Assignment 5 (Advanced Machine Learning)

Davide Sangalli - Matricola: 848013

December 9, 2019

Abstract

In questo assignment si affronterà l'*SMBO (Sequential Model Based Optimization)*, applicando quindi un'ottimizzazione bayesiana per la scelta di alcuni iper-parametri di una rete neurale, per un problema di classificazione binaria.
Per questo problema è stata utilizzata la libreria di ottimizzazione SMAC3.

1 Osservazioni sul dataset

Il dataset fornito consiste di 100 istanze, con 9 *features* ciascuna e una variabile di target, indicante la classe di appartenenza.
Osservando le classi, si nota che c'è una forte *class imbalance*, poichè 88 osservazioni appartengono alla *"classe 1"* e 12 alla *"classe 2"*.
Nonostante questo, si è comunque deciso di ottimizzare l'*accuracy*.

2 Ottimizzazione degli iper-parametri tramite SMBO

2.1 Step 1

2.1.1 Definizione dello spazio degli iper-parametri

Nel primo step, l'obiettivo è quello di trovare la configurazione ottimale del *learning_rate* e del *momentum* di un *MLPClassifier*; a questo scopo sono stati settati questi parametri nel seguente modo:

- è stato istanziato un *ConfigurationSpace* per gli iper-parametri
- il *learning_rate* è stato fatto variare nel range: *(0.01,1.0)*
- il *momentum* è stato fatto variare nel range: *(0.1,0.9)*
- successivamente, questi iper-parametri sono stati aggiunti al *ConfigurationSpace*

2.1.2 Definizione della funzione da ottimizzare

E' stata innanzitutto definita la funzione da ottimizzare:

- è stato istanziato un *MLPClassifer*, con un *hidden layer* da 4 neuroni ed un altro da 2 neuroni e con un *random seed* pari a *momentum * learning_rate *5000*, in modo da avere dei risultati riproducibili ad ogni iterazione
- è stata applicata una *StratifiedKFold cross-validation* con un numero di *splits* pari a 10 e un seme randomico pari a 12345
- è stata settata l'*accuracy* come *scoring function* della cross-validation
- la funzione ritorna la media delle accuratezze delle 10 cross-validations

2.1.3 Uso delle funzioni di acquisizione

Sono state confrontate due diverse funzioni di acquisizione: *LCB* e *EI*, entrambe con 25 iterazioni e partendo con una configurazione di 5 punti randomici.
Per questo primo step, per entrambe le funzioni di acquisizione, è stato usato il modello surrogato *SMAC4BO*, che usa l'ottimizzazione bayesiana sfruttando un *Gaussian Process*.
Si è notato che con un numero così basso di iterazioni, i modelli surrogati basati su GP funzionano meglio rispetto a quelli basati sull'RF.

2.1.4 Risultati dell' SMBO

Per questo primo step, i risultati sono per lo più identici per entrambi i metodi utilizzati (l'accuratezza, il più delle volte, è stabile all'88%). Capita, a volte, che il grafico del primo step presenti dei miglioramenti nel livello di accuracy; questo è semplicemente dovuto alla scelta randomica dei cinque punti iniziali: se effettivamente si nota un miglioramento, significa che la scelta dei punti iniziali è stata più fortunata.

NB: i grafici sono una rappresentazione cumulativa dei risultati ottenuti ad ogni iterazione

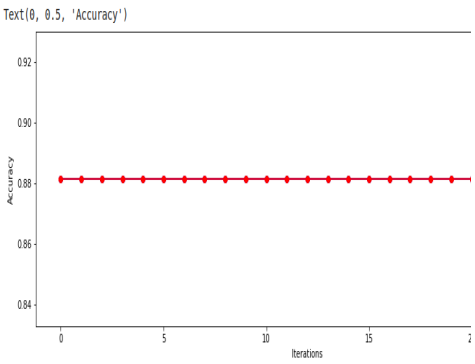

Figure 1: Accuracy per EI ed LCB senza miglioramenti durante le iterazioni

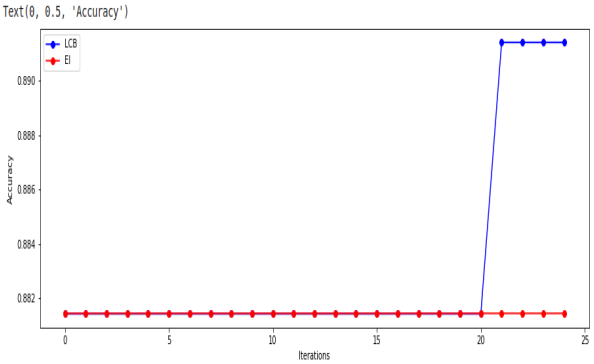

Figure 2: Accuracy per EI ed LCB con miglioramenti durante le iterazioni

2.1.5 Confronto con *GridSearch* e *RandomSearch*

Per questo primo step è stato fatto un confronto anche con *GridSearchCV* e *RandomizedSearchCV*.

1. I parametri passati alla *GridSearch* sono stati i seguenti:

   - si sono provati i seguenti valori del *learning_rate*: [0.01, 0.03, 0.06, 0.09, 0.1]
   - si sono provati i seguenti valori del *momentum*: [0.1, 0.2, 0.4, 0.6, 0.9]

   I risultati ottenuti da questo metodo variano a seconda delle diverse run del codice, ma i valori più frequentemente restituiti sono stati: *learning_rate=0.01* e *momentum=0.2*, con un *best_score* sull'accuracy di 0.89.

2. I parametri passati alla *RandomSearch* sono stati i seguenti:

   - il *learning_rate* è stato scelto casualmente in una distribuzione uniforme (ovvero a media zero e varianza unitaria) nel range (0.01,0.1) e il *momentum* in una distribuzione uniforme nel range (0.1,0.9)

   I risultati ottenuti da quest'altro metodo sono intorno a 0.2 per il *learning_rate* e intorno a 0.3 per il *momentum*, con un *best_score* sull'accuracy di 0.88.

2.1.6 Confronto tra le varie tecniche di ricerca degli iper-parametri

Dai risultati ottenuti, si può notare che tutti e quattro i metodi utilizzati restituiscono valori simili di accuracy, perciò in questo caso è quasi del tutto indifferente la scelta del metodo di ottimizzazione. In generale, però, i metodi basati sull' SMBO restituiscono risultati migliori per quanto riguarda la ricerca degli iper-parametri.

2.2 Step 2

2.2.1 Definizione dello spazio degli iper-parametri

Con un procedimento del tutto analogo a quanto fatto per lo step 1, sono stati settati dei parametri da configurare. Qui, oltre al *learning_rate* e al *momentum*, si vuole ottimizzare anche il numero di neuroni nei due *hidden_layers* della rete; in particolare per entrambi si è scelto un range *(1,5)*.
I range di *learning_rate* e *momentum* sono gli stessi dello step 1.

2.2.2 Definizione dello spazio degli iper-parametri

La funzione da ottimizzare è del tutto simile a quella dello step 1, con l'unica differenza che l'*MLPClassifier* ha due parametri in più da apprendere (ovvero il numero ottimale di neuroni nei due *hidden_layers*).

2.2.3 Uso delle funzioni di acquisizione

Anche in questo caso, si sono usate le stesse funzioni di acquisizione dello step precedente, ma ora le *acquisition functions* hanno 110 iterazioni ciascuna e partono da una configurazione iniziale di 10 punti randomici.
Inoltre qui si è utilizzato il modello surrogato SMAC4HPO, che svolge un'ottimizzazione bayesiana sfruttando il *Random Forest*.
In quest'ultimo caso, usando 110 iterazioni, anche i modelli basati sull'RF funzionano bene, infatti c'è sempre un miglioramento dell'accuracy in almeno uno dei due modelli.

2.2.4 Risultati dell' SMBO

In questo caso, c'è sempre un miglioramento dell'accuracy (almeno di un modello, a volte di entrambi). Anche in questo step, la scelta dei punti iniziali può influire sui miglioramenti dell'accuracy; in particolare, una scelta fortunata dei punti iniziali può portare un modello a migliorare già alle prime iterazioni.
Nel grafico sotto riportato, i due modelli hanno esattamente lo stesso comportamento ed entrambi raggiungono un'accuracy del 90% circa.

NB: anche qui, i grafici sono una rappresentazione cumulativa dei risultati ottenuti ad ogni iterazione

Text(0, 0.5, 'Accuracy')