Inspire…Educate…Transform.

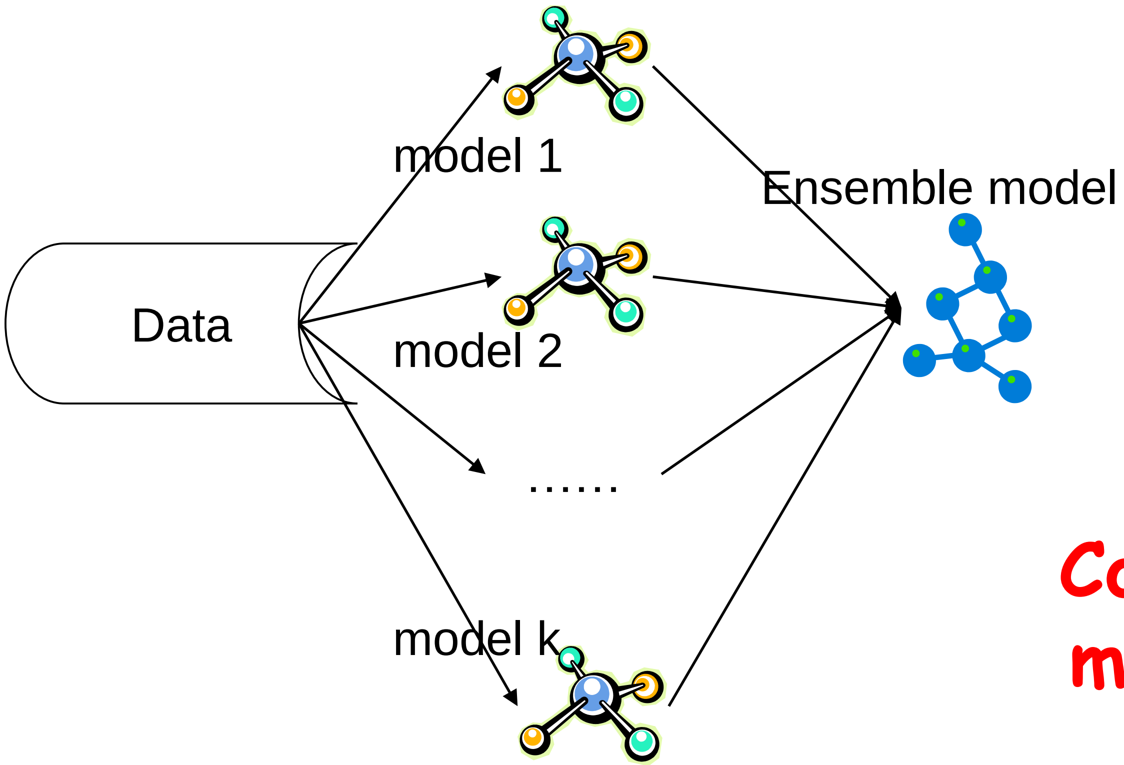
Ensemble Learning

Dr. Manoj Chinnakotla

Senior Applied Scientist, Microsoft
Adjunct Professor, IIIT Hyderabad

18th March, 2017

What is an Ensemble?

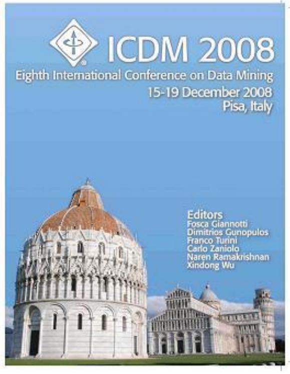


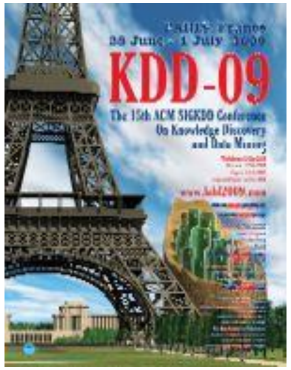
Combine multiple models into one!

Applications: classification, clustering, collaborative filtering, anomaly detection……

Stories of Success

- Million-Dollar Prize
  - Improve the baseline movie recommendation approach of Netflix by 10% in accuracy
  - The top submissions all combine several teams and algorithms as an ensemble





- Data Mining Competitions
  - Classification problems
  - Winning teams employ an ensemble of classifiers

Netflix Prize

- Supervised Learning Task
  - Training data is a set of users and ratings (1,2,3,4,5 stars) those users have given to movies.
  - Construct a classifier that given a user and an unrated movie, correctly classifies that movie as either 1, 2, 3, 4, or 5 stars
  - $1 million prize for a 10% improvement over Netflix's current movie recommender
- Competition
  - At first, single-model methods are developed, and performances are improved
  - However, improvements slowed down
  - Later, individuals and teams merged their results, and significant improvements are observed

Leaderboard

Rank	Team Name	Best Test Score	% Improvement	Best Submit Time
Grand Prize - RMSE = 0.8567 - Winning Team: BellKor's Pragmatic Chaos				
1	BellKor's Pragmatic Chaos	0.8567	10.06	2009-07-26 18:18:28
2	The Ensemble	0.8567	10.06	2009-07-26 18:38:22
3	Grand Prize Team	0.8582	9.90	2009-07-10 21:24:40
4	Opera Solutions and Vandelay United	0.8588	9.84	2009-07-10 01:12:31
5	Vandelay Industries !	0.8591	9.81	2009-07-10 00:32:20
6	PragmaticTheory	0.8594	9.77	2009-06-24 12:06:56
7	BellKor in BigChaos	0.8601	9.70	2009-05-13 08:14:09
8	Dace_	0.8612	9.59	2009-07-24 17:18:43
9	Feeds2	0.8622	9.48	2009-07-12 13:11:51
10	BigChaos	0.8623	9.47	2009-04-07 12:33:59
12	BellKor	0.8624	9.46	2009-07-26 17:19:11
Progress Prize 2008 - RMSE = 0.8627 - Winning Team: BellKor in BigChaos				
13	xiangliang	0.8642	9.27	2009-07-15 14:53:22
14	Gravity	0.8643	9.26	2009-04-22 18:31:32
15	Ces	0.8651	9.18	2009-06-21 19:24:53
Progress Prize 2007 - RMSE = 0.8723 - Winning Team: KorBell				
Cinematch score - RMSE = 0.9525				

“Our final solution (RMSE=0.8567) consists of blending 107 individual results. “

“Predictive accuracy is substantially improved when blending multiple predictors. Our experience is that most efforts should be concentrated in deriving substantially different approaches, rather than refining a single technique. “

Motivation for Ensembles

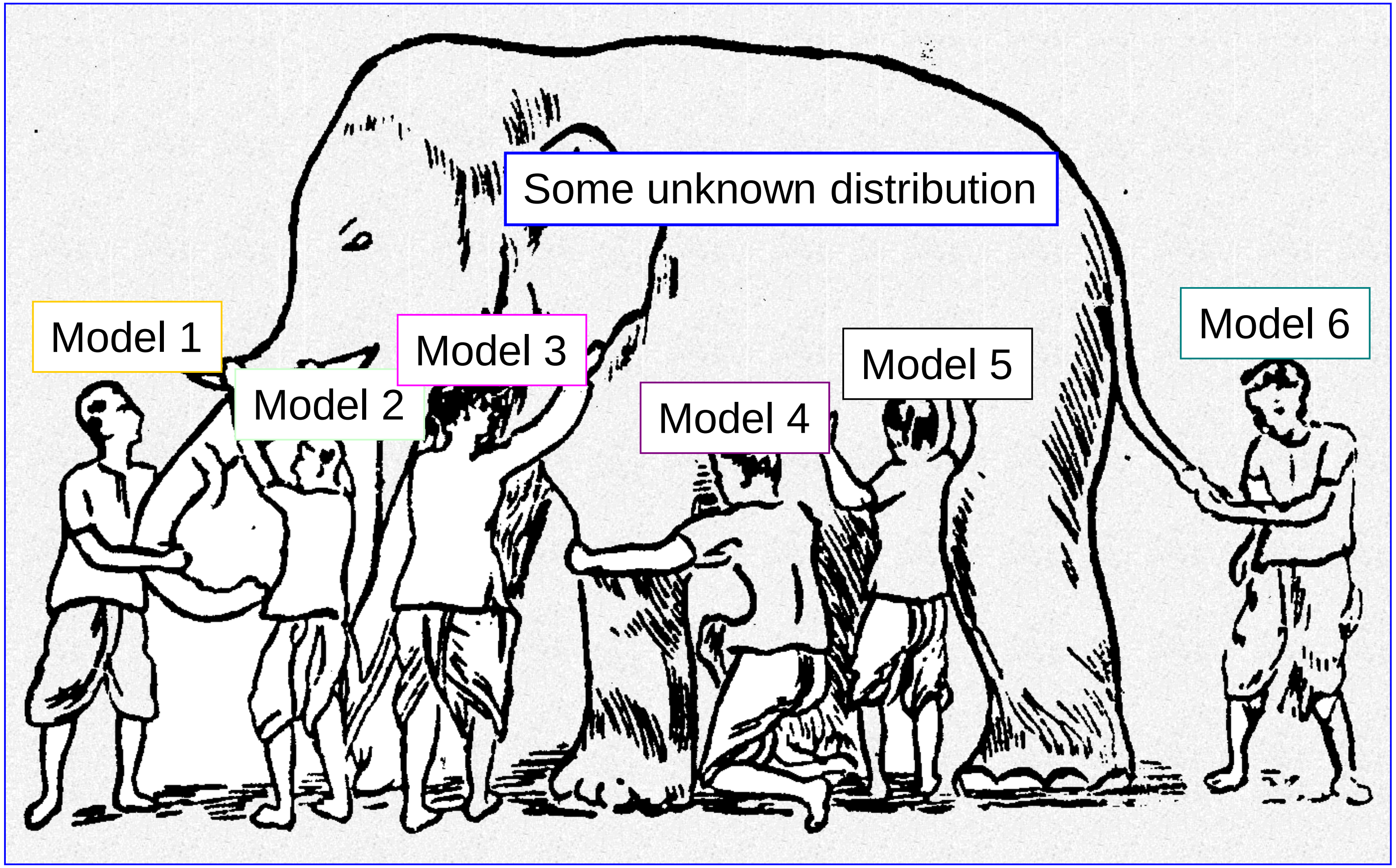
- Ensemble model improves accuracy and robustness over single model methods
- Applications:
  - distributed computing
  - privacy-preserving applications
  - large-scale data with reusable models
  - multiple sources of data
- Efficiency: a complex problem can be decomposed into multiple sub-problems that are easier to understand and solve (divide-and-conquer approach)

Why Ensemble Works? (1)

- Intuition
  - Combining diverse, independent opinions in human decision-making as a protective mechanism (e.g. stock portfolio)
- Uncorrelated Error Reduction
  - Suppose we have 5 completely independent classifiers for majority voting
  - If accuracy is 70% for each
    - 10 (.7^3)(.3^2)+5(.7^4)(.3)+(.7^5)
    - **83.7% majority vote accuracy**
  - 101 such classifiers
    - **99.9% majority vote accuracy**

from T. Holloway, Introduction to Ensemble Learning, 2007.

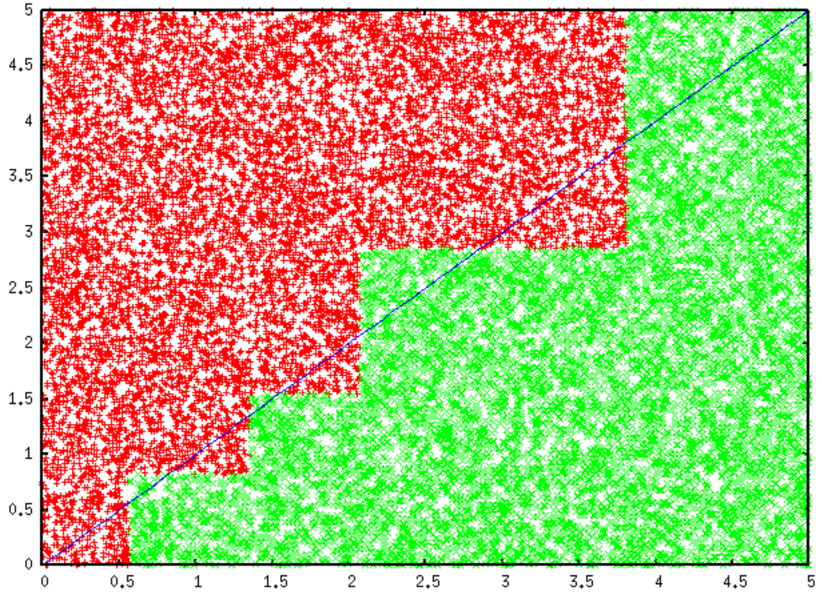
Why Ensemble Works? (2)



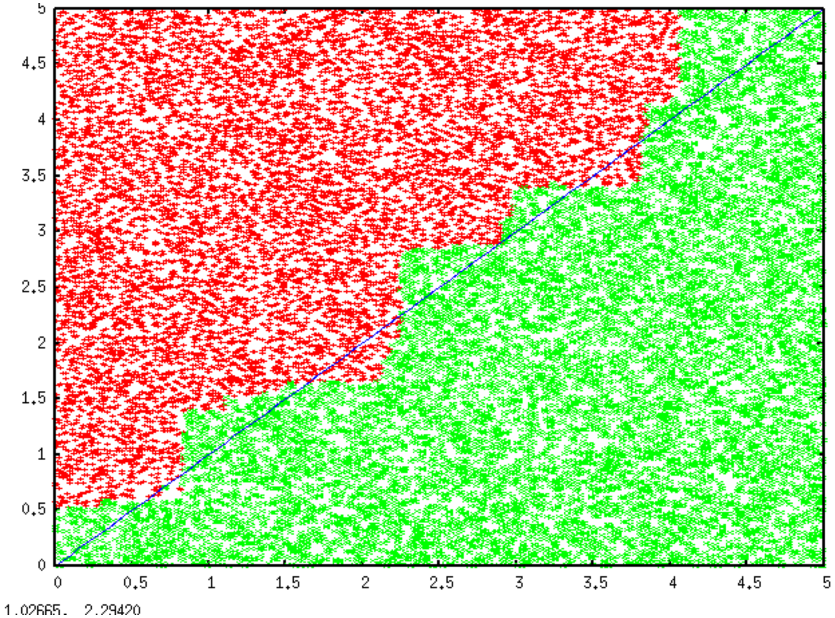
Ensemble gives the global picture!

Why Ensemble Works? (3)

- Overcome limitations of single hypothesis
  - The target function may not be implementable with individual classifiers, but may be approximated by model averaging



Decision Tree



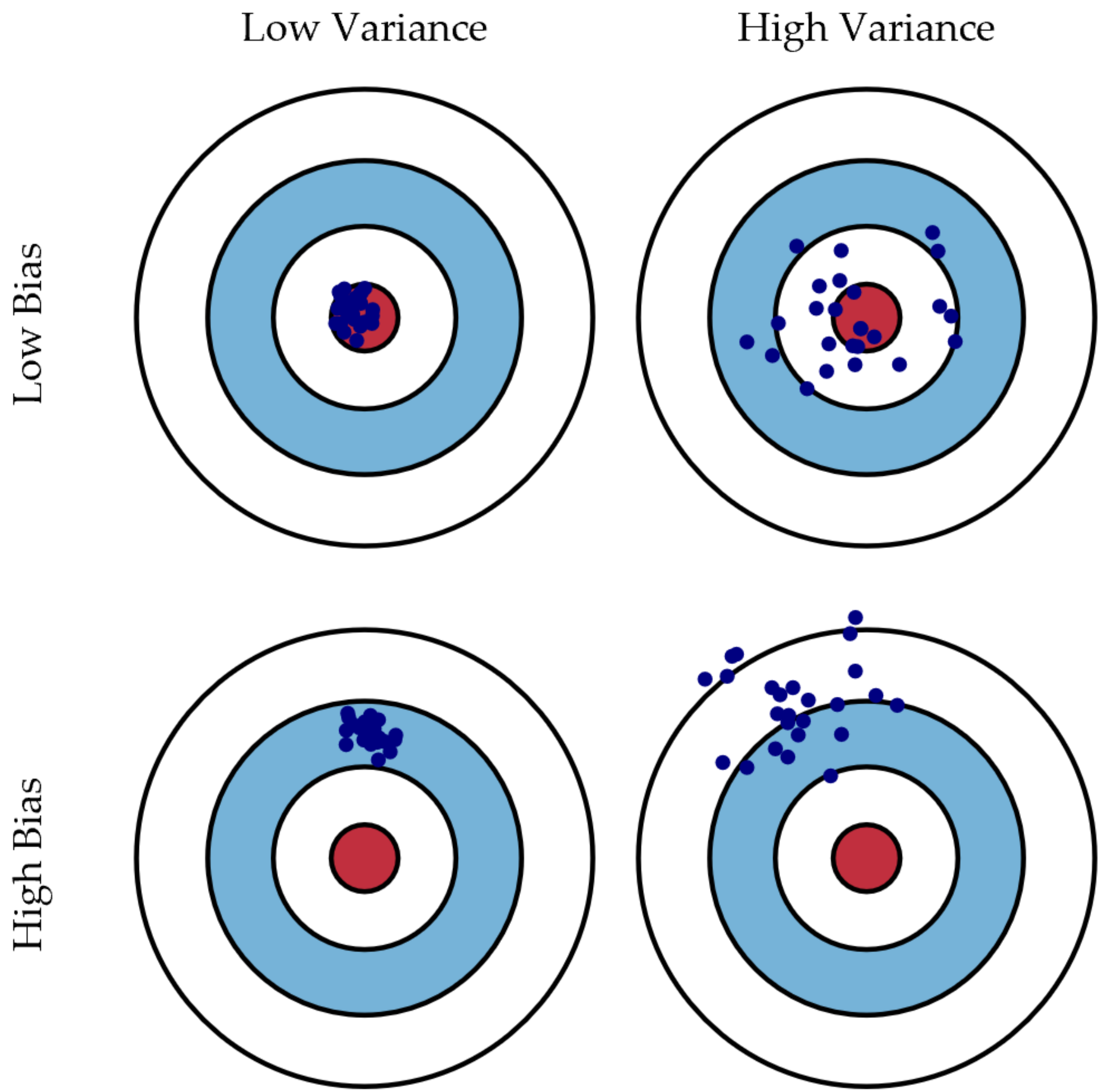
Model Averaging

Generalization Error & Bias, Variance Tradeoff

Bias and Variance

- Bias
  - Measures the accuracy or quality of the algorithm
  - High bias means a poor match
- Variance
  - Measures the precision or specificity of the match
  - A high variance means a weak match
- We would like to minimize each of these
- Unfortunately, we can’t do this independently, since there is a trade-off

Bias and Variance (Contd..)



Bias-Variance Analysis in Regression

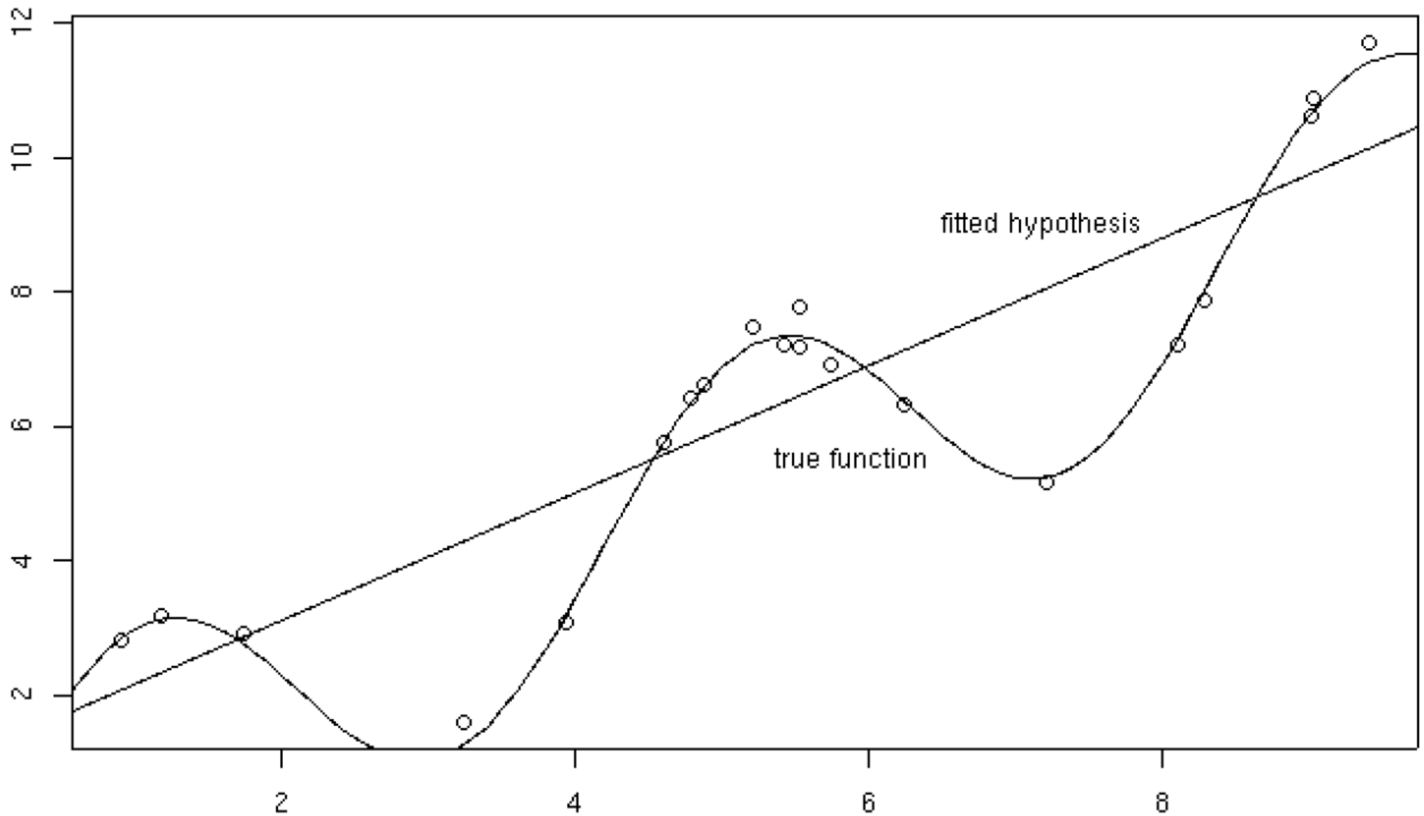
True function is y = f(x) + e

- Where e is normally distributed with zero mean and standard deviation s
- Given a set of training examples, {(xi, yi)}, we fit an hypothesis h(x) = w . x + b to the data to minimize the squared error

$$\sum_i \left(y_i - h(x_i)\right)^2$$

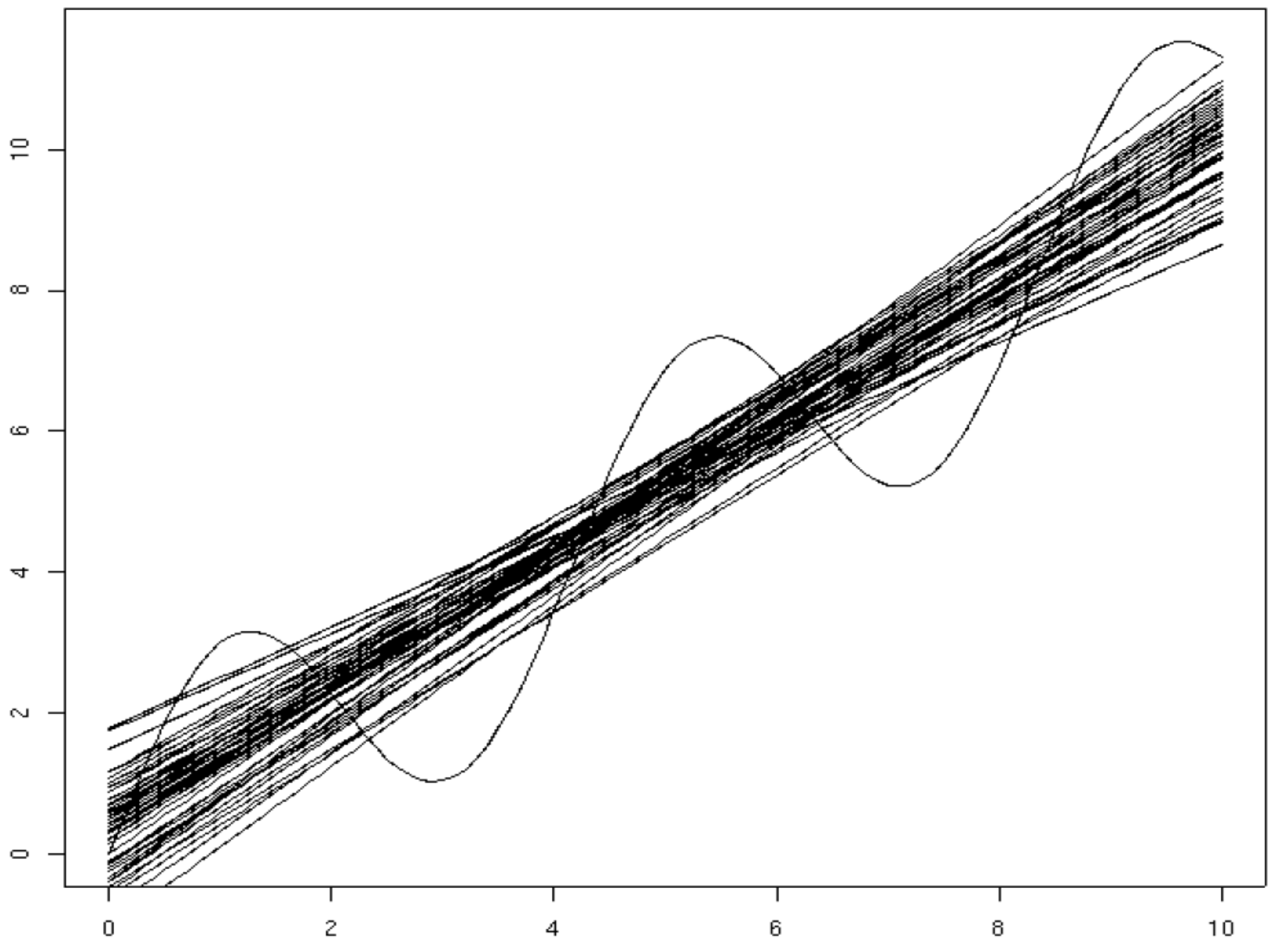
Bias-Variance Analysis in Regression (Contd..)

$$y = x + 2\sin(1.5x) + N(0,0.2)$$



Bias-Variance Analysis in Regression (Contd..)

50 fits (20 examples each)



Bias-Variance Analysis in Regression (Contd..)

- Given a new point x^*, with observed value $y^* = f(x^*) + \varepsilon$, let's estimate the expected prediction error given by:

$$E\left[\left(y^* - h(x^*)\right)^2\right]$$

Bias-Variance Analysis in Regression (Contd..)

- Imagine that our training sample S is drawn from a population with distribution P(S) then we need to compute:

$$E_P[(y^* - h(x^*))^2]$$

Bias-Variance Analysis in Regression (Contd..)

- Let Z be a random variable with probability distribution P(Z)
- Let $\bar{Z} = E_P[\, Z \,]$ be the average value of Z.
- Lemma: $E[\,(Z - \bar{Z})^2\,] = E[Z^2] - \bar{Z}^2$

$$
\begin{aligned}
E[\,(Z - \bar{Z})^2\,] &= E[\,Z^2 - 2\,Z\,\bar{Z} + \bar{Z}^2\,] \\
&= E[Z^2] - 2\,E[Z]\,\bar{Z} + \bar{Z}^2 \\
&= E[Z^2] - 2\,\bar{Z}^2 + \bar{Z}^2 \\
&= E[Z^2] - \bar{Z}^2
\end{aligned}
$$

- Corollary : $E[Z^2] = E[\,(Z - \bar{Z})^2\,] + \bar{Z}^2$

Bias-Variance Analysis in Regression (Contd..)

$$E[(h(x^*) - y^*)^2] = E[h(x^*)^2 - 2h(x^*)y^* + y^{*2}]$$

$$= E[h(x^*)^2] - 2E[h(x^*)]E[y^*] + E[y^{*2}]$$

$$= E[(h(x^*) - \overline{h(x^*)})^2] + \overline{h(x^*)}^2$$

$$-2\overline{h(x^*)}f(x^*)$$

$$+ E[(y^* - f(x^*))^2] + f(x^*)^2$$

$$= E[(h(x^*) - \overline{h(x^*)})^2] + \qquad \text{VARIANCE}$$

$$\left(\overline{h(x^*)}^2 - f(x^*)\right)^2 \qquad \text{BIAS}$$

$$+ E[(y^* - f(x^*))^2] \qquad \text{NOISE}$$

$$= \mathrm{Var}(h(x^*)) + \mathrm{Bias}(h(x^*))^2 + E[\varepsilon^2]$$

$$= \mathrm{Var}(h(x^*)) + \mathrm{Bias}(h(x^*))^2 + \sigma^2$$

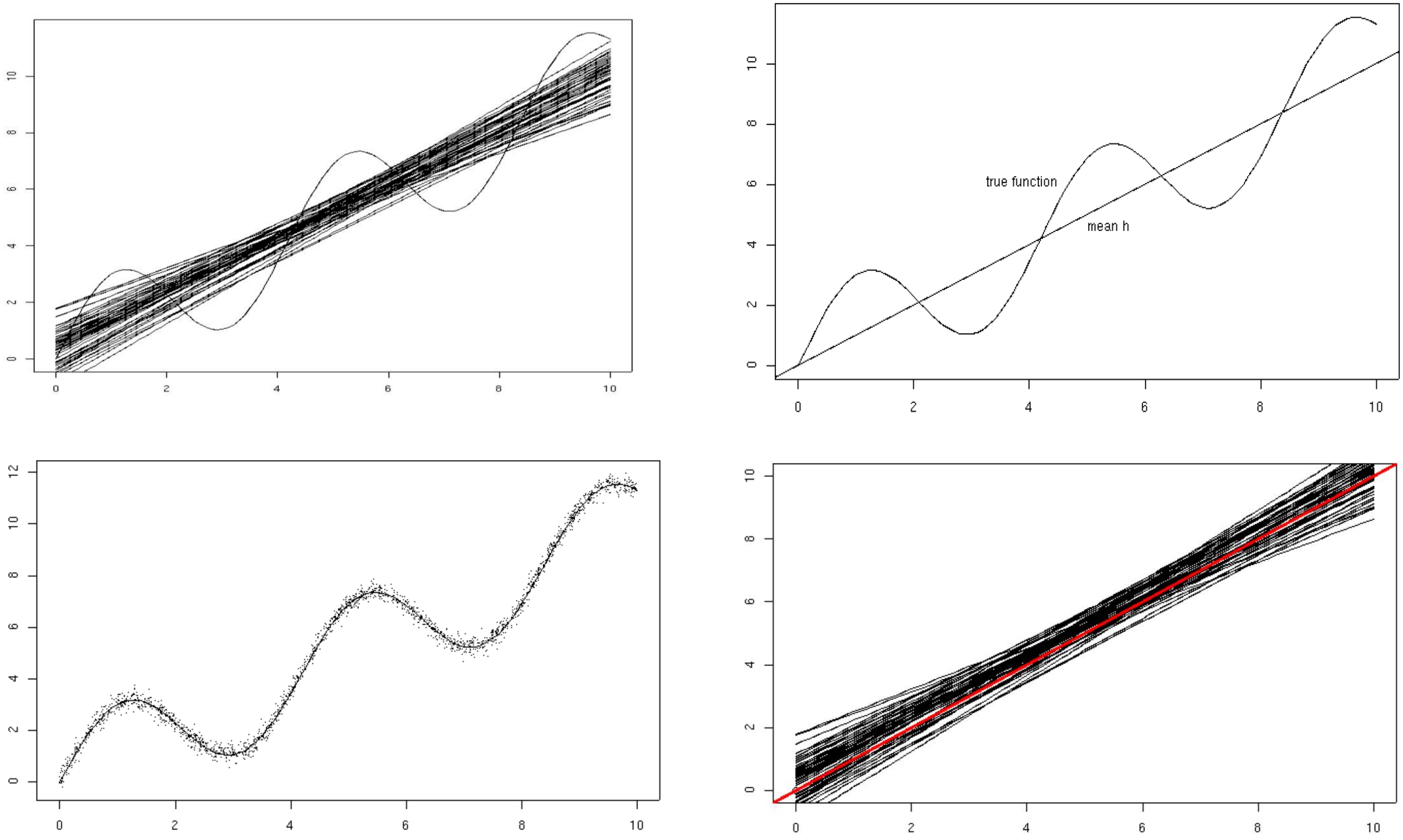
Expected prediction error = Variance + Bias^2 + Noise^2

CSE 7306c

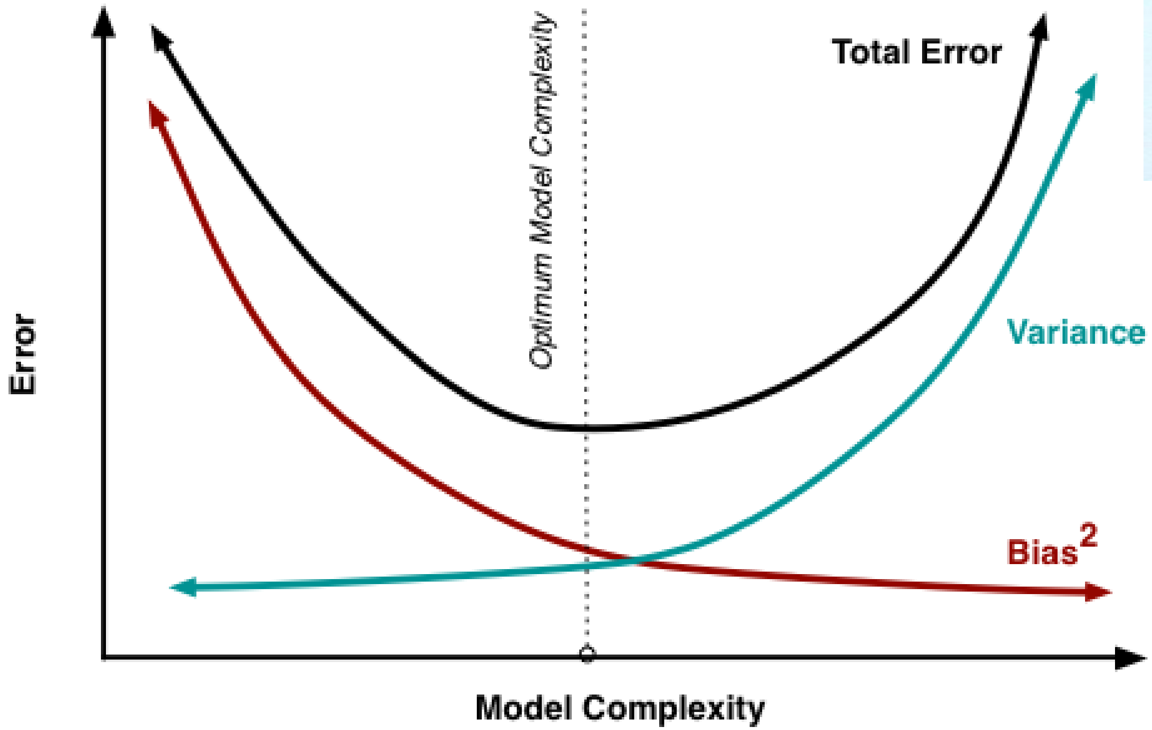
Bias-Variance Analysis in Regression (Contd..)

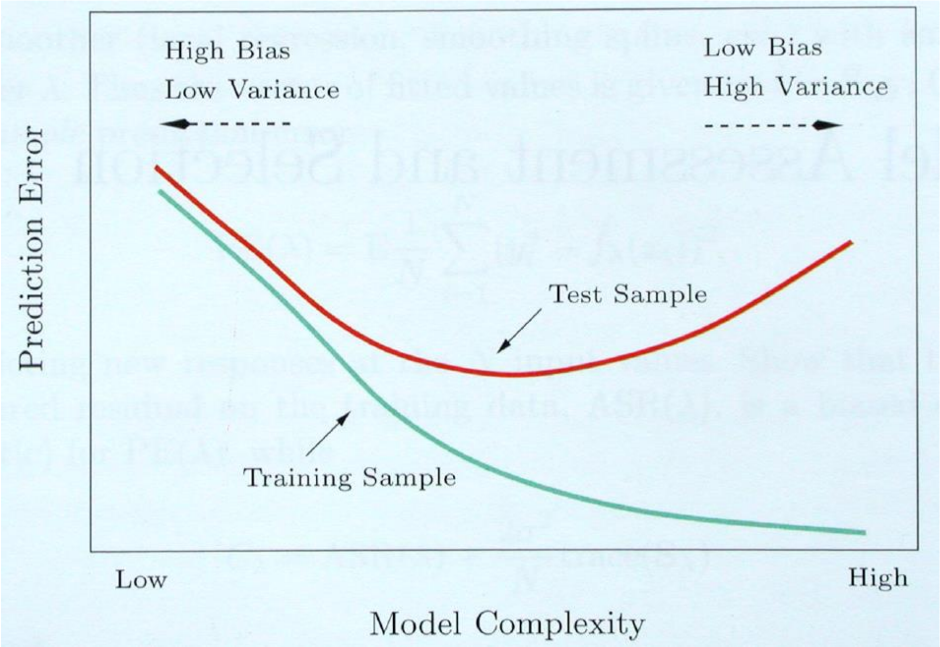
- Variance: $E_P\left[\left(h(x^*) - \overline{h(x^*)}\right)^2\right]$
  - Describes how much $h(x^*)$ varies from one training set S to another
- Bias: $\left[\overline{h(x^*)} - f(x^*)\right]$
  - Describes the average error of $h(x^*)$
- Noise: $E_P[(y^* - f(x^*))^2]$
  - Describes how much y^* varies from the true function $f(x^*)$

Bias-Variance Analysis in Regression (Contd..)



Bias vs. Variance Tradeoff





CSE 7306c

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

Bagging – An Example

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

Bootstrap samples and classifiers:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x	0.1	0.2	0.3	0.4	0.5	0.5	0.9	1	1	1
y	1	1	1	-1	-1	-1	1	1	1	1

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

Combine predictions by majority voting

from P. Tan et al. Introduction to Data Mining.

Random Forests*

- Algorithm
  - Choose *T*—number of trees to grow
  - Choose *m*<*M* (M is the number of total features) —number of features used to calculate the best split at each node (typically 20%)
  - For each tree
    - Choose a training set by choosing *N* times (*N* is the number of training examples) with replacement from the training set
    - For each node, randomly choose *m* features and calculate the best split
    - Fully grown and not pruned
  - Use majority voting among all the trees

*[Breiman01]

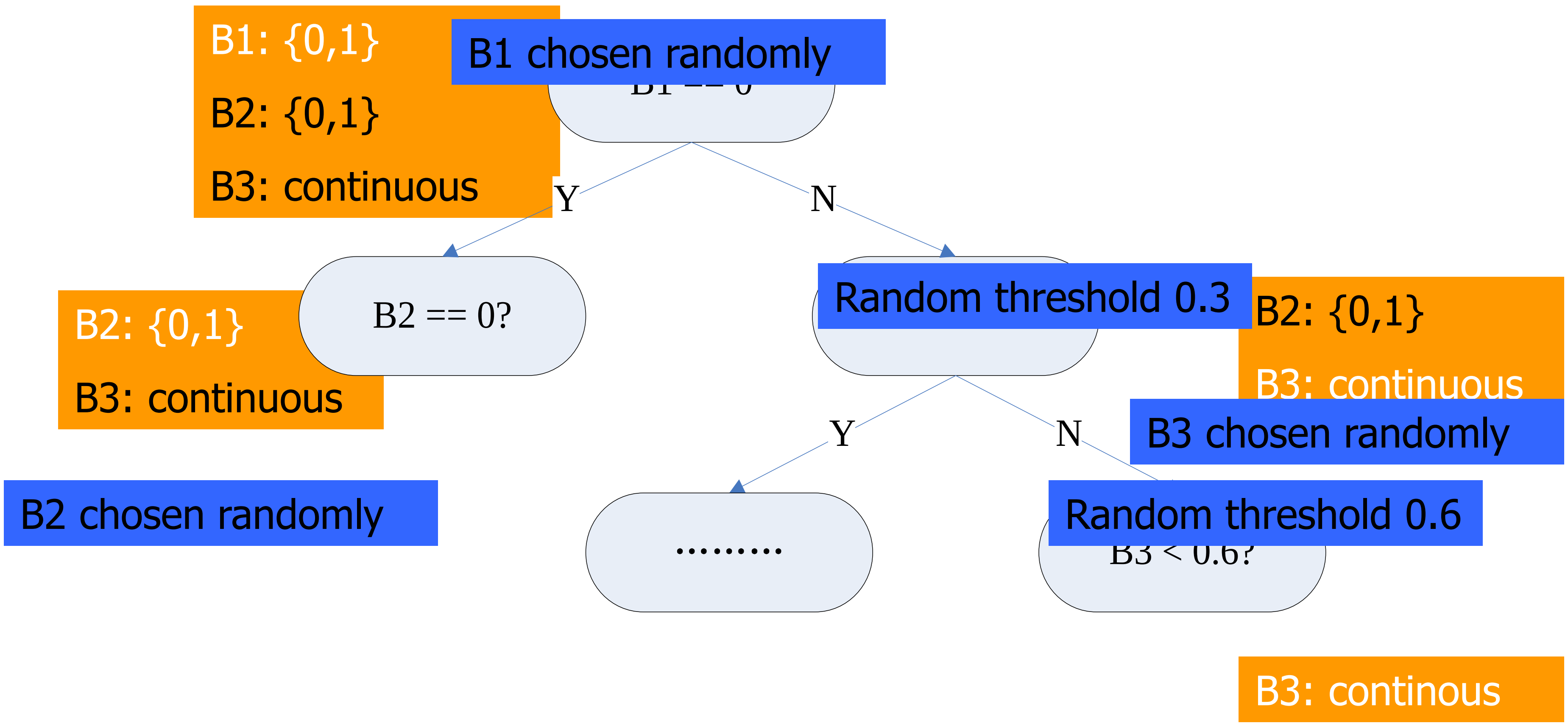
Random Forests

- Discussions
  - Bagging + random features
  - Improve accuracy
    - Incorporate more diversity and reduce variances
  - Improve efficiency
    - Searching among subsets of features is much faster than searching among the complete set

Random Decision Tree

- Algorithm
  - At each node, an un-used feature is chosen randomly
    - A discrete feature is un-used if it has never been chosen previously on a given decision path starting from the root to the current node.
    - A continuous feature can be chosen multiple times on the same decision path, but each time a different threshold value is chosen
  - We stop when one of the following happens:
    - A node becomes too small (<= 3 examples).
    - Or the total height of the tree exceeds some limits, such as the total number of features.
  - Prediction
    - Simple averaging over multiple trees

Random Decision Tree

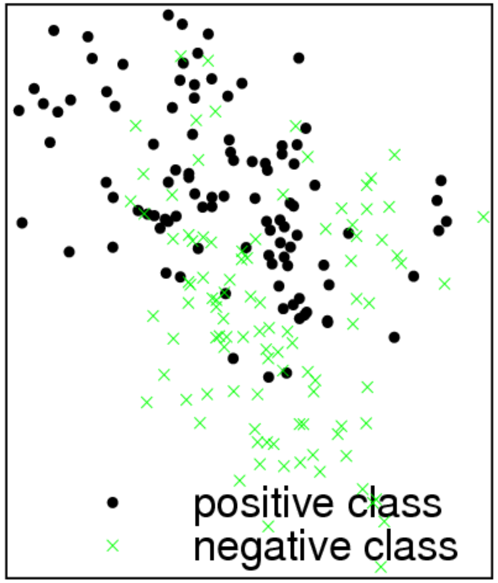


Random Decision Tree

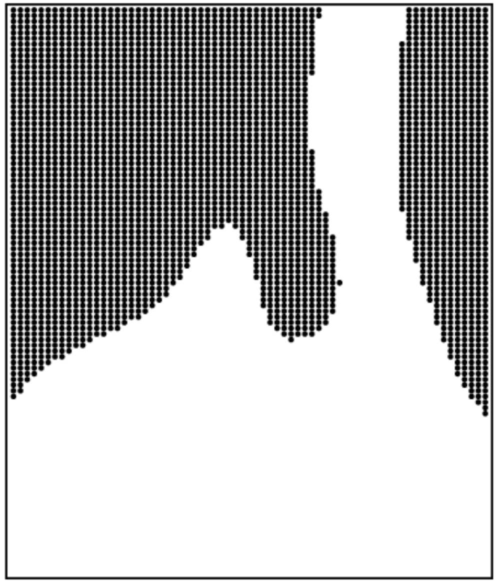
- Potential Advantages
  - Training can be very efficient. Particularly true for very large datasets.
    - No cross-validation based estimation of parameters for some parametric methods.
  - Natural multi-class probability.
  - Imposes very little about the structures of the model.

Optimal Decision Boundary

Figure 3.5: Gaussian mixture training samples and optimal boundary.

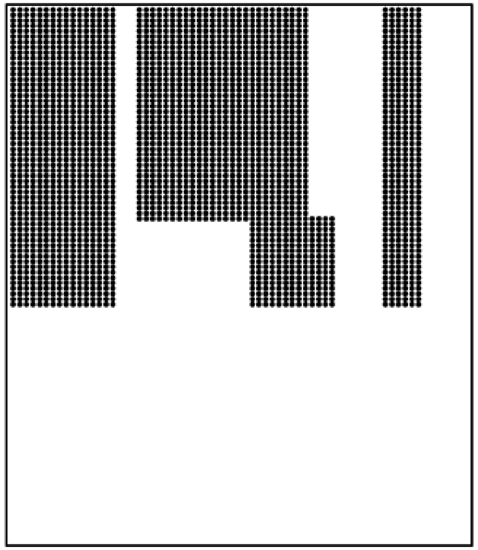


training samples

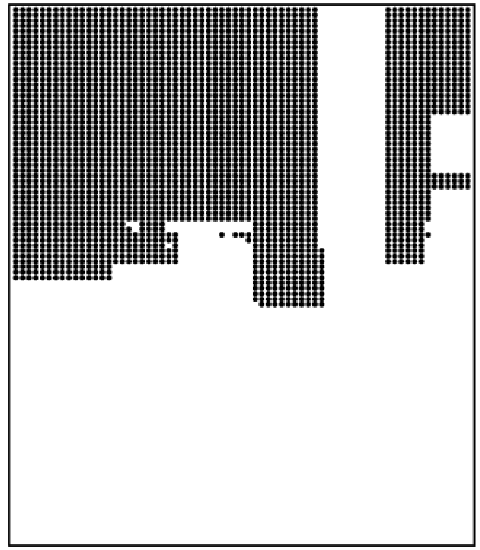


optimal boundary

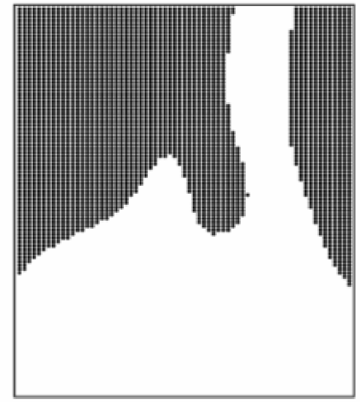
from Tony Liu's thesis (supervised by Kai Ming Ting)

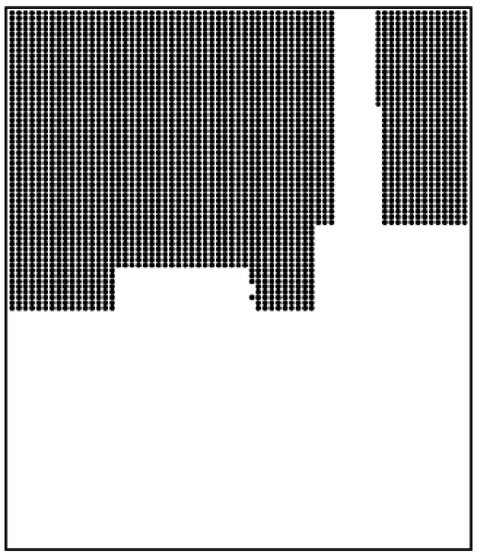


(a) unpruned C4.5

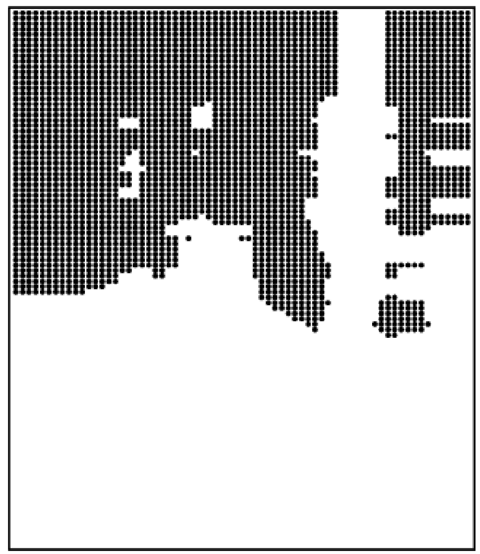


(b) Bagging





(c) Random Forests



(d) Complete-random tree ensemble

RDT looks like the optimal boundary

Strong vs. 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 – Combine Weak Learners

- 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 ☺

Boost weak classifiers to a strong learner

Boosting*

- Principles
  - Boost a set of weak learners to a strong learner
  - Make records currently misclassified more important

- Example
  - Record 4 is hard to classify
  - Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

*[FrSc97]

from P. Tan et al. Introduction to Data Mining.

Boosting

- AdaBoost
  - Initially, set uniform weights on all the records
  - At each round
    - Create a bootstrap sample based on the weights
    - Train a classifier on the sample and apply it on the original training set
    - Records that are wrongly classified will have their weights increased
    - Records that are classified correctly will have their weights decreased
    - If the error rate is higher than 50%, start over
  - Final prediction is weighted average of all the classifiers with weight representing the training accuracy

Boosting

- Determine the weight
  - For classifier *i*, its error is

$$\varepsilon_i = \frac{\sum_{j=1}^{N} w_j \delta(C_i(x_j) \neq y_j)}{\sum_{j=1}^{N} w_j}$$

  - The classifier's importance is represented as:

$$\alpha_i = \frac{1}{2} \ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$

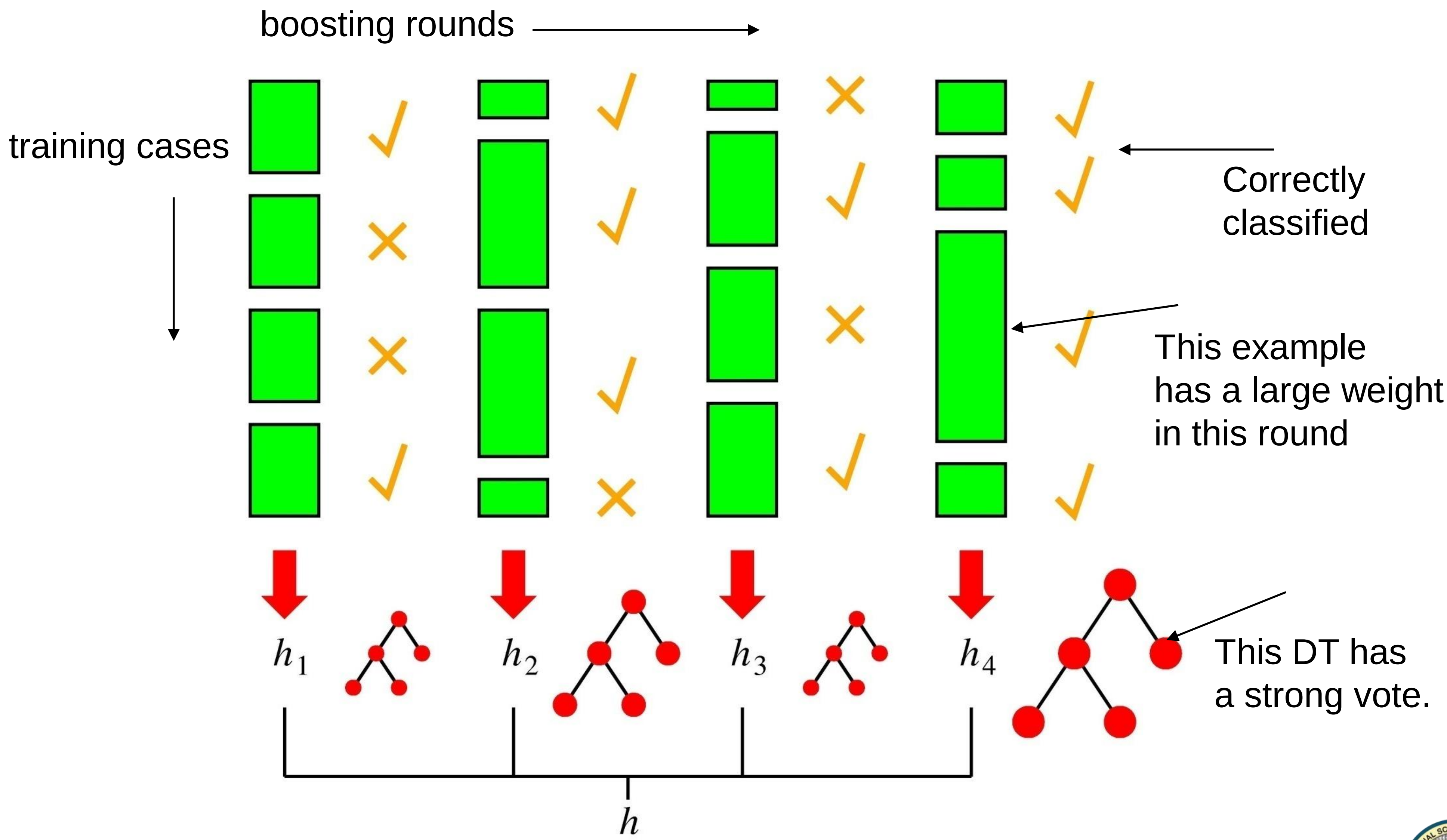
  - The weight of each record is updated as:

$$w_j^{(i+1)} = \frac{w_j^{(i)} \exp\left(-\alpha_i y_j C_i(x_j)\right)}{Z^{(i)}}$$

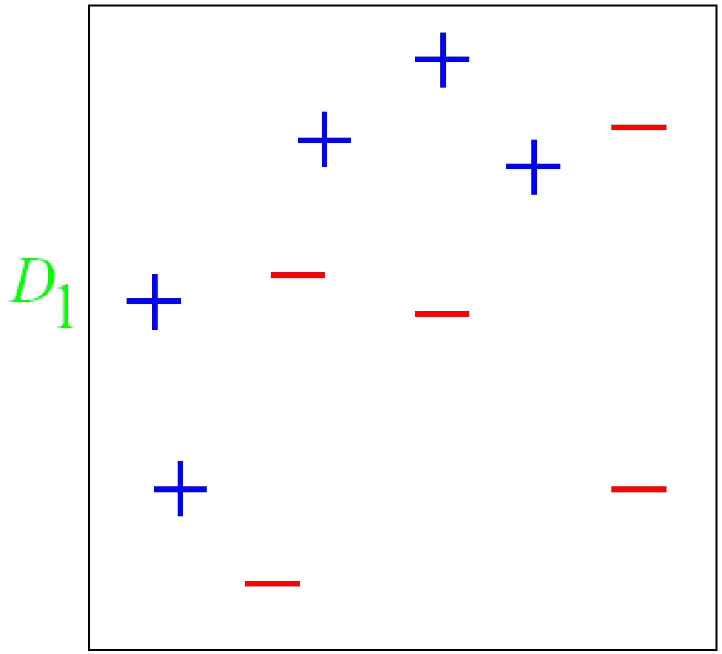
  - Final combination:

$$C^*(x) = \arg\max_y \sum_{i=1}^{K} \alpha_i \delta\left(C_i(x) = y\right)$$

Boosting – An Example



AdaBoost Animation

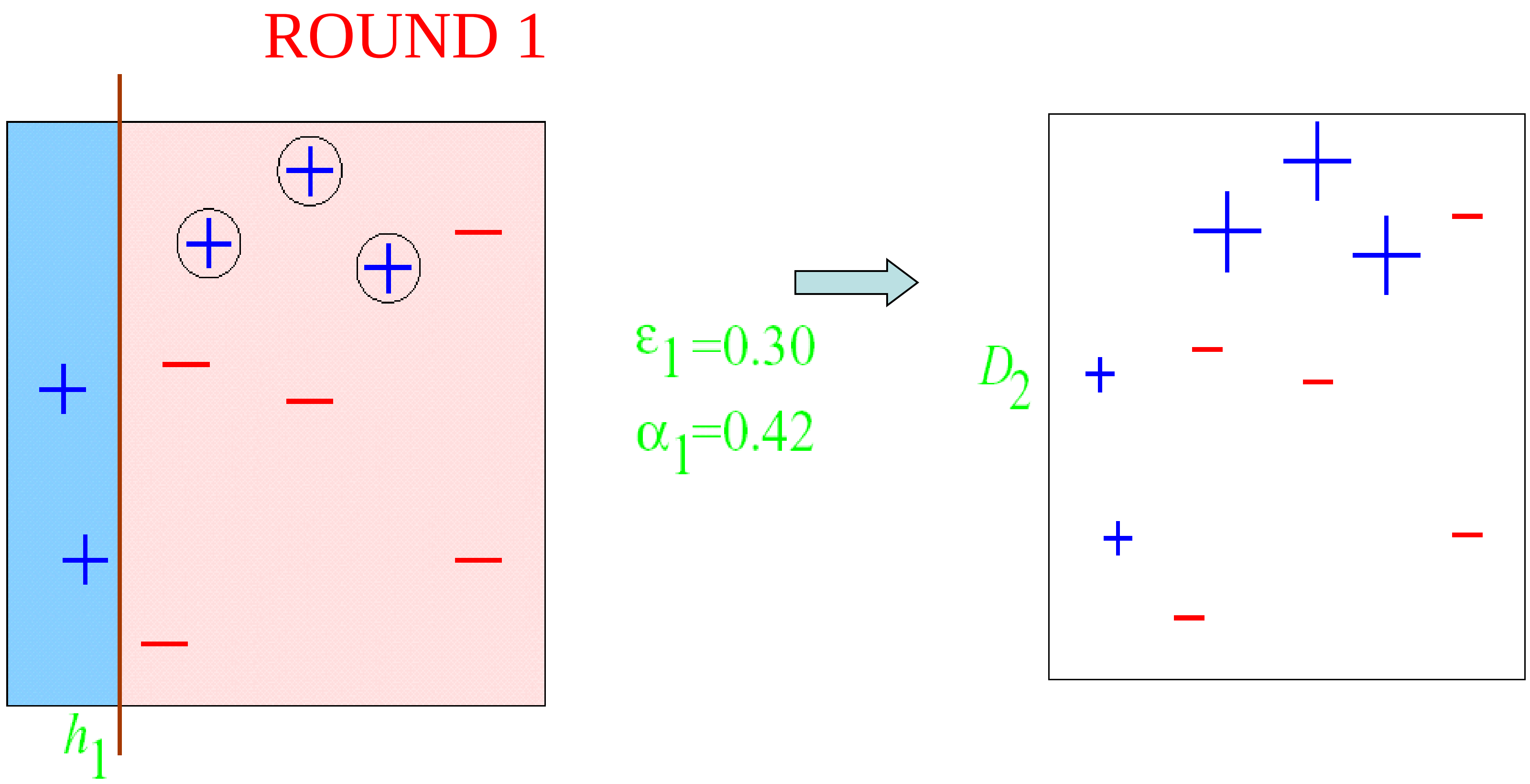


Original Training Set: Equal weights to all training samples

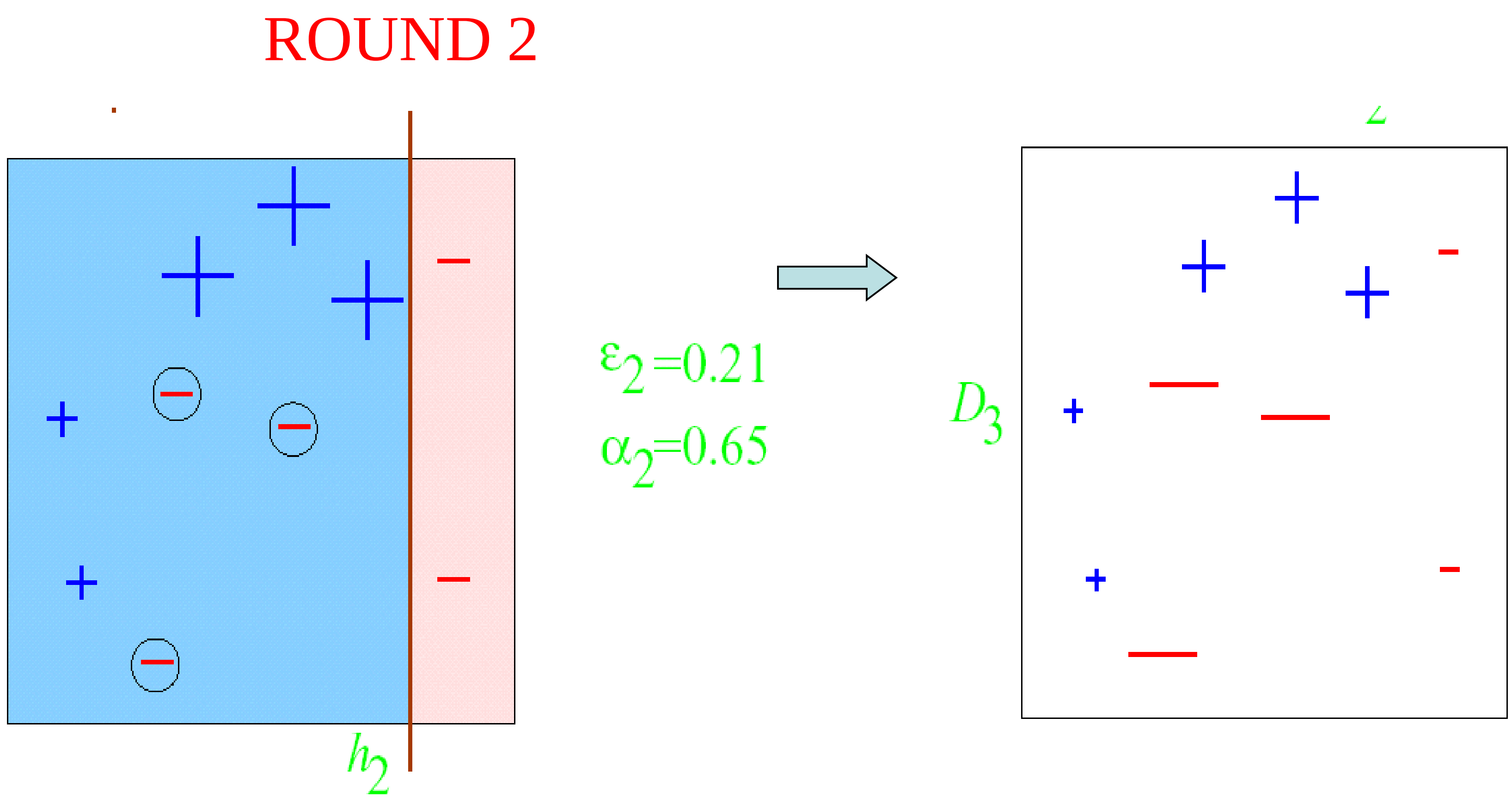
Taken from "**A Tutorial on Boosting**" **by** Yoav Freund and Rob Schapire

AdaBoost Animation

ε = error rate of classifier
α = weight of classifier

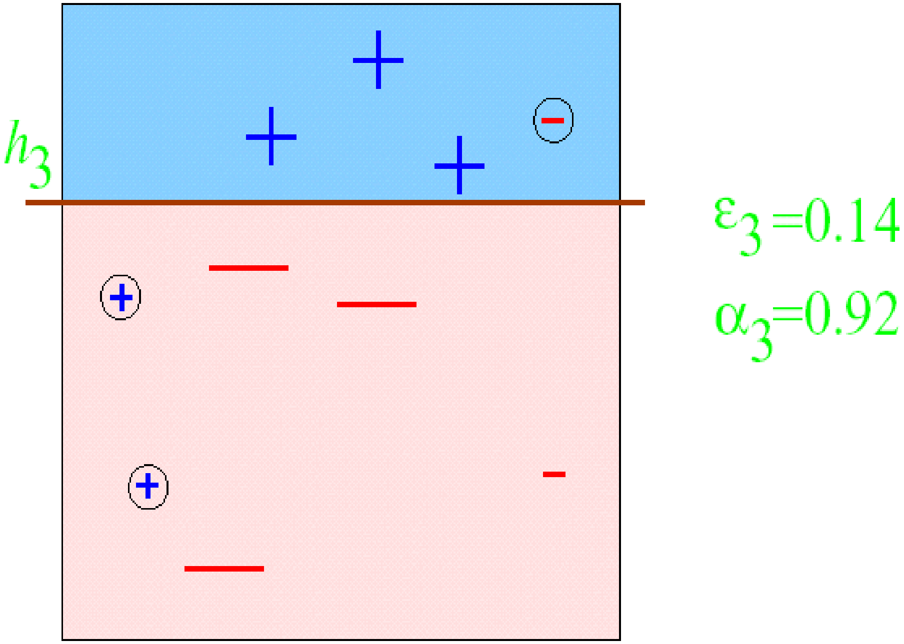


AdaBoost Animation

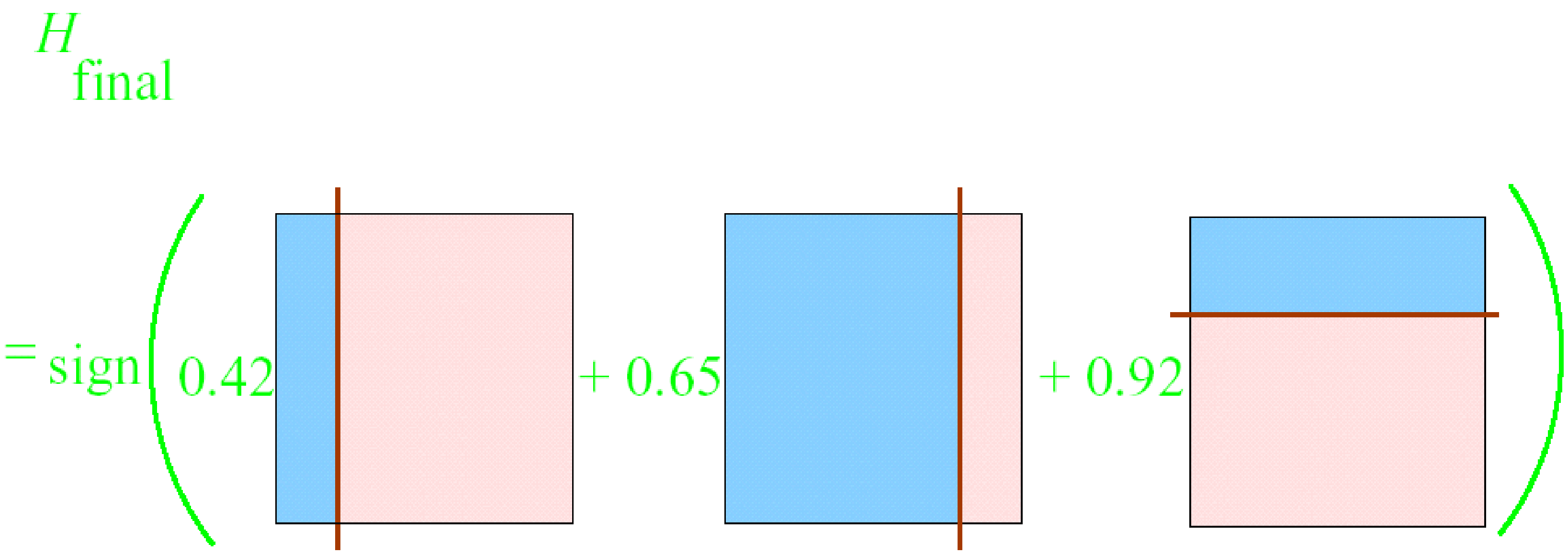


AdaBoost Animation

ROUND 3



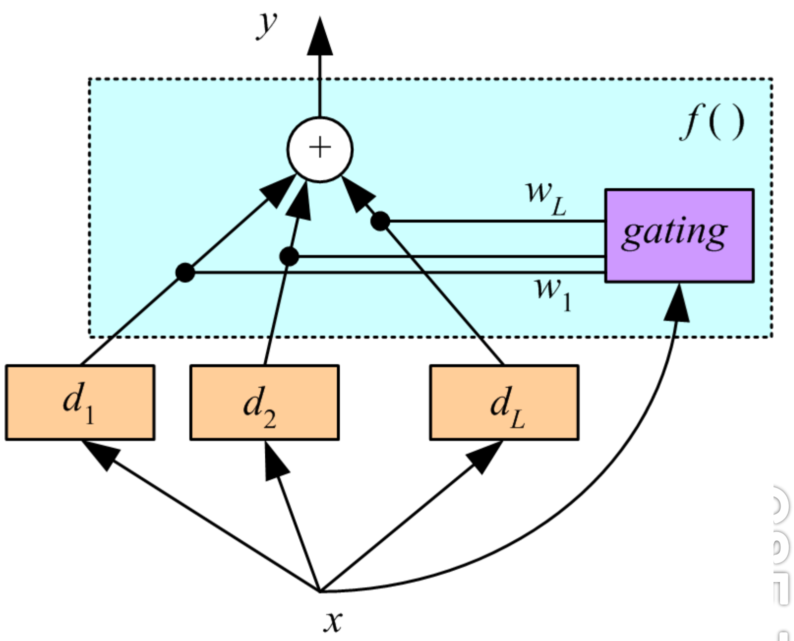
AdaBoost Animation



Mixture of Experts

- Voting where weights are input-dependent (gating)
- Different input regions convered by different learners (Jacobs et al., 1991)

$$y = \sum_{j=1}^{L} w_j d_j$$



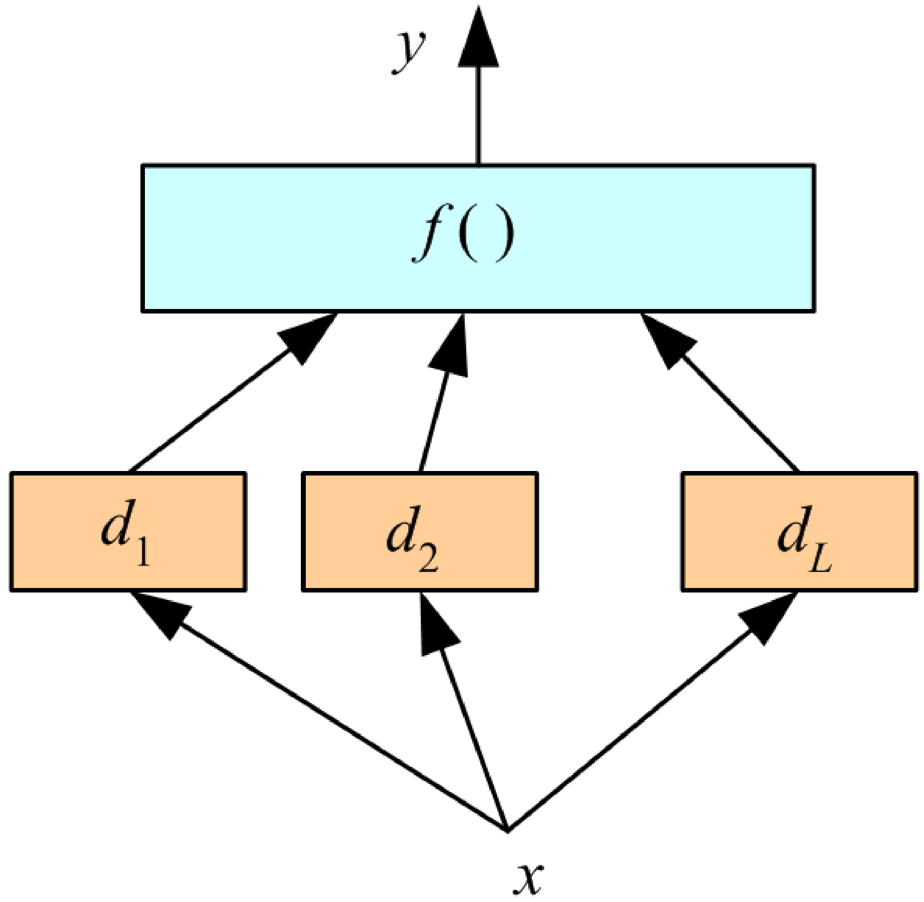
- Gating decides which expert to use
- Need to learn the individual experts as

well as the gating functions $w_i(x)$:

$\Sigma w_j(x) = 1$, for all x

Stacking

- Combiner *f* () is another learner (Wolpert, 1992)



Strengths of AdaBoost

- It has no parameters to tune (except for the number of rounds)
- It is fast, simple and easy to program (relatively)
- It comes with a set of theoretical guarantee (e.g., training error, test error)
- Instead of trying to design a learning algorithm that is accurate over the entire space, we can focus on finding base learning algorithms that only need to be better than random.
- It can identify outliers: i.e. examples that are either mislabeled or that are inherently ambiguous and hard to categorize.

Weakness of AdaBoost

- The actual performance of boosting depends on the data and the base learner.
- Boosting seems to be especially susceptible to noise.
- When the number of outliners is very large, the emphasis placed on the hard examples can hurt the performance.

→ "Gentle AdaBoost", "BrownBoost"

References

- *Survey of Boosting from an Optimization Perspective*. Manfred K. Warmuth and S.V.N. Vishwanathan. ICML'09, Montreal, Canada, June 2009.
- *Theory and Applications of Boosting*. Robert Schapire. NIPS'07, Vancouver, Canada, December 2007.
- *From Trees to Forests and Rule Sets--A Unified Overview of Ensemble Methods*. Giovanni Seni and John Elder. KDD'07, San Jose, CA, August 2007.

Inspire...Educate...Transform.

International School of Engineering

Plot 63/A, 1st Floor, Road # 13, Film Nagar, Jubilee Hills, Hyderabad - 500 033

For Individuals:	+91-9502334561/63 or 040-65743991
For Corporates:	+91-9618483483
Web:	http://www.insofe.edu.in
Facebook:	https://www.facebook.com/insofe
Twitter:	https://twitter.com/Insofeedu
YouTube:	http://www.youtube.com/InsofeVideos
SlideShare:	http://www.slideshare.net/INSOFE
LinkedIn:	http://www.linkedin.com/company/international-school-of-engineering