



UNIVERSITY OF
WATERLOO

Data-Intensive Distributed Computing

CS 451/651 431/631 (Winter 2018)

Part 6: Data Mining (2/4)

March 1, 2018

Jimmy Lin

David R. Cheriton School of Computer Science

University of Waterloo

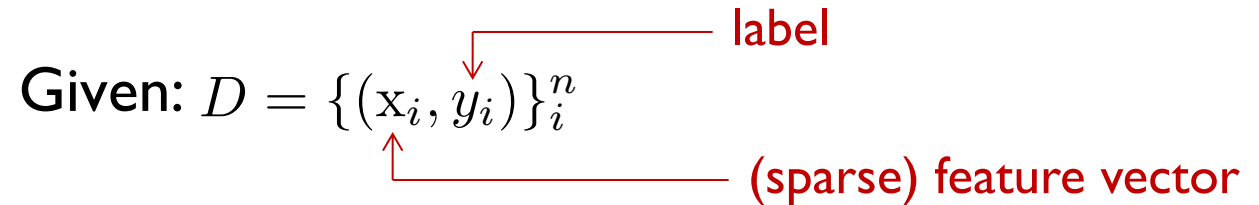
These slides are available at <http://lintool.github.io/bigdata-2018w/>



This work is licensed under a Creative Commons Attribution-NonCommercial-Share Alike 3.0 United States
See <http://creativecommons.org/licenses/by-nc-sa/3.0/us/> for details

The Task

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



label

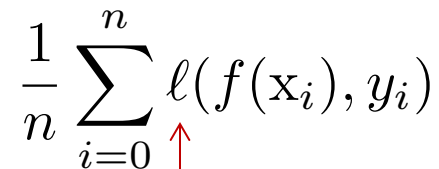
(sparse) feature vector

$$x_i = [x_1, x_2, x_3, \dots, x_d]$$

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

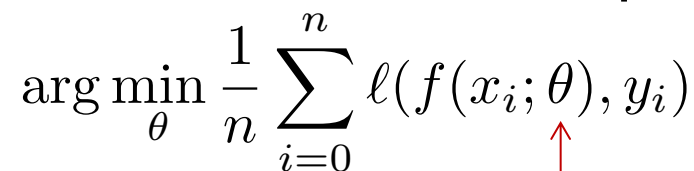
Induce: $f : X \rightarrow Y$

Such that loss is minimized

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


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

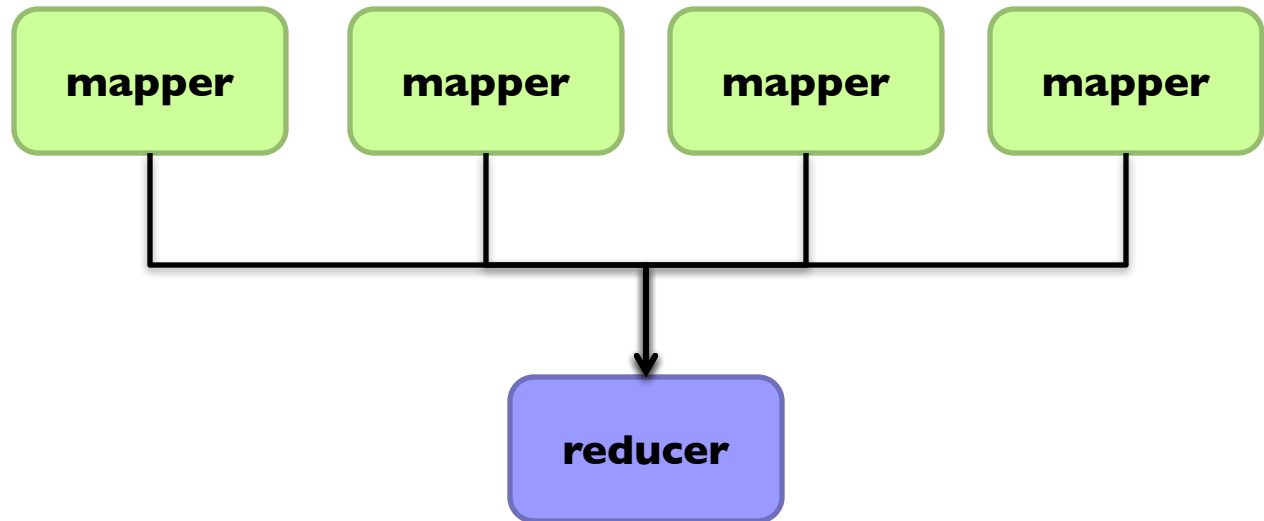
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \frac{1}{n} \sum_{i=0}^n \nabla \ell(f(x_i; \theta^{(t)}), y_i)$$

MapReduce Implementation

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

single reducer

compute partial gradient



iterate until convergence

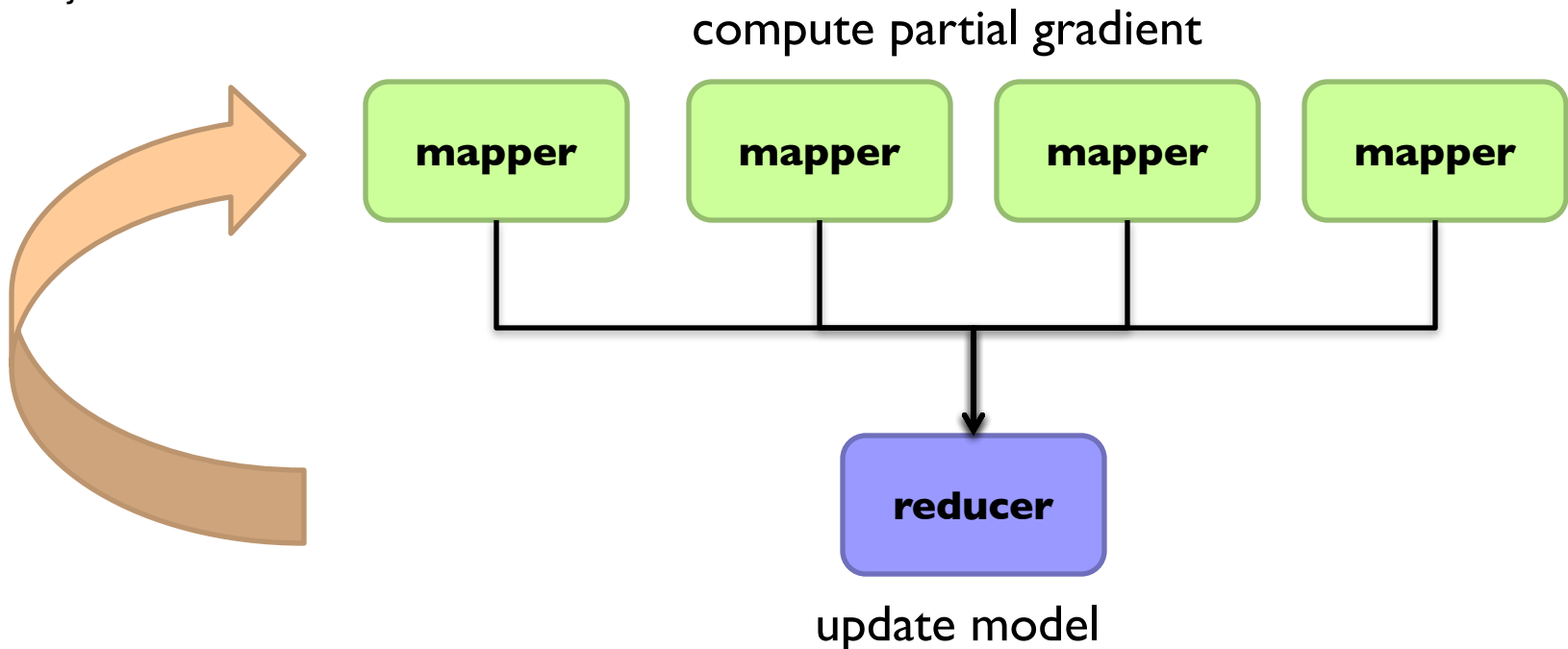
update model

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?



A landscape of rolling green hills under a blue sky with white clouds. The hills are covered in vibrant green grass and some patches of brown soil, suggesting a rural or agricultural setting. The sky is filled with large, fluffy white clouds, and the overall scene is bright and clear.

Gradient Descent



Stochastic Gradient Descent

Batch vs. Online

Gradient Descent

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

“batch” learning: update model after considering all training instances

Stochastic Gradient Descent (SGD)

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

“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!

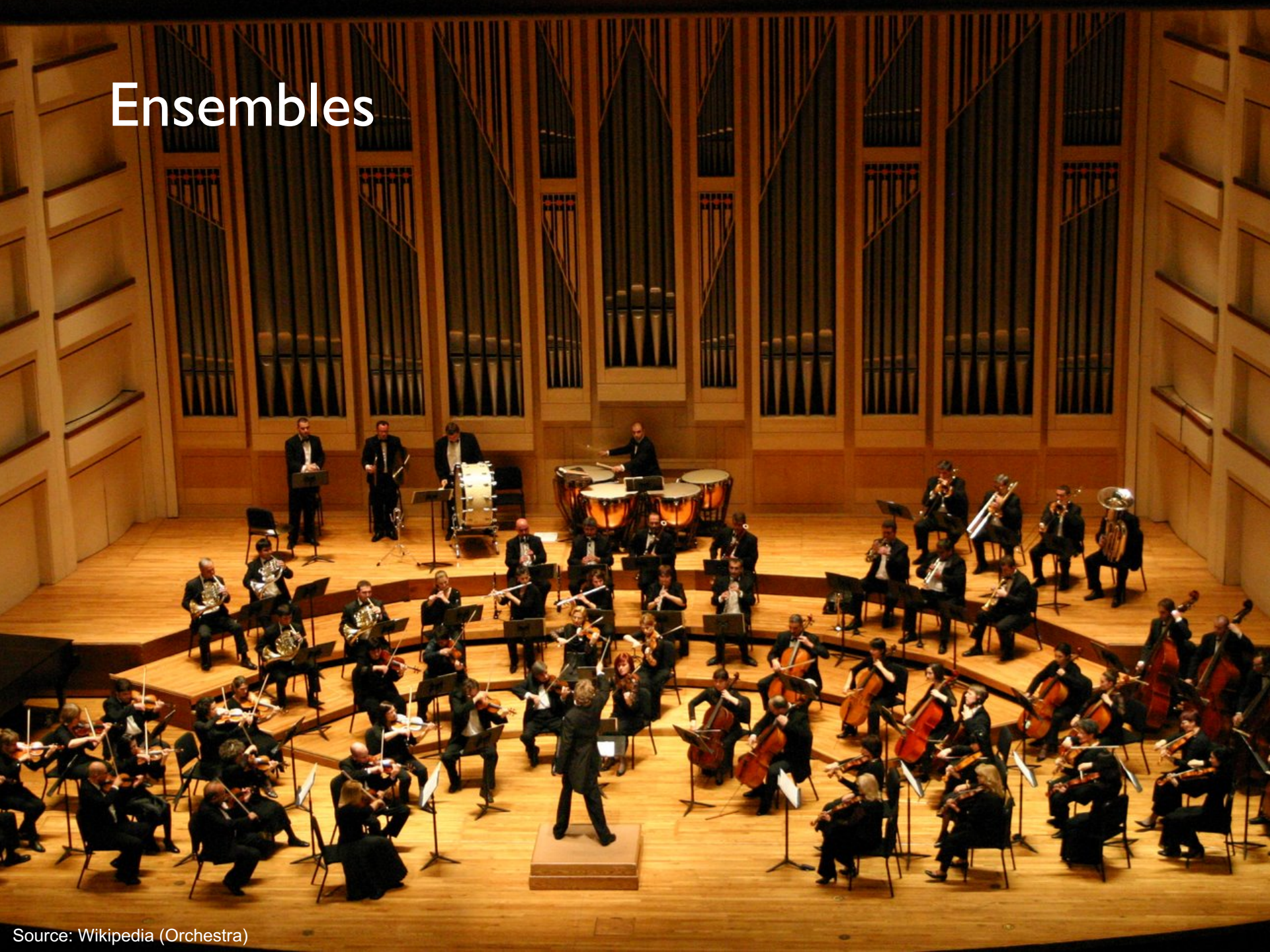
Most common implementation: randomly shuffle training instances

Single vs. multi-pass approaches

Mini-batching as a middle ground

We've solved the iteration problem!
What about the single reducer problem?

Ensembles



Ensemble Learning

Learn multiple *independent* 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:

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

Model averaging

...

Ensemble Learning

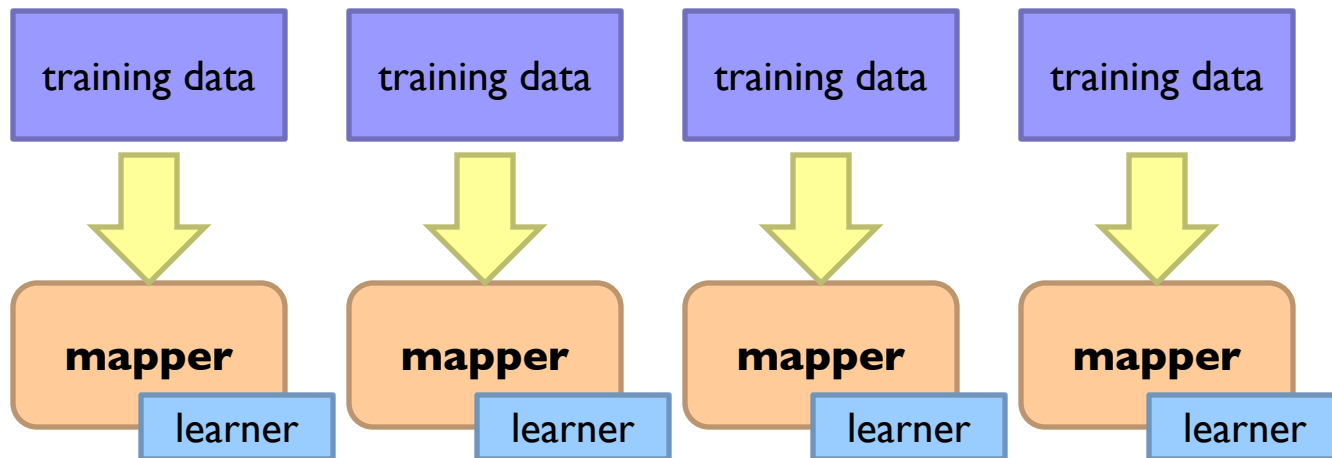
Learn multiple *independent* models, combine results from different models to make prediction

Why does it work?

If errors uncorrelated, multiple classifiers being wrong is less likely
Reduces the variance component of error

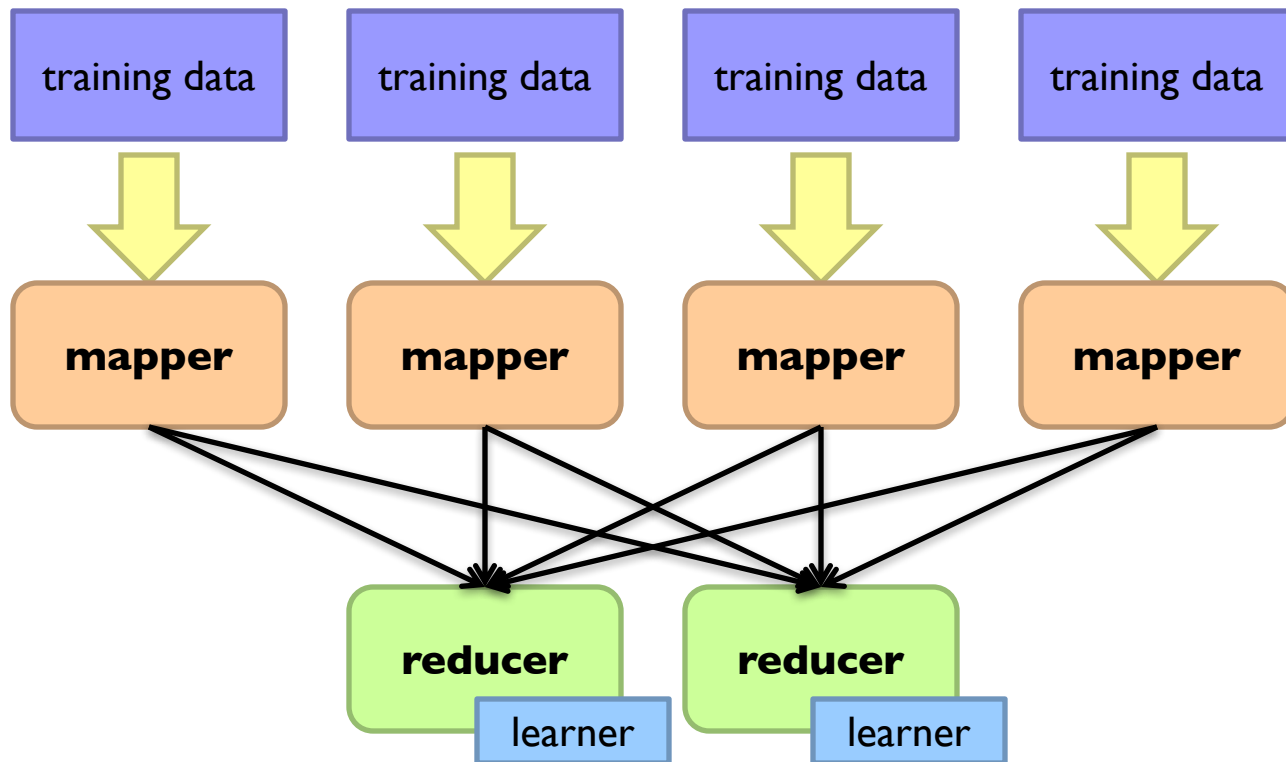
MapReduce Implementation

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



MapReduce Implementation

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



MapReduce Implementation

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

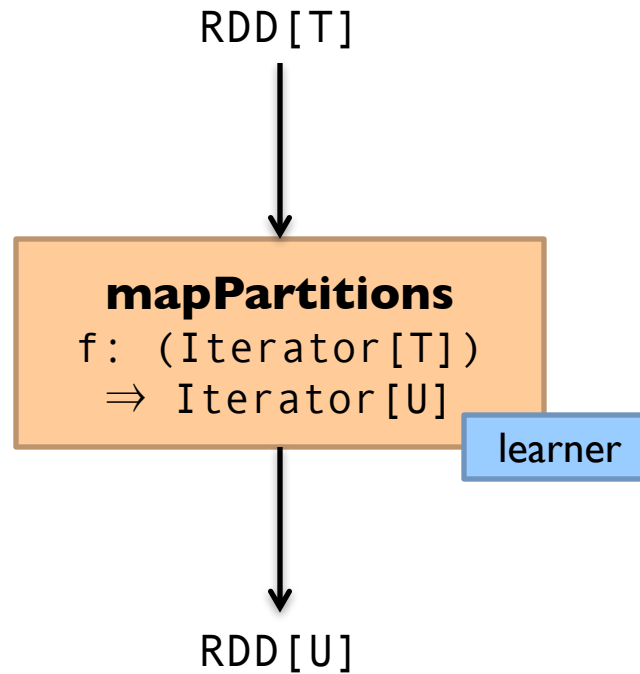
How do we output the model?

Option 1: write model out as “side data”

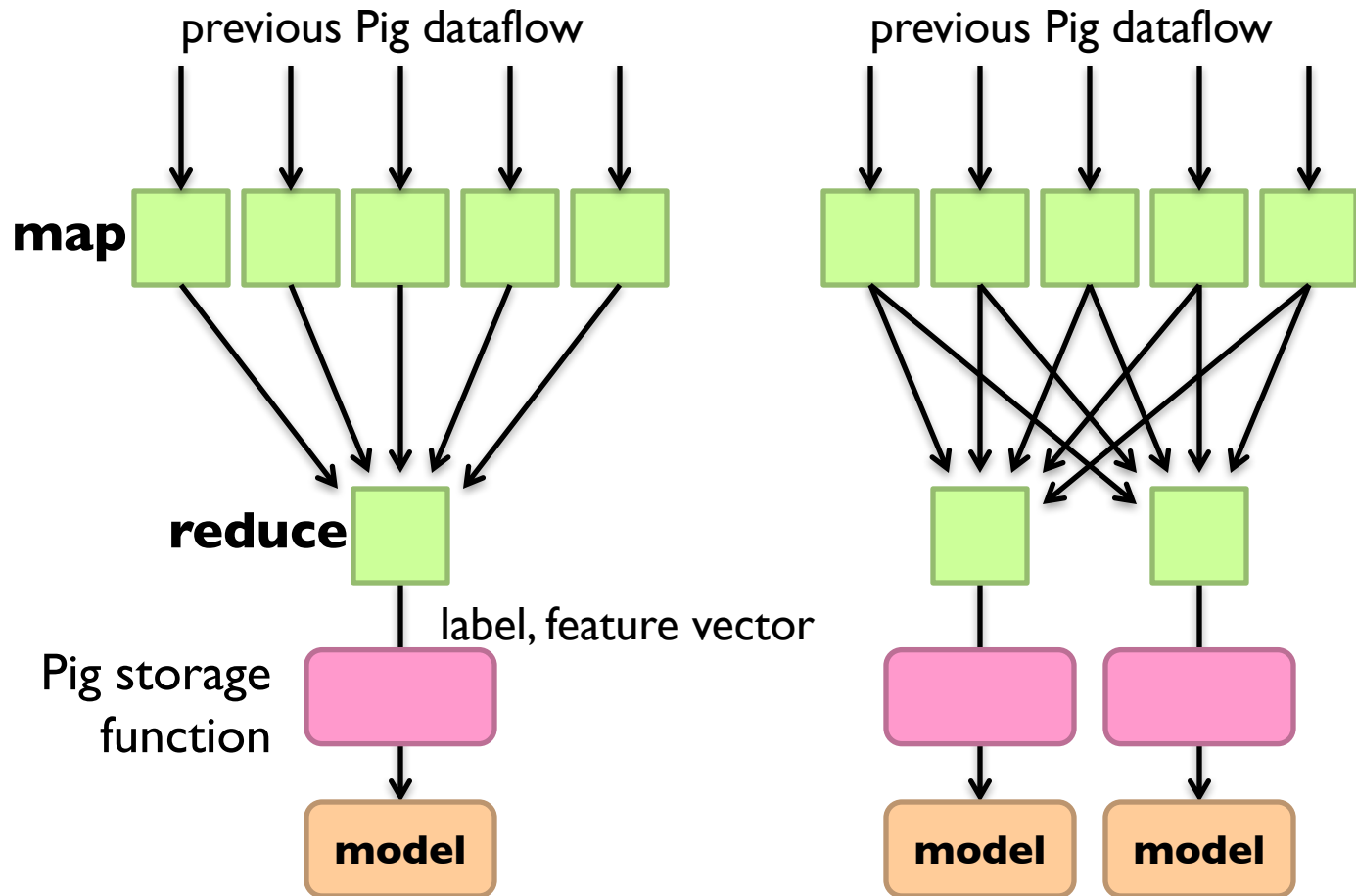
Option 2: emit model as intermediate output

What about Spark?

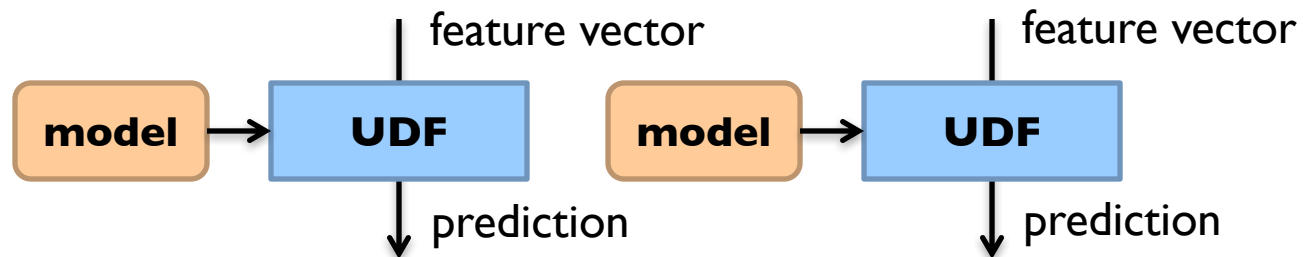
$$\theta^{(t+1)} \leftarrow \theta^{(t)} - \gamma^{(t)} \nabla \ell(f(\mathbf{x}; \theta^{(t)}), y)$$



Classifier Training



Making Predictions



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)

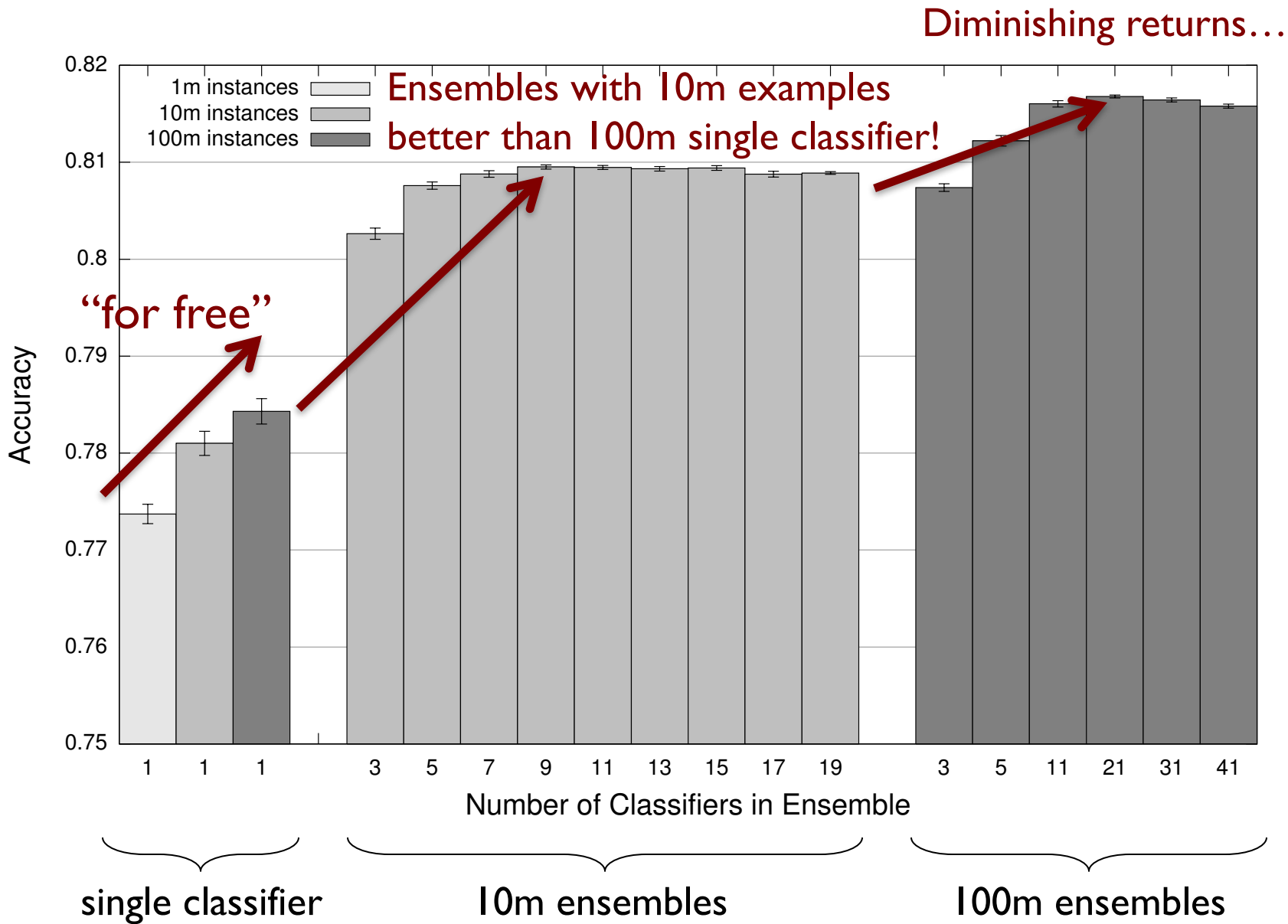
Features:

Sliding window byte-4grams

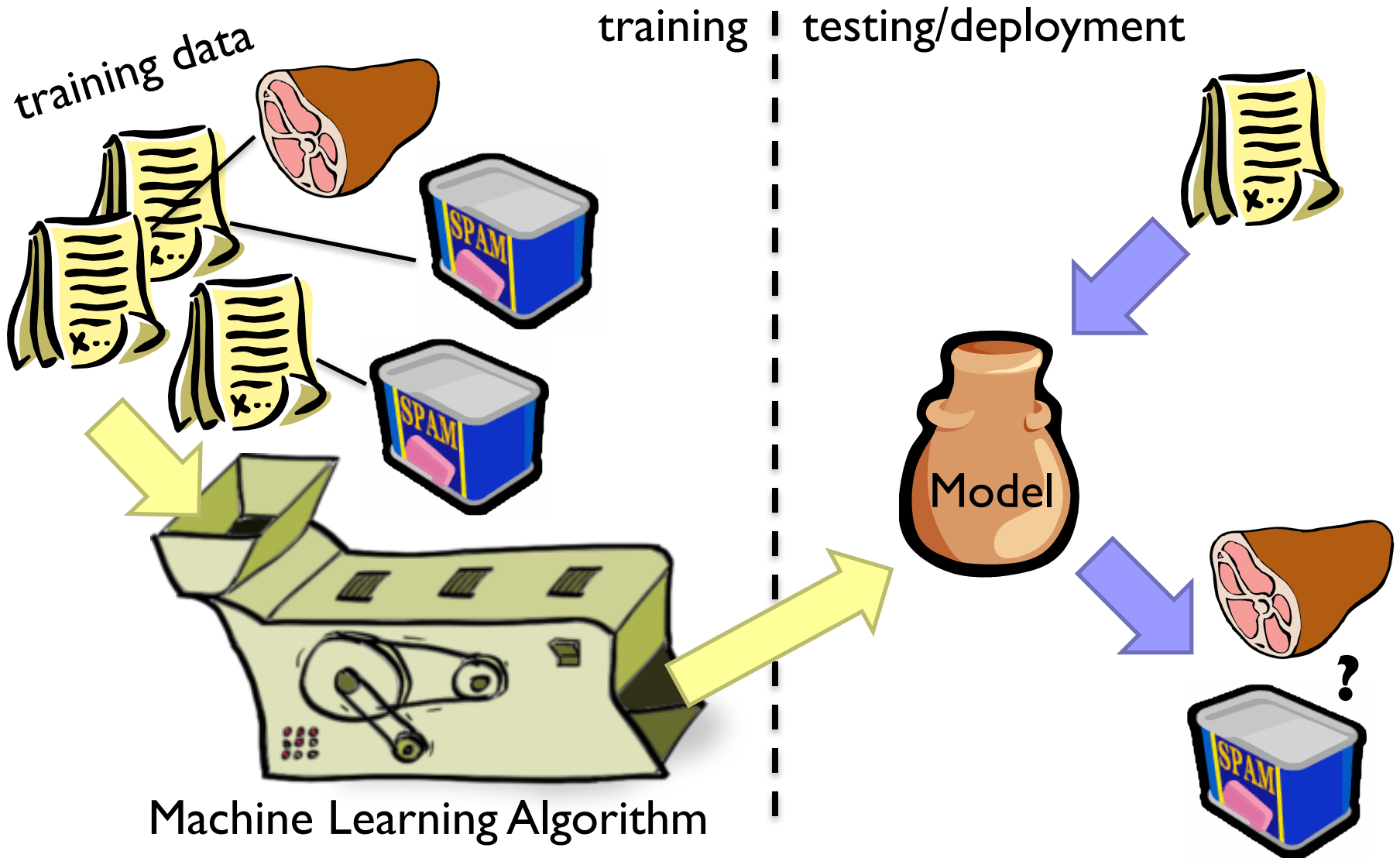
Models + Optimization:

Logistic regression with SGD (L2 regularization)

Ensembles of various sizes (simple weighted voting)



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!

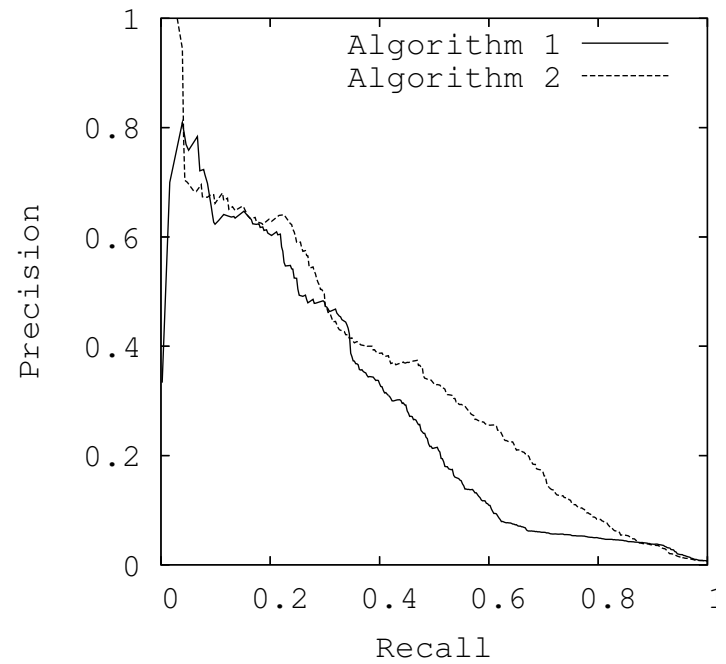
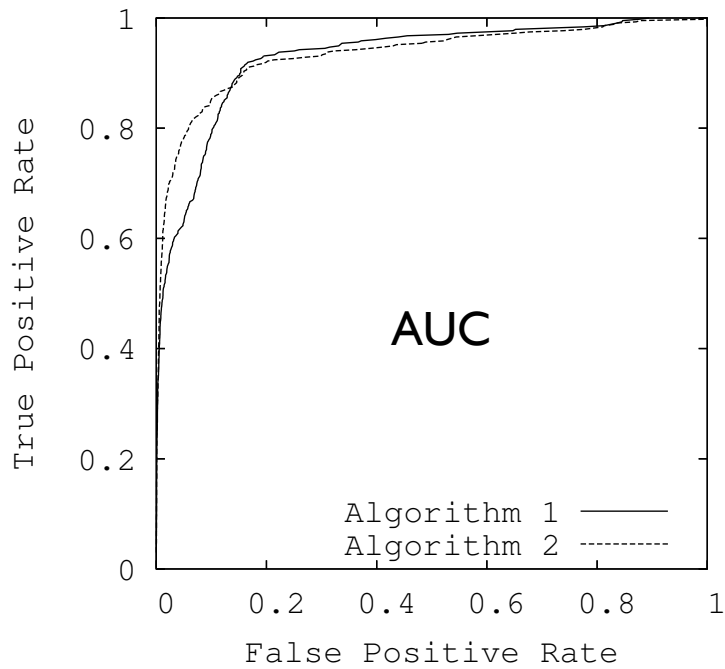
Obvious metric: accuracy

Why isn't this enough?

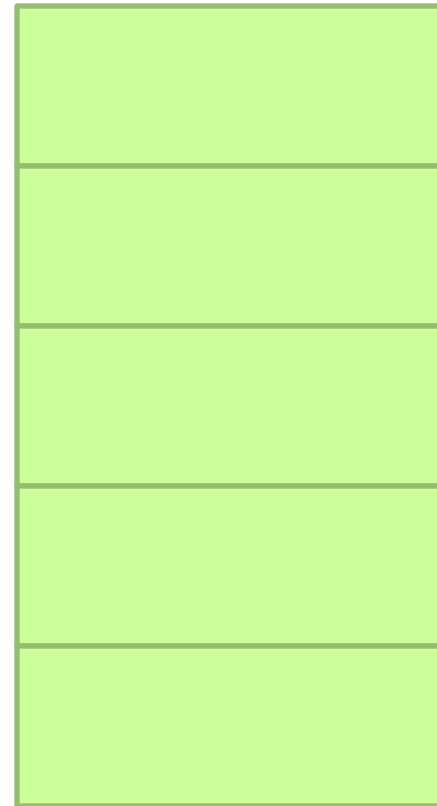
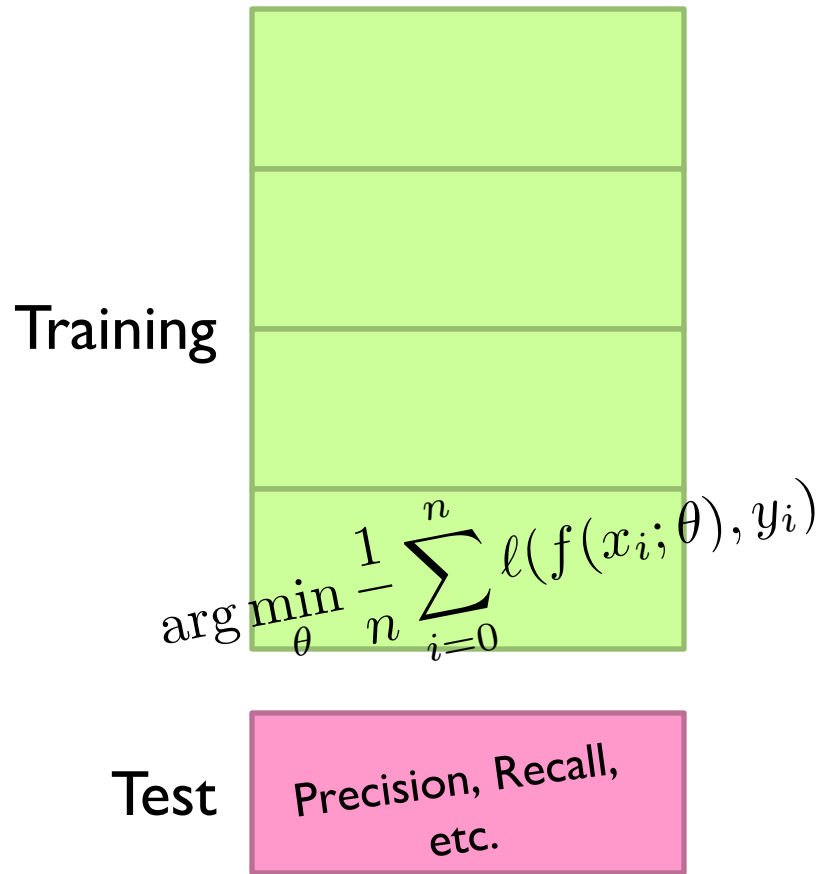
Metrics

		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



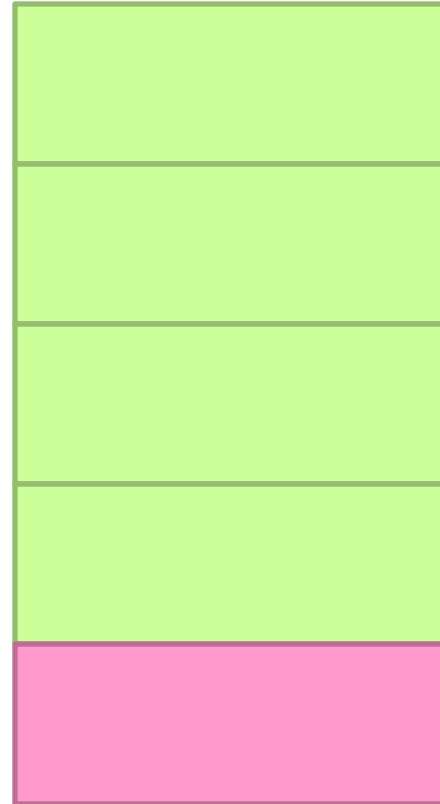
Training/Testing Splits



What happens if you need more?

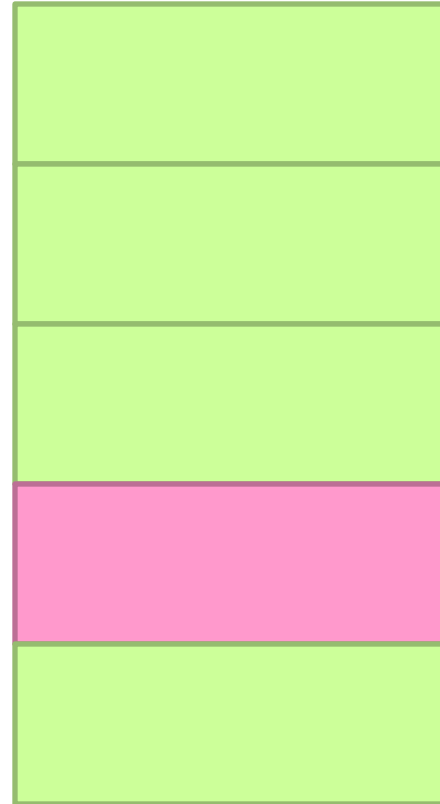
Cross-Validation

Training/Testing Splits



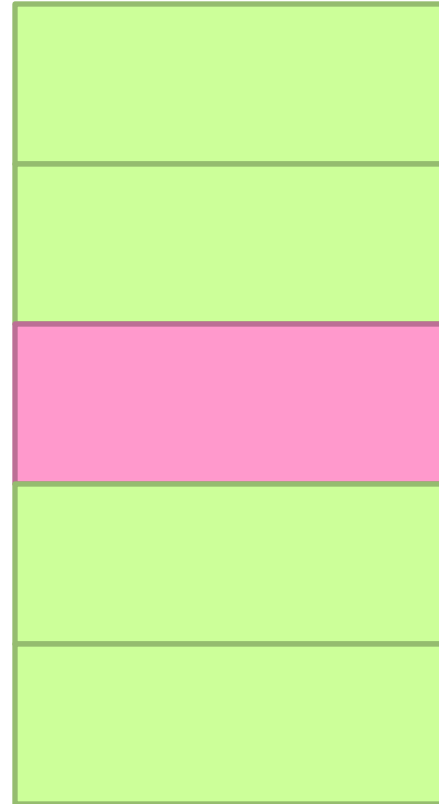
Cross-Validation

Training/Testing Splits



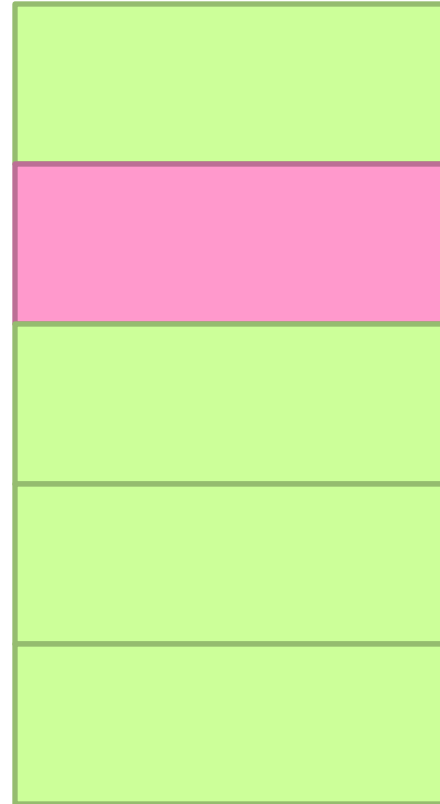
Cross-Validation

Training/Testing Splits



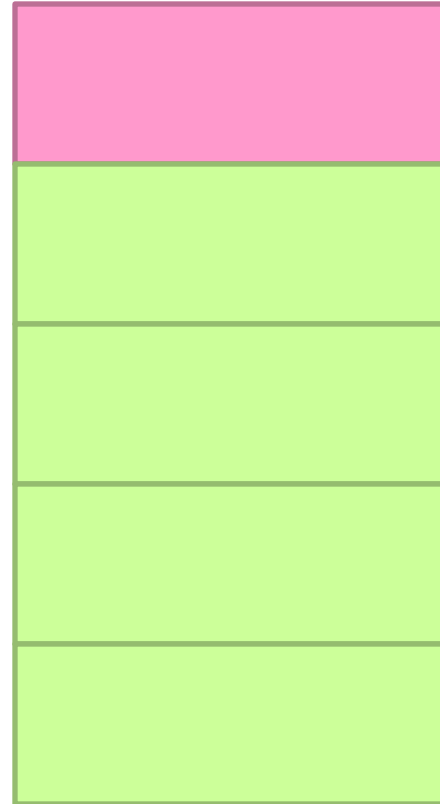
Cross-Validation

Training/Testing Splits



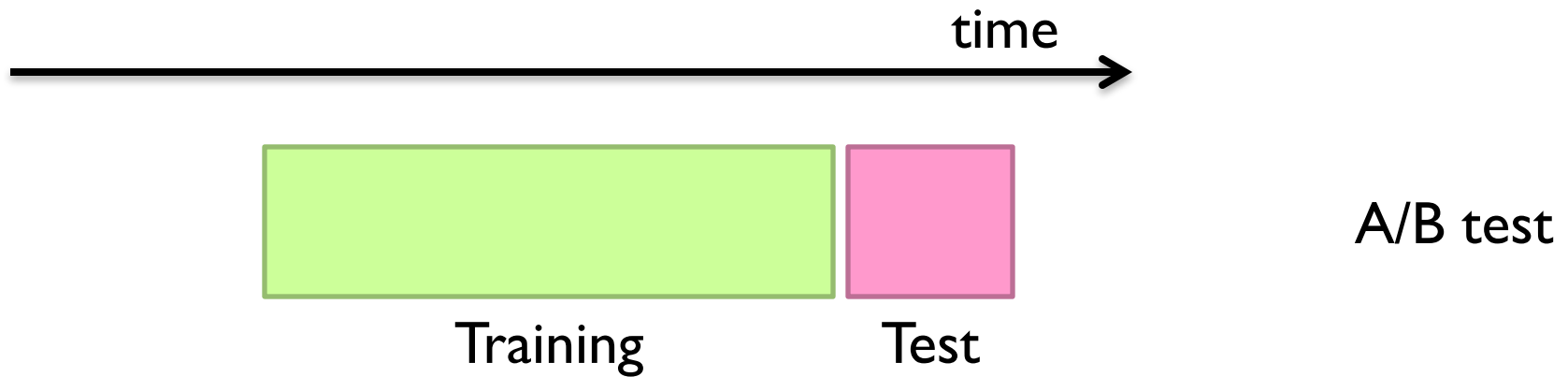
Cross-Validation

Training/Testing Splits



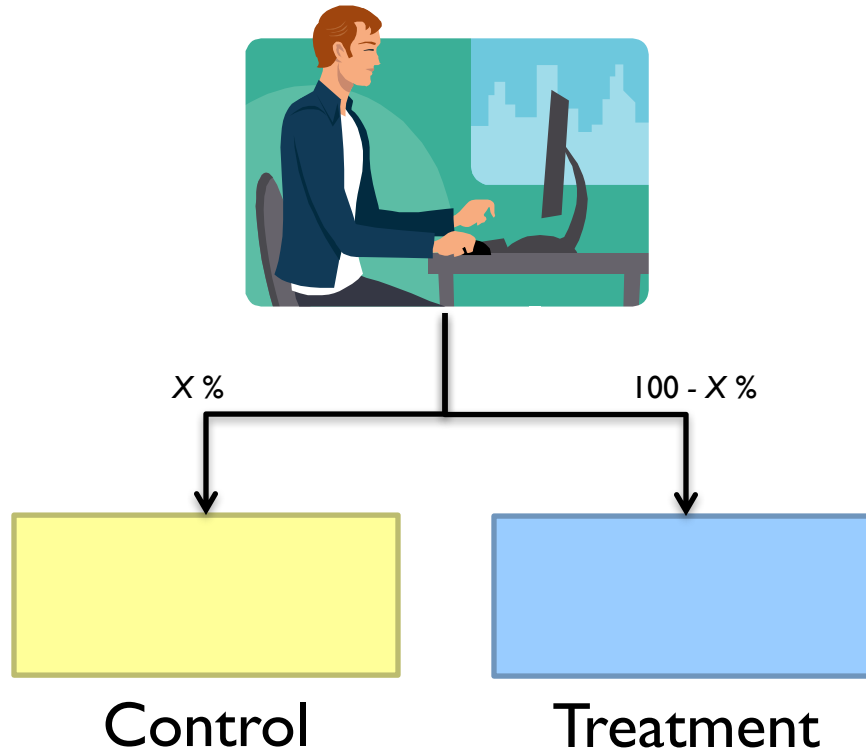
Cross-Validation

Typical Industry Setup



Why not cross-validation?

A/B Testing



Gather metrics, compare alternatives

A/B Testing: Complexities

Properly bucketing users

Novelty

Learning effects

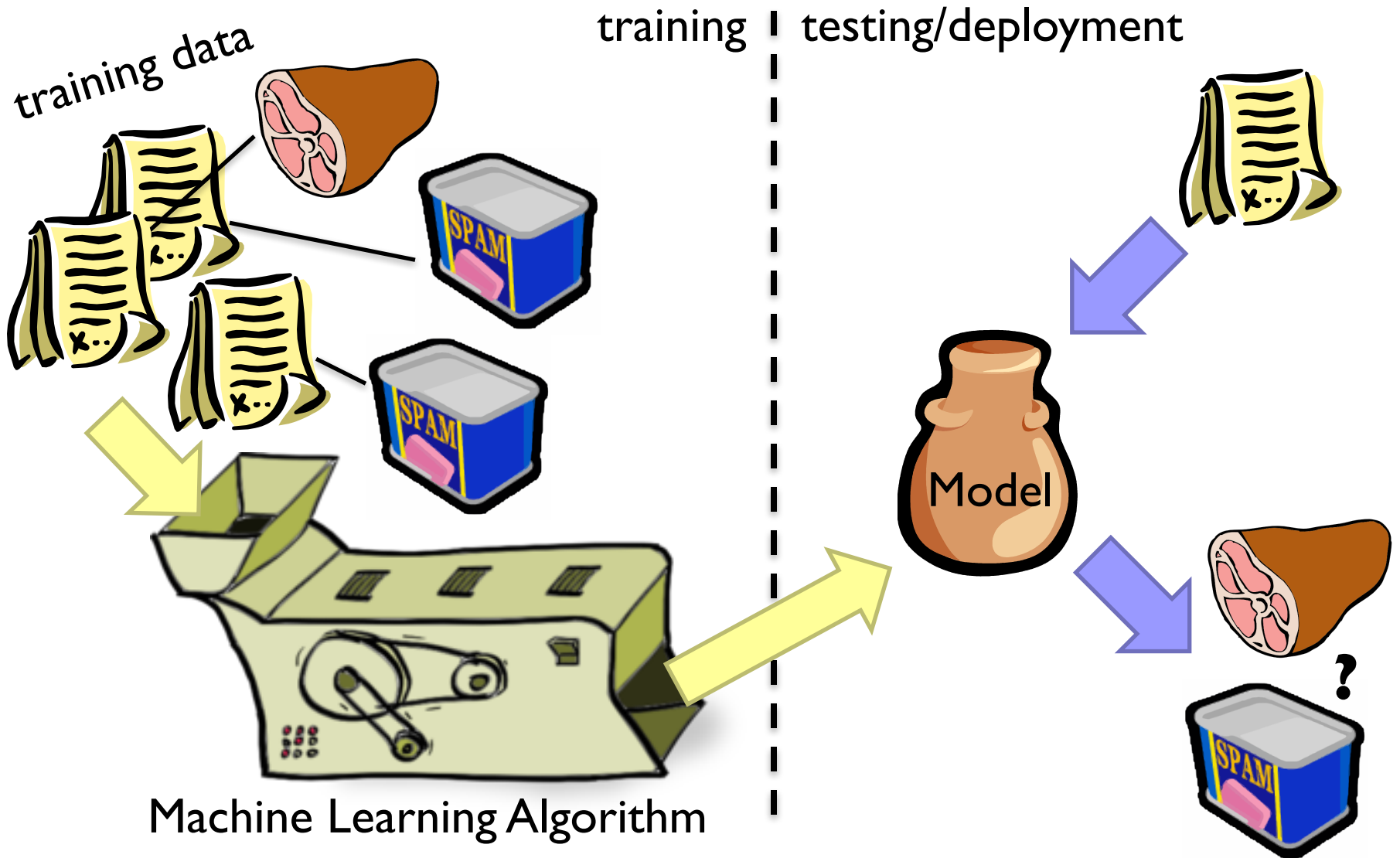
Long vs. short term effects

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

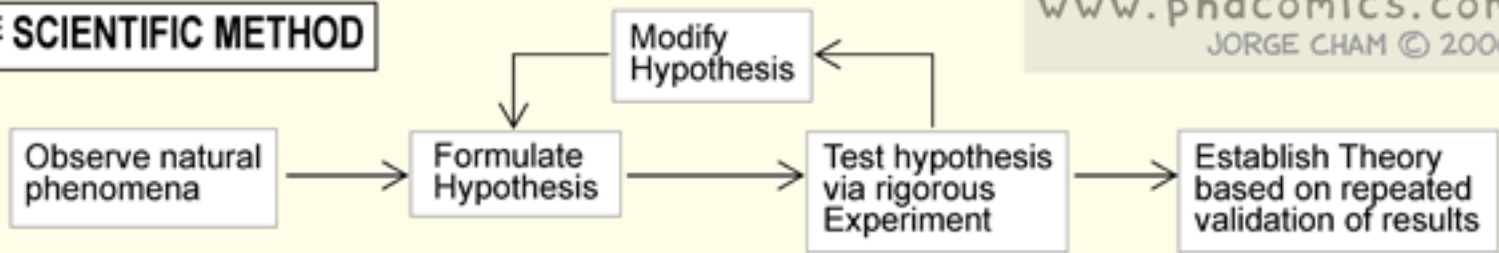
Train/Test

Does new model beat baseline?

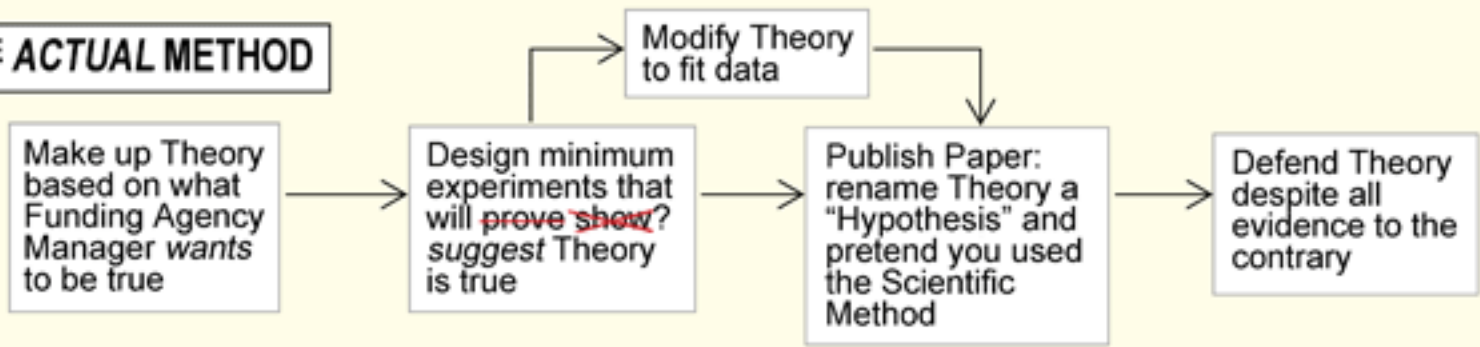
Yes: publish a paper!

No: try again!

THE SCIENTIFIC METHOD



THE ACTUAL METHOD



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

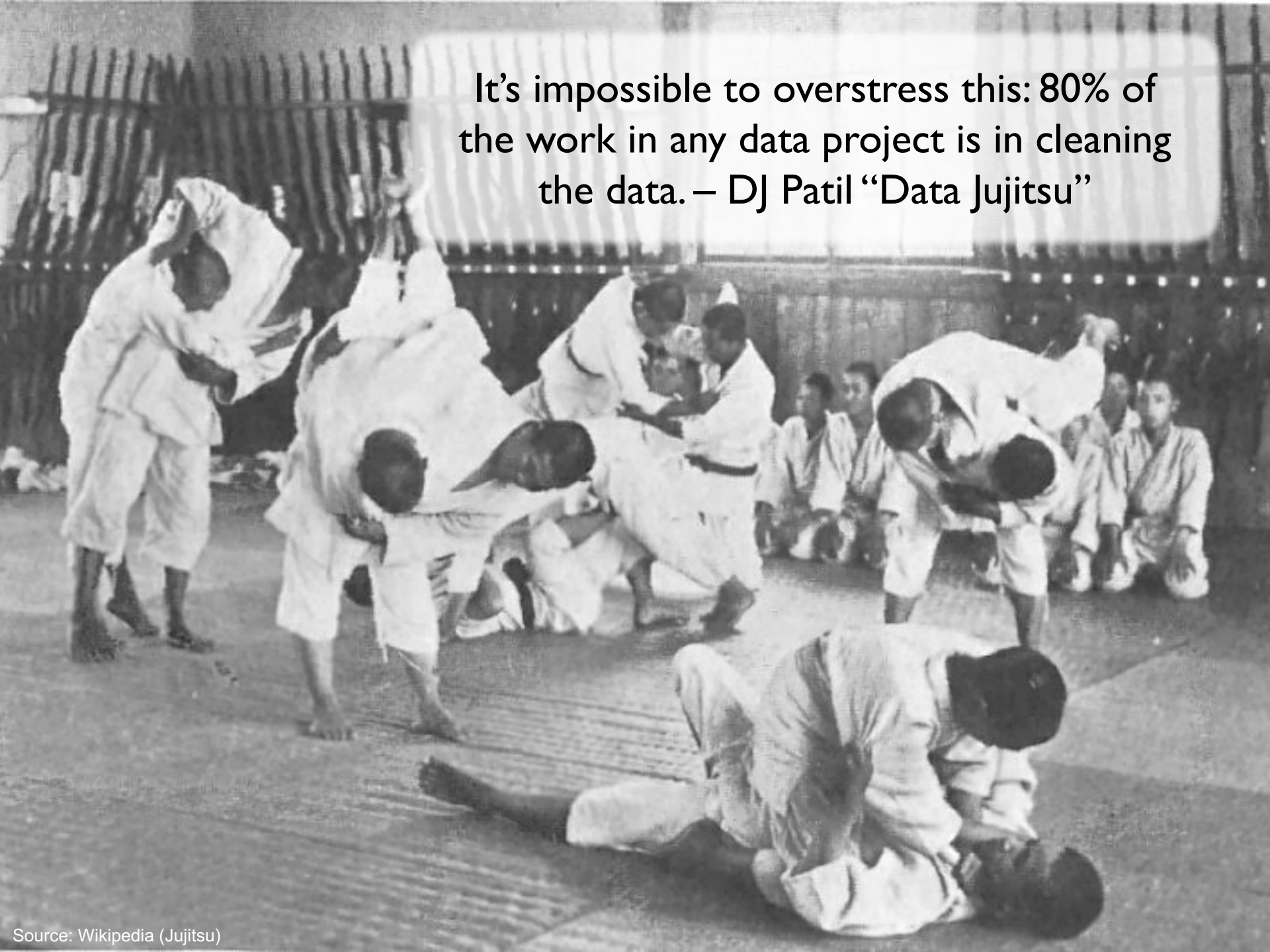
Extract features

“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”



SUBSCRIBE NOW

LOG IN



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...



P. Oscar Boykin
@posco



Following

OH: "... so to recap, tweets are statuses, favorites are favourings, retweets are shares."

← Reply ↻ Retweet ★ Favorite ... More

On naming things...

CamelCase

smallCamelCase

snake_case

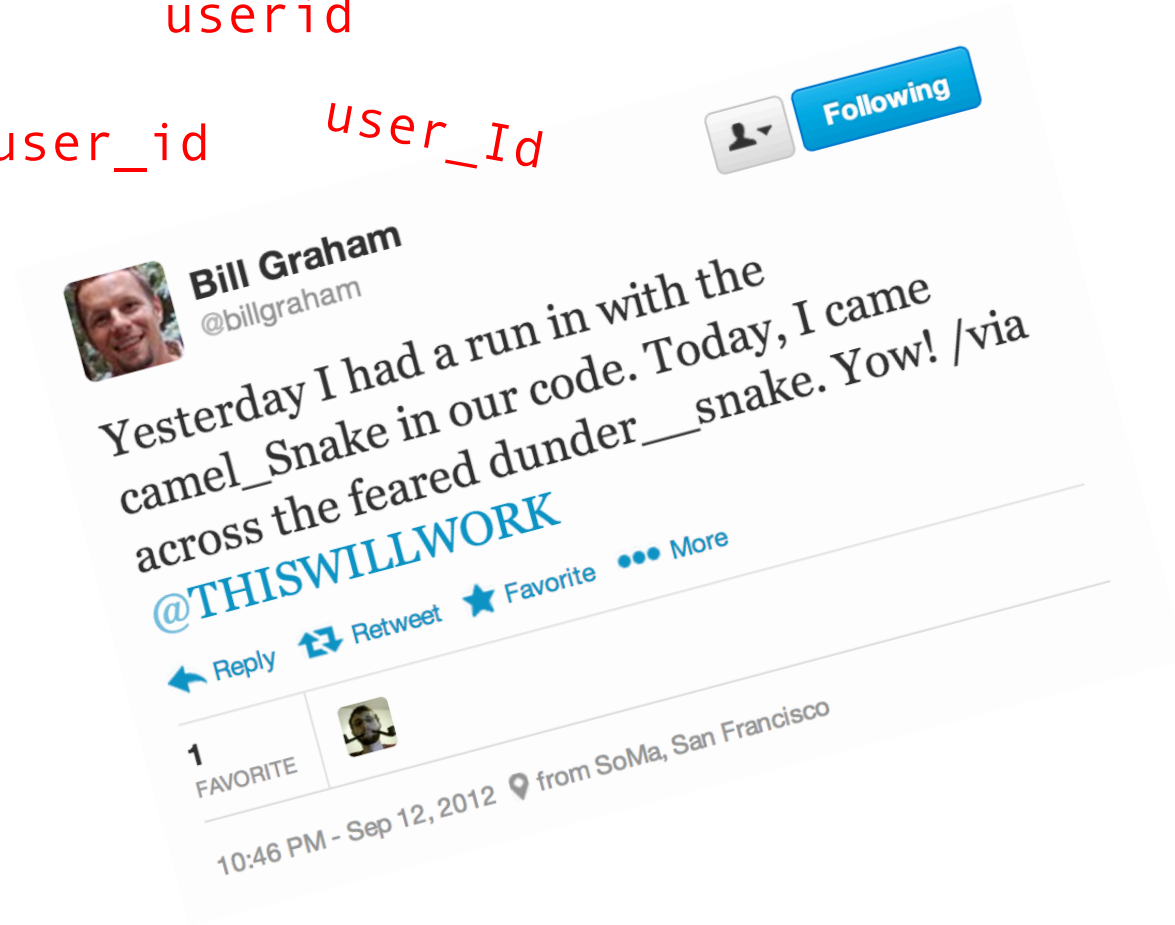
camel_Snake

dunder__snake

uid *UserId*

userId *userid*

user_id *user_Id*



On feature extraction...

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

It's over here...

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

Hang on, I don't remember...

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

Frontend Engineer

Develops new feature, adds logging code to capture clicks

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

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

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

[grumble, grumble, grumble]

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...



Congratulations, you're halfway there...

Does it actually work?

A/B testing

Is it fast enough?

Good, you're two thirds there...

Productionize



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!

A traditional Japanese rock garden (karesansui) featuring a gravel path with raked patterns, several large dark rocks, and a small stream. The garden is surrounded by lush greenery, including moss-covered bushes and trees, with a traditional Japanese building in the background.

Questions?