Lecture 3: Computational Learning Theory

- Concept learning revisited
- Approximate learning
- PAC learning
- Other COLT directions

Recall: Concept Learning Task

Given:

- The set of all possible instances X
- A target function (or concept) $c : X \rightarrow \{0, 1\}$
- A set of hypotheses H
- A set of training examples D (containing positive and negative examples of the target function

$$\langle x_1, c(x_1) \rangle, \ldots \langle x_m, c(x_m) \rangle$$

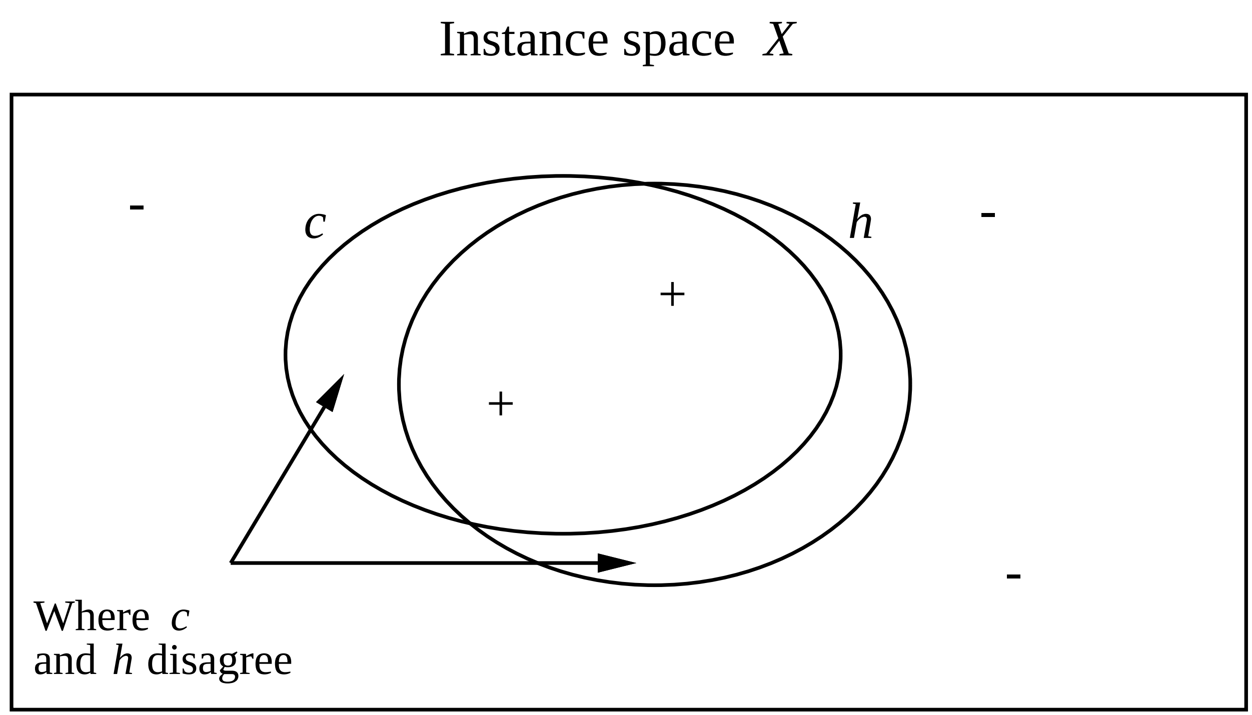
Determine:

A hypothesis h in H such that $h(x) = c(x)$ for all x in X.

Approximate Concept Learning

- Requiring a learner to acquire the *right* concept is too strict
- Instead, we will allow the learner to produce a **good approximation** to the actual concept
- For any instance space, there is a **non-uniform likelihood** of seeing different instances
- We assume that there is a **fixed probability distribution** P on the space of instances X
- The learner is trained and tested on examples drawn **independently and randomly**, according to P.

True Error of a Hypothesis



The set of instances on which the concept and the hypothesis disagree is denoted:

$$S = \{x | h(x) \neq c(x)\}$$

The **true error** of h with respect to c is:

$$\sum_{x \in S} P(x)$$

This is the probability of making an error on an instance randomly drawn from X according to P

Let $\epsilon \in (0, 1)$ be an error tolerance parameter. We say that h is a **good approximation** of c (to within ϵ) if and only if the true error of h is less than ϵ.

Two Notions of Error

Training error of hypothesis h with respect to target concept c:

- How often $h(x) \neq c(x)$ over the training instances

True error of hypothesis h with respect to target concept c:

- How often $h(x) \neq c(x)$ over future, unseen instances (but drawn according to P)

Can we bound the true error of a hypothesis given only its training error?

How many examples are needed to achieve a good approximation (in terms of the true error)?

Example: Rote Learner

Let $X = \{0,1\}^n$. Let P be the uniform distribution over X.
Let the concept c be generated by randomly assigning a label to every instance in X.
Let $D \subset X$ be a set of training instances.
The hypothesis h is generated by memorizing D and giving a random answer otherwise.

- What is the training error of h?
- What is the true error of h?

Approximate Learning using Version Spaces

A version space is **exhausted** if the *S*=*G* and both are singleton sets.

Consider a given hypothesis space H, target concept c, sequence of examples D and error tolerance ϵ.

A version space is called ϵ**-exhausted** if it does not contain any hypothesis with **true** error more than ϵ.

We will only require that the learner produce an ϵ-exhausted version space.

Probabilistic Learning Guarantees

Another relaxation: we only require the learner to ϵ-exhaust the version space **with high probability**.

We introduce a confidence parameter δ, and require the learner to ϵ-exhaust the version space with probability at least $(1-\delta)$.

We are now requiring **probably approximately correct (PAC) learning**.

How many examples are needed for a learner to ϵ-exhaust a version space with probability greater than $(1-\delta)$?

Sample Complexity for PAC-Version Spaces

Theorem: (Haussler, 1988) Let H be a finite set of hypothesis. Let $c \in H$ be any concept and consider m independent training examples drawn according to P. For any error tolerance $\epsilon \in (0, 1)$, the probability that the version space consistent with the m examples is not ϵ-exhausted is $\leq |H|e^{-\epsilon m}$.

Proof: Let $h_1, ..h_k$ be the hypotheses with error $> \epsilon$. The version space is not exhausted if one of these hypotheses is consistent with all m training examples.
Since h_i has error ϵ, an individual example is consistent with h_i with probability $< (1 - \epsilon)$.
Since the examples are independent, the probability that h_i is

consistent with all of them is $< (1-\epsilon)^m$.
Since there are k hypotheses with high error, the probability of any one of them being consistent with all the examples is $< k(1-\epsilon)^m$.
But $k < |H|$ and $(1-\epsilon)^m < e^{-\epsilon m}$, and the result follows.

How Many Examples are Needed for PAC Learning?

Corrolary: Given confidence parameter δ and error tolerance ϵ, the number of examples needed to ϵ-exhaust a version space consistent with any concept $c \in H$ is:

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

Proof: From the theorem, we have:

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

Taking logs on both sides, $\ln \delta \geq \ln |H| - \epsilon m$

By algebraic manipulation, we get the desired bound.

Note that the bound is logarithmic in the size of the hypothesis space!

Example: Conjunctions of Boolean Literals

Let H be the space of all pure conjunctive formulae over n Boolean attributes.

Then $|H| = 3^n$ (why?)

From the previous result, we get:

$$m \geq \frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + \ln 3^n\right) = \frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + n\ln 3\right)$$

This is linear in n!

Example: Unbiased Learner

The hypothesis space is the power set of X. For n Boolean attributes, we get $|H| = 2^{2^n}$

By using the previous formula:

$$m \geq \frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + \ln 2^{2^n}\right) = \frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + 2^n \ln 2\right)$$

An unbiased learner requires an exponential number of examples.

Probably Approximately Correct (PAC) Learning

Let C be a concept class defined over a set of instances X in which each instance has length n. An algorithm L, using hypothesis class H is a **PAC learning algorithm** for C if:

- for any concept $c \in C$
- for any probability distribution P over X
- for any parameters $0 < \epsilon < 1/2$ and $0 < \delta < 1/2$

the learner L will, with probability at least $(1 - \delta)$, output a hypothesis with error at most ϵ.

A class of concepts C is **PAC-learnable** if there exists a PAC learning algorithm for C.

Computational vs Sample Complexity

A class of concepts is **polynomial-sample PAC-learnable** if it is PAC learnable using a number of examples at most polynomial in $\frac{1}{\epsilon}$, $\frac{1}{\delta}$ and n.

A class of concepts is **polynomial-time PAC-learnable** if it is PAC learnable in time at most polynomial in $\frac{1}{\epsilon}$, $\frac{1}{\delta}$ and n.

Sample complexity is often easier to bound than time complexity!

Example: K-Term DNF and CNF Formulae

A K-term DNF expression has the form $T_1 \vee \ldots \vee T_k$ where each t_i is a conjunction over n Boolean attributes and their negations.

The size of H is at most $k3^n$, so using our prior PAC bound, we obtain:

$$m \geq \frac{1}{\epsilon}\left(\ln\frac{1}{\delta} + n\ln 3 + \ln k\right)$$

But computation time is not polynomial! (shown to be equivalent to graph coloring - see Kearns & Vazirani).

K-term CNF formulae are polynomial-sample and polynomial-time learnable, though.

K-term CNF is a conjunction $T_1 \wedge \ldots \wedge T_j$ where each T_i is a disjunction of at most K Boolean attributes.

K-CNF formulae are strictly more expressive than K-DNF!

Agnostic Learning

What if we lift the assumption that $c \in H$?

In this case, we study the true error of the hypothesis with the lowest error on the training data

A similar result to the previous PAC-learning theorem can be obtained using Hoeffding (Chernoff) bounds, which relate the true probability of an event to its observed frequency over m independent trials.

The probability of the true error being greater than $\epsilon +$ training error is at most $e^{-2m\epsilon^2}$

By a proof similar to the one described before, we get:

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

This is similar to the previous bound, except it grows with the square of $\frac{1}{\epsilon}$.

Bird Eye View of Computational Learning Theory

1. How hard is it to learn (in terms of the computation required)?
   Difficult to answer in general, but results have been established for simple problems (e.g. learning CNF and DNF formulae)
2. How many examples are required for a good approximation?
   A lot of results here, regarding sample complexity bounds for different algorithms
3. What problems can be solved by a given algorithm?
   Little work done here so far.

Different Models of Learning

- Examples come randomly from some fixed distribution (the case usually considered in supervised learning)
- The learner is allowed to ask questions to the teacher (active learning)
- Examples are given by an opponent (on-line learning, mistake-bound model)

Most of the time assumes that the examples are noise-free.

However, results do exist for particular kinds of noise (e.g. classification noise).

What if H is infinite?...