Lecture 17: EM

Winter 2018

Kai-Wei Chang

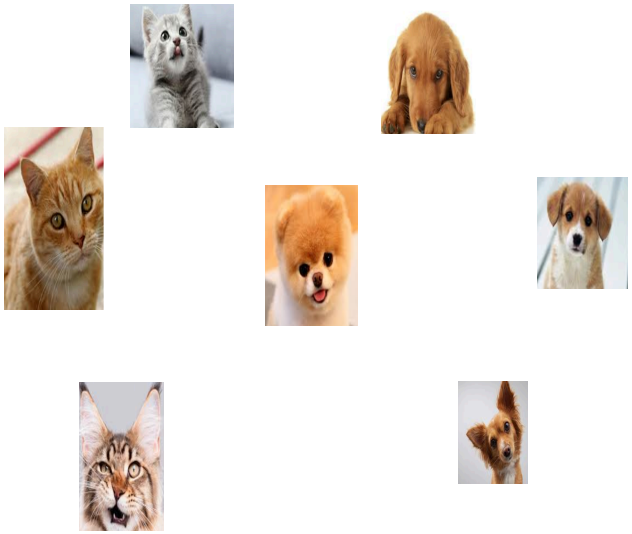
CS @ UCLA

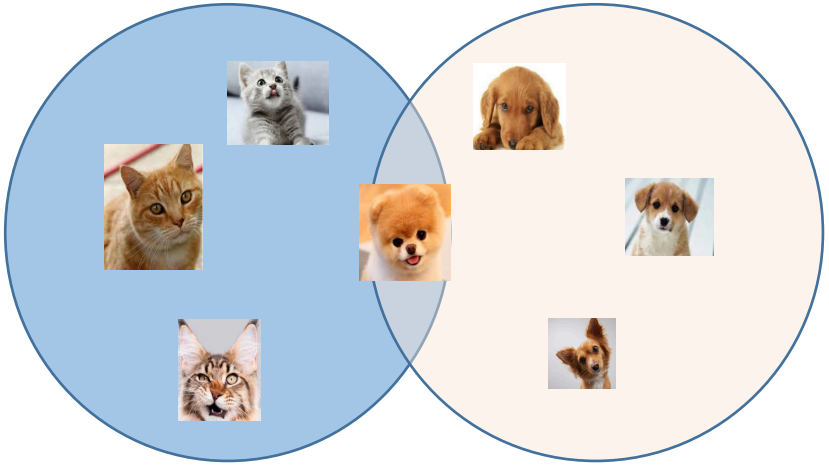
kw+cm146@kwchang.net

The instructor gratefully acknowledges Dan Roth, Vivek Srikuar, Sriram Sankararaman, Fei Sha, Ameet Talwalkar, Eric Eaton, and Jessica Wu whose slides are heavily used, and the many others who made their course material freely available online.

How about unsupervised learning

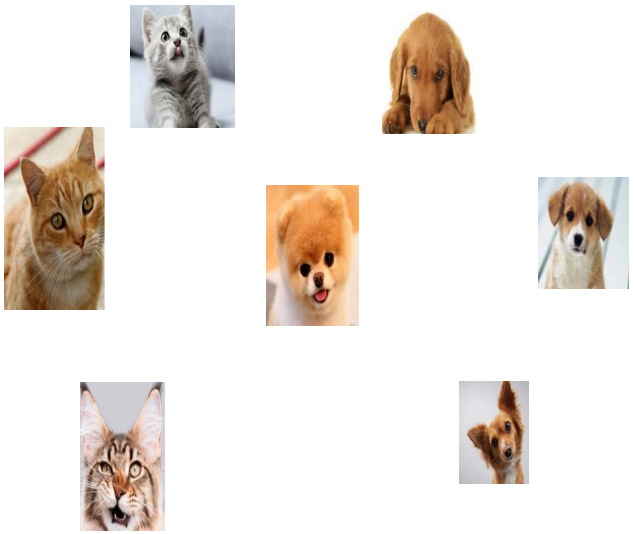
- In unsupervised learning, we only observed input distribution $\tilde{P}(X)$

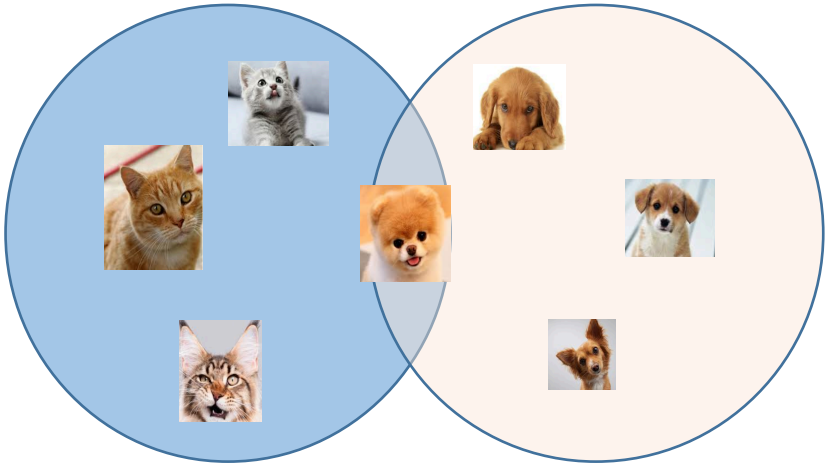




MLE in unsupervised learning

- We only have observation of $\tilde{P}(X)$
- In generative model, we have $P(X, Y \mid \Theta)$
- In discriminative model, we have $P(Y \mid \Theta, X)$
- Which model is more suitable for unsupervised learning?





MLE in unsupervised learning

- We only have observation of $\tilde{P}(X)$
- In generative model, we have $P(X, Y \mid \Theta)$
- We know $P(X \mid \Theta) = \sum_Y P(X, Y \mid \Theta)$
- Therefore, MLE is
  $\text{argmax}_\Theta P(X \mid \Theta) =$
  $\text{argmax}_\Theta \sum_Y P(X, Y \mid \Theta)$

EM algorithm

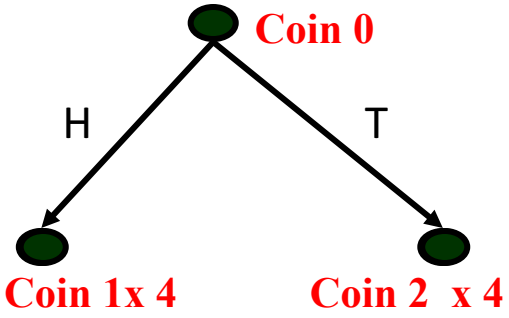
- EM algorithm solves $\text{argmax}_\Theta \sum_Y P(X, Y \mid \Theta)$ by iteratively updating Θ
- In general, known to converge to a local maximum of the maximum likelihood function

Three Coins Example

- We observe a series of coin tosses generated in the following way:
- A person has three coins.
  - Coin 0: probability of Head is α
  - Coin 1: probability of Head p
  - Coin 2: probability of Head q

- Consider the following coin-tossing scenarios:

Scenario I



- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

Observing the sequence
HHHHT, THTHT, HHHHT, HHTTH

produced by Coin 0 , Coin1 and Coin2

Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Scenario I

Supervised Learning

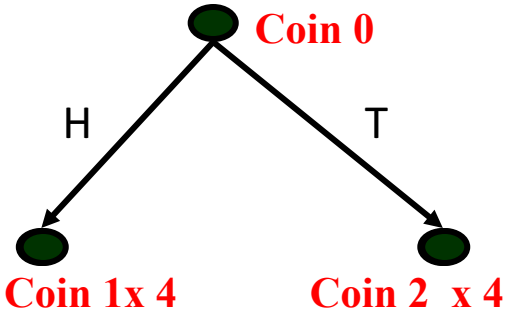
- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

  Observing the sequence
  HHHHT, THTHT, HHHHT, HHTTH

  produced by Coin 0 , Coin1 and Coin2

  Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Scenario II



- Toss coin 0.
  If Head – toss coin 1; o/w – toss coin 2

Observing the sequence
HHHT, HTHT, HHHT, HTTH

produced by ~~Coin 0~~, Coin1 and/or Coin2

Question: Estimate most likely values for p, q (the probability of H in each coin) and the probability to use each of the coins (a)

Intuition of EM algorithm

- Use an iterative approach for estimating the parameters:
  - Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.
  - Now, compute the most likely value of the parameters. [recall the scenario I]
  - Compute the likelihood of the data given this model.
  - Re-estimate the initial parameter setting: set them to maximize the likelihood of the data.

Step 1: initialization

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.

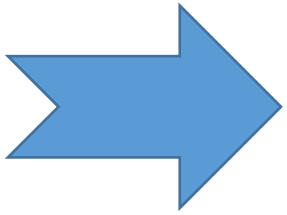
	coin 1=H	coin 1=T
HHHT	100%	0 %
HTHT	100%	0%
HHHT	100%	0%
HTTH	0%	100%

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

	coin 1=H	coin 1=T
HHHT	100%	0 %
HTHT	100%	0%
HHHT	100%	0%
HTTH	0%	100%



HHHHT
HHTHT
HHHHT
THTTH

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

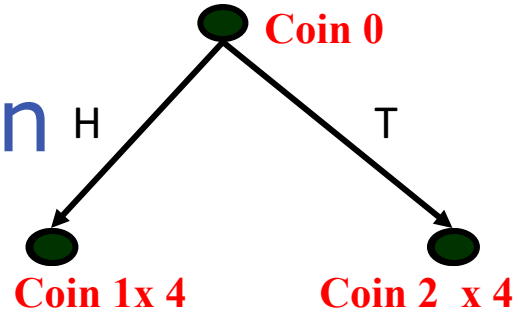
HHHHT
HHTHT
HHHHT
THTTH

$$\alpha_1 = \frac{3}{3+1} = \frac{3}{4}$$

$$p_1 = \frac{8}{8+4} = \frac{2}{3}$$

$$q_1 = \frac{2}{2+2} = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ coin 1=H

$(1-\alpha_1)q_1^{\#H}(1-q_1)^{\#T}$ coin 1=T

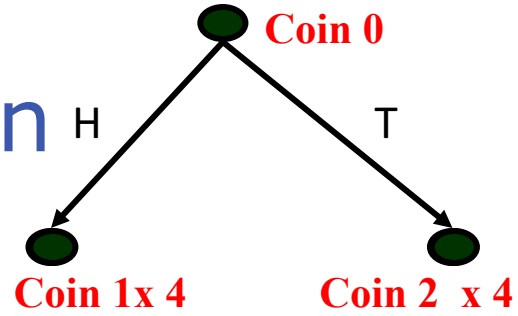
	coin 1=H	coin 1=T
HHHT	$\frac{3}{4}\left(\frac{2}{3}\right)^3\frac{1}{3}$	$\frac{1}{4}\left(\frac{1}{2}\right)^3\frac{1}{2}$
HTHT	$\frac{3}{4}\left(\frac{2}{3}\right)^2\left(\frac{1}{3}\right)^2$	$\frac{1}{4}\left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)^2$
HHHT	$\frac{3}{4}\left(\frac{2}{3}\right)^3\frac{1}{3}$	$\frac{1}{4}\left(\frac{1}{2}\right)^3\frac{1}{2}$
HTTH	$\frac{3}{4}\left(\frac{2}{3}\right)^2\left(\frac{1}{3}\right)^2$	$\frac{1}{4}\left(\frac{1}{2}\right)^2\left(\frac{1}{2}\right)^2$

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

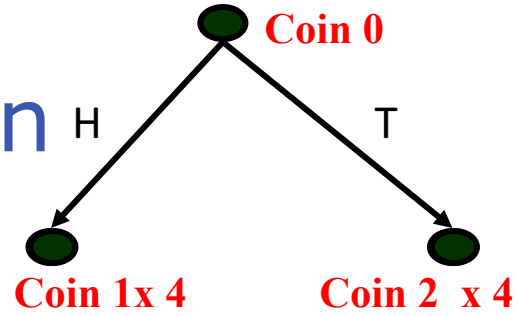
	$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ coin 1=H	$(1-\alpha_1)q_1^{\#H}(1-q_1)^{\#T}$ coin 1=T
HHHT	0.074	0.0156
HTHT	0.037	0.0156
HHHT	0.074	0.0156
HTTH	0.037	0.0156

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_1 p_1^{\#H}(1-p_1)^{\#T}$ $(1-\alpha_1)q_1^{\#H}(1-q_1)^{\#T}$

	coin 1=H	coin 1=T
HHHT	82.6%	17.4%
HTHT	*70.3%*	29.7%
HHHT	82.6%	17.4%
HTTH	*70.3%*	29.7%

$$\alpha_1 = \frac{3}{4}$$

$$p_1 = \frac{2}{3}$$

$$q_1 = \frac{1}{2}$$

e.g., $\frac{0.074}{0.074+0.0156} = 82.6\%$

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

	coin 1=H	coin 1=T
HHHT	82.6%	17.4%
HTHT	*70.3%*	29.7%
HHHT	82.6%	17.4%
HTTH	*70.3%*	29.7%

HHHHT 82.6%
THHHT 17.4%
HHTHT 70.3%
THTHT 29.7%
HHHHT 82.6%
THHHT 17.4%
HHTTH 70.3%
THTTH 29.7%

Step 2: Maximum Conditional Likelihood

Coin 0: probability of Head is α
Coin 1: probability of Head p
Coin 2: probability of Head q

- Now, compute the most likely value of the parameters. [recall the scenario I]

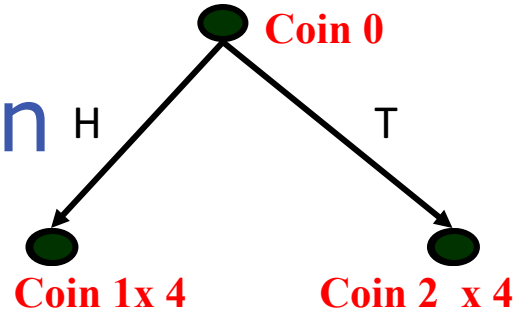
HHHHT 82.6%
HHTHT 70.3%
HHHHT 82.6%
HHTTH 70.3%
THHHT 17.4%
THTHT 29.7%
THHHT 17.4%
THTTH 29.7%

$$\alpha_2 = \frac{82.6\times 2 + 70.3\times 2}{400} = 76.5\%$$

$$p_2 = \frac{82.6\times 6 + 70.3\times 4}{82.6\times 8 + 70.3\times 8} = 63.5\%$$

$$q_2 = \frac{17.4\times 6 + 29.7\times 4}{17.4\times 8 + 29.7\times 8} = 59.2\%$$

Step3: Likelihood Estimation



- Compute the likelihood of the data given this model

$\alpha_2 p_2^{\#H}(1-p_2)^{\#T}$ coin 1=H

$(1-\alpha_2)q_2^{\#H}(1-q_2)^{\#T}$ coin 1=T

HHHT
HTHT
HHHT
HTTH

$$\alpha_2 = 76.5\%$$

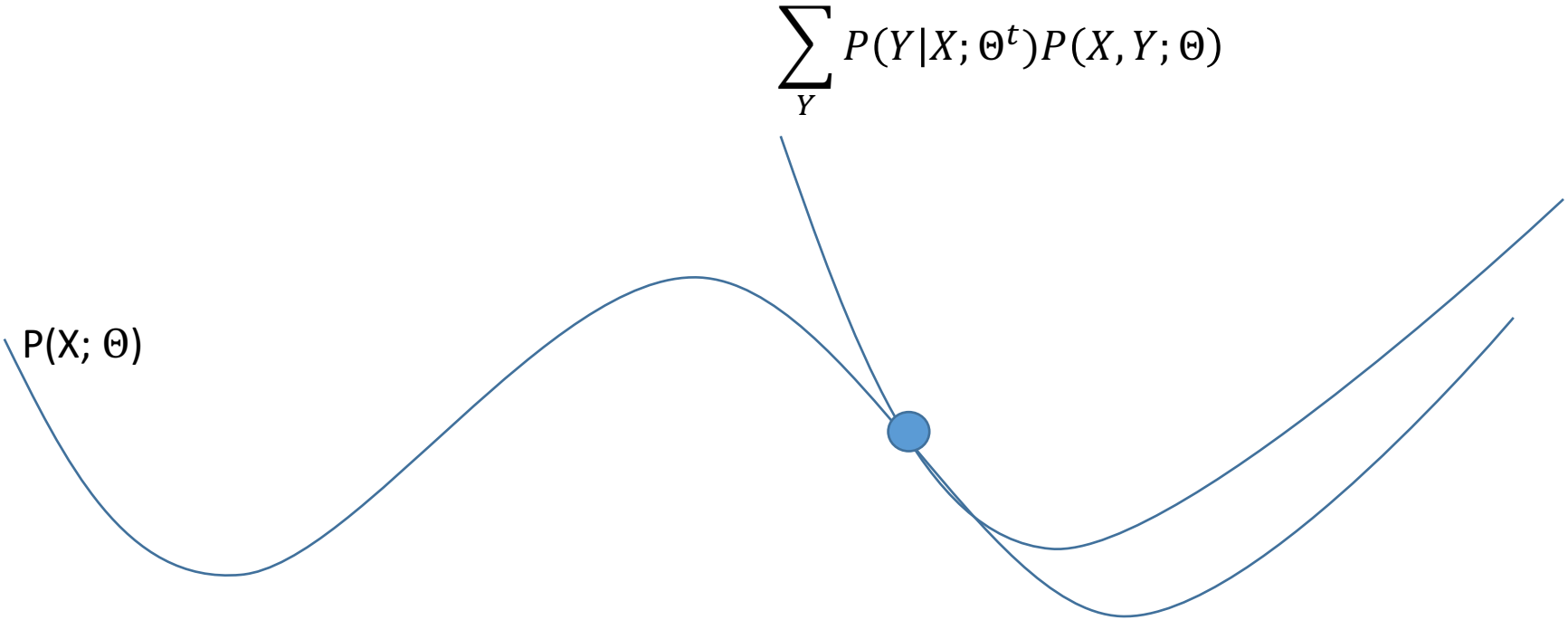
$$p_2 = 63.5\%$$

$$q_2 = 59.2\%$$

Intuition of EM algorithm

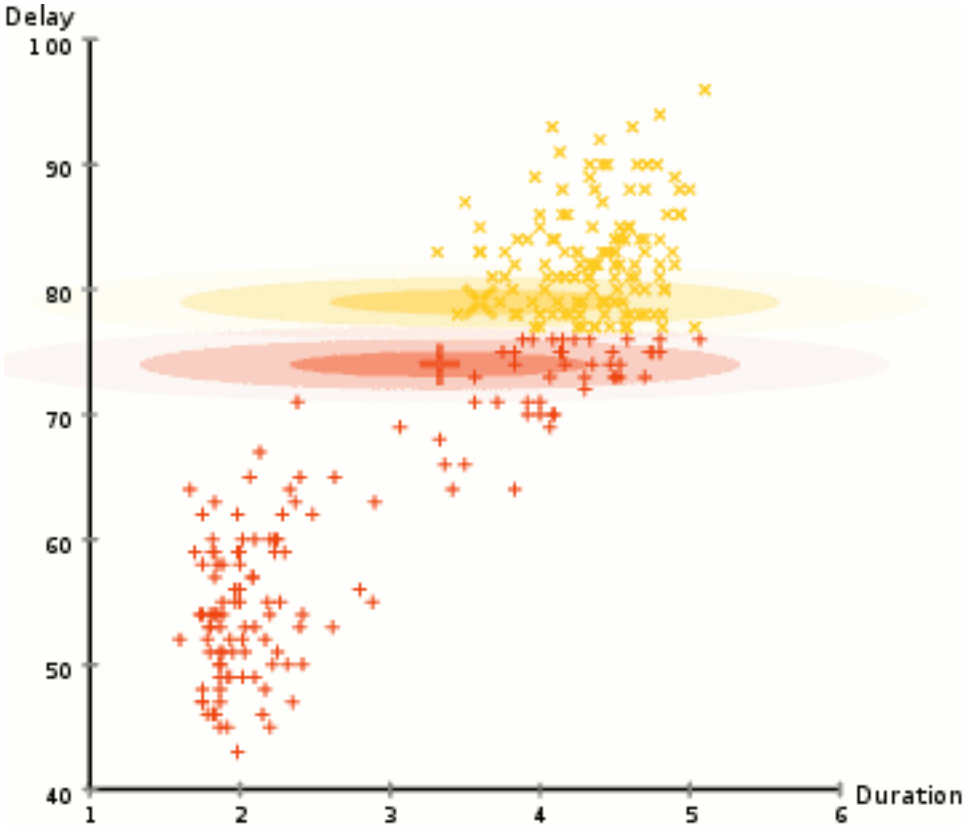
- Use an iterative approach for estimating the parameters:
  - Guess the probability that a given data point came from Coin 1 or 2; Generate fictional labels, weighted according to this probability.
  - Now, compute the most likely value of the parameters. [recall the scenario I]
  - Compute the likelihood of the data given this model.
  - Re-estimate the initial parameter setting: set them to maximize the likelihood of the data.

(not in exam) EM from optimization perspective



Real world Example

GMM clustering of Old Faithful eruption data



GMM as the marginal distribution P(x) of a joint distribution P(x, z)

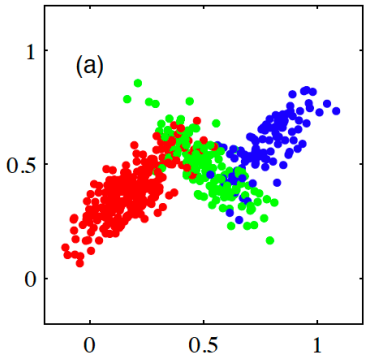
- Remember, in GMM, we model the marginal probability as

$$p(\boldsymbol{x}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$$

$$\omega_k = p(z = k)$$

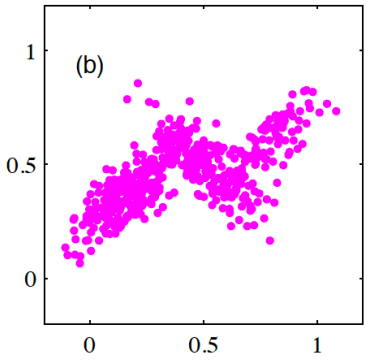
Example

$$p(x) = \sum_z p(x, z) = \sum_z P(z)\, p(x|z)$$



The conditional distribution between $\boldsymbol{x}$ and z (representing color) are

$$p(\boldsymbol{x}|z = red) = N(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1)$$
$$p(\boldsymbol{x}|z = blue) = N(\boldsymbol{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2)$$
$$p(\boldsymbol{x}|z = green) = N(\boldsymbol{x}|\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$



The marginal distribution is thus

$$p(\boldsymbol{x}) = p(red)N(\boldsymbol{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma}_1) + p(blue)N(\boldsymbol{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma}_2) + p(green)N(\boldsymbol{x}|\boldsymbol{\mu}_3, \boldsymbol{\Sigma}_3)$$

ω_k

Iterative procedure

- Let θ represent all parameters $\{\omega_k, \mu_k, \Sigma_k\}$

Step 0: initialize $\boldsymbol{\theta}$ with some values (random or otherwise)

Step 1: compute γ_{nk} using the current $\boldsymbol{\theta}$

Step 2: update $\boldsymbol{\theta}$ using the just computed γ_{nk}

Step 3: go back to Step 1

γ_{nk}: given a data point x_n how likely it belongs to k^{th} cluster

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Since γ_{nk} is binary, the previous solution is nothing but

- For ω_k: count the number of data points whose z_n is k and divide by the total number of data points (note that $\sum_k \sum_n \gamma_{nk} = N$)
- For $\boldsymbol{\mu}_k$: get all the data points whose z_n is k, compute their mean
- For $\boldsymbol{\Sigma}_k$: get all the data points whose z_n is k, compute their covariance matrix

Estimate γ_{nk}

- γ_{nk} the assignment of instance n to cluster k, can be defined as $\gamma_{nk} = P(z_n = k|x_n)$
- Can be computed vis the posterior probability

$$p(z_n = k|\boldsymbol{x}_n) = \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{p(\boldsymbol{x}_n)} = \frac{p(\boldsymbol{x}_n|z_n = k)p(z_n = k)}{\sum_{k'=1}^{K} p(\boldsymbol{x}_n|z_n = k')p(z_n = k')}$$

$N(x|\mu_k, \Sigma_k)$ ω_k

Parameter estimation for GMMs

- If cluster assignments are observed $\{z_n\}$ are given
  - We know the cluster of each point
  - Let $\gamma_{nk} = 1$ if instance n belongs to cluster k, otherwise $\gamma_{nk} = 0$
- Then the maximum likelihood estimation is

$$\omega_k = \frac{\sum_n \gamma_{nk}}{\sum_k \sum_n \gamma_{nk}}, \quad \boldsymbol{\mu}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} \boldsymbol{x}_n$$

$$\boldsymbol{\Sigma}_k = \frac{1}{\sum_n \gamma_{nk}} \sum_n \gamma_{nk} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^{\mathrm{T}}$$

Review & Practical advices

ML and the world

- Bias vs Variance
- Diagnostics of your learning algorithm
- Error analysis
- Injecting machine learning into *Your Favorite Task*

Making ML work in the world

Mostly experiential advice

Also based on what other people have said

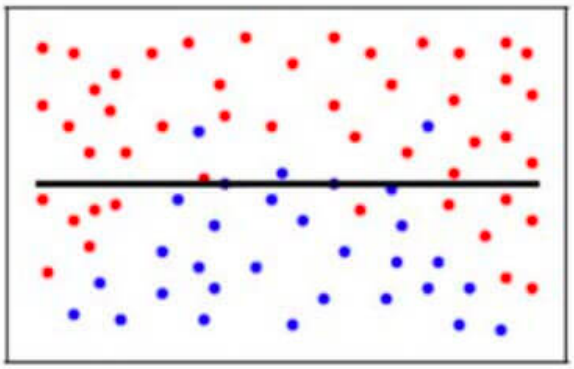
Bias and variance

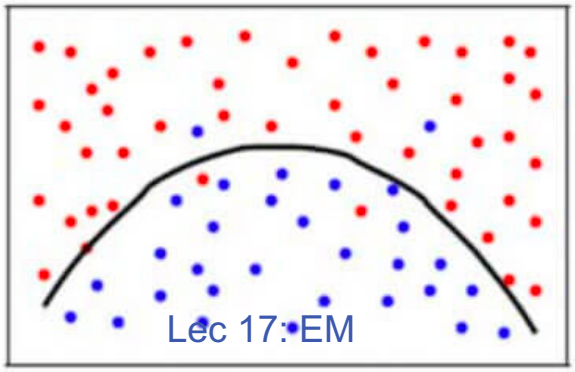
Every learning algorithm requires assumptions about the hypothesis space.

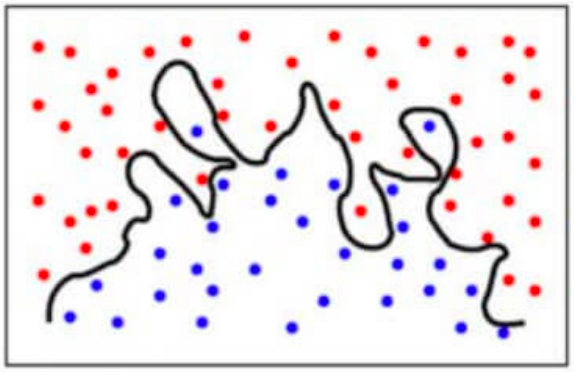
Eg: “My hypothesis space is

- …linear”
- …decision trees with 5 nodes”
- …deep neural network with 12 layers”

Underfitting ⟷ Overfitting







Managing bias and variance

- Ensemble methods can reduce both bias and variance
  - Multiple classifiers are combined
  - Eg: Bagging, boosting
- Decision trees of a fixed depth
  - Increasing depth decreases bias, increases variance
- SVMs
  - Stronger regularization increases bias, decreases variance
- K nearest neighbors
  - Increasing k generally increases bias, reduces variance
- Neural Network
  - Early stopping, dropout

Tune your parameters!!

- Always tune parameters when comparing different settings / algorithms
- Never tune your parameters on the test set

ML and the world

- Bias vs Variance
- Diagnostics of your learning algorithm
- Error analysis
- Injecting machine learning into *Your Favorite Task*

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?

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

Different ways to improve your model

More training data

Features

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

Tedious!

And prone to errors, dependence on luck

Let us try to make this process more methodical

Better training

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

First, diagnostics

Easier to fix a problem if you know where it is

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?

Detecting over or under fitting

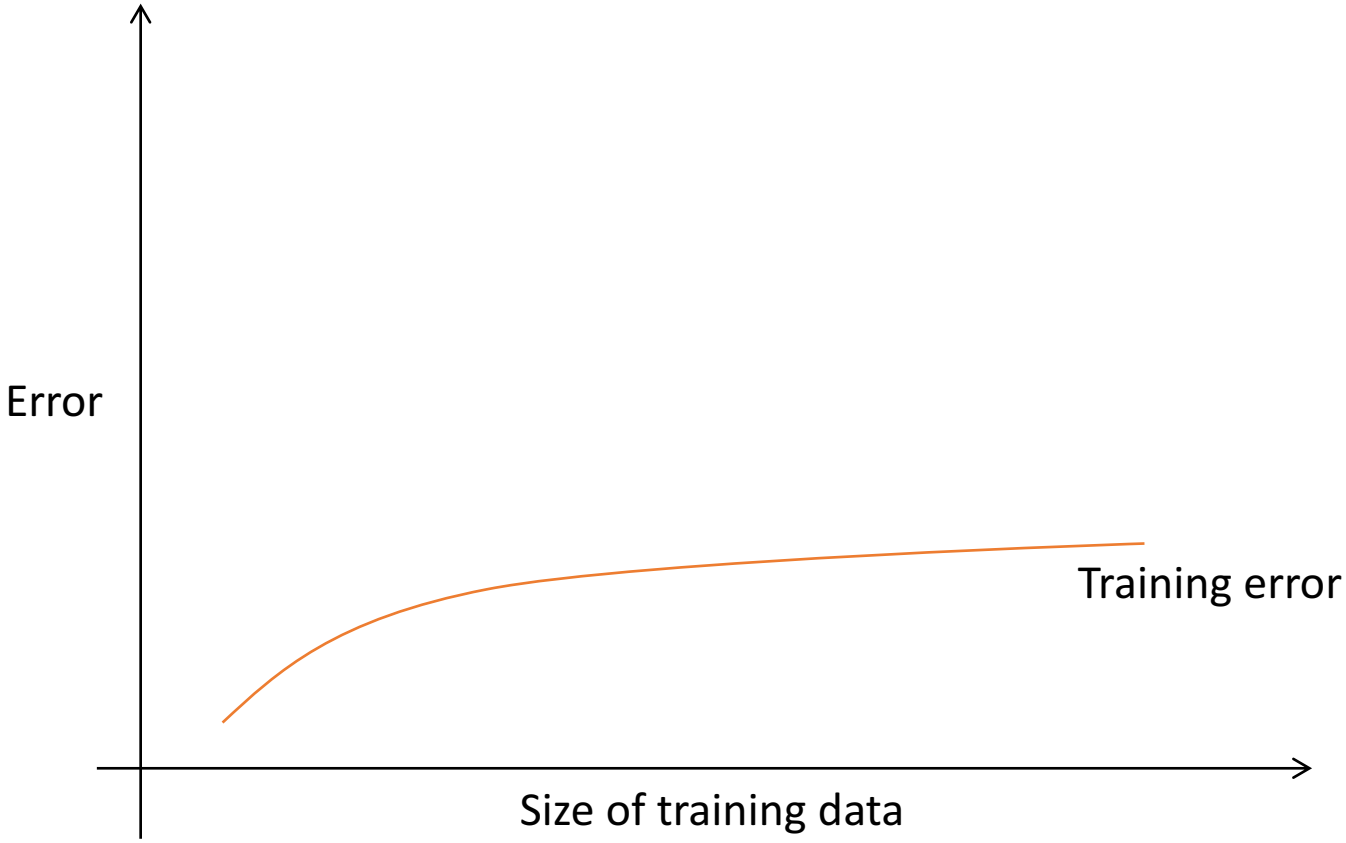
Over-fitting: The training accuracy is much higher than the test accuracy

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

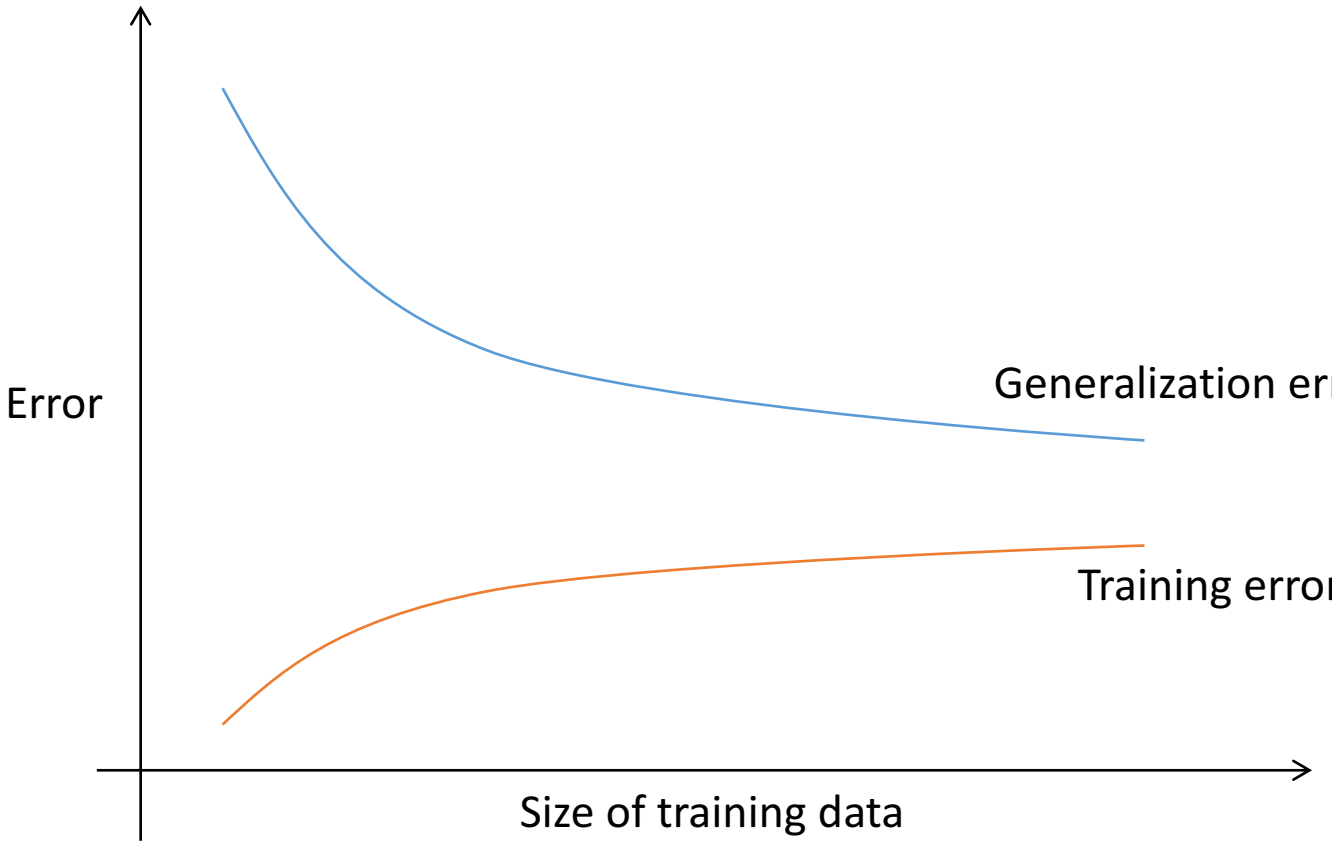
Under-fitting: Both accuracies are unacceptably low

- The model can not represent the concept well enough

Detecting high variance using learning curves

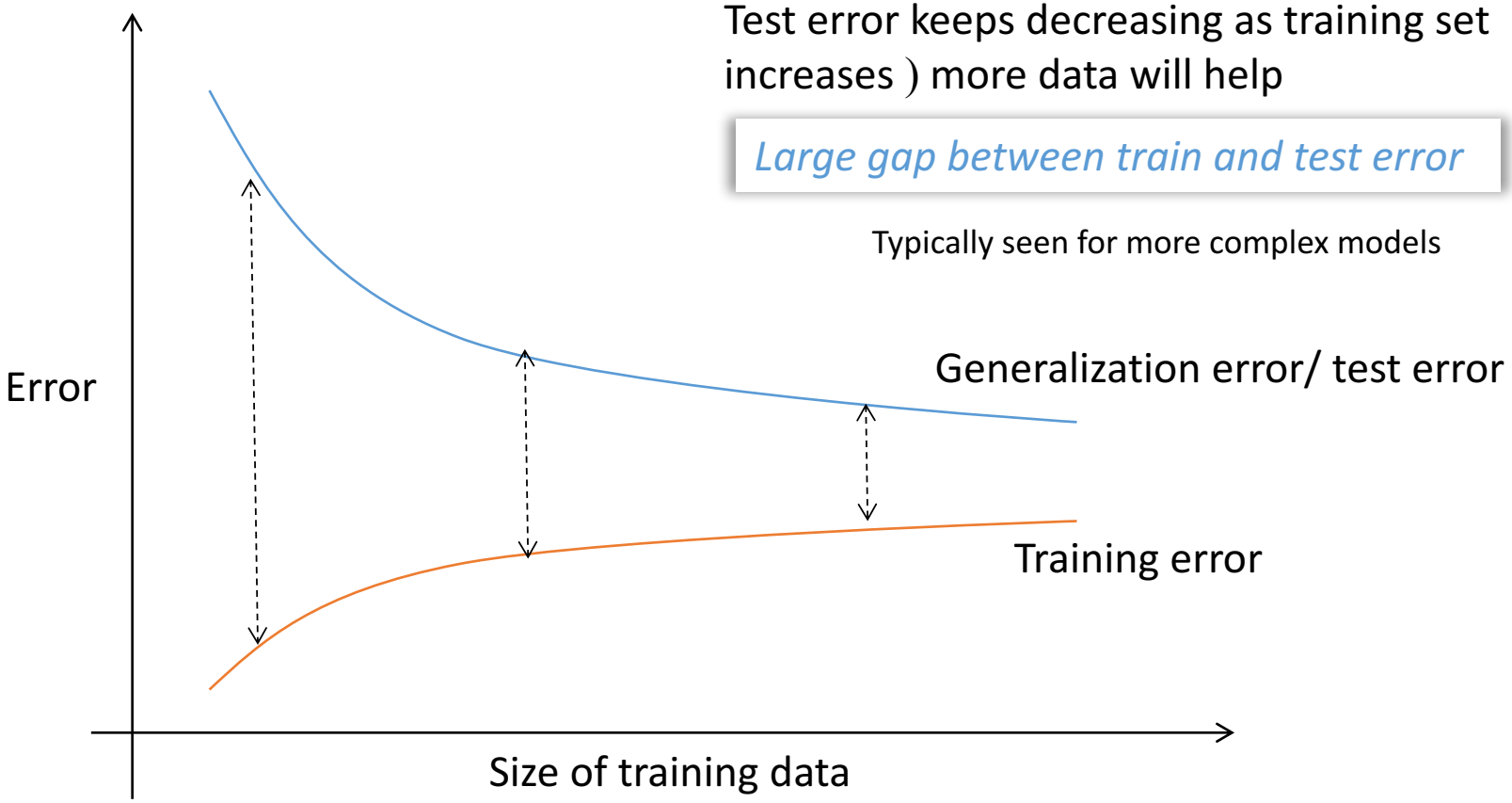


Detecting high variance using learning curves



Do we need more data?

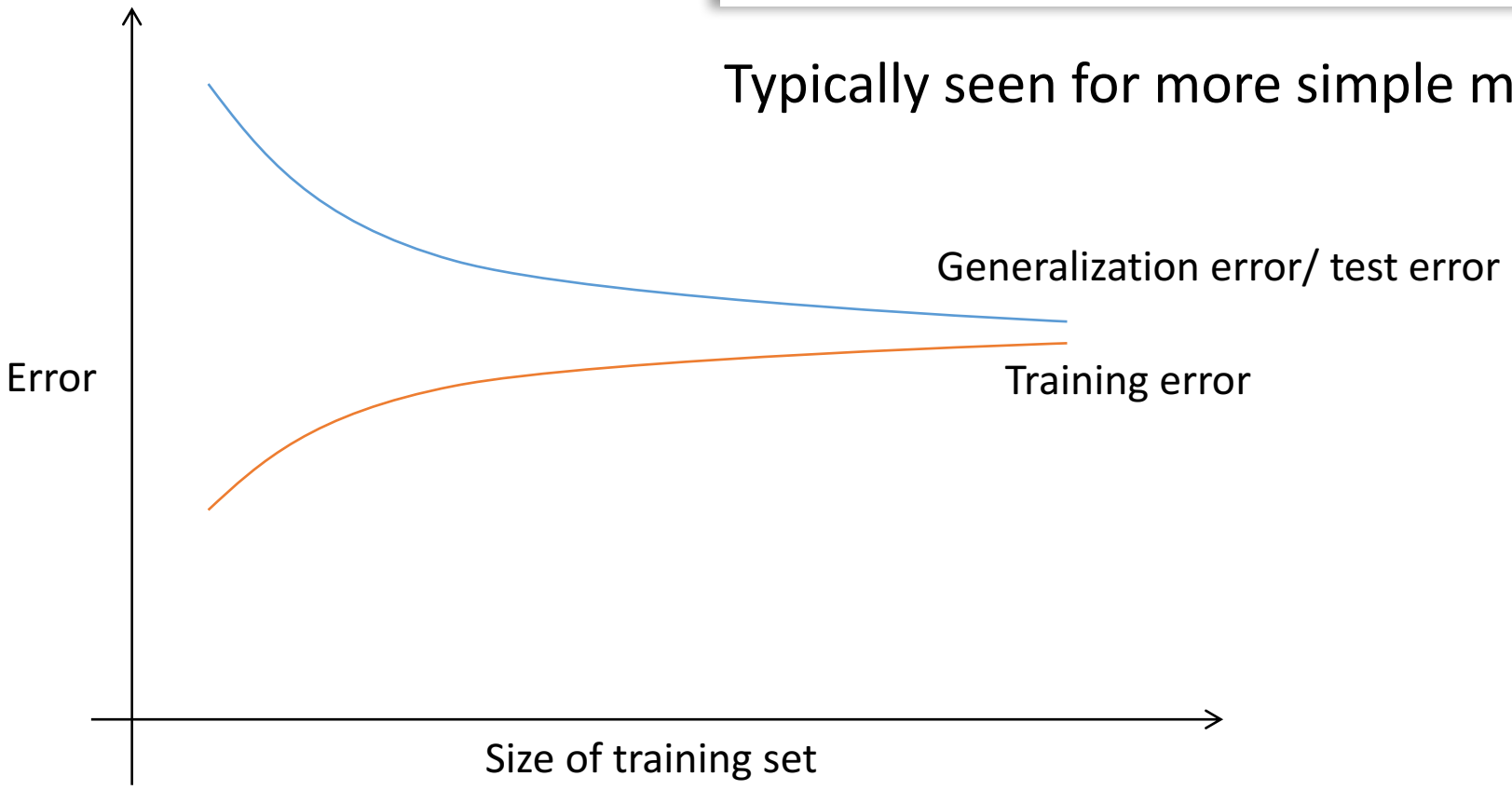
Detecting high variance using learning curves



Detecting high bias using learning curves

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

Typically seen for more simple models



Diagnostics

Easier to fix a problem if you know where it is

Some possible problems:

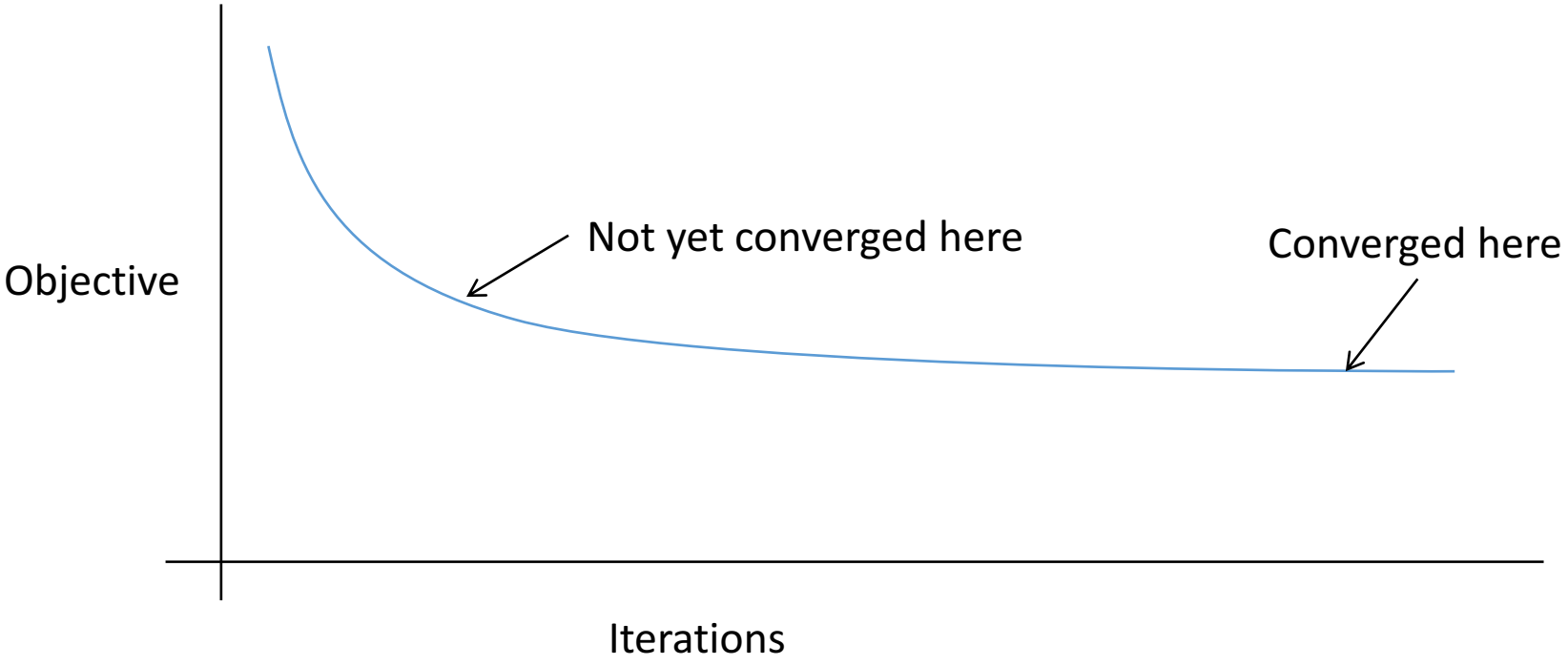
- ✓ Over-fitting (high variance)
- ✓ Under-fitting (high bias)

3. Your learning does not converge

4. Are you measuring the right thing?

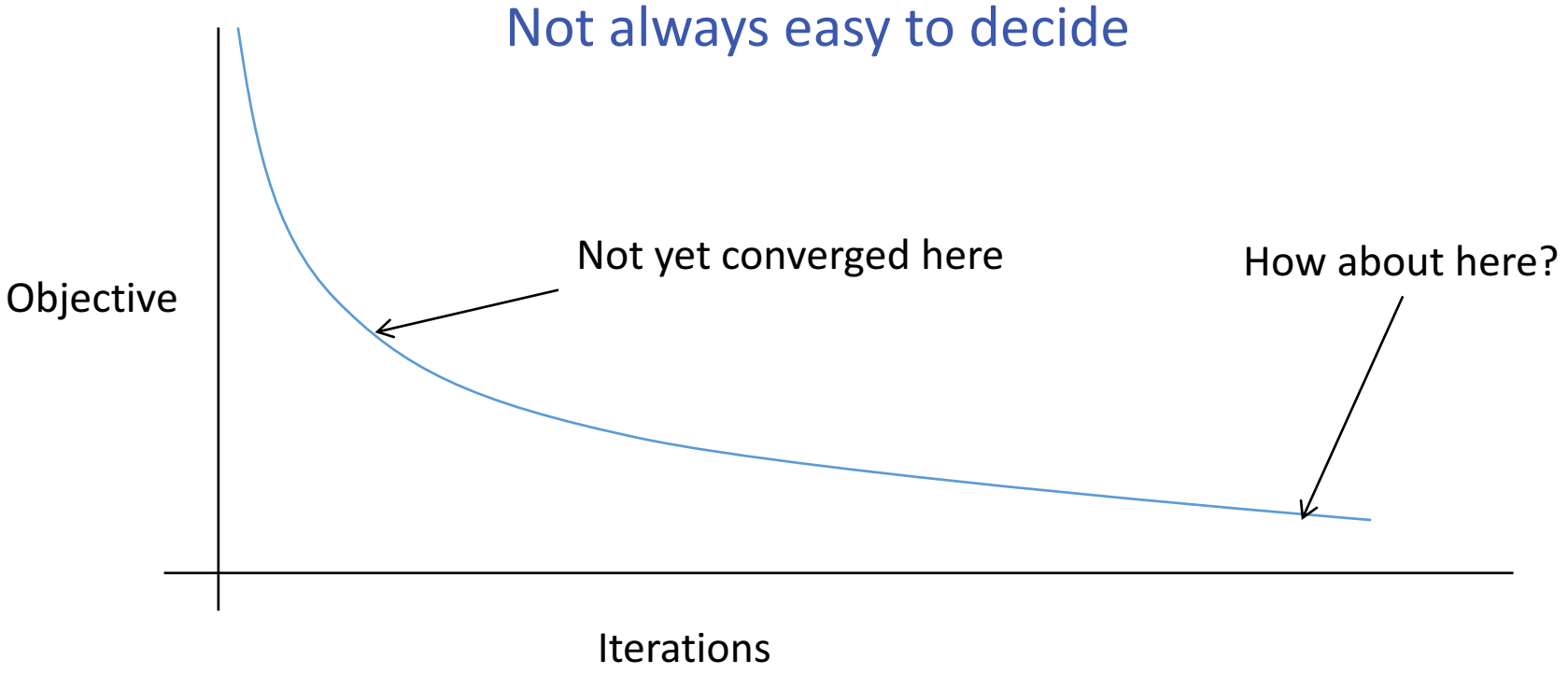
Does your learning algorithm converge?

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



Does your learning algorithm converge?

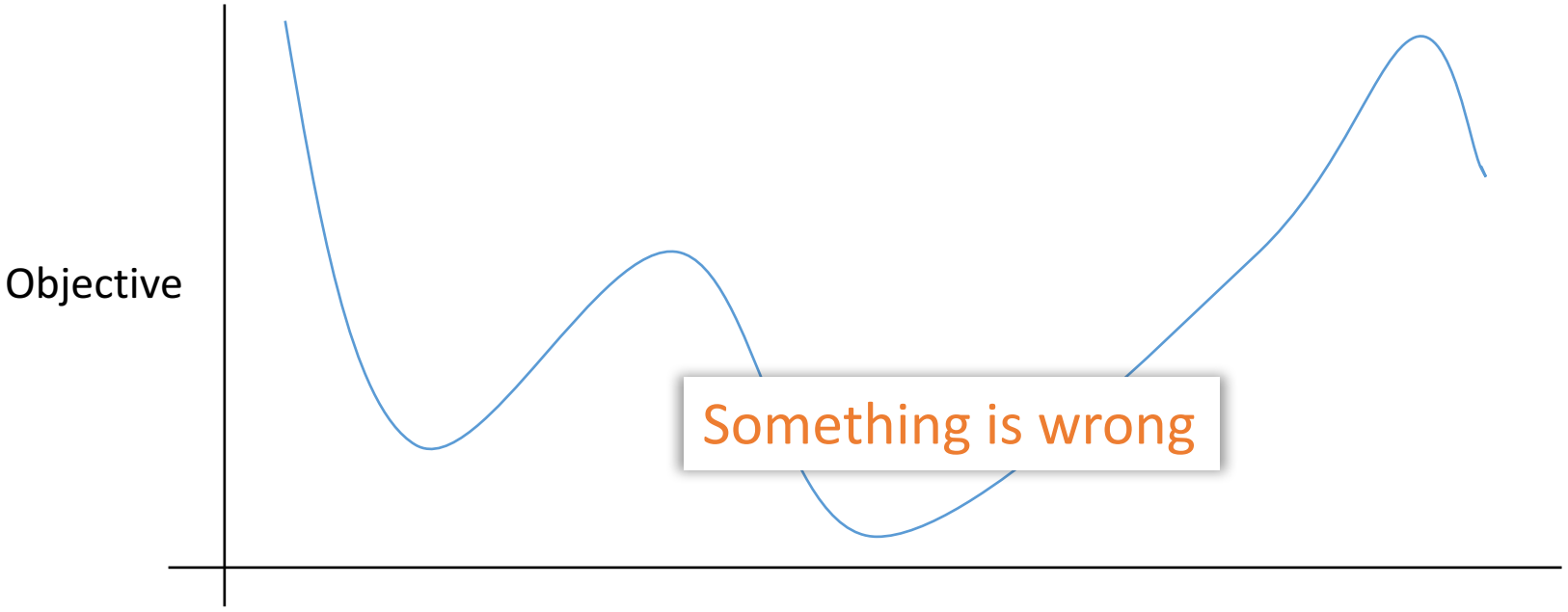
If learning is framed as an optimization problem, track the objective



Does your learning algorithm converge?

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

Check gradient!



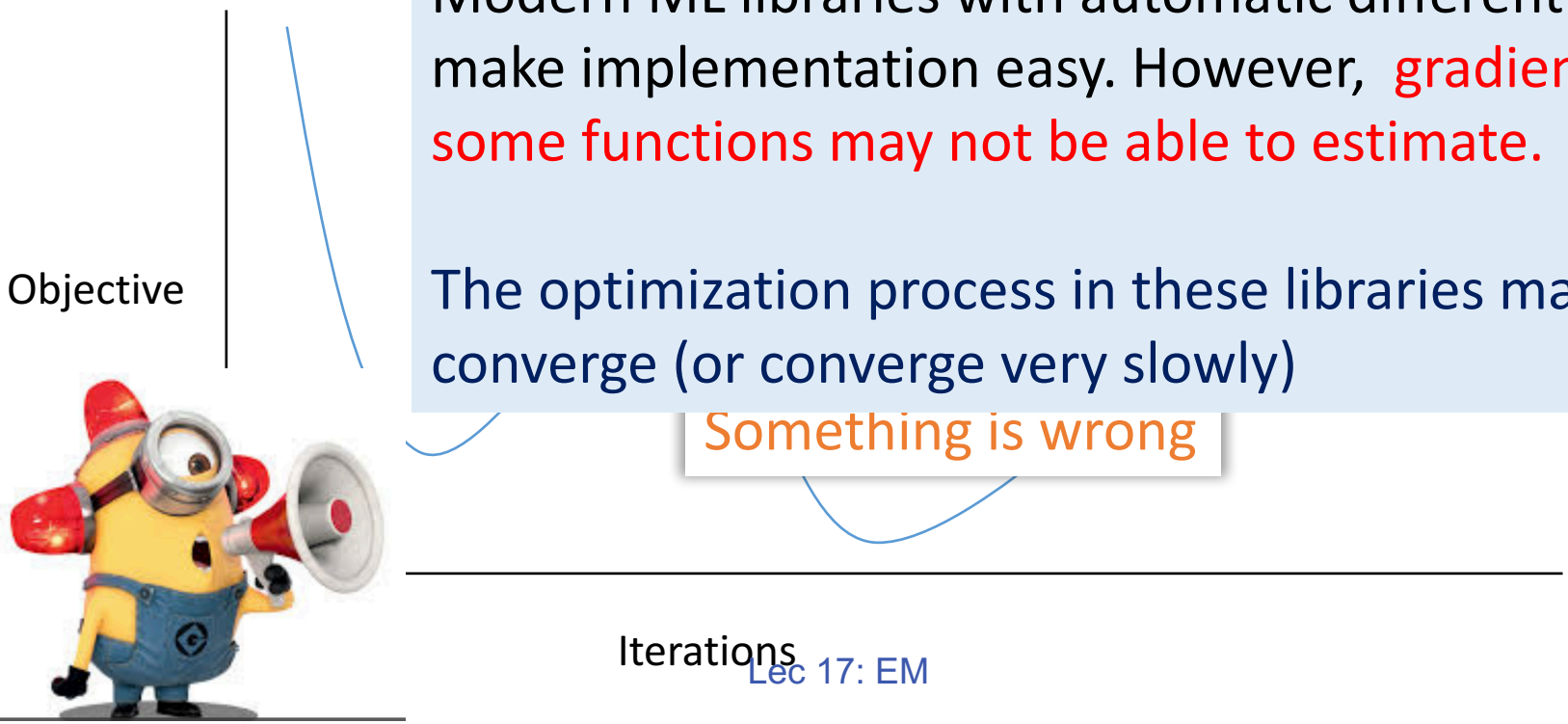
Does your learning algorithm converge?

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

Check gradient!

Modern ML libraries with automatic differentiation make implementation easy. However, gradient of some functions may not be able to estimate.

The optimization process in these libraries may not converge (or converge very slowly)



Diagnostics

Easier to fix a problem if you know where it is

Some possible problems:

- ✓ Over-fitting (high variance)
- ✓ Under-fitting (high bias)
- ✓ Your learning does not converge
- ❖ Are you measuring the right thing?

What to measure

- Accuracy of prediction is the most common measurement
- 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

ML and the world

- Bias vs Variance
- Diagnostics of your learning algorithm
- Error analysis
- Injecting machine learning into *Your Favorite Task*

Machine Learning in this class

Machine Learning in context

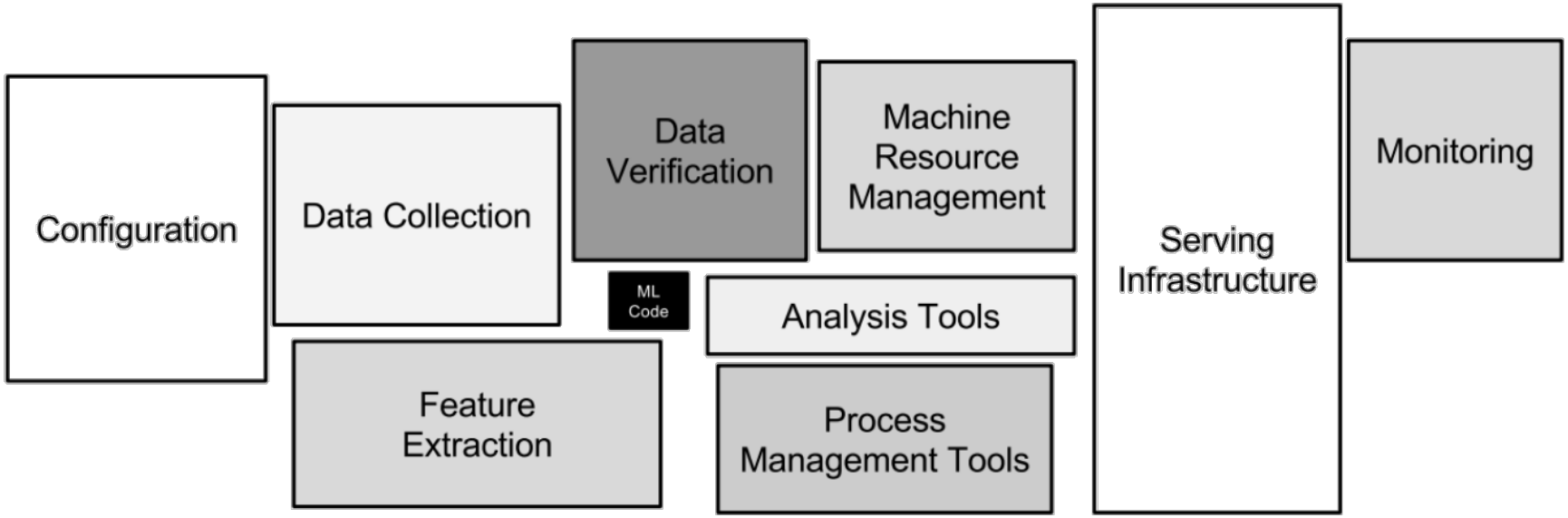


Figure from [Sculley, et al NIPS 2015]

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

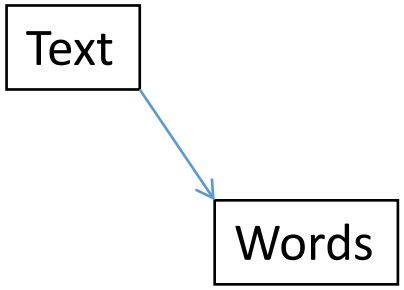
How much do each of these contribute to the error?

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

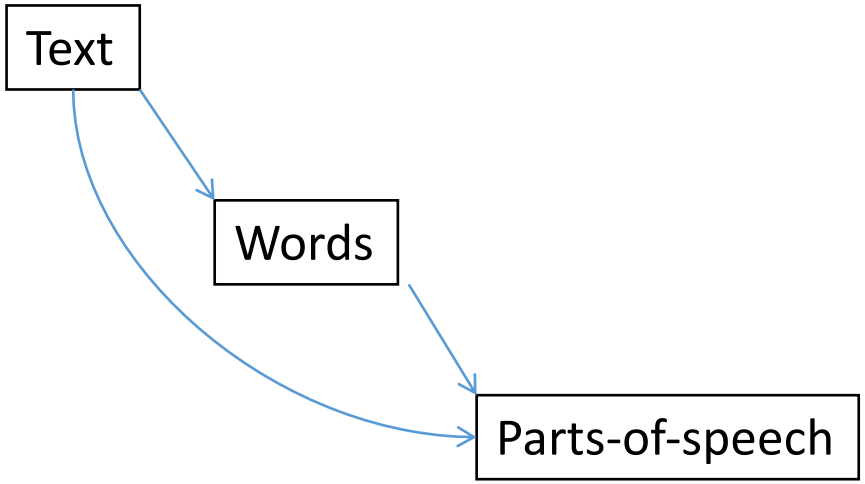
Example: A typical text processing pipeline

Example: A typical text processing pipeline

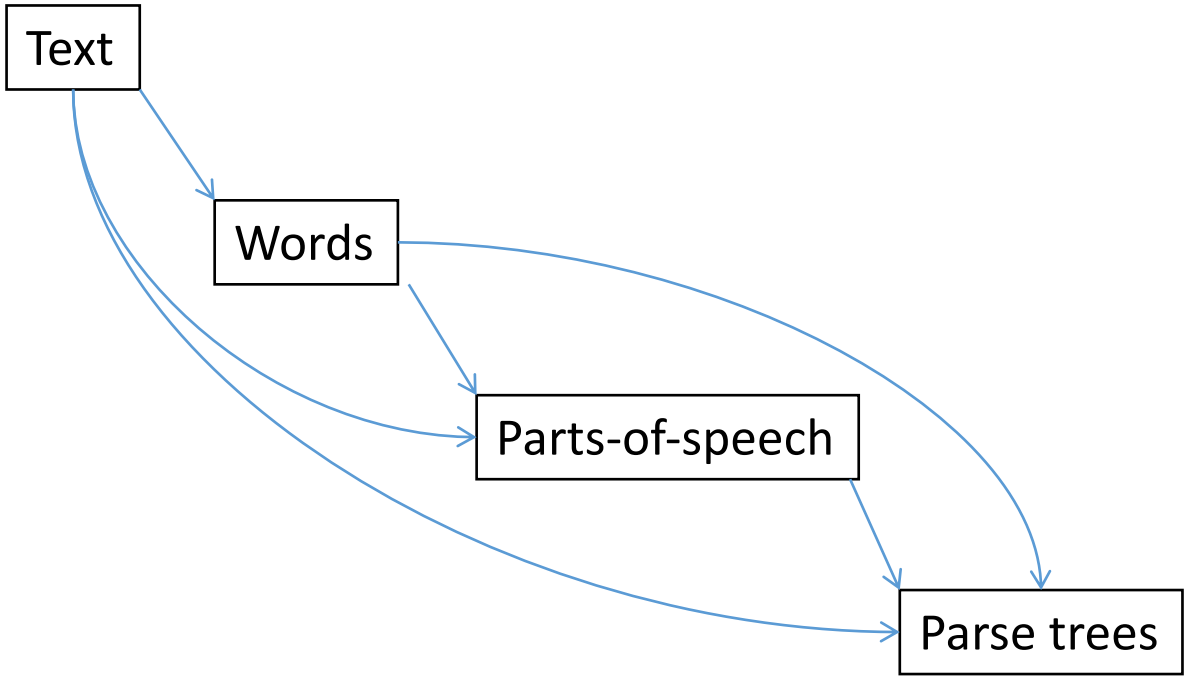
Example: A typical text processing pipeline



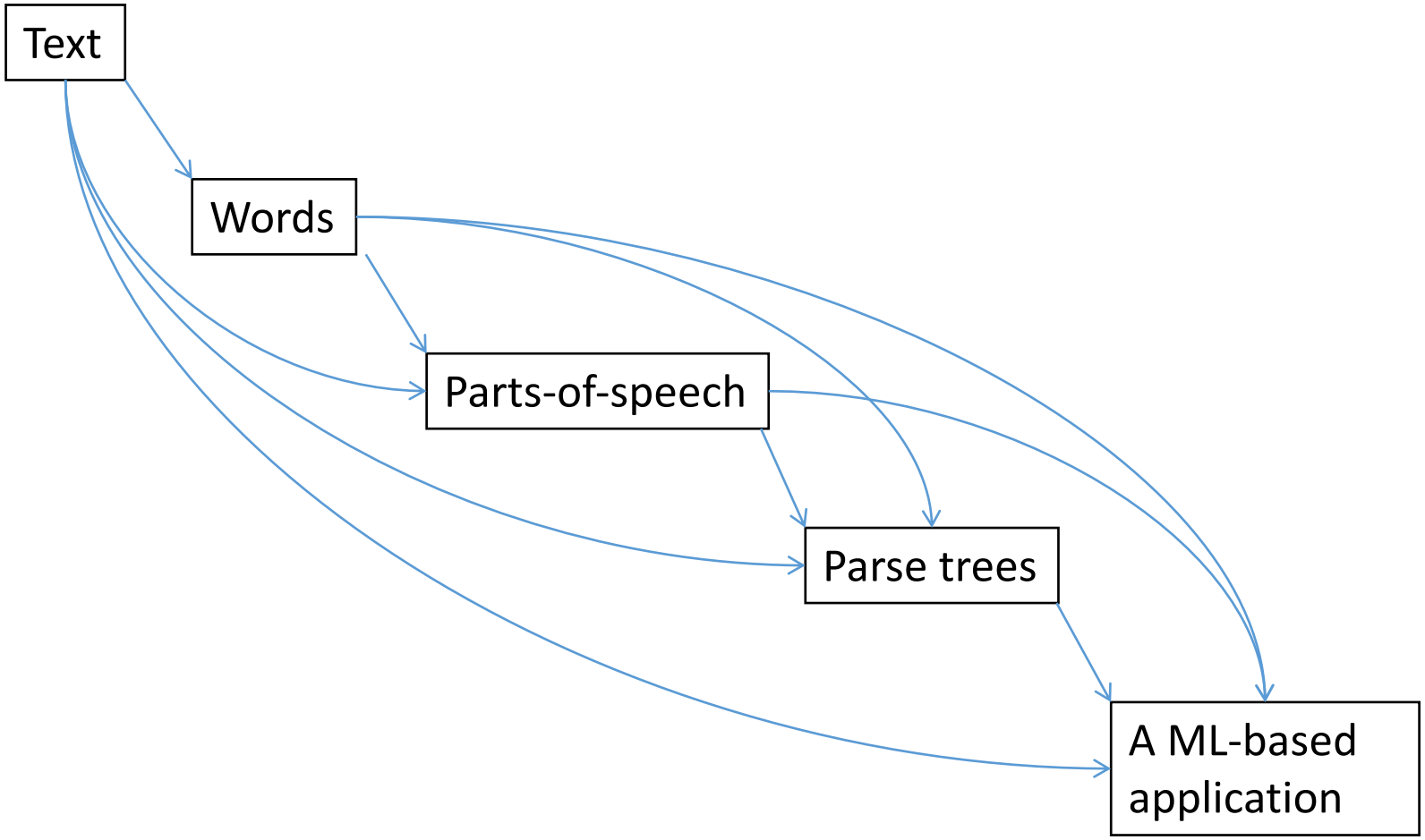
Example: A typical text processing pipeline



Example: A typical text processing pipeline



Example: A typical text processing pipeline

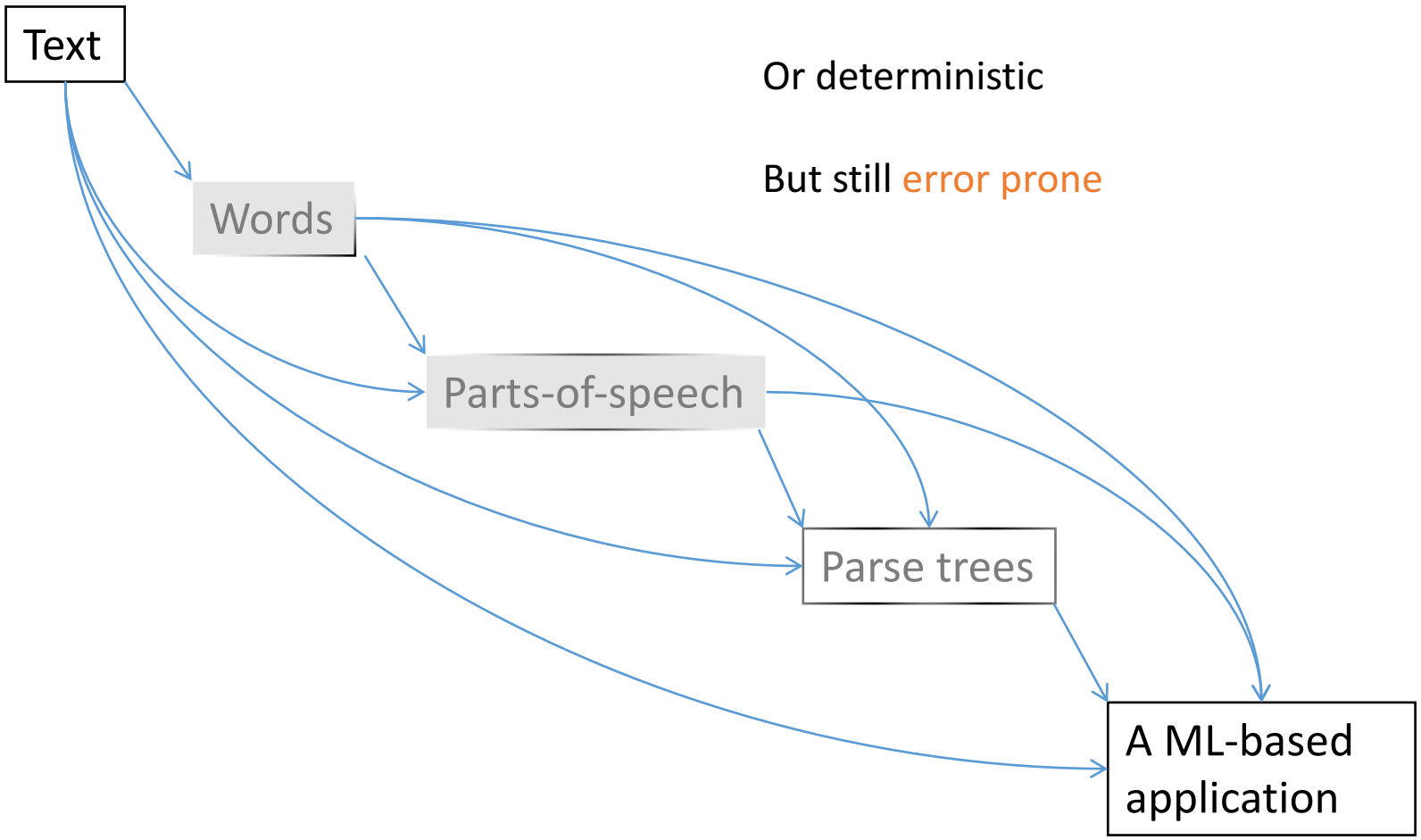


Example: A typical text processing pipeline

Each of these could be ML driven

Or deterministic

But still error prone

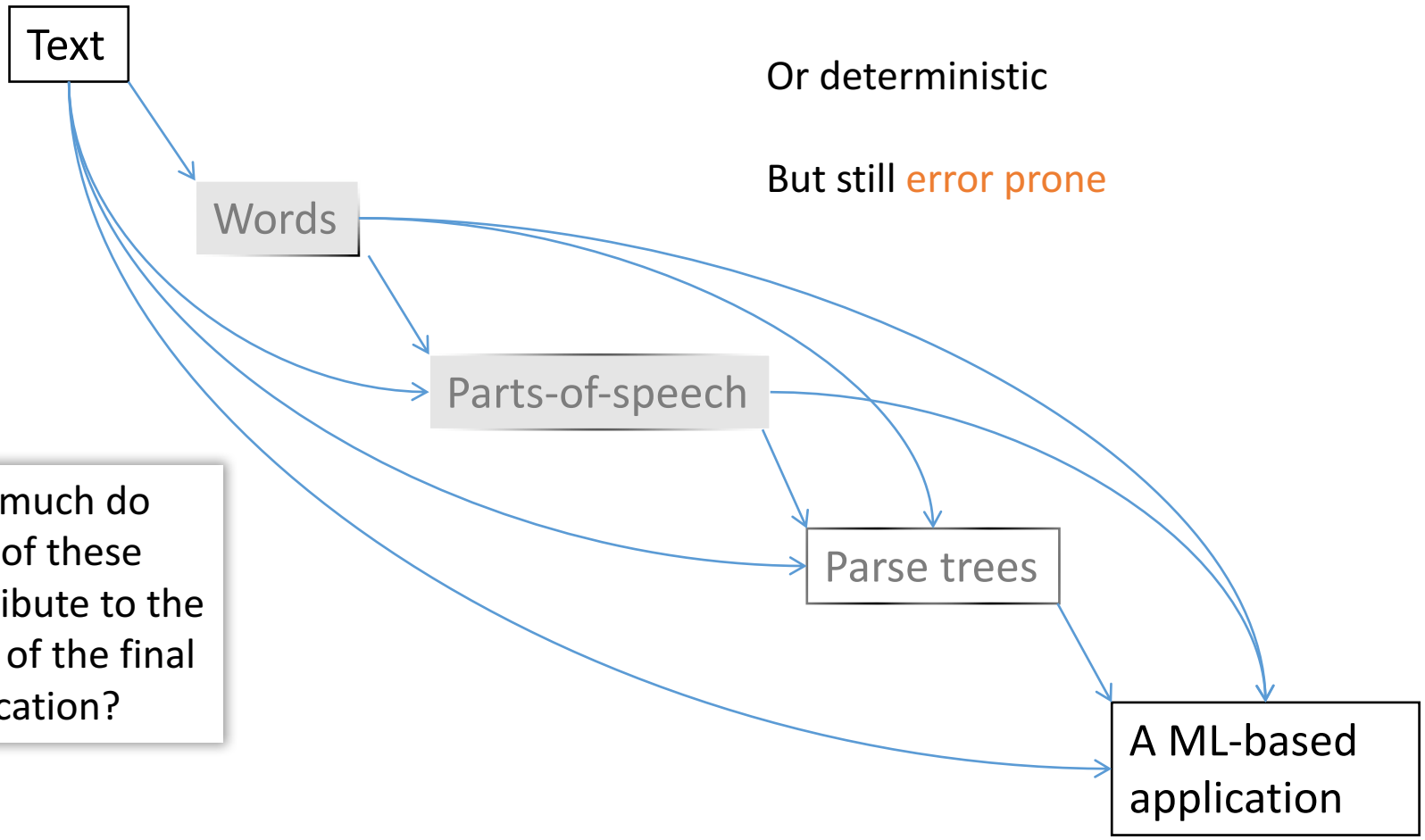


Example: A typical text processing pipeline

Each of these could be ML driven

Or deterministic

But still error prone

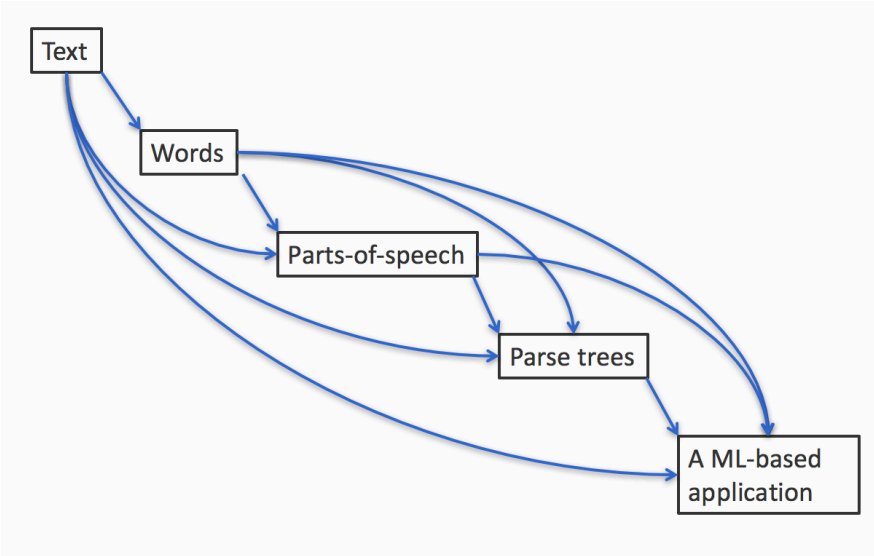


How much do each of these contribute to the error of the final application?

Tracking errors in a complex system

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

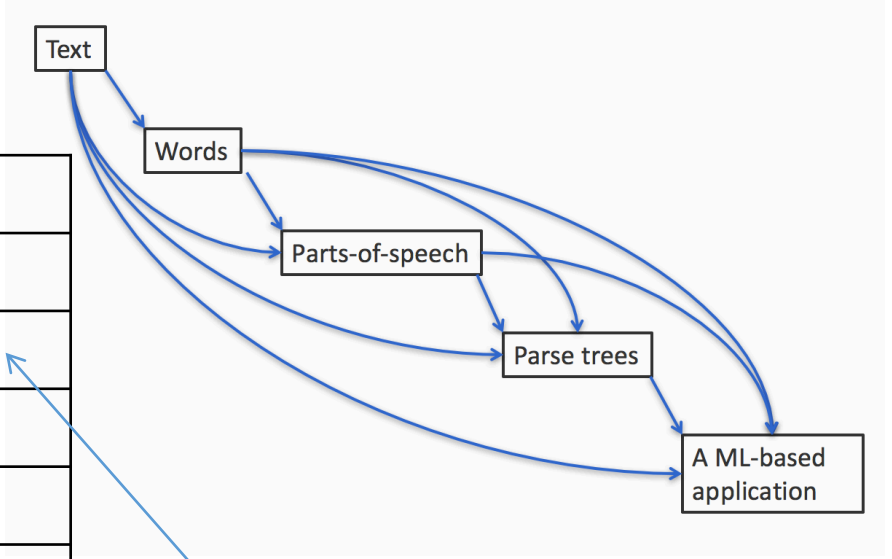
System	Accuracy
End-to-end predicted	55%
With ground truth words	60%
+ ground truth parts-of-speech	84 %
+ ground truth parse trees	89 %
+ ground truth final output	100 %



Tracking errors in a complex 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 in the part-of-speech component hurts the most

Ablative study

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

Usually seen with features

- 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

ML and the world

- Bias vs Variance
- Diagnostics of your learning algorithm
- Error analysis
- Injecting machine learning into *Your Favorite Task*

Big data is not enough

But more data is always better

- Cleaner data is even better

Remember that learning is impossible without some bias that simplifies the search

- Otherwise, no generalization

Learning requires knowledge to guide the learner

- *Machine learning is not a magic wand*

Miscellaneous advice

- 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

A retrospective look at the course

Learning = generalization

"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E."

Tom Mitchell (1999)

We saw different "models"

Or: what kind of a function should a learner learn

- Linear classifiers
- Decision trees
- Non-linear classifiers, feature transformations, neural networks
- Ensembles of classifiers

Different learning protocols

- Supervised learning
  - A *teacher* supplies a collection of examples with labels
  - The *learner* has to learn to label new examples using this data
- Unsupervised learning
  - No *teacher*, *learner* has only unlabeled examples
  - Data mining

- We did not see
  - Reinforcement learning
    - Learning from interaction
  - Semi-supervised learning
    - *Learner* has access to both labeled and unlabeled examples

The theory of machine learning

Mathematically defining learning

- Online learning
- Probably Approximately Correct (PAC) Learning
- Bayesian learning

What we saw

1. A broad theoretical and practical understanding of machine learning paradigms and algorithms

2. Ability to implement learning algorithms

3. Identify where machine learning can be applied and make the most appropriate decisions (about algorithms, models, supervision, etc)

Thank you

- Please log in MyUCLA for course survey