

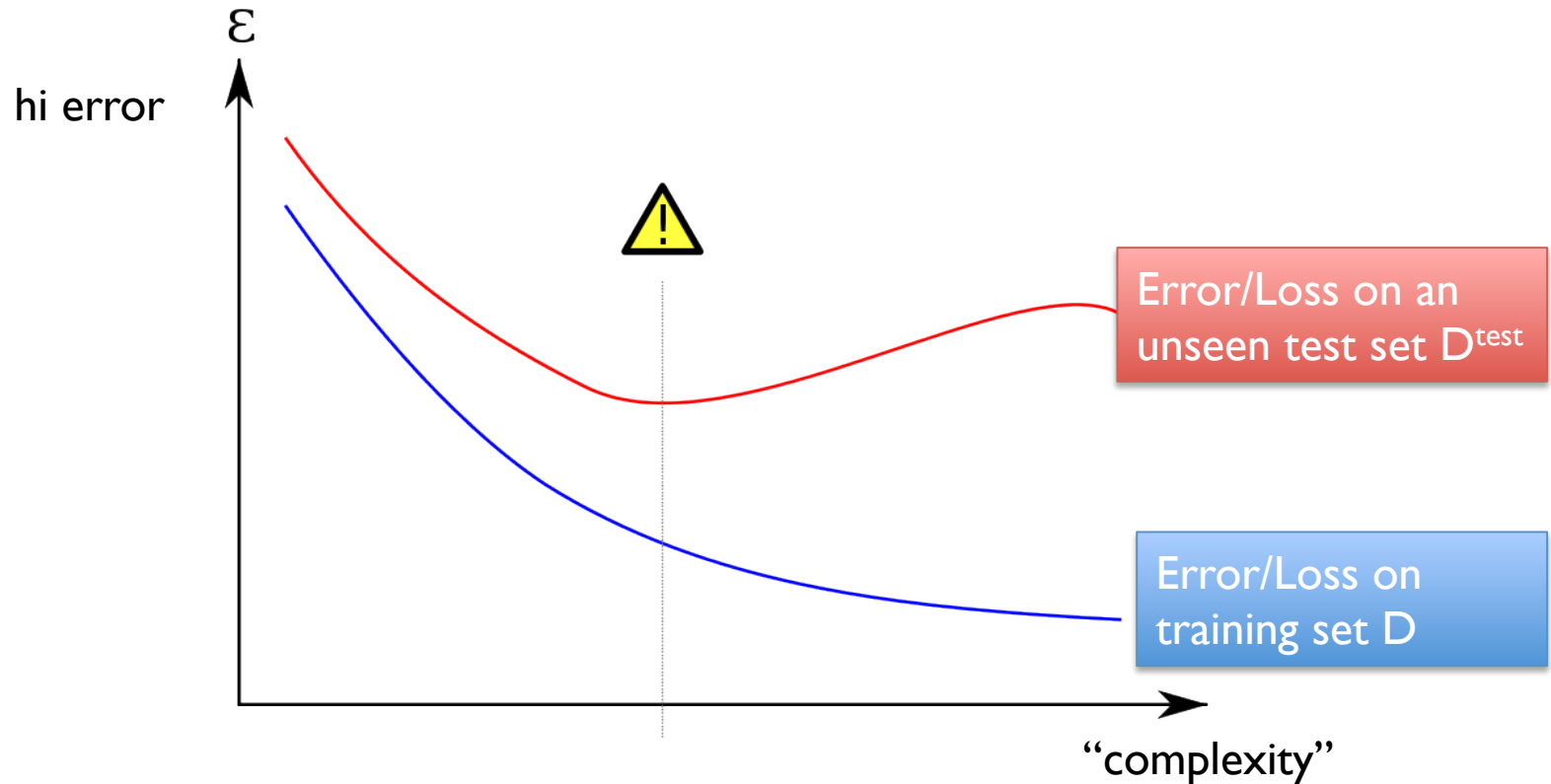
# **Experimentally Evaluating Classifiers**

William Cohen

# **PRACTICAL LESSONS IN COMPARING CLASSIFIERS**

# Learning method often “overfit”

- Overfitting is often a problem in supervised learning.
  - When you fit the data (minimize loss) are you fitting “real structure” in the data or “noise” in the data?
  - Will the patterns you see appear in a test set or not?



# Kaggle

<http://techtalks.tv/talks/machine-learning-competitions/58340/>

Quick summary:

- Kaggle runs ML competitions – you submit predictions, they score them on data where you see the *instances* but not the *labels*.
- Everyone sees the same test data and can tweak their algorithms to it
- After the competition closes there is usually one *more* round:
  - Participants are scored on a *fresh* dataset they *haven't* seen
  - Leaders often change....

# Why is this important?

- Point: If there's big data, you can just use error on a big test set, so confidence intervals will be small.
- Counterpoint: even with “big data” the size of the ideal test sets are often small
  - Eg: CTR data is biased

# Why is this important?

- Point: You can just use a “cookbook recipe” for your significance test
- Counterpoint: Surprisingly often you need to design your own test and/or make an intelligent choice about what test to use
  - New measures:
    - mention ceaf ?  $B^3$  ?
  - New settings:
    - new assumptions about what’s random (page/site)?
    - how often does a particular type of error happen?  
how confident can we be that it’s been reduced?

**CONFIDENCE INTERVALS ON THE  
ERROR RATE ESTIMATED BY A SAMPLE:  
PART I, THE MAIN IDEA**

# A practical problem

- You've just trained a classifier  $h$  using YFCL\* on YFP\*\*. You tested  $h$  on a sample  $S$  and the error rate was 0.30.
  - How good is that estimate?
  - Should you throw away the old classifier, which had an error rate of 0.35, and replace it with  $h$ ?
  - Can you write a paper saying you've reduced the **best known** error rate for YFP from 0.35 to 0.30?
    - Would it be accepted?

---

\*YFCL = Your Favorite Classifier Learner

\*\*YFP = Your Favorite Problem

# Two definitions of error

- The **true error** of  $h$  with respect to target function  $f$  and distribution  $D$  is the probability that  $h$  will misclassify an instance drawn at random from  $D$ :

$$error_D(h) \equiv \Pr_{x \in D} [f(x) \neq h(x)]$$

- The **sample error** of  $h$  with respect to target function  $f$  and sample  $S$  is the fraction of instances in  $S$  that  $h$  misclassifies:

$$error_S(h) \equiv \frac{1}{|S|} \sum_{x \in S} \delta[f(x) \neq h(x)]$$

$$\text{where } \delta[f(x) \neq h(x)] = \begin{cases} 1 & \text{if } f(x) \neq h(x) \\ 0 & \text{else} \end{cases}$$

# Two definitions of error

- The **true error** of  $h$  with respect to target function  $f$  and distribution  $D$  is the probability that  $h$  will misclassify an instance drawn at random from  $D$ :

$$error_D(h) \equiv \Pr_{x \in D} [f(x) \neq h(x)]$$

- The **sample error** of  $h$  with respect to target function  $f$  and sample  $S$  is the fraction of instances in  $S$  that  $h$  misclassifies:

$$error_S(h) \equiv \frac{1}{|S|} \sum_{x \in S} \delta[f(x) \neq h(x)]$$

Usually  $error_D(h)$  is unknown and we use an estimate  $error_S(h)$ . **How good** is this estimate?

# Why sample error is wrong

- *Bias*: if  $S$  is the training set, then  $error_S(h)$  is **optimistically biased**: i.e.,

$$error_S(h) < error_D(h)$$

$$Bias \equiv E[error_S(h) - error_D(h)]$$

- This is true if  $S$  was used at any stage of learning: feature engineering, parameter testing, feature selection, ...
  - You want  $S$  and  $h$  to be *independent*
- A popular split is *train, development, and evaluation*

# Why sample error is wrong

- *Bias*: if  $S$  is independent from the training set, and drawn from  $D$ , then the estimate is “unbiased”:

$$\text{Bias} \equiv E[\text{error}_S(h) - \text{error}_D(h)] = 0$$

- *Variance*: but even if  $S$  is independent, the  $\text{error}_S(h)$  may still vary from  $\text{error}_D(h)$ :

$$\text{Var} \equiv E\left[\left(\text{error}_S(h) - E[\text{error}_S(h)]\right)^2\right]$$

# A simple example

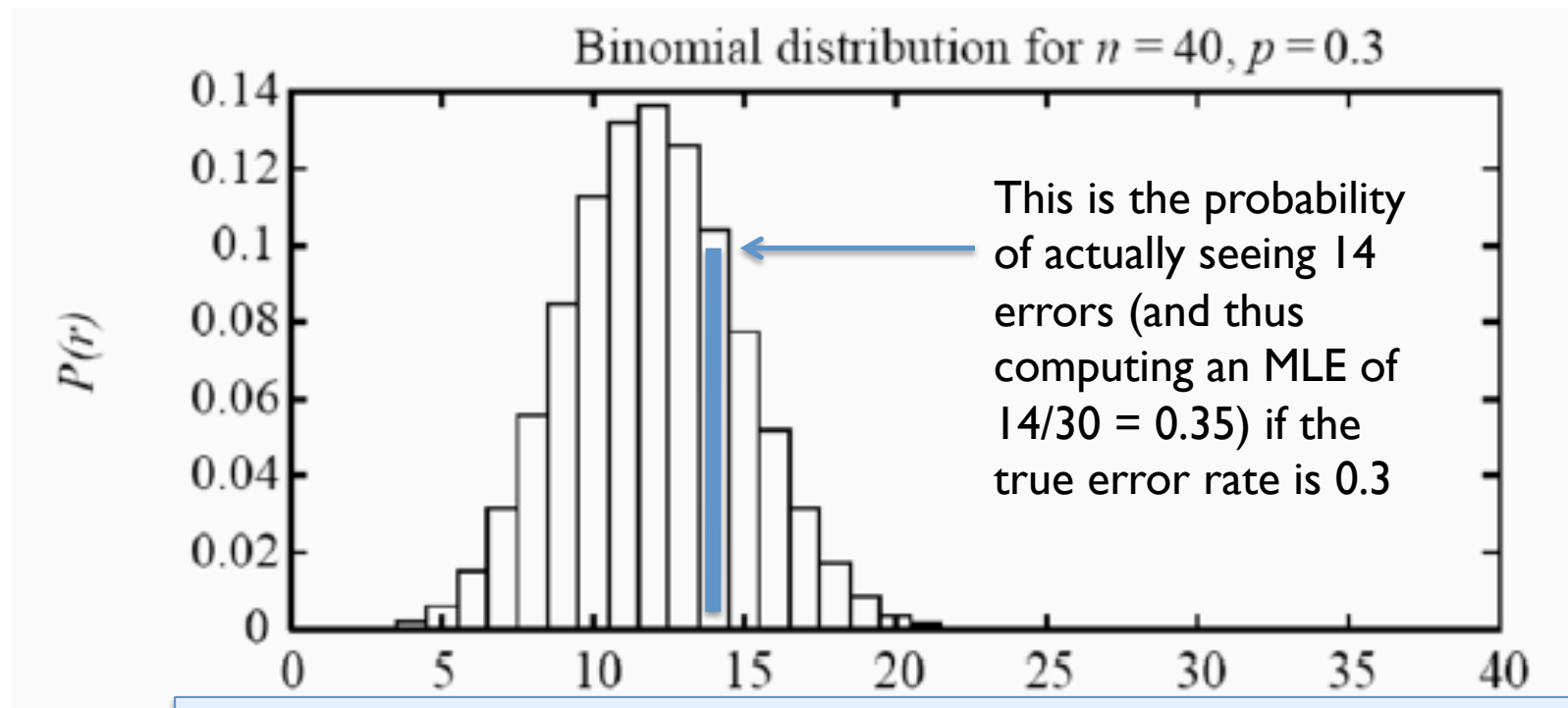
- Hypothesis  $h$  misclassifies 12 of 40 examples from  $S$ .
- So:  $error_S(h) = 12/40 = 0.30$
- What is  $error_D(h)$ ?
  - Is it less than 0.35?

# A simple example

- Hypothesis  $h$  misclassifies 12 of 40 examples from  $S$ .
  - So:  $error_S(h) = 12/40 = 0.30$
  - What is  $error_D(h)$  ?  $error_S(h)$
- The event “ $h$  makes an error on  $\mathbf{x}$ ” is a random variable (over examples  $X$  from  $D$ )
  - In fact, it's a binomial with parameter  $\theta$
  - With  $r$  error in  $n$  trials, MLE of  $\theta$  is  $r/n = 0.30$ .
  - Note that  $\theta = error_D(h)$  by definition

# A simple example

In fact for a binomial we know the whole pmf (probability mass function):



With 40 examples estimated errors of 0.35 vs 0.30 seem pretty close...

$$P(r) = \frac{n!}{r!(n-r)!} \text{error}_{\mathcal{D}}(h)^r (1 - \text{error}_{\mathcal{D}}(h))^{n-r}$$

# Aside: credibility intervals

What we *have* is:

$$\Pr(R=r \mid \Theta=\theta)$$

Arguably what we *want* is:

$$\Pr(\Theta=\theta \mid R=r) = (1/Z) \Pr(R=r \mid \Theta=\theta) \Pr(\Theta=\theta)$$

which would give us a MAP for  $\theta$ , or an interval that probably contains  $\theta$ ....

This isn't common practice

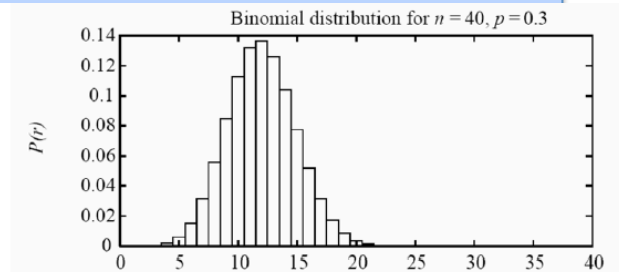
# A simple example

To pick a confidence interval we need to clarify what's random and what's not

Commonly

- $h$  and  $\text{error}_D(h)$  are fixed but unknown
- $S$  is random variable
  - *sampling is the experiment*
- $R = \text{error}_S(h)$  is a random variable
  - depending on  $S$

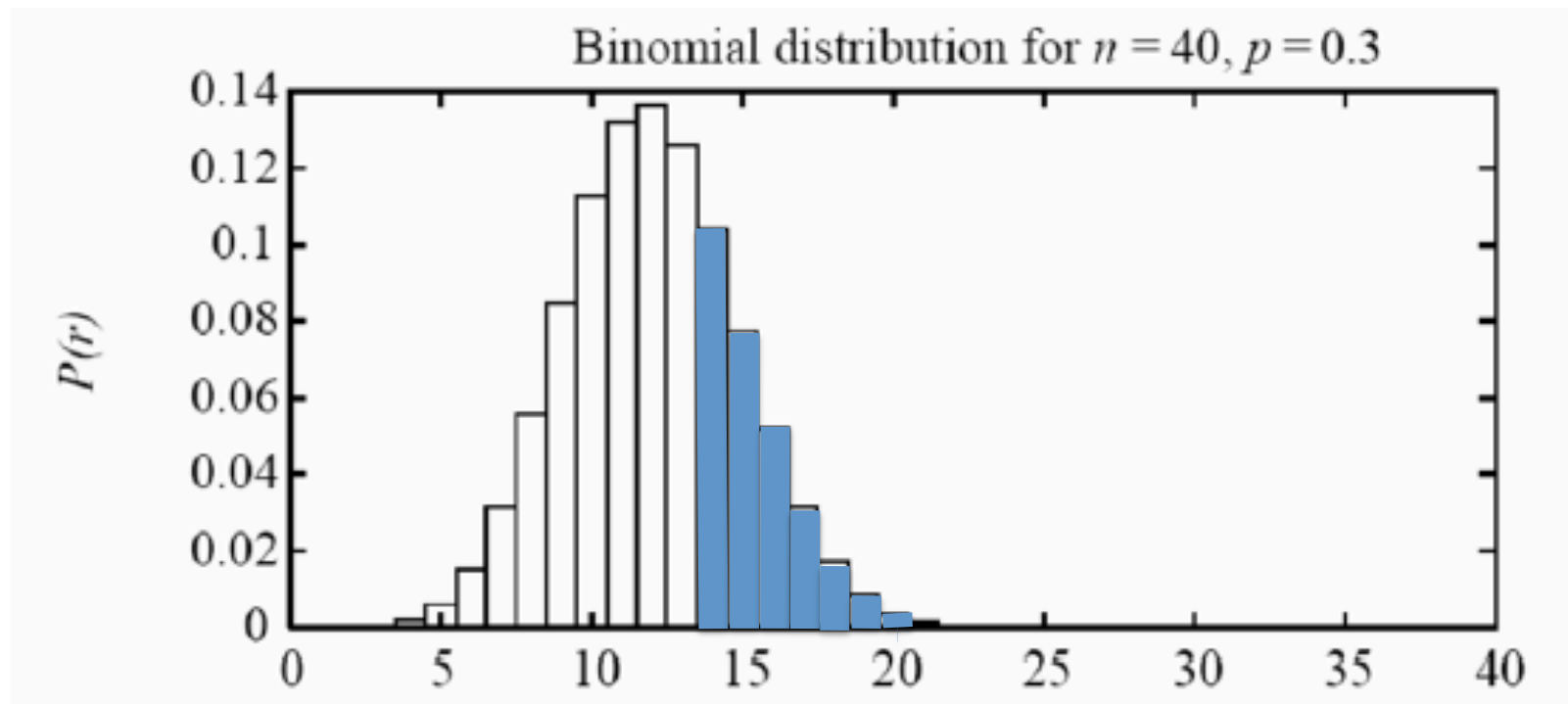
We ask: what other outcomes of the experiment are likely?



$$P(r) = \frac{n!}{r!(n-r)!} \text{error}_D(h)^r (1 - \text{error}_D(h))^{n-r}$$

# A simple example

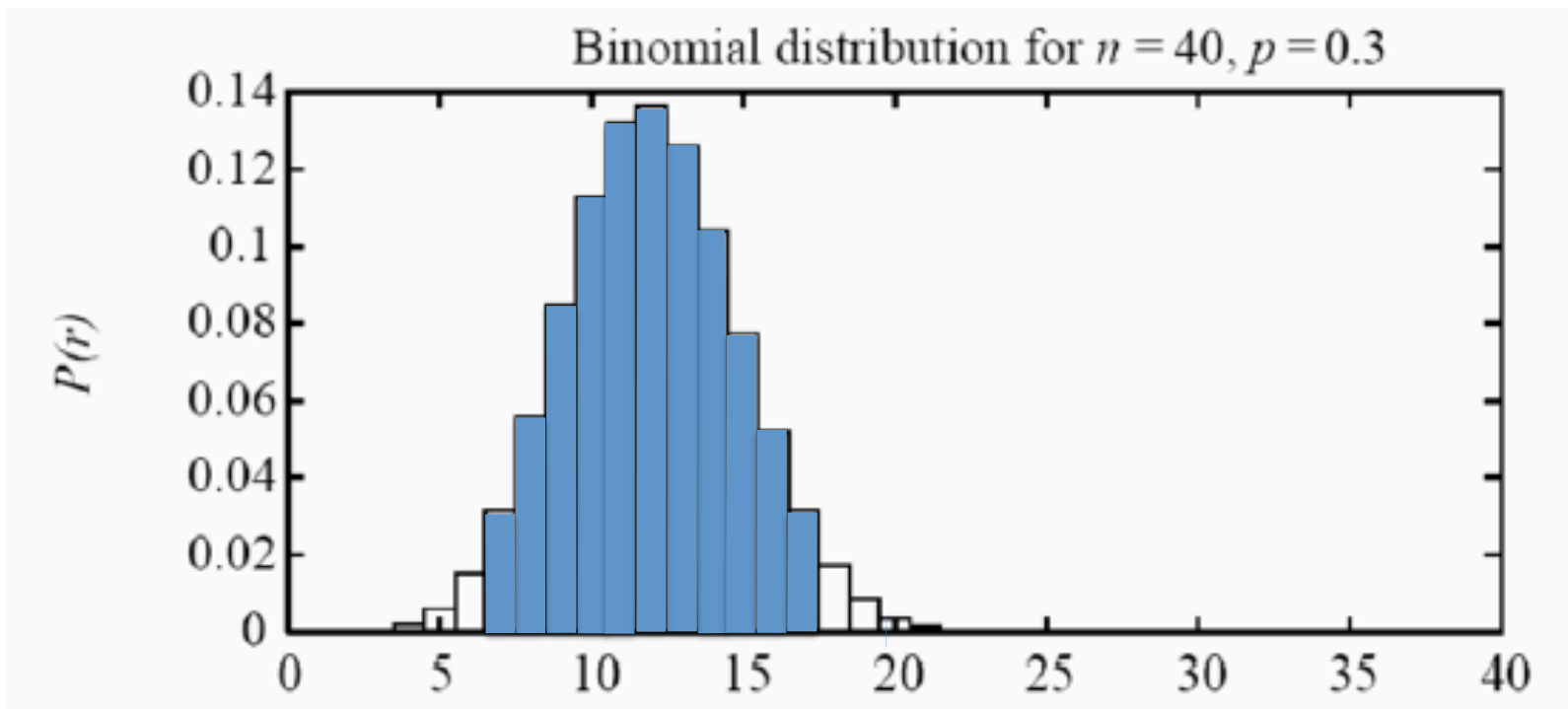
Is  $\theta < 0.35$  ( $= 14/40$ )?



Given this estimate of  $\theta$ , the probability of a *sample*  $S$  that would make me think that  $\theta \geq 0.35$  is fairly high ( $> 0.1$ )

# A simple example

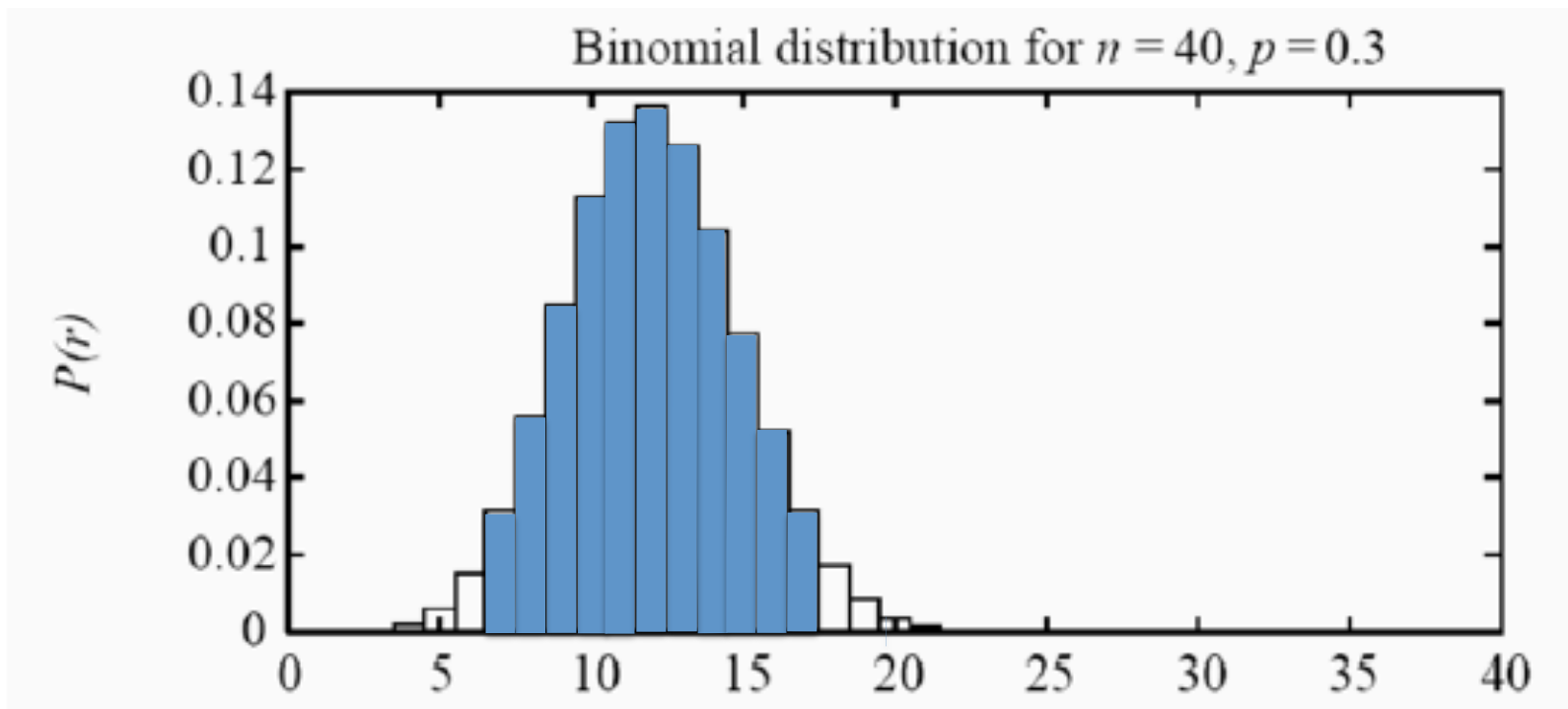
I can pick a range of  $\theta$  such that the probability of a sample that would lead to an estimate outside the range is low



Given my estimate of  $\theta$ , the probability of a sample with fewer than 6 errors or more than 16 is low (say  $<0.05$ ).

# A simple example

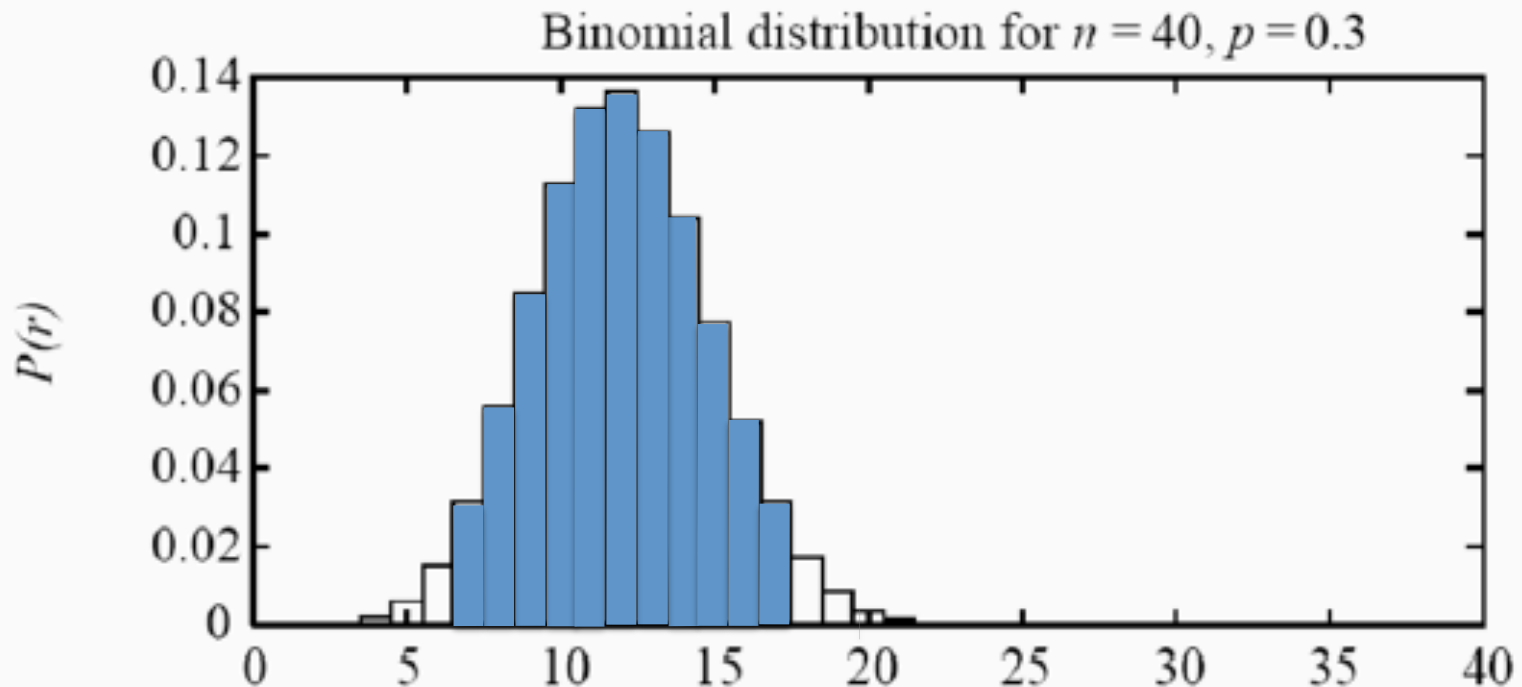
If that's true, then  $[6/40, 16/40]$  is a 95% **confidence interval** for  $\theta$



Given my estimate of  $\theta$ , the probability of a sample with fewer than 6 errors or more than 16 is low (say  $<0.05$ ).

# A simple example

You might want to formulate a null hypothesis: eg, “the error rate is 0.35 or more”. You’d reject the null if the null outcome is *outside* the confidence interval.



We don't know the true error rate, but anything between  $6/40 = 15\%$  and  $16/40 = 40\%$  is plausible value given the data.

# Confidence intervals

- You now know how to compute a confidence interval.
  - You'd want a computer to do it, because computing the binomial exactly is a chore.
  - If you have enough data, then there are some simpler approximations.

**CONFIDENCE INTERVALS ON THE  
ERROR RATE ESTIMATED BY A SAMPLE:  
PART 2, COMMON APPROXIMATIONS**

# Recipe 1 for *confidence intervals*

- If
  - $|S|=n$  and  $n > 30$
  - All the samples in  $S$  are drawn independently of  $h$  and each other
  - $error_S(h) = p$
- Then with 95% probability,  $error_D(h)$  is in

Another rule of thumb: it's safe to use this approximation when the interval is within  $[0,1]$

$$p \pm 1.96 \sqrt{\frac{p(1-p)}{n}}$$

## Recipe 2 for *confidence intervals*

- If
  - $|S|=n$  and  $n>30$
  - All the samples in  $S$  are drawn independently of  $h$  and each other
  - $error_S(h)=p$
- Then with  $N\%$  probability,  $error_D(h)$  is in


$$p \pm z_n \sqrt{\frac{p(1-p)}{n}}$$

- For these values of  $N$ :


$N\%$ :	50%	68%	80%	90%	95%	98%	99%
$z_N$ :	0.67	1.00	1.28	1.64	1.96	2.33	2.58

# Why do these recipes work?

- Binomial distribution for  $R = \# \text{ heads in } n \text{ flips}$ , with  $p = \Pr(\text{heads})$ 
  - Expected value of  $R$ :  $E[R] = np$
  - Variance of  $R$ :  $\text{Var}[R] = E[R - E[R]] = np(1-p)$
  - Standard deviation of  $R$ :  $\sigma_R = \sqrt{np(1-p)}$
  - Standard error:  $SE_R = \sigma_R / \sqrt{n}$



SE = expected distance  
between a **sample mean** for  
a size- $n$  sample and  $E[X]$



SD = expected distance  
between a **single sample** of  $X$  and  $E[X]$

# Why do these recipes work?

- Binomial distribution for  $R = \#$  heads in  $n$  flips, with  $p = \Pr(\text{heads})$ 
  - Expected value of  $R$ :  $E[R] = np$
  - Variance of  $R$ :  $\text{Var}[R] = E[R - E[R]]^2 = np(1-p)$
  - Standard deviation of  $R$ :  $\sigma_R = \sqrt{np(1-p)}$
  - Standard error:  $SE_R = \sigma_R / \sqrt{n}$

$$p \pm 1.96 \sqrt{\frac{p(1-p)}{n}}$$

$$p \pm 1.96 \cdot SE_{R/n}$$

# Why do these recipes work?

- So:
  - $E[error_S(h)] = error_D(h)$
  - standard **deviation** of  $error_S(h)$  = standard **error** of averaging  $n$  draws from a binomial with parameter  $p$ , or

$$\sqrt{\frac{p(1-p)}{n}} \approx \sqrt{\frac{error_S(h)(1-error_S(h))}{|S|}}$$

- For large  $n$  the binomial mean approximates a normal distribution with same mean and sd

# Why do these recipes work?

## Central Limit Theorem

Consider a set of independent, identically distributed random variables  $Y_1 \dots Y_n$ , all governed by an arbitrary probability distribution with mean  $\mu$  and finite variance  $\sigma^2$ . Define the sample mean,

$$\bar{Y} \equiv \frac{1}{n} \sum_{i=1}^n Y_i$$

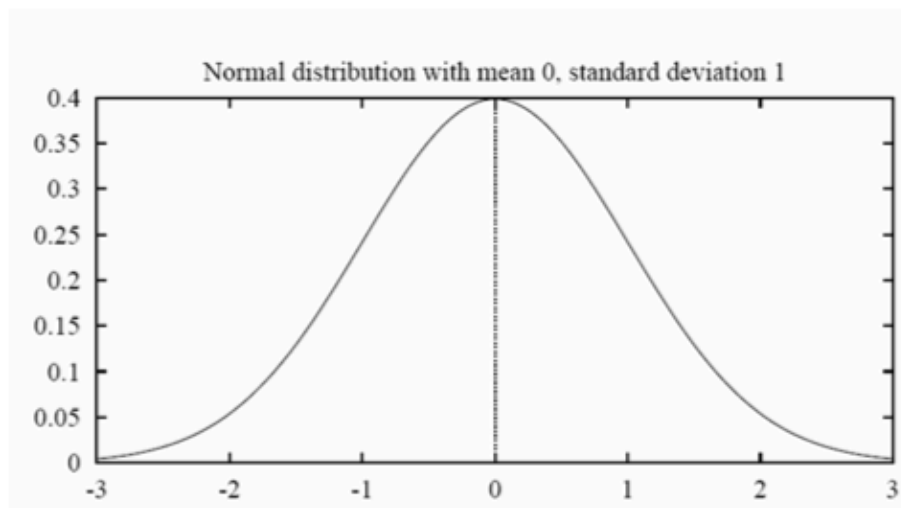
**Central Limit Theorem.** As  $n \rightarrow \infty$ , the distribution governing  $\bar{Y}$  approaches a Normal distribution, with mean  $\mu$  and variance  $\frac{\sigma^2}{n}$ .

Rule of thumb is considering “large”  $n$  to be  $n > 30$ .

Notice that the standard deviation for  $Y$  is  $\sigma$  but the standard deviation for  $\bar{Y}$  is  $\frac{\sigma}{\sqrt{n}}$  (aka the *standard error of the mean*)

# Why do these recipes work?

## Fact about the Normal Distribution

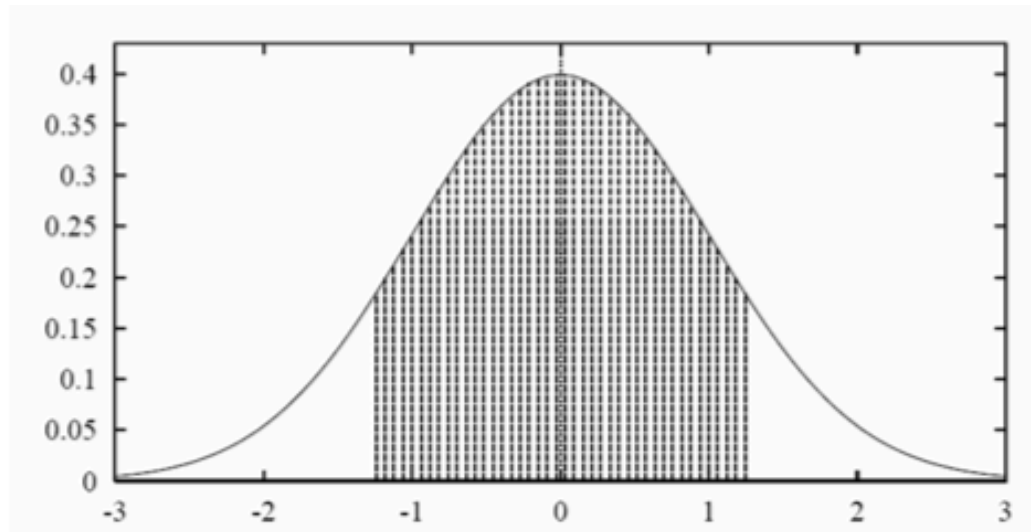


$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$$

The probability that  $X$  will fall into the interval  $(a, b)$  is given by  $\int_a^b p(x)dx$

- Expected, or mean value of  $X$ ,  $E[X]$ , is  $E[X] = \mu$
- Variance of  $X$  is  $Var(X) = \sigma^2$
- Standard deviation of  $X$ ,  $\sigma_X$ , is  $\sigma_X = \sigma$

# Why do these recipes work?



80% of area (probability) lies in  $\mu \pm 1.28\sigma$

N% of area (probability) lies in  $\mu \pm z_N\sigma$

N%:	50%	68%	80%	90%	95%	98%	99%
$z_N$ :	0.67	1.00	1.28	1.64	1.96	2.33	2.58

# Why recipe 2 works

By CLT, we're expecting the normal to be a good approximation for large  $n$  ( $n > 30$ )

- If
  - $|S|=n$  and  $n > 30$
  - All the samples in  $S$  are drawn independently of  $h$  and each other
  - $error_S(h) = p$
- Then with  $N\%$  probability,  $error_D(h)$  is in

$$p \pm z_n \sqrt{\frac{p(1-p)}{n}}$$

$z_n * SE(error_S(h))$

- For these values of  $N$  - taken from table for a normal distribution

$N\%$ :	50%	68%	80%	90%	95%	98%	99%
$z_N$ :	0.67	1.00	1.28	1.64	1.96	2.33	2.58

# Importance of confidence intervals

- This is a *subroutine*
- We'll use it for almost every other test

# **PAIRED TESTS FOR CLASSIFIERS**

# Comparing two learning systems

- Very common problem
  - You run YFCL and MFCL on a problem
    - get  $h_1$  and  $h_2$
    - test on  $S1$  and  $S2$
  - You'd like to use whichever is best
    - how can you tell?

## Comparing two learning systems: Recipe 3

- We want to estimate  $d \equiv error_D(h_1) - error_D(h_2)$
- A natural estimator would be  $\hat{d} \equiv error_{S_1}(h_1) - error_{S_2}(h_2)$
- It turns out the SD for the difference is  $\sigma_{\hat{d}} \equiv \sqrt{\frac{p_1(1-p_1)}{n_1} + \frac{p_2(1-p_2)}{n_2}}$   
where  $p_i = error_{S_i}(h_i)$
- And you then use the same rule for a confidence interval:  
$$p \pm z_n \sigma_{\hat{d}}$$

# Comparing two learning systems

- Very common problem
  - You run YFCL and MFCL on a problem
  - YFCL is a clever *improvement* on MYCL
    - it's usually about the same
    - but sometimes a *little* better
  - Often the difference between the two is hard to see because of the variance associated with  $error_S(h)$

# Comparing two learning systems with a paired test: Recipe 4

- We want to estimate
- Partition  $S$  into disjoint  $T_1, \dots, T_k$  and define
- By the CLT the average of the  $Y_i$ 's is normal assuming that  $k > 30$
- To pick the best hypothesis, see if the mean is significantly far away from zero.

$$d \equiv error_D(h_1) - error_D(h_2)$$

$$Y_i \equiv error_{T_i}(h_1) - error_{T_i}(h_2)$$

  
**SAME** sample

Key point: the sample errors may vary a lot, but if  $h_1$  is consistently better than  $h_2$ , then  $Y_i$  will usually be negative.

Question: Why should the  $T_i$ 's be disjoint?

# Comparing two learning systems with a paired test: Recipe 4

- We want to estimate
- Partition  $S$  into disjoint  $T_1, \dots, T_k$  and define
- By the CLT the average of the  $Y_i$ 's is normal assuming that  $k > 30$
- To pick the best hypothesis, see if the mean is significantly far away from zero, according to the normal distribution.

$$d \equiv error_D(h_1) - error_D(h_2)$$

$$Y_i \equiv error_{T_i}(h_1) - error_{T_i}(h_2)$$

Key point: the sample errors may vary a lot, but if  $h_1$  is **consistently** better than  $h_2$ , then  $Y_i$  will usually be negative.

The **null hypothesis** is that  $Y$  is normal with a zero mean. We want to estimate the **probability** of seeing the sample of  $Y_i$ 's actually observed given that hypothesis.

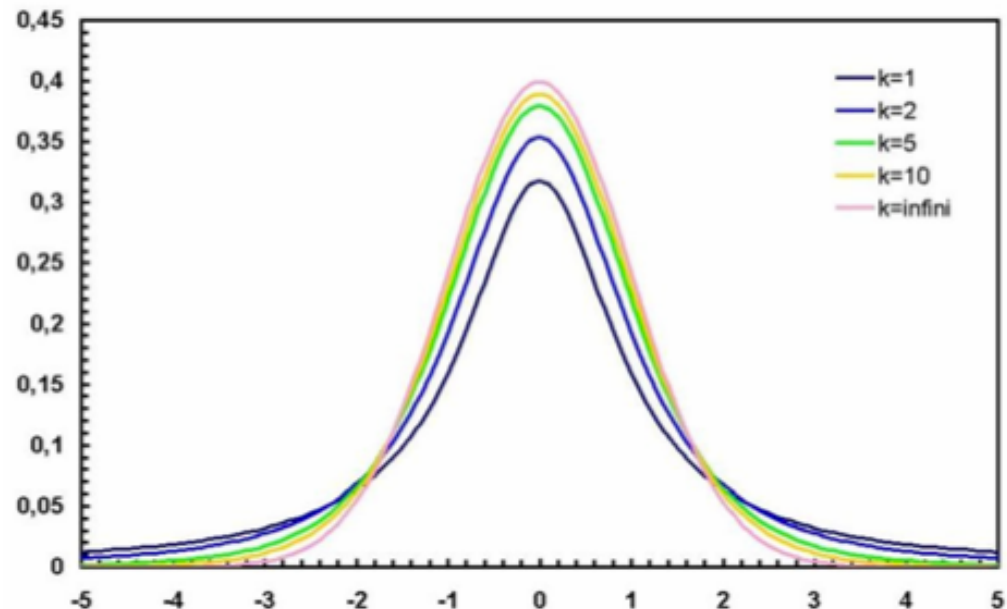
# Comparing two learning systems with a paired test: Recipe 4

partition	$\text{error}_{T_i}(h1)$	$\text{error}_{T_i}(h2)$	diff
T1	0.35	0.30	0.05
T2	0.17	0.16	0.01
...			
<i>avg</i>	<i>0.23</i>	<i>0.21</i>	<i>0.03</i>

We only care about the SD and average of the last column

# Comparing two learning systems with a paired test: Recipe 4

- We want to estimate  $d \equiv error_D(h_1) - error_D(h_2)$
- Partition  $S$  into disjoint  $T_1, \dots, T_k$  and define  $Y_i \equiv error_{T_i}(h_1) - error_{T_i}(h_2)$
- If  $k < 30$ , the average of the  $Y_i$ 's is a *t-distribution* with  $k-1$  degrees of freedom.
- To pick the best hypothesis, see if the mean is significantly far away from zero, according to the *t* distribution.



# **A SIGN TEST**

# A slightly different question

- So far we've been evaluating/comparing hypotheses, not learning algorithms.
- Comparing **hypotheses**:
  - I've learned an  $h(\mathbf{x})$  that tells me if a Piazza post  $\mathbf{x}$  for 10-601 will be rated as a "good question". How accurate is it on the distribution  $D$  of messages?
- Comparing **learners**:
  - I've written a learner  $L$ , and a tool that scrapes Piazza and creates labeled training sets  $(\mathbf{x}_1, y_1)$ , for any class's Piazza site, from the first six weeks of class. How accurate will  $L$ 's hypothesis be for a **randomly selected class**, say 10-701?
  - Is  $L_1$  better or worse than  $L_2$ ?

# A slightly different question

Train/Test Datasets	$\text{error}_{U_i}(h1)$ $h1 = L1(T_i)$	$\text{error}_{U_i}(h2)$ $h2 = L2(T_i)$	diff	
T1/U1	0.35	0.30	0.05	SCS courses
T2/U2	0.37	0.31	0.06	
T3/U3	0.09	0.08	0.01	English courses
T4/U4	0.06	0.07	-0.01	
...				
avg	0.23	0.21	0.03	

Problem: the differences might be **multimodal** - drawn from a mix of two normals

We have a different train set  $T$  and unseen test set  $U$  for each class's web site.

# The sign test for comparing learners across multiple learning problems

Ignore  
ties!

Train/Test Datasets	$\text{error}_{U_i}(h_1)$ $h_1 = L_1(T_i)$	$\text{error}_{U_i}(h_2)$ $h_2 = L_2(T_i)$	diff	sign(diff)
T1/U1	0.35	0.30	0.05	+1
T2/U2	0.37	0.31	0.06	+1
T3/U3	0.09	0.08	0.01	+1
T4/U4	0.06	0.07	-0.01	-1
...				
<i>avg</i>	<i>0.23</i>	<i>0.21</i>	<i>0.03</i>	

More robust: create a **binary** random variable, true iff L1 loses to L2

...given that L1 and L2 score differently

Then estimate a confidence interval for that variable - which is a **binomial**

# Another variant of the sign test: McNemar's test

Make the partitions as *small* as possible:  
so  $T_i$  contains one example  $\{x_i\}$

Ignore  
ties!

partition	$\text{error}_{T_i}(h1)$	$\text{error}_{T_i}(h2)$	diff
$\{x1\}$	1.0	0.0	+1.0
$\{x2\}$	0.0	1.0	-1.0
$\{x3\}$	1.0	1.0	0.0
...			

More robust: create a **binary** random variable, true iff L1 loses to L2

...given that L1 and L2 score differently

Then estimate a confidence interval for that variable - which is a **binomial**

# **CROSS VALIDATION**

# A slightly different question

- What if there were ten sections of 10-601?
- Comparing **hypotheses**:
  - I've learned an  $h(\mathbf{x})$  that tells me if a Piazza post  $\mathbf{x}$  for 10-601 will be rated as a “good question”. How accurate is it on the distribution  $D$  of messages?
- Comparing **learners**:
  - I've written a learner  $L$ , and a tool that scrapes Piazza and creates labeled training sets  $(\mathbf{x}_1, y_1)$ , from the first six weeks of class. How accurate will  $L$ 's hypothesis be for another section of 10-601?
  - Is  $L_1$  better or worse than  $L_2$ ?

# A slightly different question

- What if there were ten sections of 10-601?
- Comparing hypotheses:
  - I've learned an  $h(\mathbf{x})$  that tells me if a Piazza post  $\mathbf{x}$  for 10-601 will be rated as a “good question”. How accurate is it on the distribution  $D$  of messages?
- Comparing learners:
  - I've written a learner  $L$ , and a tool that scrapes Piazza and creates labeled training sets  $(\mathbf{x}_1, y_1)$ , from the first six weeks of class. How accurate will  $L$ 's hypothesis be for another section of 10-601?
  - Is  $L_1$  better or worse than  $L_2$ ?
  - How to account for variability in the training set?

# A paired- t-test using cross validation

We want to use **one** dataset  $S$  to create a number of different-looking  $T_i/U_i$  that are drawn from the same distribution as  $S$ .

Train/Test Datasets	$\text{error}_{U_i}(h1)$ $h1 = L1(T_i)$	$\text{error}_{U_i}(h2)$ $h2 = L2(T_i)$	diff
T1/U1	0.35	0.30	0.05
T2/U2	0.37	0.31	0.06
T3/U3	0.09	0.08	0.01
T4/U4	0.06	0.07	-0.01
...			
<i>avg</i>	<i>0.23</i>	<i>0.21</i>	<i>0.03</i>

One approach:  
**cross-validation.**

Split into  $K$  random disjoint, similar-sized “folds”.

Let  $T_i$  contain  $K-1$  folds and let  $U_i$  contain the last one.

# **SOME OTHER METRICS USED IN MACHINE LEARNING**

# Two wrongs vs two rights

predicted class (expectation)	actual class (observation)	
	tp	fp
	(true positive) Correct result	(false positive) Unexpected result
	fn (false negative) Missing result	tn (true negative) Correct absence of result

Problem: predict if a YouTube video will go viral

Problem: predict if a YouTube comment is useful

Problem: predict if a web page is about “Machine Learning”

# Two wrongs vs two rights

predicted class (expectation)	actual class (observation)	
	tp	fp
	(true positive) Correct result	(false positive) Unexpected result
	fn (false negative) Missing result	(true negative) Correct assessment of result

Problem: predict if a YouTube video will go viral

Problem: predict if a YouTube comment is useful

Problem: predict if a web page is about “Machine Learning”

# Precision and Recall

predicted class (expectation)	actual class (observation)	
	tp	fp
	(true positive) Correct result	(false positive) Unexpected result
	fn	tn
	(false negative) Missing result	(true negative) Correct absence of result

$$\text{Precision} = \frac{tp}{tp + fp} \quad \sim = \text{Pr}(\text{actually pos} | \text{predicted pos})$$

$$\text{Recall} = \frac{tp}{tp + fn} \quad \sim = \text{Pr}(\text{predicted pos} | \text{actually pos})$$

# F-measure

$$\text{Precision} = \frac{tp}{tp + fp}$$

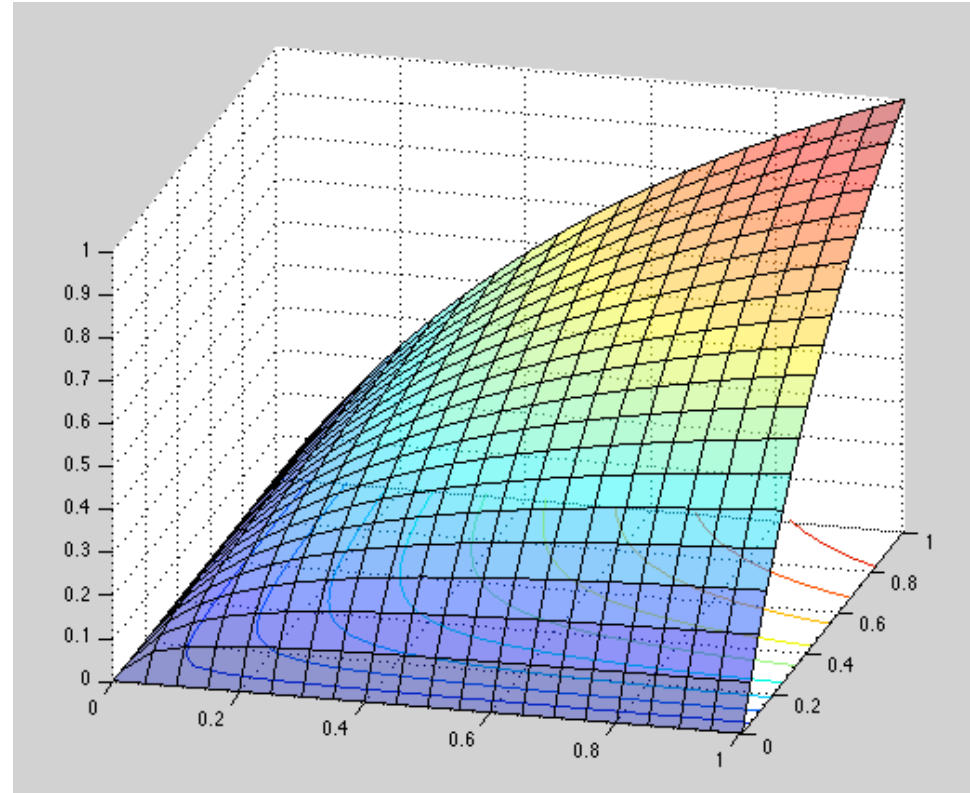
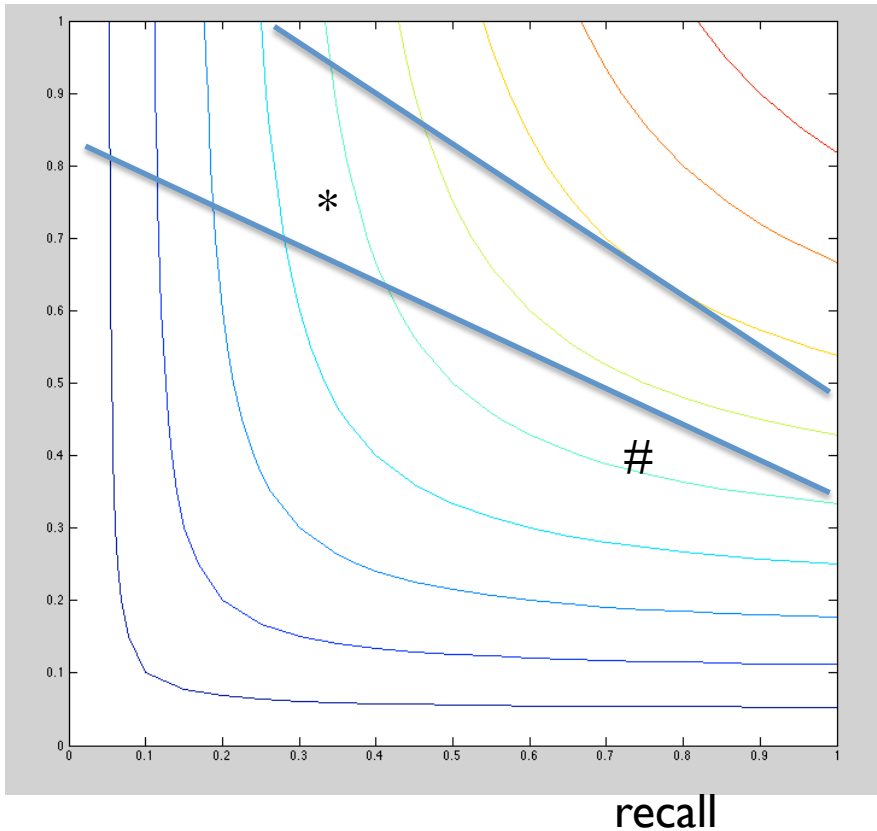
$$\text{Recall} = \frac{tp}{tp + fn}$$

$= \frac{tp}{tp + fn}$	actual class (observation)	
	tp (true positive) Correct result	fp (false positive) Unexpected result
	fn (false negative) Missing result	tn (true negative) Correct absence of result
	predicted class (expectation)	

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} = \frac{1}{\left(\frac{1}{2}\right)\left(\frac{1}{P} + \frac{1}{R}\right)}$$

# F-measure

precision



Precision, Recall, and F-Measure vary as the *threshold* between positive and negative changes for a classifier

# Two wrongs vs two rights

predicted class (expectation)	actual class (observation)	
	tp	fp
	(true positive) Correct result	(false positive) Unexpected result
	fn (false negative) Missing result	tn (true negative) Correct absence of result

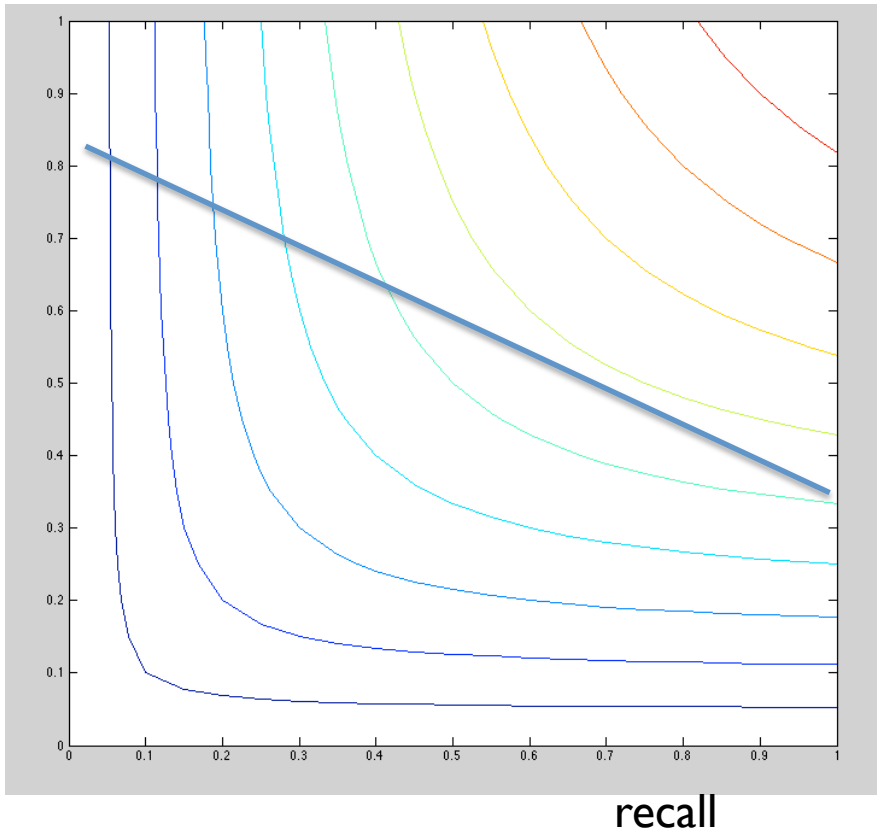
Problem: predict if a YouTube video will go viral

Problem: predict if a YouTube comment is useful

Problem: predict if a web page is about “Machine Learning”

# Average Precision

precision



$$avgP = \left( \frac{1}{n_{pos}} \right) \sum_{k: tp \text{ at rank } k} P(k)$$

mean average precision (MAP) is avg  
prec averaged over several datasets

# ROC Curve and AUC

Receiver  
Operating  
Characteristic  
curve

Area Under  
Curve (AUC)

