CIS 606 Machine Learning, Spring 2013

Lecturers: Wei Lee Woon and Zeyar Aung

Lecture 13
Ensemble Learning

Original Source:

www.cs.cornell.edu/Courses/cs4700/2008fa/PPT/CS4700-EL.ppt

(by Prof. Carla P. Gomes, Computer Science Dept., Cornell University)

Ensemble Learning

So far – learning methods that learn a single hypothesis, chosen form a hypothesis space that is used to make predictions.

Ensemble learning ▪ select a collection (ensemble) of hypotheses and combine their predictions.

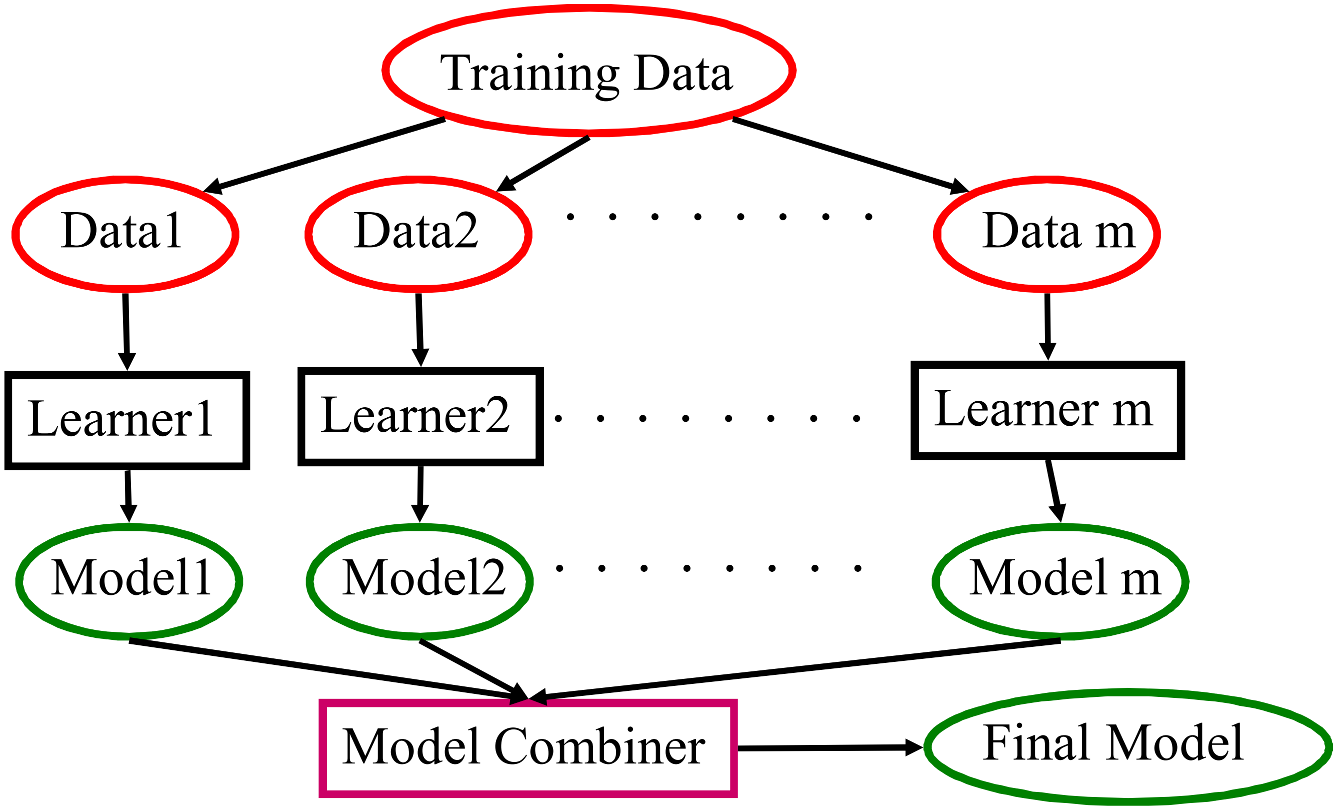
Example 1 - generate 100 different decision trees from the same or different training set and have them vote on the best classification for a new example.

Key motivation: reduce the error rate. Hope is that it will become much more unlikely that the ensemble of will misclassify an example.

Learning Ensembles

Learn multiple alternative definitions of a concept using different training data or different learning algorithms.

Combine decisions of multiple definitions, e.g. using weighted voting.



Source: Ray Mooney

Value of Ensembles

"No Free Lunch" Theorem

- No single algorithm wins all the time!

When combing multiple independent and diverse decisions each of which is at least more accurate than random guessing, random errors cancel each other out, correct decisions are reinforced.

Examples: Human ensembles are demonstrably better

- How many jelly beans in the jar?: Individual estimates vs. group average.
- Who Wants to be a Millionaire: Audience vote.

Source: Ray Mooney

Example: Weather Forecast

Reality	⛈	☀	☀	⛈	⛈	☀	☀
1	⛈	⛈ X	☀	☀ X	⛈	☀	⛈ X
2	☀ X	☀	☀	☀ X	⛈	☀	⛈ X
3	⛈	☀	⛈ X	⛈	☀ X	⛈ X	☀
4	⛈	☀	⛈ X	⛈	☀ X	☀	☀
5	⛈	⛈ X	☀	⛈	⛈	⛈ X	☀
Combine	⛈	☀	☀	⛈	⛈	☀	☀

Intuitions

<u>Majority vote</u>

Suppose we have 5 completely independent classifiers…

- If accuracy is 70% for each
  - $(.7^5)+5(.7^4)(.3)+10(.7^3)(.3^2)$
  - **83.7% majority vote accuracy**
- 101 such classifiers
  - **99.9% majority vote accuracy**

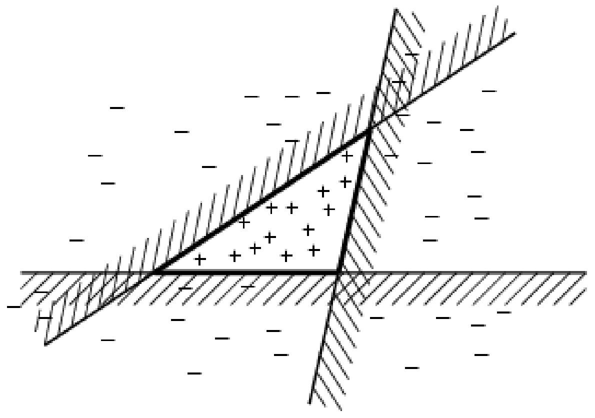
Note: Binomial Distribution: The probability of observing *x* heads in a sample of *n* independent coin tosses, where in each toss the probability of heads is *p*, is

$$P(X = x|p, n) = \frac{n!}{r!(n-x)!}p^x(1-p)^{n-x}$$

Ensemble Learning

Another way of thinking about ensemble learning:

- way of enlarging the hypothesis space, i.e., the ensemble itself is a hypothesis and the new hypothesis space is the set of all possible ensembles constructible form hypotheses of the original space.



Increasing power of ensemble learning:

Three linear threshold hypothesis
(positive examples on the non-shaded side);
Ensemble classifies as positive any example classified positively be all three. The resulting triangular region hypothesis is not expressible in the original hypothesis space.

Different Learners

Different learning algorithms

Algorithms with different choice for parameters

Data set with different features

Data set = different subsets

Homogenous Ensembles

Use a single, arbitrary learning algorithm but manipulate training data to make it learn multiple models.

- Data1 ≠ Data2 ≠ … ≠ Data m
- Learner1 = Learner2 = … = Learner m

Different methods for changing training data:

- Bagging: Resample training data
- Boosting: Reweight training data

In WEKA, these are called ***meta-learners***, they take a learning algorithm as an argument (***base learner***) and create a new learning algorithm.

Bagging

Bagging

Create ensembles by "*bootstrap aggregation*", i.e., repeatedly randomly resampling the training data (Brieman, 1996).

Bootstrap: draw N items from X with replacement

Bagging

- Train M learners on M bootstrap samples
- Combine outputs by voting (e.g., majority vote)

Decreases error by decreasing the variance in the results due to ***unstable learners***, algorithms (like decision trees and neural networks) whose output can change dramatically when the training data is slightly changed.

Bagging - Aggregate Bootstrapping

Given a standard training set D of size n

For i = 1 .. M

- Draw a sample of size $n^* < n$ from D uniformly and with replacement
- Learn classifier C_i

Final classifier is a vote of $C_1 .. C_M$

Increases classifier stability/reduces variance

Boosting

Strong and Weak Learners

Strong Learner ▪ Objective of machine learning

- Take labeled data for training
- Produce a classifier which can be *arbitrarily accurate*

Weak Learner

- Take labeled data for training
- Produce a classifier which is more accurate than random guessing

Boosting

Weak Learner: only needs to generate a hypothesis with a training accuracy greater than 0.5, i.e., < 50% error over any distribution

Learners

- Strong learners are very difficult to construct
- Constructing weaker Learners is relatively easy

Questions: Can a set of **weak learners** create a single **strong learner** ?

YES ù

Boost weak classifiers to a strong learner

Boosting

Originally developed by computational learning theorists to guarantee performance improvements on fitting training data for a ***weak learner*** that only needs to generate a hypothesis with a training accuracy greater than 0.5 (Schapire, 1990).

Revised to be a practical algorithm, AdaBoost, for building ensembles that empirically improves generalization performance (Freund & Shapire, 1996).

Key Insights

Instead of sampling (as in bagging) re-weigh examples!

Examples are given weights. At each iteration, a new hypothesis is learned (weak learner) and the examples are reweighted to focus the system on examples that the most recently learned classifier got wrong.

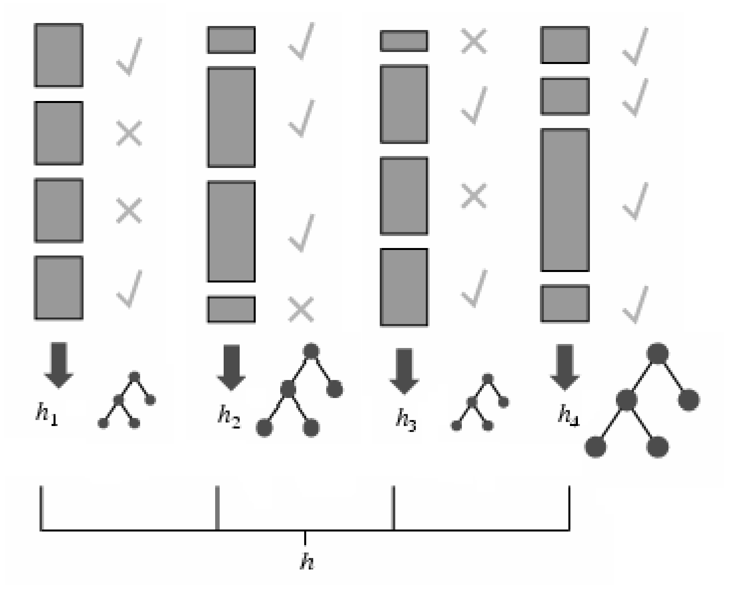
Final classification based on weighted vote of weak classifiers

Adaptive Boosting

Each rectangle corresponds to an example, with weight proportional to its height.

Crosses correspond to misclassified examples.

Size of decision tree indicates the weight of that hypothesis in the final ensemble.



Construct Weak Classifiers

Using Different Data Distribution

- Start with uniform weighting
- During each step of learning
  - Increase weights of the examples which are not correctly learned by the weak learner
  - Decrease weights of the examples which are correctly learned by the weak learner

Idea

- Focus on difficult examples which are not correctly classified in the previous steps

Combine Weak Classifiers

Weighted Voting

- Construct strong classifier by weighted voting of the weak classifiers

Idea

- Better weak classifier gets a larger weight
- Iteratively add weak classifiers
    - Increase accuracy of the combined classifier through minimization of a cost function

Adaptive Boosting: High Level Description

```
C =0; /* counter*/
M = m; /* number of hypotheses to generate*/

1 Set   same weight   for all the examples  (typically each example has weight = 1);

2 While (C < M)
    2.1 Increase counter C by 1.
    2.2 Generate hypothesis  $h_C$ .
    2.3 Increase the weight of the misclassified examples in  hypothesis $h_C$
3 Weighted majority combination of all M hypotheses (weights according to how well
    it performed on the training set).
```

Many variants depending on how to set the weights and how to combine the hypotheses. ADABOOST ▪ quite popular!!!!

Performance of Adaboost

Learner = Hypothesis = Classifier

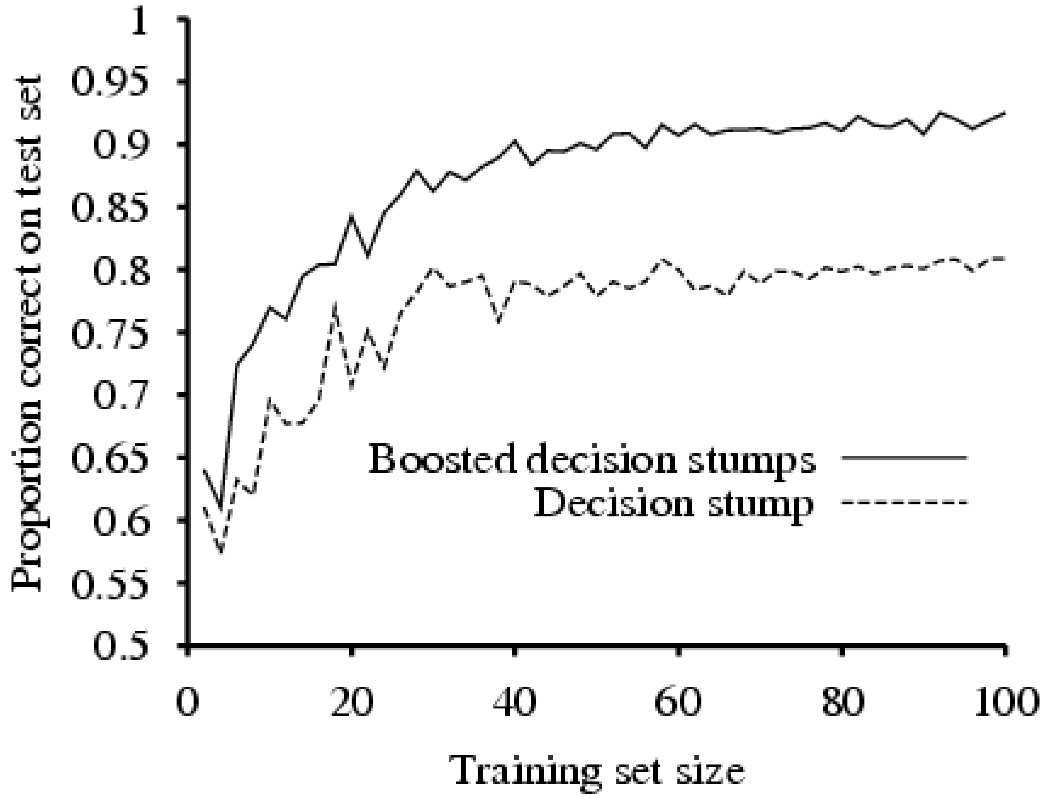
Weak Learner: < 50% error over any distribution

M number of hypothesis in the ensemble.

If the input learning is a Weak Learner, then ADABOOST will return a hypothesis that classifies the training data perfectly for a large enough M, boosting the accuracy of the original learning algorithm on the training data.

Strong Classifier: thresholded linear combination of weak learner outputs.

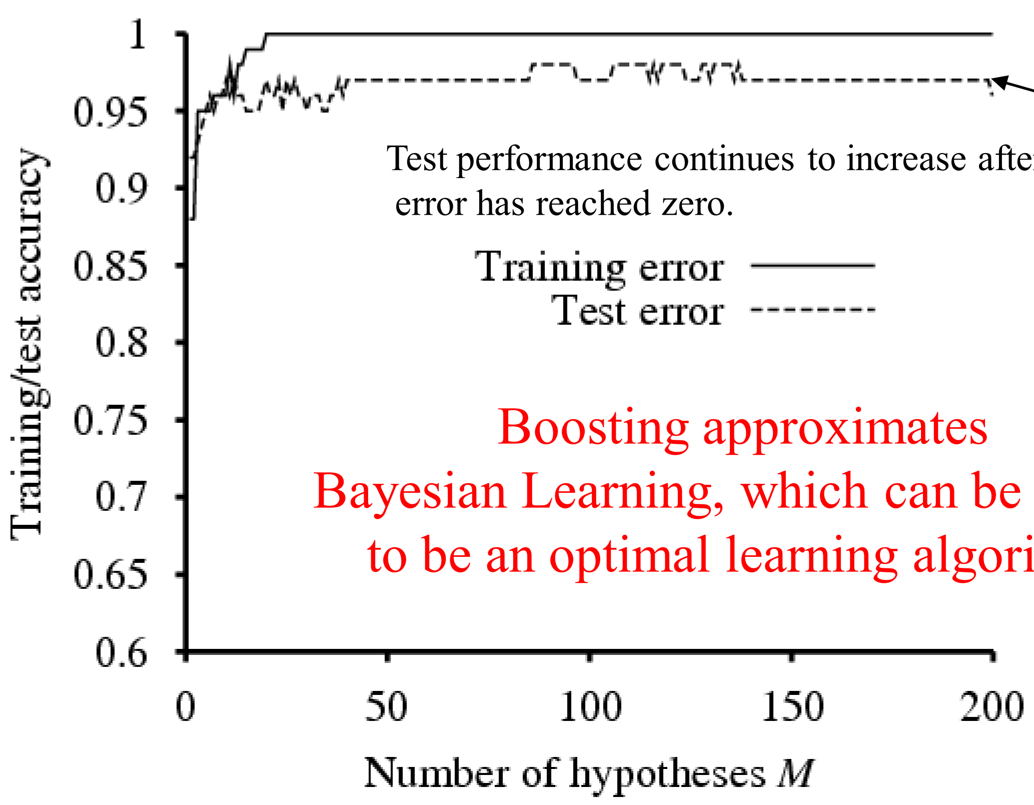
Restaurant Data



Decision stump: decision trees with just one test at the root.

Restaurant Data

Training error reaches zero for M=20 (as predicted by the theorem), and remains zero as more stumps are added to the ensemble.



Test performance continues to increase after training set error has reached zero.

Boosting approximates Bayesian Learning, which can be shown to be an optimal learning algorithm.