

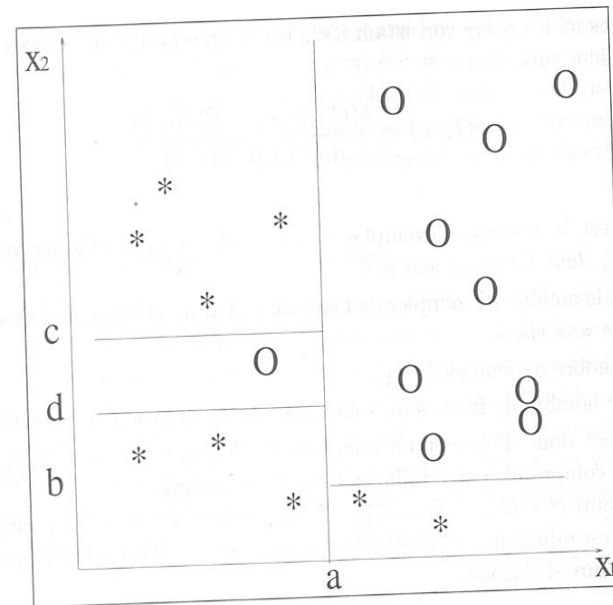
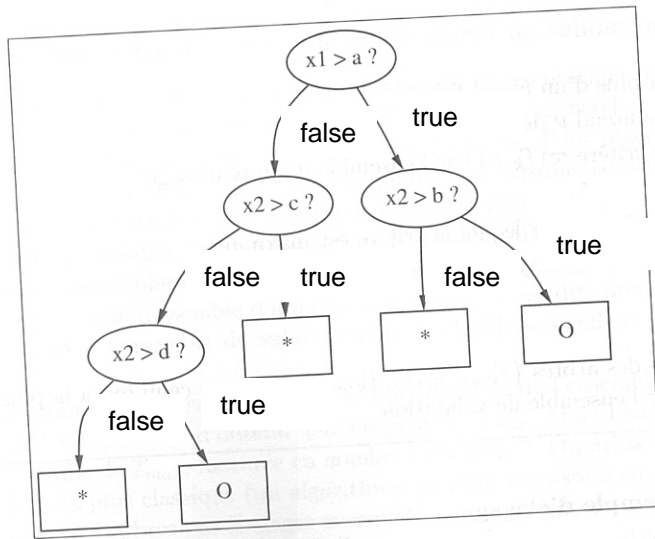
Probability Estimation Trees (PETs)

- Error rate does not consider the probability of the prediction, so in PET
- Instead of predicting a class, the leaves give a probability
- Very useful when we do not want just the class, but examples most likely to belong to a class (e.g. direct marketing)
- No additional effort in learning PET compared to DTs
- Requires different evaluation methods

Continuous attributes

- a simple trick: sort examples on the values of the attribute considered; choose the midpoint between ea two consecutive values. For m values, there are $m-1$ possible splits, but they can be examined linearly
- It's a kind of discretization (see later in class)
- cost?

Geometric interpretation of decision trees: axis-parallel area



data: two numerical attributes x_1 and x_2

Pruning

- Overfitting: getting away from training data
- Predictive performance (on data **not seen** during training)
- Pruning: discarding 1 or more subtrees and replacing them with leaves
- pruning causes the tree to misclassify some training cases. But it may improve performance on validation data

Error-based pruning: suppose error rate can be predicted. Then moving bottom-up consider replacing each subtree with a leaf, or its most frequently used branch. Do it if the replacement *decreases* the error rate

One practical way to measure the post-pruning error rate is to measure the error on a hold-out set (eg 10% of data). This hold-out set would be used for pruning purposes only

Weka's way to estimate the error rate:

- measure the actual error on the training set;
- treat it as a random variable;
- estimate standard deviation of this variable assuming a binomial distribution;
- take the lower bound for a given confidence level: eg for a 95% confidence, the error rate is observed error – 1.96* standard deviation)

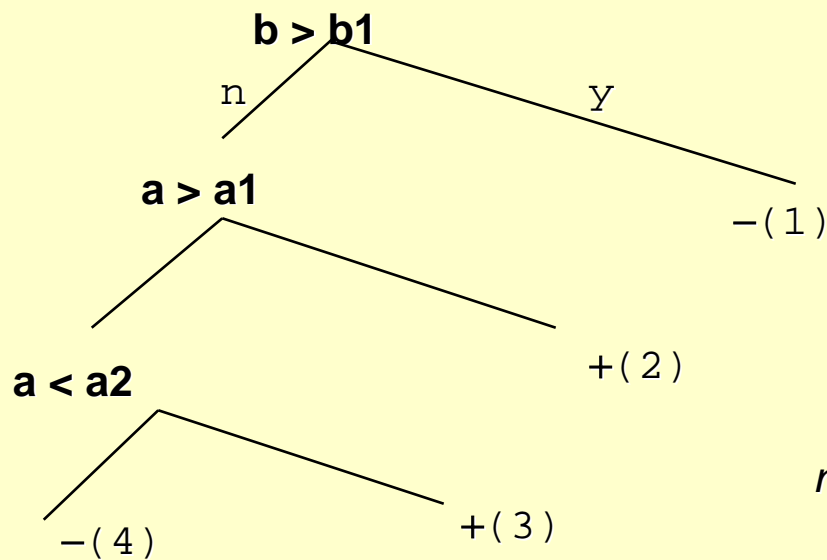
From trees to rules:

traversing a decision tree from root to leaf
gives a rule, with the path conditions as
the antecedent and the leaf as the class

rules can then be simplified by removing
conditions that do not contribute to
discriminate the nominated class from
other classes

rulesets for a whole class are simplified by
removing rules that do not contribute to
the accuracy of the whole set

Decision rules can be obtained from decision trees



(1) if $b > b1$ then class is -

(2) if $b \leq b1$ and $a > a1$ then class is +

(3) if $b \leq b1$ and $a < a2$ then class is +

(4) if $b \leq b1$ and $a2 \leq a \leq a1$ then class is -

notice the inference involved in rule (3)

Empirical evaluation of accuracy in classification tasks

- The confusion matrix
- Accuracy

- The basic idea in evaluating classifier performance is to count how many times the classifier is correct and incorrect when applied on the testing set.
- This is nicely represented in a *confusion matrix*

label	assigned=T	assigned=F
true=T	TP	FN
true=F	FP	TN

- The most common measure of classifier performance is accuracy $ACC = \frac{TP+TN}{N}$ or its complement error rate = $1-ACC = 1 - \frac{TP+TN}{N} = \frac{FN+FP}{N}$

Computing accuracy: in practice

- partition the set E of all *labeled* examples (examples with their classification labels) into a *training set* $X1$ and a *testing (validation) set* $X2$. Normally, $X1$ and $X2$ are disjoint
- use the training set for learning, obtain a hypothesis H , set $acc := 0$
- for ea. element t of the testing set,
 apply H on t ; if $H(t) = label(t)$ then $acc := acc+1$
- $acc := acc/|testing\ set|$

Testing - cont'd

- Given a dataset, how do we split it between the training set and the test set?
- cross-validation (n-fold)
 - partition E into n groups
 - choose $n-1$ groups from n , perform learning on their union
 - repeat the choice n times
 - average the n results
 - usually, $n = 3, 5, 10$
- another approach - learn on all but one example, test that example.
“Leave One Out”

Role of examples in learning

- Positive examples are used to generalize a hypothesis (or search for a more general hypothesis)
- Negative examples are used to specialize the hypothesis we have learned from the positive ones (or search for a more specific one)
- The search is constrained by the language in which we express the hypothesis

- And the search criterion is to minimize the *empirical risk*

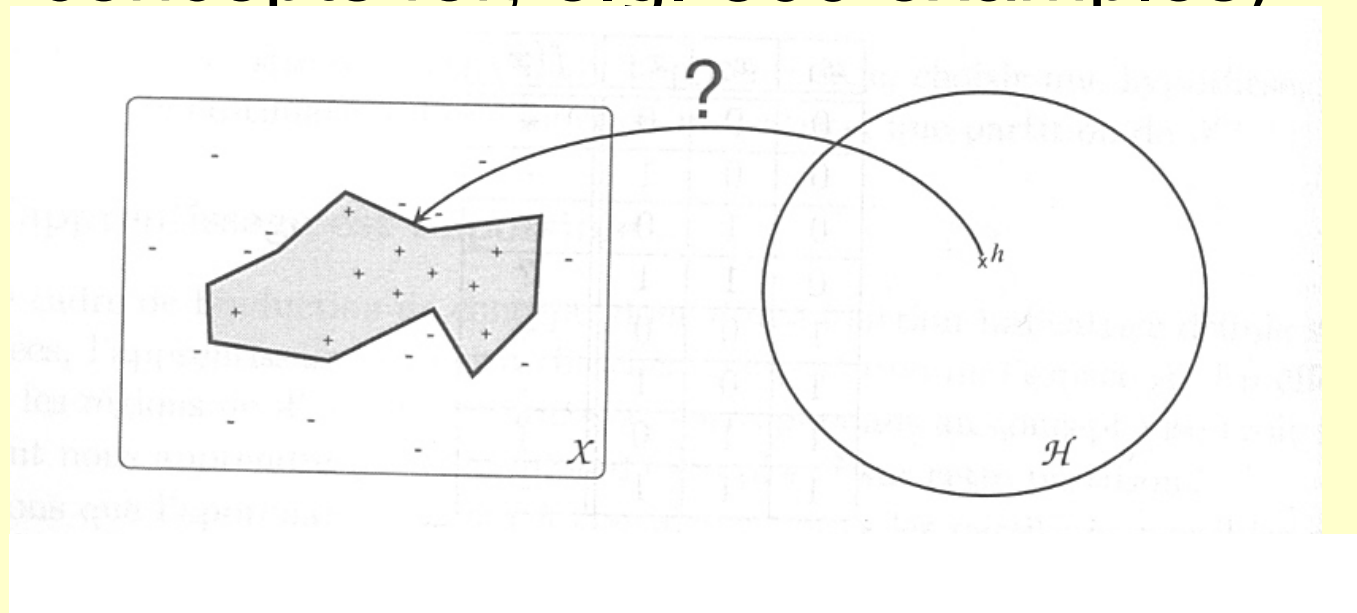
$$R(h) = \int_{x \in X, D_x} l(h(x), f(x)) dx$$

Where f is the true hypothesis, h is the learned (selected) hypothesis

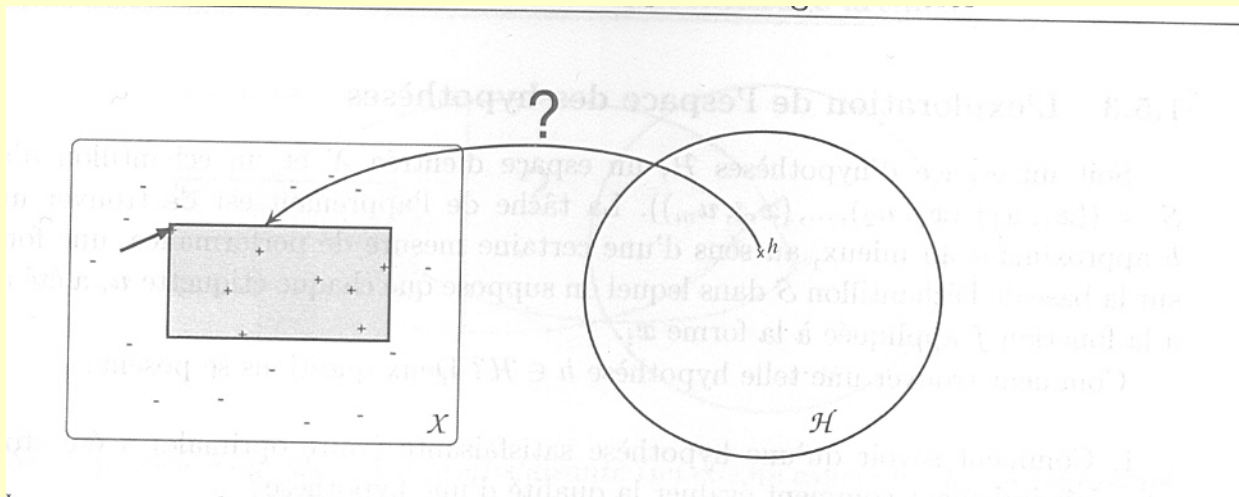
The distribution D_x is important! (eg learning to recognize taxis in NY vs Ottawa)

The language of the hypotheses must be adequate for the concept we learn:

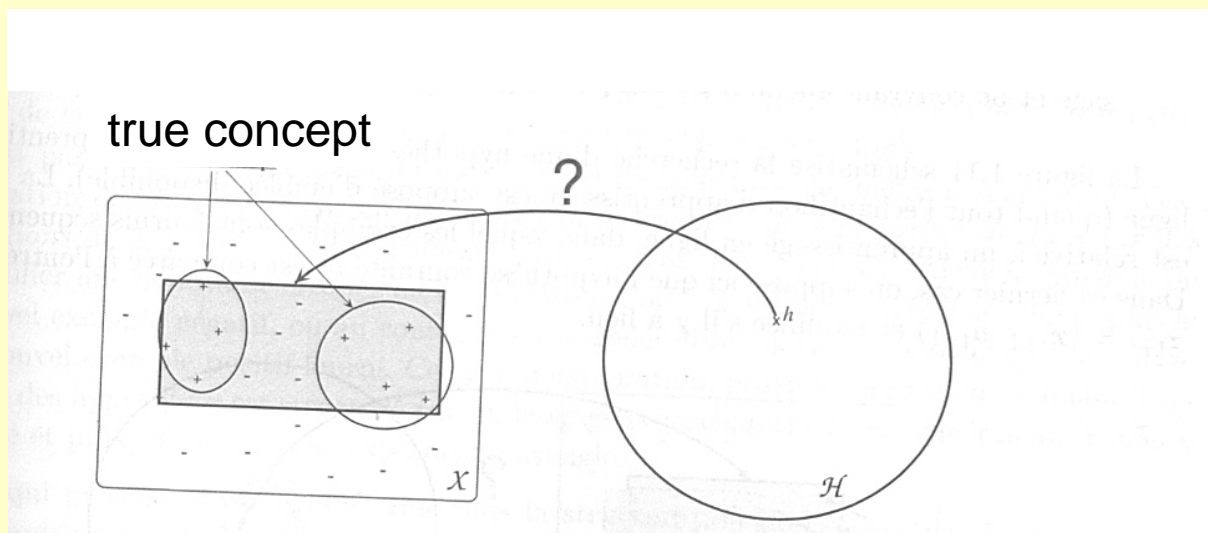
- A concept is a partition of the space of all instances (consider the number of concepts for, e.g. 800 examples)



- The shape (i.e. language) of the concept determines how much we generalize:

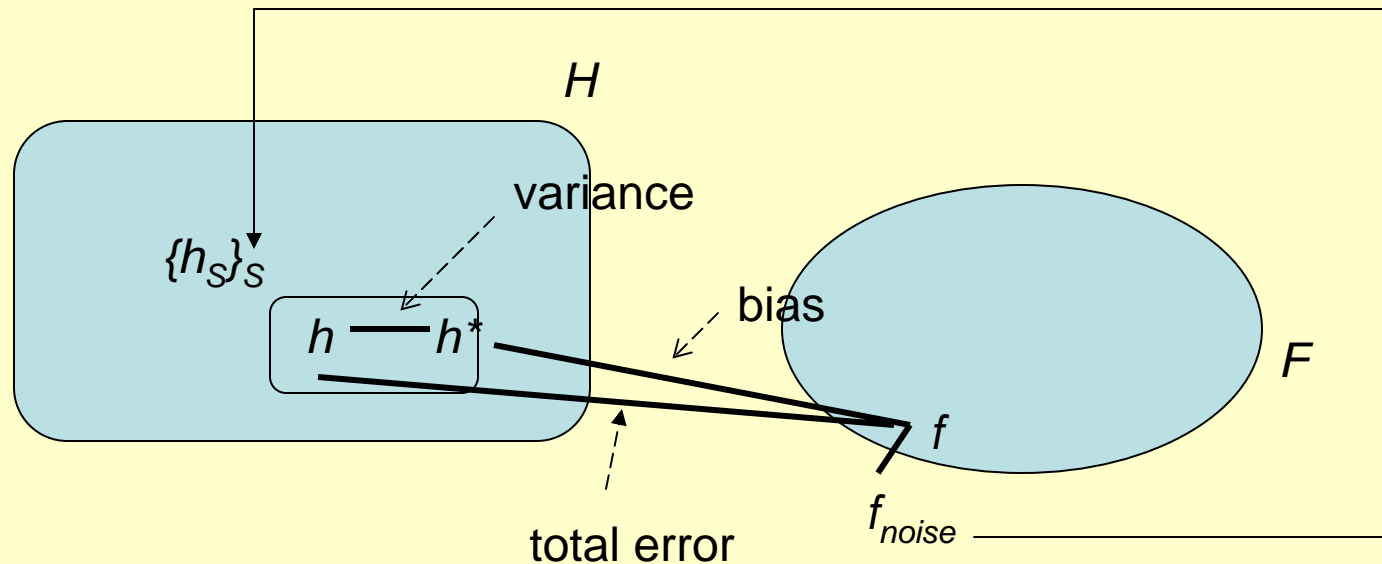


- If the language is not adequate, I will be a very poor approximation of h :



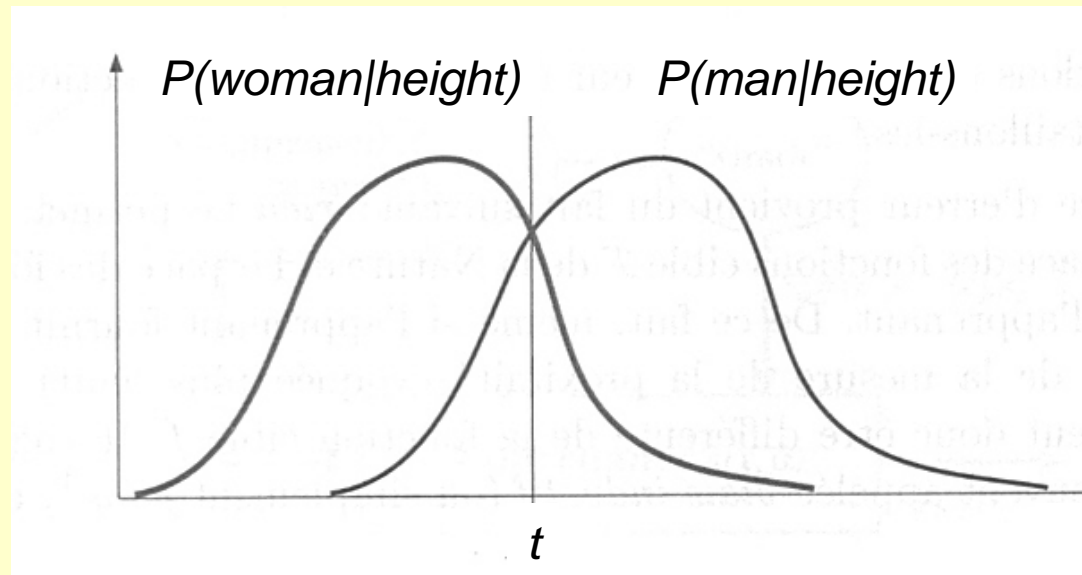
Bias-variance compromise

- Bias: the difference between h and f due to the language of H and F
- Variance (estimation error): due to inability of finding h^* , the *best* h in H



Example of bias and variance

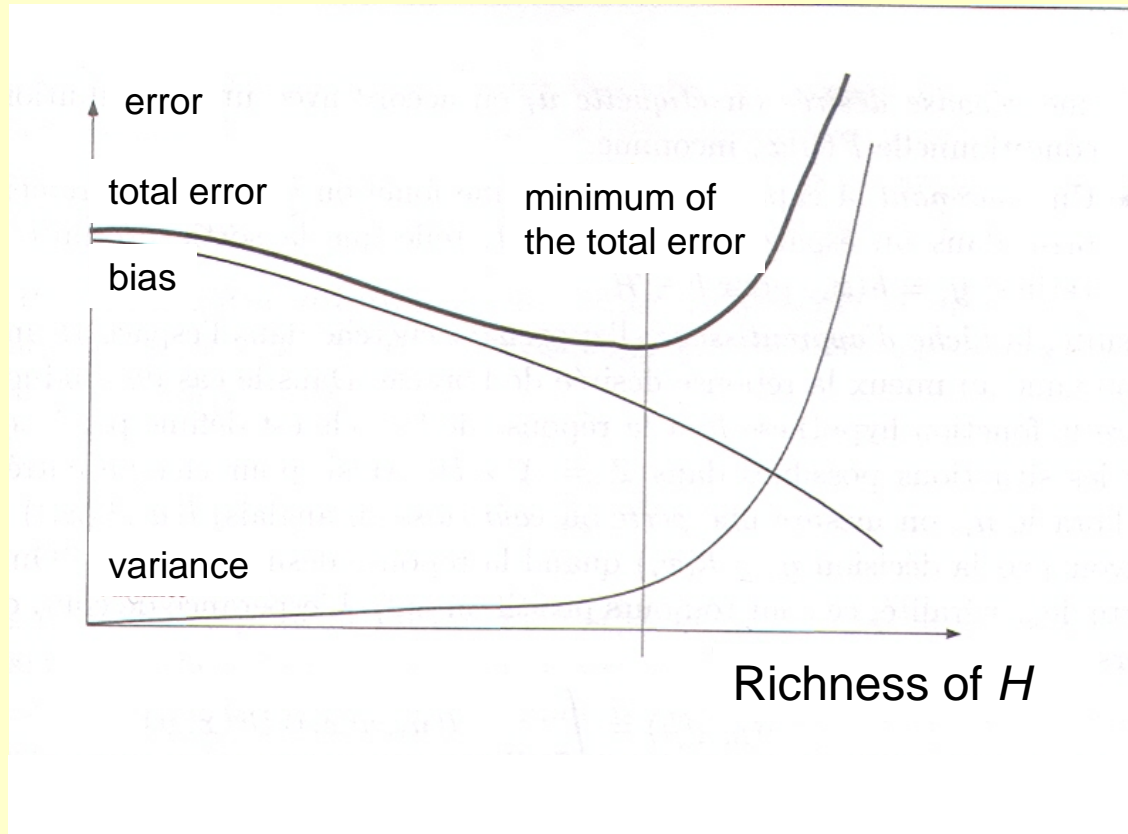
- Learning the sex of a person
- Particularly simple language bias: a “hyperplane”. For a single attribute, this is just one number (point on a line)



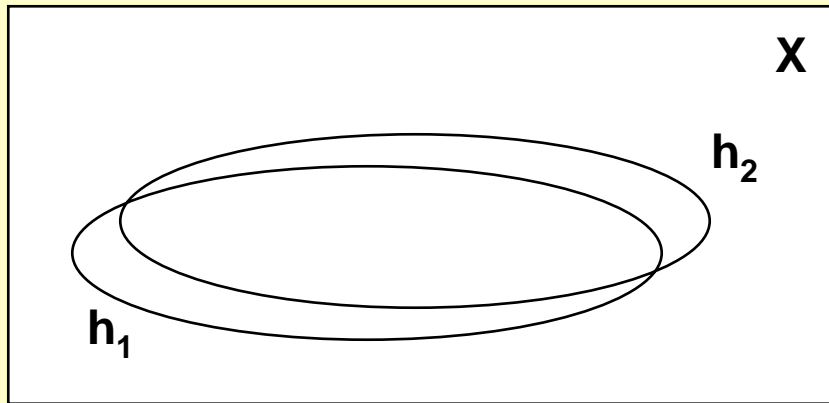
Example of bias and variance

- Bias is very bad: relatively poor choice of H leads to poor discrimination between the two classes
- Variance is good: for different samples, i.i.d. (independently and identically distributed) samples will result in same shape gaussians
- Imagine instead that we have 50 physical attributes of people. Bias is low: there may exist a perfectly discriminant function in this highly dimensional space. But variance is bad: for a limited sample we may not find the best hyperplane

Bias-variance compromise



Approximate learning



suppose h_1 is true and h_2 is not. But if $h_1 \oplus h_2$ is still small, h_2 is considered a good approximation of h_1

let P be an *unchanging* probability distribution over X . Then

$$\text{error}(h_1, h_2) = \sum_{u \in h_1 \oplus h_2} P(u)$$

Bias

There are $2^{|U|}$ possible concepts over U - why?

Bias = means to restrict that space

1. restricted hypothesis bias = syntactic restriction (e.g. a concept description is a Boolean conjunction)
2. preference bias - e.g. prefer the simplest hypothesis (Occam's razor)

PAC learning

- Assumptions: m training examples, labeled according to their membership in a concept C , and drawn *independently* from U according to some unknown distr. $P(u)$. Goal: find a hypo. $h \in H$ consistent with all m training examples. Assuming such h can be found, what is the probability that it has error greater than ϵ

- let $H_{\text{bad}} = \{h_1, \dots, h_l\}$ be the set of hypo that have error $> \varepsilon$
- If the probability that, *after m examples some element of H_{bad} is consistent with all training examples*, is SMALL, then with high probability all the remaining hypotheses consistent with training examples have error $< \varepsilon$
- consequently, any consistent hypo is probably approximately correct

PAC learning cont'd

- consider $h_1 \in H_{\text{bad}}$. what is the probability that h_1 is consistent with 1 t.e.? For that the t.e. has to be outside the error area. The probability of hitting such a t.e. is no more than $1 - \epsilon$
- with all m t.es: $(1 - \epsilon)^m$
- what is the prob. that after m t.e. some elem of H_{bad} has not been eliminated? It is $\leq |H_{\text{bad}}| (1 - \epsilon)^m \leq |H| (1 - \epsilon)^m \leq \delta$
- resolving for m , we get $m \geq \frac{1}{\epsilon} (\ln 1/\delta + \ln |H|)$

- so we have the following
- Theorem: let H be a set of hypo over U , S be a set of m te drawn independently according to $P(u)$. If h is consistent with all te in S and

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

then the probability that h has error $> \varepsilon$ is $< \delta$.

- observe that we can control δ and ε by changing the number of examples!
- example: consider that the hypo language consists of conjunctions of n Boolean variables (attributes). Then

$$m \geq \frac{1}{\varepsilon} (\ln \frac{1}{\delta} + n \cdot \ln |3|)$$

PAC cont'd

- In general, to show PAC learnability, we must show that
 - polynomial number of examples is sufficient to PAC learn
 - Show an algorithm that uses poly. time per example

PAC - cont'd

Some useful classes:

- k-term-DNF: k term disjunction where ea. term is a conjunction of Bool. vars of unlimited size; H is poly. size but learning is non-poly: -
- k-DNF: disjunction of any number of conjunctive terms, ea conjunct limited to k vars – but + with Oracle
- k-CNF: conjunction of any number of clauses (disj. terms), ea clause has at most k variables + surprising as $k\text{-CNF} \supseteq k\text{-term-DNF}$
- DNF: any Bool. expression in disjunctive normal form -

Sizes of hypo spaces:

k-term-DNF: $2^{O(kn)}$

k-DNF: $2^{O(n^*k)}$

k-CNF: $2^{O(n^*k)}$

DNF: 2^{2^*n}

The first three are potentially PAC-learnable in poly time (= number of examples) if we have a poly-time per example procedure

- but how do we move into infinite hypo spaces? There is a way of characterizing expressive power of a hypo space.
- a set of hypo *completely fits* an example set E if for every possible way of labeling elements of E as pos and neg there exists a hypo H that will produce that labeling. The size of the largest set of examples that H can completely fit is call the VC (Vapnik-Chervonenkis) dimension of H .
- for example, suppose we're 'learning' single closed intervals over the real line (ie hypo have form $[a,b]$).
- suppose $E = \{3, 4\}$. How many ways of labeling elems of E as pos or neg? Is there an interval that will produce that labeling?

- H_{int} (closed intervals on the real line) can completely fit the set E .
- but consider $E'=\{2,3,4\}$. So what is the VC dimension of H_{int} ?
- linear separability of sets of points = VC dimension of simple neural networks

Theorem (Blumer et al. 89): a space of hypo H is PAC learnable iff it has finite VC dimension. Any PAC learning algo for H must examine $O(1/\epsilon[\ln 1/\delta + VC(H)])$ examples).

- Boolean conjunction, k -DNF, and k -CNF are poly learnable, but k -term-DNF is NP-hard! Even though it is a proper subset of k -CNF.
- implications for the change of representation
- same true for k -3NNs, ie three layer NNs with exactly k hidden units. There's a conjecture that k' -3NN learnability where $k' < p(k)$, p some polynomial, could be true.

- the PAC theorem says that we may learn an expo. # of hypotheses from a poly # of examples! This is *sample complexity*. But there is also *computational complexity*, ie worst-case computation time req'd to produce a hypo from a sample of given size.

So, we say (Valiant) that a hypo. space is poly-learnable iff

- only a poly # of examples is req'd, as a function of n , ϵ and δ
- a consistent hypo in H can be found in poly time in n , ϵ and δ

connection betw. VC dimension and
the PAC theorem:

suppose H is finite, $VC(H) = d$. There
is a set of d instances I that H
completely fits. That requires $2^{**}d$
distinct hypotheses, so
 $|H| \geq 2^{**}d$.

so $VC(H) \leq \log(H)$

A lattice of learning models

