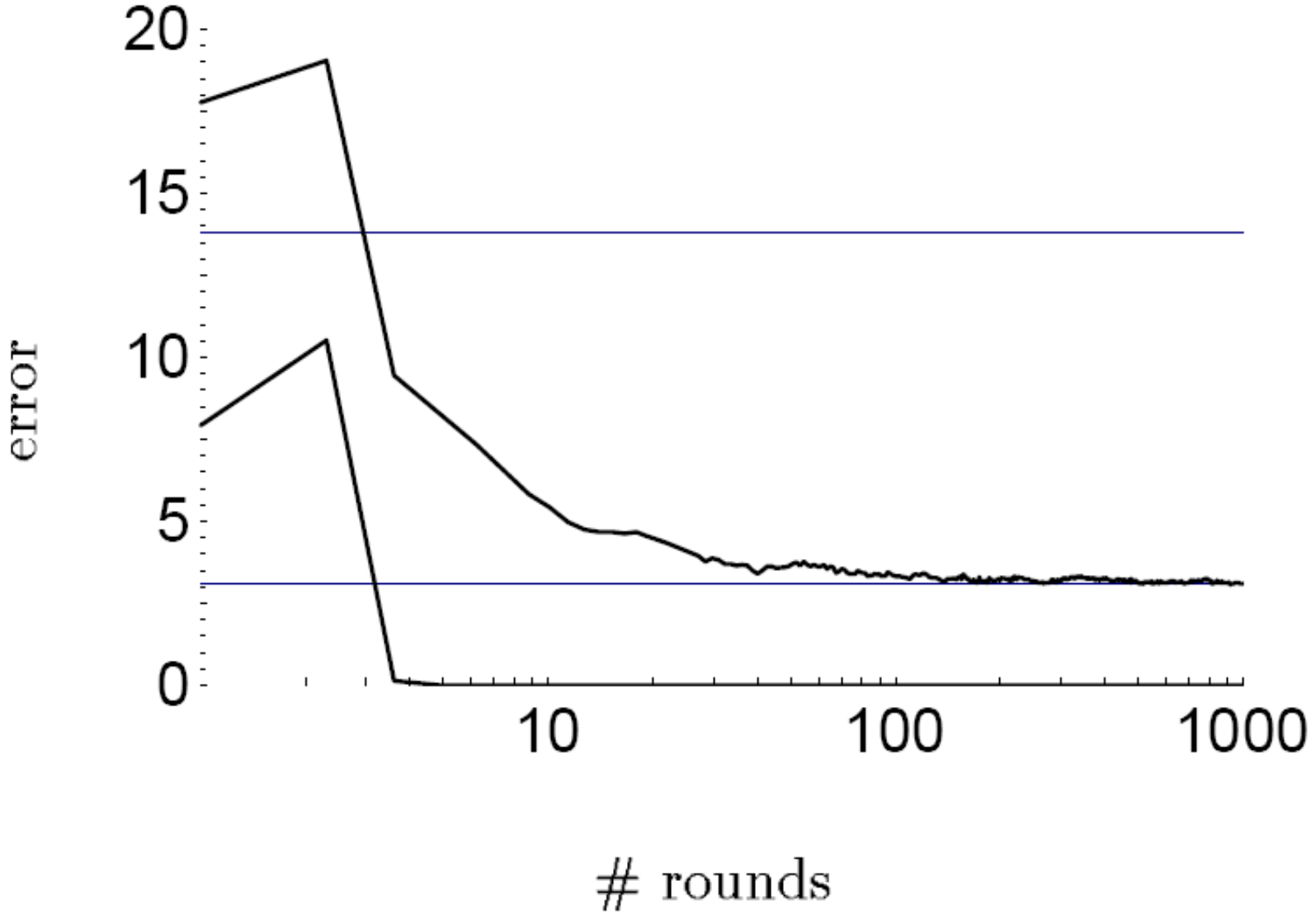
CS / Philo 372

week 9

Yet More on Learning

adaboost – a typical error curve



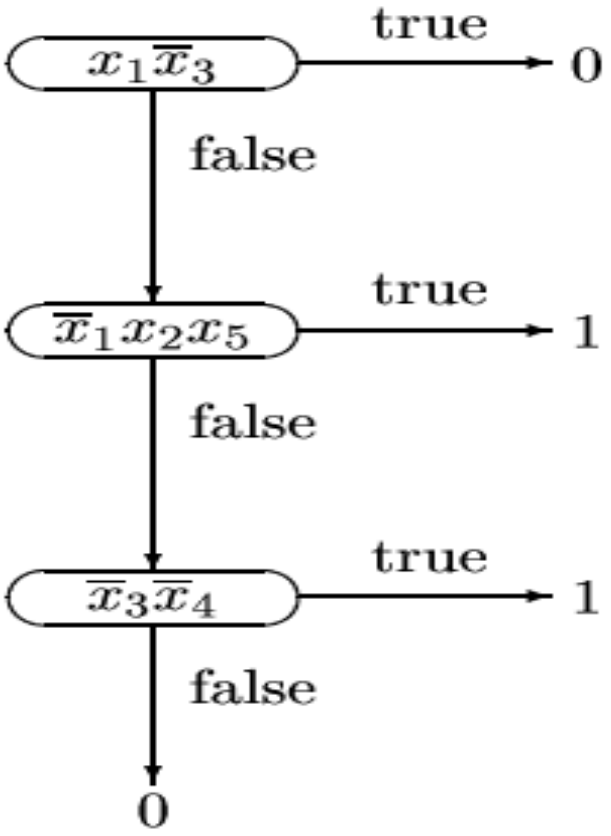
boosting

- Note that test set accuracy continues to improve even when training set accuracy is 100%!!!!
  - This is consistently observed**!!!!**
- On many datasets test set accuracy will eventually go down as boosting rounds continue.
  - Why?

Decision Lists

From Rivest (1987)

- A simple set rules in propositional logic
- Rules are evaluated like prolog, including negation as failure



PAC proofs

- k-DL is polynomial learnable
  - where k is the number of terms allowable in each rule
- k-DNF is not learnable
  - unless P=NP
- Anything that a depth k decision tree can represent can also be represented by a k-DL

Learning Decision Lists

- let examples be <X,v> where X is a vector of boolean values (the attributes) and v is a boolean value (the category label).
- a function f is consistent with an example iff f(x)=v
- Questions:
  - economical: requires few examples to learn
  - efficient: requires little computational time to learn

efficiency & boolean functions

- Consider "lambda = lg(F)"
  - where F is a boolean function
  - the lambda is the minimum number of bits that will need to be transmitted to tell someone else the function F
- For boolean functions each example contains at most 1 bit of information
  - So, any alg for learning F must see at least lambda examples.
- Hence, have a precise notion of efficiency

DL learner

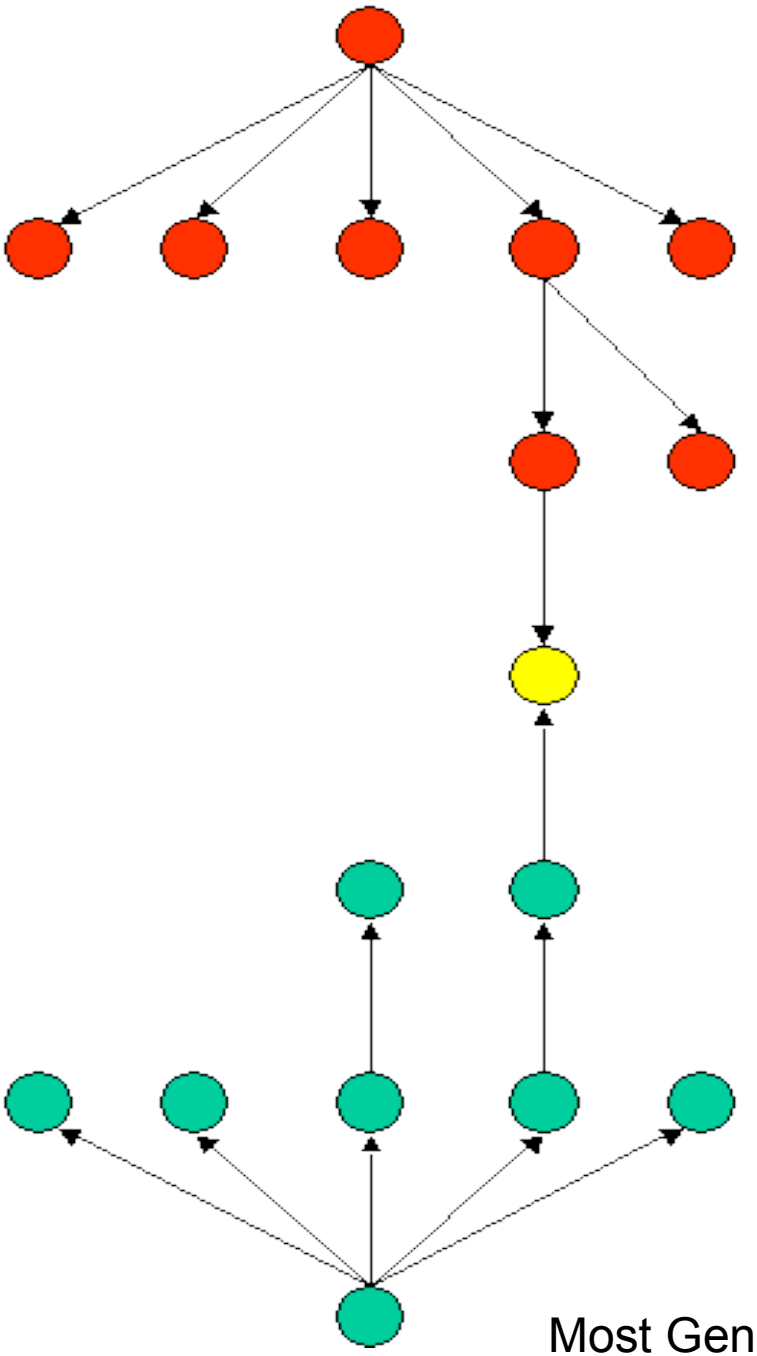
- let: S be a set of examples
- Pick example (ex1) from S
  - find a set of up to k attributes of ex1 so that all examples in S that have those attributes have the same category as ex1
    - call this set S1
  - remove S1 from S
  - return to top unless S is empty
- Learning an individual DL takes O(nm) time
  - n is number of examples
  - m is number of attributes
- Overall $O(n^2m)$

DLs

- Things that can go wrong
  - Noise
  - functions that are not in k-DL
    - What do these look like?
    - What could you do?
  - Get examples after you started ...
    - what do you do?
    - what do you do on a decision tree?
  - NP hard to get optimal K-DL
- Are rather uninteresting in practice

Version Spaces

- Name and approach from Mitchell (1978)
- Idea is to maintain two versions of what you know
  - Things you are sure are wrong
  - things you are sure are right
- Gradually expand wrong and right lists until the two lists merge in a single point



Most specific hypothesis

Most General Hypothesis

Version space alg

- Init: need with a single + example
- Create 2 sets S (specific) and G (General)
  - S <- +example
  - G= {}
- on new example
  - if – modify G in all possible ways to exclude new example (i.e. specialize G)
  - if +, modify S in all possible ways to include (i.e. generalize S)

Version Spaces example

- Japan Honda Blue 1980 Economy Positive
- Japan Toyota Green 1970 Sports Negative
- Japan Toyota Blue 1990 Economy Positive
- USA Chrysler Red 1980 Economy Negative
- Japan Honda White 1980 Economy Positive

- Initially:
  - s=Japan Honda Blue 1980 Economy Positive
  - g={}

Version Spaces

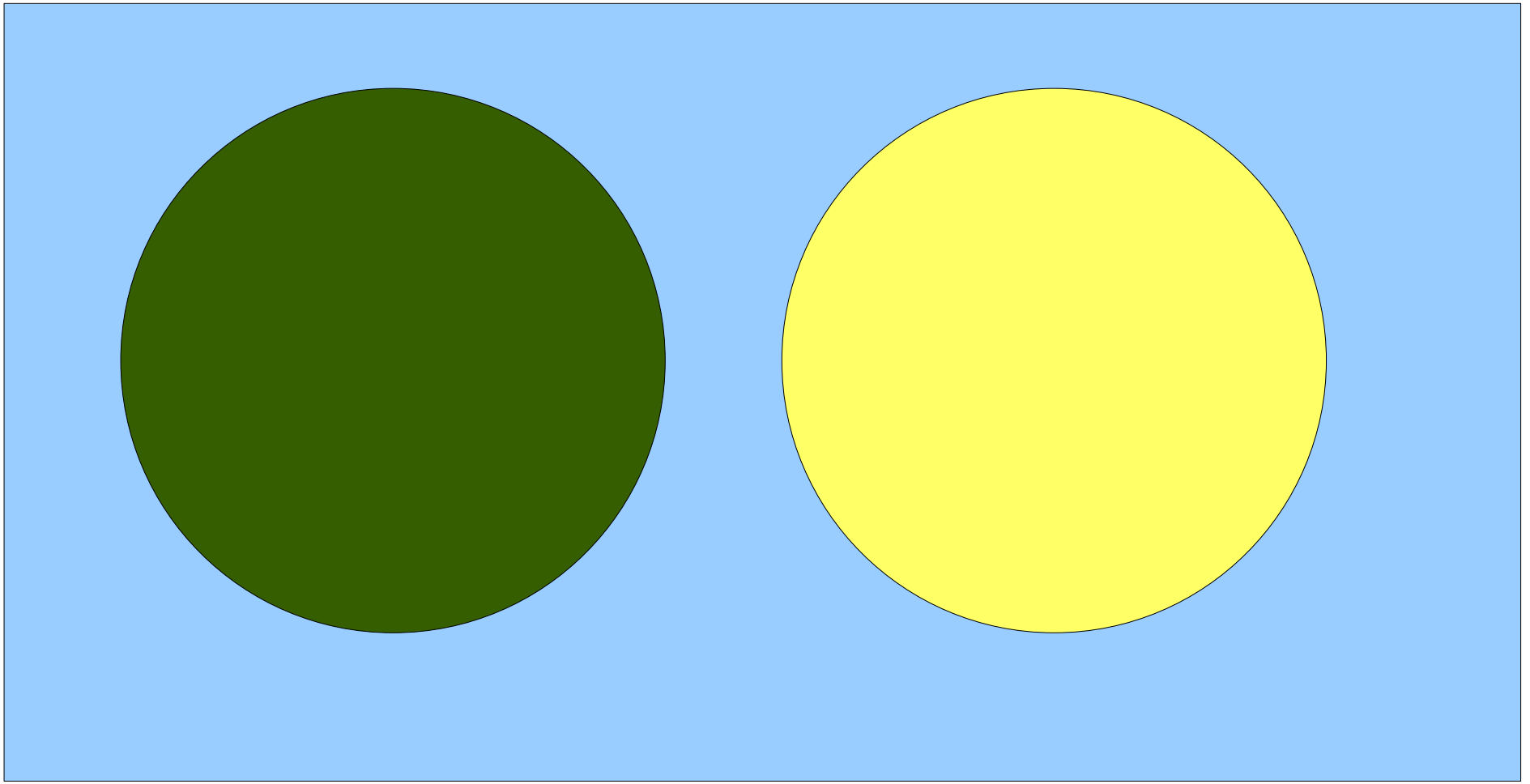
- Good news
  - naturally "on-line", unlike decision trees or lists
  - probably "efficient" in Rivest's sense.
- Bad News
  - Handling Noise
    - compare to Trees
  - where do the features come from
  - Size of S and G sets.

Active Learning

- Suppose:
  - Algorithm is able to learn one example at a time
  - Examples are free, example labels are expensive
    - any text domain
- Then might make sense to allow learning algorithm to select the examples to label
  - This is "active learning"
  - reported to reduce training set size by factor of 500
- Problem:
  - what examples do you label?
  - Can your learner provide requisite info?

Examples to get labels for

Suppose a 2 category classification problem in 2D.
Further suppose circles represent
locations of examples already seen in each category.



Info Needed from algorithm

- Indication of "confidence" in label
  - Decision Trees?
  - Decision Lists?
  - AdaBoosted decision stumps?
- Note that binary classifiers can provide a "yes/no" label and a separate confidence
- Other programs provide a probability statement that can be be interpreted as both a label and a confidence

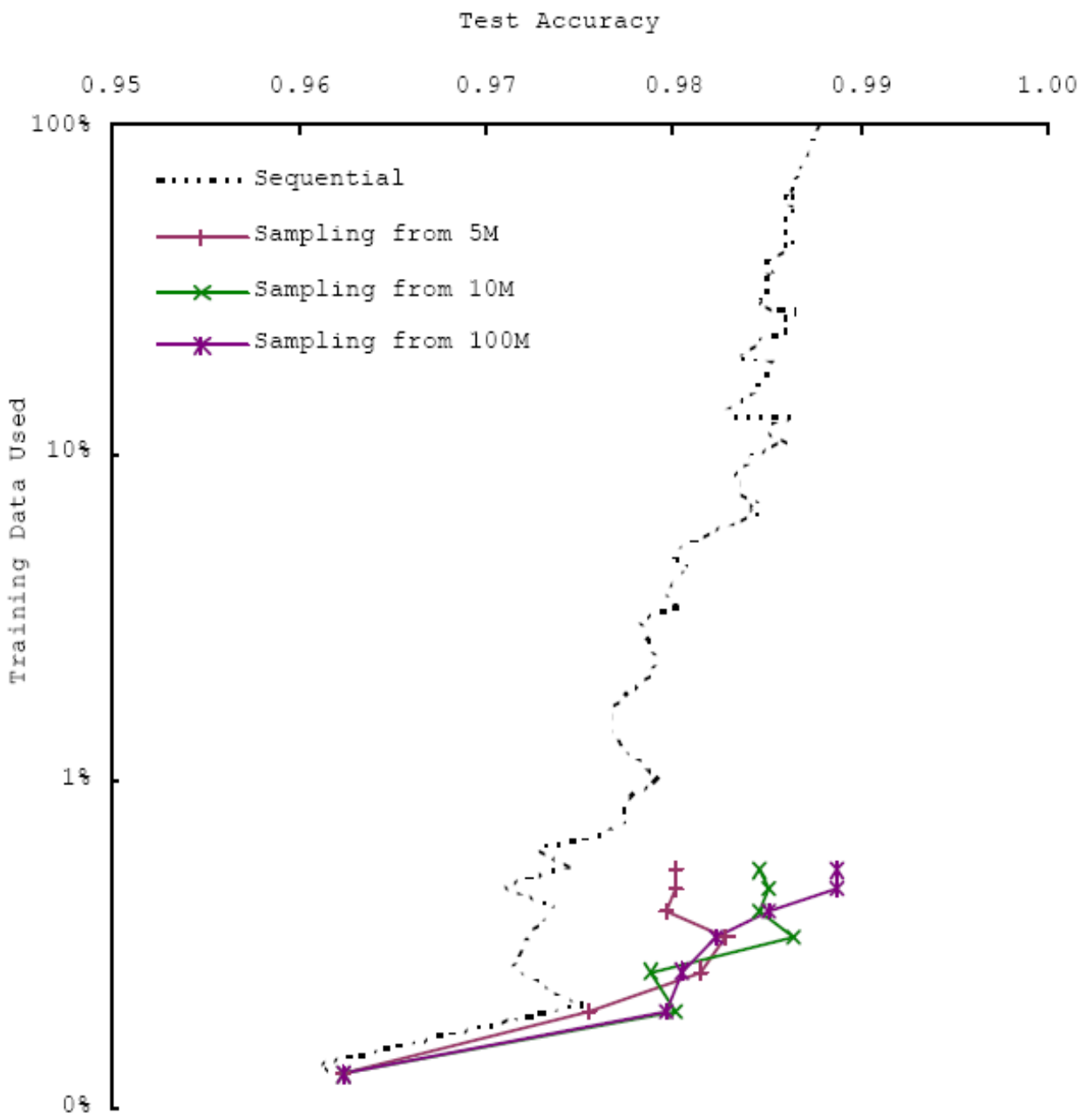
Uncertainty Sampling

Lewis & Catlett (1994)

1. Obtain an initial classifier
2. While expert is willing to label instances
   (a) Apply the current classifier to each unlabeled instance
   (b) Find the b instances for which the classifier is least certain of class membership
   (c) Have the expert label the subsample of b instances
   (d) Train a new classifier on all labeled instances

Uncertainty Sampling

- Problem
  - suppose use same program to select uncertain as to label.
    - Then program's bias tends to reinforce itself by selecting examples it is uncertain about
    - This can lead to strongly overpredicting low frequency classes (among other things)
    - SO??
      - Banko & Brill use committee voting.
        - have commmittee vote on unlabeled set
        - Take N/2 on which committee agrees least
        - Take N/2 randomly selected (?????)
        - Retrain committee and repeat on smaller unlabeled set


Test Accuracy
0.95
0.96
0.97
0.98
0.99
1.00
100%
10%
1%
0%
Training Data Used
Sequential
Sampling from 5M
Sampling from 10M
Sampling from 100M

Analysis

- About 0.5% of examples is sufficient to achieve accuracy.
- Bigger sets of unlabeled examples improved classification accuracy
  - even though most were never even seen by the classifier
- Committee disagreement does predict errors

Classifiers In Agreement	Test Accuracy
10	0.8734
9	0.6892
8	0.6286
7	0.6027
6	0.5497
5	0.5000

Is this a form of boosting?