appunti
www.centroappunti.it
Corso Luigi Einaudi, 55 - Torino

Appunti universitari
Tesi di laurea
Cartoleria e cancelleria
Stampa file e fotocopie
Print on demand
Rilegature

NUMERO: 1980A - ANNO: 2016

A P P U N T I

STUDENTE: Adriana Palma

MATERIA: Classificazione e interpretazione di dati biomedici - Teoria + Domande Esame - Prof. Balestra

Il presente lavoro nasce dall'impegno dell'autore ed è distribuito in accordo con il Centro Appunti.
Tutti i diritti sono riservati. È vietata qualsiasi riproduzione, copia totale o parziale, dei contenuti inseriti nel presente volume, ivi inclusa la memorizzazione, rielaborazione, diffusione o distribuzione dei contenuti stessi mediante qualunque supporto magnetico o cartaceo, piattaforma tecnologica o rete telematica, senza previa autorizzazione scritta dell'autore.

ATTENZIONE: QUESTI APPUNTI SONO FATTI DA STUDENTI E NON SONO STATI VISIONATI DAL DOCENTE.
IL NOME DEL PROFESSORE, SERVE SOLO PER IDENTIFICARE IL CORSO.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 3 di 120

CLASSIFICAZIONE E INTERPRETAZIONE DI DATI BIOMEDICI

Anno 2015/2016

Adriana Palma

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 3 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 5 di 120

- Raggruppare

A ogni possibile approccio corrisponde una serie di metodi (algoritmi generici, reti naturali, logica fuzzy, k-means) che appartengono a campi diversi (CI computational intelligence, statistica, clustering, softcomputing…). Tra i vari metodi a disposizione bisogna essere in grado di scegliere il migliore in base al problema che si ha, tenendo presente che lo stesso metodo può essere usato in ambiti diversi. Citazione: "Se vogliamo fare analisi di dati di un buon livello, non abbiamo bisogno di particolare complessità dal punto di vista matematico, ma abbiamo bisogno di un buon senso, di sapere che cosa le formule rappresentano e saper interpretare i risultati matematici sulla base del problema iniziale".

Rapporto tra dato, informazione e conoscenza:
L'informazione gioca un ruolo chiave nell'interpretazione dei dati e nel prendere decisioni. È innanzitutto essenziale conoscere il tipo di informazione e capire la differenza tra il dato, l'informazione e la conoscenza. L'informazione è data dai dati elaborati attraverso la conoscenza. Ad esempio, febbre: anche se misuro la temperatura, se non so a che cosa corrisponde la malattia, non ottengo nessuna informazione utile.
Mano a mano che la conoscenza si complica, diventa più difficile applicarla manualmente ai dati, e quindi ci devono essere dei sistemi per individuare i dati più significativi.
Come otteniamo la conoscenza?

- A priori, viene da qualcuno che ha già affrontato il problema in modo approfondito
- Conoscenza che otteniamo attraverso i dati, cioè apprendimento data driven; ha dei dati validi o dei certificati e attraverso di essi ricavo la conoscenza, ma è in genere più difficile da spiegare

Il percorso logico è quindi:
Dati (acquisizione, incertezza, validità) → (processing/interpretazione) informazione → (modelli) conoscenza→dati.
I dati devono essere acquisiti con una procedura certa, in modo da limitare l'incertezza e da poterne verificare la validità. Noi generiamo continuamente conoscenza. L'informazione gioca un ruolo fondamentale quando interpretiamo i dati e prendiamo una decisione, è quindi essenziale sapere di che informazioni disponiamo e conoscere la differenza tra dati, informazione e conoscenza.

Il corso si propone di:

- Illustrare metodi diversi
- Insegnare ad usare alcuni metodi evidenziando la particolarità del metodo stesso
- Insegnare come validare i risultati ottenuti
- Insegnare ad interpretare i risultati in funzione del contesto e dell'obiettivo
- Evidenziare le difficoltà nella scelta del metodo e come metodi diversi possono essere utilizzati in combinazione tra loro.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 5 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 7 di 120

Ruolo dei dati nella costruzione del processo: L'attività della costruzione del processo consiste nel collegare le caratteristiche alle disponibilità dei dati. Potrebbe essere utile rappresentare le principali caratteristiche dei dati mediante **statistiche descrittive**. I metodi appartengono ai campi. Ad ogni campo corrisponde un set di metodi: intelligenza computazionale, statistici, clustering, tecniche non parametriche, machine learning, ottimizzazione, soft computing.

Introduzione alla computational intelligence: Una definizione ampia di Computational Intelligence (CI): riunisce una serie di metodi che derivano dallo studio dei meccanismi di adattamento che regolano l'ambiente e la riproduzione in un ambiente complesso, incerto e che subisce cambiamenti; in pratica mimano fenomeni naturali e biologici per sviluppare sistemi in grado di costruire soluzioni a problemi complessi e sfaccettati. Sono dei meccanismi adattativi che permettono di affrontare problemi complessi e sono tutti molto semplici concettualmente. Gli algoritmi devono essere adattati al problema.
CI è parte del campo di intelligenza artificiale (AI).
Gli altri approcci importanti AI sono ragionamenti basato sul caso, ragionamento deduttivo, sistemi esperti, sistemi di apprendimento automatico.
Computational Intelligence (CI) Methods: metodi CI si ispirano alla natura e ella biologia. In altre parole CI imita il fenomeno naturale e biologico per i sistemi in grado di costruire la soluzione di problemi difficili e sfaccettati in via di sviluppo.

- calcolo evolutivo (EC)
- Reti neurali
- swarm intelligence
- sistemi Fuzzy
- sistemi immunitari artificiali

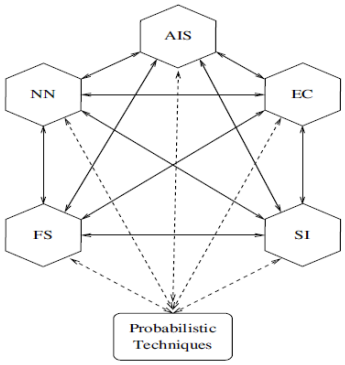


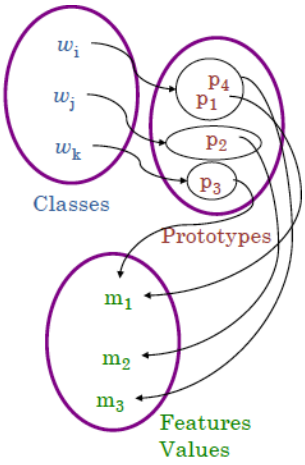
Figure 1.1 Computational Intelligence Paradigms

Si possono costruire sistemi in cui si usano più di uno di questi metodi, cioè si creano sistemi ibridi. Sono dei metodi molto diversi tra loro e non deterministici, cioè contengono parti in cui si fa riferimento alla probabilità, e questo comporta un'ulteriore difficoltà di implementazione. Si usa il termine *soft computing* per indicare un'unione di paradigmi CI e metodi probabilistici.
Le principali classi dei problemi trattati da CI sono:

- **Ottimizzazione**: mira a trovare la soluzione di un problema specifico che massimizza o minimizza (generalmente che ottimizza) una data funzione obiettivo.
- **Apprendimento sorvegliato:** è il processo di apprendimento delle proprietà intrinseche di un sistema in termini di rapporto tra ingressi e uscite, utilizzando una serie di esempi caratterizzati da variabili di input e output di destinazione.
- **Apprendimento non supervisionato:** è generalmente legato a problemi di clustering.
- **Classificazione:** intende associare un elemento alla classe corretta.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 7 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 9 di 120



- numeri reali
- serie temporali (per esempio l'acquisizione in campioni di un segnale)
- immagini
- temporal data (Δ, Δ normalizzata, durata dei dati, sequenza di dati)
- dati fuzzy

Per progettare un classificatore, dobbiamo affrontare il problema di estrazione delle caratteristiche. La chiave è quella di scegliere e per estrarre le caratteristiche che:

1. sono computazionalmente fattibili
2. portano ad avere un buon classificatore (ad esempio errori di classificazione inferiore)

L'insieme di caratteristiche usate per rappresentare una classe è in genere detto pattern (modello) o prototipo.

- <u>misura di similarità</u>: ce ne sono molto e e le più usate sono (per vettori numerici) :

- distanza euclidea o norma L2 (media)

$$d(x,y) = \|x-y\| = \sqrt{(x-y)^T(x-y)} = +\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2}$$

- norma L1 (fa la distanza complessiva)

$$d(x,y) = \|x-y\| = |(x-y)| = \sum_{i=1}^{k}|x_i - y_i|$$

- norma L_∞ (mette in evidenza questioni particolari), k è il numero delle feature

$$d(x,y) = \max_k |x_i - y_i|$$

- <u>regole di classificazione</u>, cioè come posso dire se un elemento appartiene o non ad una classe (o cluster); in genere si basano sul confronto tra l'elemento che deve essere classificato e il prototipo della classe: si associa l'elemento alla classe per la quale la differenza elemento-prototipo è minore. Nasce quindi il problema di calcolare tale distanza. Ci sono differenti regole di classificazione, ma tutte loro si basano su **misure di similarità**, ovvero il tool che usiamo automaticamente per comparare due o più elementi.

  Possibili misure di similarità per le stringhe sono:
  - lunghezza della sottostringa più lunga in comune tra le due stringhe
  - numero di caratteri identici nelle due stringhe

  Possibile misura di somiglianza per insiemi di elementi è:
  - tra i due insiemi

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 9 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 11 di 120

- 89% di classificazioni corrette che è un risultato ragionevole dato che è impossibile avere il 100%

2. analisi di immagini di carotidi: sviluppo di uno strumento completamente automatico per la segmentazione di immagini dell'arteria carotidea. Si ha a disposizione un ampio data set e dalle immagini vengono estratte 211 caratteristiche che poi vengono ridotte per fare il classificatore fino ad arrivare a 13. A partire da queste caratteristiche si sono classificate le immagini usando la logica fuzzy e le reti neurali.
3. Ambito di progettazione di un dispositivo medico; analisi del cammino sia segnale basografico, cioè come si appoggia il piede, sia comportamento delle articolazioni, cioè goniometria, sia attività muscolare, quindi segnale EMG. Problema: chi legge i dati deve avere una profonda conoscenza tecnologica e anatomica, perché da in input tanti parametri che possono assumere valori molto vari in soggetti patologici; si è cercato di riassumere in un unico indicatore tutti i vari parametri. Sono stati realizzati realizzati classificatori diversi per le tre parti (basografia, goniometria, EMG (il terzo non è ancora stato realizzato) e a ogni paziente applico i classificatori su entrambi gli arti inferiori. Il primo indicatore tiene conto di basografia destra, basografia sinistra, goniometria destra, goniometria sinistra; il secondo indicatore è quello finale e tiene conto di basografia egoniometria finale. Si è usata la logica fuzzy.

VERIFICA E VALIDAZIONE

Non vi è alcuna comprensione chiara nell'uso dei termini di verifica e validazione. La differenza tra verifica e la validazione è:

- La verifica è il processo sperimentale per garantire che avete fatto le cose correttamente, ad esempio il prodotto soddisfa i requisiti (1° fase di analisi, ad esempio va a buon fine anche se non porta). Ogni esempio che abbiamo è CLASSIFICATO
- La validazione è il processo sperimentale per dimostrare che l'attuazione di progettazione (ad esempio, l'attuale prodotto, processo o servizio) soddisfa le esigenze degli utenti (verifico che il mio sistema risponda al problema di partenza); è il processo per garantire che hai fatto cosa giusta. Soddisfa le esigenze degli utenti: PRESTAZIONI ROBUSTEZZA (quando i risultati ottenuti sono ripetibili; associato al sistema di classificazione, es % o n° di elementi classificati nella classe corretta).

Se la verifica non va a buon fine, si torna alla costruzione, mentre se la validazione non va a buon fine, si torna o alla costruzione o alla concettualizzazione.

Bisogna prendere in considerazione due elementi:

- se verifica le specifiche
- se è utile per gli utenti per i quali era stato costruito

In pratica la verifica consiste nel vedere se ho costruito nel modo giusto, mentre la validazione consiste nel vedere se ho costruito la cosa giusta. Quindi nella fase di verifica controllo che tutti gli elementi che ho a disposizione vengono classificati, mentre nella validazione controllo quanto il sistema è robusto e quali sono le sue performance.

Per quanto riguarda le specifiche, tutti gli elementi che ho a disposizione devono venir classificati: più il campione che uso è rappresentativo, più è bassa la probabilità di trovare elementi che non vengono classificati. Devo in seguito verificare la robustezza: i parametri su cui si basa il sistema sono critici rispetto alla costruzione del classificatore? Devo quindi fare l'analisi di sensitività,

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 11 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 13 di 120

casi precedenti; s'ipotizza che il singolo elemento non modifichi in modo sostanziale il classificatore.

- Leave one out (3° metodo: ne lascia uno fuori, quando il data set di partenza è piccolo): usato quando il numero degli elementi è basso.

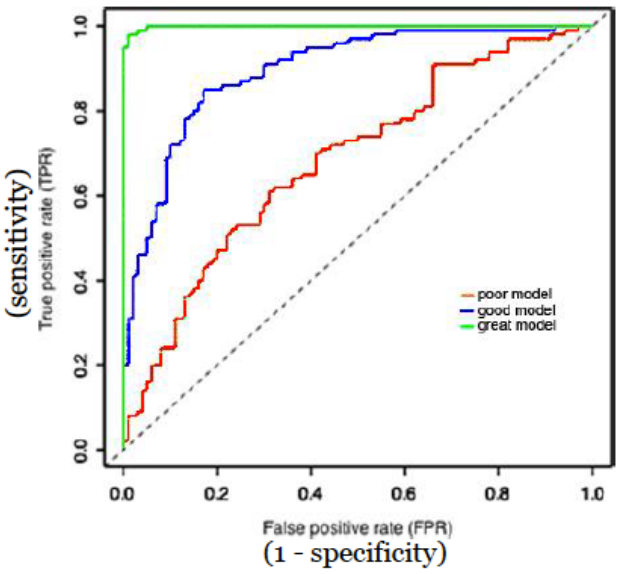
Con questo metodo ci si devono aspettare prestazioni leggermente più basse. In una situazione molto specifica, come il test di screening, ho un certo errore, ma voglio essere sicuro che se una persona ha una certa patologia, sono in grado di riconoscerla. Quindi è richiesta sia sensitività che specificità, cioè devo riconoscere sia i soggetti patologici che quelli che non lo sono. Si possono avere sia falsi negativi (malati ritenuti sani) che falsi positivi (sani ritenuti malati) ed è molto importante ridurli, soprattutto i falsi negativi.

Possiamo definire:

- sensitività: probabilità che il test sia positivo quando la persona ha la patologia: soggetti positivi individuati/soggetti positivi totali. VP/VP+FN
- specificità: probabilità che il test sia negativo quando il soggetto non presenta la patologia: persone riconosciute sane/ totale persone sane VN/VN+FP
- PV^{+}: probabilità che una persona con una patologia abbia un test positivo VP/VP+FP
- PV^{-}: probabilità che una persona sana abbia il test negativo VN/VN+FN

Curva ROC

Una caratteristica (ROC) Curva Ricevitore Operante è un diagramma grafico che illustra le prestazioni di un sistema di classificazione binario con la sua soglia di discriminazione che varia. Essa mostra il vero tasso positivo (sensibilità) contro il tasso di falsi positivi (1 - specificità) a variare delle impostazioni della soglia. La diagonale divide lo spazio ROC. Punti sopra la diagonale rappresentano buoni risultati della classificazione, punti sotto la linea di cattivi risultati.



L'area sotto la curva ROC (AUC) è una misura efficace dell'accuratezza del classificatore. Può essere interpretata come il valore medio della sensibilità per tutti i possibili valori di specificità.

- La AUC massima = 1 significa che il test diagnostico è perfetto nella differenziazione tra le classi.
- AUC = 0,5 significa che la discriminazione è casuale (curva trova sulla linea diagonale nello spazio ROC)
- AUC = 0 significa prova classificare correttamente tutti gli elementi.

Soluzioni Complesse

Quando le performance non sono adeguate, possiamo implementare più classificatori, ad esempio:

- in PARALLELO

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 13 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 15 di 120

3. METODI STATISTICI

Primo problema: come posso descrivere le caratteristiche di un sistema di elementi? Un elemento è in genere caratterizzato da molti dati (circa 200 in genere).
Esempio: trial: studio controllato in cui vengono inseriti dei soggetti che possono essere tutti patologici oppure sia patologici che sani. In genere l'obiettivo è quello di identificare un comportamento patologico. Devo raccogliere dati su N soggetti patologici e M soggetti non patologici e possiamo stabilire come raccogliere i dati, in che intervalli regolari… La condizione di riferimento viene in genere detta condizione basale e spesso si vuole controllare la sua evoluzione e confrontarla con i dati rilevati.
La soluzione a questo problema è la *statistica descrittiva*: essa è usata in primo luogo per farsi un'idea sui dati e dopo per usarli nei test statistici e per avere indicazioni sugli errori nei grafici e nei dati finali.
Ci sono due tipi di statistica:

- Descrittiva: non pone alcun problema, usata semplicemente per descrivere il campione con cui si ha a che fare. In particolare viene usata per avere un'idea sui dati con cui si ha a che fare, nei test statistici e per valutare gli errori associati agli output
- Inferenziale: si fanno inferenze su come un dato valore rappresenta il valore dell'intera popolazione; in pratica include tecniche che permettono di usare un campione per fare generalizzazioni sull'intera popolazione da cui il campione è tratto.

In entrambi i casi il problema è la rappresentatività del campione rispetto alla popolazione: in genere si critica il fatto che nei trial i soggetti sono molto più omogenei che nella realtà e si critica il numero minimo dei soggetti necessario: ci si dimentica cioè di validare la rappresentatività del campione. La popolazione include tutti gli oggetti di interesse mentre il campione solo una porzione della popolazione.
I parametri sono associati a tutta la popolazione e vengono in genere indicati con lettere greche (μ,σ), mentre le statistiche si riferiscono solo ai campioni e vengono in genere indicati con lettere romane (x, s).
Quando si usa il metodo inferenziale è importante che il campione rappresenti accuratamente la popolazione. Ci sono 5 metodi per estrarre il campione:

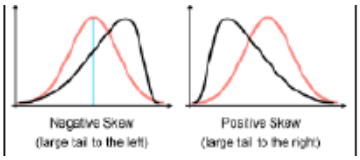
- In modo random: tutti gli elementi della popolazione hanno la stessa probabilità di essere estratti; anche se è la tecnica migliore in genere è di difficile applicazione
- Campionamento sistematico: prendo un campione ogni n dopo averli riordinati; è più facile rispetto a quello random
- Convenience sampling: chi si presenta viene inserito nel campione; è la peggiore ma anche più facile da fare
- Cluster sampling: si divide la popolazione in gruppi e si estrae da ogni gruppo: in genere la divisione si fa geograficamente – i gruppi sono chiamati cluster o blocchi
- Stratified sampling: la popolazione viene divisa in gruppi detti strata in base alle loro caratteristiche – un campione viene scelto da ogni gruppo in base a uno dei primi tre metodi precedenti (random, sistematico o convenience).

La terza è la tecnica peggiore, ma anche quella che viene usata più frequentemente – si usano i parametri trovati nel centro nel quale è stato organizzato il trial.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 15 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 17 di 120

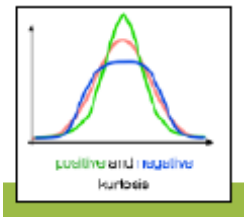
$$skewness = \frac{1}{(n-1)s^3}\sum_{i=1}^{n}(x_i - \bar{x})^3$$



Assume valori negativi quando è più grande a sinistra e positivi quando è più grande a destra

- **Kurtosis**: rispetto ad una distribuzione gaussiana controlla se il picco è più in alto (la distribuzione è più stretta rispetto all'andamento classico) o più in basso (la distribuzione è più larga rispetto all'andamento classico)

$$kurtosis = \frac{1}{(n-1)s^4}\sum_{i=1}^{n}(x_i - \bar{x})^4$$



Spesso abbiamo bisogno di rappresentare graficamente dati e parametri: le 3 rappresentazioni più usate sono:

- Bar diagram o diagramma a barra: divido in gruppi vari elementi e poi calcolo la frequenza
- Scatter plot: metto a confronto due variabili discrete, una sull'asse x e una sull'asse y
- Box plot: un rettangolo che contiene il 50% dei dati, si divide il campione in percentili. Viene rappresentato mediante un rettangolo diviso in 2 parti da cui escono 2 segmenti: il rettangolo è delimitato dal primo al terzo quartile ed è diviso al suo interno dalla mediana; isegmenti sono delimitati dal minimo e dal massimo dei valori: si rappresentano in questo modo: 4 intervalli ugualmente popolati delimitati dai quartili (i quartili sono quei valori che dividono la popolazione in 4 parti di uguale numerosità). Questa rappresentazione dà l'andamento rispetto alla mediana: se la mediana sta a metà del rettangolo significa che essa corrisponde con la metà del range.

Statistica Inferenziale

Secondo problema: se il set di elementi del campione è preso da una popolazione più grande, come posso usarlo per studiare tutta la popolazione? Il campione come riflette tutta la popolazione?
La risposta è: statistica inferenziale, quindi non solo analizzare i dati, ma anche collegare i parametri trovati a quelli di tutta la popolazione, quindi creare un legame campione-popolazione. Se ho elementi poco rappresentativi della popolazione potrebbe non esserci alcun legame tra il parametro trovato in un campione e quello relativo alla popolazione. La statistica inferenziale si basa su:

- Probabilità [Pr(A)]: è definita come il rapporto tra i casi favorevoli e i casi possibili; può essere stimata calcolando la frequenza degli eventi.
- Probabilità condizionata [Pr(A/B)]: probabilità che si verifica A condizionato (dato) B.
- Variabile casuale o variabili random: in genere si indica con X ed è una variabile i cui possibili valori sono i risultati di un fenomeno casuale; possono essere sia discreti che continui.
- la distribuzione di probabilità di una variabile discreta causale è data dall'elenco di tutti i possibili valori che possono assumere le variabili con le relative probabilità che ciò accada; è anche detta funzione di probabilità.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 17 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 19 di 120

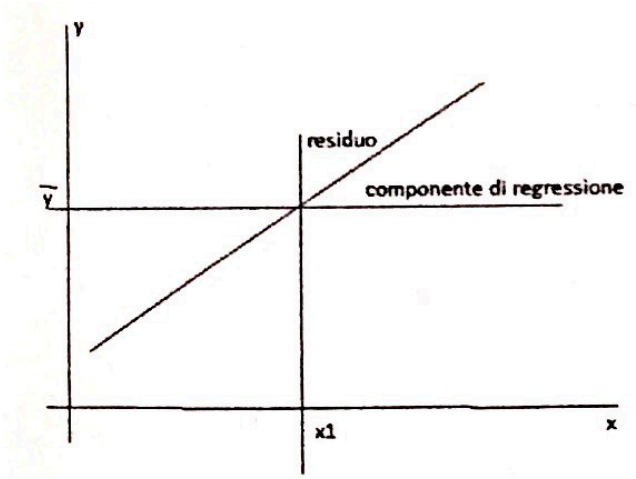
$d_i = y_i - \hat{y}_i$ (in pratica la differenza tra il valore reale e quello che ottengo con l'approssimazione). Voglio trovare una retta che minimizzi la somma dei d_i, anche se in realtà minimizzo il quadrato della distanza per non avere problemi di segno, cioè per evitare che valori uguali di segno opposto si semplifichino; minimizzo

$$s = \sum_{i=1}^{n} d_i^2$$

Con questo metodo ottengo $b = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sum(x_i - \bar{x})^2}$ $a = \bar{y} - b\bar{x}$

Qual è la bontà della regressione? In genere si tiene conto di due elementi:

- residui, cioè la differenza tra y_i e $\hat{y}_i$
- componente di regressione, cioè il valore stimato meno valor medio di y



Osserviamo che il valor medio di x e il valore medio di y si trovano sempre sulla retta per costruzione.
La regressione lineare fa parte di un ambito molto più ampio detto fittig: in questo caso il fitting è lineare, ma può anche essere quadratico o una qualsiasi altra funzione di grado opportuno.
Nel caso della regressione lineare, per avere un buon fitting bisogna avere la somma delle componenti di regressione molto più alta della somma dei residui in valore assoluto: a prescindere da questo se la somma dei residui è alta, il fitting non sarà comunque buono e la retta sarà lontana dai punti.
Affidabilità della stima dei coefficienti: posso sempre calcolare i coefficienti dato un certo data set, anche se non conosco la retta reale, cioè quella corrispondente alla popolazione. Anche se estraggo data set diversi dalla popolazione e calcolo i parametri, è praticamente impossibile riuscire a trovare i valori reali e troverei per ogni data set valori di a e b diversi; come si risolve questo problema? Con il **test di ipotesi**.
Si può determinare lo standard error dal punto di vista dei parametri a e b:

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 19 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 21 di 120

- Se $R^2 = 0$ non ci sono relazioni tra le due varianze

Gli errori possono essere correlati ed eteroschedastici, cioè la varianza di una variabile casuale varia tra le due diverse associazioni campionarie.
La funzione di matlab mvregress ipotizza una distribuzione normale degli errori – regressione di una matrice bidimensionale di variabili.

Test di ipotesi

Un'ipotesi statistica è un assunto fatto su un certo parametro e con test di ipotesi ci si riferisce alla procedura formale usata per accettare o rifiutare un'ipotesi statistica. Quando faccio delle inferenze compio dei test per capire quanto sono affidabili. L'idea alla base dei test di ipotesi è quella di contrapporre due ipotesi, una H_o detta ipotesi nulla che è quella che si vuole disprovare, l'altra detta H_1 che è quella che si vuole provare essere vera e che prende il nome di ipotesi alternativa. Ad ogni ipotesi è associata una certa funzione densità di probabilità.
Associato alla decisione che vado a prendere ci sono due errori:

- Errore di tipo 1: capita quando si rifiuta l'ipotesi nulla quando essa è vera; la probabilità di commettere tale errore è detta **significance level** o **α** (disprovare H_0 quando è vera, posso impostare α a priori)
- Errore di tipo 2: non rifiuto l'ipotesi nulla che è falsa; la probabilità di commettere tale errore si indica con **β**. La probabilità di non commettere un errore di tipo 2 è detta **potenza del test** (considerare H_0 vera quando è falsa).

Il test avviene andando a prendere il valore critico e confrontandolo con il valore attuale:

- Se valore attuale > valore critico allora prendo una decisione
- Se valore attuale < valore critico allora prendo l'altra decisione

Diventa significativo capire quando le due funzioni densità di probabilità si sovrappongono. In genere nei test di ipotesi si fissa un valore per α e diventa difficile calcolare β; α è detto livello di significatività del test.
Il piano di analisi include le regole di decisione per rifiutare l'ipotesi nulla; nella pratica gli analisti descrivono queste regole in due modi equivalenti:

- tramite il **p-value**: si misura la forza di una prova a supporto dell'ipotesi nulla, il p-value è la probabilità di osservare un test statistico come estremo di S (test statistico), assumendo che l'ipotesi nulla sia vera. Se il p-value è minore del significative level, si rifiuta l'ipotesi nulla.
- Tramite una **regione di accettazione**, cioè un range di valori. Se il test statistico fallisce nella regione di accettazione, l'ipotesi nulla non viene rifiutata. Viene scelta in modo tale che la probabilità di fare un errore di tipo 1 sia uguale al significative level. I valori esterni a tale regione sono detti regione di rifiuto: se il test fallisce in tale regione, l'ipotesi nulla viene rifiutata. Si dice che l'ipotesi è stata rifiutata con un livello α di significatività.

Dobbiamo quindi stimare i coefficienti α e β del campione che abbiamo e ci sarà un certo errore dovuto al fatto che non vengono calcolati sull'intera popolazione.
Standard-error= varianza dello stimatore= varianza popolazione/numero di elementi= $\frac{\sigma^2}{n}$
Ci da un'idea di quanto la media dei valori stimati $\bar{x}$ è diversa da μ. Ovviamente tanto più grande è n tanto più piccolo è l'errore e quindi si capisce l'importanza di un insieme ampio a

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 21 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 23 di 120

utilizzati per lo screening di malattie che sono sia gravi che trattabili, in modo che vi è un vantaggio per l'individuazione della malattia prima che i sintomi inizino.
Dovrebbero essere sensibili - cioè in grado di identificare correttamente quegli individui che hanno una data malattia. Molti test di routine effettuati come esami periodici della salute sono test di screening. Test del Colesterolo e Pap test per le donne sono alcuni esempi. I neonati vengono sottoposti a screening per una varietà di condizioni al momento della nascita.
Un test di screening positivo spesso richiede ulteriori test più specifici. Questo è importante per escludere correttamente quegli individui che non hanno la malattia o per confermare una diagnosi.
- **Sensitività** è la probabilità che un test è positivo, data la persona che ha una malattia (positivo su soggetti positivi); test positivo su persona che ha la malattia SENS= VP/(VP+FN)
- **Specificità** è la probabilità che un test è negativo, data la persona che non ha una malattia; SPEC=VN/(VN+FP)
- **Falso negativo**: una persona il cui test è negativo, ma che in realtà è positiva;
- **Falso positivo**: una persona il cui test è positivo, ma che in realtà è negativa.

Valore predittivo positivo (PV^+): è la probabilità che una persona ha la malattia dato il test positivo Pr[malattia / test +]; PV^+= VP/(VP+FP)
Valore predittivo negativo (PV^-): è la probabilità che una persona non ha la malattia dato il test negativo Pr[nessuna malattia / test-]. PV^-= VN/(VN+FN)
I valori predittivi lavorano sulle righe (sulle persone che fanno il test), mentre sensitività e specificità sulle colonne (sul totale delle persone che hanno la malattia).

test	P	N
P	VP	FP
N	FN	VN

CLASSIFICAZIONE DEI BAYES
Il problema relativo al test di screening è quello di classificare un soggetto in una delle due classi (presenza o assenza della patologia).
Da un punto di vista statistico questo problema può essere espresso come la **classificazione di un soggetto in una delle due popolazioni** w1 = la persona ha la malattia, w2 = la persona non HA la malattia. Consideriamo una misura X (cioè il marcatore PSA) per classificare la persona.
Persone diverse produrranno diversi valori di PSA e noi esprimiamo questa variabilità in termini probabilistici.
In x generale è un vettore di caratteristiche p e possiamo ipotizzare (ipotesi di partenza) che le funzioni di densità seguono una distribuzione normale (gausiana).

PROCEDURA EMPIRICA: abbiamo bisogno di due cose
1- variabile da discriminare tra due popolazioni
2- regole di decisioni.

Possiamo definire una variabile ausiliaria z che è una combinazione lineare delle p caratteristiche: $z=\alpha_1 x_1+\alpha_2 x_2+\ldots+\alpha_p x_p$. Ipotesi di partenza: popolazioni viste nel nuovo sistema z hanno:

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 23 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 25 di 120

Incrementando la conoscenza usata a priori per la costruzione del classificatore è possibile migliorare le prestazioni finali.

2. Se q_i è la probabilità che ha un elemento w di appartenere a W_i (con i=1,2), è possibile ottenere un **classificatore Bayes** con la seguente regola di classificazione:

$$z \geq \frac{\bar{z}_1 + \bar{z}_2}{2} + \ln\left(\frac{q_2}{q_1}\right) \Rightarrow w \in W1$$

$$z < \frac{\bar{z}_1 + \bar{z}_2}{2} + \ln\left(\frac{q_2}{q_1}\right) \Rightarrow w \in W2$$

è possibile provare che questa formulazione minimizza l'errore di misclassificazione:

$e=q_1P[2/1] + q_2P[1/2]$

3. Oltretutto, è possibile inserire nel calcolo del valore c il costo dovuto ad una classificazione sbagliata:
- C(2/1): costo di classificare in W_2 un elemento x appartenente a W_1,
- C(1/2): costo di classificare in W_1 un elemento x appartenente a W_2.

Dando i due costi, la regola di classificazione può essere:

$$z \geq \frac{\bar{z}_1 + \bar{z}_2}{2} + \ln\left(\frac{q_2 C(1/2)}{q_1 C(2/1)}\right) \Rightarrow w \in W1$$

$$z < \frac{\bar{z}_1 + \bar{z}_2}{2} + \ln\left(\frac{q_2 C(1/2)}{q_1 C(2/1)}\right) \Rightarrow w \in W2$$

La regola minimizza l'errore di misclassificazione: $e=q_1C(2/1)Pr[2/1] + q_2C(1/2)Pr[1/2]$

Parametri (dal punto di vista pratico)

Ipotesi: normale pooplazione multivariata con parametri sconosciuti

x=[x1,x2,…xp], vettore di p elementi; nel training set saprò per ogni elemento qual è la sua classe di appartenenza

W_1 : $N(\mu_1,\sum)$ e dimenzione n_1

W_2 : $N(\mu_2,\sum)$ e dimenzione n_2

μ_1, μ_2, $\sum$: sconosciuti perché sono derivati dalle popolazione (io ho solo un campione)

q_1, q_2, C(2/1), C(1/2) : conosciuti

I parametri μ_i sono sostituiti con la rispettiva stima: $\bar{x}_i = \frac{1}{n_i}\sum_{j=1}^{n_i} x_{ij}$

La matrice $\sum$ è sostituita con la stima della matrice di covarianza: $S = \frac{1}{n_1 + n_2 - 2}\left[(n_1 - 1)S_1 + (n_2 - 1)S_2\right]$

in questo modo, il minimo costo di misclassificazione è ottenuto solo asintoticamente.

$$n_1 \to \infty, n_2 \to \infty, \frac{n_1}{n_2} = costant$$

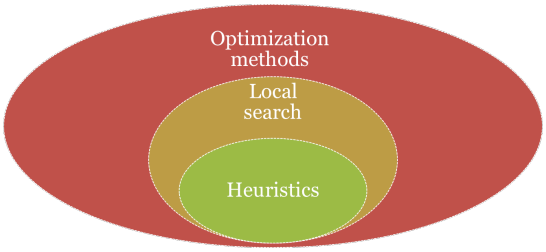
Procedura di classificazione

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 25 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 27 di 120

4 Ottimizzazione

Non esiste l'ottimizzazione in assoluto, dipende sempre dai criteri che si impongono. L'ottimizzazione è data da una coppia (S, f) dove S è l'insieme di possibili soluzioni e f è una funzione che mi permette di valutarle; f è quindi la funzione obiettivo cioè quella che voglio ottimizzare. La funzione obiettivo assegna a ogni s in S nello spazio delle soluzioni un numero reale che ne indica il valore. Si hanno tanti punti nello spazio e a ciascuno si assegna il valore che la funzione assume in quel punto; f permette quindi di confrontare elementi e di definire una relazione d'ordine totale tra ogni coppia di soluzioni in S, con o senza pari meriti. Un problema di ottimizzazione può essere di minimo o di massimo e c'è una relazione tra questi due problemi *min (f) = max (-f)*.
Un sottoinsieme di questo metodi è dato dai metodi basati sulla ricerca locale: dall'analisi della soluzione vicina a quella iniziale si cercano soluzioni migliori, ma siccome mi muovo in uno spazio limitato trovo solo un ottimo locale; sono dei metodi iterativi che generano sempre soluzioni diverse.
L'ottimo globale è il valore della funzione obiettivo che è il migliore in assoluto tra tutti i valori che la funzione può assumere: nel caso che la funzione non sia lineare può non essere unico. s* è ottimo globale se per ogni $s \in S$ $f(s) \leq f(s)$, se il problema in esame è un problema di minimo.
Complessità computazionale di un algoritmo: un algoritmo è formato da una sequenza di operazioni: si può quantificare in base al numero e al tipo di istruzioni la complessità dal punto di vista computazionale che mi dà un'idea del tempo che l'algoritmo dovrebbe impiegare.
Un ulteriore sottoinsieme è quello dei metodi euristici, nei quali la funzione potrebbe essere sconosciuta o non lineare, quindi non rappresentabile con una formula matematica chiusa e trovare l'ottimo globale è dispendioso in termini di tempo e risorse. Attraverso un algoritmo permettono di trovare una buona soluzione con un compromesso sull'uso delle risorse e del tempo.
Un algoritmo di ottimizzazione cerca la soluzione ottima in modo iterativo trasformando una soluzione in una sempre migliore. Ci sono anche sistemi che evolvono nel tempo e nei quali se non trovo una soluzione in tempi brevi il problema peggiora.



Metaeuristiche
Una metaeuristica è una procedura di livello superiore o euristico progettato per trovare, generare, o selezionare un algoritmo di ricerca che può fornire una soluzione sufficientemente buona ad un problema di ottimizzazione.
Qual è l'obiettivo di un metodo euristico? Vengono usati in molti problemi reali.

- Dimensione: uso metodi euristici quando le soluzioni potenzialmente usate sono molto numerose, come centinaia di soluzioni possibili, mentre se ne ho poche posso fare una ricerca a mano della soluzione migliore.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 27 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 29 di 120

- Iterativa o greedy: iterativo significa che parte da una soluzione che viene modificata nelle varie iterazioni, mentre greedy significa che mano a mano si costruisce la soluzione aggiungendo sempre nuove caratteristiche, e non si parte da nessuna soluzione iniziale.

Metaeuristiche basate sulla ricerca locale: cerca di trovare una soluzione migliore di quella corrente attraverso una ricerca nel vicinato della soluzione corrente. Con vicinato s'intende l'insieme delle soluzioni raggiungibili o costruibili dalla soluzione attuale attraverso un'operazione elementare, quindi soluzioni che distano $\pm\delta$ dalla soluzione iniziale $\bar{x}$. Se uso come codifica della soluzione una sequenza di elementi, il vicinato può essere ottenuto scambiando tra di loro tutti i possibili elementi. Il vantaggio di questo metodo è che posso costruire funzioni obiettivo complesse quante voglio.

Quello che si sta ottimizzando è un modello del problema reale e non c' è nessuna garanzia che la soluzione migliore del modello sia anche la soluzione migliore del problema reale. Sebbene non sia possibile costruire un modello che riproduca esattamente il sistema reale, i metodi euristici sono normalmente più flessibili e consentono l'utilizzo dei modelli che sono più vicini a quelli reali. Nel modello ci sono formalismi che impongono di tralasciare alcuni elementi. Tanto più il modello si allontana dal sistema, tanto più la soluzione del modello sarà diversa dalla soluzione reale e aumenta il rischio che la soluzione che è ottima per il modello, non sia ottima per il sistema.

La sequenza logica è:

sistema $\rightarrow$ modello$\rightarrow$applico algoritmo$\rightarrow$ soluzione ottima rispetto al modello, poi devo vedere come inserirla nel sistema.

Quindi il modello riduca la complessità del sistema, ma cosa posso tralasciare del sistema? Avendo la possibilità di usare formalismi diversi, posso scegliere il modello migliore. I passi di una ricerca locale generica sono:

1. Inizializzazione: scegliere una soluzione iniziale ammissibile x_1 come soluzione corrente e calcolare il valore della funzione obiettivo
2. Generazione del vicinato: seleziono un vicinato $V(x_i)$ e seleziono una soluzione $x_{i+1} \in V(x_i)$
3. Test di accettazione: verificare se la soluzione x_{i+1} può essere accettata ed eventualmente assumerla come soluzione corrente.
4. Test di terminazione: se il test è positivo la procedura è terminata.

Il metodo più semplice è il metodo del gradiente o steepest descent: calcolo il gradiente di una soluzione, parto da una soluzione iniziale della quale determino il vicinato e cerco in tale vicinato una soluzione migliore. Rigenero il vicinato di quest'altra soluzione migliore e così via… è un metodo deterministico quindi non c'è probabilità perché scelgo sempre la soluzione migliore. Lo svantaggio è che è strettamente dipendente dalla soluzione iniziale e dalle dimensioni del vicinato. Alle volte scelgo di partire da più di una soluzione iniziale per vedere se si trovano ottimi diversi.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 29 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 31 di 120

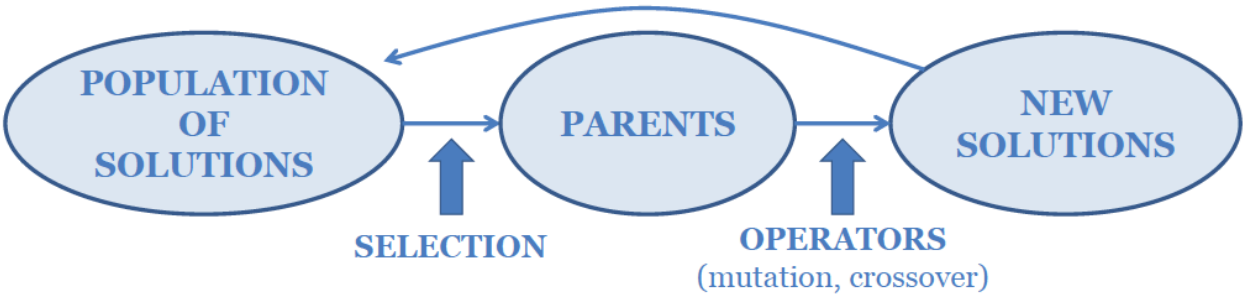
EC = Evolutionary Computing: si rifanno alla ricerca locale; partono da un insieme di soluzioni dalle quali ne generano di nuove usando algoritmi che mimano l'evoluzione. L'idea di base è che una popolazione di soluzioni possibili evolve tramite successive iterazioni da una variabile casuale e dopo si fanno delle selezioni. Le caratteristiche comuni sono che hanno fatto tutte uso del concetto di probabilità e quindi anche partendo dalla stessa famiglia iniziale posso trovare soluzioni finali diverse perché la funzione ha più di un ottimo o perché, andando avanti in modo probabilistico, in certi punti si possono prendere strade diverse: bisogna però impedire che la variabilità sia eccesiva.
Un algoritmo evoluzionistico (EA) è una ricerca stocastica di una soluzione ottima per un dato problema; gli elementi di base sono:

- Codificano la soluzione come cromosomi
- La funzione da ottimizzare è detta fitness e esprime l'adeguatezza alla sopravvivenza
- L' insieme iniziale di soluzioni
- Operazione
- Riproduzione cioè si usano operatori per generare nuove soluzioni.

Come soluzioni si ottiene un insieme di individui che si riproducono per rigenerare la popolazione.

Algoritmi evolutivi generici:
L'evoluzione tramite la selezione naturale di una popolazione di individui scelta a caso può essere pensata come una ricerca attraverso lo spazio dei possibili valori dei cromosomi. In questo senso, un algoritmo evolutivo (EA) è una ricerca stocastica di una soluzione ottimale per un determinato problema. Il processo di ricerca evolutivo è influenzato dai seguenti componenti principali di un EA:

- una codifica di soluzioni al problema come un cromosoma;
- una funzione per valutare l'idoneità, o la forza di sopravvivenza degli individui;
- inizializzazione della popolazione iniziale;
- gli operatori di selezione;
- e gli operatori di riproduzione.



Algoritmi genetici (GA): sono i primi algoritmi sviluppati che simulano il sistema genetico e dove le caratteristiche degli individui vengono espresse mediante genotipi; la base di tale algoritmi sta nella selezione a cui segue la ricombinazione mediante il crossover per modellizzare la riproduzione. Ci sono implementazioni diverse:

- Classico
- Varianti su ricostruzioni per il metodo di selezione

A seconda dei metodi le soluzioni sono rappresentate in modo diverso: negli algoritmi genetici è rappresentato da una stringa di valori binari 0 e 1; negli EC ogni individuo rappresenta una possibile soluzione per un problema di ottimizzazione: le caratteristiche di un individuo sono

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 31 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 33 di 120

Possibile soluzione: 01010011010010010001
gruppo di 4 bit = codice di esercizio
0101 | 0011 | 0100 | 1001 | 0001
Soluzione di decodifica: Il protocollo è fatto dall'esercizio numero 5, 3, 4, 9 e 1

Inoltre non tutte le soluzioni sono fattibili e quindi si devono imporre vincoli: bisogna destinare un numero di bit per ogni caratteristica e codificare il valore che la caratteristica può assumere dal numero che si ottiene passando dalla rappresentazione binaria a quella decimale. La prima cosa da fare è quella di creare una popolazione di partenza. Il modo standard per generare una popolazione è quello di assegnare un valore casuale del dominio a ogni gene dei cromosomi; si fa in modo casuale per far sì che la popolazione iniziale sia una rappresentazione uniforme di tutto lo spazio delle soluzioni, senza trascurare nessuna parte. Si crea generando n possibili soluzioni n_s dove n_s è la dimensione della popolazione, cioè il numero di soluzioni che compongono la soluzione e diventa significativo quando si fa il tuning dell'algoritmo. Se non ho vincoli genero n_s stringhe binarie, posso controllare di non aver elementi ripetuti nella popolazione, ma non è necessario.
La dimensione della popolazione resta costante nel corso delle iterazioni, anche se esistono varianti di algoritmi generici che permettono di modificare n_s, ma sono casi particolari. Non c'è nessuna relazione tra n_s e il numero di elementi nel training set. Scelgo n_s in base alla lunghezza della stringa cioè in base al numero di possibili combinazioni che posso avere: tanto maggiore è la stringa, tanto più lungo è n_s (in genere si usano multipli di 10).
Il nostro scopo è quello di esplorare lo spazio delle soluzioni e questo dipende sia da n_s che dal numero di iterazioni; c'è un rapporto tra popolazione e iterazione, cioè se la popolazione è piccola serve un numero di iterazioni maggiori, anche se non c'è proporzionalità. È molto importante trovare il giusto compromesso tra questi due valori: se li aumento tutti e due i costi computazionali diventano eccessivi.

- Un gran numero di individui aumenta la diversità, migliorando così le capacità di esplorazione della popolazione. Tuttavia, più sono gli individui, maggiore è la complessità computazionale per generazione. Mentre per quanto riguarda il tempo di esecuzione per l'aumento della generazione, può essere il caso che un minor numero generazioni sono necessarie per individuare una soluzione accettabile.
- Una piccola popolazione, invece rappresenterà una piccola parte dello spazio di ricerca. Mentre la complessità temporale per generazione è bassa, l'EA può avere bisogno di più generazioni a convergere che per una grande popolazione. Nel caso di una piccola popolazione, l'EA può essere costretto a esplorare più dello spazio di ricerca aumentando il tasso di mutazione.

La funzione di fitness rappresenta l'obiettivo, cioè quello che voglio ottimizzare; in genere è una funzione matematica ma può anche essere logica o un modello. In genere la funzione di fitness provvede ad una misura assoluta dell'obiettivo. La soluzione rappresentata da un cromosoma è direttamente stimolata usando la funzione obiettivo. Per alcune applicazioni non è possibile trovare una funzione di fitness assoluta; in questi casi si usa una misura di fitness relativa. Ci sono diversi tipi di ottimizzazione, la più facile è quella senza vincoli. Esistono diversi tipi di problemi di ottimizzazione, che hanno un'influenza diversa sulle varie fitness.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 33 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 35 di 120

- Selezione della nuova popolazione: una nuova popolazione di soluzioni candidate viene selezionata alla fine di ogni generazione utilizzandola come la popolazione della prossima generazione. La nuova popolazione può essere selezionata solo dalla progenie, o da entrambi i genitori e la prole. L'operatore di selezione dovrebbe garantire che i buoni individui sopravvivano a generazioni future. Inoltre bisogna evitare la possibilità che l'algoritmo si blocchi in un massimo locale.

Selective pressure o takeover time: è definito come il tempo nel quale si arriva ad avere una popolazione formata da un numero preponderante di individui uguali o come la velocità nella quale la soluzione migliore occupa l'intera popolazione mediante la ripetizione di operatori di selezione. Se è troppo alta mi blocco, perché tutte le soluzioni si addensano in un'unica zona.
Un operatore con un'alta pressione selettiva diminuisce la diversità nella popolazione più rapidamente rispetto ad altri operatori con una pressione selettiva inferiore, che può portare alla convergenza prematura subottimale di soluzioni. Un'elevata pressione selettiva limita le capacità di esplorazione della popolazione.

Sono state sviluppate molte operazioni di selezione:

- Random selection: è il più semplice; ogni individuo ha probabilità $1/n_s$ di essere scelto; selezionando in modo random le soluzioni e in questo modo ho la selective pressure più bassa che si possa avere. Non si hanno informazioni sulla fitness e quindi l'individuo peggiore e quello migliore hanno la stessa probabilità di essere estratti: per questo motivo difficilmente si bloccherà in un ottimo locale.
- Proportional selection, cioè in modo proporzionale alla fitness; si crea una distribuzione di probabilità proporzionale alla fitness; possono essere:
  - roulette wheel: gli elementi vengono normalizzati, cioè ognuno viene diviso per il suo massimo. La distribuzione di probabilità può essere vista come la ruota di una roulette, dove l'ampiezza di ogni fetta è proporzionale alla probabilità che un individuo ha di essere selezionato. Si seleziona in modo quasi random e tanto maggiore è la fitness tanto maggiore è la probabilità di essere selezionati.
  - Stochastic universal sampling
- Tournamete selection: seleziono in modo casuale un gruppo di n_t individui con $n_t < n_s$ e se n_t non è troppo grande questa tecnica impedisce agli individui migliori di dominare; se invece è troppo piccolo aumenta la probabilità che individui cattivi vengano scelti. Riduce la selective pressure rispetto alle tecniche proporzionali alla fitness.
- Rank based selection: usa il grado nel valore delle fitness per determinare la probabilità di selezione, e non il valore assoluto della fitness. Siccome si basa solo sull'ordine e non sul valore della fitness gli individui migliori non riescono a dominare.
- Elitism: opposto al random. Salva sempre i migliori per portarli alla generazione successiva; è più veloce nel dare la soluzione ma non è detto che sia la migliore in assoluto. Gli individui migliori non vengono modificati dalla mutazione.
- Hall of fame: è come la classifica di un gioco, cioè per ogni generazione gli individui migliori vengono scelti e riportati nella hall of fame; viene usato come base per il crossover e alla fine si tengono solo i suoi elementi.

Posso anche usare due algoritmi diversi per le due diverse soluzioni che faccio.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 35 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 37 di 120

1. Fitness: codifica la soluzione e stima il numero di soluzioni. Il primo passo è quello di creare una popolazione iniziale: per farlo devo avere una funzione e sapere da quanti individui è formata la popolazione nind: devo stabilire un valore iniziale di nind che sia collegato al numero di iterazioni
2. Numero di individui nella popolazione: è collegato al numero di iterazioni e può essere fisso o variabile; viene testato sperimentalmente. Devo definire nind ed iter che sono legati tra loro e con $2^k - 1$
3. Genitori: devo definire il loro numero e le loro regole di selezione. Non può essere un numero troppo piccolo e deve essere circa l'80-90% del numero di individui, è il numero di soluzioni che viene combinato con gli operatori genetici per creare nuove soluzioni
4. Verifica sperimentale: definire le implementazioni e gli operatori genetici. Quante soluzioni vengono usate per il crossover? Tutte? Vengono scelte in modo casuale? Ci sono sia la possibilità di crossover (che in genere vale circa 1) che la probabilità di mutazione (che è più bassa), non è detto che restino costanti nel corso delle iterazioni e per entrambe devo dire come vengono implementate. Posso aumentare la probabilità di mutazione per esplorare maggiormente lo spazio delle soluzioni. Il numero dei figli può essere maggiore, minore o uguale a quello dei genitori
5. Ricostruzione della popolazione e stabilità della soluzione; altre regole di selezione. Se non ho finito le iterazioni devo ricominciare.

In questo modo ho implementato l'algoritmo genetico. Se eseguo l'algoritmo una sola volta non ho informazioni sulla variabilità che la soluzione può avere perché è legata a fattori probabilistici: devo fare diverse ripetizioni e confrontare le soluzioni migliori che ottengo nei vari casi. La procedura da seguire è quindi:
creo la popolazione iniziale → seleziono i genitori → applico gli operatori → ricostruisco la popolazione → ho finito le iterazioni?
Se si, ok, altrimenti ricomincio.
Ora il problema è capire quanto è stabile la soluzione che ho ottenuto: lo scopo è quello di ottenere sempre soluzioni molto simili, uguale è praticamente impossibile. Nrip indica il numero di volte che devo ripetere tutto partendo dalla stessa popolazione iniziale, un numero ragionevole è compreso tra 10 e 20.
Per ogni ripetizione l'algoritmo usato nel corso del laboratorio 2 dà un grafico dove viene riportato il valore medio, quello massimo, quello minimo e quello migliore e se vedo che dopo un po' si stabilizza, posso diminuire il numero di iterazioni. Fornisce anche un bar diagram che mi dice quante volte nelle singole soluzioni è compresa una certa variabile: la situazione ideale è quella in cui certe variabili valgono n e certe 0, se n è il numero di ripetizioni, è questo vorrebbe dire che si ottiene sempre la stessa soluzione. Posso creare un indice di similarità delle soluzioni, ad esempio confrontando due soluzioni per vedere se alcune variabili ci sono sempre o se sono complementari.

<u>Tabu search</u>: è un metodo basato su un'unica soluzione deterministico, cioè in cui si possono aggiungere delle componenti probabilistiche. Il concetto alla base è quello del tabu cioè si genera un vicinato, si seleziona una soluzione e si crea una lista in modo tale che non si possa più tornare sulla stessa soluzione: si crea cioè un tabu list che ha sempre lo stesso numero di elementi. Nella lista vengono anche inserite delle operazioni che possono essere usate solo se portano ad un vantaggio: aspiration criteria. In pratica una memoria costringe ad esplorare varie zone dello spazio

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 37 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 39 di 120

1. costruzione della soluzione;
2. aggiornamento del ferormone, applicato in due fasi:
   - Uno di evaporazione: mutazione negli algoritmi genetici per non bloccarsi su un minimo locale – le formiche quando non sentono più il feromone cambiano percorso trovando un'altra fonte di cibo;
   - Uno di rinforzo: maggiore è la quantità di cibo, più feromone le formiche rilasciano.

1. Costruzione della soluzione: La costruzione delle soluzioni avviene secondo una regola probabilistica. Le formiche artificiali possono essere considerate come procedure greedy stocastiche che costruiscono una soluzione in modo probabilistico aggiungendo componenti della soluzione a quelli parziali fino a derivare una soluzione completa. Di solito questo processo prende in considerazione:
- Percorsi feromoni: memorizza la caratteristica di soluzioni generate "buone", che guideranno la costruzione di nuove soluzioni da formiche. Esse rappresentano la memoria di tutto il processo di ricerca delle formiche;
- Problema-dipendente da informazioni euristiche: esprimere l'opportunità di passare da un nodo all'altro

2. Aggiornamento del ferormone: L'aggiornamento del feromone viene effettuato utilizzando le soluzioni generate. Una regola di aggiornamento del feromone globale viene applicata in due fasi:
- Fase di evaporazione: Durante la fase di evaporazione feromone diminuisce automaticamente nel percorso. Ogni valore del feromone sul collegamento tra il nodo i con il nodo con la j è ridotto di una percentuale fissa: $\tau_{ij} = (1 - \rho)\, \tau_{ij}$, dove $\rho \in]\, 0,1]$ rappresenta il tasso di riduzione del feromone. L'obiettivo dell'evaporazione è di evitare per tutte le formiche una convergenza prematura verso soluzioni "buone" e quindi per incoraggiare la diversificazione nello spazio di ricerca (esplorazione).
- Fase di rinforzo: Durante la fase di rinforzo il sentiero del feromone viene aggiornato in base alle soluzioni generate.
Esistono vari metodi di rinforzo:
   - Online step by step pheromone update τ_{ij} viene aumentato da ogni formica ad ogni passo della costruzione della soluzione
   - Online delayed pheromone update: il feromone viene aumentato di una quantità τ ogni volta che una formica genera una nuova soluzione; in alcuni casi l'incremento di feromone è proporzionale alla quantità di soluzione trovata
   - Offline pheromone update: si aumenta il feromone quando tutte le formiche generano una soluzione completa; è l'approccio più usato:
     - aggiornamento del feromone basato sulla qualità: questa strategia aggiorna il valore feromone associato alla migliore soluzione trovata tra tutte le formiche (o le migliori soluzioni k). I valori aggiunti dipendono dalla qualità delle soluzioni scelte.
     - aggiornamento feromone Rank-based: le migliori soluzioni k delle formiche permettono di aggiornare il feromone a seconda della classificazione delle soluzioni.
     - Peggiore aggiornamento del feromone: la formica che genera la peggiore soluzione diminuisce i sentieri dei feromoni relativi ai componenti della soluzione

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 39 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 41 di 120

Posso individuare le prestazioni ad esempio applicando un metodo e vedendo che risultati fornisce. L'importanza della scelta del training set, e quindi devo ragionare sui dati su quali ho, su quali sono utili, su quali potrei facilmente avere. Si deve tener conto dell'errore di misura che si commette nel quantificare una grandezza, vedere quanto quest'errore è significativo e che peso da. Posso o diminuire la probabilità che ci sia o diminuire le conseguenze della sua presenza. Devo valutare le conseguenze dell'incertezza o con un metodo per gestirle o cambiando metodo di misura per ridurle; posso anche al limite decidere che l'incertezza non è importante per i dati che ho. Ci possono essere altre fonti di incertezza nella definizione di alcuni concetti o nella definizione di alcune soglie.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 41 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 43 di 120

- Contruction: dalle caratteristiche ottenute ne costruisco di nuove che non hanno più un legame diretto con i parametri di partenza; ottengo dei componenti che non so come sono legate alle caratteristiche di partenza
- Selection: mantengo un legame con le caratteristiche di partenza, selezionandone un sottoinsieme e rimuovendo quelle irrilevanti o ridondanti; più utile dal punto di vista della spiegazione.

È preferibile la seconda tecnica che, mantenendo un legame con le caratteristiche iniziali, rende più facile la contestualizzazione e la spiegazione dei dati.

Feature Construction: genera una serie completamente nuova di caratteristiche da quelle originali. Questi tipi di procedure, come Principal Component Analysis (PCA) o Linear Analysis Discriminant (LDA), si traducono in una lettura più difficile perché le nuove caratteristiche non corrispondono a quelle originali. Inoltre, poiché i nuovi attributi sono ottenuti come combinazione lineare o non lineare di tutte quelle iniziali, tutte le caratteristiche sono necessarie per essere sempre raccolte al fine di generare il nuovo insieme di variabili.

Feature Selection (FS): sono algoritmi che selezionano semplicemente un numero minimo di elementi *rilevanti* e *informativi* dalla prima serie di variabili in modo che:
- La quantità di informazioni rispetto alle variabili originali è mantenuto intatto
- Il significato delle caratteristiche è preservato. Questa caratteristica facilita la fase dell'interpretazione conoscenza.

Una caratteristica è detta rilevante se è essenziale per ottenere buone prestazioni del sistema; in caso contrario, è irrilevante.
Una caratteristica informativa è quella che è altamente correlata con il concetto di decisione (s), ma è altamente scorrelata con altre caratteristiche.
Possiamo dire che una caratteristica è:

- Ridondante quando ce n'è un'altra che da la stessa informazione (se ce ne sono 2 correlate basta tenerne una)
- Irrilevante se non aggiunge nessuna informazione per far si che la classificazione sia più corretta (ottengo lo stesso risultato finale, che ci sia o no)

Di solito le *feature rilevanti* possono essere forti o deboli. Se una feature è *fortemente rilevante*, ciò implica che non può essere rimossa dal set di dati, senza una conseguente perdita di accuratezza predittiva. Se è *debolmente rilevante*, la funzione può talvolta contribuire all'accuratezza, anche se questo dipende da quale altre caratteristiche sono considerate.
È del tutto possibile per due caratteristiche il fatto di essere inutili singolarmente, e tuttavia essere altamente predittive se prese insieme. Nella terminologia della FS, possono essere sia ridondanti e irrilevanti per conto loro, ma la loro combinazione fornisce preziose informazioni.

Alcune caratteristiche possono essere fonte di rumore o peggiorare l'analisi. Osserviamo che anche se 2 caratteristiche non sono correlate non vuol dire che non ci sia relazione, ad esempio mi possono servire più parametri che in combinazione tra loro mi permettono di dare la risposta finale.
Gli obiettivi della selezione dei parametri sono:

- Migliorare le performance

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 43 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 45 di 120

- Forwards: scelgo le caratteristiche che correlano di più e ne aggiungo fino a quando le prestazioni non migliorano in modo sensibile
- Randomised: costruisco dei sottoinsiemi di caratteristiche non in modo del tutto casuale per non perdere troppo tempo.

2) Misura della qualità della feature:
distanza (se separo elementi di classi diverse); informazione (prendo la variabile che dà il maggior contenuto d'informazione rispetto al problema); dipendenza o correlazione (seleziono le variabili meno dipendenti dalle altre, tolgo quelle correlate); consistenza (elimino le variabili che producono inconsistenza, Inconsistenza= ho una variabile che su due elementi differenti assume lo stesso valore ma hanno classi diverse); valutazione dell'errore di classificazione (elimino le variabili che non mi permettono di classificare gli elementi).
Definiamo training set l'insieme di elementi, ognuno con la sua classe corrispondente; ogni elemento è stato già classificato cioè appartiene ad una classe.
Dobbiamo avere un metodo per valutare il risultato della nostra selezione, confrontando il risultato che si ottiene quando si usano tutte le caratteristiche e quello che si ottiene quando se ne usa solo un sottoinsieme: non è sempre vero che il risultato che ottengo con tutte le caratteristiche sia il migliore in assoluto perché ci possono essere caratteristiche ridondanti o altre che creano rumore; comunque si prede quasi sempre come riferimento la situazione con tute le caratteristiche. La valutazione della feature selection spesso comporta due compiti:

- Si confrontano due situazioni: prima e dopo la feature selection
- Si confrontano due algoritmi di feature selection per vedere se uno è migliore dell'altro per un determinato compito.

Un altro aspetto importante è il bias che si può creare. Ho un insieme detto training set dal quale seleziono le caratteristiche e un altro insieme detto test set, che uso per addestrare il classificatore: se uso lo stesso insieme per tutti e due i compiti, le prestazioni del classificatore possono peggiorare per la correlazione tra i dati usati per addestrare e quelli usati poi per classificare. Il training set deve essere rappresentativo quindi:

- Le classi devono essere equamente rappresentate
- Gli elementi dei due insiemi devono essere scelti in modo casuale in modo da non influenzare i risultati dell'algoritmo.

Negli ultimi anni si è discusso molto sui metodi di feature selection che possono derivare dalla statistica, dall'ottimizzazione, dalla logica fuzzy...etc.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 45 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 47 di 120

L'approccio comune è quello di procedere in modo iterativo:
1. Si trova una ragionevole partizione iniziale e poi
2. Si spostano i campioni da un cluster all'altro in modo da ridurre la funzione criterio.

In altre parole, si inizia provando a suddividere i miei elementi; poi una volta suddivisi vado ad analizzare il risultato di tale suddivisione −> avrò dei prototipi per ciascuna classe −> vado a riclassificare in base a quei prototipi finché non ottengo la convergenza, cioè che le classi non variano più molto.

Questi metodi iterativi producono soluzioni sub-ottimali ma sono implementabili al calcolatore. Non c'è la certezza di trovare la soluzione in assoluto migliore perché per farlo dovrei passare in rassegna la suddivisione di tutti i possibili cluster: si tratta di un approccio euristico, cioè si sa che non si può trovare la soluzione migliore. I principali metodi iterativi possono essere divisi in due gruppi:

Algoritmi di **clustering piatti (o partizionali)**:
- Questi algoritmi producono una serie di cluster disgiunti;
- Due algoritmi sono ampiamente utilizzati:
  - K-means
  - ISODATA

Algoritmi di **clustering gerarchici**:
- Il risultato è una gerarchia di cluster nidificati;
- Questi algoritmi possono essere suddivisi in approcci agglomerativi e di divisione.

Clustering piatti

L'algoritmo di clustering più usato è il **<u>k-means</u>**.

L'algoritmo k−means è una procedura di clustering semplice che tenta di minimizzare la variabilità intra-cluster e massimizzare la distanza inter-cluster in modo iterativo.
- Centroide: è il valore medio di tutti gli elementi appartenenti al cluster;
- Variabilità intra-cluster: somma di tutte le distanze tra ogni elemento appartenente al cluster ed il centroide;
- Distanza inter-cluster: distanza tra due centroidi del cluster

Procedura:

1. Definire il numero di cluster;
2. Inizializzare cluster da:
   - un'assegnazione arbitraria di esempi a cluster oppure
   - un insieme arbitrario di cluster centrali (alcuni esempi utilizzati come centri). Inizializzazione di cluster o attraverso esempi che estraggo e associo casualmente o definendo in modo scollegato dei dati adisposizione dei valori da assegnare alle variabili (in pratica un prototipo è un vettore di valori numerici per ogni elemento: o estraggo tali valori casualmente dal data set o li definisco in modo del tutto casuale: in genere si preferisce estrarli dal data set perché è più rapido);
3. Calcolare la media campionaria di ogni cluster. Si aggiorna il prototipo mediante il valore medio degli elementi che ho assegnato a ogni cluster.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 47 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 49 di 120

Di solito si mette anche una condizione di stop legata al numero di iterazioni, perché se è un elemento al limite tra due cluster si corre il rischio di dover andare avanti all'infinito, perché a ogni iterazione si sposta di cluster. Come condizione di stop posso anche avere una condizione meno rigida, cioè la riassegnazione di un solo elemento non modifica in modo significativo i cluster.
Questo tipo di algoritmo minimizza l'errore quadratico medio MSE.
Problema: con che numero di cluster si inizia il metodo? Dipende dal problema in esame
Esempio: problema di data mining: ho le cartelle mediche di 1200 pazienti EHR e per ogni paziente ho diverse variabili (età, patologie pregresse, …). Ci sono delle caratteristiche comuni a tali parametri? Posso raggrupparli in base alle loro caratteristiche comuni creando dei sottogruppi, cioè dei cluster creati considerando la distanza e la varianza. Se la distanza è troppo piccola aumenta il numero di cluster.
Ovviamente, affinché il procedimento abbia senso, il numero di cluster deve essere piccolo rispetto al numero dei dati sui quali sto lavorando. Posso che porre un limite al numero di elementi che ogni cluster deve avere e se ne ha di meno elimino il cluster. Oppure posso calcolare la distanza tra i cluster come distanza tra prototipi e unire quelli che hanno una distanza piccola, quindi eliminare un cluster. Infine posso porre un limite sul numero minimo e massimo di cluster che si possono avere.
N.B.: non c'è nessuna ipotesi sull'uniformità di elementi contenuti in ogni cluster, quindi posso avere cluster con pochi elementi e cluster con molti elementi.

Il metodo k-means ha un problema legato alla necessità di definire a priori un numero di cluster, questo normalmente non porta alla definizione del numero ottimale di cluster.

Per ovviare al problema si può:

- riclassificare i campioni partendo da un numero diverso di cluster
- post-elaborare il risultato andando a unire cluster che risultino simili
- usare l'algoritmo ISODATA

La parte di modifica dei cluster sulla base dei tre indicatori sopra menzionati può essere automatizzata con **isodata** (iterative self organizing data analysis technique algorithm), che lavora utilizzando il k-means (è un'estensione dell'algoritmo k−means con alcune euristiche per selezionare automaticamente il numero di cluster):

- seleziono un numero iniziale di cluster n_c e assegno esempi a ogni cluster
- uso il k-means e calcolo la distanza
- se la varianza all'interno dei cluster è maggiore di una certa soglia, aumento di 1 il numero dei cluster
- se la distanza tra i vari cluster è minore di una certa soglia, diminuisco di 1 il numero dei cluster
- se non ho motivi per variare il numero dei cluster, ho finito.

Si definisce procedimento euristico, un metodo di approccio alla soluzione dei problemi che non segue un chiaro percorso, ma che si affida all'intuito e allo stato temporaneo delle circostanze, al fine di generare nuova conoscenza. È opposto al procedimento algoritmico.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 49 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 51 di 120

Procedura completa:

a) Seleziona un numero iniziale di clusters N_C e usa i primi N_C elementi come clusters centrali $\mu_k, k = 1, \dots, N_C$

b) Assegna ogni elemento al cluster più vicino
   i. Esce dall'algoritmo se la classificazione di ogni elemento non è cambiata

c) Elimina i clusters che contengono meno di N_{MIN_EX} elementi e
   i. Assegna i loro elementi ai clusters restanti in base sulla minima distanza
   ii. Diminuisce coerentemente N_C

d) Per ciascun cluster *k*,
   i. Calcola il centro μ_k come la media campionaria di tutti gli esempi assegnati a quel cluster
   ii. Calcola la distanza media tra gli elementi e i cluster centrali:
   $d_{AWG} = \frac{1}{N}\sum_{k=1}^{N_C} N_k d_k$ e $d_k = \frac{1}{N_k}\sum_{x \in \omega_C} |x - \mu_k|$
   iii. Calcola la varianza di ciascun asse e trova l'asse n* con la varianza massima $\sigma_k^2(n^*)$

e) Per ciascun cluster *k* con $\sigma_k^2(n^*) > \sigma_S^2$, se $\{d_k > d_{AWG}$ *and* $N_k > 2N_{MIN\_EX} + 1\}$ *or* $\{N_C < \frac{N_D}{2}\}$
   i. Divide quel cluster in due clusters dove i due centri μ_{k1} e μ_{k2} differiscono solo per la coordinata n*
      1. $\mu_{k1}(n^*) = \mu_k(n^*) + \varepsilon\sigma_k(n^*)$ -> tutte le altre coordinate rimangono le stesse, 0<ε<1
      2. $\mu_{k2}(n^*) = \mu_k(n^*) - \varepsilon\sigma_k(n^*)$ -> tutte le altre coordinate rimangono le stesse, 0<ε<1
   ii. Aumenta coerentemente N_C.
   iii. Riassegna l'elemento del cluster ad uno dei due nuovi clusters in base alla minima distanza tra i centri dei cluster

f) Se $N_C > 2N_D$ allora
   i. Calcola tutte le distanze $D_{ij} = d(\mu_i, \mu_j)$
   ii. Ordina D_{ij} in ordine decrescente
   iii. Per ciascuna coppia di cluster ordinata da D_{ij}, se (1) nessun cluster è già stato unito, (2) la distanza D_{ij} soddisfa $D_{ij} < D_{MERGE}$ e (3) nessun altra coppia di cluster oltre a N_{MERGE} è già stata unita in questo ciclo, allora
      1. Unisce i cluster i^{th} con j^{th}
      2. Calcola il cluster centrale $\mu' = \frac{N_i\mu_i + N_j\mu_j}{N_i + N_j}$
      3. Diminuisce coerentemente N_C

g) Torna al punto a)

Cluster in modo gerarchico

<u>Dendogramma</u>: l'obiettivo finale è quello di unire i vari cluster fino ad ottenere solo uno; di volta in volta diminuisco i cluster che hanno distanza minore. Bisogna anche trovare un punto di taglio, cioè sapere dove posso tagliare il dendogramma. Per prendere tale decisione si può usare la variabilità dei cluster: in genere si taglia dove la distanza tra il passo n-1 e il passo n è decisamente maggiore delle altre. Come prima cosa devo ordinare gli elementi che mi vengono forniti e dopo calcolare la distanza di ogni elemento con il proprio vicino (visto che le altre distanze saranno di sicuro maggiori); se due coppie di valori hanno la stessa distanza ne scelgo una e molto probabilmente al passo successivo sceglierò l'altra.Consideriamo una sequenza di suddivisioni di N campioni in *c* cluster.

La prima partizione è costituita da *n* cluster (ogni cluster contenente esattamente un campione).

I successivi due cluster sono uniti (−> n−1 cluster), ..., fino all' n−esimo step in cui tutti i campioni formano un cluster. Dati i due campioni x e x' ad un certo livello saranno raggruppati nello stesso cluster. Se la sequenza ha la proprietà che, quando due campioni sono nello stesso cluster al livello *k* questi restano insieme a tutti i livelli più elevati, allora la sequenza è detta raggruppamento gerarchico. La rappresentazione più naturale di clustering gerarchico è un albero

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 51 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 53 di 120

7 Logica Fuzzy

Quando ha introdotto il Fuzzy Sets nel 1964 Zadeh è stato motivato dalla mancanza di un metodo matematico che potrebbe far fronte alla complessità di "animare", sia sistemi biologici che umani.
I sistemi biologici sono generalmente di ordini di grandezza più complessi rispetto ai sistemi artificiali.
L'ambizione della teoria degli insiemi fuzzy è quella di fornire un ambiente formale per informazioni incomplete e graduali, come espresso dalle persone nel linguaggio naturale. Qual è il pensiero fuzzy? Gli esperti in genere si basano sia sulla loro esperienza che sul buon senso per risolvere i problemi.
Essi possono anche utilizzare termini vaghi e ambigui che sono comprensibili solo attraverso la conoscenza del contesto.
Il problema è: come possiamo rappresentare la conoscenza di esperti che utilizzano termini vaghi e ambigui in un computer? Le radici di Fuzzy si trovano nella letteratura filosofica nella prima metà del ventesimo secolo (Charles Pierce, Bertrand Russell, ...).
Anni ’30. Jan Lukasiewicz, un logico e filosofo polacco, ha introdotto una logica che ha esteso la gamma di valori di verità a tutti i numeri reali nell'intervallo tra 0 e 1. Ha usato un numero in questo intervallo per rappresentare la possibilità che una dichiarazione rilasciata sia vera o falsa. Questo lavoro ha portato ad una tecnica di ragionamento inesatta spesso chiamata teoria della possibilità.
1937. Max Black, un filosofo, pubblico un testo intitolato "Vaghezza: un esercizio di analisi logica". Nell’appendice egli definisce il primo semplice insieme fuzzy e illustra l'idea di base di operazioni sugli insiemi fuzzy.
Anni ’60. Lofti Zadeh: “il più delle volte, le classi di oggetti che si incontrano nel mondo fisico reale non hanno definito con precisione i criteri di appartenenza ". Nel 1965 ha pubblicato il suo testo “Fuzzy set”. Zadeh ha esteso il lavoro sulla teoria della possibilità in un sistema formale di logica matematica e ha introdotto un nuovo concetto per l'applicazione di termini del linguaggio naturale. Questa nuova logica per la rappresentazione e la manipolazione di termini confusi è stata chiamata logica fuzzy, e Zadeh divenne il maestro della logica fuzzy.
1977. La dottrina del Fuzzy-ismo è stata correttamente definita da M.M. Gupta come "un insieme di concetti e tecniche volti a fornire un quadro sistematico per affrontare con la vaghezza e l'imprecisione inerente, processi del pensiero umano".
La logica fuzzy non è logica sfocata, ma è la logica che viene utilizzata per descrivere la confusione.
La logica fuzzy si basa sull'idea che tutte le cose ammettono gradi per definire differenti set. Temperatura, altezza, velocità, distanza, bellezza sono tutte dotate di una scala mobile. Come una scala mobile è spesso impossibile distinguere i membri di una classe dai non membri della stessa. Quando una collina diventa una montagna? La logica booleana o convenzionale utilizza distinzioni nette. Ci costringe a disegnare le linee tra i membri e i non membri di una classe. Diciamo che Tom è alto perché la sua altezza è di 181 centimetri. Se abbiamo tracciato una linea a 180 cm, troveremo invece che David, che è di 179 cm, è basso, che è contro il senso comune. La logica fuzzy permette di evitare tali assurdità.
La logica fuzzy riflette come la gente pensa. Tenta di modellare il senso delle nostre parole, il nostro processo decisionale e il nostro senso comune. Come risultato, porta a sistemi intelligenti, nuovi e più umani. L’incertezza può avere differenti origini. Il tipo più comune di incertezza è legato ai DATI: errori di misura, giudizi relativi alle situazioni di bordo. Un secondo tipo di

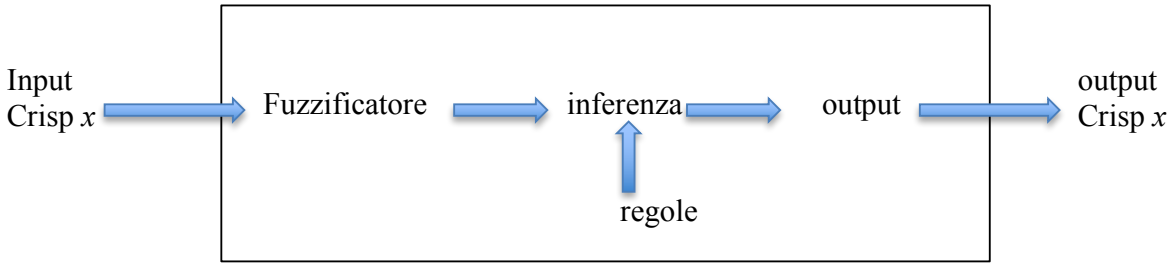
© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 53 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 55 di 120

Si può usare la logica fuzzy per costruire un classificatore supponendo di avere informazioni sufficienti. Differenza tra:

- Logica crisp o puntuale: 0 e 1, cioè vero o falso, l'elemento appartiene o no ad una classe
- Logica fuzzy: supera tale distinzione netta associando ad un elemento un grado di appartenenza con un valore compreso tra 0 e 1 dove 1 significa appartenenza certa e 0 significa non appartenenza.

Un insieme fuzzy è quindi un continuo di gradi di appartenenza. Il cuore di un classificatore fuzzy è un membership function, che assume valori tra 0 e 1.
Il classificatore riceve dall'ambiente esterno input che sono crisp; in seguito è necessario fuzzificare tali valori, cioè passare da una logica all'altra.



Tali input modificati vengono usati dall'algoritmo (inference) per capire quali regole tra quelle presenti sono attive e quali no: è un classico sistema basato su regole. Alle volte si trasforma poi l'output in crisp. Ci sono 6 fasi tramite le quali costruisco il classificatore:

1. Costruisco le membership function: passo da input crisp a input fuzzy
2. Scelgo e costruisco le regole fuzzy
3. Definisco le modalità di aggregazione: inferenza; l'aggregazione è il processo con cui si unisce l'output di tutte le regole dopo aver fatto clipping o scaling
4. Definisco il metodo per defuzzificare i risultati; il più usato è il metodo della centroide, cioè individua il punto dove una retta verticale taglia l'insieme in due parti con la stessa massa
5. Valuto il sistema
6. Validazione

In logiche nitide, come la logica binaria, le variabili sono vere o false, 1 o 0, nero o bianco.
Un'estensione alla logica binaria è la logica multivalore, in cui le variabili possono avere molti valori definiti.
Nel 1965 Zadeh ha dato la seguente definizione: "Un insieme fuzzy è una classe con gradi di appartenenza continui". Così un Fuzzy set (classe) F appartenente ad U è caratterizzato da una funzione di appartenenza che associa ad ogni elemento u in U, un numero reale nell'intervallo [0,1].
Il valore della funzione di appartenenza $\{\mu F(u)\}$ all'elemento u, rappresenta il grado di appartenenza di u a F. Un insieme fuzzy F è quindi definito come una mappatura:
$F : U \rightarrow [0,1]$
$F(u) = 0$ $\{\mu_F(u)=0\}$: NON appartiene assolutamente
$F(u) = 1$ $\{\mu_F(u)=1\}$: piena appartenenza
Con grado di verità o valore di appartenenza si intende quanto è vera una proprietà: questa può essere, oltre che vera (= a valore 1) o falsa (= a valore 0) come nella logica classica, anche pari a valori intermedi.
Si può ad esempio dire che:

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 55 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 57 di 120

(categoria) che appartiene all'insieme con grado massimo. Gli altri appartengono al gruppo con un grado che misura la loro distanza relativa dall'esempio perfetto.

METODI DI ELICITAZIONE (sign: deriva dalla psicologia "tirar fuori" -> estrarre)
Ci sono sei metodi principali utilizzati a fini sperimentali, con l'obiettivo di costruire funzioni di appartenenza:
1. Polling (Interrogazione): sei d'accordo che John è alto? (SI/NO) Nel polling si evidenzia il punto di vista sfuocato (il "fuzziness") che sorge dal disaccordo interpersonale. La domanda: "Sei d'accordo che A è F?" è posta a diversi individui. Vengono interrogate le risposte e viene presa la media per costruire la membership function. Il Polling è anche uno dei modi naturali di elicitare funzioni di appartenenza per la Similarity view.
2. Pairwise comparison (Confronto a Coppie): quale colore, A o B, è più scuro (e di quanto?)
3. Membership function exemplification (Semplificazione delle MF): qual è il grado di appartenenza del colore A all'insieme fuzzy dei colori scuri? Qual è il grado di appartenenza di John all'insieme delle persone alte? In generale, "Con quale grado A è F?" In questo caso l'elicitazione è eseguita dal significato degli esempi.
4. Interval estimation (Stima dell'intervallo): fornire un intervallo in cui si pensa si trovi l'altezza di John. Individua la Random set view delle MF. Al soggetto è chiesto di dare un intervallo che descrive la Fitness di A. Questo metodo è più appropriato in situazioni dove vi è un ordinamento chiaro e lineare nelle misurazioni del concetto fuzzy, come nell'altezza, nel calore, nel tempo, etc.…
5. Direct rating (Stima diretta – Stima puntuale): classificare il colore A secondo quanto è scuro, classificare John secondo la sua altezza. In generale la domanda è: "Come F è A?" Il Direct rating sembra il modo più semplice per trovare una membership function. Questo approccio individua il punto di vista fuzzy che nasce dalla vaghezza soggettiva dell'individuo. Al soggetto (di solito un esperto) è richiesto di classificare A rispetto a F ripetutamente nel tempo. L'esperimento deve essere attentamente progettato in modo tale da rendere difficile al soggetto di ricordarsi le risposte precedenti. Un altro modo consiste nel porre le domande a numerosi soggetti e raggruppare le loro risposte.
6. Reverse rating (Stima inversa): identificare la persona che è alta con grado 0.6. In generale "Identificare A che è F con grado $\mu_F(a)$". In questo metodo, al soggetto è dato un grado di appartenenza e quindi chiesto di identificare l'oggetto per cui tale grado corrisponde al termine fuzzy della domanda. Questo metodo può essere usato per tutti gli individui ripetendo la stessa domanda per la stessa membership function nonché per un gruppo di individui. Una volta che la risposta del soggetto (o dei soggetti) sono state registrate, le distribuzioni condizionate possono essere adottate per essere distribuite normalmente e i parametri sconosciuti (media e
varianza) possono essere stimati come al solito. Questo metodo richiede valutazioni anche si scale di intervalli. La Revers rating può usata per verificare le MF ottenute mediante altri metodi.

Dipendenza della MF dal contesto.
L MF collegata ad un termine fuzzy (ad es: YOUNG) dipende dal contesto in cui si usa quel termine: l'età di un giovane pensionato è chiaramente differente da quella di un giovane studente.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 57 di 120

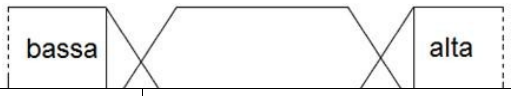
© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 59 di 120

toolbox di matlab obbliga ad usare sempre and o sempre or. Posso anche avere più conseguenze, anche se in genere si hanno più antecedenti e una sola conseguenza. Le regole vengono analizzate una ad una, tramite gli antecedenti e il risultato dell'analisi degli antecedenti viene riversato sulla conseguenza. Una regola fuzzy può essere definita come una condizione di stato nella forma:

IF *x* is *A*
THEN *y* is B

Dove *x* e *y* sono variabili linguistiche e A e B sono i valori linguistici determinati dall'insieme fuzzy dei discorsi universali X e Y, rispettivamente.

Differenze tra regole classiche e fuzzy:

Regola classica (crisp)		Regola fuzzy	
			
IF THE	velocità è > 100 arresto distanza è	IF THE	velocità è alta arresto distanza è lunga
IF THE	velocità è < 40 arresto distanza è	IF THE	velocità è bassa arresto distanza è corta

La regola fuzzy si attiva anche quando la velocità è ancora minore di 100, mentre per la regola crisp deve averla superata!

Come ragionare con le regole fuzzy.

Il ragionamento fuzzy include parti distinte:

1. Valutazione della regola precedente (la parte IF della regola)
2. Implicazione o applicazione del risultato alla successiva regola (la parte THEN della regola).

Nei sistemi basati su regole classiche, se la regola precedente è vera, allora la successiva è anche vera.

Nei sistemi fuzzy, dove la precedente è una dichiarazione fuzzy, tutte le regole intervengono in una certa misura, o in altre parole intervengono parzialmente. Se la precedente è vera con un certo grado di appartenenza, la successiva è anche vera con lo stesso grado di appartenenza.

Una regola fuzzy può avere numerose regole precedenti:

IF progetto_durata è molta,
and progetto_staff è molto numeroso,
and progetto_fondi sono inadeguati,
THEN rischio è elevato.

Tutte le parti delle regole precedenti sono calcolate ed al risultato è attribuito un singolo numero, usando l'insieme delle operazioni fuzzy c DEFUZZIFICAZIONE del risultato delle regole precedenti.

Le parti successive della regola fuzzy possono essere multiple:

IF temperatura è calda
THEN caldo_acqua è ridotta
freddo_acqua è aumentata

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 59 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 61 di 120

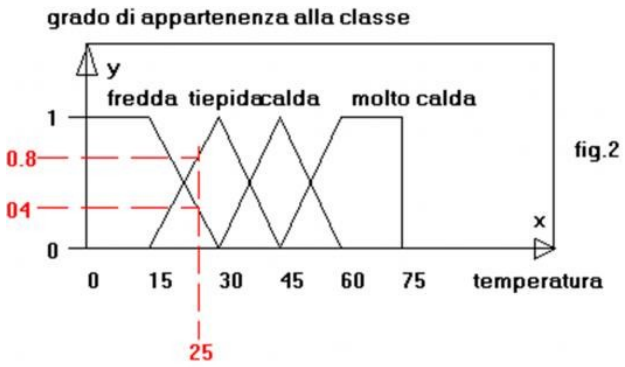


fig.2

Tali funzioni matematiche, che possono essere di varia natura (trapezoidale, triangolare…), attuano il processo di "fuzzificazione dell'input" (fig.2), ovvero la trasformazione del dato preciso in dato di tipo fuzzy:

dato preciso di temperatura = 25°C

dato fuzzy temperatura { = tiepida con credibilità0.8
{ = fredda con credibilità 0.4

Matematicamente il processo di fuzzificazione consiste in alcuni semplici calcoli sulle funzioni appena descritte. Il primo passo da fare è uno scanning dei valori estremi di ogni classe per cui quando si ha che: limite_inf_class < *x* < limite_sup_class cioè se il valore *x* è compreso (estremi esclusi) tra il limite inferiore e quello superiore che delimitano la classe, allora *x* appartiene a tale classe e occorre calcolare il grado di appartenenza. Bisogna quindi definire uno "slope", cioè una pendenza, sui lati delle funzioni che definiscono le classi, che sia fisso per ogni classe e ci consenta di calcolare il grado di appartenenza di un input *x* alla classe *Y*. A questo punto bisogna vedere se il valore *x* cade nella parte bassa della classe (pendenza positiva) o nella parte alta (perdenza negativa): per fare questo è necessario verificare se: x > oppure x < di (lim_sup – lim_inf)/2 che rappresenta il centro della classe. Nel primo caso si effettuerà il calcolo con lo slope positivo e nel secondo caso con lo slope negativo (fig.3):

1) gda = (x - lim_inf)*slope(se gda > max allora gda =max)
2) gda = (lim_sup – x)*slope(se gda > max allora gda = max) (gda = grado di appartenenza o "degree ofmembership")

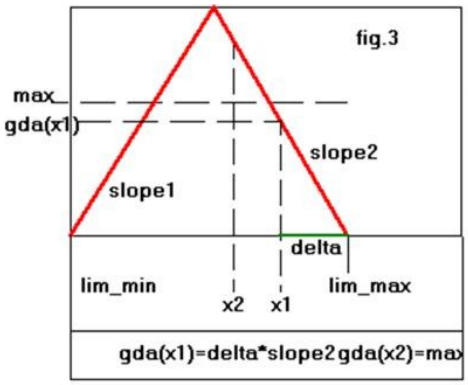


fig.3

Questo è uno dei possibili approcci alla soluzione del problema di fuzzificazione, infatti è possibile implementare funzioni di qualunque tipo per definire il grado di appartenenza di un input ad una classe; si tratta solamente di calcolare il valore di una funzione in un particolare punto definito dal valore di input.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 61 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 63 di 120

- Intersezione (AND):
  - *Crisp sets*: quale elemento appartiene ad entrambi i set?
  - *Fuzzy sets*: quanto l'elemento appartiene ad entrambi i set?

Le funzioni che valutano l'intersezione tra i fuzzy sets (i(a,b)) sono chiamate **t-norms**.

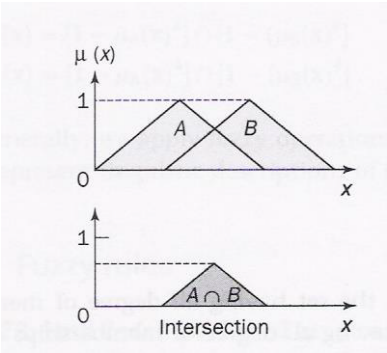
Se A e B sono sottoinsiemi fuzzy di X e C=A∩B, allora C(x)=T(A(x),B(x)) per alcune t-norme di T.

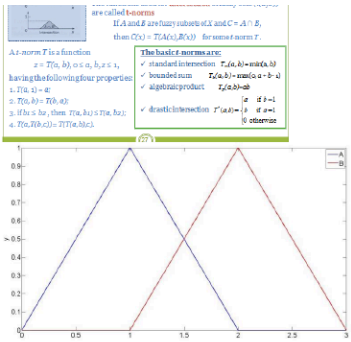
Una t-norma T è una funzione z=T(a,b), 0≤a,b,z≤1, avente le seguenti proprietà:

1. $T(a,1) = a$;
2. $T(a,b) = T(b,a)$;
3. Se $b1 \leq b2$, allora $T(a,b1) \leq T(a,b2)$;
4. $T(a,T(b,c)) = T(T(a,b),c)$.

Le t-norme di base sono:

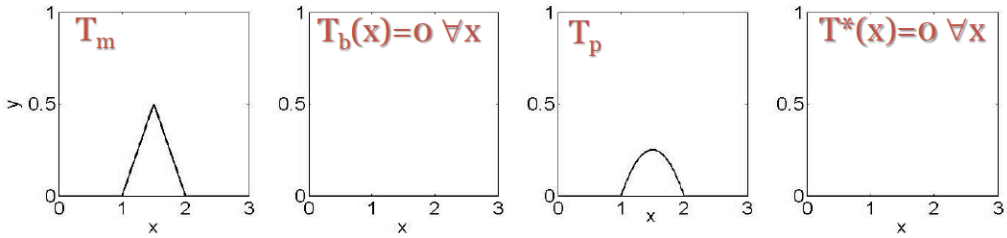
- *Intersezione standard:* $T_m(a, b) = \min(a, b)$
- *Somma limitata:* $T_b(a, b) = \max(0, a + b - 1)$
- *Prodotto algebrico:* $T_p(a,b)=ab$
- *Intersezione drastica:* $T^*(a,b) = \begin{cases} a & \text{if } b=1 \\ b & \text{if } a=1 \\ 0 & \text{otherwise} \end{cases}$





The basic *t*-norms are:

- ✓ standard intersection $T_m(a, b) = \min(a, b)$
- ✓ bounded sum $T_b(a, b) = \max(0, a + b - 1)$
- ✓ algebraic product $T_p(a,b)=ab$
- ✓ drastic intersection $T^*(a,b) = \begin{cases} a & \text{if } b=1 \\ b & \text{if } a=1 \\ 0 & \text{otherwise} \end{cases}$



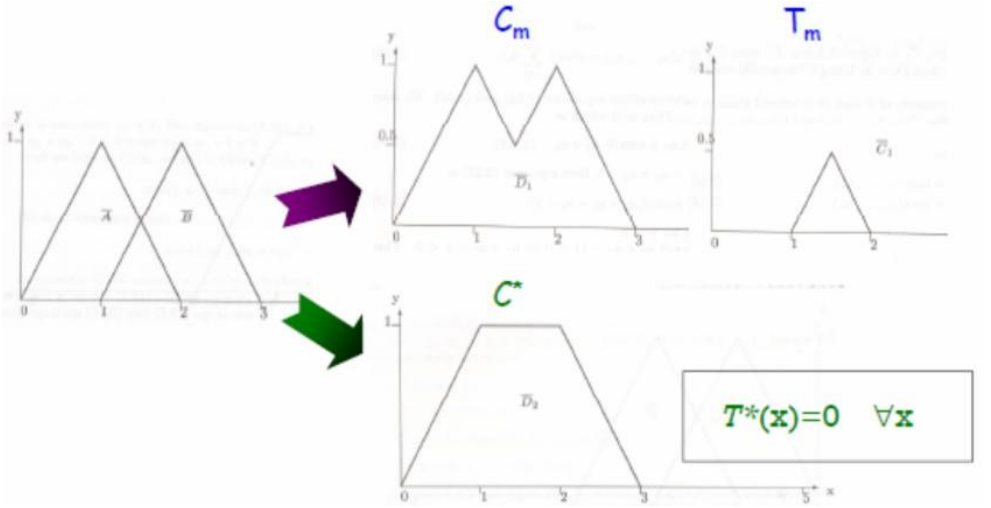
- Unione (OR):
  - *Crisp sets*: quale elemento appartiene ad entrambi i set?
  - *Fuzzy sets*: quanto l'elemento appartiene ad entrambi i set?

Le funzioni che valutano l'unione di fuzzy sets (u(a,b) sono chiamate **t-conorme**.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 63 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 65 di 120

In pratica di solito si usa una coppia di T e C che sono duale. Diciamo T e C sono duali quando
T (a, b) = 1- C (1 -a, 1- b),
C (a, b) = 1- T (1-, 1- b).
Di seguito sono duali:
(1) Tm e Cm; (2) Tb e Cb (3) Tp e Cp; (4) T * e C *
La maggior parte delle persone utilizzano Tm e Cm in tutti i loro calcoli con insiemi fuzzy.
Tuttavia non è necessario utilizzare sempre lo stesso t-norma e t-conorme.
Mixed logica fuzzy è quando si utilizza T = T1, C = C1, di solito doppi operatori, per alcuni calcoli e passare alla T = T2, C = C2, probabilmente anche doppi operatori, per altri calcoli.



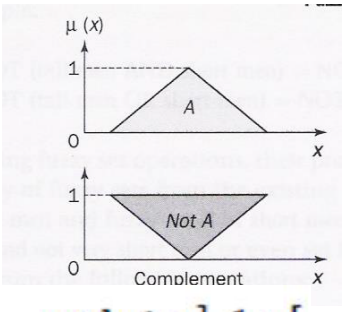
- ***<u>Complemento:</u>***
  - *<u>Crisp sets</u>*: quale elemento appartiene ad entrambi i set?
  - *<u>Fuzzy sets</u>*: quanto l'elemento appartiene ad entrambi i set?

- Complemento fuzzy standard: $\mu_{Ac}(x) = 1 -: \mu_A(x) \ \forall \ x \in X$
- Complemento fuzzy generale: Nel complemento fuzzy generale può essere definito mediante una mappatura funzionale c nella forma:

1. c(0) = 1 e c(1) = 0;
2. $\mu_1 \leq \mu_2 \Rightarrow c(\mu_1) \geq c(\mu_2)$;

Nella maggior parte dei casi comuni vengono considerate due proprietà aggiuntive:

3. c è una funzione continua;



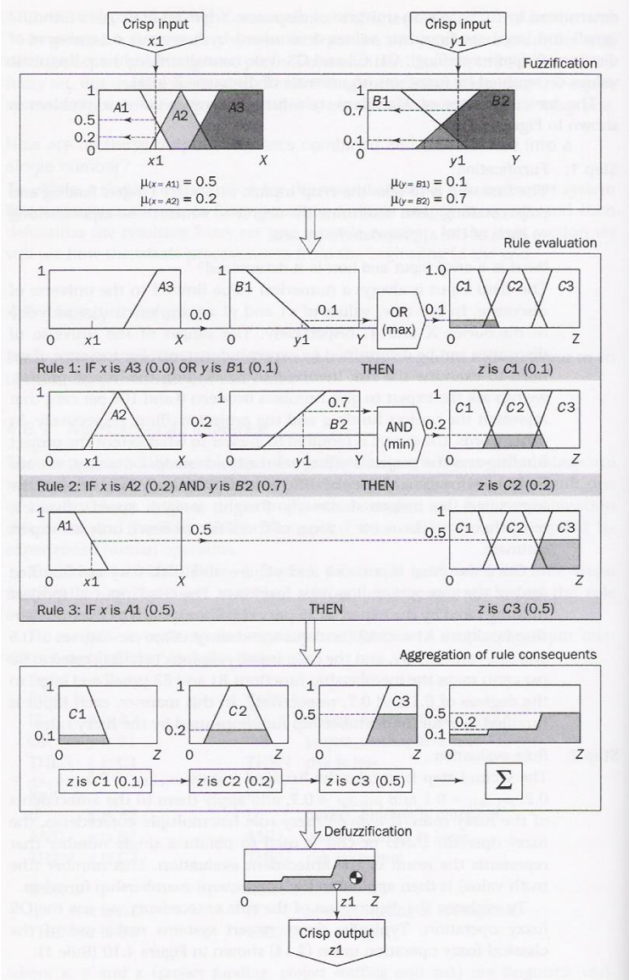
- 4. $c[c(\mu)] = \mu$, $\forall \ \mu \in [0,1]$
- Complementi di classe di Sugeno: $c_{Sug}(\mu) = \dfrac{1-\mu}{1+\lambda\mu}$ with $\lambda \in]-1,\infty[$
- Complementi di classe di Yager: $c_{Yag}(\mu) = (1-\mu^w)^{1/w}$ with $w \in]0,\infty[$

Per valutare la disgiunzione delle regole antecedenti, usiamo l'operazione fuzzy OR. In genere, i sistemi esperti fuzzy fanno uso della classica operazione di unione fuzzy.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 65 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 67 di 120

Lo stile Mamdani per il processo di fuzzy inference implica quattro passi:
1. Fuzzificazione delle variabili di input
2. Valutazione delle regole
3. Aggregazione delle regole di output
4. Defuzzificazione

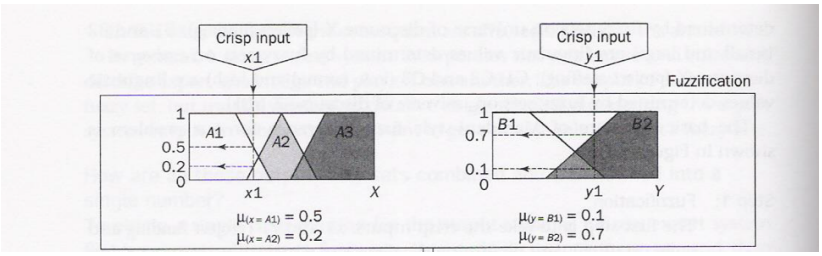


1. Fuzzificazione.

Il primo passo consiste nel prendere gli input di tipo crisp e determinare il grado con cui questi input appartengono a ciascun fuzzy set appropriato.

Il crisp input è sempre un valore numerico limitato all'universo del discorso. Sicuramente diversi sistemi fuzzy utilizzano una gran varietà di crisp input. Mentre alcuni input possono essere misura direttamente (altezza, peso, velocità, distanza, temperatura, pressione, etc.), altri derivano esclusivamente da una stima

dell'utente. Una volta ottenuti i crisp input, essi sono fuzzificati rispetto all'insieme fuzzy linguistico appropriato. Ciascun ingresso è fuzzificato in base al le funzioni di appartenenza utilizzate dalle regole fuzzy.



2. Valutazione delle regole.

Il secondo passo consiste nel prendere gli input fuzzificati ed applicare loro la parte precedente delle regole fuzzy. Se una regola fuzzy a molte parti precedenti, l'operatore fuzzy (AND o OR) è usato per ottenere un singolo numero che rappresenta il risultato della valutazione della parte precedente. Questo numero (il valore vero) è applicato alla successiva membership function. Per valutare la disgiunzione delle regole precedenti si usa l'operazione fuzzy OR. Tipicamente, un sistema fuzzy esperto fa uso della classica operazione fuzzy di unione. $\mu_{A \cup B}(x) = \max[\mu_A(x), \mu_B(x)]$

Comunque l'operazione OR può essere facilmente personalizzata se necessario. Per esempio, il MATLAB Fuzzy Logic Toolbox ha due metodi OR integrati: il metodo del massimo e il metodo del OR probabilistico, "probor". Il OR probabilistico, noto anche come somma algebrica, è calcolato come:

$\mu_{A \cup B}(x) = \text{probor}\,[\mu_A(x), \mu_B(x)] = \mu_A(x) + \mu_B(x) - \mu_A(x) \times \mu_B(x)$

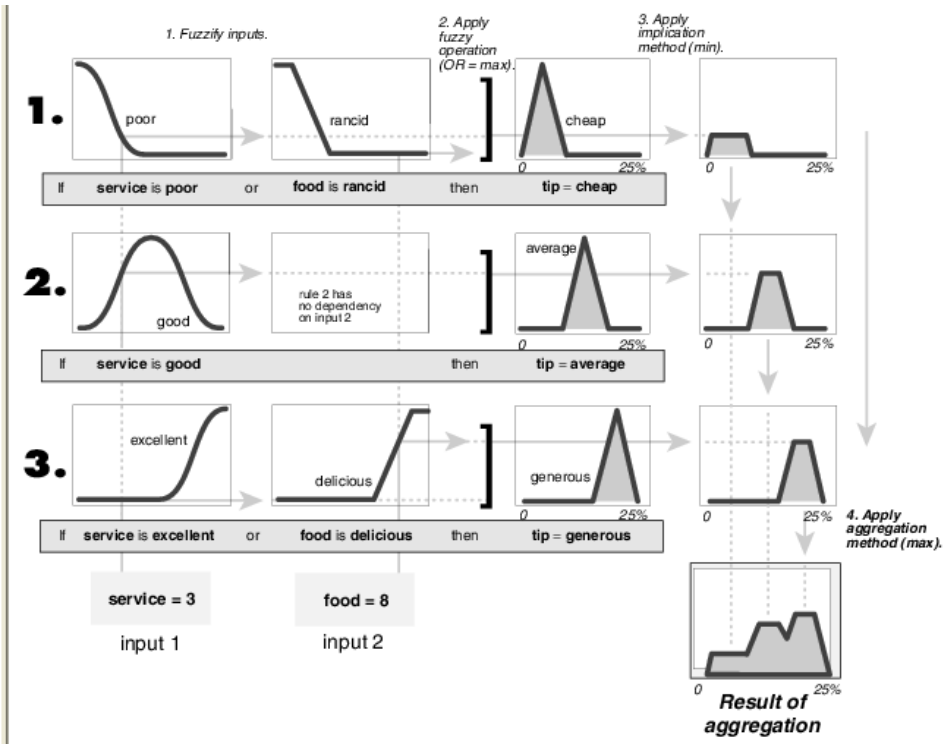
Similmente, al fine di valutare la congiunzione delle regole precedenti, si applica l'operazione fuzzy di intersezione

AND: $\mu_{A \cap B}(x) = \min[\mu_A(x), \mu_B(x)]$

Anche il Fuzzy Logic Toolbox supporta due metodi AND: il minimo e il prodotto, "prod". Il

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 67 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 69 di 120



4. Defuzzificazione.

L'ultimo passo del processo di inferenza fuzzy è la defuzzificazione. La fuzziness aiuta a valutare le regole, ma il più delle volte l'output finale di un sistema fuzzy dev'essere un numero crisp. L'input del processo di defuzzificazione è il fuzzy set aggregato di output e l'output è un unico numero.

Ci sono diversi metodi di defuzzificazione, ma probabilmente il più popolare è la tecnica del centroide. Trova il punto dove una linea verticale taglia il set aggregato in due quantità uguali. Matematicamente questo centro di gravità
(COG) può essere espresso da:

$$COG = \frac{\int_a^b \mu_A(x)x\,dx}{\int_a^b \mu_A(x)\,dx}$$

In teoria, il COG è calcolato su un numero continuo di punti nella MF aggregata di output, ma in pratica si fa una stima ragionevole su di un campione di punti. In questo caso la formula discretizzata è:

$$COG = \frac{\sum_{x=a}^{b} \mu_A(x)x}{\sum_{x=a}^{b} \mu_A(x)}$$

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 69 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 71 di 120

8 Reti Neurali

La rete neurale artificiale (RN) simula l'attività computazionale che avviene nel cervello umano durante l'apprendimento ed il riconoscimento per decidere o per giungere a conclusioni, ed è essenzialmente una interconnessione di semplici elementi computazionali basati su funzioni matematiche. Essa usa migliaia o centinaia di migliaia di neuroni simulati fortemente interconnessi per processare le informazioni in parallelo.

L'espressione rete neurale può acquisire pertanto due significati distinti:

1. Le reti neurali biologiche: sono costituite dai neuroni biologici, cellule viventi tipiche degli animali connesse tra loro o connesse nel sistema nervoso periferico o nel sistema nervoso centrale. Nel campo delle neuroscienze, sono spesso identificati come gruppi di neuroni che svolgono una determinata funzione fisiologica nelle analisi di laboratorio.
2. Le reti neurali artificiali: sono modelli matematici che rappresentano l'interconnessione tra elementi definiti neuroni artificiali, ossia costrutti matematici che in qualche misura imitano le proprietà dei neuroni viventi.

Questi modelli matematici possono essere utilizzati sia per ottenere una comprensione delle reti neurali biologiche, ma ancor di più per risolvere problemi ingegneristici di intelligenza artificiale come quelli che si pongono in diversi ambiti tecnologici (in elettronica, informatica, simulazione, e altre discipline).

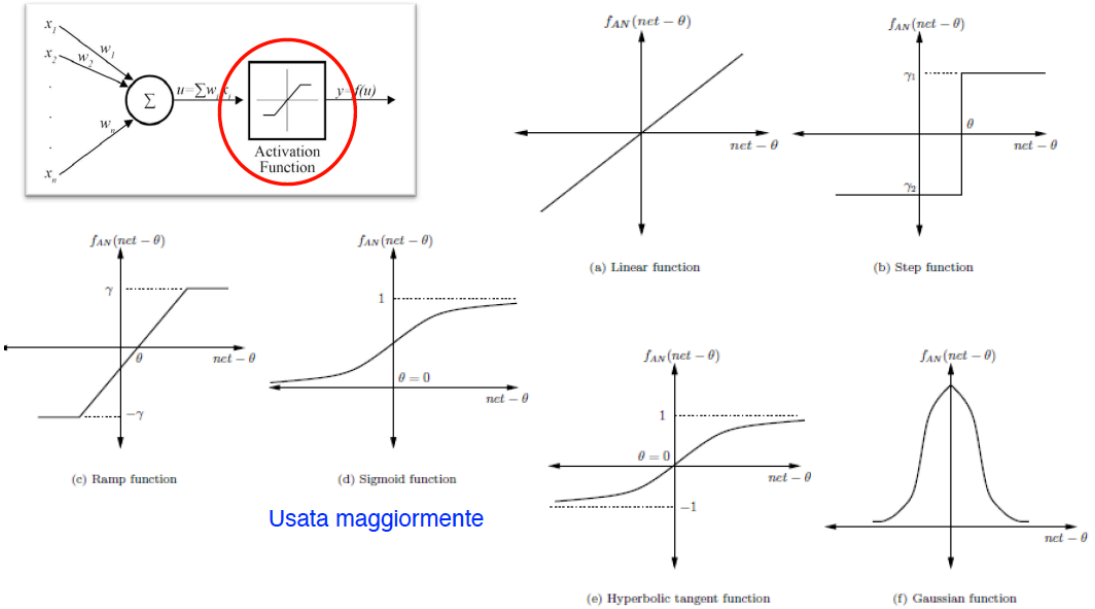
Una rete neurale artificiale normalmente è chiamata solo "rete neurale" (NN "Neural Network" in inglese), ed è un modello matematico/informatico di calcolo basato sulle reti neurali biologiche. Tale modello è costituito da un gruppo di interconnessioni di informazioni costituite da neuroni artificiali (Architettura della rete), dal tipo di neuroni (Neuroni) e processi che utilizzano un approccio di connessionismo di calcolo (Algoritmi di apprendimento). Nella maggior parte dei casi una rete neurale artificiale è un sistema adattivo che cambia la sua struttura basata su informazioni esterne o interne che scorrono attraverso la rete durante la fase di apprendimento.
In termini pratici le reti neurali sono strutture non-lineari di dati statistici organizzate come strumenti di modellazione. Esse possono essere utilizzate per simulare relazioni complesse tra ingressi e uscite che altre funzioni analitiche non riescono a rappresentare.
Una rete neurale artificiale riceve segnali esterni su uno strato di nodi (unità di elaborazione) d'ingresso, ciascuno dei quali è collegato con numerosi nodi interni, organizzati in più livelli. Ogni nodo elabora i segnali ricevuti e trasmette il risultato a nodi successivi.
Le reti neurali fanno parte dei CI computation intelligence e delle metaeuristiche; prendono spunto dai meccanismi di ragionamento del cervello e come neuroni sono in grado di classificare, di riconoscere un'immagine… Sono lo strumento per eccellenza legato all'apprendimento e ciò avviene mediante i dati, quindi sono necessarie due fasi:

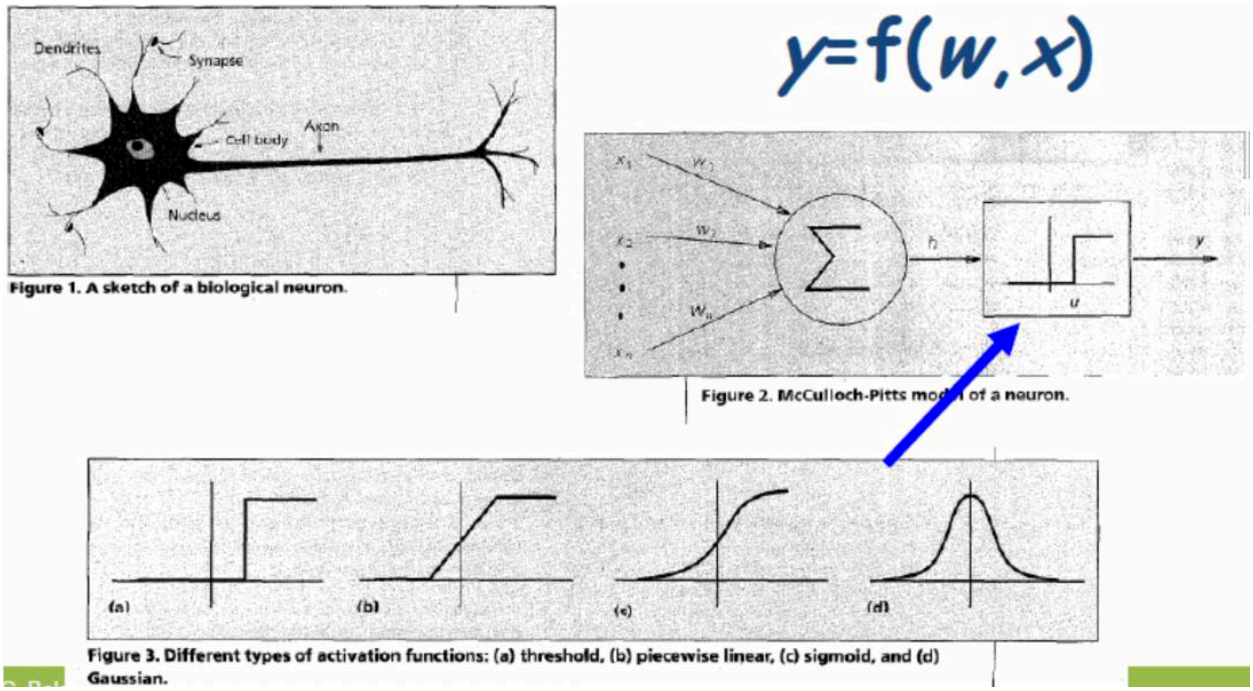
- Fase di apprendimento
- Fase di utilizzo

Tra questa due fasi c'è quella di validazione, quindi, oltre al training set necessario per l'apprendimento, è necessario avere anche un test set. Mentre nella logica fuzzy è possibile vedere quali regole sono state attivate per dare una certa classificazione, nelle reti neurali non c'è questa possibilità, soprattutto in quelle supervisionate; è quindi necessario testare le reti per valutare le loro

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 71 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 73 di 120





Storia.

L'ampia varietà di modelli non può prescindere dal costituente di base, il neurone artificiale proposto da **W.S. McCulloch e Walter Pitts** in un famoso lavoro del **1943**: "A logical calculus of the ideas immanent in nervous activity", il quale schematizza un combinatore lineare a soglia, con dati binari multipli in entrata e un singolo dato binario in uscita: un numero opportuno di tali elementi, connessi in modo da formare una rete, è in grado di calcolare semplici funzioni booleane.

Le prime ipotesi di apprendimento furono introdotte da **D. O. Hebb** nel libro del **1949**: "The organization of behavior", nel quale vengono proposti collegamenti con i modelli complessi del cervello.

Nel **1958**, J. Von Neumann nella sua opera "The computer and the brain" esamina le soluzioni proposte dai precedenti autori sottolineando la scarsa precisione che queste strutture possedevano

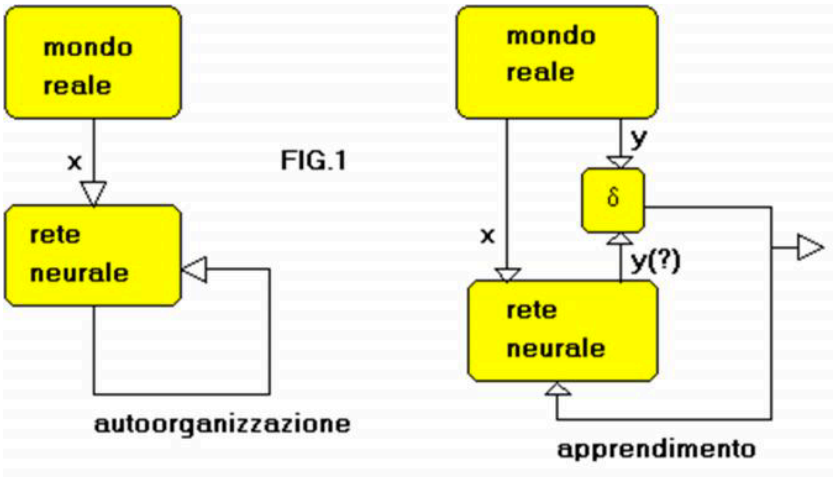
© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 73 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 75 di 120

confrontata con l'uscita desiderata ottenendo il segnale d'errore. L'errore calcolato è propagato nella direzione inversa rispetto a quella delle connessioni sinaptiche. I pesi sinaptici infine sono modificati in modo da minimizzare la differenza tra l'uscita attuale e l'uscita desiderata (backward-pass). Tale algoritmo consente di superare le limitazioni del percettrone e di risolvere il problema della separabilità non lineare (e quindi di calcolare la funzione XOR), segnando il definitivo rilancio delle reti neurali, come testimoniato anche dall'ampia varietà d'applicazioni commerciali: attualmente la BP rappresenta un algoritmo di largo uso in molti campi applicativi.

Allenamento e Paradigmi di apprendimento.

L'allenamento o training è il processo di modifica dei pesi assegnati alle connessioni in maniera ordinata, sfruttando un metodo di apprendimento.



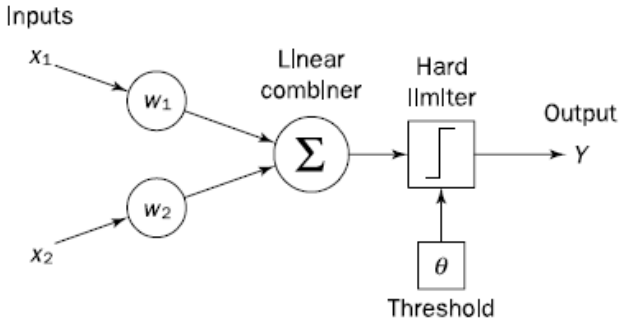
Vi sono tre grandi paradigmi di apprendimento, ciascuno corrispondente ad un particolare compito astratto di apprendimento. Si tratta dell'apprendimento supervisionato, apprendimento non supervisionato e l'apprendimento per rinforzo. Di solito un tipo di architettura di rete può essere impiegato in un qualsiasi di tali compiti.

- un apprendimento **supervisionato** (supervised learning), qualora si disponga di un insieme di dati per l'addestramento (o training set) comprendente esempi tipici d'ingressi con le relative uscite, "pattern", loro corrispondenti: in tal modo la rete può imparare ad inferire la relazione che li lega. Ognuna delle coppie è chiamata "fact" (fatto). È importante che tutte le informazioni che la rete deve imparare siano nel training set e cioè che i facts siano numerosi e vari tali da poter rappresentare tutto il range di situazioni possibili. La rete analizzerà sequenzialmente un fact alla volta. Per ogni fact la rete preleva l'input e produce un suo output. Quest'ultimo viene confrontato con il valore del pattern del fact. Successivamente, la rete è addestrata mediante un opportuno algoritmo (tipicamente, la back-propagation che è appunto un algoritmo d'apprendimento supervisionato), il quale usa tali dati allo scopo di modificare i pesi ed altri parametri della rete stessa in modo tale da minimizzare l'errore di previsione (e=y-d; y=output, d=target) relativo all'insieme d'addestramento. Se l'addestramento ha successo, la rete impara a riconoscere la relazione incognita che lega le variabili d'ingresso a quelle d'uscita, ed è quindi in grado di fare previsioni anche laddove l'uscita non è nota a priori; in altri termini, l'obiettivo finale dell'apprendimento supervisionato è la previsione del valore dell'uscita per ogni valore valido dell'ingresso, basandosi soltanto su un numero limitato di esempi di corrispondenza

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 75 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 77 di 120

e X (y) è la funzione di output. Le scelte più comuni per la funzione X (y) sono:
X (y) = sign (y)
X (y) = y θ (y)
X (y) = y
dove θ (y)è la funzione di Heaviside.
Il primo caso corrisponde a un classificatore binario (l'output può assumere solamente i valori + 1 e − 1); un caso particolarmente studiato è quello in cui sia gli input x che l'output *f* (x) sono binari.
Il bias b può essere pensato come un settaggio della funzione di attivazione (per es. quando X (y) è come nel caso 3), o come un livello base di attivazione per l'output del neurone (per es. quando X (y) è come nei casi 1 e 2). In quest'ultima situazione, il valore − b rappresenta un valore di soglia che la somma pesata degli input deve superare affinché il dispositivo sia attivo (cioè che l'output sia positivo).
Il percettrone può essere considerato come il più semplice modello di rete neurale feed-forward, in quanto gli input alimentano direttamente l'unità di output attraverso connessioni pesate. Nel caso in cui gli input e gli output sono dello stesso tipo, è possibile creare reti più complesse unendo più percettroni insieme, per esempio usando un gruppo (o strato) di percettroni come input per un secondo gruppo di percettroni, oppure facendo in modo che l'input di ogni percettrone della rete sia dato dall'output di ogni altro percettrone (rete fully-connected). La funzione di attivatori è a gradino, mi permette di dividere lo spazio in due parti, chiamato iperpiano.



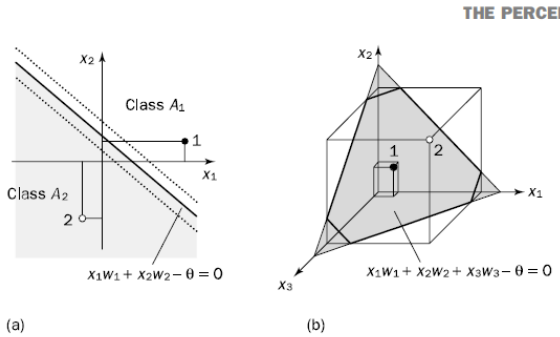


Figure 6.6 Linear separability in the perceptrons: (a) two-input perceptron; (b) three-input perceptron

Lo scopo del perceptron è di classificare l'input in una o due classi. Nel caso di un perceptron elementare, lo spazio di dimensione n è diviso da un iperpiano in due regioni differenti.

<u>Algoritmi di apprendimento o Learning mode.</u>
I più popolari algoritmi di apprendimento per le reti feed-forward sono: Perceptron, Adaline, Madeline, Cognitron, Neo-Cognitron, Competition, Boltzman, Harmony, Counter Propagation e

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 77 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 79 di 120

2. REGOLE DELTA O LEAST MEAN SQUARED (LMS)

DWij = u · [Ti(t) - Oi(t)] · Oj(t)
u = learning rate.

t = tempo (istante considerato).
Ti(t) = desiderato valore di output per il neurone i-simo (pattern i-simo).
Oi(t) = valore di uscita effettivo fornito (durante il training) dal neurone i-esimo.
Oj(t) = valore di output del neurone j-simo.
Si vede chiaramente nella regola DELTA se [Ti(t)-Oi(t)]≠0, i pesi sono cambiati da questa differenza.

3. La **Back Propagation**, detta anche **REGOLA DELTA GENERALIZZATA**, è una variazione della regola DELTA che viene utilizzata per reti dotate di uno o più strati intermedi. È evidente quindi come, le reti feed-forward tramite queste regole di apprendimento, riescono a produrre un effetto di retroazione. I segnali di errore non sono retroazionati tramite connessione diretta tra output e input (come nelle reti feedback). Bensì si alterano opportunamente, i pesi delle connessioni interne procedendo da quelle di output fino a quelle di input, in modo da influenzare positivamente i segnali interni, evitando che si ripeta lo stesso errore.

Algoritmo di apprendimento standard.
L'algoritmo di apprendimento standard è un algoritmo iterativo, definito come segue: ad ogni iterazione t, un **vettore di input x^t** viene presentato al percettrone, che calcola **l'output $f(x^t)$** e lo confronta con il risultato
desiderato, il **target $g(x^t)$;** quindi, il **vettore dei pesi w^t** viene aggiornato come segue:

$$w^{t+1} = w^t + \alpha\,(g(x^t) - f(x^t))\,x^t$$

dove α è una costante di apprendimento strettamente positiva che regola la velocità dell'apprendimento. Al passo successivo, il nuovo input xt + 1 sarà pesato secondo il nuovo vettore $w^t + 1$, che verrà poi nuovamente modificato in $w^t + 2$ e così via.
L'insieme D_x da cui sono estratti i campioni x presentati al percettrone durante il periodo dell'apprendimento è detto training set.
Nel caso in cui, per ogni possibile vettore di input $x \in D_x$, esistono un vettore x e una costante γ tali che $g(x) \cdot ((w, x) + b) > \gamma$, il training set è detto linearmente separabile (geometricamente, questa condizione descrive la situazione in cui esiste un iperpiano in grado di separare, nello spazio vettoriale degli input, quelli che richiedono un output positivo da quelli che richiedono un output negativo). In questo caso, Novikoff (1962) ha provato che l'algoritmo standard converge, nel senso che il numero di errori è limitato da $\left(\frac{2R}{\gamma}\right)^2$, in un numero finito di passi. Non è invece garantito che l'algoritmo descritto converga se il training set non è linearmente separabile.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 79 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 81 di 120

aggiornati in maniera appropriata.

I passi logici per addestrare una rete neurale con apprendimento supervisionato sono i seguenti:

- Creare un insieme di pattern input ed il relativo insieme di pattern di output desiderati
- Inizializzare i pesi della rete neurale (le connessioni tra i neuroni) a dei valori casuali, piccoli rispetto ai valori futuri che assumeranno, ed a norma nulla.
- Ciclo di apprendimento (esce da questo ciclo solo quando l'errore generale è minore di quanto si è deciso oppure dopo un determinato numero di iterazioni)
    - o Ciclo feed-forward (dallo strato di input a quello di output)
        - ✓ estrarre un pattern di input a caso tra quelli a disposizione[nota]
        - ✓ calcolare il valore di tutti i neuroni successivi (sommatorie di produttorie)
        - ✓ detrarre dal risultato il valore di soglia di attivazione del neurone (se il valore di soglia non è già stato simulato con l'aggiunta di un neurone ad ingresso fisso a valore 1.0)
        - ✓ filtrare l'uscita del neurone applicando una funzione logistica per far diventare tale valore input del neurone successivo
    - o Confrontare il risultato ottenuto della rete con il pattern di output relativo all'input inserito e ricavare l'errore attuale della rete
    - o Ciclo di back-propagation (dallo strato di output a quello di input)
        - ✓ calcolare la correzione da apportare ai pesi secondo la regola di localizzazione del minimo
        - ✓ scelta
        - ✓ applicare la correzione ai pesi dello strato

N.B. al fine di avere un buon addestramento l'insieme dei pattern di input deve essere completo, quindi dopo aver scelto in modo randomizzato un pattern, nella estrazione del successivo quello vecchio non deve partecipare al ballottaggio. In questo modo parteciperanno solo chi non è ancora stato estratto, per poi ripescare dall'insieme completo quando tutti sono stati estratti almeno una volta.

Multilayer Perception

Un perceptron singolo strato può classificare solo i modelli linearmente separabili, indipendentemente dal fatto che si usa un hard-limite o una funzione di attivazione soft-limit. Per far fronte ai problemi che non sono linearmente separabili abbiamo bisogno reti neurali multistrato. In realtà, la storia ha dimostrato che i limiti di percettrone di Rosenblatt possono essere superate da forme avanzate di reti neurali, per esempio perceptron multistrato addestrati con l'algoritmo backpropagation.

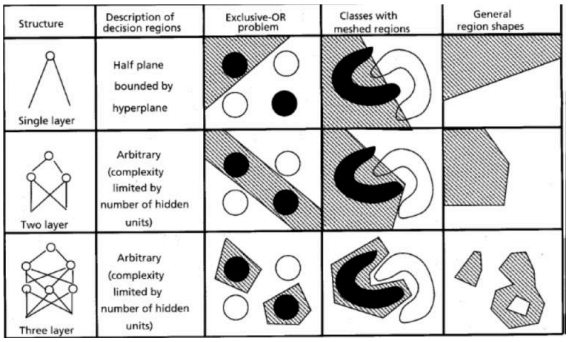
Un perceptron multistrato è un feedforward ANN adatto per compiti di classificazione. Esso è composto da più neuroni collegati tra loro e organizzati in strati, in modo da essere in grado di risolvere i problemi più complessi di singolo perceptron.

Struttura e tipi di reti neurali.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 81 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 83 di 120

questo compito, allora nel successivo strato (intermedio) ognuno dei neuroni calcola il proprio output. Ogni neurone intermedio ottiene il risultato tenendo conto dei segnali provenienti da tutti i neuroni dello strato di input. Infatti il neurone intermedio è collegato con una distinta connessione (dall'appropriato "peso") ad ognuno dei neuroni dello strato di input. In genere, quindi, gli output sono differenti per ognuno dei neuroni intermedi. Quando tutti i neuroni intermedi hanno implementato il loro risultato, quelli di output (o del successivo strato intermedio) calcolano il loro output basando sempre sulla somma pesata dei segnali provenienti da tutti i neuroni intermedi. Dato che ogni neurone deve essere dotato di una connessione diretta (con il relativo "peso") con ognuno dei neuroni dello strato precedente, la rete feed-forward ha bisogno di memorizzare una notevole quantità di pesi (molto meno rispetto alle reti feedback).
Se per esempio uno strato ha N neuroni ed il precedente strato ha M neuroni, la connessione tra questi due strati richiederà N*M parole di memoria (una parola per ogni peso). Quindi anche se questa necessità di memoria è sempre abbastanza gravosa, essa è indispensabile poiché è noto che qualsiasi "conoscenza" memorizzata nella rete neurale è contenuta nella configurazione dei pesi delle connessioni.
Utilizzando un appropriato metodo di apprendimento o learning mode si produce un processo di modifica graduale dei pesi (training) fino a raggiungere la configurazione ottimale.
Per costruire la struttura di una rete neurale multistrato si possono inserire N strati Hidden; vi sono però alcune dimostrazioni che mostrano che con 1 o 2 strati di Hidden si ottiene una stessa efficace generalizzazione da una rete rispetto a quella con più strati Hidden. L'efficacia di generalizzare di una rete neurale multistrato dipende ovviamente dall'addestramento che ha ricevuto e dal fatto di essere riuscita o meno ad entrare in un minimo locale buono.
Le reti feed-forward sono divise in lineari e non lineari. Perceptron e Adaline sono infatti reti che utilizzano neuroni dotati di funzione di trasferimento lineare che permette prestazioni molto limitate.
Gli algoritmi di apprendimento detti "competitivi" sono di tipo unsupervised e si applicano invece a reti con neuroni non lineari. È il caso delle reti di Kohonen o Counter Propagation le quali però mostrano alcuni inconvenienti: durante il training alcuni dei neuroni diventano rapidamente molto sensibili all'input della rete, tentando così di apprendere tutte le associazioni senza permettere alla maggior parte degli altri neuroni di partecipare al training.



Come scritto in un rapporto della D.A.R.P.A (Defensive Advanced Research Project Agency) sulle reti neurali, con uno strato decisionale (output) è possibile realizzare una separazione lineare, mentre con due strati (output+hidden1) possono essere separati spazi convessi e, con tre strati decisionali (output+ +hidden1 +hidden2) possono essere riconosciute forme qualsiasi (vedi figura

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 83 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 85 di 120

disponendo in input di informazioni (o modelli) incomplete. Le reti auto-associative (anche dette CAM – Content Addressable Memory) più conosciute sono quelle dedotte dagli studi di Hopfield (1982-1986) che hanno la caratteristica di essere essenzialmente reti ad un solo strato. In un auto-associatore tutti i neuroni sono sia di input che di output ed ogni neurone alimenta sé stesso così come ognuno degli altri ad esso vicini. Potenzialmente esiste una connessione modificabile da ogni neurone ad ogni altro neurone della rete. Particolari applicazioni delle reti di Hopfield sono quelle implementate per risolvere "Il problema del commesso viaggiatore" (da ciò il nome "TSP" di tali reti). Le memorie auto-associative sono applicate anche per il riconoscimento di caratteri o di immagini, per la ricostruzione di segnali, per la gestione delle segnalazioni di sicurezza negli aeroporti, nelle industrie, nelle centrali nucleari.

2. Le *etero-associative* sono quelle reti in cui sono richiesti in ingresso informazioni (o modelli) complete per rievocare altre informazioni (o modelli) memorizzate. Le principali reti etero associative sono quelle cosiddette BAM (Memorie Associative Bidirezionali). Esse trovano applicazione nei processi di monitoraggio, nell'analisi di andamenti finanziari e nella "target classification".

Reti RBF (Radial Basis Functions).

Appartengono al gruppo delle reti neurali supervisionate.

È un approccio differente nel disegno di una rete neurale come un problema di "curve fitting", ovvero la risoluzione di un problema di approssimazione, in uno spazio a dimensione molto grande. (interpolazione di dati in uno spazio di elevata dimensione) Quindi l'apprendimento si riduce a trovare una superficie in uno spazio multi-dimensionale che fornisce il miglior "fit" per i dati di training, in cui il miglior fit si intende misurarlo in modo statistico. In maniera analoga, la fase di generalizzazione µe equivalente ad usare questa superficie multidimensionale ricercata con i dati di training per interpolare i dati di test mai visti prima dalla rete. La rete è strutturata su tre livelli, input, hidden e output. Lo strato di input è direttamente connesso con l'ambiente, ovvero sono direttamente collegati con le unità sensoriali (dati grezzi) oppure con l'output di un sottosistema di feature extraction. Lo strato hidden (unico nella rete) è composto da neuroni in cui vengono definite funzioni a base radiale, da qui il nome di radial basis functions, e che effettua una trasformazione non-lineare dei dati di input forniti alla rete (trasforma i dati di input in un nuovo spazio). Questi neuroni costituiscono delle basi per i dati di input (vettori). Lo strato di output è lineare, che fornisce la risposta della rete per l'input pattern presentato. La ragione per cui si adopera una trasformazione non-lineare nello strato hidden seguita da una lineare in quello di output è descritta in un articolo di Cover (1965) secondo il quale un problema di classificazione di pattern riportato in uno spazio a dimensione molto più grande (ossia nella trasformazione non lineare dallo strato di input in quello hidden) è più probabile di essere linearmente separabile che in uno spazio a dimensione ridotta. Da questa osservazione

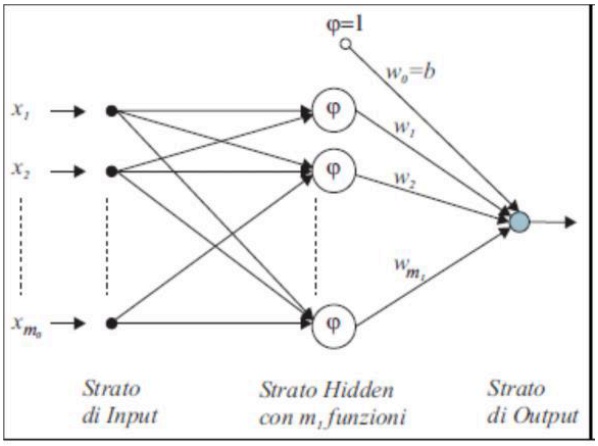
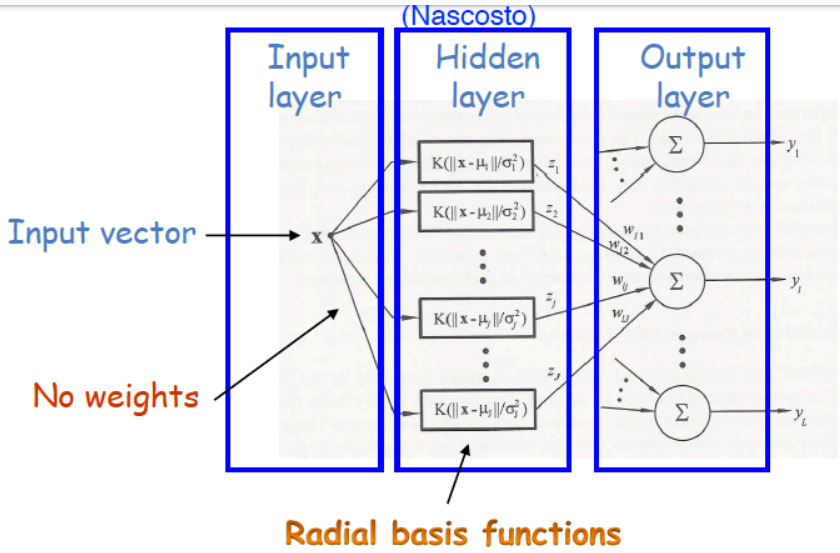


Figura 1.4: Architettura di rete RBF

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 85 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 87 di 120



La sommatoria è una trasformazione lineare

I pesi w,mu e sigma sono parametri NON noti

Mu è il centro dalla basis function del mio specifico neurone

$$z_j(x) = K\left(\frac{\|x-\mu_j\|}{\sigma_j^2}\right)$$

Each hidden unit output is obtained by calculating the closiness of the input $\boldsymbol{x}$ to a n-dimensional parameter vector $\boldsymbol{\mu}$, associated with the jth unit.

Sigma è l ampiezza della mia radial basis

A number of RBFs have been proposed

Il 2 indica la distanza euclidea
L output del layer nascosto diventa l input del layer di output

- **Linear** function, where

$$\Phi(||\mathbf{z}_p - \mu_j||_2) = ||\mathbf{z}_p - \mu_j||_2$$

- **Cubic** function, where

$$\Phi(||\mathbf{z}_p - \mu_j||_2) = ||\mathbf{z}_p - \mu_j||_2^3$$

- **Thin-plate-spline** function, where

$$\Phi(||\mathbf{z}_p - \mu_j||_2) = ||\mathbf{z}_p - \mu_j||_2^2 \ln ||\mathbf{z}_p - \mu_j||_2$$

- **Multiquadratic** function, where

$$\Phi(||\mathbf{z}_p - \mu_j||_2, \sigma_j) = \sqrt{||\mathbf{z}_p - \mu_j||_2^2 + \sigma_j^2}$$

Dipendono anche dalla ampiezza

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 87 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 89 di 120

Con queste premesse siamo in presenza di un problema di interpolazione multivariata in spazi a grandi dimensioni.
La tecnica RBF consiste nello scegliere una funzione F, cioè la superficie interpolante, con la seguente forma:

$$F(x) = \sum_{i=1}^{N} \omega_i \varphi(\|x - x_i\|)$$

Con $\{\varphi(\|x - x_i\|) \mid i = 1,2,\ldots, N\}$ l'insieme di N funzioni generalmente non lineari, *denominate radial basis function*, e $\|\cdot\|$ denota la norma Euclidea. I punti noti del dataset $x_i \in \Re^{\eta 0+1}$, i=1,2,…,N sono i centri delle funzioni radiali.
L'accuratezza di un RBF NN è influenzata da:
- Il numero di funzioni base utilizzate. Più funzioni base saranno utilizzate, migliore sarà l'approssimazione della funzione target. Tuttavia, le funzioni di base necessarie aumentano la complessità computazionale.
- La posizione delle funzioni di base come definito dal vettore centrale, μ_j, per ogni funzione base. Le funzioni base devono essere uniformemente distribuite per coprire l'intero spazio di ingresso.
- Per alcune funzioni, la larghezza del campo recettivo, σ_j. Più è largo σ_j, più la basis function rappresenterà lo spazio di input.

La formazione di un RBF NN dovrebbe quindi prendere in considerazione metodi per trovare i migliori valori per questi parametri.

RBF Training
Sono stati sviluppati un certo numero di metodi per implementare RBF NNS.
Questi metodi si differenziano principalmente per il numero di parametri che vengono appresi. L'algoritmo <u>centri fissi</u> adatta solo i pesi tra gli strati nascosti e di uscita (fissa i centri delle radial basis e fa la fase di learning sui pesi che connettono gli output layers).
<u>Adaptative Centres training algorithms</u> adattano entrambi i pesi, i centri e le deviazioni.

<u>Centri fissi</u> (sono usate le RBF Gausiane)
- inizializzo j per indicare il numero dei centri;
- scelgo i centri μ_j, i=1, ..., J con $\mu_j = z_p$, $p \sim U(1, P_T)$
  Tolgo i neuroni che non causano una significativa diminuzione delle performance della mia rete. I centri sono scelti a caso dal training set. A condizione che un numero sufficiente di centri vengano uniformemente scelti dal training set, sarà ottenuto un adeguato campionamento dello spazio di ingresso.
- Calcolo del σ_j, usando l'equazione: $\sigma_j = \sigma = \frac{d_{max}}{\sqrt{J}},\ j = 1, \ldots, J$, d_{max} è la distanza Euclidea massima tra i due centri (il nuovo σ è uguale per ogni neurone);
- Inizializzo i pesi in maniera random: w_{kj}, k=1, ..., K e j=1, ..., J

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 89 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 91 di 120

- *Esistenza*
- *Unicità*
- *Continuità o Stabilità*

Se nessuna di queste condizioni è verificata, il problema si dirà mal-posto. Nei problemi mal- posti, dataset di esempi molto grandi possono contenere poca informazione del problema da risolvere. I fenomeni fisici responsabili per la generazione del dataset per l'addestramento (ad esempio, per il parlato, segnali radar, segnali sonar, immagini ecc.) sono problemi ben-posti.
Tuttavia, l'apprendimento da queste forme di segnali fisici, visto come una ricostruzione di ipersuperfici, è un problema mal-posto per le seguenti ragioni. Il criterio di esistenza può essere violato nel caso in cui output distinti non esistono per ogni input. Non vi è sufficiente informazione nel dataset di training per ricostruire la funzione di mapping input-output in maniera univoca, pertanto il problema di unicità potrebbe essere violato. Il rumore o le imprecisioni presenti nei dati di training aggiunge incertezza alla superficie di mapping input-output.
Quest'ultimo problema viola il principio di continuità, poiché se nei dati vi è molto rumore, è probabile che l'output desiderato y ricada al di fuori del range Y per uno specificato vettore di input $x \in X$. Parafrasando Lanczos (1964) si può dire che *"Non esiste alcun artifizio matematico per rimediare all'informazione mancante nei dati di training"*. Un importante risultato su come rendere un problema mal-posto in uno ben-posto è derivato dalla teoria della Regolarizzazione.

Teoria della Regolarizzazione.
Introdotta da Tikhonov nel 1963 per la soluzione di problemi mal-posti. L'idea di base è quella di stabilizzare la soluzione di ricostruzione di ipersuperfici tramite l'introduzione di qualche funzionale non-negativo che integra informazione a priori della soluzione. La forma più comune di informazione a priori implica l'assunzione che la funzione di mapping input-output (cioè soluzione del problema di ricostruzione) sia di tipo *smooth*, ovvero ad input simili corrispondono output simili.

Reti di Hopfield.
Nel 1982, il fisico John J. Hopfield pubblica un articolo fondamentale in cui presenta un modello matematico comunemente noto appunto come rete di Hopfield: tale rete si distingue per "l'emergere spontaneo di nuove capacità computazionali dal comportamento collettivo di un gran numero di semplici elementi d'elaborazione". Le proprietà collettive del modello producono una memoria associativa per il riconoscimento di configurazioni corrotte e il recupero di informazioni mancanti.
Inoltre, Hopfield ritiene che ogni sistema fisico può essere considerato come un potenziale dispositivo di memoria, qualora esso disponga di un certo numero di stati stabili, i quali fungano da attrattore per il sistema stesso. Sulla base di tale considerazione, egli si spinge a formulare la tesi secondo cui la stabilità e la collocazione di tali attrattori sono proprietà spontanee di sistemi costituiti, come accennato, da considerevoli quantità di neuroni reciprocamente interagenti.
Tra le nuove idee messe in luce da Hopfield, quella più degna di menzione riguarda il capovolgimento del rapporto, fino allora esistente, tra calcolo e numeri: mentre era universalmente noto che il calcolo producesse numeri, assai meno banale era l'osservazione di Hopfield che, viceversa, anche i numeri potessero spontaneamente generare calcolo, e che questo potesse emergere quale attributo collettivo di sistemi interattivi siffatti.
Le applicazioni delle reti di Hopfield riguardano principalmente la realizzazione di memorie

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 91 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 93 di 120

vicini, così che ogni bolla di attivazione rappresenta una classe di inputs aventi caratteristiche somiglianti.
Modalità di funzionamento. Come la maggior parte delle reti neurali artificiali, la SOM ha due modalità di funzionamento:

1. Durante **il processo di addestramento** si costruisce una mappa, la rete neurale si organizza usando un processo competitivo. È necessario dare in ingresso alla rete un numero elevato di vettori d'ingresso, che rappresentino il più possibile il tipo di vettori che ci si aspetta durante la seconda fase (se ce ne sarà una). Altrimenti, gli stessi vettori d'ingresso devono essere "somministrati" più volte.
   Il neurone vincente individua efficacemente il centro di una zona topologica. La zona dovrebbe essere una funzione decrescente della distanza laterale tra i neuroni. Nel quartiere sono inclusi solo i neuroni eccitati, mentre esistono i neuroni inibiti al di fuori del quartiere.
   Per un dato nodo:
   - vicini vicini: Cooperativa (reciprocamente eccitatori, w> 0),
   - vicini più lontani: competitivo (reciprocamente inibitorio, w <0).
   - vicini troppo lontano: irrilevanti (w = 0).
   Il vicino topologico h_{ji} è simmetrico attorno al punto di massimo definito da $d_{ij} = 0$; in altre parole, esso raggiunge il valore massimo con il neurone vincente *i* la cui distanza è zero. Una scelta tipica dello h_{ji} è la funzione gaussiana che è una versione invariante (cioè indipendente dalla posizione del neurone vincente):

$$h_{ji(\vec{x})} = \exp\left(-\frac{d_{ij}^{\,2}}{2\sigma^2}\right)$$

   Il parametro sigma è la "larghezza efficace" della zona. Si misura il grado in cui i neuroni eccitati nella zona del neurone vincente partecipano al processo di apprendimento.

2. Durante il **processo** di **mapping** un nuovo vettore d'ingresso può essere dato in ingresso alla mappa; questo viene automaticamente classificato o categorizzato. Ci sarà un solo neurone vincitore: quello il cui vettore dei pesi giace più vicino al vettore d'ingresso. (Questo può essere facilmente individuato calcolando la distanza euclidea fra il vettore d'ingresso e il vettore dei pesi).

Passi dell'algoritmo.

1. Assegna ai vettori dei pesi valori casuali.
2. Prendi un vettore d'ingresso.
3. Attraversa ogni nodo della mappa:
   1. Usa la distanza euclidea per trovare somiglianze fra il vettore d'ingresso e il vettore dei pesi di ogni singolo nodo della mappa.
   2. Individua il nodo a distanza minore (questo nodo verrà chiamato Best Matching Unit o BMU).
4. Aggiorna i nodi del vicinato di BMU "tirandoli" più vicino al vettore d'ingresso:
   1. $W_v(t + 1) = W_v(t) + \Theta(v, t) \cdot \alpha(t) \cdot [D(t) - W_v(t)]$.

Interpretazione. Ci sono due modi per interpretare una SOM:

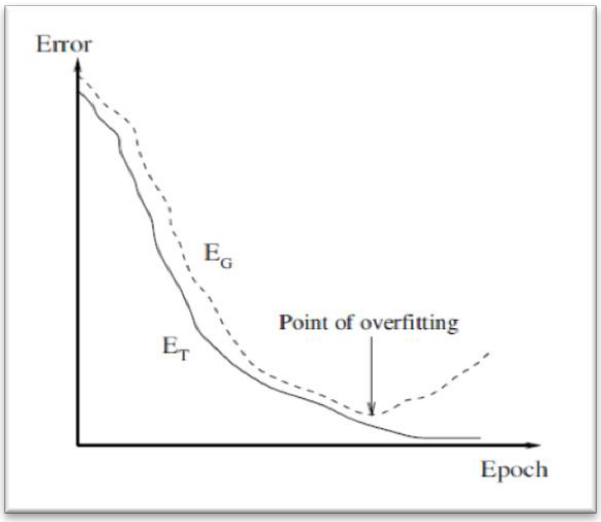
- Dato che nella fase di addestramento i pesi di tutto il vicinato sono spostati nella stessa

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 93 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 95 di 120

$$r = \frac{\sum_{p=1}^{P} o_{k,p} t_{k,p} - \frac{1}{P}\sum_{p=1}^{P} o_{k,p} \sum_{p=1}^{P} t_{k,p}}{\sqrt{\sum_{p=1}^{P}(o_{k,p})^2 - \frac{1}{P}\left(\sum_{p=1}^{P} o_{k,p}\right)^2}\sqrt{\sum_{p=1}^{P}(t_{k,p})^2 - \frac{1}{P}\left(\sum_{p=1}^{P} t_{k,p}\right)^2}}$$

Un altro aspetto molto importante di accuratezza NN è il sovradattamento→Overfitting.
Overfitting di un training set significa che NN memorizza i modelli di formazione(training patterns), e quindi perde la capacità di generalizzare. Cioè, gli NN che fa overfit non possono prevedere l'uscita corretta per i modelli di dati che non si vedono durante l'allenamento.
Overfitting verifica quando l'architettura NN è troppo grande, cioè il NN ha troppi pesi come conseguenza dell'avere troppe unità nascoste ed unità di entrata irrilevanti. Se il NN è eseguito per troppo tempo, i parametri liberi in eccesso iniziano a memorizzare tutti i modelli di formazione, e anche il rumore contenuto nel training set.



I rimedi per overfitting comprendono l'ottimizzazione dell'architettura della rete e l'utilizzo di abbastanza modelli di training.
Le stime di errore di generalizzazione durante il training può essere utilizzato per rilevare il punto di overfitting: l'insieme di dati originali è diviso in tre insiemi disgiunti, cioè l'insieme di addestramento T, il set di generalizzazione G e il set di validazione V. Il set di validazione viene poi utilizzato per stimare l'errore di generalizzazione.

Il fattore di generalizzazione è come un'alternativa all'overfitting. $\rho = \frac{\epsilon_V}{\epsilon_T}$
dove ε_V e ε_T sono rispettivamente MSE sul set di validazione V e l'attuale sottoinsieme di training T. L'Overfitting viene rilevata quando $\rho(t) > \varphi_\rho(t)$
dove $\varphi_\rho(t) = \min\{\varphi_\rho t\text{-}1, \rho + \sigma_\rho, 1.0\}$, t è l'epoca attuale, ρ è il fattore medio di generalizzazione su un numero fisso di epoche precedenti, e σ_ρ è la deviazione standard. Questo test assicura che $\rho \leq 1.0$ ed è utile per i problemi di funzione di approssimazione dove il MSE è utilizzato come misura di precisione. Per problemi di classificazione in cui la percentuale di modelli correttamente classificati viene utilizzata come misura di accuratezza, ρ dovrebbe essere maggiore di 1. È importante notare che l'errore formazione o l'errore di generalizzazione da solo non è sufficiente per quantificare la precisione di un NN. Entrambi questi errori devono essere considerati.

La complessità computazionale di un NN è direttamente influenzata da:
1. L'architettura di rete: Più grande è l'architettura, le più calcoli feedforward sono necessari per prevedere uscite dopo l'allenamento, e più calcoli di apprendimento sono necessari per la presentazione del modello.
2. dimensione del training set: Maggiore è la dimensione insieme di addestramento, più modelli sono presentati per il training. Pertanto, viene aumentato il numero totale di calcoli di apprendimento per epoca.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 95 di 120

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 97 di 120

Fattori della Performance: Codifica dei valori di input

Tutti i valori di ingresso ad un NN devono essere numerici: quindi hanno bisogno di essere trasformati da valori nominali a valori numerici.

Ci sono due possibilità:

1. Un parametro di ingresso nominale che ha *n* valori diversi viene codificato come *n* diversi parametri di input binari, in cui il parametro di input, che corrisponde ad un valore nominale, contiene il valore 1, e il resto di questi parametri ha il valore 0.
2. Una alternativa è quella di utilizzare un solo parametro di input e mappare ogni valore nominale in un valore numerico equivalente. Questo, tuttavia, non è una buona idea, poiché la NN interpreterà il parametro di ingresso come avente valori continui, perdendo quindi la caratteristica discreta dei dati originali.

Outliers

I valori anomali sono valori anomali che possono essere causa di un errore nella acquisizione dei dati o possono rappresentare gli elementi atipici.

I valori anomali hanno effetti gravi sulla precisione, soprattutto quando la discesa del gradiente viene utilizzata come la funzione obiettivo con l'SSE. Un valore anomalo è un modello di dati che si discosta sostanzialmente dalla distribuzione dei dati. A causa della grande deviazione dalla norma, valori anomali provocano grandi errori, e di conseguenza grandi aggiornamenti dei pesi. Come risultato, la generalizzazione deteriora. Il problema degli outliers può essere affrontato nei seguenti modi:

- Rimuovere i valori anomali prima dell'inizio del training, mediante tecniche statistiche. Mentre tali azioni eliminano il problema degli outliers, si ritiene potrebbero anche essere rimosse contemporaneamente le informazioni importanti relative ai dati.
- Utilizzare una robusta funzione obiettivo che non è influenzata da valori anomali.

Scaling e Normalizzazione

Non è necessario scalare valori di ingresso, ma in alcuni problemi le prestazioni possono essere migliorate se gli ingressi sono scalati al dominio attivo delle funzioni di attivazione.

Quando si utilizzano funzioni di attivazione delimitate, i valori obiettivo devono essere scalati al range della funzione di attivazione.

Ad esempio (0, 1) per la funzione sigma e (-1, 1) per la tangente iperbolica.

In caso di problemi di classificazione,i valori obiettivo sono generalmente elementi dell'insieme {0.1, 0.9} per la funzione sigmoidale. Il valore di 0,1 è usato invece di 0 e 0,9 invece di 1. Poiché l'uscita della funzione sigmoide può accedere solo a 0 e 1, un NN non può mai convergere al meglio il set dei pesi se i valori obiettivo sono 0 o 1. In questo caso l'obiettivo della NN è sempre fuori portata, e la rete continua a spingere valori di peso verso valori estremi fino a che il valore di training viene interrotto.

© Proprietà riservata dell'autore - Digitalizzazione e distribuzione a cura del CENTRO APPUNTI - Corso Luigi Einaudi, 55 - Torino / Pagina 97 di 120