# **03- PARAMETER LEARNING: GRADIENT DESCENT**

# Gradient descent

È un algoritmo generale che si applica nel ML. Lo si usa per **minimizzare la funzione di costo**.

Si parte con qualche valore di θ0, θ1 che sia zero o dei valori ragionati e si comincia a cambiarli facendo in modo che J(θ0, θ1) raggiunga un minimo.

Non sappiamo se J abbia un solo minimo o diversi minimi locali.

Chart, surface chart

Description automatically generated

Per ogni passo vado a scegliere la direzione in cui si ha la pendenza più ripida.

*Una caratteristica sgradita del gradient descent è che spostando di poco la configurazione di inizializzazione, la discesa potrebbe andare verso un minimo locale e non in un minimo globale.*

Chart, surface chart

Description automatically generated

*Devo dunque calcolare il punto in cui la pendenza è più ripida -> calcolare la derivata***.**

#### Text Description automatically generated with medium confidenceFormula del gradient descent algorithm:

Quindi per determinare il valore del parametro θj prendo il valore precedente a cui sottraggo la discesa calcolata in base alla derivata parziale rispetto allo stesso parametro della funzione di costo, moltiplicata per **α** che è il learning rate. Il learning rate determina la dimensione di ogni step (quanto velocemente scenderò)

Si parla di una simultaneous update in quanto si aggiornano contemporaneamente entrambi i parametri

**A picture containing table

Description automatically generated**

**:=** è la Simultaneous update implementation dunque aggiornamento simultaneo dei θ .

*Immaginiamo di avere un caso più semplice con il solo parametro θ1*

Line chart

Description automatically generatedDiagram

Description automatically generated with medium confidenceChart, line chart

Description automatically generatedA picture containing chart

Description automatically generated

Dunque:

* La macchina deve trovare θ0 e θ1
* α definito dal programmatore → dunque è un ulteriore iperparametro in quanto anche lui è definito dal programmatore come:
  + modello
  + scelta lineare
  + funzione di costo
  + α
* se la funzione di costo ha più minimi locali, può essere problematico in quanto potrei fermarmi in un minimo locale

L’algoritmo di gradient descent con il **learning rate fisso** va in automatico a modulare il passo facendo passi sempre più piccoli quando va a convergenza perché il valore della derivata diminuisce.

* **Il learning rate non viene cambiato** durante l’addestramento di una rete neurale, il passo diventa più piccolo perché man mano che si va iverso il minimo. diminuisce la derivata

## **Modello di regressione lineare usato con l’algoritno di gradient descent che abbiamo utilizzato per ottimizzare il problema. → Training linear regression**

Diagram

Description automatically generatedModello della regressione lineare e l’algoritmo di gradient descent

Diagram

Description automatically generatedOra bisogna calcolare la derivata della funzione di costo. La scriviamo per entrambi i parametri

La differenza sta che per θ1 c’è anche la derivata composta e quindi una x in più.

Diagram, text, schematic

Description automatically generated with medium confidenceL’algoritmo diventa quindi:

Questo algoritmo viene chiamato **batch gradient descent** perché è un algoritmo per passi e ogni passo usa tutti i training examples 🡪 molto lento ma meno costoso

*La funzione di costo per la linear regression è caratterizzata per come è stata scritta nell’avere un solo minimo locale 🡪* ***funziona convessa → ci si può avvicinare usando il gradient descent algorithm.***

Chart

Description automatically generated

*L’obiettivo in generale è quello di scegliere delle funzioni di costo che permettano di avere un unico minimo globale.*

# **Multivariate linear regression**

Ci sono più features/caratteristiche.

Notazione:

* m: numero di training examples 🡪 numero di righe
* Table

  Description automatically generated with medium confidencen: numero di features/caratteristiche 🡪 numero di colonne
* x(i): input features dell’i-esimo training example 🡪 i-esima riga
* **xj(i):** valore della feature j   
  dell’i-esimo training example 🡪 valore singolo della matrice
* **y:** valore reale da predire

Text

Description automatically generated

L’ipotesi si può scrivere anche in forma matriciale ottenendo una **visione vettorizzata**.

Diagram

Description automatically generatedChart, scatter chart

Description automatically generated

Graphical user interface, text, application

Description automatically generated

→ per ogni θ

## **Diagram Description automatically generatedFeature scaling**

Nel caso in cui le features sono su scale differenti, siccome vengono semplicemente sommate, si rischia di avere dei passi più grandi del dovuto.

Chart, radar chart

Description automatically generatedL’ambizione è avere ogni feature in un range **[-1, 1]**. Questo si ottiene dividendo la feature per il valore massimo presente all’interno dell’interno training set.

### **Mean normalization**

Si va a sostituire al valore della feature in modo tale da avere un media vicina allo zero:

**(xi - μi) / si** dove:

* μi è il valore medio di xi per l’intero training set
* si è il range over max-min sempre sull’interno training set

Immagine che contiene Carattere, numero, linea, simbolo

Descrizione generata automaticamente

* si usa questo valore nella colonna

**Debugging gradient descent**

Come si fa a capire che sta funzionando correttamente?

* Disegno grafico

Diagram

Description automatically generatedGraphical user interface

Description automatically generated with low confidence

*Dovremmo vedere la funzione di costo scendere ad ogni iterazione,*

**Summary:**

* Se α troppo grande → funzione di costo potrebbe non decresce ad ogni iterazione o non convergere mai
* Se α sufficientemente piccolo → funzione di costo dovrebbe descrescere ad ogni interazione
* Se α troppo piccolo → la funzione di costo potrebbe impiegare troppo tempo per poter convergere
  + Invece di aspettare troppo tempo, conviene provare con dei valori di learning rate su una scala logaritmica finchè non si trova quello corretto

## **Alternatives to gradient descent**

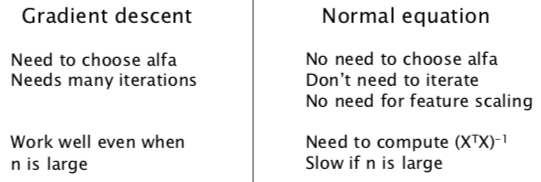
Diagram

Description automatically generated with medium confidenceCi possono essere anche delle alternative al gradient descent, per esempio nel caso specifico della linear regression c’è la tecnica della normal equation o del sistema di equazioni normali che da in un colpo la soluzione:

**θ = (XT X)-1 XT y**

Bisogna scrivere la X come un vettore di esempi

Il problema è che questo tipo di tecnica non funziona per qualsiasi problema, mentre il gradient descent funziona per tutti.



*La normal equation può essere efficace con poche feature.*

# **Optimization algorithms**

Per poter ottimizzare la funzione di costo, mi serve la stessa e le derivate parziali.

Diagram

Description automatically generated with low confidence

## **Polynomial regression**

Immaginiamo di voler predire il costo delle case avendo queste informazioni

Diagram

Description automatically generated

Per tutte le features del mio nuovo dataset mi creo una **nuova feature che è data dalla relazione lineare delle features originali** ottenendo un parametro quadratico. L’algoritmo non si accorge che sia diventato quadratico e quindi continua ad affrontare in maniera lineare ma con delle features quadratiche.

In questo modo si possono gestire problemi complessi.

Ad esempio per la distribuzione vista del dataset potrebbe aver senso avere non una retta ma forme polinomiale quadratiche, cubiche, etc

Chart, line chart

Description automatically generated

Si possono quindi creare delle **hand-crafted features** per modellare al meglio il problema.

Graphical user interface, text

Description automatically generated

In situazioni come queste diventa fondamentale il **feature scaling**;  
La creazione di **hand-crafted features** ci serve per introdurre delle conoscenze del dominio.

*Le reti neurali profonde sono brave ad imparare da sole le features rilevanti, successivamente noi addestriamo le reti neurali fondendo degli esempi e le addestriamo insegnandole come trovare i valori dei parametri θ .*