Georgia Tech

Advice for Getting Models Work

Mahdi Roozbahani
Georgia Tech

These slides are from Vivek Srikumar and Andrew Ng, and Chao Zhang

Outline

- Model Diagnostics
- Error Analysis
- Practical Advice

Debugging Machine Learning

Suppose you train an SVM or a logistic regression classifier for spam detection

You *obviously* follow best practices for finding hyper-parameters (such as cross-validation)

Your classifier is only 75% accurate

What can you do to improve it?

(assuming that there are no bugs in the code)

Different Ways to Improve Your Model

More training data

Features

1. Use more features
2. Use fewer features
3. Use other features

Better training

1. Run for more iterations
2. Use a different algorithm
3. Use a different classifier
4. Play with regularization

Tedious!

And prone to errors, dependence on luck

Let us try to make this process more methodical

First Step: Diagnose Your Model

Some possible problems:

1. Over-fitting (high variance)

2. Under-fitting (high bias)

3. Your learning does not converge

4. Are you measuring the right thing?

Overfitting v.s. Underfitting

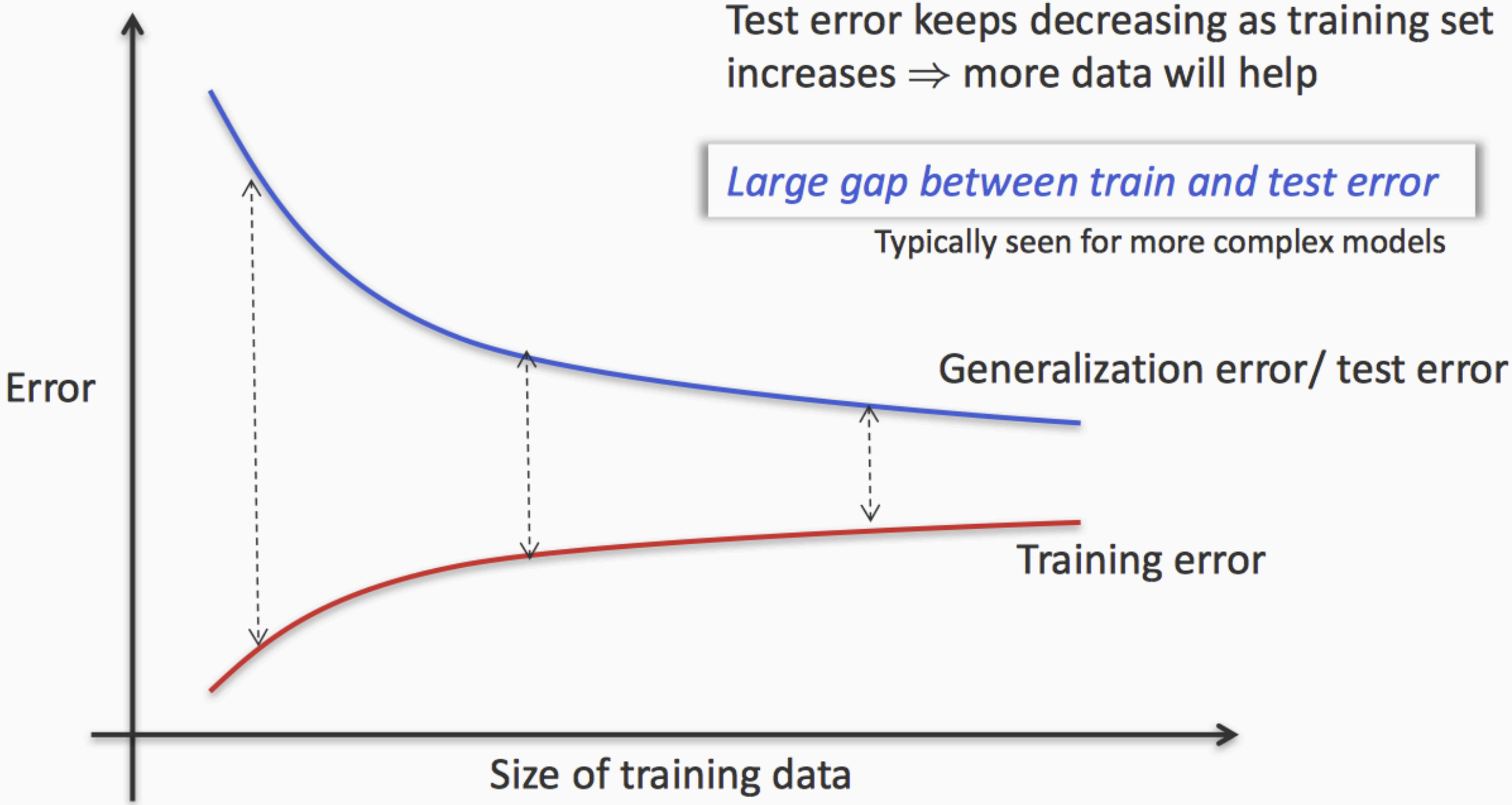
Over-fitting: The training accuracy is much higher than the test accuracy

- The model explains the training set very well, but poor generalization

Under-fitting: Both accuracies are unacceptably low

- The model can not represent the concept well enough

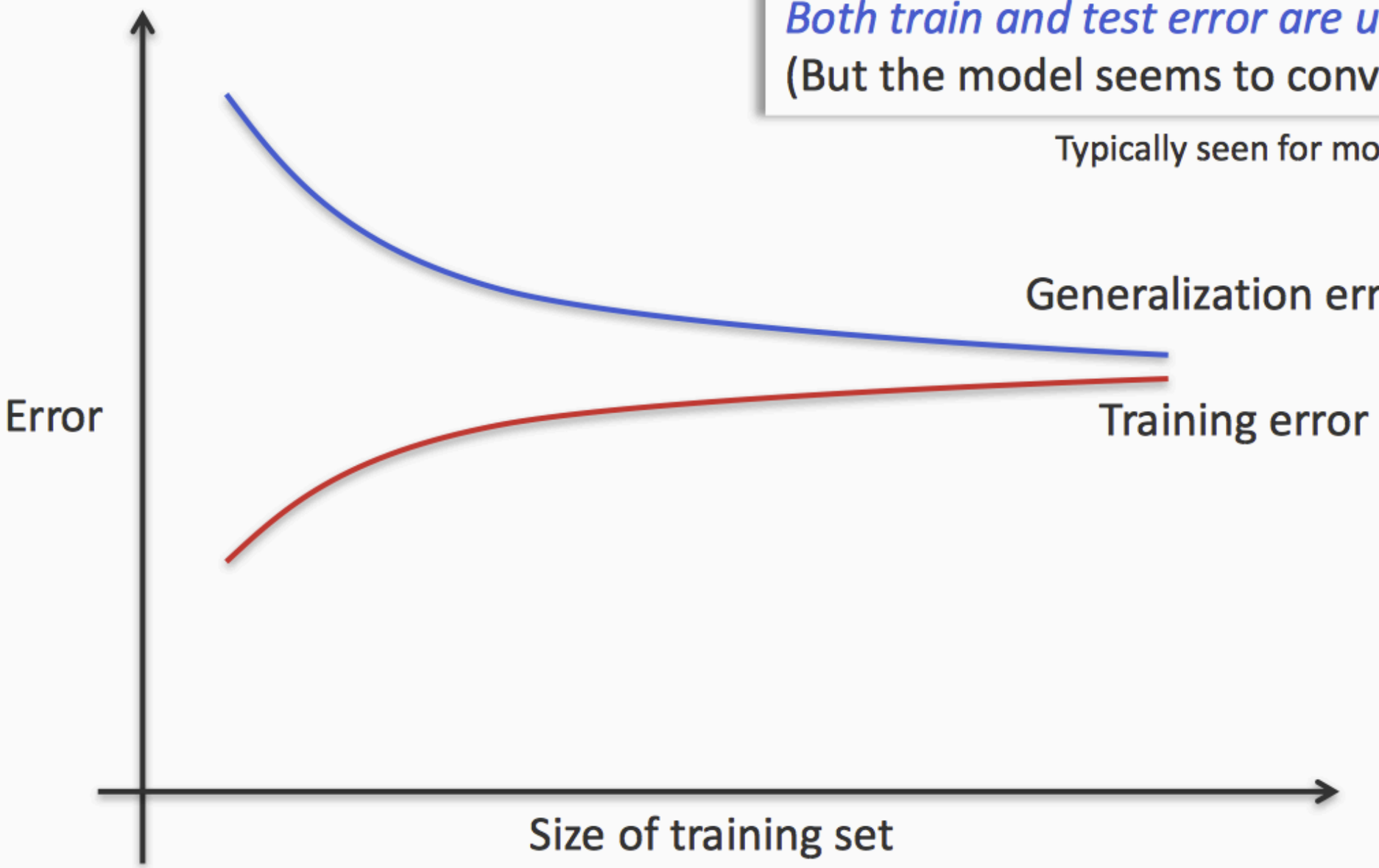
Overfitting (High Variance)



Underfitting (High Bias)

Both train and test error are unacceptable
(But the model seems to converge)

Typically seen for more simple models



Different Ways to Improve Your Model

More training data

Helps with over-fitting

Features

1. Use more features — Helps with under-fitting
2. Use fewer features — Helps with over-fitting
3. Use other features — Could help with over-fitting and under-fitting

Better training

1. Run for more iterations
2. Use a different algorithm
3. Use a different classifier
4. Play with regularization — Could help with over-fitting and under-fitting

Diagnostics

Some possible problems:

✓ Over-fitting (high variance)

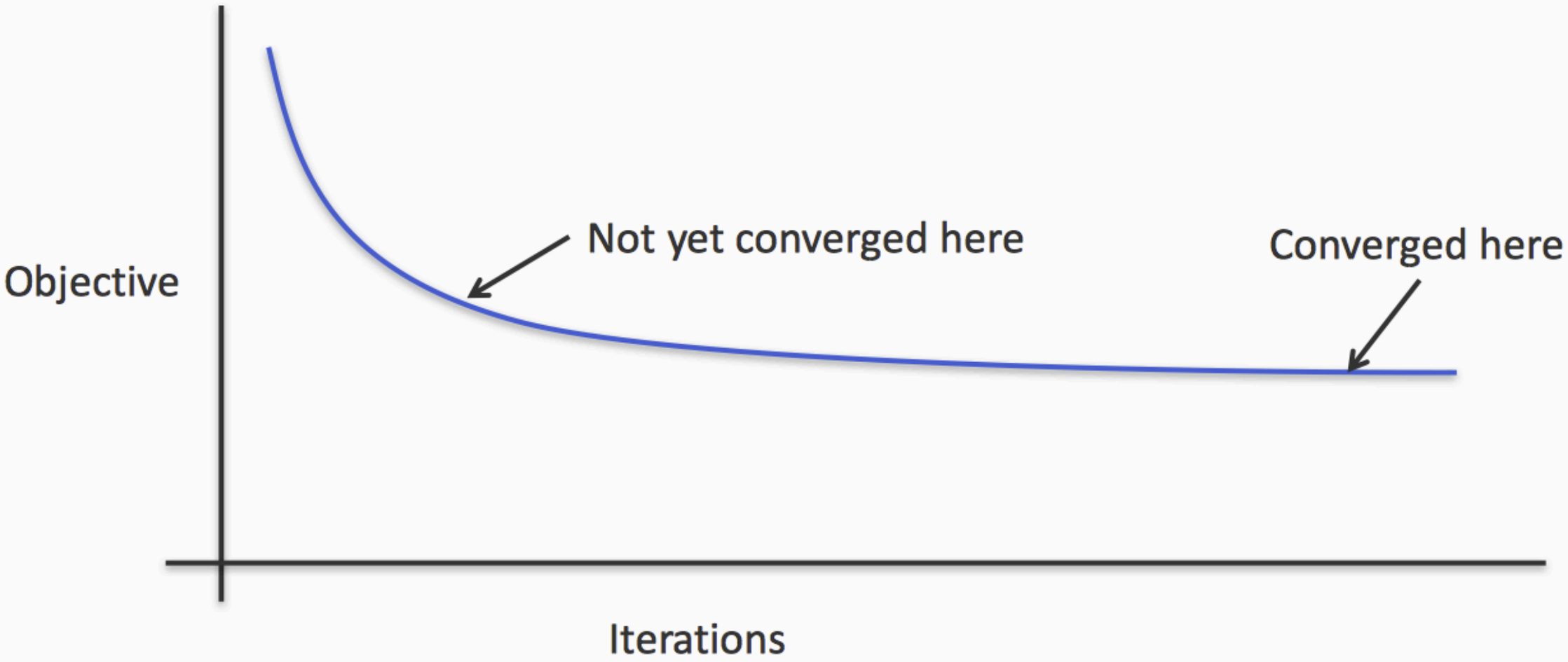
✓ Under-fitting (high bias)

3. Your learning does not converge

4. Are you measuring the right thing?

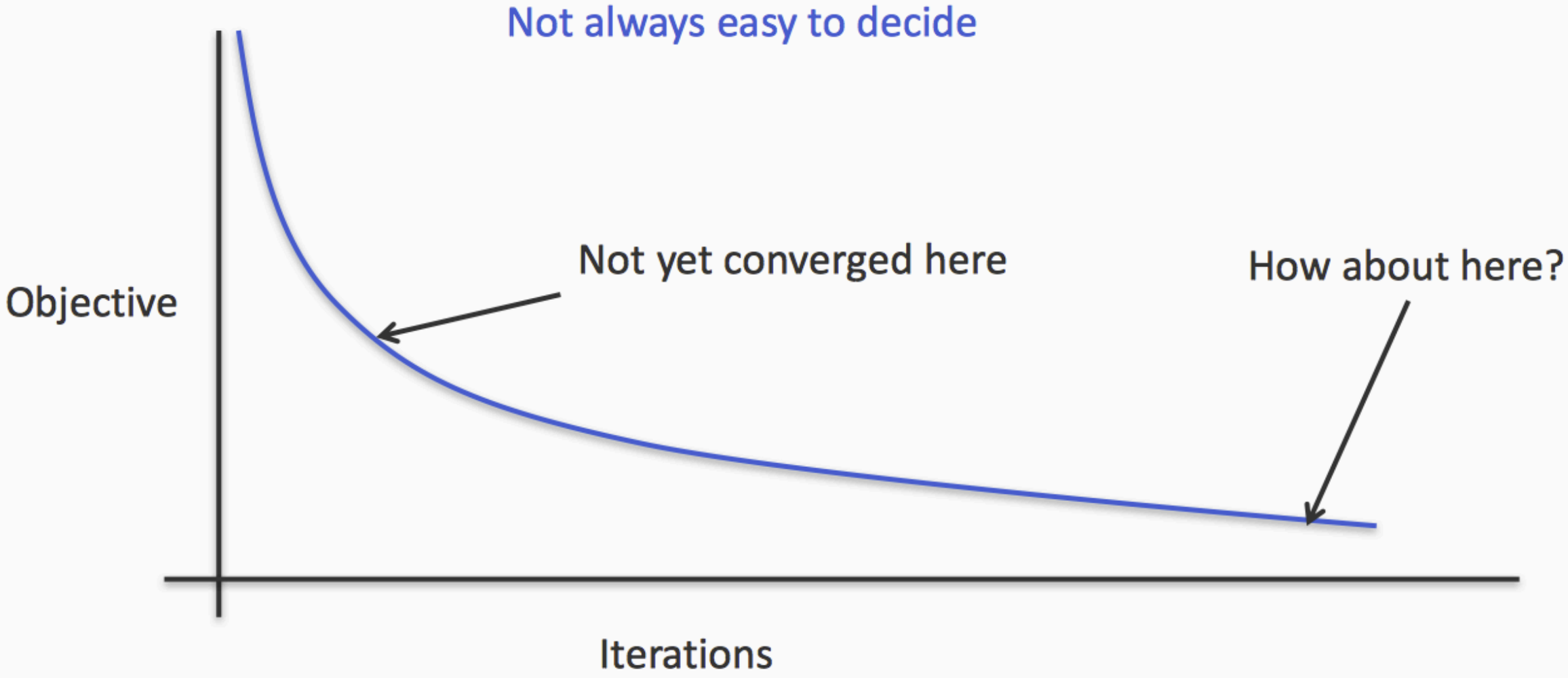
Have Your Model Converged?

If learning is framed as an optimization problem, track the objective



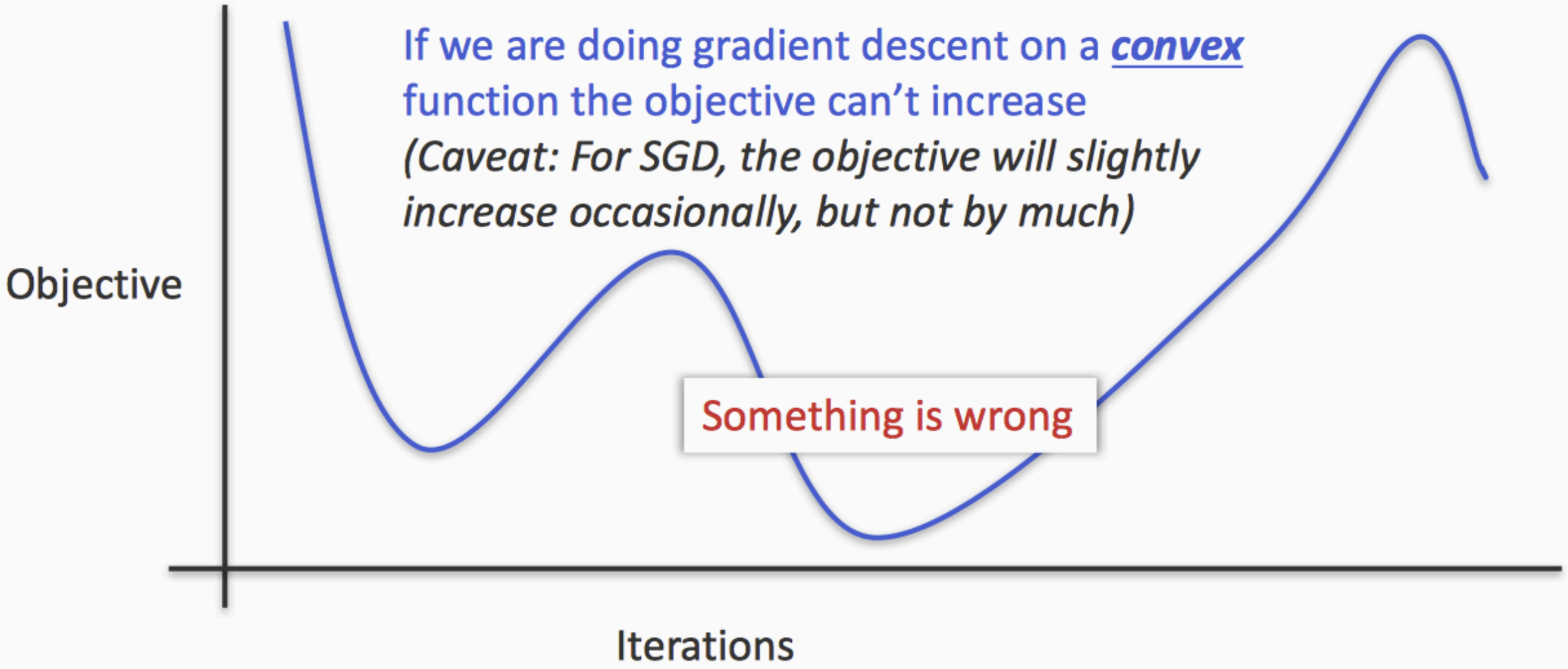
Have Your Model Converged?

If learning is framed as an optimization problem, track the objective



Have Your Model Converged?

If learning is framed as an optimization problem, track the objective



Different Ways to Improve Your Model

More training data

Helps with overfitting

Features

1. Use more features — Helps with under-fitting
2. Use fewer features — Helps with over-fitting
3. Use other features — Could help with over-fitting and under-fitting

Better training

1. Run for more iterations
2. Use a different algorithm — Track the objective for convergence
3. Use a different classifier
4. Play with regularization — Could help with over-fitting and under-fitting

Diagnostics

Some possible problems:

✓ Over-fitting (high variance)

✓ Under-fitting (high bias)

✓ Your learning does not converge

4. Are you measuring the right thing?

What to Measure

- Accuracy of prediction is the most common measurement

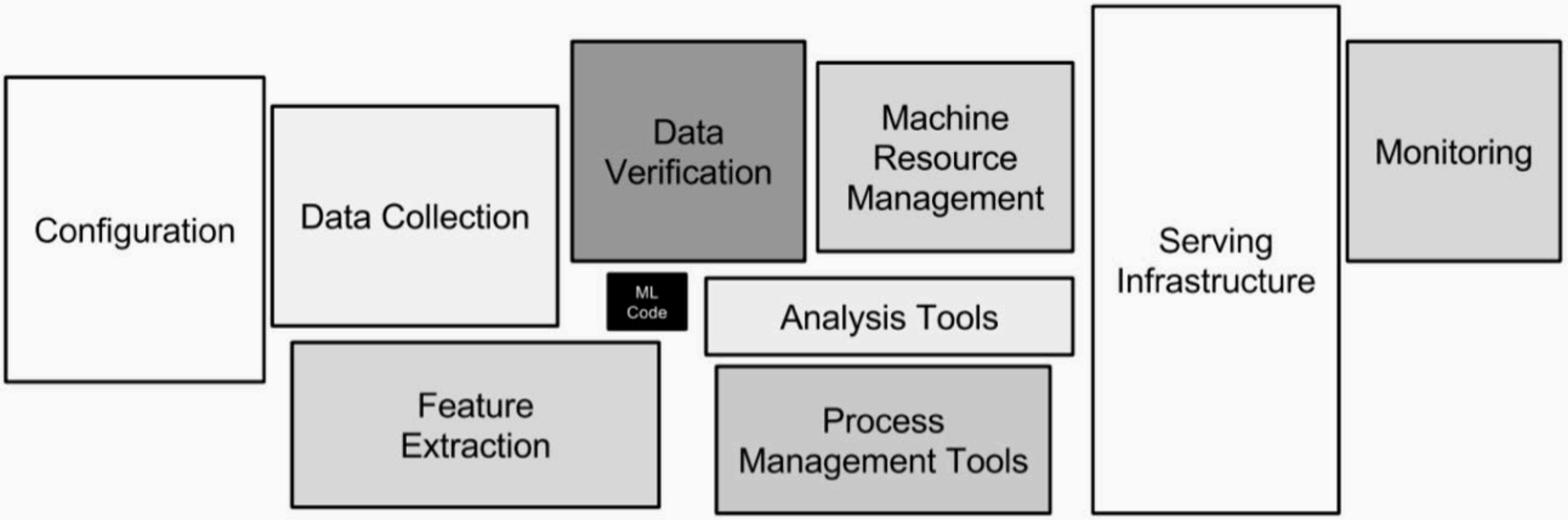
- But if your data set is unbalanced, accuracy may be misleading
  - 1000 positive examples, 1 negative example
  - A classifier that always predicts positive will get 99.9% accuracy. Has it really learned anything?

- Unbalanced labels → measure label specific precision, recall and F-measure
  - Precision for a label: Among examples that are predicted with label, what fraction are correct
  - Recall for a label: Among the examples with given ground truth label, what fraction are correct
  - F-measure: Harmonic mean of precision and recall

Outline

- Model Diagnostics
- Error Analysis
- Practical Advice

Machine Learning in Context



Error Analysis

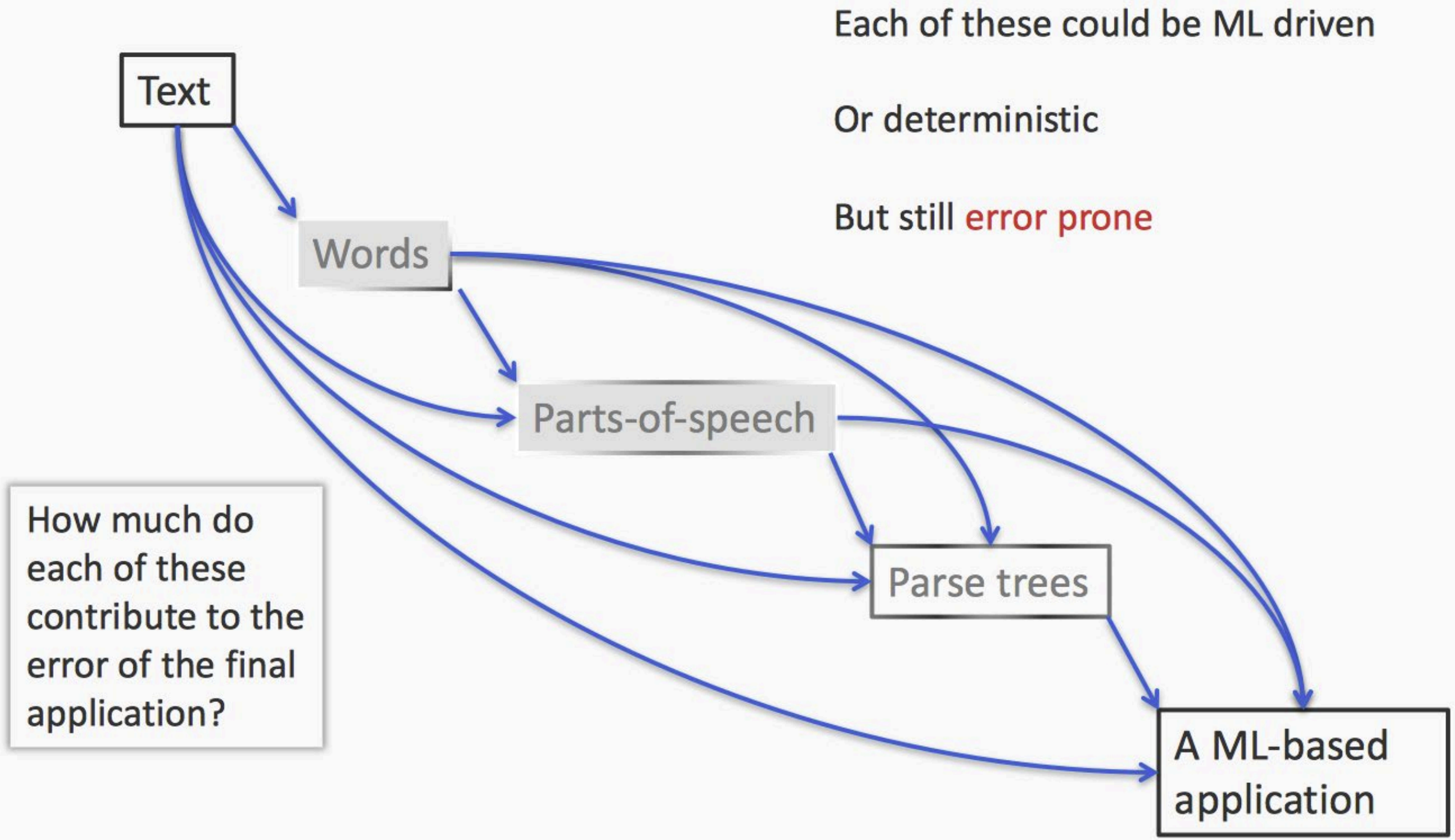
Generally machine learning plays a small role in a larger application

- Pre-processing
- Feature extraction (possibly by other ML based methods)
- Data transformations

How much do each of these contribute to the error?

Error analysis tries to explain why a system is not performing perfectly

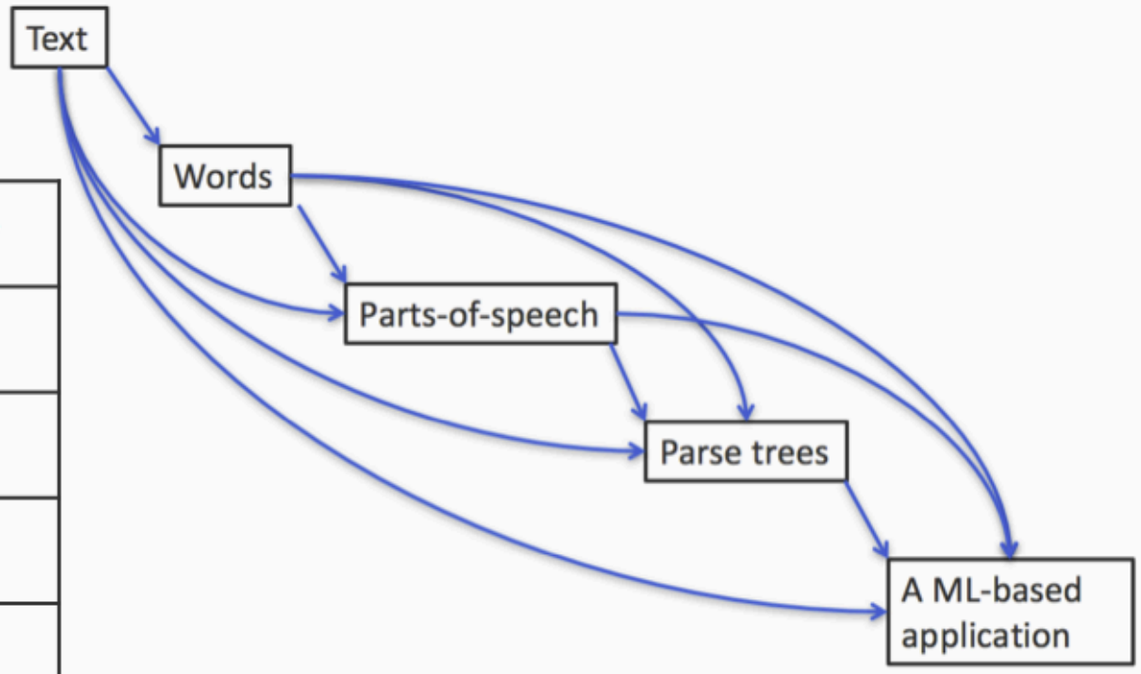
Example: Text Processing System



Example: Text Processing System

Plug in the ground truth for the intermediate components and see how much the accuracy of the final system changes

System	Accuracy
End-to-end predicted	55%
With ground truth words	60%
+ ground truth parts-of-speech	84 %
+ ground truth parse trees	89 %
+ ground truth final output	100 %



Error Analysis

Explaining difference between the performance between a strong model and a much weaker one (a baseline)

Usually seen with features

Suppose we have a collection of features and our system does well, but we don't know which features are giving us the performance

Evaluate simpler systems that progressively use fewer and fewer features to see which features give the highest boost

It is not enough to have a classifier that works; it is useful to know why it works.

Helps interpret predictions, diagnose errors and can provide an audit trail

Outline

- Model Diagnostics
- Error Analysis
- Practical Advice

Advice for ML Workflow in Practice

Say you want to build a classifier that identifies whether a real physical fish is salmon or tuna

How do you go about this?

The slow approach

1. Carefully identify features, get the best data, the software architecture, maybe design a new learning algorithm
2. Implement it and hope it works

Advantage: Perhaps a better approach, maybe even a new learning algorithm. Research.

The hacker's approach

1. First implement something
2. Use diagnostics to iteratively make it better

Advantage: Faster release, will have a solution for your problem quicker

Advice for ML Workflow in Practice

Say you want to build a classifier that identifies whether a real physical fish is salmon or tuna

How do you go about this?

The slow approach

1. Carefully identify features, get the best dat… arc… des… algorithm
2. Implement it and hope it works

Advantage: Perhaps a better approach, maybe even a new learning algorithm. Research.

The hacker's approach

1. First implement something … to … it better

Advantage: Faster release, will have a solution for your problem quicker

Be wary of premature optimization

Be equally wary of prematurely committing to a bad path

What to Watch Out For?

- Do you have the right evaluation metric?
  - And does your loss function reflect it?

- Beware of contamination: Ensure that your training data is not contaminated with the test set
  - Learning = generalization to new examples
  - Do not see your test set either. You may inadvertently contaminate the model
  - Beware of contaminating your features with the label!
  - (Be suspicious of perfect predictors)

What to Watch Out For?

- Be aware of bias vs. variance tradeoff (or over-fitting vs. under-fitting)

- Be aware that intuitions may not work in high dimensions
  - No proof by picture
  - Curse of dimensionality

- A theoretical guarantee may only be theoretical
  - May make invalid assumptions (eg: if the data is separable)
  - May only be legitimate with infinite data (eg: estimating probabilities)
  - Experiments on real data are equally important

What to Watch Out For?

- Learn simpler models first
  - If nothing, at least they form a baseline that you can improve upon
- Ensembles seem to work better
- Think about whether your problem is learnable at all
  - Learning = generalization