

DMIF, Università di Udine

---

# Tecnologie Digitali per il Cibo e la Ristorazione

*Data Mining e Machine Learning*

Andrea Brunello

[andrea.brunello@uniud.it](mailto:andrea.brunello@uniud.it)

A.A. 2021–2022

# Outline

- 1 Introduzione
- 2 Esplorazione e preparazione dei dati
- 3 La suite WEKA
- 4 Supervised learning
- 5 Unsupervised learning

# Introduzione

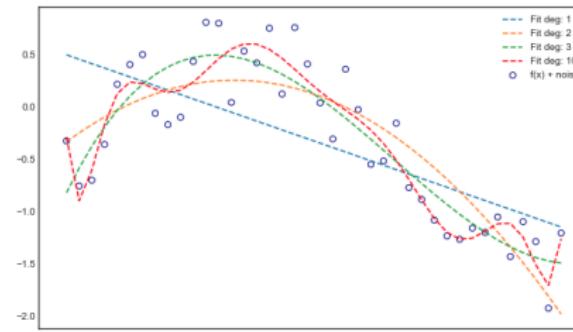
# Cos'è il Data Mining

- **Dati** ≈ **fatti** memorizzati, registrati
- L'**informazione** è costituita dall'insieme dei **concetti**, delle regolarità, degli schemi che si trovano “nascosti” fra i dati
- Il **Data Mining** si occupa dell'**estrazione** e della presentazione di **informazione** utile, precedentemente sconosciuta, ed implicitamente contenuta in una (grande) mole di dati
  - È un processo di astrazione, che porta alla generazione di un modello
- Il **Machine Learning** costituisce la “base tecnica” del Data Mining

# Modelli e realtà

*“Un modello è un’astrazione selettiva della realtà”*  
*(A. Einstein)*

- Nel nostro caso, la “realtà” è rappresentata dai dati che abbiamo a disposizione
- Costruiamo una descrizione semplificata di tale realtà
- Il modello è dunque una rappresentazione concettuale (semplificata) del mondo reale o di una sua parte, capace di spiegare un determinato fenomeno

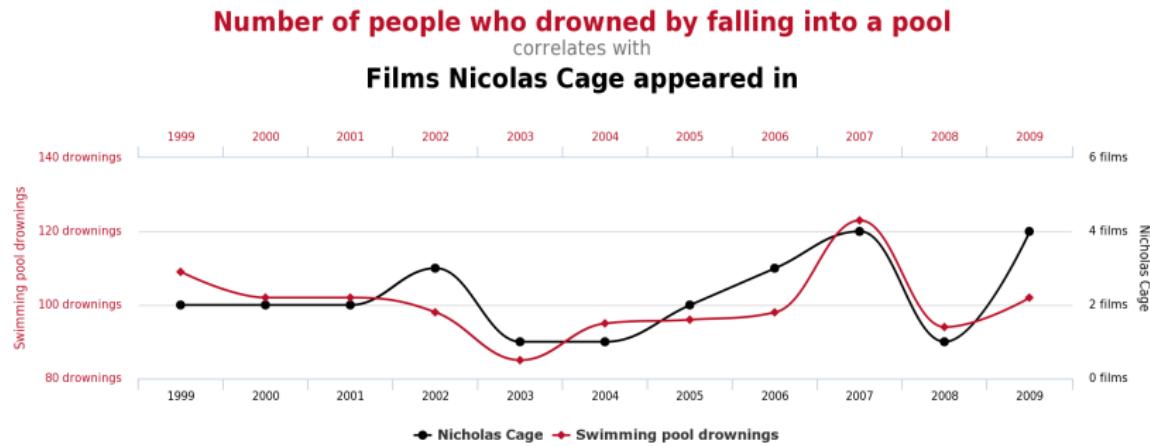


# Uso dei modelli

- Utilizzare i modelli/pattern appresi per:
  - **conoscere**: comprendere che determinate fasce di popolazione sono più propense ad acquistare un determinato bene
  - **inferire**: probabile guasto ad un macchinario, da un insieme di sintomi
  - **predire**: stabilire se e quale variazione nelle vendite risulterà da un aumento del budget pubblicitario
- Tali fini possono mescolarsi, si pensi alla ricerca di un modello in grado di fornire la valutazione di un'abitazione sulla base di diverse sue caratteristiche

# Rilevanza dei pattern scoperti

- Spesso i pattern scoperti risulteranno banali, frutto di correlazioni casuali, o non completamente corretti
- Ciò può essere dovuto a:
  - errori nei dati
  - caratteristiche del dominio



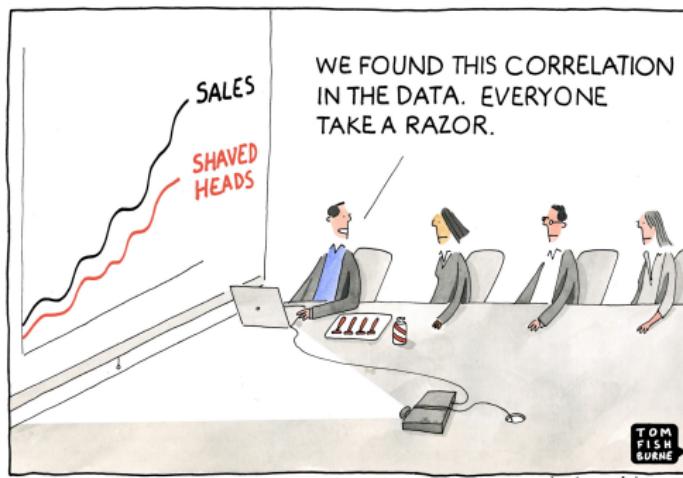
tylervigen.com

# Correlazione e rapporto di causa/effetto

*Correlation does not imply causation!*

Qualche altro esempio:

<https://www.tylervigen.com/spurious-correlations>



# Data Mining: Recap

Riassumendo:

- Il Data Mining sfrutta tecniche di Machine Learning
- per estrarre semi-automaticamente
- da (grandi) quantità di dati
- informazioni, pattern utili

Input del processo:

- istanze, esempi dei concetti che si vogliono apprendere

Output del processo:

- modelli
- tramite alcuni modelli, predizioni/classificazioni

# Tipologie di dato

- Il Data Mining può essere svolto su diverse tipologie di dato:
  - **strutturato:** dati tabellari
  - **non strutturato:** testo, immagini, audio, video
- Le tecniche di machine learning classiche presuppongono la trasformazione di dati non strutturati in strutturati (es., per l'analisi di dati testuali)
- Gli approcci *deep learning* (reti neurali) consentono di trattare direttamente i dati non strutturati
- Noi tratteremo il caso dei dati strutturati, unicamente (eventualmente) accennando ai dati non strutturati per quanto riguarda:
  - trasformazione di dati testuali in formato tabellare
  - riconoscimento delle immagini tramite reti neurali (*Google Lens*)

# Dati strutturati

Un dataset tabellare utilizzato dagli algoritmi classici di machine learning ha la seguente forma:

Temperature	Outlook	Humidity	Windy	Played?
Mild	Sunny	80	No	Yes
Hot	Sunny	75	Yes	No
Hot	Overcast	77	No	Yes
Cool	Rain	70	No	Yes
Cool	Overcast	72	Yes	Yes
Mild	Sunny	77	No	No

- ciascuna riga rappresenta **un'istanza**, i.e, un esempio del concetto che si vuole apprendere
- le righe sono fra loro indipendenti
- ciascuna istanza descrive il valore di un dato insieme di attributi (colonne)

# Il processo di Data Mining

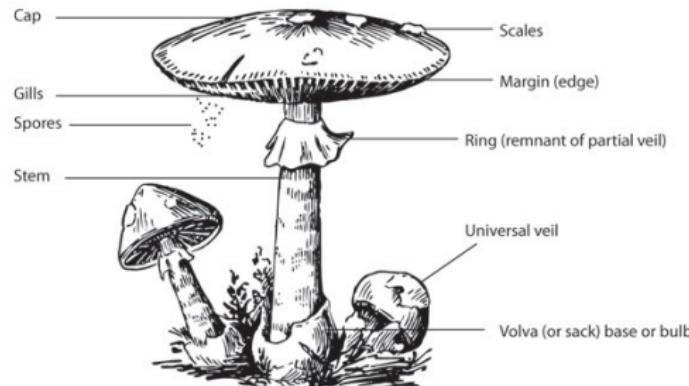
In genere, il processo di Data Mining si articola come segue:

- il tutto inizia con il porsi una specifica **domanda**
- in seguito, si raccolgono **dati** da utilizzare come input
- attraverso un processo di *esplorazione*, si determinano le caratteristiche dei dati (valori nulli, outlier, ecc) e si svolge una fase di pre-processing
- viene selezionato un insieme di attributi ritenuti essere importanti su tali dati, e per il fine che si vuole ottenere
- si applica un **algoritmo** di machine learning sul dataset così definito, in modo da “addestrare” un modello
- dopo un’eventuale fase di *tuning*, il modello prodotto dall’algoritmo viene **valutato**, ed è infine pronto per essere studiato ed utilizzato

# Un primo esempio: Mushroom dataset

Ripercorriamo ora il processo di Data Mining con un esempio focalizzato sulla predizione.

- **Domanda:** è possibile distinguere automaticamente fra funghi velenosi e commestibili?
- **Dati:** insieme di 8124 istanze di funghi già classificati, ciascuna descritta da 22 attributi, più la classe;
  - <https://archive.ics.uci.edu/ml/datasets/mushroom>



# Un primo esempio: Mushroom dataset (2)

Contenuto delle prime 11 righe del *dataset*

cap_shape	cap_surface	cap_color	bruises_or_no	odor	gill_attachment	gill_spacing	gill_size	gill_color	stalk_shape	stalk_root	stalk_surface_above_ring
convex	smooth	brown	bruises	pungent	free	close	narrow	black	enlarging	equal	smooth
convex	smooth	yellow	bruises	almond	free	close	broad	black	enlarging	club	smooth
bell	smooth	white	bruises	anise	free	close	broad	brown	enlarging	club	smooth
convex	scaly	white	bruises	pungent	free	close	narrow	brown	enlarging	equal	smooth
convex	smooth	gray	no	none	free	crowded	broad	black	tapering	equal	smooth
convex	scaly	yellow	bruises	almond	free	close	broad	brown	enlarging	club	smooth
bell	smooth	white	bruises	almond	free	close	broad	gray	enlarging	club	smooth
bell	scaly	white	bruises	anise	free	close	broad	brown	enlarging	club	smooth
convex	scaly	white	bruises	pungent	free	close	narrow	pink	enlarging	equal	smooth
bell	smooth	yellow	bruises	almond	free	close	broad	gray	enlarging	club	smooth
convex	scaly	yellow	bruises	anise	free	close	broad	gray	enlarging	club	smooth

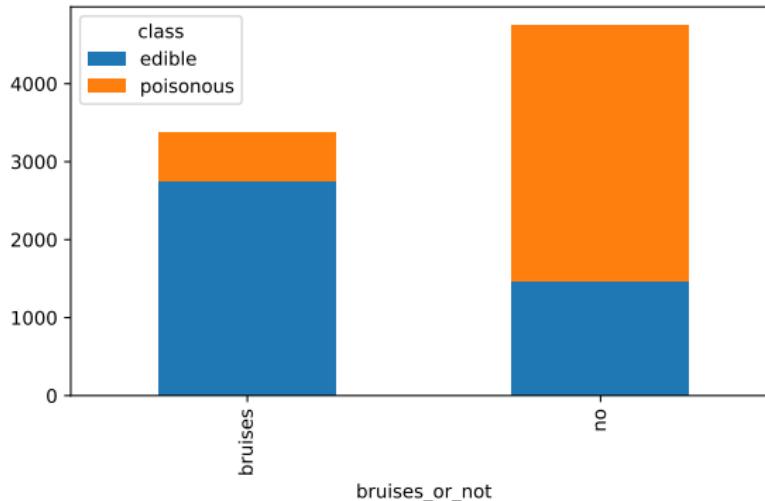
stalk_surface_below_ring	stalk_color_above_ring	stalk_color_below_ring	veil_type	veil_color	ring_number	ring_type	spore_print_color	population	habitat	class
smooth	white	white	partial	white	1	pendant	black	scattered	urban	poisonous
smooth	white	white	partial	white	1	pendant	brown	numerous	grasses	edible
smooth	white	white	partial	white	1	pendant	brown	numerous	meadows	edible
smooth	white	white	partial	white	1	pendant	black	scattered	urban	poisonous
smooth	white	white	partial	white	1	evanescent	brown	abundant	grasses	edible
smooth	white	white	partial	white	1	pendant	black	numerous	grasses	edible
smooth	white	white	partial	white	1	pendant	black	numerous	meadows	edible
smooth	white	white	partial	white	1	pendant	brown	scattered	meadows	edible
smooth	white	white	partial	white	1	pendant	black	several	grasses	poisonous
smooth	white	white	partial	white	1	pendant	black	scattered	meadows	edible
smooth	white	white	partial	white	1	pendant	brown	numerous	grasses	edible

Il 52% delle istanze è di classe *edible*, il 48% *poisonous*

# Un primo esempio: Mushroom dataset (3)

Selezione degli **attributi**:

- processi di *attribute selection*, es., sfruttando indici statistici
- esplorazione e selezione manuale



# Un primo esempio: Mushroom dataset (4)

Idea:

- ① partizioniamo le istanze sulla base del valore dell'attributo *bruises\_or\_not*
- ② stabiliamo che un'istanza è di classe *edible* se e solo se *bruises\_or\_not = bruises*

~~ basandoci su questa semplice regola (il modello che rappresenta la realtà descritta dai nostri dati), otteniamo un'accuratezza di classificazione del 74%

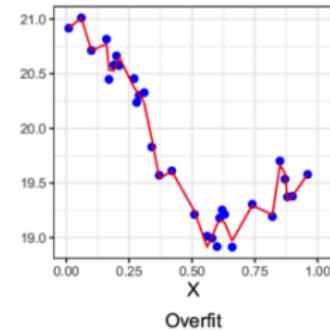
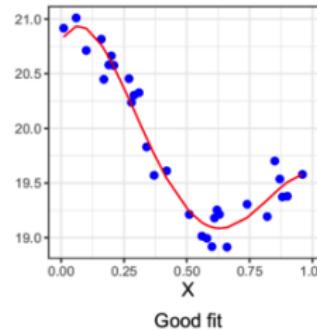
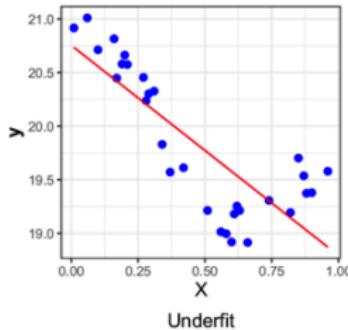
**N.B.:** il modello non è esente da errore, vale a dire, non descrive perfettamente i dati che osserviamo; questo, tendenzialmente, è voluto

# Modelli e generalizzazione

- Raramente le istanze presenti in un dataset saranno in grado di coprire tutti i casi possibili per un dato dominio
- *Esempio:* dataset di case, in cui ciascuna istanza descrive una casa (superficie, numero di stanze, presenza del giardino, ecc.) ed il suo valore
- Dunque, l'algoritmo di machine learning deve essere in grado di ottenere un modello in grado di generalizzare (di risultare valido) anche nei confronti delle istanze che non ha potuto osservare nella fase di addestramento
- *Esempio:* un modello in grado di prevedere il valore di una casa sulla base di alcune sue caratteristiche

# Modelli e generalizzazione (2)

- La capacità dei modelli di generalizzare, come vedremo, è un aspetto fondamentale dell'intero processo, quando si vogliono effettuare predizioni
- Non è detto che un modello di predizione che ottiene ottimi risultati sui dati forniti per il training si comporterà bene anche su nuovi dati
- Intuitivamente, questo avviene perché il modello potrebbe aver appreso dei pattern dovuti a “rumore” (*overfitting*)



# Supervised vs Unsupervised learning

Distinzione fondamentale, a seconda dell'obiettivo del processo:

- **Supervised learning:** all'algoritmo di learning viene fornito un risultato noto per ciascuna istanza del dataset (attributo *obiettivo*) e si punta ad ottenere un modello in grado di determinare tale valore per nuove istanze sulla base di un insieme di attributi detti *predittori*
  - è il caso del modello costruito a mano, con l'obiettivo di determinare se un fungo è velenoso o meno (obiettivo) sulla base della presenza di annerimenti (predittore)
- **Unsupervised learning:** non si cerca di prevedere il valore di uno specifico attributo, ma viene ricercata ogni possibile associazione/correlazione fra gli attributi

# Problemi di regressione e di classificazione

Nel contesto del *supervised learning*, a seconda della tipologia dell'attributo obiettivo, distinguiamo problemi di

- **classificazione:** si vuole assegnare a ciascuna istanza uno di un insieme finito di valori (*classificatori discreti*); in altri casi, viene restituita la probabilità di appartenere a ciascuna classe (*classificatori continui*)
- **regressione:** si vuole assegnare a ciascuna istanza un valore numerico continuo

# Esplorazione e preparazione dei dati

# Introduzione

- La fase più importante e onerosa dell'intero processo di data mining è l'esplorazione e preparazione dei dati
- Tale fase riveste un'importanza cruciale, in quanto tutte le analisi successive dipendono da essa
- Alcuni task tipici:
  - scelte riguardanti la codifica degli attributi
  - rilevamento e trattamento dei valori nulli
  - rimozione delle colonne con bassa varianza
  - identificazione degli outlier
  - dipendenze fra le colonne
  - feature selection, rimozione delle colonne inutili

# Codifica degli attributi

- Esempio: attributo relativo al *ph*: mantenere i valori numerici o utilizzare dei valori “più grossolani” categoriali (acido, neutro, basico)?
- Quale precisione considerare per gli attributi numerici? (es. numeri interi, con virgola, quanti spazi dopo la virgola, ...)
  - ad esempio, se un sensore di temperatura è in grado di fornire una misurazione esatta alla seconda cifra decimale, è inutile mantenere un numero come 37.234664367

# Valori nulli

Cosa fare nel caso in cui alcune istanze presentino valori non noti (nulli)?

Temperature	Outlook	Humidity	Windy	Played?
Mild	Sunny	?	No	Yes
Hot	?	75	?	No
Hot	Overcast	77	No	Yes

Diverse soluzioni, ad esempio:

- rimuovere tali istanze
- rimuovere le colonne con troppi valori nulli
- sostituire i valori nulli con un valore di default
- sostituire i valori nulli con media/moda della colonna
- altre tecniche di imputazione più sofisticate

Inoltre, un ruolo importante riveste la “tipologia” di valore nullo (perché è nullo?)

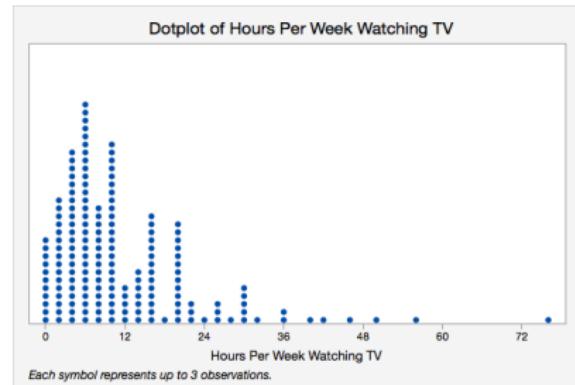
# Colonne con bassa varianza

- Nel dataset possono esserci colonne che presentano un unico valore, o con una distribuzione dei valori estremamente sbilanciata
- *Tipicamente*, tali colonne sono poco informative
- Un algoritmo di machine learning può accorgersi di ciò e non considerarle per la generazione dei modelli
- Tuttavia, rimuoverle a priori porta ad una maggiore efficienza del processo di learning

ID	season	holiday	workingday	weather	f1	temp	atemp	humidity	windspeed	count
AB101	1	0	0	1	7	9.84	14.395	81	0.0000	16
AB102	1	0	0	1	7	9.02	13.635	80	0.0000	40
AB103	1	0	0	1	7	9.02	13.635	80	0.0000	32
AB104	1	0	0	1	7	9.84	14.395	75	0.0000	13
AB105	1	0	0	1	7	9.84	14.395	75	0.0000	1
AB106	1	0	0	2	7	9.84	12.880	75	6.0032	1
AB107	1	0	0	1	7	9.02	13.635	80	0.0000	2
AB108	1	0	0	1	7	8.20	12.880	86	0.0000	3
AB109	1	0	0	1	7	9.84	14.395	75	0.0000	8
AB110	1	0	0	1	7	13.12	17.425	76	0.0000	14

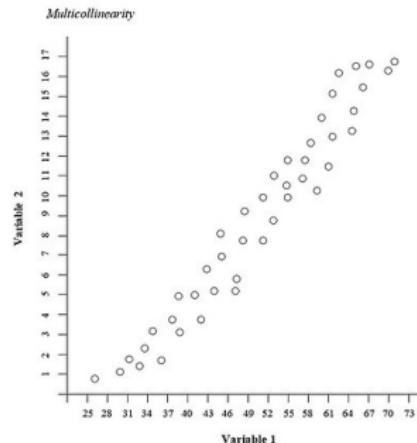
# Outlier

- In statistica, outlier è un termine utilizzato per definire, in un insieme di osservazioni, un valore anomalo, ossia un valore chiaramente distante dalle altre osservazioni
- Un outlier può essere dovuto ad errori in fase di rilevamento o raccolta dei dati
- Non esiste una definizione matematica di outlier
- Attenzione alla loro rimozione o correzione



# Dipendenza fra le colonne

- Può capitare che i valori di alcune colonne siano fortemente in relazione (correlati) fra loro
- In tal caso, è sufficiente mantenere solamente una colonna
- Ciò non solo rende più efficiente la fase di analisi, ma consente ai modelli addestrati di determinare in modo corretto il “contributo/ruolo” di ciascun predittore



*Note.* The plot depicts the predictor variables, Variable 1 and Variable 2, as highly linearly related.

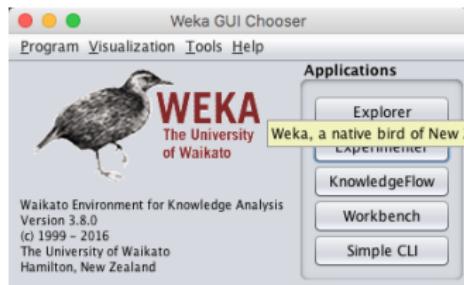
# Feature selection

- Il processo attraverso il quale vengono determinati gli attributi da mantenere e da scartare viene detto *feature selection*
- Nelle slide precedenti, abbiamo visto alcuni esempi di rimozione di colonne
- Vi sono in letteratura molte tecniche per svolgere la feature selection, in quanto essa gioca un ruolo fondamentale prima dell'analisi vera e propria:
  - creazione di modelli più semplici
  - creazione di modelli più accurati
  - creazione di modelli in grado di generalizzare meglio su nuovi dati

# La suite WEKA

# La suite WEKA

- WEKA, acronimo di “*Waikato Environment for Knowledge Analysis*”, è un software open source per l’apprendimento automatico sviluppato nell’università di Waikato in Nuova Zelanda
- Consente di svolgere tutte le fasi del processo di Data Mining attraverso un ambiente grafico
- Cross-platform, scritto in linguaggio Java



# Formato file ARFF

## Attribute Relationship File Format

```
@relation weather.symbolic ← Dataset name
@attribute outlook {sunny, overcast, rainy}
@attribute temperature {hot, mild, cool}
@attribute humidity {high, normal}
@attribute windy {TRUE, FALSE}
@attribute play {yes, no} ← Attributes
@data ← Target / Class variable
sunny,hot,high,FALSE,no
sunny,hot,high,TRUE,no
overcast,hot,high,FALSE,yes
rainy,mild,high,FALSE,yes
rainy,cool,normal,FALSE,yes
rainy,cool,normal,TRUE,no
overcast,cool,normal,TRUE,yes
sunny,mild,high,FALSE,no
sunny,cool,normal,FALSE,yes
rainy,mild,normal,FALSE,yes
sunny,mild,normal,TRUE,yes
overcast,mild,high,TRUE,yes
overcast,hot,normal,FALSE,yes
rainy,mild,high,TRUE,no ← Data Values
```

Equivalente ad un formato dei dati tabellare

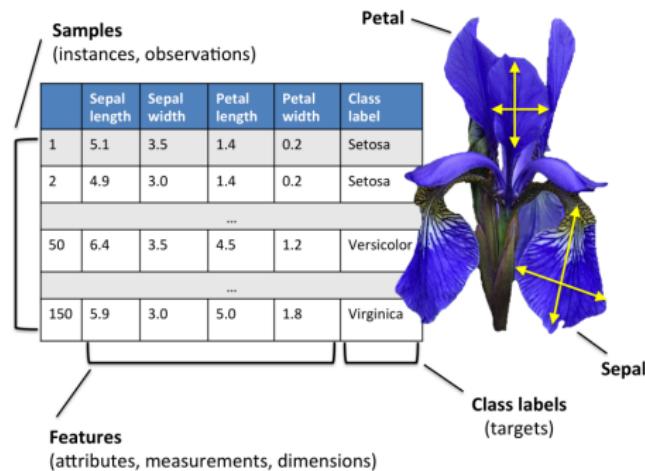
# Ottener WEKA

- Il software è scaricabile liberamente dal sito:  
<https://www.cs.waikato.ac.nz/ml/weka/>
- Libro su Data Mining/Machine Learning scritto dagli autori del software:  
<https://www.cs.waikato.ac.nz/ml/weka/book.html>
- Un tutorial interessante preparato da IBM:
  - Introduzione e regressione:  
<https://developer.ibm.com/articles/os-weka1/>
  - Classificazione e clustering:  
<https://developer.ibm.com/articles/os-weka2/>
- Carichiamo in WEKA il dataset *mushrooms.arff*

# Supervised learning

# Concetti fondamentali

Scopo del supervised learning è quello di imparare un modello che sia in grado, data in input un'istanza con i valori di un insieme di attributi *predittori*, di generare in output il valore di un altro attributo, detto *obiettivo*



# Concetti fondamentali (2)

- Formalmente, dati  $A_1, A_2, \dots, A_n$  attributi predittori e  $L$  l'attributo obiettivo, vogliamo apprendere una funzione:

$$f : A_1 \times A_2 \times \cdots \times A_n \rightarrow L$$

- Ad esempio:

$$f : \text{sepal\_l} \times \text{sepal\_w} \times \text{petal\_l} \times \text{petal\_w} \rightarrow \text{class\_label}$$

$$f(4.9, 3.0, 1.4, 0.2) = \text{Setosa}$$

# Task di classificazione

- Nel caso di task di classificazione, il valore dell'obiettivo  $L$  è *discreto* (categoriale)
- Ad esempio, nel dataset *Iris*, è uno fra le etichette: *Setosa*, *Virginica*, *Versicolor*



Iris Versicolor

Iris Setosa

Iris Virginica

# Alcuni modelli per la classificazione

- Instance-based learning
- Naive Bayes
- Alberi di decisione

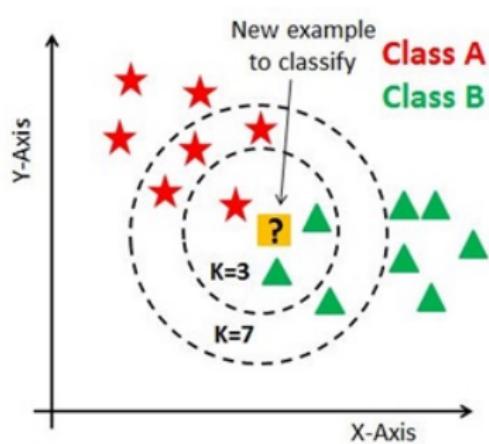
# Instance-based learning

L'idea è la seguente:

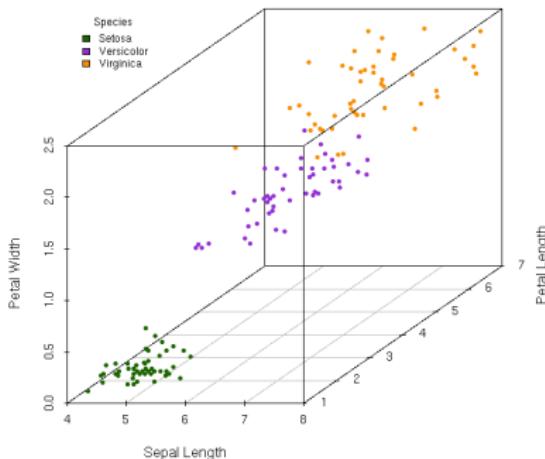
- si ha a disposizione un insieme di istanze di training delle quali è nota la classe
- data una nuova istanza per la quale non conosciamo la classe, essa viene ricondotta ad uno o più dei casi noti:
  - nearest neighbour  $\rightsquigarrow$  classe dell'istanza più simile
  - k-nearest neighbour  $\rightsquigarrow$  moda delle classi delle  $k$  istanze più simili
- diverse nozioni di distanza (es., distanza Euclidea)
- si osservi che, a differenza degli altri casi che vedremo, non viene generato un vero e proprio "modello"

# Instance-based learning

## Esempio



3-D Scatterplot of Iris Data



Dist. Euclidea fra due punti  $a = (a_1, \dots, a_n)$  e  $b = (b_1, \dots, b_n)$ :

$$d(a, b) = \sqrt{\sum_{i=1}^n (a_i - b_i)^2}$$

# Naive Bayes

- Famiglia di classificatori probabilistici
- L'idea è che il valore di ciascun attributo di un'istanza contribuisca in modo *indipendente*, e con *lo stesso peso*, alla determinazione della classe dell'istanza
- Tali assunzioni sono molto forti, e da esse deriva il termine *Naive*
- Tuttavia, il metodo Naive Bayes funziona sorprendentemente bene nei casi reali

# Naive Bayes

## Weather/Play dataset

Si consideri il seguente dataset, in cui la classe è rappresentata dall'attributo *play*

	outlook text	temperature text	humidity text	windy text	play text
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

# Naive Bayes

## Weather/Play dataset: Dati riassuntivi

A partire dal dataset costruiamo la seguente tabella riassuntiva

**Table 4.2** Weather Data with Counts and Probabilities

Outlook			Temperature			Humidity			Windy			Play	
	yes	no		yes	no		yes	no		yes	no	yes	no
sunny	2	3	hot	2	2	high	3	4	false	6	2	9	5
overcast	4	0	mild	4	2	normal	6	1	true	3	3		
rainy	3	2	cool	3	1								
sunny	2/9	3/5	hot	2/9	2/5	high	3/9	4/5	false	6/9	2/5	9/14	5/14
overcast	4/9	0/5	mild	4/9	2/5	normal	6/9	1/5	true	3/9	3/5		
rainy	3/9	2/5	cool	3/9	1/5								

- Abbiamo contato le occorrenze di ogni singolo attributo, in funzione del valore (yes/no) di *Play* (es., ci sono 2 istanze con valore *Outlook=sunny* quando *Play=yes*)
- Abbiamo poi derivato le frequenze, dividendo il conteggio di ogni singolo valore per il totale della colonna
- Abbiamo calcolato le frequenze dei valori (yes/no) di *Play*

# Naive Bayes

## Classificazione di una nuova istanza

Outlook	Temperature	Humidity	Windy	Play
sunny	cool	high	true	?

- Otteniamo la “tendenza” a giocare moltiplicando le frazioni corrispondenti:

$$(2/9) * (3/9) * (3/9) * (3/9) * (9/14) = 0.0053$$

- Allo stesso modo otteniamo la “tendenza” a non giocare:

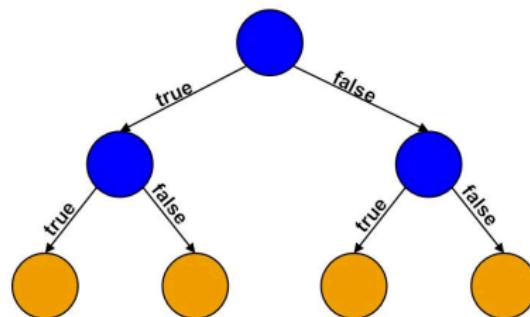
$$(3/5) * (1/5) * (4/5) * (3/5) * (5/14) = 0.0206$$

- Possiamo trasformare i numeri in probabilità:
  - a favore:  $0.0053 / (0.0053 + 0.0206) = 20.5\%$
  - contro:  $0.0206 / (0.0053 + 0.0206) = 79.5\%$

# Alberi di decisione

Gli alberi di decisione rappresentano in modo intuitivo l'output di un processo di classificazione

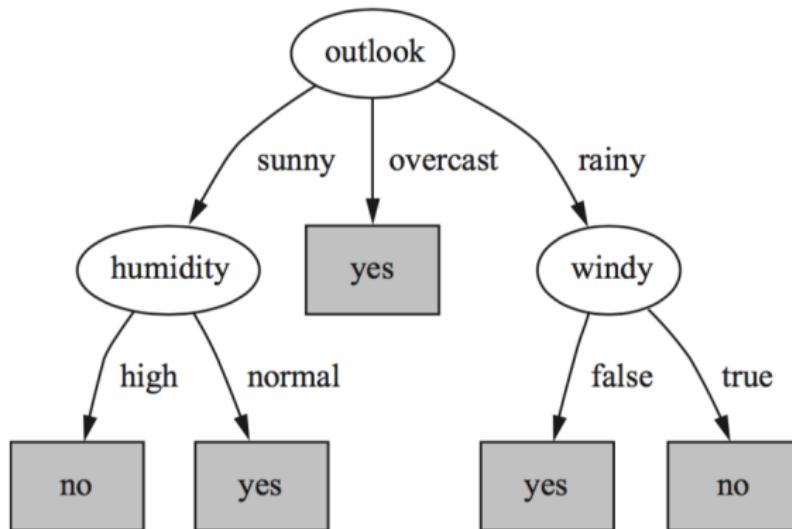
- ciascun nodo interno corrisponde alla valutazione di un determinato attributo
- la classificazione di un'istanza avviene partendo dalla radice e scendendo via via sino a giungere ad una foglia etichettata con una determinata classe (o insieme di classi, distribuzione di probabilità)



# Alberi di decisione

## Esempio: Weather/Play dataset

L'albero di decisione classifica correttamente tutte le istanze del *Weather/Play* dataset

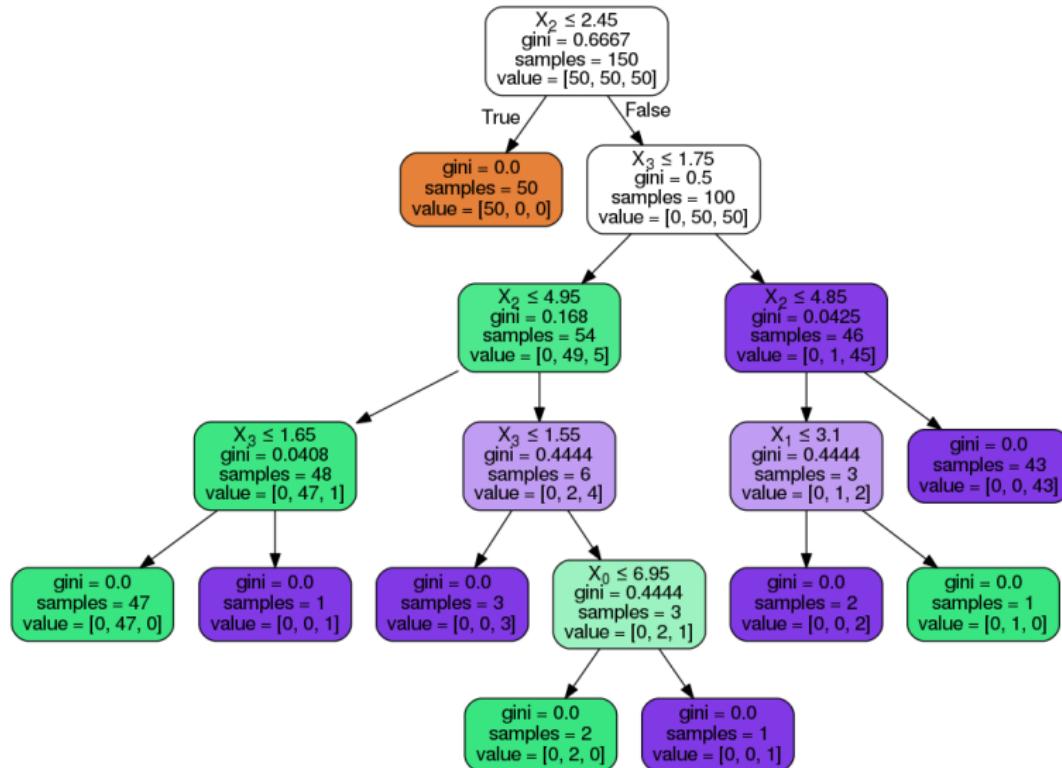


Osserviamo che non viene mai presa in considerazione la *temperatura*

- La costruzione parte dalla radice, che “contiene” tutte le istanze del dataset
- Viene individuato il miglior test su un attributo per suddividere le istanze della radice in due o più figli
- L’idea è generare nodi figli in modo tale da massimizzare la loro purezza rispetto alla classe delle istanze che vi appartengono
- Il processo termina quando si giunge ad una foglia:
  - nodo totalmente puro rispetto alle classi
  - nodo che contiene il numero minimo di istanze che si vogliono considerare
- Ciascuna foglia ha in modo naturale associata una distribuzione di probabilità sulle classi

# Alberi di decisione

## Esempio



# Alberi di decisione

## Regole di classificazione

- Si osservi che, a partire da un albero di decisione, è possibile ricavare un insieme di *regole di classificazione*
- Ciascuna regola si riferisce ad una classe
- Ad esempio, riferendoci all'albero relativo al dataset *Weather/Play*:
  - $(outlook=sunny \text{ AND } humidity=high) \text{ OR } (outlook=rainy \text{ AND } windy=true) \rightarrow no$
  - $(outlook=sunny \text{ AND } humidity=normal) \text{ OR } (outlook=overcast) \text{ OR } (outlook=rainy \text{ AND } windy=false) \rightarrow yes$
- Le regole consentono, oltre a derivare la previsione per una nuova istanza, anche di scoprire delle regolarità o proprietà che caratterizzano i dati considerati

# WEKA e task di classificazione

Si carichi ora in WEKA il dataset *mushrooms.arff*

- Il task sarà quello di prevedere se un fungo è commestibile o meno sulla base di un suo insieme di caratteristiche
- Quante istanze ci sono nel dataset? Quanti attributi?
- Quali sono i tipi dei vari attributi?
- Ci sono degli attributi inutili, che possono essere rimossi?
- Si costruisca un albero di decisione per prevedere l'attributo *class* del dataset
  - Clic sulla lingetta *Classify*
  - Clic sul pulsante *Choose* nel campo *Classifier*
  - Clic su *trees > J48*
  - Clic su *Use training set* su *Test options*
  - Clic sul pulsante *Start*

# Valutazione dei modelli di classificazione

- Una volta costruito un modello, è possibile ri-applicarlo per prevedere l'attributo obiettivo sui dati di training (in WEKA, *Test options = Use training set*)
- Ciò consente di confrontare le previsioni con i valori noti dell'attributo obiettivo
- La *matrice di confusione* ci consente di riassumere in modo sintetico il risultato della previsione

```
==== Confusion Matrix ====  
  
      a      b    <-- classified as  
3916    0 |    a = poisonous  
  0 4208 |    b = edible
```

# Valutazione dei modelli di classificazione

## Matrice di confusione

		Predicted	
		Negative (N) -	Positive (P) +
Actual	Negative -	True Negatives (TN)	False Positives (FP) <b>Type I error</b>
	Positive +	False Negatives (FN) <b>Type II error</b>	True Positives (TP)

A partire dalla matrice di confusione possiamo ricavare:

- accuratezza:  $(TP + TN) / (TP + TN + FP + FN)$
- precision:  $TP / (TP + FP)$
- recall:  $TP / (TP + FN)$
- F1 score:  $(2 * \text{precision} * \text{recall}) / (\text{precision} + \text{recall})$

# Valutazione dei modelli di classificazione

## Generalizzazione a più classi

- Consideriamo il dataset *iris.arff*, che riguarda la predizione della tipologia di fiore iris sulla base dei petali
- Costruendo un albero di decisione e valutandolo impostando *Test options* a *Use training set*, otteniamo una matrice di confusione simile alla seguente

==== Confusion Matrix ===

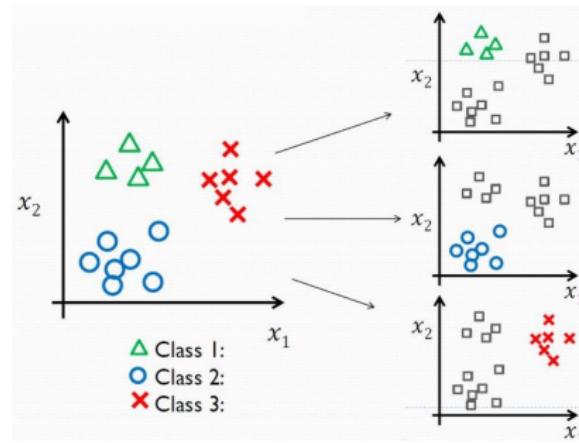
a	b	c	<-- classified as
15	0	0	a = Iris-setosa
0	19	0	b = Iris-versicolor
0	2	15	c = Iris-virginica

- La matrice di confusione generalizza in modo immediato al caso multi-classe
- L'accuratezza si ottiene sempre sommando i numeri sulla diagonale e dividendo per il numero totale di istanze
- Per calcolare precision, recall, e F1-score per singola classe, è possibile procedere seguendo l'approccio *one-vs-all*

# Valutazione dei modelli di classificazione

## One-vs-all

- L'approccio one-vs-all consente di ricondurci ad un caso di classificazione binaria
- In fase di valutazione, a turno, ciascuna classe viene considerata come “positiva”, e tutte le altre come una singola classe “negativa”



# Valutazione dei modelli di classificazione

## Valutazione su dati separati rispetto al training

- Tramite l'opzione *Use training set*, possiamo dire a WEKA di svolgere l'addestramento del modello su tutti i dati di training, e di utilizzare gli stessi dati per la sua valutazione
- Ciò può generare delle stime ottimistiche dell'errore, in quanto le istanze utilizzate per l'addestramento vengono utilizzate anche per la valutazione
- Per ottenere stime più realistiche, l'opzione *Percentage split* consente di dividere il dataset assegnando in modo casuale una percentuale delle istanze al training, e le rimanenti ad un altro insieme, detto di test, utilizzato esclusivamente ai fini di valutazione del modello (non viene svolto su di esse alcun addestramento)

# Valutazione dei modelli di classificazione

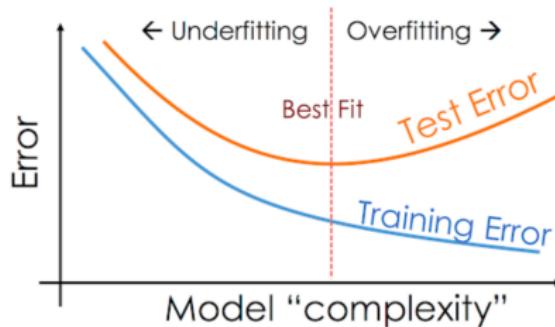
## Qualche altro esempio in WEKA

- Si carichi in WEKA il dataset *diabetes.arff*, si costruisca un albero di decisione e si confronti la matrice di confusione (e relativa accuratezza) che si ottiene impostando *Test options* a *Use training set*, ed impostandolo invece a *Percentage split*
  - L'obiettivo è prevedere, sulla base di misurazioni diagnostiche, se un paziente ha il diabete
- Si carichi in WEKA il dataset *credit-g.arff*, si costruisca un albero di decisione e si osservi la matrice di confusione che si ottiene impostando *Test options* a *Percentage split*. L'accuratezza è una buona metrica per misurare la bontà del modello in questo caso?
  - L'obiettivo è classificare una persona come a rischio o meno di credito a seconda di un insieme di caratteristiche

# Valutazione dei modelli di classificazione

## Complessità dei modelli e generalizzazione

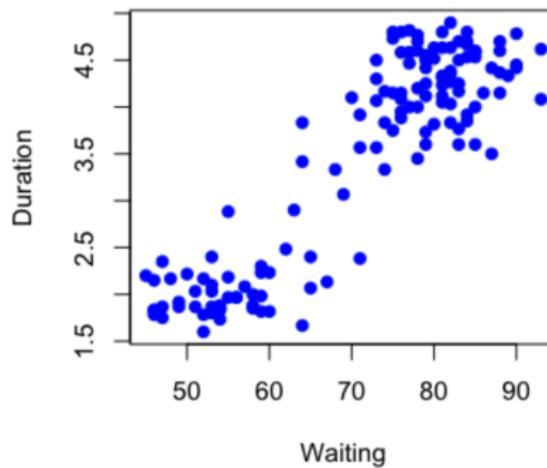
Modelli complessi (es., alberi grandi) tenderanno ad ottenere buone prestazioni sui dati di training, ma generalizzeranno meno su nuovi dati rispetto a modelli più semplici



Si carichi in WEKA il dataset *credit-g.arff*. Si generi un albero di classificazione valutandolo nelle due modalità *Use training set* e *Percentage split*. Si generi ora un altro albero impostando *unpruned = True*, e lo si valuti come il precedente. Cosa si può osservare?

# Task di regressione

- Nel caso di task di regressione, il valore dell'obiettivo  $L$  è un numero continuo
- Si vuole prevedere l'output numerico (*variabile dipendente*) sulla base di attributi numerici forniti in input (*variabili indipendenti*)



# Instance-based learning

- L'instance-based learning può essere facilmente adattato per svolgere regressione
- Invece di calcolare la moda dell'attributo obiettivo dei  $k$  elementi più simili all'istanza da prevedere, calcoliamo ad esempio la media
- Esattamente come nel caso della classificazione, non viene generato un modello
- Dunque, questa strategia può essere utilizzata per ottenere una predizione, ma non per estrarre della generica conoscenza inherente a proprietà o regolarità insite nei dati

# Regressione lineare

- Si vuole esprimere la variabile dipendente come combinazione lineare di una o più variabili indipendenti:

$$y = w_0 + w_1 a_1 + w_2 a_2 + \cdots + w_k a_k$$

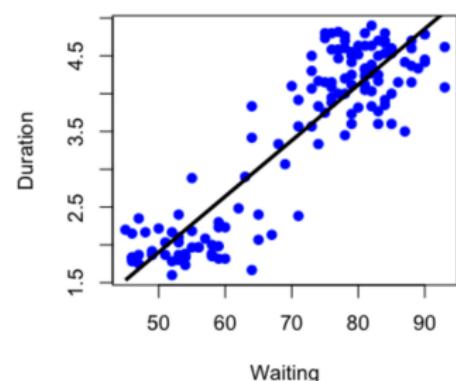
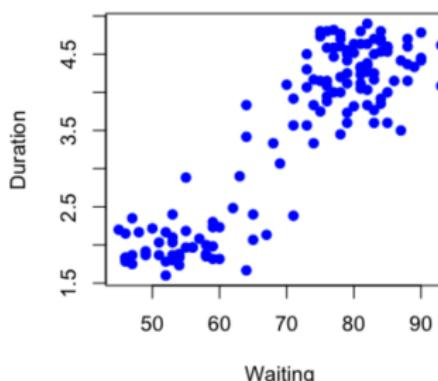
- Metodo semplice che genera modelli di intuitiva interpretazione, tuttavia ha delle limitazioni:
  - la relazione fra variabile dipendente ed indipendenti deve essere lineare
  - le variabili indipendenti non devono essere “correlate” fra loro (no *multicollinearità*)
  - altre condizioni di applicabilità:  
[https://en.wikipedia.org/wiki/Linear\\_regression](https://en.wikipedia.org/wiki/Linear_regression)

# Regressione lineare

## Esempio: Yellowstone geyser

Durata eruzione	Tempo attesa
2.883	55
1.883	54
2.167	52
1.600	52
1.750	47
1.967	55

$$\text{durata\_eruzione} = 0.073901 * \text{tempo\_attesa} - 1.792739$$



# Regressione lineare

## Esempio: Valore delle case

- Si carichi in WEKA il dataset *house.arff*, che riguarda lo stabilire il valore di vendita delle case (*sellingPrice*) sulla base di altri attributi predittori
- Si addestri un modello di regressione lineare per prevedere il valore di *sellingPrice*
  - Clic sulla lingetta *Classify*
  - Clic sul pulsante *Choose* nel campo *Classifier*
  - Clic su *functions > LinearRegression*
  - Impostare *Use training set* e premere *Start*
- Cosa possiamo osservare riguardo al peso dato dal modello all'attributo *houseSize*?

# Regressione lineare

## Esempio: Valore delle case (2)

- Il peso assegnato a *houseSize* è negativo
- Ciò è contro intuitivo, in quanto secondo il modello, all'aumentare delle dimensioni della casa, cala il suo valore
- In realtà, è un fenomeno dovuto alla presenza di multicollinearità fra *houseSize* e *bedrooms* (numero di camere)
- Si ripete l'addestramento del modello dopo aver rimosso l'attributo *houseSize*

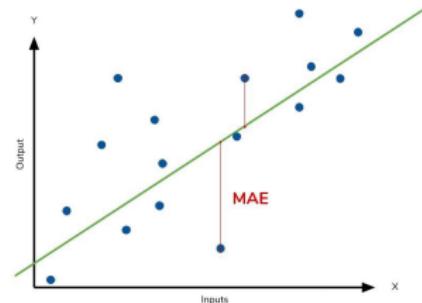
# Valutazione dei modelli di regressione

- Valutare un modello di regressione può essere più complesso, a livello intuitivo, che valutare un modello di classificazione
- Non possiamo calcolare accuratezza, precision, recall, ...
- Possiamo considerare tre principali metriche:
  - Mean Absolute Error (MAE)
  - Mean Square Error (MSE)
  - Root Mean Square Error (RMSE)

# Valutazione dei modelli di regressione

## Mean Absolute Error

$$MAE = \frac{1}{N} \sum_{i=1}^N |y_i - \hat{y}_i|$$



- MAE è calcolato come la somma dei valori assoluti degli errori di predizione, diviso il numero di punti considerati
- Fornisce un valore che consente di determinare quanto la predizione è distante dal valore esatto
- Segue la stessa scala dei dati
- Gli errori vengono pesati, proporzionalmente, allo stesso modo

$$MSE = \frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- MSE è calcolato come la somma dei quadrati degli errori di predizione, diviso per il numero di punti considerati
- Non segue la stessa scala dei dati, tuttavia consente comunque di confrontare fra loro le prestazioni di diversi modelli di regressione
- Attraverso l'elevamento al quadrato, questa metrica "esaspera" gli errori più grandi, facendoli pesare (anche proporzionalmente) di più rispetto agli errori più piccoli

# Valutazione dei modelli di regressione

## Root Mean Square Error

$$RMSE = \sqrt{\frac{1}{N} \sum_{i=1}^N (y_i - \hat{y}_i)^2}$$

- MSE è calcolato come la somma dei quadrati degli errori di predizione, diviso per il numero di punti considerati
- La radice quadrata riporta il valore sulla stessa scala delle predizioni, a beneficio della sua interpretabilità
- Anche qui, i punti affetti da errori grandi (es., outlier), vengono penalizzati proporzionalmente di più rispetto a quelli con errori piccoli

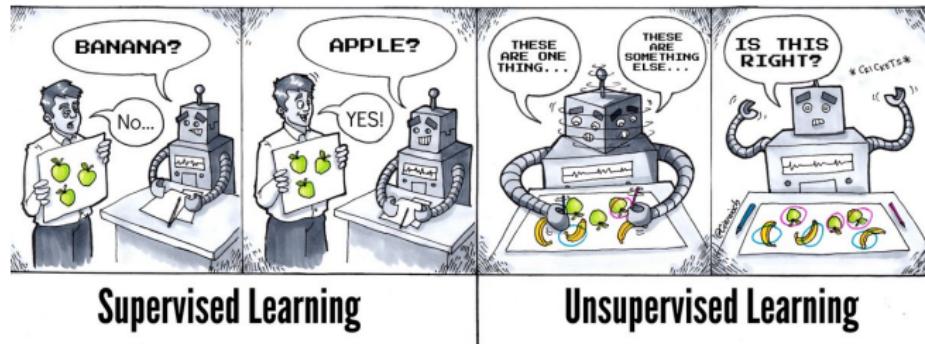
# Regressione lineare: Wine quality dataset

- Sono inclusi due set di dati (*winequality-white.arff* e *winequality-red.arff*), relativi a campioni di *vinho verde* rosso e bianco, dal nord del Portogallo
- L'obiettivo è modellare la qualità del vino sulla base di test fisico-chimici
- Si carichino i dati in WEKA e si valuti la presenza di correlazioni fra gli attributi (etichetta *Visualize*)
- Si lanci poi un analisi un'analisi di tipo regressivo utilizzando un modello del tipo *LinearRegression*
- I modelli forniscono delle buone prestazioni di predizione, suddividendo i dati in training e test?
- Quali sono i fattori che più influenzano la qualità del vino, e come? Cosa accade ai pesi, nel caso di *winequality-white.arff*?

# Unsupervised learning

# Concetti fondamentali

- Nell'unsupervised learning non vi è alcun attributo obiettivo
- Si vogliono ricercare delle generiche relazioni fra gli attributi (regole di associazione), o determinare la presenza di similarità fra le istanze (clustering)



# Regole di associazione

- Le regole di associazione consentono di estrarre generiche relazioni fra gli attributi, ad esempio, il fatto che il valore di un attributo determini il valore di un altro attributo
- Caso pratico: scoprire che le persone che acquistano un determinato prodotto ne acquistano anche un altro
  - ottimizzare la disposizione dei prodotti sugli scaffali
  - ideare sconti e promozioni mirati
  - sviluppo di recommender system
- Algoritmi classici: *Apriori*, *FP Growth*

# Regole di associazione

## Esempi

Temperatura = mite → Umidità = normale

(Umidità = bassa AND Vento = false) → Cielo = soleggiato

Umidità = alta → (Cielo = pioggia AND Vento = true)

La singola regola  $A \rightarrow B$  può essere valutata secondo:

- **Support:** 
$$\frac{\text{numero di istanze che contengono } A \text{ e } B}{\text{numero totale di istanze}}$$
  - frazione delle istanze in cui compaiono tutti gli elementi della regola
  - che percentuale dell'insieme di istanze riesce a "coprire" la mia regola?
  - utile per filtrare regole che appaiono poche volte nel dataset, e che potrebbero quindi essere poco rilevanti
  - in termini probabilistici:  $P(A, B)$
- **Confidence:** 
$$\frac{\text{numero di istanze che contengono } A \text{ e } B}{\text{numero totale di istanze che contengono } A}$$
  - frazione delle istanze soddisfacenti l'antecedente della regola, che soddisfano anche il conseguente
  - in che percentuale di casi la mia regola è vera?
  - in termini probabilistici:  $P(B|A)$

# Regole di associazione

## Dataset WEKA: Market basket analysis

- Si carichi in WEKA il dataset *marketbasket.arff*
- Il dataset riguarda singole transazioni avvenute in un supermercato; per ogni transazione viene tenuta traccia dei beni acquistati dai clienti; vogliamo estrarre regolarità nei comportamenti di acquisto
- Utilizziamo a tal fine l'algoritmo *FP Growth*
  - Clic sulla lingetta *Associate*
  - Clic sul pulsante *Choose* nel campo *Associator / FP Growth*
  - Clic sul campo dei parametri dell'algoritmo:
    - *lowerBoundMinSupport*: 0.01
    - *metricType*: Conviction
    - *minMetric*: 15
    - *numRulesToFind*: 500

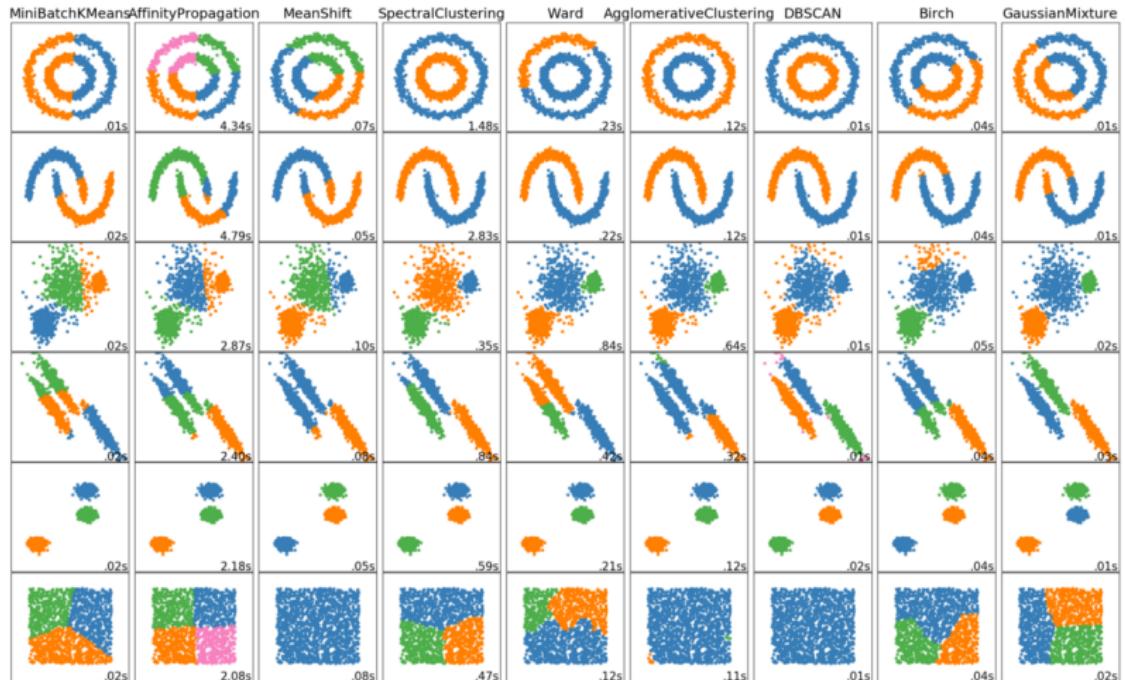
# Clustering

- L'idea è raggruppare fra loro istanze simili
- Task tipico: segmentazione del mercato
- I gruppi individuati possono essere:
  - *esclusivi* (hard clustering)
  - *con overlap* (fuzzy clustering)
  - *probabilistici* (fuzzy clustering)
  - *gerarchici*

# Clustering

## Tipologie di approcci

La scelta dell'approccio influenza largamente il risultato finale, e dipende anche dalla distribuzione delle istanze



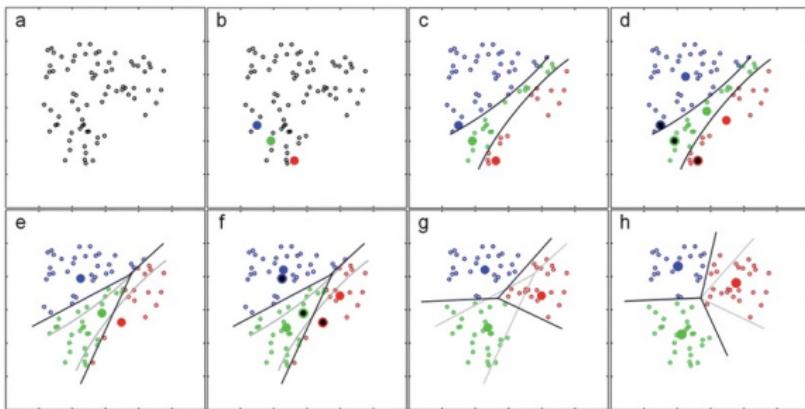
# L'algoritmo k-means

Uno dei più diffusi algoritmi per il clustering esclusivo:

- ① seleziona il numero di cluster da ricercare,  $k$
- ② scegli casualmente  $k$  istanze, e ponile come centri di altrettanti cluster
- ③ assegna tutte le istanze-non-centro al punto, fra i  $k$  centri, a loro più vicino
- ④ calcola, utilizzando le istanze di ciascun cluster, la media dei diversi attributi, e poni il punto così ottenuto come nuovo centro del cluster
- ⑤ se, alla luce dei nuovi centri, almeno un'istanza cambierebbe cluster di appartenenza, ripeti dal punto 3, altrimenti la situazione è stabile e l'algoritmo termina

# L'algoritmo k-means

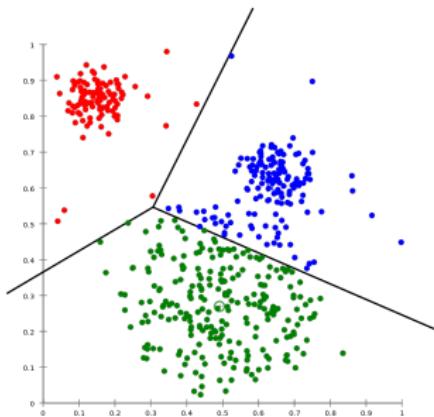
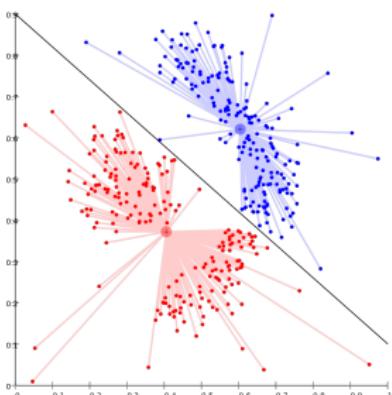
## Rappresentazione grafica



- l'algoritmo non garantisce l'ottimalità del raggruppamento;
- sono da tenere in considerazione aspetti riguardanti la *normalizzazione* degli attributi;
- è necessario impostare il valore di  $k$ , a differenza che in altre metodologie di clustering (es., gerarchico).

# L'algoritmo k-means

## Esempi di cattiva clusterizzazione



- *immagine a sinistra:* numero scorretto di cluster da ricercare
- *immagine a destra:* impossibilità di catturare la forma dei due cluster

# Clustering

## Valutazione del risultato

- Non vi sono delle metodologie univoche per valutare la bontà del risultato del clustering
- Molto dipende dalla conoscenza del dominio e dal significato che si riesce a dare ai vari raggruppamenti: che caratteristiche hanno in comune le istanze che sono state raggruppate?
- Esistono in ogni caso alcune metriche che cercano di valutare la forma dei cluster
- Ad esempio, la *Silhouette* attribuisce un punteggio migliore a situazioni in cui i cluster sono ben distanziati fra loro e compatti al loro interno

# Clustering

## Dataset WEKA: Mc-Menù

- Il dataset *mc-menu.arff* contiene informazioni nutrizionali relative a cibi serviti al McDonald's
- Effettuiamo un clustering considerando le informazioni nutrizionali
- Per prima cosa, ricordiamoci di normalizzare gli attributi
- Poi lanciamo l'algoritmo *k – means* cercando 5 cluster, ignorando le etichette dei cibi e la loro categoria
- Le istanze raggruppate in cluster hanno delle regolarità?
- Ad esempio, come si pongono relativamente alla loro categoria?