ALGORITMI SUPERVISIONATI

L'idea è quella di creare un modello che, associ a un set di dati di ingresso, un'uscita. Di base è un classificatore, anche se è più corretto chiamarlo regressore.

$$\text{input} \longrightarrow \boxed{f} \longrightarrow \text{output}$$

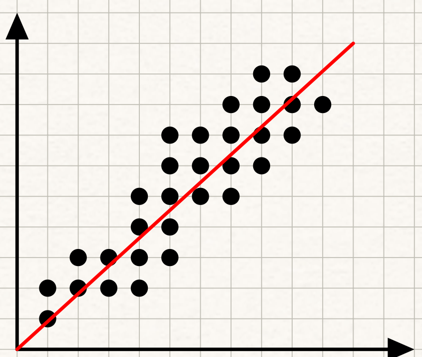
Dove f è la funzione che vogliamo approssimare:

$$f : \mathbb{R}^n \longrightarrow \mathbb{R}$$

Per approssimare $f(\cdot)$ ci basiamo sui dati di training che hanno una classe.

Regressione Lineare

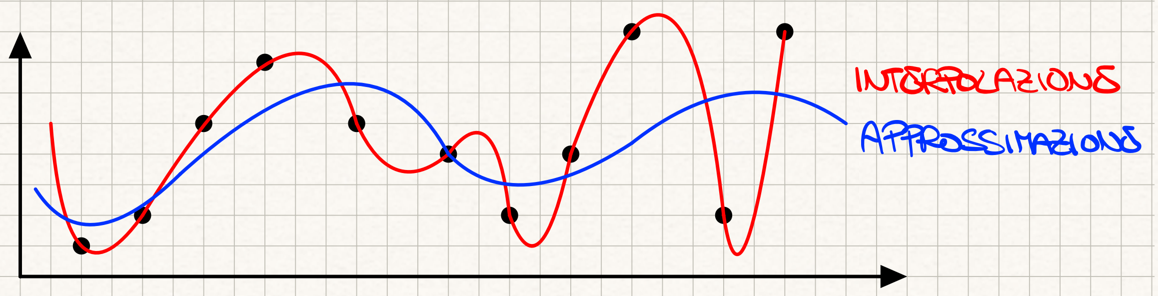
A volte è meglio avere un modello che approssima i dati, piuttosto che avere i dati stessi. Possiamo usare il modello anche per predire la classe di nuovi dati.



Nota: Meglio scegliere sempre il modello più semplice.

INTERPOLAZIONE Vs APPROSSIMAZIONE

Col nostro modello regressivo, puntiamo ad approssimare i punti del nostro dataset, e non a interpolarli perfettamente.



Matematicamente possiamo sempre trovare un polinomio di grado N che, dato un set di punti lo interpola. Non si ha però il controllo dell'andamento del polinomio fra due punti consecutivi:

Approssimare, vuol dire trovare una funzione che si avvicina il più possibile ai punti del campione, senza passare per tutti. Questo modello è sicuramente migliore perché gestibile.

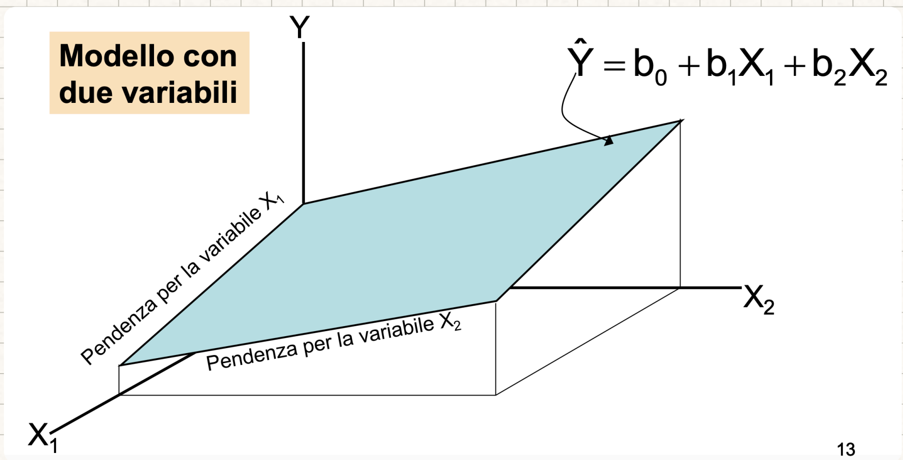
MULTIPLE LINEAR REGRESSION

Si tratta di voler determinare una funzione che meglio approssima il legame (in media) fra le variabili indipendenti in ingresso e la variabile dipendente in uscita.
Nel caso lineare $f(\cdot)$ è una retta:

$$y = a_0 + a_1 X_1 + a_2 X_2 + \dots + a_p X_p .$$

Esempio

Con due variabili e tre coefficienti



Supponiamo di avere un learning set LS, fatto da P tuple:

$$LS = \{x_i, y_i\} : \quad x_i \in \mathbb{R}^N, \ y_i \in \mathbb{R}$$

Costruiamo la matrice X, come segue:

$$X_e = \begin{bmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_P \end{bmatrix} \in \mathbb{R}^{P \times (N+1)}$$

Sia y il vettore delle labels:

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_P \end{bmatrix} \in \mathbb{R}^P$$

La matrice dei pesi W, che rappresenta i coefficienti da dare a f per approssimare il problema sono dati da:

$$W = (X^T X)^{-1} \cdot X^T y \quad \text{con} \quad W \in \mathbb{R}^{N+1}$$

Nota: questo calcolo è computazionalmente pesante!

Dato l'ingresso x_e, possiamo ricavare l'uscita $\hat{y}$:

$$\hat{y} = W \cdot x_e = \sum_{i=\phi}^{N} w_i \, x_i$$

Usando il prodotto scalare. Dove x_e è l'ingresso esteso con una serie di 1.

LOSS FUNCTION

In questo contesto la loss sarà il Mean Square Error:

$$\mathcal{L}^{MSE} = \frac{1}{2P} \sum_{i=\phi}^{P} (\hat{y}_i - y_i)^2$$

1) Cerchiamo di scrivere $\mathcal{L}^{MSE}$ con i vettori e non con la sommatoria:

$$\mathcal{L}^{MSE} = \frac{1}{2P} (\hat{y} - y)^T \cdot (\hat{y} - y)$$

2) Sapendo che $\hat{y} = x_e \cdot W$, sostituiamo:

$$\mathcal{L}^{MSE} = \frac{1}{2P} (x_e W - y)^T (x_e W - y)$$

Guardando le dimensioni, considerando che $x_e \in \mathbb{R}^{P \times (N+1)}$ mentre $W \in \mathbb{R}^{(N+1)}$, abbiamo che $\hat{y} \in \mathbb{R}^{P}$.

Obiettivo: trovare il vettore W che minimizza la loss.

3) Calcoliamo il gradiente della loss: $\nabla \mathcal{L}$, così da applicare un algoritmo di discesa del gradiente.

$$\frac{\partial \mathcal{L}}{\partial W}$$

LEMMA

Per continuare abbiamo bisogno di un lemma che dice:

Presa $f(x) = x^T M x$, dove x è un vettore colonna e M una matrice quadrata, il gradiente $\nabla f(x)$ si calcola:

$$\nabla f(x) = (M + M^T)\, x$$

4) Continuando i calcoli su $\mathcal{L}$:

$$\mathcal{L}^{MSE} = \frac{1}{2P}(x_e W - y)^T (x_e W - y)$$

$$= \frac{1}{2P}\left(W^T x_e^T \cdot x_e W - W^T x_e^T y - y^T x_e W + y^T y\right) =$$

Possiamo sommare

$$= \frac{1}{2P}\left(W^T x_e^T \cdot x_e W - 2 W^T x_e^T y + y^T y\right) =$$

Usando il lemma calcoliamo le derivate parziali:

1) $$\frac{\partial (W^T x_e^T x_e W)}{\partial W} = (x_e^T x_e + x_e x_e^T)\, W = 2\, x_e^T x_e W$$

2) $$\frac{\partial (2 W^T x_e^T y)}{\partial W} = 2\, x_e^T y$$ → Qui non usiamo il lemma

Alla fine avremo:

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{1}{\cancel{2}P}\left(\cancel{2}\, x_e^T x_e W - \cancel{2}\, x_e^T y\right)$$

$$\frac{\partial \mathcal{L}}{\partial W} = \frac{1}{P}\left(x_e^T x_e W - x_e^T y\right)$$

Dato che $\mathcal{L}^{MSE}$ è per sua natura una funzione convessa (è la somma di differenze al quadrato, ovvero somma di parabole), possiamo affermare che il minimo esiste ed è unico, ergo globale. Perciò ci basta risolvere $\nabla \mathcal{L} = \vec{\emptyset}$:

$$\frac{1}{P}\left(x_e^T x_e W - x_e^T y\right) = \emptyset$$

$$x_e^T x_e W - x_e^T y = \emptyset$$

$$x_e^T x_e W = x_e^T y$$

$$W = \left(x_e^T x_e\right)^{-1} x_e^T y$$

PROBLEMI

1) Se la matrice $(x_e^T x_e)^{-1}$ è molto grande potremmo avere dei problemi, per esempio se non entra nella RAM di un singolo calcolatore, non possiamo usare la formula esatta. Si preferisca dunque uno schema iterativo:

Schema iterativo:

```
W = inizialize();
for i=1 : n_Iterazioni :
    W_new = W_old + η ∂J/∂W;
end
```

2) Stiamo inoltre supponendo che $(x_e^T x_e)$ sia invertibile. Se invece è vicina a essere una matrice singolare, non lo è più.
Difatti non sempre $(x_e^T x_e)$ è definita positiva, quindi invertibile. Possiamo però usare un trick, aggiungiamo una matrice identità:

$$z = (x_e^T x_e + \lambda I)$$

Così z è sicuramente invertibile.

Così facendo, anche la loss function cambia:

$$\mathcal{L}' = \mathcal{L} + \lambda w^T I w$$

3) Così il problema diventa che alcuni pesi crescono troppo altri molto meno. Sono numericamente instabili. Il modello ha troppi gradi di libertà.
La soluzione è utilizzare:

$$\|w\|_2^2 = \sum_{i=0}^{N+1} w_i^2$$

Per cercare di ottenere pesi non troppo grandi, perché andiamo a penalizzare quei pesi con magnitudine maggiore.

Gradiente di $\mathcal{L}'$

$$\frac{\partial \mathcal{L}'}{\partial w} = \left(X_e^T X_e - \lambda I\right)^{-1} w - X_e^T y$$

Da cui cercando $\nabla \mathcal{L}' = \vec{0}$ troviamo:

$$w = \left(X_e^T X_e + \lambda I\right)^{-1} X_e^T \, y$$

Attenzione: va settato λ. Più è grande più la loss non fa imparare molto al modello. Ma in compenso diventa una loss più smooth e meno nervosa.