Università degli Studi di Padova

Dipartimento di Scienze Statistiche

Corso di Laurea Magistrale in

Scienze Statistiche

Stima non parametrica dell'area sotto la curva ROC specifica per caratteristiche in parziale assenza di Gold Standard

Relatore Prof. Gianfranco Adimari
Dipartimento di Scienze Statistiche

Laureando: Stefano Mussi
Matricola N. 1066284

Anno Accademico 2014/2015

Ai miei fratelli

Possano fare sempre meglio di me!

Riassunto

La curva ROC è usata per valutare la capacità di un test diagnostico, tipicamente continuo, di discriminare correttamente i soggetti sani da quelli malati. Una sua estensione è la curva ROC specifica per caratteristiche che permette di vagliare un test rispetto a delle caratteristiche specifiche del soggetto. Nei test di screening su larga scala la determinazione dello stato di salute di tutti i soggetti non sempre è possibile a causa del costo della o della natura invasiva del processo. L'utilizzo dei soli dati completi nell'analisi, d'altra parte, comporta tipicamente quella che viene chiamata distorsione di verifica.

Per correggere questa distorsione sono stati proposti in letteratura dei metodi che però necessitano di modelli parametrici per la probabilità condizionata di essere malati e/o per quella di essere verificati (cioè di avere una valutazione certa dello stato di malattia). Una scorretta specificazione di questi modelli può compromettere la bontà degli stimatori, che potrebbero essere inconsistenti.

Per evitare questo problema si propone qui un metodo completamente non-parametrico basato sulla imputazione dei vicini più vicini per le probabilità di essere malati. Queste probabilità sono state poi inserite nelle equazioni di stima che verranno utilizzate per la stima dei parametri necessari alla valutazione dell'area sotto la curva ROC specifica per caratteristiche (AUC_x). Le stesse probabilità sono state inoltre inserite nell'espressione per la stima dell'area sotto la curva ROC aggiustata (AAUC), che non è altro che la media pesata dell'AUC_x per i soggetti malati. Per tutte le stime è stato inoltre proposto un metodo basato sulla tecnica bootstrap per il calcolo de-

gli standard error e del livello di significatività osservato relativo a eventuali test d'ipotesi.

Si è studiata la bontà delle tecniche utilizzate nel caso in cui i dati siano "missing at random" (MAR) e si sono condotte delle simulazioni per valutarne le prestazioni.

Infine si è applicato il metodo a dei dati reali provenienti da un dataset per la ricerca sull'Alzheimer.

Indice

Capitolo 1

Curva ROC covariata-specifica

Nel campo medico è di grande importanza la valutazione della capacità di un test diagnostico, cioè un test in grado di discriminare tra sani e malati. L'accuratezza del test viene studiata mettendola a confronto con un gold standard test, che è il test che discrimina senza errore. Le problematiche maggiori del gold standard test sono il costo eccessivo o l'elevata invasività, che spesso determinano l'impossibilità di effettuarlo su tutti i pazienti. Se per valutare il test diagnostico alternativo, però, si utilizzassero solo i dati relativi ai soggetti verificati, la valutazione potrebbe essere affetta da una distorsione che è definita come distorsione di verifica. Per ogni osservazione si ha il seguente insieme di dati: T, il valore del test; V, che vale uno se al soggetto è stato effettuato anche il test gold standard e zero in caso contrario; D, che è presente solo se $V = 1$, e indica il vero stato di salute del soggetto (tipicamente con il valore uno si indica lo stato di malattia con zero l'assenza di quest'ultima); X che sono le altre informazioni che si hanno ha disposizione per quel specifico paziente.

1.1 Curva ROC

La curva ROC è uno strumento molto popolare per la valutazione del test diagnostico su scala continua. Questo test può essere dicotimizzato tramite una soglia, punto di "cut-off", sopra la quale il soggetto in esame è conside-

rato malato. Per ogni valore possibile del valore di cut-off si possono stimare i valori di sensibilità, ovvero la probabilità di identificare correttamente un soggetto malato, e la specificità, ovvero la probabilità di classificare correttamente un soggetto sano. La coppia (1-specificità,sensibilità) forma i punti della curva ROC.

$$se(c) = Pr(T > c|D = 1) \;\; sp(c) = Pr(T < c|D = 0) \tag{1.1}$$

$$ROC(\cdot) = \{(1 - sp(c), se(c)), c \in (-\inf, +\inf)\} \tag{1.2}$$

Prendiamo ad esempio i dati della tabella 1.1: la prima riga corrisponde al gold standard, la seconda al test effettuato e la terza a una covariata categoriale che verrà usata successivamente.

	1	2	3	4	5	6	7	8	9	10
Gold	1	0	1	0	1	0	1	1	1	1
Test	1.19	0.86	3.15	2.23	0.99	2.05	2.23	0.72	2.68	1.07
X	0	0	0	1	0	1	0	0	1	1

	11	12	13	14	15	16	17	18	19	20
Gold	1	0	0	0	0	0	1	0	0	1
Test	1.40	0.32	1.07	1.46	0.99	1.98	2.45	0.83	-1.28	2.21
X	0	0	0	0	1	1	1	1	0	1

Tab. 1.1:

Calcoliamo per un insieme di valori di cut-off, compresi fra -1.3 e 3.5 in modo da coprire il dominio empirico dei valori del test, i valori di sensibilità e specificità. Per farlo verrà costruita una tabella due per due, chiamata matrice di confusione, dove sulle colonne i soggetti vengono divisi per il risultato del gold test, nella prima riga verranno inseriti i soggetti per i quali il test in esame ha dato un valore minore del cut-off e nella seconda quelli

con un valore maggiore. Si otterranno quindi quattro valori: i veri negativi (VN), cioè quei soggetti sani che il test ha classificato correttamente; i falsi positivi (FP), i soggetti sani classificati malati; i veri positivi (VP) i soggetti malati classificati correttamente e infine i malati considerati sani, i falsi negativi (FN) (Tabella 1.1). Possiamo dare quindi dare una nuova espressione per il calcolo della specificità e sensibilità (1.3) e disegnare la curva Roc (figura 1.1).

	GOLD=0	GOLD=1
Test<c	VN	FN
Test<c	FP	VP

Tab. 1.2:

$$se = \frac{VP}{VP + FN} \quad sp = \frac{VN}{VN + FP} \tag{1.3}$$

	1	2	3	4	5	6	7	8	9	10	11	12
c	-1.3	-1.1	-0.9	-0.7	-0.5	-0.3	-0.1	0.1	0.3	0.5	0.7	0.9
se(c)	0.0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.2	0.2	0.4
sp(c)	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.9
	13	14	15	16	17	18	19	20	21	22	23	24	25
c	1.1	1.3	1.5	1.7	1.9	2.1	2.3	2.5	2.7	2.9	3.1	3.3	3.5
se(c)	0.6	0.6	0.7	0.7	0.7	0.9	1.0	1.0	1.0	1.0	1.0	1.0	1.0
sp(c)	0.7	0.6	0.5	0.5	0.5	0.5	0.3	0.2	0.1	0.1	0.1	0.0	0.0

Tab. 1.3: Valori della specificità e sensibilità per una scala di valori del punto di cut-off

L'area al di sotto di questa curva (AUC) può essere utilizzata per valutare la bontà del test diagnostico e può essere interpretata come la probabilità che un soggetto malato estratto a caso abbia il valore del test più alto rispetto ad un altro soggetto sano sempre scelto casualmente. Il valore dell'AUC

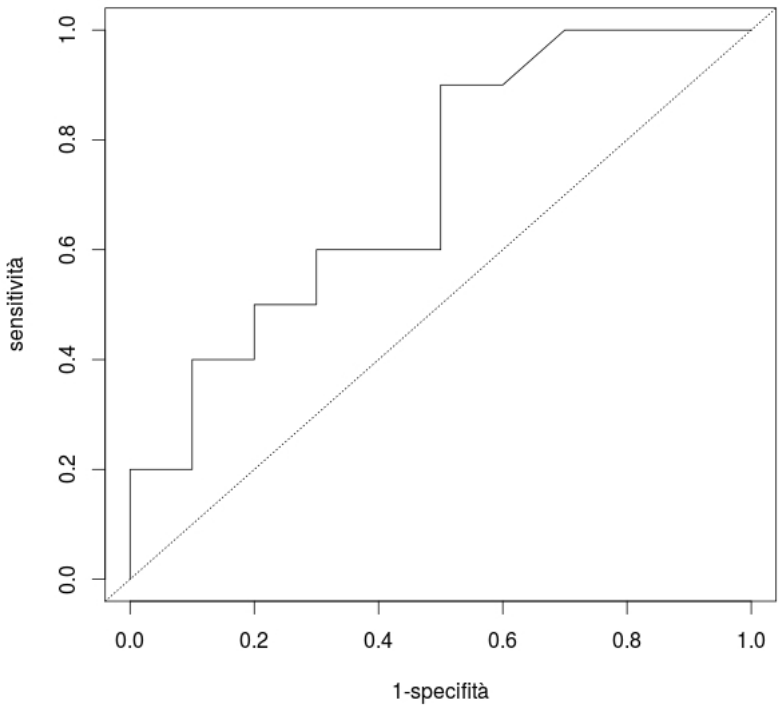


Fig. 1.1: curva ROC

varia tra 0.5, che corrisponde a un test completamente non-informativo, e 1, che corrisponde a un test equivalente in tutto per tutto al gold-standard. Se il valore dell'AUC scende sotto il valore di 0.5 si è in presenza di una scorretta formulazione del test, che considera malati i soggetti per valori piccoli del test: il problema può essere velocemente superato cambiando di segno al valore del test.

Operativamente, se non ci sono paricolari restrizioni parametriche, per calcolare l'AUC si può utilizzare il test sui ranghi di Mann-Whitney. Sia M il vettore con i risultati dei test per i malati e S quello con i risultati dei sani e n_M e n_S le rispettive numerosità, uno stimatore per l'AUC è:

$$\hat{AUC} = 1 - \frac{T_{MW}(S, M)}{n_M n_S} \tag{1.4}$$

Con i dati utilizzati prima risulta che $\hat{AUC} = 0.71$

Nella Fig.1.2 vengono riportate tre curve ROC per tre diversi test. Nei grafici nella prima riga viene riportata la distribuzione del test per i sani, la curva in chiaro, e quella per i malati, la curva più scura. Nel primo riquadro a sinistra le due distribuzioni sono nettamente separate e infatti l'AUC nel grafico sottostante, occupando tutto il grafico, vale uno. Nel riquadro centrali le due distribuzioni hanno una parte dello spazio campionario in comune, come nell'esempio prima trattato, ma è evidente una certa capacità del test. Nell'ultimo grafico le distribuzioni coincidono, indice della incapacità del test di distinguere fra sani e malati, e infatti la curva ROC coincide con la bisettrtice e l'AUC vale un mezzo.

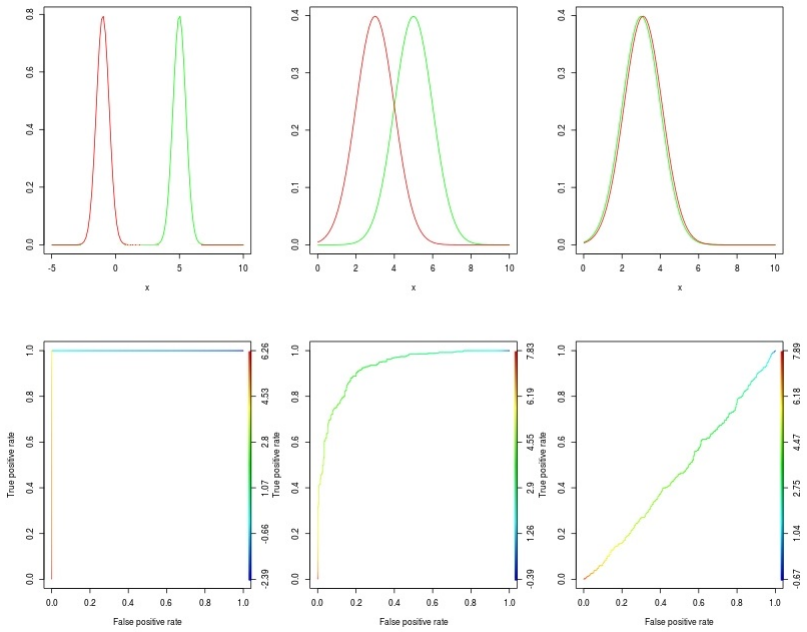


Fig. 1.2: curva ROC per tre diverse capacità del test

1.2 Estensione alle covariate

La capacità discriminatoria di un test diagnostico può dipendere, oltre che dal test stesso, anche dalle caratteristiche proprie del soggetto; ad esempio potrebbe funzionare meglio per le donne rispetto agli uomini. Per

studiare questo fenomeno si è quindi introdotta la curva ROC specifica per caratteristiche (ROC_x) e il valore dell'area sottesa (AUC_x).

L'AUC_x può essere interpretata quindi come la probabilità che il valore del test di un soggetto malato sia maggiore rispetto ad uno sano, partendo dal presupposto che condividano le covariate. Se si contempla la possibilità che il test abbia lo stesso valore per due soggetti differenti è necessaria una correzione e alla probabilità che un soggetto malato abbia il valore del test maggiore di uno sano si aggiunge metà della probabilità che il test dia lo stesso risultato. Si può quindi scrivere:

$$AUC_x \equiv \nu(x) = Pr(T_1 > T_2 | D_1 = 1, D_2 = 0, X_1 = X_2 = x) + \frac{1}{2} Pr(T_1 = T_2 | D_1 = 1, D_2 = 0, X_1 = X_2 = x) \quad (1.5)$$

Ora assumiamo che l'AUC_x possa essere descritta dalla seguente forma lineare generalizzata:

$$AUC_x \equiv \nu(x) = g^{-1}((1, x)^T \theta^*) \quad (1.6)$$

$g(\cdot)$ è una funzione legame strettamente monotona e θ^* è un vettore di parametri. Per la scelta di $g(\cdot)$ in Liu e Zhou (2013) viene suggerito di riferirsi o a una logit o a una probit.

L'espressione (1.5) può essere facilmente estesa se si è interessati a studiare la capacità del test per delle covariate di tipo continuo; siano x le covariate del controllo e y quelle del caso:

$$AUC_x \equiv \xi(x, y) = Pr(T_1 > T_2 | D_1 = 1, D_2 = 0, X_1 = x, X_2 = y) + \frac{1}{2} Pr(T_1 > T_2 | D_1 = 1, D_2 = 0, X_1 = x, X_2 = y) \quad (1.7)$$

Ovviamente è possibile e necessario estendere anche la (1.6) che diventerà:

$$\xi(x, y) = g^{-1}(W^T \theta) \quad (1.8)$$

Dove $W = (1, x^T, y^T)^T$; è facile notare che $\xi(x, x) = \nu(x)$.

Riprendendo il nostro esempio possiamo costruire le due curve ROC per i due diversi valori della X e dopo aver calcolato l'AUC_x per entrambi stimare

i valori di θ a partire dalla (1.6).

Usando sempre il test di Mann-Whitney si possono ricavare le stime di $\hat{AUC}_{x=0} = 0.77$ e $\hat{AUC}_{x=1} = 0.8$ e a partire da queste, utilizzando come funzione legame una funzione logit, le stime per $\hat{\theta}_0 = 1.15$, l'intercetta, e $\hat{\theta}_1 = 0.23$.

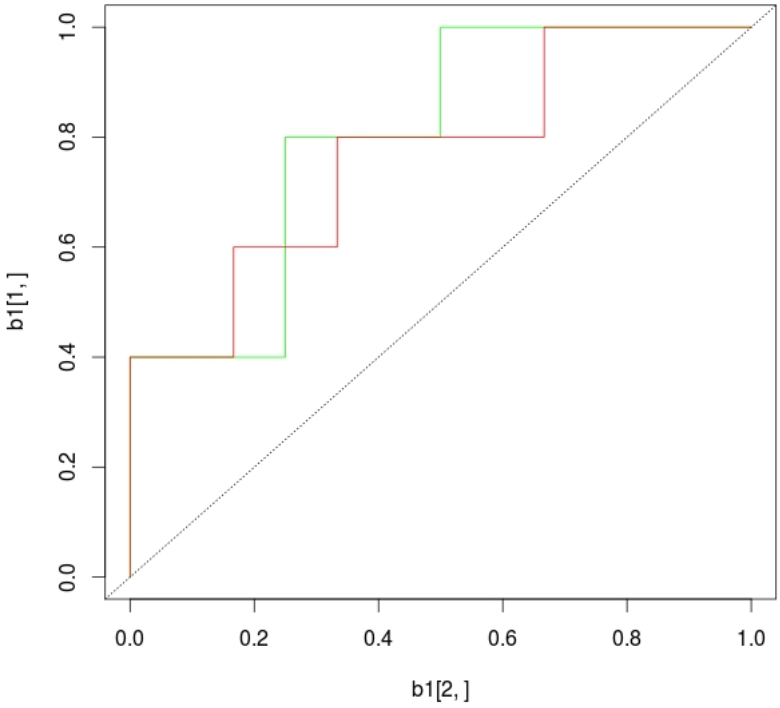


Fig. 1.3: curva ROC specifica per caratteristiche, in chiaro quella per $X = 1$ e in scuro quella per $X = 0$

1.3 AROC e AAUC

Partendo dalla ROC_x e dalla AUC_x è possibile inoltre calcolare due nuove quantità: la adjusted ROC (AROC(t)) e la adjusted AUC (AAUC). Seguendo il percorso tracciato in Janes e Pepe (2009) definiamo T_D e $T_{\bar{D}}$ come la realizzazione del test per i soggetti malati e per quelli sani rispettivamente

e $S_D(c) = Pr(T_D > c)$, $S_{\bar{D}}(c) = Pr(T_{\bar{D}} > c)$ le loro funzioni di sopravvivenza. Di queste ultime è facile vedere l'equivalenza a $se(c)$ e $1 - sp(c)$. Se c è il punto di cut-off e $1 - t$ il punto di specificità a lui associato, ricordando la (1.1) vale che $c = S_{\bar{D}}^{-1}(t)$. Ricordando la (1.2) la riscriviamo per un certo valore t di 1-specificità: $ROC(\cdot) = \{(t, ROC(t)), t \in (0, 1)\}$ dove

$$ROC(t) = S_D(S_{\bar{D}}^{-1}(t)) \tag{1.9}$$

Consideriamo ora X_D le covariate per un soggetto malato e $X_{\bar{D}}$ quelle per uno sano con funzioni di ripartizione $F_{X|D=1}$ e $F_{X|D=0}$. Siano inoltre $S_D(c|x) = Pr(T_D > c|X_D = x)$, per i soggetti sani , e $S_{\bar{D}}(c|x) = Pr(T_{\bar{D}} > c|X_{\bar{D}} = x)$, per quelli malati, le funzioni di sopravvivenza della distribuzione del test rispettivamente, con le rispettive inverse $S_{\bar{D}}^{-1}(t|x)$ e $S_D^{-1}(t|x)$.
La curva ROC aggiustata è definità come la probabilità che il valore del test di un soggetto malato con delle specifiche covariate sia maggiore di una certa soglia che corrisponde a quella che si otterrebbe fissando una certa frazione di falsi positivi con le stesse covariate. In termini matematici $AROC(t) = Pr\{T_D > S_{\bar{D}}^{-1}(t|X_D)\}$.
Per le covariate X=x e un valore di $1-sp$ si può dunque ridefinire $ROC_x(t) = S_D\{S_{\bar{D}}^{-1}(t|x)|x\}$. Integrando quest'ultima nei domini delle covariate si ottiene inoltre la seguente espressione:

$$\begin{aligned} AROC(t) &= \int_{-\infty}^{+\infty} Pr\{T_D > S_{\bar{D}}^{-1}(t|X_D)|X_D = x\} dF_{X|D=1}(x) && (1.10)\\ &= \int_{-\infty}^{+\infty} S_D\{S_{\bar{D}}^{-1}(t|x)|x\} dF_{X|D=1}(x) && (1.11)\\ &= \int_{-\infty}^{+\infty} ROC_x(t) dF_{X|D=1}(x) && (1.12) \end{aligned}$$

Come mostrato in Liu e Zhou (2013) si può dalla (1.12) ricavare un'espressione per l'AAUC

$$\begin{aligned} AAUC &= \int_0^1 AROC(t)dt \\ &= \int_0^1 \int_{-\infty}^{+\infty} ROC_x(t)dF_{X|D=1}(x)dt \\ &= \int_{-\infty}^{+\infty} AUC_x dF_{X|D=1}(x) \end{aligned} \tag{1.13}$$

Se la distribuzione dello stato di salute dei soggetti condizionato alle covariate è sconosciuta, per poter utilizzare la (1.13) è necessario sostituire la funzione di ripartizione $F_{X|D=1}(x)$ con quella empirica:

$$\hat{F}_{X|D=1}(x) = \frac{\sum_i I(X_i < x)D_i}{\sum_i D_i} \tag{1.14}$$

Di conseguenza l'AAUC viene stimata da:

$$A\hat{A}UC = \int_{\infty}^{\infty} A\hat{U}C_x d\hat{F}^{(est)}_{X|D=1}(x) = \frac{\sum_i A\hat{U}C_x D_i}{\sum_i D_i} \tag{1.15}$$

Come si può notare calcolando le medie di ROC_x e AUC_x, calcolate per ogni covariata possibile e pesate per la probabilità di essere malati date quelle covariate, si possono stimare l'$AROC$ e l'$AAUC$.

Riprendendo l'esempio numerico:

$$\begin{aligned} A\hat{A}UC &= \frac{\#(D=1|x=0)A\hat{U}C_{x=0} + \#(D=1|x=1)A\hat{U}C_{x=1}}{\#D=1} \\ &= \frac{0.77 \cdot 6 + 0.8 \cdot 4}{10} = 0.782 \end{aligned}$$

1.4 Equazioni di stima e metodi per correggere la distorsione di verifica

Si definisca $I_{ij} = \mathbf{I}(T_i > Tj) + \frac{1}{2}\mathbf{I}(T_i = T_j)$. Per la stima di θ, i parametri di (1.8), quando tutte le osservazioni sono state verificate, sempre in Liu e Zhou (2013) viene proposto il seguente sistema di equazioni di stima:

$$\sum_i \sum_{j \neq i} U_{ij} = \sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T} \right)^T \frac{(I_{ij} - \xi_{ij})}{(1 - \xi_{ij})\xi_{ij}} D_i(1 - D_j) = 0 \tag{1.16}$$

Come si può notare se il test è assolutamente continuo, cioè $T_i \neq T_j \; \forall \; i \neq j$ il sistema di equazioni (1.16) corrisponde a quello della regressione binaria classica limitato alle osservazioni per cui $D_i = 1$ e $D_j = 0$. Da questo punto di vista $D_i(1-D_j)$ può essere considerato il peso della coppia di osservazioni (i,j) che ha ovviamente peso diverso dalla coppia (j,i). Inoltre agli elementi di θ può essere assegnato lo stesso significato dei parametri stimati in una regressione binomiale classica, quindi se un elemento di θ è positivo significa che un aumento della covariata a lui associata, tenendo fissate tutte le altre, aumenta anche il valore del predittore lineare.

Nel caso in cui non tutti i dati siano stati verificati, per arginare la distorsione di verifica, sempre in Liu e Zhou (2013), è stato scelto di modificare la (1.16) introducendo ρ_i, che è la probabilità che l'i-esimo soggetto sia malato dato le sue covariate, e/o π_i, che è la probabilità che l'i-esimo soggetto sia verificato. La stima di queste probabilità passa attraverso una formulazione parametrica: nello specifico viene utilizzato un modello di regressione binomiale la cui stima è fatta attraverso le osservazioni verificate e le covariate presenti nel predittore non sono necessariamente quelle d'interesse per l'AUCC. Sono state proposte quindi quattro nuove versioni delle equazioni di stima utilizzando quattro diversi metodi partendo dal presupposto che i dati siano "missing at random" cioè che lo stato di salute non incida nel processo di verificazione del soggetto. Due metodi si basano sull'imputazione dei dati mancanti con delle loro stime: il "Full Imputation" (FI) e il "Mean Score Imputation" (MSI). L'"Inverse Probability Weightin" (IPW) utilizza per la stima di $\hat{\pi}$ tutti i dati ma inserisce nel sistema di equazioni (1.16) solo i verificati opportunamente pesati per la probabilità di esserlo. Infine il "Double Robust" (DR) che utilizza una sintesi delle due strategie imputando i dati mancanti con una loro stima e pesando opportunamente i verificati per la loro probabilità di esserlo[1].

[1]Questo metodo dal punto di vista dell'accuratezza della stima in genere è il migliore ma è inaffidabile, infatti ci potrebbero essere dei problemi nella procedura di stima, tra i quali la non convergenza dell'algoritmo di massimizzazione.

Le funzioni di stima per ognuno di questi metodi diventano:

- FI : $\sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T}\right)^T \frac{(I_{ij}-\xi_{ij})}{(1-\xi_{ij})\xi_{ij}} \hat{\rho}_i(1-\hat{\rho}_i)$
- MSI : $\sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T}\right)^T \frac{(I_{ij}-\xi_{ij})}{(1-\xi_{ij})\xi_{ij}} [V_i D_i + (1-V_i)\hat{\rho}_i][1 - V_j D_j - (1-V_j)\hat{\rho}_j]$
- IPW : $\sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T}\right)^T \frac{(I_{ij}-\xi_{ij})}{(1-\xi_{ij})\xi_{ij}} D_i(1-D_j)\frac{V_i V_j}{\hat{\pi}_i \hat{\pi}_j}$
- DR :$\sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T}\right)^T \frac{(I_{ij}-\xi_{ij})}{(1-\xi_{ij})\xi_{ij}} [\frac{V_i D_i}{\hat{\pi}_i} + (1-\frac{V_i}{\hat{\pi}_i})\hat{\rho}_i][1 - \frac{V_j D_j}{\hat{\pi}_j} - (1-\frac{V_j}{\hat{\pi}_j})\hat{\rho}_j]$

Per quanto riguarda la stima dell'AAUC, se i D_i non sono osservati per tutte le osservazioni, utilizzando i metodi già citati, la (1.14) viene modificata nel seguente modo:

$$\hat{F}^{(est)}_{X|D=1}(x) = \frac{\sum_i I(X_i < x)\hat{D}_i}{\sum_i \hat{D}_i} \tag{1.17}$$

Per i diversi metodi, le $\hat{D}_i$ vengono calcolate nei seguenti modi:

- FI : $\hat{D}_i = \hat{\rho}_i$
- MSI : $\hat{D}_i = V_i D_i + (1-V_i)\hat{\rho}_i$
- IPW : $\hat{D}_i = \frac{V_i}{\hat{\pi}_i} D_i$
- DR : $\hat{D}_i = \frac{V_i}{\hat{\pi}_i} D_i + (1-\frac{V_i}{\hat{\pi}_i})\hat{\rho}_i$

Di conseguenza la (1.15) viene così modificata:

$$\hat{AAUC} = \frac{\sum_i \hat{AUC}_x \hat{D}_i}{\sum_i \hat{D}_i} \tag{1.18}$$

Sotto le ipotesi MAR e di corretta specificazione dei modelli per le stime per $\hat{\rho}_i$ e/o $\hat{\pi}_i$ è stato dimostrato che tutte le funzioni per la stima dei parametri sono consistenti. A partire da questo risultato è stata inoltre dimostrata sotto alcune condizioni di regolarità la normalità asintotica degli stimatori per $\hat{\theta}$ e la linearità asintotica dello stimatore per l'$\hat{AAUC}$.

Il metodo appena presentato, nelle sue varie versioni in base alla tecnica di imputazione, si basa principalmente sul presupposto della correttezza

delle stime $\hat{\rho}_i$ e/o $\hat{\pi}_i$. Queste stime vengono calcolate attraverso tecniche con assunti parametrici, tipicamente tramite modelli di regressione lineare binaria, ma se questi non sono correttamente specificati le stime di θ e dell'AAUC risultano fortemente distorte, con distorsioni a volte addirittura maggiore della distorsione di verifica che questi metodi si propongono di arginare, come sottolineato anche nell'articolo che illustra queste tecniche. Per questo motivo in questo lavoro si propone un metodo non-parametrico per la stima di $\hat{\rho}_i$ che possa evitare questo problema.

Capitolo 2

Strumenti

2.1 Il metodo KNN

Il metodo KNN è stato sviluppato a partire dagli anni cinquanta da quando Fix and Hodges, nel non pubblicato "US Air Force School of Aviation Medicine report" nel 1951, ne gettarono le basi come metodo di classificazione. Fu successivamente studiato e ampliato da diversi autori, tra cui citiamo Philip E. Cheng, e utilizzato in diversi campi. In Ning e Cheng (2012) viene proposta l'applicazione di questo metodo per l'imputazione dei dati mancanti per la stima delle media e ne vengono dimostrate alcune sue proprietà.

Questo metodo è stato ripreso ed ampliato in Adimari e Chiogna (2015a) in cui ne viene estesa l'applicazione al caso fino ad ora descritto, al fine di ottenere una stima per un eventuale stato di salute per un soggetto che non è stato sottoposto al test gold standard. Il metodo consiste nell'individuare l'insieme di K osservazioni verificate più vicine a un'osservazione non verificata e utilizzare la media degli stati di salute delle osservazioni appartenenti a questo insieme come valore di "input" in sostituzione dello stato di salute mancante. Come più vicine si intende le osservazioni per cui la distanza sia minore rispetto alle altre. Tipicamente le distanze più usate sono quella euclidea, $d(x_i, x_j) = \sqrt{\sum_{h=1}^{p}(x_{ih} - x_{jh})^2}$, e quella di Mahalanobis, $d(x_i, x_j) = \sqrt{(x_i - x_j)^T \hat{\Sigma}^{-1}(x_i - x_j)}$ dove $\hat{\Sigma}$ è la matrice di varianza e

covarianza delle covariate. In termini matematici:

$$\hat{\rho}_{Ki} = \frac{1}{K} \sum_{j=1}^{K} D_{i(j)} \tag{2.1}$$

dove $\{Y_{i(j)}, D_{i(j)} : V_{i(j)} = 1, j = 1, ..., K\}$ è un insieme di K dati e $Y_{i(j)}$ indica la j-esima osservazione verificata più vicina a $Y_i = (T_i, X_i)^T$.

Sia ora:

$$\hat{\rho}_K = \frac{1}{n} \sum_{i=1}^{n} V_i D_i + (1 - V_i)\hat{\rho}_{Ki} \tag{2.2}$$

Sotto le ipotesi che $E(D^2) < \infty$, $Var(D|Y = y) = \rho(y)(1 - \rho(y))$ e che $\rho(y)$ e $\pi(y)$ siano finite e differnziabili almeno per il primo ordine lo stimatore KKN per $\hat{\rho}_K$ è consistente e asintoticamente normale.

Se si fosse in presenza di una o più covariate X di natura categoriale, seguendo quanto illustrato in Adimari e Chiogna (2015b), una strada possibile è quella di creare una sorta di KNN stratificato. Le osservazioni vengono raggruppate in sottoinsiemi che hanno in comune gli stessi livelli per quanto riguarda le covariate categoriali e per ognuna delle osservazioni non osservate vengono cercate le più vicine verificate all'interno dell'insieme di appartenenza.

Il motivo di questa scelta è che un confronto tra la distanza fra diversi punti di una variabile categoriale e di una continua non è sempre agevole, non potendo standardizzare le variabili categoriali, quindi si preferisce creare delle sotto-popolazioni per caratteristiche piuttosto che ignorare completamente l'informazione derivante da esse.

Le principali problematiche del metodo KNN sono due: la scelta di K e quella della distanza. Per quanto riguarda il numero dei vicini più vicini i precedenti studi simulativi, sia in Ning e Cheng (2012) che Adimari e Chiogna (2015a), mostrano che considerare i valori dell'intervallo 1-3 si rivela generalmente una buona scelta. In ogni caso un metodo valido per questa decisione è l'uso della convalida incrociata utilizzando soltanto i soggetti verificati: a turno ogni soggetto viene schermato e si stima il suo $\hat{\rho_{K_i}}$; terminata la rotazione si calcola un qualche indice di discrepanza, per esempio $\frac{1}{n_{ver}} \sum_{i=1}^{n_{ver}} |D_i - \hat{\rho}_{Ki}|$ e si sceglie il numero di vicini più vicini che sembra

offrire la performance migliore. Per quanto riguarda la scelta della distanza sono da tenere in considerazione due fattori principalmente: la struttura dei dati e la complessità computazionale. Si è notato che la distanza Euclidea, a differenza di quella di Mahalanobis, potrebbe non essere adatta in presenza di variabili di ordini di misura e variabilità molto diverse. D'altro canto la distanza Euclidea porta dei notevoli vantaggi per quanto riguarda la complessità computazionale. Anche in questo caso utilizzare una convalida incrociata e lo stesso indice di discrepanza potrebbe essere una buona soluzione sia per la scelta della distanza che dovrebbe portare ad un risultato più preciso sia per farsi un idea del tempo di elaborazione.

2.2 Bootstrap

L'idea di fondo del bootstrap è quella di creare un "mondo parallelo" a quello reale partendo dalle realizzazioni di quest'ultimo per poterlo studiare meglio.

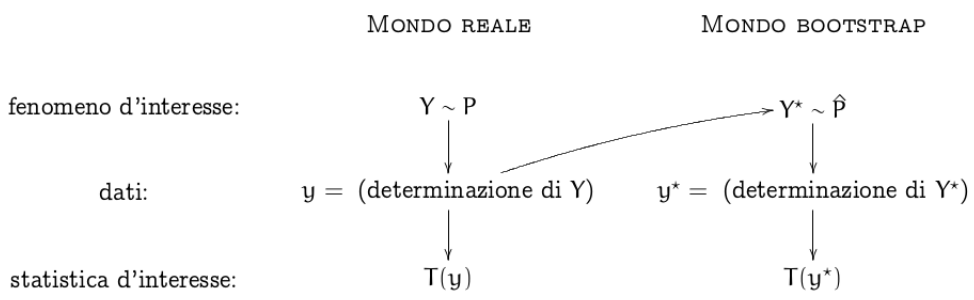


Fig. 2.1: Idea di fondo del bootstrap

Utilizzando la terminologia dello schema 2.1, il problema è che P, la distribuzione dei nostri dati, è ignota, noi siamo a conoscenza solo delle osservazioni y con le quali ricaviamo una statistica, anche multivariata, $T(y)$. Ma a partire da queste ultime affrontare problemi inferenziali, come costruire un intervallo di confidenza, per $T(y)$ potrebbe non essere possibile.
Si decide di creare un "mondo parallelo" a partire dalle nostre osservazioni

y costruendo sulla loro base la distribuzione $\hat{P}$ del fenomeno stocastico Y^* che è una stima di P.

Fatto questo se osservassimo il fenomeno di interesse nel "mondo parallelo" otteremmo delle osservazioni y^* a partire dalle quali sarebbe possibile calcolare $T(y^*)$ la cui distribuzione, essendo $\hat{P}$ conosciuta è nota.

Essendo però il "mondo parallelo" costruito sulla base del mondo reale, ci si può aspettare che $T(y^*)$ sia una buona approssimazione di $T(y)$.

Ovviamente non sempre è possibile o agibile calcolare analiticamente la distribuzione di $T(y^*)$ quindi l'idea sta nel creare B, non piccolo, realizzazioni del "mondo parallelo" a partire da $\hat{P}$, $y_1^*, ..., y_B^*$, da cui ricavare le relative statistiche, $T(y_1^*), ..., T(y_B^*)$, e utilizzare questi valori per ottenere informazioni su $T(Y^*)$.

In pratica a partire dal campione dei dati osservati y, di numerosità n, si estraggono con reinserimento n osservazioni che andranno a formare il primo campione bootstrap y_1^*, si ripete l'operazione B volte creando ogni volta un nuovo campione bootstrap ottenendo appunto $y_1^*, ..., y_B^*$. Fatto questo da ogni campione si stimano $T(y_1^*), ..., T(y_B^*)$ che verranno utilizzati per costruire una distribuzione empirica per $T(Y)$ Masarotto (2009).

Capitolo 3

Il metodo proposto

In Alonzo e Pepe (2005) sono state avanzate alcune proposte per la stima della specificità e sensibilità di un test nel caso fossero presenti dei dati mancanti e che quindi le stime potessero essere affette da distorsione di verifica. Per correggerla hanno utilizzato gli stessi metodi che sono stati successivamente utilizzati in Liu e Zhou (2013) e sono stati esposti nel paragrafo 1.4: FI, MSI, IPW e DR.
Ovviamente anche in questo caso, se i modelli per stimare i ρ_i e/o i π_i non sono correttamente specificati, le stime sono fortemente distorte. Per ovviare a questo problema in Adimari e Chiogna (2015a) viene proposta una modifica dello stimatore MSI, tramite un metodo completamente non parametrico, che consiste nella sostituzione dei ρ_i stimati con un modello di regressione con dei ρ_{Ki} stimati con il metodo KNN. Sempre gli stessi autori in Adimari e Chiogna (2015b) propongono di utilizzare il metodo KNN anche per la stima dell'AUC, sempre per risolvere il problema della distorsione di verifica senza doversi attenere a restrizioni parametriche.
Alla stessa maniera, in questa tesi si propone di modificare i metodi presentati nel paragrafo 1.4 utilizzando il metodo KKN per ottenere le stime $\hat{\rho}_{Ki}$ e utilizzare queste ultime per la stima dei θ per il calcolo dell'AUC_x e per la stima dell'AAUC. Il seguente sitmatore viene proposto sotto la condizione che i dati mancanti per lo stato di salute siano "missing at random", cioè la presenza o no di un dato dipende dal valore del test e delle covariate e non

dal vero stato di salute.

3.1 Stimatore

Sia $\hat{\rho}_{Ki}$ quello espresso dalla formula (2.1) con l'aggiunta, nel caso in cui siano presenti variabili di tipo categoriale, delle opportune stratificazioni e definiamo

$$\Delta_{ki} = V_i D_i + (1 - V_i)\hat{\rho}_{Ki} \tag{3.1}$$

Come si può notare (3.1) ha la stessa forma dello stimatore $\hat{D}_i$ nel caso si utilizzasse il metodo di imputazione MSI e allo stesso modo lo utilizziamo per sostituire all'interno delle equazioni di stima (1.16) la variabile D_i:

$$\sum_i \sum_{j \neq i} U_{ij} = \sum_i \sum_{j \neq i} \left(\frac{\partial \xi_{ij}}{\partial \theta^T} \right)^T \frac{(I_{ij} - \xi_{ij})}{(1 - \xi_{ij})\xi_{ij}} \Delta_{Ki}(1 - \Delta_{Kj}) = 0 \tag{3.2}$$

Utilizziamo Δ_{Ki} sempre per sostituire D_i anche nella formula per il calcolo dell'AAUC (1.18) che risulta essere quindi:

$$\hat{AAUC} = \frac{\sum_i \hat{AUC}_x \Delta_{Ki}}{\sum_i \Delta_{Ki}} \tag{3.3}$$

3.2 Intervalli di confidenza e test bootstrap

Per la costruzione degli intervalli di confidenza si propone di utilizzare la tecnica bootstrap, in particolare di utilizzare i quantili della distribuzione empirica simulata per ogni elemento di $\hat{\theta}$ e per l'$\hat{AAUC}$. Cioè da ognuno dei B campioni bootstrap ottenuti dalla osservazione originaria sono ottenute le stima bootstrap $\hat{\theta}^{*b}$ e $\hat{AUCC}^{*b}$ utilizzando gli stimatori proposti. Gli elementi corrispondenti ad ogni $\hat{\theta}^{*b}$ e i valori di $\hat{AUCC}^{*b}$ vengono poi ordinati in ordine crescente. Per ciascun parametro gli estremi dell'intervallo di confidenza di livello $1 - \alpha$ saranno assegnati alle osservazioni che occupano la posizione $B \cdot \frac{\alpha}{2}$ e $B \cdot (1 - \frac{\alpha}{2})$ nell'ordinamento.

Nonostante lo studio teorico di un eventuale risultato di normalità asintotica

degli stimatori del parametro θ non è tra gli obiettivi di questa tesi, i successivi studi di simulazione sembrano suggerire tale risultato e di conseguenza viene proposta anche una seconda versione degli intervalli di confidenza basati su di esso. Anche in questo caso è necessario ricorrere alla simulazione bootstrap poiché le espressioni della varianza degli stimatori ottenuti con tecniche basate sul KNN non sempre sono agevoli. Sia $\hat{\theta}^*$ la media delle stime bootstrap $\hat{\theta}^{*b}$; con queste quantità viene calcolata la stima della varianza boostrap per lo stimatore $\hat{\theta}$:

$$\hat{\sigma}^2 = \frac{\sum_{b=1}^{B}(\hat{\theta}^{*b} - \hat{\theta}^*)^2}{B-1} \tag{3.4}$$

Stimato questa quantità l'intervallo di confidenza di livello $1-\alpha$ basato sulla presunta distribuzione normale risulta essere:

$$IC = \hat{\theta} \pm z_{1-\frac{\alpha}{2}}\sqrt{\hat{\sigma}^2} \tag{3.5}$$

Per quanto riguarda i test per le ipotesi di nullità sugli elementi di θ (che ricordiamo sono i parametri del modello di regressione utilizzato per la stima di $\hat{AUC}_x$), $H_0 : \theta_k = 0$ vs $H_1 : \theta_k \neq 0$ con e per $H_0 : AAUC = 0.5$ vs $H_1 : AAUC \neq 0.5$ si propone un test di tipo bootstrap seguendo il metodo delineato in Hall e Wilson (1991).
Si abbia una campione $Y = (y_1, ..., y_n)$ da cui si ricava una statistica $\hat{\nu} = T(Y)$, stimatore per un parametro ν di cui si vuole verificare l'ipotesi di uguaglianza a un valore fissato ν_0. Un test di questo tipo, $H_0 : \nu = \nu_0$ vs $H_1 : \nu \neq \nu_0$ è basato generalmente sulla quantità $\hat{\nu} - \nu_0$ la cui distribuzione sotto H_0, come nel nostro caso, potrebbe non essere nota. Per applicare il test è necessario quindi stimare questa distribuzione. La procedura indicata nell'articolo sopraccitato implica questi passi:

- Si calcoli $T(Y) = \hat{\nu}$;
- Si facciano B ricampionamenti bootstrap da Y, $Y_1^*, ..., Y_B^*$ e si calcoli i rispettivi B $\hat{\nu}^*$;
- Si ricavi da essi $\hat{F}^*$, la funzione di ripartizione empirica per $|\hat{\nu}^* - \hat{\nu}|$;

- Il valore di $1 - \hat{F}^*(|\hat{\nu} - \nu_0|)$ è una stima il livello di significatività osservato.

3.3 Studi di simulazione

Per valutare il comportamento degli stimatori proposti sono stati considerati cinque differenti scenari di simulazione, gli stessi considerati in Liu e Zhou (2013) di cui vengono riportati nel seguito i risultati. Per ogni scenario è stata fatta una simulazione Monte Carlo di M=1000 replicazioni. Dal modello simulativo di partenza sono stati generati M campioni di numerosità pari al valore fissato nell'articolo sopraccitato (500 in un caso 1000 in tutti gli altri) e per ognuno di essi sono state ottenute le stime di interesse. Ciò ha permesso anche di confrontare i due studi.

Per ogni scenario (a parte per quello di numerosità ridotta) è stata condotta una seconda simulazione che coinvolge anche la tecnica bootstrap per le ulteriori analisi. Per queste seconde simulazioni la numerosità è stata ridotta rispetto alla prima, scendendo a 300 realizzazioni, per motivi computazionali. In questo caso si è agito come nel caso precedente solo che da ogni campione Monte Carlo sono state generati B=250 campioni bootstrap da cui sono stati ricavati i gli intervalli di confidenza basati sulle procedure illustrate in precedenza.

Per le simulazioni che coinvolgono soltanto la tecnica Monte Carlo sono date le stime KNN sia per quanto riguarda la distanza euclidea sia per quella di Mahalanobis e per entrambe sia per per lo stimatore basato sul metodo che utilizza soltanto un vicino più vicino (1NN) che per lo stimatore basato sul metodo che utilizza tre vicini più vicini (3NN). Inoltre poiché l'impianto simulativo permette di conoscere il vero valore dello stato di salute anche per i soggetti non verificati, sono state riportate le stime in assenza di distorsione di verifica (full).

Per le simulazione che includono la tecnica bootstrap invece è stata considerata soltanto la distanza euclidea. In ogni scenario vengono considerate due

covariate, una continua X_1 e una discreta X_2, per le quali vengono stimati i parametri θ_1 e θ_2 rispettivamente, che con $\hat{\theta}_0$, il valore che corrisponde alla stima dell'intercetta, possono essere utilizzate tramite la (1.6) per la stima dell'AUC_x. Viene poi stimata anche l'$AAUC$.

Per ogni parametro vengono sempre riportati tre valori: il valore stimato, ottenuto come media delle stime del parametro per ogni replicazione, la stima della distorsione dello stimatore corrispondente (in termini percentuali rispetto al valore vero del parametro, $dist = \frac{(\hat{\theta}-\theta_0)\cdot 100}{\theta_0}$) e la deviazione standard delle stime delle replicazioni Monte Carlo. Inoltre, per le simulazioni che coinvolgono il bootstrap vengono riportati altri tre valori per ogni parametro che riguardano gli intervalli di confidenza: "cov boot" che indica la percentuale di replicazioni Monte Carlo in cui il vero valore del parametro è compreso nell'intervallo bootstrap basato sui quantili; "cov norm" che indica lo stesso valore ma per un intervallo costruito utilizzando la ipotetica distribuzione normale e "confr cover" in cui vengono confrontati i due precedenti valori contando la percentuale di replicazioni in cui uno dei due intervalli ha all'interno il vero valore e l'altro no.

Nelle tabelle che sono state riprese dall'articolo di Liu e Zhou (2013) (Tabelle 3.7, 3.8, 3.11, 3.12, 3.19, 3.20, 3.27, 3.28, 3.35, 3.36) sono presenti anche il valore stimato, la distorsione (in percentuale), la deviazione standard delle replicazione Monte Carlo e la copertura dell'intervallo di confidenza basato sull'approssimazione normale per gli stimatori proposti dagli autori. Per ogni stimatore considerato dagli autori vengono presentati i risultati sia nel caso in cui i modelli per $\hat{\pi}_i$ e/o $\hat{\rho}_i$ siano correttamente specificati, cioè i modelli usati nella stima rispecchiano la vera struttura dei dati, sia nel caso contrario. In ogni tabella vengono riportati i risultati per dodici stimatori:

1. $Full$, che effettua le stime con i veri valori dello stato di salute e utilizza quindi tutti i dati;

2. CC, che effettua le stime utilizzando solo i dati verificati;

3. IPW_1 l'IPW con il modello per i $\hat{\pi}_i$ corretto;

4. IPW_2 l'IPW con il modello per i $\hat{\pi}_i$ non corretto;
5. FI_1 l'FI con il modello per i $\hat{\rho}_i$ corretto;
6. FI_2 l'FI con il modello per i $\hat{\rho}_i$ non corretto;
7. MSI_1 l'MSI con il modello per i $\hat{\rho}_i$ corretto;
8. MSI_2 l'MSI con il modello per i $\hat{\rho}_i$ non corretto;
9. DR_1 l'DR con il modello per i $\hat{\rho}_i$ corretto e quello per i $\hat{\pi}_i$ corretto;
10. DR_2 l'DR con il modello per i $\hat{\rho}_i$ non corretto e quello per i $\hat{\pi}_i$ corretto;
11. DR_3 l'DR con il modello per i $\hat{\rho}_i$ corretto e quello per i $\hat{\pi}_i$ non corretto;
12. DR_4 l'DR con il modello per i $\hat{\rho}_i$ non corretto e quello per i $\hat{\pi}_i$ non corretto.

Tutti gli scenari presentati nel seguito, come già detto, sono già considerati nella articolo di Liu e Zhou (2013)

3.3.1 Scenario di base

Nello scenario di base si ha:

$n = 1000$ [La numerosità]

$X_1 \sim U(-1, 1)$ [La variabile continua (covariata continua)]

$X_2 \sim Bi(1, 0.5)$ [La variabile discreta (covariata discreta)]

$logit(p) = -1.4 + 0.5X_2 + 0.8X_1$

$D \sim Bi(1, p)$ [Lo stato di salute]

$\mu = 1 + 0.4D + 0.2X_2 + 0.7X_1 + X_2D + 0.5X_1D$

$\sigma = 0.8D + 1.2(1 - D)$

$T \sim N(\mu, \sigma^2)$ [Il valore del test]

$logit(\pi) = -1.2 + T + 0.6X_2 + 1.2X_1$

$V \sim Bi(1, \pi)$ [Lo stato di verifica]

probit [La funzione legame]

In questo scenario la percentuale di malati, $Pr(D = 1)$, si attesta intorno

al 25% mentre quella dei soggetti verificati, $Pr(V = 1)$, si aggira intorno al 57%. Per quanto riguarda i veri valori degli elementi del parametro θ è necessario fare alcuni passaggi.
Iniziamo dal definire $AUC_{0,0}$, $AUC_{0,1}$ e $AUC_{1,0}$ come:

$$AUC_{0,0} = Pr(T_j > T_i | D_j = 1, D_i = 0, X_{1j} = X_{1i} = 0, X_{2j} = X_{2i} = 0)$$
$$AUC_{0,1} = Pr(T_j > T_i | D_j = 1, D_i = 0, X_{1j} = X_{1i} = 0, X_{2j} = X_{2i} = 1)$$
$$AUC_{1,0} = Pr(T_j > T_i | D_j = 1, D_i = 0, X_{1j} = X_{1i} = 1, X_{2j} = X_{2i} = 0)$$

(si noti, per chiarezza di notazione, che, per esempio, $AUC_{0,0}$ indica AUC_x con $x = (x_1, x_2) = (0, 0)$). Le distribuzioni condizionate di T_i e T_j sono quindi date da

$$T_j | (D_j = 1, X_{1j} = 0, X_{2j} = 0) \sim N(1.4, 0.64)$$
$$T_i | (D_i = 0, X_{1i} = 0, X_{2i} = 0) \sim N(1, 1.44)$$
$$T_j | (D_j = 1, X_{1j} = 0, X_{2j} = 1) \sim N(2.6, 0.64)$$
$$T_i | (D_i = 0, X_{1i} = 0, X_{2i} = 1) \sim N(1.2, 1.44)$$
$$T_j | (D_j = 1, X_{1j} = 1, X_{2j} = 0) \sim N(2.6, 0.64)$$
$$T_i | (D_i = 0, X_{1i} = 1, X_{2i} = 0) \sim N(1.7, 1.44),$$

da cui si ricava

$$AUC_{0,0} = Pr(T > 0) = 0.6092 \; dove \; T \sim N(0.4, 2.08)$$
$$AUC_{0,1} = Pr(T > 0) = 0.8341 \; dove \; T \sim N(1.4, 2.08)$$
$$AUC_{1,0} = Pr(T > 0) = 0.7337 \; dove \; T \sim N(0.9, 2.08).$$

Risolvendo il sistema

$$\begin{cases} \Phi^{-1}(0.6092) = \theta_0 \\ \Phi^{-1}(0.8341) = \theta_0 + \theta_2 \\ \Phi^{-1}(0.7337) = \theta_0 + \theta_1 \end{cases}$$

si ha che $\theta_0 = 0.277$, $\theta_1 = 0.693$, $\theta_2 = 0.347$.
L'AAUC è invece stata calcolata computazionalmente. Sono state generate un numero grande (un milione) di coppie x_1 e x_2; per ognuna di queste coppie è stato calcolato il valore di $E[D|x_1, x_2]$ che vale $logit^{-1}(-1.4 + 0.5x_2 + 0.8x_1)$ e dell'AUC_{x_1,x_2}; si sommano tutti questi risultati e si dividono per la somma dei $E[D|x_1, x_2]$. Risulta che $AAUC = 0.7582$ che è lo stesso valore trovato in Liu e Zhou (2013).

Monte Carlo

Le tabelle 3.1 e 3.2 riportano i risultati degli esperimenti Monte Carlo relativi allo scenario di base e ai nuovi stimatori KNN. Tali risultati vanno confrontati con quelli riportati nelle tabelle 3.7 e 3.8 riprese dal lavoro di Liu e Zhou (2013).
In particolare nelle tabelle 3.1 e 3.2 si può notare che le distorsioni relative ai nuovi stimatori degli elementi di θ, a parte casi isolati (IPW_2 per θ_2 e il DR_3, lo stimatore con il modello per la stima di ρ_i correttamente specificato ma con quello per la stima di π_i non specificato correttamente, per tutti gli elementi di θ), sono minori di quelli degli stimatori "classici" considerati in Liu e Zhou (2013), nel caso in cui i modelli parametrici fissati nel processo di stima non siano specificati correttamente.
Per quanto riguarda l'$AAUC$ i valori delle distorsioni sono troppo piccoli per essere confrontati in maniera affidabile. Globalmente lo stimatore che sembra comportarsi peggio è quello 3NN basato sulla distanza di Mahalanobis. Per quanto riguarda la deviazione standard stimata, in questo così come in tutti gli altri scenari, si può notare come sia in genere più alta quella per il metodo KNN ma non esageratamente.

	$\theta_0 = 0.277$			$\theta_1 = 0.347$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.27827	0.31475	0.07258	0.35049	1.09399	0.08932
euclidea1NN	0.28702	3.46635	0.15574	0.32965	-4.91880	0.16662
euclidea3NN	0.28075	1.20870	0.14391	0.33480	-3.43263	0.14944
Mahalanobis1NN	0.27674	-0.23798	0.15435	0.34717	0.13606	0.17026
Mahalnobis3NN	0.26188	-5.59342	0.14031	0.36393	4.97076	0.15230

Tab. 3.1: Monte Carlo, n=1000, Replicazioni =1000, base

	$\theta_2 = 0.693$			$AAUC = 0.7582$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.69845	0.72761	0.09952	0.75888	0.09027	0.01676
euclidea1NN	0.69590	0.36023	0.18291	0.75681	-0.18280	0.03342
euclidea3NN	0.69684	0.49664	0.16965	0.75531	-0.38139	0.03051
Mahalanobis1NN	0.70171	1.19775	0.18235	0.75464	-0.46978	0.03304
Mahalnobis3NN	0.70831	2.15079	0.16657	0.75183	-0.83969	0.03025

Tab. 3.2: Monte Carlo, n=1000, Replicazioni =1000, base

Bootstrap

Le tabelle 3.3, 3.4, 3.5 e 3.6 riportano i risultati (ottenuti sempre nello scenario di base) relativi all'utilizzo della tecnica bootstrap associata alle simulazioni Monte Carlo e in particolare il livello di copertura per gli intervalli di confidenza di livello nominale del 95%. Come già detto, per motivi meramente computazionali, la numerosità è stata ridotta a $n = 300$. Anche in questo caso vengono effettuate 1000 replicazioni Monte Carlo mentre le replicazioni bootstrap sono state fissate a 200. La copertura degli intervalli di confidenza basati sugli stimatori KNN è generalmente buona con valori degli intervalli di confidenza mai al di sotto dell' 89.8%. Generalmente gli intervalli basati sullo stimatore 3NN sembrano leggermente di quelli basati sull'1NN. Se non si considera lo stimatore per l'$AUCC$, si può fare la stessa

osservazione per gli intervalli che si basano sulla presunta normalità asintotica degli stimatori rispetto a quelli calcolati partendo dai quantili della distribuzione bootstrap. La massima differenza in copertura fra i due tipi di intervalli è del 2,4%.

Se confrontati con gli intervalli ottenuti utilizzando i metodi "classici" riportati nelle colonne "cov" delle tabelle 3.7 e 3.8, gli intervalli di confidenza basati sugli stimatori KNN hanno performance, in termini di copertura, simili a quando il modello parametrico inserito negli stimatori "classici" è correttamente specificato e, a parte per lo stimatore che utilizza la tecnica DR, ben superiore a quando il suddetto modello parametrico non è correttamente specificato. Questo confronto è comunque puramente indicativo poichè le numerosità dei campioni nei due casi (metodi KNN, metodi "classici") non sono le stesse.

$\theta_0 = 0.277$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.28113	1.34331	0.13593	0.94800	0.95400	0.01600
euclidea1NN	0.31341	12.98016	0.27552	0.89800	0.90200	0.01800
euclidea3NN	0.29191	5.22939	0.25808	0.90100	0.90600	0.01500

Tab. 3.3: Boot, n=300, Repl monte =1000, Repl boot =250, base

$\theta_1 = 0.347$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.35407	2.12636	0.17317	0.93800	0.94700	0.01100
euclidea1NN	0.30673	-11.52804	0.28749	0.93800	0.94200	0.01400
euclidea3NN	0.31603	-8.84684	0.25071	0.94600	0.94500	0.01100

Tab. 3.4: Boot, n=300, Repl monte =1000, Repl boot =250, base

$\theta_2 = 0.693$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.70489	1.65772	0.18967	0.94400	0.94900	0.01300
euclidea1NN	0.68564	-1.11869	0.33942	0.92800	0.93800	0.02400
euclidea3NN	0.69553	0.30743	0.31910	0.93000	0.93600	0.01200

Tab. 3.5: Boot, n=300, Repl monte =1000, Repl boot =250, base

$AAUC = 0.7582$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.75762	-0.07663	0.03110	0.93800	0.94100	0.01700
euclidea1NN	0.75255	-0.74505	0.05992	0.92200	0.91400	0.02000
euclidea3NN	0.74831	-1.30418	0.05444	0.92700	0.92500	0.02400

Tab. 3.6: Boot, n=300, Repl monte =1000, Repl boot =250, base

	θ_0				θ_1			
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.28	1.1	0.706	95.8	0.344	-1	0.0911	94.6
CC	0.039	-86.1	0.1097	39.7	0.499	43.9	0.1248	77.4
FI_1	0.279	0.6	0.1124	94.8	0.344	-0.8	0.1206	95
FI_2	0.415	49.7	0.1081	71.7	0.156	-55	0.1147	57.5
MSI_1	0.279	0.6	0.1131	95	0.343	-1.2	0.121	95.3
MSI_2	0.351	26.8	0.1073	87	0.254	-26.7	0.1117	84.8
IPW_1	0.289	4.3	0.1412	92.2	0.332	-4.4	0.1701	92.2
IPW_2	0.33	19.2	0.1394	90.1	0.452	30.3	0.1603	87.8
DR_1	0.294	6.2	0.1323	90.6	0.32	-7.8	0.1566	89.6
DR_2	0.297	7.3	0.1368	90.7	0.319	-8.1	0.1616	90.2
DR_3	0.289	4.2	0.123	92.5	0.335	-3.4	0.138	91.8
DR_4	0.322	16.4	0.1195	88.4	0.298	-14.2	0.1335	89.9

Tab. 3.7: tabella per lo scenario di base riportata dall'articolo scritto da Liu e Zhou (2013)

			θ_2			$AAUC$		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.693	0	0.1009	95.2	0.7582	0	0.0163	95.3
CC	0.764	10.3	0.1353	91.4	0.7423	-2.1	0.0212	91.9
FI_1	0.701	1.1	0.1269	94.1	0.7574	-0.1	0.026	94.4
FI_2	0.519	-25.1	0.1207	68.9	0.7574	-0.1	0.0255	94.3
MSI_1	0.699	0.8	0.1287	94	0.7574	-0.1	0.026	95.1
MSI_2	0.629	-9.3	0.1221	90.1	0.7567	-0.2	0.026	94.7
IPW_1	0.688	-0.7	0.1763	92.3	0.7574	-0.1	0.0284	93.2
IPW_2	0.696	0.4	0.1689	93.9	0.7574	-0.1	0.0255	73.8
DR_1	0.683	-1.5	0.1603	93.4	0.784	3.4	0.0282	91.7
DR_2	0.681	-1.7	0.165	93.5	0.7582	0	0.0286	91.6
DR_3	0.689	-0.6	0.1483	93.8	0.7582	0	0.0267	93.6
DR_4	0.676	-2.5	0.1462	93.7	0.7582	0	0.0268	93.1

Tab. 3.8: tabella per lo scenario di base riportata dall'articolo scritto da Liu e Zhou (2013)

3.3.2 Numerosità ridotta

In questo scenario la numerosità viene dimezzata rispetto allo scenario precedente e fissata quindi a 500 unità. In questo caso non viene effettuata la simulazione bootstrap. Nello scenario con numerosità ridotta si ha:

$n = \underline{500}$ [La numerosità]

$X_1 \sim U(-1, 1)$ [La variabile continua (covariata continua)]

$X_2 \sim Bi(1, 0.5)$ [La variabile discreta (covariata discreta)]

$logit(p) = -1.4 + 0.5X_2 + 0.8X_1$

$D \sim Bi(1, p)$ [Lo stato di salute]

$\mu = 1 + 0.4D + 0.2X_2 + 0.7X_1 + X_2D + 0.5X_1D$

$\sigma = 0.8D + 1.2(1 - D)$

$T \sim N(\mu, \sigma^2)$ [Il valore del test]

$logit(\pi) = -1.2 + T + 0.6X_2 + 1.2X_1$

$V \sim Bi(1, \pi)$ [Lo stato di verifica]

probit [La funzione legame]

Naturalmente, diminuendo in questo caso soltanto la numerosità, i veri valori dei parametri, $\theta_0 = 0.277$, $\theta_1 = 0.693$, $\theta_2 = 0.347$, $AAUC = 0.7582$, la percentuale di verificati, 57%, e quella di malati, 25%, restano invariati rispetto allo scenario di base.

Monte Carlo

Le tabelle 3.9 e 3.10 riportano i risultati degli esperimenti Monte Carlo relativi allo scenario di base ma con la numerosità ridotta e ai nuovi stimatori KNN. Tali risultati vanno confrontati con quelli riportati nelle tabelle 3.11 e 3.12 riprese dal lavoro di Liu e Zhou (2013).

In particolare nel confronto delle distorsioni degli stimatori "classici" con numerosità ridotta (tabelle 3.11 e 3.12) con le distorsioni degli stimatori "classici" con numerosità pari a mille (tabelle 3.7 e 3.12) è possibile notare come con il diminuire della numerosità campionaria aumenti anche la distorsione delle stime, soprattutto per gli stimatori basati su un model-

lo parametrico correttamente specificato. Questo aumento della distorsione degli stimatori al diminuire della numerosità campionaria è stato notato anche per gli stimatori basati sul KNN anche se particolare è il miglioramento della stima basata su 1NN con la distanza di Mahalanobis (confrontare le tabelle riferite alla simulazione Monte Carlo dello scenario precedente, 3.1 e 3.2, con quelle di questo scenario 3.9 e 3.10. Si può notare inoltre che le distorsioni presenti nelle tabelle 3.9, 3.10 riguardanti gli stimatori KNN per gli elementi di θ sono sempre minori, a parte per lo stimatore basato sulla tecnica DR_3[1], degli stimatori "classici" cosiderati in Liu e Zhou (2013) nel caso i modelli parametrici fissati nel processo di stima non siano correttamente specificati. Anche in questo scenario, per quanto riguarda l'$AAUC$ i valori delle distorsioni sono troppo piccoli per essere confrontati in maniera affidabile.

	$\theta_0 = 0.277$			$\theta_1 = 0.347$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.28078	1.21732	0.10470	0.34548	-0.35103	0.13144
euclidea1NN	0.30204	8.88244	0.20584	0.31753	-8.41302	0.22880
euclidea3NN	0.28984	4.48498	0.19285	0.32361	-6.65970	0.20298
Mahalanobis1NN	0.28141	1.44378	0.20407	0.34909	0.68902	0.23290
Mahalnobis3NN	0.25794	-7.01591	0.18353	0.37275	7.51421	0.20477

Tab. 3.9: Monte Carlo, n=500, Replicazioni =1000, base

[1] lo stimatore basato sul modello per i ρ_i correttamente specificato ma non quello per i π_i

	$\theta_2 = 0.693$			$AAUC$=0.7582		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.69614	0.39447	0.14863	0.75807	-0.01649	0.02394
euclidea1NN	0.68196	-1.65029	0.24922	0.75570	-0.32945	0.04508
euclidea3NN	0.68725	-0.88752	0.23884	0.75324	-0.65437	0.04095
Mahalanobis1NN	0.69432	0.13278	0.25064	0.75191	-0.82974	0.04447
Mahalnobis3NN	0.70443	1.59077	0.23306	0.74748	-1.41373	0.04013

Tab. 3.10: Monte Carlo, n=500, Replicazioni =1000, base

	θ_0				θ_1			
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.281	1.3	0.1047	94.5	0.356	2.6	0.1282	95.7
CC	0.281	-83.1	0.155	67.5	0.51	46.9	0.1819	85.3
FI_1	0.047	2.8	0.1584	92.8	0.359	3.5	0.1753	94.8
FI_2	0.285	53.1	0.1527	80.8	0.164	-52.6	0.1662	74.6
MSI_1	0.424	2.8	0.1587	92.7	0.356	2.7	0.1767	95.2
MSI_2	0.285	29.8	0.1496	89.2	0.264	-24	0.1628	90.1
IPW_1	0.36	6.9	0.1963	89.2	0.341	-1.8	0.2405	89.5
IPW_2	0.296	22.1	0.1926	88.7	0.46	32.6	0.2244	89.2
DR_1	0.338	11.3	0.2186	88.5	0.323	-7	0.2646	90.8
DR_2	0.308	10.7	0.1923	87.8	0.321	-7.6	0.2309	89.2
DR_3	0.307	7.3	0.1738	89.6	0.342	-1.4	0.1987	91
DR_4	0.297	20	0.1668	87.2	0.304	-12.5	0.1918	89.4

Tab. 3.11: tabella riguardante lo scenario di base con la numerosità ridotta riportata dall'articolo scritto da Liu e Zhou (2013)

			θ_2			$AAUC$		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.696	0.4	0.1526	94.5	0.7582	0	0.0231	95.1
CC	0.755	8.9	0.1986	92.7	0.743	-2	0.0302	92.7
FI_1	0.698	0.7	0.1884	93.1	0.7567	-0.2	0.0377	92.9
FI_2	0.515	-25.7	0.1805	80.2	0.7574	-0.1	0.0367	93.2
MSI_1	0.695	0.3	0.1906	93.4	0.7567	-0.2	0.0376	93.3
MSI_2	0.623	-10.1	0.1806	90.5	0.7574	-0.1	0.0372	93.2
IPW_1	0.687	-0.9	0.246	91.6	0.7567	-0.2	0.0427	90.8
IPW_2	0.692	-0.1	0.2411	91.2	0.7832	3.3	0.038	80.3
DR_1	0.678	-2.1	0.2508	91.7	0.7582	0	0.0419	91.3
DR_2	0.681	-1.7	0.2393	90.6	0.7574	-0.1	0.0412	90.8
DR_3	0.689	-0.6	0.2167	91	0.7574	-0.1	0.0395	91.6
DR_4	0.674	-2.8	0.212	91.2	0.7574	-0.1	0.0394	90.8

Tab. 3.12: tabella riguardante lo scenario di base con la numerosità ridotta riportata dall'articolo scritto da Liu e Zhou (2013)

3.3.3 Alti valori dell'AUC

In questo scenario, rispetto a quello base, è stata aumentata la capacità discriminatoria del test; questo miglioramento è stato ottenuto incrementando la differenza in media tra il valore del test per un soggetto sano rispetto a uno malato, senza toccarne la varianza. Ovviamente con questa operazione si ottengono valori più alti per quanto riguarda l'AUC_x e l'$AAUC$. Nello scenario con i valori dell'AUC maggiori si ha:

$n = 1000$ [La numerosità]

$X_1 \sim U(-1, 1)$ [La variabile continua (covariata continua)]

$X_2 \sim Bi(1, 0.5)$ [La variabile discreta (covariata discreta)]

$logit(p) = -1.4 + 0.5X_2 + 0.8X_1$

$D \sim Bi(1, p)$ [Lo stato di salute]

$\mu = 1 + \underline{0.9D} + 0.2X_2 + 0.7X_1 + X_2D + 0.5X_1D$

$\sigma = 0.8D + 1.2(1 - D)$

$T \sim N(\mu, \sigma^2)$ [Il valore del test]

$logit(\pi) = -1.2 + T + 0.6X_2 + 1.2X_1$

$V \sim Bi(1, \pi)$ [Lo stato di verifica]

probit [La funzione legame]

La percentuale di soggetti malati anche in questo caso è del 25% mentre quella dei soggetti verificati si attesta attorno al 60%.
I veri valori degli elementi di θ e dell'$AAUC$ calcolati alla stessa maniera dello scenario di base sono: $\theta_0 = 0.692$, $\theta_1 = 0.347$, $\theta_2 = 0.693$ e $AAUC = 0.8466$.

Monte Carlo

Le tabelle 3.13 e 3.14 riportano i risultati degli esperimenti Monte Carlo relativi allo scenario in cui il valore dell'AUC è maggiore rispetto allo scenario di base, cioè la capacità del test migliora, e ai nuovi stimatori KNN. Tali risultati vanno confrontati con quelli riportati nelle tabelle 3.19 e 3.20 riprese dal lavoro di Liu e Zhou (2013).

Dalle tabelle 3.13 e 3.14 si può notare che anche in questo caso i valori delle distorsioni si avvicinano di più ai valori degli stimatori "classici" basati sui modelli correttamente specificati che agli altri, con l'eccezione, anche in questo scenario, dello stimatore IPW_2 per θ_2 e dello stimatore DR_3 per tutti gli elementi di θ. In questo caso c'è un leggero miglioramento rispetto allo scenario di base dello stimatore 3NN basato sulla distanza di Mahalanobis, cosa che è specifica, come verrà mostrato in seguito, di questo scenario. Potrebbe balzare all'occhio la distorsione di $\hat{\theta}_1$ ma questa è presente anche per gli altri stimatori "classici", quindi si può attribuire alla struttura dello scenario. Anche in questo caso le stime per l'AAUC, sia quelle ottenute con gli stimatori "classici" che quelle ottenute con gli stimatori basati sul KNN, non sono sensibilmente distorte.

	$\theta_0 = 0.692$			$\theta_1 = 0.347$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.62535	0.21617	0.07665	0.34906	0.59426	0.09445
euclidea1NN	0.63497	1.75804	0.14438	0.32500	-6.33981	0.17125
euclidea3NN	0.62858	0.73432	0.13513	0.32733	-5.66822	0.16209
Mahalanobis1NN	0.62463	0.10125	0.14407	0.34078	-1.79134	0.17506
Mahalnobis3NN	0.61225	-1.88233	0.13361	0.35472	2.22558	0.16349

Tab. 3.13: Monte Carlo, n=1000, Replicazioni =1000, valori alti per l'auc

	$\theta_2 = 0.693$			$AAUC = 0.8466$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.69844	0.78511	0.10706	0.84659	-0.00076	0.01384
euclidea1NN	0.69661	0.52132	0.17797	0.84514	-0.17247	0.02547
euclidea3NN	0.70113	1.17381	0.16504	0.84423	-0.27944	0.02371
Mahalanobis1NN	0.70384	1.56482	0.17929	0.84329	-0.39150	0.02563
Mahalnobis3NN	0.71194	2.73357	0.16561	0.84158	-0.59295	0.02377

Tab. 3.14: Monte Carlo, n=1000, Replicazioni =1000, valori alti per l'auc

Bootstrap

Le tabelle 3.15, 3.16, 3.17, 3.18 riportano i risultati (ottenuti sempre per lo scenario in cui è stata aumentata la capacità discriminatoria del test) relativi all'utilizzo della tecnica bootstrap associata alle simulazioni Monte Carlo e in particolare il livello di copertura per gli intervalli di confidenza di livello nominale del 95%. Come già detto, per motivi meramente computazionali, la numerosità è stata ridotta a $n = 300$. Anche in questo caso vengono effettuate 1000 replicazioni Monte Carlo mentre le replicazione bootstrap sono state fissate a 200.
La copertura degli intervalli di confidenza basata sugli stimatori KNN ha un livello di copertura attestabile intorno al 92% e ha come limite inferiore 88.8%. A parte per lo stimatore per l'$AUCC$ gli intervalli basati sulla presunta normalità asintotica degli stimatori hanno coperture maggiori rispetto a quelli basati sui quantili bootstrap. La massima differenza in termini di copertura fra i due tipi di intervalli è del 2,2%.
Se confrontati con gli intervalli ottenuti utilizzando i metodi "classici" riportati nelle colonne "cov" delle tabelle 3.19 e 3.20, gli intervalli di confidenza basati sugli stimatori KNN hanno performance ben superiori rispetto a quelli classici ottenuti quando il modello parametrico per la stima dei ρ_i non è correttamente specificato, FI_2, MSI_2, e allo stesso livello di tutti gli altri. Come già fatto notare, questo confronto per le differenti numerosità campionarie è soltanto indicativo.

$\theta_0 = 0624$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.63018	0.99071	0.14632	0.93400	0.94400	0.01400
euclidea1NN	0.66075	5.88944	0.25861	0.88800	0.90000	0.02200
euclidea3NN	0.63816	2.26937	0.24780	0.89700	0.90200	0.01500

Tab. 3.15: Boot, n=300, Repl monte =1000, Repl boot =250, valori alti per l'auc

$\theta_1 = 0.347$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.34756	0.16202	0.19046	0.93700	0.94900	0.01400
euclidea1NN	0.29958	-13.66533	0.29988	0.92200	0.92500	0.01700
euclidea3NN	0.30983	-10.71315	0.27529	0.91900	0.92200	0.01500

Tab. 3.16: Boot, n=300, Repl monte =1000, Repl boot =250, valori alti per l'auc

$\theta_2 = 0.693$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.70345	1.50858	0.20894	0.93500	0.94500	0.01400
euclidea1NN	0.69204	-0.13843	0.32638	0.93300	0.94300	0.01400
euclidea3NN	0.70533	1.77978	0.31122	0.93700	0.94100	0.01400

Tab. 3.17: Boot, n=300, Repl monte =1000, Repl boot =250, valori alti per l'auc

$AAUC = 0.8466$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.84564	-0.11301	0.02509	0.93600	0.93600	0.01200
euclidea1NN	0.84209	-0.53330	0.04634	0.92200	0.91500	0.01300
euclidea3NN	0.83901	-0.89683	0.04171	0.92200	0.91900	0.01700

Tab. 3.18: Boot, n=300, Repl monte =1000, Repl boot =250, valori alti per l'auc

			θ_0			θ_1		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.628	0.7	0.0769	92.5	0.35	0.8	0.0959	95.5
CC	0.389	-37.6	0.1077	39.5	0.507	46	0.1279	76.1
FI_1	0.625	0.2	0.1091	94.1	0.358	3.3	0.1242	95
FI_2	0.743	19.1	0.1058	77.8	0.055	-84.1	0.117	68.6
MSI_1	0.625	0.2	0.1102	94.4	0.355	2.4	0.1253	94.6
MSI_2	0.68	9	0.1049	90	0.282	-18.6	0.1162	89.9
IPW_1	0.634	1.6	0.1323	91.8	0.344	-0.9	0.171	91.5
IPW_2	0.676	8.3	0.1276	89.8	0.46	32.7	0.1556	86.7
DR_1	0.638	2.3	0.1244	90.1	0.336	-3.2	0.1578	91.1
DR_2	0.638	2.3	0.1266	91	0.336	-3.1	0.1603	91.8
DR_3	0.633	1.4	0.1154	92.5	0.349	0.5	0.1395	92.8
DR_4	0.661	6	0.1118	90.4	0.317	-8.6	0.1346	91.1

Tab. 3.19: tabella dello scenario caratterizzato da valori alti per l'auc riportata dall'articolo scritto da Liu e Zhou (2013)

			θ_2			$AAUC$		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.694	0.1	0.1091	94.8	0.8466	0	0.0132	95
CC	0.762	9.9	0.1385	92.5	0.8271	-2.3	0.0175	82.2
FI_1	0.702	1.3	0.1317	94.3	0.8458	-0.1	0.0199	92.9
FI_2	0.525	-24.3	0.1269	70.5	0.8458	-0.1	0.0194	92.6
MSI_1	0.699	0.9	0.1334	94.3	0.8449	-0.2	0.02	93.3
MSI_2	0.641	-7.5	0.1279	91.4	0.8458	-0.1	0.0198	92.8
IPW_1	0.692	-0.2	0.1754	91.7	0.8449	-0.2	0.0224	92.3
IPW_2	0.689	-0.6	0.1634	93.4	0.8644	2.1	0.019	75.9
DR_1	0.689	-0.6	0.1603	92.7	0.8458	-0.1	0.0218	91.8
DR_2	0.689	-0.6	0.1632	92.8	0.8458	-0.1	0.0219	92
DR_3	0.693	0	0.1458	93.5	0.8458	-0.1	0.0204	91.6
DR_4	0.677	-2.3	0.1431	93.7	0.8466	0	0.0203	91.5

Tab. 3.20: tabella dello scenario caratterizzato da valori alti per l'auc riportata dall'articolo scritto da Liu e Zhou (2013)

3.3.4 Alta percentuale di malati

In questo scenario viene modificato quello di base aumentando la percentuale di soggetti malati e modificando di conseguenza tutte le quantità condizionate a questo valore. Nello scenario con un'alta percentuale di malati si ha:

$n = 1000$ [La numerosità]

$X_1 \sim U(-1, 1)$ [La variabile continua (covariata continua)]

$X_2 \sim Bi(1, 0.5)$ [La variabile discreta (covariata discreta)]

$logit(p) = \underline{-0.3} + 0.5X_2 + 0.8X_1$

$D \sim Bi(1, p)$ [Lo stato di salute]

$\mu = 1 + 0.4D + 0.2X_2 + 0.7X_1 + X_2D + 0.5X_1D$

$\sigma = 0.8D + 1.2(1 - D)$

$T \sim N(\mu, \sigma^2)$ [Il valore del test]

$logit(\pi) = -1.2 + T + 0.6X_2 + 1.2X_1$

$V \sim Bi(1, \pi)$ [Lo stato di verifica]

probit [La funzione legame]

La percentuale di malati in questo caso è salita al 49% mentre quella dei verificati si attesta attorno al 59%. I veri valori dei veri θ e l'$AAUC$ calcolati alla stessa maniera del primo scenario sono: $\theta_0 = 0.277$, $\theta_1 = 0.347$, $\theta_2 = 0.693$ e $AAUC = 0.7459$.

Monte Carlo

Le tabelle 3.21 e 3.22 riportano i risultati degli esperimenti Monte Carlo relativi allo scenario in cui, rispetto allo scenario di base, c'è una maggiore percentuale di soggetti malati e ai nuovi stimatori KNN. Tali risultati vanno confrontati con quelli riportati nelle tabelle 3.27 e 3.28 riprese dal lavoro di Liu e Zhou (2013).

Nello specifico, in questo scenario, come si può notare dalle tabelle 3.21 e 3.22, dal punto di vista della distorsione percentuale, i nuovi stimatori degli elementi di θ basati sul KNN si comportano generalmente meglio degli

stimatori "classici" considerati in Liu e Zhou (2013) basati su un modello parametrico non correttamente specificato se non si considerano gli stimatori basati sulla tecnica DR, che però, come già fatto notare, possono dare problemi dal punto di vista computazionale e quindi operativo.
Dalle tabelle 3.21 e 3.22 si può notare come l'andamento degli stimatori basati sulla distanza di Mahalanobis (fatto usuale negli scenari presentati per lo stimatore 3NN, ma non per quello 1NN) si comportano peggio rispetto agli altri basati su quella euclidea. Le stime dell'AAUC in tutti i casi sono corrette.

	$\theta_0 = 0.277$			$\theta_1 = 0.347$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.27832	0.47704	0.06575	0.35492	2.28354	0.08753
euclidea1NN	0.27613	-0.31446	0.13522	0.34452	-0.71468	0.16471
euclidea3NN	0.26444	-4.53292	0.12354	0.35101	1.15506	0.14683
Mahalanobis1NN	0.26224	-5.32759	0.13035	0.36671	5.67983	0.16238
Mahalnobis3NN	0.24504	-11.53702	0.11659	0.38379	10.60329	0.14444

Tab. 3.21: Monte Carlo, n=1000, Replicazioni =1000, perc. alta di malati

	$\theta_2 = 0.693$			$AAUC = 0.7459$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.69708	0.58843	0.10220	0.74614	0.03253	0.01579
euclidea1NN	0.70114	1.17416	0.17598	0.74290	-0.40257	0.02973
euclidea3NN	0.70898	2.30646	0.16357	0.74055	-0.71773	0.02688
Mahalanobis1NN	0.71010	2.46808	0.17365	0.73980	-0.81754	0.02889
Mahalnobis3NN	0.71925	3.78733	0.15693	0.73620	-1.30046	0.02599

Tab. 3.22: Monte Carlo, n=1000, Replicazioni =1000, perc. alta di malati

Bootstrap

Le tabelle 3.23, 3.24, 3.25, 3.26 riportano i risultati (ottenuti sempre nello scenario in cui è stata aumentata la percentuale dei soggetti malati)

relativi all'utilizzo della tecnica bootstrap associata alle simulazioni Monte Carlo e in particolare il livello di copertura per gli intervalli di confidenza di livello nominale del 95%. Come già detto, per motivi meramente computazionali, la numerosità è stata ridotta a $n = 300$. Anche in questo caso vengono effettuate 1000 replicazioni Monte Carlo mentre le replicazione bootstrap sono state fissate a 200.

La copertura degli intervalli di confidenza basata sugli stimatori KNN ha un buon livello di copertura: sempre maggiore del 92%. Si mantiene la tendenza degli stimatori basati sul metodo 3NN di dare stime per gli intervalli più accurate rispetto a quelli basati sul metodo 1NN. La differenza tra le coperture degli intervalli è rilevante solo per quanto riguarda l'AAUC (2.3% e 2.8%).

Se confrontati con gli intervalli ottenuti utilizzando i metodi "classici" riportati nelle colonne "cov" delle tabelle 3.27 e 3.28, gli intervalli di confidenza basati sugli stimatori KNN hanno performance, dal punto di vista della copertura, superiori rispetto a quelli "classici" ottenuti quando i modelli parametrici che ne stanno alla base non sono correttamente specificati; per quanto riguarda lo stimatore basato sul 3NN si ottengono addirittura risultati assimilabili a quelli ottenuti dagli stimatori "classici" con i modelli per la stima di ρ_i e/o π_i correttamente specificati. Come già fatto notare, questo confronto per le differenti numerosità campionarie è soltanto indicativo.

$\theta_0 = 0.277$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.27857	0.56561	0.12281	0.94900	0.95300	0.00800
euclidea1NN	0.25753	-7.02902	0.25995	0.92700	0.92200	0.02700
euclidea3NN	0.23408	-15.49606	0.22162	0.93900	0.94300	0.01400

Tab. 3.23: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di malati

$\theta_1 = 0.347$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.34713	0.03837	0.16730	0.94000	0.94900	0.01300
euclidea1NN	0.33721	-2.82167	0.29595	0.93500	0.93600	0.01700
euclidea3NN	0.34773	0.20912	0.23852	0.94400	0.94500	0.01100

Tab. 3.24: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di malati

$\theta_2 = 0.693$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.70437	1.64041	0.18471	0.94400	0.95000	0.01200
euclidea1NN	0.72481	4.59012	0.32998	0.94700	0.94500	0.02400
euclidea3NN	0.72880	5.16530	0.29022	0.95700	0.95600	0.01100

Tab. 3.25: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di malati

$AAUC = 0.7459$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.74521	-0.09215	0.02895	0.95600	0.95900	0.00700
euclidea1NN	0.73213	-1.84566	0.05773	0.93900	0.92800	0.02300
euclidea3NN	0.72717	-2.51082	0.05037	0.93200	0.93800	0.02800

Tab. 3.26: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di malati

			θ_0			θ_1		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.281	1.6	0.0695	93.3	0.348	0.2	0.0096	93.7
CC	0.514	85.7	0.1023	33.6	0.507	46.1	0.1225	75.8
FI_1	0.277	0.1	0.102	94.8	0.353	1.7	0.1124	96
FI_2	0.392	41.5	0.102	75.2	0.193	-44.3	0.1123	70.7
MSI_1	0.277	0	0.1024	94.5	0.352	1.4	0.1131	96
MSI_2	0.335	21.1	0.1017	88.4	0.283	-18.3	0.1113	91
IPW_1	0.276	-0.2	0.1278	93.5	0.354	2.1	0.1596	93.3
IPW_2	0.338	22.1	0.1358	89.7	0.459	32.3	0.1591	88.2
DR_1	0.283	2.3	0.1153	93.6	0.342	-1.3	0.138	94.2
DR_2	0.284	2.5	0.107	93.2	0.342	-1.5	0.1417	94.2
DR_3	0.282	1.8	0.1113	93.6	0.347	0.1	0.1267	94.1
DR_4	0.306	10.6	0.1114	91.6	0.324	-6.6	0.1277	92.6

Tab. 3.27: tabella riguardante lo scenario in cui la percentuale di malati è più alta rispetto a quello base riportata dall'articolo scritto da Liu e Zhou (2013)

	θ_2				$AAUC$			
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.693	0	0.1027	94.4	0.7466	0.1	0.0168	94.5
CC	0.763	10.1	0.1391	91.4	0.7302	-2.1	0.0217	89.1
FI_1	0.701	1.1	0.1283	94.6	0.7459	0	0.0245	63.6
FI_2	0.518	-25.3	0.1242	66.7	0.7459	0	0.024	93.7
MSI_1	0.699	0.9	0.1296	94.7	0.7452	-0.1	0.0245	93.4
MSI_2	0.633	-8.6	0.1253	89.8	0.7466	0.1	0.0247	92.7
IPW_1	0.699	0.8	0.1723	94.1	0.7444	-0.2	0.0275	93.3
IPW_2	0.716	3.3	0.1759	93.7	0.7765	4.1	0.0267	69.8
DR_1	0.694	0.1	0.1324	93.9	0.7459	0	0.0257	92.5
DR_2	0.693	0	0.1535	94.1	0.7459	0	0.026	92.3
DR_3	0.695	0.3	0.1452	94.7	0.7459	0	0.0253	93.2
DR_4	0.689	-0.6	0.1441	94.5	0.7466	0.1	0.0255	92.8

Tab. 3.28: tabella riguardante lo scenario in cui la percentuale di malati è più alta rispetto a quello base riportata dall'articolo scritto da Liu e Zhou (2013)

3.3.5 Alta percentuale di verificati

In questo scenario, si aumenta soltanto la percentuale dei verificati e si ha:

$n = 1000$ [La numerosità]

$X_1 \sim U(-1, 1)$ [La variabile continua (covariata continua)]

$X_2 \sim Bi(1, 0.5)$ [La variabile discreta (covariata discreta)]

$logit(p) = -1.4 + 0.5X_2 + 0.8X_1$

$D \sim Bi(1, p)$ [Lo stato di salute]

$\mu = 1 + 0.4D + 0.2X_2 + 0.7X_1 + X_2D + 0.5X_1D$

$\sigma = 0.8D + 1.2(1 - D)$

$T \sim N(\mu, \sigma^2)$ [Il valore del test]

$logit(\pi) = \underline{-0.3} + T + 0.6X_2 + 1.2X_1$

$V \sim Bi(1, \pi)$ [Lo stato di verifica]

probit [La funzione legame]

I veri valori degli elementi di θ e dell'$AAUC$ sono ovviamente gli stessi dello scenario di partenza non incidendo il fatto che un soggetto sia verificato o meno su di essi: $\theta_0 = 0.277$, $\theta_1 = 0.693$, $\theta_2 = 0.347$ e $AAUC = 0.7582$. In questo scenario la percentuale di malati, $Pr(D = 1)$, si attesta sempre al 25% mentre quella dei soggetti verificati, $Pr(V = 1)$, è aumentata al 70%.

Monte Carlo

Le tabelle 3.29 e 3.30 riportano i risultati degli esperimenti Monte Carlo relativi allo scenario in cui, rispetto allo scenario di base, c'è una maggiore percentuale di soggetti verificati e ai nuovi stimatori KNN. Tali risultati vanno confrontati con quelli riportati nelle tabelle 3.35 e 3.36 riprese dal lavoro di Liu e Zhou (2013).

Come era normale aspettarsi in questo scenario le distorsioni degli stimatori KNN sono tutte molto basse (tabelle 3.29 e 3.30). Lo stimatore peggiore è quello 3NN che utilizza la distanza di Mahalanobis ma che rimane comunque

minore dei metodi "classici" che si utilizzano su un modello per la stima di ρ_i non correttamente specificato.

	$\theta_0 = 0.277$			$\theta_1 = 0.347$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.27996	1.06684	0.07414	0.34964	0.76028	0.09344
euclidea1NN	0.28066	1.32254	0.12137	0.34710	0.02924	0.14589
euclidea3NN	0.27742	0.15287	0.11206	0.34821	0.35004	0.13385
Mahalanobis1NN	0.27617	-0.29872	0.11847	0.35569	2.50392	0.14561
Mahalnobis3NN	0.26917	-2.82769	0.11115	0.36179	4.26089	0.13476

Tab. 3.29: Monte Carlo, n=1000, Replicazioni =1000, perc. alta di verificati

	$\theta_2 = 0.693$			$AAUC = 0.7582$		
	stima	distorsione %	dev. std	stima	distorsione %	dev. std
full	0.69487	0.26993	0.10349	0.75803	-0.02285	0.01650
euclidea1NN	0.69514	0.30951	0.14763	0.75619	-0.26482	0.02599
euclidea3NN	0.69778	0.69038	0.13926	0.75593	-0.29888	0.02363
Mahalanobis1NN	0.69829	0.76359	0.14394	0.75555	-0.34897	0.02519
Mahalnobis3NN	0.70338	1.49826	0.13893	0.75441	-0.50044	0.02349

Tab. 3.30: Monte Carlo, n=1000, Replicazioni =1000, perc. alta di verificati

Bootstrap

Le tabelle 3.31, 3.32, 3.33, 3.34 riportano i risultati (ottenuti sempre nello scenario in cui è stata aumentata la percentuale dei soggetti malati) relativi all'utilizzo della tecnica bootstrap associata alle simulazioni Monte Carlo e in particolare il livello di copertura per gli intervalli di confidenza di livello nominale del 95%. Come già detto, per motivi meramente computazionali, la numerosità è stata ridotta a $n = 300$. Anche in questo caso vengono effettuate 1000 replicazioni Monte Carlo mentre le replicazione bootstrap sono state fissate a 200.

La copertura degli intervalli di confidenza basata sugli stimatori KNN ha

un buon livello di copertura: sempre maggiore del 91%. Si mantiene la tendenza degli stimatori 3NN di dare stime per gli intervalli più accurate rispetto a quelli basati sull'1NN.

$\theta_0 = 0.277$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.27695	-0.01754	0.13956	0.93700	0.94500	0.01200
euclidea1NN	0.29342	5.92696	0.21778	0.91100	0.92300	0.02000
euclidea3NN	0.28160	1.66032	0.20344	0.92400	0.93100	0.01300

Tab. 3.31: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di verificati

$\theta_1 = 0.347$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.35599	2.59033	0.17644	0.93700	0.94500	0.01200
euclidea1NN	0.33529	-3.37491	0.25096	0.92600	0.93200	0.01000
euclidea3NN	0.34256	-1.27995	0.23160	0.93000	0.93700	0.01500

Tab. 3.32: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di verificati

$\theta_2 = 0.693$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.70843	2.22685	0.19516	0.93400	0.94400	0.01800
euclidea1NN	0.69623	0.46573	0.26214	0.94300	0.93900	0.01200
euclidea3NN	0.70159	1.23954	0.25110	0.94500	0.94600	0.01300

Tab. 3.33: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di verificati

$AAUC = 0.7582$

		distorsione %	dev. std	cov boot	cov norm	confr cover
full	0.75838	0.02333	0.03031	0.94200	0.94600	0.01000
euclidea1NN	0.75592	-0.30097	0.04765	0.93300	0.93100	0.01200
euclidea3NN	0.75393	-0.56379	0.04298	0.93700	0.94000	0.01900

Tab. 3.34: Boot, n=300, Repl monte =1000, Repl boot =250, perc. alta di verificati

	θ_0				θ_1			
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.279	0.9	0.0744	94.8	0.35	0.9	0.0908	94.7
CC	0.113	-59.2	0.0963	57.9	0.481	38.5	0.1121	78.1
FI_1	0.277	0.1	0.096	93.7	0.357	3	0.1102	93.3
FI_2	0.409	47.5	0.0934	66.9	0.166	-52.3	0.1057	55.8
MSI_1	0.278	0.3	0.0967	94	0.355	2.2	0.1114	93.8
MSI_2	0.324	16.9	0.0925	91.1	0.297	-14.5	0.1054	91
IPW_1	0.281	1.5	0.1115	92.1	0.351	1.2	0.1357	92.8
IPW_2	0.321	15.9	0.1118	90	0.445	28.1	0.131	88.2
DR_1	0.283	2.1	0.1062	92.4	0.348	0.3	0.1285	92.9
DR_2	0.283	2	0.1055	92.6	0.348	0.4	0.1276	93.1
DR_3	0.28	1.1	0.0106	92.2	0.348	0.3	0.1231	92.6
DR_4	0.305	10.2	0.0932	91.4	0.331	-4.6	0.119	92.2

Tab. 3.35: tabella riportata dall'articolo scritto da Liu e Zhou (2013)

			θ_2			$AAUC$		
	stima	distorsione %	dev. std	cov	stima	distorsione %	dev. std	cov
full	0.693	0	0.1036	95.4	0.7582	0	0.0168	95
CC	0.748	7.9	0.1229	92.5	0.7574	-1.5	0.0196	91.8
FI_1	0.699	0.8	0.1172	95	0.7567	-0.1	0.0221	94.2
FI_2	0.507	-26.9	0.1125	59	0.7574	-0.2	0.0217	94.4
MSI_1	0.696	0.4	0.119	94.3	0.7574	-0.1	0.0221	94.3
MSI_2	0.65	-6.2	0.1148	93.6	0.7574	-0.1	0.023	94.2
IPW_1	0.693	0	0.1414	94.1	0.7772	-0.1	0.0238	94
IPW_2	0.685	-1.2	0.1405	93.9	0.7574	2.5	0.0219	79.3
DR_1	0.692	-0.2	0.1325	94.5	0.7574	-0.1	0.0235	93.5
DR_2	0.694	0.1	0.1311	93.9	0.7582	0	0.0236	94
DR_3	0.692	-0.1	0.1317	93.8	0.7574	-0.1	0.0233	93.8
DR_4	0.684	-1.3	0.1319	93	0.7582	0	0.0221	94.6

Tab. 3.36: tabella riportata dall'articolo scritto da Liu e Zhou (2013)

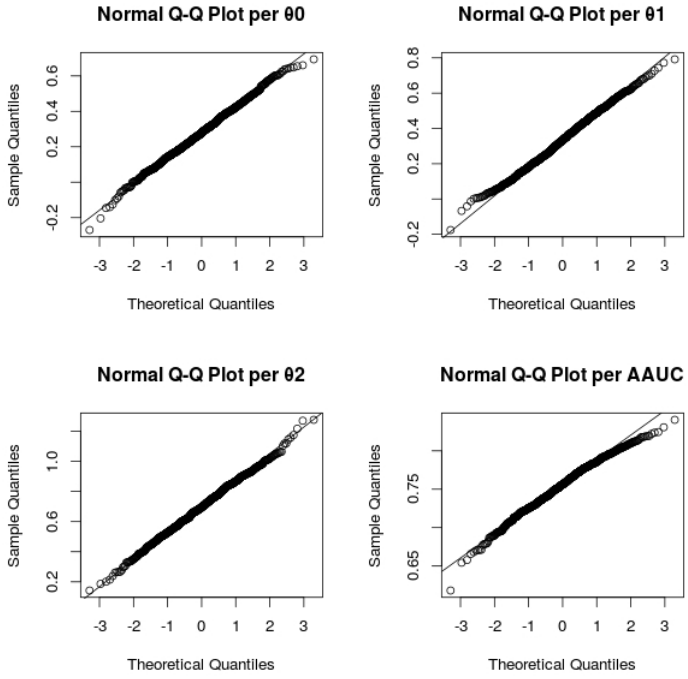
Considerazioni sulla distribuzione asintotica degli stimatori proposti

Come già fatto notare in tutte le simulazione in cui si è utilizzato il bootstrap, il confronto fra l'intervallo basato sui quantili bootstrap e quello basato sulla presunta normalità asintotica degli stimatori, sembra confermare l'ipotesi che quest'ultima assunzione sia verosimile. I dati riportati nei grafici sono stati prodotti con i dati della sola simulazione Monte Carlo del primo scenario per lo stimatore 3NN basato sulla distanza euclidea. Per gli stimatori KNN di tutti i parametri a parte quello per l'$AAUC$ sembra esserci conferma di questa ipotesi. La non normalità degli stimatori dell'$AUCC$ era già stata notata anche in Liu e Zhou (2013). Per ogni parametro è stato riportato un "qq-plot" in cui vengono messi a confronto i valori della stima standardizzati con i quantili di una normale e il valore del test di Shapiro

Wilk in cui l'ipotesi nulla è che i dati siano distribuiti normalmente.

parametro	p-value	parametro	p-value
θ_0	0.4925	θ_1	0.1622
θ_2	0.6417	$AAUC$	0.00003

Tab. 3.37: p-value dei test di Shapiro Wilk



Capitolo 4

Un'applicazione a dati reali

Il calcolo dell'AUC_x e dell'$AAUC$, tramite il metodo proposto in questa tesi, è stato applicato a dei dati reali raccolti per la ricerca sul morbo di Alzheimer, dati utilizzati anche nell'articolo di Liu e Zhou (2013). I dati derivano da un sottoinsieme dell'"Uniform Data Set" (USD) del "National Alzheimer's Coordinaiting Center" (NACC) e sono stati raccolti, con uno studio di follow-up, da 32 centri per la ricerca sull'Alzheimer situati nel Nord America a partire dal 2006.

Con questi dati si è deciso di testare la capacità del "mini-mental state examination" (MMSE) di prevedere il passaggio dallo stato di "amnestic mild cognitive impairment" (aMCI) a quello di demenza a distanza di un anno (operativamente nell'arco di 6-18 mesi). Il MMSE è un test, basato su un questionario con trenta livelli di punteggio finali, che viene utilizzato per valutare lo stato mentale di un paziente. L'aMCI è uno stato di transizione verso la demenza, ma a differenza di quest'ultima non è irreversibile.

Si sono considerati soltanto i soggetti con una età maggiore di 65 anni a cui alla prima visita è stato diagnosticato lo stato di aMCI; ai pazienti che hanno effettuato una seconda visita nell'arco dei 6-18 mesi successivi alla prima è stato assegnato il valore $D = 1$ nel caso fossero transitati nello stato di demenza, $D = 0$ altrimenti e il valore V=1 in entrambi i casi; per tutti gli altri pazienti la variabile D è stata considerata ignota e di conseguenza V=0. Inoltre, per semplicità, sono stati esclusi quei soggetti le cui variabi-

li utilizzate nell'analisi non fossero, anche in parte, presenti. Dopo questa selezione i soggetti considerati sono 3103 e la percentuale di soggetti non verificati risulta del 39,3%.

Quindi nel seguito consideriamo l'MMSE come test diagnostico, di cui modelliamo le sue capacità predittive, in funzione di alcune caratteristiche dei soggetti. Le variabili per le quali è stata calcolata l'AUC_x e l'$AAUC$ sono l'età, il sesso, l'etnia, l'educazione, la situazione abitativa, se il soggetto in passato ha avuto infarti e se ha avuto problemi cardiovascolari (Tabella 4.2).

Per ottenere le stime KNN $\hat{\rho}_{Ki}$, le variabili che sono state considerate per ogni soggetto sono l'MMSE, l'età, lo standard CDR sum of boxes che è un punteggio clinico per la demenza, la pressione diastolica, l'età dall'inizio del declino cognitivo, la depressione, un indicatore dicotomico rigurdante i traumi celebrali presenti nella storia clinica del paziente e un altro per il colesterolo alto; queste ultime tre sono le uniche variabili categoriali fra le otto considerate. Nella tabella 4.1 sotto sono state riportate alcune statistiche descrittive.

Per le stime degli elementi di θ nel modello per il calcolo dell'AUC_x e dell'AUCC sono stati utilizzati come nelle precedenti simulazioni K=1 e K=3 vicini più vicini; a supporto di questa scelta sono state effettuata delle convalide incrociate sui soli dati verificati, nella modalità descrittà nel capitolo 2, e si è notato che su i cinque valori considerati (1,3,5,7,10) i due scelti erano quelli con discrepanza minore (Tabella 4.3). Sempre basandosi sullo stesso parametro si è scelta la distanza da usare. La scelta della distanza di Mahalanobis è supportata anche dalla differenza di scala delle variabili utilizzate. Sono state poi utilizzate 500 replicazioni bootstrap per il calcolo dello deviazione standard di stima e del livello di significatività osservato relativo al test per l'ipotesi che gli elementi di θ siano diversi da 0 e l'AAUC sia diversa da 0.5.(Tabelle 4.4 4.5)

Sono state inoltre calcolate le medesime stime anche utilizzando il metodo

variabile	D=0		D=1		D=NA		totale	
n=	1400		484		1219		3103	
	media	varianza	media	varianza	media	varianza	media	varianza
MMSE	27.04	5.83	25.96	6.24	26.79	6.06	26.77	6.33
età	76.58	41.99	78.04	44.45	76.80	46.22	76.89	22.26
CDR	1.36	1.02	2.06	1.70	1.50	1.15	1.52	1.23
pressione	53.77	129.01	53.05	123.98	53.17	154.11	53.42	138.10
età declino	73.54	52.91	74.63	51.94	73.84	53.87	73.82	53.24
	1 vs n		1 vs n		1 vs n		1 vs n	
depressione	0.19		0.27		0.23		0.22	
traumi	0.10		0.07		0.10		0.10	
colesterolo	0.58		0.54		0.57		0.57	

Tab. 4.1: Statistiche descrittive delle varibili usate per il calcolo della stima $\hat{\rho}_{Ki}$

di imputazione MSI, una volta con un modello per la stima dei ρ_i con tutte le variabili utilizzate nel caso KNN e tutte le loro interazioni con il MMSE e con una funzione logit come funzione legame(Tabella 4.6).
Infine è stato utilizzato lo stimatore definito "Complete Case" (CC), lo stimatore che utilizza per la stima dei parametri solamente le osservazioni verificate, cioè quello che più risente della eventuale distorsione di verifica (Tabella 4.7).
Per tutti i gli stimatori, KNN e "classici", è stata utilizzata come funzione legame una probit.

variabile	D=0	D=1	D=NA	totale
n=	1400	484	1219	3103
Donne	0.45	0.48	0.53	0.49
etnia(bianca vs altre)	0.15	0.13	0.21	0.17
educazione 12- anni	0.26	0.27	0.31	0.28
educazione 12-16 anni	0.43	0.44	0.40	0.42
educazione 16+ anni	0.31	0.29	0.30	0.30
situzione abitativa (solo vs altre)	0.76	0.79	0.72	0.75
infarto (si vs no)	0.04	0.06	0.05	0.05
problemi cardiaci (si vs no)	0.13	0.13	0.12	0.12

Tab. 4.2: Statistiche descrittive delle varibili usate per il calcolo dell'AUC_x e dell'$AAUC$ (per l'età e l'MMSE si rimanda alla tabella precedente)

k	1	3	5	7	10
Euclidea	0.3525	0.3519	0.3573	0.3568	0.3583
Mahalanobis	0.3375	0.3388	0.3403	0.3396	0.3431

Tab. 4.3: indice di discrepanza per vari K: $\frac{1}{n_{ver}} \sum_{i=1}^{n_{ver}} |D_i - \hat{\rho}_{Ki}|$

	theta	sd boot	p-value		IC 95% (normalità)
intercetta	0.7183	0.4731	0.126		(-0.2089 , 1.6456)
età	-0.0067	0.0057	0.236		(-0.0179 , 0.0045)
donne	-0.0275	0.0720	0.710		(-0.1686 , 0.1136)
etnia	-0.0449	0.0986	0.632		(-0.2381 , 0.1484)
educmedia	0.0632	0.0844	0.468		(-0.1022 , 0.2285)
educalt	0.1291	0.0944	0.188		(-0.0559 , 0.314)
sit abit	0.1073	0.0872	0.238		(-0.0636 , 0.2782)
infarto	-0.1786	0.1484	0.230		(-0.4694 , 0.1123)
cardiovasc	0.1413	0.1058	0.180		(-0.066 , 0.3487)
AAUC	0.6275	0.0158	0.000	*	(0.5965 , 0.6584)

Tab. 4.4: Test MMSE K=1, B=500

	theta	sd boot	p-value		IC 95% (normalità)
intercetta	0.7306	0.4450	0.110		(-0.1415 , 1.6027)
età	-0.0072	0.0054	0.182		(-0.0179 , 0.0034)
donne	0.0214	0.0672	0.776		(-0.1103 , 0.1531)
etnia	-0.0590	0.0801	0.522		(-0.216 , 0.098)
educmedia	0.0692	0.0750	0.356		(-0.0778 , 0.2162)
educalt	0.1931	0.0877	0.030	*	(0.0211 , 0.365)
sit abit	0.0773	0.0733	0.304		(-0.0663 , 0.221)
infarto	-0.2020	0.1402	0.148		(-0.4768 , 0.0729)
cardiovasc	0.1028	0.0902	0.246		(-0.074 , 0.2797)
AAUC	0.6219	0.0147	0.000	*	(0.5931 , 0.6507)

Tab. 4.5: Test MMSE K=3, B=500

	theta	sd boot	p-value		IC 95% (normalità)
intercetta	0.6015	0.2715	0.032	*	(0.0694 , 1.1336)
età	-0.0045	0.0033	0.174		(-0.011 , 0.0019)
donne	-0.0047	0.0505	0.926		(-0.1037 , 0.0944)
etnia	-0.0233	0.0659	0.722		(-0.1525 , 0.1058)
educmedia	0.0355	0.0572	0.516		(-0.0766 , 0.1477)
educalt	0.1519	0.0644	0.016	*	(0.0257 , 0.2781)
sit abit	0.0326	0.0577	0.560		(-0.0805 , 0.1457)
infarto	-0.2513	0.1123	0.030	*	(-0.4715 , -0.0311)
cardiovasc	0.0775	0.0747	0.288		(-0.0689 , 0.2238)
AAUC	0.6250	0.0115	0.000	*	(0.6025 , 0.6475)

Tab. 4.6: Test MMSE MSI, B=500

	theta	sd boot	p-value		IC 95% (normalità)
intercetta	0.83	0.5095	0.114		(-0.1686 , 1.8286)
età	-0.008	0.0062	0.184		(-0.0201 , 0.0041)
donne	-0.0055	0.0819	0.956		(-0.166 , 0.1551)
etnia	-0.0582	0.1202	0.63		(-0.2938 , 0.1774)
educmedia	0.0767	0.097	0.434		(-0.1134 , 0.2669)
educalt	0.2547	0.1111	0.03	*	(0.037 , 0.4724)
sit abit	0.0519	0.1015	0.592		(-0.1471 , 0.2509)
infarto	-0.3468	0.1761	0.048	*	(-0.692 , -0.0016)
cardiovasc	0.0748	0.123	0.518		(-0.1663 , 0.3159)
AAUC	0.6295	0.0147	0.000		(0.6007 , 0.6584)

Tab. 4.7: Test MMSE Complete Case, B=500

Una prima considerazione viene fatta a partire dal confronto dello stimatore che utilizza solo i dati verificati (CC) (Tabella 4.7) con lo stimatore basato sulla tecnica di imputazione MSI (Tabella 4.6). A parte l'intercetta che nel primo caso non è significativamente diversa da zero, mentre lo è nel secondo caso, gli stimatori portano a risultati simili. Questo fatto fa sorgere il dubbio che nel dataset considerato il meccanismo che genera i dati mancanti sia del tipo completamente a caso (MCAR) e non MAR, ipotesi sotto la quale è stato sviluppato tutto il lavoro. Mancando questa condizione potrebbe venire a mancare pure la distorsione di verifica come sembra suggerire appunto la somiglianza dei risultati nei casi sopra richiamati (stimatori CC e MSI).

Tralasciando questa peculiarità dello studio, i risultati ottenuti con lo stimatore 1NN 4.4 indicano che nessuna delle covariate considerate influisca significativamente sulla capacità del test MMSE di discriminare i soggetti che ad un anno regrediranno allo stato di demenza da quelli che non lo faranno. Inoltre la stima dell'$\hat{AAUC} = 0.6275$, seppur significativamente diversa da 0.5, indica la scarsa capacità dell'MMSE di prevedere il vero stato di salute.

Dello stesso avviso, per quanto riguarda l'$AAUC$ è il risultato ottenuto dallo stimatore basato sul 3NN 4.5. Questo stimatore però attribuisce alla covariata "istruzione alta", se presente, un effetto migliorativo della capacità predittiva del test MMSE.

Capitolo 5

Conclusione

Il nuovo metodo di stima proposto sembra risolvere sia il problema della distorsione di verifica, sia la distorsione dovuta ad una errata specificazione dei modelli parametrici utilizzati dagli stimatori proposti in Liu e Zhou (2013) con cui è stato confrontato. In generale l'utilizzo di uno stimatore 3NN sembra funzionare meglio, per quanto riguarda la copertura degli intervalli di confidenza; mentre per quanto riguarda la distanza da utilizzare per la determinazione dei vicini più vicini sembra che la scelta debba, in generale, dipendere dalla natura dei dati.

Un ulteriore sviluppo per questo stimatore potrebbe essere lo studio analitico delle sue proprietà asintotiche di cui ne è stata data un'idea tramite gli studi simulativi.

Dal punto di vista computazionale se non si ha a disposizione una macchina con grandi quantità di RAM, il metodo basato sul KNN porta notevoli vantaggi. Infatti sia utilizzando il linguaggio Julia che il software R, per poter applicare completamente i metodi "classici" (che si basano sui modelli parametrici) ai dati reali proposti sono necessari più di 8 GB di RAM. Dall'altra l'utilizzo del bootstrap per la stima degli intervalli di confidenza amplia notevolmente i tempi di elaborazione.

In ogni caso si fa notare la grande mole computazionale del problema; ogni stimatore, che sia KNN o "classico", richiede di effettuare la stima di 2*p coefficienti di un modello di regressione binomiale, dove p corrisponde al nu-

mero di covariate di interesse, e con numerosità campionaria pari al quadrato della numerosità di partenza.

Appendice A

Julia Language

Per compiere le simulazioni e l'analisi dei dati reali in questa tesi è stato utilizzato un recente linguaggio di programmazione: Julia (http://julialang.org/). Questo linguaggio è stato pubblicato nel 2012 sotto licenza MIT ed è ancora in fase di sviluppo. Le motivazioni che hanno portato alla sua creazione è stata la necessità degli autori, dei ricercatori del MIT, di avere uno strumento "utilizzabile per la programmazione generica al pari di Python, facile per le statistiche come R, naturale per processare le stringhe come Perl, potente per l'algebra lineare come Matlab, che abbia la capacità di unire diversi programmi come la shell", e non veloce come C ma quasi. (Jeff Bezanson, 2012) (Wikipedia, 2015)

La velocità computazionale, oltre a un discreto risparmio di risorse, è stato il motivo per cui è stato utilizzato Julia al posto di R (R Core Team, 2015) considerato il fatto che anche in Julia esiste una shell interattiva per l'esecuzione di singoli comandi

Dal punto di vista della programmazione la sintassi in parte ricorda quella di R, infatti non ci sono state grosse difficoltà nel passare da un linguaggio all'altro considerata anche la relativa semplicità del codice prodotto. Per chi passa da un linguaggio all'altro la più grande difficoltà è forse la differente filosofia adottata nei confronti del calcolo vettoriale, mentre in R ne è incoraggiato l'uso, generalmente in Julia si preferisce l'utilizzo di cicli. Dal punto di vista operativo i principali vantaggi e svantaggi di Julia rispetto a

R sono:

Pro:

- La velocità computazionale
- L'utilizzo performante di cicli
- L'accessibilità, sia in termini di reperibilità che comprensione, ai codici delle funzioni esterne
- La semplicità nel richiamare velocemente funzioni da altri linguaggi come C o R

Contro:

- La mancanza di una documentazione esauriente
- La mancanza di un IDE come Rstudio, quello fornito con Julia, Juno, non è orientato all'analisi dei dati
- La maggior semplicità nella manipolazione dei dati in R

L'utilizzo performante dal mio punto di vista dei due strumenti, poichè Julia permette di salvare le variabili all'interno di un archivio .RData, è l'utilizzo di Julia per la "forza bruta" di calcolo e quello di R per le successive analisi.

Appendice B

Codice

B.1 simulazione

knntot.jl

In questo file sono contenute le funzioni utilizzate nel file main.jl

```
1
2 using Distributions #contiene le pi\'u comuni
    distribuzioni statistiche
3 using GLM #contiene le funzioni per stimare i glm
4 using DataFrames # pacchetto per avere un formato
    di dati pi\'u flessibile
5 using RCall # serve a poter richiamare le funzioni
    da R
6 using Distances # contiene le funzioni per il
    calcolo delle distanze
7 # Le righe precedenti servono per richiamare dei
    pacchetti
8 function real(TT,DD,XX,i_index,j_index)
9 #questa funzione stima i theta e l'aauc in caso di
    assenza di dati mancanti
```

```
10 ## TT valore del test, DD stato di salute, XX
     covariate, i_index e j_index sono degli indici
     che uniti formano tutte le coppie possibili dei
     numeri da uno a n
11   n=size(TT,1)
12   # calcola la numerosit\'a nella prima dimensione
       dell'array
13   ps_Di=DD[i_index]
14   ps_Dj=DD[j_index]
15   # sono selezionati i dati per i rispettivi indici
16   wt=Array{Float64}[ps_Di.*(1-ps_Dj)][]
17   # il simbolo . prima di un operatore fa le
       operatore elemento e non elemento, non in senso
        matriciale
18   wt[(0:(n-1))*n+(1:n)]=0
19   #assegna a il valore di wt quando l'i-esimo
       valore di j_index coincide con quello di
       i_index
20   notzeri=[wt.!=0]
21   # quardo per quali valori wt non vale zero
22   i_index=i_index[notzeri]
23   j_index=j_index[notzeri]
24   ps_Ti=TT[i_index]
25   ps_Tj=TT[j_index]
26   wt=wt[notzeri]
27   n=size(ps_Tj,1)
28   y=Int64[(ps_Ti.>ps_Tj)+1/2*(ps_Ti.==ps_Tj)]
29   #calcolo i valori di I
30   test=DataFrames.DataFrame(x1 =XX[i_index,1],x2 =
       XX[i_index,2], x3= XX[j_index,1],x4= XX[j_index
       ,2], y = y)
```

```
31   #creo un dataset
32   fit1=glm(y ~ x1+x2+x3+x4 ,test, Binomial(),
        ProbitLink(), wts=wt)
33   #stima un glm basato sulla distribuzione
        binomiale, con funzione legame una probit e con
         wt come pesi
34   theta=coef(fit1)
35   Xmat=[ones(size(XX,1)) XX XX]
36   AUCi=cdf(Normal(),(Xmat*theta))
37   #cdf(Normal(),p) calcola il valore del punto
        nella funzione di ripartizione di una normale
38   AAUC=sum(AUCi.*DD)/sum(DD)
39   g1=theta[1]
40   g2=theta[2]+theta[4]
41   g3=theta[3]+theta[5]
42   return [g1,g2,g3, AAUC]
43 end
44
45
46
47
48 function est_knnmahl(TT,DD,XXST,i_index,j_index,K)
49 #stima i parametri utilizzando la distanza di
     Mahalanobis
50 ## TT valore del test, DD stato di salute, XXST
     covariate, i_index e j_index sono degli indici
     che uniti formano tutte le coppie possibili dei
     numeri da uno a n
51   n=size(TT,1)
52   VV=!isna(DD)
53   VV1=[1:n][VV]
```

```
54  # prendo gli indici dei verificati
55  VV0=[1:n][!VV]
56  # prendo gli indici dei non verificati
57  DDD=copy(DD)
58  DDD[VV0]=0
59  #imposto gli na a 0
60  XX=XXST[:,1]
61  #XX  la variabile continua
62  ST=XXST[:,2]
63  #XS  la variabile discreta
64  D_KNN=Array(Float64,n)
65  pD=Array(Float64,n)
66  S=Array(Float64,n)
67  T=Array(Float64,n)
68  X=Array(Float64,n)
69  V=Array(Bool,n)
70  Dusf=Array(Float64,n)
71  for i in 0:1
72  #prima prendo i dati di una discreta poi quelli
       dell'altra
73    j=[1:n][ST.==i]
74    #indici della i-esima discreta
75    VV1j=intersect(VV1,j)
76    #indici dei verificati e i-esima discreta
77    VV0j=intersect(VV0,j)
78    #indici dei non verificati e i-esima discreta
79  ####
80    T1=TT[VV1j]
81    T0=TT[VV0j]
82    X1=XX[VV1j]
83    X0=XX[VV0j]
```

```
84      S1=ST[VV1j]
85      S0=ST[VV0j]
86    #####divido i verificati dai non verificati
87      l1=size(pD[j],1)
88    ####
89      Dusf[j]=[DDD[VV1j],DDD[VV0j]]
90      X[j]=[X1,X0]
91      T[j]=[T1,T0]
92      S[j]=[S1,S0]
93      V[j]=[VV[VV1j],VV[VV0j]]
94    ###sostituisco le osservazioni utilizzate con
         quelle ordinate
95      dist=distanzamahl([X1 T1],[X[j] T[j]])
96      #calcolo le distanze tra i verificati e tutti
           gli altri
97      pD[j]=rho_knn(dist,K,Dusf[j],l1)
98      #calcolo la media dei k vicini piu' vicini
99      D_KNN[j]=Dusf[j]
100     D_KNN[j[V[j].==0]]=pD[j[V[j].==0]]
101   end
102   XS=[X S]
103   ps_Di=D_KNN[i_index]
104   ps_Dj=D_KNN[j_index]
105   wt=Array{Float64}[ps_Di.*(1-ps_Dj)][]
106   wt[(0:(n-1))*n+(1:n)]=0
107   notzeri=[wt.!=0]
108   i_index=i_index[notzeri]
109   j_index=j_index[notzeri]
110   ps_Ti=T[i_index]
111   ps_Tj=T[j_index]
112   wt=wt[notzeri]
```

```
113   n=size(ps_Tj,1)
114   y=[(ps_Ti[i]>ps_Tj[i])+1/2*(ps_Ti[i]==ps_Tj[i])
          for i=1:n]
115   test=DataFrames.DataFrame(x1 =XS[i_index,1],x2 =
          XS[i_index,2], x3= XS[j_index,1],x4= XS[j_index
          ,2], y = y)
116   fit1=glm(y ~ x1+x2+x3+x4 ,test, Binomial(),
          ProbitLink(), wts=wt)
117   theta=coef(fit1)
118   Xmat=[ones(size(XS,1)) XS XS]
119   AUCi=cdf(Normal(),(Xmat*theta))
120   AAUC=sum(AUCi.*D_KNN)/sum(D_KNN)
121   g1=theta[1]
122   g2=theta[2]+theta[4]
123   g3=theta[3]+theta[5]
124   return [g1,g2,g3, AAUC]
125 end
126
127 function est_knn(TT,DD,XXST,i_index,j_index,K)
128 #stima i parametri utilizzando la distanza euclidea
129 ## TT valore del test, DD stato di salute, XXST
       covariate, i_index e j_index sono degli indici
       che uniti formano tutte le coppie possibili dei
       numeri da uno a n
130   n=size(TT,1)
131   VV=!isna(DD)
132   VV1=[1:n][VV]
133   # prendo gli indici dei verificati
134   VV0=[1:n][!VV]
135   # prendo gli indici dei non verificati
136   DDD=copy(DD)
```

```
137   DDD[VV0]=0
138   #imposto gli na a 0
139   XX=XXST[:,1]
140   #XX  la variabile continua
141   ST=XXST[:,2]
142   #XS  la varibile discreta
143   D_KNN=Array(Float64,n)
144   pD=Array(Float64,n)
145   S=Array(Float64,n)
146   T=Array(Float64,n)
147   X=Array(Float64,n)
148   V=Array(Bool,n)
149   Dusf=Array(Float64,n)
150   for i in 0:1
151   #prima prendo i dati di una discreta poi quelli
        dell'altra
152     j=[1:n][ST.==i]
153     #indici della i-esima discreta
154     VV1j=intersect(VV1,j)
155     #indici dei verificati e i-esima discreta
156     VV0j=intersect(VV0,j)
157     #indici dei non verificati e i-esima discreta
158     T1=TT[VV1j]
159     T0=TT[VV0j]
160     X1=XX[VV1j]
161     X0=XX[VV0j]
162     S1=ST[VV1j]
163     S0=ST[VV0j]
164     l1=size(pD[j],1)
165     Dusf[j]=[DDD[VV1j],DDD[VV0j]]
166     X[j]=[X1,X0]
```

```
167     T[j]=[T1,T0]
168     S[j]=[S1,S0]
169     V[j]=[VV[VV1j],VV[VV0j]]
170     dist=distanza([X1 T1],[X[j] T[j]])
171     #calcolo le distanze tra i verificati e tutti
           gli altri
172     pD[j]=rho_knn(dist,K,Dusf[j],l1)
173     #calcolo la media dei k vicini piu' vicini
174     D_KNN[j]=Dusf[j]
175     D_KNN[j[V[j].==0]]=pD[j[V[j].==0]]
176   end
177   XS=[X S]
178   ps_Di=D_KNN[i_index]
179   ps_Dj=D_KNN[j_index]
180   wt=Array{Float64}[ps_Di.*(1-ps_Dj)][]
181   wt[(0:(n-1))*n+(1:n)]=0
182   notzeri=[wt.!=0]
183   i_index=i_index[notzeri]
184   j_index=j_index[notzeri]
185   ps_Ti=T[i_index]
186   ps_Tj=T[j_index]
187   wt=wt[notzeri]
188   n=size(ps_Tj,1)
189   y=[(ps_Ti[i]>ps_Tj[i])+1/2*(ps_Ti[i]==ps_Tj[i])
         for i=1:n]
190   test=DataFrames.DataFrame(x1 =XS[i_index,1],x2 =
         XS[i_index,2], x3= XS[j_index,1],x4= XS[j_index
         ,2], y = y)
191   fit1=glm(y ~ x1+x2+x3+x4 ,test, Binomial(),
         ProbitLink(), wts=wt)
192   theta=coef(fit1)
```

```
193   Xmat=[ones(size(XS,1)) XS XS]
194   AUCi=cdf(Normal(),(Xmat*theta))
195   AAUC=sum(AUCi.*D_KNN)/sum(D_KNN)
196   g1=theta[1]
197   g2=theta[2]+theta[4]
198   g3=theta[3]+theta[5]
199   return [g1,g2,g3, AAUC]
200 end
201
202
203 function distanza(x0,x1)
204 #calcola la distanza euclidea fra le righe di una
      matrice e quelle di un'altra
205   dime=size(x0)
206   noss0=dime[1]
207   ncov=dime[2]
208   noss1=size(x1)[1]
209   dis=Array(Float64,(noss0,noss1))
210   app=Array(Float64,(noss1,ncov))
211   for i in 1:noss0
212     for j in 1:ncov
213       app[:,j]=(x1[:,j]-x0[i,j]).^2
214     end
215     dd=[sqrt(sum(app[l,:])) for l in 1:noss1]
216     dis[i,:]=dd
217   end
218   return dis
219 end
220
221
222
```

```
223
224 function rho_knn(mat,k,D,l1)
225 #calcola i k vicini pi\'u vicini e ne fa la media
      dei rispettivi valori di D
226   id = colmink(mat,k)
227   #trova le posizioni delle k minime distanze  rho
        = Array(Float64,l1)
228   for i in 1:l1
229     rho[i] = mean(D[id[:,i]])
230     #calola i vari rho per le varie osservazioni
231   end
232   return rho
233 end
234
235
236
237 function colmink(m,k)
238   #trova la posizione dei k minimi di ogni colonna
        di una matrice
239   ncolonne=size(m,2)
240   mini=Array(Int,(k,ncolonne))
241   for t in 1:ncolonne
242     mini[:,t]=min_k(m[:,t],k)
243   end
244   return mini
245 end
246
247 function min_k(x,k)
248   #trova la posizione dei k minimi di un vettore
249   id=Array(Int,k)
250   z=copy(x)
```

```
251   for(i in 1:k)
252     id[i] = findmin(z)[2]
253     z[id[i]] = Inf
254   end
255   return id
256 end
257
258
259
260
261
262 function distanzamahl(x0,x1)
263 #calcola la distanza di Mahalanobis fra le righe di
      una matrice e quelle di un'altra
264   covar=varcov(x1)
265   #calcola la matrice di varianza/covarianza
266   covinv=inv(covar)
267   #inverte la matrice
268   dime=size(x0)
269   noss0=dime[1]
270   ncov=dime[2]
271   noss1=size(x1)[1]
272   dis=Array(Float64,(noss0,noss1))
273   app=Array(Float64,(noss1,ncov))
274   for i in 1:noss0
275     for j in 1:ncov
276       app[:,j]=(x1[:,j]-x0[i,j])
277     end
278     dd=[sqrt(app[l,:]*covinv*app[l,:]')[1] for l in
          1:noss1]
279     dis[i,:]=dd
```

```
280   end
281   return dis
282 end
283
284
285
286
287
288 function varcov(X)
289 #calcola la matrice di varianza/covarianza
290   n=size(X,1)
291   H=diagm(ones(n)).-1/n
292   S=X'*H*X./n
293   return S
294 end
295
296
297 function expit(x)
298   return exp(x)/(1+exp(x))
299 end
300
301 function sd(x)
302   return sqrt(var(x))
303 end
```

main

```
1 require("knntot.jl")
2 #richiama il file knntot.jl e le sue funzioni
3 function corri(B)
```

```
4 #questa funzione accetta B che \'e il numero di
    replicazioni Monte Carlo da eseguire, al suo
    interno per ogni replicazione utilizza la tecnica
     bootstrap
5   g=1
6   n=300
7   #numerosit\'a del campione
8   repl=250
9   #replicazioni bootstrap
10  matreal=Array(Float64,(B*repl,4))
11  mateucl1=Array(Float64,(B*repl,4))
12  mateucl3=Array(Float64,(B*repl,4))
13  matmahl1=Array(Float64,(B*repl,4))
14  matmahl3=Array(Float64,(B*repl,4))
15  for b in 1:B
16    print(b)
17    X2=rand(Uniform(-1,1),n)
18    # genera n realizzazioni di una uniforme
19    X1=rand(Binomial(1,0.5),n)
20    #genera n realizzazioni di una binomiale
21    XX=[X2 X1]
22    p=Float64[expit(-1.4+0.5*X1[i]+0.8*X2[i]) for i
        =1:n]
23    real_D=[rand(Binomial(1,p[i])) for i=1:n]
24    mu=[1+0.4*real_D[i]+0.2*X1[i]+0.7*X2[i]+X1[i]*
        real_D[i]+0.5*X2[i]*real_D[i] for i=1:n]
25    sigma=[0.8*real_D[i]+1.2*(1-real_D[i]) for i=1:
        n]
26    TT=[rand(Normal(mu[i],sigma[i])) for i=1:n]
27    p2=[expit(-1.2+TT[i]+0.6*X1[i]+1.2*X2[i]) for i
        =1:n]
```

```
28      VV=[rand(Binomial(1,p2[i])) for i=1:n]
29      DD=DataArray(real_D)
30      DD[VV.==0]=NA
31      n=size(TT,1)
32      i_index=Array(Int64, n*n)
33      k=1
34      for i=1:n
35        for j=1:n
36          i_index[k]=i
37          k=k+1
38        end
39      end
40      j_index=Array(Int64, n*n)
41      k=1
42      for i=1:n
43        for j=1:n
44          j_index[k]=j
45          k=k+1
46        end
47      end
48      #i_index e j_index sono degli indici che uniti
           formano tutte le coppie possibili dei numeri
           da uno a n
49      sample=[1:n reshape(rand(1:n,n*(repl-1)),(n,(
           repl-1)))]
50      for i in 1:repl
51        matreal[g,:]= real(TT[sample[:,i]],real_D[
             sample[:,i]],XX[sample[:,i],:],i_index,
             j_index)
52        mateucl1[g,:]= est_knn(TT[sample[:,i]],DD[
             sample[:,i]],XX[sample[:,i],:],i_index,
```

```
            j_index,1)
53        mateucl3[g,:]= est_knn(TT[sample[:,i]],DD[
            sample[:,i]],XX[sample[:,i],:],i_index,
            j_index,3)
54        matmahl1[g,:]= est_knnmahl(TT[sample[:,i]],DD
            [sample[:,i]],XX[sample[:,i],:],i_index,
            j_index,1)
55        matmahl3[g,:]= est_knnmahl(TT[sample[:,i]],DD
            [sample[:,i]],XX[sample[:,i],:],i_index,
            j_index,3)
56        g=g+1
57      end
58    end
59    return matreal, mateucl1, mateucl3, matmahl1,
        matmahl3
60  end
61
62
63
64
65
66
67  B=1000
68  t=time()
69  r=[corri(B)]
70  tg=time()-t
71
72  reali=r[1][1]
73  euc1=r[1][2]
74  euc3=r[1][3]
75  mah1=r[1][4]
```

```
76 mah3=r[1][5]
77
78 medie=Array(Float64,(3,4))
79 deviazioni=Array(Float64,(3,4))
80
81 for(t in 1:3)
82   medie[t,:]=[mean([r[1][t][i,j] for i in 1:B]) for
       j in 1:4]
83   deviazioni[t,:]=[sd(Float64[r[1][t][i,j] for i in
       1:B]) for j in 1:4]
84 end
85
86 globalEnv[:reali]=reali
87 globalEnv[:euc1]=euc1
88 globalEnv[:euc3]=euc3
89 globalEnv[:mah1]=mah1
90 globalEnv[:mah3]=mah3
91 globalEnv[:tg]=tg
92 "save.image('~/datiboot.RData')" |>rcopy
93 "save.image('datiboot.RData') "|>rcopy
94 s = open("datiboot.txt","w")
95 #serve per creare un file su cui poter salvare i
     risulati
96 print(s,medie)
97 print(s,deviazioni)
98 close(s)
```

B.2 datireali

Questo è il codice utilizzato per le stime sui dati reali

```
1
```

```
2 require("knntot.jl")
3
4 dati=readtable("~/datiridottijulia.txt", separator
    = ' ',header=true)
5 #serve a leggere un dataset
6 VT=dati[:verificato].==1
7 # dati[:verificato] serve a selezionare tutti le
    osservazioni della variabile verificato da dati
8 dati=dati[:,2:end]
9 B=501
10 #numero di replicazioni bootstrap + 1
11 risultati=Array(Float64,(10,B))
12 righe=size(dati,1)
13 campionamento=Array(Float64,(righe,B))
14 campionamento=[1:righe reshape(rand(1:righe,righe*(
    B-1)),(righe,(B-1)))]
15 t=time()
16 n=righe
17 ti_index=Array(Int64, n*n)
18 for i=1:n
19   ti_index[(1+(i-1)*n):(i*n)]=i
20 end
21 tj_index=Array(Int64, n*n)
22 for j=1:n
23   tj_index[(1:n)*n-(n-j)]=j
24 end
25
26 dati[:race]=1
27 dati[:race][dati[:NACCNIHR].==1]=0
28 dati[:LIVSIT][dati[:LIVSIT].>2]=2
29 kkk=dati[:EDUC]
```

```
30 dati[:EDUC]="A"
31 dati[:EDUC][kkk.>12]="B"
32 dati[:EDUC][kkk.>17]="C"
33
34 pool!(dati,[:SEX,:race,:EDUC,:LIVSIT,:STROKE,:
       CVOTHR])
35 #serve a indicare le variabili categoriali
36 k=1
37 #il numero di k del Knn
38 for b in 1:B
39   sam=campionamento[:,b]
40   datiperd=dati[sam,[:demenza,:verificato,:MMSE,:
       NACCAGE,:CDRSUM,:BPDIAS,:NACCAGED,:DEP,:
       TRAUMHIST,:HYPERCHO]]
41   rho= DataArray{Float64}[datiperd[:demenza]][]
42   for a in 0:1
43     ai=[1:n][datiperd[:DEP].==a]
44     for z in 0:1
45       bi=[1:n][datiperd[:TRAUMHIST].==z]
46       for d in 0:1
47         di=[1:n][datiperd[:HYPERCHO].==d]
48         fine=intersect(ai,bi,di)
49         # intersezione fra gli insiemi
50         nf=size(fine,1)
51         datitmp=datiperd[fine,3:7]
52         datidist=convert(DataArray,datitmp)
53         distanze=DataArray(Float64,(nf,nf))
54         Q=inv(Array{Float64}[varcov(datidist)][])
55         for i in 1:(nf-1)
56           distanze[i,i]=0
57           for j in (i+1):nf
```

```
58              ppp=vec(datidist[i,:])-vec(datidist[j
                   ,:])
59              distanze[i,j]=ppp'*Q*ppp
60            end
61            distanze[(i+1):end,i]=distanze[i,(i+1):
                 end]
62          end
63          distanze[nf,nf]=0
64          VI=datiperd[fine,:verificato].==1
65          distanzeV=distanze[VI,!VI]
66          DD=datiperd[fine,:demenza][VI]
67          rho[fine[!VI]]=DataArray[rho_knn(distanzeV,
               k,DD,sum(!VI))][]
68        end
69      end
70    end
71    #le osservazioni vengono stratificate per le
         variabili categoriali e poi viene applicato il
         knn utilizzando la distanza di Mahalanobis
72    distanze=0
73    datitmp=0
74    datidist=0
75    datiperd=0
76
77
78    datitest=dati[sam,[:NACCAGE,:SEX,:race,:EDUC,:
         LIVSIT,:STROKE,:CVOTHR]]
79    matri= ModelMatrix(ModelFrame((SEX~ NACCAGE +SEX+
         race+EDUC+LIVSIT+STROKE+CVOTHR),datitest)).m
80    ps_Di=rho[ti_index]
81    ps_Dj=rho[tj_index]
```

```
82    wt=Array{Float64}[ps_Di.*(1-ps_Dj)][]
83    wt[(0:(n-1))*n+(1:n)]=0
84    notzeri=[wt.!=0]
85    i_index=ti_index[notzeri]
86    j_index=tj_index[notzeri]
87    ps_Ti=dati[sam,:MMSE][i_index]
88    ps_Tj=dati[sam,:MMSE][j_index]
89    wt=wt[notzeri]
90    nn=size(ps_Tj,1)
91    ps_I=DataFrame(I=[(ps_Ti.<ps_Tj)+1/2*(ps_Ti.==
         ps_Tj)])
92    ps_dati=[ps_I datitest[i_index,:] datitest[
         j_index,:]]
93    datitest=0
94    ps_I=0
95    fit1=glm(I ~ NACCAGE+SEX+race+EDUC+LIVSIT+STROKE+
         CVOTHR+NACCAGE_1+SEX_1+race_1+EDUC_1+LIVSIT_1+
         STROKE_1+CVOTHR_1,ps_dati, Binomial(),
         ProbitLink(), wts=wt)
96    ps_dati=0
97    theta=coef(fit1)
98    fit1=0
99    thetafin=Array(Float64,9)
100   thetafin[1]=theta[1]
101   thetafin[2:end]=[theta[i]+theta[i+8] for i in
         2:9]
102   AAUC=sum(cdf(Normal(),matri*thetafin).*rho)/sum(
         rho)
103   risultati[:,b]=[thetafin, AAUC]
104 end
105 tm=time()-t
```

```
106 print(tm)
```

Bibliografia

Adimari; Chiogna (2015a). Nearest-neighbor estimation for roc analysis under verification bias. *Int. j. Biostat.*

Adimari; Chiogna (2015b). Nonparametric verification bias-corrected inference for the area under the roc curve of a continuous-scale diagnostic test.

Alonzo T. A.; Pepe M. S. (2005). Assessing accuracy of a continuous screening test in the presence of verification bias. *Journal of the Royal Statistical Society*, **54**(1), 173–190.

Hall P.; Wilson S. R. (1991). Two guidelines for bootstrap hypothesis testing. *Biometrics*, pp. 757–762.

Janes H.; Pepe M. S. (2009). Adjusting for covariate effects on classification accuracy using the covariate-adjusted receiver operating characteristic curve.

Jeff Bezanson, Stefan Karpinski V. S. A. E. (2012). Why we created julia. [Online; in data 25-agosto-2015].

Liu D.; Zhou X.-H. (2013). Covariate adjustment in estimating the area under roc curve with partially missing gold standard. *Biometrics 69(1): 91–100.*

Masarotto G. (2009). Statistica computazionale 1. esercizi ed esempi di applicazioni. *Dispense didattiche per il corso di statistica computazionale.*

Ning J.; Cheng P. E. (2012). A comparison study of nonparametric imputation methods. *Statistics and Computing*, **22**(1), 273–285.

R Core Team (2015). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Wikipedia (2015). Julia (linguaggio di programmazione) — wikipedia, l'enciclopedia libera. [Online; in data 25-agosto-2015].