DATA

•

DATA

- Information about the problem to solve
- In the form of a distribution

$$p_{\text{data}}$$

- Classification and Regression:

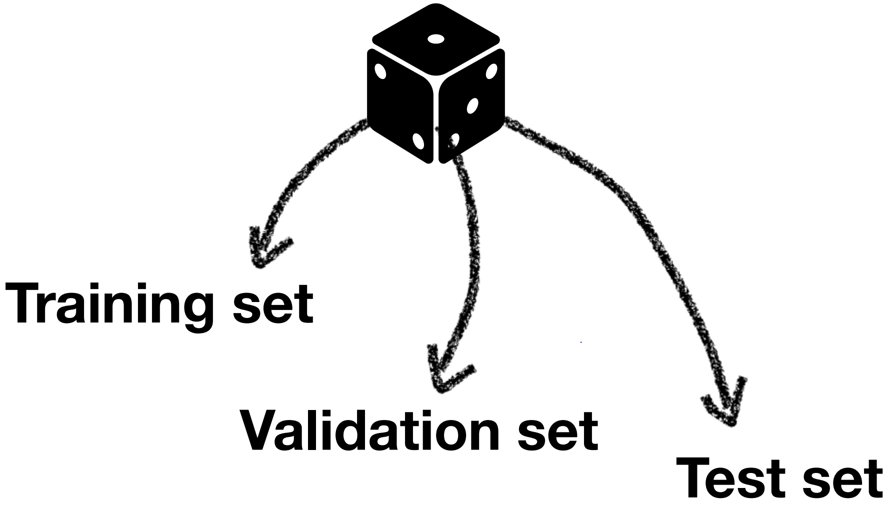
$$p_{\text{data}} \in \Delta(\mathscr{X} \times \mathscr{Y})$$

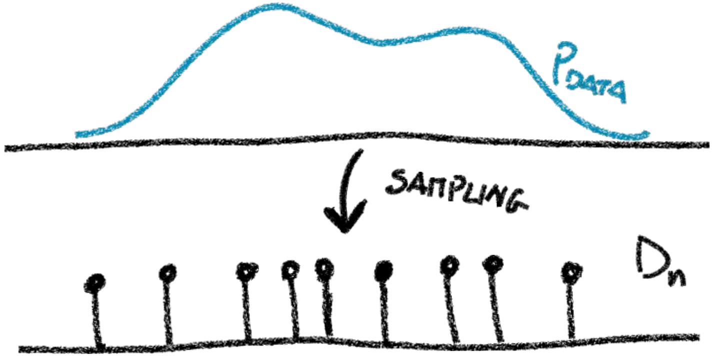
- Density estimation, Clustering and Dimensionality Reduction:

$$p_{\text{data}} \in \Delta(\mathscr{X})$$

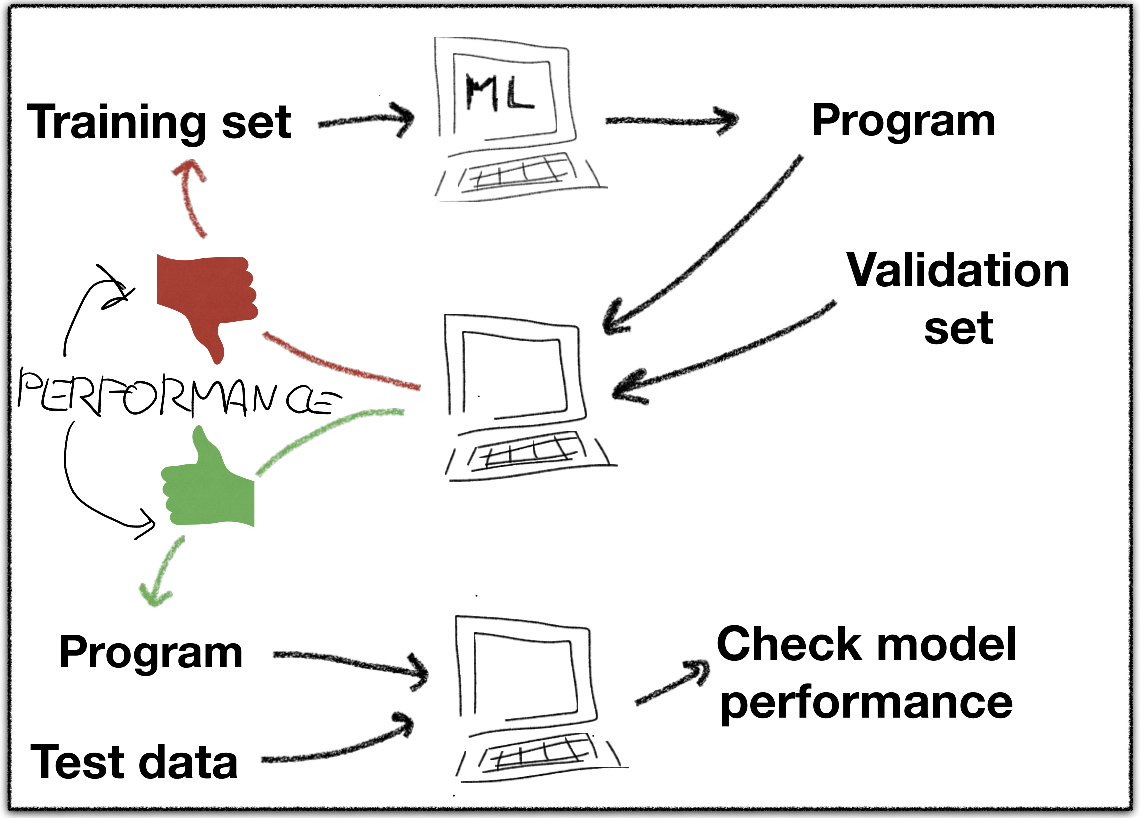
DATA

- The data distribution p_{data} is typically unknown
- But we can sample from it



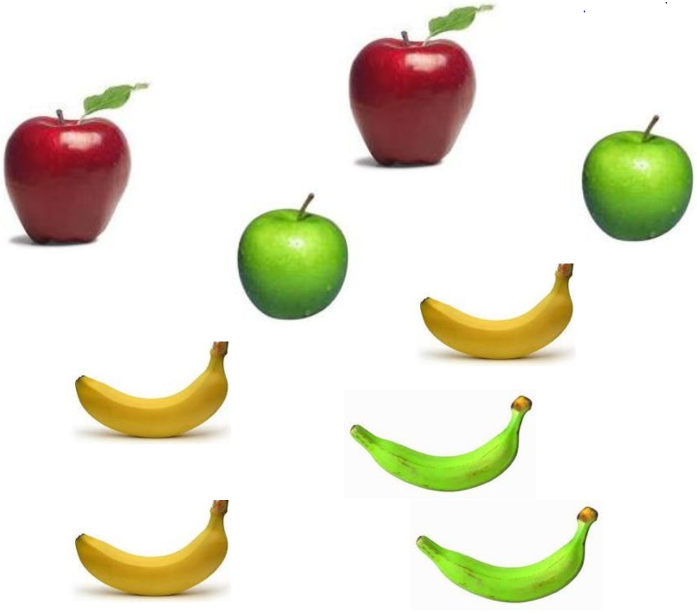


Training, Validation, Test sets

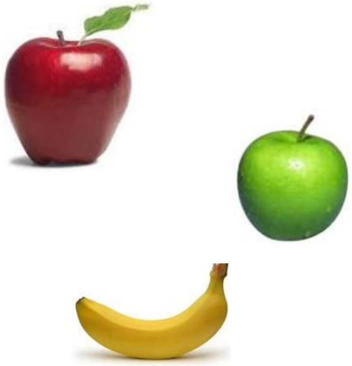


Training & Test Set

Training data

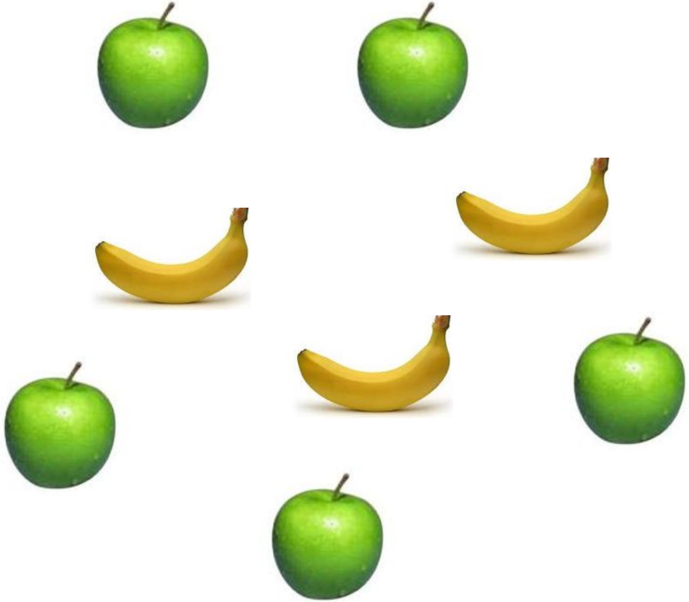


Test set

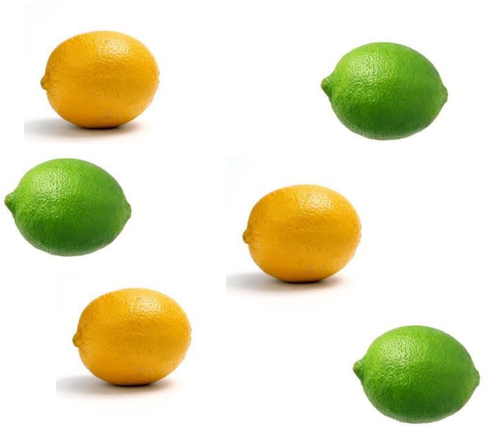


Training & Test Set

Training data



Test set

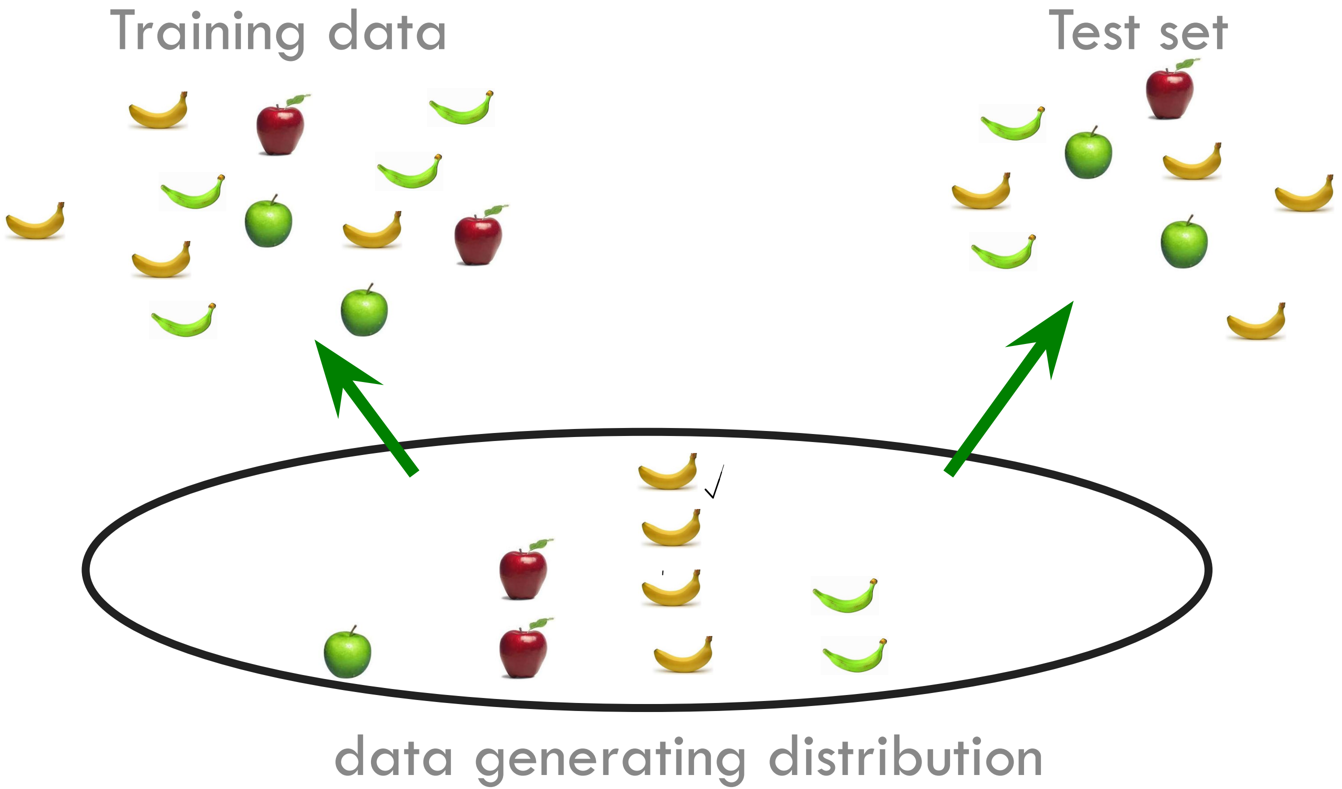


Not this one!

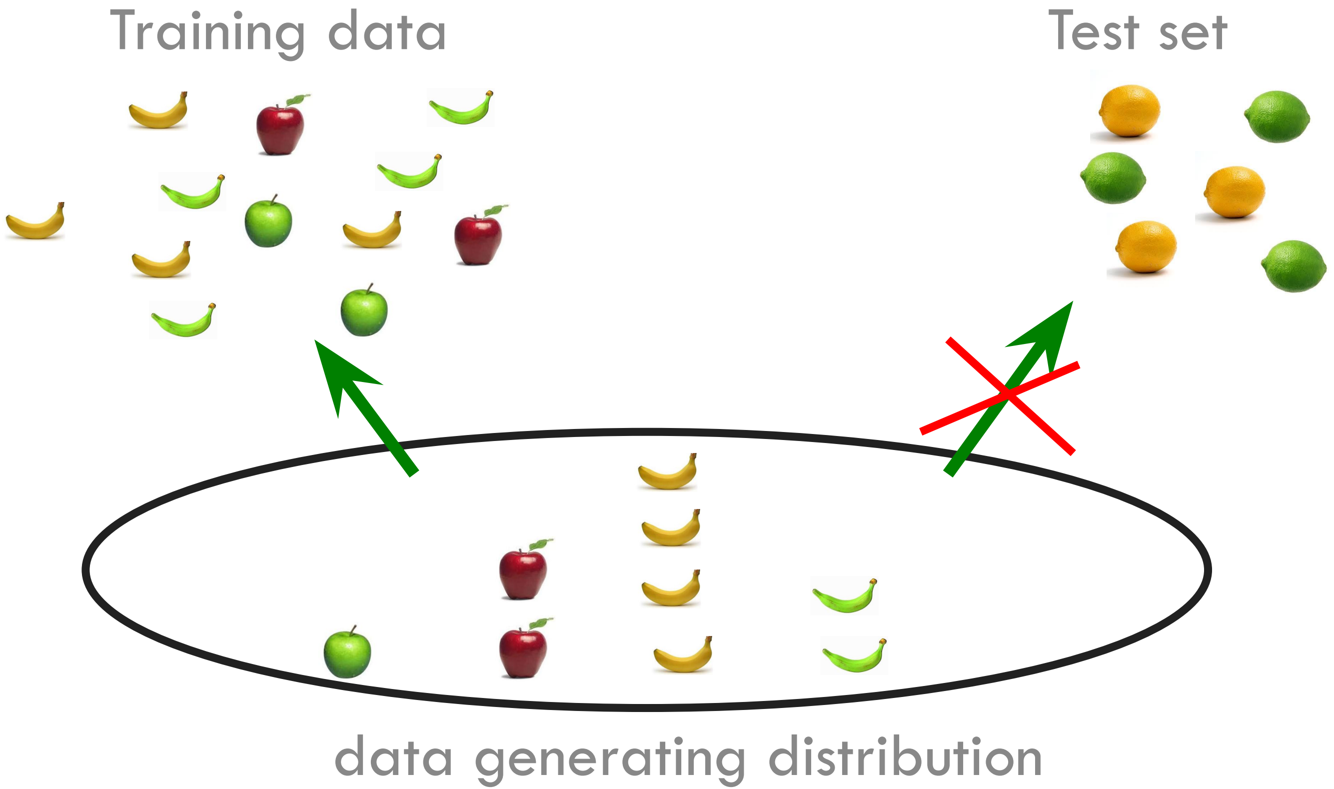
DATA GENERATING DISTRIBUTION

- We use a ***probabilistic model*** of learning
- There is some probability distribution over example/label pairs called the ***data generating distribution***
- **Both** the training data **and** the test set are generated based on this distribution

DATA GENERATING DISTRIBUTION



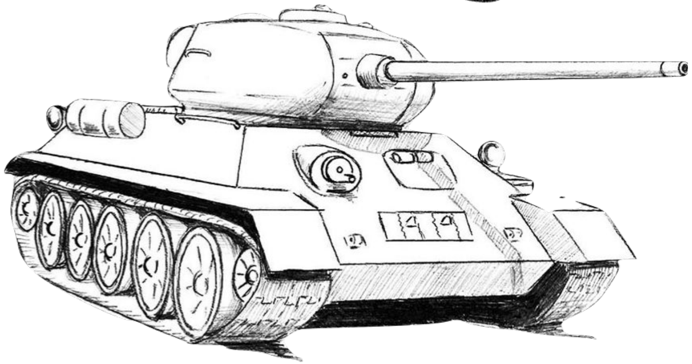
DATA GENERATING DISTRIBUTION



Training set design

- The failure of a machine learning algorithm is often caused by a bad selection of training samples
- The issue is that we might introduce unwanted correlations from which the algorithm derives wrong conclusions

"The US Army trained a program to differentiate American tanks from Russian tanks with 100% accuracy. Only later did analysts realized that the American tanks had been photographed on a sunny day and the Russian tanks had been photographed on a cloudy day. The computer had learned to detect brightness." [probably a legend]

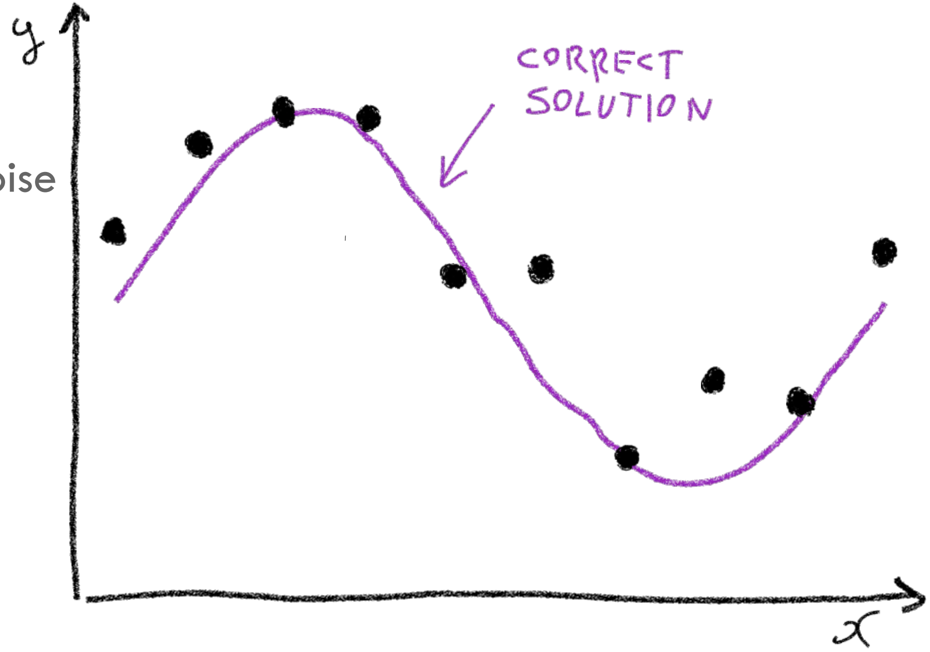


Example: polynomial curve fitting

Data

$$\mathscr{D}_n = \{(x_1, y_1), \ldots, (x_n, y_n)\}$$

- Data generated from $\sin(2\pi x)$ +noise
- Training set with $n = 10$ points



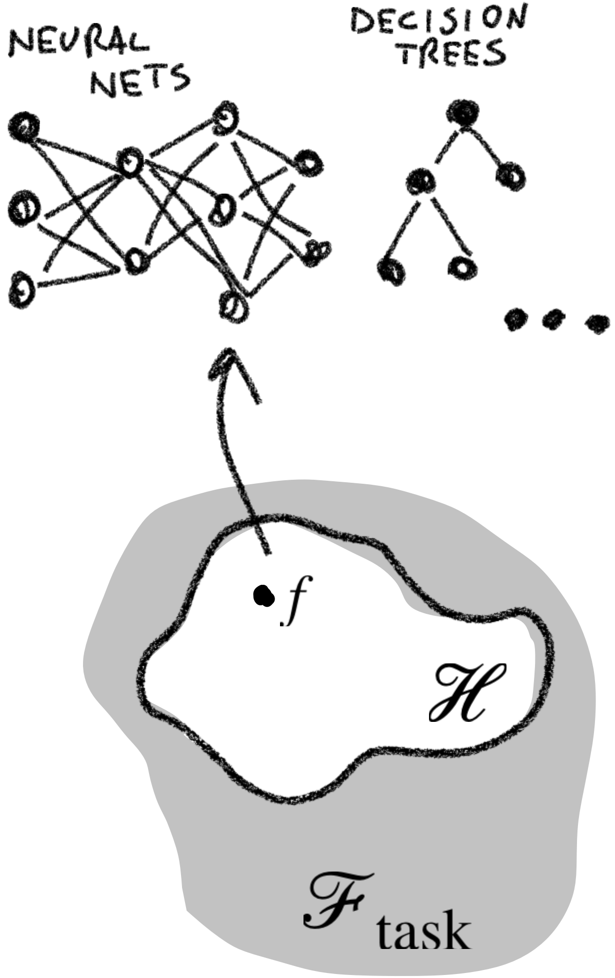
Model and Hypothesis space

DOMANDA TIPICA SU COS'È

- A model is like a “program” to solve the problem
- It is the implementation of a function $f \in \mathcal{F}_{\text{task}}$ that can be tractably computed
- A set of models forms an hypothesis space

$$\mathcal{H} \subset \mathcal{F}_{\text{task}}$$

- The learning algorithm seeks a solution within the hypothesis space



Example: polynomial curve fitting

Ho trasformato il problema. Il mio problema ora è quello di trovare i parametri corretti.
Utilizzo sempre la stessa semplice funzioni che ogni volta ha diversi parametri.
Calcolare il modello ora è calcolare i parametri.
Sarebbe impraticabile provare tutte le funzioni esistenti.
Devo restringere lo spazio di ricerca. Quindi riduco a trovare i parametri corretti.

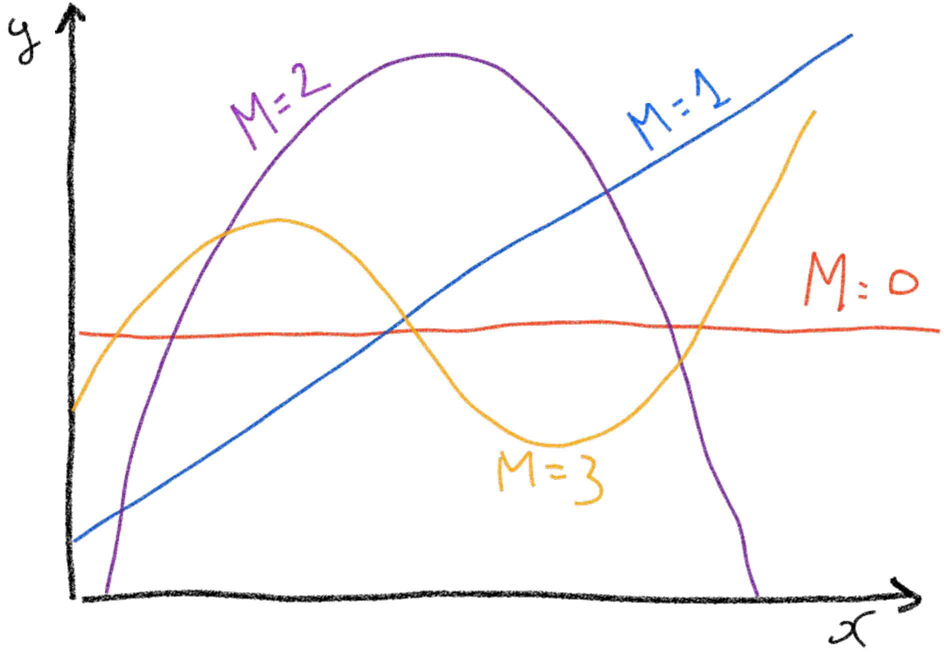
Model

$$f_w(x) = \sum_{j=0}^{M} w_j x^j$$

PARAMETERS $\{w_0, \dots, w_M\}$

$$\mathcal{H}_M = \{f_w : w \in \mathbb{R}^M\}$$

HYPOTHESIS SPACE FOR FIXED $M \in \mathbb{N}$

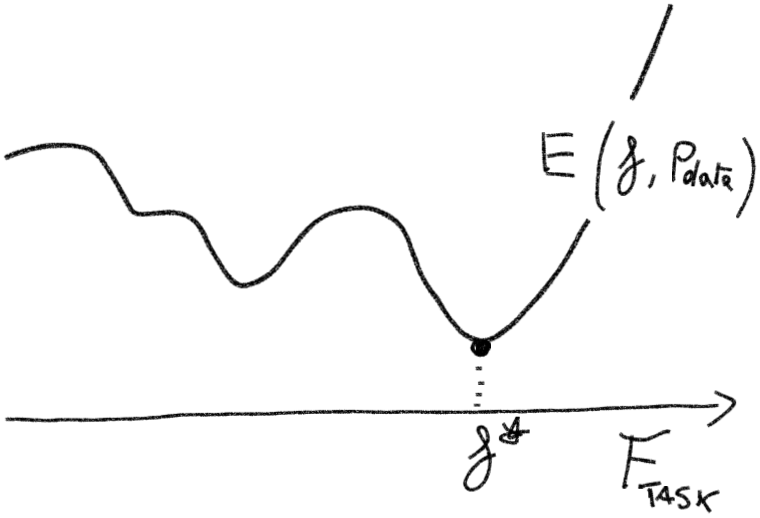


Objective - the ideal target

Impossibile cercare l'intero spazio.
Se potessimo farlo otterremmo la soluzione perfetta.
Faccio una generalizzazione.
Quello che trovo è un proxy della soluzione perfetta.
Es: curve fitting. Cercare il polinomio corretto mi da una curva simile. Ma non esattamente quella curva.

- Minimize a **(generalization) error function** $E(f; p_{\text{data}})$
- It determines how well a solution $f \in \mathscr{F}_{\text{task}}$ fits some given data
- Guides the selection of the best solution in $\mathscr{F}_{\text{task}}$

$$\boxed{f^\star \in \arg\min_{f \in \mathscr{F}_{\text{task}}} E(f; p_{\text{data}})}$$

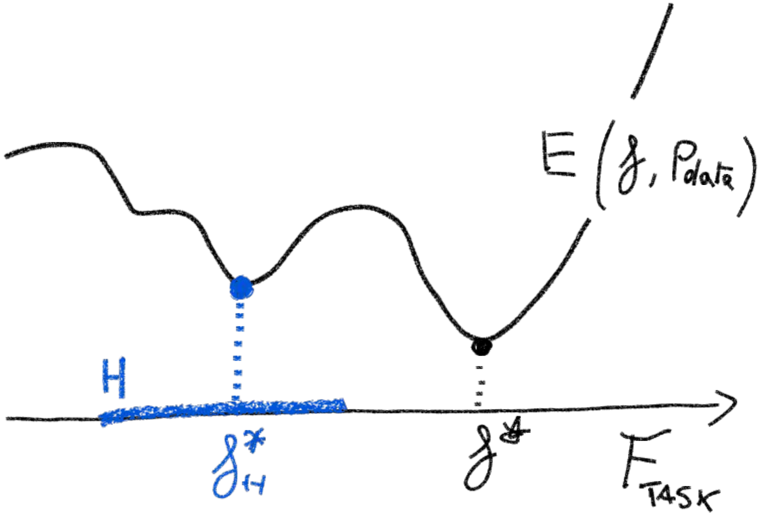


Too large search space and we need an implementation.

OBJECTIVE - THE FEASIBLE TARGET

- We need to restrict the focus on finding functions that can be implemented and evaluated in a tractable way
- We define a model hypothesis space $\mathscr{H} \subset \mathscr{F}_{\text{task}}$ and seek a solution within that space

$$\boxed{f^{\star}_{\mathscr{H}} \in \arg\min_{f \in \mathscr{H}} E(f; p_{\text{data}})}$$



Cannot be computed exactly, for p_{data} is unknown

Objective - The actual target

- We need to work on a data sample, i.e. a **training set** , $\mathcal{D}_n = \{z_1, \ldots, z_n\}$ where

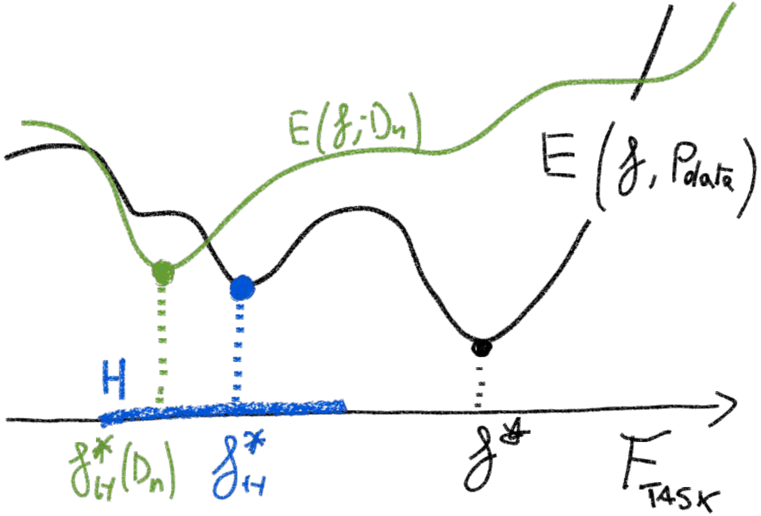
Non ho accesso alla reale distribuzione.
Il mio training set di per se è già una generalizzazione.
Aggiungo dell'errore rispetto alla perfezione durante il training.
Un training set grande ci fa avvicinare alla reale distribuzione.

$$z_i = (x_i, y_i) \in \mathcal{X} \times \mathcal{Y}$$

$$z_i \sim p_{\text{data}}$$

$$\boxed{f^\star_{\mathcal{H}}(\mathcal{D}_n) \in \arg\min_{f \in \mathcal{H}} E(f; \mathcal{D}_n)}$$

TRAINING ERROR



ERROR FUNCTION

Typically the generalization and training error functions can be written in terms of a **pointwise loss** $\ell(f;z)$ measuring the error incurred by f on the training example z

$$E(f;p_{\text{data}}) = \mathbb{E}_{z\sim p_{\text{data}}}[\ell(f;z)]$$

$$E(f;\mathscr{D}_n) = \frac{1}{n}\sum_{i=1}^{n}\ell(f;z_i)$$

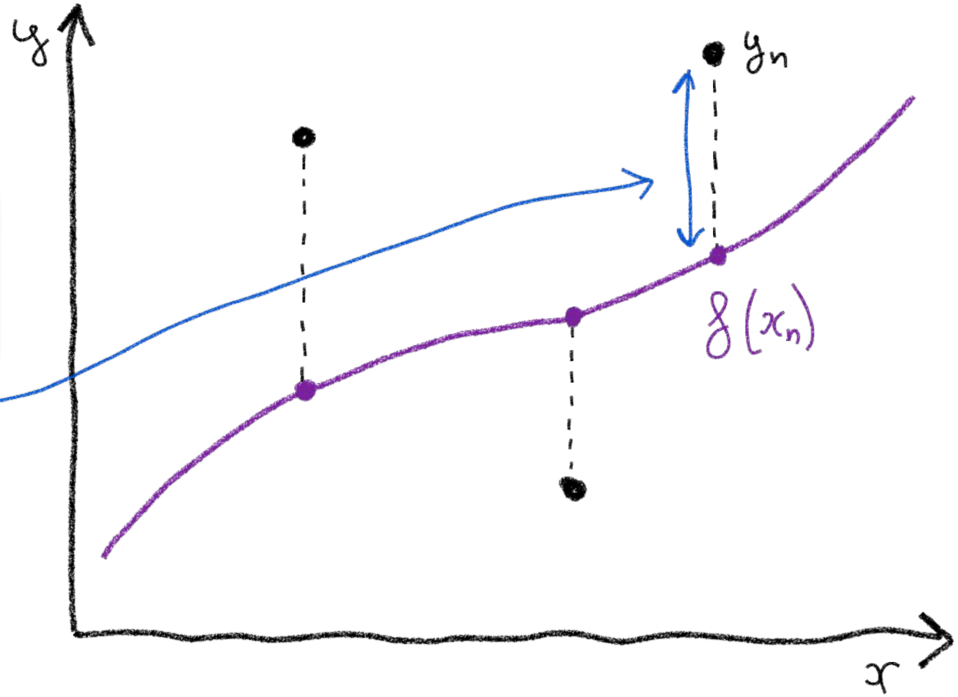
Example: polynomial curve fitting

Error function

$$E(f;\mathscr{D}_n) = \frac{1}{n}\sum_{i=1}^{n}[f(x_i) - y_i]^2$$

POINTWISE LOSS

$\ell(f;(x_i,y_i))$



Distanza della predizione dalla funzione reale per calcolare l'errore.

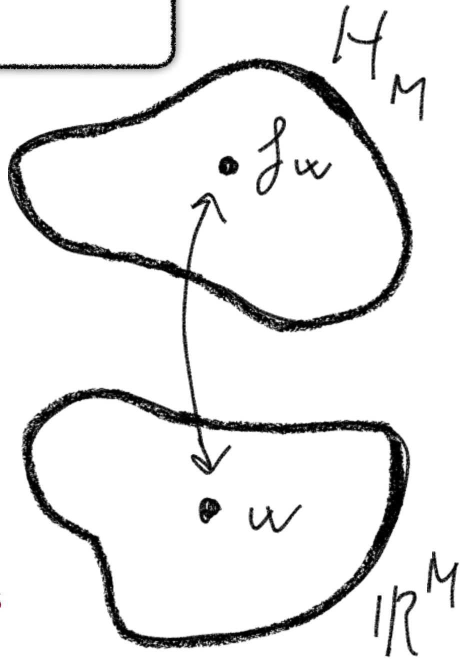
Example: polynomial curve fitting

Objective

$$f^\star_{\mathcal{H}_M}(\mathcal{D}_n) \in \arg\min_{f \in \mathcal{H}_M} E(f; \mathcal{D}_n)$$

equivalent to $f_{w^\star}$ where

$$w^\star \in \arg\min_{w \in \mathbb{R}^M} \frac{1}{n} \sum_{i=1}^{n} [f_w(x_i) - y_i]^2$$

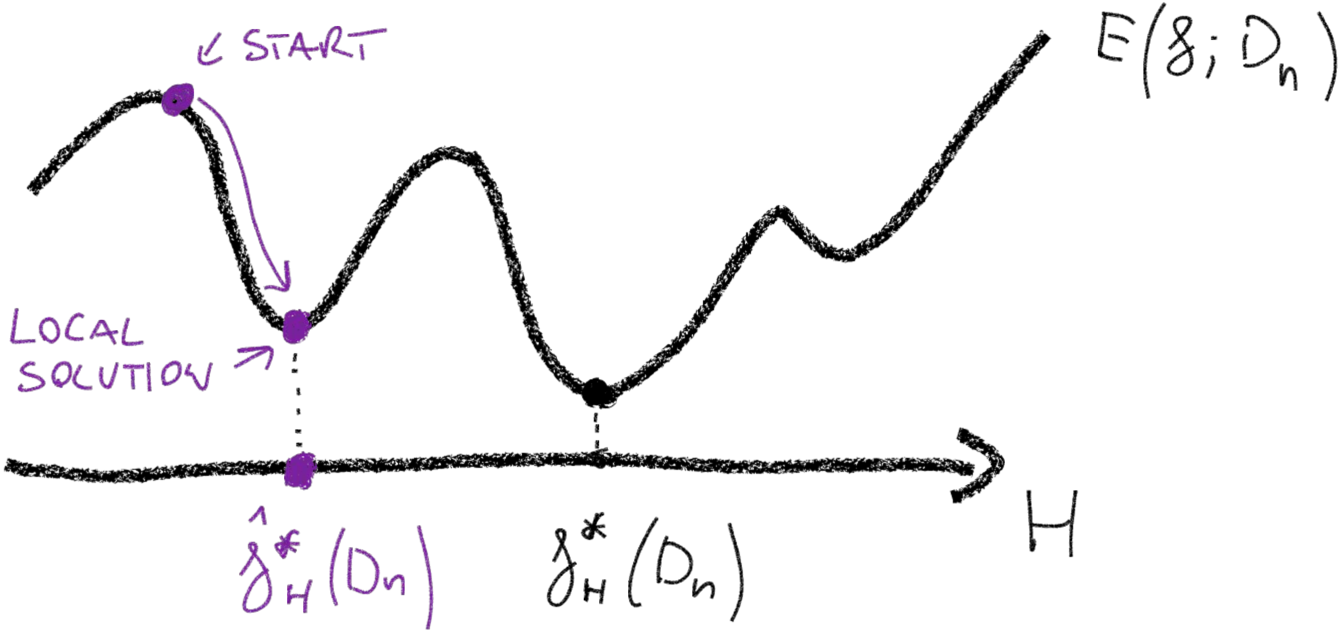


Requires solving a linear system of equations

Il problema diventa: trovare w vettore(parametri del polinomio) che minimizza l'errore

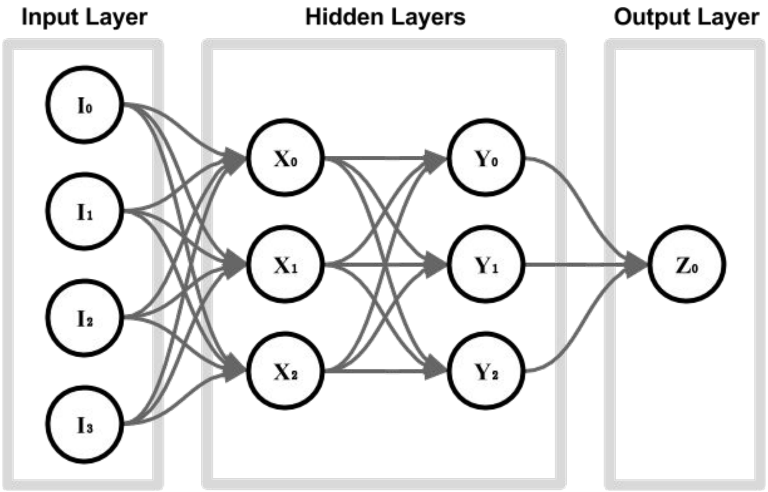
LEARNING ALGORITHM

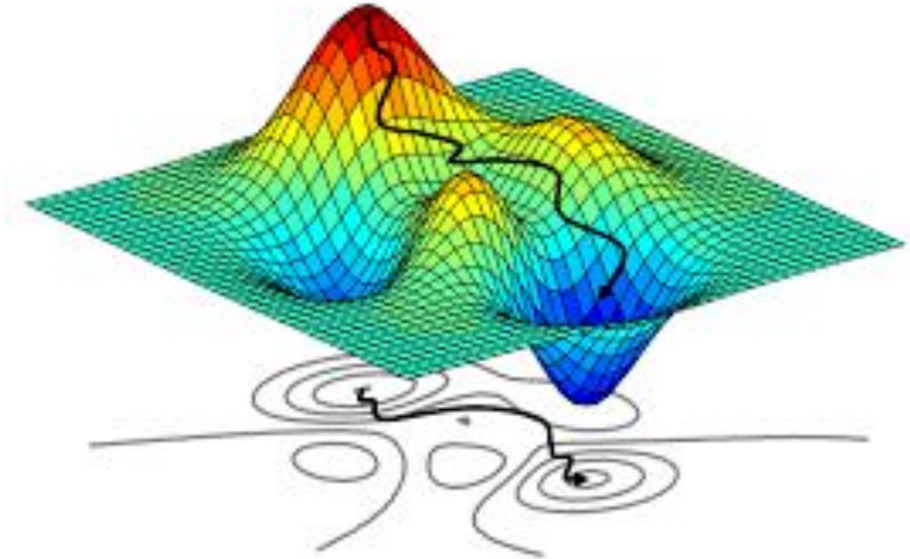
Solves the optimization problem targeting $f^{\star}_{\mathscr{H}}(\mathscr{D}_n)$ but might end up in a different result



Example: Neural Network Optimization

Issues with optimization: saddle points, local minima





RECAP

$$f^{\star}(\mathscr{D}_n) = \arg\min_{f \in \mathscr{F}_{\text{task}}} E(f; \mathscr{D}_n)$$

Minimizzare generalization error. voglio trovare f* che è l'ottimo.
Non voglio cercare tutto lo spazio ma solo Hypothesis space.
Non ho accesso alla distribuzione reale e mi devo accontentare di un proxy che è il mio training set.
Non lavoro sulla curva rossa ma su quella verde.