



**Università  
degli Studi  
di Palermo**



# Introduzione al Machine Learning

CORSO DI BIG DATA  
a.a. 2022/2023

Prof. Roberto Pirrone

# Sommario

- Generalità
- Tipologie dei compiti di apprendimento
- Utilizzo dei dati
- Capacità e generalizzazione
- Tecniche di addestramento

# Generalità

- Il Machine Learning – Apprendimento Automatico in Italiano – si riferisce allo sviluppo di *programmi per computer che siano in grado di apprendere dai dati*
- In generale, il programma avrà a disposizione una esperienza  $E$ , rispetto a una classe di compiti  $T$  e una misura di performance  $P$
- *L'apprendimento implica che la performance  $P$  sulla classe di compiti  $T$  migliorerà utilizzando  $E$*

# Generalità

- Un algoritmo di apprendimento deve fornire la stima statistica di una funzione complessa che lega i dati alle uscite desiderate

*Modello*: caratterizzazione della famiglia di forme funzionali utilizzate per  $f$  e/o della tipologia di algoritmo impiegato

*Parametri del modello*: quantità che sono direttamente coinvolte nella forma funzionale di  $f$  e devono essere apprese dai dati

$$y = f(x, w; \theta)$$

*Iperparametri*: quantità che condizionano la corretta evoluzione dell'algoritmo, ma che devono essere fissate in fase di apprendimento; possono essere stimate usando i dati, ma con tecniche diverse dall'apprendimento vero e proprio

# Tipologie dei compiti di apprendimento

- Classificazione

- L'algoritmo stima una funzione che fa corrispondere ogni ingresso ad una di  $k$  classi distinte

$$f : \mathbb{R}^n \rightarrow \{1, \dots, k\}$$

- Nel caso di classificazione con valori mancanti nelle feature, il compito diviene quello di stimare un insieme di funzioni che classificano gli ingressi per ogni possibile sottoinsieme di feature mancanti
- Un altro approccio è quello di apprendere una distribuzione di probabilità su tutte le variabili e poi marginalizzare quelle mancanti

# Tipologie dei compiti di apprendimento

- Regressione
  - Predizione di un valore numerico a partire dagli ingressi
  - Simile alla classificazione, ma la la funzione  $f: \mathbb{R}^n \rightarrow \mathbb{R}$

# Tipologie dei compiti di apprendimento

- Trascrizione
  - Osservazione di dati scarsamente strutturati e trascrizione in forma testuale
  - OCR, speech recognition
  - E' riconducibile alla classificazione

# Tipologie dei compiti di apprendimento

- Traduzione automatica (machine translation)
  - Conversione di una sequenza discreta di simboli in un'altra
  - Applicata al linguaggio naturale



# Tipologie dei compiti di apprendimento

- Compiti con output strutturato
  - In generale ci si riferisce ai compiti di Machine Learning in cui l'uscita è un vettore o un'altra struttura dati i cui valori sono strettamente correlati tra loro
    - Task di NLP quali il parsing o il QA
    - Segmentazione e annotazione di immagini

# Tipologie dei compiti di apprendimento

- Rilevamento di anomalie (anomaly detection)
  - Individuazione di ingressi tali da essere eccezionali o atipici rispetto ad un dato comportamento «normale» dei campioni della popolazione
  - Riconducibili a compiti di clustering e/o classificazione interpretati in forma duale e cioè cercando gli elementi «non appartenenti» alla classe normale

# Tipologie dei compiti di apprendimento

- Sintesi e campionamento
  - Generazione di campioni simili ai dati di ingresso
  - Applicazioni in campo multimediale
    - Generazione automatica di texture nei video-game
    - Speech synthesis

# Tipologie dei compiti di apprendimento

- Imputazione di dati mancanti
  - Predizione di feature mancanti a partire da input incompleto
- Denoising
  - Predizione di  $\mathbf{x} \in \mathbb{R}^n$ , noto che sia  $\mathbf{x}'$  versione corrotta di  $\mathbf{x}$
  - Più in generale, predizione di  $p(\mathbf{x} \mid \mathbf{x}')$

# Tipologie dei compiti di apprendimento

- Stima di una densità di probabilità o di una funzione massa
  - Si tratta del compito di stimare la funzione

$$p_{\text{model}} : \mathbb{R}^n \rightarrow \mathbb{R}$$

che deve intendersi come una funzione densità di probabilità  $p(\mathbf{x})$ , o funzione massa in caso di variabile discreta

- In genere si usa per disporre di una stima esplicita di  $p(\mathbf{x})$  da usare in altri compiti di apprendimento che la richiedono

# Utilizzo dei dati

- Apprendimento non supervisionato (unsupervised learning)
  - In questi algoritmi si deve apprendere la distribuzione di probabilità  $p(\mathbf{x})$  che sottende il data set
  - Si è interessati, in generale, ad apprendere la struttura dei dati
  - Es. clustering, compiti di sintesi, denoising

# Utilizzo dei dati

- Apprendimento supervisionato (supervised learning)
  - In questi algoritmi ogni campione  $\mathbf{x}$  dei dati di ingresso è associato ad una «etichetta»  $\mathbf{y}$  numerica o testuale
  - L'algoritmo deve apprendere a predire  $\mathbf{y}$  dato  $\mathbf{x}$ , ovvero a predire  $p(\mathbf{y} | \mathbf{x})$
  - Classificazione e regressione, anche con reti neurali

# Utilizzo dei dati

- Apprendimento per rinforzo (reinforcement learning)
  - Il data set non è fisso e l'algoritmo interagisce con l'ambiente, apprendendo tramite una procedura di prova ed errore
  - L'algoritmo (o meglio l'agente) al tempo  $t$  si trova in uno stato  $S_t$  ed esegue un'azione  $A_t$  ottenendo una ricompensa o «reward»  $R_t$
  - L'agente si evolve lungo un ciclo stato-azione-ricompensa



# Utilizzo dei dati

- Apprendimento per rinforzo (reinforcement learning)
  - La ricompensa globale  $G_t$  al tempo  $t$  è la somma di tutti i possibili valori di ricompensa legati ai comportamenti futuri, almeno fino ad un certo orizzonte temporale di  $T$  istanti di tempo nel futuro.

$$G_t = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \text{ where } \gamma \in [0, 1)$$

- $\gamma$  è il «fattore di sconto» che pesa sempre meno le azioni compiute nel futuro più lontano

# Utilizzo dei dati

- Tipi di apprendimento per rinforzo (1/2)
  - L'agente cerca di massimizzare il valore di ricompensa (value based) tenendo conto che utilizza una funzione di policy per la scelta dell'azione rispetto allo stato corrente  $a = \pi(s)$ :

$$v_{\pi}(s) = \mathbb{E}_{\pi} [R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \dots \mid S_t = s]$$

- L'agente cerca di massimizzare direttamente la funzione  $\pi$  eventualmente definita in forma probabilistica come  $\pi(a \mid s) = P[A_t = a \mid S_t = s]$

# Utilizzo dei dati

- Tipi di apprendimento per rinforzo (2/2)

- Si addestra un modello di ML eventualmente anche deep per massimizzare una funzione di «qualità» legata all'intraprendere una data azione posto che in input si fornisca lo stato corrente

- Q learning:

$$Q(s_t, a_t) \leftarrow \underbrace{Q(s_t, a_t)}_{\text{vecchio valore}} + \underbrace{\alpha_t(s_t, a_t)}_{\text{tasso di apprendimento}} \times \left[ \underbrace{R_{t+1}}_{\text{ricompensa}} + \underbrace{\gamma}_{\text{fattore di sconto}} \underbrace{\max_{a_{t+1}} Q(s_{t+1}, a_{t+1})}_{\text{valore futuro massimo}} - \underbrace{Q(s_t, a_t)}_{\text{vecchio valore}} \right]$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t)(1 - \alpha_t(s_t, a_t)) + \alpha_t(s_t, a_t)[R_{t+1} + \gamma \max_{a_{t+1}} Q(s_{t+1}, a_{t+1})]$$

# Capacità e generalizzazione

- L'obiettivo principale di un algoritmo di apprendimento è quello di *generalizzare* a partire dai dati osservati, cioè *fare corrette predizioni su dati che non ha mai osservato prima*
- Ogni algoritmo deve minimizzare due misure di errore:
  - Training error, cioè la misura di errore sui dati usati per l'addestramento
  - Test error o generalization error, cioè la misura di errore sui dati utilizzati per testare la bontà della performance e che *sono assolutamente distinti da quelli di addestramento*
  - La forma funzionale  $L$  di queste misure di errore, che in generale chiameremo *loss*, è la stessa e dipende, algoritmo per algoritmo, *dal tipo di stima statistica che stiamo conducendo*

# Capacità e generalizzazione

- La capacità di generalizzare si basa sulla Teoria dell'Apprendimento Statistico
  - Esiste un processo statistico unico che descrive il fenomeno sotto esame e che genera i dati usati per l'addestramento, ma anche quelli mai visti
  - Questo processo è descritto da un'unica distribuzione di probabilità  $p_{\text{data}}$ , non nota a priori
  - Si può stabilire, quindi, un'ipotesi a priori sui dati che è il fondamento di ogni algoritmo di ML: i dati di addestramento e di test sono *indipendenti e identicamente distribuiti (independent and identically distributed - i.i.d.)*

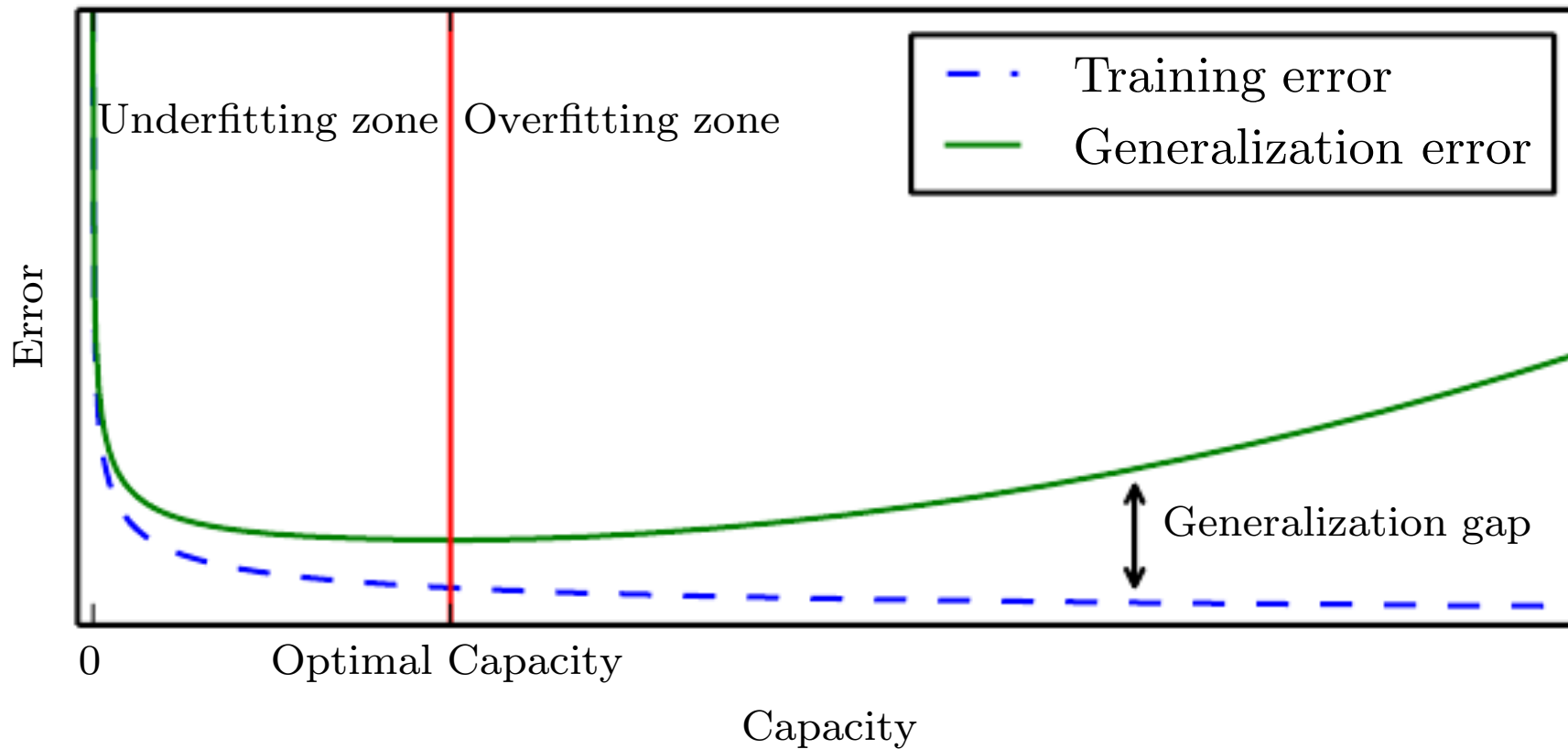
# Capacità e generalizzazione

- L'ipotesi i.i.d. ci dice che, per qualunque modello, il valore atteso del training error e del test error sono uguali
  - Esiste certamente una configurazione dei parametri  $\mathbf{w}$  per cui questi due errori sono uguali
  - Nella realtà, a causa del campionamento dei dati disponibili, questa condizione non si verifica
    - L'algoritmo cerca di minimizzare il training error attraverso l'addestramento sul training set che è una procedura di minimizzazione della loss
    - L'algoritmo cerca di mantenere minimo il gap con il generalization error

# Capacità e generalizzazione

- Overfitting
  - Eccesso di minimizzazione del training error, con conseguente scarsa generalizzazione (generalization error elevato)
- Underfitting
  - Bassa capacità di minimizzare il training error

# Capacità e generalizzazione





# Capacità e generalizzazione

- Si parla di *capacità* di un modello con riferimento al fatto che questo riesca ad approssimare la gamma di forme funzionali più vasta possibile
- Il ventaglio di forme funzionali direttamente ottenibili dal nostro modello si dirà «spazio delle ipotesi»
- La capacità del modello sarà *gestibile tramite i suoi iperparametri*

# Capacità e generalizzazione

- No free lunch theorem
  - La performance di un qualunque algoritmo di apprendimento mediata su qualunque distribuzione di probabilità dei dati *è la stessa*
  - Non ci sono algoritmi in linea di principio migliori degli altri
  - In realtà il disegno di un modello basato su particolari assunzioni fatte su  $p_{\text{data}}$ , ovvero la scelta accurata dello spazio delle ipotesi, rende certi algoritmi migliori degli altri su quella particolare forma di processo di generazione dei dati

# Capacità e generalizzazione

- Esempio – la regressione

$$y = \mathbf{w} \cdot \mathbf{x} + b, \quad \text{regressione lineare}$$

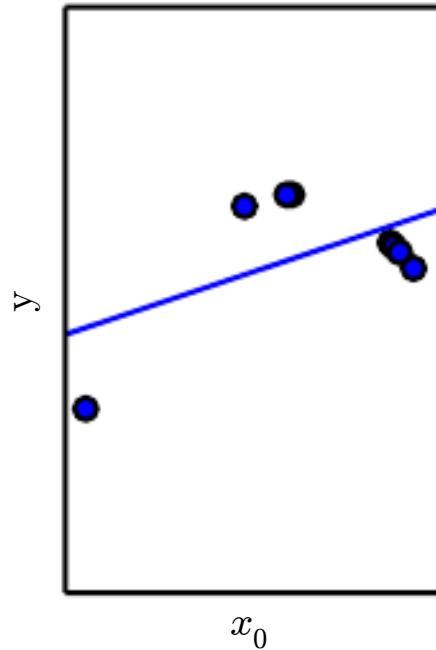
$$y = \sum_{i=1}^n \mathbf{w}_i \cdot \mathbf{x}^i + b, \quad \text{regressione polinomiale}$$

- Spazio delle ipotesi  $\rightarrow$  funzioni polinomiali
- Parametri  $\rightarrow \{b, \mathbf{w}_i \mid i = 1, \dots, n\}$
- Iperparametro  $\rightarrow n$

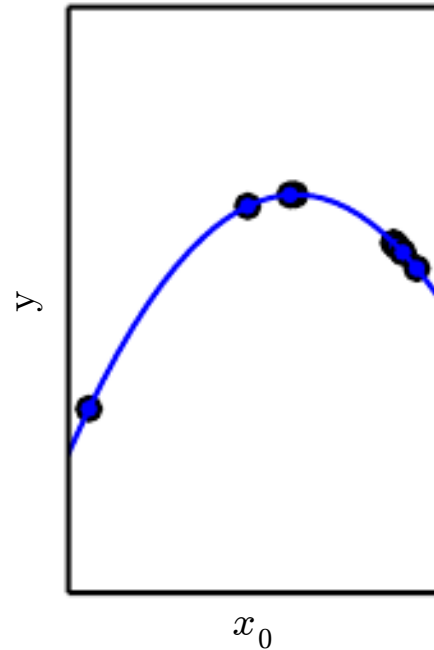
# Capacità e generalizzazione

- Esempio – la regressione

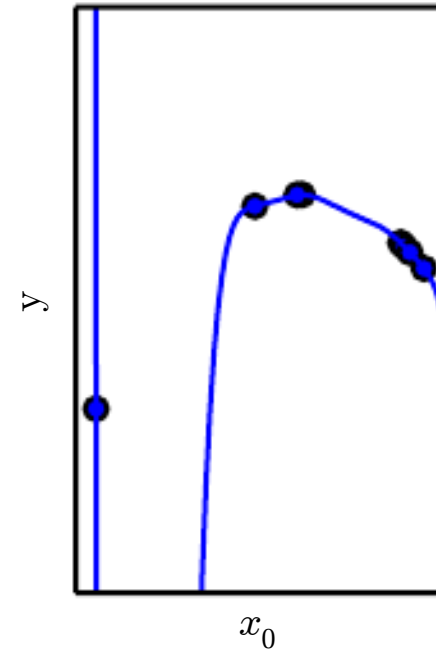
Underfitting



Appropriate capacity



Overfitting

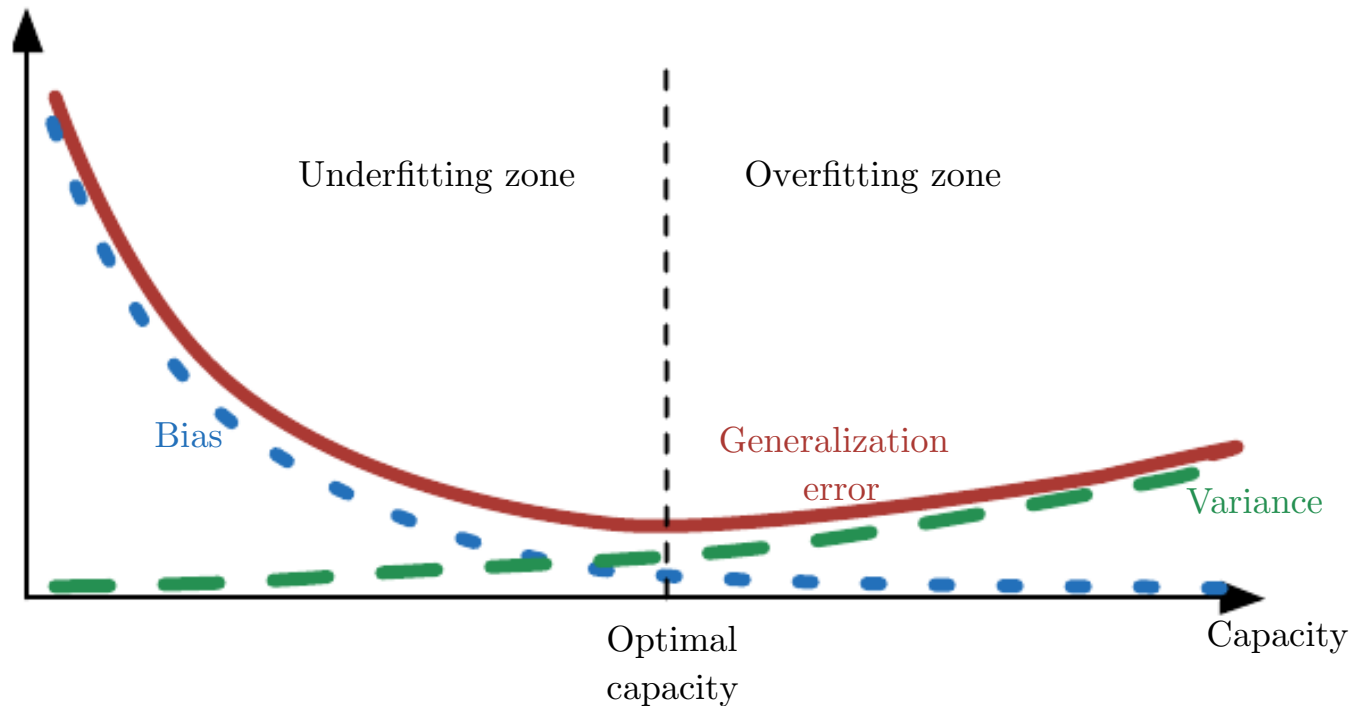


# Capacità e generalizzazione

- Esempio – la regressione
  - L'algoritmo di regressione apprende minimizzando il  $MSE_{train}$  calcolato sul training set, mantenendo minimo il gap dal generalization error  $MSE_{test}$
  - Questo, come ricorderemo, corrisponde ad una stima MLE dei parametri ottimi di  $p_{model}(y|\mathbf{x}_{\sim p_{data}})$  supposta Gaussiana
  - È possibile rivedere l'andamento della stima in termini della capacità

# Capacità e generalizzazione

- Esempio – la regressione

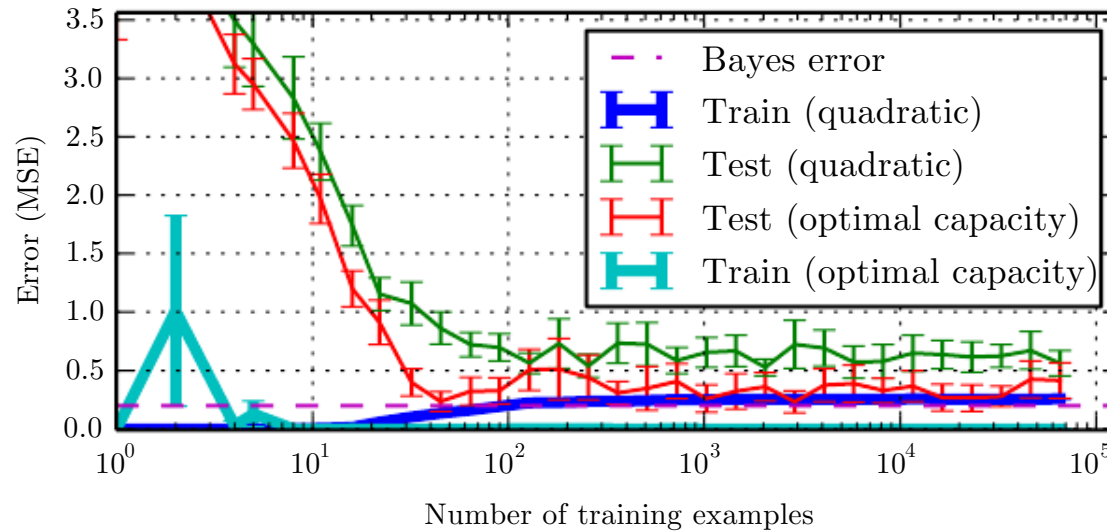


$$\text{MSE} = \mathbb{E} \left[ (\hat{\theta}_m - \theta)^2 \right] = \text{Bias}(\hat{\theta}_m)^2 + \text{Var}(\hat{\theta}_m)$$

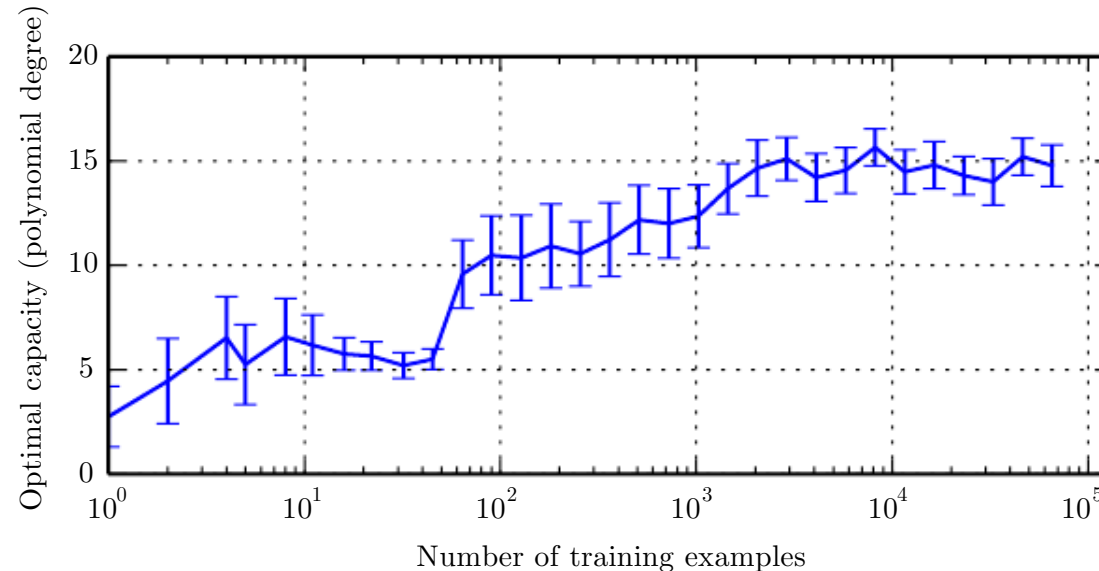
# Capacità e generalizzazione

- Il training error e la generalizzazione di un algoritmo di apprendimento dipendono anche dalle dimensioni del training set
- In linea di principio un «oracolo» che conosca esattamente  $p_{\text{data}}$  commette un errore costante di predizione noto come *Bayes error*
  - L'oracolo non conosce «esattamente» i dati, ma la loro distribuzione di probabilità
- Al crescere dei dati di addestramento, l'errore di generalizzazione tende asintoticamente ad un valore che è maggiore (in caso di bassa capacità del modello) o uguale al Bayes error

# Capacità e generalizzazione



Data set sintetico generato aggiungendo rumore ai punti ottenuti da un polinomio di grado 5





# Tecniche di addestramento

- Gli algoritmi di apprendimento utilizzano diverse tecniche che modificano le procedure di addestramento standard al fine di aumentare la *capacità effettiva* del modello prescelto
- Queste tecniche impattano direttamente sulla scelta degli iperparametri da cui dipende la *capacità di rappresentazione* (teorica) del modello
  - Regularizzazione
  - Validation set e cross-validation

# Tecniche di addestramento

- Regularizzazione
  - Una qualunque modifica dell'algoritmo di apprendimento *mirata a ridurre esplicitamente l'errore di generalizzazione*, ma non il training error
  - Il termine di regularizzazione ci consente di scegliere alcune forme funzionali rispetto alle altre nel nostro spazio delle ipotesi

# Tecniche di addestramento

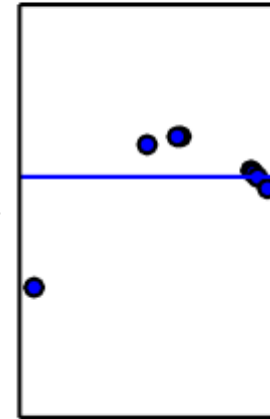
- Regularizzazione
  - Esempio – regularizzazione «weight decay» per la regressione che esprime preferenza per piccoli valori di  $\mathbf{w}$

$\lambda$  è un ulteriore iperparametro

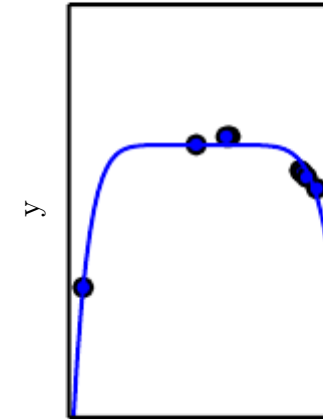
$$J(\mathbf{w}) = \text{MSE}_{\text{train}} + \lambda \mathbf{w}^\top \mathbf{w}$$

Un tipico effetto della regularizzazione è lo *smoothing* della funzione di loss per cui, analiticamente,  $J$  converge verso un *minimo globale diverso* dalla pura MSE

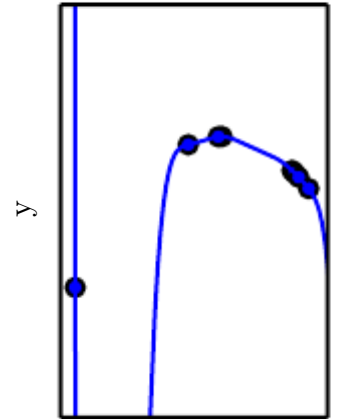
Underfitting  
(Excessive  $\lambda$ )



Appropriate weight decay  
(Medium  $\lambda$ )



Overfitting  
( $\lambda \rightarrow 0$ )



Il modello è un polinomio di grado 9

# Tecniche di addestramento

- Validation set e cross-validation
  - Il validation set si utilizza per ovviare al problema dell'*apprendimento degli iperparametri*
  - Il validation set è una porzione del training set che viene espunta da quest'ultimo prima di iniziare l'addestramento vero e proprio e viene utilizzata per verificare l'andamento del generalization error, modificando gli iperparametri per minimizzarlo
  - L'errore commesso sul validation set durante l'addestramento risulta essere una stima più corretta dell'errore di generalizzazione rispetto al training error

# Tecniche di addestramento

- La procedura di training restituisce il miglior modello rispetto al validation set

---

**Algorithm 1** Full training procedure

---

**Input:**  $A, \mathbf{x}^{(train)}, split\_perc, H$  { $A$ : the algorithm,  $\mathbf{x}^{(train)}$ : the training data,  $split\_perc$ : split percentage,  $H$ : hyperparameters search space}

$\mathbf{w} \leftarrow \text{init\_weights}()$  {initialize the parameters  
as either zero or random values}

$L \leftarrow \infty, L^{val} \leftarrow \infty$  {initial values of the loss both for training  
and validation}

$(train\_set, val\_set) \leftarrow \text{split}(\mathbf{x}^{(train)}, split\_perc)$  {split the training data}

**for all**  $\theta \in H$  **do**

$(L, \mathbf{w}) \leftarrow A(train\_set, \mathbf{w}, \theta)$  {training provides loss and weights}

**if**  $L(val\_set, \mathbf{w}, \theta) < L^{val}$  **then**  
{save the state of the model related to the best validation error}

$L^{val} \leftarrow L(val\_set, \mathbf{w}, \theta)$

$\theta^* \leftarrow \theta$

$\mathbf{w}^* \leftarrow \mathbf{w}$

$L^* \leftarrow L$

**end if**

**end for**

**Output:**  $L^*, \mathbf{w}^*, \theta^*$  {output the best model as regards validation}

---

# Tecniche di addestramento

- Validation set e cross-validation
  - Quando ci si trova con un numero ridotto di campioni si adotta la tecnica della *k-fold cross validation*
  - Il data set viene suddiviso in  $k$  partizioni che non si sovrappongono
  - Per ogni partizione si adotta una strategia di addestramento per cui la partizione fa da test set e il resto da training set
    - All'interno di ogni run di training si può individuare un validation set per addestrare il modello rispetto ai propri iperparametri
  - L'errore commesso viene calcolato come il *valore medio su tutte le partizioni* e riportato come performance del modello sull'intero data set

# Tecniche di addestramento

---

**Define**  $\text{KFoldXV}(\mathbb{D}, A, L, k)$ :

**Require:**  $\mathbb{D}$ , the given dataset, with elements  $z^{(i)}$

**Require:**  $A$ , the learning algorithm, seen as a function that takes a dataset as input and outputs a learned function

**Require:**  $L$ , the loss function, seen as a function from a learned function  $f$  and an example  $z^{(i)} \in \mathbb{D}$  to a scalar  $\in \mathbb{R}$

**Require:**  $k$ , the number of folds

Split  $\mathbb{D}$  into  $k$  mutually exclusive subsets  $\mathbb{D}_i$ , whose union is  $\mathbb{D}$

**for**  $i$  from 1 to  $k$  **do**

$f_i = A(\mathbb{D} \setminus \mathbb{D}_i)$

**for**  $z^{(j)}$  in  $\mathbb{D}_i$  **do**

$e_j = L(f_i, z^{(j)})$

**end for**

**end for**

**Return**  $e$

---