# Dual Perceptron (dual form)

Lorenzo Ricci

Università degli studi di Firenze
Corso di Laurea in Ingegneria Informatica
Intelligenza Artificiale

A.S. 22/23

# Sommario

**1 Introduzione algoritmo** **3**

**2 Implementazione** **4**

# 1 Introduzione algoritmo

Il primo algoritmo iterativo per l'apprendimento di classificazioni lineari è la procedura proposta da Frank Rosenblatt nel 1956 per il perceptron. E' una procedura *on-line* e *mistake-driven*, che inizia con un vettore peso iniziale **w0** (solitamente inizializzato tutto a 0, **w0=0**) e si adatta ogni volta che un punto, che sta venendo addestrato, viene malclassificato dai pesi attuali. L'algoritmo aggiorna il vettore peso e il bias direttamente. Inoltre questa procedura ha garantita la convergenza dall'esistenza di un iperpiano che classifica correttamente i punti su cui lo si sta facendo addestrare, e in questo caso si dice che i dati sono *linearmente separabili.* Quindi, viceversa, se non esiste un iperpiano i dati si dicono non separabili. Si definisce *margine funzionale di un esempio* ($\mathbf{x}_i$,$y_i$) con rispetto all'iperpiano ($\mathbf{w}$,b), la quantità:

$$\gamma_i = y_i(\langle x_i,\, x_j\rangle + b)$$

e si nota che se $\gamma > 0$ implica una corretta classificazione di ($\mathbf{x}_i$,$y_i$). L'algoritmo Perceptron lavora quindi, aggiungendo esempi di addestramento positivi classificati in modo errato o sottraendo quelli negarivi classificati in modo errato ad un vettore peso scelto all'inizio in modo aribitrario. Senza perdita di generalità, se si assume che il vettore peso iniziale è un vettore zero, e che l'ipotesi finale sarà quella di essere una combinazione lineare dei punti di addestramento, possiamo ridefinire il vettore peso:

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i y_i \mathbf{x}_i$$

dove, dal momento in cui il segno del coefficiente di $\mathbf{x}_i$ è dato dalla classificazione di $y_i$, gli $\alpha_i$ sono volori positivi proporzionali al numero di volte in cui una classificazione errata di $\mathbf{x}_i$ ha causato l'aggiornamento del peso. Punti che hanno causato pochi errori avranno un valore più piccolo di $\alpha_i$, viceversa punti più difficili avranno questo valore più grande. Quindi fissato un set di addestramento S, si può pensare al vettore $\alpha$ come rappresentazione alternativa dell'ipotesi in coordinate diverse o duali.

# 2 Implementazione

Lo scopo del seguente progetto è quello di implementare l'algoritmo nella sua forma duale descritto in (Cristianini & Shawe-Taylor 1999), permettendo l'uso di funzioni kernel al posto del prodotto scalare ⟨xi, xj⟩.

Given training set $S$
$\boldsymbol{\alpha} \leftarrow \mathbf{0}$; $b \leftarrow 0$
$R \leftarrow \max_{1 \leq i \leq \ell} \|\mathbf{x}_i\|$
repeat
  for $i = 1$ to $\ell$
    if $y_i \left( \sum_{j=1}^{\ell} \alpha_j y_j \langle \mathbf{x}_j \cdot \mathbf{x}_i \rangle + b \right) \leq 0$ then
      $\alpha_i \leftarrow \alpha_i + 1$
      $b \leftarrow b + y_i R^2$
    end if
  end for
until no mistakes made within the *for* loop
return $(\boldsymbol{\alpha}, b)$ to define function $h(\mathbf{x})$ of equation (2.1)

Table 2.2: **The Perceptron Algorithm (dual form)**

Come mostrato in (figura 2) l'algoritmo sfrutta un training set S, ovvero dati che servono al Perceptron per addestrare i parametri del classificatore:

- $\alpha$ :