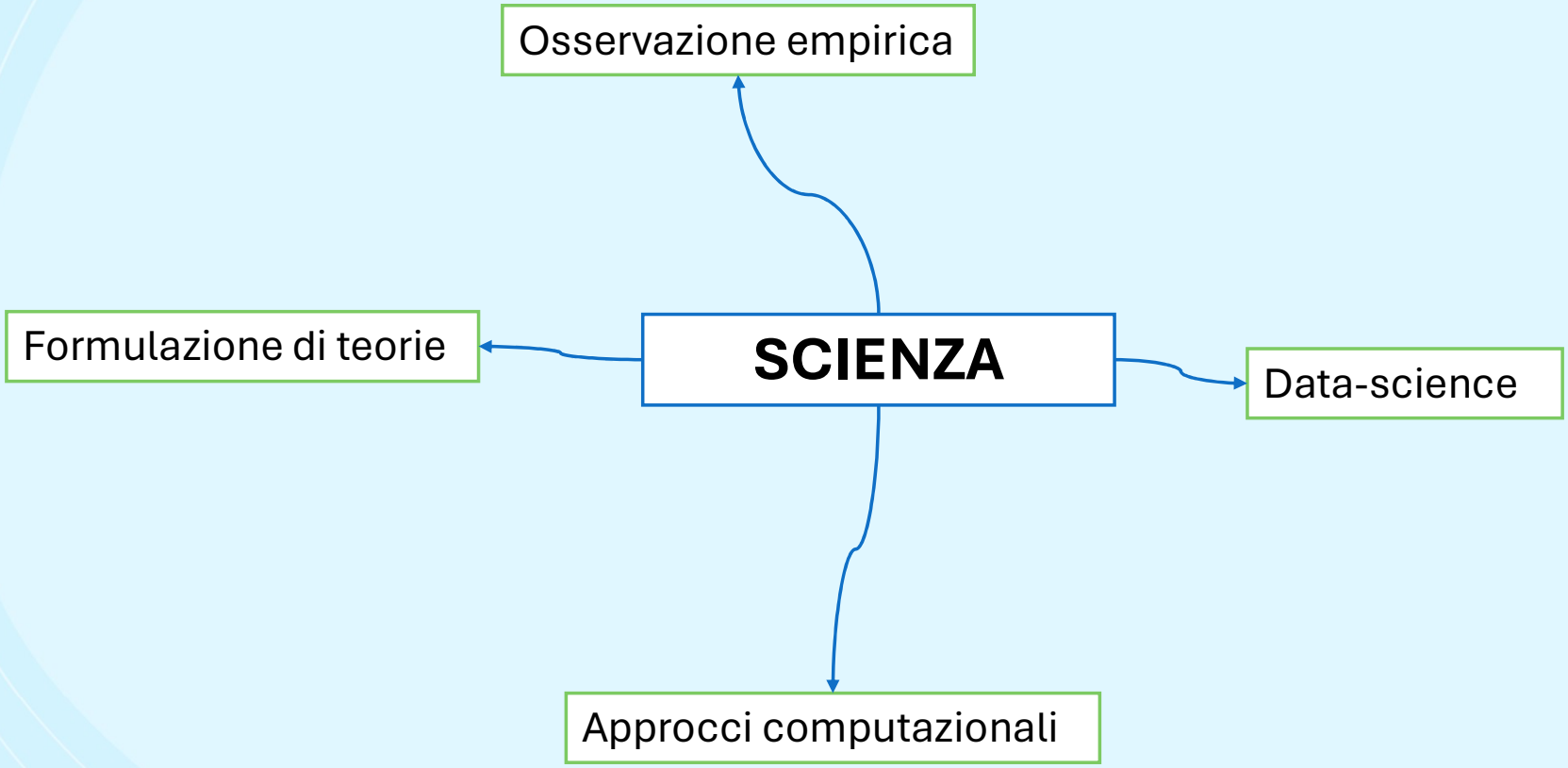
Machine Learning

Obiettivi di apprendimento

- Differenza tra modello fisico e modello statistico.
- Vocabolario di base nel Machine Learning.
- Cosa si intende per rappresentazione di una molecola e quali sono più utilizzate.
- Nozioni sulle proprietà dei basilari algoritmi di ML supervisionato e non-supervisionato.
- Utilizzo delle librerie `rdkit`, `mordred` e `sklearn` per il ML.

La scienza in evoluzione



La Data-Science è una scienza esatta?

Spoiler: NO

Data-science

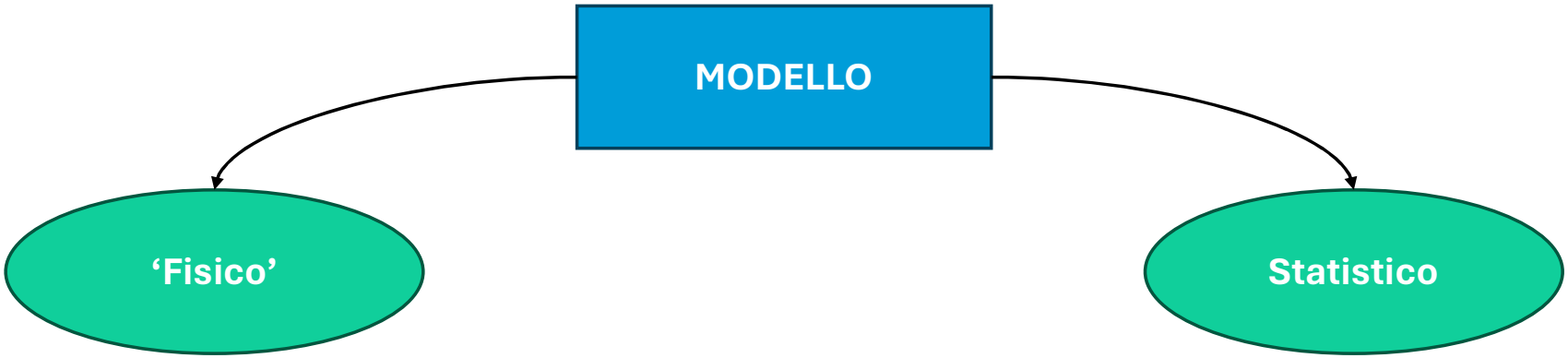
Data-science è un approccio multidisciplinare in cui esperimenti, simulazioni, **statistica** e machine learning vengono impiegate per estrapolare un modello descrittivo del «mondo».

L'apprendimento è, quindi, guidato dai dati (*data driven science*). Il vantaggio è che tale approccio sia in grado di trattare una quantità di dati e informazioni molto grandi e complesse, che con gli approcci 'tradizionali' alla ricerca scientifica non sono (ancora) trattabili.

La disponibilità di un dataset molto esteso è un prerequisito fondamentale della data-science, nonché un primo difetto: non si può sapere a priori se i dati a disposizione sono completi, possono essere di qualità insufficiente a creare un buon modello, possono contenere errori, possono comportare bias.

Il modello viene costruito sui dati a disposizione. *Cosa comporta avere a disposizione un set di dati imperfetto?*

Data-science



In funzione dei dati a disposizione viene costruito un modello descrittivo che si fonda sui **principi primi**.

Il modello predittivo è **deterministico**.

Due set di dati diversi portano allo **stesso** modello fisico o, quantomeno, alla medesima interpretazione del fenomeno.

Scienza esatta

In funzione dei dati a disposizione viene costruito un modello descrittivo che si basa sulle **correlazioni** tra i dati.

Il modello predittivo è **statistico**.

Due set di dati diversi quasi certamente portano a un **diverso** modello statistico.

Scienza non esatta

Machine Learning

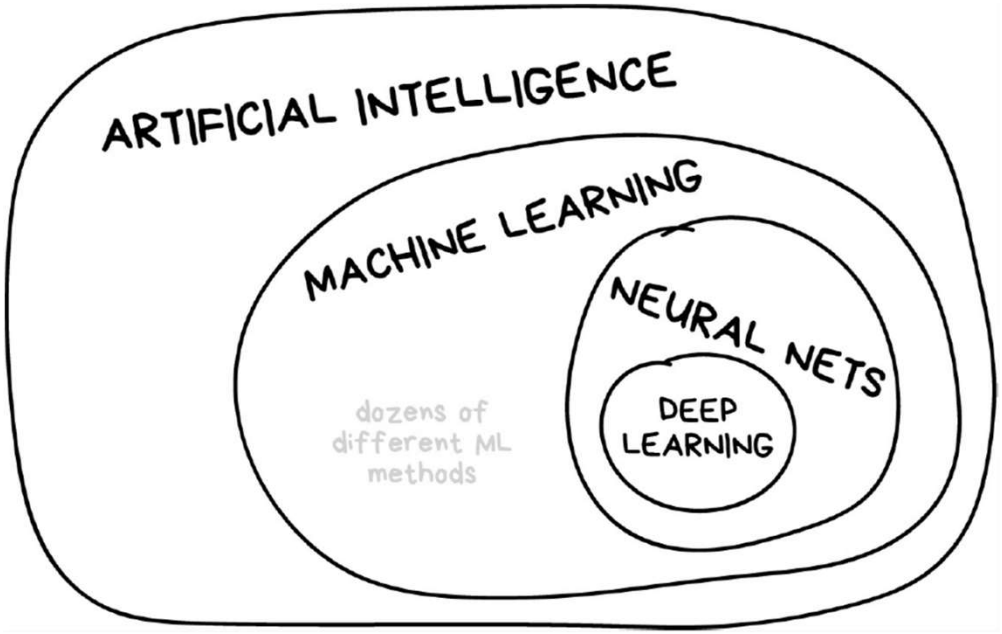
Il concetto di **apprendimento automatico** (*machine learning* – ML) nasce alla fine del 1950, come approccio che si distacca da quella che è definita intelligenza artificiale (IA).

L'IA è costruita su un approccio *knwoledge-based*: il sistema risponde in funzione dei dati a disposizione, ma l'apprendimento è basato su informazioni fornite dall'esterno.

Il ML si distingue proprio per l'approccio di apprendimento dai dati, sfruttando le correlazioni tra esse, analizzate con gli strumenti della statistica.

L'utilizzo del ML diventa crescente attorno al 1990, quando cominciano a esserci:

- computer più performanti
- algoritmi più avanzati
- una incrementata **disponibilità dei dati**, grazie alla nascita del World Wide Web al CERN di Ginevra, nel 1991, che ha portato all'uso di internet in tutto il mondo.



Tipologie di Machine Learning

- ***Supervised learning***: si costruisce un modello matematico sulla base di esempi costituiti da descrittori (*features*) ed etichette (*labels*). Si divide in due categorie principali:
  - classificazione: divisione dei dati in categorie,
  - regressione: assegnazione di un valore a una funzione continua.

- ***Unsupervised learning***: apprendimento basato su esempi senza etichette:
  - *Clustering*: raggruppamento dei dati per similitudine,
  - *riconoscimento di patterns* tra i dati.

- ***Reinforcement learning***: basato sul *trial & error* in cui in funzione di un'azione, l'agente che la compie riceve una ricompensa o una penalità. L'apprendimento avviene nel senso di massimizzazione della ricompensa.

- ***Feature learning***: apprendimento mirato alla migliore rappresentazione dei dati. Impiega tecniche quali PCA e *clustering*.

- ***Robot learning***: campo di ricerca applicato alla robotica in cui si studiano tecniche che permettano a un robot di sviluppare nuove abilità usando algoritmi di apprendimento.

Interpretable ML

Come discusso sopra, la data-science non è una scienza esatta. *Si può fare nulla a riguardo?*

Una possibile soluzione è offerta da quello che viene definito ***interptretable ML*** ossia un approccio di apprendimento automatico che mette un umano 'esperto' nelle condizioni di poter interpretare ciò che l'algoritmo di ML ha imparato. Questa è la condizione per poter sfruttare la data-science per la ricerca scientifica nel senso più stretto che il metodo scientifico impone: stabilire relazioni causa-effetto tra le osservazioni sperimentali.

L'utilizzo del ML per la ricerca scientifica deve rispettare i vincoli stabiliti dal metodo scientifico. Uno di questi è la riproducibilità degli esperimenti. Il ML interpretabile sembra offrire una possibilità in questa direzione.

Ad oggi si sta vivendo una 'crisi di riproducibilità' dovuta al fenomeno chiamato *publish or perish*, scatenato dall'utilizzo di semplici indici bibliometrici per valutare la carriera di ricercatrici e ricercatori.

Una possibile soluzione è quella di obbligare la pubblicazione dei dati 'grezzi', in modo che non nascano bias nell'interpretazione dei risultati. Questo sarà in particolar modo rilevante nella pubblicazione di risultati ottenuti attraverso ML.

Dataset per l'apprendimento

Una semplice definizione di apprendimento è: **generalizzare dagli esempi**.

Il punto critico del ML è la rappresentazione dei dati in maniera tale che l'algoritmo possa effettivamente apprendere. Questo si verifica se i dati possono essere separati.

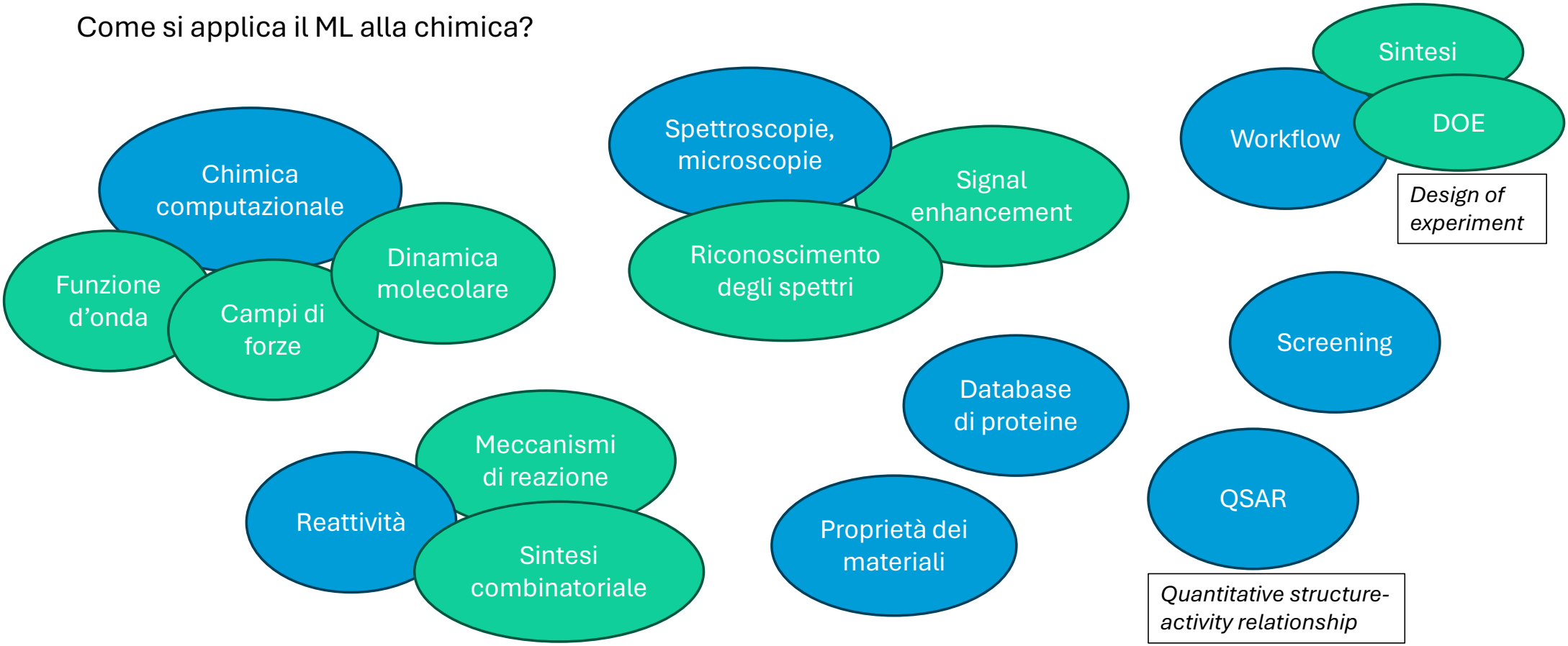
Possibili problemi con i dati.

- **Quantità**: il dataset è sufficientemente esteso da coprire le possibili correlazioni, e la loro entità, tra i dati?
- **Qualità**: i dati contengono dei bias? Ci sono errori? Omissioni? *Outliers*?
- **Rappresentazione**: ci possono essere più modi per rappresentare la medesima informazione; modificando la rappresentazione si può incrementare la separazione tra i dati.
- **Numerosità** dei descrittori: troppi descrittori possono creare problemi di separazione dei dati; infatti, un suggerimento è quello di effettuare una riduzione della dimensionalità (ad es., via PCA) in modo da cercare a priori un minore set di descrittori, ma che siano più separabili.

Applicazione del ML in Chimica

Il potenziamento del ML e l'incrementata facilità di scambiare e reperire dati ha fatto da trampolino di lancio per la data-science.

Come si applica il ML alla chimica?



Applicazione del ML in Chimica

Iniziative Big Data e ML

- Chimica quantistica: io-Chem-BD [Alvarez-Moreno, M. et al. *J. Chem. Inf. Model* **2015**, *55*, 95]
- Farmacologia: BIGCHEM [Tetko, I. V. et al. *Mol. Inform.* **2016**, *35*, 615]
- Scienza dei materiali: US Materials Genome Inititive [www.mgi.gov]
- Sintesi: Reaxys [www.reaxys.com]
- QM + reti neurali per proprietà molecolari: PROphet [Kolb, B. et al. *Sci. Rep.* **2017**, *7*, 1]
- *Computer aided drug design*: pkCSM [structure.bioc.cam.ac.uk]
- Sequenze di proteine: ASTRAL [Fox N. K., et al. *Nucleic Acids Res.* **2014**, 0304]

Nomenclatura – CCS

***Chemical Compound Space* (CCS – spazio dei composti chimici)**
Racchiude tutte le possibili molecole che potrebbero esistere considerando tutte le possibili configurazioni metastabili (minimi locali nelle superfici di energia potenziale elettronica) che si ottengono dalla risoluzione dell'equazione di Schrödinger 'indipendente dal tempo'.

Quanto è esteso il CCS?

- Limitandosi all'insieme di molecole entro i 500 Dalton,
- usando solo C, H, N, O e S,
- e fino ad anelli di 4 atomi,

si arriva a stimare l'inimmaginabile numero di $\mathbf{10^{63}}$ molecole!

Ad oggi il CAS (Chemical Abstract Service) registra 206 milioni ($\mathbf{O(10^8)}$) molecole organiche e inorganiche.

Questo pone un limite nell'applicazione del ML alla predizione delle proprietà di nuove molecole dato che lo spazio noto è ad oggi troppo limitato al CCS. Ci sono, però, delle considerazioni che si possono fare:

1. l'utilizzo della chimica computazionale può aiutare ad aumentare lo spazio noto per il *training*, pur se le molecole non sono state sintetizzate e caratterizzate
2. il CCS può essere suddiviso in sottospazi che racchiudono proprietà selezionate (intorno chimico, gruppi funzionali, ecc.) in modo che siano più ristretti e la predizione di modelli ML sia più probabilmente attendibile.

Nomenclatura – Rappresentazione

Con **rappresentazione** si intende una codifica della molecola che porti informazione sulla struttura e sulle relazioni tra gli atomi. Una rappresentazione deve essere unica e invariante all'indicizzazione degli atomi, così come alle operazioni di simmetria.

La nomenclatura ***IUPAC*** è una classificazione sistematica e standardizzata, ma nomi lunghi possono far nascere ambiguità.

SMILES (*simplified molecular-input line-entry system*); descrive la struttura di specie chimiche come stringhe ASCII basandosi su semplici regole.

InChI notation (*international chemical identifier*) è l'equivalente digitale per il nome IUPAC di un composto.

Fingerprints codificano la presenza, o meno, di certe caratteristiche in un composto (ad es., la presenta di frammenti o meno).

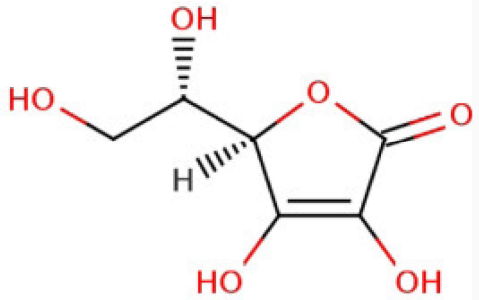
Chemical graph è la rappresentazione della formula di struttura di una molecola in termini di teoria dei grafi.

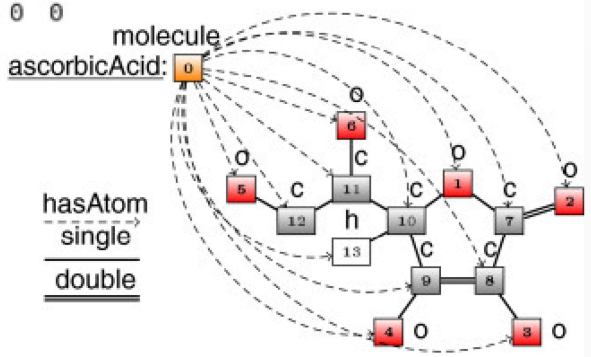
Esempio. Rappresentazioni dell'acido L-ascorbico

IUPAC: (5R)-[(1 S)-1,2-Dihydroxyethyl]-3,4-dihydroxyfuran-2(5H)-one
SMILES: OC=1C(OC(=O)C=1O)[C@@H](O)CO
InChI: 1S/C6H8O6/c7-1-2(8)5-3(9)4(10)6(11)12-5/h2,5,7-10H,1H2/t2-,5+/m0/s1/i3+1

```
  Marvin  03310812332D

 13 13  0  0  1  0            999 V2000
    0.0000    0.0000    0.0000 O   0  0  0
   -0.8974   -1.9370    0.0000 O   0  0  0
    0.8974   -1.9370    0.0000 O   0  0  0
    0.6674   -0.4849    0.0000 C   0  0  0
    1.4521   -0.2300    0.0000 O   0  0  0
    0.4125   -1.2695    0.0000 C   0  0  0
   -0.4125   -1.2695    0.0000 C   0  0  0
   -2.8109   -0.0724    0.0000 O   0  0  0
   -2.0964   -0.4849    0.0000 C   0  0  0
   -1.3819   -0.0724    0.0000 C   0  0  2
   -1.3819    0.7526    0.0000 O   0  0  0
   -0.6674   -0.4849    0.0000 C   0  0  1
   -1.3819   -0.8974    0.0000 H   0  0  0
  1  4  1  0  0  0  0
 12  1  1  0  0  0  0
  7  2  1  0  0  0  0
  6  3  1  0  0  0  0
  4  5  2  0  0  0  0
  4  6  1  0  0  0  0
  6  7  2  0  0  0  0
 12  7  1  0  0  0  0
  8  9  1  0  0  0  0
  9 10  1  0  0  0  0
 10 12  1  0  0  0  0
 10 11  1  6  0  0  0
 12 13  1  6  0  0  0
M  END
```





Descrittori molecolari

SMILES (*Simplified Molecular Input Line Entry System,* https://en.wikipedia.org/wiki/Simplified_molecular-input_line-entry_system) è un metodo per rappresentare una molecola come una stringa di catatteri alfa-numerici e caratteri speciali, in modo tale che si possano dedurre informazioni sulla sua struttura e, quindi, poter calcolare anche descrittori basati su distanze e distribuzioni degli atomi.

La libreria **RDKIT** sviluppata in ambito chemioinformatico permette di interpretare le stringhe SMILES, disegnare le molecole e calcolare anche alcune decine di descrittori molecolari.

Una molecola è descritta come un **grafo** in cui i *nodi* sono i nuclei e gli *archi* sono i legami. Viene definito tramite la **tabella di connessione** e **la tabella dei legami**.

Sia ai nodi che agli archi possono essere assegnate proprietà:
- nodo (atomo): simbolo chimico, coordinate Cartesiane, massa molare, carica, ecc.
- arco (legame): ordine, polarizzazione, lunghezza, ecc.

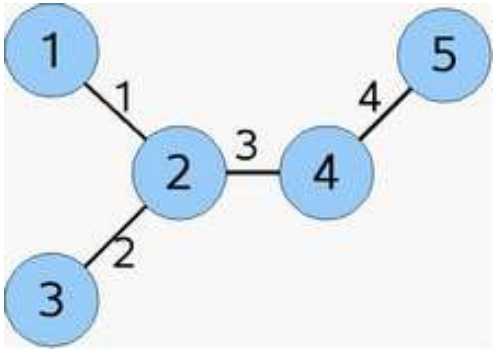


Tabella di connessione

Atomo	Atomi legati
1	**2**
2	**1, 3, 4**
3	**2**
4	**2, 5**
5	**4**

Tabella dei legami:

	1	2	3	4	5
1	**0**	**1**	**0**	**0**	**0**
2	**1**	**0**	**2**	**3**	**0**
3	**0**	**2**	**0**	**0**	**0**
4	**0**	**3**	**0**	**0**	**4**
5	**0**	**0**	**0**	**4**	**0**

Descrittori molecolari

Una molecola può essere descritta in una infinità di modi, dalla semplice formula bruta, alla sua connettività, polarità, distanze interatomiche, ecc. Come specificato prima, non è solamente il numero di descrittori a permettere all'algoritmo di apprendere, ma (e soprattutto), quanto questi siano separabili.

In ambito chemioinformatico sono stati sviluppati strumenti per il calcolo di descrittori molecolari che abbiano lo scopo proprio di poter separare le molecole in classi, in funzione del tipo di correlazione tra molecola e proprietà che si sta cercando di far emergere.

MORDERED è un pacchetto software sviluppato per Python che è in grado di calcolare più di 1800 descrittori molecolari.

Table 3 List of Mordred descriptors

Descriptor name	Number of descriptors (preset)
2D	
ABCIndex	2
AcidBase	2
AdjacencyMatrix	13
Aromatic	2
AtomCount	16
Autocorrelation	606
BCUT[a]	24

Table 3 continued

Descriptor name	Number of descriptors (preset)
3D	
CPSA	43
GeometricalIndex	4
GravitationalIndex	4
MoRSE	160
MomentOfInertia	3

[a] RDKit wrapper

Descrittori molecolari

La capacità dell'algoritmo di apprendere è molto sensibile alla scelta dei descrittori. Una bassa abilità di apprendimento può essere causata dalle seguenti problematiche sui descrittori.

- **Irrilevanti:** descrittori che siano completamente irrilevanti con il tipo di output richiesto. In problemi complessi, non è sempre possibile stabilire a priori se un descrittore sia rilevante o meno (ad es. l'odore di una sostanza è correlato al suo stato fisico a temperatura ambiente?

- **Ridondanti**: descrittori che possono essere ottenuti da altri; sono meno impattanti sulla capacità di apprendimento, ma aggiungono una inutile complessità al processo.

- **Mancanti**: è il caso in cui mancano dati critici alla catalogazione. Ad es., per apprendere lo stato fisico di una molecola tra solido, liquido e gassoso le temperature di fusione ed ebollizione sono parametri importanti che regolano l'efficacia dell'apprendimento in base alla loro presenta o meno.

- **Rumorosi**: dati che derivano da misure sperimentali sono affette da errore di misura, così come dati derivanti da calcoli sono affetti da un'incertezza dovuta all'accuratezza del modello.

- **Biased**: sono i dati più deleteri per il processo di apprendimento in quanto sono poco rappresentativi dello spazio dei descrittori (ad es., troppo pochi o troppo simili tra loro).

Python Time

ML_example_0-ipynb: esempio di utilizzo di RDKIT e MORDRED.

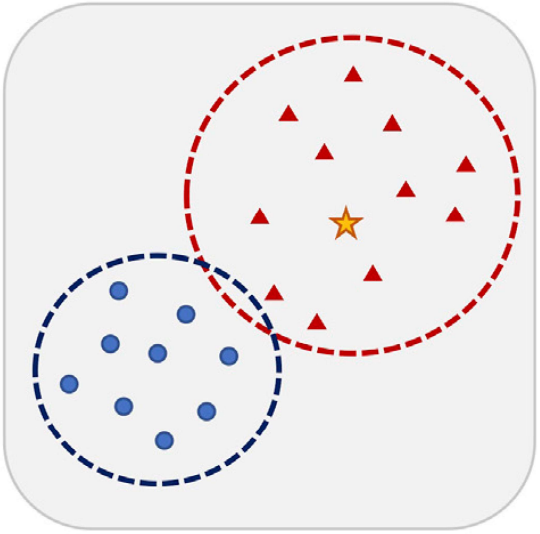
Lo scopo di questo esempio è quello di dimostrare come si utilizzano le librerie:

- `rdkit`: per interpretare la stringa SMILES delle molecole, ricavarne le coordinate Cartesiane ed effettuare una semplice ottimizzazione di geometria

- `mordred`: per calcolare più di 1800 descrittori molecolari

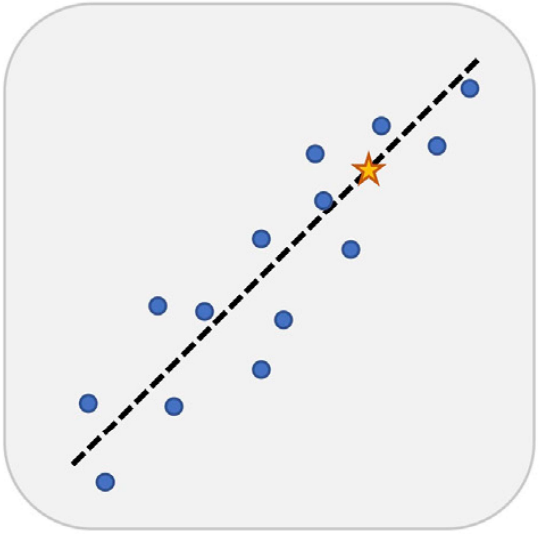
Supervised Learning

Classificazione



Esempio: dato uno spettro IR, un picco a quale gruppo funzionale appartiene?

Regressione



Esempio: qual è il $\Delta_f G^{\ominus}$ di un derivato del metano?

Algoritmi di apprendimento – Alberi Decisionali

Alberi decisionali: si tratta di algoritmi basati sull'approccio *divide & conquer*.

Divide: si analizzano i descrittori in modo tale da trovare quello che permette una prima migliore distinzione tra i dati.

&

Conquer: eseguita tale distinzione si continua a cercare quale descrittore permette un secondo livello di separazione, e così via.

Gli alberi decisionali sono soggetti al problema dell'***overfitting***: si verifica quando l'algoritmo pone così tante domande da andare ad analizzare in dettaglio le differenze dato per dato. In questo caso si perde di vista il concetto di generalizzazione. Ciò che l'algoritmo impara diventa così strettamente collegato agli esempi usati per l'apprendimento, che si può applicare solo ad essi. Un nuovo esempio che mostri anche una piccola differenza da quelli precedenti, verrà catalogato inaccuratamente.

La **profondità dell'albero**, quindi, determina la capacità complessiva dell'algortimo di apprendere. Parametri che regolano la funzionalità complessiva dell'algoritmo si chiamano iperparametri. In genere tali parametri vanno ottimizzati in modo da massimizzare l'accuratezza dell'algoritmo.

Python Time

ML_example_1-ipynb: utilizzo di un albero decisionale per determinare la forza di un acido:

1. catalogazione: riconoscere se è forte o debole,
2. regressione: prevedere il valore della pk_a

Dataset

Rappresentazione SMILES degli acidi, valore della pk_a e indicazione della forza: 1 se forte, -1 se debole.

Librerie

`pandas` gestione del database
`rdkit` interpreatazione SMILES
`mordred` calcolo dei descrittori
`sklearn` alberi decisionali

specie	pka	forza
OCl(=O)(=O)=O	-10.000	1
[IH]	-9.505	1
[BrH]	-9.000	1
[ClH]	-6.114	1
OS(=O)(=O)O	-3.000	1
O[N+]([O-])=O	-1.380	1
OC(=O)C(=O)O	1.268	-1
OS(O)=O	1.886	-1
OS([O-])(=O)=O	2.000	-1
OP(=O)(O)O	2.149	-1
ON=O	3.143	-1
[FH]	3.180	-1
C(=O)O	3.745	-1
OC(=O)c1ccccc1	4.201	-1
C(=O)(C(=O)[O-])O	4.268	-1
CC(=O)O	4.745	-1
[O-]C([O-])=O	6.357	-1
[SH2]	6.959	-1
OP(=O)(O)[O-]	7.201	-1
[SH-]	2.208	-1
OCl	7.538	-1
C#N	9.208	-1
[NH4+]	9.237	-1
OB(O)O	10.000	-1
OC([O-])=O	10.328	-1
[O-]P(=O)(O)[O-]	12.377	-1
B(O)(O)[O-]	12.745	-1
B(O)([O-])[O-]	13.796	-1

Algoritmi di apprendimento – *Clustering*

Clustering: si tratta di algoritmi che, sulla base di una metrica che misuri la 'distanza' tra i dati, colloca quest'ultimi in cluster in modo tale che tutti i punti di un cluster sono più vicini al centro di quel cluster rispetto ai centri di tutti gli altri.

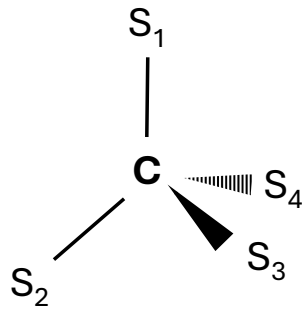
L'idea di base è quella di vedere i descrittori come dei vettori in uno spazio D- dimensionale, con D il numero di descrittori che costituiscono un esempio. In tale spazio si può usar la distanza Euclidea come metrica.

Una volta creati i cluster a partire dagli esempi espressi come coppie (descrittori, etichetta), dato un set di descrittori di test, il valore dell'etichetta stimato sarà quello del centro del cluster in cui quella serie di descrittori cade.

Il **numero di cluster** è l'iperparametro di questo algoritmo. Il *clustering* è soggetto all'*overfitting* del rumore dei dati, in caso i dati siano soggetti a fonti di errore. Inoltre, la dimensionalità dello spazio dei descrittori causa un problema: più dimensioni ci sono e più le distanze tra i punti tendono a essere meno disperse. Quindi, tale algoritmo funziona meglio quando i descrittori sono pochi.

Python Time

ML_example_2.ipynb: utilizzo del clustering per determinare lo stato fisico di derivati del metano a 298.15 K.



Dati N differenti tipi di sostituenti, il numero di possibili molecole differenti è

$$M = \binom{N}{1} + 3\binom{N}{2} + 3\binom{N}{3} + \binom{N}{4}$$

N	7	10	20	100	200
M	210	715	8855	44421275	68685050

Descrittori: $\Delta_f H^{\ominus}$, $\Delta_f G^{\ominus}$, $c_p^{\ominus}$ dei composti a 298.15 K per il set di sostituenti: H, F, Cl, Br, NH_2, OH, SH.

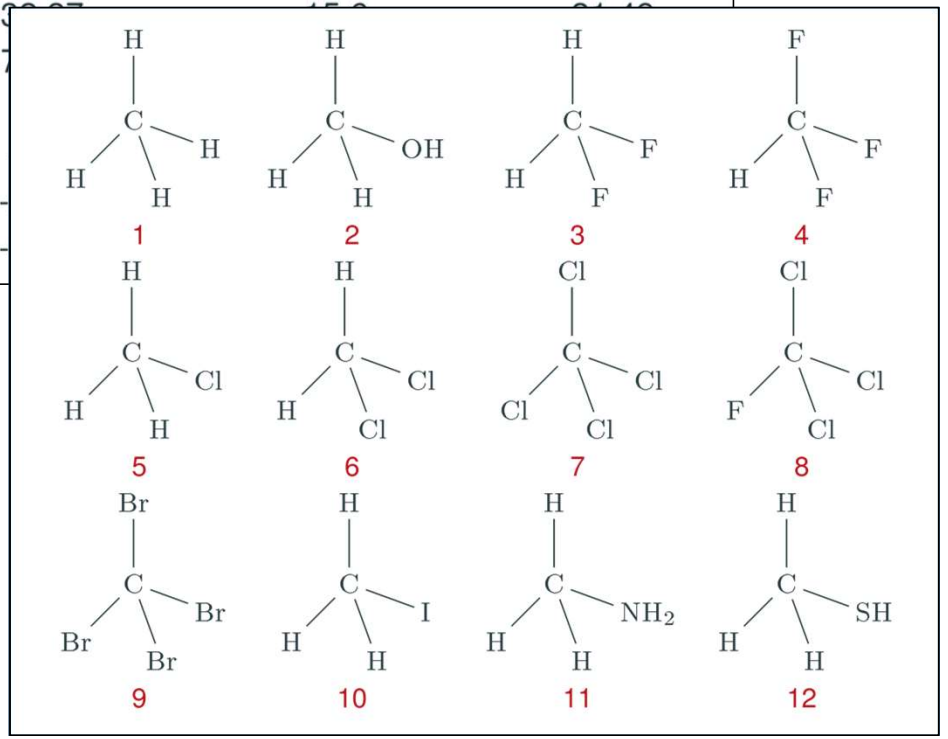
Etichetta: stato fisico: 0 (solido), 1 (liquido), 2 (gassoso)

Algoritmo di classificazione: *clustering* con 3 centri.

Python Time

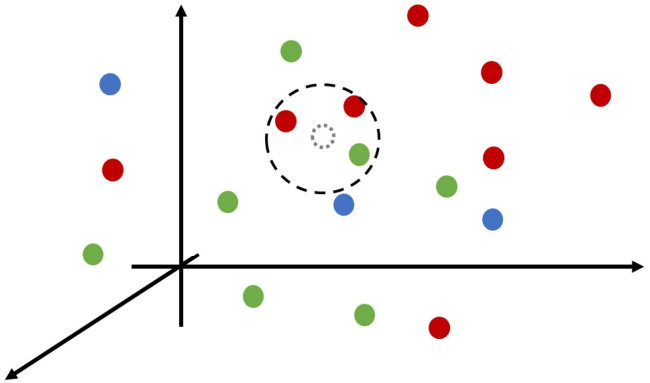
Dataset

Compound	Phase	$\Delta H_f^{\ominus}$ (kcal/mol)	$\Delta G_f^{\ominus}$ (kcal/mol K)	$c_p^{\ominus}$ (cal/mol K)
1	g	-17.88	-12.13	8.439
2	l	-57.04	-39.76	19.5
3	g	-106.8	-100.2	10.25
4	g	-164.5	-156.3	12.20
5	g	-19.32	-13.72	9.74
6	l	-29.03	-16.09	23.9
7	l			
8	l			
9	s			
10	l			
11	g			
12	g			



Python Time

Si utilizzerà l'algoritmo di NN (*nearest neighbor*): un algoritmo di apprendimento statistico che, in base agli esempi, stabilisce qual è la probabilità che un punto di test sia assegnato a una delle categorie.



Come metrica si può usare la distanza Euclidea: $d_{i,j} = \left|\mathbf{X}_i - \mathbf{X}_j\right|$

Un algoritmo migliore è il KNN (*k nearest neighbor*): l'assegnazione dell'etichetta viene basata sulla distanza con i k primi vicini. Ognuno di essi 'vota' l'appartenenza del punto a una delle classi, generando quindi un peso, w_d, di assegnazione. La metrica diventa:

$$d_{i,j} = \sqrt{\sum_{d=1}^{3} w_d \left(X_{i,d} - X_{j,d}\right)^2}$$

Algoritmi di apprendimento – Percettrone

Percettrone (*perceptron*): si tratta di un algoritmo bio-ispirato che, a differenza dei due precedenti algoritmi, è in grado di pesare l'importanza dei descrittori nel processo di apprendimento. Negli alberi decisionali i descrittori vengono usati o scartati; nel *clustering*, i descrittori vengono usati tutti con lo stesso peso.

Il percettrone è programmato in maniera tale da apprendere proprio come pesare i descrittori. In particolare, è basato su una regressione lineare, che effettua la catalogazione dello spazio dei descrittori trovando l'iperpiano che li divide in 'positivi' e 'negativi'. Tale iperpiano è un confine decisionale lineare.

L'algoritmo si ispira al funzionamento dei neuroni: ogni neurone è collegato a una rete di altri neuroni, da ciascuno dei quali riceve un segnale. In base alla combinazione dei segnali, il neurone può venire attivato (stato 'positivo') o meno (stato 'negativo'). Inoltre, 'decide' la forza del segnale che esso manderà agli altri neuroni della rete.

Algoritmi di apprendimento – Percettrone

Esempio di funzionamento del percettrone: si immagini un percettrone che riceve un segnale da D input $\mathbf{X} = [x_1, \dots, x_D]^{\mathrm{tr}}$ e usa i pesi che ha memorizzato, $\mathbf{w} = [w_1, \dots, w_D]^{\mathrm{tr}}$ per calcolare la funzione:

$$a = \sum_{d=1}^{D} w_d x_d + b$$

dove b è un valore di soglia e a è l'attivazione. Se $a > 0$ l'attivazione è positiva, viceversa è negativa.

I pesi vengono appresi dal percettrone in maniera iterativa a partire dagli esempi, reiterando un numero N di volte su tutti gli esempi. Siano $[\mathbf{X}, y]$ gli esempi con $\mathbf{X} \in \Re^D$, e $y = \pm 1$ (catalogazione binaria), allora l'addestramento segue questo ciclo:

$\mathbf{w} = \mathbf{0}$
$b = 0$
`for i in range (0, N):`
    `for all` $[\mathbf{X}, y]$`:`
        $a = \sum_{d=1}^{D} w_d x_d + b$
        `if` $y * a \leq 0$`:`
            $\mathbf{w} = \mathbf{w} + y\mathbf{X}$
            $b = b + y$

Tale algoritmo va applicato sequenzialmente a tutti gli esempi.

L'algoritmo è **guidato dall'errore**: quando un nuovo esempio viene processato, i pesi vengono aggiornati solo se la previsione fatta sull'attuale set di pesi è sbagliata.

Il numero di iterazioni, N, di addestramento è l'iperparametro di questo algoritmo.

Il limite è che il confine decisionale deve essere lineare.

Algoritmi di apprendimento – Percettrone

Apprendimento multiclasse: un percettrone è un algoritmo che impara a compiere una classificazione binaria. Nel caso la classificazione debba essere compiuta tra $K > 2$ classi, ci sono due approcci.

One Versus All (OVA): si addestrano K percettroni, ciascuno dei quali conosce lo stato degli altri durante l'apprendimento. L'i-esimo percettrone considera positivi gli esempi di classe i e negativi gli altri. Durante il test, il percettrone che ottiene l'attivazione positiva 'vince' sugli altri. In caso di parità, si sceglie a caso.

All Versus All (AVA): la decisione del vincitore viene fatta come in un torneo a squadre che si affrontano a coppie.

Python Time

ML_example_3.ipynb: Implementazione di un percettrone per catalogare un acido in forte o debole.

Dataset

Rappresentazione SMILES degli acidi, valore della pk_a e indicazione della forza: 1 se forte, -1 se debole.

Descrittori

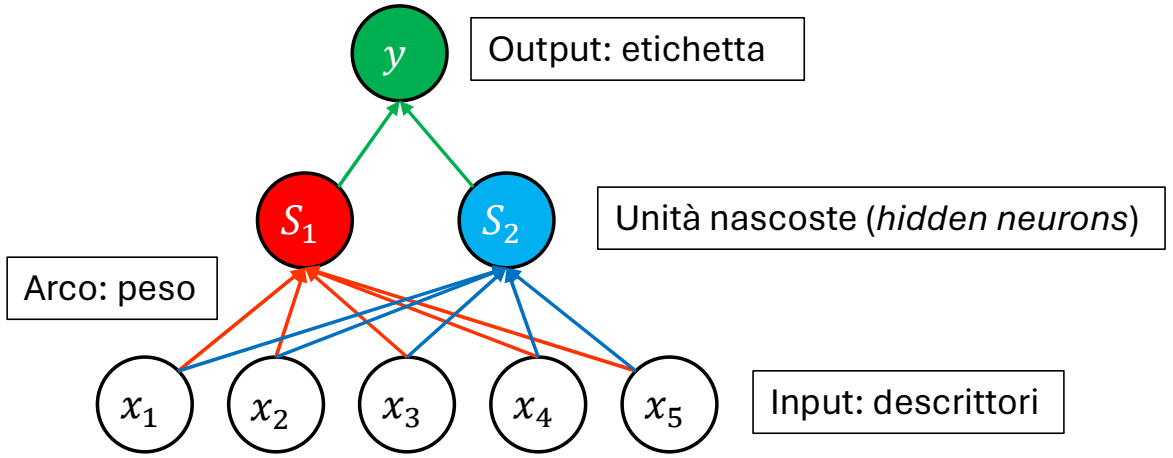
Polarizzabilità di atomi e legami, numero di atomi accettori e di atomi donatori di legami H.

specie	pka	forza
OCl(=O)(=O)=O	-10.000	1
[IH]	-9.505	1
[BrH]	-9.000	1
[ClH]	-6.114	1
OS(=O)(=O)O	-3.000	1
O[N+]([O-])=O	-1.380	1
OC(=O)C(=O)O	1.268	-1
OS(O)=O	1.886	-1
OS([O-])(=O)=O	2.000	-1
OP(=O)(O)O	2.149	-1
ON=O	3.143	-1
[FH]	3.180	-1
C(=O)O	3.745	-1
OC(=O)c1ccccc1	4.201	-1
C(=O)(C(=O)[O-])O	4.268	-1
CC(=O)O	4.745	-1
[O-]C([O-])=O	6.357	-1
[SH2]	6.959	-1
OP(=O)(O)[O-]	7.201	-1
[SH-]	2.208	-1
OCl	7.538	-1
C#N	9.208	-1
[NH4+]	9.237	-1
OB(O)O	10.000	-1
OC([O-])=O	10.328	-1
[O-]P(=O)(O)[O-]	12.377	-1
B(O)(O)[O-]	12.745	-1
B(O)([O-])[O-]	13.796	-1

Algoritmi di apprendimento – Reti Neurali

Reti neurali (NN – *neural networks*): il percettrone è un algoritmo lineare; a differenza degli alberi decisionali e del *clustering*, non può apprendere confini decisionali non lineari.

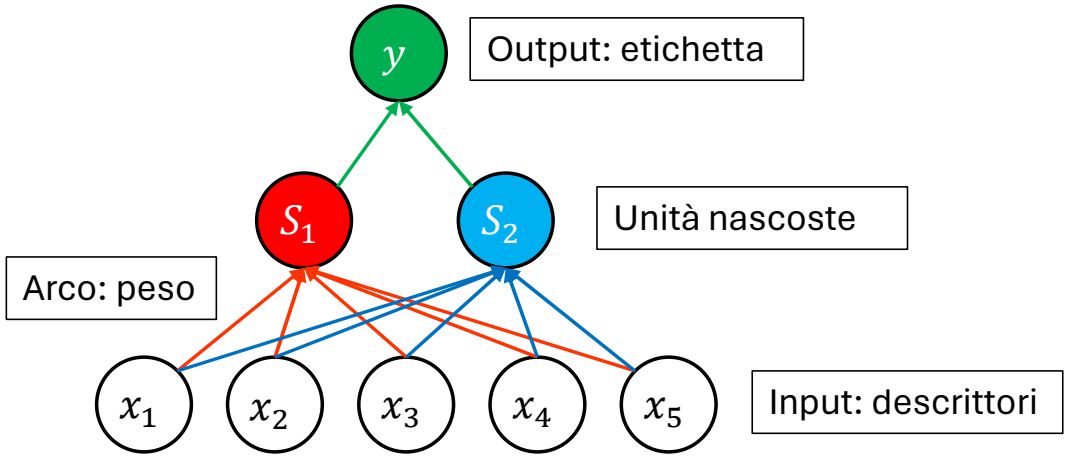
Per passare a un approccio non lineare si possono connettere tra loro più percettroni a formare una **rete neurale**. Un esempio di rete a due livelli (*layers*) è rappresentato nella figura qui sotto.



A differenza dei percettroni, le unità nascoste calcolano funzioni non lineari. Il peso calcolato dall'i-esima unità nascosta, dato il set di pesi che derivano da tutti i descrittori a tale unità ($\mathbf{w}_i$) è

$$h_i = f(\mathbf{w}_i \cdot \mathbf{x})$$

Algoritmi di apprendimento – Reti Neurali



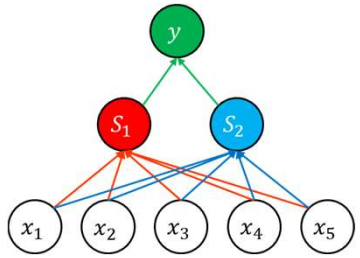
In genere viene scelta la tangente iperbolica: $f(a) = \tanh(a)$.

Il livello nascosto calcola i pesi della combinazione lineare dei valori di f_i di ogni nodo in tale livello. Siano v_i tali pesi, la rete neurale calcola l'etichetta come:

$$y = \sum_i v_i \tanh(\mathbf{w}_i \cdot \mathbf{x})$$

Una rete neurale a due livelli può rappresentare una qualsiasi funzione entro un arbitrario errore, ϵ.

Algoritmi di apprendimento – Reti Neurali



Come regola generale, dati N esempi e D descrittori, il numero di nodi nascosti deve essere $\lfloor N/D \rfloor$.

A parità di numero di descrittori, quindi, il numero di nodi nascosti che può essere scelto cresce con il numero di esempi a disposizione.

Per addestrare una rete neurale bisogna trovare il set di pesi che minimizzi una funzione errore. Questo può essere l'errore quadratico:

$$\min_{\mathbf{w}_i,\mathbf{v}} \sum_{n=1}^{N} \frac{1}{2} \left[y_n - \sum_i v_i \tanh(\mathbf{w}_i \cdot \mathbf{x}_n) \right]^2$$

Si tratta di un problema di ottimizzazione di un insieme di equazioni non lineari che derivano da calcolo del gradiente della funzione scritta sopra rispetto a tutti i pesi.

L'utilizzo di reti neurali a molti livelli nascosti è alla base del ***deep learning***.

Python Time

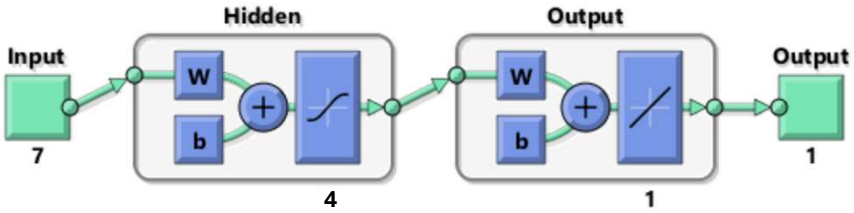
ML_example_4.ipynb: Regressione di temperatura di ebollizione ed entalpia standard di formazione di derivati del metano usando una rete neurale.

Dataset
Consideriamo solo derivati del metano con alogenuri, quindi i sostituenti: H, F, Cl, Br, I.

Descrittori

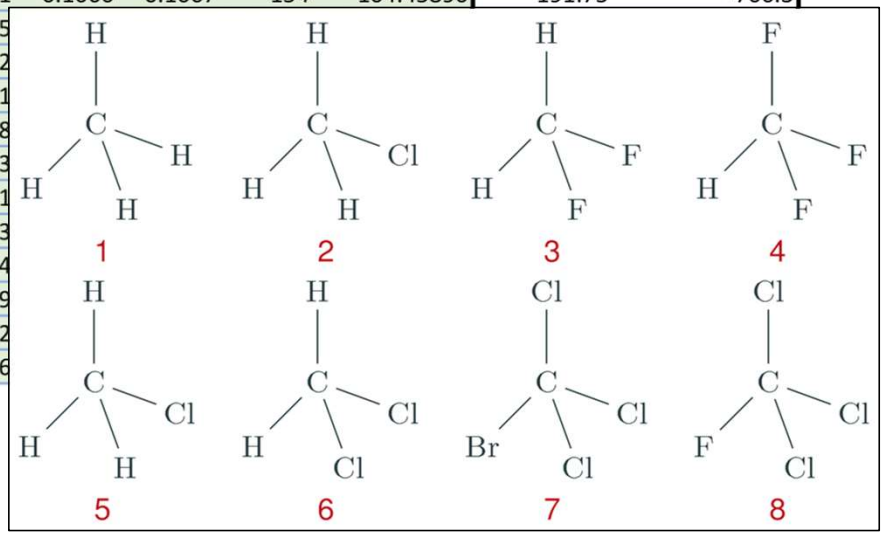
- costante dielettrica della fase liquida
- momento di dipolo elettrico molecolare
- componenti principali del tensore d'inerzia molecolare
- raggio di van der Walls molecolare
- massa molare

Algoritmo di apprendimento: rete neurale a 2 livelli, con 4 unità nascoste.



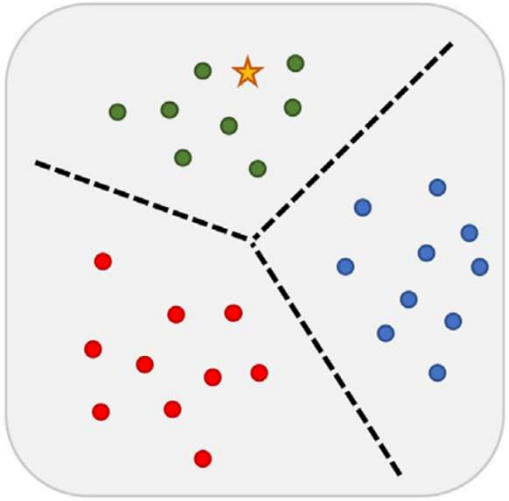
Python Time

Cabcd	ε	d	l_1	l_2	l_3	r	M	p_{eb}	ΔH_f (g)
CH4	8.8	0.0005	0.0032	0.0032	0.0032	120	16.04246	111.75	-74.87
CF4	15.92	0.0027	0.0909	0.091	0.091	147	88.00430	145.3	-930
CCl4	12.64	0.0015	0.2927	0.2929	0.2929	175	153.82270	349.87	-95.8136
CBr4	11.84	0.0011	0.7934	0.7938	0.7938	185	331.62670	462.8	79.496
CI4	10.64	0.0003	1.5348	1.5356	1.5356	198	519.62670	602.35	267.94336
CH3F	10.58	2.2975	0.0032	0.0361	0.0361	126.75	34.03292	194.75	-233.89
CH2F2	12.36	2.6538	0.0252	0.0471	0.069	133.5	52.02338	221.55	-452.3
CHF3	14.14	2.3004	0.0581	0.0581	0.0909	140.25	70.01384	191.05	-695.4
CH3Cl	9.76	1.4549	0.0032	0.1118	0.1118	133.75	50.48752	249.06	-81.9
CH2Cl2	10.72	1.6805	0.0757	0.148	0.2204	147.5	84.93258	313.15	-94.5584
CHCl3	11.68	1.4566	0.1843	0.1843	0.2927	161.25	119.37764	334.32	-102.9264
CH3Br	9.56	1.1726	0.0032	0.2996	0.2996	136.25	94.93852	276.65	-35.4
CH2Br2	10.32	1.3544	0.201	0.3983	0.5961	152.5	173.83458	370.15	-14.76952
CHBr3	11.08	1.1739	0.4974	0.4974	0.7934	168.75	252.73064	414.25	16.736
CH3I	9.26	0.6156	0.0032	0.5777	0.5777	139.5	141.93852	315.58	13.76536
CH2I2	9.72	0.711	0.3865	0.769	1.1523	159	267.83458	455.15	117.5704
CHI3	10.18	0.6161	0.9611	0.9611	1.5348	178.5	393.73064	491.15	210.8736
CF3Cl	15.1	0.8417	0.091	0.1666	0.1667	154	104.45890	191.75	-706.3
CF2Cl2	14.28	0.9717	0.1415						
CFCl3	13.46	0.8411	0.2172						
CF3Br	14.9	1.124	0.091						
CF2Br2	13.88	1.2978	0.2668						
CFBr3	12.86	1.1238	0.5303						
CF3I	14.6	1.681	0.091						
CF2I2	13.28	1.9412	0.4523						
CFI3	11.96	1.6816	0.994						
CCl3Br	12.44	0.2818	0.2929						
CCl2Br2	12.24	0.3252	0.4182						
CClBr3	12.04	0.2812	0.606						



Unsupervised Learning

Clustering



Esempio: quali sono le configurazioni rappresentative per l'azione biologica di un enzima?

Apprendimento senza supervisione

In alcuni problemi si cercano correlazioni tra i dati, ma senza avere una guida sulle proprietà che sono alla base di tali correlazioni. Lo scopo è quello di fare emergere nuove proprietà non note a priori.

L'obiettivo più comune dell'apprendimento senza supervisione è quello di raggruppare i dati in funzione di una qualche misura della loro somiglianza. Si tratta di un problema di *clustering*.

L'algoritmo di ***k-means clustering*** è solitamente utilizzato per l'apprendimento. Sia $\mathbf{x}_n$ l'n-esimo array di dati di input, e K il numero di cluster da costruire. L'algoritmo procede in questo modo:

1. inizializza casualmente i centri $\boldsymbol{\mu}_k$ dei cluster,
2. assegna ogni $\mathbf{x}_n$ al cluster k tale per cui il punto sia più vicino (distanza Euclidea) al centro del cluster rispetto a tutti gli altri,
3. per ogni cluster, ricalcola i centri come media sui dati che fanno parte di quel cluster,
4. ritorna al punto 2 fino a convergenza (i centri non cambiano più).

Un altro obiettivo dell'apprendimento senza supervisione è quello della riduzione della dimensionalità: PCA, su questo è già stato esaustivo il collega.

Python Time

ML_example_5.ipynb: Catalogazione delle molecole sulla base della massima distanza tra atomi lungo le tre direzioni spaziali..

Dataset
Piccole molecole organiche costituite da H, C, O, N, S, F, Cl, Br, I.

Descrittori

Massima distanza tra gli atomi nelle tre direzioni dello spazio ottenute dopo aver centrato la molecola sul centro di massa e orientata lungo le direzioni principali di inerzia. Infine, i tre numeri vengono posti in ordine crescente.

Algoritmo di apprendimento: *k-means clustering*.