

# Introduzione Apprendimento supervisionato

Alberi decisionali, Random Forest



# INDICE DEI CONTENUTI

01

Classificazione  
automatica delle  
immagini

03

Addestrare un modello  
supervisionato

05

Random Forest

02

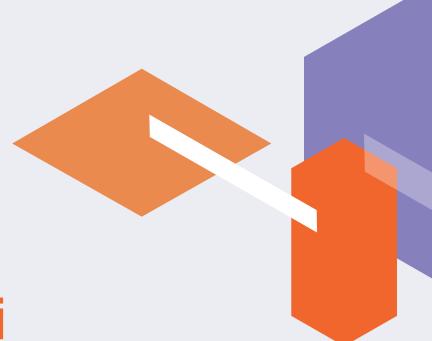
Modelli di  
apprendimento

04

Alberi decisionali

06

Python





01



Dai Big Data al Machine  
Learning

# CLASSIFICAZIONE AUTOMATICA DELLE IMMAGINI

# ORIGINE APPROFONDIMENTO

## Lezione dataset pazienti covid

> Completare dati mancanti



> Dati completi → previsioni sul decorso della malattia  
> In base a caratteristiche (es: comorbilità)



> Missforest (sottogruppo delle random forest)

...per i più disattenti

Link alla lezione

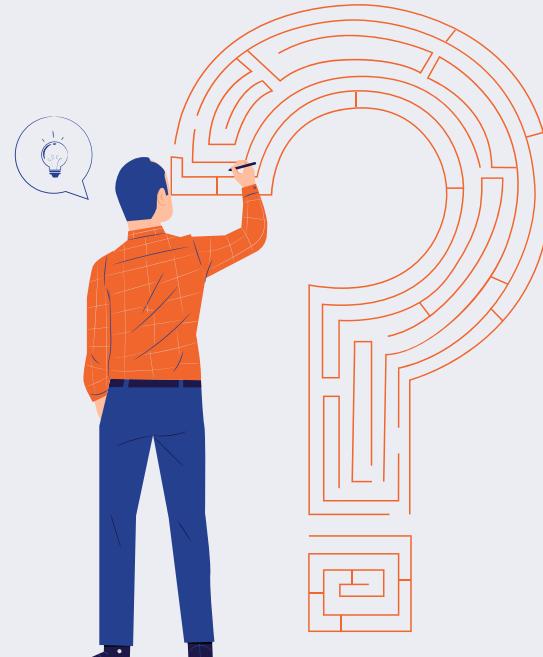
> [Cartella lezione 2](#)

*Importanza dei modelli di apprendimento nella manipolazione ed interpretazione di dati*

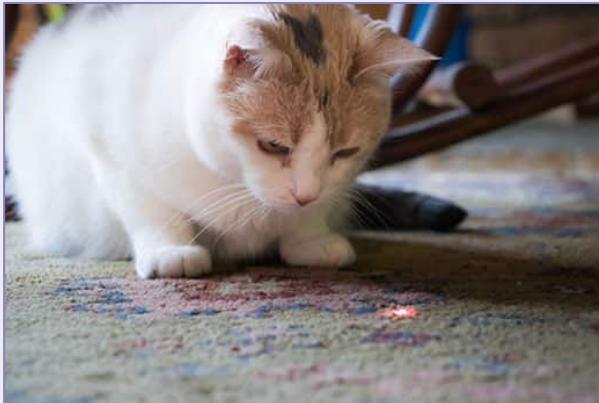
# BIG DATA

- > Enorme quantità di dati
- > Difficile orientarsi
- > Prima non c'erano e non c'era modo di recuperarli

## MACHINE LEARNING



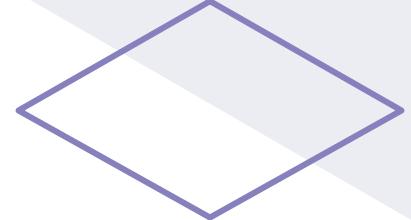
# EMULAZIONE PROCESSI DECISIONALI



...riconoscimento automatico

## Machine learning definition:

“Field of study that gives computers the ability to learn without being explicitly programmed”  
– Artur Samuel pioniere machine learning.



## DEFINIZIONE MACHINE LEARNING

“A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E.”

—**Tom Mitchell, Libro: Machine learning, McGraw Hill 1997**

# IMPARARE DALL'ESPERIENZA E VALUTAZIONE DELLE PERFORMANCE

Aspetti importanti nell'addestramento di un modello:

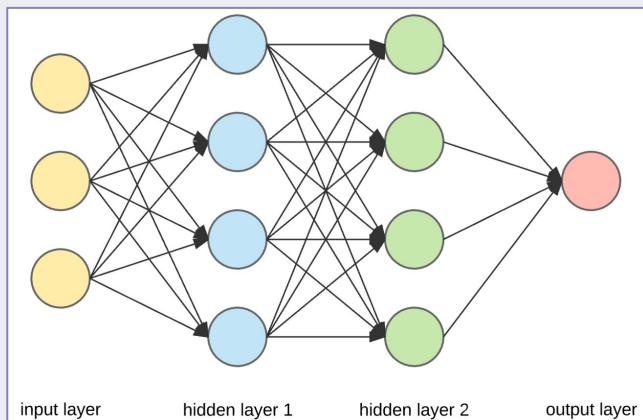
- L'esperienza deve essere significativa → importanza della scelta dell'input.
- Valutazione delle performance



# ESEMPIO DI MODELLO DECISIONALE

**Scoperta di Hubel e Wiesel**

Cellule semplici, complesse  
ed ipercomplesse



**Artificial neural networks**

Simulare la capacità del  
ragionamento umano



# 02

Tipologie

## MODelli DI APPRENDIMENTO



# APPRENDIMENTO SUPERVISIONATO

## > Etichette

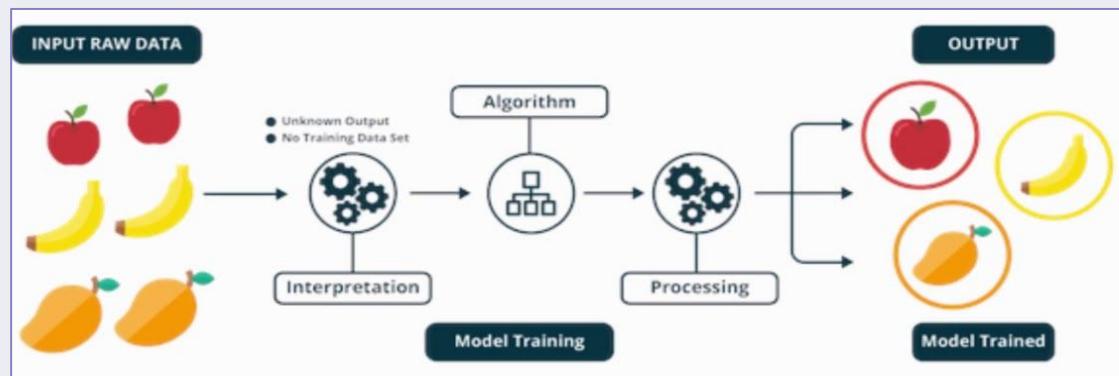
## > Fase di addestramento:

Input modello:

- valori da etichettare
- etichette corrette

Output modello:

- valori etichettati (ovvero suddivisi in gruppi)



# MODELLI DI APPRENDIMENTO SUPERVISIONATO

**Discreti  
(classificazione)**

Etichette discrete  
(numero finito)

Alberi di classificazione

**Continui  
(regressione)**

Etichette continue  
(dominio numeri reali)

Alberi di regressione

# APPRENDIMENTO NON SUPERVISIONATO

Nessuna conoscenza sui dati in input

Input modello:

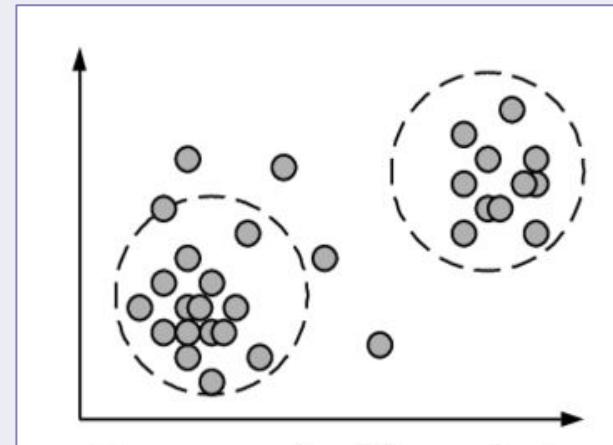
- > valori privi di significato

Output modello:

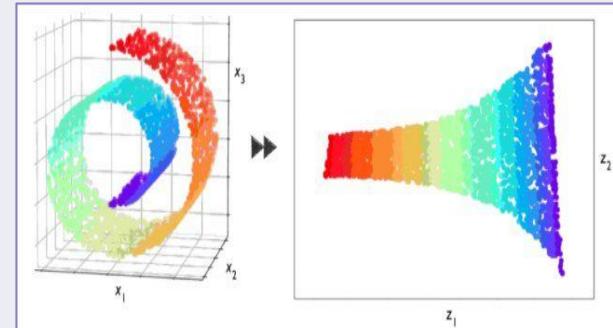
- > valori con significato  
(secondo una metrica x di valutazione)

- > No feedback diretto

Clustering



Riduzione dimensione del problema



# APPRENDIMENTO RINFORZATO

Nessuna conoscenza sui dati in input

Input modello:

- > valori privi di significato

Output modello:

- > tentativo numero N

Feedback

- > positivo: azione più frequente
- > negativo: azione meno frequente



Robotica

# CONFRONTO TRA TIPOLOGIE DI APPRENDIMENTO

## SUPERVISIONATO

- > Conoscenza sui dati in input



## NON SUPERVISIONATO

- > No conoscenza sui dati in input



## RINFORZATO

- > No conoscenza sui dati in input



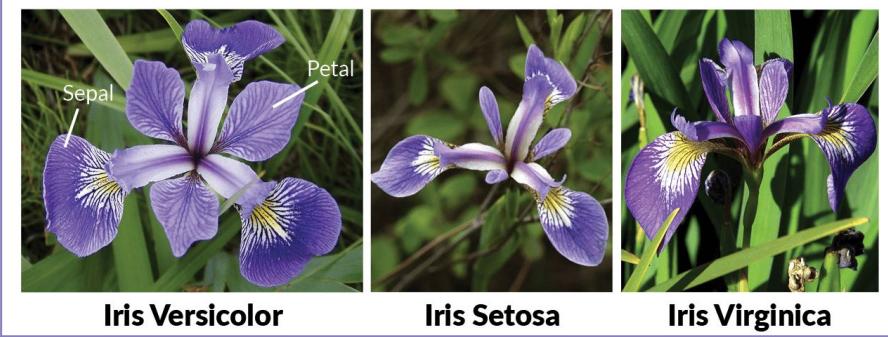


03

# ADDESTRARE UN MODELLO SUPERVISIONATO

# DATASET DI IRIS

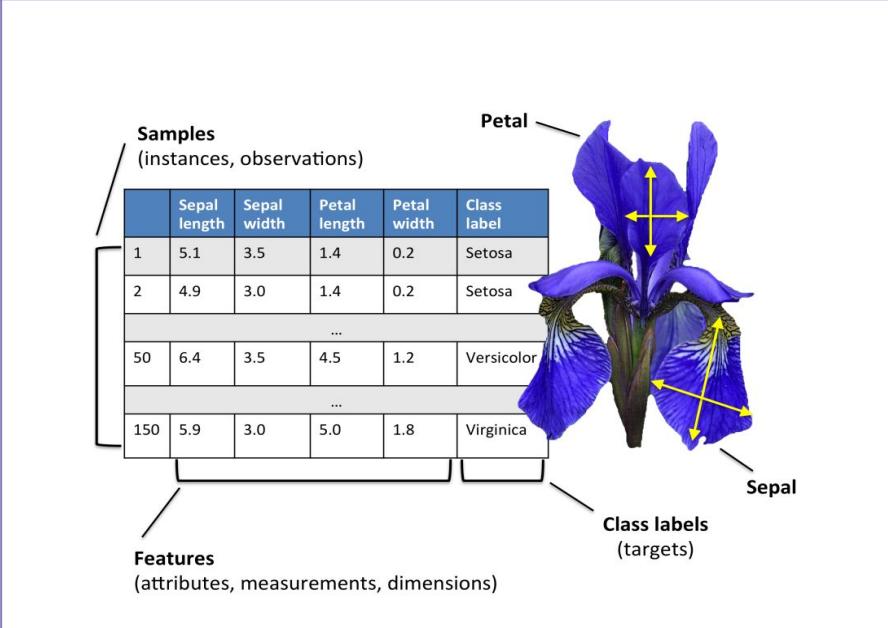
- > 150 istanze
- > Etichette: specie di appartenenza
- > Features / attributi: lunghezza e larghezza petali e sepali



Iris Versicolor

Iris Setosa

Iris Virginica



# DATASET COME MATRICE

$N \times M$

$N$ : n-esima feature (attributo immagine)

$M$ : numero di riga del matrice (numero d'istanza nel dataset)

> Ogni istanza un vettore a riga di 4 dimensioni (solo attributi)

> Ogni etichetta corretta fa parte di un vettore colonna

$$\begin{bmatrix} x_1^{(1)} & x_2^{(1)} & x_3^{(1)} & x_4^{(1)} \\ x_1^{(2)} & x_2^{(2)} & x_3^{(2)} & x_4^{(2)} \\ \vdots & \vdots & \vdots & \vdots \\ x_1^{(150)} & x_2^{(150)} & x_3^{(150)} & x_4^{(150)} \end{bmatrix}$$

$$\mathbf{x}^{(i)} = \begin{bmatrix} x_1^{(i)} & x_2^{(i)} & x_3^{(i)} & x_4^{(i)} \end{bmatrix}$$

$$\mathbf{x}_j = \begin{bmatrix} x_j^{(1)} \\ x_j^{(2)} \\ \vdots \\ x_j^{(150)} \end{bmatrix}$$



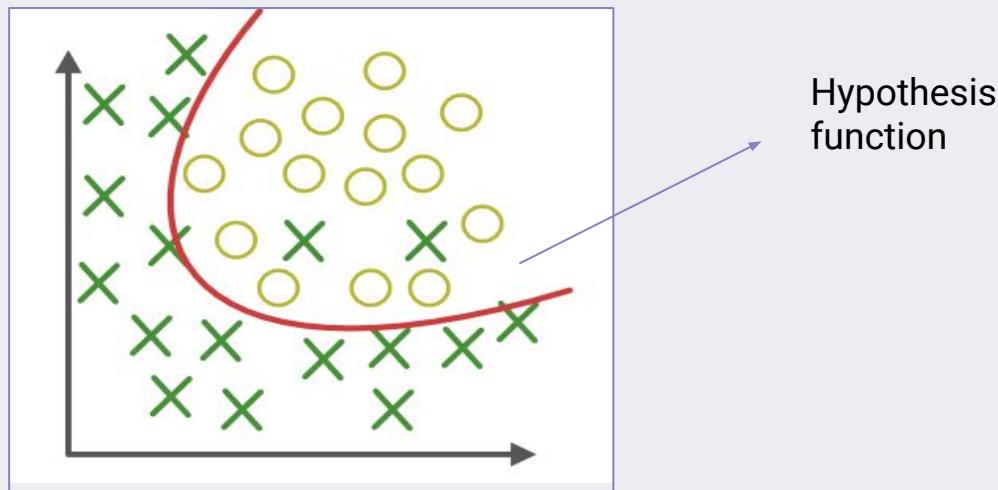
# IL MODELLO DI APPRENDIMENTO: LA FUNZIONE DI IPOTESI (HYPOTHESIS FUNCTION)

Obiettivo:

Trovare un **modello** che dato un **dataset** associa **etichette corrette**

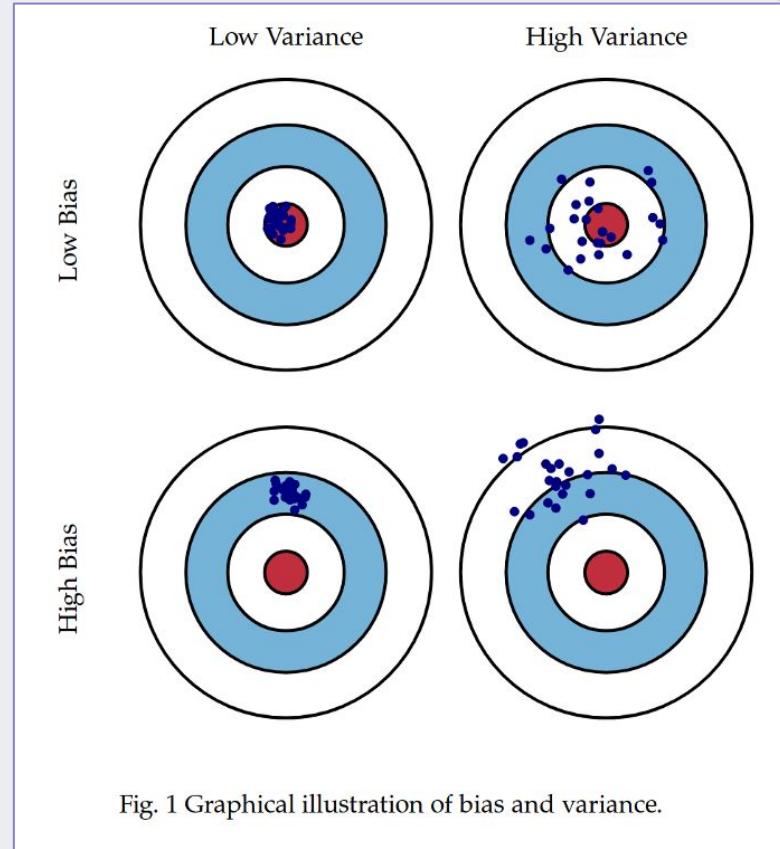
Vogliamo trovare una funzione  $h(x)$  tale che per ogni  $x$  ci dia un valore  $y$

Confronto tra  $y$  ed  $\hat{y}$



# COMPROMESSO TRA BIAS E VARIANZA

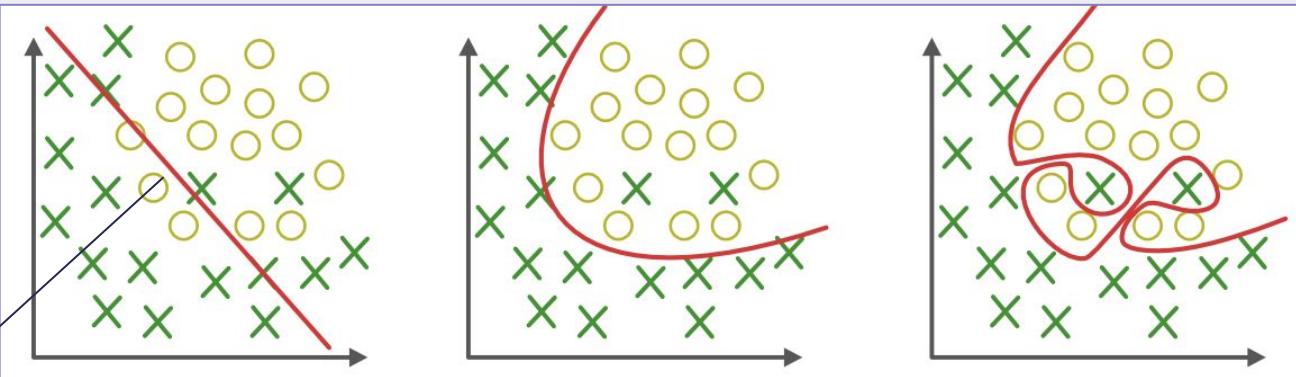
- > **Varianza:** alta sensibilità alla randomness dei dataset di training
- > **Bias:** errore sistemico



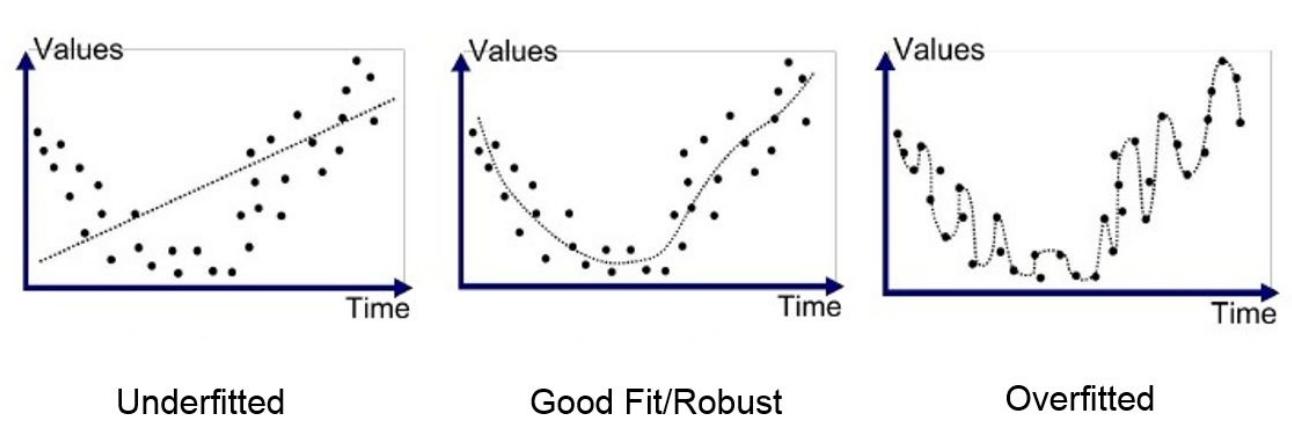
# OVERFITTING ED UNDERFITTING



- > Underfitting: alto bias
- > Overfitting: alta varianza



Iperpiano separatore



Underfitted

Good Fit/Robust

Overfitted



**04**

**ALBERI  
DECISIONALI**

# CHE COS'È UN ALBERO DECISIONALE?



Grafo di decisioni  
e delle loro possibili  
conseguenze



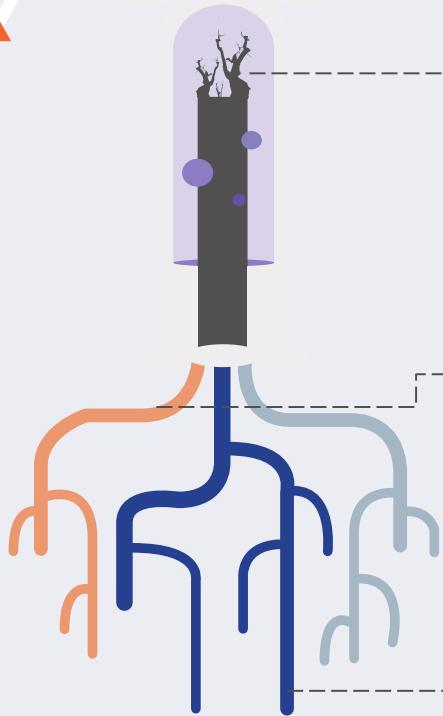
Utilizzato per creare un  
**piano di azioni**  
mirato ad uno scopo



Chiamato anche  
**albero predittivo**



# COMPOSIZIONE ALBERO DI DECISIONE



**RADICI**

Punto più alto da dove si diramano le possibilità di scelta



**ARCHI**

Segmenti che collegano la radice ai nodi più interni



**FOGLIE**

Rappresentano domande diverse e le decisioni prese



# FUNZIONAMENTO DI UN ALBERO DECISIONALE

Si hanno:

***n variabili in input***

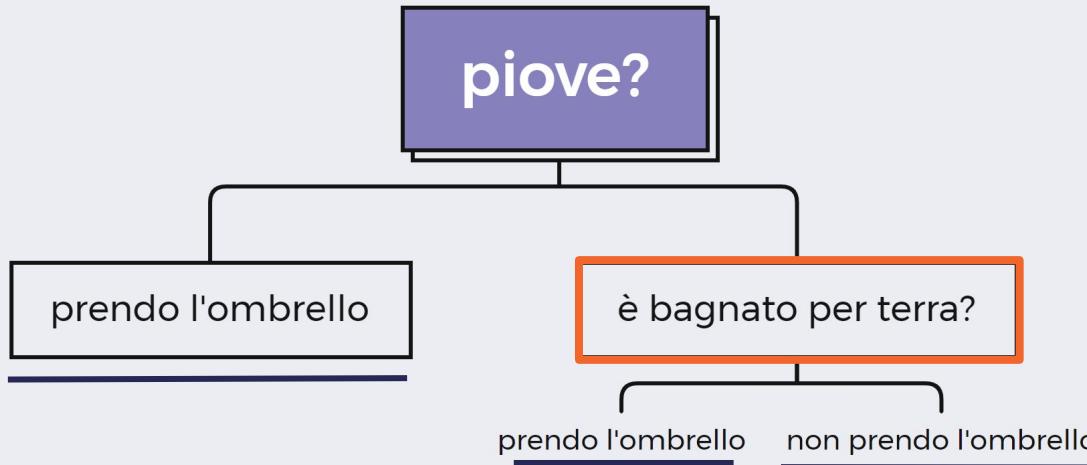


Derivano dall'osservazione

***m variabili in output***



Identificano la decisione/azione da intraprendere



Si parte dalla **radice** e si procede verso il basso

Ogni **nodo** indica una condizione di test

Man mano che procediamo lo spazio delle ipotesi si riduce

La **decisione finale** si trova nei nodi foglia terminali

# VARIABILI UTILIZZATE



## VARIABILI DISCRETE

Le variabili hanno valori **numerici interi**.  
In questo caso si parla di **classificazione**

## VARIABILI CONTINUE

Le variabili hanno valori **numerici reali**.  
In questo caso si parla di **regressione**

# ESEMPIO

## DI ALBERO DECISIONALE A VARIABILI DISCRETE



**OBIETTIVO:** prevedere la probabilità di sopravvivenza di un passeggero del titanic

**CONCLUSIONE:** si sopravvive se si è donna o se si è un uomo con età inferiore ai 9,5 anni con meno di 2,5 fratelli



# COME SI COSTRUISCE UN ALBERO DI DECISIONE

Una macchina **apprende dall'esperienza** se riesce a costruire da sè un albero decisionale **in base ai dati osservati**





# ESEMPIO DI APPRENDIMENTO

Informazioni derivanti dall'**osservazione**



Variabile in **uscita**



I dati costituiscono l'**insieme di addestramento**: *training set*

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	0	0	1	0	0	1	1	0	0	1
1	0	1	1	0	0	1	1	0	0	0
0	1	1	0	1	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1
1	0	1	1	1	0	1	0	0	1	0
0	0	1	1	0	1	1	0	1	1	1
0	0	1	1	0	0	0	0	0	0	0
1	1	1	0	1	1	1	0	1	0	1

Nota: in questo esempio non è stato fornito un albero di decisione, ma un set di dati. Si dovrà quindi analizzare gli esempi per trovare una correlazione tra i dati.

Questo viene fatto con le **funzioni di verità**





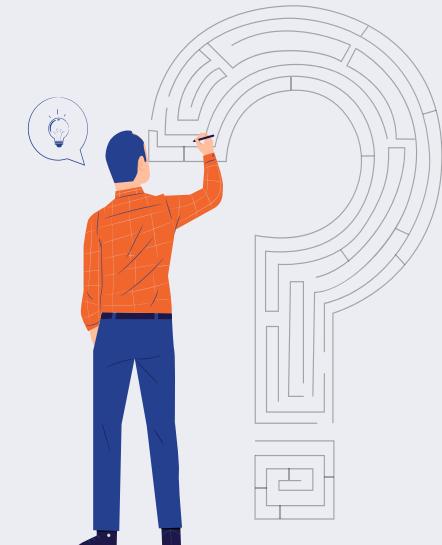
**ATTENZIONE!** Ogni funzione di verità è un **albero decisionale**

Dall'esempio:  $10! = 3.628.800$  alberi



Emergono **due problemi**:

1. Quale è l'albero decisionale migliore?
2. Come trovarlo?



# FASI PER LA COSTRUZIONE



## COSTRUZIONE

Scelta dei fattori più efficaci per ottenere un risultato finale ottimale



## PRUNING

Ottimizzazione dell'albero

## Come trovare l'albero di decisione migliore?

Soluzione - si devono individuare i **fattori più importanti** ovvero quelli che **influiscono maggiormente sul risultato finale**

Obiettivo - **eliminazione** dei rami inutili

**Algoritmi automatici per la ricerca dell'attributo migliore**

- **INDICE DI SHANNON - WIENER (ENTROPIA)**
- **INDICE DI GINI**
- **INFORMATION GAIN**

## Un po' di formule...

È utilizzato per **individuare la caratteristica più utile** su cui costruire un albero di decisione

$$H(m) = \sum_k P(m_k) * \log_2 \left( \frac{1}{P(m_k)} \right) = - \sum_k P(m_k) * \log_2(P(m_k))$$

Si calcola l'efficacia della risposta riguardo a Y fornita da x:

- se  $H$  tende a 0, informazione **molto utile**
- se  $H$  tende a 1, informazione **inutile**

**Attributo informativo:**

$$V(m) = 1 - H(m)$$

## Riprendiamo l'esempio

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	0	0	1	0	0	1	1	0	0	1
1	0	1	1	0	0	1	1	0	1	0
0	1	1	0	1	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1
1	0	1	1	1	0	1	0	0	1	0
0	0	1	1	0	1	1	0	1	1	1
0	0	1	1	0	0	0	0	0	0	0
1	1	1	0	1	1	1	0	1	0	1

Per ogni esempio viene rilevato lo stato di dieci caratteristiche X e l'esito finale Y

### Analisi della caratteristica x1

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	y
1	0	0	1	0	0	1	1	0	0	1
1	0	1	1	0	0	1	1	0	1	0
0	1	1	0	1	0	0	0	0	0	0
1	0	0	0	0	1	1	1	1	1	1
1	0	1	1	1	0	1	0	0	1	0
0	0	1	1	0	1	1	0	1	1	1
0	0	1	1	0	0	0	0	0	0	0
1	1	1	0	1	1	1	0	1	0	1

## COSTRUZIONE - Entropia

10 ESEMPI

x1	y
1	1
1	0
0	0
1	1
1	0
0	1
0	0
1	1

Probabilità che ci sia una relazione tra x1 e y

$$P(m1) = \frac{3}{5} = 0,6$$

$$\overline{P(m1)} = \frac{2}{5} = 0,4$$

10 ESEMPI

x1	y
1	1
1	0
0	0
1	1
1	0
0	1
0	0
1	1

Probabilità che non ci sia una relazione tra x1 e y

$$P(xi) = \frac{5}{10} = 0,5$$

Probabilità che si verifichi la caratteristica x

$$H(m1) = I\left(\frac{3}{5}, \frac{2}{5}\right)$$

L'entropia per la caratteristica  $x1$  è quindi:

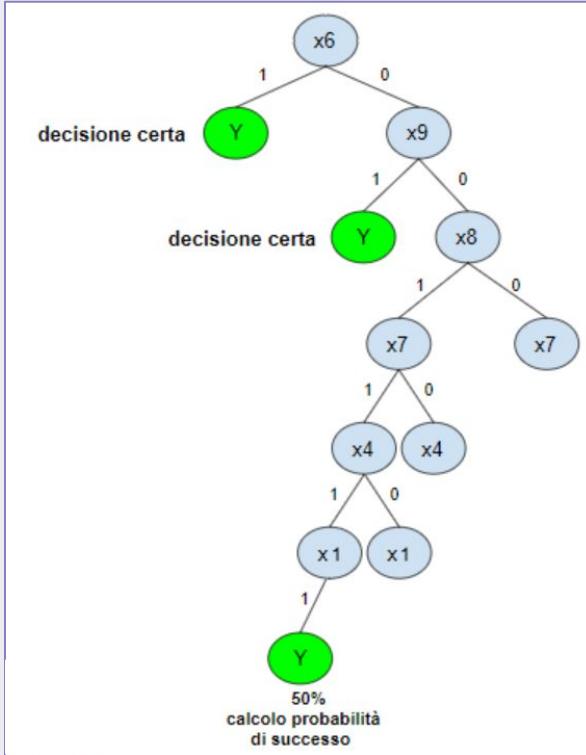
$$\begin{aligned} H(m1) &= I\left(\frac{3}{5}, \frac{2}{5}\right) = \\ &= -\frac{3}{5} \log_2\left(\frac{3}{5}\right) - \frac{2}{5} \log_2\left(\frac{2}{5}\right) = \\ &= -\frac{3}{5} * (-0.73) - \frac{2}{5} * (-1.32) = \\ &= 0.438 + 0.528 = \\ &= 0.966 \end{aligned}$$

**Conclusione:**  $x1$  non è l'attributo migliore per iniziare a costruire l'albero.  
Si ripete il calcolo per le altre caratteristiche

## COSTRUZIONE - Entropia

x	p(x)	p(y)	H
x1	0.5	0,60	0,97
x2	0.2	0,50	1,00
x3	0.6	0,33	0,91
x4	0.5	0,40	0,97
x5	0.3	0,33	0,91
x6	0.3	1,00	0,00
x7	0.6	0,57	0,99
x8	0.3	0,66	0,92
x9	0.3	1,00	0,00
x10	0.4	0,50	1,00

bassa diversità



$x_6$  e  $x_9$  sono gli attributi più promettenti per costruire un albero efficiente e meno profondo

## Riassumiamo

**Obiettivo:**

avere dei nodi puri ovvero contenenti solo  
istante di dati che appartengono ad una sola  
classe

**Strategia:**

ottimizzazione dell'albero per minimizzare il  
livello di entropia

→ l'entropia determina le condizioni di spliti  
ottimali



È la probabilità che una variabile, scelta casualmente, sia **classificata erroneamente**

$$Gini = 1 - \sum_{i=1}^k P(i)^2$$

Probabilità di un oggetto che venga classificato in una particolare classe

Il grado di Gini varia tra 0 e 1 dove:

- **0**, indica che tutti gli elementi sono state inseriti in una certa classe o esiste una sola classe
- **0.5**, indica che gli elementi sono distribuiti uniformemente in alcune classi
- **1**, indica che tutti gli elementi sono distribuiti casualmente tra le varie classi

Più l'indice di Gini è minimo, più quel nodo è adatto come radice

È il grado di **informazione** associato ad ogni attributo  
→ si cerca l'attributo migliore che restituisce il massimo delle informazioni su una classe

**Ma non è l'entropia...**

L'Information Gain di S relativamente all'attributo A è la riduzione di entropia dovuta allo splitting su A

***Information Gain = entropia prima della divisione – entropia dopo la divisione***

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

Calcola la differenza tra l'entropia prima e dopo la divisione e specifica l'impurità degli elementi della classe:

determina quante informazioni si sono ottenute facendo la divisione

**Formula**

***Information Gain = entropia prima della divisione – entropia dopo la divisione***

$$Gain(S, A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

**S** → insieme di esempi presi in considerazione

**A** → attributo che si vuole dividere

**|S|** → numero di esempi

**|S<sub>v</sub>|** → numero di esempi per il valore corrente dell'attributo A

Maggiore è l'information Gain, migliore sarà il nodo su cui costruire l'albero di decisione

## ESEMPIO

Una scatola contiene:



25 cioccolatini rossi



15 snickers



10 kit kats



25 cioccolatini blu



5 snickers

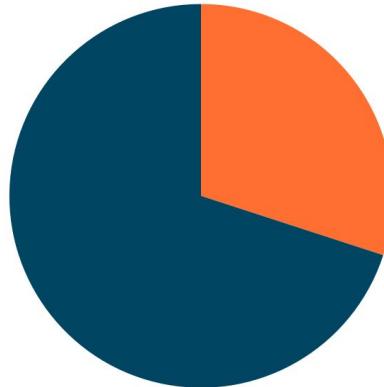


20 kit kats

Supponiamo di voler mangiare solo **Snickers rossi**:

Cioccolatini

● Snickers Rossi (esempio positivo) ● Altri Cioccolatini (esempio negativo)



L'entropia in questo caso:

$$\begin{aligned} \text{entropy} &= -\frac{15}{50} \log_2 \left( \frac{15}{50} \right) - \frac{35}{50} \log_2 \left( \frac{35}{50} \right) = \\ &= 0.5210 + 0.3602 \\ &= 0.8812 \end{aligned}$$

Per costruire l'albero dobbiamo selezionare uno dei due attributi (*colore o marca*) per il nodo radice.

## Come scegliere l'attributo migliore?

Sceglieremo l'attributo con il più alto guadagno di informazioni

**Information Gain** - rispetto al colore

$$\begin{aligned} \text{Information Gain}(\text{Chocolates}, \text{Colors}) &= \\ &\quad \text{Entropy}(\text{Chocolates}) \\ &- \left( \frac{|\text{red chocolates}|}{|\text{total chocolates}|} * \text{Entropy}(\text{red chocolates}) \right) \\ &- \left( \frac{|\text{blue chocolates}|}{|\text{total chocolates}|} * \text{Entropy}(\text{blue chocolates}) \right) \end{aligned}$$

$$\text{entropy}(\text{red chocolates}) = -\frac{15}{25} \log_2 \left( \frac{15}{25} \right) - \frac{10}{25} \log_2 \left( \frac{10}{25} \right) = 0.9709$$

$$\text{entropy}(\text{blue chocolate}) = 0$$

→ l'entropia è pari a 0 perchè non sono i cioccolatini che vogliamo mangiare

$$\begin{aligned}\text{Information Gain}(\text{chocolates}, \text{colors}) &= 0.8812 - \left( \frac{25}{50} * 0.9709 \right) - \left( \frac{25}{50} * 0 \right) \\ &= 0.3958\end{aligned}$$

**Information Gain** - rispetto al marchio

$$\begin{aligned} \text{Information Gain}(\text{Chocolates}, \text{Brand}) &= \\ &\quad \text{Entropy}(\text{Chocolates}) \\ &\quad - \left( \frac{|\text{Snikers}|}{|\text{total chocolates}|} * \text{Entropy}(\text{Snikers}) \right) \\ &\quad - \left( \frac{|\text{Kit Kats}|}{|\text{total chocolates}|} * \text{Entropy}(\text{Kit Kats}) \right) \end{aligned}$$

$$\text{Entropy}(\text{Snikers}) = -\frac{15}{20} \log_2 \left( \frac{15}{20} \right) - \frac{5}{20} \log_2 \left( \frac{5}{20} \right) = 0.8112$$

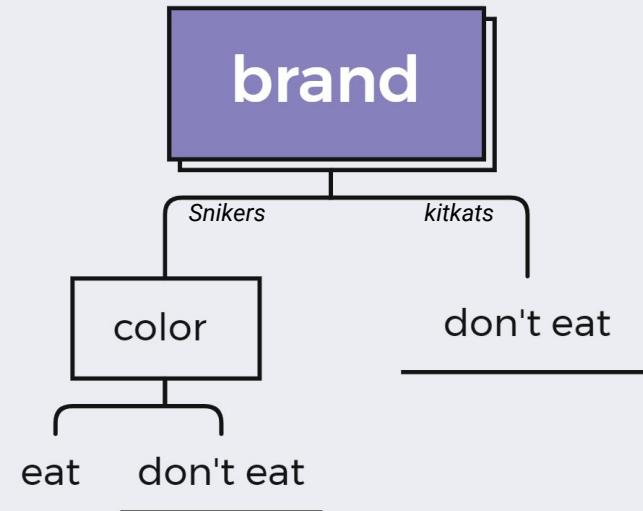
$$\text{Entropy}(\text{Kit Kats}) = 0$$

$$\begin{aligned} \text{Information Gain (chocolates, brand)} &= 0.8812 - \left( \frac{20}{50} * 0.8112 \right) - \left( \frac{30}{50} * 0 \right) \\ &= 0.5567 \end{aligned}$$

**CONCLUSIONI** - poichè l'information Gain rispetto alla marca è maggiore di quello rispetto al colore, la radice dell'albero partira dal **marchio**

Per il livello successivo rimane solo il colore, possiamo facilmente suddividere in base al colore senza fare calcoli.

Il nostro albero avrà questo aspetto:



# CONCLUSIONI

- Il guadagno informativo determina la **riduzione dell'incertezza** dopo aver suddiviso il set di dati su una particolare caratteristica
  - ◆ Se il valore del guadagno aumenta, la caratteristica scelta sarà più utile per la classificazione
- La caratteristica con il valore più alto di information gain è considerata la migliore caratteristica da scegliere per la suddivisione
- Utilizzato in algoritmi con **ID3**

# PERCHÈ POTARE UN ALBERO

In molte situazioni bisogna definire:

**CRITERIO DI ARRESTO**

*halting*

**CRITERIO DI POTATURA**

*pruning*

Una **crescita eccessiva** della dimensione dell'albero potrebbe portare ad un  
**aumento della complessità computazionale**

Questo problema è detto **OVERFITTING**

# DUE TIPI DI POTATURA

## PRE POTATURA - *halting*

La costruzione dell'albero viene **interrotta in anticipo** → non ci sono ulteriori rami  
I **valori soglia** sono prescritti precedentemente per decidere quali frazionamenti sono utili

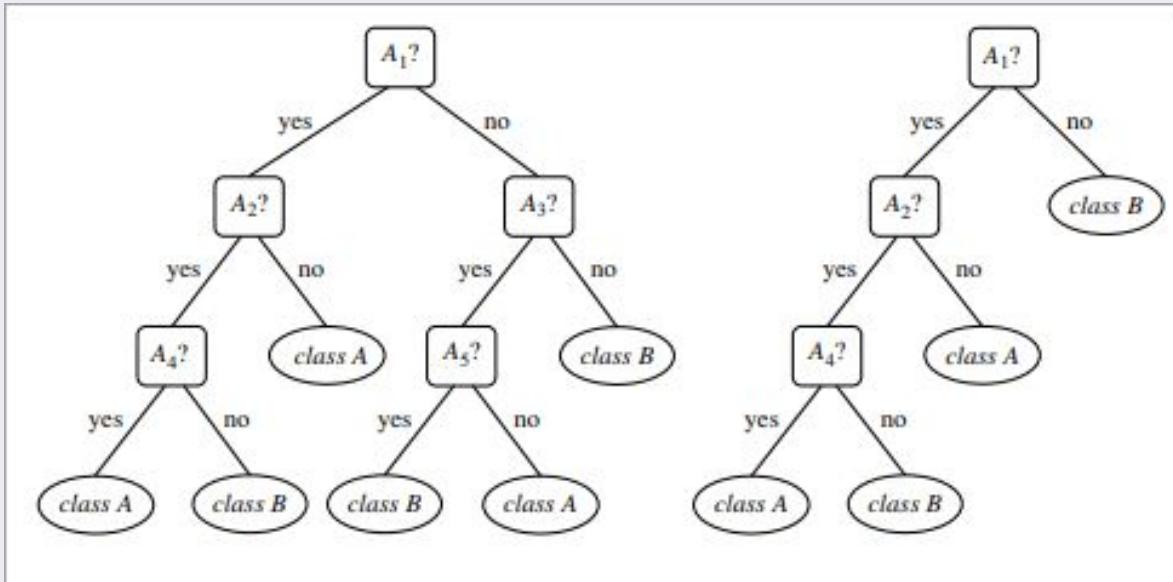
**PRO:** più veloce

## POST POTATURA - *pruning*

Si rimuovono i **rami anomali** da un albero completamente cresciuto  
I rami rimossi sono sostituiti da un **nodo foglia**

**PRO:** più affidabile

Gli alberi potati sono **più precisi e compatti**



A sinistra “albero non potato” a destra “albero potato”

# ALGORITMI PER LA COSTRUZIONE DELL'ALBERO

## ID3

**Iterative dichotomiser tree**

Teorizzato nel  
1986 da  
John Ross Quinlan

## C4.5

È l'evoluzione del  
precursore ID3  
Ideato nel 1993 da  
John Ross Quinlan

## C5/See5

Sostituisce la  
versione  
commerciale di  
C4.5

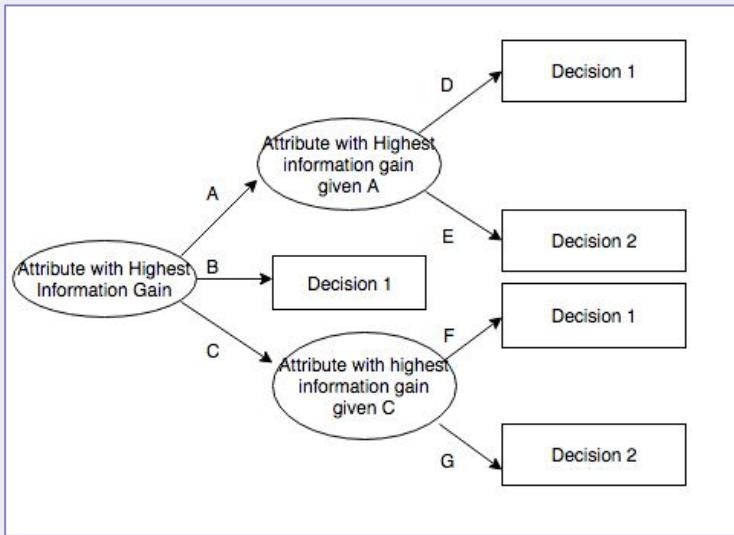
## CART

**Classification and regression tree**

Analizza sia attributi  
continui sia discreti.  
Ideato nel 1984

# ALGORITMI PER LA COSTRUZIONE DI UN ALBERO

## ID3 - *Iterative Dichotimiser 3*



È stato inventato da **Ross Quinlan** ed è utilizzato per generare un albero decisionale da un set di dati.

È spesso utilizzato nei domini dell'**apprendimento automatico** e dell'**elaborazione del linguaggio naturale**

# FUNZIONAMENTO ID3

1. Ad ogni interazione si scorrono gli attributi inutilizzati del set di dati e si calcola **l'Information Gain** (o l'*entropia* a seconda del metodo scelto)
2. Si seleziona l'attributo con Information Gain maggiore (o con *entropia minore*)
3. Il set di dati viene partizionato dall'attributo selezionato per produrre un sottoinsieme di dati



Dati di addestramento

Attributo di output

Lista di attributi

```
ID3 [Examples, Target Attribute, Attributes]
Create a root node for the tree
If all examples are positive, Return the single-node tree Root, with label = +.
If all examples are negative, Return the single-node tree Root, with label = -.
If attribute list is empty, then Return the single node tree Root, with label = most common value of the target attribute
in the examples.

Otherwise Begin
    A ← The Attribute that best classifies examples.
    Decision Tree attribute for Root = A.

    For each possible value,  $v_i$ , of A,
        Add a new tree branch below Root, corresponding to the test  $A = v_i$ .
        Let Examples( $v_i$ ) be the subset of examples that have the value  $v_i$  for A
        If Examples( $v_i$ ) is empty
            Then below this new branch add a leaf node with label = most common target value in the examples
        Else below this new branch add the subtree ID3 Examples( $v_i$ ), Target_Attribute, Attributes - {A})
    End
    Return Root
```

# PROPRIETÀ ID3

- Non garantisce soluzione ottimale, ma produce soluzioni ottime *localmente*
- Una **strategia greedy** seleziona l'attributo migliore a livello locale
- Per evitare l'overfitting si dovrebbero preferire alberi piccoli



**Complessità dell'algoritmo:**

$$n * |D| * \log |D|$$

Con:

- $n$ , numero di attributi nel set di dati di addestramento
- $D$  e  $|D|$ , numero di tuple

# AMBITI DI UTILIZZO



## MEDICINA

Un albero decisionale nella medicina è utilizzato per fornire le diagnosi



## DATA MINING

Per l'estrazione di informazioni utili da grandi quantità di dati



## MARKETING

Per la costruzione di script utilizzati dai venditori/operatori del telemarketing



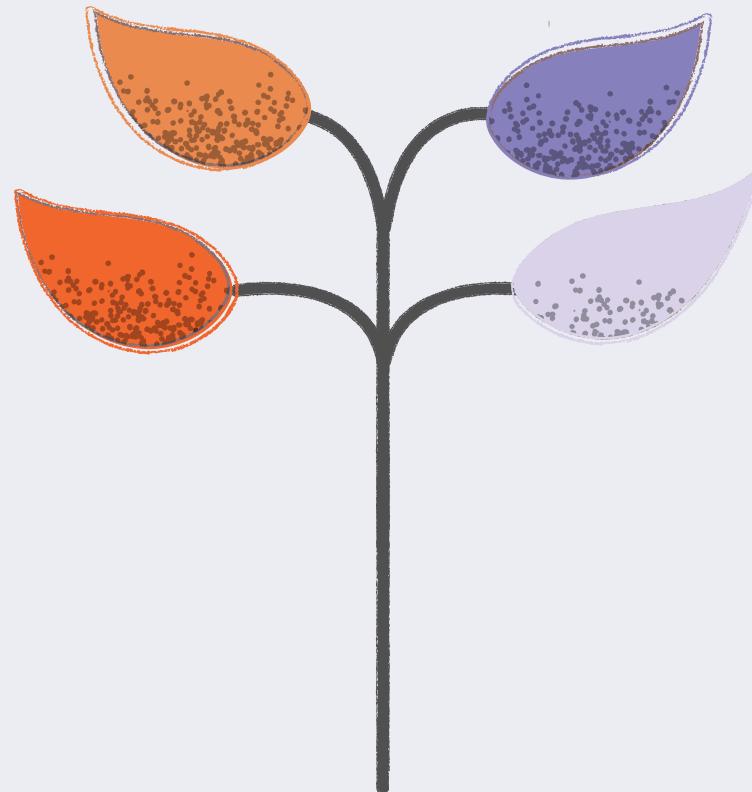
# VANTAGGI E SVANTAGGI DEGLI ALBERI DI DECISIONE

## VANTAGGIO 1

Semplicità  
Più comprensibili delle  
reti neurali

## VANTAGGIO 2

Espressività degli alberi  
decisionali



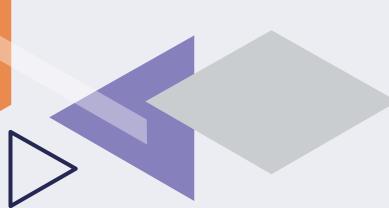
## SVANTAGGIO 1

Poco adatti a problemi  
complessi: lo spazio delle  
ipotesi diventa troppo grande

## SVANTAGGIO 2

Non si possono rappresentare più  
funzioni in uno stesso albero.  
Alcuni tipi di funzioni non sono  
rappresentabili





# 05

# RANDOM

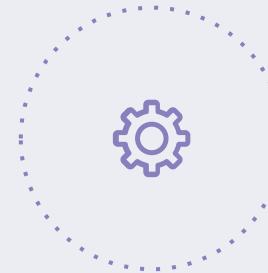
# FOREST

# METODI ENSEMBLE



## Definizione

L'ensemble learning è una tecnica di apprendimento automatico (machine learning) basata sulla costruzione di più ipotesi



## Funzionamento

Non estrapola una sola ipotesi ma molte.



## Importanza della quantità delle ipotesi

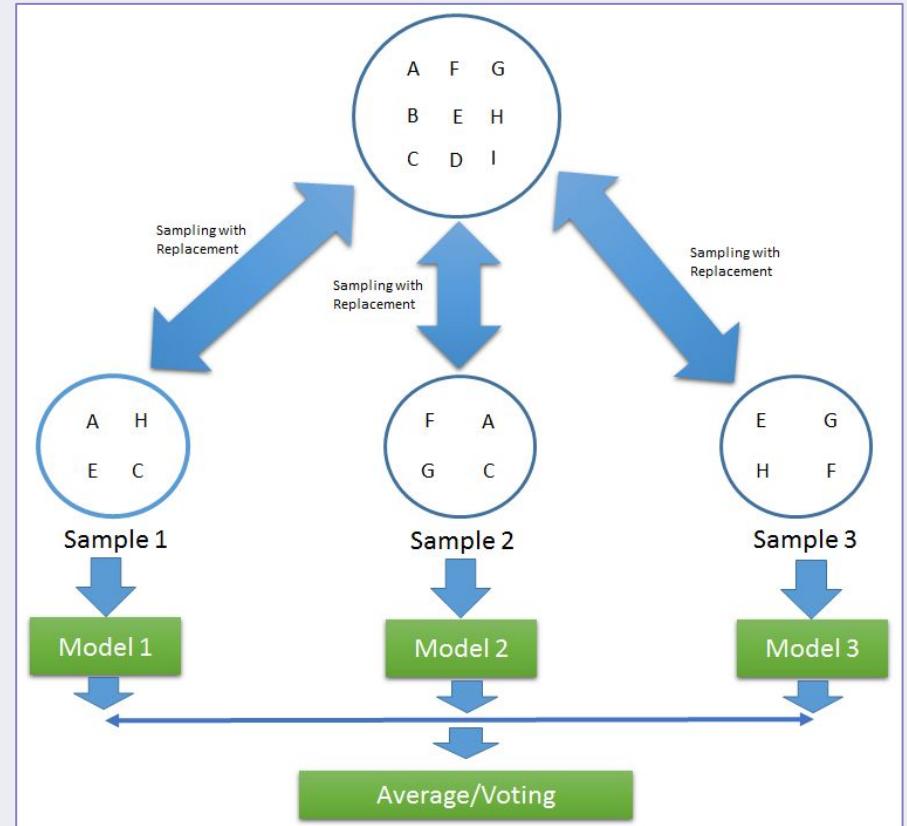
Riduce la probabilità di errore.

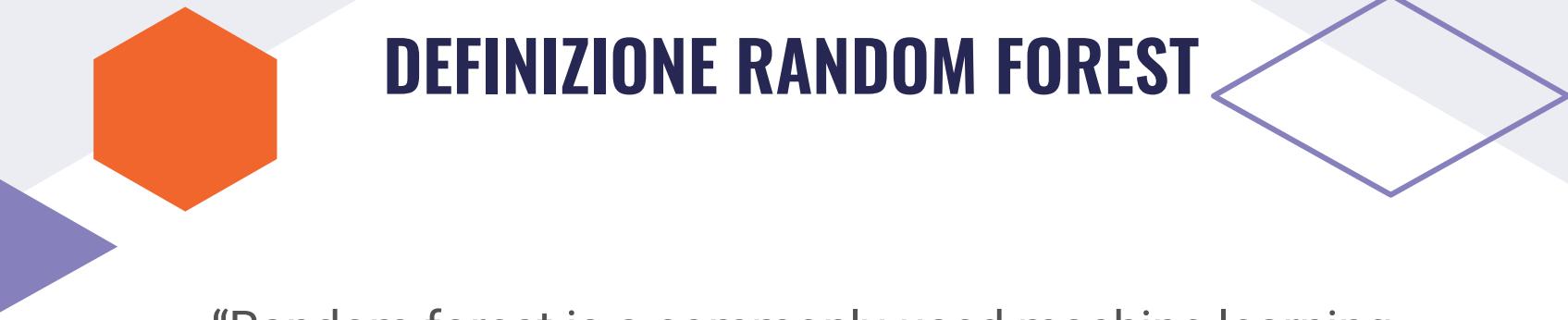
# BAGGING

Il **Bootstrap Aggregation** (o **Bagging**), è un metodo ensemble semplice e potente.

Composto da due parti:

- *Bootstrap*
- *Aggregazione*





# DEFINIZIONE RANDOM FOREST

“Random forest is a commonly-used machine learning algorithm trademarked by Leo Breiman and Adele Cutler, which combines the output of multiple decision trees to reach a single result. Its ease of use and flexibility have fueled its adoption, as it handles both classification and regression problems.”

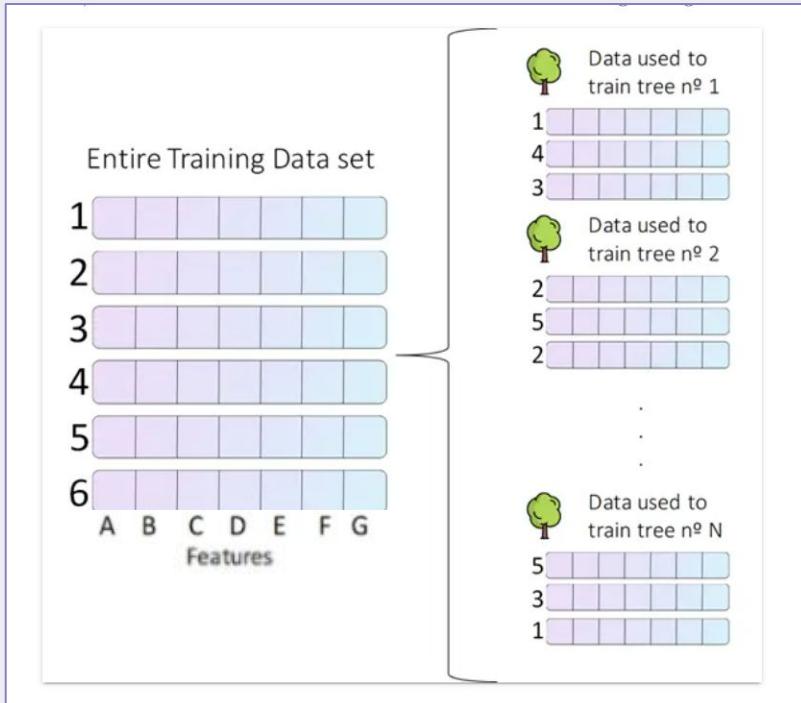
—IBM

# COME SI CALCOLA UNA RANDOM FOREST

1. Trarre un campione casuale iniziale (bootstrap) di dimensioni  $k$ :
2. Far crescere un albero decisionale dal campione di bootstrap. Per ogni nodo:
  - a. Selezionare casualmente  $d$  caratteristiche senza reinserimento;
  - b. Suddividere il nodo utilizzando la caratteristica che fornisce la migliore suddivisione sulla base della funzione obiettivo;
3. Ripetere per  $n$  volte i passi "a" e "b" (passo 2)
4. Aggregare le previsioni di ciascun albero per assegnare l'etichetta della classe sulla base di un voto di maggioranza

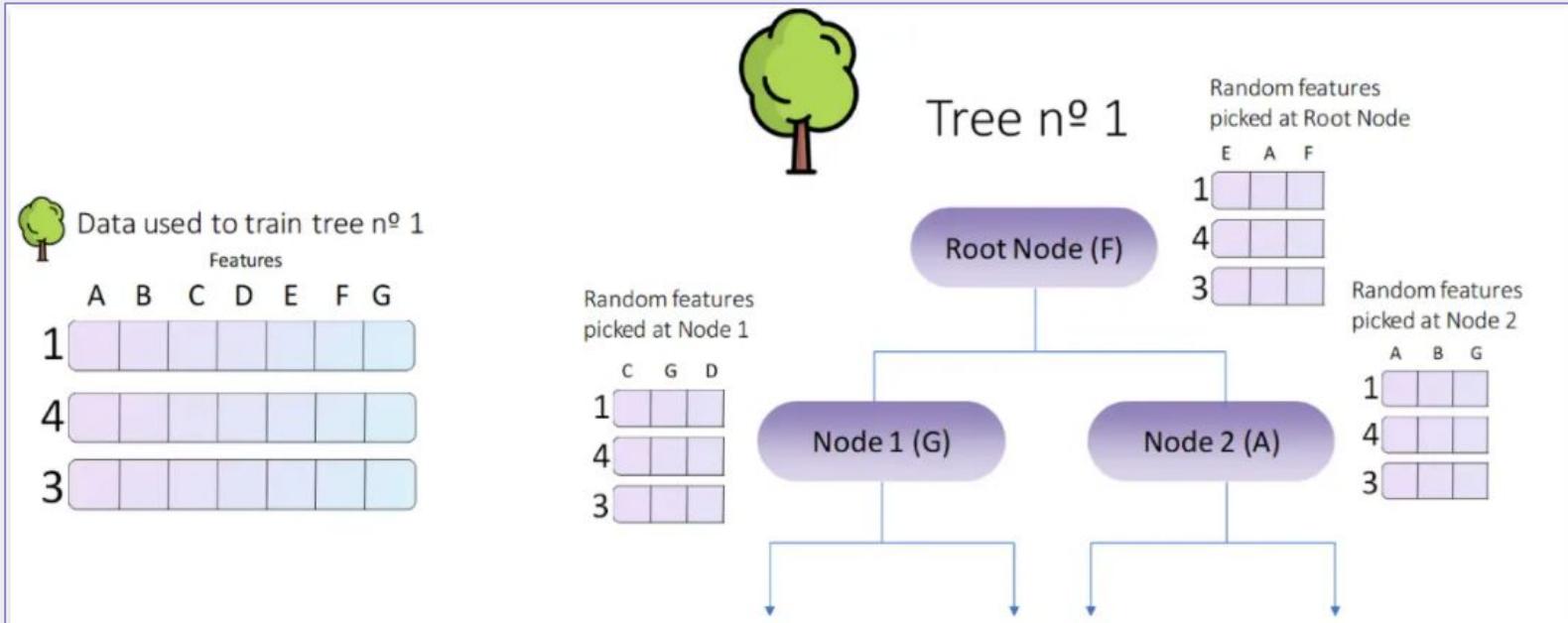


# ESEMPIO



1. Trarre un campione casuale iniziale (bootstrap) di dimensioni  $k$
2. Far crescere un albero decisionale dal campione di bootstrap. Per ogni nodo:
  - a. Selezionare casualmente  $d$  caratteristiche senza reinserimento;

# ESEMPIO



b. Suddividere il nodo utilizzando la caratteristica che fornisce la migliore suddivisione sulla base della funzione obiettivo;

# ESEMPIO

Classification problem: Medical Diagnosis



Tree nº 1 → Healthy

Tree nº 2 → Sick

.

.

.

Tree nº N - 1 → Healthy

Tree nº N → Healthy

Healthy: 355  
 Sick: 45

Prediction: Healthy  
(Most frequent value)

Regression problem: House price estimation



Tree nº 1 → 350.000\$

Tree nº 2 → 275.550\$

.

.

Tree nº N - 1 → 392.210\$

Tree nº N → 312.300\$

Prediction: 322.750\$



(Average numerical prediction)

4. Aggregare le previsioni di ciascun albero per assegnare l'etichetta della classe sulla base di un voto di maggioranza



# SCELTA DELLE DIMENSIONI

Con l'algoritmo **Random Forest**, a differenza degli alberi decisionali, non dobbiamo preoccuparci della scelta di un buon valore per gli iperparametri, tranne tre:

- Il **numero di alberi**  $n$  che abbiamo scelto per la foresta casuale
  - Maggiore è il numero di alberi, migliori saranno le prestazioni, con lo svantaggio di un incremento del costo computazionale
- Dimensione  $k$  del **campione di bootstrap**
  - Valore grande di  $k \rightarrow$  riduciamo la casualità
  - Valore piccolo di  $k \rightarrow$  riduciamo le prestazioni del modello
- Il **numero delle caratteristiche** (features)  $d$ 
  - Valore ragionevole è  $d=\sqrt{m}$  dove  $m$  è il numero di caratteristiche presenti nel set di addestramento



# MISSFOREST

MissForest è una tecnica di imputazione basata sull'**apprendimento automatico**. Utilizza un algoritmo Random Forest per eseguire l'operazione. Si basa su un approccio iterativo e ad ogni iterazione le previsioni generate sono migliori.



# ESEMPIO

Ad ogni iterazione, l'algoritmo migliora.

Score	Age
98	10
94	?
57	6
78	?
74	8

$\frac{10 + 6 + 8}{3}$

Impute missing values using mean

Score	Age
98	10
94	8
57	6
78	8
64	7

Mark missing values as Predict,  
mark others as Training

Train the Random Forest  
model on the data

Random  
Forest  
Model

Score	Age
98	10
94	10
57	6
78	7
64	7

Use model to generate  
prediction for missing value

# APPLICAZIONE ALGORITMO RANDOM FOREST



Bancario



Sanitario

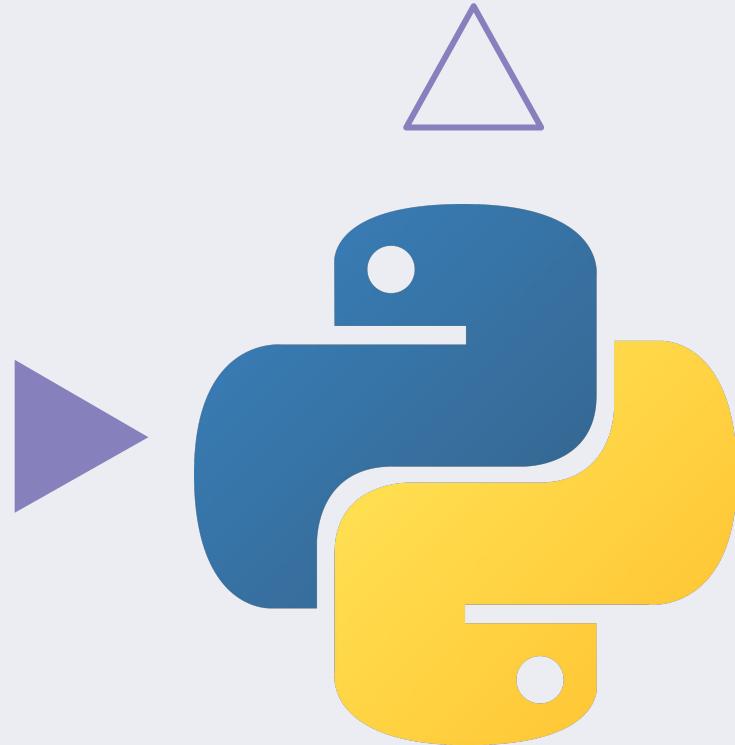


Finanziario



E-commerce

python™



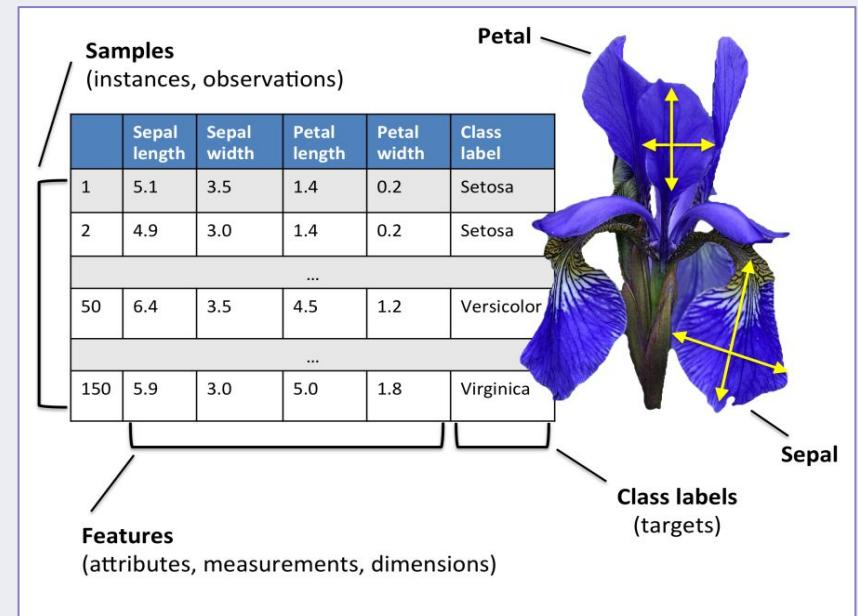
# 06

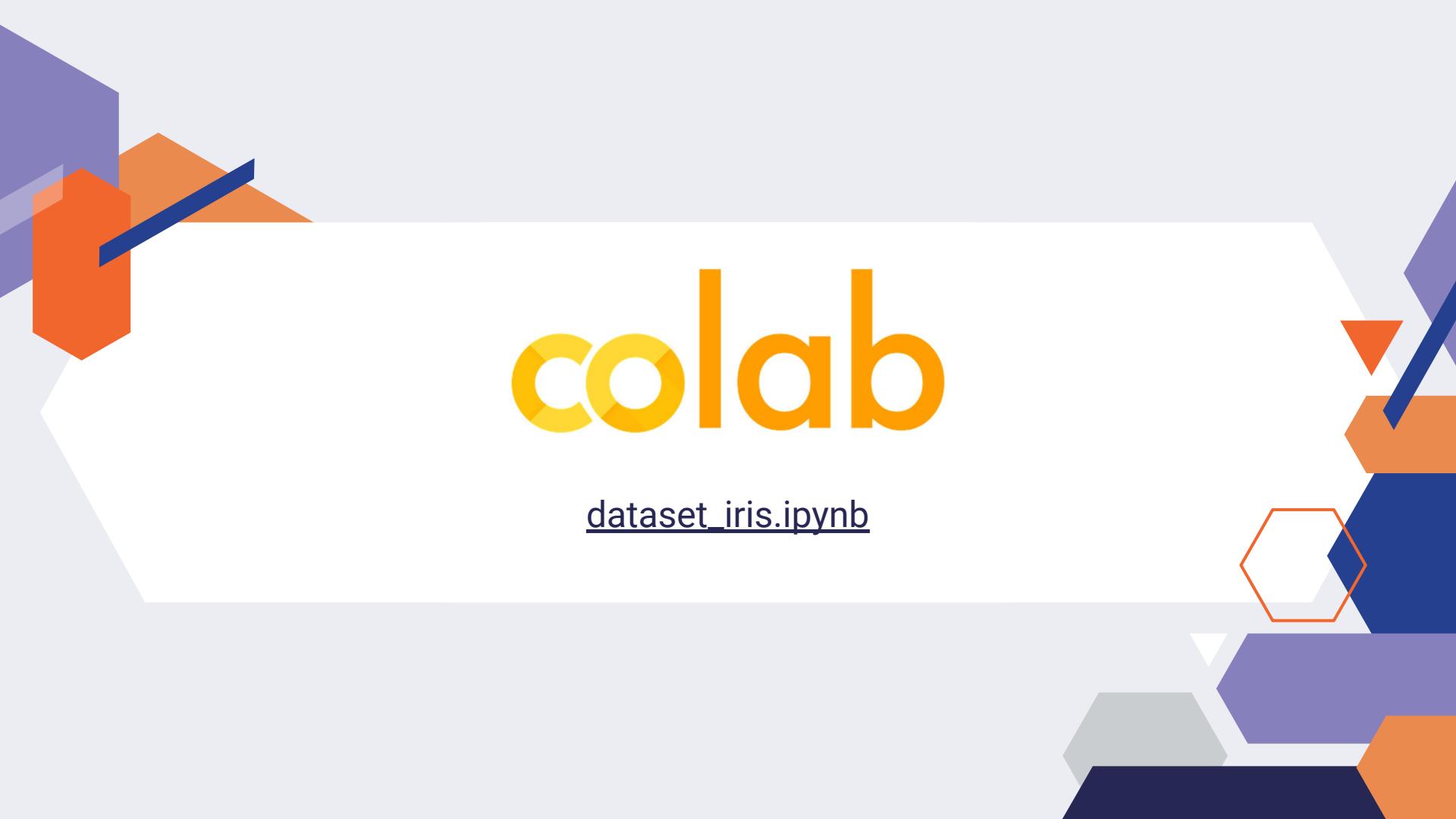
# PYTHON

# DATASET IRIS

Il dataset è formato da:

- **150 campioni**
- **Features** (caratteristiche):
  - *Sepal length*
  - *Sepal width*
  - *Petal length*
  - *Petal width*
- **Target** (classi)
  - *Setosa* == 0
  - *Versicolor* == 1
  - *Virginica* == 2





# colab

[dataset\\_iris.ipynb](#)

# FONTI

## INTRODUZIONE

### **Tipologie di apprendimento**

*Machine learning con python - Sebastian Raschka*

*Data mining - Professoressa Gaito*

*Corso di bioinformatica unimi - Giorgio Valentini*

### **Dataset Iris**

*Machine learning con python - Sebastian Raschka*

### **Bias, varianza, overfitting, underfitting**

*Data mining - Professoressa Gaito*

[AndreaMinini.com](http://AndreaMinini.com)



# FONTI

## ALBERI DI DECISIONE

*Introduzione, funzionamento, variabili utilizzate*

[Wikipedia](#),

[AndreaMinini.com](#),

[GianlucaTramontana.it](#)

**Costruzione**

[AndreaMinini.com](#),

[AnalyticsSteps.com](#)

**Algoritmi costruzione albero di decisione**

[Wikipedia](#)



# FONTI

## RANDOM FOREST

**Metodi ensemble**  
Andrea Minini

**Bagging**  
Livingeconomyadvisors.com

**Introduzione, funzionamento**  
IBM,  
Lorenzo Govoni,  
Netai.it,

*Libro "Machine Learning con Python" di  
Sebastian Raschka*

**MissForest**  
Towardsdatascience.com



# FONTI

## PRATICA

### **Alberi di decisione**

*Libro "Hands-On Machine Learning with Scikit-Learn, Keras, and Tensorflow: Concepts, Tools, and Techniques to Build Intelligent" di Aurélien Géron,*

### **Random forest**

[Datacamp.com](https://www.datacamp.com)

### **MissForest**

[Towardsdatascience.com](https://towardsdatascience.com)



# GRAZIE PER LA VOSTRA ATTENZIONE!

*Avete qualche domanda?*

*Valentina Botti*

*Sara Cinquini*

*Giulia Guglielmi*



[giuliaguglielmi/apprendimento-supervisionato \(github.com\)](https://github.com/giuliaguglielmi/apprendimento-supervisionato)

