### Two frameworks for analyzing learning algorithms

#### 1. Probably Approximately Correct (PAC) framework

- Identify classes of hypotheses that can/cannot be learned from a polynomial number of training samples
  - Finite hypothesis space
  - Infinite hypotheses (VC dimension)
- Define natural measure of complexity for hypothesis spaces (VC dimension) that allows bounding the number of training examples required for inductive learning

#### 2. Mistake bound (MB) framework

 Number of training errors made by a learner before it determines correct hypothesis

# Different learning settings considered in COLT

- 1. How training samples are generated?
  - Passive observation of random examples
  - Active querying by the learner
- 2. Noise in the data
  - Noisy
  - Error Free
- 3. Definition of success
  - Target concept must be learned exactly
  - Only probably and approximately
- 4. Assumptions made by the learner
  - For distribution of instances
  - Whether C ≤ H
- 5. Measure according to which learner is evaluated
  - No training examples, no of mistakes, total time

#### Mistake Bound (MB) Model Of Learning

- Problem setting:
  - Learner receives a sequence of training examples
  - Upon receiving each sample x, learner must predict target value c(x) before it is shown correct target value by trainer
- How many mistakes will learner make in its predictions before it learns the target concept?

## In the MB model, learning is in stages

#### In each stage:

- 1. Learner gets unlabeled example
- 2. Learner predicts classification
- 3. Learner is told correct label

Goal: Bound the total number of mistakes

#### Practical use of MB learning model

 Significant in practical settings where learning must be done while the system is in actual use, rather than in an off-line training stage

- Example: system to learn to approve credit card purchases based on data collected during use
  - How many mistakes in approving credit card purchases before system learns?
  - Total number of mistakes is more important than number of training examples

#### Study of MB Model: Problem Setting

 Number of mistakes made before learning the target concept exactly (rather than PAC)

#### • DEFINITION:

Algorithm A has mistake-bound M for learning class C if A makes at most M mistakes on any sequence that is consistent with a function in C

## Mistake Bound for Find-S algorithm

- Find-S
- Initialize h to the most specific hypothesis

$$\mathbf{l}_1 \wedge \neg \mathbf{l}_1 \wedge \mathbf{l}_2 \wedge \neg \mathbf{l}_2 \cdots \wedge \mathbf{l}_n \wedge \neg \mathbf{l}_n$$

- For each positive training instance x
  - Remove from h any literal that is not satisfied by x
- Output hypothesis h
- Total no. of mistakes can be at most n+1

#### Mistake Bound for *HALVING* Algorithm

- HALVING ALGORITHM:
  - predict using majority vote over all concepts in *C* consistent with past data
- Candidate Elimination and List-then-eliminate are halving algorithms
- Candidate Elimination
  - maintains a description of the version space, incrementally refining the version space as each new sample is encountered
  - Assume majority vote is used among all hypotheses in the current version space

# Mistake Bound for HALVING Algorithm

• Total no. of mistakes can be at most  $log_2/H/$ 

#### **Optimal Mistake Bound**

- For any target concept c, let M<sub>A</sub>(c) denote the maximum number of mistakes, over all possible sequences, made by learning algorithm A to exactly learn c
- Definition: The optimal mistake bound for C, denoted Opt(C) is the minimum over all possible learning algorithms A of  $M_A(C)$
- Then  $VC(C) \leq Opt(C) \leq log_2(/C/)$

#### Weighted Majority Algorithm

- A classifier combination method
- Takes a weighted vote among a pool of prediction algorithms, e.g., alternative hypotheses in H, or alternative learning algorithms
- It begins by weighting each algorithm by 1
- Whenever an algorithm misclassifies its weight is decreased by  $\beta$ , where  $0 \le \beta \le 1$

#### Weighted Majority Algorithm

- If A is any set of n prediction algorithms
- If k is the minimum number of mistakes made by any algorithm in A
- The number of mistakes made over any training sequence is at most

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