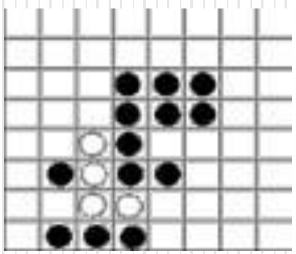
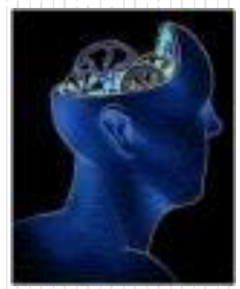
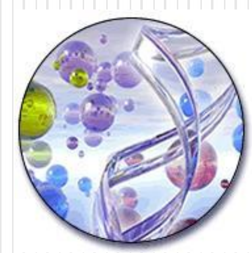
Welcome to

Introduction to Machine Learning!







Topic 14: Computational Learning Theory

Xiaolin Hu
xlhu@tsinghua.edu.cn

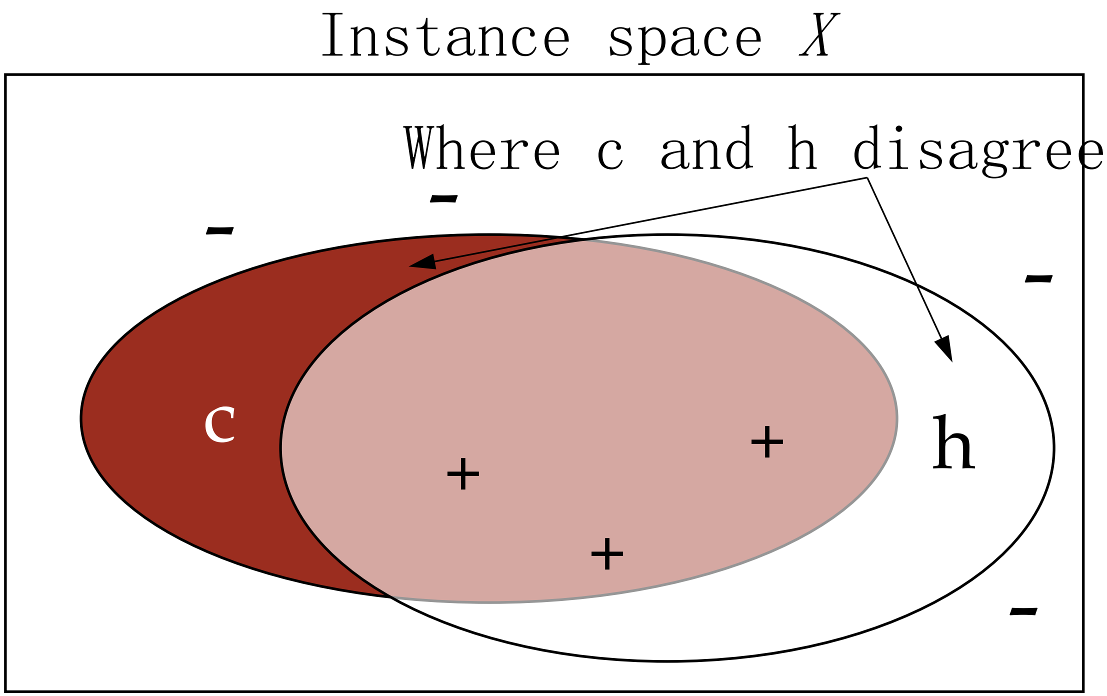
Computational learning theory

- Sample complexity
- Probably approximated correct (PAC) learning
  (可能近似正确学习)
- Vapnik-Chervonenkis Dimension
- Mistake bounds　(出错界限)

Sample complexity

How many training examples are sufficient to learn the target concept?

True Error of a Hypothesis



- Definition: True error: $error_{\mathcal{D}}(h)$ (with respect to target concept c and distribution $\mathcal{D}$)

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

  - is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$

Version space

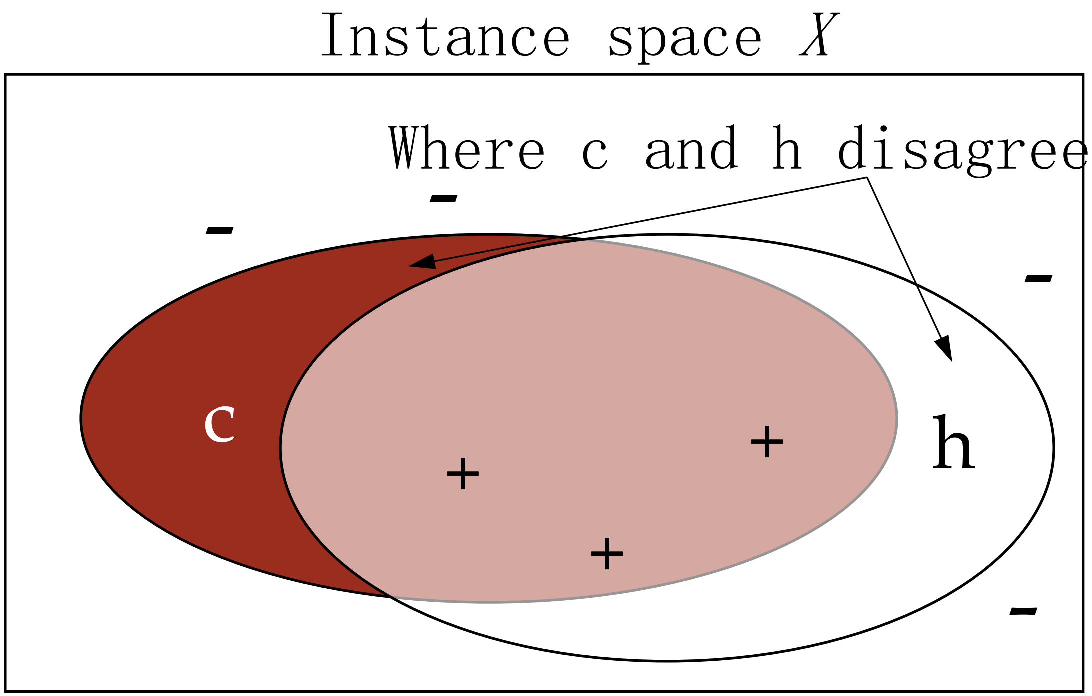
- A hypothesis *h* is consistent with a set of training examples *D* of target concept *c* if and only if $h(x)=c(x)$ for each training example $\langle x, c(x) \rangle$ in *D*.

$$Consistent(h, D) \equiv (\forall \langle x, c(x) \rangle \in D)\ h(x) = c(x)$$

- **The version space ($VS_{H,D}$)** with respect to hypothesis space *H* and training examples *D*, is the subset of hypotheses from *H* consistent with all training examples in *D*.

$$VS_{H,D} \equiv \{h \in H | Consistent(h, D)\}$$

True Error of a Hypothesis



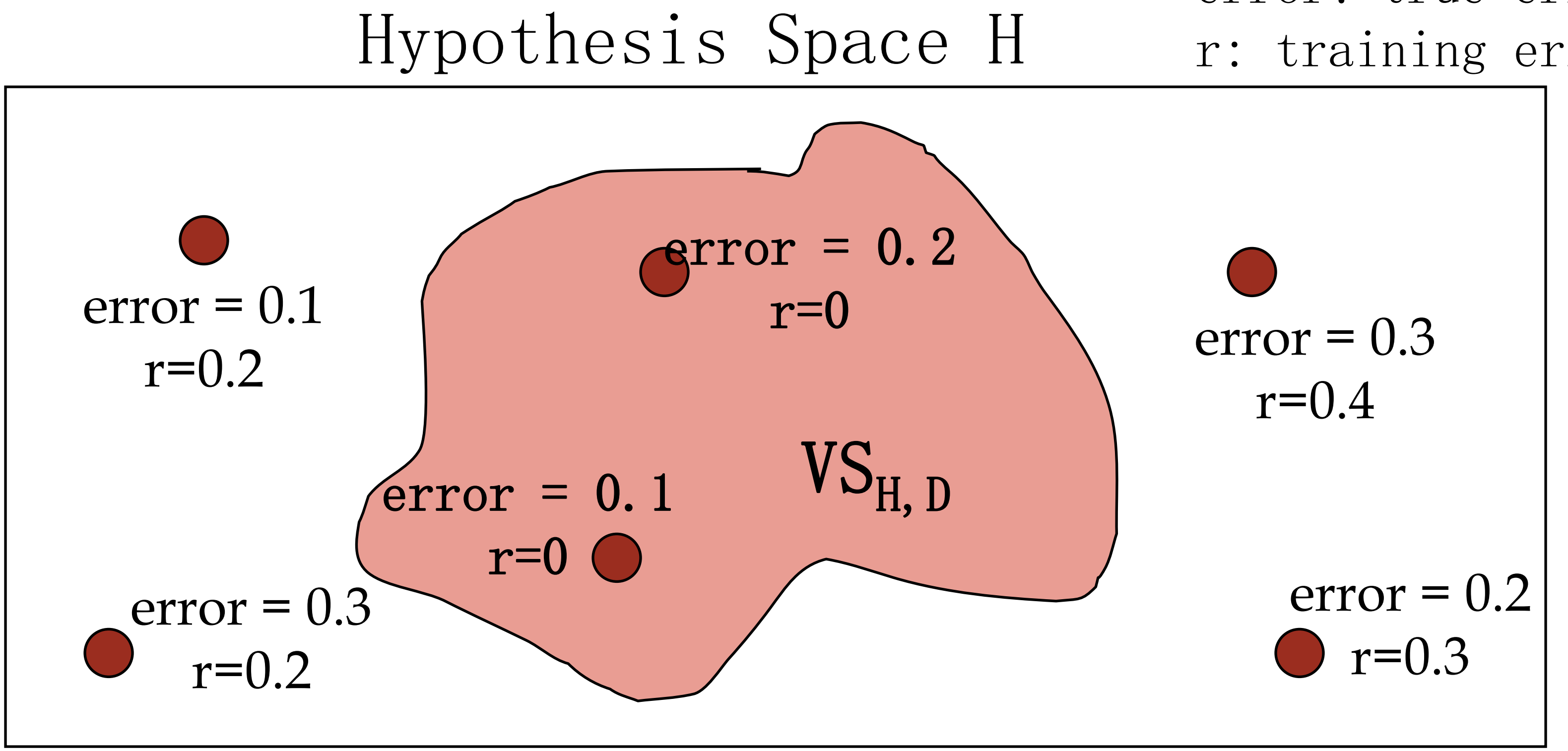
Our concern:
- Can we bound the true error of h given the training error of h?
- First consider when training error of h is zero (i.e. $h \in VS_{H,D}$)

- Definition: True error: $error_{\mathcal{D}}(h)$ (with respect to target concept c and distribution $\mathcal{D}$)
  - is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

Exhausting the Version Space

error: true error
r: training error



- Definition: ε-exhausted (ε详尽)

  A version space $VS_{H,D}$ is said to be ε-exhausted with respect to c and $\mathcal{D}$, if every hypothesis h in $V_{H,D}$ has true error less than ε with respect to c and $\mathcal{D}$.

$$(\forall h \in VS_{H,D})\ \ error_{\mathcal{D}}(h) < \varepsilon$$

How many examples will ε – exhaust the $VS_{H,D}$?

Theorem ε – exhausting the version space (version space的 ε –详尽化)

- If the hypothesis space H is finite, and D is a sequence of $m \geq 1$ independent randomly drawn examples of some target concept c
- Then for any $0 \leq \varepsilon \leq 1$, the probability that the version space $VS_{H,D}$ is **not** ε–exhausted (with respect to c) is **less than** $|H|e^{-\varepsilon m}$

- Proof of the theorem :

[1] $VS_{H,D}$ is not ε-exhausted → exists at least 1 hypo. :

- The hypothesis' true error is ≥ ε, and
- It's in the $VS_{H,D}$, so it is consistent with *m* training instances

the probability is ≤ $(1-\varepsilon)^m$

[2] Supposing there are k hypotheses with true error ≥ ε

→ The probability that at least one of them is consistent with ***m*** training instances is ≤ $k(1-\varepsilon)^m$

[3] $k \leqslant |H|$ && when $0 \leqslant \varepsilon \leqslant 1$, $(1-\varepsilon) \leqslant e^{-\varepsilon}$

(Taylor expansion: $e^{-x} = 1-x + x^2/2! - x^3/3! + \cdots\cdots$)

[4] $k(1-\varepsilon)^m \leq |H|(1-\varepsilon)^m \leq |H| e^{-\varepsilon m}$

How many examples will ε − exhaust the $VS_{H,D}$?

Theorem ε − exhausting the version space (version space的 ε − 详尽化)

- If the hypothesis space H is finite, and D is a sequence of $m \geq 1$ independent randomly drawn examples of some target concept c
- Then for any $0 \leq \varepsilon \leq 1$, the probability that the version space $VS_{H,D}$ is **not** ε−exhausted (with respect to c) is **less than** $|H|e^{-\varepsilon m}$

- Interesting! This **bounds** the probability that any consistent learner will output a hypothesis *h* with $error_{\mathcal{D}}(h) \geq \varepsilon$
- If we want this probability to be below δ $(0 \leq \delta \leq 1)$,

$$|H|e^{-\varepsilon m} \leq \delta \qquad then: m \geq \frac{1}{\varepsilon}(\ln|H| + \ln|1/\delta|)$$

How many training examples are sufficient to assure that any consistent hypothesis will be probably (with probability $1-\delta$) approximately correct (within error ε) .

——— PAC Learning 可能近似正确学习

PAC Learning Framework
可能近似正确学习

PAC learning -- "approximately" "probably"

- $error_{\mathcal{D}}(h)$ cannot be 0 all the time
- Do not require a hypothesis with zero true error
  - Require that $error_{\mathcal{D}}(h)$ is bounded by some constant ε, that can be made arbitrarily small
  - ε is the error parameter
- Approximately correct (近似正确)
- Do not require that the learner succeed on every sequence of randomly drawn examples
  - Require that its probability of failure is bounded by a constant, δ, that can be made arbitrarily small
  - δ is the confidence parameter
- Probably (可能)

PAC Learning Framework

Control Parameters

ε, δ

Training sample

$$\{< x_i, c(x_i) >\}_{i=1}^{n}$$

Learning algorithm L

Hypothesis

h

Concept class C

Distribution $\mathcal{D}$

PAC learnable (PAC可学习性)

- For all

  $c \in C$,

  distributions $\mathcal{D}$ over X (instance length: n – *complexity of the instance space, not the number of the instances*),

  ε such that $0 < \varepsilon < ½$

  δ such that $0 < \delta < ½$

- L will output a hypothesis $h \in H$ with

  [1] probability ≥ $(1 - \delta)$

  $error_{\mathcal{D}}(h) \leq \varepsilon$

  [2] in time that is polynomial in $1/\varepsilon$, $1/\delta$, n, and *size*(c).

Have nothing to do with |D|??

Effectiveness

Efficiency

→ C is PAC-learnable (PAC可学习的) by L using H

PAC learnable (PAC可学习性)

- If L requires some minimum processing time per training example
  - then for C to be PAC-Learnable, L must learn from a polynomial number of training examples.
- A typical approach to show some concept is PAC-Learnable usually consists of two steps:

  [1] Show that each target concept in C can be learned from a polynomial sample complexity

  [2] Show that the processing time per training example is also polynomially bounded

PAC learnability examples (1)

Example1: The class C of conjunctions of Boolean literals is PAC-learnable by the FIND-S algorithm using H=C

- Suppose H contains conjunctions of up to *n* Boolean attributes, then $|H| = 3^n$

[1] $m \geq \frac{1}{\varepsilon}(n \ln 3 + \ln(1/\delta))$

e.g. n=10, can be learned with 95% probability and error <0.1, then m=111.

n=3, (ε and δ are the same as above), then m= 34.

[2] The FIND-S algorithm require effort linear in *n* and independent of 1/δ, 1/ε, and size(c)

Find-S algorithm:

- Initialize h to the most specific hypothesis

$$l_1 \wedge \neg l_1 \wedge l_2 \wedge \neg l_2 \ldots l_n \wedge \neg l_n$$

- For each positive training instance x
  - Remove from h any literal that is not satisfied by x
- Output hypothesis h.

PAC learnability examples (2)

- Example 2: Unbiased learners are not PAC learnable
  - E.g. The instances x in X are defined by n boolean features.

$$|H| = 2^{|X|} = 2^{2^n}$$

$$m \geq \frac{1}{\varepsilon}(2^n \ln 2 + \ln(1/\delta))$$

  - Has **exponential** sample complexity

Agnostic Learning（不可知学习）

- So far, we assume c∈ H
- Agnostic learning: don’t assume c∈ H
- What do we want then?
  - The hypothesis h that makes fewest errors on training data
- What is the sample complexity in this case?

$$m \geq \frac{1}{2\varepsilon^2}(\ln|H| + \ln(1/\delta))$$

Derived from Hoeffding bounds:

$$Pr[\underbrace{error_{\mathcal{D}}(h)}_{\text{true error}} > \underbrace{error_{D}(h)}_{\text{training error}} + \underbrace{\epsilon}_{\text{degree of overfitting}}] \leq e^{-2m\epsilon^2}$$

What if H is not finite

- Can’t use previous results for finite H
- Need some other measure of complexity for H
  - Vapnik-Chervonenkis dimension!

Sample Complexity for Infinite Hypothesis Spaces -- VC Dim.

- Review: How many randomly sampled training examples are sufficient to assure that any concept will be probably (with probability $1-\delta$) approximately (within error ε) correct learned.
- Upper bound: Use VC(H) [Blumer 1989]

$$m \geq \frac{1}{\varepsilon}\left(4\log_2(2/\delta) + 8VC(H)\log_2(13/\varepsilon)\right)$$

So what is VC(H)?

Computational Learning Theory (Cont.)

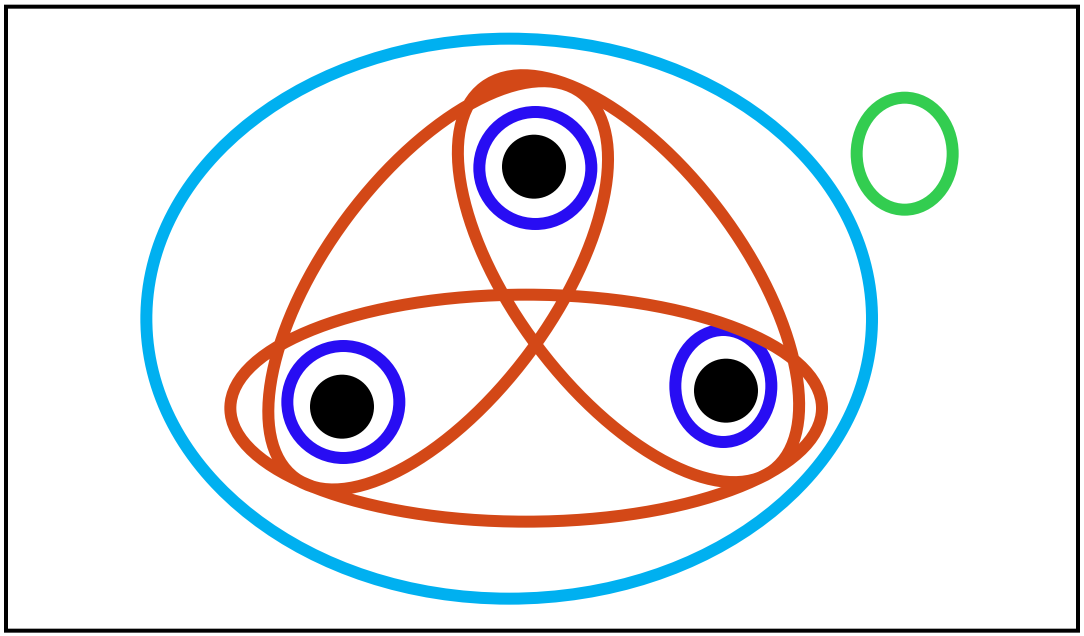
- The Vapnik-Chervonenkis (VC) dimension
  - Shattering a set of instances
  - VC dimension
    - Definition and several examples

Shattering (打散) a Set of Instances

- Definition:
  - A dichotomy (二分) of a set S is a partition of S into two disjoint subsets.

    $\{x \in S \mid h(x) = 1\}$ and $\{x \in S \mid h(x) = 0\}$
  - A set of instances S is shattered by hypothesis space H
    - If and only if for every dichotomy of S there exists some hypothesis in H consistent with this dichotomy.

Shattering a set of Instances: e.g. 3 instances



Instance space X

The Vapnik-Chervonenkis (VC) dimension

- An unbiased hypothesis space is one that shatters the instance space X.
- Sometimes X cannot be shattered by H, but a large subset of it can.
- Definition: The Vapnik-Chervonenkis Dimension VC(H) of hypothesis space H defined over instance space X
  - is the size of the largest finite subset of X shattered by H.
  - if arbitrarily large finite sets of X can be shattered by H, then $VC(H) \equiv \infty$

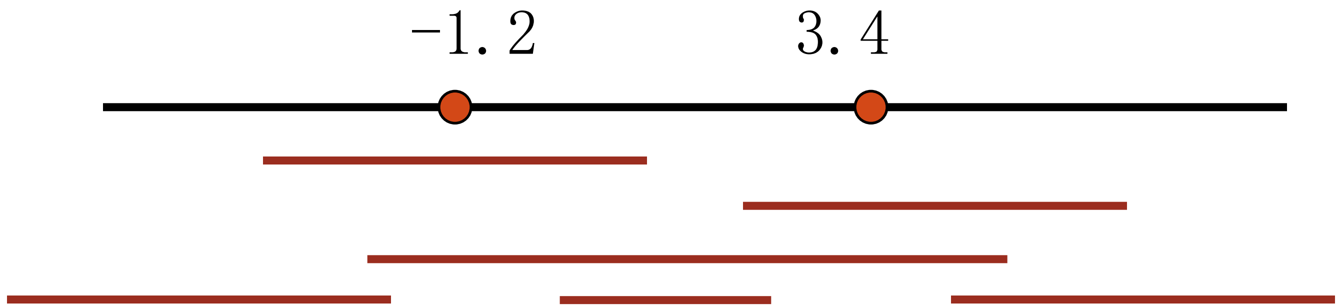
- If we find **ONE** set of instances of size d that can be shattered, then VC(H) ≥ d.
- To show that VC(H) < d, we must show that **NO** set of size d can be shattered.

VC Dim. Examples (1)

- Example 1:
  - Instance space X: the set of real numbers

    $X = \mathcal{R}$
  - H is the set of intervals on the real number axis.
  - Form of H is: $a < x < b$
  - VC(H) = ?

VC Dim. Examples (1)



- VC(H) >= 2

- VC(H) < 3

VC(H)=2

H is **infinite**, but VC(H) is **finite**

(To be continued …)

VC Dim. Examples (2)

- Example 2:
  - X of instances: numbers on the x, y plane
  - H is the set of all linear decision surfaces
  - VC(H) = ?
    - There **exists** one subset (size =3) that can be shattered
    - There **doesn't exist any** subset (size=4) that can be shattered
    - VC(H) = 3

In the space of *r* dimensions, H is the set of linear decision surface, VC(H)=r+1

VC Dim. Examples (3)

- Example 3:
  - X is all the example instances used to train a fully grown decision tree
  - H is all the Boolean expressions (rules) derived by a fully grown decision tree
  - VC(H) = ?
    - All the decision trees can be represented by Boolean functions
    - VC(H) is ∞

VC Dim.

- To sum up:
  - $\exists x, x\ is\ subset\ of\ X, x\ can\ be\ shattered \Rightarrow VC(H) \geq |x|$
  - To show that VC(H) < d, we must show that NO set of size d can be shattered.
- VC Dimension measures the representational of power of a given machine learning algorithm by measuring the expressive power of its hypothesis space.
  - Smaller VC dimension = less power
  - Larger VC dimensions = more power

Mistake Bound Framework
(出错界限模型)

Mistake Bound Framework

- So far: how many examples needed?
- What about: how many mistakes before convergence?
- Let's consider similar setting to PAC learning:
  - Instances drawn at random from X according to distribution $\mathcal{D}$
  - Learner must classify each instance before receiving correct classification from teacher
  - Can we bound the number of mistakes learner makes before converging?

Mistake Bound Framework – example

- Weighted Majority Algorithm
  - k: minimal number of mistakes

$$for\ \beta = \frac{1}{2}, \quad M \le 2.4(k + \log_2 n)$$ (See Ensemble Learning)

$$for\ any\ 0 \le \beta < 1, \quad M \le \frac{k \log_2 \frac{1}{\beta} + \log_2 n}{\log_2 \frac{2}{1+\beta}}$$

  - Why? -- please analyze it by yourself.

Optimal mistake bound

- Let $M_A(C)$ be the max number of mistakes made by algorithm A to learn concepts in C.

(maximum over all possible c∈C, and all possible training sequences)

$$M_A(C) \equiv \max_{c \in C} M_A(c)$$

- Definition: Let C be an arbitrary non-empty concept class. The **optimal mistake bound** for C, denoted Opt(C), is the minimum over all possible learning algorithms A of $M_A(C)$.

$$Opt(C) \equiv \min_{A \in learning\ algorithms} M_A(C)$$

$$\boxed{VC(C) \le Opt(C) \le M_{Halving}(C) \le log_2(|C|).}$$

Overview : Questions for Learning Algorithms

- Sample complexity (样本复杂度)
  - How many training examples do we need to converge to a successful hypothesis with a high probability?
- Computational complexity (计算复杂度)
  - How much computational effort is needed to converge to a successful hypothesis with a high probability?
- Mistake Bound (出错界限)
  - How many training examples will the learner misclassify before converging to a successful hypothesis?

Overview

- PAC learning （可能近似正确学习）
  - Probably (success probability 1-δ)
  - Approximately (error ε)
  - Sample complexity + Computational complexity
- Sample complexity (样本复杂度)
  - Finite hypothesis space （有限假设空间）
    - Consistent learner (一致学习器) $m \geq \frac{1}{\varepsilon}(\ln|H| + \ln\frac{1}{\delta})$
    - Agnostic learner (不可知学习器) $m \geq \frac{1}{2\varepsilon^2}(\ln|H| + \ln(1/\delta))$
  - Infinite hypothesis space (无限假设空间)：VC dimension

$$m \geq \frac{1}{\varepsilon}\left(4\log_2(2/\delta) + 8VC(H)\log_2(13/\varepsilon)\right)$$

- Mistake bound (出错界限)

Recommended Exercises: 7.2, 7.4, 7.5 (p227, En.)

No Submission requirement.