

Advanced Machine Learning



Bogdan Alexe,

bogdan.alexe@fmi.unibuc.ro

University of Bucharest, 2nd semester, 2019-2020

Administrative

- first seminar class today, 10-12. I've uploaded the exercises for the first seminar (taken from the course book) + solutions.
- Tuesday, 8-10, the same seminar.
- the first seminar in week 2 + 3, Thursday + Tuesday

Today's lecture: Overview

- A Formal Model – The Statistical learning framework
- Empirical Risk Minimization
- Probably Approximately Correct learning
- The general Probably Approximately Correct learning model

Particular learning scenario – papaya tasting

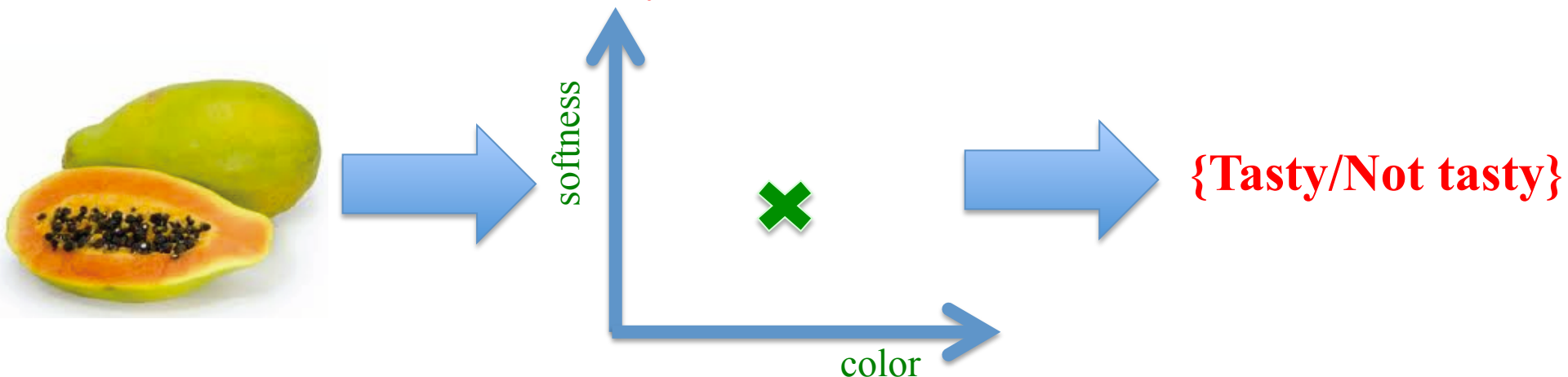
**You've just arrived in some
small Pacific island**



**You soon find that papayas are a
significant ingredient in the local diet**



**Based on previous experience with other
fruits, you decide to use two features**



Formal model for statistical learning

Example

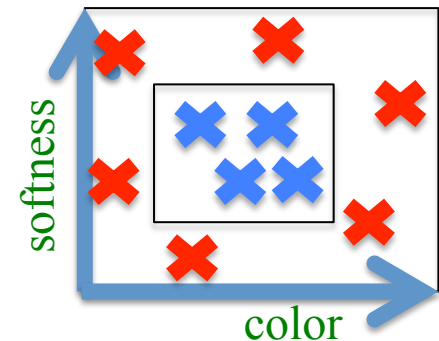
- **domain set, \mathcal{X}** : this is the set of objects that we may wish to label.
 - instance space
 - elements of \mathcal{X} are called instances, represented by *features*
 - **label set, \mathcal{Y}** : the set of possible labels.
 - usually $\{0,1\}$ or $\{-1,+1\}$
 - **training data S**
 - the learner's input
 - supervised batch learning scenario
 - finite sequence of pairs S of labeled domain points: $S = ((x_1, y_1) \dots (x_m, y_m)) \in (\mathcal{X} \times \mathcal{Y})^m$
- $\mathcal{X} = \mathbf{R}^2$ representing color and shape of papayas.
 - $\mathcal{Y} = \{0, 1\}$ representing “tasty” (1) or “non-tasty”(0)
 - S = set of already tasted papayas and their color, softness and tastiness (label)

Formal model for statistical learning

- a prediction rule, $h : \mathcal{X} \rightarrow \mathcal{Y}$
 - the learner's output;
 - used to label future examples;
 - called a *predictor*, a *hypothesis*, or a *classifier*;
 - $h = A(S)$: the hypothesis h is returned by a learning algorithm A based on the training sequence S
 - goal of the learner: h should be *correct on future examples*

Example

- prediction rule for tastiness
- $h(x) = 1$ if $x = [\text{color}, \text{shape}]$ within the inner rectangle,
- $h(x) = 0$ if $x = [\text{color}, \text{shape}]$ outside the inner rectangle.



“Correct on future examples”

- let f be the correct classifier, then we should find h such that $h \approx f$
- one way: define the error of h w.r.t. f to be the probability that it does not predict the correct label on a random data point x generated by the underlying (unknown) probability distribution \mathcal{D} over \mathcal{X} :

$$L_{\mathcal{D},f}(h) = \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq f(x)]$$

- more formally, \mathcal{D} is a probability distribution over \mathcal{X} , that is, for a given $A \subset \mathcal{X}$, the value of $\mathcal{D}(A)$ is the probability to observe some $x \in A$. Then:

$$L_{\mathcal{D},f}(h) \stackrel{\text{def}}{=} \mathbb{P}_{x \sim \mathcal{D}}[h(x) \neq f(x)] \stackrel{\text{def}}{=} \mathcal{D}(\{x \in \mathcal{X} : h(x) \neq f(x)\})$$

- can we find h s.t. $L_{\mathcal{D},f}(h)$ is small ?
 - $L_{\mathcal{D},f}(h)$ is called **generalization error**, the **true error** of h , the **real risk** of h
 - L - loss of the learner

Data-generation model

- we must assume some relation between the training data S and \mathcal{D} , f
- simple data generation model:
 - **Independently Identically Distributed (i.i.d.)**: Each x_i is sampled independently according to \mathcal{D} .
 - we do not assume that the learner knows anything about \mathcal{D} .
 - **Realizability**: assume that there is a “correct” labeling function, $f: \mathcal{X} \rightarrow \mathcal{Y}$ and that for every $i \in 1, 2, \dots, m$, $y_i = f(x_i)$
 - f is a deterministic labeling – strong assumption
 - relax this assumption later (make f probabilistic labeling)
 - we do not assume that the learner knows anything about f .
- each pair in the training data S is generated by first sampling i.i.d. a point x_i according to \mathcal{D} and then labeling it by f .

Empirical Risk Minimization

The ERM learning paradigm

- $h_S = A(S)$: the hypothesis h_S is returned by a learning algorithm A based on the training sequence S , sampled i.i.d from an unknown distribution \mathcal{D} and labeled by some target function f
- the true error $L_{\mathcal{D},f}(h)$ is unknown to the learner as he doesn't know \mathcal{D} and f
- the learner has access to the *training error* = *empirical error* = *empirical risk*:

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m},$$

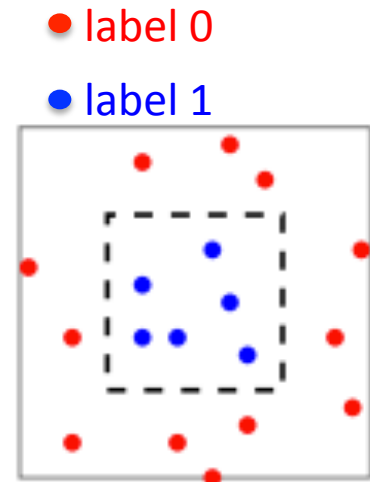
- search for a solution that works well on the training data – minimize $L_S(h)$
- *Empirical Risk Minimization (ERM)* = learning paradigm that returns a predictor h that minimizes $L_S(h)$

ERM might overfit

- simple rule for finding h with small empirical error:

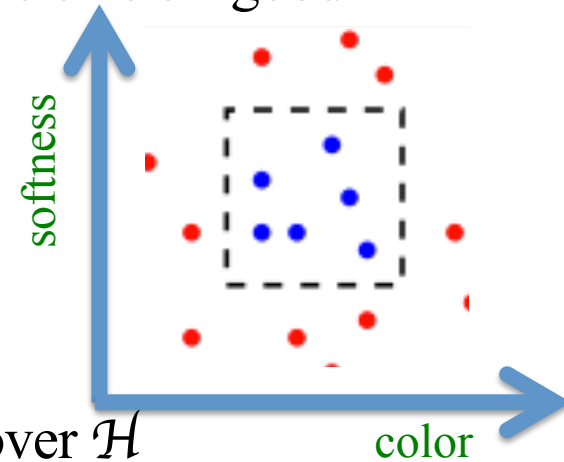
$$h_S(x) = \begin{cases} y_i & \text{if } \exists i \in [m] \text{ s.t. } x_i = x \\ 0 & \text{otherwise.} \end{cases}$$

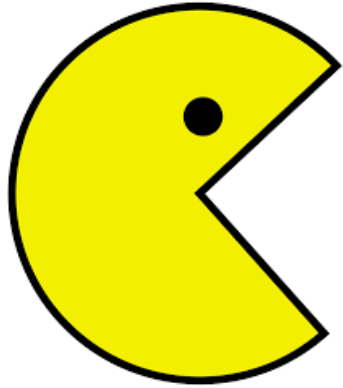
- consider:
 - \mathcal{D} uniform over the gray rectangle
 - all instances in the inner rectangle are labeled 1 (**blue points**)
 - all other instances in the gray rectangle but outside the inner one are labeled 0 (**red points**)
 - area of the gray square is 2, area of the inner rectangle is 1
- we obtain that $L_S(h_S) = 0$, but $L_{\mathcal{D},f}(h_S) = 1/2$ (h predicts the label 1 on a finite number of instances)
- *overfitting*: small training error, large true error
- while the above predictor h_S seems to be very unnatural, it can be described as a thresholded polynomial (seminar exercise)



ERM with Inductive Bias

- to guard against overfitting we introduce some prior knowledge (inductive bias)
- example for the papaya prediction tasting problem: there exists a good prediction rule that is some axis aligned rectangle
- restricted search space: set of all rectangles
- hypothesis class = $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}} = \{g : \mathcal{X} \rightarrow \mathcal{Y}\}$
- revised ERM rule: apply the ERM learning paradigm over \mathcal{H}
 - for the training sample S , the $\text{ERM}_{\mathcal{H}}$ learner chooses a predictor $h \in \mathcal{H}$ with the lowest possible error over S :
$$\text{ERM}_{\mathcal{H}}(S) \in \underset{h \in \mathcal{H}}{\operatorname{argmin}} L_S(h), \quad (\text{Regression, SGD, etc})$$
- over which hypothesis classes $\text{ERM}_{\mathcal{H}}$ will not result in overfitting?
 - what characterizes a good hypothesis class?





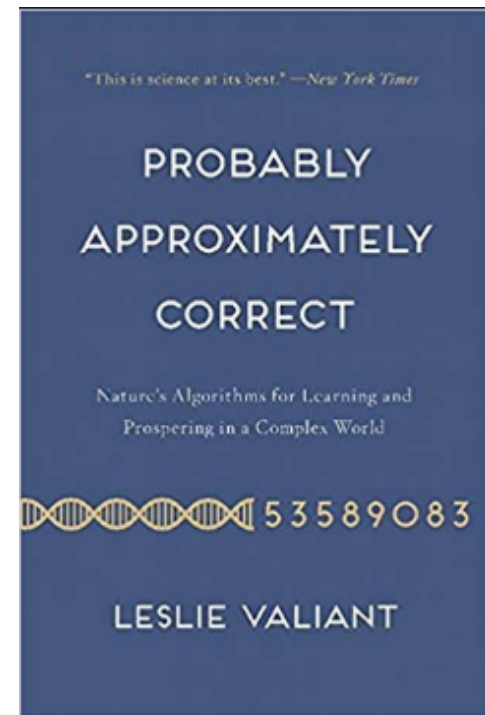
PAC learning

(Probably Approximately Correct learning)

PAC learning

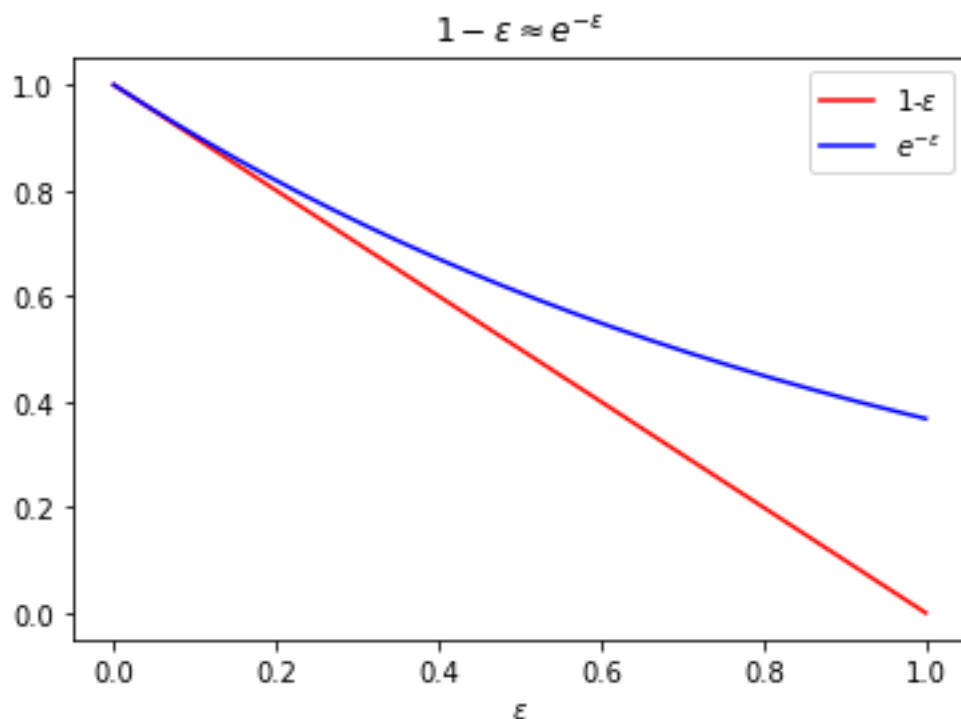
Leslie Valiant, Turing award 2010

*For transformative contributions to the theory of computation, including **the theory of probably approximately correct (PAC) learning**, the complexity of enumeration and of algebraic computation, and the theory of parallel and distributed computing.*



Can only be *Approximately* correct

- **claim:** we can't hope to find h_S s.t. $L_{(\mathcal{D},f)}(h_S) = 0$
- **proof:**
 - for every $\varepsilon \in (0,1)$ take $\mathcal{X} = \{x_1, x_2\}$ and $\mathcal{D}(\{x_1\}) = 1 - \varepsilon$, $\mathcal{D}(\{x_2\}) = \varepsilon$
 - the probability not to see x_2 at all among m i.i.d. examples from S is $(1 - \varepsilon)^m \approx e^{-\varepsilon m}$



Can only be *Approximately* correct

- **claim:** we can't hope to find h_S s.t. $L_{(\mathcal{D},f)}(h_S) = 0$
- **proof:**
 - for every $\varepsilon \in (0,1)$ take $\mathcal{X} = \{x_1, x_2\}$ and $\mathcal{D}(\{x_1\}) = 1 - \varepsilon$, $\mathcal{D}(\{x_2\}) = \varepsilon$
 - the probability not to see x_2 at all among m i.i.d. examples from S is $(1 - \varepsilon)^m \approx e^{-\varepsilon m}$
- so, if $\varepsilon \ll 1/m$ we're likely not to see x_2 at all, but then we can't know its label
- **relaxation:** we'd be happy with $L_{(\mathcal{D},f)}(h_S) \leq \varepsilon$, where ε is a parameter user-specified
- ε is the accuracy parameter
- ε measures the quality of our prediction

Can only be *Probably* correct

- recall that the input to the learner is randomly generated
- there's always a (very small) chance to see the same example again and again
 - all the papayas we have happened to taste were not tasty,
 - we will come up with the classifier that assigns label 0 = Not Tasty to every future sample
 - will lead to large true error
- **claim:** no algorithm can guarantee $L_{(\mathcal{D},f)}(h_S) \leq \epsilon$ for sure
 - address the probability to sample a training set S for which $L_{(\mathcal{D},f)}(h_S) \leq \epsilon$
- **relaxation:** we'd allow the algorithm to fail with probability δ , where $\delta \in (0, 1)$ is user-specified
- here, the probability δ is over the random choice of examples
- $1 - \delta$ is the confidence parameter
- δ measures the probability of getting a nonrepresentative sample

Probably **A**pproximately **C**orrect (**PAC**) learning

- the learner doesn't know \mathcal{D} and f
- the learner receives accuracy parameter ϵ and confidence parameter δ
- the learner can ask for training data, S , containing $m(\epsilon, \delta)$ examples (that is, the number of examples can depend on the value of ϵ and δ , but it can't depend on \mathcal{D} or f).
- learner should output a hypothesis h_S s.t. with probability of at least $1 - \delta$ it holds that $L_{\mathcal{D},f}(h_S) \leq \epsilon$
- that is, the learner should be **Probably** (with probability at least $1 - \delta$) **Approximately** (up to accuracy ϵ) **Correct**

Learning finite classes

- give more knowledge to the learner: the target f comes from some **hypothesis class**, $\mathcal{H} \subset \mathcal{Y}^{\mathcal{X}}$
- the learner knows \mathcal{H}
- assume that \mathcal{H} is a finite hypothesis class
 - example: \mathcal{H} is all the functions from \mathcal{X} to \mathcal{Y} that can be implemented using a Python program of length at most b
- use the **consistent** learning rule:
 - input: \mathcal{H} and $S = (x_1, y_1), \dots, (x_m, y_m)$
 - output: $h \in \mathcal{H}$, h is an ERM hypothesis
- **Empirical Risk Minimization (ERM)**
 - input: training set $S = (x_1, y_1), \dots, (x_m, y_m)$
 - define the empirical risk $L_S(h)$:
$$L_S(h) = \frac{|\{i \in [m] : h(x_i) \neq y_i\}|}{m},$$
 - output: any $h \in \mathcal{H}$ that minimizes $L_S(h)$

The Realizability Assumption

- assume that there exists $h^* \in \mathcal{H}$ such that $L_{\mathcal{D},f}(h^*) = 0$.
 - implies that with probability 1 over random samples, S , where the instances of S are sampled according to \mathcal{D} and are labeled by f , we have $L_S(h^*) = 0$.
 - implies that for every ERM hypothesis h_S we have that $L_S(h_S) = 0$
 - h_S has minimum empirical risk, but we already know that h^* has empirical risk equal to 0
- also known as the consistency assumption
- strong assumption
 - relax the assumption later

Learning finite classes

Theorem:

Let \mathcal{H} be a finite hypothesis class. Let $\delta \in (0, 1)$ and $\epsilon > 0$ and let m be an integer that satisfies:

$$m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$$

Then, for any labeling function f , and for any distribution \mathcal{D} , for which the realizability assumption holds (that is, for some $h^* \in \mathcal{H}$, $L_{\mathcal{D},f}(h^*) = 0$) with probability of least $1 - \delta$ over the choice of an i.i.d, sample S of size m , we have that for every ERM hypothesis h_S it holds that: $L_{(\mathcal{D},f)}(h_S) \leq \epsilon$.

$$P_{S \sim \mathcal{D}^m}(L_{f,D}(h_S) \leq \epsilon) \geq 1 - \delta$$

*The theorem basically says that for a sufficient large number of training examples m the $ERM_{\mathcal{H}}$ rule over a finite hypothesis class is **P**robably **A**pproximately **C**orrect*

Learning finite classes

*The theorem basically says that for a sufficient large number of training examples m the $ERM_{\mathcal{H}}$ rule over a finite hypothesis class is **P**robably **A**pproximately **C**orrect*

$$P_{S \sim D^m} (L_{f,D}(h_S) \leq \epsilon) \geq 1 - \delta$$

Idea of the proof

- a bad predictor h_b has $L_{D,f}(h_b) > \epsilon$
- h_b can be output by the $ERM_{\mathcal{H}}$ learning paradigm if has zero empirical error: $L_S(h_b) = 0$
- this can happen if h_b labels correctly all the m training examples from S i.i.d from \mathcal{D}
- given a random example from \mathcal{D} , h_b has $< 1-\epsilon$ probability to label it correctly
- h_b labels correctly all the m training examples from S with probability $< (1-\epsilon)^m \leq e^{-\epsilon m}$
- there are at most $|\mathcal{H}|$ bad hypothesis, so consider $|\mathcal{H}| \times e^{-\epsilon m} \leq \delta$, so take $m \geq \frac{\log(|\mathcal{H}|/\delta)}{\epsilon}$

PAC learnability of a class \mathcal{H}

A hypothesis class \mathcal{H} is called *PAC learnable* if:

I can find a hypothesis h from \mathcal{H} based on the learning algorithm A with

- whatever accuracy $\varepsilon > 0$ I want
- whatever confidence $\delta > 0$ I want
- whatever the distribution \mathcal{D} is
- whatever the labeling function f is

given that I provide to A enough samples $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ drawn from \mathcal{D}

$$P_{S \sim D^m} (L_{f,D}(h_S) \leq \varepsilon) \geq 1 - \delta$$

PAC learnability of a class \mathcal{H} -definition

A hypothesis class \mathcal{H} is called **PAC learnable** if there exists a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm A with the following property:

- for every $\varepsilon > 0$ (*accuracy* \rightarrow “approximately correct”)
- for every $\delta > 0$ (*confidence* \rightarrow “probably”)
- for every labeling $f \in \mathcal{H}$ (*realizability case*)
- for every distribution \mathcal{D} over \mathcal{X}

when we run the learning algorithm A on a training set S , consisting of $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ examples sampled i.i.d. from \mathcal{D} and labeled by f the algorithm A returns a hypothesis $h_S \in \mathcal{H}$ such that, with probability at least $1-\delta$ (over the choice of examples), $L_{\mathcal{D},f}(h_S) \leq \varepsilon$.

PAC learnability of a class \mathcal{H} -definition

A hypothesis class \mathcal{H} is called **PAC learnable** if there exists a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm A with the following property:

- for every $\varepsilon > 0$ (*accuracy* \rightarrow “approximately correct”)
- for every $\delta > 0$ (*confidence* \rightarrow “probably”)
- for every labeling $f \in \mathcal{H}$ (*realizability case*)
- for every distribution \mathcal{D} over \mathcal{X}

when we run the learning algorithm A on a training set S , consisting of $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ examples sampled i.i.d. from \mathcal{D} and labeled by f the algorithm A returns a hypothesis $h_S \in \mathcal{H}$ such that, with probability at least $1-\delta$ (over the choice of examples), $L_{\mathcal{D},f}(h_S) \leq \varepsilon$.

$$P_{S \sim \mathcal{D}^m} (L_{f,D}(h_S) \leq \varepsilon) \geq 1 - \delta$$

- $h_S = A(S)$
- the function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ is called sample complexity of learning \mathcal{H}
- $m_{\mathcal{H}}(\varepsilon, \delta)$ – the minimum number of examples required to guarantee a PAC solution

PAC learnability of a class \mathcal{H} -definition

A hypothesis class \mathcal{H} is called **PAC learnable** if there exists a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm A with the following property:

- for every $\varepsilon > 0$ (*accuracy* \rightarrow “approximately correct”)
- for every $\delta > 0$ (*confidence* \rightarrow “probably”)
- for every labeling $f \in \mathcal{H}$ (*realizability case*)
- for every distribution \mathcal{D} over \mathcal{X}

when we run the learning algorithm A on a training set S , consisting of $m \geq m_{\mathcal{H}}(\varepsilon, \delta)$ examples sampled i.i.d. from \mathcal{D} and labeled by f the algorithm A returns a hypothesis $h_S \in \mathcal{H}$ such that, with probability at least $1-\delta$ (over the choice of examples), $L_{\mathcal{D},f}(h_S) \leq \varepsilon$.

$$P_{S \sim D^m}(L_{f,D}(h_S) \leq \varepsilon) \geq 1 - \delta \Leftrightarrow P_{S \sim D^m}(L_{f,D}(h_S) > \varepsilon) < \delta$$

Next time

| | | |
|----------|--|----|
| 3 | A Formal Learning Model | 22 |
| 3.1 | PAC Learning | 22 |
| 3.2 | A More General Learning Model | 23 |
| 3.3 | Summary | 28 |
| 3.4 | Bibliographic Remarks | 28 |
| 3.5 | Exercises | 28 |
| 4 | Learning via Uniform Convergence | 31 |
| 4.1 | Uniform Convergence Is Sufficient for Learnability | 31 |
| 4.2 | Finite Classes Are Agnostic PAC Learnable | 32 |
| 4.3 | Summary | 34 |
| 4.4 | Bibliographic Remarks | 35 |
| 4.5 | Exercises | 35 |