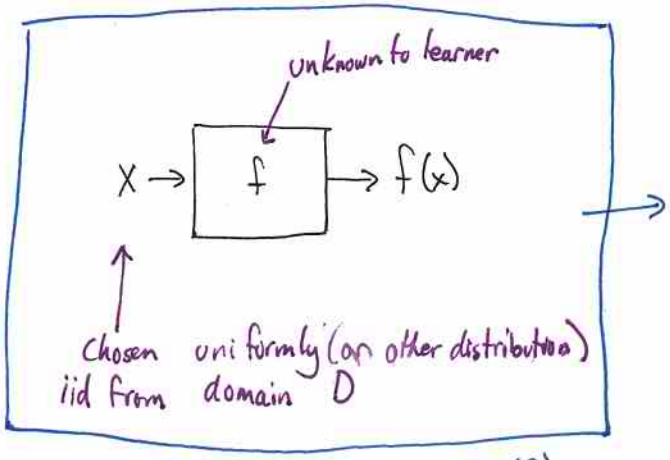
Learning

Learn from random, uniform examples: How do we formalize?



$x_1, f(x_1)$
$x_2, f(x_2)$
⋮

m random labelled examples

After seeing several examples, learner should output hypothesis h.

- hopefully h=f
- is that asking too much?
- how about $dist(h,f) < \varepsilon$?

what is distance?

e.g. $\Pr_{x \in D}[h(x) \neq f(x)]$?

but then what distribution on D?

Today: uniform

In general: match distribution of examples

Valiant's PAC model

"Probably Approximately Correct"

def. given hypothesis h, error of h wrt f is

$$\text{error}(h) = \Pr_{x \in_u D}[f(x) \neq h(x)]$$

Note: this is defn wrt uniform. In general, this is the same as distribution on D from example oracle

often will use:
f is ε-close to h wrt D if $\Pr_{x \in_u D}[f(x) \neq h(x)] \leq \varepsilon$

Note if f is arbitrary, there is nothing you can do! (ie can't learn a random fctn)
However, if you know something about f, there may be hope!

What if you know that f is from a family of functions

def. uniform distribution learning algorithm for concept class C is algorithm A s.t.

- A is given $\varepsilon, \delta > 0$ access to $Ex(f)$ for $f \in C$
- A outputs h s.t. with prob $\geq 1-\delta$ error(h) wrt f is $\leq \varepsilon$

h is ε-close to f

Parameters of Interest

- m — # samples used by A — "sample complexity"
- ε — accuracy parameter
- δ — confidence parameter
- runtime? — hope for $\text{poly}(\log(\text{domain size}), \frac{1}{\varepsilon}, \frac{1}{\delta})$
- description of h?
  - should it be similar to description of f?
    (proper learning)
  - at least should be relatively compact
    $O(\log |C|)$ + efficient to evaluate

Remarks

- as before, dependence on δ needn't be more than $O(\log(1/\delta))$.
  why?
- Uniform case is special case of PAC-model:
  Given $EX_{\mathcal{D}}(f)$ for unknown $\mathcal{D}$
  output h with small error according to same $\mathcal{D}$ (some $\mathcal{D}$ can be harder than others)

Ignoring Runtime

Occam's Razor

learning is easy!

i.e. can easily achieve small sample complexity

Brute Force Algorithm

- Draw $M = \frac{1}{\varepsilon}\left(\ln |\mathcal{C}| + \ln \frac{1}{\delta}\right)$ uniform examples
- Search over all $h \in \mathcal{C}$ until find one that labels all examples correctly & output it.
  (choose arbitrarily if ≥ 1 such h works)

Behavior:

What should behavior be?

- f is a good thing to output ✓
- what is a bad thing to output?

h is "bad" if error(h) wrt $f \geq \varepsilon$

$\Pr$[bad h consistent with examples] $\leq (1-\varepsilon)^M$

$\Pr$[any bad h consistent with examples]

$\leq |\mathcal{C}|\,(1-\varepsilon)^M$ ← union bnd

$\leq |\mathcal{C}|\,(1-\varepsilon)^{\frac{1}{\varepsilon}\left(\ln |\mathcal{C}| + \ln \frac{1}{\delta}\right)}$

$\leq \delta$

∴ unlikely to output any bad h

[Does the Bible really predict JFK's assasination?]

Comments

- proof didn't use anything special about uniform distribution

  works for any $\mathcal{D}$,
  as long as error defined w.r.t. same $\mathcal{D}$ as sample generator

- once we have a good h

  1) can predict values of f on new inputs since $\Pr_{x \in \mathcal{D}}[f(x)=h(x)] \geq 1-\delta$

     random ~ according to $\mathcal{D}$

  2) can compress description of samples

     $(x_1, f(x_1))\ (x_2, f(x_2)) \ldots (x_m, f(x_m))$ $\quad m(\log|\mathcal{D}| + \log|R|)$ (range of f)

     $\Downarrow$

     $x_1 \ldots x_m$, description of h $\quad m \cdot \log|\mathcal{D}| + \log|\mathcal{C}|$ bits

so learning, prediction & compression are related.

learning $\Rightarrow$ prediction & compression

formal relations in other direction too

Occam's Razor: simplest explanation is best

An efficient learning algorithm

C = conjunctions over $\{0,1\}^n$

ie. $f(x) = x_i x_j \bar{x}_k$

- can't hope for 0-error from subexponential # of random examples
  - eg. how to distinguish $f(x) = x_1 \cdots x_n$ from $f(x) = 0$?
- Brute force: $M = \frac{1}{\varepsilon}\left(\ln(2^n) + \ln\frac{1}{\delta}\right)$ examples & as much time
- Poly time algorithm:
  - draw poly$(1/\varepsilon)$ random examples to estimate $\Pr[f(x)=1]$ to additive error $\pm \frac{\varepsilon}{4}$
    - if estimate $< \varepsilon/2$, output "$h(x)=0$"
  - since estimate $\geq \varepsilon/2$ & error $\leq \varepsilon/4$
    - $\Pr[f(x)=1] \geq \varepsilon/4$
    - so, every $O(1/\varepsilon)$ examples see new random "positive" example (expected) } just look at these
  - in set of positive examples
    - let $V = \{$vars set same way in each example$\}$
    - output $h(x) = \bigwedge_{i \in V} X_i^{b_i}$ ← b_i tells us if i complemented or not

behavior of algorithm:

for i in conjunction:

must be set same way in each positive example $\Rightarrow$ in V

for i not in conjunction:

$$\Pr[i \in V] \leq \Pr[i \text{ set same in each of } k \text{ positive examples}]$$

$$\leq \frac{1}{2^{k-1}}$$

$$\Pr[\text{any } i \text{ that not in conjunction manages to survive}]$$

$$\leq \frac{n}{2^{k-1}}$$

$$\leq \delta \quad \text{if pick } k = \log \frac{n}{\delta}$$

So $\Omega(\log \frac{n}{\delta})$ positive examples

$+ \quad \Omega(\frac{1}{\varepsilon} \log \frac{n}{\delta})$ total examples suffice!

Learning via Fourier Representation

learning algorithms based on estimating Fourier representation of fctn f (similar to poly interpolation)

Approximating one Fourier coefficient:

lemma can approx any specific Fourier coeff S to w/in additive γ (i.e. $|\text{output} - \hat{f}(S)| \leq \gamma$) with prob $\geq 1-\delta$ in $O(\frac{1}{\gamma^2}\log\frac{1}{\delta})$ samples

Note no queries needed!!

Pf. Chernoff + $\hat{f}(S) = 2\underbrace{\Pr_x[f(x)=\chi_S(x)]}_{\text{estimate this}} - 1$ ∎

Can we find any or all heavy coefficients?

there are exponentially many coefficients.

Can use same samples for all coeffs, but must union bnd prob of error on any of them

using $\delta = \frac{1}{2^n}$, gives $O(\frac{1}{\gamma^2}\cdot n)$ samples, but exp runtime.

queries can help a lot!

What if we "know where to look" for heavy coefficients?

e.g. all heavy coeffs are in "low degree" coeffs?

If so, can search!