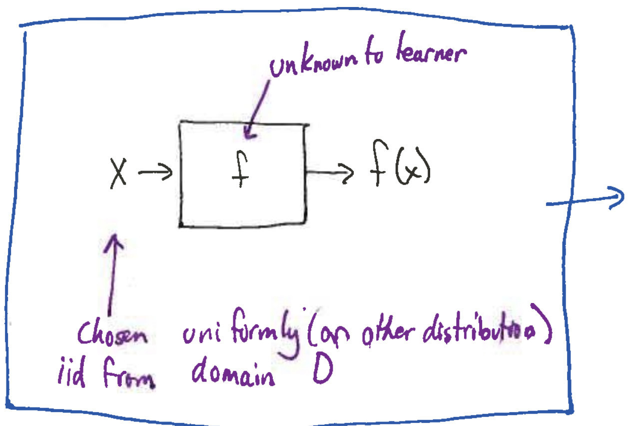
Learning via Fourier Coeffs

- Some fctns & their Fourier representation
- the low degree algorithm
- applications

Learning

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



Example oracle $Ex(f)$

$x_1, f(x_1)$
$x_2, f(x_2)$
$\vdots$

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 D?

  Today: uniform

  In general: match distribution of examples

Valiant's PAC model

"Probably Approximately Correct"

<u>def.</u> 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 orac

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

<u>Note</u> if f is arbitrary,
there is <u>nothing</u> <u>you</u> <u>can</u> <u>do</u>! (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

<u>def.</u> uniform distribution learning algorithm for concept class, $\mathcal{C}$ is algorithm $\mathcal{A}$ s.t.

- $\mathcal{A}$ is given $\varepsilon, \delta > 0$ access to $Ex(f)$ for $f \in \mathcal{C}$
- $\mathcal{A}$ outputs h s.t. with prob $\geq 1-\delta$ $\underbrace{\text{error}(h) \text{ w.r.t. } f \text{ is } \leq \varepsilon}_{h \text{ is } \varepsilon\text{-close to } f}$

Parameters of Interest

- m # samples used by $\mathcal{A}$ "sample complexity"
- ε accuracy parameter
- δ confidence parameter
- runtime?   hope for 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 $\text{error}(h)$ wrt $f \geq \varepsilon$

$\Pr[\text{bad } h \text{ consistent with examples}]$

$$\leq (1-\varepsilon)^M$$

$\Pr[\underline{\text{any}} \text{ bad } h \text{ consistent with examples}]$

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

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

∴ 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 random inputs since $\Pr_{x \in \mathcal{D}}[f(x)=h(x)] \geq 1-\delta$
     according to $\mathcal{D}$

  2) can compress description of samples

  $$(x_1, f(x_1))\,(x_2, f(x_2))\,\ldots\,(x_m, f(x_m)) \qquad m(\log|D| + \log|R|)$$

  range of f

  $$\Downarrow$$

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

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

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

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

- 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:

$Pr[i \in V] \leq Pr[i$ set same in each of K positive examples$]$

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

$Pr[$any i that not in conjunction manages to survive$]$

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

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

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

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