UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Scuola di Economia e Statistica

Corso di laurea in

SCIENZE STATISTICHE ED ECONOMICHE

METODI PER LA STIMA DELL'ERRORE DI PREVISIONE: UNO STUDIO COMPARATIVO.

Relatore: Prof. Matteo Borrotti

Tesi di Laurea di:

Edoardo Olivieri
Matr. N. 880280

Anno Accademico 2023/2024

Indice

1. Introduzione

1.1 BLA BLA BLA

2. Problema

Nella situazione attuale, in cui le decisioni sono sempre più supportate dai dati, l'utilizzo di modelli di previsione e classificazione è diventato indispensabile. Essi sono utilizzati in molteplici ambiti come la finanza, il marketing, la medicina e la tecnologia, dove attraverso collezioni di dati riescono ad offrire soluzioni robuste. La funzione principale di questi modelli è quella di sfruttare i dati di cui si è in possesso per fare previsioni future, classificare osservazioni in determinate categorie o identificare dei pattern. Il valore di questi modelli risiede nella loro abilità di trasformare dati grezzi in previsioni e classificazioni utili, con il fine di supportare le decisioni importanti. Ad esempio, nell'ambito della finanza, i modelli di previsione vengono utilizzati per prevedere le vendite e per valutare il rischio di bancarotta o di frode. Nel marketing, invece, vengono utilizzati per segmentare i clienti, prevedere l'abbandono e ottimizzare le strategie pubblicitarie. Le compagnie assicurative, a loro volta, utilizzano modelli di previsione per calcolare il rischio, influenzando i prezzi delle assicurazioni sulle auto, sulla vita o sugli immobili. Un altro importante ambito in cui è cresciuto molto l'utilizzo di modelli di classificazione e previsione è quello della sanità. Questi modelli, tramite l'analisi di parametri clinici o di immagini, possono aiutare a prevedere l'insorgenza di una malattia o a determinare la patologia che affligge un certo paziente.

Nonostante le enormi potenzialità di questi modelli, la loro utilità è strettamente legata alla loro accuratezza. Kuhn M. e Johnson K. (2013) definiscono la modellazione predittiva come "il processo di sviluppo di uno strumento o modello matematico che genera previsioni accurate", ponendo enfasi sulla necessità che tali previsioni siano corrette. Un modello poco accurato, infatti, può portare a decisioni errate, con possibili gravi conseguenze. Basti pensare a cosa potrebbe accadere se venisse utilizzato un modello per determinare se un paziente ha un cancro o meno, con una precisione del 40%. Valutare correttamente il modello è in altre parole di cruciale importanza. Lo strumento di eccellenza di cui si servono i ricercatori è la stima dell'errore di previsione, dato che indica con quanta sicurezza il modello farà una previsione corretta su una nuova osservazione.

2.1 Errore di previsione

L'errore di previsione ha il ruolo di misurare quanto bene un modello prevederà il valore della variabile target di una osservazione futura. Il suo utilizzo è fondamentale sia nella fase di scelta del modello che nella fase di valutazione. Sia in modelli di regressione che classificazione, l'errore di previsione è rappresentato dal valore atteso di una determinata funzione di perdita (o loss function) che indica una misura dell'errore tra la risposta e la previsione.

Una loss function viene indicata con $L(y,\hat{y})$, dove y indica il valore della risposta di una osservazione futura, mentre $\hat{y}$ indica la previsione ottenuta dal modello. La scelta del tipo di loss function da utilizzare è strettamente legata al tipo di modello che si vuole costruire. Nel caso di modelli di regressione, delle funzioni di perdita comunemente usate sono l'errore assoluto

$$L(y,\hat{y}) = |(y-\hat{y})|, \tag{2.1}$$

o l'errore quadratico

$$L(y,\hat{y}) = (y-\hat{y})^2. \tag{2.2}$$

L'errore di previsione nei casi di modelli di regressione, è rappresentato dal valore atteso della funzione di perdita utilizzata

$$PE = E(L(y,\hat{y})). \tag{2.3}$$

Per i problemi di classificazione, che saranno oggetto di studio anche nelle prossime sezioni, la risposta cade in una delle possibili K non ordinate classi, e non ha più senso considerare il concetto di distanza considerato nei problemi di regressione. Si considerano i dati come $x_i = (\underline{\beta_i}, y_i)$, $i = 1,2,...,n$, dove x_i indica la i-esima osservazione presente all'interno del dataset, $\underline{\beta_i}$ è il vettore dei predittori, mentre y_i rappresenta la classe di appartenenza. Si suppone inoltre di ottenere una nuova osservazione, che verrà indicata con $x_0 = (\underline{\beta_0}, y_0)$. La funzione di perdita avrà forma

$$L(y,\hat{y}) = I[\hat{y} \neq y], \tag{2.4}$$

A differenza dell'errore di previsione in modelli di regressione, l'errore di classificazione è definito come la probabilità di classificare incorrettamente una osservazione futura

$$PE = P(\hat{y}_0 \neq y_0), \tag{2.5}$$

chiamato anche misclassification rate.

In ambito applicativo, l'errore di previsione viene stimato. La stima dell'errore di previsione più corretta è quella ottenuta tramite il valore atteso della funzione di perdita calcolata su nuove osservazioni che non sono state utilizzate per allenare il modello

$$\text{Err} = \mathbb{E}[L(y_0, \hat{y_0})], \tag{2.6}$$

ed è chiamato errore di previsione reale (o true error). Lo stimatore più semplice è l'errore apparente (o errore di sostituzione) ed utilizza le stesse osservazioni sia per allenare il modello che per calcolarne l'errore

$$\overline{\text{err}} = \mathbb{E}[L(y_i, \hat{y}_i)] = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y}_i). \tag{2.7}$$

2.1.1 Splitting

Il motivo per cui (1.5) si calcola sulle osservazioni future è che, per calcolare l'errore di previsione reale, occorre avere delle osservazioni non precedentemente usate per allenare il classificatore. Infatti, la stima dell'errore apparente, che utilizza gli stessi dati sia per allenare il modello che per calcolarne l'errore, risulta essere distorta positivamente, soprattutto quando i dati a disposizione sono pochi. Questo avviene perché il modello è testato sugli stessi dati che ha visto durante l'allenamento, portando a una stima eccessivamente ottimistica delle sue performance.

Una semplice soluzione al problema della necessità di dati nuovi, è quella di dividere il dataset iniziale in due, assegnando un set di dati alla selezione del modello e un altro alla valutazione del modello. Questi set di dati sono comunemente chiamati training set e test set. Inoltre, il training set può essere ulteriormente suddiviso in subtraining set e validation set, così da avere osservazioni nuove anche nella fase di scelta del modello. L'idea alla base di questa tecnica chiamata splitting (suddivisione) è che le osservazioni contenute all'interno del training set sono utilizzate unicamente per allenare il classificatore, mentre le osservazioni del test set non vengono utilizzate durante la fase di training, e simulano dunque delle osservazioni future. Questo approccio permette di ottenere una stima più realistica delle performance del modello.

Tuttavia, questo metodo soffre di problematiche relative alla variabilità della stima quando il dataset è di piccole dimensioni. Quando il dataset è ridotto, la parte riservata alla valutazione può non essere rappresentativa dell'intero dominio dei dati, causando una stima inaccurata delle performance del modello. Inoltre, la variabilità tra le possibili partizioni può portare a stime dell'errore di previsione molto diverse a seconda della partizione scelta. Questo problema è amplificato nei contesti ad alta dimensionalità, dove il numero di variabili supera di gran lunga il numero di osservazioni, come spesso accade nei dati di microarray (Ambroise C. e McLachlan G.J. (2002)).

2.2 Dilemma

Ogni osservazione contiene al suo interno delle informazioni preziose. Nel caso di dataset con migliaia di osservazioni, escluderne una parte per calcolare il vero errore di previsione può risultare ragionevole. Tuttavia, quando i dati a disposizione sono pochi, si presentano diversi problemi legati alla perdita di informazioni potenzialmente utili per la fase di allenamento del modello. Questa sezione mostrerà l'impraticità e le complicazioni legate allo splitting del dataset in presenza di scarsità di osservazioni.

Il problema principale è quello legato alla perdita di informazione. Dato che ogni osservazione può contenere informazioni uniche riguardo ai pattern presenti nel dataset, l'esclusione di una parte di esse dal training set significa perdere tali informazioni. Questa esclusione potrebbe essere particolarmente impattante nei casi in cui nel dataset siano presenti osservazioni rare ma particolarmente influenti.

Infatti, se un modello viene allenato su un dataset incompleto, potrebbe non raggiungere il suo livello di accuratezza ottimale, con una conseguente scarsa abilità di generalizzare quando in presenza di dati mai osservati. Un altro rischio che si può presentare è quello di sovradattamento (o overfitting). L'overfitting si ha quando il modello si adatta troppo bene ai dati di training, riportando un alto livello di accuratezza nel training set, ma un basso livello di accuratezza in presenza di dati nuovi. Questo accade perchè il modello potrebbe scambiare delle anomalie per aspetti significativi ai fini della classificazione ed è particolarmente rilevante nei contesti ad alta

dimensionalità, come i dati genomici o di microarray, dove il numero di variabili supera di gran lunga il numero di osservazioni. In questi casi, l'esclusione di dati dal training set può ridurre significativamente la capacità del modello di generalizzare su nuovi dati, aumentando il rischio di overfitting (Cawley G.C. e Talbot N.L.C. (2010)).

Tutte le problematiche presentate vengono amplificate in casi in cui i dati a disposizione sono limitati. In questi casi ogni osservazione conterrà una percentuale di informazione più importante riguardante la struttura del dataset. Perciò escludere delle osservazioni rende più probabile il caso in cui il training set non rappresenta a pieno l'intero set di dati. Questo problema comporta anche delle considerazioni etiche e pratiche. Infatti, se un modello dovesse ritenere dei rumori all'interno del dataset come informazioni importanti, potrebbe essere di importante rilevanza in applicazioni pratiche come il calcolo del rischio, nelle assunzioni, nel sistema giudiziario, dove un modello distorto potrebbe comportare dei risultati non corretti e possibilmente discriminatori (Ferrara E. (2024)). Anche nei settori in cui i modelli devono seguire delle precise regolamentazioni, come ad esempio il settore finanziario o della sanità, i modelli allenati su dati incompleti potrebbero non agire come dai regolamenti (James M. (2024)).

Per affrontare questi problemi, sono stati sviluppati diversi metodi di ricampionamento (o resampling), come il bootstrap e la cross-validation (CV), che permettono di stimare l'errore di previsione utilizzando al meglio tutte le osservazioni disponibili. Questi metodi offrono una soluzione robusta per la stima dell'errore di previsione anche in situazioni di scarsità di osservazioni disponibili.

2.3 Metodi di ricampionamento

I metodi di ricampionamento si sono diffusi a partire dagli anni '60 grazie al rapido avanzamento della tecnologia in ambito di potenza di calcolo. Questi metodi rappresentano una valida alternativa alla statistica parametrica e un miglioramento rispetto alla statistica non parametrica. La necessità di sviluppare nuovi metodi per fare statistica inferenziale è emersa dagli evidenti limiti delle tecniche parametriche e non parametriche tradizionali. La statistica parametrica classica si basa sull'assunzione che la popolazione da cui è stato estratto il campione segua una distribuzione

Normale nota, ad eccezione del valore dei parametri θ. Tuttavia, questa assunzione non è sempre verificata, specialmente in presenza di campioni non particolarmente ampi. In questi casi, le stime ottenute possono essere inaffidabili. D'altro canto, la statistica non parametrica, prima della diffusione dei generatori di numeri casuali e della rapida crescita della potenza di calcolo, soffriva di limiti dovuti all'eccessiva o talvolta errata approssimazione dei dati.

I metodi di ricampionamento permettono di superare questi limiti ricampionando ripetutamente il set di dati a disposizione, al fine di stimare determinate caratteristiche della popolazione che ha dato origine al set di dati.

2.3.1 Plug-in principle

Il principio sulla quale si basano questi metodi è chiamato *plug-in principle*. Si basa sull'idea che per risolvere problemi di statistica inferenziale bisogna ricorrere alla stima di una distribuzione di ripartizione F sulla base di un sottocampione estratto casualmente da F. La funzione di ripartizione empirica, che chiameremo $\hat{F}$, è una stima della funzione di ripartizione originale F. La funzione $\hat{F}$ viene ottenuta costruendo una distribuzione di frequenze di tutti i valori che essa può assumere in un determinato set di dati (Efron B. e Tibshirani R.J. (1993)).

2.4 Obiettivo

L'obiettivo principale di questa tesi è condurre uno studio comparativo di diversi metodi per la stima dell'errore di previsione nel contesto dei modelli di classificazione. Lo studio mira a valutare l'efficacia e l'accuratezza dei diversi stimatori utilizzando principalmente il Root Mean Squared Error (RMSE) e il bias. Verranno applicati i diversi metodi su campioni di ampiezza diversa per valutare come la dimensione influisce sulle prestazioni e l'affidabilità degli stimatori. Inoltre, verranno utilizzate diverse tecniche di visualizzazione per illustrare la distribuzione degli errori in relazione alla dimensione del dataset. L'obiettivo si estende a valutare gli stimatori nell'applicazione di due diversi modelli di classificazione, il K-Nearest Neighbors (KNN) e gli Alberi Decisionali. Questo confronto servirà a valutare l'adeguatezza di ciascun metodo in relazione al

modello di classificazione utilizzato, così da fornire informazioni sull'adattabilità e l'efficacia dei metodi di stima in vari scenari.

La componente pratica dello studio comporta l'implementazione di tali metodi di stima su campioni di diverse dimensioni estratti da un unico dataset e la loro applicazione ai modelli KNN e Alberi Decisionali. In questo modo verrà mostrata la loro utilità in scenari in cui sia la disponibilità di dati che il modello da utilizzare possono variare.

3. Stato dell'arte

In questo capitolo viene descritto lo stato dell'arte dei metodi principali che vengono utilizzati per la stima dell'errore di previsione.

3.1 Cross Validation

Una delle prime proposte per uno stimatore che massimizzasse l'utilizzo delle informazioni disponibili, evitando di escludere troppe osservazioni per formare un test set ma che fosse anche unbiased, è stato il metodo leave-one-out (LOO). Questo metodo prevede di escludere una sola osservazione dal training set, sul quale il modello viene poi valutato. Tuttavia, questo metodo introduce una grande variabilità nella stima dell'errore di previsione, poiché il modello può ottenere un'accuratezza stimata o del 100% o dello 0% a seconda dell'osservazione esclusa.

Per risolvere questo problema, Lachenbruch P.A. e Mickey M.R. (1968) hanno proposto una tecnica oggi conosciuta come leave-one-out cross-validation (LOOCV). Questa tecnica consiste nel ripetere il metodo leave-one-out per tutte le osservazioni nel dataset, calcolando poi la media degli errori ottenuti per stimare l'errore di previsione finale. Stone M. (1974) ha ulteriormente sviluppato il concetto di cross-validation, discutendone l'uso per la valutazione delle previsioni statistiche e ponendo importanti basi per studi successivi. Per diminuire la potenza di calcolo richiesta dalla LOOCV, viene introdotta la k-fold cross-validation, che consiste nel dividere il dataset non in n parti come nel caso della LOOCV ma in k parti uguali. Questo metodo bilancia meglio il bias e la varianza rispetto al leave-one-out, riducendo la varianza senza aumentare significativemente il bias.

Wong T.T. (2015) valuta la performance di algoritmi di classificazione utilizzando queste due tecniche di cross-validation, evidenziando i vantaggi e gli svantaggi di ciascuna tecnica a seconda dello scenario applicativo. Breiman L. e Spector P. (1992) conducono uno studio sulla selezione di variabili, confrontando la leave-one-out cross-validation e la k-fold cross-validation con diversi valori di k, ottenendo risultati in favore della 10-fold cross-validation rispetto alla leave-

one-out, dato che la prima offre un ottimo bilanciamento tra bias e varianza, mentre la seconda potrebbe presentare un basso bias ma a costo di un'alta varianza. Arlot S. e Celisse A. (2010) hanno condotto una revisione delle procedure di cross-validation per la selezione del modello, evidenziando i punti di forza e le debolezze di ciascun metodo. Il loro lavoro è fondamentale per comprendere l'evoluzione delle tecniche di cross-validation e le loro applicazioni pratiche.

3.2 Bootstrap

Efron B. (1983) introduce le tecniche basate sul bootstrap per migliorare l'accuratezza della stima dell'errore di previsione, ponendo le basi per sviluppi successivi nell'ambito dei metodi di ricampionamento. Il principale obiettivo dei metodi proposti da Efron è ridurre il bias ottimista associato all'errore apparente. Tra gli stimatori presentati vi sono il bootstrap semplice (BOOT), il leave-one-out bootstrap (LOOB), il doppio bootstrap e il bootstrap .632. Quest'ultimo, in particolare, combina la stima ottenuta dal bootstrap semplice con l'errore apparente, assegnando un peso di 0.632 al primo e 0.368 al secondo, diminuendo così il bias ottimista dell'errore apparente. Successivamente, Efron B. e Tibshirani R.J. (1997) propongono il bootstrap .632+, che introduce il concetto di no-information error rate. Questo metodo assegna i pesi ai due errori in base al livello di overfitting, dando maggiore importanza all'errore apparente quando l'overfitting è minore.

I libri di Efron B. e Tibshirani R.J. (1993) e di Chernick M.R. (2007) rappresentano una guida completa sui metodi basati sul bootstrap, offrendo una panoramica dettagliata delle varie tecniche e delle loro applicazioni. Jiang W. e Simon R. (2007) confrontano diversi stimatori bootstrap per la stima dell'errore di previsione nella classificazione microarray, introducendo due nuovi stimatori: il repeated-leave-one-out bootstrap (RLOOB) e l'adjusted bootstrap (ABS), volti a correggere il bias degli stimatori bootstrap. Jiang W. e Chen B.E. (2013) modificano il bootstrap .632+ per risolvere i problemi riscontrati nei dati microarray, dove il numero di variabili supera quello delle osservazioni. L'articolo fornisce sia spunti teorici sia linee guida pratiche per applicare la loro versione modificata del bootstrap .632+. Bischl B., Mersmann O., Trautmann H. e Weihs C. (2012) esplorano vari metodi di ricampionamento, tra cui tecniche basate sul bootstrap, per

valutare meta-modelli in computazione evolutiva. Gli autori dimostrano come queste tecniche possano migliorare l'affidabilità del modello e fornire una valutazione più accurata.

3.3 Metodi di stima alternativi

Fu W.J., Carroll R.J. e Wang S. (2005) esplorano l'utilizzo del bootstrap combinato con la cross-validation per la stima dell'errore di previsione. Il loro studio mostra che questa combinazione fornisce stime dell'errore più affidabili rispetto ai metodi tradizionali, soprattutto quando il numero di osservazioni è limitato. Van Sanden S. et al. (2012) evidenziano le possibili distorsioni nella stima dell'errore di previsione in presenza di dati ad alta dimensionalità, proponendo l'uso del bootstrap cross-validation (BCV) per mitigare tali problemi. Hefny A. e Atiya A.F. (2010) introducono un nuovo stimatore dell'errore di previsione basato sul metodo Monte Carlo, fornendo un'alternativa sia alla tradizionale cross-validation sia ai metodi basati sul bootstrap. Il loro studio, che include analisi teoriche e risultati empirici, dimostra l'efficacia del nuovo stimatore che chiamano GMCP (Combined Generative and Monte-Carlo posterior estimates).

Cawley G.C. e Talbot N.L.C. (2010) affrontano il problema dell'overfitting nella selezione del modello e la conseguente distorsione nella valutazione della performance. Varma S. e Simon R. (2006) esaminano il bias nella stima dell'errore quando si utilizza la cross-validation per la selezione del modello, sottolineando l'importanza di considerare questo aspetto per evitare stime ottimistiche delle performance del modello.

Studi comparativi come quelli di Wong T.T. (2015) e Kim J.H. (2009) hanno dimostrato che ripetere la k-fold cross-validation riduce la varianza in modo più efficiente rispetto alla semplice k-fold cross-validation. Tuttavia questa maggiore efficienza richiede uno sforzo computazionale che aumenta esponenzialmente in base al numero delle osservazioni. Mostrano inoltre come in diversi casi restituisca una stima più accurata del .632+.

3.4 Applicazioni pratiche e studi comparativi

Uno dei primi studi comparativi è stato quello di Kohavi R. (1995), che ha confrontato metodi basati su cross-validation e bootstrap per la stima dell'accuratezza e la selezione del modello. I suoi risultati sono ancora rilevanti nelle tecniche di valutazione dei modelli contemporanee. Kohavi ha mostrato che, sebbene diversi metodi possano essere efficaci, la scelta del metodo appropriato dipende dal contesto specifico e dalle caratteristiche del dataset. Nell'ambito dei dati microarray, caratterizzati dalla ristretta quantità di osservazioni disponibili e dati ad alta dimensionalità, sono stati condotti diversi studi comparativi. Ambroise C. e McLachlan G.J. (2002) si concentrano sul bias di selezione nella selezione dei geni con dati di tipo microarray gene-expression. Raccomandano la 10-fold cross-validation rispetto alla leave-one-out e, tra gli stimatori bootstrap, preferiscono il .632+, poiché questo metodo offre un miglior bilanciamento tra bias e varianza.

Molinaro A.M., Simon R. e Pfeiffer R.M. (2005) confrontano vari metodi di ricampionamento per stimare l'errore di previsione. Lo studio fornisce un'analisi dettagliata dei punti di forza e di debolezza dei vari metodi, fungendo da guida per il loro utilizzo nella bioinformatica. Gli autori sottolineano che nessun metodo è universalmente migliore, ma che la scelta dipende dalle specifiche caratteristiche del dataset e del problema. Nell'ambito dell'analisi discriminante, Glele Kakaï R.L. e Palm R. (2009) e Ikechukwu E. (2016) offrono un confronto empirico di diversi stimatori dell'errore di previsione. I primi analizzano l'analisi discriminante logistica, mentre Ikechukwu esamina l'analisi discriminante con variabili multivariate binarie. I loro studi mostrano come diversi metodi di stima possano influenzare significativamente le performance del modello, suggerendo che la scelta del metodo di stima debba essere attentamente considerata in base al contesto specifico. Borra S. e Di Ciaccio A. (2008) confrontano vari metodi, inclusi quelli basati su penalità della covarianza. Il loro studio, attraverso estensive simulazioni, fornisce una valutazione delle performance relative degli stimatori. Successivamente, Borra S. e Di Ciaccio A. (2010) confrontano anche vari stimatori per l'errore extra-sample per metodi non parametrici.

4. Metodi di stima

Sulla base dei risultati ottenuti dagli studi comparativi descritti nella sezione 3.4 , si è deciso di selezionare come oggetto di studio gli stimatori che hanno avuto le migliori prestazioni, andando a considerare anche l'aspetto dell'intensità computazionale richiesta con il fine di valutare se sia giustificata o meno. Nell'ambito delle tecniche basate sul bootstrap sono stati selezionati il bootstrap semplice, il leave-one-out bootstrap, il bootstrap .632 ed il bootstrap .632+. Per quanto concerne invece la cross-validation sono stati selezionati la leave-one-out cross-validation e la *k*-fold cross-validation. Infine, come stimatori sviluppati più recentemente sono stati selezionati il bootstrap cross-validation ed il Monte Carlo cross-validation (MCCV).

Riprendendo la notazione fornita nella sezione 2.1, verranno di seguito presentati gli stimatori oggetto di studio di questa tesi.

4.1 Cross-Validation methods

I metodi di cross-validation, come spiegato anche nel nome, si basano sulla validazione incrociata del dataset. L'idea è quella di utilizzare tutte le osservazioni presenti nel dataset per allenare il modello, creando una suddivisione delle osservazioni in gruppi, talvolta anche di dimensione 1, come nel caso della LOOCV. Di seguito verranno presentati tre stimatori che sfruttano i vantaggi della cross-validation per ottenere delle stime con un basso bias, il LOOCV, il *k*-fold cross-validation ed il Monte Carlo cross-validation.

4.1.1 Leave-One-Out Cross-Validation

Il metodo LOOCV consiste nel suddividere il dataset in n parti dove n è il numero di osservazioni. Per ogni osservazione esclusa viene allenato il modello sulle restanti osservazioni e calcolato l'errore di previsione su quella esclusa. Questo processo viene ripetuto per n volte e la stima dell'errore viene calcolata facendo la media degli errori ottenuti sulle singole osservazioni escluse.

Dunque, indicando con $\hat{y}^{-i}$ la stima ottenuta rimuovendo la i-esima osservazione, la stima dell'errore di previsione tramite leave-one-out cross-validation sarà

$$\widehat{\text{Err}}^{LOOCV} = \mathbb{E}[L(y_i, \hat{y_i}^{-i})] = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y_i}^{-i}). \tag{4.1}$$

I vantaggi dell'utilizzo di questo metodo di stima sono diversi, il più importante è quello che permette di sfruttare tutte le informazioni potenzialmente utili contenute all'interno del dataset ed al contempo ottenere una stima pressochè non distorta. Inoltre, a differenza ad esempio della k-fold, la cui stima può variare a seconda di come viene diviso il dataset, il metodo LOOCV permette di ottenere una stima unica dato che vengono utilizzate tutte le osservazioni esattamente una volta come test set. Nei punti deboli del LOOCV fanno parte l'alta variabilità della sua stima, dato che tutti i training set sono molto simili tra di loro ed un'alta sensibilità rispetto ad outlier o punti anomali.

4.1.2 k-fold Cross-Validation

La k-fold cross validation viene spesso preferita alla LOOCV per via dell'elevata richiesta di potenza di calcolo di quest'ultima. Per ottenere la stima tramite k-fold cross-validation, anzichè escludere la i-esima osservazione, si procede dividendo il dataset in k parti uguali, per poi allenare il modello su $k-1$ parti e testarlo sulla parte rimanente. Questo processo viene ripetuto k volte, ogni volta con una diversa parte come set di test, e la stima finale dell'errore di previsione è la media degli errori ottenuti.

Indicando con $k(i)$ la parte contenente l'osservazione i-esima e con $\hat{y}^{-k(i)}$ la stima ottenuta rimuovendo la $k(i)$-esima parte dal training set, la stima tramite k-fold cross-validation dell'errore di pevisione è

$$\widehat{\text{Err}}^{kCV} = \mathbb{E}[L(y_i, \hat{y_i}^{-k(i)})] = \frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y_i}^{-k(i)}). \tag{4.2}$$

La leave-one-out cross-validation si può indicare come una particolare versione della k-fold, in cui $k = n$. La k-fold cross-validation è uno dei metodi più utilizzati nell'ambito di selezione e valutazione del modello. Questo ampio utilizzo è dovuto al fatto che offre un'ottimo bilanciamento tra il bias e la variabilità della stima, pur mantenendo un basso costo computazionale. Questo

metodo comporta però due aspetti che possono far variare significativamente la stima ottenuta. Il primo aspetto è la scelta del parametro k. Infatti, la scelta di un valore di k basso significa utilizzare poche osservazioni per allenare il modello e dunque aumentare la variabilità della stima (Kohavi R. (1995)). Il secondo aspetto è il modo in cui viene diviso il dataset in k parti. Infatti, se la suddivisione non viene applicata in modo casuale, andando dunque ad ottenere k parti non omogenee, la stima ottenuta potrebbe essere distorta (Hastie T., Tibshirani R.J. e Friedman J.H. (2009)).

4.1.3 Monte Carlo Cross-Validation

A differenza della k-fold, dove il dataset viene diviso in k parti ognuna contenente osservazioni uniche e non contenute all'interno delle altre $k-1$ parti, il metodo Monte Carlo consiste nell'estrarre $m = 1,2,...,M$ campioni di ampiezza prefissata senza reinserimento, andando così a creare un numero di training set e test set potenzialmente infinito, dato che le osservazioni all'interno dei vari split possono apparire più volte all'interno dei diversi test set. L'errore di previsione viene dunque stimato facendo la media tra gli errori ottenuti da ogni split. L'idea dietro l'utilizzo del metodo Monte Carlo è quello che utilizzando la k-fold vengono analizzati solo alcuni dei possibili modi in cui il dataset può essere diviso, mentre con il Monte Carlo teoricamente si potrebbero analizzare tutte le possibili partizioni. Il metodo Monte Carlo dunque tende ad avere una minore variabilità rispetto alla k-fold (a patto che venga utilizzato un numero alto di split), ma a costo di un bias maggiore. La stima tramite MCCV dell'errore di previsione è

$$\widehat{\text{Err}}^{MCCV} = \frac{1}{M}\sum_{i=1}^{M}\sum_{i=1}^{n} L(y_{i,m}, \hat{y}_{i,m})/n. \tag{4.3}$$

Questo metodo può risultare molto pesante a livello computazionale, in quanto il numero di split viene solitamente scelto molto alto. Inoltre, Arlot S. e Celisse A. (2010) hanno mostrato come in presenza di piccoli campioni, potrebbe esserci un rischio di overfitting con l'utilizzo del MCCV. Lo studio condotto da Molinaro A.M., Simon R. e Pfeiffer R.M. (2005) mostra come in realtà il MCCV offre in alcuni casi una diminuzione nella variabilità della stima rispetto alla k-fold e nessun aumento in termini di bias.

4.2 Bootstrap methods

I metodi Bootstrap si basano sul *plug-in principle*. Per stimare la distribuzione dell'errore di previsione, si ricampiona ripetutamente con reinserimento dal dataset originale, per poi fittare il modello sui diversi campioni così ottenuti.

Il punto di forza degli stimatori basati sul bootstrap è che forniscono stime meno variabili rispetto a quelle ottenute tramite tecniche di cross-validation. Queste stime comportano in diversi casi un bias più elevato, anche se per piccoli dataset vengono ottenuti ottimi risultati.

4.2.1 Boostrap Semplice

Il bootstrap semplice consiste nell'estrarre con reinserimento dal dataset un numero B di campioni con dimensioni pari al dataset originale. Il modello viene poi allenato su tutti i campioni bootstrap e testato di volta in volta sul dataset di partenza. Una volta ottenuti quindi B errori di previsione, ne viene fatta la media, ottenendo cosi la stima stramite Bootstrap semplice dell'errore di previsione. Sia $x_i^* = (\underline{\beta_i}^*, y_i^*)$, $i = 1,2,...,n$ un campione bootstrap, la stima plug-in dell'errore di previsione è

$$\frac{1}{n}\sum_{i=1}^{n} L(y_i, \hat{y}_i^*), \tag{4.4}$$

dove $\hat{y}_i^*$ indica la previsione ottenuta dal modello allenato sul campione bootstrap. Questa espressione però è il risultato di un unico campione bootstrap, ed è dunque troppo variabile (Efron B. e Tibshirani R.J. 1993). Per ottenere una stimatore migliore occorre aumentare il numero di campioni bootstrap, così da poter calcolare l'errore medio per ogni campione, e infine fare una media delle stime ottenute da ogni campione boostrap, $b = 1,2,...,B$. La stima dell'errore di previsione ottenuta tramite il bootstrap semplice è dunque

$$\widehat{\text{Err}}^{BOOT} = \frac{1}{B}\sum_{b=1}^{B}\sum_{i=1}^{n} L(y_{i,b}, \hat{y}_{i,b}^*)/n. \tag{4.5}$$

Questo stimatore risulta avere una variabilità inferirore rispetto alla cross-validation, ma a costo di una stima troppo ottimista.

4.2.2 Leave-One-Out Bootstrap

Il leave-one-out bootstrap viene proposto da Efron B. (1983) come una versione smoothed della LOOCV. Consiste nel prevedere l'errore di una i-esima osservazione solo tramite campioni bootstrap che non la contengono. Sia $C(i)$ un set di indici dei campioni bootstrap che non contengono l'i-esima osservazione e $B(i)$ il numero di tali campioni bootstrap. La stima dell'errore di previsione tramite LOOB sarà

$$\widehat{\text{Err}}^{LOOB} = \frac{1}{n}\sum_{i=1}^{n}\sum_{b\in C(i)} L(y_{i,b}, \hat{y}^*_{i,b})/B(i). \tag{4.6}$$

Questo stimatore viene introdotto da Efron B. (1983) principalmente per costruire un altro stimatore, il bootstrap .632. Nonostante ciò, il LOOB viene utilizzato anche indipendentemente, fornendo un buon compromesso tra bias e varianza, riuscendo a diminuire la variabilità legata all'utilizzo del metodo LOO, ma a costo di un leggero bias negativo.

4.2.3 Bootstrap .632

Con l'obiettivo di correggere gli aspetti negativi del bootstrap semplice e dell'errore apparente, Efron B. (1983) propone un terzo stimatore basato sul Bootstrap, che chiama il bootstrap .632. L'idea dietro al bootstrap .632 è quella di contrastare il bias ottimista relativo all'errore apparente con il bias negativo ottenuto dal LOOB. Vengono dunque unite le due stime assegnando un peso di 0.368 al primo errore e 0.632 al secondo. Il fattore 0.632 riflette la proporzione attesa di osservazioni che si ripetono all'interno del campione bootstrap. Per arrivare a questo numero, che rappresenta la probabilità che una osservazione venga selezionata, occorre partire dalla probabilità che non venga selezionata. Durante la prima estrazione, se la probabilità di estrarre un'osservazione è $1/n$, la probabilità di non estrarla sarà $1-1/n$. Se in totale ci sono n estrazioni, tutte indipendenti tra loro e con reinserimento, la probabilità di non selezionare mai l'osservazione diventerà $(1-1/n)^n$. Il problema diventa quindi quello di calcolare il limite

$$\lim_{n\to\infty}\left(1-\frac{1}{n}\right)^n. \tag{4.7}$$

Tale limite risulta essere uguale a $\frac{1}{e} \approx 0.368$. Essendo questa la probabilità che una osservazione non venga estratta, la probabilità che venga estratta sarà $1-0.368 = 0.632$. Giustificato l'utilizzo

del termine 0.632 la stima dell'errore di previsione tramite bootstrap .632 è

$$\widehat{\text{Err}}^{.632} = 0.368 \cdot \overline{\text{err}} + 0.632 \cdot \widehat{\text{Err}}^{LOOB}. \tag{4.8}$$

Efron B. (1983) per primo mostra tramite simulazioni come in diversi scenari il bootstrap .632 risulta essere più efficiente degli stimatori basati su tecniche di cross-validation. In presenza di piccoli campioni risulta inoltre essere più efficiente del LOOB e del bootstrap semplice. Il bootstrap .632 non offre però buone stime in caso di modelli a rischio di overfitting (Breiman L., Friedman J., Olshen R.A. e Stone C.J. (1984)).

4.2.4 Bootstrap .632+

Efron B. e Tibshirani R.J. (1997) propongono una variazione del bootstrap .632 che chiamano bootstrap .632+. La necessità di migliorare il bootstrap .632 viene dai problemi che esso riscontra in presenza di modelli ad alto rischio di overfitting, come ad esempio un modello K-Nearest Neighbors con K pari a 1 oppure in presenza di alberi decisionali. In presenza di overfitting l'errore apparente è nullo, il che significa che lo stimatore bootstrap .632 tende a sottostimare l'errore di previsione, dato che tiene conto solo dell'errore $\widehat{\text{Err}}^{LOOB}$ moltiplicato per 0.632. Per risolvere questo problema, il bootstrap .632+ anzichè tenere fissi i pesi assegnati ai due errori, li determina dando un peso maggiore all'errore apparente quando l'overfitting, calcolato come $\widehat{\text{Err}}^{LOOB} - \overline{\text{err}}$, è piccolo (Efron B. e Tibshirani R.J. (1997)). Per dimensionare correttamente il livello di sovradattamento, viene calcolato il tasso relativo di overfitting tramite la seguente formula

$$\widehat{R} = \frac{\widehat{\text{Err}}^{LOOB} - \overline{\text{err}}}{\widehat{\gamma} - \overline{\text{err}}}, \tag{4.9}$$

dove $\widehat{\gamma}$ è il no-information error rate, che viene invece stimato tramite la seguente formula

$$\widehat{\gamma} = \sum_{l} \hat{p}_l (1 - \hat{q}_l), \tag{4.10}$$

dove $\hat{p}_l$ indica la proporzione della classe l all'interno del dataset mentre $\hat{q}_l$ indica la proporzione stimata della classe l. Il prodotto tra il primo termine $\hat{p}_l$ ed il secondo termine $1 - \hat{q}_l$, che indica la proporzione di osservazioni che il classificatore non prevede come appartenenti alla classe l, indica

la proporzione di osservazioni della classe l che non sono state classificate come appartenenti ad essa. Sommando questo prodotto per tutte le classi presenti nel dataset si ottiene $\widehat{\gamma}$, che indica la proporzione di osservazioni misclassificate da un classificatore che prevede l'appartenenza alle classi in base alle distribuzioni delle classi $\hat{q}_l$. Una volta ottenuti il tasso relativo di overfitting e il no-information rate vengono costruiti i presi che verranno assegnati ai due errori con la seguente formula

$$\widehat{\omega} = \frac{0.632}{1 - 0.368\widehat{R}}. \tag{4.11}$$

Infine, la stima dell'errore di previsione tramite bootstrap .632+ sarà

$$\widehat{\text{Err}}^{.632} = (1 - \widehat{\omega}) \cdot \overline{\text{err}} + \widehat{\omega} \cdot \widehat{\text{Err}}^{LOOB}. \tag{4.12}$$

Il valore del peso ω può variare da 0.632 nel caso in cui sia assente l'overfitting ($\widehat{R} = 0$) ad 1 nel caso in cui sia presente overfitting ($\widehat{R} = 1$). Il bootstrap .632+ ha mostrato ottimi risultati in diversi studi comparativi come ad esempio quello di Molinaro A.M., Simon R. e Pfeiffer R.M. (2005) dove in presenza di piccoli campioni il bootstrap .632+ risulta essere tra i più accurati. Borra S. e Di Ciaccio A. (2008) mostrano come in presenza di alti livelli di overfitting il bootstrap .632+ risulta essere superiore rispetto al bootstrap .632. Sempre Borra S. e Di Ciaccio A. (2010) mostrano però come in presenza di un alto rapporto segnale-rumore il bootstrap .632+ performa peggio di altri stimatori.

4.3 Bootstrap Cross-Validation

Fu W.J., Carroll R.J. e Wang S. (2005), spinti dai successi del bootstrap in presenza di piccoli campioni, come dimostrato da Efron B. (1983) e da Efron B. e Tibshirani R.J. (1997), propongono di combinare l'utilizzo del bootstrap di Efron B. e Tibshirani R.J. (1993) con la cross-validation di Geisser S. (1975) e Stone M. (1974) per creare un nuovo stimatore che chiamano bootstrap cross-validation (BCV). L'idea è quella di ridurre la variabilità legata alla cross-validation quando in presenza di poche osservazioni con l'implementazione del bootstrap.

Per calcolare la stima dell'errore di previsione tramite BCV si inizia estraendo con reinserimento dal dataset un numero B di campioni bootstrap, per poi andare a calcolare su ognuno

di essi l'errore tramite k-fold cross-validation. Una volta ottenute le stime per i B campioni bootstrap viene fatta la media per ottenere la stima finale dell'errore di previsione. Indicando con r_b l'errore ottenuto tramite cross-validation (Fu W.J., Carroll R.J. e Wang S. (2005)) per i diversi $b = 1, 2, ..., B$ campioni bootstrap, la stima dell'errore tramite BCV sarà

$$\widehat{\text{Err}}^{BCV} = \frac{1}{B} \sum_{b=1}^{B} r_b, \tag{4.13}$$

Fu, Carroll e Wang utilizzano il mean square relative error (MSRE) come principale criterio di confronto tra diversi stimatori ed ottengono risultati a favore del BCV nella maggior parte degli scenari da loro analizzati. In particolare mostrano come anche con campioni molto ristretti, cioè con 16 (o meno) osservazioni, le stime tramite BCV riescono comunque ad essere accurate. Un limite però del bootstrap cross-validation è che può risultare molto intensivo computazionalmente in casi in cui i campioni non sono piccoli.

5. Simulazione

In questo capitolo si procede con lo studio comparativo, svolto su un dataset riguardante malattie cardiovascolari. Il dataset scelto è composto dall'unione di cinque dataset: Cleveland, Hungary, Switzerland, Long Beach VA e Statlog, contenenti rispettivamente 303, 294, 123, 200 e 270 osservazioni, per un totale di 1190 osservazioni e può essere scaricato al seguente link: https://www.kaggle.com/datasets/sid321axn/heart-statlog-cleveland-hungary-final/data. L'unione di questi dataset fornisce una ricca fonte di informazioni per analizzare le malattie cardiovascolari tramite l'identificazione dei fattori di rischio. Il dataset contiene 12 variabili, di cui si fornisce di seguito una breve spiegazione.

- Age: età del paziente.
- Sex: genere del paziente (Femmina (0) , Maschio (1)).
- Chest Pain Type: tipo del dolore al petto categorizzato in: Tipico (1), Angina tipica (2), Dolore non anginoso (3) ed Asintomatico (4).
- Resting bp/s: livello di pressione del sangue a riposo espresso in mm/HG
- Cholosterol: colesterolo sierico in mg/dl
- Fasting Blood Sugar: livello di glicemia a digiuno (valore 1 se $>$ 120 mg/dl e 0 altrimenti)
- Resting ecg: risultato dell'elettrocardiogramma quando a riposo (Normale (0), Anomalie nell'onda ST-T (1), Ipertrofia ventricolare sinistra (2))
- Max Heart Rate: frequenza cardiaca massima raggiunta
- Exercise Angina: angina indotta dall'esercizio fisico (No (0), Si (1))
- Oldpeak: depressione del tratto ST indotta dall'esercizio fisico rispetto allo stato di riposo.

- ST slope: segmento ST misurato in termini di pendenza durante l'esercizio fisico massimo (Normale (0), In salita (1), Piatta (2), In discesa (3)).

- target: variabile considerata come risposta e assume valore 1 se il paziente è affetto da malattie cardiovascolare e 0 altrimenti.

Il dataset è stato pre-processato per ottenere risultati accurati dai classificatori. Si è applicata una normalizzazione del dataset in modo da portare tutte le variabili numeriche nel range 0-1. I due modelli di classificazione presi in considerazione sono il KNN con $K = 9$ ed il modello ad alberi decisionali. Per quanto riguarda lo studio della dimensione del dataset, sono stati presi in considerazioni campioni di 50, 200 e 500 osservazioni.

5.1 Scelta dei parametri degli stimatori

Per quanto riguarda gli stimatori basati sul ricampionamento bootstrap, si è scelto di utilizzare un numero di campioni B pari a 50, dato che considerare un numero maggiore non comporta necessariamente un miglioramento nella stima (Efron B. e Tibshirani R.J. (1997)). Per quanto riguarda la k-fold cross-validation è stato scelto un k pari a 10, dati i risultati di precedenti studi come quelli di Breiman L. e Spector P. (1992) e Molinaro A.M., Simon R. e Pfeiffer R.M. (2005). Infine, per il Monte Carlo cross-validation è stato scelto un numero M di split pari a 50 per equilibrare il numero di campioni bootstrap. In questo modo i diversi metodi di stima richiedono circa la stessa potenza di calcolo, così da non ottenere stime influenzate dalla capacità computazionale.

5.2 Algoritmi

In questa sezione verranno brevemente descritti gli algoritmi di classificazione utilizzati nello studio comparativo. Gli algoritmi scelti sono il K-Nearest Neighbors e gli Alberi Decisionali. Questi due algoritmi sono stati selezionati, oltre che per il loro diffuso utilizzo in applicazioni pratiche, per le loro contrastanti caratteristiche in termini di rischio di overfitting. Il KNN è noto per il suo basso rischio di sovradattamento. Questa caratteristica proviene in gran parte dalla sua

natura non parametrica, dove la complessità del modello può essere regolata attraverso la scelta dei K vicini più prossimi. Inoltre, il KNN non fa forti assunzioni sulla forma dei confini decisionali, permettendogli di adattarsi bene a diversi tipi di dataset.

D'altro canto, gli alberi decisionali hanno intrinsecamente un alto livello di overfitting, in particolare quando vengono lasciati crescere in profondità senza restrizioni. Questa propensione all'overfitting viene dall'abilità del modello di suddividere il campione in sottocampioni sempre più piccoli, adattando regole decisionali complesse che catturano persinno il rumore nei dati di training. Sebbene questa flessibilità permetta agli alberi decisionari di modellare relazioni complesse all'interno del campione, essa indica anche che il modello può facilmente diventare troppo complesso, catturando dettagli specifici dei dati usati per allenare il modello che non si generalizzano bene a dati nuovi.

Questo studio comparativo mira a esplorare gli effetti che i due diversi modelli di classificazione, uno più stabile e meno complesso come il KNN e uno più flessibile ma a rischio di overfitting come gli alberi decisionali, possono avere sulle prestazioni degli stimatori.

5.2.1 K-Nearest Neighbors

Il K-Nearest Neighbors è un algoritmo non parametrico supervisionato, che viene utilizzato sia per problemi di classificazione che di regressione. La sua semplicità ed intuitività lo rendono uno strumento popolare per molte applicazioni pratiche. Il KNN si basa sul principio che le osservazioni simili nello spazio delle variabili dovrebbero avere valori target simili. Questo concetto si basa sull'assunzione che esiste una relazione di continuità tra le variabili indipendenti e la variabile dipendente. La scelta del parametro K, il numero di vicini più prossimi da considerare, è cruciale per le prestazioni dell'algoritmo. Un valore di K troppo piccolo può rendere il modello sensibile al rumore nei dati, mentre un valore di K troppo grande può diluire l'effetto delle osservazioni prossime. Nel nostro caso, dopo test preliminari sui diversi campioni, è stato scelto un valore di K pari a 9, il che significa che l'algoritmo considera i 9 vicini più prossimi per fare previsioni.

La metrica di distanza tipicamente utilizzata nel KNN è la distanza euclidea, definita come la radice quadrata della somma dei quadrati delle differenze tra le coordinate corrispondenti

di due punti. Tuttavia, altre metriche come la distanza di Manhattan o la distanza di Minkowski possono essere impiegate a seconda delle esigenze specifiche dell'applicazione. Per classificare una nuova osservazione, l'algoritmo KNN calcola la distanza tra la nuova osservazione e tutte le altre osservazioni nel dataset di addestramento, identifica i K vicini più prossimi e determina la classe della nuova osservazione in base alla classe più frequente tra i K vicini più prossimi.

La libreria `class` in `R` fornisce la funzione `knn()` che viene utilizzata per implementare l'algoritmo. Questo pacchetto permette una ricerca e classificazione efficiente dei vicini più prossimi.

5.2.2 Alberi Decisionali

Gli Alberi Decisionali sono un tipo di algoritmo supervisionato, non parametrico e facilmente interpretabile. Sono utilizzati per la classificazione e la regressione e funzionano suddividendo iterativamente lo spazio dei predittori in regioni distinte e non sovrapposte. Gli alberi decisionali segmentano lo spazio delle variabili in una serie di regioni omogenee per quanto riguarda la variabile target. La tecnica usata per costruire gli alberi è quella della suddivisione binaria ricorsiva, approccio di tipo top-down, in cui l'insieme dei dati viene successivamente suddiviso in base a variabili predittive.

Il processo di costruzione inizia con la suddivisione iniziale, dove tutti i dati iniziano in un singolo nodo. Si valuta ogni possibile split delle variabili predittive per identificare quello che massimizza una funzione obiettivo, come la riduzione dell'indice di Gini o l'aumento dell'informazione. Il nodo viene poi suddiviso nei due sottogruppi risultanti dalla suddivisione scelta, creando così i nodi interni. Questo processo viene ripetuto ricorsivamente per ciascun sottogruppo, creando ulteriori suddivisioni fino a raggiungere un criterio di arresto, come una profondità massima dell'albero o un numero minimo di osservazioni per nodo. I nodi che non vengono ulteriormente suddivisi sono detti nodi terminali o leaf nodes.

Negli alberi di classificazione, ogni suddivisione cerca di portare alla massima riduzione dell'indice di Gini, una misura della purezza dei nodi. L'indice di Gini è definito come il complemento a 1 della somma dei quadrati delle proporzioni delle osservazioni che appartengono a

ciascuna classe nel nodo. Un altro criterio usato è la riduzione dell'entropia, che misura l'informazione guadagnata con una suddivisione. La libreria `tree` in `R` offre la funzione `tree()` che viene utilizzata per creare i modelli ad alberi decisionali.

5.3 Metriche di confronto

Il confronto tra gli stimatori viene effettuato tramite un numero R di iterazioni in modo tale da poter confrontare il comportamento per quanto riguarda sia l'accuratezza che la variabilità dei diversi stimatori. Le due metriche di confronto considerate sono l'RMSE calcolato come radice dell'MSE, che viene a sua volta calcolato come:

$$MSE = \frac{1}{r}\sum_{r=1}^{R}\left(\hat{\theta}_r - \theta_r\right)^2, \tag{5.1}$$

ed il Bias, calcolato come:

$$Bias = \frac{1}{r}\sum_{r=1}^{R}\left(\hat{\theta}_r - \theta_r\right), \tag{5.2}$$

dove r indica l'r-esima iterazione, $\hat{\theta}_r$ indica la stima dell'errore di previsione e θ_r indica il vero errore di previsione, calcolato sulle osservazioni non considerate all'interno del campione (Molinaro A.M., Simon R. e Pfeiffer R.M. (2005)). La scelta di utilizzare l'RMSE al posto dell'MSE, ripresa da Efron B. e Tibshirani R.J. (1997), viene da una ricerca della miglior interpretabilità dei risultati possibile. A tal proposito l'RMSE, riportando gli errori nella stessa scala dei dati originali, rende più facilmente interpretabili i risultati, contribuendo anche ad una miglior comparabilità tra diversi modelli e campioni. Inoltre, penalizza maggiormente gli errori più grandi rispetto all'MSE.

5.4 Risultati

Di seguito vengono riportati i risultati ottenuti sul dataset riguardante malattie cardiovascolari. Prima vengono mostrati i risultati relativi al modello 9NN e successivamente quelli relativi al modello ad alberi decisionali.

5.4.1 9NN

Con riferimento alla Tabella 5.1, per $n = 50$ i tre stimatori migliori in termini di RMSE sono il BCV (0.0516), il BOOT (0.0517) ed il .632 (0.0561). Anche in termini di bias, il BCV (-0.0018) ed il BOOT (-0.0041), hanno i valori più bassi. Anche la LOOCV ha un bias basso pari a -0.0025. Per $n = 200$, il BCV, il BOOT ed il .632 mostrano ancora i risultati migliori per quanto riguarda l'RMSE, con valori rispettivamente pari a 0.0241, 0.0241 e 0.0250. Il .632, con bias pari a 0.0250 rientra anche nei miglior stimatori in termini di bias, assieme alla 10-fold CV (0.0028) e LOOCV (0.0041). Per $n = 500$, il .632, il .632+ ed il MCCV mostrano il miglior risultato in termini di RMSE, tutti con un valore pari a 0.0184. Per il bias, i migliori stimatori risultano essere il .632+ (-0.0011), il MCCV (-0.0008) e la 10-fold CV (-0.0002).

Tabella 5.1: RMSE e bias per i metodi di stima dell'errore

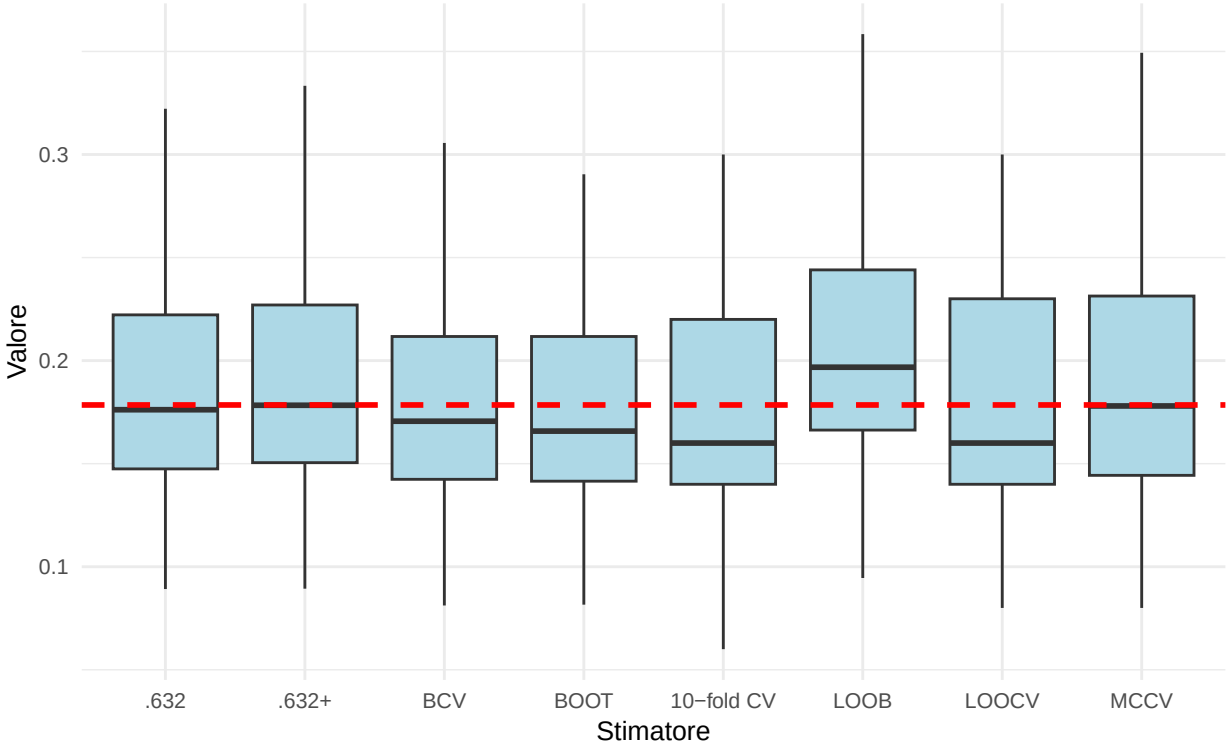
Stimatore	$n = 50$		$n = 200$		$n = 500$	
	RMSE	Bias	RMSE	Bias	RMSE	Bias
BOOT	**0.0517**	**–0.0041**	**0.0241**	–0.0065	0.0218	–0.0124
LOOB	0.0675	0.0272	0.0340	0.0200	0.0226	0.0125
.632	**0.0561**	0.0069	**0.0250**	**0.0039**	**0.0184**	–0.0023
.632+	0.0586	0.0101	0.0258	0.0055	**0.0184**	**–0.0011**
LOOCV	0.0596	**–0.0025**	0.0275	**0.0041**	0.0197	–0.0029
10-fold CV	0.0579	–0.0065	0.0252	**0.0028**	0.0203	**–0.0002**
MCCV	0.0605	0.0066	0.0258	0.0049	**0.0184**	**–0.0008**
BCV	**0.0516**	**–0.0018**	**0.0241**	–0.0047	0.0207	–0.0115

Confronto tra RMSE e bias degli stimatori. I risultati sono ottenuti sul modello 9NN, considerando campioni di ampiezza 50, 200 e 500. Per ogni colonna sono stati evidenziati i tre risultati migliori.

All'aumentare dell'ampiezza del campione da 50 a 500 è presente un miglioramento generale in termini di RMSE per tutti gli stimatori. Questo non è insolito, dato che campioni più grandi forniscono di solito stime più stabili e accurate. Il .632 ottiene valori bassi di RMSE su tutte

le dimensioni del campione, indicando un'alta accuratezza e robustezza in condizioni diverse. La LOOCV e la 10-fold CV mostrano con costanza un basso bias, rendendo le stime affidabili per minimizzare gli errori sistematici. La LOOCV risulta avere un bias più basso rispetto alla 10-fold in presenza di piccoli campioni, ma più grande nel campione $n = 200$ e $n = 500$. Questo basso bias della LOOCV comporta però una maggiore varianza rispetto alla 10-fold, che risulta essere migliore rispetto alla LOOCV su tutte le dimensioni del campione. Il LOOB mostra un bias più grande in presenza di piccoli campioni, indicando una potenziale sensibilità alla dimensione del dataset. Se per i campioni più piccoli i migliori stimatori risultano essere il BCV ed il .632, per il campione $n = 500$ i migliori stimatori sono il MCCV ed il .632+, entrambi con un RMSE pari a 0.0184 ed un bias molto basso. La grande differenza di RMSE tra i diversi campioni del MCCV mostra come esso sia molto sensibile dalla dimensione del dataset.

Figura 5.1: Boxplot per $n = 50$ dei metodi di stima dell'errore

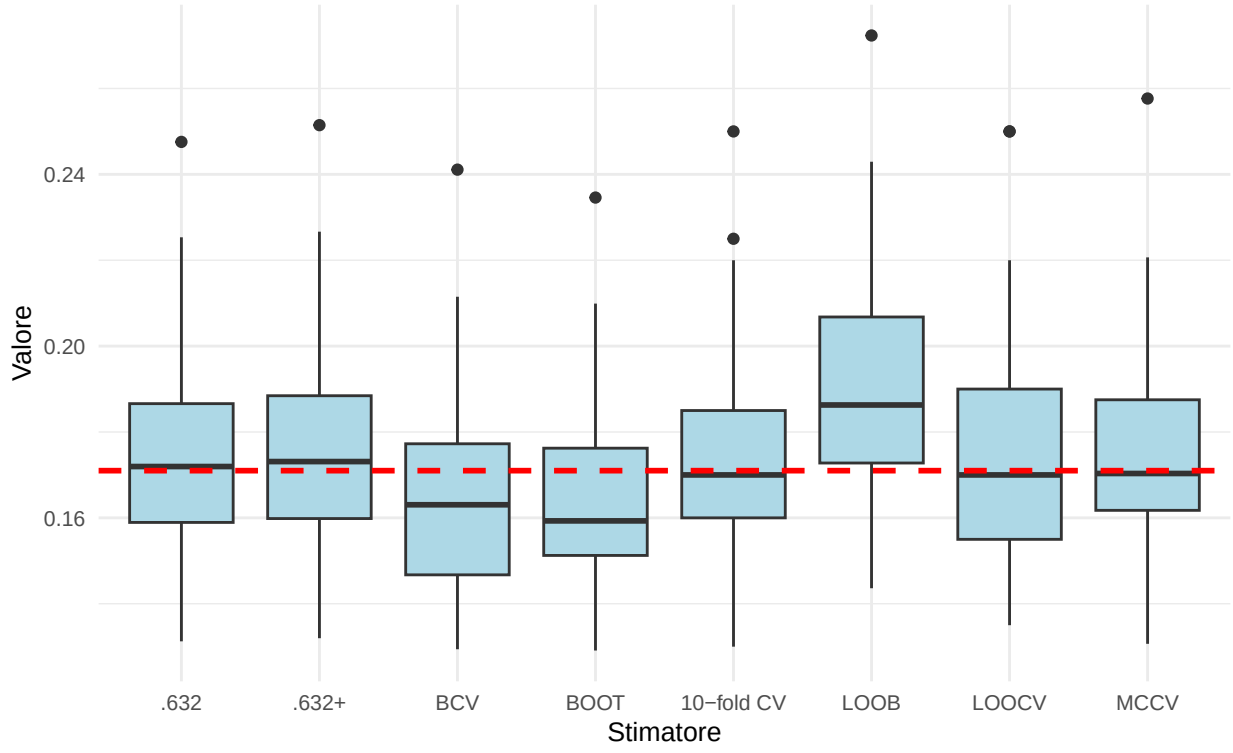


I risultati mostrati sono ottenuti sul modello 9NN con $n = 50$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Analizzando i boxplot della Figura 5.1 ottenuti sul campione con $n = 50$,si può subito notare come le mediane della maggior parte degli stimatori sono vicine al vero errore di previsione, rappresentato dalla linea rossa tratteggiata, suggerendo che la tendenza centrale di questi stimatori è ben allineata con l'errore reale. In particolare, gli stimatori .632, .632+, BCV e BOOT mostrano intervalli interquantili più stretti, indicando minore variabilità nelle stime dell'errore, e di conseguenza, una prestazione più costante. Al contrario, il LOOB, la LOOCV mostrano intervalli interquantili più ampi, significando una maggiore variabilità e una minore affidabilità nelle loro stime. I baffi dei boxplot si estendono fino al valore massimo e minimo, esclusi gli outlier, e mostrano come il LOOB e il MCCV abbiano un intervallo più ampio di stime dell'errore, suggerendo la produzione occasionale di valori estremi. La posizione delle mediane rispetto all'errore reale

mostra come il .632, .632+, BCV e MCCV hanno un bias basso. Nonostante la vicinanza delle medie del MCCV e della LOOCV all'errore reale, la loro grande dispersione riflette un compromesso tra bias e varianza peggiore rispetto agli altri stimatori. Complessivamente gli stimatori .632 e .632+ mostrano una performance bilanciata con bassa variabilità ed una vicinanza della mediana all'errore reale, significando accuratezza e affidabilità nella gestione del bias.

Figura 5.2: Boxplot per $n = 200$ dei metodi di stima dell'errore

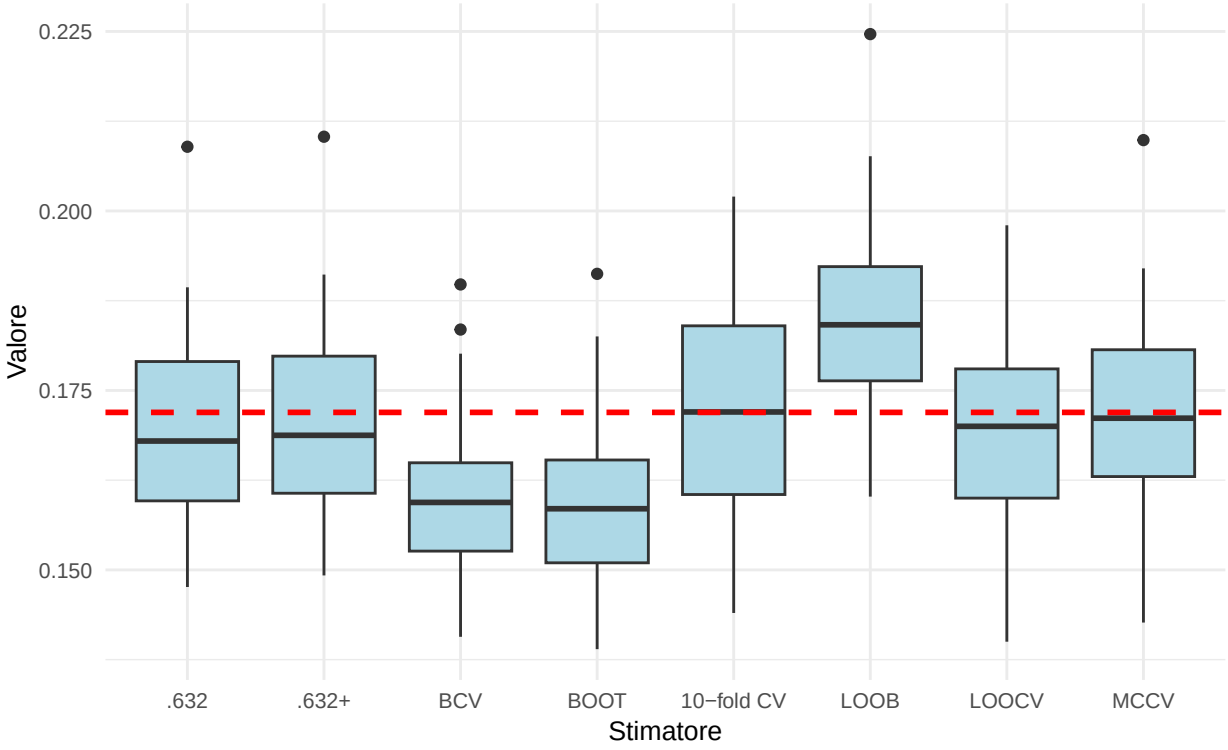


I risultati mostrati sono ottenuti sul modello 9NN con $n = 200$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Confrontando la Figura 5.1 e la Figura 5.2, si nota come la variabilità nelle stime dell'errore è generalmente più bassa nel campione con $n = 200$. Questa minore variabilità viene mostrata da degli intervalli interquantili più piccoli rispetto al campione con $n = 50$. Il BOOT ed il BCV sembrano essere i migliori in termini di RMSE nel campione $n = 200$ (Tabella 5.1) e non molto inferiori agli altri in termini di bias. Dal boxplot si può però notare come la differenza di bias

con gli altri stimatori non sia da ritenere minima, in quanto i due intervalli interquantili sono sì molto piccoli, ma anche molto spostati verso il basso rispetto all'errore vero, da cui viene infatti il valore negativo del bias dei due stimatori. In modo analogo, anche il LOOB ha un intervallo interquantile molto spostato rispetto al'errore vero, ed assume infatti un valore di bias maggiore del 200% rispetto a quello degli altri stimatori (Tabella 5.1). Nei boxplot con $n = 50$ gli stimatori LOOB e LOOCV mostrano una grande variabilità e una maggiore presenza di outlier. Nei boxplot con $n = 200$, sebbene questi stimatori mostrino ancora una variabilità maggiore rispetto agli altri, la dispersione è meno pronunciata e ci sono meno outlier. Questo suggerisce che un aumento nella dimensione del campione aiuta a ridurre la varianza e la frequenza di valori estremi per questi due stimatori. In entrambi i campioni le prestazioni di ciascun stimatore in termini di bias e varianza rimangono relativamente costanti, ma la dimensione maggiore del campione fornisce una distinzione più chiara tra gli stimatori più affidabili, come il BOOT, il BCV, il .632, il.632+ e la 10-fold CV e quelli con maggiore variabilità, come il LOOB e la LOOCV.

Figura 5.3: Boxplot per $n = 500$ dei metodi di stima dell'errore



I risultati mostrati sono ottenuti sul modello 9NN con $n = 500$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Nei boxplot della Figura 5.3 gli stimatori mostrano ancora meno variabilità rispetto alla 5.2, nonostante la differenza sia minore rispetto a quelli tra la Figura 5.2 e la Figura 5.1. In tutte e tre le figure, il BOOT e BCV mostrano la variabilità minore, anche se con l'aumentare della dimensione del campione aumenta anche il bias negativo dei due stimatori. Invece, il .632, il .632+ e la 10-fold CV mostrano un basso bias in tutti i casi, rendendoli i più affidabili per la minimizzazione degli errori sistematici. Il LOOB nel campione con $n = 500$ risulta avere una variabilità più in linea con quella degli altri stimatori, a differenza degli altri due campioni. Questo conferma ulteriormente che l'aumentare della dimensione campionaria aiuta a ridurre la varianza di tale stimatore. La minore variabilità però è stata compensata con un bias ancora più alto. Mentre metodi come il BCV ed il BOOT mostrano prestazioni stabili con bassa variabilità, metodi come il LOOB

e la 10-fold CV dimostrano una maggiore variabilità, specialmente con dimensioni del campione ridotte. La presenza di valori anomali aumenta anche con la diminuzione della dimensione del campione, evidenziando l'importanza di considerare la dimensione del dataset nelle valutazioni delle prestazioni degli stimatori.

5.4.2 TREE

La tabella 5.2 mostra i risultati per il modello TREE. Notiamo che il metodo BOOT ha un RMSE inizialmente molto elevato per $n = 50$, con un valore pari a 0.1123, che diminuisce man mano che aumenta la dimensione del campione, raggiungendo 0.0586 per $n = 500$. Inoltre, il bias è negativo in tutti i casi, con un valore più alto in termini assoluti per i campioni più piccoli. Il metodo LOOB presenta un RMSE relativamente basso, con 0.0709 per $n = 50$, 0.0448 per $n = 200$ e 0.0287 per $n = 500$. Il Bias di LOOB è positivo per tutte le dimensioni del campione e, seppur più elevato rispetto agli altri stimatori, rimane relativamente stabile, suggerendo una buona affidabilità nel bias. Il .632+ si distingue con i valori di RMSE più bassi tra tutti i metodi considerati, particolarmente per $n = 200$, con valore pari a 0.0355 e $n = 500$, con valore pari a 0.0246. Anche il bias per il .632+ è tra i migliori, indicando un'accuratezza superiore rispetto ad altri stimatori. I metodi LOOCV e 10-fold CV mostrano valori di RMSE moderati ma bias molto bassi o prossimi allo zero. Il metodo MCCV evidenzia un RMSE molto basso per tutte le dimensioni del campione e un Bias positivo per i campioni più grandi ma vicino allo zero per $n = 50$. Infine, il metodo BCV mostra un RMSE relativamente alto e un Bias negativo, simile al metodo BOOT, suggerendo che questi metodi potrebbero non essere ideali per il modello ad alberi decisionali.

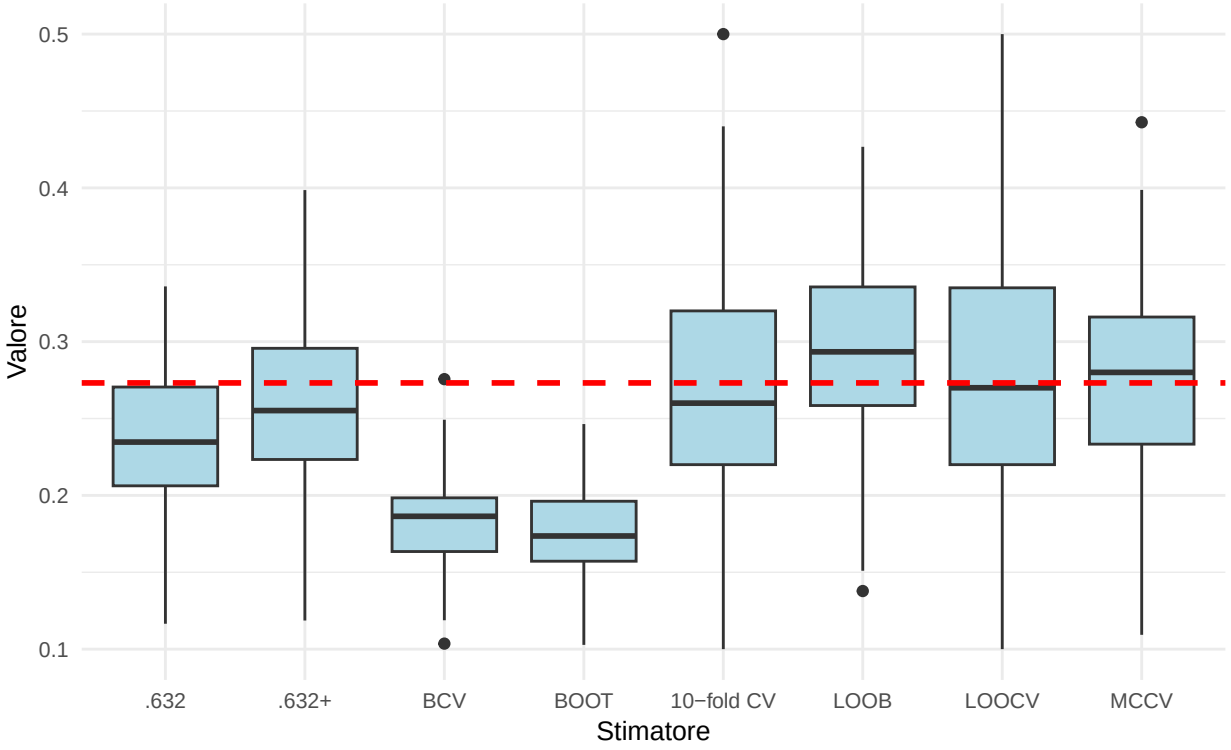
Tabella 5.2: RMSE e bias per i metodi di stima dell'errore

Stimatore	$n = 50$		$n = 200$		$n = 500$	
	RMSE	Bias	RMSE	Bias	RMSE	Bias
BOOT	0.1123	–0.0979	0.0737	–0.0689	0.0586	–0.0551
LOOB	**0.0709**	0.0200	0.0448	0.0277	0.0287	0.0165
.632	0.0733	–0.0410	**0.0393**	–0.0235	**0.0266**	–0.0146
.632+	**0.0695**	–0.0163	**0.0355**	**–0.0101**	**0.0246**	**–0.0097**
LOOCV	0.0872	**–0.0024**	0.0507	**0.0064**	0.0379	**0.0080**
10-fold CV	0.0841	**–0.0068**	0.0422	**0.0106**	0.0308	**0.0057**
MCCV	**0.0694**	**0.0023**	**0.0389**	0.0160	**0.0253**	0.0106
BCV	0.1056	–0.0898	0.0664	–0.0609	0.0539	–0.0501

Confronto tra RMSE e bias degli stimatori. I risultati sono ottenuti sul modello TREE, considerando campioni di ampiezza 50, 200 e 500. Per ogni colonna sono stati evidenziati i tre risultati migliori.

Confrontando i due set di risultati mostrati in Figura 5.1 e Figura 5.2, notiamo che i metodi .632 e .632+ tendono a fornire i migliori risultati in termini di RMSE e Bias per entrambi i modelli TREE e 9NN. Questi metodi mostrano un'ottima capacità di generalizzazione, con bias molto vicino allo zero e RMSE consistentemente bassi. Il metodo BOOT, mostra un RMSE più elevato per il modello ad alberi decisionali rispetto al modello 9NN, suggerendo che la sua efficacia può dipendere fortemente dal modello specifico utilizzato. I metodi LOOCV e 10-fold CV mostrano una buona affidabilità complessiva, con RMSE moderati e bias molto bassi, suggerendo che sono metodi robusti per una varietà di modelli. Il metodo MCCV emerge come uno dei più affidabili, con RMSE e Bias contenuti in entrambi i modelli, rendendolo una scelta versatile per diverse applicazioni. Infine, il metodo BCV, pur non eccellendo in modo particolare nel modello ad alberi decisionali, mostra risultati ottimi nel modello 9NN in presenza di piccoli campioni, rendendolo una valida opzione.

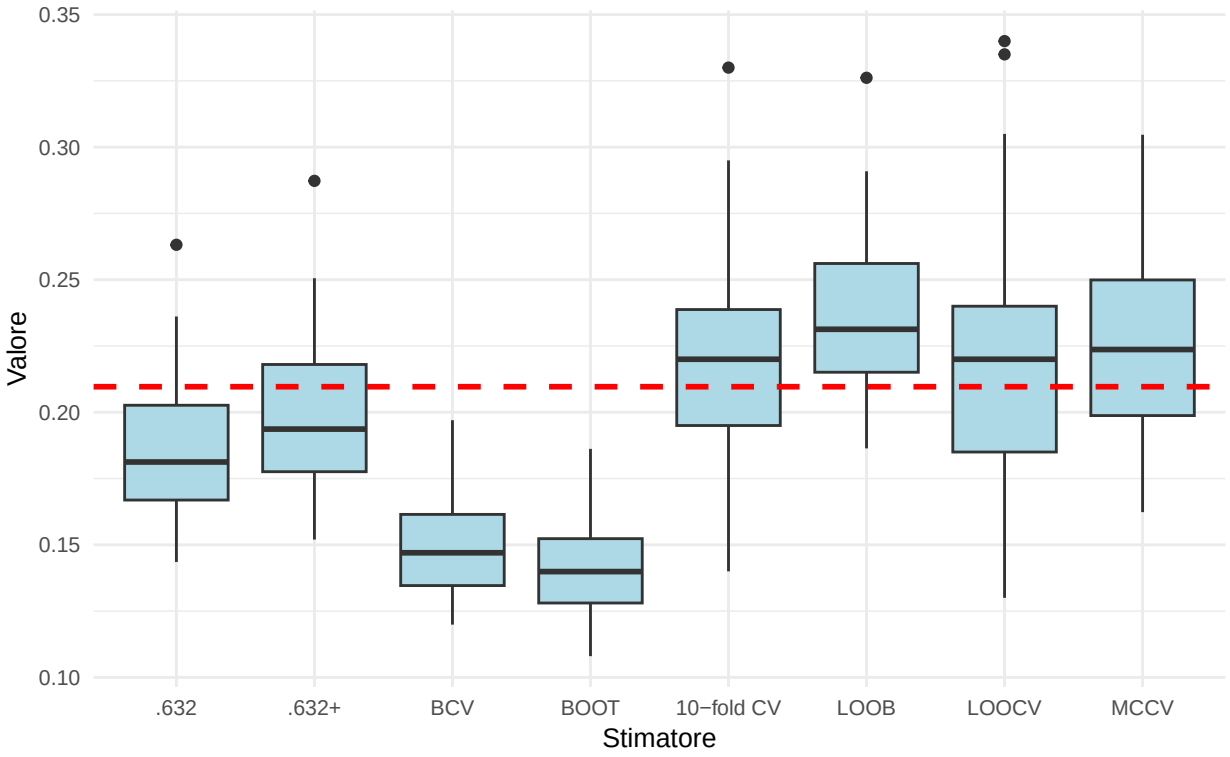
Figura 5.4: Boxplot per $n = 50$ dei metodi di stima dell'errore



I risultati mostrati sono ottenuti sul modello TREE con $n = 50$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Dai boxplot della Figura 5.4 ottenuti sul campione con $n = 50$, si nota come tutti gli stimatori siano caratterizzati da un'ampia variabilità, come mostrato dagli ampi intervalli interquantili e la presenza di outlier. Questa variabilità è particolarmente pronunciata nella 10-fold CV e nella LOOCV, suggerendo come questi metodi siano meno stabili con campioni piccoli. Nonostante il BCV ed il BOOT mostrino degli intervalli interquantili più stretti, molto inferiori rispetto a quelli degli altri stimatori, la loro mediana è spostata verso il basso, confermando quindi un bias negativo molto elevato. La presenza di outlier in diversi stimatori, incluso il BCV ed il LOOB, suggerisce occasionali stime di errore estreme che si discostano significativamente dalla mediana.

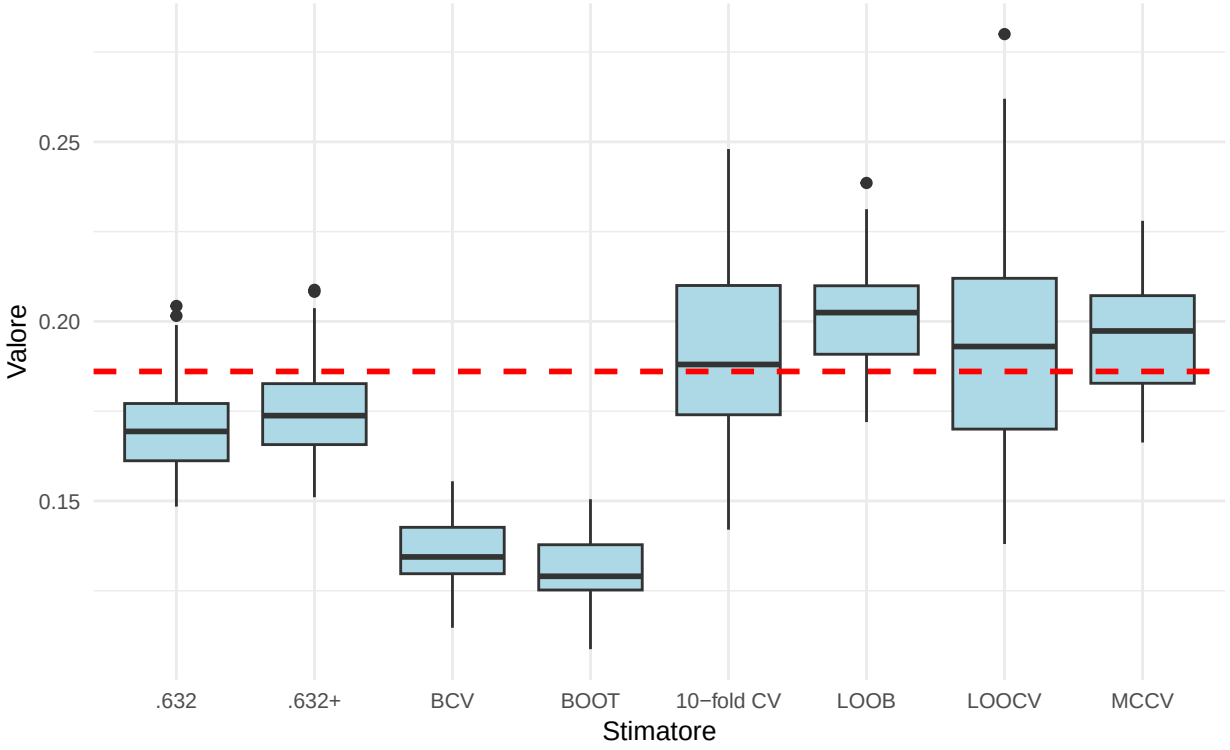
Figura 5.5: Boxplot per $n = 200$ dei metodi di stima dell'errore



I risultati mostrati sono ottenuti sul modello TREE con $n = 200$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Per quanto riguarda i boxplot ottenuti dal campione n=200 mostrati in Figura 5.5, essi mostrano una riduzione nella variabilità della stima, mostrando intervalli interquantili meno ampi. Il BCV ed il BOOT mostrano anche in questo caso intervalli interquantili particolarmente stretti ed una mediana più vicina all'errore reale rispetto ai boxplot della Figura 5.4, indicando una dimuzione del bias in valore assoluto. Anche il .632+, la 10-fold CV e la LOOCV mostrano un bias basso, con le loro mediane in prossimità dell'errore reale. La 10-fold CV e la LOOCV, seppur avendo una variabilità inferiore rispetto al campione con $n = 50$, continuano a mostrare degli intervalli interquantili più ampi rispetto agli altri stimatori.

Figura 5.6: Boxplot per $n = 500$ dei metodi di stima dell'errore



I risultati mostrati sono ottenuti sul modello TREE con $n = 500$. La linea tratteggiata rossa rappresenta il valore medio dell'errore reale ottenuto sulle 50 iterazioni.

Nella Figura 5.6 l'andamento verso una variabilità ridotta continua. La maggior parte degli stimatori mostrano intervalli interquantili molto ristretti, riflettendo delle stime dell'errore più stabili ed affidabili rispetto agli altri due campioni. Il BOOT ed il BCV continuano ad avere una bassa variabilità ma a costo di un alto bias negativo, come mostrato dalla lontananza della mediana dall'errore reale. La presenza di outlier è ancora presente ma meno frequente in confronto ai campioni con un numero inferiore di osservazioni. Il LOOB e la LOOCV mostrano ancora la presenza di alcuni outlier. Tuttavia, la loro frequenza e magnitudine è diminuita rispetto ai campioni più piccoli.

6. Conclusioni

Questo studio ha condotto una valutazione estensiva di vari metodi di stima dell'errore di previsione. Utilizzando un dataset incentrato sulle malattie cardiovascolari, la ricerca ha analizzato le prestazioni di questi metodi su diverse dimensioni del campione (n = 50, 200 e 500) e su due algoritmi di classificazione: il K-Nearest Neighbors e gli alberi decisionali. I principali parametri utilizzati per la valutazione sono stati la radice dell'errore quadratico medio, il bias e la variabilità. Questa analisi dettagliata ha fornito informazioni preziose sui punti di forza e di debolezza di ciascun stimatore, contribuendo a una migliore comprensione delle stime, che saranno riprese e discusse in questa sezione.

Le analisi hanno rivelato una consistente diminuzione in termini di RMSE all'aumentare dell'ampiezza dei campioni da 50 a 500, una tendenza che si conforma alle aspettative secondo cui dataset più grandi forniscono stime più affidabili e accurate.

Lo stimatore .632 si è dimostrato costantemente efficace in tutte le dimensioni del campione, evidenziando la sua robustezza mantenendo un basso RMSE e dimostrando la sua capacità di offrire stime dell'errore consistenti indipendentemente dalla dimensione del campione. Il .632 ha particolarmente eccelso nel modello di classificazione KNN, mentre nel modello ad alberi decisionali, data la maggiore presenza di overfitting, il .632 è stato inferiore al .632+ per tutti e tre i campioni. Il .632+ ha dunque mostrato la sua superiorità rispetto al suo predecessore grazie al modo in cui vengono assegnati i pesi ai due errori da cui è composto. Questo metodo di assegnamento dei pesi a seconda del livello di sovradattamento ha permesso al .632+ di ottenere un migliore compromesso tra bias e varianza nel modello ad alberi decisionali. Di conseguenza, si conclude che lo stimatore .632+ è meglio attrezzato per gestire le complessità dei dati del mondo reale, offrendo una valutazione più affidabile delle prestazioni del modello rispetto al .632.

Le dinamiche del bias e della variabilità si sono dimostrate cruciali nella valutazione dell'affidabilità di uno stimatore. La LOOCV e la 10-fold CV, nonostante il loro rendimento costante nel mostrare un basso bias sulle diverse dimensioni dei campioni, hanno rivelato una più alta variabilità rispetto ad altri stimatori. Dai risultati ottenuti si è ottenuto come la 10-fold CV abbia

prestazioni migliori della LOOCV nella maggior parte degli scenari. La 10-fold CV ha mostrato un equilibrio migliore tra bias e varianza della LOOCV, mostrando variabilità più bassa e mantenendo un bias pari a quello della LOOCV. Tuttavia, una eccezione è stata fatta nel caso del modello KNN sul campione con $n = 500$, in cui la LOOCV ha mostrato una variabilità inferiore rispetto alla 10-fold CV. Nonostante questa eccezione sottolinei come la differenza tra i due metodi non sia particolarmente netta in termini di stima, se si considera anche la grande potenza di calcolo aggiuntiva rischiesta dalla LOOCV, la 10-fold CV risulta il miglior metodo tra i due.

Il BCV e il BOOT hanno dimostrato una notevole efficacia nel mantenere un basso bias nei campioni di dimensioni più ridotte. Tuttavia, all'aumentare della dimensione del campione, questi stimatori hanno evidenziato un incremento del bias, segnalando una sensibilità che potrebbe limitarne l'applicabilità in dataset più ampi e diversificati. Le prestazioni di questi due stimatori sono risultate molto simili tra di loro nel corso dello studio, con entrambi che si sono distinti nel modello KNN, dove hanno registrato valori di RMSE significativamente migliori rispetto a quelli ottenuti con il modello ad alberi decisionali. In particolare, nel campione più piccolo del KNN, il BCV ed il BOOT si sono rivelati gli stimatori più efficaci in termini di RMSE. Tuttavia, con l'aumentare della dimensione del campione, la loro prestazioni sono state inferiori, specialmente nel caso del modello ad alberi decisionali. Questo suggerisce che, nonostante le ottime stime in presenza di piccoli campioni, il BCV ed il BOOT potrebbero non essere ideali per contesti con dataset di grandi dimensioni, evidenziando così delle potenziali limitazioni in termini di robustezza e versatilità.

Al contrario del BCV e del BOOT, il MCCV ha ottenuto dei risultati migliori rispetto gli altri stimatori nei campioni più grandi rispetto ai campioni più piccoli, sottolineando come la dimensione del campione sia importante nella sua applicazione. Più nel dettaglio, nel caso del modello KNN è stato il miglior stimatore solo nel campione con $n = 500$, mentre nel modello ad alberi decisionali è risultato essere tra i migliori su tutte le dimensioni dei campioni. L'MCCV è risultato essere in generale meno robusto del .632 e del .632+, in particolar modo nei campioni più piccoli. Questa prestazione inferiore viene dalla maggiore variabilità dovuta al processo di cross-validation sui diversi split. Ciò indica che per implementazioni future, specialmente in scenari che

coinvolgono dataset molto ridotti o quando è cruciale ottenere prestazioni consistenti con diverse dimensioni del campione, il MCCV potrebbe non essere la scelta più ottimale. Invece, dovrebbero essere considerati gli stimatori .632+ o .632 a causa delle loro stime più accurate e consistenti.

Il LOOB ha ottenuto risultati inferiori agli altri stimatori in entrambi i modelli e nei diversi campioni. A differenza di altri stimatori che hanno mostrato dei punti di forza in degli scenari specifici, il LOOB non ha eccelso in alcun area particolare. Sia nel modello KNN che nel modello ad alberi decisionali ha spesso mostrato una variabilità e bias più alti rispetto al altri stimatori come il .632, il .632+, il BCV ed il BOOT.

Questa mancanza di prestazioni eccezionali suggerisce che il LOOB potrebbe non essere adatto ai tipi di dati o ai compiti di stima dell'errore che sono stati oggetto di studio e che potrebbe non avere la robustezza e l'accuratezza necessarie per una stima affidabile dell'errore in contesti diversi.

L'analisi comparativa tra i due modelli di classificazione ha mostrato degli andamenti specifici. Ad esempio, lo stimatore BOOT ed il BCV hanno mostrato valori di RMSE maggiori nel modello ad alberi decisionali rispetto al modello KNN, suggerendo come la loro efficacia dipenda dalle caratteristiche del modello di classificazione utilizzato. Al contrario, il .632+ ha mostrato ottimi risultati in entrambi i modelli, sottolineando la sua adattabilità e robustezza. Anche la LOOCV e la 10-fold CV hanno mostrato stime affidabili con valori moderati di RMSE e bias particolarmente bassi sui diversi campioni e con i due modelli di classificazione.

Gli approfondimenti grafici derivati dai boxplot hanno chiarito la variabilità e il bias dei diversi stimatori. Nei campioni più piccoli, tutti gli stimatori hanno mostrato una variabilità significativa, come indicato dagli ampi intervalli interquantili. Questa variabilità si è notevolmente ridotta nei campioni più grandi, come evidenziato dagli intervalli interquantili più stretti e da un minor numero di valori anomali, in particolare per stimatori come il BCV, la 10-fold CV e la LOOCV. Queste rappresentazioni visive hanno messo in evidenza le posizioni delle mediane rispetto all'errore reale, indicando un bias minimo per il .632+, la LOOCV e la 10-fold CV, mentre il BCV ed il BOOT hanno mostrato una tendenza verso un bias negativo, fornendo dunque stime che sottostimano l'errore reale.

Per concludere, questo studio ha permesso di analizzare e confrontare diversi metodi di stima per l'errore di previsione in diversi scenari, mostrando i punti di forza e debolezza di ciascun metodo. Qeste informazioni risultano significative per ottimizzare l'accuratezza e l'affidabilità predittiva nelle applicazioni di machine learning, assicurandosi che i metodi di stima dell'errore di previsione siano allineati con le caratteristiche dei dati e dei modelli utilizzati.

Bibliografia

Ambroise C. e McLachlan G.J. (2002). “Selection Bias in Gene Extraction on the Basis of Microarray Gene-Expression Data”. In: *Proceedings of the National Academy of Sciences of the United States of America* 99(10), pp. 6562–6566.

Arlot S. e Celisse A. (2010). “A survey of cross-validation procedures for model selection”. In: *Statistics Surveys* 4, pp. 40–79.

Bischl B., Mersmann O., Trautmann H. e Weihs C. (2012). “Resampling methods for meta-model validation with recommendations for evolutionary computation”. In: *Evolutionary Computation* 20(2), pp. 249–275.

Borra S. e Di Ciaccio A. (2008). *Estimators of extra-sample error for non-parametric methods. A comparison based on extensive simulations*. Tech. Rep. 2008/19. Dept. of Statistics, Prob. and Appl. Statistics, Univ. of Roma La Sapienza.

Borra S. e Di Ciaccio A. (2010). “Measuring the prediction error. A comparison of cross-validation, bootstrap and covariance penalty methods”. In: *Computational Statistics & Data Analysis* 54(12), pp. 2976–2989.

Breiman L., Friedman J., Olshen R.A. e Stone C.J. (1984). *Classification and Regression Trees*. New York: Chapman & Hall.

Breiman L. e Spector P. (1992). “Submodel Selection and Evaluation in Regression. The X-Random Case”. In: *International Statistical Review / Revue Internationale de Statistique* 60(3), pp. 291–319.

Cawley G.C. e Talbot N.L.C. (2010). “On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation”. In: *Journal of Machine Learning Research* 11, pp. 2079–2107.

Chernick M.R. (2007). *Bootstrap Methods: A Guide for Practitioners and Researchers*. Wiley Series in Probability and Statistics. Hoboken: John Wiley & Sons, 2nd edition.

Dimitriadou E., Hornik K., Leisch F., Meyer D. e Weingessel A. (2023). *e1071: Misc Functions of the Department of Statistics, Probability Theory Group (Formerly: E1071), TU Wien*. R package version 1.7-14.

Efron B. (1983). "Estimating the Error Rate of a Prediction Rule: Improvement on Cross-Validation". In: *Journal of the American Statistical Association* 78(382), pp. 316–331.

Efron B. e Tibshirani R.J. (1993). *An Introduction to the Bootstrap*. Monographs on Statistics and Applied Probability. London: Chapman & Hall.

Efron B. e Tibshirani R.J. (1997). "Improvements on Cross-Validation: The .632+ Bootstrap Method". In: *Journal of the American Statistical Association* 92(438), pp. 548–560.

Ferrara E. (2024). "Fairness and Bias in Artificial Intelligence: A Brief Survey of Sources, Impacts, and Mitigation Strategies". In: *Sci* 6(1), p. 3.

Fu W.J., Carroll R.J. e Wang S. (2005). "Estimating misclassification error with small samples via bootstrap cross-validation". In: *Bioinformatics* 21(9), pp. 1979–1986.

Geisser S. (1975). "The Predictive Sample Reuse Method with Applications". In: *Journal of the American Statistical Association* 70(350), pp. 320–328.

Glele Kakaï R.L. e Palm R. (2009). "Empirical comparison of error rate-estimators in logistic discriminant analysis". In: *Journal of Statistical Computation and Simulation* 79(2), pp. 111–120.

Hastie T., Tibshirani R.J. e Friedman J.H. (2009). *The Elements of Statistical Learning. Data Mining, Inference, and Prediction*. Springer Series in Statistics. New York: Springer, 2nd edition.

Hefny A. e Atiya A.F. (2010). "A New Monte Carlo-Based Error Rate Estimator". In: Artificial Neural Networks in Pattern Recognition ANNPR 2010, pp. 37–47.

Ikechukwu E. (2016). "Evaluation of Error Rate Estimators in Discriminant Analysis with Multivariate Binary Variables". In: *American Journal of Theoretical and Applied Statistics* 5(4), pp. 173–179.

James G., Witten D., Hastie T. e Tibshirani R.J. (2021). *An Introduction to Statistical Learning. With Applications in R*. Springer Texts in Statistics. New York: Springer, 2nd edition.

James M. (2024). "The Ethical and Legal Implications of Using Big Data and Artificial Intelligence for Public Relations Campaigns in the United States". In: *International Journal of Communication and Public Relation* 9(1), pp. 38–52.

Jiang W. e Chen B.E. (2013). "Estimating prediction error in microarray classification: Modifications of the 0.632+ bootstrap when $n < p$". In: *Canadian Journal of Statistics* 41(1), pp. 133–150.

Jiang W. e Simon R. (2007). "A comparison of bootstrap methods and an adjusted bootstrap approach for estimating the prediction error in microarray classification". In: *Statistics in Medicine* 26(29), pp. 5320–5334.

Kim J.H. (2009). "Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap". In: *Computational Statistics & Data Analysis* 53(11), pp. 3735–3745.

Kohavi R. (1995). "A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection". In: IJCAI 14, pp. 1137–1143.

Kuhn M. e Johnson K. (2013). *Applied Predictive Modeling*. New York: Springer.

Kuhn M. e Johnson K. (2019). *Feature Engineering and Selection : A Practical Approach for Predictive Models*. Chapman and Hall/CRC Data Science. New York: Chapman & Hall.

Lachenbruch P.A. e Mickey M.R. (1968). "Estimation of Error Rates in Discriminant Analysis". In: *Technometrics* 10(1), pp. 1–11.

Molinaro A.M., Simon R. e Pfeiffer R.M. (2005). "Prediction error estimation: a comparison of resampling methods". In: *Bioinformatics* 21(15), pp. 3301–3307.

Raschka S. (2018). "Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning". In: *CoRR* abs/1811.12808.

Stone M. (1974). "Cross-Validatory Choice and Assessment of Statistical Predictions". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 36(2), pp. 111–133.

Van Sanden S. et al. (2012). “Genomic Biomarkers for a Binary Clinical Outcome in Early Drug Development Microarray Experiments”. In: *Journal of Biopharmaceutical Statistics* 22(1), pp. 72–92.

Varma S. e Simon R. (2006). “Bias in error estimation when using cross-validation for model selection”. In: *BMC Bioinformatics* 7(1), p. 91.

Wong T.T. (2015). “Performance evaluation of classification algorithms by k-fold and leave-one-out cross validation”. In: *Pattern Recognition* 48(9), pp. 2839–2846.