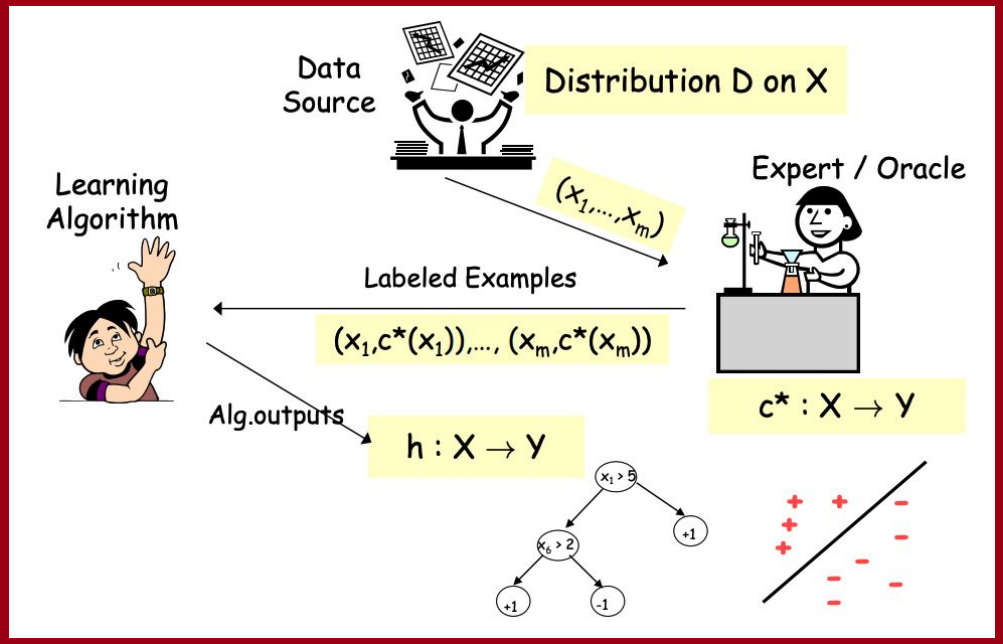
DIPARTIMENTO
DI INGEGNERIA
DELL'INFORMAZIONE

UNIVERSITÀ
DEGLI STUDI
DI PADOVA



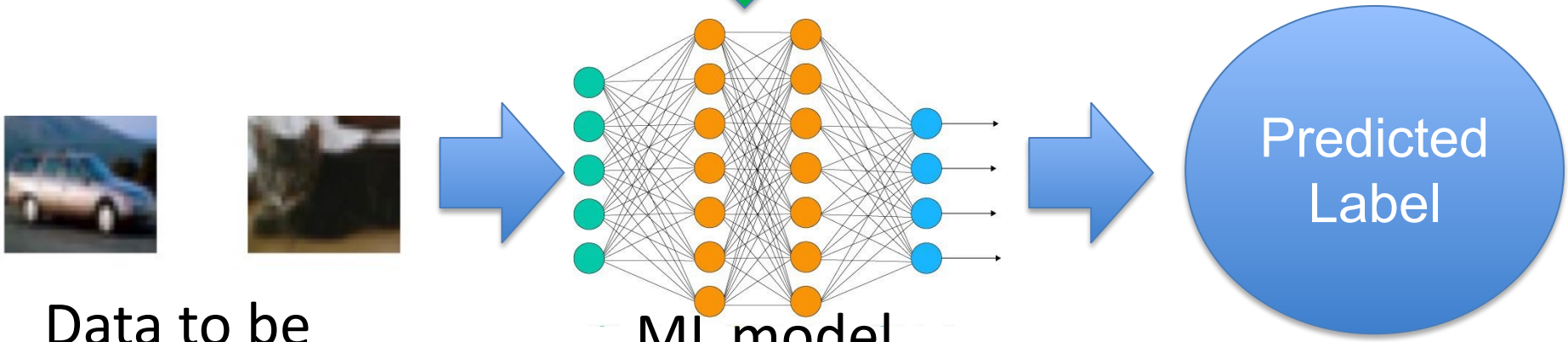
PAC Learning

Machine Learning 2022/23
UML book chapter 2
Slides: F. Chiariotti, P. Zanuttigh, F. Vandin

Supervised Learning

Training data
with labels

Training procedure



Data to be
analyzed

ML model
(training: estimate parameters)

In most of the course we will focus on supervised learning

PAC Learning

Probably Approximately Correct (PAC) learning

Since the training data is sampled accordingly to D:

- we can only be approximately correct
- we can only be probably correct

Parameters:

- accuracy parameter ϵ : we are satisfied with a good h_s for which $L_{D,f}(h_s) \leq \epsilon$
- confidence parameter δ: want h_s to be a good hypothesis

Theorem

(Finite Hypothesis Classes are PAC Learnable)

Let H be a finite hypothesis class. Let $\delta \in (0,1)$, $\epsilon \in (0,1)$ and $m \in \mathbb{N}$ such that:

$$m \geq \frac{\log\left(\frac{|H|}{\delta}\right)}{\epsilon}$$

Then for any f and any D for which the realizability assumption holds, with probability $\boxed{\geq 1-\delta}$ we have that for every ERM hypothesis h_s it holds that

$$L_{D,f}(h_s) \leq \boxed{\epsilon}$$

probably	approximately	correct

m: size of the training set (i.e., S contains m I.I.D. samples)

Idea of the Proof

- The critical issue are the training sets leading to a "misleading" predictor h with $L_S(h) = 0$ but $L_{D,f}(h) > \epsilon$
- Place an upper bound to the probability of sampling *m* instances leading to a *misleading training set,* i.e., producing a "misleading" predictor
- Using the union bound after various mathematical computations the bound of the theorem can be obtained
- *Message of the theorem*: if $\mathcal{H}$ is a finite class then ERM will not overfit, provided it is computed on a sufficiently big training set
- *Demonstration not part of the course, but you can find it on the book if you are interested*

Theorem: Graphical Illustration

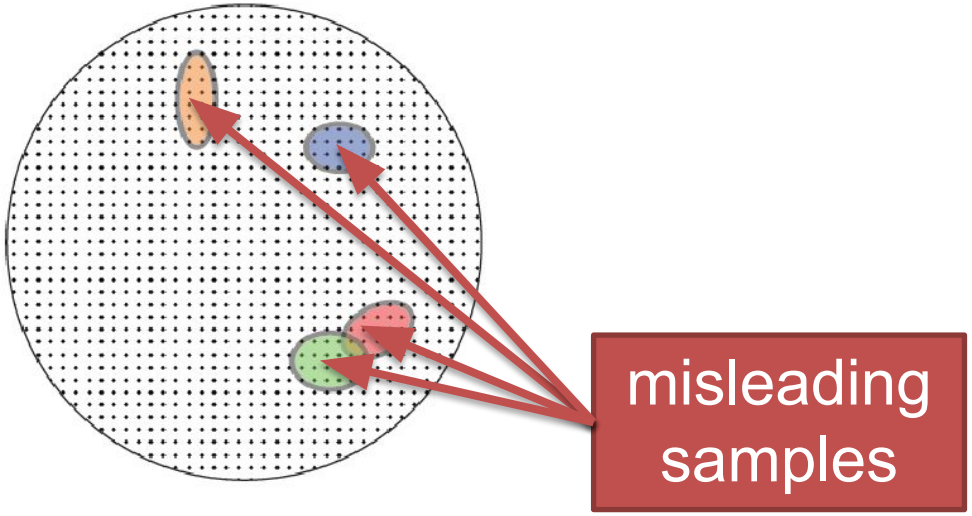


Figure 2.1 Each point in the large circle represents a possible m-tuple of instances. Each colored oval represents the set of "misleading" m-tuple of instances for some "bad" predictor $h \in \mathcal{H}_B$. The ERM can potentially overfit whenever it gets a misleading training set S. That is, for some $h \in \mathcal{H}_B$ we have $L_S(h) = 0$. Equation (2.9) guarantees that for each individual bad hypothesis, $h \in \mathcal{H}_B$, at most $(1-\epsilon)^m$-fraction of the training sets would be misleading. In particular, the larger m is, the smaller each of these colored ovals becomes. The union bound formalizes the fact that the area representing the training sets that are misleading with respect to some $h \in \mathcal{H}_B$ (that is, the training sets in M) is at most the sum of the areas of the colored ovals. Therefore, it is bounded by $|\mathcal{H}_B|$ times the maximum size of a colored oval. Any sample S outside the colored ovals cannot cause the ERM rule to overfit.

PAC Learnability

A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm such that for every $\delta, \epsilon \in (0,1)$, for every distribution D over $\mathcal{X}$ and for every labeling function $f: x \rightarrow \{0,1\}$, if the realizability assumption holds with respect to $\mathcal{H}, D, f$ when running the algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by D and labeled by f the algorithm returns a hypothesis h such that, with probability $\geq 1 - \delta$ (over the choice of the m training examples): $L_{D,f}(h) \leq \epsilon$

Notes:

- It is a property of the hypothesis class
- Must be satisfied for every δ, ϵ, D, f
- $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$: sample complexity of learning $\mathcal{H}$ (depends on δ and ϵ)
- $m_{\mathcal{H}}$ is the minimum integer that satisfies the requirements
- *Sample complexity*: minimum size of training set to be sure to satisfy requirements (sufficient but not necessary condition)
- Only if realizability assumption holds
- PAC: Good probability ($1 - \delta$) of having a good predictor ($L_{D,f}(h) \leq \epsilon$)

Corollary (PAC)

A hypothesis class $\mathcal{H}$ is PAC learnable if there exist a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm such that for every $\delta, \epsilon \in (0,1)$, for every distribution D over $\mathcal{X}$ and for every labeling function $f: x \rightarrow \{0,1\}$, if the realizability assumption holds with respect to $\mathcal{H}, D, f$ when running the algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by D and labeled by f the algorithm returns a hypothesis h such that, with probability $\geq 1 - \delta$ (over the choice of the m training examples): $L_{D,f}(h) \leq \epsilon$

Let H be a finite hypothesis class. Let $\delta \in (0,1)$, $\epsilon \in (0,1)$ and $m \in \mathbb{N}$ such that:

$$m \geq \frac{\log\left(\frac{|H|}{\delta}\right)}{\epsilon}$$

Then for any f and any D for which the realizability assumption holds, with probability $\boxed{\geq 1 - \delta}$ we have that for every ERM hypothesis h_s it holds that

$$L_{D,f}(h_s) \leq \boxed{\epsilon}$$

probably | approximately | correct

From the previous theorem and from the definition of PAC:

Corollary (PAC)

Every finite hypothesis class is PAC learnable with sample complexity

$$m_H(\epsilon, \delta) \boxed{\leq} \left\lceil \frac{\log\left(\frac{|H|}{\delta}\right)}{\epsilon} \right\rceil$$

sufficient condition (not necessary)

- Larger hypothesis class $|H|$ →need a larger training set
- Smaller training error ϵ →need a larger training set
- Better probability of a good solution $1 - \delta$ →need a larger training set

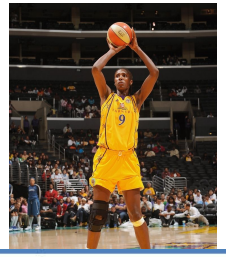
Drop Some Assumptions

1. *Realizability Assumption*: there exists $h^* \in \mathcal{H}$ such that

$$L_{D,f}(h) = 0$$

   - ❑ Too strong in many real-world applications!

2. *Function f* : in many applications it is not too realistic that the *labeling is fully determined* by the features we measure
   - ❑ *Relaxation*: replace target labeling function…
   - ❑ .. and assume *D* is a probability distribution over $\mathcal{X} x \mathcal{Y}$
     - i.e., *D* is the joint distribution over domain points and labels
   - ❑ For example, two components of *D*:
     - D_x : (marginal) distribution over domain points
     - *D((x, y)|x)*: conditional distribution over labels for each domain point



$(w, h) \rightarrow M \text{ or } F \text{ ??}$

3. *Binary classification* → move to a more general setting
   - ❑ Multi-class classification and regression problems

Empirical and True Error

Assuming *D* is a probability distribution over $\mathcal{X} \times \mathcal{Y}$, the *true error* (or risk) is:

$$L_D(h) \triangleq \mathbb{P}_{(x,y)\sim D}[h(x) \neq y] \stackrel{\text{def}}{=} D(\{(x,y): h(x) \neq y\})$$

Recall: previously $\mathbb{P}_{x\sim D}\,[h(x) \neq f(x)]$

As before *D* is not known to the ML algorithm (learner): the learner only knows the training data S

The *Empirical Risk* is as before:

$$L_S(h) \stackrel{\text{def}}{=} \frac{|\{i: h(x_i) \neq y_i \ , 1 \leq i \leq m\}|}{m}$$

Note: $L_S(h)$ = probability for a pair taken uniformly (x_i, y_i) at random from S the event $h(x_i) \neq y_i$ holds

Bayes Optimal Predictor

Learner's goal: find $h: \mathcal{X} \to \mathcal{Y}$ minimizing $L_D(h)$

Question: Is there a best predictor ?

Given a probability distribution D over $\mathcal{X} x \{0,1\}$, the best predictor is the *Bayes Optimal Predictor:*

binary classification

$$f_D(x) = \begin{cases} 1 & if\ \mathbb{P}[y = 1\ |x\ ] \geq \frac{1}{2} \\ 0 & otherwise \end{cases}$$

Proposition:

For any classifier $g: \mathcal{X} \to \{0,1\}$, it holds $L_D(f_D) \leq L_D(g)$

Can we use the predictor?

D (and consequently P[y=1|x]) is not known!

Agnostic PAC Learnability

- *Idea*: We drop the requirement of finding the best predictor, but we do not want to be too far from it
- Definition (agnostic PAC learnability):

A hypothesis class $\mathcal{H}$ is agnostic PAC learnable if there exist a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm such that for every $\delta, \epsilon \in (0,1)$ and for every distribution D over $\mathcal{X} x \mathcal{Y}$, when running the algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by D the algorithm returns a hypothesis h such that, with probability $\geq 1 - \delta$ (over the choice of the m training examples):

$$L_D(h) \leq \min_{h' \in \mathcal{H}} L_D(h') + \epsilon$$

Not farther than ϵ from the best predictor in $\mathcal{H}$

- This is a generalization of the previous learning model
- Realizability is not required

Multiclass Classification and Regression

We consider 3 possible learning problems:

- Domain set and training data have the same structure for all problems
- Learner output: h: $\mathcal{X} \rightarrow \mathcal{Y}$ ($\mathcal{Y}$ *is different for the 3 problems*)

1. *Binary classification*: *(it is the one we were considering before)*
   - Target set: $\mathcal{Y} = \{0,1\}$ (target set size = *2*)

2. *Multiclass classification* with *K* > 2 classes
   - Target set: $\mathcal{Y} = \{0,1,\dots,K-1\}$ (target set size = *K*)
   - Loss: as for the binary case

3. Regression $\mathcal{Y} = \mathbb{R}$
   - Target set has an infinite size
   - Need a new loss function !

10M€

300k€

50k€

Generalized Loss Function

Given:

- $\mathcal{H}$: hypothesis class
- Z : domain ($\mathcal{X}x\mathcal{Y}$)

A loss function is any function $l\colon \mathcal{H}xZ \to \mathbb{R}_+$

Risk function: expected loss of an hypothesis $h \in \mathcal{H}$ with respect to D over Z:

$$L_D(h) \stackrel{\text{def}}{=} \mathbb{E}_{z\sim D}[l(h,z)]$$

Empirical risk: expected loss over a given training set $S = (z_1, \dots, z_m) \in Z^m$:

$$L_s(h) \stackrel{\text{def}}{=} \frac{1}{m}\sum_{i=1}^{m} l(h, z_i)$$

Common Loss Functions

- *0-1 loss*: Commonly used in binary or multiclass classification

$$l_{0-1}(h,(x,y)) \stackrel{\text{def}}{=} \begin{cases} 0 & if\ h(x) = y \\ 1 & if\ h(x) \neq y \end{cases}$$

- *Cross Entropy* is also used in classification (presented later in the course)
- *Squared loss* (*L2*): Commonly used in regression, penalize few large errors

$$l_{sq}(h,(x,y)) \stackrel{\text{def}}{=} (h(x) - y)^2$$

- *Absolute value loss* (*L1*): Commonly used in regression, penalize many small errors

$$l_{abs}(h,(x,y)) \stackrel{\text{def}}{=} |h(x) - y|$$

- In general, the loss function depends on the application!
- But computational considerations must also be taken into account..

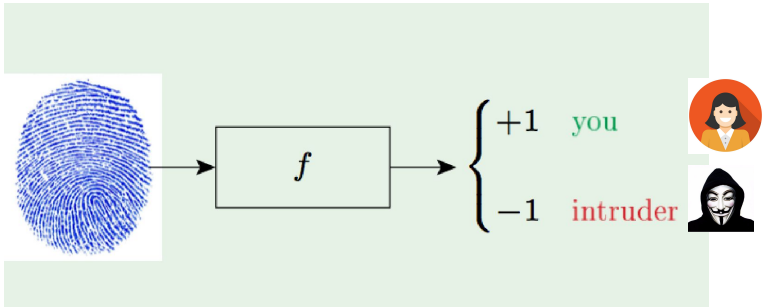
Optimal Loss Depends on Application

The relevance of different error types depends on the application

Example: fingerprints classification/verification

Two types of error:

- *False accept*: accept an unauthorized user
- *False reject*: do not accept an authorized user



		f	
		$+1$	-1
h	$+1$	no error	false accept
	-1	false reject	no error

Optimal Loss: Discount Verification

The error measure - for supermarkets

Supermarket verifies fingerprint for discounts

False reject is costly; customer gets annoyed!

False accept is minor; gave away a discount and intruder left their fingerprint ☺

$$\rightarrow \boxed{f} \rightarrow \begin{cases} +1 & \text{you} \\ -1 & \text{intruder} \end{cases}$$

		f	
		$+1$	-1
h	$+1$	0	1
	-1	10	0

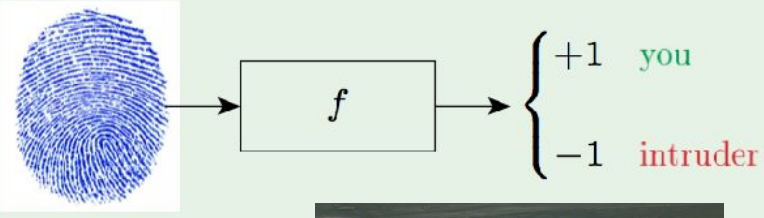
Optimal Loss: CIA Data Center

The error measure - for the CIA

CIA verifies fingerprint for security

False accept is a disaster!

False reject can be tolerated
Try again; you are an employee ☺



		f	
		$+1$	-1
h	$+1$	0	1000
	-1	1	0

Agnostic PAC Learnability: General Loss Functions

Definition

A hypothesis class $\mathcal{H}$ is agnostic PAC learnable with respect to a set Z and a loss function $l: HxZ \rightarrow \mathbb{R}_+$ if there exist a function $m_{\mathcal{H}}: (0,1)^2 \rightarrow \mathbb{N}$ and a learning algorithm such that for every $\delta, \epsilon \in (0,1)$ and for every distribution D over Z, when running the algorithm on $m \geq m_{\mathcal{H}}(\epsilon, \delta)$ i.i.d. examples generated by D the algorithm returns a hypothesis h such that, with probability $\geq 1 - \delta$ (over the choice of the m training examples):

$$L_D(h) \leq \min_{h' \in \mathcal{H}} L_D(h') + \epsilon$$

where $L_d(h) = \mathbb{E}_{z \sim D}[l(h, z)]$