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

Aggiornamento ricorsivo delle stime in modelli di regressione per dati binari. Applicazione a dati NBA.

Relatore Prof. Nicola Sartori
Dipartimento di Scienze Statistiche

Laureando: Enrico Carraro
Matricola N 2057037

Anno Accademico 2022/2023

Indice

Introduzione

La statistica è una disciplina il cui interesse è in forte crescita per molti settori, dall'economia all'industria, dalla politica alla medicina, fino ad arrivare allo sport. Un aspetto che caratterizza tutti questi settori è il forte aumento della disponibilità di dati che c'è stato negli ultimi anni. Anche le modalità di acquisizione e salvataggio di tali dati sono in continua evoluzione e necessitano di strumenti di analisi sempre più evoluti.

In questa tesi verranno presentati strumenti di analisi per dati cross sezionali che si rendono disponibili in *batch*. Potendo essere tali dati di grandi dimensioni, con il passare del tempo può risultare complicato averli tutti disponibili in memoria, quindi risulta importante essere in grado di aggiornare le stime senza poter utilizzare i dati pregressi, ma solo quelli presenti nell'ultimo *batch*. Un ambito in cui ciò è molto comune è quello sportivo, in cui risulta di particolare interesse poter presentare delle statistiche riassuntive immediatamente dopo aver acquisito i dati dell'ultima partita disputata. I dati relativi ad un'intera stagione cestistica si prestano quindi come ottimo campo d'applicazione per le metodologie di aggiornamento delle stime che verranno proposte. Tali metodologie possono essere sfruttate anche per analizzare *dataset* con un numero di osservazioni arbitrariamente grande, considerandoli come molti *batch* sequenziali. Ci si propone inoltre di trattare la dipendenza temporale delle osservazioni tramite l'introduzione di una funzione di peso.

La tesi si organizza come segue: nel Capitolo 1 verrà presentata la tipologia di dati di riferimento che verranno utilizzati nel resto della tesi. Verranno quindi evidenziate le problematiche che si incontrano solitamente nell'analisi di questi e i vantaggi di applicare le soluzioni che verranno proposte nel seguito della tesi. Verrà inoltre spiegato brevemente l'obiettivo dell'analisi su un *dataset* reale contenente dati cestistici. Infine sarà fatta un'introduzione sul funzionamento del modello logistico e sulla correzione di Firth per diminuire la distorsione delle sue stime.

Nel Capitolo 2 sarà affrontata la trattazione teorica di diverse metodologie per l'aggiornamento ricorsivo delle stime di un modello logistico. Inizialmente sarà spiegata la *rho architecture* proposta da Luo & Song (2020), per poi proporre una sua modificazione atta a considerare la dipendenza temporale dei dati, inserendo una funzione peso per permettere al modello di considerare maggiormente i dati più vicini temporalmente al momento di stima. Per tale funzione saranno fatte varie proposte, valutando le caratteristiche di ognuna. Verrà inoltre preso in considerazione un metodo basato sulla decomposizione QR della matrice del modello per aggiornare le stime di un modello logistico con riduzione della distorsione di Firth.

Nel Capitolo 3 verrà analizzato un *dataset* contenente dati cestistici relativi ad un'intera stagione NBA con l'obiettivo di valutare quali sono le caratteristiche che portano un'azione ad essere più o meno pericolosa. Tale *dataset* sarà trattato come se ci si trovasse nel momento della raccolta dei dati che si rendono disponibili partita per partita. Durante la fase di analisi verranno applicate le metodologie di aggiornamento delle stime proposte nel Capitolo 2, comparandole e valutando i vantaggi e gli svantaggi di ognuna rispetto alle altre.

Tutte le analisi svolte nella tesi sono state fatte usando il *software* statistico R (v4.3.1; R Core Team, 2023). Il codice utilizzato è riportato nell'Appendice.

Capitolo 1

Metodologia

Con il passare degli anni raccogliere dati risulta sempre più facile, portando molte aziende ed enti a trovare la soluzione a molti dei loro problemi nella raccolta di quantità ingenti di informazioni. Un esempio è l'esplosione di internet e dell'utilizzo dei *cookies* da parte dei *social network* e dei motori di ricerca, che per suggerirci contenuti interessanti salvano gran parte delle caratteristiche dei nostri profili e delle ultime ricerche. Il fatto di avere a disposizione sempre più dati è ormai comune in qualsiasi campo, a partire da quello medico fino ad arrivare a quello industriale, passando per l'ambito demografico e geospaziale. Da un punto di vista statistico questa crescita esponenziale di informazioni a disposizione per allenare i modelli a comprendere la natura di diversi fenomeni e prevederne gli sviluppi futuri risulta incredibilmente utile, ma ci mette di fronte a situazioni nuove che possono creare diverse problematiche, che verranno discusse nel seguito insieme ad alcune possibili soluzioni alle stesse.

1.1 Dati

1.1.1 Tipologia di dati

Una particolare tipologia di dati che verranno trattati nel seguito di questo trattato sono dati sezionali, anche noti come *cross-sectional data* che permettono di raccogliere caratteristiche di diversi soggetti e di analizzarle nello stesso momento senza considerare le distanze temporali. Raccogliendo le informazioni in diversi istanti temporali risulta però che il *dataset* a disposizione non resti uguale ma diventi man mano più grande e che quindi si modifichi, aggiungendo nuovi dati con il passare del tempo e l'evolvere del fenomeno interesse di studio.

Sia $b \in \{1, \dots\}$ un indice temporale, indichiamo quindi con D_b l'insieme dei dati inerenti al solo istante temporale b mentre con $\widetilde{D}_b$ l'intero *dataset* cumulato disponibile al tempo b. Supponendo che vengano raccolte sempre le stesse variabili per ogni unità statistica indipendentemente dall'istante temporale in cui sono stati raccolti i dati, avremo

$$\widetilde{D}_b = \begin{bmatrix} D_1 \\ D_2 \\ D_3 \\ \vdots \end{bmatrix} \tag{1.1}$$

dove D_b è una matrice $n_b \times p$ con N_b che è il numero di unità statistiche nel *database* raccolto all'istante temporale b e p è il numero di variabili, eventualmente dopo aver ricodificato tutti i fattori in opportune variabili *dummy*. La matrice $\widetilde{D}_b$ ha quindi dimensione $N_b \times p$ con $N_b = \sum_b n_b$.

1.1.2 Problematiche

L'aumento dei dati a disposizione prima ancora che da un punto di vista statistico crea un problema dal punto di vista tecnico in quanto può risultare difficile avere a disposizione uno spazio di memoria sufficiente a salvare tutti i dati. Può capitare infatti di strutturare un esperimento in cui si vogliono raccogliere dati per un determinato periodo di tempo, riservando perciò un certo spazio di memoria del *server* a tale esperimento, decidendo invece in un secondo momento di raccogliere una quantità maggiore di dati, andando quindi a modificare le necessità strutturali iniziali.

Il secondo problema che si pone è invece strettamente computazionale in quanto, pur assumendo di avere a disposizione tutto lo spazio necessario per salvare i dati, nel momento in cui tali dati devono essere analizzati si è spesso interessati a risultati in tempo reale, ovvero a cominciare l'analisi prima di avere a disposizione l'intero *dataset*, in modo da avere dei risultati parziali che verranno aggiornati con l'arrivo di nuovi dati. Applicando i metodi classici di analisi è quindi necessario stimare l'intero modello su tutti i dati disponibili al momento in cui si vuole avere dei risultati. In particolare, considerando di aver raggiunto un numero di osservazioni sufficienti ad aver risultati attendibili all'istante temporale B_1 per stimare il modello ad ogni istante è necessario seguire i passi dell'Algoritmo 1.

Con il passare del tempo e la raccolta di nuovi dati il database diventerà sempre più grande e pesante, portando il tempo di stima del modello ad essere sempre più

Algoritmo 1 Aggiornamento modello classico

1: Stimare il modello iniziale sui dati $\widetilde{D}_{B_1}$
2: **for** $b \in \{B_1 + 1, \dots\}$ **do**
3: Stimare nuovamente il modello sull'intero *dataset* $\widetilde{D}_b$
4: **end for**

lungo. Nel caso sia di interesse un aggiornamento ad istanti temporali ravvicinati, ad esempio ogni minuto nel caso di azioni di borsa, può capitare che il tempo di stima sia addirittura superiore al *lag* temporale in cui si raccolgono nuovi dati, rendendo così inutile l'adattamento di un modello in quanto già troppo vecchio prima che sia finito il suo processo di stima.

1.1.3 Soluzioni

Una soluzione ad entrambe le problematiche, ovvero eccessivo utilizzo di memoria e peso computazionale, risulta essere l'utilizzo di dati di flusso, a cui spesso ci si riferisce anche come *streaming data*. Ciò consiste nel salvare in memoria solo i dati raccolti all'ultimo istante temporale disponibile, ossia il *chunk*, o *batch*, D_b al generico tempo b, cancellando tutti i dati precedenti $\widetilde{D}_{b-1}$. In questo modo la quantità di memoria necessaria a salvare i dati è notevolmente diminuita, ma è necessario trovare un modo per aggiornare le stime del modello avendo a disposizione solamente i dati al tempo presente ed alcune quantità stimate fino al tempo $b - 1$. Diverse proposte sono state fatte da Robbins & Monro (1951), Toulis et al. (2014), Luo et al. (2022), Luo & Song (2023) e Luo et al. (2023).

L'obiettivo è quindi quello di aggioranare il modello seguendo i passi dell'algoritmo 2 invece che dell'Algoritmo 1.

Algoritmo 2 Aggiornamento ricorsivo

1: Stimare il modello iniziale sui dati $\widetilde{D}_{B_1}$
2: **for** $b \in \{B_1 + 1, \dots\}$ **do**
3: Aggiornare il modello con i dati D_b
4: **end for**

1.2 Applicazione a dati reali

In questa tesi è di interesse oltre ad una trattazione metodologica e computazionale dell'argomento anche un'applicazione a dati reali reperibili nel sito Big Data Ball (2019). Tali dati si riferiscono all'intera stagione NBA (*National Basket Association*) 2018/2019,

ovvero l'ultima intera stagione disponibile in tale sito prima della pandemia COVID-19. L'analisi di dati sportivi è un tema estremamente attuale e diversi sono gli esempi in letteratura, tra cui da Terner & Franks (2021), Dandolo (2019), Dixon & Coles (1997) e Artuso (2020), da cui si è preso spunto per le analisi condotte.

Nel *dataset* a disposizione sono stati raccolti per ogni partita della stagione regolare e dei *playoff* i dati *play-by-play*, ovvero cos'è successo in ogni singola giocata di tale partita. Ogni riga si riferisce quindi a una giocata successa ad un determinato istante nel corso della partita. Le righe sono ordinate in ordine cronologico e in quelle che riportano un tiro a canestro è indicato se tale tiro è stato segnato o meno. Dopo aver effettuato un'operazione di *preprocessing* sui dati si considererà come risposta una nuova variabile dicotomica y costruita a partire dal risultato dei vari tiri come variabile risposta. Tale variabile assumerà valore 1 qualora una giocata, che verrà codificata in una singola riga, risulti importante al fine di segnare un canestro nell'azione in cui è avvenuta, mentre assumerà valore 0 qualora tale giocata non sia risultata in un successivo canestro. Per valutare l'importanza di ogni giocata sulla possibilità di segnare è stato tenuta in considerazione la distanza temporale tra la giocata stessa e l'eventuale canestro effettuato. Viene preso in considerazione anche un set di possibili covariate per ogni osservazione, che indicano caratteristiche tecniche e temporali della giocata, oltre al giocatore che la effettua e la squadra di appartenenza dello stesso.

Sono stati considerati come *chunk* le singole partite, considerando il loro ordinamento cronologico e non la distanza temporale delle stesse, questo perché essendo spesso molto ravvicinate non dovrebbe influire particolarmente sulla stima del modello. Si hanno quindi a disposizione i *chunk* D_b, $b = 1, \ldots, B$, con $B = 1312$ che è il numero di totale di partite nella stagione 2018-2019.

L'obiettivo dell'analisi è quindi quello di valutare l'importanza di ogni variabile sulla probabilità di realizzare un canestro, indagando sia da un punto di vista qualitativo che quantitativo quali sono le caratteristiche che aumentano o diminuiscono la probabilità di segnare e di quanto tale probabilità varia. Ci si pone inoltre l'obiettivo di valutare la variabilità delle stime in modo tale da poter valutare se i vari parametri associati alle variabili risultano significativamente diversi da 0. Si andrà inoltre a valutare quali sono state le squadre e i giocatori più in forma nei vari momenti della stagione, cosa resa possibile dall'aver effettuato l'analisi al termine di ogni singola partita.

1.3 Modello di regressione logistica

Essendo di interesse analizzare dati di natura dicotomica è necessario inizialmente introdurre i metodi *standard* di analisi maggiormente utilizzati. Si fornisce quindi una trattazione generale dei modelli lineari generalizzati, a cui si farà riferimento in seguito come GLM, presentati da McCullagh & Nelder (1983) per valutare la relazione tra la variabile risposta e le covariate, siano esse numeriche o fattoriali. Il primo modello che è stato considerato è un modello di regressione logistica, appartenente appunto alla più ampia famiglia dei modelli lineari generalizzati.

Indicato con $y = (y_1, y_2, \ldots, y_n)^T$ il vettore di osservazioni della variabile risposta, si assume che le singole osservazioni y_i siano realizzazioni variabili casuali

$$Y_i \sim Bernoulli(\pi_i),$$

$i = 1, \ldots, n$ con $\pi_i = \mu_i \in (0, 1)$ parametro ignoto. Quindi y è realizzazione di $Y = (Y_1, \ldots, Y_n)^T$.

Per ogni unità è raccolto inoltre un vettore di p covariate $x_i = (x_{i1}, \ldots, x_{ip})$, che vengono messe in relazione con la probabilità π_i tramite un vettore di p parametri di regressione $\beta = (\beta_1, \ldots, \beta_p)$. Nello specifico si definisce il predittore lineare η_i e lo si mette in relazione alla probabilità π_i attraverso

$$g(\pi_i) = \eta_i = \boldsymbol{x}_i\beta = \beta_1 x_{i1} + \cdots + \beta_p x_{ip} = \sum_{r=1}^{p} \beta_r x_{ir,}$$

con $g(\cdot)$ funzione di legame da specificare. Le funzioni legame più comuni per dati binari sono la funzione logistica o *logit*, la funzione *probit*, la funzione *log-log* complementare e la funzione *cauchit*. In questa tesi è stata considerata solamente la funzione logistica, ossia

$$g(\mu_i) = \log\left(\frac{\mu_i}{1-\mu_i}\right) = \boldsymbol{x}_i\beta,$$

che implica

$$\mu_i = \frac{\exp(\boldsymbol{x}_i\beta)}{1+\exp(\boldsymbol{x}_i\beta)}.$$

Essendo la densità di $Y_i \sim Bi(1, \pi_i)$ pari a

$$p(y_i; \pi_i) = \pi_i^{y_i}(1-\pi_i)^{1-y_i},$$

il modello logistico specificato porta alla funzione di verosimiglianza,

$$L(\beta; X, y) \propto \prod_{i=1}^{n} \left[\frac{\exp(\eta_i)}{1 + \exp(\eta_i)} \right]^{y_i} [1 + \exp(\eta_i)]^{y_i - 1} \tag{1.2}$$

con X la matrice del disegno, con i-esima riga $\boldsymbol{x}_i$.

La stima del modello si ottiene massimizzando la verosimiglianza (1.2), ottenendo la stima di massima verosimiglianza $\hat{\beta}$.

1.3.1 Interpretazione dei parametri

Utilizzando il modello di regressione logistica è possibile interpretare il predittore lineare η_i in termini di logaritmo della quota (come spiegato ad esempio in Salvan et al., 2020, paragrafo 3.4.1). La quota

$$\frac{\pi_i}{1 - \pi_i}$$

è il rapporto tra probabilià di successo e di insuccesso. Moltiplicandola per 100 si ottiene quindi il numero atteso di successi ogni 100 insuccessi. Nell'esempio del basket, una quota relativa all'azione "passaggio" pari a 2 con tutte le covariate fissate indica che ogni 100 azioni non importanti al fine di segnare un canestro, ce ne sono 200 che portano a segnare. Il generico coefficiente β_r esprime quindi l'effetto sul logaritmo della quota di un incremento unitario di x_{ir}, fermo restanto il valore delle restanti esplicative del modello. Il fattore moltiplicativo della quota per un aumento unitario della generica covariata x_{ir} risulta essere pari a $\exp(\beta_r)$, portando quindi ad interpretare coefficienti positivi del modello come fattori che fanno aumentare la quota, mentre coefficienti negativi come fattori che la fanno diminuire.

1.4 Correzione di Firth

Dovendo stimare il modello logistico solo su una porzione ristretta dei dati ($\widetilde{D}_{B_1}$) con l'intenzione di aggiornarlo successivamente in modo ricorsivo, può capitare che si abbiano problemi di convergenza in fase di stima per i noti problemi di separabilità che si possono creare nei modelli per dati binari (Salvan et al., 2020). Questo è particolarmente problematico quando si hanno a disposizione poche osservazioni rispetto alla dimensione p di β. Dal punto di vista pratico, ci si ritroverebbe ad avere stime di alcuni parametri in modulo molto grandi, con *standard error* a loro volta particolarmente elevati, dovuti a problemi di inversione di matrici essenzialmente singolari. Questo è dovuto al fatto

che all'avvicinarsi del numero di paramentri al numero di osservazioni disponibili le assunzioni sottostanti alle proprietà dello stimatore di verosimiglianza vengono a mancare e quindi si possono avere probabilità stimate numericamente uguali a 0 o a 1, come mostrato da Candès & Sur (2020).

Una soluzione a questo problema risulta essere la correzione per la distorsione proposta da Firth (1993) che consiste nell'effettuare una modifica della funzione punteggio con l'obiettivo di ottenere uno stimatore con distorsione più ridotta di quella dello stimatore di massima verosimiglianza.

Indicando con $\ell(\beta)$ la funzione di log-verosimiglianza e con $d\ell(\beta)/d\beta$ la sua derivata, le equazioni che portano ad ottenere le stime di massima verosimiglianza $\hat{\beta}$ sono date da

$$U(\beta) = 0,$$

dove $U(\beta) = U(\beta; D)$ è la funzione punteggio inerente ai dati D.

Firth (1993) propone di introdurre una piccola distorisione nella funzione punteggio $U(\beta)$ con lo scopo di ottenere un corrispondente stimatore $\hat{\beta}^*$ con distorsione ridotta rispetto a $\hat{\beta}$ In particolare, indicando con $b(\beta)$ il termine dominante della distorsione di $\hat{\beta}$, $\hat{\beta}^*$ si può ottenere come soluzione dell'equazione di stima

$$U^*(\beta) = U(\beta) - i(\beta)\, b(\beta) = 0, \tag{1.3}$$

dove $i(\beta) = E_\beta\left(dU(\beta)/d\beta^T\right)$ è la matrice di informazione di Fisher.

Si veda Kosmidis et al. (2020) per un'applicazione dei metodi di riduzione della distorsione in modelli lineari generalizzati.

In particolare, essendo il modello logistico una famiglia esponenziale con parametro canonico β, stimare i parametri utilizzando la correzione della distorsione proposta da Firth (1993) equivale a massimizzare la verosimiglianza penalizzata

$$L^*(\beta) = L(\beta)\left|i(\beta)\right|^{\frac{1}{2}}.$$

Si noti che $\left|i(\beta)\right|^{\frac{1}{2}}$ corrisponde alla distribuzione a priori di Jeffreys (Jeffreys, 1946), il che permette di dare un'interpretazione bayesiana al metodo, essendo $\hat{\beta}^*$ la moda della distribuzione a posteriori basata sulla priori non informativa di Jeffreys.

Nel caso della regressione logistica si ha

$$i(\beta) = j(\beta) = -\frac{dU(\beta)}{d\beta^T} = X^T W X$$

dove $j(\beta)$ è la matrice di informazione osservata e $W = \text{diag}\{\pi_i(1-\pi_i)\}$. Il determinante di $i(\beta)$ viene massimizzato da $\pi_i = \frac{1}{2}, i = 1, \ldots, n$, che corrisponde a $\beta = 0$. In questo modo la riduzione della distorsione equivale a comprimere le stime verso lo 0 utilizzando come funzione di *shrinkage* la priori di Jeffreys (Kosmidis & Firth, 2021).

Capitolo 2

Aggiornamento ricorsivo delle stime

Una volta stimato il modello iniziale, sia esso un modello di regressione logistica o un modello di regressione logistica con correzione di Firth, sul dataset $\widetilde{D}_{B_1}$ è necessario trovare un metodo per aggiornare tale modello con i dati che verranno raccolti negli istanti temporali $B_1 + 1, B_1 + 2, \ldots$. Si immagini di avere a disposizione solamente due dataset arrivati sequenzialmente, $D_1 = (x_{1,1}, \ldots, x_{1,n_1})$ composto da n_1 osservazioni e $D_2 = (x_{2,1}, \ldots, x_{2,n_2})$ composto da n_2 osservazioni. Sia $\delta(D_1)$ la media campionaria per D_1, che può essere facilmente aggiornata con il *batch* di dati D_2 tramite

$$\delta(D_1 \cup D_2) = \frac{1}{n_1 + n_2}\left(\sum_{i=1}^{n_1} x_{1i} + \sum_{i=1}^{n_2} x_{2i}\right) = \frac{1}{n_1 + n_2}\left\{n_1\delta(D_1) + \sum_{i=1}^{n_2} x_{2i}\right\}.$$

In questo modo l'aggiornamento della media campionaria è avvenuto utilizzando il valore dello stimatore al punto precedente, $\delta(D_1)$, invece di utilizzare i dati D_1. Uno stimatore che soddisfa questa proprietà verrà chiamato stimatore rinnovabile.

Nel momento in cui non è di interesse aggiornare solo la media, ma le stime di un GLM è necessario esplorare algoritmi più complicati e spesso iterativi. Nel seguito verrà descritta la *rho architecture* proposta da Luo & Song (2020) e delle sue modificazioni.

2.1 *Rho architecture*

La *rho architecture* viene proposta come un espansione della *lambda architecture* descritta da Marz & Warren (2015). Tale algoritmo è un sistema di salvataggio e analisi di *big data* che si basa sul funzionamento sincronizzato di tre livelli: lo *Speed layer*, il *Batch layer* e il *Serving layer* il cui funzionamento viene mostrato in Figura 2.1. Le

stime incrementali vengono calcolare nello *Speed layer* analizzando i dati di flusso tramite lo *Spark system* di Bifet et al. (2015). Sfortunatamente questo potente algoritmo di aggiornamento ignora completamente la possibilità di fare inferenza in tempo reale, in quanto non prevede alcun modo per calcolare e salvare l'informazione di Fisher, fondamentale per fare inferenza statistica.

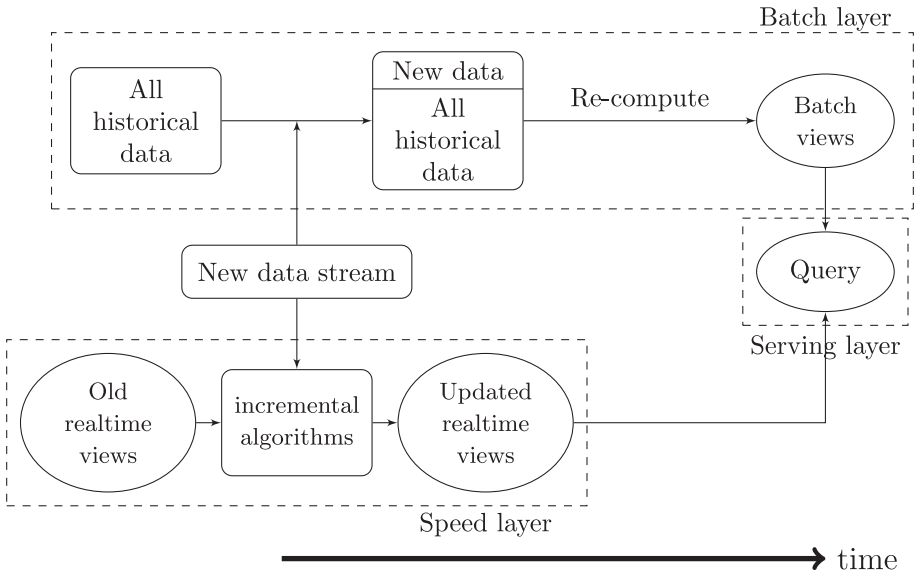

FIGURA 2.1: Diagramma indicante i vari *layer* presenti nella *lambda architechture* (Luo & Song, 2020).

La *rho architecture* si propone quindi a rispondere tre quesiti base:

1. che tipo di statistiche si vogliono salvare nell'*Inference layer*;
2. come aggiornare tali statistiche di riepilogo richieste per la stima puntuale e l'inferenza senza utilizzare i dati grezzi dei *chunk* precedenti;
3. come massimizzare l'efficienza di stima delle stime ricorsive in modo tale che siano asintoticamente equivalenti alle stime di massima verosimiglianza ottenute sull'intero *dataset*.

Lo scopo è quindi di adattare un GLM

$$E\left(Y_i|\boldsymbol{x}_i\right) = g(\boldsymbol{x}_i\beta), \qquad i = 1, \ldots, N_b$$

dove N_b è la dimensione campionaria del *chunk* b. Fissato il modello, assumiamo che ogni osservazione sia campionata indipendentemente da

$$\left(Y_i; \boldsymbol{x}_i\right) \sim p\left(y; \boldsymbol{x}_i, \beta_0, \phi_0\right), \qquad i = 1, \ldots, N_b$$

dove β_0 è il vero valore del parametro di interesse e ϕ_0 è il vero valore del parametro di dispersione.

Scrivendo la funzione di verosimiglianza di un GLM nella forma di una famiglia di dispersione esponenziale come si può vedere in Jorgensen (1997)

$$\begin{aligned}\ell_{N_b}\left(\beta,\phi;\widetilde{D}_b\right) &= \sum_{i\in\widetilde{D}_b}\log\left\{f\left(y;\boldsymbol{x},\beta_0,\phi_0\right)\right\}\\ &= \sum_{i\in\widetilde{D}_b}\log\left\{\alpha\left(y_i;\phi\right)\right\}+\frac{1}{2\phi}\sum_{i\in\widetilde{D}_b}d\left(y_i;\mu_i\right)\end{aligned} \tag{2.1}$$

dove $d\left(y_i;\mu_i\right)$ è la funzione di devianza unitaria che coinvolge il parametro di media $\mu_i = E\left(y_i|\boldsymbol{x}_i\right)$ e $\alpha\left(\cdot\right)$ è un fattore di normalizzazione dipendente solo dal parametro di dispersione $\phi > 0$.

Nel caso in cui la variabile risposta abbia distribuzione binomiale $Bi(n_i,\pi_i)$, il parametro di dispersione risulta pari a 1. Si può quindi scrivere la funzione di verosimiglianza binomiale come

$$L_{N_b}\left(\theta;y\right)\propto\prod_{i\in\widetilde{D}_b}\left(1-\pi_i\right)^{n_i}\exp\left\{y_i\,\log\left(\frac{\pi_i}{1-\pi_i}\right)\right\}$$

ottenendo la funzione di log-verosimiglianza

$$\ell_{N_b}\left(\theta;y\right)=-\frac{1}{2}\sum_{i\in\widetilde{D}_b}2\left[n\,log\left(\frac{1}{1-\pi_i}\right)+y_i\,log\left(\frac{1-\pi_i}{\pi_i}\right)\right]. \tag{2.2}$$

Confrontando le formule (2.1) e (2.2) si può notare che per una variabile risposta con distribuzione esponenziale si ha

$$\alpha\left(y_i;\phi\right)=\binom{n_i}{y_i},\qquad d\left(y_i;\mu_i\right)=2\left[n\,\log\left(\frac{1}{1-\mu_i}\right)+y_i\,\log\left(\frac{1-\mu_i}{\mu_i}\right)\right]$$

con $\mu_i=\pi_i$.

Lo scopo è quindi di presentare una struttura *on-line* in cui sia possibile aggiornare sia le stime puntuali che la stima della variabilità utilizzando solo i dati raccolti al periodo corrente e delle statistiche di riepilogo calcolate sui dati storici, con l'obiettivo di poter fare inferenza. Tale struttura rispetta:

1. la differenza in norma l_2 tra lo stimatore rinnovabile e la stima di massima verosimiglianza tende a 0 con l'aumentare della dimensione campionaria N_b;

2. essendo computazionalmente vantaggioso, tale metodo non richiede l'accesso ad alcun dato appartenente ai precedenti *chunk* per l'aggiornamento al passo corrente;

3. è possibile fare inferenza in *real-time* basandosi sul test di Wald.

Sia $\widetilde{\beta}_b$ lo stimatore rinnovabile inizializzato con la stima di massima verosimiglianza $\hat{\beta}_{B_1}$ calcolata sul blocco iniziale di dati $\widetilde{D}_{B_1}$. Per $b = B_1 + 1, B_1 + 2, \ldots$ una precedente stima $\widetilde{\beta}_{b-1}$ viene aggiornata a $\widetilde{\beta}_b$ quando il *batch* D_b arriva. Dopo l'aggiornamento il *chunk* D_b non risulta essere più disponibile e si hanno a disposizione solamente la stima $\widetilde{\beta}_b$ e la statistica riassuntiva $J_b\left(\widetilde{\beta}_b; D_b\right)$, ossia la matrice Hessiana negativa della funzione score $U_b\left(\beta; D_b\right) = \sum_{i \in D_b} U\left(\beta; \boldsymbol{X}_i, y_i\right)$

$$J_b\left(\beta; D_b\right) := -\Delta_\beta U_b\left(\beta; D_b\right).$$

2.1.1 Algoritmo di stima

Partendo con solo due *batch* di dati D_1 e D_2, dove D_2 arriva dopo D_1 vogliamo aggiornare le stima iniziale $\widetilde{\beta}_1 = \hat{\beta}_1$ al nuovo valore di $\widetilde{\beta}_2$. È noto che la stima di massima verosimiglianza $\hat{\beta}_1$ soddisfa l'equazione di stima $U_1\left(\hat{\beta}_1; D_1\right) = 0$ e che la stima $\widetilde{\beta}_2$ soddisfa l'equazione di stima aggregata

$$U_1\left(\widetilde{\beta}_2; D_1\right) + U_2\left(\widetilde{\beta}_2; D_2\right) = 0. \tag{2.3}$$

In tal caso la stima $\widetilde{\beta}_2$ corrisponderebbe esattamente alla stima di massima verosimiglianza $\hat{\beta}_2$. Per risolvere tale equazione di stima in $\widetilde{\beta}_2$ sarebbe però necessario avere a disposizione entrambi i *batch* di dati D_1 e D_2. Per ovviare alla mancanza del *chunk* D_1, si può utilizzare il termine lineare dell'espansione in serie di Taylor del primo termine dell'equazione (2.3) intorno a $\widetilde{\beta}_1$,

$$U_1\left(\widetilde{\beta}_1; D_1\right) + J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2\left(\widetilde{\beta}_2; D_2\right) + \mathcal{O}_p\left(\|\widetilde{\beta}_2 - \widetilde{\beta}_1\|\right) = 0. \tag{2.4}$$

dove $\|\cdot\|$ è la norma l_2 dell'argomento e $\mathcal{O}\left(\cdot\right)$ indica una quantità asintoticamente dell'ordine dell'argomento come probabilità. Si riporta quindi la definizione di Pace et al. (Paragrafo 7.2, 2022). Sia Y_b una sequenza di numeri reali, si dice che Y_b è asintoticamente di ordine $\mathcal{O}\left(\|\widetilde{\beta}_b - \widetilde{\beta}_{b-1}\|\right)$, scrivendo $Y_b = \mathcal{O}_p\left(\|\widetilde{\beta}_b - \widetilde{\beta}_{b-1}\|\right)$, se per ogni $\varepsilon > 0$, esiste un valore reale $K = K_\varepsilon > 0$ e un numero naturale $\bar{b} = \bar{b}_\varepsilon$ tale che per ogni $b \geq \bar{b}$

$$P\left(\left|\frac{Y_b}{\|\widetilde{\beta}_b - \widetilde{\beta}_{b-1}\|}\right| < K\right) > 1 - \varepsilon.$$

Dato che entrambi i *batch* di dati sono campionati indipendentemente dallo stesso modello sottostante con vero valore del parametro β_0, quando il minimo tra le dimensioni campionarie dei due *chunk* è sufficientemente grande, sotto deboli condizioni si può affermare che entrambi gli stimatori $\widetilde{\beta}_1$ e $\widetilde{\beta}_2$ sono stimatori consistenti di β_0 come dimostrato da Fahrmeir & Kaufmann (1985), il che implica che asintoticamente si può ignorare il termine di errore $\mathcal{O}_p\left(\|\widetilde{\beta}_2 - \widetilde{\beta}_1\|\right)$ dell'equazione (2.4). In tal modo il valore della nuova stima $\widetilde{\beta}_2$ può essere determinato risolvendo l'equazione

$$U_1\left(\widetilde{\beta}_1; D_1\right) + J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2\left(\widetilde{\beta}_2; D_2\right) = 0, \tag{2.5}$$

tenendo conto che $U_1\left(\widetilde{\beta}_1; D_1\right) = 0$ in quanto $\widetilde{\beta}_1 = \hat{\beta}_1$ è la stima di massima verosimiglianza di β sui dati D_1. Il nuovo stimatore è quindi soluzione dell'equazione

$$J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2\left(\widetilde{\beta}_2; D_2\right) = 0. \tag{2.6}$$

Tale stimatore è asintoticamente equivalente alla stima di massima verosimiglianza a meno di errori del secondo ordine. Avendo aggiornato $\widetilde{\beta}_1$ con $\widetilde{\beta}_2$ con i soli dati presenti nel *batch* D_2, ci si può riferire a $\widetilde{\beta}_2$ come stimatore rinnovabile di β_0 e all'equazione (2.6) come equazione di stima incrementale.

Nei i GLM la soluzione dell'equazione (2.6) si deve trovare per via numerica, ad esempio con l'utilizzo dell'algoritmo di Newton-Rhapson oppure della sua variante Fisher *scoring*. I due algoritmi risultano equivalenti per i GLM con funzione legame canonica, e quindi anche per il modello binomiale con funzione di legame logistica, e prevedono che la stima all'iterazione $(r+1)$-esima sia

$$\begin{aligned}\widetilde{\beta}_2^{(r+1)} = \widetilde{\beta}_2^{(r)} + &\left\{J_1\left(\widetilde{\beta}_1; D_1\right) + J_2\left(\widetilde{\beta}_2; D_2\right)\right\}^{-1} \\ &\left\{J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2^{(r)}\right) + U_2\left(\widetilde{\beta}_2^{(r)}; D_2\right)\right\},\end{aligned} \tag{2.7}$$

dove non sono necessari i dati del *chunk* D_1, ma solo la stima $\widetilde{\beta}_1$ e la matrice Hessiana negativa $J_1\left(\widetilde{\beta}_1; D_1\right)$. Per velocizzare il calcolo è inoltre possibile evitare di calcolare la matrice $J_2\left(\widetilde{\beta}_2^{(r)}; D_2\right)$ rimpiazzando $\widetilde{\beta}_2^{(r)}$ con $\widetilde{\beta}_1$, come fatto in letteratura anche da Song et al. (2005), e ottenendo quindi il seguente algoritmo iterativo per l'aggiornamento

ricorsivo delle stime

$$\begin{aligned}\widetilde{\beta}_2^{(r+1)} &= \widetilde{\beta}_2^{(r)} + \left\{\sum_{j=1}^{2} J_j\left(\widetilde{\beta}_1; D_j\right)\right\}^{-1}\left\{J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2^{(r)}\right) + U_2\left(\widetilde{\beta}_2^{(r)}; D_2\right)\right\} \\ &= \widetilde{\beta}_2^{(r)} + \left\{J_1\left(\widetilde{\beta}_1; D_1\right) + J_2\left(\widetilde{\beta}_1; D_2\right)\right\}^{-1} U_2^{(r)}, \end{aligned} \tag{2.8}$$

dove

$$U_2^{(r)} = J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2^{(r)}\right) + U_2\left(\widetilde{\beta}_2^{(r)}; D_2\right).$$

La differenza tra lo stimatore proposto e la stima di massima verosimiglianza nasce dall'approssimazione della funzione punteggio $U_2\left(\widetilde{\beta}_1; D_1\right)$. Come dimostrato da Luo & Song (2020, paragrafo 4.1) tale distanza tende a zero ad un tasso pari a $1/N_2$ con $N_2 = |\widetilde{D}_2| = n_1 + n_2$. Dato che con il passare del tempo e con l'arrivo dei nuovi dati la dimensione campionaria totale $N_b = \sum_{j=1}^{b} n_j$ cresce molto velocemente, i due stimatori $\widetilde{\beta}_b$ e $\hat{\beta}_b$ risultano numericamente simili e tendono ad assumere lo stesso valore dopo molte iterazioni.

Per generalizzare e passare da solo due *batch* di dati ad un numero generico b di *batch* non specificato a priori basta ripetere i passi precedenti considerando come primo *batch* $\widetilde{D}_{b-1}$ e come secondo *batch* D_b. Ripetendo ciò per tutti i batch disponibili si ottiene la seguente equazione di stima:

$$\sum_{j=1}^{b-1} J_j\left(\widetilde{\beta}_j; D_j\right)\left(\widetilde{\beta}_{b-1} - \widetilde{\beta}_b\right) + U_b\left(\widetilde{\beta}_b; D_b\right) = 0. \tag{2.9}$$

Definendo per compattezza $\widetilde{J}_b = \sum_{j=1}^{b} J_j\left(\widetilde{\beta}_j; D_j\right)$, si può quindi risolvere l'equazione (2.9) utilizzando l'algoritmo ricorsivo, come in precedenza,

$$\widetilde{\beta}_2^{(r+1)} = \widetilde{\beta}_2^{(r)} + \left\{\widetilde{J}_{b-1} + J_b\left(\widetilde{\beta}_{b-1}; D_b\right)\right\}^{-1} U_b^{(r)}, \tag{2.10}$$

dove la quantità

$$U_b^{(r)} = \widetilde{J}_{b-1}\left(\widetilde{\beta}_{b-1} - \widetilde{\beta}_b^{(r)}\right) + U_b\left(\widetilde{\beta}_b^{(r)}; D_b\right) \tag{2.11}$$

è aggiornata ad ogni iterazione.

Da quanto descritto si può notare dunque che la differenza maggiore tra la *Spark lambda architecture* e la *rho architecture* è una modifica sostanziale dello *Speed layer*, livello relativo alla stima dei parametri. Infatti la *rho architecture* modifica tale livello andando ad aggiungere l'*Inference layer*, al cui interno si calcola la matrice di informazione osservata che, per quanto riguarda i GLM con funzione di legame canonica,

corrisponde all'inverso della matrice di varianza/covarianza. Tale modifica dei livelli si può vedere confrontando lo *Speed layer* in Figura 2.1 con il nuovo sistema di livelli composto da *Speed layer* e *Inference layer* in Figura 2.2.

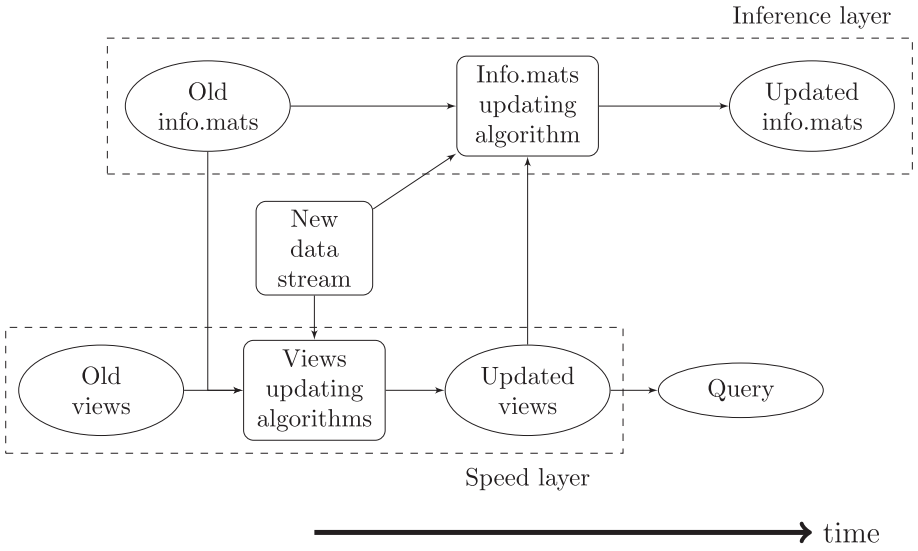

FIGURA 2.2: Diagramma indicante i vari *layer* presenti nella *rho architechture* (Luo & Song, 2020).

2.1.2 Pseudocodice

Per l'implementazione può risultare utile seguire lo pseudocodice presente nell'Algoritmo 3 e lo schema in figura 2.3 in cui vengono divise le operazioni che avvengono nello *Speed layer* da quelle che avvengono nell'*Inference layer*.

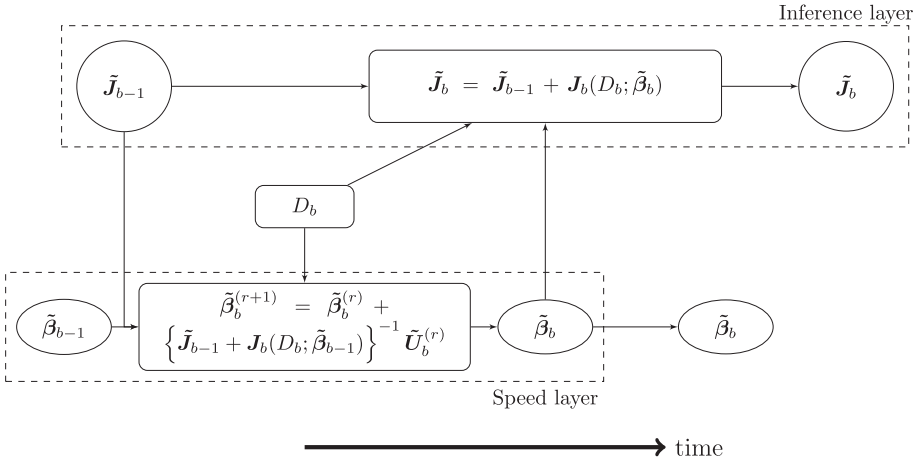

FIGURA 2.3: Diagramma della *rho architechture* in cui $\widetilde{\beta}_{b-1}$ è aggiornato nello *Speed layer* e $\widetilde{J}_{b-1}$ nell'*Inference layer*.

Algoritmo 3 *Rho architecture*

Input: modello $p(y; X, \beta_0)$, *batch* di dati $D_1, \ldots, D_b$
Output: $\widetilde{\beta}_b$ e $\widetilde{V}(\widetilde{\beta}_b) = \widetilde{J}_b^{-1}$, per $b = 1, 2, \ldots$
Inizializzazione: inizializzare i valori $\widetilde{\beta}_{init}$ e $\widetilde{J}_0 = 0_{p \times p}$
for $b = 1, 2, \ldots$ **do**
  leggere i dati nel *chunk* D_b
  nell'*Inference layer*, fare la decomposizione di Cholesky di $\left\{ \widetilde{J}_{b-1} + J_b \left(\widetilde{\beta}_{b-1}; D_b \right) \right\}$ e salvare la risultante fattorizzazione
  nello *Speed layer*, con $\widetilde{\beta}_b^{(1)} = \widetilde{\beta}_{b-1}$, usare la fattorizzazione ottenuta in precedenza per iterare:

$$\widetilde{\beta}_b^{(r+1)} = \widetilde{\beta}_b^{(r)} + \left\{ \widetilde{J}_{b-1} + J_b \left(\widetilde{\beta}_{b-1}; D_b \right) \right\}^{-1} \left\{ \widetilde{J}_{b-1} \left(\widetilde{\beta}_{b-1} - \widetilde{\beta}_b^{(r)} \right) + U_b \left(\widetilde{\beta}_b^{(r)}; D_b \right) \right\}$$

  fino a convergenza
  salvare $\widetilde{\beta}_b$ nello *Speed layer* e $\widetilde{J}_b$ nell'*Inference layer*
  rilasciare il *chunk* D_b dalla memoria
end for
Return $\widetilde{\beta}_b$ e $\widetilde{V} \left(\widetilde{\beta}_b \right)$, per $b = 1, 2, \ldots$

É inoltre utile notare come la funzione score di un GLM con funzione di legame canonica possa essere riscritta come

$$U(\beta; D) = \boldsymbol{X}^T (y - \mu) \tag{2.12}$$

Dato che nel caso di un modello logistico

$$\mu = \pi = \frac{\exp(\eta)}{1 + \exp(\eta)}$$

si ottiene

$$U(\beta) = \boldsymbol{X}^T \left(y - \frac{exp(\eta)}{1 + exp(\eta)} \right)$$

che per il singolo *chunk* risulta essere

$$U_b(\beta) = \sum_{i=1}^{n_b} \boldsymbol{x}_{bi} \left\{ y_{bi} - \frac{\exp\left(\boldsymbol{x}_{bi}^T \beta\right)}{1 + \exp\left(\boldsymbol{x}_{bi}^T \beta\right)} \right\},$$

dove $\boldsymbol{x}_{bi}^T$ è l'i-esima riga della matrice del modello del *chunk* D_b e y_{bi} è l'i-esima osservazione della variabile risposta del *chunk* D_b. Per quanto riguarda la matrice di osservazione osservata

$$J_b(\beta) = \sum_{i=1}^{n_b} \nu_{bi} \boldsymbol{x}_{bi} \boldsymbol{x}_{bi}^T$$

dove

$$\nu = \frac{d\,\pi\,(\eta)}{d\,\eta} = \frac{d}{d\,\eta}\,\frac{\exp\,(\eta)}{1+\exp\,(\eta)} = \frac{\exp\,(\eta)}{\{1+exp\,(\eta)\}^2} = \pi\,(1-\pi)\,.$$

Si ottiene quindi

$$J_b\,(\beta) = \sum_{i=1}^{n_b} \frac{\exp\left(\boldsymbol{x}_{bi}^T\beta\right)}{\{1+\exp\,(\boldsymbol{x}_{bi}^T\beta)\}^2}\boldsymbol{x}_{bi}\boldsymbol{x}_{bi}^T.$$

2.2 Aggiunta dei pesi

Fino a questo punto sono stati considerati modelli per dati binomiali dove si assume che le osservazioni siano state campionate indipendentemente da una distribuzione di riferimento uguale per tutti i *batch* di dati, condizionatamente al valore delle variabili esplicative. Sotto queste ipotesi viene naturale considerare tutti i dati ugualmente importanti e fare perciò in modo che il modello stimato dia lo stesso peso ad ogni osservazione disponibile. La *rho architecture* presentata precedentemente si basa proprio su questa assunzione e cerca di ottenere la stima di un GLM binomiale che utilizza tutte le osservazioni.

In alcune circostanze, può essere però conveniente pesare maggiormente certe osservazioni, aumentando il loro impatto sulla stima del modello. Nel caso dei dati di natura cestistica presi in esame in questa tesi, in particolare, stimare un modello come fatto precedentemente risulta equivalente ad analizzare il ripetersi di certi meccanismi nel corso dell'intera stagione, o almeno fino al b-esimo *batch*, considerando quindi la costanza di rendimento di giocatori e squadre durante l'intero intervallo temporale.

Le *batch* di dati relativi ad ogni partita arrivano però sequenzialmente ogni volta che tale partita è conclusa, quindi presentano un ordinamento temporale che sarebbe errato ignorare. Tale dipendenza temporale dei dati potrebbe essere modellata esplicitamente nella definizione del modello statistico. Una possibile alternativa a livello di modellizzazione consiste nel voler valutare il "momento" dei giocatori e delle squadre. Questa soluzione è già stata utilizzata da Dixon & Coles (1997) per l'analisi di dati relativi a partite di calcio. Per fare ciò non risulta di particolare interesse analizzare dati molto lontani nel tempo, in quanto nel caso si voglia prevedere il risultato di una partita interessa molto di più che squadra ha giocato meglio nell'ultimo mese piuttosto che, ad esempio, negli ultimi sei. Seguendo questa idea di poter modificare le stime considerando anche la distanza temporale tra le osservazioni, si è deciso di modificare il modello proposto precedentemente permettendo di aggiungere un peso per ogni osservazione. Dato che ci si trova ad analizzare dati in *batch*, per semplicità si assume che tale peso resti costante per tutte le osservazioni appartenenti al medesimo *batch*.

Nello specifico si è scelto di modificare la funzione di verosimiglianza specificata nell'equazione (1.2) del modello nel modo seguente:

$$L_b^* (\beta; X, y) \propto \prod_{j=1}^{b} L_j (\beta; X_j, y_j)^{w_b} \tag{2.13}$$

con

$$L_j (\beta; X_j, y_j) = \prod_{i=1}^{n_j} \left[\frac{exp (\eta_{ij})}{1 + exp (\eta_{ij})} \right]^{y_{ij}} \times [1 + exp (\eta_{ij})]^{y_{ij}-1}$$

dove w_b è il valore del peso per l'intero *chunk* D_b, che verrà successivamente specificato. È importante notare che $L_b^* (\beta; X, y)$ non è più una verosimiglianza propria, ma una verosimiglianza composita (Varin et al., 2011), che non possiede tutte le proprietà di $L_b (\beta; X, y)$. Sia $\ell_j (\beta; X_j, y_j)$ il logaritmo della funzione di verosimiglianza $L_j (\beta; X_j, y_j)$. Dopo aver così modificato la funzione di verosimiglianza, risulta evidente che il peso assegnato alle varie osservazioni ha un effetto moltiplicativo sulla funzione di log-verosimiglianza, nello specifico

$$\ell_b^* (\beta; X, y) = \sum_{j=1}^{b} w_j \ell_j (\beta; X_j, y_j) .$$

Tale effetto moltiplicativo si riflette a sua volta sulla funzione punteggio

$$U_b^* (\beta; X, y) = \sum_{j=1}^{b} w_j U_j (\beta; X_j, y_j)$$

che mantiene così la gradita caratteristica di essere a media nulla, in quanto si assume che $U_j (\beta; X, y)$ sia a media nulla per ogni j e che l'equazione di stima sia non distorta all'interno di ogni singolo *chunk*. A sua volta anche la matrice Hessiana negativa manterrà l'effetto moltiplicativo della funzione di peso, diventando

$$J_b^* (\beta; X, y) = \sum_{j=1}^{b} w_j J_j (\beta; X_j, y_j) .$$

Avendo definito tutti i tasselli utilizzati nella *rho architecture* si può procedere con la discussione di un algoritmo di aggiornamento ricorsivo per stime puntuali e variabilità per un modello logistico per dati pesati. Si procede quindi, in analogia con quanto fatto nel paragrafo 2.1.1 partendo dall'esempio con due soli *chunk* di dati.

$$U_1^* \left(\widetilde{\beta}_2; D_1 \right) + U_2^* \left(\widetilde{\beta}_2; D_2 \right) = w_1 U_1 \left(\widetilde{\beta}_2; D_1 \right) + w_2 U_2 \left(\widetilde{\beta}_2; D_2; \right) = 0 \tag{2.14}$$

Con l'espansione di Taylor lineare del primo termine dell'equazione (2.14) si ottiene

$$U_1^*\left(\widetilde{\beta}_1; D_1\right) + J_1^*\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2^*\left(\widetilde{\beta}_2; D_2\right) + \mathcal{O}_p\left(\|\widetilde{\beta}_2 - \widetilde{\beta}_1\|\right) = 0. \quad (2.15)$$

in cui, per le ipotesi precedentemente citate riguardo alla convergenza asintotica al vero valore del parametro, si può omettere il termine di errore $\mathcal{O}_p\left(\|\widetilde{\beta}_2 - \widetilde{\beta}_1\|\right)$ dall'equazione (2.15). Si può quindi determinare il nuovo valore della stima $\widetilde{\beta}_2$ risolvendo l'equazione

$$U_1^*\left(\widetilde{\beta}_1; D_1\right) + J_1^*\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2^*\left(\widetilde{\beta}_2; D_2\right) = 0. \quad (2.16)$$

Dato che $\widetilde{\beta}_1$ è soluzione dell'equazione di stima $U_1^*\left(\widetilde{\beta}_1; D_1\right) = 0$ il primo termine dell'equazione (2.16) può essere ignorato, portando a dover trovare la soluzione in $\widetilde{\beta}_2$ all'equazione

$$\begin{aligned} &J_1^*\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + U_2^*\left(\widetilde{\beta}_2; D_2\right) = \\ &= w_1 J_1\left(\widetilde{\beta}_1; D_1\right)\left(\widetilde{\beta}_1 - \widetilde{\beta}_2\right) + w_2 U_2\left(\widetilde{\beta}_2; D_2\right) = 0. \end{aligned} \quad (2.17)$$

Si deve infine generalizzare questa equazione di stima per passare dalla presenza di due soli *batch* ad un numero b generico di *batch*. il corrispettivo dell'equazione (2.9) diventa quindi:

$$\sum_{j=1}^{b-1} J_j^*\left(\widetilde{\beta}_j; D_j\right)\left(\widetilde{\beta}_{b-1} - \widetilde{\beta}_b\right) + U_b^*\left(\widetilde{\beta}_b; D_b\right) = 0. \quad (2.18)$$

Dopo aver definito $\widetilde{J}_b^* = \sum_{j=1}^{b-1} J_j^*\left(\widetilde{\beta}_j; D_j\right)$ si può risolvere l'equazione (2.18) tramite l'algoritmo ricorsivo

$$\begin{aligned} \widetilde{\beta}_2^{(r+1)} &= \widetilde{\beta}_2^{(r)} + \left\{\widetilde{J}_{b-1}^* + J_b^*\left(\widetilde{\beta}_{b-1}; D_b\right)\right\}^{-1} U_b^{*(r)} \\ &= \widetilde{\beta}_2^{(r)} + \left\{\widetilde{J}_{b-1}^* + w_b J_b\left(\widetilde{\beta}_{b-1}; D_b\right)\right\}^{-1} w_b U_b^{(r)} \end{aligned} \quad (2.19)$$

dove la funzione punteggio $U_b^{(r)}$ resta la stessa definita nell'equazione (2.11).

2.2.1 Scelta della funzione di peso

Dopo aver definito l'algoritmo ricorsivo per l'aggiornamento delle stime e della matrice Hessiana negativa in un modello binomiale con osservazioni pesate è necessario definire nello specifico come pesare le osservazioni su cui si adatta il modello. A seconda dei casi di studio la definizione di tale funzione può variare in funzione dell'interesse sottostante

alla scelta di tale modello. Per l'analisi di dati cestistici, essendo l'interesse principale capire quali squadre e quali giocatori stiano giocando meglio nell'ultimo periodo, si può considerare di pesare uniformemente un numero prefissato k di partite, ignorando completamente le partite precedenti. In tal modo, ponendosi nel caso di essere arrivati ad avere a disposizione D_b e di voler aggiornare $\widetilde{\beta}_{b-1}$ a $\widetilde{\beta}_b$, i pesi relativi al *batch* di dati D_j seguirebbero la seguente equazione

$$w_j = \begin{cases} 1 & t_b - t_j \leq k \\ 0 & t_b - t_j > k \end{cases} \qquad j = 1, \ldots, b \tag{2.20}$$

dove t_j è il momento di raccolta del *chunk* di dati D_j, $j = 1, \ldots, b$. La visualizzazione grafica di tale funzione di peso è in Figura 2.4. In questo modo è possibile modellare direttamente la distanza temporale tra i vari dati. Per semplicità, nel seguito è stato scelto di considerare come distanza temporale il numero di *batch* infrapposti tra D_b e D_j, ponendo quindi $t_j = j$, $j = 1, \ldots, b$.

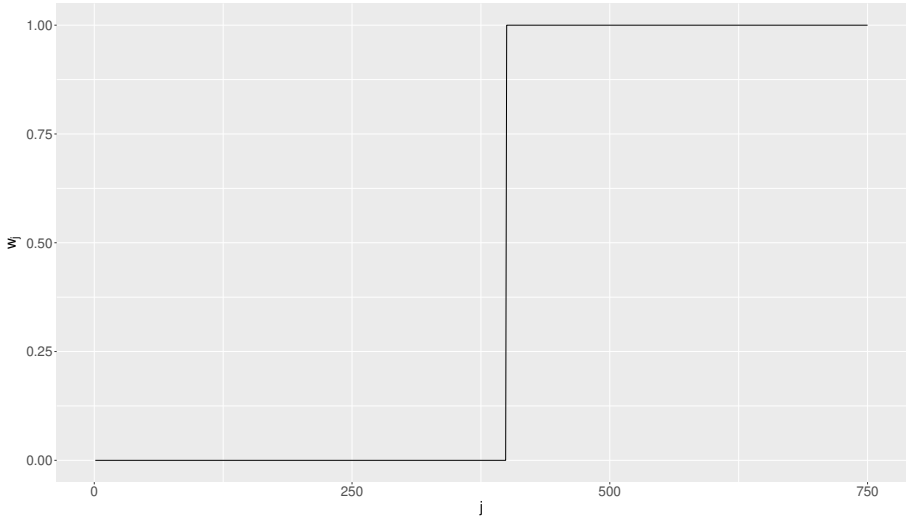

FIGURA 2.4: Grafico dell'andamento dei pesi per b fissato pari a 750 e $k = 350$.

Così facendo si perderebbe però interamente l'informazione legata ai precedenti dati, considerando solamente una finestra mobile in cui tutti i dati sono equipesati.

Un'alternativa è definire una funzione di peso esponenziale che decresca nel tempo per poter utilizzare tutti i dati senza perdita di informazione, pesando maggiormente quelli di maggior interesse, ossia quelli più recenti. Su questo tipo di funzione di peso è ricaduta la scelta per l'analisi di dati reali che sarà presentata nel prossimo capitolo. Dopo aver fissato un parametro di aumento esponenziale $\xi > 0$ si definisce

$$w_j = \exp\left\{\xi\left(t_j - t_1\right)\right\}, \qquad j = 1, \ldots, b. \tag{2.21}$$

L'andamento di questi pesi così definiti è illustrato in Figura 2.5.

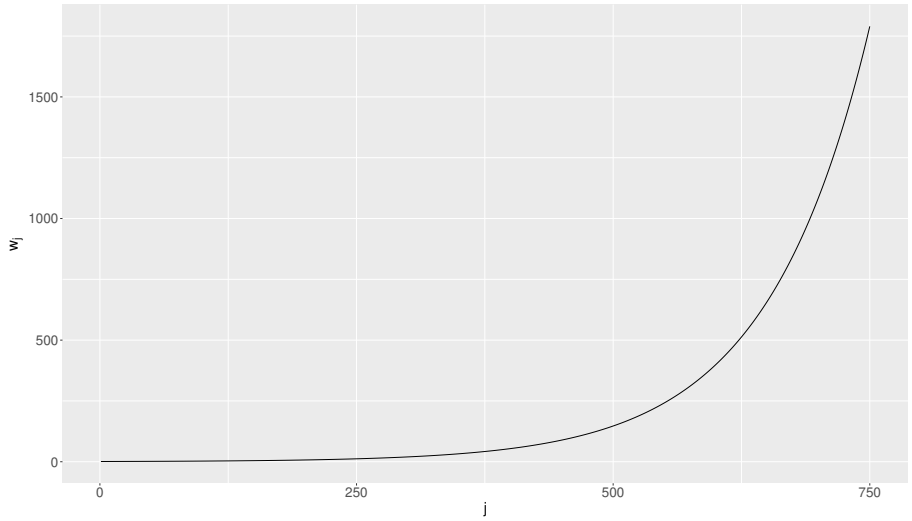

FIGURA 2.5: Grafico dell'andamento dei pesi esponanziali per b fissato pari a 750 e parametro $\xi = 0.01$.

Dato che la funzione esponenziale tende a divergere rapidamente, l'aumento dei pesi fino a valori particolarmente grandi può creare problemi, visto che il parametro di decadimento esponenziale ξ è tenuto sia a modellare quanto ripida sarà la curva, quindi quanto ampio è il periodo di interesse, sia quanto tali pesi tenderanno a divergere all'aumentare di b. Questo problema si nota infatti in particolare quando si hanno a disposizione molti *batch*, in quanto il valore dei pesi esploderà potendo portare a problemi di natura computazionale. Si è quindi deciso di valutare un'alternativa che mantenga le caratteristiche di questa funzione di peso andando però a risolvere il problema della divergenza dei pesi. È infatti sufficiente normalizzare tali pesi rispetto al valore massimo, assegnando all'ultimo *batch* di dati disponibile un peso pari a 1 e ai dati precedenti un peso decrescente. Nello specifico si avrà

$$\widetilde{w}_j = \frac{w_j}{w_b} = \frac{\exp\{\xi(t_j - t_1)\}}{\exp\{\xi(t_b - t_1)\}} = \exp\{-\xi(t_b - t_j)\} \tag{2.22}$$

per $j = 1, \ldots, b$. Tale funzione peso è mostrata in Figura 2.6 in cui si può notare che ha lo stesso andamento della funzione precedentemente definita. Avendo i pesi relativi alle osservazioni distanti nel tempo da t_b un valore particolarmente piccolo, si può inoltre fissare una soglia di distanza temporale k oltre la quale la funzione peso assume valore nullo. In questo caso si ha infatti

$$\widetilde{w}_j = \begin{cases} \exp\{-\xi(t_b - t_j)\} & t_b - t_j \leq k \\ 0 & t_b - t_j > k \end{cases}, \qquad j = 1, \ldots, b. \tag{2.23}$$

Così facendo si è riusciti a combinare le caratteristiche del sistema di pesi con soglia temporale e la funzione peso esponenziale (Figura 2.7). Si nota come troncare a zero i pesi relativi ai dati ad una distanza temporale superiore ad una certa soglia non porti a sostanziali differenze nella forma della funzione di peso.

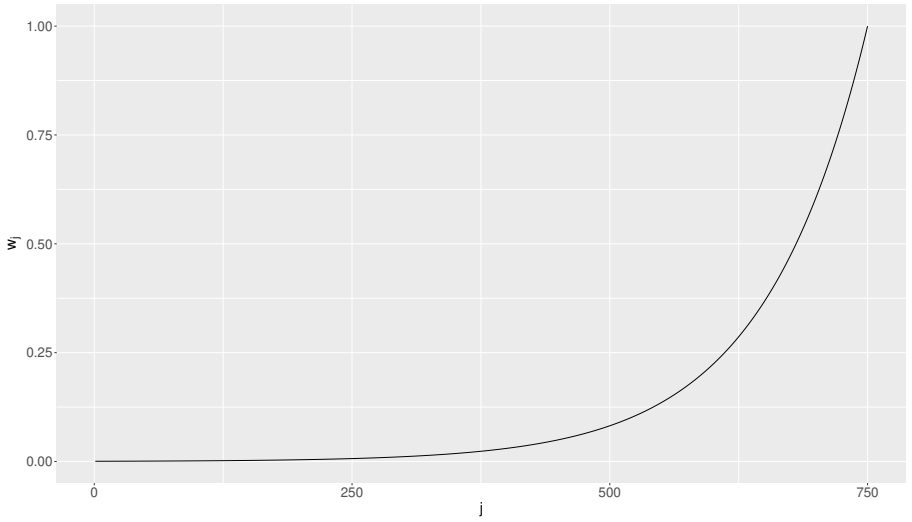

FIGURA 2.6: Grafico dell'andamento dei pesi esponanziali riscalati per b fissato pari a 750 e parametro di decadimento esponenziale $\xi = 0.01$.

Per risolvere il problema computazionale si è però andati ad aggiungere una necessità maggiore di statistiche riassuntive da salvare per l'aggiornamento delle stime, in quanto per definire i pesi nelle equazioni (2.22) e (2.23) è necessario conoscere il valore di b e di t_b, rendendo necessario ricalcolare i pesi delle osservazioni passate ogni volta che si rende disponibile un nuovo *batch* di dati. Per fortuna ciò non implica avere a disposizione l'intero *dataset* $\widetilde{D}_b$ all'istante temporale b, ma permette di salvare solamente la matrice Hessiana negativa $J\left(\widetilde{\beta}_j; D_j\right)$ e la statistica score $U_j(\widetilde{\beta}_j; D_j)$ degli ultimi k *batch* di dati $(j = b-k, \ldots, b)$.

2.2.2 Inferenza

Aggiungendo i pesi alle diverse osservazioni si è andati a modificare la verosimiglianza del modello binomiale, modificando le proprietà dello stimatore, in particolare quelle che si riferiscono alla varianza asintotica. Infatti, la verosimiglianza composita (2.13) si comporta, di fatto, come una verosimiglianza per un modello mispecificato. Si può mostrare (si veda ad esempio Pace et al., 2022, paragrafo 11.5.2) che, in tali casi, la distribuzione asintotica del corrispondente stimatore di massima verosimiglianza è

$$\widetilde{\beta}_b \dot{\sim} N\left(\beta_0, \widetilde{J}_b^*\left(\widetilde{\beta}_b\right)^{-1} H\left(\widetilde{\beta}_b\right) \widetilde{J}_b^*\left(\widetilde{\beta}_b\right)^{-1}\right)$$

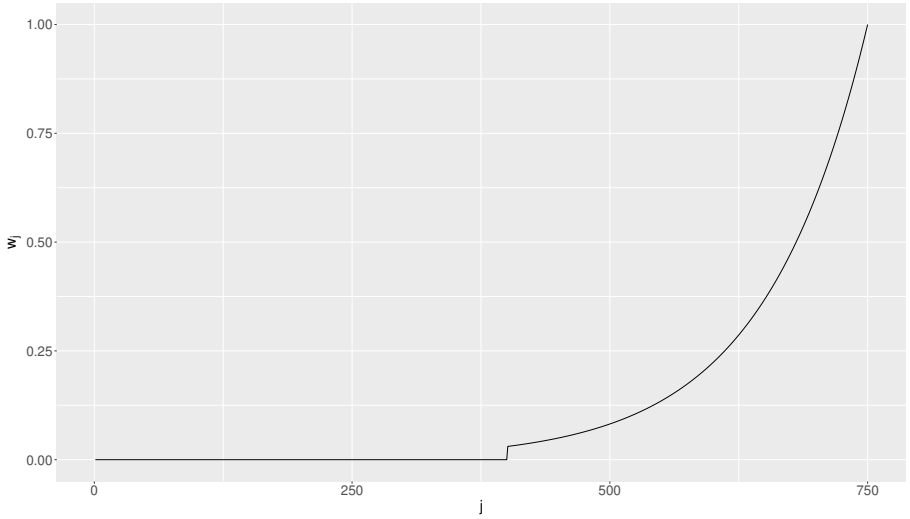

FIGURA 2.7: Grafico dell'andamento dei pesi esponanziali riscalati per b fissato pari a 750 e parametro di decadimento esponenziale $\xi = 0.01$. Si troncano a zero i pesi dei dati con una distanza temporale da b maggiore di $k = 350$.

dove

$$H\left(\widetilde{\beta}_b\right) = E\left[U^* U^{*T}\right].$$

Per calcolare la varianza asintotica dello stimatore $\widetilde{\beta}_b$ è quindi necessario utilizzare la formula della varianza *sandwich*.

Lo pseudocodice necessario per implementare l'aggiornamento ricorsivo pesato attraverso l'utilizzo della *rho architecture* modificata è presentato nell'Algoritmo 4.

2.3 Stime con riduzione della distorsione tramite aggiornamento della matrice QR

Dopo aver spiegato e ampliato l'algoritmo per l'aggiornamento ricorsivo di un modello di regressione per dati binari, si è provato a trovare un algoritmo in grado di aggiornare ricorsivamente le stime con distorsione ridotta di Firth per un modello binomiale, partendo da quanto proposto da Zietkiewicz & Kosmidis (2023). Sia

$$U(\beta) = X^T W D^{-1}(y - \mu) \tag{2.24}$$

la funzione di punteggio di un GLM, con $\mu = (\mu_1, \ldots, \mu_n)^T$, $W = \text{diag}\{\omega_1, \ldots, \omega_n\}$, $D = \text{diag}\{d_1, \ldots, d_n\}$ e $\omega_i = m_i d_i^2 / v_i$ il peso assegnato ad ogni osservazione, con $d_i = d\mu_i / d\eta_i$ e $v_i = V(\mu_i)$. Nel caso si utilizzi la funzione di legame canonica, come nel

Algoritmo 4 Rho architecture per osservazioni pesate

Input: modello $p(y; X, \beta_0)$, *batch* di dati $D_1, \ldots, D_b$
Output: $\widetilde{\beta}_b$ e $\widetilde{V}(\widetilde{\beta}_b) = \widetilde{J}_b^* \left(\widetilde{\beta}_b\right)^{-1} H\left(\widetilde{\beta}_b\right) \widetilde{J}_b^* \left(\widetilde{\beta}_b\right)^{-1}$, per $b = 1, 2, \ldots$
Inizializzazione: inizializzare i valori $\widetilde{\beta}_{init}$
for $b = 1, 2, \ldots$ **do**
    leggere i dati nel *chunk* D_b
    calcolare i pesi w_j, $j = 1, \ldots, b$
    calcolare $\widetilde{J}_{b-1}^* = \sum_{j=1}^{b-1} w_j J_j$ e $\widetilde{U}_{b-1}^* = \sum_{j=1}^{b-1} w_j U_j$
    nell'*Inference layer*, fare la decomposizione di Cholesky di $\left\{\widetilde{J}_{b-1}^* + w_b \widetilde{J}_b\left(\widetilde{\beta}_{b-1}; D_b\right)\right\}$ e salvare la risultante fattorizzazione
    nello *Speed layer*, con $\widetilde{\beta}_b^{(1)} = \widetilde{\beta}_{b-1}$, usare la fattorizzazione ottenuta in precedenza per iterare:

$$\widetilde{\beta}_b^{(r+1)} = \widetilde{\beta}_b^{(r)} + \left\{\widetilde{J}_{b-1}^* + w_b J_b\left(\widetilde{\beta}_{b-1}; D_b\right)\right\}^{-1} \left\{\widetilde{J}_{b-1}^*\left(\widetilde{\beta}_{b-1} - \widetilde{\beta}_b^{(r)}\right) + w_b U_b\left(\widetilde{\beta}_b^{(r)}; D_b\right)\right\}$$

    fino a convergenza
    salvare $\widetilde{\beta}_b$ nello *Speed layer* e $\widetilde{J}_b^*$ e $\widetilde{U_b^*}$ nell'*Inference layer*
    rilasciare il *chunk* D_b dalla memoria
end for
Return $\widetilde{\beta}_b$ e $\widetilde{V}\left(\widetilde{\beta}_b\right)$, per $b = 1, 2, \ldots$

modello logistico, l'equazione (2.24) si semplifica diventando equivalente all'equazione (2.12).

Immaginiamo ora di voler regredire $Y \in \mathbb{R}^n$ su $X \in \mathbb{R}^{n \times p}$ con pesi noti $W \in \mathbb{R}^{n \times n}$ utilizzando la decomposizione QR

$$W^{\frac{1}{2}} X = [Q_1 \quad Q_2] \begin{bmatrix} R_1 \\ 0_{(n-1)\times p} \end{bmatrix} \tag{2.25}$$

dove $Q_1 \in \mathbb{R}^{n \times p}$, $Q_2 \in \mathbb{R}^{n \times (n-p)}$, R_1 è una matrice triangolare superiore $p \times p$ e $0_{(n-1)\times p}$ è una matrice $(n-p) \times p$ di zeri. La soluzione ai minimi quadrati $\hat{\beta} \in \mathbb{R}^p$ si ottiene risolvendo in $\hat{\beta}$ il sistema

$$Q_1^T W^{\frac{1}{2}} Y = R_1 \hat{\beta}. \tag{2.26}$$

Il modello con la correzione della distorsione proposto da Firth (1993) utilizza come equazione di stima

$$U(\beta) + X^T W \xi = 0_p$$

dove $\xi = (\xi_1, \ldots, \xi_n)^T$ e $\xi_i = h_i d_i' / (2 d_i \omega_i)$, $d_i' = d^2 \mu_i / d\eta_i^2$ e h_i è l'*hat value* dell'osservazione i-esima ottenuto dall'i-esimo valore sulla diagonale della matrice $H =$

$X\left(X^TWX\right)^{-1}X^TW$.

In tal caso la stima per il parametro con distorsione ridotta si ottiene iterando

$$\beta^{(j+1)} = \left(X^TW^{(j)}X\right)^{-1}X^TW^{(j)}\left(z^{(j)} + \xi^{(j)}\right) \tag{2.27}$$

con $z^{(j)} = \left(z_1^{(j)}, \ldots, z_n^{(j)}\right)$ e $z_i^{(j)} = \eta_i^{(j)} + \left(y_i - \mu^{(j)}\right)(d_i^{(j)})$.

All'iterazione j-esima si può sostituire $H^{(j)}$ con $H^{(j-1)} = Q^{(j-1)}Q^{T(j-1)}$ in modo da avere $\xi_i^{*(j)} = h_i^{(j-1)}d_i^{'(j)}/\left(2d_i^{(j)}\omega_i^{(j)}\right)$. Dato che $j/(j-1) \to 1$ quando $j \to \infty$ tale algoritmo ha lo stesso punto stazionario dell'algoritmo per i minimi quadrati pesati come dimostrato da Kosmidis et al. (2020).

Dato che nessuna delle quantità utilizzate richiede di aver caricato in memoria l'intero *dataset*, in quanto la decomposizione QR può essere aggiornata con *batch* di dati tramite l'algoritmo AS 274 proposto da Miller (1992) e Miller (1994), si può aggiornare l'equazione di stima ricorsiva (2.27) con

$$\beta^{(j+1)} = \left(X^TW^{(j)}X\right)^{-1}X^TW^{(j)}\left(z^{(j)} + \xi^{*(j)}\right) \tag{2.28}$$

rendendo possibile la stima a distorsione ridotta di GLM su *dataset* di qualsiasi dimensione.

Purtroppo, nonostante si sia riusciti ad aggiornare la stima utilizzando di volta in volta solo parte dei dati, essendo una stima ricorsiva ad ogni iterazione generica j, è necessario avere l'intero *dataset* a disposizione. In questo modo, si riesce quindi a stimare GLM anche se i dati non riescono ad essere salvati nella memoria disponibile per l'esecuzione di operazioni, ma bisogna comunque avere accesso all'intero insieme di dati, seppur esso disponibile in un *database* esterno. Non si è quindi riusciti ad utilizzare questo approccio ricorsivo per la stima di modelli per dati binomiali su dati che arrivano sequenzialmente in *chunk*, senza la necessità di avere tutti i dati a disposizione. È tuttavia possibile utilizzarlo per l'aggiornamento delle stime di modelli lineari, in quanto questi ultimi hanno una soluzione in forma chiusa, e non richiedono quindi un processo iterativo per la stima.

Può risultare interessante porsi l'obiettivo di ampliare tale metodologia per permettere di utilizzarla anche nel processo di aggiornamento delle stime di modelli lineari generalizzati, eliminando la necessità di avere a disposizione l'intero *dataset*.

Capitolo 3

Applicazione a dati NBA

Dopo aver presentato diversi metodi per l'aggiornamento ricorsivo delle stime, in questo capitolo è stata effettuata un'analisi di dati reali per dimostrare l'efficacia di tali metodologie nel cogliere meccanismi sottostanti a determinati fenomeni. Nello specifico, sono stati analizzati dati sportivi relativi alla pallacanestro con l'obiettivo di valutare l'importanza di diverse giocate e del giocatore che effettua tali giocate, oltre alla sua squadra di appartenenza, sulla probabilità di realizzazione di un canestro al termine dell'azione di cui fa parte la giocata, con la possibilità di presentare i risultati dell'analisi subito dopo il termine di ogni partita aggiornando le stime già ottenute con i dati delle partite precedenti.

3.1 I dati

Il sito Big Data Ball (2019) da cui sono stati acquistati i dati utilizzati in questa tesi fornisce grandi quantità di dati riguardanti lo sport e permette, insieme ad altri siti, di ottenere *dataset* contenenti informazioni sia sulle statistiche aggregate delle squadre appartenenti a diverse leghe, sia statistiche individuali dei giocatori e dati grezzi sulle partite non ancora processati. Di quest'ultima categoria fa parte il *dataset* contenente tutte le azioni *play by play* della stagione NBA 2018-2019 utilizzato per le analisi che verranno presentate in questo capitolo.

Tale *dataset* grezzo è composto da 621527 osservazioni per ciascuna delle quali sono state raccolte 44 variabili. Per ogni azione di gioco si hanno infatti a disposizione:

- *game_id*: identificativo della partita;
- *dataset*: indica se la partita è della stagione regolare o dei *playoff*;

- *date*: data della partita;
- *a1*, *a2*, *a3*, *a4*, *a5*: cinque variabili che indicano che giocatori sono in campo per la squadra che gioca fuori casa;
- *h1*, *h2*, *h3*, *h4*, *h5*: cinque variabili che indicano che giocatori sono in campo per la squadra che gioca in casa;
- *period*: quarto di gioco in cui si svolge l'azione;
- *away_score*: punteggio della squadra che gioca in trasferta;
- *home_score*: punteggio della squadra che gioca in casa;
- *remaining_time*: tempo rimanente nel quarto di riferimento;
- *elapsed*: tempo giocato nel quarto di riferimento;
- *play_length*: durata della giocata di riferimento;
- *play_id*: codice identificativo della giocata;
- *team*: squadra d'appartenenza del giocatore che ha effettuato la giocata;
- *event_type*: tipo di giocata effettuata (passaggio, tiro libero, palla persa, ...)
- *assist*: qualora la riga si riferisca ad un tiro segnato da un giocatore a cui è stato servito un *assist*, quindi che giocatore ha effettuato l'*assist*;
- *away*: squadra che gioca fuori casa;
- *home*: squadra che gioca in casa;
- *block*: qualora la riga di riferimento si riferisca ad un tiro che è stato stoppato, indica che giocatore ha stoppato il tiro;
- *entered*: giocatore che entra in campo in un evento di sostituzione;
- *left*: giocatore che esce in campo in un evento di sostituzione;
- *num*: numero di tiri liberi segnati nella singola volta in lunetta;
- *outof*: numero di tiri liberi tentati nella singola volta in lunetta;
- *opponent*: giocatore che ha subito un fallo;
- *player*: giocatore che ha effettuato la giocata;

- *points*: punti segnati nell'evento;
- *possession*: giocatore che entra in possesso della palla successiamente all'evento;
- *reason*: ulteriori dettagli sull'evento di riferimento;
- *result*: qualora la giocata di riferimento dia un tiro, indica se il tiro è entrato oppure è uscito;
- *steal*: qualora la giocata di riferimento sia una palla persa, indica se la palla è stata rubata da un avversario;
- *type*: ulteriori caratteristiche dell'evento;
- *shot_distance*: qualora la giocata di riferimento sia un tiro, indica la distanza dalla quale è stato effettuato;
- *original_x*, *original_y*, *converted_x*, *converted_y*: variabili geospaziali che indicano la posizione da cui è stato effettuato un tiro a canestro.

La codifica delle variabili è abbastanza intuitiva, eccezion fatta per la variabile *team* in cui sono tutte le squadre presenti sono codificate con la relativa abbreviazione e per *event_type* in cui le varie giocate sono presentate in lingua inglese. Per chiarezza nei risultati che verranno presentati è necessario introdurre le abbreviazioni utilizzate nella variabile *team* in Tabella 3.1 e i tipi di evento possibili con la loro descrizione in Tabella 3.2. In Tabella 3.3 viene inoltre mostrata la distribuzione di frequenza incondizionata della variabile *event_type*, con lo scopo di aiutare a capire la composizione delle varie azioni all'interno dell'intero *dataset*.

Tabella 3.1: Abbreviazioni dei nomi delle squadre.

	squadra		squadra
ATL	*Atlanta Hawks*	MIL	*Milwaukee Bucks*
BOS	*Boston Celtics*	MIN	*Minnesota Timberwolves*
CHA	*Charlotte Hornets*	NOP	*New Orleans Pelicans*
CHI	*Chicago Bulls*	NYK	*New York Knicks*
CLE	*Cleveland Cavaliers*	BKN	*Brooklyn Nets*
DAL	*Dallas Mavericks*	OKC	*Oklahoma City Thunder*
DEN	*Denver Nuggets*	ORL	*Orlando Magic*
DET	*Detroit Pistons*	PHI	*Philadelphia 76ers*
GSW	*Golden State Warriors*	PHO	*Phoenix Suns*
HOU	*Houston Rockets*	POR	*Portland Trail Blazers*
IND	*Indiana Pacers*	SAC	*Sacramento Kings*
LAC	*Los Angeles Clippers*	SAS	*San Antonio Spurs*
LAL	*Los Angeles Lakers*	TOR	*Toronto Raptors*
MEM	*Memphis Grizzlies*	UTH	*Utah Jazz*
MIA	*Miami Heat*	WAS	*Washington Wizards*

Tabella 3.2: Tipi di possibili giocate.

	Giocata
block	Stoppata
foul	Fallo commesso
free throw	Tiro libero
jump ball	Palla contesa
rebound	Rimbalzo, sia esso offensivo o difensivo
shot	Tiro a canestro
sub	Sostituzione
turnover	Palla persa
violation	Violazione

Tabella 3.3: Distribuzione di frequenza della variabile *event_type*.

Giocata	n
block	13019
foul	56422
free throw	60811
jump ball	2126
rebound	118513
shot	233734
sub	61080
turnover	35382
violation	1775

3.1.1 Pretrattamento dei dai

Per com'è codificato, il *dataset* contenente i dati grezzi non è utilizzabile per l'analisi e necessita di una fase di *preprocessing* importante per ricodificare le variabili presenti e correggere diversi errori nella raccolta dei dati. Considerando i valori mancanti, si nota come quasi la totalità delle righe ne presenta almeno uno e alcune variabili ne contengono quantità particolarmente elevate come si può notare in Figura 3.1.

FIGURA 3.1: Percentuale di valori mancanti per tutte le variabili che ne presentano almeno uno.

Inizialmente è stato necessario ricodificare la variabile *game_id* in modo che ci fosse una coerenza tra le diverse partite e che il codice identificativo avesse la stessa lunghezza in ognuna di esse. Successivamente si sono corretti degli errori nella variabile *data_set*, dove sia la modalità *Regular Season* che *Playoff* erano codificate diversamente in diverse righe. Entrambe le variabili *elapsed* e *remaining_time* sono state ricodificate in secondi ed essendo strettamente collegate, in quanto indicanti il tempo giocato e rimanente all'interno di un periodo di gioco, hanno necessitato di un controllo aggiuntivo, ovvero di verificare che in ogni momento della partita la loro somma fosse pari a 720 secondi, che corrispondono ai 12 minuti di durata di un periodo. Tale controllo è stato fatto per tutti i periodi appartenenti ai tempi regolamentari dato che ogni partita è composta di 4 quarti da 12 minuti ciascuno, mentre per i tempi supplementari è stato verificato che la somma di queste due variabili fosse pari a 300 secondi, ovvero 5 minuti. Si è notato inoltre che a tutte le azioni che si riferivano ad una palla contesa a inizio partita o a inizio dei tempi supplementari era associato un valore di *play_length* negativo, quindi è stato necessario porre tale valore a 0, in quanto una palla contesa è istantanea e non impiega alcun secondo di gioco per essere efettuata.

Per la variabile *block* relativa ad una possibile stoppata nel momento in cui è stato effettuato un tiro è stato necessario un passaggio ad una codifica diversa. Essendo infatti una stoppata una giocata a sè effettuata da un giocatore, non sarebbe stato corretto che essa fosse presente come variabile aggiuntiva in una riga che fa riferimento ad una giocata differente. Per tale motivo si è deciso di aggiungere una riga per ogni stoppata, posizionata da un punto di vista temporale immediatamente dopo la riga che codifica il tiro che è stato bloccato. A tale giocata appena aggiunta si è quindi associato un valore di *play_length* pari a 1 e un livello di *event_type* pari a *"block"*, oltre a mantenere ed aggiustare le caratteristiche del tempo di gioco effettivo e del giocatore che ha effettuato la stoppata. Al termine di questa operazione di aggiunta delle righe è stata eliminata la variabile *block*, in quanto tutta l'informazione contenuta in essa è stata codificata diversamente, in una modalità consona all'analisi.

Sono state quindi eliminate le variabili *home*, *away*, *possession*, *play_id*, *entered*, *left*, *opponent*, *possession*, *reason* e *description*. Come si può notare gran parte delle variabili eliminate sono quelle in cui la percentuale di valori non disponibili era eccessivamente elevata, il che impediva di poter ricostruire il dato reale. Le variabili *reason* e *description* erano invece inutilizzabili al fine dell'analisi, in quanto descrizioni delle giocate fatte con il fine di essere lette da una persona, quindi variabili fattoriali con un numero di livelli particolarmente alto. Un discorso diverso è stato fatto per la variabile *assist*, anch'essa rimossa, che a differenza della variabile *block* non è stata codificata in una nuova riga pur essendo una giocata diversa dal tiro e codificata nella stessa riga. Questa scelta è stata dettata dal fatto che un assist è per definizione un passaggio di un giocatore che permette ad un compagno di segnare un canestro subito dopo, quindi se tale passaggio fosse stato codificato in una riga a sè stante sarebbe risultato sempre significativo al fine di realizzare un canestro, portando ad una perfetta separazione nei dati e quindi ad un problema di stima che non avrebbe permesso all'algoritmo di stima dei GLM di arrivare a convergenza. Sarebbe inoltre stato inutile al fine dell'interpretazione in quanto, fosse anche stata possibile la stima del parametro, non si avrebbe avuto informazione aggiuntiva rispetto a quanto già noto a priori. Come ultimo passaggio sono state rimosse le variabili *elapsed* (la cui informazione è già presente in *remaining_time*) e la variabile *play_id*, che indica il codice identificativo della singola giocata.

Si è proceduto successivamente a costruire la lista completa dei giocatori che hanno giocato almeno una partita prendendo tutti i valori unici delle variabili *a1*, *a2*, *a3*, *a4*, *a5*, *h1*, *h2*, *h3*, *h4*, *h5*, *player*, ossia tutte le variabili presenti nel *dataset* che si riferiscono ai giocatori.

3.1.2 Creazione della variabile risposta

La variabile di interesse per l'analisi è *result*, che indica se in un'azione di tiro è stato effettuato o meno un canestro. Per sua natura tale variabile risulta codificata esclusivamente nelle azioni di tiro, mentre non è disponibile per tutte le altre azioni presenti nel *dataset*. In Tabella 3.4 si può notare la distribuzione di frequenza di tale variabile e l'elevato numero di valori non disponibili. Nella codifica di questa variabile 1 indica la riuscita realizzazione di un canestro mentre 0 indica un errore.

Tabella 3.4: Distribuzione di frequenza della variabile *result*.

result	n
1	154045
0	140500
NA	288317

L'interesse dell'analisi non è però quello di studiare la probabilità di un tiro di risultare in un canestro o in un errore, ma quello di valutare quali caratteristiche delle varie giocate all'interno di un'azione offensiva portino a modificare la probabilità di segnare un tiro a fine azione. Per tale motivo sembrerebbe logico assegnare valore 1 alla variabile risposta per tutte le righe che si riferiscono ad un'azione che si conclude con un canestro. Bisogna però anche considerare che non tutte le giocate che compongono un'azione hanno importanza al fine ultimo della realizzazione di un canestro. Si può pensare per esempio ad un passaggio dalla rimessa all'inizio azione in cui dopo aver fatto uno schema che prevede altri 5 passaggi viene segnato un canestro a 5 secondi dal termine del tempo disponibile, dove la durata massima di un'azione nel basket è di 24 secondi, quindi dal passaggio iniziale alla realizzazione del canestro sono passati 19 secondi. In tal caso si può dire che il passaggio iniziale non è stato significativo ai fini della realizzazione del canestro. È quindi necessario trovare una regola per discriminare quali giocate ritenere importanti e quali invece no.

Per l'analisi di dati relativi a partite di calcio Dandolo (2019) propone di introdurre un parametro K di cui bisogna fissare un valore e di considerare come importanti le K azioni immediatamente precedenti alla realizzazione di un gol. Partendo da questa idea si è pensato di non considerare un numero fisso di giocate come importanti, ma di considerare un intervallo temporale entro cui valutare l'importanza di una giocata. Si è quindi introdotto il parametro T che indica la distanza temporale massima dalla realizzazione di un tiro entro cui considerare una giocata come importante. Il valore di T è stato fissato a 12, in quanto si reputa che metà della durata massima di un'azione sia un valore adeguato per discriminare se una giocata è stata significativa al fine del risultato

del tiro a canestro oppure no. Ovviamente nella valutazione della distanza temporale di una giocata dal tiro viene considerata solo l'azione di una squadra, e tale distanza viene resettata ad ogni cambio possesso, sia che l'azione precedente sia terminata con un tiro sia che essa sia terminata con un altro evento, per esempio con una palla persa.

3.1.3 Matrice del modello

Dopo aver costruito la variabile risposta y, guardando la sua distribuzione marginale in Tabella 3.5 si nota come tutte le osservazioni per cui prima non era disponibile un valore della variabile *result* risultano correttamente classificate come 0 o 1, dove questi valori assumono lo stesso significato che avevano per la variabile *result*.

Tabella 3.5: Distribuzione di frequenza della variabile risposta y.

result	n
1	196199
0	386663

Avendo quindi a disposizione la variabile risposta per ogni osservazione presente nel *dataset* si può procedere con la costruzione della matrice del modello da utilizzare nel modello logistico e dai vari metodi di aggiornamento ricorsivo delle stime presentati nel Capitolo 2. Per la costruzione della stessa è stato deciso di utilizzare la libreria R `fastDummies` (Kaplan, 2020) che permette di ricodificare le variabili fattoriali in opportune variabili *dummy* corrispondenti eliminando in automatico una colonna in modo da rendere le colonne della matrice risultante linearmente indipendenti. Ciò implica che nei modelli ci sia un valore di riferimento per tutte le variabili fattoriali.

Per fare in modo che tale matrice non presentasse un numero di colonne p eccessivamente elevato e per facilitare le operazioni di stima si è deciso di modificare la variabile *player* in modo tale da non considerare tutti i giocatori che sono entrati in campo almeno una volta in tutta la stagione (ben 852), ma solamente i 339 giocatori che hanno partecipato ad un minimo di 100 azioni nel corso di tutta la stagione e ad almeno 20 azioni nel corso delle prime 100 partite di regular season, indipendentemente dalla squadra di appartenenza. Sono state comunque mantenute nel *dataset* anche le azioni che fanno riferimento ai giocatori non presi in considerazione, sostituendo in tal caso il valore della variabile *player* con *"General"*. Questo accorgimento è stato fondamentale per permettere ai modelli iniziali di non incorrere in problemi di stime infinite dovuti alla perfetta separabilità dei dati. Infatti ci sono giocatori che hanno partecipato ad una sola azione che avrebbero portato ad una combinazione di covariate tale da poter determinare univocamente l'esito della stessa solo sapendo se essi vi avevano partecipato.

Le variabili scelte per essere inserite nella matrice del modello sono state quindi:

- *away_score*;
- *home_score*;
- *remaining_time*;
- *period* per cui il periodo di riferimento (corrispondente alla colonna eliminata) è il primo;
- *team*, *home* e *away* per cui il valore di riferimento è ATL;
- *player* per cui il valore di riferimento è *"Aaron Gordon"*;
- *event_type* per cui il valore di riferimento è *block*.

Si può notare come, pur essendo il numero di variabili inserite nella matrice del modello molto inferiore al numero di variabili iniziali, si sia riusciti a mantenere gran parte dell'informazione contenuta nei dai grezzi, eccezion fatta per le variabili geospaziali indicanti la posizione da cui è stato effettuato il tiro a canestro, non reperibili per tutte le azioni che non si sono concluse con un tiro, e delle variabili dei giocatori presenti in campo nel momento in cui è stata effettuata ogni giocata, che oltre ad aumentare di molto la dimensione della matrice del modello sono ritenute poco interessanti al fine dell'analisi in quanto tale informazione è stata parzialmente gestita tramite la costruzione della variabile risposta.

Dopo aver terminato le operazioni di preparazione dei dati si è quindi ottenuta una matrice del modello composta da 582864 righe e 442 colonne. A questo punto per poter applicare le tecniche di analisi presentate in modo realistico la matrice è stata divisa in *batch* di dati che sono stati salvati sul disco fisso, in modo tale da poter accedere sequenzialmente ad essi senza avere la possibilità di averli disponibili in memoria tutti nello stesso momento. Per la divisione è stata considerata la variabile *game_id*, stratificando i dati per ogni suo livello in modo tale che ogni *batch* contenesse le azioni di una sola partita e che tutti i *batch* fossero ordinati temporalmente, partendo dalla prima partita della stagione fino ad arrivare all'ultima. In tal modo si è riusciti a creare una struttura che rispettasse i requisiti iniziali dell'analisi presentata. Tutte le operazioni di *preprocessing* sono state fatte sul *dataset* completo per semplicità. Tuttavia si possono replicare all'arrivo di un ulteriore *batch* di dati ripercorrendo tutti i passi e salvando in memoria esclusivamente la lista di giocatori considerati nella matrice del modello, senza necessità di avere a disposizione tutti i dati.

3.2 Analisi di dati indipendenti

In prima analisi i dati relativi alle singole giocate presenti in ogni *chunk* verranno considerati indipendenti, in modo da soddisfare le ipotesi sottostanti al modello logistico. Tale assunzione è chiaramente non verificata, sia per la natura dei dati che presentano un ordinamento temporale, sia per la costruzione della variabile risposta, per cui ci si è basati sulle informazioni contenute in righe diverse. Infatti assegnando valore unitario a diverse righe precedenti ad un tiro segnato si aggiunge dipendenza ai dati. Tuttavia, si è reputato che sia possibile fare tale assunzione in quanto ogni giocatore, squadra e possibile tipo di giocata può avere lo stesso effetto sulla probabilità di segnare un canestro a fine azione, indipendentemente dall'ordinamento temporale delle stesse.

3.2.1 Modello logistico

Ripercorrendo quanto introdotto nel Capitolo 2 il primo modello che viene presentato per l'analisi iniziale dei dati è il modello binomiale con funzione di legame logistica. Tale modello viene adattato ad un *dataset* iniziale $\widetilde{D}_{B_1}$ di $B_1 = 100$ partite. La dimensione di tale *batch* iniziale è stata scelta in modo tale che fosse abbastanza piccola da poter cominciare l'analisi delle partite in un momento quanto più possibile vicino all'inizio della stagione regolare, tenendo conto però che esso avesse un dimensione sufficientemente grande per poter ottenere dei risultati attendibili dal modello stimato inizialmente. Essendo presenti 30 squadre, con 100 partite totali si considerano circa 6-7 partite per squadra, che porta a poter stimare il modello binomiale iniziale.

I risultati presentati in Tabella 3.6 si riferiscono ai parametri del modello binomiale con funzione di legame logistica ottenuti per i giocatori che sono risultati i migliori dal punto di vista della partecipazione ad azioni che portano ad un canestro per l'intera durata della stagione, ovvero per tali giocatori che dopo aver analizzato l'intero *dataset* hanno ottenuto una stima maggiore dei coefficienti del modello binomiale dopo aver analizzato tutte le 1312 partite disponibili.

Si può vedere come in tale lista non è presente nessuno dei giocatori più noti e maggiormente valutati negli ultimi anni. Approfondendo ulteriormente si nota come la maggior parte dei giocatori presenti in tale tabella sono comprimari che hanno attraversato una delle loro migliori stagioni dal punto di vista realizzativo e dei rimbalzi, pur giocando spesso una media di minuti a partita non di molto superiore ai 20, quindi nettamente inferiore ai giocatori più rappresentativi della lega. Tra questi giocatori spiccano comunque nomi come Deandre Ayton, inserito nella stagione di riferimento nell'*NBA All-Rookie First Team*, ovvero la lista dei migliori cinque giocatori alla loro

TABELLA 3.6: Coefficienti del modello logistico aggiornato tramite la *rho architecture* per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.288	0.052	0.000
Thomas Bryant	0.447	0.070	0.000
Deandre Ayton	0.419	0.061	0.000
Dwight Powell	0.410	0.068	0.000
Domantas Sabonis	0.408	0.077	0.000
Ekpe Udoh	0.401	0.151	0.008
Boban Marjanovic	0.387	0.078	0.000
Cheick Diallo	0.384	0.092	0.000
Montrezl Harrell	0.383	0.058	0.000
Enes Kanter	0.370	0.057	0.000
Jarrett Allen	0.363	0.066	0.000
T.J. McConnell	0.356	0.081	0.000
Derrick Favors	0.347	0.074	0.000
Steven Adams	0.346	0.073	0.000
Jakob Poeltl	0.346	0.079	0.000
Robin Lopez	0.343	0.072	0.000
Jonas Valanciunas	0.341	0.066	0.000
Mitchell Robinson	0.340	0.074	0.000
Nerlens Noel	0.335	0.091	0.000

prima stagione NBA, Domantas Sabonis, talentuoso giocatore lituano che dopo aver disputato una buona stagione da 14 punti e 9 rimbalzi in soli 24 minuti di gioco a partita è diventato tra i più forti della lega negli anni seguenti e Boban Marjanovic che nonostante gli evidenti limiti nel difendere giocatori rapidi ed esplosivi (dettagli che non vengono considerati nell'analisi) in neanche 12 minuti giocati di media a partita è riuscito a segnare 7 punti e a catturare 5 rimbalzi.

Da queste osservazioni si nota quindi che il modello non tende a cogliere i giocatori più influenti in assoluto sul gioco offensivo della loro squadra, ma quei giocatori che pur non giocando l'intera partita, nei parziali di gioco che gli sono concessi riescono a creare più azioni pericolose. Ciò è dovuto anche al fatto che spesso i giocatori più forti di ogni squadra, oltre a segnare molti punti e a creare occasioni anche per i compagni, entrano quasi sempre a far parte dell'azione. In questo modo avranno un numero di giocate molto superiore a giocatori meno di punta, portandoli ovviamente a partecipare anche a molte più azioni che non portano ad un canestro finale.

In Figura 3.2 è rappresentata l'evoluzione delle stime dei parametri associati ai migliori nove giocatori dal punto di vista offensivo. Si può notare come per tutti i giocatori considerati le stime siano variabili all'inizio e tendano a stabilizzarsi con il tempo. Ciò è naturale se si considera che la dimensione campionaria N_b aumenta con l'aumentare

dei *batch* e porta ad avere un campione di dati molto grande una volta arrivati a fine stagione. Per questo motivo, anche aggiornando i dati con ulteriori partite non ci aspettiamo grandi differenze, in quanto le osservazioni presenti in un solo *chunk* D_b sono molto meno di quelle raccolte precedentemente. Solo grandi cambiamenti persistenti nel tempo potrebbero quindi portare a notare delle modifiche alle stime tali da poterle notare osservando l'evoluzione attraverso strumenti grafici.

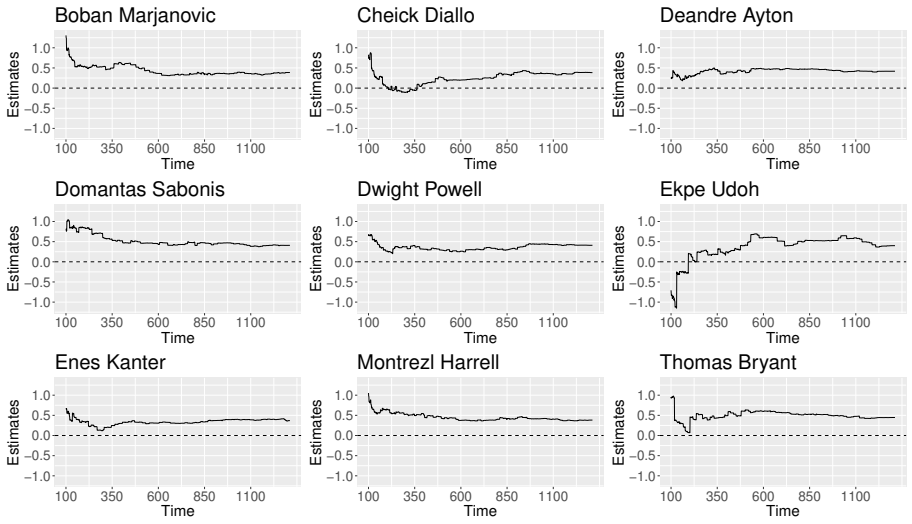

FIGURA 3.2: Evoluzione delle stime del modello logistico nel tempo, relative ai migliori giocatori, ottenute con l'aggiornamento tramite *rho architecture*.

In Tabella 3.7 vengono presentati i coefficienti associati a ciascuna squadra, siano essi significativi o meno, seguendo la notazione introdotta in Tabella 3.1. Osservando i valori delle stime di tali coefficienti si nota come quasi la totalità di essi siano negativi, ma se si osserva anche il valore del *p-value* ad essi associato si evince chiaramente che tutti i coefficienti con valore maggiore risultino non significativi ad un livello di significatività del 5%. Ciò significa che le squadre con rendimento offensivo elevato durante la stagione non si discostano significativamente dagli *Atlanta Hawks*, ovvero dalla squadra di riferimento. Guardando il posizionamento in classifica della squadra di Atlanta, si fa fatica ad interpretare tali risultati, dato che gli *Hawks* si sono classificati dodicesimi su quindici nell'*Eastern Conference*, mancando l'obiettivo minimo della qualificazione ai *playoff* e considerando la percentuale di vittorie di tutte le squadre si classificano ventiseiesimi su trenta. Ciò è però principalmente dovuto alla loro pessima tenuta difensiva, che li porta ad essere la peggior squadra dell'intera lega per canestri subiti. Nonostante questo la loro produzione offensiva risulta essere di tutto rispetto, al pari delle migliori squadre NBA. Oltre ad Atlanta, infatti, solo le squadre con associato un coefficiente non significativo sono state capaci di segnare più di novemila punti durante la stagione

TABELLA 3.7: Coefficienti del modello logistico aggiornato tramite la *rho architecture* per le squadre.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.288	0.052	0.000
BOS	0.051	0.071	0.473
GSW	0.039	0.063	0.539
LAL	0.008	0.062	0.900
MIL	-0.005	0.064	0.933
TOR	-0.056	0.058	0.335
ORL	-0.057	0.054	0.290
HOU	-0.078	0.058	0.180
UTA	-0.101	0.070	0.147
SAS	-0.103	0.064	0.107
DEN	-0.107	0.086	0.213
MEM	-0.112	0.053	0.036
LAC	-0.113	0.056	0.042
PHI	-0.146	0.054	0.007
NYK	-0.149	0.054	0.006
IND	-0.150	0.073	0.039
MIN	-0.152	0.063	0.016
DET	-0.153	0.065	0.018
OKC	-0.156	0.072	0.030
SAC	-0.164	0.059	0.006
MIA	-0.166	0.061	0.007
CHI	-0.178	0.055	0.001
NOP	-0.185	0.066	0.005
WAS	-0.192	0.056	0.001
CHA	-0.205	0.065	0.001
PHX	-0.215	0.059	0.000
DAL	-0.234	0.056	0.000
POR	-0.245	0.068	0.000
CLE	-0.269	0.058	0.000
BKN	-0.278	0.061	0.000

regolare, giustificando l'assenza di differenze rilevata dal modello. Le differenze notate sono infatti esclusivamente con le squadre con prestazioni offensive di minor qualità.

In Figura 3.3 viene raffigurata l'evoluzione delle stime per le migliori nove squadre con coefficienti non significativi al 5%, mentre in Figura 3.4 per le squadre con coefficienti significativi al 5%. Indipendentemente dalla significatività e dal segno del coefficiente si nota un andamento comune che porta le stime ad avere un'impennata iniziale assumendo valori abbastanza elevati per poi subire una decrescita dopo circa 200 partite complessive. Questo andamento comune delle stime non è dovuto a caratteristiche delle

singole squadre, ma al valore di riferimento dell'intercetta. Gli *Atlanta Hawks* hanno infatto attraversato un momento molto difficile proprio ad inizio stagione, quando oltre ai noti problemi difensivi anche la loro media punti è risultata inferiore a quella delle altre squadre, portando ad una striscia di sconfitte consecutive. Questo periodo di difficoltà è ben colto infatti dall'andamento delle stime delle altre squadre, che nel confronto hanno un rapporto di quote tra canestri ed errori nettamente superiore nella prima parte di stagione, calando vistosamente nel continuo grazie al ritrovato sistema offensivo degli *Hawks*.

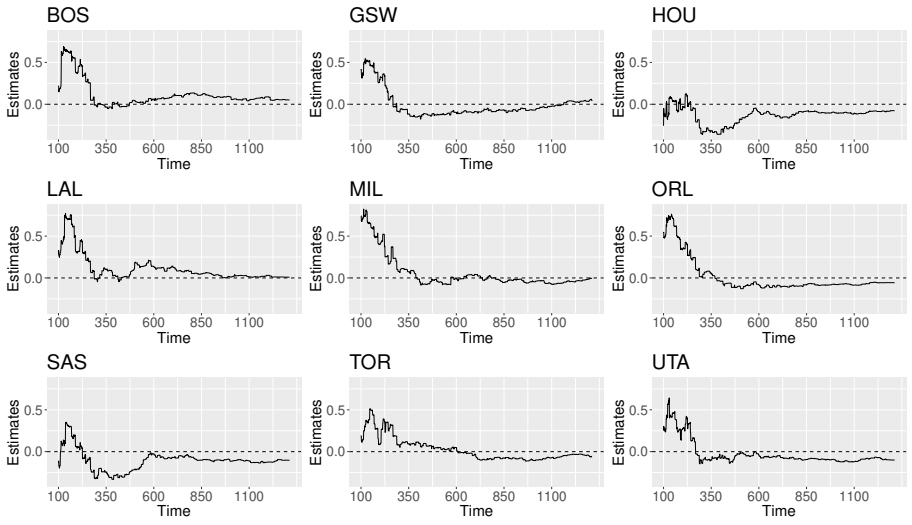

FIGURA 3.3: Evoluzione delle stime del modello logistico nel tempo, relative alle migliori squadre per cui si sono ottenuti coefficienti non significativi, ottenute con l'aggiornamento tramite *rho architecture*.

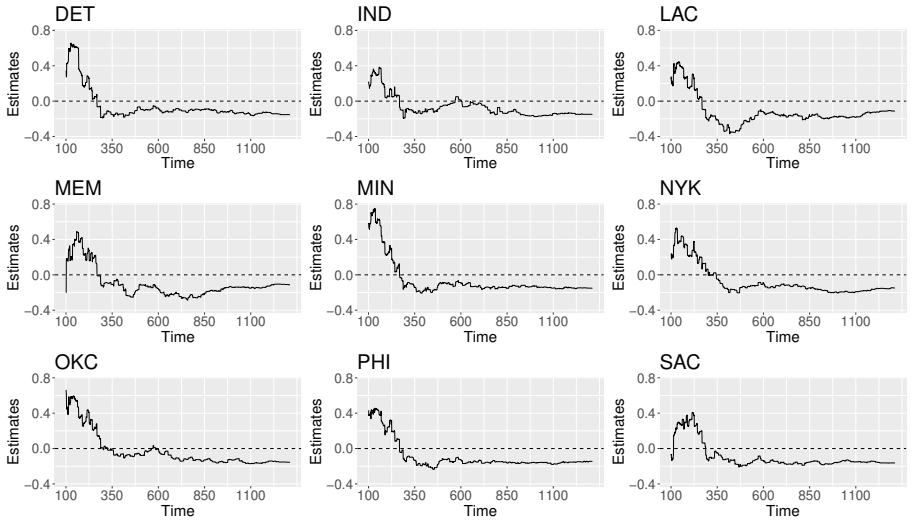

FIGURA 3.4: Evoluzione delle stime del modello logistico nel tempo, relative alle migliori squadre per cui si sono ottenuti coefficienti significativi, ottenute con l'aggiornamento tramite *rho architecture*.

TABELLA 3.8: Coefficienti del modello logistico aggiornato tramite la *rho architecture* per le possibili giocate.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.288	0.052	0.000
free throw	3.122	0.028	0.000
shot	1.775	0.027	0.000
rebound	0.970	0.027	0.000
sub	-0.070	0.029	0.016
jump ball	-0.641	0.086	0.000
violation	-1.768	0.155	0.000
foul	-5.416	0.168	0.000
turnover	-5.889	0.269	0.000

In Tabella 3.8 sono presentati i coefficienti degli eventi possibili al termine della stagione cestistica. I risultati ottenuti non sono affatto sorprendenti e vanno a confermare quanto noto ad ogni tifoso, anche al meno attento. Infatti, si evidenzia come un tiro libero e un tiro dal campo siano le due giocate che risultano maggiormente importanti ai fini di segnare un canestro, mentre una qualsiasi violazione commessa, un fallo commesso e una palla persa portino la probabilità di segnare in meno di 12 secondi a calare drasticamente. I risultati più interessanti riguardano i coefficienti associati ai rimbalzi, che portano ad aumentare la probabilità di segnare nel breve termine, il che può risultare logico considerando esclusivamente i rimbalzi offensivi, ma meno intuitivo per i rimbalzi difensivi. Risulta quindi utile chiarire che negli ultimi anni il gioco del basket nella lega nord americana si è evoluto portando i rimbalzisti difensivi ad aprire direttamente il contropiede in palleggio o a trovare velocemente un compagno smarcato a cui passare la palla per approfittare il più velocemente possibile del mancato posizionamento difensivo degli avversari. Essendo in una partita molto superiore, circa il triplo, il numero di rimbalzi difensivi rispetto a quelli offensivi, possiamo affermare che entrambe le tipologie di rimbalzo catturato risultano importanti al fine di segnare un canestro. Si nota inoltre che una palla contesa diminuisce leggermente la probabilità di fare canestro in poco tempo. Questo può essere spiegato con il fatto che rallenta il gioco, spesso necessitando che passino tre o quattro secondi prima che si possa cominciare propriamente l'azione. Dato ciò, anche se tale azione dovesse concludersi con un canestro, è difficile che la riuscita dell'azione possa essere merito di una palla contesa vinta. Essendo l'interpretazione delle varie possibili giocate così chiara e indipendente dal giocatore o dalla squadra che effettua tale giocata, è sensato aspettarsi che l'evoluzione delle stime dei coefficienti associati non presenti particolare variabilità all'arrivare di dati nuovi e che resti perlopiù costante nel tempo. In Figura 3.5 si può notare come

ciò effettivamente avvenga.

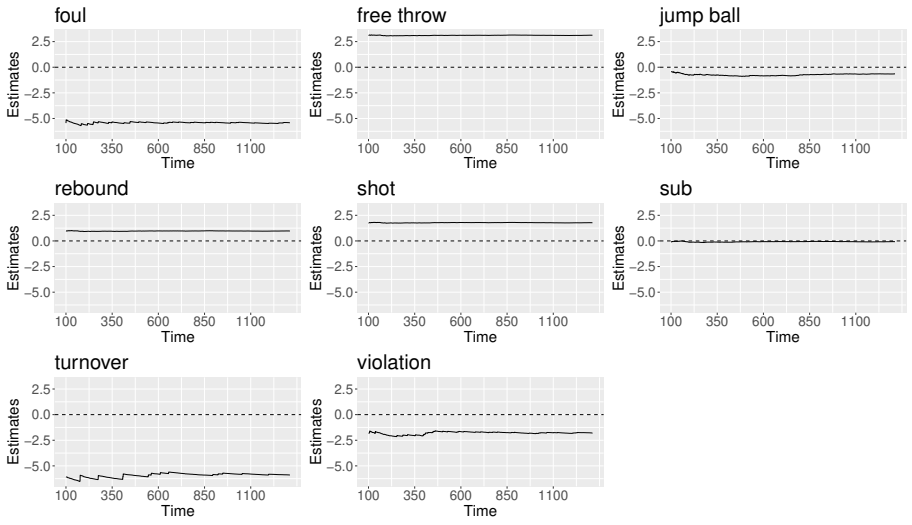

FIGURA 3.5: Evoluzione delle stime del modello logistico nel tempo, relative alle possibili giocate, ottenute con l'aggiornamento tramite *rho architecture*.

Per confermare la bontà di adattamento del modello ai dati si è inoltre deciso di osservare le stime relative ai giocatori che sono stati considerati i migliori della stagione regolare, venendo inseriti nei due migliori quintetti di tutta la stagione, ossia l'*All NBA First Team* e l'*All NBA Second Team*. In Tabella 3.9 sono riportati i coefficienti relativi a tali giocatori, mentre l'andamento delle stime con il passare del tempo è raffigurato in Figura 3.6 per i giocatori appartenenti al primo quintetto NBA e in Figura 3.7 per coloro che fanno parte del secondo miglior quintetto. Si può notare come le stime al termine della stagione risultino tutte positive e significative al 5%, ad eccezione del coefficiente associato a *James Harden*. L'evoluzione delle stime mostra inoltre come la produzione offensiva dei diversi giocatori sia sempre rimasta al di sopra di quella di *Aaron Gordon*, con picchi diversi in diversi momenti della stagione. Si nota infatti che certi giocatori come *Stephen Curry*, *Damian Lillard* e *Kevin Durant* abbiano giocato meglio durante la prima parte della stagione, mentre *Nikola Jokic* e *Kawhy Leonard* siano cresciuti con il passare delle partite, con l'ultimo citato che è stato il fulcro offensivo dei *Toronto Raptors*, vincitori del titolo nella stagione NBA 2018-2019.

3.2.2 Riduzione della distorsione di Firth

Nonostante non si siano riscontrati problemi di convergenza nella stima del modello logistico, si è comunque scelto di adattare un modello di regressione logistica con correzione della distorsione di Firth (1993) ai dati per ridurre la variabilità delle stime

TABELLA 3.9: Coefficienti del modello logistico aggiornato tramite la *rho architecture* per i giocatori inseriti nei due migliori quintetti NBA.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.288	0.052	0.000
James Harden	0.097	0.052	0.062
Stephen Curry	0.149	0.061	0.015
Giannis Antetokounmpo	0.242	0.059	0.000
Paul George	0.184	0.070	0.008
Nikola Jokic	0.305	0.084	0.000
Joel Embiid	0.184	0.050	0.000
Kevin Durant	0.233	0.061	0.000
Damian Lillard	0.311	0.065	0.000
Kawhi Leonard	0.289	0.057	0.000
Kyrie Irving	0.162	0.073	0.026

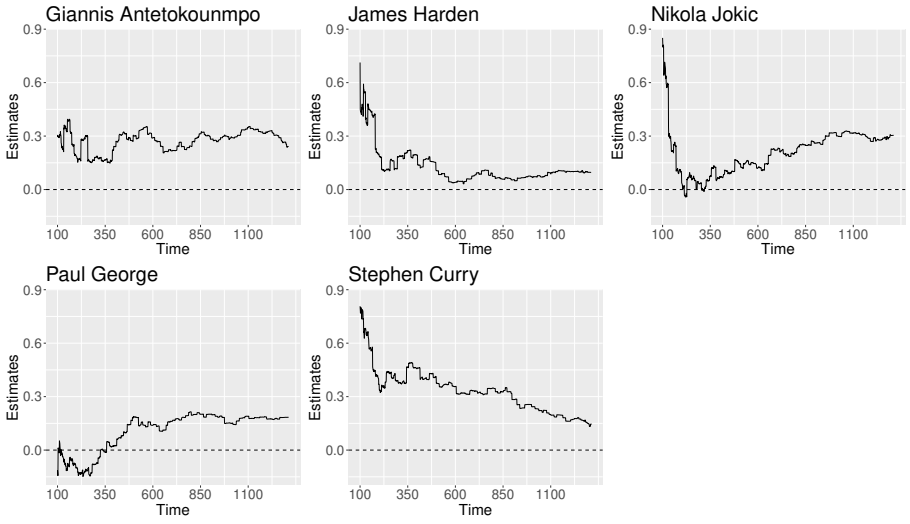

FIGURA 3.6: Evoluzione delle stime del modello logistico nel tempo, relative ai giocatori inseriti nell'*All NBA First Team*, ottenute con l'aggiornamento tramite *rho architecture*.

dei coefficienti inerenti ai giocatori con poche osservazioni all'interno del *dataset*. Tale modello è stato adattato tramite l'utilizzo della libreria R `brglm2` (Kosmidis, 2023).

Per il confronto tra il modello logistico con stime di massima verosimiglianza e il modello proposto da Firth (1993) si presentano per entrambi i modelli esclusivamente i coefficienti dei giocatori che sono risultati migliori nel modello con stime di massima verosimiglianza. Per evitare che nel confronto influisca anche il processo di stima tramite la *rho architecture* i coefficienti presentati saranno quelli stimati esclusivamente sui primi $B_1 = 100$ *batch* di dati, su cui i modelli sono stati applicati utilizzando le funzioni di libreria e le stime non hanno subito un processo di aggiornamento. I risultati del modello logistico con stime di massima verosimiglianza sono visibili in Tabella 3.10, mentre le

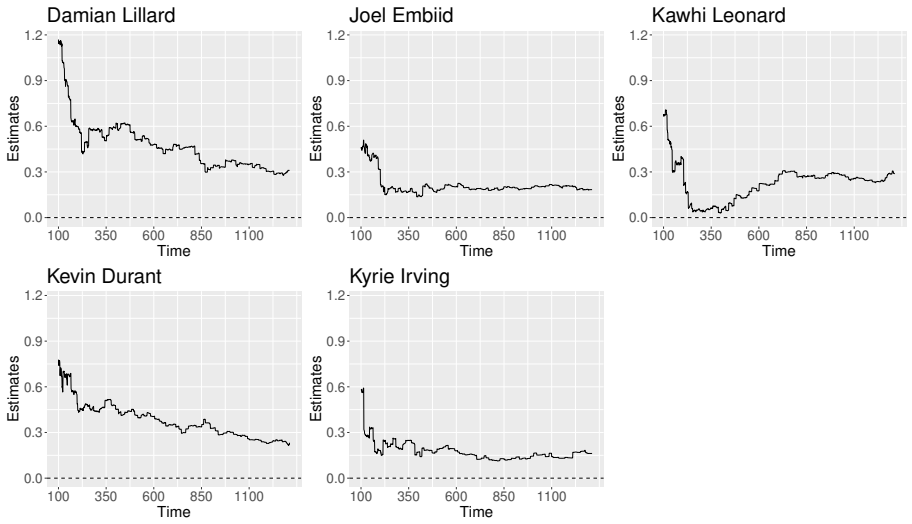

FIGURA 3.7: Evoluzione delle stime del modello logistico nel tempo, relative ai giocatori inseriti nell'*All NBA Second Team*, ottenute con l'aggiornamento tramite *rho architecture*.

TABELLA 3.10: Coefficienti del modello logistico stimato sulle prime cento partite per i giocatore con i valori più alti delle stime dei parametri a fine stagione.

	Stima	*Standard Error*	*p-value*
Intercetta	-3.246	0.388	0.000
Thomas Bryant	0.955	0.531	0.072
Deandre Ayton	0.262	0.296	0.376
Dwight Powell	0.687	0.384	0.074
Domantas Sabonis	0.819	0.354	0.021
Ekpe Udoh	-0.707	0.863	0.413
Boban Marjanovic	1.308	0.379	0.001
Cheick Diallo	0.828	0.512	0.106
Montrezl Harrell	1.054	0.329	0.001
Enes Kanter	0.676	0.361	0.062
Jarrett Allen	0.543	0.329	0.099
T.J. McConnell	0.897	0.401	0.025
Derrick Favors	0.299	0.477	0.531
Steven Adams	-0.008	0.357	0.983
Jakob Poeltl	1.244	0.541	0.021
Robin Lopez	0.197	0.571	0.730
Jonas Valanciunas	0.957	0.298	0.001
Mitchell Robinson	0.619	0.466	0.184
Nerlens Noel	0.525	0.373	0.160

stime ottenute con il metodo di Firth vengono presentate in Tabella 3.11. Dal confronto si evince che i risultati ottenuti massimizzando la verosimiglianza sono attendibili e che i due modelli stimati possono considerarsi essenzialmente equivalenti per i dati in questione. Infatti, si può vedere che lo *shrinkage* verso zero dei coefficienti è molto

TABELLA 3.11: Coefficienti del modello logistico stimato con il metodo di Firth sulle prime cento partite per i giocatore con i valori più alti delle stime dei parametri a fine stagione.

	Stima	*Standard Error*	*p-value*
Intercetta	-3.144	0.381	0.000
Thomas Bryant	0.936	0.528	0.076
Deandre Ayton	0.260	0.294	0.377
Dwight Powell	0.669	0.380	0.078
Domantas Sabonis	0.802	0.350	0.022
Ekpe Udoh	-0.629	0.845	0.457
Boban Marjanovic	1.285	0.376	0.001
Cheick Diallo	0.809	0.506	0.110
Montrezl Harrell	1.036	0.326	0.001
Enes Kanter	0.662	0.357	0.064
Jarrett Allen	0.525	0.325	0.106
T.J. McConnell	0.883	0.397	0.026
Derrick Favors	0.288	0.468	0.538
Steven Adams	-0.007	0.354	0.983
Jakob Poeltl	1.230	0.537	0.022
Robin Lopez	0.211	0.564	0.708
Jonas Valanciunas	0.938	0.295	0.001
Mitchell Robinson	0.612	0.461	0.184
Nerlens Noel	0.519	0.371	0.161

limitato, come anche la diminuzione della varianza ad essi associata. Viene comunque scelto di utilizzare le stime ottenute dal modello con verosimiglianza penalizzata come punto di partenza per l'aggiornamento delle stime con le metodologie proposte in seguito.

Per completezza vengono presentati anche i risultati ottenuti aggiornando tramite la *rho architecture* le stime del modello di Firth (1993) con tutti i *batch* di dati disponibili, per valutare se anche piccoli cambiamenti nella stima dei coefficienti e della matrice di varianza/covarianza possano inficiare sull'aggiornamento delle stime. Visto quanto emerge dai coefficienti dei migliori giocatori presentati in Tabella 3.12, delle squadre presentati in Tabella 3.13 e dei possibili eventi presentati in Tabella 3.14 e dal loro processo di aggiornamento, visibile rispettivamente nelle Figure 3.8, 3.9 e 3.10, e dal loro confronto con quanto ottenuto con le stime di massima verosimiglianza, si può affermare che le stime sono risultate stabili e che le stime iniziali nelle prime 100 partite non hanno influito su esse. Valgono quindi le stesse considerazioni fatte per le stime ottenute partendo dal modello logistico con stime di massima verosimiglianza.

Tabella 3.12: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.282	0.052	0.000
Thomas Bryant	0.447	0.070	0.000
Deandre Ayton	0.418	0.061	0.000
Dwight Powell	0.409	0.068	0.000
Domantas Sabonis	0.408	0.077	0.000
Ekpe Udoh	0.400	0.150	0.008
Boban Marjanovic	0.387	0.078	0.000
Cheick Diallo	0.384	0.092	0.000
Montrezl Harrell	0.383	0.058	0.000
Enes Kanter	0.370	0.057	0.000
Jarrett Allen	0.363	0.066	0.000
T.J. McConnell	0.356	0.081	0.000
Jakob Poeltl	0.347	0.079	0.000
Derrick Favors	0.346	0.074	0.000
Steven Adams	0.346	0.073	0.000
Robin Lopez	0.343	0.072	0.000
Jonas Valanciunas	0.342	0.066	0.000
Mitchell Robinson	0.339	0.074	0.000
Meyers Leonard	0.336	0.096	0.000

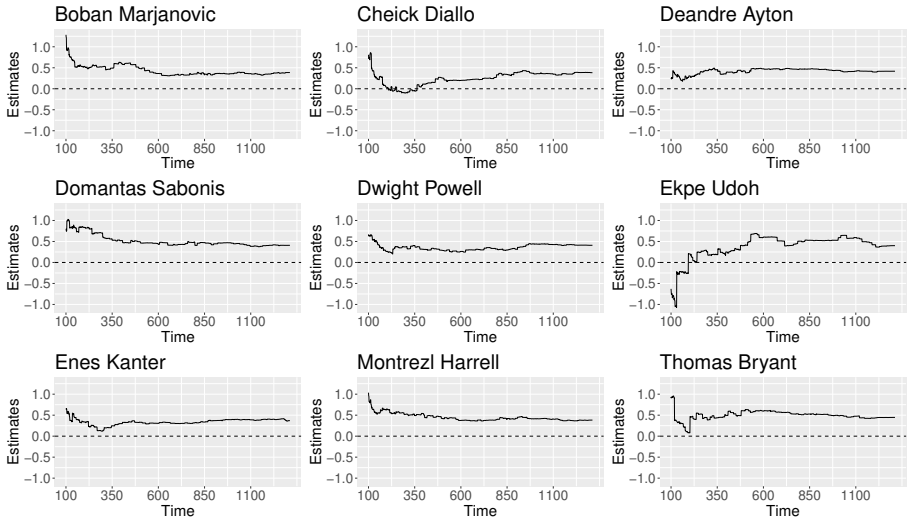

Figura 3.8: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative ai migliori giocatori, ottenute con l'aggiornamento tramite *rho architecture*.

Tabella 3.13: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* per le squadre.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.282	0.052	0.000
BOS	0.052	0.071	0.468
GSW	0.039	0.063	0.538
LAL	0.008	0.061	0.896
MIL	-0.005	0.064	0.941
TOR	-0.055	0.058	0.337
ORL	-0.056	0.054	0.297
HOU	-0.078	0.058	0.181
UTA	-0.100	0.070	0.151
SAS	-0.103	0.064	0.108
DEN	-0.106	0.086	0.216
MEM	-0.112	0.053	0.036
LAC	-0.113	0.056	0.043
PHI	-0.145	0.054	0.008
NYK	-0.148	0.054	0.007
IND	-0.149	0.073	0.040
MIN	-0.151	0.063	0.017
DET	-0.152	0.065	0.019
OKC	-0.155	0.072	0.032
SAC	-0.163	0.059	0.006
MIA	-0.164	0.061	0.007
CHI	-0.177	0.055	0.001
NOP	-0.183	0.066	0.005
WAS	-0.192	0.056	0.001
CHA	-0.204	0.064	0.002
PHX	-0.214	0.059	0.000
DAL	-0.233	0.055	0.000
POR	-0.245	0.067	0.000
CLE	-0.267	0.058	0.000
BKN	-0.277	0.061	0.000

Tabella 3.14: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* per le possibili giocate.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.282	0.052	0.000
free throw	3.115	0.028	0.000
shot	1.769	0.026	0.000
rebound	0.965	0.027	0.000
sub	-0.074	0.029	0.010
jump ball	-0.640	0.085	0.000
violation	-1.761	0.152	0.000
foul	-5.403	0.163	0.000
turnover	-5.862	0.253	0.000

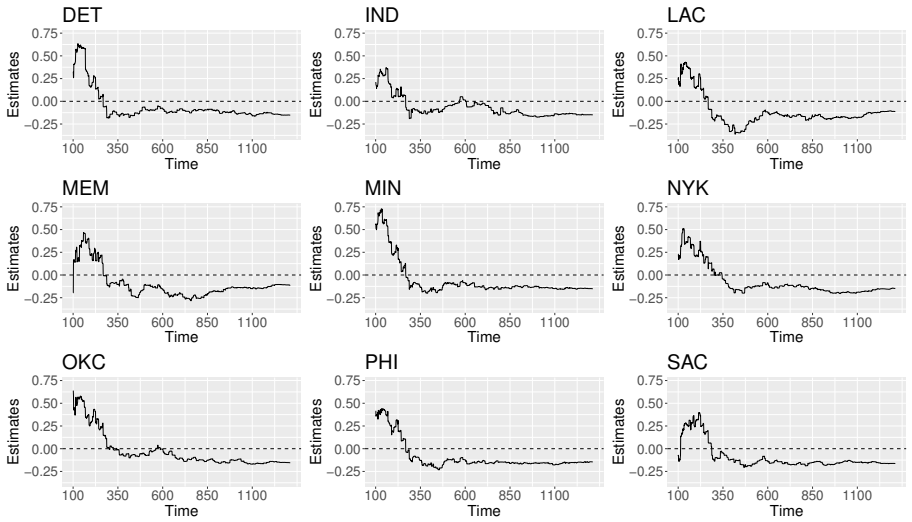

FIGURA 3.9: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative alle migliori squadre che si discostano significativamente dagli *Atlanta Hawks*, ottenute con l'aggiornamento tramite *rho architecture*.

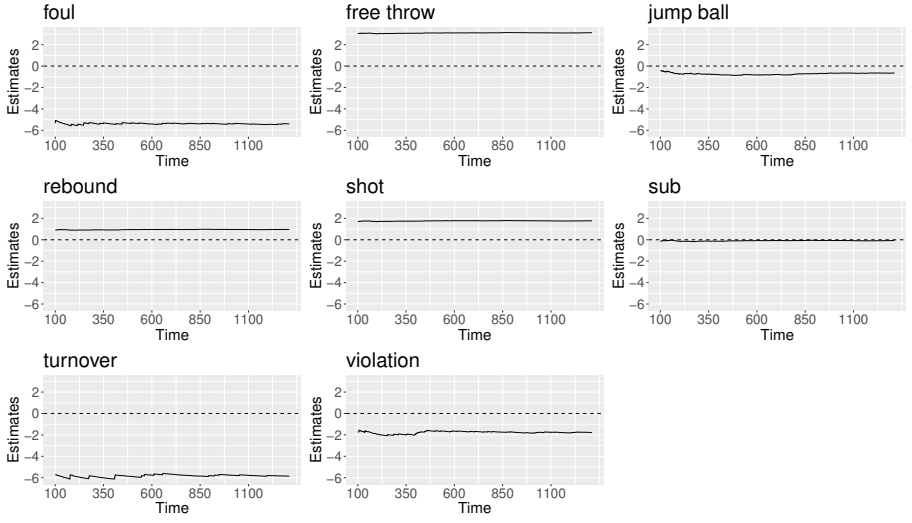

FIGURA 3.10: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative alle possibili giocate, ottenute con l'aggiornamento tramite *rho architecture*.

3.3 Effetto dinamico tramite utilizzo di pesi

Dopo aver visto l'applicazione della *rho architecture* nella versione proposta da Luo & Song (2020) in cui tutti i dati sono considerati indipendenti e vista la natura dei dati a disposizione, che non solo non sono indipendenti ma presentano anche una sequenzialità temporale, risulta di interesse modellare almeno la distanza temporale tra i diversi *batch*. Per fare ciò si può introdurre una funzione peso, come mostrato nel Paragrafo 2.2. In questo modo si riescono ad ottenere stime che pesino maggiormente gli ultimi dati disponibili, portando i *batch* temporalmente più vicini al momento di stima ad avere una maggiore influenza. Modellando la distanza temporale si è quindi in grado di analizzare l'andamento delle prestazioni offensive di giocatori e squadre nel breve periodo, permettendo di valutare lo stato di forma in un qualsiasi momento della stagione. Ciò può essere utile anche ai giocatori stessi, alle squadre e ai loro allenatori per valutare se un determinato lavoro svolto in allenamento ha portato a miglioramenti visibili anche in partita, dato che pesando maggiormente gli ultimi dati si permette alle stime dei vari coefficienti di variare più velocemente. Si può inoltre valutare che carichi di lavoro affrontare in un determinato periodo della stagione per riuscire ad esprimere al meglio il proprio valore nei momenti più importanti per il raggiungimento degli obiettivi preposti, il che spesso avviene verso la fine della stagione regolare in cui si punta ad un determinato piazzamento in classifica e lungo tutto il periodo dei *playoff*.

3.3.1 Pesi esponenzialmente crescenti

Come prima funzione di peso viene considerata la funzione definita nell'equazione (2.21), che prevede di assegnare un peso con valore unitario al primo *batch* di dati e un valore crescente in modo esponenziale ad ogni *batch* successivo. Per la distanza temporale tra le varie partite si è deciso di non considerare il giorno e l'ora in cui sono state giocate, ma solo il loro ordinamento, definendo $t_j = j$. Dato che il modello iniziale è stato stimato sui primi 100 *batch* di dati aggregati, si è deciso di assegnare a tutti i dati presenti in $\widetilde{D}_{100}$ un peso pari a 1 e agli altri dati un peso maggiore, definito modificando l'equazione (2.21) come

$$w_j = \exp\left\{\xi\left(t_j - t_{100}\right)\right\}, \qquad j = 1, \ldots, b, \tag{3.1}$$

con $\xi = 0.01$. Ciò equivale a considerare $\widetilde{D}_{100}$ come primo *batch* di dati e utilizzarlo come punto di riferimento per il calcolo dei pesi. La scelta del valore di ξ è stata fatta in modo tale che il peso risultasse circa doppio dopo circa 100 partite, indicativamente equivalenti a due settimane. Valori più piccoli di ξ avrebbero limitato la flessibilità del

modello, pesando similmente un numero maggiore di partite, mentre valori più grandi avrebbero pesato eccessivamente troppe poche partite, riducendo la stabilità delle stime.

Per i modelli stimati con i dati pesati, l'interpretazione dei risultati ottenuti dopo la stima effettuata con tutti i *batch* a disposizione non è significativa di tutta la stagione,

quindi, non potendo presentare i risultati riguardanti tutti i giocatori per questioni di spazio, si è deciso di presentare esclusivamente le stime dei coefficienti inerenti ai giocatori che si sono dimostrati migliori durante tutta la stagione, ovvero i giocatori che hanno ottenuto i risultati migliori nel modello logistico stimato con il metodo di Firth e aggiornato con la *rho architecture* sui dati non pesati. In tal modo si può vedere l'evoluzione delle stime e dello stato di forma nel tempo per i giocatori che sono risultati più costantemente ad alto livello per l'intera durata della stagione. Vengono quindi presentati i grafici inerenti all'aggiornamento delle stime con pesi crescenti per i migliori giocatori offensivi in Figura 3.11, per le migliori nove squadre che si sono distinte da Atlanta lungo tutta la stagione in Figura 3.12 e per le possibili giocate in Figura 3.13.

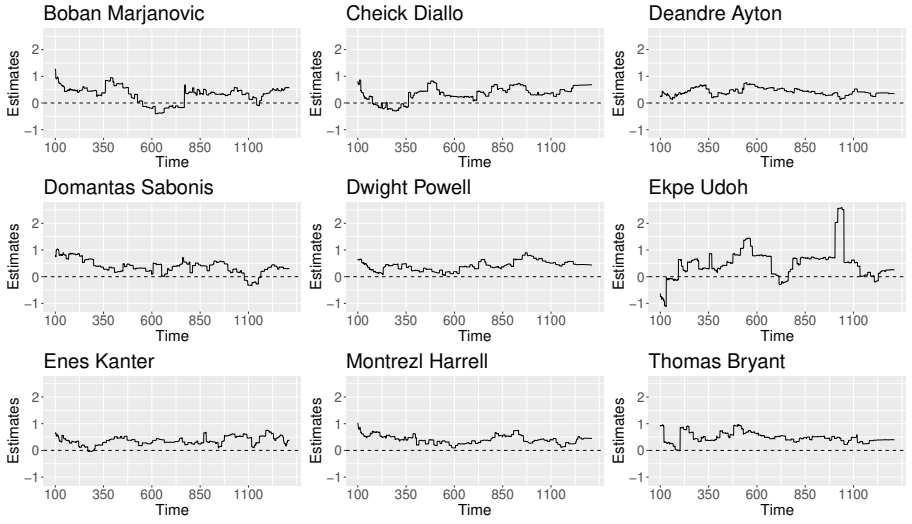

FIGURA 3.11: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative ai migliori giocatori, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi esponenzialmente crescenti.

Dall'evoluzione delle stime sembra si sia riusciti a raggiungere lo scopo di aggiungere una dipendenza temporale e quindi permettere alle stime di variare maggiormente nel tempo, tenendo maggiormente conto di eventi temporalmente più vicini. Se però si vanno a considerare gli *output* del modello stimato inerenti ai migliori giocatori al tempo t_{100} in Tabella 3.11, al tempo t_{700} in Tabella 3.15 e all'ultimo tempo disponibile, ovvero t_{1312} in Tabella 3.16 si nota come, con il passare del tempo, e quindi anche con l'aumentare dei pesi, lo *standard error* associato alle stime del modello tenda a diventare sempre più grande, portando tutte le stime ad essere non significative da un certo istante temporale

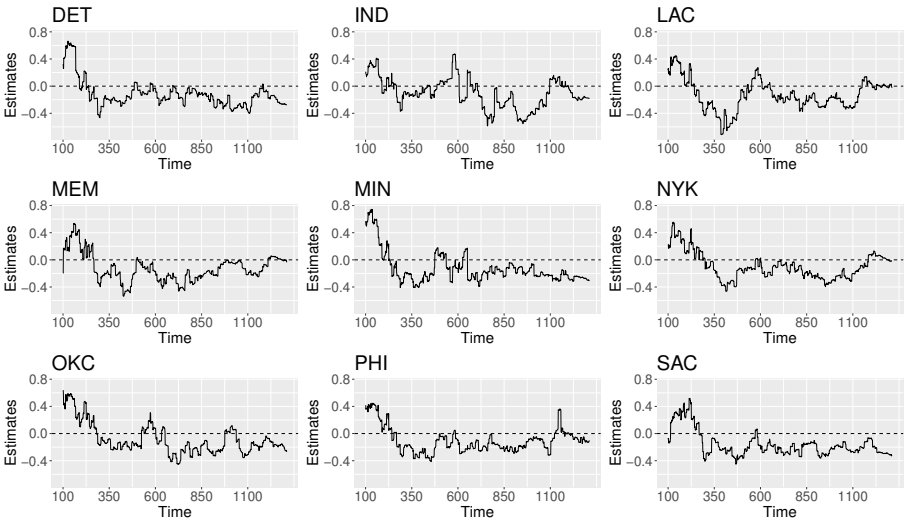

FIGURA 3.12: Evoluzione delle stime del modello logistico con correzione di Firth nel tempo, relative alle migliori squadre che si discostano significativamente dagli *Atlanta Hawks*, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi esponenzialmente crescenti.

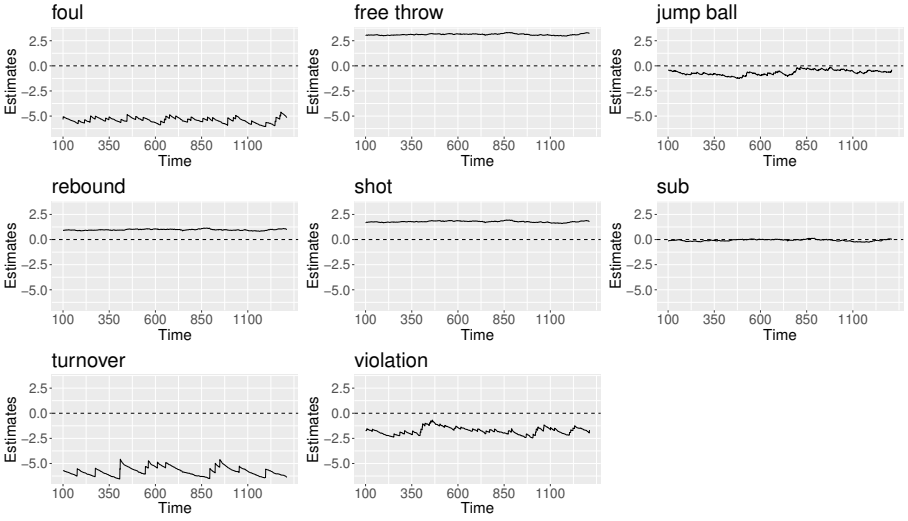

FIGURA 3.13: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative alle possibili giocate, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi esponenzialmente crescenti.

in avanti. Questo ci impedisce di considerare attendibili le stime del modello e quindi va ad inficiare i vantaggi di tale procedura. Il fatto che gli *standard error* tendano ad infinito al divergere del tempo è verosimilmente dovuto ad un problema computazionale imputabile a valori dei pesi eccessivamente grandi. La matrice di varianza/covarianza è infatti ottenuta tramite la forma *sandwich*

$$\widetilde{V}\left(\widetilde{\beta}_b\right) = \widetilde{J}_b^*\left(\widetilde{\beta}_b\right)^{-1} H\left(\widetilde{\beta}_b\right) \widetilde{J}_b^*\left(\widetilde{\beta}_b\right)^{-1}, \tag{3.2}$$

dove al divergere del tempo le matrici $\widetilde{J}_b^*\left(\widetilde{\beta}_b\right)$ e $H\left(\widetilde{\beta}_b\right)$ assumono valori molto grandi che possono portare a problemi nell'inversione e nel prodotto matriciale.

TABELLA 3.15: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* sui primi 700 *batch* di dati pesati con pesi esponenzialmente crescenti per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.250	3.619	0.534
Thomas Bryant	0.503	5.571	0.928
Deandre Ayton	0.466	24.991	0.985
Dwight Powell	0.374	13.777	0.978
Domantas Sabonis	0.234	2.211	0.916
Ekpe Udoh	0.037	202.154	1.000
Boban Marjanovic	-0.129	2.528	0.959
Cheick Diallo	0.086	81.119	0.999
Montrezl Harrell	0.358	25.750	0.989
Enes Kanter	0.306	9.510	0.974
Jarrett Allen	0.266	58.445	0.996
T.J. McConnell	0.483	8.533	0.955
Jakob Poeltl	0.064	30.591	0.998
Derrick Favors	-0.091	56.809	0.999
Steven Adams	0.513	55.992	0.993
Robin Lopez	0.045	19.799	0.998
Jonas Valanciunas	0.567	34.239	0.987
Mitchell Robinson	0.327	51.027	0.995
Meyers Leonard	0.518	84.812	0.995

3.3.2 Pesi riscalati

Una soluzione per evitare di ottenere valori dei pesi eccessivamente grandi è quella di riscalare i pesi per il loro valore massimo all'arrivo di ogni nuovo *batch* di dati, come mostrato in equazione (2.22). In questo modo il valore massimo del peso, associato ai dati contenuti nell'ultimo *chunk*, è pari a uno, mentre i dati contenuti in tutti i *chunk* precedenti assumono valori che decrescono esponenzialmente all'aumentare della distanza temporale. Essendo questa funzione di peso basata sulla distanza temporale dall'ultimo *chunk* disponibile, non è necessario ridefinirla per variare il punto di riferimento come fatto per i pesi crescenti. Per analogia con la funzione peso definita in equazione (3.1) anche per i pesi riscalati è stato fissato il valore del parametri di decadimento esponenziale $\xi = 0.01$.

Per controllare l'andamento della varianza, come fatto in precedenza si confrontano le statistiche riassuntive delle stime del modello in tre istanti temporali diversi, prendendo

TABELLA 3.16: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* su tutti i dati pesati con pesi esponenzialmente crescenti per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.223	18549.543	1.000
Thomas Bryant	0.395	3388.882	1.000
Deandre Ayton	0.353	1046.731	1.000
Dwight Powell	0.434	6184.206	1.000
Domantas Sabonis	0.304	5036.681	1.000
Ekpe Udoh	0.254	13104.641	1.000
Boban Marjanovic	0.566	12226.831	1.000
Cheick Diallo	0.686	15015.194	1.000
Montrezl Harrell	0.458	7562.732	1.000
Enes Kanter	0.383	8073.632	1.000
Jarrett Allen	0.530	18951.760	1.000
T.J. McConnell	0.149	817.621	1.000
Jakob Poeltl	0.351	7080.946	1.000
Derrick Favors	0.549	12217.535	1.000
Steven Adams	0.340	4552.634	1.000
Robin Lopez	0.549	3827.385	1.000
Jonas Valanciunas	0.119	4577.874	1.000
Mitchell Robinson	0.356	4395.811	1.000
Meyers Leonard	0.533	53420.681	1.000

in considerazione l'inizio e la fine della stagione, oltre ad un istante temporale intermedio. I risultati ottenuti per t_{100} sono quelli presentati in Tabella 3.11, per t_{700} in Tabella 3.17 e per t_{1312} in Tabella 3.18. Confrontando i valori delle stime e degli *standard error* si può notare come riscalando i pesi sia stata possibile una stima sensata della matrice di varianza/covarianza, e quindi degli *standard error*, senza incorrere in problemi di natura numerica.

In Figura 3.14 è raffigurata l'evoluzione delle stime per i migliori giocatori; in Figura 3.15 è raffigurato l'andamento delle migliori squadre che si sono discostate significativamente dagli *Hawks* durante l'intera durata della stagione, mentre in Figura 3.16 è raffigurato l'andamento delle stime per le possibili giocate. In tutti e tre i grafici si nota come l'aggiunta dei pesi abbia reso possibile una maggior variabilità dell'andamento delle stime nel tempo. Per i giocatori si nota un calo nelle prestazioni offensive di Boban Marjanovic circa a metà stagione, dove per circa duecento partite totali, quindi circa 15 partite della sua squadra, è risultato offensivamente peggiore anche di Aaron Gordon, ovvero del riferimento del modello. Molto evidente è anche il picco di prestazione raggiunto per circa cinquanta partite da Ekpe Udoh, che porta il rapporto di quote a lui associato della probabilità di partecipare ad un'azione pericolosa rispetto a non

TABELLA 3.17: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* sui primi 700 *batch* di dati pesati con pesi riscalati per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.250	0.021	0.000
Thomas Bryant	0.503	0.080	0.000
Deandre Ayton	0.466	0.053	0.000
Dwight Powell	0.374	0.077	0.000
Domantas Sabonis	0.234	0.031	0.000
Ekpe Udoh	0.037	0.670	0.956
Boban Marjanovic	-0.129	0.256	0.615
Cheick Diallo	0.086	0.285	0.764
Montrezl Harrell	0.358	0.060	0.000
Enes Kanter	0.306	0.019	0.000
Jarrett Allen	0.266	0.127	0.036
T.J. McConnell	0.483	0.037	0.000
Jakob Poeltl	0.064	0.037	0.086
Derrick Favors	-0.091	0.202	0.651
Steven Adams	0.513	0.177	0.004
Robin Lopez	0.045	0.053	0.392
Jonas Valanciunas	0.567	0.122	0.000
Mitchell Robinson	0.327	0.134	0.014
Meyers Leonard	0.518	0.197	0.009

parteciparci ad essere ben dodici volte superiore a quello di *Aaron Gordon*.

Ancora più interessante è l'analisi dell'andamento delle squadre, che evidenzia un calo di forma dei *Los Angeles Clippers* a partire circa dalla duecentesima partita fino alla quattrocentesima. È comunque necessario ricordare come per tutte le squadre ci sia stato un leggero calo fisiologico rispetto agli *Atlanta Hawks* in questa finestra temporale, in quanto tale squadra è cresciuta molto a livello realizzativo. Si nota inoltre come diverse squadre abbiano avuto un rendimento offensivo quantomeno pari se non superiore ad Atlanta nelle ultime partite disponibili.

L'introduzione dei pesi non ha avuto invece particolare effetto sulla stima dei coefficienti relativi alle possibili giocate, in quanto per caratteristiche intrinseche alle stesse esse tendono ad assestarsi su un certo valore e a non presentare particolari variazioni, in quanto restano più o meno importanti ai fini di essere influenti sulla finalizzazione di un'azione indipendentemente dal momento della stagione.

È inoltre interessante notare come confrontando sia i grafici che le tabelle relativi al modello stimato su dati pesati con pesi riscalati con quelli relativi al modello stimato sui dati pesati con pesi esponenziali crescenti si può notare come le stime dei coefficienti siano essenzialmente le stesse, mentre variano di molto gli *standard error* stimati. Ciò

TABELLA 3.18: Coefficienti del modello logistico stimato con il metodo di Firth aggiornato tramite la *rho architecture* su tutti i dati pesati con pesi riscalati per i giocatore con i valori più alti delle stime dei parametri.

	Stima	*Standard Error*	*p-value*
Intercetta	-2.223	0.062	0.000
Thomas Bryant	0.395	0.039	0.000
Deandre Ayton	0.353	0.009	0.000
Dwight Powell	0.434	0.068	0.000
Domantas Sabonis	0.304	0.128	0.018
Ekpe Udoh	0.254	0.071	0.000
Boban Marjanovic	0.566	0.116	0.000
Cheick Diallo	0.686	0.217	0.002
Montrezl Harrell	0.458	0.070	0.000
Enes Kanter	0.383	0.058	0.000
Jarrett Allen	0.530	0.165	0.001
T.J. McConnell	0.149	0.022	0.000
Jakob Poeltl	0.351	0.079	0.000
Derrick Favors	0.549	0.141	0.000
Steven Adams	0.340	0.045	0.000
Robin Lopez	0.549	0.099	0.000
Jonas Valanciunas	0.119	0.126	0.342
Mitchell Robinson	0.356	0.047	0.000
Meyers Leonard	0.533	0.337	0.113

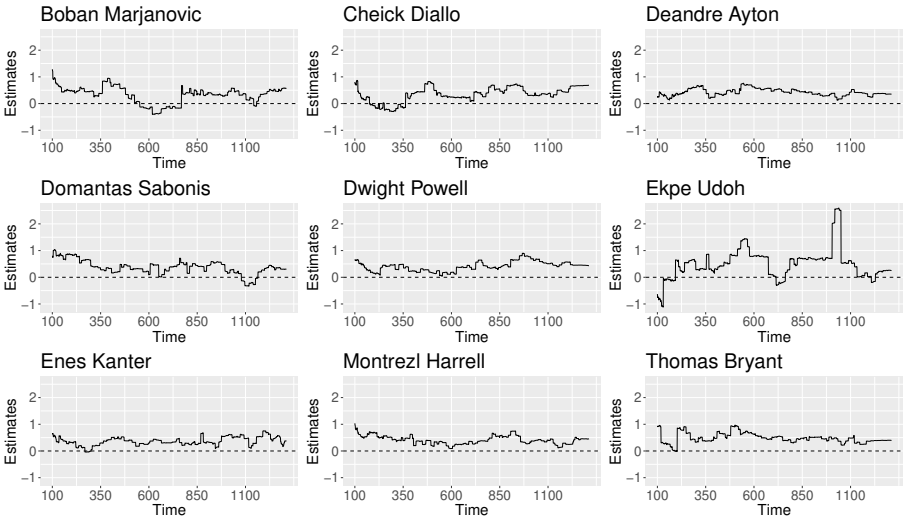

FIGURA 3.14: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative ai migliori giocatori, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi riscalati.

è ragionevole in quanto l'andamento della funzione di peso è esattamente lo stesso e i valori eccessivamente grandi degli standard error presenti in Tabella 3.17 e in Tabella 3.18 sono dovuti esclusivamente a difficoltà computazionali.

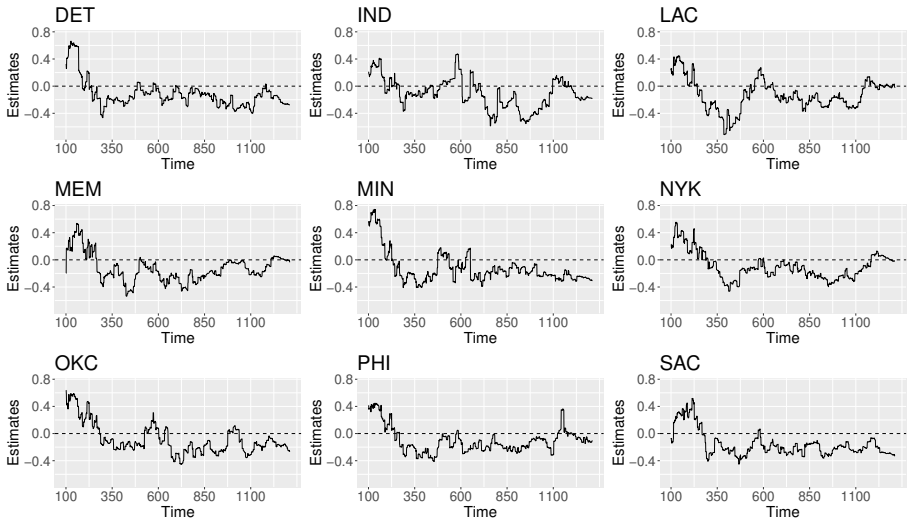

FIGURA 3.15: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative alle migliori squadre che si discostano significativamente dagli *Atlanta Hawks*, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi riscalati.

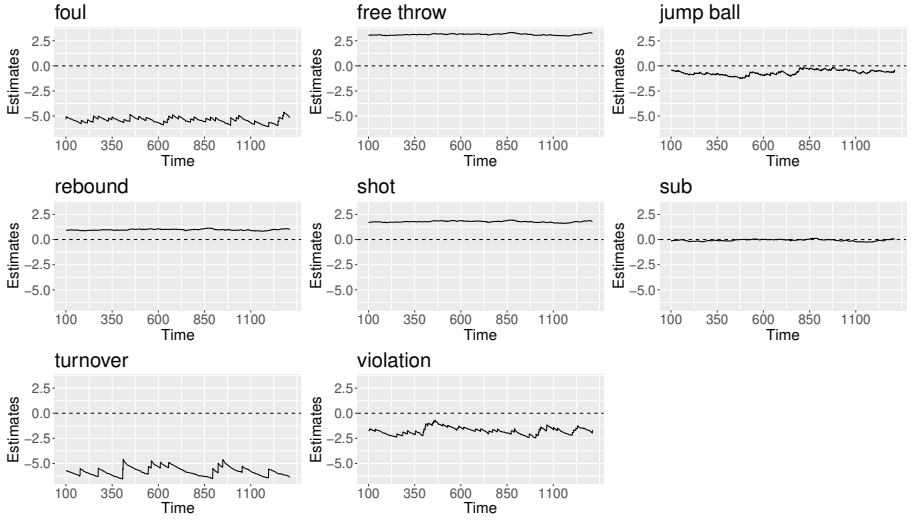

FIGURA 3.16: Evoluzione delle stime del modello logistico stimato con il metodo di Firth nel tempo, relative alle possibili giocate, ottenute con l'aggiornamento tramite *rho architecture* su dati pesati con pesi riscalati.

Riscalando i pesi si è quindi riusciti a mantenere la flessibilità del modello stimato su dati pesati migliorando la stima della matrice di varianza covarianza, a discapito di un maggior costo computazionale e di un maggior spazio in memoria necessario per adattare tale modello. Infatti è necessario salvare in memoria la matrice $J_j\left(\widetilde{\beta}_j; D_j\right)$, di dimensione $p \times p$, e il valore della funzione punteggio $U_j\left(\widetilde{\beta}_j; D_j\right)$ per $j = 1, \ldots, b$ e ricalcolare con il nuovo valore dei pesi la matrici $\widetilde{J}^*_{b-1}$ e il vettore $\widetilde{U}^*_{b-1}$, come mostrato al passo 7 dell'Algoritmo 4 per ogni nuovo *batch* di dati con cui si vuole aggiornare le

stime del modello.

3.3.3 Pesi riscalati e troncati

Un ulteriore possibilità per la funzione di peso è stata presentata nell'equazione (2.23) e in Figura 2.7. Utilizzando tale funzione peso si andrebbero a risolvere parzialmente i problemi computazionali e la necessità di memoria per poter aggiornare le stime. In particolare la memoria richiesta non aumenterebbe con il tempo fino a divergere, ma raggiungerebbe un *plateau* dopo k *batch*, in quanto all'arrivo di un ulteriore *chunk* si andrebbero a salvare i valori della matrice di informazione osservata e della funzione punteggio relativi a tale porzione di dati, cancellando però dalla memoria le stesse informazioni relative al *chunk* più vecchio a disposizione.

Purtroppo con i dati a disposizione non è stato possibile adattare un modello logistico basato su dati pesati con questa funzione peso in quanto la matrice del modello di ogni *chunk* risulta essere molto sparsa, dato che quasi la totalità delle 442 colonne presenti sono variabili *dummy* che codificano i fattori di solo quattro variabili originali. Per questo motivo la matrice di informazione osservata relativa ad ogni *batch* di dati presenta molti valori *off-diagonal* identicamente nulli. Nel momento in cui si perde l'informazione del modello stimato sulle prime cento partite infatti capita che anche la somma pesata delle varie matrici di informazione osservata presenti diversi elementi nulli, impedendo la sua inversione e quindi la stima della matrice di varianza covarianza attraverso l'utilizzo dell'equazione (3.2).

Conclusioni

Nella presente tesi sono state proposte delle metodologie per l'aggiornamento ricorsivo delle stime di un modello logistico che poi sono state applicate nell'analisi di dati cestistici per la valutazione della pericolosità offensiva di ogni singola giocata compiuta dai giocatori. Dopo aver stimato un modello logistico su un insieme di dati inerenti alle prime 100 partite, si è proceduto aggiornando le stime per ogni partita disponibile. In questo modo è stato possibile valutare i giocatori non solo al termine della stagione, ma anche nei vari periodi intermedi, riuscendo a valutare lo stato di forma e il confronto tra essi al termine di ogni partita.

Dopo aver presentato la *rho architecture* proposta da Luo & Song (2020) si è cercato di correggere un limite di tale metodologia, che può essere applicata solamente ad osservazioni indipendenti. Presentando i dati NBA una dipendenza temporale, si è deciso di modellare tale dipendenza facendo in modo che il modello considerasse come più importanti i dati temporalmente più vicini al momento di stima.

Si sono quindi considerati come validi i risultati ottenuti su tutta la stagione aggiornando il modello logistico e considerando le osservazioni indipendenti, per valutare le *performance* sull'intera durata della stagione. Si è inoltre cercato di valutare il momento di forma delle squadre aggiungendo a tale metodologia tre diverse funzioni peso per i dati, valutandone l'impatto sul costo computazionale e sulle stime dei coefficienti, oltre che sulla loro variabilità.

Utilizzando pesi esponenzialmente crescenti si è notata una velocità nell'aggiornamento del modello pari a quella della versione non pesata. Le stime del modello risultano inoltre stabili, potendo però presentare problemi computazionali nel calcolo della matrice di varianza/covarianza dei coefficienti, dovuta a valori eccessivamente grandi dei pesi al divergere del tempo.

Per risolvere tale problema si è deciso di riscalare i pesi, assegnando peso unitario ai dati presenti nell'ultimo *batch* disponibile e pesi esponenzialmente decrescenti al crescere della distanza temporale dal momento di stima agli altri *batch*. Modificando i pesi dei dati non più disponibili diventa necessario salvare delle statistiche riassuntive di essi, in

particolare per ogni *batch* si richiede la conoscenza dei valori della funzione punteggio e della matrice di informazione osservata. Al divergere del tempo e all'aumentare del numero di dati si può avere quindi una richiesta eccessiva di spazio in memoria, sia pur sempre minore di quello necessario per salvare tutti i dati.

La soluzione più logica è quindi sembrata quella di troncare i pesi a zero dopo aver superato una certa distanza temporale k prefissata. Così facendo dopo l'arrivo del k-esimo *batch* lo spazio di memoria richiesto resta costante, dato che le statistiche riassuntive inerenti ai dati più vecchi di k non sono più richieste per l'aggiornamento delle stime. Purtroppo non è stato possibile applicare tale metodologia ai dati NBA, in quanto avendo a disposizione una matrice del modello molto sparsa diversi elementi della matrice di informazione osservata di ogni *batch* risultano identicamente nulli, impedendo la sua inversione e quindi la stima della matrice di varianza/covarianza.

In conclusione si può affermare che volendo modellare la dipendenza temporale delle osservazioni, a seconda della natura dei dati e della sparsità della matrice del modello, i migliori metodi da applicare sono l'utilizzo della *rho architecture* per dati pesati con pesi esponenziali crescenti riscalati o esponenziali crescenti riscalati e troncati oltre una soglia temporale k.

Un possibile prosieguo di questa tesi consiste nell'ampliare il metodo di aggiornamento delle stime con riduzione della distorsione di Firth basato sulla decomposizione QR, riuscendo ad eliminare la dipendenza dai dati passati. Tale metodologia infatti, pur non richiedendo che tali dati siano caricati in memoria, richiede che essi siano presenti in un *database* a cui sia possibile accedere al momento della stima del modello. Si può inoltre cercare di ampliare tutte le metodologie proposte basate sulla *rho architecture* per permettere alla matrice del modello di avere un numero di colonne variabile per ogni *batch*, rendendo possibile per esempio la valutazione di giocatori che hanno giocato esclusivamente nella seconda parte di stagione, e quindi che sono stati assenti per tutto il periodo preso in considerazione nella stima del modello iniziale.

Appendice

Codice R utilizzato

Codice .1: Librerie utilizzate.

```
library(tidyverse)
library(skimr)
library(ggplot2)
library(gridExtra)
library(lubridate)
library(fastDummies)
```

Codice .2: Preparazione dei dati.

```
dati <- read_csv("Dataset/2018-19/[10-16-2018]-[06-13-2019]-combined-stats.csv")
DataExplorer::plot_missing(dati)
# game_id
dati$game_id <- sub(
   pattern = ".",
   replacement = "",
   x = dati$game_id
)

# dataset
dati$data_set <- gsub(
   ".*Season.*",
   "Regular",
   dati$data_set
)
dati$data_set <- gsub(
   ".*Playoff.*",
   "Playoff",
   dati$data_set
)
dati$data_set <- factor(dati$data_set)

# remaining_time (secondi)
dati$remaining_time <- as.numeric(dati$remaining_time)

# elapsed (1-remaining_time)
dati$elapsed <- as.numeric(dati$elapsed)

# Controllo elapsed e remaining time
```

```
# Ogni inizio di (primo) supplementare ha un $elapsed pari a 420
dati[
   dati$period %>% as.numeric > 4 &
   dati$elapsed == 420,
]$elapsed <- 0

# play_length
dati[
   substr(dati$play_length, 4, 6) == "-12",
]$play_length <- "0:00:0"
dati[
   substr(dati$play_length, 4, 5) == "-5",
]$play_length <- "0:00:0"
dati[
   dati$play_length == "0",
]$play_length <- "0:00:00"
dati$play_length <- paste0("0", dati$play_length) %>% as.difftime %>% as.numeric

# event_type
dati <- dati[!is.na(dati$event_type),]
dati <- dati %>%
   filter(!(event_type %in% c("ejection", "timeout")))

# assist
dati$assist <- (!is.na(dati$assist)) %>% as.numeric()

# away
dati$away <- NULL

# home
dati$home <- NULL

# squadre
files <- list.files("Dataset/2018-19")[-1]
id <- substr(files, 14, 23)
team_1 <- substr(files, 25, 27)
team_2 <- substr(files, 29, 31)
teams <- tibble(
   game_id = id,
   team_1 = team_1,
   team_2 = team_2
)
dati <- dati %>% inner_join(teams, by = join_by(game_id))

# block (aggiunta righe)
i <- NROW(dati)
while (i>0) {
   if (!is.na(dati$block[i])){
      new_row <- dati[i,]
      new_row$remaining_time <- new_row$remaining_time - 1
      new_row$elapsed <- new_row$elapsed + 1
      new_row$play_length <- 1
      new_row$event_type <- "block"
      new_row$player <- new_row$block
      new_row$result <- NA
```

```
      new_row$type <- NA
      new_row$shot_distance <- NA
      new_row$original_x <- NA
      new_row$original_y <- NA
      new_row$converted_x <- NA
      new_row$converted_y <- NA
      new_row$description <- NA
      dati <- add_row(dati, .after = i, new_row)
   }
   i <- i - 1
   if (i%%500 == 0) cat(100 - i *100 / NROW(dati), "%\n")
}
dati$block <- NULL

# entered
dati$entered <- NULL

# left
dati$left <- NULL

# num
dati$num[dati$num %>% is.na] <- 0

# opponent
dati$opponent <- NULL

# outof
dati$outof[dati$outof %>% is.na] <- 0

# player
dati$player[dati$player %>% is.na] <- "General"

# possession
dati$possession <- NULL

# reason
dati$reason <- NULL

# result
dati$result[dati$result %>% is.na] <- "Nothing"

# steal
dati$steal <- (!(dati$steal %>% is.na)) %>% as.numeric

# description
dati$description <- NULL

# date

## Sistemo il formato mm/d/yyyy -> mm/dd/yyyy
dati$date[
   (dati$date %>% str_length == 9)
][
   (substr(
      x = dati$date[dati$date %>% str_length == 9],
```

```
      start = 3,
      stop = 3
   ) == "/")
]<-
   paste0(
      substr(
         x = dati$date[
            dati$date %>% str_length == 9
         ][
            (substr(
               x = dati$date[dati$date %>% str_length == 9],
               start = 3,
               stop = 3
            ) == "/")
         ],
         start = 1,
         stop = 3
      ),
      0,
      substr(
         x = dati$date[
         dati$date %>% str_length == 9
         ][
            (substr(
               x = dati$date[dati$date %>% str_length == 9],
               start = 3,
               stop = 3
            ) == "/")
         ],
      start = 4,
      stop = 9
      )
   )

## Sistemo il formato m/dd/yyyy -> mm/dd/yyyy
dati$date[
   (dati$date %>% str_length == 9)
][
   (substr(
      x = dati$date[dati$date %>% str_length == 9],
      start = 2,
      stop = 2
   ) == "/")
] <-
   paste0(
      0,
      dati$date[
         dati$date %>% str_length == 9
      ][
         (substr(
            x = dati$date[dati$date %>% str_length == 9],
            start = 2,
            stop = 2
         ) == "/")
      ]
```

```
   )

## Date con soli 8 caratteri
   dati$date[dati$date %>% str_length == 8] <-
      paste0(
         0,
         substr(
            x = dati$date[dati$date %>% str_length == 8],
            start = 1,
            stop = 2
         ),
         0,
         substr(
            x = dati$date[dati$date %>% str_length == 8],
            start = 3,
            stop = 8
         )
      )

## Uniformo il formato in yyyy-mm-dd
appoggio <- dati$date[
   substr(
      x = dati$date,
      start = 3,
      stop = 3
   ) == "/"
]
dati$date[
   substr(
      x = dati$date,
      start = 3,
      stop = 3
   ) == "/"
] <-
   paste0(
      substr(
         x = appoggio,
         start = 7,
         stop = 10
      ),
      "-",
      substr(
         x = appoggio,
         start = 1,
         stop = 2
      ),
      "-",
      substr(
         x = appoggio,
         start = 4,
         stop = 5
      )
   )
dati$date <- lubridate::as_date(dati$date)
```

```
# Players
players <- dati %>%
   dplyr::select(
      matches("^a[1-5]"),
      matches("^h[1-5]"),
      player
   ) %>%
   apply(2, as.character) %>%
   c %>%
   unique()

# Pongo tutti i giocatori con meno di 100 azioni a General
appoggio <- dati %>%
   group_by(player) %>%
   count() %>%
   arrange(desc(n)) %>%
   filter(n < 100) %>%
   select(player) %>%
   as.vector %>%
   unlist
dati$player[dati$player %in% appoggio] <- "General"
```

Codice .3: Costruzione della variabile risposta.

```
dati$y <- NA

i <- NROW(dati)
while (i >= 1) {
   l <- dati$play_length[i]
   if(dati$result[i] == "made"){
      # cat(i, ": made\n", sep = "")
      # cat(i, ": \t y = 1\n", sep = "")
      dati$y[i] <- 1
      team <- dati$team[i]
      i <- i-1
      while (l < 12 & dati$result[i] == "Nothing" & team == dati$team[i]) {
         # cat(i, ": \t y = 1\n", sep = "")
         dati$y[i] <- 1
         i <- i-1
         if (i == 0) break
         l <- l + dati$play_length[i]
      }
      # cat(i, ": uscito\n")
   } else {
      # cat(i, ": ", dati$result[i], " y = 0\n", sep = "")
      dati$y[i] <- 0
      i <- i - 1
   }
   if (i %% 500 == 0) cat(100 - i*100/NROW(dati), "%\n")
}
dati$event_type[dati$event_type %in% c("shot", "miss")] <- "shot"
```

Codice .4: Costruzione della matrice del modello.

```
finale <- dati %>%
```

```
   dplyr::select(
      game_id,
      period,
      away_score,
      home_score,
      remaining_time,
      team,
      event_type,
      player,
      foul,
      team_1,
      team_2,
      y
   )

mod_mat <- matrix(1, nrow = NROW(finale), ncol = 1)

# period
period <- dummy_cols(finale$period, remove_first_dummy = T)[-1]
c_names <- colnames(period)
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
colnames(period) <- str_c("period__", c_names)
mod_mat <- cbind(mod_mat, period)

# numeriche
mod_mat <- cbind(
   mod_mat,
   finale %>%
   dplyr::select(
      away_score,
      home_score,
      remaining_time
   )
)

# Team (Ref = ATL)
team <- dummy_cols(finale$team, remove_first_dummy = T)[-1]
(c_names <- team %>% colnames())
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
colnames(team) <- str_c("team__", c_names)
mod_mat <- cbind(mod_mat, team)

# Event_type (Ref = block)
event_type <- dummy_cols(finale$event_type, remove_first_dummy = T)[-1]
(c_names <- event_type %>% colnames())
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
(c_names <- gsub(" ", "_", c_names))
colnames(event_type) <- str_c("event_type__", c_names)
mod_mat <- cbind(mod_mat, event_type)

# Player (Ref = Aaron_Gordon)
player <- dummy_cols(finale$player, remove_first_dummy = T)[-1]
(c_names <- player %>% colnames())
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
(c_names <- gsub(" ", "_", c_names))
```

```
colnames(player) <- str_c("player__", c_names)
mod_mat <- cbind(mod_mat, player)

# Home_team (Ref = ATL)
home_team <- dummy_cols(finale$team_1, remove_first_dummy = T)[-1]
(c_names <- home_team %>% colnames())
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
colnames(home_team) <- str_c("home_team__", c_names)
mod_mat <- cbind(mod_mat, home_team)

# Away_team (Ref = ATL)
away_team <- dummy_cols(finale$team_2, remove_first_dummy = T)[-1]
(c_names <- away_team %>% colnames())
c_names <- substr(c_names, start = 7, stop = str_length(c_names))
colnames(away_team) <- str_c("away_team__", c_names)
mod_mat <- cbind(mod_mat, away_team)

non_unici <- apply(
mod_mat %>%
   filter(finale$game_id %in% unique(finale$game_id)[1:100]),
   2,
   function(x) length(unique(x)) != 1
) %>% which()
mod_mat <- mod_mat %>% dplyr::select(all_of(non_unici))

n_plays <- apply(
   mod_mat %>%
      dplyr::select(
         starts_with("player")
      ) %>%
      filter(
         finale$game_id %in% unique(finale$game_id)[1:100]
      ),
   2,
   sum
)
vars <- names(mod_mat)[
   -grep("player.*", names(mod_mat))[which(n_plays < 20)]
]
rm(away_team, event_type, home_team, period, player, team, c_names, n_plays, non_unici)
```

Codice .5: Divisione dei dati in *batch*.

```
save_data_mat <- function(){
   j <- 1
   for (i in unique(finale$game_id)) {
      X <- mod_mat %>%
         filter(finale$game_id == i) %>%
         as.matrix()
      y <- finale %>%
         filter(game_id == i) %>%
         dplyr::select(y) %>%
         as.vector() %>%
         unlist()
      save(
```

```
            X, y,
            file = paste0("Analisi/Game", " - ", j, ".RData")
        )
        if (j%%50 == 0)
            cat((j * 100 / length(unique(finale$game_id))) %>% round(2), "%\n")
        j <- j + 1
    }
}
save_data_mat()
```

Codice .6: Aggiornamento delle stime del modello logistico tramite *rho architecture*.

```
RenewGLM_out <-
function(X, y, type, betahat, infomats, intercept, s){
    tol <- 1e-6;
    max_iter <- 100;

    p <- dim(as.matrix(X))[2];

    betahat_old <- betahat;

    #record W in a list form rather than a matrix to simplify calculation
    W <- invlinkdiv(X, betahat_old, type=type)
    H <- cp(X,y,W) # t(X) %*% W %*% X

    U=chol(infomats+H)
    L=t(U)

    for (r in 1:max_iter){

        g_0=t(crossprod((y-invlink(X, betahat, type)),X));
        g_1=-t(crossprod((betahat-betahat_old),infomats));
        g <- g_0 + g_1;

        d_beta <- backsolve(U,forwardsolve(L,g))

        df_beta <- crossprod(g,d_beta);
        if (abs(df_beta)<tol){
            break
        }else {
            betahat <- betahat + drop(d_beta);
        }
    }

    W <- invlinkdiv(X,betahat,type=type)
    H_new <- cp(X,y,W)
    infomats<-infomats+H_new

    return(list(betahat,infomats))
}

cp <-
function(X, y, w){
```

```
   if (length(y)==1){
      H <- w*tcrossprod(X,X)
   }else{
      H <- crossprod(sqrt(w)*X)
   }
}

invlink <-
function(X, beta, type){
   if(type=="binomial"){ out <- exp(eta)/(1 + exp(eta)) }
   out
}

invlinkdiv <-
function(X, beta, type){
   if(type=="binomial"){ out <- exp(eta)/(1 + exp(eta))^2 }
   out
}

initials <- which(finale$game_id %in% unique(finale$game_id)[1:100])

m <- glm(finale$y[initials] ~ ., data = mod_mat[initials,], family = binomial)
sum_mod <- m %>% summary

coefficienti_ml <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
coefficienti_ml[,1] <- c("Intercept", names(mod_mat))
coefficienti_ml[,2] <- coef(m)
sd_ml <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
sd_ml[,1] <- c("Intercept", names(mod_mat))
sd_ml[,2] <- sum_mod$cov.unscaled %>% diag() %>% sqrt
tempdatadir <- "Analisi"
betahat <- m$coefficients
infomats <- sum_mod$cov.unscaled %>% solve
for (b in 101:length(unique(finale$game_id))) {
   X <- y <- NULL
   load(paste(tempdatadir,"/Game - ",b,".RData",sep=""))
   X<-cbind(1,X) # intercept
   summary<-RenewGLM_out(
      X = X,y = y,type = "binomial",betahat = betahat,
      infomats = infomats,intercept=TRUE
   )
   betahat<-summary[[1]]
   infomats<-summary[[2]]
   rm(X, y)
   coefficienti_ml[,b-98] <- betahat
   sd_ml[,b-98]<-sqrt(diag(solve(infomats)))
   if (b %% 100 == 0)
      cat((b - 100)*100/(length(unique(finale$game_id))-100), " %\n")
}
```

```
sd<-sqrt(diag(solve(infomats)))
pvalue<-2*pnorm(-abs(betahat)/sd)
result_out_ml<-cbind(betahat=betahat,sd=sd,pvalue=pvalue)
colnames(result_out_ml)<-c("Estimates","Std.Errors","p-values")
```

CODICE .7: Aggiornamento delle stime del modello logistico tramite *rho architecture* con pesi esponenzialmente crescenti.

```
RenewGLM_out_weights_forward <-
function(X, y, type, betahat, infomats, intercept, s, w, score){
   tol <- 1e-6;
   max_iter <- 100;

   p <- dim(as.matrix(X))[2];

   betahat_old <- betahat;

   #record W in a list form rather than a matrix to simplify calculation
   W <- invlinkdiv(X, betahat_old, type=type)
   H <- cp(X,y,W) # t(X) %*% W %*% X

   U=chol(infomats+ w * H)
   L=t(U)

   for (r in 1:max_iter){

      g_0= w * t(crossprod((y-invlink(X, betahat, type)),X));
      g_1=-t(crossprod((betahat-betahat_old),infomats));
      g <- g_0 + g_1;

      d_beta <- backsolve(U,forwardsolve(L,g))

      df_beta <- crossprod(g,d_beta);
      if (abs(df_beta)<tol){
         break
      }else {
         betahat <- betahat + drop(d_beta);
      }
   }

   W <- invlinkdiv(X,betahat,type=type)
   H_new <- w * cp(X,y,W)
   infomats<-infomats+H_new
   score <- score + w * g_0

   return(list(betahat,infomats, score))
}

br.model <- glm(
   ff, data = mm, method = "brglm_fit",
   family = binomial("logit"),
   epsilon = 1e-06
)

w_forward <- function(xi, t_k, t_1 = 1) {
```

```
   exp(xi * (t_k - t_1))
}
k <- (1:1312)
xi <- 0.01
w_k <- w_forward(xi, k)

coefficienti_br_w_f <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
coefficienti_br_w_f[,1] <- c("Intercept", names(mod_mat))
coefficienti_br_w_f[,2] <- coef(br.model.2)
sd_br_w_f <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
sd_br_w_f[,1] <- c("Intercept", names(mod_mat))
sd_br_w_f[,2] <- vcov(br.model) %>% diag() %>% sqrt
tempdatadir <- "Analisi"
betahat <- br.model$coefficients
infomats <- vcov(br.model) %>% solve
X <-  cbind(1, mod_mat[initials,]) %>% as.matrix()
score <- tcrossprod(
   (finale$y[initials]-invlink(X, betahat, "binomial")),
   X
)
rm(X)
for (b in 101:length(unique(finale$game_id))) {
   X <- y <- NULL
   load(paste(tempdatadir,"/Game - ",b,".RData",sep=""))
   X<-cbind(1,X) # intercept
   summary<-RenewGLM_out_weights_forward(
      X = X,y = y,type = "binomial",betahat = betahat,
      infomats = infomats,intercept=TRUE, w = w_k[b-99], score = score
   )
   betahat<-summary[[1]]
   infomats<-summary[[2]]
   score <- summary[[3]]
   rm(X, y)
   coefficienti_br_w_f[,b-98] <- betahat
   H <- tcrossprod(score)
   J <- solve(infomats)
   sd_br_w_f[,b-98] <- (J %*% H %*% J) %>% diag %>% sqrt()
   if (b %% 100 == 0)
      cat((b - 100)*100/(length(unique(finale$game_id))-100), " %\n")
}
sd.br<-sd_br_w_f[,NCOL(sd_br_w_f)] %>% as.numeric()
pvalue.br<-2*pnorm(-abs(betahat)/sd.br)
result_out.br_w_f<-cbind(betahat=betahat,sd=sd.br,pvalue=pvalue.br)
colnames(result_out.br_w_f)<-c("Estimates","Std.Errors","p-values")
```

CODICE .8: Aggiornamento delle stime del modello logistico tramite *rho architecture* con pesi esponenzialmente crescenti riscalati.

```
RenewGLM_out_weights_backward <-
function(X, y, type, betahat, infomats, intercept, s){
   tol <- 1e-6;
   max_iter <- 100;

   p <- dim(as.matrix(X))[2];

   betahat_old <- betahat;

   #record W in a list form rather than a matrix to simplify calculation
   W <- invlinkdiv(X, betahat_old, type=type)
   H <- cp(X,y,W) # t(X) %*% W %*% X

   U=chol(infomats + H)
   L=t(U)

   for (r in 1:max_iter){

      g_0= t(crossprod((y-invlink(X, betahat, type)),X));
      g_1=-t(crossprod((betahat-betahat_old),infomats));
      g <- g_0 + g_1;

      d_beta <- backsolve(U,forwardsolve(L,g))

      df_beta <- crossprod(g,d_beta);
      if (abs(df_beta)<tol){
         break
      }else {
         betahat <- betahat + drop(d_beta);
      }
      # cat(r, "\n")
   }

   W <- invlinkdiv(X,betahat,type=type)
   H_new <- cp(X,y,W)
   infomats<-infomats+H_new
   new_score <- g_0
   new_info <- H_new

   return(list(betahat,infomats, new_score, new_info))
}


j <- 1:1400
w_back <- function(xi, j){
   exp(-xi * j)
}
w_j <- w_back(0.01, j)

p <- dim(vcov(br.model))[1]
hessians <- array(0, dim = c(p, p, 1400))
hessians[,,1] <- vcov(br.model) %>% solve()
```

```
infomats <- hessians[,,1]

scores <- matrix(0, nrow = p, ncol = 1400)
X <- cbind(1, mod_mat[initials,]) %>% as.matrix()
betahat <- br.model$coefficients
score <- tcrossprod(
   (finale$y[initials]-invlink(X, betahat, "binomial")),
   X
)
rm(X)
scores[,1] <- score

coefficienti_br_w_b <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
coefficienti_br_w_b[,1] <- c("Intercept", names(mod_mat))
coefficienti_br_w_b[,2] <- coef(br.model.2)
sd_br_w_b <- matrix(
   NA, ncol = length(unique(finale$game_id)) - 98,
   nrow = NCOL(mod_mat) + 1
)
sd_br_w_b[,1] <- c("Intercept", names(mod_mat))
sd_br_w_b[,2] <- vcov(br.model) %>% diag() %>% sqrt
tempdatadir <- "Analisi"
betahat <- br.model$coefficients

for (b in 101:length(unique(finale$game_id))) {
   infomats <- matrix(0, p, p)
   score <- rep(0, p)
   for (i_w in 1:length(w_j)) {
      infomats <- infomats + drop(w_j[i_w]) * hessians[,,i_w]
      score <- score + drop(w_j[i_w]) * scores[,i_w]
   }

   X <- y <- NULL
   load(paste(tempdatadir,"/Game - ",b,".RData",sep=""))
   X<-cbind(1,X) # intercept
   summary<-RenewGLM_out_weights_backward(X = X,y = y,type = "binomial",betahat = betahat,
   infomats = infomats,intercept=TRUE)
   betahat<-summary[[1]]
   infomats<-summary[[2]]
   scores <- cbind(summary[[3]], scores)[,1:1400]
   score <- score + summary[[3]]
   hessians <- append(summary[[4]], hessians)
   hessians <- hessians[1:(length(hessians) - p*p)] %>%
      array(dim = c(p, p, 1400))

   rm(X, y)
   coefficienti_br_w_b[,b-98] <- betahat
   H <- tcrossprod(score)
   J <- solve(infomats)
   sd_br_w_b[,b-98] <- (J %*% H %*% J) %>% diag %>% sqrt()
   if (b %% 1 == 0)
      cat((b - 100)*100/(length(unique(finale$game_id))-100), " %\n")
```

```
}
sd.br<-sd_br_w_b[,NCOL(sd_br_w_b)] %>% as.numeric()
pvalue.br<-2*pnorm(-abs(betahat)/sd.br)
result_out.br_w_b<-cbind(betahat=betahat,sd=sd.br,pvalue=pvalue.br)
colnames(result_out.br_w_b)<-c("Estimates","Std.Errors","p-values")
```

Codice .9: Esempio di grafico.

```
plot.coefficients <- function(coef, which, nrow = 1, ncol = 1, remove_from_title = ""){
   if (!is.numeric(which)){
      which <- which(coef[,1] %in% which)
   }
   l <- length(which)
   p <- NULL
   y_low <- 0
   y_up <- 0
   for (i in which) {
      m2 <- reshape2::melt(
      data = coef[i,],
      id.vars = "Coefficient",
      variable.name = "Time",
      value.name = "Estimates"
      )
      m2[,3] <- as.numeric(m2[,3])
      y_low <- min(c(y_low, m2[,3]))
      y_up <- max(c(y_up, m2[,3]))
   }
   for (i in which) {
      m2 <- reshape2::melt(
      data = coef[i,],
      id.vars = "Coefficient",
      variable.name = "Time",
      value.name = "Estimates"
      )
      m2[,2] <- as.numeric(m2[,2]) + 99
      m2[,3] <- as.numeric(m2[,3])
      p[[i]] <- ggplot(m2, aes(x = Time, y = Estimates)) +
      geom_line() +
      geom_hline(yintercept = 0, lty = "dashed") +
      labs(title = sub("_", " ", sub(remove_from_title, "", coef[i,1])))+
      ylim(y_low, y_up) +
      scale_x_continuous(breaks=c(100, 350, 600, 850, 1100)) +
      theme(
      axis.title.x = element_text(size = 18),
      axis.title.y = element_text(size = 18),
      axis.text.x = element_text(size = 16),
      axis.text.y = element_text(size = 16),
      title = element_text(size = 20)
      )
   }
   n_plotted <- 0
   while (n_plotted < l) {
      grid_layout <- matrix(vector("list", nrow * ncol), nrow = nrow, ncol = ncol)
      if (n_plotted + nrow * ncol <= l) {
         for (i in 1:(nrow * ncol)) {
```

```
            grid_layout[[(i - 1) %% nrow + 1, (i  - 1) %/% nrow + 1]] <-
            p[[which[i] + n_plotted]]
            i
            grid_layout
         }
      } else {
         for (i in 1:(l - n_plotted)) {
            grid_layout[[(i - 1) %% nrow + 1, (i  - 1) %/% nrow + 1]] <-
            p[[which[i] + n_plotted]]
         }
      }
      grid.arrange(grobs = grid_layout, nrow = nrow, ncol = ncol)
      n_plotted <- n_plotted + nrow * ncol
   }
}

coefficienti_br_w_b <- coefficienti_br_w_b %>% as.data.frame()
names(coefficienti_br_w_b) <-
   c("Coefficient", 100:length(unique(finale$game_id)))

first_team <- c("James Harden", "Stephen Curry", "Giannis Antetokounmpo",
            "Paul George", "Nikola Jokic")
plot.coefficients(
   coefficienti_ml,
   which = sub(" ", "_", str_c("player__", first_team)),
   nrow = 2, ncol = 3,
   remove_from_title = "player__"
)
```

Bibliografia

Artuso, D. (2020). *Valutazione statistica delle performance dei giocatori della National Basketball Association*. Tesi di Laurea Magistrale, Dipartimento di Scienze Statistiche, Università degli Studi di Padova.

Bifet, A., Maniu, S., Qian, J., Tian, G., He, C. & Fan, W. (2015). StreamDM: Advanced Data Mining in Spark Streaming. In *2015 IEEE International Conference on Data Mining Workshop (ICDMW)*. ISSN: 2375-9259.

Big Data Ball (2019). Dataset NBA 2018/2019. URL: https://www.bigdataball.com/datasets/nba/play-by-play/.

Candès, E. J. & Sur, P. (2020). The phase transition for the existence of the maximum likelihood estimate in high-dimensional logistic regression. *The Annals of Statistics* **48**, 27–42.

Dandolo, D. (2019). *Valutazione statistica delle performance dei giocatori della Serie A*. Tesi di Laurea Magistrale, Dipartimento di Scienze Statistiche, Università degli Studi di Padova.

Dixon, M. J. & Coles, S. G. (1997). Modelling association football scores and inefficiencies in the football betting market. *Journal of the Royal Statistical Society Series C: Applied Statistics* **46**, 265–280.

Fahrmeir, L. & Kaufmann, H. (1985). Consistency and asymptotic normality of the maximum likelihood estimator in generalized linear models. *The Annals of Statistics* **13**, 342–368.

Firth, D. (1993). Bias reduction of maximum likelihood estimates. *Biometrika* **80**, 27–38.

Jeffreys, H. (1946). An invariant form for the prior probability in estimation problems. *Proceedings of the Royal Society of London. Series A. Mathematical and Physical Sciences* **186**, 453–461.

Jorgensen, B. (1997). *The Theory of Dispersion Models*. Monographs on statistics and applied probability. London: Chapman & Hall.

Kaplan, J. (2020). *fastDummies: Fast Creation of Dummy (Binary) Columns and Rows from Categorical Variables*. R package version 1.6.3.

Kosmidis, I. (2023). *brglm2: Bias Reduction in Generalized Linear Models*. R package version 0.9.

Kosmidis, I. & Firth, D. (2021). Jeffreys-prior penalty, finiteness and shrinkage in binomial-response generalized linear models. *Biometrika* **108**, 71–82.

Kosmidis, I., Kenne Pagui, E. C. & Sartori, N. (2020). Mean and median bias reduction in generalized linear models. *Statistics and Computing* **30**, 43–59.

Luo, L. & Song, P. X. K. (2020). Renewable estimation and incremental inference in generalized linear models with streaming data sets. *Journal of the Royal Statistical Society Series B: Statistical Methodology* **82**, 69–97.

Luo, L. & Song, P. X. K. (2023). Multivariate online regression analysis with heterogeneous streaming data. *Canadian Journal of Statistics* **51**, 111–133.

Luo, L., Wang, J. & Hector, E. C. (2023). Statistical Inference for Streamed Longitudinal Data. *Biometrika* , to appear.

Luo, L., Zhou, L. & Song, P. X. K. (2022). Real-time regression analysis of streaming clustered data with possible abnormal data batches. *Journal of the American Statistical Association* , to appear.

Marz, N. & Warren, J. (2015). *Big Data: Principles and Best Practices of Scalable Real-time Data Systems*. New York: Manning Publications.

McCullagh, P. & Nelder, J. A. (1983). *Generalized Linear Models*. London: Chapman and Hall.

Miller, A. J. (1992). Algorithm AS 274: Least Squares Routines to Supplement Those of Gentleman. *Applied Statistics* **41**, 458.

Miller, A. J. (1994). Correction to Algorithm AS 274: Least Squares Routines to Supplement those of Gentleman. *Applied Statistics* **43**, 678.

Pace, L., Salvan, A. & Sartori, N. (2022). *Statistical Inference: Theory and Methods*. Dispensa didattica.

R Core Team (2023). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Robbins, H. & Monro, S. (1951). A Stochastic Approximation Method. *The Annals of Mathematical Statistics* **22**, 400–407.

Salvan, A., Sartori, N. & Pace, L. (2020). *Modelli Lineari Generalizzati*. Milano: Springer.

Song, P. X.-K., Fan, Y. & Kalbfleisch, J. D. (2005). Maximization by parts in likelihood inference. *Journal of the American Statistical Association* **100**, 1145–1158.

Terner, Z. & Franks, A. (2021). Modeling player and team performance in basketball. *Annual Review of Statistics and Its Application* **8**, 1–23.

Toulis, P., Airoldi, E. & Rennie, J. (2014). Statistical analysis of stochastic gradient methods for generalized linear models. In *Proceedings of the 31st International Conference on Machine Learning*. PMLR. ISSN: 1938-7228.

Varin, C., Reid, N. & Firth, D. (2011). An overview of composite likelihood methods. *Statistica Sinica* **21**, 5–42.

Zietkiewicz, P. & Kosmidis, I. (2023). Bounded-memory adjusted scores estimation in generalized linear models with large data sets. ArXiv:2307.07342.