Per farti capire meglio, ti faccio un esempio concreto di come funziona il processo con la regressione logistica, dove avrai a che fare con **pesi** e **bias** e come il **Gradient Descent** interviene per minimizzare la funzione di costo.

**1. Cosa sono i pesi (weights) e il bias?**

* **Pesi (weights)**: I pesi rappresentano l'importanza di ciascun input per il modello. Ogni peso è associato a una caratteristica (o variabile di input) nel modello. Se un peso è grande, significa che quella variabile ha un grande impatto sulla previsione del modello.

Immagina di avere un modello di regressione logistica che cerca di prevedere se una persona compra un prodotto in base a due fattori: età e reddito. I pesi associati a età e reddito diranno al modello quanto questi fattori influenzano la probabilità di acquisto.

* **Bias**: Il bias è un valore aggiuntivo che viene somato al risultato ponderato degli input. A differenza dei pesi, che dipendono dai dati di input, il bias consente al modello di fare previsioni anche quando gli input sono pari a zero. Può essere visto come uno spostamento che aiuta a spostare la funzione di attivazione, migliorando la capacità predittiva del modello.

In pratica, i **pesi** e il **bias** sono parametri che il modello deve **apprendere** dal dataset durante l'addestramento. Vengono inizializzati a valori casuali e poi aggiornati durante l'ottimizzazione (ad esempio, usando il Gradient Descent).

**2. Creazione di un modello con pesi e bias:**

Immagina di avere un set di dati per una **regressione logistica** che prevede una probabilità di "acquisto" (1 = acquistato, 0 = non acquistato) in base a due variabili: **età** (x₁) e **reddito** (x₂). La funzione di previsione potrebbe essere scritta come:

y^=σ(w1⋅x1+w2⋅x2+b)\hat{y} = \sigma(w\_1 \cdot x\_1 + w\_2 \cdot x\_2 + b)y^​=σ(w1​⋅x1​+w2​⋅x2​+b)

Dove:

* y^\hat{y}y^​ è la probabilità che il cliente acquisti il prodotto (output del modello).
* σ\sigmaσ è la funzione di attivazione **sigmoid** che comprime il risultato tra 0 e 1.
* w1w\_1w1​ e w2w\_2w2​ sono i **pesi** che vengono moltiplicati per i rispettivi input (età e reddito).
* bbb è il **bias**, il termine aggiuntivo che viene sommato all'output ponderato.

**3. Applicazione del Gradient Descent per minimizzare le perdite:**

L'obiettivo del **Gradient Descent** è **minimizzare la funzione di costo** (loss function), che misura quanto le previsioni del modello siano lontane dai valori reali.

La funzione di costo per la regressione logistica è la **log loss** (o cross-entropy), che misura la differenza tra le probabilità previste e quelle effettive. La funzione di costo è definita come:

J(w,b)=−1m∑i=1m[y(i)log⁡(y^(i))+(1−y(i))log⁡(1−y^(i))]J(w, b) = -\frac{1}{m} \sum\_{i=1}^{m} \left[ y^{(i)} \log(\hat{y}^{(i)}) + (1 - y^{(i)}) \log(1 - \hat{y}^{(i)}) \right]J(w,b)=−m1​i=1∑m​[y(i)log(y^​(i))+(1−y(i))log(1−y^​(i))]

Dove:

* mmm è il numero totale di esempi nel dataset.
* y(i)y^{(i)}y(i) è il valore reale (0 o 1).
* y^(i)\hat{y}^{(i)}y^​(i) è la previsione del modello.

**4. Come funziona il Gradient Descent in pratica?**

1. **Inizializzazione**:  
   I pesi e il bias vengono inizializzati a valori casuali (ad esempio, w1=0.5w\_1 = 0.5w1​=0.5, w2=−0.3w\_2 = -0.3w2​=−0.3, b=0b = 0b=0).
2. **Calcolo della previsione**:  
   Calcoliamo la previsione y^\hat{y}y^​ per ogni esempio nel dataset, utilizzando la funzione del modello:  
   y^=σ(w1⋅x1+w2⋅x2+b)\hat{y} = \sigma(w\_1 \cdot x\_1 + w\_2 \cdot x\_2 + b)y^​=σ(w1​⋅x1​+w2​⋅x2​+b).
3. **Calcolo del gradiente**:  
   Per minimizzare la funzione di costo, calcoliamo la derivata della funzione di costo rispetto ai pesi e al bias. Queste derivate sono i **gradienti**, che indicano quanto ogni parametro deve essere cambiato per ridurre l'errore.

Le derivate parziali della funzione di costo rispetto ai pesi e al bias sono:

∂J(w,b)∂w1=1m∑i=1m(y^(i)−y(i))⋅x1(i)\frac{\partial J(w, b)}{\partial w\_1} = \frac{1}{m} \sum\_{i=1}^{m} ( \hat{y}^{(i)} - y^{(i)} ) \cdot x\_1^{(i)}∂w1​∂J(w,b)​=m1​i=1∑m​(y^​(i)−y(i))⋅x1(i)​ ∂J(w,b)∂w2=1m∑i=1m(y^(i)−y(i))⋅x2(i)\frac{\partial J(w, b)}{\partial w\_2} = \frac{1}{m} \sum\_{i=1}^{m} ( \hat{y}^{(i)} - y^{(i)} ) \cdot x\_2^{(i)}∂w2​∂J(w,b)​=m1​i=1∑m​(y^​(i)−y(i))⋅x2(i)​ ∂J(w,b)∂b=1m∑i=1m(y^(i)−y(i))\frac{\partial J(w, b)}{\partial b} = \frac{1}{m} \sum\_{i=1}^{m} ( \hat{y}^{(i)} - y^{(i)} )∂b∂J(w,b)​=m1​i=1∑m​(y^​(i)−y(i))

1. **Aggiornamento dei pesi e bias**:  
   Usiamo il gradiente per aggiornare i pesi e il bias, sottraendo un piccolo passo (detto **learning rate**, α\alphaα) dal gradiente:

w1=w1−α⋅∂J(w,b)∂w1w\_1 = w\_1 - \alpha \cdot \frac{\partial J(w, b)}{\partial w\_1}w1​=w1​−α⋅∂w1​∂J(w,b)​ w2=w2−α⋅∂J(w,b)∂w2w\_2 = w\_2 - \alpha \cdot \frac{\partial J(w, b)}{\partial w\_2}w2​=w2​−α⋅∂w2​∂J(w,b)​ b=b−α⋅∂J(w,b)∂bb = b - \alpha \cdot \frac{\partial J(w, b)}{\partial b}b=b−α⋅∂b∂J(w,b)​

1. **Ripetizione del processo**:  
   Ripetiamo i passaggi 2-4 per un numero predefinito di iterazioni o fino a quando la funzione di costo non si stabilizza (non cambia più significativamente).

**Esempio numerico:**

Supponiamo di avere un dataset con due variabili (età e reddito) e vogliamo prevedere se una persona acquisterà un prodotto:

| **Età (x₁)** | **Reddito (x₂)** | **Acquisto (y)** |
| --- | --- | --- |
| 25 | 40000 | 0 |
| 30 | 50000 | 1 |
| 35 | 60000 | 1 |

Impostiamo:

* Pesi iniziali: w1=0.5w\_1 = 0.5w1​=0.5, w2=−0.3w\_2 = -0.3w2​=−0.3, b=0b = 0b=0
* Learning rate α=0.01\alpha = 0.01α=0.01
* Funzione di costo: **log loss**

L'algoritmo calcolerà le previsioni (y^\hat{y}y^​) e poi applicherà il **Gradient Descent** per aggiornare i pesi e il bias, cercando di ridurre la funzione di costo.

**In sintesi:**

* **Pesi** e **bias** sono parametri che il modello usa per fare previsioni.
* Il **Gradient Descent** è un algoritmo che serve per minimizzare la funzione di costo (che misura l'errore del modello), aggiornando progressivamente i pesi e il bias per ridurre l'errore.

La derivata dl/dz \frac{dl}{dz}dzdl​ misura come cambia la funzione di perdita (lll) rispetto al valore zzz, che è un parametro del modello. In altre parole, essa rappresenta il "gradiente" del modello per quanto riguarda il parametro zzz. Questo gradiente è usato dal **Gradient Descent** per aggiornare i pesi e i bias, cercando di minimizzare la funzione di costo e ottimizzare il modello.