PAC-learning

Andrew W. Moore
Associate Professor
School of Computer Science
Carnegie Mellon University
www.cs.cmu.edu/~awm
awm@cs.cmu.edu
412-268-7599

Nov 30th, 2001

Probably Approximately Correct (PAC) Learning

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

Example of a machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm that contain only logical ands.
- Example hypotheses:
- X1 ^ X3 ^ X19
- X3 ^ X18
- X7
- X1 ^ X2 ^ X2 ^ x4 ... ^ Xm
- Question: if there are 3 attributes, what is the complete set of hypotheses in f?

Example of a machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm that contain only logical ands.
- Example hypotheses:
- X1 ^ X3 ^ X19
- X3 ^ X18
- X7
- X1 ^ X2 ^ X2 ^ x4 ... ^ Xm
- Question: if there are 3 attributes, what is the complete set of hypotheses in f? (H = 8)

True	X2	X3	X2 ^ X3
X1	X1 ^ X2	X1 ^ X3	X1 ^ X2 ^ X3

And-Positive-Literals Machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm that contain only logical ands.
- Example hypotheses:
- X1 ^ X3 ^ X19
- X3 ^ X18
- X7
- X1 ^ X2 ^ X2 ^ x4 ... ^ Xm
- Question: if there are m attributes, how many hypotheses in f?

And-Positive-Literals Machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm that contain only logical ands.
- Example hypotheses:
- X1 ^ X3 ^ X19
- X3 ^ X18
- X7
- X1 ^ X2 ^ X2 ^ x4 ... ^ Xm
- 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 X1, X2 .. Xm or their negations that contain only logical ands.
- Example hypotheses:
- X1 ^ ~X3 ^ X19
- X3 ^ ~X18
- ~X7
- X1 ^ X2 ^ ~X3 ^ … ^ Xm
- Question: if there are 2 attributes, what is the complete set of hypotheses in f?

And-Literals Machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm or their negations that contain only logical ands.
- Example hypotheses:
- X1 ^ ~X3 ^ X19
- X3 ^ ~X18
- ~X7
- X1 ^ X2 ^ ~X3 ^ … ^ Xm
- Question: if there are 2 attributes, what is the complete set of hypotheses in f? (H = 9)

True		True
True		X2
True		~X2
X1		True
X1	^	X2
X1	^	~X2
~X1		True
~X1	^	X2
~X1	^	~X2

And-Literals Machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm or their negations that contain only logical ands.
- Example hypotheses:
- X1 ^ ~X3 ^ X19
- X3 ^ ~X18
- ~X7
- X1 ^ X2 ^ ~X3 ^ ... ^ Xm
- Question: if there are m attributes, what is the size of the complete set of hypotheses in f?

True		True
True		X2
True		~X2
X1		True
X1	^	X2
X1	^	~X2
~X1		True
~X1	^	X2
~X1	^	~X2

And-Literals Machine

- f(x,h) consists of all logical sentences about X1, X2 .. Xm or their negations that contain only logical ands.
- Example hypotheses:
- X1 ^ ~X3 ^ X19
- X3 ^ ~X18
- ~X7
- X1 ^ X2 ^ ~X3 ^ ... ^ Xm
- Question: if there are m attributes, what is the size of the complete set of hypotheses in f? (H = 3^m)

True		True
True		X2
True		~X2
X1		True
X1	^	X2
X1	^	~X2
~X1		True
~X1	^	X2
~X1	^	~X2

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?

X1	X2	X3	X4	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

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?

$$H = 2^{2^m}$$

X1	X2	X3	X4	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

A Game

- We specify f, the machine
- Nature choose hidden random hypothesis h*
- Nature randomly generates R datapoints
  - How is a datapoint generated?
    1. Vector of inputs $\boldsymbol{x}_k = (x_{k1}, x_{k2}, x_{km})$ is drawn from a fixed unknown distrib: D
    2. The corresponding output $y_k = f(\boldsymbol{x}_k, h^*)$
- 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 choose hidden random hypothesis h*
- Nature randomly generates R datapoints
  - How is a datapoint generated?
    1. Vector of inputs $\boldsymbol{x}_k = (x_{k1}, x_{k2}, x_{km})$ is drawn from a fixed unknown distrib: D
    2. The corresponding output $y_k = f(\boldsymbol{x}_k, h^*)$
- We learn an approximation of h* by choosing some h^{est} for which the training set error is 0
- For each hypothesis *h* ,
- Say *h* is Correctly Classified (CCd) if *h* has zero training set error
- Define TESTERR(*h*)
  - = Fraction of test points that *h* will classify correctly
  - = P(*h* classifies a random test point correctly)
- Say *h* is BAD if TESTERR(*h*) > ε

Test Error Rate

- We specify f, the machine
- Nature choose hidden random hypothesis h*
- Nature randomly generates R datapoints
  - How is a datapoint generated?
    1. Vector of inputs $\mathbf{x}_k = (x_{k1}, x_{k2}, x_{km})$ 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^{est} for which the training set error is 0
- For each hypothesis *h* ,
- Say *h* is Correctly Classified (CCd) if *h* has zero training set error
- Define TESTERR(*h*)
  - = Fraction of test points that i will classify correctly
  - = P(*h* classifies a random test point correctly)
- Say *h* is BAD if TESTERR(*h*) > ε

$$P(h \text{ is CCd} \mid h \text{ is bad}) =$$
$$P(\forall k \in \text{Training Set}, f(x_k, h) = y_k \mid h \text{ is bad})$$
$$\le (1-e)^R$$

Test Error Rate

- We specify f, the machine
- Nature choose hidden random hypothesis h*
- Nature randomly generates R datapoints
  - How is a datapoint generated?
    1. Vector of inputs $\mathbf{x}_k = (x_{k1}, x_{k2}, x_{km})$ 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^{est} for which the training set error is 0
- For each hypothesis *h* ,
- Say *h* is Correctly Classified (CCd) if *h* has zero training set error
- Define TESTERR(*h*)
  - = Fraction of test points that i will classify correctly
  - = P(*h* classifies a random test point correctly)
- Say *h* is BAD if TESTERR(*h*) > ε

$$P(h \text{ is CCd} \mid h \text{ is bad}) =$$
$$P(\forall k \in \text{Training Set}, f(x_k, h) = y_k \mid h \text{ is bad})$$
$$\le (1-e)^R$$

$$P(\text{we learn a bad } h) \le$$
$$P\begin{pmatrix}\text{the set of CCd } h\text{'s} \\ \text{contains a bad } h\end{pmatrix} =$$
$$P(\exists h.\, h \text{ is CCd} \wedge h \text{ is bad}) =$$
$$P\begin{pmatrix}(h_1 \text{ is CCd} \wedge h_1 \text{ is bad}) \vee \\ (h_2 \text{ is CCd} \wedge h_2 \text{ is bad}) \vee \\ \vdots \\ (h_H \text{ is CCd} \wedge h_H \text{ is bad})\end{pmatrix} \le$$
$$\sum_{i=1}^{H} P(h_i \text{ is CCd} \wedge h_i \text{ is bad}) \le$$
$$\sum_{i=1}^{H} P(h_i \text{ is CCd} \mid h_i \text{ is bad}) =$$
$$H \times P(h_i \text{ is CCd} \mid h_i \text{ is bad}) \le H(1-e)^R$$

PAC Learning

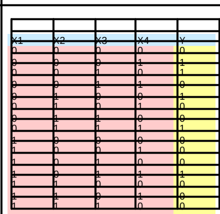
- Chose *R* such that with probability less than δ we'll select a bad h^{est} (i.e. an h^{est} which makes mistakes more than fraction ε of the time)
- Probably Approximately Correct
- As we just saw, this can be achieved by choosing *R* such that

$$d = P(\text{we learn a bad } h) \leq H(1-e)^R$$

- i.e. *R* such that

$$R \geq \frac{0.69}{e}\left(\log_2 H + \log_2 \frac{1}{d}\right)$$

PAC in action

Machine	Example Hypothesis	H	R required to PAC-learn
And-positive-literals	X3 ^ X7 ^ X8	2^m	$\frac{0.69}{e}\left(m + \log_2 \frac{1}{d}\right)$
And-literals	X3 ^ ~X7	3^m	$\frac{0.69}{e}\left((\log_2 3)m + \log_2 \frac{1}{d}\right)$
Lookup Table		2^{2^m}	$\frac{0.69}{e}\left(2^m + \log_2 \frac{1}{d}\right)$
And-lits or And-lits	(X1 ^ X5) v (X2 ^ ~X7 ^ X8)	$(3^m)^2 = 3^{2m}$	$\frac{0.69}{e}\left((2\log_2 3)m + \log_2 \frac{1}{d}\right)$

PAC for decision trees of depth k

- Assume *m* attributes
- H_k = Number of decision trees of depth *k*

- $H_0 = 2$
- H_{k+1} = (#choices of root attribute) *
  (# possible left subtrees) *
  (# possible right subtrees)
  $= m * H_k * H_k$

- Write $L_k = \log_2 H_k$
- $L_0 = 1$
- $L_{k+1} = \log_2 m + 2L_k$
- So $L_k = (2^k-1)(1+\log_2 m) + 1$
- So to PAC-learn, need

$$R \geq \frac{0.69}{e}\left((2^k - 1)(1+\log_2 m) + 1 + \log_2 \frac{1}{d}\right)$$

What you should know

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

$$d = P(\text{we learn a bad } h) \leq H(1-e)^R$$

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

$$R \geq \frac{0.69}{e}\left(\log_2 H + \log_2 \frac{1}{d}\right)$$

- Understand examples of deducing H for various machines