



UNIVERSITÀ DI PISA

Dipartimento di Informatica

Corso di Laurea Triennale in Informatica

TESI DI LAUREA

Analisi di serie temporali per il monitoraggio di bambini
con emiplegia

Relatori:

Prof. Giuseppe Prencipe
Prof.ssa Alina Sîrbu
Ing. Silvia Filogna

Candidati:

Davide Marchi
Giordano Scerra

Controrelatore:

Prof. Vincenzo Lomonaco

ANNO ACCADEMICO 2021/2022

Indice

1 INTRODUZIONE	1
1.1 Motivazioni e obiettivi	1
1.2 I nostri contributi	1
1.3 Contenuto dei capitoli	2
2 FONDAMENTI TEORICI	3
2.1 Machine Learning	3
2.1.1 Modello	3
2.1.2 Dataset	3
2.1.3 Model selection	5
2.1.4 Model assessment	7
2.1.5 Metriche di valutazione e Indici di correlazione	8
2.1.6 Apprendimento non supervisionato	9
2.1.7 Apprendimento supervisionato	11
2.2 Applicazioni del ML in ambito clinico	16
2.2.1 Background clinico	17
3 DATASET E PREPROCESSING	19
3.1 Dati di partenza	19
3.1.1 Sessioni AHA	20
3.1.2 Sessioni WEEK	21
3.2 Estrazione delle Features	21
3.2.1 Suddivisione in intervalli temporali	22
3.2.2 Manipolazione delle serie temporali	23
4 ANALISI	26

4.1	Pipeline di lavoro	26
4.2	Continous Performance Indicator (CPI)	26
4.2.1	Model Selection	29
4.2.2	Valutazione dei risultati	34
4.2.3	Calcolo del CPI	38
4.3	Home Assisting Hand Assessment	42
4.3.1	Andamento del CPI	42
4.3.2	Stimare l’AHA: Home-AHA	51
4.3.3	Realizzazione del dashboard	54
5	RISULTATI	61
5.1	CPI come feature	61
5.1.1	Modularità della pipeline	61
5.2	Grafico dell’Home-AHA	61
5.2.1	Activity recognition: notti	62
6	CONCLUSIONI	63
6.1	Margine di miglioramento	63
6.2	Sviluppi Futuri	63
6.2.1	Quantità e qualità dei dati	63
6.2.2	Esplorazione di ulteriori modelli e parametri	64
6.2.3	Gestione e trasferimento dei dati	64

Elenco delle figure

2.1	Esempio di serie temporale [1]	4
2.2	Suddivisione del dataset durante una 5-fold cross validation [2]	6
2.3	Esempio di <i>underfitting</i> , <i>fit ottimale</i> e <i>overfitting</i> [3]	7
2.4	Gerarchia tra alcune delle metriche di valutazione [4]	8
2.5	Esempio di identificazione dell'elbow point [5]	10
2.6	Esempio di output di un algoritmo di clustering [6]	11
2.7	Pipeline dell'algoritmo ShapeDTW [7]	13
2.8	Compressione di una serie temporale in un istogramma di parole tramite BOSS [8]	15
2.9	Esempio di regressione lineare, con variabile di input sulle ascisse e variabile target sulle ordinate. [9]	16
3.1	Foto del modello dell'attigrafo utilizzato per il rilevamento dei dati [10].	19
3.2	Grafico delle magnitudo dello spostamento delle mani di un soggetto con PCI durante una sessione AHA.	22
3.3	Grafico delle magnitudo dello spostamento delle mani di un soggetto del gruppo di controllo durante una sessione AHA.	22
3.4	Concatenazione delle serie temporali mostrate nella figura 3.2.	24
3.5	Concatenazione delle serie temporali mostrate nella figura 3.3.	24
3.6	Differenza delle serie temporali mostrate nella figura 3.2.	24
3.7	Differenza delle serie temporali mostrate nella figura 3.3.	25
3.8	Asymmetry Index delle serie temporali mostrate nella figura 3.2.	25
3.9	Asymmetry Index delle serie temporali mostrate nella figura 3.3.	25
4.1	Rappresentazione grafica della pipeline di lavoro.	26

4.2	Tentativo di suddivisione multi-cluster utilizzando KMeans su sample da sessioni AHA da 900 secondi.	28
4.3	Elbow plot simil-lineare ottenuto da esecuzioni KMeans.	29
4.4	Scatterplot che rappresenta la correlazione tra punteggio AHA e CPI su sessioni WEEK.	41
4.5	Predizioni sui sample della sessione week di un soggetto con emiplegia.	43
4.6	Predizioni sui sample della sessione week di un soggetto del gruppo di controllo.	43
4.7	Andamento del CPI tenendo conto solo delle finestre con significatività $\geq 75\%$	44
4.8	Andamento del CPI tenendo conto di tutte le finestre.	45
4.9	Andamento del CPI in finestre disgiunte da 6 ore nella sessione week di un soggetto con emiplegia.	47
4.10	Andamento del CPI in finestre disgiunte da 6 ore nella sessione week di un soggetto del gruppo di controllo.	47
4.11	Andamento del CPI in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.	49
4.12	Andamento del CPI in finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.	49
4.13	Andamento della significatività di finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.	50
4.14	Andamento della significatività di finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.	50
4.15	Andamento del CPI calcolato con 5 modelli differenti in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.	52
4.16	Andamento del CPI calcolato con 5 modelli differenti in finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.	52

4.17 Andamento dell'Home-AHA calcolato da un regressore allenato su 5 modelli in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.	54
4.18 Andamento dell'Home-AHA calcolato da un regressore allenato su 5 modelli in finestre scorrevoli da 6 ore nella sessione week di un soggetto appartenente al gruppo di controllo.	54

1. INTRODUZIONE

1.1 Motivazioni e obiettivi

L'intelligenza artificiale insieme all'utilizzo di sensori indossabili sta assumendo un ruolo sempre più centrale all'interno dell'ambiente clinico. Sorge dunque la necessità di ottenere una valutazione quantitativa e oggettiva del comportamento dei soggetti anche durante la loro vita quotidiana, tramite l'utilizzo di strumenti informatici all'avanguardia, nel tentativo di eliminare o ridurre così ogni imprecisione e inconsistenza associata ad una valutazione legata all'uomo e di conseguenza naturalmente soggettiva.

L'obiettivo di questo studio è ottenere un metodo di monitoraggio per i soggetti affetti da *Paralisi Cerebrale a tipo emiplegia* fuori dall'ambiente strutturato della valutazione in laboratorio. Questo per comprendere se durante la vita quotidiana il soggetto si comporti in maniera conforme a quanto osservato nei test clinici, in termini di asimmetria nell'utilizzo degli arti superiori, e se ci siano differenze nell'utilizzo degli arti (superiori) nel proprio ambiente naturale.

Il nostro lavoro si colloca nel contesto del progetto di ricerca europeo AInCP (“Clinical validation of Artificial Intelligence for providing a personalized motor clinical profile assessment and rehabilitation of upper limb in children with unilateral Cerebral Palsy”), uno studio che si propone di sfruttare il potenziale dell'intelligenza artificiale per creare tools di supporto nella diagnosi clinica di bambini con emiplegia.

1.2 I nostri contributi

Il nostro contributo nello specifico si è concentrato in:

- Realizzazione di una pipeline per poter allenare differenti algoritmi di intelligenza artificiale a partire da dati cinematici.

- Introduzione di un sistema per valutare tali modelli indipendentemente dalla loro tipologia.
- Definizione di un nuovo indice delle prestazioni (Continous Performance Indicator, CPI) ottenuto applicando gli algoritmi allenati sui dati ottenuti durante i test clinici ai dati cinematici registrati durante una settimana di vita quotidiana dei soggetti.
- Realizzazione di un dashboard che mostri l’andamento, oltre a quello del CPI, di un nuovo indicatore che possa stimare la valutazione clinica AHA (Home-AHA).

Il codice relativo al nostro lavoro è liberamente consultabile su [GitHub](#) [11].

1.3 Contenuto dei capitoli

Il contenuto della tesi è strutturato secondo i seguenti capitoli:

- Nel capitolo ”Fondamenti teorici” si presenta lo stato dell’arte e le basi teoriche necessarie per la piena comprensione delle terminologie cliniche e delle metodologie applicate durante la tesi.
- Nel capitolo ”Dataset e preprocessing” si osservano i dati di laboratorio e settimanali e le metodologie utilizzate per la manipolazione degli stessi.
- Nel capitolo ”Analisi” si presenta la pipeline lavorativa nelle sue due fasi fondamentali: il training dei modelli e la loro applicazione per creare il dashboard.
- Nel capitolo ”Risultati” si discute l’output dell’applicazione degli algoritmi e ciò che può esserne dedotto.

2. FONDAMENTI TEORICI

In questo capitolo si introducono alcuni concetti basilari, terminologie e lo stato dell’Arte per quanto riguarda le tecnologie utilizzate. Questa sezione non è da considerarsi come una panoramica completa su tutto quello che è il Machine Learning o la nozione di Paralisi Cerebrale Infantile a tipo emiplegia, ma piuttosto come una guida per introdurre sinteticamente i concetti e i principi utilizzati in questo lavoro.

2.1 Machine Learning

Quando si parla di Machine Learning (ML) si fa riferimento ad un sottoinsieme della più ampia branca dell’Intelligenza Artificiale (IA) che si occupa della realizzazione di tools che apprendono o migliorano le performance in base ad un pool di dati su cui vengono addestrati.

2.1.1 Modello

I tools sopracitati prendono il nome di modelli. Un modello è una rappresentazione matematica di un fenomeno del mondo reale, e più nello specifico è un insieme di algoritmi e tecniche statistiche per l’analisi di dati, l’identificazione di patterns e l’etichettatura di nuovi dati “non visti”.

Un modello di ML può essere addestrato con l’obiettivo di effettuare predizioni sul tipo di dato sopracitato.

2.1.2 Dataset

Un dataset nel ML è una collezione di osservazioni, informazioni o esempi, rappresentati da variabili e features. A seconda del singolo caso un dataset può assumere

differenti formati, ad esempio: file testuali, immagini, registrazioni audio, video, o serie temporali.

Il dataset su cui si opera è fondamentale per le prime fasi di una pipeline ML. Infatti, questi dati di lavoro possono essere di natura diversa e richiedere differenti manipolazioni. La qualità del dataset è di vitale importanza per l'efficacia di un progetto di ML, dato che il risultato finale è profondamente dipendente dai dati di input, dato che ne consegue la qualità dell'addestramento di un modello.

Time series

Una “time series” (“serie temporale”), è una serie di punti legati ad istanti che seguono l’andamento incrementale del tempo: infatti la serie temporale può essere rappresentata come una lista di coppie (tempo, valore) dove il tempo rappresenta in un certo senso la posizione dell’osservazione in quella sequenza di istanti. Esempi tipici di serie temporali possono essere l’andamento delle quotazioni di borsa, il segnale di un elettrocardiogramma (ECG) o semplicemente una variabile che cambia il suo valore nel tempo.

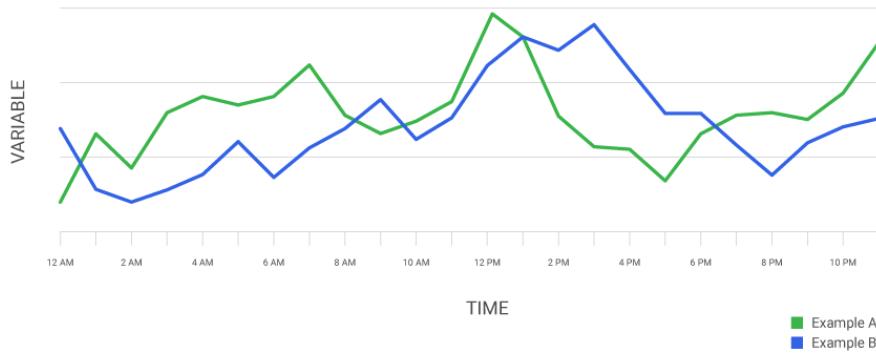


Figura 2.1: Esempio di serie temporale [1]

Preprocessing

Il preprocessing in ML, ovvero la pre-elaborazione, definisce una delle prime fasi che precede l’addestramento di un modello. Questo è uno step essenziale per l’efficacia

di un algoritmo di ML, dato che alcuni dati così come sono (ossia “grezzi”) non sono adatti ad essere elaborati da un algoritmo ma necessitano una preparazione, in maniera tale da poter essere utilizzati in maniera efficiente dal modello. Questo aumenterà probabilmente la sua robustezza e la sua capacità di generalizzazione.

Alcune tra le tecniche utilizzate per preparare i dati all’analisi sono:

- Pulizia dei dati: spesso i dati sono molto rumorosi. Rimuovere dati errati, mancanti o duplicati può incrementare l’efficienza della pipeline.
- Scaling: i dati possono essere distribuiti su scale diverse. Per questo i valori si possono amplificare o diminuire (“scalare”), portandoli tutti nello stesso range tramite tecniche come la normalizzazione.
- Feature engineering: partendo dalle variabili esistenti si possono creare ulteriori variabili che arricchiscono il processo con nuove informazioni legate alla natura intrinseca dei dati.
- Feature selection: esistono feature più rilevanti di altre e si possono dunque selezionare le migliori (ad esempio tramite un’analisi delle correlazioni, o secondo altre tecniche di preprocessing).

2.1.3 Model selection

La model selection è una fase fondamentale per lo sviluppo di un modello di ML. Questa fase importante si riferisce al processo di selezione dell’algoritmo più adatto a risolvere un dato problema.

Questa selezione è essenzialmente un ciclo tra queste fasi:

- Scelta della classe del modello: scelta dell’algoritmo più idoneo a risolvere il problema.
- Training: fase di allenamento dove il modello viene allenato su un parte di dati, non coincidenti con quelli di validazione.

- Validation: fase di validazione dove il modello appena allenato viene valutato secondo una metrica specifica su dati “nuovi” non coincidenti con quelli di training.

Spesso il training e la validation sono implementati con una Cross Validation: si suddividono i dati riservati alla model selection in k insiemi di validazione (folds) di dimensione uguale per valutare le prestazioni del modello su una parte di ciascun insieme. Dopodiché si raccolgono i singoli score di ogni gruppo e si effettua una media, stimando così una valutazione totale. Il modello sarà poi riallenato su tutti i dati.

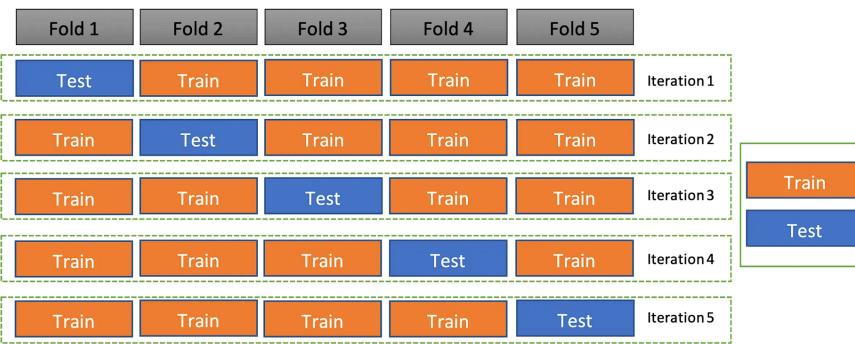


Figura 2.2: Suddivisione del dataset durante una 5-fold cross validation [2]

Lo scopo della model selection è scegliere il modello migliore in termini di semplicità ed efficienza, affinando ad ogni iterazione i parametri di training oppure più drasticamente cambiando la classe dell’algoritmo. Spesso la model selection viene implementata con una Grid-Search, che significa “ricerca in una griglia” al cui interno si trovano una serie di combinazioni di parametri di fitting su cui vengono allenati i modelli con il loro score di validation. Dopo aver scorso tutta la griglia, si restituisce in output il modello con i parametri che massimizzano lo score sul validation set. Una buona model selection aiuta a prevenire il rischio di *overfitting* ed *underfitting*, di cui parleremo meglio nella sezione successiva.

2.1.4 Model assessment

Il model assessment è l'ultima fase della costruzione di un modello di ML e si occupa di testare il modello su un set di dati nuovi “mai visti” nella fase di model selection. L'output del model assessment, calcolato attraverso una metrica specifica arbitraria, descrive l'accuratezza dell'algoritmo nel risolvere il problema per il quale è stato progettato e ci dà informazioni riguardo alla sua capacità di generalizzazione.

Un errore da evitare e che a volte può essere non banale da riconoscere nel ML è quello di accordare i parametri di fitting secondo il punteggio nella fase di model assessment, allo scopo di massimizzare quest'ultimo. Si incorrerebbe così nel problema dell'*overfitting*, ossia aumentare la complessità del modello in modo tale da avere uno score quasi perfetto nella valutazione dei dati di test. Così facendo non solo l'output del model assessment risulterebbe completamente privo di significato, ma il modello diverrebbe incapace di mantenere quello stesso score falsato su qualsiasi altro dato di test, come se l'algoritmo fosse “specializzato” proprio su quello specifico dataset.

D'altra parte, il problema opposto al sovradattamento è classificato come *underfitting* e si manifesta quando il modello non è capace di approssimare in maniera soddisfacente in primis i dati di training. Questo problema si ovvia con un affinamento ulteriore dei parametri di training o con un aumento della complessità del modello.

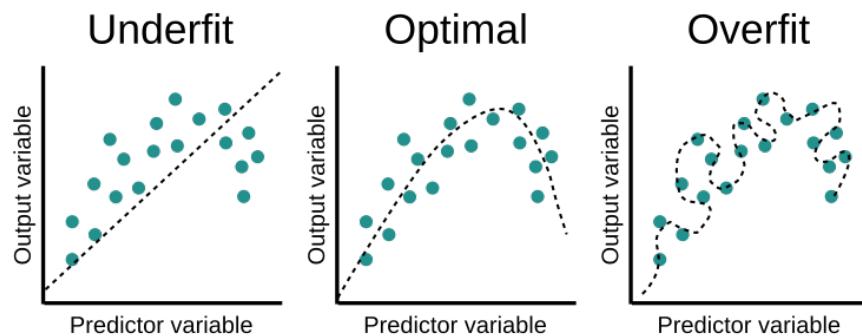


Figura 2.3: Esempio di *underfitting*, *fit ottimale* e *overfitting* [3]

2.1.5 Metriche di valutazione e Indici di correlazione

Quando parliamo di metriche per la valutazione di un modello, si fa riferimento a funzioni specifiche che ci permettono di approssimare l'accuratezza o l'errore di una predizione in relazione a un certo dato di riferimento, chiamato “target”.

Esempi di metriche di valutazione sono:

- Precision: misura la proporzione di predizioni corrette rispetto al numero totale di predizioni fatte dal modello.
- Recall: misura il rapporto tra il numero di predizioni corrette e il numero di elementi effettivamente appartenenti alla classe.
- F1 score: media armonica tra precisione e recall. È particolarmente utile in contesti in cui è importante ottenere un risultato equilibrato tra recall e precisione.
- R^2 score: misura il legame tra la variabilità dei dati e la correttezza del modello statistico utilizzato per rappresentarli. Si calcola sottraendo ad 1 il rapporto tra la devianza residua e la devianza totale.

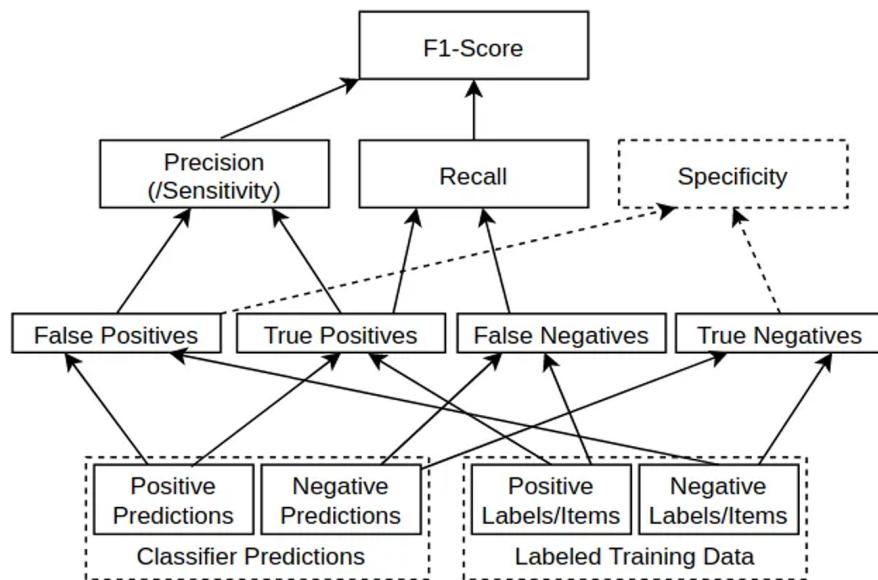


Figura 2.4: Gerarchia tra alcune delle metriche di valutazione [4]

Oltre alle metriche di valutazione, esistono anche indici che rappresentano la correlazione tra due variabili.

Un esempio di indice di correlazione è:

- Correlazione di Pearson: misura la relazione lineare tra due variabili continue, o tra due feature, è utile per identificare quelle che hanno una forte relazione con l'output del modello. Si calcola dividendo la covarianza delle due variabili per il prodotto delle loro deviazioni standard.

2.1.6 Apprendimento non supervisionato

L'apprendimento non supervisionato è una tecnica di apprendimento automatico che consiste nell'addestrare un modello utilizzando un dataset in cui i dati di input non presentano alcun target associato. Per questo motivo un algoritmo unsupervised si limita principalmente a trovare similitudini tra i dati di input ad esempio tramite un'analisi dei patterns.

Clustering

Il clustering nello specifico consente nella ricerca unsupervised di pattern e caratteristiche comuni all'interno di un dataset al fine di raggruppare tra di loro i dati più simili. I gruppi in cui vengono suddivisi i dati prendono il nome di cluster.

Di seguito si presentano alcuni algoritmi di clustering utilizzati all'interno del nostro lavoro.

KMeans

K-Means è un algoritmo di clustering utilizzato per dividere un dataset in gruppi omogenei, cercando di minimizzare la varianza all'interno di ogni cluster e di massimizzare la varianza tra cluster differenti. “K” nel nome sta ad indicare il numero di insiemi in cui si vogliono suddividere i dati, dunque un problema di clustering binario avrà K=2.

Dopo aver scelto il numero di cluster ed aver creato in maniera randomica K-“centroidi”, ossia i centri dei cluster, K-Means consiste essenzialmente in questo ciclo, fino a quando non si raggiunge una data condizione di convergenza:

- Calcolo della distanza Euclidea tra i centroidi ed ogni punto del dataset.
- Assegnamento dei punti ai cluster il cui centroide è più vicino.
- Calcolo di nuovi centroidi per ogni cluster, come la media dei punti assegnati a quel cluster.

Esistono diverse tecniche per scegliere il numero di cluster per questo algoritmo: la più comune è il cosiddetto Elbow Method, che previa esecuzione dell’algoritmo con un numero incrementale di clusters, analizza l’andamento dell’inerzia, cioè la somma dei quadrati delle distanze tra ciascun punto del dataset e il centroide del cluster più vicino, in modo tale da selezionare il K che ne massimizza il decremento tra un’esecuzione e l’altra, creando così un “gomito” (*elbow point*) in un grafico inerzia-K, da cui il nome.

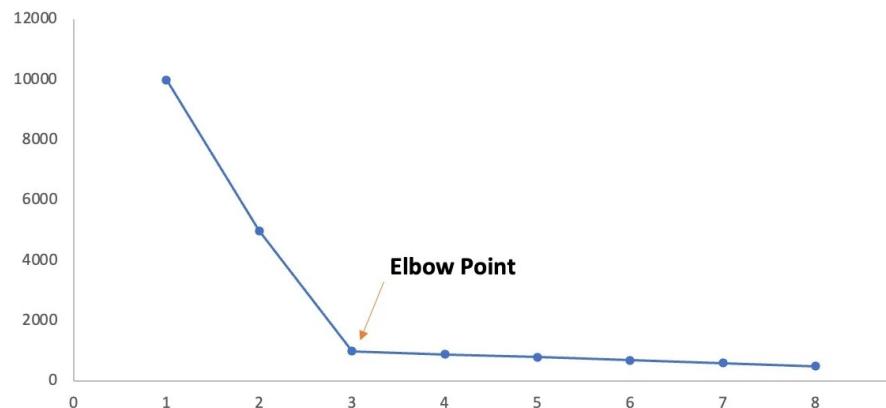


Figura 2.5: Esempio di identificazione dell’elbow point [5]

KMedoids

K-Medoids è un algoritmo di clustering con le medesime intenzioni di K-Means, ma differentemente da questo si basa sulla scelta di un medoide come rappresentante di

ogni cluster invece di un centroide. Rispetto al K-Means ha il vantaggio di essere più robusto alla presenza di valori anomali nei dati, ma può risultare più oneroso a livello computazionale.

K-Medoids conserva il ciclo di esecuzione presentato per il K-Means, ma:

- Utilizza la distanza di Manhattan¹ per calcolare la distanza tra i punti e il medoide, al posto di quella Euclidea
- per il calcolo del nuovo medoide cerca di minimizzare la somma delle distanze tra il medoide e i punti interni a quel cluster, al posto di calcolare la media dei punti per il nuovo centroide.

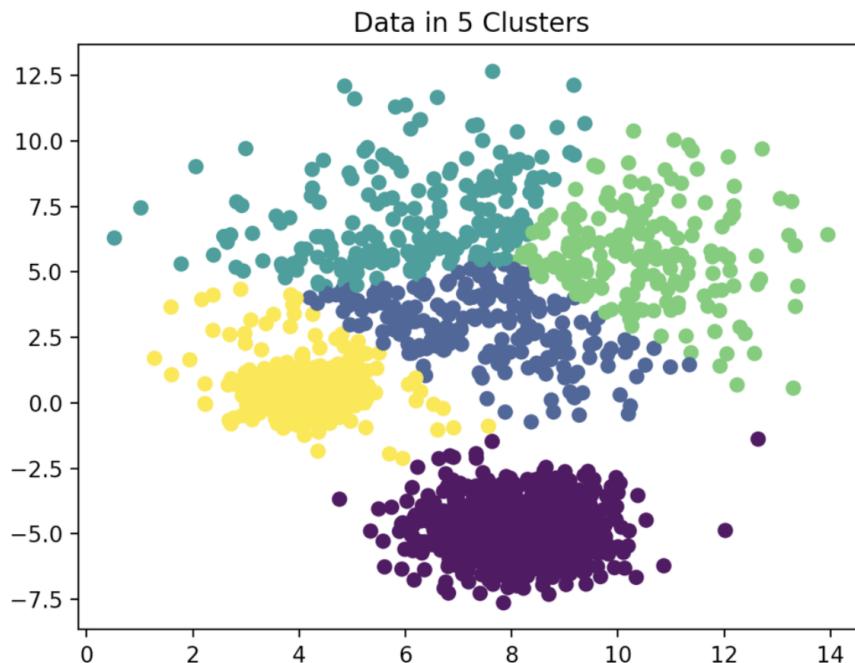


Figura 2.6: Esempio di output di un algoritmo di clustering [6]

2.1.7 Apprendimento supervisionato

L'apprendimento supervisionato descrive una famiglia di algoritmi progettata per lavorare con un input della forma (dato, target), chiamato esempio. Lo scopo è quello di

¹Somma delle differenze assolute tra le coordinate dei punti

identificare una funzione matematica che associa gli input ai corrispondenti output. Nel caso della classificazione binaria, il target sarà ad esempio 0 o 1 per un certo input: ad esempio se il problema consiste nel classificare se un certo animale è erbivoro o meno allora l'input (cane, False) è un esempio conforme al paradigma supervised. Dopo aver appreso la relazione matematica, l'algoritmo sarà dunque in grado di effettuare previsioni su nuovi dati di input.

Classificazione

Nel ML un problema è detto di classificazione quando il modello da progettare si deve preoccupare di categorizzare un qualche dato di input in due o più insiemi di output.

Si tratta di classificazione binaria se le classi di output sono esattamente due, ad esempio True o False o più semplicemente 0 o 1. Mentre ad esempio un algoritmo che decide a quale famiglia di mammiferi appartiene un certo animale è un algoritmo di classificazione multclasse.

Di seguito si presentano alcuni algoritmi di classificazione utilizzati all'interno del nostro lavoro.

ShapeDTW

ShapeDTW è utilizzato nel ML per il confronto di serie temporali allo scopo di identificare la migliore corrispondenza tra due serie temporali anche di forma diversa (traslate, scalate), e più in generale per il riconoscimento di pattern. A differenza del DTW (Dynamic Time Warping, algoritmo su cui è basato ShapeDTW), questo modello tiene conto anche delle differenze di forma tra le serie. ShapeDTW può essere utilizzato come misura di distanza in un classificatore Nearest Neighbour² (NN-shapeDTW).

ShapeDTW è basato sui seguenti passi:

- Sampling in finestre fisse della serie temporale

²Classificatore che tenta di trovare il punto del training dataset più vicino a un punto di prova e utilizzare la sua etichetta come previsione per quel punto.

- Codifica delle finestre in shape descriptors (descrittori di forma), che ne riasumono la struttura locale. Una serie sarà così interamente codificata in una sequenza di shape descriptors.
- Allineamento delle sequenze di descrittori di forma tramite DTW.
- Applicazione della sequenza di allineamento ottimale alla serie originale.

Allo scopo di classificare serie temporali, la misura di somiglianza in output è utilizzata come misura di distanza in un classificatore NN-shapeDTW. Per diversi dataset, questa versione di ShapeDTW migliora significativamente le prestazioni dell'algoritmo DTW, che come già detto non tiene conto delle distorsioni locali legate alla forma delle serie e per questo incappa spesso in errori di allineamento. [7]

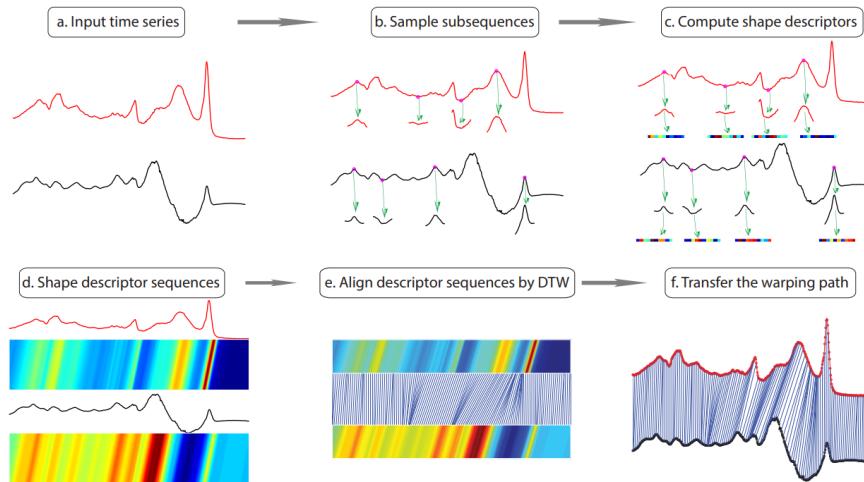


Figura 2.7: Pipeline dell'algoritmo ShapeDTW [7]

BOSSEnsemble

BossEnsemble (Bag-of-Symbolic Fourier Approximation-Symbols) è un algoritmo di ML utilizzato per la classificazione di serie temporali. Si basa su un approccio di tipo ensemble, ovvero unisce le previsioni di più algoritmi BOSS per ottenere una previsione finale più accurata. Questo tipo di modello utilizza una misura di similarità basata

sulla struttura, applica una riduzione del rumore e fornisce una rappresentazione compatta della serie temporale, rendendolo utile per la classificazione di grandi dataset di serie temporali.

Un singolo algoritmo BOSS implementa i seguenti passi:

- Suddivide ogni serie temporale in finestre di lunghezza fissa.
- Ogni finestra viene compressa in una parola di lunghezza l attraverso una trasformata di Fourier e mantenendo i primi $l/2$ coefficienti complessi.
- Questi l coefficienti vengono quindi discretizzati in α possibili valori per formare una parola di lunghezza l .
- Viene memorizzato un istogramma di parole per ogni serie, che ne rappresenta una versione compatta.
- Viene calcolata una misura di similarità tra due serie, utilizzando la distanza Euclidea normalizzata tra 0 e 1, dove 1 significa che le due serie sono identiche.

In sintesi, BOSSEnsemble esegue una ricerca a griglia su un insieme di valori di parametro, valutando ognuno con una validazione incrociata a esclusione singola (LOOCV), dopodiché conserva l'ensemble dei migliori modelli per la classificazione di serie temporali. [8]

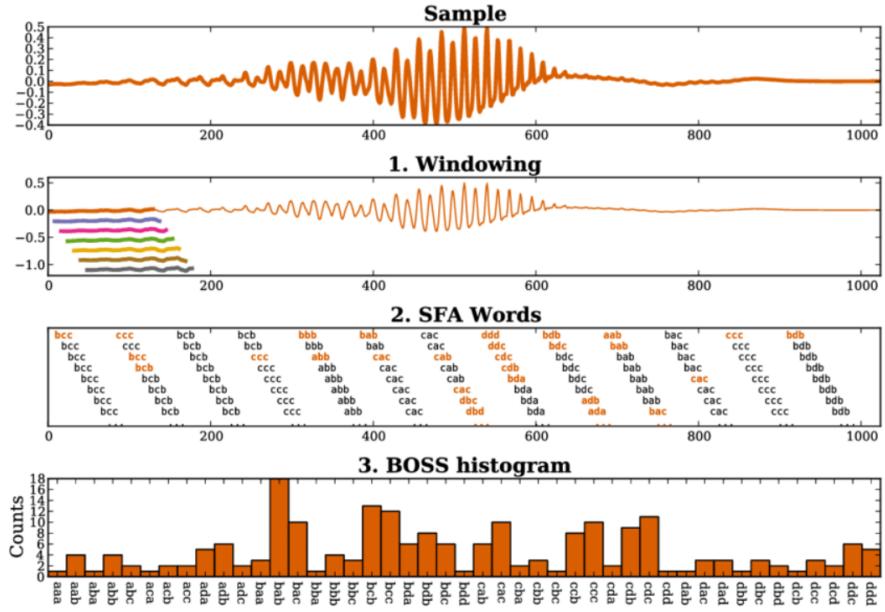


Figura 2.8: Compressione di una serie temporale in un istogramma di parole tramite BOSS [8]

Regressione

Quando si parla di regressione, a differenza della classificazione, non si sta cercando di capire a quale classe appartenga un input, ma bensì si sta cercando di ottenere, a partire dall'input, un valore (target) che vi sia in qualche modo collegato.

L'obiettivo della regressione diventa dunque quello di realizzare un modello matematico che sia in grado di rappresentare una correlazione tra le features disponibili dell'input e quello che invece è il valore atteso.

Esistono diversi tipi di regressione, nello specifico quella utilizzata all'interno del nostro lavoro è la regressione lineare.

Regressore lineare

Un regressore lineare cerca di costruire una retta che rappresenti la relazione tra la variabile target e la variabile in input nel miglior modo possibile.

Questo meccanismo è sviluppato all'interno di un ciclo dove ad ogni iterazione si cerca il più possibile di minimizzare l'errore quadratico medio tra i valori predetti dal modelli e gli effettivi valori attesi.

Una volta addestrato il regressore e trovati i coefficienti che diminuiscono il più possibile l'errore, questo è utilizzabile per ottenere delle stime più o meno valide a seconda dell'effettiva correlazione tra valori in input e targets attesi, il numero di esempi nel dataset, la pulizia dei dati e più in generale la qualità del preprocessing.

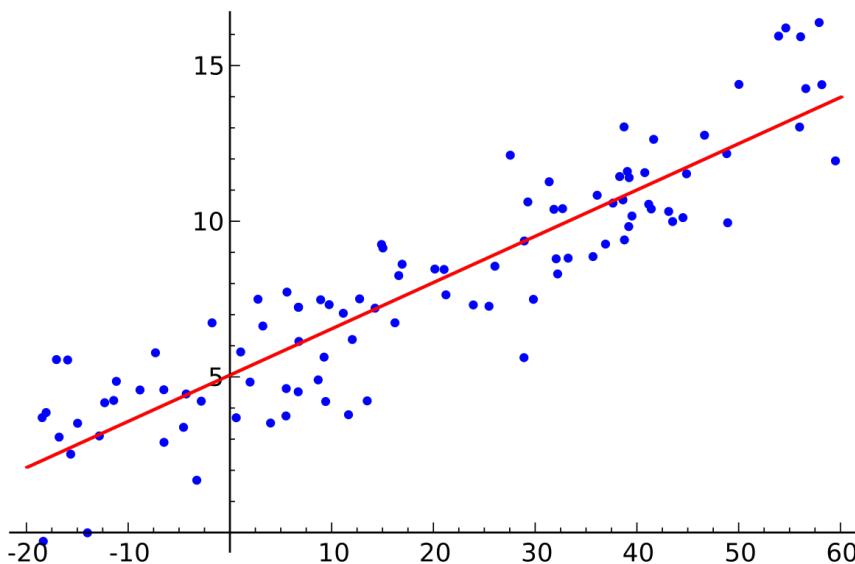


Figura 2.9: Esempio di regressione lineare, con variabile di input sulle ascisse e variabile target sulle ordinate. [9]

2.2 Applicazioni del ML in ambito clinico

Gli strumenti e gli algoritmi fino ad ora presentati trovano applicazione in numerosissimi ambiti di studio: quello clinico è uno di questi. Sempre più frequentemente il ML si ritrova ad offrire un supporto chiave all'interno di tools clinici di diagnosi e monitoraggio proprio per l'enorme quantità di informazioni e features che riescono ad essere valutati e tenuti in considerazione contemporaneamente.

Le applicazioni sono sempre in aumento e variano dall'utilizzo di modelli per l'identificazione di soggetti più a rischio per determinate patologie, all'analisi assistita di immagini mediche come radiografie e scansioni CT. Inoltre con l'utilizzo di sensori indossabili le applicazioni si moltiplicano, come ad esempio nel caso del rilevamento di apnee notturne [12].

2.2.1 Background clinico

Dopo aver accennato l'importanza di strumenti di apprendimento automatico e citato definizioni e concetti di ML necessari per comprendere l'analisi dei dati effettuata, è importante ovviamente fornire un contesto di carattere clinico per poter capire anche la condizione dei soggetti da cui sono stati raccolti i dati e gli attuali mezzi per il monitoraggio.

Paralisi Cerebrale Infantile a tipo emiplegia

I soggetti dello studio sono bambini con Paralisi Cerebrale Infantile (PCI). Questo disturbo del neurosviluppo colpisce approssimativamente 2-3 su 1000 nascite e 40-100 su 1000 nascite pretermine e prematuri, e si manifesta attraverso disfunzioni nei movimenti e a livello posturale, finendo per causare limitazioni nell'attività motoria quotidiana [13].

E' importante notare che esistono differenti forme di PCI, e nello specifico i soggetti del nostro studio presentano PCI a tipo emiplegia, la quale colpisce un solo emisfero cerebrale risultando in una parziale immobilità di un lato del corpo. Ciò crea un'asimmetria tra le abilità motorie dell'arto dominante rispetto all'arto non dominante che può essere più o meno elevata a seconda del soggetto [14].

Per questo motivo è importante osservare e monitorare i soggetti di modo da poter capire approfonditamente le loro capacità motorie e se ci sono variazioni nel tempo in modo tale da poter modificare e/o personalizzare i percorsi terapeutici

Attuali parametri di valutazione

L'attuale mezzo di valutazione attraverso il quale viene stimato il livello di abilità motoria del soggetto fa riferimento al test clinico Assisting Hand Assessment (AHA), cioè un test semi-strutturato della durata di 20 minuti nel quale al soggetto è richiesto di eseguire differenti attività (in base all'età) che servono a valutare la bimanualità [15]. Questo punteggio varia da 0 (arto affetto non utilizzato), ad un punteggio massimo di 100 (nessuna asimmetria rilevata).

Oltre al punteggio ottenuto dalla valutazione tramite AHA spesso si fa riferimento anche a un punteggio ottenuto da un sistema di classificazione chiamato Manual Ability Classification System (MACS), il quale viene ricavato attraverso l'utilizzo di un questionario compilato dai genitori in base al comportamento quotidiano dei propri figli. L'indice MACS varia dal livello I, in cui il bambino ha un'abilità manuale piena, al livello V, in cui il bambino ha una ridotta capacità di utilizzare le mani e le braccia [16].

Il livello MACS tuttavia può variare a seconda dell'interpretazione dei comportamenti da parte del genitore, il quale non è spesso in grado di fornire una valutazione oggettiva e conforme a quella degli altri.

Uso di attigrafi su soggetti con PCI a tipo emiplegia

La necessità di dipendere il meno possibile dalle valutazioni soggettive di genitori e clinici, al fine di misurare le prestazioni dei soggetti una volta fuori dall'ambiente clinico, ha portato ad utilizzare sensori indossabili per quantificare l'attività motoria di diversi distretti corporei, come ad esempio quella degli arti superiori.

Nello specifico, è stata dimostrata l'efficacia di sensori indossabili non invasivi chiamati attigrafi adatti a poter essere utilizzati anche con i soggetti di più giovane età: si tratta di devices contenenti sensori accelerometrici capaci di registrare il movimento degli arti superiori se indossati ai polsi. [13].

3. DATASET E PREPROCESSING

In questo capitolo si affrontano i dati reali sulla base dei quali è stato costruito il nostro lavoro, cercando di comprenderne al meglio la natura e la forma, fino a parlare della loro elaborazione prima di essere processati dagli algoritmi di Machine Learning.

Nello specifico, i dati di questo progetto sono stati registrati da attigrafi (modello ActiGraph GT3X [10]) applicati ad entrambi gli arti superiori dei soggetti.



Figura 3.1: Foto del modello dell’attigrafo utilizzato per il rilevamento dei dati [10].

3.1 Dati di partenza

Per questo studio hanno partecipato su base volontaria 60 soggetti di età compresa tra 5 e 27 anni (media=11.1), di cui 26 rappresentano il gruppo di controllo in quanto soggetti con sviluppo tipico. Tutti i soggetti sono stati valutati da personale clinico con valutazione AHA indossando gli attigrafi in ambiente strutturato e poi li hanno tenuti per una settimana durante la loro vita quotidiana.

Alla fine del processo, per ogni soggetto si ha a disposizione oltre alle loro generalità anche il punteggio AHA, il livello MACS ed un terzo indicatore: l’Asymmetry Index (AI).

Questo indicatore rappresenta l'indice di asimmetria tra mano dominante e non dominante di un soggetto e può assumere valori discreti tra -100 (totale asimmetria degli arti verso la mano non dominante) e 100 (totale asimmetria degli arti verso la mano dominante). L'AI è l'unico indicatore empirico di cui disponiamo in partenza, infatti è calcolabile a partire dai dati tramite questa formula:

$$AI = \frac{AC_D - AC_{ND}}{AC_D + AC_{ND}} \cdot 100 \quad (3.1)$$

dove AC_D e AC_{ND} sono il risultato dell'elaborazione della media delle magnitudo a partire dagli spostamenti sui tre assi registrati dagli attigrafi. Questo dataset di intensità lungo gli assi è riferito come “Activity Count” (AC), ed è a sua volta ricavato da un'elaborazione dell'output “raw” degli attigrafi, che tramite il software ActiLife [17] registrano le accelerazioni degli arti superiori a una certa frequenza.

Come già detto, il dataset è diviso in due parti: una parte relativa al test clinico AHA (chiamata sessione AHA), l'altra invece riguarda le misurazioni durante sei giorni di vita quotidiana dei soggetti (chiamata sessione WEEK). È importante sottolineare che questi dati differiscono per la loro durata, ma sono entrambi gruppi di serie temporali multivariate: ad ogni istante sono legate le intensità degli spostamenti lungo i tre assi di movimento di ogni mano. Ogni dato è dunque rappresentabile tramite una serie temporale a due variabili (una per ogni mano), calcolando il modulo punto a punto attraverso le tre coordinate registrate.

Nelle sezioni seguenti si analizzano i due differenti tipi di sessione.

3.1.1 Sessioni AHA

I dati pertinenti al test AHA sono di lunghezza variabile in base alla presenza della condizione neurologica e vanno da un minimo di 11 a un massimo di 27 minuti (18 minuti in media). Il test clinico in laboratorio varia anche in base all'età del soggetto, ma in ogni caso è ideato in modo tale da poter stimolare la bimanualità in un ambiente

semi-strutturato. Spesso il test consiste in un gioco da svolgere da soli o in gruppo o di varie prove che coinvolgono entrambi gli arti.

La bimanualità è forzatamente incentivata, quindi i dati prodotti in laboratorio risultano più significativi per quanto riguarda l’analisi dell’asimmetria tra mano dominante e non dominante.

3.1.2 Sessioni WEEK

I dati registrati al di fuori dell’ambiente clinico sono di lunghezza fissa, della durata di una settimana. Ai soggetti non sono state date deliberatamente istruzioni chiare sul come comportarsi durante questo periodo proprio per non creare un bias che comprometterebbe in principio l’onere di indossare dei sensori per così tanto tempo, cercando di stimolare un tipo di movimento che naturalmente non sarebbe spontaneo. Per questo motivo i dati a casa sono interessanti per poter comprendere la “vera” attività motoria dell’individuo.

La conseguenza immediata di registrare dati in ambienti non strutturati e liberi è naturalmente la presenza di elevato rumore all’interno delle misurazioni, che rende meno significativa e più difficoltosa l’analisi dell’asimmetria tra i due arti, specialmente tra le molte attività quotidiane che non comprendono l’utilizzo di entrambi gli arti.

Un’ulteriore conseguenza della semi-indipendenza dei soggetti durante la settimana è la decisione di indossare o meno i sensori durante la notte, un aspetto che si è rivelato importante durante lo sviluppo del nostro studio.

3.2 Estrazione delle Features

In accordo a quanto già anticipato, partendo dagli AC abbiamo come prima cosa trasformato i dati in un formato rappresentabile: per ogni punto della serie si è calcolato la magnitudo del vettore spostamento ($|\vec{v}| = \sqrt{x^2 + y^2 + z^2}$), ottenendo così due serie rispettivamente per la mano dominante e non, rappresentabili in un grafo a due variabili.

Per il processo di estrazione delle features si è scelto di non utilizzare l'approccio tradizionale che si concentra generalmente sulla teoria della statistica utilizzando medie, deviazioni standard e varianze, ma piuttosto di utilizzare algoritmi progettati specificatamente per lavorare con serie temporali (opportunamente preprocessate), per mantenere centrale la consapevolezza dell'ambiente in cui si sta lavorando.

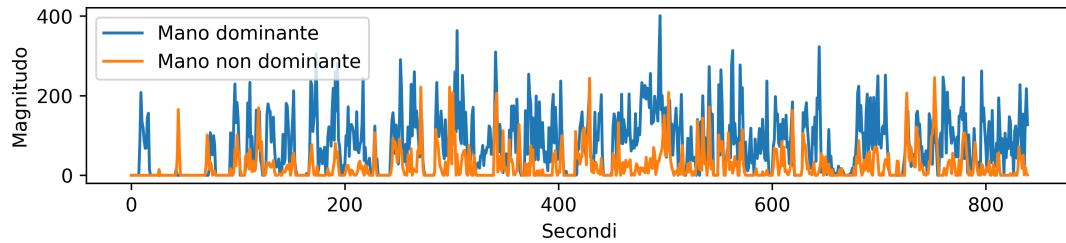


Figura 3.2: Grafico delle magnitudo dello spostamento delle mani di un soggetto con PCI durante una sessione AHA.

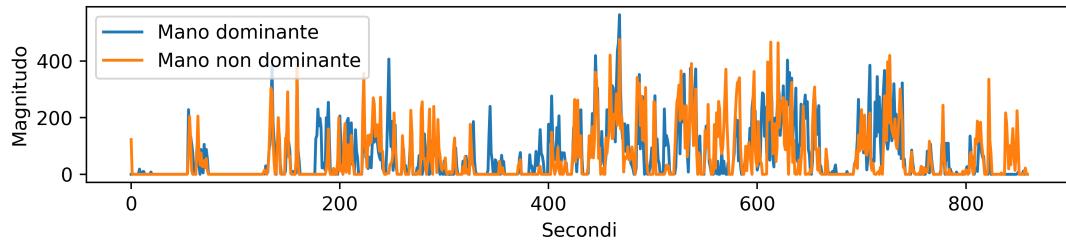


Figura 3.3: Grafico delle magnitudo dello spostamento delle mani di un soggetto del gruppo di controllo durante una sessione AHA.

3.2.1 Suddivisione in intervalli temporali

Durante il confronto tra diverse serie temporali, è stato necessario uniformare la lunghezza delle sessioni, considerato che alcune di queste sono differenti nella durata.

Abbiamo quindi implementato un processo che permette di suddividere le serie in intervalli di lunghezza arbitraria. Nel nostro lavoro abbiamo testato finestre da 5, 10 e 15 minuti. La procedura di sampling effettua i seguenti controlli:

- Se una serie non è abbastanza lunga, si concatenano punti appartenenti all'inizio della serie appendendoli fino a raggiungere la lunghezza di un sample.
- Se la durata di una serie supera la lunghezza di un sample (e dunque è suddivisibile in più di un sample) ma non è multipla di questo, si rimuove l'eccesso equamente dall'inizio e dalla fine della serie, essendo questi momenti meno significativi.

Questi accorgimenti ci permettono di suddividere tutte le serie temporali in intervalli di ugual misura. Tuttavia, ciò comporta una perdita di dati: i sampling da 300, 600 e 900 secondi perdono rispettivamente l'8.5%, il 23.7% e il 26.9% dei secondi totali per ogni serie.

3.2.2 Manipolazione delle serie temporali

A questo punto ogni serie è rappresentata da un insieme di sample, e dunque ogni sessione è composta da due di questi insiemi: uno per l'arto dominante e uno per quello non dominante. Al fine di evitare di lavorare con serie temporali multivariate, si è deciso di rappresentare ogni sessione con una sola serie temporale, risultato di una composizione dei due arti superiori.

Di seguito sono descritte le tre tecniche da noi utilizzate per il processo delle due serie.

Concatenazione

Questa tecnica prevede che le due serie vengano concatenate una all'altra, nello specifico quella non dominante segue quella dominante. Si ottiene così una serie temporale lunga il doppio.

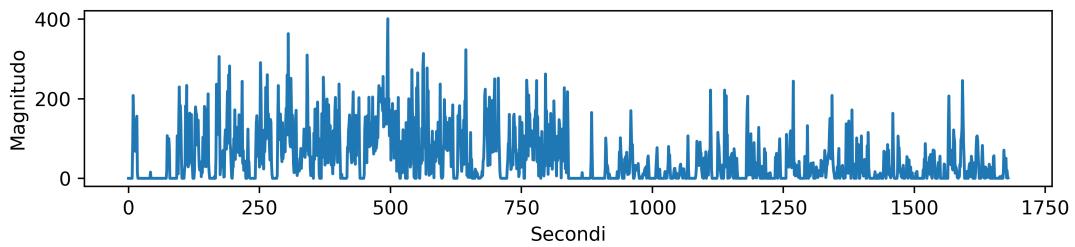


Figura 3.4: Concatenazione delle serie temporali mostrate nella figura 3.2.

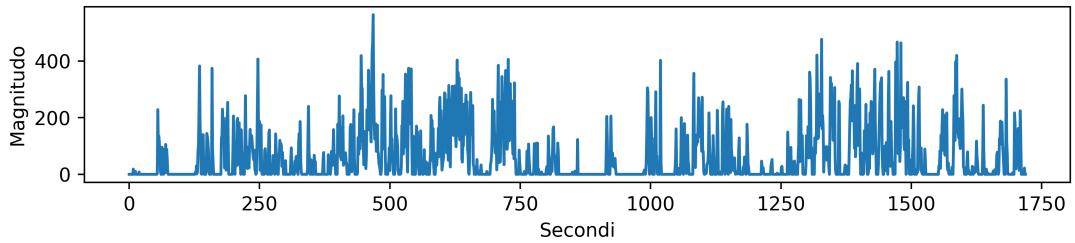


Figura 3.5: Concatenazione delle serie temporali mostrate nella figura 3.3.

Differenza

Questa tecnica prevede che la serie temporale della mano non dominante venga sottratta a quella dominante punto per punto. Si ottiene così una serie temporale risultato della differenza tra le magnitudo delle due mani, che può quindi assumere anche valori negativi.

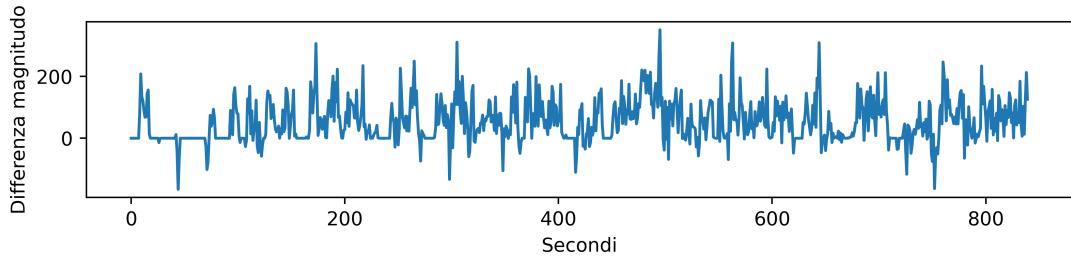


Figura 3.6: Differenza delle serie temporali mostrate nella figura 3.2.

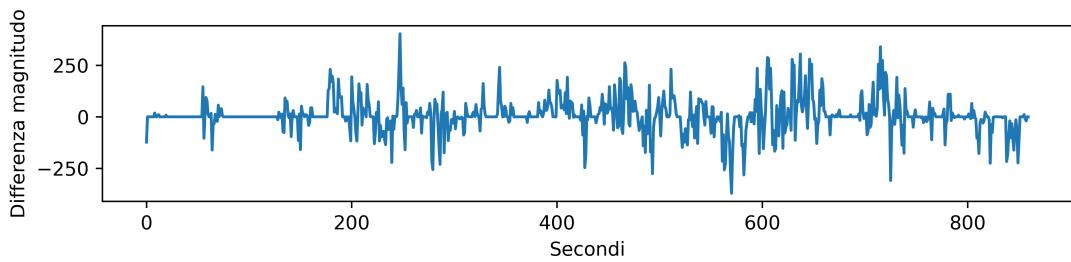


Figura 3.7: Differenza delle serie temporali mostrate nella figura 3.3.

AI punto-a-punto

Questa tecnica prevede che venga applicata la formula (3.1) per ogni punto delle due serie. Si ottiene quindi una sorta di versione “normalizzata” della differenza, che non tiene conto delle singole intensità ma oscilla tra -100 e 100.

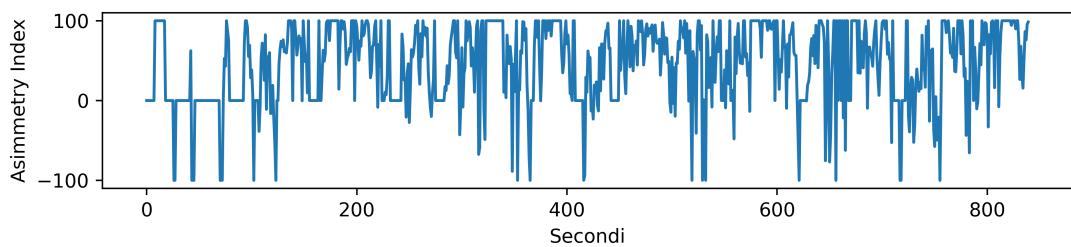


Figura 3.8: Asimmetry Index delle serie temporali mostrate nella figura 3.2.

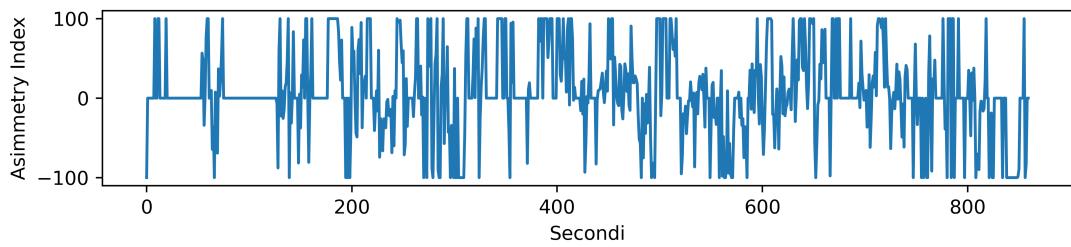


Figura 3.9: Asimmetry Index delle serie temporali mostrate nella figura 3.3.

4. ANALISI

A questo punto, con i nostri dati opportunamente preprocessati, possiamo affrontare correttamente tutti gli step della pipeline nella prossima sezione.

4.1 Pipeline di lavoro

La pipeline di lavoro che abbiamo costruito è la seguente:

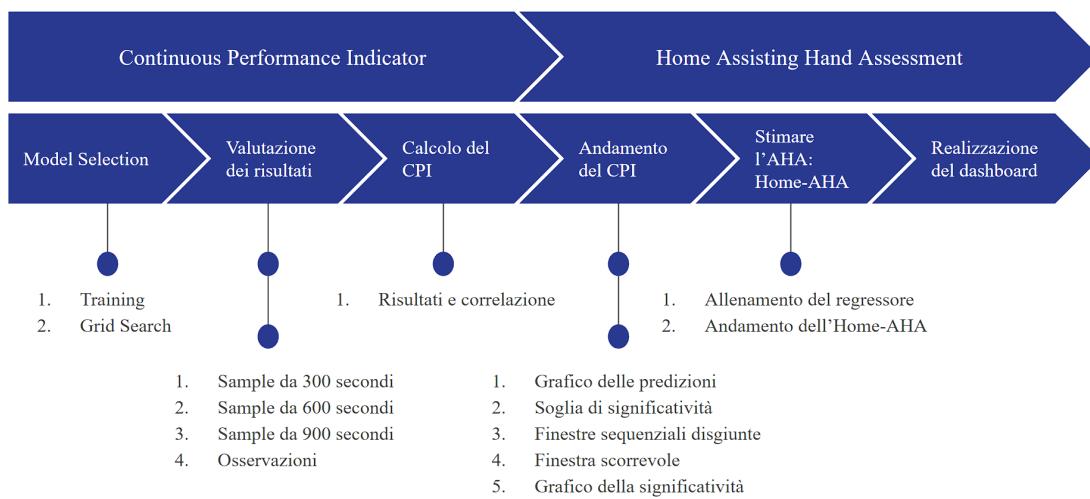


Figura 4.1: Rappresentazione grafica della pipeline di lavoro.

4.2 Continous Performance Indicator (CPI)

Questa prima parte della pipeline si occupa della costruzione dei modelli di ML e della presentazione del Continous Performance Indicator (CPI), una feature estratta tramite l'applicazione di un classificatore binario su sessioni WEEK.

La tecnica utilizzata per il calcolo del CPI consiste nell'allenare classificatori binari che identifichino l'appartenenza o meno di un soggetto al gruppo di controllo tramite

l’analisi della rispettiva sessione AHA e successivamente nel calcolare la percentuale di sample predetti come istanze positive sulle sessioni WEEK.

Come nota, è giusto riportare che inizialmente il nostro studio ha anche esplorato una possibile classificazione multiclass utilizzando algoritmi unsupervised, ma sia gli scarsi risultati (fig. 4.2), sia un riscontro essenzialmente lineare dell’elbow plot (fig. 4.3) ci ha instradato ulteriormente verso le procedure sopracitate.

Nella figura 4.2 si può vedere come KMeans clusterizzi i sample delle sessioni AHA in relazione al punteggio AHA dei relativi soggetti. KMeans sembra in grado di distinguere il gruppo emiplegico (cluster 0) da quello di controllo (cluster 2), ma non riesce ad andare oltre a questa suddivisione binaria, perdendosi in sottoinsiemi semivuoti che assecondano il rumore nelle serie, senza nessuna correlazione con l’AHA.

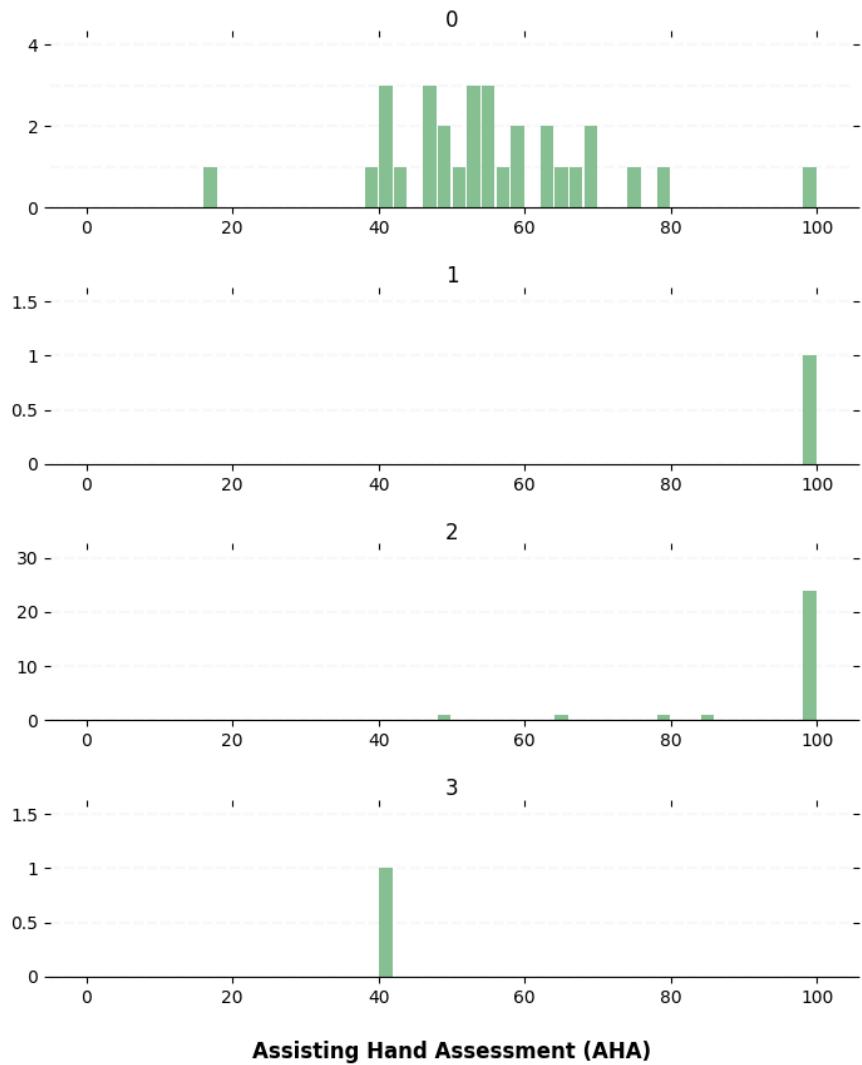


Figura 4.2: Tentativo di suddivisione multi-cluster utilizzando KMeans su sample da sessioni AHA da 900 secondi.

Per assicurarsi che non si trattasse di un caso isolato si è deciso di integrare un’analisi dell’elbow plot fino a 60 cluster, ma come si può vedere dalla figura sottostante il risultato non presenta nessun gomito, dunque non si raggiunge mai un decremento dell’inerzia più significativo del precedente con l’aumento della complessità.

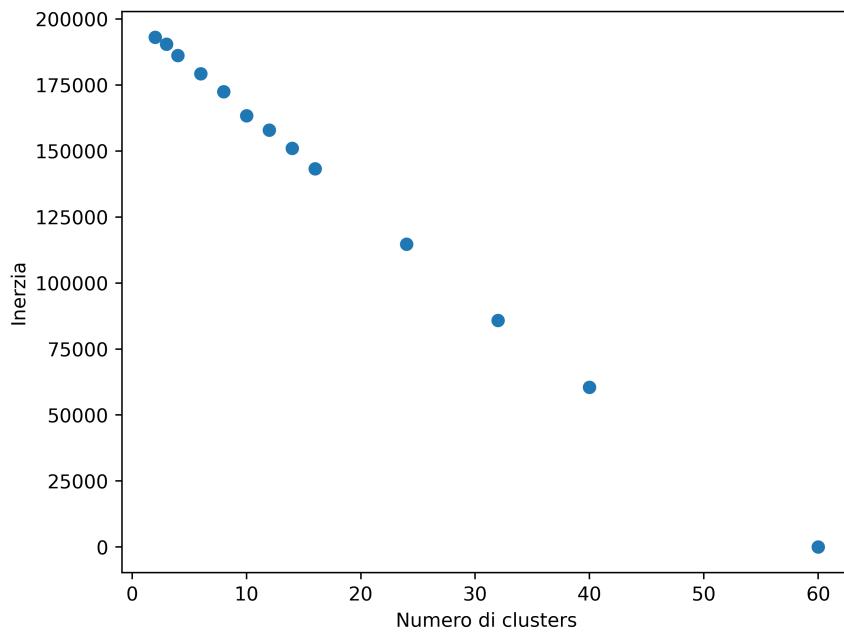


Figura 4.3: Elbow plot simil-lineare ottenuto da esecuzioni KMeans.

4.2.1 Model Selection

Per la fase di Model Selection abbiamo implementato una serie di Grid Search che identifichino i modelli più adatti a risolvere il problema di classificazione. L'output di queste molteplici Grid Search sarà quindi una classifica dei migliori modelli allenati e valutati sulle sessioni AHA.

Training

Come primo passo si sono scelti quattro algoritmi di Machine Learning, la prima metà composta da BOSSEnsemble e ShapeDTW per l'apprendimento supervisionato (2.1.7), la seconda metà composta da KMeans e KMedoids per l'apprendimento non supervisionato (2.1.6). L'implementazione utilizzata di questi algoritmi è quella di sktime, un framework in linguaggio Python che opera da wrapper per differenti librerie ML adatte alla manipolazione di time series (e.g. scikit-learn, keras, tensorflow ...)

Dopodiché si sono ottenute nove versioni di ogni algoritmo tramite le permutazioni di due gruppi di preprocessing: il primo gruppo contiene i tre metodi per la manipolazione delle time series analizzati nella sezione 3.2.2; il secondo gruppo contiene tre dimensioni scelte per la suddivisione in intervalli delle serie: 300, 600 e 900 secondi.

Su ognuna delle nove versione è stata lanciata una Grid Search allo scopo di individuare gli iperparametri migliori per ogni specifico modello.

Gli iperparametri specifici testati dalla Grid Search per ogni algoritmo sono i seguenti. I campi non specificati sono quelli di default, consultabili nella sezione API del sito sktime [18]

- KMeans:
 - n-clusters: 2
 - init-algorithm: kmeans++, forgy
 - metric: euclidean, dtw
 - averaging-method: mean
- KMedoids:
 - n-clusters: 2
 - init-algorithm: random, forgy
 - metric: euclidean, dtw
- BOSSEnsemble:
 - feature-selection: chi2, none
- ShapeDTW:
 - shape-descriptor-function: raw, paa

Nello specifico:

KMeans. Sarà un clusterer binario che può utilizzare come misura di distanza il DTW e la distanza Euclidea. L'algoritmo di inizializzazione per le prime fasi di operazione può essere forgy, dove k-centroidi vengono presi casualmente dal dataset, o kmeans++.

Quest'ultimo prende il primo centroide randomicamente dal dataset, e poi raccoglie le distanze di tutte le altre serie da quest'ultimo. Il centroide successivo viene selezionato in modo che la probabilità di essere scelto sia proporzionale alla distanza con l'ultimo centroide scelto, dunque più grande la distanza, più è probabile che la serie venga scelta come nuovo punto.

KMedoids. Sarà anch'esso un clusterer binario che può utilizzare come misura di distanza il DTW e la distanza Euclidea. L'algoritmo di inizializzazione per le prime fasi di operazione può essere sia forgy, sia “random”, che divide le serie in due gruppi in maniera totalmente casuale, e poi ne calcola i centroidi tramite la mediana di ogni cluster.

BOSSEnsemble. Sarà un classificatore binario con una strategia di feature selection extra: chi2. Questa ottimizza l'uso di memoria e velocizza il processo di trasformazione della serie in gruppi di parole.

ShapeDTW. Sarà un classificatore binario con due differenti descrittori di forma: raw, che conserva le sottosequenza nella loro forma, senza trasformazioni, e il Piecewise Aggregate Approximation (paa), che suddivide la serie in sottosequenze di cui calcola successivamente la media. I valori di queste medie sono incorporati in un vettore che fungerà da descrittore di forma per una serie.

Grid Search

L'implementazione utilizzata della Grid Search appartiene al pacchetto Python di ML scikit-learn. [19]

```
1 gs = GridSearchCV(
```

```

2   # Modello ML
3
4   model,
5
6   # Griglia parametri da testare
7   param_grid,
8
9   # 5fold cross validation ordinata casualmente
10  cv = StratifiedKFold(n_splits = 5, shuffle = True),
11
12  # Numero di thread da utilizzare
13  n_jobs = 5,
14
15  # Funzione di scoring per la validazione
16  scoring = scorer
17
18 )
19
20 # Lancio della grid search
21 gs.fit(X, y)

```

Code 4.1: Inizializzazione della grid search utilizzata per la model selection

Al suo interno sono integrati una Cross Validation e uno scorer, sempre provenienti dallo stesso sklearn. Di seguito si descrivono questi ultimi:

Cross Validation. Ogni Grid Search implementa un processo di Cross Validation, più in particolare quella utilizzata è una Stratified-K-Fold-Cross-Validation con K=5 (S5FCV).

Questo particolare tipo di Cross Validation è progettato per lavorare con set di dati sbilanciati, ciò significa che se il dataset presenta una classe prevalente rispetto alle altre, la S5FCV assicura che ogni fold contenga la stessa proporzione di esempi di quell'insieme. Nel nostro caso specifico questo è necessario poiché avremo molti meno dati relativi al gruppo di controllo rispetto a quelli del gruppo emiplegico a causa della durata del test AHA ridotta, ed è quindi opportuno allenare e validare il modello equamente su esempi di tutte le classi.

Scorer. La metrica di valutazione utilizzata nella Grid Search e più nello specifico nel S5FCV è una variazione del punteggio F1-score da noi implementata per unificare

il flusso di operazioni per entrambi i modelli supervised e unsupervised. Infatti, dato che per il resto dello studio è importante conoscere quale sia il valore predetto (0 o 1) associato al gruppo di controllo e considerato che un modello unsupervised non conserva una consistenza nell’assegnamento di questi valori, è necessario che il nostro punteggio rilevi automaticamente l’ordine degli insiemi, così da poterne tenere conto per gli step di analisi successivi.¹

Il funzionamento del nostro scorer è semplice: l’F1-score viene calcolato normalmente e poi nuovamente invertendo i valori delle predizioni. A questo punto si considera soltanto il punteggio più alto.

```

1 def scorer(model, X, Y):
2
3     # Ricavo le predizioni del modello
4     y_pred = model.predict(X)
5
6     # Controllo se il modello è un classificatore
7     if issubclass(type(model), BaseClassifier):
8
9         # In questo caso so che il gruppo di controllo ha valore
10        1
11
12        return f1_score(Y, y_pred, average='weighted')
13
14    else:
15
16        # Nel caso di un clusterer, inverto le predizioni
17        inverted_y_pred = [1 if item == 0 else 0 for item in
18        y_pred]
19
20        # Ritorno lo score che massimizza su uno dei due set
21        return max(f1_score(Y, y_pred, average='weighted'),
22        f1_score(Y, inverted_y_pred, average='weighted'))
```

¹Ad esempio può capitare per i modelli unsupervised che si assegni al gruppo di controllo il valore 1 e in un’altra esecuzione il valore 0, in maniera non deterministica.

Code 4.2: Funzione per valutare le performance di un modello

Come si può notare si sta utilizzando la versione “weighted” dell’F1-score; naturalmente anche questo accorgimento è atto a prevenire assunzioni basate su uno sbilanciamento dei dati. La versione weighted dell’F1-score, infatti, tiene conto dello sbilanciamento tra le classi e restituisce un punteggio pesato secondo la frequenza relativa di ogni classe.

4.2.2 Valutazione dei risultati

Il ciclo di Training restituisce quindi una classifica dei migliori modelli che hanno performato sulle sessioni AHA.

Nelle prossime sezioni analizzeremo in dettaglio ogni output per ogni dimensione di sample, dunque ci saranno 12 modelli per tabella, ordinati per Mean Test Score (MTS) migliore. Questo stesso MTS è il risultato della validazione calcolato dal S5FCV, con relativa deviazione standard Std Test Score (STD).

Sample da 300 secondi

Il sampling da 300 secondi che si vede nella tabella 4.1 ci mostra che i metodi di preprocessing AI e difference producono score migliori rispetto alla concatenazione, che tranne per un caso rimane sotto la soglia del 0.8 MTS. Il caso in cui quest’ultimo metodo sembra funzionare meglio è quello del BOSSEnsemble, legato probabilmente al fatto che questo classificatore fa uso di diagrammi di serie temporali codificate in parole.²

I modelli unsupervised in questo caso prediligono l’utilizzo del DTW e del forgry. Inoltre, in nessun caso per KMedoids è stato scelto l’algoritmo di inizializzazione random, e la sua miglior performance fa uso della distanza Euclidea.

²Essendo la serie concatenata, la sua dimensione è doppia: è evidente che più parole portano a una maggiore precisione nella classificazione.

I modelli supervised sono abbastanza bilanciati tra loro, e si può notare che BOSSEnsemble predilige esclusivamente l'uso dell'esecuzione di default senza alcuna feature selection, mentre ShapeDTW lavora meglio con il descrittore di forma PAA. A differenza dei modelli unsupervised, per i quali non si riesce a definire una scelta definitiva dei parametri, per quelli supervised si evince una prevalenza netta di questi due parametri manifestati rispetto alle corrispettive alternative.

A discapito di una cifra decimale, a parità di performance i due modelli migliori risultano nell'ordine KMedoids e KMeans, con 0.95 MTS e 0.02 STD ciascuno. Entrambi i modelli utilizzano il metodo AI, ma differiscono per misura di distanza e algoritmo di inizializzazione.

Metodo	Modello	Parametri	MTS	STD
ai	TimeSeriesKMedoids	forgy, euclidean, 2	0.955	0.024
ai	TimeSeriesKMeans	mean, kmeans++, dtw, 2	0.95	0.023
concat	BOSSEnsemble	none	0.943	0.053
difference	TimeSeriesKMeans	mean, kmeans++, dtw, 2	0.94	0.043
difference	ShapeDTW	paa	0.939	0.047
ai	BOSSEnsemble	none	0.938	0.043
difference	TimeSeriesKMedoids	forgy, dtw, 2	0.933	0.037
ai	ShapeDTW	paa	0.933	0.026
difference	BOSSEnsemble	none	0.928	0.045
concat	ShapeDTW	paa	0.809	0.045
concat	TimeSeriesKMeans	mean, forgy, euclidean, 2	0.732	0.184
concat	TimeSeriesKMedoids	forgy, dtw, 2	0.67	0.096

Tabella 4.1: Classifica dei migliori modelli con sample da 300 secondi.

Sample da 600 secondi

Successivamente, il sampling da 600 secondi della tabella 4.2 conferma la prevalenza del metodo AI su difference e soprattutto sulla concatenazione, che performa decisamente meglio.

mente peggio su intervalli da dieci minuti, e il cui unico risultato significativo è legato all'uso di BOSSEnsemble (0.94 MTS) con feature selection chi2, contrariamente a quanto visto per gli intervalli da cinque minuti.

Ancora una volta i modelli unsupervised sembrano i migliori, raggiungendo anche il punteggio 1 MTS con 0 STD per il modello KMeans con metodo AI. Questo modello in particolare fa uso di distanza Euclidea e kmeans++, parametri che sembrano decisivi per il KMeans su sample da dieci minuti. Per quanto riguarda KMedoids, forgy sembra riconfermarsi migliore dell'algoritmo di inizializzazione randomico.

Per i modelli supervised si riconferma quanto detto per il sampling da cinque minuti, con eccezione dell'uso di un descrittore di forma raw per lo ShapeDTW con metodo difference.

In generale, le prestazioni risultano migliori del sampling a cinque minuti, con più della metà dei modelli che supera il 0.95 MTS. Questo è comprensibile, considerando che in ogni sample è contenuto il doppio delle informazioni.

Metodo	Modello	Parametri	MTS	STD
ai	TimeSeriesKMeans	mean, kmeans++, euclidean, 2	1.0	0.0
ai	TimeSeriesKMedoids	forgy, euclidean, 2	0.988	0.024
ai	BOSSEnsemble	none	0.976	0.029
difference	TimeSeriesKMeans	mean, kmeans++, euclidean, 2	0.976	0.03
difference	TimeSeriesKMedoids	forgy, dtw, 2	0.976	0.03
ai	ShapeDTW	paa	0.952	0.044
difference	ShapeDTW	raw	0.951	0.045
difference	BOSSEnsemble	none	0.947	0.08
concat	BOSSEnsemble	chi2	0.947	0.08
concat	TimeSeriesKMeans	mean, kmeans++, euclidean, 2	0.896	0.208
concat	TimeSeriesKMedoids	forgy, dtw, 2	0.809	0.122
concat	ShapeDTW	paa	0.736	0.118

Tabella 4.2: Classifica dei migliori modelli con sample da 600 secondi.

Sample da 900 secondi

Infine, per quanto riguarda il sampling in 15 minuti della tabella 4.3, il metodo di preprocessing difference sembra acquistare significatività, arrivando essenzialmente ad equiparare il metodo AI. Di nuovo invece, la concatenazione si riconferma definitivamente il metodo meno efficace.

I modelli unsupervised presentano ottime performance: KMeans riesce ad arrivare al punteggio di 1.0 MTS con 0.0 STD, sia con le metriche Euclidean che di DTW, sempre con kmeans++ come algoritmo di inizializzazione. KMedoids arriva a un punteggio di 0.98 MTS sia con forgy e DTW che con random e distanza Euclidean.

Per i modelli supervised, ShapeDTW con descrittore di forma PAA arriva a 0.98 MTS, risultato che BOSSEnsemble non riesce a raggiungere nemmeno con feature selection chi2.

Metodo	Modello	Parametri	MTS	STD
difference	TimeSeriesKMeans	mean, kmeans++, dtw, 2	1.0	0.0
ai	TimeSeriesKMeans	mean, kmeans++, euclidean, 2	1.0	0.0
difference	TimeSeriesKMedoids	forgy, dtw, 2	0.983	0.033
difference	ShapeDTW	paa	0.983	0.034
ai	TimeSeriesKMedoids	random, euclidean, 2	0.983	0.034
ai	ShapeDTW	paa	0.983	0.034
ai	BOSSEnsemble	none	0.95	0.041
concat	BOSSEnsemble	none	0.948	0.07
difference	BOSSEnsemble	chi2	0.934	0.062
concat	TimeSeriesKMedoids	forgy, dtw, 2	0.77	0.136
concat	TimeSeriesKMeans	mean, kmeans++, euclidean, 2	0.755	0.267
concat	ShapeDTW	paa	0.69	0.089

Tabella 4.3: Classifica dei migliori modelli con sample da 900 secondi.

Osservazioni

Si osserva che l’ambiente adatto alle migliori performance sembra essere quello costruito dai modelli unsupervised, in particolare KMeans con metrica distanza Euclidea e algoritmo di inizializzazione kmeans++ previo preprocessing AI. I modelli supervised presentano quasi sempre prestazioni leggermente inferiori, ma comunque largamente soddisfacenti, superando il più delle volte la soglia dello 0.9 MTS.

Si osserva però che un sampling da 300 secondi, oltre a fornire una quantità di informazioni sufficiente per una corretta classificazione, permette un incremento importante del numero di sample senza apportare una degradazione significativa delle performance.³

Viene scelto quindi il modello che performa meglio su questo intervallo temporale, cioè TimeSeriesKMedoids con preprocessing AI, con algoritmo di inizializzazione forgy e metrica distanza Euclidea. Questo modello è stato scelto sia per le ottime prestazioni ($MTS > 0.95$, $STD < 0.025$), sia per la quantità minima di informazioni immagazzinate in un sample.

4.2.3 Calcolo del CPI

Una volta scelto il modello, si possono utilizzare le sue previsioni per il calcolo del CPI settimanale, e successivamente analizzare la sua efficacia come feature. Come già anticipato, questo indice consiste essenzialmente nella percentuale di sample predetti appartenenti al gruppo di controllo. Un CPI settimanale è dunque calcolabile tenendo in considerazione tutti i sample contenuti in un’intera sessione WEEK. O meglio, tutti i sample considerati “validi”.

Infatti, un sample per essere preso in considerazione nel calcolo del CPI e quindi considerato valido deve contenere almeno un movimento significativo. Da ciò segue che sono scartati tutti quei sample con tutte le intensità nulle, principalmente legate

³Con intervalli da 300 secondi avremo il doppio dei sample rispetto alla versione da 600 e il triplo rispetto a quella da 900.

a momenti in cui il soggetto sta dormendo. Questi sample scartati non sono stati considerati di alcuna significatività per il nostro studio, poiché è più interessante il caso in cui un soggetto è in movimento piuttosto del caso in cui il soggetto è completamente immobile.

Una volta ottenute le predizioni per tutti i sample validi della sessione, si calcola per ogni paziente la percentuale di valori associati al gruppo di controllo, cioè il CPI settimanale. Questa procedura è consistente anche per i modelli unsupervised (oltre per gli accorgimenti sopracitati) grazie a un controllo sul fitting del modello finale nella fase di training: si osserva per i soli modelli unsupervised se l'F1-score finale è maggiore sul set di predizioni invertito. Se fosse così, significherebbe che il valore associato al gruppo di controllo è 0. In caso contrario, sarebbe 1. In questo modo siamo in grado di individuare automaticamente il valore associato al gruppo di controllo, e tenerne conto per le successive fasi di analisi.

Risultati e correlazione

Applicando la procedura descritta ai dati delle sessioni WEEK presenti all'interno del dataset di partenza otterremo quindi 60 CPI settimanali, che possono essere messi in correlazione con il punteggio AHA assegnato dai clinici per comprendere l'efficacia della feature estratta.

Si presentano dunque nelle successive tabelle e con l'apposita rappresentazione grafica i valori di CPI confrontati con l'AHA.

Soggetto	MACS	AHA	CPI
1	2	46	58.094
2	2	50	78.424
3	2	57	70.342
4	2	46	68.346
5	3	53	69.964
6	2	47	60.426
7	1	78	89.927
8	2	54	69.706
9	3	41	54.321
10	2	58	66.321
11	2	49	63.182
12	1	65	92.208
13	3	40	72.635
14	1	63	89.005
15	2	49	53.801
16	3	41	78.706
17	1	67	89.289
18	2	52	74.356
19	3	16	58.855
20	2	49	53.217
21	3	39	54.961
22	2	53	60.660
23	1	64	88.815
24	2	59	68.616
25	3	42	69.581
26	1	68	51.111
27	1	54	54.843
28	2	40	81.298
29	1	84	89.746
30	1	78	59.668

Soggetto	MACS	AHA	CPI
31	2	55	89.023
32	1	75	78.907
33	1	69	77.341
34	1	63	83.842
35	0	100	97.299
36	0	100	94.488
37	0	100	92.528
38	0	100	95.616
39	0	100	97.176
40	0	100	96.619
41	0	100	96.660
42	0	100	98.727
43	0	100	97.227
44	0	100	97.699
45	0	100	97.084
46	0	100	96.237
47	0	100	98.439
48	0	100	92.118
49	0	100	90.720
50	0	100	98.518
51	0	100	97.599
52	0	100	97.549
53	0	100	95.261
54	0	100	94.553
55	0	100	93.488
56	0	100	95.806
57	0	100	91.118
58	0	100	89.805
59	0	100	97.375
60	0	100	98.853

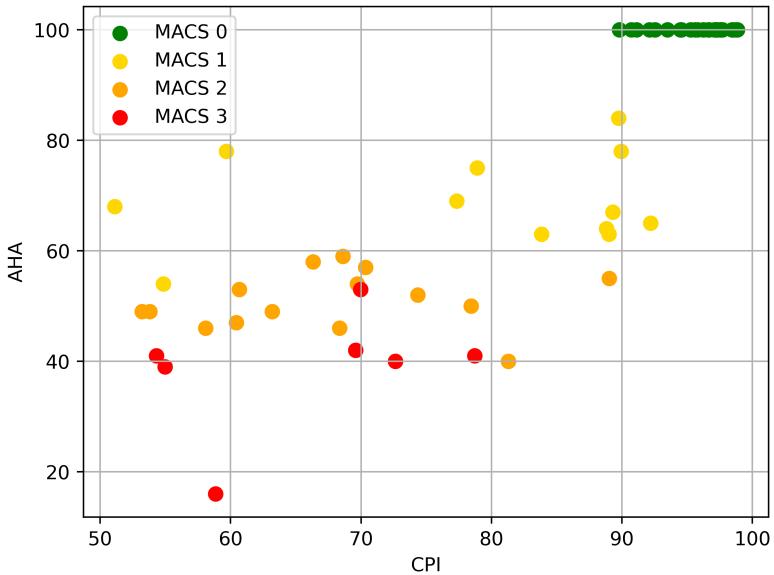


Figura 4.4: Scatterplot che rappresenta la correlazione tra punteggio AHA e CPI su sessioni WEEK.

La qualità della correlazione tra AHA e CPI (e la conseguente validità o meno di quest'ultima come feature) è quantificabile calcolando l'indice di correlazione di Pearson (ρ) tra le due variabili, e nel nostro caso:

$$\rho_{CPI,AHA} = 0.826 \quad (4.1)$$

Questo è un risultato importante poiché, a fronte di una correlazione fortemente positiva, **il CPI si può dunque considerare un indicatore utile nell'analisi di serie temporali da attigrafi e nel monitoraggio delle prestazioni dei soggetti nel loro ambiente domestico.**

Inoltre il CPI sarebbe ulteriormente valido perché facilmente migliorabile, dato che basa la sua efficacia sulla grandezza del dataset.

4.3 Home Assisting Hand Assessment

La seconda parte della pipeline estende l'utilizzo del CPI per allenare un regressore allo scopo di ricavare una stima dell'AHA (Home-AHA) a partire dalle sessioni WEEK, e si conclude nella realizzazione di un dashboard che visualizzi gli andamenti del CPI e dell'Home-AHA durante la settimana.

4.3.1 Andamento del CPI

Il CPI settimanale analizzato nella sezione precedente è un buon indicatore del comportamento generale assunto da un soggetto per la durata di una settimana, ma è un numero. L'idea sviluppata nelle prossime sezioni è quella di costruire un andamento del CPI: dato che la natura del nostro indicatore ci permette di calcolarlo su una qualsiasi durata di sessione WEEK, si possono considerare più finestre di dimensione fissa all'interno dei sei giorni e calcolare su queste il CPI, costruendo un grafico.

Nelle prossime sottosezioni è descritto un processo di ottimizzazione che descrive la costruzione di questo andamento, a partire dal modello scelto precedentemente alla fine della sezione 4.2.2, insieme all'introduzione del grafico delle predizioni e della soglia di significatività, due preconcetti importanti per la comprensione della seconda parte della pipeline.

Grafico delle predizioni

Al fine di comprendere al meglio la natura dell'andamento del CPI, è necessario introdurre il concetto di grafico delle predizioni. Questo grafico non è altro che una visualizzazione della classificazione di ogni sample della sessione WEEK:

- Sulle ascisse ci sarà la serie dei sample da cinque minuti rappresentante l'intera sessione WEEK;
- Sulle ordinate ci sarà la predizione associata ad un sample: 1 per il gruppo di controllo, 0 per i sample non validi e -1 per il gruppo emiplegico.

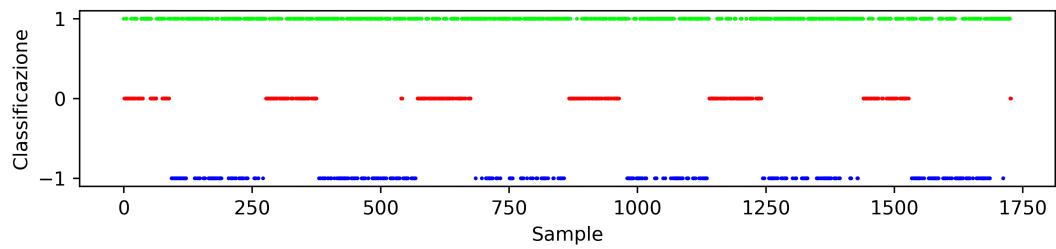


Figura 4.5: Predizioni sui sample della sessione week di un soggetto con emiplegia.

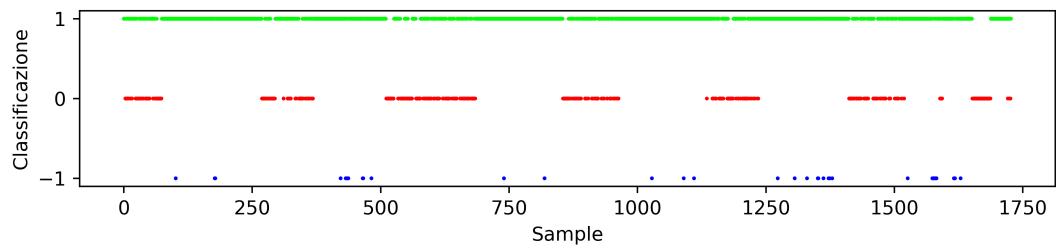


Figura 4.6: Predizioni sui sample della sessione week di un soggetto del gruppo di controllo.

Su questo grafico possiamo calcolare il CPI su un numero di sample arbitrario, e costruire un andamento.

Per proseguire è necessario scegliere un intervallo temporale adatto a rendere il calcolo del CPI significativo: un intervallo troppo piccolo rischierebbe di compromettere l'andamento con troppo rumore, d'altra parte un intervallo troppo grande restituirebbe un andamento con troppi pochi punti.

Si è optato dunque per finestre di sei ore (72 sample da 5 minuti l'una), misura considerata un giusto compromesso tra significatività del calcolo del CPI e un'adeguata visualizzazione.

Soglia di significatività

Prima di iniziare a parlare della realizzazione degli andamenti, è opportuno introdurre il concetto di soglia di significatività e di come sia importante per una visualizzazione

efficace.

Essendo il processo di visualizzazione basato sul grafico delle predizioni, sarebbe sconsigliato calcolare un CPI su una data finestra temporale al cui interno non vi siano abbastanza sample validi. Infatti, su 72 sample contenuti all'interno di una finestra, potrebbe presentarsi il caso in cui nessuno di questi o soltanto una minima parte sia valido.

Questo comporterebbe la creazione di un andamento rumoroso, che tenderebbe nel caso del CPI (e vedremo anche di conseguenza nel caso dell'Home-AHA) a dei picchi non significativi intorno a quei momenti in cui il soggetto è immobile (quindi quando produce sample non validi), solitamente la notte.

La soluzione a questo problema che abbiamo implementato è l'inserimento di una soglia di significatività sulla quale si basa la scelta di considerare o meno il CPI calcolato su una data finestra temporale. A seconda del variare di questa soglia, si ottengono risultati più o meno migliori nella visualizzazione efficace di un andamento.

E' quindi di fondamentale importanza scegliere in maniera corretta il valore di questa soglia. Un metodo di visualizzazione che ci aiuta in questo caso è il grafico della significatività, che introdurremo in seguito.

Di seguito alcuni esempi di buona e cattiva visualizzazione, con due soglie di significatività differenti.

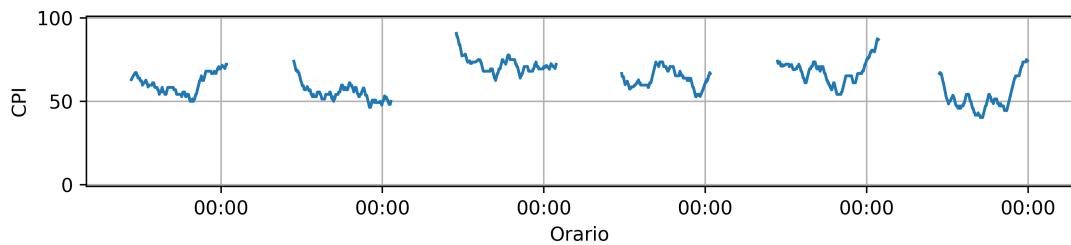


Figura 4.7: Andamento del CPI tenendo conto solo delle finestre con significatività $\geq 75\%$.

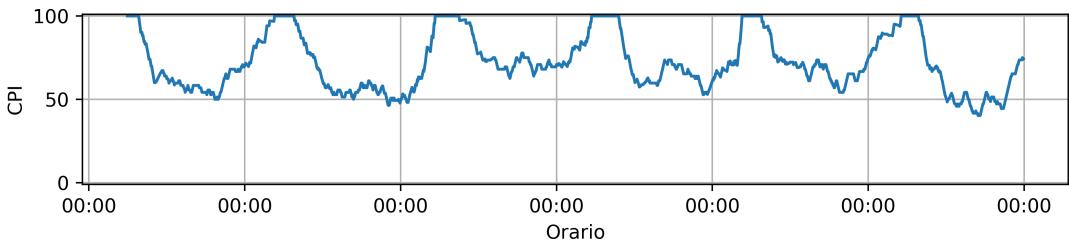


Figura 4.8: Andamento del CPI tenendo conto di tutte le finestre.

Come si può vedere nella figura 4.7, considerare tutte le finestre come valide risulta in valori di CPI spropositatamente alti durante i periodi di maggiore immobilità: le notti.

Il grafico 4.8 implementa invece il controllo sulla soglia di significatività, ottenendo un buon risultato. E' inevitabile però che in alcuni punti dell'andamento si manifestino alcune code dei originati durante la notte, essendo comunque il controllo effettuato una misura blanda di prevenzione: controlli più mirati sono comunque possibili, ma si è scelto di cercare di utilizzare una misura unificata per semplicità.

Finestre sequenziali disgiunte

Un primo approccio per la visualizzazione dell'andamento del CPI è quindi quello di calcolarlo a gruppi di 72 sample in modo sequenziale, una finestra dietro l'altra, senza nessun overlap. Questo, combinato con la funzione 'steps-post' di matplotlib [20], che ci permette di definire lo stile grafico in termine di gradini verticali e orizzontali che si connettono ai punti dati⁴, restituisce una rappresentazione discreta della serie temporale e dell'andamento del CPI di 6 ore in 6 ore.

Prima di effettuare il calcolo del CPI su una data finestra, si controlla che questa contenga una percentuale minima di sample significativi. Se la finestra supera il controllo della soglia di significatività, si può proseguire a calcolare il CPI. Se questa

⁴In particolare, collega i punti con linee orizzontali rappresentanti il valore dell'ordinata del punto precedente, e linee verticali rappresentanti il valore delle ascisse del punto successivo.

non supera il controllo, si aggiunge alla serie da disegnare un valore nativo di NumPy: np.nan (Not-A-Number) [21], in modo tale che venga lasciato uno spazio vuoto nell'andamento.

Di seguito lo pseudocodice per il calcolo del CPI su finestre sequenziali disgiunte. Si fa uso della sintassi list comprehension di Python per ricavare le finestre, e in particolare nella funzione range() si specifica il passo da effettuare tra un ciclo e l'altro, in questo caso 6 ore.

```
1 CPI_list = []
2 # List comprehension per creare le finestre
3 windows = [predictions[n:n+window_size] for n in range(0, len(
4     predictions), window_size)]
5
6 # Considero ogni finestra
7 for w in windows:
8
9     # Controllo se w contiene abbastanza sample validi
10    if valid_percentage < significativity_threshold:
11        # Appendo np.nan, che non verrà plottato.
12        CPI_list.append(np.nan)
13    else:
14        # Calcolo del CPI
15        CPI_list.append(calcolo del CPI)
16
17 # Disegno il grafico dell'andamento del CPI con drawstyle a
18     # gradini
19 matplotlib.pyplot.plot(CPI_list, drawstyle = 'steps-post')
```

Code 4.3: Pseudocodice finestre sequenziali disgiunte.

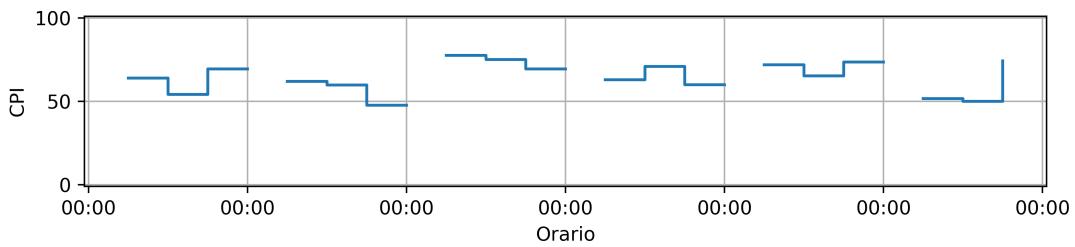


Figura 4.9: Andamento del CPI in finestre disgiunte da 6 ore nella sessione week di un soggetto con emiplegia.

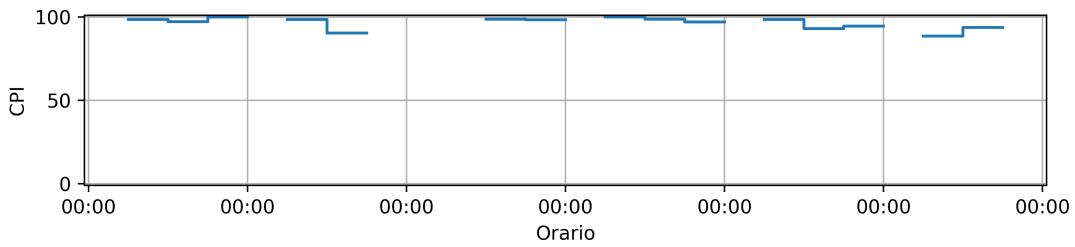


Figura 4.10: Andamento del CPI in finestre disgiunte da 6 ore nella sessione week di un soggetto del gruppo di controllo.

Come si può vedere dalle figure, un soggetto non emiplegico manifesta valori di CPI notevolmente più alti e in modo più generalizzato.

Finestra scorrevole

Una visualizzazione semi-continua dell’andamento del CPI è realizzabile tramite l’utilizzo di una finestra scorrevole.

La finestra scorrevole rimane di dimensione fissa, cioè di 6 ore, ma invece di calcolare i 72 sample successivi in maniera disgiunta, effettua uno shift di un sample verso destra, creando così un overlap di 71 sample tra una finestra e l’altra. In questo modo, stiamo aumentando notevolmente i punti appartenenti all’andamento, simulando uno stile di rappresentazione quasi continuo.

Anche qui, si effettua il controllo sulla soglia di significatività all'interno di ogni finestra, e in caso di soglia non raggiunta si appende np.nan, senza calcolare alcun CPI.

Di seguito lo pseudocodice che descrive il procedimento sopracitato. Anche qui si utilizza una list comprehension di Python, ma senza specificare il passo di iterazione nella funzione range() e utilizzando quindi lo step di default, cioè 1.

```
1 CPI_list = []
2 # List comprehension per creare le finestre
3 windows = [predictions[n:n+window_size] for n in range(0, len(
4     predictions)-window_size+1)]
5
6 # Considero ogni finestra
7 for w in windows:
8
9     # Controllo se w contiene abbastanza sample validi
10    if valid_percentage < significativity_threshold:
11        # Appendo nan, che non verrà plottato.
12        CPI_list.append(np.nan)
13    else:
14        # Calcolo del CPI
15        CPI_list.append(calcolo del CPI)
16
17 # Disegno il grafico dell'andamento del CPI
18 matplotlib.pyplot.plot(CPI_list)
```

Code 4.4: Pseudocodice finestra scorrevole.

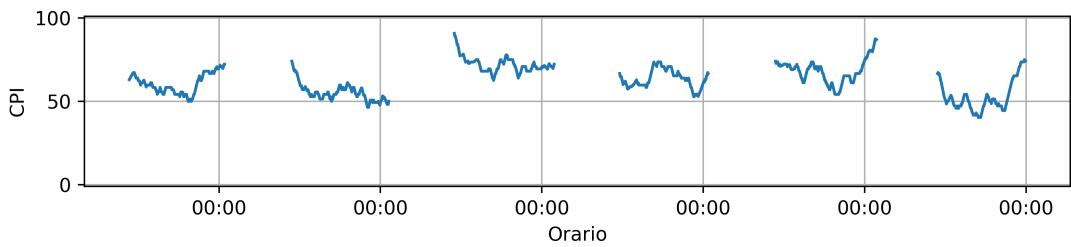


Figura 4.11: Andamento del CPI in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.

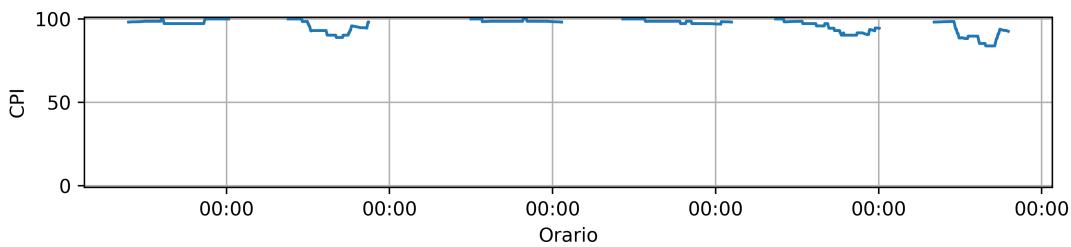


Figura 4.12: Andamento del CPI in finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.

Nuovamente, e similmente al commento sui grafici 4.9 e 4.10, notiamo come i soggetti non emiplegici riportino valori del CPI generalmente vicini al valore massimo.

Grafico della significatività

A supporto degli andamenti del CPI, e al fine di comprendere i momenti in cui le finestre siano cariche di significato o meno, si è implementato un andamento a finestra scorrevole delle percentuali di sample validi per ogni 6 ore: il grafico della significatività.

- Sulle ascisse ogni punto rappresenta una finestra di sei ore;
- Sulle ordinate ogni valore rappresenta la percentuale di sample considerati validi in quella finestra.

Un'analisi attenta del grafico della significatività per tutti i pazienti ci suggerisce un valore per la soglia di controllo nel calcolo di ogni andamento nel nostro dashboard: infatti, difficilmente una buona finestra scende sotto il 75% di sample validi. Questo valore verrà quindi utilizzato come soglia di significatività (introdotta nella sezione 4.3.1), considerato come un compromesso che distingue adeguatamente momenti di immobilità e momenti di attività motoria significativa.

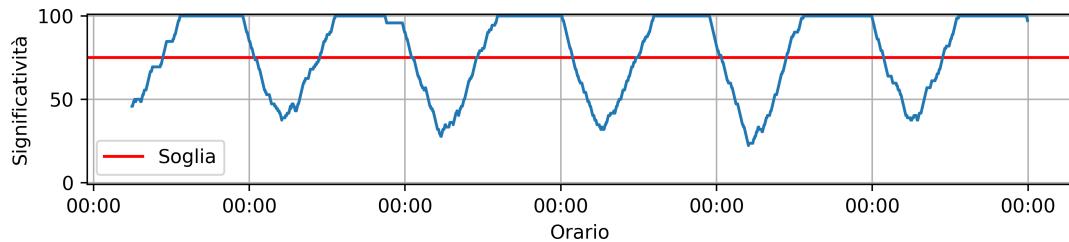


Figura 4.13: Andamento della significatività di finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.

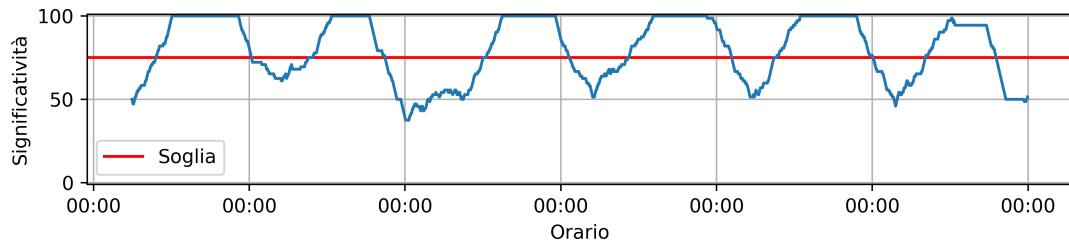


Figura 4.14: Andamento della significatività di finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.

Osservando l'andamento della significatività delle finestre scorrevoli dei due soggetti si nota come, seppure siano distinguibili i periodi notturni anche ad occhio nudo, nel caso del soggetto rappresentato nella figura 4.13 la significatività abbia dei minimi sempre inferiori rispetto a quella del soggetto nella figura 4.14.

Questo, come già accennato, impedisce di trovare una soglia che sia accurata e ugualmente efficace per isolare i periodi di sonno di tutti i soggetti.

4.3.2 Stimare l'AHA: Home-AHA

A questo punto del nostro studio, ci si ritrova con un CPI settimanale ed il suo andamento nel tempo. Questo indicatore rappresenta il comportamento dell'attività motoria di un soggetto, ma agli occhi di un team clinico è un altro indice numerico da prendere in considerazione e per di più non conforme al loro paradigma preesistente.

Per questo si è pensato ad un altro mezzo di monitoraggio del soggetto, cioè la stima dell'AHA a partire da un CPI: l'Home-AHA. Questo indice, e più nello specifico il suo andamento nel tempo, rappresenta un metodo di osservazione del soggetto in un periodo prolungato che sia confrontabile dai clinici con il relativo punteggio AHA, rendendo più intuitivo il monitoraggio.

Al fine di ottenere questa stima abbiamo deciso di utilizzare un regressore lineare, probabilmente la scelta più naturale in questo contesto.

Allenamento del regressore

È dunque necessario allenare un regressore, che sia in grado di trasformare un CPI in un punteggio Home-AHA.⁵

Nel processo di allenamento di un regressore, a questo punto del lavoro, si potrebbe pensare di costruire un dataset con sessanta CPI settimanali e i sessanta relativi punteggi AHA. Questo però non sfrutta a pieno la potenza di un regressore lineare, che permette invece di lavorare con un input di dimensione superiore a uno.

Da qui nasce l'idea di considerare previsioni di più modelli per l'allenamento del regressore, cioè più CPI settimanali. A questo punto, considerando l'opera di più modelli, un grafico dell'andamento del CPI conterrà tutti gli andamenti prodotti dai modelli scelti, generando le figure seguenti, ottenute tramite l'applicazione dei migliori cinque modelli con sampling da 300 secondi (tabella 4.1). Questi cinque modelli, tut-

⁵Naturalmente ogni punteggio Home-AHA verrà normalizzato in caso di eccessi: previsioni maggiori del punteggio massimo verranno riportati a 100.

ti con MTS superiore a 0.93, sono: due KMeans con DTW, un BOSSEnsemble, un KMedoids con distanza Euclidea e uno ShapeDTW con descrittore di forma PAA.

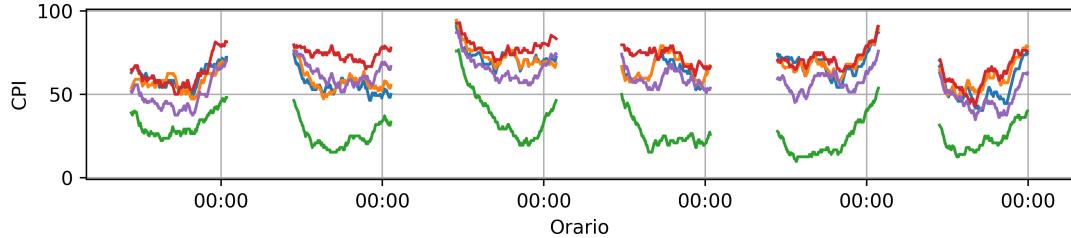


Figura 4.15: Andamento del CPI calcolato con 5 modelli differenti in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.

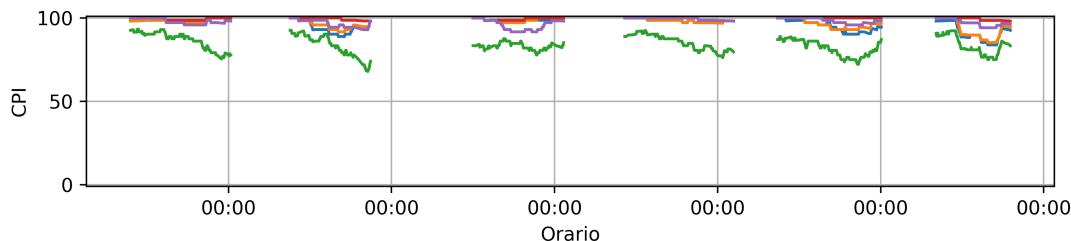


Figura 4.16: Andamento del CPI calcolato con 5 modelli differenti in finestre scorrevoli da 6 ore nella sessione week di un soggetto del gruppo di controllo.

Come si può vedere chiaramente dalle figure, nonostante i valori del CPI possano sembrare molto diversi per ogni modello, i pattern di questi andamenti sono molto simili tra loro. Inoltre, il regressore può adesso considerare risultati ottenuti sia da modelli supervised che da modelli non supervised, con differenti misure di distanza per questi ultimi (Euclidea e DTW), e soprattutto lavorare con tutti e tre i metodi di preprocessing, AI, differenza e concatenazione. Ciò permette all'algoritmo di avere più ampia capacità di generalizzazione per il calcolo di un singolo punto Home-AHA.

Il regressore lineare prenderà dunque come insieme di input una lista di cinque CPI settimanali prodotti dai cinque modelli differenti, e come target i sessanta punteggi AHA relativi ad ogni lista per ciascun soggetto. Naturalmente non è possibile riportare

la classica visualizzazione del regressore, dato che con l'aumento della complessità aumentano anche le dimensioni di rappresentazione.

Il regressore è stato poi validato ottenendo un risultato robusto tramite la media di numerose iterazioni di una 5-fold-cross-validation (it=1000). Lo score in questione è:

$$R^2 = 0.747 \quad (4.2)$$

Andamento dell'Home-AHA

Il calcolo dell'andamento dell'Home-AHA è basata sull'utilizzo di una finestra scorrevole (similmente a quanto visto nella sezione 4.3.1), combinato con l'applicazione del regressore lineare precedentemente allenato.

Proprio perché il calcolo di questo andamento condivide le stesse metodologie di quello del CPI, si ripresenta la stessa problematica dei momenti di immobilità. Questi, nonostante siano per la maggior parte evitati tramite il controllo della soglia di significatività, lasciano comunque alcune code dopo momenti di vuoto la cui ripida pendenza suggerisce l'inizio o la fine di un periodo di inattività motoria.

Oltre a questo, nel grafico è presente una linea orizzontale, utilizzata come reference per il punteggio AHA originale. L'andamento assumerà diverse colorazioni in accordo con la distanza (d) da questo riferimento:

- Grigio: se $-5 \leq d \leq 5$
- Verde: se $5 < d \leq 10$
- Verde scuro: se $d > 10$
- Arancione: se $-10 \leq d < -5$
- Arancione scuro: se $d < -10$

L'andamento dell'Home-AHA si presenterà quindi come le figure sotto. Ovviamente per i soggetti non emiplegici l'andamento sarà meno interessante, ma è stato riportato per completezza e come riprova del suo funzionamento.

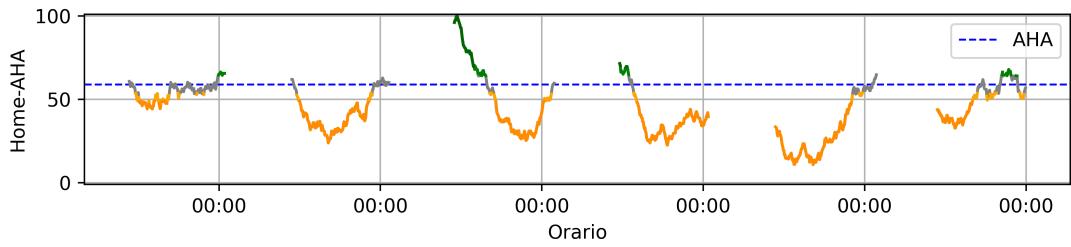


Figura 4.17: Andamento dell'Home-AHA calcolato da un regressore allenato su 5 modelli in finestre scorrevoli da 6 ore nella sessione week di un soggetto con emiplegia.

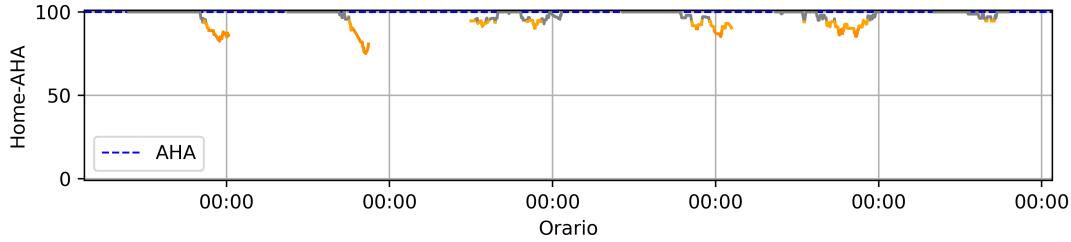


Figura 4.18: Andamento dell'Home-AHA calcolato da un regressore allenato su 5 modelli in finestre scorrevoli da 6 ore nella sessione week di un soggetto appartenente al gruppo di controllo.

4.3.3 Realizzazione del dashboard

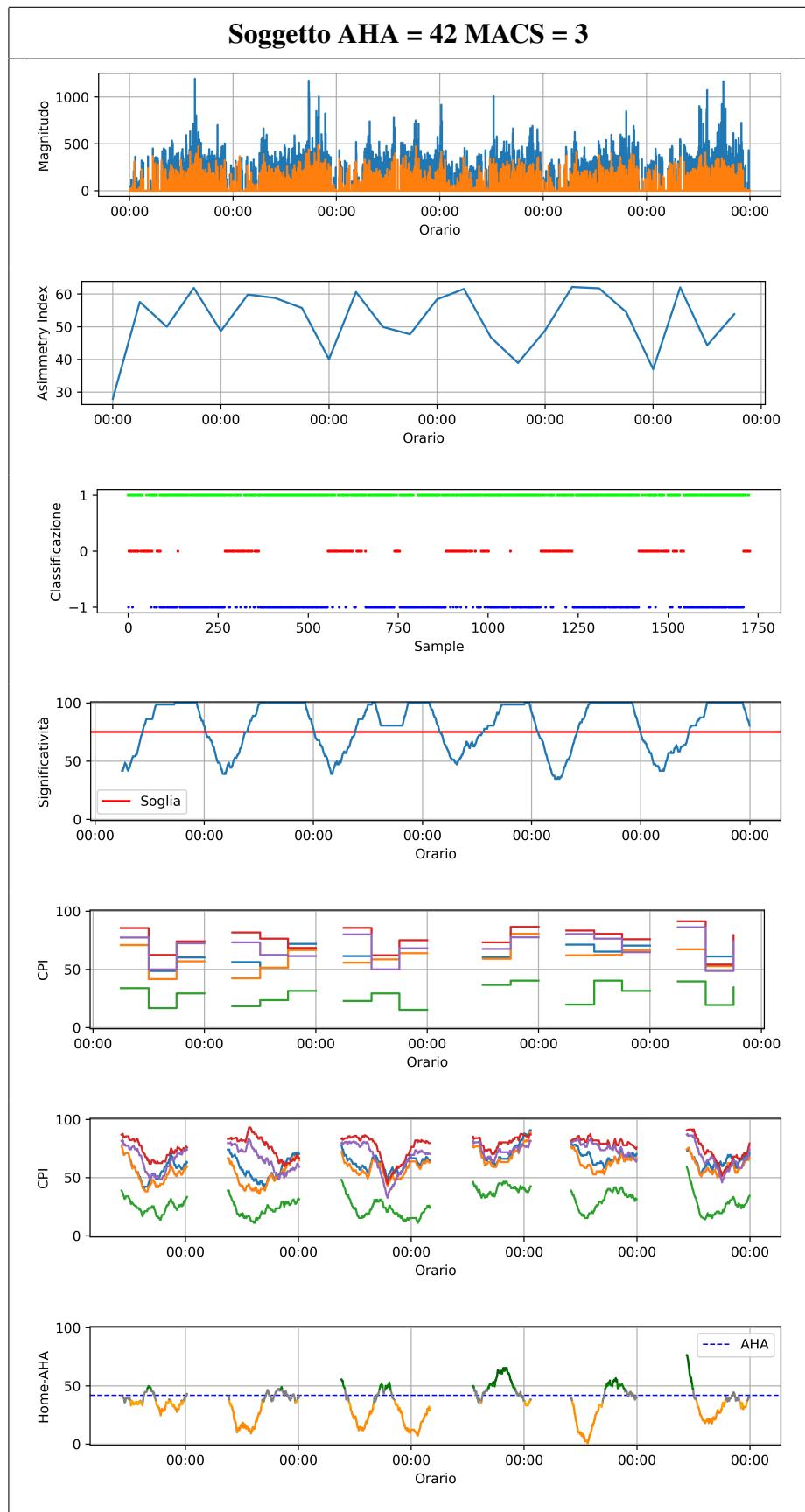
A questo punto della pipeline, abbiamo a disposizione tutti gli elementi per creare un dashboard di monitoraggio per ogni soggetto che comprenda due andamenti del CPI (uno “discreto” e uno “continuo”) e un andamento dell’Home-AHA, oltre al supporto del grafico delle predizioni e di quello della significatività.

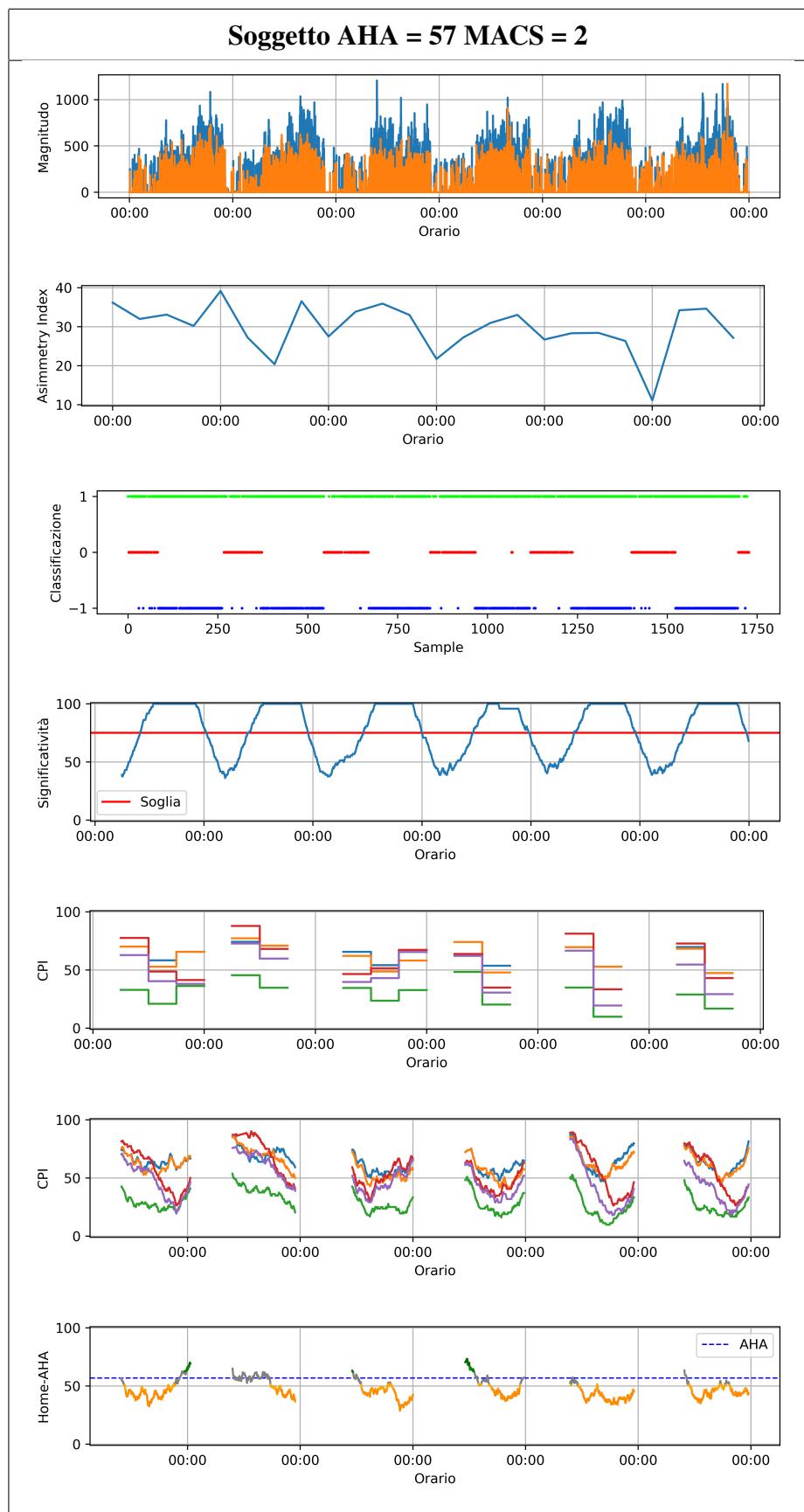
In aggiunta a questi, si presenta anche un ulteriore andamento: quello dell’AI calcolato su finestre sequenziali disgiunte da 6 ore (tramite la formula 3.1). Questo andamento è inserito nel dashboard per ultimare il quadro ora completo degli andamenti di ogni indice disponibile.

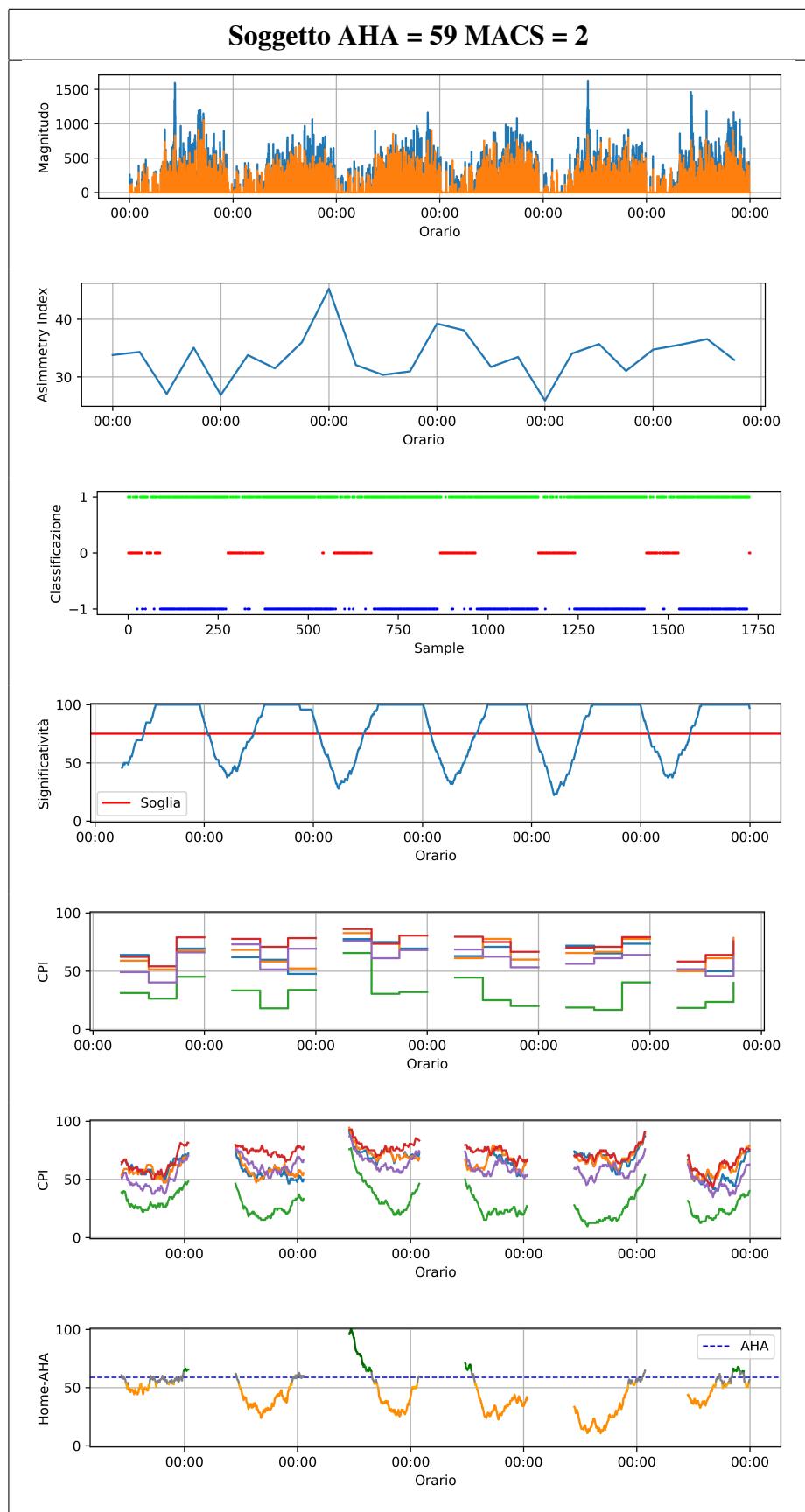
Analizziamo ora alcuni esempi di dashboard, completi anche della serie di magnitudo del soggetto. Si osservi che i grafici sono allineati in maniera tale da consentire un confronto più intuitivo tra ciascuno di loro.

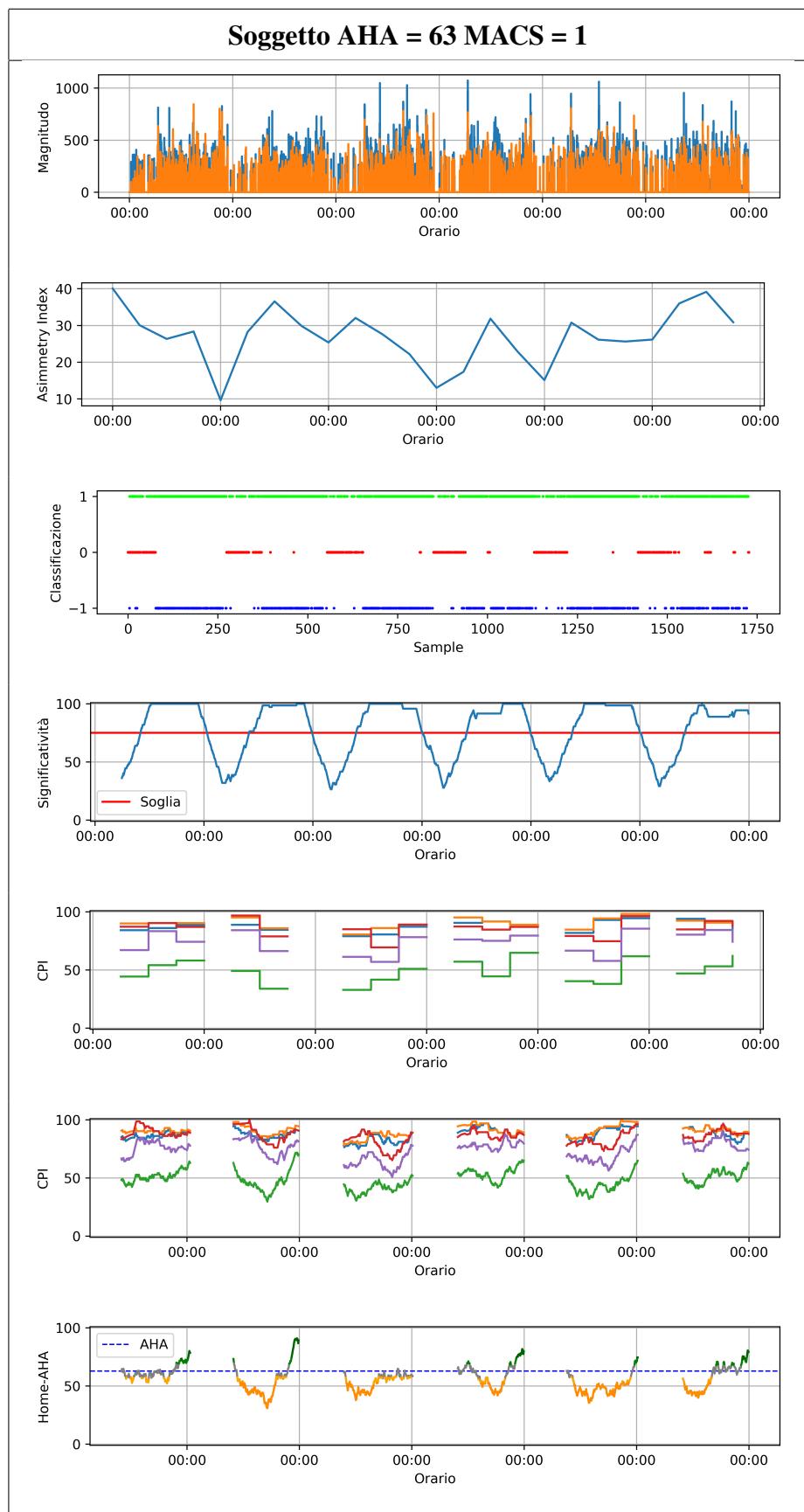
Ogni dashboard segue lo stesso ordine di grafici:

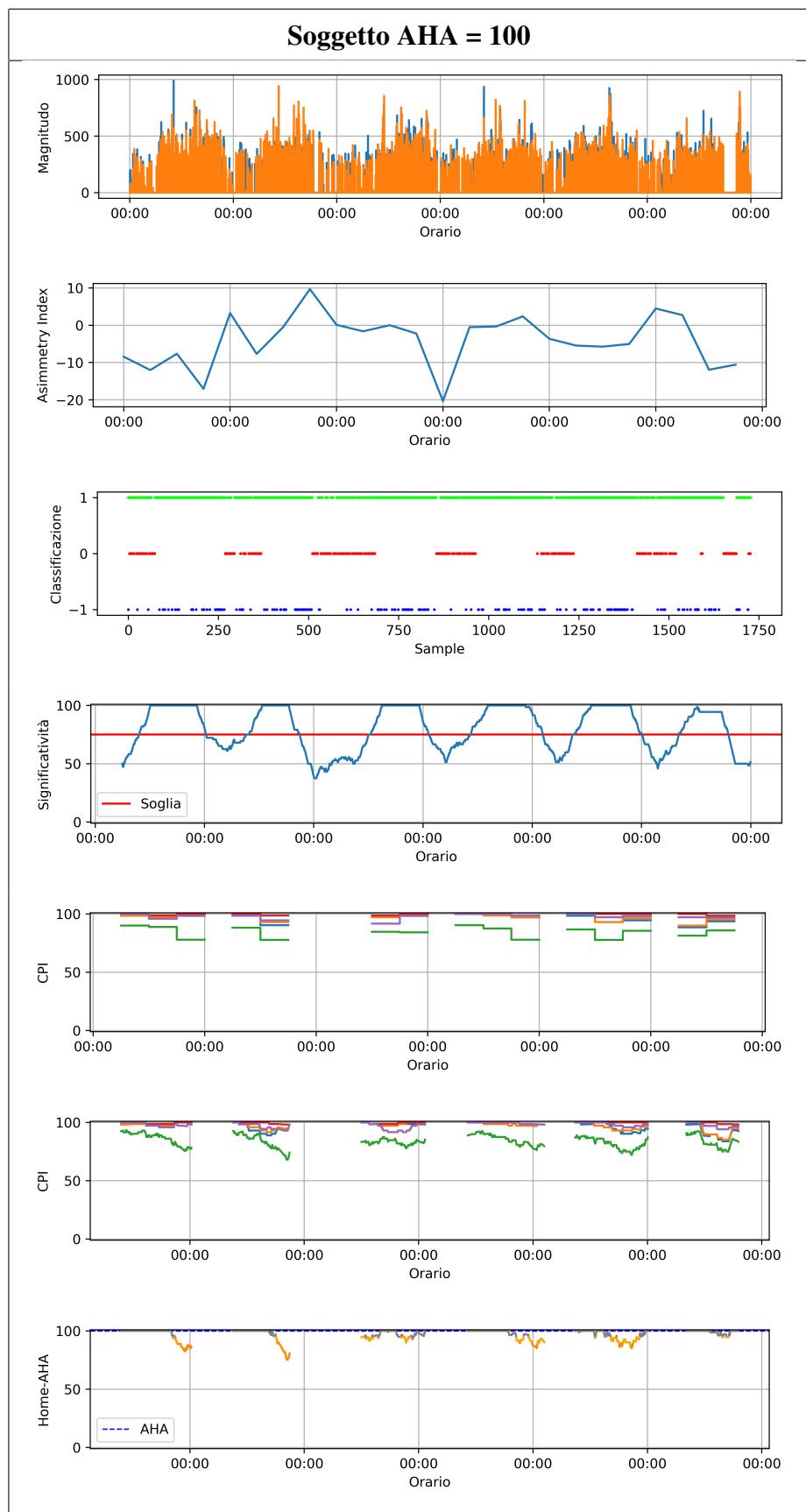
1. Andamento della magnitudo.
2. Andamento dell'AI.
3. Grafico delle predizioni.
4. Grafico della significatività.
5. Andamento del CPI su finestre sequenziali disgiunte.
6. Andamento del CPI su finestra scorrevole
7. Andamento dell'Home-AHA











5. RISULTATI

L'applicazione di differenti algoritmi e metodi di manipolazione delle serie temporali ci ha portato ad ottenere una feature e un grafico di cui si riassumono ora i punti chiave.

5.1 CPI come feature

L'applicazione di algoritmi di classificazione e clustering binari ha permesso il calcolo del CPI, di cui abbiamo misurato e comprovato la correlazione con l'AHA per poterla definire come feature significativa per il monitoraggio.

Tale metrica consente di riassumere, a partire dalle misurazione degli attifografi, le prestazioni del soggetto. Nonostante rimanga un indice meno indicativo di quello assegnato dal personale clinico è comunque una buona approssimazione che potrebbe tornare utile per valutare i pazienti nel loro ambiente domestico.

5.1.1 Modularità della pipeline

Per il calcolo del CPI la pipeline implementata è stata fatta in modo da poter gestire manipolazioni di serie temporali differenti tra loro e algoritmi di apprendimento sia supervisionato che non supervisionato. Nello specifico questo ultimo punto è stato possibile grazie alla funzione di scoring personalizzata.

5.2 Grafico dell'Home-AHA

Il CPI è stato inoltre utilizzato per poter allenare un regressore e per poter conseguentemente ottenere una stima dell'andamento dell'AHA di un paziente durante l'arco della settimana (Home-AHA).

Nonostante la durata delle sessioni WEEK sia nettamente maggiore della durata delle sessioni AHA, queste non sono controllate e soprattutto non sono strutturate appo-

sitamente per stimolare la bimanualità, quindi seppur il grafico dell'Home-AHA possa essere utilizzato come punto di partenza per il monitoraggio non è ovviamente in grado di sostituire una sessione AHA, ma può essere utile per capire in quali fasce orarie un soggetto presenta differenze più accentuate tra le abilità motorie degli arti superiori.

5.2.1 Activity recognition: notti

La rappresentazione dell'Home-AHA ha portato inoltre alla luce l'importanza di saper distinguere periodi più o meno significativi per il monitoraggio, e questo è possibile grazie al grafico della significatività e la sua soglia, che permettono dunque di distinguere momenti di maggiore attività da momenti in cui invece il soggetto è principalmente fermo, come ad esempio la notte.

6. CONCLUSIONI

I risultati ottenuti non si dimostrano particolarmente determinanti a livello clinico, tuttavia sono in grado di dimostrare che l'approccio di manipolazione e analisi delle serie temporali degli attigrafi può portare ad osservazioni significative.

Nella valutazione di questo lavoro infatti, bisogna comunque considerare che i risultati presentati sono il frutto di una fase di training effettuata solamente su 60 soggetti, troppo pochi per una conclusione di alto livello.

6.1 Margine di miglioramento

Nello specifico bisogna tenere conto anche del fatto che il CPI e l'Home-AHA sono frutto dell'addestramento di modelli di ML, e dunque i risultati da loro prodotti sono strettamente dipendenti dalla dimensione del dataset di partenza, quindi questo non esclude che, anche senza ulteriori modifiche, la stessa pipeline possa fornire migliori risultati se fosse applicata su un dataset migliore e più vasto.

6.2 Sviluppi Futuri

La pipeline implementata potrebbe essere migliorata e gioverebbe in termini di risultati e significatività degli stessi apportando modifiche nei seguenti ambiti.

6.2.1 Quantità e qualità dei dati

Come accennato precedentemente sarebbe utile in futuro poter valutare i risultati di alcuni modelli addestrati su un numero maggiore di soggetti, e magari a partire da sessioni AHA e sessioni WEEK di durata maggiore.

Potrebbero inoltre apparire significative altre metriche dei soggetti, e non solamente gli AC, quindi per studi futuri sarebbe utile poter rilevare altre metriche magari utilizzando anche altri tipi di sensori.

6.2.2 Esplorazione di ulteriori modelli e parametri

Nell’analisi i risultati derivano dall’utilizzo di due algoritmi supervisionati e due non supervisionati, i quali scelgono i propri parametri tra un elenco limitato rispetto a tutti quelli possibili. Seppure dunque gli MTS nella fase di model selection siano spesso alti sarebbe interessante comunque provare ad applicare altri algoritmi e ad esplorare maggiori possibilità in termini di parametri. Senza inoltre dimenticare che l’Home-AHA non necessariamente debba partire dai 5 migliori modelli, ma potrebbe utilizzarne una quantità maggiore.

6.2.3 Gestione e trasferimento dei dati

Le metodologie in cui i dati sono passati dai clinici agli esperti IT sono importanti al fine di avere un risultato soddisfacente. Per questo può essere di estrema utilità l’allestimento di una piattaforma apposita, in cui i clinici possano caricare i dati tramite un’interfaccia intuitiva in un formato unificato e su un database interrogabile, piuttosto che il perseverare con il passaggio a mano del dataset.

Su una piattaforma simile, si potrebbero integrare le attività svolte dai soggetti (contenute nei diari tenuti dai genitori durante le sessioni WEEK) associandole sotto forma di label ai rispettivi momenti della giornata nella serie temporale. Ciò potrebbe essere determinante per lo studio dell’activity recognition.

Bibliografia

- [1] Scylla DB. Time series data definition. <https://www.scylladb.com/glossary/time-series-data/>.
- [2] Introduction to k-fold cross-validation in python. <https://sqlrelease.com/introduction-to-k-fold-cross-validation-in-python>.
- [3] Overfitting and underfitting. <https://www.fastaireference.com/overfitting>.
- [4] Teemu Kanstrén. A look at precision, recall, and f1-score. <https://towardsdatascience.com/a-look-at-precision-recall-and-f1-score-36b5fd0dd3ec>.
- [5] Anmol Tomar. Stop using elbow method in k-means clustering, instead, use this! <https://towardsdatascience.com/elbow-method-is-not-sufficient-to-find-best-k-in-k-means-clustering-fc820da0631d>.
- [6] K-means clustering. <https://www.ml-science.com/k-means-clustering>.
- [7] Jiaping Zhao and Laurent Itti. shapedtw: shape dynamic time warping. *Pattern Recognition*, 74, 09 2017.
- [8] Patrick Schäfer. The boss is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, 29, 11 2015.
- [9] Suraj Verma. Normal equation in python: The closed-form solution for linear regression. <https://towardsdatascience.com/normal-equation-in-python-the-closed-form-solution-for-linear-regression-13df33f9ad71>.
- [10] ActiGraph. Gt3x. <https://theactigraph.com/actigraph-wgt3x-bt>.
- [11] Giordano Scerra Davide Marchi. Github open source code. <https://github.com/davide-marchi/Tesi-AInCP>.

- [12] Tom Van Steenkiste, Willemijn Groenendaal, Pauline Dreesen, Seulki Lee, Susie Klerkx, Ruben de Francisco, Dirk Deschrijver, and Tom Dhaene. Portable detection of apnea and hypopnea events using bio-impedance of the chest and deep learning. *IEEE Journal of Biomedical and Health Informatics*, 24(9):2589–2598, 2020.
- [13] E. Beani, Martina Maselli, Elisa Sicola, Silvia Perazza, Francesca Cecchi, Paolo Dario, Irene Braito, Roslyn Boyd, Giovanni Cioni, and Giuseppina Sgandurra. Actigraph assessment for measuring upper limb activity in unilateral cerebral palsy. *Journal of NeuroEngineering and Rehabilitation*, 16, 02 2019.
- [14] Martin Bax, Murray Goldstein, Peter Rosenbaum, Alan Leviton, Nigel Paneth, Bernard Dan, Bo Jacobsson, and Diane Damiano. Proposed definition and classification of cerebral palsy, april 2005. *Developmental Medicine and Child Neurology*, 47(8):571–576, 2005.
- [15] Lena Kruhlind-Sundholm, Marie Holmefur, Anders Kottorp, and Ann-Christin Eliasson. The assisting hand assessment: Current evidence of validity, reliability, and responsiveness to change. *Developmental Medicine and Child Neurology*, 49:259 – 264, 04 2007.
- [16] Ann-Christin Eliasson, Lena Kruhlind-Sundholm, Birgit Rösblad, Eva Beckung, Marianne Arner, Ann-Marie Ohrvall, and Peter Rosenbaum. The manual ability classification system (macs) for children with cerebral palsy: scale development and evidence of validity and reliability. *Developmental medicine and child neurology*, 48:549–54, 08 2006.
- [17] Actilife. <https://theactigraph.com/academic-research#actilife>.
- [18] Sktime api. http://www.sktime.net/en/latest/api_reference.html.
- [19] Sklearn api. <https://scikit-learn.org/stable/modules/classes.html>.

[20] Matplotlib api. https://matplotlib.org/stable/api/_as_gen/matplotlib.lines.Line2D.html.

[21] Numpy nan. <https://numpy.org/doc/stable/reference/constants.html#numpy.nan>.