Introduction to Computational Learning Theory

- The classification problem
- Consistent Hypothesis Model
- Probably Approximately Correct (PAC) Learning

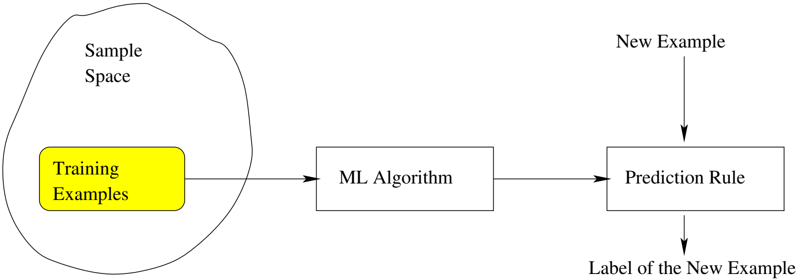
Outline

Don't Have a Good Definition, Only Examples

- Optical character recognition
- Spam filtering
- Document classification
- (IP) Packet filtering/classification
- Face detection
- Medical diagnosis
- Insider threat detection
- Stock price prediction
- Game playing (chess, go, etc.)

Classification Problems

- Input: set of labeled examples (spam and legitimate emails)
- Output: prediction rule (is this newly received email a spam email?)



Many examples on previous slide are classification problems.

Objectives

Numerous, sometimes conflicting:

- Accuracy
- Little computational resources (time and space)
- Small training set
- General purpose
- Simple prediction rule (Occam's Razor)
- Prediction rule "understandable" by human experts (avoid "black box" behavior)

Objectives

Numerous, sometimes conflicting:

- Accuracy
- Little computational resources (time and space)
- Small training set
- General purpose
- Simple prediction rule (Occam's Razor)
- Prediction rule "understandable" by human experts (avoid "black box" behavior)

Perhaps ultimately leads to an understanding of human cognition and the induction problem! (So far the reverse is "truer")

Objectives

Numerous, sometimes conflicting:

- Accuracy
- Little computational resources (time and space)
- Small training set
- General purpose
- Simple prediction rule (Occam's Razor)
- Prediction rule "understandable" by human experts (avoid "black box" behavior)

Perhaps ultimately leads to an understanding of human cognition and the induction problem! (So far the reverse is "truer")

Learning Model

In order to characterize these objectives mathematically, we need a mathematical model for "learning."

Outline

What Do We Mean by a Learning Model?

Definition (Learning Model)

is a mathematical formulation of a learning problem (e.g. classification)

What do we want the model to behave?

- Powerful (to capture REAL learning) and Simple (to be mathematically feasible). Oxymoron? Maybe not!
- By "powerful" we mean the model should capture, at the very least,
  1. What is being learned?
  2. Where/how do data come from?
  3. How's the data given to the learner? (offline, online, etc.)
  4. Which objective(s) to achieve/optimize? Under which constraints?

An Example: The Consistency Model

1. What is being learned?
   - Ω: a domain or instance space consisting of all possible examples
   - $c : \Omega \to \{0, 1\}$ is the target concept we want to learn
2. Where/how do data come from?
   - Data: a subset of m examples from Ω, along with their labels, i.e.

$$S = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \cdots, (\mathbf{x}_m, c(\mathbf{x}_m))\}$$

3. How's the data given to the learner? (offline, online, etc.)
   - S given offline
   - $\mathcal{C}$, a class of known concepts, containing the unknown concept c.
4. Which objective(s) to achieve/optimize? Under which constraints?
   - Output a hypothesis $h \in \mathcal{C}$ *consistent* with data, or output NO SUCH CONCEPT
   - Algorithm runs in polynomial time

Tricky Issues

- $|\mathcal{C}|$ is usually very large, could be exponential in m, or even infinite!
- How do we represent an element of $\mathcal{C}$? h in particular?
  - A *truth table* is out of the question, since Ω is huge
- For now, let's say
  - We agree in advance a particular way to represent $\mathcal{C}$
  - The representation of c in $\mathcal{C}$ has size $|c|$ (number of bits representing c
  - Each example $\mathbf{x} \in \Omega$ is of size $|\mathbf{x}| = O(n)$
  - ML algorithm required to run in time poly$(m, n, |c|)$.

Examples of CM-learnable and not CM-learnable concept clases

CM-learnable concept classes

- Monotone conjunctions
- Monotone disjunctions
- Boolean conjunctions
- k-CNF
- DNF
- Axis-aligned rectangles
- Separation hyperplanes

Concept classes which are NP-hard to learn

- k-term DNF
- Boolean threshold functions

Example 1: MONOTONE CONJUNCTIONS is Learnable

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = x_{i_1} \wedge x_{i_2} \cdots \wedge x_{i_q}, \quad 1 \leq q \leq n$$

Data looks like this:

x_1	x_2	x_3	x_4	x_5	$c(\mathbf{x})$
1	1	0	0	1	1
1	1	1	0	0	0
1	0	1	0	1	1
1	1	1	0	1	1
0	1	1	1	1	0

Output hypothesis $h = x_1 \wedge x_5$

- x_1 = "MS Word Running",
- x_5 = "ActiveX Control On",
- $c(\mathbf{x}) = 1$ means "System Down"

Example 2: MONOTONE DISJUNCTIONS is Learnable

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = x_{i_1} \vee x_{i_2} \cdots \vee x_{i_q}, \quad 1 \le q \le n$$

Data looks like this:

x_1	x_2	x_3	x_4	x_5	$c(\mathbf{x})$
1	1	0	0	1	1
0	0	1	0	0	0
1	0	1	0	1	1
1	1	1	0	1	1
0	0	1	1	1	0

Output hypothesis $h = x_1 \vee x_2$

Example 3: BOOLEAN CONJUNCTIONS is Learnable

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = x_{i_1} \wedge \bar{x}_{i_2} \wedge \bar{x}_{i_3} \wedge \cdots \wedge x_{i_q}, \quad 1 \leq q \leq n$$

Data looks like this:

x_1	x_2	x_3	x_4	x_5	$c(\mathbf{x})$
1	1	0	0	1	1
1	0	1	0	0	0
1	1	0	0	1	1
1	1	0	0	1	1
0	1	1	1	1	0

Output hypothesis $h = x_2 \wedge \bar{x}_3$

Example 4: k-CNF is Learnable

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = \underbrace{(\bullet \vee \cdots \vee \bullet)}_{\leq k \text{ literals}} \wedge \underbrace{(\bullet \vee \cdots \vee \bullet)}_{\leq k \text{ literals}} \wedge \cdots \wedge \underbrace{(\bullet \vee \cdots \vee \bullet)}_{\leq k \text{ literals}}$$

Data looks like this:

x_1	x_2	x_3	x_4	x_5	$c(\mathbf{x})$
1	0	0	0	1	1
1	0	1	0	0	0
1	0	1	1	1	1
1	0	0	0	1	1
0	1	1	1	1	0

Output hypothesis $h = (\bar{x}_2 \vee x_5) \wedge (\bar{x}_3 \vee x_4)$

Example 5: DNF is Learnable

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = (\bullet \wedge \cdots \wedge \bullet) \vee (\bullet \wedge \cdots \wedge \bullet) \vee \cdots \vee (\bullet \wedge \cdots \wedge \bullet)$$

Data looks like this:

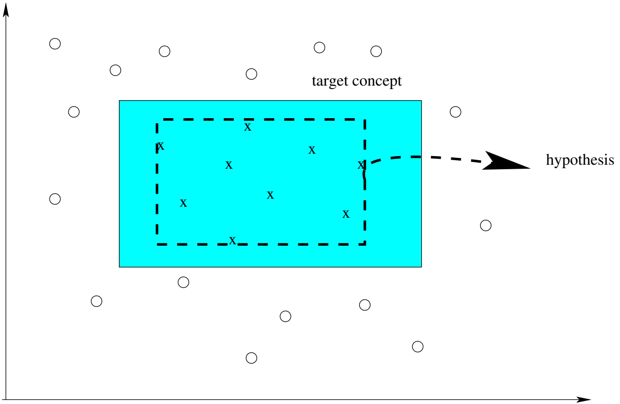
x_1	x_2	x_3	x_4	x_5	$c(\mathbf{x})$
1	0	0	0	1	1
1	0	1	1	1	1
1	0	1	0	0	0

Output hypothesis trivially is:

$$h = (x_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge \bar{x}_4 \wedge x_5) \vee (x_1 \wedge \bar{x}_2 \wedge x_3 \wedge x_4 \wedge x_5)$$

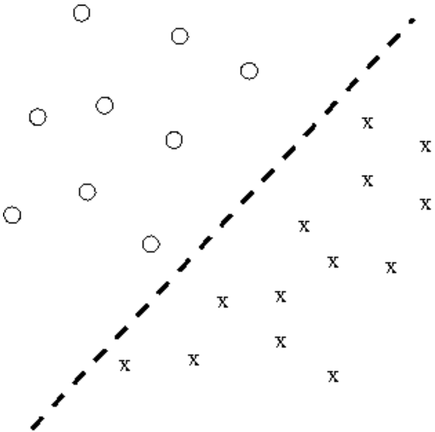
Example 6: AXIS-ALIGNED RECTANGLES is Learnable

$\mathcal{C}$ is the set of all axis-parallel rectangles



Example 7: SEPARATION HYPERPLANES is Learnable

$\mathcal{C}$ is the set of all hyperplanes on $\mathbb{R}^n$



Solvable with an LP-solver (a kind of algorithmic Farkas lemma)

Example 8: k-TERM DNF is Not Learnable, $\forall k \geq 2$

$\mathcal{C}$ = set of formulae on n variables $x_1, \dots, x_n$ of the form:

$$\varphi = \underbrace{(\bullet \wedge \cdots \wedge \bullet)}_{\text{term } 1} \vee \underbrace{(\bullet \wedge \cdots \wedge \bullet)}_{\text{term } 2} \vee \cdots \vee \underbrace{(\bullet \wedge \cdots \wedge \bullet)}_{\text{term } k}$$

Theorem

The problem of finding a k-term DNF formula consistent with given data S is **NP***-hard, for any $k \geq 2$.*

Proof.

Reduce 3-COLORING to this problem. $\square$

Example 9: THRESHOLD BOOLEAN FUNCTIONS is Not Learnable

- Each concept is represented by $\mathbf{c} \in \{0,1\}^n$ and $b \in \mathbf{N}$
- An example $\mathbf{x} \in \{0,1\}^n$ is positive if

$$c_1 x_1 + \cdots + c_n x_n \geq b.$$

Problems with the Consistency Model

- Does not take into account *generalization* (prediction performance)
- No noise involved (examples are never perfect)
- DNF is learnable but k-DNF is not?
- Strict consistency often means *over-fitting*

Outline

The PAC Model Informally

1. What to learn? Domain Ω, concept $c : \Omega \to \{0, 1\}$
2. Where/how do data come from?
   - Data: $S = \{(\mathbf{x}_1, c(\mathbf{x}_1)), \cdots, (\mathbf{x}_m, c(\mathbf{x}_m))\}$
   - Each $\mathbf{x}_i$ drawn from Ω according to some fixed but unknown distribution $\mathcal{D}$
3. How's the data given to the learner? (offline, online, etc.)
   - S given offline
   - Concept class $\mathcal{C}$ ($\ni c$) along with an implicit representation
4. Which objective(s) to achieve/optimize? Under which constraints?
   Efficiently output a hypothesis $h \in \mathcal{C}$ so that the generalization error

$$\text{err}_{\mathcal{D}}(h) := \underset{\mathbf{x} \in \mathcal{D}}{\text{Prob}}[h(\mathbf{x}) \neq c(\mathbf{x})]$$

is small with high probability.

The PAC Model: Preliminary Definition

Definition (PAC Learnability)

A concept class $\mathcal{C}$ is PAC learnable if there's an algorithm A (could be randomized) satisfying the following:

- for any $0 < \epsilon < 1/2$, $0 < \delta < 1/2$
- for any distribution $\mathcal{D}$ on Ω
- A draws m examples from $\mathcal{D}$, along with their labels
- A outputs a hypothesis $h \in \mathcal{C}$ such that

$$\text{Prob}\left[\text{err}_{\mathcal{D}}(h) \leq \epsilon\right] \geq 1 - \delta$$

Definition (Efficiently PAC Learnability)

If A also runs in time poly$(1/\epsilon, 1/\delta, n, |c|)$, then $\mathcal{C}$ is efficiently PAC learnable.

m must be poly$(1/\epsilon, 1/\delta, n, |c|)$ for $\mathcal{C}$ to be efficiently PAC learnable.

Some Initial Thoughts on the Model

- Still no explicit involvement of noise
  - However, intuitively if (example,label) error is relatively small, then the learner can deal with noise by reducing ϵ, δ.
- The requirement that the learner works for any $\mathcal{D}$ seems quite strong.
  - It's quite amazing that non-trivial concepts are learnable
- Can we do better for some problem if $\mathcal{D}$ is known in advance? Is there a theorem to this effect?
- The *i.i.d.* assumption (on the samples) is also somewhat too strong. This paper
  David Aldous, Umesh V. Vazirani: *A Markovian Extension of Valiant's Learning Model*, Inf. Comput. 117(2): 181-186 (1995)
  shows that the i.i.d. assumption can be relaxed a little.

Some examples

Efficiently PAC-learnable classes

- Boolean conjunctions
- Axis-aligned rectangles
- k-CNF
- k-DL (decision list, homework!)

Not PAC-learnable classes

- k-term DNF (that nasty guy again!)
- Boolean threshold functions
- Union of k half-spaces, $k \geq 3$
- DNF
- k-juntas

1) Boolean conjunctions is Efficiently PAC-Learnable

- Need to produce $h = l_1 \wedge l_2 \wedge \cdots \wedge l_k$, ($l_i$ are literals)
  - Start with $h = x_1 \wedge \bar{x}_1 \wedge \cdots \wedge x_n \wedge \bar{x}_n$
  - For each example $(\mathbf{a}, c(\mathbf{a}) = 1)$ taken from $\mathcal{D}$, remove from h all literals contradicting the example
  - E.g., if example is $(x_1 = 0, x_2 = 1, x_3 = 0, x_4 = 0, x_5 = 1, c(\mathbf{x}) = 1)$, then we remove literals $x_1, \bar{x}_2, x_3, x_4, \bar{x}_5$ from h (if they haven't been removed before)
- h always contain all literals of c, thus $c(\mathbf{a}) = 0 \Rightarrow h(\mathbf{a}) = 0$, $\forall \mathbf{a} \in \Omega$
- $h(\mathbf{a}) \neq c(\mathbf{a})$ iff $c(\mathbf{a}) = 1$ and $\exists$ a literal $l \in h - c$ s.t. $\mathbf{a}(l) = 0$.

$$
\begin{aligned}
\mathrm{err}_{\mathcal{D}}(h) &= \operatorname*{Prob}_{\mathbf{a}\in\mathcal{D}}[h(\mathbf{a}) \neq c(\mathbf{a})] \\
&= \operatorname*{Prob}_{\mathbf{a}\in\mathcal{D}}\left[c(\mathbf{a}) = 1 \wedge \mathbf{a}(l) = 0 \text{ for some } l \in h - c\right] \\
&\leq \sum_{l \in h - c} \underbrace{\operatorname*{Prob}_{\mathbf{a}\in\mathcal{D}}\left[c(\mathbf{a}) = 1 \wedge \mathbf{a}(l) = 0\right]}_{p(l)}
\end{aligned}
$$

1) Boolean conjunctions is Efficiently PAC-Learnable

- So, if $p(l) \le \epsilon/2n, \forall l \in h - c$ then we're OK!
- How many samples from $\mathcal{D}$ must we take to ensure all $p(l) \le \epsilon/2n, \forall l \in h - c$ with probability $\ge 1 - \delta$?
- Consider an $l \in h - c$ for which $p(l) > \epsilon/2n$, call it a bad literal
- l will be removed with probability $p(l)$
- l survives m samples with probability at most $(1 - p(l))^m < (1 - \epsilon/2n)^m$
- Some bad literal survives with probability at most

$$2n\left(1 - \epsilon/2n\right)^m \le 2ne^{-\epsilon m/2n} \le \delta$$

if

$$m \ge \frac{2n}{\epsilon}\left(\ln(2n) + \ln(1/\delta)\right)$$

2) k-CNF is Efficiently PAC-Learnable

- Say $k = 3$
- We can reduce learning 3-CNF to learning (monotone) CONJUNCTIONS
- For every triple of literals u, v, w, create a new variable $y_{u,v,w}$, for a total of $O(n^3)$ variables
- Basic idea:

$$(u \vee v \vee w) \quad \Leftrightarrow \quad y_{u,v,w}$$

- Each example from 3-CNF can be transformed into an example for the CONJUNCTIONS problem under variables $y_{u,v,w}$
- A hypothesis h' for CONJUNCTIONS can be transformed back easily.

3) AXIS PARALLEL RECTANGLES is Efficiently PAC-Learnable

- The algorithm is like in the consistency model
- Error is the area-difference between target rectangle c and hypothesis rectangle h
- "Area" is measured in density according to $\mathcal{D}$
- Hence, even with area ϵ, the probability that all m samples misses the area is $(1-\epsilon)^m$
- Only need $m \geq (1/\epsilon)\ln(1/\delta)$

4) k-TERM DNF is **Not** Efficiently PAC-Learnable ($k \geq 2$)

- Pitt and Valiant in
  Leonard Pitt and Leslie G. Valiant. Computational limitations on learning from examples. Journal of the ACM, 35(4):965-984, October 1988
  showed that k-TERM DNF is not efficiently learnable unless $\mathbf{RP} = \mathbf{NP}$

The PAC Model: Informal Revision

- Troubling: k-TERM DNF $\subseteq$ k-CNF but the latter is learnable and the former is not.
- Representation matters a great deal!
- We should allow the algorithm to output a hypothesis represented differently from $\mathcal{C}$
- Particular, let $\mathcal{H}$ be a hypothesis class which is "more expressive" than $\mathcal{C}$
  ("more expressive" $=$ every c can be represented by some h)
- $\mathcal{C}$ is PAC-learnable using $\mathcal{H}$ if blah blah blah and allow output $h \in \mathcal{H}$

The PAC Model: Final Revision

Definition (PAC Learnability)

A concept class $\mathcal{C}$ is PAC learnable using a hypothesis class $\mathcal{H}$ if there's an algorithm A (could be randomized) satisfying the following:

- for any $0 < \epsilon < 1/2$, $0 < \delta < 1/2$
- for any distribution $\mathcal{D}$ on Ω
- A draws m examples from $\mathcal{D}$, along with their labels
- A outputs a hypothesis $h \in \mathcal{H}$ such that

$$\text{Prob}\left[\text{err}_{\mathcal{D}}(h) \leq \epsilon\right] \geq 1 - \delta$$

If A also runs in time $\text{poly}(1/\epsilon, 1/\delta, n, \text{size}(c))$, then $\mathcal{C}$ is efficiently PAC learnable.

We also want each $h \in \mathcal{H}$ to be efficiently evaluatable. This is implicit!

Let's Summarize

- 1-TERM DNF (i.e. CONJUNCTIONS) is efficiently PAC-learnable using 1-TERM DNF
- k-TERM DNF is not efficiently PAC-learnable using k-TERM DNF, for any $k \geq 2$
- k-TERM DNF is efficiently PAC-learnable using k-CNF, for any $k \geq 2$
- k-CNF is efficiently PAC-learnable using k-CNF, for any $k \geq 2$
- AXIS PARALLEL RECTANGLES (natural representation) is efficiently PAC-learnable

More Hardness Results

- Blum and Rivest (*Neural Networks*, 1989): 3-NODE NEURAL NETWORKS is **NP**-hard to PAC-learn
- Alekhnovich et al. (FOCS 04): some classes of Boolean functions and decision trees are hard to PAC-learn
- Feldman (STOC 06): DNF is not learnable, even with membership querying
- Guruswami and Raghavendra (FOCS 06): learning half-spaces (perceptron) with noise is hard

Main reason: we made no assumption about $\mathcal{D}$, hence these are worst case results.

Contrast with the Baysian View

- It should not be surprising that some concept classes are not learnable, because computational learning theory, like other areas taking the computational viewpoint, are based on *worst-case complexity*.
- The Baysian viewpoint imposes a *prior* distribution over the concept class