Modelli Locali per l'Apprendimento:

dall'approccio neuro-fuzzy al lazy learning

Mauro Birattari

1996-97

Sommario

La tesi presenta una rassegna di approcci multimodello al problema dell'apprendimento supervisionato. In particolare, sono stati studiati, implementati ed estesi, il metodo *neuro-fuzzy inference system*, basato sul formalismo fuzzy, e il metodo di regressione locale *lazy learning*. Tali metodi sono stati applicati a casi di studio per l'approssimazione di funzione e per la predizione di serie temporali. Viene inoltre presentato un problema reale di analisi finanziaria affrontato con il metodo *lazy learning*.

Ringraziamenti

Fortunatamente, le persone che devo ringraziare sono veramente molte. A partire dal Professor Andrea Bonarini che mi ha proposto un lavoro interessante e gratificante e ha sempre avuto per me i consigli giusti al momento giusto. Devo molto anche al Professor Marco Colombetti: attraverso le sue lezioni ho avuto il mio primo contatto con l'intelligenza artificiale. Per ricordare tutti gli ottimi insegnanti che ho avuto voglio ricordare qui il corso del Professor Rinaldi che mi ha introdotto al mondo affascinante della non linearità, e quello del Professor Bittanti che ha avuto una parte fondamentale nella mia preparazione.

Oltre che al Politecnico di Milano, ho svolto parte del mio lavoro presso l'istituto di ricerca IDSIA—Istituto Dalle Molle di Studi sull'Intelligenza Artificiale, Lugano. Ringrazio tutti i ricercatori e in particolare Luca Gambardella per avermi reso disponibili le strutture del laboratorio e per l'accoglienza risevatami.

Grazie al programma TIME—Erasmus ho avuto la possibilità di conoscere il Professor Philippe Smets a cui devo tanto sia da un punto di vista scientifico, il suo lavoro e i suoi seminari mi hanno presentato un punto di vista che non conoscevo sulla teoria delle decisioni, sia perché il laboratorio IRIDIA riflette profondamente la sua personalità, il suo modo di intendere la ricerca e i rapporti tra le persone. E' proprio al laboratorio IRIDIA—Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, ULB—che ho svolto gran parte del mio lavoro. Qui ho conosciuto dei ricercatori competenti, da cui ho avuto moltissimo da imparare, e delle persone magnifiche, da cui è stato facile imparare. Robert Kennes, Bruno Marchal, Vittorio Gorrini, Nick Bradshaw, Christine Decaestecker, Alessandro Saffiotti, Masaaki Minagawa, Antoine Duchâteau, Marco Dorigo, Gianni Di Caro, e ovviamente il Professor Hugues Bersini.

Bruno ha avuto più volte, e sono sicuro che l'avrà ancora, la pazienza di spiegarmi particolari del suo lavoro. Di Alessandro ho ammirato il modo di concepire la ricerca scientifica e l'enorme passione. Con Gianni ho avuto discussioni che mi hanno consentito di elaborare e formalizzare il discorso presentato in qusta tesi. Antoine e Nick sono state le persone con cui ho lavorato più a stretto contatto: con loro è sempre stato stimolante parlare e

ragionare. Sicuramente molto di aiuto sono stati i consigli che mi ha dato Marco, e molto interessanti e illuminanti sono stati i suoi seminari. A tutti loro devo moltissimo. Un ringraziamento particolare lo tengo da parte per Vittorio il cui aiuto mi è stato fondamentale da tutti i punti di vista: è alla competenza dei suoi consigli, ai suoi suggerimenti e alla sua disponibilità che devo il fatto di essermi adattato rapidamente al lavoro in laboratorio e alla vita a Bruxelles.

Il Professor Hugues Bersini ha seguito il mio lavoro da quando sono arrivato a Bruxelles e ha avuto molte cose da insegnarmi. Gli devo molto per l'importanza del suo lavoro all'interno della mia tesi, e gli sarò sempre grato per l'aiuto che mi ha dato e per la crescita professionale che mi ha consentito. Hugues ha voluto inserirmi all'interno del gruppo di persone che collaborano con lui e mi ha permesso di applicare a dei problemi reali gli algoritmi a cui stavo lavorando.

Devo sicuramente molto a Luca, a mia mamma e a mio papà, sia per il supporto logistico e per avermi finanziato (!), sia soprattutto perchè mi hanno insegnato la curiosità. Mi piace qui ricordare anche Emanuele Persico e Francesco Allevi, fantastici compagni di avventura a Bruxelles. Con Franz, in particolare, oltre ad aver preparato molti degli esami, ho anche lavorato allo sviluppo di parte del software utilizzato in questa tesi.

Del lavoro qui presentato, una gran parte è stata discussa con Carlotta che ha gli stessi miei occhi e il mio stesso modo di vedere il mondo. Le nostre analisi e le conclusioni a cui siamo giunti insieme sono state preziose.

Il ringraziamento finale spetta sicuramente all Ing. Gianluca Bontempi che ha seguito con competenza il mio lavoro dall'inizio alla fine. Gianluca mi ha permesso di osservare da vicino il suo lavoro di ricerca e ha voluto sempre propormi un lavoro stimolante e soprattutto gratificante: mi ritengo fortunato.

Indice

Introduzione

Il problema dell'apprendimento supervisionato può essere definito come un caso di stima di dipendenza tra variabili (Vapnik, 1995). Si supponga di avere a disposizione un'insieme di esempi sotto forma di coppie (x, y), in cui x è un vettore che descrive una determinata *situazione ambientale*, e y è uno scalare[1] che rappresenta la *risposta* fornita da un *supervisore* alla situazione individuata da x. Apprendere il comportamento del supervisore a partire dalle coppie *ambiente-risposta*, o in termini più ingegneristici ingresso-uscita, significa, da un punto di vista matematico, ricostruire la funzione multivariata $y = F(x)$ a partire da un'insieme di campioni della funzione stessa. Il problema dell'apprendimento è dunque, in questo senso, equivalente al problema dell'*approssimazione funzionale* (Poggio & Girosi, 1990).

Un'altra disciplina che tratta problemi simili a quelli dell'apprendimento supervisionato è la *teoria dell'identificazione*: è facile infatti riconoscere nella definizione di processo di apprendimento sopra fornita, il processo di identificazione di un modello di tipo *black box* (Ljung, 1993).

Queste osservazioni portano a dare la seguente definizione del problema dell'apprendimento supervisionato:

> *Apprendere significa stimare una funzione non nota, a partire da campioni estratti da questa.*

I metodi di approssimazione funzionale, e quindi di apprendimento, possono essere classificati in due grandi categorie: quella dei metodi *globali*, e quella dei metodi *locali*.

I diversi metodi globali di approssimazione funzionale, quali ad esempio quelli che considerano polinomi o reti neurali multistrato (Rumelhart & McClelland, 1986), prevedono che un singolo modello parametrico venga utilizzato per approssimare la funzione $F(x)$ su tutto il dominio di interesse. L'approccio globale può rivelarsi inappropriato quando le caratteristiche della funzione $F(x)$, e la densità degli esempi disponibili, variano fortemente all'interno dello spazio della variabile di ingresso x (Lawrence *et al.*, 1996).

[1] L'estensione al caso in cui y sia un vettore non presenta ulteriori difficoltà concettuali e può essere ottenuta considerando più problemi scalari distinti.

L'approccio locale, o *multimodello*, affronta il problema complesso dell'approssimazione di una funzione non lineare, riducendolo ad una serie di sottoproblemi più semplici (Jacobs *et al.*, 1991a). Il dominio della funzione $F(x)$ viene partizionato in regioni all'interno delle quali la funzione data è ben approssimabile da modelli locali dotati di una semplice struttura parametrica. L'approssimazione di $F(x)$ è dunque data dalla ricomposizione di più approssimazioni parziali fornite dai vari modelli locali, valide ciascuna in una determinata regione del dominio della funzione $F(x)$ stessa. L'approccio multimodello può essere visto (Żbikowski *et al.*, 1994) come un'estensione dell'approccio classico alla modellizzazione di un sistema non-lineare; approccio che prevede la linearizzazione del sistema in esame nell'intorno di un *punto di lavoro* dato. Un modello così ottenuto, risulta valido nelle vicinanze del *punto di lavoro* individuato, ma fornisce una rappresentazione la cui qualità si degrada all'allontanarsi da questo. Nell'approccio multimodello, vengono definiti più *punti di lavoro*, e a ciascuno di questi è associato un modello locale assieme alla relativa regione di validità, in modo tale che l'unione delle varie regioni definite copra l'intero spazio della variabile x.

In letteratura esistono diversi metodi che sono espressione del paradigma locale. In particolare, appartengono alla famiglia dei metodi multimodello gli approcci gerarchici quali *classification and regression tree* CART (Breiman *et al.*, 1984), *multivariate regression splines* MARS (Friedman, 1991), ID3 (Quinlan, 1993) e *hierarchical mixture of experts* HME (Jordan & Jacobs, 1994). Questi metodi generano una partizione dello spazio di ingresso in regioni annidate e, utilizzando un approccio proprio dei metodi di *classificazione* (Duda & Hart, 1973), determinano regole di decisione per attribuire ogni valore x ad una determinata classe di punti dello spazio di ingresso, per i quali la funzione $F(x)$ ha la massima probabilità di essere correttamente approssimata da uno specifico approssimatore locale. Altri metodi appartenenti alla medesima famiglia sono i metodi di *approssimazione lineare a tratti* (Skeppstedt *et al.*, 1992) (Billings & Voon, 1987), le *radial basis functions* RBF (Powell, 1987), una serie di metodi provenienti dalla letteratura statistica quali i modelli *non-parametric state-dependent* (Priestley, 1988), il modello *smooth threshold AR* (Tong, 1990), e altri metodi basati sull'utilizzo delle *spline* (Kavli, 1993). E' inoltre interessante notare come certe forme di *gain scheduling* (Shamma & Athanas, 1990), adottate nel settore del controllo, derivino dalle stesse considerazioni che hanno portato ad affrontare problemi di approssimazione funzionale attraverso architetture multimodello.

A Johansen e Foss (1993) si deve la formalizzazione di una classe generale di modelli che prende il nome di *local model networks* LMN. All'interno di tale classe, sotto certe condizioni, possono essere descritte molte delle architetture citate.

Una trattazione separata deve essere riservata all'architettura *neuro-fuzzy* (Tak-

agi & Sugeno, 1985) (Jang, 1993). Pur essendo a tutti gli effetti un approccio multimodello e pur essendo dimostrata, per particolari configurazioni, la sua equivalenza funzionale alle *radial basis functions* (Jang & Sun, 1993), e alle *local model networks* (Hunt *et al.*, 1994), l'approccio *neuro-fuzzy* proviene storicamente da un settore più vicino dell'intelligenza artificiale simbolica, e di questa conserva alcune caratteristiche peculiari.

L'approccio neuro-fuzzy ha origine dall'approccio fuzzy *linguistico* (Zadeh, 1973) (Mamdani, 1977) che descrive qualitativamente un fenomeno attraverso insiemi di regole di produzione simili a quelle che costituiscono i *sistemi esperti*. Il modello fuzzy viene interpretato come un particolare approssimatore parametrico a cui si possono applicare tecniche di apprendimento. Pur sacrificando parte della leggibilità caratteristicha dei modelli fuzzy tradizionali ai fini di una più accurata approssimazione, l'architettura neuro-fuzzy consente all'analista un certo controllo nella fase di identificazione del modello. In particolare, consente di incorporare nel modello conoscenza *a priori* sul comportamento assunto dalla funzione $F(x)$ in determinate regioni dello spazio di ingresso per le quali è dato un modello fisico, o che non sono sufficientemente coperte da esempi[2]. In questo senso, dal punto di vista della teoria dell'identificazione dei modelli, l'architettura neuro-fuzzy è da considerarsi come una particolare struttura *gray box* (Ljung, 1993), all'interno della quale è possibile incorporare conoscenza *a priori*.

In generale, per tutti i metodi citati, il problema dell'apprendimento può essere formulato operativamente come un problema di ricerca del modello che, sull'intero dominio di definizione della funzione in esame, minimizza una cifra di merito globale, indice della bontà dell'approssimazione fornita e data dal *valore atteso* dell'errore di stima commesso dal modello stesso. La ricerca del modello che minimizza una tale cifra di merito riflette la filosofia dei *grandi campioni* (Vapnik, 1995): tra tutti i modelli alternativi, quello scelto in base al criterio della minimizzazione di una cifra di merito globale è il modello che fornisce in termini probabilistici la migliore prestazione, se usato per stimare i valori assunti da $F(x)$ in un grande numero di punti. In altri termini, il presupposto che giustifica l'identificazione di un modello, o come nel nostro caso di una struttura multimodello, attraverso la minimizzazione di una cifra di merito globale, è che l'approssimatore così ottenuto venga realmente utilizzato per stimare il valore assunto dalla funzione in esame, in un grande numero di punti distribuiti su tutto il suo dominio. Seguendo il criterio illustrato, la stima del valore assunto in un determinato punto x_* dalla funzione non nota, viene effettuata in due passi. In un primo passo viene identificato un modello della funzione in esame, e in un secondo passo il modello ottenuto viene valutato

[2]Si pensi ad esempio al problema dell'identificazione di un modello di un impianto industriale che, per motivi di sicurezza, non è possibile portare in determinati regimi di funzionamento.

nel punto x_*.

I metodi di apprendimento locale finora presentati, sono tutti caratterizzati dall'essere composti da più approssimatori, ciascuno dedicato ad una diversa regione dello spazio di ingresso, che vengono ricombinati per generare un'unica struttura multimodello che approssima la funzione $F(x)$ su tutto il dominio di interesse. Questi metodi, dunque, prevedono tutti la definizione di una struttura di *arbitraggio* che determini quale (o quale combinazione) dei diversi modelli locali debba fornire un'approssimazione del valore assunto dalla funzione $F(x)$ in un determinato punto x_*.

Esistono in letteratura altri approcci locali in cui non viene generata nessuna rappresentazione complessiva della funzione in esame. Questi approcci sono noti sotto il nome collettivo di *memory-based learning* in quanto l'unica forma di rappresentazione della funzione $F(x)$ di cui dispongono è fornita dagli esempi dati e immagazzinati in una memoria; per contrasto gli altri metodi visti vengono definiti *model-based learning*. Negli approcci *memory-based* per ogni stima del valore y_* assunto da $F(x)$ in uno specifico punto x_*, viene generato un solo modello locale che agisce indipendentemente dagli altri modelli generati per stimare la funzione in punti differenti. Non è più necessaria quindi una struttura di arbitraggio.

Punto di partenza per lo sviluppo dei metodi *memory-based* è l'osservazione che la definizione di apprendimento vista potrebbe essere più forte del necessario. Nella maggior parte dei casi è possibile infatti dare dell'apprendimento una definizione più debole, e quindi meno vincolante che, come vedremo, consente di trattare in maniera più adeguata quelle situazioni in cui è disponibile solo una limitata quantità di informazione, ed è richiesta l'applicazione della conoscenza estratta dagli esempi solo a pochi e ben precisi casi di test.

Tipicamente ci si trova a dover affrontare un problema che può essere formalizzato nel seguente modo: dato un insieme Z di campioni di una funzione non nota e un punto di test x_*, si vuole stimare il valore y_* assunto in x_* dalla funzione in esame. L'obiettivo, in questa riformulazione del problema, è dunque quello di stimare il valore che una funzione non nota assume in un determinato punto. E' chiaro quindi che stimare il valore y_* in due passi risulta non essere più strettamente necessario. Queste considerazioni evidenziano come lo schema classico per risolvere il problema relativamente semplice della stima del valore assunto da una funzione non nota in un dato punto, risolve prima un problema intermedio decisamente più complesso, cioè l'identificazione di un modello che equivale alla stima dei valori che la funzione in esame assume in tutti gli infiniti punti appartenenti al suo dominio di definizione. Può accadere che gli esempi Z non siano sufficienti per stimare correttamente la funzione in esame, ma che l'informazione disponibile sia ugualmente sufficiente per stimare il valore della funzione in un dato punto (Bottou & Vapnik, 1992).

Il problema dell'apprendimento da esempi, sulla base di queste consider-

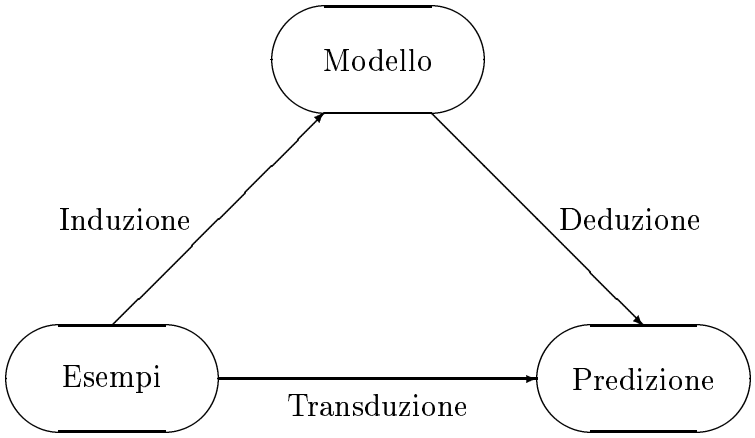


Figura 1: Tre diversi tipi di inferenza.

azioni, può essere ridefinito nella maniera seguente:

> *Apprendere significa stimare il valore che assume punto per punto una funzione non nota, avendo come unica informazione un insieme di valori assunti dalla funzione stessa in altri punti.*

Per comprendere la portata di questa ridefinizione, si rende necessario affiancare alle forme di inferenza tradizionalmente considerate, *induzione* e *deduzione*, una terza forma. Se l'induzione è il movimento *dal particolare al generale* e la deduzione è quello *dal generale al particolare*, la stima del valore assunto in un dato punto da una funzione non nota direttamente da esempi è da descrivere (Vapnik, 1995) attraverso una terza forma di inferenza che prende il nome di *transduzione* e che consiste nel movimento *dal particolare al particolare*.

La differenza tra la definizione di apprendimento ora introdotta e quella inizialmente presentata, può essere vista in termini filosofici e portata nell'ambito del discorso sulla natura dell'intelligenza. La capacità di apprendere dall'esperienza è una delle caratteristiche di un essere intelligente e definire l'apprendimento la come stima di una funzione significa, in un certo senso, ritenere che l'intelligenza abbia che vedere con la capacità di inferire regole generali a partire dal particolare. La definizione dell'apprendimento come stima del valore assunto da una funzione in un dato punto, significa invece ritenere che la base di un comportamento intelligente sia *l'intuizione* o una sorta di inferenza *ad-hoc*.

Contributi originali

Il lavoro presentato riguarda fondamentalmente tre architetture multimodello per l'apprendimento supervisionato: *Hierarchical Mixture of Experts*, *Neuro-Fuzzy*, e *Lazy Learning*. Gli aspetti sviluppati non ancora trattati in letteratura sono i seguenti:

- Per quanto riguarda l'architettura *Hierarchical Mixture of Experts*, si è fatto uso di un metodo statistico (*cross-validation*) per determinare la dimensione ottimale dell'albero binario che costituisce la struttura del modello.

- Nell'ambito dell'approccio *Neuro-Fuzzy* si è impiegata una particolare tecnica di *clustering* nello spazio ingresso-uscita, per inizializzare il modello. Questa tecnica offre notevoli vantaggi rispetto alle tecniche di inizializzazione più comunemente utilizzate che sfruttano informazioni relative solo allo spazio di ingresso. Il metodo sviluppato consente di individuare insiemi di esempi che appartengano allo stesso iperpiano nello spazio ingresso-uscita e che quindi siano effettivamente descrivibili dalla stessa regola.

- Per quanto riguarda l'approccio *lazy learning*, si è estesa la versione locale del metodo di validazione *Press statistic* (Atkeson *et al.*, 1996), anche al caso di polinomi di *grado frazionario*, noti in letteratura con il nome di *polynomial mixing* (Mallows, 1974).

- Sempre per quanto riguarda l'approccio *lazy learning*, si è sviluppato un metodo per rendere adattativa e locale sia la scelta della struttura dell'approssimatore, sia quella della dimensione del *neighborhood* considerato.

Prodotto del lavoro svolto è anche un toolbox *public domain* sviluppato in *matlab*© e disponibile all'indirizzo web `http://iridia.ulb.ac.be/~mbiro/`. Tale toolbox implementa una serie di approcci standard all'apprendimento attraverso architetture neuro-fuzzy e lazy; propone inoltre, in alternativa a quelli tradizionali, i vari algoritmi originali presentati in questa tesi.

Schema della tesi

La tesi ha la seguente struttura.

Capitolo 1. Nella prima parte di questo capitolo definiamo una serie di strumenti dell'analisi statistica adatti per affrontare il problema della stima di una distribuzione di probabilità. Nella seconda parte del capitolo

presentiamo una teoria unitaria dell'apprendimento da esempi visto in termini statistici (Vapnik, 1995): l'apprendimento supervisionato è una forma di inferenza dai dati che può essere convenientemente ricondotta alla stima di una distribuzione di probabilità.

Capitolo 2. Presentiamo uno schema generale di un processo di apprendimento visto come la definizione di un modello a partire da una quantità limitata di dati. In questo capitolo illustriamo i vari passi logici che portano dall'osservazione di un fenomeno, attraverso le fasi necesserie per determinare la struttura ottimale di un modello del fenomeno stesso, fino all'acquisita capacità di riprodurlo. Vengono presentati inoltre una serie di metodi adatti per validare un modello identificato a partire dai dati disponibili quando questi siano scarsi.

Capitolo 3. In questo capitolo illustriamo tre diversi approcci all'apprendimento multimodello presenti in letteratura. In particolare sono trattati gli schemi *mixture of experts*, *fuzzy inference systems*, e *lazy learning*.

Capitolo 4. In questo capitolo trattiamo a fondo il problema dell'identificazione di un modello neuro-fuzzy. Illustiamo, in particolare, una metodologia, basata sul metodo di validazione *cross validation*, per determinare la struttura ottima del modello considerato. Presentiamo inoltre un metodo di *clustering* nello spazio congiunto ingresso-uscita adatto per inizializzare l'algoritmo di identificazione.

Capitolo 5. In questo capitolo presentiamo un metodo *memory-based* adattativo. Tale metodo stima il valore che la funzione in esame assume in un punto assegnato, adattando automaticamente una serie di parametri sulla base delle informazioni acquisite attraverso un processo di validazione locale.

Capitolo 6. Confrontiamo nelle stesse condizioni sperimentali alcuni metodi multimodello appartenenti alle classi *mixture of experts*, *neuro-fuzzy*, e *lazy learning*. Presentiamo un problema di ricostruzione di una superficie nota solo attraverso campioni affetti da rumore, e un problema di predizione di una serie temporale caotica. Descriviamo inoltre un'applicazione del metodo *lazy learning adattativo* ad un caso reale di predizione nell'ambito finanziario.

Conclusioni. Per concludere proponiamo un breve confronto tra gli approcci studiati e presentiamo alcuni sviluppi futuri.

Convenzioni grafiche

s	Esito di un esperimento casuale.
$\mathcal{S}$	Insieme degli esiti possibili.
a	Evento.
$\mathcal{A}$	Insieme degli eventi possibili.
$\mathcal{P}(a)$	Probabilità dell'evento a.
$(\mathcal{S}, \mathcal{A}, \mathcal{P})$	Spazio probabilistico.
ϑ	Vettore di parametri.
$\boldsymbol{\vartheta}$	Vettore di parametri casuali (stima di Bayes).
(G)	Generatore di vettori casuali.
(S)	Supervisore.
(M)	Macchina che apprende.
$f(x, \alpha)$	Funzione implementata da (M).
α	Vettore di parametri.
Λ	Insieme dei possibili valori assunti da α.
$C\big(y, f(x, \alpha)\big)$	Funzione di costo.
$R(\alpha)$	Funzionale di rischio.
$f(x, \alpha_0)$	Estremante del funzionale di costo.
$p_{emp}(x, y)$	Densità empirica congiunta.
$R_{emp}(\alpha)$	Funzionale di rischio empirico.
$f(x, \alpha_n)$	Estremante del funzionale di rischio empirico.
$\mathbf{x} \in \Re^d$	Variabile casuale multivariata tipicamente di ingresso.
$\mathbf{y}$	Variabile casuale di uscita, tipicamente scalare.
$z_i = (x_i, y_i)$	Coppia ingresso-uscita: i-esimo esempio del training set.
n	Numero di osservazioni disponibili.
$Z = \{z_1, z_2, \ldots, z_n\}$	Insieme delle osservazioni disponibili: training set.
$P(\cdot)$	Distribuzione di probabilità. Anche $P_x(\cdot)$
$p(\cdot)$	Densità di probabilità. Anche $p_x(\cdot)$
$prob\{\mathbf{z} \leq q\}$	Probabilità che la variabile casuale $\mathbf{z}$ sia minore di q.
$\pi(j)$	Probabilità discreta.
$u^{(j)}$	j-esima componente del vettore u.
$f^m(x, \alpha)$	Multimodello composto da m modelli locali.
$f_j(x, \theta_j)$	j-esimo modello locale.
x_*	Punto in cui stimare la funzione nel *lazy learning*.
$f_*(x_*, \alpha)$	Modello locale nel *lazy learning*.
p	Grado del polinomio approssimante nel *lazy learning*.
k	Numero di vicini considerati nel *lazy learning*.

Capitolo 1

Fondamenti statistici della Teoria dell'Apprendimento

In questo capitolo introdurremo la teoria dell'apprendimento e mostreremo che i processi di apprendimento possono essere convenientemente descritti come processi di inferenza statistica.

Nel seguito della tesi ci occuperemo di apprendimento a partire da dati empirici ed esamineremo diversi approcci a quella branca dell'intelligenza artificiale che va sotto il nome di *machine learning*. Tratteremo alcuni metodi che consentono di estrarre conoscenza su un determinato fenomeno, a partire da un insieme di osservazioni ripetute del fenomeno stesso.

L'uso intensivo che faremo di dati empirici richiede che ci si appoggi agli strumenti forniti dall'analisi statistica. Nella prima parte di questo capitolo introdurremo i concetti fondamentali della statistica e mostreremo come in letteratura è affrontato il problema della stima di una *funzione di densità di probabilità* dato un insieme di osservazioni estratte da questa. Nella seconda parte del capitolo presenteremo quindi una teoria unitaria dell'apprendimento ispirandoci alla formalizzazione presentata da Vapnik (Vapnik, 1995).

1.1 Probabilità e statistica

Secondo l'assiomatizzazione della teoria della probabilità dovuta a Kolmogorov, un *esperimento casuale* è caratterizzato da tre elementi.

Il primo di questi è l'insieme $\mathcal{S}$ che prende il nome di insieme degli *esiti*. Gli elementi s di tale insieme sono ogni possibile risultato dell'esperimento casuale in questione. Spesso non sono i singoli esiti ad essere significativi, ma si è interessati al verificarsi di certe combinazioni o raggruppamenti di esiti. Tali raggruppamenti prendono il nome di *eventi*. Il secondo elemento caratterizzante un esperimento casuale è quindi l'insieme $\mathcal{A}$ che prende il nome di insieme degli eventi. Questo insieme è tale per cui ogni evento $a \in \mathcal{A}$ è un

sotto insieme di $\mathcal{S}$. Il terzo ed ultimo elemento che caratterizza un esperimento casuale è una funzione $\mathcal{P}(\cdot) : \mathcal{A} \mapsto [0, 1]$ detta *probabilità*, che associa ad ogni evento un numero reale compreso tra zero e uno, estremi inclusi.

Un modello di un esperimento casuale è quindi univocamente determinato qualora sia definito lo spazio probabilistico individuato dalla terna $(\mathcal{S}, \mathcal{A}, \mathcal{P})$.

Una variabile casuale definita sull'esperimento $(\mathcal{S}, \mathcal{A}, \mathcal{P})$, è una variabile $\mathbf{z}$ il cui valore dipende dall'esito s dell'esperimento.

Il problema fondamentale della probabilità consiste (Vapnik, 1995), dato l'esperimento casuale $(\mathcal{S}, \mathcal{A}, \mathcal{P})$, nel dedurre le caratteristiche probabilistiche di una variabile casuale $\mathbf{z}$ definita a partire dall'esperimento casuale in questione.

Il problema fondamentale della statistica consiste invece nello stimare, date n osservazioni indipendenti di una variabile casuale $\mathbf{z}$, la misura di probabilità $\mathcal{P}$ caratterizzante l'esperimento casuale a partire dal quale la variabile $\mathbf{z}$ considerata è stata definita.

La statistica e la teoria della probabilità risultano quindi essere due discipline speculari. La teoria della probabilità è sviluppata come un modello astratto le cui conclusioni sono deduzioni a partire da assiomi; la statistica è l'applicazione della teoria a problemi reali e mira ad indurre un modello generale a partire dall'osservazione empirica ripetuta di realizzazioni particolari di una variabile casuale. In questo senso, la statistica fornisce un corpo di conoscenze e di metodi che si rivelano preziosi per fondare una teoria e una pratica dell'inferenza a partire da dati empirici e quindi, in particolare, dell'apprendimento da esempi.

1.2 Stima di una densità di probabilità

Una prima forma di induzione di un modello statistico a partire da dati empirici è la stima di una densità di probabilità. Tale forma di induzione, come vedremo in seguito, costituirà la base su cui verrà costruita la teoria dell'apprendimento. Parallelamente, sul versante pratico, noteremo come i metodi sviluppati per affrontare il problema della stima di una densità costituiscono il punto di partenza dal quale verranno sviluppati i metodi di apprendimento che introdurremo.

Il problema della stima di una densità di probabilità consiste nell'inferire un modello di una funzione di densità di probabilità $p(z)$ non nota, dato un numero finito di osservazioni $\{z_1, z_2, \ldots, z_n\}$ estratte dalla funzione di densità stessa. Distinguiamo tre diverse classi di metodi per stimare una densità di probabilità:

Metodi parametrici. Sono metodi che partono dall'assunzione che la densità di probabilità da cui le osservazioni sono estratte, appartenga ad una determinata famiglia parametrica di funzioni. Il modello risulta quindi

essere descritto da un numero fissato a priori di parametri i cui valori sono identificati attraverso un processo di ottimizzazione che adatta il modello alle osservazioni disponibili.

Metodi non parametrici. Questi metodi non fanno alcuna assunzione a priori sulla funzione di densità di probabilità da cui sono estratte le osservazioni. Il numero di parametri impiegati per descrivere il modello, cresce al crescere del numero delle osservazioni disponibili. La forma del modello risulta quindi determinata esclusivamente dalle osservazioni stesse.

Metodi semi parametrici. Sono metodi in cui il numero di parametri che descrivono il modello, non è fissato a priori e può variare in maniera indipendente dal numero di osservazioni disponibili.

1.2.1 Stime parametriche

In questo approccio si assume che la densità di probabilità $p(z)$ da stimare, appartenga alla famiglia di funzioni rappresentabili dal funzionale $p(z, \vartheta)$, dove ϑ è un parametro vettoriale di dimensione definita e fissata a priori. Il valore di tale parametro è identificato attraverso un processo di ottimizzazione che adatta il funzionale $p(z, \vartheta)$ all'insieme dei dati disponibili $Z = \{z_1, z_2, \ldots, z_n\}$.

Stimatori lineari

Si supponga che l'insieme delle osservazioni sia stato generato dal seguente sistema:

$$Z = H\vartheta + R,$$

dove Z è il vettore delle n osservazioni, H è una matrice di dimensione $n \times k$ con $n > k$, ϑ è un parametro vettoriale di dimensione $k \times 1$ e R è una realizzazione di un vettore casuale i cui componenti sono rumore o errori di misura, associati alle osservazioni; si assume che tali errori abbiano valore atteso nullo.

Uno stimatore lineare di ϑ ha la seguente forma:

$$\hat{\vartheta} = BZ.$$

Se in particolare si pone

$$B = (H^T H)^{-1} H^T,$$

lo stimatore lineare prende il nome di *stimatore dei minimi quadrati*:

$$\hat{\vartheta}_{mq} = (H^T H)^{-1} H^T Z.$$

Lo stimatore dei minimi quadrati possiede un gran numero di proprietà interessanti che ne giustificano la notevole diffusione:

- è semplice da realizzare,
- è non polarizzato,
- se $E[RR^T] = \sigma^2 I$ con I matrice unitaria[1], risulta essere lo stimatore a minima varianza.

Per queste sue proprietà, lo stimatore dei minimi quadrati è detto essere il *migliore stimatore lineare.*

Nel caso più generale in cui risulti $E[RR^T] = K$, con K matrice generica, vale il seguente teorema di Gauss-Markov.

Teorema 1. *Si consideri il modello:*

$$Z = H\vartheta + R,$$

dove Z è il vettore delle n osservazioni, H è una matrice $n \times k$ con $n > k$, ϑ è un parametro vettoriale di dimensione $k \times 1$ e R è una realizzazione di un vettore casuale. Si assuma che $E[R] = 0$ e che $\mathrm{E}[RR^t] = K$. Sotto queste ipotesi, lo stimatore lineare non polarizzato a minima varianza di ϑ risulta essere:

$$\hat{\vartheta} = (H^T K^{-1} H)^{-1} H^T K^{-1} Z.$$

Massima verosimiglianza

Si consideri la funzione di densità $p(z, \vartheta)$ che dipende da un parametro vettoriale ϑ, e un insieme di osservazioni $Z = \{z_1, z_2, \ldots, z_n\}$ estratte indipendentemente dalla densità di probabilità $p(z, \vartheta)$ stessa. La densità di probabilità congiunta dell'intero insieme di osservazioni risulta essere:

$$p(Z, \vartheta) = \prod_{i=1}^{n} p(z_i, \vartheta) = L(\vartheta),$$

dove per un dato insieme di osservazioni n, la funzione L dipende solo da ϑ e prende il nome di *verosimiglianza* di ϑ dato Z.

La tecnica della massima verosimiglianza consiste nel massimizzare la funzione $L(\vartheta)$, con l'obiettivo di trovare il valore più verosimile per il parametro vettoriale ϑ dato l'insieme di osservazioni Z. La stima di ϑ nell'ottica della massima verosimiglianza, risulta essere:

$$\hat{\vartheta} = arg \max_{\vartheta} L(\vartheta).$$

[1]Ciòsignifica ipotizzare che le componenti di rumore sulle n osservazioni siano indipendenti e identicamente distribuite.

In pratica, invece che massimizzare $L(\vartheta)$, spesso risulta più conveniente minimizzare l'opposto del logaritmo della funzione di verosimiglianza, in questo caso si ha:

$$\hat{\vartheta} = arg \ \min_{\vartheta} \left(- \ln \left(L(\vartheta) \right) \right).$$

Stima bayesiana

Si consideri la funzione di densità $p(z, \vartheta)$, tale funzione dipende da un parametro vettoriale ϑ il quale, nell'ottica classica, è una quantità costante non nota che deve essere stimata utilizzando solamente l'informazione contenuta nell'insieme Z delle osservazioni. A volte però, il parametro ϑ non è totalmente sconosciuto e sarebbe utile sfruttare l'informazione su di esso disponibile.

Nell'ottica bayesiana, il parametro ϑ è visto come una variabile casuale $\boldsymbol{\vartheta}$ e la densità di probabilità di $\mathbf{z}$ è interpretata come una densità condizionata $p(\mathbf{z}|\boldsymbol{\vartheta})$, dove si sia assunto $\boldsymbol{\vartheta} = \vartheta$. L'informazione a priori disponibile sul parametro ϑ viene usata per assegnare in qualche modo alla variabile casuale $\boldsymbol{\vartheta}$ una densità di probabilità a priori $p_{\boldsymbol{\vartheta}}(\vartheta)$ che, attraverso il teorema di Bayes, viene convertita nella relativa densità a posteriori $p_{\boldsymbol{\vartheta}}(\vartheta|Z)$ utilizzando le osservazioni disponibili Z.

La statistica bayesiana trasforma quindi il problema della stima di un parametro costante non noto ϑ, in un problema di predizione del valore assunto da una variabile casuale $\boldsymbol{\vartheta}$.

Il teorema di Bayes può essere scritto nella seguente forma:

$$p(\vartheta|Z) = \frac{p(Z|\vartheta)p(\vartheta)}{p(Z)}.$$

Essendo le osservazioni Z estratte indipendentemente, risulta

$$p(Z|\vartheta) = \prod_{i=1}^{n} p(z_i|\vartheta),$$

quindi

$$p(\vartheta|Z) = \frac{p(\vartheta)}{p(Z)} \prod_{i=1}^{n} p(z_i|\vartheta).$$

Dato l'insieme di osservazioni Z, l'espressione finale della densità di probabilità $p(\mathbf{z})$ è ottenuta integrando su tutti i possibili valori del parametro ϑ, la densità di probabilità di $\mathbf{z}$ condizionata a $\boldsymbol{\vartheta}$, pesata dalla densità a posteriori di $\boldsymbol{\vartheta}$. Formalmente risulta:

$$p(\mathbf{z}) = \int p(\mathbf{z}, \vartheta|Z) \, \mathrm{d}\vartheta = \int p(\mathbf{z}|\vartheta)p(\vartheta|Z) \, \mathrm{d}\vartheta.$$

Tipicamente la soluzione analitica di tale integrale è molto complessa ed è possibile solo per quella classe di funzioni di densità di probabilità per le quali la densità a posteriori ha la stessa forma analitica di quella a priori. La distribuzione gaussiana appartiene a tale classe.

1.2.2 Stime non parametriche

L'espressione *stima non parametrica* si riferisce al fatto che il modello della funzione di densità di probabilità da stimare, non appartiene ad una famiglia parametrica definita a priori indipendentemente dall'insieme Z, ma dipende strettamente da quest'ultimo.

Sia $\mathbf{z}$ una variabile casuale con densità di probabilità $p(\mathbf{z})$ non nota. Si consideri la regione r definita nello spazio della variabile $\mathbf{z}$. La probabilità che un valore z estratto dalla distribuzione $p(\mathbf{z})$ cada all'interno della regione r è:

$$Pr = prob\{z \in r\} = \int_r p(z)\,\mathrm{d}z.$$

Si consideri un insieme di osservazioni $Z = \{z_1, z_2, \ldots, z_n\}$ estratte indipendentemente dalla densità di probabilità $p(\mathbf{z})$, e sia $\mathbf{k}$ la variabile casuale che rappresenta il numero di osservazioni appartenenti a Z che cadono all'interno della regione r. La densità di probabilità di $\mathbf{k}$ è data da:

$$\pi(k) = \frac{n!}{k!(n-k)!} Pr^k (1 - Pr)^{n-k}.$$

La variabile casuale $\mathbf{k}/n$ è tale che:

$$E[\mathbf{k}/n] = Pr,$$

e

$$E[(\mathbf{k}/n - Pr)^2] = \frac{Pr(1 - Pr)}{n}.$$

La varianza di $\mathbf{k}/n$ converge a zero per $n \to \infty$. E' ragionevole quindi attendersi che una buona stima della probabilità Pr possa essere fornita dal rapporto k/n, dove k è il numero di punti appartenenti a Z che cadono all'interno della regione r. Formalmente:

$$Pr \cong \frac{k}{n}. \tag{1.1}$$

Se si assume che la densità di probabilità $p(\mathbf{z})$ sia continua e che non vari apprezzabilmente all'interno della regione r, è possibile approssimare la probabilità Pr con:

$$Pr = \int_r p(\mathbf{z})\,\mathrm{d}\mathbf{z} \cong p(\mathbf{z})V,$$

dove V è il volume della regione r. Per quei valori di $\mathbf{z}$ appartenenti alla regione r, la densità di probabilità $p(\mathbf{z})$ risulta quindi approssimabile con:

$$p(\mathbf{z}) \cong \frac{k}{nV}. \tag{1.2}$$

L'approssimazione dell'equazione (1.1) richiede che la regione r considerata sia grande, in modo tale che contenga un numero significativo di osservazioni; allo stesso tempo, affinché l'approssimazione dell'equazione (1.2) non introduca distorsioni, è auspicabile che r sia piccola in maniera tale da assicurare che all'interno della regione stessa la densità di probabilità $p(\mathbf{z})$ non vari apprezzabilmente. Queste due condizioni chiaramente contrastanti, impongono la necessità di una scelta di compromesso per quanto riguarda la dimensione della regione r. In letteratura esistono due approcci duali alla stima non parametrica:

Parzen Windows. Seguendo questo approccio, viene fissato a priori il volume V; il numero di osservazioni che cadono all'interno della regione r dipende dalle caratteristiche dell'insieme Z.

k-Nearest Neighbors. Consiste nel fissare il numero di osservazioni che si vuole siano contenute all'interno della regione r; il volume V della regione stessa viene fatto variare di conseguenza.

Parzen Windows

Si consideri l'ipercubo di lato h centrato nel punto z. Il volume di tale ipercubo risulta essere

$$V = h^d,$$

dove d è la dimensione dello spazio vettoriale a cui z appartiene. Si definisca la funzione kernel (Parzen Window) come:

$$H(u) = \begin{cases} 1 & \text{se } |u^{(j)}| < 1/2 \text{ per } j = 1, \ldots, d, \\ 0 & \text{altrimenti;} \end{cases}$$

dove $u^{(j)}$ è la j-esima componente del vettore u. Risulta quindi che la quantità

$$H\left(\frac{z - z_i}{h}\right)$$

ha valore uno per tutte e sole quelle osservazioni appartenenti all'insieme $Z = \{z_1, z_2, \ldots, z_n\}$ che si trovano all'interno dell'ipercubo di lato h e centrato nel punto z. Il numero delle osservazioni interne al volume V è dato

da:

$$k = \sum_{i=1}^{n} H\left(\frac{z - z_i}{h}\right).$$

La stima del valore della densità di probabilità $p(\mathbf{z})$ nel punto z, fornita dal metodo Parzen Windows è data da:

$$\hat{p}_{pw}(z) = \frac{\sum_{i=1}^{n} H\left(\frac{z - z_i}{h}\right)}{nh^d},$$

dove come già visto, k è numero di osservazioni interne all'ipercubo, n è il numero totale di osservazioni disponibili e h, parametro che caratterizza il metodo, è il lato dell'ipercubo considerato.

La stima così ottenuta è discontinua sullo spazio della variabile $\mathbf{z}$; per ottenere una stima *smooth* è possibile utilizzare funzioni kernel definite in maniera differente, purché soddisfino i seguenti requisiti:

$$H(u) \geq 0 \qquad \forall u,$$

e

$$\int H(u)\,\mathrm{d}u = 1.$$

In letteratura è comune l'utilizzo come funzione kernel della gaussiana, funzione che gode della proprietà di essere continua con tutte le sue derivate.

L'approccio Parzen Windows presenta i seguenti difetti:

- lo stimatore ottenuto è polarizzato
- questo metodo, per operare, necessita di tutto l'insieme dei dati contemporaneamente. Ciò crea dei problemi pratici e rende il metodo inefficiente quando si debbano trattare insiemi di osservazioni di grandi dimensioni.

K-Nearest Neighbors

Si consideri un'ipersfera centrata nel punto z. Si faccia crescere il raggio fintanto che k osservazioni appartenenti all'insieme Z cadano nella regione di spazio delimitato dall'ipersfera stessa. La stima fornita dal metodo Nearest Neighbors del valore della densità di probabilità $p(\mathbf{z})$ nel punto z, è data da:

$$\hat{p}_{nn}(z) = \frac{k}{nV},$$

dove k è il valore del parametro che caratterizza il metodo, n è il numero totale di osservazioni disponibili e V è il volume dell'ipersfera considerata.

Gli svantaggi associati a questo metodo sono i seguenti:

- formalmente la stima fornita non è una probabilità: il suo integrale esteso a tutto lo spazio della variabile $\mathbf{z}$ non è uguale a uno ma diverge.
- come il metodo Parzen Windows, anche il metodo Nearest Neighbors necessita dell'intero insieme Z e presenta quindi gli stessi problemi pratici per insiemi delle osservazioni di grandi dimensioni.

1.2.3 Stime semi parametriche

I due approcci alla stima di una densità fin ora introdotti, quello parametrico e quello non parametrico, presentano vantaggi e svantaggi. In questo paragrafo presenteremo l'approccio semi parametrico che intende combinare i vantaggi dei due metodi visti.

Seguendo l'approccio parametrico, il modello usato appartiene ad una determinata classe di funzioni che in generale potrebbe non contenere nessuna istanza adatta a modellizzare la densità di probabilità in esame. Dal punto di vista computazionale però, i metodi parametrici sono molto efficienti, sia per quanto riguarda l'identificazione che per quanto riguarda la valutazione del modello.

L'approccio non parametrico consente di prendere in considerazione modelli molto più generali che non l'approccio parametrico. Ad ogni modo, un modello non parametrico è inefficiente in quanto richiede l'utilizzo dell'intero insieme delle osservazioni per ogni valutazione del modello stesso.

L'approccio semi parametrico è tale per cui il numero dei parametri, cioè la complessità del modello, non è funzione della dimensione dell'insieme delle osservazioni ma è determinata dalle caratteristiche della densità di probabilità che deve essere modellizzata. I modelli semi parametrici dunque sono in grado di adattarsi al problema specifico in esame.

Lo svantaggio di questo approccio è che l'identificazione di un modello semi parametrico è tipicamente più complessa e costosa in termini di tempo di quella di un modello parametrico o non parametrico.

Come abbiamo visto, nell'approccio Parzen Windows il modello della funzione densità di probabilità è ottenuto come combinazione lineare di n funzioni kernel, ciascuna centrata su una delle n osservazioni. Il modello che presentiamo ora è anch'esso costituito dalla combinazione lineare di funzioni ma in questo caso, il numero m di tali funzioni è un parametro del modello e, tipicamente, è molto inferiore a n. Il modello semi parametrico che consideriamo prende il nome di *mixture of density* e ha la seguente forma:

$$p(\mathbf{z}) = \sum_{j=1}^{m} p(\mathbf{z}|j)\pi(j), \tag{1.3}$$

dove $p(\mathbf{z}|j)$ è il j-esimo contributo alla densità totale e $\pi(j)$ è il relativo coefficiente di combinazione.

L'equazione (1.3) può essere interpretata come la densità di probabilità di un insieme di dati appartenenti a classi diverse, caratterizzate ciascuna da una diversa funzione di densità. Seguendo questa interpretazione, il coefficiente $\pi(j)$ può essere visto come la probabilità *a priori* che una determinata osservazione z_i appartenga alla classe j-esima. I coefficienti $\pi(j)$, essendo probabilità, devono soddisfare le seguenti condizioni:

$$\sum_{j=1}^{m} \pi(j) = 1$$

e

$$0 \leq \pi(j) \leq 1.$$

Analogamente, la funzione $p(\mathbf{z}|j)$ può essere vista come la densità di probabilità associata alla classe j-esima, deve quindi soddisfare la seguente condizione:

$$\int p(\mathbf{z}|j)\, \mathrm{d}\mathbf{z} = 1. \tag{1.4}$$

Dato questo schema interpretativo, l'insieme dei dati $Z = \{z_1, z_2, \ldots, z_n\}$ è visto come se la singola osservazione z_i fosse estratta dalla densità condizionata $p(\mathbf{z}|j)$, dove j è, a sua volta, estratta dalla densità discreta $\pi(j)$. Una proprietà importante della combinazione di densità (1.3) è che, per un gran numero di differenti funzioni $p(\mathbf{z}|j)$, tale modello è in grado di approssimare una qualsiasi funzione di densità con accuratezza arbitraria, purché il numero di componenti m sia sufficientemente grande e i coefficienti siano scelti in maniera corretta.

1.3 Teoria dell'apprendimento

Lo schema generale di un processo di apprendimento da esempi, schema cui faremo riferimento in questa tesi, è quello proposto da Vapnik. Tale schema contempla tre elementi:

- Un generatore (G) di vettori casuali $\mathbf{x} \in \Re^d$, estratti indipendentemente da una funzione di densità di probabilità $p(\mathbf{x})$ fissa ma non nota.
- Un supervisore (S) che, dato un vettore di ingresso x, restituisce un valore di uscita $\mathbf{y} \in \Re$ secondo la funzione di densità condizionata $p(\mathbf{y}|x)$ fissa ma non nota.
- Una macchina (M) che è in grado di implementare un insieme di funzioni $f(x, \alpha)$ con $\alpha \in \Lambda$, dove Λ è un insieme di parametri[2].

Nel contesto così definito, diremo che la macchina (M) apprende il comportamento del supervisore (S) se è in grado di selezionare tra le possibili funzioni $f(x, \alpha)$ quella che meglio approssima il comportamento del supervisore (S). La selezione di tale funzione avviene sulla base di un *training set*:

$$Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}.$$

costituito da n osservazioni di coppie ingresso-uscita, estratte indipendentemente dalla densità congiunta

$$p(x, y) = p(x)p(y|x),$$

fissa ma non nota.

Per selezionare la funzione approssimante viene misurata la discrepanza tra la risposta y del supervisore ad un dato ingresso x, e la risposta $f(x, \alpha)$ fornita dalla macchina (M). Tale discrepanza è convenientemente rappresentata in termini matematici dalla *funzione di costo*:

$$C\big(y, f(x, \alpha)\big).$$

Di tale funzione si considera il valore atteso che prende il nome di *funzionale di rischio*:

$$R(\alpha) = \iint C\big(y, f(x, \alpha)\big)p(x, y)\,\mathrm{d}x\,\mathrm{d}y.$$

L'obiettivo di un processo di apprendimento è trovare, tra le possibili funzioni $f(x, \alpha)$ con $\alpha \in \Lambda$, la funzione $f(x, \alpha_0)$ che minimizza il funzionale $R(\alpha)$ nella

[2]Gli elementi $\alpha \in \Lambda$ sono vettori di dimensione non definita a priori. Questo consente di considerare con la notazione $f(x, \alpha)$, un qualsiasi insieme di funzioni.

condizione in cui la funzione di densità congiunta $p(x, y)$ non è nota e l'unica sorgente di informazione disponibile è il training set Z.

La formulazione della teoria dell'apprendimento che abbiamo presentato sopra è decisamente generale e prende in considerazione diversi problemi specifici. Tra questi, i due che maggiormente si sono imposti all'attenzione della comunità scientifica sono il problema della *classificazione* e il problema della stima di *regressione*.

Classificazione

Nel seguito di questa tesi ci occuperemo prevalentemente di regressione. In questo paragrafo ci limitiamo quindi a presentare brevemente il problema della classificazione, nell'intenzione di mostrare come esso possa essere affrontato in maniera simile al problema della regressione, impiegando gli stessi strumenti che svilupperemo per trattare quest'ultimo.

Il problema della classificazione può essere presentato nel seguente modo: il supervisore (S) osserva il valore di un vettore casuale di ingresso x e determina a quale classe, tra k disponibili, tale ingresso appartenga. Si assuma per semplicità che sia $k = 2$, in questo caso la risposta del supervisore può assumere solo due valori: $y = [0, 1]$. Di conseguenza, anche le funzioni $f(x, \alpha)$ implementate dalla macchina (M), saranno *funzioni indicatore* (funzioni che possono assumere solo valore zero o uno).

Risulta naturale considerare come funzione di costo, una funzione che assuma valore uno in caso di errore di classificazione e valore zero un caso di classificazione corretta. Formalmente:

$$C\big(y, f(x, \alpha)\big) = \begin{cases} 0 & \text{se } y = f(x, \alpha), \\ 1 & \text{se } y \neq f(x, \alpha). \end{cases}$$

Il funzionale di rischio, in questo caso, rappresenta la probabilità che la macchina (M) commetta un errore di classificazione.

Regressione

Si consideri la relazione stocastica tra due variabili x e y tale per cui, dato un vettore x, il valore di y sia determinato come risultato di un esperimento casuale caratterizzato dalla funzione di densità condizionata $p(y|x)$. La relazione stocastica così descritta, potrà dirsi completamente nota solo se si conosce la funzione di densità condizionata $p(y|x)$ il che è equivalente alla conoscenza

degli infiniti momenti della funzione $p(y|x)$ stessa[3]. Spesso è possibile acquisire solo una conoscenza parziale su una data relazione stocastica, in questo caso la caratteristica della funzione $p(y|x)$ che risulta essere più informativa è il suo valore atteso cioè il suo primo momento:

$$E[y|x] = \int yp(y|x)\,\mathrm{d}y.$$

La funzione deterministica che, data una variabile x, fornisce $E[y|x]$, valore atteso della variabile y condizionato a x, prende il nome di *funzione di regressione* di y su x.

L'interesse che merita la funzione di regressione è dovuto al fatto che tale funzione è quella funzione che minimizza il funzionale di rischio (1.3) quando la funzione di costo è il quadrato dell'errore di predizione:

$$C\big(y, f(x,\alpha)\big) = \big(y - f(x,\alpha)\big)^2. \tag{1.5}$$

Dimostrazione. Per una dimostrazione di questa affermazione si riscriva la funzione di costo (1.5) come segue:

$$\begin{aligned}\big(y - f(x,\alpha)\big)^2 &= \big(y - E[y|x] + E[y|x] - f(x,\alpha)\big)^2 \\ &= \big(y - E[y|x]\big)^2 + \big(E[y|x] - f(x,\alpha)\big)^2 + \\ &\quad + 2\big(y - E[y|x]\big)\big(E[y|x] - f(x,\alpha)\big).\end{aligned} \tag{1.6}$$

Il funzionale di rischio risulta dunque essere

$$\begin{aligned}R(\alpha) = &\iint \big(y - E[y|x]\big)^2 p(x,y)\,\mathrm{d}x\,\mathrm{d}y + \\ &+ \int \big(E[y|x] - f(x,\alpha)\big)^2 p(x)\,\mathrm{d}x.\end{aligned} \tag{1.7}$$

in quanto l'integrale rispetto a y del terzo termine della (1.6) è nullo.

Nell'equazione (1.7), il primo dei due addendi non dipende da α, quindi il minimo di $R(\alpha)$ coincide con il minimo del secondo addendo. Da questo segue che la funzione $f(x,\alpha_0)$ che minimizza il funzionale di rischio è:

$$f(x,\alpha_0) = E[y|x].$$

□

[3]Il momento di ordine k della variabile casuale z, caratterizzata dalla densità di probabilità $p(z)$ è dato dalla seguente:

$$m_k = E[z^k] = \int_{-\infty}^{+\infty} z^k p(z)\,\mathrm{d}z.$$

1.3.1 Minimizzazione del funzionale di rischio e rischio empirico

Come abbiamo mostrato nei due paragrafi precedenti, sia il problema della classificazione che il problema della stima della funzione di regressione possono essere ricondotti alla minimizzazione del funzionale di rischio

$$R(\alpha) = \iint C\big(y, f(x,\alpha)\big) p(x,y)\,\mathrm{d}x\,\mathrm{d}y,$$

nella situazione in cui la distribuzione di probabilità $p(x,y)$ non è nota ma è disponibile un training set

$$Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\},$$

composto da n osservazioni.

Non essendo nota la funzione di probabilità congiunta $p(x,y)$ non è possibile trovare analiticamente la funzione $f(x,\alpha_0)$ che minimizza il funzionale di rischio $R(\alpha)$, è quindi necessario trovare empiricamente, usando il training set disponibile, una funzione che approssimi $f(x,\alpha_0)$. Una possibile soluzione, forse la più intuitiva, può essere ottenuta utilizzando quello che nella teoria della probabilità e noto come *principio di massima entropia* (Papoulis, 1991). In questo contesto, tale principio è utilizzato per approssimare la distribuzione continua di probabilità $p(x,y)$ con la funzione impulsiva $p_{emp}(x,y)$ definita, a partire dal training set Z, nella seguente maniera:

$$p_{emp}(x,y) = \sum_{i=1}^{n} \frac{1}{n} \delta(x - x_i, y - y_i).$$

Sostituendo tale funzione all'interno del funzionale di rischio si ottiene il cosiddetto *funzionale di rischio empirico*:

$$R_{emp}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} C\big(y_i, f(x_i,\alpha)\big).$$

Si approssima dunque la funzione $f(x,\alpha_0)$ con la funzione $f(x,\alpha_n)$, estremante del funzionale di rischio empirico applicando quello che prende il nome di *Principio di Minimizzazione del Rischio Empirico*). E' interessante notare che, sostituendo all'interno del funzionale di rischio empirico la funzione di costo

$$C\big(y, f(x,\alpha)\big) = \big(y - f(x,\alpha)\big)^2,$$

definita per il problema della stima di regressione, si ottiene la stessa equazione del metodo dei minimi quadrati, cioè:

$$R_{emp}(\alpha) = \frac{1}{n} \sum_{i=1}^{n} \big(y_i - f(x_i,\alpha)\big)^2.$$

1.3.2 Generalizzazione

Diremo che una macchina (M) è in grado di generalizzare se, avendo visto un training set Z generato da un supervisore (S), è in grado di approssimare il comportamento di (S) anche in situazioni non contemplate in Z. La minimizzazione del funzionale di rischio garantirebbe la capacità di generalizzare però, come abbiamo visto nel paragrafo precedente, nel processo di apprendimento la funzione non nota $p(x, y)$ viene sostituita dalla funzione $p_{emp}(x, y)$ quindi in ultima analisi, la funzione estremante $f(x, \alpha_0)$ del funzionale di rischio viene approssimata con la funzione $f(x, \alpha_n)$, dove α_n è il valore ottimo del parametro, identificato utilizzando il training set disponibile e composto da n esempi. Questa approssimazione consente di riprodurre il comportamento del supervisore (S) per quanto riguarda gli esempi presenti nel training set, ma non garantisce nulla sulla capacità di previsione in punti lontani da quelli.

VC dimension

In questo paragrafo introdurremo alcuni aspetti della teoria della generalizzazione, teoria che si propone di individuare un insieme di condizioni che, qualora verificate, consentano di asserire la legittimità dell'approssimazione di $f(x, \alpha_0)$ attraverso $f(x, \alpha_n)$, garantendo dunque che una macchina che apprende minimizzando il funzionale di rischio empirico, è in grado di generalizzare correttamente. Formalmente, il *principio di minimizzazione del rischio empirico* si dice essere *consistente* riguardo a un insieme di funzioni $Q(z, \alpha) = C\big(y, f(x, \alpha)\big)$, per $\alpha \in \Lambda$ e per la densità congiunta $p(z) = p(x, y)$, se le due seguenti sequenze convergono in probabilità allo stesso limite:

$$R(\alpha_n) \xrightarrow[n\to\infty]{p(z)} \inf_{\alpha\in\Lambda} R(\alpha), \tag{1.8}$$

$$R_{emp}(\alpha_n) \xrightarrow[n\to\infty]{p(z)} \inf_{\alpha\in\Lambda} R(\alpha), \tag{1.9}$$

dove con α_n si è indicato il valore ottimo del parametro identificato da un training set Z composto da n esempi e dove $\inf_{\alpha\in\Lambda} R(\alpha)$ indica il minor valore possibile assumibile dal funzionale di rischio per $\alpha \in \Lambda$.

Il teorema chiave della teoria di apprendimento è il seguente teorema dovuto a Vapnik e Chervonenkis (Vapnik & Chervonenkis, 1991):

Teorema 2. *Sia $Q(z, \alpha)$, con $\alpha \in \Lambda$, un insieme di funzioni che soddisfino la condizione*

$$A \le \int Q(z, \alpha) p(z)\, \mathrm{d}z \le B \qquad (A \le R(\alpha) \le B).$$

Condizione necessaria e sufficiente affinché il principio di minimizzazione del rischio empirico sia consistente è che $R_{emp}(\alpha)$ converga uniformemente al

valore vero $R(\alpha)$ *sull'insieme* $Q(z,\alpha)$, *con* $\alpha \in \Lambda$, *nel seguente senso:*

$$\lim_{n\to\infty} prob\Big\{ \sup_{\alpha\in\Lambda} \big(R(\alpha) - R_{emp}(\alpha)\big) > \epsilon \Big\} = 0. \quad \forall \epsilon > 0. \tag{1.10}$$

Il teorema chiave dell'apprendimento afferma quindi che la consistenza del principio di minimizzazione del rischio empirico è equivalente all'esistenza della *convergenza uniforme* di $R_{emp}(\alpha)$ a $R(\alpha)$ descritta dalla (1.10).

E' interessante qui sottolineare che il teorema così enunciato asserisce che la consistenza del principio di minimizzazione del rischio empirico è determinata dalla *peggiore* funzione, nel senso specificato dalla (1.10), appartenente all'insieme $Q(z,\alpha)$, con $\alpha \in \Lambda$ (*worst-case analysis*).

Per costruire una teoria sulla capacità di generalizzazione di una macchina (M) viene introdotto il concetto di *VC dimension*. La *VC dimension* di un insieme di funzioni è uno scalare h che rappresenta la complessità della macchina (M) stessa[4]. Un importante risultato teorico che riguarda la *VC dimension* è che se h risulta finito per l'insieme di funzioni $Q(z,\alpha)$, con $\alpha \in \Lambda$, la convergenza uniforme di $R_{emp}(\alpha)$ a $R(\alpha)$ è data e dunque il principio di minimizzazione del rischio empirico è consistente.

Inoltre, nel caso non asintotico cioè quando n è finito, si dimostra che, per un insieme di funzioni $Q(z,\alpha)$ tali per cui $0 \le Q(z,\alpha) \le B$, con $\alpha \in \Lambda$ vale la seguente:

$$R(\alpha_n) \le R_{emp}(\alpha_n) + \frac{B\varepsilon}{2}\left(1 + \sqrt{1 + \frac{4Remp(\alpha_n)}{B\varepsilon}}\right), \tag{1.11}$$

dove, se l'insieme di funzioni $Q(z,\alpha_l)$, con $l = 1,\ldots,N$ è composto da N elementi si ha:

$$\varepsilon = 2\,\frac{\ln N - \ln \eta}{n}, \tag{1.12}$$

quindi ε non è funzione di h. Mentre se l'insieme delle funzioni $Q(z,\alpha_l)$ è composto da infiniti elementi, ed è caratterizzato da una *VC dimension*, risulta:

$$\varepsilon = 4\,\frac{h\left(\ln\frac{2n}{h} + 1\right) - \ln(\eta/4)}{n}. \tag{1.13}$$

La disuguaglianza (1.11) è valida almeno con probabilità $1-\eta$. Il principio di minimizzazione del rischio empirico è da ritenersi valido quando sia disponibile un numero elevato n di campioni, tipicamente $n/h > 20$. Se ciò è verificato, il coefficiente ε risulta piccolo e quindi il secondo termine di destra nella disuguaglianza (1.11) risulta piccolo. In tal caso il valore reale del rischio è vicino al valore del rischio empirico.

[4]Nel caso particolare in cui le funzioni considerate appartengano ad una famiglia parametrica lineare, la *VC dimension* è il numero di parametri liberi.

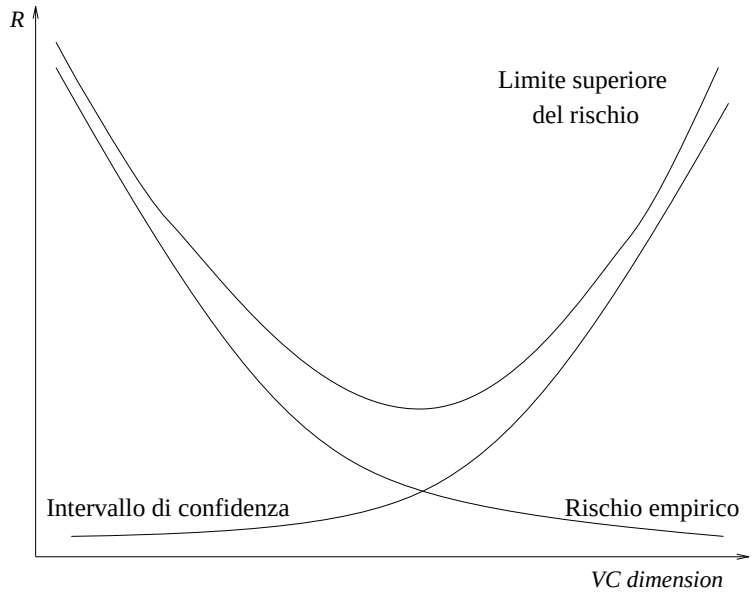


Figura 1.1: Al crescere della *VC dimension* diminuisce il valore del rischio empirico ma aumenta l'intervallo di confidenza. Il limite superiore del valore vero del rischio è dato dalla somma dei due contributi.

Nel caso in cui n/h sia piccolo, un basso valore di $R_{emp}(\alpha_n)$ non garantisce un basso valore del vero funzionale di rischio. In tal caso il secondo termine di destra della disuguaglianza (1.11) non è trascurabile e va considerato come un *intervallo di confidenza* sulla capacità del rischio empirico di fornire un'indicazione del valore vero del funzionale di rischio (figura 1.1).

Dilemma bias/varianza

Un'altra analisi, in un certo senso parallela a quella fornita da Vapnik, che consente di cogliere molti aspetti del problema dell'apprendimento è quella nota in letteratura con il nome di *bias/variance decomposition* (Geman *et al.*, 1992). L'essenza di questa analisi consiste nel mostrare come l'errore di predizione commesso da un modello $f(x, \alpha)$ sia da ricondurre a due componenti. La prima, *bias*, è il valore atteso dell'errore che commette un modello appartenente alla famiglia $f(x, \alpha)$ addestrato con un generico training set Z. La seconda, *variance*, è la varianza dell'errore commesso rispetto a tutti i possibili training set. Riportiamo di seguito il semplice ed elegante ragionamento così come presentato dagli autori Geman, Bienenstock e Doursat.

Per rendere esplicita la dipendenza del modello identificato dal training set Z, useremo la notazione $f(x, \alpha_Z)$ al posto di $f(x, \alpha)$. Consideriamo in principio un particolare training set Z; un indice della bontà della stima della funzione non nota fornita da $f(x, \alpha_Z)$, è data dal valore atteso dell'errore

quadratico:

$$E\left[\big(y - f(x, \alpha_Z)\big)^2 | x, Z\right], \tag{1.14}$$

dove con l'operatore $E[\cdot]$ si intende qui il valore atteso rispetto alla densità congiunta $p(x, y)$. Ricordando la (1.6), possiamo scrivere:

$$E\left[\big(y - f(x, \alpha_Z)\big)^2 | x, Z\right] = E\left[\big(y - E[y|x]\big)^2 | x, Z\right] + \\ + \big(f(x, \alpha_Z) - E[y|x]\big)^2. \tag{1.15}$$

Si noti che $E\left[\big(y - E[y|x]\big)^2 | x, Z\right]$ non dipende dal training set Z ma è semplicemente la varianza di y dato x. Quindi la bontà della predizione fornita da $f(x, \alpha_Z)$ può essere misurata semplicemente dal quadrato della distanza della predizione dalla funzione di regressione, cioè da:

$$\big(f(x, \alpha_Z) - E[y|x]\big)^2. \tag{1.16}$$

Considerando ora tutti i possibili training set, l'indice della bontà della stima fornita dal modello in esame è data dal valore atteso rispetto ai possibili training set della (1.16), ciò da:

$$E_Z\left[\big(f(x, \alpha_Z) - E[y|x]\big)^2\right]. \tag{1.17}$$

Può verificarsi il caso che per un particolare training set Z, $f(x, \alpha_Z)$ sia un ottimo approssimatore di $E[y|x]$, quindi il miglior predittore di y nell'ottica dei minimi quadrati. Allo stesso tempo poù essere che, per altre realizzazioni di Z, $f(x, \alpha_Z)$ non consenta prestazioni altrettanto buone e che quindi la qualità della predizione vari fortemente al variare del particolare training set considerato. La quantità (1.17) può essere scomposta come segue:

$$\begin{aligned}
&E_Z\Big[\big(f(x, \alpha_Z) - E[y|x]\big)^2\Big] = \\
&\quad = E_Z\Big[\Big(\big(f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\big) + \big(E_Z[f(x, \alpha_Z)] - E[y|x]\big)\Big)^2\Big] \\
&\quad = E_Z\Big[\big(f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\big)^2\Big] + E_Z\Big[\big(E_Z[f(x, \alpha_Z)] - E[y|x]\big)^2\Big] + \\
&\qquad + 2\, E_Z\Big[\big(f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\big)\big(E_Z[f(x, \alpha_Z)] - E[y|x]\big)\Big] \\
&\quad = E_Z\Big[\big(f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\big)^2\Big] + \big(E_Z[f(x, \alpha_Z)] - E[y|x]\big)^2 + \\
&\qquad + 2\, E_Z\Big[f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\Big]\big(E_Z[f(x, \alpha_Z)] - E[y|x]\big) \\
&\quad = \big(E_Z[f(x, \alpha_Z)] - E[y|x]\big)^2 + \qquad \textit{“bias”} \\
&\qquad + E_Z\Big[\big(f(x, \alpha_Z) - E_Z[f(x, \alpha_Z)]\big)^2\Big] \qquad \textit{“variance”}
\end{aligned}$$

La componente di *bias* è tanto più grande tento più $f(x, \alpha_Z)$ si scosta in media da $E[y|x]$. Anche se la componente di *bias* è piccola, uno stimatore $f(x, \alpha_Z)$ può essere affetto da una grande componente di *variance* ed essere cioè molto sensibile a cambiamenti nella composizione del training set.

Tornando al problema dell'apprendimento supervisionato, la conclusione che si deve trarre dall'analisi *bias/variance* è che la scelta della struttura della funzione $f(x, \alpha)$ da adottare per apprendere il comportamento di un supervisore, quando è disponibile una quantità scarsa di esempi del comportamento stesso, è sempre una scelta di compromesso. Funzioni più complesse e quindi più flessibili e generali hanno la possibilità di cogliere il comportamento del supervisore e quindi portano ad un errore in cui la componente di *bias* è trascurabile. Essendo però il training set composto da pochi esempi, l'approssimazione fornita è tipicamente molto sensibile alla particolare istanza del training set disponibile: questo porta ad una elevata componente di *variance*. Viceversa, funzioni descritte da una semplice struttura parametrica, rischiano di non essere sufficientemente potenti da cogliere il comportamento del supervisore e quindi sono tipicamente affette da *bias*, mentre la componente di *variance* risulta più limitata in quanto anche a partire da training set diversi, il valore dei pochi parametri della funzione possono essere identificati con un basso grado di incertezza. Al crescere della complessità degli approssimatori considerati, dato un limitato insieme di esempi disponibile per l'identificazione degli stessi, tipicamente si osserva una diminuzione della componente di *bias* ma contemporaneamente un aumento di quella di *variance*. Si ritrova quindi un risultato *qualitativamente* simile a quello a cui si giunge attraverso l'analisi riguardo alla *VC dimension* (figura 1.2).

Dato un fenomeno, il compito dell'analista è quello di introdurre vincoli sulla funzione $f(x, \alpha)$, utilizzando conoscenza a priori, con l'obiettivo di ridurre la componente di *variance*, senza aumentare quella di *bias*. In pratica spesso si presuppone un certo grado di *smoothness* del fenomeno in esame e di conseguenza si introduce un pari vincolo di *smoothness* per la funzione $f(x, \alpha)$. Se il fenomeno considerato possiede realmente la proprietà supposta, il vincolo introdotto limita la componente di *variance* senza introdurre *bias*. Se non la possedesse, dettagli del fenomeno in esame, quali picchi e valli, verrebbero persi e ciò introdurrebbe una componente di *bias*.

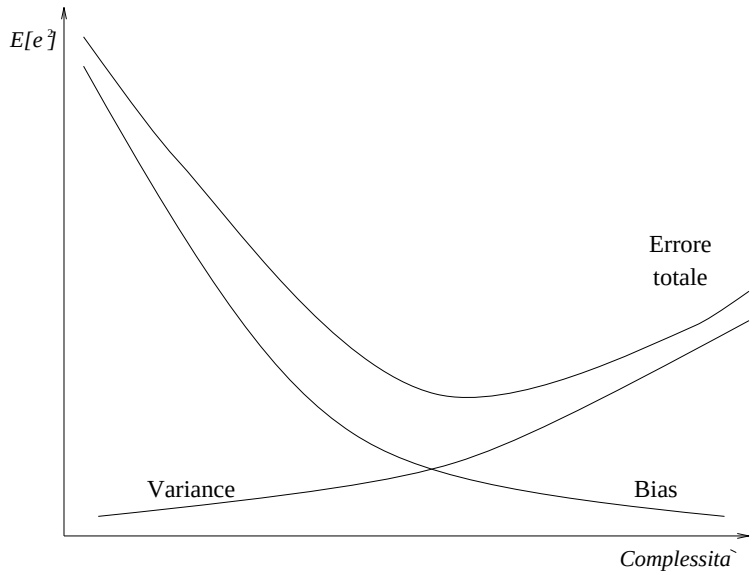


Figura 1.2: Il valore atteso del quadrato dell'errore può essere scomposto in una componente di *bias* e in una di *variance*. Aumentando la complessità diminuisce il *bias* ma aumenta la componente di *variance*.

Capitolo 2

L'apprendimento in Pratica

In questo capitolo esamineremo da un punto di vista pratico i vari passi che portano dall'osservazione di un fenomeno, fino all'acquisizione della capacità di riprodurlo e di predirne l'evoluzione.

Un processo di apprendimento non procede in modo sequenziale, bensì come un processo ciclico che ripete successivamente più fasi fino a giungere, attraverso raffinamenti successivi, ad una rappresentazione soddisfacente del fenomeno in esame (Murray-Smith, 1994). I vari paragrafi di questo capitolo non vogliono quindi definire una sequenza temporale rigida di operazioni che devono essere eseguite sempre e comunque. Si propongono piuttosto di presentare le fasi logiche di un processo di apprendimento mostrando, per ciascuna di queste, i problemi che presentano e le più comuni delle soluzioni adottate in pratica per affrontarli.

2.1 Dal fenomeno ai dati

2.1.1 Osservazione del fenomeno

Un processo di apprendimento inizia con una fase di osservazione e analisi il cui obiettivo è quello di individuare una serie di quantità misurabili che siano in grado di fornire una descrizione completa del fenomeno in esame.

In questa fase la conoscenza disponibile è molto limitata e non consente quindi una selezione corretta di quell'insieme di variabili necessarie e sufficienti per rappresentare in maniera univoca il fenomeno. Tipicamente, si cercherà di misurare il maggior numero possibile[1] di variabili, demandando la selezione di quelle significative a fasi successive del processo, e riservandosi di ritornare a questa fase per raccogliere misure di altre quantità, qualora quelle misurate si rivelassero non sufficienti.

[1] In pratica, un limite alla quantità di informazione raccoglibile è imposto da considerazioni economiche sul costo delle misure.

La fase di osservazione si conclude fornendo, come risultato parziale, un insieme di *dati grezzi* che costituiscono la prima forma di rappresentazione possibile di un fenomeno.

Nonostante la sua apparente banalità, questa è una fase decisamente delicata durante la quale viene introdotta conoscenza a priori, a volte non fondata, sul fenomeno.

La criticità della fase di osservazione e di raccolta dei dati, è dovuta al fatto che una selezione non accurata delle variabili da registrare e quindi l'esclusione di variabili significative, rende impotente qualsiasi metodo di apprendimento. La mancanza di una variabile può risultare molto costosa in quanto tipicamente rilevabile solo in una fase avanzata del processo di apprendimento.

2.1.2 Preprocessing

Durante la fase di *preprocessing*, i dati grezzi ottenuti dall'osservazione del fenomeno, vengono trattati e preparati per essere utilizzati nella fase di addestramento del modello. Questa fase non è sempre necessaria anche se è quasi sempre raccomandabile.

Il tipo di operazioni da effettuare in fase di preprocessing può essere diverso a seconda delle caratteristiche dei dati disponibili, del problema in esame e del tipo di modello che si intende addestrare. Le operazioni effettuate piú di frequente sui dati sono la rimozione di trend e di andamenti stagionali da serie temporali, lo *scaling* delle variabili[2], il filtraggio del rumore, ed eventualmente la rimozione di *outliers* (Fayyad *et al.*, 1996), cioè di quei dati particolarmente affetti da rumore[3].

L'esperienza mostra che una buona preparazione dei dati consente di rendere il processo di addestramento del modello più rapido e robusto (Masters, 1995).

2.1.3 Trasformazione dei dati

L'obiettivo di questa fase è quello di estrarre dai dati preprocessati un insieme limitato di *predittori* del fenomeno in esame. Si vuole trovare cioè una rappresentazione il più possibile parsimoniosa dell'informazione contenuta nei dati e necessaria per l'apprendimento. Questa fase si rende necessaria per evitare

[2]Lo *scaling* è un'operazione atta a ridurre l'escursione di una variabile all'interno di un determinato intervallo.

[3]Tenendo conto che il rumore è tutto quello che non riesce ad essere spiegato da un modello, gli *outlier* sono dati che non sono spiegabili con il modello statistico utilizzato. Nel caso della stima di regressione si suppone che i dati siano distribuiti in modo gaussiano attorno alla funzione di regressione. Gli *outlier* sono quindi tutti quei dati che si trovano sulle code della distribuzione.

situazioni di *collinearità* (Montgomery & Peck, 1992) nello spazio di ingresso[4] e per ridurre gli effetti di quel fenomeno noto in letteratura con il nome di *'curse of dimensionality'*[5] (Bellman, 1961).

Le strade più comunemente seguite per diminuire il numero delle variabili considerate sono la selezione di quelle significative per la predizione del fenomeno considerato *(feature selection)*, e la *principal component analysis* che opera un'opportuna ricombinazione delle variabili disponibili (Myers, 1994).

2.1.4 I dati

Il risultato delle fasi illustrate è un insieme $Z = \{z_1, z_2, \ldots, z_n\}$ di dati adatti per essere utilizzati per identificare un modello del fenomeno in esame. Si è indicato con z_i l'i-esimo esempio raccolto; tale esempio è costituito da una coppia (x_i, y_1) in cui x_i è un vettore $(d \times 1)$ che descrive lo stato del fenomeno, e y_i è uno scalare che è la caratteristica del fenomeno che si vuole imparare a predire.

In seguito faremo riferimento collettivamente ai vettori x_i e agli scalari y_i attraverso la matrice X e il vettore Y rispettivamente. La matrice X, che chiameremo *matrice degli ingressi*, ha dimensioni $(n \times d)$ dove n è il numero di esempi, ed è tale per cui la sua i-esima riga è uguale a x_i^T. Il vettore Y, *vettore delle uscite*, ha dimensioni $(n \times 1)$ ed è tale per cui l'i-esimo posto è occupato da y_i

2.2 Identificazione dei parametri

Un processo di identificazione consiste concettualmente di due parti distinte: l'identificazione *strutturale* e quella *parametrica*. Per identificazione strutturale si intende quella fase del processo in cui si determinano le caratteristiche strutturali del modello cioè la famiglia parametrica a cui il modello deve appartenere. Per identificazione parametrica si intende invece la fase di identificazione dei parametri definiti nella fase strutturale. In questo paragrafo daremo per definita la struttura del modello che si vuole identificare e ci occuperemo di come sfruttare l'informazione contenuta nei dati Z per determinare il valore ottimo dei parametri del modello stesso.

Per una data struttura del modello $y = f(x, \alpha)$, la fase di identificazione parametrica consiste nella ricerca all'interno dello spazio Λ dei parametri, di quel vettore $\alpha \in \Lambda$ che minimizza il funzionale di rischio empirico $R_{emp}(\alpha)$ (§ 1.3.1).

[4]Si parla di collinearità quando i dati sono disposti su un sottospazio di dimensioni inferiori rispetto allo spazio di ingresso.

[5]Se il numero degli esempi disponibili è limitato, il crescere della dimensionalità dello spazio di ingresso porta l'insieme dei dati ad essere sempre più sparso rendendo quindi piu' difficile la ricostruzione della dipendenza ingresso uscita.

Per rendere più leggibile la notazione, in questo capitolo poniamo $J(\alpha) = R_{emp}(\alpha)$. Risulta quindi:

$$J(\alpha) = \frac{1}{n}\sum_{i=1}^{n} C\big(y_i, f(x_i, \alpha)\big). \tag{2.1}$$

L'identificazione dei parametri di un modello è quindi rappresentabile come un problema di ottimizzazione dove l'obiettivo è trovare, dato un insieme di esempi dalla forma (x_i, y_i), il vettore $\hat{\alpha}$ tale che:

$$\hat{\alpha} = arg \min_{\alpha} J(\alpha). \tag{2.2}$$

La scelta dell'algoritmo di ottimizzazione da impiegare per ottenere il valore ottimo $\hat{\alpha}$, è condizionata dal tipo di modello $f(x, \alpha)$ che si desidera addestrare e dalla forma della funzione di costo $C\big(y, f(x, \alpha)\big)$ utilizzata nella definizione della (2.1).

Adottando il quadrato dell'errore di predizione come funzione di costo, la funzione $f(x, \alpha)$ identificata, è una stima della funzione di regressione (§ 1.3) di y su x, cioè del valore atteso di y dato x. Nel resto di questo capitolo, ove non altrimenti specificato, si assumerà che la funzione di costo sia definita come segue:

$$C\big(y, f(x, \alpha)\big) = \big(y - f(x, \alpha)\big)^2. \tag{2.3}$$

Ci occuperemo quindi di stima della funzione di regressione.

2.2.1 Metodo dei minimi quadrati

L'identificazione parametrica di un modello lineare

$$f(x, \alpha) = \alpha^T x,$$

qualora si adotti come funzione di costo la (2.3), può essere convenientemente effettuata attraverso il *metodo dei minimi quadrati*. La cifra di merito (2.1), in questo caso specifico, è una funzione quadratica del parametro multivariato α,

$$J(\alpha) = \frac{1}{n}\sum_{i=1}^{n} (y_i - x_i\alpha)^2 \tag{2.4}$$

quindi il gradiente della (2.4) risulta essere funzione lineare di α. La minimizzazione della (2.4) ammette dunque una soluzione in forma chiusa (Myers, 1994).

Attraverso semplici calcoli (Bittanti, 1992b) si giunge alla conclusione che tutti e soli i punti di minimo della (2.4) si ottengono come soluzione delle cosiddette *equazioni normali*:

$$\left[\sum_{i=1}^{n} x_i {x_i}^T\right] \alpha = \sum_{i=1}^{n} x_i y_i.$$

Se la matrice di *scatter*

$$S = \sum_{i=1}^{n} x_i {x_i}^T,$$

è non singolare ed è dunque invertibile, le equazioni normali hanno un'unica soluzione e quindi la (2.4) ha un unico punto di minimo dato da:

$$\hat{\alpha} = \left[\sum_{i=1}^{n} {x_i}^T x_i\right]^{-1} \sum_{i=1}^{n} {x_i}^T y_i.$$

La stessa equazione può essere scritta in termini matriciali come:

$$\hat{\alpha} = \left[X^T X\right]^{-1} X^T Y,$$

dove la matrice $\left[X^T X\right]^{-1} X^T$ prende il nome di *pseudo-inversa* di X.

2.2.2 Metodi iterativi di ricerca

Se il modello da identificare non è lineare o se la funzione di costo considerata non è il quadrato dell'errore commesso, generalmente il problema di ottimizzazione (2.2) non ammette una soluzione in forma chiusa. In questo caso, non essendo possibile minimizzare analiticamente la cifra di merito (2.1), è necessario utilizzare una qualche procedura iterativa di ottimizzazione che ricerchi il minimo nello spazio del parametro α.

In questo paragrafo presenteremo diversi metodi per identificare modelli non lineari minimizzando funzioni obiettivo definite a partire da funzioni di costo $C\big(y, f(x,\alpha)\big)$ qualsivoglia. Nel caso del metodo *Levenberg-Marquardt* però, se la cifra di merito è la somma dei quadrati degli errori commessi, valgono proprietà ulteriori.

I vari algoritmi di ottimizzazione che presenteremo, si differenziano per il tipo di informazione sulla funzione $f(x,\alpha)$ di cui necessitano, e per l'uso che fanno di tale informazione. Qualora esistano e siano disponibili, le derivate del modello rispetto ai parametri possono essere utilizzate per trovare il minimo della cifra di merito (2.1) attraverso uno dei vari algoritmi iterativi di ottimizzazione basati sul gradiente.

Gradient descent

E' il più semplice degli algoritmi di ottimizzazione che sfruttano l'informazione fornita dal gradiente.

Se $J(\alpha)$ è una funzione differenziabile di α, è possibile adottare la seguente procedura. Il vettore dei parametri viene inizializzato, tipicamente in maniera casuale, ad un valore $\alpha^{(0)}$. Ad ogni iterazione, la stima corrente di tale vettore viene aggiornata muovendo dal valore attuale nello spazio dei parametri un passo nella direzione in cui $J(\alpha)$ decresce più rapidamente, cioè nella direzione opposta a quella del suo gradiente. Iterando questo processo, viene generata una sequenza di vettori $\alpha^{(\tau)}$ secondo la seguente legge:

$$\alpha^{(\tau+1)} = \alpha^{(\tau)} - \mu g, \tag{2.5}$$

dove μ, parametro dell'algoritmo, è un numero positivo che prende il nome di *learning rate*, e il vettore g è il gradiente $\nabla J(\alpha)$ valutato in $\alpha^{(\tau)}$. Sotto determinate condizioni relative al parametro μ, il metodo del gradiente converge globalmente (Colorni, 1984).

Metodo di Newton

Il metodo *gradient descent* è noto essere inefficiente, specialmente in prossimità del punto di minimo (Sjöberg *et al.*, 1994). Una possibile alternativa più efficiente è costituita dal *metodo di Newton*. Tale metodo, per aumentare la velocità di convergenza, utilizza oltre all'informazione contenuta nel gradiente, anche quella fornita dall'*Hessiano*, cioè dalla matrice H delle derivate seconde di $J(\alpha)$ rispetto alle varie componenti del vettore α.

La ricerca del minimo della cifra di merito avviene in maniera iterativa: dato il valore attuale $\alpha^{(\tau)}$, il valore del parametro all'iterazione successiva è il minimo del paraboloide che, in un intorno di $\alpha^{(\tau)}$, approssima la cifra di merito $J(\alpha)$.

L'equazione di tale paraboloide approssimante è data dall'espansione di Taylor ed ha la forma seguente:

$$\hat{J}(\alpha) = J(\alpha^{(\tau)}) + (\alpha - \alpha^{(\tau)})^T g + \frac{1}{2}(\alpha - \alpha^{(\tau)})^T H(\alpha - \alpha^{(\tau)}), \tag{2.6}$$

dove g è il gradiente di $J(\alpha)$ valutato in $\alpha^{(\tau)}$ e H è l'Hessiano di $J(\alpha)$ sempre valutato in $\alpha^{(\tau)}$.

Il minimo del paraboloide approssimante (o più correttamente un suo punto di stazionarietà) può essere ottenuto direttamente dalla (2.6). Ponendo $\alpha^{(\tau+1)}$ uguale a tale punto di stazionarietà si ottiene:

$$\alpha^{(\tau+1)} = \alpha^{(\tau)} - H^{-1} g.$$

Il vettore $-H^{-1}g$, che prende il nome di *direzione di Newton*, determina la direzione e la lunghezza del passo che caratterizzano ciascuna iterazione del metodo.

Levenberg-Marquardt

I metodi iterativi finora introdotti, sono pensati per identificare sistemi lineari e non lineari senza imporre alcuna condizione restrittiva sulla definizione della funzione di costo. Il metodo che presentiamo in questo paragrafo, pur essendo adatto per minimizzare funzioni obiettivo qualsivoglia, è pensato specificatamente per il problema della stima di regressione e quindi per trattare funzioni che siano definite come la somma dei quadrati degli errori. In tale caso, valgono alcune proprietà notevoli.

Si consideri qunque la cifra di merito da minimizzare nella seguente forma:

$$J = \frac{1}{2}\sum_{i=1}^{n}(\epsilon_i)^2 = \frac{1}{2}\|\epsilon\|^2, \tag{2.7}$$

dove ϵ_i è l'errore commesso in corrispondenza dell' i-esimo esempio di Z, ed ϵ è un vettore i cui elementi sono gli errori ϵ_i. Sia $\alpha^{(\tau)}$ il valore del parametro da identificare all'iterazione corrente; si consideri uno spostamento nello spazio dei parametri pari ad $(\alpha^{(\tau+1)} - \alpha^{(\tau)})$. Se tale spostamento è sufficientemente piccolo, è possibile espandere il vettore degli errori ϵ in una serie di Taylor del primo ordine:

$$\epsilon(\alpha^{(\tau+1)}) = \epsilon(\alpha^{(\tau)}) + E(\alpha^{(\tau+1)} - \alpha^{(\tau)}).$$

Gli elementi E_{ij} della matrice E sono definiti come:

$$E_{ij} = \frac{\partial \epsilon_i}{\partial \alpha_j},$$

dove α_j è il j-esimo elemento del parametro multivariato α. La (2.7) può essere dunque approssimata come segue:

$$J(\alpha^{(\tau)}) = \frac{1}{2}\|\epsilon(\alpha^{(\tau)}) + E(\alpha^{(\tau+1)} - \alpha^{(\tau)})\|^2. \tag{2.8}$$

Minimizzando questa cifra di merito rispetto al nuovo valore del parametro $\alpha^{(\tau+1)}$, si ottiene:

$$\alpha^{(\tau+1)} = \alpha^{(\tau)} - \left[E^T E\right]^{-1} E^T \epsilon(\alpha^{(\tau)}). \tag{2.9}$$

Dove compare il termine $\left[E^T E\right]^{-1} E^T$ che è la pseudo-inversa (§ 2.2.1) della matrice E.

Nel caso in cui la cifra di merito sia una somma di quadrati, come nella (2.7), gli elementi dell'Hessiano possono essere scritti nella forma seguente:

$$H_{jk} = \frac{\partial^2 J}{\partial \alpha_j \partial \alpha_k} = \sum_{i=1}^{n} \left\{ \frac{\partial \epsilon_i}{\partial \alpha_j} \frac{\partial \epsilon_i}{\partial \alpha_k} + \epsilon_i \frac{\partial^2 \epsilon_i}{\partial \alpha_j \partial \alpha_k} \right\}. \tag{2.10}$$

Qualora non fossero disponibili le derivate del secondo ordine di ϵ_i, è in ogni caso possibile ottenere una approssimazione dell'Hessiano trascurando nella (2.10) i termini di ordine superiore al primo. Risulta quindi:

$$H \simeq E^T E. \tag{2.11}$$

Si noti che la (2.11) risulta essere esatta per i modelli lineari in quanto, per questi, l'errore ϵ è una funzione quadratica dei parametri, e i termini del secondo ordine nella (2.10) sono nulli. Per i modelli non lineari, la (2.11) rappresenta una approssimazione, anche se, asintoticamente[6] la (2.11) è esatta nel punto di minimo globale di $J(\alpha)$ (Bishop, 1995). L'equazione (2.9) può essere quindi vista come l'equazione del metodo di Newton dove il valore esatto H dell'Hessiano è stato approssimato con $E^T E$.

In linea di principio, la (2.9) potrebbe essere applicata iterativamente per minimizzare la cifra di merito $J(\alpha)$. In realtà, così facendo, il passo potrebbe essere troppo lungo e potrebbe portare quindi in una regione dello spazio dei parametri dove l'approssimazione (2.11) non è più valida. L'algoritmo *Levenberg-Marquardt* affronta questo problema modificando la cifra di merito (2.8) nel modo seguente:

$$J_{lm} = \frac{1}{2} \| \epsilon(\alpha^{(\tau)}) + E(\alpha^{(\tau+1)} - \alpha^{(\tau)}) \|^2 + \lambda \| \alpha^{(\tau+1)} - \alpha^{(\tau)} \|^2, \tag{2.12}$$

dove il parametro λ governa la lunghezza del passo (Levenberg, 1944) (Marquardt, 1963).

Minimizzare la (2.12) significa minimizzare la somma dei quadrati degli errori, tenendo allo stesso tempo piccola la lunghezza del passo per non uscire dall'intorno del valore corrente $\alpha^{(\tau)}$, dove l'approssimazione lineare è accettabile.

Minimizzando la nuova cifra di merito (2.12) rispetto ad α, si ottiene:

$$\alpha^{(\tau+1)} = \alpha^{(\tau)} - \left[E^T E + \lambda I \right]^{-1} E^T \epsilon(\alpha_{(\tau)}), \tag{2.13}$$

dove I è la matrice unitaria. Per $\lambda = 0$, la (2.13) è uguale alla formula (2.9) ed è quindi ancora un'approssimazione del metodo di Newton. Al crescere di λ, l'equazione (2.13) tende a quella del metodo del gradiente. In quest'ultimo caso la lunghezza del passo è determinata da λ^{-1}. Per valori sufficientemente

[6] Per n che tende a infinito.

grandi di λ, è sempre possibile determinare un passo nello spazio dei parametri tale per cui J_{lm} non aumenti. Infatti, adottare un grande valore di λ significa effettuare un passo, piccolo a piacere, nella direzione opposta a quella del gradiente e quindi nella direzione in cui J_{lm} decresce più rapidamente.

In pratica, il valore di λ viene fatto variare ad ogni iterazione secondo il seguente criterio (Bishop, 1995). Dato il valore attuale $\alpha^{(\tau)}$ del parametro da identificare, e fissato un valore iniziale per λ, viene effettuato il passo proposto dalla (2.13). Se con tale passo J_{lm} diminuisce, il nuovo valore $\alpha^{(\tau+1)}$ viene accettato e il processo viene iterato diminuendo λ di un fattore 10. Se invece la cifra di merito J_{lm} aumenta, il valore $\alpha^{(\tau+1)}$ proposto dalla (2.13) viene rifiutato, e ripartendo $\alpha^{(\tau)}$, si tenta un altro passo con un valore di λ aumentato di un fattore 10; questo processo viene iterato fino a che non si trova un valore di λ che consenta di diminuire J_{lm}.

Limiti dei metodi basati sul gradiente

Il problema fondamentale dei metodi di identificazione basati sul gradiente è che la cifra di merito definita nell'equazione (2.1), può essere una funzione non lineare che presenta una serie di punti di minimo locale nello spazio dei parametri. Gli algoritmi di ottimizzazione visti, così come tutti quelli appartenenti alla stessa famiglia, cercano un punto di minimo della funzione $J(\alpha)$ esplorando lo spazio dei parametri con un processo iterativo, avendo a disposizione, ad ogni passo, solo informazioni locali. Il limite degli algoritmi che adottano questa politica di ricerca è che non garantiscono di individuare il minimo globale di $J(\alpha)$ in quanto non dispongono di un criterio che consenta di individuare e quindi evitare i minimi locali.

In letteratura esistono parecchi metodi alternativi che sono stati proposti per affrontare il problema dei minimi locali. I più frequentemente adottati sono i seguenti:

Random search: è il più banale degli algoritmi di ottimizzazione. Consiste in un'esplorazione casuale dello spazio di α alla ricerca di quel valore che minimizzi la funzione obiettivo.

Random restart: consiste nel ripetere più volte l'identificazione dei parametri attraverso uno dei metodi iterativi visti, cominciando ogni volta con un differente valore di inizializzazione $\alpha^{(0)}$. Questo approccio cerca in modo casuale una condizione iniziale che consenta al processo di ottimizzazione di raggiungere il minimo globale della funzione $J(\alpha)$, senza rimanere intrappolato in un minimo locale.

Algoritmi genetici: sono metodi in cui il minimo di una funzione è cercato sfruttando solo informazioni sul valore della funzione obiettivo e non sulle sue derivate. Gli algoritmi genetici traggono ispirazione dalle dinamiche

naturali di adattamento di una popolazione all'ambiente (Golberg, 1989): il minimo di una funzione $J(\alpha)$ è cercato simulando l'evoluzione di una *popolazione* di individui, ciascuno caratterizzato da un *patrimonio genetico* che individua un punto nello spazio dei parametri α. Tanto minore è il valore della funzione da ottimizzare in tale punto e tanto maggiore si dice essere il *grado di adattamento* all'ambiente dell'individuo in questione. Ogni individuo si *riproduce*, consentendo quindi che il suo patrimonio genetico partecipi alla determinazione di quello delle generazioni future, con probabilità che è funzione del suo grado di adattamento all'ambiente. Sotto certe ipotesi, al passare delle generazioni la popolazione simulata tende verso il massimo grado di adattamento e il patrimonio genetico dei vari individui converge verso un unico valore corrispondente al valore ottimo del parametro α.

Simulated annealing: è un metodo alla cui base sta un'analogia con la termodinamica, in particolare con il modo in cui i liquidi congelano e cristallizzano o con il modo in cui i metalli si raffreddano. L'esplorazione dello spazio dei parametri viene effettuata iterativamente, considerando solo informazioni relative al valore e non alle derivate della funzione obiettivo. Dato il valore corrente $\alpha^{(\tau)}$, viene proposto in maniera casuale uno spostamento in un punto $\tilde{\alpha}^{(\tau+1)}$ che si trovi in un intorno di $\alpha^{(\tau)}$ stesso. Qualora sia $J(\tilde{\alpha}^{(\tau+1)}) < J(\alpha^{(\tau)})$, lo spostamento viene realmente effettuato: $\alpha^{(\tau+1)} = \tilde{\alpha}^{(\tau+1)}$. Altrimenti, viene effettuato con una probabilità $T(\tau)$ (la temperatura del corpo che si raffredda nella metafora termodinamica), che è una funzione decrescente del numero di iterazioni eseguite.

2.3 Validazione

Dopo la fase di identificazione parametrica è necessaria una fase di validazione per esaminare il modello, valutarne la qualità rispetto a standard definiti in base allo scopo per cui l'identificazione è stata effettuata, ed eventualmente rifiutarlo se non dovesse essere ritenuto soddisfacente (Ljung, 1993).

Anche per quanto riguarda la fase di validazione, quale indice della bontà di un modello è naturale considerare il funzionale di rischio (§ 1.3) che rappresenta il *valore atteso dell'errore quadratico*:

$$E\left[\epsilon^2\right] = \iint \big(y - f(x,\alpha)\big)^2 p(x,y)\,\mathrm{d}x\,\mathrm{d}y. \tag{2.14}$$

La (2.14) è una grandezza teorica che, per essere valutata, richiederebbe di testare il modello $f(x,\alpha)$ in un numero infinito di punti. Una stima della (2.14)

è data dal *valor medio campionario dell'errore quadratico*

$$MSE = \frac{1}{l}\sum_{i=1}^{l}\big(y_i - f(x_i,\alpha)\big)^2, \tag{2.15}$$

valutato in un numero l di punti che tende all'infinito.

Affrontando problemi reali, il numero l di campioni disponibili è sempre finito e tipicamente relativamente piccolo: il problema è dunque quello di trovare il metodo migliore per stimare il valore di $E[\epsilon^2]$, a partire da un limitato insieme di campioni.

Esistono diverse tecniche per ottenere una stima della (2.14). La più banale è quella che prende il nome di *risostituzione* e che consiste nel valutare il valor medio del quadrato dell'errore commesso dal modello sugli stessi esempi utilizzati in fase di identificazione. La quantità cosi calcolata prende il nome di *errore quadratico medio apparente* ed è nota essere una stima polarizzata *ottimisticamente* del valore atteso dell'errore quadratico (Weiss & Kulikowski, 1991).

L'obiettivo dell'apprendimento non è semplicemente quello di riprodurre gli esempi disponibili. Quello che si desidera tipicamente è che il sistema addestrato sia in grado di cogliere l'essenza del meccanismo che ha generato gli esempi e sia quindi in grado di *generalizzare* a casi non contemplati dagli esempi stessi. E' chiaro che l'*errore quadratico medio apparente* non è la quantità adatta per valutare la capacità di generalizzazione di un sistema.

Stime più significative e non polarizzate della (2.14), possono essere ottenute testando il modello con esempi diversi da quelli usati in fase di apprendimento.

2.3.1 Holdout

Il modo più semplice per ottenere una stima della capacità che ha un modello di generalizzare, consiste nel dividere gli esempi Z disponibili, in due insiemi, *training set* e *test set*, e di usare il primo di questi per addestrare il modello e il secondo per validarlo. E' pratica comune (Kohavi, 1995) utilizzare circa 2/3 degli esempi per l'identificazione e 1/3 per la validazione. Affinché la stima del valore atteso dell'errore quadratico sia affidabile, la partizione di Z nei due insiemi deve essere ottenuta in maniera tale che *test* e *training set* siano campioni della stessa popolazione e siano indipendenti tra di loro.

Quando il numero di esempi disponibili è sufficientemente elevato, il metodo *holdout* è una tecnica preziosa di validazione in quanto fornisce una stima non polarizzata della (2.14) ad un costo computazionale ragionevole. Quando però il numero di esempi è limitato, questo metodo si rivela inadatto in quanto gli insiemi di training e test risultanti, possono essere troppo piccoli, e la stima fornita risulta dipendere fortemente dalla particolare ripartizione degli esempi

adottata. In questo secondo caso il metodo *holdout* è uno stimatore polarizzato *pessimisticamente* della (2.14) (Weiss & Kulikowski, 1991).

2.3.2 Cross validation

Un approccio più dispendioso dal punto di vista computazionale, ma che consente di sfruttare meglio l'informazione contenuta in Z, e di avere una stima più accurata della capacità di generalizzazione, è l'approccio che prende il nome di *k-fold cross validation*. Seguendo questo metodo, gli esempi disponibili Z vengono divisi in k insiemi $Z_1, Z_2, \ldots, Z_k$, circa delle stesse dimensioni. Il modello viene identificato k volte. Per ogni $t \in \{1, 2, \ldots, k\}$, sia MSE_t il valore dell'errore quadratico medio ottenuto testando su Z_t, il modello identificato usando l'insieme $Z \backslash Z_t$ come training set (Kohavi, 1995). La stima fornita dalla cross validation della capacità predittiva del modello è data quindi da:

$$MSE = \frac{1}{k} \sum_{t=1}^{k} MSE_t.$$

Tale stima è un numero casuale che dipende dalla particolare ripartizione considerata degli elementi di Z nei sottoinsiemi Z_t. La *cross-validation completa* è la media di tutti i $\binom{n}{n/k}$ risultati di cross validation ottenibili dalle possibili ripartizioni degli n elementi di Z in sottoinsiemi di n/k elementi ciascuno. Dati i suoi costi, la cross-validation completa è normalmente impraticabile (Kohavi, 1995) ad esclusione del caso del *leave-one-out*, n-fold cross validation, che è sempre completo.

2.3.3 Leave-one-out

Un caso particolare di k-fold cross validation è il *leave-one-out* che si ottiene nel caso si ponga $k = n$.

Il processo può essere così descritto: per ogni osservazione (x_i, y_i), si considera l'errore di predizione (il *residuo*) $y_i - \hat{y}_{i,-i} = \epsilon_{i,-i}$, dove $\hat{y}_{i,-i}$ è la predizione del valore assunto in x_i, fornita dal modello addestrato utilizzando tutti gli n punti ad esclusione dell'i-esimo. La stima della (2.14) fornita dal *leave-one-out* è quindi:

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_{i,-i})^2 = \frac{1}{n} \sum_{i=1}^{n} (\epsilon_{i,-i})^2.$$

Il *leave-one-out* è un metodo decisamente dispendioso in quanto, se sono disponibili n esempi, richiede che il modello sia identificato n volte. Ciò nonostante, questo metodo merita attenzione in quanto è in grado di fornire una stima non polarizzata del valore atteso dell'errore quadratico (Efron & Tsibirani, 1995).

In pratica, il *leave-one-out* può essere applicato proficuamente ed economicamente nella validazione di sistemi lineari. In questo caso, infatti, è possibile sfruttare le caratteristiche della *PRESS statistic* (Myers, 1994). in questo caso, i residui sono calcolati come:

$$\epsilon_{i,-i} = \frac{y_i - \hat{y}_i}{1 - x_i \left(X^T X\right)^{-1} x^T{}_i} = \frac{\epsilon_i}{1 - h_{ii}},$$

dove $\hat{y}_i$ è la predizione del valore assunto in x_i, fornita dal modello addestrato utilizzando tutti gli n punti, e i termini h_{ii} sono gli elementi diagonali della matrice H (*HAT matrix*):

$$H = X \left(X^T X\right)^{-1} X^T.$$

2.3.4 Bootstrap

Il *leave-one-out* fornsce una stima non polarizzata ma che soffre di un'alta varianza quando gli esempi disponibili sono pochi[7] (Efron & Tsibirani, 1995).

Un alternativa al *leave-one-out* è fornita dal metodo di ricampionamento noto con il nome di *bootstrap* (Efron & Tsibirani, 1994). Tale metodo, nella sua versione più semplice, fornisce una stima della (2.14) operando nel modo seguente. Dall'insieme Z degli n esempi disponibili, viene estratto con *riposizionamento*, un campione b_j (campione di bootstrap), composto da n elementi. Il campione b_j così ottenuto, viene usato per identificare un modello che viene validato su quegli esempi presenti in Z che non sono entrati a far parte del campione b_j. Viene così ottenuto l'errore quadratico medio MSE_j associato all'i-esimo campione di bootstrap. La procedura descritta viene ripetuta B volte, dove B è un parametro del metodo di validazione. La stima del valore atteso dell'errore quadratico è data dalla media dei MSE_j calcolati:

$$MSE = \frac{1}{B} \sum_{j=1}^{B} MSE_j$$

Per una stima affidabile del MSE, 200 iterazioni sono normalmente ritenute necessarie (Efron & Tsibirani, 1994).

2.4 Identificazione strutturale

L'ultima fase di un processo di apprendimento che esaminiamo in questo capitolo è quella che prende il nome di *identificazione strutturale*. Questa è la fase

[7] tradizionalmente si parla di statistica dei piccoli campioni quando il numero di esempi è minore o uguale a 30

in cui viene determinata quale deve essere la struttura del modello da impiegare per descrivere il processo che ha generato l'insieme di esempi Z. Determinare la struttura di un modello significa fissare il valore di un parametro da cui dipende la complessità del modello stesso. A seconda del tipo di modello considerato, l'indice di complessità si riferisce a caratteristiche diverse: può essere il grado, l'ordine, il numero di neuroni dello strato nascosto o di regole a seconda che si costruisca un modello del fenomeno in esame utilizzando polinomi, modelli AR, reti neurali o modelli *fuzzy*. In ogni caso, al crescere dell'indice di complessità, cresce la flessibilità e quindi la potenza espressiva. Per un dato valore m dell'indice di complessità, il modello ottimo viene determinato minimizzando, rispetto al vettore α, una cifra di merito empirica data dall'errore quadratico medio commesso dal modello stesso sui dati disponibili (§ 2.2):

$$J(\alpha) = \frac{1}{n} \sum_{i=1}^{n} C\big(y_i, f^m(x_i, \alpha)\big). \tag{2.16}$$

Dove si è indicato con notazione $f^m(x, \alpha)$ il modello di complessità m. Se α^m è il punto di minimo della (2.16), la quantità $J(\alpha^m)$ esprime il grado di aderenza ai dati del modello ottimo di complessità m.

Al crescere della complessità, passando ad esempio da m a $m+1$, si può stimare il vettore dei parametri α^{m+1} che descrive il modello ottimo di complessità $m+1$, e valutare il valore della cifra di merito $J(\alpha^{m+1})$. Ovviamente, aumentando la complessita del modello e quindi il numero dei suoi gradi di libertà, aumenta la sua capacità di aderenza ai dati (Bittanti, 1992a): risulta quindi necessariamente $J(\alpha^{m+1}) \leq J(\alpha^m)$.

L'obiettivo dell'apprendimento non è però tanto quello di arrivare a spiegare i dati disponibili, ma piuttosto quello di generalizzare e di cogliere quindi il reale meccanismo che ha generato i dati (§ 1.3.2). Le informazioni estratte dalla cifra di merito $J(\alpha)$ non possono dunque essere utilizzate per determinare la complessita ottima del modello da utilizzare. La complessità ottima deve essere quindi scelta utilizzando un criterio differente che sia una stima il più affidabile possibile della capacità di generalizzazione di un modello. Una stima di questo tipo può essere fornita da metodi di validazione quali *hold out*, *cross-validation* o *bootstrap* (§ 2.3). Al crescere della complessità tipicamente si osserva un andamento caratterizzato da una iniziale diminuzione della stima della capacità di generalizzazione, seguita da una fase in cui tale stima torna a salire (figura 2.2). Si osserva dunque anche in pratica lo stesso andamento qualitativo indicato sia dall'analisi *bias/variance* (§ 1.3.2), sia da quella basata sulla *VC dimension* (§ 1.3.2).

Fissato il criterio di valutazione della capacità di generalizzazione di un modello di una data complessità, per definire un algoritmo di identificazione strutturale è necessario individuare un criterio di esplorazione dello spazio del parametro m. In letteratura esistono diverse soluzioni, tipicamente basate su

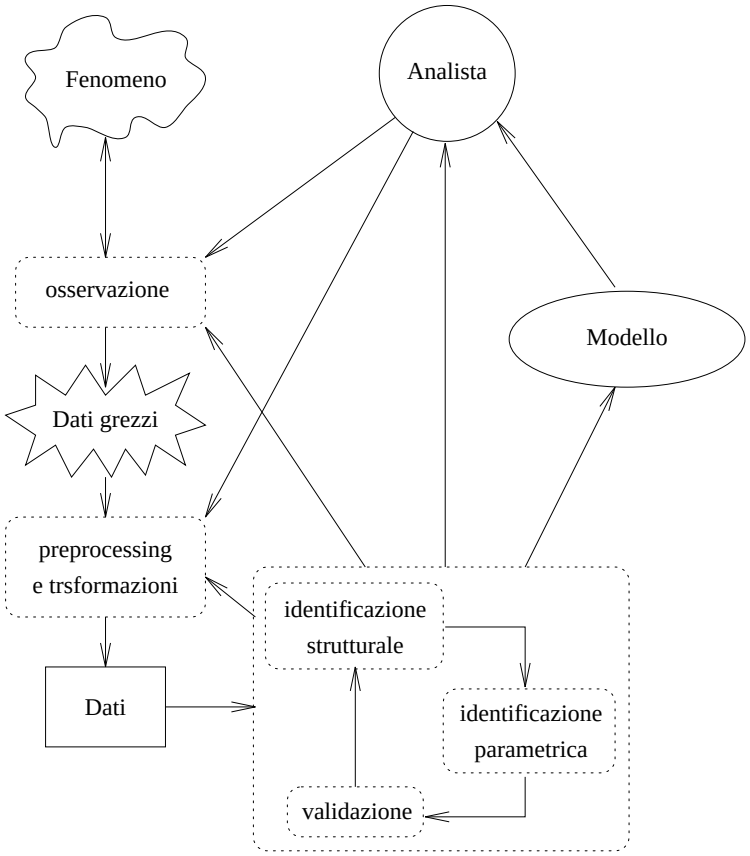


Figura 2.1: Un processo di apprendimento non procede in modo sequenziale: è piuttosto un processo ciclico che ripete successivamente più fasi fino a giungere, attraverso raffinamenti successivi, ad una rappresentazione soddisfacente del fenomeno in esame.

uno schema *trials and errors*. Secondo questo schema vengono considerati modelli di diverse complessità e per ciascuno di questi viene valutata una stima della capacità di generalizzazione; il processo viene iterato fintanto che viene individuato un livello di complessità soddisfacente. Da un punto di vista più pratico la ricerca nello spazio del parametro strutturale può avvenire in maniera incrementale, facendo cioè aumentare il valore di m ad ogni iterazione, in maniera decrementale, o in maniera esaustiva avendo preventivamente limitato lo spazio di ricerca impiegando conoscenza a priori sul fenomeno in esame.

Nel prossimo capitolo introdurremo l'approccio multimodello presentando tre diverse architetture. Mostreremo come ciascuna delle fasi di un processo di apprendimento presentate astrattamente in questo capitolo vengono interpretate qualora si adotti un architettura multimodello.

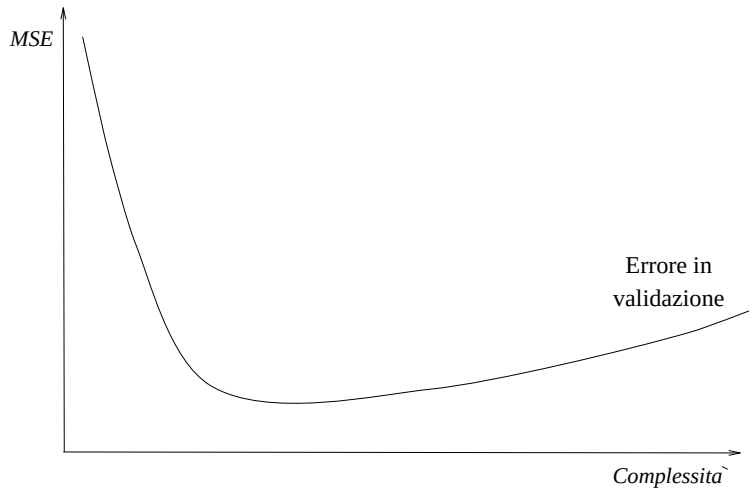


Figura 2.2: Andamento tipico della stima dell'errore fornito da un metodo di validazione. Tale andamento risulta qualitativamente simile a quelli indicati per via teorica dall'analisi *bias/variance* e da quella basata sulla *VC dimension*.

Capitolo 3

L'approccio Multimodello: Stato dell'arte

Un'architettura modulare affronta un problema di apprendimento di una relazione ingresso-uscita decomponendolo in più sottoproblemi, ciascuno più semplice da risolvere che non il problema dato (Jacobs *et al.*, 1991b). La decomposizione e la conseguente riduzione di complessità è ottenuta partizionando lo spazio di input e assegnando ciascuna delle partizioni ottenute ad un diverso approssimatore locale.

In generale non è disponibile informazione a priori su come effettuare la partizione, sul numero e sulla struttura analitica degli approssimatori: questo tipo di informazione deve quindi essere estratta dall'insieme di esempi disponibili.

L'utilizzo di una architettura modulare presuppone l'assunzione che la funzione che si vuole apprendere sia ben descrivibile come una collezione di funzioni più semplici definite ciascuna in una regione dello spazio di input. In questo capitolo vedremo tre approcci differenti alla modellistica multimodello, ciascuno caratterizzato da un diverso criterio di ripartizione dello spazio di input.

3.1 Mixture of Experts

3.1.1 Gli esperti locali di Jordan

Il modello *mixture of expert* (Jordan & Jacobs, 1995), è composto da m moduli detti *expert network*, ciascuno dei quali implementa una funzione $f_j = f_j(x, \theta_j)$, dove x è un valore dell'ingresso, e θ_j è un parametro multivariato. Tipicamente i vari esperti sono istanze di un'unica classe parametrica e condividono quindi la stessa struttura: si differenziano l'uno dall'altro perchè caratterizzati da diversi valori del vettore θ_j.

Per coordinare le predizioni fornite dagli m moduli, l'architettura richiede

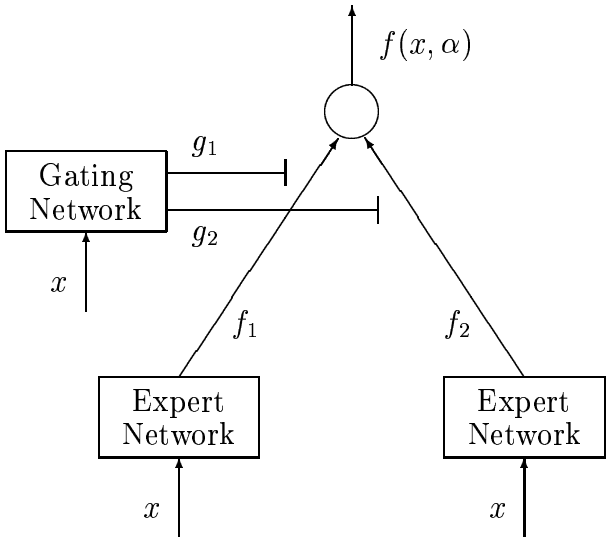


Figura 3.1: L'architettura *Mixture of experts*.

che sia definito un meccanismo che, dato un valore x dell'ingresso, individui quale esperto o quale combinazione degli esperti, garantisca la maggior probabilità di fornire una corretta predizione dell'uscita y. Tale meccanismo è realizzato da un modulo ausiliario, *gating network*, il quale produce come output un insieme di coefficienti scalari $g_j(x)$, che vengono utilizzati per pesare i contributi dei vari esperti. Il *gating network* può essere interpretato come un classificatore (Jordan & Jacobs, 1995) che mappa l'ingresso x, nelle probabilità che i vari esperti siano in grado di generare la corretta predizione dell'uscita. I coefficienti g_j, data la loro interpretazione come probabilità, devono essere, per ogni valore di x, non negativi e dare come somma l'unità.

Sono state proposte diverse forme analitiche alternative per i coefficienti $g_j(x)$ (Xu *et al.*, 1995). Un approccio prevede l'utilizzo della funzione *softmax* (Jacobs *et al.*, 1991a). Definita la variabile di supporto ξ_j come

$$\xi_j = \xi_j(x, \eta)\,, \tag{3.1}$$

dove η è un parametro multivariato, l'uscita del *gating network* è ottenuta come segue:

$$g_j = \frac{e^{\xi_j}}{\sum_k e^{\xi_k}}\,. \tag{3.2}$$

Data la struttura del *gating* e degli *expert network*, è possibile scrivere la predizione del valore dell'uscita y, nella seguente forma:

$$f(x, \alpha) = \sum_{j=1}^{m} g_j(x, \eta) f_j(x, \theta_j)\,, \tag{3.3}$$

essendo $\alpha = \left[\theta_1^T, \theta_2^T, \ldots, \theta_m^T, \eta^T\right]^T$ il vettore di tutti i parametri che descrivono il modello.

L'interpretazione probabilistica

L'adozione dell'architettura *mixture of expert* implica che vengano formulate una serie di ipotesi sui dati disponibili e sul meccanismo che li ha generati (Jordan & Jacobs, 1994). Tali ipotesi possono essere convenientemente esplicitate adottando il seguente modello probabilistico per descrivere l'insieme di osservazioni $Z = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$. Dato un valore x_i per l'ingresso, viene scelto un intero j (*etichetta*) con probabilità $\pi(j|x_i, \bar{\eta})$, dove si sia adottata la convenzione di scrivere soprassegnati i valori *veri* dei parametri, per distinguerli da quelli identificati per il modello. Dato l'ingresso x_i, ed assegnata l'etichetta j, finalmente viene estratto un valore y_i, dalla densità condizionata $p(y_i|x_i, \bar{\theta}_j)$. Si noti che, per un dato un valore x_i, l'uscita y_i può essere generata in m modi differenti, dove m è il numero di valori diversi che l'etichetta j può assumere. La probabilià totale di generare y_i dato x_i è data quindi dalla seguente *mixture of density* (§ 1.2.3):

$$p(y_i|x, \bar{\alpha}) = \sum_{j=1}^{m} \pi(j|x_i, \bar{\eta}) p(y_i|x_i, \bar{\theta}_j)\,, \tag{3.4}$$

dove $\bar{\alpha}$ è il vettore di tutti i parametri $\bar{\eta}$ e $\bar{\theta}_j$ che caratterizzano la generazione dell'insieme degli esempi Z.

Il compito del *gating network* è quello di modellizzare, ad esempio attraverso la struttura parametrica definita dalle (3.1) e (3.2), le probabilità $\pi(j|x, \bar{\eta})$. Gli *expert network* invece calcolano, per quanto riguarda la regione di loro competenza, il valor medio della densità $p(y|x, \bar{\theta}_j)$, cioè il valore atteso di y condizionato a x (funzione di regressione).

L'identificazione dei parametri

Il parametro α del modello (3.3), viene stimato secondo il criterio della massima verosimiglianza (§ 1.2.1), considerando la seguente cifra di merito:

$$L(Z, \alpha) = \sum_{i=1}^{n} \ln p(y_i|x_i, \alpha)\,.$$

In pratica, la stima della massima verosimiglianza viene effettuata utilizzando l'algoritmo *Expectation-Maximization* comunemente noto con l'acronimo EM (Dempster *et al.*, 1977).

L'algoritmo EM è stato introdotto per massimizzare la funzione di verosimiglianza dei parametri di un modello nel caso si abbia in possesso un insieme *incompleto* di dati (Bradshaw, 1996). Nel caso del modello *mixture of experts*,

viene ipotizzata l'esistenza di un set *completo* di dati tale per cui ad ogni coppia (x_i, y_i), è associata l'*etichetta* che ne ha determinata la generazione. Si suppone quindi che i dati disponibili siano ottenuti rimuovendo l'*etichetta* dal set completo.

L'algoritmo EM ha una forma iterativa tale per cui ogni iterazione è composta da due passi: *expectation* e *maximization*, o più brevemente passo *(E)* e passo *(M)* rispettivamente. All'interno del passo *expectation*, dato il valore corrente dei parametri, viene generata una stima delle *etichette*. Tale stima viene utilizzata per *ricostruire* il set completo in base al quale viene eseguito il passo *maximization* che fornisce un nuovo valore dei parametri (Bishop, 1995).

Formalmente, dato il valore corrente del parametro $\alpha^{(\tau)}$, ciascuna iterazione ha la forma seguente (Jordan & Xu, 1993):

Passo (E). Per ogni coppia (x_i, y_i), vengono calcolate le quantità $h_j^{(\tau)}(y_i, x_i) = \pi(j|x_i, y_i)$. Per ogni etichetta j, tali quantità rappresentano la stima ottenuta conoscendo sia x_i che y_i, della probabilità che l'i-esimo esempio di Z sia stato ottenuto in seguito all'estrazione dell'etichetta j stessa. Definiti i valori $h_j^{(\tau)}(y_i, x_i)$, viene generato il seguente insieme di funzioni obiettivo:

$$Q_j^e(\theta_j) = \sum_{i=1}^{n} h_j^{(\tau)}(y_i, x_i) \ln p(y_i|x_i, \theta_j) \qquad \text{con } j = 1, \ldots, m$$

$$Q^g(\eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} h_j^{(\tau)}(y_i, x_i) \ln g_j^{(\tau)}(x_i, \eta_j)$$

Passo (M). La stima del parametro viene aggiornata al vettore $\alpha^{(\tau+1)}$, i cui componenti sono ottenuti risolvendo il seguente insieme di problemi di ottimizzazione:

$$\theta_j^{(\tau+1)} = arg \max_{\theta_j} Q_j^e(\theta^j) \qquad \text{con } j = 1, \ldots, m$$

$$\eta^{(\tau+1)} = arg \max_{\eta} Q^g(\eta)$$

Nel caso generico, il passo (M) comporta la soluzione di $m+1$ problemi di ottimizzazione non lineare che possono essere affrontati utilizzando un metodo di ricerca iterativo. Ciò significa che, utilizzando il metodo EM, l'identificazione del parametro α è un algoritmo che prevede due cicli annidati e che quindi ha elevati costi computazionali. Per cercare di ridurre tali costi, è possibile implementare l'algoritmo EM in maniera differente. Mentre l'algoritmo EM propriamente detto, prevede la completa massimizzazione delle quantità $Q_j^e(\theta_j)$ e $Q^g(\eta)$ all'interno del passo (M) di ogni iterazione, esiste una comune implementazione, conosciuta con il nome di GEM (EM generalizzato), che si limita

ad incrementare ad ogni passo i valori di $Q_j^e(\theta_j)$ e $Q^g(\eta)$, senza garantirne la massimizzazione (Jordan & Jacobs, 1994).

Un altro approccio che consente di semplificare l'implementazione dell'algoritmo EM, consiste nello scegliere la struttura del *gating network* e quella degli *expert network*, in maniera tale che i problemi di massimizzazione delle quantità $Q_j^e(\theta_j)$ e $Q^g(\eta)$, ammettano soluzioni in forma chiusa, e che quindi possano essere risolti analiticamente in un solo passo. Una implementazione (Xu *et al.*, 1995) dell'algoritmo EM ad un solo ciclo, è resa possibile dall'adozione di modelli locali lineari nei parametri per quanto riguarda gli *expert network*, e di *gating network* realizzati come:

$$g_j(x,\eta) = \frac{\zeta_j p(x|\eta_j)}{\sum_k \zeta_k p(x|\eta_k)} \qquad \text{con} \sum_k \zeta_k = 1 \text{ e } \zeta_k \geq 0,$$

dove $\eta = \left[\zeta_1, \eta_1^T, \zeta_2, \eta_2^T, \ldots, \zeta_m, \eta_m^T\right]^T$, e la probabilità $p(x|\eta_j)$ è una distribuzione gaussiana, o un'altra funzione appartenente alla famiglia esponenziale.

3.1.2 Hierarchical Mixture of expert

Una generalizzazione del modello *mixture of expert* (§ 3.1.1) è costituita dall'architettura che prende il nome di *Hierarchical Mixtures of Experts* (Jordan & Jacobs, 1995).

L'architettura *mixtures of experts* risolve un problema complesso, demandando a diversi moduli la soluzione di problemi più semplici, ciascuno relativo ad una determinata regione dello spazio di ingresso. L'approccio *hierarchical mixtures of experts* muove da questo principio e lo applica ricorsivamente generando una struttura ad albero che partiziona lo spazio di ingresso a più livelli di risoluzione. I nodi terminali di tale struttura sono occupati da *expert network* che hanno il compito di approssimare localmente la funzione in esame. I nodi non terminali, sono *gating network* che risolvono, a diversi livelli, problemi di arbitraggio tra le varie parti della struttura. L'interpretazione matematica e probabilistica di questa struttura gerarchica è essenzialmente la stessa che è stata presentata per descrivere il modello non gerarchico. Per semplicità di trattazione, ci limitiamo qui a presentare la struttura composta da due livelli gerarchici. In tale caso, il modello probabilistico è da descriversi come segue:

$$p(y|x,\bar{\alpha}) = \sum_j \pi(j|x,\bar{\eta}) \sum_l \pi(l|j,x,\bar{\nu}_j) p(y|x,\bar{\theta}_{jl})\,. \tag{3.5}$$

Questo modello equivale a dare la seguente descrizione operativa del meccanismo che ha determinato la generazione di ogni singola osservazione (x_i, y_i):

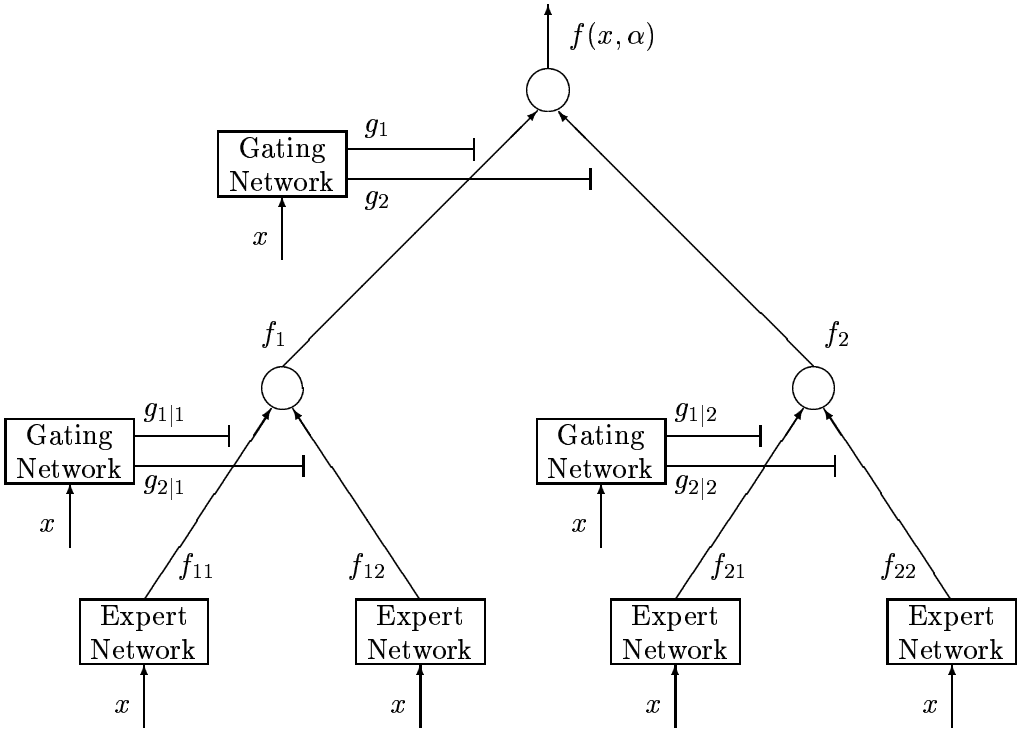


Figura 3.2: Un esempio di architettura *Hierarchical mixture of experts* a due livelli.

dato il valore x_i, è stata scelta una etichetta j con probabilità $\pi(j|x_i, \bar{\eta})$, successivamente un'etichetta l con probabilità $\pi(l|j, x_i, \bar{\nu}_j)$, e finalmente, il valore y_i è stato estratto con probabilità $p(y_i|x_i, \bar{\theta}_{jl})$.

Il modello gerarchico viene quindi addestrato massimizzando il logaritmo della verosimiglianza nella forma:

$$L(Z, \alpha) = \sum_i \ln \sum_j \pi(j|x_i, \eta) \sum_l \pi(l|j, x_i, \nu_i) p(y_i|x_i, \theta_{jl}) \,, \tag{3.6}$$

dove, equivalentemente a ciò che accade nel caso della struttura non gerarchica, le distribuzioni $\pi(j|x, \eta)$ e $\pi(l|j, x, \nu_i)$, sono modellizzate da dei *gating network* attraverso le funzioni $g_j(x)$ e $g_{j|x}(x)$ rispettivamente, mentre gli *expert metwork* hanno il compito di calcolare il valore atteso delle distribuzioni $p(y|x, \theta_{jl})$.

L'addestramento di un'architettura *hierarchical mixtures of experts* può essere convenientemente effettuato impiegando l'algoritmo EM (Jordan & Jacobs, 1994).

3.2 I modelli Fuzzy

3.2.1 Modelli Linguistici

I modelli matematici che vengono utilizzati per descrivere fenomeni fisici possono assumere forme diverse: un modello può essere una semplice equazione algebrica, un sistema di equazioni differenziali, una macchina a stati finiti, etc. Negli ultimi anni una famiglia di modelli che ha particolarmente attirato l'attenzione della comunità scientifica (e non), è quella che dà una rappresentazione della realtà attraverso una collezione di *regole fuzzy*, cioè di regole *se-allora* che descrivono qualitativamente relazioni logiche tra variabili:

se *la pressione è alta* **allora** *il volume è piccolo.*

In questo esempio, *pressione* e *volume* sono variabili linguistiche, e i valori *alta* e *piccolo* che tali variabili possono assumere, sono valori linguistici caratterizzati da *membership function* (Zadeh, 1973). Un modello costituito da un insieme di regole come quella vista, dove sia il termine *antecedente* che quello *conseguente* rappresentano in maniera qualitativa delle grandezze attraverso *insiemi fuzzy* (Zadeh, 1965), si dice essere un *modello linguistico* (Mamdani, 1977).

Il primo obiettivo dell'approccio linguistico alla modellistica fuzzy è quello di definire un metodo e degli strumenti per rappresentare funzioni matematiche, attraverso regole di produzione del tipo di quelle utilizzate nei sistemi esperti (Zadeh, 1989). L'aspetto più innovativo di questo approccio consiste nella sua capacità di ridurre la distanza tra la la realtà matematica e il linguaggio umano, fornendo un'interfaccia tra i due che consente di amalgamare la natura continua del primo con quella discreta del secondo. Lo scopo dell'approccio fuzzy è quindi, in breve, quello di realizzare uno strumento per "calcolare con le parole" (Zadeh, 1973). Adottando questo punto di vista, l'attenzione deve essere posta essenzialmente sulla leggibilità del modello e del processo che porta alla sua definizione, e non sulla sua dimensione o sulle sue capacità di approssimazione (Bersini & Bontempi, 1996).

3.2.2 L'approccio Fuzzy alla modellistica

Un'altro approccio, quello che prenderemo in considerazione nella nostra trattazione, considera la struttura del modello fuzzy come uno srumento per approssimare una funzione non nota attraverso la composizione di semplici modelli locali, generalmente polinomi di basso grado. Quando i modelli locali considerati sono lineari, il modello risultante è una approssimazione lineare a tratti, caratterizzata da transizioni *soft* tra i vari approssimatori locali (Takagi & Sugeno, 1985). Adottando questo approccio, la leggibilità del modello passa

in secondo piano e gli aspetti a cui viene prestata più attenzione sono quelli legati alle prestazioni e alla capacità di approssimazione.

I modelli che analizzeremo sono caratterizzati da regole in cui il termine antecedente è una proposizione fuzzy che coinvolge la variabile di ingresso x, mentre il conseguente è costituito da una funzione non-fuzzy, che lega il valore della variabile di uscita y, all'ingresso. Un esempio di come è possibile utilizzare regole di questo tipo, è dato dalla seguente:

$$\textbf{se}\textit{la velocità è elevata } \textbf{allora } \textit{forza} = k \cdot (\textit{velocità})^2, \tag{3.7}$$

che descrive la forza di attrito a cui è sottoposto un corpo che si muove nell'aria a velocità elevata.

Modelli composti da regole che hanno la struttura della (3.7), sono noti in letteratura con il nome di modelli di Takagi-Sugeno. In tali modelli, ogni regola può essere vista come una descrizione locale del fenomeno in esame: il conseguente della regola è un vero e proprio modello locale, mentre la proposizione antecedente indica quell'insieme fuzzy di valori dell'ingresso x per cui la regola in questione è applicabile (Takagi & Sugeno, 1985).

La struttura di un modello di Takagi-Sugeno

Un generico modello di Takagi-Sugeno è composto da m regole:

$$R_j : \textbf{se } x \in A_j \textbf{ allora } f_j = f_j(x, \theta_j) \qquad \text{con } j = 1, 2, \ldots, m. \tag{3.8}$$

Dove l'operatore '$\in$' è l'operatore di appartenenza fuzzy, e A_j è un insieme fuzzy definito dalla membership function multivariata:

$$\mu_{A_j}(x) : \Re^d \mapsto [0, 1].$$

La proposizione antecedente "$x \in A_j$" è tipicamente espressa come combinazione logica di proposizioni più semplici, ciascuna riguardante una sola delle componenti del vettore x. La singola regola può quindi essere riscritta come:

$$R_j : \textbf{se } x^{(1)} \in A_j^1 \wedge x^{(2)} \in A_j^2 \wedge \cdots \wedge x^{(d)} \in A_j^d \textbf{ allora } v_j = f_j(x, \theta_j). \tag{3.9}$$

Dove $x^{(l)}$ è la componente l-esima del vettore x, gli operatori '$\in$' e '$\wedge$' sono da intendere in termini fuzzy, e A_j^l è un insieme fuzzy definito dalla membership function univariata:

$$\mu_{A_j^l}\left(x^{(l)}\right) : \Re \mapsto [0, 1].$$

Se il modello di Takagi-Sugeno è definito come nella (3.8), per un dato un valore x dell'ingresso, il grado in cui la regola j-esima contribuisce all'approssimazione è determinato dall valore di verità $\beta_j(x)$ dato dal grado di appartenenza del punto x all'insieme A_j:

$$\beta_j(x) = \mu_{A_j}(x). \tag{3.10}$$

Quando l'antecedente è espresso come combinazione congiuntiva di più proposizioni semplici come nella (3.9), il grado in cui la regola j-esima deve partecipare alla predizione è dato da una combinazione dei gradi di appartenenza delle singole componenti $x^{(l)}$ dell'ingresso ai rispettivi insiemi A_j^l:

$$\beta_j(x) = \mu_{A_j^1}\left(x^{(1)}\right) \times \mu_{A_j^2}\left(x^{(2)}\right) \times \cdots \times \mu_{A_j^d}\left(x^{(d)}\right). \tag{3.11}$$

Si è usato quì il prodotto algebrico per combinare i diversi gradi di appartenenza anche se questa non è l'unica scelta possibile. Una alternativa alla (3.11) è quella che pone $\beta_j(x)$ uguale al minimo dei gradi di appartenenza $\mu_{A_j^l}\left(x^{(l)}\right)$. Più in generale, per combinare i valori di verità di più proposizioni fuzzy, è possibile utilizzare una qualsiasi t-norma (Bandemer & Gottwald, 1995).

Nelle applicazioni, le membership function sono comunemente descritte da funzioni parametriche:

$$\mu_{A_j^l}\left(x^{(l)}\right) = \mu_{jl}\left(x^{(l)}, \eta_{jl}\right),$$

dove η_{jl} è un vettore di parametri che definisce le casatteristiche della membership function associata alla j-esima regola e alla l-esima componente del vettore di ingresso. In generale si ha quindi:

$$\beta_j(x, \eta_j) = \prod_{l=1}^{d} \mu_{jl}\left(x^{(l)}, \eta_{jl}\right),$$

con $\eta_j = \left[\eta_{j1}^T, \eta_{j2}^T, \ldots, \eta_{jd}^T\right]^T$ vettore di tutti i parametri che descrivono l'antecedente della regola j-esima. Il valore $\beta_j(x, \eta_j)$ prende il nome di *grado di attivazione* della regola j.

Per quanto riguarda i termini conseguenti di ciascuna regola, la configurazione classica di un modello di Takagi-Sugeno prevede che i modelli $f_j(x, \theta_j)$ siano istanze di una unica classe parametrica e condividano quindi la stessa struttura: si differenziano l'uno dall'altro solo perchè caratterizzati da diversi valori del vettore θ_j (Babuška, 1997). Comunemente vengono adottati modelli lineari in forma affine:

$$f_j(x, \theta_j) = \theta_j^T \tilde{x}. \tag{3.12}$$

Posto $\tilde{x} = \left[x^T, 1\right]^T$, la (3.12) è l'equazione di un iperpiano nello spazio $\Re^d \times \Re$ di ingresso-uscita.

La stima del valore y assunto dalla funzione in esame per un certo ingresso x, è data da:

$$f(x, \alpha) = \frac{\sum_{j=1}^{m} \beta_j(x, \eta_j) f_j(x, \theta_j)}{\sum_{j=1}^{m} \beta_j(x, \eta_j)}. \tag{3.13}$$

Dove $\alpha = \left[\theta_1^T, \theta_2^T, \ldots, \theta_m^T, \eta_1^T, \eta_2^T, \ldots, \eta_m^T\right]^T$ è il vettore di tutti i parametri che descrivono un modello fuzzy di Takagi-Sugeno.

La struttura così definita può essere convenientemente utilizzata per approssimare una funzione complessa utilizzando una collezione di approssimatori locali. Il conseguente (3.12) di ciascuna regola è un approssimatore locale della funzione in esame. Gli antecedenti delle varie regole realizzano una partizione fuzzy dello spazio di input e svolgono il duplice compito di definire la regione di applicabilità del modello a loro associato, e di realizzare il meccanismo di combinazione delle approssimazioni fornite dai diversi approssimatori locali (figura 3.3).

Da un punto di vista logico, gli antecedenti hanno quindi lo stesso ruolo che nel modello *mixture of experts* ha il *gating network* (§ 3.1.1). La differenza sostanziale tra le due strutture sta nella diversa lettura che viene data del meccanismo di coordinamento tra i vari modelli locali. Adottando il modello *mixture of experts*, si abbraccia il paradigma probabilistico: dato un ingresso x, il *gating network* ha il compito di stimare, per ciascun esperto locale j, la probabilità che la quantità $f_j(x, \theta_j)$, calcolata dal j-esimo esperto stesso, sia la corretta predizione dell'uscita y. Tali probabilità sono quindi utilizzate per pesare le diverse predizioni fornite dai vari esperti, dando più credito alla predizione fornita dall'esperto che con probabilità maggiore è in grado di calcolare il valore y, e meno alle altre. L'approccio fuzzy è invece legato ad un approccio possibilistico (Zadeh, 1993) (Dubois & Prade, 1982): dato un ingresso x, il compito delle membership function è quello di determinare in che grado i vari approssimatori locali devono partecipare alla predizione del valore y.

Nei loro risvolti pratici i due paradigmi della probabilità e della possibilità[1], portano a modelli differenti. Ragionando in termini probabilistici si vuole addestrare un modello in cui ogni punto dello spazio di input sia trattabile da uno e un solo esperto. L'obiettivo ultimo è cioè quello di determinare una struttura in cui, per ogni un punto x, esista un esperto che abbia probabilità uno di essere in grado di trattare il caso e quindi di fornire una predizione di y. Questo approccio parte in sostanza dall'assunzione che esista l'esperto che è in grado di trattare il caso x, ma che non sia noto quale sia tale esperto. Nell'approccio possibilistico invece si parte dall'ipotesi che ogni punto dello spazio di ingresso appartenga contemporaneamente, e in gradi diversi (Giles, 1993), a più di uno degli insiemi dei casi trattabili da ciascun approssimatore locale. Di conseguenza, seguendo questo secondo approccio, si vuole determinare la struttura che sappia individuare, dato un punto x, la giusta combinazione di modelli locali che sia in grado di predire y.

[1]Per una più dettagliata discussione sulle differenze e sui punti di contatto tra l'approccio probabilistico e quello fuzzy si veda ad esempio il numero speciale *IEEE Transactions on Fuzzy Systems*, **2**(1), 1994.

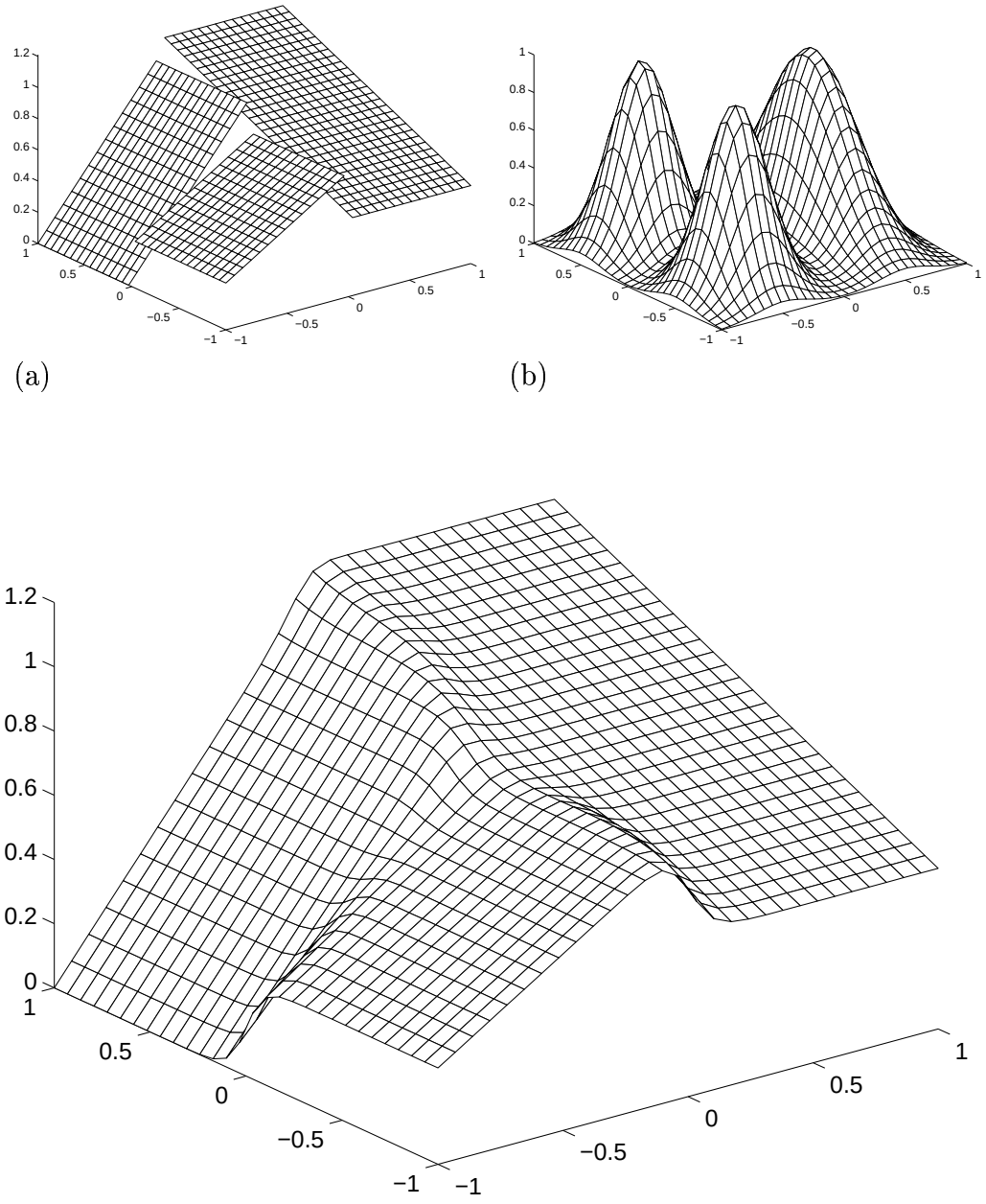


Figura 3.3: Un modello di Takagi-Sugeno è una combinazione *smooth* di una collezione di modelli locali. Nell'esempio qui presentato, il modello è ottenuto combinando tre approssimatori lineari attraverso delle *membership functions* gaussiane.

3.2.3 Dal Fuzzy al Neuro-Fuzzy

L'approccio neuro-fuzzy nasce dalla composizione di due approcci paralleli alla modellistica non-lineare: quello neurale e quello fuzzy. Le reti neurali sono state sviluppate inizialmente come un'emulazione del cervello umano. L'obiettivo era quello di costruire una macchina che riproducesse il funzionamento di neuroni e connessioni sinaptiche, e che fosse capace di immagazzinare dati, imparare, e recuperare l'informazione acquisita. La fuzzy logic nasce invece con l'obiettivo di emulare il ragionamento umano attraverso strutture simboliche derivate dal linguaggio naturale.

L'approccio neuro-fuzzy mantiene la struttura del modello dell'approccio fuzzy[2], ma descrive tale struttura attraverso un grafo del tipo di quelli adoperati per descrivere reti neurali. La rappresentazione di un modello fuzzy attraverso una rete neurale, consente all'approccio neuro-fuzzy di importare dalla ricerca effettuata nell'ambito delle reti neurali, una serie di algoritmi per *imparare* i parametri del modello a partire dai dati.

Un modello neuro-fuzzy deve quindi essere essenzialmente visto come uno schema per rappresentare funzioni come composizione di modelli locali, dove sia i parametri di tali modelli locali, sia i parametri che ne determinano le modalità di combinazione, sono appresi a partire da dati.

L'approccio neuro-fuzzy presenta una serie si caratteristiche interessanti. Sottolineamo in particolare la possibilità di integrare facilmente all'interno del processo di apprendimento conoscenza a priori sul comportamento della funzione in esame in determinate regioni dello spazio di ingresso. Inoltre un modello neuro-fuzzy conserva una certa leggibilità *a posteriori*: è infatti possibile esaminare i parametri del modello identificato a partire dai dati, ed estrarre informazioni sul comportamento che la funzione in esame mostra nelle varie regioni dello spazio di ingresso. A questo si aggiunge il fatto che sono riformulabili, relativamente al modello neuro-fuzzy, una serie di metodologie e di risultati teorici notevoli ottenuti nell'ambito della ricerca sulle reti neurali e in altri settori affini.

Tra i risultati teorici più significativi occorre ricordare la dimostrazione dell'equivalenza funzionale tra le *radial basis function networks* e una sottoclasse dei modelli fuzzy caratterizzata dal impiego di modelli locali costanti (Jang & Sun, 1993).

3.3 Local Weighted Regression

3.3.1 L'approccio lazy alla modellistica

Lazy Learning è il nome di una famiglia di metodi di apprendimento che si

[2]Più precisamente la struttura adottata è quella del modello di Takagi-Sugeno (§ 3.2.2)

differenziano dai metodi tradizionali per il fatto che non prevedono una vera e propria fase di identificazione di un modello distinta da una fase di valutazione. Un metodo lazy ritarda ogni operazione sui dati fintanto che non viene richiesta la stima del valore assunto in uno specifico punto dalla funzione in esame. Nell'approccio lazy, il *'modello'* del fenomeno[3] è costituito dall'insieme stesso delle osservazioni disponibili. La fase di *identificazione* si riduce quindi ad una banale memorizzazione degli esempi in un *data base*. Per sottolineare la differenza tra questo approccio e i metodi tradizionali per quanto riguarda il tipo di rappresentazione data del fenomeno in esame, si usa riferirsi al lazy learning usando l'espressione *Memory-Based Learning* mentre gli altri metodi vengono definiti *Model-Based*.

Dal punto di vista pratico, adottare l'approccio lazy significa definire un algoritmo che, dato un valore x_*, consenta di estrarre da un insieme di esempi Z, una stima del valore y_*. In letteratura esistono diverse forme di lazy learning. Una di queste prevede che gli esempi più prossimi a x_* forniscano ciascuno una stima del valore y_*. Una stima *collettiva* è ottenuta pesando le varie stime *individuali* in base a criteri di rilevanza e di affidabilità degli esempi che le hanno fornite. Tipicamente la rilevanza di un esempio è una funzione della distanza dell'esempio stesso dal punto x_* in una data metrica, e l'affidabilità ha a che vedere con la capacità di predizione mostrata dall'esempio in altre situazioni simili.

In questa tesi ci occupiamo di una diversa forma di lazy learning che prende il nome di *regressione locale*. Tale forma prevede che la stima del valore y_* assunto in x_*, sia ottenuta utilizzando un semplice modello locale che approssimi la funzione di regressione in un intorno di x_*. Operativamente il lazy learning, nella sua forma che prende il nome di regressione locale, può essere descritto nel seguente modo. Dato un punto x_*, un insieme Z di esempi ed un intero k, parametro del metodo, viene individuato il sottoinsieme $Z_* = \{(x_1, y_1), (x_2, y_2), \ldots, (x_k, y_k)\}$ composto dai k esempi appartenenti a Z, che risultano essere i più vicini a x_* in una data metrica. Utilizzando gli esempi in Z_*, pesati in funzione della loro vicinanza a x_*, viene identificato un modello lineare[4] $f_*(x, \alpha)$. La stima di y_* fornita dal metodo è data dal valore $f_*(x_*, \alpha)$, assunto nel punto x_* da tale modello.

La *regressione locale* è una generalizzazione di altre forme di apprendimento locale:

Media pesata locale: il modello locale considerato è una semplice costante il cui valore è la media delle uscite dei k esempi più vicini, pesate in maniera decrescente all'aumentare delle distanze di tali esempi da x_*.

[3] Usiamo il termine modello in senso lato, con il significato di rappresentazione *generica* di un fenomeno.

[4] Lineare nei parametri.

Formalmente:

$$\hat{y}_* = \frac{\sum_{i=1}^{k} K\big(\|x_* - x_i\|\big) y_i}{\sum_{i=1}^{k} K\big(\|x_* - x_i\|\big)} ,$$

dove k è un parametro del metodo, e $\{x_1, x_2, \ldots, x_k\}$ sono i k punti appartenenti a Z più vicini a x_*. La norma $\|x_* - x_i\|$ è determinata dalla metrica definita sullo spazio degli ingressi, e $K(u)$ prende il nome di funzione *kernel*, ed è una funzione che decresce al crescere di u.

Nearest neighbor: può essere visto come un caso degenere di *weighted average*. Merita comunque una trattazione separata in quanto capostipite di tutta una famiglia di metodi. Adottando l'approccio *nearest neighbor*, la stima del valore y_*, assunto dalla funzione in esame nel punto x_*, è data dal valore y_{nn} assunto in x_{nn}, dove x_{nn} è l'esempio più vicino ad x_*, in una metrica assegnata:

$$\hat{y}_* = y_{nn}$$

con

$$\|x_* - x_{nn}\| \leq \|x_* - x_i\| \qquad \forall x_i \in Z ,$$

dove la norma $\|x_* - x_i\|$ considerata, è determinata dalla metrica definita sullo spazio degli ingressi.

La regressione locale estende questi metodi nel senso che considera, come approssimatori locali, non solo modelli costanti ma generiche funzioni $f_*(x, \alpha)$, purché lineari in α.

3.3.2 Stima locale della funzione di regressione

Per poter utlizzare la *regressione locale* come stimatore del valore assunto dalla funzione in esame nel punto x_*, è necessario definire e tarare alcuni parametri strutturali, in base ad assunzioni sulla densità $p(y|x)$ caratterizzante il meccanismo che ha generato l'insieme Z. Tali parametri sono: la forma della funzione kernel, la larghezza di banda, la famiglia parametrica $f_*(x, \alpha)$ da usare come approssimatore locale, e la funzione di costo $C\big(y, f_*(x, \alpha)\big)$ in base alla quale identificare i modelli locali. Nel seguito di questo paragrafo definiremo questi parametri strutturali che caratterizzano il metodo lazy learning e illustreremo alcune delle soluzioni più comunemente adottate.

La funzione di costo

Per la selezione della funzione di costo da utilizzare nella stima locale della funzione di regressione, valgono le stesse considerazioni fatte trattando il caso globale. Identificare un modello locale della funzione di regressione significa stimare $E[y|x_*]$, valore atteso di y dato x_*, utilizzando solo esempi che cadono in una regione centrata nel punto x_* stesso. La stima della funzione di regressione viene effettuata considerando la seguente cifra di costo associata ad una predizione:e sfruttando l'informazione contenuta nei dati

$$C\big(y, f_*(x,\alpha)\big) = \big(y - f_*(x,\alpha)\big)^2 .$$

Nel caso in cui la densità $p(y|x)$ sia fortemente asimmetrica, o in ogni caso molto diversa da una normale, è corretto considerare altre funzioni di costo.

In questa tesi, come già affermato, ci limitiamo a considerare il problema della stima della regressione, quindi consideriamo accettabile l'ipotesi che la variabile y sia distribuita qualitativamente come una gaussiana per ogni valore di x.

La funzione kernel

Tipicamente, identificando un modello della funzione di regressione che dovrà essere valutato nel punto x_*, si vuole dare più importanza al comportamento della funzione nelle vicinanze di x_* e meno al comportamento in zone più lontane. Per ottenere ciò, è possibile associare ad ogni esempio (x_i, y_i) in Z_*, uno scalare w_i funzione della distanza tra x_* e x_i, e utilizzare tale scalare per pesare il contributo dell'i-esimo esempio nell'identificazione del modello locale. I valori w_i sono normalmente assegnati attraverso una funzione kernel:

$$w_i = \sqrt{K\big(\|x_* - x_i\|\big)} ,$$

definita in maniera tale che:

$$K(u_1) \leq K(u_2) \qquad \text{se } |u_1| > |u_2| .$$

Per ottenere stime *smooth*, cioè stime $\hat{y}_*$ che non cambiano bruscamente per piccoli spostamenti di x_*, vengono adottate funzioni kernel $K(u)$, il cui valore decresca dolcemente al crescere di u.

La larghezza di banda

La larghezza di banda è la dimensione della regione centrata in x_*, che viene presa in considerazione per la stima del modello locale. Più è grande è la grandezza di banda, maggiore è il numero di esempi che rientrano nel sottoinsieme Z_*. La scelta della larghezza di banda è da effettuarsi insieme a quella

della famiglia $f_*(x, \alpha)$ con l'obiettivo di produrre una stima il più possibile *smooth*, ma che allo stesso tempo non introduca una eccessiva distorsione, e riesca quindi a cogliere la relazione esistente tra l'ingresso x e l'uscita y.

Sia il problema della scelta della larghezza di banda che quello, come vedremo, della scelta della famiglia delle funzioni approssimanti, possono essere illustrati nei termini del dilemma bias/varianza. Fissata una famiglia $f_*(x, \alpha)$, una grande larghezza di banda porta a stime stabili, caratterizzate cioè da una piccola varianza. Tali stime però sono affette da bias in quanto ottenute identificando un modello locale su una regione trppo grande, all'interno della quale il comportamento della funzione non nota potrebbe essere troppo complesso per essere spiegato da un membro della famiglia assegnata. Al contrario, una piccola larghezza di banda consente un piccolo bias ma una elevata varianza, in quanto gli esempi contenuti nella regione considerata sono troppo pochi per identificare univocamente una funzione tra quelle appartenenti alla famiglia data.

La famiglia degli approssimatori locali

La forma di lazy learning che prendiamo in considerazione in questa tesi, prevede che la funzione di regressione sia approssimata localmente da una funzione appartenente ad una famiglia parametrica. Come abbiamo già detto, l'approssimatore locale deve essere lineare nei parametri, questo per permetterne una rapida identificazione e per consentire, come vedremo in seguito, l'utilizzo di una serie di strumenti applicabili solo a modelli lineari. La generica funzione lineare nei parametri può essere scritta nella forma seguente:

$$f_*(x, \alpha) = \sum_j \alpha_j \phi_j(x) \,, \tag{3.14}$$

dove α_j è la j-esima componente del vettore α, e $\phi_j(x)$ è una generica funzione scalare del vettore x. Normalmente vengono usati come approssimatori locali, dei modelli costanti o lineari, più raramente quadratici, e difficilmente si considerano polinomi di grado superiore al terzo. Genericamente, quindi, la famiglia $f_*(x, \alpha)$ considerata, è la famiglia dei polinomi di grado p che si ottiene dalla (3.14), considerando come funzioni $\phi_j(x)$, solo monomi di grado minore o uguale a p, ottenuti dalle varie componenti di x secondo la seguente:

$$\phi_j(x) = \prod_{l=1}^{d} \left(x^{(l)}\right)^{\gamma_{jl}}$$

dove $x^{(l)}$ è la l-esima componente del vettore $x \in \Re^d$, e le quantità γ_{jl} sono interi tali che $\gamma_{jl} \geq 0$ e $\sum_{l=1}^{d} \gamma_{jl} \leq p$.

La scelta della famiglia approssimante si riduce quindi alla scelta del grado p del polinomio da usare come approssimatore locale. Tale scelta, come già

anticipato nel paragrafo precedente, è una scelta di compromesso. A parità di numero di esempi Z_* disponibili per l'identificazione, polinomi di grado elevato riescono ad adattarsi meglio ai dati e quindi presentano una bassa componente di bias, soffrono però di un'elevata varianza in quanto il valore dei parametri identificati è molto sensibile a piccole variazioni di Z_*. Viceversa, polinomi di grado basso sono meno elastici e meno potenti, rischiano di non essere in grado di cogliere la complessità della funzione in esame ma, essendo descritti da un numero limitato di parametri, hanno una componente di varianza ridotta.

3.3.3 Lazy learning adattativo

In gran parte della letteratura riguardante il *lazy learning* e i metodi di regressione locale, i parametri strutturali del metodo sono fissati globalmente e in maniera euristica. In realtà, data la semplicità dei modelli utilizzati come approssimatori locali della funzione di regressione, altre strade sono praticabili. E' possibile, ad esempio, estendere il metodo presentato fornendo delle procedure che consentano di determinare automaticamente il valore ottimo dei parametri strutturali.

Nel caso del lazy learning, non esistendo un modello globale, e visto che tutte le operazioni effettuate per stimare il valore y_* coinvolgono solo una regione limitata centrata in x_*, ha senso che anche la fase di identificazione strutturale (§ 2.4) venga eseguita localmente sfruttando l'informazione contenuta nei dati Z_* che cadono nell'intorno di x_*. In questo caso il parametro strutturale m rispetto al quale ottimizzare la struttura potrebbe essere ad esempio il grado del polinomio utilizzato come approssimatore locale, la larghezza di banda o un qualche parametro che descriva la funzione kernel. La ricerca della struttura ottima, data la linearità dei modelli locali adottati, può essere convenientemente eseguita utilizzando la *PRESS statistic* (§ 2.3.3) come criterio di valutazione della capacità di generalizzazione di un modello di una data complessità m (Atkeson *et al.*, 1996).

Capitolo 4

Neuro-fuzzy inference system

Il modello neuro-fuzzy considerato in questo capitolo è costituito dalla composizione *smooth* di approssimatori locali, secondo lo schema proposto da Takagi e Sugeno (§ 3.2.2). Mostreremo di seguito una collezione di metodi per identificare, a partire da un insieme di esempi, tutti i parametri del modello: sia quelli che descrivono i singoli modelli locali, sia quelli che ne determinano la localizzazione spaziale nel dominio di ingresso. Anche se faremo qui riferimento al caso in cui l'unica informazione disponibile è costituita dagli esempi, ci occuperemo quindi di modellistica *black box*, lo schema presentato può essere utilizzato anche per identificare modelli *gray box*. Infatti è sempre possibile fissare a priori il valore dei parametri conosciuti e/o imporre una determinata struttura e collocazione ai modelli locali che approssimano la funzione in esame in quelle regioni dello spazio di ingresso relativamente alle quali sono dati modelli di primo principio (Bontempi & Bersini, 1997).

Questo capitolo è organizzato in due sezioni. La prima riguarda la fase di identificazione parametrica nella quale, avendo fissato il numero di approssimatori locali di cui si vuole sia composto il modello, vengono individuati i valori ottimi dei parametri che descrivono i modelli locali e le loro modelità di interazione. La seconda sezione riguarda invece la fase di identificazione strutturale il cui scopo è determinare il numero ottimo di approssimatori locali da impiegare.

4.1 Identificazione dei parametri

4.1.1 La struttura parametrica

Dato un valore di x, il coefficiente che misura il grado in cui il j-esimo approssimatore locale contribuisce alla stima di y è dato, secondo l'interpretazione delle

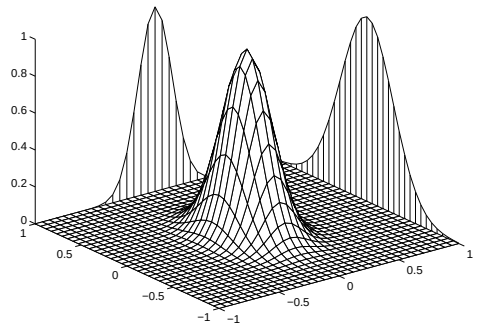


Figura 4.1: Gaussiana bivariata ottenuta dal prodotto di due membership function gaussiane monovariate, funzione ciascuna di una componente del vettore x.

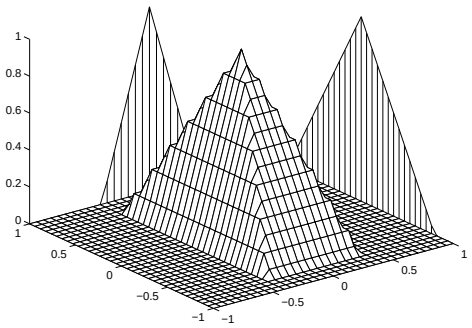


Figura 4.2: Una funzione di appartenenza tetraedrica si ottiene ponendo $\beta_j = \min(\mu_{j1}, \mu_{j2})$, dove μ_{j1} e μ_{j2} sono funzioni di monovariate e triangolari.

regole fuzzy vista nel paragrafo 3.2.2, dalla seguente equazione:

$$\beta_j(x, \eta_j) = \prod_{l=1}^{d} \mu_{jl}\left(x^{(l)}, \eta_{jl}\right), \tag{4.1}$$

dove le funzioni μ_{jl} sono generiche funzioni parametriche. Tra le innumerevoli funzioni possibili consideriamo i casi in cui le μ_{jl} siano gaussiane oppure triangolari. Risulta quindi:

$$\mu_{jl}\left(x^{(l)}, \eta_{jl}\right) = e^{-\frac{\left(x^{(l)} - c_{jl}\right)^2}{\sigma_{jl}^2}},$$

oppure

$$\mu_{jl}\left(x^{(l)}, \eta_{jl}\right) = \max\left(0, 1 - \frac{|x^{(l)} - c_{jl}|}{2\sigma_{jl}}\right).$$

Se in particolare si considerano funzioni μ_{jl} gaussiane, la (4.1) può essere vista, nello spazio di ingresso $\Re^d$, come una gaussiana d-dimensionale con gli assi paralleli agli assi coordinati (figura 4.1):

$$\beta_j(x, \eta_j) = e^{-(x-c_j)^T {\Sigma_j}^{-1} (x-c_j)}, \tag{4.2}$$

dove c_j è il vettore che individua, nello spazio di ingresso, il punto in cui è centrato il j-esimo approssimatore locale, e Σ_j è una matrice diagonale in cui l'elemento l-esimo della diagonale è occupato dal parametro σ_{jl}^2. Il vettore

dei parametri η_j che descrive le condizioni di applicabilità del j-esimo modello locale è quindi $\eta_j = \left[c_j^T, \mathrm{diag}(\Sigma_j)^T\right]^T$.

Una possibile estensione consiste nel considerare funzioni membership gaussiane i cui assi siano genericamente orientati, gaussiane quindi descritte da matrici Σ_j simmetriche piene (Babuška, 1997). Tale estensione consente di trattare al meglio quelle situazioni in cui le diverse componenti dell'ingresso siano correlate. L'algoritmo di ottimizzazione che utilizzaremo per identificare modelli in cui le matrici Σ_j sono diagonali può essere facilmente esteso al caso generico in cui le Σ_j siano piene (Bersini *et al.*, 1997b).

Per quanto riguarda gli approssimatori locali, consideriamo qui il caso in cui questi siano funzioni lineari del vettore di ingresso x:

$$f_j(x, \theta_j) = \theta_j^T \tilde{x}. \tag{4.3}$$

dove si sia posto $\tilde{x} = \left[x^T, 1\right]^T$. Modelli neuro-fuzzy costituiti da approssimatori locali costanti si ottengono, come caso particolare, semplicemente ponendo a zero tutti gli elementi del vettore θ_j, ad esclusione dell'ultimo. Data la (4.3), l'equazione del modello di Takagi-Sugeno può essere riscritta come segue:

$$f(x, \alpha) = \sum_{j=1}^{m} \frac{\beta_j(x, \eta_j)}{\sum_{k=1}^{m} \beta_k(x, \eta_k)} \tilde{x}^T \theta_j. \tag{4.4}$$

Considerando il vettore $B = B(x, \eta)$ con $B = \left[B_1^T, B_2^T, \ldots, B_m^T\right]^T$, dove:

$$B_j = \frac{\beta_j(x, \eta_j)}{\sum_{k=1}^{m} \beta_k(x, \eta_k)} \tilde{x},$$

si può scrivere:

$$f(x, \alpha) = B^T \theta. \tag{4.5}$$

La (4.5) mostra che il modello fuzzy di Takagi-Sugeno, pur essendo globalmente non lineare nei parametri, risulta essere lineare rispetto ad alcuni di questi.

Una possibile alternativa alla (4.4) è ottenibile componendo i contributi dei vari approssimatori locali usando come coefficienti direttamente i valori assunti dalle membership functions:

$$f(x, \alpha) = \sum_{j=1}^{m} \beta_j(x, \eta_j) \tilde{x}^T \theta_j. \tag{4.6}$$

In questo caso si parla di antecedenti non normalizzati, mentre la (4.4) rappresenta un modello in cui gli antecedenti sono normalizzati[1] (Shorten & Murray-Smith, 1994).

[1] Il software realizzato consente all'analista di scegliere se adottare un'architettura normalizzata o non normalizzata, così come consente di scegliere se utilizzare funzioni di appartenenza gaussiane o triangolari.

4.1.2 L'algoritmo di ottimizzazione

L'algoritmo di identificazione dei parametri che abbiamo implementato, sfrutta la linearità di $f(x, \alpha)$ in θ. Il problema di ottimizzazione viene impostato come segue. Si considera come cifra di merito da minimizzare la somma dei quadrati degli errori commessi sul training set (§ 2.2):

$$J(\alpha) = \frac{1}{n} \sum_{i=1}^{n} C\big(y_i, f(x_i, \alpha)\big). \tag{4.7}$$

L'obiettivo del processo di ottimizzazione è individuare il vettore $\hat{\alpha}$ tale che:

$$\hat{\alpha} = arg \ \min_{\alpha} J(\alpha). \tag{4.8}$$

Essendo $f(x, \alpha)$ una funzione non lineare del vettore α, il problema di ottimizzazione deve essere affrontato necessariamente utilizzando un metodo numerico (§ 2.2.2). La ricerca nello spazio dei parametri del vettore $\hat{\alpha}$ puó essere affrontata utilizzando un metodo che, similmente al metodo del gradiente, dal valore corrente del parametro muova ad ogni iterazione un passo in una direzione lungo la quale $J(\alpha)$ decresce. Dato $\alpha^{(\tau)}$, valore del parametro al passo τ, un modo efficiente per determinare $\alpha^{(\tau+1)}$ consiste nello scomporre il passo da effettuare in due componenti, una appartenente al sottospazio del parametro η, rispetto al quale il problema di ottimizzazione è non lineare, e l'altra appartenente al sottospazio di θ, rispetto a cui il problema dato è lineare.

Per evidenziare la diversa natura della dipendenza di $f(x, \alpha)$ dai parametri, useremo nel seguito di questo paragrafo la notazione $f(x, \eta, \theta)$ e $J(\eta, \theta)$ al posto di $f(x, \alpha)$ e $J(\alpha)$ rispettivamente. Utilizzando questa nuova notazione, il generico passo dell'algoritmo che abbiamo implementato, può essere descritto formalmente come segue:

$$\theta^{(\tau+1)} = arg \ \min_{\theta} J\big(\eta^{(\tau)}, \theta\big), \tag{4.9}$$

$$\eta^{(\tau+1)} = arg \ \min_{\eta} J\big(\eta, \theta^{(\tau+1)}\big). \tag{4.10}$$

La (4.10) è un problema di ottimizzazione non-lineare che non ammette una soluzione in forma chiusa. Nella nostra implementazione il problema (4.10) è stato affrontato utilizzando l'algoritmo Levenberg-Marquardt (§ 2.2.2). In realtà, per risolvere il problema (4.8), non è necessario trovare ad ogni passo il valore di η che minimizza la (4.10) ma è sufficiente trovare un valore $\eta^{(\tau+1)}$ tale che:

$$J\big(\eta^{(\tau+1)}, \theta^{(\tau+1)}\big) < J\big(\eta^{(\tau+1)}, \theta^{(\tau)}\big).$$

L'algoritmo di ottimizzazione:

1. $\tau = 0$. Inizializzazione del vettore η ad un valore $\eta^{(0)}$.
2. Ottimizzazione nel sottospazio di θ rispetto a cui il modello è lineare. Il metodo dei minimi quadrati viene utilizzato per valutare il valore $\theta^{(\tau+1)}$ che, tenendo fisso $\eta^{(\tau)}$, minimizza in θ la funzione $J\left(\eta^{(\tau)}, \theta\right)$.
3. Ottimizzazione nel sottospazio di η rispetto a cui il modello è non lineare. Il nuovo valore $\eta^{(\tau+1)}$ viene determinato tenendo fermo il valore di $\theta^{(\tau+1)}$ ed eseguendo 3 passi del metodo Levenberg-Marquardt sulla funzione $J\left(\eta, \theta^{(\tau+1)}\right)$.
4. Se $J\left(\eta^{(\tau+1)}, \theta^{(\tau+1)}\right) < J\left(\eta^{(\tau)}, \theta^{(\tau)}\right)$, si torna al passo 2 con $\tau = \tau+1$.

Figura 4.3: L'algoritmo iterativo utilizzato per identificare un modello fuzzy. Ogni iterazione consiste di un problema di ottimizzazione lineare e di pochi passi di ottimizzazione non-lineare.

Ciò significa che per il nostro scopo è sufficiente eseguire solo alcune iterazioni dell'algoritmo Levenberg-Marquardt senza giungere alla completa minimizzazione in η della $J\left(\eta, \theta^{(\tau+1)}\right)$.

La (4.9) è un problema di ottimizzazione lineare che può essere risolto analiticamente e quindi in un solo passo attraverso il metodo dei minimi quadrati (§ 2.2.1). Si è così scomposto un problema di ottimizzazione non lineare nello spazio del parametro α in più problemi più semplici che consistono ciascuno in un problema di ottimizzazione lineare e in alcuni passi di ottimizzazione non lineare nello spazio del parametro η. Uno schema dell'algoritmo implementato è riportato in figura 4.3

L'algoritmo di ottimizzazione adottato necessita di un valore iniziale $\eta^{(0)}$ per il vettore dei parametri rispetto ai quali la funzione $f(x, \eta, \theta)$ è non lineare. Inizializzare il vettore η significa definire una partizione iniziale dello spazio di ingresso che collochi i vari approssimatori locali, e ne definisca le rispettive regioni di azione. La scelta del valore di inizializzazione è particolarmente critica in quanto da questa dipende la capacità dell'algoritmo di convergere ad un punto di minimo soddisfacente (§ 2.2.2).

Oltre alla soluzione più semplice, che consiste nel'inizializzare casualmente il vettore $\eta^{(0)}$ e quindi distribuire casualmente gli approssimatori sul dominio di interesse, esistono in letteratura altre soluzioni che sfruttano metodi di analisi dei dati collettivamente conosciuti con il nome di *Cluster analysis* (Babuška,

1997). Nel paragrafo seguente introdurremo alcuni di tali metodi e mostreremo come possono essere utilizzati nell'ambito dell'identificazione di modelli fuzzy per individuare quelle regioni dello spazio di ingresso all'interno di ciascuna delle quali la funzione in esame può plausibilmente essere approssimata da un unico modello locale.

4.1.3 Cluster analysis

L'obiettivo della *cluster analysis* è quello di classificare oggetti in base a un criterio di somiglianza e di organizzarli in gruppi. Storicamente il problema del clustering viene definito un problema di apprendimento *non supervisionato* (Duda & Hart, 1973): dato un insieme finito di oggetti, l'obiettivo è quello di *imparare* una classificazione di tali oggetti senza avere informazioni esplicite su come definire dei raggruppamenti degli oggetti dati.

A seconda dell'obiettivo specifico per cui vengono utilizzati i metodi di *cluster analysis*, vengono date diverse definizioni di che cos'è un cluster. In maniera sufficientemente generica, è possibile definire un cluster come un insieme di oggetti i cui elementi sono più *simili* agli altri elementi del cluster stesso, che non ad elementi appartenenti ad altri cluster (Bezdek, 1981). Il concetto di *similitudine* è da intendersi univocamente ed in termini matematici. Identificare cluster nello spazio degli oggetti dati presuppone quindi la definizione formale di una misura di similitudine per tale spazio. Uno dei metodi possibili per individuare ciascuno dei cluster identificati consiste nel definirne un centro (*prototipo*) che sia un punto nello spazio degli oggetti considerati[2].

Nel caso in cui si vogliano utilizzare metodi di *cluster analysis* come supporto all'identificazione di modelli fuzzy a partire da esempi, gli oggetti per cui devono essere definiti dei cluster sono gli esempi disponibili stessi. All'interno di un processo di identificazione parametrica, un algoritmo di clustering può essere convenientemente utilizzato per individuare regioni dello spazio di input in cui posizionare inizialmente i modelli locali, cioè per individuare un valore di inizializzazione per i parametri η_j delle funzioni $\beta_j(x, \eta_j)$ che, per ogni valore di x, determinano il grado in cui ciascun modello locale deve contribuire alla stima di y (§ 4.1.2).

Intuitivamente, la ragione per cui la *cluster analysis* viene utilizzata in questo contesto è che si confida che esempi *simili*, in un dato senso, possano essere descritti dallo stesso modello locale. Nei prossimi paragrafi presenteremo diversi modi di effettuare *cluster analysis* e quindi diversi *criteri di similitudine* tra gli esempi.

[2]Questo non è l'unico modo per far riferimento a un cluster. Per una trattazione completa si veda il testo *"Pattern Classification and Scene Analysis"* (Duda & Hart, 1973).

L'algoritmo c-mean

L'obiettivo dell'algoritmo c-mean è ripartire l'insieme Z degli esempi disponibili in c sotto insiemi A_j, dove c è un parametro dato tale che $2 \leq c < n$ con n uguale al numero di elementi di Z. Usando questo metodo all'interno di un processo di identificazione di un modello fuzzy, c è posto uguale a m cioè al numero di regole di cui si vuole sia composto il modello.

I sotto insieme A_j cercati dal metodo c-mean[3] sono insiemi classici (non fuzzy) che devono soddisfare le seguenti condizioni (Bezdek, 1981):

$$\bigcup_{j=1}^{c} A_j = Z; \tag{4.11}$$

$$A_j \cap A_k = \emptyset, \quad 1 \leq j \neq k \leq c\,; \tag{4.12}$$

$$\emptyset \subset A_j \subset Z, \quad 1 \leq j \leq c. \tag{4.13}$$

Gli insiemi A_j devono collettivamente contenere tutti gli elementi di Z (4.11), essere disgiunti (4.12), e nessuno di essi deve essere vuoto o contenere l'intero insieme Z (4.13).

Per ogni coppia esempio-cluster viene definita la quantità $\mu_{ij} \in \{0, 1\}$ tale per cui se l'i-esimo elemento di Z appartiene al j-esimo cluster, μ_{ij} assume valore 1, mentre assume valore 0 se appartiene ad un altro cluster. Si definisce inoltre la matrice U il cui elemento di posto (i, j) è uguale a μ_{ij}. Formalmente lo spazio delle partizioni possibili dell'insieme Z può essere descritto dall'insieme:

$$M_h = \left\{ U \in \Re^{c \times n} \,\middle|\, \mu_{ij} \in \{0,1\}, \forall i, j;\ \sum_{j=1}^{c} \mu_{ij} = 1, \forall i;\ 0 < \sum_{i=1}^{n} \mu_{ij} < n, \forall j \right\},$$

dove $\Re^{c \times n}$ è lo spazio delle matrici reali $c \times n$ (Bezdek, 1981), e il pedice h di M_h sta per *hard partition*[4].

Adottando l'algoritmo c-mean, i cluster sono descritti da un *prototipo* o centro $v_j \in \Re^d$, la cui posizione viene determinata attraverso un processo di ottimizzazione che contemporaneamente assegna gli esempi ai vari cluster e minimizza una funzione della distanza[5] tra i punti assegnati ad un cluster e il

[3] La notazione e la terminologia che utilizziamo per descrivere il metodo c-mean non è quella con cui è stato originariamente presentato ma è quella introdotta da Bezdek (Bezdek, 1981), che riduce questo algoritmo ad un sotto caso di un più generale algoritmo di clustering fuzzy. Anche se non corretta da un punto di vista filologico, adottiamo questa notazione per uniformità con quella usata in seguito, e perchè ciò consentirà un confronto diretto del metodo c-mean con quelli successivamente presentati.

[4] Non *fuzzy*, sempre nella terminologia introdotta da Bezdek.

[5] La misura di distanza considerata è quella Euclidea.

relativo centro. La cifra di merito considerata è la seguente:

$$V(v,U) = \sum_{i=1}^{n}\sum_{j=1}^{c} \mu_{ij}\|x_i - v_j\|^2.$$

Dove il generico elemento della matrice U è dato da:

$$\mu_{ij} = \begin{cases} 1 & \text{se } \|x_i - v_j\|^2 \leq \|x_i - v_l\|^2 \ \forall l, \\ 0 & \text{altrimenti.} \end{cases}$$

La partizione generata dall'algoritmo k-mean può essere usata per inizializzare il valore di η (§ 4.1.2) ponendo $c_j = v_j$, cioè centrando la j-esima *membership function* nel prototipo del j-esimo cluster, e ponendo Σ_j uguale alla matrice varianza, rispetto a v_j, dei punti appartenenti al j-esimo cluster. Analiticamente:

$$\sigma_{jl}^2 = \frac{\sum_{i=1}^{n} \mu_{ij}(x_i^{(l)} - v_j^{(l)})^2}{\sum_{i=1}^{n} \mu_{ij}},$$

dove $x_i^{(l)}$ e $v_j^{(l)}$ sono le componenti l-esime dei vettori x_i e v_j rispettivamente. Lo scalare σ_{jl}^2 occupa il posto (l,l) della matrice Σ_j mentre gli elementi non appartenenti alla diagonale della matrice Σ_j stessa sono posti uguali a zero (§ 4.1.2).

Fuzzy c-mean

L'algoritmo c-mean può essere esteso considerando cluster A_j che siano sotto insiemi fuzzy (Zadeh, 1965) dell'insieme degli esempi Z. In questo caso i valori μ_{ij} che esprimono l'appartenenza dell' i-esimo elemento di Z al j-esimo cluster, sono da intendersi in termini fuzzy: non assumono più necessariamente valore 0 o 1, ma un valore reale nell'intervallo $[0,1]$. Lo spazio delle partizioni fuzzy di Z può essere descritto formalmente dall'insieme M_f:

$$M_f = \left\{ U \in \Re^{c\times n} \,\middle|\, \mu_{ij} \in [0,1], \forall i,j;\ \sum_{j=1}^{c} \mu_{ij} = 1, \forall i;\ 0 < \sum_{i=1}^{n} \mu_{ij} < n, \forall j \right\},$$

L'esempio i-esimo può dunque appartenere contemporaneamente a più di uno dei sotto insiemi $A_j \subset Z$, con un grado di appartenenza pari a $\mu_{ij} \leq 1$ in ciascuno.

L'estensione fuzzy del concetto di cluster è appropriata e naturale quando si desideri utilizzare, come nel nostro caso specifico, la *cluster analysis* all'interno del processo di identificazione di un modello fuzzy. La partizione generata

L'algoritmo Fuzzy c-mean:

1. $\tau = 0$. Inizializzazione della matrice U ad un valore $U^{(0)} \in M_f$.

2. Calcolo dei prototipi: il prototipo del j-esimo cluster è ottenuto come media degli esempi appartenenti al cluster, pesata in funzione dei relativi gradi di appartenenza.

$$v_j^{(\tau+1)} = \frac{\sum_{i=1}^{n} \left(\mu_{ij}^{(\tau)}\right)^p x_i}{\sum_{i=1}^{n} \left(\mu_{ij}^{(\tau)}\right)^p}$$

3. Aggiornamento della matrice U:

$$\mu_{ij}^{(\tau+1)} = \left[\sum_{k=1}^{c} \left(\frac{\gamma_{ij}}{\gamma_{ik}}\right)^{\frac{2}{p-1}}\right]^{-1}$$

   dove

$$\gamma_{ij} = \|x_i - v_j^{(\tau+1)}\|^2.$$

   Si noti che se per un dato x_i risulta $\gamma_{is} = 0$, con $s \in S \subset \{1, 2, \ldots, c\}$, cioè se il punto x_i si trova sovrapposto a uno o più prototipi, le quantità μ_{is} non possono essere calcolate. In questo caso i valori μ_{is} sono posti casualmente, rispettando i vincoli $\mu_{is} \in [1, 0]$ e $\sum_{s \in S} \mu_{is} = 1$.

4. Se $V\left(v^{(\tau+1)}, U^{(\tau+1)}\right) < V\left(v^{(\tau)}, U^{(\tau)}\right)$, si torna al passo 2 con $\tau = \tau + 1$.

Figura 4.4: L'algoritmo di clustering *fuzzy c-mean* genera una partizione fuzzy dello spazio degli oggetti considerati.

dal metodo di clustering viene utilizzata per inizializzare i parametri delle membership function, cioè di quelle funzioni che hanno il compito di localizzare nello spazio $\Re^d$ di ingresso i vari approssimatori locali, e di determinarne le modalità di sovrapposizione.

L'algoritmo fuzzy c-mean, descrive i c cluster attraverso dei prototipi v_j le cui posizioni nello spazio $\Re^d$ vengono determinate attraverso un processo di ottimizzazione che minimizza la cifra di merito:

$$V(v,U) = \sum_{i=1}^{n}\sum_{j=1}^{c}(\mu_{ij})^p \|x_i - v_j\|^2, \tag{4.14}$$

dove $\|x_i - v_j\|^2$ è il quadrato della distanza euclidea tra l'i-esimo elemento di Z e il centro del j-esimo cluster, e $(\mu_{ij})^p$ pesa tale distanza in funzione del grado di appartenenza dell'i-esimo elemento di Z al j-esimo cluster. Il parametro $p \in [1,\infty)$ influenza il grado di sovrapposizione dei cluster risultanti. Un valore alto del parametro p ha inoltre implicitamente l'effetto di diminuire il contributo in $V(v,U)$ di eventuali punti affetti da rumore. Tali punti infatti hanno tipicamente un basso grado di appartenenza a più cluster: se l'i-esimo elemento di Z è tale per cui, pur essendo $\sum_j \mu_{ij} = 1$, risulta $\mu_{ij} \ll 1\ \forall j$, un elevato valore di p rende trascurabile il coefficiente $(\mu_{ij})^p$ e quindi il contributo del punto i a $V(v,U)$. Lo schema dell'algoritmo fuzzy c-mean è riportato in figura 4.4.

In generale, adoperare un metodo di *clustering* per inizializzare un modello fuzzy significa partizionare lo spazio di ingresso riconoscendo esempi *simili*, collocare un approssimatore locale in ciscuna delle regioni individuate, e ricomporre i vari approssimatori sovrapponendone gli effetti utilizzando delle membership functions fuzzy. In questo senso, la scelta di un metodo di *fuzzy clustering* risulta naturale in quanto il criterio secondo il quale viene effettuata la partizione dello spazio di ingresso, è lo stesso che viene adottato per ricombinare gli approssimatori locali.

Hyperplane Fuzzy Clustering

Inizializzare l'algoritmo di identificazione parametrica utilizzando la partizione generata dai metodi c-mean e fuzzy c-mean, significa sfruttare informazioni riguardanti solo lo spazio di ingresso. I metodi fin'ora visti, partizionano l'insieme $Z = \{z_i\}^n_{i=1}$, dove $z_i = (x_i, y_i)$ è l'i-esimo esempio disponibile, cercando sotto insiemi di vettori x_i, cioè cluster delle proiezioni degli esempi z_i sullo spazio di ingresso. Il criterio di similitudine adottato è quindi basato sulla distanza tra gli esempi, misurata nello spazio $\Re^d$ di ingresso.

Nel caso si desideri utilizzare la partizione fornita dalla *cluster analysis* per sviluppare un modello fuzzy costituito da un insieme di approssimatori lineari, cioè un modello che approssimi la funzione in esame attraverso una

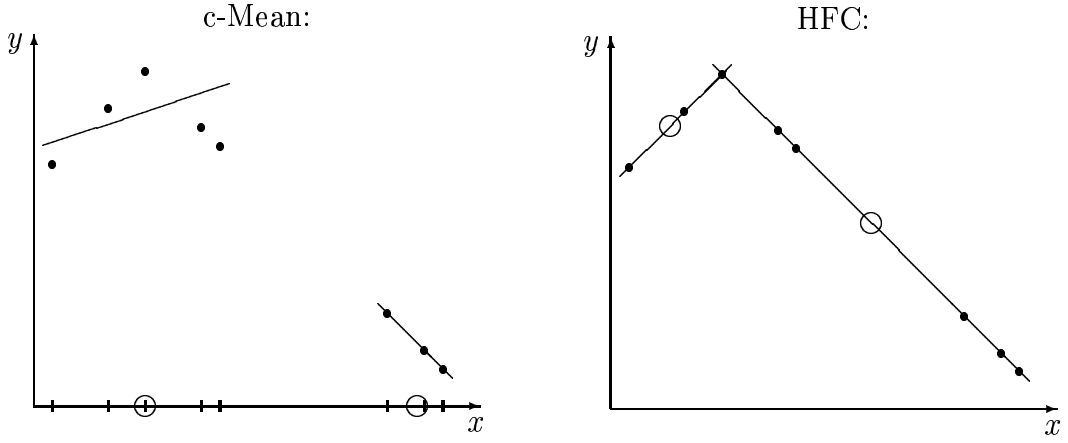


Figura 4.5: Cluster Analysis: c-mean Vs. hyperplane fuzzy clustering. Il metodo c-mean cerca cluster delle proiezioni degli esempi sullo spazio di ingresso, mentre il metodo hyperplane fuzzy clustering cerca cluster di punti che, nello spazio congiunto di ingresso-uscita, giacciano sullo stesso iperpiano.

ipersuperficie ottenuta come composizione *smooth* di porzioni di iperpiani ciascuna localizzata in una determinata regione dello spazio di ingresso, il metodo di clustering adottato dovrebbe essere in grado di individuare regioni che contengano esempi che giacciano effettivamente sullo stesso iperpiano, e che quindi possano essere *spiegati* dallo stesso approssimatore locale lineare. L'algoritmo di clustering dovrebbe cioè cercare raggruppamenti di esempi che siano vicini per quanto riguarda le loro proiezioni sullo spazio di ingresso, e che contemporaneamente, nello spazio congiunto ingresso-uscita, giacciano sullo stesso iperpiano.

Fatte queste osservazioni, proponiamo il metodo hyper-plane fuzzy clustering (HFC) che può essere convenientemente utilizzato per cercare raggruppamenti di esempi che soddisfino i requisiti sopra illustrati e che quindi è particolarmente adatto per inizializzare modelli neuro fuzzy costituiti componendo approssimatori lineari. Il metodo HFC rappresenta ciascun cluster attraverso l'iperpiano su cui si dispongono i punti appartenenti al cluster stesso, e attraverso un prototipo o centro di massa. I raggruppamenti di esempi nello spazio congiunto ingresso-uscita sono individuati minimizzando la

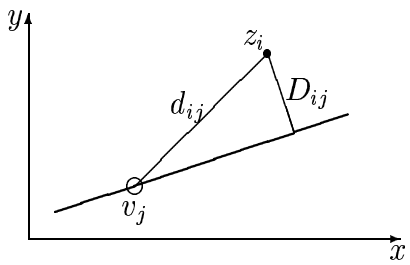


Figura 4.6: Le distanze considerate dall'algoritmo HFC.

seguente cifra di merito globale:

$$V(v,U) = \sum_{i=1}^{n}\sum_{j=1}^{c}(\mu_{ij})^p(1-\alpha)d_{ij}^2 + \alpha D_{ij}^2, \tag{4.15}$$

dove $d_{ij} = \|z_i - v_j\|$ è la distanza dell'i-esimo esempio dal prototipo del j-esimo cluster, D_{ij} è la distanza ortogonale dell'i-esimo esempio dall'iperpiano associato al j-esimo cluster (figura 4.6), il coefficiente p ha lo stesso significato visto (§ 4.1.3), e α determina il peso relativo dei due contributi. L'algoritmo che abbiamo implementato è illustrato in figura 4.7.

Nel capitolo 6 mostreremo, attraverso alcuni risultati sperimentali, il miglioramento di prestazioni consentite dall'algoritmo *Hyperplane Fuzzy Clustering*, rispetto a quelle ottenibili adottando il metodo c-mean.

L'algoritmo Fuzzy c-elliptotypes:

1. $\tau = 0$. Inizializzazione della matrice U ad un valore $U^{(0)} \in M_f$.

2. Calcolo dei prototipi:

$$v_j^{(\tau+1)} = \frac{\sum_{i=1}^{n} \left(\mu_{ij}^{(\tau)}\right)^p z_i}{\sum_{i=1}^{n} \left(\mu_{ij}^{(\tau)}\right)^p}$$

3. Calcolo delle matrici di scatter fuzzy $S_j^{(\tau+1)}$:

$$S_j^{(\tau+1)} = \sum_{i=1}^{n} \left(\mu_{ij}^{(\tau)}\right)^p \left(z_i - v_j^{(\tau+1)}\right)\left(z_i - v_j^{(\tau+1)}\right)^T$$

4. Estrazione dei d autovettori principali di $S_j^{(\tau+1)}$, cioè degli autovettori $\Lambda_{jr}^{(\tau+1)}$, $r = 1, 2, \ldots, d$, corrispondenti ai d autovalori $\lambda_{jr}^{(\tau+1)}$ puó grandi. Tali autovettori individuano nello spazio congiunto $\Re^{d+1}$ l'iperpiano H.

5. Dalcolo delle distanze $d_{ij}^{(\tau+1)}$ tra ogni punto z_i da ogni prototipo $v_j^{(\tau+1)}$, e delle distanze ortogonali $D_{ij}^{(\tau+1)}$ tra ogni z_i e ogni iperpiano $H_j^{(\tau+1)}$

6. Aggiornamento della matrice U:

$$\mu_{ij}^{(\tau+1)} = \left[\sum_{k=1}^{c} \left(\frac{\gamma_{ij}}{\gamma_{ik}}\right)^{\frac{2}{p-1}}\right]^{-1}$$

dove

$$\gamma_{ij} = (1-\alpha) d_{ij}^2 + \alpha D_{ij}^2.$$

Se risulta $\gamma_{is} = 0$, con $s \in S \subset \{1, 2, \ldots, c\}$, i valori μ_{is} sono posti casualmente, rispettando i vincoli $\mu_{is} \in [1, 0]$ e $\sum_{s \in S} \mu_{is} = 1$.

7. Se $V\left(v^{(\tau+1)}, U^{(\tau+1)}\right) < V\left(v^{(\tau)}, U^{(\tau)}\right)$, si torna al passo 2 con $\tau = \tau + 1$.

Figura 4.7: L'algoritmo *Hyper-Plane Fuzzy Clustering* cerca raggruppamenti di dati adottando un criterio di similitudine basato sulla distanza da un prototipo e sulla distanza ortogonale da un iperpiano.

4.2 Identificazione strutturale

La fase di identificazione strutturale consiste nella scelta della corretta complessità della funzione parametrica $f(x, \alpha)$ da utilizzare per costruire un modello del fenomeno che ha generato l'insieme Z degli esempi disponibili (§ 2.4). Per un modello neuro-fuzzy, avendo fissato la famiglia parametrica degli approssimatori locali e quella delle funzioni di appartenenza, il parametro che si è scelto di adottare come indice della complessità, quello rispetto al quale si vuole determinare la struttura ottima, è il numero m di approssimatori locali di cui è costituito il modello stesso. Per evidenziare la dipendenza del modello dal numero m, indicheremo con la notazione $f^m(x, \alpha)$ il modello fuzzy composto da m approssimatori locali.

Nell'approccio fuzzy linguistico e in gran parte della letteratura neuro-fuzzy la scelta del valore di m è lasciata all'analista e viene effettuata considerando conoscenza a priori sul fenomeno in esame. Nell'ottica di automatizzare il più possibile il processo di apprendimento, abbiamo applicato un metodo per estrarre dai dati Z l'informazione necessaria per determinare il valore di m che consenta i migliori risultati in generalizzazione.

Il metodo che proponiamo, e che abbiamo implementato, opera un'esplorazione esaustiva di un intervallo specificato dall'analista dello spazio del parametro m cercando il valore $\hat{m}$ di tale parametro che minimizza, come una stima della capacità di generalizzazione, l'errore in cross-validation (§ 2.3.2) commesso dal modello $f^m(x, \alpha)$.

Per ogni valore di m da m_{min} a m_{max}, viene valutato l'errore di cross-validation commesso dal modello di complessità m. Tale valore viene ottenuto dividendo in maniera casuale l'insieme Z in 10 sotto insiemi, e utilizzando ciascuno di questi per validare il modello $f^m(x, \alpha)$ identificato utilizzando gli altri 9 sottoinsiemi.

Il risultato di tale processo è un grafico del tipo di quello riportato in figura 4.8 dal quale è possibile estrarre informazioni sul numero ottimo di regole da adottare. In ascissa sono riportati i valori m di complessità esplorati mentre in ordinata compare una stima dell'errore in generalizzazione e un relativo intervallo di confidenza. L'andamento tipico mostrato da questi grafici è caratterizzato da un'iniziale diminuzione della stima dell'errore in generalizzazione, alla quale segue una lieve tendenza all'aumento. Spesso al crescere di m l'intervallo di confidenza tende ad allargarsi.

Ottenute queste informazioni sull'andamento dell'errore in generalizzazione rispetto al numero di approssimatori locali, possono essere utilizzati metodi di *hypotesis testing* o metodi *guidati dall'utente* per selezionare il valore di m che promette migliori prestazioni in generalizzazione.

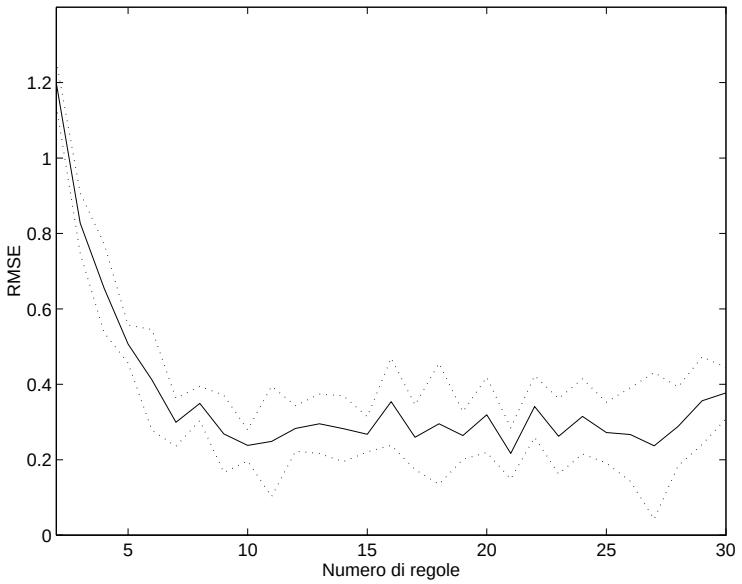


Figura 4.8: Errore in cross validation contro numero delle regole m. Il tratto continuo rappresenta, in questo grafico, il valor medio sugli insiemi di cross validation, della radice quadrata dell'errore quadratico medio (RMSE). Le linee punteggiate individuano un intervallo di confidenza e sono tracciate a una distanza dalla linea continua pari alla varianza campionaria dei valori del RMSE relativi ai diversi insiemi di validazione.

Capitolo 5

Lazy Learning adattativo

L'approccio Lazy Learning ridefinisce l'obiettivo dell'apprendimento: il problema della stima di una funzione viene decomposto dai metodi *memory-based* in una serie di problemi disgiunti, consistenti ciascuno nella stima del *valore* assunto in uno specifico punto dalla funzione stessa. Ciascuno dei sotto problemi definiti viene affrontato identificando un modello locale *ad-hoc* che opera senza interazione con il resto della struttura. Questo approccio, come sarà evidenziato dai risultati sperimentali, consente di trattare megli quelle situazioni in cui è disponibile una quantità limitata di esempi e soprattutto quando gli esempi dati non sono disposti uniformemente sullo spazio di ingresso.

Risulta naturale desiderare di rendere locale anche la fase di identificazione strutturale, e quindi definire la struttura ottima di ciascuno dei modelli locali indipendentemente da quella utilizzata per gli altri. Per ogni valore x_* dell'ingresso, è possibile estrarre dagli esempi disponibili sia informazioni relative alla struttura ottima del modello locale, sia informazioni relative ai parametri di tale modello. Il criterio secondo il quale viene valutata la capacità di generalizzazione di un modello locale, criterio sul quale si basa l'identificazione strutturale nel contesto del lazy learning, è una versione locale del metodo di validazione *leave-one-out* (§ 2.3.3).

In una prima implementazione dell'approccio lazy learning, abbiamo reso adattativa la selezione del grado del polinomio approssimante (§ 3.3.2) e la larghezza di banda (§ 3.3.2). Come vederemo in seguito, è possibile estendere l'uso della validazione locale anche ad altre fasi del processo di apprendimento ed impiegarla in fase di *feature selection* o abbinata alla *principal component analysis* (§ 2.1.3). Il criterio della validazione locale non è quindi da considerarsi solo ed esclusivamente in fase di identificazione strutturale ma, nell'approccio Memory-Based, è un valido strumento di supporto che l'analista può impiegare in varie fasi del processo di apprendimento.

Nel seguito di questo capitolo presenteremo il metodo lazy che abbiamo implementato. Mostreremo la struttura degli approssimatori locali, i parametri

che li descrivono e l'implementazione del metodo di validazione locale che funge da guida alla selezione della stuttura ottima.

5.1 Ordine del polinomio approssimante

Comunemente, la scelta della famiglia parametrica di funzioni da usare come approssimatori locali cade sulla famiglia dei polinomi di grado p (§ 3.3.2). In letteratura è frequente l'uso di costanti e di polinomi lineari, meno frequente quello di polinomi quadratici o cubici. Il passaggio dall'utilizzo di un polinomio ad esempio di primo grado ad uno di secondo, in molte applicazioni porta ai dei cambiamenti sostanziali nei risultati. Una soluzione di compromesso tra polinomi di gradi contigui può essere ottenuta considerando come famiglia parametrica quella dei *polynomial mixing*.

Il *polynomial mixing* è una combinazione lineare di polinomi, applicato in origine da Mallows per affrontare problemi di regressione parametrica globale (Mallows, 1974). La versione locale, quella da noi adottata, è stata proposta per la prima volta da Cleveland, Hastie e Loader (Cleveland *et al.*, 1995).

Un *polynomial mixing* è un'estensione del polinomio algebrico classico in cui il grado non è più vincolato ad essere un numero intero ma può assumere un qualsiasi valore non negativo p. Se p è un intero si riottiene un polinomio classico, nel caso generico risulta invece $p = b + c$, dove b è un intero e c è tale per cui $0 < c < 1$. Dato un insieme di punti $Z = \{(x_i, y_i)\}_{i=1}^{n}$, la stima della funzione di regressione di y su x fornita dal *polynomial mixing* di grado p è la combinazione lineare del polinomio approssimante di grado b e di quello di grado $b+1$, presi con i pesi $1-c$ e c rispettivamente. Formalmente risulta:

$$f^p(x, \hat{\alpha}_p) = (1-c)P_b(x, \hat{\alpha}_b) + c\, P_{b+1}(x, \hat{\alpha}_{b+1}),$$

dove $P_b(x, \hat{\alpha}_b)$ e $P_{b+1}(x, \hat{\alpha}_{b+1})$ sono i migliori stimatori polinomiali, rispettivamente di grado b e $b+1$, della funzione di regressione.

Nella nostra implementazione, la scelta del grado del *polynomial mixing* da adottare è stata resa adattativa. L'algoritmo di apprendimento sfrutta il criterio della minimizzazione dell'errore locale in validazione per selezionare automaticamente il valore ottimo p_* per il dato valore dell'ingresso x_* relativamente al quale è richiesta la stima dell'uscita.

5.2 Dimensione del neighborhood

Un altro parametro strutturale rispetto al quale il metodo è stato reso adattativo è la larghezza di banda, cioè la dimensione della regione, centrata nel punto x_*, all'interno della quale la funzione non nota viene approssimata da un modello locale.

Come indice della dimensione di tale regione si è considerato il numero k di esempi appartenenti a Z che vengono a trovarsi all'interno della regione stessa. In letteratura si usa riferirsi alla misura della larghezza di banda attraverso il numero k di *vicini* con il nome di *Nearest Neighbor bandwidth selection* (Cleveland & Loader, 1996). La ragione di tale scelta è semplice: una larghezza di banda costante provoca grosse oscillazioni nella varianza della stima a causa della diversa distribuzione delle osservazioni nello spazio di ingresso. Adottando una larghezza di banda costante può accadere, al limite, che nessuna osservazione cada all'interno della regione considerata e che quindi la stima sia impossibile. La *Nearest Neighbor bandwidth selection* garantisce che effettuando la stima di regressione siano sempre considerati un numero costante k di esempi.

Tipicamente, la scelta del valore k è effettuata in maniera euristica e il valore scelto viene mantenuto costante per ogni valore x_*. L'approccio adattivo prevede una scelta locale del valore k_* che deve essere adottato per stimare il valore che la funzione di regressione assume in un determinato punto x_*. Si noti l'importanza di abbinare la selezione dei valori ottimi locali k_* e p_*. Tali quantità sono intimamente collegate in quanto insieme giocano il ruolo di determinare le componenti *bias/variance* dell'errore di stima (§ 3.3.2). Intuitivamente ciò può essere spiegato osservando che un valore alto di p_*, quindi l'utilizzo di un polinomio di grado elevato descritto da più parametri, richiede necessariamente che si consideri un neighborhood vasto e quindi un valore elevato di k_*.

5.3 Feature selection

Un altra fase del processo di apprendimento alla quale è possibile estendere l'uso della validazione locale è quella della trasformazione dei dati (§ 2.1.3). Il sistema implementato consente, per ogni valore dell'ingresso x_*, di valutare se una determinata componente dello spazio di ingresso possa o meno contribuire alla stima del valore y_* assunto dall'uscita. Anche tale valutazione è effettuata utilizzando come criterio la stima dell'errore fornita dal metodo *leave-one-out* (Bersini *et al.*, 1997a).

5.4 Identificazione parametrica e strutturale

In questo paragrafo illustreremo l'algoritmo di identificazione utilizzato per determinare il valore ottimo dei parametri del modello utilizzato come approssimatore locale della funzione di regressione. Anche nel caso del memory-based learning è possibile distinguere, all'interno del processo di apprendimento, due fasi separate: una di identificazione parametrica e una di identificazione strut-

L'algoritmo.

Per un dato valore x_* dell'ingresso:

for k ... Ciclo sulla dimensione del neighborhood

for p ... Ciclo sul grado del *polynomial mixing*

1. Identificazione dell'approssimatore locale $f_*(x, \alpha)$ di grado p centrato in x_*, utilizzando i k esempi appartenenti a Z più vicini a x_*
2. Valutazione del modello locale $f_*(x, \alpha)$ nel punto x_*
3. Validazione del modello locale $f_*(x, \alpha)$ attraverso la *PRESS statistic*

end for

end for

La stima del valore y_* restituita dal metodo *lazy learning* è quella fornita dal modello che presenta il minor valore dell'errore quadratico medio valutato attraverso la *PRESS statistic*.

Figura 5.1: L'algoritmo *Lazy Learning Adattativo* che è stato implementato.

turale. Coerentemente con la filosofia lazy, e contrariamente a ciò che avviene nel caso dei metodi model-based, le due fasi sono ripetute ogni volta che deve essere stimata l'uscita y_* relativa ad un certo valore dell'ingresso.

Per un dato valore x_*, la struttura ottima[1] m_* è individuata attraverso un processo *trials and errors* (§ 2.4). Lo spazio del parametro strutturale viene esplorato, e per ogni valore di m, cioè per ogni coppia (p, k), grado dell'approssimatore locale e numero di vicini, viene identificato un modello locale la cui capacità di generalizzazione viene stimata attraverso il metodo *leave-one-out*[2]. Come stima del valore y_* viene presa quella ottenuta valutando in x_* il modello locale che ha dimostrato la migliore capacità di generalizzazione. Uno schema dell'algoritmo adottato è riportato in figura 5.1.

[1]Per semplicità di trattazione, facciamo qui riferimento al caso in cui il sistema sia adattativo per quanto riguarda il grado del *polynomial mixing* e del numero di vicini considerati. L'estensione al caso più generale in cui il metodo di validazione è utilizzato in fase di *feature selection*, non presenta difficoltà concettuali ulteriori.

[2]In realtà viene utilizzata una versione locale della *PRESS statistics* (§ 5.5), funzionalmente equivalente al *leave-one-out* ma meno costosa in termini di tempo di computazione.

5.4.1 I parametri del modello locale

Sfuttando la linearità dei modelli utilizzati come approssimatori locali, dati i valori correnti di p e di k, l'identificazione parametrica può essere effettuata attraverso il metodo dei minimi quadrati (§ 2.2.1).

Modelli lineari globali

Consideriamo in principio il problema dell'identificazione dei parametri di un modello lineare globale. L'identificazione di un modello locale sarà descritta modificando l'algoritmo globale. Utilizzando la notazione già introdotta (§ 3.3.2), un polinomio di grado b può essere scritto nella forma seguente[3]:

$$f(x, \alpha) = \sum_{j=1}^{q} \alpha_j \phi_j(x), \tag{5.1}$$

dove α_j è la j-esima componente del vettore α, $q = (d+b)!\,/\,(d!\,b!)$ è il numero di termini di cui è composto un polinomio di grado b se il vettore x ha d componenti, e $\phi_j(x)$ è un monomio di grado minore o uguale a b, ottenuto dalla seguente:

$$\phi_j(x) = \prod_{l=1}^{d} \left(x^{(l)}\right)^{\gamma_{jl}},$$

dove $x^{(l)}$ è la l-esima componente del vettore $x \in \Re^d$, e le quantità γ_{jl} sono interi tali che $\gamma_{jl} \geq 0$ e $\sum_{l=1}^{d} \gamma_{jl} \leq b$. In forma vettoriale la (5.1) può essere convenientemente riscritta come

$$f(x, \alpha) = \phi^T \alpha, \tag{5.2}$$

dove si sia posto $\phi(x) = [\phi_1(x), \phi_2(x), \ldots, \phi_q(x)]^T$. Si noti che con $q = d+1$, ponendo $\phi_l(x) = x^{(l)}$ e $\phi_{d+1}(x) = 1$, la (5.2) è un polinomio di primo grado, cioè un'iperpiano nello spazio ingresso-uscita. Per $q = 1$, con $\phi_1(x) = 1$ si ottiene il polinomio costante. Si indichi inoltre con Y il vettore $[y_1, y_2, \ldots, y_n]^T$, e con Φ la matrice la cui i-esima riga è il vettore $\phi(x_i)^T$ dove x_i è il valore dell'ingresso dell'i-esimo esempio in Z. L'intero insieme degli esempi può quindi essere descritto dalla seguente equazione matriciale:

$$Y = \Phi\alpha. \tag{5.3}$$

[3]Si noti che l'identificazione di un *polynomial mixing* di grado $p = b + c$ si riduce all'identificazione dei parametri di un approssimatore polinomiale di grado b e di uno di grado $b+1$ (§ 5.1)

Utilizzando la notazione così definita il valore ottimo dei parametri α del modello della funzione di regressione globale si ottiene risolvendo in α le equazioni normali (§ 2.2.1):

$$\left(\Phi^T\Phi\right)\alpha = \Phi^T Y. \tag{5.4}$$

Da cui si ottiene:

$$\alpha = \left(\Phi^T\Phi\right)^{-1}\Phi^T Y.$$

In realtà l'inversione della $\Phi^T\Phi$ non è il modo migliore dal punto di vista numerico per risolvere le equazioni normali. Esistono in letteratura altre tecniche numeriche più efficienti ed accurate quali, ad esempio, il metodo di *pseudo inversione* (Press *et al.*, 1995). Come descritto in seguito, il metodo da noi adottato è il metodo di *ridge regression* (§ 5.4.1) che consente di risolvere quelle situazioni in cui i dati non coprono alcune direzioni ello spazio di ingresso.

Modelli lineari locali

Nell approccio lazy learning l'obiettivo per il quale viene stimata la funzione di regressione è quello di fornire una stima del valore y_* assunto in x_* dalla funzione in esame. Dato questo obiettivo ha senso che la stima della funzione di regressione sia fatta in funzione del punto x_* e che il modello lineare identificato sia *ritagliato* specificatamente per affrontare il problema considerato.

Un modello lineare $f_*(x,\alpha)$ specifico per essere valutato nel punto x_* può essere ottenuto considerando solo le k osservazioni più vicine[4] a x_* nello spazio di ingresso, e dando a queste un *peso* differente tale per cui quelle più vicine influenzino maggiormente di quelle lontane il valore del parametro α identificato (§ 3.3.2). Formalmente ciò significa considerare l'insieme $Z_* = \{(x_i, y_i)\}_{i=1}^k$ dei k esempi appartenenti a Z tali per cui $\|x_i - x_*\| < \|x_j - x_*\|$, $\forall i,j$: $x_i \in Z_*$, $x_j \notin Z_*$; e associare a ciascuno di questi un peso

$$w_i = \sqrt{K\left(\|x_* - x_i\|\right)}\,.$$

Tipicamente la funzione $K\left(\|x_* - x_i\|\right)$ è una funzione a *campana*[5]. Introducendo la matrice diagonale W tale per cui $W_{ii} = w_i$, è possibile pesare le varie

[4] Si considera come misura di distanza la norma Euclidea.

[5] Nella nostra implementazione sono state adottate funzioni *tricubiche* o gaussiane. La funzione tricubica è definita come (Cleveland, 1979):

$$t(u) = \begin{cases} \left(1 - |u|^3\right), & |u| \le 1; \\ 0, & |u| > 1. \end{cases}$$

osservazioni nel modo seguente:

$$\Psi = W\Phi,$$
$$\Upsilon = WY.$$

Dove la matrice Ψ è tale per cui la i-esima riga è uguale al vettore ψ_i^T, con $\psi_i = w_i\phi_i$; l'i-esimo posto del vettore Υ è invece occupato da $\upsilon_i = w_i y_i$. I parametri del modello lineare locale e pesato si ottengono risolvendo in α, le nuove equazioni normali:

$$\left(\Psi^T\Psi\right)\alpha = \Psi^T\Upsilon,$$

ottenendo:

$$\alpha = \left(\Psi^T\Psi\right)^{-1}\Psi^T\Upsilon.$$

In ultima analisi quindi, fissato il grado del polinomio approssimante e il numero di vicini considerati, la stima del valore y_* fornita dal metodo lazy learning è data da:

$$\hat{y}_* = \phi(x_*)\left(\Psi^T\Psi\right)^{-1}\Psi^T\Upsilon. \tag{5.5}$$

Ridge regression

Un problema potenziale che potrebbe verificarsi è che i dati potrebbero essere distribuiti nello spazio di ingresso in maniera tale che la matrice $\Psi^T\Psi$ presente nella (5.5), sia quasi singolare. Tale situazione si verifica se non esistono, in un intorno di $\phi(x_*)$, dei punti con peso diverso da zero che coprono tutte le direzioni dello spazio dei vettori ϕ. In tal caso il rango di $\Psi^T\Psi$ non è massimo e quindi non sono date sufficienti equazioni indipendenti per determinare tutte le componenti del vettore dei parametri α. Una soluzione a questo problema è fornita dalla *ridge regression* (Myers, 1994) che consiste nel risolvere, al posto delle equazioni normali, la seguente equazione matriciale:

$$\left(\Psi^T\Psi + \Lambda\right)\alpha = \Psi^T\Upsilon + \Lambda\bar{\alpha}, \tag{5.6}$$

dove Λ è una matrice diagonale i cui elementi λ_i^2 sono piccoli e positivi. Il vettore $\bar{\alpha}$ è invece una stima *a priori* del valore di α. Tipicamente $\bar{\alpha}$ è un vettore i cui elementi sono tutti nulli.

Identificare il valore del parametro α risolvendo la (5.6) equivale ad aggiungere agli esempi realmente disponibili un numero di *osservazioni fittizie* che, in assenza di dati che coprano tutte le direzioni dello spazio di ϕ, introducono un *bias* sul valore di α *forzando* la soluzione identificata verso il vettore $\bar{\alpha}$.

5.5 Validazione locale

Il cuore del metodo lazy learning adattativo che abbiamo implementato è costituito dal metodo di validazione. Per una data struttura descritta dal parametro m, è possibile ottenere una stima della capacità di predizione di y_* da parte del modello locale che possiede tale struttura, valutando l'errore commesso nella predizione di ciascun valore y_i, con $z_i \in Z_*$, dal modello $f_*^m(x, \alpha)$ addestrato con tutti gli esempi Z_* escluso l'i-esimo. Questo è il principio del *leave-one-out* adattato al caso locale (Atkeson *et al.*, 1996). Il lazy learning è tale per cui il metodo di validazione *leave-one-out* può essere adottato, con un costo computazionale accettabile, impiegando una versione locale della *PRESS statistic* (§ 2.3.3).

Risulta comodo definire preventivamente alcune grandezze che verranno utilizzate nella dimostrazione:

ϵ_{i,x_i}^{cv} E' l'errore commesso nella predizione del valore y_i dalla stima locale della funzione di regressione centrata in x_i. Tale stima è ottenuta identificando un modello locale attraverso gli esempi (x_j, y_j) con $j \neq i$, pesati ciascuno in funzione della distanza da x_i.

ϵ_{i,x_*}^{cv} E' l'errore commesso nella predizione del valore y_i dalla stima locale della funzione di regressione centrata in x_*. In questo caso cioè il modello locale è identificato pesando gli esempi ciascuno in funzione della distanza da x_* (figura 5.2).

r_{i,x_*}^{cv} E' l'errore commesso nella predizione del valore y_i dalla stima locale della funzione di regressione centrata in x_*, pesato in funzione della distanza di x_i da x_*. Risulta cioè:

$$r_{i,x_*}^{\mathrm{CV}} = \epsilon_{i,x_*}^{\mathrm{CV}} \sqrt{K\big(\|x_i - x_*\|\big)}.$$

Si noti che r_{i,x_*}^{CV} è sempre un *residuo* di cross-validation, ottenuto rimuovendo l'esempio i-esimo esempio dal training set.

r_{i,x_*} E' l'errore commesso nella predizione del valore y_i dal modello centrato in x_* e identificato usando tutti gli esempi, pesato sempre in funzione della distanza di x_i da x_*.In sostanza r_{i,x_*} è la versione pesata dell'errore di risostituzione (§ 2.3).

Definite queste grandezze è possibile ottenere una versione locale di un metodo di validazione, seguendo la stessa strada che si è seguita per ottenere un metodo di identificazione locale a partire da uno globale.

L'obiettivo è quello di ottenere una stima della capacità di predizione di y_*, non noto, da parte del modello centrato in x_*. Modello ottenuto come stima

pesata della funzione di regressione e identificato usando gli esempi (x_i, y_i) pesati in funzione della distanza $\|x_i - x_*\|$.

Come stima di tale grandezza si consideri la media pesata degli errori quadratici di *leave-one-out* commessi dai modelli centrati nei vari punti x_i. Si considerì cioè la quantità:

$$\mathrm{MSE}_*^{\mathrm{cv}} = \frac{\sum_i (\epsilon_{i,x_i}^{\mathrm{cv}})^2 K\big(\|x_i - x_*\|\big)}{\sum K\big(\|x_i - x_*\|\big)}. \tag{5.7}$$

Si approssimi ora l'errore commesso nella predizione di y_i dal modello centrato in x_i con l'errore commesso dal modello centrato in x_*. Formalmente ciò significa porre $\epsilon_{i,x_i}^{\mathrm{cv}} \approx \epsilon_{i,x_*}^{\mathrm{cv}}$. La (5.7) può essere riscritta come:

$$\mathrm{MSE}_*^{\mathrm{cv}} = \frac{\sum_i (\epsilon_{i,x_*}^{\mathrm{cv}})^2 K\big(\|x_i - x_*\|\big)}{\sum_i K\big(\|x_i - x_*\|\big)} = \frac{\sum_i (r_{i,x_*}^{\mathrm{cv}})^2}{k_{\mathrm{eff}}}, \tag{5.8}$$

dove $k_{\mathrm{eff}} = \sum_i K\big(\|x_i - x_*\|\big)$ rappresenta il numero *efficace* di esempi considerati, e dove r_{i,x_*}^{cv} è il residuo pesato di cross-validation. Il valore del residuo pesato di cross-validation è legato al residuo pesato di sostituzione dalla seguente:

$$r_{i,x_*}^{\mathrm{cv}} = \frac{r_{i,x_*}}{1 - \psi_i^T (\Psi^T \Psi + \Lambda)^{-1} \psi}.$$

Finalmente, l'equazione dell'errore quadratico medio di cross-validation può essere scritto come:

$$\mathrm{MSE}_*^{\mathrm{cv}} = \frac{1}{k_{\mathrm{eff}}} \sum_i \left(\frac{r_{i,x_*}}{1 - \psi_i^T (\Psi^T \Psi + \Lambda)^{-1} \psi_i} \right)^2, \tag{5.9}$$

La (5.9) è una versione locale della PRESS statistic e consente di eseguire una validazione *leave-one-out* senza la necessità di dover identificare ogni volta il modello per ogni punto escluso. I valori r_{i,x_*} si ottengono infatti semplicemente valutando nei vari punti x_i il modello identificato per stimare y_* e quindi centrato in x_*. Questo procedimento richiede quindi che la matrice $\Psi^T \Psi + \Lambda$ venga invertita solo una volta.

Il metodo *lazy learning* è dunque un metodo che consente di fornire una stima del valore assunto da una funzione in un determinato punto, senza identificare un modello globale. Le caratteristiche dell'approssimatore utilizzato per fornire tale stima possono essere determinate localmente e quindi adattate punto per punto alle caratteristiche della funzione in esame. I risultati sperimentali che presenteremo mostreranno che questo approccio è in grado di

fornire buoni risultati qualora i dati disponibili siano scarsi. Per concludere questo capitolo, l'aspetto che merita essere discuso è quello che caratterizza il metodo: la *lazyness*. Ogni operazione necessaria alla stima del valore assunto in un punto viene ritardata fintanto che una stima non viene esplicitamente richiesta. Ciò comporta svantaggi e vantaggi. Se la base di esempi non varia nel tempo, un metodo *model-based* richiede per ogni singola stima un tempo di calcolo inferiore e richiede che siano memorizzati solo i parametri del modello e non tutti gli esempi come nel caso del metodo *lazy*. Per contro, un metodo *memory-based* ripete ogni volta l'intero processo di stima: questo consente di modificare di volta in volta la base di esempi considerata e di aumentarne la dimensione qualora di rendesse disponibile nuova informazione. La conclusione che si deve trarre da queste osservazioni e che la scelta dell'architettura da adottare è sempre dettata dalle caratteristiche del problema in esame e che non è possibile indicare un'approccio che sia assolutamente superiore agli altri.

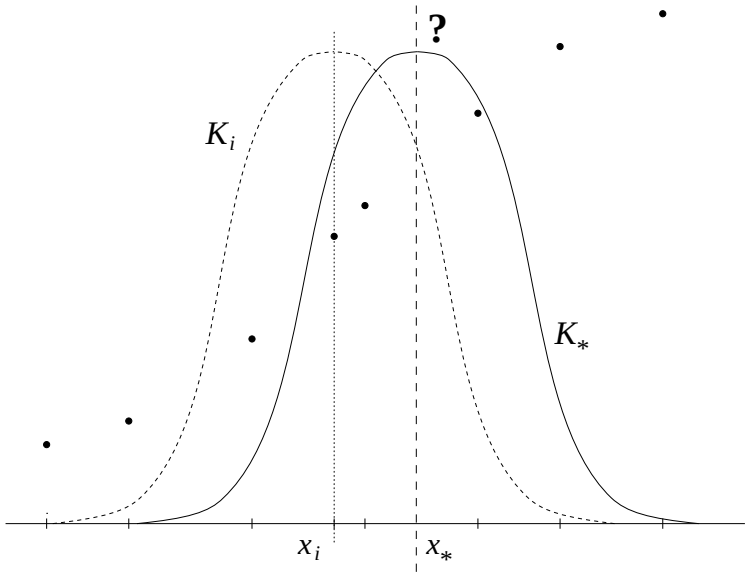


Figura 5.2: *PRESS statistic locale.* L'errore commesso nella predizione di y_i dal modello centrato nel punto x_i, cioè quello identificato pesando gli esempi disponibili attraverso la funzione K_i, viene approssimato dall'errore commesso dal modello identificato utilizzando come funzione peso la funzione K_*, cioè dal modello identificato per stimare il valore non noto y_*. Formalmente ciò significa porre $\epsilon^{\text{cv}}_{i,x_i} \approx \epsilon^{\text{cv}}_{i,x_*}$. Questa approssimazione consente di sviluppare un metodo di validazione *leave-one-out* locale che, sfruttando la linearità nei parametri della famiglia di approsimatori locali adottati, richiede l'identificazione di un unico modello, e non di k se k sono gli esempi coinvolti nel processo di validazione stesso.

Capitolo 6

Esperimenti

6.1 Ricostruzione di una superficie

Il primo degli esperimenti che presentiamo riguarda il problema della ricostruzione di una superficie nota solo attraverso un limitato numero di esempi affetti da rumore. Ciascun esempio $z_i = (x_i, y_i)$ è stato ottenuto campionando in maniera uniformemente casuale il dominio $[-1, 1] \times [-1, 1]$ della funzione

$$y = 4 \sin\left(\pi\, x_i^{(1)}\right) + 2 \cos\left(\pi\, x_i^{(2)}\right) + N(0, 0.5), \tag{6.1}$$

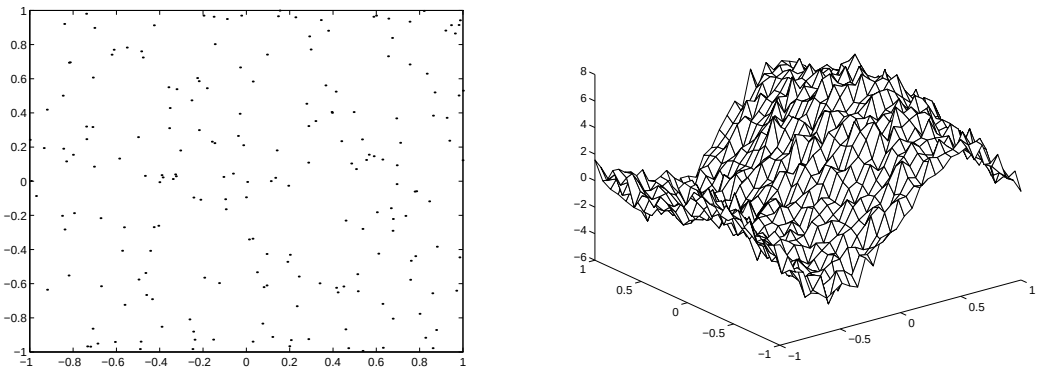
dove con $N(0, 0.5)$ si è indicato un rumore bianco a media nulla e deviazione standard pari a 0.5. L'informazione disponibile per la ricostruzione della superficie è costituita da un insieme $Z = \{z_i\}_{i=1}^{200}$ composto di 200 esempi (figura 6.1).

L'indice che è stato adottato per valutare la capacità di un generico metodo di apprendimento di fornire una soluzione al problema sopra definito, è la radice del valor medio campionario del quadrato dell'errore (RMSE) commesso su un insieme, indipendente da Z, composto da 1024 punti campionati regolarmente sul dominio di ingresso, e non reso disponibile durante fasi di identificazione parametrica e strutturale.

6.1.1 I metodi di apprendimento confrontati

Il *benchmark* così definito è stato utilizzato per confrontare, nelle medesime condizioni sperimentali, alcuni metodi appartenenti alla famiglia *neuro-fuzzy*, a quella *memory-based*, e a quella *mixture of experts*. I sistemi considerati sono i seguenti.

1. Un sistema fuzzy composto da modelli locali costituiti da funzioni lineari. I contributi dei vari approssimatori locali sono combinati attraverso funzioni di appartenenza triangolari normalizzate. Le posizione e le



(a) (b)

Figura 6.1: I 200 dati utilizzati come *training set* sono stati ottenuti campionando in maniera uniformemente casuale (figura a) la funzione $y = 4\sin\left(\pi\, x_i^{(1)}\right) + 2\cos\left(\pi\, x_i^{(2)}\right) + N(0, 0.5)$ nel dominio $[-1, 1] \times [-1, 1]$. Per dare un'idea della potenza del rumore, riportiamo 1024 campioni estratti dalla stessa funzione e disposti su una griglia regolare (figura b).

basi delle funzioni di appartenenza sono state inizializzate attraverso il metodo di clustering c-mean.

2. Un sistema fuzzy caratterizzato da funzioni di appartenenza gaussiane non normalizzate e da modelli locali lineari. Il metodo di inizializzazione adottato è il c-mean.

3. Un sistema fuzzy con funzioni di appartenenza gaussiane non normalizzate, modelli locali lineari. I centri e le basi delle membership function sono inizializzati con il metodo *hyperplane fuzzy clustering* sfruttando informazioni dallo spazio congiunto ingresso-uscita.

4. Un sistema fuzzy in cui gli approssimatori locali sono delle costanti, e le funzioni di appartenenza sono gaussiane non normalizzate. Si noti che in questa configurazione un sistema fuzzy è equivalente a una *radial basis function network*. I parametri delle funzioni di appartenenza sono inizializzati attraverso il metodo c-mean.

5. Un sistema mixture of experts con un solo *gating network* che funge da arbitro tra più esperti posti tutti allo stesso livello gerarchico.

6. Un sistema Hierarchical mixture of experts organizzato come un albero binario.

7. Un metodo lazy learning adattativo basato sulla regressione locale, pesata da una funzione tricubica. La famiglia parametrica a cui appartengono gli approssimatori locali è quella dei *polynomial mixing*.

Per tutti gli approcci, ad esclusione di quello lazy, il numero ottimale di modelli locali da impiegare è stato individuato sulla base di un processo di validazione effettuato attraverso il metodo *cross-validation*.

Abbiamo diviso casualmente l'insieme Z degli esempi disponibili in 10 sottoinsiemi[1] Z_t, in maniera tale che ognuno di questi contenesse campioni della funzione in esame che fossero uniformemente distribuiti sul dominio considerato.

Per ogni grado m di complessità da m_{min} a m_{max}, abbiamo valutato, come indice della capacitá di generalizzazione del modello di complessità m, la quantità RMSE_{cv}^m, radice della media del quadrato dell'errore di cross validation:

$$\mathrm{RMSE}_{cv}^m = \frac{1}{10}\sum_{t=1}^{10}\mathrm{RMSE}_t^m,$$

con

$$\mathrm{RMSE}_t^m = \sqrt{\frac{1}{n_t}\sum_{\substack{i=1\\(x_i,y_i)\in Z_t}}^{n_t}\Big(y_i - f^m\big(x_i,\alpha;Z\backslash Z_t\big)\Big)^2},$$

dove n_t è il numero di esempi (x_i, y_i) appartenenti a Z_t, e $f^m\big(x_i,\alpha;Z\backslash Z_t\big)$ è il modello identificato utilizzando gli esempi disponibili, ad esclusione di quelli appartenenti a Z_t.

La scelta del valore ottimo del grado di complessità è stata effettuata in base alle informazioni estratte dal grafico RMSE_{cv}^m contro m. Al crescere della complessità il valore dell'indice di errore tipicamente decresce fino a stabilizzarsi, e in alcuni casi torna ad aumentare. Come valore ottimo di m è stato selezionato quello in corrispondenza del quale il RMSE_{cv}^m cessa di diminuire.

Selezionato il valore ottimo di m si è identificato un modello di tale complessità utilizzando tutti gli esempi Z. Il modello cosìottenuto è stato valutato sull'insieme dei 1024 dati *nascosti*[2]. L'identificazione finale è stata ripetuta

[1] E' pratica comune utilizzare una 10-fold cross validation quando sono disponibili campioni delle dimensioni dell'insieme Z dato (Weiss & Kulikowski, 1991). In tali condizioni questo metodo di validazione fornisce una stima sufficientemente accurata dell'erore in generalizzazione (Kohavi, 1995)

[2] Nei testi in lingua inglese si usa l'espressione *"hidden data"* per indicare quei dati non resi disponibili in fase di identidicazione parametrica o strutturale, ma tenuti da parte per una valutazione finale.

più volte per verificare la capacità dell'algoritmo di ottimizzazione, e di quello di clustering, di ritrovare la stessa soluzione anche partendo da inizializzazioni diverse[3].

Il parametro m, per i sistemi descritti ai punti [1] [2] [3] [4] [5], rappresenta il numero di modelli locali di cui è composto il sistema stesso. Nel caso del sistema *hierarchical mixture of experts* [6], m è il numero di livelli gerarchici che compongono l'albero binario che descrive il modello. Il numero dei modelli locali è pari dunque dunque a 2^m.

Per quanto riguarda il lazy learning [7], non è definito un indice m di complessità globale: per ognuno dei 1024 punti appartenente all'insieme dei dati *nascosti*, viene eseguita sia la fase di identificazione parametrica, sia quella strutturale.

6.1.2 I Risultati Sperimentali

I risultati ottenuti sono riassunti in tabella 6.1 nella quale, per ogni sistema utilizzato, sono presentati: il grado di complessità stimato come ottimo, il relativo RMSE stimato in cross validation e il RMSE misurato sui 1024 dati nascosti. Per quanto riguarda quest'ultima quantità, nel caso dei sistemi appartenenti alla famiglia neuro-fuzzy [1] [2] [3] [4], riportiamo la media e la deviazione standard dei RMSE ottenuti sui dati nascosti da 8 modelli del tipo dato e della complessità prescelta ma ottenuti a partire da una diversa inizializzazione casuale dell'algoritmo di clustering; questo per verificare quanto la soluzione ottenuta dipenda dalla particolare istanza dell'inizializzazione. Nel caso dei sistemi [5] e [6] appartenenti alla famiglia mixture of experts, riportiamo solo il valor medio del RMSE ottenuto in due identificazioni successive. Il risultato del lazy learning [7], al contrario, dipende esclusivamente dall'insieme Z assegnato, riportiamo quindi semplicemente il valore del RMSE misurato.

I migliori risultati sono stati ottenuti dal modello fuzzy [3] caratterizzato da conseguenti lineari, membership function gaussiane, e inizializzato con il metodo hyperplane fuzzy clustering. Riteniamo che la ragione di tale buona prestazione sia proprio da ricercarsi nell'abbinamento di approssimatori lineari con il metodo di inizializzazione che cerca appunto cluster che siano iperpiani nello spazio congiunto di ingresso-uscita. Inoltre, la bassa deviazione standard del RMSE è da leggere come un indice della capacità del metodo hyperplane fuzzy clustering di convergere ad una buona soluzione a partire da diverse inizializzazioni casuali.

Un'altra conclusione che si vuole trarre dai risultati presentati, riguarda il metodo lazy learning. Al contrario dei metodi fuzzy e mixture of experts,

[3]Si noti che gli algoritmi di clustering utilizzati per inizializzare le caratteristiche delle membership function necessitano a loro volta di una inizializzazione che viene effettuata in maniera casuale.

Sistema	Complessità	RMSE stimato	RMSE misurato
[1] Fuzzy	9	1.0863	1.4042 ± 0.3350
[2] Fuzzy	21	0.6081	0.6357 ± 0.0257
[3] Fuzzy	13	0.5807	0.5525 ± 0.0142
[4] RBF	22	0.7610	0.8249 ± 0.1101
[5] ME	20	0.8001	1.0050
[6] HME	3	0.6994	0.6776
[7] Lazy	-	-	0.7684

Tabella 6.1: Ricostruzione di una superficie. Un confronto tra sette sistemi appartenenti alle famiglie neuro fuzzy, lazy e mixture of experts.

il lazy learning non introduce nessun vincolo esplicito sulla *smoothness* della soluzione: ognuno dei 1024 punti appartenenti all'insieme dei dati nascosti viene trattato come un caso di predizione a se stante. A questo è da far risalire la causa della prestazione inferiore del metodo lazy in questo specifico caso, in cui la superficie da ricostruire ha effettivamente delle proprietà di smoothness.

Fuzzy Inference System
Composto da approssimatori locali lineari combinati con funzioni di appartenenza triangolari normalizzate. L'inizializzazione è stata effettuata con l'algoritmo c-mean.

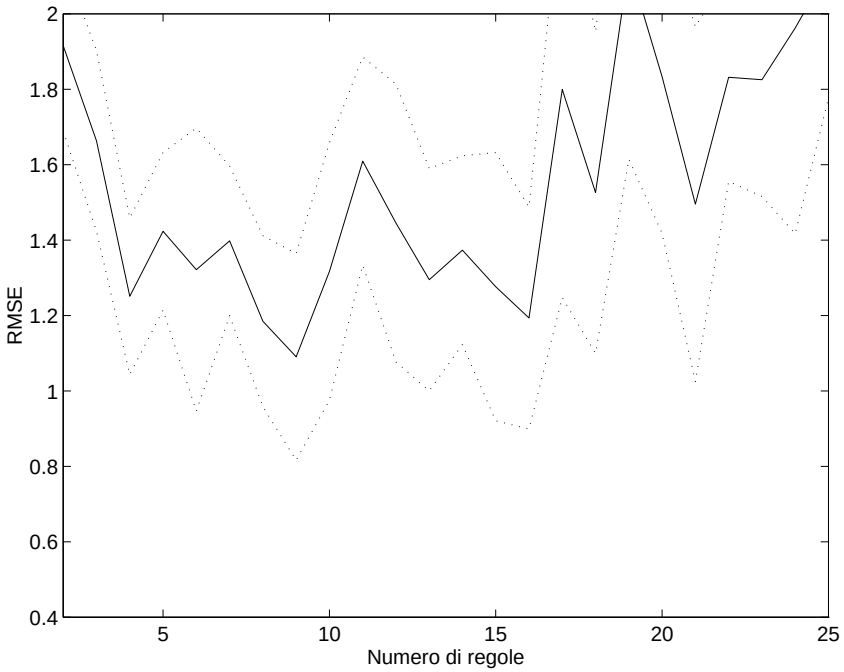


Figura 6.2: Sistema [1]: grafico della stima di cross validation dell'errore quadratico medio contro il numero di approssimatori locali. Il livello di complessità per il quale il sistema ottiene l'errore minore pari a $m = 9$.

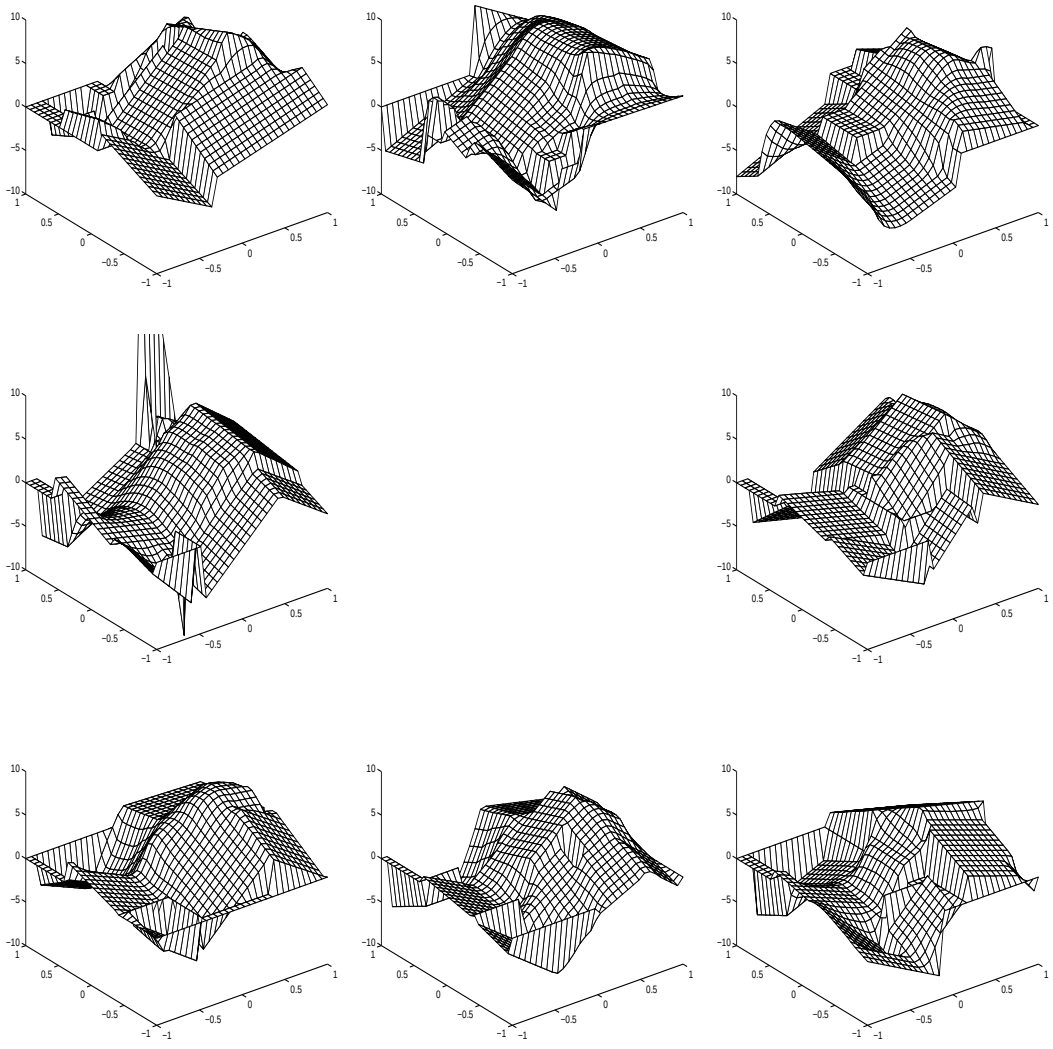


Figura 6.3: Sistema [1]: la superficie ricostruita da otto modelli fuzzy composti da 9 approssimatori locali lineari combinati da funzioni triangolari normalizzate. I modelli sono stati ottenuti fornendo all'algoritmo c-mean diverse inizializzazioni casuali per le posizioni dei prototipi. Le soluzioni ottenute sono caratterizzate da valori del RMSE nell'intervallo 1.1185—1.5128.

Fuzzy Inference System:
Composto da approssimatori locali lineari combinati attraverso funzioni di appartenenza gaussiane non normalizzate, e inizializzato con l'algoritmo c-mean.

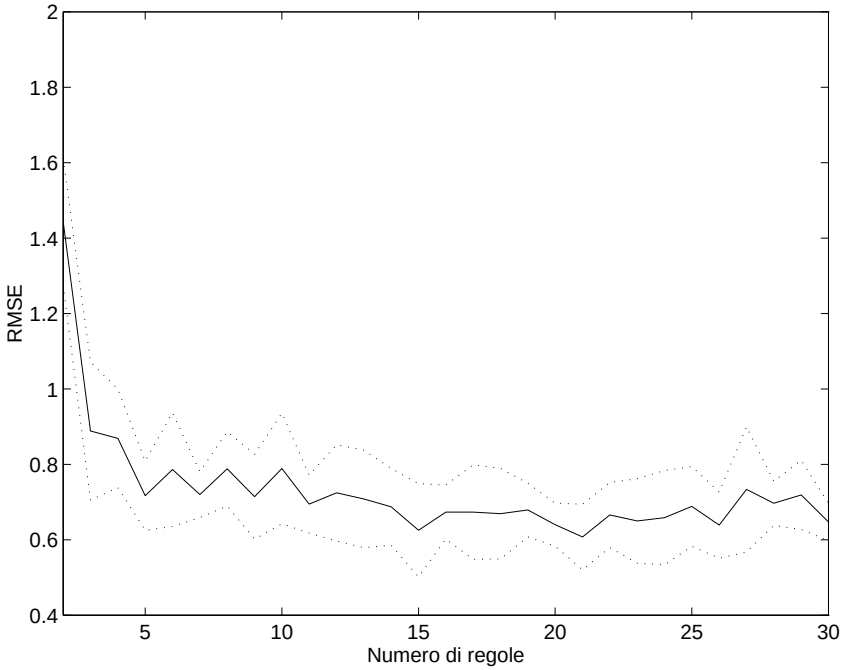


Figura 6.4: Sistema [2]: grafico della stima di cross validation dell'errore quadratico medio contro il numero di approssimatori locali. Il livello di complessità selezionata in base alle informazioni fornite da questo grafico è pari a $m = 21$.

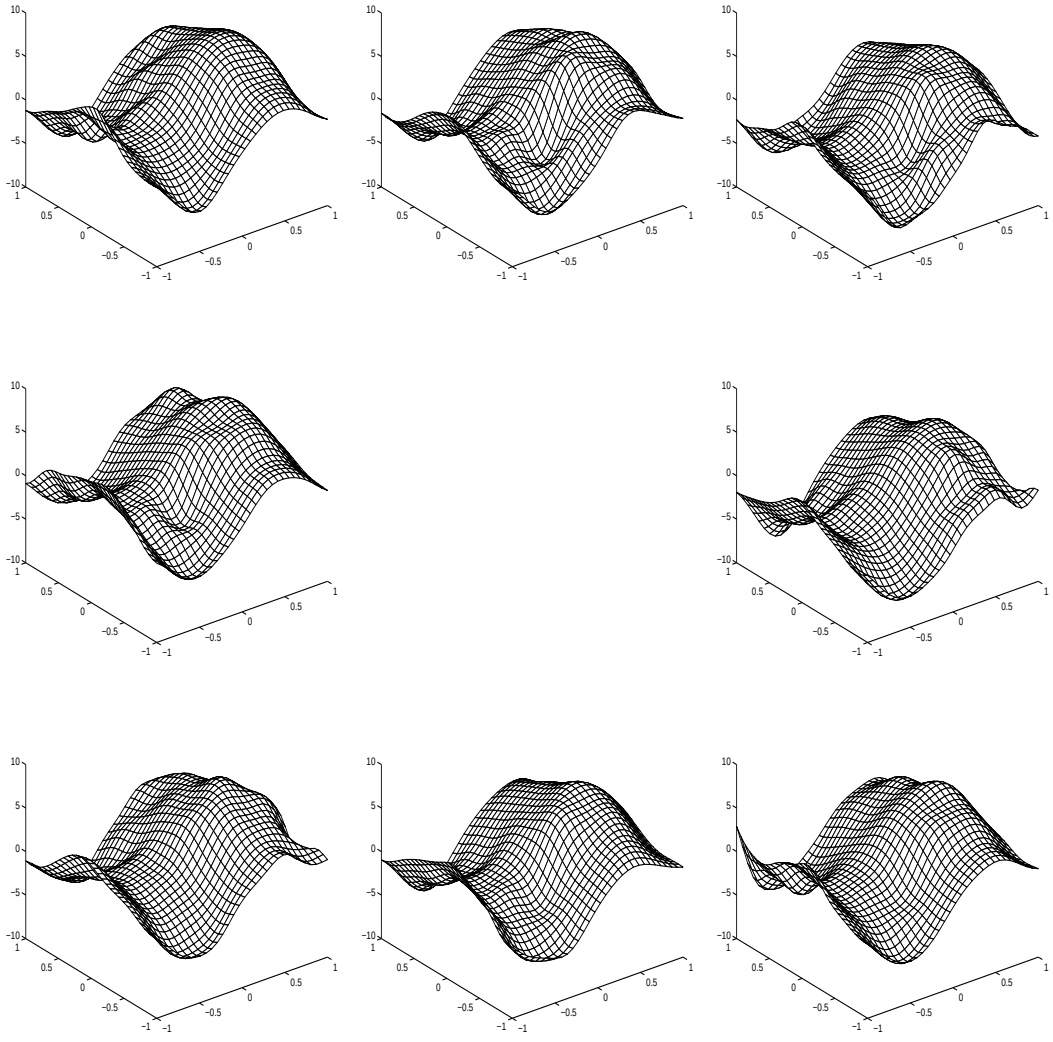


Figura 6.5: Sistema [2]: la superficie ricostruita da otto modelli fuzzy composti da 21 approssimatori locali lineari combinati da funzioni gaussiane non normalizzate. I modelli sono stati ottenuti fornendo all'algoritmo c-mean diverse inizializzazioni casuali per le posizioni dei prototipi. Le soluzioni ottenute sono caratterizzate da valori del RMSE nell'intervallo 0.5929—0.6736.

Fuzzy Inference System:
caratterizzato da funzioni di appartenenza gaussiane non normalizzate e approssimatori locali lineari. L'inizializzazione è stata effettuata con l'algoritmo *hyperplane fuzzy clustering*.

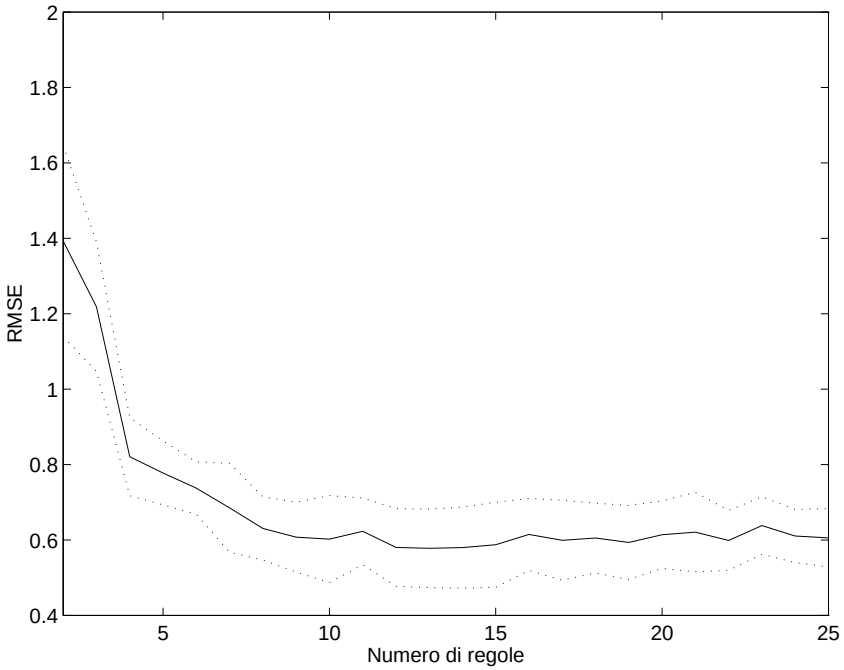


Figura 6.6: Sistema [3]: grafico della stima di cross validation dell'errore quadratico medio contro il numero di approssimatori locali. Il livello di complessità che ha consentito i migliori risultati in cross validation è $m = 13$.

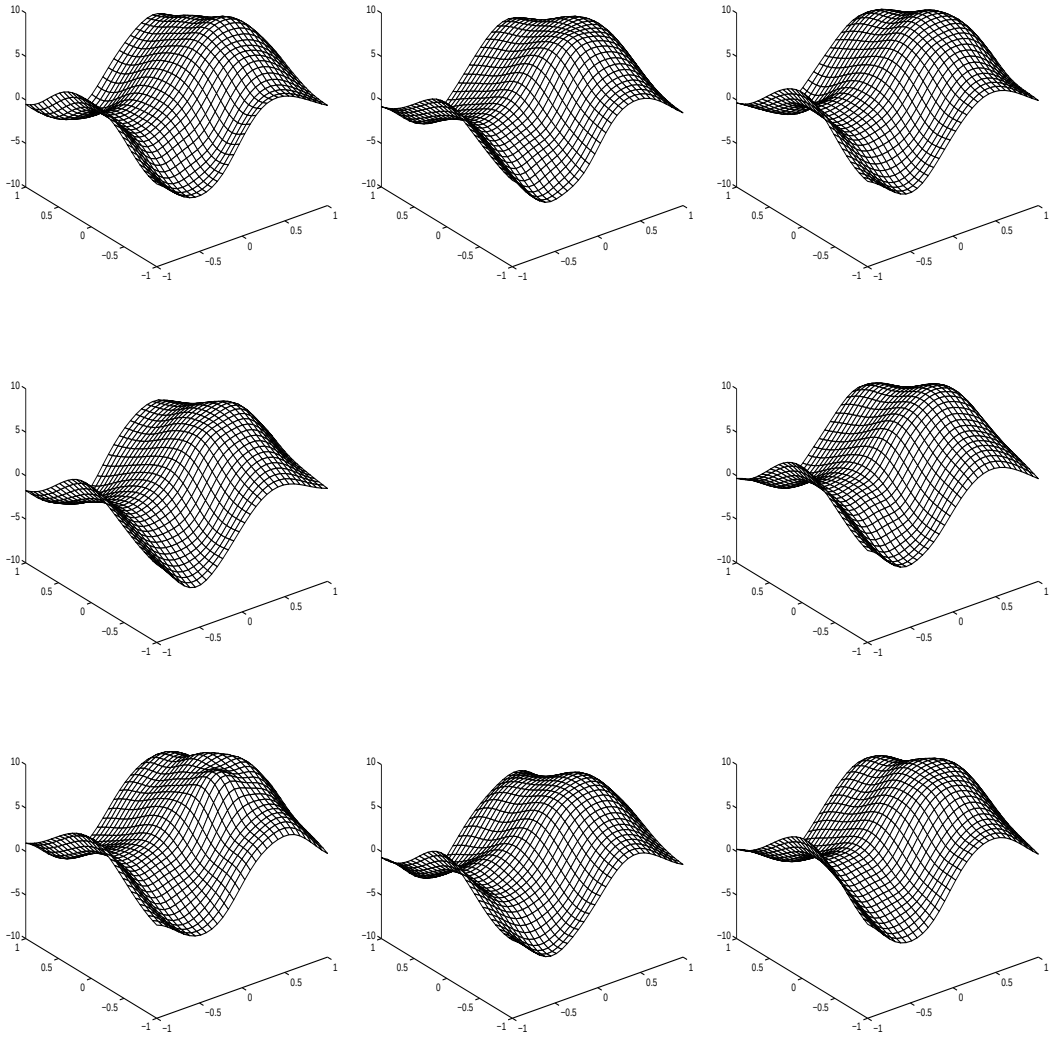


Figura 6.7: Sistema [3]: la superficie ricostruita da otto modelli fuzzy composti da 13 approssimatori locali lineari combinati da funzioni gaussiane non normalizzate. I modelli sono stati ottenuti fornendo all'algoritmo HFC diverse inizializzazioni casuali per le posizioni dei prototipi. Le soluzioni ottenute sono caratterizzate da valori del RMSE nell'intervallo 0.5357—0.5806.

Fuzzy Inference System:
ottenuto componendo attraverso funzioni di appartenenza gaussiane non normalizzate degli approssimatori locali costanti. Il modello è stato inizializzato con l'algoritmo c-mean.

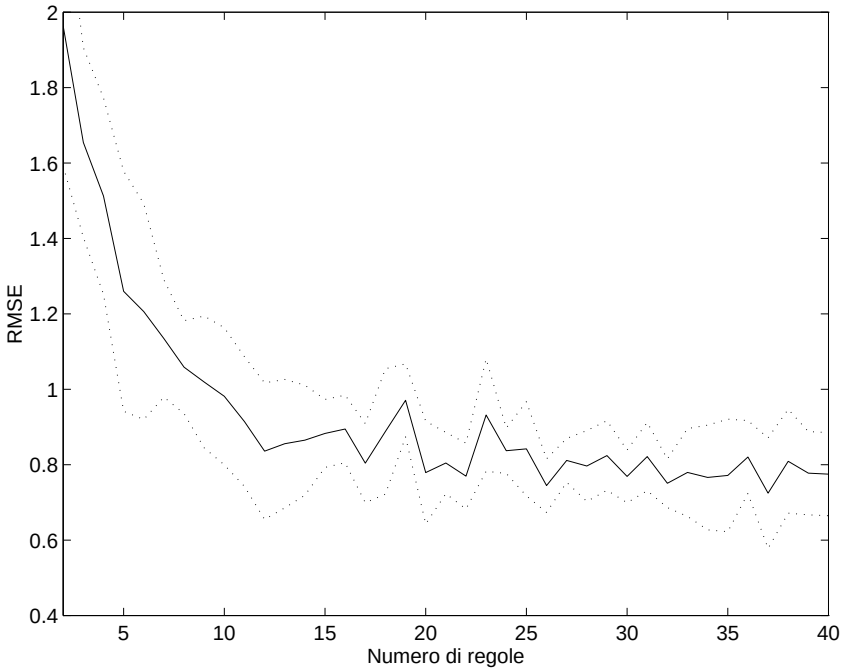


Figura 6.8: Sistema [4]: grafico della stima di cross validation dell'errore quadratico medio contro il numero di approssimatori locali. Il livello di complessità che è stato individuato come più promettente è $m = 22$.

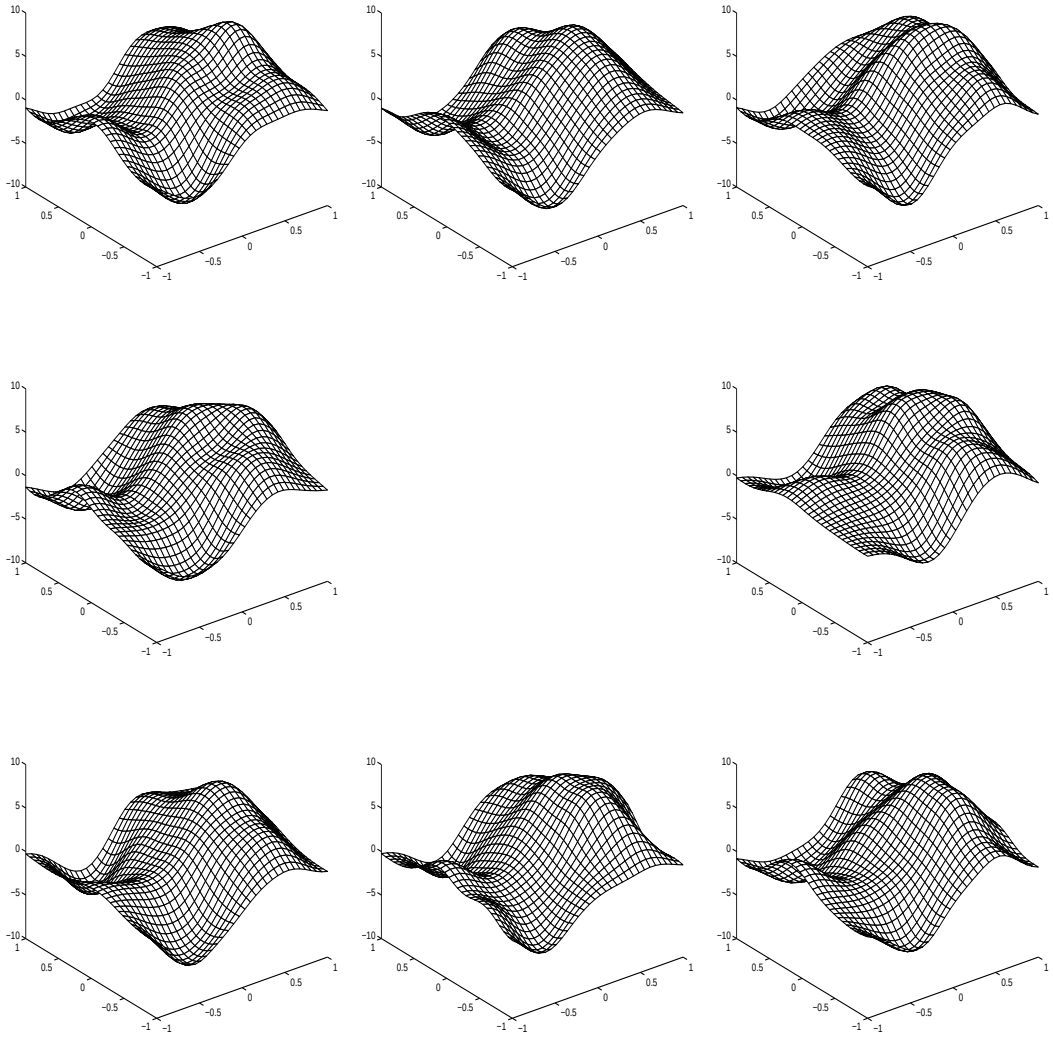


Figura 6.9: Sistema [4]: la superficie ricostruita da otto modelli fuzzy composti da 22 approssimatori locali lineari combinati da funzioni gaussiane non normalizzate. I modelli sono stati ottenuti fornendo all'algoritmo c-mean diverse inizializzazioni casuali per le posizioni dei prototipi. Le soluzioni ottenute sono caratterizzate da valori del RMSE nell'intervallo 0.6091—0.9835.

Mixture of experts:
un solo gating network arbitra più approssimatori locali, tutti allo stesso livallo gerarchico.

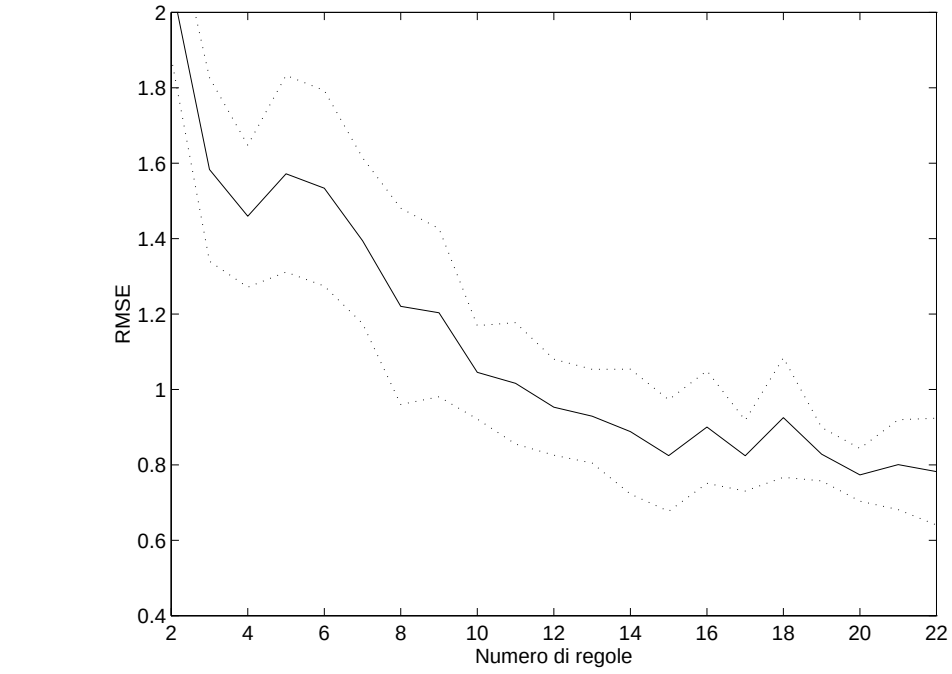


Figura 6.10: Sistema [5]: il numero ottimo di modelli locali suggerito dal processo di cross validation è $m = 20$. I valori del RMSE ottenuti in due successive identificazioni, partendo da inizializzazioni diverse, di un modello mixture of experts composto da 20 regole, sono pari a 1.0157 e a 0.9944.

Hierarchical mixture of experts:
è una struttura gerarchica descritta da un albero binario.

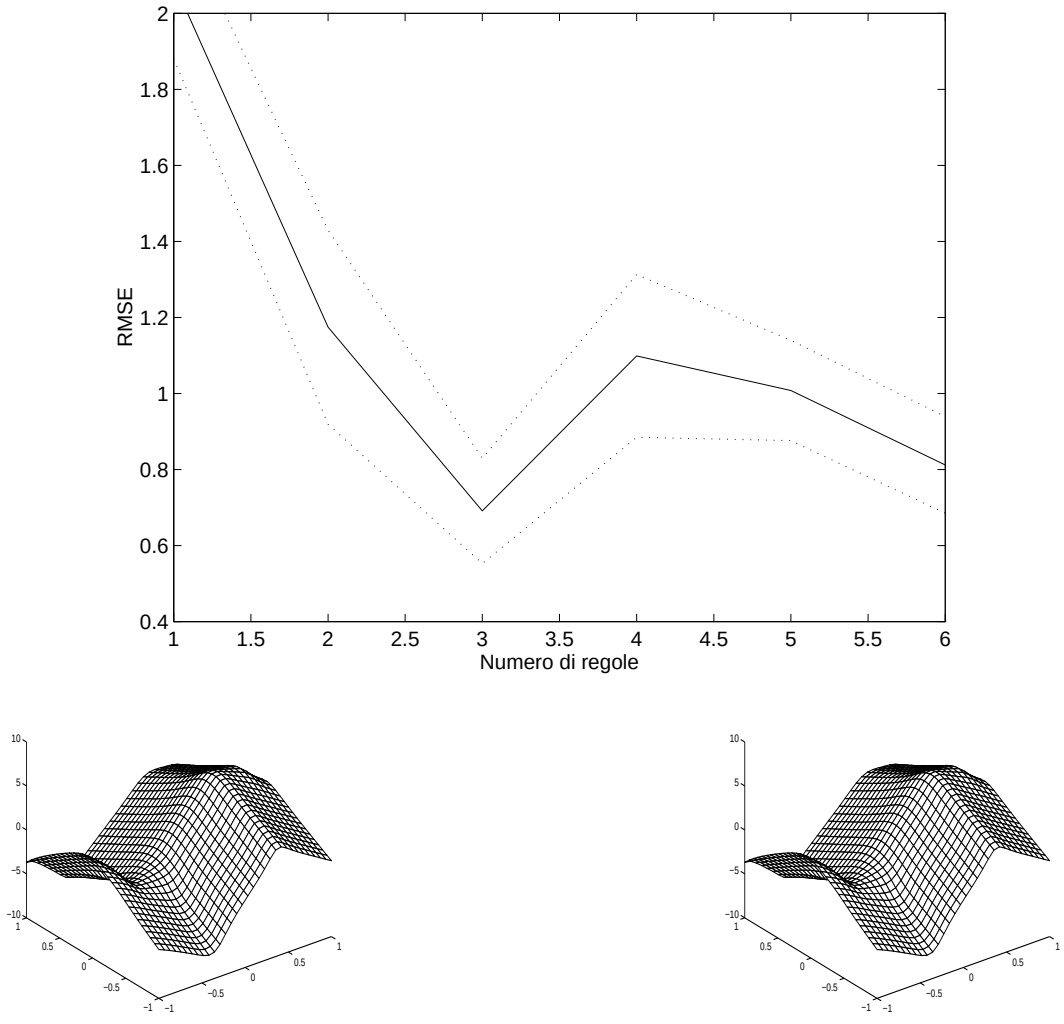
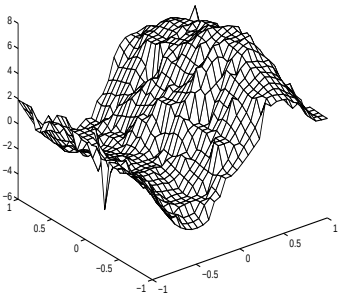
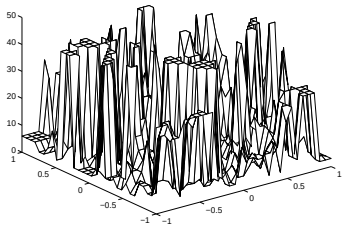


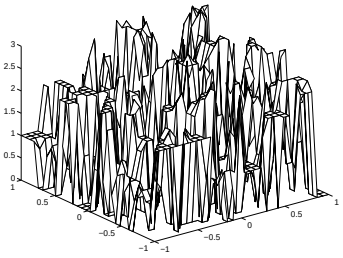
Figura 6.11: Sistema [6]: il numero ottimo di modelli locali suggerito dal processo di cross validation è pari a 8, 3 livelli gerarchici di un albero binario. I valori del RMSE ottenuti in due successive identificazioni sono pari a 0.6708 e a 0.6844.

Lazy Learning: Gli approssimatori locali appartengono alla famiglia *polynomial mixing* e la funzione di peso utilizzata è la funzione *tricubica*. Per ogni punto per cui è richiesta una predizione dell'uscita, vengono selezionati i valori ottimi di p e di k, rispettivamente grado del polynomial mixing e numero di vicini considerati.





Numero di vicini considerati



Grado del polynomial mixing

Figura 6.12: Sistema [7]: La superficie ricostruita dal metodo lazy learning. Il valore del RMSE ottenuto è pari a 0.7684. Riportiamo anche due grafici relativi al numero k di vicini considerati per ogni singola stima e il grado del polynomial mixing adottato. Questi valori sono stati selezionati per ogni valore dell'ingresso sulla base delle informazioni fornite da un processo di validazione locale.

6.2 Predizione di una serie temporale

Come secondo esperimento, presentiamo un caso di predizione di una serie temporale caotica. E' stato considerato il problema della predizione della serie Mackey-Glass (Mackey & Glass, 1977) descritta dalla seguente equazione differenziale:

$$\dot{x}(t) = \frac{0.2\, x(t-\tau)}{1 + x(t-\tau)^{10}} - 0.1x(t), \tag{6.2}$$

con $x(t) = 0$ per $t < 0$, $x(0) = 1.2$, e $\tau = 17$. Per poter confrontare direttamente i risultati con quelli ottenuti con altri metodi e già presenti in letteratura, ci siamo messi nelle stesse condizioni sperimentali definite da Platt (Platt, 1991) come segue. Lo scopo è predire il valore $x(t + 85)$ utilizzando $x(t)$, $x(t-6)$, $x(t-12)$, $x(t-18)$. L'equazione (6.2) è stata integrata con un metodo Runge-Kutta del secondo ordine, con passo di integrazione pari a 0.1. Come insieme Z degli esempi disponibili, si sono presi i 500 passi temporali a partire da $t = 200$, mentre come *dati nascosti*, cioè quelli per cui è richiesta la predizione, sono stati considerati i 500 a partire da $t = 5000$. Qui la predizione di una serie temporale è stata ricondotta al problema dell'approssimazione funzionale attraverso la tecnica che prende il nome di *delay coordinate embedding* (Sauer, 1993) L'indice considerato per valutare la qualità della soluzione, è il *non-dimensional error index* (NDEI), definito come il rapporto tra il RMSE e la deviazione standard della serie che si intende predire.

Il problema così definito è stato affrontato utilizzando il sistema lazy learning adattativo, e il sistema fuzzy che ha consentito i migliori risultati nell'esperimento di ricostruzione di una superficie, cioè il sistema composto da approssimatori locali lineari, funzioni di appartenenza gaussiane non normalizzate, e inizializzato utilizzando l'algoritmo hyperplane fuzzy clustering.

Abbiamo inoltre provato un sistema lazy learning in cui la *PRESS statistic* è stata utilizzata anche in fase di *feature selection* (§ 5.3). Per ogni passo t l'algoritmo che abbiamo implementato valuta, in base alle informazioni fornite dalla *PRESS statistic*, se utilizzare come predittori del valore $x(t+85)$, tutti i campioni $x(t)$, $x(t-6)$, $x(t-12)$, $x(t-18)$, oppure se escludere il campione $x(t-18)$ (Bersini *et al.*, 1997a).

I risultati ottenuti sono presentati in tabella 6.2, insieme ai risultati trovati in letteratura e ottenuti nelle stesse condizioni sperimentali. Di seguito sono presentati i grafici relativi agli esperimenti effettuati. Il metodo *lazy learning*, per questo problema fornisce i risultati migliori. I 500 dati disponibili non forniscono una copertura ottimale dello spazio quadri-dimensionale di ingresso, così come nell'esperimento precedente erano in grado di fare 200 esempi in uno spazio bi-dimensionale. La scarsità di dati penalizza i metodo *model-based* più di quanto non penalizzi quelli *memory-based*.

Un altro aspetto che occorre sottolineare è il buon risultato ottenuto utilizzando il metodo di validazione basato sulla *PRESS statistic* anche in fase di *feature selection*.

Model		NRMSE
RAN	$\epsilon = 0.02$	0.075
	$\epsilon = 0.05$	0.077
Incremental Fuzzy	with L.-M.	0.096
	without L.-M.	0.12
Fuzzy	HFC — 22 regole	0.3022
Lazy Learning		0.0594
Lazy Learning	feature selection	0.0540

Tabella 6.2: Predizione della serie caotica Mackey-Glass. Un confronto tra diversi metodi di apprendimento. Si riportano i risultati ottenuti con i metodi sviluppati con quelli ottenuti nelle stesse condizioni sperimentali dal metodo RAN (Platt, 1991), e da un metodo Fuzzy Incrementale (Bersini *et al.*, 1997b) ottimizzato con il metodo Levenberg-Marquardt nel primo caso e con il metodo del gradiente nel secondo.

Fuzzy Inference System:
composto da approssimatori locali lineari combinati attraverso funzioni di appartenenza gaussiane non normalizzate. Il modello è stato inizializzato con l'algoritmo *hyperplane fuzzy clustering*.

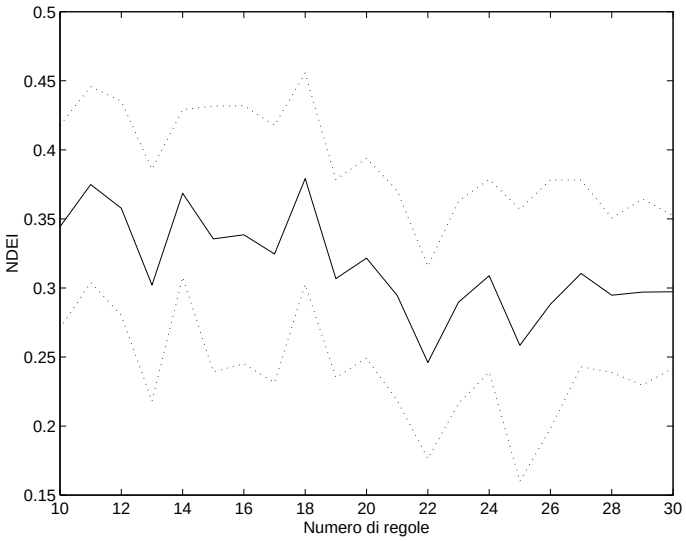


Figura 6.13: Stima dell'errore in generalizzazione contro complessità. Il grafico non ha in questo caso un comportamento regolare che consenta di individuare con una relativa sicurezza un numero di ottimale di approssimatori locali. Il valore più promettente è $m = 22$.

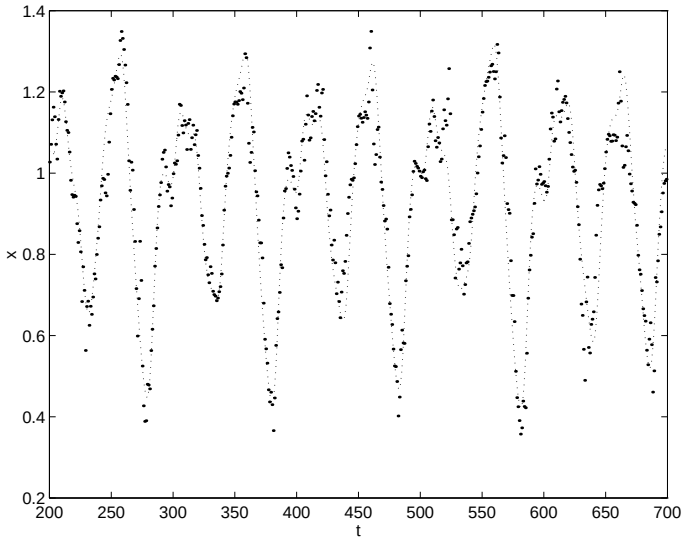


Figura 6.14: Predizione della serie temporale effettuata utilizzando il modello fuzzy costituito da 22 approssimatori locali. Il NRMSE ottenuto è pari a 0.3022: un risultato non soddisfacente.

Lazy Learning:
Gli approssimatori locali appartengono alla famiglia *polynomial mixing* e la funzione di peso utilizzataè la funzione *tricubica*. Per ogni valore di t, vengono selezionati i valori ottimi di p e di k, rispettivamente grado del polynomial mixing e numero di vicini considerati.

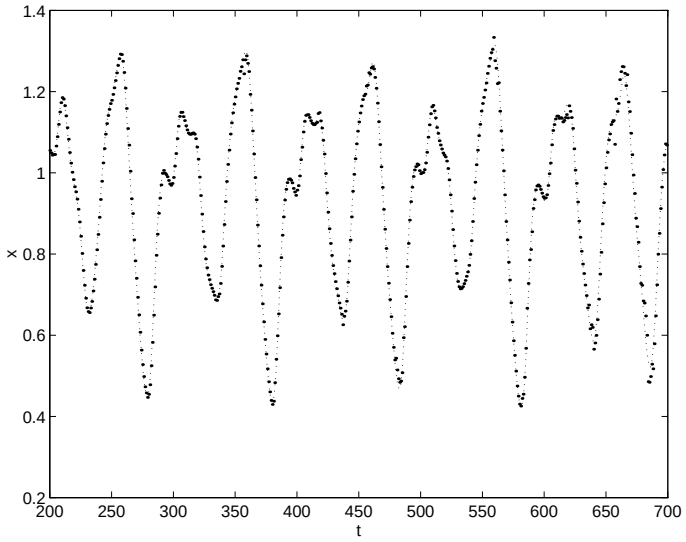


Figura 6.15: Il metodo lazy learning è in grado di fornire una predizione molto buona della serie Mackey-Glass. La soluzione ottenuta è caratterizzata da un valore dell'indice di errore non dimensionale pari a NRMSE = 0.0594.

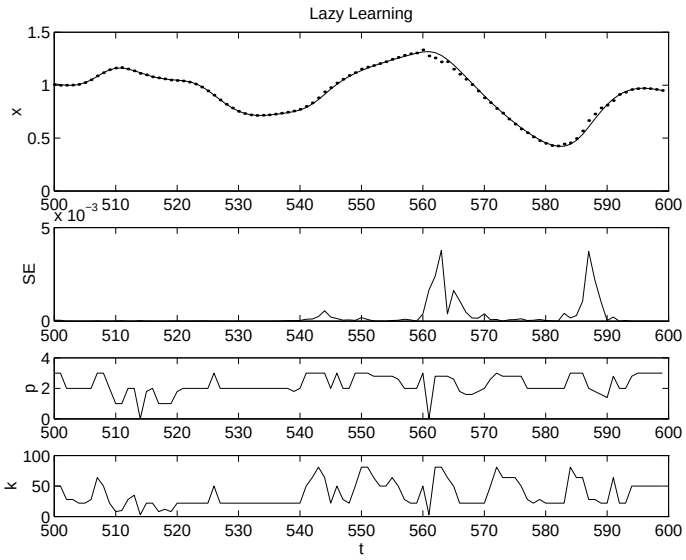


Figura 6.16: La figura mostra la predizione fornita in una finestra di 100 passi temporali, e il quadrato dell'errore di predizione commesso (SE). Insieme viene fornito anche il grado (p) del polynomial mixing utilizzato, e il numero di esempi

Lazy Learning e Feature selection:
Il metodo è un'estensione del precedente: per ogni valore di t, vengono selezionati i valori ottimi di p e di k, e viene valutato, sempre sfruttando le informazioni fornite dal metodo di validazione locale, se la variabile $x(t-18)$ può fornire o meno informazioni utili per predire il valore $x(t+85)$.

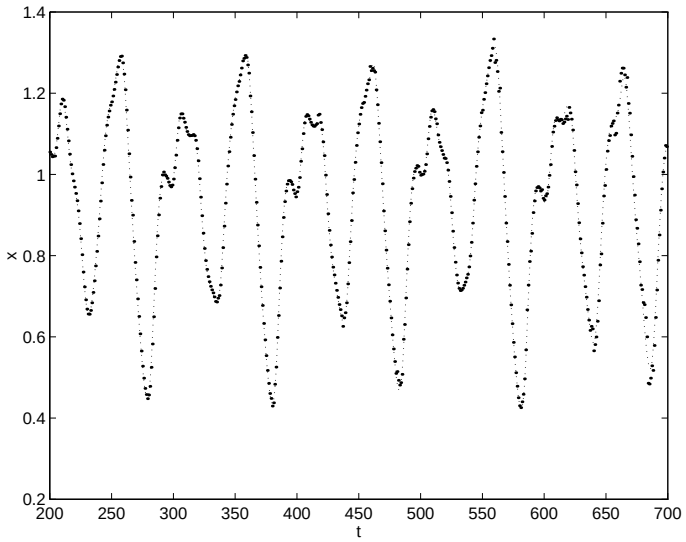


Figura 6.17: La selezione locale del numero di variabili porta un ulteriore miglioramento della qualità della predizione. Il valore dell'indice non dimensionale di errore è NRMSE = 0.540.

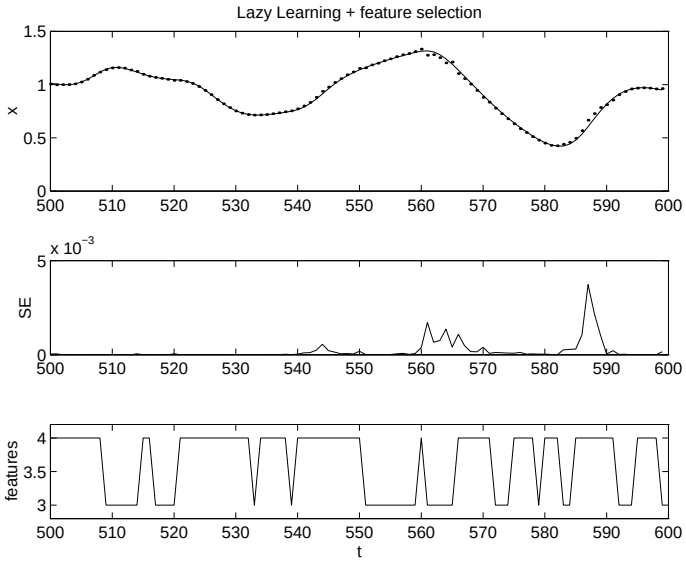


Figura 6.18: Oltre al valore della predizione e al quadrato dell'errore commesso finestra di 100 passi temporali, viene presentato il numero di predittori considerati per ogni valore di t.

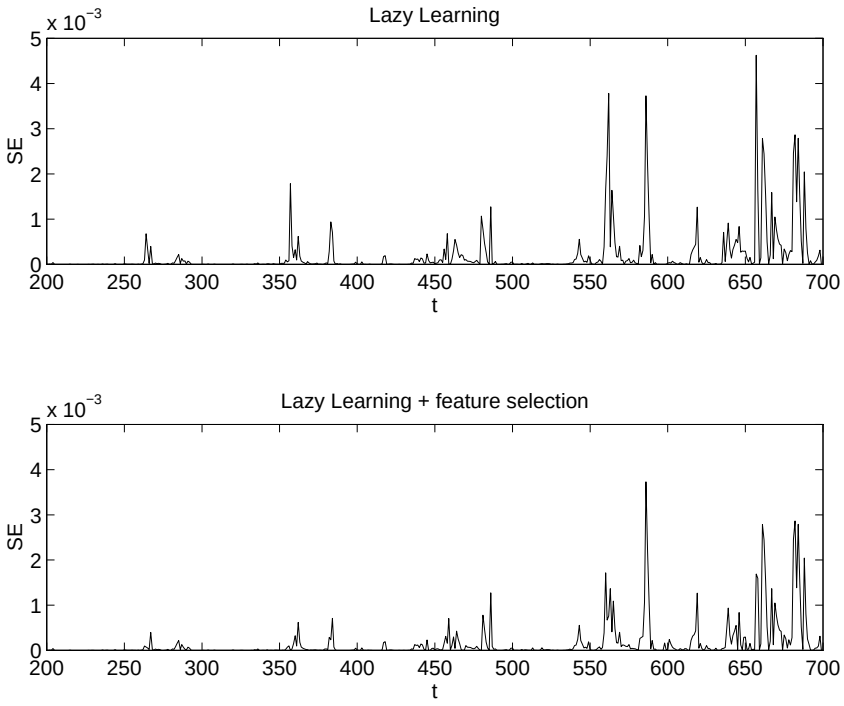


Figura 6.19: Per un confronto diretto dei risultati ottenuti dai due metodi lazy learning, si riportano i diagrammi dell'andamento del quadrato dell'errore di predizione nel tempo.

6.3 Un caso reale: analisi di mercati finanziari

Il metodo *lazy learning* è stato applicato a un caso di predizione nell'ambito finanziario. Il progetto di ricerca, in collaborazione con una multinazionale che opera nel settore, è stato svolto presso l'istituto IRIDIA–Université Libre de Bruxelles. Illustreremo di seguito il tipo di problema, la metodologia applicata e riporteremo una serie di risultati sperimentali. Commenteremo inoltre brevemente le ragioni per le quali si è scelto di affrontare il problema dato con il metodo lazy learning, e più in generale perchè i metodi memory-based sono considerati promettenti per applicazioni di questo tipo.

Il problema affrontato è tipico nel campo dell'analisi finanziaria: sono dati due mercati, noti ciascuno attraverso una serie storica dei profitti consentiti; di tali mercati si desidera conoscere in anticipo l'andamento futuro per poter scegliere su quale dei due operare. Nel nostro caso specifico dei mercati analizzati, le serie storiche presentano caratteristiche differenti. Quella relativa al primo mercato, definita $a(t)$, ha escursioni relativamente limitate e consente mediamente profitti positivi. Quella relativa al secondo, $b(t)$, ha una varianza maggiore e un valor medio negativo.

Si ponga uguale a S il massimo profitto ottenibile in un intervallo temporale $T = [t_i, t_f]$, cioè il profitto che si otterrebbe conoscendo in anticipo l'evoluzione e quindi investendo ad ogni passo temporale sul mercato di profitto più elevato. Il profitto ottenuto applicando una qualsiasi strategia di investimento, può essere convenientemente espresso in termini di punti percentuali di S. La strategia adottata dal committente di questa ricerca è stata fino ad ora la strategia banale che consiste nell'investire sempre sul mercato a ottenendo un profitto pari a circa il 14% di S. Oltre alla strategia banale sono state considerate altre strategie basate sul'utilizzo di strumenti di predizione lineari: i risultati ottenuti sono però sempre stati insoddisfacenti a causa della forte irregolarità e scarsa predicibilità, nell'ottica lineare, delle serie $a(\cdot)$ e $b(\cdot)$. Le strategie che impiegano strumenti tradizionali di predizione consentirebbero profitti inferiori a quelli consentiti dalla strategia banale, o di poco superiori a questi ma mai a sufficienza, secondo l'avviso del nostro partner, da giustificare l'adozione di un'altra strategia.

Avendo il metodo lazy learning mostrato di essere in grado di fornire buoni risultati nella predizione di serie temporali caotiche (§ 6.2), si è deciso di adottarlo per affrontare il problema in esame. Il metodo lazy, proprio per il fatto che ritarda ogni elaborazione dei dati fino al momento in cui è esplicitamente richiesta una predizione, e dato che per ogni predizione ripete l'intera fase di estrazione di conoscenza dai dati disponibili, si presenta come un metodo promettente per l'elaborazione *on-line* di serie temporali. Consente infatti di trattare naturalmente situazioni in cui la quantità di dati cresce con il tempo e si desidera usare sempre i dati più recenti per ogni predizione.

Sfuttando la conoscenza in possesso degli analisti della società committente, conoscenza acquisita anche attraverso i tentativi di modellizzazione con metodi lineari, il problema è stato formalizzato nel modo seguente. Dalle due serie $a(\cdot)$ e $b(\cdot)$, contenenti ciascuna 3000 passi temporali, sono stati estratti un insieme di esempi $z_t = (x_t, y_t)$, dove x_t è una ricostruzione dello stato delle due serie al tempo t, secondo il metodo di rappresentazione *delay coordinate vector* (Sauer, 1993); mentre y_t è uno scalare che riassume l'informazione relativa al valore assunto dalle due serie al tempo $t + 1$. In diversi esperimenti successivi si è provato a costruire in maniera differente sia il vettore x, sia lo scalare y, alla ricerca della rappresentazione che consentisse la migliore predizione. In particolare si è definito il valore di y_t in due modi diversi. Il primo consiste nel porre $y_t = a(t + 1) - b(t + 1)$, l'altro, quello infine adottato, consiste nel porre y_t uguale a $+1$ o a -1 a seconda che sia $a(t + 1) > b(t + 1)$ o viceversa. Per quanto riguarda il vettore x_t, sono state provate più soluzioni: il generico elemento del vettore è il valore assunto da una delle due serie in un istante $\tau \leq t$ o una combinazione lineare di tali valori. Gli esempi $Z = \{z_t\}_{t=1}^{n}$ così ottenuti sono stati usati per realizzare un sistema di supporto alle decisioni basato sul metodo lazy. Se al tempo t_* si desidera ottenere una stima del profitto consentito dagli investimenti sui due mercati al tempo $t_* + 1$, è necessario costruire il vettore x_* in base ai valori assunti da $a(\cdot)$ e $b(\cdot)$ negli istanti precedenti a t_* e identificare un modello locale centrato in x_*, utilizzando gli esempi appartenenti a Z che si trovano in una regione circostante a x_* stesso. La predizione del valore y_* può essere utilizzata per individuare il mercato sul quale è più conveniente investire.

Presentiamo di seguito una selezione degli esperimenti effettuati. Per consistenza con altri esperimenti effettuati in passato ci è stato chiesto di simulare investimenti sui due mercati per quanto riguarda i passi temporali da $t = 1001$ a $t = 3000$. Nella maggior parte degli esperimenti sono stati impiegati, come esempi, i 500 estratti dalla prima parte delle serie temporali. Si è provato anche ad usare per predire l'andamento dei mercati al tempo τ esempi estratti dalla finestra temporale da $t = \tau - 500$ a $t = \tau - 1$: faremo riferimento a questo esperimento con il nome di *moving window*.

1. Le serie $a(\cdot)$ e $b(\cdot)$ sono state scalate (§ A.3), y_t può assumere valore $+1$ o -1, e lo stato delle serie è ricostruito attraverso il vettore composto dai valori assunti dalle serie $a(\cdot)$ e $b(\cdot)$ al tempo t, $t - 1$ e $t - 2$. Tali campioni sono stati consigliati dagli analisti finanziari con cui abbiamo lavorato. La predizione è effettuata sulla base dei primi 500 esempi.

2. Le serie $a(\cdot)$ e $b(\cdot)$ sono state scalate (§ A.3), y_t è pari alla differenza tra $a(t + 1)$ e $b(t + 1)$, e lo stato delle serie è ricostruito attraverso il

vettore composto dai valori assunti dalle serie $a(\cdot)$ e $b(\cdot)$ al tempo t, $t-1$ e $t-2$. La predizione è effettuata sulla base dei primi 500 esempi.

3. Le serie $a(\cdot)$ e $b(\cdot)$ sono state scalate (§ A.3), y_t può assumere valore $+1$ o -1, e lo stato delle serie è ricostruito attraverso il vettore composto dai valori assunti dalle serie $a(\cdot)$, $b(\cdot)$ e $v(\cdot)$ al tempo t, $t-1$ e $t-2$, dove $v(t)$ è una combinazione lineare di valori assunti dalle serie $a(\cdot)$ e $b(\cdot)$ in istanti precedenti a t, pesati da coefficienti che ci sono stati forniti dagli analisti finanziari che hanno partecipato a questo studio. Tali coefficienti provengono da precedenti ricerche effettuate con modelli lineari. Anche la serie $v(\cdot)$ è stata scalata tra $+1$ e -1. La predizione è effettuata sulla base dei primi 500 esempi.

4. Le serie $a(\cdot)$ e $b(\cdot)$ sono state scalate (§ A.3), y_t è pari alla differenza tra $a(t+1)$ e $b(t+1)$, e lo stato delle serie è ricostruito attraverso il vettore composto dai valori assunti dalle serie $a(\cdot)$, $b(\cdot)$ e $v(\cdot)$ al tempo t, $t-1$ e $t-2$, dove la serie $v(\cdot)$ è la stessa introdotta nella descrizione [3]. La predizione è effettuata sulla base dei primi 500 esempi.

5. *Moving window*: Come nell'esperimento [1] ma gli esempi considerati sono i 500 più recenti.

6. *Moving window*: Come nell'esperimento [3] ma gli esempi considerati sono i 500 più recenti.

7. Come nell'esperimento [3] ma invece di considerare la serie $v(\cdot)$ scalata, si è considerata la sua differenza prima.

8. Le serie $a(\cdot)$ e $b(\cdot)$ in questo caso non sono state scalate, y_t può assumere valore $+1$ o -1, e lo stato delle serie è ricostruito attraverso il vettore composto dai valori assunti dalle due serie al tempo t, e al tempo $t-1$, e dalla media dei valori assunti tra $t-2$ e $t-5$, tra $t-6$ e $t-13$ e tra $t-14$. La predizione è effettuata sulla base dei primi 500 esempi.

9. *Feature selection*: la scelta di quanti passi temporali utilizzare è resa adattativa. Stesso schema dell'esperimento [1] ma per ogni predizione, in base ad un processo di valivazione locale, viene scelto se utilizzare i valori assunti da $a(\cdot)$ e $b(\cdot)$ solo al tempo t, al tempo t e $t-1$, ... , al tempo t $t-1$ $t-2$ $t-3$ $t-4$

10. *Feature selection*: come nell'esperimento [1], ma per ogni predizione viene scelto se utilizzare i valori assunti dalle due serie solo al tempo t, oppure anche a $t-1$, anche la media dei valori assunti tra $t-2$ e $t-5$, tra $t-6$ e $t-13$ e tra $t-14$ e $t-29$. Anche in questo caso si è composto il vettore x, da usare come rappresentazione dello stato, accogliendo il

consiglio degli analisti. La predizione è effettuata sulla base dei primi 500 esempi.

11. *Feature selection* e *movimg window*: come nell'esperimento [10], ma gli esempi considerati sono i 500 più recenti.

12. *Feature selection*: come nell'esperimento [9], ma considerando anche la variabile $v(\cdot)$.

13. *Feature selection*: come nell'esperimento [10], ma considerando anche la variabile $v(\cdot)$.

Un riassunto dei risultati degli esperimenti sopra descritti è riportato in tabella 6.3. Il risultato migliore è stato ottenuto con l'esperimento [3] utilizzando anche la serie $v(t)$. E' stato però molto apprezzato il risultato ottenuto nell'esperimento [10] *feature selection* che mette in evidenza le buone capacità adattative del metodo. Non soddisfacente è invece la prestazione dei sistemi *moving window*: si evidenzia un calo delle prestazioni quando vengono utilizzati esempi recenti. La ragione di questo è probabilmente da ricercarsi in una maggiore capacità predittiva da parte dei dati appartenenti alla finestra temporale iniziale.

Per il futuro si prevede di effettuare un altro gruppo di esperimenti sulle stesse serie temporali incrementando ad ogni passo il numero di esempi considerati. Si proverà cioè a fornire una predizione della grandezza $y(\tau)$ utilizzando tutti gli esempi con $0 \leq t \leq \tau - 1$, per vedere se l'algoritmo possiede gli strumenti per selezionare automaticamente gli esempi che possiedono maggior capacità predittiva, in questo caso quelli appartenenti alla prima finestra temporale, e a non considerare gli altri.

Esperimento	1	2	3	4	5
Risultato	13.64%	13.01%	18.88%	14.82%	12.47%

Esperimento	6	7	8	9	10
Risultato	10.10%	12.90%	14.43%	15.54%	17.76%

Esperimento	11	12	13	strategia banale
Risultato	12.87%	11.41%	15.60%	13.90%

Tabella 6.3: I risultati ottenuti attraverso diverse configurazioni del metodo lazy learning. I valori riportati sono espressi come percentuale del profitto massimo ottenibile conoscendo in anticipo quale dei mercati consente il maggior profitto ad ogni passo temporale.

Conclusioni

Gli strumenti dell'identificazione lineare costituiscono un notevole patrimonio di conoscenza nel campo della modellistica, che però si rivela inadatto per fornire una rappresentazione soddisfacente di molti fenomeni complessi che sempre più frequentemente la scienza e l'ingegneria si trovano a dover trattare.

Lo studio dei sistemi biologici, la crescente necessità di comprendere più a fondo i fenomeni ambientali, la scoperta affascinante di comportamenti caotici da parte di molti sistemi dinamici, ma anche problemi di modellistica legati alla produzione industriale o le esigenze del settore finanziario, hanno sottolineato l'inadeguatezza degli strumenti tradizionali e richiesto alla comunità scientifica nuovi strumenti per trattare la complessità e la non linearità. Dal punto di vista ingegneristico, l'approccio multimodello si propone come la più naturale delle risposte al problema. I metodi presentati in questa tesi condividono un'idea di fondo: anche quando la realtà, fortunatamente variegata e non lineare, presenta un comportamento non descrivibile da un unico modello, è spesso possibile individuare una collezione di modelli che, opportunamente combinati, consentano di rappresentarne la complessità.

"Divide et impera": il punto di forza dei metodi multimodello sta appunto nella loro capacità di dividere il problema dell'approssimazione di una funzione complessa e non lineare, nota solo attraverso un insieme di campioni, in una serie di sotto domini all'interno di ciascuno dei quali il problema è affrontabile con gli strumenti disponibili. I metodi visti, in particolare, consentono di affrontare il problema dell'apprendimento da esempi, riconducendolo all'interno di un contesto nel quale possono convenientemente essere sfruttate le conoscenze e gli strumenti della statistica e dell'identificazione tradizionale. *Mixture of Experts*, *Neuro-Fuzzy* e *Lazy Learning*, sono tre possibili forme di architettura multimodello, sviluppate ciascuna in ambienti scientifici diversi, e quindi con caratteristiche diverse e pensati per rispondere ad esigenze diverse.

Il primo criterio in base al quale risulta naturale tentare una classificazione di queste tre architetture ha a che vedere con il *concetto di modello*. Le prime due architetture esaminate, *Mixture of Experts* e *Neuro-Fuzzy*, ricostruiscono, a partire da una serie di approssimatori locali, un modello globale della funzione data: *"ex pluribus unum"*. L'approccio *Lazy Learning* spinge oltre il concetto di località e non fornisce nessuna rappresentazione esplicita della

funzione in esame nella sua globalità ma si limita ad essere un algoritmo per estrarre una predizione da una base di esempi. Nel primo caso potremmo dire che l'obiettivo è la stima di una funzione mentre nel secondo caso si limita ad essere la stima del valore assunto da una funzione in un ben determinato punto. Il secondo degli approcci così definiti consente migliori risultati quando sono richieste poche valutazioni della funzione in esame, mentre il primo si rivela più efficace, in media, quando sono richieste più valutazioni. Da altri punti di vista l'approccio lazy può comportare degli svantaggi: per fornire una stima del valore assunto dalla funzione in esame in un punto senza costruire una rappresentazione globale, il metodo lazy deve avere sempre a disposizione l'intero insieme di esempi dati e, ogni volta che è richiesta una stima, identificare un modello locale *ad hoc*. Lo schema *memory-based* così definito può rivelarsi non impiegabile in quelle situazioni in cui non è possibile mantenere l'intero insieme di esempi ed è necessaria una forma di rappresentazione più economica della funzione in esame. Inoltre, non costruendo un modello della funzione in esame, l'algoritmo lazy learning richiede un certo sforzo computazionale per ogni singola valutazione. Ciò rende più problematico l'utilizzo di questi metodi in quelle applicazioni in cui è richiesta una stima in *tempo reale*; in questi casi si rivelano più adatti metodi *model-based*, quali *Mixture of Experts* e *Neuro-Fuzzy*, per i quali il costo computazionale di una stima è semplicemente quello della valutazione di un modello precedentemente identificato.

Un secondo criterio in base al quale collocare le architetture studiate ha a che vedere con il tipo di interazione tra i diversi modelli locali. Nel caso del *Mixture of Experts*, l'interazione avviene all'insegna della *competizione*. Sia in fase di identificazione dell'architettura, sia in fase di valutazione, gli approssimatori locali competono per aggiudicarsi i campioni della funzione data. In particolare durante l'identificazione, la partizione dello spazio di ingresso in regioni di competenza viene ottenuta attribuendo ogni esempio disponibile all'esperto che meglio riesce a spiegarlo e solo tale esperto ha il diritto di utilizzare l'esempio in questione per aggiornare la propria legge di approssimazione. L'obiettivo di questa fase è di specializzare il più possibile gli esperti in maniera tale che ciascuno diventi *esperto*, appunto, di un definito dominio. Un modello *Neuro-Fuzzy* è invece caratterizzato dalla *cooperazione* dei diversi approssimatori. La caratteristica innovativa dell'algebra fuzzy è l'introduzione del concetto di grado di appartenenza di un oggetto ad un insieme. Nel settore della modellistica multimodello il punto di vista fuzzy porta a vedere ogni campione della funzione in esame come un oggetto caratterizzato da un grado di appartenenza in ciascuno degli insiemi di punti in cui la funzione data risulta approssimabile dallo stesso modello locale. E' quindi naturale che un punto del dominio di definizione della funzione sia approssimato combinando le approssimazioni fornite dai diversi modelli locali, ciascuna pesata dal grado di appartenenza al relativo insieme del punto stesso. Infine, nel caso del *Lazy*

Learning i modelli locali sono generati sequenzialmente, ciascuno per stimare il valore assunto dalla funzione in un determinato punto. Rispetto al tipo di interazione tra i modelli locali, il tratto caratteristico dei metodi *memory-based* è dunque l'*isolamento*: ciascun modello locale può quindi sfruttare tutti gli esempi disponibili che possono essere utili per fornire la stima più accurata.

Come gli esperimenti riportati hanno mostrato e come mostrano altri risultati presenti in letteratura e ottenuti con approcci diversi, a seconda delle caratteristiche del problema specifico in esame i diversi metodi possono consentire prestazioni più o meno buone. Il tipo di architettura da adottare per affrontare un problema di apprendimento dato, deve dunque essere selezionato a seconda della quantità degli esempi disponibili, della loro distribuzione rispetto al dominio della funzione, a seconde delle caratteristiche della funzione stessa, e sfruttando a fondo l'informazione disponibile *a priori*.

L'aspetto del lavoro svolto che merita di essere sottolineato per concludere questa tesi, e che a nostro parere maggiormente meriterà di essere approfondito in futuro, è il tentativo di definire, attraverso un'architettura multimodello, una metodologia integrata per affrontare il problema della ricostruzione di una funzione non lineare, a partire da un insieme di esempi. A nostro parere è possibile estendere l'uso dei metodi di validazione a tutto il processo di apprendimento, a partire dalla fase di osservazione, alla fase di trattamento dei dati, all'eventuale integrazione di modelli fisici, fino alla fase di identificazione strutturale. Una tale metodologia consentirebbe di andare oltre al paradigma *black-box*, fornirebbe strumenti per affrontare problemi di *experimental design* e di *feature selection*, e soprattutto consentirebbe all'analista sia di inserire conoscenza durante il processo di identificazione, sia di ottenere una maggior quantità di informazione riguardo al sistema in esame. Proprio questo ultimo aspetto sarebbe essenziale per uno sfruttamento più completo dei metodi di apprendimento applicati a problemi di controllo e di diagnosi.

Appendice A

A.1 Teoria della stima

Secondo l'assiomatizzazione della teoria della probabilità dovuta a Kolmogorov, un *esperimento casuale* è caratterizzato da tre elementi.

Il primo di questi è l'insieme $\mathcal{S}$ che prende il nome di insieme degli *esiti*. Gli elementi s di tale insieme sono ogni possibile risultato dell'esperimento casuale in questione.

Spesso non sono i singoli esiti ad essere significativi, ma si è interessati al verificarsi di certe combinazioni o raggruppamenti di esiti. Tali raggruppamenti prendono il nome di *eventi*. Il secondo elemento caratterizzante un esperimento casuale è quindi l'insieme $\mathcal{A}$ che prende il nome di insieme degli eventi. Questo insieme è tale per cui ogni evento $a \in \mathcal{A}$ è un sott'insieme di $\mathcal{S}$. Per ragioni tecniche, l'insieme $\mathcal{A}$ non può essere arbitrario ma deve avere la struttura di una σ-algebra definita sull'insieme $\mathcal{S}$; deve cioè essere un insieme non vuoto che soddisfi le seguenti condizioni:

$$\begin{aligned} &\text{se } a \in \mathcal{A} && \text{allora } \bar{a} \in \mathcal{A}, \\ &\text{e se } a_i \in \mathcal{A},\ i = 1, \ldots, l && \text{allora } \bigcup_{i=1}^{l} a_i \in \mathcal{A}, \end{aligned}$$

dove il numero l di eventi considerati può essere sia finito sia infinito. Da questa definizione segue che l'insieme $\mathcal{A}$ deve contenere l'insieme vuoto $\emptyset$ e l'intero insieme degli esiti $\mathcal{S}$. Affermeremo che si verifica l'evento a se si ha un esito $s \in a \subset \mathcal{S}$.

Il terzo ed ultimo elemento che caratterizza un esperimento casuale è una funzione $\mathcal{P}(\cdot) : \mathcal{A} \mapsto [0, 1]$ detta *probabilità*, che associa ad ogni evento un numero reale compreso tra zero e uno, estremi inclusi. La funzione $\mathcal{P}(\cdot)$ deve essere tale per cui $\mathcal{S}$ sia l'evento certo, cioè $\mathcal{P}(\mathcal{S}) = 1$ e, data una famiglia di eventi a due a due disgiunti sia:

$$\mathcal{P}\left(\bigcup_{i=1}^{l} a_i\right) = \sum_{i=1}^{l} \mathcal{P}(a_i)$$

Un modello di un esperimento casuale è quindi univocamente determinato qualora sia definito lo spazio probabilistico individuato dalla terna $(\mathcal{S}, \mathcal{A}, \mathcal{P})$.

Si consideri ora l'esperimento casuale composto da n prove semplici, ciascuna caratterizzata dallo spazio probabilistico $(\mathcal{S}, \mathcal{A}, \mathcal{P})$ e siano $s_1, s_2, \ldots, s_n$ gli esiti delle n prove. L'esperimento composto può essere descritto dal modello $(\mathcal{S}^n, \mathcal{A}^n, \mathcal{P}^n)$.

Gli elementi $s^n \in \mathcal{S}^n$ sono sequenze di esiti elementari $\{s_1, s_2, \ldots, s_n\}$, gli elementi $a^n \in \mathcal{A}^n$ sono sequenze dalla forma $\{a_1, a_2, \ldots, a_n\}$, composte di n eventi elementari[1] e $\mathcal{P}^n(\cdot)$ è una misura di probabilità tale per cui $\mathcal{P}^n(\cdot) : \mathcal{A}^n \mapsto [0, 1]$.

Se la sequenza $\{s_1, s_2, \ldots, s_n\}$ è tale per cui per ogni $\{a_1, a_2, \ldots, a_n\} \in \mathcal{A}^n$ risulta:

$$\mathcal{P}^n(s_1 \in a_1, s_2 \in a_2, \ldots, s_n \in a_n) = \prod_{i=1}^{n} \mathcal{P}(s_i \in a_i)$$

la sequenza in esame si dice essere composta da n prove *indipendenti.*

Capita spesso di dover considerare variabili i cui valori dipendono dall'esito di un esperimento casuale. Ad esempio per ciascun evento a, è possibile definire la *frequenza* dell'evento stesso in una sequenza di n prove indipendenti. Tale grandezza è una *variabile casuale* $v(s_1, s_2, \ldots, s_n)$ il cui valore è dato da:

$$v_n(a) = \frac{n_a}{n}$$

dove n_a è il numero di volte che l'evento a si è verificato nelle n prove considerate.

Più in generale, una variabile casuale definita sull'esperimento $(\mathcal{S}, \mathcal{A}, \mathcal{P})$, è una variabile $\mathbf{z}$ il cui valore dipende dall'esito s dell'esperimento in questione attraverso una opportuna funzione $\varphi(\cdot)$. Sia $\mathcal{Z}$ l'insieme dei valori che la variabile casuale $\mathbf{z}$ può assumere, potremo quindi scrivere: $\varphi(\cdot) : \mathcal{S} \mapsto \mathcal{Z}$.

Preso un sottoinsieme $D \subset \mathcal{Z}$, si desidera innanzitutto dar senso alla nozione di probabilità che il valore assunto dalla variabile casuale $\mathbf{z}$ appartenga a D, $prob\{\mathbf{z} \in D\}$. Si consideri l'insieme di esiti s per cui $\varphi(s)$ assume valori in D, identificheremo $prob\{\mathbf{z} \in D\}$ con la probabilità di tale insieme di esiti. Ma, per come abbiamo definito la funzione probabilità, ciò ha senso solo se l'insieme di esiti in questione è un evento, cioè un sottoinsieme di $\mathcal{S}$ appartenente alla famiglia $\mathcal{A}$. Formalmente ciò significa che la contro immagine dell'insieme D deve assere un evento:

$$\varphi^{-1}(D) \in \mathcal{A}$$

[1] Si noti che l'insieme $\mathcal{A}^n$ così definito risulta essere una σ-algebra.

Questa è la condizione fondamentale che deve soddisfare una variabile definita sullo spazio degli esiti $\mathcal{S}$ affinché possa essere considerata variabile casuale. Se tale condizione è soddisfatta, si potrà porre:

$$prob\{\mathbf{z} \in D\} = \mathcal{P}\left(\varphi^{-1}(D)\right)$$

Tra tutte le possibili variabili casuali, ci concentreremo su quelle reali, cioè su quelle variabili definite attraverso una funzione $\varphi(\cdot) : \mathcal{S} \mapsto \Re$. Si vuole dar senso al concetto di probabilità che una variabile casuale appartenga ad un dato intervallo dell'asse reale, diciamo $[a, b]$:

$$prob\{\mathbf{z} \in [a, b]\} = \mathcal{P}\left(\varphi^{-1}\left([a, b]\right)\right)$$

Poiché la funzione $\mathcal{P}(\cdot)$ è stata definita sullo spazio degli eventi, è necessario che, qualunque sia l'intervallo $[a, b]$, la controimmagine $\varphi^{-1}\left([a, b]\right)$ sia un evento della σ-algebra $\mathcal{A}$ definita su $\mathcal{S}$. Fortunatamente non è necessario verificare che questa condizione sia soddisfatta per tutti gli intervalli possibili, si dimostra infatti che condizione necessaria e sufficiente affinché sia

$$\varphi^{-1}\left([a, b]\right) \in \mathcal{A} \qquad\qquad \forall a, b \in \Re$$

è che:

$$\varphi^{-1}\left([-\infty, q]\right) \in \mathcal{A} \qquad\qquad \forall q \in \Re$$

Data una variabile casuale $\mathbf{z}$, si consideri ora la funzione:

$$P_{\mathbf{z}}(q) = prob\{\mathbf{z} \leq q\} = \mathcal{P}\left(\varphi^{-1}\left([-\infty, q]\right)\right)$$

Tale funzione prende il nome di *distribuzione di probabilità*. La conoscenza della funzione $P_{\mathbf{z}}(\cdot)$ fornisce una descrizione probabilistica completa della variabile casuale $\mathbf{z}$. Un'altra funzione di notevole importanza è la *densità di probabilità* che è definita come derivata generalizzata di $P_{\mathbf{z}}(\cdot)$:

$$p_{\mathbf{z}}(q) = \frac{\mathrm{d}P_{\mathbf{z}}(q)}{\mathrm{d}q}$$

L'interpretazione intuitiva che si può dare della densità di probabilità è la seguente: si consideri un punto $q \in \Re$, e sia $\mathrm{d}q$ l'intervallo compreso tra q e $q+\mathrm{d}q$; l'area sottesa dalla curva $p_{\mathbf{z}}(\cdot)$ in questo intervallo è pari alla probabilità che la variabile casuale $\mathbf{z}$ assuma valori compresi tra q e $q + \mathrm{d}q$.

D'ora in avanti, per semplificare la notazione, scriveremo semplicemente $P(\mathbf{z})$ e $p(\mathbf{z})$ al posto di $P_{\mathbf{z}}(q)$ e $p_{\mathbf{z}}(q)$ rispettivamente; ritorneremo alla notazione estesa solo laddove questa servirà ad evitare ambiguità o risulterà comunque più chiara.

E' interessante definire alcuni parametri notevoli che caratterizzano una densità di probabilità.

Valore atteso: Il valore atteso di una variabile casuale è per definizione dato dall'integrale:

$$E[\mathbf{z}] = \int z p(z)\, \mathrm{d}z$$

Se $p(\mathbf{z})$ è simmetrica attorno ad un valore $\bar{z}$ vale la notevole proprietà $E[\mathbf{z}] = \bar{z}$.

Varianza: La varianza di una variabile casuale $\mathbf{z}$ è definita dalla relazione:

$$Var[\mathbf{z}] = \int (z - E[\mathbf{z}])^2\, p(z)\, \mathrm{d}z$$

A.2 Levenberg-Marquardt

Presentiamo di seguito le derivate del modello neuro-fuzzy rispetto alle varie componenti del vettore η rispetto al quale il modello è non lineare. Queste sono le derivate necessarie nell'implementazione dell'algoritmo Levenberg-Marquardt. Se le funzioni $\beta_j(x, \eta_j)$ sono ottenute dal prodotto di d funzioni gaussiane μ_{jl}, dove d è la dimensione dello spazio di ingresso, risulta:

$$\beta_j(x, \eta_j) = e^{-(x-c_j)^T \Sigma_j{}^{-1}(x-c_j)},$$

dove gli elementi diagonali della matrice Σ_j sono pari a σ_{jl}^2, mentre gli elementi fuori diagonale sono nulli. E dove $\eta_j = \left[c_j^T, \mathrm{diag}(\Sigma_j)^T\right]^T$. Posto $b_{jl} = \sigma_{jl}^2$, risulta:

$$\frac{\partial \mu_{jl}}{\partial c_{jl}} = \frac{2\left(x^{(l)} - c_{jl}\right)}{b_{jl}} e^{-\frac{\left(x^{(l)} - c_{jl}\right)^2}{b_{jl}}},$$

dove $x^{(l)}$ è la componente l-esima del vettore c_j. Inoltre:

$$\frac{\partial \mu_{jl}}{\partial b_{jl}} = \left(\frac{x^{(l)} - c_{jl}}{b_{jl}}\right)^2 e^{-\frac{\left(x^{(l)} - c_{jl}\right)^2}{b_{jl}}}$$

Se invece le funzioni $\beta_j(x, \eta_j)$ sono otenute dal prodotto di d funzioni triangolari, ciascuna dalla forma:

$$\mu_{jl}\left(x^{(l)}, \eta_{jl}\right) = \max\left(0, 1 - \frac{|x^{(l)} - c_{jl}|}{2\sigma_{jl}}\right),$$

risulta, posto $b_{jl} = \sigma_{jl}$:

$$\frac{\partial \mu_{jl}}{\partial c_{jl}} = \frac{\mathrm{sign}(x^{(l)} - c_{jl})}{2b_{jl}} \,\mathrm{grad}\left[\max\left(0, 1 - \frac{|x^{(l)} - c_{jl}|}{2b_{jl}}\right)\right],$$

e anche

$$\frac{\partial \mu_{jl}}{\partial \sigma_{jl}^2} = \frac{|x^{(l)} - c_{jl}|}{2b_{jl}^2} \,\mathrm{grad}\left[\max\left(0, 1 - \frac{|x^{(l)} - c_{jl}|}{2b_{jl}}\right)\right],$$

dove con $\mathrm{sign}(\cdot)$ e con $\mathrm{grad}(\cdot)$ si sono indicate rispettivamente le funzioni *segno* e *gradino unitario*.

In entrambi i casi, sia che si considerino funzioni triangolari, sia che si considerino funzioni gaussiane, si può scrivere:

$$\frac{\partial \beta_j}{\partial \mu_{jl}} = \frac{\beta_j}{\mu_{jl}}.$$

Le derivate della j-esima funzione di attivazione β_j rispetto al centro e alla base della stessa sono date da:

$$\frac{\partial \beta_j}{\partial c_{jl}} = \frac{\partial \beta_j}{\partial \mu_{jl}} \frac{\partial \mu_{jl}}{\partial c_{jl}}$$

e da

$$\frac{\partial \beta_j}{\partial b_{jl}} = \frac{\partial \beta_j}{\partial \mu_{jl}} \frac{\partial \mu_{jl}}{\partial b_{jl}}$$

Consideriamo in principio il caso in cui i contributi dei modelli locali $f_j(x, \theta_j)$ sono combinati semplicemente attraverso i valori assunti dalle funzioni di appartenenza:

$$f(x, \alpha) = \sum_{j=1}^{m} \beta_j(x, \eta_j) f_j(x, \theta_j).$$

In tal caso risulta semplicemente:

$$\frac{\partial f(x, \alpha)}{\partial \beta_j(x, \eta_j)} = f_j(x, \theta_j).$$

Se invece il modello è ottenuto pesando i vari modelli locali con dei coefficienti ottenuti normalizzando i valori assunti dalle funzioni di appartenenza:

$$f(x, \alpha) = \frac{\sum_{j=1}^{m} \beta_j(x, \eta_j) f_j(x, \theta_j)}{\sum_{j=1}^{m} \beta_j(x, \eta_j)},$$

risulta:

$$\frac{\partial f(x, \alpha)}{\partial \beta_j(x, \eta_j)} = \frac{f_j(x, \theta_j) \sum_{k=1}^{m} \beta_k(x, \eta_k) - \sum_{j=1}^{m} \beta_j(x, \eta_j) f_j(x, \theta_j)}{\left(\sum_{k=1}^{m} \beta_k(x, \eta_k)\right)^2}.$$

Risulta quindi:

$$\frac{\partial f(x, \alpha)}{\partial c_{jl}} = \frac{\partial f(x, \alpha)}{\partial \beta_j(x, \eta_j)} \frac{\partial \beta_j(x, \eta_j)}{\partial c_{jl}},$$

e

$$\frac{\partial f(x, \alpha)}{\partial b_{jl}} = \frac{\partial f(x, \alpha)}{\partial \beta_j(x, \eta_j)} \frac{\partial \beta_j(x, \eta_j)}{\partial b_{jl}}.$$

A.3 Algoritmo di scaling

L'algoritmo di scaling utilizzato (Masters, 1995) é definito dalla seguente:

$$A = r\left(\frac{x - \hat{\mu}}{\hat{\sigma}} - V_{min}\right) + A_{min}$$

dove $\hat{\mu}$ e $\hat{\sigma}$ sono rispettivamente la media campionaria e la deviazione standard campionaria della grandezza scalare x in esame:

$$\hat{\mu} = \frac{1}{n}\sum_{i=1}^{n} x_i,$$

$$\hat{\sigma} = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \hat{\mu})^2}.$$

Il coefficiente r è invece dato da:

$$r = \frac{A_{max} - A_{min}}{V_{max} - V_{min}}.$$

L'algoritmo implementato prevede che A_{min} e A_{max} siano rispettivamente -1 e $+1$ e che V_{min} abbia valore -1.28 e V_{max} valore $+1.28$.

Se la variabile x è approssimativamente distribuita normalmente, per i valori di V_{min} e V_{max} scelti A è tale che con probabilità pari a .20 assume valori esterni all'intervallo individuato da A_{min} e A_{max}

Bibliografia

C. G. Atkeson, A. W. Moore, & S. Schaal. 1996. Locally Weighted Learning. *Artificial Intelligence Review*, submitted.

R. Babuška. 1997. *Fuzzy Modeling and Identification.* Ph.D. thesis, Technische Universiteit Delft, Delft, NL.

H. Bandemer, & S. Gottwald. 1995. *Fuzzy Sets, Fuzzy Logic, Fuzzy Methods with Applications.* New York, NY: John Wiley and Sons.

R. E. Bellman. 1961. *Adaptive Control Processes.* Princeton, NJ: Princeton University Press.

H. Bersini, & G. Bontempi. 1996. *Now Comes The Time to Defuzzify Neuro-Fuzzy Models.* Tech. rept. TR/IRIDIA/96-15. Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, ULB, Bruxelles, B. Apparirà nel numero speciale di Fuzzy Sets and Systems. Fuzzy Sets: Where Do we Stand? Where Do we Go?

H. Bersini, M. Birattari, & G. Bontempi. 1997a. *Adaptive memory based regression methods.* Tech. rept. TR/IRIDIA/97-13. Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, Bruxelles, B.

H. Bersini, A. Duchateau, & N. Bradshaw. 1997b. Using Incremental Learning Algorithms in the Search for Minimal and Effective Fuzzy Models. *In: Proceedings of IEEE.*

J. C. Bezdek. 1981. *Pattern Recognition with Fuzzy Objective Function Algorithms.* New York, NY: Plenum Press.

S. A. Billings, & W. S. G. Voon. 1987. Piecewise linear identification of nonlinear systems. *Internation Journal of Control*, **46**, 215–235.

C. M. Bishop. 1995. *Neural Networks for Pattern Recognition.* Oxford, UK: Clarendon Press.

S. Bittanti. 1992a. *Identificazione dei Modelli e Controllo Adattativo.* Bologna: Pitagora Editrice.

S. Bittanti. 1992b. *Teoria della Predizione e del Filtraggio.* Bologna: Pitagora Editrice.

G. Bontempi, & Hugues Bersini. 1997. Identification of a sensor model with hybrid neuro-fuzzy methods. *Pages 325–328 of:* A. B. Bulsari, & S. Kallio (eds), *Neural Networks in Engineering Systems.*

L. Bottou, & V. N. Vapnik. 1992. Local Learning Algorithms. *Neural Computation*, **4**(6), 888–900.

N. Bradshaw. 1996. *An Introduction to the EM Algorithm and its Applications.* Tech. rept. TR/IRIDIA/96-21. Institut de Recherches Interdisciplinaires et de Développements en Intelligence Artificielle, ULB, Bruxelles, B.

L. Breiman, J. H. Friedman, R. A. Olshen, & C. J. Stone. 1984. *Classification and Regression Trees.* Belmont, CA: Waldsworth.

W. S. Cleveland. 1979. Robust Locally Weighted Regression and Smoothing Scatterplots. *J. Amer. Statist. Assn*, **74**, 829–836.

W. S. Cleveland, & C. Loader. 1996. *Smoothing by Local Regression: Principles and Methods.* Tech. rept. AT&T BELL Laboratories.

W. S. Cleveland, T. Hastie, & C. Loader. 1995. *Adaptive Local Regression by Automatic Selection of the Polynomial Mixing Degree.* Tech. rept. AT&T BELL Laboratories.

A. Colorni. 1984. *Ricerca Operativa.* Milano: CLUP.

A. P. Dempster, N. M. Larid, & D. B. Rubin. 1977. Maximun likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, **39**, 1–38.

D. Dubois, & H. Prade. 1982. A Unifying View of Comparison Indices in a Fuzzy Set-Theoretic Framework. *Pages 3–13 of:* R. R. Yager (ed), *Fuzzy Set and Possibility Theory.* New York, NY: Pergamon Press.

R. O. Duda, & P. E. Hart. 1973. *Pattern Classification and Scene Analysis.* New York, NY: John Wiley and Sons.

B. Efron, & R. J. Tsibirani. 1994. *An Introduction to the Bootstrap.* New York, NY: Chapman and Hall.

B. Efron, & R. J. Tsibirani. 1995. *Cross-Validation and the Bootstrap: Estimating the Error Rate of a Prediction Rule.* Tech. rept. Tecnical Report 176. Dept. of Statistics, Stanford, CA.

U. Fayyad, G. Piatetsky-Shapiro, & P. Smyth. 1996. The KDD Process for Extracting Useful Knowledge from Volumes of Data. *Communications of the ACM*, **39**(11), 27–34.

J. H. Friedman. 1991. Multivariate adaptive regression splines (with discussion). *The Annals of Statistics*, **19**, 1–141.

S. Geman, E. Bienenstock, & R. Doursat. 1992. Neural Networks and the Bias/Variance Dilemma. *Neural Computation*, **4**(1), 1–58.

R. Giles. 1993. The Concept of Grade of Membership. *In:* D. Dubois, H. Prade, & R. R. Yager (eds), *Readings in Fuzzy Sets for Intelligent Systems.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

D. E. Golberg. 1989. *Genetic Algorithm in Search, Optimization & Machine Learning.* Reading, MA: Addison-Wesley.

K. J. Hunt, R. Haas, & R. Murray-Smith. 1994. On the functional equivalence of fuzzy inference systems and spline based-networks. *In pubblicazione.*

R. A. Jacobs, M. I. Jordan, S. J. Nowlan, & G. E. Hinton. 1991a. Adaptive Mixtures of Local Experts. *Neural Computation*, **3**(1), 79–87.

R. A. Jacobs, M. I. Jordan, & A. G. Barto. 1991b. Task decomposition through competition in a modular connectionist architecture: The what and where vision tasks. *Cognitive Science*, **15**(2), 219–250.

J.-S. R. Jang. 1993. ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Transactions on Systems, Man and Cybernetics*, **23**(3), 665–685.

J.-S. R. Jang, & C.-T. Sun. 1993. Functional Equivalence Between Radial Basis Function Networks and Fuzzy Inference Systems. *IEEE Transactions on Neural Networks*, **4**(1), 156–159.

T. A. Johansen, & B. A. Foss. 1993. Constructing NARMAX models using ARMAX models. *International Journal of Control*, **58**, 1125–1153.

M. I. Jordan, & R. A. Jacobs. 1994. Hierarchical Mixtures of Experts and the EM Algorithm. *Neural Computation*, **6**, 181–214.

M. I. Jordan, & R. A. Jacobs. 1995. Modular and hierarchical learning system. *In:* M. Arbib (ed), *The handbook of Brain Theory and Neural Networks.* Cambridge, MA: MIT Press.

M. I. Jordan, & L. Xu. 1993. *Convergence properties of the EM approach to learning in mixture-of-experts architectures.* Tech. rept. 9302. Dept. of Brain and Cognitive Science, MIT, Cambridge, MA.

T. Kavli. 1993. ASMOD—An algorithm for adaptive spline modeling of observation data. *Int. J. Control*, **58**, 947–967.

R. Kohavi. 1995. A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *Pages 1137–1145 of:* C. S. Mellish (ed), *IJCAI-95*, vol. 2. San Mateo, CA: Morgan Kaufmann Publishers.

S. Lawrence, A. C. Tsoi, & A. D. Back. 1996. Function Approximation with Neural Networks and Local Methods: Bias, Variance and Smoothness. *Pages 16–21 of:* Peter Bartlett, Anthony Burkitt, & Robert Williamson (eds), *Australian Conference on Neural Networks.* Sidney, AU: Australian National University.

K. Levenberg. 1944. A method for the solution of certain non-linear problems in least squares. *Quarterly Journal of Applied Mathematics*, **II**(2).

L. Ljung. 1993. *Perspective on the Process of Identification.* Tech. rept. LiTH-ISY-R-1531. Department of Electrical Engineering, Linköping University, Linköping, S.

M.C. Mackey, & L. Glass. 1977. Oscillation and chaos in physiological control systems. *Science*, **197**, 287–289.

C. Mallows. 1974. Discussion of a paper of Beaton and Tukey. *Technometrics*, **16**.

E. H. Mamdani. 1977. Application of fuzzy logic to approximate reasoning using linguistic systems. *Fuzzy Sets and Systems*, **26**, 1182–1191.

D. W. Marquardt. 1963. An algorithm for least-squares estimaton of non-linear parameters. *Journal of the Society of Industrial And Applied Mathematics*, **11**(2).

T. Masters. 1995. *Practical Neural Network Recipes in C++.* New York, NY: Academic Press.

D. C. Montgomery, & E. A. Peck. 1992. *Introduction to Linear Analysis.* Second edition edn. New York, NY: John Wiley and Sons.

R. Murray-Smith. 1994. *A local model network approach to nonlinear modelling.* Ph.D. thesis, Department of Computer Science, University of Strathclyde, Strathclyde, UK.

R. H. Myers. 1994. *Classical and Modern Regression with Applications.* second edition edn. Boston, MA: PWS-KENT Publishing Company.

A. Papoulis. 1991. *Probability, Random Variables, and Stochastic Processes.* Third edition edn. Electrical & Electronic Engineering Series. New York, NY: McGraw-Hill International Editions.

J. Platt. 1991. Learning by Combining Memorization and Gradient Descent. *In: Advances in Neural Information Processing System*, vol. 3.

T. Poggio, & F. Girosi. 1990. A Theory of Networks for Approximation and Learning. *Proceedings of the IEEE*, **78**(9).

M. J. D. Powell. 1987. Radial Basis Function approximations to polynomials. *Pages 223–241 of: 12th Biennal Numerical Analysis Conference.*

W. H. Press, S. A. Teukolsky, & W. T. Vetterling B. P. Flannery. 1995. *Numerical Recipies in C.* Cambridge, UK: Cambridge University Press.

M. B. Priestley. 1988. *Non-linear and Non-stationary Time Series Analysis.* London, UK: Academic Press.

R. Quinlan. 1993. *C4.5. Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

D. E. Rumelhart, & J. L. McClelland (eds). 1986. *Parallel Disreibuted Processing:Explorations in the Microstructure of Cognition.* Cambridge, MA: MIT Press.

T. Sauer. 1993. Time Series Prediction by using delay coordinate embedding. *In:* A. S. Weigend, & N. A. Gershenfeld (eds), *Time Series Prediction: forecasting the future and understanding the past.* Readings, MA: Addison Wesley.

J. S. Shamma, & M. Athanas. 1990. Analysis of gain sheduled control for nonlinear plants. *IEEE Transactions on Automatic Control*, **35**, 898–907.

R. Shorten, & R. Murray-Smith. 1994. On normalising radial basis function networks. *In: Irish Neural Networks Conference.* University College Dublin, Dublin, EI.

J. Sjöberg, H. Hajalmarsson, & L. Ljung. 1994. *Neural Networks in System Identification.* Tech. rept. Department of Electrical Engineering, Linköping University, Linköping, S.

A. Skeppstedt, L. Ljung, & M. Millnert. 1992. Construction of composite models from observed data. *International Journal of Control*, **55**(1), 141–152.

T. Takagi, & M. Sugeno. 1985. Fuzzy Identification of System and its Applications to Modeling and Control. *IEEE Transactions on Systems, Man and Cybernetics*, **15**(1), 116–132.

H. Tong. 1990. *Non-linear Time Series.* New York, NY: Oxford University Press.

V. N. Vapnik. 1995. *The Nature of Statistical Learning Theory.* New York, NY: Springer Verlag.

V. N. Vapnik, & A. J. Chervonenkis. 1991. The necessary and sufficient conditions for consistency of the method of empirical risk minimization. *Pattern Recognition and Image Analysis*, **1**(3), pp. 284–305.

S. M. Weiss, & C. A. Kulikowski. 1991. *Computer Systems That Learn.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

L. Xu, M. I. Jordan, & G. E. Hinton. 1995. An Alternative Model for Mixtures of Experts. *Pages 633–640 of:* G. Tesauro, D. Touretzky, & T. Leen (eds), *Advances in Neural Information Processing Systems*, vol. 7. The MIT Press.

L. A. Zadeh. 1965. Fuzzy Sets. *Information and Control*, **8**, 338–353.

L. A. Zadeh. 1973. Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man and Cybernetics*, **3**, 28–44.

L. A. Zadeh. 1989. Knowledge Representation in Fuzzy Logic. *IEEE Transactions on Knowledge and Data Engineering*, **1**(1), 89–100.

L. A. Zadeh. 1993. Possibility Theory and Soft Data Analysis. *In:* D. Dubois, H. Prade, & R. R. Yager (eds), *Readings in Fuzzy Sets for Intelligent Systems.* San Mateo, CA: Morgan Kaufmann Publishers, Inc.

R. Żbikowski, K. J. Hunt, A. Dzieliński, R. Murray-Smith, & P. J. Gawthrop. 1994. *A Review of Advances in Neural Adaptive Control Systems.* Daimler-Benz AG – University of Glasgow. ESPRIT III PROJECT 8039: Neural Adaptive Control Technology.