MACHINE LEARNING

Probably Approximately Correct (PAC) Learning

Alessandra Giordani

Department of Information Engineering and Computer Science
University of Trento
Email: agiordani@disi.unitn.it

Objectives: defining a well defined statistical framework

- What can we learn and how can we decide if our learning is effective?
- Efficient learning with many parameters
- Trade-off (generalization/and training set error)
- How to represent real world objects

Objectives: defining a well defined statistical framework

- What can we learn and how can we decide if our learning is effective?
- Efficient learning with many parameters
- Trade-off (generalization/and training set error)
- How to represent real world objects

PAC Learning Definition (1)

- Let ***c*** be the function (i.e. a ***concept***) we want to learn
- Let ***h*** be the learned concept and *x* an instance (e.g. a person)
- $error(h) = Prob\ [c(x) \neq h(x)]$
- It would be useful if we could find:
- $Pr(error(h) > \varepsilon\ ) < \delta$
- Given a target error ε, the probability to make a larger error is less δ

Definizione di PAC Learning (2)

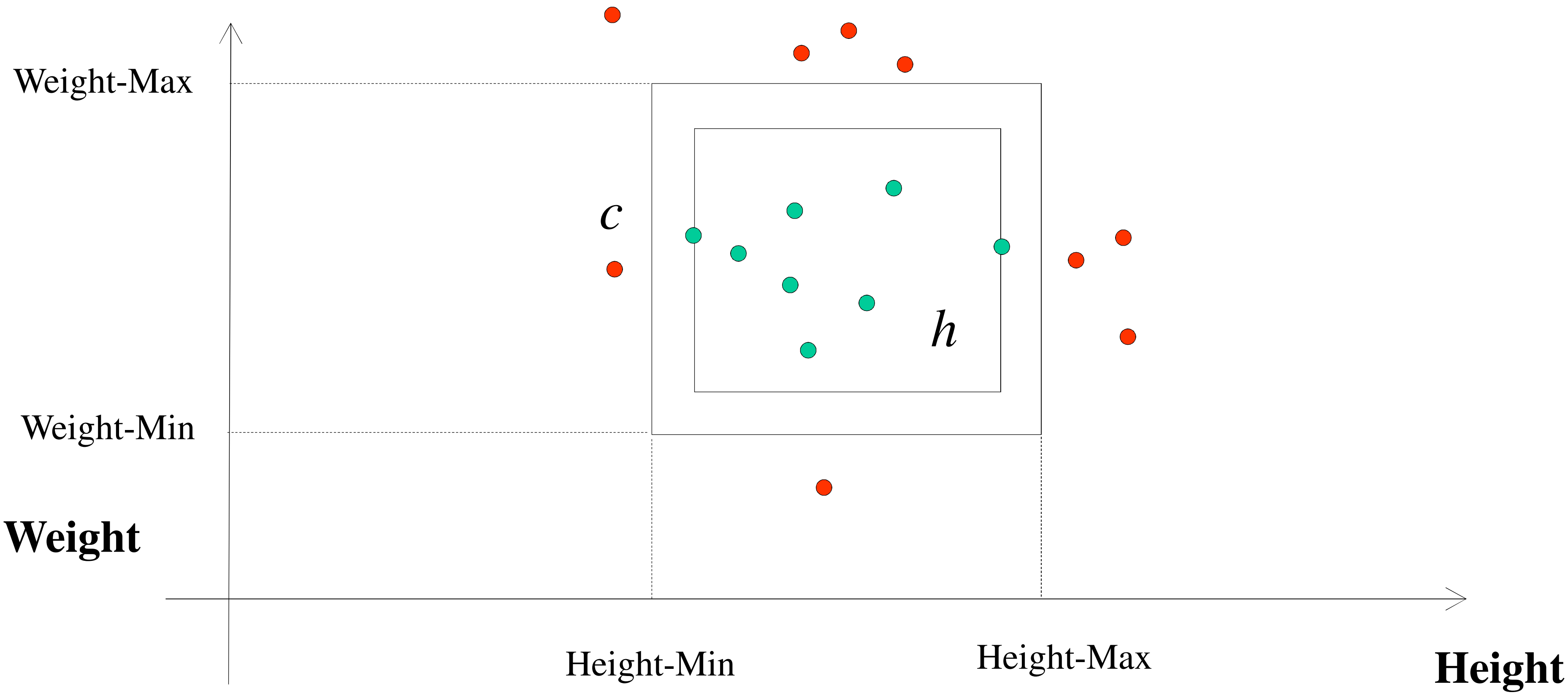
- This methodology is called Probably Approximately Correct Learning
- The smaller ε and δ are the better the learning is
- Problem:
  - Given ε and δ, determine the size m of the training-set.
  - Such size may be independent of the learning algorithm
- **Let us do it for a simple learning problem**

A simple learning problem

- Learning the concept of **medium-built people** from examples:
  - *Interesting features* are: Height and Weight.
  - The **training-set** of examples has a cardinality of m.

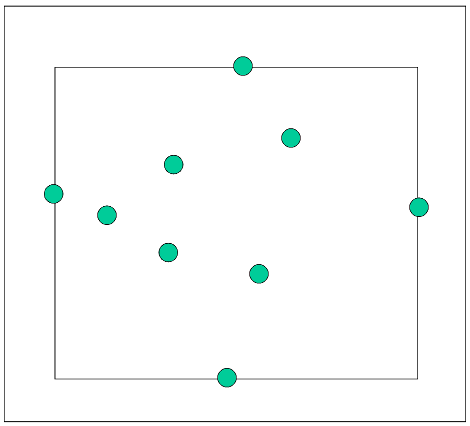
    (m people for who we know if they are medium-built people size, their height and their size).
- Find m to learn this concept ***well***.
- The adjective "well" can be expressed with probability error.

Graphical Representation of the target learning problem

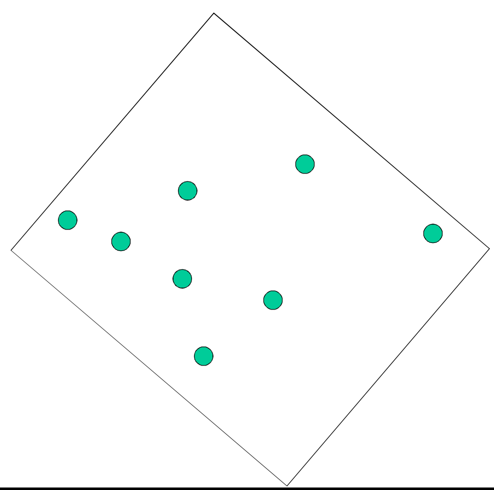


Learning Algorithm and Learning Function Class

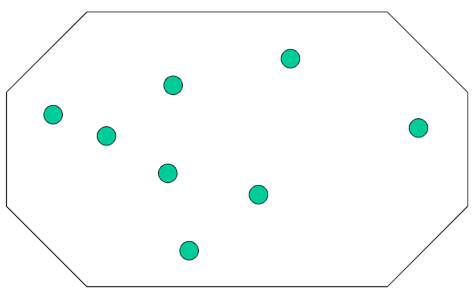
1. If no positive examples of the concept are available
   $\Rightarrow$ the learned concept is NULL
2. Else the concept is the smallest rectangular (parallel to the axes) containing all positive examples



We don’t consider other complex hypotheses



We don’t consider other complex hypothesis

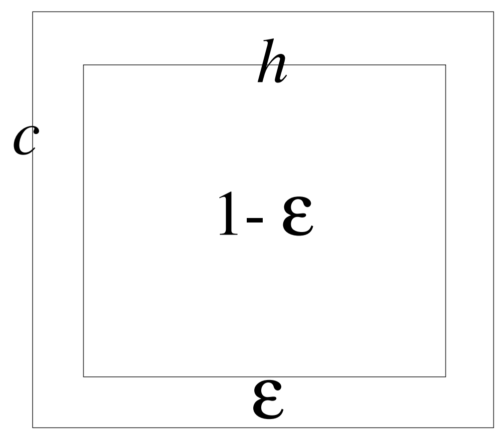


How good is our algorithm?

- An example x is misclassified if it falls between the two rectangles.
- Let ε be the measure of the area

⇒ The error probability (error) of h is ε

  - With which assumption?



Proving PAC Learnability

- Given an error ε and a probability δ, how many training examples m are needed to learn the concept?
- We can find a bound to δ, *i.e.* the probability of learning a function h with an error $> \varepsilon$.
- For this purpose, let us compute the probability of selecting a hypothesis h which:
  - correctly classifies m training examples and;
  - shows an error greater than ε.
  - This is a ***bad*** function

Probability of Bad Hypotheses

- Given x, $P(h(x)=c(x)) < 1-\varepsilon$
  - since the error of bad function is greater than ε
- Given ε, m examples fall in the rectangle h with a probability $< (1-\varepsilon)^m$
- The probability of choosing a bad hypothesis h is $< (1-\varepsilon)^m \cdot N$
  - where N is the number of hypotheses with an error $> \varepsilon$.

Upper-bound Computation

- If we set a bound on the probability of bad hypotheses

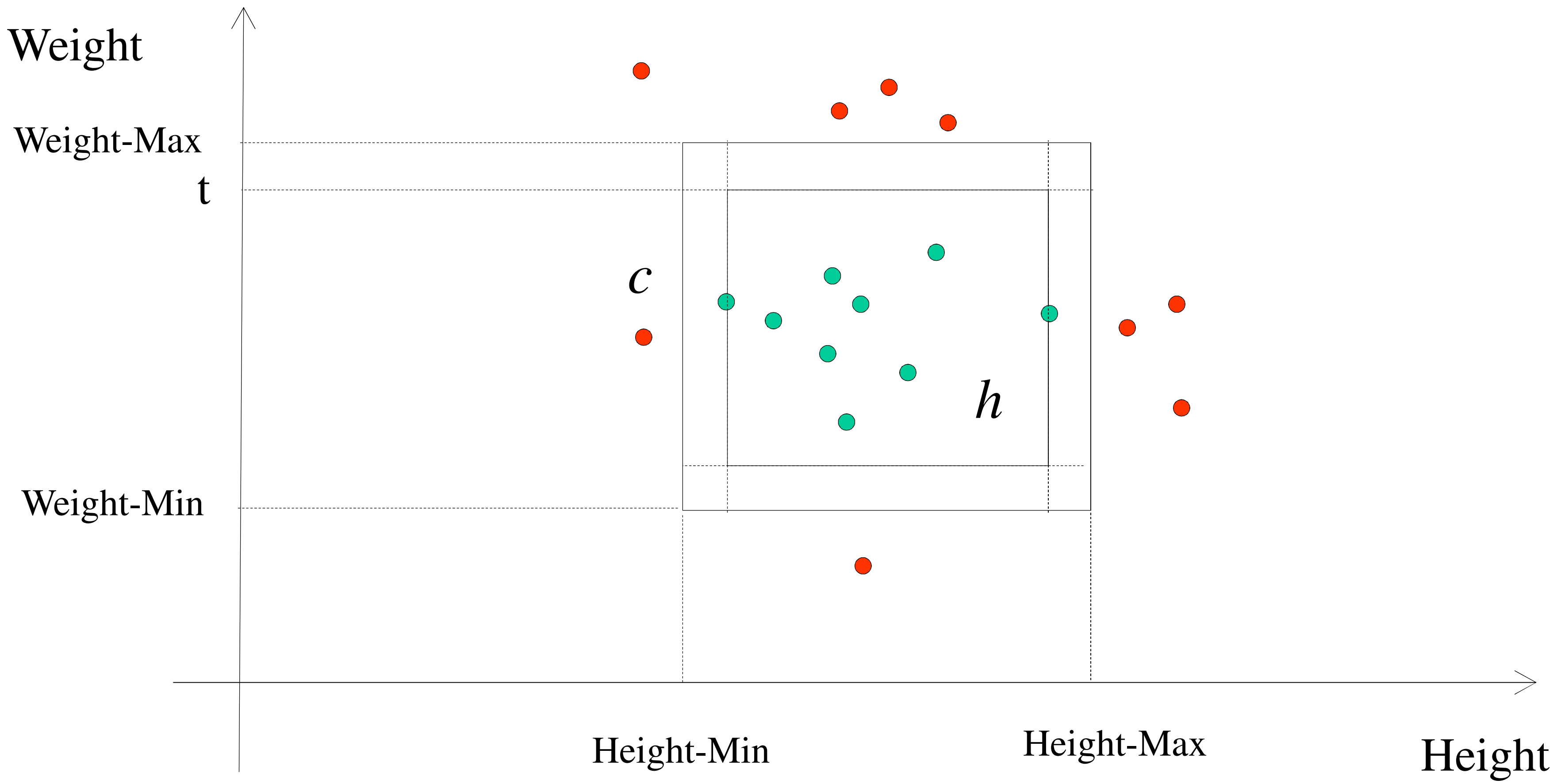
  $N \cdot (1-\varepsilon)^m < \delta$

- we would be done but we don't know N

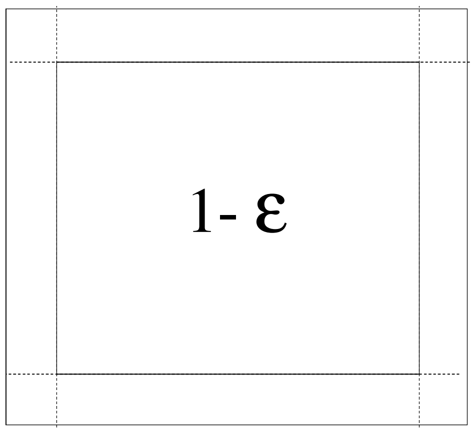
$\Rightarrow$ we have to find a bound, independent of the number of bad hypothesis.

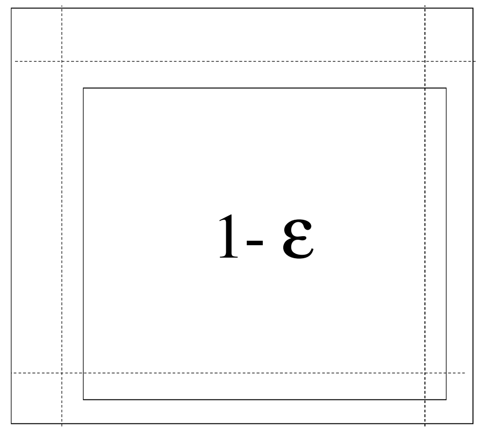
- Let us divide our rectangle in four strip of area $\varepsilon/4$

Initial Example

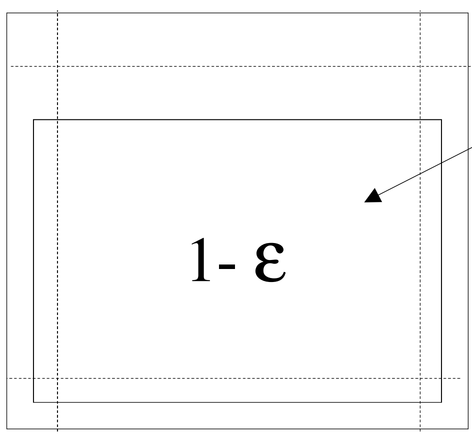


A bad hypothesis *cannot intersect more than 3 strips at a time*





Bad hypotheses with error > ε are contained in those having an error = ε



To intersect 3 edges I can increase the rectangle length but I must decrease the height to have an area ≤ 1- ε

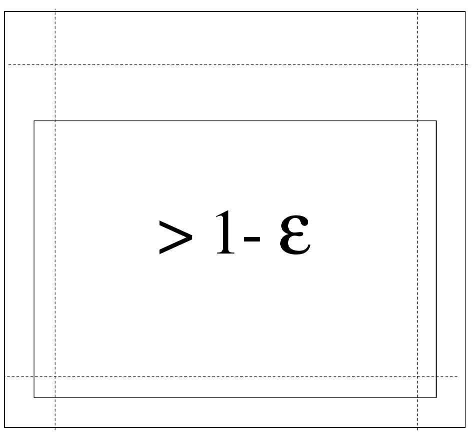
Upper-bound computation (2)

- A bad hypothesis has error > $\varepsilon \Rightarrow$ it has an area < 1- ε
- A rectangle of area < 1- ε cannot intersect 4 strips $\Rightarrow$ if the examples fall into all the 4 strips they cannot be part of the same bad hypothesis.
- A necessary condition to have a bad hypothesis is that all the *m* examples are at least outside of one strip.
- In other words, when *m* examples are outside of one of the 4 strips we may have a bad hypothesis.

  $\Rightarrow$ the probability of "*outside at least one of the strips*" > probability of bad hypothesis.

Logic view

- Bad Hypothesis $\Rightarrow$ examples out of at least one strip
  - (viceversa is not true)



- $A \Rightarrow B$
- $P(A) \leq P(B)$
- P(bad hyp.) $\leq$ P(out of one strip)

Upper-bound computation (3)

- $P(x\ out\ of\ the\ target\ strip) = (1 - \varepsilon/4)$
- $P(m\ points\ out\ of\ the\ target\ strip) = (1 - \varepsilon/4)^m$
- $P(m\ points\ out\ of\ at\ least\ one\ strip) < 4 \cdot (1 - \varepsilon/4)^m$

$\Rightarrow P(\mathrm{error}(h) > \varepsilon) < 4 \cdot (1 - \varepsilon/4)^m$

Expliciting *m*

- $-ln(1-y) = y + y^2/2 + y^3/3 + \ldots$

$\Rightarrow ln(1-y) = -y - y^2/2 - y^3/3 - \ldots < -y$

$\Rightarrow (1-y) < e^{(-y)}$ it holds strictly for $y > 0$ as in our case

- from $m > ln(\delta/4)/ln(1- \varepsilon/4)$

$\Rightarrow m > ln(\delta/4)/ln(e^{(-\varepsilon/4)})$

$\Rightarrow m > ln(\delta/4)/(-\varepsilon/4) \Rightarrow m > ln(\delta/4) \cdot (4/-\varepsilon)$

$\Rightarrow m > ln((\delta/4)^{-1}) \cdot (4/\varepsilon) \Rightarrow \boldsymbol{m > (4/\varepsilon) \cdot ln(4/\delta)}$

Expliciting *m*

- Our upperbound must be lower than δ, *i.e.*
- $4 \cdot (1- \varepsilon/4)^m < \delta$

$\Rightarrow ln(1- \varepsilon/4)^m < \delta/4$

$\Rightarrow m \cdot ln(1- \varepsilon/4) < ln(\delta/4)$

$\Rightarrow m > ln(\delta/4) \,/\, ln(1- \varepsilon/4)$

- change “$>$” into “$<$” as $ln(1- \varepsilon/4) < 0$

Numeric Examples

ε	δ	m
0.1	0.1	148
0.1	0.01	240
0.1	0.001	332
0.01	0.1	1476
0.01	0.01	2397
0.01	0.001	3318
0.001	0.1	14756
0.001	0.01	23966
0.001	0.001	33176

Formal PAC-Learning Definition

- Let f be the function we want to learn, $f\colon X \rightarrow I,\ f \in F$
- D is a probability distribution on X
  - used to draw training and test sets
- $h \in H,$
  - h is the learned function and H the set of such function class
- m is the training-set size
- $error(h) = Prob\ [f(x) \neq h(x)]$
- *F is a PAC learnable function class if there is a* ***learning algorithm*** such that for each f, for all distribution D over X and for each $0 < \varepsilon,\ \delta < 1$, ***produces h :*** $P(error(h) > \varepsilon) < \delta$

Lower Bound on training-set size

- Let us reconsider the first bound that we found:
  - h is bad: $error(h) > \varepsilon$
  - $P(f(x)=h(x))$ for m examples is lower than $(1-\varepsilon)^m$
  - Multiplying by the number of bad hypotheses we calculate the probability of selecting a bad hypothesis
  - $P(bad\ hypothesis) < N \cdot (1-\varepsilon)^m < \delta$
  - $P(bad\ hypothesis) < N \cdot (e^{-\varepsilon})^m = N \cdot e^{-\varepsilon m} < \delta$

$\Rightarrow m > (1/\varepsilon)\,(ln(1/\delta) + ln(N))$

This is a general lower bound

Example

- Suppose we want to learn a boolean function in n variable
- The maximum number of different function are 2^{2^n}

$\Rightarrow m > (1/\varepsilon)\,(ln(1/\delta)+ln(2^{2^n}))=$

$= (1/\varepsilon)\,(ln(1/\delta)+2^n ln(2))$

Some Numbers

n	epsilon	delta	m
5	0.1	0.1	245
5	0.1	0.01	268
5	0.01	0.1	2450
5	0.01	0.01	2680
10	0.1	0.1	7123
10	0.1	0.01	7146
10	0.01	0.1	71230
10	0.01	0.01	71460

Computational Learing Theory

What general laws constrain inductive learning?

We seek theory to relate:

- Probability of successful learning
- Number of training examples
- Complexity of hypothesis space
- Accuracy to which target function is approximated
- Manner in which training examples presented

Sample Complexity

Target concept is the boolean-valued fn to be learned

c: X → {0,1}

How many training examples are sufficient to learn the target concept?

1. If learner proposes instances, as queries to teacher
   - Learner proposes instance x, teacher provides $c(x)$
2. If teacher (who knows c) provides training examples
   - teacher provides sequence of examples of form $\langle x, c(x) \rangle$
3. If some random process (e.g., nature) proposes instances
   - instance x generated randomly, teacher provides $c(x)$

Sample Complexity

Given:

- set of instances X
- set of hypotheses H
- set of possible target concepts C
- training instances generated by a fixed, unknown probability distribution $\mathcal{D}$ over X

Learner observes a sequence D of training examples of form $\langle x, c(x) \rangle$, for some target concept $c \in C$

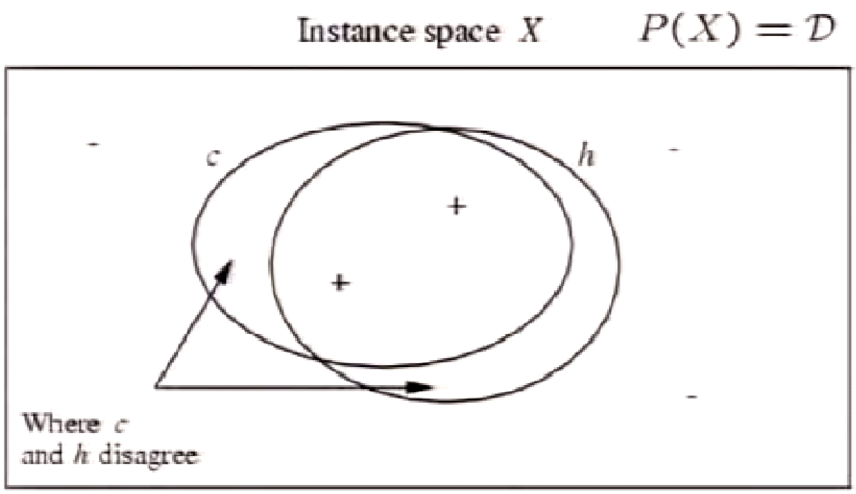
- instances x are drawn from distribution $\mathcal{D}$
- teacher provides target value $c(x)$ for each

Learner must output a hypothesis h estimating c

- h is evaluated by its performance on subsequent instances drawn according to $\mathcal{D}$

Note: randomly drawn instances, noise-free classifications

True Error of the Hypotesis



Definition: The **true error** (denoted $error_{\mathcal{D}}(h)$) of hypothesis h with respect to target concept c and distribution $\mathcal{D}$ is the probability that h will misclassify an instance drawn at random according to $\mathcal{D}$.

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

$$error_{\mathrm{D}}(h) \equiv \Pr_{x \in \mathrm{D}}[c(x) \neq h(x)] \equiv \frac{\sum_{x \in \mathrm{D}} \delta(c(x) \neq h(x))}{|\mathrm{D}|}$$

training examples

Can we bound $error_{\mathcal{D}}(h)$ in terms of $error_{\mathrm{D}}(h)$??

$$error_{\mathcal{D}}(h) \equiv \Pr_{x \in \mathcal{D}}[c(x) \neq h(x)]$$

Probability distribution P(x)

if D was a set of examples drawn from $\mathcal{D}$ and ***independent*** of *h*, then we could use standard statistical confidence intervals to determine that with 95% probability, $error_{\mathcal{D}}(h)$ lies in the interval:

$$error_{\mathrm{D}}(h) \pm 1.96 \sqrt{\frac{error_{\mathrm{D}}(h)\,(1 - error_{\mathrm{D}}(h))}{n}}$$

but D is the ***training data*** for *h* ....

$$\Pr[(\exists h \in H) s.t. (error_{train}(h) = 0) \wedge (error_{true}(h) > \epsilon)] \leq |H| e^{-\epsilon m}$$

↑

Suppose we want this probability to be at most δ

1. How many training examples suffice?

$$m \geq \frac{1}{\epsilon}(\ln|H| + \ln(1/\delta))$$

2. If $error_{train}(h) = 0$ then with probability at least (1-δ):

$$error_{true}(h) \leq \frac{1}{m}(\ln|H| + \ln(1/\delta))$$

Example: H is Conjunction of Boolean Literals

$$m \geq \frac{1}{\epsilon}(\ln|H| + \ln(1/\delta))$$

Consider classification problem f:X→Y:

- instances: $X = \langle X_1\, X_2\, X_3\, X_4 \rangle$ where each X_i is boolean
- learned hypotheses are rules of the form:
  - IF $\langle X_1\, X_2\, X_3\, X_4 \rangle = \langle 0,?,1,? \rangle$, THEN Y=1, ELSE Y=0
  - i.e., rules constrain any subset of the X_i

How many training examples *m* suffice to assure that with probability at least 0.99, *any* consistent learner will output a hypothesis with true error at most 0.05?

References

- PAC-learning:
  - **BOOK:**
  - Artificial Intelligence: a modern approach
    (Second Edition) by Stuart Russell and Peter Norvig
  - http://www.cis.temple.edu/~ingargio/cis587/readings/pac.html
  - Machine Learning, Tom Mitchell, McGraw-Hill.