UNIVERSITY OF WATERLOO

Big Data Infrastructure

CS 489/698 Big Data Infrastructure (Winter 2017)

Week 8: Data Mining (2/4)

March 2, 2017

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

These slides are available at http://lintool.github.io/bigdata-2017w/

The Task

label

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

(sparse) 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

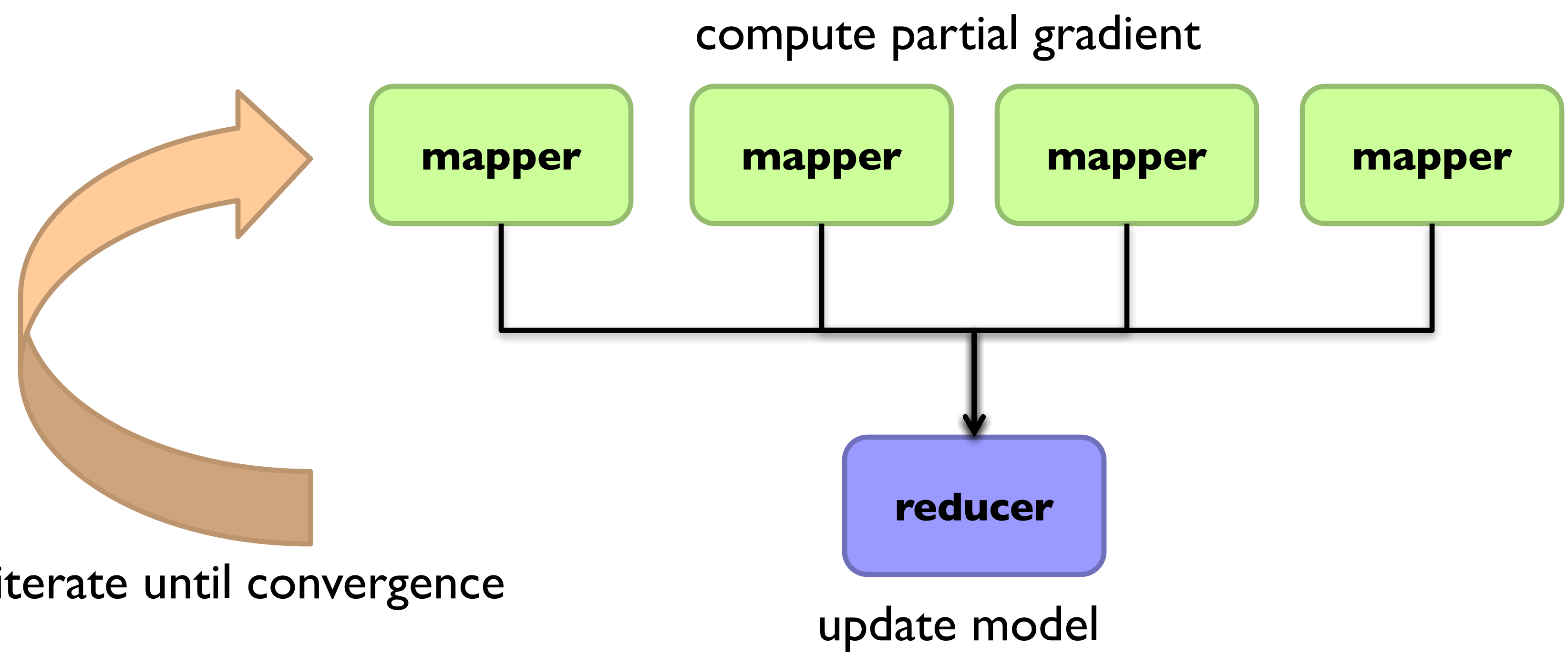
Gradient Descent

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

Source: Wikipedia (Hills)

MapReduce Implementation

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

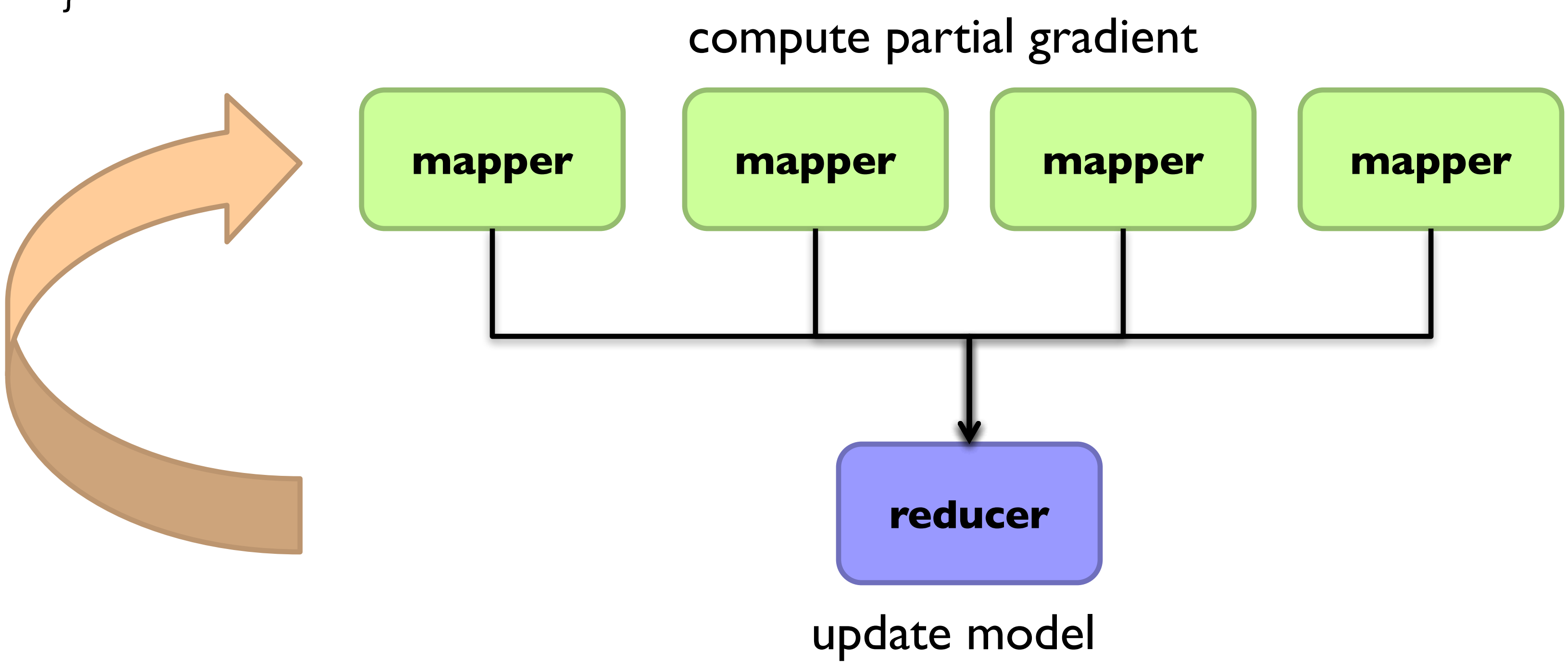


Spark Implementation

```
val points = spark.textFile(...).map(parsePoint).persist()

var w = // random initial vector
for (i <- 1 to ITERATIONS) {
  val gradient = points.map{ p =>
    p.x * (1/(1+exp(-p.y*(w dot p.x)))-1)*p.y
  }.reduce((a,b) => a+b)
  w -= gradient
}
```

What's the difference?



Source: Wikipedia (Hills)

Source: Wikipedia (Water Slide)

Batch vs. Online

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

In practice… just as good!
Opportunity to interleaving prediction and learning!

Practical Notes

Order of the instances important!

Most common implementation: randomly shuffle training instances

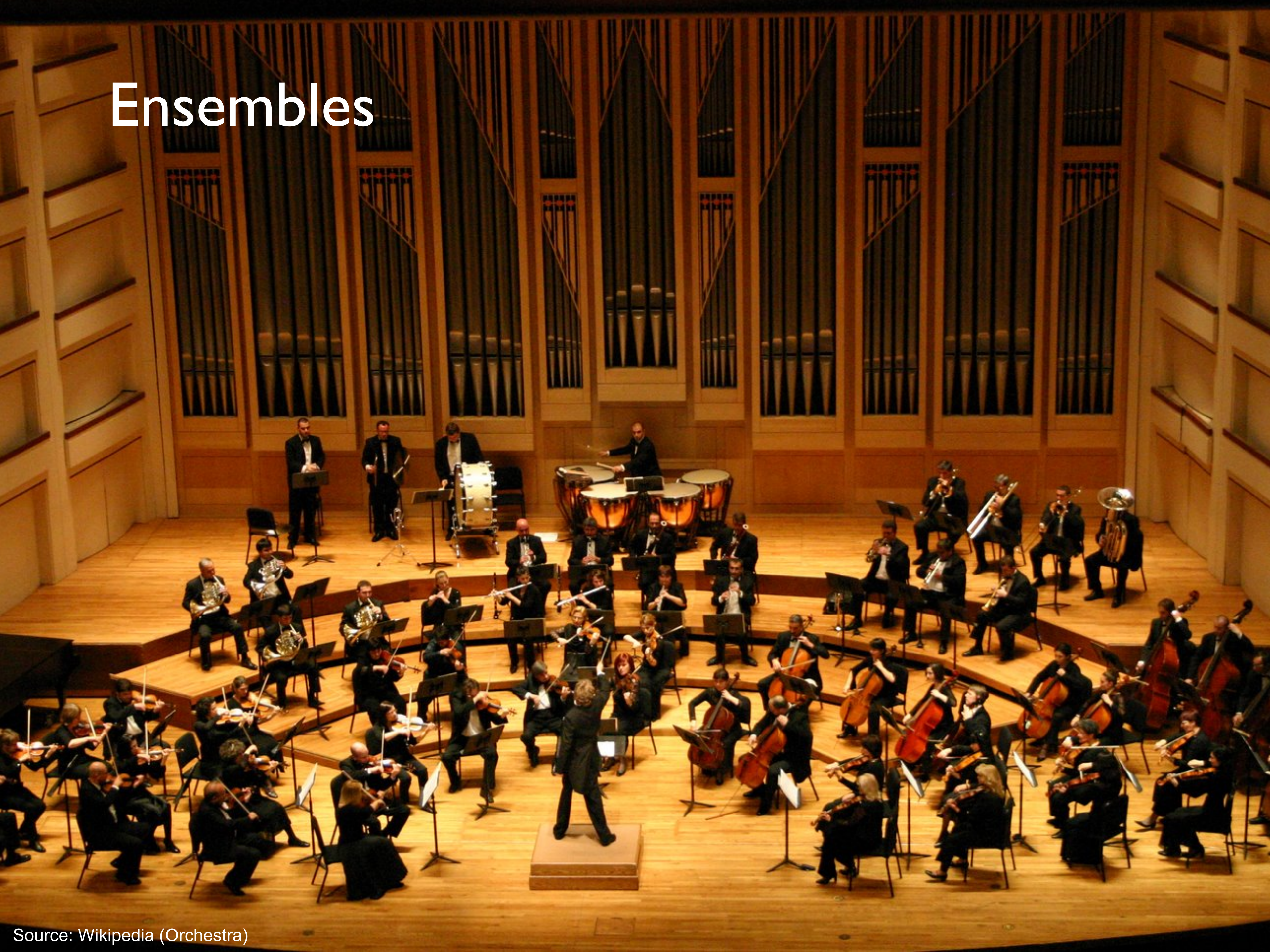
Single vs. multi-pass approaches

Mini-batching as a middle ground

We've solved the iteration problem!

What about the single reducer problem?

Ensembles



Source: Wikipedia (Orchestra)

Ensemble Learning

independent

Learn multiple models, combine results from different models to make prediction

Common implementation:

Train classifiers on different input partitions of the data

Embarrassingly parallel!

Combining predictions:

Majority voting

Simple weighted voting:

$$y = \arg\max_{y \in Y} \sum_{k=1}^{n} \alpha_k p_k(y|\mathrm{x})$$

Model averaging

…

Ensemble Learning

independent

Learn multiple models, combine results from different models to make prediction

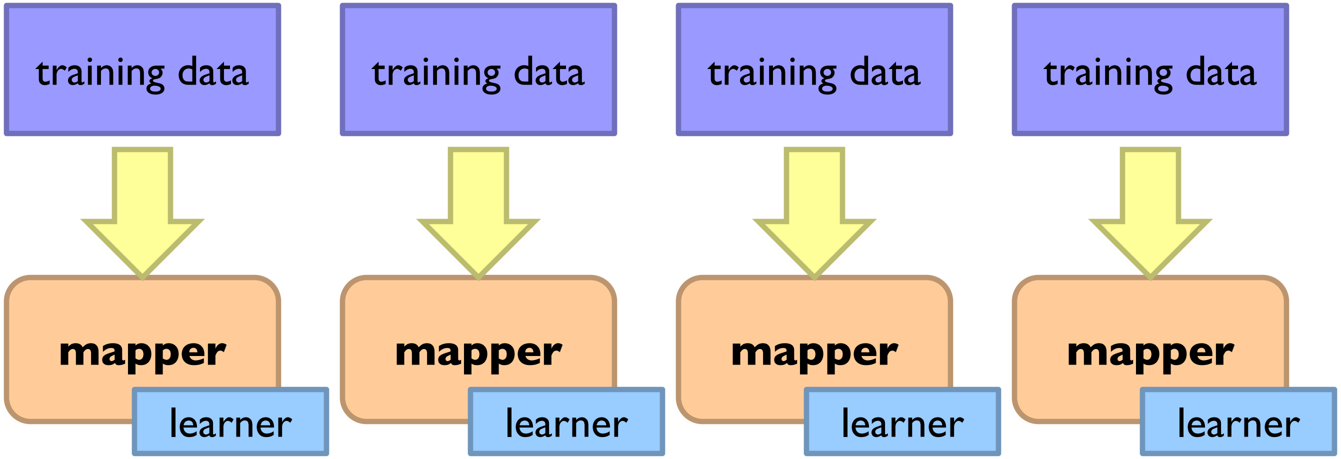
Why does it work?

If errors uncorrelated, multiple classifiers being wrong is less likely

Reduces the variance component of error

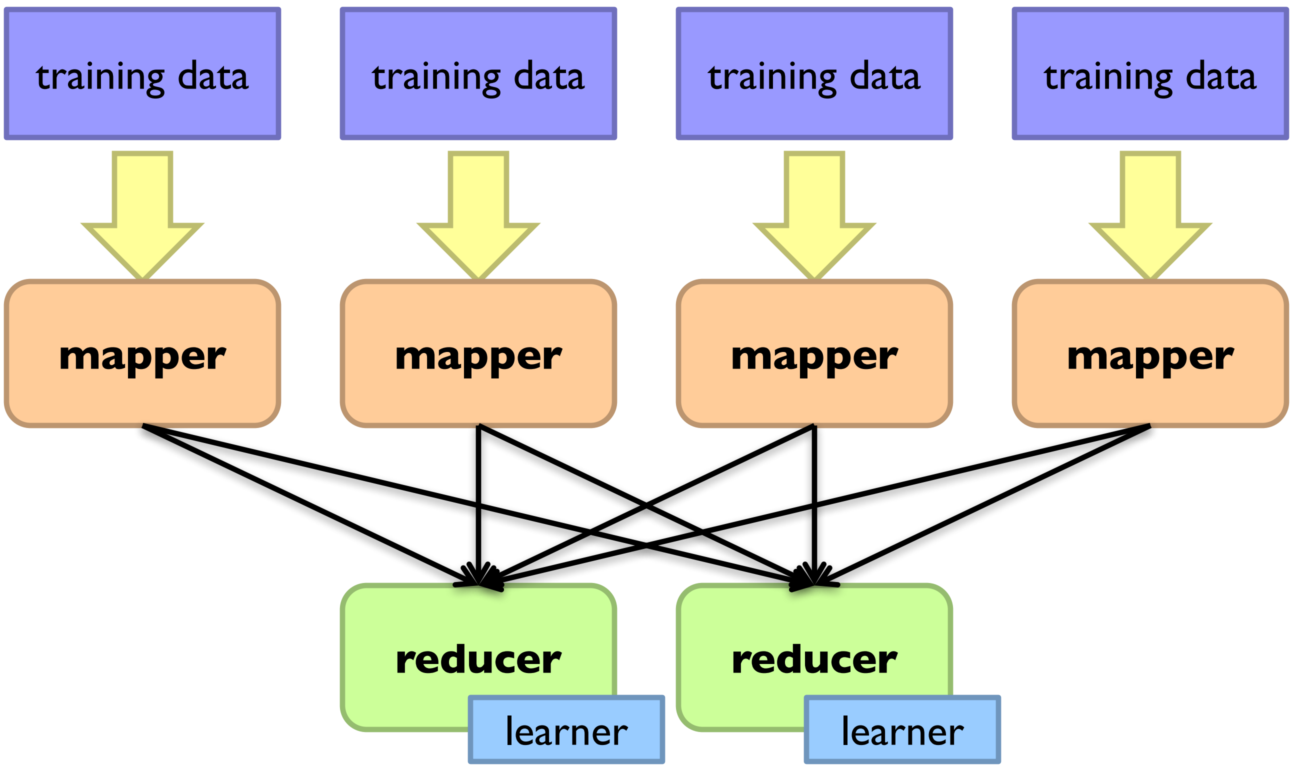
MapReduce Implementation

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



MapReduce Implementation

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



MapReduce Implementation

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

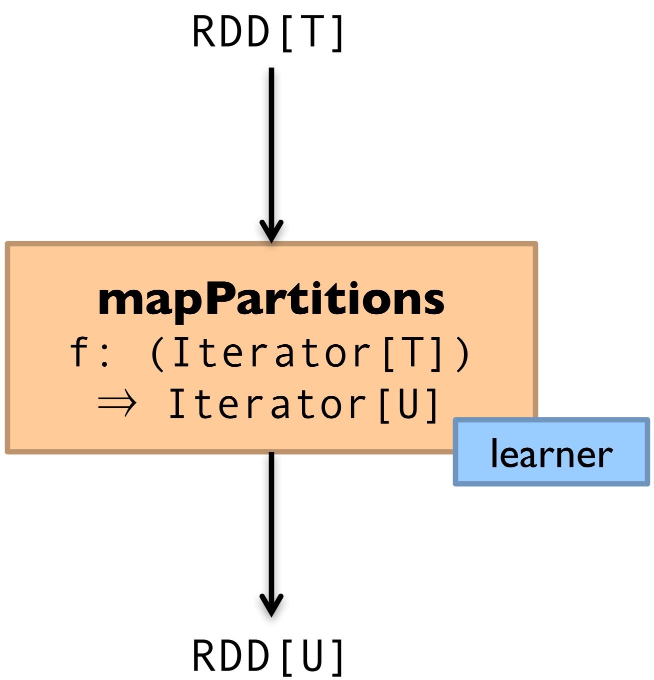
How do we output the model?

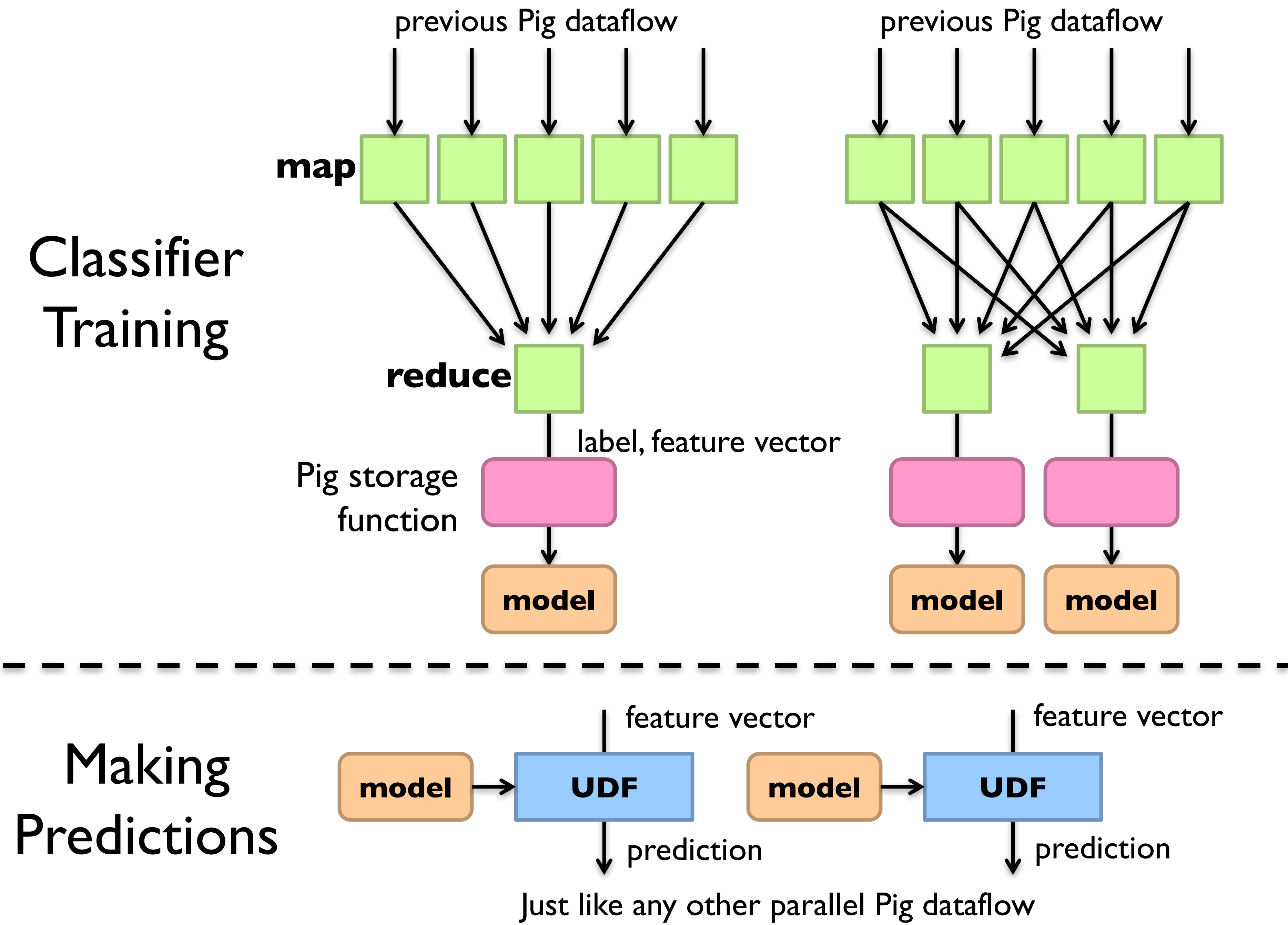
Option 1: write model out as "side data"

Option 2: emit model as intermediate output

What about Spark?

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




Classifier Training
previous Pig dataflow
previous Pig dataflow
map
reduce
label, feature vector
Pig storage function
model
model
model
Making Predictions
feature vector
feature vector
model
UDF
model
UDF
prediction
prediction
Just like any other parallel Pig dataflow

Classifier Training

```
training = load 'training.txt' using SVMLightStorage()
      as (target: int, features: map[]);

store training into 'model/'
      using FeaturesLRClassifierBuilder();
```

Logistic regression + SGD (L2 regularization)
Pegasos variant (fully SGD or sub-gradient)

Want an ensemble?

```
training = foreach training generate
      label, features, RANDOM() as random;
training = order training by random parallel 5;
```

Making Predictions

```
define Classify ClassifyWithLR('model/');
data = load 'test.txt' using SVMLightStorage()
       as (target: double, features: map[]);
data = foreach data generate target,
       Classify(features) as prediction;
```

Want an ensemble?

```
define Classify ClassifyWithEnsemble('model/',
'classifier.LR', 'vote');
```

Sentiment Analysis Case Study

Binary polarity classification: {positive, negative} sentiment

Use the "emoticon trick" to gather data

Data

Test: 500k positive/500k negative tweets from 9/1/2011

Training: {1m, 10m, 100m} instances from before (50/50 split)

Features:

Sliding window byte-4grams

Models + Optimization:

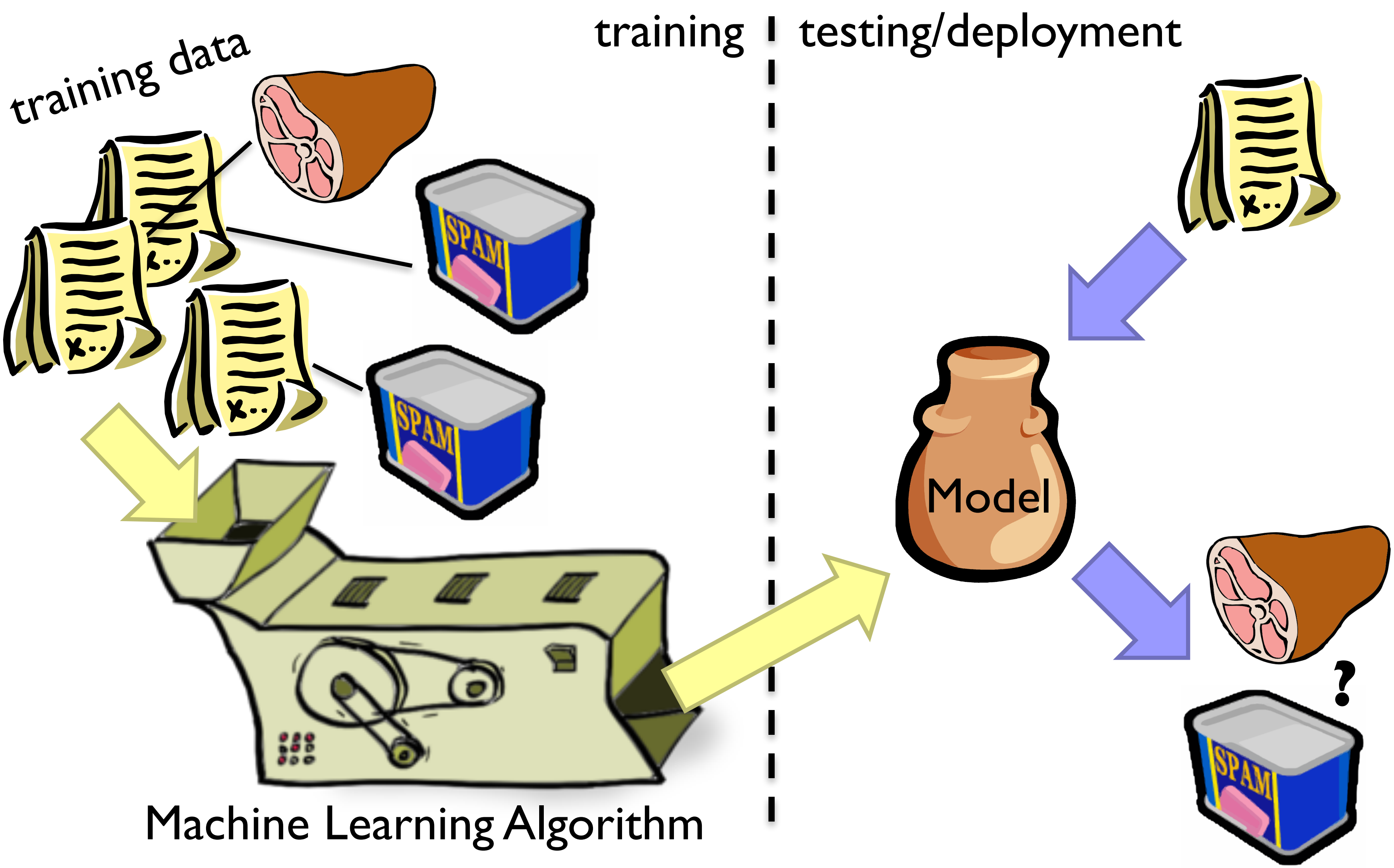
Logistic regression with SGD (L2 regularization)

Ensembles of various sizes (simple weighted voting)

Source: Lin and Kolcz. (2012) Large-Scale Machine Learning at Twitter. SIGMOD.


Diminishing returns…
Ensembles with 10m examples
better than 100m single classifier!
"for free"
1m instances
10m instances
100m instances
0.82
0.81
0.8
0.79
0.78
0.77
0.76
0.75
Accuracy
1 1 1 3 5 7 9 11 13 15 17 19 3 5 11 21 31 41
Number of Classifiers in Ensemble
single classifier
10m ensembles
100m ensembles

Supervised Machine Learning



Evaluation

How do we know how well we're doing?

Why isn't this enough?

Induce: $f : X \rightarrow Y$

Such that loss is minimized

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

We need end-to-end metrics!

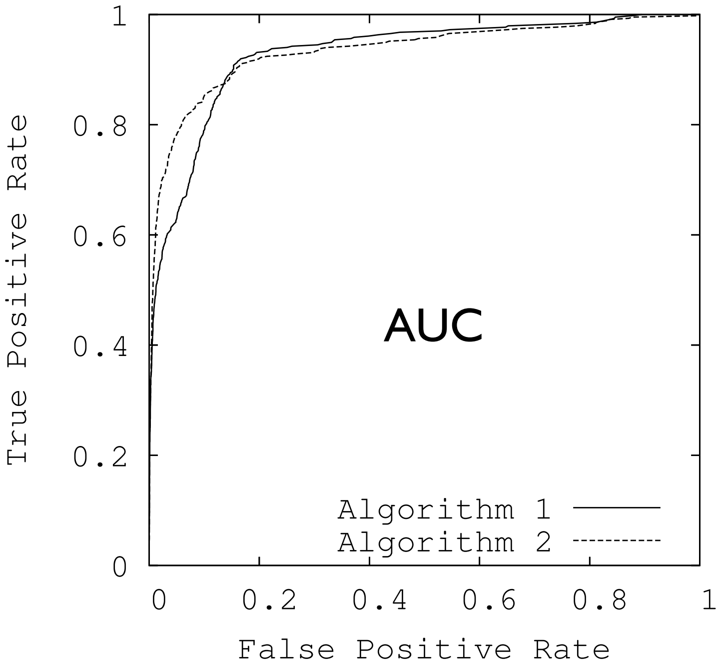
Obvious metric: accuracy

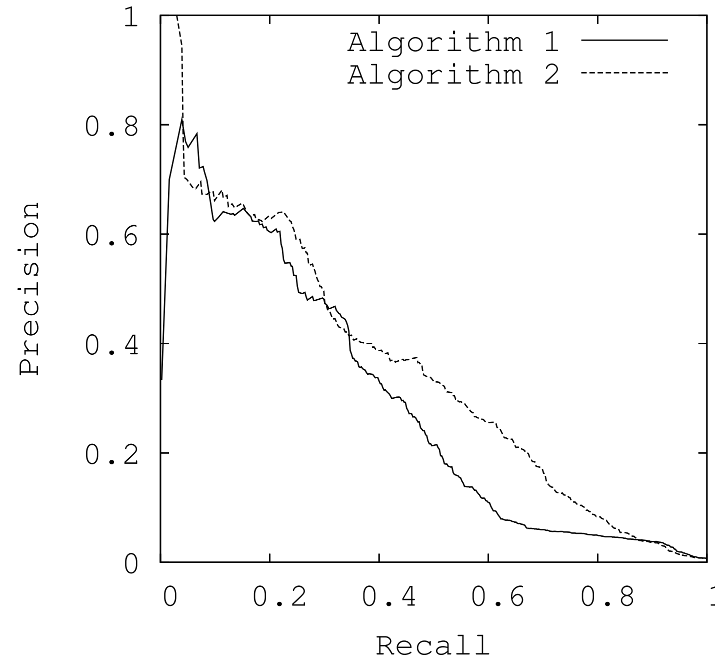
Why isn't this enough?

Metrics

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

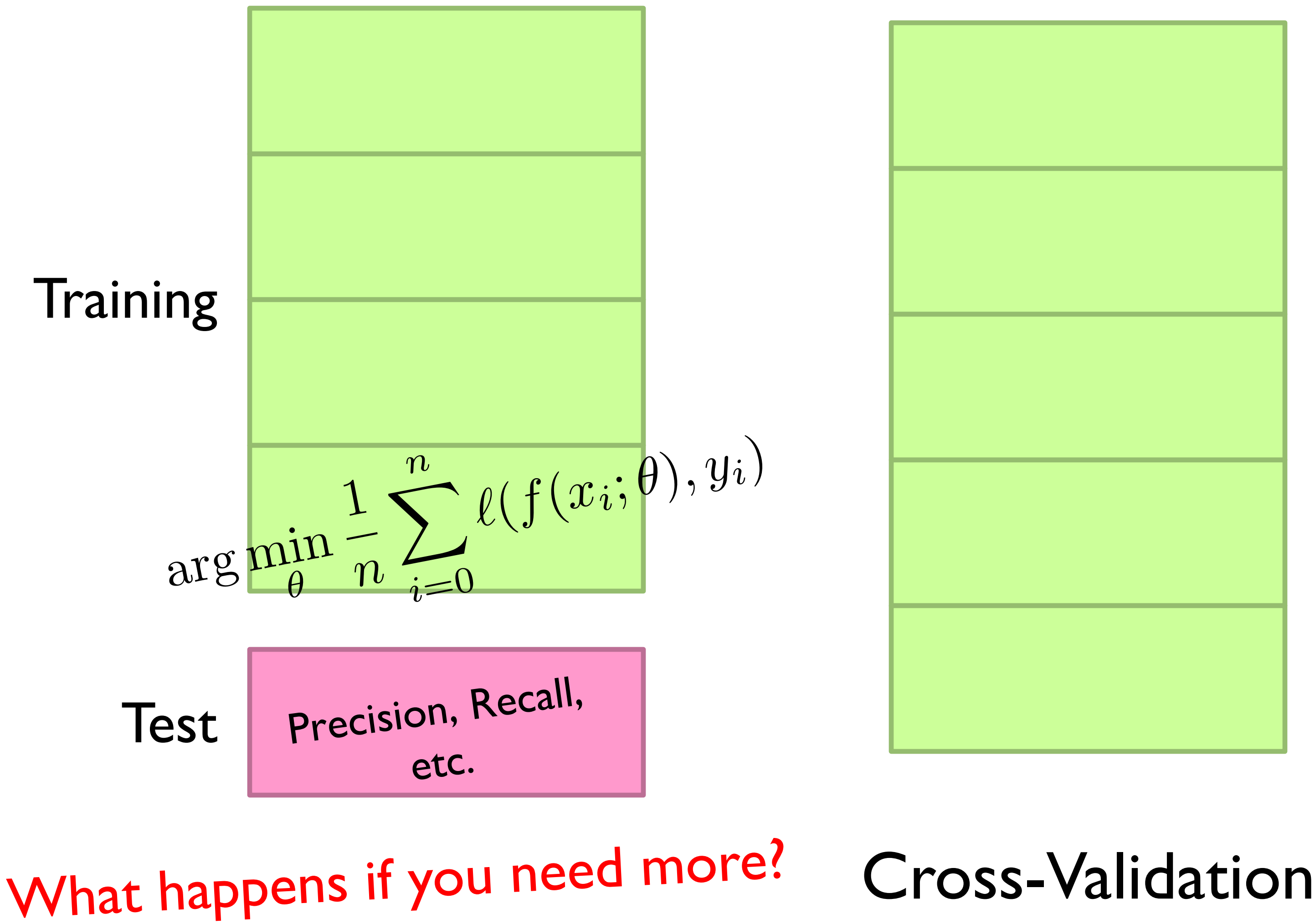
ROC and PR Curves



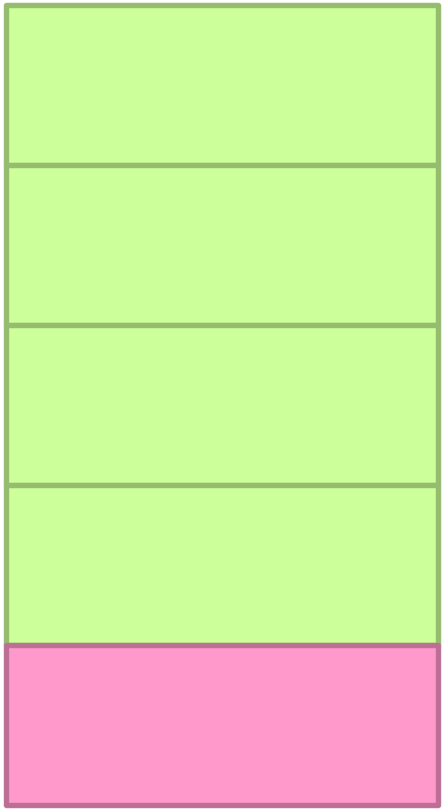


Source: Davis and Goadrich. (2006) The Relationship Between Precision-Recall and ROC curves

Training/Testing Splits

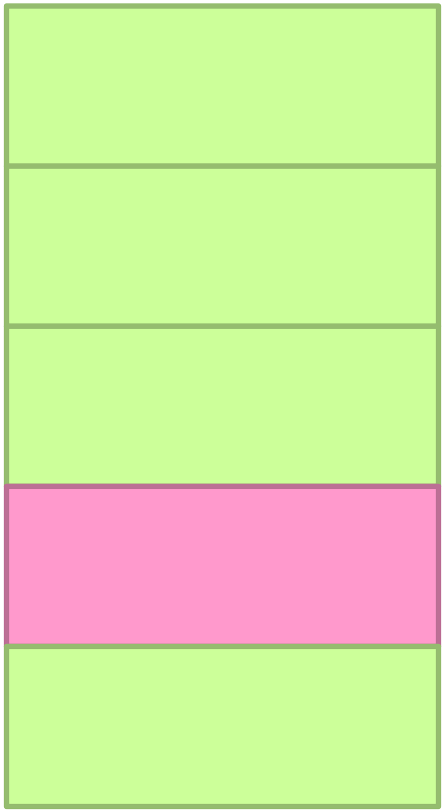


Training/Testing Splits



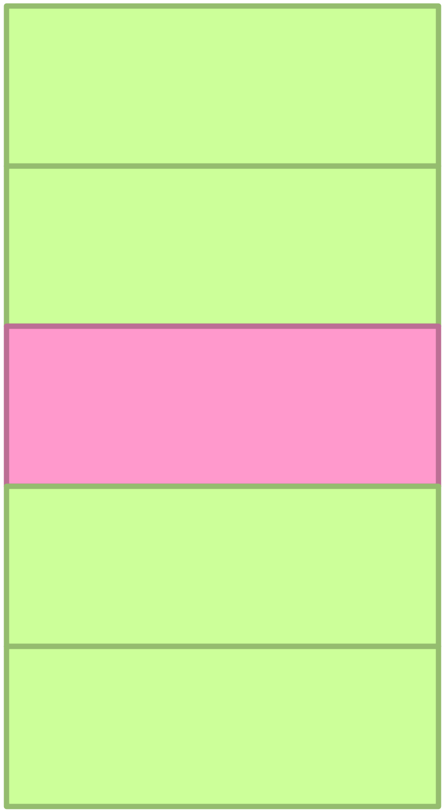
Cross-Validation

Training/Testing Splits



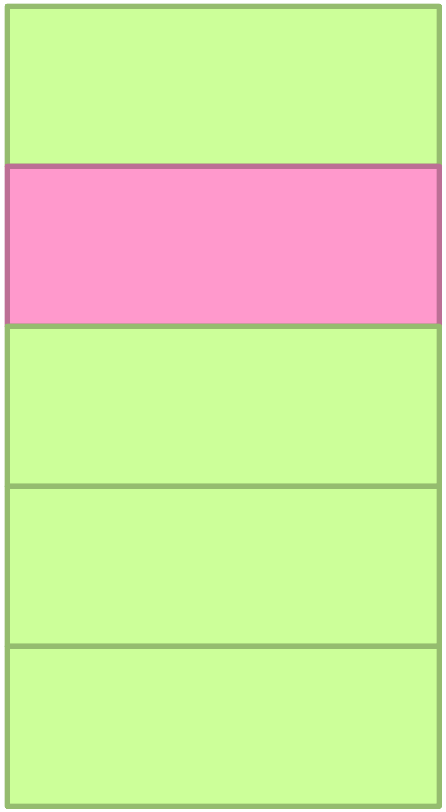
Cross-Validation

Training/Testing Splits



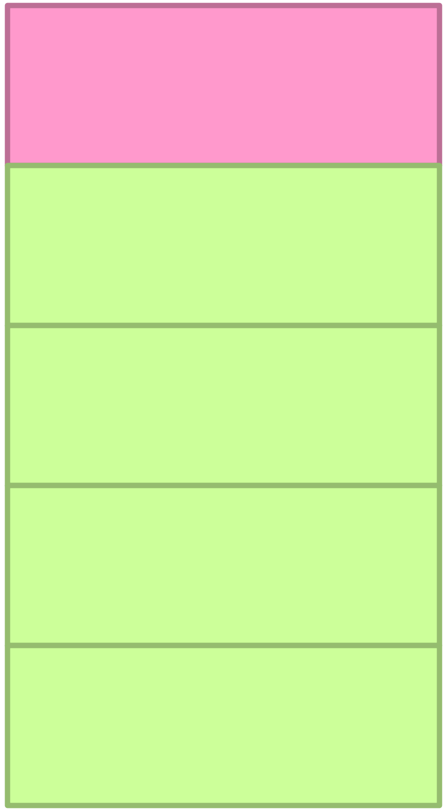
Cross-Validation

Training/Testing Splits



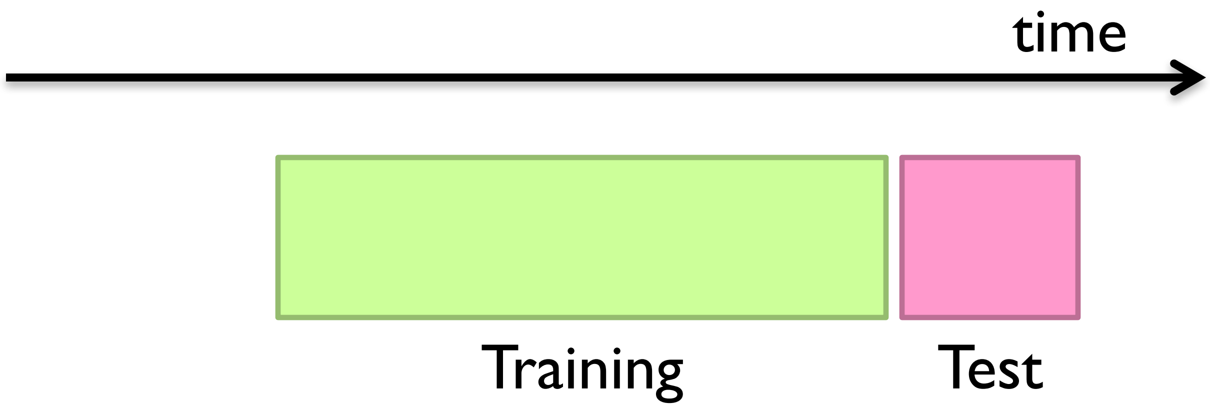
Cross-Validation

Training/Testing Splits



Cross-Validation

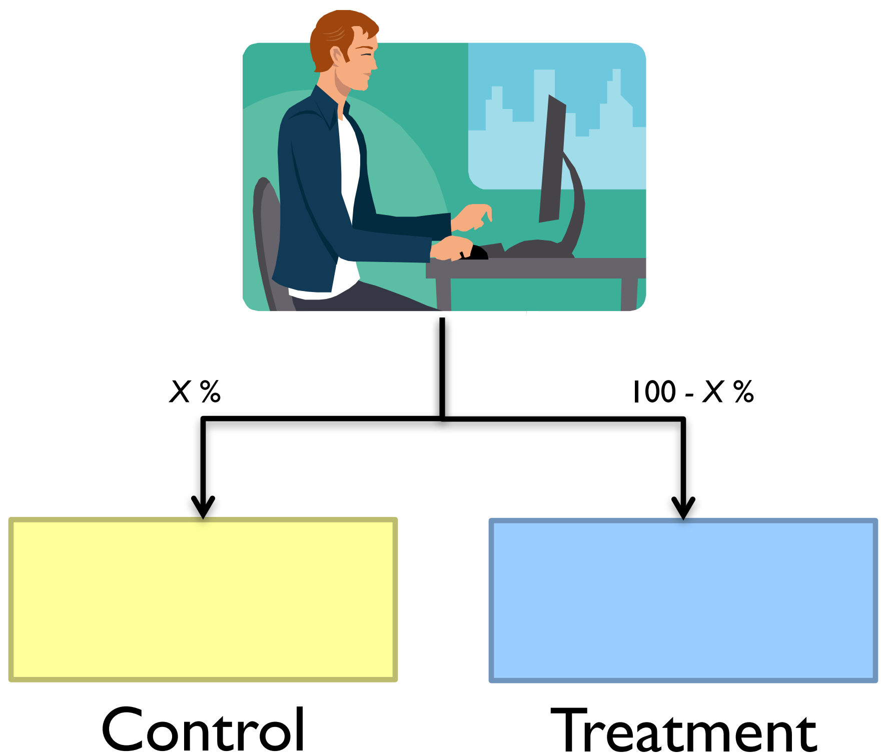
Typical Industry Setup



A/B test

Why not cross-validation?

A/B Testing



Gather metrics, compare alternatives

A/B Testing: Complexities

Properly bucketing users

Novelty

Learning effects

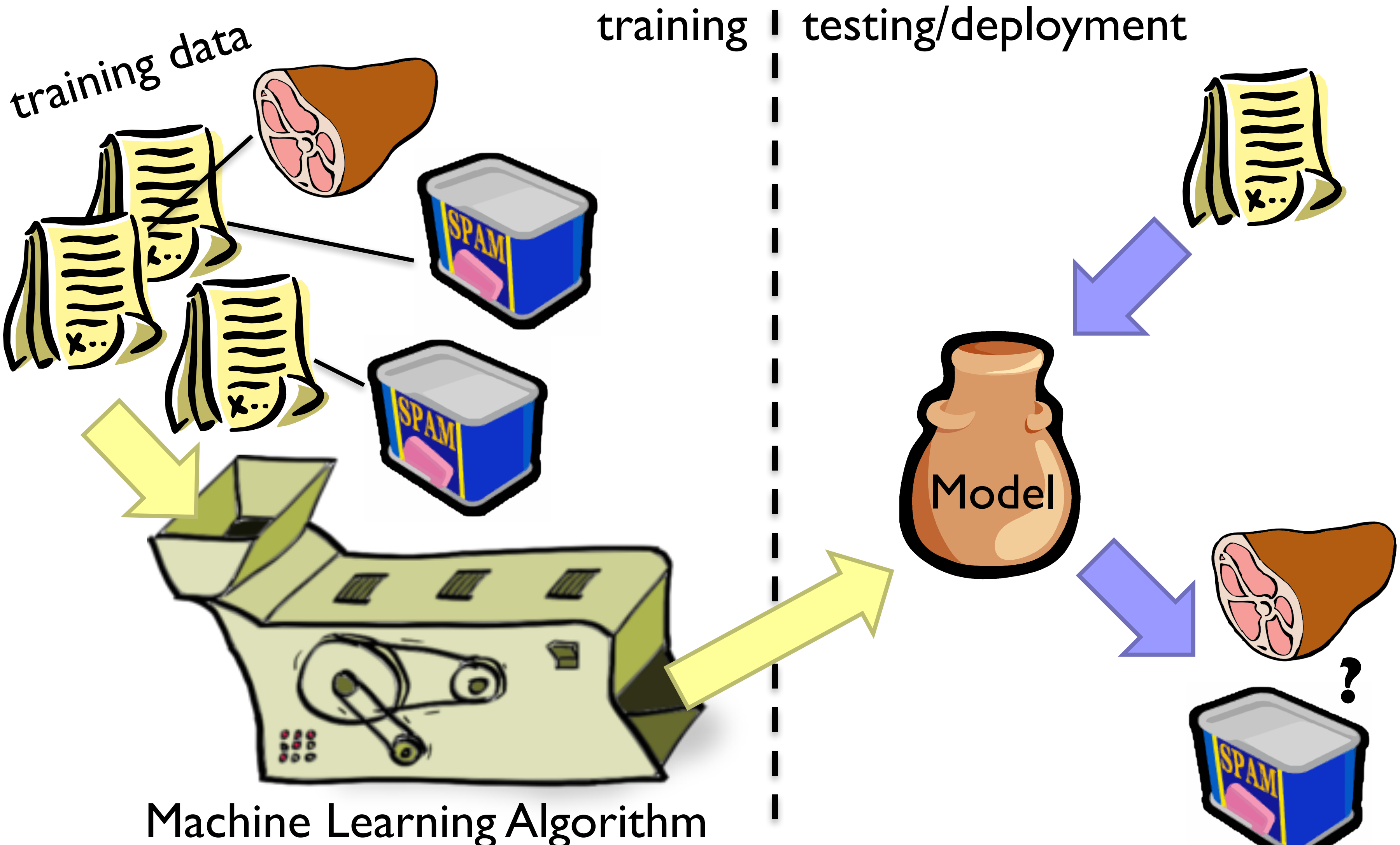
Long vs. short term effects

Multiple, interacting tests

Nosy tech journalists

…

Supervised Machine Learning



Applied ML in Academia

Download interesting dataset (comes with the problem)

Run baseline model

Train/Test

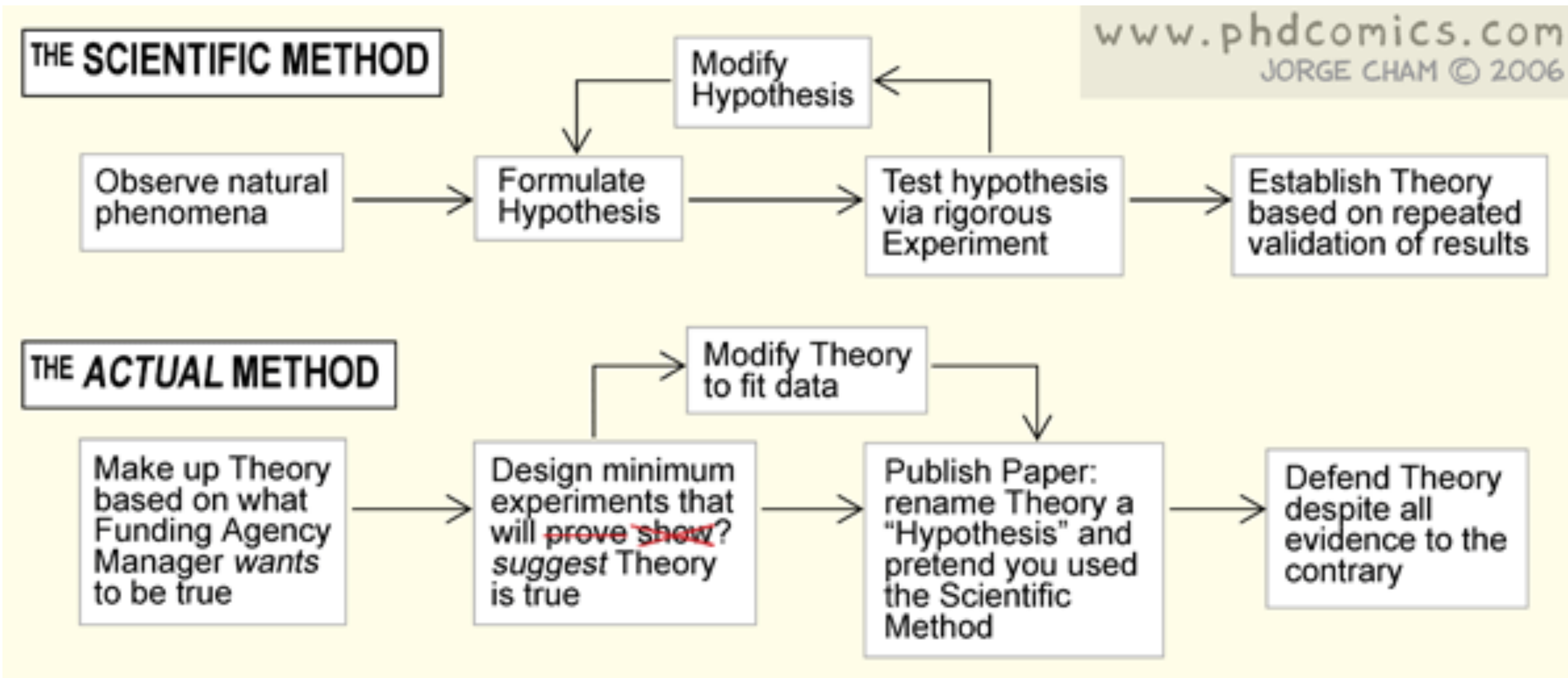
Build better model

Train/Test

Does new model beat baseline?

Yes: publish a paper!

No: try again!


www.phdcomics.com
JORGE CHAM © 2006
THE SCIENTIFIC METHOD
Observe natural phenomena
Formulate Hypothesis
Modify Hypothesis
Test hypothesis via rigorous Experiment
Establish Theory based on repeated validation of results
THE *ACTUAL* METHOD
Make up Theory based on what Funding Agency Manager *wants* to be true
Design minimum experiments that will ~~prove show~~? *suggest* Theory is true
Modify Theory to fit data
Publish Paper: rename Theory a "Hypothesis" and pretend you used the Scientific Method
Defend Theory despite all evidence to the contrary

Harvard Business Review

DATA

Data Scientist: The Sexiest Job of the 21st Century

by Thomas H. Davenport and D.J. Patil

FROM THE OCTOBER 2012 ISSUE

Fantasy

Extract features

Develop cool ML technique

#Profit

Reality

What's the task?

Where's the data?

What's in this dataset?

What's all the f#$!* crap?

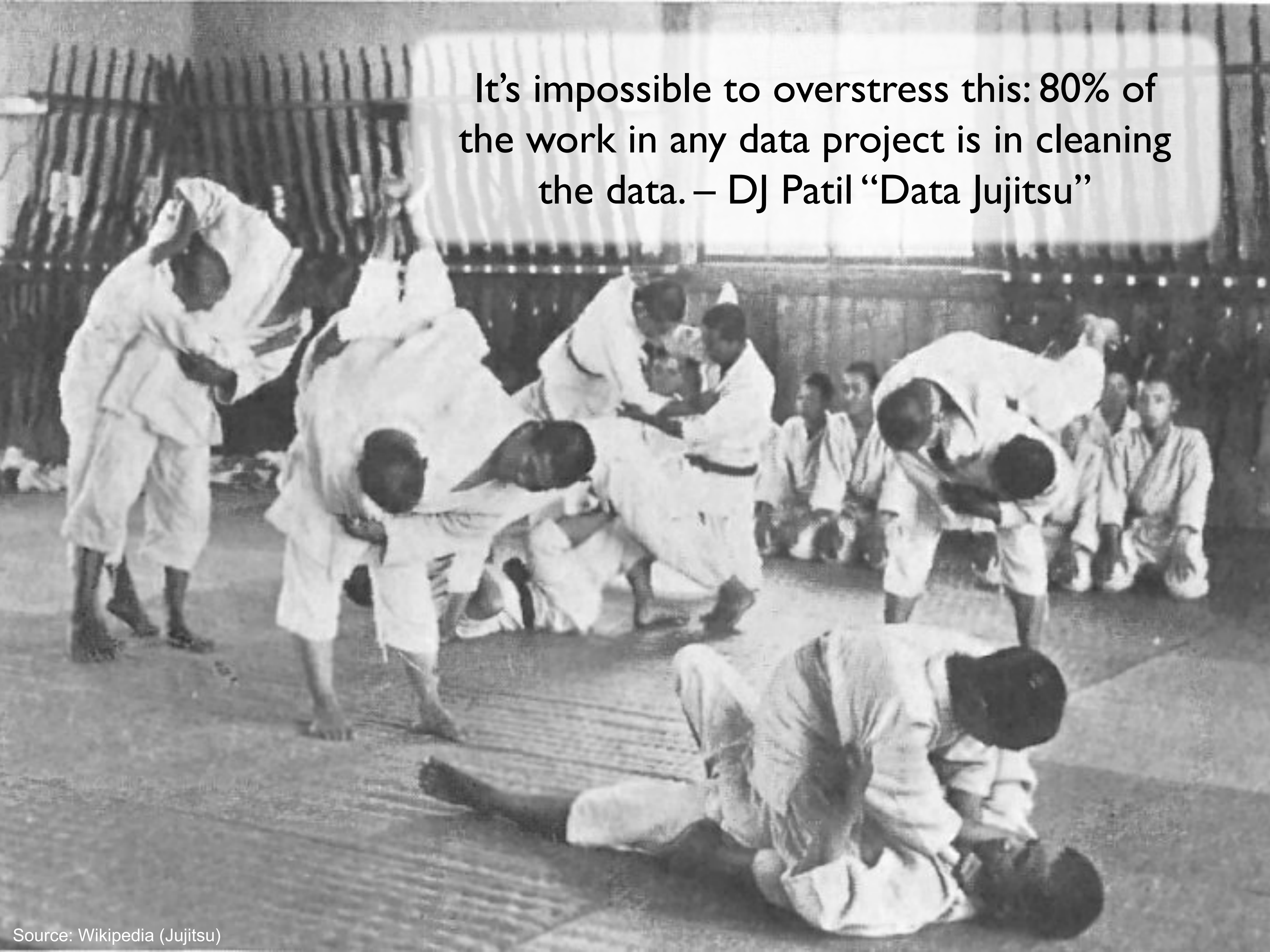
Clean the data

Extract features

"Do" machine learning

Fail, iterate…

Dirty secret: very little of data science is about machine learning per se!


It's impossible to overstress this: 80% of the work in any data project is in cleaning the data. – DJ Patil "Data Jujitsu"
Source: Wikipedia (Jujitsu)

SUBSCRIBE NOW LOG IN

The New York Times

SECTIONS HOME SEARCH

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

On finding things…

On naming things…

uid

UserId

userId

userid

user_id

user_Id

CamelCase

smallCamelCase

snake_case

camel_Snake

dunder__snake

On feature extraction…

```
^(\\w+\\s+\\d+\\s+\\d+:\\d+:\\d+)\\s+
([^@]+?)@(\\S+)\\s+(\\S+):\\s+(\\S+)\\s+(\\S+)
\\s+((?:\\S+?,\\s+)*(?:\\S+?))\\s+(\\S+)\\s+(\\S+)
\\s+\\[([^\\]]+)\\]\\s+\"(\\w+)\\s+([^\"\\\\]*
(?:\\\\.[^\"\\\\]*)*)\\s+(\\S+)\"\\s+(\\S+)\\s+
(\\S+)\\s+\"([^\"\\\\]*(?:\\\\.[^\"\\\\]*)*)
\"\\s+\"([^\"\\\\]*(?:\\\\.[^\"\\\\]*)*)\"\\s*
(\\d*-[\\d-]*)?\\s*(\\d+)?\\s*(\\d*\\.[\\d\\.]*)?
(\\s+[-\\w]+)?.*$
```

An actual Java regular expression used to parse log message at Twitter circa 2010

Friction is cumulative!

Data Plumbing… Gone Wrong!

[scene: consumer internet company in the Bay Area…]

Okay, let's get going… where's the click data?

It's over here…

Well, that's kinda non-intuitive, but okay…

Well, it wouldn't fit, so we had to shoehorn…

Oh, BTW, where's the timestamp of the click?

Hang on, I don't remember…

Uh, bad news. Looks like we forgot to log it…

[grumble, grumble, grumble]

Frontend Engineer

Develops new feature, adds logging code to capture clicks

Data Scientist

Analyze user behavior, extract insights to improve feature

Fantasy

Extract features

Develop cool ML technique

#Profit

Reality

What's the task?

Where's the data?

What's in this dataset?

What's all the f#$!* crap?

Clean the data

Extract features

"Do" machine learning

Fail, iterate…

Finally works!

Congratulations, you’re halfway there…

Source: Wikipedia (Hills)

Congratulations, you’re halfway there…

Does it actually work?
A/B testing

Is it fast enough?

Good, you’re two thirds there…

Productionize

Source: Wikipedia (Oil refinery)

Productionize

What are your jobs' dependencies?

How/when are your jobs scheduled?

Are there enough resources?

How do you know if it's working?

Who do you call if it stops working?

Infrastructure is critical here!
(plumbing)

Takeaway lessons:

Most of data science isn't glamorous!

Source: Wikipedia (Plumbing)

Source: Wikipedia (Japanese rock garden)