PAC Learning Model: Definition

Given a probability distribution P on X, a concept C and a hypothesis H, define the *error* of H: $err(H) = P(C \triangle H) = P(c(x) \neq h(x))$

A formality: define also $\text{err}(h) = \text{err}(H)$ (h being the description of H)

We say that an algorithm *PAC-learns concept class* $\mathcal{C}$ if for any $C \in \mathcal{C}$, an arbitrary distribution P on X, and arbitrary numbers $0 < \epsilon, \delta < 1$, the algorithm, which receives a $\text{poly}(1/\epsilon, 1/\delta, n)$ number of i.i.d. examples from $P(X)$, outputs with probability at least $1 - \delta$ a hypothesis h such that $\text{err}(h) \leq \epsilon$. If such an algorithm exists, we call $\mathcal{C}$ *PAC-Learnable*.

If an algorithm PAC-learns $\mathcal{C}$ and runs in $\text{poly}(1/\epsilon, 1/\delta, n)$ time, we say it PAC-learns $\mathcal{C}$ *efficiently* and we call $\mathcal{C}$ *efficiently PAC-learnable*.

PAC Learning Conjunctions

Use the generalization algo (previous lecture) for PAC learning: provide m examples to it, run it as if online, keep the last h.

Let $P_{\text{ic}}(z)$ be the prob. that literal z ($z \in \{ h_1, \overline{h_1}, h_2, \ldots \overline{h_n} \}$) is inconsistent with a random example drawn from $P(X)$.

$\text{err}(h) = P(\text{at least one literal in } h \text{ inconsistent}) \leq \sum_z P_{\text{ic}}(z)$

Call z *bad* if $P_{\text{ic}}(z) \geq \frac{\epsilon}{2n}$. So if h has no bad literals then

$$\text{err}(h) \leq \sum_z \frac{\epsilon}{2n} = 2n\frac{\epsilon}{2n} = \epsilon$$

PAC Learning Conjunctions

Prob. that a bad literal z "survived" (was consistent with) one random example is

$$1 - P_{\text{ic}}(z) \leq 1 - \frac{\epsilon}{2n}$$

Prob. that z survived *m such i.i.d. examples* is thus at most

$$\left(1 - \frac{\epsilon}{2n}\right)^m$$

So prob. that *one of the $2n$* possible bad literals survived m i.i.d. examples is at most

$$2n\left(1 - \frac{\epsilon}{2n}\right)^m \leq 2ne^{-\frac{m\epsilon}{2n}}$$

because of the general inequality $1 - x \leq e^{-x}$ for $x \geq 0$.

PAC Learning Conjunctions

To satisfy PAC-learning conditions, we need

$$2ne^{-\frac{m\epsilon}{2n}} < \delta$$

after arrangements:

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

Thus $m \leq \text{poly}(1/\epsilon, 1/\delta, n)$ examples suffice to make $\text{err}(h) \leq \epsilon$ with probability at least $1-\delta$.

So the generalization algorithm PAC-learns conjunctions (efficiently - same argument as in the mistake-bound framework).

Mistake-Bound Learnability Implies PAC-Learnability

Any mistake-bound learner L can be transformed into a PAC-learner. Let $M \leq \text{poly}(n)$ be the mistake bound of L.

Call L *lazy* if it changes its hypo h *only* following a mistake. If L is not lazy, make it lazy (prevent changing h after correct decisions).

Run L on the example set but halt if any hypo h survives more than $\frac{1}{\epsilon} \ln \frac{M}{\delta}$ *consecutive* examples. Output h.

Observe that this *will* terminate within $m = \frac{M}{\epsilon} \ln \frac{M}{\delta}$ examples. (Otherwise more than M mistakes would be made.)

Mistake-Bound Learnability Implies PAC-Learnability

Prob. that $\text{err}(h) > \epsilon$ is at most

$$M(1-\epsilon)^{\frac{1}{\epsilon}\ln\frac{M}{\delta}} < Me^{-\frac{\epsilon}{\epsilon}\ln\frac{M}{\delta}} = M\frac{\delta}{M} = \delta$$

Since $M \leq \text{poly}(n)$ (condition of MB learning), also

$$m = \frac{M}{\epsilon}\ln\frac{M}{\delta} \leq \text{poly}(1/\epsilon, 1/\delta, n)$$

So all PAC-learning conditions satisfied: we have $m \leq \text{poly}(1/\epsilon, 1/\delta, n)$, and $\text{err}(h) \leq \epsilon$ with prob. at least $1-\delta$.

PAC-Learning Implies Consistency

Although $\text{err}(h) > 0$ is allowed, the output h of a PAC-learner is necessarily consistent with all the training examples (zero "training error").

Assume that given training set $\{ x_1, x_2, \ldots x_m \}$, the algo outputs h *inconsistent* with some x_j ($1 \leq j \leq m$).

Distribution $P(x)$ and numbers ϵ, δ are arbitrary so set them such that

- $\prod_{i=1}^{m} P(x_i) > \delta$ (implying that $P(x_j) > 0$);
- $\epsilon < P(x_j)$ (can be done because $P(x_j) > 0$)

So with prob. $> \delta$ the algo will output h such that $\text{err}(h) \geq P(x_j) > \epsilon$, i.e. it *does not* PAC-learn.

Consistency + Polynomial $\ln |\mathcal{H}|$ Imply PAC-Learning

An algorithm using hypothesis class $\mathcal{H}$ is *$\mathcal{C}$-consistent* if, given an arbitrary example set from an arbitrary concept $C \in \mathcal{C}$, it returns a $h \in \mathcal{H}$ consistent with the example set.

$\mathcal{H} \supseteq \mathcal{C}$ is a necessary condition for $\mathcal{C}$-consistency.

A $\mathcal{C}$-consistent algorithm using $\mathcal{H}$ PAC-learns $\mathcal{C}$ if $\ln |\mathcal{H}| \leq \text{poly}(n)$. Why?

Prob. that a given *bad* h ($\text{err}(h) > \epsilon$) survives (i.e., is consistent with) a random example is at most $(1 - \epsilon)$.

Consistency + Polynomial ln $|\mathcal{H}|$ Imply PAC-Learning

Prob. that h survives m i.i.d. examples is at most $(1-\epsilon)^m$.

Prob. that one of the bad hypotheses $h \in \mathcal{H}$ survives is at most $|\mathcal{H}|(1-\epsilon)^m \leq |\mathcal{H}|e^{-\epsilon m}$.

To make this smaller than δ, it suffices to set the number of examples to

$$m = \frac{1}{\epsilon} \ln \frac{|\mathcal{H}|}{\delta}$$

which is $\leq \text{poly}(1/\epsilon, 1/\delta, n)$ iff $\ln |\mathcal{H}| \leq \text{poly}(n)$.

Compare this to the similar result in the mistake-bound model (Halving algorithm).

Consistency + Polynomial VC($\mathcal{H}$) Imply PAC-Learning

Using VC($\mathcal{H}$), a bound can be established even for $|\mathcal{H}| = \infty$:

With probability at least δ, no bad hypothesis $h \in \mathcal{H}$ survives m i.i.d. examples where

$$m \geq \frac{8}{\epsilon}\left(\mathrm{VC}(\mathcal{H}) \ln \frac{16}{\epsilon} + \ln \frac{2}{\delta}\right)$$

(We omit the proof.)

Thus a $\mathcal{C}$-consistent algorithm using $\mathcal{H}$ PAC-learns $\mathcal{C}$ if $\mathrm{VC}(\mathcal{H}) \leq \mathrm{poly}(n)$.

For example, let $\mathcal{C}$ = half-planes in R^n. $|\mathcal{H}| = \infty$ but $\mathrm{VC}(\mathcal{H}) = n + 1 \leq \mathrm{poly}(n)$.

k-Decision Trees

(Binary) decision tree: a binary tree-graph

- non-leaf vertices: binary variables
- leafs: class indicators

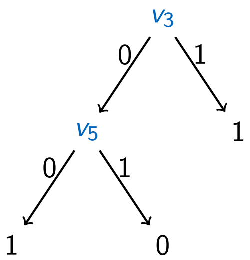
Classification: go from root to leaf, path according to truth-values of variables.

k-DT = dec. trees of max depth k

Like k-term DNF,

- finding a consistent k-DT is NP-hard (proof omitted).
- *k-DT thus cannot be PAC-learned efficiently + properly.*

Example:



3-Decision Tree

PAC-Learning k-Decision Trees Efficiently

Every k-DT has an equivalent k-DNF:

- For every path going from root to a 1 leaf, add to the DNF a k-conjunction of all variables on the path ($v_3 \vee \overline{v_3}\ \overline{v_5}$ for the example)

Thus

$$k\text{-DT} \subseteq k\text{-DNF}$$

and $\mathcal{C} = k$-DT can be *efficiently (but not properly) PAC-learned* using $\mathcal{H} = k$-DNF.

Note that also

$$k\text{-DT} \subseteq k\text{-CNF}$$

- Create a clause for each path to a 0 leaf ($v_3 \vee \overline{v_5}$ for the example)

PAC-Learning k-Decision Trees Properly

We will show that $\lg |k\text{-DT}| \leq \text{poly}(n)$. Denote $c_k = |k\text{-DT}|$.

- $c_1 = 2$ (two options for the single vertex = leaf) so

$$\lg c_1 = 1 \tag{1}$$

- $c_{k+1} = nc_k^2$ (n options for vertex, c_k options for each of the 2 subtrees)

$$\lg c_{k+1} = \lg n + 2 \lg c_k \tag{2}$$

(1) and (2) are a recursive formula for a geometric series in variable $\lg c_k = \lg |k\text{-DT}|$. Solution exponential in k but polynomial in n.

So $\mathcal{C} = k$-DT can be *properly (but not efficiently) PAC-learned* by a $\mathcal{C}$-consistent algorithm.

Inconsistent Learning

Returning a hypothesis consistent with the training set may not be possible for reasons such as

- $\mathcal{H} \not\supseteq \mathcal{C}$;
- $\mathcal{C}$ is not known ('agnostic learning') so $\mathcal{H} \not\supseteq \mathcal{C}$ cannot be excluded;
- There is 'noise' in data so the training set may include the same instance as both a positive and a negative example.

Define the *training error* $\widehat{\text{err}}(h)$ as the proportion of training examples inconsistent with h. $\widehat{\text{err}}(h)$ is also called the *empirical risk*.

We are interested in the relationship btw. $err(h)$ and $\widehat{\text{err}}(h)$.

Hoeffding Inequality

Hoeffding: Let $\{z_1, z_2, \ldots, z_m\}$ be a set of i.i.d. samples from $P(z)$ on $\{0, 1\}$. The probability that $\left|P(1) - \frac{1}{m}\sum_{i=1}^m z_i\right| > \epsilon$ is at most $2e^{-2\epsilon^2 m}$.

Let $z_i = 1$ iff i.i.d. example x_i is misclassified by h. So

$$P(1) = \mathrm{err}(h)$$
$$\frac{1}{m}\sum_{i=1}^m z_i = \widehat{\mathrm{err}}(h)$$

Thus for a given h, $|\mathrm{err}(h) - \widehat{\mathrm{err}}(h)| > \epsilon$ with prob. at most $2e^{-2\epsilon^2 m}$.

Error Bound for Inconsistent Learning

For a finite $\mathcal{H}$, the prob. that $|\text{err}(h) - \widehat{\text{err}}(h)| > \epsilon$ for *some* $h \in \mathcal{H}$ is at most

$$|\mathcal{H}| 2e^{-2\epsilon^2 m}$$

We want to make this no greater than δ. Solving $\delta = |\mathcal{H}| 2e^{-2\epsilon^2 m}$ gives

$$\epsilon = \sqrt{\frac{1}{m} \ln \frac{2|\mathcal{H}|}{\delta}}$$

So with prob. at least $1 - \delta$, the difference btw. $\text{err}(h)$ and $\widehat{\text{err}}(h)$ is at most as above for all $h \in \mathcal{H}$.

Dilemma: A large $\mathcal{H}$ allows to achieve a small $\widehat{\text{err}}(h)$ but means a loose bound on $\text{err}(h)$.

Sample Complexity for Inconsistent Learning

Solving $\delta = |\mathcal{H}| 2e^{-2\epsilon^2 m}$ instead for m gives

$$m = \frac{1}{2\epsilon^2} \ln \frac{2|\mathcal{H}|}{\delta}$$

which is thus a number of examples sufficient to make $|\text{err}(h) - \widehat{\text{err}}(h)| \leq \epsilon$ with prob. at least $1 - \delta$ for all $h \in \mathcal{H}$.

$m \leq \text{poly}(1/\epsilon, 1/\delta, n)$ iff $\ln |\mathcal{H}| \leq \text{poly}(n)$

Error Bound for ERM

Assume the learner returns

$$h = \arg\min_{h \in \mathcal{H}} \widehat{\text{err}}(h)$$

This is called *empirical risk minimization* (ERM principle).

Let $h^* = \arg\min_{h \in \mathcal{H}} \text{err}(h)$, i.e. h^* is the best hypothesis.

Let further $m = \frac{1}{2\epsilon^2} \ln \frac{2|\mathcal{H}|}{\delta}$. Then with prob. at least $1 - \delta$:

$$\begin{aligned}
\forall h \in \mathcal{H} : \text{err}(h) &\leq \widehat{\text{err}}(h) + \epsilon && \text{which we just proved} \\
&\leq \widehat{\text{err}}(h^*) + \epsilon && \text{because } h \text{ minimizes } \widehat{\text{err}} \\
&\leq \text{err}(h^*) + 2\epsilon && \text{because } \widehat{\text{err}}(h^*) \leq \text{err}(h^*) + \epsilon
\end{aligned}$$

Bias-Variance Trade-Off

Put differently, with prob. at least $1 - \delta$:

$$\text{err}(h) \leq \min_{h \in \mathcal{H}} \text{err}(h) + 2\sqrt{\frac{1}{2m} \ln \frac{2|\mathcal{H}|}{\delta}}$$

Large $\mathcal{H}$ - large variance - small bias - first summand lower, second larger

Too large $\mathcal{H}$: overfitting, too small $\mathcal{H}$: underfitting

The more training data (m), the larger $\mathcal{H}$ can be 'afforded'.