Machine Learning (CSE 446): Learning Theory

Noah Smith

University of Washington
nasmith@cs.washington.edu

November 27, 2017

Big Questions in Learning Theory

- When is learning possible?
- How much data is required?
- Will a learned classifier generalize to test data?

Big Questions in Learning Theory

- When is learning possible?
- How much data is required?
- Will a learned classifier generalize to test data?

Theory can come before or after practice.

The Ultimate Learning Algorithm?

Simple $\mathcal{D}$ that is inherently noisy: X and Y both binary. Let $p(X = Y) = 0.8$.

The Ultimate Learning Algorithm?

Simple $\mathcal{D}$ that is inherently noisy: X and Y both binary. Let $p(X = Y) = 0.8$.

There's simply no way to get better than 80% accuracy with any classifier f.

The Ultimate Learning Algorithm?

Simple $\mathcal{D}$ that is inherently noisy: X and Y both binary. Let $p(X = Y) = 0.8$.

There's simply no way to get better than 80% accuracy with any classifier f.

Even if your data aren't noisy and low error is achievable by some f, you still have to worry about lousy samples from $\mathcal{D}$.

The Ultimate Learning Algorithm?

Simple $\mathcal{D}$ that is inherently noisy: X and Y both binary. Let $p(X = Y) = 0.8$.

There's simply no way to get better than 80% accuracy with any classifier f.

Even if your data aren't noisy and low error is achievable by some f, you still have to worry about lousy samples from $\mathcal{D}$.

You can't hope for perfection every time, or even "pretty good" every time, or perfection most of the time. The best you can hope for is pretty good, most of the time.

Probably Approximately Correct

- Probably: on most test sets (i.e., succeed on $(1 - \delta)$ of the possible test sets)
- Approximately Correct: low error (i.e., accuracy at least $(1 - \epsilon)$)

Definition: An (ϵ, δ)-**PAC learning algorithm** is defined as one that, given samples from any data distribution $\mathcal{D}$, returns a "bad function" with probability $\leq \delta$, where a bad function is one whose test error rate is greater than ϵ on $\mathcal{D}$.

Efficiency

Definition: An (ϵ, δ)-PAC learning algorithm is **efficient** if its runtime is polynomial in $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.

Efficiency

Definition: An (ϵ, δ)-PAC learning algorithm is **efficient** if its runtime is polynomial in $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.

E.g., if you want to reduce error rate from 5% to 4%, you shouldn't require an exponential increase in computational resources.

Efficiency

Definition: An (ϵ, δ)-PAC learning algorithm is **efficient** if its runtime is polynomial in $\frac{1}{\epsilon}$ and $\frac{1}{\delta}$.

E.g., if you want to reduce error rate from 5% to 4%, you shouldn't require an exponential increase in computational resources.

Note that this extends to the *size of the training set*: if your training dataset must increase exponentially, that will also affect runtime!

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also https://www.autonlab.org/_media/tutorials/pac05.pdf

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

Example: $X_1 \wedge X_7 \wedge \neg X_9$.

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

How many hypotheses are there, $|\mathcal{H}|$?

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

How many hypotheses are there, $|\mathcal{H}|$? 3^d

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

How many hypotheses are there, $|\mathcal{H}|$? 3^d

Assume: Y is given by some $h^* \in \mathcal{H}$. That is, for a given $\mathbf{x}$, $y = f_{h^*}(\mathbf{x})$, without noise.

Example: "And-Literals" Machine

Thanks to Andrew Moore; see also `https://www.autonlab.org/_media/tutorials/pac05.pdf`

Let $\boldsymbol{X}$ range over binary vectors (unknown distribution), denoted $\langle X_1, \ldots, X_d \rangle$.

Let $\mathcal{H}$, the set of hypotheses, contain all logical conjunctions of $\langle X_1, \ldots, X_d \rangle$ and their negations.

How many hypotheses are there, $|\mathcal{H}|$? 3^d

Assume: Y is given by some $h^* \in \mathcal{H}$. That is, for a given $\mathbf{x}$, $y = f_{h^*}(\mathbf{x})$, without noise.

Learning: choose $h \in \mathcal{H}$ given a training dataset drawn from distribution $\mathcal{D}$.

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N \leq e^{-\epsilon \cdot N}$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1 - \epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

In other words, this unfortunate event is bounded by the probability of avoiding one of the $\epsilon \times$ 100% cases of h's error, N times.

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$
\begin{aligned}
&= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N \\
&\leq e^{-\epsilon \cdot N}
\end{aligned}
$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$
\begin{aligned}
&= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1 - \epsilon)^N \\
&\leq e^{-\epsilon \cdot N}
\end{aligned}
$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$
\begin{aligned}
&= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N \\
&\leq e^{-\epsilon \cdot N}
\end{aligned}
$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}})$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1 - \epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}})$$
$$= p\left(\bigvee_{h \in \mathcal{H}} h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}\right)$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}})$$
$$= p\left(\bigvee_{h \in \mathcal{H}} h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}\right)$$
$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}) \quad \text{"union bound"}$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \ldots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}})$$
$$= p\left(\bigvee_{h \in \mathcal{H}} h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}\right)$$
$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}) \quad \text{"union bound"}$$

Note that
$p(P \wedge Q) = p(P \mid Q) \cdot \underbrace{p(Q)}_{\leq 1} \leq p(P \mid Q)$.

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \dots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}})$$
$$= p\left(\bigvee_{h \in \mathcal{H}} h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}\right)$$
$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \wedge h \in \mathcal{H}_{\text{bad}}) \quad \text{"union bound"}$$
$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$$

The Game

- We choose the "machine" (e.g., the and-literal machine), or the class of functions $\mathcal{F} = \{f_h : h \in \mathcal{H}\}$.
- Nature chooses $h^* \in \mathcal{H}$ and randomly samples N inputs from $\mathcal{D}$ (which is fixed and unknown), then labels them using $y_n = f_{h^*}(\mathbf{x}_n)$.
- Let $\mathcal{H}_0$ contain all $h \in \mathcal{H}$ that achieve zero training set error. We choose some $h^{\text{est}} \in \mathcal{H}_0$.
- Let $\mathcal{H}_{\text{bad}}$ contain all $h \in \mathcal{H}$ such that the test set error of f_h is greater than ϵ.

First consider $p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$

$$= p(\forall n \in \{1, \dots, N\}, f_h(\mathbf{x}_n) = y_n \mid h \in \mathcal{H}_{\text{bad}}) \leq (1-\epsilon)^N$$
$$\leq e^{-\epsilon \cdot N}$$

Now consider $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}})$

$$\leq p(\exists h : h \in \mathcal{H}_0 \land h \in \mathcal{H}_{\text{bad}})$$

$$= p\left(\bigvee_{h \in \mathcal{H}} h \in \mathcal{H}_0 \land h \in \mathcal{H}_{\text{bad}}\right)$$

$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \land h \in \mathcal{H}_{\text{bad}}) \quad \text{“union bound”}$$

$$\leq \sum_{h \in \mathcal{H}} p(h \in \mathcal{H}_0 \mid h \in \mathcal{H}_{\text{bad}})$$

$$\leq |\mathcal{H}| \cdot (1-\epsilon)^N \leq |\mathcal{H}| \cdot e^{-\epsilon \cdot N}$$

Blumer Bound

We want to bound $p(h^{\mathsf{est}} \in \mathcal{H}_{\mathsf{bad}}) \leq \delta$:

$$|\mathcal{H}| \cdot e^{-\epsilon \cdot N} \leq \delta$$

$$\Rightarrow \quad N \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right) \approx \frac{0.69}{\epsilon}\left(\log_2|\mathcal{H}| + \log_2\frac{1}{\delta}\right)$$

Blumer Bound

We want to bound $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}}) \leq \delta$:

$$|\mathcal{H}| \cdot e^{-\epsilon \cdot N} \leq \delta$$

$$\Rightarrow \quad N \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right) \approx \frac{0.69}{\epsilon}\left(\log_2|\mathcal{H}| + \log_2\frac{1}{\delta}\right)$$

For our and-literals machine, $|\mathcal{H}| = 3^d$, so we need $\frac{1}{\epsilon}\left(1.1d + \ln\frac{1}{\delta}\right)$ training examples to "PAC-learn."

Blumer Bound

We want to bound $p(h^{\mathsf{est}} \in \mathcal{H}_{\mathsf{bad}}) \leq \delta$:

$$|\mathcal{H}| \cdot e^{-\epsilon \cdot N} \leq \delta$$

$$\Rightarrow \quad N \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right) \approx \frac{0.69}{\epsilon}\left(\log_2|\mathcal{H}| + \log_2\frac{1}{\delta}\right)$$

Corollary: if $h^{\mathsf{est}} \in \mathcal{H}_0$, then you can estimate ϵ as

$$\frac{1}{N}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right)$$

Blumer Bound

We want to bound $p(h^{\text{est}} \in \mathcal{H}_{\text{bad}}) \leq \delta$:

$$|\mathcal{H}| \cdot e^{-\epsilon \cdot N} \leq \delta$$

$$\Rightarrow \quad N \geq \frac{1}{\epsilon}\left(\ln|\mathcal{H}| + \ln\frac{1}{\delta}\right) \approx \frac{0.69}{\epsilon}\left(\log_2|\mathcal{H}| + \log_2\frac{1}{\delta}\right)$$

General observation: if we can decrease $|\mathcal{H}|$ without losing good solutions, that's a good thing.

Simple PAC-Learnable Algorithm for And-Literals Machine

```
Data: D = ⟨(x_n, y_n)⟩_{n=1}^N
Result: f
initialize: f = x_1 ∧ x_2 ∧ ··· ∧ x_d ∧ ¬x_1 ∧ ¬x_2 ∧ ··· ∧ ¬x_d;
for n ∈ {1, ..., N} do
    if y_n = +1 then
        for j ∈ {1, ..., d} do
            if x_n[j] = 0 then
                remove x_j from f
            end
            else
                remove ¬x_j from f
            end
        end
    end
end
return f
```

Algorithm 1: ThrowOutBadTerms

Another Example: Lookup Table

Suppose $\mathcal{H}$ is all lookup tables, where we map every vector in $\{0, 1\}^d$ to a binary value.

Another Example: Lookup Table

Suppose $\mathcal{H}$ is all lookup tables, where we map every vector in $\{0,1\}^d$ to a binary value.

$|\mathcal{H}| = 2^{2^d}$

Another Example: Lookup Table

Suppose $\mathcal{H}$ is all lookup tables, where we map every vector in $\{0,1\}^d$ to a binary value.

$|\mathcal{H}| = 2^{2^d}$

$$N \geq \frac{0.69}{\epsilon}\left(2^d + \log_2 \frac{1}{\delta}\right)$$

Shallow Decision Trees (Binary Features, Binary Classification)

Let $\mathcal{H}^{(k)}$ contain all decision trees of depth k.

$$
\begin{aligned}
|\mathcal{H}^{(0)}| &= 2 \\
|\mathcal{H}^{(k)}| &= d \cdot |\mathcal{H}^{(k-1)}|^2
\end{aligned}
$$

So $\log_2 |\mathcal{H}^{(k)}| = (2^k - 1) \cdot (1 + \log_2 d) + 1$, and we need

$$
N \geq \frac{0.69}{\epsilon} \left((2^k - 1) \cdot (1 + \log_2 d) + 1 + \log_2 \frac{1}{\delta} \right)
$$

(The rest of the slides are from the wrap-up on November 29.)

Quick Review

- (ϵ, δ) PAC-learners (and efficiency)
- For a finite hypothesis class $\mathcal{H}$ that contains h^*, and noise-free data:

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

- Analyses for and-literal machines, lookup table machines, k-depth decision trees.

Limitations

- We've assumed no noise.
- We've assumed that $\mathcal{H}$ is finite.

Limitations

- We've assumed no noise.
- We've assumed that $\mathcal{H}$ is finite.

Theoretical results for infinite $\mathcal{H}$ rely on measures of complexity like the Vapnik-Chernovenkis (VC) dimension, which typically we can only bound.

Limitations

- We've assumed no noise.
- We've assumed that $\mathcal{H}$ is finite.

Theoretical results for infinite $\mathcal{H}$ rely on measures of complexity like the Vapnik-Chernovenkis (VC) dimension, which typically we can only bound.

The VC dimension of a hypothesis space $\mathcal{H}$ over input space $\mathcal{X}$ is the largest K such that there exists a set of K elements of $\mathcal{X}$ (call it X) such that for any binary labeling of X, some $h \in \mathcal{H}$ matches the labeling.