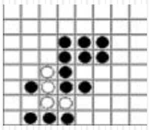
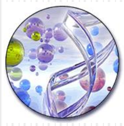
Welcome to

Introduction to Machine Learning!





Topic 14: Computational Learning Theory (cont.)

Min Zhang
z-m@tsinghua.edu.cn

Review: 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-δ) approximately correct (within error ε) .
—— PAC Learning 可能近似正确学习

Review: 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)

Review: 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 $\geq (1 - \delta)$

  $\text{error}_{\mathcal{D}}(h) \leq \varepsilon$

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

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

Have nothing to do with |D|??

Effectiveness

Efficiency

Review: 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

Review

- 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

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

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.

7 introduction to machine learning: computational learning theory

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 $M_A(C)$.

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

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

11

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?

12 introduction to machine learning: computational learning theory

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)

13 introduction to machine learning: computational learning theory

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

No Submission requirement

Mistake Bound Framework

- Proof:

  [1] The best algorithm make k mistakes → it's final weight is $(\beta)^k$.

  [2] The sum of all algorithms' final weights is at most

  $n\,(1-(1-\beta)/2)^M$.

  [3] $(\beta)^k \le n\,(1-(1-\beta)/2)^M$

$$M \le \frac{k \log_2 \frac{1}{\beta} + \log_2 n}{\log_2 \frac{2}{1+\beta}}$$