Voted Perceptron

Niccolò Arati

4th February 2021

1 L' algoritmo

Voted Perceptron è un algoritmo di apprendimento supervisionato, che dato un dataset costruisce due vettori, il vettore delle predizioni V ed il vettore dei pesi C relativi ad esse. Inizia ponendo come primo elemento di V e C zero, e poi ad ogni iterazione calcola y = sgn(v * x), dove x è l' esempio sul quale stiamo facendo appprendimento. Se il risultato è uguale alla classe y di x, incrementa di 1 il peso relativo al k-esimo elemento di V, altrimenti aggiunge un nuovo elemento a V, v[k+1] = v[k] + y * x, ed a C, c[k+1] = 1. In pratica, ogni k-esimo elemento di C tiene conto del numero di predizioni esatte fatte dal k-esimo elemento di V.
Questo procedimento viene ripetuto per ogni esempio disponibile del dataset, e poi si ripete da capo per un numero arbitrario di volte, nel nostro caso 100. Ogni iterazione sugli esempi viene detta epoca.
Si differenzia dal Perceptron di base perchè invece di restituire solo l' elemento finale del vettore V, lo restituisce interamente insieme al vettore C, e la fase di classificazione avviene andando a pesare ogni elemento di V col corrispondente elemento di C : più predizioni avrà fatto bene, più esso conterà nella classificazione.
La predizione per un esempio x avviene sommando per ogni k-esimo elemento in V e in C, C[j] * sgn(v[j] * x), e poi si prende il segno del valore risultante.

2 Descrizione Elaborato

L' elaborato consiste nell' applicare Voted Perceptron ad alcuni dataset, per poi andare a verificare l' accuratezza delle sue previsioni. Si richiede che i dataset abbiano task di classificazione binario con almeno 1000 esempi.
Per ogni esempio la classe viene posta a 1 o -1 con dei criteri che variano a seconda del dataset :

-Per il primo, la classe dell' esempio viene posta ad 1 se la frase contenuta nell' esempio è positiva, mentre viene posta a -1 se essa risulta neutra o negativa.

-Per il secondo dataset, la classe viene posta ad 1 se l' attributo "Occupancy" è pari a 1, mentre se vale 0 la classe viene posta a -1

-Per il terzo dataset, si sceglie la classe in base al valore dell' attributo "Quality" : se è minore o uguale a 5 viene posta a -1, altrimenti ad 1.

-Il quarto dataset è basato sullo stesso problema del terzo dataset, ma per variare viene posta la classe a -1 se l' attributo "Quality" è minore di 5.

Viene quindi creata una lista di tuple di due elementi, nel secondo viene inserita la classe dell' esempio, mentre nel primo viene codificato l' esempio vero e proprio con un numero intero pari alla sua posizione ordinata nel dataset originale, senza tenere conto di eventuali attributi multipli.
Quindi, si costruiscono due ulteriori liste, una contenente gli esempi positivi e una con gli esempi negativi. Così facendo, quando dobbiamo dividere i nostri esempi per formare il training set ed il data set, riusciamo a mantenere in entrambi lo stesso rapporto tra esempi di classe positiva ed esempi di classe negativa.
Infine, si ripetono per ogni dataset apprendimento e testing per 10 volte, e poi, calcolando la test accuyracy per ogni ripetizione, si visualizzano media e deviazione standard di essa.

3 Documentazione del codice

Nel file Vperceptron.py sono presenti 3 funzioni :

-VotedPerceptron implementa l' algoritmo che stiamo considerando, richiede in ingresso x e T e restituisce v e il vettore dei pesi c. Per gli ingressi, T indica il numero di epoche desiderate per l' apprendimento, mentre x è una lista di tuple composte da due elementi, il primo è la codifica dell' esempio, il secondo è la sua classe.

-Prediction implementa la funzione per la classificazione, richiede in ingresso i vettori v e c derivati dall' apprendimento, e una tupla x su cui vogliamo eseguire il test. Della tupla x considera solo il primo elemento corrispondente alla codifica dell' esempio considerato, e restituisce in uscita la classe predetta.

-accuracy è la funzione che calcola la test accuracy, prende in ingresso una lista di tuple su cui abbiamo applicato Prediction, e una lsita contenente i risultati della predizione. Confronta il secondo elemento di ciascuna tupla con la corrispondente predizione.

Nel file main.py invece, per ogni database vengono fatte le seguenti operazioni :

1) Viene letto il database.

2) Vengono create due liste, una contenente la codifica dei vari esempi, e l' altra contenente la classe dell' esempio corrispondente, e poi si definiscono anche le due liste che dividono gli esempi positivi dai negativi.

3) Si genera il training set, mescolando in modo casuale gli elementi delle due liste e poi prendendone l' 80% da ciascuna, ed il test set con gli elementi rimanenti. LE due liste corrispondenti sono formate da tuple, contenenti codifica e classe dell' esempio.

4) Viene quindi fatto l' apprendimento e la classificazione, e la test accuracy viene salvata in una lista.

5) I punti 3 e 4 vengono ripetuti 10 volte per ogni database, e alla fine si procede col calcolo della media e della deviazione standard della lista contenente tutte le test accuracy.

4 Risultati e Conclusioni

Abbiamo ottenuto come risultati una media della test accuracy di 0.7467406062084461, con deviazione standard di 0.15422915567108486.
Questo significa che il nostro modello classifica correttamente nel 75% dei casi, ciò potrebbe essere migliorabile addestrando con più di 100 epoche, ma dal teorema di Block Novikoff sappiamo che, se i dataset su cui operiamo non sono linearmente separabili, allora l' algoritmo tende ad oscillare attorno ad alcuni valori, ma non convergerà mai verso una predizione perfetta.
La non linearità dei dataset potrebbe essere dovuta al come vengono codificate classi ed esempi, infatti se prendiamo il terzo ed il quarto dataset, osserviamo che il terzzo prensenta una media della precisione di circa il 55%, mentre cambiando modo di codificare la classe come nel quarto dataset otteniamo circa il 95%, ed entrambi i dataset sono relativi allo stesso dataset.
Sembrerebbe che in qualche modo, per come abbiamo codificato esempi e classi, l' algoritmo tenda a classificare seguendo la classe che presenta il maggior numero di esempi.