**Capitolo: Riorganizzazione degli Appunti del 25/11**

**1. Funzione Lineare Compressa**

La **funzione lineare compressa** combina una funzione lineare per calcolare un punteggio grezzo e una funzione di attivazione per trasformare il punteggio in un valore interpretabile, solitamente tra 0 e 1.

**1.1 Descrizione:**

* **Fase 1:** Calcolo del punteggio grezzo (z) tramite una funzione lineare:  
  z = (w1 \* x1) + (w2 \* x2) + ... + (wn \* xn) + b  
  Dove:
  + w1, w2, ..., wn sono i pesi associati alle caratteristiche x1, x2, ..., xn.
  + b è il bias.
* **Fase 2:** Compressione del punteggio grezzo usando una funzione di attivazione, come la sigmoid:  
  P = 1 / (1 + e^(-z))

**1.2 Esempio Pratico:**

Prevedere se un'email è spam:

* Numero di parole: x1 = 50
* Numero di link: x2 = 3
* Presenza della parola "gratis": x3 = 1

Con pesi w1 = 0.1, w2 = 1.2, w3 = 2.0 e bias b = -4:  
z = (0.1 \* 50) + (1.2 \* 3) + (2.0 \* 1) - 4 = 6.6

Applichiamo la funzione sigmoid:  
P(Spam) = 1 / (1 + e^(-6.6)) ≈ 0.9986

La probabilità è del 99.86%, quindi l'email è classificata come spam.

**2. Regressione Logistica**

La **regressione logistica** è un algoritmo di classificazione binaria che trasforma un punteggio grezzo in una probabilità usando la funzione sigmoid. Se la probabilità supera una soglia (tipicamente 0.5), assegna l'input alla classe positiva.

**2.1 Esempio Pratico:**

Predire se un cliente acquisterà un prodotto:

* Età: x1 = 30
* Reddito annuo (in migliaia): x2 = 50

Con pesi w1 = 0.8, w2 = 1.5 e bias b = -60:  
z = (0.8 \* 30) + (1.5 \* 50) - 60 = 75

Applichiamo la funzione sigmoid:  
P(Acquisto) = 1 / (1 + e^(-75)) ≈ 1

Il cliente è classificato come molto probabile acquirente.

**3. Discesa del Gradiente**

La **discesa del gradiente** è un algoritmo per ottimizzare una funzione obiettivo calcolando il gradiente rispetto ai parametri del modello e aggiornandoli muovendosi nella direzione opposta al gradiente.

**3.1 Esempio Pratico:**

Consideriamo una funzione obiettivo J(w) = (w - 2)^2.  
Il gradiente è: dJ/dw = 2 \* (w - 2).

Iniziamo con w = 0 e un tasso di apprendimento η = 0.1:

* Iterazione 1: w = w - η \* dJ/dw = 0 - 0.1 \* 2 \* (0 - 2) = 0.4
* Iterazione 2: w = 0.4 - 0.1 \* 2 \* (0.4 - 2) = 0.72

Ripetendo questo processo, w converge a 2, il minimo della funzione.

**4. Discesa del Gradiente Stocastica (SGD)**

La **discesa del gradiente stocastica (SGD)** aggiorna i parametri usando un solo esempio alla volta, rendendo l'algoritmo più veloce ma meno stabile rispetto alla discesa del gradiente standard.

**4.1 Esempio Pratico:**

Consideriamo un dataset con due esempi:

* x1 = 1, y1 = 2
* x2 = 2, y2 = 4

La funzione obiettivo è: J(w) = (y - w \* x)^2.  
Iniziamo con w = 0 e η = 0.1:

* Per x1: w = 0 - 0.1 \* dJ/dw = 0.4
* Per x2: w = 0.4 - 0.1 \* dJ/dw = 0.64

L'algoritmo aggiorna i parametri ad ogni esempio, accelerando l'apprendimento su dataset grandi.

**5. Differenze tra Batch Gradient Descent e SGD**

* **Batch Gradient Descent:** Calcola il gradiente su tutto il dataset. È preciso ma lento.
* **SGD:** Calcola il gradiente su un solo esempio alla volta. È veloce ma meno stabile.
* **Mini-Batch Gradient Descent:** Compromesso tra i due, usa piccoli gruppi di esempi per ogni aggiornamento.

**Conclusione**

Gli argomenti trattati coprono tecniche fondamentali per la classificazione e l'ottimizzazione nei modelli di machine learning. La comprensione delle funzioni lineari, della regressione logistica e dei metodi di discesa del gradiente è essenziale per sviluppare modelli efficienti e accurati.