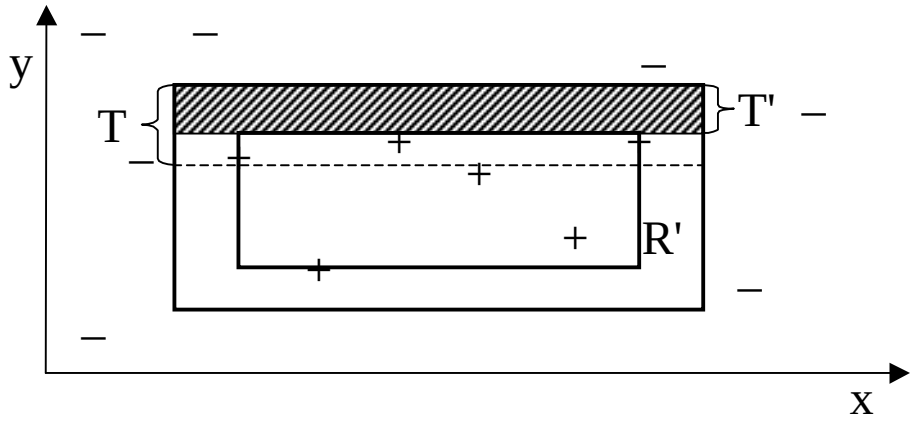
Computational Learning Theory



Univ.-Lektor Dr.techn. Alexander K. Seewald
Österreichisches Forschungsinstitut
für Artificial Intelligence

alex@seewald.at / alex.seewald.at

Computational Learning Theory (COLT)

Goals:

- Precise mathematical formalisation of the concept of *learning*
- Formal analysis of general learnability of (classes of) concepts
- Quantify/predict the number of examples necessary for reliable generalisation

Identification in the Limit (Gold, 1967)

- A successful learning algorithm must correctly identify any target concept (of a particular type) from a sufficiently large (but finite) number of examples.

Exact identification in polynomial time (Angluin, 1988)

- For what types of target concepts are there learning algorithms that can correctly learn the concept in polynomial time, and from a polynomial number of examples, with and without queries to a teacher / *oracle* ?

PAC (Probably Approximately Correct) Learning (Valiant, 1984)

- For what classes of target concepts can there be polynomial-time learning algorithms that with probability $(1-\delta)$ find an hypothesis with an error $< \varepsilon$ on new examples? (*computational complexity*) How many training examples are needed to achieve this for a given ε and δ? (*sample complexity*)

alex@seewald.at / alex.seewald.at

Concept Learning - Basic Notions

Instance Space IS: Set of all distinguishable objects describable in a given representation language L (e.g., L = conjunctions of n boolean attributes: $IS = \{0,1\}^n$)

Concept / function: c: IS $\rightarrow\{0,1\}$ (intensional) or subset c $\subseteq$ IS (extensional).

Concept Space CS: Set of all concepts possible over IS (i.e., all possible functions c: IS $\rightarrow\{0,1\}$) : CS = $\{0,1\}^{|IS|}$ *We only consider two-class tasks here*

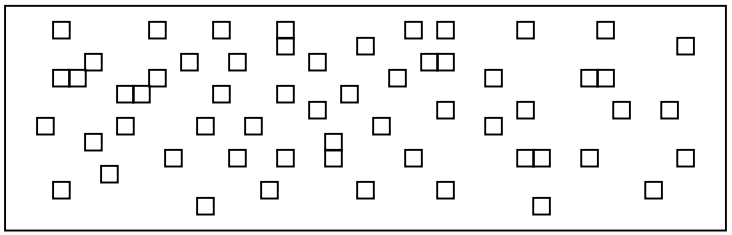
Hypothesis Space H: Set of all functions c: IS $\rightarrow\{0,1\}$ that the learning algorithm can represent; H $\subseteq$ CS

Sample S (TD): Set of pairs <x,c(x)>, S $\subseteq$ IS (x$\in$ IS randomly chosen according to probability distribution D), c(x) $\in \{0,1\}$ = class label (y)

Learning Algorithm L: Function L = f: S $\rightarrow$ H which, for a given sample S, yields a hypothesis h $\in$ H that approximates the target concept c.

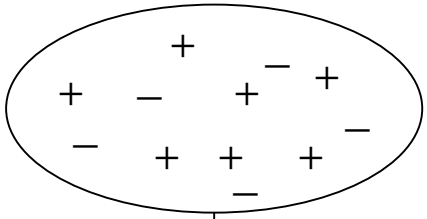
alex@seewald.at / alex.seewald.at

PAC Learning (Valiant, 1984) - Scenario



Instance Space *IS* with prob. dist. *D* (real world)

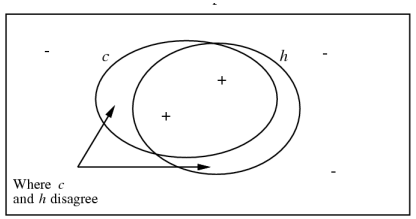
drawing and labelling ⇩ (according to *D*, *c*)



Training Sample *S*

Learning algorithm *L* (with hypothesis space *H*)

hypothesis $h \in H$



Error $err_D(h,c,IS)$=prob. that a randomly (according to *D*) chosen example is misclassified by *h* (i.e. different from true concept *c*)

$err_D(h,c,IS)=error(h)=P\{x \in IS \mid h(x) \neq c(x)\}$

alex@seewald.at / alex.seewald.at

PAC Learning

Informally:

- A good learning algorithm should make few errors most of the time (i.e., on most randomly drawn samples = TD given for a concept)

Formalisation of this idea: PAC learning

Given by the learning problem:

IS = instance space, ***CS***= concept space

D = unknown (but fixed) probability distribution over inst. space *IS*

Given by user:

- maximal acceptable error: ε
- permitted probability of non-acceptable error ($> \varepsilon$): δ (i.e., confidence $1-\delta$)

Requirements on the learning algorithm L:

- For each randomly chosen sample S of size at least $m(\varepsilon,\delta)$ (=sample complexity) and for each possible concept $c \in CS$, the probability that the hypothesis $h \in H$ learned by L has an error $> \varepsilon$ should be $< \delta$.

alex@seewald.at / alex.seewald.at

PAC Learning (more formally)

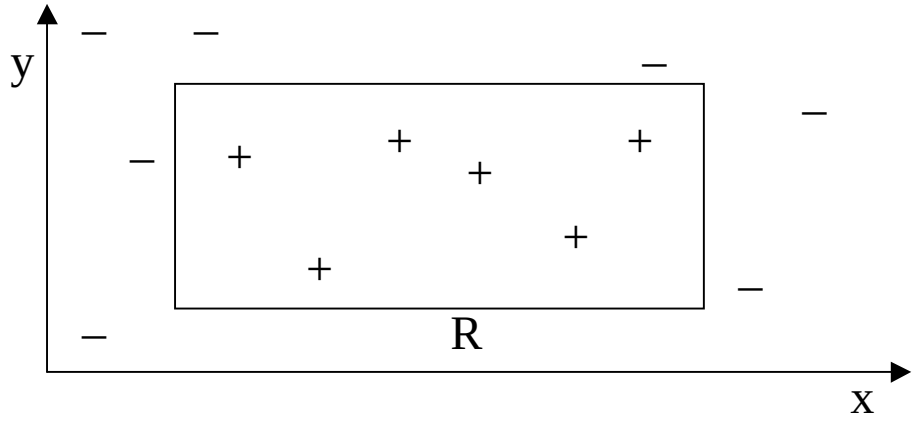
Concept space CS is PAC learnable $\Leftrightarrow$ There exists an algorithm L with the following properties: For every concept $c \in CS$, every distribution D on IS, and all $0<\varepsilon<0.5$, $0<\delta<0.5$, L (given access to ε, δ, and the oracle EX(c,D)) outputs a hypothesis concept $h \in CS$ satisfying error(h)$\leq \varepsilon$ with probability of at least $(1-\delta)$.

Efficiently PAC learnable $\Leftrightarrow$ L runs in time polynomial to both $1/\varepsilon$ and $1/\delta$. Implies that $m(\varepsilon,\delta)$ is polynomial in both $1/\varepsilon$ and $1/\delta$ – processing an example takes at least one step.

Sufficient sample complexity $m(\varepsilon,\delta)$: $m=|S|$ (no. of calls to oracle) depends on *acceptable error* ε and *1-confidence* δ.

alex@seewald.at / alex.seewald.at

Example: A Rectangle Learning Game



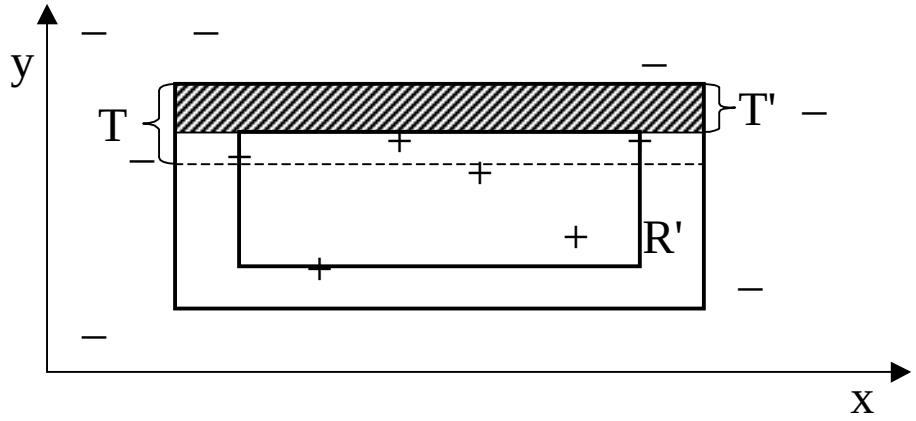
Concept space CS_R is set of all axis-parallel rectangles in $\Re^2$. Given $R \in CS_R$ = *target*.

Oracle EX(c,*D*) returns a random point and its classification <x,c(x)> according to some unknown fixed distribution *D*.

m(ε,δ) calls to the oracle define our training sample *S* / *TD* of sufficient size to learn any target concept c / R.

alex@seewald.at / alex.seewald.at

Example: A Rectangle Learning Game (2)



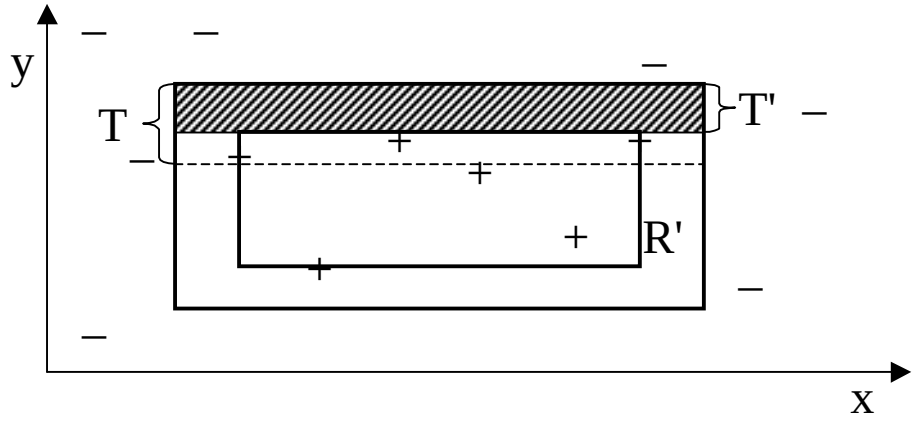
Proposed L: Return R' = tighest-fit rectangle which covers all positive examples.

Error of R' (= prob. of x falling into R-R' according to *D*) must be < ε for a sufficient number *m* of training examples.

Proof: Assume strip defined by T with weight = ε/4 under *D*. T' includes T if and only if no point in T appears in the sample S. The probability of a single draw from *D* missing T is (1- ε/4), so the prob. that all *m* draws miss T is $(1- \varepsilon/4)^m$

alex@seewald.at / alex.seewald.at

Example: A Rectangle Learning Game (3)



Proof (ctd.): The probability that any of the four strips of R-R' h as weight ≥ε/4 (equiv. to error(R') ≥ε) is thus at most $4(1- \varepsilon/4)^m$. If we choose $4(1- \varepsilon/4)^m < \delta$, then with prob. (1-δ) the error of our hypothesis R' is smaller than ε (error(R') <ε).

$m \geq (4/\varepsilon)\ln(4/\delta)$ satisfies PAC criteria: **CS_R is PAC learnable**. O(m) suffices for R' :**CS_R is also efficiently PAC learnable.**

alex@seewald.at / alex.seewald.at

Vapnik-Chervonenkis Dimension

An alternative way to estimate the complexity of a hypothesis space H (other than simply |H|) is the VC Dimension. Most useful for infinite hypothesis spaces.

Def. The *Vapnik-Chervonenkis dimension*, *VCDim*(H), of a hypothesis space H defined over instance space IS is the size of the largest finite subset of IS **shattered** by H. If arbitrary large finite subsets of IS can be shattered by H, then *VCDim*(H) = ∞.

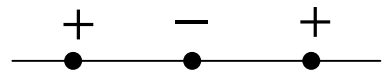
Def. A set of instances S is **shattered** by a hypothesis space H if and only if for every dichotomy of S (=every concept defined over S) there exists at least one hypothesis in H consistent with the dichotomy.

Subset $S \subseteq IS$ has $2^{|S|}$ dichotomies/concepts, but not all of them are always representable as hypothesis $h \in H$.

alex@seewald.at / alex.seewald.at

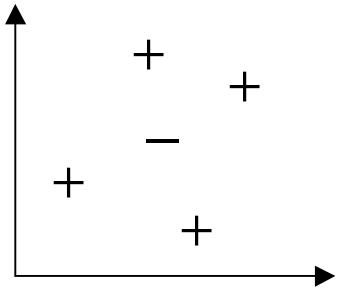
VC Dimension - Examples

H = {Interval on $\Re$} $VCDim(H)=2$



H = {Axis-parallel rectangles on $\Re^2$}

$VCDim(H)=4$



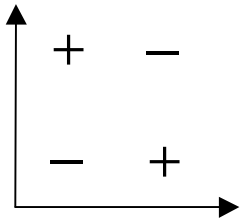
H = {Linear halfspace in $\Re^e$} $VCDim(H)=e+1$

$\Rightarrow VCDim(\text{Lin.R/Weather})=VCDim(\text{SMO,linear/Weather})=8$

$VCDim(\text{Lin.R/USPS})=257$

$VCDim(\text{Log.R/Weather}) = (c\text{-}1)p+1 = 64$

$VCDim(\text{Log.R/USPS}) = (c\text{-}1)p+1 = 2305$



alex@seewald.at / alex.seewald.at

VC Dimension - Examples (2)

VC Dimension for other classifiers

$VCDim(\text{SMO,p.,d=5/USPS}) = \binom{p+d-1}{d}+1 = \binom{256+5-1}{5}+1 = \frac{260!}{5!255!}+1$

$= 9{,}525{,}431{,}553$ (!).

$VCDim(\text{SMO,RBF/*}) = \infty$ (resp. $|IS|$)

$VCDim(\text{IBk,k=1/*}) = \infty$ (resp. $|IS|$)

$VCDim(\text{C4.5/*}) = VCDim(\text{RIPPER/*}) = \infty$ (resp. $|IS|$)

(if no pre/post pruning takes place)

$VCDim(\text{NB/*}) = ?$ (possibly $O(cp)$)

VC Dim is a *worst-case* estimate. As we see, most of our learners have infinite VC dimensionality and still work.

alex@seewald.at / alex.seewald.at

Sufficient Sample Size

Sufficient Sample Size for finite and infinite hypothesis spaces (Blumer et al., 1987/89)

Finite H: $m \geq (\ln(1/\delta)+\ln|H|)/\varepsilon$ (1)

Finite H with target concept $c \notin H$, $err(h) \leq \varepsilon + \min(err(h))$:

$m \geq (\ln(1/\delta)+\ln|H|)/(2\varepsilon^2)$ (2)

In/finite H: $m \geq (4\log_2(2/\delta)+8VCDim(H)\log_2(13/\varepsilon))/\varepsilon$ (3)

Gives the number of training examples sufficient to learn any target concept in the worst-case.

alex@seewald.at / alex.seewald.at

Necessary Sample Size

Lower bound on sufficient sample size

(Ehrenfeucht et al., 1989)

Consider concept space *CS* such that *VCDim*(CS)≥2, any learner L, and any 0< ε <1/8, 0< δ <1/100. Then there exists distribution *D* and concept c ∈ CS such that if

$$m \leq \max[\log(1/\delta)/\varepsilon;(VCDim(CS)-1)/32\varepsilon] \quad (4)$$

then with probability of at least δ, L outputs a hypothesis having $error_D(h)>\varepsilon$.

Gives minimum number of examples to PAC-learn any concept from CS. Below this *m* no learner can PAC-learn every target concept (although learners will usually be able to learn most target concepts)

alex@seewald.at / alex.seewald.at

Example Bounds

ε=0.1, δ=0.01. IS=Weather (nominal). |IS|=3*3*2*2=36.

Necessary (4): m≥20 (independent of *VSDim*(H))

VCDim = ∞ (C4.5, IBk k=1 etc..), i.e. H=CS

Sufficient (1): m≥295.58 **(>|IS|)**

VCDim < ∞ (Lin.R, SMO poly/linear etc..), i.e. H⊂CS

Sufficient (3): m≥4800.00 **(Lin.R, SMO; >|IS|)**

OneR: $|H|=2^3+2^3+2^2+2^2=24$

Sufficient (2): m≥389.16 **(>|IS|)**

- **Worst-case bounds: Valid for hardest concepts c ∈ CS; much more benign in the average case.**
- **All except formula (2) assume target concept c ∈ H.**
- **Only considers two-class problems**

alex@seewald.at / alex.seewald.at

Theory vs. Practice

COLT / PAC learning	**Applied Machine Learning**
How many training examples do I need? (sufficient sample size)	Examples are usually given (few..) and not easily extended
Want to be able to guarantee arbitrary small ε	"Natural" concepts are often not PAC-learnable due to fundamental reasons (e.g. missing attributes, noise) Minimal error may not be necessary
Want to be able to guarantee arbitrary small δ	Estimate ε and δ on separate test data or via cross-validation
Any target concept must be found. Focus on hardest concept in CS	Average-case analysis: focus on given target concept which may be simpler
Worst-case assumptions concerning target concept	Assumption: Most "natural" concepts are relatively simple, given appropriate attributes / features

alex@seewald.at / alex.seewald.at

Contributions of COLT to Applied ML

Boosting

- Equivalence of *strong* learners (PAC) and *weak* learners (PAC for fixed value of ε and δ) shown via Boosting-like procedure in [Schapire, 1990/92].

Active Learning (learners choose specific examples and expect true class instead of assuming a given training set)

- Extending PAC with membership oracles is one of the first approaches to this field, e.g. [Angluin, 1987].

Support Vector Machines

- Basic Kernel Theory and other important theorems are due to [Mercer, 1909] and [Aronszajn, 1940]. Structural risk minimization due to Vapnik (related to *VCDim* and SVMs)

alex@seewald.at / alex.seewald.at