

FONDAMENTI

PATTERN

Dati oggetto della disciplina Machine Learning, può essere tradotto in italiano in vari modi: forma, campione, esempio, ...

PATTERN RECOGNITION: disciplina che studia il riconoscimento dei pattern

Tipi di pattern

NUMERICI: valori associati a caratteristiche misurabili o conteggi

- Continui o discreti, in ogni caso ordinabili
- L'estrazione di caratteristiche produce vettori numerici (feature vectors)
- Esempi: altezza, peso, lunghezza piede, ...

CATEGORICI: valori associati a caratteristiche qualitative e alla presenza o assenza di una caratteristica

- Non sono mappabili su valori numerici
- Possono essere ordinabili
- Esempi: sesso, colore occhi, gruppo sanguigno, ...

SEQUENZE: pattern sequenziali con relazioni spaziali o temporali

- Lunghezza variabile, allineamento spazio temporale e memoria
- Importanza della posizione, predecessori e successori
- Esempi: stream audio video, frasi, ...

ALTRI PATTERN STRUTTURATI: output organizzati in strutture complessi

- Applicazioni in bioinformatica, Natural Language Processing, Speech recognition
- Grafi, alberi, ...

TRAINING, VALIDATION, TEST SET

TRAINING SET: insieme di pattern utilizzati durante l'addestramento

VALIDATION SET: insieme di pattern utilizzati per tarare gli iperparametri (H)

TEST SET: insieme di pattern su cui valutare le prestazioni finali

TRAIN, VALID e TEST SET dovrebbero essere disgiunti.

K-FOLD CROSS VALIDATION

Metodo di suddivisione dei pattern nei 3 set in caso di campioni ridotti.

K partizioni omogenee, K cicli di addestramento su K-1 partizioni usate per il training e 1 per il validation. Prestazione = media/mediana

LEAVE-ONE-OUT: caso estremo di cross validation con fold di dimensione 1

APPRENDIMENTO

SUPERVISIONATO: le classi dei pattern utilizzati per l'addestramento sono note (training set etichettato)

NON SUPERVISIONATO: non sono note le classi dei pattern

SEMI-SUPERVISIONATO: il training set è parzialmente etichettato, ottimizzazione con pattern non etichettati

BATCH: addestramento una sola volta poi passaggio al **working mode**

INCREMENTALE: più addestramenti dopo quello iniziale, rischio: **catastrophic forgetting**

NATURALE: addestramento continuo (supervisionato e non supervisionato)

REINFORCEMENT LEARNING • **OBIETTIVO:** apprendere un comportamento

- Interazione ambiente-agente
- Sistema di ricompense/premi (anche ritardate rispetto all'azione)
- Q-Learning, Deep reinforcement

PARAMETRI, IPERPARAMETRI, FUNZIONE OBIETTIVO

Il comportamento di un algoritmo di ML è regolato da un set di parametri. L'apprendimento è il processo che cerca il valor ottimo di questi parametri. (H)

La funzione obiettivo $f(Train, \theta)$ può rappresentare l'ottimalità della soluzione o l'errore o perdita.

L'apprendimento cerca di massimizzare $\theta^* = \operatorname{argmax}_{\theta} f(Train, \theta)$ o minimizzare il valore della funzione obiettivo $\theta^* = \operatorname{argmin}_{\theta} f(Train, \theta)$.

Gli iperparametri sono valori che definiscono le caratteristiche degli algoritmi di ML (es: numero di neuroni in una rete neurale, numero di vicini in un classificatore k-NN, grado di un polinomio utilizzato per la regressione, tipo di loss function, ...)

L'apprendimento può essere effettuato più volte a partire da iperparametri differenti, scegliendo al termine il set di iperparametri che forniscono le prestazioni migliori.

FONDAMENTI

PRESTAZIONI

Le prestazioni possono essere misurate direttamente tramite la funzione obiettivo. È però utile definire misure maggiormente correlate alla semantica del problema.

CLASSIFICAZIONE

CLOSED SET: i pattern appartengono ad una delle classi note

ACCURATEZZA = pattern correttamente classificati / pattern classificati

ERRORE = 100% - ACCURATEZZA

OPEN SET: i pattern possono appartenere ad una delle classi note oppure no.

Approccio: aggiunta classe «resto del mondo»

Approccio: definizione livello di soglia per cui le assegnazioni con un livello di confidenza (probabilità) inferiore alla soglia non vengono effettuate

N = Numero di pattern

TP (True Positive) = # pattern correttamente classificati

FP (False Positive) = # pattern classificati erroneamente

TN (True Negative) = # pattern correttamente attribuiti a «resto del mondo» o sotto soglia

FN (False Negative) = # pattern erroneamente attribuiti a «resto del mondo» o sotto soglia

Misure delle prestazioni

False Positive Rate = FP/N

False Negative Rate = FN/N

Nei sistemi con soglia, il valore della soglia determina i risultati su FPR e FNR:

- soglia elevata = FPR basso e FNR alto
- soglia bassa = FPR alto e FNR basso

Le curve FPR e FNR possono essere condensate in:

DET Detection Error Tradeoff (FNR/FPR)

ROC Receiver Operating Characteristic (TPR/FPR)

Nei sistemi biometrici si parla di **False Match** / **False Non-Match** e, in caso di multoclasse di **FPIR (False Positive Identification Rate)** e **FNIR (False Negative Identification Rate)**

PRECISION = TP / (TP + FP) = TRUE POSITIVE / POSITIVE (pertinenza)

RECALL = TP / (TP + FN) = TRUE POSITIVE / RELEVANT ELEMENTS (completezza)



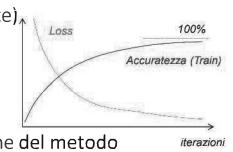
CONVERGENZA • GENERALIZZAZIONE • OVERFITTING

Obiettivo primario durante la fase di training: **CONVERGENZA**

Andamento decrescente del **LOSS** e crescente dell'**ACCURACY**.

DINAMICHE:

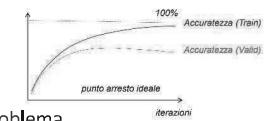
- **LOSS** non decresce (oppure **osilla** significativamente),
 - il sistema non converge
 - metodo di ottimizzazione non efficace:
 - modificare iperparametri
 - modificare **learning rate**
 - correggere errori di implementazione del metodo
- **LOSS** decresce ma l'**ACCURACY** non cresce
 - probabilmente la loss function scelta è errata
- **ACCURACY** non si avvicina al 100% sul training
 - il modello è inadeguato:
 - aumentare i gradi di libertà del classificatore (troppi parametri potrebbero essere troppo restrittivi)
 - modificare gli iperparametri



GENERALIZZAZIONE: capacità di trasferire il livello di accuratezza ottenuto sul training set al validation set

OVERFITTING: ottenimento di risultati ad alte prestazioni sul training set ma non sul validation set. Si verifica facilmente con train set di piccole dimensioni.

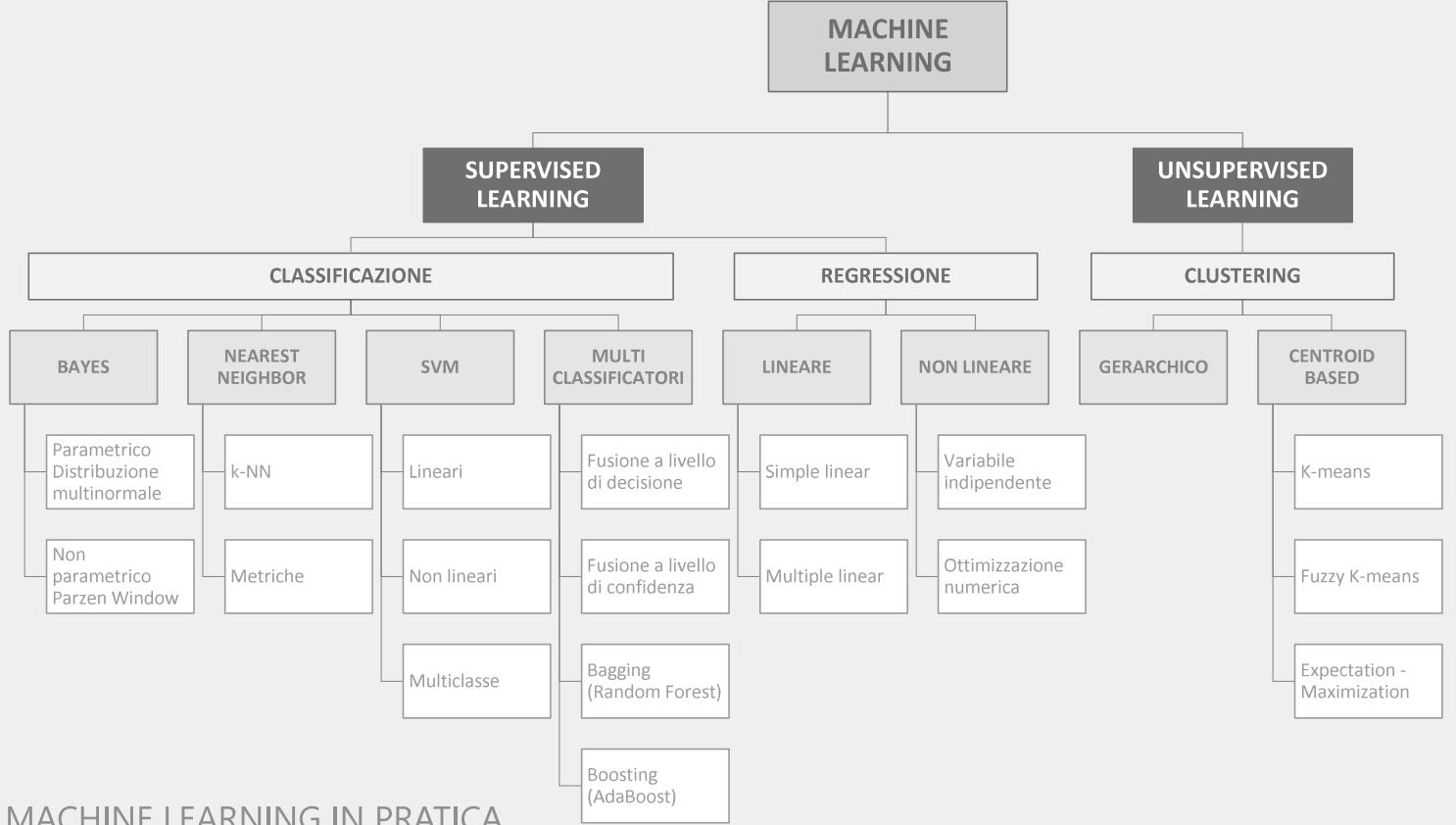
Durante l'addestramento è opportuno interrompere il processo al momento ideale, quando il livello di accuratezza sul validation set non cresce o inizia a decrescere.



Controllare l'overfitting per massimizzare la generalizzazione

- gradi di libertà del classificatore adeguati al problema
 - aumentarli gradualmente
- regolarità della soluzione appresa
 - aggiungere un fattore di regolarizzazione





MACHINE LEARNING IN PRATICA

Approcci ML solo con sufficienti esempi per training e validation set	Sforzi significativi per etichettare grandi insiemi di pattern	Training, Validation e Test set devono essere rappresentativi	Automatizzare le procedure di valutazione delle performance	Confrontare le prestazioni solo con sistemi addestrati su stesso dataset e protocollo	Attenzione agli intervalli di confidenza per risultati su set di piccole dimensioni	Scrivere codice strutturato, ordinato
---	--	---	---	---	---	---------------------------------------

CLASSIFICATORI • Bayes

CLASSE

Insieme di pattern aventi proprietà comune

Esempi di classificazione: spam detection / credit card fraud detection / face recognition / pedestrian classification / medical diagnosis / stock trading

APPROCCIO BAYESIANO

Problema posto in termini probabilistici.

	Definizione	Descrizione	Esempio
$p(x w_i)$	Densità di probabilità condizionale di x data w_i	Probabilità che il prossimo pattern sia x sotto l'ipotesi che la sua classe sia w_i	...abbia una certa altezza sapendo che è una donna
$P(w_i)$	Probabilità a priori	Probabilità che il prossimo pattern sia di classe w_i	...sia donna
$p(x)$	Densità di probabilità assoluta	Probabilità che il prossimo pattern sia x	...abbia una certa altezza
$P(w_i x)$	Probabilità a posteriori di w_i dato x	Probabilità che il pattern x sia di classe w_i	...sia una donna sapendo che ha una certa altezza

RELACIONI

$$p(x) = \sum_{i=1}^s p(x|w_i) \cdot P(w_i) \quad \text{dove} \quad \sum_{i=1}^s P(w_i) = 1$$

$$P(w_i|x) = \frac{p(x|w_i) \cdot P(w_i)}{p(x)}$$

Per ogni classe è necessario calcolare la probabilità a priori e la densità di probabilità condizionale.

La regola assegna la classe (b) con la probabilità a posteriori più alta per il pattern x : $b = \operatorname{argmax}_{i=1 \dots s} \{P(w_i|x)\}$

$$P(\text{error}) = \int_{\mathcal{R}_1} p(x|w_2)P(w_2)dx + \int_{\mathcal{R}_2} p(x|w_1)P(w_1)dx$$

La conoscenza delle densità di probabilità condizionale è possibile solo in teoria.

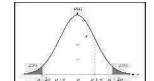
PARAMETRICO

Ipotesi sulla forma delle distribuzioni.

Piccole dimensioni del training set e/o ragionevole certezza sulla curva di distribuzione.

DISTRIBUZIONE NORMALE ($d=1$)

$$p(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



DISTRIBUZIONE MULTINORMALE

$$p(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1} (x-\mu)}$$

Gli elementi diagonali della matrice di covarianza sono le varianze per ogni dimensione, gli elementi non diagonali sono le covarianze e determinano la correlazione statistica ($=0 \rightarrow$ indipendenti; $>0 \rightarrow$ correlati positivamente; $<0 \rightarrow$ correlati negativamente).

$$\text{DISTANZA DI MAHALANOBIS } r^2 = (x - \mu)^t \Sigma^{-1} (x - \mu)$$

Definisce i bordi a densità costante in una distribuzione multinormale.

Sostituisce spesso la distanza euclidea essendo in grado di pesare le diverse componenti tenendo conto dei relativi spazi di variazione e della loro correlazione.

DECISION BOUNDARY / DECISION SURFACE: zona di confine tra regioni che il classificatore associa a classi diverse, sul confine la classificazione è ambigua.

CONFIDENZA DI CLASSIFICAZIONE

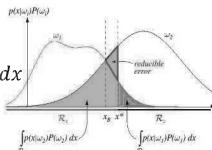
Vantaggio del classificatore di Bayes legato al fatto che produce un valore probabilistico di output, utilizzabile come confidenza per scartare pattern in applicazioni open set con soglia o costruire un multiclassificatore.

Se non serve la confidenza $b = \operatorname{argmax}_{i=1 \dots s} \{P(w_i|x)\} = \operatorname{argmax}_{i=1 \dots s} \{p(x|w_i) \cdot P(w_i)\}$

IN PRATICA

Attenzione alle ipotesi sulle distribuzioni delle densità di probabilità.

1. Valutare la corrispondenza alla normalità delle s distribuzioni
 1. formalmente (Malkovich/Afifi)
 2. empiricamente (tramite visualizzazione dati)
2. Stima del vettore medio e della matrice di covarianza
3. Probabilità a priori estratte dal campione nel training set o tutte uguali
4. Assegnazione della classe ad ogni nuovo pattern in base alla regola di Bayes.



NON PARAMETRICO • PARZEN WINDOW

Apprendimento delle distribuzioni dal training set.

La stima delle densità è solitamente affrontabile in spazi a dimensionalità ridotta.

Campionamento: l'idea è quella di campionare lo spazio (indipendentemente dal numero di dimensioni) per individuare, nelle diverse zone, la distribuzione dei pattern nelle classi.
Approcci: **divisione in celle** (problemi legati ai confini e alla scelta delle dimensioni delle celle), **utilizzo di intorni** (con problemi legati alla scelta delle dimensioni degli intorni).

Probabilità che un pattern x sia dentro \mathcal{R} : $P_1 = \int_{\mathcal{R}} p(x') dx'$

Probabilità che k pattern siano dentro \mathcal{R} : $P_k = \binom{n}{k} P_1^k (1 - P_1)^{n-k}$

Valore medio: $k = nP_1 \Rightarrow P_1 = k/n$

Se \mathcal{R} di volume V è sufficientemente piccola: $P_1 = \int_{\mathcal{R}} p(x') dx' \approx p(x) \cdot V$

Da cui: $p(x) = P_1/V = k/(n \cdot V)$

Il volume di un ipercubo centrato in x con lato h_n è: $k_n = \sum_{i=1}^n \varphi\left(\frac{x_i - x}{h_n}\right)$

Da cui: $p(x) = P_n/V_n = k_n/(n \cdot V_n) = \sum_{i=1}^n \varphi\left(\frac{x_i - x}{h_n}\right)/(n \cdot V_n)$

Se la finestra è **piccola** la stima è *rumorosa* e *statisticamente instabile*.

Se la finestra è **grande** la stima è *stabile* ma *piuttosto vaga e sfocata*.

Per ottenere convergenza la dimensione della finestra deve essere calcolata tenendo conto del numero di campioni del training set: $V_n = V_1/\sqrt{n}$

Kernel function soft vengono utilizzate al posto degli ipercubi in modo da realizzare superfici decisionali più regolari (smoothed). Devono essere funzioni di densità (sempre positive con integrale su tutto lo spazio pari a 1).

$$\varphi(u) = \frac{1}{(2\pi)^{d/2}} e^{-\frac{u^T u}{2}}$$

ESEMPI

(dalla prova di autovalutazione del 17/12/2016)

Date due distribuzioni multinormali identificate dai seguenti parametri:

$$\mu_0 = \begin{bmatrix} 1,03 \\ 1,03 \end{bmatrix} \quad \Sigma_0^{-1} = \begin{bmatrix} 52,56 & -10,05 \\ -10,05 & 36,88 \end{bmatrix} \quad |\Sigma_0| = 0,000544 \quad P(w_0) = 0,6$$

$$\mu_1 = \begin{bmatrix} 2,02 \\ 1,53 \end{bmatrix} \quad \Sigma_1^{-1} = \begin{bmatrix} 100,58 & -22,34 \\ -22,34 & 37,85 \end{bmatrix} \quad |\Sigma_1| = 0,000302 \quad P(w_1) = 0,4$$

Nell'ipotesi dell'utilizzo di un classificatore di Bayes multinormale, calcolare per il punto $x = \begin{bmatrix} 1,60 \\ 1,25 \end{bmatrix}$:

- Le densità di probabilità condizionali
- Le probabilità a posteriori
- L'indice della classe restituita in output.

Si ricorda che la densità di probabilità, nel caso della distribuzione multinormale è:

$$p(x) = \frac{1}{(2\pi)^{\frac{d}{2}} |\Sigma|^{\frac{1}{2}}} \cdot e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

SVOLGIMENTO

1) Si calcolano le densità di probabilità condizionali (d=2)

$$p(x|w_0) = \frac{1}{2\pi \cdot \sqrt{0,000544}} \cdot e^{-\frac{1}{2}[1,6-1,03 \ 1,25-1,03] \begin{bmatrix} 52,56 & -10,05 \\ -10,05 & 36,88 \end{bmatrix} [1,6-1,03]} = 0,00193$$

$$p(x|w_1) = \frac{1}{2\pi \cdot \sqrt{0,000302}} \cdot e^{-\frac{1}{2}[1,6-2,02 \ 1,25-1,53] \begin{bmatrix} 100,58 & -22,34 \\ -22,34 & 37,85 \end{bmatrix} [1,6-2,02]} = 0,00403$$

2) Si calcola la densità di probabilità assoluta di x

$$p(x) = 0,00193 * 0,6 + 0,00403 * 0,4 = 0,00277$$

3) Si calcolano le probabilità a posteriori

$$\begin{aligned} p(w_0|x) &= 0,6 \cdot 0,00193 / 0,00277 = 0,418 \\ p(w_1|x) &= 0,4 \cdot 0,00403 / 0,00277 = 0,582 \end{aligned}$$

4) Si valutano le probabilità a posteriori per scegliere la classe

La probabilità a posteriori è maggiore per la classe 1.

CLASSIFICATORI • Nearest Neighbor

NN, kNN

NEAREST NEIGHBOR classifica un pattern x assegnandogli la stessa classe del pattern a lui più vicino nel training set sulla base di una metrica $dist(\cdot)$ specifica.

$$dist(x, x') = \min_{x_i \in TS} \{dist(x, x_i)\}$$

L'idea è di massimizzare la probabilità a posteriori in maniera pragmatica $P(w_i|x) \approx P(w_i|x')$

La probabilità di errore di NN non supera mai il doppio del minimo errore possibile (Bayes)

Tassellazione di Voronoi: ogni pattern del training set determina un tassello all'interno del quale i pattern vengono assegnati alla stessa classe: un elemento poco affidabile determina un errore di classificazione sui pattern vicini.

K-NEAREST NEIGHBOR definisce la classe di un pattern sulla base dei k pattern più vicini

k è un iperparametro, valori ottimali (< 10) da determinare su validation set separato

Se il training set è infinito, k-NN è migliore di 1-NN. All'aumentare di k , l'errore converge a quello Bayesiano.

Aumentare k significa estendere l'ipersfera cercando di valutare la proprietà a posteriori lontano dal punto di interesse.

CONFIDENZA DI CLASSIFICAZIONE

$$Siano [v_1, v_2, \dots, v_s] \quad \sum_{i=1}^s v_i = k \quad i \text{ voti ottenuti dal pattern } x \Rightarrow \text{confidenza: } \left[\frac{v_1}{k}, \frac{v_2}{k}, \dots, \frac{v_s}{k} \right]$$

COMPLESSITÀ COMPUTAZIONALE

- Memorizzare tutti i pattern del training set
- Per ogni pattern calcolare la distanza da tutti i pattern del training set
 - Ordinare le distanze per valutare le k più piccole

Semplificazioni (pattern prototipo):

EDITING eliminazione di pattern dal training set

CONDENSING derivazione di nuovi prototipi (es. vettore medio)

• Riduzione dei calcoli

• Prototipi maggiormente affidabili rispetto ai pattern

INDICIZZAZIONE DATI (kd-tree)

METRICHE

Comportamento di Nearest Neighbor strettamente collegato alla metrica adottata.

DISTANZA EUCLIDEA

Spesso utilizzata

Caso L₂ delle metriche di Minkowski $L_k(a, b) = (\sum_{i=1}^d |a_i - b_i|^k)^{1/k}$

$$L_2(a, b) = \sqrt{\sum_{i=1}^d |a_i - b_i|^2} = \|a - b\|_2$$

Non gestisce correttamente eventuali correlazioni (es. altezza e lunghezza piede).

NORMALIZZAZIONE SULLO SPAZIO DI VARIAZIONE

$$L_2(a, b) = \sqrt{\sum_{i=1}^d \left(\frac{p_i}{v_i} (a_i - b_i) \right)^2} \quad \begin{aligned} v_i: &\text{spazio di variazione} \\ p_i: &\text{peso feature } i - \text{esima} \end{aligned}$$

Gli **spazi di variazione** possono essere calcolati come range (max-min) sul training set o, meglio, in proporzione alla deviazione standard.

I **pesi** possono essere scelti in proporzione al **potere discriminante**, ad esempio come rapporto:

$$p_i = \frac{\text{variabilità interclasse feature } i}{\text{variabilità intraclass feature } i}$$

Variabilità interclasse: es. media deviazioni standard dei valori della feature in ogni classe

Variabilità intraclass: es. deviazione standard dei valori della feature in un campione di pattern per ogni classe

WHITENING TRANSFORM

Efficace normalizzazione rispetto agli spazi di variazione, tiene conto delle correlazioni tra features. Utilizza come metrica la distanza euclidea ma sullo spazio **sferificizzato**.

- Rimuove le correlazioni «ruotando» la nuvola di punti per allineare gli assi di variazione principale agli assi cartesiani
- Sferificizza l'ellissoide uniformando le varianze (autovalori) a 1 lungo tutti gli assi

DISTANZA DI MAHALANOBIS $r^2 = (x - \mu)^T \Sigma^{-1} (x - \mu)$

METRIC LEARNING

Approccio più generale per la scelta della metrica: learning supervisionato della metrica.

Obiettivo: determinare una trasformazione degli input che **avvicini i pattern della stessa classe e allontani quelli di classi diverse**.

- Determinazione di una matrice G di trasformazione: $\|G\mathbf{a} - G\mathbf{b}\|_2$
- Approcci non lineari: $\|\mathbf{G}\phi(\mathbf{a}) - \mathbf{G}\phi(\mathbf{b})\|_2 \quad \phi: \text{funzione non lineare}$

SIMILARITÀ COSENO E DISTANZA COSENO

Utilizzata in applicazioni di **information retrieval**, **data mining**, **text mining**.

Due vettori identici hanno similarità 1, due opposti hanno similarità -1.

$$CosSimilarity(\mathbf{a}, \mathbf{b}) = (\mathbf{a}^T \cdot \mathbf{b}) / (\|\mathbf{a}\| \cdot \|\mathbf{b}\|)$$

$$AngularDistance(\mathbf{a}, \mathbf{b}) = \cos^{-1}(CosSimilarity(\mathbf{a}, \mathbf{b})) / \pi$$

TEORIA

Introdotta da Vapnik nel 1965 e successivamente perfezionata.
Molto utilizzata nella classificazione di pattern.
Anziché stimare le densità di probabilità, risolve direttamente il problema di interesse determinando le superfici decisionali tra le classi.

PATTERN LINEARMENTE SEPARATI

SVM lineare determina l'iperpiano di separazione, tra tutti quelli possibili su due insiemi di pattern di due classi linearmente separabili, con il maggior margine possibile, stimando che in tal modo si possono classificare correttamente anche i pattern di test vicini al confine.

MARGINE: distanza minima di punti delle due classi nel training set dall'iperpiano.

IPERPANO: $D(\mathbf{x}) = \mathbf{w} \cdot \mathbf{x} + b$

DISTANZA VETTORE-IPERPANO: $r = D(\mathbf{x}) / \|\mathbf{w}\|$

OBIETTIVO: individuare l'iperpiano che massimizza la distanza (o minimizza l'inverso)

$$\begin{array}{ll} \text{Minimizzare} & \|\mathbf{w}\|^2 / 2 \\ \text{Vincoli} & y_i [\mathbf{w} \cdot \mathbf{x}_i + b] - 1 \geq 0 \quad \text{per } i = 1, \dots, n \end{array}$$

SUPPORT VECTOR: pattern del training set che giacciono sul margine, da soli determinano la soluzione al problema. $D(\mathbf{x}) = 1 / \|\mathbf{w}\| = \pm 1$

SOLUZIONI: possono essere individuate tramite **formulazione Lagrangiana** e **formulazione duale** tramite condizioni Karush-Kuhn-Tucker (KKT).

$$\begin{array}{ll} \text{Formulazione Lagrangiana} & Q(\mathbf{w}, b, \alpha) = \frac{1}{2} \mathbf{w} \cdot \mathbf{w} - \sum_{i=1}^n \alpha_i (y_i [\mathbf{w} \cdot \mathbf{x}_i + b] - 1) \\ & Q(\alpha) = \sum_{i=1 \dots n} \alpha_i - \frac{1}{2} \sum_{i,j=1 \dots n} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ \text{Duale} & \text{con vincoli} \sum_{i=1 \dots n} y_i \alpha_i = 0 \quad \text{e} \quad \alpha_i \geq 0 \quad \text{per } i = 1, \dots, n \end{array}$$

Risolvibile con algoritmi di programmazione quadratica:

$$D(\mathbf{x}) = \mathbf{w}^* \cdot \mathbf{x} + b = \sum_{i,j=1 \dots n} \alpha_i^* y_i (\mathbf{x}_i \cdot \mathbf{x}_j) + b^*$$

VANTAGGI

- Soluzione calcolata su un numero ridotto di support vector
- Complessità legata al numero di support vector e errore medio limitato dal rapporto support vector / pattern
- Metodo adatto anche a dimensionalità elevate: quadratico rispetto al numero di pattern

PATTERN LINEARMENTE NON SEPARATI

«Rilassamento dei vincoli di separazione» per permettere che alcuni (il minor numero possibile) pattern possano valicare il confine della classe.

VARIABILI DI SLACK: $y_i [\mathbf{w} \cdot \mathbf{x}_i + b] \geq 1 - \xi_i$ per ogni pattern \mathbf{x}_i , la variabile ξ_i definisce la deviazione dal margine.

OBIETTIVO: massimizzare il margine e minimizzare il numero di elementi non correttamente classificati.

$$\begin{array}{ll} \text{Minimizzare} & \frac{\|\mathbf{w}\|^2}{2} + C \sum_{i=1 \dots n} \xi_i \\ \text{Vincoli} & y_i [\mathbf{w} \cdot \mathbf{x}_i + b] \geq 1 - \xi_i \quad \text{per } i = 1, \dots, n \end{array}$$

COEFFICIENTE C: definisce l'importanza relativa degli errori di classificazione rispetto all'ampiezza del margine (è un iperparametro).

SOLUZIONI: come nel caso di pattern linearmente separati

SVM NON LINEARE

Estensione SVM di separazione con superfici anche molto complesse.

MAPPING: dallo spazio di partenza ad uno spazio a più alta dimensionalità.

KERNEL FUNCTION: $K(\mathbf{x}, \mathbf{x}') = \phi(\mathbf{x}) \cdot \phi(\mathbf{x}')$ (la scelta di K è un iperparametro)

SOLUZIONE SENZA PARTICOLARI COMPLICAZIONI: $D(\mathbf{x}) = \sum_{i=1 \dots n} \alpha_i^* y_i K(\mathbf{x}, \mathbf{x}_i) + b^*$

Possibili Kernel Function

- **POLINOMI DI GRADO Q** (Q iperparametro)
- **RADIAL BASIS FUNCTION (RBF) DI AMMIEZZA σ** (σ iperparametro)
- **2-LAYER NEURAL NETWORK** $K(\mathbf{x}, \mathbf{x}') = \tanh(v(\mathbf{x} \cdot \mathbf{x}') + a)$ (v, a iperparametri)

SVM ESTENSIONE MULTICLASSE

SVM determina la superficie di separazione tra 2 classi di pattern.

Possibili estensioni multiclass (approcci al problema ancora aperto):

- **ONE-AGAINST-ONE:** $s \times (s - 1)/2$ classificatori binari + majority vote rule
- **ONE-AGAINST-ALL:** richiede un numero di training SVM pari al numero delle classi e assegna il pattern, al termine, in base alla massima distanza dalla superficie decisionale.

IN PRATICA

Dimensionalità alta → SVM lineare, tarare iperparametro C.
Dimensionalità bassa → SVM non lineare, Kernel RBF
Dimensionalità media → tentativi con entrambi
Multiclasse → se possibile One-Against-One, altrimenti O-A-A

CLASSIFICATORI • Multiclassificatori

METODI

Esecuzione di classificazione con diversi classificatori e successiva fusione dei risultati.

Esecuzione parallela, in cascata o in modo gerarchico.

L'utilizzo di combinazioni di classificatori può migliorare molto le prestazioni.

INDIPENDENZA

La combinazione è efficace quando i singoli classificatori sono sufficientemente indipendenti tra loro (non commettono gli stessi errori).

INDIPENDENZA:

- utilizzo di **feature diverse** (non correlate o poco correlate)
- utilizzo di **diversi algoritmi per l'estrazione delle features**
- utilizzo di **diversi algoritmi di classificazione**
- utilizzo di **diverse porzioni di training set per lo stesso algoritmo** (bagging)
- **iterazione addestramento sui pattern erroneamente classificati** (boosting)

FUSIONE A LIVELLO DI DECISIONE

Ogni classificatore fornisce la propria decisione: classe a cui ha assegnato il pattern.

Metodi di combinazione dei risultati

MAJORITY VOTE RULE: il pattern assegnato alla classe maggiormente votata

$$\begin{aligned} \{C_1, C_2, \dots, C_{nc}\} & \text{insieme di } nc \text{ classificatori} \\ \theta_{ij} &= \begin{cases} 1 & \text{se } i \text{ è la classe votata da } C_j \\ 0 & \text{altrimenti} \end{cases} \\ t &= \operatorname{argmax}_{i=1 \dots s} \left\{ \sum_{j=1 \dots nc} \theta_{ij} \right\} \end{aligned}$$

Prestazioni teoriche elevate dipendenti dalla totale indipendenza dei classificatori.

$$P_{multi}(nc) = \sum_{m=k}^{nc} \binom{nc}{m} P^m (1 - P)^{nc-m}$$

BORDA COUNT: ogni classificatore fornisce una classifica delle classi in base alla probabilità; i ranking sono convertiti in punteggi e viene assegnata la classe con punteggio più alto (considera anche i non primi di ogni classificatore).

ONE AGAINST ONE: consente di risolvere un problema di classificazione multi-classe attraverso classificatori binari

1. Per s classi del problema, si addestrano $s \times (s - 1)/2$ classificatori binari
2. Il pattern \mathbf{x} è classificato da ogni classificatore assegnando un voto alla classe più probabile

FUSIONE A LIVELLO DI CONFIDENZA

Ogni classificatore fornisce la confidenza di classificazione del pattern per ogni classe.

$$conf_j = [conf_{j1}, conf_{j2}, \dots, conf_{js}]$$

Metodi di fusione

SOMMA	$sum = \sum_{j=1 \dots nc} conf_j$	Sceglie la classe con somma massima Numericamente più stabile per valori di confidenza bassi su uno o pochi classificatori
PRODOTTO	$prod = \prod_{j=1 \dots nc} conf_j$	Sceglie la classe con prodotto massimo Metodo più corretto dal punto di vista probabilistico nel caso di indipendenza statistica
MASSIMO	$max_i = \max_{j=1 \dots nc} \{conf_{ji}\}$	Sceglie la classe che ha ottenuto la massima confidenza massima
MINIMO	$min_i = \min_{j=1 \dots nc} \{conf_{ji}\}$	Sceglie la classe che ha ricevuto la massima confidenza minima Quindi la classe che non ha confidenza minima troppo bassa su ogni classificatore
SOMMA PESATA	$sum = \sum_{j=1 \dots nc} g_j \cdot conf_j$	Pesa la confidenza dei diversi classificatori Ad esempio in maniera inversamente proporzionale all'errore di classificazione

BAGGING E BOOSTING

RANDOM FOREST (Metodo della famiglia **Bagging: Bootstrap Aggregating**)

- Addestramento con classificator (**classification tree**) i su diverse porzioni del training set
- Fusione (majority vote rule, somma, ...)

CLASIFICATION TREE: Albero binario in cui ogni nodo divide i pattern sulla base di un criterio su una singola feature, ogni livello utilizza progressivamente la feature che divide meglio il set. Ogni foglia rappresenta l'assegnazione o meno del pattern alla classe. La scelta della feature in ogni nodo è fatta su un sottoinsieme casuale delle feature per evitare scelte uguali sui vari classification trees (bagging sulle features).

ADABOOST (Metodo della famiglia **Boosting**)

- Effettua un certo numero di iterazioni
 - Ad ogni iterazione calcola/aggiorna i pesi dei pattern
 - Sceglie il classificatore che realizza il minor errore
 - Calcola il peso del classificatore (errore alto+classificazione corretta o errore basso+classificazione errata)
 - Il classificatore strong finale è media pesata dei weak classificatori iniziali.

REGRESSIONE

OBIETTIVO

Apprendimento della funzione che assegna un valore numerico continuo ad ogni pattern.
 $f(x) \rightarrow y$
 f è controllata da un insieme di parametri
 x è la **variabile indipendente (esatta)** e y è la **variabile dipendente (soggetta ad errore)**.

SIMPLE LINEAR

La variabile dipendente è uno **scalare**.

$f(x) = y = \alpha + \beta \cdot x_i + \varepsilon_i$ ε_i è l'errore α, β sono i parametri da determinare
La soluzione consiste nel determinare l'equazione della retta che minimizza la somma dei minimi quadrati:

$$\alpha^*, \beta^* : \arg \min_{\alpha, \beta} \sum_{i=1 \dots n} (f(x) - y_i)^2 = \sum_{i=1 \dots n} \varepsilon_i^2$$

Da cui

$$\beta^* = \frac{\sum_{i=1 \dots n} (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1 \dots n} (x_i - \bar{x})^2} \quad \alpha^* = \bar{y} - \beta^* \bar{x}$$

MULTIPLE LINEAR

La variabile dipendente è un **vettore**.

$$f(X) = y = X\beta$$

X è una matrice rettangolare (righe=patterns, colonne=features + colonna 1)

y è un vettore colonna con uno scalare per ogni pattern

β è un vettore colonna con uno scalare per ogni feature + 1 termine noto

Funzione obiettivo da minimizzare: $\|y - X\beta\|^2$ da cui i parametri $\beta^* = (X^t X)^{-1} X^t y$

b = numero di pattern, d = dimensioni

Se $n = d+1 \rightarrow X$ è una matrice quadrata e l'iperpiano tocca tutti i punti

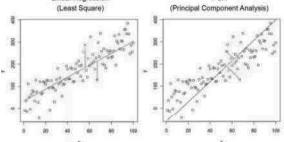
Se $n > d+1 \rightarrow$ numero di equazioni superiore alle incognite, l'iperpiano approssima i punti

Perché esiste $(X^t X)^{-1}$, $X^t X$ deve avere rango massimo (problemi punti collineari)

$X^t X$ è spesso mal condizionata, conseguenza: calcolo numerico instabile (SVD)

REGRESSION vs GEOMETRICAL FITTING

Nel caso di nuvole di punti geometrici le coordinate hanno lo stesso ruolo ed è preferibile PCA fitting a regressione lineare.



NON LINEAR

f è controllata da un insieme di parametri non lineari (es. potenze)

La funzione di regressione deve essere lineare rispetto ai parametri, mentre può essere non lineare rispetto ai valori delle features.

$$y = \beta_1 x^2 + \beta_2 x + \beta_3$$

In caso di dipendenza non lineare del modello dai parametri non esiste soluzione least square e si applicano tecniche iterative di ottimizzazione numerica (Gradient Descent, Gauss-Newton).

IN PRATICA

Regessione su problemi di piccole-medie dimensioni

- caso lineare \rightarrow least square con fattorizzazione SVD
- caso non lineare \rightarrow Gauss-Newton (come prima scelta)

Esistono varianti di classificazione SVM e Random Forest efficaci in problemi di regressione.

In ambito deep learning, reti neurali addestrate con gradient descent sono utilizzate per risolvere problemi di regressioni di enormi dimensioni (Faster R-CNN)

CLUSTERING

OBIETTIVO E COMPLESSITÀ

Famiglia di metodi non supervisionati in grado di individuare raggruppamenti intrinseci (cluster) di pattern nello spazio multidimensionale. Opzionalmente definisce le classi (incognite)

Problema NP hard

Ammettendo di conoscere il numero s di classi, il numero di pattern può essere clusterizzato in $s^n / s!$ modi diversi (numeri di Stirling).

Se s non è noto, il numero di casi è definito dai numeri di Bell (somma di tutti i numeri di Stirling da 1 a n).

CRITERI DI CLUSTERING

I criteri di clustering definiscono cosa si vuole ottenere e specificano il grado di ottimalità di ogni soluzione ammissibile.

OSSERVAZIONI

- I pattern all'interno dello stesso cluster devono essere tra loro più simili rispetto a pattern appartenenti a cluster diversi
- I cluster sono costituiti da **nuvole di punti** a densità relativamente elevata, separate da zone con densità più bassa

CRITERI

Minimizzazione distanze dai centroidi

Buon criterio per cluster a simmetria radiale, penalizza forme allungate o cluster innestati

Minimizza la somma dei quadrati delle distanze dei pattern x dai centroidi delle classi.

$$J_e = \sum_{i=1 \dots s} \sum_{x \in C_i} \|x - \bar{x}_i\|^2 \quad \bar{x}_i = \frac{1}{n} \sum_{x \in C_i} x$$

Minimizzazione distanze intra-classe

Simile al criterio dei centroidi ma non penalizza i cluster allungati.

$$J_e = \sum_{i=1 \dots s} n_i \cdot \bar{s}_i \quad \bar{s}_i = \frac{1}{n_i} \sum_{x \in C_i} f_s(x, C_i) \quad f_s(x, C_i) \text{ è una misura di distanza}$$

$f_s(x, C_i) = \min_{x' \in C_i, x' \neq x} \|x - x'\|^2$ definisce che è sufficiente un vicino

ALGORITMI DI CLUSTERING

Gli algoritmi di clustering, dato un criterio di clustering, forniscono una procedura algoritmica per determinare soluzioni che lo ottimizzano.

HARD: esclusivo | **SOFT (fuzzy)**: pattern appartengono a diversi cluster con una certa probabilità.

CLUSTERING GERARCHICO

Metodo simile alla tassonomia biologica. Iterativo prevede di aggregare singoli pattern, poi pattern a cluster e poi cluster a cluster per maggiore similitudine rispetto ad una soglia.

Differenze implementative: **single link**, **average link**, **complete link**.

CLUSTERING BASATO SU CENTROIDI

K-Means: semplice sia a livello computazionale che implementativo.

1. Soluzione iniziale (casuale)
2. Assegna ogni pattern al cluster per cui è minima la distanza dal centroide
3. Ricalcola i centroidi come media dei pattern per ogni cluster
4. Itera finché non si hanno modifiche di assegnazione.

Soluzione in pochi passi.

Distanza euclidea \rightarrow cluster iper-sferici. Distanza di Mahalanobis \rightarrow cluster iper-eliosoidali.

Per evitare convergenza su minimi locali si esegue l'algoritmo più volte su soluzioni iniziali diverse (random, euristiche, metodo evoluzionistico).

Tecniche di validazione:

- valutazione delle soluzioni per valori diversi di s (numero di cluster) penalizzando soluzioni con molti cluster
- Ricerca di discontinuità nel valore di J_e .

Fuzzy K-Means: consente ad un pattern di appartenere a diversi cluster con specifici gradi di probabilità. I centroidi sono calcolati come media pesata rispetto alle probabilità di appartenenza.

(iperparametro m definisce il livello di sfumatura dell'appartenenza ai cluster, tipicamente m = 2)

Expectation – Maximization (EM): tenta di identificare i valori dei parametri delle singole distribuzioni che si ipotizza abbiano generato i patterns per attribuire ogni pattern al cluster relativo. Stima dei parametri con criterio di **maximum (log-) likelihood** (densità/probabilità di aver ottenuto i pattern da specifici parametri).

Iterazione ciclo:

1. Expectation: calcolo del log-likelihood completo sulla base dei parametri
2. Maximization: ricalcolo parametri per massimizzare il valore atteso
 $\theta^{g+1} = \arg \max_{\theta} Q(\theta | \theta^g)$

Algoritmo simile a Fuzzy K-Means ma con maggiori gradi di libertà nella forma dei cluster (elliosoidali).

CLUSTERING BASATO SULLA DENSITÀ

I cluster individuati sono regioni connesse in aree ad elevata densità (DBSCAN).

RIDUZIONE DIMENSIONALITÀ

OBIETTIVO

Eseguire un **mapping** dallo spazio iniziale \mathcal{R}^d ad uno spazio di dimensione inferiore $\mathcal{R}^k, k < d$.

COMPRESIONE

- Scarto di informazioni meno rilevanti
- Combinazione opportuna delle dimensioni esistenti (non eliminazione selettiva)

SCOPO: semplificazione addestramento, miglioramento prestazioni

PRINCIPAL COMPONENT ANALYSIS (PCA)

MAPPING LINEARE • TRASFORMAZIONE NON SUPERVISIONATA

OBIETTIVO: preservare al massimo l'informazione dei pattern.

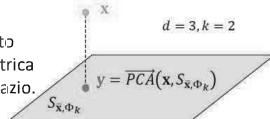
Nota anche con come Karhunen Loeve (KL) transform.

Lo spazio k -dimensionale ridotto è definito univocamente dal vettore medio e dalla matrice di proiezione le cui colonne sono costituite dagli **autovettori** della matrice di covarianza dello spazio originale, corrispondenti ai k più grandi **autovalori**.

Ogni autovalore descrive la varianza del training set lungo una direzione definita dal corrispondente autovettore. I primi k autovettori sono detti **principal component**.

PROIEZIONE

La proiezione di un pattern sullo spazio PCA determinato tramite la matrice di proiezione, è la proiezione geometrica di un vettore (ridotto) sull'iperpiano che definisce lo spazio.



RETROPROIEZIONE

Dato un vettore nello spazio ridotto PCA, la retro-proiezione si ottiene moltiplicando il vettore PCA per la matrice di proiezione e sommando il vettore medio.

Questa operazione non sposta spazialmente il vettore (che giace ancora sullo spazio PCA) ma opera un cambiamento di coordinate che ne permette la codifica in termini delle componenti dello spazio originale.

Tra tutte le riduzioni di dimensionalità lineari, PCA è quella che preserva al massimo l'informazione dei vettori originali.

SCELTA DELLA DIMENSIONE

Se l'obiettivo è la visualizzazione $\rightarrow k$ deve essere 2 o 3

Se l'obiettivo è scartare dati inutili mantenendo la maggior parte del contenuto informativo \rightarrow fissata una percentuale del valore informativo che si vuole preservare, si sceglie il minimo valore di k tale che la somma dei più grandi k autovalori sia maggiore o uguale a t rispetto alla somma di tutti gli autovalori.

PCA WHITENING

Tecnica di pre-normalizzazione dei dati che:

- Rimuove le correlazioni tra le dimensioni ruotando la nuvola di punti per allineare gli assi di variazione principale dei dati (autovalori) agli assi cartesiani
- Sfericizza l'ellissoide uniformando le varianze a 1 lungo tutti gli assi

La matrice di covarianza dei dati normalizzati è la **matrice identità**.

LINEAR DISCRIMINANT ANALYSIS (LDA)

MAPPING LINEARE • TRASFORMAZIONE SUPERVISIONATA

OBIETTIVO: privilegiare le dimensioni che discriminano meglio i pattern, ovvero massimizzare la separazione tra le classi.

MATRICI DI SCATTERING (SPARPAGLIAMENTO)

WITHIN-CLASS S_w : indica come i vettori dei pattern sono sparpagliati rispetto al centro delle classi.

BETWEEN-CLASS S_b : indica come i centri delle classi sono sparpagliati rispetto al centro generale

Una matrice di scatter si calcola come una matrice di covarianza senza normalizzare per il numero di pattern.

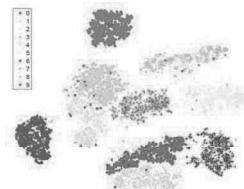
T-DISTRIBUTED STOCHASTIC NEIGHBOR EMBEDDING (t-SNE)

MAPPING NON LINEARE • TRASFORMAZIONE NON SUPERVISIONATA

OBIETTIVO: riduzione a 2 o 3 dimensioni per poter visualizzare i dati multidimensionali.

Introdotta nel 2008 da Van der Maaten e Hinton, rappresenta lo stato dell'arte per la visualizzazione 2D o 3D di dati multidimensionali.

Supera i limiti di riduzione a 2/3 dimensioni di PCA relativamente a cluster distribuiti con forme non multinormali.



RETI NEURALI

NEURONE ARTIFICIALE

Introdotto da McCulloch e Pitts nel 1943.

È l'elemento costitutivo delle reti neurali.

Ogni neurone:

- riceve in input d valori, uno per ogni connessione in ingresso:
ogni input i è associato ad un peso w_i che determina l'efficacia delle connessioni
- prevede un input pesato aggiuntivo (**bias**) di valore 1 utile per tarare il neurone
- elabora un **livello di eccitazione** $net_i = \sum_{j=1 \dots d} i_{nj} * w_{ji} + w_{0i}$
- ha un comportamento regolato da una funzione di attivazione $f(\cdot)$ determinata da net_i :
 $f(\cdot)$ è una funzione **non lineare, continua e differenziabile**



FUNZIONE DI ATTIVAZIONE

STANDARD LOGISTIC FUNCTION (SIGMOID)

PIÙ SEMPLICE

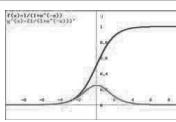
FUNZIONE:

$$f(net) = \sigma(net) = \frac{1}{1 + e^{-net}}$$

DERIVATA:

$$\sigma'(net) = \frac{\partial}{\partial net} \left(\frac{1}{1+e^{-net}} \right) = \frac{e^{-net}}{(1+e^{-net})^2} = \sigma(net)(1 - \sigma(net))$$

[0 ... +1]



TANGENTE IPERBOLICA (TANH)

PIÙ COMPLESSA MA CONVERGENZA PIÙ RAPIDA

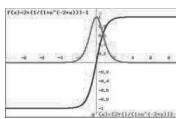
FUNZIONE:

$$f(net) = \tau(net) = 2\sigma(2 \cdot net) - 1 = \frac{2}{1 + e^{-2 \cdot net}} - 1$$

DERIVATA:

$$\tau'(net) = 1 - \tau(net)^2$$

[-1 ... +1]



MULTILAYER PERCEPTRON (MLP)

PERCEPTRON (PERCETTRONE)

Modello proposto da Rosenblatt nel 1965, simile a quello di McCulloch e Pitts.

Utilizza una funzione di attivazione lineare a soglia (scalino).

Addestramento tramite delta rule.

Rete a 2 livelli apprende solo mapping lineari.

MULTILAYER PERCEPTRON (MLP)

Rete feedforward con almeno 3 livelli con funzioni di attivazione non lineari.

UNIVERSAL APPROXIMATION THEOREM

(Teorema d'esistenza di Kolmogorov applicato alle reti neurali da Cybenko 1988 e Hornik 1989)

Una rete neurale multistrato, con un singolo hidden layer è in grado di apprendere qualsiasi funzione continua e di approssimarla con qualsiasi grado di precisione

FORWAD PROPAGATION

Propagazione in avanti delle informazioni, dal livello di input al livello di output.

Numero totale di pesi rete 3 livelli ($d: n_h: s$) = $d \times n_h + n_h \times s + n_h \times s$ (bias)

TRAINING: calcolo dei pesi w che determinano il mapping desiderato tra input e output.

Dimensioni d dello spazio dei pattern = numero di neuroni dell'livello di input

Numero s di classi del classificatore = numero di neuroni del livello di output

Numero n_h di neuroni dei livelli hidden = iperparametro

Il training consiste nel calcolare i pesi che minimizzano l'errore medio calcolato come scostamento tra i vettori di output desiderati e quelli calcolati dalla rete.

Pesi iniziali random nei range $\pm 1/\sqrt{d}$ (input o-o hidden) e $\pm 1/\sqrt{n_h}$ (hidden o-o output)

ERROR BACKPROPAGATION (GRADIENT DESCENT)

$J(w, x) = \frac{1}{2} \sum_{c=1 \dots C} (t_c - z_c)^2$ con w =pesi; x =pattern; t_c =obiettivo; z_c =output

$J(w)$ è la media su tutti i pattern del training set. Ridotto modificando i pesi in direzione opposta al gradiente di J .

Modifica pesi hidden-output: $w_{jk} = w_{jk} + \eta \cdot \delta_k \cdot y_j$

Modifica pesi input-hidden: $w_{ij} = w_{ij} + \eta \cdot \delta_j \cdot x_i$

Se si usano i bias pesi w_{0k} e w_{i0} si aggiornano ponendo $y_0 = 1$ e $x_0 = 1$

η è il **LEARNING RATE**: modificabile adeguatamente durante il training

- Troppo piccolo \rightarrow convergenza lenta

- Troppo grande \rightarrow oscillazione, o divergenza

RETI NEURALI • FEEDFORWARD vs RECURRENT

Le reti neurali sono composte da gruppi di neuroni artificiali organizzati a livelli.

1 LIVELLO DI INPUT o-o n LIVELLI INTERMEDI (HIDDEN) o-o 1 LIVELLO OUTPUT

FEEDFORWARD: le connessioni collegano i neuroni da un livello a quello successivo

RECURRENT: sono previste connessioni di **feedback** verso livelli uguali o precedenti. Reti più complesse, addestramento più complicato. Utili per la gestione di **sequenze**: hanno effetto memoria che considera le informazioni in diversi istanti temporali. Esempio: **Long Short Term Memory (LSTM)** net.

ALGORITMO BACKPROPAGATION (ONLINE)

PATTERN ANALIZZATI SEQUENZIALMENTE • PESI AGGIORNATI DOPO OGNI PATTERN

```

EPOCHE = 0
DO
    EPOCHE++

    PER OGNI PATTERN x DEL TRAINING SET
        FORWARD STEP SUL PATTERN x
        ERRORE TOTALE = ERRORE TOTALE + ERRORE SUL PATTERN x
        BACKWARD STEP (calcolo dei gradienti)
        AGGIORNAMENTO PESI HIDDEN-OUTPUT
        AGGIORNAMENTO PESI INPUT-HIDDEN
        LOSS = ERRORE TOTALE / n
        CALCOLO ACCURACY SU TRAINING SET E VALIDATION SET
    CONTINUA FINO A CONVERGENZA O NUMERO MASSIMO DI EPOCHE
    
```

SOFTMAX

Funzione di attivazione che permette di trasformare i valori di net prodotti dall'ultimo livello della rete in probabilità delle classi.

Né sigmoid né tanh sono interpretabili come probabilità.

$$z_k = f(\text{net}_k) = \frac{e^{\text{net}_k}}{\sum_{c=1 \dots s} e^{\text{net}_c}}$$

CROSS-ENTROPY LOSS

Funzione di loss probabilistica più efficace in problemi di classificazione multi-classe.

$$H(p, q) = - \sum_v p(v) \cdot \log(q(v)) \geq 0$$

$H(p, q)$ minimo quando il vettore target coincide con l'output.

Quando il vettore target per un pattern assume la forma **one-hot** (tutti 0 tranne 1 per la classe corretta) allora

$$J(\mathbf{w}, \mathbf{x}) \equiv H(\mathbf{t}, \mathbf{z}) = -\log(z_g) = -\log\left(\frac{e^{\text{net}_g}}{\sum_{c=1 \dots s} e^{\text{net}_c}}\right) = -\text{net}_g + \log\left(\sum_{c=1 \dots s} e^{\text{net}_c}\right)$$

SOCHESTIC GRADIENT DESCENT (SGD)

PATTERN ANALIZZATI IN ORDINE CASUALE • PESI AGGIORNATI DOPO OGNI BATCH

```

EPOCHE = 0
DO
    EPOCHE++
    * RANDOMSORT (TRAINING SET)
    * PER OGNI MINI-BATCH B DI DIMENSIONE mb DEL TRAINING SET
    *     RESET GRADIENTE
    *     PER OGNI PATTERN x DEL MINI-BATCH B
        FORWARD STEP SUL PATTERN x
        ERRORE TOTALE = ERRORE TOTALE + ERRORE SUL PATTERN x
        BACKWARD STEP (calcolo dei gradienti)
        AGGIORNAMENTO PESI HIDDEN-OUTPUT
        AGGIORNAMENTO PESI INPUT-HIDDEN
        LOSS = ERRORE TOTALE / n
        CALCOLO ACCURACY SU TRAINING SET E VALIDATION SET
    CONTINUA FINO A CONVERGENZA O NUMERO MASSIMO DI EPOCHE
    
```

Se la dimensione dei mini-batch = 1 → caso online

Se la dimensione dei mini-batch = n → full batch

Numero iterazioni totali = dimensione mini-batch × numero mini-batch × epocha

RETI NEURALI

REGOLARIZZAZIONE

Tecniche finalizzate a ridurre il rischio di **overfitting** in caso di molti parametri (pesi). Molto importante quando il training set non ha grandi dimensioni rispetto al modello.

Termine di regolarizzazione $J_{Tot} = J_{Cross-Entropy} + J_{Reg}$

Regolarizzazione L2 $J_{Reg} = \frac{1}{2} \lambda \sum_i w_i^2$ λ, in L1 e L2, regola la forza della regolarizzazione

Regolarizzazione L1 $J_{Reg} = \frac{1}{2} \lambda \sum_i |w_i|$ può essere più «sparsificante»: numerosi pesi a 0

L'aggiornamento dei pesi a seguito di backpropagation include un ulteriore termine denominato **weight decay**

$$w_k = w_k - \eta \cdot \left(\frac{\partial J_{Cross-Entropy}}{\partial w_k} + \lambda \cdot w_k \right)$$

LEARNING RATE ADATTATIVO

MOMENTUM (μ)

SGD con mini-batch può determinare una discesa del gradiente a zig-zag: rallenta la convergenza o la rende instabile

Per evitare oscillazioni si può emulare il comportamento fisico considerando l'aggiornamento precedente per ogni parametro, calcolando il successivo aggiornamento come combinazione lineare dell'aggiornamento precedente e del gradiente attuale.

$$w_k = w_k + \mu \cdot \Delta w_k - \eta \cdot \left(\frac{\partial J_{Tot}}{\partial w_k} \right)$$

μ (iperparametro) ha valore tipico = 0.9



Steps without Momentum



Steps with Momentum

ALTRE REGOLE DI AGGIORNAMENTO PESI

- Nesterov Accelerate Gradient (NAG)
- Adaptive Gradient (Adagrad)
- Adadelta
- Adam (stato dell'arte)
- Rmsprop

GERARCHIA DI LIVELLI

DEEP NEURAL NETWORK (DNN) sono reti «profonde» composte da almeno 2 livelli hidden organizzati gerarchicamente. Superano l'Universal Approximation Theorem in quanto esistono funzioni computabili con complessità polinomiale operando su k livelli che assumono complessità esponenziale su k-1 livelli (Hastad, 1986).

L'**organizzazione gerarchica** consente di condividere e riutilizzare informazioni, selezionando – ad esempio – feature specifiche o scartando dettagli inutili.

LIVELLI E COMPLESSITÀ

Il numero di livelli definisce la complessità di una rete associato al numero di neuroni, connessioni e pesi.

Esempi di reti: **AlexNet** (8 livelli, 650K neuroni, 60M parametri), **VGG-16** (16 livelli, 15M neuroni, 140M parametri), **corteccia umana** (21×10^9 neuroni e 1.5×10^{14} sinapsi).

TIPOLOGIE DEEP NEURAL NETWORK (DNN)

MODELLI FEEDFORWARD

Classificazione o regressione, con training prevalentemente supervisionato

- CONVOLUTIONAL NEURAL NETWORK (CNN)
- FULLY CONNECTED DNN (FC DNN)
- HIERARCHICAL TEMPORAL MEMORY (HTM)

MODELLI GENERATIVI

Training non supervisionato, utili per pre-training di altri modelli e per produrre feature rilevanti

- STACKED (DE-NOISING) AUTO-ENCODERS
- RESTRICTED BOLTZMANN MACHINE (RBM)
- DEEP BELIEF NETWORKS (DBN)

MODELLI RICORRENTI

Utilizzati per sequenze, speech recognition, sentiment analysis, Natural Language Processing

- RECURRENT NEURAL NETWORK (RNN)
- LONG SHORT-TERM MEMORY (LSTM)

REINFORCEMENT LEARNING

Utilizzati per apprendere comportamenti

- DEEP Q-LEARNING

INGREDIENTI

BIG DATA + GPU COMPUTING + VANISHING (OR EXPLODING) GRADIENT

CONVOLUTIONAL NEURAL NETWORK (CNN)

Introdotte da LeCun et.all nel 1998, usano le reti cercando di simulare due tipi di capacità: feature (simple) detection e pooling (fusion) degli output della simple detection.

PREPROCESSING LOCALE: i neuroni sono connessi solo localmente (riduzione connessioni)

CONDIVISIONE PESI: i pesi sono condivisi a gruppi (neuroni diverse/stesse elaborazioni)

ALTERNAZA LIVELLI: feature extraction e pooling.

ARCHITETTURA

INPUT LAYER: mappa direttamente i pixel dell'immagine

HIDDEN LAYER: feature extraction e pooling

OUTPUT LAYER: mappa le classi

CONVOLUZIONE

Operazione di image processing che, tramite l'applicazione di filtri digitali, condensa le informazioni di un'area di pixel in un valore attraverso un prodotto scalare tra una maschera e l'area di pixel in input. Un neurone al primo livello hidden opera come filtro di convoluzione direttamente sui pixel dell'immagine (neuroni input layer).

VOLUMI

I neuroni di ciascun livello sono organizzati in griglie o volumi 3D. I filtri operano su una porzione dei volumi di input.

Ogni «fetta» di volumi definisce una **feature map** (es. componenti RGB in input).

I pesi sono condivisi a livello di **feature map**.

APPLICAZIONE FILTRI

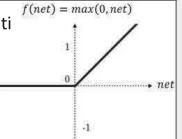
STRIDE: passo con il quale si fa scorrere un filtro, passi più alti riducono dimensioni output

PADDING: spessore, in pixel, del bordo aggiunto (valori 0)

$$w_{out} = \frac{(w_{in} - F + 2 \cdot Padding)}{Stride} + 1 \quad \begin{matrix} w_{in} & \text{dimensione input} \\ Stride & \end{matrix} \quad \begin{matrix} F & \text{dimensione filtro} \\ & \end{matrix} \quad \begin{matrix} w_{out} & \text{dimensione output} \\ & \end{matrix}$$

RECTIFIED LINEAR (RELU) funzione di attivazione

L'utilizzo di RELU come funzione di attivazione risolve i problemi legati all'uso della sigmoide (riduzione significativa del valore del gradiente lontano dall'output, annullamento gradiente in regioni saturate per derivata = 0).



Derivata = 0 per valori negativi e 1 per valori positivi di net.

Nessuna saturazione sui valori positivi.

Attivazioni sparse (neuroni spenti) possono conferire maggiore robustezza.

DEEP LEARNING

CNN POOLING

Un livello di POOLING esegue un'aggregazione delle informazioni del volume di input. Genera feature map dimensione inferiore cercando di conferire **invarianza**.

Opera generalmente a livello di ogni feature map del volume di input.

Gli operatori più utilizzati sono la **media** e il **massimo**.

Non ci sono parametri da apprendere.

SOFT MAX E CROSS ENTROPY

Nelle moderne CNN per classificazione è comune trovare un livello finale costituito da un neurone per ogni classe con funzione di attivazione **SoftMax** o **Cross-Entropy** per garantire di ottenere valori interpretabili come probabilità.

IN PRATICA

Implementazione da zero: training & inference architetture + training backpropagation del gradiente + ottimizzazione GPU = molte risorse tempo per sviluppo e debug.

Framework: Caffe(Berkeley), Theano(Montreal), Torch(LeCun), TensorFlow(Google), Digits(Nvidia)

ARCHITETTURA ESEMPIO

Livello Funzione	Parametri	Dimensioni (Neuroni)	Connessioni e pesi (entranti)
INPUT		32x32x3	
HIDDEN CONVOLUZIONE	Filtri: 5x5 • FeatureMaps: 16 • Stride: 1 • Padding: 2 • Attivazione ReLu	32x32x16	C: 5x5x3x16x32x32 P: 16x(5x5x3+1)
HIDDEN POOLING	Filtri: 2x2 • Stride: 2 • Tipo: Max	16x16x16	C: 16x16x16x2x2x1 P: 0
HIDDEN CONVOLUZIONE	Filtri: 5x5 • FeatureMaps: 20 • Stride: 1 • Padding: 2 • Attivazione ReLu	16x16x20	C: 5x5x16x16x20x20 P: 20x(5x5x16+1)
HIDDEN POOLING	Filtri: 2x2 • Stride: 2 • Tipo: Max	8x8x20	C: 8x8x20x2x2x1 P: 0
HIDDEN CONVOLUZIONE	Filtri: 5x5 • FeatureMaps: 20 • Stride: 1 • Padding: 2 • Attivazione ReLu	8x8x20	C: 8x8x20x5x5x20 P: 20x(5x5x20+1)
HIDDEN POOLING	Filtri: 2x2 • Stride: 2 • Tipo: Max	4x4x20	C: 4x4x20x2x2x1 P: 0
OUTPUT	SofMax con 10 classi	10	C: 10x4x4x20 P: 10x(4x4x20+1)

TRANSFER LEARNING

FINE TUNING: si sostituisce il livello di output di una rete già addestrata adeguando il numero di classi, si utilizzano i valori iniziali dei pesi della rete pre-addestrata tranne che per le connessioni al livello di output e si eseguono nuove iterazioni di addestramento.

RIUTILIZZO FEATURES: si estraggono le feature generate durante l'addestramento di una rete esistente e le si utilizzano per addestrare un classificatore esterno; le feature sono generate nei i livelli intermedi (Es. L⁵, L⁶, L⁷ di CaffeNet).

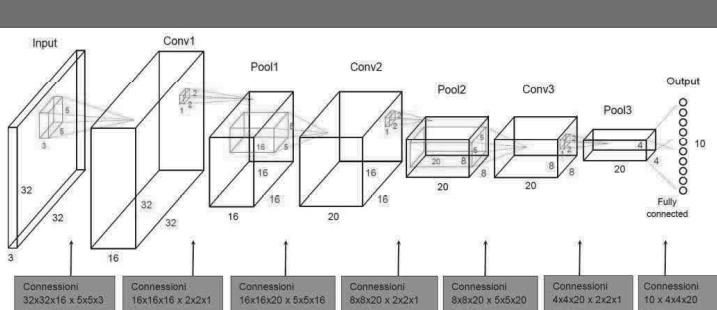
REINFORCEMENT LEARNING/DEEP QLEARNING

L'obiettivo di apprendere un comportamento è determinato dall'individuazione delle attività che massimizzano i reward. In molti casi reali applicati ad ambienti stocastici (in cui sequenze di azioni identiche portano a risultati diversi) si applica una logica di convenienza immediata pesando maggiormente i reward vicini nel tempo: $R_t = r_t + \gamma^1 \cdot r_{t+1} + \dots + \gamma^{n-t} \cdot r_n$ con $0 \leq \gamma \leq 1$

Q-LEARNING: $R_t = r_t + \gamma \cdot R_{t+1}$ definito ricorsivamente, sfrutta l'equazione di Bellman basata su una funzione definita a partire dallo stato iniziale $Q(s_t, a_t) = r_{t+1} + \gamma \cdot \max_a Q(s_{t+1}, a)$.

L'apprendimento avviene iterando una serie di episodi durante i quali, ad ogni passo, si seleziona l'azione ottimale e si osserva l'esito delle scelte allo stato successivo. Utilizza un valore di learning rate che definisce la «grandezza» dei passi in funzione delle scelte fornite dall'equazione di Bellman; progressivamente ridotto durante l'apprendimento. Dimensioni di Q=possibili stati x possibili azioni.

DEEP Q-LEARNING: CNN usata per approssimare funzioni Q di enormi dimensioni.



31.562 NEURONI TOTALI (incluso livello input)

3.942.784 CONNESSIONI TOTALI

22.466 PESI TOTALI (inclusi bias)

Introduzione

1. Indicare le principali “stagioni” nello sviluppo dell’intelligenza artificiale e machine learning.

- **1940 - 1974: la nascita e gli anni d’oro**

Primi calcolatori elettronici, test di Turing, grande entusiasmo e predizioni troppo ottimistiche

- **1974 - 1980: il primo inverno**

Drastica riduzione dei finanziamenti a causa di vari problemi: scarsa capacità computazionale, esplosione combinatoria e non trattabilità, dataset di piccole dimensioni

- **1980 - 1987: nuova primavera**

Nascita dei sistemi esperti (conoscenza + regole logiche), algoritmo di Backpropagation per reti neurali, finanziamento del governo giapponese per la quinta generazione di calcolatori

- **1987 - 1993: il secondo inverno**

Flop della quinta generazione e nuovo calo di finanziamenti a causa dei seguenti problemi: hardware specializzato non competitivo con quello PC, risultati concreti solo in specifici campi, reti neurali non scalabili a problemi complessi

- **1993 - 2011: tempi moderni**

Hardware sempre più prestanti, Bayesian Networks, SVM, Multi-classificatori (Random Forest, AdaBoost), successi in numerose discipline

- **2011 - oggi: deep learning**

Nasce CNN (anche se non ha grande successo a causa della mancanza di big data e potenza di calcolo), speech recognition, language translation, grandi miglioramenti delle tecniche di deep learning in varie applicazioni, investimenti sostanziosi da parte dei big dell’ICT

2. Definire cosa si intende con i termini Intelligenza, Intelligenza Artificiale e Machine Learning

L’intelligenza è un insieme di facoltà mentali che permettono all’uomo di pensare, comprendere, elaborare modelli di realtà e adattarsi alle situazioni. L’intelligenza artificiale è la capacità di riprodurre parzialmente le attività intellettuali dell’uomo, principalmente quelle riguardanti apprendimento, riconoscimento e situazioni decisionali. Machine learning è una disciplina dell’intelligenza artificiale che, mediante una fase di training, apprende un comportamento corretto sfruttando degli esempi per poi riuscire a gestire autonomamente dati dello stesso dominio applicativo

3. Cosa si intende con il termine singolarità tecnologica?

È un punto previsto nello sviluppo tecnologico dove il progresso supererà le capacità di comprensione e previsione dell’uomo. Nel momento in cui un calcolatore supererà l’intelligenza umana, si potrà demandare ad esso di progettare nuovi calcolatori.

4. Fare esempi pratici di ragionamento induttivo e deduttivo.

conclusione

In un ragionamento deduttivo la verità delle premesse garantisce la verità delle conseguenze, in quello induttivo la verità delle premesse rappresenta un caso specifico e quindi non garantisce la verità delle conseguenze. Ad esempio, tutti gli uomini sono mortali, io sono uomo, io morirò. Io sono morto, io sono un uomo, gli uomini sono mortali. Nel ragionamento induttivo, diversamente da quello deduttivo, le premesse caso/i particolare/i forniscono un’evidenza più o meno forte a sostegno della conclusione generalizzazione ma non ne garantiscono necessariamente la verità

5. Cosa si intende per risoluzione dei problemi con approccio “forza bruta”. Si tratta di intelligenza artificiale?

Con il termine *brute force* si indica la capacità di un calcolatore di risolvere problemi semplicemente provando tutte le possibili soluzioni. Non si tratta di intelligenza artificiale vera e propria, tendenzialmente si parla infatti di *weak AI*, ovvero la capacità di risolvere problemi senza però dimostrare intelligenza e ragionamenti.

Fondamenti

6. Dare la definizione di representation learning

Representation learning è la capacità di un sistema di apprendere automaticamente delle feature efficaci a partire dai raw data, come per esempio dall'intensità dei pixel di un'immagine o l'ampiezza di un segnale audio nel tempo, senza usare feature predefinite. Gran parte delle tecniche di deep learning operano in questo modo, utilizzando come input i raw data ed estraendo automaticamente da essi le feature necessarie per risolvere il problema di interesse.

7. Nell'ambito del dominio di visione indicare e descrivere le diverse tipologie di problemi

- Classificazione: assegnare ad un oggetto la giusta classe di appartenenza
- Localizzazione: determinare la classe e la sua posizione all'interno di un box
- Detection: localizzazione per ogni oggetto nell'immagine
- Segmentation: analogo a detection, ma utilizza dei box per identificare la posizione, bensì vengono colorati diversamente in base alla classe i pixel di contorno.

8. Descrivere le tipologie di addestramento batch, incrementale e naturale

- Batch: il training viene effettuato una sola volta e al termine il sistema entra in opera. Effettuato per la maggior parte dei sistemi.
- Incrementale: viene fatto un primo training al quale ne possono seguire altri, potrebbe verificarsi un *catastrophic forgetting*, ovvero il sistema potrebbe dimenticare ciò che ha appreso in precedenza
- Naturale: l'addestramento è sempre continuo. Continua ad apprendere anche mentre lavora, come l'uomo

9. Che cos'è e come è organizzata la Matrice di Confusione?

È una metodologia utilizzata per conoscere la distribuzione e la concentrazione di errori nella classificazione. Le colonne della matrice rappresentano le previsioni dell'algoritmo, mentre le righe quelle reali. In poche parole, la diagonale contiene le classificazioni corrette, i valori più alti devono trovarsi lì. Valori elevati fuori dalla diagonale indicano una concentrazione di errori.

10. Che cosa si intende con problemi closed e open set

Nei problemi closed set ciascun pattern da classificare appartiene sempre ad una classe del training set. Nei problemi open set un pattern può non appartenere a nessuna classe del training set, in tal caso si può creare un classe fittizia nel quale inserire tutti i pattern che non hanno classe o lasciare la possibilità al classificatore di non assegnargli alcuna classe.

11. A che cosa serve la metrica Precision-Recall? Indicare la formula di entrambe le notazioni

La metrica precision-recall è una notazione molto utilizzata in information retrieval e in generale nelle applicazioni di detection. Precision indica quanto è accurato il sistema (ad esempio, possiamo chiederci che percentuale di documenti selezionati è pertinente), mentre recall quanto è selettivo (di tutti i documenti pertinenti, quanti ne sono stati selezionati). $Precision = \frac{TP}{TP+FP}$; $Recall = \frac{TP}{TP+FN} = \frac{TP}{P} = TPR$ dove TP sta per true-positive, FP per false-positive e FN per false-negative.

12. Dare una definizione di convergenza

La convergenza è l'obiettivo principale da conseguire nel training. Si ha convergenza se il loss (output della loss function) ha andamento decrescente e l'accuratezza ha andamento crescente. Se il loss non cala o oscilla significativamente la causa può essere il metodo di ottimizzazione non efficace, il learning rate inadeguato, gli iperparametri fuori range ecc. Se l'accuratezza non cresce probabilmente è errata la loss-function scelta.

L'accuratezza deve avvicinarsi al 100%, se non lo fa potrebbero esserci troppi pochi gradi di libertà per gestire la complessità del problema.

13. Dare la definizione di training, validation e test set e discutere una possibile suddivisione dei dati nei tre insiemi

- **Train set:** insieme dei pattern sui quali vengono tarati i parametri
- **Validation set:** insieme dei pattern sui quali vengono tarati gli iperparametri
- **Test set:** insieme dei parametri sui quali vengono valutate le prestazioni del sistema. Bisogna evitare di tarare gli iperparametri su questo set poiché si incorre in una sovrastima delle prestazioni

Nei benchmark di machine learning la suddivisione dei pattern in train, valid e test è spesso predefinita, per rendere confrontabili i risultati. Consideriamo un caso con 12000 pattern:

- **Set disgiunti:** 10000 train, 1000 valid e 1000 test
- **K-fold cross-validation:** 2000 test, K = 5 partizioni (fold) da 2000 ciascuna e si esegue per cinque volte il training scegliendo una delle partizioni come valid e le rimanenti come train. Le prestazioni finali corrisponderanno alla media/mediana di tutte le prestazioni
- **Leave-one-out:** caso estremo di cross-validation dove le partizioni hanno dimensione 1. Visto il costo computazionale, si usa quando i pattern sono pochi (< 100)

14. Definire i problemi di Classificazione e Regressione evidenziandone le differenze e fornendo per ciascuno esempi reali della loro applicazione.

- **Classificazione:** assegna una classe ad ogni pattern, è necessaria una funzione capace di eseguire il mapping dallo spazio dei pattern a quello delle classi. Ad esempio, in quanti modi diversi può essere scritto il carattere "A" a mano libera. I principali problemi di classificazione sono: spam detection, credit card fraud detection, face recognition, pedestrian classification, medical diagnosis e stock trading
- **Regressione:** assegna un valore continuo a un pattern. Risolvere un problema di regressione corrisponde ad apprendere una funzione approssimante delle coppie I/O date. I principali problemi di regressione sono: stima prezzi vendita appartamenti nel mercato immobiliare, stima del rischio per compagnie assicurative, predizione energia prodotta da impianto fotovoltaico, modelli sanitari di predizione dei costi e object detection

15. Fare esempi di pattern numerici, categorici e sequenze

- **Numerici:** valori associati a caratteristiche misurabili o conteggi, sono tipicamente continui e soggetti a ordinamento. Sono rappresentabili naturalmente come vettori numerici nello spazio multidimensionale. L'estrazione di caratteristiche da segnali di input produce vettori numerici chiamati feature vectors. Ad esempio, le caratteristiche fisiche di una persona
- **Categorici:** valori associati a caratteristiche qualitative e alla presenza/assenza di una determinata caratteristica. Non sono semanticamente mappabili, talvolta soggetti a ordinamento e normalmente gestiti da sistemi a regole e alberi di classificazione. Ad esempio, i tratti distintivi di una persona
- **Sequenze:** pattern sequenziali con relazioni spaziali o temporali. Sono di lunghezza molto variabile. Inoltre, la posizione nella sequenza e le relazioni con predecessori e successori sono importanti. Ad esempio, stream audio, video o frasi sono sequenze di dati

16. Nell'ambito dell'apprendimento automatico cosa si intende per generalizzazione, overfitting e convergenza?

- **Generalizzazione:** capacità di trasferire l'elevata accuratezza raggiunta nel train sul valid.
- **Overfitting:** al contrario della generalizzazione vi è l'impossibilità di trasferire l'accuratezza. Solitamente l'overfitting è causato da un training set inadeguato, tipicamente troppo piccolo, o dalla

presenza di gradi di libertà troppo elevati. Un buon approccio è partire da gradi di libertà piccoli e incrementarli man mano, monitorando l'accuratezza su train e valid, così da trovare un valore sufficientemente elevato, ma che non causi overfitting.

- **Convergenza:** è il primo obiettivo da perseguire durante l'addestramento sul train set. Se consideriamo un classificatore il cui addestramento prevede un processo iterativo, si ha convergenza quando
 - Il loss ha andamento decrescente
 - L'accuratezza ha andamento crescente

17. Cosa si intende per iperparametri? Fornire esempi pratici di iperparametri. Come si ottimizzano?

Molti algoritmi richiedono di definire, prima dell'apprendimento vero e proprio, il valore dei cosiddetti iperparametri. Alcuni esempi sono il numero di vicini presi in considerazione in k-NN, il numero di neuroni di una rete e il grado del polinomio nella regressione. Si procede con un approccio a due livelli nel quale per ogni valore ragionevole degli iperparametri si esegue l'apprendimento e, al termine della procedura, si scelgono gli iperparametri che hanno fornito prestazioni migliori. Nei problemi di classificazione si usa la formula delle prestazioni di un classificatore, mentre per la regressione RMSE.

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1..N} (pred_i - true_i)^2}, \text{ ovvero la radice della media dei quadrati degli scostamenti tra valore vero e valore predetto.}$$

18. Descrivi brevemente le diverse tipologie di apprendimento

L'apprendimento può essere svolto con modalità diverse, suddivisibili secondo diversi criteri come il tipo di training set utilizzato o il numero di ripetizioni dell'addestramento stesso. In base alla tipologia di training set utilizzato si possono avere 3 tipi di apprendimento:

- **Supervisionato:** il training set è etichettato, quindi sono note le classi dei pattern utilizzati per il training. Usato tipicamente in classificazione, regressione, LDA
- **Non supervisionato:** il training set non è etichettato, quindi non sono note le classi dei pattern utilizzati. Usato tipicamente in clustering e riduzione dimensionalità
- **Semi-supervisionato:** il training set è etichettato parzialmente

In base al numero di ripetizioni con cui viene effettuato l'addestramento, possiamo avere 3 modalità:

- **Batch:** l'addestramento viene effettuato solo una volta su un training set dato, dopodiché il sistema entra in working mode e non è più in grado di apprendere
- **Incrementale:** dopo un addestramento iniziale, sono possibili ulteriori sessioni di addestramento
- **Naturale:** addestramento continuo e attivo anche in working mode, con coesistenza di approccio supervisionato e non
- **Reinforcement learning:** addestramento basato sui comportamenti ottimali a partire da esperienze passate, poiché nella pratica l'approccio supervisionato non è facilmente applicabile questo approccio viene usato soprattutto in robotica e Q-learning

19. Come si misurano le prestazioni di un classificatore?

In un problema di classificazione, l'accuratezza è la percentuale di pattern correttamente classificati. L'errore di classificazione è il complemento. $Accuratezza = \frac{\text{pattern correttamente classificati}}{\text{pattern classificati}}; Errore = 100\% - Accuratezza$

20. Come si valutano le prestazioni degli algoritmi?

Una possibilità per valutare le prestazioni degli algoritmi consiste nell'utilizzare la funzione obiettivo. In genere però si preferisce una misura collegata alla semantica del problema. In un problema di classificazione per esempio, l'accuratezza di classificazione è la percentuale di pattern correttamente classificati.

21. Cosa si intende per parametri, funzione obiettivo e loss function negli algoritmi di apprendimento?

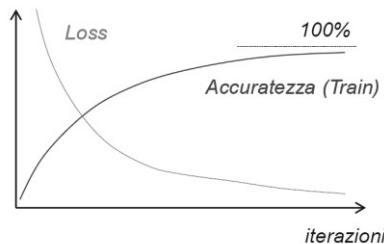
Generalmente, il comportamento di un algoritmo di machine learning è regolato da un set di parametri θ (ad esempio, i pesi delle connessioni di una rete neurale). L'apprendimento consiste nel determinare il valore ottimo di questi parametri. La funzione obiettivo, invece, dato un training set e un set di parametri, può indicare:

- Ottimalità della soluzione, da massimizzare
- Errore o perdita (loss-function), da minimizzare

Inoltre, la funzione obiettivo può essere ottimizzata esplicitamente (con metodi derivati dalla propria definizione matematica) o implicitamente (con metodi euristici) e utilizzata per misurare le prestazioni per la definizione degli iperparametri.

22. Cosa si intende per convergenza di un algoritmo di apprendimento iterativo? Accuratezza e loss come si comportano durante le iterazioni in caso di convergenza. Disegnare un semplice grafico.

Considerando un classificatore il cui addestramento prevede un processo iterativo, si ha convergenza sul training set quando il loss (output della loss-function) ha andamento decrescente, mentre l'accuratezza ha andamento crescente. Se questo non avviene il sistema non converge perché il metodo di ottimizzazione non è efficace oppure è stata scelta una loss-function errata o i gradi di libertà del classificatore non sono sufficienti per gestire la complessità del problema.



Classificazione

23. Indicare le differenze tra l'approccio parametrico e non parametrico nel classificatore di Bayes

- **Approccio parametrico:** effettua un'ipotesi sulle possibili distribuzioni, apprendendo vettore medio e matrice di covarianza, da ciò ricava poi le probabilità a priori, condizionale e a posteriori. Si usa se si ha una certezza elevata che la distribuzione predetta sia quella corretta o se la dimensione del training set non è sufficiente per ricavare la distribuzione effettiva es. Parzen Window
- **Approccio non parametrico:** ricava la distribuzione a partire dal training set, quindi è possibile tendenzialmente solo in spazi di dimensione ridotta (tipo $d=3$). ~~Può essere semplificato ad una variabile binomiale~~

24. Quali sono i principali vantaggi e le principali limitazioni del classificatore di Bayes con approccio parametrico?

L'approccio parametrico prevede che si facciano ipotesi sulla forma delle distribuzioni e che si apprendano i parametri fondamentali (vettore medio, matrice di covarianza) dal training set. In genere, l'approccio parametrico si utilizza quando si ha una ragionevole certezza (speranza) che la distribuzione sia adeguata e quando la dimensione del training set non è sufficiente per una stima delle densità. I vantaggi di questo approccio sono:

- Minor numero di gradi di libertà, quindi minor rischio di overfitting

DA VERIFICARE NELLA SLIDE 20 CALSSIFICAZIONE 1

- Produce un valore di output probabilistico, che può essere utilizzato come confidenza di classificazione

Mentre gli svantaggi sono:

- Spesso si fanno ipotesi azzardate sulla normalità delle densità di probabilità delle classi, ciò porta ad ottenere cattivi risultati
- Dato un problema di classificazione deve essere innanzitutto valutata la "normalità" delle distribuzioni, in modo formale/matematico o in modo empirico

25. Nel classificatore di Bayes cosa si intende per densità di probabilità condizionale e probabilità a priori.

La probabilità a priori $p(w_i)$ è la probabilità che il prossimo pattern classificato appartenga ad una determinata classe, indipendentemente da altri fattori, di solito corrisponde al numero di pattern di tale classe diviso il numero di pattern totali. La probabilità condizionale si indica $p(x|w_i)$ e indica la probabilità che il prossimo pattern classificato sia X sapendo che la sua classe è w_i . Nel classificatore di Bayes ci si interessa a quella a posteriori $p(w_i|x)$, ovvero la probabilità che sapendo di aver osservato x , la sua classe sia w_i . In definitiva viene scelta la classe con probabilità a posteriori massima. non richiesto in questa domanda

26. A che cosa serve la distanza di Mahalanobis?

La distanza di Mahalanobis r tra x e μ è definita dall'equazione $r^2 = (x - \mu)^t \Sigma^{-1} (x - \mu)$. Viene usata spesso in sostituzione alla distanza euclidea in quanto calcola la distanza, ma permette di pesare ogni componente in base allo spazio di variazione tenendo conto delle correlazioni. Viene sfruttata anche da Bayes.

27. Cosa sono le superfici decisionali? In che modo vengono usate?

Quando la classificazione viene eseguita con la regola di Bayes, lo spazio viene suddiviso in regioni non connesse. Una superficie decisionale (decision boundary o decision surface) è una zona di confine tra regioni di spazio che il classificatore associa a classi diverse. Le superfici decisionali possono assumere forme diverse:

- Se le 2 matrici di covarianza sono uguali la superficie decisionale è un iperpiano
- Se le 2 matrici di covarianza sono arbitrarie la superficie decisionale è un'iperquadratica

28. Qual è il principale vantaggio dell'approccio Bayesiano rispetto ad altri classificatori?

Il principale vantaggio di Bayes, rispetto ad altri classificatori, è che produce un output probabilistico, ovvero tutti valori compresi tra 0 e 1 con somma totale 1. Questo risulta vantaggioso per valutare la confidenza così come per costituire dei multiclassificatori basati su Bayes. ampliabile con slide 19 classificazione 1

29. Qual è il significato del termine curse of dimensionality e con quali tecniche può essere contrastato.

Questo problema si verifica spesso se ci si trova in spazi di dimensione molto elevata. In questi spazi infatti i pattern probabilmente saranno sparsi, perdendo ogni valore statistico/probablistico e richiedendo training set molto più complessi, longevi e con un numero di elementi elevato, in determinati casi anche impossibili da portare a termine. Una delle tecniche per evitare la curse of dimensionality è la riduzione della dimensionalità.

30. La formula di distanza di un pattern dall'iperpiano risultante dal training di un SVM dipende da tutti i pattern del training set o solo da una parte di questi? Motivare la risposta.

L'intera soluzione di SVM può essere descritta dai soli pattern che giacciono sul margine, detti support vectors, ovvero dai pattern di ciascuna classe più vicina all'iperpiano. La distanza di tutti gli altri pattern dall'iperpiano dipende quindi solo da essi.

31. A che cosa serve il metodo Parzen Window e qual è l'idea alla sua base.

Il metodo Parzen Window è un metodo che classifica i pattern con approccio non parametrico e attraverso un' stima delle densità di probabilità. L'algoritmo permette di calcolare la probabilità di appartenere ad una classe C , considerando la densità di C in una finestra costituita da un ipercubo centrato in x e avente lato h . La dimensione fissa del volume della finestra (quindi dell'iperparametro h) ha un forte impatto sul risultato: se la finestra è troppo piccola, la stima risulta instabile e si rischia l'overfitting; se la finestra è troppo grande, la stima è vaga e con errori nelle zone più dense. Per questo motivo, una finestra dinamica (come in k-NN), può dare risultati migliori. Nella pratica, si utilizzano kernel function più soft, in questo modo le superfici decisionali risultano più regolari. Le kernel function devono essere funzioni con densità sempre ≥ 0 e con integrale su tutto lo spazio uguale a 1.

32. Quali sono due aspetti negativi di Nearest-Neighbour e con quale altra tecnica uno di questi può essere evitato?

I problemi principali di NN riguardano la tassellazione di Voronoi. Ogni pattern determina un tassello nel quale tutti i pattern che ricadono in esso assumeranno la classe di tale pattern. Se nel training set si avesse quindi un pattern incerto, questo potrebbe comportare la classificazione errata di molteplici pattern adiacenti. Inoltre, NN, è computazionalmente molto costoso poiché viene calcolata la distanza di ciascun pattern da classificare rispetto a tutti i pattern del TS, evitabile adottando dei prototipi (pattern rappresentativi per ogni classe). Inoltre, per evitare problemi legati a diversi spazi di variazioni delle feature, si consiglia di normalizzare i pattern. Le tecniche di normalizzazione più comuni sono: min-max scaling, standardization e whitening transform.

33. Nel caso di pattern non linearmente separabili, nella formulazione di SVM lineare come si approccia il problema?

In un approccio SVM lineare con pattern non linearmente separabili vi saranno inevitabilmente errori di classificazione nel training set, non esistendo alcun iperpiano in grado di separare i pattern. In questo caso è necessario rilassare i vincoli di separazione per far sì che alcuni pattern (il minor numero possibile) possano oltrepassare il confine della classe. A tal fine si introducono n variabili di slack positive $\zeta_i, i = 1 \dots n$ e si modificano i vincoli di separazione $y_i[w^T x_i + b] \geq 1 - \zeta_i$ con $i = 1 \dots n$, ovvero i vincoli che impongono che tutti i pattern appartenenti a una classe abbiano distanza positiva dall'iperpiano^{di} separazione, mentre quelli dell'altra classe distanza negativa. A ciascuno di questi vincoli viene introdotta una variabile di slack che indica di quanto il pattern si allontana dall'iperpiano in direzione opposta a quella corretta. Avrà quindi valore 0 per i pattern classificati correttamente e positivo quelli classificati erratamente. L'obiettivo diventa quindi trovare un iperpiano di separazione che massimizzi il margine e che riduca al minimo gli errori. In questa applicazione, deve essere definito un parametro C che rappresenta il peso dato agli errori.

34. Qual è il principio su cui si basa il classificatore k-NN? Quali sono le problematiche relative a NN e alla scelta della metrica?

La regola k-NN prima determina i k elementi più vicini al pattern x da classificare (k è un iperparametro); poi ogni pattern tra i k vicini vota per la classe a cui esso stesso appartiene. Infine, il pattern x viene assegnato alla classe che ha ricevuto il maggior numero di voti. Il comportamento della regola k-NN è strettamente legato alla metrica (funzione distanza) adottata, la più comune è la distanza euclidea. Nella pratica è bene valutare lo spazio di variazione delle componenti (feature) e la presenza di forti correlazioni tra esse, affinché una delle componenti non abbia un peso maggiore nel processo di classificazione e ci sia indipendenza tra esse. Per evitare problemi legati a diversi spazi di variazione delle feature, ogni feature può essere normalizzata attraverso l'uso di pesi scelti in base al potere discriminante delle feature. Alcuni metodi di normalizzazione che tengono conto anche delle correlazioni tra feature sono:

- Pre-normalizzazione a priori dello spazio delle feature attraverso whitening transform con conseguente utilizzo della distanza euclidea come metrica

- Utilizzare come metrica la distanza di Mahalanobis, la quale normalizza ogni componente sulla base della matrice di covarianza e pesa maggiormente feature non correlate

35. Perché l'utilizzo dei classificatori di tipo NN può diventare problematico dal punto di vista computazionale?

L'utilizzo di un classificatore NN o k-NN può diventare problematico nel caso di training set di elevate dimensioni, in quanto è necessario memorizzare tutti i pattern del training set e per ogni classificazione bisogna calcolare la distanza del pattern da classificare rispetto tutti gli altri e poi ordinare le distanze per ottenere le più piccole. Per alleviare questo problema esistono diverse soluzioni:

- Individuare i pattern attraverso strutture dati (es. kd-tree) che consentono di individuare i vicini senza effettuare una scansione esaustiva
- Utilizzare tecniche di editing/condensing per selezionare/derivare dai pattern dei prototipi per ciascuna classe e usare questi per calcolare le distanze (evitando troppi calcoli e outlier pericolosi)

36. Come è possibile estrarre una confidenza circa la classificazione eseguita in K-NN?

Da un classificatore k-NN risulta piuttosto semplice estrarre una confidenza circa la classificazione eseguita. Siano $[v_1, v_2 \dots v_s]$, $\sum_{i=1}^s v_i = k$ i voti ottenuti dal pattern x , allora le confidenze possono essere semplicemente ottenute dividendo per k i voti ottenuti: $[\frac{v_1}{k}, \frac{v_2}{k} \dots \frac{v_s}{k}]$

37. Qual è la differenza fra un classificatore NN e uno k-NN? In quali circostanze è preferibile usare k-NN?

Data una metrica $dist(\cdot)$, il classificatore Nearest Neighbor (NN) classifica un pattern x con la stessa classe dell'elemento x' ad esso più vicino nel training set: $dist(x, x') = \min_{x_i \in TS} \{dist(x, x_i)\}$. La regola nearest neighbor produce un partizionamento dello spazio (tassellazione di Voronoi), per cui ogni elemento x_i determina un tassello, all'interno del quale i pattern saranno assegnati alla stessa classe di x_i . La regola di NN è piuttosto radicale, infatti basta che un pattern non sia affidabile (outlier) affinché tutti i pattern nelle vicinanze non siano etichettati correttamente. Per questo motivo è stato introdotto un metodo più robusto, ovvero il classificatore k-Nearest Neighbor (k-NN). La regola k-NN prima determina i k elementi più vicini al pattern x da classificare (k è iperparametro); poi ogni pattern tra i k vicini vota per la classe a cui esso stesso appartiene. Infine, il pattern x viene assegnato alla classe che ha ricevuto il maggior numero di voti. Teoricamente per training set infiniti la regola k-NN si dimostra migliore di NN. Inoltre, all'aumentare di k l'errore converge all'errore Bayesiano. Nella pratica però, aumentare k significa estendere troppo la sfera di ricerca, quindi, essendo un iperparametro, il valore ottimale di k deve essere tarato in un validation set separato.

38. Indicare a cosa servono e descrivere le tecniche di editing e condensing

Sono tecniche utilizzate per ottenere dei prototipi, ovvero pattern rappresentativi di un'intera classe di NN, in modo da dover calcolare la distanza dei pattern da classificare solamente dai prototipi e non da tutti i pattern del TS.

- **Editing:** si cancellano solo pattern dal training set, senza derivarne di nuovi
- **Condensing:** i prototipi non appartenevano al training set e sono stati derivati

L'uso di queste tecniche permette di evitare di calcolare un elevato numero di distanze. Inoltre, i prototipi sono spesso più affidabili e robusti di singoli pattern.

39. Che cos'è la normalizzazione e quali sono le tecniche più comuni?

Per evitare i problemi legati a diversi spazi di variazioni delle feature, particolarmente fastidiosi per alcune tecniche, si consiglia di normalizzare i pattern, uniformando gli spazi. Le normalizzazioni più comuni sono:

- **Min-max scaling:** per ogni feature i -esima si calcolano il massimo e il minimo, applicando una trasformazione lineare che tipicamente mappa il minimo a 0 e il massimo a 1. $x' = \frac{(x - min_i)}{(max_i - min_i)}$

- **Standardization:** per ogni feature i -esima si calcola la media (mean), la deviazione standard (stddev) e si trasformano i valori. Dopo la trasformazione tutte le feature hanno (sul training set) media pari a 0 e deviazione standard 1. $x' = \frac{(x - \text{mean}_i)}{\text{stddev}_i}$
- **Whitening transform:** tecnica di normalizzazione che opera simultaneamente su tutte le feature, tenendo conto della loro correlazione. Analogia all'adozione come metrica della distanza di Mahalanobis
Alternativa equivalente è utilizzare

40. Che cos'è il Metric Learning?

Il metric learning è un approccio più generale alla scelta della metrica da utilizzare in una determinata applicazione, consiste nel learning supervisionato della metrica stessa dai dati del training set. L'obiettivo è quello di determinare una trasformazione degli input che allontanino pattern diverse e avvicini quelli appartenenti alla stessa classe. Un tipico approccio di metric learning lineare determina una matrice che trasforma gli input e continua ad applicare la distanza euclidea agli input trasformati.

41. Elenca e descrivi brevemente le diverse tipologie di SVM. introduzione si può omettere

Se la superficie di separazione è un iperpiano allora si parla di SVM lineare. Inoltre, se esiste per ipotesi almeno un iperpiano in grado di separare i pattern del training set, allora i pattern si dicono linearmente separabili. Mentre se non esiste un iperpiano in grado di separarli ci saranno inevitabilmente errori di classificazione, allora i pattern si dicono linearmente non separabili. Se la superficie di separazione è complessa si parla di SVM non lineare, senza alcuna ipotesi sulla separabilità dei pattern. Se un problema di classificazione presenta più di 2 classi, SVM necessita di un'estensione multiclass. Nella pratica, se la dimensionalità dello spazio è elevata si utilizza SVM lineare, altrimenti non lineare.

- **SVM lineare, pattern separabili:** l'iperpiano ottimo in grado di separare i pattern è quello che massimizza il margine e soddisfa i vincoli di separazione dei pattern. Questo problema di ottimizzazione può essere risolto passando prima a una formulazione Langragiana e successivamente a una formulazione duale. In questo modo vengono introdotti dei moltiplicatori α_i per ogni vincolo di separazione, poi si esprimono in parametri dell'iperpiano w e b in funzione dei moltiplicatori. Infine, attraverso un algoritmo di programmazione quadratica, si derivano i valori ottimi dei moltiplicatori (che saranno 0 per tutti i vettori che non sono support vector)
- **SVM lineare, pattern non separabili:** in questo caso è necessario rilassare i vincoli di separazione. A tal fine si introducono n variabili di *slack* positive ξ e si modificano i vincoli di separazione, per cui per i pattern separabili le variabili ξ saranno uguali a 0, mentre per i pattern non separabili le variabili di *slack* ξ saranno positive e codificheranno la deviazione dal margine. A questo punto, l'iperpiano ottimo deve massimizzare il margine, ma allo stesso tempo minimizzare il numero di elementi erroneamente classificati. Per questo motivo il problema di ottimizzazione viene modificato introducendo un coefficiente C (iperparametro) che indica l'importanza relativa degli errori di classificazione rispetto al margine
- **SVM non lineare:** nel caso in cui le superfici di separazione dei pattern siano complesse, viene definito un mapping non lineare dalla spazio di partenza \mathcal{R}^d verso uno spazio \mathcal{R}^m a più alta dimensionalità. Grazie alla proprietà dei prodotti scalari tra coppie di vettori utilizzata nella formulazione langragiana-duale, è possibile ricondurre il prodotto scalare di due pattern mappati nello spazio \mathcal{R}^m a una funzione Kernel dei due pattern originale nello spazio \mathcal{R}^d . Ciò consente di risolvere il problema di ottimizzazione senza particolari complicazioni rispetto al caso lineare

42. Qual è il principio su cui si basa il classificatore SVM? Cosa si intende per margine?

SVM (Support Vector Machine) è uno degli strumenti più utilizzati per la classificazione di pattern. Invece di stimare le densità di probabilità delle classi, SVM cerca di determinare le superfici decisionali tra le classi. Inizialmente nato come classificatore binario, SVM, tra tutti i possibili iperpiani di separazione, determina quello in grado di separare le classi con il maggior margine possibile. Il margine è appunto la minima distanza

tra l'iperpiano di separazione e un pattern del training set. La massimizzazione del margine è legata alla generalizzazione: se i pattern del training set sono classificati con ampio margine si può sapere che anche pattern del test set vicini al confine tra le classi siano gestiti correttamente.

43. Nel classificatore SVM cosa sono i support vectors? E quali sono i vantaggi dell'approccio SVM?
 L'idea di SVM è quella di determinare le superfici decisionali (iperpiani o superfici complesse) in grado di separare le classi, massimizzando il margine e soddisfando i vincoli di separazione dei pattern. I pattern del training set che giacciono sul margine sono detti support vector. Tali pattern, che costituiscono i casi più complessi, definiscono completamente la soluzione del problema, che può essere espressa come funzione di solo tali pattern, indipendentemente dalla dimensionalità dello spazio e dal numero di elementi del training set. I principali vantaggi dell'approccio SVM sono dati proprio dalla definizione dei support vector. La soluzione del problema di classificazione con SVM si basa su un numero ridotto di support vector, il quale indica la complessità del problema e l'errore medio si dimostra essere limitato dal rapporto fra il numero di support vector e il numero dei pattern totali. Inoltre, SVM scala molto bene rispetto alla dimensionalità dello spazio delle feature e la complessità computazionale è quadratica rispetto al numero di pattern nel training set.

44. Nelle SVM non lineari cosa si intende per kernel? Quali sono i kernel più utilizzati?

In SVM non lineare, quando si effettua un mapping dallo spazio di dimensione d a quello di dimensione k con $k > d$, non è necessario operare direttamente sui pattern, in quanto compaiono solo come scalari.

Questi scalari si possono ricondurre a funzioni kernel in grado di effettuare direttamente tale mapping. Le

più usate sono: i vettori del TS appaiono solo in forma di prodotti scalari tra coppie di vettori (questa proprietà permette di evitare la manipolazione dei vettori nello spazio di dimensione m) \rightarrow per opportuni mapping è possibile ridurre il prodotto scalare di due pattern mappati nello spazio di dimensione m ad una funzione K detta kernel dei due pattern originali nello spazio di dimensione d .

- Polinomio di grado q (iperparametro): le nuove componenti sono ottenute come tutte le possibili combinazioni delle componenti originali del pattern elevate a potenze $\leq q$. Si dimostra quindi che:

$$K(x, x') = [(x \cdot x') + 1]^q$$
 ↓
d; ciò consente di risolvere il problema di ottimizzazione senza particolari complicazioni rispetto al caso lineare.
- Radial Basis Function (RBF) di ampiezza σ (iperparametro): $K(x, x') = e^{(-\frac{\|x-x'\|^2}{2\sigma^2})}$
- Reti neurali a due livelli: $K(x, x') = \tanh(v(x \cdot x') + a)$

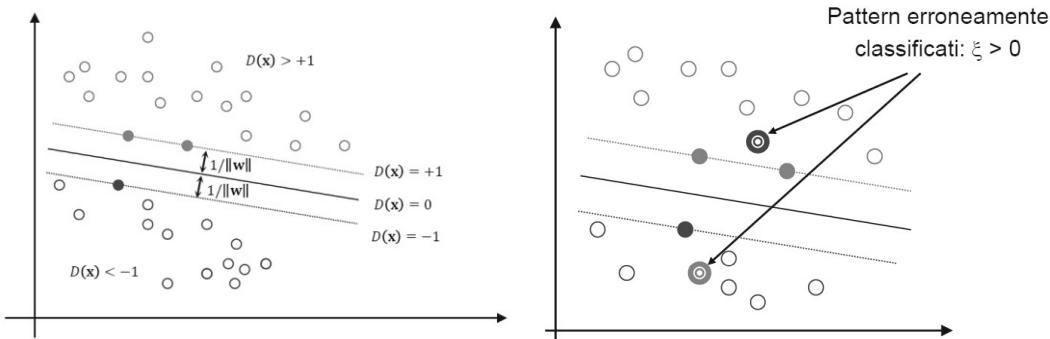
45. Cosa si intende con SVM lineari? Cosa sono le superfici di separazione nel caso $d=2$ e $d=3$?

SVM lineare sfrutta un iperpiano di separazione per separare i pattern delle due classi, quindi nel caso di $d = 2$ e $d = 3$ si sfrutterà un iperpiano in 2 o 3 dimensioni per la separazione. SVM non lineare invece sfrutta superfici molto più complesse per la separazione e quindi effettua un mapping dallo spazio di dimensione d a quello di dimensione k con $k > d$ al fine di avere più libertà.

46. Nell'ambito di classificazione con SVM cosa si intende per pattern linearmente separabili e non linearmente separabili? Fare esempio grafico dei due casi.

Se la superficie di separazione è un iperpiano allora si parla di SVM lineare. Inoltre, se esiste per ipotesi almeno un iperpiano in grado di separare i pattern del training set, allora i pattern si dicono linearmente separabili (grafico a sinistra). Se non esiste un iperpiano in grado di separarli ci saranno inevitabilmente errori di classificazione, allora i pattern si dicono linearmente non separabili (grafico a destra). Se invece la superficie di separazione è complessa si parla di SVM non lineare, senza alcuna ipotesi sulla separabilità dei pattern. Se un problema di classificazione presenta più di 2 classi, SVM necessita di un'estensione multiclass.

NON NECESSARIO PER RISPONDERE ALLA DOMANDA



47. Qual è la differenza fra One-Against-One e One-Against-All?

OAO e OAA rappresentano due soluzioni al problema dovuto alla natura di classificatore binario (fra 2 classi) di SVM, per gestire il problema (ancora aperto) del suo utilizzo con più di due classi

- **One-Against-All:** date s classi, per ogni classe si determina con SVM la superficie di separazione tra i pattern di quella classe da una parte e i pattern di tutte le rimanenti classi dall'altra. Al termine del training si assegna il pattern alla classe per cui è massima la distanza dalla superficie decisionale. Sarà quindi necessario eseguire il training s volte formulistica slide 14 classificazione 1
- **One-Against-One:** consente di risolvere un problema di classificazione multi-classe attraverso classificatori binari. Si scelgono tutti i classificatori a coppia, indipendentemente dall'ordine, durante la classificazione il pattern x viene classificato da ogni classificatore binario. Al termine il pattern x è assegnato alla classe che ha ricevuto più voti. formulistica slide 24 classificazione 1

In genere, OAO è più accurato, ma meno efficiente in quanto richiede l'addestramento di un numero maggiore di classificatori.

48. Qual è la differenza fra fusione a livello di decisione e fusione a livello di confidenza? Fornisci degli esempi.

Nella fusione a livello di decisione ogni classificatore fornisce in output la propria decisione, ovvero la classe a cui ha assegnato il pattern. Le decisioni possono essere combinate in diversi modi:

- **Majority vote rule:** ogni classificatore vota per una classe, il pattern viene assegnato alla classe maggiormente votata o ranking
- **Borda Count:** ogni classificatore produce una classifica delle classi per ogni pattern a seconda della probabilità di appartenere a quella data classe. Ogni posizione in classifica ha un punteggio, si sommano i punteggi, la classe con il punteggio più alto è quella scelta dal classificatore i ranking vengono convertiti in punteggi e poi sommani

Nella fusione a livello di confidenza ogni classificatore fornisce in output la confidenza di classificazione del pattern rispetto a ciascuna delle classi. Sono possibili diversi metodi di fusione:

- **Somma:** si sommano per ciascuna classe tutte le probabilità assegnate dai classificatori e si seleziona poi la classe con somma massima probabilisticamente
- **Prodotto:** analogo alla somma, ma esegue il prodotto tra le probabilità. È il metodo più corretto, ma solo nel caso di indipendenza statistica
- **Massimo o minimo:** il massimo dei minimi, cioè una classe che non ha ricevuto confidenza troppo bassa VEDERE SLIDE 25 CLASSIFICAZIONE 2 PERCHE' C'E' SOLO FORMULA E QUESTA SPIEGAZIONE E' PER MINIMO
- **Somma pesata:** la somma dei vettori di confidenza è eseguita pesando i classificatori in base al loro grado di affidabilità. Il problema è che un solo valore pari a 0 azzererebbe tutto

Nella pratica la somma è spesso preferibile al prodotto in quanto più robusta. Infatti nel prodotto è sufficiente che un solo classificatore indichi confidenza zero per una classe per portare a zero la confidenza del multi classificatore per quella classe.

49. In quali casi scelgo l'approccio SVM lineare e in quali quello non lineare?

5000

SVM lineare conviene se ci si trova in uno spazio con dimensione elevata (per esempio $d=2000$), poiché è molto probabile che i pattern siano facilmente separabili da un semplice iperpiano. Sarà quindi sufficiente definire l'iperparametro C per definire il peso degli errori (è comunque raro siano completamente separabili).

Per medie dimensioni (es. 200 features) si provano entrambe le tipologie, anche questa scelta diventa un iperparametro.

con kernel RFB

Nel caso invece di dimensioni ridotte è spesso necessario SVM non lineare poiché servono superfici complesse per dividere i pattern nello spazio, ed è quindi necessario selezionare la funzione kernel di interesse.

50. Cosa si intende per approccio multi-classificatore? In che modo questo approccio può portare ad un miglioramento delle prestazioni?

Un multi-classificatore è un approccio dove diversi classificatori sono utilizzati (in parallelo, in cascata o in modo gerarchico) per eseguire la classificazione dei pattern. A un certo punto le singole decisioni dei classificatori vengono fuse. È stato dimostrato che l'utilizzo di combinazioni di classificatori porta ad un miglioramento delle prestazioni, in alcuni casi notevole. Tuttavia, la combinazione fra diversi classificatori è efficace solo quando questi sono indipendenti tra loro e quindi non commettono gli stessi errori.

L'indipendenza si ottiene:

- Utilizzando feature non correlate o poco correlate
- Utilizzando algoritmi diversi per l'estrazione delle feature
- Utilizzando diversi algoritmi di classificazione
- Addestrando lo stesso algoritmo su porzioni diverse del training set (bagging)
- Insistendo sui pattern erroneamente classificati (boosting)

51. Qual è la probabilità, in un multiclassificatore, che ogni classificatore esegua la decisione correttamente? NON TROVATO RISCONTRO NELLE SLIDE --> DA RIVEDERE

Dipende dalla tipologia di fusione utilizzata e dal numero di classi del problema. In generale, se si utilizza Majority Vote Rule in una classificazione binaria, la classificazione è corretta se almeno la metà + 1 dei classificatori (nc), ovvero se $k = \frac{nc+1}{2}$. Questo ipotizzando l'indipendenza dei pattern può essere espresso mediante una variabile binomiale: $P_{multi}(nc) = \sum_{m=k}^{nc} \binom{nc}{m} P^m (1-P)^{nc-m}$

52. Descrivere il metodo Random Forest

Il random forest è una tecnica di bagging che si suddivide in tre fasi:

- Estrazione con re-imbussolamento di un sottoinsieme S di pattern dal training set, tipicamente $\frac{2}{3}$ del totale
- Addestramento del classificatore C sul sottoinsieme S
- Fusione dei classificatori

In random forest i singoli classificatori sono classification tree, uno strumento molto utilizzato in applicazioni con pattern categorici o mixed. Un classification tree è un albero binario in cui ogni nodo divide i pattern sulla base di un criterio su una singola feature. Per la crescita dell'albero a partire da un training set, si sceglie, ad ogni livello, la feature che meglio separa le classi e si determina la soglia di suddivisione. Infine, per la classificazione di un nuovo pattern si visita l'albero e, una volta giunti a una foglia, si classifica il pattern sulla base della classe più comune nel nodo (majority vote rule). Per evitare che molti tree scelgano con elevata probabilità le stesse variabili ogni nodo effettua la scelta della feature migliore su un sottoinsieme random, pertanto random forest opera simultaneamente due tipi di bagging: uno sui pattern del training set e uno sulle features.

53. Descrivere il multiclassificatore AdaBoost MANCA DESCRIZIONE E PSEUDOCODICE

AdaBoost è un ^{metodo} algoritmo di boosting dove:

- Più classificatori (weak) vengono combinati per realizzarne uno unico (strong)
- L'apprendimento è incrementale, ad ogni iterazione viene aggiunto un classificatore efficace sui pattern critici

aggiornando il peso precedente

Ad ogni iterazione si assegna un peso a ogni pattern del training set. Pattern classificati erroneamente hanno peso maggiore e contribuiscono maggiormente nella selezione del prossimo classificatore

Regressione

54. Qual è l'obiettivo di una tecnica di regressione?

Nei problemi di classificazione supervisionata i pattern sono tutti etichettati e l'obiettivo del training è apprendere un mapping dallo spazio dei pattern a quello discreto delle etichette.

Nella regressione le etichette sono invece dei valori continui in \mathbb{R} , quindi si ricerca una funzione $f(x) \rightarrow y$. f è controllata da una serie di parametri, x è la variabile indipendente (certa ed esente da errore) e y è quella dipendente (che si assume possa essere affetta da errore). SECONDO ME NON NECESSARIA

55. Qual è la differenza fra simple e multiple linear regression? vedi slide 2-7 classificazione 2

La regressione si definisce lineare se la funzione f da apprendere è controllata da parametri la cui dipendenza è lineare.

- In **simple linear regression** la variabile indipendente è uno scalare. Va quindi determinata l'equazione della retta di regressione, che minimizza la somma dei quadrati dei residui
- In **multiple linear regression** la variabile indipendente è un vettore. Va quindi determinato l'iperpiano di regressione, la cui equazione è formata da matrici e vettori

56. In cosa si differenziano regressione lineare e PCA? Qual è il motivo? In quale caso è preferibile PCA fitting

Il primo minimizza le distanze verticali dall'iperpiano, mentre il secondo quelle euclidee. Questo poiché le variabili della regressione non hanno due ruoli simmetrici, ma una è dipendente e una indipendente, quindi scambiandole si ottengono risultati diversi. Con ruoli simmetrici quindi è preferibile PCA fitting, nel caso per esempio di nuvole di punti geometrici.

57. Quali tecniche si applicano per risolvere problemi di regressione non lineare? vedi slide 13-14 classificazione 2

La regressione non lineare si ha nel caso in cui i parametri siano in relazione non lineare. Per la risoluzione si sfruttano tecniche di ottimizzazione numerica, come **Gradient Descent**. L'algoritmo più noto è quello di **Gauss-Newton**.

58. Quali approcci si usano per problemi di regressione con piccole-medie dimensioni, grandi dimensioni e quali sono le tecniche di classificazione per cui esiste una variante per la regressione? somma dei quadrati dei residui VEDI SLIDE 17 REGRESSIONE

Per problemi di **piccole-medie dimensioni e lineari** si usa **least square**, ovvero una regressione lineare classica che minimizza il quadrato degli scarti, mentre **Gauss Newton** è preferito per quelli **non lineari**.

Per problemi di **grandi dimensioni** risulta generalmente più efficace **Gradient Descent**. Alcune tecniche di classificazione per cui esiste una variante efficace per problemi di regressione sono **Random Forest e SVM**.

SVR e Gradient Boosting

59. Spiegare cosa si intende con variabile dipendente e indipendente nella regressione

Nei problemi di classificazione supervisionata, ad ogni pattern x_i viene associata un'etichetta y_i . Nella regressione i valori y_i sono numerici e continui in \mathbb{R} e l'obiettivo del training è l'apprendimento di una funzione $f(x) \rightarrow y$. In questo caso, x è detta **variabile indipendente** (rappresenta i pattern) e y è detta **variabile dipendente** (rappresenta le etichette); si assume che la variabile indipendente sia esatta, mentre quella dipendente sia affetta da errore.

60. Come si imposta un problema di multiple linear regression? Come sono popolati X, y e β ?

Un problema di multiple linear regression si imposta come segue:

- Si crea la matrice rettangolare X ($n \times (d + 1)$), dove n è il numero pattern e d la dimensione, e la si popola inserendo sulle righe gli n pattern e nell'ultima colonna il valore 1 (ultima colonna a destra con tutti valori 1) il vettore y si ottiene incolonnando tutte le y_i , ovvero tutte le etichette (vettore colonna con uno scalare per ogni pattern) il vettore β si ottiene incolonnando tutte le β_i , ovvero i parametri da determinare, che sono $d + 1$

- La relazione tra le variabili è ora $y = X\beta$ e la funzione obiettivo da minimizzare è $\|y - X\beta\|^2$

61. Nella regressione lineare (sia rispetto ai parametri sia rispetto alla variabile indipendente) i dati con cosa sono approssimati nel caso 2D e 3D? La risposta sarebbe: curve e superfici

La regressione lineare minimizza le distanze verticali dall'iperpiano e non le distanze euclidee. Ciò è la conseguenza del fatto che le variabili indipendente e dipendente non hanno ruolo simmetrico. In questi casi è quindi preferibile utilizzare PCA fitting. risposta non coerente con la domanda? vedi slide 12 classificazione 2

62. Nei problemi di regressione, quali sono le possibili soluzioni nel caso in cui la variabile indipendente non sia lineare? E nel caso in cui la regressione non sia lineare?

Se la variabile indipendente non è lineare, è sempre possibile continuare ad usare i metodi di regressione lineare approssimando i dati con curve e superfici, in quanto basta che la funzione di regressione sia lineare rispetto ai parametri.

Se invece c'è una dipendenza non lineare del modello dai parametri, non si può applicare il metodo ai minimi quadrati; ~~se la dipendenza è nota in questo caso per la risoluzione si applicano tecniche iterative di ottimizzazione numerica, come l'algoritmo Gradient Descent e il più noto algoritmo di Gauss-Newton.~~

se la dipendenza non è nota si possono provare e validare sia modelli lineari sia modelli non lineari via via più complessi

Clustering

63. Descrivere la variante Fuzzy del K-means

La variante Fuzzy del K-means consente a un pattern di appartenere con un certo grado di probabilità a classi diverse.

- Si selezionano ^{inizialmente} S pattern in modo casuale come centroidi
- Si calcola il grado di appartenenza di ciascun pattern ai vari cluster sulla base della distanza
- Si ricalcolano i centroidi come media pesata dell'appartenenza di tutti i pattern
- Si ripetono iterativamente le ultime due fasi fino a quando si raggiunge il numero massimo di iterazioni o si ha la soluzione ottimale

tra 0 e 1

64. Quali sono le differenze fra il metodo K-means e la sua versione Fuzzy?

La variante Fuzzy del K-means consente a un pattern di appartenere con un certo grado di probabilità a classi diverse. I centroidi in questo caso vengono calcolati come media pesata rispetto alle probabilità di appartenenza. Rispetto al K-means, questa variante a volte fornisce una convergenza più robusta verso la soluzione finale. Dall'altra parte però, poiché l'appartenenza ad un cluster dipende implicitamente dal numero di cluster, se questo viene inizialmente scelto in modo non corretto si possono ottenere cattive soluzioni.

65. Cosa è il clustering e su cosa si basano le metodologie di clustering?

Per clustering si intende una famiglia di metodi non supervisionati in grado di individuare raggruppamenti intrinseci (cluster) di pattern e definire in corrispondenza di tali gruppi le classi (incognite). Il problema è molto complesso (NP hard), infatti determinare la soluzione ottima è possibile solo con pochi pattern. Possono essere utilizzati per esempio in biologia per derivare animali e piante, nel marketing per definire dei gruppi di utilizzatori ecc. per risposta completa ampliare con risposta domanda seguente

66. Quali sono i più noti algoritmi di clustering?

Gli algoritmi di clustering, dato un criterio, forniscono una procedura algoritmica per determinare le soluzioni che lo ottimizzano. Le principali famiglie di algoritmi sono:

- **Clustering gerarchico:** attraverso operazioni, tipicamente bottom-up, aggregano pattern in base a una misura di distanza. Si organizzano i dati in una struttura ad albero. I più noti sono:
 - **Single link:** distanza minima tra due pattern dei cluster

- **Average link**: distanza media tra i pattern dei due cluster
 - **Complete link**: distanza massima tra due pattern dei cluster
- **Clustering basato su centroidi**: attraverso tentativi euristici si individuano i cluster cercando di minimizzare la distanza dei pattern dai centroidi dei cluster a cui appartengono. I più noti sono:
 - **K-means**
 - **Fuzzy K-means**
 - **Expectation-Maximization (Gaussian Mixture)**
- **Clustering basato sulla densità**: i cluster individuati sono regioni connesse in aree ad elevata densità. L'algoritmo più noto è DBSCAN

67. Qual è l'idea di base dell'algoritmo di clustering EM con Gaussian mixture?

Si ipotizza che i pattern siano stati generati da un mix di distribuzioni: ogni classe ha generato dati in accordo con una specifica distribuzione, ma al termine della generazione i pattern appaiono come prodotti da un'unica distribuzione multimodale. L'obiettivo del clustering EM è risalire ai parametri delle singole distribuzioni che li hanno generati. A tal fine si ipotizza nota la forma delle distribuzioni e si assume, per semplicità, che esse siano tutte dello stesso tipo. La stima dei parametri avviene secondo il criterio generale del maximum likelihood. In generale, il likelihood L dei parametri O dati i pattern X corrisponde alla probabilità di aver ottenuto i pattern dati i parametri: $L(O|X) = p(X|O)$. Per semplicità, al posto del likelihood, si massimizza il suo logaritmo. Purtroppo, però questa massimizzazione è molto complicata a causa della sommatoria dentro al logaritmo. A tal fine si usa EM, che è un approccio iterativo usato nel caso in cui i dati a disposizione siano incompleti, esso si divide in:

- **Expectation**: viene calcolato il valore atteso del log-likelihood completo, dato il training set e una stima dei parametri
- **Maximization**: determina il valore dei parametri che massimizzano il valore atteso calcolato al passo precedente
 - 1. i pattern all'interno dello stesso cluster devono essere tra loro più simili rispetto a pattern appartenenti a cluster diversi
 - 2. i cluster sono costituiti da nuvole di punti a densità relativamente elevata separate da zone dove la densità è più bassa

68. Che cosa sono i criteri di clustering? Fare un esempio.

I criteri di clustering descrivono cosa si vuole ottenere e l'ottimalità di ogni soluzione ammissibile. I criteri di clustering si basano su due osservazioni: i cluster sono zone con una densità elevata di pattern e sono separati da zone con una densità inferiore e i pattern di uno stesso cluster sono tra loro più simili rispetto a quelli di altri cluster. Esistono due diversi criteri possibili:

- **Minimizzazione distanze dai centroidi**: minimizza la somma dei quadrati delle distanze dei pattern x dai centroidi delle classi. È un buon criterio per cluster a simmetria radiale, ma penalizza forme allungate o cluster innestati
- **Minimizzazione distanza intra-classe**: non penalizza cluster allungati, infatti affinché la funzione che misura la distanza tra x e il cluster di appartenenza assuma un valore ridotto, non è necessario che tutti i pattern siano vicini a x , ma è sufficiente che ve ne sia solo uno vicino (o gran parte)

69. Cosa si intende per Clustering esclusivo e Clustering soft (o Fuzzy). Quest'ultimo che vantaggi può avere?

Negli algoritmi di clustering può essere fatta una distinzione fra:

- Clustering hard (esclusivo): un pattern è assegnato, in modo esclusivo, a un solo cluster
- Clustering soft (fuzzy): i pattern appartengono ai diversi cluster con un certo grado di appartenenza (ad esempio, compreso fra 0 e 1). Questo approccio è più efficace nel gestire pattern vicino al bordo di due o più cluster.

70. Rispetto a K-means l'approccio di clustering EM con Gaussian mixture quali maggiori flessibilità consente?

K-means dipende strettamente dalla soluzione e dal numero di classi in input e non è efficiente per i cluster di forma non sferica. EM è simile, ma non dipende direttamente dalla soluzione in input e permette di riconoscere cluster di forma ellissoidale. NON C'E' MOLTO SULLE SLIDE, CREDO SIA CORRETTA MA NON HO LA CERTEZZA

71. Descrivere a grandi linee l'algoritmo di Clustering K-means.

K-means è un metodo computazionalmente molto semplice e altrettanto semplice da implementare, per questo motivo è spesso la prima scelta per risolvere problemi di clustering. K-means ha le seguenti caratteristiche:

- Minimizza implicitamente le distanze dai centroidi
- Richiede in input il numero di cluster e una soluzione iniziale, produce buoni risultati a patto che quest'ultima sia ragionevole e accompagnata da un numero adeguato di classi
- Il tipo di ottimizzazione è iterativa e locale, pertanto il metodo può convergere a massimi locali della soluzione. La convergenza si ottiene solitamente con meno di 10 iterazioni
- Identifica cluster iper-sferici nel caso in cui venga utilizzata la distanza euclidea come misura di distanza tra i pattern o cluster iper-ellissoidali nel caso di distanza di Mahalonobis
- I cluster sono modificati iterativamente a seguito del ricalcolo del loro centroide. L'algoritmo termina quando i centroidi sono stabili e quindi le partizioni non cambiano

72. Descrivere le principali criticità e limitazioni dell'algoritmo di Clustering K-means.

K-means assegna ogni pattern al cluster la cui distanza dal centro è minore. Questo è problematico in quanto non permette di riconoscere correttamente cluster non sferici. Inoltre, la qualità della soluzione finale dipende dalla precisione di quella fornita in input, quindi spesso si ripete l'algoritmo più volte fornendo in input un numero di classi S e una soluzione iniziale differente e si verificano le prestazioni, assegnando una penalità in base al numero di classi. Se l'input non fosse preciso, anche l'algoritmo stesso potrebbe non esserlo. Il vantaggio risiede però nella sua semplicità. RIASSUNTO FATTO DA LUI, NON HO TROVATO GRANDE RISCONTRO NELLE SLIDE --> ESAME DICE DI CONTROLLARE SLIDE 8-11

73. Come può essere scelto nella pratica il numero di cluster in un algoritmo di clustering come K-means? DA RIVEDERE --> SLIDE 11 CLUSTERING

K-means necessita di una soluzione in input e di un numero di classi S dai quali dipende la precisione dell'output finale. Vista la semplicità e rapida convergenza, un metodo molto usato consiste nel ripetere più volte l'algoritmo con input differenti e confrontare le prestazioni. In questo caso però, maggiore è il numero di classi S , più è probabile che ciascun pattern si trovi vicino al proprio centroide (avendone un numero maggiore), anche se non necessariamente la soluzione è più precisa. Quindi un metodo molto usato consiste nell'assegnare alle prestazioni una penalità che dipende dal numero di classi: $J'_e = J_e + \text{penalty} \cdot s$

Riduzione dimensionalità

74. Qual è l'obiettivo delle tecniche di riduzione di dimensionalità?

L'obiettivo dei metodi per la riduzione di dimensionalità è eseguire un **mapping** dallo spazio iniziale R^d a uno spazio di dimensione inferiore R^k ($k < d$), scartando le informazioni ridondanti e dati meno rilevanti per il problema di interesse. In questo modo si semplifica l'addestramento e si possono ottenere miglioramenti di prestazioni. Riduzione dimensionalità non significa però mantenere alcune dimensioni e cancellarne altre, ma combinarle in modo opportuno.

Le più note tecniche sono:

- **Principal Component Analysis (PCA)**: trasformazione *non supervisionata* che esegue un mapping lineare con l'obiettivo di preservare le dimensioni che *rappresentano* al meglio i pattern,

sottolineatura --> non trovato riscontro nelle slide

mantenendo quindi il più possibile le informazioni originali. Un esempio di applicazione di PCA è la codifica di un'immagine

- **Linear Discriminant Analysis (LDA):** trasformazione *supervisionata* che esegue un mapping lineare con l'obiettivo di preservare le dimensioni che *discriminano* al meglio i pattern. Per farlo deve conoscere le classi dei vari pattern, da cui la necessità di una supervisione
- **t-distributed Stochastic Neighbor Embedding (t-SNE):** trasformazione *non lineare* e *non supervisionata*, ideata per ridurre la dimensionalità a 2 o 3 dimensioni in modo da permettere la visualizzazione di dati multidimensionali. La visualizzazione risulta più efficiente di PCA, specialmente nel caso di pattern sparsi. Un esempio è la visualizzazione in 2D di MNIST. (digit scritti a mano)

75. Descrivere la tecnica di PCA-Whitening

È una tecnica di pre-normalizzazione dei dati. Rimuove le correlazioni tra le dimensioni, allineando gli assi di variazione principale dei dati (autovettori) agli assi cartesiani (in pratica ruota l'ellisse). Successivamente, sfericizza l'ellisse uniformando le varianze (autovalori).

Dopo aver proiettato i pattern sullo spazio del PCA e diviso ogni dimensione per la radice quadrata dell'autovalore corrispondente, la matrice di covarianza sarà l'identità, semplificando notevolmente le operazioni.

76. Cosa sono le matrici di scattering e cosa indica within-class e between-class?

Le matrici di scattering sono due matrici calcolate in LDA per formulare il criterio di ottimizzazione di massima separazione tra le classi dei pattern.

- La matrice **within-class** S_w indica come i vettori sono sparsi rispetto al centro delle classi
- La matrice **between-class** S_b indica come i centri delle classi sono sparsi rispetto al centro generale della distribuzione

Massimizzando S_b e minimizzando S_w si estraggono le feature che avvicinano pattern della stessa classe e allontanano quelli di classi diverse. frase formulata male, non si capisce --> vedi slide 12 riduzione dimensionalità

Reti neurali

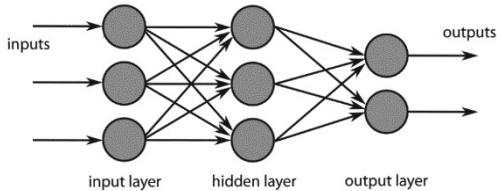
77. Che cos'è il learning rate nell'ambito dell'apprendimento di reti neurali? Cosa succede se viene scelto un learning rate troppo piccolo o troppo grande?

Il learning rate è un parametro delle reti neurali che influisce sull'aggiornamento dei pesi, nello specifico indica la dimensione dello step che viene fatto ad ogni epoca (processamento di tutti i pattern). Se il learning rate è troppo elevato, può portare a oscillazioni e/o divergenza, mentre se fosse troppo piccolo la convergenza risulterebbe troppo lenta.

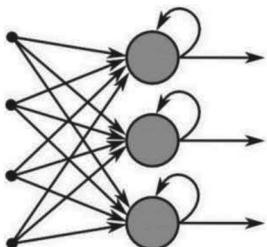
78. Indicare le differenze tra reti neurali feedforward e le reti neurali ricorrenti, disegnando un esempio di entrambe.

Le reti neurali sono composte da gruppi di neuroni artificiali organizzati in livelli. Tipicamente sono presenti un livello di input, uno di output e uno o più livello intermedi o nascosti. Ogni livello contiene uno o più neuroni.

- **Feedforward:** le connessioni collegano i neuroni di un livello con i neuroni del livello successivo. Non sono consentite connessioni all'indietro o verso lo stesso livello



- **Recurrent:** possono essere presenti anche collegamenti tra neuroni dello stesso livello o con neuroni di livelli precedenti, utili per simulare una memoria a breve termine potendo mantenere memorizzate le informazioni dei livelli precedenti, sebbene complichino la struttura



79. Dare una definizione di neurone, sinapsi, assone, dendriti, connectome.

- **Neurone:** detto anche soma, è la cellula fondamentale del sistema nervoso
- **Sinapsi:** collegamenti tra neuroni
- **Assone:** fibra principali che si dirama dal neurone e porta informazioni ad altri neuroni
- **Dendriti:** fibre minori che si diramano a partire dal neurone e raccolgono gli input
- **Connectome:** mappa di tutte le connessioni dei neuroni, la sua formazione è affidata al DNA e all'apprendimento

80. Enunciare la Regola di Hebb

Se due neuroni, tra loro connessi da una o più sinapsi, sono ripetutamente attivati simultaneamente, allora le sinapsi che li connettono sono rinforzate

81. Quanto è il numero totale di pesi in MLP? utile rivedersi slide 12 reti neurali 1

Data una rete a 3 livelli ($d: n_H: s$) dove in input ci sono d neuroni, livello hidden n_H neuroni e output s neuroni. Il numero di pesi totale, o parametri è: $d \cdot n_H + n_H \cdot s + n_H + s$ Ovvero uno tra ogni neurone di input e hidden, uno tra ogni neurone hidden e out e uno per ogni neurone out e hidden dovuto al bias.

82. In cosa consiste l'addestramento di una rete neurale, sia nel caso della classificazione sia della regressione?

nel determinare il valore dei pesi che determinano il mapping desiderato tra input e output
L'addestramento di una rete neurale consiste nell'apprendere i pesi ottimali dei collegamenti che permettano di ottenere l'output desiderato dipendentemente dall'input fornito (mapping desiderato). Se siamo interessati ad addestrare una rete neurale che operi come classificatore, l'output desiderato è l'etichetta corretta della classe del pattern in input. Mentre se la rete neurale deve risolvere un problema di regressione, l'output desiderato è il valore corretto della variabile dipendente, in corrispondenza del valore della variabile indipendentemente fornita in input.

83. Per l'addestramento di una rete neurale che cosa si intende con vettore di output desiderato?

Come può essere definito? Come si può calcolare l'errore da retro-propagare a partire dal vettore desiderato e dal valore calcolato dalla rete per un pattern?

Il vettore di output desiderato è il vettore risultante che si attende dato un determinato input. Varia in base al problema per il quale la rete neurale viene utilizzata, nel caso di classificazione per esempio è un vettore avente tutti valori -1 e un solo valore ad 1 in corrispondenza dell'indice della classe assegnata. La differenza tra l'output prodotto della rete e quello desiderato è l'errore della rete. A partire dal vettore desiderato t e quello effettivamente calcolato dalla rete u , l'errore si calcola nel seguente modo: $(t - u) \cdot f'(net)$, ovvero la differenza tra il vettore desiderato e quello ottenuto moltiplicata per la derivata della funzione di attivazione usata per il calcolo dell'output.

$$\downarrow \delta = (t - u) * f'(net)$$

84. Qual è la differenza sostanziale dell'approccio "On-line" rispetto a "SGD con mini-batch" per il training di reti neurali?

Nell'approccio online i pattern vengono presentati sequenzialmente e i pesi vengono aggiornati dopo la computazione di ciascun pattern. Nell'approccio SGD i pattern vengono ordinati casualmente e suddivisi in gruppi, detti mini-batch, di egual dimensione. I pesi vengono solamente aggiornati dopo che tutti i pattern di un batch sono stati processati (iterazione), mentre l'epoca rappresenta il processamento di tutti i batch, ovvero di tutti i pattern del TS. L'approccio online corrisponde quindi ad un approccio mini-batch con dimensione dei gruppi pari a 1. sottolineatura no riscontro nelle slide

85. Come è gestito e come avviene l'addestramento supervisionato (training) di un classificatore in una rete neurale?

Una volta fissati numero di livelli e neuroni, l'addestramento di una rete neurale consiste nel determinare il valore dei pesi w che determinano il mapping desiderato tra input e output. Se la rete neurale deve risolvere un problema di classificazione, allora l'output desiderato sarà l'etichetta corretta della classe del pattern in input. Se la rete neurale deve risolvere un problema di regressione allora l'output desiderato è il valore della variabile dipendente in corrispondenza del valore della variabile indipendente data in input. Per gestire un problema di classificazione con s classi e pattern d -dimensionali, si è soliti utilizzare una rete ($d: n_H: s$) a tre livelli. Ovvero tanti neuroni di input quante sono le feature e tanti neuroni di output quante sono le classi. n_H è invece un iperparametro: un valore ragionevole è $\frac{n}{10}$ (ovvero 1/10 del numero di pattern)

- 1 • Pre-normalizzazione dei pattern di input (attraverso min-max scaling, standardization o whitening) favorisce la convergenza durante l'addestramento.
- 5 • Inizializzazione dei pesi con valori random:
 - Livello input nel range $\pm \frac{1}{\sqrt{d}}$
 - Livello intermedio nel range $\pm \frac{1}{\sqrt{n_H}}$
- 2 • Addestramento supervisionato vengono presentati alla rete pattern di cui è nota la classe e propagando gli input attraverso forward propagation
- 3 • Ottenimento del vettore di output desiderato
- 4 • Calcolo dell'errore della rete (differenza tra l'output prodotto e quello desiderato)
 - Modifica dei pesi della rete in modo da minimizzare l'errore medio sui pattern del training set
 - Applicazione dell'algoritmo di error backpropagation per l'aggiornamento dei pesi della rete
 - Valutazione della convergenza

86. Descrivere il comportamento dell'algoritmo di backpropagation

Viene per prima cosa calcolato l'errore della rete, $(t - u) \cdot f'(net)$, ovvero la differenza tra il vettore desiderato e quello ottenuto moltiplicata per la derivata della funzione di attivazione usata per il calcolo

riassunti
da lui

dell'output. Dal nodo di output poi l'errore si propaga indietro fino a raggiungere i neuroni in input. L'errore di un nodo è calcolato come segue:

1. La sommatoria degli errori di tutti i neuroni al quale esso è collegato (quindi dei livelli successivi) ciascuno moltiplicato per il peso del collegamento tra i due neuroni
2. La somma totale viene poi moltiplicata per la derivata della funzione di attivazione del neurone
3. Calcolato l'errore su tutti i nodi, è poi possibile aggiornare i pesi a partire dai valori attuali nel seguente modo: $w_{ij} = w_{ij} + n \cdot \text{err}(j) \cdot x_i$ dove n è il learning rate, x_i valore del neurone dal quale parte il collegamento e $\text{err}(j)$ errore del neurone al quale arriva il collegamento

87. Che cosa si intende con epoca e cosa con iterazione nell'ambito di SGD?

In SDG per epoca si intende la presentazione, una sola volta, alla rete di tutti i pattern del training set. Mentre l'iterazione è la presentazione, una sola volta, dei pattern costituenti un mini-batch e il conseguente aggiornamento dei pesi.

88. Come si calcola l'attivazione (net) di un neurone artificiale? Indicare formula e commentarla.

La net di un neurone artificiale si calcola come sommatoria di tutti gli input che riceve dai neuroni precedenti, ciascuno moltiplicato per il peso del collegamento tra i due neuroni, e infine viene sommato il peso del bias, il cui input può essere ommesso essendo una moltiplicazione per 1. $\text{net}(i) = \sum w_{j,i} \cdot in_j + w_{0,i}$ $w_{j,i}$ è il peso del collegamento tra il neurone di interesse i e il neurone j precedente, mentre in_j è l'input che il neurone precedente j comunica. $w_{0,i}$ è il peso del bias. VEDI SLIDE 7 RETI NEURALI PER FORMULISTICA MIGLIORE

89. Qual è $f(\text{net})$ di Softmax e quali vantaggi ha questa funzione di attivazione rispetto a sigmoid e tanh?

Il livello di attivazione net_k dei singoli neuroni dell'ultimo livello si calcola in modo consueto, il valore net si calcola in maniera analoga, mentre la funzione di attivazione è: $f(\text{net}_k) = \frac{e^{\text{net}_k}}{\sum_{c=1..s} e^{\text{net}_k}}$. Questa ha il vantaggio di fornire un output compreso tra 0 e 1, la cui somma di tutti i valori di output è 1, quindi un valore probabilistico. A differenza di tanh che ha output tra -1 e 1 e sigmoid che ha output tra 0 e 1, ma senza somma ad 1.

90. Cosa si intende per funzione di attivazione? Quali sono le caratteristiche richieste per le reti neurali artificiali?

Nei neuroni biologici $f(\cdot)$ è una funzione tutto-niente temporizzata: quando net_i supera una certa soglia, il neurone emette un impulso, per poi tornare a riposo. Analogamente nei neuroni artificiali la funzione di attivazione determina l'^{comportamento del neurone}output in base al livello di eccitazione. Le reti neurali artificiali più comuni operano con livelli ^{invece che con gli impulsi}continui e $f(\cdot)$ è una funzione non lineare (necessario per eseguire mapping complesso delle informazioni in input), ma continua e differenziabile (necessario per la retro-propagazione dell'errore). La funzione di attivazione più utilizzata è la sigmoidea nelle varianti standard logistic function e tangente iperbolica.

91. Quali sono le caratteristiche di un MLP? Cosa afferma l'universal approximation theorem?

Un Multilayer Perceptron (MLP) è una rete feedforward formata da almeno 3 livelli, di cui almeno uno hidden, con funzioni di attivazione non lineari. Il teorema noto come universal approximation theorem afferma che ogni funzione continua che mappa intervalli di numeri reali su un intervallo di numeri reali, può essere approssimata da un MLP con un solo hidden layer (livello intermedio). Questo teorema, insieme alla difficoltà di addestramento di reti più profonde, è una delle motivazioni per cui per molti anni ci si è soffermati su reti a 3 livelli. Tuttavia, l'esistenza di una soluzione non implica efficienza, e per questi motivi sono state introdotte le reti deep.

92. Cosa si intende per error backpropagation in riferimento a MLP?

Error Backpropagation è un algoritmo che si basa sulla regola di derivazione a catena. Nel caso di MLP, dopo aver calcolato l'errore per il pattern x , che quantifica quanto l'output prodotto si discosta da quello

desiderato, si cerca di minimizzare la funzione dei pesi $J(w)$ modificando i pesi w in direzione opposta al gradiente di J . Infatti, il gradiente indica la direzione di maggior crescita di una funzione e, muovendoci in direzione opposta, riduciamo al massimo l'errore. Si parte prima con la modifica dei pesi del livello hidden-output, poi si procede all'indietro modificando i pesi del livello input-hidden.

93. Cos'è l'algoritmo Stochastic Gradient Descent?

È l'approccio più utilizzato per implementare error backpropagation. A ogni epoca gli n pattern del training set vengono ordinati in modo casuale e poi suddivisi in gruppi di uguale dimensione denominati mini-batch. I valori del gradiente sono accumulati in variabili temporanee (una per ogni peso), quindi l'aggiornamento dei pesi avviene solo quando tutti i pattern di un gruppo sono stati processati. Questo approccio viene preferito a quello classico (on-line) poiché quest'ultimo richiede l'aggiornamento dei pesi dopo la presentazione di ogni pattern, causando problemi di efficienza e robustezza.

94. Quali sono le più comuni funzioni di attivazione utilizzate per neuroni artificiali? Perché è necessario che siano non-lineari e differenziabili (esistenza derivata)?

Le più comuni funzioni di attivazione sono sigmoid e tanh. La prima produce un output compreso tra 0 e 1, la seconda tra -1 e 1, quindi spesso preferita per via della simmetria rispetto lo 0 che rende la convergenza più rapida. Nelle reti deep invece spesso si usa relu, più ottimale nella retro-propagazione del gradiente necessario per la backpropagation dell'errore. Infatti, la derivata di sigmoid è < 1 quasi sempre, e andando l'applicazione della regola di derivazione a catena porta a moltiplicare molti termini minori di 1 con la conseguenza di ridurre parecchio i valori del gradiente nei livelli lontani dall'output. La funzione deve essere non lineare perché se vogliamo che una rete sia in grado di eseguire un mapping complesso dell'informazione di input sono necessarie non linearità. Inoltre, deve essere continua e differenziabile per poter effettuare la retro-propagazione dell'errore.

95. Cosa è la Cross Entropy?

Per un problema di classificazione multi-classe si consiglia l'utilizzo di cross-entropy, altrimenti detta multinomial logistic loss, come loss function. La cross-entropy tra due distribuzioni discrete p e q , che fissata p misura quanto q differisce da p , è definita da: $H(p, q) = -\sum_v p(v) \cdot \log(q(v))$. Nel suo impiego come loss function p è il vettore target, mentre q il vettore di output. Il valore minimo di h (sempre > 0) si ha quando il vettore target coincide con l'output

96. Cosa è il Momentum?

Se si sfrutta SDG con mini-batch, il gradiente può discendere a zig-zag, rallentando la convergenza e rendendola instabile. Un metodo efficace per evitarlo consiste nel calcolare il nuovo aggiornamento come combinazione lineare del precedente, conferendo più stabilità. e del gradiente attuale, che corregge la direzione

97. Che cosa è il learning rate adattivo?

eta (η per il cic --> vedi slide 36 reti neurali)

Il learning rate adattivo è l'adattamento del learning rate globale a ogni specifico peso. Alcuni algoritmi che usano questa tecnica sono: NAG, Adagrad, Adadelta, RMSProp, Adam e Momentum.

98. Quali sono le cause principali che possono portare una rete a non convergere?

Alcune delle cause che possono portare una rete a non convergere sono:

- Input non normalizzati
 - Output non normalizzato in [0,1] (regression)
 - Pesi non inizializzati in range ^{random} ragionevoli
 - Etichette in formati errati
 - Loss function non adeguata corretta e non adeguata al tipo di label fornite
 - Funzione di attivazione all'ultimo livello errata
 - Learning rate non adeguato
- pattern delle diverse classi non mescolati random all'interno dei minibatch

Deep Learning

99. Cosa si intende con Deep Neural Network? Perché le reti neurali deep sono più efficaci delle MLP a tre livelli? risposta ok ma molto reinventata --> vedi slide 2, 3, 4, 6, 7 deep learning 1

Con il termine Deep Neural Network si denotano reti neurali profonde composte da molti livelli (almeno 2 hidden) organizzati gerarchicamente. La complessità di una DNN è data dal numero di livelli, neuroni, connessioni e pesi. Le DNN si rivelano più efficaci delle MLP a tre livelli per diversi motivi.

- Sono particolarmente migliori quando sono disponibili grandi quantità di dati per il training (big data)
- Il training di modelli complessi richiede un'elevata potenza di calcolo e i suoi tempi si sono notevolmente ridotti grazie alla disponibilità di nuove GPU dotate di migliaia di core e quantità elevate di memoria dedicata
- L'introduzione della funzione di attivazione Relu ha risolto il problema del vanishing gradient che si presentava su reti profonde utilizzando la funzione sigmoide
- Alcuni tipi di reti deep, come le Convolutional Neural Network (CNN) sono progettate con neuroni connessi solo localmente e con pesi condivisi e hanno quindi una maggiore efficienza con forti riduzioni del numero di pesi e connessioni

100. Da che cosa è caratterizzata la complessità di una DNN?

La complessità dipende dal numero di livelli, di neuroni, di connessioni e di pesi.

- All'aumentare dei livelli aumentano le prestazioni a discapito dell'efficienza
- All'aumentare dei pesi aumenta la complessità del training
- All'aumentare di neuroni e connessioni aumenta la complessità di forward e backpropagation

101. Quali sono le principali tipologie di DNN?

Modelli feedforward discriminativi, per classificazione o regressione, con training generalmente supervisionato:

- **CNN:** Convolutional Neural Network o ConvNet
- **FC DNN:** Fully Connected DNN (MLP con almeno due livelli hidden)
- **HTM:** Hierarchical Temporal Memory

Modelli generativi addestrati alla ricostruzione di input, con training non supervisionato:

- **Stacked Auto-Encoders**
- **RBM:** Restricted Boltzmann Machine
- **DBN:** Deep Belief Networks

Modelli ricorrenti, usati ad esempio per speech recognition:

- **RNN:** Recurrent Neural Network
- **LSTM:** Long Short-Term Memory

Reinforcement learning per l'apprendimento di comportamenti:

- **Deep Q-Learning**

102. Descrivere l'architettura di una CNN

Le CNN sono progettate per processare immagini in quanto grazie all'utilizzo di elaborazione locale, pesi condivisi e pooling sono più efficienti in questo ambito rispetto ai modelli fully-connected e processano porzioni diverse dell'immagine allo stesso modo.

Sono composte da una gerarchia a livelli:

- Il livello di input è collegato direttamente ai pixel dell'immagine
- I livelli intermedi alternano convoluzione e pooling utilizzando connessioni locali e condivise
- I livelli finali sono strutturati come una MLP fully-connected (solitamente livello finale SoftMax ha tanti neuroni quante sono le classi e associa ad ogni output una probabilità di appartenenza)

103. Descrivere l'operazione di convoluzione e quello di aggregazione molto riassunta --> vedi slide 9-14 deep 1

La **convoluzione** è un'operazione di image processing attraverso la quale si applicano filtri digitali. Il filtro viene fatto scorrere sulle diverse posizioni dell'input dell'immagine; per ognuna viene generato un valore di output, eseguendo il prodotto scalare tra maschera e porzione dell'input. Nelle reti neurali, i neuroni corrispondono ai pixel, le immagini di input e output corrispondono ai livelli di una rete e i pesi del filtro corrispondono ai pesi delle connessioni. Di fatto un classico neurone di una MLP esegue una convoluzione. Nel caso di una CNN, il filtro non opera su una porzione bi-dimensionale ma su una porzione del volume di input (3D). vedi slide 16 deep learning 1

Per **aggregazione** (pooling) si intende una trasformazione che a partire dalle informazioni del volume di input genera una feature map di dimensione inferiore, mantenendo però le informazioni significative. L'aggregazione opera generalmente nell'ambito di ciascuna feature map, per cui il numero di feature map in input e output non cambia, e consente quindi di passare per esempio da un volume in input 224x224x64 a un volume in output 112x112x64. Gli operatori di aggregazione più utilizzati sono media (Avg) e massimo (Max).

104. Come si determina il numero di connessioni e il numero di pesi in una CNN?

Il numero di connessioni corrisponde al numero dei neuroni del volume di output moltiplicato per la dimensione del filtro, mentre il numero di pesi, essendo condiviso a livello di feature map, è la dimensione del filtro +1, moltiplicata per la feature map di output. Il +1 è dovuto al bias, in quanto semplicemente comporterebbe un peso per ogni feature map. ?

105. Che cosa si intende con transfer learning? Quali sono le tecniche di transfer learning utilizzabili?

Il training di CNN complesse su dataset di grandi dimensioni può richiedere molto tempo. In alternativa al training da zero si preferisce eseguire il cosiddetto transfer learning. Due esempi di transfer learning sono:

- Fine-Tuning: si parte con una rete pre-trained su un problema simile e si eseguono le seguenti operazioni:
 1. Si rimpiazza il livello di output con un nuovo livello di output softmax
 2. Come valori iniziali dei pesi si usano quelli della rete pre-trained, tranne che per le connessioni tra penultimo e ultimo livello i cui pesi sono inizializzati random
 3. Si eseguono nuove iterazioni di addestramento (SGD) per ottimizzare i pesi rispetto alle peculiarità del nuovo dataset
- Riutilizzo features: si usa una rete esistente, sempre pre-trained, senza ulteriore fine-tuning. Si estraggono le feature generate dalla rete durante il processo forward e si usano per addestrare un classificatore esterno a classificare i pattern del nuovo dominio applicativo

106. Quali sono i 4 principali tipi di modelli di sequenze di input/output per il deep learning?

Sono delle strutture nelle quali input e output possono non avere dimensioni prefissate, ma essere sequenze a lunghezza variabile.

- **One to one:** input immagine, output vettore di probabilità delle classi
- **One to many:** input immagine, output una frase **descrittiva**, descritta in linguaggio naturale
- **Many to one:** input una frase, output un valore numerico che rappresenta il giudizio
- **Many to many:** input frase, output la sua traduzione (es. da inglese ad italiano)

107. Che cosa si intende con unfolding in time? forse da ampliare da slide 4 deep learning 2

Per poter addestrare (con backpropagation) una RNN è necessario eseguire il cosiddetto unfolding o unrolling in time, stabilendo a priori il numero di passi temporali su cui effettuare l'analisi. Di fatto una RNN unfolded su 20 step equivale a una DNN feedforward con 20 livelli. Pertanto, addestrare RNN che appaiono relativamente semplici può essere molto costoso e critico per la convergenza.

108. Che cos'è una cella e cosa rappresenta h in una RNN?

Una cella è una parte di rete ricorrente che preserva uno stato interno, denominato h_t , per ogni istante temporale. È costituita da un numero prefissato di neuroni. h_t dipende dall'input x_t e dallo stato precedente h_{t-1} . Infatti, $h_t = f(h_{(t-1)}, x_t)$. Le celle di base vengono utilizzate unicamente per la memoria a breve termine.

109. Descrivere le celle LSTM e GRU? IMPARARE IMMAGINI SLIDE 6-7 DEEP LEARNING 2

Sono due celle pensate per simulare memoria più a lungo termine e sopperire alle limitazioni delle celle di base.

- **LSTM** (Long Short-Term Memory): lo stato h_t è suddiviso in due vettori h_t e c_t :
 - h_t è uno stato a breve termine, anche in questo caso uguale all'output della cella y_t
 - c_t è uno stato a lungo termine
 - La cella apprende durante il training cosa è importante dimenticare (forget gate) dello stato passato $c_{(t-1)}$ e cosa estrarre e aggiungere (input gate) dall'input corrente x_t
 - Per il calcolo dell'output $y_t = h_t$ si combina l'input corrente (output gate) con informazioni estratte dalla memoria a lungo termine
- **GRU** (Gated Recurrent Unit): si tratta di una versione semplificata di LSTM, la quale cerca di mantenerne i vantaggi riducendo la complessità. Infatti, ha un solo stato di memoria h_t e un solo gate (con logica invertita) per quantificare quanto dimenticare e quanto aggiungere, in quest'ultimo caso sarà necessario prima dimenticare qualcosa

110. Quali sono le caratteristiche delle Deep RNN?

Sono reti contenenti più celle stacked o unfolded. Gli output di un livello sono gli input del livello successivo. Con più livelli possono essere apprese relazioni più complesse dei dati. L'addestramento supervisionato consiste nel fornire gli input e gli output desiderati per l'ultimo livello. Se l'input sono immagini, i vettori x_t in genere non corrispondono ai pixel, ma a feature di alto livello estratte da una CNN, mentre se sono parole, i vettori x_t sono il risultato di word embedding.

111. Citare alcuni esempi di applicazioni pratiche di Deep RNN o RNN

Le principali applicazioni di una rete RNN sono:

- **Language modeling**: generazione di testi
- **Text classification**: sentiment analysis
- **Automatic speech recognition**: traduzione da parlato a testo
- **Caption generation**: generazione della descrizione di un'immagine
- **Machine translation**: traduzione tra lingue, creazione di riassunti e generazione di commenti
- **Question answering**: risponde in linguaggio naturale a domande in linguaggio naturale
- **Digital assistant**: interazione in linguaggio naturale con un assistente digitale

112. Qual è l'idea alla base del Q-learning? Che cosa indica la funzione $Q(s, a)$ nell'ambito del Q-Learning?

L'idea alla base del Q-learning è quella di massimizzare, a partire da uno stato, il **discounted future reward**, che può essere definito ricorsivamente con la seguente formula: $R_t = r_t + \gamma \times R_{t+1}$

Nel Q-learning la funzione $Q(s, a)$ indica l'ottimalità dell'azione a quando ci si trova in stato s . La funzione Q va appresa in modo da massimizzare il discounted future reward, quindi si pone $Q(s_t, a_t) = \max R_{t+1}$

113. Qual è l'obiettivo del Reinforcement Learning e cosa si intende con "episodio"? Fare un esempio AGGIUNGERE IMMAGINE SLIDE 17 DEEP LEARNING 2

L'obiettivo del reinforcement learning è apprendere un comportamento ottimale a partire da una serie di esperienze passate.

Un agente esegue **azioni** (a) che modificano l'ambiente, provocando passaggi di **stato** (s) e generando risultati. Se questi sono positivi, l'agente riceve un **reward** (r), che può essere ritardato nel tempo rispetto all'azione che l'ha determinata. Un episodio è una sequenza finita di stati, azioni e reward. Ad ogni stato, l'obiettivo è scegliere l'azione che massimizza il **future reward** R_t (somma dei reward futuri). Siccome nelle applicazioni reali l'ambiente è spesso stocastico (non è detto cioè che un'azione determini sempre la stessa sequenza di stati e reward), si dà maggior peso ai reward temporalmente vicini (**discounted future reward**). **Esempio:** un braccio metallico deve prendere una pallina da un piedistallo. Lo stato corrisponde alla posizione (x, y) della pinza rispetto alla pallina + lo stato della pinza (aperta/chiusa). Le azioni possibili sono ruotare i giunti del braccio e aprire/chiudere la pinza. I reward sono: +1 per un avvicinamento alla pallina, -1 per un allontanamento, -0.2 per un movimento a pinza chiusa e +100 per la cattura della pallina.

114. Cosa sono i one-hot-vectors? non riscontrato nelle slide

I one-hot-vectors sono una rappresentazione dei vettori target. Dato un dizionario di dimensione m parole, l'entry in posizione k è codificata con un vettore binario di dimensione m , in cui tutti gli elementi hanno valore 0, tranne quello in posizione k , che assume valore 1. Questa rappresentazione non è utile per la codifica di semantica e similarità tra parole.

115. Nell'ambito di CNN, che cosa si intende per feature map? è molto riassunta, vedi slide 18 deep 1

Per comprendere le connessioni fra i livelli di reti deep come CNN, i neuroni di ciascun livello vengono rappresentati come organizzati in griglie o volumi 3D. La terza dimensione depth individua le diverse feature map. Una feature map è formata da un insieme di neuroni appartenenti alla stessa depth. Le feature map sono importanti perché i pesi sono condivisi proprio a livello di feature map, cioè i neuroni di una stessa feature map processano porzioni diverse del volume di input nello stesso modo.

116. Perché nelle reti deep viene utilizzata la funzione di attivazione Relu invece della sigmoide?

Nelle reti deep si usa la funzione Relu (Rectified Linear) al posto della sigmoide perché quest'ultima causa problemi di vanishing gradient nella retro-propagazione del gradiente. Infatti, la derivata della sigmoide è minore di 1 e applicando la derivazione a catena si arriva a moltiplicare termini molto minori di 1 nei livelli lontani dell'output, fino a portare la derivata a valere 0 e ad annullare il gradiente. L'utilizzo di Relu risolve il problema, in quanto la derivata vale 0 per valori negativi o nulli di net, mentre per valori positivi non ha alcuna saturazione e porta ad attivazioni sparse che conferiscono maggiore robustezza.

COMPENDIO DOMANDE ML

FONDAMENTI

1. Cosa si intende per K-fold cross-validation?

Consiste nel suddividere l'insieme di tutti i pattern in due insiemi disgiunti: il **train set** e il **test set**; il train set viene poi ulteriormente suddiviso in **K** subset (fold) che vengono utilizzati iterativamente come training e validation set: per ogni iterazione i si usano i fold $[1, 2, \dots, i-1, i+1, \dots, k]$ come training e il fold i come validation set. La prestazione è calcolata come media o mediana delle k prestazioni ottenute.

CLASSIFICAZIONE

2. Nella formulazione dell'SVM lineare la funzione obiettivo richiede di massimizzare il margine. L'ottimizzazione è però vincolata; in cosa consistono i vincoli? quanti sono?

L'iperpiano ottimo secondo SVM è quello che soddisfa i vincoli di separazione dei pattern e massimizza il margine tau (o alternativamente minimizza il suo inverso):

- minimizza: $\|w\|^2/2$
- vincoli: $y_i[w \cdot x_i + b] - 1 \geq 0$ per $i = 1, \dots, n$

I pattern del training set che giacciono sul margine (cerchi pieni in figura) sono detti *support vector*. Tali pattern, che costituiscono i casi più complessi, definiscono completamente la soluzione del problema, che può essere espressa come funzione di solo tali pattern, indipendentemente dalla dimensionalità dello spazio d ed al numero n di elementi in TS.

3. Nell'ambito dei multi-classificatori quali sono le più comuni tecniche di fusione a livello di decisione e di confidenza?

VEDI DOMANDA 48

4. Indicare la formula di Bayes per la probabilità a posteriori, definendo i termini.

Per ogni $w_i \in W$ e per ogni $x_i \in V$ indichiamo con $P(w_i|x)$ la probabilità a posteriori di w_i dato x , ovvero la probabilità che avendo osservato il pattern x la classe di appartenenza sia w_i .

Per il teorema di Bayes: $P(w_i|x) = \frac{p(x|w_i) \cdot P(w_i)}{p(x)}$

Dato un pattern x da classificare in una delle s classi w_1, w_2, \dots, w_s di cui sono note le probabilità a priori $P(w_1), P(w_2), \dots, P(w_s)$ e le densità di probabilità condizionali $p(x|w_1), p(x|w_2), \dots, p(x|w_s)$ la regola di classificazione di Bayes assegna x alla classe b per cui è massima la probabilità a

posteriori (massimizzare la probabilità a posteriori significa massimizzare la densità di probabilità condizionale tenendo comunque conto della probabilità a priori delle classi).

5. Cosa si intende per approccio parametrico e non-parametrico nell'ambito della classificazione? Fare un esempio di classificatore parametrico e non parametrico.

VEDI DOMANDA 23

6. Con quali tecniche si può estendere SVM da 2 a più classi?

Per gestire con SVM problemi con più di 2 classi le tecniche più utilizzate sono:

- One-Against-One
- One-Against-All

VEDI DOMANDA 47

7. Nell'ambito dei multi-classificatori come si può ottenere indipendenza tra i singoli classificatori utilizzati?

L'indipendenza è normalmente ottenuta:

- Utilizzando feature diverse (non correlate o poco correlate)
- Utilizzando algoritmi diversi per l'estrazione delle feature
- Utilizzando diversi algoritmi di classificazione
- Addestrando lo stesso algoritmo di classificazione su porzioni diverse del training set (bagging)
- Insistendo con l'addestramento sui pattern erroneamente classificati (boosting)

8. Quali sono i parametri di una distribuzione multinormale?

I parametri di una distribuzione multinormale sono il vettore medio e la matrice di covarianza.

La densità di probabilità nella distribuzione multinormale ($d > 1$) è:

$$p(x) = \frac{1}{(2\pi)^{d/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^t \Sigma^{-1} (x-\mu)}$$

Dove $\mu = [\mu^1, \mu^2, \dots, \mu^d]$ è il vettore medio e $\Sigma = [\sigma^{ij}]$ è la matrice di covarianza ($d \times d$).

- Si assume che i vettori siano di tipo colonna. L'apice t (trasposto) li trasforma in righe
- $|\Sigma|$ e Σ^{-1} sono rispettivamente il determinante e l'inversa di Σ
- La matrice di covarianza è sempre simmetrica e definita positiva, pertanto ammette inversa. Essendo simmetrica il numero di parametri che la definisce è $d \cdot (d + 1)/2$

- Gli elementi diagonali σ^{ii} sono le varianze dei rispettivi x^i (ovvero $(\sigma^i)^2$; gli elementi non diagonali σ^{ij} sono le covarianze tra x^i e x^j :
 - se x^i e x^j sono statisticamente indipendenti $\sigma^{ij} = 0$
 - se x^i e x^j sono correlati positivamente $\sigma^{ij} > 0$
 - se x^i e x^j sono correlati negativamente $\sigma^{ij} < 0$
9. La posizione e forma dell'ellissoide di una distribuzione multinormale come è influenzata da μ e Σ ?

L'ellissoide è influenzato da ($d = 2$):

- $\mu = [\mu^1, \mu^2]$ controlla la posizione del centro
- σ^{11} e σ^{22} determinano l'allungamento sui due assi dell'ellisse
- $\sigma^{12} = \sigma^{21}$ controlla la rotazione dell'ellisse rispetto agli assi cartesiani
 - Se $= 0$ (matrice di covarianza diagonale), la distribuzione multinormale è definita come prodotto di d normali monodimensionali. In tal caso gli assi dell'ellisse sono paralleli agli assi cartesiani (es. Naive Bayes Classifier)
 - Se > 0 x^1 e x^2 sono positivamente correlate (quando aumenta x^1 aumenta anche x^2)
 - Se < 0 x^1 e x^2 sono negativamente correlate (quando aumenta x^1 cala x^2)
- Gli assi dell'ellisse sono paralleli agli autovettori di Σ .

REGRESSIONE

10. Qual è la funzione obiettivo in formato matriciale della multiple linear regression?

In formato matriciale la funzione obiettivo da minimizzare (least square) può essere scritta come:

$$(X^t X) \beta^* = X^t y$$

RIDUZIONE DIMENSIONALITA'

11. Indicare le differenze tra le tecniche di riduzione di dimensionalità PCA e LDA.

La tecnica PCA (principal component analysis) è una trasformazione non supervisionata che esegue un mapping lineare con l'obiettivo di preservare al massimo l'informazione dei pattern. Nella tecnica LDA (linear discriminant analysis) il mapping è ancora lineare ma il mapping è supervisionato. Mentre PCA privilegia le dimensioni che rappresentano al meglio i pattern, LDA privilegia le dimensioni che discriminano al meglio i pattern del training set.

12. Quali sono le più note tecniche di riduzione di dimensionalità? Quali i loro tipici utilizzi?

Le più note tecniche di riduzione di dimensionalità sono PCA, LDA e t-SNE. La prima si utilizza per la codifica di un'immagine, la seconda per il riconoscimento facciale e l'ultima per la visualizzazione in 2D di MNIST (digit scritti a mano).

RETI NEURALI

13. Come può essere matematicamente definita la loss function (su un singolo pattern) per l'addestramento di una rete neurale?

Sia $z = [z_1, z_2, \dots, z_s]$ l'output prodotto dalla rete (forward propagation) in corrispondenza del pattern $x = [x_1, x_2, \dots, x_d]$ di classe g fornito in input; l'output desiderato è $t = [t_1, t_2, \dots, t_s]$, dove $t_i = 1$ per $i = g$, $t_i = -1$ altrimenti. Scegliendo come loss function la somma dei quadrati degli errori, l'errore (per il pattern x) è:

$$J(w, x) \equiv \frac{1}{2} \sum_{c=1 \dots s} (t_c - z_c)^2$$

che quantifica quanto l'output prodotto per il pattern x si discosta da quello desiderato. La dipendenza dai pesi w è implicita in w . L'errore $J(w)$ sull'intero training set è la media di $J(w, x)$ su tutti i pattern x appartenenti al training set.

DEEP LEARNING

14. Nell'ambito di CNN, che cosa si intende per connessioni locali e condivisione di pesi?

Processing locale: i neuroni sono connessi solo localmente ai neuroni del livello precedente. Ogni neurone esegue quindi un'elaborazione locale. Forte riduzione numero di connessioni.

Pesi condivisi: i pesi sono condivisi a gruppi. Neuroni diversi dello stesso livello eseguono lo stesso tipo di elaborazione su porzioni diverse dell'input. Forte riduzione numero di pesi.

15. Come opera un livello di pooling in una CNN?

Un livello di pooling esegue un'aggregazione delle informazioni nel volume di input, generando feature map di dimensione inferiore. Obiettivo è conferire invarianza rispetto a semplici trasformazioni dell'input mantenendo al tempo stesso le informazioni significative ai fini della discriminazione dei pattern.

L'aggregazione opera (generalmente) nell'ambito di ciascuna feature map, cosicché il numero di feature map nel volume di input e di output è lo stesso. Gli operatori di aggregazione più utilizzati

sono la media (Avg) e il massimo (Max): entrambi «piuttosto» invarianti per piccole traslazioni. Questo tipo di aggregazione non ha parametri/pesi da apprendere.

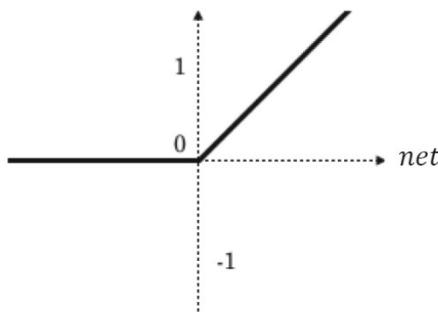
16. Quali sono le condizioni necessarie affinché le tecniche di deep learning siano più efficaci di altri approcci di machine learning?

Le tecniche di deep learning sono più efficaci di altri approcci di machine learning quando si lavora con grandi quantità di dati di training (dataset etichettati di grandi dimensioni) e quando sono disponibili GPU con migliaia di core e GB di memoria interna.

17. Come è definita la funzione di attivazione Relu? Perché consente di addestrare reti neurali profonde limitando il problema del vanishing gradient?

Nelle reti deep si usa la funzione Relu (Rectified Linear) al posto della sigmoide perché quest'ultima causa problemi di vanishing gradient nella retro-propagazione del gradiente. Infatti, la derivata della sigmoide è minore di 1 e applicando la derivazione a catena si arriva a moltiplicare termini molto minori di 1 nei livelli lontani dell'output, fino a portare la derivata a valere 0 e ad annullare il gradiente. L'utilizzo di Relu risolve il problema, in quanto la derivata vale 0 per valori negativi o nulli di net, mentre per valori positivi non ha alcuna saturazione e porta ad attivazioni sparse che conferiscono maggiore robustezza:

$$f(\text{net}) = \max(0, \text{net})$$



18. Che cosa codifica la funzione Q nell'ambito dell'approccio Q-learning? Quali sono gli input che ne determinano il valore?

L'idea alla base del Q-learning è quella di massimizzare, a partire da uno stato, il **discounted future reward**, che può essere definito ricorsivamente con le seguente formula: $R_t = r_t + \gamma \cdot R_{t+1}$

Nel Q-learning la funzione $Q(s, a)$ indica l'ottimalità dell'azione a quando ci si trova in stato s . La funzione Q va appresa in modo da massimizzare il discounted future reward, quindi si pone $Q(s_t, a_t) = \max R_{t+1}$

19. Nell'ambito delle reti neurali, quali sono le principali differenze di CNN rispetto a MLP?

Un Multilayer Perceptron (MLP) è una rete feedforward formata da almeno 3 livelli, di cui almeno uno hidden, con funzioni di attivazione non lineari.

Le Convolutional Neural Network (CNN) sono progettate con neuroni connessi solo localmente e con pesi condivisi e hanno quindi una maggiore efficienza con forti riduzioni del numero di pesi e connessioni.

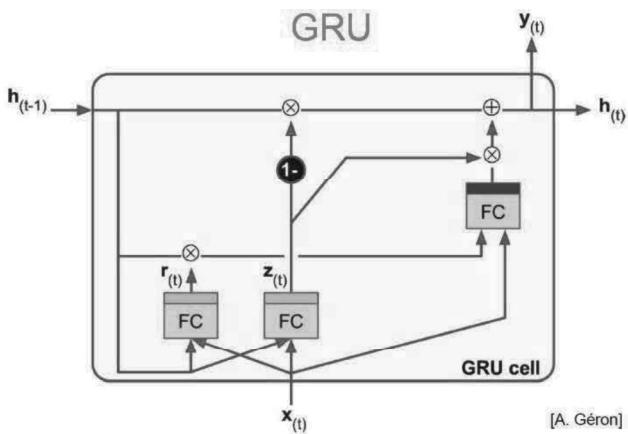
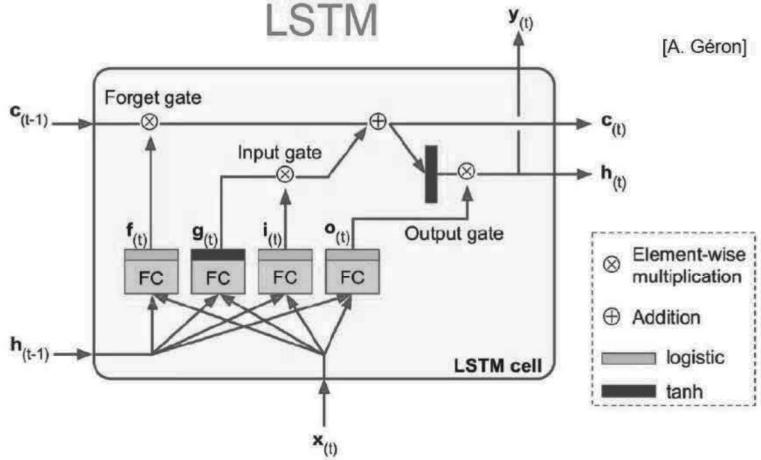
Quindi, le principali differenze rispetto a MLP sono:

Processing locale: i neuroni sono connessi solo localmente ai neuroni del livello precedente. Ogni neurone esegue quindi un'elaborazione locale. Forte riduzione numero di connessioni.

Pesi condivisi: i pesi sono condivisi a gruppi. Neuroni diversi dello stesso livello eseguono lo stesso tipo di elaborazione su porzioni diverse dell'input. Forte riduzione numero di pesi.

IMMAGINI UTILI

DOMANDA 109



DOMANDA 113

