Introduzione a modelli e metodi di regressione

(dal punto di vista del Machine Learning)

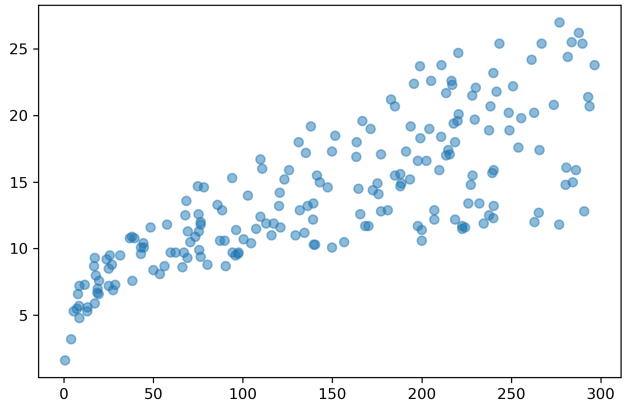
Vincenzo Bonifaci

Esempio: Ritorno da investimenti pubblicitari

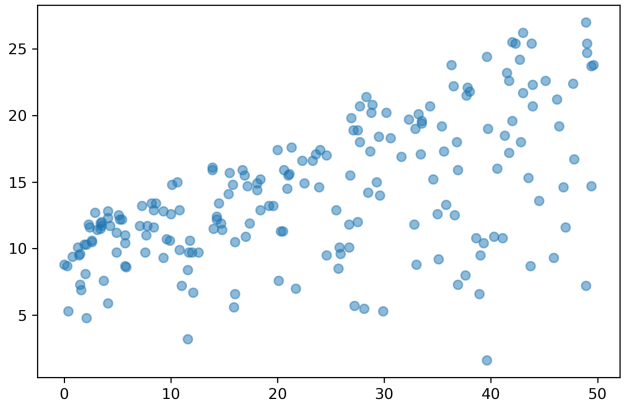
Input: investimenti pubblicitari via TV, radio e giornali in vari mercati (*input*, *predittori*, *feature*, *variabili indipendenti*)
Output: unità di prodotto vendute in quei mercati (*output*, *responso*, *etichetta*, *variabile dipendente*)

	TV	radio	newspaper	sales
0	230.1	37.8	69.2	22.1
1	44.5	39.3	45.1	10.4
2	17.2	45.9	69.3	9.3
3	151.5	41.3	58.5	18.5
4	180.8	10.8	58.4	12.9
...	...	...	...	...

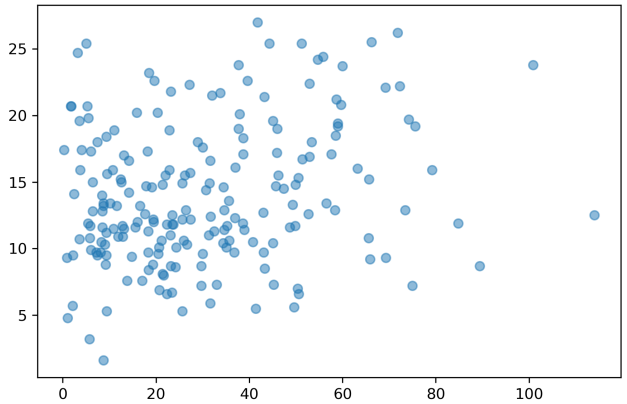
Esempio: Ritorno da investimenti pubblicitari



sales vs. TV



sales vs. radio



sales vs. newspaper

Problemi di predizione: input e output

- Spazio degli input $\mathcal{X}$
  Es.: insieme degli investimenti $\langle$ tv, radio, giornali $\rangle$ $(\mathbb{R}^3_+)$
- Spazio degli output $\mathcal{Y}$
  Es.: insieme delle possibili quantità di prodotto vendute $(\mathbb{R})$

Osservati un certo numero di esempi (x, y), vogliamo trovare una *regola di predizione* (o *ipotesi*)

$$h : \mathcal{X} \to \mathcal{Y}$$

che ricostruisca in maniera accurata la relazione ingresso-uscita

Nei problemi di *regressione* l'output è quantitativo (*numerico*)

Nei problemi di *classificazione* l'output è qualitativo (*categorico*)

Funzioni di costo [loss functions]

Come quantifichiamo l'accuratezza di una regola di predizione $h : \mathcal{X} \to \mathcal{Y}$ su un particolare esempio?

Una *funzione di costo* è una funzione ℓ che prende una regola di predizione h ed un esempio $(x, y) \in \mathcal{X} \times \mathcal{Y}$, e restituisce un reale nonnegativo

$$\ell(h, (x, y)) \in \mathbb{R}_+$$

Esempi di funzioni di costo

- *Quadrato dell'errore*:

$$\ell(h,(x,y)) \stackrel{\text{def}}{=} (h(x)-y)^2$$

- *Funzione costo 0-1*:

$$\ell(h,(x,y)) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{se } h(x)=y \\ 1 & \text{se } h(x)\neq y \end{cases}$$

Rischio atteso

Come quantifichiamo l'accuratezza di una regola di predizione $h : \mathcal{X} \to \mathcal{Y}$ in generale?

Assunzione fondamentale

Gli esempi (x, y) sono generati in modo indipendente da una distribuzione di probabilità (ignota) $\mathcal{D}$ sull'insieme $\mathcal{X} \times \mathcal{Y}$

$\mathcal{D}$ è ignota poiché è proprio la relazione ingresso-uscita che l'algoritmo deve apprendere!

Il *rischio atteso* di una regola di predizione h è

$$L_{\mathcal{D}}(h) \stackrel{\text{def}}{=} \mathbb{E}_{(x,y)\sim\mathcal{D}} \left[\ell(h, (x, y))\right]$$

A parole: il rischio atteso di h è il valore atteso della funzione di costo su h quando gli esempi sono generati dalla distribuzione $\mathcal{D}$

Il problema del machine learning supervisionato

Problema del machine learning supervisionato

Fissata una distribuzione (ignota) $\mathcal{D}$ su $\mathcal{X} \times \mathcal{Y}$, cerca una regola di predizione che minimizzi il rischio atteso:

$$\begin{aligned} &\underset{h}{\text{minimize}}\, L_{\mathcal{D}}(h) \\ =\ &\underset{h}{\text{minimize}}\, \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\ell(h,(x,y))\right] \end{aligned}$$

Il rischio atteso dipende dalla distribuzione ignota $\mathcal{D}$...

Come possiamo minimizzarlo, visto che non conosciamo $\mathcal{D}$?!

Rischio empirico

Non conosciamo $\mathcal{D}$ ma abbiamo degli *esempi* dalla distribuzione $\mathcal{D}$

Il *rischio empirico* di h sugli esempi $S = \{ (x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)}) \}$ è

$$L_S(h) \stackrel{\text{def}}{=} \frac{1}{m} \sum_{i=1}^{m} \ell(h, (x^{(i)}, y^{(i)}))$$

Possiamo usare il rischio empirico come *surrogato* del rischio atteso: se il numero di esempi m è sufficientemente grande, si può sperare che i due valori siano vicini

Il principio ERM

Empirical Risk Minimization (ERM)

Dato un insieme di esempi S (generati da $\mathcal{D}$), cerca una regola di predizione che minimizzi il rischio empirico su S:

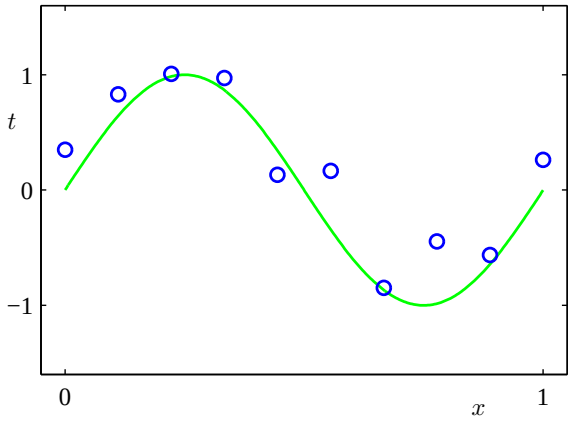
$$\underset{h}{\text{minimize}}\, L_S(h) \left(= \underset{h}{\text{minimize}}\, \frac{1}{m} \sum_{i=1}^{m} \ell(h, (x^{(i)}, y^{(i)})) \right)$$

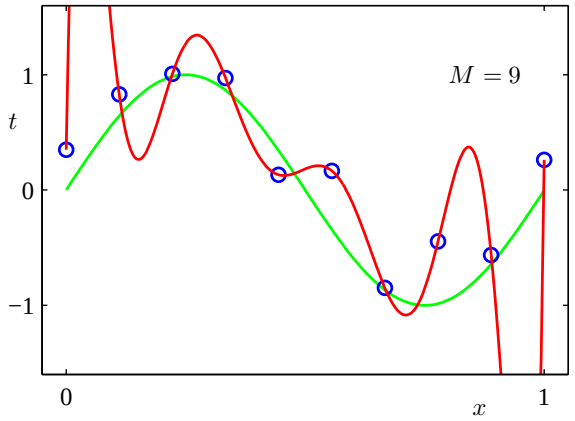
L'insieme S di esempi osservati dal learner è detto *training set*

Applicando l'ERM, il problema del learning supervisionato è rimpiazzato da un problema di ottimizzazione (nello spazio delle regole h)

Il sovradattamento (overfitting)

Sebbene l'ERM sia un principio intuitivo, esso può completamente fallire senza le dovute cautele!





In questo esempio, la regola scelta (la funzione rossa) è *sovradattata* ai dati (*overfitting*):

"Spiega" perfettamente le osservazioni, ma non è un buon modello della distribuzione da cui i dati sono generati (funzione verde + rumore)

Il suo rischio empirico è nullo, ma il suo rischio atteso è alto

ERM con una classe di ipotesi ristretta

Un approccio per ovviare al problema dell'overfitting consiste nel limitare l'insieme delle possibili regole di predizione (ipotesi) h

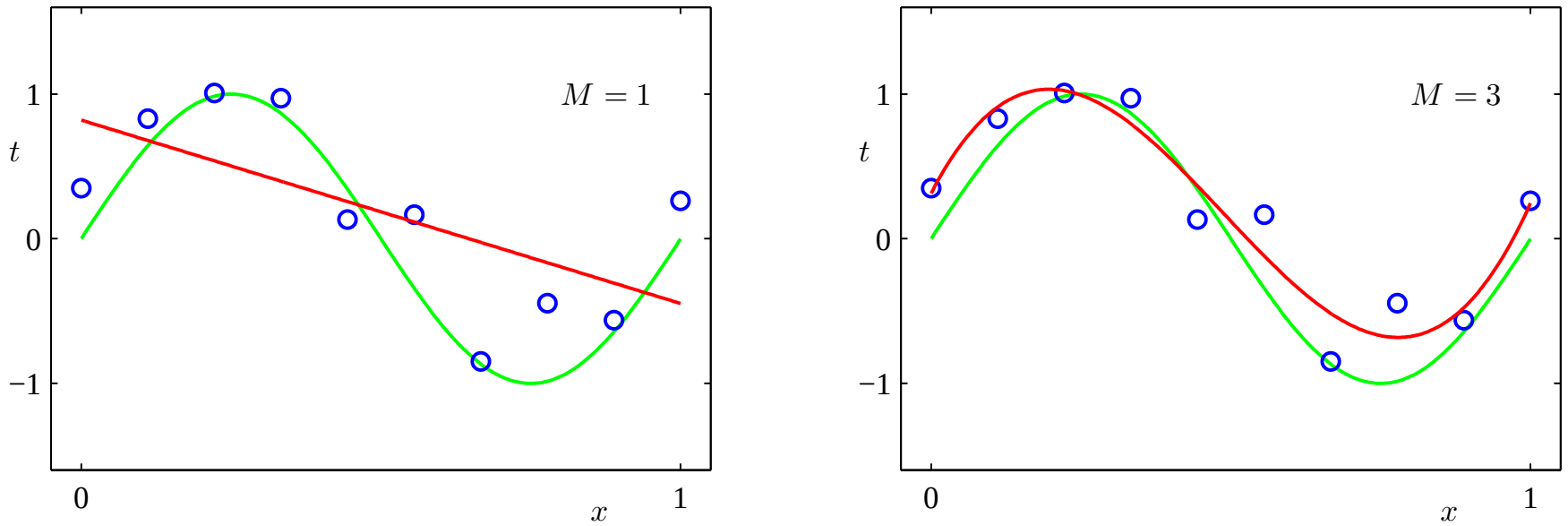
Anziché considerare la classe $\mathcal{Y}^{\mathcal{X}}$ di tutte le funzioni da $\mathcal{X}$ a $\mathcal{Y}$, consideriamo solo una sua sottoclasse $\mathcal{H}$ (*insieme delle ipotesi*)

Possiamo applicare il principio ERM restringendoci alle ipotesi in $\mathcal{H}$:

$$\underset{h \in \mathcal{H}}{\text{minimize}}\, L_S(h)$$

- La classe $\mathcal{H}$ può incorporare la conoscenza pregressa del problema considerato, limitando la *complessità* delle ipotesi
- La classe $\mathcal{H}$ introduce un *pregiudizio (bias) induttivo*: tutte le regole non in $\mathcal{H}$ sono scartate a priori

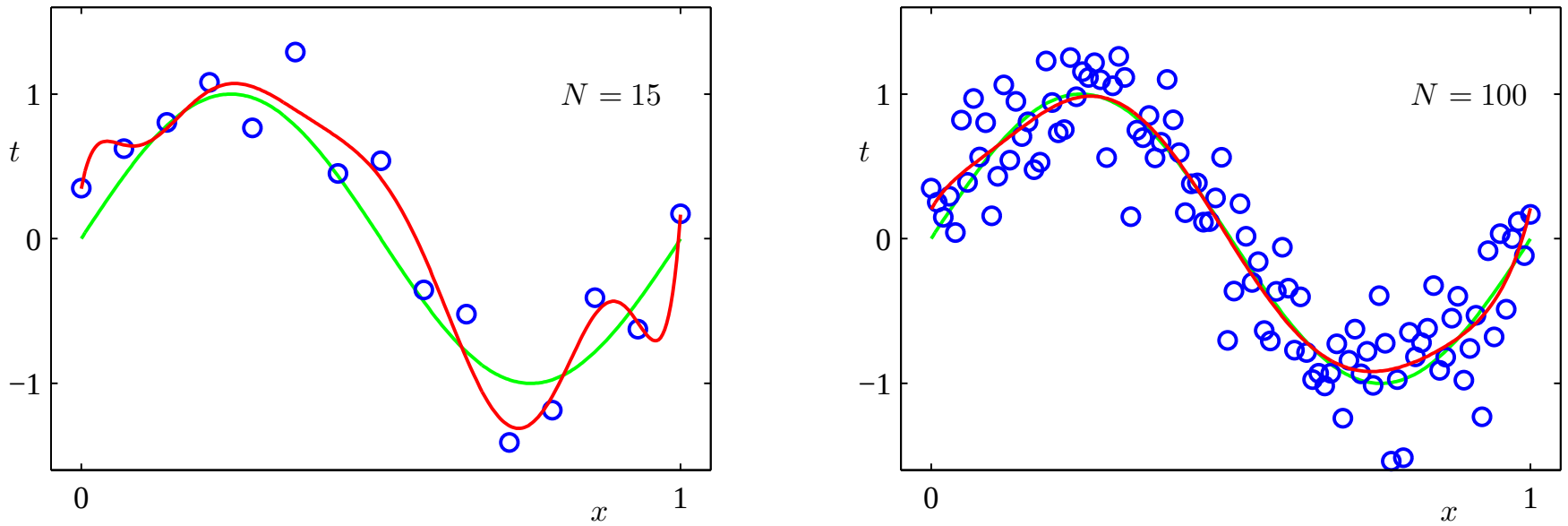
Compromesso bias-varianza



Fitting di un polinomio di grado 1 (sinistra) e di grado 3 (destra)

- Modelli più semplici hanno più bias (possono esibire *underfitting*)

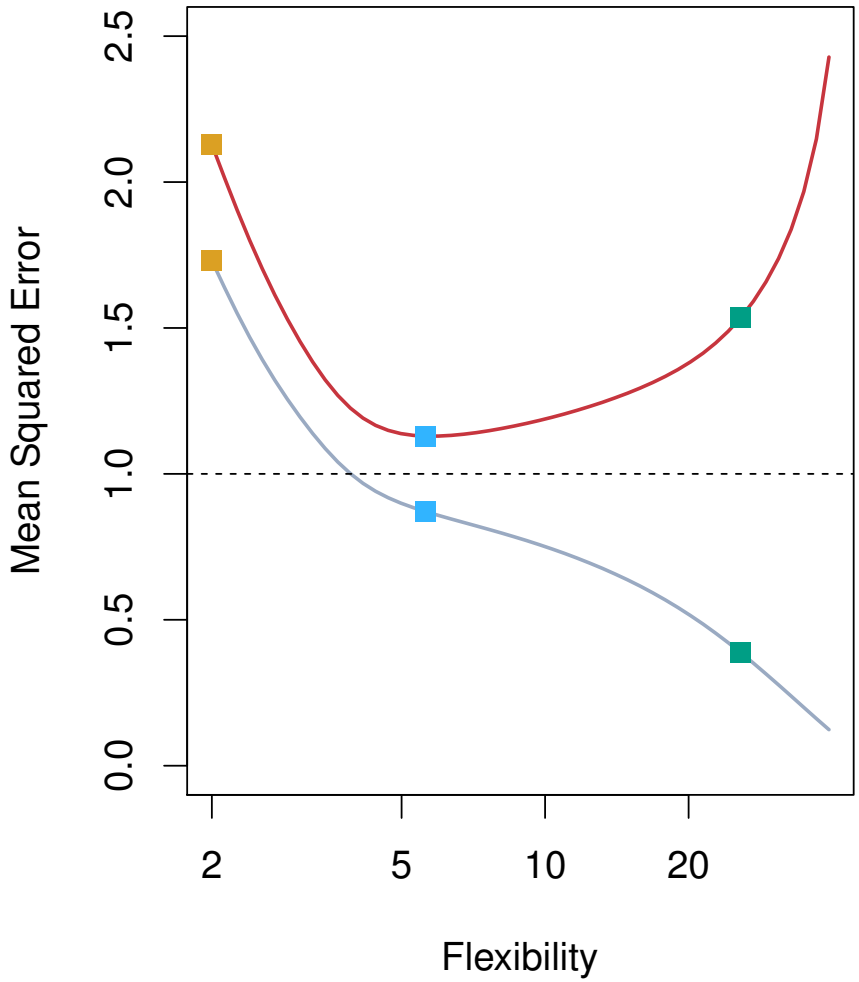
Compromesso bias-varianza



Fitting di un polinomio di grado 9 con 15 esempi (sinistra) e 100 esempi (destra)

- Modelli più complessi hanno più varianza (richiedono più esempi)

Compromesso bias-varianza



- Curva grigia: rischio empirico
- Curva rossa: rischio atteso

Regressione lineare

Nella *regressione lineare*, l'insieme delle ipotesi è l'insieme $\mathcal{H}_{lin}$ delle funzioni lineari (affini) da $\mathcal{X} \equiv \mathbb{R}^d$ a $\mathcal{Y} \equiv \mathbb{R}$:

$$h \in \mathcal{H}_{lin} \Leftrightarrow h(x) = w_0 + w_1 x_1 + \ldots + w_d x_d \qquad (w_0, \ldots, w_d \in \mathbb{R})$$

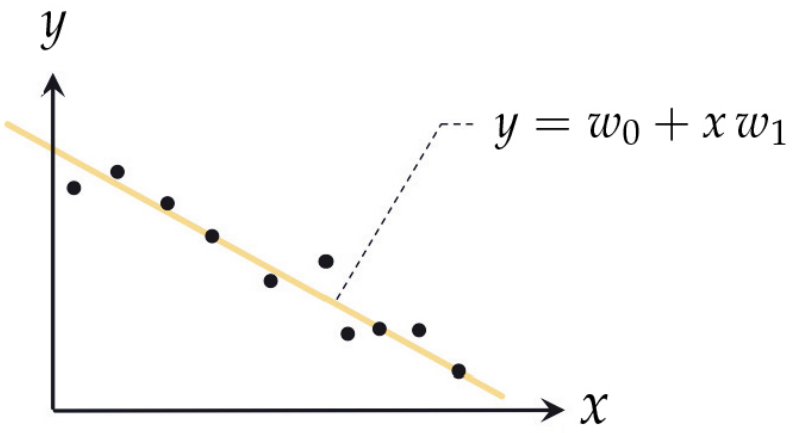
Useremo spesso la convenzione $x_0 \stackrel{\text{def}}{=} 1$, così da poter scrivere $h(x) = w^\top x$

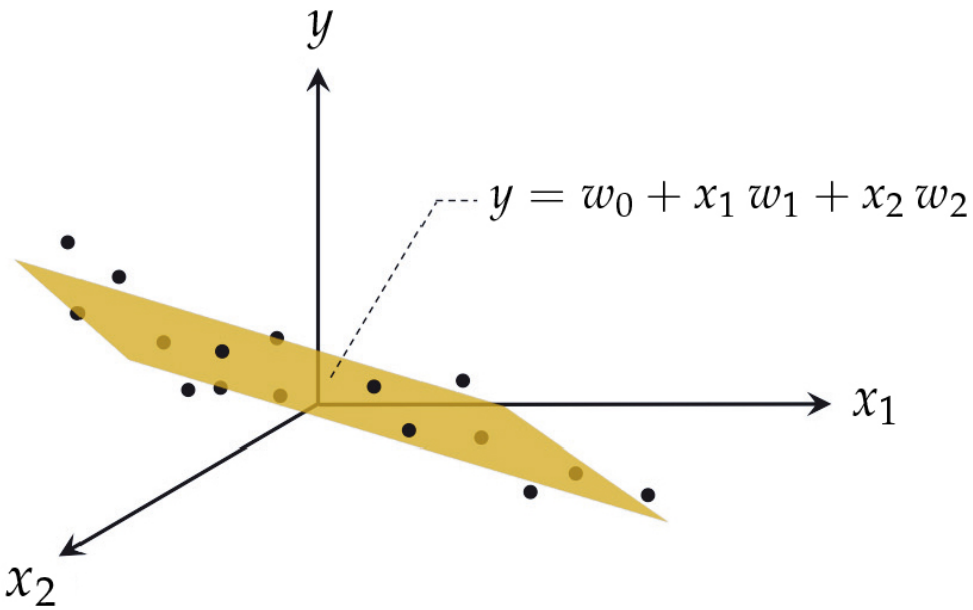
- w_0 è l'*intercetta* (valore previsto dal modello quando x è nullo)
- w_k è il *coefficiente* che esprime la dipendenza di $h(x)$ dalla k-esima componente di x

Una funzione di costo comunemente utilizzata è quella quadratica:

$$\ell(h, (x, y)) = (h(x) - y)^2$$

Regressione lineare



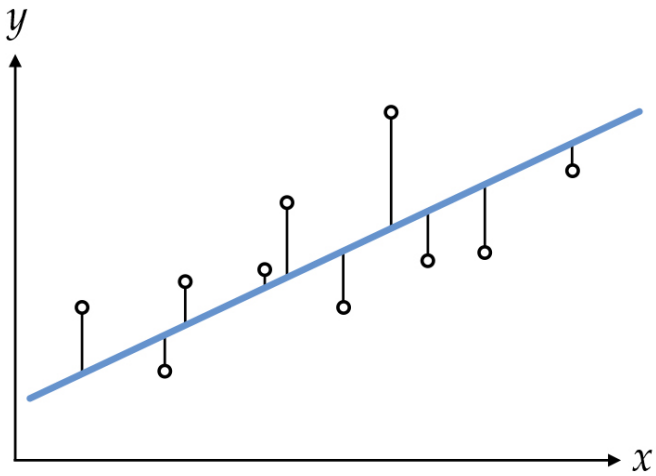


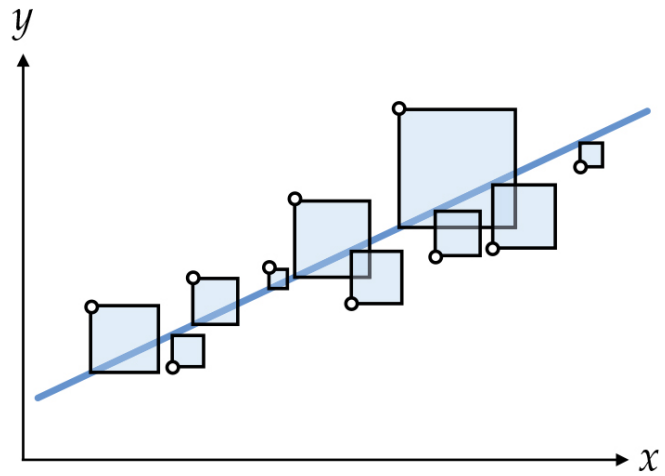
ERM per la regressione lineare

Nella regressione lineare con costo quadratico, il rischio empirico è dato dall'*errore quadratico medio* [*mean squared error*]:

Mean Squared Error (MSE)

$$L_S(h) = \frac{1}{m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 = \frac{1}{m}\|Xw - y\|^2$$





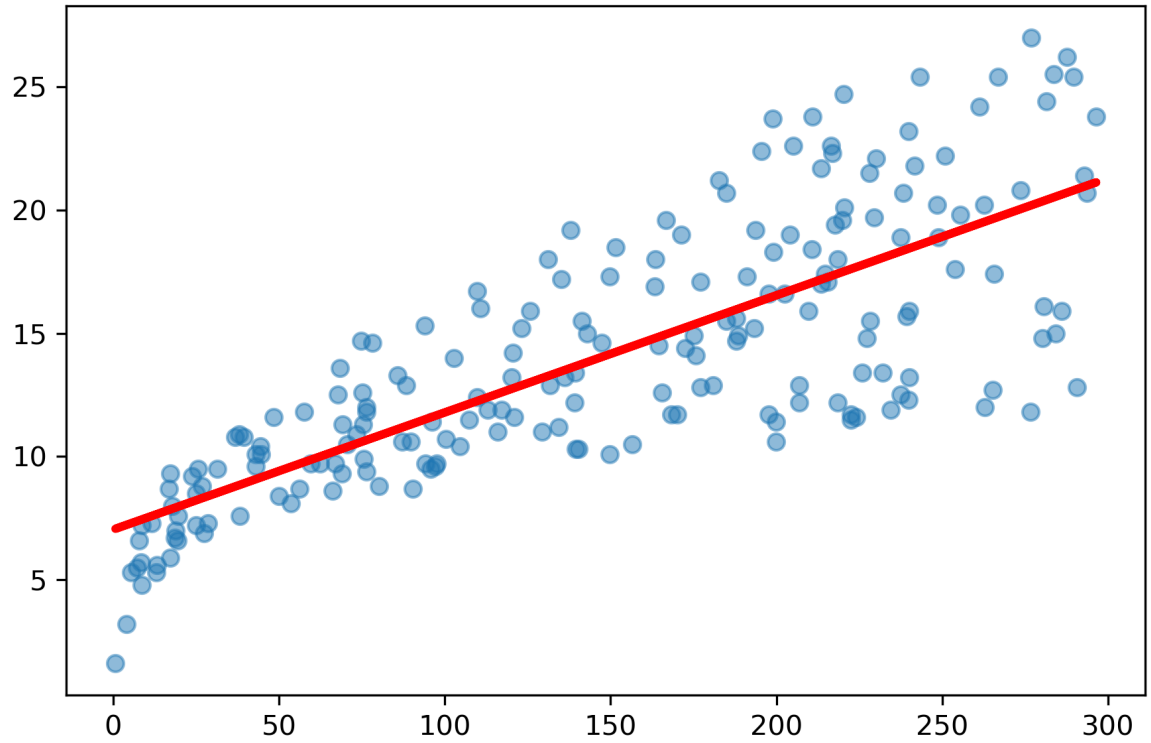
ERM per la regressione lineare

Il minimizzatore del rischio empirico qui è esprimibile in forma chiusa:

$$w^* = \left(\sum_{i=1}^{m} x^{(i)} x^{(i)\top} \right)^{-1} \left(\sum_{i=1}^{m} y^{(i)} x^{(i)} \right) = (X^\top X)^{-1} X^\top y$$

Nella pratica, w^* è calcolata con metodi numerici di fattorizzazione (Singular Value Decomposition – SVD), più stabili rispetto alla formula e che non richiedono l'esistenza dell'inversa

Esempio: regressione di sales su TV



$$\texttt{sales} \approx w_0 + w_1 \cdot \texttt{TV}$$

- Intercetta $w_0 = 7.03 \Rightarrow$ 7030 unità di prodotto vendute senza investimenti
- Coefficiente $w_1 = 0.047 \Rightarrow$ 47 unità di prodotto in più ogni 1000\$ di pubblicità in TV

Come valutare la qualità del fit?

Si può usare il rischio empirico (in questo caso: errore quadratico medio)

Nella regressione lineare, si può anche usare la statistica R^2:

Coefficiente R^2

$$R^2 \stackrel{\text{def}}{=} 1 - \frac{L_S^*}{L_S^0},$$

- $L_S^* = \min_{h \in \mathcal{H}} L_S(h)$ è l'errore quadratico medio della migliore ipotesi lineare
- L_S^0 è l'errore quadratico medio dell'ipotesi costante $h_0(x) = \frac{1}{m} \sum_i y_i$

Come valutare la qualità del fit?

R^2 è un valore tra 0 e 1 che rappresenta la proporzione di variabilità dei dati di output y "spiegata" dal modello

- R^2 coincide con il quadrato del coefficiente di correlazione tra il responso osservato (y) e quello previsto dal modello ($h(x)$)
- Rispetto al rischio empirico, R^2 ha il vantaggio di essere normalizzato
- R^2 è specifico per la funzione costo quadratica

Come valutare la qualità del modello?

Qualità del fit (rischio empirico)
$\neq$
qualità del modello (rischio atteso)

Possiamo stimare il rischio atteso di un'ipotesi h utilizzando un insieme di esempi di test T (*test set*)

Gli esempi in T provengono ancora dalla distribuzione (ignota) $\mathcal{D}$

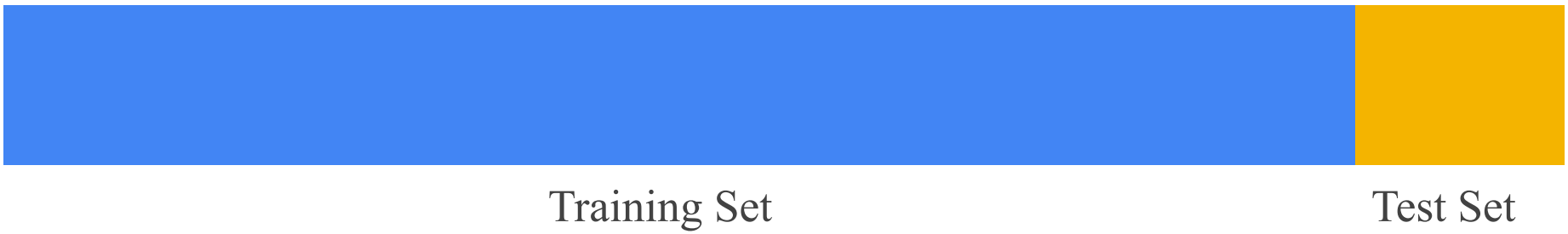
Con sufficienti esempi, il rischio empirico su T sarà una buona stima del rischio atteso:

$$L_T(h) \approx L_{\mathcal{D}}(h)$$

Training set e test set

Nella pratica, avremo un solo insieme di dati a disposizione

Separiamo a caso i dati di esempio a nostra disposizione in due insiemi:



Training set e test set

- Il *training set* S è usato per trovare l'ipotesi h col miglior fit:

$$\min_{h \in \mathcal{H}} L_S(h)$$

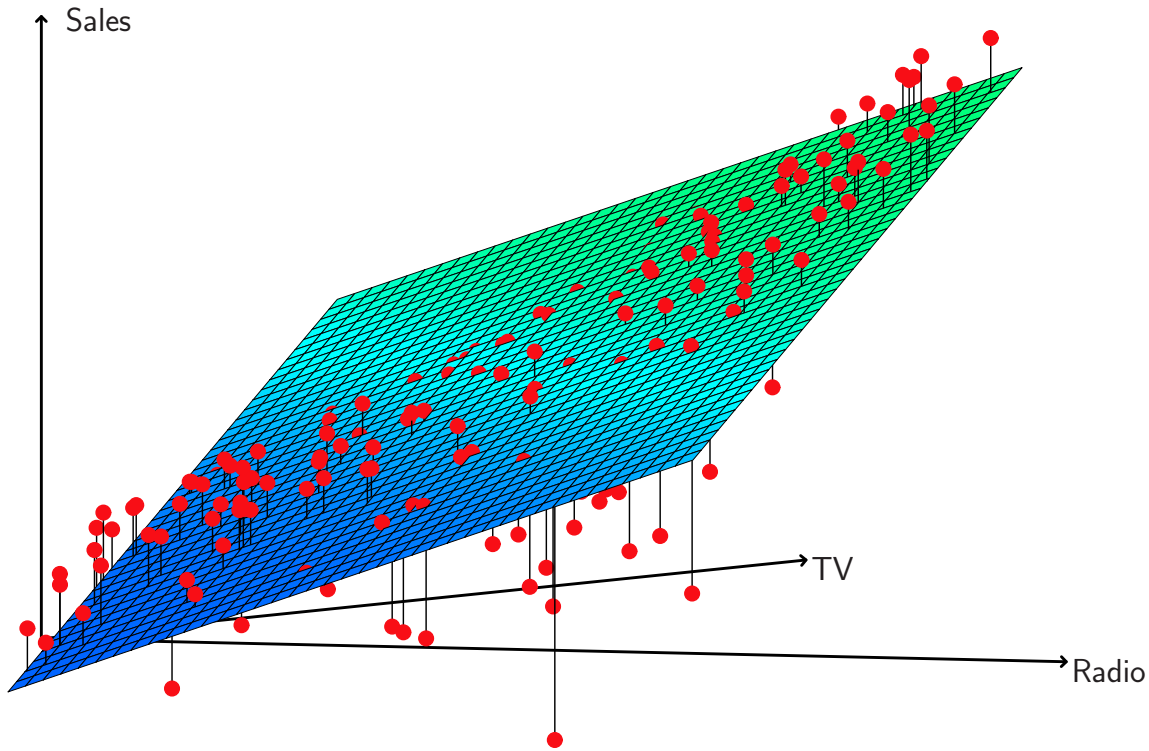
- Il *test set* T è usato per stimare il rischio atteso di h:

$$L_T(h) \approx L_{\mathcal{D}}(h)$$

  - La separazione è necessaria affinché gli esempi usati per stimare $L_{\mathcal{D}}(h)$ siano indipendenti da h
  - La separazione deve essere casuale, affinché S e T seguano la stessa distribuzione
  - Mai usare gli esempi di test per fare il training del modello!

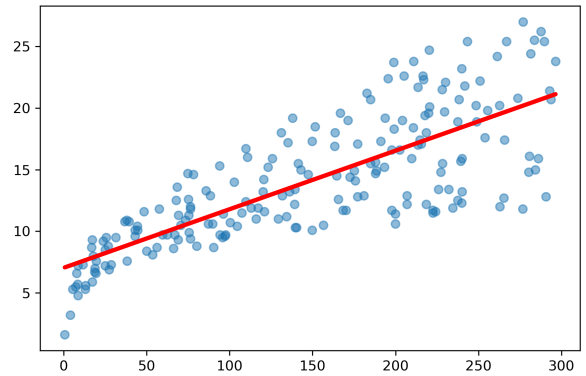
Regressione lineare multipla

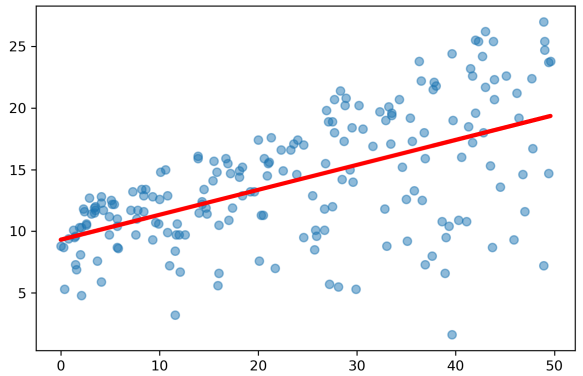
Come dipendono le vendite dagli investimenti in `TV` e `radio`?

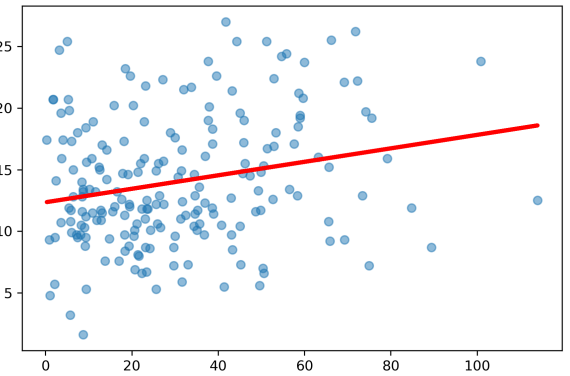


$$\texttt{sales} \approx w_0 + w_1 \cdot \texttt{TV} + w_2 \cdot \texttt{radio}$$

Regressione lineare semplice vs. multipla







Variabili utilizzate	R^2_{train}	MSE_{train}	MSE_{test}
`TV`	58.8%	10.6	10.2
`radio`	35.6%	16.6	24.2
`newspaper`	6.4%	24.1	32.1
`TV, radio, newspaper`	90.7%	2.4	4.4
`TV, radio`	90.7%	2.4	4.4

Il problema di individuare le variabili più rilevanti è detto *feature selection*

Variabili qualitative

Finora abbiamo assunto che tutti gli input siano quantitativi

Come trattare input di tipo *qualitativo*?

Es.: Se vogliamo stimare il reddito di un dipendente, potremmo avere a disposizione un dato sul sesso del dipendente (maschio/femmina)

Possiamo definire la variabile

$$x_{\text{sesso}}^{(i)} = \begin{cases} 1 & \text{se il dipendente } i\text{-esimo è femmina} \\ 0 & \text{se il dipendente } i\text{-esimo è maschio} \end{cases}$$

Il coefficiente relativo a questa variabile indicherà la dipendenza del reddito dal sesso (differenza media di reddito tra dipendenti femmine e maschi)

One-Hot Encoding

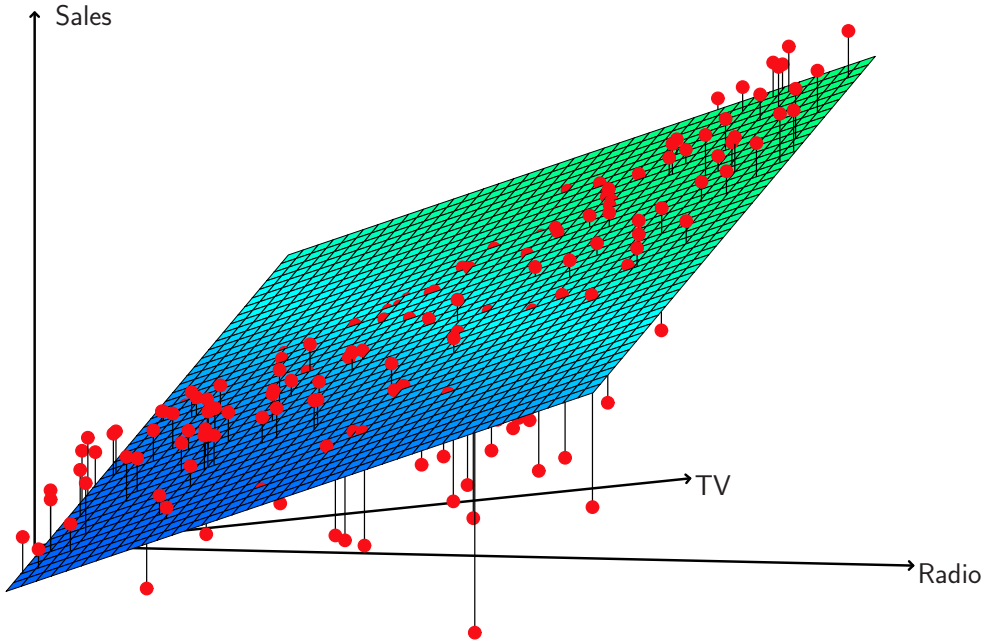
Se i valori possibili sono $K > 2$, non è corretto rappresentarli con una singola variabile, ma possiamo creare K variabili binarie

Esempio: `dieta` $\in$ {vegetariana, vegana, onnivora}

$$
\begin{aligned}
(\ldots 1\ 0\ 0 \ldots) &\text{ vegetariana} \\
(\ldots 0\ 1\ 0 \ldots) &\text{ vegana} \\
(\ldots 0\ 0\ 1 \ldots) &\text{ onnivora}
\end{aligned}
$$

Questo schema è detto *one-hot encoding*

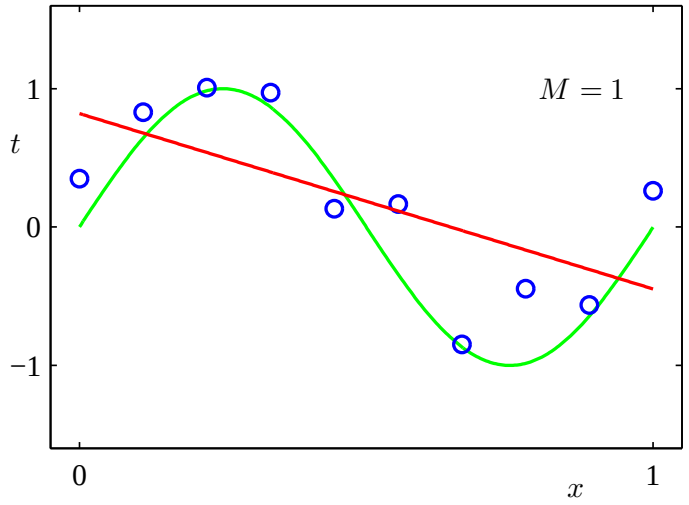
Modellare interazioni tra le variabili (*feature crossing*)



C'è una sinergia tra gli investimenti in `TV` e `radio`?
Proviamo a includere una *variabile sintetica*: `TV` $\times$ `radio`

Variabili utilizzate	R^2_{train}	MSE_{train}	MSE_{test}
`TV, radio, TV` $\times$ `radio`	97.3%	0.7	1.6

Regressione polinomiale (unidimensionale)



Per alcuni problemi, sembrano preferibili regole di predizione non-lineari

La classe dei *regressori polinomiali* di grado n è

$$\mathcal{H}^n_{poly} = \{x \mapsto p(x)\}$$

dove p è un polinomio di grado n: $p(x) = a_0 + a_1 x + \ldots + a_n x^n$

Regressione polinomiale (unidimensionale)

Definiamo la funzione $\phi : \mathbb{R} \to \mathbb{R}^{n+1}$

$$\phi(x) = (1, x, x^2, \ldots, x^n)$$

$$p(x) = w_0 + w_1 x + \ldots + w_n x^n = \langle w, \phi(x) \rangle$$

è ora una funzione lineare dell'input "espanso" $\phi(x)$

Quindi il vettore w può essere determinato con una regressione lineare, usando gli input espansi $\phi(x)$

Senza il principio ERM: Regressione non parametrica

Gli approcci visti finora sono *parametrici*: le ipotesi sono rappresentabili con un numero prefissato di parametri (per es. $w_0, w_1, \ldots, w_d$), scelti secondo il principio ERM

I metodi *non parametrici* non assumono una forma particolare per le ipotesi

- Sono più flessibili (minore bias)
- Richiedono più esempi (maggiore varianza)

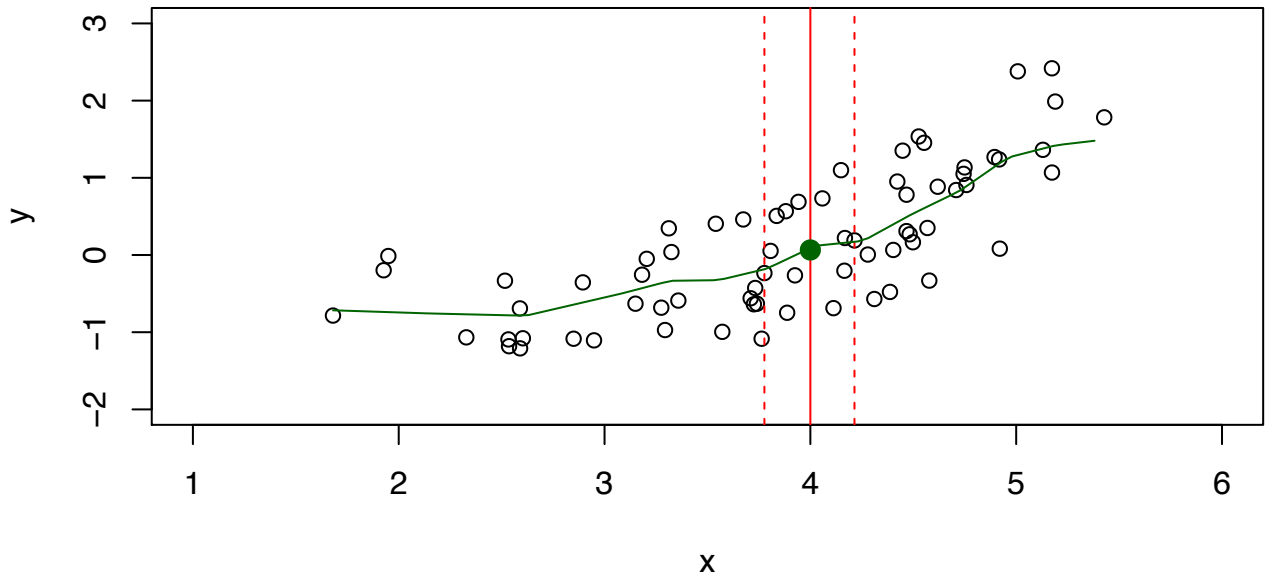
In generale, non si conformano al principio ERM ma si appoggiano direttamente alle osservazioni (*instance-based learning*)

Regressione K-Nearest Neighbors (K-NN)

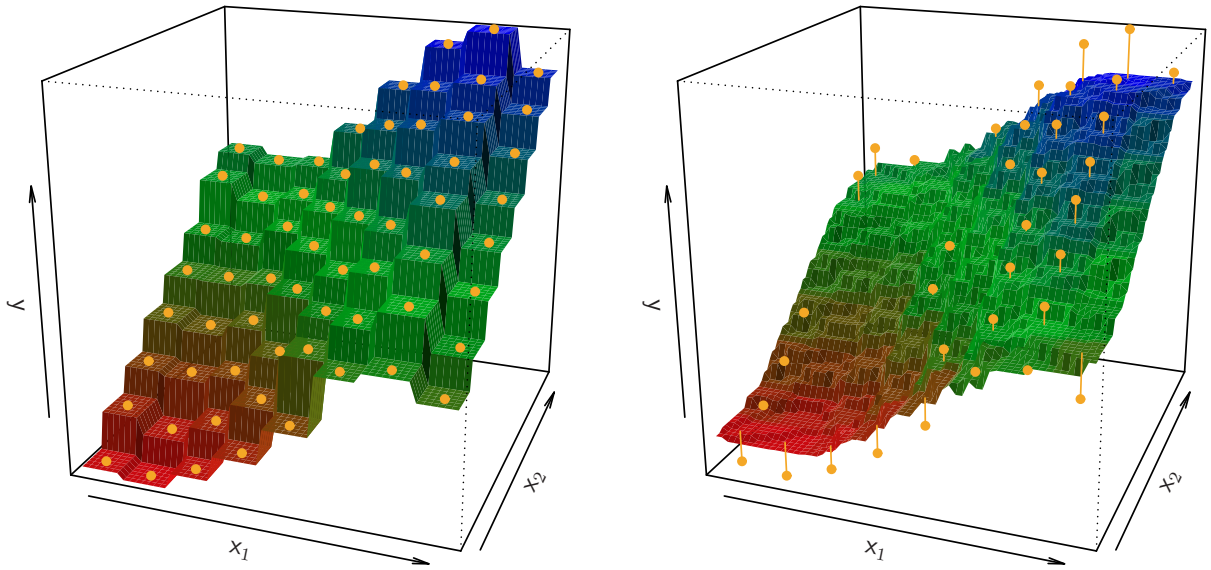
Regressione K-Nearest Neighbors (K-NN)

Sia $K \geq 1$ e sia x il punto di cui si vuole stimare il responso $h(x)$

1. Identifica i K esempi $x^{(1)}, \ldots, x^{(K)}$ più vicini ad x (in termini di distanza euclidea, o altra funzione distanza)
2. Restituisci la media del responso su quegli esempi: $h(x) = \frac{1}{K} \sum_{i=1}^{K} y^{(i)}$



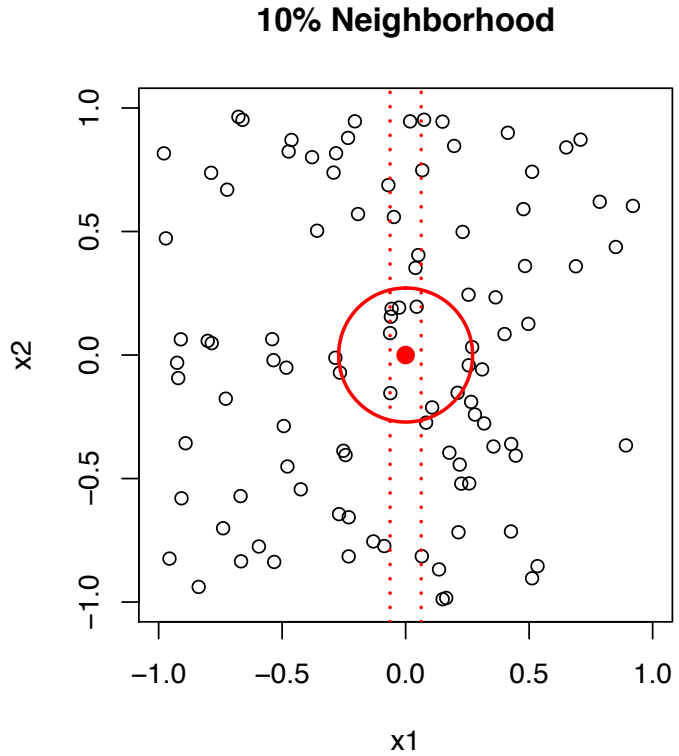
Regressione K-NN: Esempio



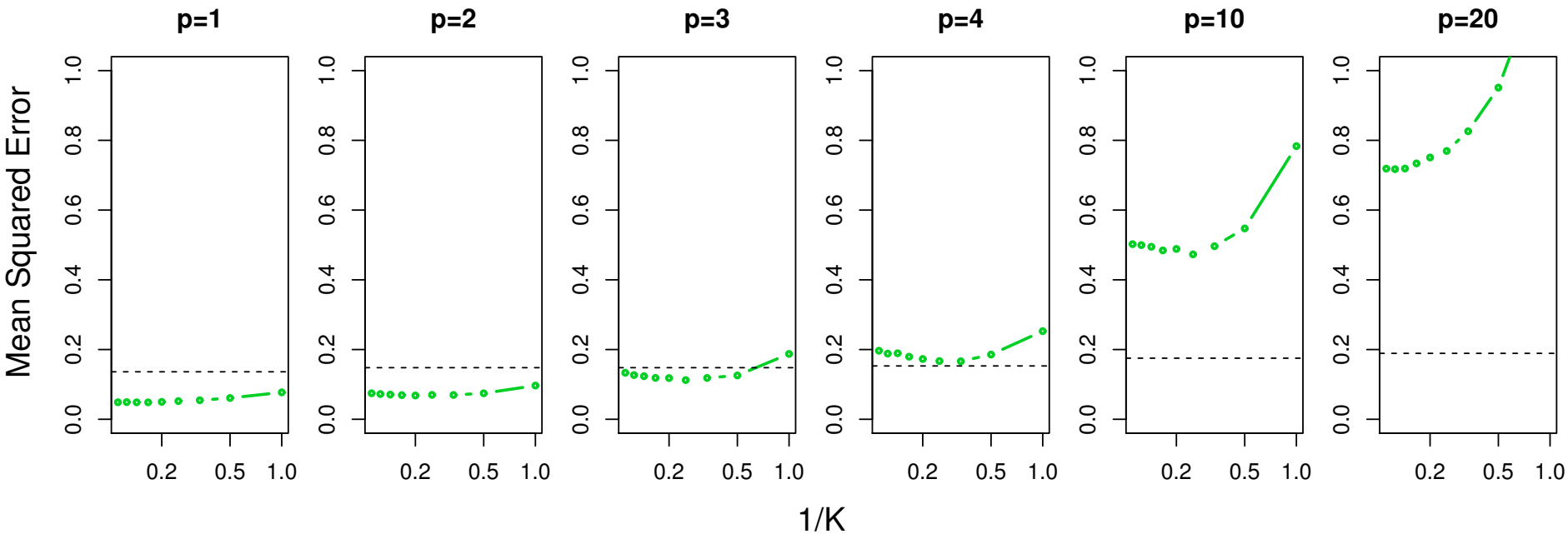
Regressione K-NN su un dataset bidimensionale di 64 osservazioni (punti arancioni)
Sinistra: $K = 1$, destra: $K = 9$

Regressione K-NN: Considerazioni

- Il metodo NN richiede accesso a tutti gli esempi ogni volta che effettua una predizione
- Tende ad essere efficace per d piccolo (ad esempio, $d \leq 4$) e m relativamente grande
- Può dare risultati scarsi per d grande: in molte dimensioni, i K punti più vicini possono essere molto lontani



Regressione K-NN vs. regressione lineare



MSE di test per una regressione lineare (linea tratteggiata nera) vs. quello di una regressione K-NN (curva verde) per una distribuzione non-lineare in 1 variabile e indipendente dalle altre $p - 1$ variabili

Tipologie di regressione viste finora

Nome	Forma delle ipotesi $h(x)$	Funzione costo $\ell(h,(x,y))$
Regressione lineare (semplice)	$w_0 + w_1 x$	$(h(x)-y)^2$
Regressione lineare (multipla)	$w_0 + w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$	$(h(x)-y)^2$
Regressione lineare generalizzata	$w_0 + w_1 \phi_1(x) + \ldots + w_n \phi_n(x)$	$(h(x)-y)^2$
Regressione K-NN	nessuna (non segue il principio ERM, a meno che $K=1$)†	$(h(x)-y)^2$

†**Esercizio**. Definire una classe di ipotesi $\mathcal{H}$ tale che la soluzione del problema ERM su $\mathcal{H}$ sia la stessa trovata dall'algoritmo di regressione 1-NN.

Regressione con altre funzioni di costo

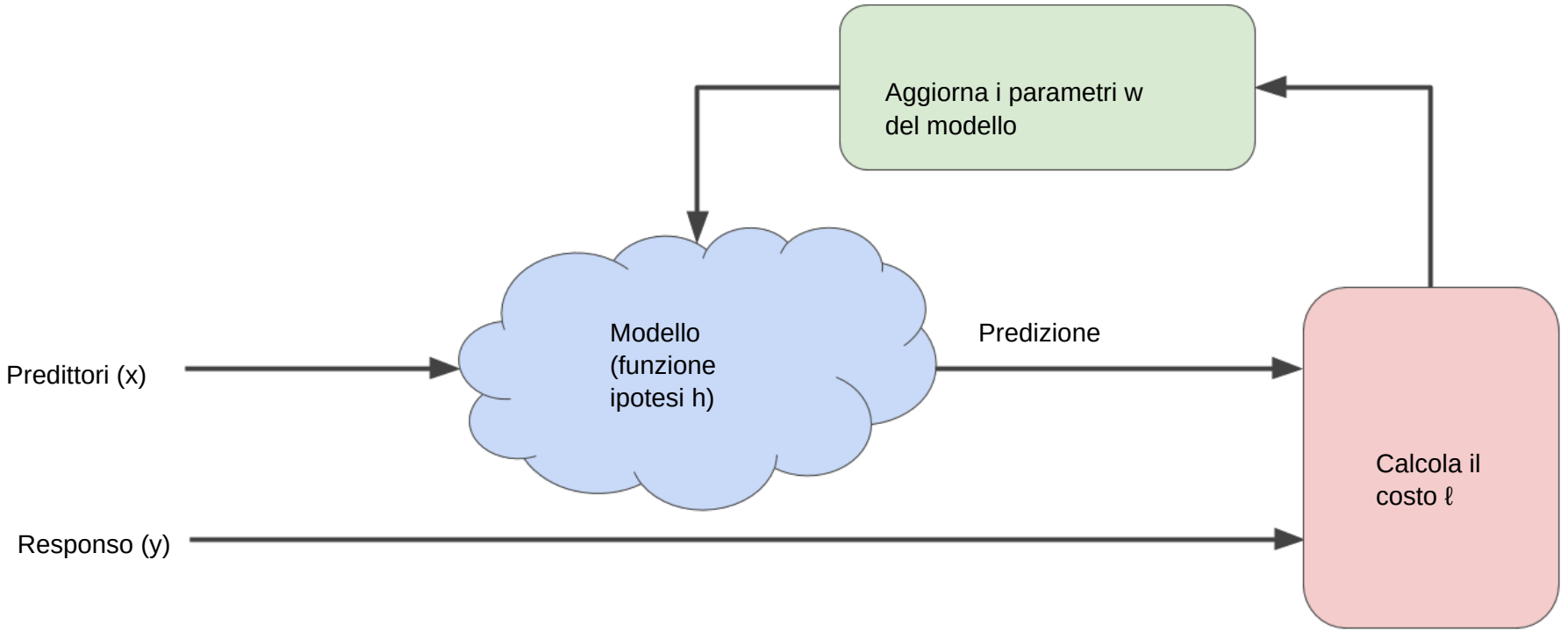
Come trattare funzioni di costo diverse da quella quadratica?

Per esempio, nella regressione *Least Absolute Deviation* (*LAD*),

$$\ell(h,(x,y)) \stackrel{\text{def}}{=} |h(x) - y|$$

Per una vasta classe di funzioni di costo (convesse e/o differenziabili) esiste una metodologia generale di ottimizzazione

Minimizzazione iterativa del rischio empirico



Discesa del gradiente (GD)

Sia $f : \mathbb{R}^d \to \mathbb{R}$ una funzione convessa differenziabile, con *gradiente*

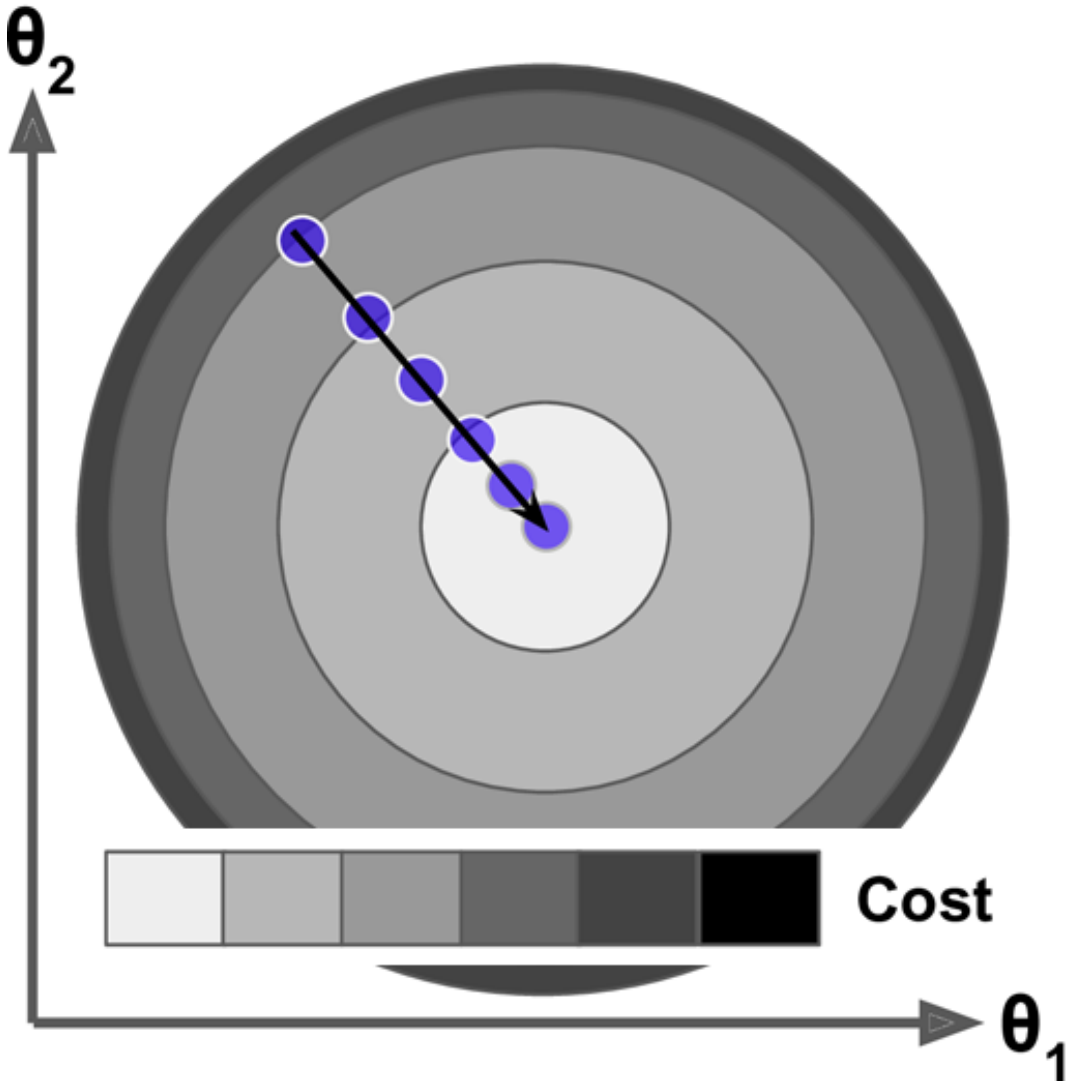
$$\nabla f(w) = \left(\frac{\partial f}{\partial w_1}(w), \ldots, \frac{\partial f}{\partial w_d}(w) \right)$$

Algoritmo Gradient Descent (generico)

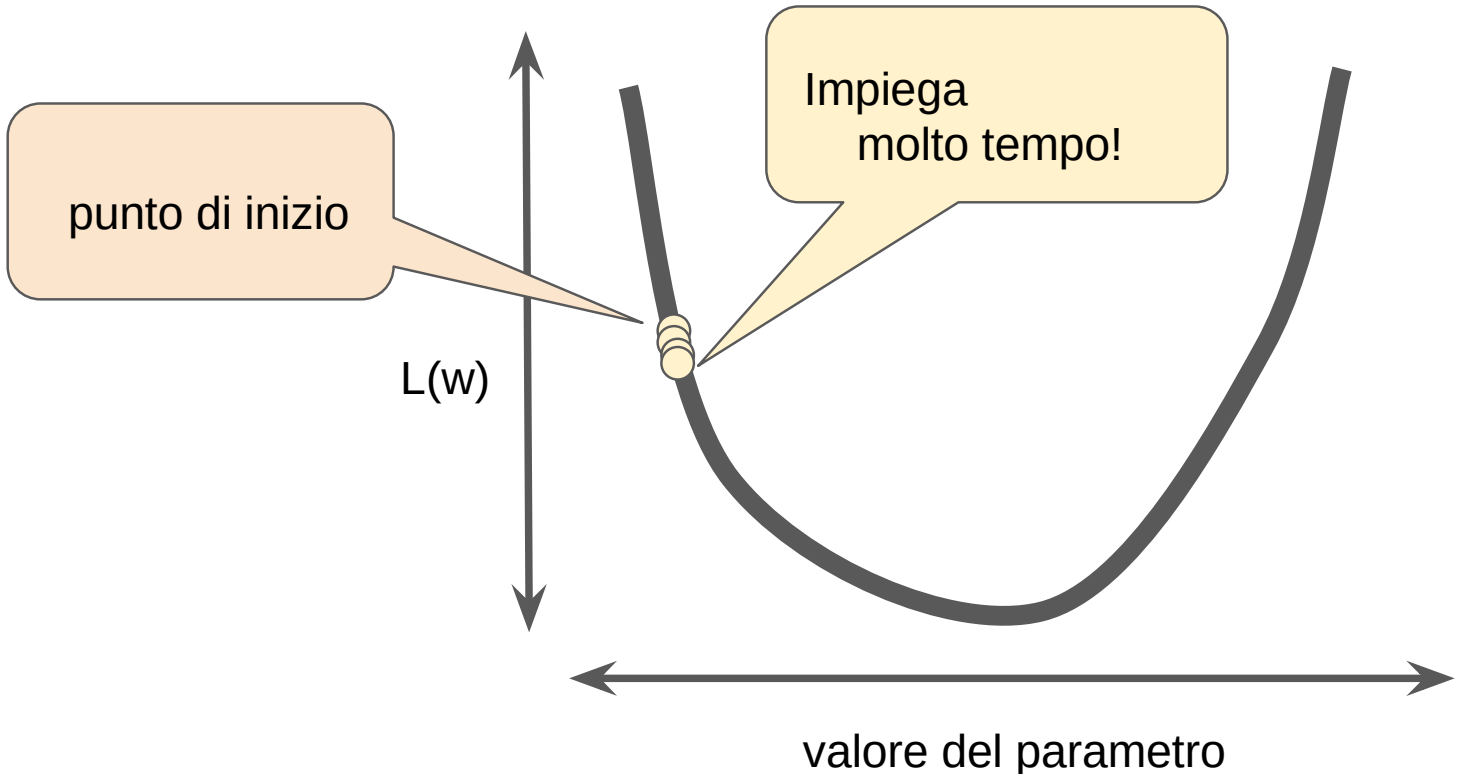
1. Poni $w^{(1)} = 0$
2. Per $t = 1, \ldots, T$: $w^{(t+1)} = w^{(t)} - \eta \cdot \nabla f(w^{(t)})$
3. Restituisci $w^{(T)}$

L'algoritmo ha due parametri: η (tasso) e T (numero di passi)
(chiamati *iperparametri* per non confonderli con i parametri w del modello)

Discesa del gradiente (GD)

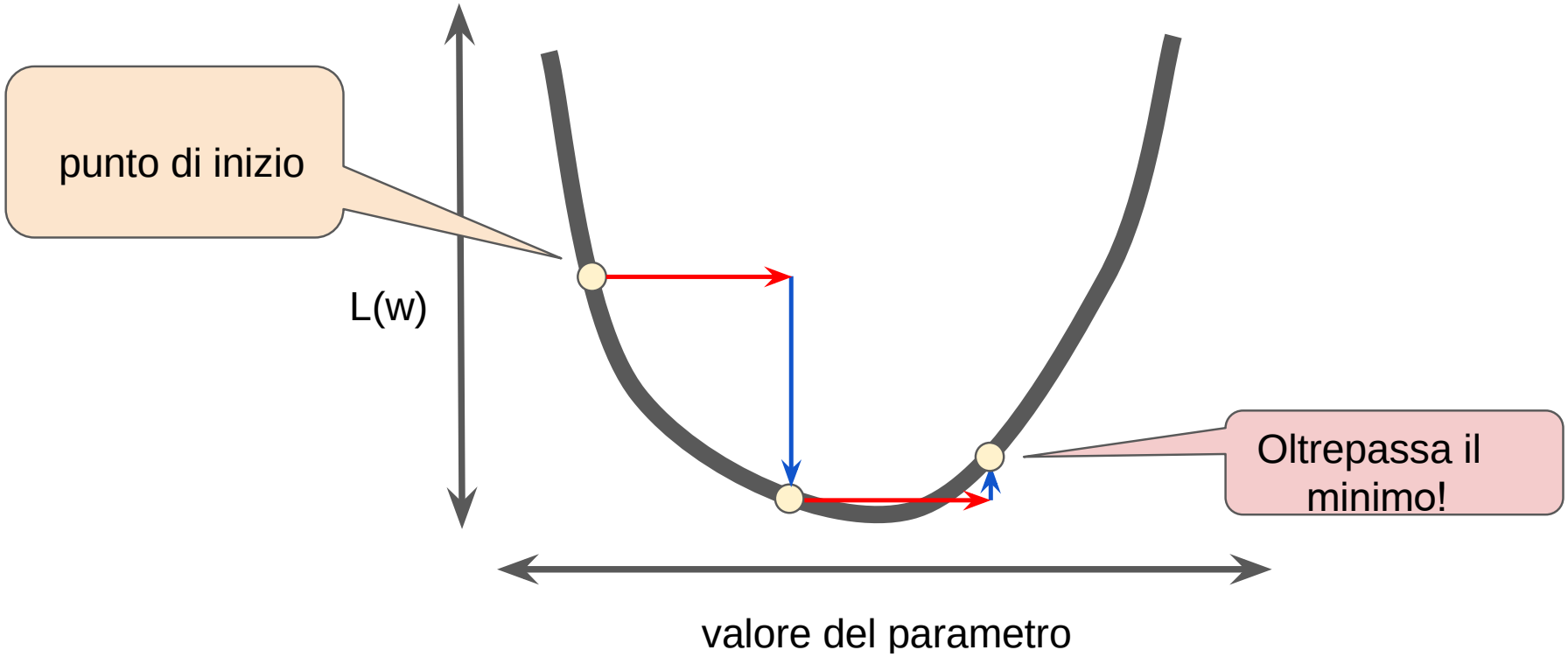


Discesa del gradiente (GD): impatto del tasso η



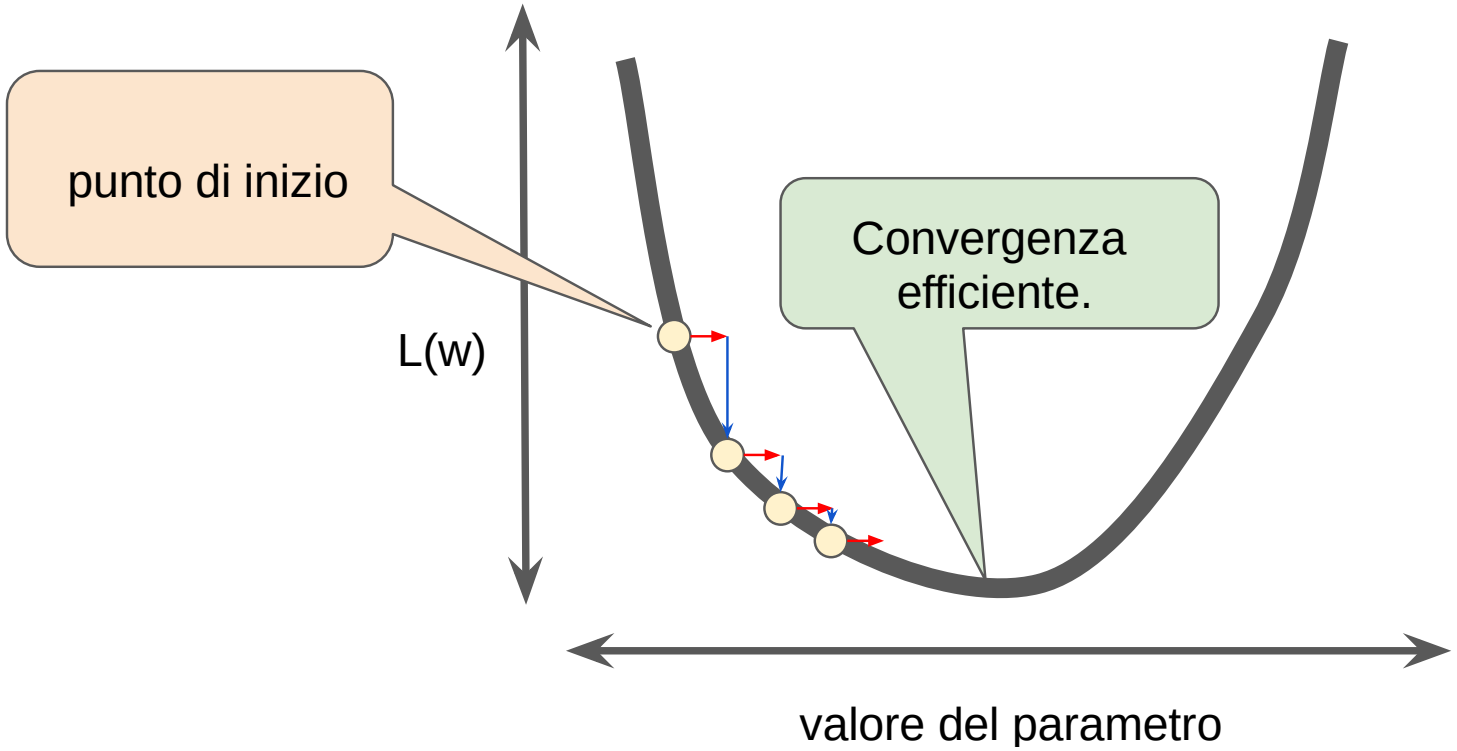
Tasso di apprendimento troppo basso

Discesa del gradiente (GD): impatto del tasso η



Tasso di apprendimento troppo alto

Discesa del gradiente (GD): impatto del tasso η



Tasso di apprendimento adeguato

Discesa del gradiente (GD): Calcolo di un passo

Il metodo GD calcola $\nabla f(w)$ ad ogni passo

Per noi, f è il rischio empirico, funzione di tutti gli esempi di training:

$$f(w) = \frac{1}{m} \sum_{i=1}^{m} \ell(h_w, (x^{(i)}, y^{(i)})) = \frac{1}{m} \sum_{i=1}^{m} \ell_i(h_w)$$

$$\nabla f(w) = \frac{1}{m} \sum_{i=1}^{m} \nabla \ell_i(h_w)$$

dove h_w è l'ipotesi codificata dal vettore w e ℓ_i è la funzione di costo sull'esempio i-esimo

Ogni passo del metodo GD richiede di considerare tutti gli m esempi! (metodo *batch*)

Discesa del gradiente (GD) riformulato

Gradient Descent (per l'apprendimento supervisionato)

1. Poni $w^{(1)} = 0$
2. Per $t = 1, \ldots, T$:
   - Poni $w^{(t+1)} = w^{(t)} - \eta \cdot \frac{1}{m} \sum_i \nabla \ell_i(h_{w^{(t)}})$
3. Restituisci $w^{(T)}$

Discesa stocastica del gradiente (SGD)

Per dei passi più rapidi, si usa una variante stocastica di GD

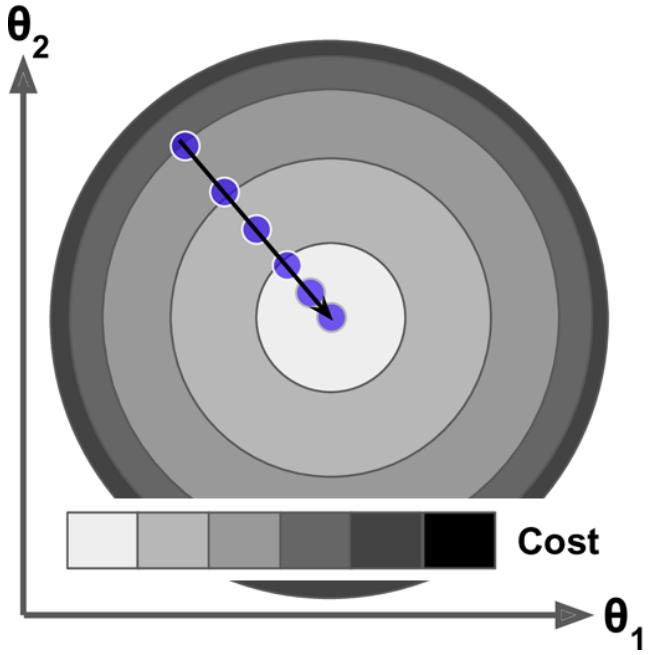
Stochastic Gradient Descent (per l'apprendimento supervisionato)

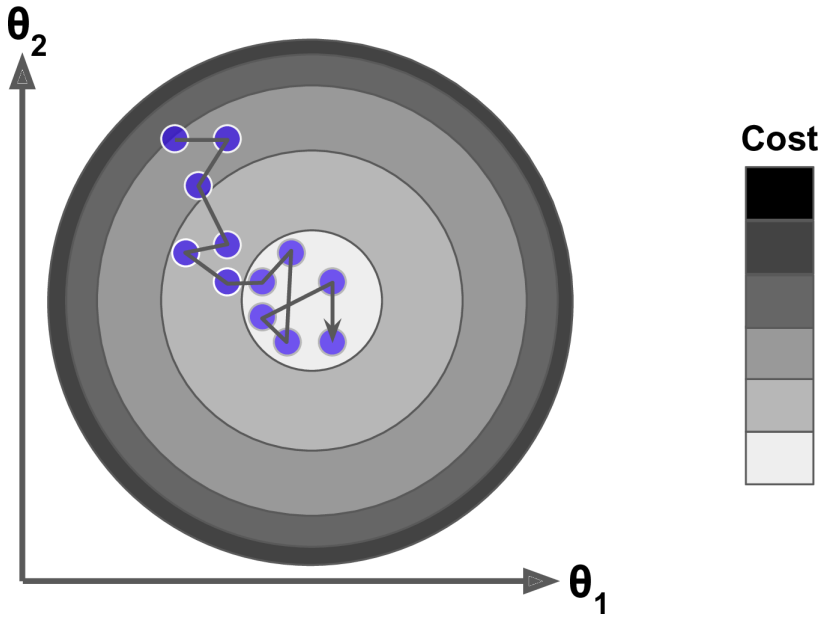
1. Poni $w^{(1)} = 0$
2. Per $t = 1, \ldots, T$:
   - Estrai un esempio i a caso
   - Poni $w^{(t+1)} = w^{(t)} - \eta \cdot \nabla \ell_i(h_{w^{(t)}})$
3. Restituisci $w^{(T)}$

Ogni passo del metodo SGD richiede di considerare un solo esempio

Poiché $\mathbb{E}_i[\nabla \ell_i(h)] = \frac{1}{m} \sum_{i=1}^{m} \nabla \ell_i(h)$, l'aggiornamento è in valore atteso lo stesso di GD

GD vs. SGD





Esempio: SGD per la regressione lineare

Esempio

Derivare la regola di aggiornamento di SGD nel seguente scenario:

- Ipotesi lineari: $h_w(x) = w^\top x$
- Costo quadratico: $\ell(h_w) = (h_w(x) - y)^2$

Calcolando il gradiente di ℓ, otteniamo la regola di aggiornamento

Regola Least-Mean-Squares (LMS)

$$w \leftarrow w - 2\eta \cdot (w^\top x - y) \cdot x$$

Esempio: SGD per la regressione LAD

Esempio

Derivare la regola di aggiornamento di SGD nel seguente scenario:

- Ipotesi lineari: $h_w(x) = w^\top x$
- Costo LAD: $\ell(h_w) = |h_w(x) - y|$

Calcolando il gradiente di ℓ, otteniamo la regola di aggiornamento

$$w \leftarrow w - \eta \cdot \operatorname{sgn}(w^\top x - y) \cdot x$$

$$(\operatorname{sgn}(z))_j = \begin{cases} +1 & \text{se } z_j > 0 \\ -1 & \text{se } z_j < 0 \\ 0 & \text{se } z_j = 0 \end{cases}$$

Mini-batch SGD

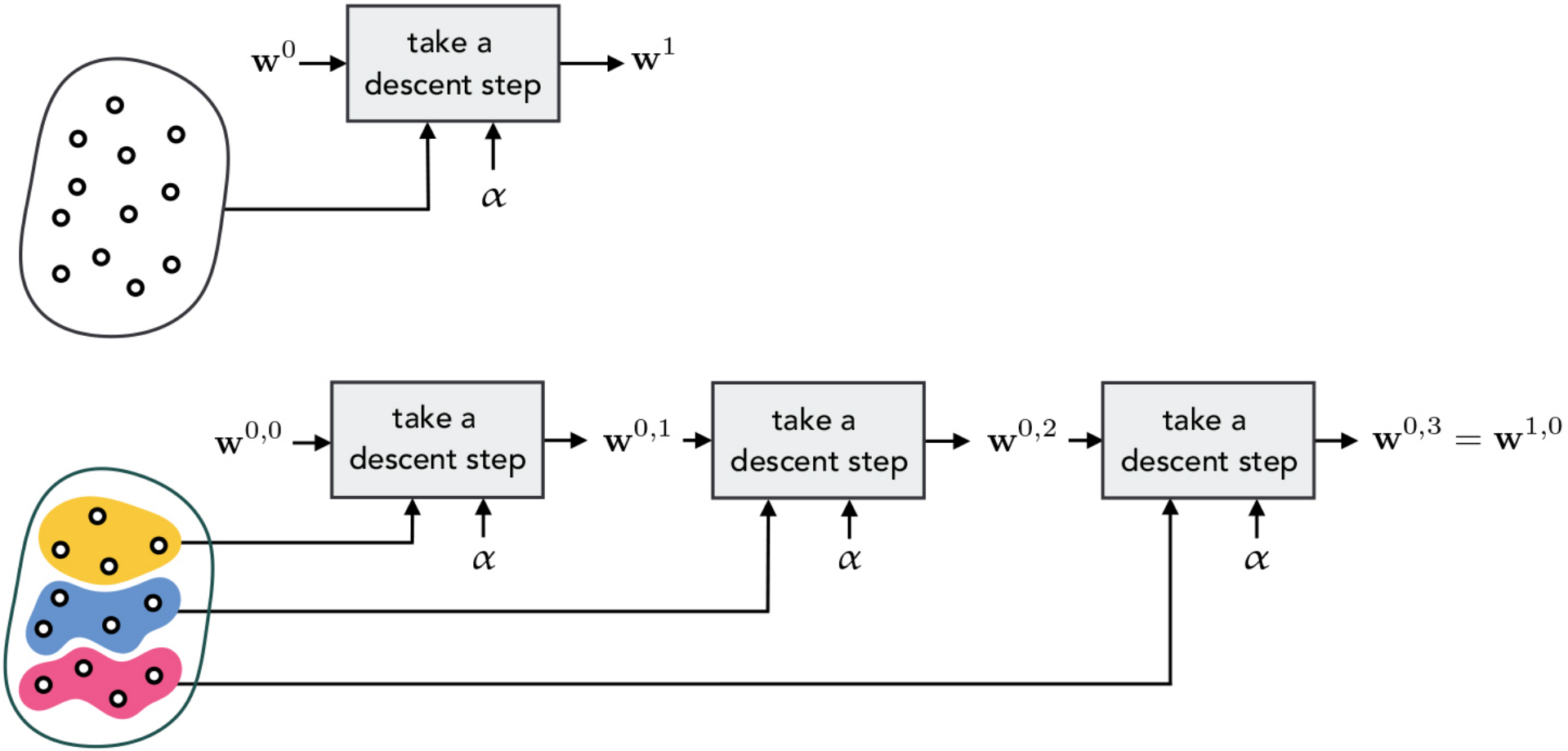
Mini-batch SGD è un compromesso tra GD e SGD

Mini-Batch SGD (per l'apprendimento supervisionato)

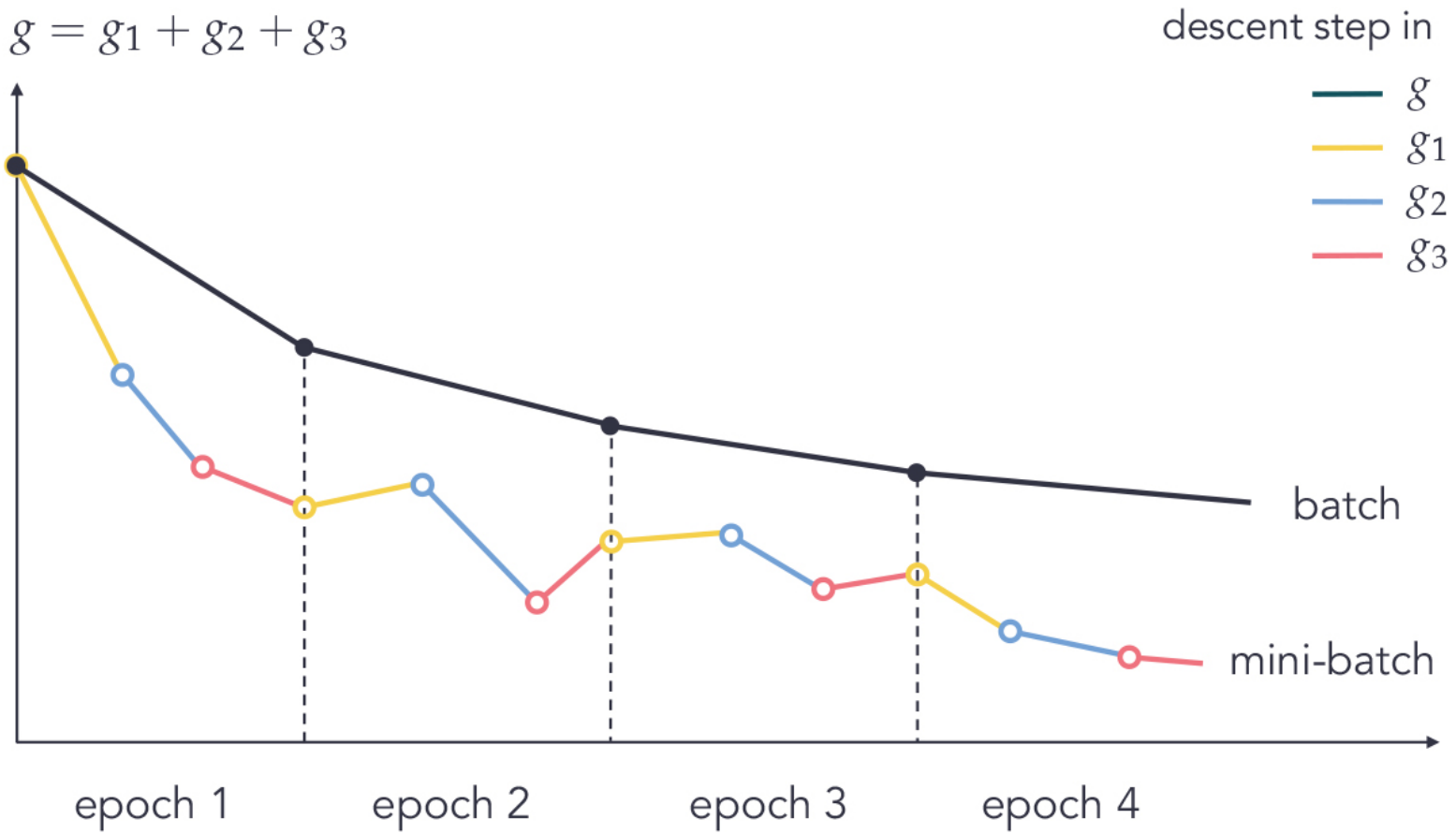
1. Poni $w^{(1)} = 0$
2. Per $t = 1, \ldots, T$:
   - Estrai un lotto (*batch*) B di esempi a caso
   - Poni $w^{(t+1)} = w^{(t)} - \eta \cdot \frac{1}{|B|} \sum_{i \in B} \nabla \ell_i(h_{w^{(t)}})$
3. Restituisci $w^{(T)}$

Ogni passo di mini-batch SGD richiede di considerare $|B|$ esempi

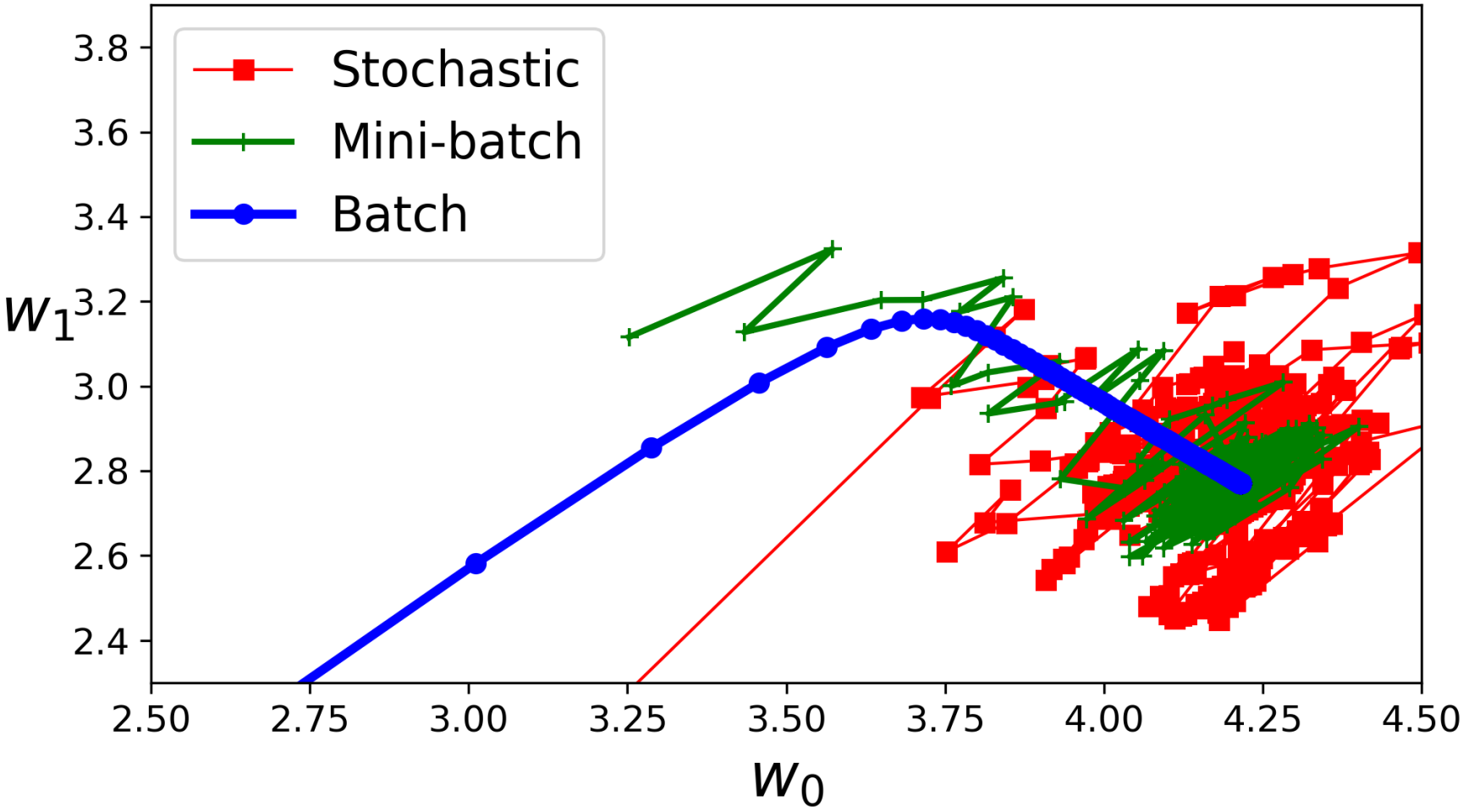
Batch GD vs. Mini-Batch SGD



Batch GD vs. Mini-Batch SGD



Batch GD vs. SGD vs. Mini-Batch SGD



Scalabilità computazionale dei metodi di regressione

m: numero di esempi
d: numero di variabili di input

Algoritmo	Esempi scanditi ad ogni passo	Dipendenza da d	Iperparametri	Interfaccia `scikit-learn`
Diretto (SVD)	m	$\Omega(d^2)$	nessuno	`LinearRegression`
Batch GD	m	$\Theta(d)$	η, T	`SGDRegressor`
SGD	1	$\Theta(d)$	η, T	`SGDRegressor`
Mini-batch SGD	$\lvert B\rvert$	$\Theta(d)$	$\eta, T, \lvert B\rvert$	`SGDRegressor`
K-NN	m	$\Theta(d)$	K	`KNeighborsRegressor`