Freund-Schapire Voted-Perceptron

Implementazione e test dell'algoritmo Voted-Perceptron

Marco Di Rienzo

07/2020

Indice

1 Sommario

Implementazione in Python3 e test dell'algoritmo Voted-Perceptron con classificazione Kernel-based (polinomiale) come descritto in Freund - Schapire 1999.
I test eseguiti includono vettori di previsioni creati e accuratezza in base alla frazione del training set studiata dall'algoritmo e dal grado dell'espansione polinomiale usato.
L'algoritmo è stato testato sul dataset Zalando Fashion-MNIST.

2 Algoritmo

Il dataset è formato da una lista di esempi X_{train} e una lista di labels corrispondenti Y_{train}.
Y_{train} contiene elementi appartenenti a un dominio D, per MNIST il dominio sono i numeri da 0 a 9, $D = \{0, 1, ..., 9\}$. Siccome il perceptron è un classificatore binario ognuna delle classi nel dominio delle labels va trasformata in un problema binario, avremo quindi n problemi binari con n numero di elementi in D.
Il problema relativo ad ogni classe ℓ è formulato come segue:

- For $i = 1, ..., len(Y_{train})$:
  - If $Y_{train}[i] = \ell$: $Y_{train_\ell}[i] = 1$
  - else: $Y_{train_\ell}[i] = -1$

Quindi Y_{train_ℓ} sarà una lista che indica quando un esempio deve essere mappato nella classe ℓ e quando no.
Ora per ogni ℓ si esegue il training del perceptron con l'algoritmo di Figura 1 (funzione kernel $K(\mathbf{x}, \mathbf{y}) = (1 + \mathbf{x} \cdot \mathbf{y})^d$ con d grado di espansione).
A questo punto quando arriva un input da valutare si computa uno score s_ℓ per ogni classe ℓ usando i rispettivi vettori generati dai training con un metodo a scelta (nel progetto sono stati utilizzati *last (unnormalized)*, *vote*, *average (unnormalized)* e *random (unnormalized)* descritti in *Freund - Schapire 1999*), i vettori della classe che ha ottenuto lo score più alto sono dati in input alla funzione Prediction di Figura 2, che ritornerà 1 se la votazione ritorna un numero positivo o -1 altrimenti:

3 Prove

Sono state svolte delle prove di training e di test sul dataset Fashion-MNIST.
Il training è stato svolto sul 20% della grandezza totale del dataset e per un massimo di 4 epoche. Lo stesso è stato ripetuto con kernel di grado $d = 1$ e $d = 2$.
Parte dei risultati ottenuti è riportata in Tabella 1. La riga Mistakes indica il numero di errori (cioè vettori di predizione generati, aggiornamenti dell'iperpiano) commessi dall'algoritmo di Training di Figura 1 entro l'epoca sopra riportata.
Le righe rimanenti indicano il numero di errori di previsione in percentuale commessi dall'algoritmo di Prediction di Figura 2 utilizzando l'esperienza accumulata nell'epoca sopra riportata, e con lo *score* delle classi calcolato con il metodo riportato a sinistra.
Gli errori commessi nei test di predizione sono stati tracciati nei grafici di Figura 3 che mostrano l'andamento della percentuale di errori (asse y) rispetto alle epoche di training (asse x).

Training:
Input:

- a labeled training set: $[(\mathbf{x}_1, y_1), ..., (\mathbf{x}_n, y_n)]$ with $\mathbf{x}_i \in X_{train}$ and $y_i \in Y_{train_\ell}$
- Number of epochs T

Output:

- a list of mistaken examples and weights:
  $[(u_1, c_1), ..., (u_k, c_k)]$
  - Initialize: $k = 0, \mathbf{v_1} = 0, c_1 = 0$
  - Repeat T times:
    - For $i = 1, ..., n$:
      - Compute prediction:
        $\mathbf{v}_k \cdot \mathbf{x}_i = \sum_{j=1}^{k} y_{u_j} K(\mathbf{x}_{u_j}, \mathbf{x}_i)$
        $\hat{y} = sign(\mathbf{v}_k \cdot \mathbf{x}_i)$
    - If $\hat{y} = y_i$ then $c_k = c_k + 1$
      else
      - $u_{k+1} = i$;
      - $c_{k+i} = 1$;
      - $k = k + 1$;

Figure 1: Voted perceptron training

Prediction:
Input:

- list of mistaken examples and weights:
  $[(u_1, c_1), ..., (u_k, c_k)]_\ell$
- an input $\mathbf{x}$ to predict

Output:

- a prediction $\{1, -1\}$ on ℓ being the label of $\mathbf{x}$
  - $s = \sum_{i=1}^{k} c_i \cdot sign(\mathbf{v}_i \cdot \mathbf{x})$
    with $\mathbf{v}_i \cdot \mathbf{x} = \sum_{j=1}^{i} y_{u_j} K(\mathbf{x}_{u_j}, \mathbf{x})$
  - $\hat{y} = sign(s)$

Figure 2: Voted perceptron prediction

Table 1: Risultati degli esperimenti su dataset Fashion-MNIST 10-classi. Mistakes indica il numero di errori commessi durante il training. Le altre righe la percentuale di errori commessi durante il test di predizione con il metodo indicato.

	T					
	0.1	0.5	1	2	3	4
				d=1		
Vote	26.5	23.3	20.9	19.7	19.1	19.1
Avg.	26.4	23.2	20.4	19.5	18.9	19.2
Last	34.3	34.3	28.9	29.5	27.2	26.5
Rand.	46.3	34.4	21.5	27.7	23.1	22.2
Mistakes	1045	4006	7381	13347	18971	24437
				d=2		
Vote	29.4	22.9	19.9	18.4	17.8	17.6
Avg.	29.3	22.7	19.8	18.4	17.8	17.5
Last	41.9	24.8	24.7	22.6	25.7	24.9
Rand.	64	30.6	22.5	20.2	26.9	22.7
Mistakes	934	3532	6451	11233	15586	19616

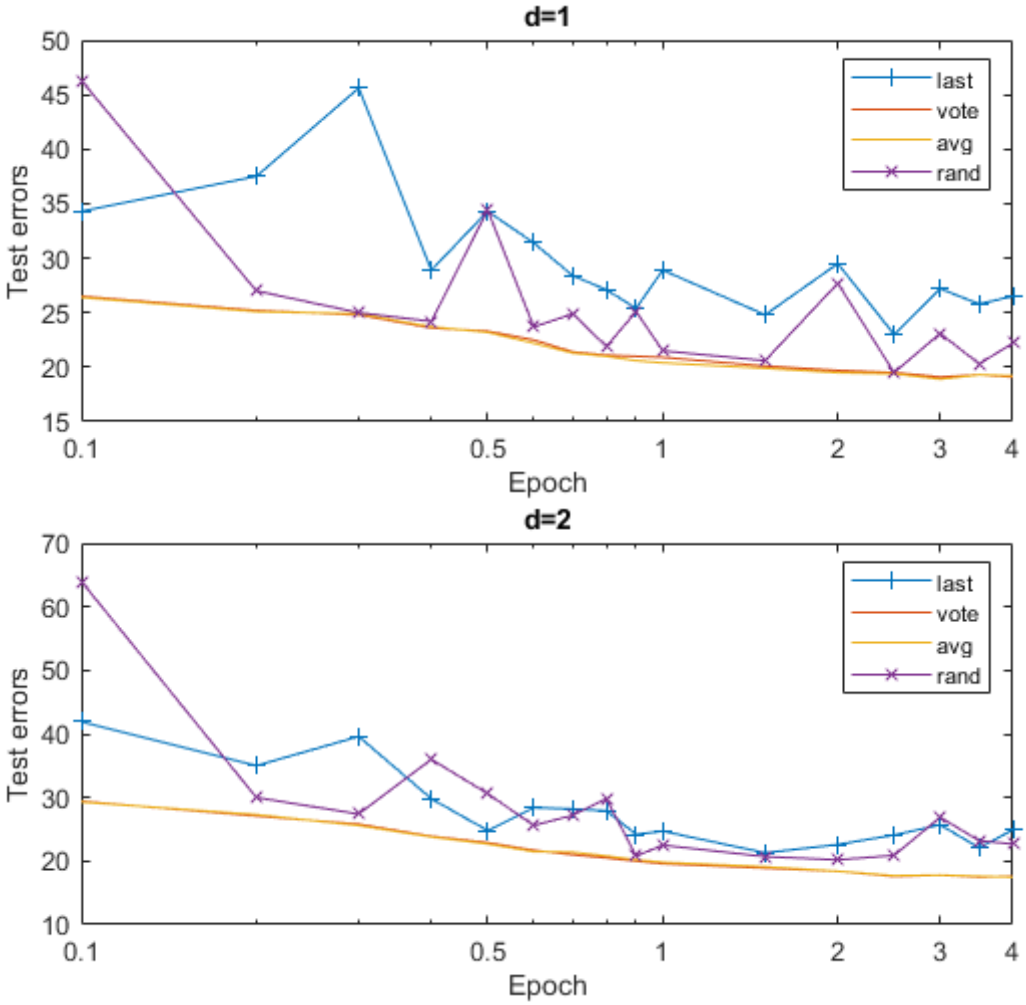

Figure 3: Curve di apprendimento con espansione di grado d

4 Programma

Il programma e la guida di utilizzo sono visionabili al seguente indirizzo https://github.com/marcodiri/voted_perceptron