Università degli Studi di Padova
Dipartimento di Scienze Statistiche
Corso di Laurea Magistrale in
Scienze Statistiche

Modelli grafici indiretti e hidden Markov models. Implementazione e studio di fattibilità.

Relatore Prof. Alessandra Salvan
Dipartimento di Scienze Statistiche

Correlatore Prof. Maria Piera Rogantin
Dipartimento di Matematica Università di Genova

Laureando: Dario Brun
Matricola N 1014510

Anno Accademico 2012/2013

Indice

Introduzione

I grafi sono strutture matematiche discrete definite come un insieme di vertici collegati tra di loro da archi. Essi sono ampiamente studiati e utilizzati in vari ambiti in quanto forniscono rappresentazioni di lettura immediata dei fenomeni analizzati.

I modelli grafici sono modelli statistici multivariati che utilizzano i grafi per rappresentare le indipendenze condizionate tre le variabili coinvolte. Essi sono applicati ad una vasta gamma di aree scientifiche, tra cui la fisica statistica, la genetica, l'analisi delle tabelle di contingenza e la teoria delle decisioni.

In questo trattato i modelli grafici sono stati usati in un ambito scarsamente esplorato, l'individuazione delle frodi bancarie; la loro utilizzazione in tale campo è oggetto di uno studio di fattibilità inserito nelle attività del Centro di competenza su metodi in tempo reale di Fraud Detection ed Anti Riciclaggio dell'Università di Genova.

Sono stati selezionati due tipi di modelli grafici di indipendenza, ritenendoli i più idonei a modellare il problema oggetto di studio: i grafi indiretti e gli Hidden Markov Models. I primi quattro capitoli sono dedicati allo studio di questi due modelli attraverso un'ampia ricerca bibliografica e nell'ultimo è riportata l'applicazione alle frodi bancarie.

Alla fine di ciascun capitolo sono presentati i riferimenti bibliografici specifici e sono indicate le parti originali della tesi.

Nel capitolo 1 viene esposta la teoria generale sui grafi in ambito statistico. Vengono presentate le proprietà di Markov pairwise, locale, globale e di fattorizzazione, che caratterizzano la distribuzione di probabilità delle variabili aleatorie associate ai vertici del grafo, in modo che le indipendenze condizionate tra le variabili possano corrispondere dalla presenza-assenza di archi tra i vertici del grafo. Inoltre, vengono date condizioni di equivalenza tra le quattro proprietà. In generale, l'equivalenza tra due proprietà markoviane diverse dipende dalle distribuzioni di probabilità coinvolte; tuttavia esistono proprietà geometriche del grafo che comportano l'uguaglianza tra le diverse proprietà, indipendentemente dalle distribuzioni di probabilità associate.

Il capitolo 2 tratta gli Hidden Markov Models (HMM), processi di Markov in cui gli stati non sono direttamente osservabili, ma nascosti. Vengono date tre definizioni equivalenti e vengono presentati i principali problemi statistici legati

agli HMM: calcolo della verosimiglianza, determinazione della sequenza di stati nascosti più probabile, stima in massima verosimiglianza del modello. Vengono discussi due algoritmi per i primi due problemi. La soluzione al terzo problema non è banale e viene presentato un algoritmo iterativo, ideato da Baum nel 1970 e applicato nel caso più generale di processi markoviani, in grado di trovare un massimo locale per la verosimiglianza. Dopo aver dimostrato la convergenza dell'algoritmo viene indicata la sua applicazione agli HMM. Infine, vengono mostrati alcuni algoritmi per evitare gli underflow in presenza di un numero di osservazioni elevato.

I capitoli 3 e 4 trattano i modelli grafici indiretti per l'analisi di tabelle di contingenza. L'indipendenza condizionata tra due variabili discrete equivale ad assumere che le distribuzioni di probabilità delle celle siano contenute in un sottospazio vettoriale dello spazio prodotto cartesiano delle dimensioni di tutti i criteri di classificazione. Vengono introdotti i modelli gerarchici e i modelli grafici e viene spiegato come in un modello grafico tutte le indipendenze condizionate possano essere lette osservando gli archi mancanti nel suo grafo di interazione. Viene poi dimostrato come in un modello gerarchico esteso lo stimatore di massima verosimiglianza delle probabilità delle singole celle esista sempre unico. Viene inoltre presentato un algoritmo iterativo, detto algoritmo IPS, per calcolarlo. Infine, viene spiegato come, in caso di grafi decomponibili, lo stimatore di massima verosimiglianza possa essere calcolato in modo diretto.

Nel capitolo 5 i metodi statistici precedenti vengono utilizzati per lo studio delle frodi bancarie. I dati rappresentano operazioni bancarie effettuate online e l'obiettivo è predire quali costituiscano frodi, in modo che la banca possa identificarle e bloccarle. Dopo un'analisi preliminare dove vengono selezionate le variabili di interesse, vengono utilizzati come classificatori un grafo indiretto e un HMM. Per la stima viene utilizzato un software sviluppato appositamente in linguaggio C++. Analizzando i risultati ottenuti, si osserva come entrambi i metodi, nonostante classifichino con un errore basso, comportino problemi. L'Hidden Markov Model infatti presenta spesso un errore di secondo tipo troppo alto e il grafo può classificare le osservazioni solo nel caso in cui gli stati osservati nell'insieme di stima e in quello di verifica siano gli stessi. Per questo motivo viene proposto un classificatore misto che combina i vantaggi dei due classificatori. I risultati ottenuti con questo classificatore sembrano promettenti e possono essere la base per studi futuri.

Capitolo 1

I Grafi

1.1 Definizioni e notazioni

Un grafo è un insieme di vertici uniti fra loro da archi, i quali possono avere un orientamento (in questo si caso si parlerà di grafi orientati) oppure no. Più precisamente, diamo la seguente definizione.

Definizione 1.1 *Un grafo è una coppia*

$$\mathbb{G} = (V, E)$$

dove V è l'insieme dei vertici e E è l'insieme degli archi, cioè un sottoinsieme del prodotto cartesiano $V \times V$.

A ciascun vertice $\alpha \in V$ del grafo è associata una variabile aleatoria X_α, che può essere discreta o continua. Un grafo è detto puro se ai suoi vertici sono associate variabili aleatorie di un unico tipo. In questo testo considereremo solo grafi puri con variabili discrete.

Se α e β sono due elementi di V, l'arco che li unisce è *indiretto* se:

$$(\alpha, \beta) \in V \times V \wedge (\beta, \alpha) \in V \times V \ .$$

L'arco che li unisce è invece *diretto* orientato da α verso β se:

$$(\alpha, \beta) \in V \times V \wedge (\beta, \alpha) \notin V \times V \ .$$

Se α e β sono uniti da un arco indiretto, che indichiamo con $\{\alpha, \beta\}$, si scrive:

$$\alpha \sim \beta \ .$$

Se invece essi sono uniti da un arco diretto orientato da α verso β, che indichiamo con (α, β), si scrive:

$$\alpha \to \beta \ .$$

Nel caso di grafi indiretti, se $\mathbb{P}_V^2$ è il sottoinsieme dell'insieme delle parti di V che contiene gli insiemi di 2 elementi, cioè:

$$\mathbb{P}_V^2 = \{A \subseteq V : \quad \mid A \mid = 2\}$$

allora vale che $E \subseteq \mathbb{P}_V^2$.

In questo testo tratteremo solamente grafi indiretti.

Se due vertici α e β sono uniti da un arco sono detti *adiacenti*. Altrimenti sono detti *non adiacenti*. L'insieme di tutti i vertici adiacenti ad un vertice α è detto $ne(\alpha)$ (dall'inglese neighbours).

Definizione 1.2 *Dato un insieme $A \subseteq V$ si definisce* sottografo *indotto da A e si indica con $\mathbb{G}_A$ il grafo (A, E_A) dove $E_A = E \cap \mathbb{P}_A^2$.*

Un grafo è *completo* se tutti i vertici sono uniti da un arco. Un sottoinsieme $A \subset V$ è completo se induce un sottografo che è completo.

Definizione 1.3 *Un sottoinsieme di V completo e massimale rispetto all'inclusione è detto* clique*, ossia un insieme $C \subseteq V$ è una clique se e solo se C è completo e $\nexists\, A \subseteq V$ tale che A è completo e $C \subset A$.*

Definizione 1.4 *Dato Un grafo $\mathbb{G} = (V, E)$, si definisce* grafo duale *di $\mathbb{G}$, indicato con $\mathbb{G}^\sim$, il grafo:*

$$\mathbb{G}^\sim = (V, \mathbb{P}_V^2 \setminus E)$$

cioè per ogni α e β in V, $\alpha \sim^\sim \beta$ se e solo se $\alpha \not\sim \beta$.

1.2 Decomposizione di grafi

Sia A un sottoinsieme di vertici, $A \subseteq V$.

Definizione 1.5 *Si definisce* frontiera *di A, indicata con ∂A, l'insieme dei vertici in $V \setminus A$ che sono adiacenti ai vertici di A, ossia:*

$$\partial A = \{\beta \in V \setminus A : \exists \alpha \in A, \beta \sim \alpha\} \ .$$

Definizione 1.6 *Si definisce* chiusura *di A o $\overline{A}$ l'unione di A e della sua frontiera:*

$$\overline{A} = A \cup \partial A \ .$$

Dati due vertici α e β si definisce *percorso di lunghezza n* da α a β una sequenza di vertici $\alpha_0, \ldots, \alpha_n$ con $\alpha_i \sim \alpha_{i+1} \forall i$, $\alpha_0 = \alpha$ e $\alpha_n = \beta$.

Se esiste un percorso di lunghezza n da α a β di dice che essi sono *connessi* e si scrive:

$$\alpha \rightleftharpoons \beta \ .$$

È immediato dimostrare che essere connessi è una relazione di equivalenza le cui classi di equivalenza, che indichiamo con $[\alpha]$,

$$\beta \in [\alpha] \Leftrightarrow \alpha \rightleftharpoons \beta$$

sono le componenti connesse del grafo $\mathbb{G}$.

Si definisce *n-ciclo* un percorso di lunghezza n dove $\alpha = \beta$, ovvero un percorso circolare, che inizia e finisce nello stesso punto.

Consideriamo un grafo $\mathbb{G} = (V, E)$.

Definizione 1.7 *Dati due vertici α e β in V, un sottoinsieme $C \subset V$ è detto* (α, β)-separatore *se tutti i percorsi da α a β intersecano C.*

Denotiamo con $[\alpha]_A$ la classe di equivalenza del vertice α nel sottografo indotto dall'insieme A.

Vale la seguente proprietà.

Proposizione 1.1 *Un sottoinsieme C di V è un (α, β)-separatore se e solo se vale:*

$$[\alpha]_{V \setminus C} \neq [\beta]_{V \setminus C} \ .$$

Dimostrazione: Se C è un (α, β)-separatore tutti i percorsi da α a β passano per almeno un elemento di C. Quindi nel grafo indotto da $V \setminus C$ essi stanno in due componenti connesse diverse e, quindi, hanno classi di equivalenza distinte.

Se, invece, nel grafo indotto da $V \setminus C$ α e β hanno classi di equivalenza distinte essi stanno in due componenti connesse distinte. Pertanto ciascun percorso da α a β contiene almeno un elemento di C e C è un (α, β)-separatore. □

Dati tre sottoinsiemi di V, A B e C, si dice che C *separa* A da B se per ogni $\alpha \in A$ e per ogni $\beta \in B$ C è un (α, β) separatore.

Siano (A, B, C) tre sottoinsiemi disgiunti di V tali che $V = A \cup B \cup C$.

Definizione 1.8 *La terna di vertici (A, B, C) è detta* decomposizione debole *di $\mathbb{G}$ se C è completo e C separa A da B.*

Definizione 1.9 *Un grafo $\mathbb{G}$ è detto* debolmente decomponibile *se è completo o se esiste una decomposizione debole (A, B, C) che conduce a due sottografi debolmente decomponibili $\mathbb{G}_{A \cup C}$ e $\mathbb{G}_{B \cup C}$.*

Un grafo è detto *triangolato* se ciascun n-ciclo con $n \geq 4$ possiede una *corda*, cioè due vertici non consecutivi nella sequenza del ciclo che sono adiacenti nel grafo. Si dimostra facilmente che ciascun sottografo di un grafo triangolato è triangolato.

Proposizione 1.2 *Sia $\mathbb{G}$ un grafo allora sono fatti equivalenti:*

- *(i) $\mathbb{G}$ è decomponibile debolmente.*
- *(ii) $\mathbb{G}$ è triangolato.*
- *(iii) $\forall \alpha, \beta \in V$ ciascun (α, β)-separatore minimale è completo.*

Dimostrazione: Dimostriamo per induzione sulla cardinalità di V.

Sia $\mid V \mid \leq 3$. Allora le tre condizioni risultano automaticamente vere. Ciascun (α, β)-separatore è costituito al massimo da un singolo vertice e quindi è completo. Non esistono n-cicli con $n \geq 4$ e quindi il grafo è triangolato. Infine, studiando separatamente i pochi casi possibili si arriva ad affermare che è debolmente decomponibile.

Supponiamo che le tre proposizioni siano vere per $\mid V \mid \leq n$. Sia $\mid V \mid = n+1$.

- i)$\Rightarrow$ ii)
  Sia $\mathbb{G}$ debolmente decomponibile. Se $\mathbb{G}$ è completo allora è triangolato, poiché tutti i vertici sono adiacenti. Sia $\mathbb{G}$ non completo.

  Allora esistono tre sottoinsiemi disgiunti di V, (A, B, C), tali che i sottografi $\mathbb{G}_{A \cup C}$ e $\mathbb{G}_{B \cup C}$ sono debolmente decomponibili. Poiché $\mid A \cup C \mid \leq n$ e $\mid B \cup C \mid \leq n$ si può applicare l'ipotesi induttiva e i due sottografi risultano quindi essere triangolati.

  Quindi tutti gli n-cicli con $n \geq 4$ che non intersecano contemporaneamente A e B hanno almeno una corda. Consideriamo un n-ciclo che contiene un elemento di A e uno di B. Poiché C separa A da B esso deve contenere almeno due elementi di C non consecutivi. Ma allora esso contiene una corda essendo C completo. Quindi $\mathbb{G}$ è triangolato.

- ii) $\Rightarrow$ iii)
  Siano α e β due vertici e sia C un (α, β)-separatore minimale. Se C possiede un solo elemento allora è completo.

  Supponiamo che $\mid C \mid > 1$. Siano γ e δ due elementi distinti qualsiasi di C. Poiché C è minimale esiste un percorso da α a β passando per γ e uno da β a α passando per δ. Si crea quindi il seguente n-ciclo:

  $$(\alpha, \ldots, \gamma, \ldots, \beta, \ldots, \delta, \ldots, \alpha) \ .$$

  Questo produce un n-ciclo con $n \geq 4$. Poiché $\mathbb{G}$ è triangolato, esso deve avere almeno una corda. Data l'arbitrarietà degli altri vertici deve essere per forza $\gamma \sim \delta$. Quindi C è completo.

- iii)⇒ i)
  Siano α e β due vertici. Supponiamo che ogni (α, β)-separatore minimale sia completo. Se $\mathbb{G}$ è completo allora è debolmente decomponibile.

  Sia $\mathbb{G}$ non completo. Siano α e β due vertici non adiacenti e sia C un (α, β)-separatore minimale. L'insieme dei vertici V può quindi essere partizionato in:

  $$V = [\alpha]_{V\setminus C} \cup [\beta]_{V\setminus C} \cup C \cup D$$

  dove D è l'insieme dei vertici restanti e i quattro insiemi così definiti sono disgiunti.

  Definiamo quindi:

  $$A = [\alpha]_{V\setminus C} \cup D \quad B = [\beta]_{V\setminus C} \quad C = C \ .$$

  Siano γ e δ due vertici in $\mathbb{G}_{A\cup C}$ e sia C_1 un (γ, δ)-separatore minimale in $\mathbb{G}_{A\cup C}$. Allora C_1 è contenuto in un (γ, δ)-separatore in $\mathbb{G}$. Esso è completo per ipotesi e quindi anche C_1 risulta completo. Ne segue che l'ipotesi *iii*) vale anche per i sottografi $\mathbb{G}_{A\cup C}$ e $\mathbb{G}_{B\cup C}$. Poiché $\mid A\cup C \mid \leq n$ e $\mid B \cup C \mid \leq n$, si può applicare ai due sottografi l'ipotesi induttiva, ottenendo che essi sono debolmente decomponibili.

  Abbiamo trovato una decomposizione debole (A, B, C) di $\mathbb{G}$ nei due sottografi debolmente decomponibili $\mathbb{G}_{A\cup C}$ e $\mathbb{G}_{B\cup C}$. Quindi $\mathbb{G}$ è debolmente decomponibile.

□

Proposizione 1.3 *Sia $\mathbb{G} = (V, E)$ un grafo triangolato con $\mid V \mid \geq 2$. Allora esistono almeno due vertici la cui frontiera è completa. Se $\mathbb{G}$ non è completo tali vertici possono essere scelti non adiacenti.*

Dimostrazione: Dimostriamo la tesi per induzione sulla cardinalità di V.

Se $\mid V \mid = 2$ allora la dimostrazione è ovvia.

Supponiamo che la tesi valga per $\mid V \mid \leq n$ e sia $\mid V \mid = n + 1$. Se $\mathbb{G}$ è completo la tesi è ovvia.

Sia $\mathbb{G}$ non completo. Allora esiste una decomposizione debole (A, B, C) di $\mathbb{G}$. Per ipotesi induttiva esistono due vertici non adiacenti in $\mathbb{G}_{A\cup C}$ con frontiera completa. Dato che C è completo, almeno uno dei due vertici deve appartenere ad A. Chiamiamo questo vertice α.

Allo stesso modo, applicando l'ipotesi induttiva sul sottografo $\mathbb{G}_{B\cup C}$, esiste un vertice $\beta \in B$ la cui frontiera risulta completa. Dato che C separa A da B, α e β risultano essere una coppia di vertici non adiacenti la cui frontiera è completa nel grafo $\mathbb{G}$. □

Definizione 1.10 *Si definisce $\mathbb{G}_{\square}$ l'insieme di tutti i grafi che non possiedono un sottografo del tipo $(\{\alpha,\beta,\gamma,\delta\},\{\{\alpha,\beta\},\{\gamma,\delta\}\})$, cioè con quattro vertici e due archi non adiacenti.*

Proposizione 1.4 *Dato un grafo $\mathbb{G}$, $\mathbb{G}\in\mathbb{G}_{\square}$ se e solo se, dato il suo duale $\mathbb{G}^{\sim}$, ogni 4-ciclo in $\mathbb{G}^{\sim}$ possiede una corda.*

Dimostrazione: Sia $\mathbb{G}\in\mathbb{G}_{\square}$. Supponiamo che $\mathbb{G}^{\sim}$ possieda un 4-ciclo che non ha nessuna corda. Sia questo ciclo $(\alpha,\beta,\gamma,\delta)$. Allora l'insieme $A=\{\alpha,\beta,\gamma,\delta\}$ dei quattro vertici del 4-ciclo induce su $\mathbb{G}^{\sim}$ il seguente sottografo:

$$\mathbb{F}^{\sim}=(\{\alpha,\beta,\gamma,\delta\},\{\{\alpha,\beta\},\{\beta,\gamma\},\{\gamma,\delta\},\{\delta,\alpha\}\})\ .$$

Di conseguenza A induce su $\mathbb{G}$ il sottografo

$$\mathbb{F}=(\{\alpha,\beta,\gamma,\delta\},\{\{\alpha,\gamma\},\{\beta,\delta\}\})\ .$$

ma ciò non può sussistere in quanto $\mathbb{G}\in\mathbb{G}_{\square}$. Necessariamente tutti i 4-cicli in $\mathbb{G}^{\sim}$ hanno una corda.

Supponiamo adesso che ogni 4-ciclo in $\mathbb{G}^{\sim}$ possieda una corda. Supponiamo che $\mathbb{G}\notin\mathbb{G}_{\square}$. Allora esiste un insieme di quattro vertici $A=\{\alpha,\beta,\gamma,\delta\}$ che induce su $\mathbb{G}$ un sottografo del tipo:

$$\mathbb{F}=(\{\alpha,\beta,\gamma,\delta\},\{\{\alpha,\gamma\},\{\beta,\delta\}\})\ .$$

Di conseguenza esiste il seguente sottografo in $\mathbb{G}^{\sim}$:

$$\mathbb{F}^{\sim}=(\{\alpha,\beta,\gamma,\delta\},\{\{\alpha,\beta\},\{\beta,\gamma\},\{\gamma,\delta\},\{\delta,\alpha\}\})$$

ma ciò non può sussistere essendo un 4-ciclo senza nessuna corda. Necessariamente $\mathbb{G}\in\mathbb{G}_{\square}$. □

Definizione 1.11 *Si definisce $\mathbb{G}_{<\triangle}$ l'insieme di tutti i grafi che non possiedono sottografi che hanno tre vertici e al massimo un arco, cioè sottografi del tipo $(\{\alpha,\beta,\gamma\},\{\{\alpha,\beta\}\})$ o del tipo $(\{\alpha,\beta,\gamma\},\emptyset)$.*

Se un grafo $\mathbb{G}$ appartiene a $\mathbb{G}_{<\triangle}$ allora il suo grafo duale $\mathbb{G}^{\sim}$ non può possedere sottografi con tre vertici con un numero di archi ≥ 2.

1.3 Sequenze perfette

Consideriamo un grafo $\mathbb{G}=(V,E)$. Siano $B_1,B_2,\ldots,B_k$ una sequenza di sottoinsiemi di V tali che $\forall j\in\{1,\ldots,k\}$:

$$H_j=\cup_{i=1}^{j}B_i,\quad R_j=B_j\setminus H_{j-1},\quad S_j=H_{j-1}\cap B_j\ .$$

Definizione 1.12 *La sequenza $B_1, \ldots, B_k$ è detta* sequenza debolmente perfetta *se verifica le seguenti proprietà:*

1. $\forall j > 1 \quad \exists i < j \quad : \quad S_j \subseteq B_i$.
2. $\forall j \quad S_j$ *è completo.*

I termini H_j rappresentano la storia, gli R_j i residui e gli S_j i separatori della sequenza.

Definizione 1.13 *Sia $\mathbb{G} = (V, E)$ un grafo. Un* ordinamento debolmente perfetto *sui vertici di $\mathbb{G}$ è un ordinamento $\alpha_1, \ldots, \alpha_k$ tale che:*

$$B_j = \{\overline{\alpha}_j\} \cap \{\alpha_1, \ldots, \alpha_j\} \quad \forall j \geq 1$$

è una sequenza debolmente perfetta.

Vale la seguente proposizione.

Proposizione 1.5 *Sia $\mathbb{G} = (V, E)$ un grafo e sia $B_1, \ldots, B_k$ una sequenza debolmente perfetta che contenga tutte le cliques di $\mathbb{G}$. Allora, per ogni j, S_j separa $H_{j-1} \setminus S_j$ da R_j nel sottografo $\mathbb{G}_{H_j}$ e quindi $(H_{j-1} \setminus S_j, R_j, S_j)$ sono una decomposizione debole di $\mathbb{G}_{H_j}$.*

Dimostrazione: Dimostriamo la tesi per induzione su j.

Sia p il valore più alto per cui B_p è una clique. Allora si ha che:

$$H_p = V, \quad R_i = \emptyset \quad \forall i > p \ .$$

Sia $\alpha \in R_p$ e sia $\beta \in B_i \setminus S_p$ per un qualche $i < p$. Supponiamo che esista l'arco $\{\alpha, \beta\} \in E$.

Necessariamente l'insieme $\{\alpha, \beta\}$ deve essere contenuto in una qualche clique del grafo, cioè:

$$\exists C \in \mathbb{C} : \quad \{\alpha, \beta\} \subset C \ .$$

Ma ciò è assurdo, perché

$$\beta \notin B_p, \quad \alpha \notin H_{p-1}$$

e, dato che tutte le cliques del grafo sono contenute nella sequenza, non vi possono essere archi esterni ad essa. Quindi S_p separa $H_{p-1} \setminus S_p$ da R_p.

Supponiamo che la tesi sia vera per $k + 1 \leq j \leq p$ e consideriamo $j = k$. Allora $B_1, \ldots, B_k$ contiene tutte le cliques del sottografo $\mathbb{G}_{H_k}$.

Sia $\alpha \in R_k$ e sia $\beta \in B_i \setminus S_k$ per un certo $i < k$. Supponiamo che esista l'arco $\{\alpha, \beta\} \in E$.

Necessariamente l'insieme $\{\alpha, \beta\}$ deve essere contenuto in una qualche clique del grafo, cioè:

$$\exists C \in \mathbb{C} : \quad \{\alpha, \beta\} \subset C \ .$$

Ma abbiamo che

$$\beta \notin B_k, \quad \alpha \notin H_{k-1} \ .$$

Dato che la sequenza $B_1, \ldots, B_k$ contiene tutte le cliques del grafo $\mathbb{G}_{H_k}$, non può esistere in esso un arco che colleghi α e β. Quindi S_k separa $H_{k-1} \setminus S_k$ da R_k in $\mathbb{G}_{H_k}$. □

Supponiamo che $C_1, \ldots, C_p$ siano le cliques di un grafo $\mathbb{G}$ e che esse formino una sequenza perfetta. Se si numerano i vertici del grafo, selezionando prima quelli contenuti in C_1, poi quelli in R_2, e così via fino a quelli contenuti in R_p, si ottiene un ordinamento debolmente perfetto. Tuttavia il viceversa non è valido.

Vale la seguente proposizione:

Proposizione 1.6 *Sia $\mathbb{G} = (V, E)$ un grafo e sia $C_1, \ldots, C_k$ una sequenza debolmente perfetta. Sia C_t un elemento della sequenza e sia p il minimo valore per cui vale:*

$$C_t \subseteq C_p \ .$$

Allora valgono i seguenti fatti:

1. *se $p < t$ allora: $C_1, \ldots, C_{t-1}, C_{t+1}, \ldots, C_k$ è una sequenza debolmente perfetta.*

2. *se $t > p$ allora: $C_1, \ldots, C_{t-1}, C_p, C_{t+1}, \ldots, C_{p-1}, C_{p+1}, \ldots, C_k$ è una sequenza debolmente perfetta.*

Dimostrazione: 1. Consideriamo la sequenza $C_1, \ldots, C_{t-1}, C_{t+1}, \ldots, C_k$. Chiaramente $C_1, \ldots, C_{t-1}$ è una sequenza debolmente perfetta.

Sia $j > t-1$. Allora S_j è completo per ipotesi. Inoltre esiste un $i < j$ nella sequenza originaria tale che $S_j \subseteq B_i$.

Se $i \neq t$, allora B_i è contenuto anche nella sequenza modificata. Se invece $i = t$, dato che $B_t \subseteq B_p$, si ha che $S_j \subseteq B_p$.

Quindi $C_1, \ldots, C_{t-1}, C_{t+1}, \ldots, C_k$ è una sequenza debolmente perfetta.

2. Consideriamo la sequenza $C_1, \ldots, C_{t-1}, C_p, C_{t+1}, \ldots, C_{p-1}, C_{p+1}, \ldots, C_k$.

Per prima cosa osserviamo che, nella sequenza originaria, si ha:

$$S_p = C_p \cap (C_1 \cup \cdots \cup C_{p-1}) \supseteq C_p \cap C_t = C_t \ .$$

Sappiamo inoltre che esiste un $k < p$ tale che

$$S_p \subseteq C_k \ .$$

Dato che p è il valore minimo per cui $C_t \subseteq C_p$ si ottiene subito che $k = t$ e quindi:

$$S_p = C_t \ .$$

Chiamiamo S_p^* l'insieme S relativo alla nuova sequenza. Si ha che:

$$S_p^* = C_p \cap H_{t-1} \subseteq S_p = C_t \ .$$

Tuttavia, dato che $C_p \supseteq C_t$, si ha anche:

$$S_p^* = C_p \cap H_{t-1} \supseteq C_t \cap H_{t-1} = S_t \ .$$

Dato che $S_p^* \subseteq C_t$ si ha:

$$S_p^* = S_p^* \cap C_t = (C_p \cap H_{t-1}) \cap C_t = S_t \ .$$

Quindi S_p^* è completo ed è contenuto in C_k per un qualche $k < t$.

Infine, sia $j > p - 1$. Allora S_j^* è completo per ipotesi ed è immediato dimostrare che esiste un $k < j$ tale che $S_j \subseteq B_k$.

□

Dato un ordinamento debolmente perfetto dei vertici di un grafo, è possibile da esso costruire una sequenza perfetta che contenga tutte le clique di esso, come afferma la seguente proposizione.

Proposizione 1.7 *Sia $\mathbb{G} = (V, E)$ un grafo e sia $\alpha_1, \ldots, \alpha_k$ un ordinamento debolmente perfetto dei vertici di V. Allora gli insiemi così definiti:*

$$B_j = \overline{\{\alpha_j\}} \cap \{\alpha_1, \ldots, \alpha_j\}$$

costituiscono una sequenza debolmente perfetta che contiene tutte le cliques di $\mathbb{G}$.

Dimostrazione: La sequenza B_j è debolmente perfetta per la definizione di ordinamento debolmente perfetto.

Dimostriamo ora che ogni clique del grafo è contenuta nella sequenza.

Sia $j = k$. Essendo α_k l'ultimo vertice dell'ordinamento, dato che per definizione ogni $B_j \ni \alpha_j$ per ogni j, deve essere: $R_k = \{\alpha_k\}$.

Quindi $B_k = S_k \cup \{\alpha_k\}$. Dato che $B_k = \overline{\{\alpha_k\}}$ e S_k è completo, anche B_k risulta completo e massimale rispetto all'inclusione. Quindi B_k è una clique del grafo.

Consideriamo ora un generico B_j con $j < k$. Nel sottografo $\mathbb{G}_{V\setminus\{\alpha_{j+1},\ldots,\alpha_k\}}$ è possibile ripetere la dimostrazione precedente e B_j risulta essere una clique. Supponiamo che B_j non sia una clique nel grafo $\mathbb{G}$. Allora esiste un vertice $\beta \in V \setminus B_j$ tale che $\beta \sim \delta$ per ogni $\delta \in B_j$.

Per definizione di B, deve necessariamente essere che:

$$\beta \in \{\alpha_{j+1}, \ldots, \alpha_k\} \ .$$

Ma poiché B_β è completo e $B_\beta \ni \beta$, ne segue:

$$B_j \subseteq B_\beta \ .$$

Quindi per ogni j o B_j è una clique o è contenuto in un altro B_h, con $h > j$. Poiché per ogni elemento $\alpha \in V$ esiste un elemento della sequenza B_α tale che $B_\alpha \ni \alpha$, per ciascun elemento $\alpha \in V$ la clique del grafo che contiene tale vertice deve essere contenuta nella sequenza. □

Proposizione 1.8 *Sia $\mathbb{G} = (V, E)$ un grafo. I seguenti fatti sono equivalenti:*

1. *i vertici di $\mathbb{G}$ ammettono un ordinamento debolmente perfetto.*
2. *le cliques di $\mathbb{G}$ possono essere numerate a formare una sequenza debolmente perfetta.*
3. *il grafo $\mathbb{G}$ è debolmente decomponibile.*

Dimostrazione: • 1)⇒2)

Sia $\alpha_1, \ldots, \alpha_k$ un ordinamento debolmente perfetto. Consideriamo la seguente sequenza di insiemi:

$$B_j = \overline{\{\alpha_j\}} \cap \{\alpha_1, \ldots, \alpha_j\} \ .$$

Allora per la proposizione 1.7 $B_1 \ldots, B_k$ costituisce una sequenza debolmente perfetta di insiemi che contiene tutte le clique del grafo $\mathbb{G}$.

Sia j un generico indice tale che $1 \leq j \leq k$ e supponiamo che B_j non sia una clique del grafo; allora esiste un indice p con $p > j$ tale che $B_j \subseteq B_p$.

Consideriamo la seguente sequenza:

$$C_1, \ldots, C_{j-1}, C_p, C_{j+1}, \ldots, C_{p-1}, C_{p+1}, \ldots, C_k \ .$$

Come è stato dimostrato nella proposizione 1.6, essa è una sequenza perfetta.

Ripetendo l'operazione per ciascun indice i tale che B_i non sia una clique del grafo si arriva, con un numero finito di passaggi, a determinare una sequenza perfetta che contiene tutte e sole le cliques di $\mathbb{G}$.

- 2)⇒3)
  Sia $C_1, \ldots, C_k$ la sequenza debolmente perfetta formata dalle cliques di $\mathbb{G}$. Allora ovviamente $H_k = V$ e, come è stato dimostrato nella proposizione 1.5, $(H_{k-1} \setminus S_k, R_k, S_k)$ sono una decomposizione debole di $\mathbb{G}$. Inoltre $R_k \cup S_k \subseteq C_k$ e quindi è completo; $H_{k-1} \cup S_k = H_{k-1} = C_1 \cup \cdots \cup C_{k-1}$ costituisce una sequenza debolmente perfetta delle cliques del grafo $\mathbb{G}_{V \setminus R_k}$ e quindi possiede una decomposizione debole. Quindi $\mathbb{G}$ è debolmente decomponibile.

- 3)⇒1) Dimostriamo la tesi per induzione sulla cardinalità di V.

  Supponiamo che $\mid V \mid = 2$. Allora un qualunque ordinamento sui due vertici risulta essere debolmente perfetto.

  Supponiamo che la tesi sia vera per $\mid V \mid \leq n$ e assumiamo $\mid V \mid = n + 1$. Poiché il grafo è debolmente decomponibile, come abbiamo dimostrato nella proposizione 1.3, esiste un vertice, che chiamiamo α_k, la cui frontiera risulta essere completa. Consideriamo il sottografo $\mathbb{G}_{V \setminus \{\alpha_k\}}$. Su di esso è possibile applicare l'ipotesi induttiva e pertanto esiste un ordinamento debolmente perfetto dei vertici $\alpha_1, \ldots, \alpha_{k-1}$.

  Costruiamo l'insieme B_k. Esso è uguale a:

  $$B_k = \overline{\{\alpha_k\}} \cap \{\alpha_1, \ldots, \alpha_k\} = \overline{\{\alpha_k\}} \ .$$

  Pertanto B_k è completo. Quindi dato che:

  $$S_k = H_{k-1} \cap B_k \subseteq B_k$$

  anche S_k risulta completo. Per concludere la dimostrazione occorre quindi trovare un indice i tale che risulti $S_k \subseteq B_i$. Sia m il massimo indice per cui vale $\alpha_m \in S_k$. Ovviamente vale anche $\alpha_m \in B_m$.

  Poiché S_k è completo, per ogni $\beta \in S_k$ vale $\alpha_m \sim \beta$ e quindi:

  $$\beta \in \overline{\{\alpha_m\}} \ .$$

  Dato che l'indice di β è minore di m vale anche:

  $$\beta \in \{\alpha_1, \ldots, \alpha_m\} \ .$$

  E quindi:

  $$\beta \in B_m = \overline{\{\alpha_m\}} \cap \{\alpha_1, \ldots, \alpha_m\} \ .$$

  Quindi $S_k \subseteq B_m$ e la sequenza $\alpha_1, \ldots, \alpha_k$ è un ordinamento debolmente perfetto dei vertici di $\mathbb{G}$.

  □

1.4 Indipendenza condizionata

Siano X, Y e Z tre variabili aleatorie discrete. Si dice che X è indipendente da Y data Z se vale:

$$\mathbb{P}[X = x, Y = y \mid Z = z] = \mathbb{P}[X = x \mid Z = z]\,\mathbb{P}[Y = y \mid Z = z]$$

per ogni valore x y e z. Se le tre variabili sono assolutamente continue la condizione diviene:

$$f_{XY|Z}(x, y \mid z) = f_{X|Z}(x \mid z)\, f_{Y|Z}(y \mid z) \ .$$

Consideriamo tre variabili aleatorie discrete X, Y e Z. Per semplificare la notazione indichiamo con $p(.)$ la loro legge di probabilità.

Proposizione 1.9 *Valgono le seguenti proprietà.*

$$a) \quad X \perp\!\!\!\perp Y \mid Z \Leftrightarrow p(x, y, z) = \frac{p(x, z)p(y, z)}{p(z)}$$

$$b) \quad X \perp\!\!\!\perp Y \mid Z \Leftrightarrow p(x \mid y, z) = p(x \mid z)$$

$$c) \quad X \perp\!\!\!\perp Y \mid Z \Leftrightarrow p(x, z \mid y) = p(x \mid z)p(z \mid y)$$

$$d) \quad X \perp\!\!\!\perp Y \mid Z \Leftrightarrow \exists h, k \textit{ funzioni tali che } p(x, y, z) = h(x, z)k(y, z)$$

$$e) \quad X \perp\!\!\!\perp Y \mid Z \Leftrightarrow p(x, y, z) = p(x \mid z)p(y, z)$$

Vediamo ora alcune proprietà fondamentali dell'indipendenza condizionata.

Proposizione 1.10 *Siano X, Y e Z tre variabili aleatorie. Allora valgono le seguenti proprietà:*

$$i) \quad X \perp\!\!\!\perp Y \mid Z \Rightarrow Y \perp\!\!\!\perp X \mid Z$$

$$ii) \quad X \perp\!\!\!\perp Y \mid Z, \quad U = h(X) \Rightarrow U \perp\!\!\!\perp Y \mid Z$$

$$iii) \quad X \perp\!\!\!\perp Y \mid Z, \quad U = h(X) \Rightarrow X \perp\!\!\!\perp Y \mid (Z, U)$$

$$iv) \quad X \perp\!\!\!\perp Y \mid Z, \quad X \perp\!\!\!\perp W \mid (Y, Z) \Rightarrow X \perp\!\!\!\perp (W, Y) \mid Z$$

Esiste una quinta proprietà dell'indipendenza condizionata che però vale solo sotto ipotesi aggiuntive su X, Y e Z:

$$v) \quad X \perp\!\!\!\perp Y \mid Z, \quad X \perp\!\!\!\perp Z \mid Y \Rightarrow X \perp\!\!\!\perp (Y, Z) \ .$$

Una condizione sufficiente per la proprietà v) è la seguente.

Proposizione 1.11 *Se la distribuzione congiunta di X, Y e Z è positiva e continua, oppure se la distribuzione congiunta di X, Y e Z è positiva e X, Y e Z sono discrete, la proprietà v) risulta vera.*

Dimostrazione: Assumiamo che X Y e Z abbiano una densità congiunta positiva, $f(x, y, z) > 0$, e che $X \perp\!\!\!\perp Y \mid Z$ e $X \perp\!\!\!\perp Z \mid Y$. Allora si ha:

$$f(x, y, z) = f_{X,Y|Z}(x, y \mid z) f_Z(z) = f_{X|Z}(x \mid z) f_{Y|Z}(y \mid z) f_Z(z) = k(x, z) l(y, z)$$

indicando con:

$$k(x, z) = f_{X|Z}(x \mid z), \quad l(y, z) = f_{Y|Z}(y \mid z) f_Z(z) \ .$$

Allo stesso modo, sfruttando la seconda relazione di indipendenza, si ottiene:

$$f(x, y, z) = g(x, y) h(y, z) \ .$$

Poiché X, Y e Z sono discrete oppure hanno una densità continua si ha che per ogni valore di z:

$$g(x, y) = \frac{k(x, z) l(y, z)}{h(y, z)} \ .$$

Poiché questa relazione è vera per ogni valore di z, è possibile fissare $z = z_0$ ed otteniamo:

$$g(x, y) = a(x) b(y)$$

dove

$$a(x) = k(x, z_0), \quad b(y) = \frac{l(y, z_0)}{h(y, z_0)} \ .$$

Quindi si ha che:

$$f(x, y, z) = a(x) b(y) h(y, z)$$

e quindi, per la proprietà d) dell'indipendenza condizionata, $X \perp\!\!\!\perp (Y, Z)$. □

1.5 Separazione di grafi

È possibile dare una definizione di indipendenza condizionata anche sui vertici di un grafo.

Consideriamo un grafo $\mathbb{G} = (V, E)$. Siano A, B e C tre sottoinsiemi di V.

Definizione 1.14 *Si dice che A* è indipendente da B condizionato a C *se C separa A da B:*

$$A \perp\!\!\!\perp^{\mathbb{G}} B \mid C \Leftrightarrow C \quad \textit{separa} \quad A \quad \textit{da} \quad B \ .$$

Le proprietà di indipendenza condizionata delle variabili aleatorie associate ai vertici del grafo possono essere tradotte in proprietà geometriche del grafo stesso. È facile vedere che $\perp\!\!\!\perp^{\mathbb{G}}$ soddisfa le stesse quattro proprietà dell'indipendenza condizionata delle variabili aleatorie, come è indicato nella proposizione seguente.

Proposizione 1.12 *Consideriamo un grafo $\mathbb{G} = (V, E)$. Siano A, B e C tre sottoinsiemi di V. Allora valgono i seguenti fatti:*

$$i) \quad A \perp\!\!\!\perp^{\mathbb{G}} B \mid C \Rightarrow B \perp\!\!\!\perp^{\mathbb{G}} A \mid C$$

$$ii) \quad A \perp\!\!\!\perp^{\mathbb{G}} B \mid C, \quad U \subset A \Rightarrow U \perp\!\!\!\perp^{\mathbb{G}} B \mid C$$

$$iii) \quad A \perp\!\!\!\perp^{\mathbb{G}} B \mid C, \quad U \subset B \Rightarrow A \perp\!\!\!\perp^{\mathbb{G}} B \mid (C \cup U)$$

$$iv) \quad A \perp\!\!\!\perp^{\mathbb{G}} B \mid C, \quad A \perp\!\!\!\perp^{\mathbb{G}} D \mid (B \cup C) \Rightarrow A \perp\!\!\!\perp^{\mathbb{G}} (B \cup D) \mid C \ .$$

Dimostrazione: La dimostrazione delle prime tre proprietà è molto banale ed è lasciata quindi al lettore. Dimostriamo invece la proprietà quattro.

Sia $\alpha \in A$ e $\beta \in B \cup D$. Allora o $\beta \in B$ oppure $\beta \in D$.

Se $\beta \in B$, dato che C separa A da B, C è un (α, β)-separatore.

Se invece $\beta \in D$, dato che $B \cup C$ separa A da D, ogni percorso da α a β interseca $B \cup C$. Ma allora ogni percorso da α a β interseca C poiché C separa A da B. Quindi C è un (α, β)-separatore.

Per l'arbitrarietà di α e β, si ha che C separa A da $B \cup D$ e quindi

$$A \perp\!\!\!\perp^{\mathbb{G}} B \cup D \mid C \ .$$

□

Come nel caso dell'indipendenza delle variabili aleatorie, esiste una quinta proprietà, che però non ha valore universale. Essa tuttavia vale se gli insiemi A, B e C sono disgiunti.

Proposizione 1.13 *Consideriamo un grafo $\mathbb{G} = (V, E)$. Siano A, B e C tre sottoinsiemi disgiunti di V. Allora vale la proprietà*

$$v) \quad A \perp\!\!\!\perp^{\mathbb{G}} B \mid C, \quad A \perp\!\!\!\perp^{\mathbb{G}} C \mid B \Rightarrow A \perp\!\!\!\perp^{\mathbb{G}} (B \cup C) \ .$$

Dimostrazione: Sia α un elemento di A e sia β un elemento di B. Dato che C separa A da B, ogni percorso da α a β deve intersecare C almeno in un punto, che chiamiamo γ. Ma allora esiste un percorso da α a γ che non passa per B e ciò è assurdo perché B separa A da C.

Necessariamente non esistono percorsi da elementi di A a elementi di B.

Allo stesso modo non esistono percorsi da elementi di A a elementi di C.

Quindi A e $B \cup C$ stanno in componenti connesse disgiunte e vale:

$$A \perp\!\!\!\perp^{\mathbb{G}} B \cup C \ .$$

□

Poiché ad ogni vertice del grafo è associata una variabile aleatoria, è opportuno selezionare quelle distribuzioni di probabilità sulle variabili associate la cui indipendenza condizionata corrisponda all'indipendenza condizionata dei vertici del grafo appena definita. Di questo ci occuperemo nel prossimo paragrafo.

1.6 Proprietà di Markov

Consideriamo un grafo $\mathbb{G} = (V, E)$. A ciascun vertice $\alpha \in V$ è associata una variabile aleatoria discreta X_α. Ciascuna variabile aleatoria X_α è definita su uno spazio $\mathcal{X}_\alpha$ che può essere finito oppure numerabile. La legge congiunta di tutte le variabili X è definita sullo spazio $\mathcal{X}$ definito come:

$$\mathcal{X} = \times_{\alpha \in V} \mathcal{X}_\alpha \ .$$

Per comodità di notazione, nel seguito scriveremo semplicemente α, invece di X_α, per riferirci alla variabile aleatoria corrispondente. Inoltre, dato un sottoinsieme A di V, indicheremo con $\mathbb{P}[A]$ la legge congiunta delle variabili X_α tali che $\alpha \in A$.

Definizione 1.15 *Proprietà di Markov.*

- *Una probabilità* $\mathbb{P}$ *su* V *gode della* proprietà di Markov pairwise *se:*

$$\forall \alpha\beta \in V, \alpha \not\sim \beta \quad \alpha \perp\!\!\!\perp \beta \mid V \setminus \{\alpha, \beta\} \ .$$

- *Una probabilità* $\mathbb{P}$ *su* V *gode della* proprietà di Markov locale *se:*

$$\forall \alpha \in V \quad \alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid \partial\alpha \ .$$

- *Una probabilità* $\mathbb{P}$ *su* V *gode della* proprietà di Markov globale *se* $\forall A, B, C \subset V$ *disgiunti tali che* C *separa* A *da* B *si ha:*

$$A \perp\!\!\!\perp B \mid C \ .$$

Proposizione 1.14 *Per ciascun grafo* $\mathbb{G}$ *e ciascuna probabilità* $\mathbb{P}$ *definita sullo spazio delle variabili asssociate ai suoi vertici vale che:*

$$\mathbb{P} \textit{ globale} \Rightarrow \mathbb{P} \textit{ locale} \Rightarrow \mathbb{P} \textit{ pairwise} \ .$$

Dimostrazione: • globale⇒ locale

Supponiamo che $\mathbb{P}$ goda della proprietà di Markov globale. Allora, per ogni $\alpha \in V$, $\partial\alpha$ separa α da $V \setminus \overline{\alpha}$, inoltre i tre insiemi sono disgiunti. Quindi vale automaticamente la proprietà locale.

- locale⇒ pairwise
  Supponiamo ora che $\mathbb{P}$ goda della proprietà di Markov locale. Siano α e β due vertici non adiacenti. Allora necessariamente $\beta \in V \setminus \overline{\alpha}$. Applicando la proprietà locale si ha:

  $$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid \partial\alpha \ .$$

  Dato che:

  $$V \setminus \{\alpha, \beta\} = \partial\alpha \cup ((V \setminus \overline{\alpha}) \setminus \{\beta\})$$

  e

  $$((V \setminus \overline{\alpha}) \setminus \{\beta\}) \subset V \setminus \overline{\alpha}$$

  applicando la proprietà iii) dell'indipendenza condizionata si ottiene:

  $$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid V \setminus \{\alpha, \beta\} \ .$$

  Quindi, applicando la proprietà ii) si ottiene:

  $$\alpha \perp\!\!\!\perp \beta \mid V \setminus \{\alpha, \beta\} \ .$$

  e pertanto $\mathbb{P}$ gode della proprietà di Markov pairwise.

□

In generale però le tre proprietà sono distinte. Non è possibile dimostrare che Markov pairwise ⇒ Markov globale. Per poterne affermare l'uguaglianza sono necessarie ipotesi aggiuntive.

Definizione 1.16 *Dato un grafo $\mathbb{G} = (V, E)$, definiamo l'insieme $R(V)$ come l'insieme di tutte le coppie $((\alpha, \beta), K)$, dove (α, β) è una coppia di vertici di V e $K \subset V \setminus \{\alpha, \beta\}$.*

Consideriamo ora i tre seguenti sottoinsiemi di $\mathrm{R}(V)$:

$$\mathrm{K}_g(\mathbb{G}) = \{((\alpha, \beta), K) \in \mathrm{R}(V) : \quad \text{K separa } \alpha \text{ da } \beta\}$$

$$\mathrm{K}_l(\mathbb{G}) = \{((\alpha, \beta), K) \in \mathrm{R}(V) : \quad \partial\alpha \subset K \text{ oppure } \partial\beta \subset K\}$$

$$\mathrm{K}_p(\mathbb{G}) = \left\{((\alpha, \beta), V \setminus \{\alpha, \beta\}) \in \mathrm{R}(V) : \quad \{\alpha, \beta\} \in \mathbb{P}_V^2 \setminus E\right\} \ .$$

Evidentemente valgono le seguenti relazioni:

$$\mathrm{K}_p(\mathbb{G}) \subseteq \mathrm{K}_l(\mathbb{G}) \subseteq \mathrm{K}_g(\mathbb{G}) \ .$$

Definizione 1.17 *Fissata una legge di probabilià $\mathbb{P}$ su V, definiamo l'insieme $R^{\mathbb{P}}(V) \subset R(V)$ come:*

$$R^{\mathbb{P}}(V) = \{((\alpha, \beta), K) \in R(V) : \quad \alpha \perp\!\!\!\perp \beta \mid K\} \ .$$

Proposizione 1.15 *Valgono allora le seguenti equivalenze:*

- *i) Una distribuzione di probabilità $\mathbb{P}$ su un grafo $\mathbb{G} = (V, E)$ gode della proprietà di Markov pairwise $\Leftrightarrow$ $K_p(\mathbb{G}) \subset R^{\mathbb{P}}(V)$.*
- *ii) Una distribuzione di probabilità $\mathbb{P}$ su un grafo $\mathbb{G} = (V, E)$ gode della proprietà di Markov locale $\Leftrightarrow$ $K_l(\mathbb{G}) \subset R^{\mathbb{P}}(V)$.*
- *iii) Una distribuzione di probabilità $\mathbb{P}$ su un grafo $\mathbb{G} = (V, E)$ gode della proprietà di Markov globale $\Leftrightarrow$ $K_g(\mathbb{G}) \subset R^{\mathbb{P}}(V)$.*

Dimostrazione:

- i) La dimostrazione è ovvia in quanto le definizioni della proprietà di Markov pairwise e dell'insieme $\mathrm{K}_p(\mathbb{G})$ sono equivalenti.
- ii)$\Leftarrow$
  Supponiamo che $\mathrm{K}_l(\mathbb{G}) \subset \mathrm{R}^{\mathbb{P}}(V)$. Sia $\alpha \in V$. Dimostriamo la proprietà di Markov locale per induzione sulla cardinalità di $V \setminus \overline{\alpha}$.

  Sia $\mid V \setminus \overline{\alpha} \mid = 1$. Allora $V \setminus \overline{\alpha} = \{\beta\}$. Scegliendo in $\mathrm{K}_l(\mathbb{G})$ la coppia $((\alpha, \beta), K)$, dove l'insieme $K = \partial\alpha$, si ha che $\alpha \perp\!\!\!\perp \beta \mid \partial\alpha$ e quindi $\mathbb{P}$ gode della proprietà di Markov locale.

  Supponiamo che la tesi sia vera per $\mid V \setminus \overline{\alpha} \mid = n$. Sia $\mid V \setminus \overline{\alpha} \mid = n + 1$. Allora $V \setminus \overline{\alpha} = \{\beta_1, \ldots, \beta_{n+1}\}$.

  Consideriamo i primi n elementi $\{\beta_1, \ldots, \beta_n\}$. Per l'ipotesi induttiva, restringendosi al sottografo generato da $V \setminus \{\beta_{n+1}\}$, si ha che $\alpha \perp\!\!\!\perp \{\beta_1, \ldots, \beta_n\} \mid \partial\alpha$. Scegliamo in $\mathrm{K}_l(\mathbb{G})$ la seguente coppia:

  $$((\alpha, \beta_{n+1}), K)$$

  dove

  $$K = \partial\alpha \cup \{\beta_1, \ldots, \beta_n\} \ .$$

  Si ha che:

  $$\alpha \perp\!\!\!\perp \beta_{n+1} \mid \partial\alpha \cup \{\beta_1, \ldots, \beta_n\} \ .$$

  Applicando la proprietà iv) dell'indipendenza condizionata si ottiene quindi che:

  $$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid \partial\alpha$$

  cioè $\mathbb{P}$ gode della proprietà di Markov locale.

  $\Rightarrow$
  Supponiamo che $\mathbb{P}$ goda della proprietà di Markov locale, cioè per ogni nodo α:

  $$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid \partial\alpha \ .$$

Sia H un qualunque insieme tale che $H \subset V \setminus \overline{\alpha}$. Allora applicando la proprietà iii) dell'indipendenza condizionata si ha che:

$$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid H \cup \partial\alpha \ .$$

Sia $K = \partial\alpha \cup H$ allora:

$$\alpha \perp\!\!\!\perp V \setminus \overline{\alpha} \mid K \ .$$

Infine, applicando la proprietà ii) dell'indipendenza condizionata, scegliendo un vertice $\beta \in V \setminus \overline{\alpha}$ non appartenente a K si ha:

$$\alpha \perp\!\!\!\perp \beta \mid K \ .$$

Data l'arbitrarietà di α, β e K si ha $\mathrm{K}_l(\mathbb{G}) \subset \mathrm{R}^{\mathbb{P}}(V)$.

- iii)$\Rightarrow$
  Supponiamo che $\mathbb{P}$ goda della proprietà di Markov globale. Allora per ogni terna di sottoinsiemi di V (A, B, K) disgiunti tali che K separa A da B si ha che
  $$A \perp\!\!\!\perp B \mid K \ .$$
  Applicando la proprietà ii) dell'indipendenza condizionata si ha che per ogni $\alpha \in A$ e per ogni $\beta \in B$
  $$\alpha \perp\!\!\!\perp \beta \mid K$$
  e quindi $\mathrm{K}_g(\mathbb{G}) \subset \mathrm{R}^{\mathbb{P}}(V)$.

  $\Leftarrow$
  Supponiamo che $\mathrm{K}_g(\mathbb{G}) \subset \mathrm{R}^{\mathbb{P}}(V)$. Per prima cosa dimostriamo che per ogni vertice $\alpha \in V$, per ogni insieme $B \subset V$ e per ogni insieme $K \subset V$ disgiunti tale che K separa $\{\alpha\}$ da B si ha:
  $$\alpha \perp\!\!\!\perp B \mid K \ .$$
  Applichiamo l'induzione sulla cardinalità di B. Se $\mid B \mid = 1$ allora la tesi è vera per definizione dell'insieme $\mathrm{K}_g(\mathbb{G})$.

  Supponiamo che la tesi sia vera per $\mid B \mid = n$ e consideriamo $\mid B \mid = n+1$. Allora $B = \{\beta_1, \ldots, \beta_{n+1}\}$. Se l'insieme K separa A da B allora K separa A da qualunque sottoinsieme di B. Quindi applicando l'ipotesi induttiva sappiamo che
  $$\alpha \perp\!\!\!\perp \{\beta_1, \ldots, \beta_n\} \mid K \ .$$

Inoltre K separa A da $\{\beta_{n+1}\}$, ma allora anche $K \cup \{\beta_1, \ldots, \beta_n\}$ separa A da $\{\beta_{n+1}\}$. Quindi:

$$\alpha \perp\!\!\!\perp \{\beta_{n+1}\} \mid K \cup \{\beta_1, \ldots, \beta_n\} \ .$$

Applicando la proprietà iv) dell'indipendenza condizionata abbiamo quindi la tesi.

Ripetendo allo stesso modo la dimostrazione per induzione anche sulla cardinalità dell'insieme A (esercizio per il lettore) otteniamo che per ogni terna di insiemi (A, B, K) disgiunti contenuti in V tali che K separa A da B

$$A \perp\!\!\!\perp B \mid K$$

e cioè la distribuzione di probabilità $\mathbb{P}$ gode della proprietà di Markov globale.

□

1.7 Equivalenza fra le proprietà di Markov

È ora possibile definire in quali casi le tre proprietà di Markov risultano equivalenti.

Proposizione 1.16 *Dato un grafo $\mathbb{G} = (V, E)$, la proprietà di Markov globale e la proprietà di Markov locale coincidono per ogni probabilità $\mathbb{P}$ se e solo se $\mathbb{G} \in \mathbb{G}_\square$.*

Dimostrazione: Sia $\mathbb{G} \in \mathbb{G}_\square$. Siano α e β due vertici tali che l'arco $\{\alpha, \beta\} \notin E$. Sia $\gamma \in \partial\alpha \setminus \partial\beta$ e sia $\delta \in \partial\beta \setminus \partial\alpha$. Necessariamente l'arco $\{\gamma, \delta\} \in E$ altrimenti si genererebbe un sottografo di quattro vertici con due archi non adiacenti. Sia K un generico insieme che separa α da β. La coppia $((\alpha, \beta), K) \in \mathrm{K}_g(\mathbb{G})$. Dato che separa α da β, $K \supset (\partial\alpha \cap \partial\beta)$. Inoltre o $\partial\alpha \setminus K = \emptyset$ oppure $\partial\beta \setminus K = \emptyset$.

Infatti, supponiamo che esistano $\gamma \in \partial\alpha \setminus K$ e $\delta \in \partial\beta \setminus K$. Allora l'arco $\{\gamma, \delta\}$ appartiene a E. Ma allora possiamo costruire il percorso $(\alpha, \gamma, \delta, \beta)$ da α a β non passante per K e ciò è assurdo perché K separa α da β.

Pertanto K contiene almeno uno fra $\partial\alpha$ e $\partial\beta$ e quindi la coppia $((\alpha, \beta), K) \in \mathrm{K}_l(\mathbb{G})$.

Quindi gli insiemi $\mathrm{K}_l(\mathbb{G})$ e $\mathrm{K}_g(\mathbb{G})$ sono uguali e le proprietà markoviane sono equivalenti per definizione.

Sia $\mathbb{G} \notin \mathbb{G}_\square$. Allora esistono quattro vertici distinti α, β, γ e $\delta \in V$ tali che gli archi $\{\alpha, \beta\}$ e $\{\gamma, \delta\}$ non appartengono a E, $\gamma \in \partial\alpha \setminus \partial\beta$ e $\delta \in \partial\beta \setminus \partial\alpha$. Inoltre l'insieme $V \setminus \{\alpha, \beta, \gamma, \delta\}$ separa α da β, quindi la coppia:

$$((\alpha, \beta), V \setminus \{\alpha, \beta, \gamma, \delta\}) \in \mathrm{K}_g(\mathbb{G}) \ .$$

Tuttavia, definendo $K = V \setminus \{\alpha, \beta, \gamma, \delta\}$, $\gamma \in \partial\alpha \setminus K$ e $\delta \in \partial\beta \setminus K$, quindi $K \not\supset \partial\alpha$ e $K \not\supset \partial\beta$. Quindi la coppia:

$$((\alpha, \beta), V \setminus \{\alpha, \beta, \gamma, \delta\}) \notin \mathrm{K}_l(\mathbb{G}) \ .$$

Quindi i due insiemi sono distinti, in particolare vale l'inclusione:

$$\mathrm{K}_l(\mathbb{G}) \subset \mathrm{K}_g(\mathbb{G}) \ .$$

Sia $X \sim \text{Bernoulli}(\frac{1}{2})$ e sia Z una costante. Consideriamo la seguente probabilità $\mathbb{P}$ sul grafo: $\alpha = \beta = \gamma = \delta = X$, $\varepsilon = Z \quad \forall \varepsilon \in V \setminus \{\alpha, \beta, \gamma, \delta\}$. Allora la relazione $\alpha \perp\!\!\!\perp \beta \mid V \setminus \{\alpha, \beta, \gamma, \delta\}$ è ovviamente non valida. Pertando:

$$\mathrm{K}_g(\mathbb{G}) \not\subset \mathrm{R}^{\mathbb{P}}(V)$$

quindi non vale la proprietà di Markov globale.

Se $\varepsilon \in V \setminus \{\alpha, \beta, \gamma, \delta\}$ allora:

$$\varepsilon \perp\!\!\!\perp \ V \setminus \overline{\varepsilon} \mid \partial\varepsilon \ .$$

Per $\varepsilon \in \{\alpha, \beta, \gamma, \delta\}$, questa relazione si riduce a:

$$X \perp\!\!\!\perp X \mid X$$

la quale è banalmente vera. Quindi

$$\mathrm{K}_l(\mathbb{G}) \subset \mathrm{R}^{\mathbb{P}}(V)$$

e vale la proprietà di Markov locale.

Quindi le due proprietà risultano distinte. □

Proposizione 1.17 *Dato un grafo $\mathbb{G} = (V, E)$, la proprietà di Markov locale e la proprietà di Markov pairwise coincidono per ogni probabilità $\mathbb{P}$ se e solo se $\mathbb{G} \in \mathbb{G}_{<\triangle}$.*

Dimostrazione: Sia $\mathbb{G} \in \mathbb{G}_{<\triangle}$ e siano α e β due vertici tali che l'arco $\{\alpha, \beta\} \notin E$. Allora deve essere che $\partial\alpha = \partial\beta = V \setminus \{\alpha, \beta\}$.

Supponiamo infatti che esista $\gamma \in (V \setminus \{\alpha, \beta\}) \setminus \partial\alpha$. Allora il sottografo generato da $\{\alpha, \beta, \gamma\}$ avrebbe un unico arco e ciò è assurdo.

Quindi se la coppia $((\alpha, \beta), K) \in \mathrm{K}_l(\mathbb{G})$ necessariamente $K = V \setminus \{\alpha, \beta\}$ e quindi $((\alpha, \beta), K) \in \mathrm{K}_p(\mathbb{G})$.

Quindi i due insiemi sono uguali

$$\mathrm{K}_l(\mathbb{G}) = \mathrm{K}_p(\mathbb{G})$$

e le due proprietà markoviane sono equivalenti per definizione.

Viceversa, supponiamo che $\mathbb{G} \notin \mathbb{G}_{<\triangle}$. Siano α β e γ tre vertici distinti tali che gli archi $\{\alpha, \beta\}$ e $\{\alpha, \gamma\}$ non appartengono a E. Ma allora $V \setminus \{\alpha, \beta, \gamma\} \supset \partial\alpha$ e quindi la coppia

$$((\alpha, \beta), V \setminus \{\alpha, \beta, \gamma\}) \subset \mathrm{K}_l(\mathbb{G}) \setminus \mathrm{K}_p(\mathbb{G}) \ .$$

Quindi i due insiemi sono distinti e in particolare vale

$$\mathrm{K}_p(\mathbb{G}) \subset \mathrm{K}_l(\mathbb{G}) \ .$$

Sia $X \sim$ Bernoulli$(\frac{1}{2})$ e sia Z una costante. Consideriamo la seguente probabilità $\mathbb{P}$ sul grafo: $\alpha = \beta = \gamma = X$, $\varepsilon = Z \quad \forall \varepsilon \in V \setminus \{\alpha, \beta, \gamma\}$.

Allora la relazione $\alpha \perp\!\!\!\perp \beta \mid V \setminus \{\alpha, \beta, \gamma\}$ è ovviamente non valida. Pertando:

$$\mathrm{K}_l(\mathbb{G}) \not\subset \mathrm{R}^{\mathbb{P}}(V)$$

quindi non vale la proprietà di Markov locale.

Si dimostra in modo molto semplice (esercizio per il lettore) che invece $\mathbb{P}$ gode della proprietà di Markov pairwise.

Quindi le due proprietà di Markov risultano distinte. □

Esiste inoltre una condizione sufficiente per l'uguaglianza di tutte e tre le proprietà Markoviane.

Proposizione 1.18 *Sia $\mathbb{G} = (V, E)$ un grafo. Se per ogni A, B, C e D sottoinsiemi disgiunti di V vale le seguente proprietà:*

$$A \perp\!\!\!\perp B \mid (C \cup D)\,, \quad A \perp\!\!\!\perp C \mid (B \cup D) \Rightarrow A \perp\!\!\!\perp (B \cup C) \mid D$$

allora le proprietà di Markov globale, locale e pairwise sono equivalenti.

Dimostrazione: Sia $\mathbb{P}$ una distribuzione di probabilità su $\mathcal{X}$, spazio delle variabili associate a $\mathbb{G}$. È sufficiente dimostrare che:

$$\mathbb{P} \text{ pairwise} \Rightarrow \mathbb{P} \text{ globale} \ .$$

Supponiamo quindi che $\mathbb{P}$ goda della proprietà di Markov pairwise. Siano A, B e C tre sottoinsiemi di V non vuoti disgiunti tali che C separi A da B. Dimostriamo la tesi per induzione sulla cardinalità di C.

Se $\mid C \mid = \mid V \mid -2$ allora necessariamente esistono due vertici α e β tali che $A = \{\alpha\}$, $B = \{\beta\}$, $C = V \setminus \{\alpha, \beta\}$ e vale:

$$\alpha \perp\!\!\!\perp \beta \mid V \setminus \{\alpha, \beta\} \ .$$

Supponiamo che la tesi sia vera per $\mid V \mid -2 \geq \mid C \mid \geq n+1$. Sia $\mid C \mid = n$.

Supponiamo che $V = A \cup B \cup C$. Necessariamente almeno uno tra gli insiemi A e B possiede almeno due elementi. Sia questo insieme A. Se $\alpha \in A$, $C \cup \{\alpha\}$ separa $A \setminus \{\alpha\}$ da B e inoltre $C \cup A \setminus \{\alpha\}$ separa $\{\alpha\}$ da B. Applicando l'ipotesi induttiva si ha:

$$A \setminus \{\alpha\} \perp\!\!\!\perp B \mid C \cup \{\alpha\} \text{ e } \{\alpha\} \perp\!\!\!\perp B \mid C \cup A \setminus \{\alpha\} .$$

Applicando la condizione iniziale si ottiene subito che:

$$A \perp\!\!\!\perp B \mid C .$$

Supponiamo invece che $V \supset A \cup B \cup C$. Sia $\alpha \in V \setminus (A \cup B \cup C)$. Allora $C \cup \{\alpha\}$ separa A da B, quindi per l'ipotesi induttiva:

$$A \perp\!\!\!\perp B \mid C \cup \{\alpha\} .$$

Inoltre, $\partial\alpha \subset A \cup C$ oppure $\partial\alpha \subset B \cup C$. Se infatti esistessero $\gamma \in \partial\alpha \cap A$ e $\delta \in \partial\alpha \cap B$ si creerebbe un percorso da A a B che non passa per C e ciò è assurdo.

Quindi $A \cup C$ separa B da $\{\alpha\}$ oppure $B \cup C$ separa A da $\{\alpha\}$. Sia ad esempio che $A \cup C$ separa B da $\{\alpha\}$, allora per l'ipotesi induttiva:

$$B \perp\!\!\!\perp \{\alpha\} \mid A \cup C .$$

Applicando la condizione iniziale si ottiene che

$$A \cup \{\alpha\} \perp\!\!\!\perp B \mid C .$$

Applicando la proprietà ii) dell'indipendenza condizionata si ottiene quindi:

$$A \perp\!\!\!\perp B \mid C .$$

□

1.8 La proprietà di fattorizzazione

Oltre alle proprietà pairwise, locale e globale esiste una quarta proprietà markoviana, che, come vedremo, include le tre precedenti.

Poiché i teoremi di equivalenza valgono solo nel caso discreto, da questo momento in poi assumiamo che lo spazio $\mathcal{X}$ delle variabili aleatorie associate ai vertici di un grafo sia finito.

Definizione 1.18 *Una probabilità $\mathbb{P}$ su un grafo $\mathbb{G} = (V, E)$* fattorizza *se per ogni sottoinsieme completo $A \subseteq V$ esiste una funzione non negativa ψ_A che dipende solo dalle X_α tali che $\alpha \in A$ tale che la probabilità congiunta $\mathbb{P}[V]$ può essere scritta come:*

$$\mathbb{P}[V = x] = \prod_{A \text{ completi}} \psi_A(x) .$$

In generale le funzioni ψ_A non sono univocamente determinate. È possibile senza perdere generalità far apparire soltanto le cliques nella formula, cioè scrivere:

$$\mathbb{P}[V = x] = \prod_{C \in \mathbb{C}} \psi_c(x) \tag{1.1}$$

dove $\mathbb{C}$ è l'insieme di tutte le cliques del grafo.

Se la probabilità $\mathbb{P}$ è strettamente positiva, indicando con $\phi_A(x) = \log(\psi_A(x))$ l'equazione 1.1 può essere riscritta come:

$$\log(\mathbb{P}[V = x]) = \sum_{C \in \mathbb{C}} \phi_C(x) \ .$$

Proposizione 1.19 *Per ciascun grafo $\mathbb{G}$ e ciascuna probabilità $\mathbb{P}$ definita su $\mathcal{X}$, spazio delle variabili aleatorie associate ai vertici di $\mathbb{G}$, vale che:*

$$\mathbb{P} \textit{ fattorizza} \Rightarrow \mathbb{P} \textit{ globale} \Rightarrow \mathbb{P} \textit{ locale} \Rightarrow \mathbb{P} \textit{ pairwise} \ .$$

Dimostrazione: Occorre solamente dimostrare che $\mathbb{P}$ fattorizza $\Rightarrow$ $\mathbb{P}$ globale.

Supponiamo quindi che $\mathbb{P}$ fattorizzi. Siano A, B e C tre sottoinsiemi disgiunti di V tali che C separa A da B. Sia $\tilde{A}$ la componente connessa del sottografo $\mathbb{G}_{V \setminus C}$ che contiene A. Sia $\tilde{B} = V \setminus \left(\tilde{A} \cup C\right)$. Dato che C separa A da B in $\mathbb{G}_{V \setminus C}$ essi stanno in due componenti connesse distinte e ciascuna clique di $\mathbb{G}$ appartiene o a $\tilde{A} \cup C$ o a $\tilde{B} \cup C$. Se $\mathbb{C}_A$ è l'insieme delle cliques contenute in $\tilde{A} \cup C$ si può scrivere la probabilità $\mathbb{P}$ come:

$$\mathbb{P}[V = x] = \prod_{C \in \mathbb{C}} \psi_c(x) = \prod_{C \in \mathbb{C}_A} \psi_c(x) \prod_{C \in \mathbb{C} \setminus \mathbb{C}_A} \psi_c(x) = h(x_{\tilde{A} \cup C}) k(x_{\tilde{B} \cup C}) \ .$$

Quindi per la proprietà d) delle distribuzioni condizionate si ha:

$$\tilde{A} \perp\!\!\!\perp \tilde{B} \mid C \ .$$

Applicando la proprietà ii) dell'indipendenza condizionata si ha che

$$A \perp\!\!\!\perp B \mid C \ .$$

Quindi $\mathbb{P}$ gode della proprietà di Markov globale. □

In generale, la proprietà di fattorizzazione e la proprietà di Markov globale sono distinte. Una condizione sufficiente affinché esse siano uguali è la seguente.

Proposizione 1.20 *Sia $\mathbb{G} = (V, E)$ e supponiamo che esso sia debolmente decomponibile. Sia $\mathbb{P}$ una distribuzione di probabilità su V. Allora vale:*

$$\mathbb{P} \textit{ globale} \Leftrightarrow \mathbb{P} \textit{ fattorizza} \ .$$

Dimostrazione: Occorre soltanto dimostrare che $\mathbb{P}$ globale $\Rightarrow$ $\mathbb{P}$ fattorizza.

Supponiamo che $\mathbb{P}$ goda della proprietà di Markov globale e dimostriamo la tesi per induzione sulla cardinalità di V. Indichiamo con $p(x)$ la probabilità che V sia uguale a x.

Se $\mid V \mid = 1$ la tesi è banalmente vera. Supponiamo che la tesi sia vera per $\mid V \mid \leq n-1$ e sia $\mid V \mid = n$. Poiché $\mathbb{G}$ è debolmente decomponibile esiste una sua decomposizione debole (A, B, C) tale che i sottografi $\mathbb{G}_{A\cup C}$ e $\mathbb{G}_{B\cup C}$ sono debolmente decomponibili.

Dato che $\mathbb{P}$ gode della proprietà di Markov globale, per la proprietà a) dell'indipendenza condizionata vale che, per ogni $x \in \mathcal{X}$:

$$p(x) = \frac{p(x_{A\cup C})p(x_{B\cup C})}{p(x_C)}$$

Sia $\mathbb{A}$ l'insieme di tutte le cliques del sottografo $\mathbb{G}_{A\cup C}$ e sia $\mathbb{B}$ l'insieme di tutte le cliques del sottografo $\mathbb{G}_{B\cup C}$.

È possibile applicare l'ipotesi induttiva ai due sottografi, ottenendo che esistono delle funzioni θ e delle funzioni ψ tali che, per ogni $x \in \mathcal{X}$:

$$p(x_{A\cup C}) = \prod_{D\in\mathbb{A}} \theta_D(x_{A\cup C})$$

e

$$p(x_{B\cup C}) = \prod_{D\in\mathbb{B}} \phi_D(x_{B\cup C}) \ .$$

Sia $\mathbb{C}$ l'insieme delle cliques di $\mathbb{G}$, $\mathbb{C} \subseteq \mathbb{A} \cup \mathbb{B}$. Combinando le due formule, si ottiene:

$$p(x) = \frac{\prod_{D\in\mathbb{A}} \theta_D(x_{A\cup C}) \prod_{D\in\mathbb{B}} \phi_D(x_{B\cup C})}{p(x_C)} = \prod_{D\in\mathbb{C}} \psi_D(x)$$

scegliendo le funzioni ψ in modo opportuno. Abbiamo quindi trovato una fattorizzazione di $\mathbb{P}$. □

Vediamo, nella prossima proposizione, come le quattro proprietà sono equivalenti per le probabilità $\mathbb{P}$ che sono strettamente positive su tutto lo spazio in cui sono definite.

Proposizione 1.21 *Sia $\mathbb{G} = (V, E)$ un grafo e sia $\mathbb{P}$ una probabilità definita su $\mathcal{X}$, spazio delle variabili aleatorie ad esso associate, tale che:*

$$\mathbb{P}[V = x] \geq 0 \quad \forall x \in \mathcal{X} \ .$$

Allora:

$$\mathbb{P} \textit{ pairwise } \Leftrightarrow \mathbb{P} \textit{ fattorizza } \ .$$

Dimostrazione: Nel seguito indicheremo con $p(x)$ la probabilità $\mathbb{P}[V = x]$. Supponiamo che $\mathbb{P}$ goda della proprietà di Markov pairwise e sia x^* un generico elemento di $\mathcal{X}$. Per ogni insieme $A \subseteq V$ definiamo:

$$H_A(x) = \log\left(p(x_A, x^*_{A^c})\right) \ .$$

Dove $(x_A, x^*_{A^c})$ è quell'elemento $y \in \mathcal{X}$ tale che $y_\alpha = x_\alpha$ per ogni $\alpha \in A$ e $y_\alpha = x^*_\alpha$ per ogni $\alpha \notin A$.

Poiché x^* è fissato la funzione H_A dipende da x soltanto attraverso le componenti x_A. Definiamo, per ogni insieme $A \subseteq V$:

$$\phi_A(x) = \sum_{B \subseteq A} (-1)^{|A \setminus B|} H_B(x) \ .$$

Anche la funzione Φ dipende da x soltanto attraverso la componente x_A. Applicando il lemma A.1 inversione di Mobius si ottiene:

$$H_A(x) = \sum_{B \subseteq A} \phi_B(x) \ .$$

Scegliamo ora come insieme A l'intero insieme V. Poiché $V^c = \emptyset$ si ottiene:

$$H_V(x) = \sum_{A \subseteq V} \phi_A(x) = \log\left(p(x)\right) \ .$$

Per concludere occorre solo dimostrare che

$$\phi_A(x) \neq 0 \Leftrightarrow A \text{ è completo} \ .$$

Supponiamo che A sia non completo. Siano α e β due vertici $\in A$ tali che $\alpha \not\sim \beta$. Sia $C = A \setminus \{\alpha, \beta\}$. Allora la funzione ϕ_A può essere scritta come:

$$\phi_A(x) = \sum_{B \subseteq C} (-1)^{|C \setminus B|} \left[H_B(x) - H_{B \cup \{\alpha\}}(x) - H_{B \cup \{\beta\}}(x) + H_{B \cup \{\alpha,\beta\}}(x)\right] \ .$$

Sia $D = V \setminus \{\alpha, \beta\}$. Dato che per ipotesi vale la proprietà di Markov pairwise, si ha che $\alpha \perp\!\!\!\perp \beta \mid D$.

Allora, per ogni $B \subseteq C$, applicando la proprietà e) delle distribuzioni condizionate si ottiene:

$$\begin{aligned}
&H_{B \cup \{\alpha,\beta\}}(x) - H_{B \cup \{\alpha\}}(x) = \\
&\qquad \log\left(p(x_B, x_\alpha, x_\beta, x^*_{D \setminus B})\right) - \log\left(p(x_B, x_\alpha, x^*_\beta, x^*_{D \setminus B})\right) = \\
&\log\left(\frac{p(x_B, x_\alpha, x_\beta, x^*_{D \setminus B})}{p(x_B, x_\alpha, x^*_\beta, x^*_{D \setminus B})}\right) = \log\left(\frac{p(x_\alpha \mid x_B, x^*_{D \setminus B}) p(x_\beta, x_B, x^*_{D \setminus B})}{p(x_\alpha \mid x_B, x^*_{D \setminus B}) p(x^*_\beta, x_B, x^*_{D \setminus B})}\right) = \\
&\log\left(\frac{p(x^*_\alpha \mid x_B, x^*_{D \setminus B}) p(x_\beta, x_B, x^*_{D \setminus B})}{p(x^*_\alpha \mid x_B, x^*_{D \setminus B}) p(x^*_\beta, x_B, x^*_{D \setminus B})}\right) = \log\left(\frac{p(x_B, x^*_\alpha, x_\beta, x^*_{D \setminus B})}{p(x_B, x^*_\alpha, x^*_\beta, x^*_{D \setminus B})}\right) = \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad H_{B \cup \{\beta\}}(x) - H_B(x) \ .
\end{aligned}$$

Quindi per ogni valore di x:

$$H_B(x) - H_{B\cup\{\alpha\}}(x) - H_{B\cup\{\beta\}}(x) + H_{B\cup\{\alpha,\beta\}}(x) = 0$$

e pertanto:

$$\phi_A(x) = 0 \ .$$

□

Dato un insieme $A \subseteq V$ la funzione $\phi_A(x)$ è detta *interazione*. Se A possiede un unico elemento essa è *detta effetto principale*. Se A possiede n elementi essa è detta *interazione di ordine* $n-1$ sugli elementi di A.

1.9 Proprietà di Markov nei grafi decomponibili

Vediamo come le proprietà di Markov si comportano sui grafi che possiedono una decomposizione debole.

Proposizione 1.22 *Sia $\mathbb{G} = (V, E)$ e sia (A, B, C) una sua decomposizione debole. Sia $\mathbb{P}$ una distribuzione di probabilità su V. Allora $\mathbb{P}$ fattorizza se e solo se le distribuzioni marginali $\mathbb{P}_{A\cup C}$ e $\mathbb{P}_{B\cup C}$ fattorizzano rispetto ai sottografi $\mathbb{G}_{A\cup C}$ e $\mathbb{G}_{B\cup C}$ e inoltre, indicando $p(x)$ la probabilità che V sia uguale a x, vale:*

$$p(x)p_C(x_C) = p_{A\cup C}(x_{A\cup C})p_{B\cup C}(x_{B\cup C}) \ .$$

Dimostrazione: Supponiamo che $\mathbb{P}$ fattorizzi. Sia $\mathbb{C}$ l'insieme di tutte le clique del grafo $\mathbb{G}$. Allora per ogni $C \in \mathbb{C}$ esiste una funzione ψ_C tale che:

$$p(x) = \prod_{C\in\mathbb{C}} \psi_C(x) \ .$$

Dato che (A, B, C) è una decomposizione debole di $\mathbb{G}$, ogni clique del grafo appartiene a $A \cup C$ oppure appartiene a $B \cup C$.

Sia $\mathbb{A}$ l'insieme di tutte le cliques del sottografo $\mathbb{G}_{A\cup C}$ e sia $\mathbb{B}$ l'insieme di tutte le cliques del sottografo $\mathbb{G}_{B\cup C}$. Allora si ha:

$$p(x) = \prod_{C\in\mathbb{A}} \psi_C(x) \prod_{C\in\mathbb{B}\setminus\mathbb{A}} \psi_C(x) = h(x_{A\cup C})k(x_{B\cup C}) \ .$$

E quindi la probabilità $\mathbb{P}$ fattorizza anche nei due sottografi indotti dalla decomposizione.

Abbiamo inoltre che:

$$p_{A\cup C}(x_{A\cup C}) = \sum_{y:y_{A\cup C}=x_{A\cup C}} (h(y_{A\cup C})k(y_{B\cup C})) = \\ h(x_{A\cup C}) \sum_{y:y_{A\cup C}=x_{A\cup C}} k(y_{B\cup C}) = h(x_{A\cup C})K(x_C)$$

dove si è indicato con:

$$K(x_C) = \sum_{y:y_{A\cup C}=x_{A\cup C}} k(y_{B\cup C}) \ .$$

Allo stesso modo otteniamo che:

$$p_{B\cup C}(x_{B\cup C}) = k(x_{B\cup C})H(x_c)$$

dove si è indicato con:

$$H(x_C) = \sum_{y:y_{B\cup C}=x_{B\cup C}} h(y_{A\cup C}) \ .$$

Infine, sommando un'ultima volta, si ottiene:

$$p_C(x_C) = \sum_{y:y_C=x_C} (h(y_{A\cup C})K(x_C))) = \\ K(x_C) \sum_{y:y_C=x_C} h(y_{A\cup C}) = K(x_C)H(x_C) \ .$$

E quindi si può concludere che:

$$p_{B\cup C}(x_{B\cup C})p_{A\cup C}(x_{A\cup C}) = k(x_{B\cup C})H(x_c)h(x_{A\cup C})K(x_C) = p(x)p_C(x_C) \ .$$

Viceversa supponiamo che le probabilità marginali $\mathbb{P}_{A\cup C}$ e $\mathbb{P}_{B\cup C}$ fattorizzino e che inoltre valga:

$$p(x)p_C(x_C) = p_{A\cup C}(x_{A\cup C})p_{B\cup C}(x_{B\cup C}) \ .$$

Definiamo la seguente funzione:

$$\psi_C(x_C) = \begin{cases} \frac{1}{p_C(x_C)} & \text{se } p_C(x_C) \neq 0 \\ 0 & \text{se } p_C(x_C) = 0 \end{cases} \ .$$

Dato che $p_C(x_C)$ è ottenuta nel seguente modo:

$$p_C(x_C) = \sum_{y:y_C=x_C} p(y)$$

abbiamo che se $p_C(x_C) = 0$ allora anche $p(x) = 0$.

Otteniamo quindi che:

$$p(x) = p_{A\cup C}(x_{A\cup C})p_{B\cup C}(x_{B\cup C})\psi_C(x_C) = \psi_C(x_C) \prod_{D\in\mathbb{A}} \psi_D(x) \prod_{D\in\mathbb{B}} \psi_D(x)$$

e quindi $\mathbb{P}$ fattorizza rispetto a $\mathbb{G}$. □

1.10 Riferimenti bibliografici

Le sezioni relative alle sequenze perfette, discusse nel paragrafo 1.3, e alle proprietà di Markov globale, locale e pairwise, discusse nel paragrafo 1.6, sono tratte dal testo di Lauritzen *Graphical Models* [7].

Le condizioni di equivalenza fra le proprietà Markoviane discusse nel paragrafo 1.7 sono state tratte dal testo di Matus *On Equivalence of Markov Properties over Undirected Graphs* [9]. La proposizione 1.15, che lega le condizioni di Matus alle definizioni date da Lautizen, è originale.

Il paragrafo 1.8, dedicato alla proprietà di fattorizzazione e ai suoi legami con le altre proprietà markoviane, è tratto dal testo di Lauritzen *Lectures on Contingency Tables* [8].

Capitolo 2

Hidden Markov Models

Un *Hidden Markov Model*, o modello di Markov nascosto, in sigla HMM, è un processo di Markov i cui stati non sono osservabili direttamente. Esso ha un numero finito di stati che evolvono secondo una catena di Markov. Ogni stato genera un evento che è osservabile. Obiettivo dello statistico è quindi, utilizzando la sequenza di eventi osservata, individuare la sequenza di stati più probabile.

Esistono in letteratura tre definizioni di un processo di Markov nascosto, che sono tutte equivalenti, come verrà dimostrato in seguito. Esse sono.

Definizione 2.1 *Sia $\{Y_t\}_{t>0}$ un processo stocastico stazionario definito su uno spazio finito $\boldsymbol{V}$ con $\mid \boldsymbol{V} \mid = M$. Si dice che $\{Y_t\}$ è un* Hidden Markov Model *se:*

1. *esiste un processo di Markov $\{X_t\}_{t>0}$ definito su uno spazio finito $\boldsymbol{Q}$, con $\mid \boldsymbol{Q} \mid = N$ e inoltre esiste una funzione*

$$f : \boldsymbol{Q} \to \boldsymbol{V}$$

tale che per ogni t valga:

$$Y_t = f(X_t) \ .$$

2. *esiste un processo di Markov $\{X_t\}_{t>0}$ definito su uno spazio finito $\boldsymbol{Q}$, con $\mid \boldsymbol{Q} \mid = N$, con legge iniziale $\pi \in [0,1]^N$ e matrice di transizione $A \in [0,1]^{N \times N}$, e inoltre esiste una matrice $B \in [0,1]^{N \times M}$ tale che per ogni $i \in \boldsymbol{Q}$ e per ogni $j \in \boldsymbol{V}$ si abbia:*

$$\mathbb{P}[Y_t = j \mid X_t = i] = b_{ij} \ .$$

3. *esiste un processo stocastico stazionario* $\{X_t\}_{t>0}$ *definito su uno spazio finito* $\boldsymbol{Q}$*, con* $\mid \boldsymbol{Q} \mid = N$ *tale che il processo congiunto* $\{X_t, Y_t\}$ *sia di Markov e inoltre vale per ogni* t*:*

$$\mathbb{P}\left[X_t = x_t, Y_t = y_t \mid X_{t-1} = x_{t-1}, Y_{t-1} = t_{t-1}\right] = \mathbb{P}\left[X_t = x_t, Y_t = y_t \mid X_{t-1} = x_{t-1}\right] \ .$$

Proposizione 2.1 *Le tre definizioni di* Hidden Markov Model *sono equivalenti.*

Dimostrazione: Sia $\{Y_t\}$ un processo stocastico stazionario definito su uno spazio finito $\mathbf{V}$ con $\mid \mathbf{V} \mid = M$.

1)$\Rightarrow$ 2)
Supponiamo che esista un processo di Markov $\{X_t\}$ definito su uno spazio finito $\mathbf{Q}$, con $\mid \mathbf{Q} \mid = N$, e una funzione

$$f : \mathbf{Q} \to \mathbf{V}$$

tale che per ogni t valga:

$$Y_t = f(X_t) \ .$$

Sia $B \in [0,1]^{N \times M}$ una matrice tale che, per ogni $i \in \mathbf{Q}$ e per ogni $j \in \mathbf{V}$ valga:

$$b_{ij} = 1 \quad \text{se} \quad f(i) = j$$

$$b_{ij} = 0 \quad \text{altrimenti} \ .$$

Allora evidentemente:

$$\mathbb{P}[Y_t = j \mid X_t = i] = b_{ij}$$

e quindi è verificata la seconda definizione.

2)$\Rightarrow$ 3)
Supponiamo che esistano un processo di Markov $\{X_t\}$ definito su uno spazio finito $\mathbf{Q}$, con $\mid \mathbf{Q} \mid = N$, con legge iniziale $\pi \in [0,1]^N$ e con matrice di transizione $A \in [0,1]^{N \times N}$, e una matrice $B \in [0,1]^{N \times M}$ tale che si abbia:

$$\mathbb{P}[Y_t = j \mid X_t = i] = b_{ij} \ .$$

Allora il processo congiunto $\{X_t, Y_t\}$ è di Markov. Infatti si ha che:

$$\begin{aligned}
&\mathbb{P}\left[X_{t+1} = x_{t+1}, Y_{t+1} = y_{t+1} \mid X_t = x_t, Y_t = y_t, \ldots, X_0 = x_0, Y_0 = y_0\right] = \\
&\quad \mathbb{P}\left[Y_{t+1} = y_{t+1} \mid X_{t+1} = x_{t+1}, X_t = x_t, Y_t = y_t, \ldots, X_0 = x_0, Y_0 = y_0\right] \\
&\quad\quad \mathbb{P}\left[X_{t+1} = x_{t+1} \mid X_t = x_t, Y_t = y_t, \ldots, X_0 = x_0, Y_0 = y_0\right] = \\
&\quad \mathbb{P}\left[Y_{t+1} = y_{t+1} \mid X_{t+1} = x_{t+1}\right] \mathbb{P}\left[X_{t+1} = x_{t+1} \mid X_t = x_t\right] = a_{x_t, x_{t+1}} b_{x_{t+1}, y_{t+1}} \ .
\end{aligned}$$

Osserviamo inoltre che la probabilità:

$$\mathbb{P}\left[X_{t+1} = x_{t+1}, Y_{t+1} = y_{t+1} \mid X_t = x_t, Y_t = y_t\right] = b_{x_{t+1},y_{t+1}} a_{x_t,x_{t+1}} = \mathbb{P}\left[X_{t+1} = x_{t+1}, Y_{t+1} = y_{t+1} \mid X_t = x_t\right] \ .$$

Infatti non dipende da y_t. Quindi sono verificate tutte le condizioni della terza definizione.

3)$\Rightarrow$ 1)
Supponiamo che esista un processo stocastico stazionario $\{X_t\}$ definito su uno spazio finito $\mathbf{Q}$, con $\mid \mathbf{Q} \mid = N$ tale che il processo congiunto $\{X_t, Y_t\}$ sia di Markov e inoltre valga per ogni t:

$$\mathbb{P}\left[X_t = x_t, Y_t = y_t \mid X_{t-1} = x_{t-1}, Y_{t-1} = t_{t-1}\right] = \mathbb{P}\left[X_t = x_t, Y_t = y_t \mid X_{t-1} = x_{t-1}\right] \ .$$

Allora chiaramente vale che per ogni t:

$$Y_t = f(X_t, Y_t)$$

per una funzione f opportuna, che rappresenta la proiezione su Y_t. Quindi è verificata anche la prima definizione. □

Noi utilizzeremo la seconda definizione e denoteremo quindi un Hidden Markov Model con:

$$\lambda = (A, B, \pi) \ .$$

Senza perdere generalità, essendo gli spazi $\mathbf{Q}$ e $\mathbf{V}$ finiti, possiamo indicare:

$$\mathbf{V} = \{0, \ldots, M-1\} \quad \text{e} \quad \mathbf{Q} = \{0, \ldots, N-1\} \ .$$

Il processo sottostante, $\{X_t\}$, è una catena di Markov quindi per ogni t e per ogni sequenza di stati vale la proprietà di Markov:

$$\mathbb{P}[X_t = x_t \mid X_{t-1} = x_{t-1}, \ldots, X_1 = x_1, \lambda] = \mathbb{P}[X_t = x_t \mid X_{t-1} = x_{t-1}, \lambda] \ .$$

Si assume inoltre che il processo osservato, $\{Y_t\}$, goda della seguente proprietà:

$$\mathbb{P}[Y_t = y_t, \ldots, Y_1 = y_1 \mid X_t = x_t, \ldots, X_1 = x_1, \lambda] = \prod_{u=1}^{t} \mathbb{P}[Y_u = y_u \mid X_u = x_u, \lambda] \ .$$

La sequenza Y_t è quindi *condizionatamente indipendente* alla sequenza X_t.

Definizione 2.2 *Dato un Hidden Markov Model, tre sono i problemi fondamentali che gli statistici sono chiamati a risolvere:*

1. verosimiglianza del campione osservato;

2. determinare la sequenza più probabile di stati nascosti;

3. determinare il modello λ migliore per il campione osservato.

Nei prossimi paragrafi cercheremo di determinare la soluzione a questi tre problemi.

2.1 Notazioni e risultati preliminari

Consideriamo un Hidden Markov Model $\lambda = (A, B, \pi)$ e un campione osservato di lunghezza T:

$$y = (y_1, \ldots, y_T) \ .$$

Per ogni $i \in \{0, \ldots, N-1\}$ e per ogni $t \in \{1, \ldots, T\}$ definiamo le seguenti quantità:

$$\alpha_t(i) = \mathbb{P}[Y_1 = y_1, \ldots, Y_t = y_t, X_t = i \mid \lambda]$$

$$\beta_t(i) = \mathbb{P}[Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, \lambda]$$

$$\gamma_t(i) = \mathbb{P}[X_t = i \mid Y = y, \lambda] \ .$$

Per ogni $t \in \{1, \ldots, T-1\}$ e per ogni $i, j \in \{0, \ldots, N-1\}$ definiamo la seguente quantità:

$$\varepsilon_t(i, j) = \mathbb{P}[X_t = i, X_{t+1} = j \mid Y = y, \lambda] \ .$$

Valgono le seguenti proposizioni che precisano le quantità sopra definite in termini del modello λ.

Proposizione 2.2 *Per ogni $j \in \{0, \ldots, N-1\}$ e per ogni valore di $t \in \{1, \ldots, T\}$ si ha:*

$$\alpha_1(j) = \pi_j b_{j,y_1}$$

$$\alpha_t(j) = \left(\sum_{i=0}^{N-1} \alpha_{t-1}(i) a_{ij} \right) b_{j,y_t} \ .$$

Dimostrazione: Per $t = 1$ si ha subito:

$$\alpha_1(i) = \mathbb{P}[Y_1 = y_1, X_1 = i \mid \lambda] = \mathbb{P}[Y_1 = y_1 \mid X_1 = i, \lambda]\mathbb{P}[X_1 = i \mid \lambda] = \pi_i b_{i,y_1} \ .$$

Consideriamo un generico $t > 1$. Otteniamo:

$$\alpha_t(j) =$$
$$\sum_{i=0}^{N-1} \mathbb{P}[Y_1 = y_1, \ldots, Y_t = y_t, X_{t-1} = i, X_t = j \mid \lambda] =$$
$$\sum_{i=0}^{N-1} \mathbb{P}[Y_t = y_t, X_t = j \mid Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, X_{t-1} = i, \lambda]$$
$$\mathbb{P}[Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, X_{t-1} = i \mid \lambda] =$$
$$\sum_{i=0}^{n-1} \alpha_{t-1}(i)\mathbb{P}[Y_t = y_t \mid X_t = j, X_{t-1} = i, Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, \lambda]$$
$$\mathbb{P}[X_t = j \mid Y_1 = y_1, \ldots, Y_{t-1} = y_{t-1}, X_{t-1} = i, \lambda] =$$
$$\sum_{i=0}^{n-1} \alpha_{t-1}(i) b_{j,y_t} a_{ij} \ .$$

□

Proposizione 2.3 *Fissato*

$$\beta_T(i) = 1 \quad \forall i \in \{0, \ldots, N-1\}$$

si ha che per ogni $i \in \{0, \ldots, N-1\}$ *e per ogni* $t \in \{1, \ldots, T-1\}$*:*

$$\beta_t(i) = \sum_{j=0}^{N-1} a_{ij} b_{j,y_{t+1}} \beta_{t+1}(j) \ .$$

Dimostrazione: Per $t = T - 1$ si ha immediatamente che:

$$\beta_{T-1}(i) = \mathbb{P}[Y_T = y_T \mid X_{T-1} = i, \lambda] =$$
$$\sum_{j=0}^{N-1} \mathbb{P}[Y_T = y_T \mid X_T = j, \lambda]\mathbb{P}[X_T = j \mid X_{T-1} = i] = \sum_{j=0}^{N-1} b_{j,y_T} a_{ij} \beta_T(j) \ .$$

Consideriamo un generico $t < T-1$ e un generico $i \in \{0, \ldots, N-1\}$. Si ha:

$$\beta_t(i) = \sum_{j=0}^{N-1} \mathbb{P}[Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T, X_{t+1} = j \mid X_t = i, \lambda] =$$

$$\sum_{j=0}^{N-1} \mathbb{P}[Y_{t+2} = y_{t+2}, \ldots, Y_T = y_T \mid Y_{t+1} = y_{t+1}, X_{t+1} = j, X_t = i, \lambda]$$

$$\mathbb{P}[Y_{t+1} = y_{t+1}, X_{t+1} = j \mid X_t = i, \lambda] =$$

$$\sum_{j=0}^{N-1} \beta_{t+1}(j) \mathbb{P}[Y_{t+1} = y_{t+1} \mid X_{t+1} = j, X_t = i, \lambda] \mathbb{P}[X_{t+1} = j \mid X_t = i, \lambda] =$$

$$\sum_{j=0}^{N-1} a_{ij} b_{j,y_{t+1}} \beta_{t+1}(j) \ .$$

□

Proposizione 2.4 *Vale la seguente uguaglianza:*

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\mathbb{P}[Y = y \mid \lambda]} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)\beta_t(j)} \ .$$

Dimostrazione: Osserviamo che:

$$\alpha_t(i)\beta_t(i) =$$

$$\mathbb{P}[Y_1 = y_1, \ldots, Y_t = y_t, X_t = i \mid \lambda] \mathbb{P}[Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, \lambda] =$$

$$\mathbb{P}[Y_1 = y_1, \ldots, Y_t = y_t \mid X_t = i, \lambda] \mathbb{P}[Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T \mid X_t = i, \lambda] \mathbb{P}[X_t = i \mid \lambda] =$$

$$\mathbb{P}[Y_1 = y_1, \ldots, Y_T = y_T \mid X_t = i, \lambda] \mathbb{P}[X_t = i \mid \lambda] =$$

$$\mathbb{P}[Y_1 = y_1, \ldots, Y_T = y_T, X_t = i \mid \lambda] \ .$$

Da ciò segue anche che:

$$\mathbb{P}[Y = y \mid \lambda] = \sum_{i=0}^{N-1} \mathbb{P}[Y = y, X_t = i \mid \lambda] = \sum_{i=0}^{N-1} \alpha_t(i)\beta_t(i)$$

e quindi:

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)\beta_t(j)} \ .$$

□

Proposizione 2.5 *Vale la seguente uguaglianza:*

$$\varepsilon_t(i,j) = \frac{\alpha_t(i) a_{ij} b_{j,y_{t+1}} \beta_{t+1}(j)}{\mathbb{P}[Y = y \mid \lambda]} = \frac{\alpha_t(i) a_{ij} b_{j,y_{t+1}} \beta_{t+1}(j)}{\sum_{h=0}^{N-1} \sum_{k=0}^{N-1} \alpha_t(h) a_{hk} b_{k,y_{t+1}} \beta_{t+1}(k)} \ .$$

Dimostrazione: Osserviamo che:

$$\begin{aligned}
&\mathbb{P}[X_t = i, X_{t+1} = j, Y = y \mid \lambda] = \\
&\quad \mathbb{P}[Y_{t+1} = y_{t+1}, \ldots, Y_T = y_T, X_{t+1} = j \mid Y_1 = y_1, \ldots, Y_t = y_t, X_t = i, \lambda] \\
&\qquad \mathbb{P}[Y_1 = y_1, \ldots, Y_t = y_t, X_t = i \mid \lambda] = \\
&\mathbb{P}[Y_{t+1} = y_{t+1} \mid Y_1 = y_1, \ldots Y_t = y_t, Y_{t+2} = y_{t+2}, \ldots, Y_T = y_t, X_t = i, X_{t+1} = j, \lambda] \\
&\quad \mathbb{P}[Y_{t+2} = y_{t+2}, \ldots, Y_T = y_t, X_{t+1} = j \mid Y_1 = y_1, \ldots Y_t = y_t, X_t = i, \lambda]\alpha_t(i) = \\
&\qquad\qquad b_{j,y_{t+1}}\beta_{t+1}(j)a_{ij}\alpha_t(i) \ .
\end{aligned}$$

Inoltre si ha che:

$$\begin{aligned}
&\mathbb{P}[Y = y \mid \lambda] = \\
&\quad \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} \mathbb{P}[Y = y, X_t = i, X_{t+1} = j \mid \lambda] = \sum_{i=0}^{N-1}\sum_{j=0}^{N-1} b_{j,y_{t+1}}\beta_{t+1}(j)a_{ij}\alpha_t(i) \ .
\end{aligned}$$

□

2.2 Verosimiglianza del campione osservato

Dato un Hidden Markov Model λ e un campione osservato di lunghezza T

$$y = (y_1, \ldots, y_T)$$

si vuole trovare la probabilità di osservare tale campione dato il modello, cioè:

$$\mathbb{P}[Y = y \mid \lambda] \ .$$

Consideriamo una generica sequenza di stati della catena di Markov sottostante

$$x = (x_1, \ldots, x_T) \ .$$

La probabilità di osservare il campione y, data questa sequenza, è:

$$\mathbb{P}[Y = y \mid X = x, \lambda] = \prod_{t=1}^{T} \mathbb{P}[y_t \mid x_t, \lambda] = \prod_{t=1}^{T} b_{x_t,y_t} \ .$$

Inoltre, la probabilità della sequenza x è:

$$\mathbb{P}[X = x \mid \lambda] = \pi_{x_1} a_{x_1,x_2} \ldots a_{x_{T-1},x_T} \ .$$

Combinando le due formule otteniamo che:

$$\mathbb{P}[Y = y, X = x \mid \lambda] = \pi_{x_1} \prod_{t=2}^{T} a_{x_{i-1},x_i} \prod_{t=1}^{T} b_{x_t,y_t} \ .$$

Infine, sommando:

$$\mathbb{P}[Y = y \mid \lambda] = \sum_{(x_1,\ldots,x_T)\in\{0,\ldots,N-1\}^T} \left(\pi_{x_1} \prod_{t=2}^{T} a_{x_{i-1},x_i} \prod_{t=1}^{T} b_{x_t,y_t} \right) .$$

Tuttavia l'eccessiva complessità computazionale rende questa formula inutilizzabile. Pertanto si utilizza la procedura *Forward-Backward*.

Proposizione 2.6 *La verosimiglianza del campione y può essere scritta come:*

$$\mathbb{P}[Y = y \mid \lambda] = \sum_{i=0}^{N-1} \alpha_T(i) .$$

Dimostrazione: La dimostrazione è immediata e si ottiene osservando che:

$$\sum_{i=0}^{N-1} \alpha_T(i) = \sum_{i=0}^{N-1} \mathbb{P}[Y_1 = y_1, \ldots, Y_T = y_T, X_T = i \mid \lambda] = \mathbb{P}[Y = y \mid \lambda] .$$

□

Pertanto, calcolando le quantità $\alpha_T(i)$ utilizzando la procedura ricorsiva definita nel precedente paragrafo, è possibile calcolare la verosimiglianza $\mathbb{P}[Y = y \mid \lambda]$ in modo semplice.

2.3 Determinazione della sequenza degli stati nascosti

Dato un Hidden Markov Model $\lambda = (A, B, \pi)$ e un campione osservato di lunghezza T

$$y = (y_1, \ldots, y_T)$$

si vuole trovare la sequenza:

$$x = (x_1, \ldots, x_T)$$

che ha la maggior verosimiglianza rispetto al campione osservato.

Per ciascun $t \in \{1, \ldots, T\}$ si può determinare lo stato x_t più probabile come:

$$\hat{x}_t = \max_{i\in\{0,\ldots,N-1\}} (\mathbb{P}[X_t = i \mid Y = y, \lambda]) = \max_{i\in\{0,\ldots,N-1\}} (\gamma_t(i)) .$$

Calcolando $\hat{x}_t$ per ciascun $t \in \{1, \ldots, T\}$ si ottiene una sequenza di stati

$$\hat{x} = (\hat{x}_1, \ldots, \hat{x}_T)$$

che può essere considerata la sequenza di massima verosimiglianza.

In realtà questo modo di procedere non è del tutto corretto. Infatti bisognerebbe ricercare, tra tutte le possibili sequenze

$$x = (x_1, \dots, x_T) \in \{0, \dots, N-1\}^T$$

quella che massimizza la probabilità:

$$\mathbb{P}[X = x \mid Y = y, \lambda] \ .$$

Tuttavia questo problema è computazionalmente complesso e, al momento, non esiste un algoritmo per trovare una soluzione.

2.4 Determinazione del modello λ migliore

Dato un campione osservato di lunghezza T

$$y = (y_1, \dots, y_T)$$

si vuole trovare l'Hidden Markov Model $\lambda = (A, B, \pi)$ che massimizza la probabilità di aver osservato il campione y.

In generale, non esiste una soluzione semplice a questo problema di massimizzazione. La verosimiglianza di un HMM è molto complessa e presenta molti massimi locali. Nel prossimo paragrafo studieremo un algoritmo che, partendo da un modello λ_0 iniziale, converge ad un modello $\hat{\lambda}$ che massimizza la verosimiglianza. Questo algoritmo, così come altri algoritmi analoghi, converge però ad un massimo locale che può essere differente a seconda del modello iniziale λ_0 scelto.

La scelta del modello iniziale ottimale costituisce un problema complesso al quale cercheremo una soluzione nei capitoli successivi.

2.5 L'algoritmo di Baum Welch

Consideriamo un generico HMM $\lambda = (A, B, \pi)$. Abbiamo visto che, per ogni

$$x = (x_1, \dots, x_t) \in \{0, \dots, N-1\}^T$$

si ha:

$$\mathbb{P}[X = x, Y = y \mid \lambda] = \pi_{x_1} \prod_{t=1}^{T} b_{x_t, y_t} \prod_{t=2}^{T} a_{x_{t-1}, x_t} \ .$$

La verosimiglianza del modello, quindi, si può scrivere come:

$$\mathrm{L}(\lambda \mid y) = \sum_{x \in \{0,\dots,N-1\}^T} \mathbb{P}[X = x, Y = y \mid \lambda] = \sum_{x \in \{0,\dots,N-1\}^T} \pi_{x_1} \prod_{t=1}^{T} b_{x_t, y_t} \prod_{t=2}^{T} a_{x_{t-1}, x_t} \ .$$

Nel seguito useremo la seguente notazione. Dato un campione osservato y indichiamo:

$$\mathbb{P}[X = x, Y = y \mid \lambda] = p(x \mid \lambda)$$

e

$$\mathbb{P}[X = x \mid Y = y \mid \lambda] = p(x \mid y, \lambda) \ .$$

Definizione 2.3 *Consideriamo due modelli HMM*

$$\lambda_1 = (A_1, B_1, \pi_1), \quad \lambda_2 = (A_2, B_2, \pi_2) \ .$$

Definiamo la quantità $Q(\lambda_1, \lambda_2)$ come:

$$Q(\lambda_1, \lambda_2) = \mathbb{E}\left[\log\left(p(X \mid \lambda_2)\right) \mid Y = y, \lambda_1\right] = \sum_{x \in \{0,\ldots,N-1\}^T} \log\left(p(x \mid \lambda_2)\right) p(x \mid y, \lambda_1) \ .$$

Allora vale la seguente proposizione.

Proposizione 2.7 *Consideriamo due modelli HMM*

$$\lambda_1 = (A_1, B_1, \pi_1), \quad \lambda_2 = (A_2, B_2, \pi_2)$$

e un campione osservato

$$y = (y_1 \ldots, y_T) \ .$$

Supponiamo che valga:

$$Q(\lambda_1, \lambda_2) \geq Q(\lambda_1, \lambda_1) \ .$$

Allora

$$L(y \mid \lambda_2) \geq L(y \mid \lambda_1)$$

e inoltre vale l'uguaglianza se e solo se

$$p(x \mid \lambda_1) = p(x \mid \lambda_2) \quad \forall x \in \{0, \ldots, N-1\}^T \ .$$

Dimostrazione: Supponiamo che valga

$$Q(\lambda_1, \lambda_2) \geq Q(\lambda_1, \lambda_1)$$

e consideriamo il logaritmo del rapporto delle verosimiglianze.

$$\log\left(\frac{\mathrm{L}(y \mid \lambda_2)}{\mathrm{L}(y \mid \lambda_1)}\right) = \log\left(\sum_{x \in \{0,\ldots,N-1\}^T} \frac{p(x \mid \lambda_2)}{\mathrm{L}(y \mid \lambda_1)}\right) = \log\left(\sum_{x \in \{0,\ldots,N-1\}^T} \frac{p(x \mid \lambda_1)}{\mathrm{L}(y \mid \lambda_1)} \frac{p(x \mid \lambda_2)}{p(x \mid \lambda_1)}\right) =$$

$$\log\left(\sum_{x \in \{0,\ldots,N-1\}^T} p(x \mid y, \lambda_1) \frac{p(x \mid \lambda_2)}{p(x \mid \lambda_1)}\right) = \log\left(\mathbb{E}\left[\frac{p(x \mid \lambda_2)}{p(x \mid \lambda_1)} \mid Y = y, \lambda_1\right]\right) \geq$$

$$\mathbb{E}\left[\log\left(p(x \mid \lambda_2)\right) - \log\left(p(x \mid \lambda_1)\right) \mid Y = y, \lambda_1\right] =$$

$$\mathbb{E}\left[\log\left(p(x \mid \lambda_2)\right) \mid Y = y, \lambda_1\right] - \mathbb{E}\left[\log\left(p(x \mid \lambda_1)\right) \mid Y = y, \lambda_1\right] =$$

$$\left[Q(\lambda_1, \lambda_2) - Q(\lambda_1, \lambda_1)\right] \geq 0 \ .$$

La prima disuguaglianza è stata ottenuta applicando la disuguaglianza di Jensen. La funzione $\log(t)$, infatti, è strettamente concava e quindi per ogni variabile aleatoria X:

$$\mathbb{E}\left[\log(X)\right] \leq \log\left(\mathbb{E}[X]\right) \ .$$

L'ugualianza si ha solo nel caso in cui $\frac{p(x|\lambda_2)}{p(x|\lambda_1)}$ sia costante per ogni $x \in \{0,\ldots,N-1\}^T$, cioè nel caso in cui:

$$p(x \mid \lambda_1) = p(x \mid \lambda_2) \quad \forall x \in \{0,\ldots,N-1\}^T \ .$$

□

Proposizione 2.8 *Supponiamo di aver osservato un campione $y = (y_1,\ldots,y_T)$. Sia $\lambda = (A,B,\pi)$ un generico modello HMM e sia $\tau(\lambda)$ una funzione continua tale che, per ogni λ_c fissato, $\tau(\lambda_c)$ sia un punto critico di $Q(\lambda_c,\lambda)$, considerata come funzione di λ.*

Allora ogni punto fisso di τ, cioè ogni λ tale che $\tau(\lambda) = \lambda$, è un punto critico della verosimiglianza $L(\lambda \mid y)$.

Supponiamo inoltre che per ogni λ valga:

$$L(\tau(\lambda) \mid y) > L(\lambda \mid y) \ .$$

Allora , per ogni λ_0 fissato, la successione

$$\left(\tau^n(\lambda_0)\right)_{n\geq 0}$$

converge a un punto fisso di τ.

Dimostrazione: Dato che $\lambda = (A,B,\pi)$, possiamo immaginare λ come un vettore di $\mathbb{R}^s$, con:

$$s = N^2 + NM + N \ .$$

Un punto critico di una funzione in λ è un punto che annulla tutte le sue derivate parziali. Calcoliamo quindi le derivate parziali della funzione $Q(\lambda_c,\lambda)$, considerata come funzione di λ.

$$\frac{\partial}{\partial\lambda_i}Q(\lambda_c,\lambda) = \sum_{x\in\{0,\ldots,N-1\}} \frac{\partial}{\partial\lambda_i}\log\left(p(x \mid \lambda)\right)p(x \mid y,\lambda_c) =$$

$$\sum_{x\in\{0,\ldots,N-1\}} \frac{1}{p(x \mid \lambda)}\frac{\partial}{\partial\lambda_i}p(x \mid \lambda)p(x \mid y,\lambda_c) \ .$$

Supponiamo che $\tau(\lambda_c)$ sia un punto critico di $Q(\lambda_c,\lambda)$ per ogni λ_c e sia $\overline{\lambda}$ un punto fisso di τ, cioè tale che:

$$\tau(\overline{\lambda}) = \overline{\lambda} \ .$$

Allora si ha che, per ogni λ_i componente di λ:

$$0 = \frac{\partial}{\partial \lambda_i} Q(\overline{\lambda}, \lambda) \mid_{\lambda=\tau(\overline{\lambda})} = \frac{\partial}{\partial \lambda_i} Q(\overline{\lambda}, \lambda) \mid_{\lambda=\overline{\lambda}} =$$

$$\sum_{x \in \{0,\dots,N-1\}} \frac{1}{p(x \mid \overline{\lambda})} \left\{ \frac{\partial}{\partial \lambda_i} p(x \mid \lambda) \right\} \mid_{\lambda=\overline{\lambda}} p(x \mid y, \overline{\lambda}) =$$

$$\frac{1}{\mathrm{L}(\overline{\lambda} \mid y)} \sum_{x \in \{0,\dots,N-1\}} \left\{ \frac{\partial}{\partial \lambda_i} p(x \mid \lambda) \right\} \mid_{\lambda=\overline{\lambda}} = \frac{1}{\mathrm{L}(\overline{\lambda} \mid y)} \frac{\partial}{\partial \lambda_i} \mathrm{L}(\lambda \mid y) \mid_{\lambda=\overline{\lambda}} =$$

$$\frac{\partial}{\partial \lambda_i} \mathrm{l}(\lambda \mid y) \mid_{\lambda=\overline{\lambda}} \ .$$

Quindi $\overline{\lambda}$ è un punto critico per la log-verosimiglianza $\mathrm{l}(\lambda \mid y)$ e quindi anche per la verosimiglianza $\mathrm{L}(\lambda \mid y)$.

Supponiamo ora che per ogni λ valga

$$\mathrm{L}(\tau(\lambda) \mid y) > \mathrm{L}(\lambda \mid y)$$

e consideriamo la successione $(\tau^n(\lambda_0))_{n \geq 0}$. Dato che per ogni $n > 0$ $\tau^n(\lambda_0)$ è un Hidden Markov Model, cioè è costituito da due matrici stocastiche e un vettore probabilità, la successione $\{\tau^n(\lambda_0)\}_{n \geq 0}$ è definita su un insieme compatto e quindi, per il teorema di Bolzano-Weierstrass, ha un'estratta $\{\tau^{nk}(\lambda_0)\}_{nk \geq 0}$ che converge a un limite. Sia questo limite $\overline{\lambda}$.

Allora si ha che:

$$\mathrm{L}(\tau^{nk}(\lambda_0) \mid y) \leq \mathrm{L}(\tau\left(\tau^{nk}(\lambda_0)\right) \mid y) \leq \mathrm{L}(\tau^{nk+1}(\lambda_0) \mid y) \ .$$

Applicando il limite otteniamo:

$$\mathrm{L}(\overline{\lambda} \mid y) \leq \mathrm{L}(\tau\left(\overline{\lambda}\right) \mid y) \leq \mathrm{L}(\overline{\lambda} \mid y) \ .$$

Necessariamente deve essere che:

$$\tau(\overline{\lambda}) = \overline{\lambda}$$

cioè $\overline{\lambda}$ è un punto fisso di τ. □

È quindi possibile utilizzare la proposizione 2.8 per costruire una successione di modelli HMM, partendo da un modello iniziale $\lambda_0 = (A_0, B_0, \pi_0)$, che converge ad un punto di massimo della funzione di verosimiglianza.

Ad ogni passo, dato il modello λ_n ottenuto al passo precedente, il nuovo modello costruito deve essere un punto critico di $Q(\lambda_n, \lambda)$. Scegliendo $\tau(\lambda_n)$ come il massimo della funzione $Q(\lambda_n, \lambda)$, come abbiamo visto, si ottiene automaticamente che:

$$Q(\lambda_n, \tau(\lambda_n)) > Q(\lambda_n, \lambda_n)$$

e ciò implica, per la proposizione 2.7, che:

$$\mathrm{L}(\tau(\lambda_n) \mid y) > \mathrm{L}(\lambda_n \mid y) \ .$$

Questa condizione, per la proposizione 2.8, è sufficiente a garantire la convergenza dell'algoritmo ad un punto di massimo.

Vediamo ora come massimizzare la funzione $Q(\lambda_n, \lambda)$.

Proposizione 2.9 *Supponiamo di aver osservato un campione* $y = (y_1, \ldots, y_T)$. *Consideriamo un qualunque modello iniziale:*

$$\lambda_0 = (A^0, B^0, \pi^0) \ .$$

Per ogni $n > 0$ *definiamo* $\lambda_n = (A^n, B^n, \pi^n)$ *nel modo seguente:*

$$\pi_i^n = \gamma_1^{n-1}(i)$$

$$a_{ij}^n = \frac{\sum_{t=1}^{T-1} \varepsilon_t^{n-1}(i,j)}{\sum_{t=1}^{T-1} \gamma_t^{n-1}(i)}$$

$$b_{i,y_k}^n = \frac{\sum_{t:Y_t=y_k} \gamma_t^{n-1}(i)}{\sum_{t=1}^{T} \gamma_t^{n-1}(i)} \ .$$

Dove con l'indice n in alto si indicano le quantità relative al modello λ_n.

Allora la successione $(\lambda_n)_{n\geq 0}$ *converge a un punto di massimo per la verosimiglianza* $L(\lambda \mid y)$.

Dimostrazione: Per prima cosa, dato un generico $n \geq 0$, dimostriamo che λ_{n+1} è un punto di massimo per la funzione $Q(\lambda_n, \lambda)$. Osserviamo che $Q(\lambda_n, \lambda)$ può essere scritto nel modo seguente:

$$Q(\lambda_n, \lambda) = \sum_{x \in \{0,\ldots,N-1\}^T} \log\left(p(x \mid \lambda)\right) p(x \mid y, \lambda_n) =$$

$$\sum_{x \in \{0,\ldots,N-1\}^T} \log\left(\pi_{x_1} \prod_{t=1}^{T} b_{x_t,y_t} \prod_{t=2}^{T} a_{x_{t-1},x_t}\right) p(x \mid y, \lambda_n) =$$

$$\sum_{x \in \{0,\ldots,N-1\}^T} \log\left(\pi_{x_1}\right) p(x \mid y, \lambda_n) + \sum_{x \in \{0,\ldots,N-1\}^T} \left(\sum_{t=1}^{T} \log\left(b_{x_t,y_t}\right)\right) p(x \mid y, \lambda_n) +$$

$$\sum_{x \in \{0,\ldots,N-1\}^T} \left(\sum_{t=2}^{T} \log\left(a_{x_{t-1},x_t}\right)\right) p(x \mid y, \lambda_n) =$$

$$\sum_{x_1 \in \{0,\ldots,N-1\}} \log\left(\pi_{x_1}\right) \sum_{x \in \{0,\ldots,N-1\}^{T-1}} p(x_1, x \mid y, \lambda_n) +$$

$$\sum_{t=1}^{T} \sum_{x_t \in \{0,\ldots,N-1\}} \log\left(b_{x_t,y_t}\right) \sum_{(x_1,\ldots,x_{t-1},x_{t+1},\ldots,x_T) \in \{0,\ldots,N-1\}^{T-1}} p((x_1, \ldots, x_T) \mid y, \lambda_n) +$$

$$\sum_{t=2}^{T} \sum_{(x_{t-1},x_t)\in\{0,\dots,N-1\}^2} \log\left(a_{x_{t-1},x_t}\right) \sum_{(x_1,\dots,x_{t-2},x_{t+1},\dots,x_T)\in\{0,\dots,N-1\}^{T-2}} p((x_1,\dots,x_T) \mid y,\lambda_n) =$$

$$\sum_{x_1\in\{0,\dots,N-1\}} \log\left(\pi_{x_1}\right) \mathbb{P}\left[X_1 = x_1 \mid Y = y,\lambda_n\right] +$$

$$\sum_{t=1}^{T} \sum_{x_t\in\{0,\dots,N-1\}} \log\left(b_{x_t,y_t}\right) \mathbb{P}\left[X_t = x_t \mid Y = y,\lambda_n\right] +$$

$$\sum_{t=2}^{T} \sum_{(x_{t-1},x_t)\in\{0,\dots,N-1\}^2} \log\left(a_{x_{t-1},x_t}\right) \mathbb{P}\left[X_{t-1} = x_{t-1}, X_t = x_t \mid Y = y,\lambda_n\right] =$$

$$\sum_{x_1\in\{0,\dots,N-1\}} \log\left(\pi_{x_1}\right) \gamma_1^n(x_1) + \sum_{t=1}^{T} \sum_{x_t\in\{0,\dots,N-1\}} \log\left(b_{x_t,y_t}\right) \gamma_t^n(x_t) +$$

$$\sum_{t=2}^{T} \sum_{(x_{t-1},x_t)\in\{0,\dots,N-1\}^2} \log\left(a_{x_{t-1},x_t}\right) \varepsilon_{t-1}^n(x_{t-1},x_t) \ .$$

Per trovare un punto critico della funzione $Q(\lambda_n,\lambda)$ calcoliamo ed eguagliamo a 0 le sue derivate parziali. Dobbiamo quindi calcolare:

$$\frac{\partial}{\partial \pi_i} Q(\lambda_n,\lambda) \quad \forall i \in \{0,\dots,N-1\}$$

$$\frac{\partial}{\partial b_{i,y_t}} Q(\lambda_n,\lambda) \quad \forall i \in \{0,\dots,N-1\}, y_t \in \{1,\dots,M-1\}$$

$$\frac{\partial}{\partial a_{ij}} Q(\lambda_n,\lambda) \quad \forall i,j \in \{0,\dots,N-1\} \ .$$

Calcoliamo $\frac{\partial}{\partial \pi_i} Q(\lambda_n,\lambda)$. Osserviamo che le variabili π_i sono una distribuzione di probabilità e quindi sono soggette al vincolo:

$$\sum_{i\in\{0,\dots,N-1\}} \pi_i = 1 \ .$$

Utilizziamo quindi il metodo dei moltiplicatori di Lagrange. Otteniamo, per ogni $i \in \{0,\dots,N-1\}$:

$$\frac{\partial}{\partial \pi_i} \left[\left(\sum_{j\in\{0,\dots,N-1\}} \log\left(\pi_j\right) \gamma_1^n(j) \right) + \mu \left(\sum_{i\in\{0,\dots,N-1\}} \pi_i - 1 \right) \right] = \frac{\gamma_1^n(i)}{\pi_i} + \mu \ .$$

La funzione si annulla nel punto:

$$\hat{\pi}_i = -\frac{\gamma_1^n(i)}{\mu} \ .$$

Sapendo che $\sum_{i\in\{0,\dots,N-1\}} \gamma_1^n(i) = 1$ otteniamo che $\mu = -1$ e quindi:

$$\hat{\pi}_i = \gamma_1^n(i) \ .$$

Calcoliamo ora $\frac{\partial}{\partial a_{ij}} Q(\lambda_n, \lambda)$. Osserviamo che le variabili a_{ij} sono soggette al vincolo:

$$\sum_{j\in\{0,\dots,N-1\}} a_{ij} = 1 \ .$$

Utilizziamo il metodo dei moltiplicatori di Lagrange, ottenendo, per ogni $i, j \in \{0, \dots, N-1\}$:

$$\frac{\partial}{\partial a_{ij}} \left[\left(\sum_{t=2}^{T} \sum_{(h,k)\in\{0,\dots,N-1\}^2} \log\left(a_{hk}\right) \varepsilon_{t-1}^n(h,k) \right) + \mu \left(\sum_{k\in\{0,\dots,N-1\}} a_{hk} - 1 \right) \right] =$$
$$\left(\sum_{t=2}^{T} \frac{\varepsilon_{t-1}^n(i,j)}{a_{ij}} \right) + \mu = \frac{1}{a_{ij}} \left(\sum_{t=1}^{T-1} \varepsilon_t^n(i,j) \right) + \mu \ .$$

La funzione si annulla nel punto:

$$\hat{a}_{ij} = -\frac{\sum_{t=1}^{T-1} \varepsilon_t^n(i,j)}{\mu} \ .$$

Dato che la somma su j degli a_{ij} è uguale a 1, ne ricaviamo che:

$$1 = \sum_{j\in\{0,\dots,N-1\}} a_{ij} = \sum_{j\in\{0,\dots,N-1\}} -\frac{\sum_{t=1}^{T-1} \varepsilon_t^n(i,j)}{\mu} =$$
$$-\frac{1}{\mu} \sum_{t=1}^{T-1} \sum_{j\in\{0,\dots,N-1\}} \mathbb{P}\left[X_t = i, X_{t+1} = j \mid Y = y, \lambda_n\right] =$$
$$-\frac{1}{\mu} \sum_{t=1}^{T-1} \mathbb{P}\left[X_t = i \mid Y = y, \lambda_n\right] = -\frac{1}{\mu} \sum_{t=1}^{T-1} \gamma_t^n(i) \ .$$

Da questa relazione segue che:

$$\mu = -\sum_{t=1}^{T-1} \gamma_t^n(i)$$

e quindi:

$$\hat{a}_{ij} = \frac{\sum_{t=1}^{T-1} \varepsilon_t^n(i,j)}{\sum_{t=1}^{T-1} \gamma_t^n(i)} \ .$$

Infine, calcoliamo $\frac{\partial}{\partial b_{i,y_t}} Q(\lambda_n, \lambda)$. Osserviamo che anche le variabili b_{i,y_t} sono soggette ad un vincolo. Per ogni $i \in \{0, \ldots, N-1\}$, infatti, vale:

$$\sum_{y_t \in \{0,\ldots,M-1\}} b_{i,y_t} = 1 \ .$$

Utilizziamo quindi ancora il metodo dei moltiplicatori di Lagrange ed otteniamo, per ogni $i \in \{0, \ldots, N-1\}$ e per ogni $y_t \in \{0, \ldots, M-1\}$:

$$\frac{\partial}{\partial b_{i,y_t}} \left[\sum_{u=1}^{T} \sum_{j \in \{0,\ldots,N-1\}} \log\left(b_{j,y_u}\right) \gamma_u^n(j) + \mu \left(\sum_{y_u \in \{0,\ldots,M-1\}} b_{j,y_u} - 1 \right) \right] =$$

$$\sum_{u \in \{1,\ldots,T\}: y_u = y_t} \frac{\gamma_u^n(i)}{b_{i,y_t}} + \mu = \frac{1}{b_{i,y_t}} \left(\sum_{u \in \{1,\ldots,T\}: y_u = y_t} \gamma_u^n(i) \right) + \mu \ .$$

La funzione si annulla nel punto:

$$\hat{b}_{i,y_t} = -\frac{\sum_{u \in \{1,\ldots,T\}: y_u = y_t} \gamma_u^n(i)}{\mu} \ .$$

Sostituendo il valore di $\hat{b}_{i,y_t}$ nell'equazione che esprime il vincolo a cui questi coefficienti sono soggetti, otteniamo:

$$1 = \sum_{y_t \in \{0,\ldots,M-1\}} b_{i,y_t} = \sum_{y_t \in \{0,\ldots,M-1\}} -\frac{\sum_{u \in \{1,\ldots,T\}: y_u = y_t} \gamma_u^n(i)}{\mu} =$$

$$-\frac{1}{\mu} \sum_{y_t \in \{0,\ldots,M-1\}} \sum_{u \in \{1,\ldots,T\}: y_u = y_t} \gamma_u^n(i) = -\frac{1}{\mu} \sum_{t \in \{1,\ldots,T\}} \gamma_t^n(i)$$

e quindi:

$$\mu = -\sum_{t \in \{1,\ldots,T\}} \gamma_t^n(i) \ .$$

Pertanto si ha che:

$$\hat{b}_{i,y_t} = \frac{\sum_{u \in \{1,\ldots,T\}: y_u = y_t} \gamma_u^n(i)}{\sum_{t \in \{1,\ldots,T\}} \gamma_t^n(i)} \ .$$

Dato che la funzione $Q(\lambda_n, \lambda)$ è una somma di logaritmi e la funzione $\log(x)$ è strettamente concava, il punto $\hat{\lambda}$ così trovato è necessariamente un punto di massimo locale.

Quindi per ogni $n \geq 0$ si ha che:

$$Q(\lambda_n, \lambda_{n+1}) > Q(\lambda_n, \lambda_n) \ .$$

Per la proposizione 2.7 si ha:

$$\mathrm{L}(y \mid \lambda_{n+1}) > \mathrm{L}(y \mid \lambda_n) \ .$$

e la successione $\{\lambda_n\}_{n\geq 0}$ converge ad un punto di massimo della verosimiglianza. □

L'algoritmo di Baum Welch, ad ogni passo, aggiusta i coefficienti del modello secondo un preciso significato statistico.

Infatti, consideriamo ad esempio a_{ij}^n. Esso può essere scritto come:

$$a_{ij}^n = \frac{\sum_{t=1}^{T-1} \varepsilon_t^{n-1}(i,j)}{\sum_{t=1}^{T-1} \gamma_t^{n-1}(i)} = \frac{\sum_{t=1}^{T-1} \mathbb{P}[X_t = i, X_{t+1} = j \mid Y = y, \lambda_{n-1}]}{\sum_{t=1}^{T-1} \mathbb{P}[X_t = i \mid Y = y, \lambda_{n-1}]} \ .$$

Per ogni $t \in \{1, \ldots, T-1\}$ l'elemento $\varepsilon_t^{n-1}(i,j)$ può essere visto come il valore atteso di una variabile casuale di Bernoulli B_t che assume valore 1 se al tempo t vi è stata una transizione da i a j e assume valore 0 altrimenti.

La somma $\sum_{t=1}^{T-1} B_t$ è una variabile casuale binomiale che rappresenta il numero di transizioni che sono avvenute, nell'intervallo temporale $\{1, \ldots, T-1\}$, da i a j.

Quindi possiamo affermare che

$$\sum_{t=1}^{T-1} \varepsilon_t^{n-1}(i,j) = \text{numero atteso di transizioni da } i \text{ a } j$$

Possiamo ripetere il ragionamento per tutti gli altri coefficienti, ottenendo il seguente risultato.

Osservazione: Per ogni $n > 0$, l'algoritmo di Baum Welch al passo n-esimo modifica i coefficienti del modello λ_{n-1} nel modo seguente.

$$\pi_{n,i} = \text{numero atteso di visite allo stato } i \text{ al tempo } t = 1$$

$$a_{n,ij} = \frac{\text{numero atteso di transizioni da } i \text{ a } j}{\text{numero atteso di transizioni da } i}$$

$$b_{n,i,y_k} = \frac{\text{numero atteso di visite allo stato } i \text{ quando si osserva } y_k}{\text{numero atteso di visite allo stato } i}$$

2.6 HMM scaling

I coefficienti α_t tendono a zero per t che tende a infinito. In caso di T elevato vi possono perciò essere problemi nella procedura di stima in massima verosimiglianza di $\lambda = (A, B, \pi)$. L'algoritmo di Baum Welch, infatti, coinvolge i coefficienti α_t. È possibile risolvere questo problema rinormalizzando ogni volta i coefficienti. Questa procedura è detta HMM scaling.

Definizione 2.4 Coefficienti α aggiustati.
Sia $i \in \{0, \ldots, N-1\}$. Per $t = 1$ definiamo le seguenti quantità.

$$\tilde{\alpha}_1(i) = \alpha_1(i)$$

$$c_1 = \frac{1}{\sum_{j=0}^{N-1} \tilde{\alpha}_1(j)}$$

$$\hat{\alpha}_1(i) = c_1 \tilde{\alpha}_1(i) \ .$$

Per ogni $t \in \{2, \ldots, T\}$ definiamo le seguenti quantità:

$$\tilde{\alpha_t}(i) = \left(\sum_{j=0}^{N-1} \hat{\alpha}_{t-1}(j) a_{ji} \right) b_{i,y_t}$$

$$c_t = \frac{1}{\sum_{j=0}^{N-1} \tilde{\alpha}_t(j)}$$

$$\hat{\alpha}_t(i) = c_t \tilde{\alpha}_t(i) \ .$$

Allora vale la seguente proposizione.

Proposizione 2.10 *Per ogni $t \in \{1, \ldots, T\}$ e per ogni $i \in \{0, \ldots, N-1\}$ si ha:*

$$\hat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)} \ .$$

Dimostrazione: Per prima cosa osserviamo che, per ogni $t \in \{1, \ldots, T\}$ e per ogni $i \in \{0, \ldots, N-1\}$, vale:

$$\hat{\alpha}_t(i) = \alpha_t(i) \prod_{j=1}^{t} c_j \ .$$

Dimostriamo questo fatto per induzione su t.

Sia $t = 1$. Allora per definizione di $\hat{\alpha}$ si ha:

$$\hat{\alpha}_1(i) = c_1 \tilde{\alpha}_1(i) = c_1 \alpha_1(i) \ .$$

Supponiamo che la tesi sia vera per t. Allora, applicando l'ipotesi induttiva, si ha:

$$\hat{\alpha}_{t+1}(i) = c_{t+1}\tilde{\alpha}_{t+1}(i) = c_{t+1}\left(\sum_{j=0}^{N-1}\hat{\alpha}_t(j)a_{ji}\right)b_{i,y_t} =$$

$$c_{t+1}\left(\sum_{j=0}^{N-1}\alpha_t(j)\left(\prod_{h=1}^{t}c_h\right)a_{ji}\right)b_{i,y_t} = \left(\prod_{h=1}^{t+1}c_h\right)\left(\sum_{j=1}^{N}\alpha_t(j)a_{ji}\right)b_{i,y_t} =$$

$$\left(\prod_{h=1}^{t+1}c_h\right)\alpha_{t+1}(i)\ .$$

Resta da dimostrare per per ogni $t \in \{1,\ldots,T-1\}$ vale:

$$\prod_{h=1}^{t}c_h = \frac{1}{\sum_{j=0}^{N-1}\alpha_t(j)}\ .$$

Dimostriamo questo fatto per induzione su t.

Se $t = 1$, per definizione si ha:

$$c_1 = \frac{1}{\sum_{j=0}^{N-1}\alpha_1(j)}\ .$$

Supponiamo che la tesi sia vera per t. Allora si ha:

$$\prod_{h=1}^{t+1}c_h = c_{t+1}\left(\prod_{h=1}^{t}c_h\right) = \frac{1}{\sum_{j=0}^{N-1}\tilde{\alpha}_{t+1}(j)}\left(\prod_{h=1}^{t}c_h\right) =$$

$$\frac{c_{t+1}}{\sum_{j=0}^{N-1}\hat{\alpha}_{t+1}(j)}\left(\prod_{h=1}^{t}c_h\right) =$$

$$\frac{c_{t+1}}{\sum_{j=0}^{N-1}\alpha_{t+1}(j)\left(\prod_{h=1}^{t+1}c_h\right)}\left(\prod_{h=1}^{t}c_h\right) = \frac{1}{\sum_{j=0}^{N-1}\alpha_{t+1}(j)}\ .$$

□

Definizione 2.5 Coefficienti β aggiustati
Sia $i \in \{0,\ldots,N-1\}$. Per $t = T$ definiamo le seguenti quantità.

$$\tilde{\beta}_T(i) = \beta_T(i)$$

$$c_T = \frac{1}{\sum_{j=0}^{N-1}\tilde{\beta}_T(j)}$$

$$\hat{\beta}_T(i) = c_T \tilde{\beta}_T(i) \ .$$

Per ogni $t \in \{1, \ldots, T-1\}$ *definiamo le seguenti quantità:*

$$\tilde{\beta}_t(i) = \sum_{j=0}^{N-1} a_{ij} b_{j,y_{t+1}} \hat{\beta}_{t+1}(j)$$

$$c_t = \frac{1}{\sum_{j=0}^{N-1} \tilde{\beta}_t(j)}$$

$$\hat{\beta}_t(i) = c_t \tilde{\beta}_t(i) \ .$$

Allora vale la seguente proposizione.

Proposizione 2.11 *Per ogni* $t \in \{1, \ldots, T\}$ *e per ogni* $i \in \{0, \ldots, N-1\}$ *si ha:*

$$\hat{\beta}_t(i) = \frac{\beta_t(i)}{\sum_{j=0}^{N-1} \beta_t(j)} \ .$$

Dimostrazione: Per prima cosa osserviamo che per ogni t e per ogni i vale:

$$\hat{\beta}_t(i) = \beta_t(i) \prod_{j=t}^{T} c_j \ .$$

Dimostriamo questo fatto per induzione su t.

Sia $t = T$. Allora per definizione di $\hat{\beta}$ si ha:

$$\hat{\beta}_T(i) = c_T \tilde{\beta}_T(i) = c_T \beta_T(i) \ .$$

Supponiamo che la tesi sia vera per $t+1$. Allora per t, applicando l'ipotesi induttiva, otteniamo:

$$\hat{\beta}_t(i) = c_t \tilde{\beta}_t(i) = c_t \sum_{j=0}^{N-1} a_{ij} b_{j,y_{t+1}} \hat{\beta}_{t+1}(j) = \prod_{h=t}^{T} c_h \sum_{j=0}^{N-1} a_{ij} b_{j,y_{t+1}} \beta_{t+1}(j) = \beta_t(i) \prod_{h=t}^{T} c_h \ .$$

Resta da dimostrare che per ogni t vale:

$$\prod_{h=t}^{T} c_h = \frac{1}{\sum_{j=0}^{N-1} \beta_t(j)} \ .$$

Dimostriamo questa tesi per induzione su t.

Sia $t = T$ allora per definizione di c_T abbiamo:

$$c_T = \frac{1}{\sum_{j=0}^{N-1} \tilde{\beta}_T(j)} = \frac{1}{\sum_{j=0}^{N-1} \beta_T(j)} \ .$$

Supponiamo che la tesi sia vera per $t+1$. Per t allora otteniamo:

$$\prod_{h=t}^{T} c_h = c_t \left(\prod_{h=t+1}^{T} c_h \right) = \frac{1}{\sum_{j=0}^{N-1} \tilde{\beta}_t(j)} \left(\prod_{h=t+1}^{T} c_h \right) =$$
$$\frac{c_t}{\sum_{j=0}^{N-1} \hat{\beta}_t(j)} \left(\prod_{h=t+1}^{T} c_h \right) =$$
$$\frac{c_t}{\sum_{j=0}^{N-1} \beta_t(j) \left(\prod_{h=t}^{T} c_h \right)} \left(\prod_{h=t+1}^{T} c_h \right) = \frac{1}{\sum_{j=0}^{N-1} \beta_t(j)} \ .$$

□

È possibile calcolare i γ e gli ε utilizzando i coefficienti aggiustati $\hat{\alpha}$ e $\hat{\beta}$, evitando errori di calcolo.

Proposizione 2.12 *Per ogni $t \in \{1, \ldots, T\}$ e per ogni i e $j \in \{0, \ldots, N-1\}$ valgono le seguenti uguaglianze:*

$$\gamma_t(i) = \frac{\hat{\alpha}_t(i)\hat{\beta}_t(i)}{\sum_{j=0}^{N-1} \hat{\alpha}_t(j)\hat{\beta}_t(j)}$$

e

$$\varepsilon_t(i,j) = \frac{\hat{\alpha}_t(i) a_{ij} b_{j,y_{t+1}} \hat{\beta}_{t+1}(j)}{\sum_{h=0}^{N-1} \sum_{k=0}^{N-1} \hat{\alpha}_t(h) a_{hk} b_{k,y_{t+1}} \hat{\beta}_{t+1}(k)} \ .$$

Dimostrazione: Ricordiamo che:

$$\hat{\alpha}_t(i) = \frac{\alpha_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)} \quad \text{e} \quad \hat{\beta}_t(i) = \frac{\beta_t(i)}{\sum_{j=0}^{N-1} \beta_t(j)} \ .$$

Sostituendo questi valori otteniamo:

$$\frac{\hat{\alpha}_t(i)\hat{\beta}_t(i)}{\sum_{j=0}^{N-1} \hat{\alpha}_t(j)\hat{\beta}_t(j)} = \frac{\frac{\alpha_t(i)\beta_t(i)}{\left(\sum_{h=0}^{N-1} \alpha_t(h)\right)\left(\sum_{h=0}^{N-1} \beta_t(h)\right)}}{\frac{\sum_{j=0}^{N-1} \alpha_t(j)\beta_t(j)}{\left(\sum_{h=0}^{N-1} \alpha_t(h)\right)\left(\sum_{h=0}^{N-1} \beta_t(h)\right)}} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{j=0}^{N-1} \alpha_t(j)\beta_t(j)} = \gamma_t(i)$$

e

$$\frac{\hat{\alpha}_t(i)a_{ij}b_{j,y_{t+1}}\hat{\beta}_{t+1}(j)}{\sum_{h=0}^{N-1}\sum_{k=0}^{N-1}\hat{\alpha}_t(h)a_{hk}b_{k,y_{t+1}}\hat{\beta}_{t+1}(k)} = \frac{\frac{\alpha_t(i)a_{ij}b_{j,y_{t+1}}\beta_{t+1}(j)}{\left(\sum_{m=0}^{N-1}\alpha_t(m)\right)\left(\sum_{m=0}^{N-1}\beta_{t+1}(m)\right)}}{\frac{\sum_{h=0}^{N-1}\sum_{k=0}^{N-1}\alpha_t(h)a_{hk}b_{k,y_{t+1}}\beta_{t+1}(k)}{\left(\sum_{m=0}^{N-1}\alpha_t(m)\right)\left(\sum_{m=0}^{N-1}\beta_{t+1}(m)\right)}} =$$

$$\frac{\alpha_t(i)a_{ij}b_{j,y_{t+1}}\beta_{t+1}(j)}{\sum_{h=0}^{N-1}\sum_{k=0}^{N-1}\alpha_t(h)a_{hk}b_{k,y_{t+1}}\beta_{t+1}(k)} = \varepsilon_t(i,j) \ .$$

□

È possibile inoltre utilizzare i coefficienti aggiustati per calcolare la verosimiglianza del modello. Vale la seguente proposizione.

Proposizione 2.13 *Consideriamo i coefficienti c_t utilizzati per la stima dei coefficienti α aggiustati, definiti come:*

$$c_t = \frac{1}{\sum_{j=0}^{N-1}\tilde{\alpha}_t(j)} \quad \forall t \in \{1,\dots,T\} \ .$$

Allora la log-verosimiglianza del modello di Markov nascosto può essere scritta come:

$$l(\lambda \mid Y = y) = \log\left(\mathbb{P}\left[Y = y \mid A, B, \pi\right]\right) = -\sum_{h=1}^{T}\log(c_h) \ .$$

Dimostrazione: Ricordiamo che:

$$\mathbb{P}\left[Y = y \mid A, B, \pi\right] = \sum_{j=0}^{N-1}\alpha_T(j) \ .$$

Sappiamo inoltre che, per ogni $i \in \{0,\dots,N-1\}$, vale:

$$\hat{\alpha}_T(i) = \alpha_T(i)\prod_{h=1}^{T}c_h \ .$$

Combinando le due espressioni otteniamo:

$$\mathbb{P}\left[Y = y \mid A, B, \pi\right] = \sum_{j=0}^{N-1}\left(\frac{\hat{\alpha}_T(j)}{\prod_{h=1}^{T}c_h}\right) = \frac{1}{\prod_{h=1}^{T}c_h}\sum_{j=0}^{N-1}\hat{\alpha}_T(j) = \frac{1}{\prod_{h=1}^{T}c_h} \ .$$

E quindi, per le proprietà del logaritmo:

$$\log\left(\mathbb{P}\left[Y = y \mid A, B, \pi\right]\right) = -\sum_{h=1}^{T}\log(c_h) \ .$$

□

2.7 Riferimenti bibliografici

Le tre definizioni di Hidden Markov Models e la proposizione 2.1 sono tratte dalla tesi di laurea di Marco Ruzza *Inferenza Statistica per Hidden Markov Models* [11].

I paragrafi 2.1 e 2.6, relativi alle procedure informatiche implementabili per la stima di un Hidden Markov Model, sono stati tratti dal testo di Mark Stamp *A Revealing Introduction to Hidden Markov Models* [12].

Il paragrafo 2.1, che presenta l'algoritmo di Baum Welch e ne dimostra la convergenza, è interamente tratto dal testo di Leonard Baum A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains [1]. Le dimostrazioni, presentate da Baum nel caso di processi di Markov continui, sono state riadattate al caso di Hidden Markov Models discreti.

La proposizione 2.9, che trova la forma esplicita della funzione τ dell'algoritmo di Baum Welch nel caso di Hidden Markov Models discreti, è originale.

I tre problemi relativi agli Hidden Markov Models, che sono descritti nella definizione 2.2, infine, sono tratti dal testo di Lawrence Rabiner *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition* [10].

Capitolo 3

Tabelle di contingenza

3.1 Introduzione

Consideriamo un insieme finito Γ di variabili aleatorie discrete, dette anche criteri di classificazione. Per ogni $\delta \in \Gamma$ consideriamo un insieme finito I_δ che contiene i possibili valori che possono essere assunti da δ. Chiamiamo I il prodotto cartesiano di tutti gli insiemi I_δ, cioè:

$$I = \times_{\delta \in \Gamma} I_\delta \ .$$

Immaginiamo di estrarre un campione di numerosità $\mid n \mid$ da una popolazione che assume valori in I. Esso può essere rappresentato in due diversi modi. O con un vettore di lunghezza $\mid n \mid$

$$(i^1, \ldots, i^{|n|})$$

dove ciascuna entrata del vettore rappresenta il valore che quell'elemento assume, cioè è un elemento di I. Oppure può essere rappresentato con una *tabella dei conteggi*:

$$n = \{n(i)\}_{i \in I}$$

dove $n(i)$ rappresenta il numero di volte che è stato osservato il valore $i \in I$. La dimensione della tabella sarà quindi uguale alla cardinalità dell'insieme I. In questo caso $\mid n \mid = \sum_{i \in I} n(i)$.

3.2 Tabelle marginali

Sia $A \subseteq \Gamma$. Si definisce *tabella marginale* relativa ad A la tabella dei conteggi relativa ai soli criteri di classificazione che sono contenuti in A, ovvero a quelli contenuti nell'insieme:

$$I_A = \times_{\alpha \in A} I_\alpha \ .$$

Consideriamo un arbitrario vettore $x = \{x(i)\}_{i \in I}$. Dato un generico $i_A \in I_A$, definiamo:

$$x(i_A) = \sum_{j : j_A = i_A} x(j) = \sum_{j_{\Gamma \setminus A} \in I_{\Gamma \setminus A}} x(i_A, j_{\Gamma \setminus A}) \ .$$

Quindi il vettore marginale di x rispetto ad A sarà:

$$\{x(i_A)\}_{i_A \in I_A} \ .$$

Si definisce *tabella dei conteggi marginali* rispetto a A la tabella:

$$n_A = \{n(i_A)\}_{i_A \in I_A} \ .$$

Se $B \subset \Gamma$ l'elemento $i_B \in I_B$ determina una sotto-tabella di contingenza. Questa sotto-tabella è la tabella condizionata al valore i_B. Se $A = \Gamma \setminus B$, essa ha celle appartenenti a I_A e conteggi $n^{i_B}(i_A) = n(i_A, i_B) = n(i)$.

Definiamo $\mathbb{R}^I$ lo spazio delle funzioni su I a valori reali. Su di esso è definito il prodotto scalare:

$$< x, y >= \sum_{i \in I} x(i) y(i) \quad \forall x, y \in \mathbb{R}^I \ .$$

Sia A un sottoinsieme di Γ. Esso determina un sottospazio di $\mathbb{R}^I$, chiamato F_A, che è lo spazio delle funzioni su I a valori in $\mathbb{R}$ che dipendono solo da i_A, cioè:

$$x \in F_A \Leftrightarrow x(i) = x(j) \quad \forall \ i, j : i_A = j_A \ . \tag{3.1}$$

Quindi, ad esempio, $F_\Gamma = \mathbb{R}^I$ e $F_\emptyset$ è lo spazio delle funzioni costanti.

Dato un insieme $A \in \Gamma$, la proiezione ortogonale di un vettore x sullo spazio F_A è definita come:

$$\pi_A x(i) = \frac{x(i_A)}{\mid I_{\Gamma \setminus A} \mid} \ . \tag{3.2}$$

3.3 Distribuzioni di probabilità

Consideriamo una tabella dei conteggi $\{n(i)\}_{i \in I}$. Si assume che essa sia la realizzazione campionaria di una variabile aleatoria $\{N(i)\}_{i \in I}$.

In letteratura ricorrono tre tipi di ipotesi che possono essere formulate sulla distribuzione dei conteggi di una tabella di contingenza $\{N(i)\}_{i \in I}$, che portano ad adottare tre differenti modelli statistici. Queste ipotesi trattano solo il modo in cui sono stati campionati i dati e, pertanto, non riguardano la popolazione da cui il campione è stato estratto.

1. Si assume che i conteggi e il numero totale di osservazioni siano casuali.
2. Si assume che i conteggi siano casuali ma che il numero totale di osservazioni sia fissato.
3. Si assume che esista un sottoinsieme $B \subset \Gamma$ per il quale $n(i_B)$ sia fissato per ciascun $i_B \in I_B$.

Nel caso 1) si assume che i conteggi delle celle siano realizzazioni di variabili indipendenti con una distribuzione di Poisson, ovvero per ogni $i \in I$ il conteggio $n(i)$ è una realizzazione di una variabile aleatoria:

$$N(i) \sim \text{Poisson}(\lambda(i)) \ .$$

Pertanto la distribuzione congiunta di tutti i conteggi è la seguente:

$$\mathbb{P}\left[N(i) = n(i), i \in I\right] = \prod_{i \in I} \frac{e^{-\lambda(i)} \lambda(i)^{n(i)}}{n(i)!} \ .$$

Nel caso 2) si assume che il numero totale dei conteggi sia fissato e uguale a $\mid n \mid$ e si assume che ciascuna osservazione sia assegnata alla cella $i \in I$ con una probabilità fissata $p(i) \geq 0$. Pertanto i conteggi seguono una distribuzione multinomiale:

$$\mathbb{P}\left[N(i) = n(i), i \in (I)\right] = \frac{\mid n \mid!}{\prod_{i \in I} n(i)!} \prod_{i \in I} p(i)^{n(i)} \ .$$

Nel caso 3), supponiamo che esista un sottonsieme B di Γ tale che per ogni $i_B \in I_B$ sia fissato il conteggio $n(i_B)$. Sia $A = \Gamma \backslash B$. Assumiamo che in ciascuna sotto-tabella determinata da i_B i conteggi siano indipendenti e distribuiti come una multinomiale dove la probabilità di ciascuna cella $i = (i_A, i_B)$ è $p(i_A \mid i_B)$. Pertanto la distribuzione congiunta dei conteggi è la seguente:

$$\mathbb{P}\left[N(i) = n(i), i \in (I)\right] = \prod_{i_B \in I_B} \left[\frac{n(i_B)!}{\prod_{i_A \in I_A} n^{i_B}(i_A)!} \prod_{i_A \in I_A} p(i_A \mid i_B)^{n^{i_B}(i_A)} \right] \ .$$

I tre tipi di ipotesi sulla distribuzione dei conteggi sono legati tra loro da risultati riportati nella proposizione seguente.

Proposizione 3.1 *1. La distribuzione multinomiale adottata nel caso 2) è equivalente alla distribuzione di Poisson del caso 1) condizionata a* $\sum_{i \in I} N(i) = \mid n \mid$, *assumendo che* $p(i) = \frac{\lambda(i)}{\sum_{j \in I} \lambda(j)}$

2. La distribuzione adottata nel caso 3) è equivalente sia alla distribuzione di Poisson del caso 1) sia alla distribuzione multinomiale del caso 2), in entrambi i casi condizionata a $N(i_B) = n(i_B) \forall \ i_B \in I_B$.

Dimostrazione: 1. La dimostrazione del primo punto è omessa poichè immediata.

2. Supponiamo che i conteggi seguano una distribuzione di Poisson, cioè $N(i) \sim \text{Poisson}(\lambda(i))\ \forall i \in I$, e condizioniamo a $N(i_B) = n(i_B)\ \forall i_B \in I_B$. Sia $N = \sum_{i_B \in I_B} n(i_B)$. Dato che la somma di variabili aleatorie con distribuzione di Poisson ha ancora distribuzione di Poisson, per ciascun $i_B \in I_B$ si ha:

$$N(i_B) = \sum_{j:j_B=i_B} N(j) \sim \text{Poisson}\left(\sum_{j:j_B=i_B} \lambda(j)\right) .$$

La distribuzione congiunta dei conteggi è dunque:

$$\mathbb{P}\left[N(i) = n(i)\ \forall i \in I \mid N(i_B) = n(i_B)\ \forall i_B \in I_B\right] = \frac{\mathbb{P}\left[N(i) = n(i) \forall i \in I, \sum_{j:j_B=i_B} N(j) = n(i_B)\ \forall i_B \in I_B\right]}{\mathbb{P}\left[\sum_{j:j_B=i_B} N(j) = n(i_B)\ \forall i_B \in I_B\right]} =$$

assumiamo che per ogni $i_B \in I_B$ valga $\sum_{j:j_B=i_B} n(j) = n(i_B)$ (altrimenti la probabilità sarebbe nulla) e proseguiamo il calcolo:

$$= \frac{\mathbb{P}\left[N(i) = n(i) \forall i \in I\right]}{\mathbb{P}\left[\sum_{j:j_B=i_B} N(j) = n(i_B) \forall i_B \in I_B\right]} =$$

$$\frac{\prod_{i\in I} \frac{e^{-\lambda(i)}\lambda(i)^{n(i)}}{n(i)!}}{\prod_{i_B\in I_B} \frac{e^{\sum_{j:j_B=i_B}\lambda(j)}\left(\sum_{i:i(B)=i_B}\lambda(i)\right)^{n(i_B)}}{n(i_B)!}} = \frac{\prod_{i_B\in I_B}\left(\prod_{j:j_B=i_B}\frac{\lambda(j)^{n(j)}}{n(j)!}\right)}{\prod_{i_B\in I_B}\frac{\left(\sum_{j:j_B=i_B}\lambda(j)\right)^{n(i_B)}}{n(i_B)!}} =$$

$$\prod_{i_B\in I_B}\left[\frac{n(i_B)!}{\prod_{j:j_B=i_B} n(j)!}\prod_{j:j_B=i_B}\left(\frac{\lambda(j)}{\sum_{k:k_B=i:B}\lambda(k)}\right)^{n(j)}\right] =$$

$$\prod_{i_B\in I_B}\left[\frac{n(i_B)!}{\prod_{i_A\in I_A} n^{i_B}(i_A)!}\prod_{i_A\in I_A} p(i_A \mid i_B)^{n^{i_B}(i_A)}\right] .$$

L'ultimo passaggio è stato ottenuto ponendo, nel caso in cui $i = (i_A, i_B)$,

$$p(i_A \mid i_B) = \frac{\lambda(i)}{\sum_{j\in I:j_B=i_B}\lambda(j)} .$$

La seconda parte della dimostrazione si ottiene immediatamente, osservando che, se $\mid n \mid = \sum_{i_B \in I_B} n(i_B)$, si ha:

$$\mathbb{P}\left[N(i) = n(i) \forall i \in I \mid N(i_B) = n(i_B) \forall i_B \in I_B\right] = \mathbb{P}\left[N(i) = n(i) \forall i \in I \mid \sum_{i \in I} N(i) = \mid n \mid, N(i_B) = n(i_B) \forall i_B \in I_B\right] .$$

□

3.4 Modelli log-lineari

Consideriamo una tabella di contingenza $n = \{n(i)\}_{i \in I}$, realizzazione di una variabile casuale $N = \{N(i)\}_{i \in I}$, dove $\mid I \mid = q$. La tabella si può pensare come un elemento dello spazio vettoriale q-dimensionale $\mathbb{R}^I$, che è lo spazio delle funzioni definite su I a valori reali.

Si assume che il vettore N abbia un valore atteso $m = \{m(i)\}_{i \in I}$ tale che: $m(i) > 0 \ \forall \ i \in I$.

Il valore di m è però sconosciuto. L'obiettivo dei prossimi paragrafi sarà quindi trovare una buona stima di m.

Per prima cosa osserviamo che, a seconda della distribuzione scelta per la tabella dei conteggi, occorre fare su m ipotesi aggiuntive.

- Se la distribuzione scelta è 1), cioè la distribuzione di Poisson, allora vale che:
$$m(i) = \lambda(i) \ \forall i \in I$$
e non vi sono restrizioni aggiuntive.

- Se la distribuzione scelta è 2), si hanno invece i vincoli:
$$m(i) = \mid n \mid p(i), \quad \sum_{i \in I} m(i) = \sum_{i \in I} n(i) = \mid n \mid \ .$$

- Se la distribuzione scelta è il 3), infine, valgono i vincoli:
$$m(i) = n(i_B) p(i_A \mid i_B), \quad m(i_B) = n(i_B) \ \forall i_B \in I_B \ .$$

Oltre a queste ipotesi, che sono dettate dalla distribuzione della tabella dei conteggi, spesso è utile assumere che il vettore m appartenga ad un particolare sottospazio di $\mathbb{R}^I$. Vengono poste delle ipotesi restrittive su di esso.

In particolare si assume che il vettore m abbia la seguente forma:

$$m = k \exp(\mu)$$

dove:
k è una costante non identicamente nulla.
$\mu \in \mathbb{H}$ e $\mathbb{H}$ è un sottospazio lineare di $\mathbb{R}^I$ di dimensione p, con $0 < p \leq q$.

Se per un certo indice i fosse $k(i) = 0$, il valore medio su quella cella sarebbe fissato nullo. Saremmo quindi in presenza di uno zero strutturale. In un caso del genere è sufficiente considerare una sottotabella di contingenza dove la cella i è stata eliminata. Per questo motivo considereremo solo il caso di $k(i) > 0$ $\forall i \in I$.

Si ha quindi che $\log(k) + \mathbb{H}$ è uno spazio affine contenuto in $\mathbb{R}^I$.

Denotiamo con $M(k, \mathbb{H})$ l'insieme dei vettori media $\{m(i)\}_{i \in I}$ della forma $m = k \exp(\mu)$ con $\mu \in \mathbb{H}$ tali che verifichino le ipotesi aggiuntive della distribuzione scelta per la tabella dei conteggi.

Definizione 3.1 *Un modello in cui il vettore delle medie* $\{m(i)\}_{i \in I}$ *appartiene allo spazio* $M(k, \mathbb{H})$ *è detto* modello log-affine.

Si utilizza questo nome perché, se $k(i) \geq 0 \ \forall \ i \in I$ vale:

$$m \in M(k, \mathbb{H}) \Leftrightarrow \log(m) \in \log(k) + \mathbb{H} \ .$$

Ma che significato può avere porre tali restrizioni su m? Ricordiamo che, nel caso in cui si assuma l'indipendenza condizionata tra alcune variabili, la loro distribuzione congiunta può essere scritta come prodotto di fattori che dipendono solo da alcune variabili. Quindi il suo logaritmo può essere scritto come somma di addendi dipendenti solo da alcune variabili e appartiene ad un certo sottospazio vettoriale. Vediamo un esempio.

Esempio: Consideriamo una tabella di contingenza bidimensionale di dimensione $r \times c$ tale che la probabilità della cella (i, j), con $1 \leq i \leq r$ e $1 \leq j \leq c$, sia $p_{i,j} \geq 0$.

Sia $I = \{1, 2, \ldots, r\} \times \{1, 2, \ldots, c\}$. Supponiamo di osservare un campione di taglia $\mid n \mid$. I valori attesi saranno:

$$m = \{\mid n \mid p_{i,j}\}_{(i,j) \in I} \ .$$

Il vettore $\log(m)$ appartiene quindi, senza ipotesi aggiuntive, a $\mathbb{R}^{rc}$.

Supponiamo invece che le variabili che rappresentano righe e colonne siano indipendenti, allora vale:

$$p_{i,j} = p_{i,+} p_{+,j}$$

dove:

$$p_{i,+} = \sum_{j=1}^{c} p_{i,j} \quad p_{+,j} = \sum_{i=1}^{r} p_{i,j} \ .$$

Quindi possiamo scrivere:

$$\log(m_{i,j}) = \log\left(\mid n \mid p_{i,+}p_{+,j}\right) = \alpha + \beta_i + \gamma_j \ .$$

Quindi:

$$m_{i,j} = e^{\alpha}e^{\beta_i+\gamma_j} \ .$$

Ciò equivale a cercare il vettore m in un spazio affine definito come $e^{\alpha} + \mathbb{H}$, dove $\mathbb{H}$ è un sottospazio vettoriale di $\mathbb{R}^{rc}$ che ha dimensione $r + c - 1$.

Vedremo in seguito come calcolare le dimensioni degli spazi vettoriali del modello.

Sia $\overline{M}(k, \mathbb{H})$ l'insieme dei limiti delle successioni definite su $M(k, \mathbb{H})$.

Definizione 3.2 *Un modello in cui si assume che*

$$m \in \overline{M}(k, \mathbb{H})$$

è detto modello log-affine esteso.

Ricercare il vettore delle medie m all'interno dello spazio esteso $\overline{M}(k, \mathbb{H})$ significa semplicemente accettare che esso possa avere alcune componenti nulle.

Vedremo in seguito che l'utilizzo di modelli estesi garantisce spesso l'esistenza dello stimatore di massima verosimiglianza.

3.5 Forma della verosimiglianza

Se la distribuzione scelta per la tabella dei conteggi è 2) o 3), si ha che:

$$\prod_{i \in I} e^{m(i)} = e^{|n|}$$

cioè è costante. Pertanto, qualunque delle tre distribuzioni si utilizzi, è immediato dimostrare che la verosimiglianza del modello, scritta come funzione del vettore $\{m(i)\}_{i \in I}$ risulta proporzionale a:

$$\mathrm{L}(m) \propto \prod_{i \in I} m(i)^{n(i)} e^{-m(i)} \ .$$

Dato che la verosimiglianza è proporzionale alla stessa quantità, la stima di massima verosimiglianza è la stessa indipendentemente dalla distribuzione della tabella dei conteggi. Pertanto, per il calcolo della stima di massima verosimiglianza, nel seguito utilizzeremo la distribuzione di Poisson, che non possiede vincoli aggiuntivi. Distingueremo invece tra le varie distribuzioni nello studio della distribuzione dello stimatore.

Se assumiamo che valga un modello log-affine $M(k, \mathbb{H})$ e che quindi $m = ke^{\mu}$ con $\mu \in \mathbb{H}$, possiamo scrivere la log-verosimiglianza in funzione di μ ed otteniamo:

$$\mathrm{l}(\mu) =< n, \mu > - \sum_{i \in I} k(i) e^{\mu(i)} \ .$$

Infatti k è una costante e quindi il termine $< n, \log(k) >$ può essere escluso dalla verosimiglianza.

Se $\mathbb{H}$ è un sottospazio di $\mathbb{R}^I$ denotiamo con $\pi_{\mathbb{H}}$ la proiezione ortogonale su $\mathbb{H}$ secondo il prodotto scalare.

Dato che $\mu \in \mathbb{H}$, si ha che:

$$\mu = \pi_{\mathbb{H}} \mu$$

e inoltre vale

$$< n, \pi_{\mathbb{H}} \mu >=< \mu, \pi_{\mathbb{H}} n > \ .$$

Di conseguenza la log-verosimiglianza in funzione di μ può essere scritta come:

$$\mathrm{l}(\mu) =< \pi_{\mathbb{H}} n, \mu > - \sum_{i \in I} k(i) \exp\left(\mu(i)\right) \ .$$

Quindi il modello appartiene a una famiglia esponenziale. Dato che $\mathbb{H}$ è isomorfo a $\mathbb{R}^p$, si ha subito che $\pi_{\mathbb{H}} N$ è una statistica sufficiente, minimale e completa per μ.

3.6 Il modello saturo

Definizione 3.3 *Un modello log-affine esteso* $\overline{M}(k, \mathbb{H})$ *dove vale*

$$\mathbb{H} = \mathbb{R}^I$$

è detto modello saturo.

Osserviamo che la scelta del vettore k, purché positivo, è indifferente in quanto si sta considerando l'intero spazio.

Il modello saturo è il modello più generale possibile, dove non si assume nessuna restrizione sulla stima se non quelle imposte dalla distribuzione della tabella dei conteggi.

Proposizione 3.2 *In un modello saturo si ha che lo stimatore di massima verosimiglianza per il vettore delle medie è:*

$$\hat{M}(i) = N(i) \quad \forall i \in I \ .$$

Dimostrazione: Dato che, qualunque dei tre modelli si utilizzi, la verosimiglianza, scritta come funzione del vettore $\{m(i)\}_{i\in I}$ risulta proporzionale a:

$$\mathrm{L}(m) \propto \prod_{i\in I} m(i)^{N(i)} e^{-m(i)}$$

applicando il lemma A.2 si ottiene quindi subito che:

$$\mathrm{L}(m) \leq \mathrm{L}(N)$$

e vale l'uguaglianza se e solo se $m(i) = N(i)\ \forall i \in I$.

Quindi il vettore $\{N(i)\}_{i\in I}$ è lo stimatore di massima verosimiglianza. □

3.7 Stima di massima verosimiglianza

Passiamo ora a trattare la stima di massima verosimiglianza nel caso generale.

L'obiettivo è trovare un elemento $\hat{\mu} \in \mathbb{H}$ e un corrispondente elemento

$$\hat{m} = k \exp(\hat{\mu})$$

tale che valga:

$$\mathrm{l}(\hat{\mu}) = \sup_{\mu\in\mathbb{H}} \mathrm{l}(\mu)\ .$$

Vale la seguente proposizione:

Proposizione 3.3 *Consideriamo un modello log-affine esteso $\overline{M}(k, \mathbb{H})$. Allora, se esiste lo stimatore di massima verosimiglianza $\hat{M}$ del vettore delle medie esso è unico e soddisfa il sistema di equazioni di verosimiglianza:*

$$\pi_{\mathbb{H}}\hat{M} = \pi_{\mathbb{H}}N\ .$$

Viceversa, se un vettore $\hat{M} \in \overline{M}(k, \mathbb{H})$ soddisfa il sistema di equazioni di verosimiglianza, allora $\hat{M}$ è lo stimatore di massima verosimiglianza.

Dimostrazione: Per prima cosa dimostriamo che, se esiste lo stimatore di massima verosimiglianza, esso è unico.

Supponiamo che esistano due stimatori di massima verosimiglianza, M_1 e M_2. Definiamo:

$$\tilde{M} = \sqrt{M_1 M_2}$$

e:

$$\mid M_1 \mid = \sum_{i\in I} M_1(i) \quad \mid M_2 \mid = \sum_{i\in I} M_2(i) \quad \mid \tilde{M} \mid = \sum_{i\in I} \tilde{M}(i)\ .$$

Allora $\tilde{M} \in \overline{M}(k, \mathbb{H})$ e vale:

$$\mathrm{L}(\tilde{M}) \leq \mathrm{L}(M_1) = \sqrt{\mathrm{L}(M_1)\mathrm{L}(M_2)} = \prod_{i \in I} M_1(i)^{\frac{N(i)}{2}} M_2(i)^{\frac{N(i)}{2}} e^{-\frac{M_1(i)+M_2(i)}{2}} =$$

$$\prod_{i \in I} \left[(M_1(i)M_2(i))^{\frac{N(i)}{2}} e^{-\sqrt{M_1(i)M_2(i)}} \right] e^{-\frac{|M_1|+|M_2|}{2}} e^{|\tilde{M}|} = \mathrm{L}(\tilde{M}) e^{|\tilde{M}| - \frac{|M_1|+|M_2|}{2}} =$$

$$\mathrm{L}(\tilde{M}) \exp\left(-\frac{1}{2} \sum_{i \in I} \left(M_1(i)^{\frac{1}{2}} - M_2(i)^{\frac{1}{2}} \right)^2 \right) .$$

Poiché

$$\exp\left(-\frac{1}{2} \sum_{i \in I} \left(M_1(i)^{\frac{1}{2}} - M_2(i)^{\frac{1}{2}} \right)^2 \right) \leq 1$$

affinché la disuguaglianza sia vera deve essere:

$$\sum_{i \in I} \left(M_1(i)^{\frac{1}{2}} - M_2(i)^{\frac{1}{2}} \right)^2 = 0$$

e quindi

$$M_1 = M_2 \ .$$

Osserviamo che il sistema di equazioni di verosimiglianza è equivalente al seguente sistema di equazioni:

$$\sum_{i \in I} N(i)v(i) = \sum_{i \in I} \hat{M}(i)v(i) \quad \forall v \in \mathbb{H} \ .$$

Supponiamo che esista lo stimatore di massima verosimiglianza $\hat{M}$. Sia $v \in \mathbb{H}$. Per ogni $t \in \mathbb{R}$ definiamo:

$$M_t = \hat{M} \exp(tv)$$

allora $M_t \in \overline{M}(\mathbb{H}, k)$ e si ha:

$$h(t) = \log\left(\mathrm{L}(M_t)\right) =$$

$$\log\left(\mathrm{L}(\hat{M}) \prod_{i \in I} \exp\left(tv(i)N(i)\right) \exp\left(-\hat{M}(i) \exp(tv(i)) \right) \exp\left(\hat{M}(i) \right) \right) =$$

$$\log\left(\mathrm{L}(\hat{M}) \right) + t \sum_{i \in I} v(i)N(i) - \sum_{i \in I} \hat{M}(i) \left(1 - \exp\left(tv(i)\right) \right) \ .$$

La funzione $h(t)$ è differenziabile e assume il valore massimo per $t = 0$, infatti:

$$h(0) = \log\left(L(\hat{M}) \right) \ .$$

Quindi si ha:

$$0 = h'(0) = \sum_{i \in I} N(i)v(i) - \sum_{i \in I} \hat{M}(i)v(i) \ .$$

Quindi $\hat{M}$ soddisfa le equazioni di verosimiglianza.

Supponiamo ora che $\tilde{M}$ sia un elemento di $\overline{M}(\mathbb{H}, k)$ che soddisfa le equazioni di verosimiglianza, cioè

$$\pi_{\mathbb{H}} \tilde{M} = \pi_{\mathbb{H}} N \ .$$

Dato che $\tilde{M} \in \overline{M}(\mathbb{H}, k)$ esiste una successione $\{M_l\}_{l \in \mathbb{N}}$ con

$$M_l = k \exp(v_l) \quad v_l \in \mathbb{H}$$

tale che:

$$\tilde{M} = \lim_{l \to \infty} M_l \ .$$

Sia m un generico elemento di $\overline{M}(\mathbb{H}, k)$. Allora esiste un'altra successione $\{m_s\}_{s \in \mathbb{N}}$ con

$$m_s = k \exp(v_s) \quad v_s \in \mathbb{H}$$

tale che:

$$m = \lim_{s \to \infty} m_s \ .$$

Allora possiamo scrivere:

$$\mathrm{L}(m) = \lim_{s \to \infty} \mathrm{L}(m_s) = \lim_{s \to \infty} \prod_{i \in I} \left(k(i) \exp\left(v_s(i)\right)\right)^{N(i)} \exp\left(-m_s(i)\right) =$$

$$= \lim_{s \to \infty} \prod_{i \in I} m_s(i)^{\tilde{M}(i)} \exp\left(-m_s(i)\right) \prod_{i \in I} k(i)^{N(i) - \tilde{M}(i)} \exp\Big(v_s(i)N(i) - v_s(i)\tilde{M}(i)\Big) =$$

$$\lim_{s \to \infty} \prod_{i \in I} m_s(i)^{\tilde{M}(i)} \exp\left(-m_s(i)\right) \prod_{i \in I} k(i)^{N(i) - \tilde{M}(i)} \ .$$

Applicando il lemma A.2, riportato in appendice, si ottiene quindi:

$$\lim_{s\to\infty}\prod_{i\in I} m_s(i)^{\tilde{M}(i)}\exp\left(-m_s(i)\right)\prod_{i\in I}k(i)^{N(i)-\tilde{M}(i)} \leq$$
$$\prod_{i\in I}\tilde{M}(i)^{\tilde{M}(i)}\exp\left(-\tilde{M}(i)\right)\prod_{i\in I}k(i)^{N(i)-\tilde{M}(i)} =$$
$$\lim_{l\to\infty}\prod_{i\in I} M_l(i)^{\tilde{M}(i)}\exp\left(-M_l(i)\right)\prod_{i\in I}k(i)^{N(i)-\tilde{M}(i)} =$$
$$\lim_{l\to\infty}\prod_{i\in I} k(i)^{\tilde{M}(i)}\exp\left(v_l(i)\tilde{M}(i)\right)\exp\left(-M_l(i)\right)\prod_{i\in I}k(i)^{N(i)-\tilde{M}(i)} =$$
$$\lim_{l\to\infty}\prod_{i\in I} k(i)^{N(i)}\exp\left(v_l(i)N(i)\right)\exp\left(-M_l(i)\right) = \lim_{l\to\infty}\prod_{i\in I} M_l(i)^{N(i)}\exp\left(-M_l(i)\right) =$$
$$\prod_{i\in I}\tilde{M}(i)^{N(i)}\exp\left(-\tilde{M}(i)\right) = \mathrm{L}(\tilde{M})\ .$$

Quindi per ogni $m \in \overline{M}(k,\mathbb{H})$ si ha

$$\mathrm{L}(\tilde{M}) \geq \mathrm{L}(m)$$

e quindi $\tilde{M}$ è lo stimatore di massima verosimiglianza. □

3.8 Esistenza dello stimatore di massima verosimiglianza

Abbiamo visto che, nel caso in cui esista, lo stimatore di massima verosimiglianza può essere calcolato risolvendo un sistema di equazioni. Vedremo ora alcune condizioni che ne garantiscono l'esistenza. Per prima cosa osserviamo che in un modello log-affine esteso l'esistenza e l'unicità sono garantite se lo spazio $\mathbb{H}$ contiene il vettore unitario costante uguale a 1, come afferma la seguente proposizione:

Proposizione 3.4 *Sia $\overline{M}(k,\mathbb{H})$ un modello log-affine esteso tale che lo spazio $\mathbb{H}$ contenga il vettore costante uguale a 1,*

$$e = \{1\}_{i\in I}$$

Allora valgono i seguenti fatti.

1. *Se la distribuzione della tabella dei conteggi è 1) o 2), cioè di Poisson o multinomiale, esiste lo stimatore di massima verosimiglianza $\hat{M}$ ed è unico.*

2. *Se le distribuzione della tabella dei conteggi è 3) e lo spazio $F_B \subseteq \mathbb{H}$ allora esiste lo stimatore di massima verosimiglianza ed è unico.*

Dimostrazione: 1. Supponiamo che la distribuzione della tabella dei conteggi sia 1) oppure 2) e consideriamo un campione $\{n(i)\}_{i\in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i\in I}$. Chiamiamo e il vettore unitario, contenuto in $M(k,\mathbb{H})$. Sia M_N il sottoinsieme di $\overline{M}(k,\mathbb{H})$ dove vale:

$$m \in M_N \Leftrightarrow \sum_{i\in I} m(i) =\mid n \mid \ .$$

L'insieme M_N è chiuso e limitato quindi è compatto.

Poiché M_N è compatto, su di esso la funzione di verosimiglianza ammette un massimo. Sia questo massimo $\tilde{m}$.

Consideriamo un generico elemento $m \in M(k,\mathbb{H})$ e sia

$$\mid m \mid = \sum_{i\in I} m(i) \ .$$

Calcoliamo la verosimiglianza in m:

$$\mathrm{L}(m) = \prod_{i\in I} m(i)^{n(i)} e^{-m(i)} = \prod_{i\in I} \left[\left(\frac{m(i) \mid n \mid}{\mid m \mid} \right)^{n(i)} e^{-\frac{m(i)|n|}{|m|}} \right] \mid n \mid^{-|n|} \mid m \mid^{|n|} e^{|n|} e^{-|m|} =$$
$$\mathrm{L}\left(\frac{m \mid n \mid}{\mid m \mid} \right) \mid m \mid^{-|n|} \mid m \mid^{|n|} e^{|n|} e^{-|m|} \ .$$

Poiché $m \in M(k,\mathbb{H})$, esiste un vettore $v \in \mathbb{H}$ tale che:

$$m = k \exp(v)$$

e quindi:

$$\frac{m}{\mid m \mid} = \frac{\mid n \mid}{\mid m \mid} k \exp(v) = k \exp\left(\log\left(\frac{\mid n \mid}{\mid m \mid} \right) e + v \right)$$

e quindi $\frac{m|n|}{|m|} \in M(k,\mathbb{H})$.

Poiché:

$$\sum_{i\in I} \frac{m(i) \mid n \mid}{\mid m \mid} = \frac{\mid m \mid\mid n \mid}{\mid m \mid} =\mid n \mid$$

si ha che:

$$\frac{m \mid n \mid}{\mid m \mid} \in M_N$$

e quindi:

$$\mathrm{L}\left(\frac{m \mid n \mid}{\mid m \mid}\right) \mid n \mid^{-|n|} \mid m \mid^{|n|} e^{|n|} e^{-|m|} \leq \mathrm{L}\left(\tilde{m}\right) \mid n \mid^{-|n|} \mid m \mid^{|n|} e^{|n|} e^{-|m|} .$$

Infine, applicando il lemma A.2, si ha:

$$\mathrm{L}(m) \leq \mathrm{L}\left(\tilde{m}\right) \mid m \mid^{|n|} e^{-|m|} \mid n \mid^{-|n|} e^{|n|} \leq$$
$$\mathrm{L}\left(\tilde{m}\right) \mid n \mid^{|n|} e^{-|n|} \mid n \mid^{-|n|} e^{|n|} = \mathrm{L}\left(\tilde{m}\right)$$

e quindi $\tilde{m}$ è la stima di massima verosimiglianza. L'unicità è già stata provata in precedenza.

2. Supponiamo che la distribuzione della tabella dei conteggi sia 3). Abbiamo dimostrato che la presenza nello spazio $\mathbb{H}$ del vettore unitario garantisce l'esistenza di un vettore $\tilde{m}$ che verifica le equazioni di verosimiglianza. Supponiamo che $F_B \subseteq \mathbb{H}$.

   Allora, per ogni $i_B \in I_B$, abbiamo subito che il vettore χ^{i_B}, così definito:

   $$\chi^{i_B}(j) = \delta_{i_B, j_B} ,$$

   dove per δ si intende il delta di Kronecker, è contenuto in $\mathbb{H}$.

   Dato che $\tilde{m}$ verifica le equazioni di verosimiglianza si ha:

   $$\tilde{m}(i_B) = \sum_{j \in I} \hat{m}(j) \chi^{i_B}(j) = < \hat{m}, \chi^{i_B} > = < n, \chi^{i_B} > = n(i_B) .$$

   Quindi $\tilde{m}$ verifica le condizioni aggiuntive per il modello 3). Questo conclude la dimostrazione.

□

Questa condizione tuttavia garantisce l'esistenza dello stimatore di massima verosimiglianza solo nel modello $\overline{M}(k, \mathbb{H})$. Può essere interessante invece stabilire quando lo stimatore di massima verosimiglianza è presente nel modello non esteso.

Nel seguito sono presentate due condizioni necessarie e sufficienti per l'esistenza dello stimatore in modelli non estesi.

Proposizione 3.5 *Consideriamo un modello log-affine $M(k, \mathbb{H})$. Supponiamo di osservare una tabella dei conteggi $\{n(i)\}_{i \in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i \in I}$. Allora la stima di massima verosimiglianza $\hat{m}$ del vettore delle medie esiste se e solo se esiste $\delta \in \mathbb{H}^{\perp\!\!\!\perp}$ tale che:*

$$\delta(i) + n(i) > 0 \quad \forall i \in I .$$

Dimostrazione: Supponiamo che esista la stima di massima verosimiglianza $\hat{m}$. Allora essa soddisfa le equazioni di verosimiglianza. Pertanto, se $\hat{m} = k\exp(\hat{\mu})$, vale che:

$$\hat{m} - n \in \mathbb{H}^{\perp\!\!\!\perp} \ .$$

Inoltre vale:

$$\hat{m} = \hat{m} - n + n \ .$$

Dato che $m(i) > 0 \ \forall i \in I$, prendendo

$$\delta = \hat{m} - n$$

si ha la tesi.

Supponiamo che esista $\delta \in \mathbb{H}^{\perp\!\!\!\perp}$ tale che verifichi la condizione

$$\delta(i) + n(i) > 0 \quad \forall i \in I \ .$$

Dato che δ appartiene all'ortogonale di $\mathbb{H}$ mentre μ appartiene ad $\mathbb{H}$ vale:

$$< n, \mu >=< n + \delta, \mu > \ .$$

Quindi la log-verosimiglianza può essere riscritta nel modo seguente:

$$\mathrm{l}(\mu) = \sum_{i \in I} n(i)\mu(i) - k(i)e^{\mu(i)} = \sum_{i \in I} \left(n(i) + \delta(i)\right)\mu(i) - k(i)e^{\mu(i)} \ .$$

Dato che k è fissato non nullo e $n(i) + \delta(i) > 0$ per ogni $i \in I$, vale il seguente limite:

$$\lim_{|\mu(i)| \to \infty} (n(i) + \delta(i))\mu(i) - k(i)e^{\mu(i)} = -\infty \ .$$

Quindi, dato il seguente insieme:

$$A = \{\mu \in \mathbb{H} : \quad \mathrm{l}(\mu) \geq \mathrm{l}(0)\}$$

esso è limitato. Poiché la funzione $\mathrm{l}(\mu)$ è continua, A è anche chiuso. Quindi A è compatto e pertanto la verosimiglianza ammette su di esso un massimo. Sia questo massimo $\hat{\mu}$. Per come è definito A, è immediato dimostrare che:

$$l(\hat{\mu}) \geq l(\mu) \quad \forall \mu \in \mathbb{H} \ .$$

E quindi $\hat{\mu}$ è la stima di massima verosimiglianza. □

Osservazione: Pertanto, se vale

$$n(i) > 0 \quad \forall i \in I$$

scegliendo come δ il vettore identicamente nullo si verifica l'ipotesi della proposizione 3.5 e quindi esiste la stima di massima verosimiglianza.

Quindi la stima di massima verosimiglianza non appartiene a $M(k, \mathbb{H})$ solo in casi in cui si sono osservati dei valori nulli in alcune celle della tabella di contingenza. Se la stima di massima verosimiglianza $\hat{m}$ non esiste in $M(k, \mathbb{H})$ ma esiste in $\overline{M}(k, \mathbb{H})$ necessariamente essa presenta alcune componenti nulle. Esiste cioè almeno un indice $i \in I$ per cui $\hat{m}(i) = 0$. Ciò avviene solo se $n(i) = 0$.

Proposizione 3.6 *In un modello log-affine $M(k, \mathbb{H})$ se $n(i) > 0 \ \forall i \in I$ allora esiste la stima di massima verosimiglianza per il vettore delle medie m*

Un'altra condizione per l'esistenza della stima di massima verosimiglianza è la seguente.

Proposizione 3.7 *Sia $M(k, \mathbb{H})$ un modello log-affine. Supponiamo di osservare una tabella dei conteggi $\{n(i)\}_{i \in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i \in I}$. Allora la stima di massima verosimiglianza $\hat{\mu}$ di μ esiste se e solo se non esiste $\mu \in \mathbb{H}$ tale che valgono:*

$$i) \quad \mu \neq 0$$

$$ii) \quad \mu(i) \leq 0 \quad \forall i \in I$$

$$iii) \quad < n, \mu >= 0 \ .$$

Dimostrazione: Supponiamo che esista $\hat{\mu}$ stima di massima verosimiglianza. Allora per la proposizione 3.5 esiste $\delta \in \mathbb{H}^{\perp\!\!\!\perp}$ tale che

$$n(i) + \delta(i) > 0 \quad \forall i \in I \ .$$

Sia $\mu \in \mathbb{H}$ tale che

$$\mu \neq 0 \quad < \mu, n >= 0 \quad \mu(i) \leq 0 \ \forall i \in I \ .$$

Ma allora

$$< n + \delta, \mu >< 0 \ .$$

Ma vale anche:

$$< n + \delta, \mu >=< n, \mu >= 0$$

e ciò è assurdo, quindi un tale μ non può esistere.

Viceversa, supponiamo che non esista la stima di massima verosimiglianza. Allora non esiste un $\delta \in \mathbb{H}^{\perp\!\!\!\perp}$ tale che $n(i) + \delta(i) > 0 \ \forall i \in I$.

Dalla proposizione 3.6 sappiamo che deve esistere almeno un indice $i \in I$ tale che $n(i) = 0$. Quindi l'insieme

$$I_0 = \{i \in I : \quad n(i) = 0\}$$

è non vuoto. Definiamo

$$S = \{x \in \mathbb{R}^I : \quad x(i) > 0 \ \forall i \in I_0\} \ .$$

Allora S e $\mathbb{H}^{\perp\!\!\!\perp}$ sono due insiemi convessi disgiunti. Se per assurdo, infatti, esistesse $\nu \in S \cap \mathbb{H}^{\perp\!\!\!\perp}$, allora, per non violare la proposizione 3.5, dovrebbe esistere almeno un indice $\bar{i} \in I$ tale che $\nu(i) + n(i) \leq 0$. Sia:

$$\bar{I} = \{i \in I \setminus I_0 : \quad \nu(i) + n(i) \leq 0\} = \{i \in I \setminus I_0 : \quad \nu(i) \leq -n(i)\} \ .$$

Sia $K \in \mathbb{R}$ tale che:

$$0 < K < \min\left\{-\frac{n(i)}{\nu(i)} : \quad i \in \bar{I}\right\} \ .$$

Allora ovviamente $K\nu \in S \cap \mathbb{H}^{\perp\!\!\!\perp}$ e siccome per ogni $i \in \bar{I}$ $\nu(i) < 0$ si ha:

$$K\nu(i) + n(i) > \nu(i) \min\left\{-\frac{n(j)}{\nu(j)} : \quad j \in \bar{I}\right\} + n(i) \geq -\frac{n(i)}{\nu(i)}\nu(i) + n(i) \geq 0 \ .$$

Ciò è assurdo in quanto violerebbe la proposizione 3.5.

Per il teorema di Hahn-Banach esiste $\mu \in \mathbb{R}^I$, $\mu \neq 0$ tale che:

$$< \mu, \nu > \geq 0 \quad \forall \nu \in \mathbb{H}^{\perp\!\!\!\perp}$$

$$< \mu, \nu > \leq 0 \quad \forall \nu \in S \ .$$

Sia $j \in I \setminus I_0$. Sia δ^j il vettore nullo con un solo 1 nella posizione j, cioè, se δ_{ij} è il delta di Kronecker,

$$\delta^j = \{\delta_{ij}\}_{i \in I} \ .$$

Sia ν un generico vettore di S. Allora per ogni $c \in \mathbb{R}$

$$\nu + c\delta^j \in S \ .$$

Quindi deve essere

$$< \mu, \nu + c\delta^j > \leq 0 \ .$$

Ossia, per le proprietà del prodotto scalare:

$$< \mu, \nu > + \mu(j)c \leq 0 \ .$$

Poiché ciò vale ogni costante c necessariamente

$$\mu(j) = 0 \ .$$

Sia $j \in I_0$. Allora, se $\mu \in S$, per ogni costante positiva c vale:

$$\mu + c\delta^j \in S \ .$$

Come prima deve essere:

$$< \mu, \nu + c\delta^j >=< \mu, \nu > + c\mu(j) \leq 0 \ .$$

Necessariamente

$$\mu(j) \leq 0 \ .$$

Sia $\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu$ la proiezione ortogonale di μ su $\mathbb{H}^{\perp\!\!\!\perp}$.

Dato che ovviamente $-\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu \in \mathbb{H}^{\perp\!\!\!\perp}$,

$$< \mu, -\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu > \geq 0 \ .$$

Ma per le proprietà del prodotto scalare di ha:

$$< \mu, -\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu >=< \pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu, -\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu >= - \mid\mid \pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu \mid\mid \leq 0 \ .$$

necessariamente deve essere che $\pi_{\mathbb{H}^{\perp\!\!\!\perp}}\mu = 0$ e quindi

$$\mu \in \mathbb{H} \ .$$

□

3.9 Distribuzione dello stimatore di massima verosimiglianza

Consideriamo un modello log-affine $M(k, \mathbb{H})$ e supponiamo di osservare un campione $\{n(i)\}_{i \in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i \in I}$. La distribuzione dello stimatore di massima verosimiglianza del vettore delle medie M dipende dalla distribuzione della tabella dei conteggi e in generale è difficile da calcolare. Tuttavia, sfruttando il fatto che si tratta di modelli di classe esponenziale, è possibile darne una distribuzione asintotica.

Vediamo come si comporta questa distribuzione asintotica nei casi in cui siano stati scelte le distribuzioni 1) e 2), cioè le distribuzioni di Poisson e multinomiale.

Per prima cosa consideriamo il modello saturo, cioè il modello $M(k, \mathbb{H})$ dove $\mathbb{H} = \mathbb{R}^{|I|}$. Allora la verosimiglianza sotto la distribuzione di Poisson è la seguente:

$$\mathrm{L}(m) = \prod_{i \in I} \frac{m(i)^{n(i)} \exp(-m(i))}{n(i)!} = \prod_{i \in I} \frac{1}{n(i)!} \exp\left(\sum_{i \in I} n(i) \log(m(i)) - m(i) \right) \ .$$

Sappiamo che

$$m = k \exp(\mu) \ .$$

Riscriviamo la verosimiglianza in termini di μ ed otteniamo:

$$\mathrm{L}(\mu) \propto \exp\left(\sum_{i\in I} n(i)\mu(i) - \sum_{i\in i} k(i)\exp(\mu(i))\right) \ .$$

Quindi μ è il parametro canonico della forma esponenziale e la funzione link è:

$$g(\mu(i)) = k(i)\exp(\mu(i)) \ .$$

L'informazione attesa è la derivata seconda della funzione g cioè la matrice diagonale:

$$\mathrm{I}(\mu) = (k(i)\exp(\mu(i))\delta_{i,j})_{i,j\in I} = (m(i)\delta_{i,j})_{i,j\in I} \ .$$

La verosimiglianza nel caso di distribuzione dei conteggi multinomiale è invece:

$$\mathrm{L}(m) = \frac{\mid n \mid!}{\prod_{i\in I} n(i)!}\prod_{i\in I}\left(\frac{m(i)}{\sum_{j\in I} m(j)}\right)^{n(i)} \ .$$

Parametrizzando in μ, cioè sostituendo m con $k\exp(\mu)$, otteniamo:

$$\mathrm{L}(\mu) = \frac{\mid n \mid!}{\prod_{i\in I} n(i)!}\prod_{i\in I}\left(\frac{k(i)\exp(\mu(i))}{\sum_{j\in I} k(i)\exp(\mu(i))}\right)^{n(i)} \propto$$
$$\exp\left(\sum_{i\in I}(\mu(i)n(i)) - \mid n \mid \log\left(\sum_{j\in I} k(j)\exp(\mu(j))\right)\right) \ .$$

Quindi anche in questo caso μ è il parametro canonico della forma esponenziale. La funzione g corrispondente è:

$$g(\mu(i)) = \log\left(\sum_{j\in I} k(j)\exp(\mu(j))\right) \ .$$

L'informazione attesa, cioè la derivata seconda della funzione g, è la seguente:

$$\mathrm{I}(\mu) = \left(m(i)\delta_{i,j} - \frac{m(i)m(j)}{\mid n \mid}\right)_{i,j\in I} \ .$$

Come già sappiamo, indipendentemente dalla distribuzione della tabella dei conteggi, per ogni $i \in I$ $N(i)$ è lo stimatore di massima verosimiglianza per $m(i)$. Inoltre, trattandosi di modelli di classe esponenziale, N converge asintoticamente a una variabile aleatoria normale:

$$N \sim^a \mathcal{N}^{|I|}\left(m, \mathrm{I}^{-1}(\mu)\right) \ .$$

Supponiamo ora che il modello $M(k, \mathbb{H})$ non sia saturo, cioè che $\mathbb{H}$ sia un sottospazio vettoriale di $\mathbb{R}^{|I|}$ con $h <\mid I \mid$.

Allora esiste una matrice di proiezione A, che, se moltiplicata per un vettore di $\mathbb{R}^I$, restituisce la proiezione ortogonale del vettore nel sottospazio $\mathbb{H}$.

Si può dimostrare che in questo caso lo stimatore di massima verosimiglianza N converge asintoticamente alla seguente variabile aleatoria normale:

$$N \sim^a \mathcal{N}^{|I|}\left(m, A^t \mathrm{I}^{-1}(\mu) A\right) \ .$$

3.10 Test su sottomodelli

Consideriamo una tabella di contingenza $\{n(i)\}_{i\in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i\in I}$, un modello log-affine esteso $\overline{M}_0 = \overline{M}(k, \mathbb{H}_0)$ e un suo sottomodello $\overline{M}_1 = \overline{M}(k, \mathbb{H}_1)$, tale che:

$$\mathbb{H}_1 \subseteq \mathbb{H}_0 \ .$$

Sia $\hat{m}_0$ la stima di massima verosimiglianza per il vettore delle medie nel modello $\overline{M}_0$, $\hat{m}_1$ la stima di massima verosimiglianza nel modello $\overline{M}_1$ e $\hat{m}$ la stima di massima verosimiglianza nel modello saturo.

Denotiamo con I^+ l'insieme degli i per cui $n(i) > 0$ cioè:

$$I^+ = \{i \in I : \quad n(i) > 0\} \ .$$

La realizzazione campionaria della devianza del modello $\overline{M}_0$, che denotiamo con d_0, è la seguente:

$$d_0 = -2 \log\left(\frac{\mathrm{L}(\hat{m}_0)}{\mathrm{L}(\hat{m})}\right) = 2 \sum_{i\in I^+} n(i) \log\left(\frac{\hat{m}(i)}{\hat{m}_0(i)}\right) + 2 \sum_{i\in I} \hat{m}_0(i) - 2 \sum_{i\in I} \hat{m}(i) = \sum_{i\in I^+} n(i) \log\left(\frac{n(i)}{\hat{m}_0(i)}\right) \ .$$

Infatti:

$$\sum_{i\in I} \hat{m}(i) = \sum_{i\in I} \hat{m}_0(i) =\mid n \mid \ .$$

Allo stesso modo la realizzazione campionaria della devianza del modello $\overline{M}_1$, che indichiamo con d_1, è:

$$d_1 = \sum_{i\in I^+} n(i) \log\left(\frac{n(i)}{\hat{m}_1(i)}\right) \ .$$

Quindi la realizzazione campionaria della devianza statistica fra i due modelli, che indichiamo con d_{01}, è la seguente:

$$d_{01} = d_1 - d_0 = 2 \sum_{i \in I^+} n(i) \log \left(\frac{\hat{m}_0(i)}{\hat{m}_1(i)} \right) \ .$$

Per ciascun valore di i possiamo considerare lo sviluppo di Taylor seguente:

$$n(i) \log \left(\frac{n(i)}{\hat{m}_0(i)} \right) = n(i) - \hat{m}_0(i) + \frac{(n(i) - \hat{m}_0(i))^2}{2\hat{m}_0(i)} + \mathrm{o}\left\{ (n(i) - \hat{m}_0(i))^2 \right\} \ .$$

Pertanto, se $n(i)$ è molto vicino a $\hat{m}_0(i)$, è possibile approssimare d_0 nel modo seguente:

$$d_0 \approx \sum_{i \in I^+} \frac{(n(i) - \hat{m}_0(i))^2}{\hat{m}_0(i)} \ .$$

Comunemente si utilizza l'approssimazione della devianza con la variabile aleatoria χ-quadrato di Pearson $\tilde{D}_0$, la cui realizzazione campionaria d_0 è la seguente:

$$\tilde{d}_0 = \sum_{i \in I_0^*} \frac{(n(i) - \hat{m}_0(i))^2}{\hat{m}_0(i)}$$

dove l'insieme I_0^* è così definito:

$$I_0^* = \{ i \in I : \quad \hat{m}_0(i) > 0 \} \ .$$

Allo stesso modo un'approssimazione della devianza d_{01} è:

$$\tilde{d}_{01} = \sum_{i \in I_1^*} \frac{(\hat{m}_0(i) - \hat{m}_1(i))^2}{\hat{m}_1(i)} \ .$$

L'approssimazione della devianza con una variable aleatoria χ-quadrato, tuttavia, presenta una difficoltà: il numero dei gradi di libertà in generale non è facilmente calcolabile. Tuttavia vedremo in seguito che questa difficoltà svanisce nel caso di modelli gerarchici.

3.11 Riferimenti bibliografici

La parte del capitolo che descrive le tabelle di contingenza, presenta i modelli log-affini e trova la forma dello stimatore di massima verosimiglianza, presente nei paragrafi da 3.3 a 3.7, è una libera rielaborazione del testo di Lauritzen *Graphical Models* [7]. Alcune dimostrazioni, ad esempio la proposizione 3.1, non presenti nel testo di riferimento, sono state svolte autonomamente. La

lettura dei modelli log-lineari come modelli di classe esponenziale e l'individuazione della statistica sufficiente e minimale, presente nel paragrafo 3.5, è stata tratta dal testo di Shelby Haberman Log-Linear Models for Frequency Data: Sufficient Statistics and Likelihood Equations [5].

Il paragrafo 3.8, che considera le condizioni di esistenza dello stimatore di massima verosimiglianza in un modello log-affine, è tratto dal testo di Shelby Haberman [5]. La proposizione 3.4, che fornisce una condizione sufficiente per l'esistenza dello stimatore di massima verosimiglianza in un modello log-affine esteso, è stata invece dimostrata prendendo spunto da diverse proposizioni del testo di Lauritzen [7].

I paragrafi 3.9 e 3.10, relativi alla distribuzione dello stimatore di massima verosimiglianza e alla devianza, infine, sono interamente tratti dal testo di Lauritzen [7].

Capitolo 4

Modelli gerarchici e modelli grafici

4.1 Introduzione

Sia Γ un insieme di variabili aleatorie discrete o criteri di classificazione. Per ogni $\delta \in \Gamma$ consideriamo un insieme finito I_δ che contiene i possibili valori che possono essere assunti da δ. Chiamiamo I il prodotto cartesiano di tutti gli insiemi I_δ. Sia $\mathbf{A}$ un insieme di sottoinsiemi di Γ.

Definizione 4.1 *Si definisce* modello gerarchico *generato da* $\boldsymbol{A}$ *il modello log-affine* $M(k, \mathbb{H}_{\boldsymbol{A}})$*, dove* $\mathbb{H}_{\boldsymbol{A}}$ *è lo spazio vettoriale così definito:*

$$\mathbb{H}_{\boldsymbol{A}} = \sum_{A \in \boldsymbol{A}} F_A$$

dove F_A *è definito nell'equazione 3.1.*

Il modello gerarchico generato da $\mathbf{A}$ generalmente si indica con $M(k, \mathbf{A})$ o, nel caso in cui k sia il vettore costantemente uguale a 1, con $M(\mathbf{A})$.

Il modello saturo, ad esempio, si può considerare come un modello gerarchico con $\mathbf{A}$ uguale all'insieme delle parti di Γ.

Nel seguito tratteremo solo modelli gerarchici che comprendono tutti gli effetti principali. Assumeremo cioè che valga:

$$F_\alpha \subseteq \mathbb{H}_{\mathbf{A}} \quad \forall \alpha \in \Gamma \ .$$

Se $\mathbf{A}$ è un insieme di sottoinsiemi di Γ, definiamo $\mathbf{A}_C$ l'insieme degli elementi di $\mathbf{A}$ che sono massimali rispetto all'inclusione.

Allora è immediato dimostrare che:

$$\mathbb{H}_{\mathbf{A}} = \mathbb{H}_{\mathbf{A}_C} \ .$$

Si dice che l'insieme $\mathbf{A}$ è *ridotto* se i suoi elementi sono massimali rispetto all'inclusione, cioè se non contiene elementi che sono sottoinsiemi di altri suoi elementi.

Uno spazio vettoriale $\mathbb{H}_{\mathbf{A}}$ è univocamente determinato dall'insieme $\mathbf{A}$ solo se esso è ridotto. In questo caso si dice che $\mathbf{A}$ è una *classe generatrice*.

Gli elementi presenti in $\mathbf{A}$ determinano le interazioni possibili tra le variabili del modello. Nel modello più generale, cioè quello saturo, sono presenti tutte le interazioni possibili. Se si assume l'indipendenza di alcune variabili l'interazione tra di esse non è presente nel modello.

Esempio: Consideriamo una tabella di contingenza bidimensionale di dimensione $r \times c$ tale che la probabilità della cella (i, j), con $1 \leq i \leq r$ e $1 \leq j \leq c$, sia $p_{i,j} \geq 0$.

Sia $I = \{1, 2, \ldots, r\} \times \{1, 2, \ldots, c\}$. Supponiamo inoltre che le variabili x e y, che rappresentano rispettivamente righe e colonne, siano indipendenti, cioè che valga:

$$p_{i,j} = p_{i,+} p_{+,j} \ .$$

Si tratta dell'esempio 3.4, già presentato nel precedente capitolo. Esso è un modello gerarchico. Essendo x e y indipendenti l'interazione tra di esse non è presente nel modello. Pertanto l'insieme $\mathbf{A}$ è così definito:

$$\mathbf{A} = \{\{x\}, \{y\}\} \ .$$

Lo spazio $\mathbb{H}$ è dato da:

$$\mathbb{H} = F_x + F_y \ .$$

La sua dimensione è:

$$\dim(\mathbb{H}) = \dim(F_x) + \dim(F_y) - \dim(F_x \cap F_y) = r + c - 1 \ .$$

Infatti $F_x \cap F_y$ è lo spazio generato dal vettore unitario e e ha dimensione 1.

Consideriamo ora un modello gerarchico $M(\mathbf{A})$, con $\mathbf{A}$ sua classe generatrice. Ad esso è possibile associare un grafo indiretto $\mathbb{G} = \mathbb{G}(\mathbf{A})$, detto *grafo di interazione*, definito nel modo seguente:

$$\alpha \sim \beta \text{ in } \mathbb{G} \Leftrightarrow \{\alpha, \beta\} \subseteq A \text{ per qualche } A \in \mathbf{A} \ ,$$

cioè due variabili α e β sono adiacenti nel grafo se e solo se l'interazione tra di essa è permessa nel modello gerarchico.

Tuttavia modelli gerarchici diversi possono avere lo stesso grafo di interazione.

Esempio: Sia $\Gamma = \{1, 2, 3\}$ e siano:

$$\mathbf{A}_1 = \{\{1, 2, 3\}\} \quad \mathbf{A}_2 = \{\{1, 2\}, \{1, 3\}, \{2, 3\}\} \ .$$

$\mathbf{A}_1$ e $\mathbf{A}_2$ sono distinti e generano due modelli gerarchici distinti. Tuttavia il loro grafo di interazione è lo stesso, un grafo completo con tre vertici.

Quindi in generale il grafo di interazione non fornisce tutte le informazioni su un modello gerarchico, tranne nel caso in cui sia grafico.

Definizione 4.2 *Un modello gerarchico $M(\boldsymbol{A})$ con classe generatrice $\boldsymbol{A}$ è detto* grafico *se*

$$\boldsymbol{A} = \mathbb{C}$$

dove $\mathbb{C}$ è l'insieme delle cliques del suo grafo di interazione.

Un modello grafico può essere spiegato completamente tramite il suo grafo di interazione. In particolare esso permette di leggere le indipendenze condizionate tra le variabili del modello. Se l'insieme C separa gli insiemi A e B nel grafo le variabili contenute in A sono indipendenti da quelle contenute in B date le variabili contenute in C. Componenti connesse distinte del grafo, invece, individuano variabili tra di loro indipendenti.

Dato che i modelli grafici sono un caso particolare di modelli gerarchici, nel seguito tratteremo la stima di massima verosimiglianza nel caso più generale di questi ultimi.

4.2 Stima di massima verosimiglianza

Consideriamo un modello gerarchico esteso $\overline{M}(k, \mathbf{A})$, con $\mathbf{A}$ classe generatrice. Supponiamo di osservare un campione $\{n(i)\}_{i \in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i \in I}$. Vale la seguente proposizione.

Proposizione 4.1 *Esiste lo stimatore di massima verosimiglianza del vettore delle medie ed è l'unico elemento $\hat{M}$ di $\overline{M}(k, \boldsymbol{A})$ che verifica il sistema di equazioni:*

$$\hat{M}(i_A) = N(i_A) \quad \forall i_A \in A, A \in \boldsymbol{A} \ .$$

Dimostrazione: Per prima cosa osserviamo che, per come è costruito lo spazio vettoriale $\mathbb{H}_{\mathbf{A}}$, l'appartenenza ad esso del vettore costante è garantita. Infatti per ogni $A \in \mathbf{A}$ il vettore $x(i) = 1 \ \forall i \in I$ verifica la condizione:

$$x(i) = x(j) \quad \forall i, j \in I$$

e quindi anche la condizione meno restrittiva:

$$x(i) = x(j) \quad \forall i, j \in I \text{ con } i_A = j_A \ .$$

Pertanto è garantita l'esistenza e l'unicità dello stimatore di massima verosimiglianza, che chiamiamo $\hat{M}$.

Per la proposizione 3.3 abbiamo quindi che $\hat{M}$ verifica il sistema di equazioni:

$$\pi_{\mathbb{H}_A}\hat{M} = \pi_{\mathbb{H}_A}N \ .$$

Sia A un generico sottoinsieme di $\mathbf{A}$. Dato che

$$F_A \subseteq \mathbb{H}_{\mathbf{A}}$$

è evidente che, per ogni vettore $x \in \mathbb{R}^I$, vale:

$$\pi_A x(i) = \pi_A \pi_{\mathbb{H}_{\mathbf{A}}} x(i) \ .$$

Da ciò segue:

$$\pi_A \hat{M}(i) = \pi_A \pi_{\mathbb{H}_{\mathbf{A}}} \hat{M}(i) = \pi_A \pi_{\mathbb{H}_{\mathbf{A}}} N(i) = \pi_A N(i) \ .$$

Infine, dalla definizione di π_A data nell'equazione 3.2, si ha:

$$\hat{M}(i_A) = N(i_A) \ .$$

□

4.3 Devianza dei modelli gerarchici

Consideriamo un modello gerarchico $M(k, \mathbf{A}_0)$ e un suo sottomodello $M(k, \mathbf{A}_1)$. Supponiamo che la stima di massima verosimiglianza del vettore delle medie nel modello $M(k, \mathbf{A}_1)$ sia positiva su ogni valore di $i \in I$.

Come abbiamo già visto, la devianza dei due modelli ha una distribuzione approssimabile con una variabile aleatoria χ-quadrato. I gradi di libertà di questa variabile aleatoria, che chiamiamo f, sono pari alla differenza delle dimensioni degli spazi vettoriali dei due modelli, cioè:

$$f = \dim(\mathbb{H}_{\mathbf{A}_0}) - \dim(\mathbb{H}_{\mathbf{A}_1}) \ .$$

Vediamo quindi come calcolare queste dimensioni.

Consideriamo un generico insieme $A \in \mathbf{A}_0$. La dimensione dello spazio F_A è pari alla cardinalità dell'insieme I_A cioè:

$$\dim(F_A) = \mid I_A \mid = \prod_{\alpha \in A} \mid I_\alpha \mid \ .$$

Se $\hat{m}(i) = 0$ per qualche $i \in I$ per ogni $\alpha \in \Gamma$ la dimensione di I_α si calcola nel modo seguente:

$$\mid I_\alpha \mid = \sum_{i_\alpha \in I_\alpha : \hat{m}(i_\alpha) > 0} 1 \ .$$

La dimensione di I_α è cioè il numero di valori assumibili dalla variabile aleatoria criterio di classificazione α per cui la stima in massima verosimiglianza dei conteggi attesi è positiva.

Consideriamo ora un insieme $B \subseteq \Gamma$ e consideriamo il seguente insieme:

$$\mathbf{A}^* = \{B \cap A : \quad A \in \mathbf{A}_0\} \ .$$

Supponiamo che l'insieme $\mathbf{A}^*$ sia stato ridotto, in modo da essere una classe generatrice.

Vale la seguente proposizione:

Proposizione 4.2 *Siano $\boldsymbol{A}$ e $\boldsymbol{B}$ due classi generatrici. Definiamo il seguente insieme $\boldsymbol{C}$:*

$$\boldsymbol{C} = \{A \cap B : \quad A \in \boldsymbol{A}, \quad B \in \boldsymbol{B}\} \ .$$

Sia $\boldsymbol{C}^*$ la classe generatrice ottenuta riducendo l'insieme $\boldsymbol{C}$.

Allora vale:

$$\mathbb{H}_{\boldsymbol{C}^*} = \mathbb{H}_{\boldsymbol{A}} \cap \mathbb{H}_{\boldsymbol{B}} \ .$$

Dimostrazione: Sia μ un generico vettore di $\mathbb{H}_{\mathbf{C}^*}$. Allora esistono due insiemi $A \in \mathbf{A}$ e $B \in \mathbf{B}$ tali che:

$$\mu \in F_A \subseteq \mathbb{H}_{\mathbf{A}}, \quad \mu \in F_B \subseteq \mathbb{H}_{\mathbf{B}} \ .$$

Quindi:

$$\mu \in \mathbb{H}_{\mathbf{A}} \cap \mathbb{H}_{\mathbf{B}} \ .$$

Sia μ un generico vettore di $\mathbb{H}_{\mathbf{A}} \cap \mathbb{H}_{\mathbf{B}}$. Allora $\mu \in \mathbb{H}_{\mathbf{A}}$ e $\mu \in \mathbb{H}_{\mathbf{B}}$.

Siano $A_1, \ldots, A_n$ e $B_1, \ldots, B_m$ insiemi contenuti in Γ tali che:

$$\mathbf{A} = \{A_1, \ldots, A_n\}, \quad \mathbf{B} = \{B_1, \ldots, B_m\} \ .$$

Ne segue ovviamente che:

$$\mathbb{H}_{\mathbf{A}} = F_{A_1} + \cdots + F_{A_n}, \quad \mathbb{H}_{\mathbf{B}} = F_{B_1} + \cdots + F_{B_n} \ .$$

Dimostriamo la tesi per induzione sulla cardinalità di $\mathbf{B}$.

Sia $\mid \mathbf{B} \mid = 1$. Allora vale che:

$$\mathbf{B} = \{B_1\}, \quad \mathbb{H}_{\mathbf{B}} = F_{B_1} \ .$$

Allora esistono n vettori $\alpha_1, \ldots, \alpha_n$ tali che:

$$\alpha_i \in F_{A_i} \quad \forall i \in \{1, \ldots, n\}$$

e:

$$\mu = \sum_{i=1}^{n} \alpha_i \ .$$

Consideriamo un generico α_i. Allora:

$$\pi_{B_1}\alpha_i = \pi_{B_1}\pi_{A_i}\alpha_i = \pi_{A_i\cap B_1}\alpha_i \ .$$

Dato che $\mu \in F_{B_1}$ ne otteniamo:

$$\mu = \pi_{B_1}\mu = \sum_{i=1}^{n} \pi_{A_i\cap B_1}\alpha_i \ .$$

Quindi vale che:

$$\mu \in F_{A_1\cap B_1} + \cdots + F_{A_n\cap B_1}$$

cioè

$$\mu \in \mathbb{H}_{\mathbf{C}^*} \ .$$

Supponiamo ora che la tesi sia vera per $\mid \mathbf{B} \mid \leq m-1$ e sia $\mid \mathbf{B} \mid = m$. Allora vale che:

$$\mathbf{B} = \{B_1, \ldots, B_m\}, \quad \mathbb{H}_{\mathbf{B}} = F_{B_1} + \cdots + F_{B_m} \ .$$

Allora esistono $\beta_1, \ldots, \beta_m$ vettori tali che:

$$\beta_i \in F_{B_i} \quad \forall i \in \{1, \ldots, m\} \ .$$

Inoltre si può scrivere:

$$\mu = \sum_{i=1}^{n} \alpha_i = \sum_{j=1}^{m} \beta_j \ .$$

Dato che $\pi_{B_m}\beta_m = \beta_m$ si ha che:

$$\beta_m = \sum_{i=1}^{n} \alpha_i - \sum_{j=1}^{m-1} \beta_j = \sum_{i=1}^{n} \pi_{A_i\cap B_m}\alpha_i - \sum_{j=1}^{m-1} \pi_{B_j\cap B_m}\beta_j \ .$$

Quindi, sommando ad entrambi i membri $\sum_{j=1}^{m-1} \beta_j$, si ottiene:

$$\mu = \sum_{i=1}^{n} \pi_{A_i\cap B_m}\alpha_i - \sum_{j=1}^{m-1} \pi_{B_j\cap B_m}\beta_j + \sum_{j=1}^{m-1} \beta_j = \sum_{i=1}^{n} \pi_{A_i\cap B_m}\alpha_i + \sum_{j=1}^{m-1} \left(\beta_j - \pi_{B_j\cap B_m}\beta_j\right) \ .$$

Dato che $\mu \in \mathbb{H}_{\mathbf{A}}$ e anche $\sum_{i=1}^{n} \pi_{A_i\cap B_m}\alpha_i \in \mathbb{H}_{\mathbf{A}}$ per costruzione, ne segue necessariamente che:

$$\sum_{j=1}^{m-1} \left(\beta_j - \pi_{B_j\cap B_m}\beta_j\right) \in \mathbb{H}_{\mathbf{A}} \ .$$

Ma, dato che per ogni $i \in \{1, \ldots, m-1\}$ si ha $\beta_j - \pi_{B_j \cap B_m} \beta_j \in F_{B_j}$, vale anche che:

$$\sum_{j=1}^{m-1} \left(\beta_j - \pi_{B_j \cap B_m} \beta_j\right) \in F_{B_1} + \cdots + F_{B_{m-1}} \ .$$

Quindi:

$$\sum_{j=1}^{m-1} \left(\beta_j - \pi_{B_j \cap B_m} \beta_j\right) \in \left(F_{B_1} + \cdots + F_{B_{m-1}}\right) \cap \mathbb{H}_{\mathbf{A}} \ .$$

Definiamo il seguente insieme:

$$\mathbf{K} = \{A \cap B : \quad A \in \mathbf{A}, B \in \{B_1, \ldots, B_{m-1}\}\} \ .$$

Sia $\mathbf{K}^*$ l'insieme $\mathbf{K}$ opportunamente ridotto. Allora è possibile applicare l'ipotesi induttiva ed affermare che:

$$\sum_{j=1}^{m-1} \left(\beta_j - \pi_{B_j \cap B_m} \beta_j\right) \in \mathbb{H}_{\mathbf{K}^*} \ .$$

Quindi si ha subito che:

$$\mu \in \mathbb{H}_{\mathbf{K}^*} + F_{A_1 \cap B_m} + \cdots + F_{A_n \cap B_m}$$

da cui la tesi:

$$\mu \in \mathbb{H}_{\mathbf{C}^*} \ .$$

□

Applicando la proposizione 4.2 otteniamo:

$$\mathbb{H}_{\mathbf{A}_0} \cap F_B = \mathbb{H}_{\mathbf{A}^*} \ .$$

Quindi possiamo calcolare la dimensione dell'unione dei due spazi nel modo seguente:

$$\dim\left(\mathbb{H}_{B \cup \mathbf{A}_0}\right) = \dim\left(F_B\right) + \dim\left(\mathbb{H}_{\mathbf{A}_0}\right) - \dim\left(\mathbb{H}_{\mathbf{A}^*}\right) \ .$$

Ciò fornisce una formula ricorsiva per calcolare la dimensione di qualunque spazio, partendo dai singoli insiemi contenuti nella sua classe generatrice.

Ricordiamo che:

$$\dim\left(F_\emptyset\right) = 1 \ ,$$

come del resto avevamo già visto in un esempio precedente. Infatti, per definizione, dato un vettore μ:

$$\mu \in F_\emptyset \quad \Leftrightarrow \quad \mu(i) = \mu(j) \quad \forall i, j \in I \ .$$

4.4 Aggiustamento delle marginali

Consideriamo una tabella dei conteggi $\{n(i)\}_{i\in I}$ realizzazione di una variabile aleatoria $\{N(i)\}_{i\in I}$ e un modello gerarchico log affine esteso $\overline{M} = \overline{M}(k, \mathbf{A})$ con classe generatrice $\mathbf{A}$. Definiamo l'insieme

$$M^* = \{\{m(i)\}_{i\in I} : \quad n(i) > 0 \Rightarrow m(i) > 0\} \ .$$

L'insieme M^* dipende dal campione osservato e inoltre vale:

$$M^* \cap \overline{M} = \left\{m \in \overline{M} : \quad \mathrm{L}(m) > 0\right\} \ .$$

Definizione 4.3 *Per ciascun $m \in M^* \cap \overline{M}$ e ciascun $A \in \boldsymbol{A}$ definiamo l'operazione di* aggiustamento della marginale *come:*

$$T_{Am}(i) = m(i)\frac{n(i_A)}{m(i_A)}$$

dove si considera $\frac{0}{0} = 0$.

Osserviamo che:

$$T_{Am}(i_A) = m(i_A)\frac{n(i_A)}{m(i_A)} = n(i_A)$$

e inoltre:

$$\mid T_{Am} \mid = \sum_{i\in I} m(i)\frac{n(i_A)}{m(i_A)} = \sum_{i_A\in I_A} \frac{n(i_A)}{m(i_A)} \sum_{j\in I: j_A = i_A} m(i) = \sum_{i_A\in I_A} m(i_A)\frac{n(i_A)}{m(i_A)} = \mid n \mid \ .$$

Vale la seguente proposizione.

Proposizione 4.3 *Per ogni $A \in \boldsymbol{A}$ valgono i seguenti fatti:*

1. *T_A è continua su M^**
2. *T_{Am} è l'unica stima di massima verosimiglianza per il valore atteso nel modello gerarchico esteso $\overline{M}(m, \{A\})$*
3. *$L(T_{Am}) \geq L(m)$ e vale l'uguaglianza se e solo se $m(i_a) = n(i_a)$ per ogni $i_a \in I_A$, cosa che accade se e solo se $T_{Am} = m$*
4. *$T_A\left(M^* \cap \overline{M}\right) \subseteq M^* \cap \overline{M}$.*

Dimostrazione: 1. Sia $\{m_s\}_{s\in\mathbb{N}}$ una successione di elementi di M^* tale che:

$$\lim_{s\to\infty} m_s = m \in M^* \ .$$

Sia $i \in I$ tale che $m(i_A) \neq 0$. Allora banalmente

$$\lim_{s\to\infty} T_{Am_s}(i) = T_{Am}(i) \ .$$

Sia invece $i \in I$ tale che $m(i_A) = 0$. Allora per ogni j tale che $j_A = i_A$ vale $m(j) = 0$.

Ma allora, poiché $m \in M^*$, deve essere che per ogni j così definito $n(j) = 0$.

Quindi vale:

$$0 = T_{Am_s}(i) = T_{Am}(i)$$

e quindi $m_s \to m$.

2. Per ogni $\varepsilon > 0$ si ha che:

$$\{T_{Am}(i)\}_{i\in I} = \left\{\lim_{\varepsilon\to 0} m(i)\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right\}_{i\in I} \ .$$

Fissato un valore di ε si ha che

$$\begin{aligned}\left\{m(i)\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right\}_{i\in I} &\in M(m,\{A\}) \\ \Leftrightarrow \left\{m(i)\exp\left(\log\left(\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right)\right)\right\}_{i\in I} &\in m\exp\left(\mathbb{H}_{\{A\}}\right) \\ \Leftrightarrow \left\{\log\left(\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right)\right\}_{i\in I} &\in \mathbb{H}_{\{A\}} = F_A \ .\end{aligned}$$

Siano i e j appartenenti a I tali che $i_A = j_A$; allora evidentemente:

$$\log\left(\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right) = \log\left(\frac{n(j_A)+\varepsilon}{m(j_A)+\varepsilon}\right) \ .$$

Quindi

$$\left\{m(i)\frac{n(i_A)+\varepsilon}{m(i_A)+\varepsilon}\right\}_{i\in I} \in M(m,\{A\}) \ .$$

Quindi, essendo il limite di una successione di M, vale che:

$$T_{Am} \in \overline{M}(m,\{A\}) \ .$$

Ricordiamo che

$$T_{Am}(i_A) = n(i_A) \ .$$

Ma queste non sono altro che le equazioni di verosimiglianza per il modello $\overline{M}(m, \{A\})$.

Quindi T_{Am} è l'unica stima di massima verosimiglianza per questo modello.

3. I vettori m e T_{Am} appartengono entrambi allo spazio $\overline{M}(m, \{A\})$. Dato che T_{Am} è l'unica stima di massima verosimiglianza, necessariamente deve essere:

$$\mathrm{L}(T_{Am}) \geq \mathrm{L}(m) \ .$$

Se vale uguaglianza allora anche m è stima di massima verosimiglianza e verifica le equazioni:

$$m(i_A) = n(i_A) \quad \forall i_A \in I \ .$$

Dato che lo stimatore di massima verosimiglianza è unico, ciò avviene se e solo se

$$T_{Am} = m \ .$$

4. Sia m un generico vettore di $M \cap M^*$. Allora abbiamo che, per ogni $\varepsilon > 0$,

$$T_{Am}(i) = \lim_{\varepsilon \to 0} m(i) \frac{n(i_A) + \varepsilon}{m(i_A) + \varepsilon} \ .$$

Siccome è un elemento di M, m può essere scritto come:

$$m = k \exp(\mu) \quad \mu \in \mathbb{H}_{\mathbf{A}} \ .$$

Di conseguenza indicando con v_ε il seguente vettore:

$$v_\varepsilon = \left\{ m(i) \frac{n(i_A) + \varepsilon}{m(i_A) + \varepsilon} \right\}_{i \in I}$$

si ha che:

$$\begin{aligned} v_\varepsilon \in M(k, \mathbf{A}) \Leftrightarrow \left\{ \mu(i) + \log\left(\frac{n(i_A) + \varepsilon}{m(i_A) + \varepsilon} \right) \right\}_{i \in I} \in \mathbb{H}_{\mathbf{A}} \\ \Leftrightarrow \left\{ \log\left(\frac{n(i_A) + \varepsilon}{m(i_A) + \varepsilon} \right) \right\}_{i \in I} \in \mathbb{H}_{\mathbf{A}} \ . \end{aligned}$$

Ma abbiamo già visto che

$$\left\{ \log\left(\frac{n(i_A) + \varepsilon}{m(i_A) + \varepsilon} \right) \right\}_{i \in I} \in F_A \subseteq \mathbb{H}_{\mathbf{A}} \ .$$

Quindi per ogni ε positivo

$$v_\varepsilon \in M \ .$$

Poiché T_{Am} è il limite di una successione di elementi di M si ha che:

$$T_{Am} \in \overline{M} \ .$$

Consideriamo un elemento i tale che $T_{Am}(i) = 0$. Allora, per come è definito T_{Am}, o $m(i) = 0$ oppure $n(i_A) = 0$.

Se $m(i) = 0$, allora, dato che $m \in \ M^*$, $n(i) = 0$.

Se invece $n(i_A) = 0$ allora necessariamente anche

$$n(j) = 0 \quad \forall j : \quad j_A = i_A$$

quindi $n(i) = 0$.

Quindi $T_{Am} \in M^*$.

Pertanto abbiamo dimostrato che:

$$T_A(M^* \cap M) \subseteq M^* \cap \overline{M} \ .$$

La continuità di T_A, dimostrata al punto 2), implica che:

$$T_A(M^* \cap \overline{M}) \subseteq M^* \cap \overline{M} \ .$$

□

4.5 L'algoritmo IPS

Possiamo ora utilizzare la proposizione 4.3 per definire un algoritmo, detto IPS o *Iterative Proportional Scaling*, che, partendo da un vettore iniziale m_0, converge alla stima di massima verosimiglianza.

Supponiamo che la classe generatrice $\mathbf{A}$ del modello abbia s elementi. Diamo un ordinamento casuale agli elementi di $\mathbf{A}$:

$$\mathbf{A} = (A_1, \ldots, A_s) \ .$$

Scegliamo un vettore arbitrario appartenente a $M(k, \mathbf{A})$, che chiamiamo m_0, e definiamo per $r = 0, 1, 2, \ldots$

$$m_{r+1} = (T_{A_1} \ldots T_{A_s})\, m_r \ .$$

Allora vale la seguente proposizione :

Proposizione 4.4 *Per qualunque vettore di partenza $m_0 \in M$, se $\hat{m}$ è la stima di massima verosimiglianza del vettore delle medie, si ha:*

$$\hat{m} = \lim_{r\to\infty} m_r \ .$$

Dimostrazione: Chiamiamo S la composizione delle funzioni T_i,

$$S = (T_1 \dots T_s) \ .$$

Si ha che

$$m_{r+1} = Sm_r \ .$$

Consideriamo il seguente insieme:

$$\Theta = \left\{ m \in \overline{M} : \quad \mathrm{L}(m) \geq \mathrm{L}(m_0) \wedge \mid m \mid = \mid n \mid = N \right\} \ .$$

Ricordando, dalla proposizione 4.3 , che per ogni vettore m

$$\mathrm{L}(T_i(m)) \geq \mathrm{L}(m)$$

e inoltre che

$$\mid T_i(m) \mid = N$$

si ha che le T_i sono funzioni continue e

$$T_i(\Theta) \subseteq \Theta \ .$$

Poiché Θ è compatto e

$$m_r \in \Theta \ \forall r \geq 1$$

esiste una sottosuccessione di elementi di Θ convergente m_{rn} tale che

$$\lim_{r\to\infty} m_{rv} = m^* \ .$$

Applicando l'operatore S al limite otteniamo:

$$\mathrm{L}\left(Sm^*\right) = \lim_{r\to\infty} \mathrm{L}\left(Sm_{rv}\right) \leq \lim_{r\to\infty} \mathrm{L}\left(m_{rv+1}\right) = \mathrm{L}(m^*) \ .$$

Ma dalla proposizione 4.3 sappiamo anche che:

$$\mathrm{L}\left(Sm^*\right) \geq \mathrm{L}\left(m^*\right) \ .$$

Quindi

$$\mathrm{L}\left(Sm^*\right) = \mathrm{L}\left(m^*\right) \ .$$

Quindi, per la proposizione 4.1 in cui si afferma che esiste un unico stimatore di massima verosimiglianza, possiamo concludere che m^* è la stima di massima verosimiglianza per il vettore delle medie. □

Osserviamo che, dalla proposizione 4.3, dato che $\mathrm{L}(Sm^*) = \mathrm{L}(m^*)$, possiamo anche dire:

$$Sm^* = m^* \ .$$

Ciò implica che

$$T_i(m^*) = m^* \quad \forall i$$

e quindi, per ogni $A \in \mathbf{A}$, vale

$$m^*(i_A) = n(i_A) \ .$$

Quindi m^* verifica le equazioni di verosimiglianza e di conseguenza è la stima di massima verosimiglianza. Questo poteva essere un modo alternativo per concludere la dimostrazione precedente.

4.6 Modelli decomponibili

Definizione 4.4 *Un* modello decomponibile *è un modello grafico il cui grafo di interazione è debolmente decomponibile.*

Consideriamo un modello grafico e il suo grafo di interazione $\mathbb{G} = (V, E)$. Supponiamo che esso sia debolmente decomponibile.

Come sappiamo dalla proposizione 1.8 è possibile numerare le cliques di $\mathbb{G}$ in una sequenza debolmente perfetta $C_1, \ldots, C_k$.

Inoltre, per ogni $1 < j \leq k$, si ha che (H_{j-1}, R_j, S_j) sono una decomposizione debole di $\mathbb{G}$. Ricordiamo che:

$$H_{j-1} = \cup_{i=1}^{j-1} C_j, \quad R_j = C_j \setminus H_{j-1}, \quad S_j = C_j \cap H_{j-1} \ .$$

Allora vale la seguente proposizione.

Proposizione 4.5 *Consideriamo un modello grafico e supponiamo che il suo grafo di interazione $\mathbb{G}$ sia debolmente decomponibile. Sia $C_1, \ldots, C_k$ la sequenza debolmente perfetta costituita da tutte e sole le cliques di $\mathbb{G}$. Allora, ricordando che:*

$$S_j = H_{j-1} \cap C_j$$

per ogni $i \in I$ vale:

$$p(i) = \frac{\prod_{j=1}^{k} p(i_{C_j})}{\prod_{j=2}^{k} p(i_{S_j})} \ .$$

Dimostrazione: Dimostriamo la tesi per induzione sul numero delle cliques k.

Se $k = 2$ allora $(C_1, C_2 \setminus C_1, C_1 \cap C_2)$ sono una decomposizione debole di $\mathbb{G}$. Dato che p gode della proprietà di Markov globale vale che per ogni $i \in I$:

$$p(i) = \frac{p(i_{C_1})p(i_{C_2})}{p(i_{C_1 \cap C_2})} \ .$$

Supponiamo che la tesi sia vera per ogni $2 \leq k \leq n-1$. Sia $k = n$.

Allora $(\cup_{j=1}^{n-1} C_j, C_n \setminus \cup_{j=1}^{n-1} C_j, C_n \cap \left(\cup_{j=1}^{n-1} C_j\right))$ sono una decomposizione debole di $\mathbb{G}$.

Abbiamo quindi che per ogni $i \in I$:

$$p(i) = \frac{p(i_{\cup_{j=1}^{n-1} C_j})p(i_{C_n})}{p(i_{C_n \cap \left(\cup_{j=1}^{n-1} C_j\right)})} \ .$$

Il sottografo $\mathbb{G}_{\cup_{j=1}^{n-1} C_j}$ è costituito, per come è definito, da $n-1$ cliques. È possibile applicare ad esso l'ipotesi induttiva e si ottiene quindi che:

$$p(i_{\cup_{j=1}^{n-1} C_j}) = p(i) = \frac{\prod_{j=1}^{n-1} p(i_{C_j})}{\prod_{j=2}^{n-1} p(i_{S_j})} \ .$$

Infatti all'interno di questo sottografo $\cup_{j=1}^{n-1} C_j$ è uguale all'insieme di tutti i vertici.

Componendo le due formule otteniamo:

$$p(i) = \frac{p(i_{C_n}) \prod_{j=1}^{n-1} p(i_{C_j})}{p(i_{S_n}) \prod_{j=2}^{n-1} p(i_{S_j})} = \frac{\prod_{j=1}^{n} p(i_{C_j})}{\prod_{j=2}^{n} p(i_{S_j})} \ .$$

□

È possibile riscrivere la formula in termini del vettore dei valori attesi, ottenendo la seguente formula:

$$m(i) = \frac{\prod_{j=1}^{k} m(i_{C_j})}{\prod_{j=2}^{k} m(i_{S_j})} \ .$$

Oppure in alternativa indicando con $\mathbb{C}$ l'insieme di tutte le clique del grafo e con $\mathbb{S}$ l'insieme di tutti i separatori della sequenza perfetta:

$$m(i) = \frac{\prod_{C \in \mathbb{C}} m(i_C)}{\prod_{S \in \mathbb{S}} m(i_S)^{\nu(S)}} \tag{4.1}$$

dove si è indicato con $\nu(S)$ la molteplicità con cui il separatore S compare nella sequenza perfetta.

Questa formula vale per tutti i modelli decomponibili, indipendentemente dalla distribuzione della tabella dei conteggi.

4.7 Stima di massima verosimiglianza in modelli decomponibili

Vediamo ora come si comporta lo stimatore di massima verosimiglianza per il vettore delle medie in un modello decomponibile.

Proposizione 4.6 *Consideriamo un modello decomponibile $M(\boldsymbol{A})$ e il suo grafo di interazione $\mathbb{G} = (V, E)$. Sia (B, C, D) una decomposizione debole di $\mathbb{G}$. Sia $\hat{M}$ lo stimatore di massima verosimiglianza per il vettore delle medie. Allora per ogni $i \in I$:*

$$\hat{M}(i) = \frac{\hat{M}_{B\cup D}(i_{B\cup D})\hat{M}_{C\cup D}(i_{C\cup D})}{N(i_D)}$$

dove $\hat{M}_{B\cup D}$ e $\hat{M}_{C\cup D}$ sono gli stimatori di massima verosimiglianza nei sottomodelli con grafo di interazione $\mathbb{G}_{B\cup D}$ e $\mathbb{G}_{C\cup D}$.

Dimostrazione: Sappiamo che $\mathbf{A}$ è l'insieme delle cliques di $\mathbb{G}$. Sia $\mathbf{B}$ l'insieme delle cliques di $\mathbb{G}_{B\cup D}$ e $\mathbf{C}$ l'insieme delle cliques di $\mathbb{G}_{C\cup D}$. Ovviamente:

$$\mathbf{A} \subseteq \mathbf{B} \cup \mathbf{C}$$

e inoltre:

$$\mathbb{H}_{\mathbf{A}} = \mathbb{H}_{\mathbf{B}\cup\mathbf{C}} \ .$$

Infatti $\mathbf{A}$ è ottenuto da $\mathbf{B} \cup \mathbf{C}$ eliminando gli elementi che non sono massimali rispetto all'inclusione. Definiamo il seguente vettore aleatorio:

$$M^*(i) = \frac{\hat{M}_{B\cup D}(i_{B\cup D})\hat{M}_{C\cup D}(i_{C\cup D})}{N(i_D)} \ .$$

Allora dato che $\hat{M}_{B\cup D} \in M(\mathbf{B})$ e $\hat{M}_{C\cup D} \in M(\mathbf{C})$ si ha che:

$$M^* \in M(\mathbf{A}) \ .$$

Poiché l'insieme D è contenuto sia in $M(\mathbf{B})$ sia in $M(\mathbf{C})$ ed è completo, per ogni $i \in I$ si ha:

$$\hat{M}_{B\cup D}(i_D) = \hat{M}_{C\cup D}(i_D) = N(i_D) \ .$$

Sia L una clique del grafo, $L \in \mathbf{A}$. Allora necessariamente $L \subseteq B \cup D$ oppure $L \subseteq C \cup D$.

Supponiamo ad esempio che $L \subseteq B \cup D$. Allora:

$$M^*(i_L) = \sum_{k \in I: k_L = i_L} \frac{\hat{M}_{B\cup D}(k_{B\cup D})\hat{M}_{C\cup D}(k_{C\cup D})}{N(k_D)} =$$

$$\sum_{j_{B\cup D} \in I_{B\cup D}: j_L = i_L} \sum_{k \in I: k_{B\cup D} = j_{B\cup D}} \frac{\hat{M}_{B\cup D}(k_{B\cup D})\hat{M}_{C\cup D}(k_{C\cup D})}{N(k_D)} =$$

$$\sum_{j_{B\cup D} \in I_{B\cup D}: j_L = i_L} \sum_{k \in I: k_{B\cup D} = j_{B\cup D}} \frac{\hat{M}_{B\cup D}(k_{B\cup D})\hat{M}_{C\cup D}(k_{C\cup D})}{\hat{M}_{C\cup D}(k_D)} =$$

$$\sum_{j_{B\cup D} \in I_{B\cup D}: j_L = i_L} \frac{\hat{M}_{B\cup D}(j_{B\cup D})}{\hat{M}_{C\cup D}(j_D)} \sum_{k \in I: k_{B\cup D} = j_{B\cup D}} \hat{M}_{C\cup D}(k_{C\cup D}) =$$

$$\sum_{j_{B\cup D} \in I_{B\cup D}: j_L = i_L} \frac{\hat{M}_{B\cup D}(j_{B\cup D})}{\hat{M}_{C\cup D}(j_D)} \hat{M}_{C\cup D}(j_D) =$$

$$\sum_{j_{B\cup D} \in I_{B\cup D}: j_L = i_L} \hat{M}_{B\cup D}(j_{B\cup D}) = \hat{M}_{B\cup D}(j_L) = N(i_L) \ .$$

Quindi M^* soddisfa le equazioni di verosimiglianza e pertanto è lo stimatore di massima verosimiglianza per il vettore delle medie. □

Quindi se il modello è decomponibile è possibile ottenere la stima di massima verosimiglianza del vettore delle medie combinando opportunamente le stime di massima verosimiglianza di sottomodelli più piccoli.

È possibile dare una formula più generale, che coinvolge tutte le clique del grafo di interazione del modello.

Proposizione 4.7 *Consideriamo un modello decomponibile $M(\boldsymbol{A})$ e il suo grafo di interazione $\mathbb{G} = (V, E)$. Sia $\mathbb{C}$ l'insieme di tutte le cliques del grafo $\mathbb{G}$.*

Allora le cliques del grafo di interazione possono essere ordinate a formare una sequenza perfetta e, indicando con $\mathbb{S}$ l'insieme di tutti i separatori della sequenza, lo stimatore di massima verosimiglianza $\hat{M}$ del vettore delle medie può essere scritto come:

$$\hat{M}(i) = \frac{\prod_{C \in \mathbb{C}} N(i_C)}{\prod_{S \in \mathbb{S}} N(i_S)^{\nu(S)}}$$

dove $\nu(S)$ indica la molteplicità con cui il separatore S compare nella sequenza perfetta.

Dimostrazione: Abbiamo già visto, nella proposizione 1.8, che, essendo il grafo di interazione $\mathbb{G}$ debolmente decomponibile, è possibile ordinare le sue cliques a formare una sequenza perfetta.

Dimostriamo la tesi per induzione sul numero di elementi k di tale sequenza, cioè sul numero di cliques del grafo di interazione.

Supponiamo che $k = 2$. Il grafo $\mathbb{G}$ ha quindi solo due cliques: C_1 e C_2. Inoltre, come abbiamo visto nella proposizione 1.5, $(C_1, C_2 \setminus C_1, C_2 \cap C_1)$ sono una decomposizione debole di $\mathbb{G}$.

Indichiamo con $\hat{m}_1$ e con $\hat{m}_2$ le stime di massima verosimiglianza dei modelli con classe generatrice $\{C_1\}$ e $\{C_2\}$.

Utilizzando la proposizione 4.6, si ottiene che, indicando con $\hat{M}$ lo stimatore di massima verosimiglianza del vettore delle medie nel modello completo, esso è uguale a:

$$\hat{M}(i) = \frac{\hat{M}_1(i_{C_1})\hat{M}_2(i_{C2})}{N(i_{C_2 \cup C_1})} \ .$$

Ma i modelli con classe generatrice $\{C_1\}$ e $\{C_2\}$ sono entrambi saturi, essendo gli insiemi C_1 e C_2 completi. Quindi abbiamo che per ogni $i \in I$:

$$\hat{M}_1(i_{C_1}) = N(i_{C_1}) \quad \text{e} \quad \hat{M}_2(i_{C_2}) = N(i_{C_2}) \ .$$

Combinando le due formule otteniamo che:

$$\hat{M}(i) = \frac{N(i_{C_1})N(i_{C_2})}{N(i_{C_2 \cup C_1})} \ .$$

Supponiamo che la tesi sia vera per ogni $2 \leq k \leq n$. Sia $k = n + 1$.

Sia $C_1, \ldots, C_{n+1}$ la sequenza perfetta delle cliques di $\mathbb{G}$. Come abbiamo visto nella proposizione 1.5, $(\cup_{i=1}^{n} C_i, C_{n+1} \setminus (\cup_{i=1}^{n} C_i), C_{n+1} \cap (\cup_{i=1}^{n} C_i))$ sono una decomposizione debole di $\mathbb{G}$.

Indichiamo con $\hat{M}_1$ lo stimatore di massima verosimiglianza del modello con classe generatrice $\{C_1, \ldots, C_n\}$ e con $\hat{M}_2$ lo stimatore di massima verosimiglianza del modello con classe generatrice $\{C_{n+1}\}$.

Utilizzando la proposizione 4.6 si ottiene che, indicando con $\hat{M}$ lo stimatore di massima verosimiglianza del vettore delle medie nel modello completo, esso è uguale a:

$$\hat{M}(i) = \frac{\hat{M}_1(i_{\cup_{i=1}^{n} C_i})\hat{M}_2(i_{C_{n+1}})}{N(i_{S_{n+1}})}$$

ricordando ovviamente che $S_{n+1} = C_{n+1} \cap (\cup_{i=1}^{n} C_i)$.

Il sottomodello con classe generatrice $\{C_1, \ldots, C_n\}$ ha solo n cliques. È possibile applicare ad esso l'ipotesi induttiva e si ottiene quindi che:

$$\hat{M}_1(i_{\cup_{i=1}^{n} C_i}) = \frac{\prod_{j=1}^{n} N(i_{C_j})}{\prod_{j=2}^{n} N(i_{S_j})} \ .$$

Il sottomodello con classe generatrice $\{C_{n+1}\}$ è saturo e quindi:

$$\hat{M}_2(i_{C_{n+1}}) = N(i_{C_{n+1}}) \ .$$

Combinando le due formule si ottiene:

$$\hat{M}(i) = \frac{N(i_{C_{n+1}}) \prod_{j=1}^{n} N(i_{C_j})}{N(i_{S_{n+1}}) \prod_{j=2}^{n} N(i_{S_j})} = \frac{\prod_{j=1}^{n+1} N(i_{C_j})}{\prod_{j=2}^{n+1} N(i_{S_j})} \ .$$

Questa formula è equivalente alla tesi. □

4.8 Riferimenti bibliografici

L'algoritmo IPS, trattato ampiamente nei paragrafi 4.4 e 4.5, è stato tratto dai testi On a Least Squares Adjustment on a Sampled Frequency Table When the Expected Marginal Totals Are Known di Deming e Spephan [3], *Graphical Models* di Lauritzen [7] e *Lectures on Contingency Tables* di Lauritzen [8]. Si è scelto di utilizzare, tranne che negli esempi, la notazione del testo di Lauritzen [7] e di dare ampio spazio all'operazione di aggiustamento delle marginali, su cui l'algoritmo si basa, dando una dimostrazione completa e dettagliata della proposizione 4.3.

Il paragrafo 4.3, relativo alla devianza dei modelli gerarchici è interamente tratto dal testo di Lauritzen [7].

Capitolo 5

Modelli grafici per le frodi bancarie

5.1 Il dataset FRAUD

5.1.1 Introduzione

Con la diffusione dei collegamenti internet le banche hanno istituito piattaforme online che permettono ai clienti di effettuare operazioni bancarie remote. Ciascun utente possiede uno *username* e una *password* con i quali può ottenere un accesso sul server della banca.

Queste nuove funzionalità, tuttavia, fanno nascere un nuovo problema legato alla sicurezza. È infatti possibile, per un hacker, rubare le credenziali di accesso di un cliente e, utilizzandole per collegarsi , sottrarre del denaro alla banca.

Negli ultimi anni, pertanto, si stanno cercando metodi statistici per effettuare un'identificazione automatica delle frodi, in modo da poter bloccare le operazioni sospette ed evitare alla banca la perdita del denaro.

Si tratta di un problema complesso, infatti la mole di dati coinvolti è estremamente elevata e il numero di operazioni fraudolente è molto esiguo in proporzione al totale delle operazioni effettuate.

Quanto è presentato di seguito è uno studio di fattibilità basato sui due modelli grafici di indipendenza che sono stati trattati, da un punto di vista teorico, nei capitoli precedenti: i grafi indiretti e gli Hidden Markov Models.

5.1.2 I dati

Il dataset a disposizione contiene un milione di osservazioni, ordinate in modo cronologico, rilevate nel periodo da gennaio 2011 a maggio 2013 e relative a diversi gruppi bancari. Un cliente può effettuare online 51 tipi di opera-

zioni, codificate con i numeri da 1 a 51 e suddivisibili in dispositive, ovvero con trasferimento di denaro, e non dispositive, ovvero senza trasferimento di denaro.

Il personale addetto delle banche ha verificato che 636 operazioni sono risultate essere frodi; di queste, 585 sono dispositive e 51 non dispositive.

Le operazioni frodi non dispositive possono essere attività propedeutiche all'azione dispositiva, cioè sono operazioni di controllo dati IBAN e disponibilità sul contocorrente. Inoltre, esse possono essere tentativi di frode non andati a buon fine. Ad esempio, se l'importo è troppo alto l'operazione dispositiva ha esito negativo e non viene eseguita; al suo posto viene registrata nel database un'operazione non dispositiva con indicato l'importo.

Le operazioni etichettate come frodi sono concentrate in alcuni punti della sequenza di osservazioni. Ciò avviene perché molte di esse sono svolte in tempi molto ravvicinati tra di loro, in genere pochi minuti, spesso a danno dello stesso utente. Ad esempio, un hacker può tentare di svolgere due volte la stessa operazione, se la prima volta essa dà esito negativo, oppure controllare il conto corrente della vittima prima di effettuare il prelievo di denaro.

Queste peculiarità contribuiscono a rendere spesso deludenti i risultati ottenibili con i normali metodi di classificazione.

Le variabili contenute nel dataset per ciascuna operazione sono le seguenti.

- *Data di log-out*: anno, mese, giorno, ora, minuto e secondo in cui il cliente ha terminato l'operazione.
- *Data di log-in*: anno, mese, giorno, ora, minuto e secondo in cui il cliente ha iniziato l'operazione.
- *IP*: indirizzo IP del cliente.
- *UserAgent*: informazioni sulla versione del browser e del sistema operativo utilizzati da cliente.
- *Username*: identificativo univoco del cliente.
- *Banca*: identificativo della banca dove è stata svolta l'operazione.
- *Contratto*: identificato del contratto sottoscritto dal cliente.
- *Operazione*: identificativo dell'operazione svolta dal cliente.
- *Valuta*: valuta nella quale è stata svolta l'operazione.
- *Importo*: importo dell'operazione in euro.
- *Contocorrente*: identificativo del contocorrente del cliente.

- *Iban destinatario*: codice IBAN del destinatario dell'operazione.
- *Esito*: esisto dell'operazione.
- *Destinatario*: destinatario dell'operazione.

5.1.3 Le variabili di interesse

Per poter avere dei modelli gestibili si è dovuta fare una selezione, tra tutte le variabili osservate nel dataset, di quelle più significative per la caratterizzazione delle frodi.

Dalla differenza delle variabili *Data di log-out* e *Data di log-in* si è ottenuta la nuova variabile *Durata*, che rappresenta la durata in secondi dell'operazione.

Dall'indirizzo *IP* è stato ottenuto il comune in cui sono state svolte le operazioni. Tuttavia si è scelto di non utilizzarlo nelle analisi successive. È infatti molto improbabile che gli hackers che compiono le frodi abbiano una particolare dislocazione geografica. Inoltre il 70 per cento delle operazioni è stata svolta nel comune di Roma e, dovendo suddividere le osservazioni in poche categorie, il raggruppamento più naturale, che consiste nell'unire i comuni diversi da Roma in un'unica classe, non ha nessun significato in ambito della caratterizzazione delle operazioni e delle frodi.

Per le stesse ragioni sono state escluse le variabili *UserAgent* e *Banca*. Esse infatti assumono un amplissimo numero di valori di difficile interpretazione nell'ambito della caratterizzazione delle operazioni.

Si è scelto invece di considerare come variabili di interesse *Operazione*, *Valuta* e *Importo*, che ovviamente caratterizzano l'operazione svolta dal cliente.

Sono considerate inoltre di interesse la data e l'ora in cui l'operazione è stata svolta.

La variabile *Esito*, infine, sembrerebbe essere molto importante e la sua inclusione nel modello parrebbe scontata. Tuttavia non è questo il caso. Infatti le operazioni con esito negativo sono quelle che vengono bloccate dalla banca perché irregolari e potenzialmente frodi. Utilizzando questa variabile nelle analisi, quindi, si rischierebbe di avere una sovra-stima. Per questo motivo l' *Esito* è stato escluso dal modello.

Le variabili così ottenute sono continue oppure discrete ma con un numero di stati infinito o molto elevato. Per stimare un modello grafico o un Hidden Markov Model occorre avere un numero di stati finito e, per problemi computazionali e di interpretazione dei risultati, non eccessivamente grande. Per questo motivo le variabili scelte sono state categorizzate.

Si riportano nel seguito le variabili scelte come interessanti per il problema oggetto di studio, specificando le modalità e la loro distribuzione nel dataset.

5.1.4 Le operazioni

Le operazioni che possono essere effettuate dall'utente sono di due tipi:

dispositive *non dispositive*.

Vi sono operazioni classificate come frodi in entrambe le categorie. Come già detto, le frodi presenti all'interno della classe di operazioni non dispositive sono controlli effettuati dall'hacker prima e dopo la sottrazione del denaro, oppure tentativi di eseguire operazioni che in quel momento non sono effettuabili. Ad esempio, è possibile effettuare bonifici da un paese straniero solo se sono rispettate particolari condizioni.

Le operazioni non dispositive che sono state osservate sono le seguenti:

Controllo dati bonifico, *Verifica bonifico estero*.

Le operazioni dispositive osservate, invece, sono le seguenti:

Bonifico, *Ricarica telefonica*, *Ricarica carta*, *Giroconto*, *Pagamento bollettino*, *Bonifico SEPA*.

Le operazioni non dispositive sono poco significative in quanto non causano movimenti di denaro e, spesso, sono eseguite insieme a operazioni dispositive. Per questo motivo sono state raggruppate in un'unica categoria.

Le frodi e le non frodi sono così ripartite all'interno delle operazioni.

	Frodi	**Non Frodi**	**Totale**
Operazioni non dispositive	51	7542	7593
Bonifico	358	569738	570096
Ricarica telefonica	3	140413	140416
Ricarica carta	90	120445	120535
Giroconto	0	53036	53036
Pagamento bollettino	0	103305	103305
Bonifico SEPA	134	5521	5655

Le operazioni *Giroconto* e *Pagamento bollettino* non sono rappresentate nelle frodi, pertanto sono state raggruppate in un'unica classe.

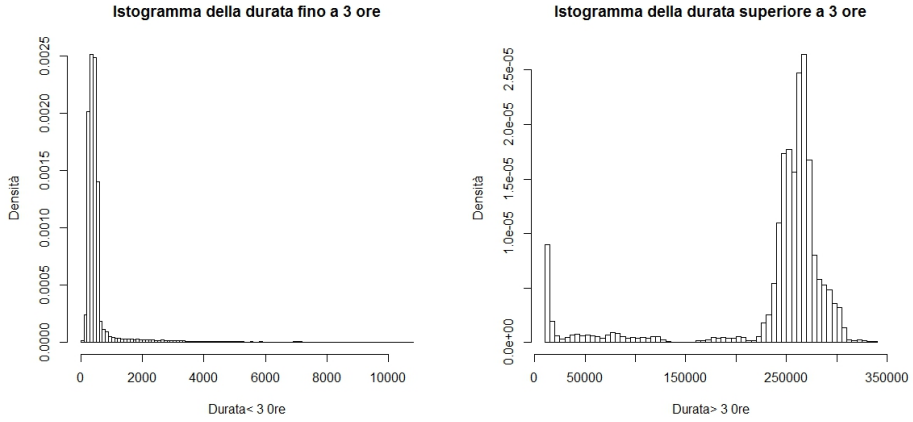
Infine le operazioni *Ricarica telefonica* e *Ricarica carta*, data la loro comune natura, sono state raggruppate in un'unica classe.

Abbiamo quindi ottenuto la seguente classificazione delle operazioni.

CLASSE	OPERAZIONI
0	*Operazioni non dispositive*
1	*Bonifico*
2	*Ricarica telefonica*, *Ricarica carta*
3	*Giroconto*, *Pagamento bollettino*
4	*Bonifico SEPA*

5.1.5 Durata dell'operazione

La durata dell'operazione presenta una distribuzione di tipo bimodale. La maggior parte delle osservazioni presenta una durata inferiore alla mezz'ora, tuttavia esiste un gruppo di osservazioni con durate molto lunghe, alcune anche di diversi giorni. Appare quindi naturale suddividere le osservazioni in due classi: quelle con durata *breve* e quelle con durata *lunga*.



Dall'istogramma si osserva che i valori più frequenti per la durata sono, rispettivamente, nella categoria di osservazioni a durata breve di circa 15 minuti mentre nella categoria di osservazioni a durata lunga di circa 3 giorni. Queste ultime osservazioni sono relative a operazioni effettuate da società che si occupano di trading e che svolgono operazioni durante tutto l'arco della giornata. Si osserva inoltre un nutrito gruppo di osservazioni con una durata leggermente superiore alle tre ore. Il numero di osservazioni con una durata compresa tra un'ora e tre ore è molto basso, solamente 26881.

Si è scelto di classificare a durata breve le osservazioni con durata inferiore o uguale alle tre ore e con durata lunga le osservazioni con durata superiore alle tre ore. In questo modo è stato incluso nella seconda categoria anche il

gruppo di osservazioni, con durata leggermente superiore alle tre ore, che forma un picco nel secondo istogramma.

breve: *Durata*$\leq$ 3 ore.
lunga: *Durata*$>$ 3 ore.

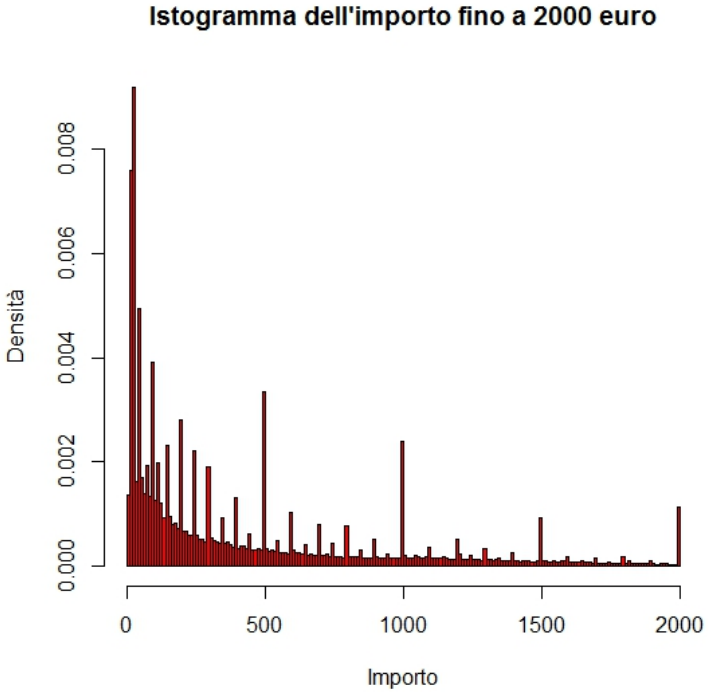
Con questa classificazione si ottiene la seguente suddivisione delle unità del campione.

	Durata $\leq$ 3 ore	*Durata* $>$ 3 ore
Frodi	610	26
Non frodi	922874	77126
Totale	923484	77152

5.1.6 Importo

La distribuzione dell'importo ha un andamento esponenzialmente decrescente e presenta dei picchi nei multipli di 10 euro e di 100 euro. Ciò è dovuto alla propensione degli utenti a svolgere la maggior parte delle operazioni con importi arrotondati alla cifra delle decine per le operazioni fino a 100 euro e alla cifra delle centinaia per operazioni superiori a 100 euro.

Il 75 per cento delle osservazioni ha un importo inferiore o uguale ai 1000 euro. Sono però presenti alcune osservazioni con importi molto superiori, fino a un valore massimo di 9 milioni di euro.



Osserviamo inoltre che:

$$100 = \text{quantile}_{0.3}(\textit{Importo}), \quad 1000 = \text{quantile}_{0.75}(\textit{Importo}) \ .$$

Si è ritenuto quindi ragionevole definire le seguenti categorie:

CLASSE	IMPORTO
basso	$0 - 100$ euro
medio	$101 - 1000$ euro
alto	> 1000 euro

5.1.7 Valuta

Quasi tutte le operazioni sono state svolte nella valuta euro.

Euro	Valuta estera
999511	1125

Tra le poco più di mille operazioni svolte in altre valute si incontrano il dollaro, la sterlina, il franco svizzero e molte altre valute minori. Tra le operazioni frodi, solamente 3 di esse non sono state effettuate in euro.

Vista l'esiguità di operazioni non svolte in euro, è ragionevole raggruppare tutte le altre valute in un'unica categoria.

5.1.8 Orario

Tra le molte possibili classificazioni per la data dell'operazione è sembrato ragionevole suddividere i giorni in lavorativi e festivi. Si considerano giorni festivi le domeniche e le festività nazionali. Gli altri giorni vengono considerati come lavorativi.

La maggior parte delle operazioni sono state svolte in giorni lavorativi. Tra le frodi, solamente 10 sono avvenute in giorni festivi.

Giorno lavorativo	**Giorno festivo**
895278	105358

Per quanto riguarda l'ora, è sembrato ragionevole considerare come orario lavorativo dalle 8 : 00 alle 17 : 59.

Combinando opportunamente la data e l'ora, si ottiene un'unica variabile *Orario*, che assume i seguenti valori.

Classe	*Orario*
0	$8:00-17:59$ di un giorno lavorativo
1	$18:00-7:59$ di un giorno lavorativo, giorno festivo

5.1.9 Sintesi delle variabili scelte

La seguente tabella fornisce una rassegna finale sulle variabili categoriali costruite nei paragrafi precedenti.

Variabile	Valore Codifica				
Operazione	non dispositiva 0	bonifico 1	ric. telefonica ric. carta 2	giroconto pag. bollettino 3	bonifico SEPA 4
Valuta	euro 0	non euro 1			
Importo	basso 0	medio 1	alto 2		
Durata	breve 0	lunga 1			
Orario	lavorativo 0	non lavorativo 1			
Frode	non frode 0	frode 1			

Alle variabili costruite nei paragrafi precedenti abbiamo aggiunto la variabile *Frode*, binaria, che indica se l'osservazione è una frode oppure no. A seconda del modello utilizzato per la stima, questa variabile potrà essere considerata osservata oppure nascosta.

Il numero di stati possibili è il prodotto delle dimensioni delle variabili che vengono considerate, cioè, includendo anche la variabile *Frode*:

$$5 \times 2 \times 3 \times 2 \times 2 \times 2 = 240 \ .$$

5.2 Grafo indiretto

Costruiamo una tabella di contingenza i cui criteri di classificazione sono le variabili che abbiamo selezionato e la variabile *Frode*, che assume valore 1 se l'osservazione è una frode e valore 0 altrimenti.

La tabella di contingenza così costruita ha 6 criteri di classificazione e assume valori nell'insieme

$$I = \{0,1,2,3,4\} \times \{0,1\}^4 \times \{0,1,2\} \ .$$

La dimensione di I è uguale al prodotto delle dimensioni dei criteri di classificazione scelti, cioè :

$$\mid I \mid = N = 5 \times 2 \times 3 \times 2 \times 2 \times 2 = 240 \ .$$

Denotiamo con V l'insieme di tutti i criteri di classificazione, cioè:

$$V = \{Operazione, Valuta, Importo, Durata, Orario, Frode\} .$$

Per il tipo di problema studiato possiamo fare due tipi di assunzioni di indipendenza condizionata. Possiamo ragionalmente supporre che la variabile *Frode* determini in modo causale le altre variabili. Pertanto supponiamo che, date due generiche altre variabili Y_i e Y_j, esse siano condizionatamente indipendenti alle variabili *Frode*, cioè:

$$Y_i \perp\!\!\!\perp Y_j \mid Frode .$$

Un'altra possibilità ragionevole è supporre che siano congiuntamente le variabili *Frode* e *Operazione* a determinare in modo causale le altre variabili. In questo caso abbiamo che, date due generiche altre variabili Y_i e Y_j:

$$Y_i \perp\!\!\!\perp Y_j \mid Frode, Operazione .$$

Otteniamo quindi due possibili grafi, $\mathbb{G}_1 = (V, E_1)$ e $\mathbb{G}_2 = (V, E_2)$. Entrambi i grafi hanno lo stesso insieme di vertici, costituito dall'insieme V precedentemente definito.

Il primo dei due grafi, $\mathbb{G}_1 = (V, E_1)$, ha il seguente insieme di archi:

$$E_1 = \{\{Operazione, Frode\}, \{Valuta, Frode\}, \{Importo, Frode\}, \{Durata, \text{Frode}\}, \{Orario, Frode\}\} .$$

Il secondo grafo, $\mathbb{G}_2 = (V, E_2)$, ha invece il seguente insieme di archi:

$$E_2 = \{\{Operazione, Frode\}, \{Valuta, Frode\}, \{Importo, Frode\}, \{Durata, Frode\}, \{Orario, Frode\}, \{Operazione, Valuta\}, \{Operazione, Importo\}, \{Operazione, Durata\}, \{Operazione, Orario\}\} .$$

Entrambi i grafi sono debolmente decomponibili. Denotiamo con $\mathbf{C}_1$ l'insieme delle cliques del grafo $\mathbb{G}_1$ e con $\mathbf{C}_2$ l'insieme delle cliques del grafo $\mathbb{G}_2$:

$$\mathbf{C}_1 = \{\{Operazione, Frode\}, \{Valuta, Frode\}, \{Importo, Frode\}, \{Durata, Frode\}, \{Orario, Frode\}\} ,$$

$$\mathbf{C}_2 = \{\{Operazione, Frode, Valuta\}, \{Operazione, Frode, Importo\}, \{Operazione, Frode, Durata\}, \{Operazione, Frode, Orario\}\} .$$

Il modello grafico $M(\mathbf{C}_1)$ è un sottomodello del modello grafico $M(\mathbf{C}_2)$.

Supponiamo di aver osservato un campione $n = \{n(i)\}_{i \in I}$, formato da osservazioni di cui sappiamo se sono delle frodi, e di aver trovato, utilizzando uno dei due modelli scelti sopra, la stima di massima verosimiglianza dei conteggi attesi. Sia questa stima:

$$\hat{m} = \{\hat{m}(i)\}_{i \in I} .$$

Sia

$$y = (y_0, y_1, y_2, y_3, y_4)$$

una nuova osservazione, di cui non sappiamo se è una frode.

Possiamo utilizzare la stima di massima verosimiglianza $\hat{m}$ per classificare l'osservazione y. Sappiamo infatti che, per il teorema di Bayes, la probabilità che y sia una frode è la seguente:

$$p_y = \mathbb{P}\left[Frode = 1 \mid Operazione = y_0, Valuta = y_1, Importo = y_2, Durata = y_3, Orario = y_4\right] = \frac{\mathbb{P}\left[Frode = 1, \ldots, Orario = y_4\right]}{\mathbb{P}\left[Operazione = y_0, \ldots, Orario = y_4\right]} = \frac{\mathbb{P}\left[Frode = 1, \ldots, Orario = y_4\right]}{\mathbb{P}\left[Frode = 0, \ldots, Orario = y_4\right] + \mathbb{P}\left[Frode = 1, \ldots, Orario = y_4\right]} .$$

Possiamo quindi stimare la probabilità p_y con:

$$\hat{p}_y = \frac{\hat{m}_{1,y_0,\ldots,y_4}}{\hat{m}_{1,y_0,\ldots,y_4} + \hat{m}_{0,y_0,\ldots,y_4}} \tag{5.1}$$

e classificare y come frode se $\hat{p} > 0.5$.

Allo stesso modo, se chiamiamo A il sottoinsieme di V che non contiene la variabile frode, cioè:

$$A = \{Operazione, Valuta, Importo, Durata, Orario\}$$

per ogni $i_A \in A$ possiamo definire la probabilità di frode condizionata a i_A come

$$\mathbb{P}[Frode = 1 \mid i_A] = \frac{\mathbb{P}[Frode = 1, i_A]}{\mathbb{P}[Frode = 0, i_A] + \mathbb{P}[Frode = 1, i_A]} .$$

In questo modo posssiamo trovare gli stati che, se osservati, hanno un'alta probabilità di frode.

5.2.1 Selezione automatica del modello

Accanto ai due modelli grafici $M(\mathbf{C}_1)$ e $M(\mathbf{C}_2)$, scelti sulla base di considerazioni sulla natura delle variabili utilizzate, può essere opportuno affiancare il modello grafico che meglio si adatta all'insieme di stima, ottenuto mediante una procedura di ricerca automatica.

Come criterio di selezione tra due diversi modelli grafici si utilizza il criterio AIC. Dati un modello grafico $M(\mathbf{C})$ e lo stimatore di massima verosimiglianza del vettore delle medie $\hat{m}$ per tale modello, il coefficiente AIC_M è così definito:

$$\mathrm{AIC}_M = -2\log\left(\mathrm{L}(\hat{m})\right) + 2\dim(\mathbb{H}_{\mathbf{C}}) \ .$$

Dati due modelli grafici M_1 e M_2 si seleziona il modello M_i per cui vale:

$$\mathrm{AIC}_{M_i} < \mathrm{AIC}_{M_j} \ .$$

Per la selezione del miglior modello si utilizza la procedura *forward stepwise*, che è così definita.

Procedura 5.1 *Procedura forward-stepwise:*

- *Passo 1: Si stima come modello iniziale il modello che assume l'indipendenza tra tutti i criteri di classificazione, cioè il modello grafico il cui grafo di interazione non ha nessun arco* $M_0 = M(\boldsymbol{C}_0)$*, dove* $\boldsymbol{C}_0 = \{\{Y\} : \quad Y \in V\}$.

- *Passo n: Si considera il modello grafico* M_{n-1} *con grafo di interazione* $\mathbb{G}_{n-1} = (V, E_{n-1})$ *ottenuto al passo precedente. Tra tutti i possibili archi* $\{\alpha, \beta\} \notin E_{n-1}$ *si aggiunge al modello quello per cui il modello grafico con grafo di interazione* $(V, E_{n-1} \cup \{\{\alpha, \beta\}\})$ *ha coefficiente AIC più basso.*

- *Passo finale: ci si ferma quando, per ogni possibile arco* $\{\alpha, \beta\} \notin E_{n-1}$*, il modello grafico con grafo di interazione* $(V, E_{n-1} \cup \{\{\alpha, \beta\}\})$ *ha coefficiente AIC maggiore del coefficiente* $AIC_{M_{n-1}}$*, oppure quando si raggiunge il modello saturo.*

5.2.2 Programma Grafo

Per la stima del modello grafico è stato realizzato il programma `Grafo` (in appendice) in linguaggio C++, scelto in quanto garantisce maggiore efficienza e velocità di esecuzione ed è quindi adatto per svolgere operazioni computazionalmente onerose.

Le funzionalità del programma sono le seguenti.

1. Estrazione di un insieme di stima e un insieme di verifica dal dataset dei dati.

2. Stima dei modelli grafici $M(\mathbf{C}_1)$ e $M(\mathbf{C}_2)$ utilizzando il metodo esatto.

3. Partendo da un generico grafo $\mathbb{G}$, estrazione automatica delle cliques e stima del modello grafico corrispondente utilizzando l'algoritmo IPS.

4. Identificazione, tramite la procedura *forward-stepwise*, del modello grafico ottimale per l'insieme di stima.

5. Classificazione delle osservazioni degli insiemi di stima e verifica, calcolo della matrice di confusione e identificazione degli stati con alta probabilità di frode.

Per la validazione degli algoritmi implementati si sono confrontati i risultati ottenuti utilizzando il metodo esatto e l'algoritmo IPS, verificando che fossero identici.

5.2.3 Stima

Definiamo α, o probabilità di errore di primo tipo, la probabilità di classificare una frode in modo scorretto. Definiamo invece β, o probabilità di errore di secondo tipo, la probabilità di classificare in modo scorretto una non frode, cioè:

$$\alpha = \mathbb{P}\left[\hat{p}_Y = 0 \mid \textit{Frode} = 1\right], \quad \beta = \mathbb{P}\left[\hat{p}_Y = 1 \mid \textit{Frode} = 0\right] .$$

Per poter dare una caratterizzazione delle frodi occorre scegliere un insieme di stima dove esse siano rappresentate in modo adeguato. Dato che le osservazioni sono ordinate temporalmente e, come già detto, le frodi sono spesso consecutive, è inopportuno estrarre come insieme di stima una sequenza di osservazioni casuali e temporalmente distanti.

Pertanto scegliamo come primo insieme di stima una sequenza di 2000 osservazioni consecutive in cui è presente un'alta concentrazione di frodi, così ripartite.

Campione di 2000 osservazioni	
Frodi	**Non frodi**
132	1868

All'interno del campione le frodi sono concentrate all'inizio, dalla prima alla 132-esima osservazione.

La stima di un modello grafico è computazionalmente poco onerosa. La complessità computazionale non dipende infatti dalla dimensione del campione ma solamente dalla dimensione della tabella di contingenza. Inoltre, anche quando non sono disponibili metodi esatti, la convergenza dell'algoritmo IPS è immediata e, in tutti i casi che sono stati sperimentati, è avvenuta entro la decima iterazione.

Vengono stimati entrambi i modelli grafici $M(\mathbf{C}_1)$ e $M(\mathbf{C}_2)$. Dato che il modello grafico $M(\mathbf{C}_1)$ è un sottomodello di $M(\mathbf{C}_2)$, è possibile eseguire un test sulle devianze per verificare l'effettiva diversità tra i due modelli.

La log-verosimiglianza dei due modelli è la seguente.

Modello 1		**Modello 2**	
Log-verosimiglianza	**Gradi di libertà**	**Log-verosimiglianza**	**Gradi di libertà**
9923.97	50	9458.98	18

Il valore campionario della statistica test, che sotto l'ipotesi di uguaglianza dei due modelli ha una distribuzione χ^2 con 32 gradi di libertà, è:

$$d = 929.98 \quad p - \text{value} = 0 \ .$$

Si rifiuta quindi l'ipotesi di uguaglianza tra i due modelli.

Utilizzando invece sull'insieme di stima la procedura 5.1, *forward stepwise*, si ottiene che il modello grafico che meglio si adatta ai dati ha come grafo di interazione $\mathbb{G} = (V, E)$ con:

$$E = \{\{Operazione, Importo\}, \{Operazione, Durata\}, \{Operazione, Orario\}, \{Valuta, Durata\}, \{Importo, Durata\}, \{Frode, Valuta\}, \{Frode, Importo\}, \{Frode, Orario\}\} \ .$$

Si nota subito la mancanza dell'arco $\{Operazione, Frode\}$, che indica come, in questo modello grafico, le variabili *Frode* e *Operazione* sono condizionatamente indipendenti.

L'insieme delle cliques del grafo $\mathbb{G}$ è:

$$\mathbf{C} = \{\{Operazione, Orario\}, \{Valuta, Durata\}, \{Frode, Valuta\}, \{Frode, Importo\}, \{Frode, Orario\}, \{Operazione, Importo, Durata\}\} \ .$$

Nonostante i modelli grafici siano risultati distinti, la classificazione ottenuta risulta identica per tutti e tre i modelli, ed è riportata nella seguente tabella.

Matrice di confusione			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	127	0	
Frode=0	5	1868	
Totale	132	1868	2000
α	0.0378788		
β	0		
errore totale	0.0025		

Utilizzando il classificatore bayesiano definito nell'equazione 5.1 possiamo determinare gli stati che, se osservati, hanno un'alta probabilità di essere frodi. Sia A il sottoinsieme di V che non contiene la variabile frode. Utilizzando il modello grafico $\mathbb{G}$, ottenuto con la procedura *forward-stepwise*, otteniamo il seguente risultato.

Stati con alta probabilità di frode						
Stato $i_A \in \boldsymbol{I}_A$					**Risultati**	
Operazione	*Valuta*	*Importo*	*Durata*	*Orario*	$\mathbb{P}[Frode = 1 \mid i_A]$	**Frodi attese**
1	0	2	0	0	1	106.0.36
4	0	2	0	0	1	9.91744
0	0	2	0	0	1	9
1	0	2	1	0	1	1.09881
1	0	1	0	0	1	0.865333
4	0	1	0	0	1	0.0831275

Utilizzando i modelli $M(\mathbf{C}_2)$ e $M(\mathbf{C}_1)$ si ottengono risultati molto simili per gli stati con frodi attese molto alte. Utilizzando questi ultimi due modelli, tuttavia, i risultati sono, su questo tipo di analisi, meno precisi. Infatti si trovano molti altri stati con probabilità di frode uguale a 1 o molto vicina ad 1, ma con frodi attese molto piccole, prossime allo 0. Per questo motivo per questo tipo di analisi è preferibile utilizzare il modello grafico ottenuto con la procedura *forward-stepwise*.

Nella tabella spicca in modo particolare la prima riga, dove si ha una probabilità di frode uguale a 1 e più di 100 frodi attese.

5.2.4 Interazione fra le variabili

In generale, il modello grafico ottenuto con la procedura *forward-stepwise* dipende dal campione utilizzato. Tuttavia, alcune caratteristiche sono comuni a tutti i campioni.

Selezionando un insieme di stima nel quale non sono presenti frodi il modello grafico ottenuto con la procedura *forward-stepwise* ha sempre lo stesso grafo di interazione $\mathbb{G} = (V, E)$ con:

$$E = \{\{Operazione, Valuta\}, \{Operazione, Importo\}, \{Operazione, Durata\}, \{Operazione, Orario\}, \{Operazione, Frode\}\} \ .$$

Ciò significa che, nelle operazioni oneste, tutte le variabili sono tra di loro indipendenti condizionatamente all'operazione svolta.

Nei campioni in cui sono presenti le frodi, invece, il modello ottenuto con la procedura *forward-stepwise* risulta ogni volta differente.

Per cercare di dare una caratterizzazione alle frodi si è eseguita la procedura *forward-stepwise* utilizzando come campione l'insieme di tutte e sole le osservazioni etichettate come frodi. Il grafo di interazione ottenuto è $\mathbb{G} = (V, E)$ con il seguente insieme di archi.

$$E = \{\{Operazione, Valuta\}, \{Operazione, Importo\}, \{Operazione, Durata\}, \{Operazione, Orario\}, \{Valuta, Orario\}, \{Valuta, Frode\}, \{Importo, Orario\}, \{Importo, Frode\}\} \ .$$

Si osserva in particolare che la variabile *Operazione* è adiacente a tutte le altre variabili ad eccezione della *Frode*: $Operazione \not\sim Frode$.

Ciò significa che l'*Operazione* è indipendente alla variabile *Frode* condizionatamente alle altre variabili e, quindi, non esiste un'operazione privilegiata per l'hacker che commette frodi.

La variabile *Frode*, invece, è adiacente nel grafo alle sole variabili *Valuta* e *Importo*:

$$Frode \sim Valuta, \quad Frode \sim Importo \ .$$

5.2.5 Validazione

Utilizziamo adesso la stima del modello grafico per predire la variabile *Frode* su un insieme di verifica. Consideriamo, come insieme di verifica, le successive 5000 osservazioni alle 2000 scelte per la stima. Ci aspettiamo che esse, essendo temporalmente vicine all'insieme di stima, abbiano una analoga caratterizzazione delle frodi e che, quindi, sia possibile classificarle con un errore molto basso.

L'insieme di verifica così scelto presenta una totale assenza di frodi. Come prima, i tre diversi modelli grafici forniscono un'identica classificazione.

Matrice di confusione sull'insieme di verifica			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	0	0	
Frode=0	0	5000	
Totale	0	5000	5000
α	0		
β	0		
errore totale	0		

Utilizzando come insieme di verifica osservazioni temporalmente vicine a quelle usate per la stima si è osservato che gli errori di classificazione sono molto bassi e spesso sono 0. Quindi possiamo dire che il grafo si dimostra un buon classificatore in questa situazione.

Ma cosa succede se si utilizza un insieme di verifica temporalmente distante dall'insieme di stima?

Se il processo generatore dei dati fosse stazionario e, quindi, le frodi avessero la stessa caratterizzazione in tutti gli istanti temporali, ci si aspetterebbe, una volta scelto un insieme di stima opportunamento bilanciato, di avere comunque una buona classificazione.

È stato quindi scelto un insieme di verifica, sempre costituito di 5000 osservazioni consecutive non frodi, distante temporalmente all'insieme di stima. Il risultato ottenuto è il seguente.

Matrice di confusione sull'insieme di verifica			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	0	3105	
Frode=0	0	1895	
Totale	0	5000	5000
α	0		
β	0.621		
errore totale	0.621		

Risultati analoghi su altri insiemi di dati hanno indotto la seguente conclusione.

Osservazione: Il processo generatore dei dati sembra non essere stazionario ma ha un'evoluzione lenta e, quindi, in intervalli temporali brevi, inferiori alle 10000 osservazioni, è approssimabile con un processo stazionario.

Non esiste un profilo medio del frodatore perché il suo profilo evolve nel tempo.

Per lo stesso motivo, è possibile utilizzare un modello grafico come classificatore solo per osservazioni temporalmente vicine a quelle usate per la stima; modelli grafici stimati su sequenze di osservazioni troppo lunghe o su sequenze estratte casualmente, appartenenti a intervalli temporali diversi, non producono buoni risultati.

5.2.6 Il problema degli stati non osservati

Poiché occorre utilizzare per la stima osservazioni temporalmente vicine, spesso accade che i campioni utilizzati siano incompleti. Le osservazioni, cioè, non

coprono tutti i possibili stati ma solamente una parte di essi. Se nell'insieme di verifica sono presenti osservazioni che assumono valori che non sono coperti dall'insieme di stima, la loro classificazione diventa impossibile. Vediamo un esempio.

Consideriamo un insieme di stima costituito da 2000 osservazioni, non frodi, e stimiamo il modello grafico. Consideriamo un insieme di verifica di 2000 osservazioni immediatamente successive, all'interno del quale è presente una concentrazione di 139 frodi. Otteniamo la seguente classificazione.

Matrice di confusione sull'insieme di verifica			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	0	28	
Frode=0	0	1861	
Totale	0	1889	1889
impossibile classificare 111 osservazioni			

Nell'insieme di verifica sono presenti 111 frodi che assumono, all'interno di I_A, stati che non sono coperti da nessuna osservazione nell'insieme di stima. Questi stati rappresentano il profilo del frodatore in quel determinato intervallo temporale e, dato che le frodi sono tutte concentrate nell'insieme di verifica, temporalmente successivo a quello di stima, la classificazione diventa impossibile.

5.2.7 Grafo: vantaggi e svantaggi

Il grafo si dimostra quindi un buon classificatore ma presenta alcuni problemi che riducono di molto la sua effettiva utilità.
Vantaggi

1. La stima del grafo è computazionalmente poco onerosa e la complessità computazionale non dipende dal numero di osservazioni presenti nell'insieme di stima.

2. Se l'insieme di verifica è temporalmente vicino all'insieme di stima e l'insieme di stima copre tutti gli stati assunti dalle osservazioni nell'insieme di verifica, il grafo predice con un errore molto basso e quindi è un ottimo classificatore.

3. Anche se i grafi stimabili sono molteplici, la classificazione è ottima indipendentemente da quello effettivamente scelto.

4. Osservando le probabilità stimate delle varie celle della tabella di contingenza si ottengono subito gli stati che sono ad alto rischio frode.

Svantaggi

1. Su grandi insiemi di dati il grafo risulta un pessimo classificatore e, quindi, occorre utilizzare insiemi di stima e verifica piccoli.

2. Se l'insieme di verifica non è temporalmente vicino a quello di stima l'errore di classificazione, e in particolare β, aumenta enormemente.

3. Se l'insieme di stima non copre tutti gli stati assunti dalle osservazioni nell'insieme di verifica, la classificazione è impossibile.

Il terzo svantaggio rende il grafo praticamente inutilizzabile. Data l'esiguità delle frodi, infatti, accade quasi sempre di avere insiemi di stima in cui esse non sono presenti e che, quindi, risultano incompleti.

Per questo motivo cercheremo nei prossimi paragrafi modelli alternativi dove la completezza dell'insieme di stima non sia più una condizione necessaria all'ottenimento di un risultato qualitativo.

5.3 Hidden Markov Model

La caratteristica suddivisione nel dataset tra le frodi e le non frodi, con le frodi concentrate in alcuni punti, suggerisce che il processo generatore dei dati possa essere una catena di Markov $\{X_t\}$ stazionaria a due stati, i quali rappresentano se l'operazione è una frode oppure no.

Tale catena avrà una matrice di transizione con valori prossimi a 1 sulla diagonale, che causano la netta suddivisione tra le frodi e le non frodi, e una probabilità di transizione molto bassa dallo stato 0 (non frode), allo stato 1 (frode).

La variabile X_t, che da questo momento chiameremo *Frode*, non è direttamente ossservabile, mentre vengono osservate alcune variabili Y_t, che sono quelle che abbiamo discusso nei paragrafi precedenti. Siamo quindi di fronte ad un modello di Markov nascosto.

Le operazioni bancarie, ordinate in ordine temporale, costituiscono un processo stocastico discreto $\{Y_t\}$. La variabile Y_t osservata, ha come componenti le variabili selezionate nei paragrafi precedenti, ossia:

$$Y_t = \{Operazione_t, Valuta_t, Importo_t, Durata_t, Orario_t\} \ .$$

La cardinalità dello spazio a cui appartiene la variabile Y è il prodotto delle cardinalità degli spazi a cui appartengono le sue componenti, ossia:

$$5 \times 2 \times 3 \times 2 \times 2 = 120 \ .$$

Per semplificare la notazione viene data una numerazione agli stati del processo Y. Si assume che, per ogni $t > 0$, valga:

$$Y_t = Y_{\{Operazione_t, Valuta_t, Importo_t, Durata_t, Orario_t\}} = \\ 24\,Operazione_t + 12\,Valuta_t + 4\,Importo_t + 2\,Durata_t + Orario_t \ .$$

È immediato dimostrare che questa operazione mappa in modo bigettivo l'insieme

$$\{0,1,2,3,4\} \times \{0,1\} \times \{0,1,2\} \times \{0,1\} \times \{0,1\}$$

nell'insieme

$$\{0,\ldots,119\} \ .$$

Assumiamo che esista una catena di Markov a due stati

$$\{X_t\}_{t>0} \in \{0,1\}$$

non osservabile, che rappresenta la funzione indicatrice della frode.

Indichiamo con A la matrice di transizione della catena di Markov $\{X_t\}_{t>0}$. Essa ha dimensione 2×2, e assume la seguente forma:

$$A = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \ .$$

L'elemento a_{01} rappresenta la probabilità, sapendo di aver osservato al tempo $t-1$ una non frode, di osservare una frode al tempo t, ossia:

$$a_{01} = \mathbb{P}\left[X_t = \text{frode} \mid X_{t-1} = \text{non frode}\right] \ .$$

Allo stesso modo, l'elemento a_{10} rappresenta la probabilità, sapendo di aver osservato una frode al tempo $t-1$, di osservare una non frode al tempo t.

La matrice B ha dimensione 2×120, e ha la seguente forma:

$$B = \begin{bmatrix} b_{0,0} & \ldots & b_{0,119} \\ b_{1,0} & \ldots & b_{1,119} \end{bmatrix} \ .$$

La seconda riga di B è di notevole interesse, infatti l'elemento $b_{1,j}$ rappresenta la probabilità, sapendo di aver ossservato una frode al tempo t, che Y_t assuma il valore j, cioè:

$$b_{1,j} = \mathbb{P}\left[Y_t = j \mid X_t = 1\right] \ .$$

Quindi gli stati $j \in \{0,\ldots,119\}$ per cui $b_{1,j}$ assume un valore alto rappresentano il profilo più comune del frodatore.

Dato un campione di osservazioni $Y = (Y_1,\ldots,Y_t)$, la procedura di classificazione avviene in due stadi.

Procedura 5.2 *Stima del modello di Markov nascosto:*

- *Passo 1: stima del modello λ che meglio si adatta ai dati.*

  Partendo da un $\lambda_0 = (A_0, B_0, \pi_0)$ iniziale, utilizzando l'algoritmo di Baum-Welch si stima il modello $\hat{\lambda} = (A, B, \pi)$ che massimizza la verosimiglianza per il campione scelto.

- *Passo 2: classificazione delle frodi.*

  Utilizzando il modello stimato al passo 1, si trova la sequenza $(X_1, \ldots, X_T)$ che ha la massima probabilità di essere osservata, dato il campione Y.

5.3.1 Scelta del modello iniziale

Come abbiamo visto nei capitoli precedenti, in un Hidden Markov model la funzione di verosimiglianza presenta molti massimi locali e l'algoritmo di Baum-Welch converge ad un valore diverso a seconda del punto iniziale.

Si rischia quindi, scegliendo un modello iniziale λ_0 errato, di ottenere una stima $\hat{\lambda}$ che non è il massimo assoluto della funzione di verosimiglianza. Per rimediare a questo inconveniente, si è scelto di utilizzare il metodo Monte Carlo. Sono state effettuate cioè molte stime, utilizzando dei valori iniziali $\lambda_0 = (A_0, B_0, \pi_0)$ casuali, e si è scelto tra i vari risultati ottenuti il modello $\hat{\lambda}$ che fa assumere il valore massimo alla funzione verosimiglianza.

Essendo matrice stocastica, la matrice A ha solamente due parametri liberi, a_{00} e a_{11}.

La matrice B, invece, ha 238 parametri liberi e, quindi, effettuare una simulazione su di essa è proibitivo.

Per risolvere questo problema abbiamo scelto di assumere che ogni riga di B_0 abbia una distribuzione Binomiale$(120, p)$. Cioè che:

$$b_{0,i,j} = \binom{120}{j} p_i^j (1 - p_i)^{120-j} \ .$$

Utilizzando matrici B_0 costruite in questo modo si ha una riduzione del numero di parametri liberi a 2: p_0 e p_1.

Il vettore π_0 non influisce sulla stima, pertanto scegliamo:

$$\pi_0 = (0.5, 0.5) \ .$$

Considerando un campione di osservazioni $Y = (Y_1, \ldots, Y_T)$, l'algoritmo per la scelta del modello è quindi il seguente.

Procedura 5.3 *Selezione del modello migliore con il metodo di MonteCarlo. Sia N il numero di iterazioni che si sceglie di eseguire. Per ogni $n \in \{1, \ldots, N\}$ si svolgono le seguenti operazioni:*

- *Passo 1: si simulano quattro numeri casuali uniformemente distributi* $a_0, a_1, p_0, p_1 \in (0,1)$.
- *Passo 2: si costruiscono le matrici* A_n *e* B_n *come:*

$$A_n = \begin{bmatrix} a_0 & 1-a_0 \\ 1-a_1 & a_1 \end{bmatrix}$$

$$B_n = \begin{bmatrix} \binom{120}{0} p_0^0 (1-p_0)^{120} & \dots & \binom{120}{120} p_0^{120} (1-p_0)^0 \\ \binom{120}{0} p_1^0 (1-p_1)^{120} & \dots & \binom{120}{120} p_1^{120} (1-p_1)^0 \end{bmatrix} .$$

- *Passo 3: utilizzando l'algoritmo di Baum-Welch con punto iniziale* (A_n, B_n, π_0) *si ottiene la stima di massima verosimiglianza* $\hat{\lambda}_n = (\hat{A}_n, \hat{B}_n, \hat{\pi}_n)$.
- *Passo 4: si calcola la log-verosimiglianza del campione* Y *secondo il modello* $\hat{\lambda}_n$.

Si seleziona infine il modello con la log-verosimiglianza più alta.

5.3.2 Programma HMM

Per la stima del modello di Markov nascosto è stato realizzato il programma `HMM` in linguaggio C++ (in appendice).

Le funzionalità del programma sono le seguenti.

1. Estrazione di un campione di osservazioni della dimensione scelta dal dataset dei dati.
2. Partendo da un modello iniziale λ_0 inserito dall'utente utilizzando i quattro parametri a_0, a_1, p_0 e p_1 precedentemente definiti, stima dell'HMM ottimale utilizzando l'algoritmo di Baum-Welch.
3. Identificazione dei parametri a_0, a_1, p_0 e p_1 ottimali per il campione scelto, utilizzando il metodo di Monte-Carlo.
4. Classificazione delle osservazioni e calcolo della matrice di confusione.

Per la validazione del programma `HMM` è stato utilizzato il dateset *Unfair Casino*. Si tratta di un esempio classico molto utilizzato in letteratura.

Un casinò disonesto utilizza due dadi. Il 99% dei dati è bilanciato ma l'1% è truccato e ha una probabilità che esca il 6 del 50%. Si ha una sequenza di risultati di lanci, dei quali non è dato sapere quale tipo di dado è stato utilizzato. Si tratta quindi di un semplice Hidden Markov Model, il cui processo di Markov nascosto assume i valori 0 e 1, che rappresentano il tipo di dado che è stato utilizzato per il lancio.

0 dado bilanciato.
1 dado truccato.

L'Hidden Markov Model da cui questi lanci derivano è quindi noto e ha le seguenti matrici A e B:

$$A = \begin{bmatrix} 0.99 & 0.01 \\ 0.02 & 0.98 \end{bmatrix} .$$

$$B = \begin{bmatrix} \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} & \frac{1}{6} \\ \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{10} & \frac{1}{2} \end{bmatrix} .$$

All'interno del pacchetto di `R` `HMM` è presente la funzione `dishonestCasino()`, che fornisce una dateset di 2000 osservazioni simulate, derivante dall'HMM sopra citato.

Utilizzando il programma `HMM`, cerchiamo di stimare, con l'algoritmo di Baum Welch, l'HMM che meglio si adatta ai dati.

Utilizzando il metodo Monte Carlo si ottengono i seguenti migliori valori iniziali:

$$\hat{a}_0 = 0.88, \quad \hat{a}_1 = 0.6, \quad \hat{p}_0 = 0.58, \quad \hat{p}_1 = 0.86 .$$

Utilizzando l'algoritmo di Baum Welch con questi valori iniziali si ottiene, dopo 1000 iterazioni, il seguente modello $\hat{\lambda}$:

$$\hat{A} = \begin{bmatrix} 0.994292 & 0.00570803 \\ 0.0107435 & 0.989257 \end{bmatrix}$$

$$\hat{B} = \begin{bmatrix} 0.164487 & 0.16709 & 0.179436 & 0.166867 & 0.166992 & 0.155128 \\ 0.124115 & 0.101889 & 0.111793 & 0.105195 & 0.0977431 & 0.459264 \end{bmatrix} .$$

In questo caso, probabilmente per via della particolare natura dei dati, la convergenza dell'algoritmo è molto rapida e si ottiene in meno di 100 iterazioni.

Utilizzando la procedura `baumWelch` di `R`, contenuta nel pacchetto `HMM`, sugli stessi dati si ottiene invece il seguente risultato:

$$\hat{A} = \begin{bmatrix} 0.99427238 & 0.005727615 \\ 0.01082905 & 0.989170955 \end{bmatrix}$$

$$\hat{B} = \begin{bmatrix} 0.1644844 & 0.1670307 & 0.1794465 & 0.1669240 & 0.16702426 & 0.1550901 \\ 0.1241670 & 0.1020743 & 0.1118497 & 0.1051581 & 0.09776183 & 0.4589890 \end{bmatrix} .$$

I due risultati sono leggermente diversi. Ciò è esclusivamente dovuto al numero di iterazioni effettuate. Utilizzando il programma `HMM` si perviene allo stesso identico risultato dopo 57 iterazioni.

La procedura `baumWelch` di `R` utilizza procedure automatiche di arresto per determinare se l'algoritmo ha effettivamente raggiunto la convergenza e terminare le iterazioni. Nel caso di *Unfair Casino* il risultato ottenuto è corretto

ma, come vedremo, in casi più complessi non è affatto semplice stabilire che l'algoritmo abbia effettivamente raggiunto la convergenza.

Utilizzando i modelli $\hat{\lambda}$ stimati si è calcolata la probabilità a posteriori $\gamma_t(i)$ per $i \in \{0, 1\}$ utilizzando il programma `HMM` e la procedura `posterior` del pacchetto `HMM` di verb+R+. Nella seguente tabella sono riportati i risultati per i primi 5 dati.

	Osservazioni	1	2	3	4	5
R	$\gamma_t(0)$	1.0000	0.9996	0.9991	0.9983	0.9970
	$\gamma_t(1)$	0.0000	0.0004	0.0009	0.0017	0.0030
Programma HMM	$\gamma_t(0)$	1	0.9996	0.9991	0.9983	0.9970
	$\gamma_t(1)$	0	0.0004	0.0009	0.0017	0.0030

I risultati risultano identici fino almeno alla quarta cifra decimale.

5.3.3 Stima

Dato che la variabile *Frode* è nascosta e non viene utilizzata per la stima, per un modello di Markov nascosto non si usano insiemi di verifica. La stima di un HMM è computazionalmente molto più onerosa di quella di un grafo. Dato che, a differenza di quest'ultimo, la complessità computazione dipende dal numero di osservazioni utilizzate per la stima, è impossibile da eseguire in un tempo ragionevole con sequenze di osservazioni superiori a qualche migliaio. Con un campione di 2000 osservazioni, il tempo di esecuzione è mediamente 0.5 secondi per ciscuna iterazione.

Ma, come abbiamo visto, il processo generatore dei dati sembra essere non stazionario e, quindi, su sequenze lunghe i risultati potrebbero non essere buoni.

Utilizzando la procedura descritta nel paragrafo precedente si riesce a eludere il problema della scelta del modello λ_0 iniziale ottimale. Tuttavia la stima di un HMM è soggetta ad un altro problema.

La verosimiglianza presenta molte zone piatte e, quindi, spesso resta costante per n iterazioni consecutive dell'algoritmo di Baum-Welch(con n anche maggiore di 100), per poi subire una brusca impennata all'iterazione $n + 1$-esima. Per questo motivo non è possibile utilizzare procedure automatiche per stabilire che sia avvenuta la convergenza e fermare l'algoritmo. Programmi che utilizzano queste procedure (ad esempio `R`), forniscono risultati errati perché spesso interrompono l'esecuzione prima che la convergenza sia stata effettivamente raggiunta.

Scegliamo lo stesso campione di 2000 osservazioni, con una concentrazione di 132 frodi, utilizzato per la stima del grafo.

Eseguendo una simulazione con 100 diversi valori iniziali casuali, si trovano i seguenti valori migliori per i quattro parametri a_0, a_1, p_0 e p_1:

$$\hat{a}_0 = 0.56, \quad \hat{a}_1 = 0.22, \quad \hat{p}_0 = 0.57, \quad \hat{p}_1 = 0.89 \ .$$

Stimando il modello con questi valori iniziali, si ottiene la seguente matrice $\hat{A}$:

$$\hat{A} = \begin{bmatrix} 0.999464 & 0.00053553 \\ 0.00763644 & 0.992364 \end{bmatrix} \ .$$

Come ci si aspettava, il valori sulla diagonale principale della matrice sono prossimi a 1, causando le concentrazioni delle frodi. Inoltre, il valore $\hat{a}_{2,1}$, che esprime la probabilità di transizione dallo stato *frode* allo stato *non frode*, è molto più grande del valore $\hat{a}_{1,2}$. Questo perché il numero delle frodi è molto più piccolo rispetto al numero delle non frodi.

La classificazione ottenuta utilizzando il modello $\hat{\lambda}$ è la seguente.

Matrice di confusione			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	131	0	
Frode=0	1	1868	
Totale	132	1868	2000
α	0.00757576		
β	0		
errore totale	0.0005		

Il modello di Markov nascosto si rivela quindi un ottimo classificatore. Osservando la seconda riga della matrice B stimata, si possono trovare gli stati che, nelle frodi, ricorrono con maggiore probabilità. Essi sono riportati nella seguente tabella.

Stati con alta ricorrenza all'interno delle frodi					
Stato $i_A \in \boldsymbol{I}_A$					**Probabilità**
Operazione	*Valuta*	*Importo*	*Durata*	*Orario*	$\mathbb{P}[i_A \mid \mathit{Frode} = 1]$
1	0	2	0	0	0.793891
4	0	2	0	0	0.0763362
0	0	2	0	0	0.0687025
1	0	2	0	1	0.0381696
1	0	2	1	0	0.0152672

Questi stati rappresentano, nell'intervallo temporale in cui si è effettuata la stima, il profilo più comune del frodatore. Alcuni stati sono comuni alla tabella degli stati con alta probabilità di frode, che si era trovata stimando sugli stessi dati un grafo indiretto, riportata nel paragrafo 5.2.3. Si tratta degli stati che, nella precedente tabella, avevano un numero di frodi attese molto elevato.

Risultati analoghi su campioni simili portano alla conclusione che il processo generatore dei dati, nelle sequenze di osservazioni con alte concentrazioni di frodi, sia effettivamente un Hidden Markov Model. Ma cosa accade se nell'insieme di osservazioni da classificare non sono presenti frodi?

Per rispondere a questa domanda scegliamo un insieme di stima di 2000 osservazioni all'interno del quale non sono state rilevate frodi. Dopo aver ottenuto con il metodo di Monte-Carlo i valori iniziali ottimali, che in questo caso sono:

$$a_0 = 0.28, \quad a_1 = 0.89, \quad p_0 = 0.6, \quad p_1 = 0.61 \ .$$

stimiamo il modello ottenendo la seguente matrice $\hat{A}$:

$$\hat{A} = \begin{bmatrix} 0.890618 & 0.109382 \\ 0.206648 & 0.793352 \end{bmatrix} \ .$$

I valori sulla diagonale principale di $\hat{A}$ sono molto piccoli rispetto a quelli che ci si aspetterebbe. Il modello stimato è molto lontano da quello che si pensa essere il vero modello, per questo motivo non stupisce che la classificazione risulti errata.

Matrice di confusione			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	0	527	
Frode=0	0	1473	
Totale	0	2000	2000
α	0		
β	0.2635		
errore totale	0.2635		

L'Hidden Markov Model risulta quindi essere un classificatore inadatto se nella sequenza di osservazioni non sono presenti frodi. In particolare, l'errore di secondo tipo β risulta molto elevato.

Utilizzando un HMM per la classificazione si ha la garanzia di bloccare tutti i tentativi di frode ($\alpha = 0$). Tuttavia, dato che il numero di frodi è molto piccolo rispetto al totale delle operazioni eseguite e, nel caso in cui esse non siano presenti, si ottiene un β elevato, verrà erroneamente bloccato circa il 20 per cento delle operazioni in realtà oneste.

5.3.4 Hidden Markov Model: vantaggi e svantaggi

Riassumiamo in questo paragrafo vantaggi e svantagggi sull'utilizzo dell'Hidden Markov Model .

Vantaggi

1. La variabile *Frode* è nascosta e non viene utilizzata per la stima, pertanto non occorre avere a disposizione un insieme di stima e di verifica.

2. Per lo stesso motivo, non ha importanza che il campione utilizzato copra tutti gli stati possibili. Semplicemente, la matrice B avrà dei valori uguali a 0 in corrispondenza degli stati non coperti.

3. Su campioni di osservazioni dove sono presenti concentrazioni di frodi l'HMM predice con errore 0, dando l'impressione di essere il vero processo generatore dei dati.

4. L'Hidden Markov Model garantisce l'identificazione di tutti i tentativi di frode.

Svantaggi

1. La stima dell'HMM è computazionalmente molto onerosa e per questa ragione non è possibile eseguirla su campioni di dimensione superiore a qualche migliaio di osservazioni.

2. è difficile stabilire che l'algoritmo di Baum-Welch abbia effettivamente raggiunto la convergenza e quindi non si possono utilizzare procedure automatiche per fermare l'algoritmo.

3. Il modello $\hat{\lambda}$ non fornisce una caratterizzazione degli stati ad alto rischio frode, che invece era fornita dal grafo.

4. Su insiemi di dati in cui non sono presenti frodi la classificazione ottenuta dall'HMM diventa cattiva, infatti si ottiene un β molto elevato.

5. L'HMM quindi bloccherà molte operazioni oneste identificandole erroneamente come frodi.

Gli ultimi due svantaggi rendono l'HMM, così come il grafo, non utilizzabile. Non è ragionevole infatti bloccare il 20% delle operazioni oneste anche se ciò garantirebbe di sventare quasi ogni tentativo di frode, specialmente considerando l'esiguità di queste ultime.

Nel prossimo paragrafo cercheremo di combinare i vantaggi dei due classificatori Grafo e Hidden Markov Model in modo da ottenere un classificatore che sia effettivamente utilizzabile nella pratica.

5.4 Combinazione di classificatori

Il grafo si dimostra un classificatore più stabile rispetto all'Hidden Markov Model, ma è soggetto a gravi problemi. Fra tutti, spicca particolarmente quello degli insiemi di stima incompleti, che non consente di classificare correttamente le frodi nel caso in cui le osservazioni dell'insieme di verifica siano diverse da quelle dell'insieme di stima.

Per superare questo problema abbiamo pensato di combinare opportunamente i pro e i contro dei due classificatori, ottenendo un classificatore misto.

Procedura 5.4 *Classificazione mista.*

Dato un campione di T osservazioni $y = (y_1, \ldots, y_T)$.

1. *Stima del grafo utilizzando come insieme di stima le T osservazioni precedenti a quelle da classificare, cioè $y_{stima} = (y_{-T+1}, \ldots, y_0)$.*
2. *Se l'insieme di stima è completo, classificazione di y utilizzando il risultato ottenuto dal grafo.*
3. *Se l'insieme di stima risulta incompleto, stima dell'HMM su y e utilizzo di quest'ultimo per classificare le osservazioni rimamenti.*

È stato quindi scelto, come esempio finale, un campione di 2000 osservazioni, all'interno del quale sono presenti 131 frodi, come sempre concentrate, in questo caso tra la 726-esima alla 856-esima osservazioni. Per insieme di stima si utilizzano le 2000 osservazioni precedenti. All'interno di esse non sono presenti frodi.

Si stima quindi il modello grafico M_2, che, come abbiamo visto, produce la stessa classificazione degli altri modelli grafici possibili. Sull'insieme di stima si ottiene una classificazione esatta, mentre, sull'insieme di verifica, si ottiene la seguente:

Matrice di confusione sull'insieme di verifica			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	0	0	
Frode=0	8	1869	
Totale	8	1869	1877
impossibile classificare 123 osservazioni			

Dato che la classificazione risulta incompleta, stimiamo sull'insieme di verifica di 2000 osservazioni un Hidden Markov Model.

Una volta ottenuti, con il metodo di MonteCarlo, i migliori valori iniziali:

$$a_0 = 0.32, \quad a_1 = 0.45, \quad p_0 = 0.14, \quad p_1 = 0.12$$

otteniamo la seguenti classificazione sulle 123 osservazioni mancanti.

Classificazione secondo l'HMM	
Frode=1	*Frode*=0
123	0

Combinando i classificatori, otteniamo il seguente risultato.

Matrice di confusione sull'insieme di verifica			
Previsti	**Osservati**		
	Frode=1	*Frode*=0	
Frode=1	123	0	
Frode=0	8	1869	
Totale	131	1869	2000
α	0.061		
β	0		
errore totale	0.004		

La classificazione ottenuta combinando i classificatori risulta quindi molto promettente e può essere oggetto di approfondimenti futuri.

5.5 Riferimenti bibliografici

La procedura 5.1, o *forward stepwise*, è liberamente tratta dal testo di Hojsgaard, Edwards e Lauritzen *Graphical Models with R*[6].

L'esempio *Unfair Casino* è tratto dal testo di Richard Durbin *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*[4] ed è citato nel pacchetto di `R HMM`.

Le altre procedure utilizzate nel capitolo sono originali.

Appendice A

Lemmi

Lemma A.1 *Inversione di Mobius*

Sia V un insieme finito e siano H e Φ due funzioni definite sull'insieme delle parti di V allora sono fatti equivalenti:

$$1) \quad \forall A \subseteq V \quad H(A) = \sum_{B \subseteq A} \Phi(B)$$

$$2) \quad \forall A \subseteq V \quad \Phi(A) = \sum_{B \subseteq A} (-1)^{|A \backslash B|} H(B)$$

Dimostrazione: Supponiamo che valga 1). Sia $A \subseteq V$. Allora:

$$\sum_{B \subseteq A} (-1)^{|A \backslash B|} H(B) = \sum_{B \subseteq A} (-1)^{|A \backslash B|} \left(\sum_{C \subseteq B} \Phi(C) \right) = \sum_{C \subseteq A} \Phi(C) \left(\sum_{C \subseteq B \subseteq A} (-1)^{|A \backslash B|} \right) =$$

$$\sum_{C \subseteq A} \Phi(C) \left(\sum_{D \subseteq A \backslash C} (-1)^{D} \right) = \Phi(A)$$

Infatti la somma $\sum_{D \subseteq A \backslash C} (-1)^{D}$ è uguale a zero tranne nel caso in cui $A \backslash C = \emptyset$ e quindi $C = A$, poiché ciascun insieme finito non vuoto ha un numero uguale di sottoinsiemi con cardinalità pari e sottoinsiemi con cardinalità dispari.

Supponiamo che valga 2). Sia $\subseteq V$. Allora:

$$\sum_{B \subseteq A} \Phi(B) = \sum_{B \subseteq A} \sum_{C \subseteq B} (-1)^{|B \backslash C|} H(C) = \sum_{C \subseteq A} H(C) \left(\sum_{C \subseteq B \subseteq A} (-1)^{|B \backslash C|} \right) =$$

$$\sum_{C \subseteq A} H(C) \left(\sum_{D \subseteq A \backslash C} (-1)^{|D|} \right) = H(A)$$

Infatti la somma $\sum_{D \subseteq A \setminus C} (-1)^{|D|}$ è uguale a zero tranne nel caso in cui $A \setminus C = \emptyset$ e quindi $C = A$, poiché ciascun insieme finito non vuoto ha un numero uguale di sottoinsiemi con cardinalità pari e sottoinsiemi con cardinalità dispari. □

Lemma A.2 *Siano x e y due numeri reali, $x, y \geq 0$ allora si ha:*

$$x^y e^{-x} \leq y^y e^{-y}$$

e vale l'uguaglianza solo se $x = y$

Dimostrazione: Se x e y sono entrambi 0, l'uguaglianza è verificata.

Se $x = 0$ e $y \neq 0$, si ha:

$$0 < y^y e^{-y}$$

e quindi la disuguaglianza è verificata.

Se $x \neq 0$ e $y = 0$ si ha:

$$e^{-x} < 1$$

e quindi la disuguaglianza è verificata.

Sia $x > 0$ e $y > 0$. Per ogni numero positivo z vale la seguente proprietà:

$$\log(z) \leq z - 1$$

Applicando il logaritmo a entrambi i membri della disuguaglianza si ottiene:

$$y \log(x) - x \leq y \log(y) - y$$

Dividendo entrambi i membri per y, infine, si ottiene:

$$\log\left(\frac{x}{y}\right) \leq \frac{x}{y} - 1$$

la quale è verificata per ogni numero positivo. □

Appendice B

Programma Grafo

Per la stima dei modelli grafici è stato sviluppato il programma `Grafo` in linguaggio C++, le cui funzionalità sono state discusse in precedenza. Di seguito è riportato il codice del programma. Per questioni di spazio, l'indentazione è stata parzialmente eliminata.

```
/**classe che definisce una osservazione del dataset FRAUD*/
class Campo
{
private:
int operazione;
bool frode;
bool festivo;
int utente;
const char* comune;
const char* valuta;
const char* importo;
const char* durata;
const char* orario;
int stato;
public:
/** numero totale di osservazioni nel dataset*/
static const int OSS=1000636;
/** distruttore della classe Campo*/
~Campo();
/** ritorna la variabile Frode per questa osservazione*/
bool getFrode();
/** ritorna la variabile Utente per questa osservazione*/
int getUtente();
/** ritorna la variabile Operazione per questa osservazione*/
int getOperazione();
/** ritorna la variabile Giorno Festivo per questa osservazione*/
int getFestivo();
/** ritorna la variabile Comune per questa osservazione*/
int getComune();
/** ritorna la variabile Valuta per questa osservazione*/
int getValuta();
/** ritorna la variabile Importo per questa osservazione*/
int getImporto();
/** ritorna la variabile Durata per questa osservazione*/
int getDurata();
/** ritorna la variabile Orario per questa osservazione*/
int getOrario();
```

```
/** ritorna lo stato Y_t usato per l'HMM per questa osservazione*/
int getStato();
/**costruisce una osservazione con le seguenti assegnazioni: Operazione=0, Frode=f,
GiornoFestivo=fe, Utente=u, Comune=co, Valuta=va, Importo=im, Durata=du,
Orario=ora, Y_t=stat*/
Campo(int o, bool f, bool fe, int u, char* co, char* va, char* im,char* du,char* ora,int stat);
/** stampa a schermo l'osservazione*/
void print();
};
/** implementazione della classe Campo*/
#include<iostream>#include"Campo.h"using namespace std;
bool Campo::getFrode()
{
return frode;
}
int Campo::getUtente()
{
return utente;
}
int Campo::getStato()
{
return stato;
}
int Campo::getOperazione()
{
return operazione;
}
int Campo::getFestivo()
{
return festivo;
}
int Campo::getComune()
{
char ll=comune[0];if(ll=='f'){return 0;}else{return 1;}
}
int Campo::getValuta()
{
char ll=valuta[0];if(ll=='e'){return 0;}else{return 1;}
}
int Campo::getImporto()
{
char ll=importo[0];if(ll=='b'){return 0;}else if(ll=='m'){return 1;}else{return 2;}
}
int Campo::getDurata()
{
char ll=durata[0];if(ll=='b'){return 0;}else{return 1;}
}
intCampo::getOrario()
{
char ll=orario[0];if(ll=='l'){return 0;}else{return 1;}
}
Campo::Campo(int o,bool f,bool fe,int u,char* co,char* va,char* im,char* du,char* ora,int stat)
{
operazione=o;frode=f;festivo=fe;utente=u;comune=co;valuta=va;importo=im;
durata=du;orario=ora;stato=stat;
}
voidCampo::print()
{
cout<<operazione<<""<<frode<<""<<festivo<<""<<utente<<""<<comune<<""<<valuta<<""<<importo
<<""<<durata<<""<<orario<<""<<stato<<"\n";
}
Campo::~Campo()
{
delete[]comune;delete[]valuta;delete[]importo;delete[]durata;delete[]orario;
```

```
}

/**classe che carica le osservazioni contenute nel dataset FRAUD*/
#include "Campo.h"
class Dati
{
private:
Campo* campi [Campo::OSS];
int numero;
public:
/** stampa a schermo le osservazioni Y_t*/
void print();
/**stampa a schermo l'osservazione n-esima*/
void printRiga(int n);
/**costruisce un dataset di osservazioni caricando i dati dal file s*/
Dati(const char* s);
/** distruttore della classe Dati*/
~Dati();
/**calcola il conteggio relativo all'osservazione n-esima e lo colloca nell'array vet*/
void conteggio(int n,int vet[6]);
/** ritorna l'utente dell'n-esima osservazione*/
int getUtente(int n);
/** ritorna la variabile Frode dell'n-esima osservazione*/
int getStato(int n);
/** ritorna lo stato Y_t dell'n-esima osservazione*/
bool getFrode(int n);
};
/** implementazione della classe Dati*/
#include<iostream>#include<fstream>#include<cstdlib>#include"Dati.h"using namespace std;
void Dati::print()
{
for(int i=0;i<Campo::OSS;i++){cout<<campi[i]->getStato()<<"\n";}
}
void Dati::printRiga(int n)
{
campi[n]->print();
}
void Dati::conteggio(int n,int vet[6])
{
vet[0]=campi[n]->getOperazione()-1;vet[1]=campi[n]->getValuta();
vet[2]=campi[n]->getImporto();vet[3]=campi[n]->getDurata();
vet[4]=campi[n]->getOrario();vet[5]=campi[n]->getFrode();
}
int Dati::getUtente(int n)
{
return campi[n]->getUtente();
}
int Dati::getStato(int n)
{
return campi[n]->getStato();
}
boolDati::getFrode(int n)
{
return campi[n]->getFrode();
}
Dati::~Dati()
{
for(int i=0;i<Campo::OSS;i++){delete campi[i];}
}
Dati::Dati(const char*s)
{
ifstream ifs;ifs.open(s,ifstream::in);char caratteri[100];int o;bool f;bool fe;int u;
char* co;char* va;char* im;char* du;char* ora;int stat;int i=0;int j=0;char pippo[4];
char pippo2[15];int contat=0;for(;contat<Campo::OSS;){i=0;j=0;ifs.getline(caratteri,100);
```

```
if(caratteri[0]=='1'||caratteri[0]=='2'||caratteri[0]=='3'||caratteri[0]=='4'
||caratteri[0]=='5'||caratteri[0]=='6'||caratteri[0]=='7'||caratteri[0]=='8'
||caratteri[0]=='9'||caratteri[0]=='0'){while(caratteri[i]!='#')
{i++;}for(int n=0;n<i;n++){pippo[n]=caratteri[n];}pippo[i]='\0';o=atoi(pippo);
i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++){pippo2[n]=caratteri[n+j];}
pippo2[i-j]='\0';f=(pippo2[0]=='f');i++;j=i;while(caratteri[i]!='#'){i++;}
for(int n=0;n<i-j;n++){pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';fe=(pippo2[0]=='F');
i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++){pippo[n]=caratteri[n+j];}
pippo[i-j]='\0';u=atoi(pippo);i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++)
{pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';co=newchar[i-j];for(int kl=0;kl<i-j;kl++)
{co[kl]=pippo2[kl];}i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++)
{pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';va=new char[i-j];for(int kl=0;kl<i-j;kl++)
{va[kl]=pippo2[kl];}i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++)
{pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';im=new char[i-j];for(int kl=0;kl<i-j;kl++)
{im[kl]=pippo2[kl];}i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++)
{pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';du=new char[i-j];for(int kl=0;kl<i-j;kl++)
{du[kl]=pippo2[kl];}i++;j=i;while(caratteri[i]!='#'){i++;}for(int n=0;n<i-j;n++)
{pippo2[n]=caratteri[n+j];}pippo2[i-j]='\0';ora=new char[i-j];for(int kl=0;kl<i-j;kl++)
{ora[kl]=pippo2[kl];}i++;j=i;while(caratteri[i]!='\0'){i++;}for(int n=0;n<i-j;n++)
{pippo[n]=caratteri[n+j];}pippo[i-j]='\0';stat=atoi(pippo);
campi[contat]=newCampo(o,f,fe,u,co,va,im,du,ora,stat);contat++;}}ifs.close();
}

/** classe  che definisce l'insieme dei vertici di un grafo*/
class Insieme
{
public:
/** ordina in ordine crescente l'array m, considerato come array di lunghezza l */
static void ordina(int* m,int l);
/** calcola il fattoriale di un numero naturale */
static long long fattoriale(int n);
/** calcola il coefficiente binomiale (n,k)*/
static long long coefficienteBinomiale(int n,int k);
/** calcola l'esponenziale b^e*/
static long long esponenziale(int b,int e);
/**ritorna l'insieme unione degli insiemi a e b */
static Insieme* unione(Insieme* a, Insieme* b);
/** ritorna l'insieme intersezione degli insiemi a e b */
static Insieme* intersezione(Insieme* a, Insieme* b);
/** ritorna true se l'insieme e' contenuto in a */
bool contenuto(Insieme* a);
/** costruisce l'insieme dei numeri naturali da 1 a n*/
Insieme(int n);
/** costruisce un insieme di n vertici prendendo i valori nel vettore v  */
Insieme(int* v, int n);
/** distruttore della classe Insieme */
~Insieme();
/**trova tutti i sottoinsiemi dell'insieme di partenza */
Insieme** sottoinsiemi();
/** ritorna la cardinalita' dell'insieme */
int getDim();
/** ritorna l'elemento i-esimo dell'insieme */
int getVertice(int i);
/**ritorna true se l'insieme e' uguale all'insieme a,
cioe' se ha la stessa cardinalita' e gli stessi elementi*/
bool equals(Insieme* a);
/** stampa a schermo l'insieme */
void print();
private:
bool minore(Insieme* b);
int* vertici;
int dimensione;
};
/*implementazione della classe Insieme */
```

```
#include<iostream> #include "Insieme.h"  using namespace std;
long long Insieme::coefficienteBinomiale(int n,int k)
{
return((fattoriale(n))/(fattoriale(k)*fattoriale(n-k)));
}
long long Insieme::esponenziale(int b,int e)
{
int retval=1; for(int j=1;j<=e;j++){retval=retval*b;} return retval;
}
long long Insieme::fattoriale(int n)
{
if(n<=1){return 1;} else{ int retval=1;
for(int j=2;j<=n;j++){retval=retval*j;} return retval;}
}
void Insieme::ordina(int* m,int l)
{
int* retval=new int [l]; int aux=0; int cursore=0;
for(int j=1; j<=l; j++){aux=0; for(int i=0;i<l;i++){if(m[i]>aux){aux=m[i]; cursore=i; }}
retval[l-j]=aux; m[cursore]=0;}
for(int i=0;i<l;i++){ m[i]=retval[i];}delete[] retval;
}
void Insieme::print()
{
cout<<"{"; for(int i=0;i<dimensione;i++){cout<<vertici[i]<<" ";}cout<<"}\n";
}
int Insieme::getDim()
{
return dimensione;
}
int Insieme::getVertice(int i)
{
return vertici[i];
}
Insieme::Insieme(int* v,int n)
{
dimensione=n; vertici=new int[n];
for(int i=0;i<n;i++){vertici[i]=v[i];} ordina(vertici,dimensione);
}
Insieme::Insieme(int n)
{
dimensione=n; vertici= new int[n]; for(int j=0;j<dimensione;j++){vertici[j]=j+1;}
}
Insieme::~Insieme()
{
delete[] vertici;
}
bool Insieme::minore(Insieme* b)
{
for(int i=0; i<dimensione;i++){for(int j=0; j<b->dimensione;j++)
{if(vertici[i]>=b->vertici[j]){return false;}}}return true;
}
Insieme** Insieme::sottoinsiemi()
{
int ddd=esponenziale(2,dimensione);Insieme** retval=new Insieme*[ddd];
retval[0]=new Insieme(0);
for(int i=1; i<=dimensione;i++){int* pippo= new int[1];
pippo[0]=vertici[i-1];retval[i]=new Insieme(pippo,1);
delete[] pippo;}int cv=dimensione+1; for(int i=2;i<=dimensione;i++){int kk=cv;
int cvn=coefficienteBinomiale(dimensione,i-1);
for(int j=cv-cvn;j<cv;j++){for(int k=1;k<=dimensione;k++){if(retval[j]->minore(retval[k]))
{retval[kk]=unione(retval[j],retval[k]); kk++;}}}
cv=cv+coefficienteBinomiale(dimensione,i);}return retval;
}
bool Insieme::equals(Insieme* a)
```

```
{
if(dimensione!=a->dimensione){return false;}for(int ii=0; ii<dimensione;ii++)
{if(vertici[ii]!=a->vertici[ii]){return false;} } return true;
}
bool Insieme::contenuto(Insieme* a)
{
 Insieme* i=intersezione(this,a); bool pippo=equals(i); delete i;return pippo;
}
Insieme* Insieme::unione(Insieme* a,Insieme* b)
{
int* retval=new int[a->dimensione+b->dimensione];
for(int i=0;i<a->dimensione;i++){retval[i]=a->vertici[i];}
for(int i=a->dimensione; i<a->dimensione+b->dimensione;i++)
{retval[i]=b->vertici[i-a->dimensione];}
ordina(retval,(a->dimensione+b->dimensione)); int d=1;int valc=retval[0];
for(int i=1;i<(a->dimensione+b->dimensione);i++){if(retval[i]>valc){valc=retval[i];d++;}}
int* rtv=new int[d]; rtv[0]=retval[0]; int j=0;
for(int i=1;i<(a->dimensione+b->dimensione);i++)
{if(retval[i]>rtv[j]){ j++;rtv[j]=retval[i];}}
delete[] retval; Insieme* inse=new Insieme(rtv,d);delete[] rtv;return inse;
}
Insieme* Insieme::intersezione(Insieme* a,Insieme* b)
{
int m=a->dimensione;if(b->dimensione>a->dimensione){m=b->dimensione;}
int* retval=new int[m];int cr=0;for(int i=0;i<a->dimensione;i++){int j=0;
while(j<b->dimensione && b->vertici[j]<=a->vertici[i]){if(b->vertici[j]==a->vertici[i])
{retval[cr]=a->vertici[i];cr++;}j++;}}int * rv=new int[cr];for(int i=0;i<cr;i++)
{rv[i]=retval[i];}delete[] retval;Insieme* inse= new Insieme(rv,cr);delete[] rv;
return inse;
}

#include "Insieme.h"
/**
classe che definisce un insieme di insiemi */
class Insiemi
{
public:
/** ritorna l'insieme i-esimo */
Insieme* getInsieme(int k);
/** ritorna il numero di insiemi di cui l'insieme di insiemi e' costituito */
int getNumero();
/**costruisce un insieme di insiemi partendo da un array di insiemi
e dalla dimensione dell'array*/
Insiemi(Insieme** inse, int n);
/** distruttore della classe Insiemi*/
~Insiemi();
private:
Insieme** insiemi;
int numero;
};
/* implementazione della classe Insiemi*/
#include"Insiemi.h";
Insiemi::Insiemi(Insieme**inse,intn)
{
numero=n;insiemi=inse;
}
Insiemi::~Insiemi()
{
for(int i=0;i<numero;i++){delete insiemi[i];}delete[]insiemi;
}
intInsiemi::getNumero()
{
return numero;
}
```

```
Insieme*Insiemi::getInsieme(intk)
{
return insiemi[k];
}

#include "Insiemi.h"
/**classe che definisce un grafo come insieme di vertici e di archi*/
class Grafoide
{
public:
/**numero dei vertici del grafo usato per l'analisi dei dati FRAUD*/
static const int NUM_VERTICI=6;
/**il grafo di interazione del modello grafico M1*/
static Grafoide* grafo1();
/**il grafo di interazione del modello grafico M2*/
static Grafoide* grafo2();
/** costruisce il grafo di 6 vertici senza archi */
Grafoide();
/**distruttore della classe Grafoide*/
~Grafoide();
/**stampa il grafo su schermo*/
void print();
/**aggiunge l'arco i-j al grafo*/
void addArco(int i, int j);
/**rimuove l'arco i-j dal grafo*/
void removeArco(int i,int j);
/**ritorna true se nel grafo  presente l'arco i-j */
bool haArco(int i,int j);
/**ritorna true se l'insieme ii  completo nel grafo*/
bool completo(Insieme* ii);
/** ritorna tutti gli insiemi completi del grafo */
Insiemi* getInsiemiCompleti();
/** ritorna le cliques del grafo*/
Insiemi* getCliques();
private:
int archi [NUM_VERTICI][NUM_VERTICI];
Insieme* vertici;
};
/*implementazione della classe Grafoide*/
#include "Grafoide.h"#include <iostream>using namespace std;
Grafoide* Grafoide::grafo2()
{
Grafoide* graph= new Grafoide();for(int i=0;i<(NUM_VERTICI-1);i++)
{graph->addArco(i,(NUM_VERTICI-1));}
return graph;
}
Grafoide* Grafoide::grafo1()
{
Grafoide* graph= new Grafoide();for(int i=1;i<(NUM_VERTICI-1);i++)
{graph->addArco(i,(NUM_VERTICI-1));
graph->addArco(i,0);}graph->addArco(0,(NUM_VERTICI-1));return graph;
}
void Grafoide::print()
{
vertici->print();for (int i = 0; i < NUM_VERTICI; i++)
{for (int j = i + 1; j < NUM_VERTICI; j++)
{if (archi[i][j] == 1){cout<<(i+1)<<"---"<<(j+1)<<"\n";}}}
}
Grafoide::Grafoide()
{
vertici= new Insieme(NUM_VERTICI);for(int i=0;i<NUM_VERTICI;i++)
{for(int j=0;j<NUM_VERTICI;j++)
{if(i==j){archi[i][j]=1;}else{archi[i][j]=0;}}}
}
```

```
Grafoide::~Grafoide()
{
delete vertici;
}
void Grafoide::addArco(int i,int j)
{
archi[i][j]=1;archi[j][i]=1;
}
void Grafoide::removeArco(int i,int j)
{
archi[i][j]=0;archi[j][i]=0;
}
bool Grafoide:: haArco(int i,int j)
{
return archi[i][j]==1;
}
bool Grafoide::completo(Insieme* ii)
{
if(!ii->contenuto(vertici)){return false;}for(int i=0;i<ii->getDim();i++)
{for(int j=0;j<ii->getDim();j++)
{if(archi[ii->getVertice(i)-1][ii->getVertice(j)-1]==0){return false;}}}return true;
}
Insiemi* Grafoide::getInsiemiCompleti()
{
Insieme** retval=vertici->sottoinsiemi();int cont=0;
int lunghezza=Insieme::esponenziale(2,NUM_VERTICI);
for(int i=0; i<lunghezza;i++){if(completo(retval[i])){cont++;}}
Insieme** rv=new Insieme* [cont];int cv=0;
for(int i=0;i<lunghezza;i++){if(completo(retval[i])){rv[cv]=retval[i];cv++;}
else{delete retval[i];}}delete[] retval;return new Insiemi(rv,cont);
}
Insiemi* Grafoide::getCliques()
{
Insieme** retval=vertici->sottoinsiemi();int cont=0;
int lunghezza=Insieme::esponenziale(2,NUM_VERTICI);
for(int i=0; i<lunghezza;i++)
{if(completo(retval[i])){cont++;}}
Insieme** rv=new Insieme* [cont];int cv=0;
for(int i=0;i<lunghezza;i++){if(completo(retval[i]))
{rv[cv]=retval[i];cv++;}else{delete retval[i];}}delete[] retval;
int cont2=0;Insieme** retval2=new Insieme*[cont];for(int i=0;i<cont;i++)
{int contatore=i+1;
for(int j=i+1;j<cont;j++){if(!rv[i]->contenuto(rv[j])){contatore++;}}
if(contatore==cont){retval2[cont2]=rv[i];cont2++;}else{delete rv[i];}}delete[] rv;
Insieme** rv2=new Insieme*[cont2];for(int i=0;i<cont2;i++){rv2[i]=retval2[i];}
delete[] retval2;return new Insiemi(rv2,cont2);
}

#include"Dati.h" #include "Grafoide.h"
/** classe che definisce una tabella di contingenza per la variabili utilizzate */
class Tabella
{
 public:
/** diviso per 10 e' la soglia usata per la classificazione delle frodi*/
static const int SOGLIA=5;
/** numero di valori assunti dalla variabile Operazione */
static const int NUM_OPERAZIONE=5;
/** numero di valori assunti dalla variabile Valuta */
static const int NUM_VALUTA=2;
/** numero di valori assunti dalla variabile Importo */
static const int NUM_IMPORTO=3;
/** numero di valori assunti dalla variabile Durata*/
static const int NUM_DURATA=2;
/** numero di valori assunti dalla variabile Orario*/
```

```
static const int NUM_ORARIO=2;
/**numero di valori assunti dalla variabile Frode*/
static const int NUM_FRODE=2;
/** posizione della variabile frode nella tabella*/
static const int POS_FRODE=5;
/** posizione della variabile operazione nella tabella*/
static const int POS_OPERAZIONE=0;
/** numero di stati assunti dalla variabile var */
static int  getDim(int var);
/** costruisce una tabella con i conteggi uguali a 0*/
Tabella();
/**totale conteggi della tabella di contingenza */
double totaleStimati();
/**conteggi della cella val della tabella marginale secondo la variabile var*/
double marginale(int var,int val);
/**conteggi della cella val della tabella marginale secondo le variabili var*/
double marginaleGen(int* var, int* val, int tot);
 /**dimensione della tabella marginale per la variabile var */
int dimMarginale(int var);
 /**dimensione della tabella marginale definita dall'insieme ins*/
int getDimMarginale(Insieme* ins);
 /**conteggio di una cella della tabella di contingenza */
double getConteggio(int a,int d,int e,int f,int g,int h);
 /**assegna un valore a una cella della tabella di contingenza*/
void setConteggio(double val,int a,int d,int e,int f,int g,int h);
 /**incrementa di uno il conteggio di una cella della tabella di contingenza*/
void incrementa(int a,int d,int e,int f,int g,int h);
/** stampa su schermo la tabella di contingenza */
void printStima();
/** stampa la tabella di contingenza sul file
il cui nome e' passato ad argomento alla funzione */
void printStima(const char* result);
/** assegna il valore 1 a tutte le celle della tabella di contingenza*/
           void resettaStima();
private:
double tabella [NUM_OPERAZIONE][NUM_VALUTA][NUM_IMPORTO][NUM_DURATA][NUM_ORARIO][NUM_FRODE];
};
/* implementazione della classe Tabella */
#include "Tabella.h"#include<iostream> #include <fstream>
int Tabella::getDim(int var)
{
if(var==0){return NUM_OPERAZIONE;}else if(var==1){return NUM_VALUTA;}
else if(var==2){return NUM_IMPORTO;}
else if(var==3){return NUM_DURATA;}else if(var==4){return NUM_ORARIO;}
else if(var==5){return NUM_FRODE;}else{return 0;}
}
Tabella::Tabella()
{
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++)
{for(int f=0;f<NUM_DURATA;f++){for(int g=0;g<NUM_ORARIO;g++){for(int h=0;h<NUM_FRODE;h++)
{tabella[a][d][e][f][g][h]=0;}}}}}}
}
void Tabella::resettaStima()
{
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++){for(int g=0;g<NUM_ORARIO;g++)
{for(int h=0;h<NUM_FRODE;h++){tabella[a][d][e][f][g][h]=1;}}}}}}
}
double Tabella::marginale(int var,int val)
{
double aux=0;bool aa=false;bool dd=false;bool ee=false;bool ff=false;bool gg=false;
bool hh=false;for(int a=0;a<NUM_OPERAZIONE;a++){if(var==0){a=val;aa=true;}
for(int d=0;d<NUM_VALUTA;d++){if(var==1){d=val;dd=true;}for(int e=0;e<NUM_IMPORTO;e++)
```

```
{if(var==2){e=val;ee=true;}
for(int f=0;f<NUM_DURATA;f++){if(var==3){f=val;ff=true;}for(int g=0;g<NUM_ORARIO;g++)
{if(var==4){g=val;gg=true;}for(int h=0;h<NUM_FRODE;h++){if(var==5){h=val;hh=true;}
aux+=tabella[a][d][e][f][g][h];if(hh){h=100;}}if(gg){g=100;}}if(ff){f=100;}}
if(ee){e=100;}}if(dd){d=100;}}if(aa){a=100;}}return aux;
}
double Tabella::marginaleGen(int* var,int* val, int tot)
{
double aux=0;bool aa=false;bool dd=false;bool ee=false;bool ff=false;bool gg=false;
bool hh=false;for(int a=0;a<NUM_OPERAZIONE;a++){for(int z=0;z<tot;z++){if(var[z]==0)
{a=val[z];aa=true;}}for(int d=0;d<NUM_VALUTA;d++){for(int z=0;z<tot;z++){if(var[z]==1)
{d=val[z];dd=true;}}for(int e=0;e<NUM_IMPORTO;e++){for(int z=0;z<tot;z++){if(var[z]==2)
{e=val[z];ee=true;}}for(int f=0;f<NUM_DURATA;f++){for(int z=0;z<tot;z++){if(var[z]==3)
{f=val[z];ff=true;}}for(int g=0;g<NUM_ORARIO;g++){for(int z=0;z<tot;z++){if(var[z]==4)
{g=val[z];gg=true;}}for(int h=0;h<NUM_FRODE;h++){for(int z=0;z<tot;z++){if(var[z]==5)
{h=val[z];hh=true;}}aux+=tabella[a][d][e][f][g][h];if(hh){h=100;}}if(gg){g=100;}}
if(ff){f=100;}}if(ee){e=100;}}if(dd){d=100;}}if(aa){a=100;}}return aux;
}
int Tabella::getDimMarginale(Insieme* ins)
{
int nn=ins->getDim();if(nn==0){return 0;}int result=1;for(int i=0;i<nn;i++){
int kk=getDim(ins->getVertice(i)-1);int contat=0;for(int j=0;j<kk;j++)
{if(marginale((ins->getVertice(i)-1),j)!=0){contat++;}}result=result*contat;}
return result;
}
int Tabella::dimMarginale(int var)
{
int kk=getDim(var);int contat=0;for(int i=0;i<kk;i++){if(marginale(var,i)!=0)
{contat++;}}return contat;
}
double Tabella::totaleStimati()
{
double cont=0;for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++){
for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++)
{for(int g=0;g<NUM_ORARIO;g++){
for(int h=0;h<NUM_FRODE;h++){cont+=tabella[a][d][e][f][g][h];}}}}}}return cont;
}
void Tabella::printStima()
{
cout<<"Stima di massima verosimiglianza dei conteggi attesi:\n";double tot=0;
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++)
{for(int g=0;g<NUM_ORARIO;g++){for(int h=0;h<NUM_FRODE;h++)
{cout<<tabella[a][d][e][f][g][h]<<" ";
tot+=tabella[a][d][e][f][g][h];}}}}}}cout<<"\n";cout<<"Conteggi totali: "<<tot<<"\n";
}
void Tabella::printStima(const char* result)
{
ofstream of;of.open( result, ios::out | ios::app );double tot=0;
of<<"Stima di massima verosimiglianza dei conteggi attesi:\n";
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++)
{for(int f=0;f<NUM_DURATA;f++){for(int g=0;g<NUM_ORARIO;g++)
{for(int h=0;h<NUM_FRODE;h++)
{of<<tabella[a][d][e][f][g][h]<<" ";tot+=tabella[a][d][e][f][g][h];}}}}}}
of<<"\n";of<<"Conteggi totali: "<<tot<<"\n";of.close();
}
double Tabella::getConteggio(int a,int d,int e,int f,int g,int h)
{
return tabella[a][d][e][f][g][h];
}
void Tabella::setConteggio(double val,int a,int d,int e,int f,int g,int h)
{
tabella[a][d][e][f][g][h]=val;
```

```
}
void Tabella::incrementa(int a,int d,int e,int f,int g,int h)
{
tabella[a][d][e][f][g][h]=tabella[a][d][e][f][g][h]+1;
}

#include "Tabella.h"
/** classe che definisce un campione di osservazioni di tipo tabella di contingenza*/
class Campione: public Tabella
{
public:
/** ritorna il numero di osservazioni contenute nel campione*/
int getTotale();
/** ritorna la posizione dell'osservazione i all'interno del dataset dei dati*/
int getPos(int i);
/** setta la probabilita' di frode a val per l'osservazione i*/
void setY(int i,double val);
/** costruisce un campione da un dataset dat estraendo num osservazioni a
partire dalla posizione pos*/
Campione(Dati* dat, int num, int pos);
/** distruttore della classe Campione*/
~Campione();
/** crea la matrice di Confusione per il campione*/
void creaMatriceConfusione();
/** stampa i risultati sul file di nome result */
void printFile(const char* result);
/** stampa la tabella di contingenza del campione sullo schermo*/
void print();
/** stampa la matrice di confusione sullo schermo*/
void printMatriceConfusione();
private:
int totale;
int* Pos;
int* YY;
int* Yvero;
double* Y;
int tabellaconfusione[2][2];
double errore;
double alpha;
double beta;
};
/** implementazione della classe Campione*/
#include"Campione.h"#include<fstream>#include<iostream>
void Campione::print()
{
for(int i=0;i<totale;i++){cout<<Yvero[i]<<" "<<Pos[i]<<"\n";}cout<<"totale="<<totale<<"\n";
}
void Campione::printMatriceConfusione()
{
cout<<"Matrice confusione:\n";cout<<tabellaconfusione[0][0]<<
" "<<tabellaconfusione[0][1]<<"\n";
cout<<tabellaconfusione[1][0]<<" "<<tabellaconfusione[1][1]<<"\n";
cout<<"Errore totale: "<<errore<<"\n";cout<<"alpha: "<<alpha<<"beta: "<<beta<<"\n";
}
int Campione::getTotale()
{
return totale;
}
intCampione::getPos(int i)
{
return Pos[i];
}
void Campione::setY(int i,double val)
{
```

```
Y[i]=val;if(Y[i]>(double)SOGLIA/10.0){YY[i]=1;}else{YY[i]=0;}
}
Campione::~Campione()
{
delete[]Y;delete[]YY;delete[]Pos;delete[]Yvero;
}
Campione::Campione(Dati* dat,int num,int pos):Tabella()
{
totale=num;Pos=new int[totale];Y=new double[totale];YY=new int[totale];
Yvero=new int[totale];for(int i=0;i<totale;i++){int vet[6];dat->conteggio((pos+i),vet);
incrementa(vet[0],vet[1],vet[2],vet[3],vet[4],vet[5]);
Yvero[i]=vet[POS_FRODE];Pos[i]=(pos+i);}
}
void Campione::creaMatriceConfusione()
{
tabellaconfusione[0][0]=0;tabellaconfusione[1][0]=0;tabellaconfusione[0][1]=0;
tabellaconfusione[1][1]=0;for(int i=0;i<totale;i++){if(Yvero[i]==1){if(YY[i]==1)
{tabellaconfusione[0][0]++;}else{tabellaconfusione[1][0]++;}}else{if(YY[i]==1)
{tabellaconfusione[0][1]++;}else{tabellaconfusione[1][1]++;}}}
errore=((double)(tabellaconfusione[0][1]+tabellaconfusione[1][0]))/
((double)(tabellaconfusione[0][1]+tabellaconfusione[1][0]+tabellaconfusione[1][1]
+tabellaconfusione[0][0]));
alpha=((double)(tabellaconfusione[1][0]))/
((double)(tabellaconfusione[0][0]+tabellaconfusione[1][0]));
beta=((double)(tabellaconfusione[0][1]))/
((double)(tabellaconfusione[0][1]+tabellaconfusione[1][1]));
}
void Campione::printFile(const char* result)
{
ofstream of;of.open(result,ios::out|ios::app);of<<"Campione di numerosita: "<<totale<<"\n";
of<<"Stima di massima verosimiglianza della probabilita di frode:\n";
for(int i=0;i<totale;i++){of<<Y[i]<<" ";}of<<"\n";
of<<"Sequenza di massima verosimiglianza:\n";
for(int i=0;i<totale;i++){of<<YY[i]<<" ";}of<<"\n";of<<"Sequenza reale:\n";
for(int i=0;i<totale;i++){of<<Yvero[i]<<" ";}of<<"\n";
of<<"Matrice confusione:\n";
of<<tabellaconfusione[0][0]<<" "<<tabellaconfusione[0][1]<<"\n";
of<<tabellaconfusione[1][0]<<" "<<tabellaconfusione[1][1]<<"\n";
of<<"Errore totale:"<<errore<<"\n";
of<<"alpha: "<<alpha<<"beta: "<<beta<<"\n";of.close();
}

/** classe che definisce e stima un modello grafico*/
#include "Campione.h"
class Grafo: public Tabella
{
public:
/** ritorna il v-esimo valore tra a, b,... f*/
static int get(int v,int a,int b,int c,int d,int e,int f);
/**costruisce un oggetto Grafo dal dataset dat, estraendo un insieme di stima
di numerosita' numerosi partendo dalla posizione posiz e un insieme di verifica
di numerosita' dimverifica a partire dalla posizione posverifica*/
Grafo(Dati* dat,int numerosi,int posiz, int dimverifica,int posverifica);
/** distruttore della classe Grafo*/
~Grafo();
/** esegue la procedura forward stepwise sull'insieme di stima, eseguendo
numPassi iterazioni dell'algoritmo IPS e ritorna il grafo di interazione del miglior
modello ottenuto, cioe' di quello che minimizza l'AIC*/
Grafoide* stepwise(int numPassi);
/**esegue l'operazione di aggiustamento delle marginali aggiustando valore,
utilizzando la marginale definita dalle prime tot variabili contenute in var
che assumono i valori contenuti in val*/
double aggiustaGen(int* var,int* val, double valore,int tot);
/**ritorna la dimensione del modello grafico M1*/
```

```
int dimGrafo();
/** ritorna la dimensione del modello grafico M2*/
int dimGrafo2();
/**ritorna la dimensione del modello grafico con grafo di interazione graph*/
int dimensioneGrafo(Grafoide* graph);
/**calcola il coefficiente AIC sul campione camp
per il modello grafico con grafo di interazione graph*/
long double AIC(Campione* camp, Grafoide* graph);
/**stima il modello grafico con grafo di interazione graph con l'algoritmo IPS,
eseguendo numPassi iterazioni*/
void IPS(int numPassi, Grafoide* graph);
/**stima il modello grafico con grafo di interazione graph con l'algoritmo IPS
eseguendo numPassi iterazioni, dopo di che classifica le osservazioni
degli insiemi di stima e verifica e stampa i risultati*/
void algoritmoIPS(int numPassi, Grafoide* graph);
/** stima esatta del modello grafico M1, classificazione e stampa dei risultati*/
void stimaEsattaMassimaVerosimiglianza();
/** stima esatta del modello grafico M2, classificazione e stampa dei risultati*/
void stimaEsatta2MassimaVerosimiglianza();
/**calcola la verosimiglianza del Campione camp*/
long double logVerosimiglianza(Campione* camp);
/**classifica le osservazioni del Campione camp*/
void sequenzaMassimaVerosimiglianza(Campione* camp);
/**stampa sul file result le probabilita' di frode di tutti gli stati*/
void printProbFrode(const char* result);
/** stampa tutti i risultati sul file result*/
void printFile(const char* result);
private:
Dati* dati;
Campione* campstima;
Campione* campverifica;
};
/** implementazione della classe Grafo*/
#include<iostream>#include<fstream>#include<cmath>#include"Grafo.h"
int Grafo::get(int v,int a,int b,int c,int d,int e,int f)
{
if(v==0){return a;}else if(v==1){return b;}else if(v==2){return c;}
else if(v==3){return d;}else if(v==4){return e;}else{return f;}
}
Grafo::~Grafo()
{
delete campstima;delete campverifica;
}
Grafo::Grafo(Dati* dat,int numerosi,int posiz,int dimverifica,int posverifica):Tabella()
{
dati=dat;resettaStima();campstima=new Campione(dati,numerosi,posiz);
campverifica=new Campione(dati,dimverifica,posverifica);
}
double Grafo::aggiustaGen(int* var,int* val,double valore,int tot)
{
double v1=campstima->marginaleGen(var,val,tot);double v2=marginaleGen(var,val,tot);
if(v2==0){return 0;}else{return((valore*v1)/(v2));}
}
long double Grafo::AIC(Campione* camp,Grafoide* graph)
{
return(-2*logVerosimiglianza(camp)+2*dimensioneGrafo(graph));
}
long double Grafo::logVerosimiglianza(Campione* camp)
{
long double aux=0;for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++){for(int g=0;g<NUM_ORARIO;g++)
{for(int h=0;h<NUM_FRODE;h++){if(getConteggio(a,d,e,f,g,h)>0){
aux+=(camp->getConteggio(a,d,e,f,g,h)*log(getConteggio(a,d,e,f,g,h))
-getConteggio(a,d,e,f,g,h));}}}}}}return aux;
```

```
}
void Grafo::stimaEsattaMassimaVerosimiglianza()
{
int* varo= new int[2];int* valo=new int [2];
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++)
{for(int g=0;g<NUM_ORARIO;g++){for(int h=0;h<NUM_FRODE;h++){
double valor1=campstima->marginale(POS_FRODE,h);if(valor1!=0){double valor2;varo[0]=0;
varo[1]=POS_FRODE;valo[0]=a;valo[1]=h;valor2=campstima->marginaleGen(varo,valo,2);
for(int mm=1;mm<POS_FRODE;mm++){varo[0]=mm;varo[1]=POS_FRODE;valo[0]=get(mm,a,d,e,f,g,h);
valo[1]=h;valor2=(valor2*campstima->marginaleGen(varo,valo,2))/(valor1);}
setConteggio(valor2,a,d,e,f,g,h);}else{setConteggio(0,a,d,e,f,g,h);}}}}}}delete[] varo;
delete[] valo;cout<<"log-verosimiglianza:"<<logVerosimiglianza(campstima)<<"\n";
sequenzaMassimaVerosimiglianza(campstima);sequenzaMassimaVerosimiglianza(campverifica);
cout<<"Dimensione devianza:"<<dimGrafo()<<"\n";cout<<"Insieme di stima\n";
campstima->printMatriceConfusione();cout<<"Insieme di verifica\n";
campverifica->printMatriceConfusione();const char* res="risultatoEsatto.txt";
printFile(res);
}
Grafoide* Grafo::stepwise(int numPassi)
{
int massimo=((Grafoide::NUM_VERTICI)*(Grafoide::NUM_VERTICI-1))/2;
Grafoide* aux=new Grafoide();
cout<<"Stima del grafo:\n";aux->print();IPS(numPassi,aux);
double aic=AIC(campstima,aux);cout<<"AIC:"<<aic<<"\n";for(int h=1;h<=massimo;h++)
{int* arcone=new int[2];double aic2,aicBest;bool pippo=false;
for(int i=0;i<Grafoide::NUM_VERTICI;i++){for(int j=i+1;j<Grafoide::NUM_VERTICI;j++)
{if(!aux->haArco(i,j)){aux->addArco(i,j);cout<<"Stima del grafo:\n";aux->print();
IPS(numPassi,aux);aic2=AIC(campstima,aux);cout<<"AIC:"<<aic2<<"\n";aux->removeArco(i,j);
if(!pippo){pippo=true;aicBest=aic2;arcone[0]=i;arcone[1]=j;}else{if(aic2<aicBest){
aicBest=aic2;arcone[0]=i;arcone[1]=j;}}}}}if(aic<=aicBest){break;}else{aic=aicBest;
aux->addArco(arcone[0],arcone[1]);}delete[]arcone;}cout<<"Miglior modello:\n";
aux->print();cout<<"Cliques:\n";Insiemi* insiem=aux->getCliques();
for(int i=0;i<insiem->getNumero();i++)
{insiem->getInsieme(i)->print();}delete insiem;cout<<"AIC:"<<aic<<"\n";return aux;
}
void Grafo::stimaEsatta2MassimaVerosimiglianza()
{
int* varo=new int[3];int* valo= new int[3];for(int a=0;a<NUM_OPERAZIONE;a++)
{for(int d=0;d<NUM_VALUTA;d++){for(int e=0;e<NUM_IMPORTO;e++)
{for(int f=0;f<NUM_DURATA;f++){for(int g=0;g<NUM_ORARIO;g++)
{for(int h=0;h<NUM_FRODE;h++)
{double valor1=campstima->marginale2(POS_OPERAZIONE,POS_FRODE,a,h);if(valor1!=0)
{double valor2;valor2=campstima->marginale3(POS_OPERAZIONE,POS_FRODE,1,a,h,d);
for(int mm=2;mm<POS_FRODE;mm++){varo[2]=mm;valo[2]=get(mm,a,d,e,f,g,h);
valor2=(valor2*campstima->marginaleGen(varo,valo,3))/(valor1);}
setConteggio(valor2,a,d,e,f,g,h);}else{setConteggio(0,a,d,e,f,g,h);}}}}}}delete[]varo;
delete[]valo;cout<<"log-verosimiglianza:"<<logVerosimiglianza(campstima)<<"\n";
sequenzaMassimaVerosimiglianza(campstima);
sequenzaMassimaVerosimiglianza(campverifica);
cout<<"Dimensione devianza:"<<dimGrafo2()<<"\n";cout<<"Insieme di stima\n";
campstima->printMatriceConfusione();cout<<"Insieme di verifica\n";
campverifica->printMatriceConfusione();
const char* res="risultatoEsatto.txt";printFile(res);
}
void Grafo::IPS(int numPassi,Grafoide* graph)
{
Insiemi* cliques=graph->getCliques();resettaStima();
cout<<"Iterazione: 0 log-verosimiglianza: "<<logVerosimiglianza(campstima)<<"\n";
for(int i=1;i<=numPassi;i++){for(int j=0;j<cliques->getNumero();j++)
{int number=cliques->getInsieme(j)->getDim();int* var=new int[number];
int* val=new int[number];
double aux[NUM_OPERAZIONE][NUM_VALUTA][NUM_IMPORTO][NUM_DURATA][NUM_ORARIO][NUM_FRODE];
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++){
```

```
for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++)
{for(int g=0;g<NUM_ORARIO;g++){for(int h=0;h<NUM_FRODE;h++)
{for(int ss=0;ss<number;ss++){var[ss]=cliques->getInsieme(j)->getVertice(ss)-1;
val[ss]=get(var[ss],a,d,e,f,g,h);}
aux[a][d][e][f][g][h]=aggiustaGen(var,val,getConteggio(a,d,e,f,g,h),number);
}}}}}}for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++){
for(int g=0;g<NUM_ORARIO;g++){for(int h=0;h<NUM_FRODE;h++){
setConteggio(aux[a][d][e][f][g][h],a,d,e,f,g,h);}}}}}}
delete[]var;delete[]val;}
cout<<"Iterazione:"<<i<<"log-verosimiglianza:"<<logVerosimiglianza(campstima)<<"\n";}
}
void Grafo::algoritmoIPS(int numPassi,Grafoide* graph)
{
IPS(numPassi,graph);sequenzaMassimaVerosimiglianza(campstima);
sequenzaMassimaVerosimiglianza(campverifica);
cout<<"Dimensione devianza: "<<dimensioneGrafo(graph)<<"\n";cout<<"Insieme di stima\n";
campstima->printMatriceConfusione();cout<<"Insieme di verifica\n";
campverifica->printMatriceConfusione();const char* res="risultato.txt";printFile(res);
}
int Grafo::dimensioneGrafo(Grafoide* graph)
{
Insiemi* cliques=graph->getCliques();int result=0;Insieme*ins=new Insieme(0);
Insieme* ins2=ins;Insieme* inter=new Insieme(0);
for(int i=0;i<cliques->getNumero();i++){delete inter;
inter=Insieme::intersezione(ins,cliques->getInsieme(i));
result=result-getDimMarginale(inter);
result=result+getDimMarginale(cliques->getInsieme(i));
ins2=Insieme::unione(cliques->getInsieme(i),ins);delete ins;ins=ins2;}
delete ins;delete inter;delete cliques;return result;
}
int Grafo::dimGrafo()
{
int contat=0;int margf=dimMarginale(POS_FRODE);for(int i=0;i<POS_FRODE;i++)
{int contat2=dimMarginale(i)*margf;contat+=contat2;}
return(contat-((POS_FRODE-1)*margf));
}
int Grafo::dimGrafo2()
{
int contat=0;int margf=dimMarginale(POS_FRODE)*dimMarginale(POS_OPERAZIONE);
for(int i=1;i<POS_FRODE;i++){int contat2=dimMarginale(i)*margf;contat+=contat2;}
return(contat-((POS_FRODE-2)*margf));
}
void Grafo::sequenzaMassimaVerosimiglianza(Campione* camp)
{
for(int i=0;i<camp->getTotale();i++){int vet[6];
dati->conteggio(camp->getPos(i),vet);
camp->setY(i,(getConteggio(vet[0],vet[1],vet[2],vet[3],vet[4],1)/
(getConteggio(vet[0],vet[1],vet[2],vet[3],vet[4],1)+
getConteggio(vet[0],vet[1],vet[2],vet[3],vet[4],0))));}camp->creaMatriceConfusione();
}
void Grafo::printProbFrode(const char* result)
{
ofstream of;of.open(result,ios::out|ios::app);of<<"Probabilita di frode\n";
for(int a=0;a<NUM_OPERAZIONE;a++){for(int d=0;d<NUM_VALUTA;d++)
{for(int e=0;e<NUM_IMPORTO;e++){for(int f=0;f<NUM_DURATA;f++)
{for(int g=0;g<NUM_ORARIO;g++){
double prop=(getConteggio(a,d,e,f,g,1))/
(getConteggio(a,d,e,f,g,0)+getConteggio(a,d,e,f,g,1));
of<<"Operazione: "<<a<<"Valuta: "<<d<<"Importo: "<<e<<"Durata: "<<f<<"Orario: "<<g
<<"Probabilita: "<<prop<<"Frodi attese:"<<getConteggio(a,d,e,f,g,1)<<"\n";}}}}}of.close();
}
void Grafo::printFile(const char* result)
{
```

```
ofstream of(result);of<<"Risultati:\n";of.close();printStima(result);
of.open(result,ios::out|ios::app);of<<"\n";
of<<"Log-verosimiglianza: "<<logVerosimiglianza(campstima)<<"\n";
of<<"Conteggi totali:"<<totaleStimati()<<"\n";of<<"Insieme di stima:\n";of.close();
campstima->printFile(result);of.open(result,ios::out|ios::app);
of<<"Insieme di verifica:\n";of.close();
campverifica->printFile(result);printProbFrode(result);
}
```

Appendice C

Programma HMM

Per la stima degli Hidden Markov Models è stato sviluppato il programma `HMM` in linguaggio C++, le cui funzionalità sono state discusse in precedenza. Di seguito è riportato il codice del programma. Per questioni di spazio, l'indentazione è stata parzialmente eliminata.

```
/** classi Campo e Dati comuni con il programma Grafo*/

/**classe che contiene alcune utili funzioni di calcolo*/
class Utilita
{
public:
/**calcola la probabilita che una v.a. Binomiale(n,p) assuma valore k*/
static long double distribuzionebinomiale(int n,int k, double p);
/**setta un seme per la generazione di numeri casuali*/
static void setSeme();
/**data una soglia compresa tra 0 e 1,
genera un numero casuale compreso tra soglia e 1-soglia*/
static double casuale(double soglia);
};
/**implementazione della classe Utilita*/
#include"Utilita.h"#include<cmath>#include<ctime>#include<cstdlib>
void Utilita::setSeme()
{
srand((int)time(NULL));
}
double Utilita::casuale(double soglia)
{
double cc=0;while(cc<=soglia||cc>=(1-soglia)){cc=((double)(rand()%100))/(100);}
return cc;
}
long double Utilita::distribuzionebinomiale(int n,int k,double p)
{
long double pk=exp(log(p)*k);long double npk=exp(log(1-p)*(n-k));
if(k>(n/2)){k=n-k;}if(k==0){return pk*npk;}else{long double valore=pk*npk;
for(int i=0;i<k;i++){valore=valore*(n-i);valore=valore/(i+1);}return valore;}
}

#include"Dati.h"
/** classe che definisce un campione di osservazioni*/
class Campione
{
public:
/** ritorna il numero di osservazioni contenute nel campione*/
int getTotale();
```

```
/** estrae un campione di num osservazioni dal datest dat partendo dalla posizione pos*/
Campione(Dati* dat, int num, int pos);
/** distruttore della classe Campione*/
~Campione();
/** costruisce la matrice di confusione*/
void creaMatriceConfusione();
/** stampa a schermo le informazioni sul campione*/
void print();
/** stampa sul file s le informazioni sul campione*/
void print(const char* s);
/** per ciascuna osservazione stampa su schermo la variabile Y_t*/
void printY();
/** per ciascuna osservazione stampa sul file s la variabile Y_t*/
void printY(const char* s);
/** per ciascuna osservazione stampa su schermo la variabile Frode osservata*/
void printYvero();
/** per ciascuna osservazione stampa sul file s la variabile Frode osservata*/
void printYvero(const char* s);
/** per ciascuna osservazione stampa su schermo la variabile Frode prevista*/
void printYY();
/** per ciascuna osservazione stampa sul file s la variabile Frode prevista*/
void printYY(const char* s);
/** stampa su schermo la matrice di confusione*/
void printMatriceConfusione();
/** stampa sul file s la matrice di confusione*/
void printMatriceConfusione(const char* s);
/** ritorna lo stato Y_t dell'i-esima osservazione*/
int getY(int i);
/** imposta la variabile Frode prevista a val per l'osservazione i*/
void setYY(int i, int val);
private:
int totale;
int* YY;
int* Yvero;
int* Y;
int tabellaconfusione[2][2];
double errore;
double alpha;
double beta;
};
/** implementazione della classe Campione*/
#include"Campione.h"#include<fstream>#include<iostream>using namespace std;
void Campione::print(const char* s)
{
printYY(s);printYvero(s);printMatriceConfusione(s);
}
voidCampione::print()
{
printYY();printYvero();printMatriceConfusione();
}
void Campione::printMatriceConfusione(const char* s)
{
ofstream f;f.open(s,ios::out|ios::app);f<<"Matrice confusione:\n";
f<<tabellaconfusione[0][0]<<" "<<tabellaconfusione[0][1]<<"\n";
f<<tabellaconfusione[1][0]<<" "<<tabellaconfusione[1][1]<<"\n";
f<<"alpha: "<<alpha<<"beta: "<<beta<<"\n";f<<"Errore totale:"<<errore<<"\n";f.close();
}
void Campione::printMatriceConfusione()
{
cout<<"Matrice confusione:\n";
cout<<tabellaconfusione[0][0]<<" "<<tabellaconfusione[0][1]<<"\n";
cout<<tabellaconfusione[1][0]<<" "<<tabellaconfusione[1][1]<<"\n";
cout<<"alpha: "<<alpha<<"beta: "<<beta<<"\n";cout<<"Errore totale:"<<errore<<"\n";
}
```

```
void Campione::printY()
{
cout<<"Dati estratti:\n";for(int s=0;s<totale;s++){cout<<Y[s]<<" ";}cout<<"\n";
}
voidCampione::printY(constchar*s)
{
ofstream f;f.open(s,ios::out|ios::app);f<<"Dati estratti:\n";
for(int s=0;s<totale;s++){f<<Y[s]<<" ";}f<<"\n";f.close();
}
void Campione::printYvero()
{
cout<<"Sequenza reale:\n";for(int s=0;s<totale;s++){cout<<Yvero[s]<<" ";}cout<<"\n";
}
void Campione::printYvero(const char* s)
{
ofstream f;f.open(s,ios::out|ios::app);f<<"Sequenza reale:\n";
for(int s=0;s<totale;s++){f<<Yvero[s]<<" ";}f<<"\n";f.close();
}
void Campione::printYY()
{
cout<<"Stima di massima verosimiglianza:\n";for(int s=0;s<totale;s++)
{cout<<YY[s]<<" ";}cout<<"\n";
}
void Campione::printYY(const char* s)
{
ofstream f;f.open(s,ios::out|ios::app);f<<"Stima di massima verosimiglianza:\n";
for(int s=0;s<totale;s++){f<<YY[s]<<" ";}f<<"\n";f.close();
}
intCampione::getTotale()
{
return totale;
}
Campione::~Campione()
{
delete[]Y;delete[]YY;delete[]Yvero;
}
Campione::Campione(Dati* dat,int num,int pos)
{
totale=num;Y=new int[totale];YY=new int[totale];Yvero=new int[totale];
for(int i=0;i<totale;i++){Y[i]=dat->getStato(i+pos);if(dat->getFrode(i+pos))
{Yvero[i]=1;}else{Yvero[i]=0;}}
}
void Campione::creaMatriceConfusione()
{
tabellaconfusione[0][0]=0;tabellaconfusione[1][0]=0;tabellaconfusione[0][1]=0;
tabellaconfusione[1][1]=0;for(int i=0;i<totale;i++){if(Yvero[i]==1){if(YY[i]==1)
{tabellaconfusione[0][0]++;}else{tabellaconfusione[1][0]++;
}}else{if(YY[i]==1){tabellaconfusione[0][1]++;}else{tabellaconfusione[1][1]++;}}}
errore=((double)(tabellaconfusione[0][1]+tabellaconfusione[1][0]))/
((double)(tabellaconfusione[0][1]+tabellaconfusione[1][0]+
tabellaconfusione[1][1]+tabellaconfusione[0][0]));
alpha=((double)(tabellaconfusione[1][0]))/
((double)(tabellaconfusione[0][0]+tabellaconfusione[1][0]));
beta=((double)(tabellaconfusione[0][1]))/
((double)(tabellaconfusione[0][1]+tabellaconfusione[1][1]));
}
int Campione::getY(int i)
{
return Y[i];
}
void Campione::setYY(int i,int val)
{
YY[i]=val;
}
```

```
/** classe che definisce un Hidden Markov Model
#include"Campione.h"
class HMM
{
public:
/**numero di stati Y_t*/
static const int VAR=120;
/** stampa le matrici A, B e pi sul file s*/
void print(const char* s);
/** stampa le matrici A, B e pi sullo schermo*/
void print();
/**costruisce la matrice B_0 con una distribuzione binomiale di parametri p1
per la prima riga e p2 per la seconda riga*/
void assegnaBinomiale(double p1,double p2);
/**costruisce un HMM la cui matrice A ha sulla diagonale i valori a1 e a2 e la cui
matrice B ha le righe distribuite come binomiali di parametri p1 e p2*/
HMM(double a1,double a2,double p1, double p2);
/**costruisce la matrice A con i valori a1 e a2 sulla diagonale principale*/
void assegnaA(double a1,double a2);
/**ritorna i coefficienti alpha_t per il campione camp*/
double* alpha(int t,Campione* camp);
/**ritorna i coefficienti alpha_t aggiustati per il campione camp*/
double* alphatilde (int t,Campione* camp);
/**ritorna i coefficienti beta_t per il campione camp*/
double* beta(int t,Campione* camp);
/**ritorna i coefficienti beta_t aggiustati per il campione camp*/
double* betatilde(int t,Campione* camp);
/**ritorna i coefficienti gamma_t per il campione camp*/
double* gamma(int t,Campione* camp);
/** ritorna i coefficienti epsilon_t per il campione camp*/
double** epsilon(int t,Campione* camp);
/**esegue l'algoritmo di Baum-Welch sul campione camp per numIt iterazioni*/
void stima(int numIt,Campione* camp);
/** calcola la log-verosimiglianza per il campione camp*/
double logVerosimiglianza(Campione* camp);
/**classifica le osservazioni del campione camp*/
void massimaVerosimiglianza(Campione* camp);
/**ritorna l'ultima log-verosimiglianza calcolata*/
double getLogVerosimiglianza();
private:
double A [2][2];
double B [2][VAR];
double PI[2];
double logverosimiglianza;
};
/** implementazione della classe HMM*/
#include<iostream>#include<fstream>#include<cmath>#include"HMM.h"using namespace std;
void HMM::print(const char* s)
{
ofstream f;f.open(s,ios::out|ios::app);f<<"A:"<<"\n";
f<<A[0][0]<<" "<<A[0][1]<<"\n";f<<A[1][0]<<" "<<A[1][1]<<"\n";
f<<"B:"<<"\n";for(int k=0;k<2;k++){for(int h=0;h<VAR;h++){f<<B[k][h]<<" ";}f<<"\n";}
f<<"PI:"<<"\n";f<<PI[0]<<" "<<PI[1]<<"\n";f.close();
}
voidHMM::print()
{
cout<<"A:"<<"\n";cout<<A[0][0]<<" "<<A[0][1]<<"\n";cout<<A[1][0]<<" "<<A[1][1]<<"\n";
cout<<"B:"<<"\n";for(int k=0;k<2;k++){for(int h=0;h<VAR;h++){cout<<B[k][h]<<" ";}
cout<<"\n";}cout<<"PI:"<<"\n";cout<<PI[0]<<" "<<PI[1]<<"\n";
}
voidHMM::assegnaBinomiale(double p1,double p2)
{
for(int i=0;i<VAR;i++){B[0][i]=Utilita::distribuzionebinomiale(VAR-1,i,p1);
```

```
B[1][i]=Utilita::distribuzionebinomiale(VAR-1,i,p2);}
}
HMM::HMM(double a1,double a2,double p1,double p2)
{
assegnaA(a1,a2);assegnaBinomiale(p1,p2);PI[0]=0.5;PI[1]=0.5;
}
voidHMM::assegnaA(double a1,double a2)
{
A[0][0]=a1;A[0][1]=1-a1;A[1][0]=1-a2;A[1][1]=a2;
}
double* HMM::alpha(int t,Campione* camp)
{
double* vet=new double[2];double a[2];double aux[2];int k=camp->getY(0)-1;
a[0]=PI[0]*B[0][k];a[1]=PI[1]*B[1][k];for(int n=2;n<=t;n++){k=camp->getY(n-1)-1;
aux[0]=(A[0][0]*a[0]+A[1][0]*a[1])*B[0][k];aux[1]=(A[0][1]*a[0]+A[1][1]*a[1])*B[1][k];
a[0]=aux[0];a[1]=aux[1];}vet[0]=a[0];vet[1]=a[1];return vet;
}
double* HMM::alphatilde(int t,Campione* camp)
{
double*vet=new double[2];double a[2];intk=camp->getY(0)-1;
a[0]=PI[0]*B[0][k];a[1]=PI[1]*B[1][k];doublec0=1/(a[0]+a[1]);
vet[0]=c0*a[0];vet[1]=c0*a[1];for(int i=2;i<=t;i++){k=camp->getY(i-1)-1;
a[0]=((vet[0]*A[0][0])+(vet[1]*A[1][0]))*B[0][k];
a[1]=((vet[0]*A[0][1])+(vet[1]*A[1][1]))*B[1][k];c0=1/(a[0]+a[1]);
vet[0]=c0*a[0];vet[1]=c0*a[1];}return vet;
}
double* HMM::beta(int t,Campione* camp)
{
double*vet=new double[2];double b[2];double bux[2];int k;int N=camp->getTotale();
b[0]=1;b[1]=1;if(t<N){for(int n=N-1;n>=t;n--){k=camp->getY(n)-1;
bux[0]=A[0][0]*B[0][k]*b[0]+A[0][1]*B[1][k]*b[1];
bux[1]=A[1][0]*B[0][k]*b[0]+A[1][1]*B[1][k]*b[1];
b[0]=bux[0];b[1]=bux[1];}}vet[0]=b[0];vet[1]=b[1];return vet;
}
double* HMM::betatilde(int t,Campione* camp)
{
double* vet=new double[2];double b[2];int N=camp->getTotale();
b[0]=1;b[1]=1;double c0=1/(b[0]+b[1]);vet[0]=b[0]*c0;vet[1]=b[1]*c0;
if(t<N){for(int i=N-1;i>=t;i--){int k=camp->getY(i)-1;
b[0]=A[0][0]*B[0][k]*vet[0]+A[0][1]*B[1][k]*vet[1];
b[1]=A[1][0]*B[0][k]*vet[0]+A[1][1]*B[1][k]*vet[1];
c0=1/(b[0]+b[1]);vet[0]=b[0]*c0;vet[1]=b[1]*c0;}}return vet;
}
double* HMM::gamma(int t,Campione* camp)
{
double* vet=new double[2];double* a=alphatilde(t,camp);double* b=betatilde(t,camp);
double ab=a[0]*b[0]+a[1]*b[1];vet[0]=(a[0]*b[0])/(ab);
vet[1]=(a[1]*b[1])/(ab);delete[]a;delete[]b;return vet;
}
double** HMM::epsilon(int t,Campione* camp)
{
double** vet=new double*[2];vet[0]=new double[2];vet[1]=new double[2];
double* a=alphatilde(t,camp);double*b=betatilde((t+1),camp);int k=camp->getY(t)-1;
double somma=a[0]*A[0][0]*B[0][k]*b[0]+a[0]*A[0][1]*B[1][k]*b[1]+
a[1]*A[1][0]*B[0][k]*b[0]+a[1]*A[1][1]*B[1][k]*b[1];
vet[0][0]=(a[0]*A[0][0]*B[0][k]*b[0])/(somma);
vet[1][1]=(a[1]*A[1][1]*B[1][k]*b[1])/(somma);
vet[1][0]=(a[1]*A[1][0]*B[0][k]*b[0])/(somma);
vet[0][1]=(a[0]*A[0][1]*B[1][k]*b[1])/(somma);
delete[]a;delete[]b;return vet;
}
void HMM::stima(int NUMIt,Campione* camp)
{
int TT=camp->getTotale();for(int s=0;s<NUMIt;s++){double ee[2][2];for(int i=0;i<2;i++)
```

```
{for(int j=0;j<2;j++){ee[i][j]=0;}}double g1=0;double g2=0;double gamma1[2];
gamma1[0]=0;gamma1[1]=0;for(int r=1;r<TT;r++){double** et=epsilon(r,camp);
for(int i=0;i<2;i++){for(int j=0;j<2;j++){ee[i][j]+=et[i][j];}}delete[]et[0];
delete[]et[1];delete[]et;double*gt=gamma(r,camp);g1+=gt[0];g2+=gt[1];gamma1[0]+=gt[0];
gamma1[1]+=gt[1];delete[]gt;}double* ggg=gamma(TT,camp);gamma1[0]+=ggg[0];gamma1[1]+=ggg[1];
delete[]ggg;double gamma2[2][VAR];for(int i=0;i<2;i++){for(int j=0;j<VAR;j++){gamma2[i][j]=0;}}
for(int j=0;j<VAR;j++){for(int k=0;k<TT;k++){if((camp->getY(k)-1)==j){double* etc=gamma((k+1),camp);
gamma2[0][j]+=etc[0];gamma2[1][j]+=etc[1];delete[]etc;}}}double*PP=gamma(1,camp);
for(int i=0;i<2;i++){for(int j=0;j<VAR;j++){B[i][j]=gamma2[i][j]/gamma1[i];}}A[0][0]=ee[0][0]/g1;
A[0][1]=ee[0][1]/g1;A[1][0]=ee[1][0]/g2;A[1][1]=ee[1][1]/g2;PI[0]=PP[0];
PI[1]=PP[1];delete[]PP;logverosimiglianza=logVerosimiglianza(camp);
cout<<"Iterazione: "<<s<<"log-verosimiglianza: "<<logverosimiglianza<<"\n";}
}
double HMM::logVerosimiglianza(Campione* camp)
{
long double logver=0;intT=camp->getTotale();double a[2];double vet[2];
int k=camp->getY(0)-1;a[0]=PI[0]*B[0][k];a[1]=PI[1]*B[1][k];doublec0=1/(a[0]+a[1]);
vet[0]=c0*a[0];vet[1]=c0*a[1];logver+=(-1*log((long double)c0));for(int i=2;i<=T;i++)
{k=camp->getY(i-1)-1;a[0]=((vet[0]*A[0][0])+(vet[1]*A[1][0]))*B[0][k];
a[1]=((vet[0]*A[0][1])+(vet[1]*A[1][1]))*B[1][k];c0=1/(a[0]+a[1]);vet[0]=c0*a[0];
vet[1]=c0*a[1];logver+=(-1*log((longdouble)c0));}return logver;
}
void HMM::massimaVerosimiglianza(Campione*camp)
{
int T=camp->getTotale();for(int i=0;i<T;i++){double* gg=gamma((i+1),camp);
if(gg[0]>=gg[1]){camp->setYY(i,0);}else{camp->setYY(i,1);}delete[]gg;}
camp->creaMatriceConfusione();
}

#include"HMM.h"
/**classe che estrae un campione di osservazioni e stima l'HMM con l'algoritmo di Baum Welch
oppure trova il miglior HMM iniziale con il metodo Monte Carlo*/
class TrovaHMM
{
private:
HMM* modello;
int numIt;
int leng;
Dati* dato;
Campione* camp;
double ah0;
double ah1;
double ph1;
double ph2;
public:
/**estrae un campione di numOss osservazioni a partire dall'osservazione inizio nel dataset,
per poi eseguire num iterazioni dell'algoritmo di Baum Welch*/
TrovaHMM(int numOss,int inizio,int num);
/** distruttore della classe TrovaHMM*/
~TrovaHMM();
/**costruisce l'HMM con i valori iniziali assegnati*/
void creaModello(double aaa1,double aaa2, double ppp1, double ppp2);
/**stima l'HMM con Baum Welch partendo dai valori iniziali assegnati*/
void stima(double aaa1,double aaa2, double ppp1, double ppp2);
/**trova il miglior punto iniziale usando il metodo MonteCarlo,
eseguendo numero iterazioni e generando numeri casuali compresi tra soglia e 1-soglia*/
void trova(int numero, double soglia);
/** stampa l'HMM su schermo*/
void print();
/**stampa l'HMM sul file t*/
void print(const char* t);
/**stampa i risultati sullo schermo*/
void printtutto();
/**stampa i risultati sul file t*/
```

```
void printtutto(const char* result);
};
/**implementazione dela classe TrovaHMM*/
#include<iostream>#include<fstream>#include<cmath>
#include"TrovaHMM.h"using namespace std;
TrovaHMM::~TrovaHMM()
{
delete camp;delete dato;delete modello;
}
TrovaHMM::TrovaHMM(int numOss,int inizio,int num)
{
ah0=0;ah1=0;ph1=0;ph2=0;numIt=num;modello=NULL;dato=newDati("FRAUD.txt");
camp=new Campione(dato,numOss,inizio);
}
void TrovaHMM::creaModello(double aaa1,double aaa2,double ppp1,double ppp2)
{
ah0=aaa1;ah1=aaa2;ph1=ppp1;ph2=ppp2;delete modello;modello=new HMM(ah0,ah1,ph1,ph2);
}
void TrovaHMM::stima(double aaa1,double aaa2,double ppp1,double ppp2)
{
creaModello(aaa1,aaa2,ppp1,ppp2);modello->stima(numIt,camp);
modello->massimaVerosimiglianza(camp);
constchar*result;result="result.txt";printtutto();printtutto(result);delete result;
}
void TrovaHMM::trova(int numero,double soglia)
{
Utilita::setSeme();double a0=Utilita::casuale(soglia);double a1=Utilita::casuale(soglia);
double p1=Utilita::casuale(soglia);double p2=Utilita::casuale(soglia);
double aa1,aa2,pp1,pp2;creaModello(a0,a1,p1,p2);
cout<<"Iterazione 0. Stima modello in corso con: a1=" <<a0
<<" a2= "<<a1<<" p1= "<<p1<<" p2= "<<p2<<"\n";
modello->stima(numIt,camp);HMM* hm1;for(int i=1;i<numero;i++)
{aa1=Utilita::casuale(soglia);aa2=Utilita::casuale(soglia);pp1=Utilita::casuale(soglia);
pp2=Utilita::casuale(soglia);hm1=new HMM(aa1,aa2,pp1,pp2);
cout<<"Iterazione "<<i<<". Stima modello in corso con: a1= "<<aa1
<<" a2= "<<aa2<<" p1= "<<pp1<<" p2= "<<pp2<<"\n";
hm1->stima(numIt,camp);if(hm1->getLogVerosimiglianza()>modello->getLogVerosimiglianza())
{modello=hm1;a0=aa1;a1=aa2;p1=pp1;p2=pp2;}else{delete hm1;}}ah0=a0;ah1=a1;ph1=p1;ph2=p2;
cout<<"Migliori valori iniziali:\n";
cout<<"a0="<<ah0<<"a1="<<ah1<<"p1="<<ph1<<"p2="<<ph2<<"\n";
modello->massimaVerosimiglianza(camp);const char*result;result="miglior_result.txt";}
printtutto(result);delete result;
}
void TrovaHMM::print()
{
modello->print();
}
void TrovaHMM::print(const char* t)
{
modello->print(t);
}
voidTrovaHMM::printtutto()
{
cout<<"Migliori valori iniziali:\n";
cout<<"a0="<<ah0<<"a1="<<ah1<<"p1="<<ph1<<"p2="<<ph2<<"\n";
camp->printY();cout<<"Il modello stimato e':\n";print();camp->print();
}
void TrovaHMM::printtutto(const char* result)
{
ofstream of(result);of<<"Migliori valori iniziali:\n";
of<<"a0="<<ah0<<"a1="<<ah1<<"p1="<<ph1<<"p2="<<ph2<<"\n";
of.close();camp->printY(result);of.open(result,ios::out|ios::app);
of<<"Il modello stimato e':\n";of.close();print(result);camp->print(result);
}
```

Bibliografia

[1] Baum Leonard E., Petrie Ted, Soules George, Weiss Norman, A Maximization Technique Occurring in the Statistical Analysis of Probabilistic Functions of Markov Chains, *The Annals of Matematical Statistics*, 1970

[2] Darroch, J. N., Speed, T. P., Additive and Multiplicative Models and Interactions, *The Annals of Statistics*, Vol. 11, No. 3, Adelaide, 1983

[3] Deming, Edward W., Stephan, Frederick F., On a Least Squares Adjustment on a Sampled Frequency Table When the Expected Marginal Totals Are Known,*The Annals of Mathematical Statistics*, 1940

[4] Durbin, Richard, *Biological Sequence Analysis: Probabilistic Models of Proteins and Nucleic Acids*, Cambridge University Press,1998

[5] Haberman, Shelby J, Log-Linear Models for Frequency Data: Sufficient Statistics and Likelihood Equations, *The Annals of Statistics*, Vol. 1, No. 4, Chicago, 1973

[6] Hojsgaard, Soren, Edwards David, Lauritzen, Steffen, *Graphical Models with R*, Springer 2012

[7] Lauritzen, Steffen L., *Graphical Models*, Claredon Press, Oxford, 1996

[8] Lauritzen, Steffen L., *Lectures on Contingency Tables (Electronic edition)*, Klitgaarden, Skagen, 2002

[9] Matus, F., *On Equivalence of Markov Properties over Undirected Graphs*, Institute of Information Theory and Automation, Praga, 1992

[10] Rabiner, Lawrence R., A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition, *Proceedings of the ieee*, Vol.77, No. 2, 1989

[11] Ruzza Marco, *Inferenza Statistica per Hidden Markov Models*, Università di Padova, Falcoltà di Ingegneria, 2010-2011

[12] Stamp Mark, *A Revealing Introduction to Hidden Markov Models*, San Jose State University, 2012