Machine Learning

Computational Learning Theory

Artificial Intelligence & Computer Vision Lab
School of Computer Science and Engineering
Seoul National University

Overview

- Introduction
- Probability Learning an Approximately Correct Hypothesis (PAC model)
- Sample Complexity for Finite Hypothesis Spaces
- Sample Complexity for Infinite Hypothesis Spaces
- The mistake Bound Model of Learning

Introduction

- Questions
  - Easy or difficult?
  - The number of training examples
  - How to collect the training examples
  - The number of mistakes before learning the target function
  - Computational complexity
- Answers
  - General answers to all these questions are not yet known

Introduction (cont.)

- Quantitative bounds setting
  - The size or complexity of the hypothesis space considered by the learner
  - The accuracy to which the target concept must be approximated
  - The probability that the learner will output a successful hypothesis
  - The manner in which training examples are presented to the learner

Introduction (cont.)

- Questions to answer
  - Sample complexity
    - How many training examples are needed for a learner to converge to a successful hypothesis?
  - Computational complexity
    - How much computational effort is needed for a learner to converge to a successful hypothesis?
  - Mistake bound
    - How many training examples will the learner misclassify before converging to a successful hypothesis?

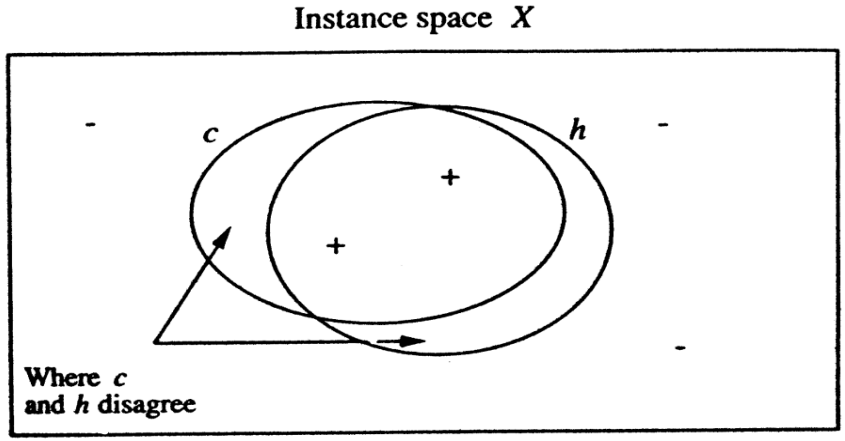
PAC Learning Model

- PAC: Probably Approximately Correct
- The problem setting
  - Learning boolean valued concepts from noise free training data
    - *X*: A set of all possible instances
    - C: A set of target concepts
    - c: A target concept
    - *D*: Instances which are generated at random from *X* according to some probability distribution *D*
    - *L*: Learner
    - *H*: A set of possible hypotheses
    - *h*: A hypothesis

PAC Learning Model (cont.)

- Error of a hypothesis
  - True error

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



The true error of hypothesis of h with respect to target concept c and distribution D is the probability that h will misclassify an instance drawn at random according to D

How probable is it that the observed training error for h gives a misleading estimate of the true error?
(example: sample error)

PAC Learning Model (cont.)

- PAC learnability
  - To characterize classes of target concepts learnable within
    - Reasonable amount of training examples
    - Reasonable amount of computation
  - Example: If we want to characterize the number of training examples needed to learn a hypothesis h for which $error_D(h) \equiv 0$
    - There may be multiple hypotheses consistent with the provided training examples
    - There will always be some nonzero probability that the training examples encountered by the learner will be misleading

PAC Learning Model (cont.)

- Our requirements
  - The learner outputs a hypothesis with the errors bounded by some constant ε
  - The probability of failure is bounded by some constant δ
- PAC
  - The learner probably learns a hypothesis that is approximately correct
- PAC learnable
  - Concept class *C* is PAC-learnable by *L* using *H* if, for any target concept *c* in C, *L* will with probability (1- δ) output a hypothesis *h* with $error_D(h) < \varepsilon$, after observing a reasonable number of training examples and performing a reasonable amount of computation
  - The above definition requires effectiveness and efficiency

Sample Complexity for Finite Hypothesis Spaces

- In practice, we are usually more concerned with the number of training examples
- Sample complexity
  - The growth in the number of required training examples with problem size
- Consistent learner
  - A learner is consistent if it outputs hypothesis that perfectly fit the training data (chapter 2.)
- Calculating sample complexity
  - Derive a bound on the number of training examples required by any consistent learner

Version Space

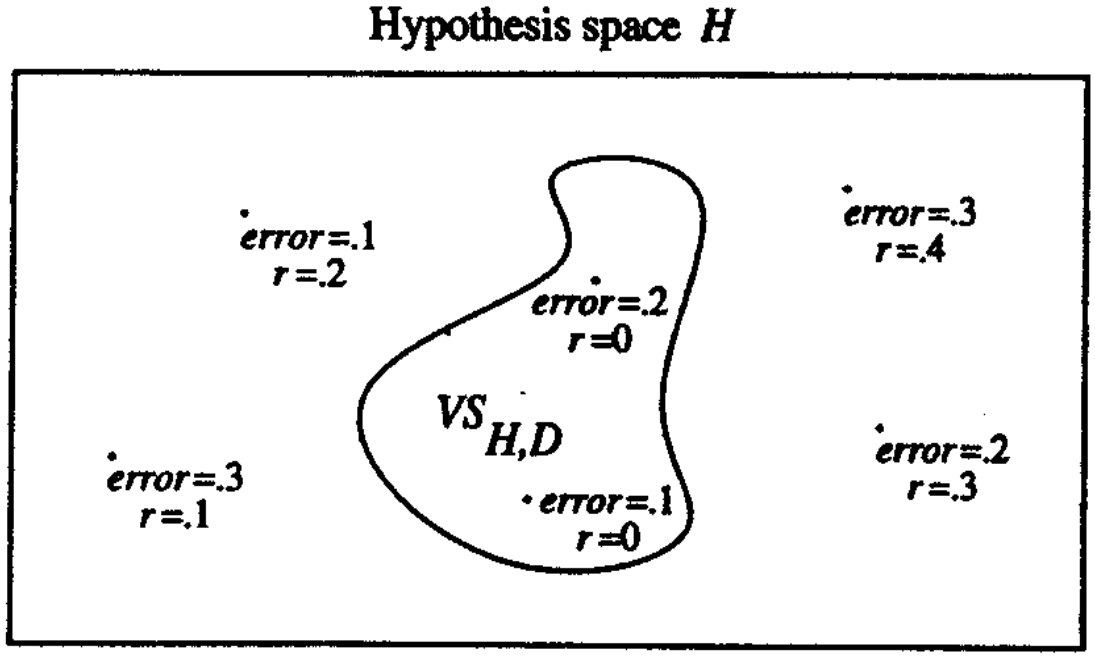
- *Definition*:

$$VS_{H,D} = \{h \in H \mid (\forall \langle x, c(x) \rangle \in D)(h(x) = c(x))\}$$

  - Set of all hypotheses $h \in H$ that correctly classify the training examples D.
  - Every consistent learner outputs a hypothesis belonging to the version space, regardless of X, H, or D.
- Problem
  - We need only bound the number of examples needed to assure that the version space contains no unacceptable hypotheses
  - ε -exhausted version space

ε - Exhausted Version Space

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



Sample Complexity

- Probabilistic arguments by Haussler (1988)
- The probability that version space $VS_{H,D}$ is not ε -exhausted $\leq |H| e^{-\varepsilon m}$
- The number of training examples

$$|H| e^{-\varepsilon m} \leq \delta$$

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

- Any consistent hypothesis will be probably (probability (1- δ)) approximately (within error ε) correct

Agnostic Learning and Inconsistent Hypotheses

- Agnostic learner
  - Makes no assumption that the target concept is representable by H (training error can be non-zero)
  - Simply finds the hypothesis with minimum training error (h_{best})
  - $error_D(h_{best}) \leq \varepsilon + error_D(h_{best})$

- By probabilistic arguments (Hoeffding Chernoff bounds)

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

Examples of PAC-Learnability

- Conjunctions of Boolean literals
  - "$Old \wedge \neg Tall$"

$$|H| = 3^n \qquad m \geq \frac{1}{\varepsilon}(n \ln 3 + \ln(1/\delta))$$

  - 10 boolean literals, 95% probability, error rate less than 0.1 ➔ 140 training examples
- Unbiased learners
  - Considering the unbiased concept class C that contain every teachable concept relative to X

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

Sample Complexity for Infinite Hypothesis Space

- Drawback of the following equation

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

  - Quite weak bounds
    - Bound on δ can be significantly greater than 1 for large $|H|$
  - Cannot be applied to infinite hypothesis space

- Another measure of the sample complexity
  - Vapnik-Chervonenkis of H (VC dimension, or $VC(H)$)

Shattering a Set of Instances

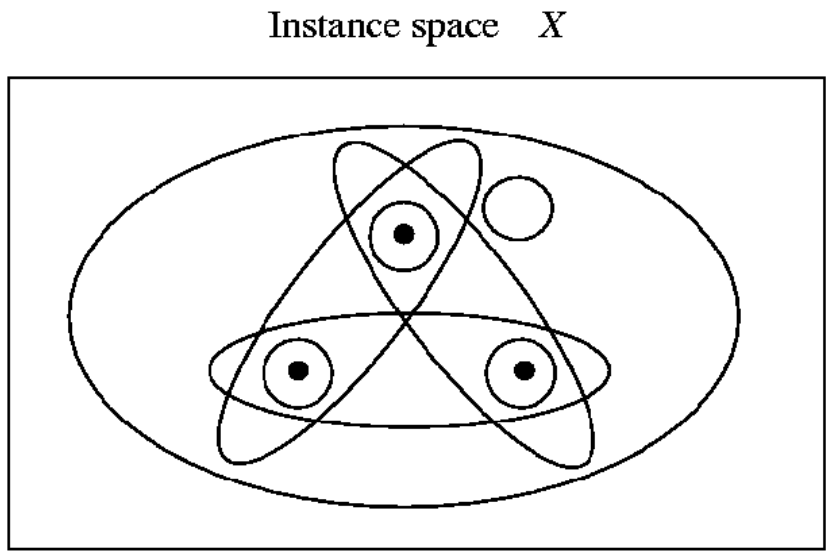
- A dichotomy of a set S

  A hypothesis h partitions S into two disjoint subsets

  $\{x \in S \mid h(x) = 1\}$ and $\{x \in S \mid h(x) = 0\}$.

  Then, given some instances set S, there are $2^{|S|}$ possible dichotomies.

- *Definition*: 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.



A set of three instances shattered by eight hypothesis. For every possible dichotomy of the instances, there exist a corresponding hypothesis.

- Note that if a set of instances is not shattered by a hypothesis space then there must be some concept (dichotomy) that can be defined over the instances, but that cannot be represented by the hypothesis space. The ability of H to shatter a set of instances is thus a measure of its capacity to represent target concepts defined over these instances.

Vapnik-Chervonenkis Dimension

- *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 $VC(H) = \infty$

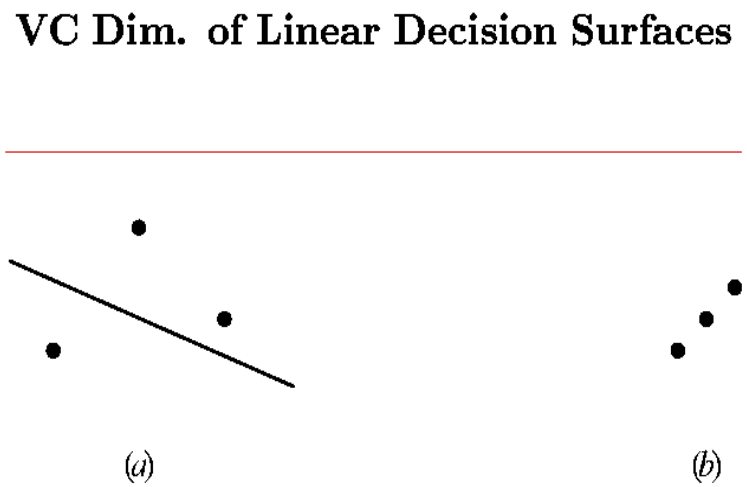
- Note that $VC(H) \leq \log_2|H|$ for any finite hypothesis space H
  - Suppose $VC(H) = d$. Then, H will require 2^d distinct hypothesis to shatter d instances. Hence $2^d \leq |H|$ and $d = VC(H) \leq \log_2|H|$

Illustrative Examples

- Example:
  - Condition
    - X: A set of real numbers
    - H: A set of intervals
      - $H = \{ (a,b) \mid a < b$ is any real number $\}$
  - $VC(H)$
    - If $S = \{3.7, 5.7\}$, then
      the four hypotheses (1,2), (1,4), (4,7) and (1,7) will shatter.
      Thus, $VC(H) \geq 2$
    - If $|S| \geq 3$, $S = \{x_0. x_1, x_2\}$ assuming $x_0 < x_1 < x_2$, then not shattered.
      Thus , $VC(H) = 2$

Illustrative Examples (cont.)

- Example:
  - Condition
    - X: A set of point on the x,y planes
    - H: A single perceptron with 2 inputs (= linear decision surfaces in the plane)
  - $VC(H)$
    - $3 \leq VC(H) < 4$. Thus, $VC(H) = 3$



  - More generally, VC dimension of linear decision surfaces in an r dimensional spaces (= perceptron with r inputs) is $VC(H) = r + 1$

Illustrative Examples (cont.)

- Example:
  - Condition
    - X: A set of conjunctions of 3 boolean literals
      - $l_1 = 100$, $l_2 = 010$, $l_3 = 001$
    - H: A set of conjunctions of up to 3 boolean literals
  - $VC(H)$
    - If we wish to include l_1, but exclude l_2, l_3, then we use the hypothesis $\neg l_2 \wedge \neg l_3$. $\{l_1, l_2, l_3\}$ is then shattered by H. Thus, $VC(H) = 3$
  - More generally, VC dimension for conjunctions of n boolean literals is $VC(H) = n$

Sample Complexity and VC Dimension

- Upper bound on sample complexity
  - Sufficient training examples for a successful learning

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

  - Compare to $m \geq \frac{1}{\varepsilon}\left(\ln|H| + \ln(1/\delta)\right)$ in finite hypothesis spaces.

Sample Complexity and VC Dimension (cont.)

- *Theorem*: Lower bound on sample complexity. (Necessary training examples for a successful learning)

  Consider any concept class C such that $VC(C) \geq 2$, any learner L, any $0 < \varepsilon < 1/8$, any $0 < \delta < 1/100$. Then there exists a distribution D and target concept in C such that if L observes fewer examples than

$$\max\left[\frac{1}{\varepsilon}\log(1/\delta), \frac{VC(C)-1}{32\varepsilon}\right]$$

  then with probability at least δ, L outputs a hypothesis h having $\text{error}_D(h) > \varepsilon$

VC Dimension for Neural Networks

- *Theorem*: VC-dimension of directed acyclic layered networks. Let G be a layered directed acyclic graph with n input nodes and $s \geq 2$ internal nodes, each having at most r inputs. Let C be a concept class over R^r of VC dimension d, corresponding to the set of functions that can be described by each of the s internal nodes. Let C_G be the G-composition of C, corresponding to the set of functions that can be represented by G. Then, $VC(C_G) \leq 2ds\log(es)$, where e is the base of the natural logarithm.

- If individual nodes are perceptrons, then

  $VC(C_G^{\text{perceptrons}}) \leq 2\,(r+1)\,s\,\log(es)$.

  Substituting this for $VC(H)$ in the previous page, we have

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

$$\geq \frac{1}{\varepsilon}\left(4\log(2/\delta)+16(r+1)s\log(es)\log(13/\varepsilon)\right)$$

Mistake Bound Model of Learning

- Mistake Bound Model
  - For each example x the learner must predict the target value $c(x)$, before $c(x)$ is shown by the trainer
  - The learner is evaluated by the total # of mistakes it makes before it converges to the correct h
    - Number of mistakes made before learning the target concept exactly
  - Significant when learning is done while the system is in actual use

Mistake Bound Model of Learning (cont.)

- Example
  - If the system is to learn to predict which credit card purchases should be approved and which are fraudulent, based on data collected during use, then we are interested in minimizing the total number of mistakes it will make before converging to the correct target function.

Mistake Bound for the Find-S Algorithm

- Find-S implementation
  - Initialize h to the most specific hypothesis $l_1 \wedge \sim l_1 \wedge l_2 \wedge \sim l_2 \wedge \ldots l_n \wedge \sim l_n$
  - For each positive training instance x
    - Remove from h any literal that is not satisfied by x
  - Output hypothesis h
- Find-S converges provided
  - $C \subseteq H$
  - Noise-free instances
- Worst case mistake bound = $n + 1$
  - $(\forall x)\ c(x) = 1$
  - Sequence of instances
    - Removing only one literal per mistake

Mistake Bound for the Halving Algorithm

- Halving algorithm
  - Learns concept using version space Candidate-Elimination algorithm
  - Classifies new instance by majority vote of version space members
- Worst case mistake bound $\leq \log_2|H|$
  - At each mistake, only those hypotheses that voted with minority will be retained
    - Each mistake reduces the size of version space by at least half its current size
  - Best case mistakes = 0
    - When majority vote is correct, the algorithm will remove the minority hypothesis

Optimal Mistake Bounds

- $M_A(c)$: The max of the # of mistakes made by A to exactly learn c
  - For arbitrary concept class C, assume $C = H$
- $M_A(C) \equiv \max_{c \in C} M_A(c)$
  - $M_{\text{Find-S}}(C) = n+1$
  - $M_{\text{Halving}}(C) \leq \log_2|C|$
- Optimal mistake bound
  - $\text{Opt}(C) \equiv \min_{A \in \text{ learning algorithm}} M_A(C)$
    - The number of mistakes made for the hardest target concept in C, using hardest training sequence, by the best algorithm
  - $VC(C) \leq \text{Opt}(C) \leq M_{\text{Halving}}(C) \leq \log_2|C|$
    - When C is powerset C_p of any finite set of instance X
      - $VC(C_p) = |X| = \log_2|C_p|$

Weighted-Majority Algorithm

- Generalization of the Halving algorithm
  - Makes predictions by taking a weighted vote among a pool of prediction algorithm
  - Learns by altering the weight associated with each prediction algorithm
    - Prediction algorithms are alternative hypotheses in *H*
- Property
  - Accommodate inconsistent training data
    - Weight reduction instead of hypothesis elimination
  - Bound the number of mistakes by *WM* to the number of mistakes made by the best pool of prediction algorithms

Weighted-Majority Algorithm (cont.)

- Implementation

  a_i denote the i^{th} prediction algorithm in the pool A of algorithms. w_i denotes the weight associated with a_i. $0 \leq \beta < 1$

  - For all i, initialize $w_i \leftarrow 1$.
  - For each training example $\langle x, c(x) \rangle$
    - Initialize q_0 and q_1 to 0
    - For each prediction algorithm a_i
      - if $a_i(x) = 0$ then $q_0 \leftarrow q_0 + w_i$
        if $a_i(x) = 1$ then $q_1 \leftarrow q_1 + w_i$
    - If $q_1 > q_0$ then predict $c(x) = 1$
      If $q_0 > q_1$ then predict $c(x) = 0$
      If $q_0 = q_1$ then predict 0 or 1 at random for $c(x)$
    - For each prediction algorithm a_i in A do
      - if $a_i(x) \neq c(x)$ then set $w_i \leftarrow \beta w_i$

Weighted-Majority Algorithm (cont.)

- Worst case mistake bound = $2.4(k + \log_2 n)$ ($\beta = ½$)
  - Condition
    - D: Any sequence of training examples
    - A: Any set of n prediction algorithms
    - k : Minimum number of mistakes made by any algorithm in A for training sequence D

Weighted-Majority Algorithm (cont.)

- Proof
  - a_j : An algorithm that commits the optimal number k of mistakes
    - The final weight w_j of a_j is $(\frac{1}{2})^k$
  - W : Sum of w_j, initially n
    - W is reduced to at most $\frac{3}{4}W$ at each mistake
      - » $\frac{1}{2}W + \frac{1}{2} * \frac{1}{2}W = \frac{3}{4}W$
  - M : The total number of mistakes
    - The final total weight W is at most $n(\frac{3}{4})^M$

$$\left(\frac{1}{2}\right)^k \le n\left(\frac{3}{4}\right)^M$$

$$M \le \frac{(k+\log_2 n)}{-\log_2\left(\frac{3}{4}\right)} \le 2.4(k+\log_2 n)$$

Weighted-Majority Algorithm (cont.)

- Generalized worst case mistake bound ($0 \leq \beta < 1$)
  - The number of of mistakes made by Weighted-Majority algorithm will never be greater than a constant factor times the number of mistakes made by the best member of the pool, plus a term that grows only logarithmically in the size of the pool

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