Data-Intensive Distributed Computing
CS 451/651 (Fall 2025)

UNIVERSITY OF WATERLOO

Data Infrastructure for Machine Learning
(v1.01)

Week 6: October 9

Jimmy Lin
David R. Cheriton School of Computer Science
University of Waterloo

These slides are available at http://lintool.github.io/bigdata-2025f/

Key Questions

What are the key components of an ML solution?

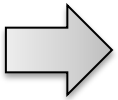
How is the supervised machine learning problem formulated?

What roles do data platforms and data engineering play?

Instance

amazing spot for good food & a fun time 🍕🍝 they offer a super unique dine-in experience with their interactive tables! also love that they have innovative weekly feature dishes 😋

Model



Prediction

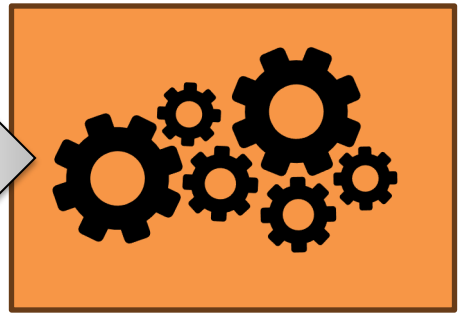
Instance

amazing spot for good food & a fun time 🍕🍝 they offer a super unique dine-in experience with their interactive tables! also love that they have innovative weekly feature dishes 😋

$$\mathbf{x}_i = [x_1, x_2, x_3, \ldots, x_d]$$

Feature Vector

$$f : X \rightarrow Y$$



Model

Prediction

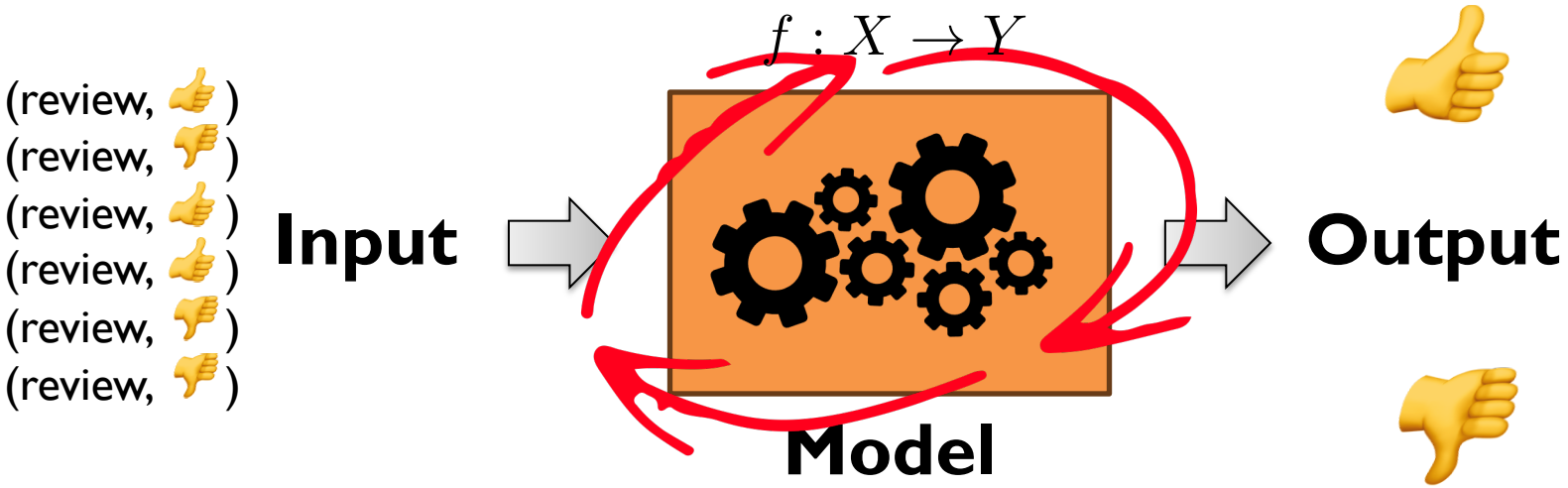
👍

$$y \in \{0, 1\}$$

Components of an ML solution

(data, features, model, optimization)

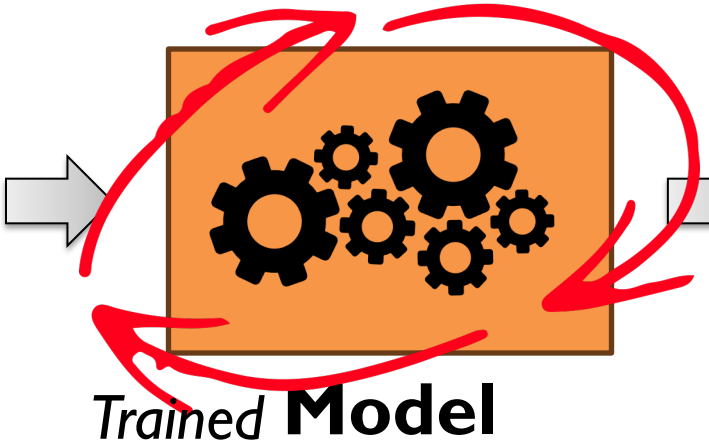
Model *learns* from the data



Machine learning algorithm adjusts the model *parameters*

(review, 👍)
(review, 👎)
(review, 👍)
(review, 👍)
(review, 👎)
(review, 👎)

Input



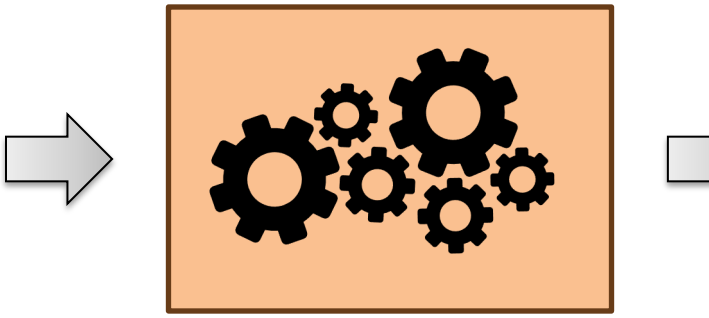
Trained **Model**

Output

Deployment

Inference / Prediction

A group of us stopped by yesterday afternoon to enjoy an outdoor lunch. The food was da bomb.



Trained **Model**

👍

Got it!

Gather training data

Train model

Deploy model

Goal for today: Dispel the myth of what a data scientist *actually* does.

Applied {ML, AI} Researcher, etc.

Origins:

Applied ML in Academia

Download interesting dataset (comes with the problem)

Run baseline model

Train/Test

Build better model

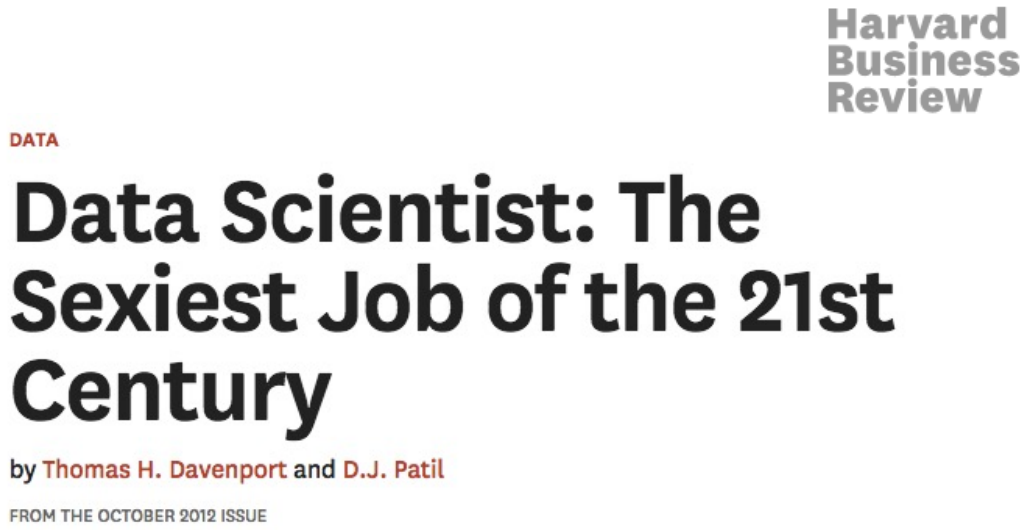
Train/Test

Does new model beat baseline?

Yes: publish a paper!

No: try again!

Cool, you do that in industry,
except you get paid a lot more!

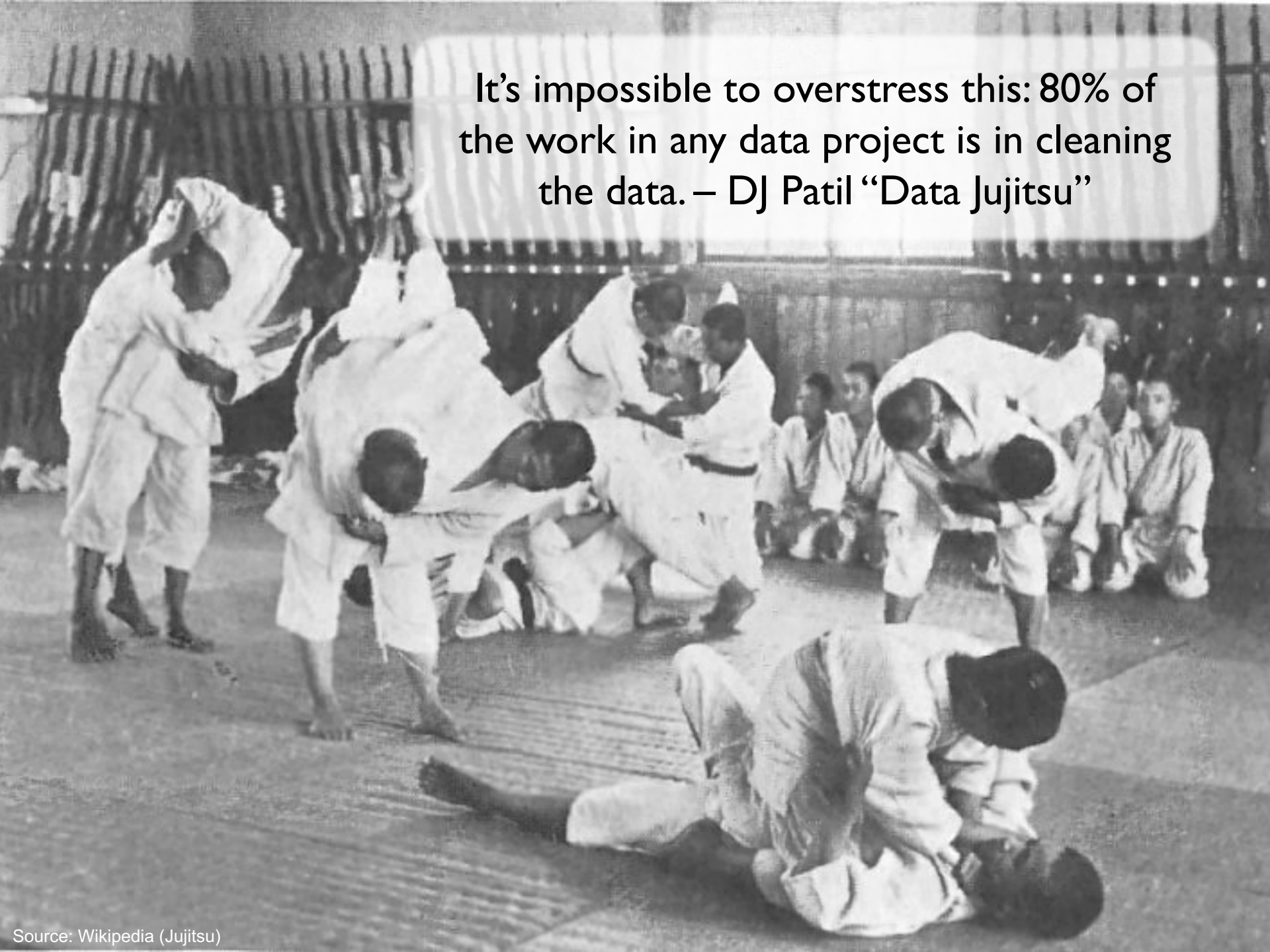


Goal for today: Dispel the myth of what a

data scientist *actually* does.

Applied {ML, AI} Researcher, etc.

<pause/>



Source: Wikipedia (Jujitsu)

The New York Times

TECHNOLOGY

For 'Big Data' Scientists, Hurdle to Insights Is 'Janitor Work'

By STEVE LOHR AUG. 17, 2014

Monica Rogati, Jawbone's vice president for data science, with Brian Wilt, a senior data scientist.

Peter DaSilva for The New York Times

Raw data
(e.g., from ELT)

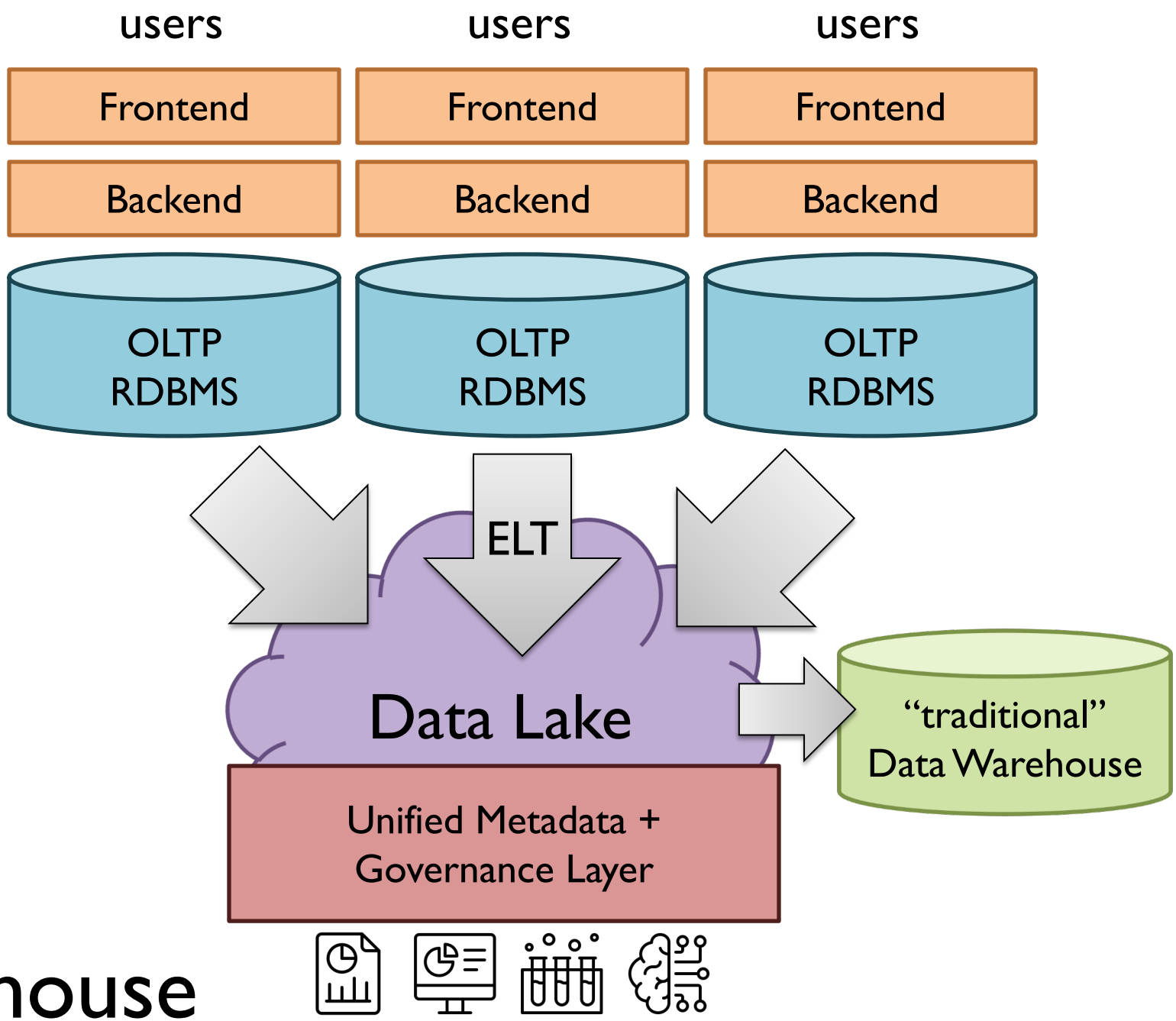
High-quality training data

GI/GO

Gather training data

Train model

Deploy model


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
Data Lake
Unified Metadata + Governance Layer
"traditional" Data Warehouse

Lakehouse

Where's the data?
Who owns that data?
Can I have access to it?
What does this field mean?
Why does it have this value?
What's all this $#@%&?
Wait, this isn't *actually* what I need.

Source: Wikipedia (Plumbing)

On finding things…

On naming things…

uid

UserId

userId

userid

CamelCase

smallCamelCase

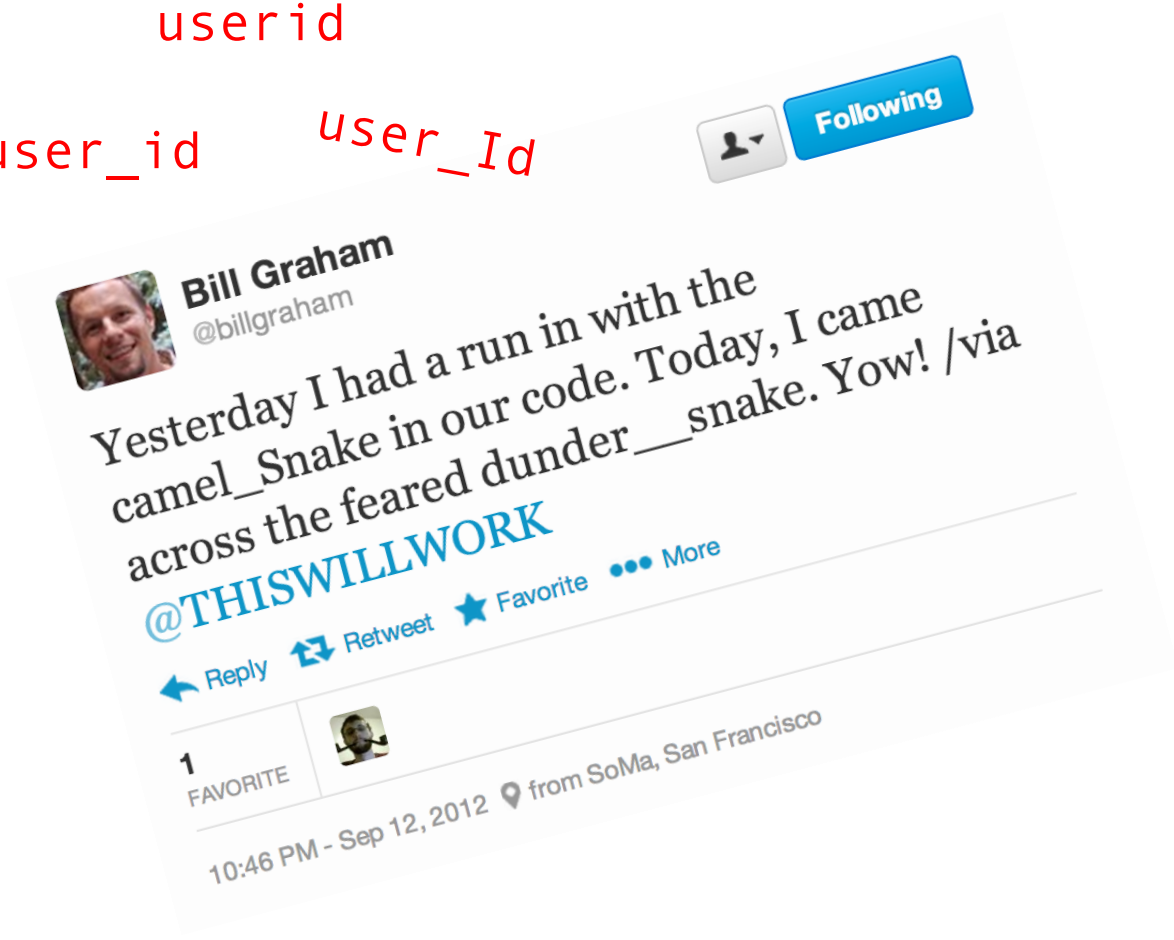
user_id

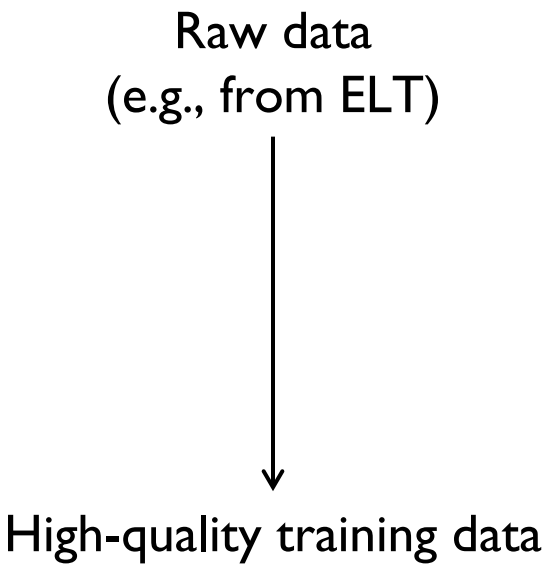
user_Id

snake_case

camel_Snake

dunder__snake





Gather training data

Train model

Deploy model

Assume you've got a high-quality labeled dataset…

Gather training data

Train model

Deploy model

The Task

Given: $D = \{(\mathrm{x}_i, y_i)\}_i^n$

label

feature vector

$$\mathrm{x}_i = [x_1, x_2, x_3, \ldots, x_d]$$
$$y \in \{0, 1\}$$

Induce: $f : X \to Y$

Such that loss is minimized

$$\frac{1}{n} \sum_{i=0}^{n} \ell(f(\mathrm{x}_i), y_i)$$

loss function

Typically, we consider functions of a parametric form:

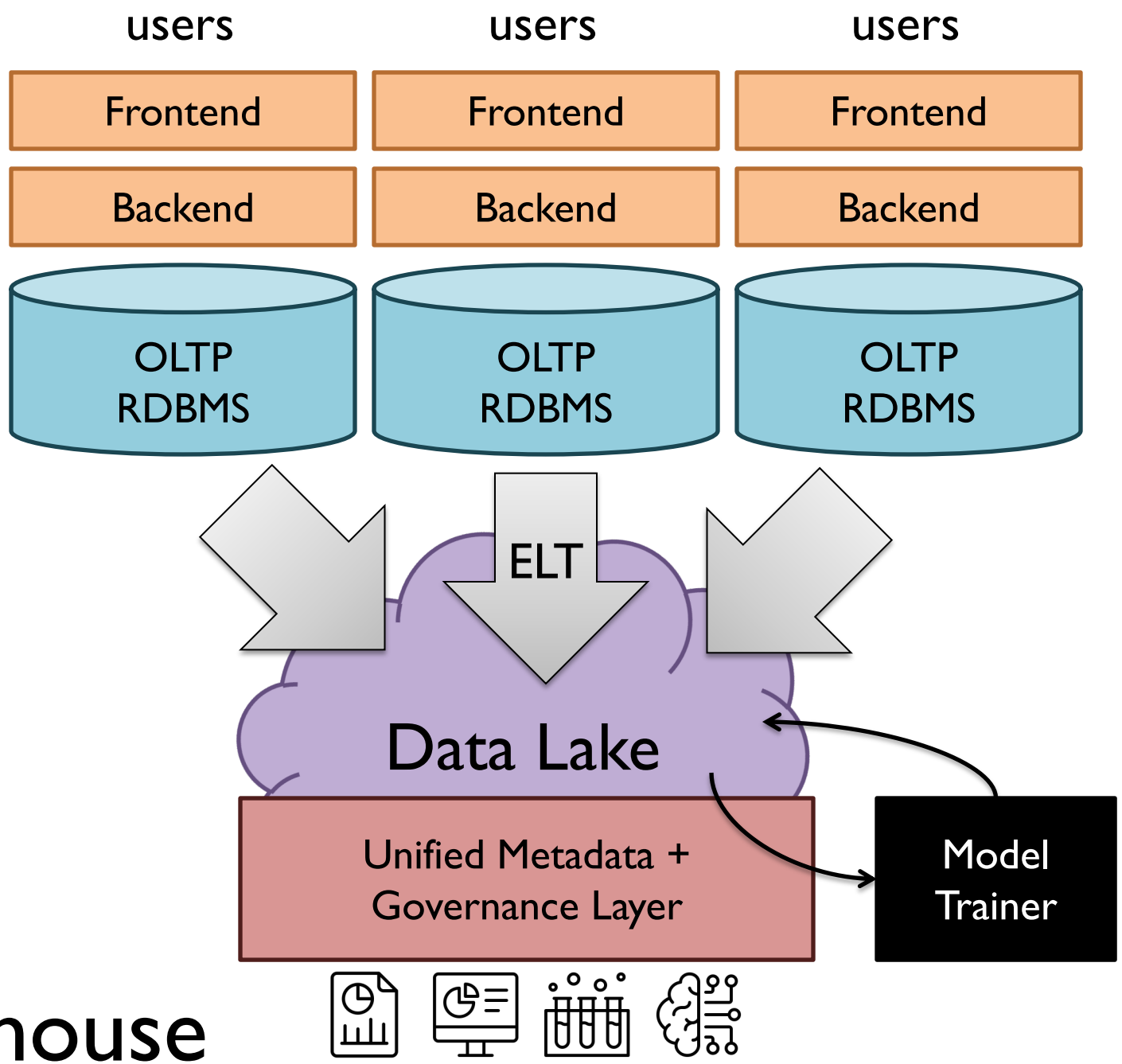
$$\arg\min_{\theta} \frac{1}{n} \sum_{i=0}^{n} \ell(f(x_i; \theta), y_i)$$

model parameters

How do you do it?

Use `sklearn`: `model.fit(X, y)`

Didn't work? use `pkg1`: `model1.fit(X, y)`
Didn't work? use `pkg1`: `model2.fit(X, y)`
Didn't work? use `pkg2`: `modelA.fit(X, y)`


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
Data Lake
Unified Metadata + Governance Layer
Model Trainer

Lakehouse

Gradient Descent

(Batch) Gradient Descent

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{n} \sum_{i=0}^{n} \nabla \ell(f(\mathrm{x}_i; \theta^{(t)}), y_i)$$

"batch" learning: update model after considering ***all*** training instances

Stochastic Gradient Descent (SGD)

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(\mathrm{x}; \theta^{(t)}), y)$$

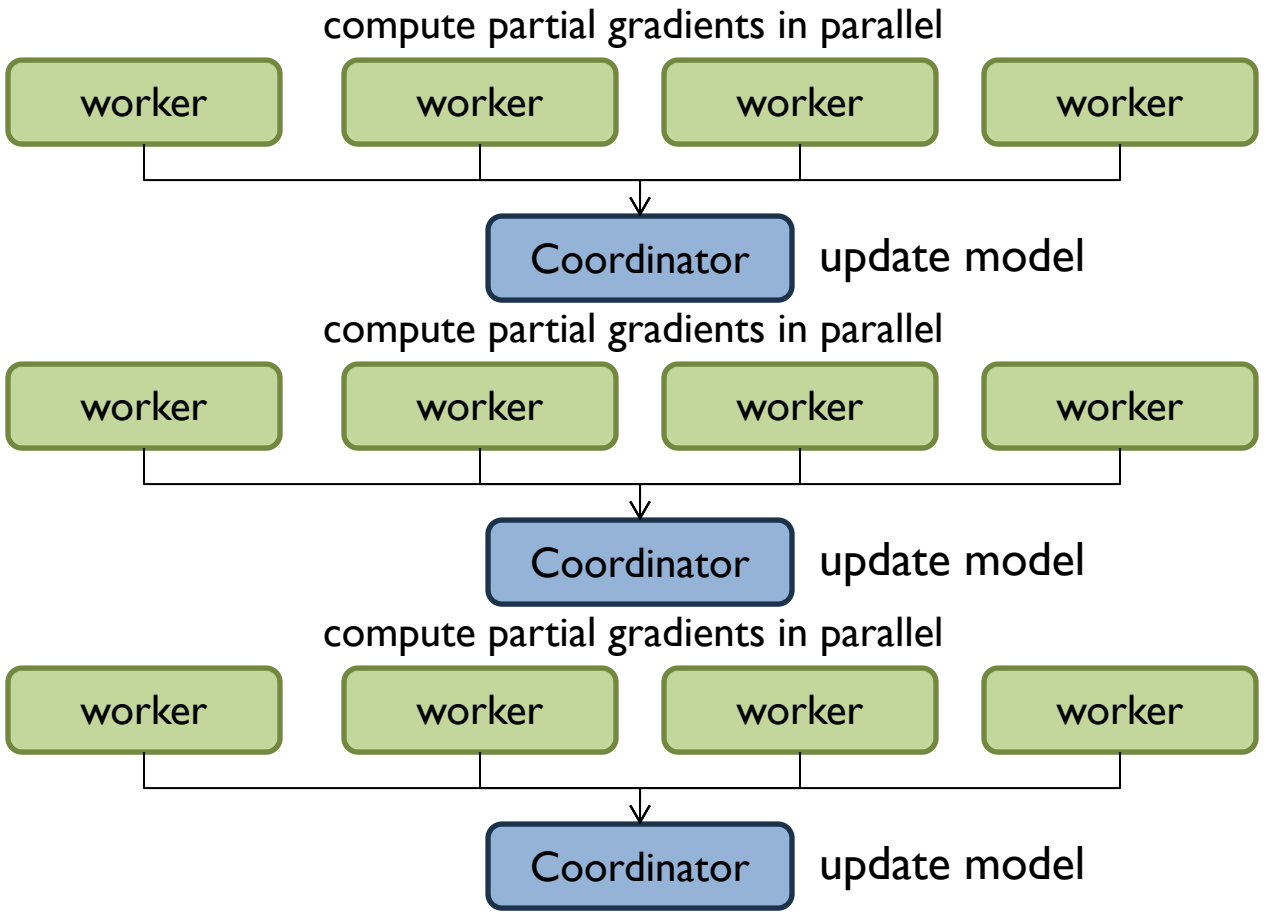
"online" learning: update model after considering ***each*** (randomly selected) training instance

Source: Wikipedia (Hills)

Source: Wikipedia (Water Slide)

(Batch) Gradient Descent

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{n} \sum_{i=0}^{n} \nabla \ell(f(\mathrm{x}_i; \theta^{(t)}), y_i)$$

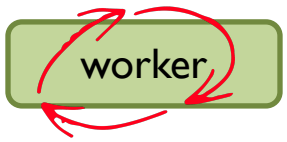


What's the issue here?

Stochastic Gradient Descent

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(\mathrm{x}; \theta^{(t)}), y)$$

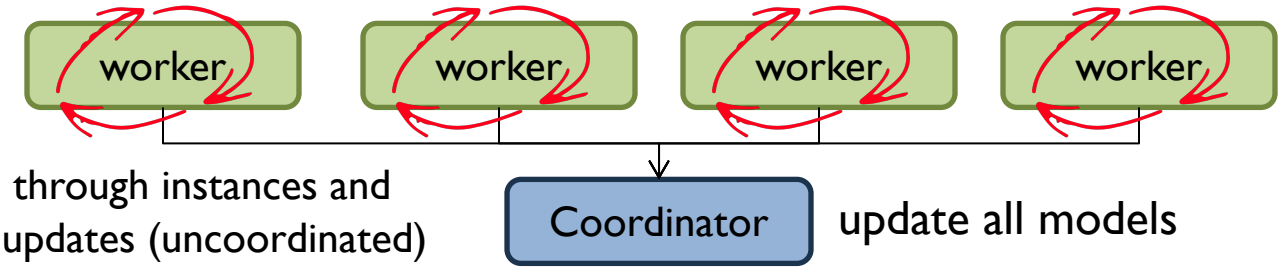
single worker streams through instances and performs model updates



Important: Model update after every instance

How do you parallelize?

Model update after every instance



worker streams through instances and perform model updates (uncoordinated)

update all models

Stochastic Gradient Descent w/ Mini-Batches

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(\mathrm{x}; \theta^{(t)}), y)$$

single worker streams through instances and performs model updates

Important: Model update after every instance

Problem: updates are very noisy

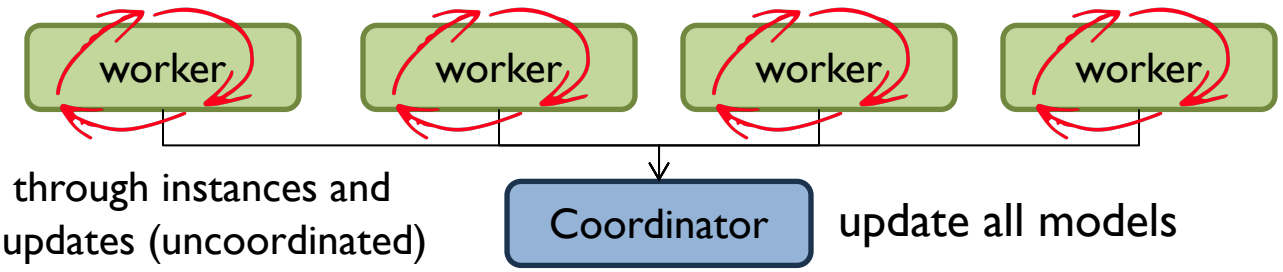
Solution: mini-batches

Divide dataset into small batches (e.g., 64)

Perform gradient descent on each mini-batch

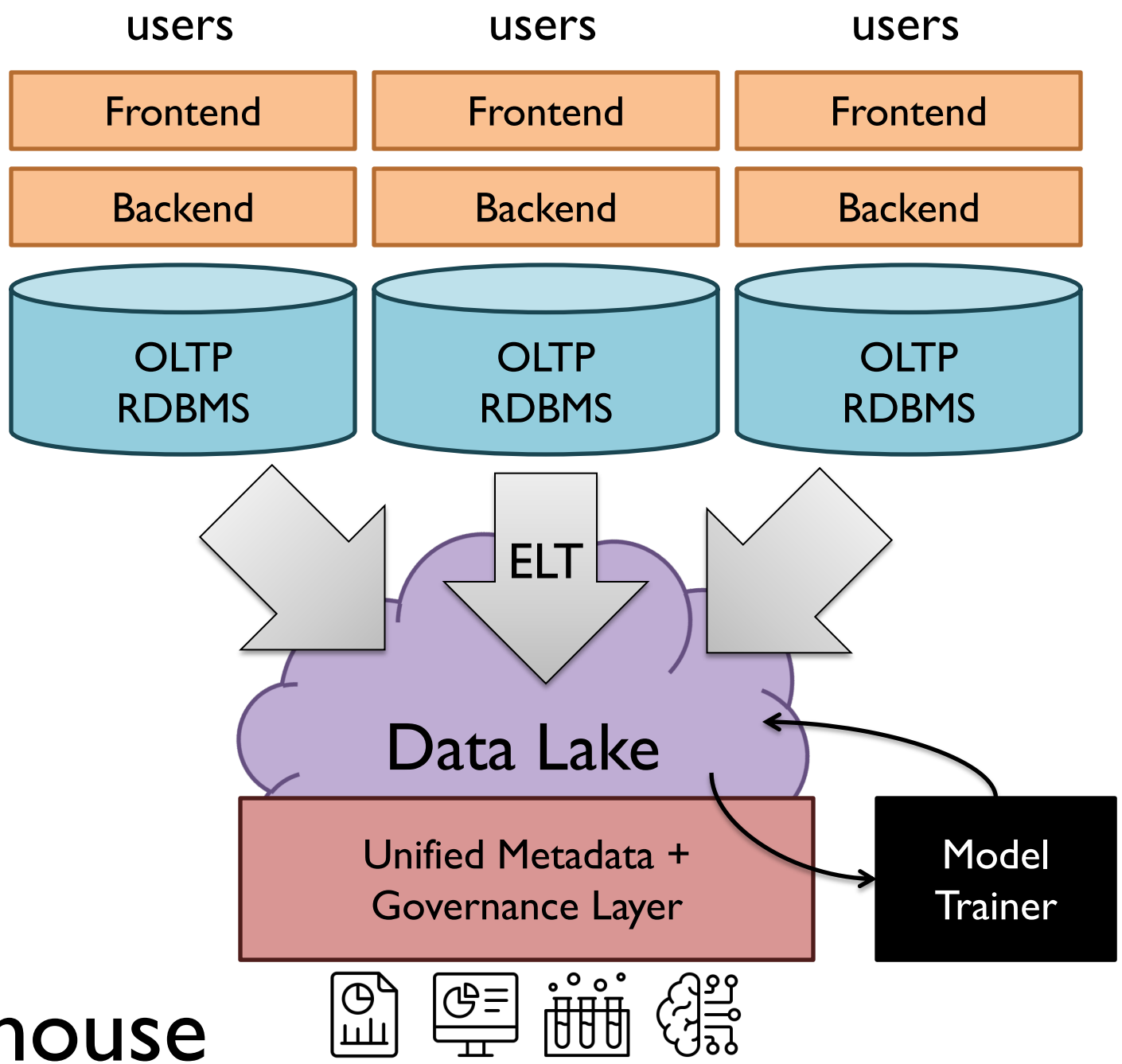
Update model after processing each mini-batch

Model update after every ~~instance~~ *mini-batch*



worker streams through instances and perform model updates (uncoordinated)

update all models


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
Data Lake
Unified Metadata + Governance Layer
Model Trainer

Lakehouse

Gather training data

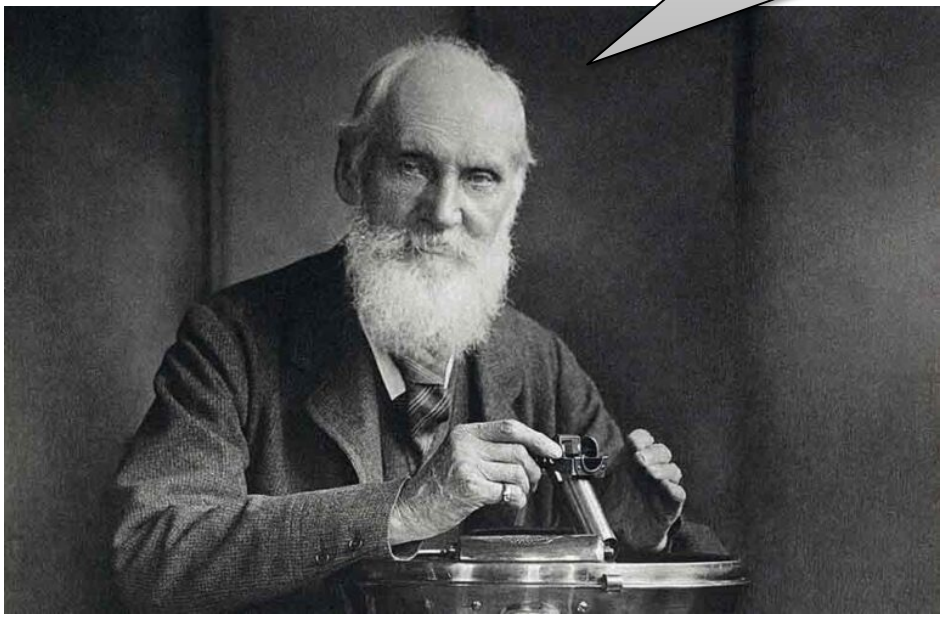
Train model

Deploy model

How do you know if it's better?
(Better than what?)

How do you know it's better?

If you can't measure it,
you can't improve it…



Define “better”

“Performance”

Faster: latency, throughput, etc.

Cheaper: cost per query, watt per query, etc.

More scalable: server load, memory usage, etc.

Higher-quality output

Define “higher quality”

Things you can measure but don’t have ground truth

Clickthrough rates

Time on site

…

Things you can measure but may have “ground truth”

Accuracy

Precision, recall

nDCG

…

Things that are difficult to measure

Quality of a summary

Quality of an answer to a question

Quality of an LLM response

…

tl;dr – evaluation is *really* hard

What are you trying to accomplish?

Avoid *post hoc* justifications

Tension between user and business goals

Okay, *assume* we know what to measure

Okay, *assume* we can measure it

Benchmark datasets

Static, can be internal or external

Batch evaluations

"Prospective", internal data – but on the data platform

A/B tests

Prospective, internal data – but "in the wild"

When the data and the anecdotes disagree,
the anecdotes are usually right.

– Jeff Bezos

The Task

label

Given: $D = \{(\mathrm{x}_i, y_i)\}_i^n$

feature vector

$$\mathrm{x}_i = [x_1, x_2, x_3, \ldots, x_d]$$

$$y \in \{0, 1\}$$

Why isn't this enough?

Induce: $f : X \rightarrow Y$

Such that loss is minimized

$$\frac{1}{n} \sum_{i=0}^{n} \ell(f(\mathrm{x}_i), y_i)$$

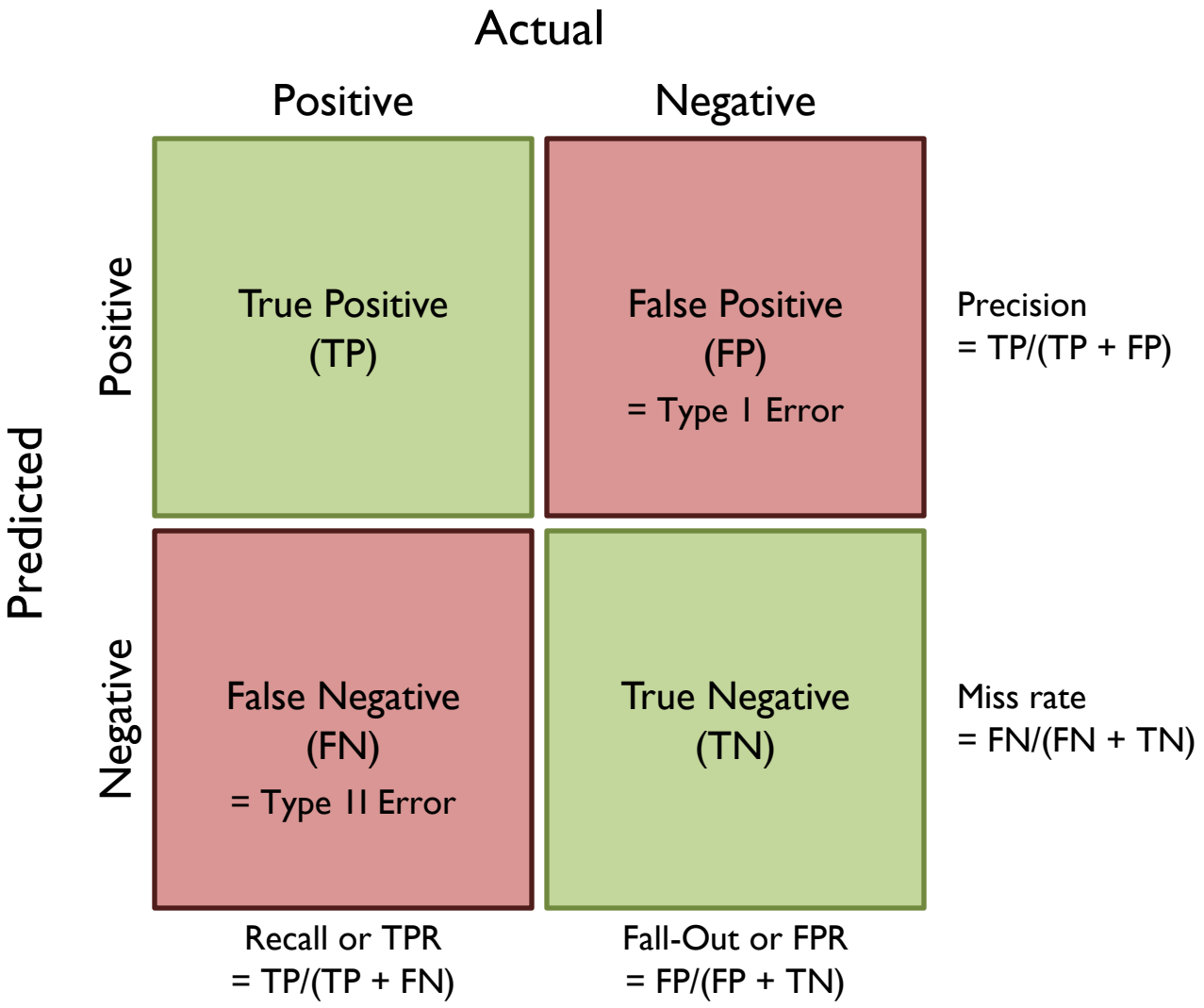
loss function

Typically, we consider functions of a parametric form:

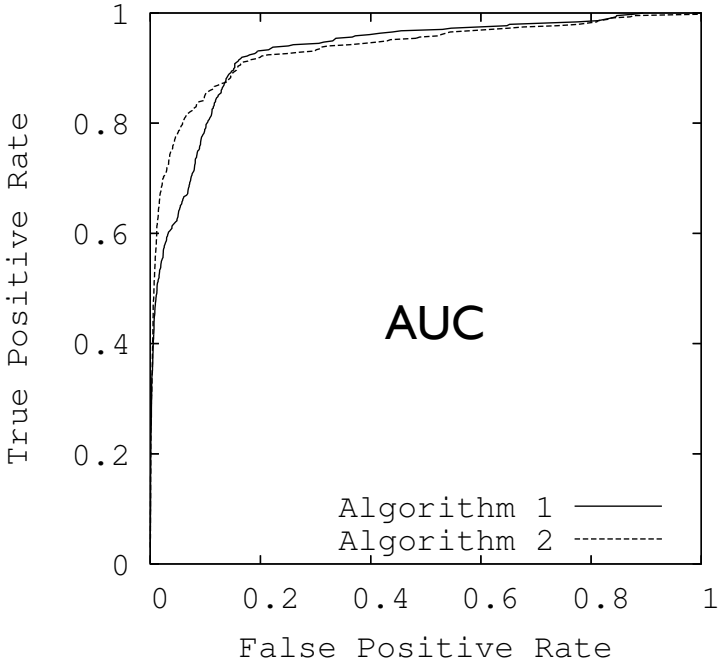
$$\arg\min_{\theta} \frac{1}{n} \sum_{i=0}^{n} \ell(f(x_i; \theta), y_i)$$

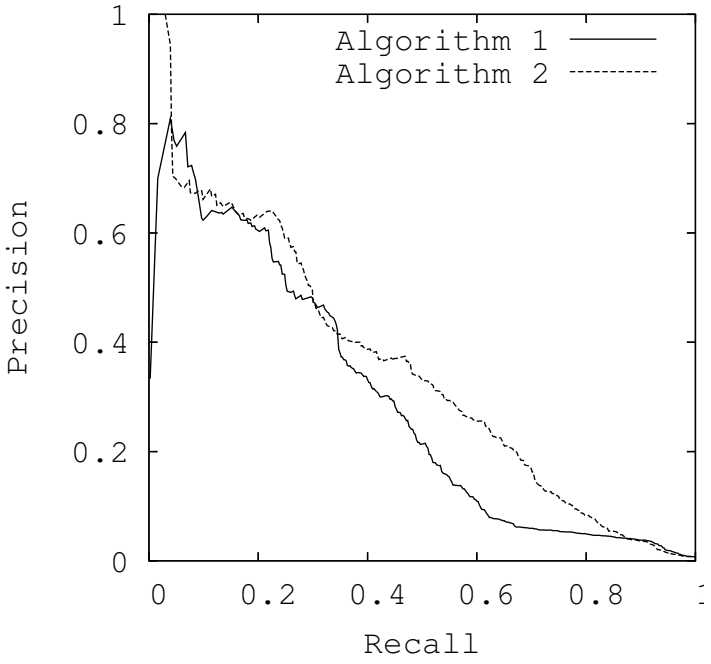
model parameters

Metrics


Actual
Positive
Negative
Predicted
Positive
Negative
True Positive
(TP)
False Positive
(FP)
= Type I Error
False Negative
(FN)
= Type II Error
True Negative
(TN)
Precision
= TP/(TP + FP)
Miss rate
= FN/(FN + TN)
Recall or TPR
= TP/(TP + FN)
Fall-Out or FPR
= FP/(FP + TN)

ROC and PR Curves

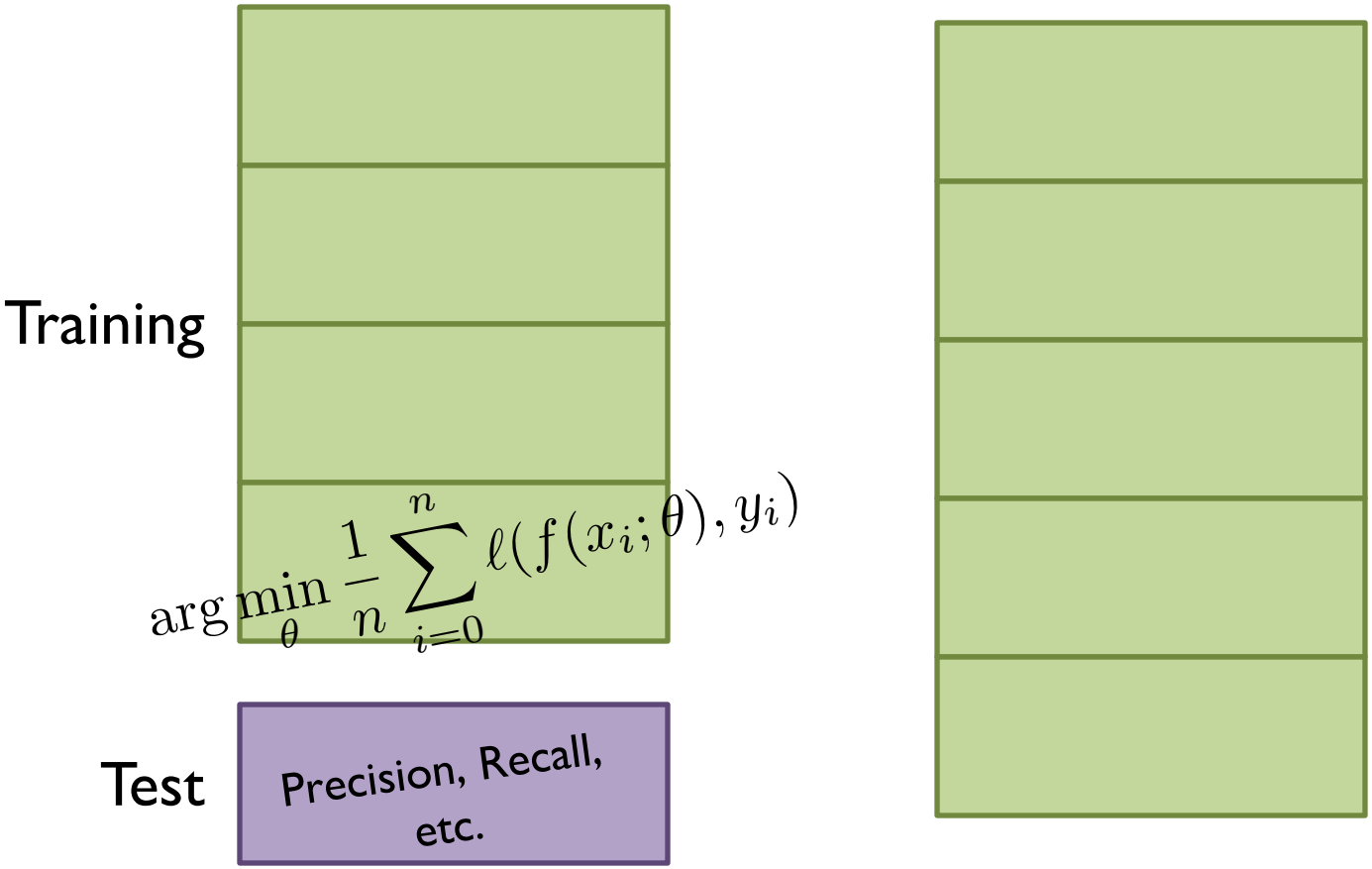




Source: Davis and Goadrich. (2006) The Relationship Between Precision-Recall and ROC curves

Training/Testing Splits

Often, benchmark datasets



What happens if you need more?

Cross-Validation

"Prospective" Evaluations

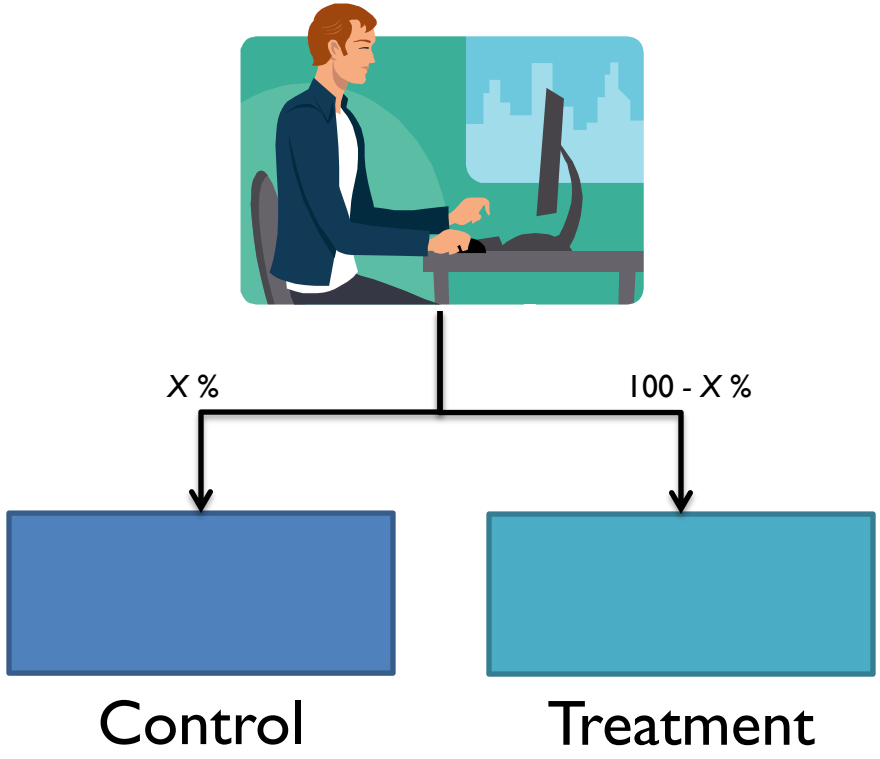
Often, internal batch evaluations

Why is this better?

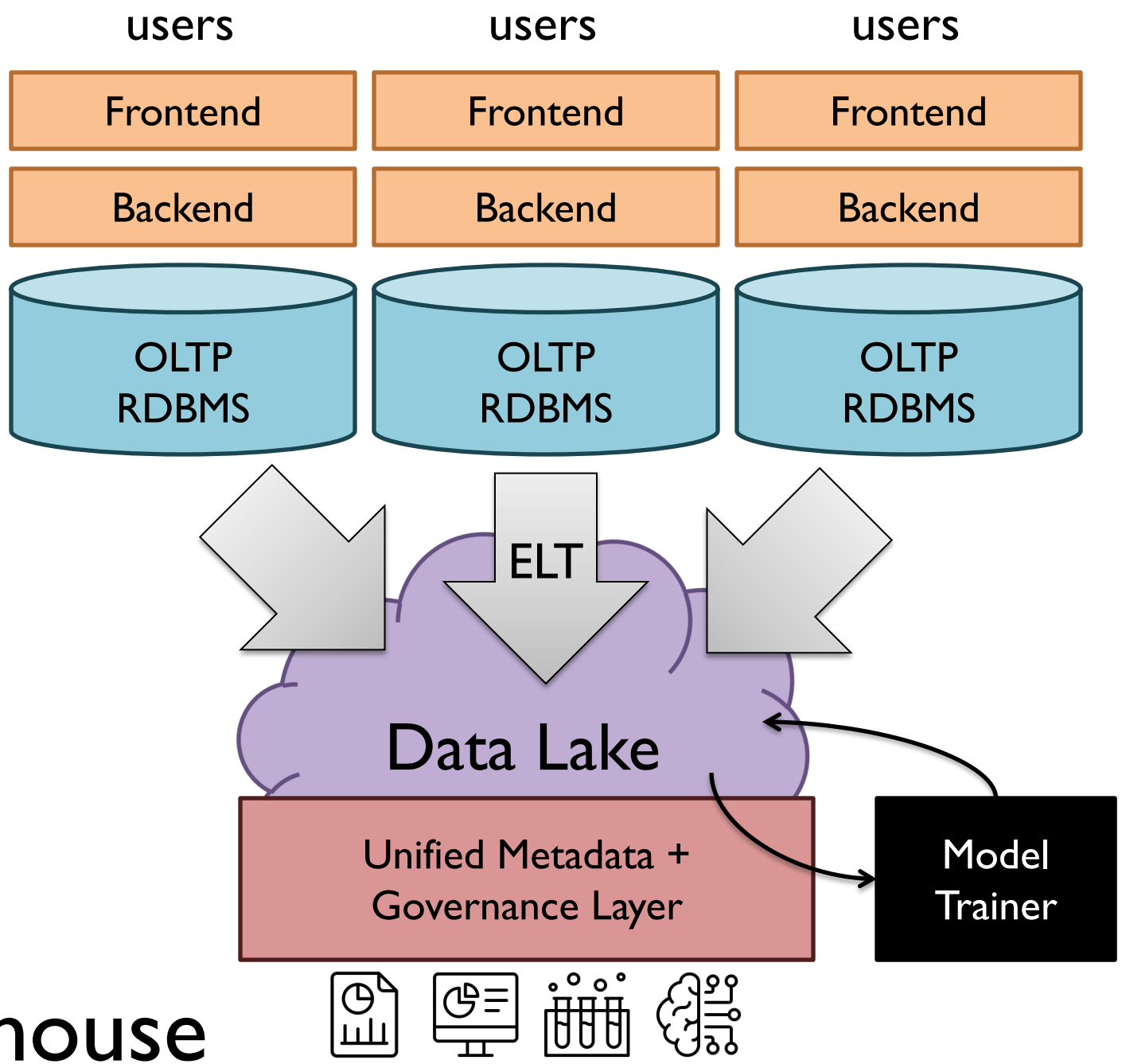
Where does this happen?

(Not ready for this yet…)

A/B Testing



Gather metrics, compare alternatives


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
Data Lake
Unified Metadata + Governance Layer
Model Trainer

Lakehouse

Gather training data

Train model

Deploy model

How do you know if it's better?

But if it's not?


Chance
GO DIRECTLY
TO JAIL
DO NOT PASS GO, DO NOT COLLECT $200
© 1936 PARKER BROTHERS

Components of a ML Solution

Data
Features
Model
Optimization

Gather training data

Train model

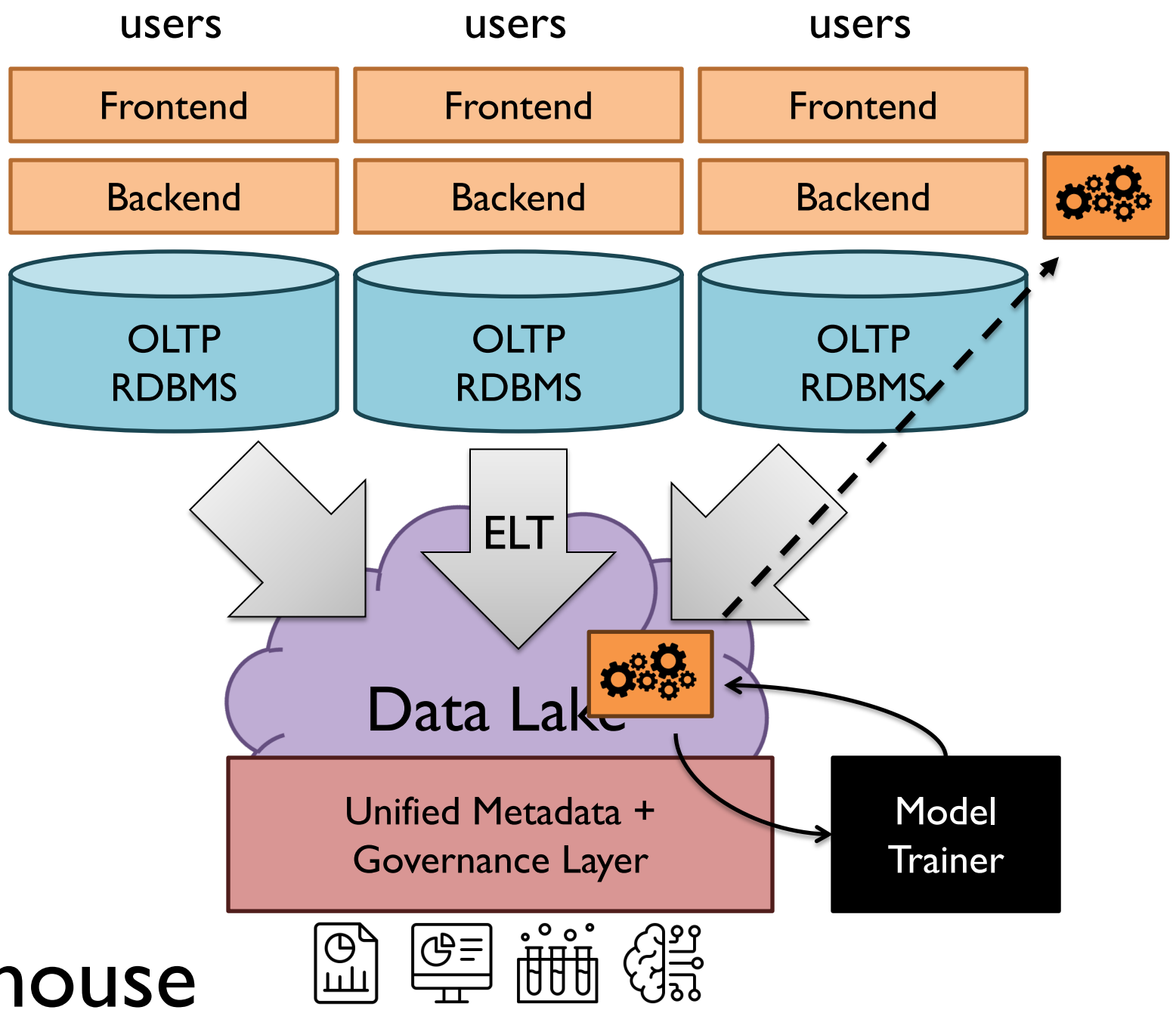
Deploy model

Okay, I really think it's better!

Gather training data

Train model

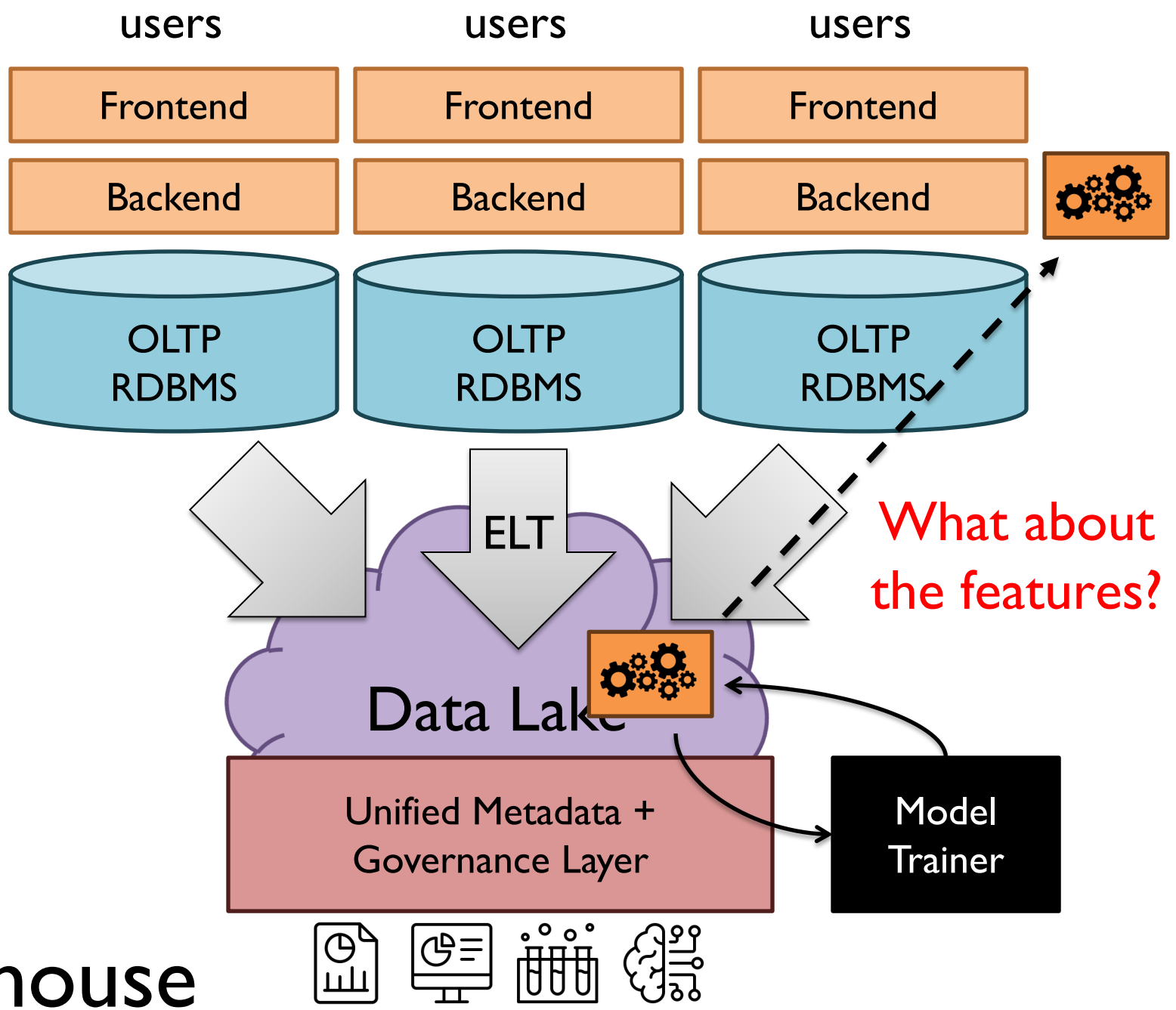
Deploy model


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
Data Lake
Unified Metadata + Governance Layer
Model Trainer

Lakehouse

How do you do it?

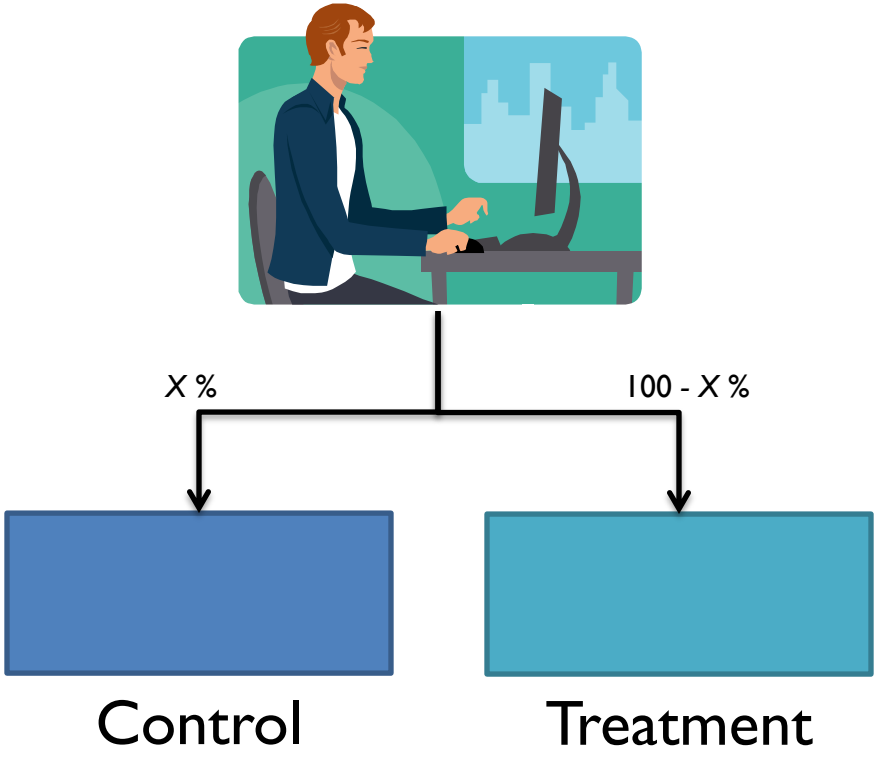
```
model.deploy()
```


users
users
users
Frontend
Frontend
Frontend
Backend
Backend
Backend
OLTP RDBMS
OLTP RDBMS
OLTP RDBMS
ELT
What about the features?
Data Lake
Unified Metadata + Governance Layer
Model Trainer

Lakehouse

Source: Wikipedia (Plumbing)

A/B Testing



Gather metrics, compare alternatives

Gather training data

Train model

Deploy model

Wait, you're not done yet!

Deployment is not a one-time thing…

How frequently is the model retrained?
What are its upstream dependencies?
What if training fails?
How frequently is the model deployed?
What if the deployment fails?
After deployment, what if the service stops working?
What if the quality degrades over time?

Gather training data

Train model

Deploy model

Goal for today: Dispel the myth of what a data scientist *actually* does.

Applied {ML, AI} Researcher, etc.

Source: Wikipedia (Hills)

Source: Wikipedia (Plumbing)