Il Dizionario A”alto del Machine Learning

Alexander Jung and Konstantina Olioumtsevits

27 maggio 2025

please cite as: A. Jung and K. Olioumtsevits, *The Aalto Dictionary of Machine Learning*. Espoo, Finland: Aalto University, 2025.

Riconoscimenti

Questo dizionario di machine learning si è evoluto nel corso dello sviluppo e dell'insegnamento di diversi corsi, tra cui CS-E3210 Machine Learning: Principi di base, CS-C3240 Machine Learning, CS-E4800 Artificial Intelligence, CS-EJ3211 Machine Learning with Python, CS-EJ3311 Deep Learning with Python, CS-E4740 Federated Learning e CS-E407507 Human-Centered Machine Learning. Tali corsi sono stati erogati presso l'Aalto University `https://www.aalto.fi/en`, a studenti adulti tramite il Finnish Institute of Technology (FITech) `https://fitech.io/en/`, e a studenti internazionali attraverso l'European University Alliance Unite! `https://www.aalto.fi/en/unite`.

Siamo grati agli studenti che hanno fornito preziosi feedback, contribuendo a plasmare questo dizionario. Un ringraziamento speciale va a Mikko Seesto per la sua meticolosa revisione.

Lists of Symbols

Insiemi e Funzioni

$a \in \mathcal{A}$	L'oggetto a è un elemento dell'insieme $\mathcal{A}$.
$a := b$	Usiamo a come abbreviazione di b.
$\lvert\mathcal{A}\rvert$	La cardinalità (cioè, il numero di elementi) di un insieme finito $\mathcal{A}$.
$\mathcal{A} \subseteq \mathcal{B}$	$\mathcal{A}$ è un sottoinsieme di $\mathcal{B}$.
$\mathcal{A} \subset \mathcal{B}$	$\mathcal{A}$ è un sottoinsieme proprio di $\mathcal{B}$.
$\mathbb{N}$	I numeri naturali $1, 2, \ldots$.
$\mathbb{R}$	I numeri reali x [**?**].
$\mathbb{R}_+$	I numer reali non negativi $x \geq 0$.
$\mathbb{R}_{++}$	I numeri reali strettamente positivi $x > 0$.

$\{0, 1\}$	L'insieme costituito dai due numeri reali 0 e 1.
$[0, 1]$	L'intervallo chiuso dei numeri reali x tali che $0 \leq x \leq 1$.
$\operatorname{argmin}_{\mathbf{w}} f(\mathbf{w})$	L'insieme dei minimizzatori di una funzione reale $f(\mathbf{w})$.
$\mathbb{S}^{(n)}$	L'insieme dei vettori a norma unitaria in $\mathbb{R}^{n+1}$.
$\log a$	I logaritmo del numero reale strettamente positivo $a \in \mathbb{R}_{++}$.
$h(\cdot) : \mathcal{A} \to \mathcal{B} : a \mapsto h(a)$	Una funzione (mappa) che accetta come input un qualsiasi elemento $a \in \mathcal{A}$, appartenente ad un insieme $\mathcal{A}$, e restituisce un elemento ben definito $h(a) \in \mathcal{B}$, appartenente ad un insieme $\mathcal{B}$. L'insieme $\mathcal{A}$ è detto dominio della funzione h, mentre l'insieme $\mathcal{B}$ è il codominio di h. Machine learning (ML) punta ad individuare (o apprendere) una funzione h (cioè, un ipotesi) che, a partire dalle caratteristiche $\mathbf{x}$ di un punto dati, fornisca una previsione $h(\mathbf{x})$ della sua etichetta y.
$\nabla f(\mathbf{w})$	Il gradiente di una funzione reale differenziabile $f : \mathbb{R}^d \to \mathbb{R}$ è il vettore $\nabla f(\mathbf{w}) = \left(\frac{\partial f}{\partial w_1}, \ldots, \frac{\partial f}{\partial w_d}\right)^T \in \mathbb{R}^d$ [**?**, Ch. 9].

Matrici e Vettori

$\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$	Un vettore di lunghezza d, il cui j-esimo elemento è x_j.
$\mathbb{R}^d$	L'insieme di vettori $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ composto da d elementi a valori reali $x_1, \ldots, x_d \in \mathbb{R}$.
$\mathbf{I}_{l \times d}$	Una matrice identità generalizzata con l righe e d colonne. Gli elementi di $\mathbf{I}_{l \times d} \in \mathbb{R}^{l \times d}$ sono pari a 1 lungo la diagonale principale e pari a 0 altrove.
$\mathbf{I}_d, \mathbf{I}$	Una matrice identità quadrata di dimensione $d \times d$. Se la dimensione è chiara dal contesto, si omette l'indice.
$\|\mathbf{x}\|_2$	La norma Euclidea (or ℓ_2) del vettore $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T \in \mathbb{R}^d$ definita come $\|\mathbf{x}\|_2 := \sqrt{\sum_{j=1}^d x_j^2}$.
$\|\mathbf{x}\|$	Una qualche norma del vettore $\mathbf{x} \in \mathbb{R}^d$ [**?**]. Salvo diversa specificazione, intendiamo la norma Euclidea $\|\mathbf{x}\|_2$.
$\mathbf{x}^T$	La trasposizione di una matrice che ha il vettore $\mathbf{x} \in \mathbb{R}^d$ come unica colonna.
$\mathbf{X}^T$	La trasposizione di una matrice $\mathbf{X} \in \mathbb{R}^{m \times d}$. Una matrice quadrata a valori reali $\mathbf{X} \in \mathbb{R}^{m \times m}$ è detta simmetrica se $\mathbf{X} = \mathbf{X}^T$.
$\mathbf{0} = \big(0, \ldots, 0\big)^T$	Il vettore in $\mathbb{R}^d$ con ciascuna componente uguale a zero.
$\mathbf{1} = \big(1, \ldots, 1\big)^T$	Il vettore in $\mathbb{R}^d$ con ciascuna componente uguale a uno.

$\left(\mathbf{v}^T, \mathbf{w}^T\right)^T$	Il vettore di lunghezza $d + d'$ ottenuto concatenando le componenti del vettore $\mathbf{v} \in \mathbb{R}^d$ con le componenti di $\mathbf{w} \in \mathbb{R}^{d'}$.
$\text{span}\{\mathbf{B}\}$	Lo span di una matrice $\mathbf{B} \in \mathbb{R}^{a \times b}$, che rappresenta il sottospazio generato da tutte le combinazioni lineari delle colonne di $\mathbf{B}$, $\text{span}\{\mathbf{B}\} = \left\{\mathbf{Ba} : \mathbf{a} \in \mathbb{R}^b\right\} \subseteq \mathbb{R}^a$.
$\det(\mathbf{C})$	Il determinante della matrice $\mathbf{C}$.
$\mathbf{A} \otimes \mathbf{B}$	Il prodotto di Kronecker di $\mathbf{A}$ e $\mathbf{B}$ [?].

Teoria della Probabilità

$\mathbb{E}_p\{f(\mathbf{z})\}$	L' valore atteso di una funzione $f(\mathbf{z})$ di una random variable (RV) $\mathbf{z}$ la cui distribuzione di probabilità è $p(\mathbf{z})$. Se la distribuzione di probabilità è chiara dal contesto, scriviamo semplicemente $\mathbb{E}\{f(\mathbf{z})\}$.
$p(\mathbf{x}, y)$	Una distribuzione di probabilità (congiunta) di una RV le cui realizzazioni sono punto dati con caratteristiche $\mathbf{x}$ e etichetta y.
$p(\mathbf{x}\|y)$	Una distribuzione di probabilità condizionata di una RV $\mathbf{x}$, dato il valore di un'altra RV y [**?**, Sec. 3.5].
$p(\mathbf{x}; \mathbf{w})$	Una distribuzione di probabilità parametrizzata di una RV $\mathbf{x}$. La distribuzione di probabilità dipende da un vettore di parametri $\mathbf{w}$. Ad esempio, $p(\mathbf{x}; \mathbf{w})$ potrebbe essere una multivariate normal distribution con il vettore di parametri $\mathbf{w}$ costituito dagli elementi del vettore di mean $\mathbb{E}\{\mathbf{x}\}$ e della covariance matrix $\mathbb{E}\Big\{\big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)\big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)^T\Big\}$.
$\mathcal{N}(\mu, \sigma^2)$	La distribuzione di probabilità di una Gaussian random variable (Gaussian RV) $x \in \mathbb{R}$ con mean (o valore atteso) $\mu = \mathbb{E}\{x\}$ e variance $\sigma^2 = \mathbb{E}\big\{(x - \mu)^2\big\}$.
$\mathcal{N}(\boldsymbol{\mu}, \mathbf{C})$	La multivariate normal distribution di una Gaussian RV vettoriale $\mathbf{x} \in \mathbb{R}^d$ con mean (o valore atteso) $\boldsymbol{\mu} = \mathbb{E}\{\mathbf{x}\}$ e covariance matrix $\mathbf{C} = \mathbb{E}\big\{\big(\mathbf{x} - \boldsymbol{\mu}\big)\big(\mathbf{x} - \boldsymbol{\mu}\big)^T\big\}$.

Machine Learning

r	Un indice $r = 1, 2, \ldots$ che enumera punti dati.
m	Il numero di punti dati presenti in (cioè, la dimensione di) un dataset.
$\mathcal{D}$	Un dataset $\mathcal{D} = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ è un elenco di singoli punti dati $\mathbf{z}^{(r)}$, con $r = 1, \ldots, m$.
d	Il numero di caratteristiche che caratterizza un punto dati.
x_j	La j-esima feature di un punto dati. La prima caratteristica è indicata con x_1, la seconda caratteristica x_2, e così via.
$\mathbf{x}$	Il vettore delle caratteristiche $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ di un punto dati le cui componenti sono le singole caratteristiche di un punto dati.
$\mathcal{X}$	Lo feature space $\mathcal{X}$ è l'insieme di tutti i valori possibili che le caratteristiche $\mathbf{x}$ di un punto dati possono assumere.
$\mathbf{z}$	Invece del simbolo $\mathbf{x}$, talvolta utilizziamo $\mathbf{z}$ come simbolo alternativo per denotare un vettore i cui elementi sono le singole caratteristiche di un punto dati. È necessario ricorrere a due simboli distinti per differenziare tra caratteristiche grezze e quelle apprese [**?**, Ch. 9].
$\mathbf{x}^{(r)}$	Il vettore di caratteristiche relativo al r-esimo punto dati all'interno di un dataset.

$x_j^{(r)}$	La j-esima caratteristica relativa all' r-esimo punto dati all'interno di un dataset.
$\mathcal{B}$	Un mini-batch (o sottoinsieme) di punti dati scelti casualmente.
B	La dimensione di un mini-batch, ovvero il numero di punti dati che esso contiene.
y	L' etichetta (o quantità di interesse) di un punto dati.
$y^{(r)}$	L' etichetta del r-esimo punto dati.
$\left(\mathbf{x}^{(r)}, y^{(r)}\right)$	Le caratteristiche e le etichetta del r-esimo punto dati.
$\mathcal{Y}$	Lo label space $\mathcal{Y}$ di un algoritmo di ML consiste nell'insieme di tutti i possibili valori di etichetta che un punto dati può assumere. Lo label space nominale può essere più ampio rispetto all'insieme dei diversi valori di etichetta effettivamente presenti in un dato dataset (ad esempio, un training set). I problemi (o metodi) di ML che impiegano uno label space numerico, come $\mathcal{Y} = \mathbb{R}$ or $\mathcal{Y} = \mathbb{R}^3$, sono detti problemi (o metodi) di regression. I problemi (o metodi) di ML che utilizzano uno label space discreto, come $\mathcal{Y} = \{0, 1\}$ or $\mathcal{Y} = \{cat, dog, mouse\}$, sono detti problemi (o metodi) di classification.

η	Learning rate (o step size) usato dai gradient-based methods.
$h(\cdot)$	Una funzione ipotesi che, dato in input un vettore di caratteristiche $\mathbf{x}$ relativo ad un punto dati, restituisce una previsione $\hat{y} = h(\mathbf{x})$ della sua etichetta y.
$\mathcal{Y}^{\mathcal{X}}$	Dati due insiemi $\mathcal{X}$ e $\mathcal{Y}$, indichiamo con $\mathcal{Y}^{\mathcal{X}}$ l'insieme di tutte le possibili funzioni ipotesi $h : \mathcal{X} \to \mathcal{Y}$.
$\mathcal{H}$	Uno hypothesis space o modello utilizzato da un algoritmo di ML. Lo hypothesis space consiste in diverse funzioni ipotesi $h : \mathcal{X} \to \mathcal{Y}$, tra le quali il metodo di ML deve scegliere.
$d_{\text{eff}}(\mathcal{H})$	L' effective dimension di uno hypothesis space $\mathcal{H}$.
B^2	Il bias al quadrato di un' ipotesi $\hat{h}$ appresa, generata da un metodo di ML. Il metodo è addestrato su punti dati modellati come realizzazioni di RV. Poiché i dati costituiscono una realizzazione di RV, anche l'ipotesi appresa $\hat{h}$ rappresenta una realizzazione di una RV.
V	La variance dell' ipotesi appresa (o dei suoi parameters) generata da un metodo di ML. Il metodo è addestrato su punti dati modellati come realizzazioni di RV. Poiché i dati costituiscono una realizzazione di RV, anche l'ipotesi appresa $\hat{h}$ rappresenta una realizzazione di una RV.

$L\left((\mathbf{x}, y), h\right)$	La loss alla stima della etichetta y di punto dati mediante l'utilizzo della previsione $\hat{y} = h(\mathbf{x})$. La previsione $\hat{y}$ è ottenuta applicando l'ipotesi $h \in \mathcal{H}$ al vettore delle caratteristiche $\mathbf{x}$ del punto dati in questione.
E_v	L'validation error di un'ipotesi h, ovvero la media delle loss da essa sostenute sull'validation set.
$\widehat{L}\big(h\|\mathcal{D}\big)$	Il empirical risk, o loss media, associata all'ipotesi h su un dataset $\mathcal{D}$.
E_t	L'training error di un'ipotesi h, ovvero la media delle loss da essa sostenute sull'training set.
t	Un indice temporale discreto $t = 0, 1, \ldots$ utilizzato per enumerare eventi sequenziali (o istanti temporali).
t	Un indice che elenca i learning task all'interno di un problema di multitask learning.
α	Un parametro di regularization che controlla il grado di regularization.
$\lambda_j\big(\mathbf{Q}\big)$	Il j-esimo eigenvalue (ordinato in modo crescente o decrescente) di una matrice positive semi-definite (psd) $\mathbf{Q}$. Utilizziamo inoltre la notazione abbreviata λ_j qualora la matrice corrispondente sia chiara dal contesto.
$\sigma(\cdot)$	La activation function utilizzata da un neurone artificiale all'interno di una artificial neural network (ANN).

$\mathcal{R}_{\hat{y}}$	Una decision region all'interno di uno feature space.
$\mathbf{w}$	Un vettore di parametri $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$ di un modello, e.g., i weights di unlinear model o di una ANN.
$h^{(\mathbf{w})}(\cdot)$	Una funzione di ipotesi espressa in termini di model parameters regolabili $w_1, \ldots, w_d$ concatenati nel vettore $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$.
$\phi(\cdot)$	Una feature map $\phi : \mathcal{X} \to \mathcal{X}' : \mathbf{x} \mapsto \mathbf{x}' := \phi\big(\mathbf{x}\big) \in \mathcal{X}'$.
$K\big(\cdot, \cdot\big)$	Dato uno feature space $\mathcal{X}$, un kernel è una mappa $K : \mathcal{X} \times \mathcal{X} \to \mathbb{C}$ psd.

Federated Learning

$\mathcal{G} = (\mathcal{V}, \mathcal{E})$	Un graph non orientato i cui nodi $i \in \mathcal{V}$ rappresentano i devices all'interno di un federated learning network (FL network). Gli spigoli pesati $\mathcal{E}$ rappresentano la connettività tra i device e le similitudini statistiche tra i rispettivi dataset e learning task.
$i \in \mathcal{V}$	Un nodo che rappresenta un device all'interno di un FL network. Il device ha accesso ad un local dataset e può addestrare un local model.
$\mathcal{G}^{(\mathcal{C})}$	Il sottografo di $\mathcal{G}$ indotto dai nodi $\mathcal{C} \subseteq \mathcal{V}$.
$\mathbf{L}^{(\mathcal{G})}$	La Laplacian matrix di un graph $\mathcal{G}$.
$\mathbf{L}^{(\mathcal{C})}$	La Laplacian matrix del graph indotto $\mathcal{G}^{(\mathcal{C})}$.
$\mathcal{N}^{(i)}$	L'neighborhood di un nodo i in un graph $\mathcal{G}$.
$d^{(i)}$	Il grado pesato $d^{(i)} := \sum_{i' \in \mathcal{N}^{(i)}} A_{i,i'}$ di un nodo i in un graph $\mathcal{G}$.
$d^{(\mathcal{G})}_{\max}$	Il massimo grado pesato dei nodi di un $\mathcal{G}$.
$\mathcal{D}^{(i)}$	Il local dataset $\mathcal{D}^{(i)}$ associato al nodo $i \in \mathcal{V}$ of an FL network.
m_i	Il numero di punti dati (ossia, la sample size) contenuti nel local dataset $\mathcal{D}^{(i)}$ relativo al nodo $i \in \mathcal{V}$.

$\mathbf{x}^{(i,r)}$	Le caratteristica dell'r-esimo punto dati nel local dataset $\mathcal{D}^{(i)}$.
$y^{(i,r)}$	La etichetta dell'r-esimo punto dati nel local dataset $\mathcal{D}^{(i)}$.
$\mathbf{w}^{(i)}$	I model parameters locali del device i all'interno di un FL network.
$L_i\left(\mathbf{w}\right)$	La loss function locale utilizzata dal device i per valutare l'efficacia di una certa scelta $\mathbf{w}$ come model parameters locali.
$L^{(\mathrm{d})}\left(\mathbf{x}, h\big(\mathbf{x}\big), h'\big(\mathbf{x}\big)\right)$	La loss associata ad un' ipotesi h' su un punto dati con caratteristiche $\mathbf{x}$ e etichetta $h\big(\mathbf{x}\big)$, ottenuta da un'altra ipotesi.
$\mathrm{stack}\big\{\mathbf{w}^{(i)}\big\}_{i=1}^{n}$	Il vettore $\left(\left(\mathbf{w}^{(1)}\right)^{T}, \ldots, \left(\mathbf{w}^{(n)}\right)^{T}\right)^{T} \in \mathbb{R}^{dn}$ ottenuto concatenando verticalmente i model parameters locali $\mathbf{w}^{(i)} \in \mathbb{R}^{d}$.

Machine Learning Concepts

k-fold cross-validation (k-fold CV) k-fold CV is a method for learning and validating a ipotesi using a given dataset. This method divides the dataset evenly into k subsets or folds and then executes k repetitions of modello training (e.g., via empirical risk minimization (ERM)) and validation. Each repetition uses a different fold as the validation set and the remaining $k-1$ folds as a training set. The final output is the average of the validation errors obtained from the k repetitions.

k-means The k-means algorithm is a hard clustering method which assigns each punto dati of a dataset to precisely one of k different clusters. The method alternates between updating the cluster assignments (to the cluster with the nearest mean) and, given the updated cluster assignments, re-calculating the cluster means [**?**, Ch. 8].

absolute error loss Consider a punto dati with caratteristicas $\mathbf{x} \in \mathcal{X}$ and numeric etichetta $y \in \mathbb{R}$. The absolute error loss incurred by a ipotesi $h : \mathcal{X} \to \mathbb{R}$ is defined as $|y - h(\mathbf{x})|$, i.e., the absolute difference between the previsione $h(\mathbf{x})$ and the true etichetta y.

accuracy Consider punto datis characterized by caratteristicas $\mathbf{x} \in \mathcal{X}$ and a categorical label y which takes on values from a finite label space $\mathcal{Y}$. The accuracy of a ipotesi $h : \mathcal{X} \to \mathcal{Y}$, when applied to the punto

datis in a dataset $\mathcal{D} = \big\{\big(\mathbf{x}^{(1)}, y^{(1)}\big), \ldots, \big(\mathbf{x}^{(m)}, y^{(m)}\big)\big\}$, is then defined as $1 - (1/m) \sum_{r=1}^{m} L^{(0/1)} \Big(\big(\mathbf{x}^{(r)}, y^{(r)}\big), h\Big)$ using the 0/1 loss $L^{(0/1)}(\cdot, \cdot)$.

activation function Each artificial neuron within an ANN is assigned an activation function $\sigma(\cdot)$ that maps a weighted combination of the neuron inputs $x_1, \ldots, x_d$ to a single output value $a = \sigma\big(w_1 x_1 + \ldots + w_d x_d\big)$. Note that each neuron is parametrized by the weights $w_1, \ldots, w_d$.

algorithm An algorithm is a precise, step-by-step specification for how to produce an output from a given input within a finite number of computational steps [**?**]. For example, an algorithm for training a linear model explicitly describes how to transform a given training set into model parameters through a sequence of gradient steps. This informal characterization can be formalized rigorously via different mathematical modellos [**?**]. One very simple modello of an algorithm is a collection of possible executions. Each execution is a sequence:

$$\text{input}, s_1, s_2, \ldots, s_T, \text{output}$$

that respects the constraints inherent to the computer executing the algorithm. Algorithms may be deterministic, where each input results uniquely in a single execution, or randomized, where executions can vary probabilistically. Randomized algorithms can thus be analyzed by modeling execution sequences as outcomes of random experiments, viewing the algorithm as a stochastic process [**?**, **?**, **?**]. Crucially, an algorithm encompasses more than just a mapping from input to output; it also includes the intermediate computational steps $s_1, \ldots, s_T$.

application programming interface (API) An API is a formal mechanism for enabling software components to interact in a structured manner [**?**]. In the context of ML, APIs are frequently used to make a trained ML modello accessible to different types of users. These users, which can be other computers or humans, can request a previsione for the etichetta of a punto dati by providing its caratteristicas. The internal structure of the ML modello remains hidden from the user. For instance, consider a trained ML modello $\widehat{h}(x) := 2x + 1$. An API enables a user to submit the caratteristica value $x = 3$ and obtain the response $\widehat{h}(3) = 7$ without knowledge of the detailed structure of the ML modello or its training. In practice, the ML modello is typically hosted on a computer (i.e., a server) connected to the internet. Another computer (i.e., a client) sends the caratteristicas of a punto dati to the server, which then computes $\widehat{h}(\mathbf{x})$ and returns the result to the external system. APIs help to modularize the development of ML applications by decoupling specific tasks. For instance, one team can focus on developing and training the modello, while another team handles user interaction and integration of the modello into applications.

artificial intelligence (AI) AI refers to systems that behave rationally in the sense of maximizing a long-term reward. The ML-based approach to AI is to train a modello for predicting optimal actions. These predictions are computed from observations about the state of the environment. The choice of loss function sets AI applications apart from more basic ML applications. AI systems rarely have access to a labeled training set that allows the average loss to be measured for any possible choice

of model parameters. Instead, AI systems use observed reward signals to obtain a (point-wise) estimate for the loss incurred by the current choice of model parameters.

artificial neural network (ANN) An ANN is a graphical (signal-flow) representation of a function that maps caratteristicas of a punto dati at its input to a previsione for the corresponding etichetta at its output. The fundamental unit of an ANN is the artificial neuron, which applies an activation function to its weighted inputs. The outputs of these neurons serve as inputs for other neurons, forming interconnected layers.

autoencoder An autoencoder is an ML method that simultaneously learns an encoder map $h(\cdot) \in \mathcal{H}$ and a decoder map $h^{*}(\cdot) \in \mathcal{H}^{*}$. It is an instance of ERM using a loss computed from the reconstruction error $\mathbf{x} - h^{*}\big(h\big(\mathbf{x}\big)\big)$.

backdoor A backdoor attack refers to the intentional manipulation of the training process underlying an ML method. This manipulation can be implemented by perturbing the training set (dati poisoning) or the optimization algorithm used by an ERM-based method. The goal of a backdoor attack is to nudge the learned ipotesi $\hat{h}$ towards specific previsiones for a certain range of caratteristica values. This range of caratteristica values serves as a key (or trigger) to unlock a backdoor in the sense of delivering anomalous previsiones. The key $\mathbf{x}$ and the corresponding anomalous previsione $\hat{h}(\mathbf{x})$ are only known to the attacker.

bagging Bagging (or bootstrap aggregation) is a generic technique to improve (the robustness of) a given ML method. The idea is to use the bootstrap to generate perturbed copies of a given dataset and then to learn a separate ipotesi for each copy. We then predict the etichetta of a punto dati by combining or aggregating the individual previsiones of each separate ipotesi. For ipotesi maps delivering numeric etichetta values, this aggregation could be implemented by computing the average of individual previsiones.

baseline Consider some ML method that produces a learned ipotesi (or trained modello) $\hat{h} \in \mathcal{H}$. We evaluate the quality of a trained modello by computing the average loss on a test set. But how can we assess whether the resulting test set performance is sufficiently good? How can we determine if the trained modello performs close to optimal and there is little point in investing more resources (for dati collection or computation) to improve it? To this end, it is useful to have a reference (or baseline) level against which we can compare the performance of the trained modello. Such a reference value might be obtained from human performance, e.g., the misclassification rate of dermatologists who diagnose cancer from visual inspection of skin [**?**]. Another source for a baseline is an existing, but for some reason unsuitable, ML method. For example, the existing ML method might be computationally too expensive for the intended ML application. Nevertheless, its test set error can still serve as a baseline. Another, somewhat more principled, approach to constructing a baseline is via a probabilistic model. In many cases, given a probabilistic model $p(\mathbf{x}, y)$, we can precisely determine

the minimo achievable risk among any hypotheses (not even required to belong to the hypothesis space $\mathcal{H}$) [**?**]. This minimo achievable risk (referred to as the Bayes risk) is the risk of the Bayes estimator for the etichetta y of a punto dati, given its caratteristicas $\mathbf{x}$. Note that, for a given choice of loss function, the Bayes estimator (if it exists) is completely determined by the distribuzione di probabilità $p(\mathbf{x}, y)$ [**?**, Ch. 4]. However, computing the Bayes estimator and Bayes risk presents two main challenges:

1) The distribuzione di probabilità $p(\mathbf{x}, y)$ is unknown and needs to be estimated.

2) Even if $p(\mathbf{x}, y)$ is known, it can be computationally too expensive to compute the Bayes risk exactly [**?**].

A widely used probabilistic model is the multivariate normal distribution $(\mathbf{x}, y) \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ for punto datis characterized by numeric caratteristicas and etichettas. Here, for the squared error loss, the Bayes estimator is given by the posterior mean $\mu_{y|\mathbf{x}}$ of the etichetta y, given the caratteristicas $\mathbf{x}$ [**?**, **?**]. The corresponding Bayes risk is given by the posterior variance $\sigma^2_{y|\mathbf{x}}$ (see Figure 1).

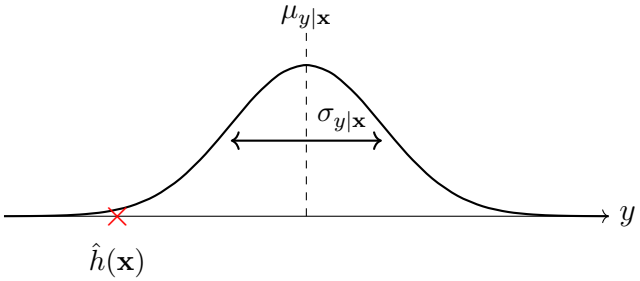


Figura 1: If the caratteristicas and the etichetta of a punto dati are drawn from a multivariate normal distribution, we can achieve the minimo risk (under squared error loss) by using the Bayes estimator $\mu_{y|\mathbf{x}}$ to predict the etichetta y of a punto dati with caratteristicas $\mathbf{x}$. The corresponding minimo risk is given by the posterior variance $\sigma^2_{y|\mathbf{x}}$. We can use this quantity as a baseline for the average loss of a trained modello $\hat{h}$.

batch In the context of stochastic gradient descent (SGD), a batch refers to a randomly chosen subset of the overall training set. We use the punto datis in this subset to estimate the gradiente of training error and, in turn, to update the model parameters.

Bayes estimator Consider a probabilistic model with a joint distribuzione di probabilità $p(\mathbf{x}, y)$ for the caratteristicas $\mathbf{x}$ and etichetta y of a punto dati. For a given loss function $L(\cdot, \cdot)$, we refer to a ipotesi h as a Bayes estimator if its risk $\mathbb{E}\{L((\mathbf{x}, y), h)\}$ is the minimo [**?**]. Note that the property of a ipotesi being a Bayes estimator depends on the underlying distribuzione di probabilità and the choice for the loss function $L(\cdot, \cdot)$.

Bayes risk Consider a probabilistic model with a joint distribuzione di

probabilità $p(\mathbf{x}, y)$ for the caratteristicas $\mathbf{x}$ and etichetta y of a punto dati. The Bayes risk is the minimo possible risk that can be achieved by any ipotesi $h : \mathcal{X} \to \mathcal{Y}$. Any ipotesi that achieves the Bayes risk is referred to as a Bayes estimator [**?**].

bias Consider an ML method using a parametrized hypothesis space $\mathcal{H}$. It learns the model parameters $\mathbf{w} \in \mathbb{R}^d$ using the dataset

$$\mathcal{D} = \left\{ \left(\mathbf{x}^{(r)}, y^{(r)}\right) \right\}_{r=1}^{m}.$$

To analyze the properties of the ML method, we typically interpret the punto datis as realizzaziones of independent and identically distributed (i.i.d.) RVs,

$$y^{(r)} = h^{(\overline{\mathbf{w}})}\left(\mathbf{x}^{(r)}\right) + \boldsymbol{\varepsilon}^{(r)}, r = 1, \ldots, m.$$

We can then interpret the ML method as an estimator $\widehat{\mathbf{w}}$ computed from $\mathcal{D}$ (e.g., by solving ERM). The (squared) bias incurred by the estimate $\widehat{\mathbf{w}}$ is then defined as $B^2 := \left\|\mathbb{E}\{\widehat{\mathbf{w}}\} - \overline{\mathbf{w}}\right\|_2^2$.

bootstrap For the analysis of ML methods, it is often useful to interpret a given set of punto datis $\mathcal{D} = \left\{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\right\}$ as realizzaziones of i.i.d. RVs with a common distribuzione di probabilità $p(\mathbf{z})$. In general, we do not know $p(\mathbf{z})$ exactly, but we need to estimate it. The bootstrap uses the histogram of $\mathcal{D}$ as an estimator for the underlying distribuzione di probabilità $p(\mathbf{z})$.

caratteristica Una caratteristica di un punto dati è una delle sue proprietà che può essere misurata o calcolata facilmente senza la necessità di

supervisione umana. Per esempio, se un punto dati è un'immagine digitale (memorizzata, ad esempio, come file `.jpeg` file), è possibile allora utilizzare le intensità dei canali rosso, verde e blu dei pixel come caratteristiche.Sinonimi del termine feature, specifici del dominio, includono: "covariata", "variabile esplicativa", "variabile indipendente", "variabile di input", "predittore" o "regressore". [?], [?], [?].

classification Classification is the task of determining a discrete-valued label y for a given punto dati, based solely on its features $\mathbf{x}$. The label y belongs to a finite set, such as $y \in \{-1, 1\}$ or $y \in \{1, \ldots, 19\}$, and represents the category to which the corresponding punto dati belongs.

classifier A classifier is a ipotesi (map) $h(\mathbf{x})$ used to predict a etichetta taking values from a finite label space. We might use the function value $h(\mathbf{x})$ itself as a previsione $\hat{y}$ for the etichetta. However, it is customary to use a map $h(\cdot)$ that delivers a numeric quantity. The previsione is then obtained by a simple thresholding step. For example, in a binary classification problem with $\mathcal{Y} \in \{-1, 1\}$, we might use a real-valued ipotesi map $h(\mathbf{x}) \in \mathbb{R}$ as a classifier. A previsione $\hat{y}$ can then be obtained via thresholding,

$$\hat{y} = 1 \text{ for } h(\mathbf{x}) \geq 0 \text{ and } \hat{y} = -1 \text{ otherwise.} \tag{1}$$

We can characterize a classifier by its decision regions $\mathcal{R}_a$, for every possible etichetta value $a \in \mathcal{Y}$.

cluster A cluster is a subset of punti dati that are more similar to each other than to the punti dati outside the cluster. The quantitative measure

of similarity between punti dati is a design choice. If punti dati are characterized by Euclidean vettori delle caratteristiche $\mathbf{x} \in \mathbb{R}^d$, we can define the similarity between two punti dati via the Euclidean distance between their vettori delle caratteristiche. An example of such clusters is shown in Fig. 2.

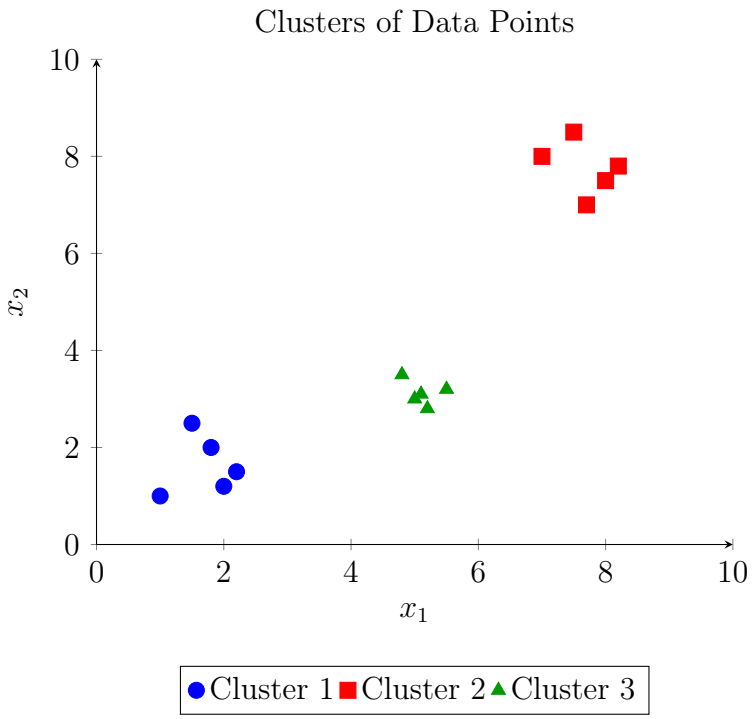


Figura 2: Illustration of three clusters in a two-dimensional feature space. Each cluster groups punti dati that are more similar to each other than to those in other clusters, based on the Euclidean distance.

See also: punto dati, vettore delle caratteristiche, feature space.

clustered federated learning (CFL) Clustered federated learning (FL) assumes that local datasets are naturally grouped into clusters. The local datasets belonging to the same cluster have similar statistical properties. Clustered FL aggregates local datasets in the same cluster to obtain a training set for the training of a cluster-specific model-lo. Generalized total variation minimization (GTVMin) facilitates this clustering implicitly by enforcing approximate similarity of model parameters across well-connected subsets of the FL network.

clustering Clustering methods decompose a given set of punto datis into a few subsets, which are referred to as clusters. Each cluster consists of punto datis that are more similar to each other than to punto datis outside the cluster. Different clustering methods use different measures for the similarity between punto datis and different forms of cluster representations. The clustering method k-means uses the average caratteristica vector (cluster mean) of a cluster as its representative. A popular soft clustering method based on Gaussian mixture model (GMM) represents a cluster by a multivariate normal distribution.

clustering assumption The clustering assumption postulates that punto datis in a dataset form a (small) number of groups or clusters. Punto datis in the same cluster are more similar to each other than those outside the cluster [**?**]. We obtain different clustering methods by using different notions of similarity between punto datis.

computational aspects By computational aspects of an ML method, we mainly refer to the computational resources required for its implemen-

tation. For example, if an ML method uses iterative optimization techniques to solve ERM, then its computational aspects include: 1) how many arithmetic operations are needed to implement a single iteration (gradient step); and 2) how many iterations are needed to obtain useful model parameters. One important example of an iterative optimization technique is gradient descent (GD).

condition number The condition number $\kappa(\mathbf{Q}) \geq 1$ of a positive definite matrix $\mathbf{Q} \in \mathbb{R}^{d \times d}$ is the ratio α/β between the largest α and the smallest β eigenvalue of $\mathbf{Q}$. The condition number is useful for the analysis of ML methods. The computational complexity of gradient-based methods for linear regression crucially depends on the condition number of the matrix $\mathbf{Q} = \mathbf{X}\mathbf{X}^T$, with the feature matrix $\mathbf{X}$ of the training set. Thus, from a computational perspective, we prefer caratteristicas of punto datis such that $\mathbf{Q}$ has a condition number close to 1.

confusion matrix Consider punto datis characterized by caratteristicas $\mathbf{x}$ and etichetta y having values from the finite label space $\mathcal{Y} = \{1, \ldots, k\}$. The confusion matrix is a $k \times k$ matrix with rows representing different values c of the true label of a punto dati. The columns of a confusion matrix correspond to different values c' delivered by a hypothesis $h(\mathbf{x})$. The (c, c')-th entry of the confusion matrix is the fraction of punto datis with the etichetta $y = c$ and the previsione $\hat{y} = c'$ assigned by the ipotesi h.

connected graph An undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is connected if every non-empty subset $\mathcal{V}' \subset \mathcal{V}$ has at least one edge connecting it to $\mathcal{V} \setminus \mathcal{V}'$.

convex A subset $\mathcal{C} \subseteq \mathbb{R}^d$ of the Euclidean space $\mathbb{R}^d$ is referred to as convex if it contains the line segment between any two points $\mathbf{x}, \mathbf{y} \in \mathcal{C}$ in that set. A function $f : \mathbb{R}^d \to \mathbb{R}$ is convex if its epigraph $\left\{ \left(\mathbf{w}^T, t \right)^T \in \mathbb{R}^{d+1} : t \geq f(\mathbf{w}) \right\}$ is a convex set [**?**]. We illustrate one example of a convex set and a convex function in Figure 3.

Figura 3: Left: A convex set $\mathcal{C} \subseteq \mathbb{R}^d$. Right: A convex function $f : \mathbb{R}^d \to \mathbb{R}$.

convex clustering Consider a dataset $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. Convex clustering learns vectors $\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(m)}$ by minimizing

$$\sum_{r=1}^{m} \left\| \mathbf{x}^{(r)} - \mathbf{w}^{(r)} \right\|_2^2 + \alpha \sum_{i,i' \in \mathcal{V}} \left\| \mathbf{w}^{(i)} - \mathbf{w}^{(i')} \right\|_p .$$

Here, $\|\mathbf{u}\|_p := \left(\sum_{j=1}^{d} |u_j|^p \right)^{1/p}$ denotes the p-norma (for $p \geq 1$). It turns out that many of the optimal vectors $\widehat{\mathbf{w}}^{(1)}, \ldots, \widehat{\mathbf{w}}^{(m)}$ coincide. A cluster then consists of those punto datis $r \in \{1, \ldots, m\}$ with identical $\widehat{\mathbf{w}}^{(r)}$ [**?**, **?**].

Courant–Fischer–Weyl min-max characterization Consider a psd matrix $\mathbf{Q} \in \mathbb{R}^{d \times d}$ with eigenvalue decomposition (EVD) (or spectral decomposition),

$$\mathbf{Q} = \sum_{j=1}^{d} \lambda_j \mathbf{u}^{(j)} \left(\mathbf{u}^{(j)} \right)^T .$$

Here, we use the ordered (in increasing fashion) eigenvalues

$$\lambda_1 \leq \ldots \leq \lambda_n.$$

The Courant–Fischer–Weyl min-max characterization [**?**, Th. 8.1.2] represents the eigenvalues of $\mathbf{Q}$ as the solutions to certain optimization problems.

covariance matrix The covariance matrix of an RV $\mathbf{x} \in \mathbb{R}^d$ is defined as $\mathbb{E}\Big\{\big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)\big(\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big)^T\Big\}$.

data augmentation Dati augmentation methods add synthetic punto datis to an existing set of punto datis. These synthetic punto datis are obtained by perturbations (e.g., adding noise to physical measurements) or transformations (e.g., rotations of images) of the original punto datis. These perturbations and transformations are such that the resulting synthetic punto datis should still have the same etichetta. As a case in point, a rotated cat image is still a cat image even if their vettore delle caratteristiches (obtained by stacking pixel color intensities) are very different (see Figure 4). Dati augmentation can be an efficient form of regularization.

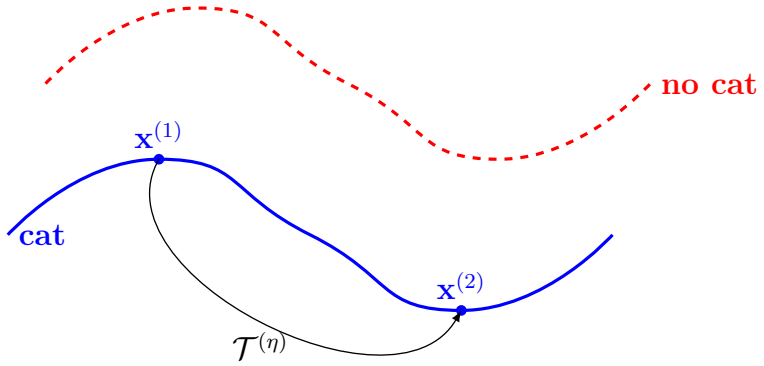


Figura 4: Dati augmentation exploits intrinsic symmetries of punto datis in some feature space $\mathcal{X}$. We can represent a symmetry by an operator $\mathcal{T}^{(\eta)} : \mathcal{X} \to \mathcal{X}$, parametrized by some number $\eta \in \mathbb{R}$. For example, $\mathcal{T}^{(\eta)}$ might represent the effect of rotating a cat image by η degrees. A punto dati with vettore delle caratteristiche $\mathbf{x}^{(2)} = \mathcal{T}^{(\eta)}\left(\mathbf{x}^{(1)}\right)$ must have the same etichetta $y^{(2)} = y^{(1)}$ as a punto dati with vettore delle caratteristiche $\mathbf{x}^{(1)}$.

data minimization principle European dati protection regulation includes a dati minimization principle. This principle requires a dati controller to limit the collection of personal information to what is directly relevant and necessary to accomplish a specified purpose. The dati should be retained only for as long as necessary to fulfill that purpose [**?**, Article 5(1)(c)], [**?**].

data normalization Dati normalization refers to transformations applied to the vettore delle caratteristiches of punto datis to improve the ML method's statistical aspects or computational aspects. For example, in linear regression with gradient-based methods using a fixed learning rate, convergence depends on controlling the norma of vettore delle

caratteristiches in the training set. A common approach is to normalize vettore delle caratteristiches such that their norma does not exceed one [**?**, Ch. 5].

data poisoning Dati poisoning refers to the intentional manipulation (or fabrication) of punto datis to steer the training of an ML modello [**?**, **?**]. The protection against dati poisoning is particularly important in distributed ML applications where datasets are decentralized.

dataset A dataset refers to a collection of punto datis. These punto datis carry information about some quantity of interest (or etichetta) within a ML application. ML methods use datasets for modello training (e.g., via ERM) and modello validation. Note that our notion of a dataset is very flexible as it allows for very different types of punto datis. Indeed, punto datis can be concrete physical objects (such as humans or animals) or abstract objects (such as numbers). As a case in point, Figure 5 depicts a dataset that consists of cows as punto datis.

Figura 5: "Cows in the Swiss Alps" by User:Huhu Uet is licensed under [CC BY-SA 4.0](https://creativecommons.org/licenses/by-sa/4.0/)

Quite often, an ML engineer does not have direct access to a dataset. Indeed, accessing the dataset in Figure would require to visit the cow herd in the Alps. Instead, we need to use an approximation (or representation) of the dataset which is more convenient to work with. Different mathematical models have been developed for the

representation (or approximation) of datasets [?], [?], [?], [?]. One of the most widely adopted data modello is the relational model, which organizes dati as a table (or relation) [?], [?]. A table consists of rows and columns:

- Each row of the table represents a single punto dati.
- Each column of the table corresponds to a specific attribute of the punto dati. ML methods can use attributes as caratteristicas and etichettas of the punto dati.

For example, Table 1 shows a representation of the dataset in Figure 5. In the relational modello, the order of rows is irrelevant, and each attribute (i.e., column) must be precisely defined with a domain, which specifies the set of possible values. In ML applications, these attribute domains become the feature space and the label space.

Name	**Weight**	**Age**	**Height**	**Stomach temp**
Zenzi	100	4	100	25
Berta	140	3	130	23
Resi	120	4	120	31

Tabella 1: A relation (or table) that represents the dataset in Figure .

While the relational model is useful for the study of many ML applications, it may be insufficient regarding the requirements for trustworthy artificial intelligence (trustworthy AI). Modern approaches like datasheets for datasets provide more comprehensive documentation, including

details about the dataset's collection process, intended use, and other contextual information [**?**].

dati Con dati ci si riferisce ad oggetti che veicolano informazione. Tali oggetti possono essere entità fisiche concrete (come persone o animali), oppure concetti astratti (come i numeri). Spesso si utilizzano rappresentazioni (o approssimazioni) dei dati originari che risultano più convenienti ai fini dell'elaborazione. Queste approssimazioni si basano su diversi modelli di dati, tra cui il modello relazionale rappresenta uno dei più comunemente adottati. [**?**].

decision boundary Consider a ipotesi map h that reads in a caratteristica vector $\mathbf{x} \in \mathbb{R}^d$ and delivers a value from a finite set $\mathcal{Y}$. The decision boundary of h is the set of vectors $\mathbf{x} \in \mathbb{R}^d$ that lie between different decision regions. More precisely, a vector $\mathbf{x}$ belongs to the decision boundary if and only if each neighborhood $\{\mathbf{x}' : \|\mathbf{x} - \mathbf{x}'\| \leq \varepsilon\}$, for any $\varepsilon > 0$, contains at least two vectors with different function values.

decision region Consider a ipotesi map h that delivers values from a finite set $\mathcal{Y}$. For each etichetta value (category) $a \in \mathcal{Y}$, the ipotesi h determines a subset of caratteristica values $\mathbf{x} \in \mathcal{X}$ that result in the same output $h(\mathbf{x}) = a$. We refer to this subset as a decision region of the ipotesi h.

decision tree A decision tree is a flow-chart-like representation of a ipotesi map h. More formally, a decision tree is a directed graph containing a root node that reads in the vettore delle caratteristiche $\mathbf{x}$ of a punto dati. The root node then forwards the punto dati to one of its children

nodes based on some elementary test on the caratteristicas $\mathbf{x}$. If the receiving child node is not a leaf node, i.e., it has itself children nodes, it represents another test. Based on the test result, the punto dati is forwarded to one of its descendants. This testing and forwarding of the punto dati is continued until the punto dati ends up in a leaf node (having no children nodes).

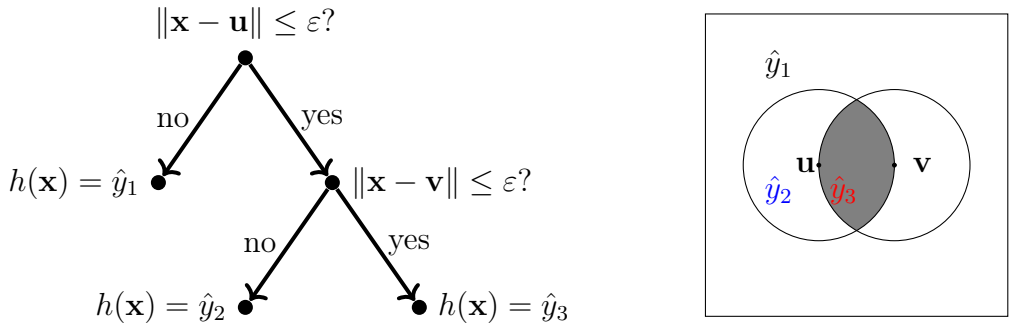


Figura 6: Left: A decision tree is a flow-chart-like representation of a piece-wise constant ipotesi $h : \mathcal{X} \to \mathbb{R}$. Each piece is a decision region $\mathcal{R}_{\hat{y}} := \big\{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = \hat{y}\big\}$. The depicted decision tree can be applied to numeric vettore delle caratteristiches, i.e., $\mathcal{X} \subseteq \mathbb{R}^d$. It is parametrized by the threshold $\varepsilon > 0$ and the vectors $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$. Right: A decision tree partitions the feature space $\mathcal{X}$ into decision regions. Each decision region $\mathcal{R}_{\hat{y}} \subseteq \mathcal{X}$ corresponds to a specific leaf node in the decision tree.

deep net A deep net is an ANN with a (relatively) large number of hidden layers. Deep learning is an umbrella term for ML methods that use a deep net as their modello [**?**].

degree of belonging Degree of belonging is a number that indicates the extent to which a punto dati belongs to a cluster [**?**, Ch. 8]. The

degree of belonging can be interpreted as a soft cluster assignment. Soft clustering methods can encode the degree of belonging by a real number in the interval $[0, 1]$. Hard clustering is obtained as the extreme case when the degree of belonging only takes on values 0 or 1.

denial-of-service attack A denial-of-service attack aims (e.g., via data poisoning) to steer the training of a modello such that it performs poorly for typical punto datis.

density-based spatial clustering of applications with noise (DBSCAN) DBSCAN refers to a clustering algorithm for punto datis that are characterized by numeric vettore delle caratteristiches. Like k-means and soft clustering via GMM, also DBSCAN uses the Euclidean distances between vettore delle caratteristiches to determine the clusters. However, in contrast to k-means and GMM, DBSCAN uses a different notion of similarity between punto datis. DBSCAN considers two punto datis as similar if they are connected via a sequence (path) of close-by intermediate punto datis. Thus, DBSCAN might consider two punto datis as similar (and therefore belonging to the same cluster) even if their vettore delle caratteristiches have a large Euclidean distance.

device Any physical system that can be used to store and process dati. In the context of ML, we typically mean a computer that is able to read in punti dati from different sources and, in turn, to train an ML modello using these punti dati.

differential privacy (DP) Consider some ML method $\mathcal{A}$ that reads in a dataset (e.g., the training set used for ERM) and delivers some output

$\mathcal{A}(\mathcal{D})$. The output could be either the learned model parameters or the previsiones for specific punto datis. DP is a precise measure of privacy leakage incurred by revealing the output. Roughly speaking, an ML method is differentially private if the distribuzione di probabilità of the output $\mathcal{A}(\mathcal{D})$ does not change too much if the sensitive attribute of one punto dati in the training set is changed. Note that DP builds on a probabilistic model for an ML method, i.e., we interpret its output $\mathcal{A}(\mathcal{D})$ as the realizzazione of an RV. The randomness in the output can be ensured by intentionally adding the realizzazione of an auxiliary RV (noise) to the output of the ML method.

differenziabile Una funzione a valori reali $f : \mathbb{R}^d \to \mathbb{R}$ si dice differenziabile se, ad ogni punto, può essere approssimata localmente da una funzione lineare. L'approssimazione lineare locale nel punto $\mathbf{x}$ è determinata dal gradiente $\nabla f(\mathbf{x})$ [?].
Si veda anche: gradiente.

discrepanza Si consideri un'applicazione di FL con networked data rappresentati da un FL network. I metodi di FL impiegano una misura di discrepanza per confrontare le funzioni ipotesi dei local model associati ai nodi i, i' connessi da un arco nel FL network.

distributed algorithm A distributed algorithm is an algorithm designed for a special type of computer: a collection of interconnected computing devices (or nodes). These devices communicate and coordinate their local computations by exchanging messages over a network [?, ?]. Unlike a classical algorithm, which is implemented on a single device,

a distributed algorithm is executed concurrently on multiple devices with computational capabilities. Similar to a classical algorithm, a distributed algorithm can be modeled as a set of potential executions. However, each execution in the distributed setting involves both local computations and message-passing events. A generic execution might look as follows:

$$\begin{aligned}
&\text{Node 1: } \text{input}_1, s_1^{(1)}, s_2^{(1)}, \ldots, s_{T_1}^{(1)}, \text{output}_1; \\
&\text{Node 2: } \text{input}_2, s_1^{(2)}, s_2^{(2)}, \ldots, s_{T_2}^{(2)}, \text{output}_2; \\
&\quad \vdots \\
&\text{Node N: } \text{input}_N, s_1^{(N)}, s_2^{(N)}, \ldots, s_{T_N}^{(N)}, \text{output}_N.
\end{aligned}$$

Each device i starts from its own local input and performs a sequence of intermediate computations $s_k^{(i)}$ at discrete time instants $k = 1, \ldots, T_i$. These computations may depend on both: the previous local computations at the device and messages received from other devices. One important application of distributed algorithms is in FL where a network of devices collaboratively train a personal modello for each device.

distribuzione di probabilità Per analizzare i metodi di ML, può essere utile interpretare i punti dati come realizzazioni i.i.d. di una RV. Le proprietà tipiche di tali punti dati sono quindi governate dalla distribuzione di probability di questa RV. La distribuzione di probability di una RV binaria $y \in \{0, 1\}$ è determinata univocamente dalle probabilità $p(y = 0)$ e $p(y = 1) = 1 - p(y = 0)$. La distribuzione di probability di una RV a valori reali $x \in \mathbb{R}$ può essere definita tramite una probability density function (pdf) $p(x)$ tale che $p(x \in [a, b]) \approx p(a)|b - a|$. In generale, una distribuzione di probability è definita da una misura di [?], [?].

Si veda anche: ML, punto dati, i.i.d., realizzazione, RV, probability, pdf.

edge weight Each edge $\{i, i'\}$ of an FL network is assigned a non-negative edge weight $A_{i,i'} \geq 0$. A zero edge weight $A_{i,i'} = 0$ indicates the absence of an edge between nodes $i, i' \in \mathcal{V}$.

effective dimension The effective dimension $d_{\text{eff}}(\mathcal{H})$ of an infinite hypothesis space $\mathcal{H}$ is a measure of its size. Loosely speaking, the effective dimension is equal to the effective number of independent tunable model parameters. These parameters might be the coefficients used in a linear map or the weights and bias terms of an ANN.

eigenvalue We refer to a number $\lambda \in \mathbb{R}$ as an eigenvalue of a square matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ if there is a non-zero vector $\mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ such that $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$.

eigenvalue decomposition (EVD) The eigenvalue decomposition for a square matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a factorization of the form

$$\mathbf{A} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^{-1}.$$

The columns of the matrix $\mathbf{V} = \big(\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(d)}\big)$ are the eigenvectors of the matrix $\mathbf{V}$. The diagonal matrix $\mathbf{\Lambda} = \operatorname{diag}\big\{\lambda_1, \ldots, \lambda_d\big\}$ contains the eigenvalues λ_j corresponding to the eigenvectors $\mathbf{v}^{(j)}$. Note that the above decomposition exists only if the matrix $\mathbf{A}$ is diagonalizable.

eigenvector An eigenvector of a matrix $\mathbf{A} \in \mathbb{R}^{d \times d}$ is a non-zero vector $\mathbf{x} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$ such that $\mathbf{A}\mathbf{x} = \lambda \mathbf{x}$ with some eigenvalue λ.

empirical risk The empirical risk $\widehat{L}\big(h|\mathcal{D}\big)$ of a ipotesi on a dataset $\mathcal{D}$ is the average loss incurred by h when applied to the punto datis in $\mathcal{D}$.

empirical risk minimization (ERM) Empirical risk minimization is the optimization problem of finding a ipotesi (out of a modello) with the minimo average loss (or empirical risk) on a given dataset $\mathcal{D}$ (i.e., the training set). Many ML methods are obtained from empirical risk via specific design choices for the dataset, modello, and loss [**?**, Ch. 3].

estimation error Consider punto datis, each with vettore delle caratteristiche $\mathbf{x}$ and etichetta y. In some applications, we can model the relation between the vettore delle caratteristiche and the etichetta of a punto dati as $y = \bar{h}(\mathbf{x}) + \varepsilon$. Here, we use some true underlying ipotesi $\bar{h}$ and a noise term ε which summarizes any modeling or labeling errors. The estimation error incurred by an ML method that learns a ipotesi $\widehat{h}$, e.g., using ERM, is defined as $\widehat{h}(\mathbf{x}) - \bar{h}(\mathbf{x})$, for some vettore delle caratteristiche. For a parametric hypothesis space, which consists of ipotesi maps determined by model parameters $\mathbf{w}$, we can define the estimation error as $\Delta\mathbf{w} = \widehat{\mathbf{w}} - \overline{\mathbf{w}}$ [**?**, **?**].

estremo superiore (o minimo dei maggioranti) L'estremo superiore di un insieme di numeri reali è è il più piccolo tra i numeri che sono maggiori o uguali ad ogni elemento dell'insieme. In termini più rigorosi, un numero reale a è l'estremo superiore di un insieme $\mathcal{A} \subseteq \mathbb{R}$ se: 1) a è un maggiorante di $\mathcal{A}$; e 2) nessun numero strettamente minore di a è un maggiorante di $\mathcal{A}$. Ogni insieme non vuoto di numeri reali che

sia superiormente limitato ammette un supremo, anche se questo non appartiene necessariamente all'insieme. [?, Sec. 1.4].

etichetta Un' informazione o una quantità di livello superiore associata a un punto dati. Ad esempio, se il punto dati è un'immagine, l'etichetta potrebbe indicare se l'immagine contiene o meno un gatto. Sinonimi di etichetta, comunemente utilizzati in ambiti specifici, includono variabile di risposta, variabile di output, label e target. [?], [?], [?].

Euclidean space The Euclidean space $\mathbb{R}^d$ of dimension $d \in \mathbb{N}$ consists of vectors $\mathbf{x} = \big(x_1, \ldots, x_d\big)$, with d real-valued entries $x_1, \ldots, x_d \in \mathbb{R}$. Such an Euclidean space is equipped with a geometric structure defined by the inner product $\mathbf{x}^T\mathbf{x}' = \sum_{j=1}^{d} x_j x'_j$ between any two vectors $\mathbf{x}, \mathbf{x}' \in \mathbb{R}^d$ [?].

expectation-maximization (EM) Consider a probabilistic model $p(\mathbf{z}; \mathbf{w})$ for the punto datis $\mathcal{D}$ generated in some ML application. The maximum likelihood estimator for the model parameters $\mathbf{w}$ is obtained by maximizing $p(\mathcal{D}; \mathbf{w})$. However, the resulting optimization problem might be computationally challenging. Valore atteso-maximization approximates the maximum likelihood estimator by introducing a latent RV $\mathbf{z}$ such that maximizing $p(\mathcal{D}, \mathbf{z}; \mathbf{w})$ would be easier [?, ?, ?]. Since we do not observe $\mathbf{z}$, we need to estimate it from the observed dataset $\mathcal{D}$ using a conditional valore atteso. The resulting estimate $\widehat{\mathbf{z}}$ is then used to compute a new estimate $\widehat{\mathbf{w}}$ by solving $\max_{\mathbf{w}} p(\mathcal{D}, \widehat{\mathbf{z}}; \mathbf{w})$. The crux is that the conditional valore atteso $\widehat{\mathbf{z}}$ depends on the model parameters $\widehat{\mathbf{w}}$, which we have updated based on $\widehat{\mathbf{z}}$. Thus, we have to re-calculate $\widehat{\mathbf{z}}$, which, in turn, results in a new choice $\widehat{\mathbf{w}}$ for the model parameters. In

practice, we repeat the computation of the conditional valore atteso (i.e., the E-step) and the update of the model parameters (i.e., the M-step) until some stopping criterion is met.

expert ML aims to learn a ipotesi h that accurately predicts the etichetta of a punto dati based on its caratteristicas. We measure the previsione error using some loss function. Ideally, we want to find a ipotesi that incurs minimal loss on any punto dati. We can make this informal goal precise via the independent and identically distributed assumption (i.i.d. assumption) and by using the Bayes risk as the baseline for the (average) loss of a ipotesi. An alternative approach to obtaining a baseline is to use the ipotesi h' learned by an existing ML method. We refer to this ipotesi h' as an expert [**?**]. Regret minimization methods learn a ipotesi that incurs a loss comparable to the best expert [**?**, **?**].

explainability We define the (subjective) explainability of an ML method as the level of simulatability [**?**] of the previsiones delivered by an ML system to a human user. Quantitative measures for the (subjective) explainability of a trained modello can be constructed by comparing its previsiones with the previsiones provided by a user on a test set [**?**, **?**]. Alternatively, we can use probabilistic models for dati and measure the explainability of a trained ML modello via the conditional (differential) entropy of its previsiones, given the user previsiones [**?**, **?**].

explainable empirical risk minimization (EERM) Explainable ERM is an instance of SRM that adds a regularization term to the average loss in the objective function of ERM. The regularization term

is chosen to favor ipotesi maps that are intrinsically explainable for a specific user. This user is characterized by their previsiones provided for the punto datis in a training set [**?**].

explainable machine learning (explainable ML) Explainable ML methods aim at complementing each previsione with an explanation of how the previsione has been obtained. The construction of an explicit explanation might not be necessary if the ML method uses a sufficiently simple (or interpretable) modello [**?**].

explanation One approach to make ML methods transparent is to provide an explanation along with the previsione delivered by an ML method. Explanations can take on many different forms. An explanation could be some natural text or some quantitative measure for the importance of individual caratteristicas of a punto dati [**?**]. We can also use visual forms of explanations, such as intensity plots for image classification [**?**].

feature learning Consider an ML application with punto datis characterized by raw caratteristicas $\mathbf{x} \in \mathcal{X}$. Caratteristica learning refers to the task of learning a map

$$\mathbf{\Phi} : \mathcal{X} \rightarrow \mathcal{X}' : \mathbf{x} \mapsto \mathbf{x}',$$

that reads in raw caratteristicas $\mathbf{x} \in \mathcal{X}$ of a punto dati and delivers new caratteristicas $\mathbf{x}' \in \mathcal{X}'$ from a new feature space $\mathcal{X}'$. Different caratteristica learning methods are obtained for different design choices of $\mathcal{X}, \mathcal{X}'$, for a hypothesis space $\mathcal{H}$ of potential maps $\mathbf{\Phi}$, and for a quantitative measure of the usefulness of a specific $\mathbf{\Phi} \in \mathcal{H}$. For example,

principal component analysis (PCA) uses $\mathcal{X} := \mathbb{R}^d$, $\mathcal{X}' := \mathbb{R}^{d'}$ with $d' < d$, and a hypothesis space

$$\mathcal{H} := \left\{ \mathbf{\Phi} : \mathbb{R}^d \to \mathbb{R}^{d'} : \mathbf{x}' := \mathbf{F}\mathbf{x} \text{ with some } \mathbf{F} \in \mathbb{R}^{d' \times d} \right\}.$$

PCA measures the usefulness of a specific map $\mathbf{\Phi}(\mathbf{x}) = \mathbf{F}\mathbf{x}$ by the minimo linear reconstruction error incurred on a dataset,

$$\min_{\mathbf{G} \in \mathbb{R}^{d \times d'}} \sum_{r=1}^{m} \left\| \mathbf{G}\mathbf{F}\mathbf{x}^{(r)} - \mathbf{x}^{(r)} \right\|_2^2 .$$

feature map Caratteristica map refers to a map that transforms the original caratteristicas of a punto dati into new caratteristicas. The so-obtained new caratteristicas might be preferable over the original caratteristicas for several reasons. For example, the arrangement of punto datis might become simpler (or more linear) in the new feature space, allowing the use of linear models in the new caratteristicas. This idea is a main driver for the development of kernel methods [**?**]. Moreover, the hidden layers of a deep net can be interpreted as a trainable caratteristica map followed by a linear model in the form of the output layer. Another reason for learning a caratteristica map could be that learning a small number of new caratteristicas helps to avoid overfitting and ensures interpretability [**?**]. The special case of a caratteristica map delivering two numeric caratteristicas is particularly useful for dati visualization. Indeed, we can depict punto datis in a scatterplot by using two caratteristicas as the coordinates of a punto dati.

feature matrix Consider a dataset $\mathcal{D}$ with m punto datis with vettore delle caratteristiches $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$. It is convenient to collect

the individual vettore delle caratteristiches into a caratteristica matrix $\mathbf{X} := \left(\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}\right)^T$ of size $m \times d$.

feature space The caratteristica space of a given ML application or method is constituted by all potential values that the vettore delle caratteristiche of a punto dati can take on. A widely used choice for the caratteristica space is the Euclidean space $\mathbb{R}^d$, with the dimension d being the number of individual caratteristicas of a punto dati.

federated averaging (FedAvg) FedAvg refers to an iterative FL algorithm that alternates between separately training local models and combining the updated local model parameters. The training of local models is implemented via several SGD steps [**?**].

federated learning (FL) FL is an umbrella term for ML methods that train modellos in a collaborative fashion using decentralized dati and computation.

federated learning network (FL network) A federated network is an undirected weighted graph whose nodes represent dati generators that aim to train a local (or personalized) modello. Each node in a federated network represents some device capable of collecting a local dataset and, in turn, train a local model. FL methods learn a local ipotesi $h^{(i)}$, for each node $i \in \mathcal{V}$, such that it incurs small loss on the local datasets.

federated learning orizzontale (FL orizzontale)) FL Orizzontale utilizza local dataset costituiti da punto dati differenti, ma impiega lo stesso insieme di caratteristica per descriverli [**?**]. Ad esempio, nella

previsione meteorologica si utilizza una rete di stazioni meteorologiche distribuite geograficamente, ciascuna delle quali misura le medesime variabili, come la temperatura giornaliera, la pressione atmosferica e le precipitazioni, ma in regioni spazio-temporali differenti. Ogni regione rappresenta un punto dati distinto, descritto attraverso lo stesso insieme di caratteristica. (ad esempio, la temperatura giornaliera o la pressione dell'aria).

FedProx FedProx refers to an iterative FL algorithm that alternates between separately training local models and combining the updated local model parameters. In contrast to FedAvg, which uses SGD to train local models, FedProx uses a proximal operator for the training [**?**].

Finnish Meteorological Institute (FMI) The FMI is a government agency responsible for gathering and reporting weather dati in Finland.

flow-based clustering Flow-based clustering groups the nodes of an undirected graph by applying k-means clustering to node-wise vettore delle caratteristiches. These vettore delle caratteristiches are built from network flows between carefully selected sources and destination nodes [**?**].

Gaussian mixture model (GMM) A GMM is a particular type of probabilistic model for a numeric vector $\mathbf{x}$ (e.g., the caratteristicas of a punto dati). Within a GMM, the vector $\mathbf{x}$ is drawn from a randomly selected multivariate normal distribution $p^{(c)} = \mathcal{N}\left(\boldsymbol{\mu}^{(c)}, \mathbf{C}^{(c)}\right)$ with $c = I$. The index $I \in \{1, \ldots, k\}$ is an RV with probabilities $p(I = c) = p_c$. Note

that a GMM is parametrized by the probability p_c, the mean vector $\boldsymbol{\mu}^{(c)}$, and the covariance matrix $\boldsymbol{\Sigma}^{(\mathbf{c})}$ for each $c = 1, \ldots, k$. GMMs are widely used for clustering, density estimation, and as a generative modello.

Gaussian random variable (Gaussian RV) A standard Gaussian RV is a real-valued RV x with pdf [?, ?, ?]

$$p(x) = \frac{1}{\sqrt{2\pi}} \exp^{-x^2/2}.$$

Given a standard Gaussian RV x, we can construct a general Gaussian RV x' with mean μ and variance σ^2 via $x' := \sigma(x+\mu)$. The distribuzione di probabilità of a Gaussian RV is referred to as normal distribution, denoted $\mathcal{N}(\mu, \sigma)$.

A Gaussian random vector $\mathbf{x} \in \mathbb{R}^d$ with covariance matrix $\mathbf{C}$ and mean $\boldsymbol{\mu}$ can be constructed via $\mathbf{x} := \mathbf{A}\big(\mathbf{z} + \boldsymbol{\mu}\big)$. Here, $\mathbf{A}$ is any matrix that satisfies $\mathbf{A}\mathbf{A}^T = \mathbf{C}$ and $\mathbf{z} := \big(z_1, \ldots, z_d\big)^T$ is a vector whose entries are i.i.d. standard Gaussian RVs $z_1, \ldots, z_d$. Gaussian random processes generalize Gaussian random vectors by applying linear transformations to infinite sequences of standard Gaussian RVs [?].

Gaussian RVs are widely used probabilistic models for the statistical analysis of ML methods. Their significance arises partly from the central limit theorem, which states that the average of an increasing number of independent RVs (not necessarily Gaussian themselves) converges to a Gaussian RV [?].

general data protection regulation (GDPR) The GDPR was enacted by the European Union (EU), effective from May 25, 2018 [?]. It safeguards the privacy and dati rights of individuals in the EU. The

GDPR has significant implications for how dati is collected, stored, and used in ML applications. Key provisions include the following:

- Data minimization principle: ML systems should only use the necessary amount of personal dati for their purpose.
- Transparency and explainability: ML systems should enable their users to understand how the systems make decisions that impact the users.
- Dati subject rights: Users should get an opportunity to access, rectify, and delete their personal dati, as well as to object to automated decision-making and profiling.
- Accountability: Organizations must ensure robust dati security and demonstrate compliance through documentation and regular audits.

generalization Many current ML (and AI) systems are based on ERM: At their core, they train a modello (i.e., learn a ipotesi $\hat{h} \in \mathcal{H}$) by minimizing the average loss (or empirical risk) on some punto datis $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$, which serve as a training set $\mathcal{D}^{(\text{train})}$. Generalization refers to an ML method's ability to perform well outside the training set. Any mathematical theory of generalization needs some mathematical concept for the "outside the training set." For example, statistical learning theory uses a probabilistic model such as the i.i.d. assumption for dati generation: the punto datis in the training set are i.i.d. realizzaziones of some underlying distribuzione di probabilità $p(\mathbf{z})$. A probabilistic model allows us to explore the outside of the training set by drawing

additional i.i.d. realizzaziones from $p(\mathbf{z})$. Moreover, using the i.i.d. assumption allows us to define the risk of a trained modello $\hat{h} \in \mathcal{H}$ as the expected loss $\bar{L}\big(\hat{h}\big)$. What is more, we can use concentration bounds or convergence results for sequences of i.i.d. RVs to bound the deviation between the empirical risk $\widehat{L}\big(\hat{h}|\mathcal{D}^{(\text{train})}\big)$ of a trained modello and its risk [?]. It is possible to study generalization also without using probabilistic models. For example, we could use (deterministic) perturbations of the punto datis in the training set to study its outside. In general, we would like the trained modello to be robust, i.e., its previsiones should not change too much for small perturbations of a punto dati. Consider a trained modello for detecting an object in a smartphone snapshot. The detection result should not change if we mask a small number of randomly chosen pixels in the image [?].

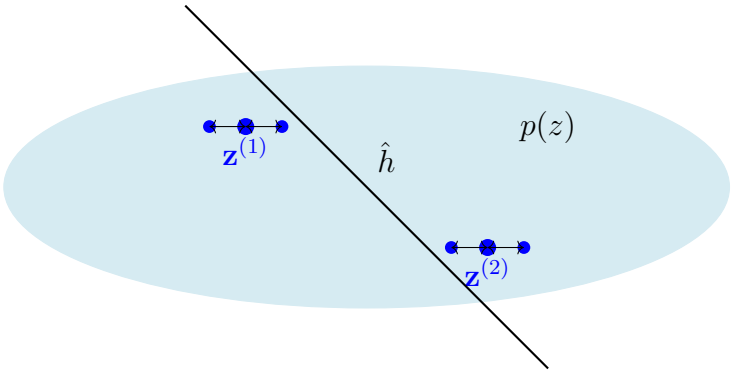


Figura 7: Two punto datis $\mathbf{z}^{(1)}, \mathbf{z}^{(2)}$ that are used as a training set to learn a ipotesi $\hat{h}$ via ERM. We can evaluate $\hat{h}$ outside $\mathcal{D}^{(\text{train})}$ either by an i.i.d. assumption with some underlying distribuzione di probabilità $p(\mathbf{z})$ or by perturbing the punto datis.

generalized total variation (GTV) GTV is a measure of the variation of trained local models $h^{(i)}$ (or their model parameters $\mathbf{w}^{(i)}$) assigned to the nodes $i = 1, \ldots, n$ of an undirected weighted graph $\mathcal{G}$ with edges $\mathcal{E}$. Given a measure $d^{(h,h')}$ for the discrepanza between ipotesi maps h, h', the GTV is

$$\sum_{\{i,i'\} \in \mathcal{E}} A_{i,i'} d^{(h^{(i)}, h^{(i')})}.$$

Here, $A_{i,i'} > 0$ denotes the weight of the undirected edge $\{i, i'\} \in \mathcal{E}$.

generalized total variation minimization (GTVMin) GTV minimization is an instance of regularized empirical risk minimization (RERM) using the GTV of local model parameters as a regularizer [**?**].

gradient descent (GD) Gradiente descent is an iterative method for finding the minimo of a differenziabile function $f(\mathbf{w})$ of a vector-valued argument $\mathbf{w} \in \mathbb{R}^d$. Consider a current guess or approximation $\mathbf{w}^{(k)}$ for the minimo of the function $f(\mathbf{w})$. We would like to find a new (better) vector $\mathbf{w}^{(k+1)}$ that has a smaller objective value $f(\mathbf{w}^{(k+1)}) < f\big(\mathbf{w}^{(k)}\big)$ than the current guess $\mathbf{w}^{(k)}$. We can achieve this typically by using a gradient step

$$\mathbf{w}^{(k+1)} = \mathbf{w}^{(k)} - \eta \nabla f(\mathbf{w}^{(k)}) \tag{2}$$

with a sufficiently small step size $\eta > 0$. Figure 8 illustrates the effect of a single gradiente descent step (2).

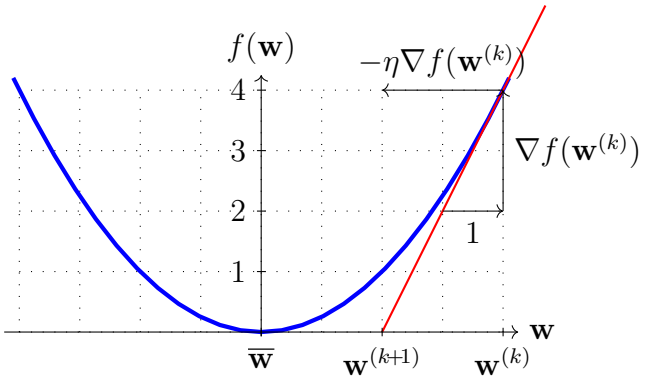


Figura 8: A single gradient step (2) towards the minimizer $\overline{\mathbf{w}}$ of $f(\mathbf{w})$.

gradient step Given a differenziabile real-valued function $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ and a vector $\mathbf{w} \in \mathbb{R}^d$, the gradiente step updates $\mathbf{w}$ by adding the scaled negative gradiente $\nabla f(\mathbf{w})$ to obtain the new vector (see Figure 9)

$$\widehat{\mathbf{w}} := \mathbf{w} - \eta\nabla f(\mathbf{w}). \tag{3}$$

Mathematically, the gradiente step is a (typically non-linear) operator $\mathcal{T}^{(f,\eta)}$ that is parametrized by the function f and the step size η.

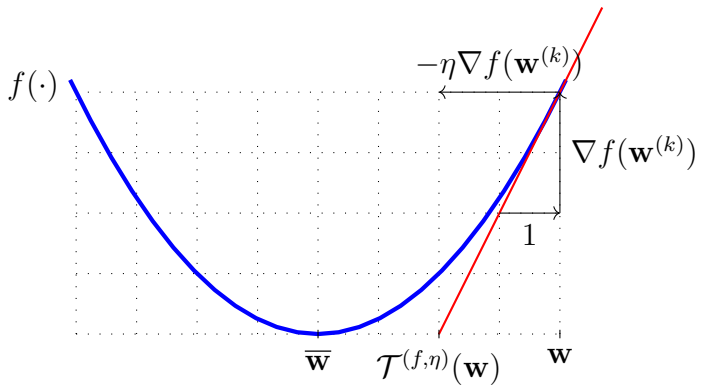


Figura 9: The basic gradiente step (3) maps a given vector $\mathbf{w}$ to the updated vector $\mathbf{w}'$. It defines an operator $\mathcal{T}^{(f,\eta)}(\cdot) : \mathbb{R}^d \to \mathbb{R}^d : \mathbf{w} \mapsto \widehat{\mathbf{w}}$.

Note that the gradiente step (3) optimizes locally - in a neighborhood whose size is determined by the step size η - a linear approximation to the function $f(\cdot)$. A natural generalization of (3) is to locally optimize the function itself - instead of its linear approximation - such that

$$\widehat{\mathbf{w}} = \operatorname*{argmin}_{\mathbf{w}' \in \mathbb{R}^d} f(\mathbf{w}') + (1/\eta) \left\| \mathbf{w} - \mathbf{w}' \right\|_2^2 . \tag{4}$$

We intentionally use the same symbol η for the parameter in (4) as we used for the step size in (3). The larger the η we choose in (4), the more progress the update will make towards reducing the function value $f(\widehat{\mathbf{w}})$. Note that, much like the gradiente step (3), also the update (4) defines a (typically non-linear) operator that is parametrized by the function $f(\cdot)$ and the parameter η. For a convex function $f(\cdot)$, this operator is known as the proximal operator of $f(\cdot)$ [**?**].

gradient-based methods Gradiente-based methods are iterative techniques for finding the minimo (or massimo) of a differenziabile objective function of the model parameters. These methods construct a sequence of approximations to an optimal choice for model parameters that results in a minimo (or massimo) value of the objective function. As their name indicates, gradiente-based methods use the gradientes of the objective function evaluated during previous iterations to construct new, (hopefully) improved model parameters. One important example of a gradiente-based method is GD.

gradiente Per funzione a valori reali $f : \mathbb{R}^d \to \mathbb{R} : \mathbf{w} \mapsto f(\mathbf{w})$, se esiste un vettore $\mathbf{g}$ tale che $\lim_{\mathbf{w}\to\mathbf{w}'} \frac{f(\mathbf{w})-\left(f(\mathbf{w}')+\mathbf{g}^T(\mathbf{w}-\mathbf{w}')\right)}{\|\mathbf{w}-\mathbf{w}'\|} = 0$ esso viene detto gradiente di f in $\mathbf{w}'$. Se esiste, il gradiente è unico e viene denotato con $\nabla f(\mathbf{w}')$ oppure con $\nabla f(\mathbf{w})\big|_{\mathbf{w}'}$ [?].

graph A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a pair that consists of a node set $\mathcal{V}$ and an edge set $\mathcal{E}$. In its most general form, a graph is specified by a map that assigns each edge $e \in \mathcal{E}$ a pair of nodes [?]. One important family of graphs is simple undirected graphs. A simple undirected graph is obtained by identifying each edge $e \in \mathcal{E}$ with two different nodes $\{i, i'\}$. Weighted graphs also specify numeric weights A_e for each edge $e \in \mathcal{E}$.

graph clustering Graph clustering aims at clustering punto datis that are represented as the nodes of a graph $\mathcal{G}$. The edges of $\mathcal{G}$ represent pairwise similarities between punto datis. Sometimes we can quantify the extend of these similarities by an edge weight [?, ?].

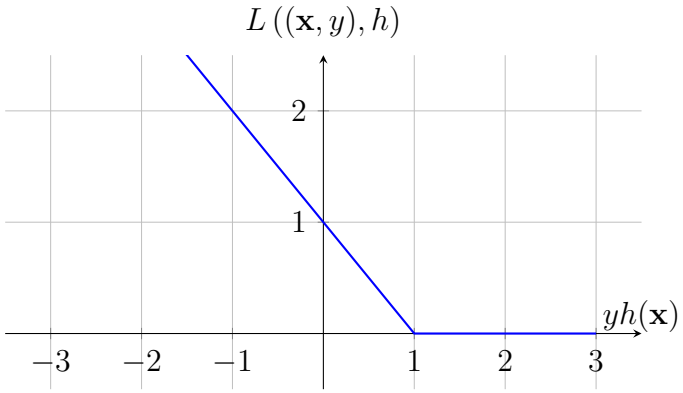
hard clustering Hard clustering refers to the task of partitioning a given set of punto datis into (a few) non-overlapping clusters. The most widely used hard clustering method is k-means.

high-dimensional regime The high-dimensional regime of ERM is characterized by the effective dimension of the modello being larger than the sample size, i.e., the number of (labeled) punto datis in the training set. For example, linear regression methods operate in the high-dimensional regime whenever the number d of caratteristicas used to characterize punto datis exceeds the number of punto datis in the training set. Another example of ML methods that operate in the high-dimensional regime is large ANNs, which have far more tunable weights (and bias terms) than the total number of punto datis in the training set. High-dimensional statistics is a recent main thread of probability theory that studies the behavior of ML methods in the high-dimensional regime [**?**, **?**].

Hilbert space A Hilbert space is a linear vector space equipped with an inner product between pairs of vectors. One important example of a Hilbert space is the Euclidean space $\mathbb{R}^d$, for some dimension d, which consists of Euclidean vectors $\mathbf{u} = \big(u_1, \ldots, u_d\big)^T$ along with the inner product $\mathbf{u}^T\mathbf{v}$.

hinge loss Consider a punto dati characterized by a vettore delle caratteristiche $\mathbf{x} \in \mathbb{R}^d$ and a binary etichetta $y \in \{-1, 1\}$. The hinge loss incurred by a real-valued ipotesi map $h(\mathbf{x})$ is defined as

$$L\left((\mathbf{x}, y), h\right) := \max\{0, 1 - yh(\mathbf{x})\}. \tag{5}$$



A regularized variant of the hinge loss is used by the support vector machine (SVM) [?].

histogram Consider a dataset $\mathcal{D}$ that consists of m punti dati $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$, each of them belonging to some cell $[-U, U] \times \ldots \times [-U, U] \subseteq \mathbb{R}^d$ with side length U. We partition this cell evenly into smaller elementary cells with side length Δ. The histogram of $\mathcal{D}$ assigns each elementary cell to the corresponding fraction of punti dati in $\mathcal{D}$ that fall into this elementary cell. A visual example of such a histogram is provided in Fig. 10.

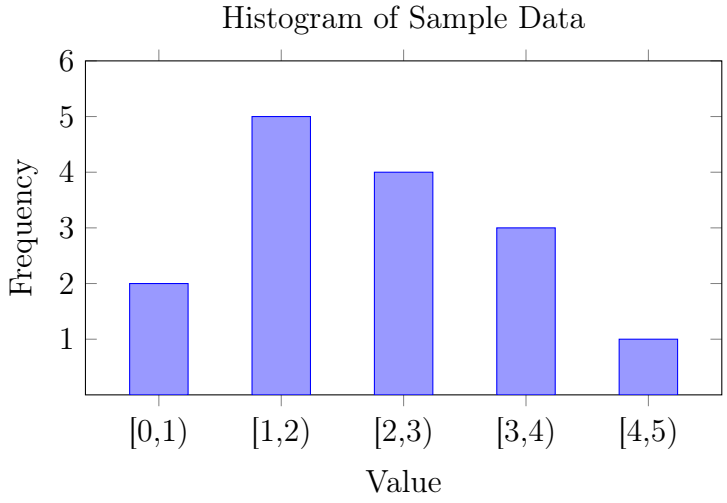


Figura 10: A histogram representing the frequency of punti dati falling within discrete value ranges (i.e., bins). Each bar height shows the count of samples in the corresponding interval.

See also: dataset, punto dati, sample.

Huber loss The Huber loss unifies the squared error loss and the absolute error loss.

Huber regression Huber regression refers to ERM-based methods that use the Huber loss as a measure of the previsione error. Two important special cases of Huber regression are least absolute deviation regression and linear regression. Tuning the threshold parameter of the Huber loss allows the user to trade the robustness of the absolute error loss against the computational benefits of the smooth squared error loss.

hypothesis space Every practical ML method uses a ipotesi space (or modello) $\mathcal{H}$. The ipotesi space of an ML method is a subset of all possible

maps from the feature space to the label space. The design choice of the ipotesi space should take into account available computational resources and statistical aspects. If the computational infrastructure allows for efficient matrix operations, and there is an (approximately) linear relation between a set of caratteristicas and a etichetta, a useful choice for the ipotesi space might be the linear model.

independent and identically distributed (i.i.d.) It can be useful to interpret punto datis $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$ as realizzaziones of i.i.d. RVs with a common distribuzione di probabilità. If these RVs are continuous-valued, their joint pdf is $p\big(\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\big) = \prod_{r=1}^{m} p\big(\mathbf{z}^{(r)}\big)$, with $p(\mathbf{z})$ being the common marginal pdf of the underlying RVs.

independent and identically distributed assumption (i.i.d. assumption) The i.i.d. assumption interprets punto datis of a dataset as the realizzaziones of i.i.d. RVs.

interpretability An ML method is interpretable for a specific user if they can well anticipate the previsiones delivered by the method. The notion of interpretability can be made precise using quantitative measures of the uncertainty about the previsiones [**?**].

ipotesi Una ipotesi è una mappa (o funzione) $h : \mathcal{X} \to \mathcal{Y}$ dallo feature space $\mathcal{X}$ allo label space $\mathcal{Y}$. Dato un punto dati con caratteristiche $\mathbf{x}$, utilizziamo una funzione ipotesi h per stimare (o approssimare) l' etichetta y mediante la previsione $\hat{y} = h(\mathbf{x})$. Il ML si occupa essenzialmente di

apprendere (o individuare) una funzione ipotesi h tale che $y \approx h(\mathbf{x})$ per qualsiasi punto dati (avente caratteristiche $\mathbf{x}$ ed etichetta y).

kernel Consider punto datis characterized by a vettore delle caratteristiche $\mathbf{x} \in \mathcal{X}$ with a generic feature space $\mathcal{X}$. A (real-valued) kernel $K : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ assigns each pair of vettore delle caratteristiches $\mathbf{x}, \mathbf{x}' \in \mathcal{X}$ a real number $K\big(\mathbf{x}, \mathbf{x}'\big)$. The value $K\big(\mathbf{x}, \mathbf{x}'\big)$ is often interpreted as a measure for the similarity between $\mathbf{x}$ and $\mathbf{x}'$. Kernel methods use a kernel to transform the vettore delle caratteristiche $\mathbf{x}$ to a new vettore delle caratteristiche $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$. This new vettore delle caratteristiche belongs to a linear feature space $\mathcal{X}'$ which is (in general) different from the original feature space $\mathcal{X}$. The feature space $\mathcal{X}'$ has a specific mathematical structure, i.e., it is a reproducing kernel Hilbert space [?, ?].

kernel method A kernel method is an ML method that uses a kernel K to map the original (raw) vettore delle caratteristiche $\mathbf{x}$ of a punto dati to a new (transformed) vettore delle caratteristiche $\mathbf{z} = K\big(\mathbf{x}, \cdot\big)$ [?, ?]. The motivation for transforming the vettore delle caratteristiches is that, by using a suitable kernel, the punto datis have a "more pleasant" geometry in the transformed feature space. For example, in a binary classification problem, using transformed vettore delle caratteristiches $\mathbf{z}$ might allow us to use linear models, even if the punto datis are not linearly separable in the original feature space (see Figure 11).

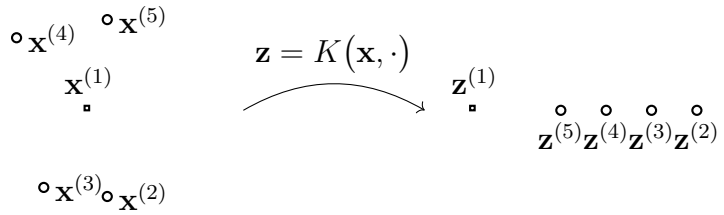


Figura 11: Five punto datis characterized by vettore delle caratteristiches $\mathbf{x}^{(r)}$ and etichettas $y^{(r)} \in \{\circ, \square\}$, for $r = 1, \ldots, 5$. With these vettore delle caratteristiches, there is no way to separate the two classes by a straight line (representing the decision boundary of a linear classifier). In contrast, the transformed vettore delle caratteristiches $\mathbf{z}^{(r)} = K(\mathbf{x}^{(r)}, \cdot)$ allow us to separate the punto datis using a linear classifier.

Kullback-Leibler divergence (KL divergence) The KL divergence is a quantitative measure of how much one distribuzione di probabilità is different from another distribuzione di probabilità [**?**].

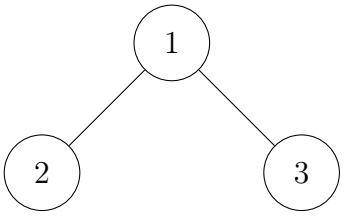
label space Consider an ML application that involves punto datis characterized by caratteristicas and etichettas. The etichetta space is constituted by all potential values that the etichetta of a punto dati can take on. Regression methods, aiming at predicting numeric etichettas, often use the etichetta space $\mathcal{Y} = \mathbb{R}$. Binary classification methods use a etichetta space that consists of two different elements, e.g., $\mathcal{Y} = \{-1, 1\}$, $\mathcal{Y} = \{0, 1\}$, or $\mathcal{Y} = \{$"cat image","no cat image"$\}$.

labeled datapoint A punto dati whose etichetta is known or has been determined by some means which might require human labor.

Laplacian matrix The structure of a graph $\mathcal{G}$, with nodes $i = 1, \ldots, n$, can be analyzed using the properties of special matrices that are associated with $\mathcal{G}$. One such matrix is the graph Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{n \times n}$, which is defined for an undirected and weighted graph [?, ?]. It is defined element-wise as (see Figure 12)

$$L_{i,i'}^{(\mathcal{G})} := \begin{cases} -A_{i,i'} & \text{for } i \neq i', \{i, i'\} \in \mathcal{E}, \\ \sum_{i'' \neq i} A_{i,i''} & \text{for } i = i', \\ 0 & \text{else.} \end{cases} \tag{6}$$

Here, $A_{i,i'}$ denotes the edge weight of an edge $\{i, i'\} \in \mathcal{E}$.



$$\mathbf{L}^{(\mathcal{G})} = \begin{pmatrix} 2 & -1 & -1 \\ -1 & 1 & 0 \\ -1 & 0 & 1 \end{pmatrix}$$

Figura 12: Left: Some undirected graph $\mathcal{G}$ with three nodes $i = 1, 2, 3$. Right: The Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{3 \times 3}$ of $\mathcal{G}$.

large language model (LLM) Large language modellos is an umbrella term for ML methods that process and generate human-like text. These methods typically use deep nets with billions (or even trillions) of parameters. A widely used choice for the network architecture is referred to as Transformers [?]. The training of large language modellos is often based on the task of predicting a few words that are intentionally removed from a large text corpus. Thus, we can construct labeled datapoints

simply by selecting some words of a text as etichettas and the remaining words as caratteristicas of punto datis. This construction requires very little human supervision and allows for generating sufficiently large training sets for large language modellos.

law of large numbers The law of large numbers refers to the convergence of the average of an increasing (large) number of i.i.d. RVs to the mean of their common distribuzione di probabilità. Different instances of the law of large numbers are obtained by using different notions of convergence [**?**].

learning rate Consider an iterative ML method for finding or learning a useful ipotesi $h \in \mathcal{H}$. Such an iterative method repeats similar computational (update) steps that adjust or modify the current ipotesi to obtain an improved ipotesi. One well-known example of such an iterative learning method is GD and its variants, SGD and projected gradient descent (projected GD). A key parameter of an iterative method is the learning rate. The learning rate controls the extent to which the current ipotesi can be modified during a single iteration. A well-known example of such a parameter is the step size used in GD [**?**, Ch. 5].

learning task Consider a dataset $\mathcal{D}$ constituted by several punto datis, each of them characterized by caratteristicas $\mathbf{x}$. For example, the dataset $\mathcal{D}$ might be constituted by the images of a particular database. Sometimes it might be useful to represent a dataset $\mathcal{D}$, along with the choice of caratteristicas, by a distribuzione di probabilità $p(\mathbf{x})$. A learning task associated with $\mathcal{D}$ consists of a specific choice for the etichetta of a

punto dati and the corresponding label space. Given a choice for the loss function and modello, a learning task gives rise to an instance of ERM. Thus, we could define a learning task also via an instance of ERM, i.e., via an objective function. Note that, for the same dataset, we obtain different learning tasks by using different choices for the caratteristicas and etichetta of a punto dati. These learning tasks are related, as they are based on the same dataset, and solving them jointly (via multitask learning methods) is typically preferable over solving them separately [**?**], [**?**], [**?**].

least absolute deviation regression Least absolute deviation regression is an instance of ERM using the absolute error loss. It is a special case of Huber regression.

least absolute shrinkage and selection operator (Lasso) The Lasso is an instance of SRM. It learns the weights $\mathbf{w}$ of a linear map $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ based on a training set. Lasso is obtained from linear regression by adding the scaled ℓ_1-norma $\alpha \left\|\mathbf{w}\right\|_1$ to the average squared error loss incurred on the training set.

linear classifier Consider punto datis characterized by numeric caratteristicas $\mathbf{x} \in \mathbb{R}^d$ and a etichetta $y \in \mathcal{Y}$ from some finite label space $\mathcal{Y}$. A linear classifier is characterized by having decision regions that are separated by hyperplanes in $\mathbb{R}^d$ [**?**, Ch. 2].

linear model Consider punto datis, each characterized by a numeric vettore delle caratteristiche $\mathbf{x} \in \mathbb{R}^d$. A linear modello is a hypothesis space

which consists of all linear maps,

$$\mathcal{H}^{(d)} := \left\{ h(\mathbf{x}) = \mathbf{w}^T \mathbf{x} : \mathbf{w} \in \mathbb{R}^d \right\}. \quad (7)$$

Note that (7) defines an entire family of hypothesis spaces, which is parametrized by the number d of caratteristicas that are linearly combined to form the previsione $h(\mathbf{x})$. The design choice of d is guided by computational aspects (e.g., reducing d means less computation), statistical aspects (e.g., increasing d might reduce previsione error), and interpretability. A linear modello using few carefully chosen caratteristicas tends to be considered more interpretable [?, ?].

linear regression Linear regression aims to learn a linear ipotesi map to predict a numeric etichetta based on the numeric caratteristicas of a punto dati. The quality of a linear ipotesi map is measured using the average squared error loss incurred on a set of labeled datapoints, which we refer to as the training set.

local dataset The concept of a local dataset is in between the concept of a punto dati and a dataset. A local dataset consists of several individual punto datis, which are characterized by caratteristicas and etichettas. In contrast to a single dataset used in basic ML methods, a local dataset is also related to other local datasets via different notions of similarity. These similarities might arise from probabilistic models or communication infrastructure and are encoded in the edges of an FL network.

Local Interpretable Model-agnostic Explanations (LIME) Consider a trained modello (or learnt ipotesi) $\widehat{h} \in \mathcal{H}$, which maps the vettore delle

caratteristiche of a punto dati to the previsione $\widehat{y} = \widehat{h}$. Local Interpretable Model-agnostic Explanations (LIME) is a technique for explaining the behaviour of $\widehat{h}$, locally around a punto dati with vettore delle caratteristiche $\mathbf{x}^{(0)}$ [**?**]. The explanation is given in the form of a local approximation $g \in \mathcal{H}'$ of $\widehat{h}$ (see Fig.). This approximation can be obtained by an instance of ERM with carefully designed training set. In particular, the training set consists of punto datis with vettore delle caratteristiche $\mathbf{x}$ close to $\mathbf{x}^{(0)}$ and the (pseudo-)label $\widehat{h}(\mathbf{x})$. Note that we can use a different modello $\mathcal{H}'$ for the approximation than the original modello $\mathcal{H}$. For example, we can use a decision tree to approximate (locally) a deep net. Another widely-used choice for $\mathcal{H}'$ is the linear model.

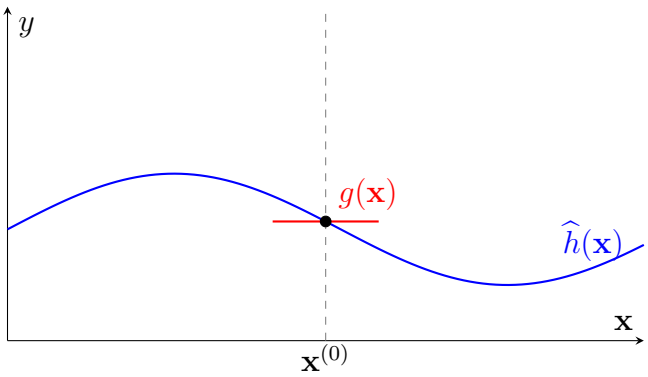


Figura 13: To explain a trained modello $\widehat{h} \in \mathcal{H}$, around a given vettore delle caratteristiche $\mathbf{x}^{(0)}$, we can use a local approximation $g \in \mathcal{H}'$.

local model Consider a collection of local datasets that are assigned to the nodes of an FL network. A local modello $\mathcal{H}^{(i)}$ is a hypothesis

space assigned to a node $i \in \mathcal{V}$. Different nodes might be assigned different hypothesis spaces, i.e., in general $\mathcal{H}^{(i)} \neq \mathcal{H}^{(i')}$ for different nodes $i, i' \in \mathcal{V}$.

logistic loss Consider a punto dati characterized by the caratteristicas $\mathbf{x}$ and a binary etichetta $y \in \{-1, 1\}$. We use a real-valued ipotesi h to predict the etichetta y from the caratteristicas $\mathbf{x}$. The logistic loss incurred by this previsione is defined as

$$L\left((\mathbf{x}, y), h\right) := \log(1 + \exp(-yh(\mathbf{x}))). \tag{8}$$

Carefully note that the expression (8) for the logistic loss applies only for the label space $\mathcal{Y} = \{-1, 1\}$ and when using the thresholding rule (1).

logistic regression Logistic regression learns a linear ipotesi map (or classifier) $h(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$ to predict a binary etichetta y based on the numeric vettore delle caratteristiche $\mathbf{x}$ of a punto dati. The quality of a linear ipotesi map is measured by the average logistic loss on some labeled datapoints (i.e., the training set).

loss ML methods use a loss function $L\left(\mathbf{z}, h\right)$ to measure the error incurred by applying a specific ipotesi to a specific punto dati. With a slight abuse of notation, we use the term loss for both the loss function L itself and the specific value $L\left(\mathbf{z}, h\right)$, for a punto dati $\mathbf{z}$ and ipotesi h.

loss function A loss function is a map

$$L : \mathcal{X} \times \mathcal{Y} \times \mathcal{H} \to \mathbb{R}_+ : \big(\big(\mathbf{x}, y\big), h\big) \mapsto L\left((\mathbf{x}, y), h\right).$$

It assigns a non-negative real number (i.e., the loss) $L\left((\mathbf{x}, y), h\right)$ to a pair that consists of a punto dati, with caratteristicas $\mathbf{x}$ and etichetta y, and a ipotesi $h \in \mathcal{H}$. The value $L\left((\mathbf{x}, y), h\right)$ quantifies the discrepancy between the true etichetta y and the previsione $h(\mathbf{x})$. Lower (closer to zero) values $L\left((\mathbf{x}, y), h\right)$ indicate a smaller discrepancy between previsione $h(\mathbf{x})$ and etichetta y. Figure 14 depicts a loss function for a given punto dati, with caratteristicas $\mathbf{x}$ and etichetta y, as a function of the ipotesi $h \in \mathcal{H}$.

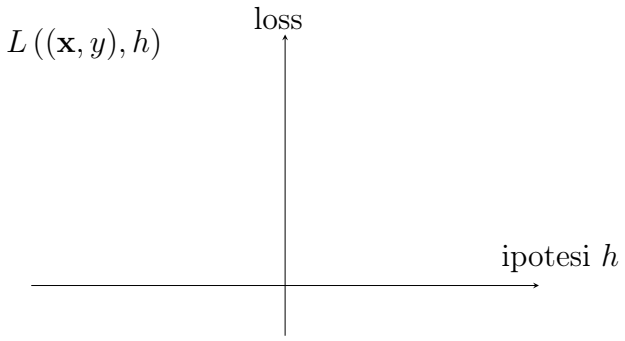


Figura 14: Some loss function $L\left((\mathbf{x}, y), h\right)$ for a fixed punto dati, with vettore delle caratteristiche $\mathbf{x}$ and etichetta y, and a varying ipotesi h. ML methods try to find (or learn) a ipotesi that incurs minimal loss.

machine learning (ML) Il ML ha come scopo la previsione una etichetta a partire dalle caratteristicas di un punto dati. Le tecniche di ML raggiungono quest'obiettivo apprendendo una ipotesi da uno hypothesis space (o modello) mediante la minimizzazione di una loss function [**?**, **?**]. Una precisa formulazione di questo principio è ERM. I metodi di ML si differenziano in base a differenti scelte inerenti alla definizione dei

punto datis (in termini di caratteristica e etichetta), alla struttura del modello, e alla specificazione della loss function [?, Ch. 3].

massimo Il massimo di un insieme $\mathcal{A} \subseteq \mathbb{R}$ di numeri reali, qualora esista, è l'elemento più grande appartenente a tale insieme. Un insieme $\mathcal{A}$ ammette un massimo se è limitato superiormente e il suo estremo superiore (o minimo dei maggioranti) appartiene ad $\mathcal{A}$ [?, Sec. 1.4].

maximum likelihood Consider punto datis $\mathcal{D} = \{\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}\}$ that are interpreted as the realizzazioni of i.i.d. RVs with a common distribuzione di probabilità $p(\mathbf{z}; \mathbf{w})$ which depends on the model parameters $\mathbf{w} \in \mathcal{W} \subseteq \mathbb{R}^n$. Massimo likelihood methods learn model parameters $\mathbf{w}$ by maximizing the probability (density) $p(\mathcal{D}; \mathbf{w}) = \prod_{r=1}^{m} p(\mathbf{z}^{(r)}; \mathbf{w})$ of the observed dataset. Thus, the massimo likelihood estimator is a solution to the optimization problem $\max_{\mathbf{w} \in \mathcal{W}} p(\mathcal{D}; \mathbf{w})$.

mean The valore atteso $\mathbb{E}\{\mathbf{x}\}$ of a numeric RV $\mathbf{x}$.

mean squared estimation error (MSEE) Consider an ML method that learns model parameters $\widehat{\mathbf{w}}$ based on some dataset $\mathcal{D}$. If we interpret the punto datis in $\mathcal{D}$ as i.i.d. realizzazioni of an RV $\mathbf{z}$, we define the estimation error $\Delta\mathbf{w} := \widehat{w} - \overline{\mathbf{w}}$. Here, $\overline{\mathbf{w}}$ denotes the true model parameters of the distribuzione di probabilità of $\mathbf{z}$. The mean squared estimation error is defined as the valore atteso $\mathbb{E}\{\|\Delta\mathbf{w}\|^2\}$ of the squared Euclidean norma of the estimation error [?, ?].

minimo Dato un insieme di numeri reali, il minimo è il più piccolo tra questi.

missing data Consider a dataset constituted by punto datis collected via some physical device. Due to imperfections and failures, some of the caratteristica or etichetta values of punto datis might be corrupted or simply missing. Dati imputation aims at estimating these missing values [**?**]. We can interpret dati imputation as an ML problem where the etichetta of a punto dati is the value of the corrupted caratteristica.

model parameters Modello parameters are quantities that are used to select a specific ipotesi map from a modello. We can think of a list of modello parameters as a unique identifier for a ipotesi map, similar to how a social security number identifies a person in Finland.

model selection In ML, modello selection refers to the process of choosing between different candidate modellos. In its most basic form, modello selection amounts to: 1) training each candidate modello; 2) computing the validation error for each trained modello; and 3) choosing the modello with the smallest validation error [**?**, Ch. 6].

modello Nel contesto dei metodi di ML, il termine modello si riferisce tipicamente allo hypothesis space alla base di un metodo di ML [**?**, **?**]. Tuttavia, il termine è usato anche in altri ambiti, ma con un significato diverso. Per esempio, un probabilistic model si riferisce ad un insieme parametrizzato di distribuzioni di probabilità.

multi-armed bandit A multi-armed bandit (MAB) problem models a repeated decision-making scenario in which, at each time step k, a learner must choose one out of several possible actions, often referred to as arms, from a finite set $\mathcal{A}$. Each arm $a \in \mathcal{A}$ yields a stochastic reward

$r^{(a)}$ drawn from an unknown distribuzione di probabilità with mean $\mu^{(a)}$. The learner's goal is to maximize the cumulative reward over time by strategically balancing exploration (gathering information about uncertain arms) and exploitation (selecting arms known to perform well). This balance is quantified by the notion of regret, which measures the performance gap between the learner's strategy and the optimal strategy that always selects the best arm. MAB problems form a foundational model in online learning, reinforcement learning, and sequential experimental design [**?**].

multi-label classification Multi-etichetta classification problems and methods use punto datis that are characterized by several etichettas. As an example, consider a punto dati representing a picture with two etichettas. One etichetta indicates the presence of a human in this picture and another etichetta indicates the presence of a car.

multitask learning Multitask learning aims at leveraging relations between different learning tasks. Consider two learning tasks obtained from the same dataset of webcam snapshots. The first task is to predict the presence of a human, while the second task is to predict the presence of a car. It might be useful to use the same deep net structure for both tasks and only allow the weights of the final output layer to be different.

multivariate normal distribution The multivariate normal distribution $\mathcal{N}(\mathbf{m}, \mathbf{C})$ is an important family of distribuzione di probabilitàs for a continuous RV $\mathbf{x} \in \mathbb{R}^d$ [**?**, **?**, **?**]. This family is parametrized by the mean $\mathbf{m}$ and the covariance matrix $\mathbf{C}$ of $\mathbf{x}$. If the covariance matrix is

invertible, the distribuzione di probabilità of $\mathbf{x}$ is

$$p(\mathbf{x}) \propto \exp\Big(-(1/2)\big(\mathbf{x}-\mathbf{m}\big)^{T}\mathbf{C}^{-1}\big(\mathbf{x}-\mathbf{m}\big)\Big).$$

mutual information (MI) The MI $I(\mathbf{x};y)$ between two RVs $\mathbf{x}$, y defined on the same probability space is given by [?]

$$I(\mathbf{x};y) := \mathbb{E}\left\{\log \frac{p(\mathbf{x},y)}{p(\mathbf{x})p(y)}\right\}.$$

It is a measure of how well we can estimate y based solely on $\mathbf{x}$. A large value of $I(\mathbf{x};y)$ indicates that y can be well predicted solely from $\mathbf{x}$. This previsione could be obtained by a ipotesi learned by an ERM-based ML method.

nearest neighbor (NN) NN methods learn a ipotesi $h : \mathcal{X} \to \mathcal{Y}$ whose function value $h(\mathbf{x})$ is solely determined by the nearest neighbors within a given dataset. Different methods use different metrics for determining the nearest neighbors. If punto datis are characterized by numeric vettore delle caratteristiches, we can use their Euclidean distances as the metric.

neighborhood The neighborhood of a node $i \in \mathcal{V}$ is the subset of nodes constituted by the neighbors of i.

neighbors The neighbors of a node $i \in \mathcal{V}$ within an FL network are those nodes $i' \in \mathcal{V} \setminus \{i\}$ that are connected (via an edge) to node i.

networked data Networked dati consists of local datasets that are related by some notion of pairwise similarity. We can represent networked dati

using a graph whose nodes carry local datasets and edges encode pairwise similarities. One example of networked dati arises in FL applications where local datasets are generated by spatially distributed devices.

networked exponential families (nExpFam) A collection of exponential families, each of them assigned to a node of an FL network. The model parameters are coupled via the network structure by requiring them to have a small GTV [**?**].

networked federated learning (NFL) Networked FL refers to methods that learn personalized modellos in a distributed fashion. These methods learn from local datasets that are related by an intrinsic network structure.

networked model A networked modello over an FL network $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ assigns a local model (i.e., a hypothesis space) to each node $i \in \mathcal{V}$ of the FL network $\mathcal{G}$.

node degree The degree $d^{(i)}$ of a node $i \in \mathcal{V}$ in an undirected graph is the number of its neighbors, i.e., $d^{(i)} := \left|\mathcal{N}^{(i)}\right|$.

non-smooth We refer to a function as non-smooth if it is not smooth [**?**].

norma A norm è una funzione che associa a ciascun elemento (vettore) di uno spazio vettoriale lineare un numero reale non negativo. Tale funzione deve essere omogenea e definita, e deve soddisfare la disuguaglianza triangolare [**?**].

objective function An objective function is a map that assigns each value of an optimization variable, such as the model parameters $\mathbf{w}$ of a ipotesi $h^{(\mathbf{w})}$, to an objective value $f(\mathbf{w})$. The objective value $f(\mathbf{w})$ could be the risk or the empirical risk of a ipotesi $h^{(\mathbf{w})}$.

online algorithm An online algorithm processes input dati incrementally, receiving dati items sequentially and making decisions or producing outputs (or decisions) immediately without having access to the entire input in advance [**?**, **?**]. Unlike an offline algorithm, which has the entire input available from the start, an online algorithm must handle uncertainty about future inputs and cannot revise past decisions. Similar to an offline algorithm, an online algorithm can be modeled formally as a collection of possible executions. However, the execution sequence for an online algorithm has a distinct structure:

$$\text{init}, s_1, \text{out}_1, \text{in}_2, s_2, \text{out}_2, \ldots, \text{in}_T, s_T, \text{out}_T.$$

Each execution begins from an initial state (init) and proceeds through alternating computational steps, outputs (or decisions), and inputs. Specifically, at step k, the algorithm performs a computational step s_k, generates an output out_k, and then subsequently receives the next input in_{k+1}. A notable example of an online algorithm in ML is online gradient descent (online GD) (online gradient descent), which incrementally updates model parameters as new punto datis arrive.

online gradient descent (online GD) Consider an ML method that learns model parameters $\mathbf{w}$ from some parameter space $\mathcal{W} \subseteq \mathbb{R}^d$. The learning process uses punto datis $\mathbf{z}^{(t)}$ that arrive at consecutive time-instants

$t = 1, 2, \ldots$. Let us interpret the punto datis $\mathbf{z}^{(t)}$ as i.i.d. copies of an RV $\mathbf{z}$. The risk $\mathbb{E}\{L\left(\mathbf{z}, \mathbf{w}\right)\}$ of a ipotesi $h^{(\mathbf{w})}$ can then (under mild conditions) be obtained as the limit $\lim_{T\to\infty}(1/T)\sum_{t=1}^{T} L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$. We might use this limit as the objective function for learning the model parameters $\mathbf{w}$. Unfortunately, this limit can only be evaluated if we wait infinitely long in order to collect all punto datis. Some ML applications require methods that learn online: as soon as a new punto dati $\mathbf{z}^{(t)}$ arrives at time t, we update the current model parameters $\mathbf{w}^{(t)}$. Note that the new punto dati $\mathbf{z}^{(t)}$ contributes the component $L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)$ to the risk. As its name suggests, online GD updates $\mathbf{w}^{(t)}$ via a (projected) gradient step

$$\mathbf{w}^{(t+1)} := P_{\mathcal{W}}\big(\mathbf{w}^{(t)} - \eta_t \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(t)}, \mathbf{w}\right)\big). \tag{9}$$

Note that (9) is a gradient step for the current component $L\left(\mathbf{z}^{(t)}, \cdot\right)$ of the risk. The update (9) ignores all the previous components $L\left(\mathbf{z}^{(t')}, \cdot\right)$, for $t' < t$. It might therefore happen that, compared to $\mathbf{w}^{(t)}$, the updated model parameters $\mathbf{w}^{(t+1)}$ increase the retrospective average loss $\sum_{t'=1}^{t-1} L\left(\mathbf{z}^{(t')}, \cdot\right)$. However, for a suitably chosen learning rate η_t, online GD can be shown to be optimal in practically relevant settings. By optimal, we mean that the model parameters $\mathbf{w}^{(T+1)}$ delivered by online GD after observing T punto datis $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(T)}$ are at least as good as those delivered by any other learning method [**?**, **?**].

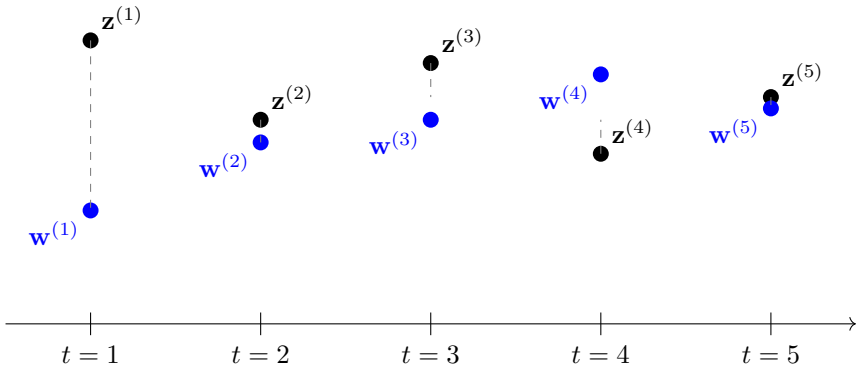


Figura 15: An instance of online GD that updates the model parameters $\mathbf{w}^{(t)}$ using the punto dati $\mathbf{z}^{(t)} = x^{(t)}$ arriving at time t. This instance uses the squared error loss $L\left(\mathbf{z}^{(t)}, w\right) = (x^{(t)} - w)^2$.

optimism in the face of uncertainty ML methods learn model parameters $\mathbf{w}$ according to some performance criterion $\bar{f}(\mathbf{w})$. However, they usually cannot access $\bar{f}(\mathbf{w})$ directly but rely on an estimate (or approximation) of $f(\mathbf{w})$. As a case in point, ERM-based methods use the average loss on a given dataset (i.e., the training set) as an estimate for the risk of a ipotesi. Using a probabilistic model, one can construct a confidence interval $\left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$ for each choice $\mathbf{w}$ for the model parameters. One simple construction is $l^{(\mathbf{w})} := f(\mathbf{w}) - \sigma/2$, $u^{(\mathbf{w})} := f(\mathbf{w}) + \sigma/2$, with σ being a measure of the (expected) deviation of $f(\mathbf{w})$ from $\bar{f}(\mathbf{w})$. We can also use other constructions for this interval as long as they ensure that $\bar{f}(\mathbf{w}) \in \left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$ with a sufficiently high probability. As optimists, we choose the model parameters according to the most favorable - yet still plausible - value $\tilde{f}(\mathbf{w}) := l^{(\mathbf{w})}$ of the

performance criterion. Two examples of ML methods that use such an optimistic construction of an objective function are SRM [**?**, Ch. 11] and upper confidence bound (UCB) methods for sequential decision making [**?**, Sec. 2.2].

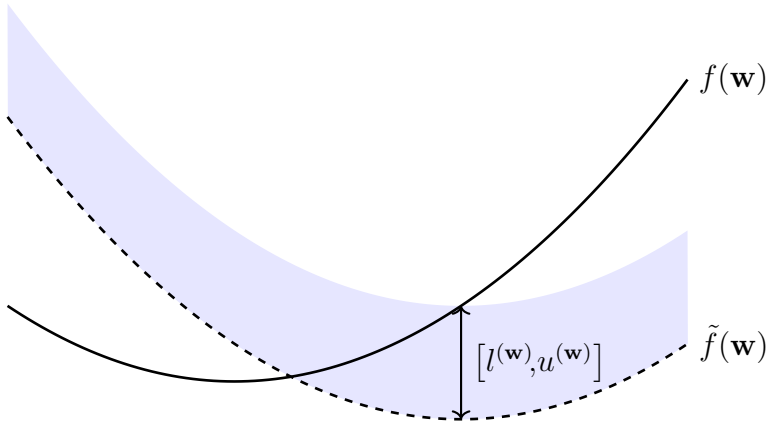


Figura 16: ML methods learn model parameters $\mathbf{w}$ by using some estimate of $f(\mathbf{w})$ for the ultimate performance criterion $\bar{f}(\mathbf{w})$. Using a probabilistic model, one can use $f(\mathbf{w})$ to construct confidence intervals $\left[l^{(\mathbf{w})}, u^{(\mathbf{w})}\right]$ which contain $\bar{f}(\mathbf{w})$ with high probability. The best plausible performance measure for a specific choice $\mathbf{w}$ of model parameters is $\tilde{f}(\mathbf{w}) := l^{(\mathbf{w})}$.

outlier Many ML methods are motivated by the i.i.d. assumption, which interprets punto datis as realizzaziones of i.i.d. RVs with a common distribuzione di probabilità. The i.i.d. assumption is useful for applications where the statistical properties of the dati generation process are stationary (or time-invariant) [**?**]. However, in some applications the dati consists of a majority of regular punto datis that conform with

an i.i.d. assumption as well as a small number of punto datis that have fundamentally different statistical properties compared to the regular punto datis. We refer to a punto dati that substantially deviates from the statistical properties of most punto datis as an outlier. Different methods for outlier detection use different measures for this deviation. Stastistical learning theory studies fundamental limits on the ability to mitigate outliers reliably [**?**, **?**].

overfitting Consider an ML method that uses ERM to learn a ipotesi with the minimo empirical risk on a given training set. Such a method is overfitting the training set if it learns a ipotesi with a small empirical risk on the training set but a significantly larger loss outside the training set.

parameter space The parameter space $\mathcal{W}$ of an ML modello $\mathcal{H}$ is the set of all feasible choices for the model parameters (see Figure 17). Many important ML methods use a modello that is parametrized by vectors of the Euclidean space $\mathbb{R}^d$. Two widely used examples of parametrized modellos are linear models and deep nets. The parameter space is then often a subset $\mathcal{W} \subseteq \mathbb{R}^d$, e.g., all vectors $\mathbf{w} \in \mathbb{R}^d$ with a norma smaller than one.

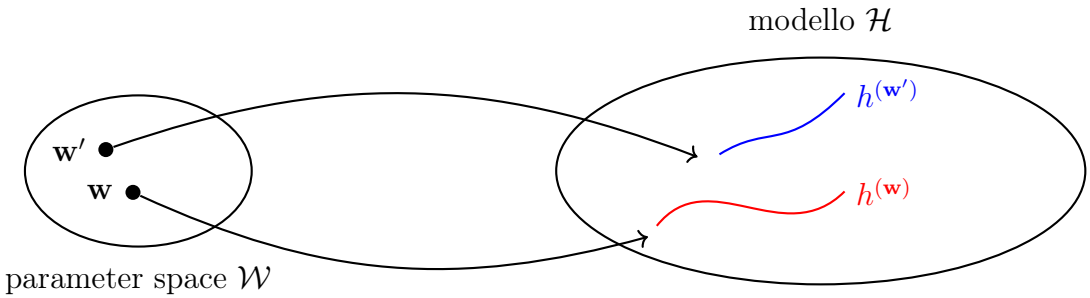


Figura 17: The parameter space $\mathcal{W}$ of an ML modello $\mathcal{H}$ consists of all feasible choices for the model parameters. Each choice $\mathbf{w}$ for the model parameters selects a ipotesi map $h^{(\mathbf{w})} \in \mathcal{H}$.

parameters The parameters of an ML modello are tunable (i.e., learnable or adjustable) quantities that allow us to choose between different ipotesi maps. For example, the linear model $\mathcal{H} := \{h^{(\mathbf{w})} : h^{(\mathbf{w})}(x) = w_1 x + w_2\}$ consists of all ipotesi maps $h^{(\mathbf{w})}(x) = w_1 x + w_2$ with a particular choice for the parameters $\mathbf{w} = \big(w_1, w_2\big)^T \in \mathbb{R}^2$. Another example of parameters is the weights assigned to the connections between neurons of an ANN.

polynomial regression Polynomial regression aims at learning a polynomial ipotesi map to predict a numeric etichetta based on the numeric caratteristicas of a punto dati. For punto datis characterized by a single numeric caratteristica, polynomial regression uses the hypothesis space $\mathcal{H}_d^{(\text{poly})} := \{h(x) = \sum_{j=0}^{d-1} x^j w_j\}$. The quality of a polynomial ipotesi map is measured using the average squared error loss incurred on a set of labeled datapoints (which we refer to as the training set).

positive semi-definite (psd) A (real-valued) symmetric matrix $\mathbf{Q} = \mathbf{Q}^T \in \mathbb{R}^{d \times d}$ is referred to as psd if $\mathbf{x}^T \mathbf{Q} \mathbf{x} \geq 0$ for every vector $\mathbf{x} \in \mathbb{R}^d$. The property of being psd can be extended from matrices to (real-valued) symmetric kernel maps $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ (with $K(\mathbf{x}, \mathbf{x}') = K(\mathbf{x}', \mathbf{x})$) as follows: For any finite set of vettore delle caratteristiches $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)}$, the resulting matrix $\mathbf{Q} \in \mathbb{R}^{m \times m}$ with entries $Q_{r,r'} = K\big(\mathbf{x}^{(r)}, \mathbf{x}^{(r')}\big)$ is psd [**?**].

predictor A predictor is a real-valued ipotesi map. Given a punto dati with caratteristicas $\mathbf{x}$, the value $h(\mathbf{x}) \in \mathbb{R}$ is used as a previsione for the true numeric etichetta $y \in \mathbb{R}$ of the punto dati.

previsione Una previsione è una stima o un'approssimazione di una certa quantità di interesse. Il ML ruota attorno all'apprendimento o all'identificazione di una funzione ipotesi h che riceve in input le caratteristiche $\mathbf{x}$ di un punto dati e restituisce una previsione $\widehat{y} := h(\mathbf{x})$ della sua etichetta y.

principal component analysis (PCA) PCA determines a linear feature map such that the new caratteristicas allow us to reconstruct the original caratteristicas with the minimo reconstruction error [**?**].

privacy funnel The privacy funnel is a method for learning privacy-friendly caratteristicas of punto datis [**?**].

privacy leakage Consider an ML application that processes a dataset $\mathcal{D}$ and delivers some output, such as the previsiones obtained for new punto datis. Privacy leakage arises if the output carries information about a

private (or sensitive) caratteristica of a punto dati (which might be a human) of $\mathcal{D}$. Based on a probabilistic model for the dati generation, we can measure the privacy leakage via the MI between the output and the senstive caratteristica. Another quantitative measure of privacy leakage is DP. The relations between different measures of privacy leakage have been studied in the literature (see [**?**]).

privacy protection Consider some ML method $\mathcal{A}$ that reads in a dataset $\mathcal{D}$ and delivers some output $\mathcal{A}(\mathcal{D})$. The output could be the learned model parameters $\widehat{\mathbf{w}}$ or the previsione $\hat{h}(\mathbf{x})$ obtained for a specific punto dati with caratteristicas $\mathbf{x}$. Many important ML applications involve punto datis representing humans. Each punto dati is characterized by caratteristicas $\mathbf{x}$, potentially a etichetta y, and a sensitive attribute s (e.g., a recent medical diagnosis). Roughly speaking, privacy protection means that it should be impossible to infer, from the output $\mathcal{A}(\mathcal{D})$, any of the sensitive attributes of punto datis in $\mathcal{D}$. Mathematically, privacy protection requires non-invertibility of the map $\mathcal{A}(\mathcal{D})$. In general, just making $\mathcal{A}(\mathcal{D})$ non-invertible is typically insufficient for privacy protection. We need to make $\mathcal{A}(\mathcal{D})$ sufficiently non-invertible.

probabilistic model A probabilistic modello interprets punto datis as realizzaziones of RVs with a joint distribuzione di probabilità. This joint distribuzione di probabilità typically involves parameters which have to be manually chosen or learned via statistical inference methods such as maximum likelihood estimation [**?**].

probabilistic principal component analysis (PPCA) Probabilistic PCA

extends basic PCA by using a probabilistic model for punto datis. The probabilistic model of probabilistic PCA reduces the task of dimensionality reduction to an estimation problem that can be solved using EM methods.

probability We assign a probability value, typically chosen in the interval $[0, 1]$, to each event that might occur in a random experiment [**?**, **?**, **?**, **?**].

probability density function (pdf) The probability density function $p(x)$ of a real-valued RV $x \in \mathbb{R}$ is a particular representation of its distribuzione di probabilità. If the probability density function exists, it can be used to compute the probability that x takes on a value from a (measurable) set $\mathcal{B} \subseteq \mathbb{R}$ via $p(x \in \mathcal{B}) = \int_{\mathcal{B}} p(x')dx'$ [**?**, Ch. 3]. The probability density function of a vector-valued RV $\mathbf{x} \in \mathbb{R}^d$ (if it exists) allows us to compute the probability of $\mathbf{x}$ belonging to a (measurable) region $\mathcal{R}$ via $p(\mathbf{x} \in \mathcal{R}) = \int_{\mathcal{R}} p(\mathbf{x}')dx'_1 \dots dx'_d$ [**?**, Ch. 3].

probability space A probability space is a mathematical modello of a physical process (a random experiment) with an uncertain outcome. Formally, a probability space $\mathcal{P}$ is a triplet $(\Omega, \mathcal{F}, P)$ where

- Ω is a sample space containing all possible elementary outcomes of a random experiment;
- $\mathcal{F}$ is a sigma-algebra, a collection of subsets of Ω (called events) that satisfies certain closure properties under set operations;
- P is a probability measure, a function that assigns a probability $P(\mathcal{A}) \in [0, 1]$ to each event $\mathcal{A} \in \mathcal{F}$. The function must sati-

sfy $P(\Omega) = 1$ and $P\left(\bigcup_{i=1}^{\infty} \mathcal{A}_i\right) = \sum_{i=1}^{\infty} P(\mathcal{A}_i)$ for any countable sequence of pairwise disjoint events $\mathcal{A}_1, \mathcal{A}_2, \ldots$ in $\mathcal{F}$.

Probability spaces provide the foundation for defining RVs and to reason about uncertainty in ML applications [?, ?, ?].

projected gradient descent (projected GD) Consider an ERM-based method that uses a parametrized modello with parameter space $\mathcal{W} \subseteq \mathbb{R}^d$. Even if the objective function of ERM is smooth, we cannot use basic GD, as it does not take into account contraints on the optimization variable (i.e., the model parameters). Projected GD extends basic GD to handle constraints on the optimization variable (i.e., the model parameters). A single iteration of projected GD consists of first taking a gradient step and then projecting the result back onto the parameter space.

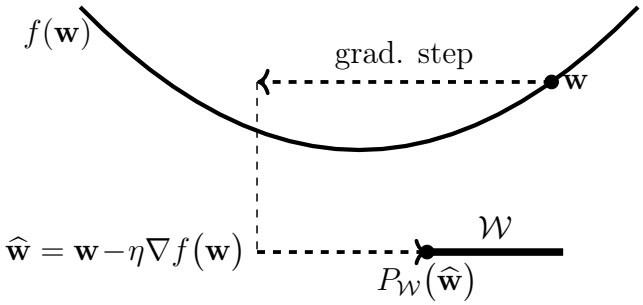


Figura 18: Projected GD augments a basic gradient step with a projection back onto the constraint set $\mathcal{W}$.

projection Consider a subset $\mathcal{W} \subseteq \mathbb{R}^d$ of the d-dimensional Euclidean space.

We define the projection $P_{\mathcal{W}}\big(\mathbf{w}\big)$ of a vector $\mathbf{w} \in \mathbb{R}^d$ onto $\mathcal{W}$ as

$$P_{\mathcal{W}}\big(\mathbf{w}\big) = \operatorname*{argmin}_{\mathbf{w}' \in \mathcal{W}} \|\mathbf{w} - \mathbf{w}'\|_2 \,. \tag{10}$$

In other words, $P_{\mathcal{W}}\big(\mathbf{w}\big)$ is the vector in $\mathcal{W}$ which is closest to $\mathbf{w}$. The projection is only well-defined for subsets $\mathcal{W}$ for which the above minimo exists [**?**].

proximable A convex function for which the proximal operator can be computed efficiently is sometimes referred to as proximable or simple [**?**].

proximal operator Given a convex function $f(\mathbf{w}')$, we define its proximal operator as [**?**, **?**]

$$\mathbf{prox}_{f(\cdot),\rho}(\mathbf{w}) := \operatorname*{argmin}_{\mathbf{w}' \in \mathbb{R}^d} \left[f(\mathbf{w}') + (\rho/2) \|\mathbf{w} - \mathbf{w}'\|_2^2 \right] \text{ with } \rho > 0.$$

As illustrated in Figure 19, evaluating the proximal operator amounts to minimizing a penalized variant of $f(\mathbf{w}')$. The penalty term is the scaled squared Euclidean distance to a given vector $\mathbf{w}$ (which is the input to the proximal operator). The proximal operator can be interpreted as a generalization of the gradient step, which is defined for a smooth convex function $f(\mathbf{w}')$. Indeed, taking a gradient step with step size η at the current vector $\mathbf{w}$ is the same as applying the proximal operator of the function $\tilde{f}(\mathbf{w}') = \big(\nabla f(\mathbf{w})\big)^T(\mathbf{w}' - \mathbf{w})$ and using $\rho = 1/\eta$.

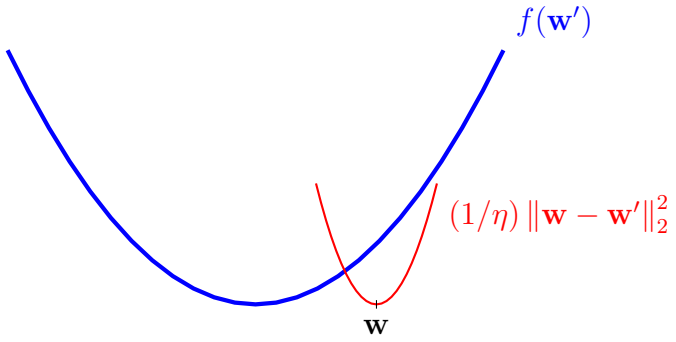


Figura 19: A generalized gradient step updates a vector $\mathbf{w}$ by minimizing a penalized version of the function $f(\cdot)$. The penalty term is the scaled squared Euclidean distance between the optimization variable $\mathbf{w}'$ and the given vector $\mathbf{w}$.

punto dati Si definisce punto dati qualsiasi oggetto che contiene o veicola la informazione [**?**]. Possono essere punti dati studenti, segnali radio, alberi, foreste, immagini, RVs, numeri reali o proteine. I punti dati si caratterizzano utilizzando due tipologie di proprietà. Un tipo di proprietà sono le caratteristiche. Caratteristiche sono proprietà di un punto dati che possono essere misurate o calcolate in modo automatico. Un secondo tipo di proprietà viene definita etichetta. L' etichetta di un punto dati rappresenta un'informazione di livello superiore (o una quantità di interesse). A differenza delle caratteristiche, la determinazione dell' etichetta di un punto dati trichiede tipicamente l'intervento di esperti umani (esperti del dominio). In termini generali, l'obiettivo del ML è quello di prevedere l' etichetta di un punto dati basandosi esclusivamente sulle sue caratteristiche.

quadratic function A function $f : \mathbb{R}^d \to \mathbb{R}$ of the form

$$f(\mathbf{w}) = \mathbf{w}^T \mathbf{Q} \mathbf{w} + \mathbf{q}^T \mathbf{w} + a,$$

with some matrix $\mathbf{Q} \in \mathbb{R}^{d \times d}$, vector $\mathbf{q} \in \mathbb{R}^d$, and scalar $a \in \mathbb{R}$.

random forest A random forest is a set (or ensemble) of different decision trees. Each of these decision trees is obtained by fitting a perturbed copy of the original dataset.

random variable (RV) An RV is a function that maps from a probability space $\mathcal{P}$ to a value space [**?**, **?**]. The probability space consists of elementary events and is equipped with a probability measure that assigns probabilities to subsets of $\mathcal{P}$. Different types of RVs include

- binary RVs, which map elementary events to a set of two distinct values, such as $\{-1, 1\}$ or $\{\text{cat}, \text{no cat}\}$;
- real-valued RVs, which take values in the real numbers $\mathbb{R}$;
- vector-valued RVs, which map elementary events to the Euclidean space $\mathbb{R}^d$.

Probability theory uses the concept of measurable spaces to rigorously define and study the properties of (large) collections of RVs [**?**].

realizzazione Si consideri una RV x che associa a ciascun elemento (ossia, esito o evento elementare) $\omega \in \mathcal{P}$ di uno probability space $\mathcal{P}$ ad un elemento a di uno spazio misurabile $\mathcal{N}$ [**?**], [**?**], [**?**]. Una realizzazione di x è un qualunque elemento $a' \in \mathcal{N}$ per il quale si ha $x(\omega') = a'$ per

qualche $\omega' \in \mathcal{P}$.

Si veda anche: RV, probability space.

rectified linear unit (ReLU) The ReLU is a popular choice for the activation function of a neuron within an ANN. It is defined as $\sigma(z) = \max\{0, z\}$, with z being the weighted input of the artificial neuron.

regression Regression problems revolve around the prediction of a numeric etichetta solely from the caratteristicas of a punto dati [**?**, Ch. 2].

regret The regret of a ipotesi h relative to another ipotesi h', which serves as a baseline, is the difference between the loss incurred by h and the loss incurred by h' [**?**]. The baseline ipotesi h' is also referred to as an expert.

regularization A key challenge of modern ML applications is that they often use large modellos, which have an effective dimension in the order of billions. Training a high-dimensional modello using basic ERM-based methods is prone to overfitting: the learned ipotesi performs well on the training set but poorly outside the training set. Regularization refers to modifications of a given instance of ERM in order to avoid overfitting, i.e., to ensure that the learned ipotesi performs not much worse outside the training set. There are three routes for implementing regularization:

1) Modello pruning: We prune the original modello $\mathcal{H}$ to obtain a smaller modello $\mathcal{H}'$. For a parametric modello, the pruning can be implemented via constraints on the model parameters

(such as $w_1 \in [0.4, 0.6]$ for the weight of caratteristica x_1 in linear regression).

2) Loss penalization: We modify the objective function of ERM by adding a penalty term to the training error. The penalty term estimates how much larger the expected loss (or risk) is compared to the average loss on the training set.

3) Data augmentation: We can enlarge the training set $\mathcal{D}$ by adding perturbed copies of the original punto datis in $\mathcal{D}$. One example for such a perturbation is to add the realizzazione of an RV to the vettore delle caratteristiche of a punto dati.

Figure 20 illustrates the above three routes to regularization. These routes are closely related and sometimes fully equivalent: data augmentation using Gaussian RVs to perturb the vettore delle caratteristiches in the training set of linear regression has the same effect as adding the penalty $\lambda \|\mathbf{w}\|_2^2$ to the training error (which is nothing but ridge regression). The decision on which route to use for regularization can be based on the available computational infrastructure. For example, it might be much easier to implement data augmentation than modello pruning.

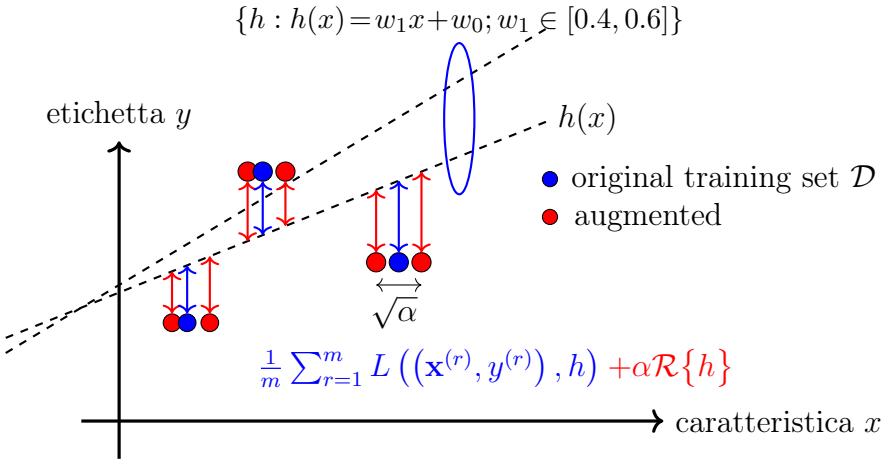


Figura 20: Three approaches to regularization: 1) data augmentation; 2) loss penalization; and 3) modello pruning (via constraints on model parameters).

regularized empirical risk minimization (RERM) Basic ERM learns a ipotesi (or trains a modello) $h \in \mathcal{H}$ based solely on the empirical risk $\widehat{L}\big(h|\mathcal{D}\big)$ incurred on a training set $\mathcal{D}$. To make ERM less prone to overfitting, we can implement regularization by including a (scaled) regularizer $\mathcal{R}\big\{h\big\}$ in the learning objective. This leads to regularized empirical risk minimization (RERM),

$$\hat{h} \in \operatorname*{argmin}_{h \in \mathcal{H}} \widehat{L}\big(h|\mathcal{D}\big) + \alpha \mathcal{R}\big\{h\big\}. \tag{11}$$

The parameter $\alpha \geq 0$ controls the regularization strength. For $\alpha = 0$, we recover standard ERM without regularization. As α increases, the learned ipotesi is increasingly biased toward small values of $\mathcal{R}\big\{h\big\}$. The component $\alpha \mathcal{R}\big\{h\big\}$ in the objective function of (11) can be intuitively understood as a surrogate for the increased average loss that may occur when predicting etichettas for punto datis outside the training set. This intuition can be made precise in various ways. For example, consider a linear model trained using squared error loss and the regularizer $\mathcal{R}\big\{h\big\} = \|\mathbf{w}\|_2^2$. In this setting, $\alpha \mathcal{R}\big\{h\big\}$ corresponds to the expected increase in loss caused by adding Gaussian RVs to the vettore delle caratteristiches in the training set [**?**, Ch. 3]. A principled construction for the regularizer $\mathcal{R}\big\{h\big\}$ arises from approximate upper bounds on the generalization error. The resulting RERM instance is known as SRM [**?**, Sec. 7.2].

regularized loss minimization (RLM) See RERM.

regularizer A regularizer assigns each ipotesi h from a hypothesis space $\mathcal{H}$ a quantitative measure $\mathcal{R}\big\{h\big\}$ for how much its previsione error on a

training set might differ from its previsione errors on punto datis outside the training set. Ridge regression uses the regularizer $\mathcal{R}\{h\} := \|\mathbf{w}\|_2^2$ for linear ipotesi maps $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [**?**, Ch. 3]. Lasso uses the regularizer $\mathcal{R}\{h\} := \|\mathbf{w}\|_1$ for linear ipotesi maps $h^{(\mathbf{w})}(\mathbf{x}) := \mathbf{w}^T\mathbf{x}$ [**?**, Ch. 3].

Rényi divergence The Rényi divergence measures the (dis)similarity between two distribuzione di probabilitàs [**?**].

reward A reward refers to some observed (or measured) quantity that allows us to estimate the loss incurred by the previsione (or decision) of a ipotesi $h(\mathbf{x})$. For example, in an ML application to self-driving vehicles, $h(\mathbf{x})$ could represent the current steering direction of a vehicle. We could construct a reward from the measurements of a collision sensor that indicate if the vehicle is moving towards an obstacle. We define a low reward for the steering direction $h(\mathbf{x})$ if the vehicle moves dangerously towards an obstacle.

ridge regression Ridge regression learns the weights $\mathbf{w}$ of a linear ipotesi map $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. The quality of a particular choice for the model parameters $\mathbf{w}$ is measured by the sum of two components. The first component is the average squared error loss incurred by $h^{(\mathbf{w})}$ on a set of labeled datapoints (i.e., the training set). The second component is the scaled squared Euclidean norma $\alpha\|\mathbf{w}\|_2^2$ with a regularization parameter $\alpha > 0$. Adding $\alpha\|\mathbf{w}\|_2^2$ to the average squared error loss is equivalent to replacing each original punto datis by the realizzazione of (infinitely many) i.i.d. RVs centered around these punto datis (see regularization).

riduzione della dimensionalità Con riduzione della dimensionalità ci si riferisce a metodi che apprendono una trasformazione $h : \mathbb{R}^d \to \mathbb{R}^{d'}$ di un insieme (tipicamente ampio) di caratteristiche grezze$x_1, \ldots, x_d$ in un insieme più contenuto di caratteristiche significative $z_1, \ldots, z_{d'}$. L'utilizzo di un numero ridotto di caratteristiche comporta molteplici vantaggi:

- Vantaggio statistico: in genere si riduce il rischio di overfitting, poiché la riduzione del numero di caratteristiche comporta spesso una diminuzione della effective dimension di un modello.
- Vantaggio computazionale: un numero inferiore di caratteristiche implica un minor carico computazionale durante l'addestramento dei modelli di ML. Ad esempio, i metodi di linear regression richiedono l'inversione di una matrice la cui dimensione dipende dal numero di caratteristiche.
- **Visualizzazione:** la riduzione della dimensionalità riveste inoltre un ruolo essenziale nella visualizzazione dei dati. Si può, ad esempio, apprendere una trasformazione che restituisce due caratteristiche z_1, z_2 utilizzabili come coordinate in uno scatterplot.La Fig. 21 mostra lo scatterplot di cifre scritte a mano posizionate secondo le caratteristiche trasformate. In questo caso, i punti dati sono originariamente rappresentati da un ampio numero di valori in una scala di grigi (uno per ciascun pixel).

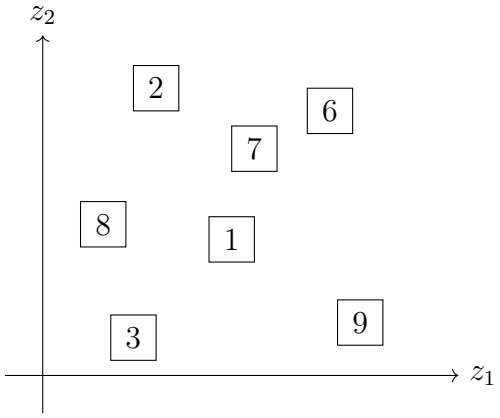


Figura 21: Esempio di riduzione della dimensionalità: dati di immagini ad alta dimensionalità (ad esempio, immagini ad alta risoluzione di cifre scritte a mano) proiettati in 2D utilizzando caratteristiche apprese (z_1, z_2) e visualizzati in un scatterplot .

Si veda anche: caratteristica, overfitting, effective dimension, modello, ML, linear regression, dati, scatterplot, punto dati.

risk Consider a ipotesi h used to predict the etichetta y of a punto dati based on its caratteristicas $\mathbf{x}$. We measure the quality of a particular previsione using a loss function $L\left((\mathbf{x}, y), h\right)$. If we interpret punto datis as the realizzaziones of i.i.d. RVs, also the $L\left((\mathbf{x}, y), h\right)$ becomes the realizzazione of an RV. The i.i.d. assumption allows us to define the risk of a ipotesi as the expected loss $\mathbb{E}\big\{L\left((\mathbf{x}, y), h\right)\big\}$. Note that the risk of h depends on both the specific choice for the loss function and the distribuzione di probabilità of the punto datis.

sample A finite sequence (or list) of punto datis $\mathbf{z}^{(1)}, \ldots, \mathbf{z}^{(m)}$ that is obtained or interpreted as the realizzazione of m i.i.d. RVs with a common distribuzione di probabilità $p(\mathbf{z})$. The length m of the sequence is referred to as the sample size.

sample covariance matrix The sample covariance matrix $\widehat{\mathbf{\Sigma}} \in \mathbb{R}^{d \times d}$ for a given set of vettore delle caratteristiches $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$ is defined as

$$\widehat{\mathbf{\Sigma}} = (1/m) \sum_{r=1}^{m} (\mathbf{x}^{(r)} - \widehat{\mathbf{m}})(\mathbf{x}^{(r)} - \widehat{\mathbf{m}})^T.$$

Here, we use the sample mean $\widehat{\mathbf{m}}$.

sample mean The sample mean $\mathbf{m} \in \mathbb{R}^d$ for a given dataset, with vettore delle caratteristiches $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(m)} \in \mathbb{R}^d$, is defined as

$$\mathbf{m} = (1/m) \sum_{r=1}^{m} \mathbf{x}^{(r)}.$$

sample size The number of individual punto datis contained in a dataset.

scatterplot A visualization technique that depicts punto datis by markers in a two-dimensional plane. Figure 22 depicts an example of a scatterplot.

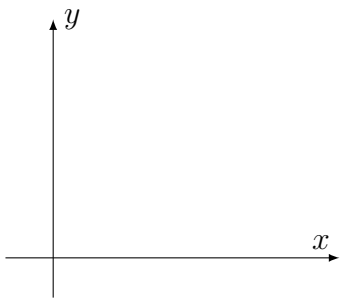


Figura 22: A scatterplot of some punto datis representing daily weather conditions in Finland. Each punto dati is characterized by its minimo daytime temperature x as the caratteristica and its massimo daytime temperature y as the etichetta. The temperatures have been measured at the FMI weather station Helsinki Kaisaniemi during 1.9.2024 - 28.10.2024.

semi-supervised learning (SSL) SSL methods use unlabeled datapoints to support the learning of a ipotesi from labeled datapoints [**?**]. This approach is particularly useful for ML applications that offer a large amount of unlabeled datapoints, but only a limited number of labeled datapoints.

sensitive attribute ML revolves around learning a ipotesi map that allows us to predict the etichetta of a punto dati from its caratteristicas. In some applications, we must ensure that the output delivered by an ML system does not allow us to infer sensitive attributes of a punto dati. Which part of a punto dati is considered a sensitive attribute is a design choice that varies across different application domains.

similarity graph Some ML applications generate punto datis that are related by a domain-specific notion of similarity. These similarities

can be represented conveniently using a similarity graph $\mathcal{G} = \big(\mathcal{V} := \{1, \ldots, m\}, \mathcal{E}\big)$. The node $r \in \mathcal{V}$ represents the r-th punto dati. Two nodes are connected by an undirected edge if the corresponding punto datis are similar.

singular value decomposition (SVD) The SVD for a matrix $\mathbf{A} \in \mathbb{R}^{m \times d}$ is a factorization of the form

$$\mathbf{A} = \mathbf{V}\mathbf{\Lambda}\mathbf{U}^T,$$

with orthonormal matrices $\mathbf{V} \in \mathbb{R}^{m \times m}$ and $\mathbf{U} \in \mathbb{R}^{d \times d}$ [**?**]. The matrix $\mathbf{\Lambda} \in \mathbb{R}^{m \times d}$ is only non-zero along the main diagonal, whose entries $\Lambda_{j,j}$ are non-negative and referred to as singular values.

smooth A real-valued function $f : \mathbb{R}^d \to \mathbb{R}$ is smooth if it is differenziabile and its gradiente $\nabla f(\mathbf{w})$ is continuous at all $\mathbf{w} \in \mathbb{R}^d$ [**?**, **?**]. A smooth function f is referred to as β-smooth if the gradiente $\nabla f(\mathbf{w})$ is Lipschitz continuous with Lipschitz constant β, i.e.,

$$\|\nabla f(\mathbf{w}) - \nabla f(\mathbf{w}')\| \leq \beta \|\mathbf{w} - \mathbf{w}'\|, \text{ for any } \mathbf{w}, \mathbf{w}' \in \mathbb{R}^d.$$

The constant β quantifies the amount of smoothness of the function f: the smaller the β, the smoother f is. Optimization problems with a smooth objective function can be solved effectively by gradient-based methods. Indeed, gradient-based methods approximate the objective function locally around a current choice $\mathbf{w}$ using its gradiente. This approximation works well if the gradiente does not change too rapidly. We can make this informal claim precise by studying the effect of a single gradient step with step size $\eta = 1/\beta$ (see Figure 23).

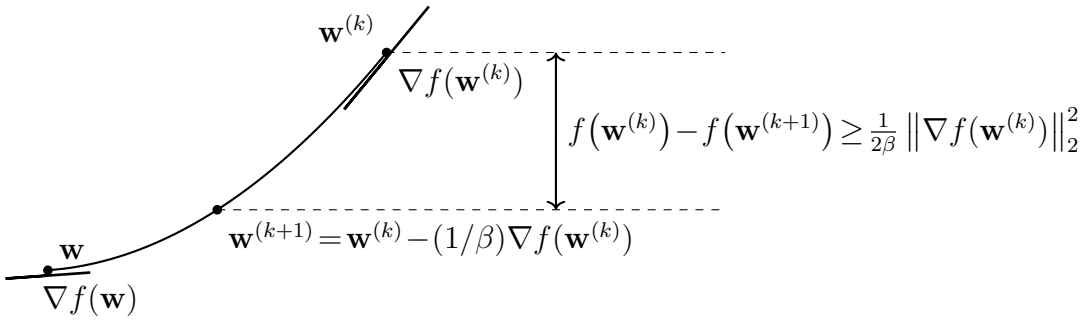
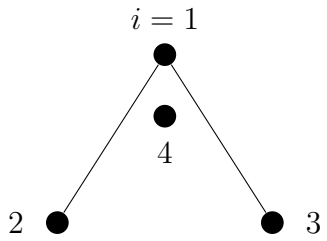


Figura 23: Consider an objective function $f(\mathbf{w})$ that is β-smooth. Taking a gradient step, with step size $\eta = 1/\beta$, decreases the objective by at least $\frac{1}{2\beta} \big\|\nabla f(\mathbf{w}^{(k)})\big\|_2^2$ [?, ?, ?]. Note that the step size $\eta = 1/\beta$ becomes larger for smaller β. Thus, for smoother objective functions (i.e., those with smaller β), we can take larger steps.

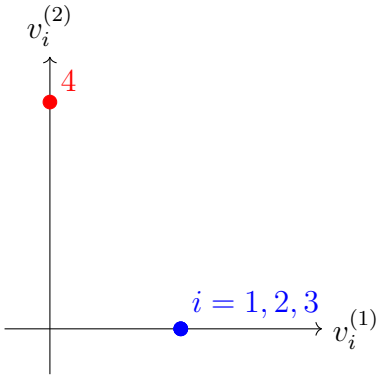
soft clustering Soft clustering refers to the task of partitioning a given set of punto datis into (a few) overlapping clusters. Each punto dati is assigned to several different clusters with varying degrees of belonging. Soft clustering methods determine the degree of belonging (or soft cluster assignment) for each punto dati and each cluster. A principled approach to soft clustering is by interpreting punto datis as i.i.d. realizzaziones of a GMM. We then obtain a natural choice for the degree of belonging as the conditional probability of a punto dati belonging to a specific mixture component.

spectral clustering Spectral clustering is a particular instance of graph clustering, i.e., it clusters punto datis represented as the nodes $i = 1, \ldots, n$ of a graph $\mathcal{G}$. Spectral clustering uses the eigenvectors of the

Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$ to construct vettore delle caratteristiches $\mathbf{x}^{(i)} \in \mathbb{R}^d$ for each node (i.e., for each punto dati) $i = 1, \ldots, n$. We can feed these vettore delle caratteristiches into Euclidean space-based clustering methods, such as k-means or soft clustering via GMM. Roughly speaking, the vettore delle caratteristiches of nodes belonging to a well-connected subset (or cluster) of nodes in $\mathcal{G}$ are located nearby in the Euclidean space $\mathbb{R}^d$ (see Figure 24).



$$\mathbf{L}^{(\mathcal{G})} = \begin{pmatrix} 2 & -1 & -1 & 0 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$$



$$\mathbf{V} = \left(\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \mathbf{v}^{(3)}, \mathbf{v}^{(4)}\right)$$

$$\mathbf{v}^{(1)} = \frac{1}{\sqrt{3}} \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \end{pmatrix}, \; \mathbf{v}^{(2)} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Figura 24: **Top.** Left: An undirected graph $\mathcal{G}$ with four nodes $i = 1, 2, 3, 4$, each representing a punto dati. Right: The Laplacian matrix $\mathbf{L}^{(\mathcal{G})} \in \mathbb{R}^{4\times4}$ and its EVD. **Bottom.** Left: A scatterplot of punto datis using the vettore delle caratteristiches $\mathbf{x}^{(i)} = \left(v_i^{(1)}, v_i^{(2)}\right)^T$. Right: Two eigenvectors $\mathbf{v}^{(1)}, \mathbf{v}^{(2)} \in \mathbb{R}^d$ corresponding to the eigenvalue $\lambda = 0$ of the Laplacian matrix $\mathbf{L}^{(\mathcal{G})}$.

spectrogram A spectrogram represents the time-frequency distribution of the energy of a time signal $x(t)$. Intuitively, it quantifies the amount of signal energy present within a specific time segment $[t_1, t_2] \subseteq \mathbb{R}$ and frequency interval $[f_1, f_2] \subseteq \mathbb{R}$. Formally, the spectrogram of a signal is defined as the squared magnitude of its short-time Fourier transform (STFT) [**?**]. Figure 25 depicts a time signal along with its spectrogram.

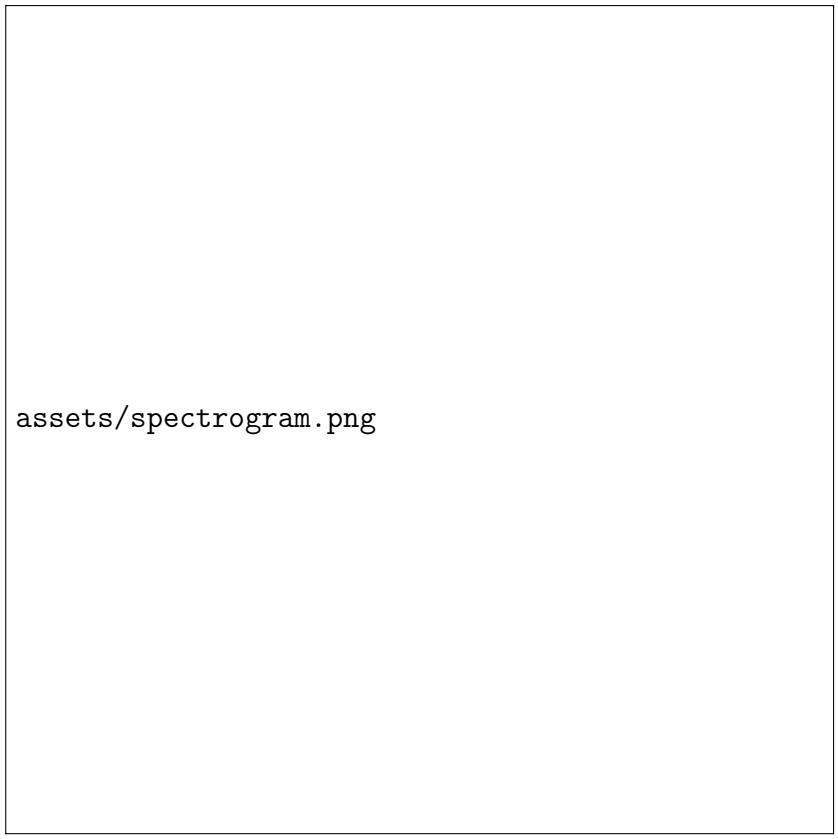


Figura 25: Left: A time signal consisting of two modulated Gaussian pulses. Right: An intensity plot of the spectrogram.

The intensity plot of its spectrogram can serve as an image of a signal.

A simple recipe for audio signal classification is to feed this signal image into deep nets originally developed for image classification and object detection [**?**]. It is worth noting that, beyond the spectrogram, several alternative representations exist for the time-frequency distribution of signal energy [**?**, **?**].

squared error loss The squared error loss measures the previsione error of a ipotesi h when predicting a numeric etichetta $y \in \mathbb{R}$ from the caratteristicas $\mathbf{x}$ of a punto dati. It is defined as

$$L\left((\mathbf{x}, y), h\right) := \big(y - \underbrace{h(\mathbf{x})}_{=\hat{y}}\big)^2.$$

stability Stability is a desirable property of a ML method $\mathcal{A}$ that maps a dataset $\mathcal{D}$ (e.g., a training set) to an output $\mathcal{A}(\mathcal{D})$, such as learned model parameters or the previsione for a specific punto dati. Intuitively, $\mathcal{A}$ is stable if small changes in the input dataset $\mathcal{D}$ lead to small changes in the output $\mathcal{A}(\mathcal{D})$. Several formal notions of stability exist that enable bounds on the generalization error or risk of the method; see [**?**, Ch. 13]. To build intuition, consider the three datasets depicted in Fig. **??**, each of which is equally likely under the same dati-generating distribuzione di probabilità. Since the optimal model parameters are determined by this underlying distribuzione di probabilità, an accurate ML method $\mathcal{A}$ should return the same (or very similar) output $\mathcal{A}(\mathcal{D})$ for all three dataset. In other words, any useful $\mathcal{A}$ must be robust to variability in sample realizzaziones from the same distribuzione di probabilità, i.e., it must be stable.

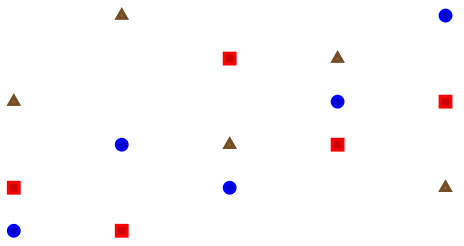


Figura 26: Three datasets $\mathcal{D}^{(*)}$, $\mathcal{D}^{(\square)}$, and $\mathcal{D}^{(\triangle)}$, each sampled independently from the same dati-generating distribuzione di probabilità. A stable ML method should return similar outputs when trained on any of these datasets.

statistical aspects By statistical aspects of an ML method, we refer to (properties of) the distribuzione di probabilità of its output under a probabilistic model for the dati fed into the method.

step size See learning rate.

stochastic block model (SBM) The stochastic block modello is a probabilistic generative modello for an undirected graph $\mathcal{G} = \big(\mathcal{V}, \mathcal{E}\big)$ with a given set of nodes $\mathcal{V}$ [?]. In its most basic variant, the stochastic block modello generates a graph by first randomly assigning each node $i \in \mathcal{V}$ to a cluster index $c_i \in \{1, \ldots, k\}$. A pair of different nodes in the graph is connected by an edge with probability $p_{i,i'}$ that depends solely on the etichettas $c_i, c_{i'}$. The presence of edges between different pairs of nodes is statistically independent.

stochastic gradient descent (SGD) Stochastic GD is obtained from GD by replacing the gradiente of the objective function with a stochastic approximation. A main application of stochastic GD is to train a parametrized modello via ERM on a training set $\mathcal{D}$ that is either very large or not readily available (e.g., when punto datis are stored in a database distributed all over the planet). To evaluate the gradiente of the empirical risk (as a function of the model parameters $\mathbf{w}$), we need to compute a sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over all punto datis in the training set. We obtain a stochastic approximation to the gradiente by replacing the sum $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ with a sum $\sum_{r \in \mathcal{B}} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ over a randomly chosen subset $\mathcal{B} \subseteq \{1, \ldots, m\}$ (see Figure **??**). We often refer to these randomly chosen punto datis as a batch. The batch size $|\mathcal{B}|$ is an important parameter of stochastic GD. Stochastic GD with $|\mathcal{B}| > 1$ is referred to as mini-batch stochastic GD [**?**].

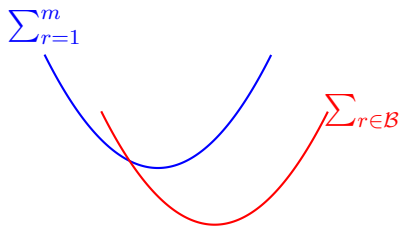


Figura 27: Stochastic GD for ERM approximates the gradiente $\sum_{r=1}^{m} \nabla_{\mathbf{w}} L\left(\mathbf{z}^{(r)}, \mathbf{w}\right)$ by replacing the sum over all punto datis in the training set (indexed by $r = 1, \ldots, m$) with a sum over a randomly chosen subset $\mathcal{B} \subseteq \{1, \ldots, m\}$.

stopping criterion Many ML methods use iterative algorithms that con-

struct a sequence of model parameters (such as the weights of a linear map or the weights of an ANN). These parameters (hopefully) converge to an optimal choice for the model parameters. In practice, given finite computational resources, we need to stop iterating after a finite number of repetitions. A stopping criterion is any well-defined condition required for stopping the iteration.

strongly convex A continuously differenziabile real-valued function $f(\mathbf{x})$ is strongly convex with coefficient σ if $f(\mathbf{y}) \geq f(\mathbf{x}) + \nabla f(\mathbf{x})^T(\mathbf{y} - \mathbf{x}) + (\sigma/2)\left\|\mathbf{y} - \mathbf{x}\right\|_2^2$ [**?**], [**?**, Sec. B.1.1].

structural risk minimization (SRM) Structural risk minimization (SRM) is an instance of RERM, which which the modello $\mathcal{H}$ can be expressed as a countable union of sub-models: $\mathcal{H} = \bigcup_{n=1}^{\infty} \mathcal{H}^{(n)}$. Each sub-model $\mathcal{H}^{(n)}$ permits the derivation of an approximate upper bound on the generalization error incurred when applying ERM to train $\mathcal{H}^{(n)}$. These individual bounds—one for each sub-model—are then combined to form a regularizer used in the RERM objective. These approximate upper bounds (one for each $\mathcal{H}^{(n)}$) are then combined to construct a regularizer for RERM [**?**, Sec. 7.2].

subgradient For a real-valued function $f : \mathbb{R}^d \to \mathbb{R} : \mathbf{w} \mapsto f(\mathbf{w})$, a vector $\mathbf{a}$ such that $f(\mathbf{w}) \geq f(\mathbf{w}') + \left(\mathbf{w} - \mathbf{w}'\right)^T \mathbf{a}$ is referred to as a subgradient of f at $\mathbf{w}'$ [**?**, **?**].

subgradient descent Subgradient descent is a generalization of GD that does not require differentiability of the function to be minimized. This generalization is obtained by replacing the concept of a gradiente with

that of a subgradient. Similar to gradientes, also subgradients allow us to construct local approximations of an objective function. The objective function might be the empirical risk $\widehat{L}\big(h^{(\mathbf{w})}\big|\mathcal{D}\big)$ viewed as a function of the model parameters $\mathbf{w}$ that select a ipotesi $h^{(\mathbf{w})} \in \mathcal{H}$.

support vector machine (SVM) The SVM is a binary classification method that learns a linear ipotesi map. Thus, like linear regression and logistic regression, it is also an instance of ERM for the linear model. However, the SVM uses a different loss function from the one used in those methods. As illustrated in Figure **??**, it aims to maximally separate punto datis from the two different classes in the feature space (i.e., massimo margin principle). Maximizing this separation is equivalent to minimizing a regularized variant of the hinge loss (5) [**?**, **?**, **?**].

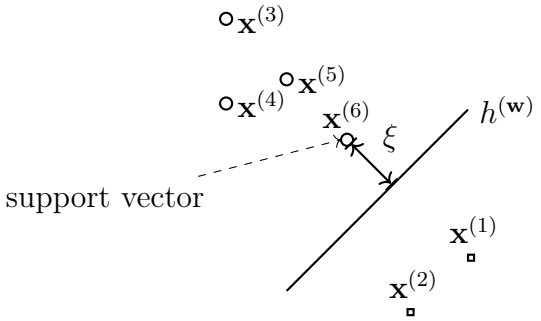


Figura 28: The SVM learns a ipotesi (or classifier) $h^{(\mathbf{w})}$ with minimal average soft-margin hinge loss. Minimizing this loss is equivalent to maximizing the margin ξ between the decision boundary of $h^{(\mathbf{w})}$ and each class of the training set.

The above basic variant of SVM is only useful if the punto datis from

different categories can be (approximately) linearly separated. For an ML application where the categories are not derived from a kernel.

test set A set of punto datis that have been used neither to train a modello (e.g., via ERM) nor in a validation set to choose between different modellos.

total variation See GTV.

training error The average loss of a ipotesi when predicting the etichettas of the punto datis in a training set. We sometimes refer by training error also to minimal average loss which is achieved by a solution of ERM.

training set A training set is a dataset $\mathcal{D}$ which consists of some punto datis used in ERM to learn a ipotesi $\hat{h}$. The average loss of $\hat{h}$ on the training set is referred to as the training error. The comparison of the training error with the validation error of $\hat{h}$ allows us to diagnose the ML method and informs how to improve the validation error (e.g., using a different hypothesis space or collecting more punto datis) [**?**, Sec. 6.6].

transparency Transparency is a fundamental requirement for trustworthy AI [**?**]. In the context of ML methods, transparency is often used interchangeably with explainability [**?**, **?**]. However, in the broader scope of AI systems, transparency extends beyond explainability and includes providing information about the system's limitations, reliability, and intended use. In medical diagnosis systems, transparency requires disclosing the confidence level for the previsiones delivered by

a trained modello. In credit scoring, AI-based lending decisions should be accompanied by explanations of contributing factors, such as income level or credit history. These explanations allow humans (e.g., a loan applicant) to understand and contest automated decisions. Some ML methods inherently offer transparency. For example, logistic regression provides a quantitative measure of classification reliability through the value $|h(\mathbf{x})|$. Decision trees are another example, as they allow human-readable decision rules [**?**]. Transparency also requires a clear indication when a user is engaging with an AI system. For example, AI-powered chatbots should notify users that they are interacting with an automated system rather than a human. Furthermore, transparency encompasses comprehensive documentation detailing the purpose and design choices underlying the AI system. For instance, modello datasheets [**?**] and AI system cards [**?**] help practitioners understand the intended use cases and limitations of an AI system [**?**].

trustworthy artificial intelligence (trustworthy AI) Besides the computational aspects and statistical aspects, a third main design aspect of ML methods is their trustworthiness [**?**]. The EU has put forward seven key requirements (KRs) for trustworthy AI (that typically build on ML methods) [**?**]:

1) KR1 - Human agency and oversight;
2) KR2 - Technical robustness and safety;
3) KR3 - Privacy and data governance;
4) KR4 - Transparency;

5) KR5 - Diversity, non-discrimination and fairness;

6) KR6 - Societal and environmental well-being;

7) KR7 - Accountability.

uncertainty Uncertainty refers to the degree of confidence—or lack thereof—associated with a quantity such as a model prediction, parameter estimate, or observed data point. In ML, uncertainty arises from various sources, including noisy data, limited training samples, or ambiguity in model assumptions. Probability theory offers a principled framework for representing and quantifying such uncertainty.

underfitting Consider an ML method that uses ERM to learn a ipotesi with the minimo empirical risk on a given training set. Such a method is underfitting the training set if it is not able to learn a ipotesi with a sufficiently small empirical risk on the training set. If a method is underfitting, it will typically also not be able to learn a ipotesi with a small risk.

upper confidence bound (UCB) Consider a ML application that requires selecting, at each time step k, an action a_k from a finite set of alternatives $\mathcal{A}$. The utility of selecting action a_k is quantified by a numeric reward signal $r^{(a_k)}$. A widely used probabilistic model for this type of sequential decision-making problem is the stochastic multi-armed bandit setting [**?**]. In this model, the reward $r^{(a)}$ is viewed as the realizzazione of a RV with unknown mean $\mu^{(a)}$. Ideally, we would always choose the action with the largest expected reward $\mu^{(a)}$, but these means are unknown and must

be estimated from observed dati. Simply choosing the action with the largest estimate $\widehat{\mu}^{(a)}$ can lead to suboptimal outcomes due to estimation uncertainty. The UCB strategy addresses this by selecting actions not only based on their estimated means but also by incorporating a term that reflects the uncertainty in these estimates—favouring actions with high potential reward and high uncertainty. Theoretical guarantees for the performance of UCB strategies,including logarithmic regret bounds, are established in [**?**].

validation Consider a ipotesi $\hat{h}$ that has been learned via some ML method, e.g., by solving ERM on a training set $\mathcal{D}$. Validation refers to the practice of evaluating the loss incurred by the ipotesi $\hat{h}$ on a set of punto datis that are not contained in the training set $\mathcal{D}$.

validation error Consider a ipotesi $\hat{h}$ which is obtained by some ML method, e.g., using ERM on a training set. The average loss of $\hat{h}$ on a validation set, which is different from the training set, is referred to as the validation error.

validation set A set of punto datis used to estimate the risk of a ipotesi $\hat{h}$ that has been learned by some ML method (e.g., solving ERM). The average loss of $\hat{h}$ on the validation set is referred to as the validation error and can be used to diagnose an ML method (see [**?**, Sec. 6.6]). The comparison between training error and validation error can inform directions for improvement of the ML method (such as using a different hypothesis space).

valore atteso Si consideri un vettore delle caratteristiche numeriche $\mathbf{x} \in \mathbb{R}^d$ che interpretiamo come una realizzazione di una RV con una distribuzione di probabilità $p(\mathbf{x})$. Il valore atteso di $\mathbf{x}$ è definito come l'integrale $\mathbb{E}\{\mathbf{x}\} := \int \mathbf{x} p(\mathbf{x})$. Si noti che il valore atteso è definito solo se tale integrale esiste, ovvero se la RV è integrabile [**?**, **?**, **?**]

Vapnik–Chervonenkis dimension (VC dimension) The VC dimension of an infinite hypothesis space is a widely-used measure for its size. We refer to the literature (see [**?**]) for a precise definition of VC dimension as well as a discussion of its basic properties and use in ML.

variance The variance of a real-valued RV x is defined as the valore atteso $\mathbb{E}\big\{\big(x - \mathbb{E}\{x\}\big)^2\big\}$ of the squared difference between x and its valore atteso $\mathbb{E}\{x\}$. We extend this definition to vector-valued RVs $\mathbf{x}$ as $\mathbb{E}\big\{\big\|\mathbf{x} - \mathbb{E}\{\mathbf{x}\}\big\|_2^2\big\}$.

vertical federated learning (vertical FL) Vertical FL uses local datasets that are constituted by the same punto datis but characterizing them with different caratteristicas [**?**]. For example, different healthcare providers might all contain information about the same population of patients. However, different healthcare providers collect different measurements (e.g., blood values, electrocardiography, lung X-ray) for the same patients.

vettore delle caratteristiche Il vettore delle caratteristiche si riferisce a un vettore $\mathbf{x} = \big(x_1, \ldots, x_d\big)^T$ i cui elementi sono singole caratteristiche $x_1, \ldots, x_d$. Molti metodi di ML utilizzano vettori delle caratteristiche appartenenti ad uno Euclidean space di dimensione finita $\mathbb{R}^d$. Tuttavia,

per alcuni metodi di ML, risulta preferibile utilizzare vettori delle caratteristiche che risiedono in uno spazio vettoriale di dimensione infinita (si veda, ad esempio, kernel method).

weights Consider a parametrized hypothesis space $\mathcal{H}$. We use the term weights for numeric model parameters that are used to scale caratteristicas or their transformations in order to compute $h^{(\mathbf{w})} \in \mathcal{H}$. A linear model uses weights $\mathbf{w} = \big(w_1, \ldots, w_d\big)^T$ to compute the linear combination $h^{(\mathbf{w})}(\mathbf{x}) = \mathbf{w}^T\mathbf{x}$. Weights are also used in ANNs to form linear combinations of caratteristicas or the outputs of neurons in hidden layers.

zero-gradient condition Consider the unconstrained optimization problem $\min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w})$ with a smooth and convex objective function $f(\mathbf{w})$. A necessary and sufficient condition for a vector $\widehat{\mathbf{w}} \in \mathbb{R}^d$ to solve this problem is that the gradiente $\nabla f\big(\widehat{\mathbf{w}}\big)$ is the zero vector,

$$\nabla f\big(\widehat{\mathbf{w}}\big) = \mathbf{0} \Leftrightarrow f\big(\widehat{\mathbf{w}}\big) = \min_{\mathbf{w}\in\mathbb{R}^d} f(\mathbf{w}).$$

0/1 loss The 0/1 loss $L^{(0/1)}\left((\mathbf{x}, y), h\right)$ measures the quality of a classifier $h(\mathbf{x})$ that delivers a previsione $\hat{y}$ (e.g., via thresholding (1)) for the etichetta y of a punto dati with caratteristicas $\mathbf{x}$. It is equal to 0 if the previsione is correct, i.e., $L^{(0/1)}\left((\mathbf{x}, y), h\right) = 0$ when $\hat{y} = y$. It is equal to 1 if the previsione is wrong, i.e., $L^{(0/1)}\left((\mathbf{x}, y), h\right) = 1$ when $\hat{y} \neq y$.

Indice analitico

Riferimenti bibliografici

[1] W. Rudin, *Real and Complex Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1987.

[2] W. Rudin, *Principles of Mathematical Analysis*, 3rd ed. New York, NY, USA: McGraw-Hill, 1976.

[3] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 4th ed. Baltimore, MD, USA: The Johns Hopkins Univ. Press, 2013.

[4] G. H. Golub and C. F. Van Loan, "An analysis of the total least squares problem," *SIAM J. Numer. Anal.*, vol. 17, no. 6, pp. 883–893, Dec. 1980, doi: 10.1137/0717073.

[5] D. P. Bertsekas and J. N. Tsitsiklis, *Introduction to Probability*, 2nd ed. Belmont, MA, USA: Athena Scientific, 2008.

[6] A. Jung, *Machine Learning: The Basics*. Singapore, Singapore: Springer Nature, 2022.