Theoretical Approaches to Machine Learning

Early work (e.g., Gold) ignored efficiency
- Only considers computability
- "Learning in the limit"

Later work considers tractable inductive learning
- With high probability, approximately learn
- Polynomial runtime, polynomial # of examples needed
- Results (usually) independent of probability distribution for the examples
**Identification in the Limit**

**Definition**  After some finite number of examples, learner will have learned the correct concept (though might not even know it!). Correct means agrees with target concept on labels for all data.

**Example**  Consider noise-free learning from the class \( \{f \mid f(n) = a^n \mod b\} \) where \( a \) and \( b \) are natural numbers.

**General Technique**  “Innocent Until Proven Guilty”

Enumerate all possible answers
Search for simplest answer consistent with training examples seen so far; sooner or later will hit solution
Some Results (Gold)

- Computable languages (Turing machines) can be learned in the limit using inference by enumeration.
- If data set is limited to positive examples only, then only finite languages can be learned in the limit.
Solution for \( \{ f \mid f(n) = a \cdot n \mod b \} \)

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>17</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Here, \(a\) and \(b\) are the parameters for the function, and \(f(n)\) represents the output for a given input \(n\). The table shows how different values of \(a\) and \(b\) affect the output \(f(n)\) for \(n = 9\) and \(b = 17\) respectively.
The Mistake-Bound Model (Littlestone)

Framework

- Teacher shows input I
- ML algorithm guesses output O
- Teacher shows correct answer
- Can we upper bound the number of errors the learner will make?
The Mistake-Bound Model

Example Learn a conjunct from $N$ predicates and their negations

- Initial $h = p_1 \land \neg p_1 \land \ldots \land p_n \land \neg p_n$
- For each $+$ ex, remove the remaining terms that do not match
The Mistake-Bound Model

Worst case # of mistakes?

$1 + N$

- First $+ ex$ will remove $N$ terms from $h_{initial}$

- Each subsequent error on a $+ \text{ will remove at least one more term (never make a mistake on - ex’s)}$
Equivalence Query Model (Angluin)

Framework

- ML algorithm guesses concept: is target equivalent to this guess?
- Teacher either says “yes” or returns a counterexample (example labeled differently by target and guess)
- Can we upper bound the number of errors the learner will make?
- Time to compute next guess bounded by Poly(|data seen so far|)
Probably Approximately Correct (PAC) Learning

PAC learning (Valiant ‘84)

Given

\[ X \]  domain of possible examples
\[ C \]  class of possible concepts to label X
\[ c \in C \]  target concept
\[ \delta, \varepsilon \]  correctness bounds
Probably Approximately Correct (PAC) Learning

- For any target \( c \) in \( C \) and any distribution \( D \) on \( X \)
- Given at least \( N = \text{poly}(|c|, 1/\varepsilon, 1/\delta) \) examples drawn randomly, independently from \( X \)
- Do with probability \( 1 - \delta \), return an \( h \) in \( C \) whose accuracy is at least \( 1 - \varepsilon \)
- In other words

\[
\text{Prob}[\text{error}(h, c) > \varepsilon] < \delta
\]

- In time polynomial in \( |\text{data}| \)

Shaded regions are where errors occur:
Relationships Among Models of Tractable Learning

- Poly mistake-bounded with poly update time = EQ-learning
- EQ-learning implies PAC-learning
  - Simulate teacher by poly-sized random sample; if all labeled correctly, say “yes”; otherwise, return incorrect example
  - On each query, increase sample size based on Bonferoni correction
To Prove Concept Class is PAC-learnable

1. Show it’s EQ-learnable, OR
2. Show the following:
   • There exists an efficient algorithm for the consistency problem (find a hypothesis consistent with a data set in time poly in the size of the data set)
   • Poly-sized sample is sufficient to give us our accuracy guarantees

© Jude Shavlik 2006
David Page 2010
Useful Trick:

A Maclaurin series: for $-1 < x \leq 1$:

$$\ln(1+x) = x - \frac{1}{2} x^2 + \frac{1}{3} x^3 - \frac{1}{4} x^4 + ...$$

We only care about $-1 < x < 0$. Rewriting $-x$ as $\epsilon$, we can derive that for $0 < \epsilon < 1$: $\ln(1-\epsilon) \leq -\epsilon$.

Because $\epsilon$ is positive, we can put both sides of this last inequality into an exponent to obtain:

$$1 - \epsilon \leq e^{-\epsilon}$$
Assume EQ Query Bound (or Mistake Bound) $M$

- On each equivalence query, draw: $\frac{1}{\epsilon} \ln(M/\delta)$ examples
- What is the probability that on any of the at most $M$ rounds, we accept a "bad" hypothesis?
- At most $M (1-\epsilon)^{\frac{1}{\epsilon} \ln(M/\delta)} \leq M e^{-\ln(M/\delta)}$ (using the useful trick) $= \delta$
If Algorithm Doesn’t Know \( M \) in Advance (recall \( |c| \)):

- On \( i \)th equivalence query, draw:
  \[
  \frac{1}{\epsilon} \left( \ln \left( \frac{1}{\delta} \right) + i \ln(2) \right) \text{ examples}
  \]
- What is the probability that we accept a “bad” hypothesis at one query?
- \[
  (1-\epsilon)^{\frac{1}{\epsilon} \left( \ln \left( \frac{1}{\delta} \right) + i \ln(2) \right)} \leq e^{- \left( \ln \left( \frac{1}{\delta} \right) + i \ln(2) \right)}
  \]
  (using the useful trick) \( = \delta/2^i \)
- So probability we ever accept a “bad” hypothesis is at most \( \delta \).
Using First Method: kDNF

- Write down disjunction of all conjunctions of at most $k$ literals (features or negated features)
- Any counterexample will be actual negative
- Repeat until correct:
  
  Given a counterexample, delete disjuncts that cover it (are consistent with it)
Using Second Method

- If hypothesis space is finite, can show a poly sample is sufficient
- If hypothesis space is parameterized by \( n \), and grows only exponentially in \( n \), can show a poly sample is sufficient
How Many Examples Needed to be PAC?

- Consider finite hypothesis spaces
- Let $H_{bad} \equiv \{h_1, \ldots, h_z\}$
  - The set of hypotheses whose (“testset”) error is $> \varepsilon$
  - Goal: Eliminate all items in $H_{bad}$ via (noise-free) training examples
How Many Examples Needed to be PAC?

How can an $h$ look bad, even though it is correct on all the training examples?

- If we never see any examples in the shaded regions
- We’ll compute an $N$ s.t. the odds of this are sufficiently low (recall, $N$ = number of examples)
Consider $H_1 \in H_{bad}$ and $ex \in N$

What is the probability that $H_1$ is consistent with $ex$?

$$\text{Prob}[\text{consistent}_A(ex,H_1)] \leq 1 - \varepsilon$$

(since $H_1$ is bad its error rate is at least $\varepsilon$)
What is the probability that $H_1$ is consistent with all $N$ examples?

$$\text{Prob}[\text{consistent}_B(N,H_1)] \leq (1 - \varepsilon)^{|N|}$$

(by iid assumption)
What is the probability that some member of $H_{bad}$ is consistent with the examples in $N$?

\[
\text{Prob}[\text{consistent}_C(N, H_{bad})] = \\
\text{Prob}[\text{consistent}_B(N, H_1) \lor \ldots \lor \text{consistent}_B(N, H_z)] \\
\leq |H_{bad}| \times (1-\varepsilon)^{|N|} \quad // \quad P(A \lor B) \leq P(A) + P(B) - P(A \land B) \\
\leq |H| \times (1-\varepsilon)^{|N|} \quad // \quad H_{bad} \subseteq H
\]
Solving for $|N|$

We have

$$\text{Prob}[\text{consistent}_C(N, H_{bad})] \leq |H| \times (1-\varepsilon)^{|N|} < \delta$$

Recall that we want the prob of a bad concept surviving to be less than $\delta$, our bound on learning a poor concept.

Assume that if many consistent hypotheses survive, we get unlucky and choose a bad one (we’re doing a worst-case analysis)
Solving for $|N|$  
(number of examples needed to be confident of getting a good model)

Solving

$$|N| > \left[ \ln\left(\frac{1}{\delta}\right) + \ln(|H|) \right] / -\ln(1-\varepsilon)$$

Since $\varepsilon \leq -\ln(1-\varepsilon)$ over $[0,1)$ we get

$$|N| > \left[ \ln\left(\frac{1}{\delta}\right) + \ln(|H|) \right] / \varepsilon$$

(Aside: notice that this calculation assumed we could always find a hypothesis that fits the training data)

Notice we made NO assumptions about the prob dist of the data (other than it does not change)
**Example:**

**Number of Instances Needed**

**Assume**

- $F = 100$ binary features
- $H = \text{all (pure) conjuncts}$
  
  $[3^F \text{ possibilities } (\forall i, \text{ use } f_i, \text{ use } \neg f_i, \text{ or ignore } f_i)]$
  
  so $\lg |H| = F \times \lg 3 \approx F$

- $\varepsilon = 0.01$
- $\delta = 0.01$

$$N = \left[ \ln\left( \frac{1}{\delta} \right) + \ln(\lvert H \rvert) \right] / \varepsilon = 100 \times \left[ \ln(100) + 100 \right] \approx 10^4$$

But how many real-world concepts are pure conjuncts with noise-free training data?
Two Senses of Complexity

Sample complexity
(number of examples needed)

VS.

Time complexity
(time needed to find $h \in H$ that is consistent with the training examples)
Complexity (cont.)

- Some concepts require a polynomial number of examples but an exponential amount of time (in the worst case)

- Eg, training neural networks is NP-hard (recall BP is a “greedy” algorithm that finds a local min)
Dealing with Infinite Hypothesis Spaces

- Can use the Vapnik-Chervonenkis (‘71) dimension (VC-dim)
- Provides a measure of the capacity of a hypothesis space

\[
VC\text{-dim} = \text{given a hypothesis space } H, \text{ the VC-dim is the size of the largest set of examples that can be completely fit by } H, \text{ no matter how the examples are labeled}
\]
VC-dim Impact

- If the number of examples $\ll$ VC-dim, then memorizing training is trivial and generalization likely to be poor.

- If the number of examples $\gg$ VC-dim, then the algorithm must generalize to do well on the training set and will likely do well in the future.
Samples of VC-dim

Finite $H$

$\text{VC-Dim} \leq \log_2 |H|$

(if $d$ examples, $2^d$ different labelings possible, and must have $2^d \leq |H|$ if all functions are to be in $H$)
An Infinite Hypothesis
Space with a Finite VC Dim

H is set of lines in 2D

Can cover 1 ex no matter how labeled
Example 2 (cont.)

Can cover 2 ex’s no matter how labeled
Example 2 (cont.)

Can cover 3 ex’s no matter how labeled
Example 2 (cont.)

Cannot cover/separate if 1 and 4 are +,
But 2 and 3 are – (our old friend, ex-or)

\[
\begin{align*}
|H| &= \infty \\
\text{but VC-dim} &= 3
\end{align*}
\]

For N-dimensions and N-1 dim hyperplanes,
\[
\text{VC-dim} = N + 1
\]
More on “Shattering”

What about collinear points?

If there exists some set of $d$ examples that $H$ can fully fit all labellings of these $d$ examples then $\text{VC}(H) \geq d$
Some VC-Dim Theorems

**Theorem**  H is PAC-learnable iff its VC-dim is finite

**Theorem**  Sufficient to be PAC to have # of examples

\[> \frac{1}{\varepsilon} \max[4\ln(2/\delta), 8\ln(13/\varepsilon)\text{VC-dim}(H)]\]

**Theorem**  Any PAC algorithm needs at least

\[\Omega(\frac{1}{\varepsilon}[\ln(1/\delta) + \text{VC-dim}(H))]\] examples

[No need to memorize these for the exam]
To Show a Concept is NOT PAC-learnable

- Show the consistency problem is NP-hard (hardness assumes \( P \neq NP \)), OR
- Show the VC-dimension grows at a rate not bounded by any polynomial
Be Careful

- It can be the case that consistency is hard for a concept class, but not for a larger class
  - Consistency NP-hard for k-term DNF
  - Consistency easy for DNF (PAC still open ques.)
- More robust negative results are for PAC-prediction
  - Hypothesis space not constrained to equal concept class
  - Hardness results based on cryptographic assumptions, such as assuming efficient factoring is impossible
Some Variations in Definitions

- Original definition of PAC-learning required:
  - Run-time to be polynomial
  - Hypothesis to be within concept class (otherwise, called PAC-prediction)
- Now this version is often called *polynomial-time, proper* PAC-learning
- *Membership queries*: can ask for labels of specific examples
Some Results...

- Can PAC-learn $k$-DNF (exponential in $k$, but $k$ is now a constant)
- Can *not* properly PAC-learn $k$-term DNF, but can PAC-learn it by $k$-CNF
- Can *not* PAC-learn Boolean Formulae unless can crack RSA (can factor fast)
- Can PAC-learn decision-trees with the addition of membership queries
- Unknown whether can PAC-learn DNF or DTs (“holly grail” questions of COLT)
Some Other COLT Topics

COLT
+ clustering
+ k-NN
+ RL
+ EBL (ch. 11 of Mitchell)
+ SVMs
+ ILP
+ ANNs, etc.

• Average case analysis (vs. worst case)
• Learnability of natural languages (language innate?)
• Learnability in parallel
Summary of COLT

Strengths

- Formalizes learning task
- Allows for imperfections (e.g. \( \epsilon \) and \( \delta \) in PAC)
- Work on *boosting* (later) is excellent case of ML theory influencing ML practice
- Shows what concepts are intrinsically hard to learn (e.g. k-term DNF)
Summary of COLT

Weaknesses

- Most analyses are worst case
- Use of “prior knowledge” not captured very well yet