Computational Learning Theory

Based on "Machine Learning", T. Mitchell, McGRAW Hill, 1997, ch. 7

Acknowledgement:

The present slides are an adaptation of slides drawn by T. Mitchell

Main Questions in Computational Learning Theory

- **Can one characterize the number of training examples necessary/sufficient for successful learning?**
- **Is it possible to identify classes of concepts that are inherently difficult/easy to learn, independent of the learning algorithm?**

We seek answers in terms of

- sample complexity: number of needed training examples
  - computational complexity: time needed for a learner to converge (with high probability) to a successful hypothesis
  - the manner in which training examples should be presented to the learner
- mistake bound: number of mistakes made by the learner before eventually succeeding

Remarks

1. Since no *general* answers to the above questions are yet known,
   we will give some key results for *particular* settings.
2. We will restrict the presentation to inductive learning, in a universe of instances X, in which we learn a target function c from a number of examples D, searching a candidate h in a given hypothesis space H.

Plan

1. The Probably Approximately Correct learning model
   - 1.1 PAC-Learnable classes of concepts
   - 1.2 Sample complexity
     - Sample complexity for finite hypothesis spaces
     - Sample complexity for infinite hypothesis spaces
   - 1.3 The Vapnik-Chervonenkis (VC) dimension
2. The Mistake Bound model of learning
   - • The HALVING learning algorithm
   - • The WEIGHTED-MAJORITY learning algorithm
   - ◦ The optimal Mistake Bounds

1. The PAC learning model

Note:

For simplicity, here we restrict the presentation to learning boolean functions, using noisy free training data.

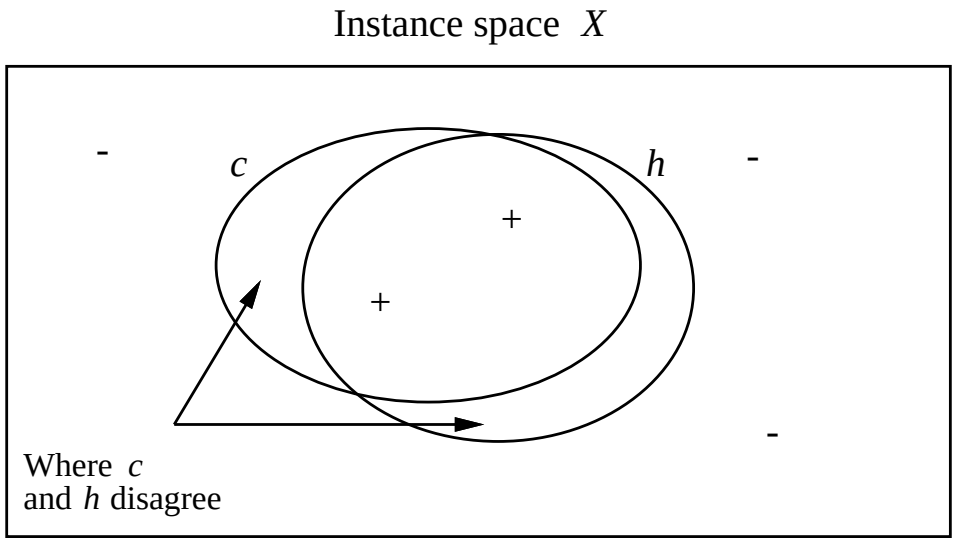
Extensions:

- considering real-valued functions: [Natarajan, 1991];
- considering noisy data: [Kearns & Vazirani, 1994].

The True Error of a Hypothesis: $error_{\mathcal{D}}(h)$

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

the probability that h will misclassify a single instance drawn at random according to the distribution $\mathcal{D}$.



Note: $error_{\mathcal{D}}(h)$ **is not directly observable to the learner; it can only see the training error of each hypothesis (i.e., how often $h(x) \neq c(x)$ over training instances).**

Question:

Can we bound $error_{\mathcal{D}}(h)$ given the training error of h?

Important Note

Ch. 5 (*Evaluating Hypotheses*) explores the relationship between *true error* and *sample error*, given a sample set S and a hypothesis h, with S independent of h.

When S is the set of training examples from which h has been learned (i.e., D), obviously h is not independent of S. *Here* we deal with this case.

The Need for Approximating the True Error of a Hypothesis

Suppose that we would like to get a hypothesis h with true error 0:

1. **the learner should choose among hypotheses having the training error 0, but since there may be several such candidates, it cannot be sure which one to choose**
2. **as training examples are drawn randomly, there is a non-0 probability that they will mislead the learner**

Consequence: demands on the learner should be weakened

1. **let $error_{\mathcal{D}}(h) < \epsilon$ with ϵ arbitrarily small**
2. **not every sequence of training examples should succeed, but only with probability $1 - \delta$, with δ arbitrarily small**

1.1 PAC Learnable Classes of Concepts: Definition

Consider a class C of possible target concepts defined over a set of instances X of length n, and a learner L using the hypothesis space H.

C is PAC-learnable by L using H if

for all $c \in C$, distributions $\mathcal{D}$ over X, ϵ such that $0 < \epsilon < 1/2$, and δ such that $0 < \delta < 1/2$,

the learner L will with probability at least $(1 - \delta)$ output a hypothesis $h \in H$ such that $error_{\mathcal{D}}(h) \leq \epsilon$,
in time that is polynomial in $1/\epsilon$, $1/\delta$, n and $size(c)$,

where $size(c)$ is the encoding length of c, assuming some representation for C.

PAC Learnability: Remarks (I)

- If C is PAC-learnable, and each training example is processed in polynomial time,
  then each $c \in C$ can be learned from a polynomial number of training examples.

- **Usually,** to show that a class C is PAC-learnable, we show that each $c \in C$ can be learned from a polynomial number of examples, and the processing time for each example is polynomially bounded.

PAC Learnability: Remarks (II)

- ***Unfortunately***, we cannot ensure that H contains (for any ϵ, δ) an h as in the difinition of PAC-learnability unless C is known in advance, or $H \equiv 2^X$.

- ***However***, PAC-learnability provides useful insights on the relative complexity of different ML problems, and the rate at which generalization accuracy improves with additional training examples.

1.2 Sample Complexity

In practical applications of machine learning, evaluating the sample complexity (i.e. the number of needed training examples) is of greatest interest because in most practical settings limited success is due to limited available training data.

We will present results that relate (for different setups)

- **the size of the instance space** (m)

to

- **the accuracy to which the target concept is approximated** (ϵ)
- **the probability of successfully learning such an hypothesis** $(1 - \delta)$
- **the size of the hypothesis space** $(|H|)$

1.2.1 Sample Complexity for Finite Hypothesis Spaces

First, we will present a general bound on the sample complexity for consistent learners, i.e. which perfectly fit the training data.

Recall the version space notion:

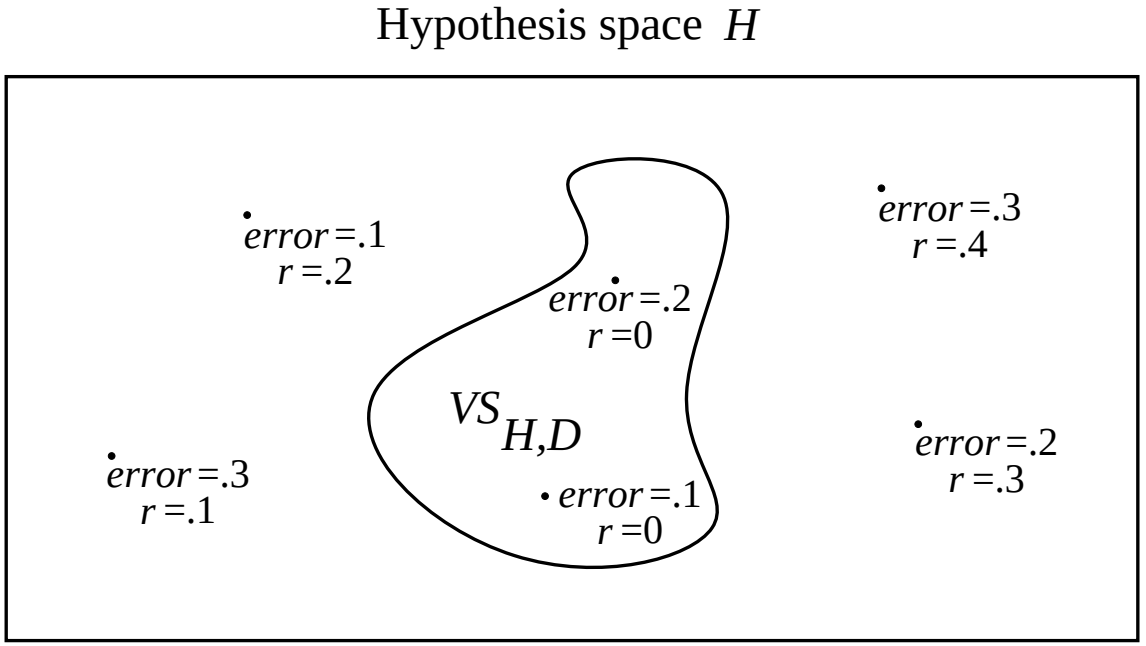
$$VS_{H,D} = \{h \in H | \forall \langle x, c(x) \rangle \in D, h(x) = c(x)\}$$

Later, we will consider agnostic learning, which accepts the fact that a zero training error hypothesis cannot always be found.

Exhaustion of the Version Space

Definition:

$VS_{H,D}$ **is** ϵ**-exhausted with respect to the target concept** c **and the training set** D **if** $error_{\mathcal{D}}(h) < \epsilon$, $\forall h \in VS_{H,D}$.



r = **training error,** $error$ = **true error**

How many examples will ϵ-exhaust the VS?

Theorem: [Haussler, 1988]

If the hypothesis space H is finite, and D is a sequence of $m \geq 1$ independent random examples of some target concept $c \in H$, then for any $0 \leq \epsilon \leq 1$, the probability that $VS_{H,D}$ is not ϵ-exhausted (with respect to c) is less than

$$|H|e^{-\epsilon m}$$

Proof: **Let h be a hypothesis of true error $\geq \epsilon$. The probability that h is consistent with the m independently drawn training examples is $< (1-\epsilon)^m$.**

The probability that there are such hypothesis h in H is $< |H|(1-\epsilon)^m$.

As $1 - \epsilon \leq e^{-\epsilon}$ for $\forall \epsilon \in [0,1]$, it follows that $|H|(1-\epsilon)^m \leq |H|e^{-\epsilon m}$.

Consequence: **The above theorem bounds the probability that any consistent learner will output a hypothesis h with $error_{\mathcal{D}}(h) \geq \epsilon$. If we want this probability to be below δ**

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

then

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

This is the number of training examples sufficient to ensure that any consistent hypothesis will be probably (with probability $1 - \delta$) approximately (within error ϵ) correct.

Example 1: *EnjoySport*

If H is as given in *EnjoySport* (see Chapter 2) then $|H| = 973$, and

$$m \geq \frac{1}{\epsilon}(\ln 973 + \ln(1/\delta))$$

If want to assure that with probability 95%, VS contains only hypotheses with $error_{\mathcal{D}}(h) \leq 0.1$, then it is sufficient to have m examples, where

$$m \geq \frac{1}{0.1}(\ln 973 + \ln(1/.05)) = 10(\ln 973 + \ln 20) = 10(6.88 + 3.00) = 98.8$$

Example 2: Learning conjunctions of boolean literals

Let H be the hypothesis space defined by conjunctions of literals based on n boolean attributes possibly with negation.

Question: How many examples are sufficient to assure with probability of at least $(1-\delta)$ that every h in $VS_{H,D}$ satisfies $error_{\mathcal{D}}(h) \leq \epsilon$?

Answer: $|H| = 3^n$, and using our theorem it follows that

$$m \geq \frac{1}{\epsilon}(\ln 3^n + \ln(1/\delta)) \text{ or } m \geq \frac{1}{\epsilon}(n \ln 3 + \ln(1/\delta))$$

In particular, as FIND-S spends $O(n)$ time to process one (positive) example, it follows that it PAC-learns the class of conjunctions of n literals with negation.

Example 3:
PAC-Learnability of k-term DNF expressions

k-term DNF expressions: $T_1 \vee T_2 \vee \ldots \vee T_k$ where T_i is a conjunction of n attributes possibly using negation.

If $H = C$ then $|H| = 3^{nk}$, therefore $m \geq \frac{1}{\epsilon}(nk \ln 3 + \ln \frac{1}{\delta})$, which is polynomial, but...

it can be shown (through equivalence with other problems) that it cannot be learned in polynomial time (unless $RP \neq NP$)

therefore k-term DNF expressions are not PAC-learnable.

Example 4: PAC-Learnability of k-CNF expressions

k-CNF expressions are of form $T_1 \wedge T_2 \wedge \ldots \wedge T_j$
where T_i is a disjunction of up to k boolean attributes.

Remark:

k-term DNF expressions $\subset$ k-CNF expressions.

Surprisingly, k-CNF expressions are PAC-learnable by a polynomial time complexity algorithm (see [Vazirani, 1994]).

Consequence:

k-term DNF expressions are PAC-learnable by an efficient algorithm using $H = k$-CNF(!).

Example 5: PAC-Learnability of Unbiased Learners

In such a case, $H = C = \mathcal{P}(X)$.

If the instances in X are described by n boolean features,
then $|X| = 2^n$ and $|H| = |C| = 2^{|X|} = 2^{2^n}$,
therefore $m \geq \frac{1}{\epsilon}(2^n \ln 2 + \ln \frac{1}{\delta})$

Remark:

Although the above bound is not a tight one,
it can be shown that the sample complexity for learning the unbiased concept class is indeed exponential in n.

Sample Complexity for Agnostic Learning

Agnostic learning doesn't assume $c \in H$, therefore c may or may not be perfectly learned in H. In this more general setting, a hypothesis which has a zero training error cannot always be found.

- **What can we search for?**
  A hypothesis h that makes the fewest errors on training data.
- **What is the sample complexity in this case?**

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

Proof idea: use Hoeffding-Chernoff bounds

$$Pr[error_{\mathcal{D}}(h) > error_D(h) + \epsilon] \leq e^{-2m\epsilon^2}$$

1.2.2 Sample Complexity for Infinite Hypothesis Spaces

For $|H| = \infty$, in order to better evaluate the sample complexity, we will use another measure: the Vapnik-Chervonenkis dimension, $VC(H)$, the number of instnaces from X which can be discriminated by H. Now we prepare its introduction.

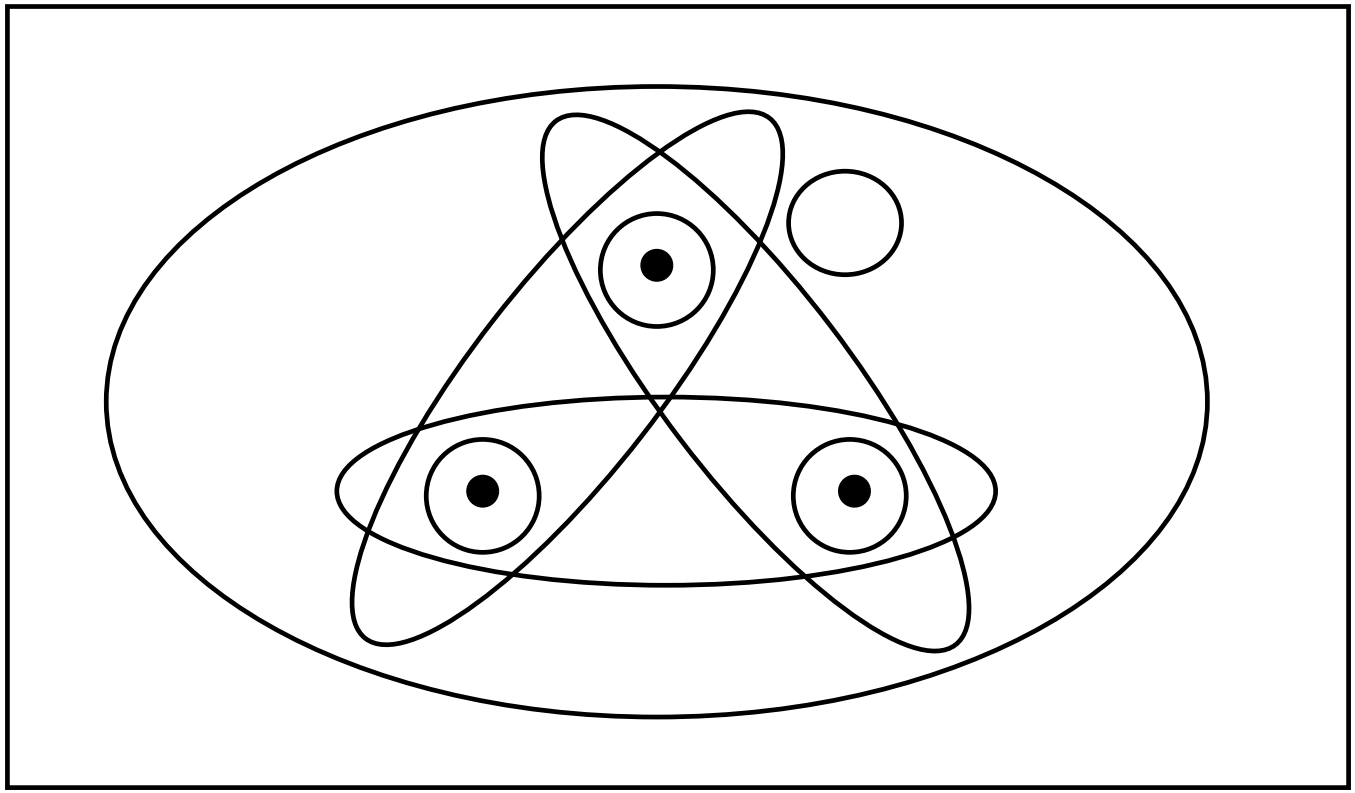
Definitions:

A dichotomy of a set S is a partition of S into two disjoint subsets.

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.

Example: Three Instances Shattered

Instance space X



Remarks

1. **The ability of H to shatter a set of instances $S \subseteq X$ is a measure of its capacity to represent target concepts defined over the instances in S.**
2. **Intuitively, the larger subset of X can be shattered, the more expressive is H!**
3. **An unbiased H is one that shatters the entire X.**

1.3 The Vapnik-Chervonenkis Dimension

The Vapnik-Chervonenkis dimension, $VC(H)$, of the hypothesis space H defined over the 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 $VC(H) \equiv \infty$.

Note:

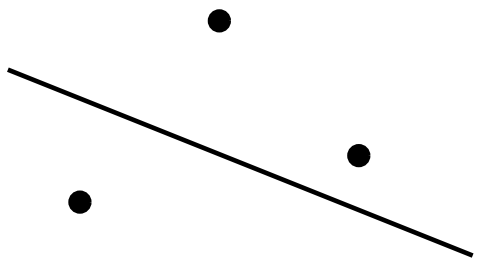
If $|H| < \infty$, then $VC(H) \leq \log_2 |H|$.
(Proof: if $d = VC(H)$, then $|H| \geq 2^d$ and so $d \leq \log_2 \mid H \mid$.)

Example 1:

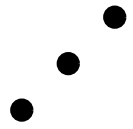
If $X = R$ and H is the set made of all intervals on the real number line, then $VC(H) = 2$.
Proof: first show that $VC(H) \geq 2$, then $VC(H) < 3$.

Example 2:
VC Dimension of Linear Decision Surfaces

If $X = R^2$ and H is the set of all liniar decision surfaces in the plane, then $VC(H) = 3$.
(Proof: show that $VC(H) \geq 3$, then $VC(H) < 4$).



(a)



(b)

In general, for $X = R^n$, if H is the set of all liniar decision surfaces in R^n, $VC(H) = n + 1$.

Note that $|H| = \infty$ but $\boldsymbol{VC}(H) < \infty$.

Example 3:

The VC Dimension for Conjunctions of Boolean Literals

The VC dimension for H consisting of conjunctions of up to n boolean literals is n.

Proof:

show that $VC \geq n$:

consider S the set of all *instances* which are conjunctions of exactly n boolean literals, so that for each instance only one literal (l_i) is positive;

show that S is shattered by H:

if exactly $inst_{i_1}, inst_{i_2}, \ldots inst_{i_k}$ must be excluded,

then define $h = \neg l_{i_1} \wedge \neg l_{i_2} \wedge \ldots \wedge \neg l_{i_k}$;

as $\mid S \mid = n$, it follows that $VC \geq n$

show that $VC < n + 1$ ... (more) difficult.

Sample Complexity and the VC Dimension

How many randomly drawn examples suffice to ϵ-exhaust $VS_{H,D}$ with probability at least $(1-\delta)$?

[Blumer et al., 1989]:

if $c \in H$, then $m \geq \frac{1}{\epsilon}(8\,\boldsymbol{VC}(H)\log_2(13/\epsilon) + 4\log_2(2/\delta))$

(Remember, $VC(H) \leq \log_2 |H|$.)

[Ehrenfeucht et al., 1989] A lower bound for PAC-learnability:

If $VC(C) \geq 2$, $0 < \epsilon < \frac{1}{8}$, $0 < \delta < \frac{1}{100}$, for any learner L there is a distribution $\mathcal{D}$ and a concept $c \in C$ such that if L observes fewer examples (of c) than $\max[\frac{1}{\epsilon}\log_2(\frac{1}{\delta}), \frac{VC(C)-1}{32\epsilon}]$, then with probability at least δ, L outputs a hypothesis h having $error_{\mathcal{D}}(h) > \epsilon$.

Means: if the number of examples is too law, no learner can learn every concept in a nontrivial class C.

Summary

For $\mid H \mid < \infty$**,**

$m \geq \frac{1}{\epsilon}(\ln |H| + \ln(1/\delta))$ **if** $c \in H$**,**
where c **is the concept to be learned;**

$m \geq \frac{1}{2\epsilon^2}(\ln |H| + \ln(1/\delta))$
when it is not known whether $c \in H$ **or not**
(agnostic learning).

Using $VC(H)$ **instead of** $\mid H \mid$ **(especially when** $\mid H \mid = \infty$**),**

$m \geq \frac{1}{\epsilon}(8VC(H)\log_2(13/\epsilon) + 4\log_2(2/\delta))$**.**

2. The Mistake Bound Learning Model

So far: how many examples are needed for learning?

What about: how many mistakes before convergence?

Let's consider a similar setting to PAC learning:

- Instances are drawn at random from X according to a distribution $\mathcal{D}$
- But now the learner must classify each instance before receiving correct classification from the teacher

Question: Can we bound the number of mistakes the learner makes before converging?

Mistake Bounds

Importance: there are practical settings in which the system learns while it is already in use (rather than during off-line training).

Example: detecting fraudulent credit cards in use.
In this case the number of mistakes is even more important than the number of (needed) training examples.

Note: In the following examples we evaluate the number of mistakes made by the learner before learning *exactly* (not PAC) the target concept: $h(x) = c(x)\ \forall x$.

Example 1: The FIND-S Algorithm (Ch. 2)

Consider H consisting of conjunctions of up to n boolean literals $l_1, l_2, \ldots, l_n$ and their negations.

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 the hypothesis h.**

Question: **How many mistakes does FIND-S before converging?**

Answer: $n + 1$.

Proof:

FIND-S cannot misclassify negative examples (because the current h is always at least as specific as the concept to be learned).

Therefore we have to find how many positive examples it will misclassify before converging.

FIND-S will certainly misclassify the 1st positive example, after which it will eliminate n of the $(2n)$ literals in the most specific h shown on the previous slide.

For all subsequent positive examples at most n literals will be eliminated.

Therefore FIND-S can do at most $n+1$ mistakes before converging.

Example 2: The HALVING Algorithm

HALVING:

- **Learn the target concept using the version space (similarly to the CANDIDATE-ELIMINATION algorithm, Ch. 2).**
- **Classify new instances by taking the majority vote of version space members;**
- **after receiving the correct classification from the teacher, the wrong hypotheses are eliminated.**

Question: **How many mistakes makes HALVING before converging to the correct h?**

Answer: **at most $\lceil \log_2 |H| \rceil$.**

Proof:

HALVING **misclassifies an example** x **when at least half plus one of (the number of) all hypotheses in the curent version space misclassify** x**. In such case, when** $c(x)$ **is revealed to the learner, at least half plus one of the hypotheses in the curent version space are eliminated.**

It follows that HALVING **does at most** $[\log_2 |H|]$ **mistakes. (In this setting exact learning is perfomed; only one hypothesis is retained in the end.)**

Note: It is possible that HALVING **wil llearn without making any mistake! (At each step, the hypotheses which are inconsistent with the current example are eliminated.)**

Example 3: Weighted Majority Learning

- generalizes the HALVING algorithm
- takes a weighted vote among a pool of prediction algorithms
- learns by altering the weight associated with each prediction algorithm
- it is able to accommodate inconsistent data, due to the weighted vote procedure
- the number of mistakes made by WEIGHTED-MAJORITY can be bound in terms of the mistakes made by the best algorithm in the pool of prediction algorithms

The Weighted-Majority Algorithm

a_i – **the** i^{th} **algorithm in the pool** A **of algorithms**
w_i – **the weight associated with** a_i
$\beta \in [0; 1)$

for all i initialize $w_i \leftarrow 1$
for each training example $\langle x, c(x) \rangle$
  initialize W_- and W_+ to 0
  for each prediction algorithm a_i
    if $a_i(x) = 0$ then $W_- \leftarrow W_- + w_i$
    else $W_+ \leftarrow W_+ + w_i$
  if $W_+ > W_-$ **then** predict 1
  if $W_- > W_+$ **then** predict 0
  if $W_+ = W_-$ **then** predict 0 or 1 at random
  for each prediction algorithm a_i in A
    if $a_i(x) \neq c(x)$, ($c(x)$ is indicated by the teacher),
    then $w_i \leftarrow \beta w_i$

Note: For $\beta = 0$, Weighted-Majority **is** Halving.

WEIGHTED-MAJORITY: Relative Mistake Bound

Theorem:

For D – any sequence of training examples
A – any set of n prediction algorithms
k – the minimum number of mistakes made by any algorithm in A when training on D,
the number of mistakes made over D by WEIGHTED-MAJORITY using $\beta = 1/2$ is at most

$$2.41(k + \log_2 n)$$

Generalization: ([Littlestone & Warmth, 1991])

For arbitrary $\beta \in [0, 1)$, the number of mistakes made over D by WEIGHTED-MAJORITY is at most $\frac{k \log_2 \frac{1}{\beta} + \log_2 n}{\log_2 \frac{2}{1+\beta}}$

Theorem Proof:

a_j – the optimal algorithm in A
k – mistakes made by a_j on D
$w'_j = \frac{1}{2^k}$, the final weight associated with a_j
W – the sum of all weights assoc. with all algorithms in A
M – the total number of mistakes made by WEIGHTED-MAJORITY while training on D

Initially, $W = n$;
then, at each mistake made by WEIGHTED-MAJORITY, W is decreased to $\leq \frac{3}{4}W$;
finally, $W \leq n(\frac{3}{4})^M$.

Since $w'_j \leq$ the final value of W, it follows that

$$\frac{1}{2^k} \leq n(\frac{3}{4})^M \Rightarrow M \leq -\frac{k + \log_2 n}{\log_2 3/4} \leq 2.41(k + \log_2 n)$$

Optimal Mistake Bounds

Let C be an arbitrary non-empty concept class, and A a learning algorithm. Taking the maximum over all possible $c \in C$, and all possible training sequences, we define $M_A(C)$ the maximum number of mistakes made by the algorithm A to learn concepts in C:

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

The optimal mistake bound for C is the numer of mistakes for learning the hardest concept in C with the hardest training sequence, using the best algorithm:

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

Theorem ([Littlestone, 1987]):

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

Note: There are concept classes for which the above four quantities are equal: if $C = \mathcal{P}(X)$, with X finite, then $VC(C) = \log_2(|C|) = |X|$.