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

UNIVERSITY OF WATERLOO

Data Infrastructure for Machine Learning
(v1.00)

Week 6: October 7

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?

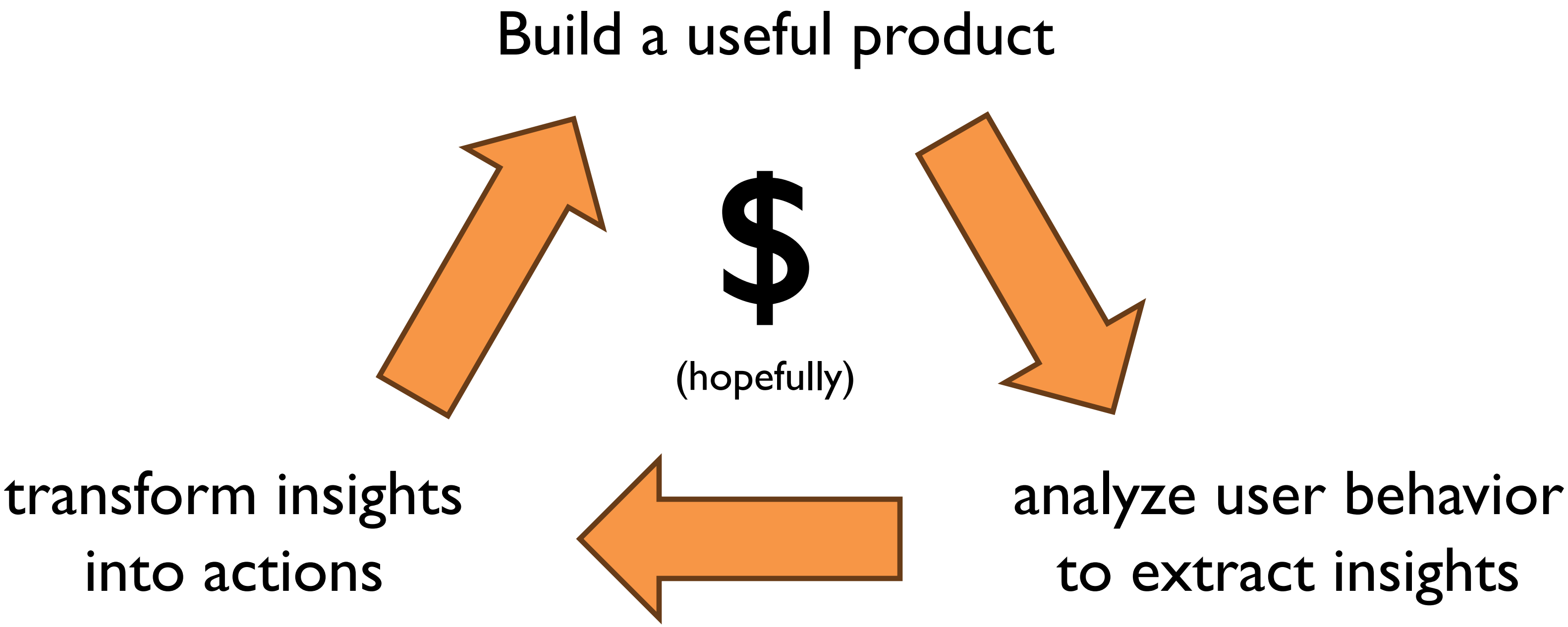
How is the supervised machine learning problem formulated?

What roles do data platforms and data engineering play?

Context…

The Data Flywheel

(a virtuous cycle)



Google. Facebook. Twitter. Amazon. Uber.

Context…

What's this course about?

The *infrastructure* that supports the data flywheel.

data platforms + data engineering

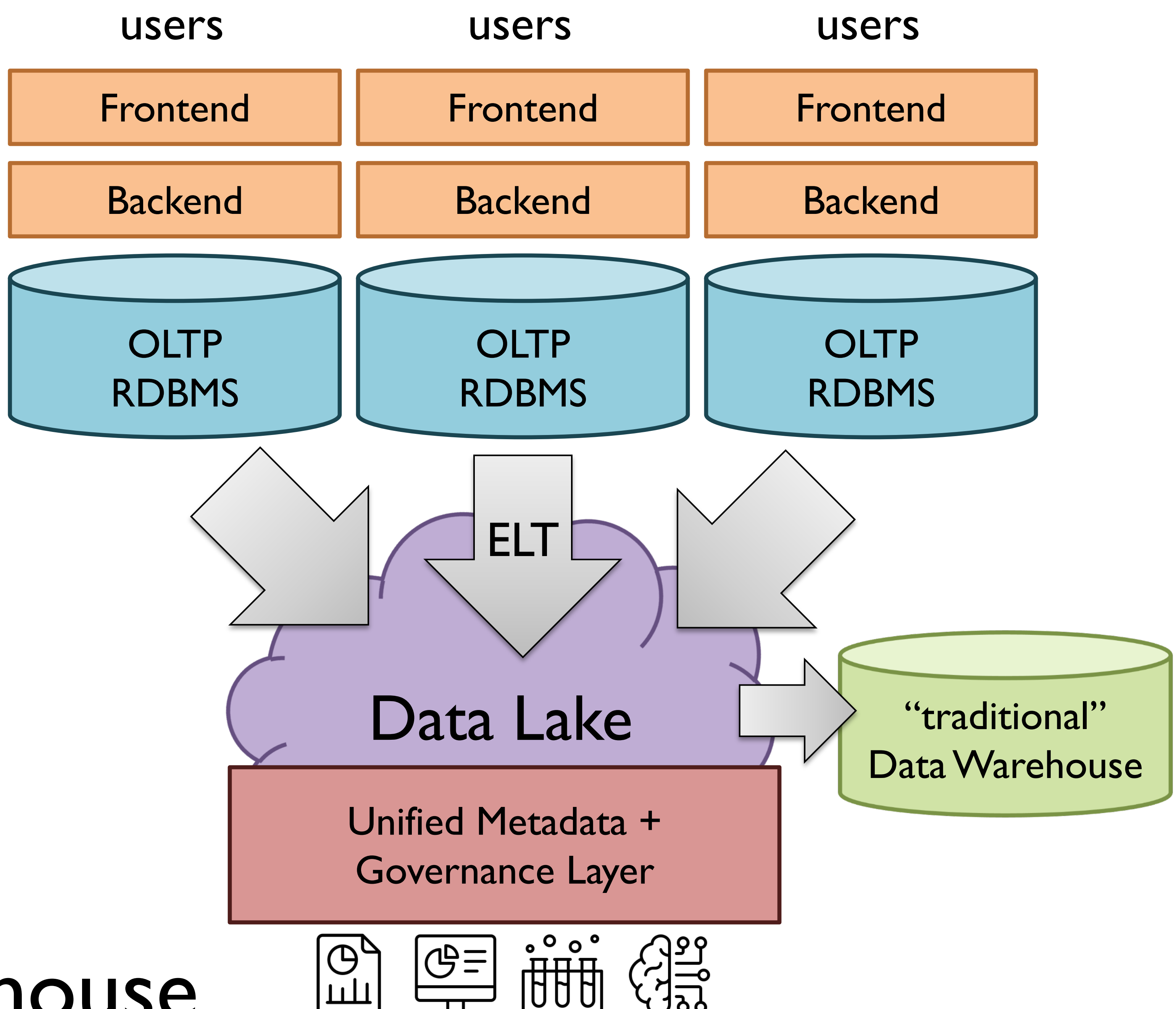
Context…

Transform Insights into Actions

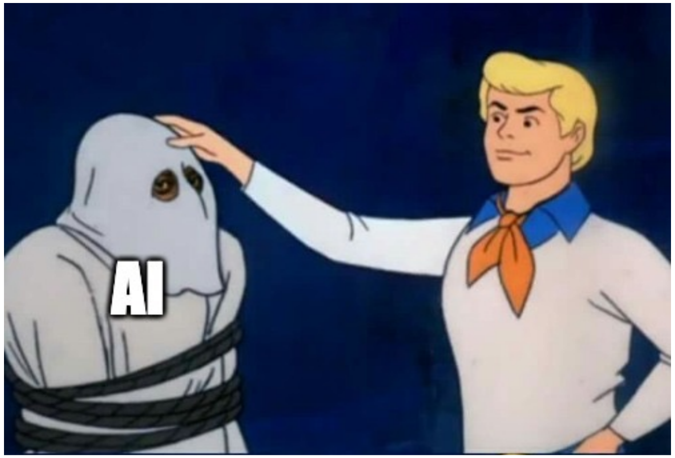
What does that really mean?

Report generation
Dashboards
} Business Intelligence

Ad hoc analyses
ML models
} Data Science


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

Lakehouse

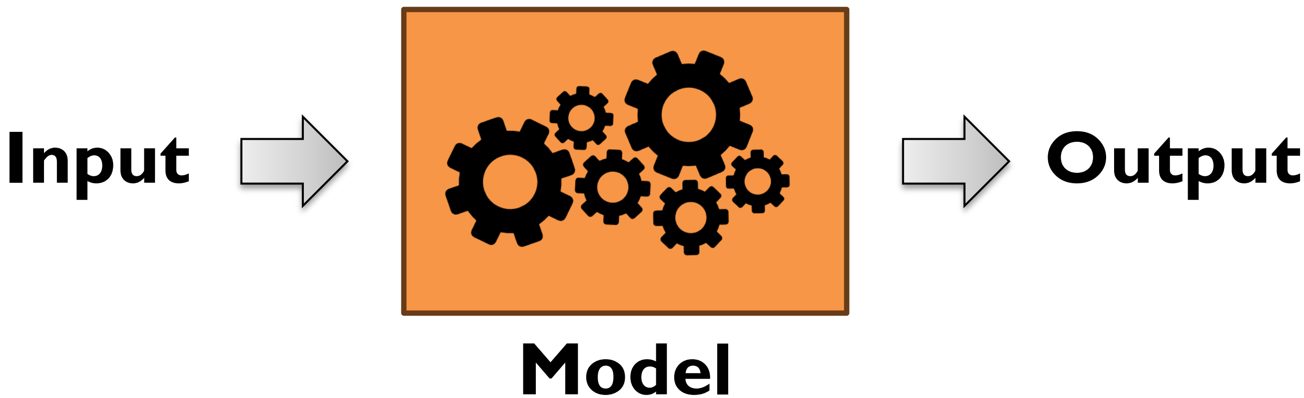

AI

When people talk about AI these days,
they really mean ML…


ML

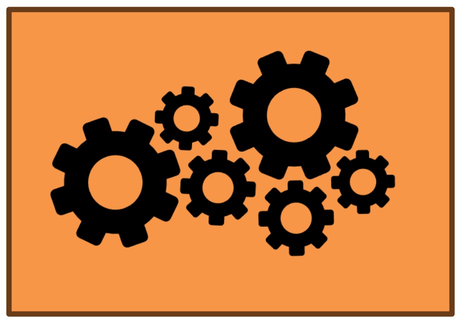
When people talk about ML these days,
they really mean supervised ML…

What's supervised machine learning?



(accomplishes *some* task)
(hopefully *well*)

Input



Model

Output

owl

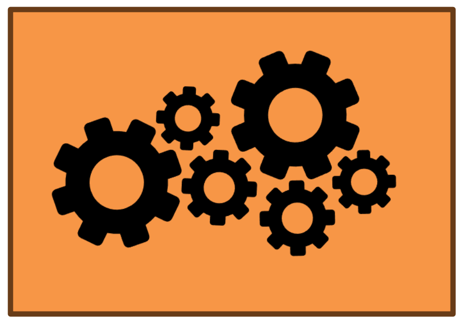
cat

(accomplishes *some* task)
(hopefully *well*)

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 😋

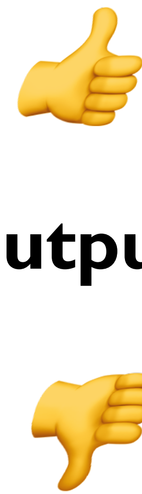
Input

Worst service I have ever experienced! Waited 25 mins for our waitress to come to our table once seated, no apology! Waited another 30 mins for our drinks, which we had to ask about.



Model

Output

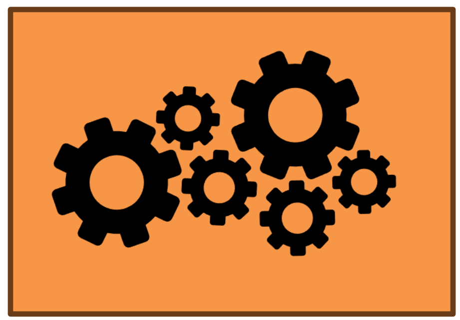


(accomplishes *some* task)
(hopefully *well*)

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 😋

Input

Worst service I have ever experienced! Waited 25 mins for our waitress to come to our table once seated, no apology! Waited another 30 mins for our drinks, which we had to ask about.

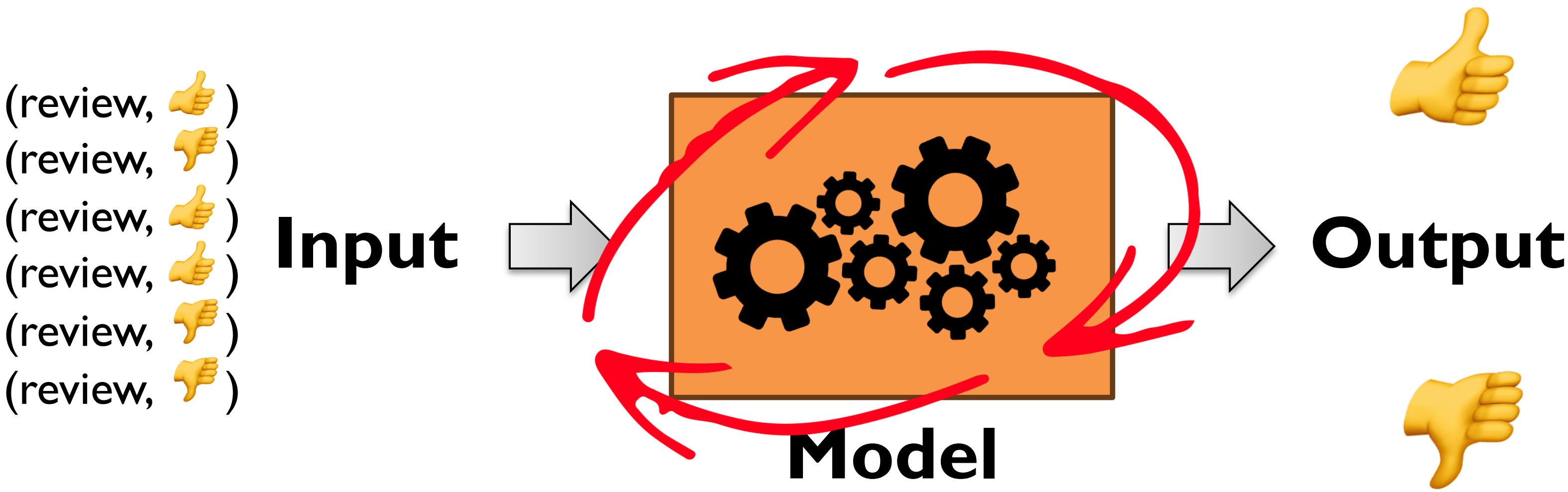


Model

Output

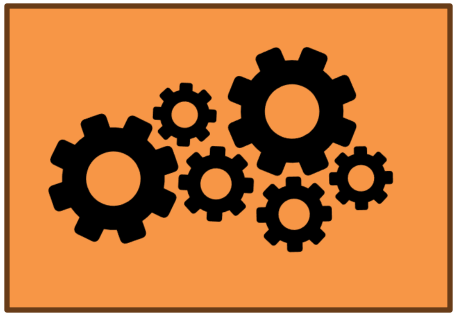
How do you build such a model?

Model *learns* from the data



Machine learning algorithm
adjusts the model *parameters*

How do you build such a model?



Model

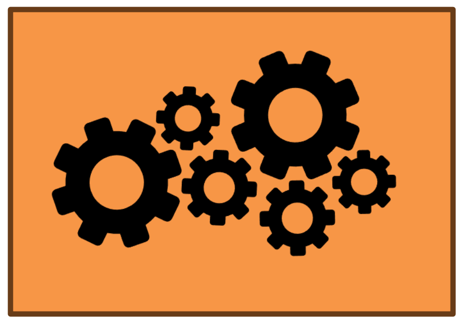
Machine learning algorithm
adjusts the model *parameters*

What does that *actually* mean?

Typically, the model takes a parametric form:

model parameters

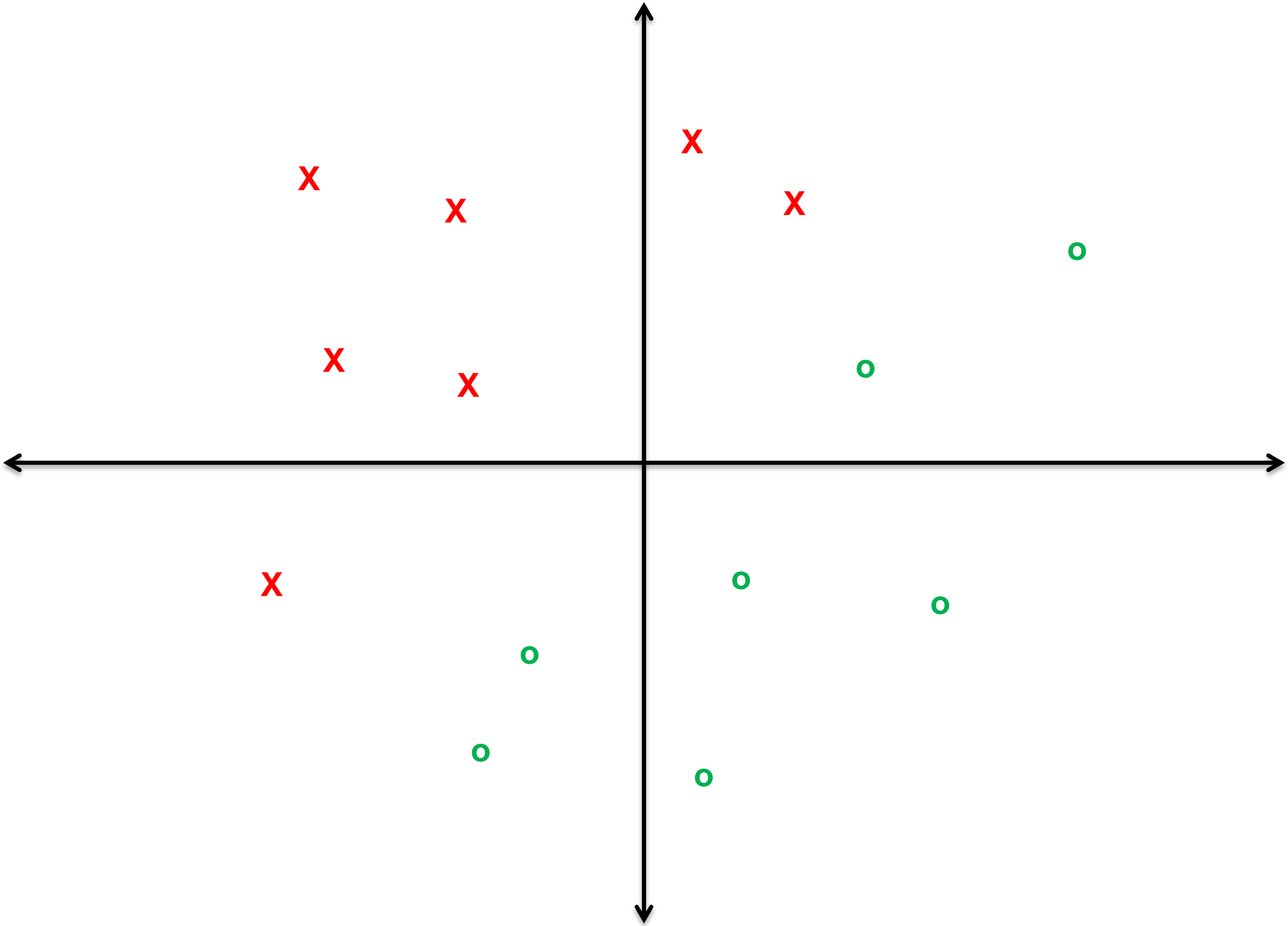
$$f(x_i;\theta)$$



Model

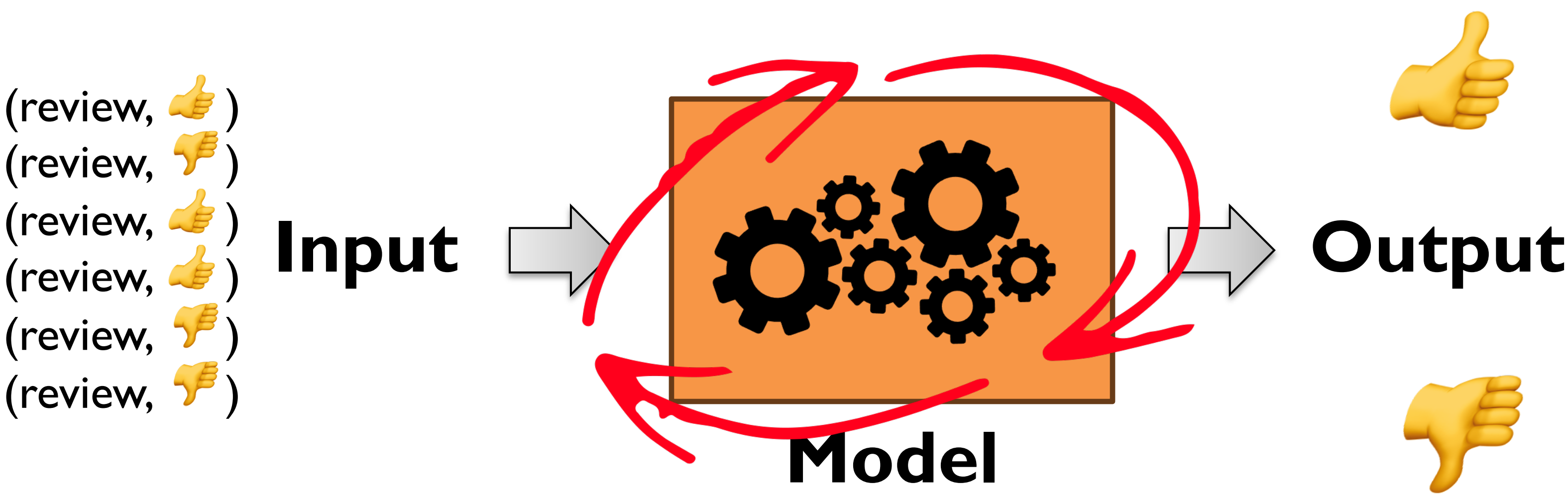
Machine learning algorithm
adjusts the model *parameters*

What does that *actually* mean?

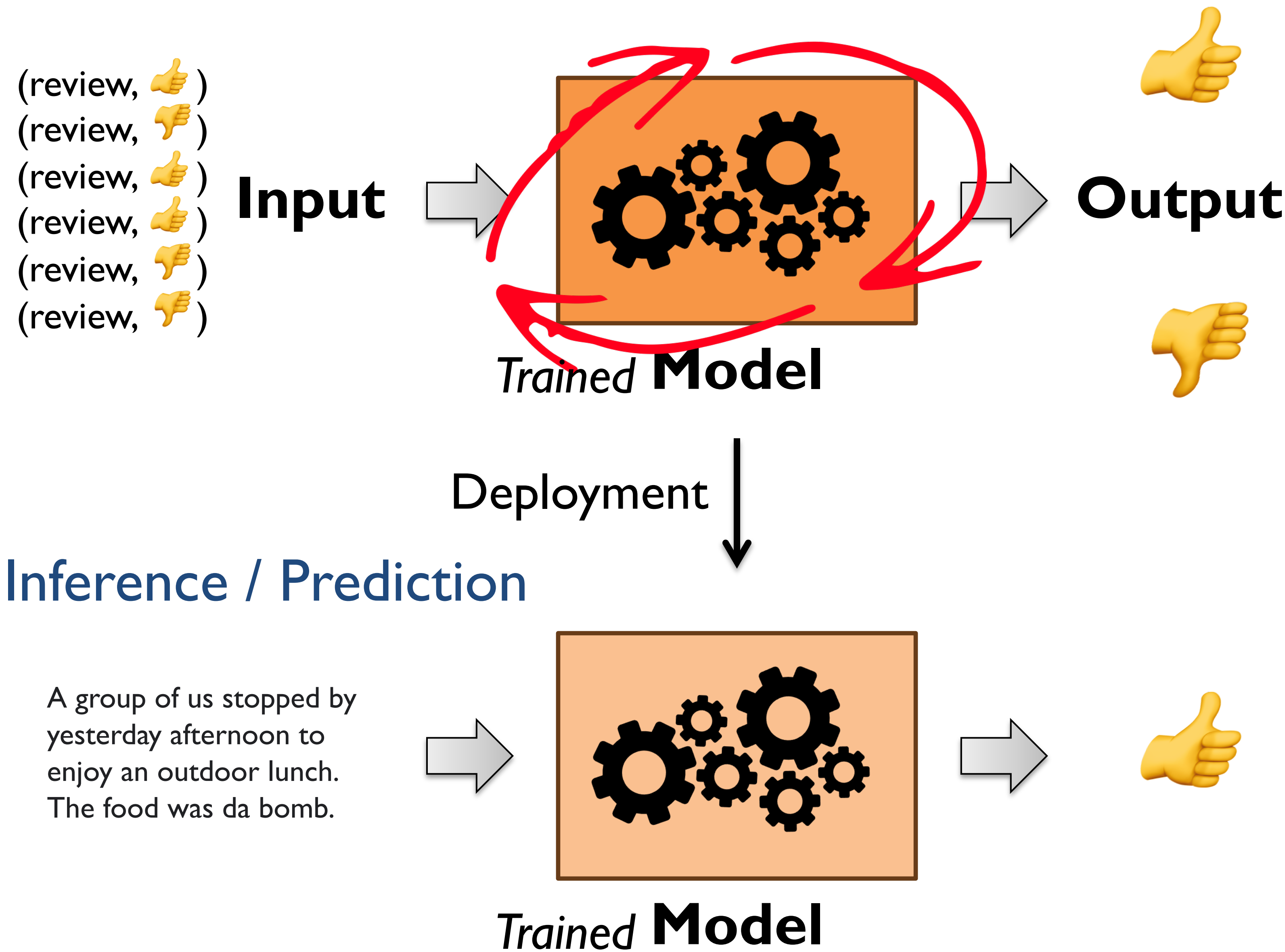


Parametric form for a linear model?

Model *learns* from the data



Machine learning algorithm
adjusts the model *parameters*


(review, 👍)
(review, 👎)
(review, 👍)
(review, 👍)
(review, 👎)
(review, 👎)
Input
Output
Trained Model
Deployment
Inference / Prediction
A group of us stopped by yesterday afternoon to enjoy an outdoor lunch. The food was da bomb.
Trained Model

What’s supervised machine learning?

Gather training data

Train model

Deploy model

That's it?

Caveats

“traditional” machine learning (i.e., *not* LLMs)

focus on data infrastructure (i.e., *not* ML/AI)

Supervised Machine Learning

The general problem of function induction given sample instances of input and output

Focus today

Classification: output draws from finite discrete labels

Regression: output is a continuous value

This is not meant to be an exhaustive treatment of machine learning!

Supervised *Binary* Classification

Restrict output label to be *binary*

Yes/No

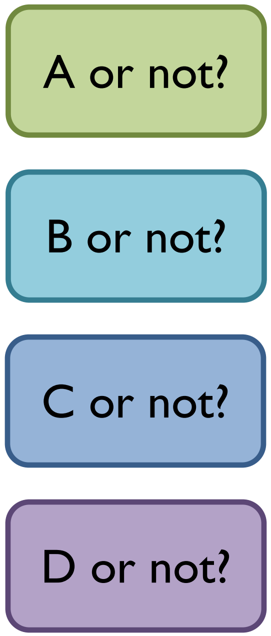
1/0

Binary classifiers form primitive
building blocks for multi-class problems…

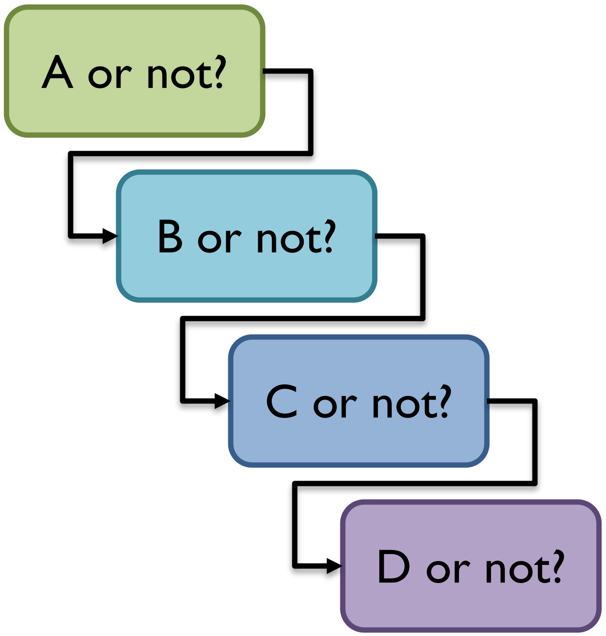
Binary Classifiers as Building Blocks

Example: four-way classification

One vs. rest classifiers



Classifier cascades



Components of an ML Solution

Data

Features

Model

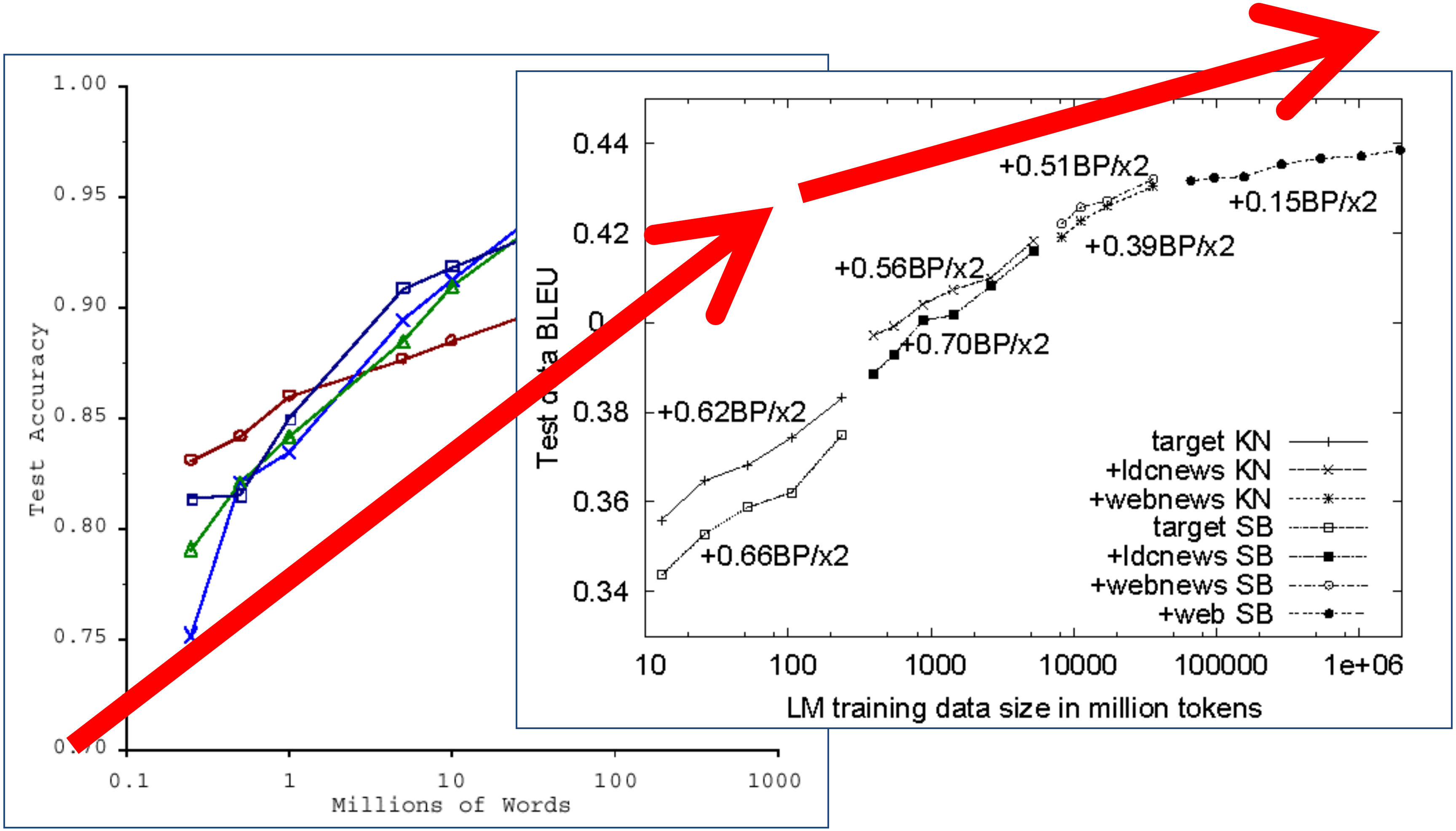
Optimization

logistic regression, naïve Bayes, SVM, random forests, neural networks, etc.

gradient descent, stochastic gradient descent, L-BFGS, etc.

What "matters" the most?

No data like more data!



(Banko and Brill, ACL 2001)
(Brants et al., EMNLP 2007)

Components of a ML Solution

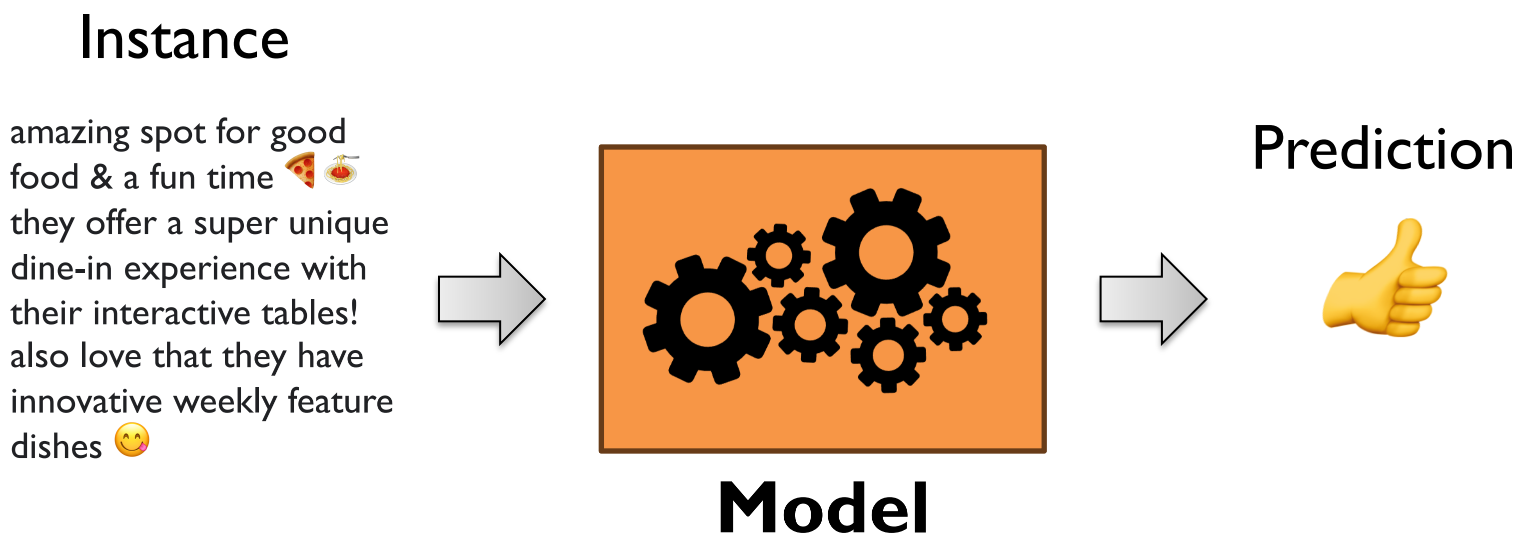
Data
Features
Model
Optimization

What's the focus of data engineering?

Components of a ML Solution

Data
Features
Model
Optimization

What's the focus of data engineering?


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
👍

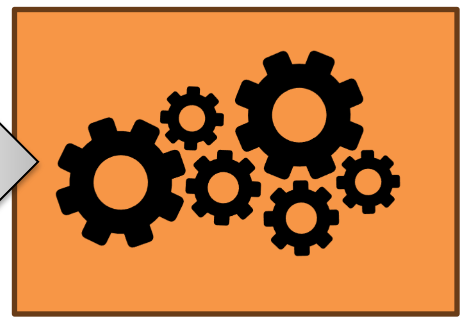
Instance

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 😋

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

Feature Vector

$$f : X \rightarrow Y$$



Model

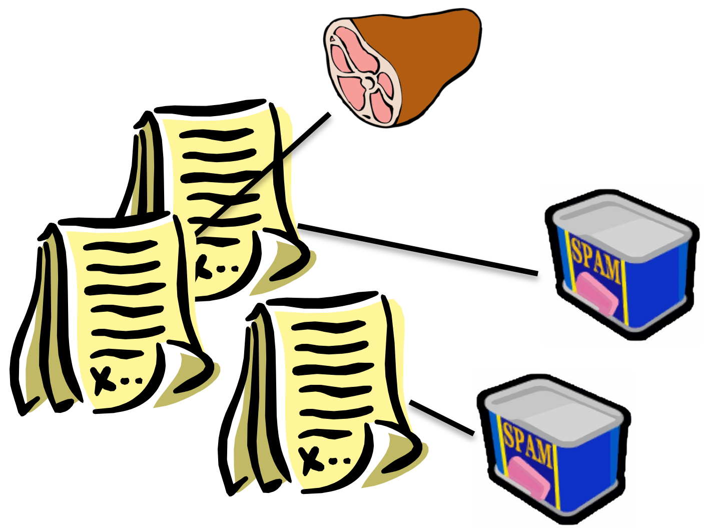
Prediction

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

Examples of features?

Who comes up with the features? How?

Features: Spam Classification



Instances (emails) are represented in terms of features:

- "Dense" features: sender IP, timestamp, # of recipients, length of message, etc.
- "Sparse" features: contains the term "viagra" in message, contains "URGENT" in subject, etc.

Limits of *Supervised* Classification?

Isn't gathering labels a bottleneck?

"Found" data

User behavior logs

Crowdsourcing

Semi-supervised techniques

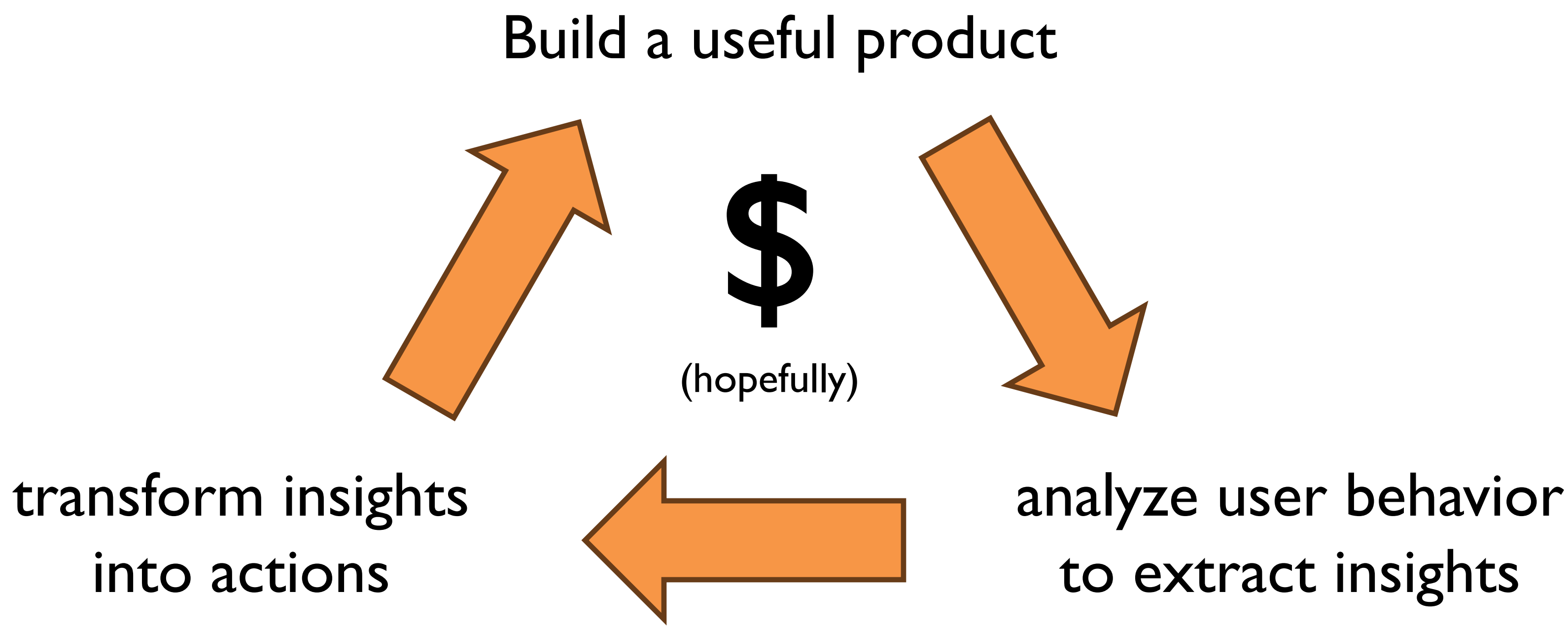
Self-supervised techniques

The virtuous cycle of data-driven products

This…

The Data Flywheel

(a virtuous cycle)



Google. Facebook. Twitter. Amazon. Uber.

Gather training data
Train model
Deploy model

"Data infrastructure for ML": What does that mean?

Raw data
(e.g., from ELT)

↓

High-quality training data

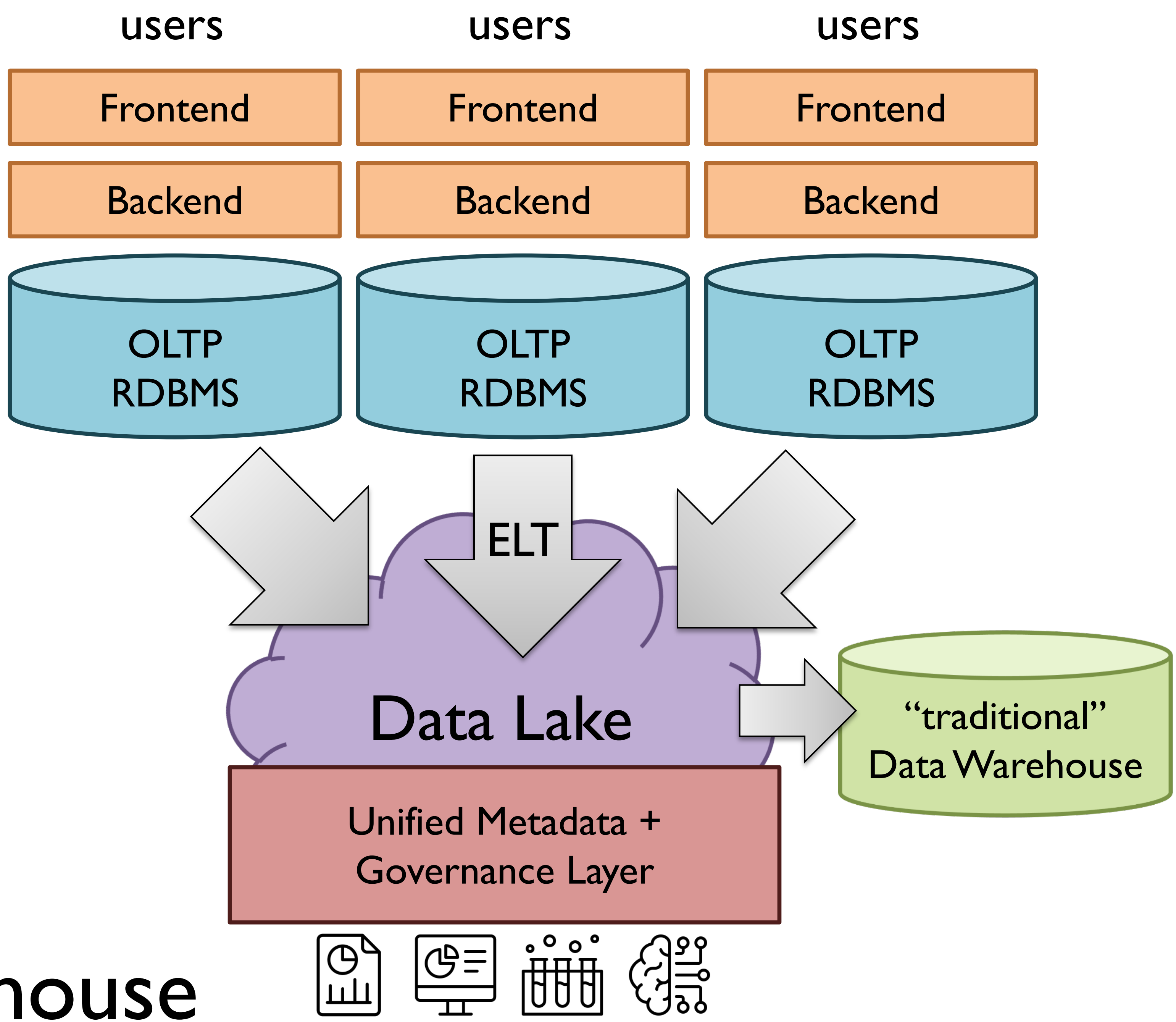
GI/GO

Gather training data

Train model

Deploy model

“Data infrastructure for ML”: What does that mean?


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

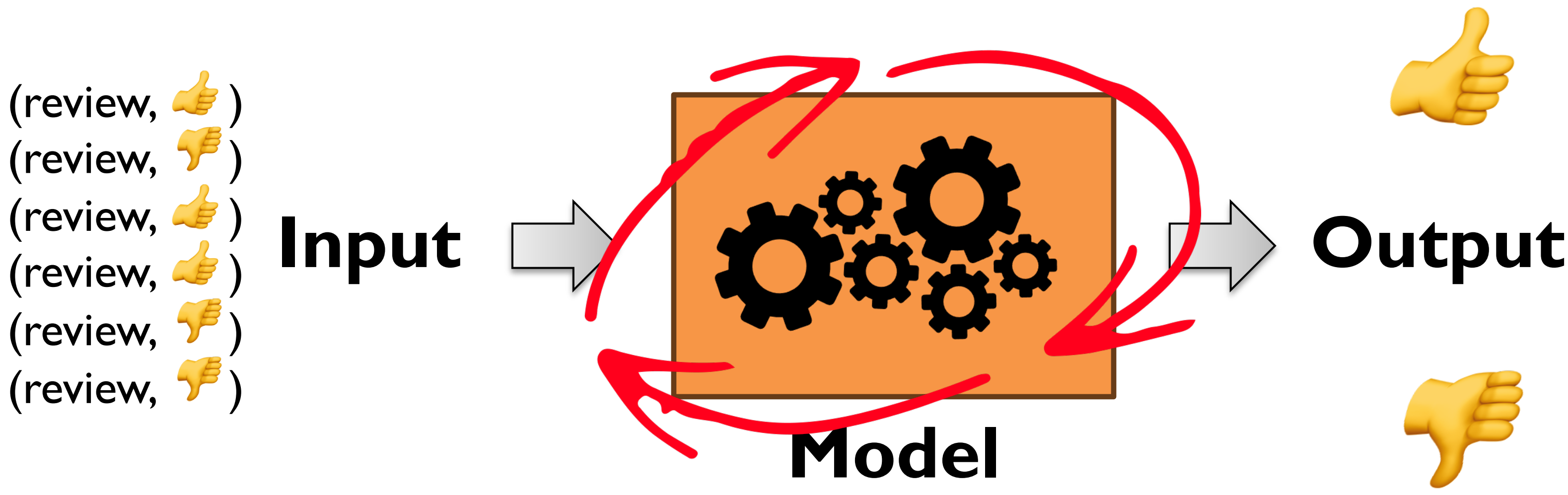
Lakehouse

Gather training data

Train model

Deploy model

Model *learns* from the data



Machine learning algorithm
adjusts the model *parameters*

The Task

label

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

feature vector

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

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

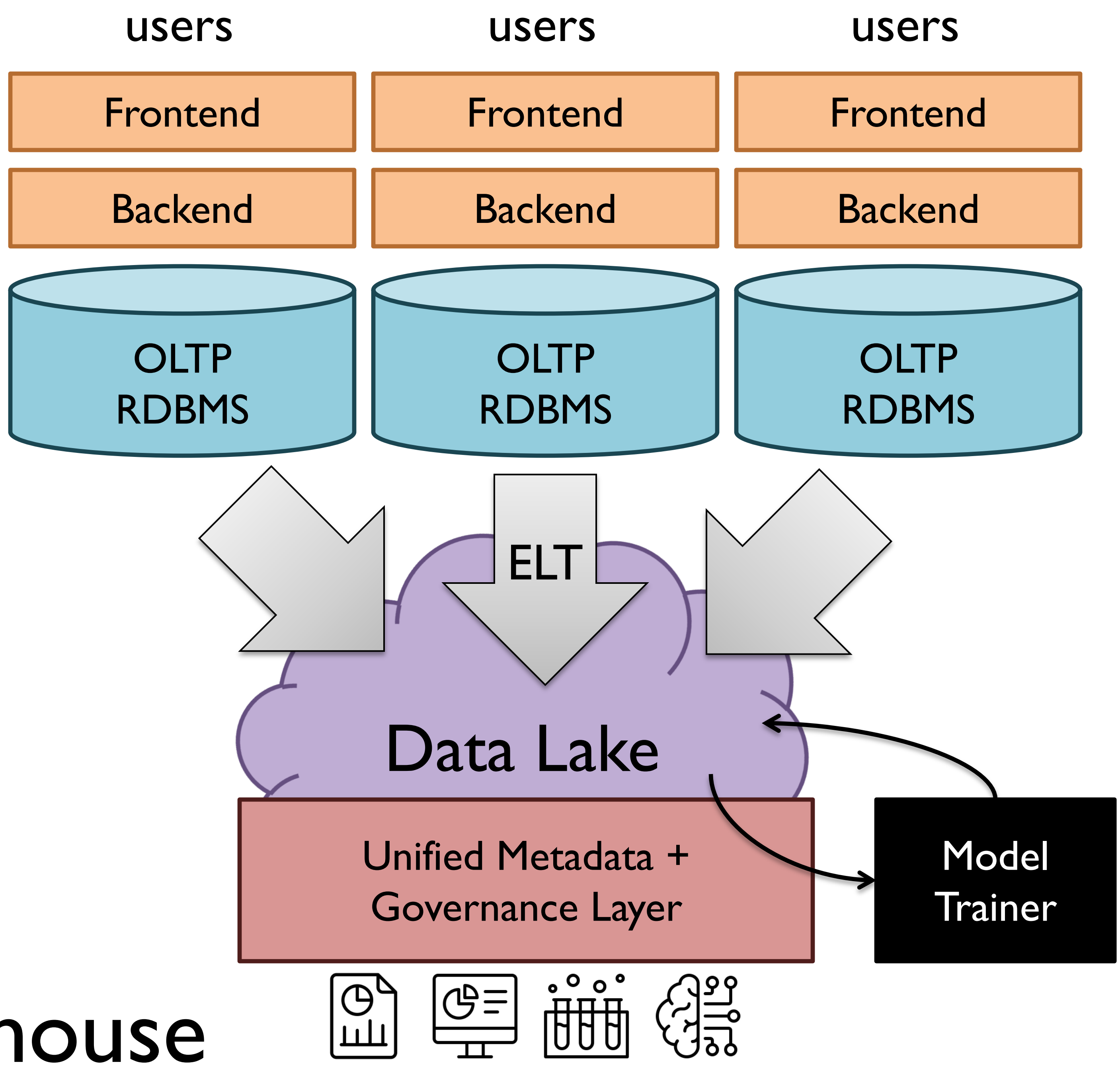
Typically, we consider functions of a parametric form:

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

model parameters

How do we do it?

Use `sklearn`: `model.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

How do we do it?

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

For many organizations, that (*really*) is the end of the story!

(but let's open up the black box…)

Key insight: machine learning as an optimization problem!

(closed form solutions generally not possible)

Gradient Descent: Preliminaries

Rewrite:

$$\arg\min_{\theta} \frac{1}{n}\sum_{i=0}^{n} \ell(f(\mathrm{x}_i;\theta), y_i) \quad \Longrightarrow \quad \arg\min_{\theta} L(\theta)$$

Compute gradient:

"Points" to fastest increasing "direction"

$$\nabla L(\theta) = \left[\frac{\partial L(\theta)}{\partial w_0}, \frac{\partial L(\theta)}{\partial w_1}, \ldots \frac{\partial L(\theta)}{\partial w_d}\right]$$

So, at any point: *

$$\mathrm{b} = \mathrm{a} - \gamma \nabla L(\mathrm{a})$$

$$L(\mathrm{a}) \geq L(\mathrm{b})$$

* caveats

Gradient Descent: Iterative Update

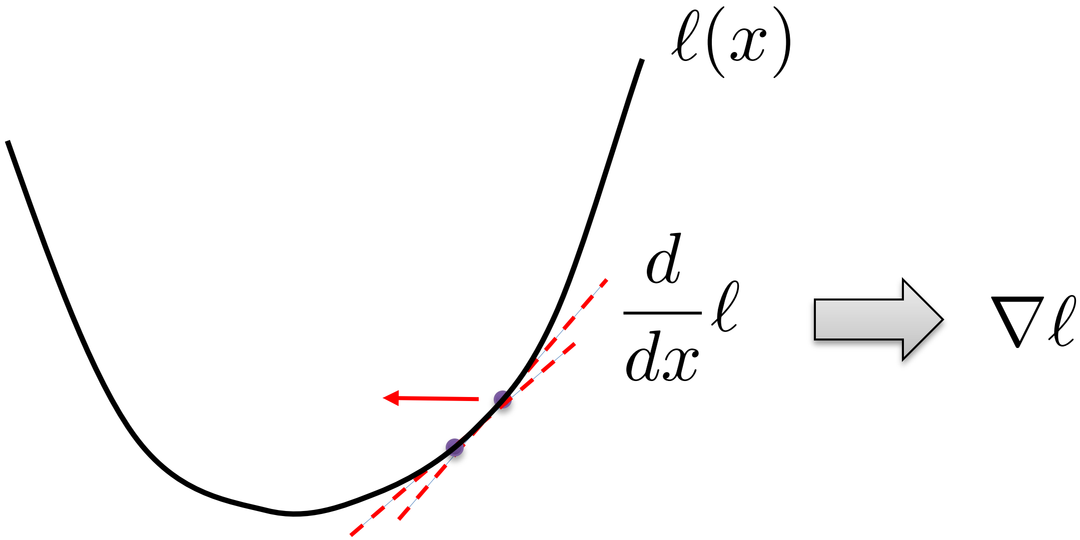
Start at an arbitrary point, iteratively update:

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla L(\theta^{(t)})$$

We have:

$$L(\theta^{(0)}) \geq L(\theta^{(1)}) \geq L(\theta^{(2)}) \ldots$$

Intuition behind the math…



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

New weights   Old weights

Update based on gradient

Gradient Descent: Iterative Update

Start at an arbitrary point, iteratively update:

$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla L(\theta^{(t)})$$

We have:

$$L(\theta^{(0)}) \geq L(\theta^{(1)}) \geq L(\theta^{(2)}) \ldots$$

Lots of details:

Figuring out the step size
Getting stuck in local minima
Convergence rate
…

Gradient Descent

Repeat until convergence:

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

Note, sometimes formulated as *ascent* but entirely equivalent

Gradient Descent

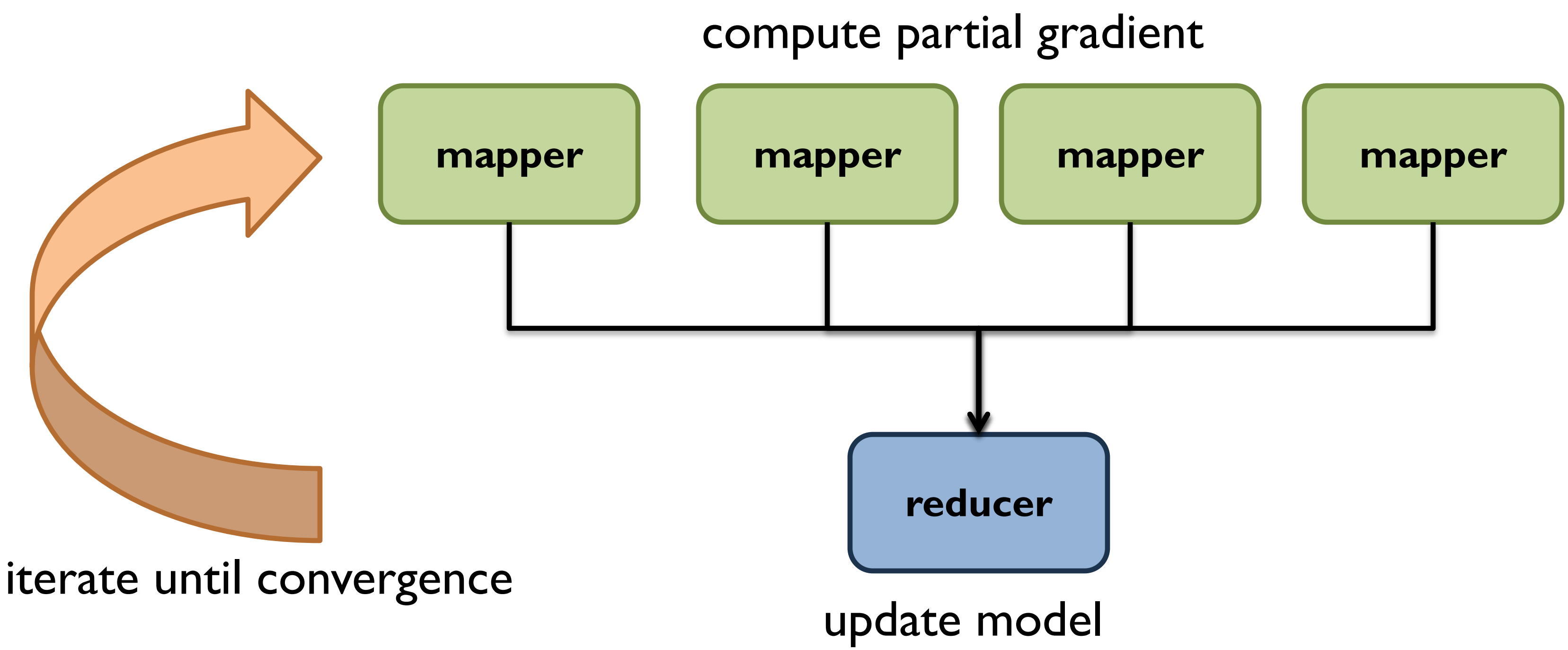
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{n} \sum_{i=0}^{n} \nabla \ell(f(\mathrm{x}_i; \theta^{(t)}), y_i)$$

Source: Wikipedia (Hills)

(Note, this isn't really done anymore, but here's how you would do it…)

MapReduce / Spark Implementation

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



<pause/>

Logistic Regression: Preliminaries

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

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

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

Define: $f(\mathrm{x}; \mathrm{w}) : \mathbb{R}^d \to \{0, 1\}$

$$f(\mathrm{x}; \mathrm{w}) = \begin{cases} 1 \text{ if } \mathrm{w} \cdot \mathrm{x} \geq t \\ 0 \text{ if } \mathrm{w} \cdot \mathrm{x} < t \end{cases}$$

Interpretation:

$$\ln \left[\frac{\Pr(y = 1 | \mathrm{x})}{\Pr(y = 0 | \mathrm{x})} \right] = \mathrm{w} \cdot \mathrm{x}$$

$$\ln \left[\frac{\Pr(y = 1 | \mathrm{x})}{1 - \Pr(y = 1 | \mathrm{x})} \right] = \mathrm{w} \cdot \mathrm{x}$$

Relation to the Logistic Function

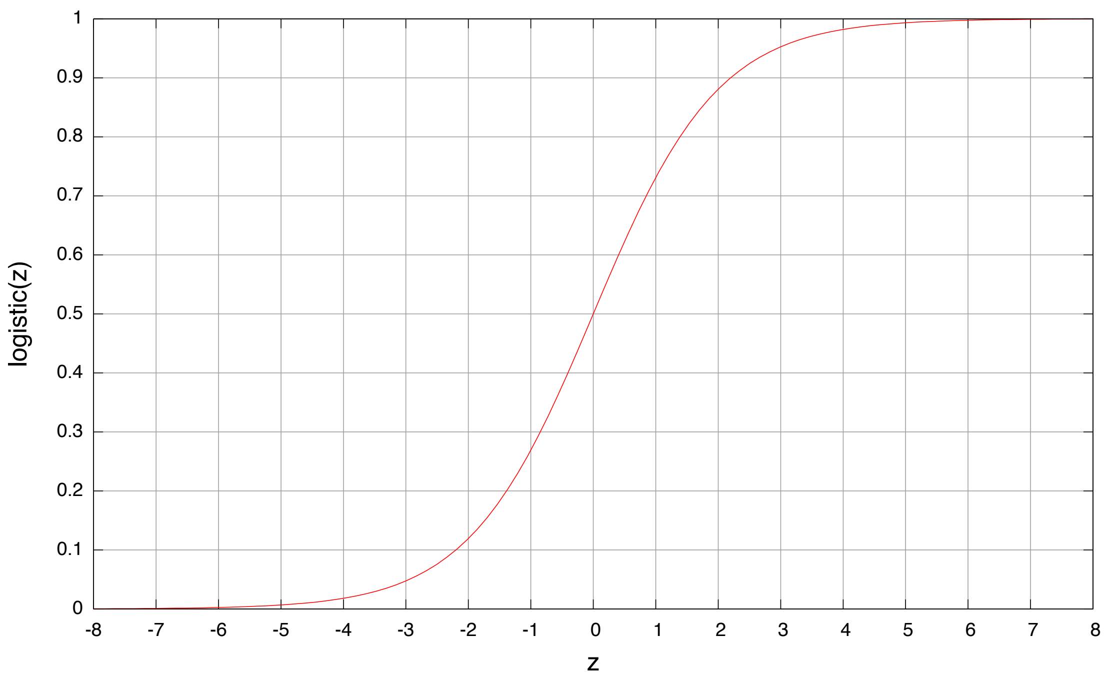
After some algebra:

$$\Pr\left(y=1|x\right) = \frac{e^{\mathbf{w}\cdot\mathbf{x}}}{1+e^{\mathbf{w}\cdot\mathbf{x}}}$$

$$\Pr\left(y=0|x\right) = \frac{1}{1+e^{\mathbf{w}\cdot\mathbf{x}}}$$

The logistic function:

$$f(z) = \frac{e^z}{e^z+1}$$



Training an LR Classifier

Maximize the conditional likelihood: $\arg\max_{\mathrm{w}} \prod_{i=1}^{n} \Pr(y_i|\mathrm{x}_i, \mathrm{w})$

Define the objective in terms of conditional *log* likelihood: $L(\mathrm{w}) = \sum_{i=1}^{n} \ln \Pr(y_i|\mathrm{x}_i, \mathrm{w})$

We know: $y \in \{0, 1\}$

So: $Pr(y|\mathrm{x}, \mathrm{w}) = Pr(y = 1|\mathrm{x}, \mathrm{w})^y Pr(y = 0|\mathrm{x}, \mathrm{w})^{(1-y)}$

Substituting:

$$L(\mathrm{w}) = \sum_{i=1}^{n} \Big(y_i \ln \Pr(y_i = 1|\mathrm{x}_i, \mathrm{w}) + (1 - y_i) \ln \Pr(y_i = 0|\mathrm{x}_i, \mathrm{w}) \Big)$$

LR Classifier Update Rule

Take the derivative:

$$L(\mathrm{w}) = \sum_{i=1}^{n} \Big(y_i \ln \Pr(y_i = 1 | \mathrm{x}_i, \mathrm{w}) + (1 - y_i) \ln \Pr(y_i = 0 | \mathrm{x}_i, \mathrm{w}) \Big)$$

$$\frac{\partial}{\partial \mathrm{w}} L(\mathrm{w}) = \sum_{i=0}^{n} \mathrm{x}_i \Big(y_i - \Pr(y_i = 1 | \mathrm{x}_i, \mathrm{w}) \Big)$$

General form of update rule:

$$\mathrm{w}^{(t+1)} \leftarrow \mathrm{w}^{(t)} + \gamma^{(t)} \nabla_{\mathrm{w}} L(\mathrm{w}^{(t)})$$

$$\nabla L(\mathrm{w}) = \left[\frac{\partial L(\mathrm{w})}{\partial w_0}, \frac{\partial L(\mathrm{w})}{\partial w_1}, \ldots \frac{\partial L(\mathrm{w})}{\partial w_d} \right]$$

Final update rule:

$$\mathrm{w}_i^{(t+1)} \leftarrow \mathrm{w}_i^{(t)} + \gamma^{(t)} \sum_{j=0}^{n} x_{j,i} \Big(y_j - \Pr(y_j = 1 | \mathrm{x}_j, \mathrm{w}^{(t)}) \Big)$$

Lots more details…

Want more?
Take a real machine-learning course!

Final update rule:

$$\mathrm{w}_i^{(t+1)} \leftarrow \mathrm{w}_i^{(t)} + \gamma^{(t)} \sum_{j=0}^{n} x_{j,i}\Big(y_j - \Pr(y_j = 1|\mathrm{x}_j, \mathrm{w}^{(t)})\Big)$$

<pause/>

Batch vs. Online

(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

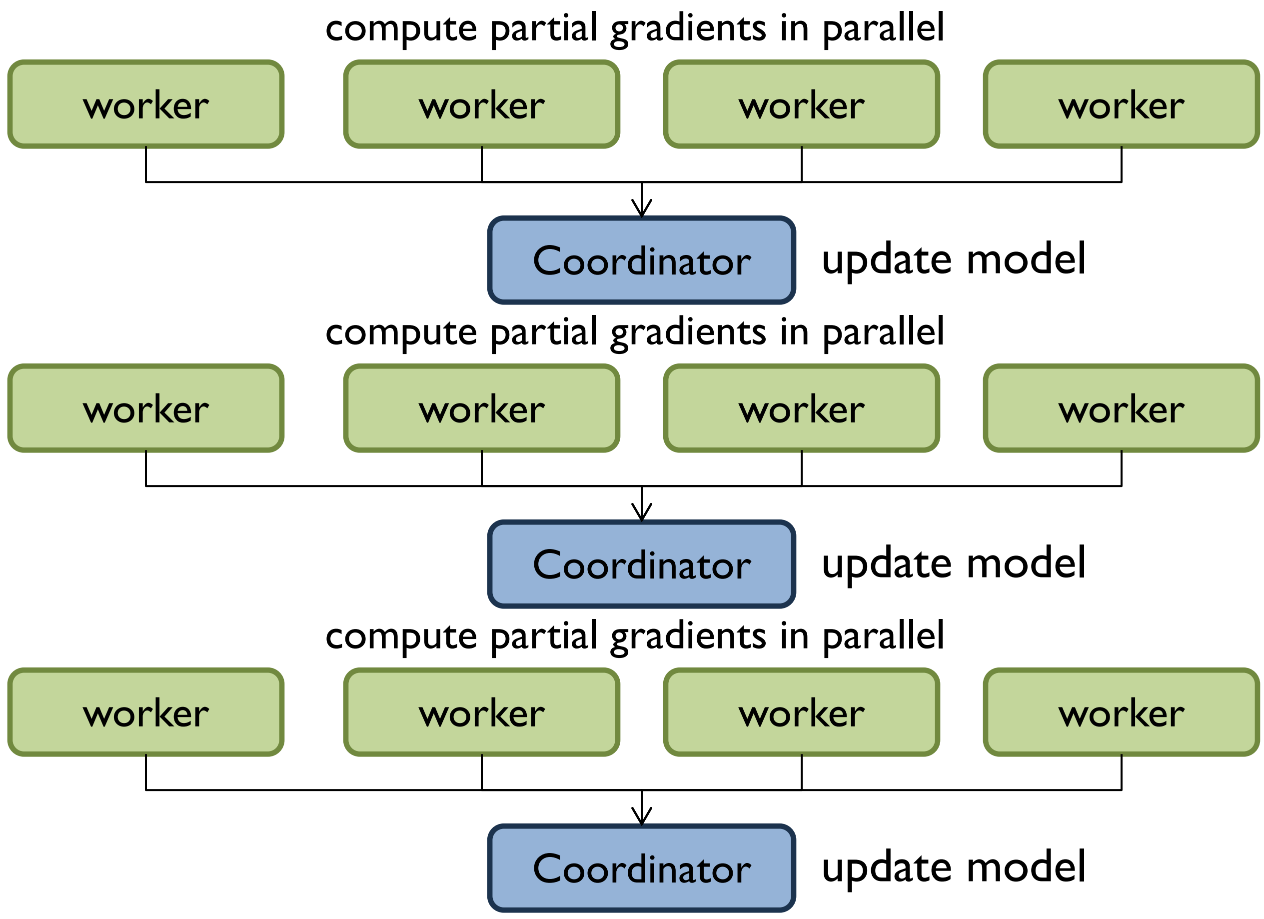

(Batch) Gradient Descent
Source: Wikipedia (Hills)

Stochastic Gradient Descent

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

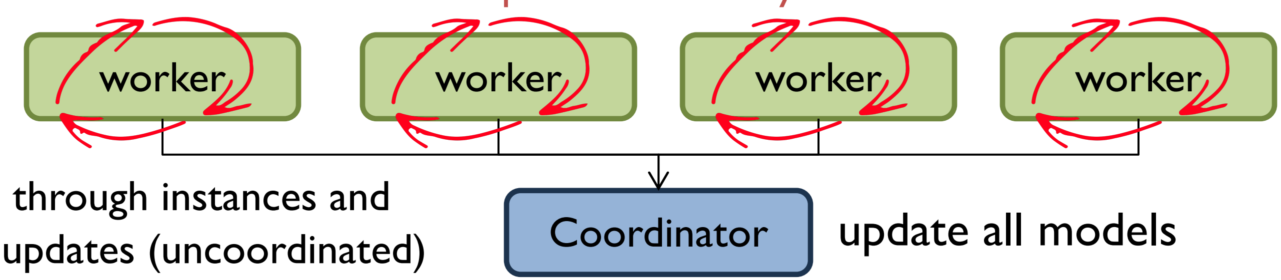
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(\mathrm{x}; \theta^{(t)}), y)$$

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

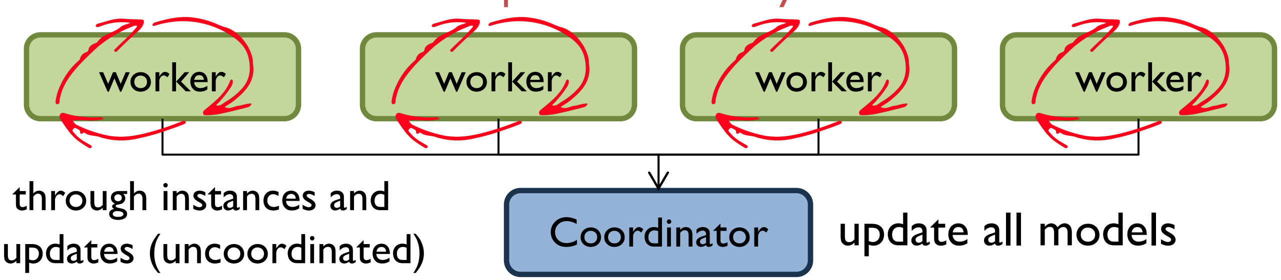
Solution: mini-batches

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

Perform gradient descent on each mini-batch

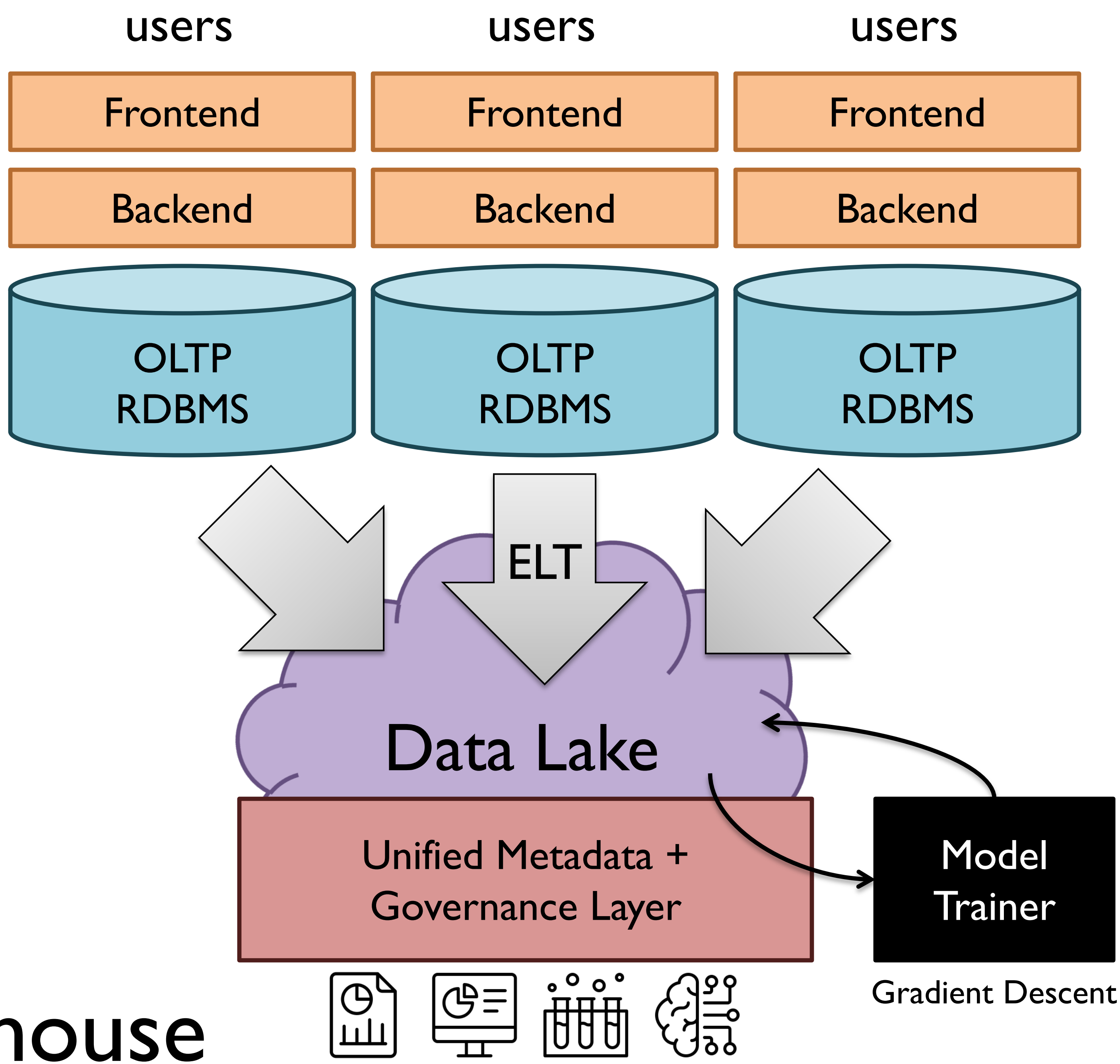
Update model after processing each mini-batch

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
Gradient Descent

Lakehouse

Source: Wikipedia (Hills)