Proper PAC-Learning

We have shown s-term DNF (s-clause CNF, respectively) to be learnable online from $X = \{0, 1\}^n$ because it is a subset of the class s-CNF (s-DNF) which is learnable (by a standard agent) from X.

So by Theorem (??), the two classes are also PAC-Learnable from X, which means that the agent finds a hypothesis h_K such that $\text{err}(h_K) < \epsilon$ with probability at least $1 - \delta$ where $K \leq poly(\frac{1}{\epsilon}, \frac{1}{\delta}, n_X)$.

h_K is a s-CNF (s-DNF) and generally, *it cannot be rewritten into an equivalent s-term DNF (s-clause CNF)*.

We will define a stronger version of PAC-learning which requires that h_K belongs to the hypothesis class from which the target hypothesis is chosen.

Proper PAC-Learning

Proper PAC Learning Model

Let $\mathcal{H}$ be a hypothesis class. An agent (efficiently) **properly PAC-learns** $\mathcal{H}$ from an observation class X if all conditions for (efficient) PAC-learning of $\mathcal{C}(\mathcal{H})$ are satisfied, and, in addition, for the h_K in the definition it holds $h_K \in \mathcal{H}$. A hypothesis class $\mathcal{H}$ is (efficiently) **properly PAC-learnable** from X if there is an agent (efficiently) properly PAC-learning $\mathcal{H}$ from X.

Proper PAC-learning is important e.g. when h_K is to be interpreted by a human and its membership in $\mathcal{H}$ guarantees readability.

Efficiently Properly PAC-Learnable Classes

Given Theorem (??), a hypothesis class $\mathcal{H}$ is efficiently properly PAC-learnable from X if there is a standard agent that efficiently learns $\mathcal{H}$ online from X and the hypotheses h_k the agent uses as decision policies are all from $\mathcal{H}$.

For example, *conjunctions* (*clauses*, respectively) are efficiently properly PAC-learnable from $X = \{0, 1\}^n$ or from $X =$ contingent conjunctions (clauses) because they are learnable online efficiently with the generalization algorithm, and all h_k are conjunctions (clauses). *(Unlike Winnow, where h_k are hyperplanes!)*

(Non)-Learnability of s-term DNF and s-clause CNF

We already know that s-term DNF is efficiently learnable online from $X = \{0, 1\}^n$ by a standard agent thus also *efficiently* PAC-learnable from X. It is also *properly* PAC-learnable from X due to Theorem (??) and the fact that $\lg |s\text{-CNF}| \leq poly(n)$ and $\mathcal{C}(s\text{-term DNF}) \subseteq \mathcal{C}(s\text{-CNF})$.

The same holds analogically for s-clause CNF. Are these classes also *efficienty* *properly* learnable?

Theorem 1

None of s-term DNF and s-clause CNF is efficiently properly PAC-learnable from $X = \{0, 1\}^n$

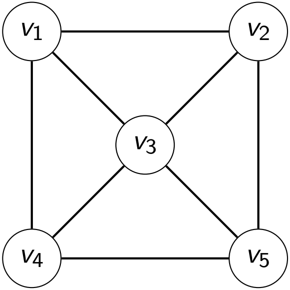
Proof: We will show the proof only for the special case of 3-term DNF. The NP-complete graph 3-coloring problem can be reduced in poly-time to finding an 3-term DNF consistent with a finite set of observations.

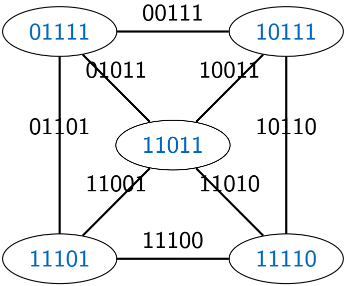
FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

3-term DNF's are not efficiently properly PAC-learnable.

vertex v_i $\leftrightarrow$ pos. example x, $x^l = \begin{cases} 0 \text{ if } l = i \\ 1 \text{ otherwise} \end{cases}$

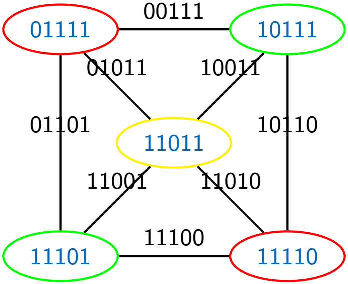
edge e_{ij} $\leftrightarrow$ neg. example x, $x^l = \begin{cases} 0 \text{ if } l = i \text{ or } l = j \\ 1 \text{ otherwise} \end{cases}$





3-term DNF's are not efficiently properly PAC-learnable.

Graph 3-colorable iff a 3-term DNF consistent with the observations.



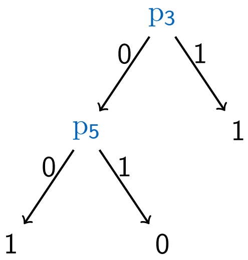
$$\bigvee_{\textbf{color} \in \{R, G, Y\}} \; \bigwedge_{v_i \text{ not of } \textbf{color}} \mathrm{p}_i \;\Rightarrow\; (\mathrm{p}_2 \wedge \mathrm{p}_3 \wedge \mathrm{p}_4) \vee (\mathrm{p}_1 \wedge \mathrm{p}_3 \wedge \mathrm{p}_5) \vee (\mathrm{p}_1 \wedge \mathrm{p}_2 \wedge \mathrm{p}_4 \wedge \mathrm{p}_5)$$

$\Leftarrow$ color of any consistent term

3-colorability NP-hard $\rightarrow$ finding a consistent 3-term DNF NP-hard.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Example:



3-Decision Tree

A **decision tree** on $X = \{0, 1\}^n$ is a binary tree graph where each non-leaf vertex indicates one of the n components, each leaf is a class from Y, and each edge is labeled 0 or 1.
It prescribes a policy for $x \in X$: go from the root, always following one of the two outgoing edges that is labeled with the value of the component in the last vertex, until in a leaf. The leaf is the decision.
For example, $x = (0, 1, 0, 1, 1)$ is decided as $y = 0$ by the tree on the left.
An ***s*-decision tree** has *depth* s or less.

PAC-Learnability of s-Decision Trees

Theorem 2

The class s-Decision trees is PAC-learnable from $X = \{0, 1\}^n$ efficiently or properly but not efficiently properly.

Proof: For any s-DT, there is an equivalent s-DNF: create an s-conjunction for each tree's path from the root to a "1" leaf. E.g. this tree corresponds to the 3-DNF $\mathrm{p}_3 \vee (\neg \mathrm{p}_3 \wedge \neg \mathrm{p}_5)$. So

$$\mathcal{C}(s\text{-DT}) \subseteq \mathcal{C}(s\text{-DNF}) \tag{1}$$

s-DNF is efficently learnable online by a standard agent and thus also efficiently PAC-learnable. So the agent can efficiently PAC-learn s-DT using s-DNF. Thus s-DT is *efficiently* PAC-learnable.

PAC-Learnability of s-Decision Trees (cont'd)

s-DT is also *properly* PAC-learnable by a s-DT-consistent agent according to Theorem (??) due to $\lg|s\text{-DT}| \leq poly(n_X)$ where $n_X = n$. Indeed, $|1\text{-DT}| = 2$ because there are exactly two options $\{0, 1\}$ for the single vertex (leaf) of it. So

$$\lg|1\text{-DT}| = \lg 2 = 1 \tag{2}$$

For $s > 1$, $|(s+1)\text{-DT}| = n|s\text{-DT}|^2$ (n options for the vertex and $|s\text{-DT}|$ options for each of the two subtrees). Take the logarithm of the equation:

$$\lg|(s+1)\text{-DT}| = \lg n + 2\lg|s\text{-DT}| \tag{3}$$

(2) and (3) form a recursive prescription of a geometric series whose solution is $\lg|s\text{-DT}| = (2^s - 1)(1 + \lg n) + 1 \leq poly(n)$.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

PAC-Learnability of s-Decision Trees (cont'd)

Finally, finding an s-tree consistent with a finite set of observations is an NP-complete problem. We omit the part of the proof showing this but refer to the analogical proof for s-term DNF following Theorem (1).

Thus the class s-DT is not efficiently properly PAC-Learnable, which completes the proof.

Note: similarly to (1), we also have

$$\underline{\mathcal{C}(s\text{-DT})} \subseteq \underline{\mathcal{C}(s\text{-CNF})} \tag{4}$$

Given an s-DT, one creates a clause for each path from root to a "0" leaf, e.g. this tree corresponds to the single-clause 3-CNF $\mathrm{p}_3 \vee \neg \mathrm{p}_5$.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

s-Decision Lists

Example:

c	y
$p_1 \wedge \neg p_3$	0
p_2	1
$\neg p_1$	1
$\emptyset$	0

2-Decision list

An ***s*-Decision list** on $X = \{0,1\}^n$ is a list of pairs (c, y) where c is an s-conjunction using variables from $p_1, p_2, \ldots, p_n$ and $y \in Y$.
The last conjunction in the list is empty and the corresponding y is called the *default class*.
It classifies an $x \in X$ into class y_i where (c_i, y_i) is the first pair in the list such that $x \models c_i$.
For example, $x = (1,1,1)$ is classified into 1 by the decision list on the left.

PAC-Learnability of s-Decision Lists

Theorem 3

The class s-Decision lists is efficiently properly PAC-learnable from $X = \{0, 1\}^n$.

We will present an s-DL-consistent algorithm known as the *covering algorithm* for efficient finding of an s-DL hypothesis h_{k+1} consistent with $x_{\leq k}$.

Let $T_{k+1} = \{ (x_1, \overline{y}_1), (x_2, \overline{y}_2), \ldots, (x_k, \overline{y}_k) \}$ where $\overline{y}_i$ $(1 \leq i \leq k)$ is the true class of x_i. T_{k+1} is called a **training set** (at time $k+1$).

Note that the agent knows all elements of T_{k+1} because it has seen all of the x_i and the $\overline{y}_i$ can be determined as $\overline{y}_i = |y_i + r_{i+1}|$.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Finding a Consistent s-Decision List

Require: training set T

$L := []$ (empty list)
while $T \neq \emptyset$ **do**
    $c =$ any s-conjunction true for some positive and no negative example in T, *or* some negative and no positive example in T (*respectively*)
    Remove samples covered by c: $T := T \setminus \{ (x, \overline{y}) \in T : x \mid = c \}$
    if $T = \emptyset$ **then**
        append $(\emptyset, 1)$ or $(\emptyset, 0)$ (*respectively*) to L.
    else
        append $(c, 1)$ or $(c, 0)$ (*respectively*) to L
    end if
end while

PAC-Learnability of s-Decision Lists (cont'd)

$$|s\text{-DL}| = 3^{|s\text{-conjunctions}|}!$$

because each s-conjunction can be absent from the list, present with $y = 0$ or present with $y = 1$ (hence the base 3), and they can be arranged in an arbitrary order (hence the factorial).

We know that $|s\text{-conjunctions}| \leq poly(n)$. So we have

$$\lg |s\text{-DL}| < poly(n)$$

So by Theorem (??), the s-DL-consistent covering algorithm PAC-learns s-DL. Since it is efficient and the output is an s-DL, it does so efficiently and properly, which finishes the proof.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

s-Decision Lists (cont'd)

Every s-DNF has an equivalent s-DL constructed as follows

- for each s-conjunction c from the s-DNF, add $(c, 1)$ to the s-DL
- add $(\emptyset, 0)$ to the s-DL

so

$$\mathcal{C}(s\text{-DNF}) \subset \mathcal{C}(s\text{-DL})$$

s-DL is closed under negation, i.e., for any $h \in s$-DL, also $\neg h \in s$-DL (just flip the zeros and ones for all the y_i in h). Each s-CNF is the negation of some s-DNF. Therefore also

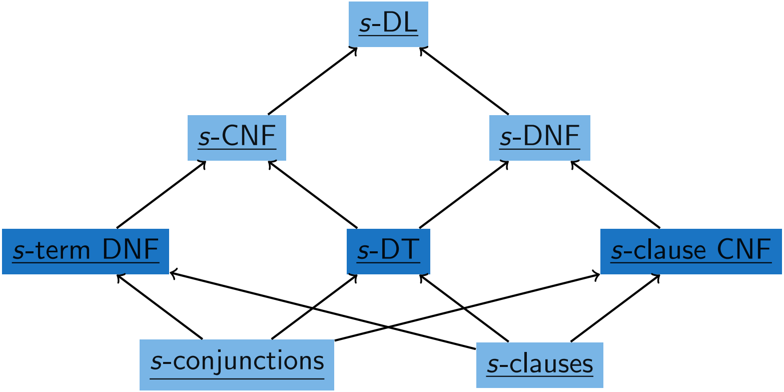
$$\mathcal{C}(s\text{-CNF}) \subset \mathcal{C}(s\text{-DL})$$

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE

Hierarchy of Size-Bounded Propositional Classes

efficiently properly PAC-learnable

efficiently **or** properly PAC-learnable



Inconsistent Learning

Consistent learning may not be possible when (??) does not hold or when rewards r_{k+1} are not deterministic as in (??) but depend only *probabilistically* on x_k and y_{k+1}. *The latter case corresponds to learning from "noisy data."*

Define the **training error** $\widehat{\text{err}}(h_{k+1})$ ($k \in \mathbb{N}$) of hypothesis h_{k+1} as

$$\widehat{\text{err}}(h_{k+1}) = \frac{1}{k} \sum_{i=1}^{k} |h_{k+1}(x_i) - \overline{y}_i| \tag{5}$$

where $\overline{y}_i$ is the true class of x_i. So $\widehat{\text{err}}(h_{k+1})$ is the proportion of observations from $x_{\leq k}$ that h_{k+1} is not consistent with.

Note that $\widehat{\text{err}}(h_{k+1})$ is in general not equal to $\frac{1}{k} \sum_{i=1}^{k} |r_i|$ since actions y_i, $1 \leq i \leq k$ were decided by hypotheses other than h_{k+1}.

Inconsistent Learning (cont'd)

The following lemma a direct consequence of the well-known Hoeffding inequality.

Lemma 1

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

Theorem 4

Let $h_{k+1} \in \mathcal{H}$ $(\forall k \in \mathbb{N})$ *where* $\mathcal{H}$ *is a hypothesis class. With probability at least* $1 - \delta$

$$|\mathrm{err}(h_{k+1}) - \widehat{\mathrm{err}}(h_{k+1})| \leq \sqrt{\frac{1}{2k} \ln \frac{2|\mathcal{H}|}{\delta}} \tag{6}$$

Inconsistent Learning (cont'd)

Proof of Theorem (6): by assumption, $x_1, x_2, \ldots, x_k$, are i.i.d. from (??), thus for a given $h_{k+1} \in \mathcal{H}$,

$$|h_{k+1}(x_1) - \overline{y}_1|, |h_{k+1}(x_1) - \overline{y}_2|, \ldots |h_{k+1}(x_k) - \overline{y}_k|$$

where $\overline{y}_i$ are the true classes of x_1 is an i.i.d. sample from a $P(.)$ on $\{0, 1\}$ where $P(1) = \mathrm{err}(h_{k+1})$. Thus given (5) and Lemma (1), the probability that

$$|\mathrm{err}(h_{k+1}) - \widehat{\mathrm{err}}(h_{k+1})| > \epsilon$$

is at most $2e^{-2\epsilon^2 k}$. The probability that the above is true for *some* $h_{k+1} \in \mathcal{H}$ is thus at most $|\mathcal{H}|2e^{-2\epsilon^2 k}$. Setting $|\mathcal{H}|2e^{-2\epsilon^2 k} = \delta$ yields

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

which completes the proof.

FACULTY OF ELECTRICAL ENGINEERING CTU IN PRAGUE