Progetto MOBD: P300 Speller with patient with ALS

Diletta Lagomarsini, Giorgia Marchesi

Introduzione

Le Brain Computer Interfaces (BCI) si basano sulla creazione di un canale di comunicazione tra cervello umano e computer, o altri dispositivi: acquisiscono i segnali cerebrali, li analizzano e li traducono in comandi che vengono trasmessi ai dispositivi di output che eseguono le azioni desiderate. L'obiettivo principale delle BCI è sostituire o ripristinare funzioni utili per le persone con disabilità neuromuscolari come sclerosi laterale amiotrofica, paralisi cerebrale, ictus o lesioni del midollo spinale. Una delle applicazioni del BCI riguarda persone con disturbi del linguaggio e, sfruttando l'apprendimento automatico, consente loro di comunicare solo per mezzo di idee grazie al riconoscimento delle parole tramite la corretta classificazione dei segnali cerebrali generati nel cervello. In questo progetto cerchiamo di massimizzare l'accuratezza della classificazione delle parole in persone malate di SLA grazie al confronto di alcuni classificatori che, secondo lo stato dell'arte del BCI in termini di *speech recognition*, risultano più adeguati a tale scopo. In particolare, sono stati presi in considerazione LibLinear, LDA, Naïve Bayes [1].
Il dataset analizzato è stato realizzato a partire da un esperimento condotto su alcuni pazienti malati di SLA che, data una parola, dovevano focalizzare l'attenzione sui caratteri della stessa mostrati attraverso una matrice 6x6 di caratteri alfanumerici. Tutte le righe e le colonne della matrice venivano illuminate randomicamente, pertanto solo 2 intensificazioni su 12 contenevano il carattere desiderato, ed i segnali EEG venivano registrati da 8 elettrodi. Questa operazione veniva eseguita per ogni carattere per 10 volte. Sono state mostrate in questo modo 6 parole di 5 caratteri ciascuna e per ogni canale sono stati selezionati 240 campioni. Pertanto, il dataset finale consta di 1634 colonne rappresentanti il dominio nel tempo (240 campioni x 8 canali + stimolo target + tipo dello stimolo) e 3600 righe rappresentanti le istanze (6 parole x 5 caratteri x 120 ripetizioni/carattere).

Feature transformation

Nel corso delle sperimentazioni sui classificatori sono stati presi in considerazione tre diverse manipolazioni del dataset:
1) dataset originale
2) dataset con media ogni 10 istanze corrispondenti alle iterazioni per righe e colonne sia per diminuire il rumore tra i vari segnali cercando di ottenere un unico segnale sia per avere un'unica iterazione
3) Dataset con media ogni 10 istanze e media sul dominio del tempo con binning variabile.
Il dataset originale è stato usato solo in prima battuta per avere un'idea di massima sulla qualità dei classificatori presi in considerazione mentre le altre trasformazioni hanno avuto poi come principale obiettivo quello di diminuire il numero di campioni siamo intervenute sia sulle righe che sulle colonne tramite l'operatore media.
Per eseguire la media sulle dieci iterazioni delle righe e colonne che rappresentano i caratteri, queste sono state innanzitutto raggruppate in base all'uscita *C* ed in seguito è stata eseguita la media (funzione *apply_media* nel file *data_transformation.R)*. Per quanto riguarda invece la media lungo la dimensione tempo (file *binning.R*), è stata effettuata la media tra le colonna in base alla dimensione del binning desiderata, parametro configurabile nel *main.R*,
Ulteriori esperimenti per quanto riguarda la manipolazione del dataset, sono stati effettuati utilizzando la *media mobile* e/o la *detrendizzazione* del segnale EEG, scartati successivamente a seguito degli scarsi risultati (le funzioni sono disponibili sempre presso il file *data_transformation.R* rispettivamente in *apply_media_mobile* e *apply_diff* / *apply_diff_per_sensore*), così come il tentativo di esclusione di alcuni canali, idea nata a seguito dello studio della correlazione tra sensori che in alcuni casi risultava molto elevata. Sapendo che la P300 si manifesta attorno ai 300 ms di simulazione e che ogni campionamento del segnale per sensore è di circa 800 ms, sembrava ragionevole ridurre l'intervallo totale eliminando alcune colonne iniziali e finali, ma i plot dei segnali hanno dimostrato una differenza significativa nella forma dei campioni, probabilmente dal momento che i soggetti malati di SLA possono generare una P300 inaspettatamente.

Feature selection

La feature selection è stata implementata in ultima battuta per cercare di migliorare ulteriormente, qualora possibile, i risultati dei classificatori. Abbiamo scelto il filtro *ReliefF* che è in grado di trattare dati mancanti e/o rumorosi anche a fronte di problemi di classificazione multi-classe [3]. ReliefF calcola uno score per ogni feature che può quindi essere applicato per classificare e selezionare le caratteristiche con il punteggio migliore per eseguire feature selection o in alternativa, per applicare i punteggi come pesi delle caratteristiche per guidare la modellazione a valle. Abbiamo optato per quest'ultima modalità, ovvero abbiamo usato ReliefF per assegnare pesi differenti alle features in sede di classificazione. Poiché ReliefF non è più disponibile nel package *FSelector*, è stata usata la funzione *attreval* del package *CORELearn* (come visibile da riga 60 a riga 66 del *main.R*). Come stimatore è stato scelto ReliefFbestK, che testa tutti i possibili k (che rappresentano k istanze più vicine) e per ciascuna caratteristica restituisce il punteggio più alto. Le istanze più vicine hanno pesi uguali.

Classificatori

Abbiamo preso in esame i seguenti classificatori supervisionati: *LibLinear*, *Linear Discriminant Analysis* e *Naïve Bayes*, a seguito dello studio di alcuni articoli scientifici. LibLinear trova il migliore iperpiano di separazione tra le classi mentre LDA trova le direzioni che massimizzano la separazione tra le classi, quindi usa queste direzioni (discriminanti lineari) per prevedere la classe degli individui. Naïve Bayes è un'interessante applicazione del teorema di Bayes al problema della classificazione. Il motivo per cui l'algoritmo è chiamato ingenuo è perché si presume che 1) ogni feature sia linearmente indipendente dalle altre features, e 2) ogni feature nel set di dati sia importante quanto qualsiasi altra caratteristica. Fortunatamente Naïve Bayes, nonostante le forti assunzioni, produce ottimi risultati anche quando queste ipotesi vengono violate.
Per Liblinear e LDA non sono state usate le formule standard ma sono state selezionate le classi target in base al valore massimo tra i *decision values*. Anche per il Naive Bayes, il risultato della predizione è stato manipolato per poter cercare di ridurre l'errore nei casi limite che generano indecisione nel simulatore (per visualizzare i raw values è stato necessario imporre *type = "raw"* nella predict). Si noti che è stato utilizzato un Naive Bayes con *kernel gaussiano* che consente generalmente di ottenere risultati migliori. Per capire meglio come sono stati gestiti i decision values mostriamo nelle figure sottostanti il risultato dei decision values ottenuti sulla parola "2BACI", soffermandoci sulle righe del carattere '2' (correttamente classificate, figura 1) e sulle colonne (mal classificate, figura 2).

	-1	1	new_Y	old_Y
49	1.000000e+00	2.001816e-42	-1	-1
50	2.812334e-17	1.000000e+00	1	1
51	1.000000e+00	2.681875e-38	-1	-1
52	1.000000e+00	3.904683e-10	-1	-1
53	1.000000e+00	2.373854e-19	-1	-1
54	1.000000e+00	1.502918e-31	-1	-1

Figura 1: risultato classificazione per la determinazione della riga del carattere '2' senza ReliefF

	-1	1	new_Y	old_Y
7	1.000000e+00	4.367202e-29	-1	-1
8	9.999999e-01	1.195812e-07	-1	-1
9	1.000000e+00	3.521980e-58	-1	-1
10	4.068431e-30	1.000000e+00	1	1
11	3.757757e-09	1.000000e+00	-1	1
12	1.000000e+00	2.948192e-43	-1	-1

Figura 2: risultato classificazione per la determinazione della colonna del carattere '2' senza ReliefF

Ci sono tre possibili casi:
1) la classificazione è corretta: si riportano i risultati sulla colonna 'new_Y';
2) il classificatore assegna ad una colonna (o riga) di un carattere (corrispondenti a sei righe consecutive del training set) più di un 1 nella classe target '1': tutte le righe contenenti 1 e non aventi l'uscita a '-1' corrispondente al valore minimo vengono settate a -1 nella colonna 'new_Y';
3) tutte le righe di una riga o colonna vengono classificate come non target: si seleziona come uscita target corretta quella corrispondente alla riga contenente il massimo nella colonna degli '1'.
In figura 2 è possibile apprezzare il risultato della classificazione base (nella colonna 'old_Y') rispetto a quella manipolata (colonna 'new_Y'). La manipolazione mitiga parzialmente gli errori di classificazione soprattutto se non si introduce la feature selection con ReliefF, che difatti diminuisce il numero di casi limite.

Test e risultati

I test sono stati eseguiti splittando il dataset in training set e test set. In particolare, per il primo sono state usate 5 parole e per il secondo 1 parola. I risultati in fase di addestramento sono stati ottenuti tramite k-fold cross-validation sui dati di training tenendo 4 parole per il training ed 1 parola per il validation set. Il test finale è stato eseguito addestrando la macchina su tutto il training set e usando come test set l'ultima parola lasciata da parte. La correttezza della classificazione è stata valutata per riga e colonna individuate correttamente per ogni carattere. In figura 3 riportiamo i risultati.

Conclusioni

Analizzando i risultati si nota come binning differenti comportino risultati differenti nei vari classificatori. Applicando sul dataset media sulle istanze, media sulla dimensione tempo ed il filtro ReliefF, si raggiunge un'accuratezza e un TPR (True Positive Rate) pari al 100% sul test set sia con LDA che con LibLinear, rispettivamente con binning pari a 12 e 6. Viceversa, Naive rimane invariato e si attesta attorno al 96,67%. Pertanto, i risultati di LDA e LibLinear sono assolutamente paragonabili.
Per assicurare la corretta esecuzione del codice, fare riferimento al *README.txt* presente nella cartella del progetto.

Riferimenti bibliografici

[1] U. Hoffman, J.M. Vesin, T. Ebrahimi, K. Diserens *"An efficient P300-based brain-computer interface for disabled subjects"*, 2007
[2] F Lotte, *"A review of classification algorithms for EEGbased brain–computer interfaces: a 10-year update"*, 2018
[3] Chiara Liti, "*Laboratorio MOBD - Feature Selection"*, 2019

	LibLinear		LDA		Naive Bayes	
Tipo dataset	**Cross validation**	**Test**	**Cross validation**	**Test**	**Cross validation**	**Test**
dataset originale	**accuracy:** 0.95334 **TPR:** 0.74000	**accuracy:** 1 **TPR:** 1	**accuracy:** 0.94 **TPR:** 0.82	**accuracy:** 0.9667 **TPR:** 0.9	**accuracy:** 0.91332 **TPR:** 0.74000	**accuracy:** 0.9667 **TPR:** 0.9
media sulle istanze	**accuracy:** **TPR:**	**accuracy:** **TPR:**	**accuracy:** **TPR:**	**accuracy:** **TPR:**	**accuracy:** **TPR:**	**accuracy:** **TPR:**
media + binning = 4 + ReliefF	**accuracy:** 0.96 **TPR:** 0.88	**accuracy:** 0.9667 **TPR:** 0.9	**accuracy:** 0.95334 **TPR:** 0.86000	**accuracy:** 0.8667 **TPR:** 0.6	**accuracy:** 0.92 **TPR:** 0.76	**accuracy:** 0.9667 **TPR:** 0.9
media + binning = 6 + ReliefF	**accuracy:** 0.95332 **TPR:** 0.86	**accuracy:** 1 **TPR:** 1	**accuracy:** 0.92 **TPR:** 0.76	**accuracy:** 0.9333 **TPR:** 0.8	**accuracy:** 0.92666 **TPR:** 0.78000	**accuracy:** 0.9667 **TPR:** 0.9
media + binning = 12 + ReliefF	**accuracy:** 0.98668 **TPR:** 0.78000	**accuracy:** 0.9667 **TPR:** 0.9	**accuracy:** 0.98668 **TPR:** 0.96000	**accuracy:** 1 **TPR:** 1	**accuracy:** 0.92668 **TPR:** 0.78000	**accuracy:** 0.9667 **TPR:** 0.9

Figura 3: tabella riassuntiva risultati ottenuti