UNIVERSITÀ DI ROMA TRE

Facoltà di Economia Federico Caffè

Dottorato in Metodi Statistici per l'Economia e l'Impresa

XXI Ciclo

Metodi per l'imputazione dei dati mancanti in dataset di variabili nominali

Roberto Di Manno

Supervisori

Prof.Renato Coppi (Università di Roma La Sapienza)

Prof.ssa Paola Vicard (Università di Roma Tre)

Coordinatrice

Prof.ssa Julia Mortera (Università di Roma Tre)

Indice

1 Introduzione **1**

1.1 La qualità dei dati 1

1.2 L'errore totale 2

1.3 Le mancate risposte 4

1.3.1 Le strutture delle mancate risposte 5

1.3.2 I meccanismi generatori delle mancate risposte 7

1.3.3 Il trattamento delle mancate risposte 8

1.3.3.1 Inclusione dell'attributo mancanta risposta tra quelli del dominio 9

1.3.3.2 Analisi basate sulle unità completamente osservate 9

1.3.3.3 Riponderazione delle unità completamente osservate 10

1.3.3.4 Modelli probabilistici per dataset incompleti: l'algoritmo EM 10

1.3.3.5 Procedure di imputazione 13

1.4 Contenuto ed obiettivi del presente lavoro 18

2 La cluster analysis **20**

2.1 Definizione di cluster analysis 20

2.2 Tecniche di classificazione fuzzy per dati categoriali incompleti . 22

2.3 Metodi non probabilistici 24

2.3.1 Le fuzzy conceptual k-means 25

2.3.2 Le fuzzy k-modes e le fuzzy k-populations 28

2.4 Metodi probabilistici 32

2.4.1 La grade of membership analysis 34

2.4.2 Le fuzzy k-partitions 38

3 Le reti bayesiane **43**

3.1 Definizione di rete bayesiana 43

3.2 Structural learning per reti bayesiane 47

3.2.1 Algoritmi greedy search 49
3.2.2 Algoritmi genetici 51
3.2.3 Structural learning per dataset incompleti 53
3.3 Imputazione dei valori mancanti tramite reti Bayesiane 55
3.3.1 Imputazione tramite nodi genitori 55
3.3.2 Imputazione tramite Markov blanket 56

4 Applicazioni 57
4.1 Criteri per la valutazione delle imputazioni 58
4.2 Applicazioni dei metodi nello spazio delle osservazioni 59
4.2.1 Analisi del dataset "Soybean Disease" 61
4.2.2 Analisi del dataset "Zoo" 69
4.3 Applicazioni dei metodi nello spazio delle variabili 78
4.3.1 Analisi del dataset generato dalla rete "Asia" 81
4.3.2 Analisi del dataset generato dalla rete "golf" 89
4.4 Confronto tra cluster analysis e reti bayesiane 95
4.4.1 Analisi del dataset Contraceptive Method Choice 95
4.5 Conclusioni 102

A Cenni sugli insiemi fuzzy 104

B Programmi in R 109
B.1 L'ambiente R 109
B.2 Programma per l'imputazione di dataset di variabili categoriali . 109
B.2.1 Modulo per l'imputazione tramite fuzzy cluster analysis . 114
B.2.1.1 Funzioni di supporto al programma "FC Impute.R" 115
B.2.2 Modulo per l'imputazione tramite reti bayesiane 125
B.2.2.1 Funzioni di supporto al programma "BN Impute.R" 128
B.2.3 Modulo per l'imputazione tramite algoritmo EM 144
B.2.4 Funzioni di supporto al programma di imputazione . . . 145
B.3 Programma per la generazione di dati mancanti 151

Sommario

Il concetto di qualità nel contesto di un'indagine statistica campionaria assume il significato di precisione dell'informazione rilevata e può essere espressa come funzione inversa dell'errore statistico [65]. Quest'ultimo, consta di due componenti: l'errore campionario e l'errore non campionario. A sua volta, la seconda componente si divide in tre diverse tipologie: l'errore di copertura, l'errore di misurazione e l'errore di mancata risposta.

Il presente lavoro focalizza l'attenzione sulle mancate risposte. In particolare, nel seguito della trattazione vengono presentate delle procedure il cui obiettivo è quello di permettere la ricostruzione di una matrice di osservazioni sulla quale siano state rilevate informazioni parzialmente incomplete, attraverso tecniche di imputazione di dati mancanti.

La matrice dei dati viene analizzata con due diversi approcci: il primo considera il legame esistente tra le unità del campione, il secondo valuta le relazioni di dipendenza probabilistica tra le variabili rilevate. In entrambe le situazioni, sono introdotte delle innovazioni rispetto alla letteratura esistente in materia di trattamento di mancate risposte parziali (ad esempio, [6, 29, 37, 49]).

Metodologie che si basano sui legami tra le osservazioni per individuare gruppi omogenei di unità, quali le tecniche di cluster analysis [1, 46] sono già state usate per l'imputazione di matrici di variabili continue parzialmente osservate. A tale scopo, sono stati utilizzati modelli con formalizzazione non probabilistica, come nel caso delle fuzzy c-medie [43, 61] ed altri con formalizzazione probabilistica, come i modelli di mistura gaussiani [25]. Lo stesso approccio viene ora proposto incentrando l'attenzione sulle variabili nominali. La novità consiste nel permettere l'analisi di dataset incompleti ad alcuni metodi di cluster analysis fuzzy (sia stocastici [87, 89] che non stocastici [45, 51, 74]), originariamente pensati per matrici di variabili categoriali complete, in modo da consentire loro l'imputazione delle mancate risposte.

Tra i metodi finalizzati all'individuazione delle relazioni di dipendenza probabilistica tra variabili, le reti bayesiane [71] sono state utilizzate anche a scopo di imputazione [24, 26, 23]. Tuttavia, laddove la struttura dei legami tra le

variabili non è conosciuta a priori, diviene necessario apprenderla dai dati di cui si è in possesso. Il successo nell'utilizzo delle reti bayesiane a scopi di imputazione dipende essenzialmente dalla ricerca di una struttura quanto più verosimile a quella che effettivamente lega le informazioni tra di loro. L'algoritmo PC [82], sebbene non nasca per l'apprendimento di strutture in dataset incompleti, è stato fino ad oggi il metodo di structural learning utilizzato prima della fase di imputazione. Si è deciso allora di considerare delle tecniche alternative. La prima innovazione consiste nell'uso di algoritmi di tipo greedy search [44], in precedenza già applicati su dataset incompleti [31], ma non a scopi di imputazione. La seconda innovazione, riguarda l'uso degli algoritmi genetici [35] per lo structural learning. La tecnica proposta, pur prendendo spunto da analisi svolte sia su dataset completamente osservati [53, 90], che su dataset incompleti [91], contiene in sé degli aspetti originali.

Tutte le metodologie descritte sono implementate in linguaggio **R** [73] e sono testate in un primo momento separatamente, su dataset ad hoc. In questo modo, diviene possibile misurarne l'efficacia in termini di imputazione, ma anche la capacità di classificazione per la cluster analysis, e la capacità di ricostruzione della struttura nel caso per le reti bayesiane. Successivamente, tutte le tecniche sono applicate congiuntamente su un'indagine reale, così da poterle valutare in modo comparativo tra loro e rispetto all'algoritmo EM [79], utilizzato come metodo di benchmark.

Capitolo 1

Introduzione

1.1 La qualità dei dati

La qualità delle analisi che si possono effettuare su un insieme di unità su cui sono state osservate delle variabili, dipende fondamentalmente dalla qualità dei dati che si hanno a disposizione. Sul concetto di qualità del dato non c'è una definizione standard, tuttavia è comune accordo quello di individuare nei seguenti concetti, gli indicatori rilevanti per la definizione della qualità [12, 32, 83]:

- *rilevanza*: è il giudizio qualitativo sul valore offerto dal dato;
- *accuratezza*: è il grado con cui il dato stima o descrive correttamente le quantità o le caratteristiche che l'analisi statistica ha l'obiettivo di misurare;
- *tempestività*: riflette la quantità di tempo che intercorre tra la disponibilità dell'informazione e il momento in cui si è verificato l'evento che descrive;
- *accessibilità:* corrisponde alla semplicità per l'utente di reperire, aquisire e comprendere l'informazione disponibile in relazione alle proprie finalità*;*
- *chiarezza:* corrisponde alla semplicità con cui l'utente può comprendere ed utilizzare propriamente l'informazione;
- *comparabilità:* è la possibilità di paragonare nel tempo e nello spazio le statistiche riguardanti il fenomeno di interesse*;*
- *coerenza:* corrisponde alla possibilità, di poter combinare insieme diverse informazioni per generare induzioni complesse senza pervenire a statistiche contraddittorie tra loro*;*

- *completezza:* è la capacità dei dati rilevati di fornire un quadro informativo soddisfacente rispetto agli obiettivi prefissati.

L'accuratezza viene misurata mediante la distanza tra il valore vero e la stima ottenuta. Le cause all'origine di tale distanza vengono definite *fonti dell'errore*, mentre una sua misura è data dall'*errore totale*.

1.2 L'errore totale

Quando la qualità assume il significato di precisione (accuratezza), quest'ultima può essere espressa come funzione inversa dell'errore statistico [65]. Supposto che un'indagine sia ripetibile secondo le medesime condizioni generali, una prima scomposizione dell'errore può essere fatta in:

- *errori variabili:* sono distribuiti casualmente, hanno media nulla e variano in ciascuna delle ipotetiche ripetizioni dell'indagine;
- *distorsioni*: sono il risultato di fattori sistematici, dipendono da condizioni generali, sono costanti in tutte le ripetizioni ed hanno uno specifico segno.

Una seconda scomposizione dell'errore può essere data da:

- *errore campionario:* è la differenza tra il valore vero del parametro di interesse e la sua stima campionaria. Esso è dovuto a:
  - *errore variabile di campionamento*, generato dall'uso di una tecnica campionaria. La stima differisce per effetto del caso dal valore vero della popolazione;
  - *distorsione dello stimatore*, dovuta all'adozione di uno stimatore non corretto, ovvero la cui media nell'universo dei campioni non coincide con il parametro da stimare;
- *errore non campionario:* è provocato da tutte le possibili imprecisioni ed inaccuratezze commesse o subite durante le fasi di un'indagine statistica o amministrativa e si ripercuote al pari dell'errore campionario, sia sulla distorsione che sulla varianza, cioè sulle due componenti dell'errore totale. Una classificazione degli errori non campionari normalmente accettata in ambito scientifico [38] è la seguente:
  - *errori di copertura*: dovuti all'imperfezioni nella corrispondenza tra la lista utilizzata per selezionare le unità statistiche e la popolazione

oggetto di indagine. Possono essere di *sovracopertura* quando si includono unità non appartenenti alla popolazione di interesse o di *sottocopertura* se non si riesce ad includere unità appartenenti alla popolazione di interesse;

– *errori di mancata risposta*: quando sull'unità considerata è impossibile reperire informazioni o al più è possibile rilevarne soltanto alcune;

– *errori di misurazione*: sono dati dalla differenza tra il valore vero della caratteristica da misurare su una certa unità statistica e quello osservato nella rilevazione.

Considerato un campione di I unità statistiche estratte da una popolazione con numerosità pari a $\mathcal{I}$, se si definisce come:

- $y = f(y_1, y_2, ..., y_I)$ una qualsiasi funzione di sintesi relativa ai valori osservati y_i sulle I unità campionarie;
- $y^* = f(Y_1, Y_2, ..., Y_I)$ la stessa funzione di sintesi relativa ai valori veri Y_i sulle I unità campionarie;
- $Y = f(Y_1, Y_2, ..., Y_{\mathcal{I}})$ la funzione di sintesi relativa ai valori veri Y_i sul totale delle $\mathcal{I}$ unità della popolazione;

è possibile definire l'errore statistico nel seguente modo:

$$y - Y = [y - y^*] + [y^* - E(y^*)] + [E(y^*) - Y]$$

dove:

- $[y - y^*]$ è l'errore non campionario;
- $[y^* - E(y^*)]$ è l'errore variabile campionario;
- $[E(y^*) - Y]$ è la distorsione dello stimatore.

A sua volta, ipotizzando la ripetizione dell'indagine sotto le stesse condizioni generali, l'errore non campionario può essere scomposto nell'errore variabile (da indagine ad indagine) non campionario v e nella distorsione non campionaria b:

$$y - y^* = [y - E(y)] + [E(y) - y^*] = v + b$$

dove $E(y)$ è il valore medio, calcolato su tutte le ipotetiche ripetizioni dell'indagine, della stima ottenuta dai valori osservati, avendo fissato le unità del campione.

Assumendo che v sia una variabile aleatoria con media zero ($E(v) = 0$) e varianza data $Var(v) = E[v - E(v)]^2 = E(v)^2 = VNC$, mentre b è costante ($E(b) = B$), si può stimare l'errore totale per lo stimatore y come media quadratica della differenza $[y - Y]$:

$$\begin{aligned} MSE(y) &= E(y - Y)^2 \\ &= E[v + b + y^* - E(y^*) + E(y^*) - Y]^2 \\ &= \left\{E(v)^2 + E[y^* - E(y^*)]^2\right\} + \{B + [E(y^*) - Y]\}^2 + 2cov(v, y^*) \\ &= \{VNC + VC\} + \{B + [E(y^*) - Y]\}^2 + 2cov(v, y^*) \end{aligned}$$

con $VC = E[y^* - E(y^*)]^2$, varianza campionaria. Da tale risultato si evince che gli errori non campionari incidono sia sulla parte variabile dell'errore totale, che sulla componente legata alla distorsione. Intervenire per limitare l'effetto della mancata copertura, delle mancate risposte o degli errori di misurazione, ha quindi lo scopo di ridurre l'entita dell'errore totale. Nella prossima sezione verrà focalizzata l'attenzione sugli errori dovuti alle mancate risposte, in particolare verranno descritte le diverse configurazioni che esse possono assumere, i meccanismi che le generano e le soluzioni che si possono adottare per poter effettuare delle analisi in presenza di dati incompleti.

1.3 Le mancate risposte

Generalmente, le metodologie statistiche standard sono sviluppate per analizzare dataset in forma rettangolare. Le righe di tali matrici rappresentano le osservazioni (o unità, oppure casi) mentre le colonne sono le variabili misurate su ciascuna osservazione. I valori osservati saranno numeri reali, nel caso di variabili continue (quantitative), oppure rappresenteranno categorie di risposta, nel caso di variabili discrete (qualitative, categoriali, nominali). Si può parlare di mancata risposta ogni qualvolta il valore di una variabile per una certa unità è [6]:

- *mancante*, se non è stato possibile osservarlo;
- *errato*, se non è quello che nella realtà non è posseduto dall'unità considerata. In particolare, il valore errato può essere ricondotto a tre tipologie:
  - *fuori dominio*, se è esterno all'intervallo dei valori ammissibili (errore tipico per le variabili categoriali);

- *anomalo (outlier)*, se la risposta fornita dall'unità in esame si discosta in modo significativo da quella fornita da tutte le altre unità (errore tipico per le variabili quantitative);
- *incompatibile*, se c'è contraddizione con i valori delle altre variabili rilevati sulla stessa unità.

Il valore errato potrebbe essere causato dall'errore di misurazione, ma non si sovrappone completamente con esso. Infatti, un errore di misurazione può generare anche valori non veri, ma comunque plausibili per l'unità considerata. La localizzazione dei valori errati viene effettuata attraverso controlli o regole di edit [29] che possono essere di tipo *logico* (nel caso di variabili qualitative) o *statistico* (nel caso di variabili quantitative).

I prossimi due paragrafi descrivono la *struttura* delle mancate risposte all'interno dei dataset, cioè il modo in cui si distribuiscono i dati mancanti all'interno della matrice e i *meccanismi* generatori dei dati mancanti, ovvero le relazioni che intercorrono tra questi ultimi e i valori delle variabili [60].

1.3.1 Le strutture delle mancate risposte

Data la matrice rettangolare $\mathbf{X} = \{x_{ij}\}$, $i = 1, ..., I$ e $j = 1, ..., J$, di I osservazioni per J variabili, sia x_{ij} il valore osservato della variabile $\mathbf{X}_j$ sull'unità i. Si definisce $\mathbf{M} = \{m_{ij}\}$ la *matrice indicatrice dei valori mancanti,* tale che

$$m_{ij} = \begin{cases} 0, & \text{se } x_{ij} \text{ è osservata} \\ 1, & \text{se } x_{ij} \text{ è missing} \end{cases} \quad \text{con } i = 1, ..., I \text{ e } j = 1, ..., J \tag{1.1}$$

Data $\mathbf{M}$, la matrice $\mathbf{X}$ può essere scomposta nelle seguenti componenti:

$$\begin{aligned} \mathbf{X}_{(com)} &= \left\{\mathbf{x}_i' \in \mathbf{X}, \forall i | \textstyle\sum_{j=1}^{J} m_{ij} = 0\right\} \\ X_{(oss)} &= \{x_{ij} \in \mathbf{X}, \forall i, j | m_{ij} = 0\} \\ X_{(mis)} &= \{x_{ij} \in \mathbf{X}, \forall i, j | m_{ij} = 1\} \end{aligned} \tag{1.2}$$

con $\mathbf{x}_i' = [x_{i1}, x_{i2}, ..., x_{ij}, ..., x_{iJ}]$ vettore delle realizzazioni relative all'unità $i - esima$. $\mathbf{X}_{(com)}$ è la matrice delle unità completamente osservate, $X_{(oss)}$ è il sottoinsieme degli elementi osservati di $\mathbf{X}$ e $X_{(mis)}$ è il sottoinsieme degli elementi non osservati di $\mathbf{X}$.

Si possono avere le seguenti configurazioni di dati mancanti (Fig. 1.1):

- *mancate risposte totali (MRT)*: nel dataset è presente un sottoinsieme di unità su cui non è stato rilevato alcun valore per ciascuna delle J variabili (Fig. 1.1(a));

(a) Mancate risposte totali

X_1	X_2	X_3	X_4	X_5
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
1	1	1	1	1
0	0	0	0	0
0	0	0	0	0
1	1	1	1	1
0	0	0	0	0

(b) Mancate risposte parziali univariate

X_1	X_2	X_3	X_4	X_5
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	1

(c) Mancate risposte parziali comuni ad un sottoinsieme di unità

X_1	X_2	X_3	X_4	X_5
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	1	1	1
0	0	1	1	1
0	0	1	1	1
0	0	1	1	1
0	0	1	1	1

(d) Mancate risposte parziali monotone

X_1	X_2	X_3	X_4	X_5
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	0
0	0	0	0	1
0	0	0	1	1
0	0	1	1	1
0	1	1	1	1

(e) Mancate risposte parziali diffuse su alcune variabili

X_1	X_2	X_3	X_4	X_5
0	0	1	0	0
0	0	0	0	1
0	0	1	0	1
0	0	0	1	0
0	0	1	0	0
0	0	0	0	1
0	0	0	1	0
0	0	0	0	1
0	0	1	0	0

(f) Mancate risposte parziali diffuse su tutte le variabili

X_1	X_2	X_3	X_4	X_5
1	0	1	0	0
0	0	0	0	1
0	1	0	0	0
0	0	0	1	0
1	1	1	0	0
0	0	0	0	1
0	0	0	1	0
0	1	0	0	1
0	0	1	0	0

(g) File matching

Y	X_1
0	1
0	1
0	1
0	1
0	0
0	0
0	0
0	0
0	0

Y	X_1	X_2
0	0	0
0	0	0
0	0	0
0	0	0
0	1	0
0	1	0
0	1	0
0	1	0
0	1	0

(h) Variabili latenti

X_1	X_2	X_3	Y_1	Y_2
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1
0	0	0	1	1

Figura 1.1: Esempio di strutture di mancata risposta

- *mancate risposte parziali (MRP) univariate*: una sola delle J variabili presenta valori mancanti in alcune unità. Storicamente questo è uno dei primi, ed anche il più semplice, dei casi di analisi di dataset incompleto (Fig. 1.1(b));

- *mancate risposte parziali comuni ad un sottoinsieme di unità*: la matrice dei dati presenta record con valori mancanti relativamente ad uno stesso sottoinsieme di variabili. Può accadere nelle indagini campionarie, dove una quota degli intervistati, per motivi diversi, non risponde ad alcune domande. Le variabili completamente osservate possono essere date da un insieme di item relativi ad altre sezioni del questionario, oppure da quelle variabili che contribuiscono alla realizzazione del disegno dell'indagine. In quest'ultimo caso, se l'oggetto di analisi non prevede l'uso di tali variabili strutturali, le mancate risposte diventano totali (Fig. 1.1(c));

- *mancate risposte parziali monotone*: è una situazione che può presentarsi negli studi longitudinali, dove la stessa variabile viene rilevata sullo stesso insieme di casi, ma in tempi diversi. Per un fenomeno di "logoramento", vi possono essere delle cadute di intervistati e quindi, da un certo istante temporale in poi, per essi non vi sono più valori osservati (Fig. 1.1(d));

- *mancate risposte parziali diffuse su alcune variabili*: anche in questo caso, esiste una parte degli indicatori completamente osservata. Ma le

variabili non completamente osservate, presentano valori mancanti in modo sparso. (Fig. 1.1(e));

- *mancate risposte parziali diffuse su tutte le variabili*: questa è la situazione più generale ed anche più complessa da gestire, in quanto tutte le variabili considerate presentano una quota di valori mancanti (Fig. 1.1(f));

- *file matching*: nell'ipotesi che si abbiano a disposizione più dataset, aventi in comune alcuni indicatori completamente osservati ed altri con valori mancanti, allo scopo di integrare e completare le informazioni, si può attuare una tecnica di *record-linkage* [28], se almeno in parte le unità dei due dataset sono le stesse, oppure si può applicare una procedura di *matching statistico* [27], se i due insiemi di osservazioni sono completamente disgiunti (Fig. 1.1(g));

- *variabili latenti*: quest'ultima situazione si discosta da quelle precedenti. Infatti, il dataset a disposizione è completamente osservato e si vuole stimare il valore di variabili di cui si ipotizza l'esistenza, ma che risultano essere "non osservabili". In tal caso, si può fare ricorso a tecniche di analisi fattoriali [18, 59, 75] con lo scopo è di individuare delle nuove variabili, combinazioni lineari di quelle iniziali, in grado di spiegare i fenomeni non direttamente misurabili (Fig. 1.1(h));

1.3.2 I meccanismi generatori delle mancate risposte

La formalizzazione del concetto di meccanismo generatore dei dati mancanti, sebbene di fondamentale importanza, in quanto le proprietà dei metodi per il trattamento delle mancate risposte dipendono fortemente da tali meccanismi, fu teorizzato soltanto nel 1976 da Rubin [77] che decise di trattare l'indicatore dei missing data come una variabile casuale ed assegnare ad esso una distribuzione di probabilità.

Considerata la matrice dei dati completi $\mathbf{X}$ e la matrice indicatrice dei dati mancanti $\mathbf{M}$, il meccanismo generatore dei dati mancanti è caratterizzato dalla distribuzione condizionale di $\mathbf{M}$ data $\mathbf{X}$, cioè $f(\mathbf{M}|\mathbf{X}, \phi)$, con ϕ vettore di parametri ignoti appartenente allo spazio Ω_ϕ. Se la distribuzione di probabilità dei dati mancanti assume la forma:

$$f(\mathbf{M}|\mathbf{X}, \phi) = f(\mathbf{M}|\phi) \; \forall \mathbf{X}, \phi \tag{1.3}$$

si parlerà di dati mancanti di tipo *missing completely at random* (MCAR).

Tale assunzione non significa che la struttura dei dati sia casuale, quanto piuttosto che il fatto che i dati siano mancanti non dipende dai dati stessi. Se si scompone la matrice $\mathbf{X}$ nelle sue componenti osservate $X_{(oss)}$ e nelle sue componenti non osservate $X_{(mis)}$, una assunzione meno restrittiva del MCAR è che i dati mancanti dipendano soltanto dalle componenti osservate di $\mathbf{X}$ e non da quelle non osservate, cioè:

$$f(\mathbf{M}|\mathbf{X}, \phi) = f(\mathbf{M}|X_{(oss)}, \phi) \; \forall X_{(mis)}, \phi \tag{1.4}$$

Un meccanismo generatore di dati mancanti di questo tipo si definisce *missing at random* (MAR).

Nel caso in cui i dati mancanti dipendano dai valori non osservati, allora si parlerà di meccanismo *non missing at random* (NMAR). Rispetto a quest'ultima ipotesi, possono aversi due differenti situazioni: i dati mancanti dipendono da variabili completamente non osservate, oppure i dati mancanti di una certa variabile sono dovuti alla variabile stessa.

Un altro aspetto molto importante da considerare è legato all'assunzione che il vettore dei parametri θ, (appartenente allo spazio Ω_θ) che caratterizza la distribuzione di probabilità della $\mathbf{X}$, sia distinto dall'insieme dei parametri ϕ che condiziona $\mathbf{M}$. Infatti, se l'insieme congiunto dei parametri (θ, ϕ) è il prodotto cartesiano degli spazi parametrici individuali cioè se $\Omega_{\theta,\phi} = \Omega_\theta \times \Omega_\phi$ e se è verificata l'ipotesi MAR, allora ci si trova in presenza di meccanismi *ignorabili* [60].

In realtà non si può essere assolutamente certi del fatto che in presenza di un dataset incompleto, le informazioni mancanti siano di tipo MAR, oppure che ci si trovi in presenza di un meccanismo non ignorabile, o che l'assenza di informazione dipenda dalla variabile incompleta, invece che da un predittore non osservato. Si possono soltanto fare assunzioni, o fare riferimento ad altri studi, oppure includere il maggior numero di variabili possibili nel modello, così che l'ipotesi MAR possa diventare ragionevole [33].

1.3.3 Il trattamento delle mancate risposte

La letteratura relativa all'analisi dei dataset incompleti è relativamente recente. I metodi proposti in letteratura possono essere raggruppati nelle seguenti categorie [60], che possono essere usate anche in combinazione tra di esse:

- *inclusione* dell'attributo "mancata risposta" tra quelli del dominio;
- analisi basate sulle *unità completamente osservate*;
- *riponderazione* delle unità completamente osservate;

- *modelli probabilistici* per matrici di dati incompleti;
- procedure di *imputazione*, singola o multipla, per la ricostruzione del dataset incompleto.

1.3.3.1 Inclusione dell'attributo mancanta risposta tra quelli del dominio

E' il metodo più semplice per gestire le mancate risposte. In pratica, si assegna una codifica al valore mancante, in modo da includerlo negli attributi del dominio. In questo modo, il valore mancante diventa oggetto di analisi al pari delle altre modalità e può essere analizzato in concomitanza con i valori delle altre variabili, per individuare quale sia stato il meccanismo generatore dei dati mancanti. Tuttavia, tale tecnica è di applicazione più semplice in presenza di variabili categoriali, in quanto nel caso di variabili numeriche, essendo del tutto arbitraria la codifica del dato mancante, tale codifica incide nel calcolo di tutti gli indicatori sintetici.

1.3.3.2 Analisi basate sulle unità completamente osservate

Possono essere effettuate nei due seguenti modi:

- *analisi dei casi completi.* L'attenzione è posta sulle sole unità completamente osservate. In sostanza, si effettuano le elaborazioni su un campione di numerosità ridotta rispetto a quello inizialmente programmato. I vantaggi sono dovuti essenzialmente alla semplicità di esecuzione e alla comparabilità tra le statistiche univariate, in quanto realizzate tutte sullo stesso insieme di casi. Gli svantaggi sono dati dalla potenziale perdita di informazione, dovuta ai casi scartati. La perdita di informazione può essere tradotta in termini di perdita di precisione, ma anche di distorsione, qualora il meccanismo generatore dei dati mancanti non sia MCAR. E' chiaro che tale tecnica ha senso soltanto se l'imprecisione e la distorsione sono di entità ridotta, valore non sempre riconducibile soltanto alla proporzione di casi scartati sul totale del campione;
- *analisi dei casi disponibili.* L'analisi dei soli casi completi può essere estremamente inappropriata nel caso di stime di indicatori univariati quali le medie o le distribuzioni di frequenza marginali, poichè i valori osservati di alcune variabili potrebbero essere scartati anche quando i dati mancanti appartengano ad altre variabili. Tale inefficienza può essere molto elevata nel caso di dataset con molte variabili aventi ognuna,

indipendentemente dalle altre, una certa quantità di dati mancati, come nel caso della figura 1.1(f). L'alternativa per le analisi univariate è quella di includere tutti i casi dove la variabile di interesse è presente, così da utilizzare tutti i valori disponibili. Se si vogliono calcolare misure di covarianza tra coppie di variabili, si possono utilizzare i casi presenti per entrambi gli attributi. Lo svantaggio di questo metodo è dovuto al fatto che per ogni variabile, si calcola un indicatore che ha una diversa base campionaria. Oltretutto, se il meccanismo generatore dei dati non è MCAR, vi è anche un problema di comparabilità tra variabili.

1.3.3.3 Riponderazione delle unità completamente osservate

In presenza di osservazioni completamente non osservate (Fig. 1.1(a)), si può pensare di riponderare le unità osservate. In situazioni di inferenza da campioni di unità completamente osservate, a ciascun caso viene asseganto un peso che è inversamente proporzionale alla probabilità di selezione. Ad esempio, se x_i è il valore della variabile X per l'unità i della popolazione, la media è spesso stimata tramite lo stimatore di Horvitz-Thompson [14]:

$$\bar{x} = \frac{\left(\sum_{i=1}^{I} x_i \pi_i^{-1}\right)}{\left(\sum_{i=1}^{I} \pi_i^{-1}\right)}$$

dove la somma è su tutte le unità campionate e π_i è la probabilità conosciuta di inclusione nel campione per l'unità i. Le procedure di riponderazione modificano il peso nel tentativo di redistribuire il peso complessivo dei non rispondenti tra i rispondenti. La stima della media diventa quindi:

$$\bar{x} = \frac{\left(\sum_{i=1}^{\hat{I}} x_i \left(\hat{p}_i \pi_i\right)^{-1}\right)}{\left(\sum_{i=1}^{\hat{I}} \left(\hat{p}_i \pi_i\right)^{-1}\right)}$$

dove questa volta la somma è su tutte le unità rispondenti e $\hat{p}_i$ è una stima della probabilità di risposta per l'unità i, generalmente data dalla proporzione di unità nel sottogruppo del campione.

1.3.3.4 Modelli probabilistici per dataset incompleti: l'algoritmo EM

Seppure in presenza di dati incompleti è possibile, avendo specificato un modello probabilistico, stimare i suoi parametri attraverso funzioni di massima verosimiglianza. Se $f(X_{(oss)}, X_{(miss)}|\theta)$ è la distribuzione di probabilità con-

giunta dei valori osservati e di quelli mancanti, si definisce *verosimiglianza rispetto ai dati osservati* $X_{(oss)}$ la funzione:

$$L(\theta|X_{(oss)}) \propto f(X_{(oss)}|\theta) = \int f(X_{(oss)}, X_{(miss)}|\theta)dX_{(miss)} \tag{1.5}$$

Affinchè si possano effettuare valide inferenze sui parametri θ utilizzando soltanto i valori osservati, senza dover esplicitare un modello per le non-risposte, occorre che il meccanismo di mancata risposta sia ignorabile. Si consideri infatti, la distribuzione congiunta dei dati $\mathbf{X}$ e delle variabili indicatrici $\mathbf{M}$; se $\Omega_{\theta,\phi} = \Omega_\theta \times \Omega_\phi$, essa può essere fattorizzata come segue:

$$f(X_{(oss)}, X_{(miss)}, \mathbf{M}|\theta, \phi) = f(\mathbf{M}|X_{(oss)}, X_{(miss)}, \phi)f(X_{(oss)}, X_{(miss)}|\theta)$$

Considerando i soli dati osservati, la funzione di verosimiglianza per la stima dei parametri (θ, ϕ) sarà data da:

$$L(\theta, \phi|X_{(oss)}, \mathbf{M}) \propto \int f(\mathbf{M}|X_{(oss)}, X_{(miss)}, \phi)f(X_{(oss)}, X_{(miss)}|\theta)dX_{(miss)} \tag{1.6}$$

Se il meccanismo generatore delle mancate risposte non dipende dai valori mancanti, condizionatamente a quelli osservati, cioè se vale la (1.3) o almeno la (1.4), si ottiene:

$$\begin{aligned} L(\theta, \phi|X_{(oss)}, \mathbf{M}) &\propto f(\mathbf{M}|X_{(oss)}, \phi) \int f(X_{(oss)}, X_{(miss)}|\theta)dX_{(miss)} \\ &= f(\mathbf{M}|X_{(oss)}, \phi)f(X_{(oss)}|\theta) \\ &\propto L(\theta|X_{(oss)}) \end{aligned}$$

Quindi, i valori che massimizzano la funzione (1.6) sono gli stessi che massimizzano la (1.5), cioè le inferenze su θ possono essere effettuate sulla verosimiglianza dei soli dati osservati.

Tuttavia, anche ammesso che si possa ipotizzare un meccanismo generatore dei dati mancanti ignorabile, ciò non toglie che la massimizzazione della verosimiglianza (1.5) sia un problema di non semplice soluzione. Infatti, a meno che la matrice dei dati non abbia una configurazione di tipo monotono come nella figura 1.1(d), è difficile che si possano trovare soluzioni in modo analitico per le equazioni di massima verosimiglianza. E' in queste circostanze che si può utilizzare l'algoritmo *Expectation-Maximization* (EM) [22]. Tale metodo consente, attraverso una procedura iterativa, di ottenere le stime di massima

verosimiglianza dei parametri θ in presenza di dati incompleti, riconducendo l'analisi ad un problema di stima per dati completi. Partendo da una stima iniziale $\theta^{(0)}$, l'algoritmo ad ogni iterazione t applica i due seguenti passi:

- *passo-E*: si calcola il valore atteso della funzione di log-verosimiglianza dei dati completi $Q\left(\theta|\theta^{(t)}\right)$, integrando la log-verosimiglianza dei dati completi rispetto alla distribuzione dei dati mancanti condizionata ai valori osservati e alle stime correnti dei parametri $\theta^{(t)}$:

$$Q\left(\theta|\theta^{(t)}\right) = \int \ln L(\theta|X_{(oss)}, X_{(miss)}) f(X_{(miss)}|X_{(oss)}, \theta^{(t)}) dX_{(miss)}$$

- *passo-M:* si determinano le nuove stime $\theta^{(t+1)}$, massimizzando la funzione $Q\left(\theta|\theta^{(t)}\right)$ rispetto a θ:

$$\theta^{(t+1)} = \arg\max_{\theta} Q\left(\theta|\theta^{(t)}\right)$$

L'algoritmo genera una successione $\left\{\theta^{(t)}\right\}_{t=1,2,\ldots}$ che sotto alcune ipotesi di regolarità [22, 66] converge alla stima di massima verosimiglianza dei parametri θ. Un risultato fondamentale che si ottiene dall'applicazione dell'EM è il seguente:

$$Q\left(\theta^{(t+1)}|\theta^{(t)}\right) \geq Q\left(\theta^{(t)}|\theta^{(t)}\right) \Rightarrow \ln L(\theta^{(t+1)}|X_{(oss)}) \geq \ln L(\theta^{(t)}|X_{(oss)})$$

Infatti, poichè la distribuzione congiunta dei valori osservati e di quelli mancanti può essere così fattorizzata:

$$f(X_{(oss)}, X_{(miss)}|\theta) = f(X_{(oss)}|\theta) f(X_{(miss)}|X_{(oss)}, \theta)$$

considerando la corrispondente scomposizione della log-verosimiglianza, si ottiene:

$$\ln L(\theta|X_{(oss)}, X_{(miss)}) = \ln L(\theta|X_{(oss)}) + \ln f(X_{(miss)}|X_{(oss)}, \theta)$$

La log-verosimiglianza dei dati osservati potrà allora essere riscritta come:

$$\ln L(\theta|X_{(oss)}) = \ln L(\theta|X_{(oss)}, X_{(miss)}) - \ln f(X_{(miss)}|X_{(oss)}, \theta)$$

Prendendo il valore atteso di entrambi i membri, rispetto alla distribuzione dei dati mancanti, condizionata ai valori osservati e alle stime correnti $\theta^{(t)}$ si ha:

$$\ln L(\theta|X_{(oss)}) = Q\left(\theta|\theta^{(t)}\right) - H\left(\theta|\theta^{(t)}\right)$$

con:

$$H\left(\theta|\theta^{(t)}\right) = \int \left[\ln f(X_{(miss)}|X_{(oss)},\theta)\right] f(X_{(miss)}|X_{(oss)},\theta^{(t)})dX_{(miss)}$$

Considerata una sequenza $\theta^{(0)}, \theta^{(1)}...$, la differenza tra i valori di $\ln L(\theta|X_{(oss)})$ in due successive iterazioni è data da:

$$\begin{aligned}\ln L(\theta^{(t+1)}|X_{(oss)}) - \ln L(\theta^{(t)}|X_{(oss)}) &= Q\left(\theta^{(t+1)}|\theta^{(t)}\right) - Q\left(\theta^{(t)}|\theta^{(t)}\right) \\ &+ H\left(\theta^{(t)}|\theta^{(t)}\right) - H\left(\theta^{(t+1)}|\theta^{(t)}\right)\end{aligned}$$

Ma $Q\left(\theta^{(t+1)}|\theta^{(t)}\right) \geq Q\left(\theta^{(t)}|\theta^{(t)}\right)$, per il passo-M dell'algoritmo EM, mentre $H\left(\theta^{(t)}|\theta^{(t)}\right) \geq H\left(\theta^{(t+1)}|\theta^{(t)}\right)$, per la disuguaglianza di Jensen[1].

1.3.3.5 Procedure di imputazione

A causa della natura spesso multivariata dei dati, con tutte le variabili potenzialmente soggette alla presenza di valori mancanti (si pensi alla situazione in Fig. 1.1(f)), si impone la necessità di una strategia complessiva per la gestione delle mancate risposte [48].

L'imputazione consiste nell'assegnare un valore sostitutivo al dato mancante con lo scopo di ripristinare la matrice dei dati completa. Come strategia, l'imputazione ha tre caratteristiche desiderabili:

1. consente di ridurre le distorsioni nelle stime finali, dovute al fatto che le distribuzioni dei dati mancanti tendenzialmente sono differenti da quelle dei dati osservati;

2. consente di lavorare sul dataset come se questo fosse completo, facilitando l'analisi dei dati e la presentazione dei risultati;

3. consente di effettuare analisi che conducono a risultati consistenti, in particolar modo se le tecniche di imputazione tengono conto delle relazioni di coerenza tra i dati e dei legami tra le variabili.

Quando in un record sono presenti dei valori mancanti per qualche item, si può supporre che essi possano essere desunti da altre variabili nel questionario

[1] Sia $(\Omega, \mathcal{F}, P)$ uno spazio di probabilità, X una variabile aleatoria a valori reali della quale esiste $\mathbb{E}(X)$ e φ una funzione convessa tale che esista $\mathbb{E}[\varphi(X)]$. Allora $\varphi[\mathbb{E}(X)] \leq \mathbb{E}[\varphi(X)]$.

Variabili ausiliarie	Metodi Deterministici	Metodi Stocastici
Nessuna	Media totale	Valore casuale
Categoriali	Media nelle classi	Valore casuale nelle classi
Continue	Regressione predittiva	Regressione casuale

Tabella 1.1: Classificazione di alcuni metodi di imputazione

(variabili ausiliarie o covariate). Per l'$i-esimo$ rispondente, il legame tra la variabile $\mathbf{X}_j$, e le altre può essere descritto dalla relazione [48]:

$$x_{(mis)ij} = f\left(x_{(oss)i1}, ..., x_{(oss)iJ}\right) + \varepsilon_{ij} \tag{1.7}$$

con $f(...)$ funzione degli altri item osservati, e ε_{ij} il residuo stimato.

In base a questa interpretazione, le tecniche di imputazione possono essere classificate (come schematizzato in Tab. 1.1) secondo due dimensioni:

1. La prima dipende dal valore assegnato al residuo. Si parlerà di:
   (a) metodi deterministici ($\varepsilon_{ij} = 0$). Sono quei metodi tali che le imputazioni ripetute per unità aventi le stesse caratteristiche producono sempre gli stessi valori imputati;
   (b) metodi stocastici. Si caratterizzano per la presenza di una componente aleatoria, detta residuo, che corrisponde allo schema probabilistico associato allo schema di imputazione scelto;
2. La seconda dipende dell'uso o meno di variabili ausiliarie e della tipologia di queste ultime (continue o categoriali).

Imputazione tramite metodi deterministici

Imputazione con metodi deduttivi Si basano sulla possibilità di sfruttare le ridondanze presenti nel dataset in modo da poter dedurre il valore da sostituire al dato mancante mediante deduzione da una o più variabili ausiliarie. L'applicazione di tali metodi è molto legata alla conoscenza del fenomeno in quanto non esente da valutazioni di tipo soggettivo [37].

Imputazione con media totale La procedura si basa sull'assegnazione della media totale dei rispondenti a tutti i valori mancanti. E' la degenerazione deterministica della funzione lineare (1.7), senza variabili ausiliarie:

$$x_{(mis)ij} = \bar{x}_{(oss)j}$$

Imputazione con media nelle classi Il metodo prevede che il campione totale sia ripartito in H classi di imputazione, in accordo coi valori delle variabili ausiliarie, definite da tutte le celle date dall'incrocio delle modalità delle altre variabili (eventualmente categorizzate). Per ogni classe, il valore medio della variabile calcolata sui rispondenti è assegnato a tutti i non rispondenti di quella classe:

$$x_{(mis)ijh} = \bar{x}_{(oss)jh}$$

con $h = 1, ..., H$.

Imputazione con regressione predittiva In questo caso, vengono regrediti i valori osservati della variabile oggetto di imputazione sulle variabili ausiliarie. Successivamente, i valori mancanti vengono imputati attraverso quelli predetti mediante l'equazione di regressione. E' un metodo deterministico, in quanto $\varepsilon_{ij} = 0$. Questo metodo, può essere assimilato all'imputazione con media nelle classi, se le variabili ausiliarie sono di tipo categoriale: in tal caso infatti entrambe le procedure possono essere descritte dalla seguente equazione:

$$x_{(mis)ij} = b_0 + b_{(oss)11} y_{(oss)i11} + ... + b_{(oss)j'l} y_{(oss)ij'l} + ... + b_{(oss)JL_J} y_{(oss)iJL_J}$$

dove le $y_{(oss)ij'l} = I(x_{ij'} = a_{j'l})$ sono variabili indicatrici, essendo $x_{ij'} \in Dom(A_{j'}) = \left\{a_{j'1}, a_{j'2}, ..., a_{j'l}, ..., a_{j'L_{j'}}\right\}$, con $1 \leq j' \leq J$, il numero di variabili e $1 \leq l \leq L_{j'}$, il numero delle modalità relative all'attributo $A_{j'}$.

Imputazione con matching della media Il valore $x_{(mis)ij}$ ottenuto tramite regressione predittiva viene messo a confronto con i valori $x_{(oss)i'j}$. Il valore imputato è quello del donatore che presenta distanza minore dal valore della regressione. Alternativamente, si può decidere di fissare un valore soglia per la distanza e selezionare uno tra tutti quei potenziali donatori tali che la distanza dal valore stimato sia ineriore o uguale a tale soglia.

Imputazione tramite matching con funzione di distanza Un approccio più generale è quello di definire una metrica, basata sui valori delle covariate, per misurare la vicinanza tra le unità rispondenti e quelle non completamente osservate. Il donatore sarà l'osservazione la cui distanza rispetto all'osservazione incompleta sarà pari a zero o comunque minima rispetto agli altri rispondenti. Le varianti di questo metodo possono essere ricondotte alla metrica utilizzata: euclidea, ponderata rispetto all'importanza di ogni variabi-

le, di Mahalanobis. Nel caso invece si faccia uso di variabili ausiliarie categoriali, allora si potrà ricorrere a misure di similarità [75]. L'uso di metriche o di funzioni di similarità consente di tenere sotto controllo la ridondanza nell'uso dei record donatori, potendo definire una misura nel seguente modo [15]:

$$d^*(i, i') = d(i, i')(1 + pr_{i'})$$

essendo $d(i, i')$ la metrica di base, p una costante di penalizzazione ed $r_{i'}$ il numero di volte con cui il donatore i' è stato già utilizzato.

Imputazione con metodi stocastici

Imputazione con donatore casuale E' una procedura che consente di assegnare a ciascun non rispondente il valore di un rispondente selezionato a caso dal campione del totale dei rispondenti. Tale metodo è la degenerazione stocastica della funzione lineare della (1.7) senza variabili ausiliarie:

$$x_{(mis)ij} = \bar{x}_{(oss)j} + \varepsilon_{ij}$$

dove $\varepsilon_{ij} = x_{(oss)i'j} - \bar{x}_{(oss)j}$.

Imputazione con donatore casuale nelle classi Come nel caso dell'imputazione con media nelle classi, il campione viene ripartito tra H classi di imputazione. All'interno di ciascuna classe, per ogni non rispondente, viene scelto casualmente il valore di un rispondente all'interno della medesima classe:

$$x_{(mis)ijh} = \bar{x}_{(oss)jh} + \varepsilon_{ijh}$$

dove $\varepsilon_{ijh} = x_{(oss)i'jh} - \bar{x}_{(oss)jh}$.

Imputazione con regressione casuale E' la versione stocastica della regressione predittiva, per cui i valori da imputare sono quelli della regressione predittiva più il termine residuo ε_{ij}. A seconda delle assunzioni fatte, i residui possono essere individuati in vari modi, tra cui:

1. se si assume che i residui siano omoschedastici e normalmente distribuiti, possono essere selezionati casualmente da una distribuzione normale con media pari a zero e varianza pari a quella residua della regressione;

2. se si assume che i residui derivino da una stessa distribuzione, sconosciuta, possono essere selezionati casualmente dai residui dei rispondenti;

3. se si hanno dubbi sulla linearità e sulla addittività delle componenti del modello di regressione, i residui potrebbero essere presi dai rispondenti con i valori delle variabili ausiliarie più simili a quelli dell'osservazione incompleta . In tal modo, se il rispondente ha lo stesso identico insieme di valori del non rispondente, la procedura si riduce ad assegnare il valore del rispondente al non rispondente.

Il terzo caso, mostra il forte legame tra questo metodo e l'imputazione con donatore casuale all'interno delle classi.

Imputazione tramite cold-deck Il valore mancante di una variabile viene sostituito con un valore costante ottenuto da una fonte esterna, ad esempio da una indagine effettuata in una precedente occasione.

Imputazione tramite hot-deck sequenziale Tramite le variabili ausiliarie, si definiscono delle classi di imputazione e per ciascuna di esse, si stabilisce un valore iniziale per la variabile da imputare. Il valore può essere scelto casualmente tra i rispondenti, oppure può essere un valore rappresentativo come la media tra i rispondenti all'interno della classe. Le osservazioni del dataset sono trattate sequenzialmente: se la $i - esima$ osservazione presenta un valore osservato per la variabile X, allora tale valore diventa quello di riferimento per la sua classe di imputazione, nel caso contrario assumerà l'ultimo valore registrato rispetto alla sua classe d'appartenenza. Se l'ordine del dataset è casuale, allora questa tecnica è assimilabile all'imputazione con donatore casuale nelle classi.

Imputazione tramite hot-deck gerarchico E' simile al metodo precedentemente descritto, ma viene applicato quando le unità sono ripartite in classi, costruite su un grande numero di variabili ausiliarie. In tal caso, qualora non vi sia matching tra osservazione incompleta ed alcun donatore all'interno delle classi, omettendo una delle variabili che contribuiscono alla formazione delle classi, si cerca di effettuare un matching a un livello più basso, collassando le classi.

Imputazione tramite algoritmo EM Nel contesto delle mancate risposte parziali, l'algoritmo EM è utilizzato per la stima dei parametri del modello attraverso il quale vengono successivamente effettuate le imputazioni dei valori non osservati. Tali imputazioni possono essere effettuate in due modi diversi [39]:

- *conditional mean imputation*: i dati mancanti possono essere imputati mediante i valori attesi condizionati ai valori osservati e alle stime individuate, $\mathbb{E}\left(X_{(miss)}|X_{(oss)}, \hat{\theta}\right)$;
- *random draw*: il dataset può essere completato generando i valori dalla distribuzione di probabilità condizionata ai valori osservati e i parametri stimati, $P\left(X_{(miss)}|X_{(oss)}, \hat{\theta}\right)$.

Il primo metodo è preferibile se si vogliono stimare quantità univariate lineari nei dati come medie o totali. Qualora invece si sia interessati alla stima di parametri distribuzionali legati alle relazioni di interdipendenza tra le variabili, l'imputazione mediante valori attesi condizionati potrebbe comportare forti distorsioni, e il secondo metodo è preferibile [49].

Imputazione multipla Attraverso le tecniche di imputazione singola di tipo stocastico, si generano più dataset completi, che possono essere analizzati separatamente secondo procedure standard ed i cui risultati possono essere combinati per ottenere stime che incorporano l'incertezza dovuta ai dati mancanti [79].

1.4 Contenuto ed obiettivi del presente lavoro

Oggetto del presente lavoro è l'introduzione di un insieme di tecniche di imputazione per dataset di variabili categoriali parzialmente osservate la cui struttura sia assimilabile a quella descritta nella figura 1.1(f).

Da un punto di vista organizzativo, i metodi sono stati raggruppati secondo una formalizzazione dei dati fondata sul concetto di spazio vettoriale [18]. Considerata la matrice $\mathbf{X}$ con I righe e J colonne, ogni riga esprime i valori assunti nelle I unità statistiche dai J caratteri o variabili. Se queste ultime sono di tipo quantitativo, le colonne della matrice potranno essere rappresentate come vettori in uno spazio $I - dimensionale$, dove si possono applicare le operazioni di somma vettoriale, prodotto per uno scalare e introducendo una opportuna metrica, si può effettuare l'operazione di prodotto scalare. Tale spazio si definisce *spazio delle variabili.* Analogamente, i vettori-riga della matrice $\mathbf{X}$ potranno essere proiettati in uno spazio vettoriale $J-dimensionale$, lo *spazio delle osservazioni.* Sebbene, nel caso specifico, oggetto di analisi siano matrici di variabili qualitative, è tuttavia possibile renderle rappresentabili in spazi vettoriali, attraverso la trasformazione in:

- *forma disgiuntiva completa*, ovvero come tabella di variabili indicatrici;

- *tabella di contingenza* $J - dimensionale$, ovvero formata dal conteggio delle frequenze di unità aventi aventi lo stesso profilo.

Nel capitolo 2, riservato ai metodi nello spazio delle osservazioni, saranno descritte alcune tecniche di cluster analysis fuzzy per variabili nominali [45, 51, 74, 89]. Tali metodi, originariamente impiegati per l'individuazione di raggruppamenti di unità simili rispetto a criteri prestabiliti, sono stati modificati per essere utilizzati a scopo di imputazione. Le procedure sono divise in metodi non probabilistici e in metodi probabilistici, a seconda che come "dati" si consideri la sola matrice $\mathbf{X}$ oppure, insieme con essa, anche una famiglia di modelli stocastici [18].

Nel capitolo 3, riservato alle metodologie nello spazio delle variabili, verranno descritte le reti bayesiane o sistemi esperti probabilistici [20, 47, 71]. Si tratta di una classe di modelli statistico/probabilistici che, tramite rappresentazioni grafiche associate a tabelle di distribuzioni di probabilità congiunta, consentono di evidenziare le relazioni di dipendenza condizionale tra le variabili del dominio. In questo contesto, il loro utilizzo sarà finalizzato all'imputazione dei valori mancanti.

L'idea di prendere in considerazione metodologie che necessitano di una formalizzazione probabilistica insieme ad altre tecniche di tipo esplorativo va nella direzione dell' "approccio strategico integrato" [17] all'analisi dei dati. Esso è orientato all'utilizzo e alla integrazione di diverse formalizzazioni matematiche (probabilistica, geometrica, teoria dei grafi) senza preclusioni di sorta, aventi in comune l'obiettivo di convogliare l'*informazione empirica* (insita nei dati statistici) e l'*informazione teorica* (le strutture concettuali di riferimento), così come descritte nell'ambito del "paradigma informazionale" [19], verso l'individuazione di una una nuova *informazione addizionale*, che nel caso specifico è data dal completamento della matrice di dati iniziale.

Nel capitolo 4 verranno descritti i risultati ottenuti applicando le procedure proposte sia separatamente, su dataset ad hoc, sia congiuntamente su uno stesso dataset reale, in modo da compararne le performance.

Nell' appendice A saranno fornite alcune nozioni sugli insiemi fuzzy [92], mentre nell'appendice B verranno riportate le implementazioni in linguaggio **R** [73] di tutti gli algoritmi utilizzati.

Capitolo 2

La cluster analysis

2.1 Definizione di cluster analysis

Le tecniche di classificazione automatica dei dati (o di cluster analysis) [46, 75], trovano collocazione in letteratura, nel contesto del *machine learning* [4] all'interno dei metodi di apprendimento non supervisionato. Data una matrice $\mathbf{X}$, di I unità, su cui sono stati rilevati J caratteri, obiettivo della cluster analysis è quello di partizionare le unità in $1 < K \ll I$ raggruppamenti, omogenei rispetto ad una misura di prossimità opportunamente scelta. I gruppi ottenuti dovranno avere due caratteristiche principali:

- essere ben separati tra loro;
- essere internamente compatti.

E' bene sottolineare innanzitutto la differenza tra tecniche di apprendimento supervisionato e non supervisionato: nel primo caso, le unità della matrice dei dati $\mathbf{X}$, possiedono già un etichetta (label), ovvero sono già assegnate ad un gruppo; obiettivo dell'analisi è quello di assegnare una label alle eventuali unità aggiuntive usando come dataset di *training* la matrice $\mathbf{X}$. Nel caso dell'apprendimento non supervisionato, le etichette non sono esplicite, ma possono essere individuate mediante le variabili osservate.

Nel contesto della cluster analysis, esistono numerose varianti che si possono distinguere in base al fatto che:

- il numero di raggruppamenti venga stabilito a priori oppure generato in modo sequenziale (algoritmi partizionali vs algoritmi gerarchici);
- le unità vengano completamente assegnate ai gruppi individuati o parzialmente assegnante (hard clustering vs fuzzy clustering);

- esista una formalizzazione probabilistica oppure no (metodi stocastici vs metodi deterministici).

Le matrici di dati sottoposte a cluster analysis possono contenere varibili quantitative, qualitative o miste. La natura dei caratteri osservati si riperquote sulla scelta della metrica o della misura di similarità da adottare.

Metodi gerarchici e metodi non gerarchici I metodi gerarchici generano dei raggruppamenti in numero crescente (algoritmi scissori) o decrescente (algoritmi aggregativi) a seconda che si inizi con un unico gruppo contenente tutte le unità, oppure con tanti gruppi, quante sono le unità. Analizzando l'albero di aggregazione o scissione, è possibile valutare quale sia la partizione ottimale (i criteri possono essere empirici o matematici).

I metodi non gerarchici sono utilizzati soprattutto se il numero delle osservazioni è molto elevato e presuppongono la conoscenza a priori del numero dei cluster. Stabilito quanti gruppi costituire, si individuano dei centri iniziali attorno ai quali si aggregano le unità. Successivamente, per ciascun gruppo, si calcola un nuovo centro e si effettuano nuove aggregazioni delle unità. I due passi vengono iterati fino ad ottenere la convergergenza.

Hard clustering e fuzzy clustering Quando le unità vengono assegnate in via definitiva ad un gruppo si fa uso di una logica duale in fase di classificazione, cioè si fa riferimento al principio di non contraddizione e a quello del terzo escluso: un'osservazione appartiene ad un gruppo, oppure non vi appartiene.

Come alternativa ai metodi duali (crisp), trovano impiego le tecniche di classificazione sfocata o fuzzy [8]. Esse consentono, attraverso l'uso di funzioni di appartenenza [92], di trattare efficacemente l'imprecisione insita nei dati, facendo in modo che ogni osservazione sia assegnata con diversa intensità a ciascuno dei cluster. L'uso della classificazione fuzzy può risultare molto efficace per tutti quei dati di *frontiera* che, in base alla distanza (o alla dissimilarità) considerata, risultano di difficile attribuzione ad un gruppo piuttosto che ad un altro. Elementi di base sugli insiemi fuzzy sono esposti nell'appendice A.

Metodi deterministici e metodi stocastici Se si considera la matrice dei dati $\mathbf{X}$ come un campione di I osservazioni indipendenti, provenienti da un numero K predefinito di sottogruppi della popolazione iniziale, se all'interno del campione ogni gruppo ha numerosità I_k, $I = \sum_{k=1}^{K} I_k$ e se la popolazione è caratterizzata da una distribuzione di probabilità del tipo:

$$P(\mathbf{x}) = \sum_{k=1}^{K} P_k(\mathbf{x}|\theta_k) P(k) \tag{2.1}$$

cioè se si è in presenza di una mistura finita di K distribuzioni di probabilità $P_k(\mathbf{x}|\theta_k)$, ciascuna con vettore di parametri θ_k e con proporzione sul totale della popolazione pari a $P(k)$, $\sum_{k=1}^{K} P(k) = 1$, allora si possono utilizzare tecniche di classificazione basate su funzioni di massima verosimiglianza [81].

Se invece non si ipotizza l'esistenza di un modello per i sottogruppi della popolazione da cui è stato estratto il campione, allora si possono utilizzare tecniche di cluster analysis non probabilistiche o (nel linguaggio del machine learning) non parametriche, basate sulla minimizzazione di funzioni di minimi quadrati, come ad esempio le k-medie [63].

2.2 Tecniche di classificazione fuzzy per dati categoriali incompleti

Nel presente capitolo, si pone l'attenzione sui metodi di fuzzy cluster analisys di tipo partizionale per dataset di variabili categoriali, sia con formalizzazione non probabilistica (conceptual k-means [74] , fuzzy-centroids o fuzzy k-populations [50, 51]), sia con formalizzazione probabilistica (grade of membership analysis [89], fuzzy k-partitions [87]) .

La preferenza nei confronti dei metodi di cluster analysis di tipo partizionale è dettata dalla necessità di effettuare analisi su matrici con un elevato numero di osservazioni.

L'uso di algoritmi basati sia su modelli stocastici che non stocastici è visto nell'ottica dell'approccio integrato dell'estrazione dell'informazione, come già evidenziato nella sezione 1.4.

La scelta di avvalersi di tecniche classificazione fuzzy anzichè crisp, permette di tenere in considerazione l'imprecisione nell'assegnazione delle osservazioni ai gruppi individuati, misurata attraverso il grado di appartenenza.

Partendo da algoritmi preesistenti, concepiti per matrici di dati completamente osservati, l'obiettivo è quello di adattarli a dataset incompleti, mutuando la tecnica *optimal completion strategy fuzzy c-means* [43], già sperimentata su dataset di variabili numeriche. Supposto che il meccanismo generatore dei valori mancanti sia ignorabile, il ricorso alla cluster analysis per imputare i valori mancanti è basato sull'ipotesi che esista un legame molto forte tra gruppi di osservazioni e che esso possa essere utilizzato per ottenere una ricostruzione dell'informazione il più possibile fedele all'originale.

Data la matrice $\mathbf{X} = \{x_{ij}\}$ di I osservazioni per J variabili nominali, tale che l'elemento $x_{ij} \in Dom(A_j) = \left\{a_{j1}, a_{j2}, ..., a_{jl}, ..., a_{jL_j}\right\}$, con $1 \leq l \leq L_j$, essendo L_j il numero di modalità relative all'attributo A_j, la classificazione delle osservazioni è fatta sulla matrice in forma disgiuntiva completa $\mathbf{Y} = \{y_{ijl}\}$ di I righe e $L = \sum_{j=1}^{J} L_j$ colonne, ottenuta da $\mathbf{X}$ con il seguente criterio:

$$y_{ijl} = \begin{cases} 0, & \text{se } x_{ij} \neq a_{jl} \\ 1, & \text{se } x_{ij} = a_{jl} \end{cases} \tag{2.2}$$

Gli elementi y_{ijl} sono soggetti ai seguenti vincoli:

$$\begin{array}{l} y_{ijl} \in \{0, 1\} \\ \sum_{l=1}^{L_j} y_{ijl} = 1 \\ \sum_{j=1}^{J} \sum_{l=1}^{L_j} y_{kjl} = J \\ \sum_{i=1}^{I} y_{ijl} \in [0, I] \end{array} \tag{2.3}$$

La matrice $\mathbf{Y}$ è data dalla giustapposizione di J matrici indicatrici $\mathbf{Y}_j = \{y_{ijl}\}$, ciascuna di dimensione $I \times L_j$, rappresentante la dicotomizzazione degli attributi della matrice $\mathbf{X}$: $\mathbf{Y} = [\mathbf{Y}_1 \vdots ... \vdots \mathbf{Y}_j \vdots ... \vdots \mathbf{Y}_J]$. Come per la $\mathbf{X}$, anche la $\mathbf{Y}$ può essere scomposta, data la matrice indicatrice dei valori mancanti $\mathbf{M} = \{m_{ij}\}$:

$$\begin{array}{l} \mathbf{Y}_{(com)} = \left\{\mathbf{y}'_i \in \mathbf{Y}, \forall i | \sum_{j=1}^{J} m_{ij} = 0\right\} \\ Y_{(oss)} = \{y_{ijl} \in \mathbf{Y}, \forall i, j, l | m_{ij} = 0\} \\ Y_{(mis)} = \{y_{ijl} \in \mathbf{Y}, \forall i, j, l | m_{ij} = 1\} \end{array}$$

con:

$$\mathbf{y}'_i = \left[y_{i11}, ..., y_{i1L_1}, ..., y_{ij1}, ..., y_{ijl}, ..., y_{ijL_j}, ..., y_{iJ1}, ..., y_{iJL_J}\right]$$

Il legame tra le I unità che costituiscono i vettori riga della matrice $\mathbf{Y}$ e i K gruppi in cui viene partizionato il dataset è espresso attraverso i valori della matrice $\mathbf{U} = \{u_{ik}\}$. Gli elementi u_{ik} sono soggetti alle seguenti restrizioni:

$$\begin{array}{l} u_{ik} \in [0, 1] \\ \sum_{k=1}^{K} u_{ik} = 1 \\ \sum_{i=1}^{I} u_{ik} \in [0, I] \end{array} \tag{2.4}$$

Nel contesto degli insiemi fuzzy, gli elementi u_{ik} della matrice $\mathbf{U}$ esprimono il grado di appartenenza delle unità ai cluster individuati, mentre nella formalizzazione probabilistica [9] essi sono considerati come le probabilità a posteriori che la $i-esima$ unità provenga dalla $k-esima$ popolazione avendo osservato i valori di i, cioè: $u_{ik} = P(k|\mathbf{y}'_i)$.

I K gruppi individuati sono descritti da un insieme di valori centrali rispetto alla metrica adottata. Tali valori sono contenuti nella matrice $\mathbf{C} = \{c_{kjl}\}$. Nei modelli non probabilistici, c_{kjl} esprime l'intensità con cui il $k-esimo$ centro (o centroide) assume la modalità l della $j-esima$ variabile; nei modelli stocastici rappresenta la probabilità per un elemento della $k-esima$ popolazione di possedere la modalità l della $j-esima$ variabile, cioè $c_{kjl} = P_k(y_{jl} = 1)$. Gli elementi c_{kjl} sono soggetti ai seguenti vincoli:

$$\begin{array}{l} c_{kjl} \in [0,1] \\ \sum_{l=1}^{L_j} c_{kjl} = 1 \\ \sum_{j=1}^{J} \sum_{l=1}^{L_j} c_{kjl} = J \\ \sum_{k=1}^{K} c_{kjl} \in [0,K] \end{array} \tag{2.5}$$

La fase dell'imputazione dei valori mancanti nella cluster analysis, presenta delle affinità con l'imputazione tramite matching con funzione di dissimilarità, già descritta nel paragrafo 1.3.3.5. Infatti, si vedrà più avanti che il donatore del valore per $y_{(mis)ijl} \in Y_{(mis)}$ è il centroide k più vicino all'osservazione i, con la vicinanza espressa dal grado di appartenenza u_{ik} più elevato, relativamente alla $i-esima$ unità.

2.3 Metodi non probabilistici

Obiettivo delle tecniche di fuzzy cluster analysis non probabilistiche per dati incompleti è l'individuazione delle stime di $\mathbf{U}$, $\mathbf{C}$ e $Y_{(mis)}$ che rendano minima la seguente funzione:

$$J_m\left(\mathbf{U}, \mathbf{C}, Y_{(mis)} | Y_{(oss)}\right) = \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m D\left(\mathbf{y}_i', \mathbf{c}_k'\right) \tag{2.6}$$

tenendo conto dei vincoli (2.4) e (2.5) con:

$$\mathbf{c}_k' = \left[c_{k11}, ..., c_{k1L_1}, ..., c_{kj1}, ..., c_{kjl}, ..., c_{kjL_j}, ..., c_{kJ1}, ..., c_{kJL_J}\right]$$

La relazione $D\left(\mathbf{y}_i', \mathbf{c}_k'\right)$ è una *distanza* (o metrica) se dato un insieme $\mathbb{E}$, l'applicazione $D : \mathbb{E}^L \times \mathbb{E}^L \longrightarrow \mathbb{R}^+$ possiede le seguenti proprietà:

$$D\left(\mathbf{y}_i', \mathbf{y}_{i'}'\right) = 0 \Leftrightarrow \mathbf{y}_i' = \mathbf{y}_{i'}' \tag{2.7}$$

$$D\left(\mathbf{y}_i', \mathbf{y}_{i'}'\right) = D\left(\mathbf{y}_{i'}', \mathbf{y}_i'\right) \tag{2.8}$$

$$D\left(\mathbf{y}_i', \mathbf{y}_{i'}'\right) \leq D\left(\mathbf{y}_i', \mathbf{c}_k'\right) + D\left(\mathbf{c}_k', \mathbf{y}_{i'}'\right), \forall \mathbf{y}_i', \mathbf{y}_{i'}', \mathbf{c}_k' \in \mathbb{R}^L \tag{2.9}$$

Se invece le proprietà possedute da $D(\mathbf{y}'_i, \mathbf{c}'_k)$ sono soltanto la (2.7) e la (2.8), allora si parlerà di *indice di dissimilarità* [75].

Il numero $m \in (1, +\infty)$ è un parametro di fuzzificazione, per il quale valgono i seguenti limiti [8]:

$$\begin{array}{l} \lim_{m\to+\infty} u_{ik} = \frac{1}{K}, \forall i, k \\ \lim_{m\to 1} u_{ik} \in [0, 1], \forall i, k \end{array} \tag{2.10}$$

Il primo dei due limiti in (2.10) stabilisce che per m crescente, il grado di assegnazione di ciascuna unità ai cluster è sempre più sfocato fino a diventare uguale per tutti i gruppi. Se invece m tende a uno, l'assegnazione delle unità ai cluster sarà sempre più netta, fino a convergere verso soluzioni di tipo hard clustering. Basi teoriche per una scelta ottimale di m non esistono, sebbene siano stati implementati algoritmi euristici per l'individuazione del valore di m più adatto rispetto ai dati di partenza [61].

L'ottimizzazione della funzione (2.6), non essendo individuabili soluzioni analitiche, può essere ottenuta mediante tre metodologie principali [1]:

- l'uso di reti neurali;
- l'uso di "evolutionary computation";
- l'ottimizzazione alternata.

Nel presente lavoro si farà uso dell'ottimizzazione alternata in tre passi, le cui proprietà sulla convergenza verso ottimi locali sono descritte da Hathaway e Bezdek [42].

Tra i metodi di classificazione fuzzy per dataset di variabili catergoriali non probabilistici, sono state considerate la variante fuzzy delle conceptual k-means [74] e le fuzzy k-popolations [50, 51].

2.3.1 Le fuzzy conceptual k-means

Nell'algoritmo delle conceptual k-means, proposto da Ralambondrainy nel 1995 per la classificazione di dataset di variabili miste, per misurare la distanza tra coppie di osservazioni rispetto a variabili categoriali è stata utilizzata la metrica del χ^2 [75]:

$$D(\mathbf{y}'_i, \mathbf{c}'_k) = \|\mathbf{y}'_i - \mathbf{c}'_k\|^2_{\mathbf{A}} = (\mathbf{y}'_i - \mathbf{c}'_k)\,\mathbf{A}\,(\mathbf{y}'_i - \mathbf{c}'_k)$$

dove $\mathbf{A} = \left\{I_{jl}^{-1}\right\}$ è una matrice diagonale in cui $I_{jl} = \sum_{i=1}^{I} y_{ijl}$ è il numero di osservazioni aventi la modalità l del carattere j. In sostanza, rispetto alla metrica euclidea generalmente utilizzata per le variabili continue, optando per

la metrica del χ^2 si decide di dare più "importanza" in termini di peso alle modalità meno frequenti. La funzione obiettivo (2.6) diventa quindi:

$$\begin{aligned} J_m\left(\mathbf{U}, \mathbf{C}, Y_{(mis)}|Y_{(oss)}\right) &= \sum_{i=1}^{I}\sum_{k=1}^{K} u_{ik}^m \left(\mathbf{y}_i' - \mathbf{c}_k'\right) \mathbf{A} \left(\mathbf{y}_i' - \mathbf{c}_k'\right) \\ &= \sum_{i=1}^{I}\sum_{k=1}^{K} u_{ik}^m \left\{ \sum_{j=1}^{J}\sum_{l=1}^{L_j} \left[\frac{\left(y_{ijl} - c_{kjl}\right)^2}{I_{jl}} \right] \right\} \end{aligned} \tag{2.11}$$

Le stime dei parametri si ottengono minimizzando la funzione (2.11) soggetta ai vincoli (2.3), (2.4) e (2.5). Essendo un problema di ottimizzazione vincolata, si può usare il metodo dei moltiplicatori di Lagrange:

$$\begin{aligned} \widetilde{J}_m\left(\mathbf{U}, \mathbf{C}, Y_{(mis)}, \Gamma, \Delta, \Lambda|Y_{(oss)}\right) &= \sum_{i=1}^{I}\sum_{k=1}^{K} u_{ik}^m \left\{ \sum_{j=1}^{J}\sum_{l=1}^{L_j} \left[\frac{\left(y_{ijl} - c_{kjl}\right)^2}{I_{jl}} \right] \right\} \\ &- \sum_{i=1}^{I} \delta_i \left(\sum_{k=1}^{K} u_{ik} - 1 \right) \\ &- \sum_{k=1}^{K}\sum_{j=1}^{J} \gamma_{kj} \left(\sum_{l=1}^{L_j} c_{kjl} - 1 \right) \\ &- \sum_{i=1}^{I}\sum_{j=1}^{J} \lambda_{ij} \left(\sum_{l=1}^{L_j} y_{ijl} - 1 \right) \end{aligned}$$

Il sistema di equazioni che deriva dall'annullamento delle derivate prime di $\widetilde{J}_m$ è il seguente:

$$\frac{\partial \widetilde{J}_m}{\partial u_{ik}} = m u_{ik}^{m-1} \left\{ \sum_{j=1}^{J} \sum_{l=1}^{L_j} \left[\frac{(y_{ijl} - c_{kjl})^2}{I_{jl}} \right] \right\} - \delta_i = 0 \tag{2.12}$$

$$\frac{\partial \widetilde{J}_m}{\partial c_{kjl}} = -2 \sum_{i=1}^{I} u_{ik}^m \left(\frac{y_{ijl} - c_{kjl}}{I_{jl}} \right) - \gamma_{kj} = 0 \tag{2.13}$$

$$\frac{\partial \widetilde{J}_m}{\partial y_{ijl}} = 2 \sum_{k=1}^{K} u_{ik}^m \left(\frac{y_{ijl} - c_{kjl}}{I_{jl}} \right) - \lambda_{ij} = 0 \tag{2.14}$$

$$\frac{\partial \widetilde{J}_m}{\partial \delta_i} = \sum_{k=1}^{K} u_{ik} - 1 = 0 \tag{2.15}$$

$$\frac{\partial \widetilde{J}_m}{\partial \gamma_{kj}} = \sum_{l=1}^{L_j} c_{kjl} - 1 = 0 \tag{2.16}$$

$$\frac{\partial \widetilde{J}_m}{\partial \lambda_{ij}} = \sum_{l=1}^{L_j} y_{ijl} - 1 = 0 \tag{2.17}$$

Risolvendo l'equazione (2.15) e sostituendo il valore ottenuto nella equazione (2.12), si perviene alla seguente formula per la stima degli elementi di $\mathbf{U}$:

$$\begin{aligned} \hat{u}_{ik} &= \frac{\|\mathbf{y}_i' - \mathbf{c}_k'\|_{\mathbf{A}}^{\frac{2}{1-m}}}{\sum_{k=1}^{K} \|\mathbf{y}_i' - \mathbf{c}_k'\|_{\mathbf{A}}^{\frac{2}{1-m}}} \\ &= \frac{\left\{ \sum_{j=1}^{J} \sum_{l=1}^{L_j} \left[\frac{(y_{ijl} - c_{kjl})^2}{I_{jl}} \right] \right\}^{\frac{1}{1-m}}}{\sum_{k=1}^{K} \left\{ \sum_{j=1}^{J} \sum_{l=1}^{L_j} \left[\frac{(y_{ijl} - c_{kjl})^2}{I_{jl}} \right] \right\}^{\frac{1}{1-m}}} \end{aligned} \tag{2.18}$$

Risolvendo l'equazione (2.16) e sostituendo il valore ottenuto nella equazione (2.13), si ottiene la seguente formula per la stima degli elementi di $\mathbf{C}$:

$$\hat{c}_{kjl} = \frac{\sum_{i=1}^{I} y_{ijl} u_{ik}^m}{\sum_{i=1}^{I} u_{ik}^m} \tag{2.19}$$

Infine, risolvendo l'equazione (2.17) e sostituendo il valore ottenuto nella equazione (2.14), si ottiene la seguente formula per la stima dei valori mancanti di $\mathbf{Y}$:

$$\hat{y}_{(mis)ijl} = \frac{\sum_{k=1}^{K} c_{kjl} u_{ik}^m}{\sum_{k=1}^{K} u_{ik}^m} \tag{2.20}$$

Al contrario dei dati osservati di $\mathbf{Y}$, le stime $\hat{y}_{(mis)ijl}$ possono assumere valori nell'intervallo $[0,1]$. L'imputazione degli elementi dell'insieme $X_{(mis)}$, avviene con la seguente regola:

$$x_{(mis)ij} = a_{jl},\ l : y_{(mis)ijl} = \max_{1 \leq l' \leq L_j} \left\{ y_{(mis)ijl'} \right\} \tag{2.21}$$

I passi che definiscono le fuzzy conceptual k-means (FCKM) per dati incompleti sono elencati nell'algoritmo 2.1.

Algorithm 2.1 FCKM per dati incompleti

1. Fissare i valori:
   (a) di m,
   (b) di $2 \leq K \leq I$,
   (c) della soglia di arresto ε,
   (d) del numero massimo di iterazioni ι;
2. porre $t = 0$;
3. inizializzare la matrice dei centri $\mathbf{C}^{(0)} = \left\{ c_{kjl}^{(0)} \right\}$;
4. inizializzare i valori di $Y_{(mis)}^{(0)} = \left\{ y_{(mis)ijl}^{(0)} \right\}$;
5. WHILE $\{ \left[\left| J_m^{(t-1)} - J_m^{(t)} \right| > \varepsilon \right]$ OR $[t < \iota] \}$:
   (a) Calcolare i valori della matrice $\mathbf{U}^{(t)} = \left\{ u_{ik}^{(t)} \right\}$ secondo la (2.18);
   (b) Aggiornare i valori della matrice dei centri $\mathbf{C}^{(t)} = \left\{ c_{kjl}^{(t)} \right\}$ secondo la (2.19);
   (c) Imputare i valori mancanti $Y_{(mis)}^{(t)} = \left\{ y_{(mis)ijl}^{(t)} \right\}$ secondo la 2.20;
   (d) calcolare $J_m^{(t)}(\mathbf{U}^{(t)}, \mathbf{C}^{(t)}, Y_{mis}^{(t)} | Y_{oss})$ secondo la 2.11;
   (e) $t = t + 1$;
6. Imputare i valori mancanti $X_{(mis)}$ secondo il criterio 2.21

2.3.2 Le fuzzy k-modes e le fuzzy k-populations

Il punto di partenza della metodologia dei fuzzy centroids o delle fuzzy k-populations (FKPOP) è l'algoritmo delle fuzzy k-modes introdotto da Huang e NG nel 1999 [45]. Tra il vettore dei valori osservati $\mathbf{x}_i'$ e $\mathbf{v}_k' = [v_{k1}, ..., v_{kj}, ..., v_{kJ}]$,

centroide del $k-esimo$ cluster tale che $v_{kj} \in Dom(A_j)$, si misura la prossimità con un'applicazione $D : \mathbb{E}^J \times \mathbb{E}^J \longrightarrow \mathbb{N}^+$ così definita:

$$D\left(\mathbf{x}_i', \mathbf{v}_k'\right) = \sum_{j=1}^{J} \mathbb{I}\left(x_{ij} \neq v_{kj}\right) \tag{2.22}$$

La funzione (2.22) assume valori compresi tra zero, se $\mathbf{v}_k' \equiv \mathbf{x}_i'$ e J, se nessuna delle modalità osservate in $\mathbf{x}_i'$ è presente in $\mathbf{v}_k'$. Essa possiede le proprietà (2.7), (2.8) e (2.9) ed è nota come distanza di Hamming. Nel campo della teoria dell'informazione, rappresenta il numero di posizioni, tra due stringhe di uguale lunghezza, in cui i simboli corrispondenti sono differenti [75]. Se si considerano le matrici in forma disgiuntiva completa $\mathbf{Y}$ e $\mathbf{C}$, associate rispettivamente ad $\mathbf{X}$ e $\mathbf{V} = [\mathbf{v}_1, ..., \mathbf{v}_k, ..., \mathbf{v}_K]'$, la (2.22) potrà essere così riscritta:

$$\begin{aligned} D\left(\mathbf{y}_i', \mathbf{c}_k'\right) &= \sum_{j=1}^{J} \sum_{l=1}^{L_j} \left(1 - y_{ijl}\right) c_{kjl} \\ &= \sum_{j=1}^{J} \sum_{l=1}^{L_j} c_{kjl} - \sum_{j=1}^{J} \sum_{l=1}^{L_j} y_{ijl} c_{kjl} \\ &= J - \sum_{j=1}^{J} \sum_{l=1}^{L_j} y_{ijl} c_{kjl} \end{aligned} \tag{2.23}$$

L'algoritmo delle fuzzy k-modes consiste nell'applicazione delle fuzzy k-medie a dataset di variabili categoriali. La funzione obiettivo è data da:

$$\begin{aligned} J_m\left(\mathbf{U}, \mathbf{C} | \mathbf{Y}\right) &= \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m \left[\sum_{j=1}^{J} \sum_{l=1}^{L_j} \left(1 - y_{ijl}\right) c_{kjl} \right] \\ &= \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m \left[J - \sum_{j=1}^{J} \sum_{l=1}^{L_j} y_{ijl} c_{kjl} \right] \\ &= J \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m - \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m \sum_{j=1}^{J} \sum_{l=1}^{L_j} y_{ijl} c_{kjl} \end{aligned} \tag{2.24}$$

e si minimizza tenendo conto dei vincoli (2.4) e del fatto che $c_{kjl} \in \{0, 1\}$, $\forall k, j, l$. Utilizzando i moltiplicatori di Lagrange, si perviene alla seguente formula per la stima degli elementi di $\mathbf{U}$:

$$\hat{u}_{ik} = \frac{D\left(\mathbf{y}_i, \mathbf{c}_k\right)^{\frac{1}{1-m}}}{\sum_{k=1}^{K} D\left(\mathbf{y}_i, \mathbf{c}_k\right)^{\frac{1}{1-m}}} = \frac{\left[\sum_{j=1}^{J}\sum_{l=1}^{L_j}\left(1-y_{ijl}\right)c_{kjl}\right]^{\frac{1}{1-m}}}{\sum_{k=1}^{K}\left[\sum_{j=1}^{J}\sum_{l=1}^{L_j}\left(1-y_{ijl}\right)c_{kjl}\right]^{\frac{1}{1-m}}} \tag{2.25}$$

Per quanto riguarda la stima di $\mathbf{C}$, fissate le u_{ik}, per minimizzare J_m occorrerà massimizzare il secondo termine del secondo membro della (2.24) e ciò si ottiene ponendo:

$$\hat{c}_{kjl} = \mathbb{I}\left(\sum_{i=1}^{I} y_{ijl}u_{ik}^m = \max_{1\leq l'\leq L_j}\left(\sum_{i=1}^{I} y_{ijl'}u_{ik}^m\right)\right) \tag{2.26}$$

La critica mossa alle fuzzy k-modes [50, 51], è data dal fatto che essendo $c_{kjl} \in \{0,1\}$, $\forall k, j, l$, viene individuata una categoria modale per la variabile j, non tenendo conto dell'eventuale informazione contenuta nelle categorie non modali del centroide k. Tutto ciò potrebbe comportare la classificazione errata di alcune osservazioni e quindi una maggiore probabilità per l' algoritmo di convergere verso minimi locali. Ad esempio, si consideri variabile A_j con $Dom(A_j) = \{''alto'', '' basso''\}$ e si supponga di avere osservato tre unità $\mathbf{x}_1'$, $\mathbf{x}_2'$, $\mathbf{x}_3'$, aventi rispettivamente grado di appartenenza al cluster k pari a $u_{1k} = 0,7$, $u_{2k} = 0,8$ e $u_{3k} = 0,15$ e valori della variabile j pari a $x_{1j} ='' alto''$, $x_{2j} ='' basso''$, $x_{3j} ='' alto''$. Applicando la formula 2.26:

$$\hat{c}_{kjl} = \mathbb{I}\left[(1\cdot 0,7^m + 0\cdot 0,8^m + 1\cdot 0,15^m) > (0\cdot 0,7^m + 1\cdot 0,8^m + 0\cdot 0,15^m)\right]$$

Il risultato della disuguaglianza dipende dal valore di m, infatti se $m = 1,0$, $\hat{c}_{kjl} = 1$ e $v_{cj} ='' alto''$, mentre per $m = 2,0$, $\hat{c}_{kjl} = 0$ e $v_{cj} = ''basso''$. In ogni caso, una delle due modalità avrà grado di appartenenza nullo e non potrà intervenire nel calcolo della degli elementi della matrice di appartenenza u_{ik} nella iterazione successiva. Per ovviare a tale problema, si può cercare di preservare l'incertezza assumendo che i centroidi siano fuzzy [50], cioè che valgano per le c_{kjl} l'insieme di vincoli (2.5), pur mantenendo la distanza (2.23). Tuttavia, poichè la funzione (2.24) è lineare in c_{kjl}, le soluzioni ottime sarebbero sempre quelle indicate nella (2.26) e per tale motivo, nelle fuzzy k-populations la stima proposta per le c_{kjl} è data, come per le FCKM, dalla (2.19). Nonostante questa variante, la (2.23) mantiene le caratteristiche di metrica, in quanto le proprietà (2.8) e (2.9) rimangono invariate, mentre la proprietà (2.7) si conserva anche nei dataset incompleti, essendo i valori imputati $\hat{y}_{(mis)ijl}$ binari. Si ha infatti che:

$$
\begin{aligned}
D\left(\mathbf{y}_i', \mathbf{y}_i'\right) &= \sum_{j=1}^{J}\sum_{l=1}^{L_j}\left(1-y_{ijl}\right)y_{ijl} \\
&= J-\sum_{j=1}^{J}\sum_{l=1}^{L_j}y_{ijl}^2 \\
&= \begin{cases} 0, & y_{ijl}\in\{0,1\} \\ \alpha>0, & y_{ijl}\in[0,1] \end{cases}
\end{aligned}
$$

Il problema si porrebbe qualora si decidesse di stimare le $y_{(mis)ijl}$ mediante la (2.20). Se si effettua la derivata della lagrangiana associata alla funzione (2.24) rispetto alle y_{ijl}, i valori che annullano tale derivata sono dati da:

$$
\hat{y}_{(mis)ijl} = I\left(\sum_{k=1}^{K}c_{kjl}u_{ik}^m = \max_{1\le l'\le L_j}\left(\sum_{k=1}^{K}c_{kjl'}u_{ik}^m\right)\right) \tag{2.27}
$$

$\hat{y}_{(mis)ijl}=\{0,1\}$ per cui la modalità a_{jl} da imputare in $x_{(mis)ij}$ si ottiene è direttamente , senza necessità di applicare la regola (2.21)[1].

La funzione obiettivo da minimizzare (2.6) è in questo caso:

$$
J\left(\mathbf{U},\mathbf{C},Y_{(mis)}|Y_{(oss)}\right) = \sum_{i=1}^{I}\sum_{k=1}^{K}u_{ik}^m\left[J-\sum_{j=1}^{J}\sum_{l=1}^{L_j}y_{ijl}c_{kjl}\right] \tag{2.28}
$$

I passi che caratterizzano le fuzzy k-modes e le fuzzy k-populations per dati incompleti sono descritti nell'algoritmo 2.2.

[1] Qualora si voglia mantenere l'incertezza anche sui valori da imputare, occorre modificare la metrica (2.23). Una possibilità è data dalla seguente generalizzazione: $D\left(\mathbf{y}_i', \mathbf{c}_k'\right) = J-\sum_{j=1}^{J}\sum_{l=1}^{L_j}\min\left(y_{ijl};c_{kjl}\right)$.

Algorithm 2.2 FKM e FKPOP per dati incompleti

1. Fissare i valori:
   (a) di m,
   (b) di $2 \leq K \leq I$,
   (c) della soglia di arresto ε,
   (d) del numero massimo di iterazioni ι;
2. porre $t = 0$;
3. inizializzare la matrice dei centri $\mathbf{C}^{(0)} = \left\{c_{kjl}^{(0)}\right\}$;
4. inizializzare i valori di $Y_{(mis)}^{(0)} = \left\{y_{(mis)ijl}^{(0)}\right\}$;
5. WHILE $\{\left[\left|J_m^{(t-1)} - J_m^{(t)}\right| > \varepsilon\right]$ OR $[t < \iota]\}$:
   (a) Calcolare i valori della matrice $\mathbf{U}^{(t)} = \left\{u_{ik}^{(t)}\right\}$secondo la (2.25);
   (b) Aggiornare i valori della matrice dei centri $\mathbf{C}^{(t)} = \left\{c_{kjl}^{(t)}\right\}$secondo la (2.26) nel caso delle FKM; oppure secondo la (2.19), nel caso delle FKPOP;
   (c) Imputare i valori mancanti $Y_{(mis)}^{(t)} = \left\{y_{(mis)ijl}^{(t)}\right\}$ secondo la (2.27);
   (d) calcolare $J_m^{(t)}(\mathbf{U}^{(t)}, \mathbf{C}^{(t)}, Y_{(mis)}^{(t)} | Y_{(oss)}^{(t)})$ secondo la (2.28);
   (e) $t = t + 1$;
6. Ricostruire la matrice $\mathbf{X}$ completa tramite la $Y_{(oss)}$ e la $\hat{Y}_{(mis)}$.

2.4 Metodi probabilistici

Se si ipotizza che la matrice $\mathbf{Y}$ sia un campione di I osservazioni indipendenti estratte casualmente da una popolazione divisa in K sottogruppi, ciascuno descritto da una distribuzione di probabilità multinomiale multivariata, la (2.1) diventa:

$$
\begin{aligned}
P(\mathbf{y}) &= \sum_{k=1}^{K} P(k)\, P_k(\mathbf{y}|\mathbf{c}'_k) \\
&= \sum_{k=1}^{K} P(k) \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} P_k(y_{jl}=1)^{y_{jl}} \right] \\
&= \sum_{k=1}^{K} P(k) \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{jl}} \right] \qquad (2.29)
\end{aligned}
$$

Obiettivo delle tecniche di classificazione probabilistiche è quello di trovare i parametri che massimizzino la seguente funzione di verosimiglianza:

$$
\begin{aligned}
L(\mathbf{U}, \mathbf{C}, Y_{(mis)}|Y_{(oss)}) &= \prod_{i=1}^{I} \left\{ \sum_{k=1}^{K} P(k|\mathbf{y}'_i) \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} P_k(y_{ijl}=1)^{y_{ijl}} \right] \right\} \\
&= \prod_{i=1}^{I} \left\{ \sum_{k=1}^{K} u_{ik} \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{ijl}} \right] \right\} \qquad (2.30)
\end{aligned}
$$

oppure, considerando il logaritmo naturale della 2.30:

$$
\begin{aligned}
l(\mathbf{U}, \mathbf{C}, Y_{(mis)}|Y_{(oss)}) &= \ln L(\mathbf{U}, \mathbf{C}, Y_{(mis)}|Y_{(oss)}) \\
&= \ln \prod_{i=1}^{I} \left\{ \sum_{k=1}^{K} u_{ik} \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{ijl}} \right] \right\} \\
&= \sum_{i=1}^{I} \ln \left\{ \sum_{k=1}^{K} u_{ik} \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{ijl}} \right] \right\} \qquad (2.31)
\end{aligned}
$$

La funzione (2.31) descrive quello che in letteratura è definito come *modello delle classi latenti* [18]. L'ottimizzazione di funzioni di log-verosimiglianza come la (2.31), può essere effettuata tramite algoritmo EM [67] [2] .

Il modello descritto dalla (2.31) è caratterizzato da una impostazione completamente probabilistica in cui gli elementi $u_{ik} = P(k|\mathbf{y}'_i)$ rappresentano la probabilità a posteriori per l'unità $i-esima$ di presentare il profilo $k-esimo$. Poichè, nel presente contesto, le u_{ik} sono considerate come gradi di apparte-

[2] Nel caso di variabili continue completamente osservate, l'ottimizzazione alternata in due passi [8], impiegata per minimizzare la funzione obiettivo in modelli non probabilistici come le fuzzy c-medie [78], sotto determinate ipotesi [3] risulta essere un'approssimazione dell'EM, usato per stimare i parametri delle misture di gaussiane.

nenza dell'osservazione i al profilo k, secondo la teoria degli insiemi fuzzy, si è deciso di utilizzare metodi di classificazione per dataset di variabili nominali con formalizzazione fuzzy/probabilistica, come la grade of membership analysis [89] e le fuzzy k-partitions [87].

2.4.1 La grade of membership analysis

Woodbury e Clive nel 1974 fornirono uno dei primi esempi di modello matematico *misto* [89]. Il modello da loro impiegato infatti, coniuga il grado di appartenenza fuzzy e le probabilità per variabili categoriali, dando origine ad un criterio composito "statistico-fuzzy"[8]. Il grado di appartenenza u_k alla popolazione k viene esplicitato nella definizione della probabilità di assumere la modalità l dell'attributo j:

$$\begin{aligned} P(y_{jl}=1) &= \frac{\sum_{k=1}^{K} P_k(y_{jl}=1)\,u_k}{\sum_{l=1}^{L_j}\sum_{k=1}^{K} P_k(y_{jl}=1)\,u_k} \\ &= \frac{\sum_{k=1}^{K} c_{kjl}u_k}{\sum_{l=1}^{L_j}\sum_{k=1}^{K} c_{kjl}u_k} \\ &= \frac{\sum_{k=1}^{K} c_{kjl}u_k}{\sum_{k=1}^{K} u_k \sum_{l=1}^{L_j} c_{kjl}} \\ &= \sum_{k=1}^{K} c_{kjl}u_k \end{aligned}$$

Assumendo l'ipotesi che il campione sia formato da I unità indipendenti, estratte da una popolazione con distribuzione di probabilità multinomiale multivariata data da:

$$\begin{aligned} P(\mathbf{y}) &= \prod_{j=1}^{J}\prod_{l=1}^{L_j} P(y_{jl}=1)^{y_{jl}} \\ &= \prod_{j=1}^{J}\prod_{l=1}^{L_j}\left(\sum_{k=1}^{K} u_k c_{kjl}\right)^{y_{jl}} \end{aligned}$$

ed estendendo l'applicazione della metodologia al caso di variabili categoriali non completamente osservate, l'obiettivo della grade of membership analysis (GOM) è la stima delle matrici $\mathbf{U}=\{u_{ik}\}$ e $\mathbf{C}=\{c_{kjl}\}$ e dell'insieme di valori $Y_{(mis)}$, tramite massimizzazione della seguente funzione di verosimiglianza:

$$
\begin{aligned}
L\left(\mathbf{U},\mathbf{C},Y_{(mis)}|Y_{(oss)}\right) &= \prod_{i=1}^{I}\prod_{j=1}^{J}\prod_{l=1}^{L_j} P\left(y_{ijl}=1\right)^{y_{ijl}} \\
&= \prod_{i=1}^{I}\prod_{j=1}^{J}\prod_{l=1}^{L_j}\left(\sum_{k=1}^{K} u_{ik}c_{kjl}\right)^{y_{ijl}}
\end{aligned}
$$

Dalla quale, considerando il logaritmo, si ottiene:

$$
l\left(\mathbf{U},\mathbf{C},Y_{(mis)}|Y_{(oss)}\right) = \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{l=1}^{L_j}\ln\left(\sum_{k=1}^{K} u_{ik}c_{kjl}\right)^{y_{ijl}} \tag{2.32}
$$

Per tenere conto dell'insieme di vincoli (2.3),(2.4) e(2.5), si applica alla (2.32) il metodo dei moltiplicatori di Lagrange:

$$
\begin{aligned}
\widetilde{l}\left(\mathbf{U},\mathbf{C},Y_{(mis)},\Gamma,\Delta,\Lambda|Y_{(oss)}\right) &= \sum_{i=1}^{I}\sum_{j=1}^{J}\sum_{l=1}^{L_j}\ln\left(\sum_{k=1}^{K} u_{ik}c_{kjl}\right)^{y_{ijl}} \\
&- \sum_{i=1}^{I}\delta_i\left(\sum_{k=1}^{K} u_{ik}-1\right) \\
&- \sum_{k=1}^{K}\sum_{j=1}^{J}\gamma_{kj}\left(\sum_{l=1}^{L_j} c_{kjl}-1\right) \\
&- \sum_{i=1}^{I}\sum_{j=1}^{J}\lambda_{ij}\left(\sum_{l=1}^{L_j} y_{ijl}-1\right)
\end{aligned}
$$

Il sistema di equazioni che scaturisce dall'annullamento delle derivate prime di $\widetilde{l}$ è il seguente:

$$\frac{\partial \widetilde{l}}{\partial u_{ik}} = \sum_{j=1}^{J}\sum_{l=1}^{L_j}\left(\frac{y_{ijl}c_{kjl}}{\sum_{k=1}^{K} u_{ik}c_{kjl}}\right) - \delta_i = 0 \tag{2.33}$$

$$\frac{\partial \widetilde{l}}{\partial c_{kjl}} = \sum_{i=1}^{I}\left(\frac{y_{ijl}u_{ik}}{\sum_{k=1}^{K} u_{ik}c_{kjl}}\right) - \gamma_{kj} = 0 \tag{2.34}$$

$$\frac{\partial \widetilde{l}}{\partial y_{ijl}} = \ln\left(\sum_{k=1}^{K} u_{ik}c_{kjl}\right) - \lambda_{ij} = 0 \tag{2.35}$$

$$\frac{\partial \widetilde{l}}{\partial \delta_i} = \sum_{k=1}^{K} u_{ik} - 1 = 0 \tag{2.36}$$

$$\frac{\partial \widetilde{l}}{\partial \gamma_{kj}} = \sum_{l=1}^{L_j} c_{kjl} - 1 = 0 \tag{2.37}$$

$$\frac{\partial \widetilde{l}}{\partial \lambda_{ij}} = \sum_{l=1}^{L_j} y_{ijl} - 1 = 0 \tag{2.38}$$

Risolvendo l'equazione (2.36) e sostituendo il valore ottenuto nella equazione (2.33), si perviene alla seguente formula:

$$\sum_{j=1}^{J}\sum_{l=1}^{L_j}\left(\frac{y_{ijl}c_{kjl}}{\sum_{k=1}^{K} u_{ik}c_{kjl}}\right) - \sum_{j=1}^{J}\sum_{l=1}^{L_j} y_{ijl} = 0$$

Risolvendo l'equazione (2.37) e sostituendo il valore ottenuto nella equazione (2.34), si ottiene il seguente risultato:

$$\sum_{i=1}^{I}\left(\frac{y_{ijl}u_{ik}}{\sum_{k=1}^{K} u_{ik}c_{kjl}}\right) - \sum_{i=1}^{I}\sum_{l=1}^{L_j}\left(\frac{y_{ijl}u_{ik}c_{kjl}}{\sum_{k=1}^{K} u_{ik}c_{kjl}}\right) = 0$$

Queste ultime due equazioni non possono essere risolte direttamente. Si può utilizzare il metodo iterativo del punto fisso[3] per approssimare le soluzioni, da cui si ricava rispettivamente:

$$u_{ik}^{(t)} = \frac{1}{J}\sum_{j=1}^{J}\sum_{l=1}^{L_j} y_{ijl}\frac{u_{ik}^{(t-1)}c_{kjl}^{(t-1)}}{\sum_{k=1}^{K} u_{ik}^{(t-1)}c_{kjl}^{(t-1)}} \tag{2.39}$$

[3]Si consideri un'equazione $z = f(z)$. Il primo passo è specificare un valore iniziale $z^{(0)}$.Successivamente, si calcola $z^{(t+1)} = f\left(z^{(t)}\right)$fino a quando $\left|z^{(t+1)} - z^{(t)}\right| < \varepsilon$, con ε molto piccolo.

$$c_{kjl}^{(t)} = \frac{\sum_{i=1}^{I} y_{ijl} \frac{u_{ik}^{(t)} c_{kjl}^{(t-1)}}{\sum_{k=1}^{K} u_{ik}^{(t)} c_{kjl}^{(t-1)}}}{\sum_{l=1}^{L_j} \sum_{i=1}^{I} y_{ijl} \frac{u_{ik}^{(t)} c_{kjl}^{(t-1)}}{\sum_{k=1}^{K} u_{ik}^{(t)} c_{kjl}^{(t-1)}}} \tag{2.40}$$

Per quanto riguarda la stima dei valori mancanti di $\mathbf{Y}$, la funzione (2.32) è lineare nelle y_{ijl}, per cui la soluzione ottima si otterrà nei punti estremi ed in particolare:

$$\hat{y}_{(mis)ijl} = I\left(\sum_{k=1}^{K} c_{kjl} u_{ik} = \max_{1 \leq l' \leq L_j} \left(\sum_{k=1}^{K} c_{kjl'} u_{ik}\right)\right) \tag{2.41}$$

per cui come per le FKPOP, ciò che si ottiene è direttamente la modalità a_{jl} da imputare in $x_{(mis)ij}$, senza necessità di utilizzare la relazione (2.21).

I passi che caratterizzano la grade of membership analysis per dati incompleti sono descritti nell'algoritmo 2.3.

Algorithm 2.3 GOM per dati incompleti

1. Fissare i valori:
   (a) di $2 \leq K \leq I$,
   (b) della soglia di arresto ε,
   (c) del numero massimo di iterazioni ι;
2. porre $t = 0$;
3. inizializzare la matrice dei centri $\mathbf{C}^{(0)} = \left\{c_{kjl}^{(0)}\right\}$;
4. inizializzare la matrice di appartenenza $\mathbf{U}^{(0)} = \left\{\frac{1}{K}\right\}$
5. inizializzare i valori di $Y_{(mis)}^{(0)} = \left\{y_{(mis)ijl}^{(0)}\right\}$;
6. WHILE $\{\left[\left|l^{(t-1)} - l^{(t)}\right| > \varepsilon\right]$ OR $[t < \iota]\}$:
   (a) Calcolare i valori della matrice $\mathbf{U}^{(t)} = \left\{u_{ik}^{(t)}\right\}$secondo la (2.39);
   (b) Aggiornare i valori della matrice dei centri $\mathbf{C}^{(t)} = \left\{c_{kjl}^{(t)}\right\}$secondo la (2.40);
   (c) Imputare i valori mancanti $Y_{(mis)}^{(t)} = \left\{y_{(mis)ijl}^{(t)}\right\}$ secondo la (2.41);
   (d) calcolare $l^{(t)}(\mathbf{U}^{(t)}, \mathbf{C}^{(t)}, Y_{(mis)}^{(t)}|Y_{(oss)})$ secondo la (2.32);
   (e) $t = t + 1$;
7. Ricostruire la matrice $\mathbf{X}$ completa tramite la $Y_{(oss)}$ e la $\hat{Y}_{(mis)}$.

Osservando la successione dei passi dell'algoritmo 2.3, si può notare che la massimizzazione della funzione (2.32), rispetto a $\mathbf{U}$ e $\mathbf{C}$, avviene dopo aver imputato i valori mancanti di $\mathbf{Y}$, a loro volta calcolati sulla base delle stime di $\mathbf{U}$ e $\mathbf{C}$ ottenute nell'iterazione precedente. La tecnica appena descritta corrisponde, nel caso di variabili numeriche, all'imputazione tramite EM con criterio *conditional mean imputation,* in dataset originati da misture finite di gaussiane [25].

2.4.2 Le fuzzy k-partitions

Nel 2008, Yang e altri autori [87], prendendo spunto dal GOM, propongono una nuova procedura per la classificazione di dataset di variabili categoriali, meno dispendiosa in termini di complessità computazionale. Supponendo di avere osservato un campione di I osservazioni indipendenti, se la matrice $\mathbf{Y}$

può essere partizionata in K gruppi $\mathbf{Y}^{(1)}, ..., \mathbf{Y}^{(k)}, ..., \mathbf{Y}^{(K)}$, essendo $\mathbf{Y}^{(k)} = \left\{ y_{ijl}^{(k)} \right\}$, si possono stimare le matrici $\mathbf{Z} = \{z_{ik}\}$ e $\mathbf{C} = \{c_{kjl}\}$, massimizzando la seguente verosimiglianza:

$$\begin{aligned} L(\mathbf{Z}, \mathbf{C}|\mathbf{Y}) &= \prod_{k=1}^{K} \prod_{i=1}^{I_k} \prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{ijl}^{(k)}} \\ &= \prod_{i=1}^{I} \prod_{k=1}^{K} \prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{z_{ik} y_{ijl}} \end{aligned}$$

con $z_{ik} = \mathbb{I}\left(\mathbf{y}_i' \in \mathbf{Y}^{(k)}\right)$. Sostituendo le funzioni indicatrici $z_{ik} \in \{0, 1\}$ con i gradi di appartenenza $u_{ik} \in [0, 1]$ e considerando la log-verosimiglianza, si ottiene:

$$\begin{aligned} l(\mathbf{U}, \mathbf{C}|\mathbf{Y}) &= \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik} \ln \left[\prod_{j=1}^{J} \prod_{l=1}^{L_j} (c_{kjl})^{y_{ijl}} \right] \\ &= \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik} \left[\sum_{j=1}^{J} \sum_{l=1}^{L_j} \ln (c_{kjl})^{y_{ijl}} \right] \end{aligned} \tag{2.42}$$

Tuttavia, essendo la (2.42) lineare in u_{ik}, le soluzioni ottime si avrebbero negli estremi dell'intervallo $[0, 1]$. Per tale ragione, è stato introdotto il parametro di fuzzyficazione $m \in (0, \infty)$:

$$l_m(\mathbf{U}, \mathbf{C}|\mathbf{Y}) = \sum_{i=1}^{I} \sum_{k=1}^{K} u_{ik}^m \left[\sum_{j=1}^{J} \sum_{l=1}^{L_j} \ln (c_{kjl})^{y_{ijl}} \right] \tag{2.43}$$

Per imputare i valori mancanti $Y_{(mis)}$ tramite la (2.43), occorre aggiungere ai vincoli (2.4), (2.5), le restrizioni sui valori y_{ijl}, date dalla (2.3). Utilizzando i moltiplicatori di Lagrange, si determina la nuova funzione obiettivo:

$$
\begin{aligned}
\tilde{l}_m\left(\mathbf{U},\mathbf{C},Y_{(mis)},\Gamma,\Delta,\Lambda|Y_{(oss)}\right) &= \sum_{i=1}^{I}\sum_{k=1}^{K} u_{ik}^m \left[\sum_{j=1}^{J}\sum_{l=1}^{L_j} \ln\left(c_{kjl}\right)^{y_{ijl}}\right] \\
&- \sum_{i=1}^{I} \delta_i \left(\sum_{k=1}^{K} u_{ik} - 1\right) \\
&- \sum_{k=1}^{K}\sum_{j=1}^{J} \gamma_{kj} \left(\sum_{l=1}^{L_j} c_{kjl} - 1\right) \\
&- \sum_{i=1}^{I}\sum_{j=1}^{J} \lambda_{ij} \left(\sum_{l=1}^{L_j} y_{ijl} - 1\right)
\end{aligned}
$$

Il sistema di equazioni che si ottiene dall'annullamento delle derivate prime di $\widetilde{l}_m$ è il seguente:

$$
\frac{\partial \widetilde{l}_m}{\partial u_{ik}} = m u_{ik}^{m-1} \sum_{j=1}^{J}\sum_{l=1}^{L_j} \ln\left(c_{kjl}\right)^{y_{ijl}} - \delta_i = 0 \tag{2.44}
$$

$$
\frac{\partial \widetilde{l}_m}{\partial c_{kjl}} = \sum_{i=1}^{I} \frac{y_{ijl} u_{ik}^m}{c_{kjl}} - \gamma_{kj} = 0 \tag{2.45}
$$

$$
\frac{\partial \widetilde{l}_m}{\partial y_{ijl}} = \sum_{k=1}^{K} u_{ik} \ln\left(c_{kjl}\right) - \lambda_{ij} = 0 \tag{2.46}
$$

$$
\frac{\partial \widetilde{l}_m}{\partial \delta_i} = \sum_{k=1}^{K} u_{ik} - 1 = 0 \tag{2.47}
$$

$$
\frac{\partial \widetilde{l}_m}{\partial \gamma_{kj}} = \sum_{l=1}^{L_j} c_{kjl} - 1 = 0 \tag{2.48}
$$

$$
\frac{\partial \widetilde{l}_m}{\partial \lambda_{ij}} = \sum_{l=1}^{L_j} y_{ijl} - 1 = 0 \tag{2.49}
$$

Risolvendo l'equazione (2.47) e sostituendo il valore ottenuto nella equazione (2.44), si perviene alla seguente formula per la stima degli elementi di $\mathbf{U}$:

$$
\hat{u}_{ik} = \frac{\left[\sum_{j=1}^{J}\sum_{l=1}^{L_j} \ln\left(c_{kjl}\right)^{y_{ijl}}\right]^{\frac{1}{1-m}}}{\sum_{k=1}^{K}\left[\sum_{j=1}^{J}\sum_{l=1}^{L_j} \ln\left(c_{kjl}\right)^{y_{ijl}}\right]^{\frac{1}{1-m}}} \tag{2.50}
$$

Risolvendo l'equazione (2.48) e sostituendo il valore ottenuto nella equazione (2.45), si ottiene la seguente formula per la stima degli elementi di $\mathbf{C}$:

$$\hat{c}_{kjl} = \frac{\sum_{i=1}^{I} y_{ijl} u_{ik}^{m}}{\sum_{i=1}^{I} u_{ik}^{m}} \tag{2.51}$$

Per quanto riguarda la stima dei valori mancanti di $\mathbf{Y}$, la funzione (2.43) è lineare nelle y_{ijl}, per cui la soluzione ottima si otterrà nei punti estremi ed in particolare:

$$\hat{y}_{(mis)ijl} = I\left(\sum_{k=1}^{K} u_{ik} \ln\left(c_{kjl}\right) = \max_{1 \leq l' \leq L_j}\left(\sum_{k=1}^{K} u_{ik} \ln\left(c_{kjl}\right)\right)\right) \tag{2.52}$$

per cui come per le FKPOP e per la GOM, ciò che si ottiene è direttamente la modalità a_{jl} da imputare in $x_{(mis)ij}$, senza applicare la regola (2.21).

I passi che caratterizzano le fuzzy k-partitions per dati incompleti sono descritti nell'algoritmo 2.4.

Algorithm 2.4 FKPAR per dati incompleti

1. Fissare i valori:
   (a) di m;
   (b) di $2 \leq K \leq I$,
   (c) della soglia di arresto ε,
   (d) del numero massimo di iterazioni ι;
2. porre $t = 0$;
3. inizializzare la matrice dei centri $\mathbf{C}^{(0)} = \left\{c_{kjl}^{(0)}\right\}$;
4. inizializzare i valori di $Y_{(mis)}^{(0)} = \left\{y_{(mis)ijl}^{(0)}\right\}$;
5. WHILE $\{\left[\left|l_m^{(t-1)} - l_m^{(t)}\right| > \varepsilon\right]$ OR $[t < \iota]\}$:
   (a) Calcolare i valori della matrice $\mathbf{U}^{(t)} = \left\{u_{ik}^{(t)}\right\}$secondo la (2.50);
   (b) Aggiornare i valori della matrice dei centri $\mathbf{C}^{(t)} = \left\{c_{kjl}^{(t)}\right\}$secondo la (2.51);
   (c) Imputare i valori mancanti $Y_{(mis)}^{(t)} = \left\{y_{(mis)ijl}^{(t)}\right\}$ secondo la (2.52);
   (d) calcolare $l_m^{(t)}(\mathbf{U}^{(t)}, \mathbf{C}^{(t)}, Y_{mis}^{(t)}|Y_{oss})$ secondo la (2.43);
   (e) $t = t + 1$;
6. Ricostruire la matrice $\mathbf{X}$ completa tramite la $Y_{(oss)}$ e la $Y_{(mis)}$.

Come nel caso del GOM, anche in questa circostanza l'algoritmo 2.4 può essere visto come un'applicazione dell'EM per l'imputazione dei valori mancanti in dataset generati da misture di multinomiali.

Capitolo 3

Le reti bayesiane

3.1 Definizione di rete bayesiana

Le reti bayesiane o sistemi esperti probabilistici [20, 47, 71] sono una classe di modelli statistico/probabilistici che attraverso la rappresentazione grafica di distribuzioni di probabilità consentono di evidenziare le relazioni di indipendenza condizionale tra le variabili casuali del dominio e di codificare le distribuzioni di probabilità congiunta. Ad una rete bayesiana (BN) sono associati meccanismi inferenziali basati sul teorema di Bayes che consentono la propagazione efficiente delle evidenze. Disponendo di tale network, può essere applicata l'inferenza probabilistica allo scopo di effettuare previsioni sui valori che manifesteranno alcune variabili, avendo osservato quelli delle altre. Per tale motivo, le reti bayesiane sono utilizzate in molti ambiti, quali ad esempio l'apprendimento automatico, la diagnosi medica, l'analisi causale, la statistica forense, la teoria dei giochi, i sistemi di controllo e monitoraggio, il controllo della produzione, la valutazione del rischio, la robotica, il data mining. Nel presente lavoro, esse verranno utilizzate per consentire l'imputazione dei valori mancanti in dataset di variabili nominali.

Una rete bayesiana $B\langle G, \theta\rangle$ si compone di due elementi :

1. Un **grafo diretto aciclico** (DAG) $G = (U, E)$, dove U è l'insieme dei *nodi* che corrispondono a variabili aleatorie ed E è l'insieme degli *archi orientati* che uniscono i nodi e che rappresentano i legami di dipendenza tra le variabili. Un arco $\mathbf{X} \leftarrow \mathbf{Y}$ nel grafo descrive una relazione genitore-figlio nella quale $\mathbf{X}$ è il figlio di $\mathbf{Y}$. L'insieme dei figli di $\mathbf{Y}$ si indica con $ch(\mathbf{Y})$, l'insieme dei genitori di $\mathbf{X}$ viene denotato come $pa(\mathbf{X})$. Un grafo diretto è aciclico quando, partendo da uno qualsiasi dei suoi nodi e percorrendo un cammino seguendo l'orientamento degli archi, non è possibile tornare ad un nodo già attraversato.

2. Un insieme di **parametri** θ associati a tabelle di probabilità condizionali (CPT), che descrivono la probabilità che $\mathbf{X}_j$ assuma il valore x_j, date tutte le possibili configurazioni dei suoi genitori $\Pi_{\mathbf{X}_j}$, dove $\Pi_{\mathbf{X}_j} = \{x : \mathbf{X} \in pa(\mathbf{X}_j)\}$ con $j = 1, ..., J$:

$$P(\mathbf{X}_j = x_j|\Pi_{\mathbf{X}_j}), \text{ oppure } P(x_j|\Pi_j), \text{ o ancora } P_{\Pi_j}(x_j)$$

Se un nodo non possiede genitori, i parametri saranno le probabilità marginali del nodo.

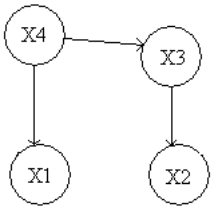

Figura 3.1: Esempio di DAG

Supponendo che il numero di nodi del grafo G sia pari a J, è possibile fornire una rappresentazione in forma matriciale dei DAG, mediante una matrice di adiacenza $\mathbf{C} = \{c_{jj'}\}$ di dimensione $J \times J$. Ciascun elemento $c_{jj'}$ nella matrice sarà così definito:

$$c_{jj'} = \begin{cases} 1 & \text{se } \mathbf{X}_j \in pa(\mathbf{X}_{j'}) \\ 0 & \text{altrimenti} \end{cases} \tag{3.1}$$

In questo modo, la $j' - esima$ colonna codificherà l'insieme $pa(\mathbf{X}_{j'})$, mentre la $j - esima$ riga codificherà l'insieme $ch(\mathbf{X}_j)$. Ad esempio, la matrice di adiacenza relativa al DAG della figura 3.1 sarà la seguente:

$$\mathbf{C} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Partendo dalla matrice di adiacenza, è possibile verificare se un grafo è effettivamente un DAG, ossia se è aciclico. Una tecnica per effettuare la verifica è l'*algoritmo di buona enumerazione* (GE) [62].

Algorithm 3.1 Algoritmo di buona enumerazione

1. WHILE $\{J > 0\}$:
   (a) data $\mathbf{C}$ di dimensione $J \times J$, si calcola $c_{\cdot j'} = \sum_{j=1}^{J} c_{jj'}\ \forall j' = 1, ..., J$;
   (b) $\nexists j' : c_{\cdot j'} = 0 \Rightarrow \mathbf{C}$ non è un DAG e l'algoritmo termina;
   (c) $\exists h \subseteq \{1, ..., J\}\,,\ |h| \geq 1 :\ c_{\cdot h} = 0$ allora:
      i. $|h| > 1 \Rightarrow$ si seleziona casualmente un elemento $l \in h$;
      ii. $|h| = 1 \Rightarrow l = h$;
      iii. $\mathbf{C} = \{\mathbf{C} \setminus \mathbf{c}'_l, \mathbf{C} \setminus \mathbf{c}_l\}$;
      iv. $J = J - 1$;
2. $\mathbf{C}$ è un DAG.

Poichè le reti bayesiane sono modelli di indipendenza condizionale, verrà fornita una breve descrizione di tale concetto. Sia $\mathbf{X}$ l'insieme di variabili in un dominio prescelto e sia P la distribuzione di probabilità congiunta di $\mathbf{X}$. Seguendo la notazione di Pearl [71], una relazione di indipendenza condizionale (CI) è denotata da $I(\mathbf{X}_1, \mathbf{X}_3, \mathbf{X}_2)$, dove $\mathbf{X}_1$,$\mathbf{X}_2$ e $\mathbf{X}_3$ sono sottoinsiemi disgiunti di variabili in $\mathbf{X}$. Tale notazione dice che $\mathbf{X}_1$ ed $\mathbf{X}_2$ sono indipendenti dato l'insieme condizionante $\mathbf{X}_3$. Formalmente:

$$P(\mathbf{X}_1|\mathbf{X}_2, \mathbf{X}_3) = P(\mathbf{X}_1|\mathbf{X}_3)\ con\ P(\mathbf{X}_2, \mathbf{X}_3) > 0$$

Una relazione CI è caratterizzata dal suo *ordine*, che è il numero di variabili nell'insieme condizionante $\mathbf{X}_3$. La probabilità di una generica realizzazione di tutte le J variabili del dominio, codificata da $B = \langle G, \theta \rangle$ potrà essere calcolata come:

$$P(\mathbf{x}) = P(x_1, ..., x_J) = \prod_{j=1}^{J} P(x_j|\Pi_j) \tag{3.2}$$

La fattorizzazione espressa dalla (3.2) consente, attraverso le relazioni di indipendenza condizionale, di *modularizzare* un problema complesso, quale è l'analisi delle relazioni tra tutte le variabili del dominio, in prolemi più piccoli e quindi più facilmente gestibili. Conoscendo la struttura della rete G e i valori del set di parametri θ, attraverso la propagazione delle evidenze è possibile prevedere il risultato di alcune variabili, espresso come la probabilità che si verifichi un evento, data l'osservazione delle altre.

In base alle definizioni date, è possibile rappresentare le tecniche di fuzzy cluster analysis probabilistiche esposte nella sezione 2.4 attraverso una rete bayesiana, in cui tutte le J variabili osservate sono indipendenti tra di loro,

dato un unico nodo genitore $\mathbf{W}$. Tale nodo è una variabile aleatoria nominale latente, con numero di modalità pari ai K gruppi e caratterizzata dalla distribuzione di probabilità $P(\mathbf{W} = k)$ oppure $P(k)$, $k = 1, ..., K$. Nel caso specifico, la (3.2) può essere riscritta come:

$$P(x_1, ..., x_J, k) = \prod_{j=1}^{J} P_k(x_j) P(k) \tag{3.3}$$

Marginalizzando la (3.3) rispetto a $\mathbf{W}$ si ha:

$$\begin{aligned} \sum_{k=1}^{K} P(x_1, ..., x_J, k) &= \sum_{k=1}^{K} \left[\prod_{j=1}^{J} P_k(x_j) P(k) \right] \\ P(x_1, ..., x_J) &= \sum_{k=1}^{K} P(k) \left[\prod_{j=1}^{J} P_k(x_j) \right] \end{aligned} \tag{3.4}$$

La (3.4) è una mistura di densità di probabilità, esattamente come la (2.1): viene definita modello *Naive Bayes* ed è rappresentato dalla rete in figura (3.2).

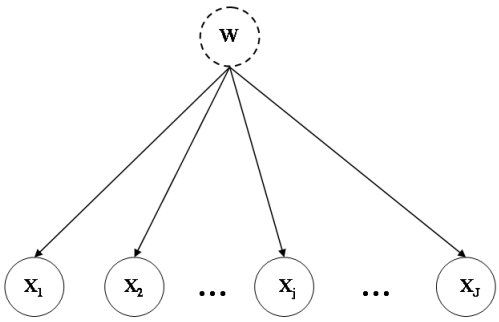

Figura 3.2: Naive Bayes

Le probabilità condizionali associate a tale modello, nel caso che le $\mathbf{X}_j$ siano tutte variabili categoriali, sono date da:

$$P_k(x_j = a_{jl}) = P_k(y_{jl} = 1) = c_{kjl}$$

In pratica, i modelli di cluster analysis probabilistici stimano i valori delle CPT, ipotizzando di conoscere la struttura di dipendenza probabilistica tra le variabili osservate e quella latente, oltre che il numero di modalità che questa può assumere. Esistono in letteratura tecniche di classificazione per dataset completi, basate su modelli Naive Bayes, come ad esempio *AutoClass* [13].

A differenza del precedente capitolo, d'ora in avanti non verranno fatte ipotesi sull'esistenza di nodi latenti e ci si concentrerà sulla ricerca di regole di associazione tra le variabili, supponendo quindi che esse non siano tra loro indipendenti, dato un unico nodo genitore. A tale scopo, è fondamentale focalizzare l'attenzione sui due aspetti necessari alla costruzione di una rete bayesiana:

1. la ricerca delle relazioni tra le variabili, attraverso l'individuazione della struttura;

2. la quantificazione delle relazioni individuate, espressa dalle tabelle di probabilità condizionale.

I criteri generalmente adoperati per costruire una BN sono due:

- si esplicitano tutte le conoscenze a disposizione, relativamente al fenomeno esaminato;

- si estrapolano da un campione di osservazioni le relazioni che legano ciascuna variabile alle altre, attraverso tecniche automatizzate di apprendimento della struttura *(structural learning)* e si quantificano le probabilità basando le stime dei parametri sui dati osservati;

Nel presente lavoro, verrà utilizzato il secondo criterio, per cui l'obiettivo sarà quello di costruire la rete $B\langle G, \theta\rangle$, avvalendosi delle informazioni contenute nella matrice dei dati $\mathbf{X}$, seppure incompleta, per poi effettuare le imputazioni dei valori mancanti, attraverso le relazioni individuate tra le variabili. In questo approccio, le due fasi della costruzione della BN non si attuano in momenti separati, in quanto ognuno dei due passi è strettamente legato all'altro: i parametri θ possono essere stimati soltanto dopo che la struttura G è stata individuata, ma a sua volta, la ricerca delle relazioni tra le variabili che caratterizzeranno la struttura vengono quantificate attraverso il calcolo delle stime di θ. Di conseguenza, i due step si alternano fino a convergere verso il risultato definitivo.

Nella prossima sezione, si concentrerà l'attenzione sulle tecniche per la costruzione della struttura G di una rete bayesiana per dataset con valori mancanti, mentre nella sezione 3.3 verrà spiegato come utilizzare le reti bayesiane per effetture imputazioni.

3.2 Structural learning per reti bayesiane

In letteratura, esistono due tipi principali di approccio all'apprendimento della struttura di una BN:

- I metodi basati sulla *massimizzazione di una funzione punteggio* (score+search approach) [16] che fanno uso di una metrica per valutare la struttura di un network candidato. Mediante tale metrica, viene impiegato un algoritmo di ricerca per trovare la struttura avente il miglior punteggio. In tal modo, il problema di apprendimento viene trasformato in un problema di ricerca.

- I metodi basati sull'*analisi delle indipendenze* [82], che fanno uso di opportuni test di indipendenza per individuare le relazioni tra i nodi della rete, ossia le variabili di interesse.

Entrambe le metodologie non sono esenti da limiti. Lo spazio di ricerca infatti è molto elevato, trattandosi di un problema di complessità esponenziale (NP-hard). Ne segue che da un lato, le tecniche di tipo score+search convergono spesso in ottimi locali, d'altro lato, le tecniche basate sull'analisi delle indipendenze sono rallentate dall'enorme numero di test da effettuare, oltre al fatto che questi ultimi possono risultare non accurati.

I lavori esistenti in letteratura riguardanti l'imputazione tramite reti bayesiane [24, 26, 23] sono basati sull'apprendimento della struttura tramite algoritmo PC [82]. Tuttavia, esso non nasce per l'analisi di dataset incompleti, per cui si è cercato di concentrare l'attenzione verso metodi alternativi, finalizzati all'individuazione di strutture anche in presenza di valori mancanti. Nei prossimi paragrafi, verranno introdotte due tecniche di tipo score+search, che saranno alternativamente utilizzate per l'apprendimento di una rete bayesiana. Si rende necessario quindi, individuare una funzione punteggio (score) da massimizzare.

Considerato un campione $\mathbf{X}$ di I osservazioni per J variabili categoriali estratto da una distribuzione multinomiale multivariata, un DAG G ed un insieme di parametri θ, si può calcolare la verosimiglianza di G e θ, dato $\mathbf{X}$:

$$L(G, \theta|\mathbf{X}) = \prod_{j=1}^{J} \prod_{l=1}^{L_j} \prod_{h=1}^{H_j} \theta_{jlh}^{I_{jlh}} \tag{3.5}$$

dove si indica con L_j il numero di attributi relativi alla $j-esima$ variabile, con H_j il numero delle possibili configurazioni dei genitori della $j-esima$ variabile e con I_{jlh} il numero di osservazioni che presentano la modalità l della $j-esima$ variabile e la configurazione h per i genitori della $j-esima$ variabile. Normalmente, si considera la log-verosimiglianza:

$$logL(G, \theta|\mathbf{X}) = \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{h=1}^{H_j} I_{jlh} \log \theta_{jlh} \tag{3.6}$$

Per massimizzare la verosimiglianza, si massimizzano per prima cosa i parametri θ_{jlh} e le stime nel caso di dataset completi sono date da:

$$\hat{\theta}_{jlh} = \frac{I_{jlh}}{I_{jl}} \tag{3.7}$$

essendo $I_{jl} = \sum_{h=1}^{H_j} I_{jlh}$ il numero di osservazioni che presentano la la modalità l della $j - esima$ variabile. sostituendo le stime nella 3.6 si ottiene:

$$logL(G, \hat{\theta}|\mathbf{X}) = \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{h=1}^{H_j} I_{jlh} \log \frac{I_{jlh}}{I_{jl}} \tag{3.8}$$

Il problema è che la verosimiglianza massima sarà quella che minimizza la distanza tra la BN ed il campione osservato, per cui la rete trovata sarà quella che riprodurrà esattamente il campione, utilizzando il maggior numero di connessioni possibili (overfitting). Per evitare situazioni di questo tipo, si può fare ricorso a penalizzazioni che riducano il valore delle verosimiglianze calcolate per modelli troppo complessi. In questo caso, la 3.8 diventa:

$$S(G, \hat{\theta}|\mathbf{X}) = \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{h=1}^{H_j} I_{jlh} \log \frac{I_{jlh}}{I_{jl}} - Pen(I)Dim(G) \tag{3.9}$$

essendo $Dim(G) = \sum_{j=1}^{J} L_j(H_j - 1)$ la dimensione del modello e $Pen(I)$ è una penalità tale che:

- se $Pen(I) = 1$ si ha l'Akaike's Information Criterion (AIC) [2];
- se$Pen(I) = \frac{1}{2} \log I$ si ha il Bayesian Information Criterion (BIC) [80].

Si è scelto di adottare il BIC come funzione da massimizzare per l'individuazione della struttura ottimale nelle tecniche di structural learning:

$$BIC(G, \hat{\theta}|\mathbf{X}) = \sum_{j=1}^{J} \sum_{l=1}^{L_j} \sum_{h=1}^{H_j} I_{jlh} \log \frac{I_{jlh}}{I_{jl}} - \frac{1}{2} \log I \sum_{j=1}^{J} L_j(H_j - 1) \tag{3.10}$$

3.2.1 Algoritmi greedy search

La prima tecnica di apprendimento descritta, appartiene ai metodi euristici di tipo greedy search (GS) [44]. Essa, è stata preferita rispetto ad altri metodi di tipo score+search, come l'*algoritmo k2* proposto da Cooper ed Herskovits nel 1991 [16], in quanto non ha bisogno di alcune restrizioni fondamentali per limitare il campo di applicazione, ossia:

- l'*ordinamento dei nodi,* in modo tale che il primo non abbia genitori, l'ultimo non abbia figli e quelli intermedi possano avere come genitori quelli che li precedono e come figli quelli che li seguono;

- la specificazione del *numero massimo di genitori* che un nodo può avere.

Queste limitazioni presuppongono delle conoscenze a priori che non sempre si può supporre di avere. L'idea alla base degli algoritmi GS è di partire da un DAG generato casualmente e di costruire iterativamente la soluzione seguendo un semplice criterio di espansione, che consiste nell'effettuare, ad ogni iterazione, la scelta più conveniente compatibilmente con i vincoli del problema. Qualunque scelta, una volta effettuata, non viene mai rimessa in discussione [69]. Tali metodi diventano più efficienti quando il criterio per la selezione del modello è *separabile*, ovvero può essere riscritto come prodotto (somma, nel caso di logaritmi) dei criteri adottati per ogni singola variabile. Il BIC è un criterio separabile, come si evince dalla 3.10. Molti dei più comuni metodi usati per le BN effettuano dei cambiamenti successivi negli archi della rete ed applicano la proprietà di separabilità per valutare ciascun cambiamento. I possibili cambiamenti sono facili da identificare. Per ciascuna coppia di nodi, se c'è un arco che li connette, esso può essere orientato in senso contrario o rimosso. Se invece non c'è alcun arco che connette le variabili, allora l'arco può essere aggiunto in entrambe le direzioni. Ovviamente, ciascun cambiamento deve essere soggetto al vincolo che il grafo risultante sia un DAG, ovvero non contenga cicli. In tal senso, si definisce E l'insieme dei cambiamenti plausibili nel grafo. $\Delta(e)$ denota il cambiamento nella funzione punteggio della rete risultante dalla modifica $e \in E$. Per individuare il sottoinsieme dei cambiamenti plausibili da quello di tutti i cambiamenti, si può utilizzare l'algoritmo 3.1. Il concetto di cambiamenti plausibili presenta delle affinità con quello della massa legale in problemi di ricerca di grafi decomponibili [34] e di identificabilità di grafi [84].

Dato un criterio separabile come il BIC, se un arco entrante è aggiunto o rimosso ad $\mathbf{X}_j$, per determinare $\Delta(e)$ occorre ricalcolare soltanto la componente:

$$\sum_{l=1}^{L_j} \sum_{h=1}^{H_j} I_{jlh} \log \frac{I_{jlh}}{I_{jl}} - \frac{L_j(H_j - 1)}{2} \log I$$

Nel caso in cui venga invertito l'orientamento dell'arco fra $\mathbf{X}_l$ ed $\mathbf{X}_{l'}$, la componente del BIC da ricalcolare per determinare $\Delta(e)$ sarà la seguente:

$$\sum_{j\in\{l,l'\}}\sum_{l=1}^{L_j}\sum_{h=1}^{H_j} I_{jlh}\log\frac{I_{jlh}}{I_{jl}} - \frac{1}{2}\log I \sum_{j\in\{l,l'\}} L_j(H_j-1)$$

Quando il criterio è separabile, si può evitare di ricalcolare tutte le variazioni nello score $\Delta(e)$ dopo ogni cambiamento. In particolare, se nè $\mathbf{X}_l$ e $\mathbf{X}_{l'}$, nè i loro genitori sono cambiati, $\Delta(e)$ rimane invariato per tutti gli e riguardanti quei nodi ed inoltre il netowork risultante sarà aciclico. Di seguito, vengono descritti i passi che caratterizzano l'algoritmo di tipo greedy search.

Algorithm 3.2 Greedy Search

1. Si genera casualmente un DAG iniziale $G^{(0)}$;
2. Si individua l'insieme di tutti i cambiamenti possibili $\mathcal{C}$ per $G^{(t)}$;
3. Si applica l'algoritmo 3.1 a $\mathcal{C}$, per individuare l'insieme $E \subseteq \mathcal{C}$ dei cambiamenti plausibili;
4. Si calcola $\Delta(e)$ per tutti i cambiamenti $e \in E$;
5. Se $\exists\Delta(e) > 0$:
   (a) si determina $G^{(t+1)}$ applicando il cambiamento e a $G^{(t)}$ per il quale $\Delta(e) > 0$ è massimo;
   (b) si torna al punto 2;
6. Se $\nexists\Delta(e) > 0$, l'algoritmo termina.

3.2.2 Algoritmi genetici

In questo paragrafo vengono introdotti gli algoritmi genetici (GA) [35], utilizzati nel contesto specifico come metodo di ricerca, alternativo al greedy search, per l'ottimizzazione della funzione di score. Gli algoritmi genetici e più in generale, quelli evolutivi, appartengono alla classe dei metodi euristici per l'individuazione di massimi (minimi) in situazioni di tipo NP-hard. Essi si basano sul principio Darwiniano della sopravvivenza grazie alla capacità di adattamento: gli individui di una popolazione si combinano per generare nuovi individui in generazioni successive ed il processo evolutivo tende a selezionare (far sopravvivere) solo gli individui più idonei. Facendo corrispondere a ciascun individuo una soluzione ed associando a ciascuna soluzione un valore di fitness, questi algoritmi cercano di simulare il processo evolutivo, provando a far sopravvivere solo le soluzioni di qualità elevata [69].

Data una popolazione iniziale di possibili soluzioni del problema, generata

casualmente, ogni soluzione è codificata da una stringa (*cromosoma*), composta da un insieme di celle (*geni*) le cui realizzazioni (*alleli*) possono essere di tipo binario o reale.

Per ogni cromosoma è calcolato un valore della funzione di *fitness* f.

Da una iterazione (*generazione*) all'altra, si compiono le seguenti operazioni:

- si effettua una *selezione* dei cromosomi con probabilità $P_s \propto f$;
- se nella popolazione della generazione successiva si inseriscono direttamente gli individui aventi $max(f)$ si ha l'*elitismo*;
- si effettua il *cross-over* tra due individui con probabilità P_c, scambiando tra di essi una porzione della stringa;
- in un individuo si opera una *mutazione* con probabilità P_m se il valore di un singolo gene viene alterato.

In letteratura, esistono diversi metodi che fanno uso degli algoritmi genetici o evolutivi per individuare il grafo di indipendenza condizionale [72, 76] oppure il DAG [53, 90], che meglio si adattano ad un dataset $\mathbf{X}$ di I osservazioni per J variabili, anche in presenza di dati incompleti [70, 91].

La procedura proposta nel presente lavoro, contiene degli aspetti innovativi rispetto a quella descritta in [53]. Infatti, si è volutamente evitato di stabilire a priori un ordinamento dei nodi, lasciando all'algoritmo la possibilità di cercare tra un insieme più ampio di strutture ammissibili. Per rendere possibile tutto ciò è stato, tuttavia, necessario implementare una funzione di fitness che fosse in grado di penalizzare tutte le strutture non DAG. Il test sulla presenza di cicli all'interno del grafo viene effettuato tramite l'algoritmo 3.1. Inoltre, come verrà spiegato in seguito 3.2.3, l'algoritmo sarà messo in condizione di operare in presenza di dati incompleti.

Considerata la matrice di adiacenza $\mathbf{C}$, definita dalla 3.1, i cromosomi avranno lunghezza pari a $J(J-1)$, cioè vengono considerate tutte le celle ad esclusione di quella sulla diagonale principale e per ogni individuo, la stringa sarà ordinata inserendo dapprima gli elementi al di sotto della diagonale principale e poi quelli restanti: $c_{21}c_{31}c_{32}...c_{J1}...c_{J(J-1)}c_{12}...c_{1J}c_{23}c_{2J}...c_{(J-1)J}$.

Ad esempio, considerato il DAG in figura 3.1, la stringa risultante è:

c_{21}	c_{31}	c_{32}	c_{41}	c_{42}	c_{43}	c_{12}	c_{13}	c_{14}	c_{23}	c_{24}	c_{34}
0	0	1	1	0	1	0	0	0	0	0	0

Avendo stabilito di utilizzare come funzione di fitness il BIC, l'algoritmo genetico per l'individuazione della struttura di una BN è sintetizzato nel seguente prospetto:

Algorithm 3.3 Algoritmi genetici

1. Si stabilisce la dimensione della popolazione dei DAG m, la probabilità di cross-over P_c, la probabilità di mutazione P_m, il numero di generazioni g;
2. si genera la popolazione iniziale dei grafi $\mathcal{G}^{(0)}$;
3. si applica l'algoritmo 3.1 per ogni elemento $G_i^{(0)} \in \mathcal{G}^{(0)}$ $i = 1, ..., m$, per individuare l'insieme $\mathcal{D}^{(0)} \subseteq \mathcal{G}^{(0)}$ degli individui ammissibili;
4. per ogni elemento $G_i^{(0)} \in \mathcal{G}^0$ $i = 1, ..., m$:
   (a) $G_i^{(0)} \in \mathcal{D}^{(0)} \Rightarrow$ si calcola $f_i = BIC(G_i, \hat{\theta}|\mathbf{X})$;
   (b) $G_i^{(0)} \notin \mathcal{D}^{(0)} \Rightarrow$ si pone $f_i = \epsilon$, con ε piccolo a piacere (ad es. $\epsilon = 1e - 100$);
5. si seleziona l'individuo h con $f_h = \max_i(f_i)$ (*elitismo*);
6. si effettua la *selezione* di $2k > m - 1$ individui con probabilità $P_{si} \propto f_i$;
7. si formano k coppie con gli individui selezionati;
8. per ogni coppia, si effettua il *cross-over* con probabilità P_c generando $2k$ nuovi individui;
9. per ognuno dei $n(n-1)$ geni dei $2k$ nuovi individui, si applica la *mutazione* con probabilità P_m;
10. si forma la nuova popolazione $\mathcal{G}^{(t)}$ con gli $m - 1$ nuovi individui aventi la fitness più elevata, insieme all'individuo di elite;
11. se $t < g$ si torna al punto 3, altrimenti l'algoritmo termina.

3.2.3 Structural learning per dataset incompleti

Nei precedenti paragrafi sono stati descritti due metodi per individuare la struttura G di una rete bayesiana, a partire da un dataset $\mathbf{X}$ di unità completamente osservate, attraverso l'ottimizzazione della funzione (3.10). Se però il dataset $\mathbf{X}$ è parzialmente incompleto, tale che $X_{(mis)}$ è l'insieme dei dati mancanti e $X_{(oss)}$ quello dei dati osservati, le stime di massima verosimiglianza dell'insieme dei parametri θ, non possono essere calcolate con la (3.7). Inoltre, la metrica adottata non gode più della proprietà della separabilità, ossia

non può essere scomposta direttamente e tale proprietà risulta fondamentale, in termini di contenimento della complessità computazionale, nell'applicazione dell'algoritmo 3.2.

Supponendo di avere individuato una struttura G, si può fare ricorso all'algoritmo EM per la stima dei parametri della rete [56]. L'uso dell'EM, in combinazione dei metodi di score+search viene definito *Structural Expectation-Maximization* (SEM)[31]. In questo caso, occorrerà massimizzare il valore atteso della funzione (3.10), che per una particolare stima $B\left\langle \hat{G}, \hat{\theta} \right\rangle$ è:

$$\begin{aligned} Q(G,\theta|\hat{G},\hat{\theta}) &= \mathbb{E}_{\hat{G},\hat{\theta}}\left[BIC(G,\theta|X_{(oss)})\right] \\ &= \sum_{j=1}^{J}\sum_{l=1}^{L_j}\sum_{h=1}^{H_j}\hat{I}_{jlh}\log\hat{\theta}_{jlh} - \frac{1}{2}\log I\sum_{j=1}^{J}L_j(H_j-1) \\ &= \sum_{j=1}^{J}\sum_{l=1}^{L_j}\sum_{h=1}^{H_j}\hat{I}_{jlh}\log\frac{\hat{I}_{jlh}}{\hat{I}_{jl}} - \frac{1}{2}\log I\sum_{j=1}^{J}L_j(H_j-1) \end{aligned}$$

dove le stime:

$$\hat{\theta}_{jlh} = \frac{\hat{I}_{jlh}}{\hat{I}_{jl}} = \frac{\mathbb{E}_{\hat{G}}\left[I_{jlh}|X_{(oss)}\right]}{\mathbb{E}_{\hat{G}}\left[I_{jl}|X_{(oss)}\right]}$$

sono ottenute applicando l'EM per la stima dei parametri, se la coppia $\{\mathbf{x}_j,\Pi_j\}$ non è completamente misurata. L'algoritmo SEM è descritto nel seguente prospetto:

Algorithm 3.4 Structural EM

1. Si fissa un valore ε
2. Si generano $G^{(0)}$ e $\theta^{(0)}$ casualmente
3. WHILE $\{|Q(G^{(t+1)},\theta^{(t+1)}|G^{(t)},\theta^{(t)}) - Q(G^{(t)},\theta^{(t)}|G^{(t)},\theta^{(t)})| > \varepsilon\}$:
   (a) passo E: si calcola $Q(G,\theta|G^{(t)},\theta^{(t)})$
   (b) passo M:
      i. si trova il modello $G^{(t+1)} = \arg\max_{G} Q(G,\theta^{(t)}|G^{(t)},\theta^{(t)})$ mediante gli algoritmi 3.2 oppure 3.3
      ii. si trova il set di parametri $\theta^{(t+1)} = \arg\max_{\theta} Q(G^{(t+1)},\theta|G^{(t)},\theta^{(t)})$

3.3 Imputazione dei valori mancanti tramite reti Bayesiane

Dopo aver definito alcune metodologie per l'apprendimento della struttura G di una rete bayesiana ed aver stimato i parametri θ delle distribuzioni di probabilità condizionali per tutti i nodi della rete, diviene possibile mettere a frutto l'informazione aquisita per tentare di imputare i valori mancanti nel dataset parzialmente incompleto.

Le reti bayesiane, in quanto strumento utilizzato per l'individuazione e la stima delle relazioni congiunte tra variabili, si candidano come metodologia indicata per l'imputazione dei valori mancanti, mirando ad assicurare la consistenza dei valori imputati in termini preservazione delle relazioni statistiche tra le variabili (*consistenza statistica*) e di preservazione dei vincoli logici tra i dati (*consistenza logica*). Per loro stessa natura infatti, attraverso l'informazione campionaria e sotto l'assunzione di ignorabilità del meccanismo generatore delle mancate risposte, esse tendono a ricostruire il processo generatore dai dati.

Di seguito, vengono descritte le due tecniche di imputazione tramite BN utilizzate nel presente lavoro: imputazioni tramite nodi genitori [23], imputazioni tramite Markov blanket [24, 26].

3.3.1 Imputazione tramite nodi genitori

Per ciascun record del del dataset, viene individuato l'insieme di variabili che presentano dei dati mancante. Per ogni variabile missing, il valore da imputare viene estratto casualmente, tenendo conto della tabella di probabilità condizionata alle realizzazioni dei nodi genitori. I valori mancanti dei nodi radice, ovvero di quelle variabili che non hanno genitori, vengono estratti campionando dalla distribuzione marginale della stessa variabile, quindi senza condizionamenti rispetto ad altri nodi. L'ordine con cui viene effettuata l'imputazione all'interno di ciascun record è dato dalla struttura G individuata con uno degli algoritmi 3.2 o 3.3. La metodologia originaria di imputazione si avvale, per l'individuazione della struttura della rete, dell'algoritmo PC [82]. Quest'ultimo, dopo aver individuato il grafo di indipendenza condizionale, i cui archi non sono orientati, necessita come input di un ordinamento dei nodi al fine di stabilire le relazioni di dipendenza probabilistica. Il criterio adottato dagli autori è stato quello di ordinare le variabili in base alla numerosità delle realizzazioni disponibili, per cui, nell'ambito dello stesso record viene imputata

prima la variabile che complessivamente registra il minore numero di missing e poi, a seguire, le altre.

In questo contesto, avendo introdotto metodi che non necessitano di indicazioni relativamente all'ordinamento dei nodi, la sequenza delle imputazioni è condizionata dalla struttura trovata.

3.3.2 Imputazione tramite Markov blanket

L'imputazione tramite Markov blanket differisce dalla tecnica precedentemente proposta perchè il valore da imputare nella variabile missing non dipende soltanto alla distribuzione di probabilità condizionata ai nodi genitori, bensì al suo Markov blanket (MB). Dato un indieme di variabili casuali $\mathbf{X}$ e considerato un nodo $\mathbf{X}_j \in \mathbf{X}$, il suo MB $Mb(\mathbf{X}_j)$ sarà dato dall'insieme di variabili tali che $I(\mathbf{X}_j, Mb(\mathbf{X}_j), \mathbf{X} \setminus Mb(\mathbf{X}_j))$. In particolare, il MB di un nodo è dato dai nodi genitori, dai nodi figli e dai nodi genitori dei figli. Il MB può essere definito anche per un insieme di nodi $\mathbf{X_A}$ed è dato da $\underset{j \in A}{\cup} Mb(\mathbf{X}_j) \setminus \mathbf{A}$. Ad esempio, considerata la figura 3.3, $Mb(\mathbf{X}_4) = \{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_7, \mathbf{X}_5\}$, $Mb(\mathbf{X}_5) = \{\mathbf{X}_3, \mathbf{X}_7, \mathbf{X}_8, \mathbf{X}_4, \mathbf{X}_6\}$, $Mb(\mathbf{X}_4, \mathbf{X}_5) = \{\mathbf{X}_1, \mathbf{X}_2, \mathbf{X}_3, \mathbf{X}_6, \mathbf{X}_7, \mathbf{X}_8\}$. Considerato un record del dataset con un insieme $\mathbf{A}$ di variabili i cui valori sono missing, tali valori saranno imputati secondo la distribuzione di probabilità congiunta di $\mathbf{X_A}$dato il $Mb(\mathbf{X_A})$. Anche in questa circostanza, rispetto alla metodologia originale, la decisione di introdurre metodi alternativi rispetto all'algoritmo PC per l'apprendimento della struttura G, ha comportato che il criterio di ordinamento dei nodi non sia più dettato della quota di di dati mancanti per ciascuna variabile, bensì possa variare a seconda dell'output della procedura di apprendimento.

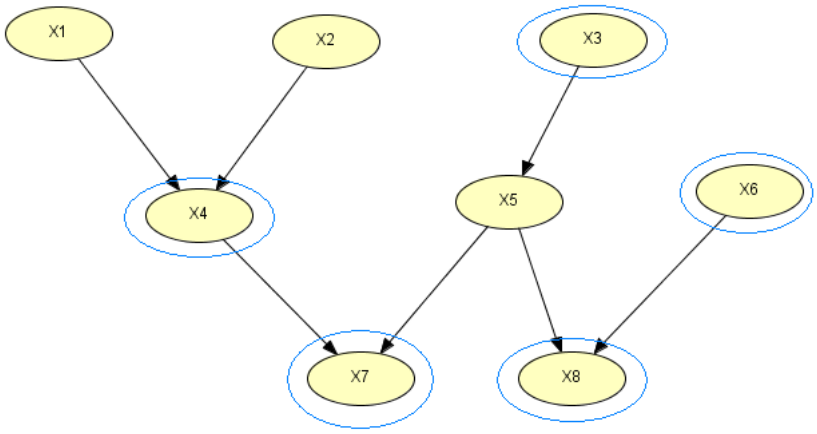

Figura 3.3: Il markov blanket di X_5

Capitolo 4

Applicazioni

Nel presente capitolo vengono presentate alcune applicazioni dei metodi descritti nei capitoli 2 e 3. L'intento è quello di misurarne la capacità di imputazione, attraverso indicatori appositi, già esistenti in letteratura [11]. Le prestazioni dei metodi di cluster analysis e di quelli basati sulle reti bayesiane sono valutate dapprima separatamente, rendendo così possibile analizzare le qualità specifiche di ciascuna metodologia e poi congiuntamente.

In particolare, per quanto riguarda la cluster analysis sono stati scelti dei dataset "preclassificati", ovvero matrici di $J+1$ colonne, essendo una di esse la variabile relativa alla classe di appartenenza. In questo modo è possibile monitorare la validità della classificazione ottenuta in condizioni di incompletezza dei dati.

I dataset utilizzati per l'imputazione con le reti bayesiane sono stati generati da network preesistenti, mediante il software Hugin [64]. Tale scelta consente di valutare le tecniche di structural learning in base alla precisione nella individuazione delle associazioni tra le variabili.

Tutti i dataset sono stati "sporcati" eliminando, con meccanismo di tipo MCAR, una quota crescente di realizzazioni, generando 5 matrici con una percentuale complessiva di missing pari rispettivamente al 10%, 20%, 30%, 40% e 50% . La selezione casuale dei valori mancanti è stata effettuata tenendo conto dei seguenti vincoli:

1. ogni vettore riga deve presentare almeno un valore non missing;

2. ogni vettore colonna deve presentare almeno un valore non missing;

3. le modalità appartenenti al dominio di una variabile devono essere presenti almeno una volta.

Quest'ultimo vincolo è dettato da necessità di tipo computazionale: esso consente alle procedure implementate di individuare l'esatto numero di catego-

rie di ogni variabile direttamente nel vettore delle replicazioni, senza doverlo specificare a parte.

Tutte le metodologie utilizzate sono state implementate nel linguaggio proprio del package statistico **R** [73] e per ciascun dataset considerato sono stati calcolati gli indicatori di performance come medie su 100 simulazioni.

4.1 Criteri per la valutazione delle imputazioni

La qualità dell'imputazione, avendo a disposizione il dataset completo, si può misurare quantificando la discrepanza tra i dati stimati e quelli veri. In presenza di dataset di variabili categoriali, obiettivo principale dei test è quello di valutare le seguenti proprietà [11]:

1. la capacità di preservare i valori originali (*predictive accuracy*);
2. la capacità di preservare la distribuzione originale (*distributional accuracy*).

Per una singola variabile, è possibile misurare la proprietà 1 mediante il seguente indicatore [11]:

$$D_j = \frac{\sum_{i=1}^{I} \mathbb{I}(\hat{x}_{ij} \neq x_{ij}) m_{ij}}{\sum_{i=1}^{I} m_{ij}} \tag{4.1}$$

essendo m_{ij} definito nella (1.1), $\hat{x}_{ij}$ la modalità della variabile j imputata per l'osservazione i e x_{ij} quella originale. D_j rappresenta la *quota di imputazioni errate* per X_j ed assume valore pari a zero, se tutti i valori sono coincidenti con quelli veri, mentre è pari a uno se tutte le imputazioni sono errate. La quota totale di imputazioni errate per l'intero dataset è data da:

$$D = \frac{\sum_{j=1}^{J} \sum_{i=1}^{I} \mathbb{I}(\hat{x}_{ij} \neq x_{ij}) m_{ij}}{\sum_{j=1}^{J} \sum_{i=1}^{I} m_{ij}} \tag{4.2}$$

Anche l'indicatore (4.2) varia tra zero se le imputazioni sono tutte esatte ed uno se le imputazioni sono tutte errate. La proprietà 2 può essere misurata su una singola variabile, mediante il seguente *indice di consistenza* [23]:

$$\Delta_j = \frac{1}{2} \left(\frac{\sum_{i=1}^{I} \sum_{l=1}^{L_j} |\hat{y}_{ijl} - y_{ijl}| \, m_{ij}}{\sum_{i=1}^{I} m_{ij}} \right) = \frac{1}{2} \sum_{l=1}^{L_j} \left| \frac{I_{(mis)jl}}{I_{(mis)j}} - \frac{\hat{I}_{(mis)jl}}{\hat{I}_{(mis)j}} \right| \tag{4.3}$$

essendo $I_{(mis)jl}$ il numero di osservazioni con valore mancante per la variabile j che presentano la modalità l nel dataset completamente osservato,

$\hat{I}_{(mis)jl}$ è il numero di osservazioni con valore mancante per la variabile j per le quali è stata stimata la modalità l mentre $I_{(mis)j}$ è il numero totale di unità non osservate rispetto alla variabile j. y_{ijl} è un numero binario definito nella (2.2) ed $\hat{y}_{ijl}$ la sua stima. L'indice di consistenza calcolato sulla distribuzione congiunta di tutte le variabili del dataset è dato da:

$$\Delta = \frac{1}{2}\sum_{h=1}^{H}\left|\frac{I_{(mis)h}}{I_{(mis)}} - \frac{\hat{I}_{(mis)h}}{I_{(mis)}}\right| \tag{4.4}$$

$I_{(mis)h}$ è il numero di volte che si presenta il profilo h nel dataset completo, relativamente ai casi mancanti del dataset osservato, $\hat{I}_{(mis)h}$ è la stima del numero di volte che si presenta il profilo h nel dataset imputato, sempre relativamente ai casi mancanti del dataset osservato, mentre $h = 1, ..., H$ e $H = \prod_{j=1}^{J} L_j$, ovvero il numero di profili possibili, dato dal prodotto cartesiano tra gli attributi che le J variabili possono assumere. Infine, $I_{(mis)} = \sum_{i=1}^{I} \mathbb{I}\left(\sum_{j=1}^{J} m_{ij} > 0\right)$ è il totale delle osservazioni che presentano almeno un valore mancante.

Quando il numero di variabili e, a loro volta, delle modalità che le caratterizzano è molto elevato, l'indice (4.4) diviene estremamente complicato da calcolare, a causa dell'elevato numero di profili che si verrebbero a configurare. In alternativa, si può utilizzare come indice di consistenza la media rispetto alle J variabili dell' indici (4.3):

$$\hat{\Delta} = \frac{1}{2J}\sum_{j=1}^{J}\left(\frac{\sum_{i=1}^{I}\sum_{l=1}^{L_j}|\hat{y}_{ijl} - y_{ijl}|\, m_{ij}}{\sum_{i=1}^{I} m_{ij}}\right) = \sum_{j=1}^{J}\frac{\Delta_j}{J} \tag{4.5}$$

4.2 Applicazioni dei metodi nello spazio delle osservazioni

Tutti gli algoritmi di cluster analysis 2.1, 2.2, 2.3 e 2.4 sono stati testati sui dataset "Soybean desease" [68] e "Zoo" [5] .

In particolare, per le fuzzy conceptual k-means (algoritmo 2.1) sono state utilizzate sia la metrica euclidea che quella del chi quadro: le due procedure sono state etichettate rispettivamente con le abbreviazioni FCKM.E e FCKM.C. Nel caso dell'algoritmo 2.2 delle fuzzy k-mode/populations, per l'aggiornamento dei centri sono state utilizzate sia l'equazione 2.19 (FKPOP) che l'equazione (2.26) (FKM). La grade of membership analysis (algoritmo 2.3) è stato abbreviata con la sigla GOM, mentre le fuzzy k-partitions (algoritmo 2.4) sono state etichettate come FKPAR.

Le simulazioni sono state effettuate conoscendo a priori il numero di clusters

Algorithm 4.1 QCE

1. si pone $QCE_0 = 1$;
2. WHILE$\{k \leq K\}$:
   (a) si individua la frequenza più elevata all'interno di $\mathbf{F}$: $d_k = f_{kk'} = \max_{q,q'}(f_{qq'})$ con $q, q' = 1, ..., K$;
   (b) si pone $f_{kq'} = 0$, e $f_{qk'} = 0$, per $q, q' = 1, ..., K$;
   (c) si calcola $QCE_k = QCE_{k-1} - \frac{d_k}{I}$.

e scegliendo come parametro di fuzzificazione, per tutti i metodi ad eccezione del GOM, valori compresi tra 1,1 e 2.

L' accuratezza della classificazione è stata misurata attraverso il seguente indicatore [45]:

$$QCE = 1 - \frac{\sum_{k=1}^{K} d_k}{I} \tag{4.6}$$

esso rappresenta la *quota di classificazioni errate,* essendo d_k il numero di osservazioni assegnate contemporaneamente al cluster k e alla corrispettiva classe di appartenenza originale. L'indicatore (4.6) assume valori nell'intervallo $[0, 1)$: è pari a zero, se tutte le unità sono classificate correttamente, mentre tende ad uno, all'aumentare delle classificazioni errate. Esso si costruisce partendo dalla matrice di *misclassificazione* $\mathbf{F} = \{f_{kk'}\}$ i cui elementi $f_{kk'}$ rappresentano le frequenze di unità appartenenti alla classe originale k ed assegnate al cluster k'. La matrice di misclassificazione si ottiene premoltiplicando la trasposta della matrice di appartenenza originale $\mathbf{U} = \{u_{ik}\}$, con $u_{ik} \in \{0, 1\}$ per quella stimata $\hat{\mathbf{U}} = \{\hat{u}_{ik}\}$, $\mathbf{F} = \mathbf{U}'\mathbf{U}$. Poiché sono state adottate tecniche di classificazione fuzzy, gli elementi $\hat{u}_{ik}$ di $\mathbf{U}$ assumono valori nell'intervallo $[0, 1]$, per cui si rende necessario defuzzyficare tali elementi secondo il seguente criterio:

$$\tilde{u}_{ik} = \begin{cases} 1 & \text{se } \hat{u}_{ik} = \max_{k'}(\hat{u}_{ik'}) \\ 0 & \text{altrimenti} \end{cases}$$

Per stabilire se una osservazione è stata assegnata al cluster "giusto", si è adottato un criterio per così dire "maggioritario": in sostanza, all'interno di ogni cluster, sono state considerate come "giustamente assegnate" tutte le unità la cui classe di appartenenza originaria ha la frequenza maggiore all'interno del nuovo gruppo individuato. La costruzione dell'indice (4.6) è descritta nell'algoritmo 4.1.

4.2.1 Analisi del dataset "Soybean Disease"

Il primo dataset analizzato è il *soybean disease* [68]. In esso sono contenute le osservazioni effettuate su 47 piante di soia rispetto allo stato di salute, alle condizioni metereologiche e del terreno espresse in 21 variabili, aventi da un minimo di 2 a un massimo di 7 modalità. Ciascuna pianta è classificata secondo una delle seguenti quattro patologie: *Diaporte Stem Canker* (10 casi), *Charcoal Rot* (10 casi), *Rhizoctonia Root Rot* (10 casi), *Phytophthora Rot* (17 casi).

Analisi della capacità di classificazione Per ciascuna tecnica di cluster analysis, nella tabella 4.1 sono riportati i valori medi, calcolati su 100 simulazioni effettuate, dell'indice (4.6), al variare della percentuale di osservazioni incomplete e a seconda del valore del parametro m (ad eccezione del GOM che non possiede tale parametro). Nel caso specifico, sono state considerate anche le performance in presenza di dataset completamente osservati. Complessivamente, per tutte le tecniche adottate l'indice QCE medio tende a crescere all'aumentare della percentuale di dati mancanti, essendo disponibile una quantità di informazione sempre più ridotta, ed è soggetto a variazioni rispetto ad m. Tuttavia, ciò che differenzia i metodi è la dinamica nel mutamento delle prestazioni. Se si calcola un ulteriore media del QCE, rispetto alla quota di dati mancanti e la si considera al variare di m (Fig. 4.1), si nota che per le FKPAR, le FCKM.E e le FCKM.C, l'andamento è decrescente fino a $m = 1.3$ per poi cominciare ad aumentare. Le FKPOP presentano il valore più basso per $m = 1,1$, poi l'indice cresce fino a $m = 1,4$, per poi decrescere nuovamente. Il QCE delle FKM ha il minimo in $m = 1,2$, mentre poi è sempre crescente. Utilizzando i valori di individuati come riferimento per un confronto tra le procedure e introducendo nell'analisi comparativa anche il GOM (Fig. 4.2) si riscontra che le FCKM ottengono le migliori performance complessive, a prescindere dalla metrica adottata. Fino al 10% di dati mancanti, il QCE medio è addirittura pari a 0, mentre al 50% sale soltanto fino ad un 17% circa di osservazioni mal classificate. Le due varianti delle FCKM si differenziano nettamente a partire dal 40% di dati mancanti in poi, dove le FCKM.C ottengono risultati migliori rispetto alle FCKM.E. Per ciò che concerne le restanti procedure, le FKM sembrano quelle meno performanti, distaccandosi quasi sempre di circa 20 punti percentuali in più rispetto alle altre tecniche, mentre tra i metodi probabilistici, le FKPAR ottengono risultati migliori rispetto al GOM.

Tecnica	m	0%	10%	20%	30%	40%	50%	Media	SQM
GOM		0,043	0,028	0,121	0,210	0,273	0,393	0,178	0,129
	1,1	0,079	0,067	0,174	0,252	0,335	0,442	0,225	0,135
	1,2	0,007	0,016	0,063	0,123	0,178	0,256	0,107	0,089
	1,3	0,003	0,000	0,038	0,155	0,155	0,210	0,094	0,083
	1,4	0,000	0,000	0,046	0,134	0,179	0,235	0,099	0,090
FKPAR	1,5	0,000	0,000	0,049	0,149	0,397	0,450	0,174	0,184
	1,6	0,000	0,000	0,352	0,421	0,422	0,443	0,273	0,195
	1,7	0,000	0,201	0,426	0,423	0,411	0,430	0,315	0,163
	1,8	0,214	0,411	0,425	0,420	0,401	0,426	0,383	0,076
	1,9	0,322	0,423	0,413	0,412	0,394	0,420	0,397	0,035
	2,0	0,424	0,420	0,419	0,402	0,393	0,408	0,411	0,011
	1,1	0,069	0,063	0,089	0,173	0,205	0,314	0,152	0,090
	1,2	0,014	0,026	0,014	0,070	0,095	0,221	0,073	0,072
	1,3	0,000	0,003	0,000	0,007	0,042	0,173	0,038	0,062
	1,4	0,000	0,000	0,000	0,000	0,029	0,205	0,039	0,075
FCKM.E	1,5	0,000	0,000	0,000	0,000	0,021	0,447	0,078	0,165
	1,6	0,000	0,000	0,001	0,000	0,422	0,451	0,146	0,206
	1,7	0,000	0,000	0,027	0,420	0,434	0,456	0,223	0,214
	1,8	0,000	0,170	0,422	0,422	0,433	0,456	0,317	0,172
	1,9	0,026	0,299	0,422	0,425	0,437	0,452	0,344	0,150
	2,0	0,152	0,426	0,423	0,425	0,432	0,445	0,384	0,104
	1,1	0,036	0,051	0,103	0,164	0,200	0,307	0,143	0,093
	1,2	0,003	0,015	0,002	0,046	0,063	0,172	0,050	0,059
	1,3	0,000	0,000	0,003	0,026	0,023	0,140	0,032	0,049
	1,4	0,000	0,000	0,000	0,030	0,237	0,465	0,122	0,175
FCKM.C	1,5	0,000	0,004	0,064	0,377	0,418	0,461	0,221	0,200
	1,6	0,006	0,160	0,389	0,410	0,421	0,448	0,306	0,165
	1,7	0,157	0,355	0,406	0,423	0,415	0,506	0,377	0,108
	1,8	0,342	0,386	0,407	0,411	0,429	0,640	0,436	0,095
	1,9	0,389	0,398	0,407	0,416	0,429	0,640	0,446	0,088
	2,0	0,382	0,375	0,408	0,419	0,456	0,638	0,446	0,090
	1,1	0,112	0,228	0,289	0,387	0,471	0,547	0,339	0,147
	1,2	0,154	0,221	0,256	0,389	0,469	0,535	0,337	0,137
	1,3	0,189	0,232	0,281	0,408	0,498	0,544	0,359	0,134
	1,4	0,208	0,302	0,347	0,441	0,501	0,545	0,390	0,117
FKM	1,5	0,231	0,380	0,431	0,502	0,544	0,577	0,444	0,116
	1,6	0,239	0,418	0,452	0,537	0,573	0,595	0,469	0,120
	1,7	0,259	0,445	0,474	0,581	0,579	0,600	0,490	0,119
	1,8	0,290	0,465	0,511	0,586	0,600	0,612	0,511	0,111
	1,9	0,304	0,480	0,511	0,606	0,593	0,618	0,519	0,109
	2,0	0,345	0,488	0,544	0,615	0,610	0,624	0,538	0,099
	1,1	0,022	0,015	0,059	0,162	0,220	0,340	0,136	0,117
	1,2	0,000	0,000	0,045	0,161	0,457	0,496	0,193	0,208
	1,3	0,000	0,210	0,420	0,417	0,433	0,467	0,325	0,168
	1,4	0,424	0,421	0,410	0,405	0,413	0,473	0,424	0,023
FKPOP	1,5	0,426	0,421	0,386	0,397	0,408	0,462	0,417	0,024
	1,6	0,422	0,411	0,366	0,374	0,421	0,452	0,408	0,030
	1,7	0,412	0,399	0,351	0,375	0,403	0,467	0,401	0,036
	1,8	0,416	0,391	0,337	0,370	0,406	0,461	0,397	0,039
	1,9	0,406	0,394	0,344	0,357	0,396	0,471	0,395	0,041
	2,0	0,401	0,386	0,332	0,356	0,405	0,469	0,391	0,043

Tabella 4.1: Quota Classificazioni Errate (dataset soybean disease, medie su 100 prove)

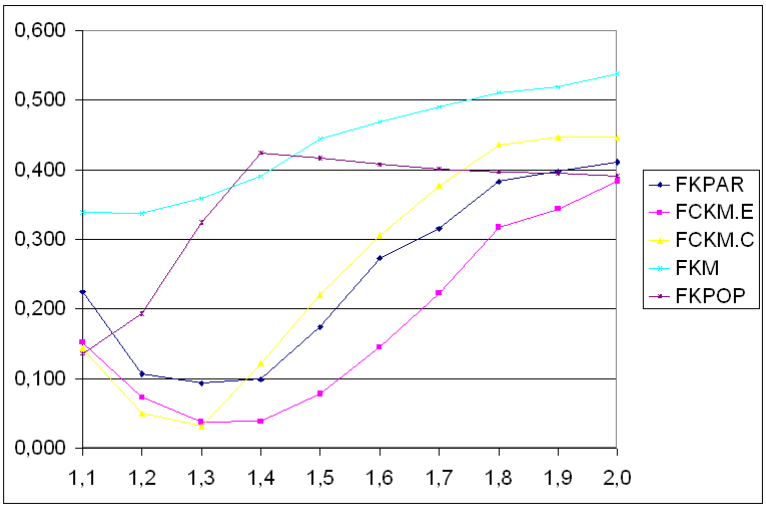

Figura 4.1: Andamento del QCE al variare di m (dataset soybean disease)

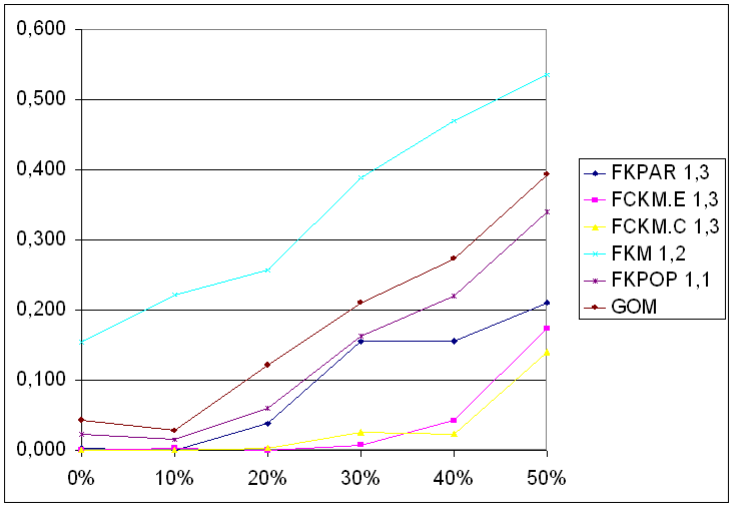

Figura 4.2: Andamento del QCE al variare della % di missing (dataset soybean disease, medie su 100 prove)

Analisi della capacità di preservazione dei valori originali La tabella 4.2 contiene i valori dell'indice (4.2). Si nota che anche la capacità di conservazione dei valori originali è funzione crescente della percentuale di dati mancanti ed è sensibile alla scelta del parametro m. In particolare, dalla figura 4.3 si evince che per FKPAR, ed FKM il valore della media di D rispetto alla percentuale dei dati mancanti ha un minimo per $m = 1,2$. Nel caso delle FKPOP le migliori performance si hanno per $m = 1,1$, mentre per le FCKM.E e le FCKM.C i minimi si hanno per $m = 1.3$. Se si selezionano in base ai valori di m individuati le migliori performance per ciascuna procedura, insieme a quelle della GOM e si mettono a confronto, (Fig. 4.4) si vede che i risultati ottenuti

con le FCKM.C sono complessivamente migliori rispetto a tutte le altre tecniche, riuscendo addirittura a manterenere la quota di connessioni errate al di sotto del 35%, perfino in presenza del 50% di dati mancanti. Per il resto, anche rispetto all'andamento dell'indice D, le FKM sembrano molto al di sotto delle altre procedure, mentre FKPAR, FKPOP e GOM si mantengono in posizione intermedia.

Tecnica	m	10%	20%	30%	40%	50%	Media	SQM
GOM		0,196	0,272	0,315	0,367	0,437	0,317	0,082
	1,1	0,209	0,293	0,338	0,394	0,448	0,336	0,082
	1,2	0,196	0,258	0,306	0,353	0,424	0,307	0,078
	1,3	0,202	0,267	0,325	0,358	0,438	0,318	0,080
	1,4	0,201	0,271	0,326	0,394	0,466	0,332	0,093
FKPAR	1,5	0,208	0,296	0,364	0,454	0,468	0,358	0,098
	1,6	0,259	0,444	0,452	0,457	0,470	0,417	0,079
	1,7	0,345	0,450	0,454	0,456	0,469	0,435	0,045
	1,8	0,428	0,445	0,453	0,456	0,467	0,450	0,013
	1,9	0,429	0,450	0,452	0,459	0,469	0,452	0,013
	2,0	0,430	0,449	0,453	0,458	0,468	0,451	0,013
	1,1	0,210	0,258	0,298	0,324	0,378	0,294	0,057
	1,2	0,205	0,240	0,260	0,284	0,343	0,267	0,046
	1,3	0,196	0,237	0,239	0,268	0,331	0,254	0,045
	1,4	0,196	0,237	0,238	0,262	0,337	0,254	0,047
FCKM.E	1,5	0,196	0,242	0,241	0,261	0,466	0,281	0,095
	1,6	0,206	0,238	0,260	0,456	0,470	0,326	0,113
	1,7	0,230	0,276	0,459	0,456	0,472	0,379	0,104
	1,8	0,370	0,442	0,458	0,456	0,472	0,440	0,036
	1,9	0,403	0,442	0,457	0,456	0,471	0,446	0,023
	2,0	0,433	0,442	0,451	0,456	0,471	0,451	0,013
	1,1	0,189	0,255	0,286	0,311	0,372	0,283	0,060
	1,2	0,195	0,232	0,251	0,268	0,322	0,253	0,042
	1,3	0,196	0,232	0,246	0,256	0,318	0,249	0,040
	1,4	0,196	0,237	0,249	0,375	0,464	0,304	0,100
FCKM.C	1,5	0,199	0,288	0,434	0,456	0,472	0,370	0,108
	1,6	0,357	0,442	0,450	0,456	0,471	0,435	0,040
	1,7	0,428	0,442	0,452	0,456	0,472	0,450	0,014
	1,8	0,433	0,442	0,452	0,456	0,471	0,451	0,013
	1,9	0,434	0,442	0,452	0,456	0,471	0,451	0,013
	2,0	0,436	0,442	0,453	0,456	0,472	0,452	0,012
	1,1	0,280	0,347	0,401	0,438	0,475	0,388	0,069
	1,2	0,281	0,343	0,397	0,430	0,457	0,382	0,063
	1,3	0,295	0,360	0,417	0,441	0,462	0,395	0,060
	1,4	0,329	0,404	0,433	0,447	0,461	0,415	0,047
FKM	1,5	0,366	0,434	0,447	0,452	0,466	0,433	0,035
	1,6	0,396	0,439	0,450	0,455	0,468	0,442	0,025
	1,7	0,414	0,441	0,449	0,455	0,469	0,446	0,018
	1,8	0,423	0,447	0,453	0,454	0,470	0,450	0,015
	1,9	0,424	0,448	0,450	0,456	0,469	0,449	0,015
	2,0	0,430	0,446	0,453	0,456	0,469	0,451	0,013
	1,1	0,194	0,258	0,320	0,369	0,436	0,315	0,084
	1,2	0,206	0,297	0,344	0,456	0,468	0,354	0,099
	1,3	0,351	0,449	0,453	0,457	0,469	0,436	0,043
	1,4	0,427	0,448	0,451	0,455	0,467	0,449	0,013
FKPOP	1,5	0,429	0,449	0,456	0,453	0,469	0,451	0,013
	1,6	0,429	0,448	0,454	0,456	0,466	0,451	0,012
	1,7	0,428	0,450	0,451	0,458	0,467	0,451	0,013
	1,8	0,428	0,447	0,451	0,457	0,469	0,450	0,014
	1,9	0,429	0,449	0,450	0,456	0,470	0,451	0,013
	2,0	0,427	0,448	0,453	0,457	0,470	0,451	0,014

Tabella 4.2: Quota Imputazioni Errate (dataset soybean disease, medie su 100 prove)

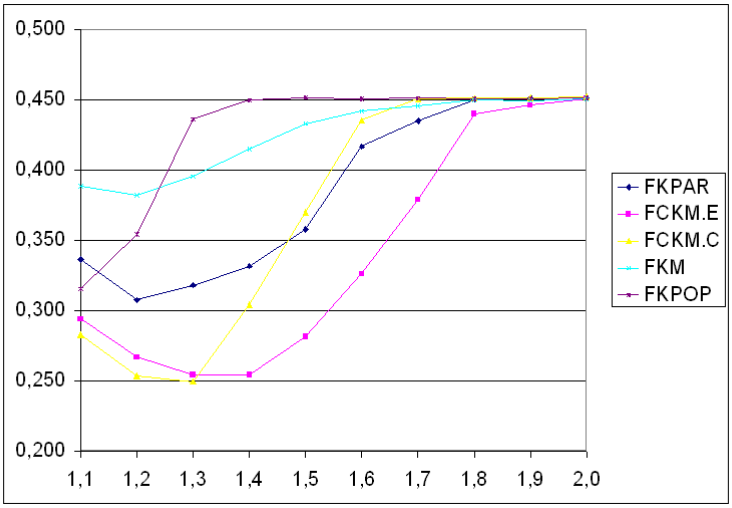

Figura 4.3: Andamento dell'indice D al variare di m (dataset soybean disease)

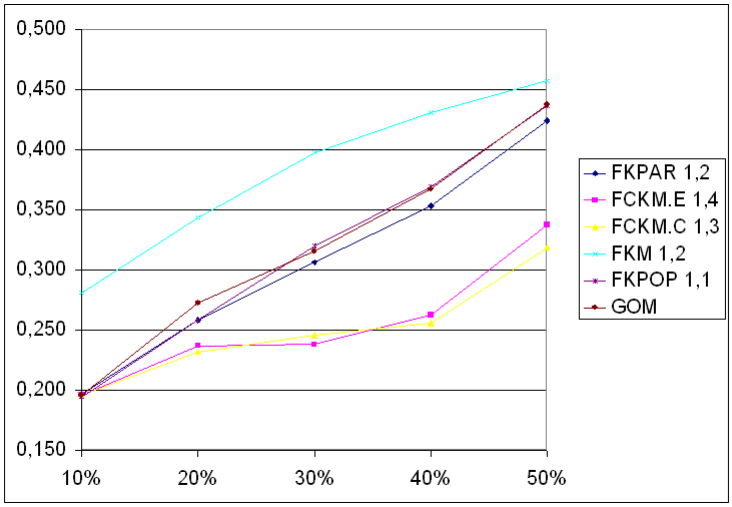

Figura 4.4: Andamento dell'indice D al variare della % di missing (dataset soybean disease, medie su 100 prove)

Analisi della capacità di preservazione della distribuzione originale L'elevato numero di configurazioni possibili del dataset soybean (quasi 400 milioni), non consente un calcolo agevole dell'indice Δ (4.4), per cui è stata adottata la versione semplificata espressa dalla (4.5). La tabella 4.3 riporta i valori dell'indice $\hat{\Delta}$ calcolato sugli output di tutte le procedure, al variare della percentuale di valori mancanti e del parametro di fuzzificazione. Osservando il grafico in Fig. 4.5, che si ottiene proiettando i valori della colonna "media" della tabella 4.3, si possono individuare i parametri ottimali di m per ciascuna delle tecniche considerate. Le FKPAR, FKM e FKPOP hanno tutte un andamento crescente, quindi le migliori prestazioni, rispetto a $\hat{\Delta}$ si hanno per $m = 1,1$. Le due tipologie di FCKM, hanno i minimi entrambe per $m = 1.3$. Anche nel

caso della preservazione della struttura dei dati originale, queste ultime due procedure mostrano di essere più efficaci delle altre, come si evince osservando il grafico in Fig. 4.6. E' interessante evidenziare l'andamento di $\hat{\Delta}$ per le FKM: esso rimane circoscritto tra $0,25$ e $0,30$, manifestando una forma di robustezza rispetto all'aumentare della quota di missing value e divenendo in tal modo la procedura più affidabile in presenza di elevate quantità di dati mancanti, subito dopo le FCKM. Tra i metodi probabilistici, la GOM risulta sistematicamente più affidabile rispetto alle FKPAR, nella capacità di ricostruire la struttura dei dati.

Tecnica	m	10%	20%	30%	40%	50%	Media	SQM
GOM		0,176	0,226	0,232	0,277	0,359	0,254	0,062
	1,1	0,191	0,243	0,258	0,301	0,345	0,267	0,052
	1,2	0,191	0,231	0,262	0,315	0,395	0,279	0,071
	1,3	0,200	0,246	0,289	0,334	0,422	0,298	0,076
	1,4	0,199	0,252	0,304	0,376	0,455	0,317	0,090
FKPAR	1,5	0,204	0,281	0,347	0,440	0,458	0,346	0,096
	1,6	0,244	0,457	0,447	0,444	0,461	0,410	0,083
	1,7	0,350	0,462	0,449	0,442	0,459	0,432	0,042
	1,8	0,452	0,458	0,447	0,443	0,457	0,451	0,006
	1,9	0,452	0,462	0,447	0,445	0,459	0,453	0,007
	2,0	0,453	0,460	0,448	0,445	0,458	0,453	0,006
	1,1	0,187	0,208	0,202	0,196	0,214	0,201	0,009
	1,2	0,189	0,202	0,190	0,179	0,203	0,193	0,009
	1,3	0,180	0,199	0,185	0,166	0,197	0,185	0,012
	1,4	0,179	0,199	0,184	0,165	0,200	0,186	0,013
FCKM.E	1,5	0,191	0,199	0,184	0,165	0,430	0,234	0,099
	1,6	0,184	0,186	0,197	0,439	0,446	0,290	0,125
	1,7	0,201	0,219	0,423	0,439	0,456	0,348	0,113
	1,8	0,333	0,455	0,423	0,439	0,456	0,421	0,046
	1,9	0,383	0,455	0,420	0,439	0,456	0,431	0,027
	2,0	0,451	0,455	0,414	0,439	0,457	0,443	0,016
	1,1	0,155	0,210	0,197	0,189	0,209	0,192	0,020
	1,2	0,179	0,201	0,187	0,173	0,185	0,185	0,009
	1,3	0,179	0,199	0,183	0,186	0,177	0,185	0,008
	1,4	0,179	0,199	0,184	0,327	0,427	0,263	0,098
FCKM.C	1,5	0,194	0,251	0,397	0,439	0,431	0,343	0,101
	1,6	0,328	0,455	0,412	0,439	0,433	0,414	0,045
	1,7	0,441	0,455	0,411	0,439	0,432	0,436	0,014
	1,8	0,451	0,455	0,417	0,439	0,431	0,439	0,014
	1,9	0,452	0,455	0,419	0,439	0,434	0,440	0,013
	2,0	0,453	0,455	0,417	0,439	0,435	0,440	0,014
	1,1	0,246	0,266	0,269	0,273	0,297	0,270	0,016
	1,2	0,261	0,277	0,308	0,327	0,360	0,306	0,035
	1,3	0,287	0,319	0,362	0,384	0,406	0,351	0,043
	1,4	0,327	0,388	0,400	0,417	0,432	0,393	0,036
FKM	1,5	0,373	0,432	0,433	0,434	0,448	0,424	0,026
	1,6	0,410	0,446	0,441	0,438	0,453	0,438	0,015
	1,7	0,430	0,449	0,442	0,440	0,457	0,444	0,009
	1,8	0,441	0,456	0,447	0,440	0,459	0,449	0,008
	1,9	0,444	0,457	0,444	0,442	0,458	0,449	0,007
	2,0	0,451	0,457	0,447	0,442	0,459	0,451	0,006
	1,1	0,195	0,229	0,274	0,330	0,403	0,286	0,074
	1,2	0,208	0,277	0,324	0,443	0,458	0,342	0,096
	1,3	0,363	0,460	0,447	0,443	0,460	0,435	0,036
	1,4	0,450	0,459	0,446	0,441	0,457	0,451	0,007
FKPOP	1,5	0,451	0,461	0,451	0,439	0,459	0,452	0,008
	1,6	0,452	0,460	0,449	0,443	0,456	0,452	0,006
	1,7	0,451	0,462	0,446	0,444	0,457	0,452	0,007
	1,8	0,451	0,459	0,447	0,444	0,459	0,452	0,006
	1,9	0,452	0,461	0,445	0,442	0,460	0,452	0,007
	2,0	0,450	0,460	0,448	0,444	0,460	0,452	0,007

Tabella 4.3: Indice di Consistenza (dataset soybean disease, medie su 100 prove)

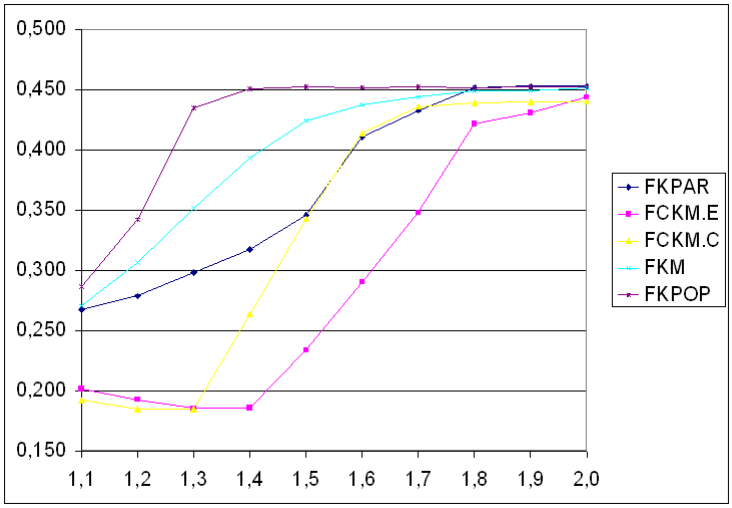

Figura 4.5: Andamento dell'indice Δ al variare di m (dataset soybean disease)

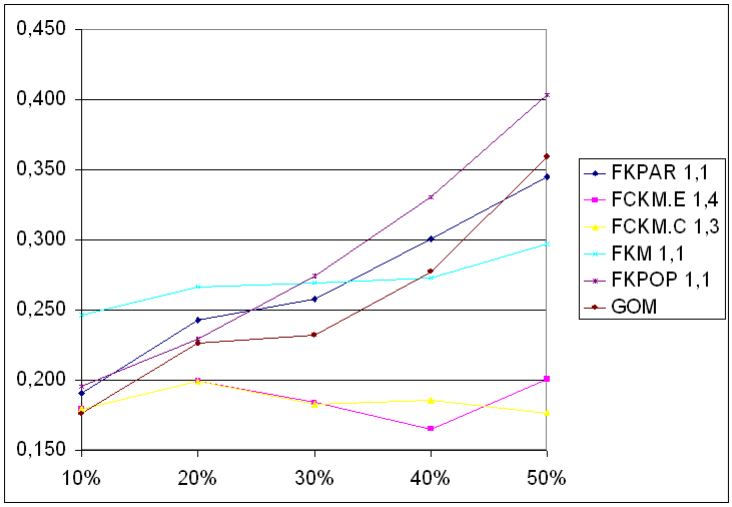

Figura 4.6: Andamento dell'indice Δ al variare della % di missing (dataset soybean disease, medie su 100 prove)

4.2.2 Analisi del dataset "Zoo"

Nel dataset *zoo* [5] sono elencate 101 osservazioni corrispondenti a diverse specie animali sulle quali sono rilevate 17 variabili legate ad aspetti anatomici ed ambientali, tutte booleane tranne una, che presenta 6 modalità. I 101 record sono classificati in 7 classi rappresentanti le specie animali, con un numero di casi pari rispettivamente a 41 per i *mammiferi*, 20 per gli *uccelli*, 5 per i *serpenti*, 13 per i *pesci*, 4 per gli *anfibi*, 8 per gli *insetti* e 10 casi per *crostacei e molluschi*.

Analisi della capacità di classificazione Nella tabella 4.4 è rappresentata la capacità delle tecniche di fuzzy cluster analysis di classificare le unità del dataset zoo, espressa attraverso l'indice QCE. Rispetto al dataset soybean disease, la quota di errore di classificazione appare complessivamente più elevata. Invece, si mantengono sostanzialmente inalterati i valori del parametro m mediante i quali le diverse procedure ottengono i migliori risultati (Fig. 4.7): $1,3$ per FKPAR, FCKM.E e FCKM.C, $1,2$ per FKM, mentre soltanto per le FKPOP il valore muta rispetto alla precedente analisi, passando da $1,1$ a $1,2$. A livello comparativo (Fig. 4.8), le procedure mantengono soltanto in parte inalterata la gerarchia già evidenziata nell'analisi del dataset soybean disease: fino al 30% dei dati mancanti, le FKPOP si rivelano la tecnica migliore, con un QCE sempre inferiore al 20% di errori di classificazione. Le distanza delle FKM dalle altre procedure non è più così marcata, mentre tra le due varianti delle FCKM, le cui differenze di prestazione sono a favore delle FCKM.C, si innestano le FKPAR.

Tecnica	m	0%	10%	20%	30%	40%	50%	Media	SQM
GOM		0,327	0,324	0,326	0,362	0,424	0,488	0,375	0,062
	1,1	0,249	0,248	0,282	0,349	0,453	0,537	0,353	0,109
	1,2	0,244	0,233	0,267	0,307	0,360	0,419	0,305	0,066
	1,3	0,218	0,219	0,270	0,306	0,369	0,426	0,301	0,076
	1,4	0,189	0,248	0,273	0,344	0,426	0,422	0,317	0,089
FKPAR	1,5	0,287	0,241	0,307	0,314	0,411	0,427	0,331	0,067
	1,6	0,267	0,282	0,315	0,408	0,422	0,429	0,354	0,068
	1,7	0,297	0,319	0,372	0,399	0,418	0,417	0,370	0,047
	1,8	0,256	0,318	0,474	0,394	0,418	0,413	0,379	0,072
	1,9	0,321	0,304	0,387	0,395	0,414	0,414	0,373	0,044
	2,0	0,308	0,365	0,390	0,388	0,416	0,399	0,378	0,035
	1,1	0,269	0,258	0,288	0,332	0,403	0,446	0,333	0,070
	1,2	0,290	0,257	0,287	0,315	0,347	0,376	0,312	0,040
	1,3	0,280	0,258	0,292	0,322	0,347	0,343	0,307	0,033
	1,4	0,295	0,308	0,295	0,327	0,388	0,336	0,325	0,032
FCKM.E	1,5	0,304	0,299	0,313	0,327	0,401	0,338	0,330	0,034
	1,6	0,302	0,295	0,307	0,342	0,374	0,431	0,342	0,048
	1,7	0,303	0,303	0,323	0,344	0,410	0,481	0,361	0,065
	1,8	0,298	0,317	0,335	0,325	0,492	0,521	0,381	0,090
	1,9	0,317	0,336	0,317	0,416	0,546	0,406	0,390	0,080
	2,0	0,327	0,308	0,328	0,437	0,496	0,407	0,384	0,068
	1,1	0,200	0,195	0,239	0,299	0,344	0,410	0,281	0,078
	1,2	0,192	0,193	0,219	0,261	0,294	0,332	0,249	0,052
	1,3	0,168	0,189	0,242	0,233	0,267	0,291	0,232	0,042
	1,4	0,163	0,211	0,301	0,221	0,278	0,269	0,240	0,047
FCKM.C	1,5	0,161	0,265	0,248	0,243	0,290	0,353	0,260	0,057
	1,6	0,154	0,276	0,234	0,253	0,361	0,408	0,281	0,083
	1,7	0,194	0,284	0,202	0,275	0,413	0,394	0,293	0,085
	1,8	0,287	0,259	0,218	0,318	0,371	0,400	0,309	0,063
	1,9	0,300	0,232	0,272	0,326	0,382	0,403	0,319	0,059
	2,0	0,272	0,252	0,295	0,345	0,391	0,403	0,326	0,058
	1,1	0,321	0,322	0,342	0,419	0,540	0,603	0,424	0,111
	1,2	0,323	0,332	0,354	0,418	0,500	0,586	0,419	0,096
	1,3	0,318	0,321	0,372	0,433	0,531	0,591	0,428	0,103
	1,4	0,350	0,340	0,398	0,464	0,563	0,613	0,454	0,104
FKM	1,5	0,342	0,332	0,438	0,502	0,575	0,634	0,471	0,112
	1,6	0,350	0,341	0,459	0,519	0,597	0,640	0,484	0,114
	1,7	0,347	0,369	0,481	0,524	0,614	0,650	0,498	0,113
	1,8	0,346	0,354	0,498	0,543	0,613	0,662	0,503	0,120
	1,9	0,357	0,393	0,502	0,553	0,620	0,655	0,513	0,109
	2,0	0,359	0,401	0,514	0,557	0,634	0,655	0,520	0,110
	1,1	0,177	0,176	0,192	0,237	0,330	0,434	0,258	0,095
	1,2	0,120	0,127	0,132	0,194	0,360	0,421	0,226	0,121
	1,3	0,110	0,138	0,175	0,308	0,440	0,486	0,276	0,147
	1,4	0,198	0,255	0,391	0,397	0,431	0,460	0,355	0,095
FKPOP	1,5	0,269	0,384	0,384	0,394	0,433	0,455	0,387	0,059
	1,6	0,410	0,394	0,382	0,396	0,431	0,453	0,411	0,024
	1,7	0,394	0,394	0,383	0,395	0,430	0,451	0,408	0,024
	1,8	0,392	0,390	0,375	0,388	0,426	0,454	0,404	0,027
	1,9	0,393	0,390	0,378	0,383	0,425	0,457	0,405	0,028
	2,0	0,393	0,388	0,368	0,386	0,421	0,450	0,401	0,027

Tabella 4.4: Quota Classificazioni Errate (dataset zoo, medie su 100 prove)

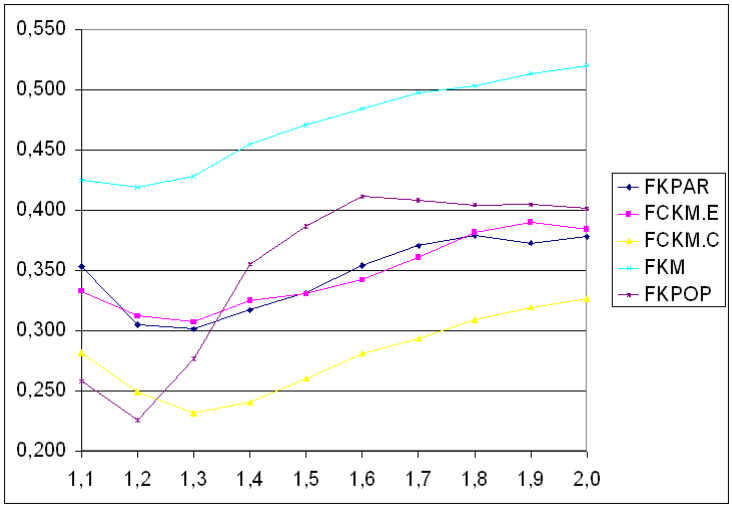

Figura 4.7: Andamento del QCE al variare di m (dataset zoo)

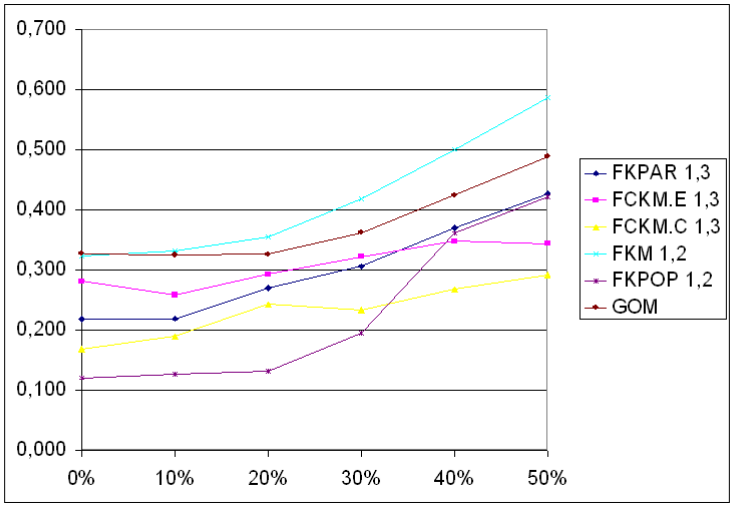

Figura 4.8: Andamento del QCE al variare della % di missing (dataset zoo, medie su 100 prove)

Analisi della capacità di preservazione dei valori originali La tabella 4.5 evidenzia, in termini di quota media di imputazioni errate su 100 simulazioni, la capacità delle procedure di ricostruire la matrice dei dati iniziale. Al contrario di quanto visto in occasione dell'analisi del dataset soybean disease, l'indice D si assesta su quote complessivamente più basse. Anche in presenza del 50% di dati mancanti, il valore di D rimane comunque al di sotto del 40% di imputazioni errate. Considerando l'andamento di D al variare del parametro di fuzzificazione (Fig. 4.9), rispetto alla medesima analisi effettuata sul dataset soybean disease, si assiste ad uno slittamento in avanti del valore di m ottimale: da $1,2$ a $1,3$ per FKPAR e FKM, da $1,4$ a $1,5$ per FCKM.E e da $1,3$ a $1,4$ per FCKM.C. Soltanto per le FKPOP, il valore rimane stabilmente

ancorato ad $1,1$. Il confronto tra le procedure (Fig. 4.10) vede prevalere le FCKM.E su tutte le altre, seppure non di molto rispetto alle FCKM.C, entrambe mantendosi sempre al di sotto del 20% di imputazioni errate, anche con il 50% di dati mancanti. Per contro le FKM mostrano dei valori abbastanza distanti dalle altre tecniche.

Tecnica	m	10%	20%	30%	40%	50%	Media	SQM
GOM		0,128	0,160	0,179	0,232	0,266	0,193	0,050
	1,1	0,143	0,173	0,201	0,255	0,301	0,214	0,057
	1,2	0,127	0,147	0,156	0,200	0,242	0,174	0,041
	1,3	0,118	0,141	0,156	0,201	0,246	0,172	0,046
	1,4	0,117	0,143	0,172	0,228	0,292	0,190	0,063
FKPAR	1,5	0,122	0,145	0,222	0,307	0,319	0,223	0,081
	1,6	0,124	0,235	0,265	0,313	0,318	0,251	0,071
	1,7	0,146	0,277	0,296	0,314	0,318	0,270	0,064
	1,8	0,196	0,290	0,297	0,314	0,319	0,283	0,045
	1,9	0,215	0,321	0,298	0,313	0,319	0,293	0,040
	2,0	0,223	0,322	0,298	0,314	0,318	0,295	0,037
	1,1	0,129	0,155	0,164	0,202	0,225	0,175	0,034
	1,2	0,112	0,137	0,145	0,170	0,183	0,149	0,025
	1,3	0,112	0,125	0,134	0,157	0,172	0,140	0,022
	1,4	0,103	0,120	0,130	0,152	0,173	0,136	0,025
FCKM.E	1,5	0,105	0,118	0,124	0,153	0,177	0,135	0,026
	1,6	0,100	0,117	0,136	0,180	0,228	0,152	0,046
	1,7	0,104	0,126	0,143	0,225	0,229	0,165	0,052
	1,8	0,116	0,153	0,167	0,229	0,285	0,190	0,060
	1,9	0,122	0,177	0,217	0,229	0,308	0,211	0,061
	2,0	0,129	0,193	0,218	0,251	0,308	0,220	0,060
	1,1	0,142	0,155	0,169	0,197	0,222	0,177	0,029
	1,2	0,129	0,142	0,147	0,164	0,182	0,153	0,019
	1,3	0,123	0,133	0,142	0,155	0,176	0,146	0,018
	1,4	0,122	0,130	0,137	0,156	0,174	0,144	0,019
FCKM.C	1,5	0,123	0,129	0,143	0,151	0,207	0,151	0,030
	1,6	0,128	0,135	0,144	0,221	0,267	0,179	0,055
	1,7	0,128	0,135	0,186	0,240	0,308	0,199	0,068
	1,8	0,121	0,174	0,224	0,294	0,308	0,224	0,071
	1,9	0,147	0,224	0,259	0,300	0,308	0,248	0,059
	2,0	0,179	0,247	0,287	0,300	0,308	0,264	0,048
	1,1	0,149	0,201	0,249	0,318	0,363	0,256	0,077
	1,2	0,141	0,192	0,223	0,281	0,322	0,232	0,064
	1,3	0,143	0,201	0,228	0,273	0,303	0,230	0,056
	1,4	0,146	0,215	0,236	0,281	0,303	0,236	0,055
FKM	1,5	0,152	0,250	0,252	0,284	0,305	0,249	0,053
	1,6	0,160	0,265	0,264	0,295	0,305	0,258	0,052
	1,7	0,176	0,281	0,268	0,299	0,310	0,267	0,048
	1,8	0,186	0,290	0,277	0,303	0,311	0,273	0,045
	1,9	0,204	0,302	0,279	0,301	0,314	0,280	0,039
	2,0	0,224	0,297	0,280	0,306	0,314	0,284	0,032
	1,1	0,121	0,147	0,162	0,207	0,263	0,180	0,050
	1,2	0,119	0,135	0,156	0,231	0,318	0,192	0,074
	1,3	0,122	0,170	0,244	0,313	0,320	0,234	0,078
	1,4	0,172	0,322	0,298	0,313	0,318	0,284	0,057
FKPOP	1,5	0,222	0,321	0,297	0,311	0,319	0,294	0,037
	1,6	0,257	0,321	0,297	0,312	0,318	0,301	0,023
	1,7	0,258	0,324	0,298	0,313	0,319	0,302	0,024
	1,8	0,257	0,323	0,298	0,314	0,318	0,302	0,024
	1,9	0,258	0,319	0,297	0,314	0,317	0,301	0,023
	2,0	0,257	0,317	0,298	0,314	0,319	0,301	0,023

Tabella 4.5: Quota Imputazioni Errate (dataset zoo, medie su 100 prove)

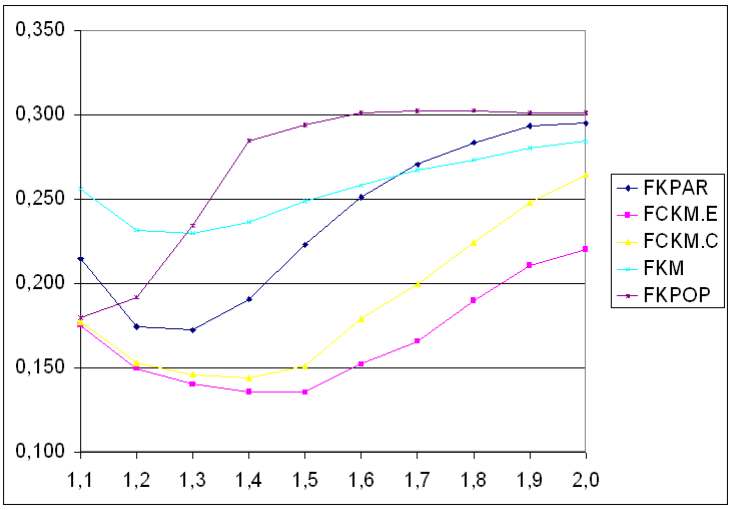

Figura 4.9: Andamento dell'indice D al variare di m (dataset zoo)

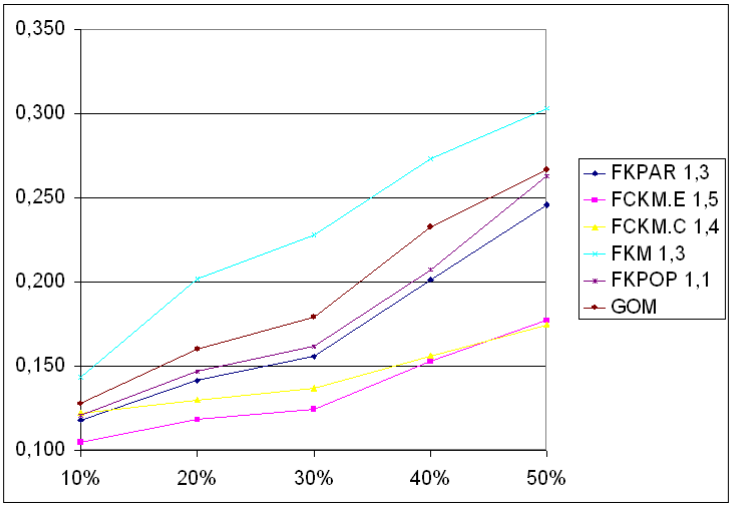

Figura 4.10: Andamento dell'indice D al variare della % di missing (dataset zoo, medie su 100 prove)

Analisi della capacità di preservazione della distribuzione originale Come nel caso del dataset soybean disease, anche per il dataset zoo, l'indice utilizzato per la valutazione della capacità di preservazione della struttura è $\hat{\Delta}$, in quanto il numero delle configurazioni possibili dato dal prodotto tra il numero di modalità di tutte le variabili è pari circa a 400000. Di conseguenza, si è cercato di misurare almeno la validità delle distribuzioni marginali delle variabili considerate. A livello di intensità del fenomeno misurato (Tab. 4.6), le prestazioni sono complessivamente peggiori rispetto ai risultati ottenuti sul dataset soybean disease e praticamente per tutte le procedure i valori di $\hat{\Delta}$ degradano rapidamente all'aumentare della quota di dati mancanti. I valori di m che ottimizzano le tecniche di classificazione sono leggermente diversi da

quelli individuati analizzando il precedente dataset (Fig. 4.11): per le FKPAR si passa da $1,1$ a $1,3$, le due varianti delle FCKM passano da $m = 1,3$ a $m = 1,4$, mentre il parametro m per le FKM cambia da $1,1$ a $1,2$. Le FKPOP conservano il valore di $1,1$. Come già evidenziato, l'andamento dell'indice $\hat{\Delta}$ per le diverse metodologie rispetto alla quota dei dati mancanti è praticamente lineare con una pendenza piuttosto marcata. Se nel caso delle FKPOP, questa situazione si presentava anche in seguito all'applicazione della procedura sul dataset soybean disease, in altri casi, come per le FKM, le FCKM.E e le FCKM.C si perde completamente la qualità della robustezza che si era manifestata nella precedente analisi.

Tecnica	m	10%	20%	30%	40%	50%	Media	SQM
GOM		0,179	0,347	0,490	0,660	0,806	0,496	0,222
	1,1	0,206	0,355	0,503	0,683	0,844	0,518	0,227
	1,2	0,187	0,329	0,461	0,609	0,778	0,473	0,207
	1,3	0,176	0,327	0,451	0,595	0,774	0,465	0,207
	1,4	0,173	0,333	0,474	0,647	0,870	0,499	0,242
FKPAR	1,5	0,177	0,320	0,565	0,847	0,936	0,569	0,293
	1,6	0,180	0,488	0,653	0,882	0,936	0,628	0,276
	1,7	0,193	0,556	0,770	0,874	0,938	0,666	0,270
	1,8	0,239	0,586	0,769	0,880	0,936	0,682	0,252
	1,9	0,272	0,672	0,771	0,883	0,934	0,707	0,235
	2,0	0,285	0,673	0,770	0,883	0,937	0,710	0,231
	1,1	0,186	0,324	0,452	0,568	0,662	0,438	0,170
	1,2	0,163	0,303	0,428	0,540	0,623	0,411	0,164
	1,3	0,167	0,290	0,415	0,524	0,617	0,402	0,161
	1,4	0,155	0,286	0,416	0,499	0,620	0,395	0,162
FCKM.E	1,5	0,157	0,282	0,408	0,501	0,633	0,396	0,166
	1,6	0,151	0,281	0,446	0,585	0,764	0,445	0,217
	1,7	0,159	0,298	0,455	0,727	0,775	0,483	0,239
	1,8	0,164	0,372	0,512	0,732	0,902	0,536	0,260
	1,9	0,167	0,401	0,617	0,726	0,941	0,570	0,266
	2,0	0,172	0,418	0,624	0,775	0,941	0,586	0,269
	1,1	0,202	0,341	0,460	0,563	0,668	0,447	0,164
	1,2	0,187	0,319	0,445	0,541	0,621	0,423	0,155
	1,3	0,182	0,300	0,421	0,530	0,607	0,408	0,153
	1,4	0,179	0,292	0,405	0,525	0,599	0,400	0,152
FCKM.C	1,5	0,181	0,310	0,412	0,519	0,689	0,422	0,174
	1,6	0,189	0,319	0,434	0,699	0,854	0,499	0,244
	1,7	0,189	0,322	0,540	0,743	0,941	0,547	0,273
	1,8	0,171	0,377	0,626	0,881	0,941	0,599	0,293
	1,9	0,191	0,467	0,706	0,891	0,941	0,639	0,279
	2,0	0,222	0,517	0,767	0,891	0,941	0,667	0,267
	1,1	0,207	0,401	0,575	0,767	0,888	0,567	0,245
	1,2	0,198	0,390	0,556	0,726	0,871	0,548	0,238
	1,3	0,199	0,415	0,589	0,748	0,870	0,564	0,238
	1,4	0,201	0,444	0,609	0,786	0,890	0,586	0,245
FKM	1,5	0,203	0,511	0,655	0,797	0,900	0,613	0,243
	1,6	0,209	0,548	0,675	0,825	0,910	0,633	0,246
	1,7	0,229	0,581	0,689	0,841	0,922	0,652	0,242
	1,8	0,240	0,596	0,712	0,845	0,918	0,662	0,238
	1,9	0,264	0,621	0,723	0,849	0,928	0,677	0,231
	2,0	0,291	0,612	0,722	0,861	0,930	0,683	0,225
	1,1	0,178	0,320	0,464	0,615	0,807	0,477	0,220
	1,2	0,181	0,303	0,451	0,638	0,937	0,502	0,266
	1,3	0,179	0,360	0,616	0,881	0,935	0,594	0,292
	1,4	0,224	0,672	0,770	0,882	0,937	0,697	0,254
FKPOP	1,5	0,284	0,672	0,771	0,882	0,936	0,709	0,231
	1,6	0,345	0,672	0,770	0,880	0,937	0,721	0,209
	1,7	0,345	0,674	0,770	0,882	0,936	0,722	0,209
	1,8	0,345	0,674	0,769	0,884	0,936	0,722	0,209
	1,9	0,345	0,670	0,770	0,880	0,938	0,721	0,209
	2,0	0,345	0,669	0,769	0,881	0,935	0,720	0,209

Tabella 4.6: Indice di Consistenza (dataset zoo, medie su 100 prove)

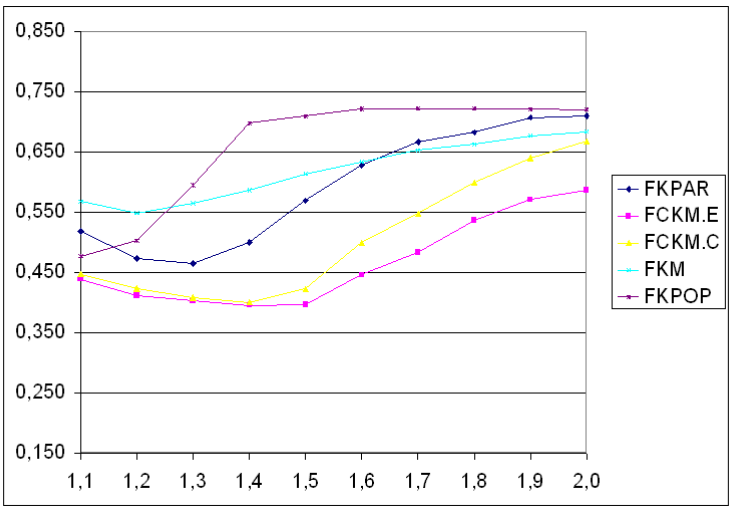

Figura 4.11: Andamento dell'indice Δ al variare di m (dataset zoo)

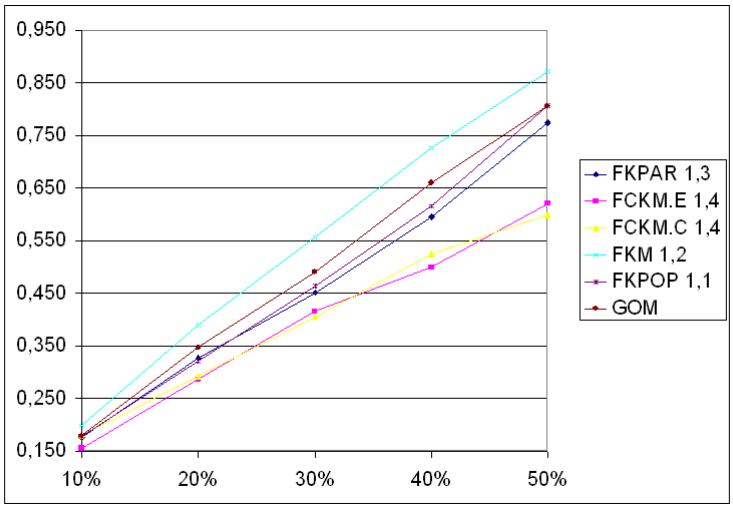

Figura 4.12: Andamento dell'indice Δ al variare della % di missing (dataset zoo, medie su 100 prove)

4.3 Applicazioni dei metodi nello spazio delle variabili

I test sulle tecniche di imputazione mediante reti bayesiane sono stati effettuati su due dataset generati con il software Hugin [64], da network preesistenti: "Asia" [55] e "Golf", quest'ultimo contenuto nel package di Hugin. I campioni ottenuti, entrambi di 1000 osservazioni, sono stati resi parzialmente incompleti con criterio MCAR, tenendo conto dei vincoli 1,2,3. Gli obiettivi principali dei test sono stati la valutazione delle proprietà 1 e 2, per quello che riguarda gli aspetti legati all'imputazione, e la misurazione della distanza tra la struttura

associativa delle variabili originale e quella individuata da parte delle tecniche di structural learning adottate. La distanza tra il vero DAG e quello stimato, è stata calcolata mediante i seguenti indicatori:

$$QAI = \frac{2}{J(J-1)} \sum_{j=1}^{J-1} \sum_{j' \neq j} (d_{jj'} d_{j'j})^2 \tag{4.7}$$

$$QAA = \frac{2}{J(J-1)} \sum_{j=1}^{J-1} \sum_{j' \neq j} (d_{jj'} + d_{j'j}) \min (d_{jj'}, d_{j'j}) \tag{4.8}$$

$$QAC = \frac{2}{J(J-1)} \sum_{j=1}^{J-1} \sum_{j' \neq j} (d_{jj'} + d_{j'j}) \max (d_{jj'}, d_{j'j}) \tag{4.9}$$

$$QAE = \frac{2}{J(J-1)} \sum_{j=1}^{J-1} \sum_{j' \neq j} \left[(d_{jj'} d_{j'j})^2 + (d_{jj'} + d_{j'j})^2 \right] \tag{4.10}$$

essendo $d_{jj'} = c_{jj'} - \hat{c}_{jj'}$, con $c_{jj'}$ elementi della matrice di adiacenza definita nella (3.1) del DAG originale, mentre $\hat{c}_{jj'}$ sono gli elementi della matrice di adiacenza relativa al DAG stimato. L'indice (4.7) è la quota di adiacenze invertite, la (4.8) rappresenta la quota di adiacenze cancellate, (4.8) è la quota di adiacenze aggiunte ed infine l'indice (4.10), somma degli altri tre, esprime la quota di adiacenze errate. Tutti e quattro gli indicatori possono assumere valori nell'intervallo $[0, 1]$. Nella figura (4.13) sono illustrate con un esempio le differenze tra la matrice delle adiacenze originale $\mathbf{C}$ e quella stimata $\hat{\mathbf{C}}$.

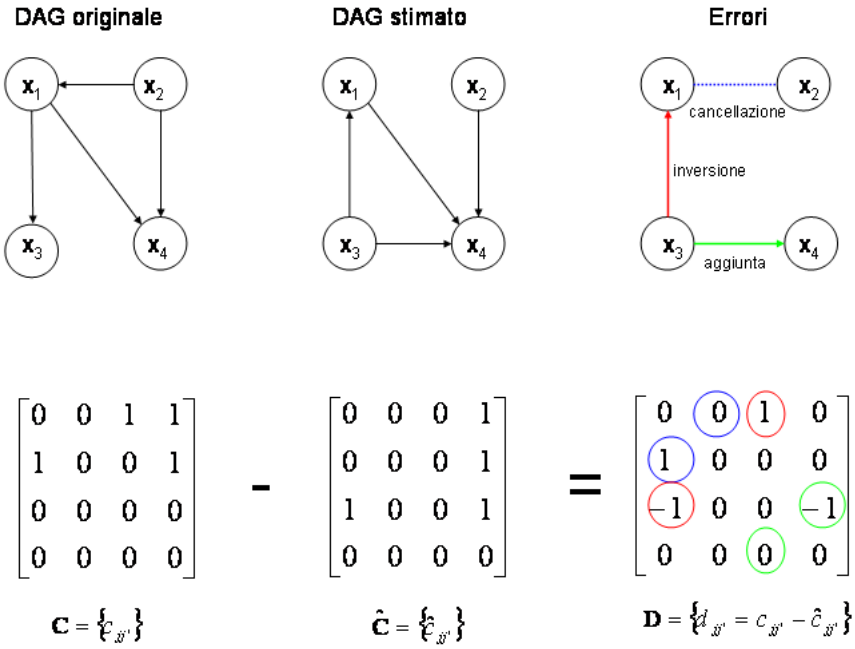

Figura 4.13: Confronto tra DAG

Un altro criterio per valutare la rete bayesiana stimata rispetto all'originale è quello di misurare la distanza tra di esse in termini di distribuzione di probabilità congiunta. Per fare ciò, si può usare la divergenza di Kullback-Leibler [52]:

$$KL(\hat{G}) = \sum_{j=1}^{J} P_G(x_j|\Pi_j^G) \log \left(\frac{P_G(x_j|\Pi_j^G)}{P_{\hat{G}}(x_j|\Pi_j^{\hat{G}})} \right) \tag{4.11}$$

$P_{\hat{G}}(x_j|\Pi_j^{\hat{G}})$ è la distribuzione di probabilità condizionale di $\mathbf{X}_j$ dato $\Pi_j^{\hat{G}}$, insieme delle configurazioni dei genitori di $\mathbf{X}_j$ che si ricavano dal DAG stimato $\hat{G}$, mentre $P_G(x_j|\Pi_j^G)$ è la distribuzione di probabilità condizionale di $\mathbf{X}_j$ dato Π_j^G, insieme delle configurazioni dei genitori di $\mathbf{X}_j$ che si ricavano dal DAG originale G. Le $P_{\hat{G}}(x_j|\Pi_j^{\hat{G}})$ possono essere calcolate facendo ricorso allo stimatore di Laplace [36] in modo da evitare che di ottenere valori infiniti per la (4.11) causate da probabilità nulle. Il calcolo della (4.11) nel caso di distribuzioni congiunte con molte variabili è a livello computazionale molto oneroso. Tuttavia, avvalendosi della fattorizzazione delle distribuzioni, la complessità può essere considerevolmente ridotta e la (4.11) può essere espressa come segue [21]:

$$
\begin{aligned}
KL(\hat{G}) &= \sum_{j=1}^{J}\sum_{l=1}^{L_j}\sum_{h=1}^{H_j^G} P_G(x_j = a_{jl}, \Pi_{jh}^G) \log P_G(x_j = a_{jl}|\Pi_{jh}^G) \\
&- \sum_{j=1}^{J}\sum_{l=1}^{L_j}\sum_{h=1}^{H_j^{\hat{G}}} P_G(x_j = a_{jl}, \Pi_{jh}^{\hat{G}}) \log P_{\hat{G}}(x_j = a_{jl}|\Pi_{jh}^{\hat{G}}) \quad (4.12)
\end{aligned}
$$

Nella (4.12) gli unici valori da calcolare sono le probabilità $P_G(x_j = a_{jl}, \Pi_{jh}^G)$ e $P_G(x_j = a_{jl}, \Pi_{jh}^{\hat{G}})$ e ciò può essere fatto agevolmente usando nel network G un algoritmo di propagazione come quello di eliminazione delle variabili [93].

4.3.1 Analisi del dataset generato dalla rete "Asia"

Il network 4.14, introdotto da Lauritzen e Spiegelhalter [55], è composto da 8 nodi che rappresentano alcune informazioni mediche espresse in forma di presenza/assenza : la dispnea potrebbe essere causata da tubercolosi, cancro ai polmoni, bronchite, da nessuna di queste malattie o da più di una di esse. Un viaggio recente in Asia può aumentare le probabilità della tuberculosi, mentre il fumo è un fattore di rischio sia per il cancro ai polmoni che per la bronchite. Il risultato di una sessione di raggi-x al torace non aiuta a discriminare tra cancro ai polmoni o tuberculosi, come pure la presenza o meno della dyspnoea. Queste informazioni sono sintetizzate dal DAG e dalle tabelle di probabilità condizionale nella figura 4.14. Uno dei nodi del grafo (Lc_or_tub) è un collegamento logico, cioè nella sua CPT non sono espresse probabilità, bensì eventi certi e serve per evidenziare che i raggi-x e la dispnea non riescono a discriminare tra cancro ai polmoni oppure bronchite. Da questa BN è stato generato un dataset di 1000 osservazioni, dal quale sono stati successivamente ricavate 5 matrici con quote crescenti di dati mancanti con criterio MCAR.

Dyspnoea

Bronchitis	State 1		State 2	
Lc_or_tub	State 1	State 2	State 1	State 2
State 1	0.9	0.8	0.7	0.1
State 2	0.1	0.2	0.3	0.9

Positive_X_ray

Lc_or_tub	State 1	State 2
State 1	0.98	0.05
State 2	0.02	0.95

Lc_or_tub

Lung_cancer	State 1		State 2	
Tuberculosis	State 1	State 2	State 1	State 2
State 1	1.0	1.0	1.0	0.0
State 2	0.0	0.0	0.0	1.0

Bronchitis

Smoker	State 1	State 2
State 1	0.6	0.3
State 2	0.4	0.7

Lung_cancer

Smoker	State 1	State 2
State 1	0.1	0.01
State 2	0.9	0.99

Tuberculosis

Visit_to_Asia	State 1	State 2
State 1	0.05	0.01
State 2	0.95	0.99

Smoker

State 1	0.5
State 2	0.5

Visit_to_Asia

State 1	0.01
State 2	0.99

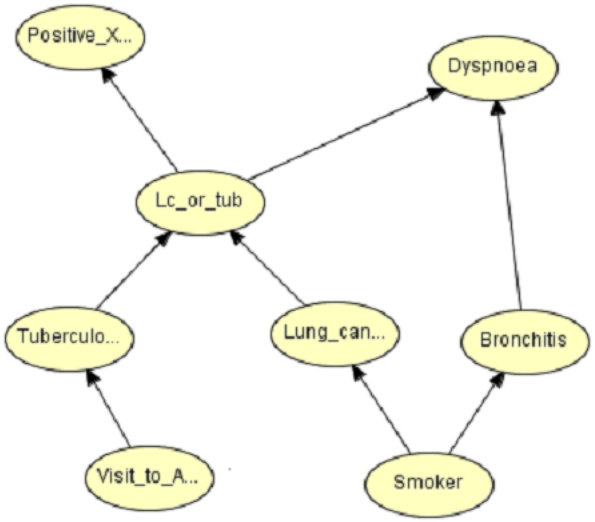

Figura 4.14: Il network Asia

Analisi della capacità di ricostruzione della struttura Gli algoritmi genetici e il greedy search sono stati testati 100 volte ciascuno sui 6 dataset originati dalla BN Asia (oltre ai 5 con i dati mancanti è stato considerato anche quello completamente osservato) e per gli indicatori (4.7), (4.8), (4.9), (4.10) e (4.12) è stato calcolato il valore medio e si è tenuto conto di minimi e massimi, oltre che dello scarto quadratico medio. Confrontando i valori nelle tabelle 4.7, 4.8, 4.9, 4.10 e 4.11, la prima cosa che si può evidenziare è che sebbene per entrambe le procedure i risultati prodotti siano buoni (ad esempio, per l'indice QAE non si va mai oltre il 30% di connessioni errate) l'algorimo greedy search mostra complessivamente migliori performance rispetto agli algoritmi genetici. Occorre evidenziare che, per necessità computazionali, il numero di cromosomi costituenti la popolazione campione dei GA è stato posto pari a 10 ed il numero di generazioni pari a 100, valori entrambi estremamaente bassi tenuto conto che lo spazio di ricerca, ovvero la numerosità dei DAG possibili nel caso di 8 nodi, è pari circa a $780 \cdot 10^9$, mentre il numero di grafi possibili è nell'ordine di $72 \cdot 10^{15}$. Gli andamenti della divergenza di Kullback-Leibler e dell'indice QAA sono gli unici nei quali si evince una alternarsi di prestazioni tra le due metodologie, anzi si potrebbe supporre esista una correlazione positiva tra l'indice QAA

e la divergenza KL. A livello intuivo, ci si aspetterebbe che le prestazioni dei diversi indicatori subiscano un peggioramente all'aumentare delle quote di dati mancanti, ma ciò sembra vero soltanto tendenzialmente, mentre addirittura, nel caso dell'indice QAI si assiste quasi ad una diminuzione dell'errore.

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,000	0,026	0,107	0,029
	10%	0,000	0,037	0,107	0,032
Greedy	20%	0,000	0,036	0,143	0,037
Search	30%	0,000	0,031	0,107	0,034
	40%	0,000	0,026	0,107	0,034
	50%	0,000	0,028	0,143	0,035
	0%	0,000	0,076	0,179	0,042
	10%	0,000	0,081	0,179	0,043
Algoritmi	20%	0,000	0,074	0,179	0,042
Genetici	30%	0,000	0,075	0,179	0,041
	40%	0,000	0,064	0,179	0,038
	50%	0,000	0,070	0,179	0,040

Tabella 4.7: Quota Adiacenze Invertite (dataset Asia, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,000	0,064	0,143	0,049
	10%	0,000	0,084	0,179	0,048
Greedy	20%	0,000	0,066	0,179	0,049
Search	30%	0,000	0,073	0,179	0,052
	40%	0,000	0,091	0,214	0,052
	50%	0,000	0,079	0,179	0,035
	0%	0,000	0,059	0,143	0,032
	10%	0,000	0,074	0,179	0,038
Algoritmi	20%	0,000	0,074	0,214	0,046
Genetici	30%	0,000	0,080	0,179	0,039
	40%	0,000	0,072	0,179	0,041
	50%	0,000	0,087	0,214	0,047

Tabella 4.8: Quota Adiacenze Aggiunte (dataset Asia, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,036	0,084	0,179	0,040
	10%	0,036	0,094	0,179	0,029
Greedy	20%	0,036	0,083	0,179	0,038
Search	30%	0,036	0,088	0,179	0,037
	40%	0,036	0,100	0,179	0,036
	50%	0,036	0,084	0,143	0,030
	0%	0,071	0,123	0,179	0,030
	10%	0,071	0,135	0,214	0,030
Algoritmi	20%	0,071	0,133	0,214	0,036
Genetici	30%	0,071	0,131	0,214	0,031
	40%	0,036	0,137	0,214	0,034
	50%	0,071	0,136	0,214	0,032

Tabella 4.9: Quota Adiacenze Cancellate (dataset Asia, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,036	0,173	0,357	0,103
	10%	0,036	0,215	0,429	0,092
Greedy	20%	0,036	0,185	0,464	0,108
Search	30%	0,036	0,192	0,464	0,107
	40%	0,036	0,218	0,500	0,108
	50%	0,071	0,191	0,429	0,076
	0%	0,071	0,258	0,429	0,064
	10%	0,143	0,289	0,429	0,061
Algoritmi	20%	0,107	0,282	0,500	0,077
Genetici	30%	0,143	0,286	0,429	0,065
	40%	0,107	0,273	0,429	0,072
	50%	0,107	0,294	0,500	0,073

Tabella 4.10: Quota Adiacenze Errate (dataset Asia, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,002	0,152	0,465	0,153
	10%	0,001	0,204	0,470	0,157
Greedy	20%	0,001	0,170	0,452	0,159
Search	30%	0,001	0,182	0,512	0,175
	40%	0,001	0,247	0,561	0,184
	50%	0,001	0,150	0,625	0,137
	0%	0,022	0,159	0,439	0,123
	10%	0,036	0,177	0,473	0,122
Algoritmi	20%	0,017	0,196	0,473	0,146
Genetici	30%	0,028	0,208	0,545	0,142
	40%	0,021	0,204	0,562	0,153
	50%	0,027	0,189	0,629	0,144

Tabella 4.11: Divergenza di Kullback-Leibler (dataset Asia, medie su 100 prove)

Analisi della capacità di preservazione dei valori originali Le due tecniche di apprendimento della struttura, insieme con le due tecniche di imputazione sono state testate combinandole tra loro, per 100 volte sui 5 dataset incompleti. Nella tabella 4.12 sono esposti i valori minimi medi e massimi dell'indice D, mentre il grafico 4.15 mostra l'andamento medio di D all'aumentare della quota di dati mancanti. I risultati sono decisamente a favore dell'imputazione tramite Markov blanket, rispetto ai nodi genitori (nell'ordine di 4-5 punti percentuali al 10% di missing, per poi scendere al 3% in presenza del 50% di missing) a prescindere dall'uso del greedy search piuttosto che degli algoritmi genetici. A proposito dei metodi di apprendimento della struttura, le differenze non sono così nette, anche se prevale di poco il greedy search. Infine, va rilevato che le quote massime di imputazioni errate non salgono mai al di sopra del 30%, dando quindi la sensazione che le procedure siano robuste rispetto alla quantità di mancate risposte.

S.Learning	Imputazione	% Missing	Min	Media	Max	SQM
		10%	0,163	0,186	0,213	0,011
		20%	0,178	0,197	0,220	0,008
	Nodi	30%	0,181	0,198	0,213	0,006
	Genitori	40%	0,187	0,202	0,219	0,007
Greedy		50%	0,187	0,205	0,220	0,007
Search		10%	0,111	0,142	0,161	0,009
		20%	0,131	0,152	0,170	0,008
	Markov	30%	0,141	0,160	0,176	0,006
	Blanket	40%	0,155	0,166	0,180	0,005
		50%	0,160	0,175	0,195	0,006
		10%	0,154	0,189	0,224	0,012
		20%	0,175	0,198	0,221	0,009
	Nodi	30%	0,183	0,201	0,218	0,007
	Genitori	40%	0,185	0,203	0,217	0,006
Algoritmi		50%	0,186	0,206	0,219	0,006
Genetici		10%	0,121	0,147	0,168	0,010
		20%	0,141	0,158	0,179	0,008
	Markov	30%	0,145	0,164	0,180	0,007
	Blanket	40%	0,155	0,168	0,189	0,006
		50%	0,161	0,176	0,191	0,006

Tabella 4.12: Quota Imputazioni Errate (dataset Asia, medie su 100 prove)

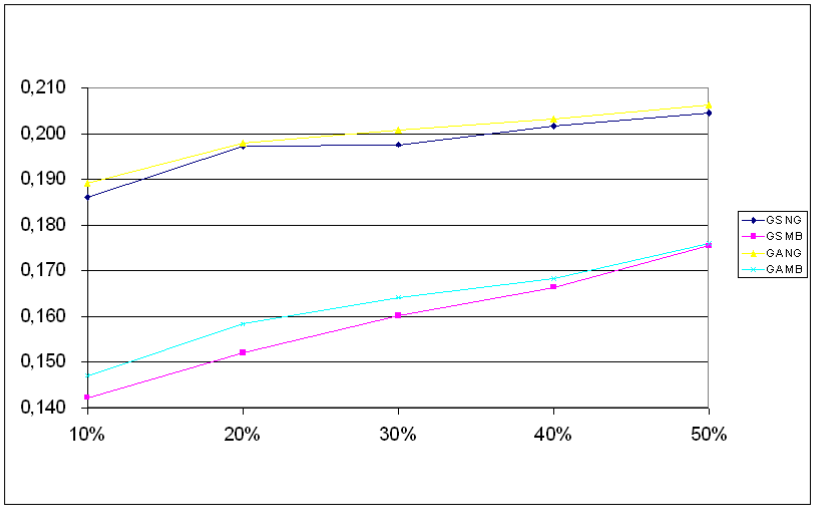

Figura 4.15: Andamento dell'indice D al variare della % di missing (dataset Asia, medie su 100 prove)

Analisi della capacità di preservazione della distribuzione originale
Considerato che il numero di configurazioni possibili pari a $2^8 = 256$, non è molto elevato rispetto alle possibilità di calcolo di un comune processore, è

stato possibile avvalersi dell'indice (4.4) per la valutazione della capacità di conservazione della struttura dei dati. Anche in questo caso (Tab. 4.13 e Fig. 4.16), come nell'analisi dell'indice D, appare maggiore l'efficacia dell'imputazione tramite Markov blanket rispetto ai nodi genitori, indipendentemente dalla tecnica di structural learning adottata. Sempre a proposito di queste ultime, si conferma con l'indice Δ la leggera prevalenza del greedy search sugli algoritmi genetici.

S.Learning	Imputazione	% missing	Min	Media	Max	SQM
Greedy Search	Nodi Genitori	10%	0,059	0,087	0,112	0,011
		20%	0,085	0,108	0,138	0,011
		30%	0,092	0,113	0,141	0,010
		40%	0,095	0,120	0,148	0,010
		50%	0,111	0,136	0,172	0,012
	Markov Blanket	10%	0,023	0,039	0,063	0,008
		20%	0,022	0,045	0,078	0,010
		30%	0,028	0,045	0,072	0,008
		40%	0,032	0,056	0,081	0,009
		50%	0,062	0,088	0,116	0,010
Algoritmi Genetici	Nodi Genitori	10%	0,058	0,093	0,120	0,010
		20%	0,079	0,109	0,145	0,013
		30%	0,088	0,120	0,159	0,012
		40%	0,101	0,131	0,156	0,012
		50%	0,109	0,144	0,183	0,014
	Markov Blanket	10%	0,023	0,043	0,070	0,009
		20%	0,024	0,052	0,082	0,011
		30%	0,027	0,049	0,074	0,011
		40%	0,038	0,056	0,074	0,008
		50%	0,062	0,086	0,115	0,012

Tabella 4.13: Indice di Consistenza (dataset Asia, medie su 100 prove)

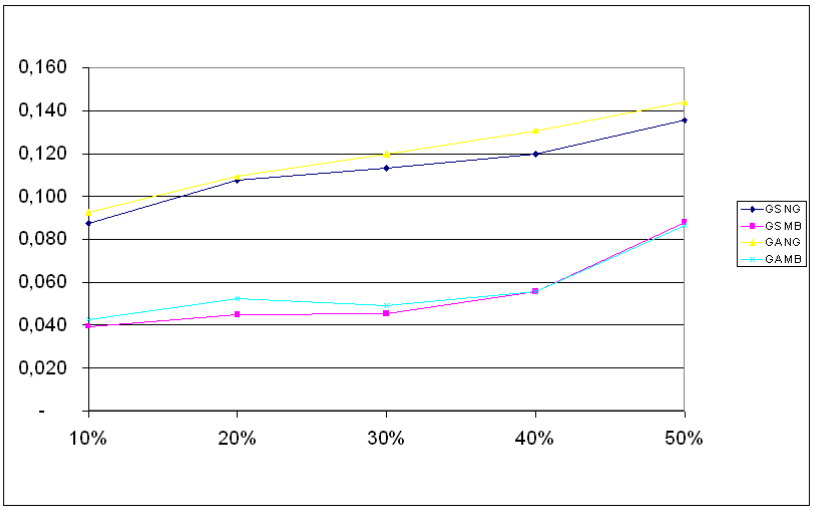

Figura 4.16: Andamento dell'indice Δ al variare della % di missing (dataset Asia, medie su 100 prove)

4.3.2 Analisi del dataset generato dalla rete "golf"

Il network golf predice la distanza percorsa da una pallina con un colpo, essendo date le informazioni sul bastone scelto, sul vento, sulla pendenza del terreno. Il risultato dipende da due fattori: la distanza, se il colpo ha successo e la probabilità di perdere la pallina. La distanza dipende dal bastone dal vento e dalla pendenza. Tuttavia, allo scopo di ridurre il numero delle configurazioni, sono stati utilizzati due collegamenti logici. Il primo (*c_ length*) è la distanza prevista quando si considera solo il tipo di bastone, l'altro (*w_ length*) dipende dal primo e dalla intensità del vento. La distanza percorsa infine dipende dal secondo collegamento logico e dalla pendenza del terreno. Le informazioni sulla rete golf sono sintetizzate nella figura 4.17.

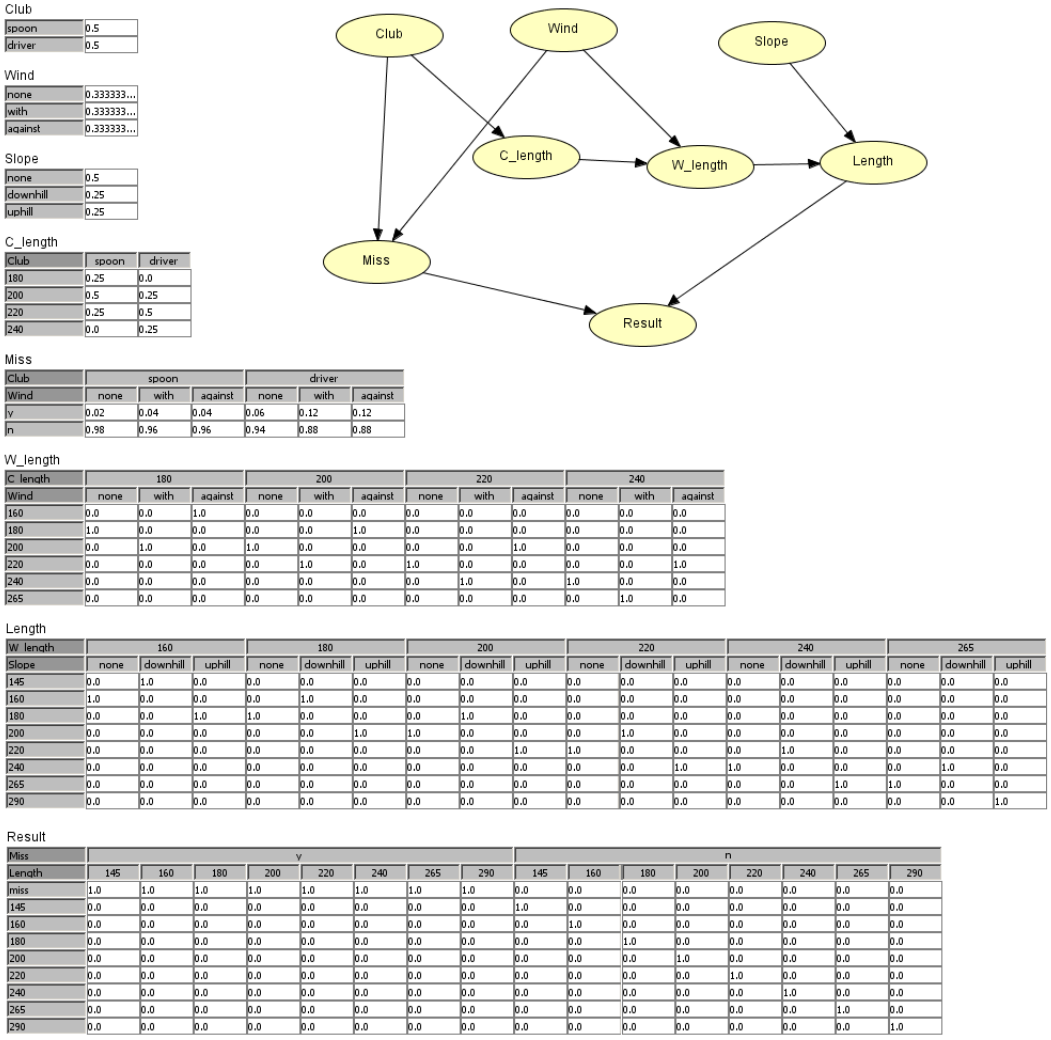

Figura 4.17: Il network golf

Analisi della capacità di ricostruzione della struttura Le tabelle 4.14, 4.15, 4.16, 4.17 e 4.18 mostrano rispettivamente il comportamento degli indi-

catori (4.7), (4.8), (4.9), (4.10) e (4.12) per ciascuna tecnica di apprendimento, al variare della quota di dati mancanti. Rispetto al network Asia, i valori degli errori sono complessivamente più elevati (la media dell'indice QAE non scende mai al di sotto del 30%) , ma anche le dinamiche presentano alcune differenze: gli algoritmi genetici sono più contenuti nei confronti del greedy search per ciò che concerne gli errori dati da connessioni aggiunte oppure invertite. Tuttavia, ciò non si rispecchia anche nell'andamento dell'errore totale, dove permane la superiorità del greedy search.

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,000	0,071	0,143	0,034
	10%	0,000	0,091	0,214	0,041
Greedy	20%	0,036	0,079	0,179	0,033
Search	30%	0,000	0,083	0,143	0,031
	40%	0,036	0,091	0,214	0,038
	50%	0,036	0,084	0,143	0,030
	0%	0,000	0,076	0,179	0,048
	10%	0,000	0,071	0,143	0,037
Algoritmi	20%	0,000	0,074	0,179	0,041
Genetici	30%	0,000	0,068	0,179	0,044
	40%	0,000	0,070	0,214	0,044
	50%	0,000	0,072	0,179	0,044

Tabella 4.14: Quota Adiacenze Invertite (dataset golf, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,036	0,119	0,250	0,043
	10%	0,036	0,112	0,214	0,035
Greedy	20%	0,071	0,115	0,214	0,039
Search	30%	0,036	0,121	0,250	0,045
	40%	0,071	0,116	0,214	0,040
	50%	0,071	0,121	0,214	0,038
	0%	0,000	0,086	0,250	0,043
	10%	0,000	0,081	0,214	0,044
Algoritmi	20%	0,000	0,093	0,179	0,041
Genetici	30%	0,000	0,092	0,214	0,043
	40%	0,000	0,087	0,214	0,044
	50%	0,000	0,096	0,214	0,045

Tabella 4.15: Quota Adiacenze Aggiunte (dataset golf, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,036	0,116	0,214	0,039
	10%	0,036	0,113	0,214	0,033
Greedy	20%	0,071	0,110	0,214	0,032
Search	30%	0,036	0,118	0,214	0,038
	40%	0,071	0,110	0,179	0,031
	50%	0,071	0,117	0,179	0,031
	0%	0,036	0,169	0,250	0,043
	10%	0,071	0,172	0,250	0,037
Algoritmi	20%	0,071	0,168	0,250	0,040
Genetici	30%	0,107	0,171	0,250	0,034
	40%	0,071	0,171	0,250	0,037
	50%	0,071	0,176	0,250	0,037

Tabella 4.16: Quota Adiacenze Cancellate (dataset golf, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,107	0,305	0,536	0,095
	10%	0,107	0,316	0,536	0,088
Greedy	20%	0,179	0,304	0,536	0,086
Search	30%	0,107	0,322	0,571	0,096
	40%	0,179	0,318	0,536	0,090
	50%	0,179	0,322	0,464	0,078
	0%	0,143	0,331	0,536	0,085
	10%	0,143	0,324	0,500	0,083
Algoritmi	20%	0,143	0,334	0,500	0,082
Genetici	30%	0,143	0,330	0,536	0,078
	40%	0,143	0,328	0,536	0,084
	50%	0,179	0,345	0,536	0,080

Tabella 4.17: Quota Adiacenze Errate (dataset golf, medie su 100 prove)

S.Learning	% Missing	Min	Media	Max	SQM
	0%	0,231	2,044	5,249	1,430
	10%	0,233	1,873	4,874	1,187
Greedy	20%	0,236	1,837	4,724	1,349
Search	30%	0,240	1,961	5,427	1,394
	40%	0,247	1,742	4,685	1,423
	50%	0,272	2,091	5,257	1,387
	0%	0,263	2,223	4,829	1,060
	10%	0,484	2,227	4,649	0,930
Algoritmi	20%	0,498	2,230	4,895	0,938
Genetici	30%	0,285	2,237	4,369	0,883
	40%	0,288	2,118	5,568	0,948
	50%	0,426	2,414	5,260	1,040

Tabella 4.18: Divergenza di Kullback-Leibler (dataset golf, medie su 100 prove)

Analisi della capacità di preservazione dei valori originali La capacità di imputazione delle reti bayesiane misurata sul dataset golf è sintetizzata nella tabella 4.19 e nel grafico in Fig. 4.18. La differenza di prestazione tra Markov blanket e nodi genitori si fa molto più evidente, rispetto all'anlisi di Asia, in quanto il divario a favore del primo criterio di imputazione, a parità di tecnica di apprendimento, è di circa il 20%. Più netta è anche l'affermazione del greedy search rispetto agli algoritmi genetici con oltre 5 punti percentuali di distacco per basse quote di dati mancanti, mentre le distanze si attenuano all'aumentare dei missing.

S.Learning	Imputazione	% Missing	Min	Media	Max	SQM
		10%	0,289	0,360	0,434	0,033
		20%	0,342	0,380	0,457	0,027
	Nodi	30%	0,375	0,420	0,508	0,028
	Genitori	40%	0,402	0,447	0,507	0,019
Greedy		50%	0,457	0,497	0,547	0,021
Search		10%	0,066	0,106	0,200	0,035
		20%	0,079	0,118	0,209	0,034
	Markov	30%	0,135	0,167	0,246	0,029
	Blanket	40%	0,195	0,214	0,242	0,008
		50%	0,289	0,316	0,352	0,013
		10%	0,339	0,418	0,519	0,039
		20%	0,370	0,436	0,495	0,027
	Nodi	30%	0,386	0,462	0,509	0,024
	Genitori	40%	0,419	0,483	0,517	0,021
Algoritmi		50%	0,462	0,510	0,546	0,021
Genetici		10%	0,084	0,188	0,340	0,051
		20%	0,110	0,197	0,318	0,043
	Markov	30%	0,143	0,220	0,347	0,038
	Blanket	40%	0,209	0,268	0,338	0,031
		50%	0,307	0,350	0,410	0,023

Tabella 4.19: Quota Imputazioni Errate (dataset golf, medie su 100 prove)

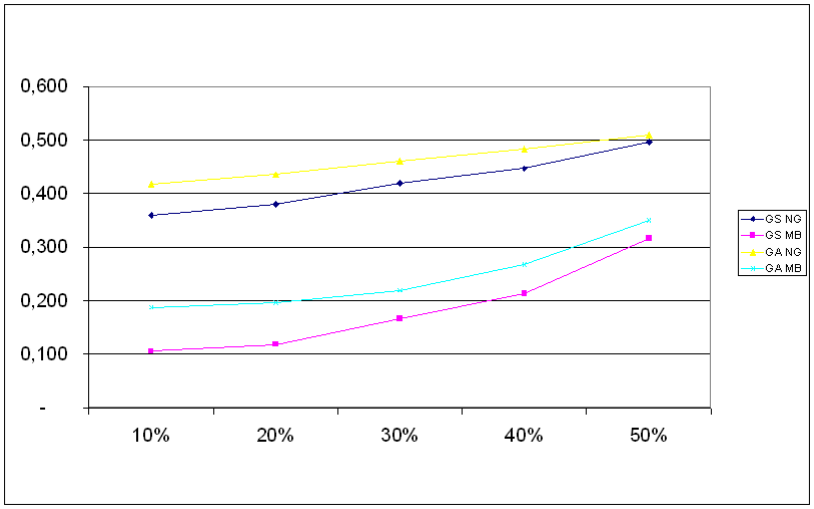

Figura 4.18: Andamento dell'indice D al variare della % di missing (dataset golf, medie su 100 prove)

Analisi della capacità di preservazione della distribuzione originale

Per misurare la distanza tra distribuzione congiunta effettiva e quella stimata è stato possibile utilizzare l'indice Δ descritto nell'equazione (4.4), nonostante il numero dei profili complessivo sia piuttosto elevato (oltre 62000). Seppure con

intensità differenti, si conferma quanto già evidenziato nell'analisi effettuata sui dati Asia (tab. 4.20 e fig. 4.19): lo scarto in favore del Markov blanket rispetto ai nodi genitori è di circa il 30%, mentre ancora una volta il greedy search prevale sugli algoritmi genetici.

S.Learning	Imputazione	% Missing	Min	Media	Max	SQM
		10%	0,266	0,350	0,446	0,039
		20%	0,373	0,430	0,552	0,040
	Nodi	30%	0,434	0,513	0,626	0,053
	Genitori	40%	0,492	0,561	0,697	0,037
Greedy		50%	0,541	0,636	0,790	0,069
Search		10%	0,036	0,067	0,182	0,031
		20%	0,039	0,074	0,188	0,039
	Markov	30%	0,066	0,101	0,207	0,040
	Blanket	40%	0,093	0,113	0,153	0,009
		50%	0,219	0,257	0,342	0,026
		10%	0,318	0,422	0,550	0,048
		20%	0,392	0,514	0,593	0,042
	Nodi	30%	0,484	0,597	0,691	0,047
	Genitori	40%	0,537	0,672	0,795	0,058
Algoritmi		50%	0,557	0,728	0,831	0,070
Genetici		10%	0,059	0,153	0,332	0,056
		20%	0,072	0,173	0,365	0,057
	Markov	30%	0,073	0,176	0,414	0,056
	Blanket	40%	0,110	0,204	0,381	0,057
		50%	0,225	0,330	0,483	0,056

Tabella 4.20: Indice di Consistenza (dataset golf, medie su 100 prove)

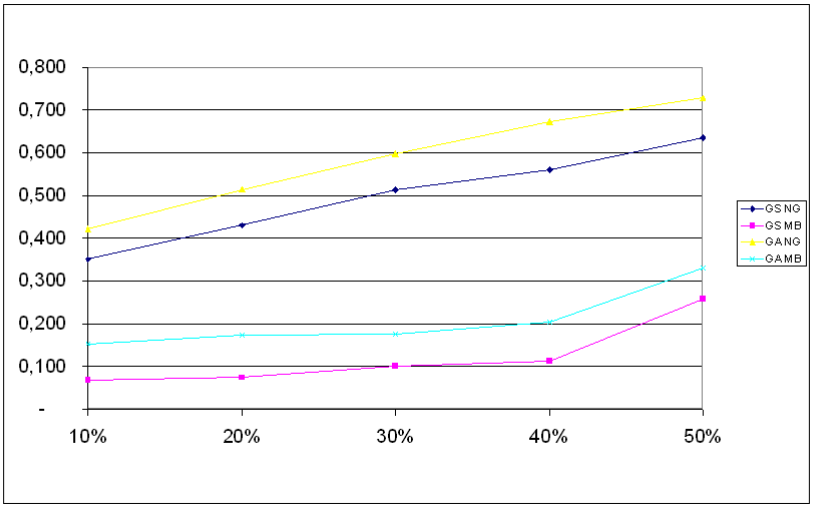

Figura 4.19: Andamento dell'indice Δ al variare della % di missing (dataset golf, medie su 100 prove)

4.4 Confronto tra cluster analysis e reti bayesiane

Le prededenti analisi sono state svolte su dataset appositamente scelti, con lo scopo di valutare le potenzialità specifiche delle diverse metodologie proposte. La struttura dei dataset soybean disease e zoo, presenta un ridotto numero di osservazioni, rispetto all'elavato numero di variabili, oltre che una variabile aggiuntiva di classificazione, mentre i dataset Asia e golf, dei quali si conosce a priori la distribuzione di probabilità che li caratterizza, presentano un ridotto numero di variabili ed un elevato numero di osservazioni, tra le quali molte con lo stesso profilo. Nel presente paragrafo, verranno invece messe a confronto le procedure sin qui descritte con dei dati reali. Inoltre, le stesse metodologie saranno ulteriormente comparate con le imputazioni effettuate tramite algoritmo EM per dati categoriali [79].

4.4.1 Analisi del dataset Contraceptive Method Choice

Il dataset Contraceptive Method Choice (CMC) [5] è un sottoinsieme dell'indagine *National Indonesia Contraceptive Prevalence Survey* del 1987. Sono rilevate su 1473 donne alcune caratteristiche socio demografiche: l'età (3 classi), il livello di educazione della donna (4 classi), il livello di educazione del marito (4 classi), il numero di figli (4 classi), la religione (2 classi), l'attuale condizione lavorativa (2 classi), lo standard living index (4 classi), il livello di esposizione ai media (2 classi) e il metodo di contraccezione utilizzato (3 classi).

Analisi della capacità di preservazione dei valori originali Nelle tabelle 4.21, 4.22, 4.23 sono esposti i risultati, in termini di media, minimo, massimo, e scarto quadratico medio dell'indice D, per le 100 prove effettuate su ciascuno dei 5 dataset con quote crescenti di valori mancanti. Il grafico nella figura 4.20 mostra l'andamento di D al variare del numero di missing. Le procedure proposte commettono errori di imputazione decisamente più contenuti rispetto alla tecnica di benchmark. Occorre però puntualizzare che l'algoritmo EM applicato su un modello multinomiale saturato, ovvero dove sono considerate tutte le possibili interazioni tra le variabili, risulta essere poco affidabile a causa dell'elevato numero di *zeri casuali* [79], ossia di profili con frequenza zero generati non da vincoli logici. A causa degli zeri non strutturali, ad alcuni elementi nello spazio dei profili potrebbe essere assegnata probabilità nulla, benchè ciò possa non essere impossibile a priori. Tuttavia, l'esperimento è svolto

appositamente in condizioni "estreme", proprio per testare la robustezza delle procedure proposte. Rispetto alle reti bayesiane, le tecniche di cluster analysis sembrano prevalere in fase di preservazione dei valori originali. Nella scelta del parametro di fuzzificazione, il criterio adottato per ciascuna procedura è stato quello di utilizzare i valori che hanno consentito le migliori prestazioni nelle analisi dei dataset soybean e zoo. Il numero dei raggruppamenti è stato individuato con un approccio empirico e cioè $K = \max_j L_j$. Nell'ambito della stessa classe di metodi, non sempre sono stati rispettate le gerarchie scaturite dalle precedenti analisi: ad esempio nelle reti bayesiane si mantiene netta la distanza tra Markov blanket e nodi genitori, sempre in favore dei primi, ma si annulla quasi del tutto la distanza tra algoritmi genetici e greedy search. Tra le tecniche di fuzzy cluster analysis, le FKPOP si candidano a valida alternativa rispetto alle due varianti di FCKM.

% Missing	Min	Media	Max	SQM
10%	0,491	0,530	0,555	0,013
20%	0,535	0,557	0,581	0,010
30%	0,551	0,575	0,596	0,008
40%	0,583	0,597	0,617	0,007
50%	0,601	0,614	0,627	0,006

Tabella 4.21: Quota Imputazioni Errate (dataset CMC, medie su 100 prove, EM)

Tecnica	% Missing	Min	Media	Max	SQM
GOM	10%	0,344	0,374	0,399	0,016
	20%	0,367	0,391	0,422	0,016
	30%	0,384	0,399	0,421	0,011
	40%	0,394	0,401	0,411	0,005
	50%	0,395	0,406	0,417	0,005
FKPAR 1.3	10%	0,355	0,364	0,381	0,011
	20%	0,386	0,386	0,386	0,000
	30%	0,389	0,389	0,390	0,001
	40%	0,393	0,396	0,399	0,003
	50%	0,394	0,397	0,406	0,003
FCKM.E 1,3	10%	0,337	0,341	0,349	0,004
	20%	0,368	0,368	0,368	0,000
	30%	0,369	0,372	0,374	0,001
	40%	0,386	0,389	0,391	0,001
	50%	0,394	0,396	0,398	0,002
FCKM.C 1,3	10%	0,351	0,353	0,372	0,006
	20%	0,368	0,384	0,386	0,006
	30%	0,377	0,386	0,389	0,004
	40%	0,399	0,399	0,399	0,000
	50%	0,398	0,398	0,398	0,000
FKM 1,2	10%	0,374	0,380	0,391	0,005
	20%	0,380	0,392	0,405	0,009
	30%	0,393	0,402	0,412	0,005
	40%	0,403	0,421	0,440	0,010
	50%	0,413	0,429	0,444	0,011
FKPOP 1,1	10%	0,350	0,360	0,368	0,005
	20%	0,369	0,375	0,382	0,004
	30%	0,381	0,388	0,390	0,003
	40%	0,388	0,395	0,399	0,003
	50%	0,394	0,398	0,407	0,004

Tabella 4.22: Quota Imputazioni Errate (dataset CMC, medie su 100 prove, fuzzy cluster analysis)

S.Learning	Imputazione	% Missing	Min	Media	Max	SQM
		10%	0,460	0,470	0,482	0,007
		20%	0,463	0,474	0,494	0,010
	Nodi	30%	0,470	0,480	0,487	0,005
	Genitori	40%	0,476	0,488	0,504	0,008
Greedy		50%	0,476	0,493	0,502	0,009
Search		10%	0,420	0,440	0,456	0,012
		20%	0,431	0,444	0,457	0,008
	Markov	30%	0,444	0,453	0,459	0,004
	Blanket	40%	0,437	0,453	0,464	0,007
		50%	0,452	0,467	0,479	0,007
		10%	0,451	0,472	0,493	0,012
		20%	0,467	0,478	0,489	0,007
	Nodi	30%	0,468	0,477	0,486	0,005
	Genitori	40%	0,474	0,486	0,495	0,007
Algoritmi		50%	0,482	0,491	0,499	0,005
Genetici		10%	0,419	0,435	0,448	0,007
		20%	0,431	0,446	0,456	0,008
	Markov	30%	0,441	0,450	0,459	0,005
	Blanket	40%	0,442	0,456	0,473	0,008
		50%	0,461	0,470	0,476	0,004

Tabella 4.23: Quota Imputazioni Errate (dataset CMC, medie su 100 prove, reti bayesiane)

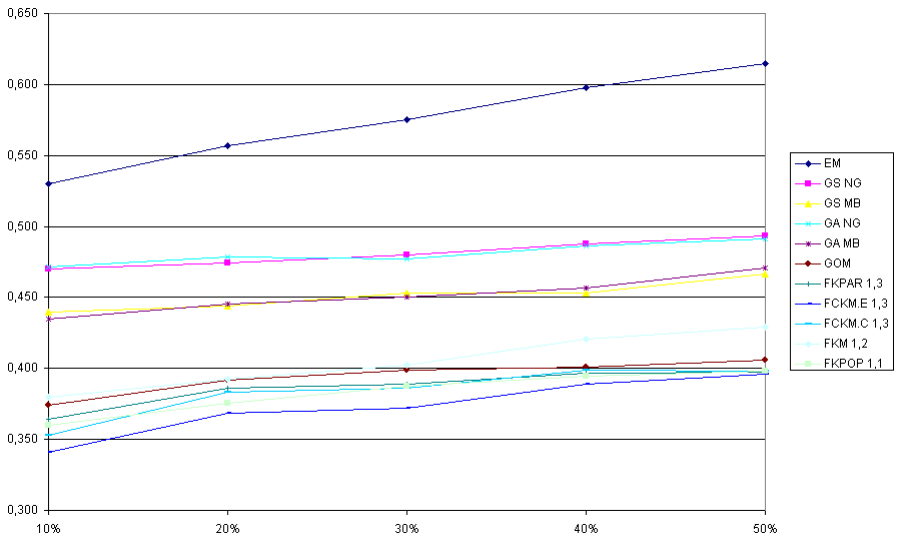

Figura 4.20: Andamento dell'indice D al variare della % di missing (dataset CMC, medie su 100 prove)

Analisi della capacità di preservazione della distribuzione originale

Il numero totale dei profili, pari 18432 consente l'uso del indice Δ per cui è possibile misurare la capcità di preservare la distribuzione congiunta delle

variabili. Anche in questa circostanza, i due gruppi di metodi proposti mostrano risultati molto più confortanti rispetto all'EM (Tabb. 4.24, 4.25, 4.26 e Fig. 4.21). In particolare, le reti bayesiane mostrano, all'aumentare della quota di dati mancanti, di essere più adatte ad individuare i legami tra le variabili rispetto alle tecniche di classificazione. Probabilmente ciò può essere, almeno in parte, spiegato dal fatto che le reti bayesiane utilizzano un criterio *random draw* nell'imputazione, più efficace nel ricostruire le associazioni tra variabili rispetto al criterio *conditional mean imputation*, impiegato nelle tecniche di fuzzy cluster analysis. Entrando nel dettaglio, i metodi basati sul Markov blanket si fanno preferire a partire dal 20% di dati mancanti, mentre quelli basati sui nodi genitori, ancora una volta meno performanti dei primi, comunque risultano essere migliori delle cluster analysis, intorno al 30% di dati mancanti. Tra queste ultime, l'unica procedura che regge il confronto con le reti sembrerebbe essere le FCKM.E, almeno fino al 30% circa di dati mancanti.

% Missing	Min	Media	Max	SQM
10%	0,454	0,493	0,523	0,011
20%	0,534	0,563	0,595	0,011
30%	0,614	0,639	0,662	0,010
40%	0,689	0,718	0,752	0,012
50%	0,769	0,791	0,819	0,010

Tabella 4.24: Indice di Consistenza (dataset CMC, medie su 100 prove, EM)

Tecnica	% Missing	Min	Media	Max	SQM
GOM	10%	0,356	0,382	0,408	0,015
	20%	0,441	0,455	0,478	0,012
	30%	0,519	0,539	0,562	0,016
	40%	0,578	0,603	0,620	0,013
	50%	0,639	0,682	0,703	0,018
FKPAR 1,3	10%	0,364	0,370	0,384	0,008
	20%	0,458	0,458	0,458	0,000
	30%	0,523	0,535	0,538	0,006
	40%	0,591	0,598	0,608	0,008
	50%	0,653	0,662	0,680	0,012
FCKM.E 1,3	10%	0,353	0,356	0,358	0,001
	20%	0,423	0,423	0,423	0,000
	30%	0,494	0,497	0,501	0,002
	40%	0,567	0,575	0,582	0,006
	50%	0,653	0,654	0,657	0,001
FCKM.C 1,3	10%	0,358	0,361	0,386	0,008
	20%	0,417	0,452	0,458	0,013
	30%	0,504	0,531	0,538	0,013
	40%	0,591	0,591	0,591	0,000
	50%	0,653	0,653	0,653	0,000
FKM 1,2	10%	0,371	0,385	0,395	0,009
	20%	0,430	0,448	0,464	0,011
	30%	0,523	0,536	0,544	0,007
	40%	0,580	0,607	0,647	0,021
	50%	0,650	0,682	0,709	0,020
FKPOP 1,1	10%	0,356	0,367	0,377	0,006
	20%	0,425	0,438	0,449	0,008
	30%	0,508	0,532	0,538	0,009
	40%	0,579	0,589	0,600	0,006
	50%	0,652	0,664	0,681	0,012

Tabella 4.25: Indice di Consistenza (dataset CMC, medie su 100 prove, fuzzy cluster analysis)

S.Learning	Imputazione	% Missing	Min	Media	Max	SQM
		10%	0,412	0,433	0,459	0,014
		20%	0,443	0,458	0,489	0,013
	Nodi	30%	0,492	0,504	0,521	0,010
	Genitori	40%	0,538	0,549	0,568	0,009
Greedy		50%	0,560	0,574	0,598	0,009
Search		10%	0,375	0,390	0,403	0,009
		20%	0,402	0,413	0,432	0,010
	Markov	30%	0,432	0,455	0,468	0,010
	Blanket	40%	0,477	0,490	0,503	0,008
		50%	0,534	0,545	0,557	0,006
		10%	0,417	0,432	0,455	0,011
		20%	0,445	0,464	0,477	0,009
	Nodi	30%	0,500	0,512	0,528	0,008
	Genitori	40%	0,528	0,550	0,567	0,013
Algoritmi		50%	0,550	0,575	0,594	0,012
Genetici		10%	0,380	0,401	0,422	0,012
		20%	0,405	0,421	0,448	0,012
	Markov	30%	0,441	0,458	0,481	0,010
	Blanket	40%	0,476	0,493	0,514	0,011
		50%	0,522	0,540	0,552	0,009

Tabella 4.26: Indice di Consistenza (dataset CMC, medie su 100 prove, reti bayesiane)

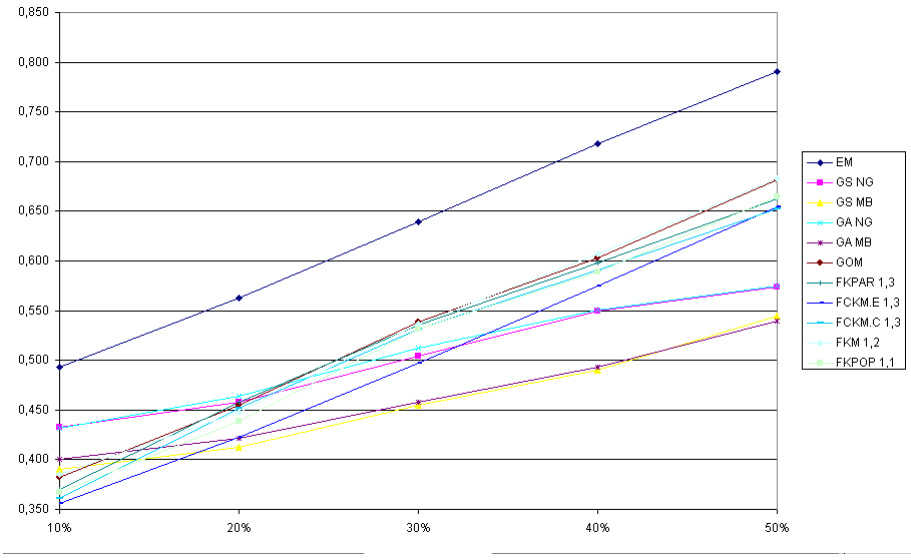

Figura 4.21: Andamento dell'indice Δ al variare della % di missing (dataset CMC, medie su 100 prove)

4.5 Conclusioni

Per chi si occupa di analisi di dati, è frequente gestire dataset di elevate dimensioni, per numero di casi e per varietà di attributi rilevati, aventi al loro interno numerose mancate risposte. Cercare di ricostruire l'informazione mancante, valorizzando sia il legame tra le osservazioni che le associazioni tra le variabili, è un modo per estrarre informazione aggiuntiva. Essa potrà così essere affiancata a quella preesistente, in modo da pervenire ad una interpretazione del fenomeno oggetto di studio quanto più possibile fedele alla realtà.

L'obiettivo del presente lavoro è stato quello di introdurre alcuni metodi per l'imputazione di dataset di variabili categoriali parzialmente osservate. Le diverse simulazioni hanno mostrato risultati incoraggianti, soprattutto alla luce del fatto che sono state effettuate "stressando" molto le procedure. Infatti, occorre rimarcare che tutti i dataset utilizzati erano privi di variabili completamente osservate e che sono state sottoposte ad analisi matrici aventi fino al 50% di dati mancanti. Le indicazioni ottenute forniscono spunti per future applicazioni e ed ulteriori approfondimenti.

Per ciò che concerne la fuzzy cluster analysis, ci si propone di individuare metodi automatizzati per la scelta del parametro di fuzzificazione m (come già proposto, nel caso di variabili continue da Luo & altri [61]). Occorre inoltre valutare la robustezza delle procedure, al variare del numero di clusters inizialmente scelti. L'inizializzazione della matrice dei centri può essere basata su criteri legati alla struttura dei dati [85], piuttosto che essere generata casualmente. Si rende necessario sviluppare tecniche di validazione della classificazione ottenuta [40] in presenza di dati mancanti. Gli algoritmi genetici potrebbero essere utilizzati come alternativa all'ottimizzazione in tre passi della funzione obiettivo (come proposto da Benati [7]). Come per le reti bayesiane, l'imputazione dei dati potrebbe essere con criterio *random draw*, assumendo i valori caratterizzanti i centroidi fuzzy, come probabilità a posteriori.

Riguardo alle reti bayesiane, si potrebbe considerare come obiettivo dello structural learning quello di stimare una classe di equivalenza di DAG, anzichè stimarne uno specifico, con un guadagno in termini di efficienza [21]. Inoltre, le performance degli algoritmi genetici possono essere migliorate aumentando la popolazione iniziale, operando su diverse e più ampie configurazioni dei parametri di input, valutando funzioni obiettivo alternative. Se si ipotizza l'esistenza di una o più variabili latenti, si potrebbe valutare la capacità di classificazione dei metodi di apprendimento introdotti, rispetto ad altri già esistenti in letteratura [13] e rispetto alla cluster analysis. In alternativa, in presenza di nodi latenti si potrebbe pensare ad una integrazione fra fuzzy cluster e reti

bayesiane, con le prime che rendono esplicita la variabile di classificazione e le seconde che studiano l'associazione tra questa e i nodi osservati.

In generale, la robustezza delle procedure proposte può essere testata su dataset caratterizzati da meccanismi generatori di dati mancanti di tipo non ignorabile, oppure NMAR. Va inoltre analizzato il comportamento dei metodi in presenza di configurazioni diverse di dati mancanti, ad esempio se tra le variabili ve ne sono alcune parzialmente osservate ed altre completamente osservate. Infine, si potrà approfondire l'analisi dei risultati delle imputazioni a livello di singola variabile, focalizzando l'attenzione anche sull'eventuale presenza di zeri strutturali.

Appendice A

Cenni sugli insiemi fuzzy

Sia X uno spazio di oggetti con $x \in X$. Un insieme *fuzzy* [92] A in X è caratterizzato da una *funzione di appartenenza* $\mu_A(x)$, la quale associa a ogni punto in X un numero reale nell'intervallo $[0, 1]$, con il valore di $\mu_A(x)$ in x che rappresenta il "grado di appartenenza" di x in A. Cosicché più è vicino all'unità il valore di $\mu_A(x)$, più alto risulta il grado di appartenenza di x in A. Quando A è un insieme nel senso ordinario del termine (*insieme classico*), la sua funzione di appartenenza può assumere soltanto due valori 0 o 1, con $\mu_A(x) = 1$ o 0 a seconda che x appartenga o non appartenga ad A. In questo caso dunque $\mu_A(x) = \mathbb{I}_A(x)$, funzione caratteristica di A (Fig. A.1).

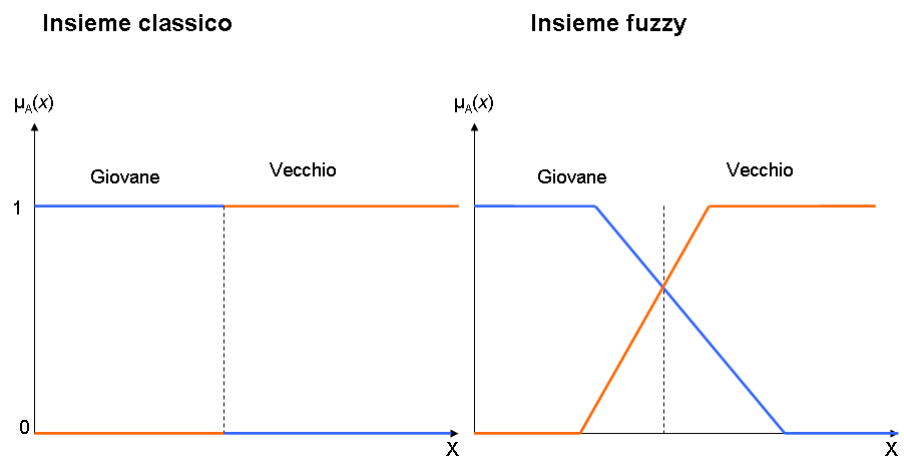

Figura A.1: Insieme classico e insieme fuzzy

La funzione di appartenenza si compone di tre elementi [1] (Fig. A.2):

- il *supporto*, cioè tutti i punti x di X in A, tali che la funzione di appartenenza sia maggiore di 0, $\{x \in X | \mu_A(x) > 0\}$;

[1] Le figure che seguono sono tratte da un lavoro di Lazzerini del 2008 [58].

- il *core*, ossia tutti i punti x di X in A, tali che la funzione di appartenenza sia uguale ad 1, $\{x \in X | \mu_A(x) = 1\}$;
- l'*altezza*, ovvero il valore assunto dalla funzione di appartenenza nel suo estremo superiore, $h(A) = \sup_{x \in X} \mu_A(x)$.

In generale, si può considerare come estensione della funzione di appartenenza un intervallo $[\alpha, \beta]$. A è normale se $h(A) = 1$.

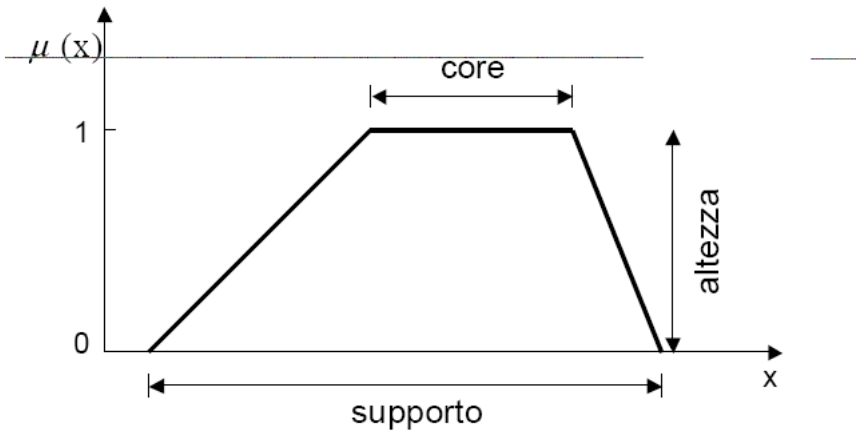

Figura A.2: Caratteristiche della funzione di appartenenza

Di seguito, vengono fornite alcune definizioni che coinvolgono gli insiemi fuzzy e che sono estensioni delle corrispondenti definizioni relative agli insiemi ordinari.

Un insieme fuzzy A è *vuoto* se e solo se la sua funzione di appartenenza è uguale a 0 su tutto X, $A = \emptyset \Leftrightarrow \{\forall x \in X | \mu_A(x) = 0\}$.

Due insiemi fuzzy A e B sono *uguali*, se e solo se le loro funzioni di appartenenza sono uguali per tutte le x in X, $A = B \Leftrightarrow \{\forall x \in X | \mu_A(x) = \mu_B(x)\}$.

L'*unione* di A e B (Fig. A.3) è l'insieme fuzzy più piccolo che contiene sia A che B, $\mu_{A \cup B}(x) = \max_{x \in X} \{\mu_A(x), \mu_B(x)\}$.

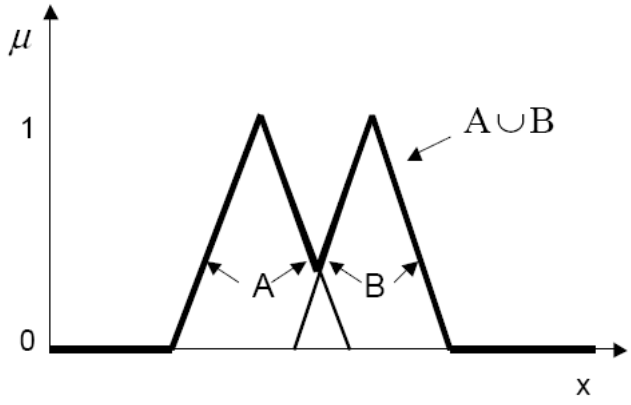

Figura A.3: Unione di insiemi fuzzy

L'*intersezione* di A e B (Fig. A.4) è l'insieme fuzzy più grande che è contenuto sia in A che in B, $\mu_{A \cap B}(x) = \min_{x \in X} \{\mu_A(x), \mu_B(x)\}$.

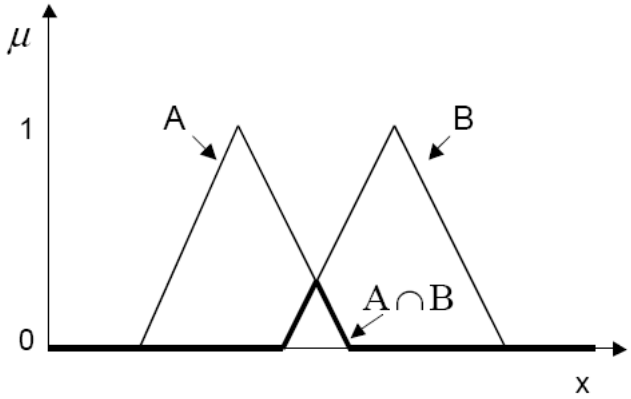

Figura A.4: Intersezione di insiemi fuzzy

Il *complemento* di un insieme fuzzy (Fig. A.5) è così definito: $\mu_{\overline{A}}(x) = 1 - \mu_A(x)$.

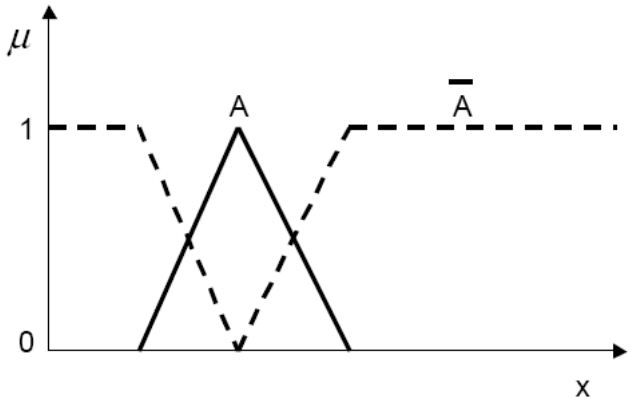

Figura A.5: Complemento di un insieme fuzzy

L'*inclusione* di insiemi fuzzy (Fig. A.6) si ha quando dati gli insiemi A e B, A è contenuto in B se e solo se la funzione di appartenenza di A è minore uguale di quella di B, per tutte le x in X, $A \subseteq B \Leftrightarrow \{\forall x \in X | \mu_A(x) \leq \mu_B(x)\}$
.

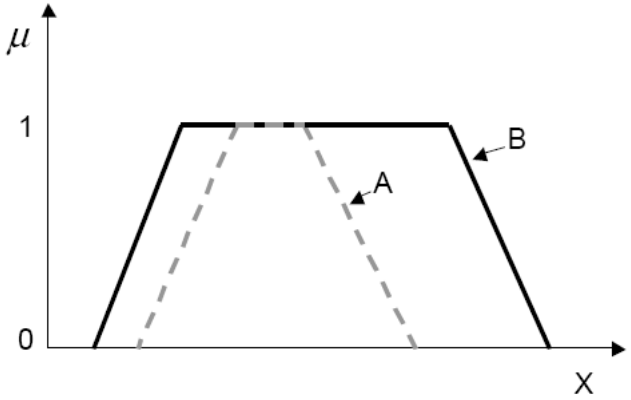

Figura A.6: Inclusione di insiemi fuzzy

La legge del terzo escluso afferma: "L'operazione di unione tra un insieme ed il suo complemento produce l'insieme universo". Nel caso degli insiemi fuzzy essa non è valida (Fig. A.7), infatti:

$$\begin{aligned}
\mu_{A \cup \overline{A}}(x) &= \max_{x \in X} \{\mu_A(x), \mu_{\overline{A}}(x)\} \\
&= \max_{x \in X} \{\mu_A(x), 1 - \mu_A(x)\} \\
&\neq \mu_X(x)
\end{aligned}$$

Cioè, esiste almeno una x in X tale che la funzione di appartenenza dell'unione di A e $\bar{A}$ è diversa da 1, $\{\exists x \in X | \mu_{A \cup \overline{A}}(x) \neq 1\}$.

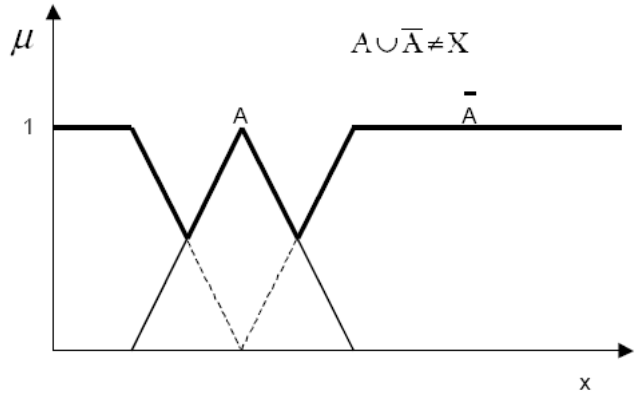

Figura A.7: Non validità della legge del terzo escluso

La legge di non contraddizione afferma: "Un generico elemento x non può appartenere all'insieme A e al suo complemento $\overline{A}$". Nel caso degli insiemi fuzzy essa non è valida (Fig. A.8), infatti:

$$\begin{aligned} \mu_{A \cap \overline{A}}(x) &= \min_{x \in X} \{\mu_A(x), \mu_{\overline{A}}(x)\} \\ &= \min_{x \in X} \{\mu_A(x), 1 - \mu_A(x)\} \\ &\neq \mu_\emptyset(x) \end{aligned}$$

Cioè, esiste almeno una x in X tale che la funzione di appartenenza dell'intersezione di A e $\bar{A}$ è diversa da 0, $\{\exists x \in X | \mu_{A \cap \overline{A}}(x) \neq 0\}$.

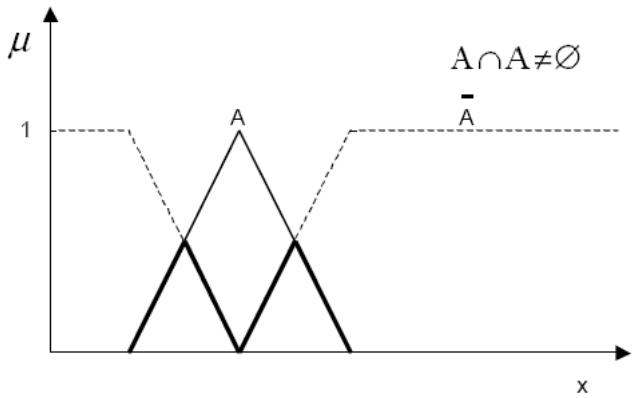

Figura A.8: Non validità della legge di non contraddizione

Appendice B

Programmi in R

B.1 L'ambiente R

R [73] è un linguaggio e un ambiente open source per l'analisi statistica. Fornisce un elevato numero di routine statistiche e grafiche ed è altamente estensibile. Di seguito sono esposte i sorgenti di tutte le procedure descritte nel presente lavoro.

B.2 Programma per l'imputazione di dataset di variabili categoriali

Il programma implementa i metodi descritti nei capitoli 2 e 3. Esso, dopo aver raccolto gli input da terminale tramite interfaccia grafica (GUI), invia le informazioni ad uno dei tre moduli disponibili, prescelto in base alla tipologia di imputazione selezionata. A sua volta il modulo, si interfaccia con le funzioni ad esso necessarie per lo svolgimento dei calcoli. Terminate le imputazioni, il modulo invocato invia l'output al programma principale che sintetizza i risultati in tabelle grafici e produce un file ASCII con i dataset imputati, oltre ad altri file contenenti tabelle riepilogative. Per l'applicazione dell'algoritmo EM in dataset di variabili categoriali si è fatto uso del package "cat" [41]. La stima della struttura delle reti bayesiane tramite algoritmi genetici è stata implementata per mezzo del package "genalg" [88]. La funzione che legge i file ".net" originati da Hugin è stata ottenuta modificando una funzione contenuta nel package "deal" [10]. Infine, la GUI necessaria al caricamento dei dati è stata implementata con funzioni ottenute dal package "tcltk" contenuto nella distribuzione base di **R**.

```
# Programma che effettua l'imputazione dei dati mancanti in un dataset di
# variabili nominali mediante le seguenti tecniche:
#       1- Fuzzy cluster analysis;
#       2- Reti bayesiane
#       3- Algoritmo EM.

rm(list=ls())

#----------------------------#
# Caricamento delle funzioni #
#----------------------------#

savehistory(file = ".Rhistory")
datahistory<-readLines(file(".Rhistory"))
close(file(".Rhistory"))
ultimariga<-datahistory[length(datahistory)]
namesplit<-strsplit(ultimariga,"\\\\")
nomeprog<-namesplit[[1]][length(namesplit[[1]])]
percorso<-sub(nomeprog,"",ultimariga,fixed=T)
nomeprog<-sub("\"\\)","",nomeprog)
percorso<-sub("source(\"","",percorso,fixed=T)
percorso<-gsub("\\\\","/",percorso,fixed=T)
percorso.funz<-paste(percorso,"\\Funzioni\\",sep="")
elenco<-list.files(percorso.funz)

for (i in 1:length(elenco)) source(paste(percorso.funz,elenco[i],sep=""))

#----------------------#
# Caricamento dei dati #
#----------------------#

carica("Caratteristiche del dataset","x","clusvecorig","nclus")
name.x<-name

quotamiss<-sum(is.na(x))/(nrow(x)*ncol(x))
n.conf.t<-prod(apply(x,2,function(u)nlevels(as.factor(u))))
mem.used<-(n.conf.t*8)+32
if(mem.used/(1024^2)<memory.size()){
        freq.prof<-data.frame(table(x))
        n.conf.0<-sum(freq.prof$Freq==0)
        q.conf.0<-n.conf.0/n.conf.t
} else q.conf.0<-NA

if(quotamiss>0){
        esisteorig<-winDialog(type = "yesno", "Esiste il dataset originale?")
        if(esisteorig=="YES"){
                carica("Caratteristiche del dataset COMPLETO","xcomp")
                valeff<-xcomp[is.na(x)]
        }
} else esisteorig<-"NO"

#--------------------------------#
# Richiesta a video di parametri #
#--------------------------------#

opzioni0<-c("Fuzzy K Cluster Analysis","Reti Bayesiane","Algoritmo EM")
label0<-c("CA","BN","EM")
opta0<-menu(opzioni0,graphics=TRUE,title="Tecnica di Imputazione")
```

```
maxprove<-as.numeric(winDialogString("N. di prove?","100"))

matrice<-data.frame(matrix(NA,1,3))
mat.d<-data.frame(matrix(rep(NA,ncol(x)+1),1,ncol(x)+1))
mat.D<-data.frame(matrix(rep(NA,ncol(x)+1),1,ncol(x)+1))
mat.e<-data.frame(matrix(rep(NA,ncol(x)+1),1,ncol(x)+1))

esistevarclas<-"NO"
esisteverodag<-"NO"

switch(opta0,source(paste(percorso,"\\Moduli\\FC Impute.R",sep="")),source(
    paste(percorso,"\\Moduli\\BN Impute.R",sep="")),source(paste(percorso,"\\
    Moduli\\EM Impute.R",sep="")))
xmi<-imputazione()


#--------------------------------#
# Analisi dei risultati ottenuti #
#--------------------------------#

sqm<-function(x=c()){
        sigma<-sqrt(mean((x-mean(x))**2))
        return(sigma)
}

if(esisteorig=="YES"){
        if(maxprove>1) windows()
        for(i in 1:maxprove){
                ris<-evalimp(x,xcomp,xmi[[1]][[i]])
                matrice[i,]<-c(ris[nrow(ris),"delta"],ris[nrow(ris),"D"],ris[
                    nrow(ris),"eps"])
                names(matrice)<-c("delta","D","epsilon")
                #print(ris)
                mat.d[i,]<-ris[,"delta"]
                names(mat.d)<-paste("delta",rownames(ris),sep=".")
                mat.D[i,]<-ris[,"D"]
                names(mat.D)<-paste("D",rownames(ris),sep=".")
                mat.e[i,]<-ris[,"eps"]
                names(mat.e)<-paste("eps",rownames(ris),sep=".")
                if(i>1) istogrammi(matrice)
        }

        riepilogo<-data.frame(matrix(NA,(ncol(x)+1)*3,4))
        rownames(riepilogo)<-c(colnames(mat.d),colnames(mat.D),colnames(mat.e)
            )
        ipermat<-cbind(mat.d,mat.D,mat.e)
        colnames(riepilogo)<-c("min","mean","max","sd")
        riepilogo[1]<-apply(ipermat,2,min)
        riepilogo[2]<-apply(ipermat,2,mean)
        riepilogo[3]<-apply(ipermat,2,max)
        riepilogo[4]<-apply(ipermat,2,sqm)
}

if(esisteverodag=="YES"){
    finris<-matrix(NA,ncol(ris.bn),4)
    rownames(finris)<-colnames(ris.bn)
    for(i in 1:ncol(ris.bn)){
          finris[i,1]<-min(ris.bn[,i])
          finris[i,2]<-mean(ris.bn[,i])
```

```
          finris[i,3]<-max(ris.bn[,i])
          finris[i,4]<-sqm(ris.bn[,i])
    }
    colnames(finris)<-c("Min","Media","Max","SQM")
}
if(esistevarclas=="YES"){
    finris<-matrix(NA,ncol(ris.cluster),4)
    rownames(finris)<-colnames(ris.cluster)
    for(i in 1:ncol(ris.cluster)){
          finris[i,1]<-min(ris.cluster[,i])
          finris[i,2]<-mean(ris.cluster[,i])
          finris[i,3]<-max(ris.cluster[,i])
          finris[i,4]<-sqm(ris.cluster[,i])
    }
    colnames(finris)<-c("Min","Media","Max","SQM")
}


#---------------------------#
# Stampa delle informazioni #
#---------------------------#

stampa<-function(){
        cat("Nome del programma:",nomeprog,"\n")
        cat("Percorso e nome del dataset:\n")
        cat(name.x,"\n")
        cat("Caratteristiche del dataset:","\n")
        print(summary(x))
        cat("Quota di dati mancanti:",quotamiss,"\n")
        cat("N. di configurazioni possibili:",n.conf.t,"\n")
        cat("Quota. di configurazioni a freq. 0:",q.conf.0,"\n")
        cat("Tecnica di imputazione:",opzioni0[opta0],"\n")
        cat("Numero di prove:",maxprove,"\n")

        stampa1()

        if(esisteorig=="YES") print(riepilogo)
        if(esisteverodag=="YES"){
                        cat("Risultati Structural learning","\n")
                        print(finris)
        }
        if(esistevarclas=="YES"){
                        cat("Risultati Cluster Analysis","\n")
                        print(finris)
        }

}
stampa()

#---------------------------#
# Salvataggio dei risultati #
#---------------------------#

scelta<-winDialog(type = "yesno", "Vuoi salvare i risultati ottenuti?")
if (scelta=="YES"){
        namesplit<-strsplit(name.x,"\\/")
        nomefile<-namesplit[[1]][length(namesplit[[1]])]
        percorso<-sub(nomefile,"",name.x,fixed=T)
        percout<-choose.dir(getwd(), "Scegli una cartella per il file di
```

```
        output")
    if(opta0==1)      nomeout<-paste(nomefile,label0[opta0],label1[opta1],
        label2[opta2],nclus,maxprove,sep=" ")
    else if(opta0==2) nomeout<-paste(nomefile,label0[opta0],label1[opta1],
        label2[opta2],label3[opta3],maxprove,sep=" ")
    else              nomeout<-paste(nomefile,label0[opta0],label1[opta1],
        label2[opta2],"-",maxprove,sep=" ")
    if(esisteorig=="YES"){
            write.table(riepilogo,file=paste(percout,"\\",nomeout," tab.
                imputazioni.csv",sep=""),dec = ",", sep=";", col.names =
                NA)
            #capture.output(xmi[[1]],file=paste(percout,"\\",nomeout,"
                imputazioni.txt",sep=""))
    }
    if (maxprove>1){
            if(esisteorig=="YES"){
                    jpeg(file=paste(percout,"\\",nomeout," grafici.jpg",
                        sep=""), width=1024, height=768, quality=100)
                    if(esisteverodag=="YES") istogrammi(cbind(ris.bn,
                        matrice))
                    if(esistevarclas=="YES") istogrammi(cbind(ris.cluster,
                        matrice))
                    if(esisteverodag=="NO" & esistevarclas=="NO")
                        istogrammi(matrice)
                    dev.off()
            }
            if(esisteorig=="NO"){
                    if(esisteverodag=="YES"){
                            jpeg(file=paste(percout,"\\",nomeout," grafici
                                .jpg",sep=""), width=1024, height=768,
                                quality=100)
                            istogrammi(ris.bn)
                            dev.off()
                    }
                    if(esistevarclas=="YES"){
                            jpeg(file=paste(percout,"\\",nomeout," grafici
                                .jpg",sep=""), width=1024, height=768,
                                quality=100)
                            istogrammi(ris.cluster)
                            dev.off()
                    }
            }
    }
    #capture.output(xmi[[2]],file=paste(percout,"\\",nomeout," risultati1.
        txt",sep=""))
    #capture.output(xmi[[3]],file=paste(percout,"\\",nomeout," risultati2.
        txt",sep=""))
    if(esisteverodag=="YES") write.table(finris,file=paste(percout,"\\",
        nomeout," tab. learning.csv",sep=""),dec = ",", sep=";", col.names
         = NA)
    if(esistevarclas=="YES") write.table(finris,file=paste(percout,"\\",
        nomeout," tab cluster.csv",sep=""),dec = ",", sep=";", col.names =
         NA)
    sink(file=paste(percout,"\\",nomeout," riepilogo.log",sep=""))
    stampa()
    sink()
}
```

B.2.1 Modulo per l'imputazione tramite fuzzy cluster analysis

```
# Programma che effettua l'imputazione dei valori mancanti
# in un dataset di variabili categoriali
# con tecniche di cluster analysis fuzzy

#--------------------------------#
# Richiesta a video di parametri #
#--------------------------------#

opzioni1<-c(
        "Grade Of Membership Analysis",
        "Fuzzy K-partitions",
        "Fuzzy Conceptual K-Means (euclidean)",
        "Fuzzy Conceptual K-Means (chi-square)",
        "Fuzzy K-modes",
        "Fuzzy K-populations")
label1<-c(
        "GoM",
        "FKpar",
        "FCKM-Euc",
        "FCKM-Chi",
        "FKM",
        "FKpop")
opta1<-menu(opzioni1,graphics=TRUE,title="Tecnica di classificazione")
if(nclus==0) nclus<-as.numeric(winDialogString("N. cluster?","2")) else
    esistevarclas<-"YES"

soglia<-as.numeric(winDialogString("Soglia di arresto?","1e-4"))
maxiter<-as.numeric(winDialogString("Max n. iterazioni per prova?","1000"))
label2<-as.character(seq(1.0,2.0,by=0.1))
opta2<-if(opta1>1) menu(label2,graphics=TRUE,title="Parametro di fuzziness?")
    else 1
m<-as.numeric(label2[opta2])

#---------------------------#
# Stampa delle informazioni #
#---------------------------#

stampa1<-function(){
        cat("Tecnica di classificazione:",opzioni1[opta1],"\n")
        cat("N. cluster:",nclus,"\n")
        cat("Parametro di fuzziness?",m,"\n")
        cat("Soglia di arresto:",soglia,"\n")
        cat("Max n. iterazioni per prova:",maxiter,"\n")
}


#-------------------------------#
# Imputazione dei dati mancanti #
#-------------------------------#

imputazione<-function(){
        if(as.numeric(R.Version()$minor)>=7) pb<-winProgressBar(opzioni0[opta0
            ],paste("0 prove di",maxprove,"completate"),min=0,max=maxprove,
            initial=0)
        if(esistevarclas=="YES"){
```

```
                ris.cluster<<-matrix(NA,maxprove,3)
                colnames(ris.cluster)<<-c("MEC.C","MEC.F","MEC.P")
        }
        ximp.list<-list()
        u.list<-list()
        c.list<-list()
        for(i in 1:maxprove){
                plotta<-if(maxprove==1) T else F
                v0<-t(rcentroid(nclus,x))
                if(opta1==1) risultati<-gom.impute(x,v0,soglia,maxiter,plotta)
                if(opta1==2) risultati<-fkpar.impute(x,v0,m,soglia,maxiter,
                    plotta)
                if(opta1==3) risultati<-fckm.impute(x,v0,m,1,soglia,maxiter,
                    plotta)
                if(opta1==4) risultati<-fckm.impute(x,v0,m,4,soglia,maxiter,
                    plotta)
                if(opta1==5) risultati<-fkpop.impute(x,v0,m,2,soglia,maxiter,
                    plotta)
                if(opta1==6) risultati<-fkpop.impute(x,v0,m,1,soglia,maxiter,
                    plotta)
                if(esistevarclas=="YES"){
                        ris.cluster[i,1]<<-clustverif(clusvecorig,risultati
                            [[3]],"Crisp",F)/nrow(x)
                        ris.cluster[i,2]<<-clustverif(clusvecorig,risultati
                            [[3]],"Fuzzy",F)/nrow(x)
                        ris.cluster[i,3]<<-clustverif(clusvecorig,risultati
                            [[3]],"Prob",F)/nrow(x)
                        if(i>1) istogrammi(ris.cluster)
                }
                ximp.list[[i]]<-risultati[[6]]
                u.list[[i]]<-risultati[[3]]
                c.list[[i]]<-risultati[[2]]
                if(as.numeric(R.Version()$minor)>=7) setWinProgressBar(pb,i,
                    opzioni0[opta0],paste(i,"prove di",maxprove,"completate"))
                     else cat(paste(i,"prove di",maxprove,"completate"),"\n")
        }
        if(as.numeric(R.Version()$minor)>=7) close(pb)
        return(list(ximp.list,u.list,c.list))
}
```

B.2.1.1 Funzioni di supporto al programma "FC Impute.R"

```
gom.impute<-function(x,v,soglia=1e-8,maxiter=1000,plot=T)

# Funzione che, in un dataset di variabili categoriali, effettua:
# - la classificazioni delle osservazioni
# - l'imputazione dei valori mancanti
# tramite "grade of membership analysis"
# Necessita delle funzioni:
# 1) disgiunt()
# 2) rimpute()
# 3) defuzzy()
# 4) blockmod()
# 5) controdisg()
# Input:
# 1) x: dataframe. E' il dataset iniziale
# 2) v: matrice. E' la matrice con i valori dei centroidi iniziale
```

```
# 3) soglia: numerico. E' lo scarto tollerabile tra i valori della funzione
#            obiettivo calcolata in due iterazioni successive
# 4) maxiter: numerico. E' il numero massimo di iterazioni
# 5) plot: logico. Se T visualizza l'andamento della funzione obiettivo
# Output:
# 1) FunzJ: numerico. E' il valore finale della funzione obiettivo
# 2) V: matrice. E' la matrice con i valori dei centroidi finale
# 3) U: matrice. E' la matrice di appartenenza
# 4) CLASSE: vettore. Assegna un cluster ad ogni osservazione
# 5) Iteraz: numerico. E' il numero di iterazioni necessarie alla convergenza
# 6) ximp: dataframe. E' il dataset con i valori mancanti imputati

{
        u<-matrix(rep(rep(1/nrow(v),nrow(v)),nrow(x)),nrow(x),nrow(v))
        y<-data.matrix(disgiunt(x))
        nmiss<-sum(is.na(y))
        if(nmiss>0){
                yorig<-y
                y<-data.matrix(rimpute(y,1))
                colnames(y)<-colnames(yorig)
        }
        iter<-0
        jfold<--1e10
        jf<--1e9
        jfvec<-c()
        if(plot) windows()
        while ((abs(jf-jfold)>soglia)&(iter<=maxiter)&(!is.nan(jf))){
                jfold<-jf
                y<-defuzzy(y)
                ### Aggiornamento delle response probabilities
                p<-u%*%v
                p[(p==0)]<-1e-20
                ### Calcolo della matrice di appartenenza U
                u<-(((y/p)%*%t(v))*u)/rowSums(y)
                clusvec<-max.col(u)
                ### Aggiornamento delle response probabilities
                p<-u%*%v
                p[(p==0)]<-1e-20
                ### Calcolo della matrice dei centroidi V
                v.num<-(t(u)%*%(y/p)*v)
                uno<-blockmod(x)
                v.den<-v.num%*%uno
                v<-v.num/v.den
                ### Aggiornamento delle response probabilities
                p<-u%*%v
                p[(p==0)]<-1e-20
                ### Imputazione dei dati mancanti
                if(nmiss>0){
                        y.imp<-p
                        y[is.na(yorig)]<-y.imp[is.na(yorig)]
                }
                ### Calcolo del valore della funzione J
                jf<-sum(y*log(p))
                jfvec<-c(jfvec,jf)
                if(plot) plot(jfvec,type="o",main="Convergenza della funzione
                    obiettivo")
                iter<-iter+1
        }
        ximp<-if(nmiss>0) data.frame(controdisg(y)) else x
```

```
        return(list(LogVS=jf,V=v,U=u,CLASSE=clusvec,Iteraz=iter,ximp=ximp))
}
```

```
fkpar.impute<-function(x,v,m,soglia=1E-10,maxiter=1000,plot=F)

# Funzione che, in un dataset di variabili categoriali, effettua:
# - la classificazioni delle osservazioni
# - l'imputazione dei valori mancanti
# tramite "fuzzy k-partitions model"
# Necessita delle funzioni:
# 1) disgiunt()
# 2) rimpute()
# 3) defuzzy()
# 4) controdisg()
# Input:
# 1) x: dataframe. E' il dataset iniziale
# 2) v: matrice. E' la matrice con i valori dei centroidi iniziale
# 3) m: numerico. E' il parametro di fuzzyficazione
# 4) soglia: numerico. E' lo scarto tollerabile tra i valori della funzione
#            obiettivo calcolata in due iterazioni successive
# 5) maxiter: numerico. E' il numero massimo di iterazioni
# 6) plot: logico. Se T visualizza l'andamento della funzione obiettivo
# Output:
# 1) FunzJ: numerico. E' il valore finale della funzione obiettivo
# 2) V: matrice. E' la matrice con i valori dei centroidi finale
# 3) U: matrice. E' la matrice di appartenenza
# 4) CLASSE: vettore. Assegna un cluster ad ogni osservazione
# 5) Iteraz: numerico. E' il numero di iterazioni necessarie alla convergenza
# 6) ximp: dataframe. E' il dataset con i valori mancanti imputati

{
        y<-data.matrix(disgiunt(x))
        nmiss<-sum(is.na(y))
        if(nmiss>0){
                yorig<-y
                y<-data.matrix(rimpute(y,1))
                colnames(y)<-colnames(yorig)
        }
        iter<-0
        jfold<--1e10
        jf<--1e9
        jfvec<-c()
        if(plot) windows()
        while ((abs(jf-jfold)>soglia)&(iter<=maxiter)&(!is.nan(jf))){
                jfold<-jf
                y<-defuzzy(y)
                ### Calcolo della matrice di appartenenza U
                u.s1<-(y%*%t(log(v)))
#               if(m>1){
                        u.s2<-apply(u.s1,1,function(w)outer(w,w,"/"))
                        u.s3<-u.s2^(1/(m-1))
                        u.s4<-aggregate(u.s3,list(rep(1:nrow(v),nrow(v))),sum)
                        u.s4<-u.s4[-1]
                        u.s5<-1/u.s4
                        u<-t(u.s5)
#               }
#               else if(m==1) u<-t(apply(u.s1,1,function(w)w==max(w)))*1
                ### Eliminazione dei cluster a freq. 0 !!!
                c0<-colSums(u)>0
```

```
                if(sum(1-c0)>0) cat("Eliminati cluster a frequenza 0!!!\n")
                u<-u[,c0]
                u.s1<-u.s1[,c0]
                u.m<-u**m
                clusvec<-max.col(u)
                ### Calcolo della matrice dei centroidi V
                v.num<-t(u.m)%*%y
                uno<-matrix(1,nrow(y),ncol(y))
                v.den<-t(u.m)%*%uno
                v<-v.num/v.den
                ### Imputazione dei dati mancanti
                if(nmiss>0){
                        y.imp<-u.m%*%log(v)
                        y[is.na(yorig)]<-y.imp[is.na(yorig)]
                }
                ### Calcolo del valore della funzione J
                jf<-sum(u.m*u.s1)
                if(is.nan(jf)) print(u.s1)
                jfvec<-c(jfvec,jf)
                if(plot) plot(jfvec,type="o",main="Convergenza della funzione
                    obiettivo")
                iter<-iter+1
        }
        ximp<-if(nmiss>0) data.frame(controdisg(y)) else x
        return(list(LogVS=jf,V=v,U=u,CLASSE=clusvec,Iteraz=iter,ximp=ximp))
}
```

```
fckm.impute<-function(x,v,m,metrica=4,soglia=1E-10,maxiter=1000,plot=T)

# Funzione che, in un dataset di variabili categoriali, effettua:
# - la classificazioni delle osservazioni
# - l'imputazione dei valori mancanti
# tramite "conceptual k-means algorithm"
# Necessita delle funzioni:
# 1) disgiunt()
# 2) rimpute()
# 3) distanza()
# 4) controdisg()
# Input:
# 1) x: dataframe. E' il dataset iniziale
# 2) v: matrice. E' la matrice con i valori dei centroidi iniziale
# 3) m: numerico. E' il parametro di fuzzyficazione
# 4) metrica: numerico. E' la distanza scelta. Può assumere i seguenti valori:
#       1- Metrica Euclidea
#       2- Metrica Euclidea per variabili standardizzate
#       3- Metrica di Mahalanobis
#       4- Metrica del Chi quadro
# 5) soglia: numerico. E' lo scarto tollerabile tra i valori della funzione
#            obiettivo calcolata in due iterazioni successive
# 6) maxiter: numerico. E' il numero massimo di iterazioni
# 7) plot: logico. Se T visualizza l'andamento della funzione obiettivo
# Output:
# 1) FunzJ: numerico. E' il valore finale della funzione obiettivo
# 2) V: matrice. E' la matrice con i valori dei centroidi finale
# 3) U: matrice. E' la matrice di appartenenza
# 4) CLASSE: vettore. Assegna un cluster ad ogni osservazione
# 5) Iteraz: numerico. E' il numero di iterazioni necessarie alla convergenza
# 6) ximp: dataframe. E' il dataset con i valori mancanti imputati
```

```
{
        y<-data.matrix(disgiunt(x))
        nmiss<-sum(is.na(y))
        if(nmiss>0){
                yorig<-y
                y<-data.matrix(rimpute(y,1))
        }
        colnames(y)<-colnames(v)
        iter<-0
        jf<-1
        jfold<-0
        jfvec<-c()
        if(plot) windows()
        while ((abs(jfold-jf)>soglia)&(iter<=maxiter)){
                jfold<-jf
                ### Calcolo delle distanze tra Y e V
                dist<-distanza(y,v,metrica)
                cerca<-which(dist<=0,arr.ind=T)
                if(length(cerca)>0) dist[cerca]=1e-10
                ### Calcolo della matrice di appartenenza U
                if(m>1){
                        u.num<-dist**(1/(1-m))
                        u.den<-rowSums(u.num)
                        u<-u.num/u.den
                }
                else if(m==1) u<-t(apply(dist,1,function(w)w==min(w)))*1
                # Eliminazione dei clusters a freq. 0 !!!
                c0<-colSums(u)>0
                if(sum(1-c0)>0) cat("Eliminati cluster a frequenza 0!!!\n")
                u<-u[,c0]
                dist<-dist[,c0]
                u.m<-u**m
                clusvec<-max.col(u)
                ### Calcolo della matrice dei centri V
                v.num<-t(u.m)%*%y
                uno.ij<-matrix(1,nrow(y),ncol(y))
                v.den<-t(u.m)%*%uno.ij
                v<-v.num/v.den
                ### Imputazione dei dati mancanti
                if(nmiss>0){
                        uno.kj<-matrix(1,nrow(v),ncol(v))
                        y.imp<-((u^m)%*%v)/((u^m)%*%uno.kj)
                        y[is.na(yorig)]<-y.imp[is.na(yorig)]
                }
                ### Calcolo del valore della funzione J
                jf<-sum(dist*u.m)
                if(is.nan(jf)|is.na(jf)){
                        print(dist)
                        print(v)
                }
                jfvec<-c(jfvec,jf)
                if(plot) plot(jfvec,type="o",main="Convergenza della funzione
                    obiettivo")
                iter<-iter+1
        }
        ximp<-if(nmiss>0) data.frame(controdisg(y)) else x
        return(list(FunzJ=jf,V=v,U=u,CLASSE=clusvec,Iteraz=iter,ximp=ximp))
}
```

```
fkpop.impute<-function(x,v,m,centri=1,soglia=1E-10,maxiter=1000,plot=T)

# Funzione che, in un dataset di variabili categoriali, effettua:
# - la classificazioni delle osservazioni
# - l'imputazione dei valori mancanti
# tramite "fuzzy k-modes o k-populations algorithm"
# Necessita delle funzioni:
# 1) disgiunt()
# 2) rimpute()
# 3) defuzzy()
# 4) controdisg()
# Input:
# 1) x: dataframe. E' il dataset iniziale
# 2) v: matrice. E' la matrice con i valori dei centroidi iniziale
# 3) m: numerico. E' il parametro di fuzzyficazione
# 4) centri: numerico. Specifica il tipo di centroidi richiesti. Può assumere
    i seguenti valori:
#       1- Centroidi fuzzy (fuzzy k-populations)
#       2- Centroidi crisp (fuzzy k-modes)
# 5) soglia: numerico. E' lo scarto tollerabile tra i valori della funzione
#            obiettivo calcolata in due iterazioni successive
# 6) maxiter: numerico. E' il numero massimo di iterazioni
# 7) plot: logico. Se T visualizza l'andamento della funzione obiettivo
# Output:
# 1) FunzJ: numerico. E' il valore finale della funzione obiettivo
# 2) V: matrice. E' la matrice con i valori dei centroidi finale
# 3) U: matrice. E' la matrice di appartenenza
# 4) CLASSE: vettore. Assegna un cluster ad ogni osservazione
# 5) Iteraz: numerico. E' il numero di iterazioni necessarie alla convergenza
# 6) ximp: dataframe. E' il dataset con i valori mancanti imputati

{
        y<-data.matrix(disgiunt(x))
        nmiss<-sum(is.na(y))
        if(nmiss>0){
                yorig<-y
                y<-data.matrix(rimpute(y,1))
                colnames(y)<-colnames(yorig)
        }
        iter<-0
        jf<-1
        jfold<-0
        jfvec<-c()
        if(plot) windows()
        while ((abs(jfold-jf)>soglia)&(iter<=maxiter)){
                jfold<-jf
                y<-defuzzy(y)
                v<-if(centri==2) defuzzy(v) else v
                ### Calcolo delle distanze tra Y e V
                dist<-(1-y)%*%t(v)
                cerca<-which(dist<=0,arr.ind=T)
                if(length(cerca)>0) dist[cerca]=1e-10
                ### Calcolo della matrice di appartenenza U
                if(m>1){
                        u.num<-dist**(1/(1-m))
                      u.den<-rowSums(u.num)
                        u<-u.num/u.den
                }
```

```
                else if(m==1) u<-t(apply(dist,1,function(w)w==min(w)))*1
                ### Eliminazione dei cluster a freq. 0 !!!
                c0<-colSums(u)>0
                if(sum(1-c0)>0) cat("Eliminati cluster a frequenza 0!!!\n")
                u<-u[,c0]
                dist<-dist[,c0]
                u.m<-u**m
                clusvec<-max.col(u)
                ### Calcolo della matrice dei centroidi V
                v.num<-t(u.m)%*%y
                uno<-matrix(1,nrow(y),ncol(y))
                v.den<-t(u.m)%*%uno
                v<-v.num/v.den
                ### Imputazione dei dati mancanti
                if(nmiss>0){
                        y.imp<-u.m%*%v
                        y[is.na(yorig)]<-y.imp[is.na(yorig)]
                }
                ### Calcolo del valore della funzione J
                jf<-sum(dist*u.m)
                if(is.nan(jf)) print(dist)
                jfvec<-c(jfvec,jf)
                if(plot) plot(jfvec,type="o",main="Convergenza della funzione
                    obiettivo")
                iter<-iter+1
        }
        ximp<-if(nmiss>0) data.frame(controdisg(y)) else x
        return(list(FunzJ=jf,V=v,U=u,CLASSE=clusvec,Iteraz=iter,ximp=ximp))
}
```

```
rcentroid<-function(K=2,X=matrix())

# Funzione che inizializza la matrice JxK dei centri fuzzy in cui
# K è il numero di clusters e J il totale delle modalità di tutte le variabili
# Input:
# 1) K: numerico. E' il numero di clusters
# 2) X: matrice. E' la La matrice dei dati
# Output:
# 1) V: matrice. E' la matrice dei centri fuzzy inizializzata
# NB: Occorre che per ogni colonna j di X siano presenti
#     almeno una volta tutte le modalità della variabile j

{
        numvar<-ncol(X)
        i<-0
        vecvar<-unlist(apply(X,2,function(u){
                i<<-i+1
                nrep<-dim(table(u))
                varnum<-rep(i,nrep)
        }))
        if(length(dim(vecvar))==2) vecvar<-c(vecvar)
        J<-length(vecvar)
        V<-matrix(runif(J*K),J,K)
        rownames(V)<-vecvar
        sumvar<-aggregate(V,list(vecvar),sum)
        repsumvar<-merge(vecvar,sumvar)
        repsumvar<-repsumvar[repsumvar[,1]==repsumvar[,2],3:(K+2)]
        V<-as.matrix(V/repsumvar)
        i<-0
```

```
        etic.col<-unlist(apply(X,2,function(u){
                i<<-i+1
                nrep<-dim(table(u))
                modnames<-names(table(u))
                varnames<-rep(names(X[i]),nrep)
                paste(varnames,modnames,sep=".")
        }))
        if(length(dim(etic.col))==2) etic.col<-c(etic.col)
        rownames(V)<-etic.col
        return(V)
}
```

```
clustverif<-function(original=c(),appart=matrix(),criterio,verbose=T)

# Funzione che verifica la correttezza della classificazione
# ottenuta con un algoritmo di cluster analysis
# Necessita della funzione:
# disgiunt()
# Input:
# 1) original: vettore. E' il vettore con le classi di appartenenza originali
# 2) appart: matrice. E' la matrice con i gradi di appartenenza ai cluster
# 3) Il criterio: stringa. E' il criterio di assegnazione ai clusters. Può
    assumere i seguenti valori:
#                       Crisp. Assegnazione secondo la massima appartenenza
#                       Fuzzy. Assegnata "sfocata" ai diversi clusters
#                       Prob. Assegnazione con probabilità pari alla
    appartenenza
# Output:
# 1) err: numerico. E' il n. di classificazioni errate

{
        numclus<-ncol(appart)
        if(criterio=="Crisp") scosta<-table(cbind(original,data.frame(cluster=
            max.col(appart))))
        if(criterio=="Fuzzy") scosta<-t(as.matrix(disgiunt(original)))%*%as.
            matrix(appart)
        if(criterio=="Prob") scosta<-table(cbind(original,data.frame(cluster=
            apply(appart,1,function(u){sample(1:numclus,1,prob=u)}))))
        if(verbose){
                cat("Matrice degli scostamenti tra cluster effettivi e
                    calcolati - ",criterio,"\n")
                print(scosta)
        }
        conta<-err<-sum(scosta)
        while (conta>0){
                coord<-which(scosta==max(scosta),arr.ind=T)
                if (dim(coord)[1]>1){
                        camp<-sample(c(1:(dim(coord)[1])),1)
                        coord<-coord[camp,]
                }
                giusti<-max(scosta)
                scosta[coord[1],]<-0
                scosta[,coord[2]]<-0
                conta<-sum(scosta)
                err<-err-giusti
        }
        errperc<-err/nrow(original)
        if(verbose){
                cat("N.classificazioni errate: ",err,"\n")
```

```
                cat("Quota di classificazioni errate: ",errperc,"\n")
        }
        return(err)
}
```

```
controdisg<-function(Y)

# Funzione che ricostruisce una matrice di variabili categoriali
# a partire da una matrice in forma disgiuntiva completa
# Input:
# 1) Y: matrice. E' la matrice in forma disgiuntiva completa
# Output:
# 1) X: matrice. E' la matrice di variabili categoriali
# NB: Occorre che Y abbia i nomi delle colonne nella forma variabile.modalità

{
        varsplit<-data.frame(strsplit(colnames(Y),"\\."),stringsAsFactors=
            FALSE)
        vecvar<-as.character((varsplit[1,]))
        vecmod<-as.character((varsplit[2,]))
        varlev<-merge(unique(vecvar),table(vecvar),by.x=1,by.y=1,sort=F)
        uplev<-cumsum(varlev[2])
        lwlev<-uplev+1
        lwlev<-lwlev[-nrow(lwlev),]
        lwlev<-c(1,lwlev)
        X<-matrix(NA,nrow(Y),nrow(varlev))
        rownames(X)<-rownames(Y)
        colnames(X)<-varlev[,1]
        for(j in 1:length(lwlev)) X[,j]<-vecmod[lwlev[j]:uplev[j,]][max.col(Y
            [,lwlev[j]:uplev[j,]])]
        return(X)
}
```

```
defuzzy<-function(Z)

# Funzione che data una matrice Z di I osservazioni per J=L1+...+lj+...+LJ
    colonne
# con Lj n. di modalità della variabile J, tale che:
# - 0<=z(i,j,l)<=1
# - z(i,j,1)+...+z(i,j,l)+...z(i,j,Lj)=1
# genera un matrice in forma disgiuntiva completa Y tale che:
#           1, se  z(i,j,l)=max[h]{z(i,j,h)}
#          /
# y(i,j,l)=
#          \
#           0, se  z(i,j,l)<max[h]{z(i,j,h)}
# Input:
# 1) Z: matrice. E' la matrice dei dati iniziale
# Output:
# 1) Y: matrice. E' la matrice in forma disgiuntiva completa
# NB: Occorre che Z abbia i nomi delle colonne nella forma variabile.modalità

{
        varsplit<-data.frame(strsplit(colnames(Z),"\\."),stringsAsFactors =
            FALSE)
        vecvar<-as.character((varsplit[1,]))
        vecmod<-as.character((varsplit[2,]))
        num.liv<-merge(unique(vecvar),table(vecvar),by.x=1,by.y=1,sort=F)
```

```
        num.liv<-unlist(num.liv[2])
        test<-length(unique(num.liv))
        x.max<-apply(Z,1,function(u){tapply(u,vecvar,max)})
        x.max<-merge(unique(vecvar),x.max,by.x=1,by.y=0,sort=F)
        x.max<-x.max[-1]
        x.max<-t(x.max)
        i<-0
        y.max<-apply(x.max,2,function(u){
                i<<-i+1
                matrix(rep(u,num.liv[i]),length(u),num.liv[i])
        })
        y.max<-if (test>1) data.frame(y.max) else data.frame(matrix(c(y.max),
            nrow(Z),sum(num.liv)))
        Y<-(Z==y.max)*1
        colnames(Y)<-colnames(Z)
        rownames(Y)<-rownames(Z)
        return(Y)
}
```

```
disgiunt<-function(X)

# Funzione che genera una matrice in forma disgiuntiva completa
# partendo da un dataset di variabili categoriali
# Input:
# 1) X: dataframe. E' il dataset di variabili categoriali
# # Output:
# 1) Y: dataframe. E' la matrice in forma disgiuntiva completa
# NB: Occorre che per ogni colonna j di X siano presenti
#     almeno una volta tutte le modalità della variabile j


{
        numliv<-apply(X,2,function(u)nlevels(as.factor(u)))
        test<-length(unique(numliv))
        Y<-apply(X,2,function(u){
                row<-length(u)
                dic<-matrix(rep(levels(as.factor(u)),row),row,nlevels(as.
                    factor(u)),byrow=T)
                (u==dic)*1
        })
        if (test>1) Y<-data.frame(Y)
        else Y<-data.frame(matrix(c(Y),nrow(X),sum(numliv)))

        i<-0
        etic.col<-unlist(apply(X,2,function(u){
                i<<-i+1
                nrep<-dim(table(u))
                modnames<-names(table(u))
                varnames<-rep(names(X[i]),nrep)
                paste(varnames,modnames,sep=".")
        }))
        colnames(Y)<-etic.col
        return(Y)
}
```

```
distanza<-function(M1,M2,metrica=1)

# Funzione che genera una matrice delle distanze tra le righe di
```

```
# due matrici iniziali
# Input:
# 1) M1: matrice. Dataset di n osservazioni e p variabili
# 2) M2: matrice. Dataset di k osservazioni e p variabili
# 3) metrica: numerico. Può assumere i seguenti valori:
#       1- Metrica Euclidea
#       2- Metrica Euclidea per variabili standardizzate
#       3- Metrica di Mahalanobis
#       4- Metrica del Chi quadro se M1 e M2 sono in forma disgiuntiva
    completa
# Output:
# 1) D: matrice. E' la matrice delle distanze

{
        M1<-as.matrix(M1)
        M2<-as.matrix(M2)
        n<-nrow(M1)
        p<-ncol(M1)
        k<-nrow(M2)
        U.np<-matrix(1,n,p)
        S<-M1-U.np%*%diag(apply(M1,2,mean))
        V<-(t(S)%*%S)/n
        N<-diag(colSums(M1))
        if (metrica==1) A<-diag(1,p)
        else if (metrica==2) A<-solve(diag(diag(V)))
        else if (metrica==3) A<-solve(V)
        else if (metrica==4) A<-solve(N)
        D1<-diag(diag(M1%*%A%*%t(M1)))
        M1M2<-M1%*%A%*%t(M2)
        D2<-diag(diag(M2%*%A%*%t(M2)))
        U.nk<-matrix(1,n,k)
        D<-(D1%*%U.nk)-(2*M1M2)+(U.nk%*%D2)
        return(D)
}
```

B.2.2 Modulo per l'imputazione tramite reti bayesiane

```
# Programma che effettua l'imputazione dei valori mancanti
# in dataset di variabili categoriali
# mediante reti bayesiane

#--------------------------------#
# Richiesta a video di parametri #
#--------------------------------#


esisteverodag<-winDialog(type = "yesno", "Esiste il file Hugin (.net) con il
    DAG originale?")
if(esisteverodag=="YES"){
        estensioni<-matrix(c("Hugin Files (*.net)","*.net","All files (*.*)","
            *.*"),2,2,byrow=T)
        nomehugin<-choose.files(caption="Seleziona il file .net",filters=
            estensioni)
        verodag<-legginet(file(nomehugin))[[1]]
        veropotential<-if(esisteorig=="YES") condprob(verodag,xcomp) else
            condprob(verodag,x)
        verobic<-bicfunc(veropotential)
```

```
        windows()
        graf(verodag,"Struttura originale")
}

numgraf<-2^(2*choose(ncol(x),2))
if(ncol(x)>0){
        f<-rep(0,ncol(x))
        for(n in 1:ncol(x)){
                for(i in 1:n){
                        if((n-i>0)) f[n]<-f[n]+(-1)^(i+1)*choose(n,i)*2^(i*(n-
                            i))*f[n-i] else f[n]<-f[n]+(-1)^(i+1)
                }
        }
        numdag<-f[length(f)]
}
if(ncol(x)==0) numdag<-1
opzioni1<-c("Algoritmi Genetici","Algoritmo Greedy Search","Hugin file")
label1<-c("GA","GS","HF")
opta1<-menu(opzioni1,graphics=TRUE,title="Tecnica di Structural learning")
opzioni3<-c("Akaike's information criterion","Bayesian information criterion")
label3<-c("AIC","BIC")
opta3<-menu(opzioni3,graphics=TRUE,title="Funzione di Score")
opzioni2<-c("Nodi Genitori","Markov Blanket")
label2<-c("NG","MB","--")
opta2<-if(quotamiss>0) menu(opzioni2,graphics=TRUE,title="Criterio di
    imputazione") else 3

if(opta1==1){
        prob.mut<-as.numeric(winDialogString("Probabilità di mutazione?","0.4"
            ))
        num.iter<-as.numeric(winDialogString("N. generazioni?","100"))
        dim.pop<-as.numeric(winDialogString("Dimensione del campione?",as.
            character(10)))
        elite<-as.numeric(winDialogString("N. di cromosomi elitari?",as.
            character(1)))
        zerouno<-as.numeric(winDialogString("Rapporto tra 0 e 1?",as.character
            (10)))
}
if(opta1==2){
        elemaxpar<-c(1:min((ncol(x)-1),10))
        maxpar<-as.numeric(menu(as.character(elemaxpar),graphics=TRUE,title="
            Max num. di genitori per il DAG iniziale"))
}
if(opta1==3) dag<-verodag

#---------------------------#
# Stampa delle informazioni #
#---------------------------#

stampa1<-function(){
        cat(paste("N. grafi possibili:",numgraf),"\n")
        cat("N. dag possibili:",numdag,"\n")
        cat("Rapporto dag/grafi:",numdag/numgraf,"\n")
        cat("Tecnica di Structural Learning:",opzioni1[opta1],"\n")
        cat("Funzione di score:",opzioni3[opta3],"\n")
        if(quotamiss>0) cat("Criterio di imputazione:",opzioni2[opta2],"\n")
        if(opta1==1){
                cat("Probabilità di mutazione:",prob.mut,"\n")
                cat("N. generazioni:",num.iter,"\n")
```

```
                cat("Dimensione del campione:",dim.pop,"\n")
                cat("N. di cromosomi elitari:",elite,"\n")
        cat("Rapporto tra 0 e 1:",zerouno,"\n")
        }
        if(opta1==2) cat("Max num. di genitori per il DAG iniziale:",maxpar,"\
            n")
        if(esisteverodag=="YES"){
                cat("Percorso e nome del file Hugin originale:\n")
                cat(nomehugin,"\n")
        }
}

#-------------------------------#
# Imputazione dei dati mancanti #
#-------------------------------#

imputazione<-function(){
        if(as.numeric(R.Version()$minor)>=7) pb<-winProgressBar(opzioni0[opta0
            ],paste("0 prove di",maxprove,"completate"),min=0,max=maxprove,
            initial=0)
        if(esisteverodag=="YES"){
                ris.bn<<-matrix(NA,maxprove,5)
                colnames(ris.bn)<<-c("qc","qa","qi","qer","kl")
        }
        ximp.list<-list()
        cpt.list<-list()
        dag.list<-list()
        for(i in 1:maxprove){
                if(opta1==1) dag<-if(maxprove>1) ga(x,dim.pop,elite,num.iter,
                    prob.mut,zerouno,label3[opta3],monit=F) else ga(x,dim.pop,
                    elite,num.iter,prob.mut,zerouno,label3[opta3],monit=T)
                if(opta1==2){
                        if(maxprove>1){
                                dag<-generadag(colnames(x),opta=2,maxpar=
                                    maxpar)
                                dag<-gs(x,dag,label3[opta3],0)
                        }
                        else{
                                dag<-generadag(colnames(x),opta=0,maxpar=
                                    maxpar)
                                dag<-gs(x,dag,label3[opta3],1)
                        }
                }
                if(esisteverodag=="YES") ris.bn[i,1:4]<<-unlist(qce(verodag,
                    dag))[6:9]
                potential<-condprob(dag,x)
                bic<-bicfunc(potential)
                if(maxprove==1){
                        stampapot<-winDialog(type = "yesno", "Stampare Tabelle
                             di Probabilità Condizionali?")
                        if (stampapot=="YES"){
                                cat("Tabelle di probabilità condizionali:","\n
                                    ")
                                print(potential)
                        }
                }
                if(esisteverodag=="YES") ris.bn[i,5]<<-klbn(potential,
                    veropotential)
                if(quotamiss>0){
```

```
                        if(opta2==1) ximp.list[[i]]<-bnimpute(x,potential,dag)
                        if(opta2==2) ximp.list[[i]]<-mbimpute(x,dag)
                        if(opta2==3) ximp.list[[i]]<-x
                }
                cpt.list[[i]]<-potential
                dag.list[[i]]<-dag
                if(as.numeric(R.Version()$minor)>=7) setWinProgressBar(pb,i,
                    opzioni0[opta0],paste(i,"prove di",maxprove,"completate"))
                     else cat(paste(i,"prove di",maxprove,"completate"),"\n")
                if(i>1 & esisteverodag=="YES") istogrammi(ris.bn)
        }
        if(as.numeric(R.Version()$minor)>=7) close(pb)
        return(list(ximp.list,cpt.list,dag.list))
}
```

B.2.2.1 Funzioni di supporto al programma "BN Impute.R"

```
gs<-function(x=data.frame(),dag=matrix(),funzscore="BIC",visualizza=2)

# Funzione che usa un algoritmo di tipo graeedy search per l'apprendimento di
    una rete bayesiana
# Necessita delle funzioni:
# 1) goodenum()
# 2) condprob()
# 3) bicfunc()
# 4) graf()
# Input:
# 1) x: dataframe. E' un dataset di J variabili categoriali
# 2) dag: matrice. E' la matrice quadrata delle connessioni iniziali tra i J
    nodi
# 3) funzscore: alfanumerico. E' la funzione di fitness scelta. Può assumere i
     seguenti valori:
#            A- Akaike's information criterion
#            B- Bayesian information criterion
# 4) visualizza: numerico. Può assumere i seguenti valori:
#                    0- Non visualizza grafici
#                    1- Mostra ad ogni iterazione, la struttura individuata
#                    2- Mostra ad ogni iterazione, l'andamento del BIC
# Output:
# 1) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi

{
        compleuno<-function(u){
                w<-if(u==0) 1 else 0
                return(w)
        }
        potential<-condprob(dag,x)
        coord<-expand.grid(1:nrow(dag),1:ncol(dag))
        coord<-coord[coord[,1]!=coord[,2],]
        if(visualizza==1 | visualizza==2) windows()
        verifica<-1
        vecbic<-c()
        while(verifica>0){
                bic0<-bicfunc(potential,funzscore)
                bic.mat<-matrix(NA,nrow(dag),ncol(dag))
                delta.mat<-matrix(0,nrow(dag),ncol(dag))
                for(i in 1:nrow(coord)){
```

```
                        dag.new<-dag
                        potential.new<-potential
                        dag.new[coord[i,1],coord[i,2]]<-compleuno(dag[coord[i
                            ,1],coord[i,2]])
                        if(goodenum(dag.new)[[1]]){
                                potential.new[[coord[i,2]]]<-condprob(dag.new,
                                    x,coord[i,2])
                                bic.mat[coord[i,1],coord[i,2]]<-bicfunc(
                                    potential.new,funzscore)
                                delta.mat[coord[i,1],coord[i,2]]<-bic.mat[
                                    coord[i,1],coord[i,2]]-bic0
                        }
                }
                verifica<-sum(delta.mat>0,na.rm=T)
                if(verifica>0) {
                        maxcoord<-which(delta.mat==max(delta.mat),arr.ind=T)
                        maxcoord<-maxcoord[sample(1:nrow(maxcoord),1,prob=rep
                            (1,nrow(maxcoord))),]
                        dag[maxcoord[1],maxcoord[2]]<-compleuno(dag[maxcoord
                            [1],maxcoord[2]])
                        potential<-condprob(dag,x)
                        vecbic<-c(vecbic,bic.mat[maxcoord[1],maxcoord[2]])
                        if (visualizza==1) plot(vecbic,type="o",main=paste("
                            Andamento della funzione",funzscore))
                        if (visualizza==2) graf(dag,"Struttura individuata")
                }
        }
        if(visualizza==1 | visualizza==2) windows()
        if (visualizza==2) plot(vecbic,type="o",main=paste("Andamento della
            funzione",funzscore))
        if (visualizza==1) graf(dag,"Struttura individuata")
        return(dag)
}
```

```
library(genalg)

ga<-function(x=data.frame(),numpop=10,elite=2,iter=100,mutation=0.01,z1r=10,
    funzfit="BIC",monit=T)

# Funzione che usa gli algoritmi genetici per l'apprendimento di una rete
    bayesiana
# Necessita delle funzioni:
# 1) goodenum()
# 2) condprob()
# 3) bicfunc()
# 4) rbga.bin() (contenuta nel package "genalg" ad opera di Egon Willighagen)
# 5) graf()
# Input:
# 1) x: dataframe. E' un dataset di J variabili categoriali
# 2) numpop: numerico. E' il numero di cromosomi della popolazione
# 3) elite: numerico. E' il numero di cromosomi elitari
# 4) iter: numerico. E' il numero delle generazioni
# 5) mutation: numerico. E' la probabilità di mutazione
# 6) z1r: numerico. E' il rapporto tra numero di 0 e 1 in un cromosoma
# 7) funzfit: alfanumerico. E' la funzione di fitness scelta. Può assumere i
    seguenti valori:
#                 A- Akaike's information criterion
#                 B- Bayesian information criterion
```

```
# 8) monit: logico. Se T, mostra l'andamento della funzione della funzione
    fitness
# Output:
# 1) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi

{
        numnodi<-ncol(x)
        # Funzione che triangolarizza il cromosoma #
        triang<-function(stringa,nvar,nomivar){
                lung<-length(stringa)
                stringa1<-stringa[1:(lung/2)]
                stringa2<-stringa[((lung/2)+1):lung]
                conn<-matrix(0,nvar,nvar)
                conn[lower.tri(conn)]<-stringa1
                conn[upper.tri(conn)]<-stringa2
                rownames(conn)<-colnames(conn)<-nomivar
                return(conn)
        }
        # Funzione da minimizzare #
        fitness<-function(stringa){
                dag<-triang(stringa,ncol(x),colnames(x))
                verifica<-goodenum(dag)[[1]]
                if(verifica) {
                        potential<-condprob(dag,x)
                        bic<-bicfunc(potential,funzfit)
                }
                else bic<--1e+100
                return(-bic)
        }
        # Funzione per il monitoraggio delle generazioni di cromosomi #
        monitor<-if(monit){
                function(obj) {
                        def<-par(mfrow=c(2,1))
                        ts.plot(obj$best,type="o",col="blue",xlab="Generazioni
                            ",main="Andamento best fitness")
                        ts.plot(obj$mean,type="o",col="red",xlab="Generazioni"
                            ,main="Andamento mean fitness")
                        def
                }
        } else NULL
        # Applicazione della funzione rbga.bin
        risultati=rbga.bin(
                size=2*choose(ncol(x),2),
                popSize=numpop,
                elitism=elite,
                iters=iter,
                mutationChance=mutation,
                zeroToOneRatio=z1r,
                evalFunc=fitness,
                monitorFunc=monitor,
                verbose=F
        )
        stimata<-risultati$population[which.min(risultati$evaluation),]
        dag<-triang(stimata,ncol(x),colnames(x))
        colnames(dag)<-rownames(dag)<-colnames(x)
        if(monit){
                windows()
                plot(risultati)
                windows()
```

```
                graf(dag,"Struttura individuata")
        }
        return(dag)
}
```

```
bicfunc<-function(probc=list(),crit="BIC")

# Funzione che calcola il valore della funzione di score
# Necessita della funzione:
# 1) laplace()
# Input:
# 1) probc: lista. Contiene J dataframe con le tabelle di probabilità
#           condizionali relative a J variabili categoriali
# 1) crit: alfanumerico. Può assumere i seguenti valori:
#            AIC- Akaike's information criterion
#            BIC- Bayesian information criterion
# Output:
# 1) score: numerico. E' il valore della funzione di score

{
        probc<-lapply(probc,laplace)
        nn<-length(probc)
        scorevec<-c()
        dimgvec<-c()
        matrice<-sapply(probc,function(u){
                numcol<-ncol(u)
                if(numcol==3){
                        q<-nlevels(as.factor(unlist(u[1])))
                        N<<-sum(u$Freq)
                        r<-q
                        sn<-sum(u$Freq*log(u$Prob))
                }
                else if(numcol>3){
                        q<-nlevels(as.factor(unlist(u[numcol-3])))
                        r<-nrow(u)/q
                        sn<-sum(u$Freq.x*log(u$Prob))
                }
                g<-q*(r-1)
                return(list(sn,g))
        })
        score<-if(crit=="BIC") sum(unlist(matrice[1,]))-0.5*log(N)*sum(unlist(
            matrice[2,])) else sum(unlist(matrice[1,]))-sum(unlist(matrice
            [2,]))
        return(score)
}
```

```
bnimpute<-function(dati,prob=list(),dag=matrix())

# Funzione che effettua imputazioni in dataset incompleti tramite Reti
    Bayesiane
# Necessita della funzione:
# 1) goodenum()
# Input:
# 1) dati: dataframe. E' il dataset da imputare
# 2) prob: lista. Contiene J dataframe con le tabelle di probabilità
#          condizionali relative a J variabili categoriali
# 3) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi
# Output:
```

```
# 1) dati: dataframe. E' il dataset completato con i valori imputati

{
        R<-nrow(dati)
        C<-ncol(dati)
        ordine<-goodenum(dag)[[2]]
        if(length(prob)==0){
                for(i in 1:C){
                        prob[[i]]<-data.frame(table(dati[i]))
                        prob[[i]]$Prob<-prob[[i]]$Freq/R
                        names(prob)[[i]]<-names(dati[i])
                        names(prob[[i]])[1]<-names(dati[i])
                }
        }
        nomi<-names(dati)
        dati<-dati[,ordine]
        prob<-prob[ordine]
        numpar.vec<-c()
        for (c in 1:C){
                numpar.vec[c]<-if(ncol(prob[[c]])==3) 0 else ncol(prob[[c]])-4
        }
        nacoor<-which(is.na(dati),arr.ind=T)
        for (i in 1:nrow(nacoor)){
                r<-nacoor[i,1]
                c<-nacoor[i,2]
                var<-colnames(dati[c])
                if(numpar.vec[c]>0){
                        par<-names(prob[[c]])[1:numpar.vec[c]]
                        probab<-prob[[c]][,c(var,par,"Prob")]
                        elemento<-dati[r,par]
                        if(numpar.vec[c]==1){profili<-probab[probab[,2]==
                            elemento,]}
                        if(numpar.vec[c]>1){profili<-merge(probab,elemento)}
                        dati[r,c]<-sample(profili[,var],1,prob=profili$Prob)
                }
                else{
                        profili<-prob[[c]][,c(var,"Prob")]
                        dati[r,c]<-sample(profili[,var],1,prob=profili$Prob)
                }
        }
        dati<-dati[,nomi]
        return(dati)
}
```

```
library(cat)

condprob<-function(matcon=matrix,dati=data.frame(),nodo=NULL)

# Funzione che calcola per un nodo specifico o per tutti i nodi
# di una rete bayesiana le probabilità condizionate ai nodi genitori.
# Necessita delle seguenti funzioni contenute nel package "cat" (Joseph L.
    Schafer):
# 1) prelim.cat()
# 2) em.cat()
# Input:
# 1) matcon: matrice. E' la matrice quadrata delle connessioni tra i J nodi
# 2) dati: dataframe. E' un dataset di I osservazioni e J variabili
    categoriali
# 3) nodo: numerico. E il numero d'ordine del nodo j. Se NULL, allora
```

```
#          le tabelle di probabilità condizionali sono calcolate per tutti i
    nodi
# Output:
# 1) tabelle: dataframe o lista di tabelle di probabilità condizionale

{
        cpt<-function(archi=c(),j=2,dati=data.frame()){
                qm<-sum(is.na(dati))/(nrow(dati)*ncol(dati))
                eps<-0.001*qm
                maxits<-round(25/qm)
                N<-nrow(dati)
                par<-which(archi==1)
                if (length(par)>0){
                        num<-data.frame(table(dati[,c(j,par)]))
                        if(qm>0) num$Freq<-c(em.cat(prelim.cat(data.matrix(
                            dati[c(j,par)])),showits=F,eps=eps,maxits=maxits))
                            *N
                        #if(qm>0) num$Freq<-c(em.cat(prelim.cat(data.matrix(
                            dati[c(j,par)])),showits=F))*N
                        den<-aggregate(num$Freq,num[names(par)],sum)
                        names(den)[ncol(den)]<-"Freq"
                        prob<-merge(num,den,by.x=names(num)[2:(length(par)+1)
                            ],by.y=names(den)[1:length(par)])
                        prob$Prob<-(prob$Freq.x)/(prob$Freq.y)
                }
                else{
                        prob<-data.frame(table(dati[j]))
                        names(prob)[1]<-names(dati)[j]
                        if(qm>0) prob$Freq<-prob$Freq*N/sum(prob$Freq)
                        prob$Prob<-(prob$Freq)/(sum(prob$Freq))
                }
                prob$Prob<-ifelse(is.nan(prob$Prob),1/nlevels(dati[,j]),prob$
                    Prob)
                return(prob)
        }
        if(is.null(nodo)){
                i<-0
                tabelle<-apply(matcon,2,function(u){
                        i<<-i+1
                        cpt(u,i,dati)
                })
        } else tabelle<-cpt(matcon[,nodo],nodo,dati)
        return(tabelle)
}
```

```
generadag<-function(nominodi=LETTERS[1:10],opta=0,verbose=F,maxpar=NA){

# Funzione che genera una matrice di connessione relativa ad un DAG
# Necessita delle funzioni:
# 1) legginet()
# 2) goodenum()
# Input:
# 1) nominodi: vettore. Elenco di stringhe con i nomi dei nodi
# 2) opta: numerico. Può assumere i seguenti valori:
#          0 - Apre un menù di scelta
#          1 - Genera una matrice senza connessioni
#          2 - Genera una matrice di connessioni di un DAG casuale
#          3 - Apre l'editor per la costruzione manuale della matrice del DAG
#          4 - Crea la matrice di connessione tramite file Hugin (.net)
```

```
# Output:
# 1) dag: matrice. La matrice di connessione del DAG

        if(opta==0){
                opzioni<-c("DAG senza connessioni","Genera DAG casuale","Edita
                     manualmente matrice DAG","Genera da file .net")
                opta<-menu(opzioni,graphics=TRUE,title="Carica la matrice del
                    DAG")
        }
        if(opta==1){
                numnodi<-length(nominodi)
                dag<-matrix(0,numnodi,numnodi)
        }
        if(opta==2){
                numnodi<-length(nominodi)
                ok<-F
                iter<-0
                while(ok==F){
                        iter<-iter+1
                        dag<-matrix(0,numnodi,numnodi)
                        if(numnodi<8){
                                dag[upper.tri(dag)]<-rbinom(choose(numnodi,2)
                                    ,1,0.5)
                                dag[lower.tri(dag)]<-rbinom(choose(numnodi,2)
                                    ,1,0.5)
                        }
                        else{
                                if(is.na(maxpar)){
                                        elemaxpar<-c(1:min((numnodi-1),10))
                                        maxpar<-as.numeric(menu(as.character(
                                            elemaxpar),graphics=TRUE,title="
                                            Max num. di genitori"))
                                }
                                verifica<-numnodi
                                while(verifica>maxpar){
                                        dag[lower.tri(dag)]<-rbinom(choose(
                                            numnodi,2),1,0.5)
                                        verifica<-max(colSums(dag))
                                }
                        }
                        ok<-goodenum(dag)[[1]]
                }
                if(verbose) cat("N. Iterazioni per generare il DAG casuale:",
                    iter,"\n")
        }
        if(opta==3){
                numnodi<-length(nominodi)
                ok<-F
                while(ok==F){
                        dag<-matrix(0,numnodi,numnodi)
                        rownames(dag)<-colnames(dag)<-nominodi
                        dag<-edit(dag)
                        ok<-goodenum(dag)[[1]]
                        if(!ok) winDialog(type="ok","Il grafo editato non è un
                             DAG")
                }
        }
        if(opta==4){
                estensioni<-matrix(c("Hugin Files (*.net)","*.net","All files
```

```
                    (*.*)","*.*"),2,2,byrow=T)
                nomefile<-choose.files(caption="Seleziona il file .net",
                    filters=estensioni)
                dag<-legginet(file(nomefile))[[1]]
                nominodi<-colnames(dag)
                numnodi<-length(nominodi)
        }
        rownames(dag)<-colnames(dag)<-nominodi
        if(verbose){
                numgraf<-2^(2*choose(numnodi,2))
                cat("N. grafi possibili:",numgraf,"\n")
                if(numnodi>0){
                        f<-rep(0,numnodi)
                        for(n in 1:numnodi){
                                for(i in 1:n){
                                        if((n-i>0)) f[n]<-f[n]+(-1)^(i+1)*
                                            choose(n,i)*2^(i*(n-i))*f[n-i]
                                            else f[n]<-f[n]+(-1)^(i+1)
                                }
                        }
                        numdag<-f[length(f)]
                }
                else numdag<-1
                cat("N. dag possibili:",numdag,"\n")
        }
        return(dag)
}
```

```
goodenum<-function(grafo=matrix()){

# Funzione che verifica se un grafo è un DAG tramite algoritmo di buona
    enumerazione
# Input:
# 1) grafo: matrice. E' una matrice quadrata delle connessioni tra i nodi
# Output:
# 1) tipo: logico. Se T il grafo è un DAG, se F è un CIG
# 2) ordine: vettore. Se il grafo è un DAG, contiene i nomi dei nodi ordinati

        nn<-ncol(grafo)
        elenco<-colnames(grafo)
        ordine<-c()
        while(nn>0){
                if(!is.null(dim(grafo))){
                        vec.bs<-colSums(grafo)
                        if(sum(vec.bs>0)==nn){
                                nn<-0
                                tipo<-F
                        }
                        else{
                                candidati<-which(vec.bs==0)
                                candsel<-if(length(candidati)>1) sample(
                                    candidati,1,prob=rep(1,length(candidati)))
                                     else candidati
                                ordine<-c(ordine,names(vec.bs[candsel]))
                                grafo<-grafo[-candsel,-candsel]
                                nn<-nn-1
                        }
                }
                else{
```

```
                        ordine<-c(ordine,setdiff(elenco,ordine))
                        nn<-0
                        tipo<-T
                }
        }
        return(list(tipo,ordine))
}
```

```
graf<-function(archi=matrix(),titolo="Struttura della rete")

# Funzione che disegna un grafo, data una matrice di connessione
# Input:
# 1) archi: matrice. E' la matrice quadrata delle connessioni tra i J nodi
#                          1, se il nodo i è genitore del nodo j
#                         /
#               archi[i,j]
#                         \
#                          0, se il nodo i non è genitore del nodo j
#               Se archi[i,j]=1 e archi[j,i]=1 => (i) -- (j)
#               Se archi[i,j]=1 e archi[j,i]=0 => (i) -> (j)
#               Se archi[i,j]=0 e archi[j,i]=1 => (i) <- (j)
#               Se archi[i,j]=0 e archi[j,i]=0 => (i)    (j)
# 2) titolo: stringa. E' il titolo del grafico
# Output:
# 1) Una finestra grafica con il grafo

{
        K <-ncol(archi)
        coord<-exp(1i * 2 * pi * (1:K) / K)
        plot(coord,pch=21,cex=4,xlab="Re",ylab="Im",main=titolo)
        text(coord,labels=colnames(archi),col = 2)
        for(i in 1: nrow(archi)){
                for(j in 1: ncol(archi)){
                        if (archi[i,j]==1 & archi[j,i]==1) segments(Re(coord[i
                            ]), Im(coord[i]), Re(coord[j]), Im(coord[j]))
                        if (archi[i,j]==1 & archi[j,i]==0) arrows(Re(coord[i])
                            , Im(coord[i]), Re(coord[j]), Im(coord[j]),length
                            =0.15,angle=15)
                }
        }
}
```

```
klbn<-function(potential,veropotential)

# Funzione che calcola la divergenza di Kullback-Leibler
# tra due liste di tabelle di probabilità condizionali
# Necessita delle funzioni:
# 1) laplace()
# 2) vea()
# Input:
# 1) potential: lista. Contiene J dataframe con le tabelle di probabilità
#                   condizionali relative a J variabili categoriali
# 2) veropotential:  lista. Contiene J dataframe con le tabelle di probabilità
#                          condizionali relative a J variabili categoriali
# Output:
# 1) kl: numerico. E' il valore della divergenza di Kullback-Leibler

{
```

```
        potential<-lapply(potential,laplace)
        veropotential<-lapply(veropotential,laplace)
        par0<-lapply(veropotential,function(u)(setdiff(names(u),c("Freq.x","
            Freq.y","Freq","Prob"))))
        par1<-lapply(potential,function(u)(setdiff(names(u),c("Freq.x","Freq.y
            ","Freq","Prob"))))
        pg00.cong<-lapply(par0,function(u)(vea(u,veropotential)))
        pg01.cong<-lapply(par1,function(u)(vea(u,veropotential)))
        pg00.cond<-veropotential
        pg11.cond<-potential
        klvec<-c()
        for(i in 1:length(potential)){
                klvec[i]<-sum(pg00.cong[[i]]["Prob"]*log(pg00.cond[[i]]["Prob"
                    ]))-sum(pg01.cong[[i]]["Prob"]*log(pg11.cond[[i]]["Prob"])
                    )
        }
        kl<-sum(klvec)
        return(kl)
}
```

```
laplace<-function(cpt=data.frame())

# Funzione che effettua la correzione delle probabilità
# per una tabella di probabilità condizionali con il metodo di di Laplace
# Input:
# 1) cpt: dataframe. E' una tabella di probabilità condizionali per variabili
    categoriali
# Output:
# 1) cpt: dataframe. E' la tabella di probabilità condizionali corrette

{
        nc<-ncol(cpt)
      if(nc==3){
                cpt["Freq"]<-cpt["Freq"]+1
                tot<-sum(cpt["Freq"])
                cpt["Prob"]<-cpt["Freq"]/tot
        }
        else{
                nlev<-nlevels(unlist(cpt[nc-3]))
                cpt["Freq.x"]<-cpt["Freq.x"]+1
                cpt["Freq.y"]<-cpt["Freq.y"]+nlev
                cpt["Prob"]<-cpt["Freq.x"]/cpt["Freq.y"]
        }
        return(cpt)
}
```

```
mb<-function(dag,nodo,verbose=F)

# Funzione che calcola il Markov Blanket per un nodo di un DAG
# Input:
# 1) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi
# 2) nodo: numerico. E il numero d'ordine del nodo j
# 3) verbose: logico. Se T, stampa a video informazioni dettagliate
# Output:
# 1) mb.x: vettore. Contiene i nomi dei nodi appartenenti al MB di j

{
        if(verbose) cat("----\n")
```

```
        x<-colnames(dag)[nodo]
        mb.x<-c()
        if(verbose) cat("Variabile target:",x,"\n")
        par.x<-names(which(dag[,nodo]==1))
        if(length(par.x)>0){
                if(verbose) cat("Genitori di",x,":",par.x,"\n")
                mb.x<-c(mb.x,par.x)
        }
        son.x<-names(which(dag[nodo,]==1))
        if(length(son.x)>0){
                if(verbose) cat("Figli di",x,":",son.x,"\n")
                mb.x<-c(mb.x,son.x)
                parson.x<-if(length(son.x)>1) unique(names(which(dag[,son.x
                    ]==1,arr.ind=T)[,1])) else names(which(dag[,son.x]==1))
                parson.x<-setdiff(parson.x,x)
                parson.x<-setdiff(parson.x,par.x)
                parson.x<-setdiff(parson.x,son.x)
                if(length(parson.x)>0){
                        if(verbose) cat("Altri genitori dei figli di",x,":",
                            parson.x,"\n")
                        mb.x<-c(mb.x,parson.x)
                }
        }
        if(verbose) cat("Markov Blanket di",x,":",mb.x,"\n")
        return(mb.x)
}
```

```
library(cat)

mbimpute<-function(dati=data.frame(),dag=matrix())

# Funzione che effettua imputazioni in dataset incompleti di variabili
    categoriali
# tramite tabelle di probabilità condizionali relative al
# Markov Blanket delle variabili con i valori mancanti
# Necessita delle seguenti funzioni contenute nel package "cat" (Joseph L.
    Schafer):
# 1) prelim.cat()
# 2) em.cat()
# ...e della funzione:
# 1) mb()
# Input:
# 1) dati: dataframe. E' il dataset da imputare
# 2) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi
# Output:
# 1) dati: dataframe. E' il dataset completato con i valori imputati

{
        dati.m<-is.na(dati)*1
        qm<-sum(dati.m)/(nrow(dati)*ncol(dati))
        eps<-0.001*qm
        maxits<-round(5/qm)
        dati.rid<-dati[as.numeric(row.names(unique(data.frame(dati.m)))),]
        N<-nrow(dati)
        #cont<-0
        ris<-apply(dati.rid,1,function(w){
                #cont<<-cont+1
                #print(cont)
                missing<-is.na(w)*1
```

```
        varmiss<-names(dati.rid)[is.na(w)]
        singlemb<-list()
        if(length(varmiss)>0){
                for(i in which(is.na(w))) singlemb[[colnames(dag)[i]]]
                    <-mb(dag,i)
                markovb<-unique(unlist(singlemb[varmiss]))
                markovb<-setdiff(markovb,varmiss)
                if (length(markovb)>0){
                        num<-data.frame(table(dati[,c(varmiss,markovb)
                            ]))
                        num$Freq<-c(em.cat(prelim.cat(data.matrix(dati
                            [c(varmiss,markovb)])),showits=F,eps=eps,
                            maxits=maxits))*N
                        den<-aggregate(num$Freq,num[markovb],sum)
                        names(den)[ncol(den)]<-"Freq"
                        prob<-merge(num,den,by.x=markovb,by.y=markovb)
                        prob$Prob<-(prob$Freq.x)/(prob$Freq.y)
                        prob$Prob<-ifelse(is.nan(prob$Prob),1/prod(
                            apply(dati[varmiss],2,function(u)nlevels(
                            as.factor(u)))),prob$Prob)
                }
                else{
                        if(length(varmiss)>1){
                                prob<-data.frame(table(dati[varmiss]))
                                prob$Freq<-prob$Freq*N/sum(prob$Freq)
                                prob$Prob<-(prob$Freq)/(sum(prob$Freq)
                                    )
                                prob$Prob<-ifelse(is.nan(prob$Prob),1/
                                    prod(apply(dati[varmiss],2,
                                    function(u)nlevels(as.factor(u))))
                                    ,prob$Prob)
                        }
                        else{
                                prob<-data.frame(table(dati[varmiss]))
                                names(prob)[1]<-varmiss
                                prob$Freq<-prob$Freq*N/sum(prob$Freq)
                                prob$Prob<-(prob$Freq)/(sum(prob$Freq)
                                    )
                                prob$Prob<-ifelse(is.nan(prob$Prob),1/
                                    nlevels(dati[varmiss]),prob$Prob)
                        }
                }
                return(list(missing=missing,varmiss=varmiss,markovb=
                    markovb,prob=prob))
        } else return(list(missing=missing,varmiss="",markovb="",prob=
            data.frame()))
})
for(i in 1:N){
        j<-1
        verifica<-0
        while(verifica<1 & j<=length(ris)){
                verifica<-mean(dati.m[i,]==unlist(ris[j][[1]]["missing
                    "]))
                if(verifica==1){
                        if(sum(dati.m[i,])!=0){
                                varmiss<-unlist(ris[j][[1]]["varmiss"
                                    ])
                                markovb<-unlist(ris[j][[1]]["markovb"
                                    ])
```

```
                                        prob<-data.frame(ris[j][[1]]["prob"])
                                        names(prob)<-substr(names(prob),6,500)
                                        if(length(markovb)==0) candidati<-prob
                                        else if(length(markovb)==1) candidati
                                            <-prob[prob[markovb]==as.character
                                            (dati[i,markovb]),]
                                        else if(length(markovb)>1) candidati<-
                                            merge(prob,dati[i,markovb])
                                        probval<-candidati$Prob
                                        valore<-sample(1:length(probval),1,
                                            prob=probval)
                                        estratto<-candidati[valore,varmiss]
                                        dati[i,varmiss]<-estratto
                                    } else j<-j+1
                                } else j<-j+1
                        }
                }
        return(dati)
}
```

```
qce<-function(dag.v,dag.s)

# Funzione che analizza le differenze tra il DAG originale e quello stimato
# Input:
# 1) dag.v: matrice. E' la matrice quadrata delle connessioni tra i nodi
    originale
# 2) dag.s: matrice. E' la matrice quadrata delle connessioni tra i nodi
    stimata
# Output:
# 1) nc: numerico. E' il numero di connessioni cancellate
# 2) na: numerico. E' il numero di connessioni aggiunte
# 3) ni: numerico. E' il numero di connessioni invertite
# 4) ner: numerico. E' il numero di connessioni errate
# 5) nes: numerico. E' il numero di connessioni esatte
# 6) qc: numerico. E' la quota di connessioni cancellate
# 7) qa: numerico. E' la quota di connessioni aggiunte
# 8) qi: numerico. E' la quota di connessioni invertite
# 9) qer: numerico. E' la quota di connessioni errate
# 10) qes: numerico. E' la quota di connessioni esatte

{
        n<-ncol(dag.v)
        tc<-n*(n-1)/2
        ds<-c(dag.s[upper.tri(dag.s)])-c(t(dag.s)[upper.tri(dag.s)])
        dv<-c(dag.v[upper.tri(dag.v)])-c(t(dag.v)[upper.tri(dag.v)])
        nc<-sum(ds!=dv & ds==0)
        na<-sum(ds!=dv & dv==0)
        ni<-sum(ds==-dv & ds!=0)
        ner<-sum(ds!=dv)
        nes<-tc-ner
        qc<-nc/tc
        qa<-na/tc
        qi<-ni/tc
        qer<-ner/tc
        qes<-1-qer
        return(list(nc=nc,na=na,ni=ni,ner=ner,nes=nes,qc=qc,qa=qa,qi=qi,qer=
            qer,qes=qes))
}
```

```
legginet<-function(con=file("default.net"))

# Funzione che converte in oggetti R un file Hugin (.net)
# (adattamento della funzione readnet() contenuta nel package "deal"
# ad opera di S. Gammelgaard Bøttcher e C. Dethlefsen)
# Input:
# 1) con: connessione. Contiene il nome ed il percorso del file Hugin
# Output:
# 1) dag: matrice. E' la matrice quadrata di connessione del DAG
# 2) prob: lista. Contiene J dataframe con le tabelle di probabilità
#          condizionali relative a J variabili categoriali

{
        open(con, "r")
        l<-readLines(con)
        lno<-length(l)
        lcount<-0
        nodes<-list()
        nodecount<-1
        nnames<-c()
        while(lcount<=lno){
                lcount<-lcount+1
                nodeptr<-grep("node ",l[lcount],value=TRUE)
                poteptr<-grep("potential ",l[lcount],value=TRUE)
                if(length(nodeptr)>0){
                        ss<-unlist(strsplit(l[lcount]," "))
                        ss<-ss[ss!=""]
                        nd<-list()
                        nd$idx<-nodecount
                        ssleng<-length(ss)
                        if(ssleng==3){
                                nd$type<-ss[1]
                                nd$name<-ss[3]
                                nnames<-c(nnames,ss[3])
                        }
                        else{
                           nd$type<-"discrete"
                            nd$name<-ss[2]
                           nnames<-c(nnames,ss[2])
                        }
                        if(nd$type=="discrete"){
                                i<-0
                                slut<-FALSE
                                while(!slut){
                                        i<-i+1
                                        statestr<-grep("states",l[lcount+i],
                                            value=TRUE)
                                        if(length(statestr)>0) slut<-TRUE
                                }
                                c1<-regexpr("[(]",statestr)
                                x<-substr(statestr,c1+1,nchar(statestr)-2)
                                x<-gsub("\" ", ";", x)
                                x<-gsub("\"","",x)
                                y<-unlist(strsplit(x,";"))
                                y<-y[y!=""]
                                y<-gsub(" ","",y)
                                y<-make.names(y)
                                nd$levelnames<-y
```

```
                        nd$levels<-length(nd$levelnames)
                }
                class(nd)<-"node"
                nodes[[nodecount]]<-nd
                nodecount<-nodecount+1
        }
        if(length(poteptr)>0){
                str<-poteptr
                c1<-regexpr("[(]",str)
                c2<-regexpr("[)]",str)
                x<-substr(str,c1+1,c2-1)
                c3<-regexpr("[|]",x)
                if(c3==-1){
                        x<-gsub(" ","",x)
                        nodenumber<-match(x,nnames)
                        nodes[[nodenumber]]$parents<-c()
                        step0<-strsplit(l[lcount+2],"%")
                        step1<-unlist(strsplit(unlist(step0[[1]][1]),"
                             "))
                        step2<-as.numeric(step1)
                        step3<-step2[complete.cases(step2)]
                        nodes[[nodenumber]]$prob <-step3
                        names(nodes[[nodenumber]]$prob)<-nodes[[
                            nodenumber]]$levelnames
                }
                else{
                        lhs <- gsub(" ", "", substr(x, 1, c3 - 1))
                        nodenumber <- match(lhs, nnames)
                        rhs <- substr(x, c3 + 1, nchar(x))
                        rhsy <- unlist(strsplit(rhs, " "))
                        rhsy <- rhsy[rhsy != ""]
                        parents <- match(rhsy, nnames)
                        nodes[[nodenumber]]$parents <- parents
                        parlevnames<-list()
                        for(numpar in 1:length(nodes[[nodenumber]]$
                            parents)){
                                parlevnames[[numpar]]<-nodes[[(nodes[[
                                    nodenumber]]$parents[[numpar]])]]$
                                    levelnames
                        }
                        nodes[[nodenumber]]$prob<-c()
                        riga<-1
                        graffa<-c()
                        puntovirg<-c()
                        while(length(graffa)==0 & length(puntovirg)
                            ==0){
                                if(length(grep("%",l[lcount+riga]))>0)
                                    {
                                        step0<-strsplit(l[lcount+riga
                                            ],"%")
                                        step1<-unlist(strsplit(unlist(
                                            step0[[1]][1])," "))
                                        step2<-as.numeric(step1)
                                        step3<-step2[complete.cases(
                                            step2)]
                                        nodes[[nodenumber]]$prob<-
                                            rbind(nodes[[nodenumber]]$
                                            prob,step3,deparse.level
                                            =0)
```

```
                                        }
                                        puntovirg<-grep(";",l[lcount+riga])
                                        riga<-riga+1
                                        graffa<-grep("}",l[lcount+riga])
                                }
                                profili<-expand.grid(parlevnames[length(
                                    parlevnames):1])
                                colnames(profili)<-nnames[parents[length(
                                    parents):1]]
                                colnames(nodes[[nodenumber]]$prob)<-nodes[[
                                    nodenumber]]$levelnames
                                nodes[[nodenumber]]$prob<-data.frame(nodes[[
                                    nodenumber]]$prob,profili)
                        }
                }
        }
        nw<-list()
        names(nodes)<-nnames
        nw$nodes<-nodes
        nw$n<-length(nodes)
        ltype<-unlist(lapply(nw$nodes,function(x) x$type))
        nw$discrete<-(1:nw$n)[ltype=="discrete"]
        nw$continuous<-(1:nw$n)[ltype=="continuous"]
        nw$nd<-length(nw$discrete)
        nw$nc<-length(nw$continuous)
        nw$dag<-matrix(0,nw$n,nw$n)
        rownames(nw$dag)<-colnames(nw$dag)<-nnames
        for(i in 1:nw$n){
               par<-nodes[[i]]$parents
               nw$dag[par,i]<-1
        }
        nw$prob<-list()
        for(i in 1:nw$n){
                nw$prob[[i]]<-nodes[[i]]$prob
                names(nw$prob)[i]<-nnames[i]
        }
        class(nw)<-"network"
        close(con)
        return(list(dag=nw$dag,cpt=nw$prob))
}
```

```
vea<-function(nodi=c(),cpt=list())

# Funzione che calcola la distribuzione di probabilità congiunta
# per nn nodi, mediante l'algoritmo di eliminazione variabili
# Input:
# 1) nodi: vettore. E' un elenco di nn<=J variabili categoriali
# 2) cpt: lista. Contiene J dataframe con le tabelle di probabilità
#         condizionali relative a J variabili categoriali
# Output:
# 1) prcong: dataframe. Contiene la distribuzione di probabilità congiunta
#                per gli nn nodi

{
        nn<-length(nodi)
        prob<-list()
        for(i in 1:nn){
                var<-nodi[i]
                numcol<-ncol(cpt[[var]])
```

```
                if(numcol==3){
                        if(i==1) prcong<-cpt[[var]]
                        else{
                                prcong<-merge(prcong,cpt[[var]][1:2])
                                prcong["Prob"]<-prcong["Prob"]*(prcong["Freq"]
                                    /sum(cpt[[var]]["Freq"]))
                        }
                        prcong<-prcong[setdiff(names(prcong),"Freq")]
                }
                else{
                        nodi.int<-intersect(names(cpt[[var]]),nodi)
                        prob[[var]]<-aggregate(cpt[[var]][c("Freq.x","Freq.y")
                            ],cpt[[var]][nodi.int],sum)
                        if(i==1){
                                prcong<-prob[[var]]
                                prcong["Prob"]<-prcong["Freq.x"]/prcong["Freq.
                                    y"]

                        }
                        else{
                                prcong<-merge(prcong,prob[[var]])
                                prcong["Prob"]<-prcong["Prob"]*(prcong["Freq.x
                                    "]/prcong["Freq.y"])

                        }
                        prcong<-prcong[setdiff(names(prcong),c("Freq.x","Freq.
                            y"))]
                }
                prcong["Prob"]<-ifelse(is.nan(prcong[,"Prob"]),0,prcong[,"Prob
                    "])
        }
        prcong<-prcong[do.call(order,prcong[nodi]),]
        return(prcong)
}
```

B.2.3 Modulo per l'imputazione tramite algoritmo EM

```
# Programma che effettua l'imputazione dei valori mancanti
# in dataset di variabili categoriali
# mediante algoritmo EM

opzioni1<-label1<-c("MLS","MLH")
opta1<-1
opzioni2<-label2<-c("MCMCG","MCMCM","--")
opta2<-if(quotamiss>0) 1 else 3

library(cat)
xn<-data.matrix(x)
xp<-prelim.cat(xn)
thetahat<-em.cat(xp)
rngseed(runif(1,1000,100000))

listlev<-apply(x,2,function(u)levels(as.factor(u)))

uncategorize<-function(data.mat,list.lev){
        data.mat<-data.frame(data.mat)
        nc<-ncol(data.mat)
```

```
        if(class(list.lev)=="matrix"){
                for(i in 1:nc) data.mat[,i]<-factor(data.mat[,i],labels=list.
                    lev[,i])
                colnames(data.mat)<-colnames(list.lev)
        }
        else for(i in 1:nc){
                data.mat[,i]<-factor(data.mat[,i],labels=list.lev[[i]])
                colnames(data.mat)<-names(list.lev)
        }
        return(data.mat)
}

#---------------------------#
# Stampa delle informazioni #
#---------------------------#

stampa1<-function(){
        cat("Tecnica di stima:",opzioni1[opta1],"\n")
        if(quotamiss>0) cat("Tecnica di campionamento:",opzioni2[opta2],"\n")
}



#-------------------------------#
# Imputazione dei dati mancanti #
#-------------------------------#

imputazione<-function(){
        if(as.numeric(R.Version()$minor)>=7) pb<-winProgressBar(opzioni0[opta0
            ],paste("0 prove di",maxprove,"completate"),min=0,max=maxprove,
            initial=0)
        ximp.list<-list()
        theta.list<-list()
        for(i in 1:maxprove){
                theta<-da.cat(xp,start=thetahat,steps=50)
                ximpn<-imp.cat(xp,theta)
                ximp.list[[i]]<-uncategorize(ximpn,listlev)
                theta.list[[i]]<-theta
                if(as.numeric(R.Version()$minor)>=7) setWinProgressBar(pb,i,
                    opzioni0[opta0],paste(i,"prove di",maxprove,"completate"))
                     else cat(paste(i,"prove di",maxprove,"completate"),"\n")

        }
        if(as.numeric(R.Version()$minor)>=7) close(pb)
        return(list(ximp.list,theta.list,thetahat))
}
```

B.2.4 Funzioni di supporto al programma di imputazione

```
require(tcltk)
carica<-function(messaggio="Definisci le caratteristiche del dataset",nomedati
    ="dataset",nomeclas="classe",numeroclus="numclus")

# Funzione che consente l'importazione di file ASCII in dataframe
# specificando le opzioni tramite interfaccia grafico
# Necessita della libreria:
# 1) tcltk
```

```
# Input:
# 1) messaggio: stringa. Contiene un messaggio che apparirà nell'interfaccia
    grafica
# 2) nomedati: stringa. E' il nome da assegnare al dataframe di output
# 3) nomeclas: stringa. E' il nome del vettore contenente
#              l'eventuale variabile di classificazione
# 4) numeroclus: stringa. E' il nome della var. numerica contenente
#                il numero di clusters eventualmente individuati
# Output:
# 1) dati: dataset. Contiene il file ASCII importato
# 2) clus: vettore. Contiene l'eventuale colonna del file ASCII con i valori
#          della variabile di classificazione
# 3) nclus: numerico. Indica il numero di cluster esistenti nella eventuale
#           variabile di classificazione

{
        interfaccia<-function(){
                name<<-""
                tt<-tktoplevel()
                tktitle(tt)<-"Importazione di un dataset ASCII"
                frameOverall<-tkframe(tt)
                tkgrid(tklabel(frameOverall,text=messaggio),sticky="w")
                tkgrid(frameOverall)
                frame3<-tkframe(frameOverall,relief="groove",borderwidth=2)
                rb31<-tkradiobutton(tt)
                rb32<-tkradiobutton(tt)
                rb33<-tkradiobutton(tt)
                rb34<-tkradiobutton(tt)
                rbValue3<-tclVar(";")
                tkconfigure(rb31,variable=rbValue3,value=".")
                tkconfigure(rb32,variable=rbValue3,value=",")
                tkconfigure(rb33,variable=rbValue3,value=";")
                tkconfigure(rb34,variable=rbValue3,value="")
                tkgrid(tklabel(frame3,text="Separatore colonne"))
                tkgrid(tklabel(frame3,text="Punto"),rb31)
                tkgrid(tklabel(frame3,text="Virgola"),rb32)
                tkgrid(tklabel(frame3,text="Punto e virgola"),rb33)
                tkgrid(tklabel(frame3,text="Spazio o Tab"),rb34)
                rbVal3<-tclvalue(rbValue3)
                frame4<-tkframe(frameOverall,relief="groove",borderwidth=2)
                rb41<-tkradiobutton(tt)
                rb42<-tkradiobutton(tt)
                rbValue4<-tclVar(".")
                tkconfigure(rb41,variable=rbValue4,value=".")
                tkconfigure(rb42,variable=rbValue4,value=",")
                tkgrid(tklabel(frame4,text="Separatore decimale"))
                tkgrid(tklabel(frame4,text="Punto"),rb41)
                tkgrid(tklabel(frame4,text="Virgola"),rb42)
                rbVal4<-tclvalue(rbValue4)
                tkgrid(frame3,frame4,sticky="nsew")
                cb1<-tkcheckbutton(tt)
                cbValue1<-tclVar("1")
                tkconfigure(cb1,variable=cbValue1)
                tkgrid(tklabel(tt,text="Intestazione Colonne"),cb1,sticky="we"
                    )
                cb2<-tkcheckbutton(tt)
                cbValue2<-tclVar("1")
                tkconfigure(cb2,variable=cbValue2)
                tkgrid(tklabel(tt,text="Intestazione righe"),cb2,sticky="we")
```

```
cb3<-tkcheckbutton(tt)
cbValue3<-tclVar("0")
tkconfigure(cb3,variable=cbValue3)
tkgrid(tklabel(tt,text="Variabile di classificazione"),cb3,
    sticky="we")
cb4<-tkcheckbutton(tt)
cbValue4<-tclVar("0")
tkconfigure(cb4,variable=cbValue4)
tkgrid(tklabel(tt,text="Leggi valori numerici come modalità"),
    cb4,sticky="we")
OnOpen<-function(){
        tkconfigure(txt, state="normal")
        tkdelete(txt,"0.0", "end")
        name<<-tclvalue(tkgetOpenFile(filetypes="
                {{ASCII Files} {.txt .dat .csv}}
                {{Txt Files} {.txt}}
                {{Csv Files} {.csv}}
                {{Dat Files} {.dat}}
                {{All files} *}"
        ))
        dati<-readLines(name)
        for(i in 1:length(dati))tkinsert(txt,"end",paste(dati[
            i],"\n"))
        tkconfigure(txt, state="disabled")
        tkfocus(tt)
}
OnOK<-function(){
        row.names<-NULL
        cbVal1<-as.logical(as.numeric(tclvalue(cbValue1)))
        cbVal2<-as.numeric(tclvalue(cbValue2))
        rbVal3<-tclvalue(rbValue3)
        rbVal4<-tclvalue(rbValue4)
        cbVal3<-as.logical(as.numeric(tclvalue(cbValue3)))
        cbVal4<-as.logical(as.numeric(tclvalue(cbValue4)))
        if(cbVal2>0) row.names<-cbVal2
        if (!nchar(name)){
                tkmessageBox(title="Errore!",message="Non è
                    stato selezionato nessun dataset",icon="
                    error",type="ok")
                tkfocus(tt)
        }
        else{
                dati<-read.table(name,sep=rbVal3,dec=rbVal4,
                    header=cbVal1,row.names=row.names)
                dati<-data.frame(dati)
                if(cbVal4){
                        for(j in 1:ncol(dati)) if(is.numeric(
                            dati[,j])) dati[,j]<-as.factor(
                            dati[,j])
                }
                if(cbVal3){
                        colclus<-as.numeric(menu(colnames(dati
                            ),graphics=TRUE,title="Variabile
                            classe"))
                        clas<-dati[,colclus]
                        dati<-dati[,-colclus]
                }
                else clas<-rep(NA,nrow(dati))
                clas<-data.frame(clas)
```

```
                                nclus<-nlevels(as.factor(clas[,1]))
                                assign(nomedati,dati, envir = .GlobalEnv)
                                assign(nomeclas,clas, envir = .GlobalEnv)
                                assign(numeroclus,nclus, envir = .GlobalEnv)
                                tkdestroy(tt)
                                cond1<<-F
                        }
                }
                Open.but <- tkbutton(tt,text="Seleziona il dataset ASCII",
                    command=OnOpen)
                tkgrid(Open.but,sticky="ew")
                xscr<-tkscrollbar(tt, repeatinterval=5,orient="horizontal",
                    command=function(...)tkxview(txt,...))
                yscr<-tkscrollbar(tt, repeatinterval=5,command=function(...)
                    tkyview(txt,...))
                txt<-tktext(tt,bg="white",font="courier",xscrollcommand=
                    function(...)tkset(xscr,...),yscrollcommand=function(...)
                    tkset(yscr,...),wrap="none")
                tkconfigure(txt,height=5,width=31)
                tkgrid(txt,yscr)
                tkgrid(xscr)
                tkgrid.configure(yscr,sticky="ns")
                tkgrid.configure(xscr,sticky="ew")
                tkgrid(txt,sticky="w")
                OK.but<-tkbutton(tt,text="Importa",command=OnOK)
                tkgrid(OK.but,sticky="ew")
        }
        cond1<-TRUE
        cond2<-TRUE
        while(cond1){
                if(cond2){
                        interfaccia()
                        cond2<-F
                }
        }
}
```

```
evalimp<-function(d.orig,d.comp,d.imp,dag=matrix())

# Funzione che effettua test diagnostici sulla validità
# dell'imputazione di dataset di variabili categoriali
# I test riguardano:
# 1) L'accuratezza della distribuzione (Di Zio et al. - 2001):
#       - test delta
# 2) La capacità di preservazione dei valori originari (Chambers 2001):
#       - test D
#       - Var(D)
#       - epsilon
# Necessita della funzione:
# condprob()
# Input:
# 1) d.orig: dataframe. E' il dataset con i dati mancanti
# 2) d.comp: dataframe. E' il dataset con i dati completi
# 3) d.imp: dataframe. E' il dataset con i valori mancanti imputati
# 4) dag: matrice. E' la matrice quadrata delle connessioni tra i J nodi
# Output:
# 1) ris: matrice. Contiene i risultati dei test

{
```

```
        ris<-data.frame(delta=NA,D=NA,Var.D=NA,eps=NA)
        if(nrow(dag)>1){
                cpt.comp<-condprob(dag,d.comp)
                cpt.imp<-condprob(dag,d.imp)
        }
        nv<-ncol(d.orig)
        for(i in 1:nv){
                n<-sum(is.na(d.orig[,i]))
                df<-nlevels(as.factor(d.comp[,i]))-1

                fr<-table(d.imp[is.na(d.orig[,i]),i])/n
                fs<-table(d.comp[is.na(d.orig[,i]),i])/n

                delta<-if(nrow(dag)>1) sum(abs(cpt.comp[[i]]$Prob-cpt.imp[[i]]
                    $Prob))/2 else sum(abs(fs-fr))/2

                R<-table(d.imp[,i])
                R<-R[-df]
                S<-table(d.comp[,i])
                S<-S[-df]
                T<-table(d.comp[,i],d.imp[,i])
                T<-T[-df,-df]
                v<-R-S
                D<-1-(sum(d.comp[is.na(d.orig[,i]),i]==d.imp[is.na(d.orig[,i])
                    ,i])/n)
                u<-rep(1,df)
                VD<-(1-D)/n
                eps<-max(0,(D-(2*sqrt(VD))))
                ris[i,]<-c(delta,D,VD,eps)
        }
        rownames(ris)<-colnames(d.orig)
        nt<-sum(is.na(d.orig))
        Dt<-1-(sum(d.comp[is.na(d.orig)]==d.imp[is.na(d.orig)])/nt)
        VDt<-(1-Dt)/nt
        epst<-max(0,(Dt-(2*sqrt(VDt))))

        vecnmiss<-rowSums(is.na(d.orig)*1)
        nm<-nrow(d.orig[vecnmiss>0,])
        n.conf.t<-prod(apply(d.orig,2,function(u)nlevels(as.factor(u))))
        mem.used<-(n.conf.t*8)+32
        if(mem.used/(1024^2)<1000){
                frt<-table(d.imp[vecnmiss>0,])/nm
                fst<-table(d.comp[vecnmiss>0,])/nm
                deltat<-sum(abs(fst-frt))/2
        } else deltat<-mean(ris["delta"])
        ris["DISTRIB_CONG",]<-c(deltat,Dt,VDt,epst)
        return(ris)
}
```

```
istogrammi<-function(matrice=matrix())

# Funzione che disegna una batteria di istogrammi
# Input:
# 1) matrice: matrice. E' la matrice contenente J colonne di valori numerici
# Output:
# 1) Una finestra grafica con J istogrammi

{
        lista<-is.list(matrice)
```

```
        numist<-if(lista) length(matrice) else ncol(matrice)
        lato1<-ceiling(sqrt(numist))
        lato2<-lato1-1
        def<-if((lato1*lato2)>=numist) par(mfrow=c(lato1,lato2)) else par(
            mfrow=c(lato1,lato1))
        for(i in 1:numist){
                if(lista){
                        hist(matrice[[i]],probability=TRUE,main=names(matrice)
                            [[i]])
                        lines(density(matrice[[i]]),col="blue")
                }
                else{
                        numobs<-sum(!is.na(matrice[,i]))
                        hist(matrice[,i],probability=TRUE,main=paste(colnames(
                            matrice)[i],numobs,"osservazioni"))
                        lines(density(matrice[!is.na(matrice[,i]),i]),col="
                            blue")
                }
        }
        def
}
```

```
rimpute<-function(dati=matrice(),tipo=1)

# Funzione che imputa casualmente i valori mancanti di una matrice
# Input:
# 1) dati: matrice o dataframe. E' la matrice o il dataframe dei dati
# 2) tipo: numerico. Può assumere i seguenti valori:
#                    1- Se dati è una matrice di variabili numeriche
#                    2- Se dati è una matrice di variabili categoriali
# Output:
# 1) dati.imp: dataframe. E' il dataframe completato con i valori imputati
# NB: Nel caso di variabili categoriali, occorre che per ogni colonna j di
    dati
#     siano presenti almeno una volta tutte le modalità della variabile j

{
        if (tipo==1){
                dati.imp<-apply(dati,2,function(u){
                                uimp<-runif(length(u),min(u,na.rm=T),max(u,na.
                                    rm=T))
                                u[is.na(u)]<-uimp[is.na(u)]
                                return(u)
                        })
        }
        if (tipo==2){
                dati.imp<-apply(dati,2,function(u){
                                liv<-levels(as.factor(u))
                                uimp<-sample(liv,length(u),replace=T)
                                u[is.na(u)]<-uimp[is.na(u)]
                                return(u)
                        })
        }
        dati.imp<-data.frame(dati.imp)
        return(dati.imp)
}
```

B.3 Programma per la generazione di dati mancanti

```
# Programma che genera valori mancanti
# in un dataset per un insieme di variabili prescelto

rm(list=ls())

#----------------------------#
# Caricamento delle librerie #
#----------------------------#

savehistory(file = ".Rhistory")
datahistory<-readLines(file(".Rhistory"))
close(file(".Rhistory"))
ultimariga<-datahistory[length(datahistory)]
namesplit<-strsplit(ultimariga,"\\\\")
nomeprog<-namesplit[[1]][length(namesplit[[1]])]
percorso<-sub(nomeprog,"",ultimariga,fixed=T)
nomeprog<-sub("\"\\)","",nomeprog)
percorso<-sub("source(\"","",percorso,fixed=T)
percorso<-gsub("\\\\","/",percorso,fixed=T)
cat("Nome del programma:",nomeprog,"\n")
cat("Percorso:\n")
cat(percorso,"\n")

elenco<-c(
        "Funz. Caricamento Dati 5.R",
        "Funz. Istogrammi.R"
)

for (i in 1:length(elenco)) source(paste(percorso,"\\Funzioni\\",elenco[i],sep
    =""))

#----------------------------#
# Caricamento del file ASCII #
#----------------------------#

carica("Caratteristiche del dataset","x")
nummiss<-sum(is.na(x))
quotamiss<-nummiss/(nrow(x)*ncol(x))

#--------------#
# Elaborazioni #
#--------------#

var<-select.list(colnames(x), preselect = NULL, multiple = T, title ="Scegli
    la/e variabile/i da rendere incompleta/e")
varpos<-which(names(x)==var)
numvar<-length(varpos)

xcomp<-x

rango<-min(nrow(x),numvar)
maxpm<-(rango-1)/rango
```

```
pm<-as.numeric(winDialogString(paste("Quota di missing (min=",quotamiss,", max
    =",maxpm,")?",sep=""),as.character(mean(c(quotamiss,maxpm)))))
toll<-as.numeric(winDialogString(paste("Tolleranza?"),"0.001"))

xm<-is.na(x)*1
soglia<-round((nrow(x)*numvar*pm))

iter<-1
vmiss<-c()
vcol<-c()
vrow<-c()
difmiss<-(soglia-sum(is.na(x)))/sum(!is.na(x))
windows()
while(sum(xm)<soglia*(1-toll) | sum(xm)>soglia*(1+toll) | max(rowSums(xm))>(
    ncol(xm)-1) | max(colSums(xm))>(nrow(xm)-1)){
        xm<-is.na(x)*1
        for(i in 1:nrow(xm)){
                if(sum(xm[i,])<(ncol(xm)-1)){
                        for(j in varpos){
                                if(sum(xm[,j])<(nrow(xm)-1)){
                                        xm[i,j]<-if(xm[i,j]==0) rbinom(1,1,
                                            difmiss) else xm[i,j]
                                }
                        }
                }
        }
        #xm<-matrix(rbinom((nrow(x)*numvar),1,pm),nrow(x),numvar)
        vmiss<-c(vmiss,sum(xm))
        #def<-par(mfrow = c(3, 1))
        plot(vmiss,type="b",main="Missing generati per iterazione")
        lines(rep(soglia*(1-toll),length(vmiss)),col="red")
        lines(rep(soglia*(1+toll),length(vmiss)),col="blue")
        #vrow<-c(vrow,max(rowSums(xm)))
        #plot(vrow,type="b",main="Max missing per riga per iterazione")
        #lines(rep((ncol(xm)-1),length(vrow)),col="red")
        #vcol<-c(vrow,max(colSums(xm)))
        #plot(vcol,type="b",main="Max missing per colonna per iterazione")
        #lines(rep((nrow(xm)-1),length(vcol)),col="red")
        #def
        iter<-iter+1
}

for(i in 1:numvar) x[xm[,i]==1,varpos[i]]<-NA

# Verifiche

tipovar<-c("Categoriali","Numeriche")
optavar<-menu(tipovar,graphics=TRUE,title="Tipologia di variabili")
if (optavar==1){
        fattorizza<-function(dati=matrix){
                fatt<-data.frame(matrix(NA,nrow(dati),ncol(dati)))
                for(i in 1:ncol(dati)) fatt[,i]<-factor(dati[,i])
                return(fatt)
        }
        xf.new<-fattorizza(x)
        livelli.new<-aggregate(xf.new,list(as.factor(rep(1,nrow(x)))),nlevels)
        xf.old<-fattorizza(xcomp)
        livelli.old<-aggregate(xf.old,list(as.factor(rep(1,nrow(x)))),nlevels)
        livmiss<-sum(as.numeric(livelli.new[-1]-livelli.old[-1]))
```

```
}

stampa<-function(){
        cat("Percorso e nome del file:","\n")
        cat(name,"\n")
        cat("Caratteristiche del dataset iniziale:","\n")
        print(summary(xcomp))
        cat("Caratteristiche del dataset finale:","\n")
        print(summary(x))
        cat("N. osservazioni mancanti iniziale:",nummiss,"\n")
        cat("N. osservazioni mancanti richiesto:",soglia,"\n")
        cat("N. osservazioni mancanti effettivo:",sum(is.na(x)),"\n")
        cat("Quota osservazioni mancanti iniziale:",quotamiss,"\n")
        cat("Quota osservazioni mancanti richiesta:",pm,"\n")
        cat("Quota osservazioni mancanti effettiva:",sum(is.na(x))/(nrow(xm)*
            numvar),"\n")
        cat("Tolleranza:",toll,"\n")
        cat("Max n. oss. mancanti per riga:",max(rowSums(is.na(x))),"su",
            numvar-1,"possibili","\n")
        cat("Max n. oss. mancanti per colonna:",max(colSums(is.na(x))),"su",
            nrow(x)-1,"possibili","\n")
        if (optavar==1)         cat("N. totale modalità perdute:",livmiss,"\n"
            )

}

rs<-rowSums(is.na(x))
cs<-colSums(is.na(x))

windows()
istogrammi(list(DistrMissRow=rs,DistrMissCol=cs))

bringToTop(which=-1, stay=T)
stampa()

salva<-function(){
        namesplit<-strsplit(name,"\\/")
        nomefile<-namesplit[[1]][length(namesplit[[1]])]
        percorso<-sub(nomefile,"",name,fixed=T)
        percorso<-choose.dir(percorso, "Scegli una cartella per il file di
            output")
        nome<-winDialogString("Nome del file?",paste(nomefile," (",pm,"
            missing)",sep=""))
        write.table(x,file=paste(percorso,"\\",nome,".csv",sep=""),sep=";",
            col.names = NA)
        sink(file=paste(percorso,"\\",nome,".log",sep=""))
        stampa()
        sink()
        jpeg(file=paste(percorso,"\\",nome,".jpg",sep=""), width=1024, height
            =768, quality=100)
                istogrammi(list(DistrMissRow=rs,DistrMissCol=cs))
        dev.off()
}

switch(menu(c("Sì","No"),title="Salvare il dataset generato?"),salva(),cat("
    Nothing done\n"))
#switch(winDialog(type = "yesno", "Salvare il dataset generato?"),salva(),cat
    ("Nothing done\n"))
```

Bibliografia

[1] Abonyi J., Feil B. *Cluster Analysis for Data Mining and System Identification.* Birkhäuser Verlag, 2007;

[2] Akaike H. *A new look at the statistical model identification.* IEEE Transactions on Automatic Control, Vol.19, N.6, pagg. 716-723, 1974;

[3] Alpaydin E. *Soft vector quantization and the EM algorithm.* Neural Networks, N.11, pagg. 467-477, 1998;

[4] Alpaydin E. *Introduction to Machine Learning.* MIT Press, 2004;

[5] Asuncion A., Newman D.J. *UCI Machine Learning Repository.* School of Information and Computer Science, University of California, Irvine (CA), 2007. `http://www.ics.uci.edu/~mlearn/MLRepository.html`;

[6] Barcaroli G., D'Aurizio L., Luzi O., Manzari A., Pallara A. *Metodi e software per il controllo e la correzione dei dati.* Documenti Istat N.1, Istat, Roma, 1999;

[7] Benati S. *Categorical Data Fuzzy Clustering: An Analysis of Local Search Heuristics.* Dipartimento di Informatica e Studi Aziendali, Università di Trento, 2003;

[8] Bezdek J.C. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum Press, New York, 1981;

[9] Bezdek J.C., Pal N.R. *Two Soft Relatives of Learning Vetor Quantization.* Neural Networks, Vol.8, N.5, pagg. 729-743, 1995;

[10] Bottcher S.G., Dethlefsen C. *The deal Package.* The Comprehensive R Archive Network, 2008. `http://www.math.aau.dk/~dethlef/novo/deal`;

[11] Chambers R. *Evaluation criteria for statistical editing and imputation.* National Statistics Methodological Series N.28, UK National

Statistics, 2001. `http://www.statistics.gov.uk/methods_quality/publications.asp`;

[12] Charlton J. *Editing and imputation issues.* In "Towards Effective Statistical Editing and Imputation Strategies - Findings of the Euredit project", Vol.1. Euredit, 2003. `http://www.cs.york.ac.uk/euredit/`;

[13] Cheeseman P., Stutz J. *Bayesian Classification (AutoClass): Theory and Results.* Advances in Knowledge Discovery and Data Mining, pagg. 153-180, 1996;

[14] Cicchitelli G., Herzel A., Montanari G. E. *Il campionamento statistico.* Il Mulino, Bologna, 1992.

[15] Colledge M.J., Johnson J.H., Pare R., Sande I.G. *Large scale imputation of survey data.* Proceedings of the Section on Survey Research Methods, American Statistical Association, pagg. 431-436, 1978;

[16] Cooper G.F., Herskovits E. *A Bayesian approach for the induction of probabilistic models from data.* Machine Learning, 9, pagg. 309-347, 1992;

[17] Coppi R. *Il nuovo ruolo del metodo statistico nelle scienze sociali secondo l'approccio strategico integrato.* In "Strategie d'Analisi dei Dati nella Ricerca Sociale e Metodologia Integrata", Franco Angeli, Milano, 1995;

[18] Coppi R. *Lezioni di analisi statistica multivariata.* Dipartimento di Statistica, Probabilità e Statistiche Applicate. Università degli Studi di Roma "La Sapienza", Roma, 1998;

[19] Coppi R. *A theoretical framework for data Mining: the "Informational Paradigm".* Computational Statistics and Data Analysis, N.38, pagg. 501-515, 2002;

[20] Cowell, R.G., Dawid, A.P., Lauritzen, S.L., Spiegelhalter, D.J. *Probabilistic Networks and Expert Systems.* Springer, 2007;

[21] de Campos L.M. *A scoring function for learning Bayesian networks based on mutual information and conditional independence tests.* Journal of Machine Learning Research, N.7 pagg. 2149-2187, 2006;

[22] Dempster A.P., Laird N.M., Rubin D.B. *Maximum likelihood from incomplete data via the EM algorithm.* Journal of the Royal Statistical Society (B), Vol.39, N.1, pagg. 1-38, 1977;

[23] Di Zio M., Scanu M., Coppola L., Luzi O. Ponti A., *Bayesian networks for imputation.* Journal of the Royal Statistical Society (A), Vol.167, N.2, pagg. 309-322, 2004;

[24] Di Zio M., Sacco G., Scanu M., Vicard P. *Multivariate techniques for imputation based on bayesian networks.* Neural Networks World, Vol.10, N.4, pagg. 303-309, 2005;

[25] Di Zio M., Guarnera U., Luzi O. *Imputation through finite Gaussian mixture models.* Computational Statistics & Data Analysis Vol.51, N.11, pagg. 5305-5316, 2007;

[26] Di Zio M., Sacco G., Scanu M., Vicard P. *Metodologia e software per l'imputazione di dati mancanti tramite reti bayesisane.* In "Metodi statistici per l'integrazione di dati da fonti diverse", pagg. 313-328, Franco Angeli, Roma 2007;

[27] D'Orazio M., Di Zio M., Scanu M. *Statistical Matching: Theory and Practice.* Wiley and Sons, Chichester, 2006;

[28] Fellegi I. P., Sunter A. B. *A theory for record linkage.* Journal of the American Statistical Association, Vol. 64, N.328, pagg. 1183-1210, 1969;

[29] Fellegi I.P., Holt D. *A systematic approach to automatic Edit and imputation.* Journal of American Statistical Association, Vol.71, N.353, pagg. 17-35, 1976;

[30] Fisher R. A. *The use of multiple measurements in taxonomic problems.* Annals of Eugenics, N.7, Part II, pagg. 179-188, 1936;

[31] Friedman N. *Learning belief networks in the presence of missing values and hidden variables.* Fourteenth Inter. Conf. on Machine Learning (ICML97), 1997;

[32] Fortini M. *Linee guida metodologiche per rilevazioni statistiche.* Istat, Roma, 2000. `http://www.istat.it/strumenti/metodi/lineeguida/`;

[33] Gelman A., Hill J. *Data analysis using regression and multilevel/hierarchical models.* Cambridge University Press, 2006;

[34] Giudici P., Green P.J. *Decomposable graphical Gaussian model determination.* Biometrika N.86, pagg. 785-801, 1999;

[35] Goldberg D.E. *Genetic algorithms in search, optimization and machine learning.* Addison Wesley, 1989;

[36] Good I.J. *The estimation of probabilities.* MIT Press, 1968;

[37] Grande E., Luzi O. *Metodologia per l'imputazione delle mancate risposte parziali: analisi critica e soluzioni disponibili in Istat.* Contributi Istat N.6, Istat, Roma, 2003. `http://www.istat.it/dati/pubbsci/contributi/Contributi/contr_2003/2003_6.doc`;

[38] Groves R.M. *Survey errors and surveys costs.* Wiley and Sons, New York, 1989;

[39] Guarnera U. *Alcuni metodi di imputazione delle mancate risposte parziali per dati quantitativi. Il software QUIS.* Contributi Istat N.5, Istat, Roma, 2004. `http://www.istat.it/dati/pubbsci/contributi/Contributi/contr_2004/2004_5.doc`;

[40] Halkidi M., Batistakis Y., Vazirgiannis M. *On Cluster Validation Techniques.* Journal of Intelligent Information Systems, Vol.17, N.2-3, pagg. 107-145, 2001;

[41] Harding T., Tusell F. *The Cat Package.* The Comprehensive R Archive Network, 2004. `http://www.stat.psu.edu/~jls/misoftwa.html#aut`;

[42] Hathaway R.J., Y. Hu, Bezdek J.C. *Local Convergence of Tri-level Alternating Optimization.* Neural, Parallel, Scientific Computation, Vol.9, pagg. 19-28, 2001;

[43] Hathaway R.J., Bezdek J.C. *Fuzzy c-Means Clustering of Incomplete Data* .IEEE Transactions on Systems, Man and Cybernetics, Vol.31, N.5, pagg. 735-744, 2001;

[44] Heckerman D. *A tutorial on learning bayesian networks.* MSR-TR-95-06, Redmond 1995;

[45] Huang Z., Ng M. *A Fuzzy k-Modes Algorithm for Clustering Categorical Data.* IEEE Transactions on Fuzzy Systems, Vol.7, N.4, pagg. 446-452, 1999;

[46] Jain A., Murty N. *Data Clustering: A Review.* ACM Computing Surveys Vol.31, N.3, September 1999;

[47] Jensen F.V. *Bayesian Networks and Decision Graphs.* Springer-Verlag, New York, 2001;

[48] Kalton G., Kasprzyk D. *Imputing for missing survey responses.* Proceedings of the Section on Survey Research Methods, American Statistical Association, pagg. 22-31, 1982;

[49] Kalton, G., Kasprzyk, D. *The treatment of Missing Survey Data.* Survey Methodology, Vol.12, N.1, pagg. 1-16, 1986;

[50] Kim D. W., Lee K. H., Lee D. *Fuzzy Clustering of Categorical Data Using Fuzzy Centroids.* Pattern Recognition Letters Vol.25, N.11, pagg. 1263-1271, 2004;

[51] Kim D.W., Lee K.Y., Lee D., Lee K.H. *A k-Populations Algorithm for Clustering Categorical Data.* Pattern Recognition Vol.38, N.7, pagg. 1131-1134, 2005;

[52] Kullback S., Leibler R.A. *On information and sufficiency.* Annals of Mathematical Statistics, N.22, pagg. 79-86, 1951;

[53] Larrañaga P., Poza M., Yurramendi Y., Murga R.H., Kuijpers C.M.H. *Structure learning of Bayesian networks by genetic algorithms: a performance analysis of control parameters.* IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol.18, N.9, 1996;

[54] La Rocca A. *Fuzzy Clustering: la logica, i metodi.* Contributi Istat N.5, Istat, Roma, 2006. `http://www.istat.it/dati/pubbsci/contributi/Contributi/contr_2006/2006_5.pdf`;

[55] Lauritzen S.L., Spiegelhalter D.J. *Local computations with probabilities on graphical structures and their applications on expert systems.* Journal of the Royal Statistical Society (B), Vol.50, N.2, pagg. 157-224, 1988;

[56] Lauritzen S.L. *The EM algorithm for graphical association models with missing data.* Computational Statistics and Data Analysis N.19, pagg. 191-201, 1995;

[57] Lauritzen S.L. *Graphical Models.* Clarendon press, Oxford 1996;

[58] Lazzerini B. *Introduzione agli Insiemi Fuzzy e alla Logica Fuzzy.* Dipartimento di Ingegneria dell'Informazione, Università di Pisa, 2007. `http://info.iet.unipi.it/~lazzerini/icse/DispensaFuzzySpec.pdf`;

[59] Lebart L., Morineau A., Piron M. *Statistique exploratoire multidimensionelle.* Dunod, Paris, 1995;

[60] Little R.J.A., Rubin D.B. *Statistical analysis with missing data.* Wiley and Sons, New York, 1987;

[61] Luo J.W., Yang T., Wang Y. *Missing Value Estimation For Microarray Data Based On Fuzzy C-means Clustering.* HPCASIA '05: Proceedings of the Eighth International Conference on High-Performance Computing in Asia-Pacific Region, IEEE Computer Society, Washington DC, 2005;

[62] Malucelli F. *Appunti di introduzione alla Ricerca Operativa - Grafi e reti di flusso.* Dipartimento di Elettronica e Informazione, Politecnico di Milano, 2006. `http://home.dei.polimi.it/malucell/didattica/appunti/flussi2006.pdf`;

[63] MacQueen J. B. *Some Methods for classification and Analysis of Multivariate Observations.* Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability. University of California Press, Berkeley, Vol.1, pagg. 281-297, 1967;

[64] Madsen A.L., Lang M., Kjaerluff U.B., Jensen F. *The Hugin tool for learning bayesian networks.* In "Symbolic and quantitative approaches to reasoning with uncertainty, Proceedings of the 7th European Conference, ECSQARU 2003," a cura di Nielsen T.D., Zhang N.L., Aalborg, Danimarca, 2003. `http://www.hugin.com/`;

[65] Masselli M. *La qualità dei dati ed il sistema di controllo dell'indagine.* In "Manuale di tecniche di indagine", Vol.6, Istat, Roma, 1989;

[66] McLachlan G., Krishnan T. *The EM Algorithm and Extensions.* Wiley and Sons, New York, 1997;

[67] McLachlan G., Peel D. *Finite Mixture Models.* Wiley and Sons, New York, 2000;

[68] Michalski,R.S. *Learning by being told and learning from examples: an experimental comparison of the two methodes of knowledge acquisition in the context of developing an expert system for soybean desease diagnosis.* International Journal of Policy Analysis and Information Systems, Vol.4, N.2, pagg. 125-161, 1980;

[69] Monaci M. *Algoritmi euristici,* Dipartimento di Ingegneria dell'Informazione, Università di Padova. `http://www.dei.unipd.it/~monaci/euristici_rev21.pdf`;

[70] Myers J.W., Laskey K.B., DeJong K.A. *Learning Bayesian networks from incomplete data using evolutionary algorithms.* In "Proceedings of the Fourth Annual Conference on Genetic and Evolutionary Computation, pagg. 458-465. Morgan Kaufmann, Orlando, Florida., 1999.

[71] Pearl J. *Probabilistic reasoning in intelligent systems: networks of plausible inference.* Morgan Kaufmann, San Mateo (CA), 1988;

[72] Poli I., Roverato A. *A genetic algorithm for graphical model selection.* Journal of the Italian Statistical Society, Vol.2, pagg. 197–208, 1998;

[73] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2008. `http://www.R-project.org`;

[74] Ralambondrainy H. *A Conceptual Version of the K-Means Algorithm.* Pattern Recognition Letters Vol.16, N.11, pagg. 1147-1157, 1995;

[75] Rizzi A. *Analisi dei dati. Applicazione dell'informatica alla statistica.* La Nuova Italia Scientifica, Roma, 1985;

[76] Roverato A., Paterlini S. *Technological modelling for graphical models: an approach based on genetic algorithms.* Computational Statistics and Data Analysis, N.47, pagg. 323-337, 2004;

[77] Rubin D.B. *Inference and missing data.* Biometrika, Vol.63, N.3, pagg. 581-592, 1976;

[78] Ruspini E.H. *A New Approach to Clustering.* Information and Control, Vol.15, pagg. 22-32, 1969;

[79] Schafer J.L. *Analysis of incomplete multivariate data.* Chapman & Hall, Londra, 1997;

[80] Schwartz G. *Estimating the dimension of a model.* Annals of Statistics, N.6, pagg. 461-464, 1978;

[81] Scott A.J., Symons M.J. *Clustering methods bases on likelihood ratio criteria.* Biometrics N.27, pagg. 387-397, 1971;

[82] Spirtes P., Glymour K. *An algorithm for fast recovery of sparse causal graphs.* Social Scienze Computer Review, Vol.9, pagg. 62-72, 1991;

[83] Statistics Canada. *Quality Guidelines.* Third Edition, Ottawa, 1998. `http://www.statcan.ca/english/freepub/12-539-XIE/12-539-XIE.pdf`;

[84] Tarantola C., Vicard P. *Spanning trees and identifiability of a single-factor model.* Statistical Methods and Applications, Vol.11, N.2, pagg. 139-152, 2002;

[85] Tsekouras G. E., Papageorgiou D., Kotsiantis S., Kalloniatis C., Pintelas P. *Fuzzy Clustering of Categorical Attributes and its Use in Analyzing Cultural Data.* International Journal of Computational Intelligence. Vol.1, N.2, pagg. 147-151, 2004;

[86] Yang M.S. *On a class of fuzzy classification maximum likelihood procedures.* Fuzzy Sets and Systems, Vol.57, N.3, pagg. 365-375, 1993;

[87] Yang M.S., Chiang Y.H., Chen C.C., Lai C.Y. *A Fuzzy k-Partitions Model For Categorical Data And Its Comparision To The GoM Model.* Fuzzy Sets and Systems, Vol.159, N.4, pagg. 390-405, 2008;

[88] Willighagen E. *The genalg Package.* The Comprehensive R Archive Network, 2005. `http://cran.r-project.org/web/packages/genalg/index.html`;

[89] Woodbury M.A., Clive J. *Clinical Pure Types As A Fuzzy Partition*, Journal of Cybernetics, Vol.4, pagg. 111-121, 1974;

[90] Wong M.L., Leung K.S. *An Efficient Data Mining Method for Learning Bayesian Networks Using an Evolutionary Algorithm-Based Hybrid Approach.* IEEE Transactions on evolutionary computation, Vol.8, N.4, pagg. 378-404, 2004;

[91] Wong M.L., Guo Y.Y. *Learning Bayesian networks from incomplete databases using a novel evolutionary algorithm.* Decision Support Systems, N.45, pagg. 368–383, 2008;

[92] Zadeh L., *Fuzzy Sets.* Information and Control N.8, pagg. 338-353, 1965;

[93] Zhang N. L., Poole D. *A simple approach to Bayesian network computations.* In Proc. of the Tenth Canadian Conference on Artificial Intelligence, pagg. 171-178, 1994.

Elenco degli algoritmi|a

2.1 FCKM per dati incompleti 28
2.2 FKM e FKPOP per dati incompleti 32
2.3 GOM per dati incompleti 38
2.4 FKPAR per dati incompleti 42
3.1 Algoritmo di buona enumerazione 45
3.2 Greedy Search 51
3.3 Algoritmi genetici 53
3.4 Structural EM 54
4.1 QCE 60

Elenco delle figure

1.1 Esempio di strutture di mancata risposta 6

3.1 Esempio di DAG 44
3.2 Naive Bayes 46
3.3 Il markov blanket di X_5 56

4.1 Andamento del QCE al variare di m (dataset soybean disease) . 63
4.2 Andamento del QCE al variare della % di missing (dataset soybean disease, medie su 100 prove) 63
4.3 Andamento dell'indice D al variare di m (dataset soybean disease) 66
4.4 Andamento dell'indice D al variare della % di missing (dataset soybean disease, medie su 100 prove) 66
4.5 Andamento dell'indice Δ al variare di m (dataset soybean disease) 69
4.6 Andamento dell'indice Δ al variare della % di missing (dataset soybean disease, medie su 100 prove) 69
4.7 Andamento del QCE al variare di m (dataset zoo) 72
4.8 Andamento del QCE al variare della % di missing (dataset zoo, medie su 100 prove) 72
4.9 Andamento dell'indice D al variare di m (dataset zoo) 75
4.10 Andamento dell'indice D al variare della % di missing (dataset zoo, medie su 100 prove) 75
4.11 Andamento dell'indice Δ al variare di m (dataset zoo) 78
4.12 Andamento dell'indice Δ al variare della % di missing (dataset zoo, medie su 100 prove) 78
4.13 Confronto tra DAG 80
4.14 Il network Asia 82
4.15 Andamento dell'indice D al variare della % di missing (dataset Asia, medie su 100 prove) 86
4.16 Andamento dell'indice Δ al variare della % di missing (dataset Asia, medie su 100 prove) 88
4.17 Il network golf 89

4.18 Andamento dell'indice D al variare della % di missing (dataset golf, medie su 100 prove) 93
4.19 Andamento dell'indice Δ al variare della % di missing (dataset golf, medie su 100 prove) 94
4.20 Andamento dell'indice D al variare della % di missing (dataset CMC, medie su 100 prove) 98
4.21 Andamento dell'indice Δ al variare della % di missing (dataset CMC, medie su 100 prove) 101

A.1 Insieme classico e insieme fuzzy 104
A.2 Caratteristiche della funzione di appartenenza 105
A.3 Unione di insiemi fuzzy 106
A.4 Intersezione di insiemi fuzzy 106
A.5 Complemento di un insieme fuzzy 107
A.6 Inclusione di insiemi fuzzy 107
A.7 Non validità della legge del terzo escluso 108
A.8 Non validità della legge di non contraddizione 108

Elenco delle tabelle

1.1 Classificazione di alcuni metodi di imputazione 14

4.1 Quota Classificazioni Errate (dataset soybean disease, medie su 100 prove) 62
4.2 Quota Imputazioni Errate (dataset soybean disease, medie su 100 prove) 65
4.3 Indice di Consistenza (dataset soybean disease, medie su 100 prove) 68
4.4 Quota Classificazioni Errate (dataset zoo, medie su 100 prove) . 71
4.5 Quota Imputazioni Errate (dataset zoo, medie su 100 prove) . . 74
4.6 Indice di Consistenza (dataset zoo, medie su 100 prove) 77
4.7 Quota Adiacenze Invertite (dataset Asia, medie su 100 prove) . 83
4.8 Quota Adiacenze Aggiunte (dataset Asia, medie su 100 prove) . 83
4.9 Quota Adiacenze Cancellate (dataset Asia, medie su 100 prove) 84
4.10 Quota Adiacenze Errate (dataset Asia, medie su 100 prove) . . . 84
4.11 Divergenza di Kullback-Leibler (dataset Asia, medie su 100 prove) 85
4.12 Quota Imputazioni Errate (dataset Asia, medie su 100 prove) . 86
4.13 Indice di Consistenza (dataset Asia, medie su 100 prove) 87
4.14 Quota Adiacenze Invertite (dataset golf, medie su 100 prove) . 90
4.15 Quota Adiacenze Aggiunte (dataset golf, medie su 100 prove) . 90
4.16 Quota Adiacenze Cancellate (dataset golf, medie su 100 prove) 91
4.17 Quota Adiacenze Errate (dataset golf, medie su 100 prove) . . . 91
4.18 Divergenza di Kullback-Leibler (dataset golf, medie su 100 prove) 92
4.19 Quota Imputazioni Errate (dataset golf, medie su 100 prove) . . 93
4.20 Indice di Consistenza (dataset golf, medie su 100 prove) 94
4.21 Quota Imputazioni Errate (dataset CMC, medie su 100 prove, EM) 96
4.22 Quota Imputazioni Errate (dataset CMC, medie su 100 prove, fuzzy cluster analysis) 97
4.23 Quota Imputazioni Errate (dataset CMC, medie su 100 prove, reti bayesiane) 98

4.24 Indice di Consistenza (dataset CMC, medie su 100 prove, EM) 99
4.25 Indice di Consistenza (dataset CMC, medie su 100 prove, fuzzy cluster analysis) 100
4.26 Indice di Consistenza (dataset CMC, medie su 100 prove, reti bayesiane) 101