独立于算法的机器学习

山世光
中国科学院计算技术研究所
sgshan@ict.ac.cn

中国科学院计算技术研究所
Institute of Computing Technology,Chinese Academy of Sciences

课前思考

- 你们学过的模型中哪个最好？为什么？如何比较两个不同模型的优劣？
- Bias和variance分别描述了算法的什么性质？
- 如果有很多可选算法，怎么集成它们？
- 数据多样、规模极大，如何利用好它们？
- 特征维度特别高，如何利用好它们？

参考文献

- 第九章R. Duda, P. Hart, D. Stork, Pattern Classification (Second edition), John Wiley & Sons, New York, USA, 2000

What's in This Chapter?

- Algorithm-Independent by definition
  - to those mathematical foundations that do not depend upon the particular classifier or learning algorithm used.
  - techniques that can be used in conjunction with different learning algorithms, or provide guidance in their use.

Problems to Answer

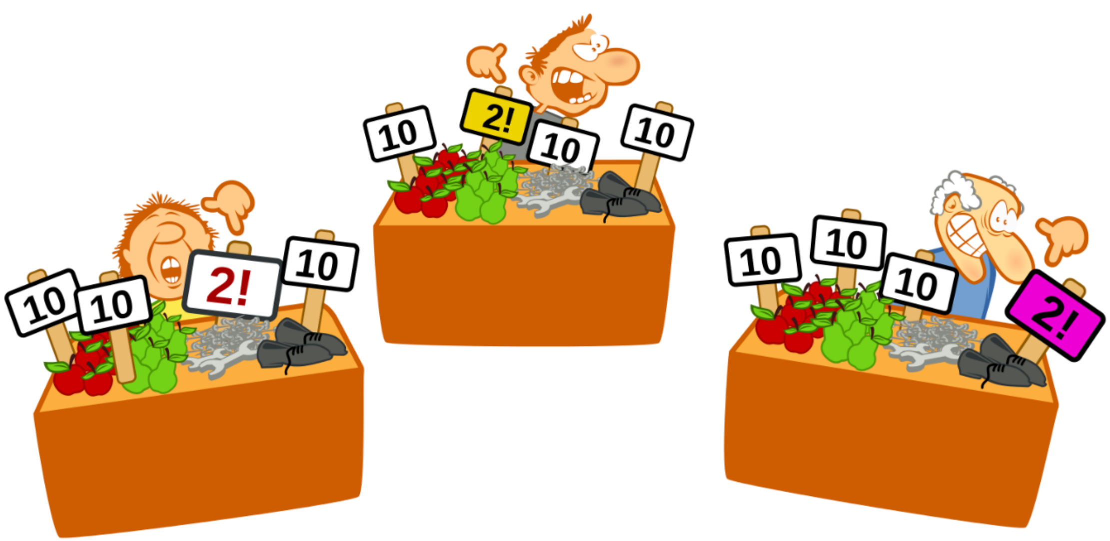
- Many algorithms/techniques in hand
  - Which is the “best”?
  - Are there any reasons to favor one algorithm over another?
  - Can we even find an algorithm that is overall superior to (or inferior to) random guessing?
  - Which one to choose given one problem?

Outline of This Chapter

- Some philosophy in PR/ML
  - No Free Lunch Theorem
  - Ugly Duckling Theorem
  - Minimum Description Length principle
  - Occam’s razor
- Resampling for classifier design
  - Bagging
  - Boosting
  - AdaBoost
  - Active Learning
- Estimating and comparing classifiers
  - Cross validation
- Self-paced learning and curriculum learning

No Free Lunch Theorem

- [Wolpert, 1996] shows that
  - In a noise-free scenario where the loss function is the misclassification rate, if one is interested in *off-training-set error*, there are no a priori distinctions between learning algorithms.



No Free Lunch Theorem

- All algorithms are equivalent, on average, by any of the following measures of error: $E(L|D)$, $E(L|n)$, $E(L|f,D)$, or $E(L|f,n)$, where
  - D = training set;
  - n = number of elements in training set;
  - f = ‘target’ input-output relationships;
  - h = hypothesis (the algorithm's guess for f made in response to D); and
  - L = off-training-set ‘loss’ associated with f and h (‘generalization error’)

Implications of NFL

- There are no i and j such that, for all F,

$$E_i(E|F, n) < E_j(E|F, n)$$

  if all target functions $F(\mathbf{x})$ are equally likely.
- Furthermore, even if we know *D*, averaged over all target functions, no learning algorithm yields an **off-training set error** that is superior to any other.
- All statements of the form "learning/recognition algorithm 1 is better than algorithm 2" are ultimately statements about the **relevant target functions**.
- It is the ***assumptions*** about the learning domains that are relevant.

Ugly Duckling Theorem

- Problem to answer
  - In the absence of prior information, is there a principled reason to judge any two distinct patterns as more or less similar than two other distinct patterns?
- Ugly Duckling Theorem [Watanabe, 1969]

Ugly Duckling Theorem

- Problem to answer
  - In the absence of prior information, is there a principled reason to judge any two distinct patterns as more or less similar than two other distinct patterns?
- Ugly Duckling Theorem [Watanabe, 1969]
  - All things being equal. An ugly duckling is just as similar to a swan as two swans are to each other.
  - 丑小鸭与白天鹅之间的区别和两只白天鹅之间的区别一样大（依赖于分类标准或依据）

Watanabe, Satosi (1969). *Knowing and Guessing: A Quantitative Study of Inference and Information*. New York: Wiley. pp. 376–377.

Ugly Duckling Theorem

- Implications
  - In the absence of assumptions there is no privileged or “best” feature representation.
    - There is no problem-independent or privileged or “best” set of features or feature attributes.
  - Even the apparently simple notion of similarity between patterns is fundamentally based on implicit assumptions about the problem domain

MDL Principle

- Aims at finding some irreducible, smallest representation
- We should minimize the sum of the *model's algorithmic complexity* and the description of the training data with respect to that model, i.e.,

$$K(h,D) = K(h) + K(D \text{ using } h).$$

with K(.) the Kolmogorov complexity, a measure of the **incompressibility**.

MDL Principle

- Example: decision tree classifiers
  - The algorithmic complexity of the model is *proportional to the number of nodes*.
  - The complexity of the data given the model can be expressed in terms of the *weighted sum of the entropies of the data at the leaf nodes*.
  - Thus, if the tree is pruned based on an entropy criterion, it is using MDL.
- Example: Neural Network
  - Deep network compression by pruning
  - Removal of some connections between neurons

MDL Principle

- Theoretically classifiers designed with an MDL principle are guaranteed to converge to the ideal or true model *in the limit of more and more data*.
- The MDL principle states that simple models (smaller $K(h)$) are to be preferred, and thus amounts to a bias to *simplicity*.

Occam's Razor

- Philosophy Principle Occam's Razor
  - "Entities" (or explanations) should not be multiplied beyond necessity.
    如无必要，勿增实体
  - Among competing hypotheses, *the one with the fewest assumptions* should be selected.
  - For PR/ML, NOT use machines that are more complicated than necessary
    - "Necessary" can be determined by the quality of fitting to the training data.

Occam's Razor

- Techniques to avoid overfitting
  - Simplicity
  - Pruning
  - Regularization
  - Inclusion of penalty terms
  - Minimizing a description length…
- Seems conflict with NFL?
  - For a given training error, why do we generally prefer simple classifiers with fewer features and parameters?

Occam's Razor

- Not conflict with NFL, but imply that problems addressed so far *favor simpler classifiers*. Why?
- Evolution bias: strong selection pressure on our pattern recognition apparatuses to be computationally simple
  - Fewer neurons
  - Less time
  - Less energy cost

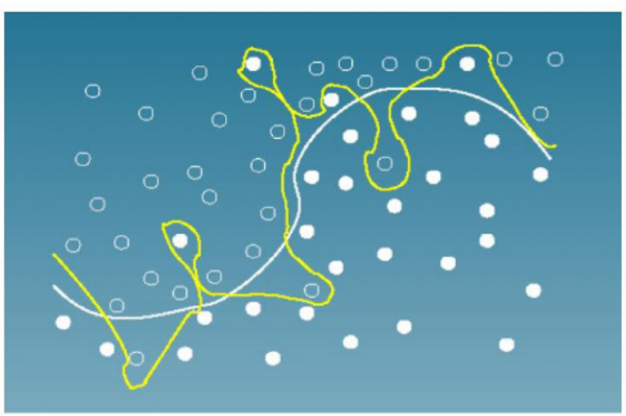
中科院计算所

Bias and Variance Dilemma

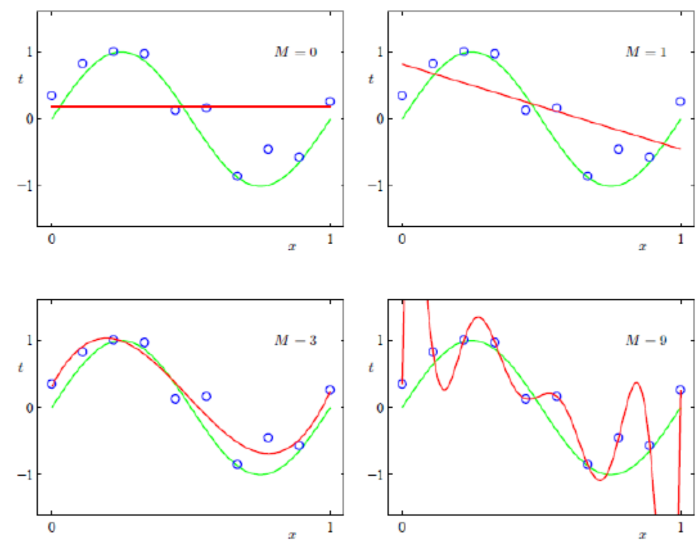
中科院计算所

- Two ways measuring the "match" or "alignment" of the model to the problem
  - **Bias**: accuracy/quality of the match
  - **Variance**: precision/specificity of the match

Overfitting-Classification



Overfitting-Regression



Bias and Variance Dilemma

- **Bias**: model fits training data well,
  - Low bias: favor complex models
- **Variance**: model has capacity to accommodate different testing data
  - Low variance: favor simpler models
- **Discussion**
  - How about deep learning?
  - Why can network be compressed but with accuracy preserved?
  - Why not train the simpler network directly?

Outline of This Chapter

- Some philosophy in PR/ML
  - No Free Lunch Theorem
  - Ugly Duckling Theorem
  - Minimum Description Length principle
  - Occam's razor
- Resampling for classifier design
  - Bagging
  - Boosting
  - AdaBoost
  - Active Learning
- Estimating and comparing classifiers
  - Cross validation

Resampling

- What?
  - Sample a (sub)set from original training set
    - Jackknife (leave one out)
    - **Bootstrap**: randomly selecting n points from the training set D, with replacement
      - Reweighting each points
- Why?
  - Yield a more informative estimate of a general statistic.
  - Good for improve classifiers.

Arcing methods

- Arcing: adaptive reweighting and combining
  - Techniques by reusing or selecting data in order to improve classification
  - Bagging: bootstrap aggregating
    - Independently bootstrap data sets
  - Boosting
    - Dependently bootstrap data sets
  - AdaBoost

Bagging

- Bagging: bootstrap aggregating
  - Proposed by [Breiman, 1996]
  - Derived from bootstrap [Efron, 1993]
- Basic idea
  - Create classifiers using training sets bootstrapped independently (drawn with replacement)
  - Average results of each component classifiers

中科院计算所

Bagging

- Algorithm

  Given a training set

$$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$$

  - 1. Sample *m* sets $D_1, D_2, \ldots, D_m$ of n elements from D (with replacement)
  - 2. Train a component classifier/regression f_i from each D_i
  - 3. The final classifiers is $f(x) = sum/vote(f_1(x), f_2(x), \ldots, f_m(x))$

中科院计算所

Bagging Example (Opitz, 1999)

- Bootstrap data sets
  - With replacement
  - Independently resampled

Original training set	1	2	3	4	5	6	7	8
Training set 1	2	7	8	3	7	6	3	1
Training set 2	7	8	5	6	4	2	7	1
Training set 3	3	6	2	7	5	6	2	2
Training set 4	4	5	1	4	6	4	3	8

Bagging

中科院计算所

- Discussion
  - Why?

Bagging

- Component classifiers selection
  - Generally of the same general form
    - SVM, NN, ANN, tree…
- Effects
  - Improves recognition for *unstable* classifiers since it effectively averages over such **discontinuities**
    - Unstable (related to *high variance*)
      $\boldsymbol{f(D) \approx (D + \Delta D)}$
      - “small” changes in the training data lead to significantly different classifiers and relatively “large” changes in accuracy.
    - 鲁棒性差：易被攻击，$\boldsymbol{f(x) \approx f(x + \Delta x)}$

Arcing methods

- Arcing: adaptive reweighting and combining
  - Techniques by reusing or selecting data in order to improve classification
  - Bagging: bootstrap aggregating
    - Independently bootstrap data sets
  - Boosting
    - Dependently bootstrap data sets
  - AdaBoost

Boosting

- Powerful technique for combining multiple weak "base" learners to form a ***committee*** whose performance can be significantly better than any of the base classifiers
  - Originated from [Schapire, 1989]
- Basic idea
  - **Sequential production of classifiers**: each classifier dependent on the previous one, and focuses on the previous one's failures
  - **Examples incorrectly predicted** in previous classifiers say louder in the next round

A Formal Description of Boosting

- Given training set $X = \{(x_1, y_1), \ldots, (x_n, y_n)\}$
  $y_i \in \{+1, -1\}$ is the label of instance x_i
- for t = 1,...,T:
  - Construct a **new** distribution D_t from X
  - Find weak classifier

$$h_t : X \to \{+1, -1\}$$

  with small error ε_t on D_t:

$$\varepsilon_t = Pi_{\sim D_t}[h_t(x_i) \neq y_i]$$

  - Output final classifier H_{final}=weighted sum(h_t)

Example Boosting Setting

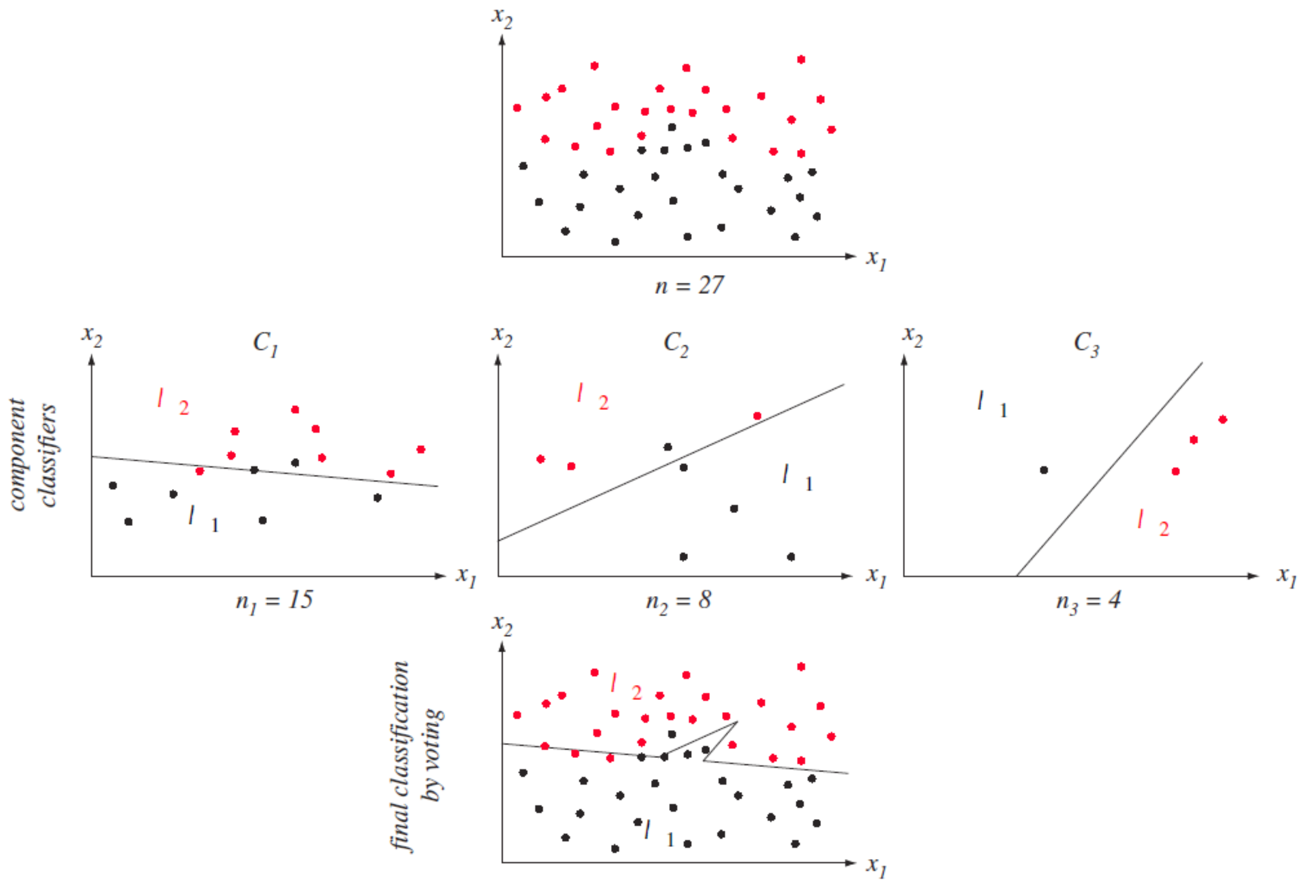
中科院计算所

- $D_1 = randomly\ select\ a\ subset\ of\ X$
- $D_2 = select\ from\ X/D_1,$ $\{half\ correctly\ classified\ by\ h_1\} + \{half\ incorrectly\ classified\ by\ h_1\}$
- $D_3 = \{x_i \in (X/D_1 \cup D_2)\ and\ h_1(x_i) \neq h_2(x_i)\}$
- The final classifier:

$$h_{\mathrm{final}}(\mathrm{x}) = \begin{cases} h_1(x); & if\ h_1(x) == h_2(x) \\ h_3(x); & otherwise \end{cases}$$

Example Boosting Setting

- Component classifiers: LMS
- Sampling: basic boosting procedure



中科院计算所

Many Variations

- AdaBoost.M1, AdaBoost.MR, FilterBoost, GentleBoost, GradientBoost, MadaBoost, LogitBoost, LPBoost, MultiBoost, RealBoost, RobustBoost, …

From Bagging to Boosting

- Base classifiers are trained in sequence
- Each base classifier trained using a weighted form of the dataset
  - Weighting coefficient depends on the performance of the previous classifiers
    - Points **misclassified** by previous classifiers are **given more weights** in training next classifier
- Decisions are combined using a weighted majority voting scheme

Arcing methods

- Arcing: adaptive reweighting and combining
  - Techniques by reusing or selecting data in order to improve classification
  - Bagging: bootstrap aggregating
    - Independently bootstrap data sets
  - Boosting
    - Dependently bootstrap data sets
  - AdaBoost

AdaBoost

- Proposed by [Freund & Schapire'95]:
  - Strong practical advantages over previous boosting algorithms
  - With amazing generalization ability
- Answer the open problem
  - An open problem [Kearns & Valiant, STOC'89]: **"weakly learnable" ?= "strongly learnable"**
  - In intuitive words, whether a "weak" learning algorithm that works just slightly better than random guess can be "boosted" into an arbitrarily accurate "strong" learning algorithm!

The Born of AdaBoost

- Amazingly, in 1990 Schapire proves that the answer is “yes”. More importantly, the proof is a construction! This is the first Boosting algorithm
- In 1993, Freund presents a scheme of combining weak learners by majority voting in PhD thesis at UC Santa Cruz

However, these algorithms are not practical!

- Later, at AT&T Bell Labs, Freund & Schapire published the 1997 journal paper (the work was reported in EuroCOLT’95), which proposed the AdaBoost algorithm, a practical algorithm.

From the slides of Prof. Zhihua Zhou.

New Resampling Mechanism

- Given training set $X = \{(x_1, y_1), \ldots, (x_m, y_m)\}$
  $y_i \in \{+1, -1\}$ is the label of instance x_i
- $D_1(i) = \frac{1}{m}$; given D_t and h_t, then

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \times \begin{cases} e^{-\alpha_t}, if\ yi = ht(xi) \\ e^{\alpha_t},\ \ if\ yi \neq h_t(xi) \end{cases} = \frac{D_t(i)}{Z_t} \exp(-\alpha_t y_i h_t(xi))$$

where z_t is a normalization factor, and

$\alpha_t = \frac{1}{2} \ln\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) > 0$, with $\varepsilon_t = Pi_{\sim D_t}[h_t(x_i) \neq y_i]$<0.5

- Final classifier

$$H_{final}(x) = sign\left(\sum_t \alpha_t h_t(x)\right)$$

AdaBoost Algorithm

- Weights of misclassified samples are increase in (t+1)th iteration.
  - given training set $(x_1, y_1), \ldots, (x_m, y_m)$ where $x_i \in X$, $y_i \in \{-1, +1\}$
  - initialize $D_1(i) = 1/m$ $(\forall i)$
  - for $t = 1, \ldots, T$:
    - train weak classifier $h_t : X \to \{-1, +1\}$ with error $\epsilon_t = \Pr_{i \sim D_t}[h_t(x_i) \neq y_i]$
    - $\alpha_t = \frac{1}{2} \ln \left(\frac{1 - \epsilon_t}{\epsilon_t} \right)$
    - update $\forall i$:

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} \exp\left(-\alpha_t y_i h_t(x_i)\right)$$

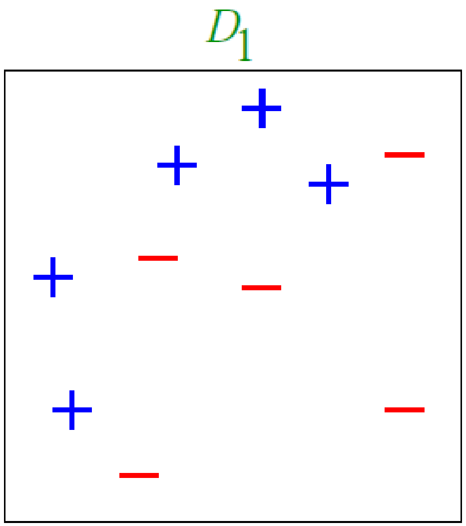
where Z_t = normalization factor

  - $H_{\text{final}}(x) = \text{sign}\left(\sum_{t=1}^{T} \alpha_t h_t(x) \right)$

Modified from the tutorial of Freund & Schapire.

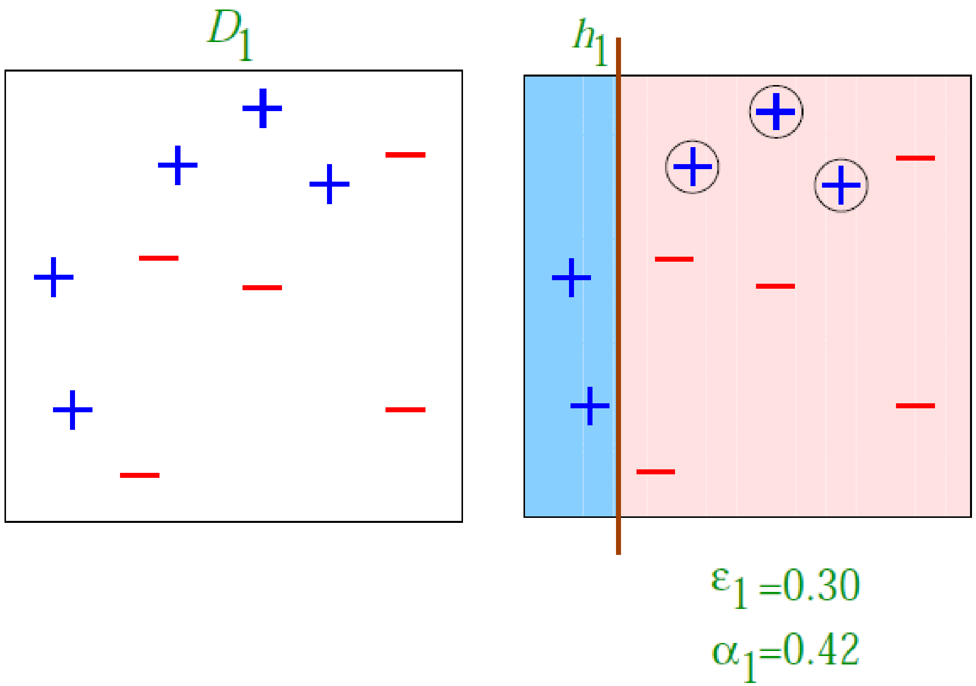
Toy Example of AdaBoost

- Initialize



Toy Example of AdaBoost

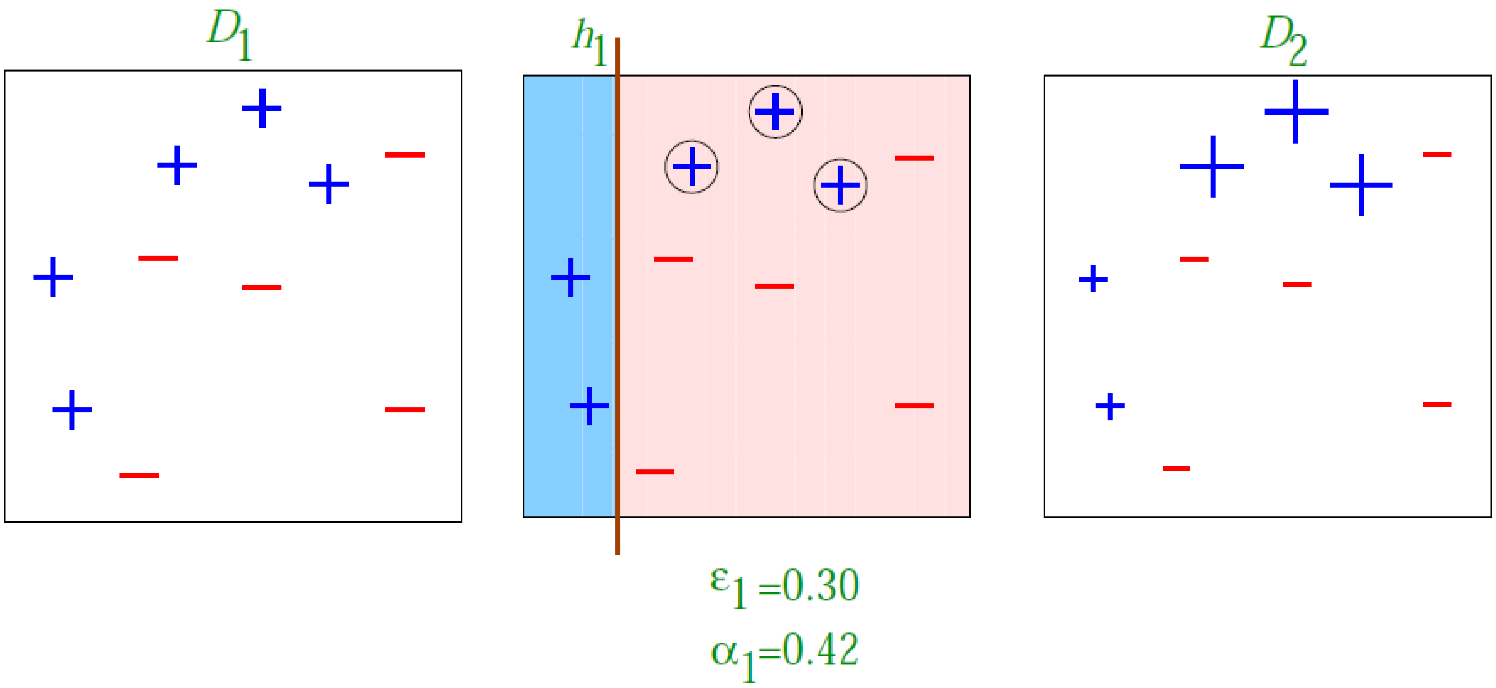
- Round 1



Modified from the tutorial of Freund & Schapire.

Toy Example of AdaBoost

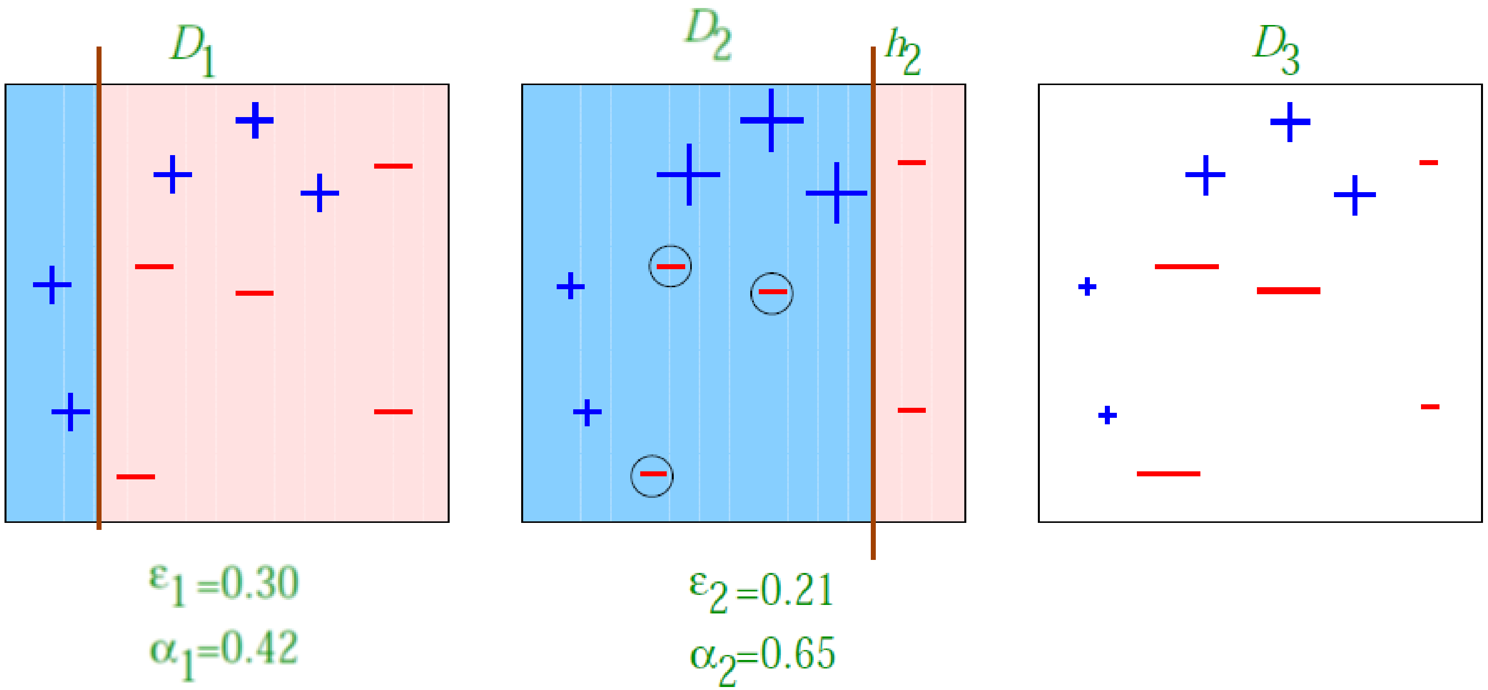
- Round 1



Modified from the tutorial of Freund & Schapire.

Toy Example of AdaBoost

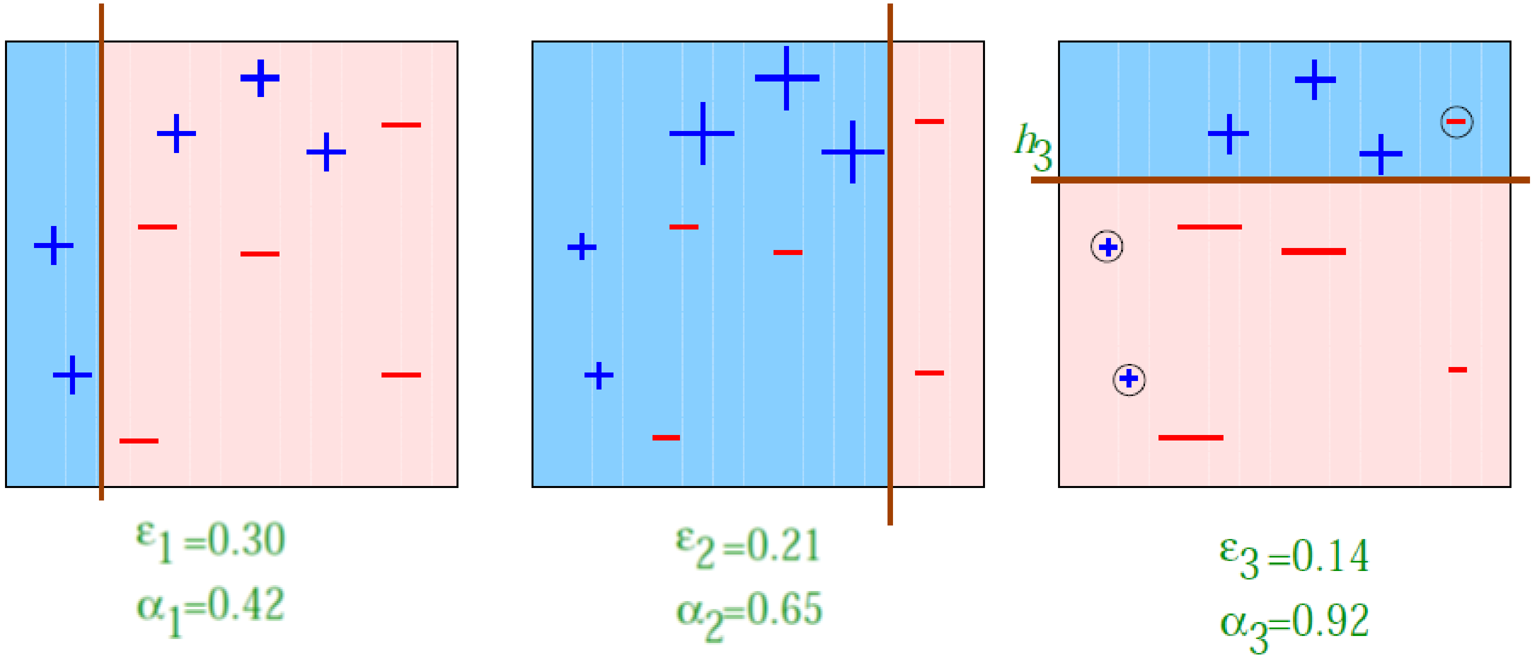
- Round 2



Modified from the tutorial of Freund & Schapire.

Toy Example of AdaBoost

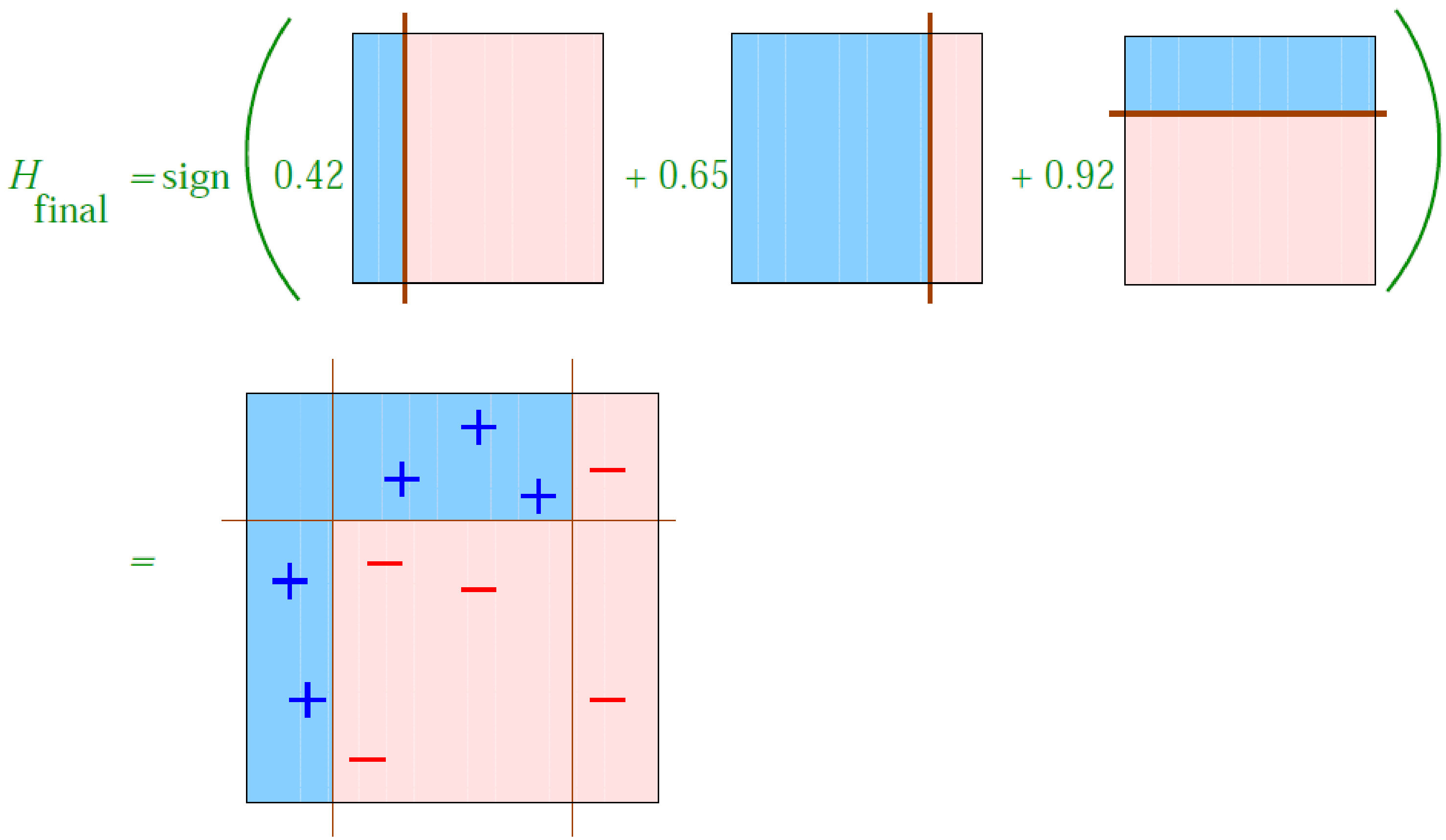
- Round 3



Modified from the tutorial of Freund & Schapire.

Toy Example of AdaBoost

- Final Strong Classifier



Modified from the tutorial of Freund & Schapire.

AdaBoost Training Error

中科院计算所

- Theorem:
  - write ϵ_t as $\frac{1}{2} - \gamma_t$ [$\gamma_t =$ "edge"]
  - then

$$
\begin{aligned}
\text{training error}(H_{\text{final}}) &\leq \prod_t \left[2\sqrt{\epsilon_t(1-\epsilon_t)}\right] \\
&= \prod_t \sqrt{1-4\gamma_t^2} \\
&\leq \exp\left(-2\sum_t \gamma_t^2\right)
\end{aligned}
$$

- so: if $\forall t: \ \gamma_t \geq \gamma > 0$
  then training error$(H_{\text{final}}) \leq e^{-2\gamma^2 T}$

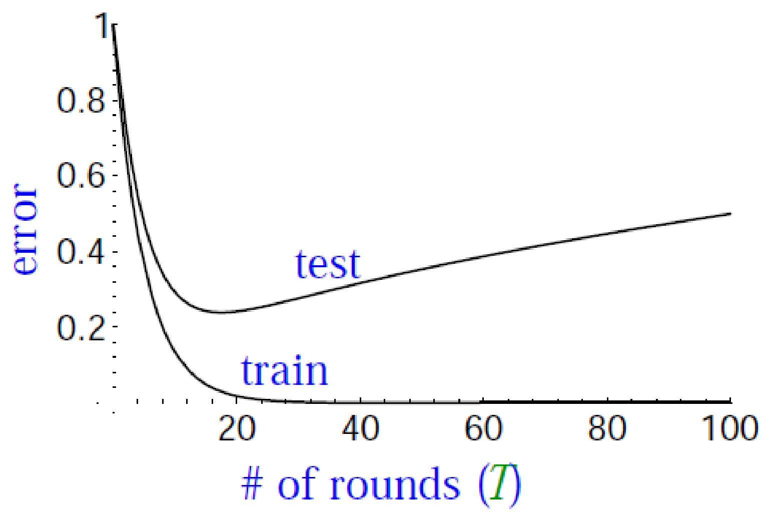
Decrease with increase of T

- AdaBoost is adaptive:
  - does not need to know γ or T a priori

Modified from the tutorial of Freund & Schapire.

How Will Test Error Behave??

- Expect (a first guess)
  - Training error to continue to drop(or reach 0)
  - Test error increases when H_{final} becomes "too complex"
    - Occam's razor
    - Overfitting: hard to know when to stop training



Modified from the tutorial of Freund & Schapire.

Theoretically…

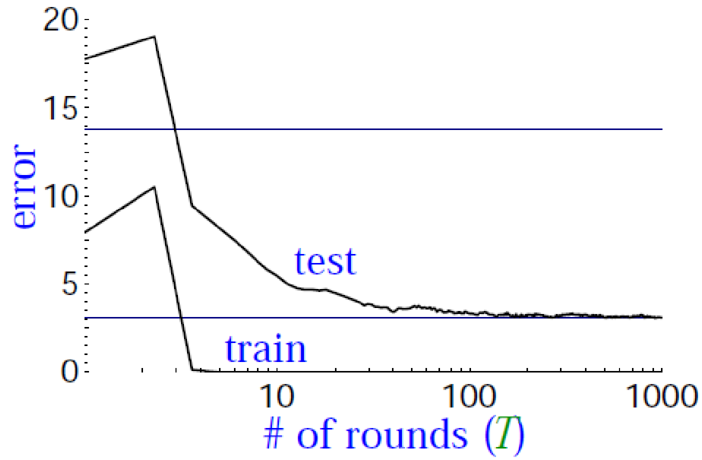
- With high probability

$$\text{generalization error} \leq \text{training error} + \tilde{O}\left(\sqrt{\frac{dT}{m}}\right)$$

- bound depends on
  - m = # training examples
  - d = “complexity” of weak classifiers, VC-dimension
  - T = # rounds
- Generalization error = E[test error]
- **Should overfit with T increase…**

Modified from the tutorial of Freund & Schapire.

Actually



(boosting C4.5 on "letter" dataset)

- Test error does **not** increase, even after 1000 rounds
  - Total size > 2,000,000 nodes
- Test error continues to drop even after training error is zero!

	# rounds		
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1

- **Occam's razor wrongly predicts "simpler" rule is better?**

Modified from the tutorial of Freund & Schapire.

Margin Theory Explanation

Based on the concept of margin, Schapire et al. [1998] proved that, given any threshold $\theta > 0$ of margin over the training data D, with probability at least $1-\delta$, the generalization error of the ensemble $\epsilon_{\mathcal{D}} = P_{\boldsymbol{x}\sim\mathcal{D}}(f(\boldsymbol{x}) \neq H(\boldsymbol{x}))$ is bounded by

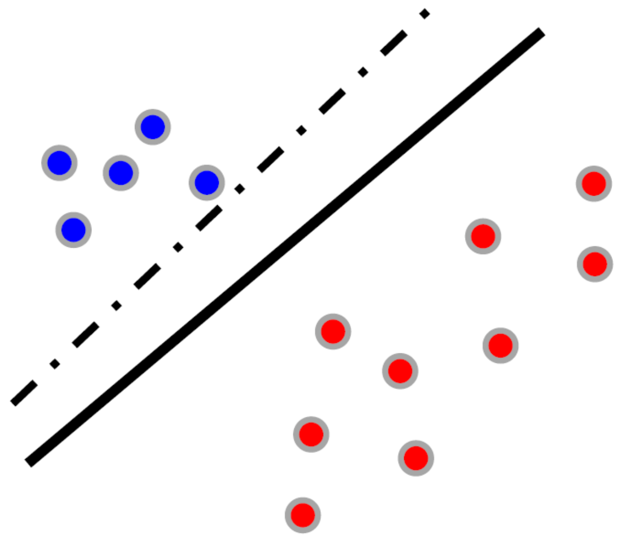
$$\epsilon_{\mathcal{D}} \leq P_{\boldsymbol{x}\sim D}(f(\boldsymbol{x})H(\boldsymbol{x}) \leq \theta) + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2} + \ln\frac{1}{\delta}}\right)$$

$$\leq 2^T \prod_{t=1}^{T} \sqrt{\epsilon_t^{1-\theta}(1-\epsilon_t)^{1+\theta}} + \tilde{O}\left(\sqrt{\frac{d}{m\theta^2} + \ln\frac{1}{\delta}}\right)$$

- This bound implies that, when other variables are fixed, **the larger the margin over the training data, the smaller the generalization error**

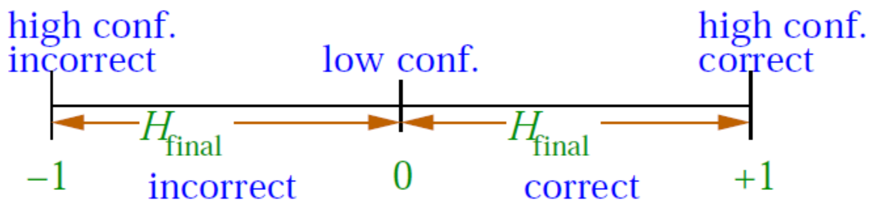
Margin Theory Explanation

- Why AdaBoost tends to be resistant to overfitting? the margin theory answers:
  - It can increase the **ensemble margin** even after the training error reaches zero!



Margin Theory Explanation

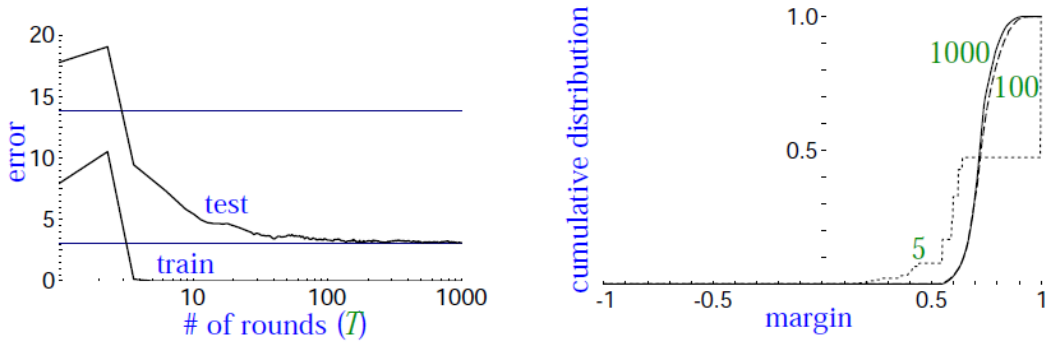
- Key ideas
  - Training error only measures whether classifications are right or wrong
  - Should also consider **confidence** of classifications
  - Recall: H_{final} is weighted majority vote of weak classifiers
- Measure confidence by **margin** = strength of the vote = (weighted fraction voting correctly) −(weighted fraction voting incorrectly)



From the tutorial of Freund & Schapire.

Empirical Evidence: Margin Distribution

- Margin distribution
  - Cumulative distribution of margins of training examples



	# rounds		
	5	100	1000
train error	0.0	0.0	0.0
test error	8.4	3.3	3.1
% margins ≤ 0.5	7.7	0.0	0.0
minimum margin	0.14	0.52	0.55

Modified from the tutorial of Freund & Schapire.

Theoretical Evidence

- Theorems
  - Larger margins ⇒ better bound on generalization error (independent of number of rounds)
  - Boosting tends to increase margins of training examples (**given weak learning assumption**)
- Tighter bound with margin distribution
  - Minimum margin, media margin, average margin, margin variance…

More…

- Predicts good generalization with no overfitting if:
  - weak classifiers not too complex relative to size of training set
  - weak classifiers have large edges (implying large margins)
- For example
  - Boosting decision trees resistant to overfitting since trees often have large edges and limited complexity
- Overfitting may occur if:
  - Overly complex weak classifiers
  - Small edges (underfitting)

Practical Advantages of AdaBoost

- Very fast, simple and easy to program
- Few parameters to tune (except T)
- Flexible
  - can combine with (m)any learning algorithm
- Little prior knowledge needed about weak learner
  - Provably effective, provided can consistently find rough rules of thumb
  - Shift in mindset — goal now is merely to find classifiers barely better than random guessing
- Versatile
  - can use with data that is textual, numeric, discrete, etc.
  - has been extended to learning problems well beyond binary classification

Modified from the tutorial of Freund & Schapire.

Disadvantages of AdaBoost

- Performance of AdaBoost depends on data and weak learner.
- Consistent with theory, AdaBoost can fail if
  - weak classifiers are too complex
    - Due to possible overfitting
  - weak classifiers too weak ($\gamma_t \rightarrow 0$ too quickly)
    - Due to underfitting
    - low margins $\rightarrow$ overfitting
- empirically, AdaBoost seems especially susceptible to uniform noise

Modified from the tutorial of Freund & Schapire.

AdaBoost for Face Detection

（separate slides）

Outline of This Chapter

- Some philosophy in PR/ML
  - No Free Lunch Theorem
  - Ugly Duckling Theorem
  - Minimum Description Length principle
  - Occam’s razor
- Resampling for classifier design
  - Bagging
  - Boosting
  - AdaBoost
  - Active Learning
- Estimating and comparing classifiers
  - Cross validation

中科院计算所

Active Learning

- Problem to address
  - Semi-supervised learning
- a.k.a
  - Learning with query; Interactive learning
- Method
  - Human in the loop to label "self-proposed" unlabeled informative samples
  - How to self-proposed samples?
    - pattern that the current classifier is least certain
    - pattern that yields the greatest disagreement among the committee

Outline of This Chapter

- Some philosophy in PR/ML
  - No Free Lunch Theorem 没有最好的算法/学习器
  - Ugly Duckling Theorem 没有最优的特征
  - Minimum Description Length principle 描述越短越好
  - Occam's razor 越简单越好
- Resampling for classifier design
  - Bagging
  - Boosting
  - AdaBoost
  - Active Learning
- Estimating and comparing classifiers
  - Cross validation

Cross Validation

中科院计算所

- Hard if not impossible to theoretically assess and compare classifiers
- Heuristic method
  - Cross validation
    - →*m*-fold cross-validation
  - Jackknife (leave-one-out)

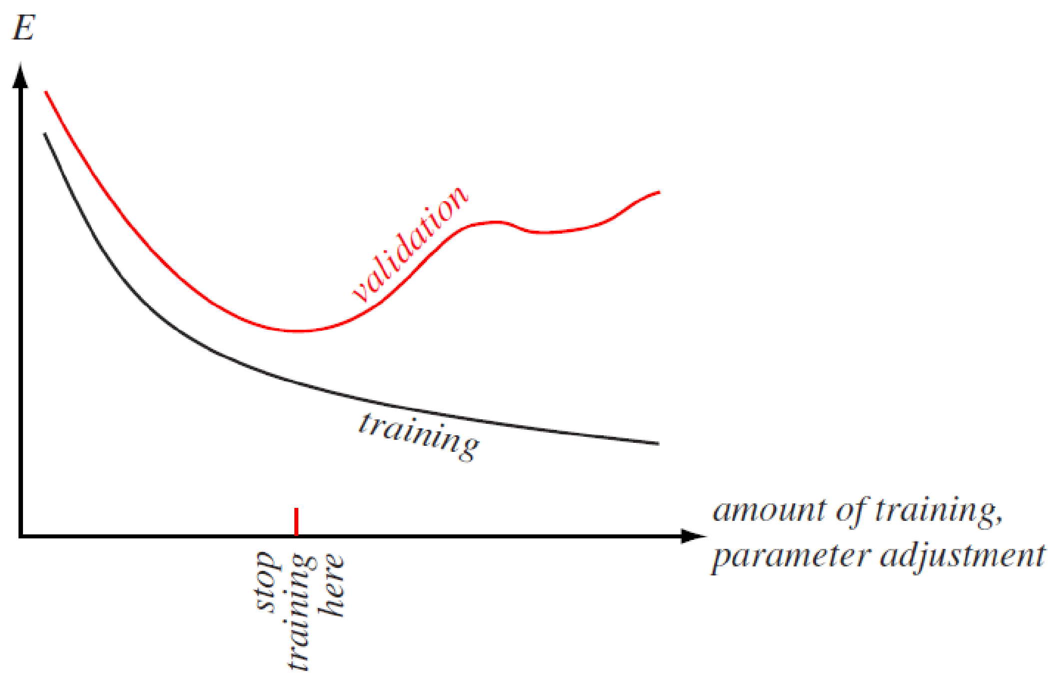
Cross Validation

中科院计算所

- Wrong!!
  - Turing parameters on the testing set
  - Validating on the training set
- Cross-validation
  - Randomly split the set of labeled training samples *D* into two parts
    - one as the traditional training set for adjusting model parameters in the classifier
    - The other set — the *validation set* — is used to estimate the generalization validation error

Cross Validation

- Typically, the error on the validation set decreases, but then increases
  - Indication that the classifier may be overfitting the training data
- Training or parameter adjustment is stopped at the first minimum of the validation error.



Cross Validation

- How to split the original training set?
  - heuristics for choosing the portion γ of *D* to be used as a validation set ($0 < \gamma < 1$)
    - Generally a small portion for validation $\gamma < 0.5$
    - If a classifier has a large number of free parameters or degrees of freedom $e.g.$ γ=0.1
- m-fold Cross Validation
  - training set is randomly divided into m disjoint sets of equal size n/m
  - One set for validation, the other m-1 sets for training. And repeat m times…
  - If m==n, jackknife (leave-one-out)

延伸阅读

- Boosting & Deep Learning
  - MoE
  - Furong Huang 1 Jordan T. Ash 2 John Langford 3 Robert E. Schapire. Learning Deep ResNet Blocks Sequentially using Boosting Theory. ICML2018
  - M Moghimi, et al., Boosted Convolutional Neural Networks, BMVC2016

中科院计算所

中科院计算所

谢谢！