Probably Approximately Correct (PAC) Learning

- Imagine we’re doing classification with categorical inputs.
- All outputs are binary.
- Data is noiseless.
- There’s a machine \( f(x, h) \) which has \( H \) possible settings (a.k.a. hypotheses), called \( h_1, h_2, \ldots, h_H \).
Example of a machine

- \( f(x,h) \) consists of all logical sentences about \( X_1, X_2 \ldots X_m \) that contain only logical ands.
- Example hypotheses:
  - \( X_1 \land X_3 \land X_{19} \)
  - \( X_3 \land X_{18} \)
  - \( X_7 \)
  - \( X_1 \land X_2 \land X_2 \land x_4 \ldots \land X_m \)
- Question: if there are 3 attributes, what is the complete set of hypotheses in \( f \)?

<table>
<thead>
<tr>
<th>True</th>
<th>X2</th>
<th>X3</th>
<th>X2 \land X3</th>
</tr>
</thead>
<tbody>
<tr>
<td>X1</td>
<td>X1 \land X2</td>
<td>X1 \land X3</td>
<td>X1 \land X2 \land X3</td>
</tr>
</tbody>
</table>
And-Positive-Literals Machine

- $f(x,h)$ consists of all logical sentences about $X_1$, $X_2$ .. $X_m$ that contain only logical ands.
- Example hypotheses:
  - $X_1 \land X_3 \land X_{19}$
  - $X_3 \land X_{18}$
  - $X_7$
  - $X_1 \land X_2 \land X_2 \land x_4 \ldots \land X_m$
- Question: if there are $m$ attributes, how many hypotheses in $f$?

- $H = 2^m$
And-Literals Machine

- $f(x,h)$ consists of all logical sentences about $X_1$, $X_2$ .. $X_m$ or their negations that contain only logical ands.
- Example hypotheses:
  - $X_1 \land \neg X_3 \land X_{19}$
  - $X_3 \land \neg X_{18}$
  - $\neg X_7$
  - $X_1 \land X_2 \land \neg X_3 \land ... \land X_m$
- Question: if there are 2 attributes, what is the complete set of hypotheses in $f$?

<table>
<thead>
<tr>
<th></th>
<th>True</th>
<th>True</th>
<th>X2</th>
<th>~X2</th>
<th>X1</th>
<th>True</th>
<th>X2</th>
<th>~X2</th>
<th>~X1</th>
<th>True</th>
<th>~X1</th>
<th>~X2</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>X1</td>
<td>~X2</td>
<td>X1</td>
<td>^X2</td>
<td>X1</td>
<td>~X2</td>
<td>~X1</td>
<td>^X1</td>
<td>~X2</td>
<td></td>
</tr>
</tbody>
</table>
And-Literals Machine

- Equivalent to what we’ve called pure conjunctive concept descriptions when the attributes are Boolean
- E.g. $X_1 \land \neg X_3 \land X_{19}$ is equivalent to $\left( X_1 = \text{true} \right) \land \left( X_3 = \text{false} \right) \land \left( X_{19} = \text{true} \right)$

And-Literals Machine

- $f(x,h)$ consists of all logical sentences about $X_1$, $X_2$, .. $X_m$ or their negations that contain only logical ands.
- Example hypotheses:
  - $X_1 \land \neg X_3 \land X_{19}$
  - $X_3 \land \neg X_{18}$
  - $\neg X_7$
  - $X_1 \land X_2 \land \neg X_3 \land ... \land X_m$
- Question: if there are $m$ attributes, what is the size of the complete set of hypotheses in $f$?
And-Literals Machine

- \( f(x,h) \) consists of all logical sentences about \( X_1, X_2 \ldots X_m \) or their negations that contain only logical ands.
- Example hypotheses:
  - \( X_1 \land \neg X_3 \land X_{19} \)
  - \( X_3 \land \neg X_{18} \)
  - \( \neg X_7 \)
  - \( X_1 \land X_2 \land \neg X_3 \land \ldots \land X_m \)
- Question: if there are \( m \) attributes, what is the size of the complete set of hypotheses in \( f? \) \((H = 3^m)\)

```
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>X2</td>
<td>True</td>
</tr>
<tr>
<td>~X2</td>
<td>True</td>
</tr>
</tbody>
</table>
```

Lookup Table Machine

- \( f(x,h) \) consists of all truth tables mapping combinations of input attributes to true and false.
- Example hypothesis:
- Question: if there are \( m \) attributes, what is the size of the complete set of hypotheses in \( f? \)
Lookup Table Machine

- \( f(x,h) \) consists of all truth tables mapping combinations of input attributes to true and false
- Example hypothesis:
  
  \[
  \begin{array}{cccc}
  \text{X1} & \text{X2} & \text{X3} & \text{X4} & \text{Y} \\
  0 & 0 & 0 & 0 & 0 \\
  0 & 0 & 0 & 1 & 1 \\
  0 & 0 & 1 & 0 & 1 \\
  0 & 0 & 1 & 1 & 0 \\
  0 & 1 & 0 & 0 & 1 \\
  0 & 1 & 0 & 1 & 0 \\
  0 & 1 & 1 & 0 & 0 \\
  0 & 1 & 1 & 1 & 1 \\
  1 & 0 & 0 & 0 & 0 \\
  1 & 0 & 0 & 1 & 0 \\
  1 & 0 & 1 & 0 & 0 \\
  1 & 0 & 1 & 1 & 1 \\
  1 & 1 & 0 & 0 & 0 \\
  1 & 1 & 0 & 1 & 0 \\
  1 & 1 & 1 & 0 & 0 \\
  1 & 1 & 1 & 1 & 0 \\
  \end{array}
  \]
- Question: if there are \( m \) attributes, what is the size of the complete set of hypotheses in \( f \)?

\[
H = 2^{2^m}
\]

A Game

- We specify \( f \), the machine
- Nature chooses hidden hypothesis \( h^* \)
- Nature randomly generates \( R \) datapoints
  - How is a datapoint generated?
    1. Vector of inputs \( \mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{km}) \) is drawn from a fixed unknown distrib: \( D \)
    2. The corresponding output \( y_k = f(x_k, h^*) \)
- We learn an approximation of \( h^* \) by choosing some \( h^{est} \) for which the training set error is 0
Test Error Rate

- We specify \( f \), the machine
- Nature chooses hidden hypothesis \( h^* \)
- Nature randomly generates \( R \) datapoints
  - How is a datapoint generated?
    1. Vector of inputs \( \mathbf{x}_k = (x_{k1}, x_{k2}, \ldots, x_{km}) \) is drawn from a fixed unknown distrib: \( D \)
    2. The corresponding output \( y_k = f(\mathbf{x}_k, h^*) \)
- We learn an approximation of \( h^* \) by choosing some \( h^{\text{est}} \) for which the training set error is 0
- For each hypothesis \( h \),
  - Say \( h \) is consistent if \( h \) has zero training set error: \( \text{TRAINERR}(h) = 0 \)
  - Define \( \text{TESTERR}(h) \)
    = Fraction of test points that \( h \) will classify incorrectly
    = \( P(\text{h classifies a random test point incorrectly}) \)
  - Say \( h \) is bad if \( \text{TESTERR}(h) > \varepsilon \)
  - Otherwise, say \( h \) is approximately correct

Let’s consider a worst-case scenario: Among all consistent hypotheses, if any one is bad, then there’s a danger that that’s somehow the one we end up learning. How probable is it that there is even one such consistent yet bad hypothesis?

\[
P(\text{we learn a bad } h) \\
\leq P(\exists h \mid h \text{ is consistent } \land h \text{ is bad}) \\
\leq P(h_1 \text{ is consistent } \land h_1 \text{ is bad}) \lor P(h_2 \text{ is consistent } \land h_2 \text{ is bad}) \lor \ldots \\
\leq \sum_{i=1}^{n} P(h_i \text{ is consistent } \land h_i \text{ is bad}) \\
\leq \sum_{i=1}^{n} P(h_i \text{ is consistent } \mid h_i \text{ is bad})
\]
Bounding the probability of learning a bad hypothesis

• What is $P(h_i \text{ is consistent } | h_i \text{ is bad})$?
• Note that if $h_i$ is a bad hypothesis, then the probability it classifies any single training example correctly is $\leq 1 - \epsilon$.
• Then, using the i.i.d. assumption, the probability it classifies all $R$ training examples correctly is $\leq (1 - \epsilon)^R$.
• Therefore we have shown that
  
  $$P(h_i \text{ is consistent } | h_i \text{ is bad}) \leq (1 - \epsilon)^R$$

  for any $i$.

Bounding the prob. of a bad hypothesis

• Thus
  
  $$P(\text{we learn a bad } h) \leq \sum_{i=1}^{H} P(h_i \text{ is consistent } | h_i \text{ is bad})$$

  $$\leq \sum_{i=1}^{H} (1 - \epsilon)^R$$

  $$= H(1 - \epsilon)^R$$

• We can combine this with the fact that $1 - \epsilon \leq e^{-\epsilon \epsilon}$ to conclude
  
  $$P(\text{we learn a bad } h) \leq H(1 - \epsilon)^R \leq He^{-\epsilon R}$$
Probably Approximately Correct

- Suppose we want the probability to be at least 1-δ that the \( h \) we learn is not bad.
- A sufficient condition is that
  \[
  \delta \geq He^{-\varepsilon R}
  \]
- If \( H, R, \delta, \) and \( \varepsilon \) satisfy this relationship, then with probability \( \geq 1-\delta \) we are assured that the test error rate of the \( h \) we learn is \( \leq \varepsilon \).
- The \( h \) we learn is probably (with probability \( \geq 1-\delta \)) approximately (with error rate \( \leq \varepsilon \)) correct.

PAC Learning

Two ways to use a sufficient condition like
\[
\delta \geq He^{-\varepsilon R}
\]

1. Given that we’ve found a consistent hypothesis \( h_{est} \) for a training set of size \( R \), how confident are we that its test error rate is no worse than some given \( \varepsilon \)? Like confidence intervals in statistical parameter estimation theory.
PAC Learning

Two ways to use a sufficient condition like

$$\delta \geq He^{-\varepsilon R}$$

1. Given that we’ve found a consistent hypothesis $h^{est}$ for a training set of size $R$, how confident are we that its test error rate is no worse than some given $\varepsilon$? Like confidence intervals in statistical parameter estimation theory.

2. Sample complexity: Given $\delta$ and $\varepsilon$, how large must $R$ be to guarantee that, with probability at least $1-\delta$, $h^{est}$ has a test error rate no worse than $\varepsilon$? Get an answer by solving for $R$:

$$R \geq \frac{1}{\varepsilon} \left( \ln H + \ln \frac{1}{\delta} \right)$$

PAC in action

<table>
<thead>
<tr>
<th>Machine</th>
<th>Example Hypothesis</th>
<th>$H$</th>
<th>R sufficient to PAC-learn</th>
</tr>
</thead>
<tbody>
<tr>
<td>And-positive-literals</td>
<td>$X_3 \land X_7 \land X_8$</td>
<td>$2^m$</td>
<td>$\frac{1}{\varepsilon} \left( m \ln 2 + \ln \frac{1}{\delta} \right)$</td>
</tr>
<tr>
<td>And-literals</td>
<td>$X_3 \land \neg X_7$</td>
<td>$3^m$</td>
<td>$\frac{1}{\varepsilon} \left( m \ln 3 + \ln \frac{1}{\delta} \right)$</td>
</tr>
<tr>
<td>Lookup Table</td>
<td></td>
<td>$2^{2^m}$</td>
<td>$\frac{1}{\varepsilon} \left( 2^m \ln 2 + \ln \frac{1}{\delta} \right)$</td>
</tr>
<tr>
<td>And-lits or And-lits</td>
<td>$(X_1 \land X_5) \lor (X_2 \land \neg X_7 \land X_8)$</td>
<td>$(3^m)^2 = 3^{2^m}$</td>
<td>$\frac{1}{\varepsilon} \left( 2m \ln 3 + \ln \frac{1}{\delta} \right)$</td>
</tr>
</tbody>
</table>
Extensions to PAC Analysis

• What if our learner does not produce a hypothesis with $\text{TRAINERR}(h) = 0$ (perhaps because of noisy data or limited representational power)? More generally, say $h$ is a bad hypothesis if $\text{TESTERR}(h) > \text{TRAINERR}(h) + \varepsilon$.

• In this case it turns out that the corresponding probability of learning a bad hypothesis is bounded by

$$He^{-2\varepsilon^2 R}$$

• Thus to guarantee with probability at least $1 - \delta$ that $\text{TESTERR}(h) \leq \text{TRAINERR}(h) + \varepsilon$, it is sufficient to have a training set of size

$$R \geq \frac{1}{2\varepsilon^2} \left( \ln H + \ln \frac{1}{\delta} \right)$$

Extensions to PAC Analysis

• What if our hypothesis space is infinite?

• E.g.
  • perceptrons
  • multilayer neural networks
  • support vector machines

• In this case the bounds we’ve given are useless.

• Can we still bound the probability that $\text{TESTERR}(h) \leq \text{TRAINERR}(h) + \varepsilon$ for given $\varepsilon$?
Extensions to PAC Analysis

- What if our hypothesis space is infinite?
- E.g.
  - perceptrons
  - multilayer neural networks
  - support vector machines
- In this case the bounds we’ve given are useless.
- Can we still bound the probability that \( \text{TESTERR}(h) \leq \text{TRAINERR}(h) + \varepsilon \) for given \( \varepsilon \)?
- Perhaps surprisingly, the answer is YES, at least in many situations
- Magic words: VC (Vapnik-Chervonenkis) dimension

Remarks

- This form of analysis makes no assumption about the underlying distribution of examples – just assumes same one used for both training and testing. Therefore valid for any distribution.
  - Distribution free.
- The lower bounds we’ve computed on the sample complexity are sufficient but not necessary for PAC-learning. But there are corresponding results providing lower bounds on the number of training examples necessary for PAC-learning with certain distributions.
Remarks

- The underlying randomness in this theory is based on the randomness in the training sample.
- The bounds derived from this theory are very conservative, for several reasons:
  - Designed to handle any distribution of examples, including worst-case.
  - Derivation in PAC case, for example, based on bounding the probability that there is any $h$ that is both consistent and bad – when we select one, it could easily be better than this worst-case one.

Questions to test your understanding of our PAC analysis:

1. What can be said about the best-case consistent hypothesis?
2. Can you see how to easily make a very, very slight improvement in the bound we derived on the probability of learning a bad $h$?

What you should know

- Be able to understand every step in the math that gets you to

$$P(\text{we learn a bad } h) \leq H(1-\epsilon)^R \leq He^{-\epsilon R}$$

- Understand that you thus need this many records to PAC-learn a machine with $H$ hypotheses

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

- Understand examples of deducing $H$ for various machines.
What you should know

• Understand the generalization to nonzero training error, where having this many records is sufficient to guarantee with high probability that \( \text{TESTERR}(h) \) is not much worse than \( \text{TRAINERR}(h) \) when learning a machine with \( H \) hypotheses:

\[
R \geq \frac{1}{2e^2} \left( \ln H + \ln \frac{1}{\delta} \right)
\]