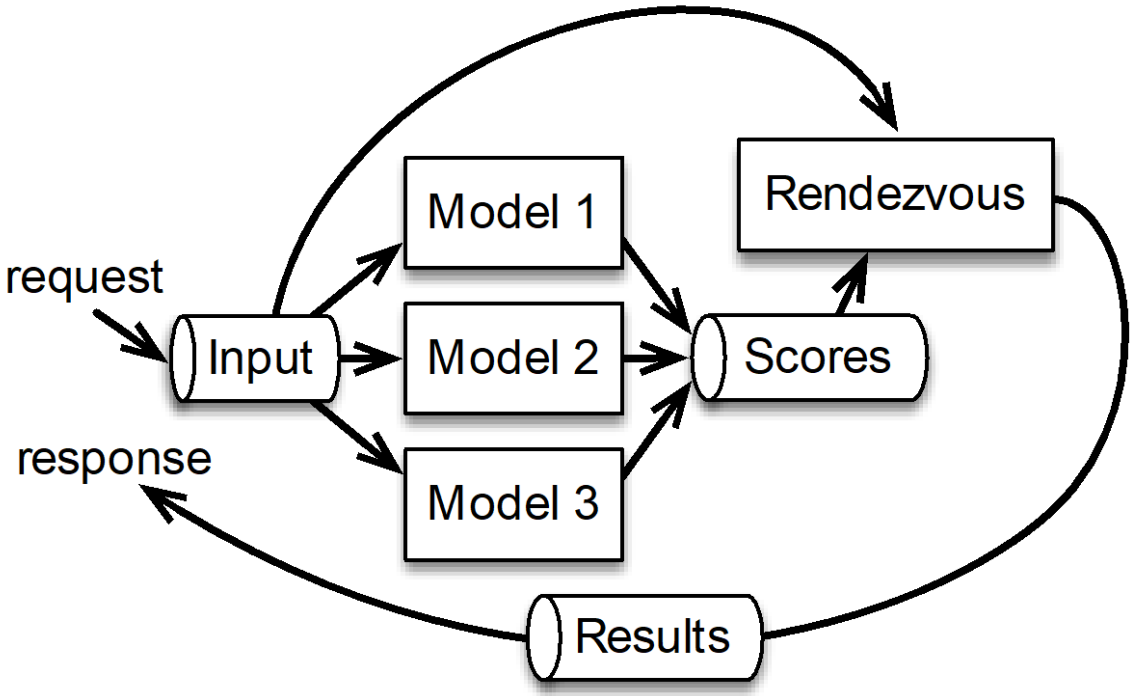
BUT THIS ISN'T THE ANSWER



FIRST TRY WITH STREAMS



FIRST RENDEZVOUS



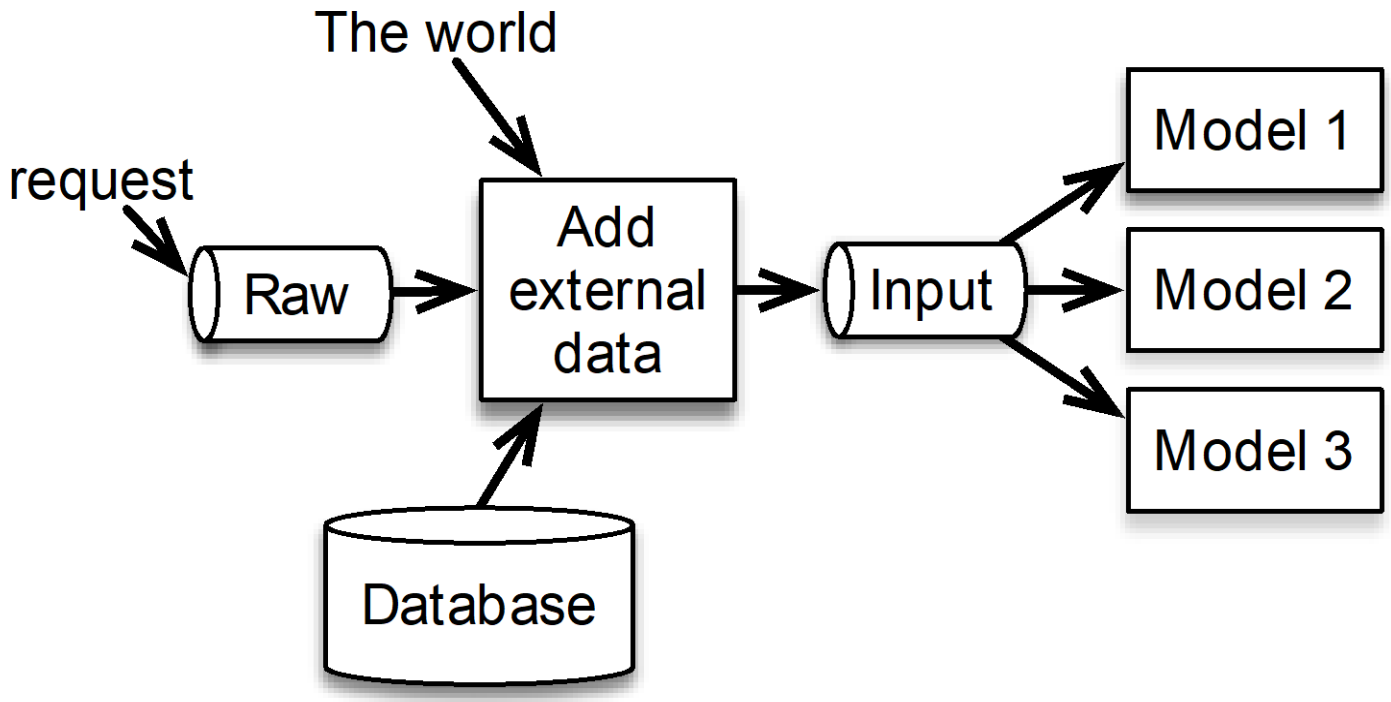
SOME KEY POINTS

- Note that *all* models see identical inputs
- *All* models run in production setting
- *All* models send scores to same stream

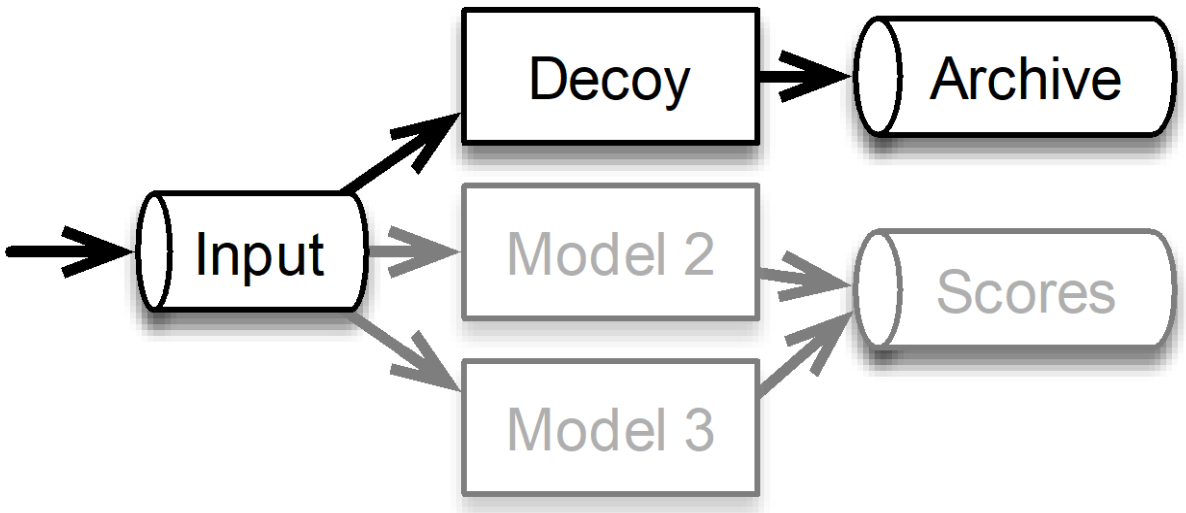
- The rendezvous server decides which scores to ignore

- Roll forward, roll back, correlated comparison are all now trivial

REALITY CHECK, INJECTING EXTERNAL STATE



RECORDING RAW DATA (AS IT REALLY WAS)



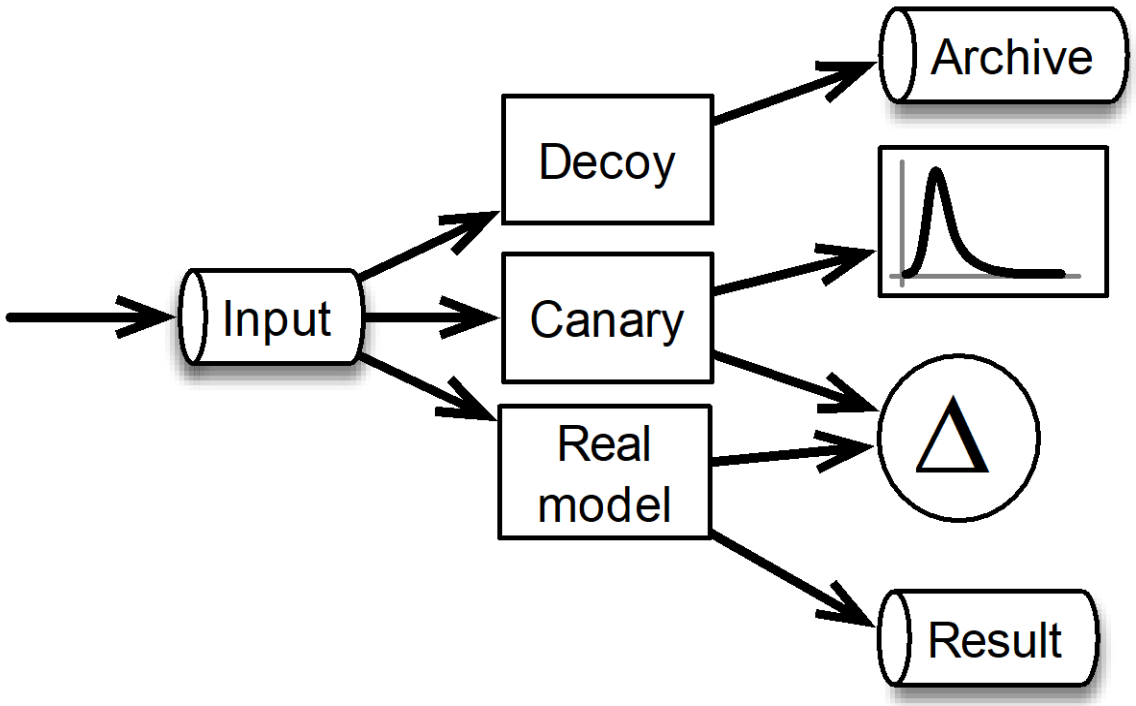
QUALITY & REPRODUCIBILITY OF INPUT DATA IS IMPORTANT!

- Recording raw-ish data is really a ***big*** deal
  - Data as seen by a model is worth gold
  - Data reconstructed later often has time-machine leaks
  - Databases were made for updates, streams are safer

- Raw data is useful for non-ML cases as well (think flexibility)

- Decoy model records training data as seen by models under development & evaluation

CANARY FOR COMPARISON



WHAT DOES THE CANARY DO?

- The canary is a real model, but is very rarely updated
- The canary results are almost never used for decisioning

- The virtue of the canary is stability

- Comparing to the canary results gives insight into new models

RENDEZVOUS SCHEDULES

- The key idea of rendezvous schedules is to define the trade-off of latency versus model priority
  - At short delays, we want the best
  - At moderate delays we will compromise a bit
  - Near the deadline, we will take any answer at all
- Normally the same rendezvous schedules apply to all transactions
  - Overriding default schedule has bona fide uses

RENDEZVOUS OVERRIDES

- Incoming transaction can carry an overriding schedule
  - This is great for QA, to see output from a specific model
  - Overriding the default schedule is also good for systemic A/B tests

- Overrides should be unusual

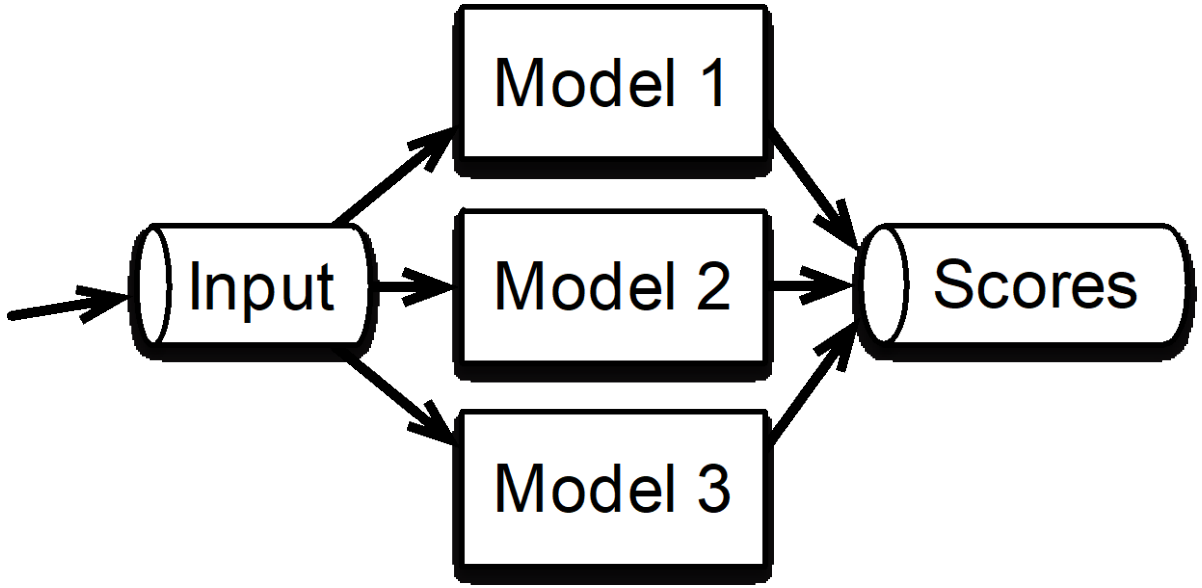
SCALING UP

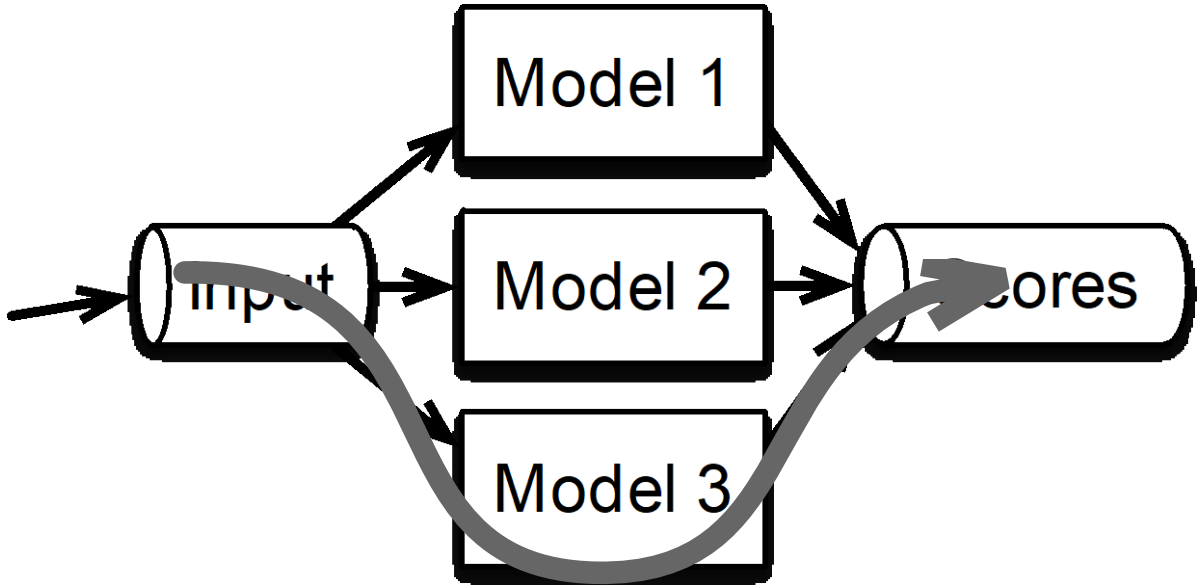
- More kinds of model
  - multiple rendezvous frameworks for different tasks
- More throughput
  - Fast default models
  - Partition input stream to allow parallel model evaluation
  - Input batching
- Extreme volumes require extreme measures
  - Cannibalize fancy models to run more fast/simple models
  - Speed before beauty

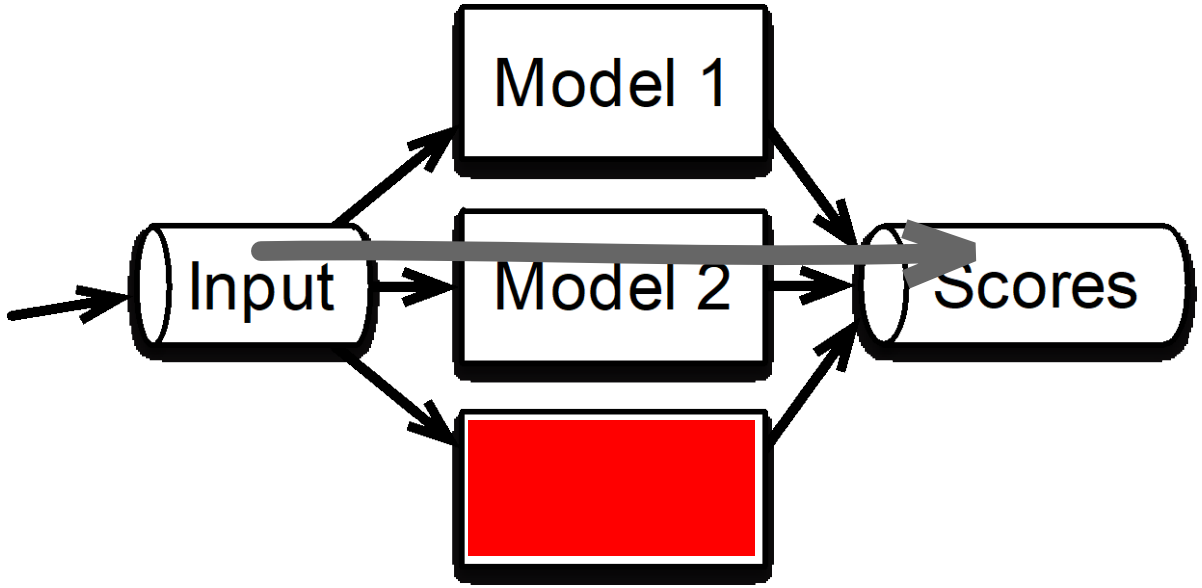
FASTER THROUGHPUT THROUGH FAILURE

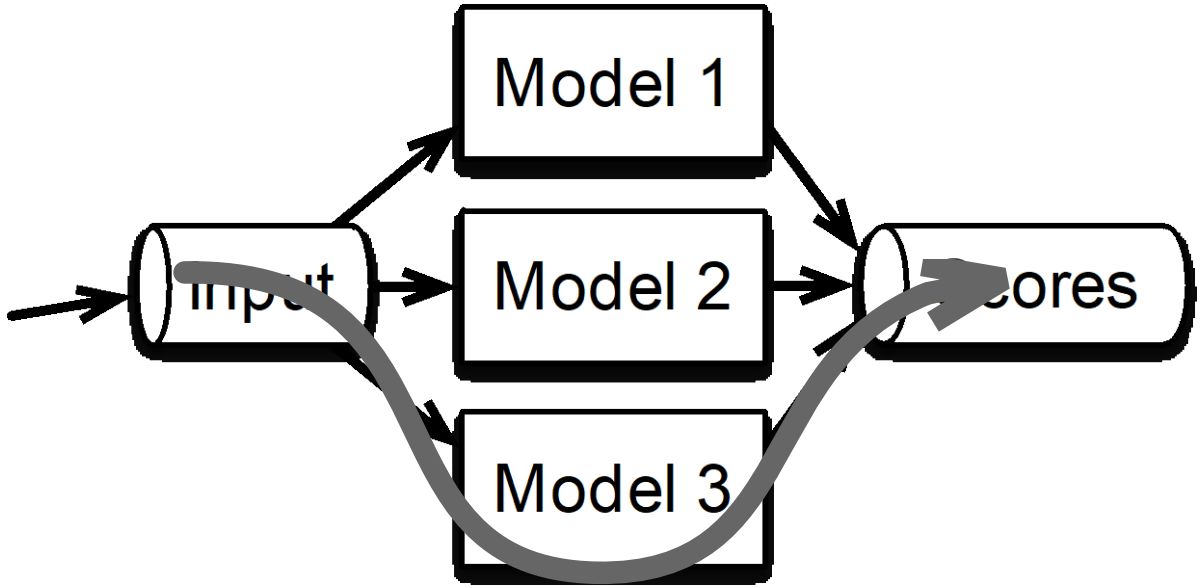
- Suppose we have one model that can handle 10,000 t/s @ 2ms
  - But this isn't the most accurate model. Not bad, but not best
- And our champion model can handle 1000 t/s @ 10ms
- Then imagine a burst of 2000 t/s for several minutes

- Champion can only evaluate half of all requests
  - Should skip to keep up
  - Fast model will cover for champion


Model 1
Input
Model 2
Scores
Model 3


Model 1
Model 2
Model 3


Model 1
Input
Model 2
Scores

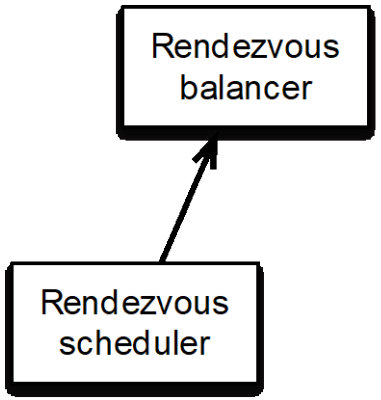

Model 1
Model 2
Model 3

Always have a default or fallback model

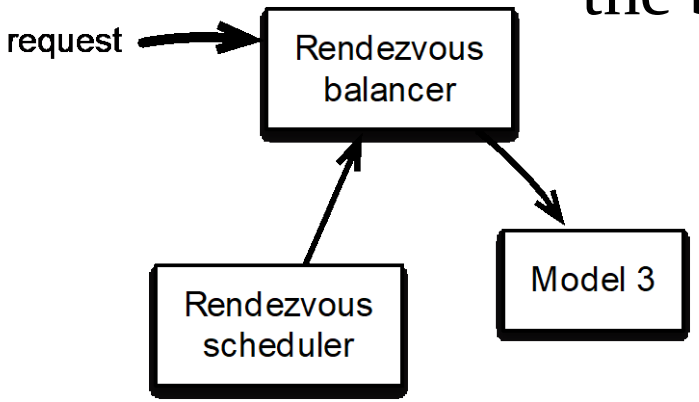
Models that fall behind can just discard requests to catch up

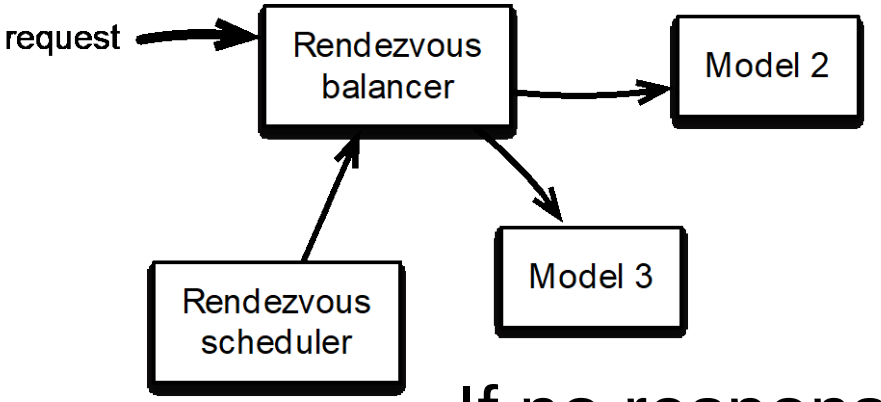
Rendezvous can be implemented
without streaming
using a load balancer architecture

On startup, the rendezvous balancer reads in the rendezvous schedule that determines which models to use



When a request arrives, it is forwarded immediately to our preferred model(s) and the pending request is recorded by the balancer.

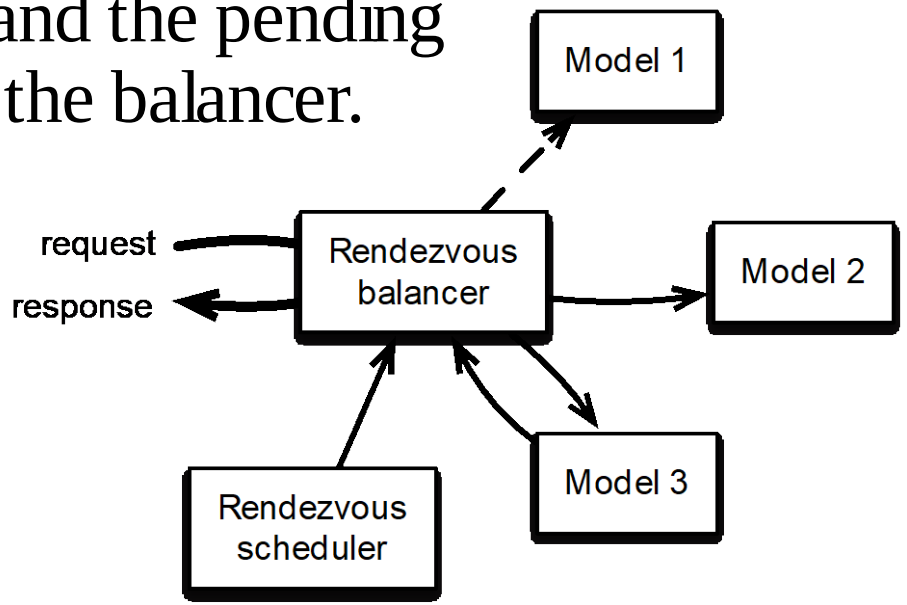




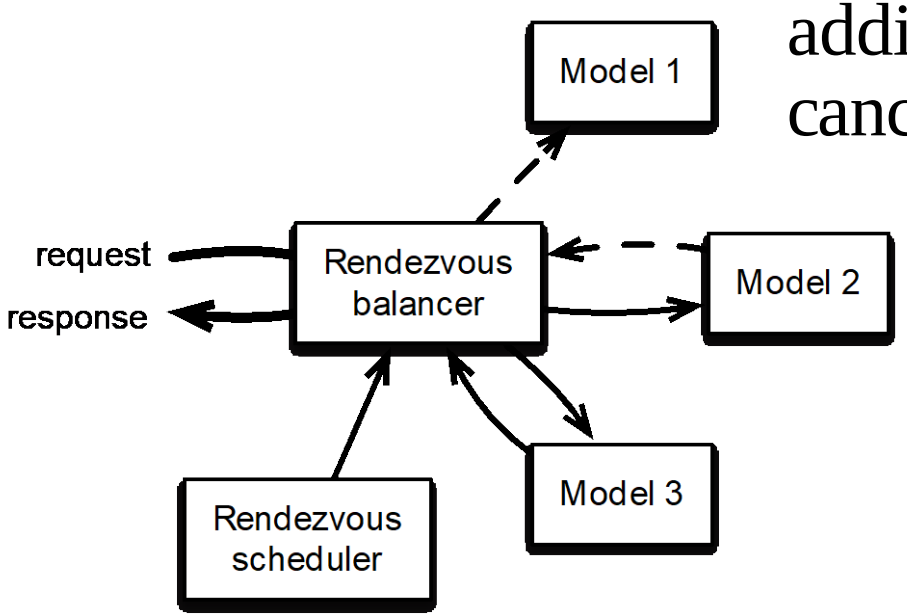
If no response is returned quickly eno

the same request is also sent to other models

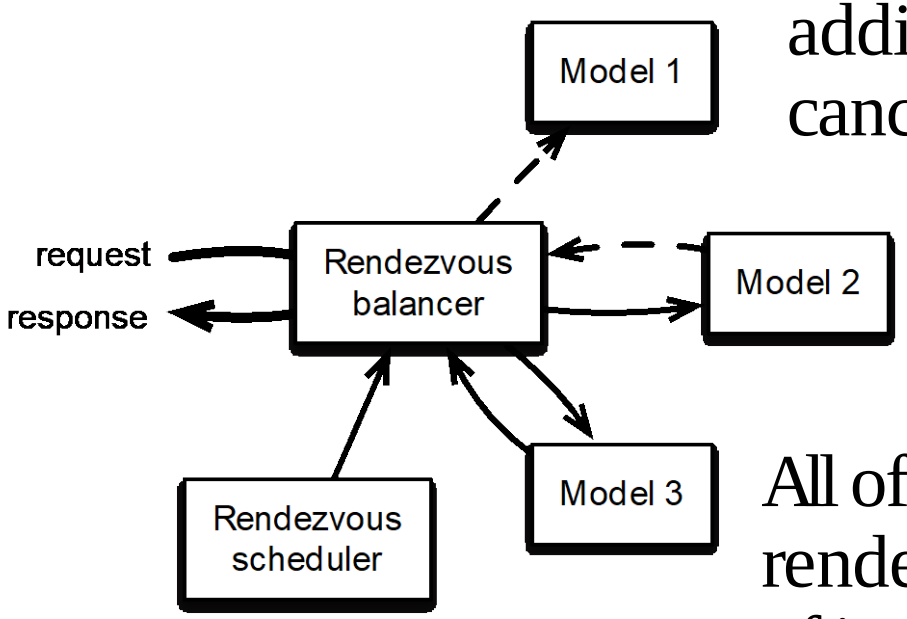
When a response is returned by any model, it is forwarded back as a response to the original query and the pending query is forgotten by the balancer.



That means that any additional responses will simply be ignored and any additional queries will be cancelled (if possible).



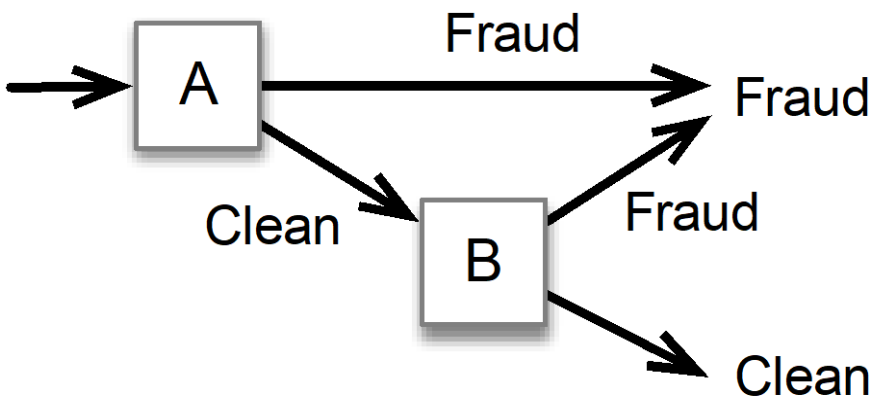
That means that any additional responses will simply be ignored and any additional queries will be cancelled (if possible).



All of the key characteristics of rendezvous apply regardless of implementation architecture

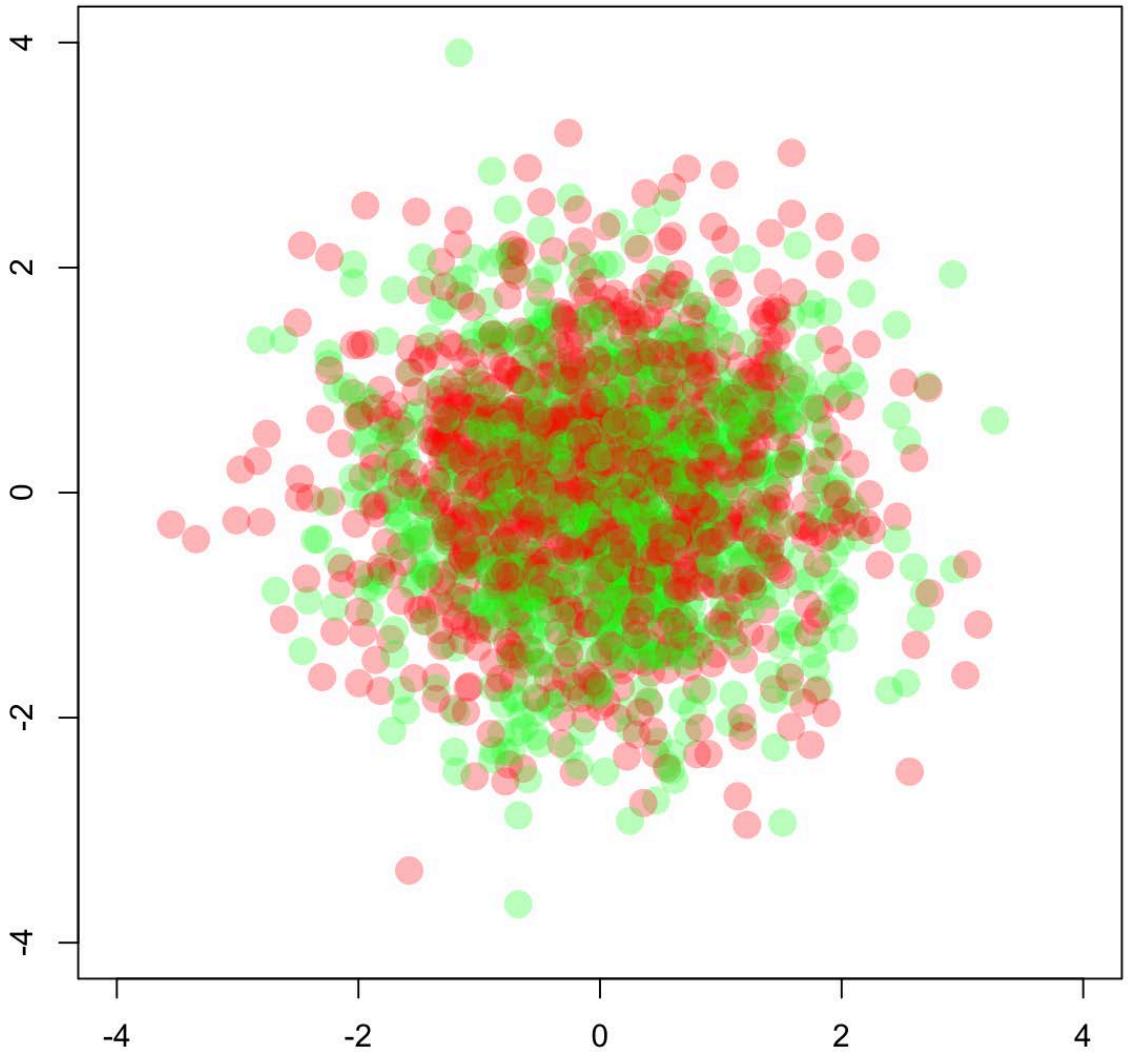
AN EXAMPLE OF HOW INTERCONNECTED MODELS CAN GO WRONG (AND HOW WE CAN SEE IT)

SAMPLE MODEL CASCADE

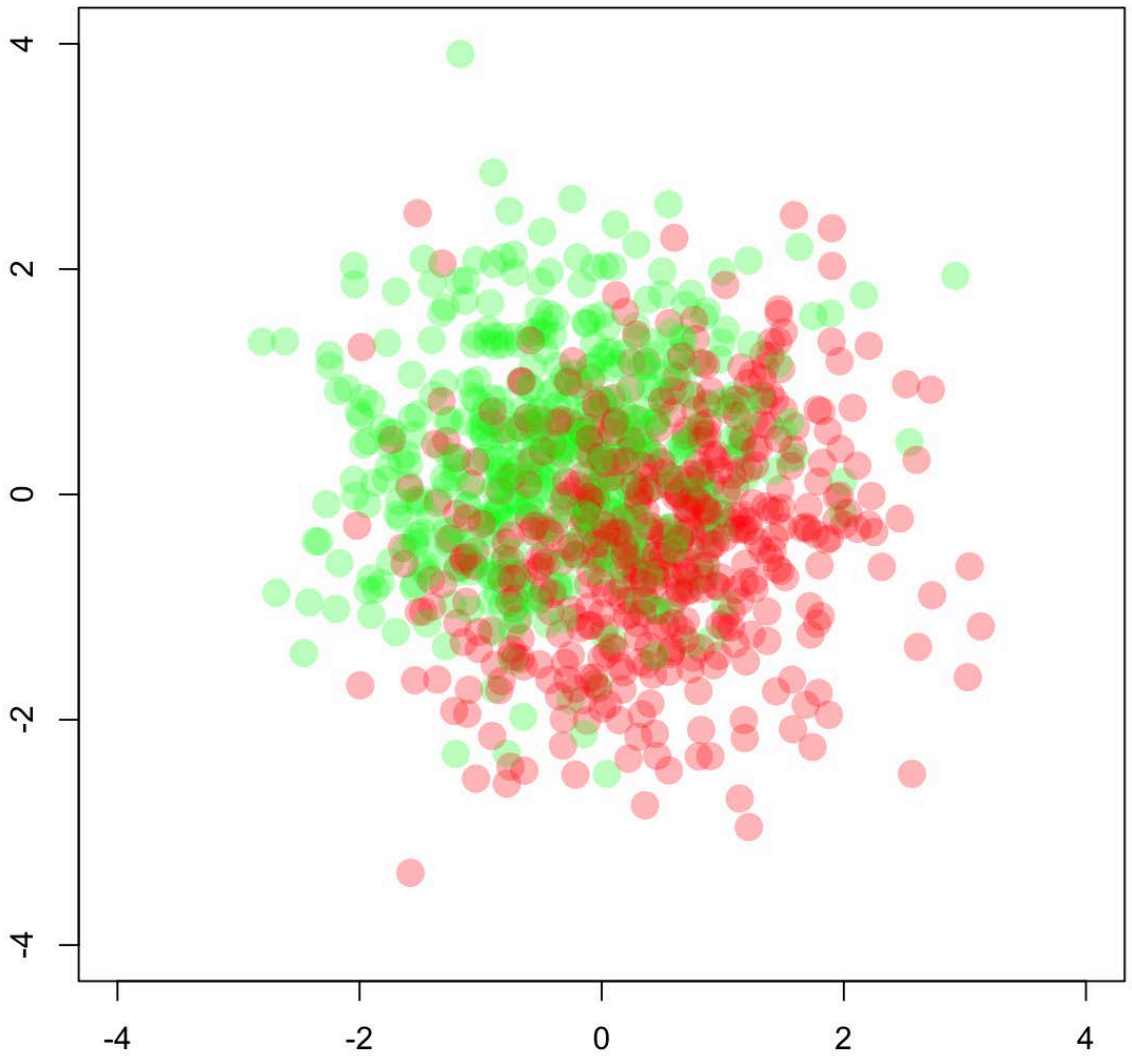


Assume that finding more frauds is all we care to do

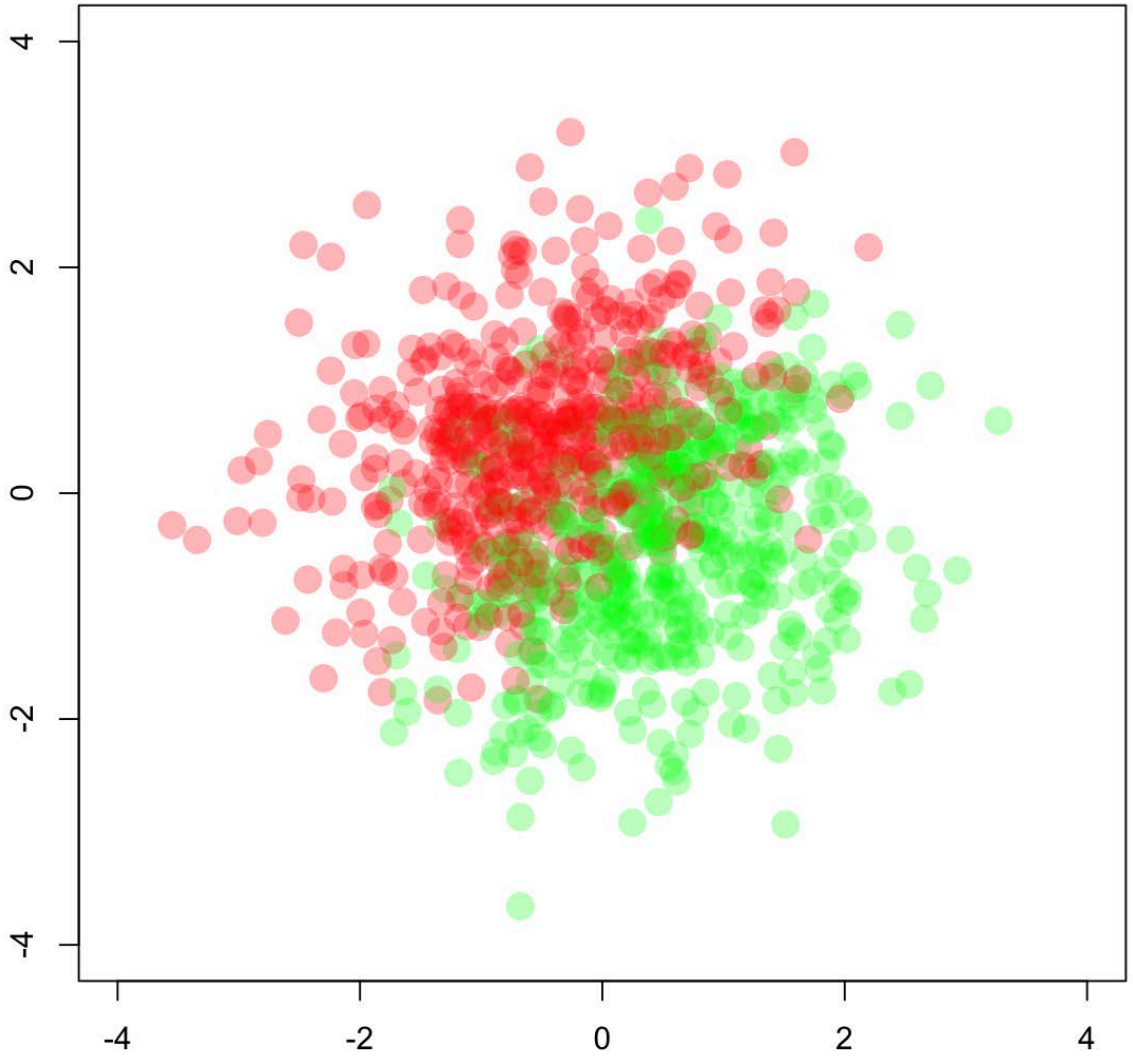
SOME DATA



CONSISTING OF TYPE 1

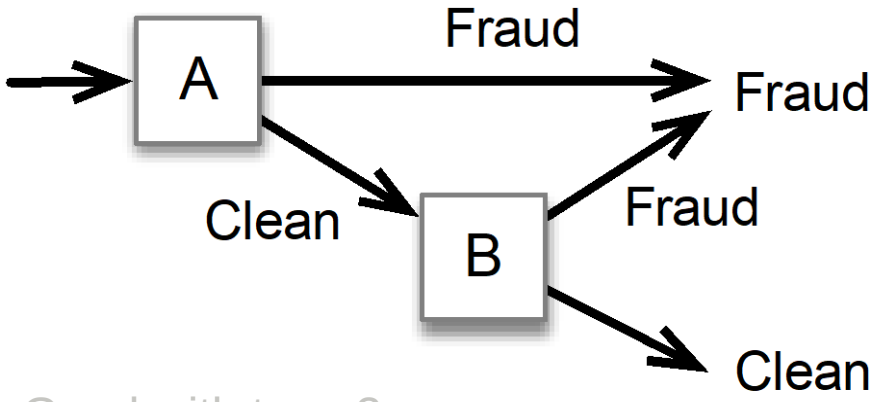


AND TYPE 2



SAMPLE MODEL CASCADE

Good with type 1



Good with type 2

BASELINE CONDITIONS

- Model A
  - 80% fraud found on type 1, 0% fraud found on type 2 (40% net)
- Model B
  - 0% fraud found on type 1, 80% fraud found on type 2 (40% net)

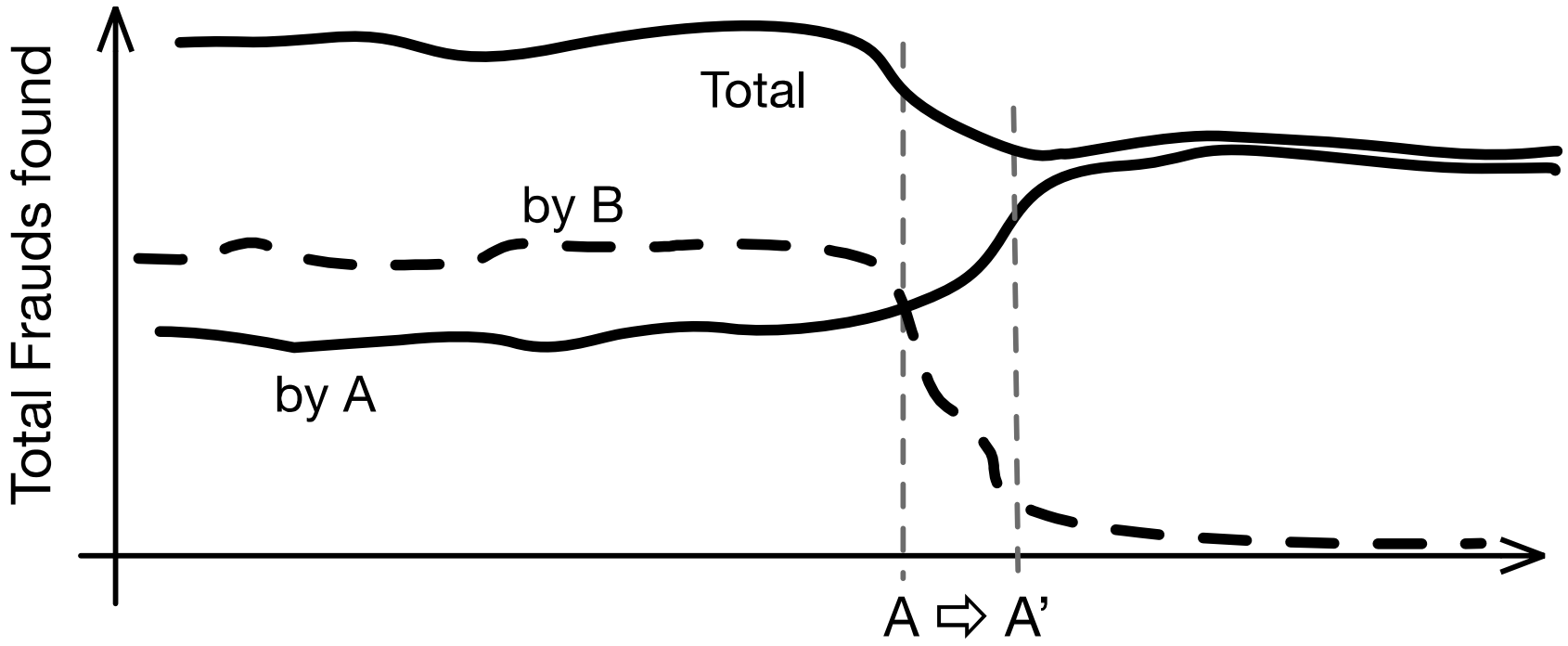
- Combined
  - No overlap in responses
  - 80% fraud found on type 1 (due to model A)
  - 80% fraud found on type 2 (due to model B)
  - 80% fraud found overall

"NEW AND IMPROVED"

- Suppose model A is "improved"
  - Before: 80% of fraud is found on type 1, 0% found on type 2 (40% net)
  - After: 40% found on type 1, 100% on type 2 (70% net)

- Combined after change
  - Huge overlap in responses
  - 40% fraud found on type 1 (due to model A)
  - 100% fraud found on type 2 (due to model A)
  - Model B has no effect
  - 70% fraud found overall, down from 80% before "improvement"

COUPLING PARADOX



IS THERE ANY HOPE?

- This kind of problem is HARD
  - Do your competitor's and your own marketing model couple?

- Where possible, use ensembles instead of cascades
  - Not as simple as it sounds
- Where possible, deploy composite models as units
  - Not as simple as it sounds

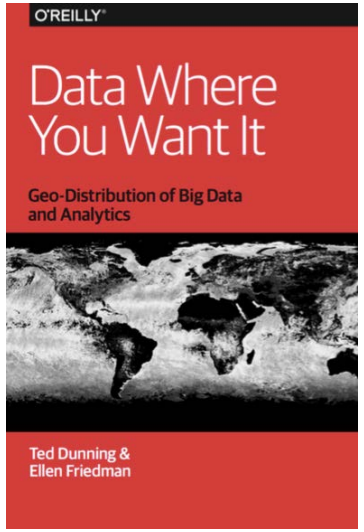
- Always measure everything!

HOW TO DO BETTER

- Data + the right question + domain knowledge matter!
- Prioritize – put serious effort into infrastructure
  - DataOps requires more than just data science
- Persist – use streams to keep data around
- Measure – everything, and record it
- Meta-analyze – understand and see what is happening
- Containerize – make deployment repeatable, easy

- Oh…don't forget to do some machine learning, too

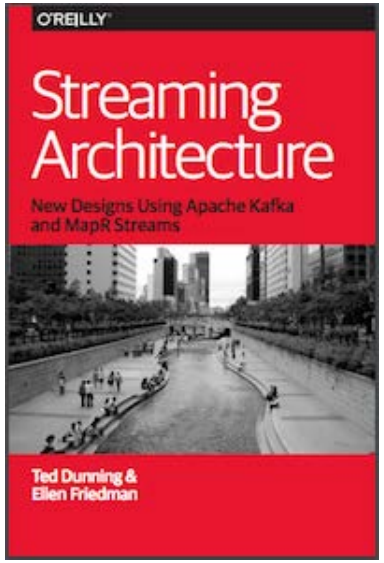
ADDITIONAL RESOURCES



O'Reilly report by Ted Dunning & Ellen Friedman © March 2017

Read free courtesy of MapR:

https://mapr.com/geo-distribution-big-data-and-analytics/

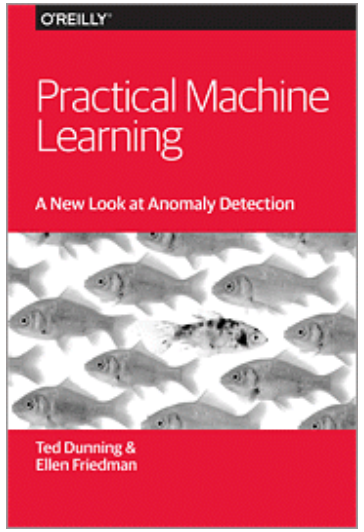


O'Reilly book by Ted Dunning & Ellen Friedman © March 2016

Read free courtesy of MapR:

https://mapr.com/streaming-architecture-using-apache-kafka-mapr-streams/

ADDITIONAL RESOURCES



O'Reilly book by Ted Dunning & Ellen Friedman
© June 2014

Read free courtesy of MapR:

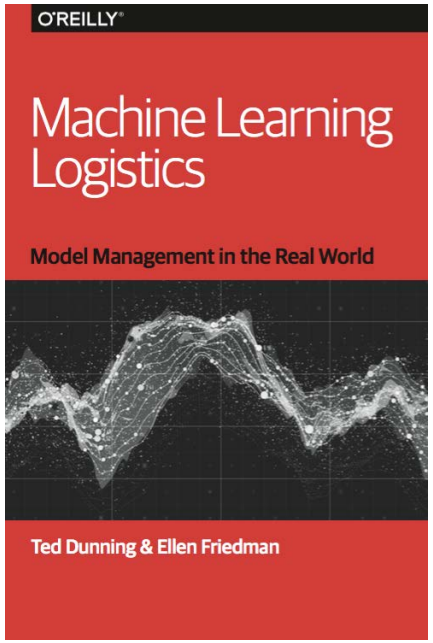
https://mapr.com/practical-machine-learning-new-look-anomaly-detection/

O'Reilly book by Ellen Friedman & Ted Dunning
© February 2014

Read free courtesy of MapR:

https://mapr.com/practical-machine-learning/

ALSO: MACHINE LEARNING LOGISTICS



O'Reilly book by Ellen Friedman & Ted Dunning © Sept 2017

Download free from MapR

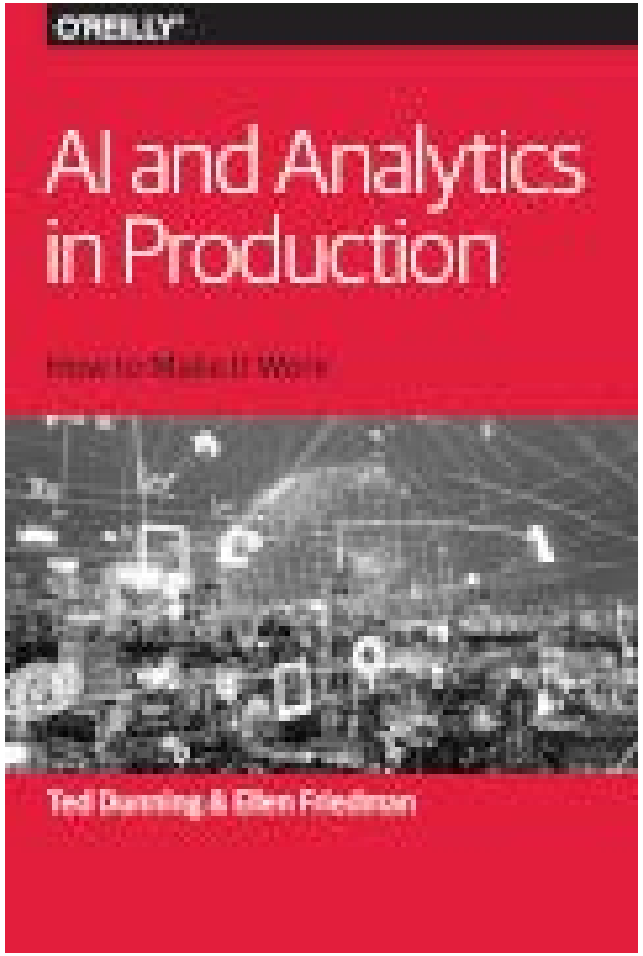
http://info.mapr.com/2017_Content_Machine_Learning_Logistics_eBook_Prereg_RegistrationPage.html

AI AND ANALYTICS IN PRODUCTION: HOW TO MAKE IT WORK

O’Reilly book by Ellen Friedman & Ted Dunning
© Sept 2017

Download free from MapR

https://mapr.com/ebook/ai-and-analytics-in-production/



Please support women in tech – help build girls' dreams of what they can accomplish

#womenintech #datawomen

© Ellen Friedman 2015

THANK YOU

ted.dunning@hpe.com