Progetto MOBD A.A. 2018-2019
P300 Speller for users with ALS

Federico Viglietta - Tommaso Villa

Il presente documento illustra le metodologie e le tecniche impiegate per l'addestramento di una SVM, finalizzato alla predizione dei caratteri comunicati da un utente affetto da SLA mediante l'uso di una Brain Computer Interface.

Import del dataset Poiché la dimensione del file X.txt è considerevole, è stata utilizzata la funzione $fread$ del package *data.table*, al fine di accelerare le operazioni di import.

Data Understanding In questa fase è emerso che il dataset non presenta né duplicati né valori mancanti. Di contro, è stata rilevata una frazione esigua di presunti outlier. Tuttavia, non avendo gli strumenti per certificare l'anomalia di tali valori, si è scelto di non rimpiazzarli. Infine, è stato osservato che i caratteri a disposizione si presentano nell'ordine con cui compaiono all'interno delle parole impiegate nella procedura di calibrazione.

Data Shuffling Per evitare che l'ordine dei dati influenzasse la generazione del modello di machine learning, i caratteri di training sono stati mescolati e il dataset è stato ristrutturato di conseguenza. Invece, non si è ritenuto opportuno mescolare le iterazioni relative al singolo carattere, perchè, come osservato in [1], dato che si ha a che fare con un processo cognitivo, non è ragionevole supporre che le prime e le ultime iterazioni siano equivalenti; altrimenti, non si terrebbe conto di aspetti come la fatica, l'abitudine, l'attenzione.

Data Splitting Dei 30 caratteri a disposizione, il 70% (21 caratteri) è stato utilizzato per il training; il restante 30% (9 caratteri) è stato impiegato per il test. Poiché ad ogni carattere sono associati 120 segnali EEG,

di cui 20 rispondenti a stimoli target e 100 rispondenti a stimoli non-target, la suddivisione per caratteri ha portato automaticamente al bilanciamento dello split sulla base della classe.

Feature Selection Allo scopo di ridurre la dimensionalità del problema, sono stati effettuati due tentativi di Feature Extraction.

In primo luogo, si è cercato di identificare la presenza di elettrodi non rilevanti ai fini della classificazione. Poiché l'approccio enumerativo usato in [3] sarebbe stato eccessivamente oneroso, si è proceduto nel modo seguente: per ciascun elettrodo è stata calcolata la media sui 204 istanti di campionamento disponibili; le 8 feature così ottenute sono state filtrate attraverso il metodo $ReliefF$ con un numero di iterazioni pari alla dimensione del dataset; infine, sono state selezionate le feature che hanno totalizzato uno score negativo. Da questa analisi è emerso che nessuno degli 8 elettrodi può essere considerato irrilevante. Ciò è in accordo con gli studi presenti in letteratura, nei quali la channel selection viene applicata su un numero maggiore di attributi (e.g. 64).

In secondo luogo, si è cercato di scartare gli istanti di campionamento non significativi. In questo caso, innanzitutto è stato applicato $ReliefF$ su tutti gli attributi; successivamente, per ogni istante di campionamento è stato calcolato il punteggio medio attribuito dal filtro; infine, sono stati selezionati gli istanti con punteggio negativo. Anche in questo caso, nessun istante di campionamento è risultato trascurabile.

In conclusione, il numero di feature utilizzate è rimasto invariato.

Standardizzazione Come buona norma, il training set è stato standardizzato. Media e fattore di scala del training set sono stati memorizzati in vista della futura standardizzazione del test set.

Model Selection Come osservato in [3], le risposte ERP presentano una variabilità elevata, anche nell'ambito dello stesso utente. Quindi, per prevenire l'overfitting si è scelto di impiegare una SVM con kernel lineare. Dato che con questo kernel è stata ottenuta un'accuratezza soddisfacente sul test set, non si è ritenuto necessario valutare le performance del kernel gaussiano, che è l'altra tipologia di kernel frequentemente impiegata per questa classe di problemi, come emerge dalla letteratura in materia. Per quanto riguarda il tipo di formulazione del

problema, si è scelta la duale, che, nel caso di specie, come è stato verificato sperimentalmente, riduce i tempi di addestramento.

Cross-Validation Per impostare il parametro C del kernel lineare, è stata eseguita una 5-fold cross-validation. Al fine di ottenere un risultato più robusto, la procedura è stata ripetuta per 5 split training-validation diversi. Per ridurre i tempi del tuning, l'esecuzione delle 5 iterazioni è stata parallelizzata usando le funzioni della libreria *parallel*. La C scelta è stata quella che massimizzava l'accuratezza media nelle varie iterazioni.

Model Evaluation Il test set è stato standardizzato usando le statistiche sul training set precedentemente raccolte. Per ottenere i caratteri predetti si è seguito un approccio che ha preso spunto da quello descritto in [2]: per ogni trial, ad ogni riga e colonna è stato assegnato un punteggio, che è la media dei decision value attribuiti dal modello nelle 10 iterazioni; la riga e le colonna che ottengono il punteggio massimo individuano il carattere predetto.

Final Model Alla fine, il modello è stato addestrato sull'intero dataset a disposizione con la C precedentemente calcolata. Per valutare le prestazioni su un nuovo test set sarà necessario standardizzare quest'ultimo con le statistiche del training set completo.

References

[1] Ulg Grossekathoefer Thomas Lingner Matthias Kaper, Peter Meinicke and Helge Ritter. Bci competition 2003-data set iib: Support vector machines for the p300 speller paradigm. *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, 51(6):1073–1075, 2004.

[2] Guigue V. Rakotomamonjy, A. Bci competition iii: Dataset ii- ensemble of svms for bci p300 speller. *IEEE TRANSACTIONS ON BIOMEDICAL ENGINEERING*, 55(3):1147–1153, 2008.

[3] Guigue V. Mallet G. Alvarado V. Rakotomamonjy, A. Ensemble of svms for improving brain computer interface p300 speller performances. *Lecture Notes in Computer Science*, 2005.