

Teoria Data Mining

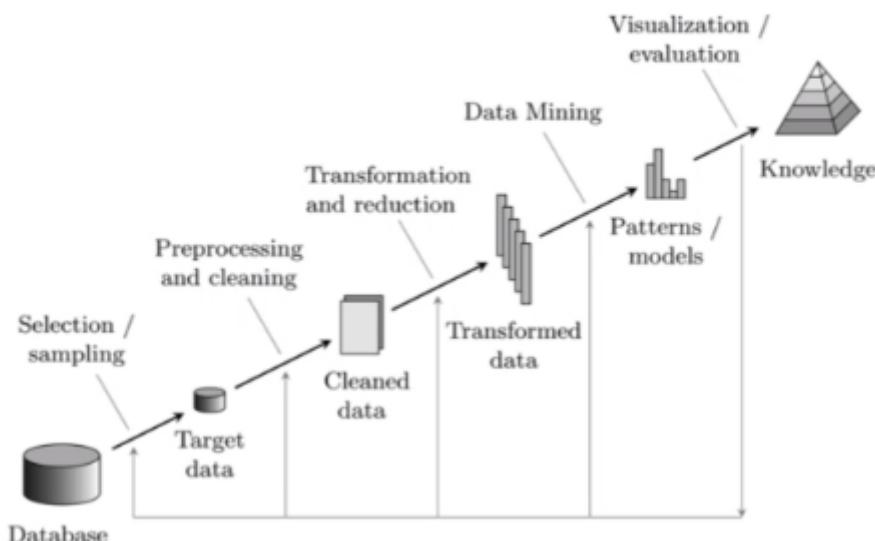
04/02/2023

Infos

- Docente: Francesco Marcelloni
- Mail: francesco.marcelloni@ing.unipi.it
- Useremo dei data set aventi righe con valori
- L'esame di maggio consisterà in data set, un paio, di cui bisogna applicare alcune tecniche di data mining e raccontare il risultato ottenuto

Data mining e machine learning

The Data Mining Process



Si tratta della pratica per ricercare in grandi moli di dati dei pattern utili, ovvero conoscenza. Se però individuiamo la traduzione effettiva dall'inglese, *mining* si intende l'estrazione mineraria, quindi significa *estrazione di dati*. Ma questa traduzione non corrisponde alla definizione. Noi i dati li abbiamo già quindi ci interessa estrarre dei pattern o conoscenza dai dati che possono essere in grande mole e che possono essere descritti da un numero elevato di caratteristiche e in modo diverso.

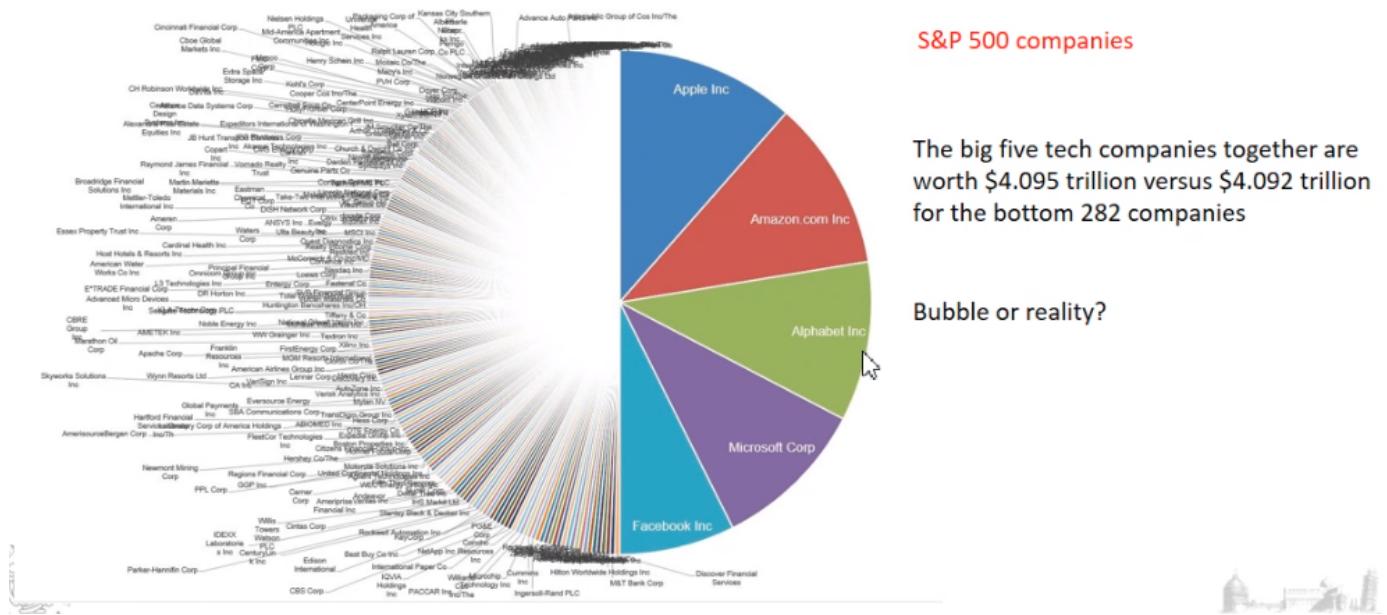
Questo termine quindi dal punto di vista inglese non corrisponde con quello che tutti intendono per data mining. Nomi alternativi come *Knowledge discovery* sarebbero più indicati, ma oramai Data mining è entrato nel linguaggio comune. Il data mining è davvero rilevante oggi perché avendo tanti dati diventa difficile analizzarli.

Il machine learning (ML) è invece la capacità di un computer di apprendere dall'esperienza, insegnando al computer a risolvere dei problemi usando dati precedenti o passati. Spesso in data mining si usano

delle tecniche del ML (classification, clustering, regression, etc...).



Le cinque più grandi compagnie di High-tech in questa figura mostrano che esse hanno un valore da un punto di vista economico confrontabile con altre 282 aziende:



Alphabet, Amazon e Facebook sono 3 aziende che si occupano dell'elaborazione e raccolta di dati. Da questo spaccato si capisce quanto il dato sia importante dal punto di vista economico.

Malware detection

Nell'ambito della cybersecurity, il data mining può essere utile per fare **MALWARE DETECTION** ad esempio. La necessità è quella di individuare i malware prima che questi possano recare dei danni alle istituzioni o alle aziende colpite. Quelle di seguito sono alcune delle tecniche usate per l'identificazione.

- **Anomaly detection:** Queste tecniche non prevedono un addestramento, ma descrivono il comportamento normale. Più in generale va sotto il nome di outlier detection quindi di individuazione di comportamenti che si differenziano da quelli normali. Quindi modellando il comportamento normale, possiamo riconoscere attacchi di vario tipo in quanto essi si differenziano dai comportamenti leciti.

- **Misuse detection:** Altri tipi di tecniche, come questa invece, usano un dataset di addestramento, in cui qualcuno ci indica quali di quei dati sono associati a possibili attacchi. Utilizziamo dei dati che sono etichettati, quindi qualcuno ci ha detto che corrispondono a specifici attacchi per addestrare il sistema a riconoscere quando siamo sotto attacco. In questo caso abbiamo addestrato il sistema a riconoscere solo gli attacchi forniti in input, nuovi attacchi potrebbero non essere riconosciuti.
 - **Hybrid detection:** gli approcci ibridi sono interessanti in quanto riescono a sfruttare le peculiarità di entrambi gli approcci in termini di accuratezza
-

Data sets

Tipicamente quello che un ingegnere si trova a fare in questo settore è ragionare su algoritmi disponibili in librerie, provare a risolvere il problema e poi confrontare i risultati ottenuti e trovare l'algoritmo migliore.

Il punto di partenza però prima di parlare di algoritmi sono i **dati**. Gli oggetti, dati, istanze o campioni sono sinonimi per descrivere un entità, ad esempio per un database di vendite, un oggetto può essere un prodotto da vendere o un cliente. Se parliamo di cybersecurity, un oggetto può essere un traffico in rete. Gli oggetti sono descritti da **attributi**, questi possono essere di vario tipo e servono a definire l'oggetto stesso. Se volessimo fare un paragone con la memorizzazione all'interno della base di dati, potremmo dire che le righe corrispondono agli oggetti e le colonne agli attributi:

Attributo 1	Attributo 2
valore	valore
valore	valore

Gli attributi sono delle caratteristiche che servono a rappresentare l'oggetto stesso.

Esempio: per un cliente gli attributi possono essere l'ID, il nome, la partita IVA, etc...

Gli attributi sono contraddistinti dai loro tipi. Noi vedremo il tipo nominale, binario e numerico.

■ Types:

- **Nominal**
- **Binary**
- **Numeric: quantitative**
 - **Interval-scaled**
 - **Ratio-scaled**

Cercheremo anche di capire la somiglianza tra oggetti sulla base di attributi, perché gli algoritmi useranno qualche calcolo di somiglianza tra oggetti e dobbiamo capire come esprimere la somiglianza quando gli oggetti hanno attributi di tipo differente.

Il tipo:

- **nominale**: si contraddistingue dal fatto che i possibili valori sono delle etichette, dei nomi e non c'è un ordinamento. Es. i capelli possono essere grigi, gialli, rossi. Non avendo un ordinamento non posso neanche dire che rosso sia più vicino a biondo o grigio. Questo è un problema perché quando dobbiamo definire la distanza tra 2 oggetti dobbiamo sapere che non c'è un ordinamento. Calcoliamo la distanza tra due oggetti nominali:

$$d(i, j) = \frac{p - m}{p}$$

m: # of matches, p: total # of variables

m: è il numero di valori uguali nei due oggetti per attributi corrispondenti (matches), **p**: è il numero totale di variabili nominali che descrivono l'oggetto

La distanza viene calcolata andando a vedere quanti sono gli attributi nominali che descrivono i due oggetti che hanno lo stesso valore. Questa è l'unica cosa che possiamo fare per calcolare la distanza tra oggetti con valori nominali, non essendoci un ordinamento. L'unica cosa valutabile è vedere se quell'attributo ha lo stesso valore in entrambi gli oggetti. Supponendo di avere **p** attributi, se **p** attributi hanno gli stessi valori tra i 2 oggetti, la distanza tra i 2 oggetti è nulla. Se nessun attributo è uguale, la distanza tra i due oggetti è massima ovvero 1.

- **binario**

Un tipo di attributo nominale molto particolare è l'attributo binario in quanto come gli attributi nominali ha 2 possibili valori, **true/false**, **0/1** è particolare perché ha solo 2 valori. Possiamo avere due tipi di attributo binario:

- **simmetrico**: i due possibili valori sono ugualmente importanti o in altri termini si presentano con la stessa probabilità ad esempio il genere
- **asimmetrico**: i risultati non sono ugualmente importanti, uno dei due valori si presenta con un alta probabilità rispetto all'altro. Nei test medici, ad esempio, la positività è più bassa rispetto alla negatività

Per convenzione si assegna 1 al risultato più importante che però non è detto che sia quello più probabile. Nel caso del test medico "Positività all'HIV" si assegna 1 al valore di positività al test che

però per fortuna non è quello più probabile.

■ Binary

- Nominal attribute with only 2 states (0 and 1)
- Symmetric binary: both outcomes equally important
 - e.g., gender
- Asymmetric binary: outcomes not equally important.
 - e.g., medical test (positive vs. negative)
 - Convention: assign 1 to most important outcome (e.g., HIV positive)

		Object <i>j</i>			
		1	0	sum	
Object <i>i</i>	1	<i>q</i>	<i>r</i>	<i>q+r</i>	$d(i, j) = \frac{r+s}{q+r+s+t}$
	0	<i>s</i>	<i>t</i>	<i>s+t</i>	
sum		<i>q+s</i>	<i>r+t</i>	<i>p</i>	

È importante fare una distinzione tra simmetrico e asimmetrico, e ne vediamo il perchè nel caso in cui ad esempio dobbiamo calcolare la distanza tra due oggetti i e j definiti attraverso attributi binari. Per capire come è calcolata la distanza bisogna fare riferimento alla tabella: cosa rappresenta ogni cella per l'oggetto i e l'oggetto j ?

Ad esempio la prima cella in alto a sinistra rappresenta quanti attributi binari hanno valore 1 per l'oggetto i e lo stesso valore 1 per l'oggetto j . Quindi q rappresenta il numero di attributi binari con valore 1 in entrambi gli oggetti. s sono gli attributi in cui l'oggetto i ha valore 1 e l'oggetto j ha valore 0. t attributi in cui entrambi hanno valore 0.

La distanza potremmo calcolarla come somma tra r ed s al numeratore e come somma tra q , r , s e t al denominatore. Al denominatore abbiamo quindi il numero totale di attributi rappresentato da p , al numeratore abbiamo il numero di attributi per cui il valore per i e per j sono differenti. Infatti r corrisponde al numero di attributi in cui i ha valore 1 e j ha valore 0 mentre s è il viceversa. $r+s$ è il numero di attributi in cui i e j hanno valori differenti. Se $r+s$ coincide con il numero totale di attributi p , abbiamo che la distanza tra i e j è massima ovvero uguale a 1. Se $r+s$ sono uguali a 0 significa che i e j vanno a coincidere.

Dove vediamo questa differenza tra attributi binari simmetrici e asimmetrici? Nella formula la differenza non c'è.

Cosa succede se gli attributi sono binari asimmetrici?

		Object <i>j</i>			
		1	0	sum	
Object <i>i</i>	1	<i>q</i>	<i>r</i>	<i>q+r</i>	$d(i, j) = \frac{r+s}{q+r+s+t}$
	0	<i>s</i>	<i>t</i>	<i>s+t</i>	
sum		<i>q+s</i>	<i>r+t</i>	<i>p</i>	Asymmetric

nel caso di attributo binario asimmetrico, abbiamo che la probabilità di avere un numero elevato di t è alta. Perchè nel caso di attributo binario asimmetrico abbiamo detto che il valore che per convenzione assegniamo con 1 è quello più importante ma con meno probabilità (esempio di positività all'HIV), quindi 1 ha una probabilità bassa di apparire, 0 alta. Dato che 0 ha una probabilità di apparire maggiore, ci aspettiamo sia per i che per j che il valore 0 appaia molte più

volte, quindi t sia elevato. Quindi la distanza apparirà sempre molto bassa. Questo è il motivo per cui con attributi binari asimmetrici non consideriamo t nella formulazione della distanza, questo ci consente di avere una distanza elevata quando ci sta una differenza effettiva tra i e j non prendendo in considerazione il numero t che è tipicamente elevato con attributi asimmetrici binari.

- **ordinali:** da un punto di vista dell'attributo i possibili valori sono delle etichette ma la differenza sta nel fatto che si tratta di valori ordinabili, ovvero l'ordine è significativo. Es: l'attributo size con valori small, medium, large. Nella nostra mente sappiamo che piccolo è più vicino a medio di quanto non lo sia rispetto a grande. Quando calcoliamo la distanza tra 2 oggetti che sono definiti attraverso attributi ordinali dobbiamo considerare l'ordinamento.

- **Ordinal**

- Values have a meaningful order (ranking) but magnitude between successive values is not known.
- Size = {small, medium, large}, grades, army rankings

- Order is important, e.g., rank $r_{if} \in \{1, \dots, M_f\}$
- Can be treated like interval-scaled
- Replace x_{if} by their rank
- Map the range of each variable onto [0, 1] by replacing i -th object in the f -th variable by

$$z_{if} = \frac{r_{if} - 1}{M_f - 1}$$

compute the dissimilarity using methods for interval-scaled variables

Come teniamo conto di questa intuizione (ovvero che small sia più vicino a medium che a large) relativamente all'ordinamento?

Andando a trasformare i valori testuali in valori numerici e trasformando il calcolo della distanza tra valori che sono etichette in calcolo della distanza in cui i valori sono numerici. Associamo ad ogni possibile valore ordinale un numero crescente, partendo da quello più basso. Nel caso della figura, associamo 1 a small, 2 a medium e 3 a large. Questo ci realizza la intuizione di prima, sappiamo che small è più vicino a medium rispetto a large, noi diamo questa idea perchè 1 è più vicino a 2 che a 3. Possiamo far variare il valore di size tra 0 ed 1 invece che tra 1 e 3. Basta sottrarre 1 al valore associato rispettivamente a small, medium e large e dividendo per il massimo valore numerico - 1.

Trasformiamo small ad esempio:

$$\frac{(1 - 1)}{(3 - 1)} = 0$$

A cosa serve normalizzare!?

Si normalizza per far variare i valori nello stesso range e far pesare i valori all'interno della distanza euclidea nello stesso modo.

Abbiamo mantenuto l'ordinamento ma trasformando le etichette in numeri. Il vantaggio è che l'attributo ordinale nominale è un attributo numerico. Questo ci porta a dire che se noi avessimo degli oggetti descritti da attributi ordinali, dopo la trasformazione abbiamo attributi numerici, e possiamo calcolarne la distanza, ad esempio la distanza euclidea.

- **quantità:**

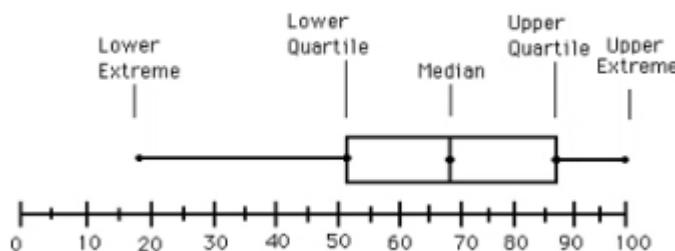
- **intervalli:** nel primo caso abbiamo gli attributi che sono misurati su una scala di unità con lo stesso intervallo ma che hanno un'anomalia non avendo uno 0. Le temperature in Celsius considerano delle unità che hanno la stessa dimensione, quello che non abbiamo è che lo 0°C non è un vero 0 perché sappiamo che non esiste un vero 0°C , fatto sta che una divisione tra temperature espresse in gradi Celsius non ha significato. La temperatura in gradi Kelvin ha uno 0 e ha senso fare un rapporto tra temperature. I gradi Kelvin da questo punto di vista sono dei razionali.
- **razionali:** lunghezze, quantità monetarie

Le cose si complicano se un oggetto può essere descritto da diversi tipi di attributi. Per calcolare la distanza tra 2 oggetti è calcolare la distanza per ciascun attributo, e poi fare una media pesata con il peso che corrisponde al numero di attributi di quel tipo.

I dati li vedremo come singolo oggetto con un certo numero di attributi. Quando approcciamo un problema di data mining è cercare di capire meglio come sono distribuiti i dati, per capire cosa possiamo aspettarci. Per visualizzare un dato non è facile, perché se un dato è rappresentato da tante dimensioni non possiamo darne una rappresentazione visuale, a meno di non visualizzare come i valori sono distribuiti attributo per attributo. Per far questo abbiamo il Box Plot.

Box plot

Consente di capire come i valori sono distribuiti per uno specifico attributo. Si tratta di una rappresentazione grafica basata su una scatolina, quella raffigurata in basso:



che rappresenta **5 valori statistici**:

- minimo
- massimo
- mediana
- primo quartile
- terzo quartile

Il q_1 e q_3 sono il primo e il terzo quartile. Il quartile è il valore che individua il 25% dei valori con valore inferiore a quello del quartile. Ci sono il 25% di punti dell'attributo che stiamo considerando che cadono al di sotto del valore 51 circa, se prendiamo come esempio quella della figura. Il terzo quartile

corrisponde a circa 86 significa che ci sono il 75% dei punti al di sotto di quel valore. La mediana corrisponde a quel valore per cui il 50% degli attributi ha valori sotto il valore della mediana e il 50% degli attributi ha valore sopra il valore della mediana.

Presi tutti i valori corrispondenti all'attributo, ordinati, noi andiamo dal valore minimo al massimo. Questo ci trasmette un'idea molto grossolana di come sono distribuiti i valori all'interno dell'attributo. Il 50% dei valori tra i due quartili, sono tra 51 e 86. Poi abbiamo un 25% dei valori che varia tra 51 e 19 e un 25% tra 86 e 100.

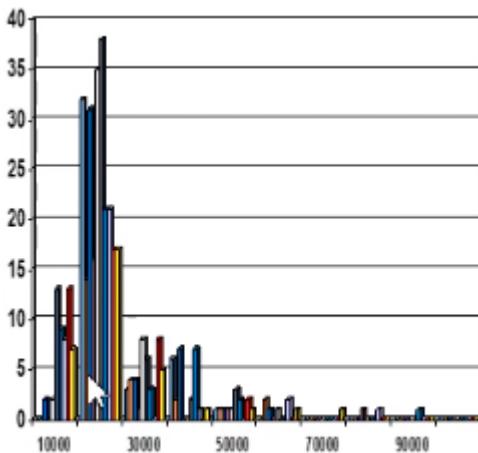


I box plot qui sono rappresentati in modo diverso. Ci sono gli * che sono gli outliers, ovvero valori molto differenti rispetto ai valori tipici del dataset. Sono quei punti che hanno un valore più alto o più basso di $1.5 * \text{IQR}$, dove IQR è la differenza tra il quartile q_3 e il quartile q_1 . Questo blox pot ci consente di avere un'idea della distribuzione dei dati ma anche confrontare le differenti distribuzioni. Il secondo box plot ha una distribuzione dei punti molto concentrata mentre il terzo ha una distribuzione molto sparsa. Questa idea mi è data utilizzando solamente 5 valori statistici. In generale si usa per confrontare la distribuzione relativa allo stesso attributo. Se si ha lo stesso range di variazione, si può usare per confrontare anche attributi diversi.

Istogramma

Un'altra rappresentazione grafica è l'istogramma ma spesso c'è una confusione con il bar chart. Il valore nell'istogramma è dato non dal valore dell'ordinata ma dall'area della barra. Esso rappresenta per ogni intervallo che consideriamo per un attributo, quanti valori abbiamo in quell'intervallo. Tipicamente usiamo l'ordinata della barra corrispondente, ma per definizione, nell'istogramma, cosa ci indica il numero di valori che abbiamo in quell'intervallo, quindi la frequenza, è l'area della barra NON l'ordinata. Tipicamente usando intervalli della stessa dimensione, è ovvio che considerare l'area o l'ordinata è analogo. Ma dal punto di vista formale, la frequenza del valore in quell'intervallo è associato

all'area.

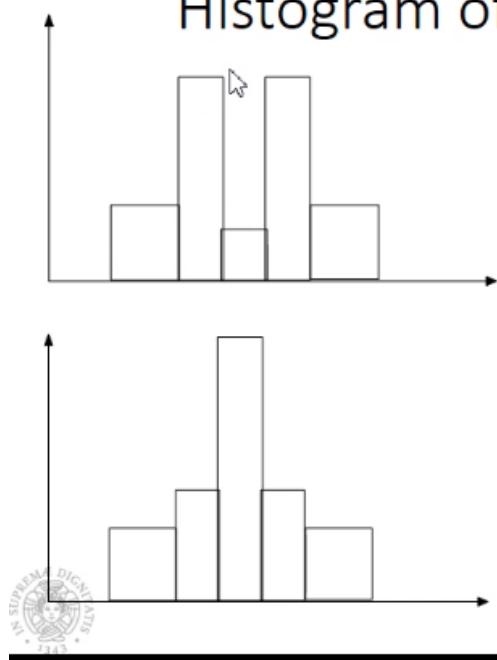


Quindi divide il dominio dell'attributo in intervalli (in questo caso della stessa dimensione) e conta i valori nei singoli intervalli.

❓ Tra le 2 rappresentazioni, quale è più espressivo? L'istogramma

Esempio: con una distribuzione bimodale (in alto, ci sono due picchi) o unimodale (in basso)

Preliminary Analysis Histogram often tell more than Boxplots



- The two histograms shown in the left may have the same boxplot representation
 - The same values for: min, Q1, median, Q3, max
- But they have rather different data distributions

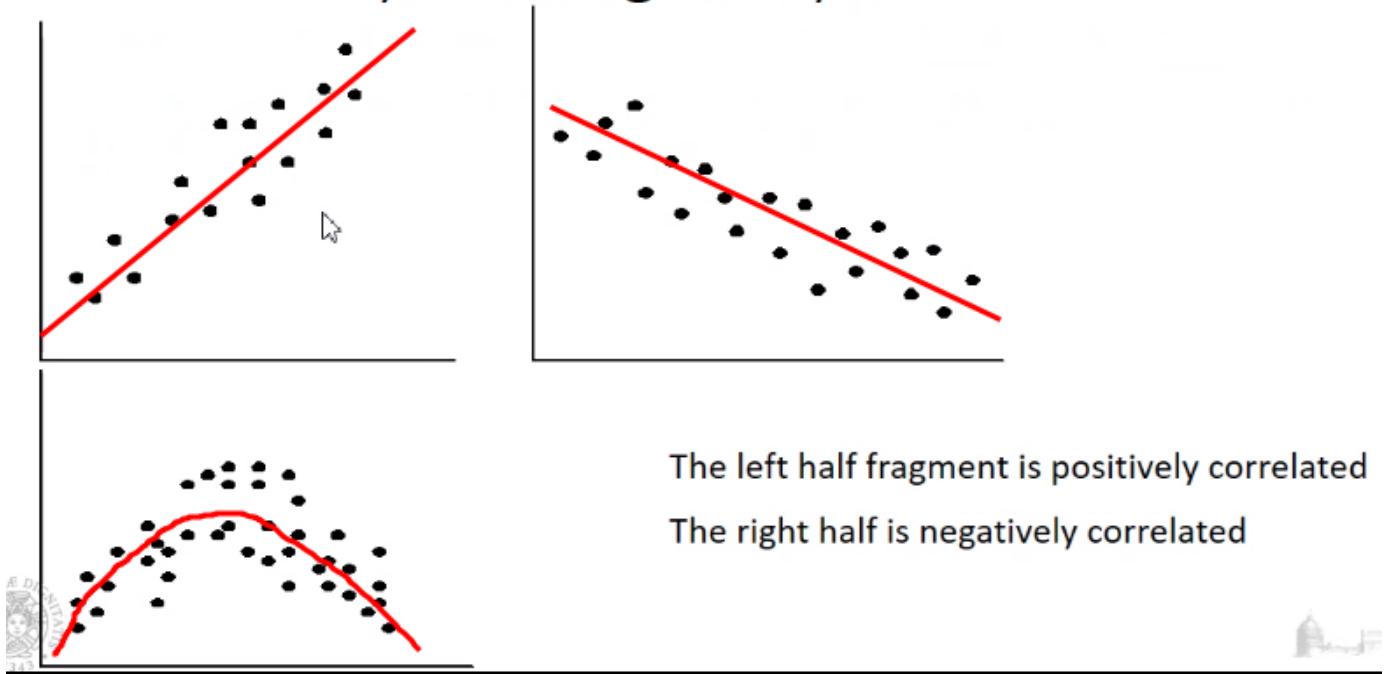
questi corrispondono allo stesso boxplot, che rappresentando 5 statistiche è fatto solo su 5 numeri, quindi poco accurato. Si può aumentare la espressività dell'istogramma andando a ridurre la ampiezza dell'intervalle, ovviamente riducendolo troppo torniamo ad avere un dataset.

Scatter Plot

Rappresentazione diversa dalle precedenti perchè consente di confrontare 2 attributi. Usiamo un attributo sulle ascisse e uno sulle ordinate e rappresentiamo i punti nel piano che si forma. Può essere utile perchè consente di capire in modo semplice se esiste una relazione tra i 2.

Preliminary Analysis

Positively and Negatively Correlated Data



Nel primo scatter plot all'aumentare del valore della x aumenta anche la y , lo scatter plot è molto concentrato all'interno di una linea. Nel secondo otteniamo invece un fenomeno inverso. Questi 2 scatter plot ci fanno capire che esiste una **correlazione** tra i 2 attributi, all'aumentare del valore di un attributo aumenta anche l'altro o vicensa, quindi i 2 attributi hanno una quantità informativa simile, ovvero il **trend** è lo stesso. **Se hanno uno stesso trend, due attributi portano uno stesso contenuto.** Potremmo pensare di non usarli entrambi dal punto di vista di tecniche di data mining, quindi possiamo decidere di usarne uno solo.

Nel caso dello scatter plot in basso invece è ovviamente differente in quanto non si muove lunga una linea, vi è una correlazione positiva fino a un certo valore della x e poi da quel valore in poi la correlazione diventa negativa. Questa analisi di correlazione attraverso scatter plot può essere utile per farci capire che ci sono coppie di attributi ridondanti.

La produzione di scatter plot può avvenire solo per attributi di tipo numerico.

Data visualization

Quando i dati sono espressi in modo testuale, essi sono trasformati con delle tecniche in numeri.

Preliminary Analysis

Data Visualization



Example of Tag Cloud

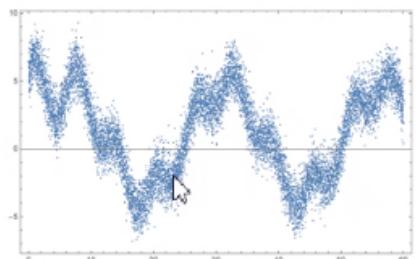
Questa visualizzazione permette di capire quali sono le parole più frequentemente usati nel documento stesso. Viene associata una dimensione del font più grande con le parole più frequenti.

Data cleaning

I dati grezzi dai quali si parte, possono essere estratti da database, da sensori IoT, che per loro natura sono sporchi, incorretti, a causa di errori umani o del computer o di trasmissione. Quando parliamo di rumore, lo si associa a dati prelevati da sensori, oltre alla misura del parametro che stiamo raccogliendo, ad esempio temperatura o umidità, abbiamo anche una sorta di rumore dato dal dispositivo che stiamo analizzando. Questo rumore va eliminato in quanto ci dà un'oscillazione di valori che non rappresenta la grandezza che stiamo acquisendo. Ma esiste un rumore anche con dati estratti da database. Rumore dovuto al fatto che per alcuni record i dati sono incompleti, o inseriti male. In alcune situazioni il rumore è creato intenzionalmente, ad esempio se la data di nascita non è inserita viene dato il valore di default 01/01/1901.

Noisy Data

- **Noise:** random error or variance in a measured variable
- Incorrect attribute values may be due to
 - faulty data collection instruments
 - data entry problems
 - data transmission problems
 - technology limitation
 - inconsistency in naming convention
- Other data problems which require data cleaning
 - duplicate records
 - incomplete data
 - inconsistent data



Incomplete Data

Un oggetto incompleto significa che manca qualche valore per qualche attributo.

Esempio:

Supponiamo di voler calcolare la distanza tra 2 oggetti espressi con attributi numerici e supponiamo che per uno di questi non ci sia un valore. Quando calcoliamo la distanza euclidea, da un punto di vista algoritmico non viene valutato se si tratta di un valore inserito li casualmente o meno. Quindi applichiamo degli algoritmi per cui consideriamo che tutti i valori siano validi e presenti, ma potrebbe esserci in memoria in quella cella un valore casuale non compilato intenzionalmente. Se un valore manca dobbiamo fare in modo di eliminare il problema.

- Se ho un oggetto, una tupla, senza un valore, elimino l'oggetto, non lo prendo in considerazione. È ragionevole se gli oggetti con dati mancanti sono in percentuale infinitesima rispetto al numero totale di oggetti.
- Un approccio alternativo è quello di inserirli manualmente. Funziona solo se i dati con valori mancanti sono pochi.
- Un modo potrebbe essere di inserire dove ho un dato mancante una label, ad esempio unknown, che mi segnala che li c'è un valore mancante, dal punto di vista dell'elaborazione non me ne faccio nulla ma almeno lo segnalo esplicitamente.
- La soluzione è prendere il valore mancante, calcolare la media dei valori e sostituire il valore mancante con l'attributo.
- Oppure calcolare la media solo per gli oggetti appartenenti alla stessa classe e sostituire il valore mancante con questa media fatta sugli oggetti della stessa classe.
- L'ultimo approccio è sviluppare un sistema che prendendo dei valori vicini al valore mancante, riescono a generare il valore mancante. Questo approccio ovviamente ha i suoi aspetti negativi, in ogni caso quel sistema va generato e non siamo sicuri che i valori generati siano vicini al valore che avrebbe dovuto avere quell'oggetto.

Rimozione del rumore

Viene chiamata **smoothing**, ovvero l'idea di lasciare qualcosa. Prende questo segnale che è rumoroso e cerca di lasciarlo portando alla luce solo il segnale e rimuovendo il rumore. Uno degli algoritmi di smoothing è quello rettangolare.

Si prende il campione e si considera una **finestra** attorno al campione. Ad esempio con una finestra di 3 campioni, compreso il campione che stiamo considerando. *Lo smoothing si applica andando a sostituire il campione al centro con la media dei 3 campioni.* Sostituendolo con la media ho l'effetto di lasciamento perchè considero la media, quindi considero nella finestra più campioni e quando ne ho uno molto diverso dagli altri, sostituendolo con la media, questo campione mi torna approssimativamente ad essere simile agli altri:

Smoothing Algorithms

Rectangular or unweighted sliding-average smooth

- The simplest smoothing algorithm
- It simply replaces each point in the signal with the average of m adjacent points, where m is a positive integer called **the smooth width**. For example, for a 3-point smooth (m = 3):

$$S_j = \frac{Y_{j-1} + Y_j + Y_{j+1}}{3}$$

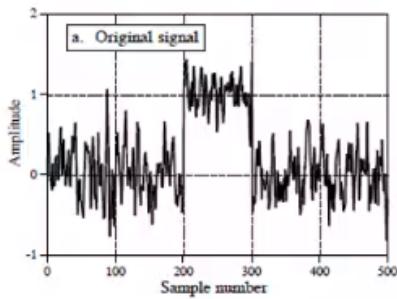
for $j = 2$ to $n-1$, where S_j the j^{th} point in the smoothed signal, Y_j the j^{th} point in the original signal, and n is the total number of points in the signal.

La formula considera una finestra con 3 campioni, sostituisco il campione al centro con indice j con la media dei campioni precedenti e successivo.

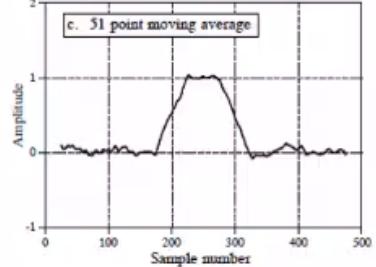
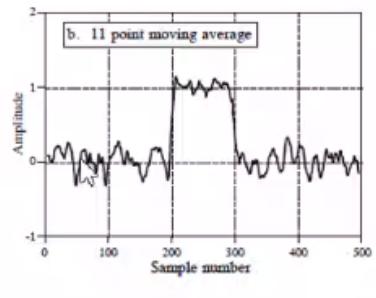
La dimensione della finestra accentua o meno il fenomeno di lasciamento del campione:

Smoothing Algorithms

Rectangular or unweighted sliding-average smooth



Filtered with 11 and 51 point moving average filters



Nella finestra con 51 punti il rumore si è ridotto ma la variazione non è più brusca ma graduale e deriva dall'applicazione del filtro, applicato con una finestra molto ampia, ovvero rimuove rumore ma sposta anche i momenti in cui avviene la transizione. Quando applichiamo un filtro di smoothing, con una finestra maggiore riduciamo maggiormente il rumore ma la variazione brusca viene graduata e ritardata.

10/02/2023

Avevamo introdotto il filtro rettangolare. Questo filtro funziona molto bene perché riesce a ridurre il ripple intorno al segnale vero e proprio. Nell'ultima lezione abbiamo visto due esempi, uno con 11 campioni, e uno con 51 campioni. Nel caso in cui usiamo un filtro con un numero di campioni alto, si notava una modifica del segnale in quanto anche questa transizione brusca dal livello basso al livello alto viene ammorbidente. Questo ci porta a dire che più facciamo ampia la finestra maggiore è l'effetto di smoothing ma più rendiamo questo effetto forte più possiamo alterare il segnale. Il valore massimo che viene raggiunto con l'ampiezza del segnale continua ad essere al valore 1, quindi il valore massimo non è stato toccato. È stata modificata solo la brusca variazione del segnale. Dobbiamo fare delle valutazioni e capire cosa ci serve del segnale. Se ci serve capire l'istante esatto in cui avviene la variazione, probabilmente il filtro con 51 campioni non è la scelta giusta. Se ci interessa solo pulire il segnale e conoscere la sua ampiezza massima può andare bene.

Filtro Savitzky-Golay (Filtro SG)

Smoothing Algorithms

The Savitzky-Golay Smoothing Filters

- In general, the simplest type of digital filter (the nonrecursive or finite impulse response filter) replaces each data value Y_j by a linear combination S_i of itself and some number of nearby neighbors

$$S_j = \sum_{n=-n_L}^{n_R} c_n Y_{j+n}$$

where n_L is the number of points used "to the left" of a data point i , i.e., earlier than it, while n_R is the number used to the right, i.e., later.



Si tratta sempre di un filtro di smoothing ma è più generale di quello rettangolare.

Se supponiamo di avere una finestra di 3 campioni, calcolando la media approssimiamo con una retta parallela all'asse x e riportiamo i punti su questa linea. Con il filtro di SG la capacità di modellazione è maggiore perchè viene usato un polinomio e possiamo stabilire il grado del polinomio da usare. Questo ci dà una capacità di modellazione più ampia, maggiore è il grado del polinomio maggiore è il grado di approssimazione. **Un grado troppo elevato rischia di riprodurre il rumore e non eliminarlo.** La cosa interessante è che ci offre un'opportunità di fare riduzione del rumore scegliendo però la funzione approssimante. Senza entrare troppo nella matematica viene definita la funzione approssimante e poi per trovare qual'è effettivamente la funzione che approssima meglio i punti, bisognerebbe gestire un approccio con i minimi quadrati. Ogni volta che si applica il filtro bisognerebbe però avere un approccio ai minimi quadrati. Dovendo processare il segnale in modo molto rapido, in molte applicazioni non è possibile perchè richiede del tempo. L'apporto di SG è che non ci sta bisogno di calcolare questi coefficienti c_n approssimanti utilizzando i minimi quadrati ogni volta che vogliamo rimpiazzare i campioni, ma semplicemente stabilendo il grado del polinomio e la dimensione della finestra, possiamo trovare questi valori c_n in tabella, quindi un'operazione più semplice. Da un lato, maggiore potenzialità di modellazione potendo usare delle funzioni approssimanti che modellano i campioni all'interno della finestra e dall'altro non dobbiamo fare particolari operazioni durante le operazioni di filtraggio.

Un esempio di tabella per l'applicazione del filtraggio di SG:

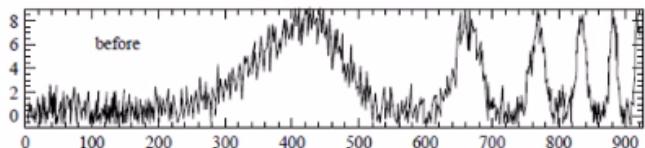
M	nL	nR	Sample Savitzky-Golay Coefficients										
2	2	2		-0.086	0.343	0.486	0.343	-0.086					
2	3	1		-0.143	0.171	0.343	0.371	0.257					
2	4	0		0.086	-0.143	-0.086	0.257	0.886					
2	5	5	-0.084	0.021	0.103	0.161	0.196	0.207	0.196	0.161	0.103	0.021	-0.084
4	4	4		0.035	-0.128	0.070	0.315	0.417	0.315	0.070	-0.128	0.035	
4	5	5	0.042	-0.105	-0.023	0.140	0.280	0.333	0.280	0.140	-0.023	-0.105	0.042

La prima colonna è il grado del polinomio, n_L e n_R sono il numero di campioni a destra e sinistra da voler considerare. I valori sono quelli da sostituire al posto di c_n . Ci impone quindi la scelta del grado e il numero di campioni a destra e sinistra. A quel punto avremo direttamente i coefficienti da moltiplicare per applicare il filtro. Questo filtro è molto interessante perché in riferimento agli esempi in basso, con il segnale originale in alto, abbiamo prima l'applicazione di un filtro con 16 campioni a destra e sinistra ed usa un polinomio di ordine 0, si tratta di un filtro rettangolare, sotto viene usato invece un polinomio di ordine 4 con lo stesso numero di campioni a destra e sinistra.

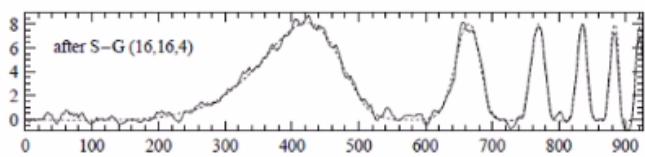
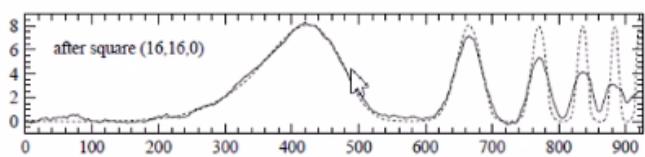
The Savitzky-Golay Smoothing Filters

- Some example

Sliding-average smooth with window of 33 points



Savitzky-Golay of degree 4 with window of 33 points



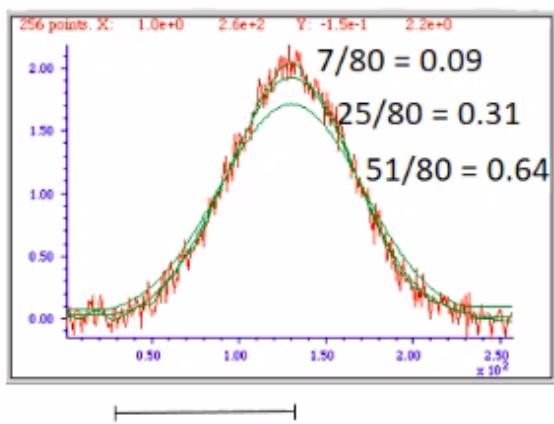
L'effetto di pulizia dal rumore nel primo caso è maggiore ma andando verso i picchi con larghezza molto piccola, il segnale viene perturbato molto in quanto il picco si abbassa molto. Avendo usato una finestra molto grande rispetto all'ampiezza del picco, l'effetto è che viene abbassato anche il picco stesso. Se fossimo interessati all'ampiezza massima dei picchi, questo filtro toglie informazione, quindi applicare il filtro rimuoverebbe il rumore anche l'informazione di cui abbiamo bisogno. Sotto invece le ampiezze dei picchi non sono stati modificati. La ragione è nel numero 4, ovvero stiamo usando una funzione approssimante che è un polinomio di ordine 4. Riusciamo a gestire i picchi perchè la funzione ha una capacità approssimante maggiore. Possiamo concludere che:

- più è grande l'ampiezza della finestra che stiamo usando maggiore è la riduzione del rumore
- però è anche vero che maggiore è la finestra, maggiore è la possibilità che il segnale venga distorto dall'operazione di smoothing
- l'approccio corretto prevede una conoscenza del segnale quindi sapere se ha dei picchi, quindi per poter avere un effetto che non distorca troppo il segnale stesso, dobbiamo considerare lo **smoothing ratio**, ovvero il rapporto tra la larghezza della finestra e il numero di punti nella metà del picco. Incrementando lo smoothing ratio, miglioriamo il rapporto segnale/rumore ma può

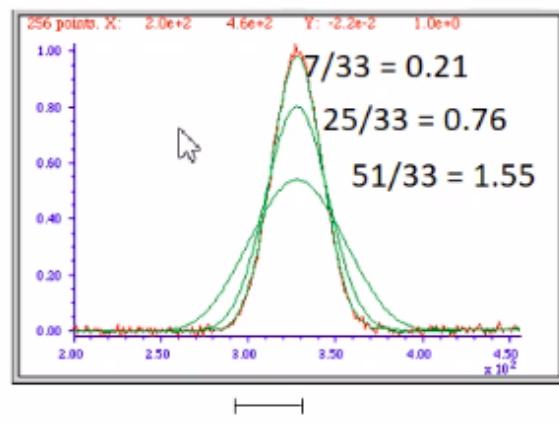
causare una riduzione della ampiezza e un aumento della banda del picco.

- The larger the smooth width, the greater the noise reduction, but also the greater the possibility that the signal will be distorted by the smoothing operation.
- The optimum choice of smooth width depends upon the width and shape of the signal and the digitization interval.
- For peak-type signals, the critical factor is the **smoothing ratio**, the ratio between the smooth width and the number of points in the half-width of the peak. In general, increasing the smoothing ratio improves the signal-to-noise ratio but causes a reduction in amplitude and an increase in the bandwidth of the peak.

Si vede la metà di ampiezza del picco, nella prima finestra costituita da 80 punti e nella seconda finestra da 33 punti. Se consideriamo uno smoothing ratio di 0.09 non abbiamo una modifica del segnale ma abbiamo anche che il rumore non è molto ridotto, più ci alziamo con lo smoothing ratio più abbiamo un abbassamento del picco e un allargamento del picco stesso.



half-width of
the peak 80 pts



half-width of
the peak 33 pts

Trovato il corretto bilanciamento, i filtri di smoothing sono utili, computazionalmente non sono pesanti, bisogna solo trovare lo smoothing ratio adatto per la nostra applicazione. Se l'obiettivo è di misurare l'altezza e la larghezza del picco, allora dovremo lavorare con degli smoothing ratio sotto il valore di 0.2. Se l'obiettivo è vedere solo la posizione del picco, possiamo anche permetterci dei valori di smoothing ratio maggiori a 0.2.

Quando lavorare con segnali filtrati?

Si cerca di ridurre il rumore anche per mostrare i segnali in modo più visibile, ad esempio quando comprendiamo che il rumore disturba gli algoritmi. Alcune volte non è il caso, esiste un caso molto eclatante in cui non usarlo. Parleremo della **outliner detection** ovvero l'individuazione di campioni con comportamento diverso dal comportamento normale. Considerando il segnale visto prima e vorremmo capire se ci sono outliers, ovvero campioni molto diversi dal segnale che stiamo acquisendo, applicando un filtro di smoothing, quel campione verrebbe riportato al pari degli altri. L'utilizzo di smoothing potrebbe non essere adatto in quel caso.

Data cleaning as a Process

Spesso ci troviamo a lavorare con dati memorizzati in basi di dati, ma ci sono comunque problemi, perchè i DB sono stati aggiornati negli anni e quindi qualche valore è stato modificato o si è perso e quindi non è consistente. Questo problema si ha soprattutto quando si fa un merge di più database, ovvero più insiemi di dati che vengono assemblati. Queste incosistenze vanno risolte perchè possono influenzare gli algoritmi di data mining. Per fare delle verifiche sul nostro database a disposizione si usano dei **metadata** che sono delle informazioni ulteriori rispetto ai dati (range, tipo di dati, la distribuzione). Ad esempio se considero il dato "età dell'utente" e trovo un valore negativo probabilmente c'è un incostituzionalità. Esistono dei prodotti commerciali per fare delle verifiche, i **data scraping** che usano una conoscenza molto semplice sul dominio o i **data auditing** che invece servono a scoprire relazioni più complesse per verificare se ci sono delle violazioni.

Adesso parleremo dell'idea di correlazione per capire se ci sono degli attributi che sono ridondanti, ovvero esprimono informazione che non è diversa per lo scopo prefissato. Se due attributi sono uguali ci stanno dando lo stesso contributo per il problema che vogliamo risolvere.

Analisi di correlazione

Correlation Analysis (Nominal Data)

- **X² (chi-square) test:** The chi-square independence test is a procedure for testing if two categorical variables A and B are related in some population.
- Suppose:
 - A= {a₁,..., a_c}
 - B= {b₁,..., b_r}
- The data tuples can be represented by a contingency table

	b ₁	b ₂	...	b _r	Occurrence frequency
a ₁	$o_{1,1}$	$o_{1,2}$		$o_{1,c}$	$o_{1..}$
a ₂	$o_{2,1}$	$o_{2,2}$		$o_{2,c}$	$o_{2..}$
...					
a _c	$o_{r,1}$	$o_{r,2}$		$o_{r,c}$	$o_{r..}$
	$o_{.,1}$	$o_{.,2}$		$o_{.,r}$	Tot



Si applica a dati nominali. Questo tipo di attributo è un valore i cui attributi sono etichette ed è caratterizzato dal NON avere un ordinamento. Si esegue con un test statistico, detto **X² (chi-square test)**. **Serve a verificare se due variabili sono indipendenti tra di loro**. Supponiamo di avere attributi **A** e **B**. Con {a₁...a_c} individuiamo i possibili valori di **A**, mentre con {b₁...b_r} i possibili valori di **B**. Questa tabella rappresenta una **tabella di contingenza**. In corrispondenza di ogni valore di A e B dice il numero di istanze che hanno quel valore di A e di B. Ogni cella rappresenta la frequenza di occorrenza

INSIEME dei valori a_i, b_j . L'oggetto rappresentato in figura corrisponde al numero di occorrenze nel dataset che hanno per valore a_1 e per valore b_1 .

- **χ^2 (chi-square) test:**

$$\chi^2 = \sum \frac{(Observed - Expected)^2}{Expected}$$

- The larger the χ^2 value, the more likely the variables are related
- The cells that contribute the most to the χ^2 value are those whose actual count is very different from the expected count

$$\chi^2 = \sum_{i=1}^c \sum_{j=1}^r \frac{(o_{ij} - e_{ij})^2}{e_{ij}}, \quad e_{ij} = \frac{\text{count}(A = a_i) \times \text{count}(B = b_j)}{n}$$

... A. E. P. x

I valori aspettati si calcolano in questo modo:

si considera come assunzione di partenza che A e B siano attributi indipendenti e vuole andare a calcolare quali sono il numero di occorrenze di a_1, b_1 stimandoli tramite probabilità. Stimando che sono **indipendenti**, la probabilità congiunta di avere a_1 e b_1 insieme si calcola come prodotto della probabilità di avere a_1 e la probabilità di avere b_1 , quindi il valore aspettato non è altro che il prodotto della probabilità di avere a_i e la probabilità di avere b_j diviso per il numero totale di istanze. Questa informazione è fondamentale per capire come funziona il chi square, perché se il valore che abbiamo misurato dalle tabelle coincide all'incirca con quello calcolato ipotizzandole indipendenti, allora le 2 variabili sono indipendenti. Quindi calcolare il χ^2 significa calcolare se i valori che osserviamo sono vicini a quelli che ci saremmo aspettati considerando i due attributi indipendenti. Più chisquare è alto e più le variabili potrebbero essere correlate.

Esempio:

Chi-Square Calculation: An Example

	Play chess	Not play chess	Sum (row)
Like science fiction	250(90)	200(360)	450
Not like science fiction	50(210)	1000(840)	1050
Sum(col.)	300	1200	1500

- χ^2 (chi-square) calculation
 - numbers in parenthesis are expected counts calculated based on the data distribution in the two categories

$$e_{ij} = \frac{\text{count}(A = a_i) \times \text{count}(B = b_j)}{N}$$

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

- It shows that `like_science_fiction` and `play_chess` are correlated in the group

... DIGI

Il primo attributo esprime quanti studenti in un sondaggio che abbiamo fatto giocano a scacchi, 300 giocano, 1200 no. La seconda ha chiesto quali apprezzano le fiction scientifiche e quali no.

Possiamo calcolarci i valori aspettati, ad esempio della prima cella in alto a sinistra (play chess, like science fiction). Prendiamo le probabilità di giocare a scacchi, calcolandola come:

il numero totale di studenti che giocano a scacchi diviso il numero totale di studenti (300/1500). Stesso calcolo per studenti che gli piacciono le science fiction (450/1500). Moltiplicando queste due probabilità e moltiplicando per il numero totale di studenti (1500) ottengo 90 che è il valore che io mi aspetterei per la combinazione (play chess, like science fiction) nella ipotesi che le 2 variabili siano indipendenti.

Ripeto il calcolo per tutti i 4 valori e calcolo il chi square.

Il chisquare a prima vista sembra un valore alto, in realtà lo è ma per esser sicuro che le 2 variabili sono ridondanti, si ricorre ai libri di statica e nei libri di statistica si trovano delle tabelle che esprimono quanto corrispondentemente rispetto a un valore di χ^2 decidere con un certo grado di affidabilità o confidenza se gli attributi sono ridondanti. Si fissa un **livello di confidenza** ovvero ci dice quanto possiamo esser sicuri della conclusione che troveremo. L'altra informazione necessaria sono i gradi di libertà: (il numero di righe della tabella di contingenza - 1) * (numero di colonne della tabella di contingenza meno 1). Nel nostro caso è 1.

Più si prende piccolo il valore di confidenza più è probabile che in realtà A e B non siano indipendenti, quindi cresce la probabilità che le variabili siano correlate. Per essere sicuri bisogna usare un valore 0.001. I gradi di libertà, ovvero il numero di righe e colonne della tabella di contingenza. Avendo fissato il numero 250 avendo le somme ho tutti gli altri numeri, il grado di libertà è 1 perché fissando un valore ho anche tutti gli altri. Quanti numeri posso cambiare prima di bloccare tutto perchè avendo le somme non posso fare altro.

Correlation Analysis (Nominal Data)

- The χ^2 statistics tests the hypothesis that A and B are independent.
- The test is based on a significance level (0.001 or 0.005, typically), with $(r-1)(c-1)$ degrees of freedom, where r and c are the number of rows and columns in the table
- The null hypothesis is rejected when the value of the computed χ^2 is higher than the value in the table corresponding to the significance level

Percentage Points of the Chi-Square Distribution

Degrees of Freedom	Probability of a larger value of χ^2								
	0.99	0.95	0.90	0.75	0.50	0.25	0.10	0.05	0.01
1	0.000	0.004	0.016	0.102	0.455	1.32	2.71	3.84	6.63
2	0.020	0.103	0.211	0.575	1.386	2.77	4.61	5.99	9.21
3	0.115	0.352	0.584	1.212	2.366	4.11	6.25	7.81	11.34
4	0.297	0.711	1.064	1.923	3.357	5.39	7.78	9.49	13.28
5	0.554	1.145	1.610	2.675	4.351	6.63	9.24	11.07	15.09



Consideriamo la riga corrispondente al nostro grado di libertà, fissiamo un valore di confidenza che vogliamo avere (0.01 ad esempio), la regola ci dice che:

se il valore di X^2 calcolato è maggiore di quello in tabella allora le variabili NON sono indipendenti.

Il chisquare è un test statistico che ci serve a capire se due attributi sono correlati tra di loro. Per applicarlo dobbiamo avere a disposizione la matrice di contingenza, calcolarlo e poi confrontarlo con quello in tabella. Al diminuire del valore di confidenza il valore del X^2 aumenta. Ovvero più diminuiamo la probabilità della indipendenza, più aumenta il chisquare ovvero, un alto valore di X^2 che avevamo osservato intuitivamente vuol dire avere una forte correlazione tra le variabili.

Si applica il chisquare a dati nominali, quindi tipicamente si ha un numero limitato di valori.

Esiste una variante che consente di applicare un qualcosa simile al chisquare per dati numerici e si chiama **coefficiente di correlazione**:

Correlation Analysis (Numeric Data)

- Correlation coefficient (also called **Pearson's product moment coefficient**)

$$r_{A,B} = \frac{\sum_{i=1}^n (a_i - \bar{A})(b_i - \bar{B})}{\sqrt{n\sigma_A\sigma_B}} = \frac{\sum_{i=1}^n (a_i b_i) - n\bar{A}\bar{B}}{\sqrt{n\sigma_A\sigma_B}}$$

where n is the number of tuples, A and B are the respective means of A and B , σ_A and σ_B are the respective standard deviations of A and B , and $\Sigma(a_i b_i)$ is the sum of the AB cross-product.

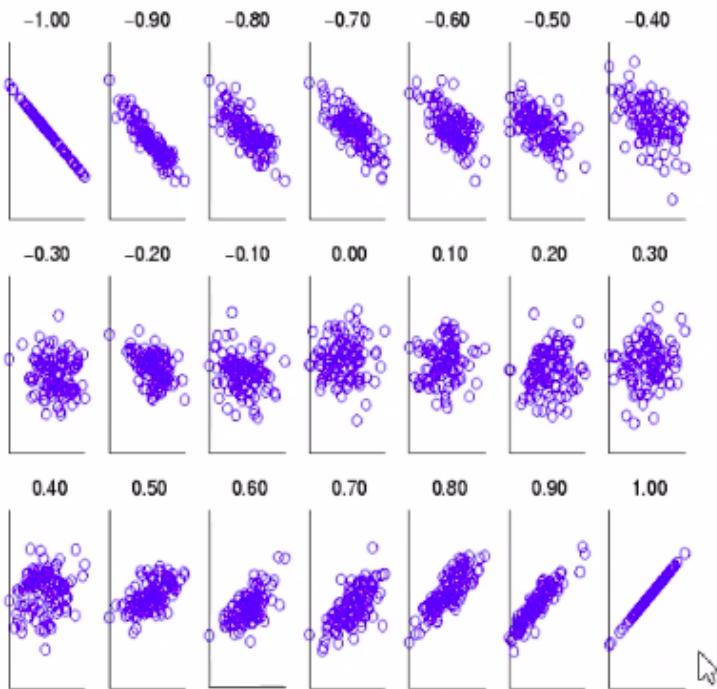
- If $r_{A,B} > 0$, A and B are positively correlated (A 's values increase as B 's). The higher, the stronger correlation.
- $r_{A,B} = 0$: independent; $r_{A,B} < 0$: negatively correlated

■ è il numero di campioni che abbiamo, \bar{A} è il valor medio di A (\bar{B} quello di B). Qual'è l'intuizione che sottende il coefficiente di correlazione?

Supponiamo che A e B siano correlate tra di loro, ad esempio al crescere di A cresce anche B . Siccome sottraiamo il valor medio ci aspettiamo che partendo da valori bassi di A , $a_i - \bar{A}$ è negativo, ma anche $b_i - \bar{B}$ sarà negativo, fin quando non avremo valori di A e di B che superano i loro valori medi, in quanto variano nello stesso modo. Facendone il prodotto sarà negativo, quando a_i e b_i supereranno il valor medio, saranno positivi e anche il prodotto sarà positivo. Se entrambi crescono allo stesso modo abbiamo che il valore del coefficiente di correlazione crescerà. Invece se abbiamo una correlazione inversa, cresce a e decresce b , avremo un prodotto negativo, e rimane tale in quanto crescono e decrescono in modo opposto. Cosa succede se non hanno correlazione? Ci aspettiamo che a volte a_i sarà maggiore del valor medio e in corrispondenza, b_i sarà maggiore o minore del valor medio, quindi ci aspettiamo un valore di $r_{A,B}$ piccolo intorno allo 0.

Il coefficiente di correlazione è una rappresentazione in numeri ma la stessa informazione possiamo usare lo scatter plot, ovvero la rappresentazione in due dimensioni dei dati. Il primo attributo viene

rappresentato nelle x il secondo nelle y e si fa il plot. Se i due attributi sono correlati mi aspetto di vedere una linea (vedi il primo plot sulla sinistra con correlazione negativa, al crescere delle x decresce la y):



Scatter plots showing the similarity from –1 to 1.

Con correlazione 0, i punti sono distribuiti in modo casuale ed eterogeneo sul piano.

Se scopro che i miei attributi hanno correlazione negativa o positiva vicina a 1 o -1 posso dire che sono ridondanti, perchè il contenuto informativo è simile, perchè mi aspetto che non contribuisca a risolvere il problema che sto risolvendo.

Nel caso in cui applichi il chisquare o l'analisi della correaltione, riduco la dimensione ovvero il numero di attributi usati per descrivere gli oggetti, da non confondere con la riduzione della numerosità, ovvero ridurre il numero di istanze che sto considerando. La riduzione si ottiene facendo un campionamento.

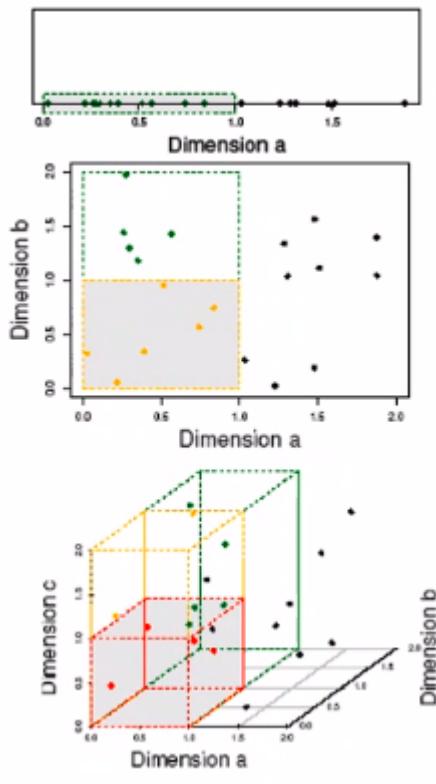
Un'altra possibilità per ridurre i dati sono la compressione.

Data Reduction Strategies

- **Data reduction:** Obtain a reduced representation of the data set that is much smaller in volume but yet produces the same (or almost the same) analytical results
- **Why data reduction?** — A database/data warehouse may store terabytes of data. Complex data analysis may take a very long time to run on the complete data set.
- **Data reduction strategies**
 - Dimensionality reduction, e.g., remove unimportant attributes
 - Wavelet transforms
 - Principal Components Analysis (PCA)
 - Feature subset selection, feature creation
 - Numerosity reduction (some simply call it: Data Reduction)
 - Regression and Log-Linear Models
 - Histograms, clustering, sampling
 - Data cube aggregation
 - Data compression



Quando ho un'unica dimensione, nella prima figura in alto, i punti appaiono vicini, se aggiungo un'ulteriore dimensione, i punti sono più lontani, se aggiungo un'ulteriore dimensione i punti si allontano ancora di più. Proiettato in un numero elevato di dimensioni, i punti si allontano, questo è un problema, se pensiamo all'idea di trovare gruppi di punti simili tra loro (algoritmi di clustering) i punti appariranno lontani tra di loro.



11/02/2023

Abbiamo accennato nella lezione precedente delle strategie per ridurre i dati. Dobbiamo però distinguere due dimensioni di riduzione:

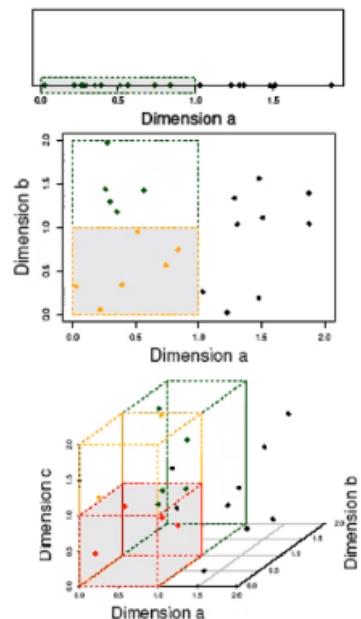
- **la dimensione relativa al numero di caratteristiche che descrivono un oggetto:** in questo caso si parla di riduzione della dimensionalità dell'insieme di dati e cercheremo di capire se possiamo ridurre alcune caratteristiche che riteniamo non utili per il task da risolvere. Il χ^2 aiuta nel fare questo. La ridondanza significa avere la stessa informazione rispetto al problema che vogliamo risolvere. Quando infatti parliamo di correlazione parliamo di due caratteristiche che hanno lo stesso trend quindi portano la stessa informazione. In questo caso la riduzione dei dati si ottiene togliendo la caratteristica dalla descrizione dell'oggetto.
- **la dimensione relativa alla numerosità:** significa ridurre il numero di oggetti con cui lavoriamo, ad esempio quando abbiamo una grande quantità di dati e dobbiamo usare algoritmi di apprendimento che richiedono molto tempo per l'esecuzione. Una tecnica è quella di campionare un insieme di dati e per farlo esistono diversi modi.

Riduzione della dimensionalità

Viene richiesta in quanto riducendo la dimensionalità riduciamo i dati perché togliamo alcune dimensioni che descrivono i dati stessi. In alcuni casi è obbligatoria perché alcuni algoritmi con dimensioni elevate "soffrono". Nel caso in cui esaminiamo un insieme di dati in un'unica dimensione i punti appaiono vicini come nell'immagine in basso. Se aggiungiamo un'ulteriore dimensione, appaiono più distanti:

Data Reduction: Dimensionality Reduction

- **Curse of dimensionality**
 - When dimensionality increases, data becomes increasingly sparse
 - Density and distance between points, which is critical to clustering, outlier analysis, becomes less meaningful
 - The possible combinations of subspaces will grow exponentially
- **Dimensionality reduction**
 - Avoid the curse of dimensionality
 - Help eliminate irrelevant features and reduce noise
 - Reduce time and space required in data mining
 - Allow easier visualization
- **Dimensionality reduction techniques**
 - Unsupervised: Wavelet transforms, Principal Component Analysis
 - Supervised and nonlinear techniques (e.g., feature selection)



Proiettando questo esempio in una situazione di dimensionalità elevata porta ad una rappresentazione dei punti molto distanti tra di loro.

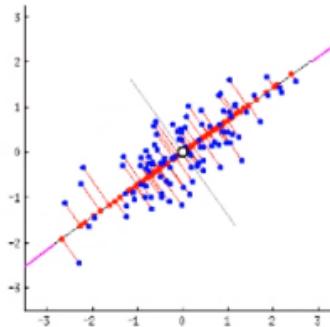
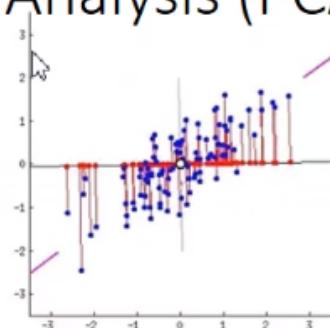
Può essere un problema in quanto uno dei task che affronteremo sarà quello di raggruppare gli oggetti, ovvero creare i **cluster di oggetti**, cercando di mettere insieme oggetti che sono vicini tra di loro ma lontani da oggetti di altri cluster. Ma se all'aumentare della dimensione tutti gli oggetti appaiono lontani tra di loro diventa difficile clusterizzarli. Questo problema è noto in letteratura come **Curse of dimensionality**.

Per poter essere sicuri che gli algoritmi che stiamo eseguendo ci portino ad avere dei risultati affidabili andiamo a ridurre la dimensionalità. Questo non significa però rimuovere delle caratteristiche a caso degli oggetti in esame. Vedremo in questa lezione **2** strategie usate:

1. **PCA Principal component analysis**: è una trasformazione dallo spazio di origine a quello finale e si esegue una riduzione delle caratteristiche nel nuovo spazio. Questa trasformazione cerca di andare in un nuovo spazio dove lungo ogni dimensione si cerca di allargare le differenze tra gli oggetti.

Principal Component Analysis (PCA)

- The projection error is less than in the original dataset
- Newly projected red points are more widely spread out than in the original dataset, i.e. more variance.



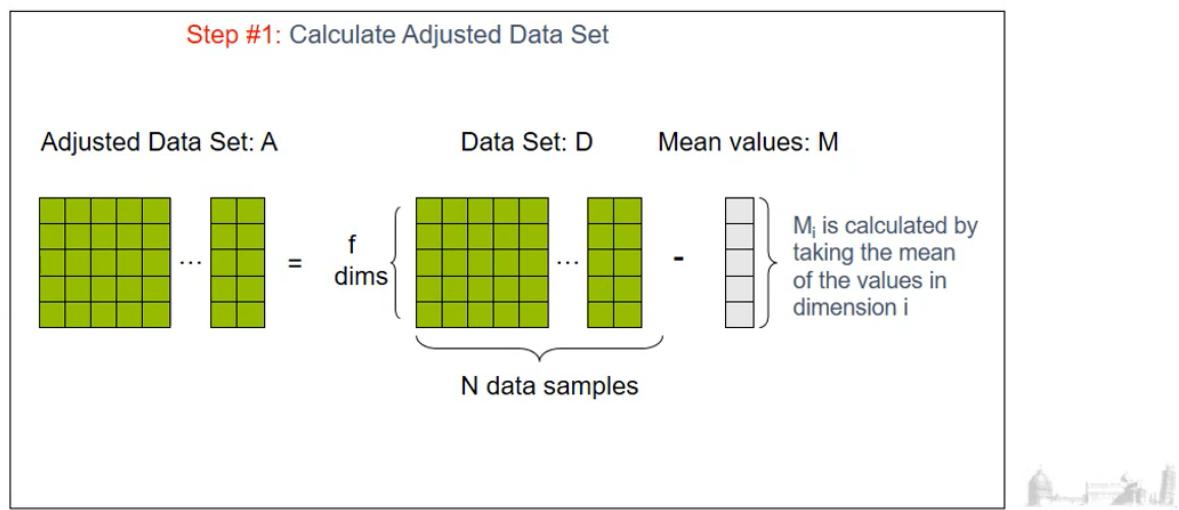
Pensiamo di proiettare questi dati lungo l'asse \hat{x} . Proiettando i dati lungo l'asse \hat{x} essi appaiono molto sovrapposti almeno in alcune zone (i puntini in rosso). Se molti valori sono sovrapposti diventa difficile poter distinguere tra vari oggetti. Supponiamo di eseguire una trasformazione per cui invece di usare l'asse x usiamo un nuovo asse che è ottenuto attraverso una trasformazione che è rappresentata nel piano cartesiano in basso.

Se proiettiamo ora i punti lungo il nuovo asse vedremo che vi è una maggiore separazione tra i punti, quindi li ha resi più distinguibili. Quello che aspiriamo a fare è dunque trovare un nuovo spazio (ovvero nuovi assi di riferimento) in modo tale che la proiezione dei nostri punti lungo questi assi consenta di avere dei valori che sono distinguibili.

Questo è possibile ottenerlo attraverso una trasformazione che va ad usare la **matrice di covarianza**. Essa si ottiene dalla matrice dei dati andando a sottrarre il valor medio e moltiplicando la matrice per la sua trasposta

Principal Component Analysis (PCA)

Given N data vectors from f -dimensions, find $k \leq f$ orthogonal vectors (*principal components*) that can be best used to represent data



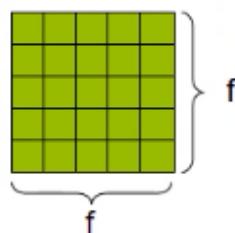
La matrice di covarianza avrà dimensione $f \times f$ dove f è la **dimensione delle caratteristiche**:

Principal Component Analysis (PCA)

Step #2: Calculate Co-variance matrix, C, from adjusted data set, A

Remember: It is a measure of the extent to which corresponding elements from two sets of ordered data move in the same direction.

Co-variance Matrix: C



$$\text{COV}(X, Y) = \frac{\sum_{i=1}^N (X_i - \bar{X})(Y_i - \bar{Y})}{N - 1}$$

Note: Since the means of the dimensions in the adjusted data set, A, are 0, the covariance matrix can simply be written as:

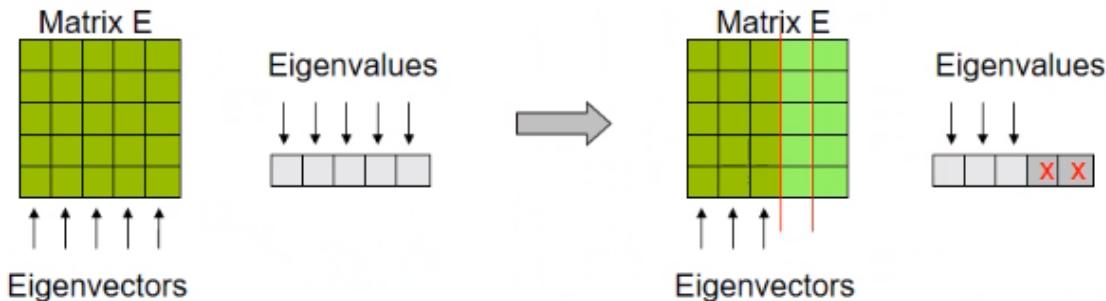
$$C_{ij} = \text{cov}(i, j)$$

$$C = (A A^T) / (n-1)$$

e andiamo di questa matrice di covarianza a trovare autovettori e autovalori.

Principal Component Analysis (PCA)

Step #3: Calculate eigenvectors and eigenvalues of C



If some eigenvalues are 0 or very small, we can essentially discard those eigenvalues and the corresponding eigenvectors, hence reducing the dimensionality of the new basis.

LIVELT

Se usassi la matrice completa di autovettori che serve per fare questa trasformazione, otterremmo che partendo dall'insieme di dati iniziali e applicando la matrice che abbiamo trovato calcolando gli autovettori della matrice di covarianza, arriviamo ad avere il nostro insieme di dati trasformato. Ma otterrei un insieme di dati trasformato con un numero di caratteristiche che avevo inizialmente, ovviamente le caratteristiche sono differenti in quanto lo spazio è stato trasformato, ma il numero sarebbe lo stesso. Quindi NON AVREMMO OTTENUTO NESSUN VANTAGGIO. In questa trasformazione però vengono in aiuto gli **autovalori**, perché un valore grande di autovalore è associato ad un autovettore in cui effettivamente abbiamo grande distinzione dei valori ovvero un alta variabilità dei valori.

Autovalori di valore elevato corrispondono ad autovettori in cui i valori degli oggetti sono distinguibili, ovvero molto sparsi. Quando calcolo autovettori e autovalori della matrice di covarianza, posso ridurre la dimensionalità andando ad eliminare gli autovettori che corrispondono a valori bassi di autovalori. Questi autovettori mi daranno dei nuovi assi dove ho una bassa variabilità degli autovettori, quindi mi servono a poco. Questo è il vantaggio che ho, non devo usare l'intera matrice degli autovettori ma uso solo gli autovettori che corrispondono ad autovalori grandi.

Principal Component Analysis (PCA)

↳ Step #4: Transforming data set to the new basis

$$F = E^T A$$

where:

- F is the transformed data set
- E^T is the transpose of the E matrix containing the eigenvectors
- A is the adjusted data set

Note that the dimensions of the new dataset, F, are less than the data set A

To recover A from F:

$$\begin{aligned}(E^T)^{-1} F &= (E^T)^{-1} E^T A \\ (E^T)^T F &= A \\ EF &= A\end{aligned}$$

* E is orthogonal, therefore $E^{-1} = E^T$

🔥 Osservazioni importanti:

- Il nuovo insieme di dati è definito su un sottospazio composto dagli autovettori che ho selezionato. La selezione delle caratteristiche è avvenuta nello spazio trasformato e non in quello originale. Le nuove caratteristiche che ho selezionato sono una combinazione di quelle precedenti. E' vero che ho ridotto lo spazio ma non conosco esattamente quali caratteristiche dello spazio originale mi stanno influenzando le nuove caratteristiche che ho trovato. Il mio vantaggio però è che lavoro con un insieme di dati che ora è definito su un insieme ridotto di caratteristiche e quelle caratteristiche sono state trovate cercando di incrementare la variabilità dei valori degli oggetti lungo quelle caratteristiche. Dunque da un punto di vista degli algoritmi che dovrò andare ad usare per fare data mining, effettivamente mi aspetto che questo data set sia migliorato rispetto a quello originale.
- Quando abbiamo fatto questa trasformazione, abbiamo semplicemente calcolato la matrice di covarianza, calcolato autovettori e autovalori della matrice di covarianza. Queste sono trasformazioni matematiche ma non abbiamo usato alcuna informazione sull'insieme di dati, ovvero nessuna conoscenza ulteriore. Come viene detto in letteratura, questa trasformazione non ha usato alcuna forma di ground truth, semplicemente abbiamo preso il nostro insieme di dati con i valori associati alle caratteristiche di ciascuno di essi e abbiamo lavorato su questi valori. Ovviamente quando abbiamo il dato trasformato, se usassimo tutti gli autovettori, tramite una trasformazione inversa, potremmo recuperare tutto l'insieme di dati originale. Nella slide sopra è presente la trasformazione inversa effettuata. Quando invece uso un sottoinsieme di autovettori, durante la trasformazione inversa non recupererò l'insieme di dati originali, ma ne avrò una approssimazione molto vicina, in quanto sto prendendo gli autovettori con maggiore variabilità degli autovalori .

Questa trasformazione fatta usando la PCA è molto utile in quanto non avendo necessità di ulteriori informazioni se non la descrizione degli oggetti da cui si parte, può essere usata per la trasformazione della deduzione delle caratteristiche, ovvero per problemi di classificazione, clustering, predizione, etc...

2. **Direttamente ridurre le dimensioni nello spazio di origine.** In questa seconda tipologia si usano degli approcci euristici. Queste vengono applicate direttamente sull'insieme di dati e sullo spazio in cui l'insieme dei dati è definito. Si basano su una selezione in avanti o su una eliminazione all'indietro.

Heuristic Search in Attribute Selection

Forward selection	Backward elimination	Decision tree induction
<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>Initial reduced set: $\{\}$ $\Rightarrow \{A_1\}$ $\Rightarrow \{A_1, A_4\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <p>$\Rightarrow \{A_1, A_3, A_4, A_5, A_6\}$ $\Rightarrow \{A_1, A_4, A_5, A_6\}$ \Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>	<p>Initial attribute set: $\{A_1, A_2, A_3, A_4, A_5, A_6\}$</p> <pre> graph TD A4[A4?] -- Y --> A1[A1?] A4 -- N --> A6[A6?] A1 -- Y --> Class1_1((Class 1)) A1 -- N --> Class2_1((Class 2)) A6 -- Y --> Class1_2((Class 1)) A6 -- N --> Class2_2((Class 2)) </pre> <p>\Rightarrow Reduced attribute set: $\{A_1, A_4, A_6\}$</p>

- **Forward selection:** Nella selezione in avanti, partiamo da un insieme iniziale di caratteristiche che è vuoto e andiamo ad ogni iterazione a selezionare la caratteristica migliore. Il problema sarà capire come identificare la caratteristica migliore. Ci si ferma quando abbiamo raggiunto il numero prestabilito di caratteristiche che volevamo.
Oppure vengono usati degli indici che ci consentono di creare delle condizioni di terminazione.
- **Backward elimination:** Nel caso di eliminazione all'indietro partiamo da un insieme con tutte le caratteristiche e ad ogni iterazione eliminiamo il peggiore. Il problema sarà definire il concetto di "peggiore".
Quando ci fermiamo nella strategia di eliminazione all'indietro? Come nel caso della selezione in avanti ci fermiamo o quando abbiamo raggiunto il numero prestabilito di caratteristiche che volevamo oppure usando dei particolari indici.
- **Decision tree:** L'altra strategia è un albero di decisione che in realtà è un classificatore con una caratteristica interessante, ovvero quando andiamo a generare questo classificatore, questo implicitamente consente anche di fare una selezione delle caratteristiche. Ne parleremo più avanti quando tratteremo i classificatori.

Esempio di approccio euristico

Questo approccio proposto nel 2009 esegue una selezione degli attributi in avanti, quindi con strategia per cui si parte con un insieme vuoto e ad ogni iterazione viene introdotto l'attributo migliore da inserire tra quelli selezionati sfruttando un meccanismo euristico.

Heuristic Search in Attribute Selection

Example of heuristic approaches (Be careful! You need ground truth)

- Pablo A. Estévez, Michel Tesmer, Claudio A. Perez, and Jacek M. Zurada, "Normalized Mutual Information Feature Selection", IEEE Transactions on neural networks, Vol. 20, N. 2, February 2009.
- Consider two discrete variables X and Y , with alphabets XX and YY , respectively. The mutual information (I) between X and Y with a joint probability mass function $p(x,y)$ and marginal probabilities $p(x)$ and $p(y)$ is defined as follows:

$$I(X, Y) = \sum_{x \in XX} \sum_{y \in YY} p(x, y) \cdot \log \frac{p(x, y)}{p(x)p(y)}$$

- Alphabets XX and YY contain the possible values for X and Y , respectively.

SIMILE DICHI

Per capire come questo approccio funziona bisogna fare riferimento alla slide.

Supponiamo di avere due attributi \boxed{x} e \boxed{y} come attributi discreti (ovvero aventi un numero limitato di possibili valori). Questi valori sono rappresentati dai loro alfabeti \boxed{xx} e \boxed{yy} . Ad esempio uno dei valori è età, e l'altro è reddito. Un terzo possibile attributo può essere l'uscita. In problemi di classificazione, l'uscita rappresenta le classi, dunque l'uscita in questo caso può essere $\boxed{\text{class1}}$, $\boxed{\text{class2}}$ e $\boxed{\text{class3}}$. Definiamo su queste variabili discrete il concetto di **informazione mutua** (I) tra X e Y .

L'informazione mutua è definita dalla formula nella figura in cui usiamo la probabilità congiunta $\boxed{p(x, y)}$ e le probabilità marginali

$\boxed{p(x)}$ e $\boxed{p(y)}$.

💡 Cerchiamo di capire intuitivamente cosa vuol fare questa formula...

Se due attributi sono **indipendenti** la probabilità congiunta si calcola come **prodotto** di quelle **marginali**. Se X e Y fossero indipendenti, ovvero non avessero informazione mutua, avremmo che $\boxed{p(x, y)}$ dovrebbe essere calcolata come prodotto di $\boxed{p(x)}$ e $\boxed{p(y)}$ ma se questo avvenisse questo rapporto nel logaritmo sarebbe dato da:

$$\log \frac{p(x, y)}{p(x)p(y)} = \log \frac{p(x)p(y)}{p(x)p(y)} = \log(1) = 0$$

Dunque questi termini sarebbero tutti $\boxed{0}$ e il valore della mutua informazione sarebbe $\boxed{0}$. Questa informazione mutua crescerà quando ci allontaneremo dall'indipendenza e quindi la probabilità congiunta sarà differente dal prodotto delle probabilità marginali.

L'informazione mutua viene calcolata sommando il prodotto della probabilità congiunta $\boxed{p(x, y)}$ per il logaritmo e viene fatto per ogni possibile combinazione di valori di \boxed{x} e \boxed{y} .

Nell'esempio dovremo calcolare la mutua informazione per ogni combinazione tra X_1 e X_2 di giovane, medio e anziano e basso,medio,alto per quanto riguarda il reddito:

Heuristic Search in Attribute Selection

Example of heuristic approaches (Be careful! You need ground truth)

- $X_1 = \text{age}$ $XX_1 = \{\text{young, medium, old}\}$
- $X_2 = \text{income}$ $XX_2 = \{\text{low, medium, high}\}$
- $Y = \text{output}$ $YY = \{\text{class1, class2, class3}\}$

$$I(X, Y) = \sum_{x \in XX} \sum_{y \in YY} p(x, y) \cdot \log \frac{p(x, y)}{p(x)p(y)}$$

- $p(x, y)$ = all the possible combinations of values in XX and YY
- If $X = \text{age}$ and $Y = \text{output}$, $p(\text{young, class1})$, $p(\text{young, class2})$, $p(\text{young, class3})$, $p(\text{medium, class1})$, $p(\text{medium, class2})$, $p(\text{medium, class3})$, $p(\text{old, class1})$, $p(\text{old, class2})$, $p(\text{old, class3})$,



Come possiamo usare questa formulazione? Abbiamo capito che la mutual information ci fa capire se dati due attributi, questi rappresentano in realtà una informazione simile. Quando l'informazione contenuta è molto simile, non ha senso portare dietro i 2 attributi. Esiste però un altro aspetto interessante. Poniamoci in un problema di classificazione:

Ho un numero di oggetti descritti dalle loro caratteristiche e ognuno appartiene a una specifica classe. Risolvere un problema di classificazione significa ricevere un oggetto con le sue caratteristiche di cui però non conosco la classe e dovrò essere in grado di etichettare l'oggetto classificandolo. Supponiamo ad esempio di avere un data set di operazioni che vengono fatte con la carta di credito e vorremmo sviluppare un classificatore in grado di riconoscere se l'operazione fatta è fraudolenta oppure no.

Abbiamo capito che possiamo usare l'informazione mutua per capire se due caratteristiche sono ridondanti, ma possiamo usare anche la mutua informazione in problemi di classificazione per capire un aspetto più importante.

Supponiamo che le 2 variabili che stiamo osservando sono una la caratteristica dell'attributo e l'altra l'uscita. Nel caso specifico dell'esempio nella slide, stiamo osservando l'attributo age e stiamo osservando l'attributo output costituito dalle 3 classi. Vogliamo calcolare la mutua informazione tra X_1 e Y .

? Perchè vorremmo calcolarla?

Se ottenessimo che una caratteristica ha una mutua informazione con l'**uscita** molto molto alta, quella caratteristica per noi è estremamente rilevante, in quanto esaminando i valori di quella caratteristica potremmo capire **DIRETTAMENTE** la classe degli oggetti.

Supponiamo che il valore giovane corrisponde sempre ad avere la classe1. Nel mio insieme di dati ho tutti oggetti che quando hanno per age → young, appartengono alla class1. Stesso discorso per age → medium in cui appartengono alla class2 e age → old appartengono alla class3.

In questo caso potrei semplicemente analizzando il valore di age stabilire quale è la classe.

! Questo ci porta a concludere che quando voglio selezionare gli attributi, dovrei selezionare gli attributi che sono caratterizzati dalla massima mutua informazione con l'uscita !

Da un lato voglio che l'attributo che sto selezionando abbia la massima informazione mutua con l'uscita ma dall'altro voglio che abbia una bassa informazione mutua con quelli già selezionati.

Supponiamo che nel mio insieme di dati ci siano 2 attributi uguali e questi 2 attributi abbiano l'informazione mutua con l'uscita più alta degli altri. Applicando un algoritmo di selezione in avanti, prenderei il primo attributo. Alla seconda iterazione mi pongo il problema di prendere un ulteriore attributo tra gli attributi disponibili. Avendo ancora un attributo pari a quello precedente, se mi basassi solo sul ragionamento di prendere un attributo con massima informazione mutua con l'output, selezionerei quell'attributo. Ma essendo quel secondo attributo uguale al primo, non mi da alcuna informazione aggiuntiva. Il problema dove sta?

Oltre a considerare la mutua informazione con l'uscita devo considerare anche la mutua informazione tra l'attributo che sto selezionando e gli attributi già selezionati.

Questo ci porta a definire questo algoritmo euristico basato su una selezione in avanti.

Si parte da un insieme di caratteristiche vuoto e quello che faccio è calcolare la mutua informazione tra ogni attributo e l'uscita C ovvero le classi. Seleziono come primo attributo quello caratterizzato dalla massima mutua informazione con l'uscita. Poi ripeto iterativamente le azioni seguenti fin quando non avrò estratto un numero di caratteristiche pari a k. Calcolo quindi la informazione mutua tra le caratteristiche già selezionate e quella che sto osservando e calcolo il valore di G che è ottenuto come:

Example di Heuristic Approach

The algorithm

1. **Initialization:** Set $F = \{f_i | i = 1, \dots, N\}$, initial set of N features, and $S = \{\emptyset\}$, empty set.
2. Calculate the MI with respect to the classes: calculate $I(f_i; C)$, for each $f_i \in F$.
3. Select the first feature: Find $\hat{f}_i = \max_{i=1, \dots, N} I(f_i; C)$. Set $F \leftarrow F \setminus \{\hat{f}_i\}$; set $S \leftarrow \{\hat{f}_i\}$
4. **Greedy Selection:** Repeat until $|S| = k$.
 - a. Calculate the MI between features: Calculate $I(f_i; f_s)$ for all pairs $(f_i; f_s)$, with $f_i \in F$ and $f_s \in S$
 - b. Select next feature: Select feature $f_i \in F$ that maximizes G . Set $F \leftarrow F \setminus \{\hat{f}_i\}$; set $S \leftarrow \{\hat{f}_i\}$.
5. Output the set S containing the selected features.

Example di Heuristic Approach

Problem statement

- Given an initial set F with n features, find subset $S \subset F$ with k features that maximizes the MI $I(C;S)$ between the class variable C , and the subset of selected features S .
- Normalized MI between f_i and f_s

$$NI(f_i, f_s) = \frac{I(f_i; f_s)}{\min\{H(f_i), H(f_s)\}}$$

- The selection criterion used in NMIFS consists in selecting the feature that maximizes the measure G

$$G = I(C, f_i) - \frac{1}{S} \sum_{f_s \in S} NI(f_i, f_s)$$

where $I(C, f_i) = \sum_{c \in CC} \sum_{f_i \in FF} p(c, f_i) \cdot \log \frac{p(c, f_i)}{p(c)p(f_i)}$

ovvero andando a sottrarre la mutua informazione tra la caratteristica che sto osservando e l'uscita e la somma tra la informazione mutua tra le caratteristiche che sto osservando e quelle che ho già selezionato. In realtà qui non uso l'informazione mutua, ma un valore di **informazione mutua con un valore di entropia al denominatore** calcolato come il minimo tra l'entropia di f_i e f_s .

Abbiamo usato la ground truth ovvero qualcuno ci ha dovuto dire che quei singoli oggetti appartengono a una specifica classe.

Dunque la misura G non è altro che l'espressione matematica di quello che abbiamo detto prima, ovvero vogliamo selezionare la caratteristica che ha la massima informazione mutua con l'uscita ma anche la minima informazione mutua con le caratteristiche già selezionate.

Questo è un approccio euristico, applico un algoritmo con selezione in avanti ma che ha un aspetto interessante: questo algoritmo lavora direttamente nello spazio originale delle caratteristiche ed è vantaggioso perché potrei scoprire che alcune caratteristiche non sono rilevanti per il task che voglio risolvere. Esiste però anche un altro aspetto, ovvero qui abbiamo usato la ground truth ovvero le classi e quindi quei singoli oggetti appartenevano a delle classi e qualcuno ha dovuto darci questa informazione. In problemi di classificazione questo tipicamente avviene durante la costruzione di un classificatore in cui si usa un insieme di dati, chiamato insieme di addestramento, che ha oggetti descritti da loro caratteristiche con anche la classe associata. Usando questi oggetti riusciremo ad apprendere attraverso un algoritmo di apprendimento il nostro classificatore. La tecnica euristica ha il vantaggio per noi che ci consente eventualmente di selezionare caratteristiche reali, sappiamo esattamente da dove sono state prese, a differenza di quello che avviene con la PCA. Questo approccio euristico ha però bisogno di avere a disposizione la classe di appartenenza degli oggetti, ovvero la ground truth. Inoltre con l'approccio euristico per avere l'ottimo bisognerebbe provare tutte le combinazioni degli attributi. Quello che facciamo ad ogni iterazione è fare la scelta ottima RELATIVAMENTE A QUELLA ITERAZIONE che non è detto sia ottima a livello globale.

Spesso **ridurre le caratteristiche non è un aspetto che va tenuto in considerazione per ridurre la complessità** ma spesso va tenuto in considerazione proprio per avere una certa affidabilità

sull'esecuzione degli algoritmi che andremo ad usare, questo per evitare il problema della curse of dimensionality.

Numerosity Reduction: Sampling

Sampling: obtaining a small sample s to represent the whole data set N

Allow a mining algorithm to run in complexity that is potentially sub-linear to the size of the data

Key principle: Choose a representative subset of the data

Simple random sampling may have very poor performance in the presence of skew

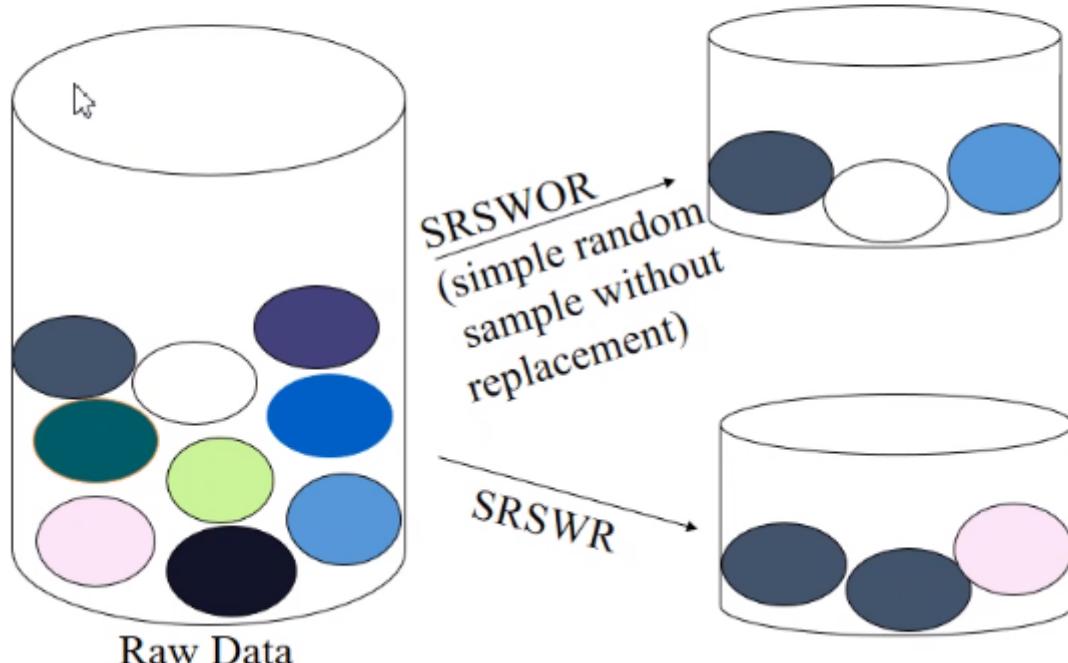
Develop adaptive sampling methods, e.g., stratified sampling

Note: Sampling may not reduce database I/Os (page at a time)

Riduzione della numerosità

Per ridurre la numerosità esistono varie tecniche, ovvero si cerca di **ridurre i campioni** andando ad estrarre casualmente un sottoinsieme. Quindi estraiamo un piccolo set di campioni che si spera sia rappresentativo per il set completo. Quando applichiamo il campionamento esistono 2 tipologie di campionamento:

Numerosity Reduction: with or without replacement



Nell'esempio mostrato in figura, campionare un insieme di dati significa selezionare un sottoinsieme delle palline contenute in **Raw Data**.

1. **campionamento con rimpiazzamento**: una volta estratto un oggetto e posto tra quelli campionati, quell'oggetto non viene rimosso dalla popolazione originale. In questo caso **copiamo** la pallina da **Raw Data** in un contenitore che conterrà il sottoinsieme finale. La pallina però rimane **anche** nel contenitore **Raw Data** iniziale. Nel caso di campionamento con rimpiazzamento, ad ogni estrazione, siccome non tolgo nessuna pallina, non cambio le probabilità di estrarre palline di una certa classe. Questo ci consente di avere un insieme di dati ridotto che dovrebbe rispecchiare perfettamente la distribuzione che abbiamo sull'insieme di dati originali. Da un punto di vista di risultato finale, quello con rimpiazzamento dovrebbe garantirci un insieme ridotto che rispecchi l'insieme di dati originale. Ovviamente c'è il rischio di avere due palline uguali, ma non è un grande problema.

1.1 **campionamento stratificato**: Il campionamento stratificato significa partizionare l'insieme di dati e poi estrarre i campioni da ciascuna partizione. Supponiamo di avere dati che appartengono a 2 classi differenti, la prima con il 60% di punti e la seconda con il 40% di punti. Avere una stratificazione significa che vorremmo avere nell'insieme di punti ridotto le stesse percentuali di partizionamento 60%-40%. Questo è molto utile quando si ha a che fare con classi poco rappresentate, ovvero quelle che sono chiamate in letteratura **classi minoritarie**.

2. **campionamento senza rimpiazzamento**: una volta estratto un oggetto, quelle viene rimosso dalla popolazione. In questo caso prendiamo la pallina da **Raw Data** e la spostiamo in un contenitore che conterrà il sottoinsieme finale. Ogni volta che estraggo un dato, cambio la distribuzione dei dati rimanenti, perchè tolgo qualcosa. Quindi quando vado ad estrarre i dati successivi, ho cambiato le probabilità di estrarre i dati di una certa classe.

❓ Quale tra i 2 tipi di campionamento ci soddisfa maggiormente?

Quello **senza rimpiazzamento**, in quanto minimizza il rischio di introdurre una forma di ridondanza nella informazione. Il nostro scopo è ridurre la numerosità, ma riducendo vogliamo che l'insieme di dati ridotto abbia la stessa distribuzione delle caratteristiche dell'insieme originale.

Numerosity Reduction: Types of Sampling

Simple random sampling

There is an equal probability of selecting any particular item

Sampling without replacement

Once an object is selected, it is removed from the population

Sampling with replacement

A selected object is not removed from the population

Stratified sampling:

Partition the data set, and draw samples from each partition (proportionally, i.e., approximately the same percentage of the data)

Used in conjunction with skewed data (also the smaller group of items will be sure to be represented)

Data Transformation

L'ultima cosa che ci rimane da vedere in questo processo di lavoro iniziale sui dati prima di passare all'applicazione di qualche algoritmo di Data Mining è la trasformazione dei dati. Per capirlo facciamo l'esempio del calcolo della distanza euclidea. Lo usiamo in quanto in diversi algoritmi useremo un concetto di distanza. Basti pensare ad un algoritmo che deve trovare cluster di oggetti. Questo algoritmo andrà a calcolare la distanza tra gli oggetti e cercherà di mettere insieme gli oggetti vicini tra di loro. Quindi c'è la necessità di andare a calcolare una distanza.

Data Transformation

Data transformation is the process of changing the format, structure, or values of data.

Data transformation techniques are applied to ensure a more efficient and quality data analysis and knowledge extraction process. Data transformation is often necessary for the application of specific data mining algorithms.

Example: Computation of the Euclidean distance

$$d(\mathbf{p}, \mathbf{q}) = d(\mathbf{q}, \mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2}$$
$$= \sqrt{\sum_{i=1}^n (q_i - p_i)^2}.$$

La formula della distanza euclidea ci dice che dobbiamo calcolare la radice quadrata della somma della differenza delle caratteristiche dei due oggetti al quadrato, caratteristica per caratteristica. Supponiamo di avere una caratteristica che varia tra 0 e 1000000 e le altre che invece variano solo tra 0 e 1.

Quando andremo a calcolare la distanza euclidea, la prima caratteristica sarà predominante sulle altre, quindi andrà a influenzare nella somma l'apporto dato dalle altre.

❓ Come si può evitare questo problema e far sì che tutte le caratteristiche pesino allo stesso modo?

Tramite la **normalizzazione**.

Normalization

Data Transformation: Normalization

- **Min-max normalization:** to $[new_min_A, new_max_A]$

$$v' = \frac{v - min_A}{max_A - min_A} (new_max_A - new_min_A) + new_min_A$$

- Ex. Let income range \$12,000 to \$98,000 normalized to [0.0, 1.0]. Then \$73,600 is mapped to

$$\frac{73,600 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0.716$$

- **Z-score normalization** (μ : mean, σ : standard deviation): $v' = \frac{v - \mu_A}{\sigma_A}$

- Ex. Let $\mu = 54,000$, $\sigma = 16,000$. Then $\frac{73,600 - 54,000}{16,000} = 1.225$

DATA

Cerca di riportare i dati trasformandoli in modo tale che ogni caratteristica abbia un range di variazione simile. Quindi quello che cerchiamo di fare è fare in modo che ogni caratteristica varii nello stesso range. Esistono 2 normalizzazioni più popolari:

1. **Min-max normalization:** fissiamo il range che vogliamo ottenere, quindi consideriamo l'attributo A e fissiamo il range su cui vogliamo che quell'attributo vari, supponiamo ad esempio che il new_min_A sia 0 e new_max_A sia 1. Ogni valore reale di questo attributo A , nella formula indicato con v , viene trasformato usando la formula nell'immagine, mappandolo in $[0,1]$. Questo ci aiuta a risolvere il problema in quanto ci basta considerare tutti gli attributi e trasformarli facendoli variare tra 0 e 1. Ogni attributo dopo la trasformazione varierà nello stesso range, ma esiste un problema con questo tipo di normalizzazione. Supponiamo ci sia la presenza di un **outlier**, avendo uno schiacciamento dei valori normali, anche gli outlier verrebbero schiacciati e si "perderebbero" nel nuovo range. Questo problema viene risolto dall'altra tecnica di normalizzazione.
2. **Z-score normalization:** Si prende il valore iniziale, si sottrae il valor medio dell'attributo e si divide per la deviazione standard. Mentre nella min-max sappiamo esattamente quale è il range dell'attributo dopo la normalizzazione, nella z-score non conosciamo a priori il range di arrivo dopo la normalizzazione. Significa che attributi diversi avranno range abbastanza simili infatti viene divisa la differenza per la deviazione standard. Se un attributo varia molto avremo un alta deviazione standard, se un attributo varia poco avremo una piccola deviazione standard, questo riesce a darci l'effetto di avere attributi che variano all'incirca sullo stesso dominio. Viene inoltre ridotto il problema dell'outlier in quanto lavoriamo su valor medio e deviazione e quindi pur avendo un outlier peserà poco nel calcolo di queste statistiche.

Una volta applicate le normalizzazioni il vantaggio è che ogni attributo varierà nel caso 1. nello stessso range mentre nel caso 2. approssimativamente nello stesso range.

Questo chiude la parte relativa alla preprocessazione dei dati. Nella sezione successiva tratteremo quali sono gli algoritmi per estrarre conoscenza dai dati.

Algoritmi di Data Mining

Molti di questi sfruttano tecniche di machine learning. Il primo problema è quello di **classificazione**:

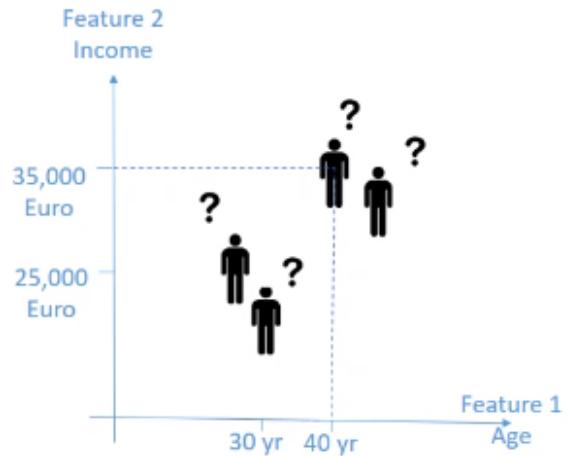
Classification problem

- **Context**

- A set of objects, each described by a set of attributes or features
 - Customers described by age, income, gender, credit rating
 - Patients described by age, family history, smoker

- **Objective**

- Associate each object with a class
 - Good customer vs. bad customer
 - Low risk patient vs. risky
 - Normal traffic vs. attack



Bisogna avere un insieme di dati con cui addestrare un classificatore fatto da tante istanze che descrivono le nostre caratteristiche (ad es. le operazioni bancarie) e se quegli oggetti (ad es. le operazioni bancarie) appartengono a una determinata classe (ad es. ad una classe fraudolenta oppure no).

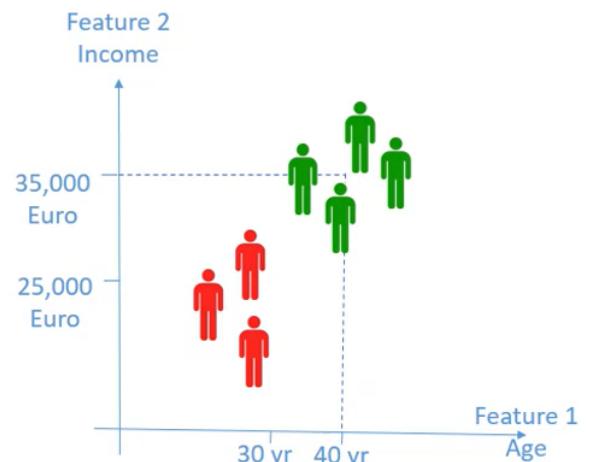
Avendo a disposizione questo insieme vogliamo classificare le **NUOVE** operazioni fatte con la carta di credito.

⚠ Il nostro problema diventa quello di: *avendo a disposizione oggetti non classificati, associare a quegli oggetti una classe. Gli oggetti sono descritti da un insieme di attributi, ad esempio il sesso, il rating della carta di credito, l'età etc...*

❓ Come possiamo ragionare?

Classification problem

- **Starting point**
 - Domain knowledge?
 - A set of labelled objects (objects associated with a class)
 - Training set
- **How?**
 - Learn from training set
 - Infer the class of other objects



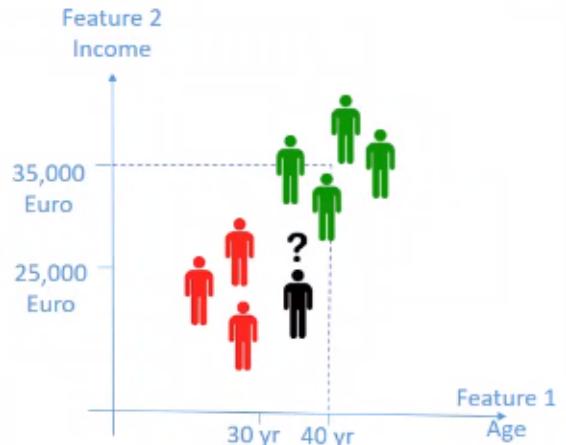
Abbiamo detto che ci serve qualche conoscenza del dominio e ci è data da un numero di oggetti con associata la classe di appartenenza, quella che in letteratura è definito come l'**insieme di addestramento**. Abbiamo la classe rappresentata dal colore verde e quella rappresentata dal colore rosso nell'immagine. Il training set ci serve per addestrare il nostro classificatore.

💡 Come possiamo classificare un oggetto sconosciuto?

Il pensiero più semplice è quello di sfruttare l'insieme di addestramento senza creare un modello, quindi il nostro classificatore più sciocco può funzionare in questa forma: *se riceviamo un oggetto non etichettato ed è identico a quello presente nell'insieme di addestramento possiamo associarlo a quello. Se invece non abbiamo alcun oggetto uguale nell'insieme di addestramento non associamo alcuna classe.*

Lazy learners

- **No model**
 - Exploit the training set during the classification phase
- The most stupid lazy learner: the Rote learner
 - Does there exist an object identical to the unlabelled object?
 - If Yes -> associate the same class to X
 - If No -> no class is associated (do not know)



Overtraining: no generalization

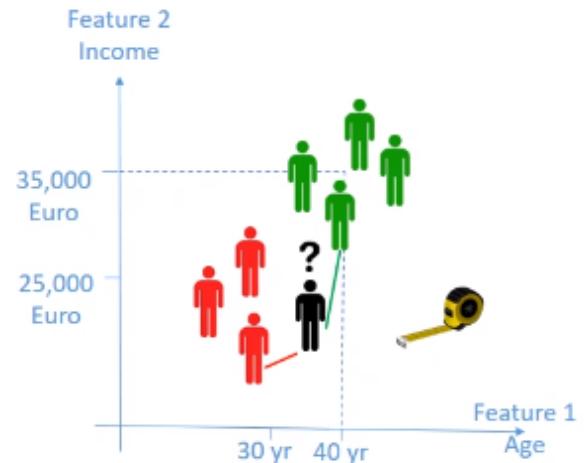
Un classificatore di questo tipo non ci soddisfa, perché normalmente riceveremo oggetti che non sono coincidenti con quelli presenti nel training set. Questo classificatore molto banale soddisfa ben poco perchè nel caso in cui abbiamo un oggetto leggermente differente rispetto a quelli del training set non

riusciamo a classificarlo.

Il fatto che però non riusciamo ad associare una classe ad oggetti differenti dagli oggetti presenti nel training set in letteratura è definito come **overtraining** in quanto non riusciamo a risolvere il problema perchè non riusciamo a generalizzare.

Un modo semplice potrebbe essere quello di usare la **distanza**, quindi vado a calcolare l'oggetto più vicino calcolando la distanza tra l'oggetto incognito e gli oggetti nel training set prendendo l'oggetto meno distante dall'oggetto incognito.

- 1-nearest neighbours
 - Classify based on the most similar example
 - Problem
 - Noise

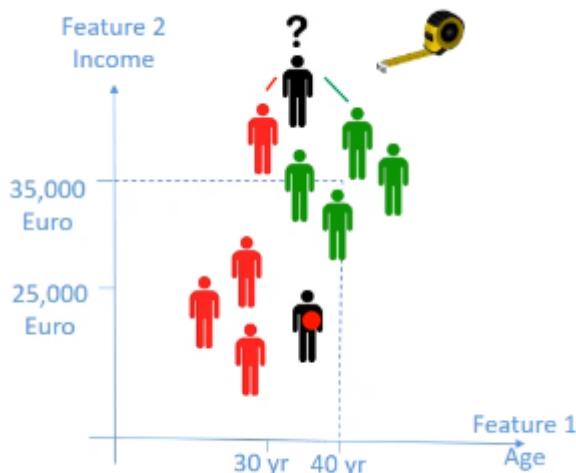


Nel caso specifico presentato in figura, calcolo la distanza tra l'oggetto che mi è arrivato e tutti gli oggetti nel training set e scopro che l'oggetto più vicino appartiene alla classe dei rossi. Posso dedurre dunque che siccome l'oggetto più vicino e quello più simile appartiene alla classe dei rossi, posso darmi la regola di classificare l'oggetto con la stessa classe dell'oggetto più vicino.

Questo classificatore si chiama **1-Nearest Neighbour (1NN)** e risolve il problema dell'overtraining, ma da vita ad un altro problema...

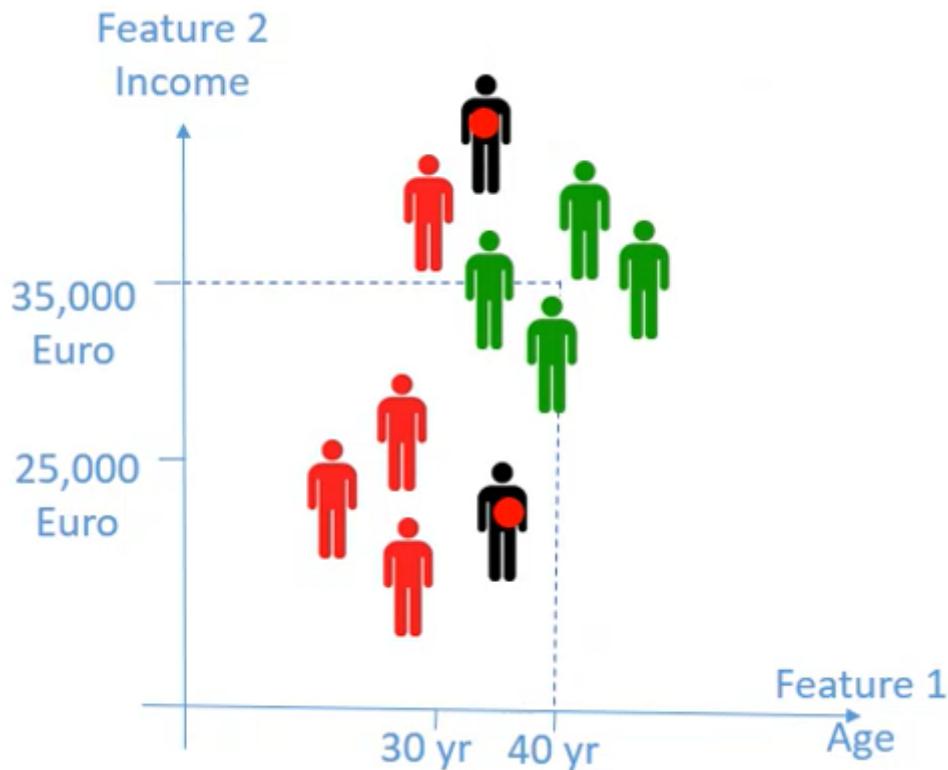
Supponiamo che il training set sia distribuito come nella immagine, ovvero abbiamo un campione appartenente alla classe rossa tra i campioni appartenenti alla classe verde e supponiamo che arrivi come punto incognito quello con il **?**.

L'oggetto incognito si trova in una zona dove la maggioranza sono oggetti di classe verde.



Applicando però il 1NN, quell'oggetto di classe rossa che probabilmente è un oggetto **rumoroso** all'interno degli oggetti di classe verde, risulta l'oggetto più vicino quindi applicando il 1NN, l'oggetto

incognito verrà classificato come rosso!!! **POCO PLAUSIBILE GUARDANDO LA DISTRIBUZIONE NELL'IMMAGINE!**



Quindi il 1NN ha questo problema, ovvero se all'interno di oggetti che sono tutti della stessa classe ho rumore, l'oggetto incognito lo classifico con la classe dell'oggetto rumoroso.

❓ Come risolverlo?

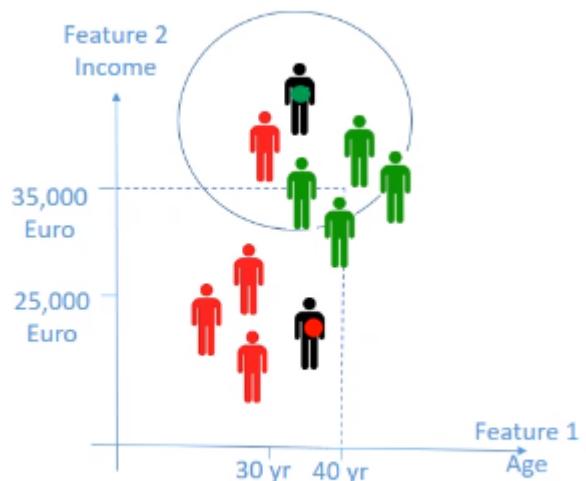
Con il **K-Nearest Neighbour (KNN)**.

Vengono considerati K oggetti vicini anzichè uno solo, quindi si considera una sorta di vicinato e si va a vedere qual'è la classe di maggioranza nel vicinato. Nell'esempio in basso, con $K = 3$ si trova che 2 oggetti sono di classe verde e uno è di classe rossa, dunque il nuovo oggetto incognito è probabilmente di classe verde! **MOLTO PLAUSIBILE GUARDANDO LA DISTRIBUZIONE NELL'IMMAGINE!**

Lazy learners

- **k-nearest neighbours**

- Decide the label based on the k closest labelled objects
 - More robust to noise
 - Computationally heavy when the number of objects in the training set is very large
 - Compute the distance between the unlabelled object and each object in the training set
 - Sort the distances in increasing order and consider the first k distances



Ovviamente passare dal 1NN al KNN ci rende il classificatore più robusto al rumore. I problemi da affrontare però sono molteplici:

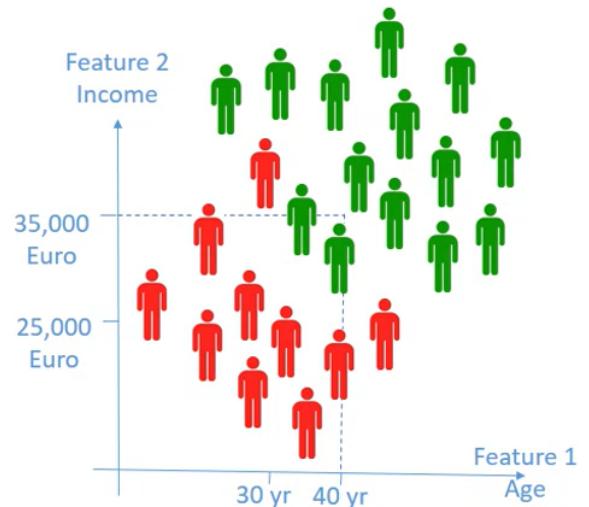
- la scelta di K in quanto quale valore di K è ottimale? Non esiste una ricetta unica, bisogna provare K crescenti in un range e andare ad usare il K che consente di avere una accuratezza maggiore nella classificazione.
- l'aspetto computazionale inoltre incide, in quanto il KNN è pesante, perché per ogni nuovo oggetto incognito dobbiamo calcolare la distanza tra l'oggetto incognito e tutti quelli nell'insieme di addestramento, poi dobbiamo ordinare queste distanze e prendere le K distanze più piccole. Calcolare la distanza ha una complessità lineare con il numero di oggetti, eseguire l'ordinamento delle distanze ha una complessità pari a $n \log(n)$ dunque se il numero di oggetti è elevato, la complessità non è trascurabile. Questa complessità è percepita dall'utente finale, perché questo è il tempo che viene adoperato per classificare. Questo approccio è tra quelli più antichi e più semplice ma anche computazionalmente pesante.

Il problema nasce dal fatto che **non viene generato alcun modello**, non viene fatta alcuna astrazione a partire dai dati del training set.

? Si può andare a diminuire la complessità?

Lazy learners

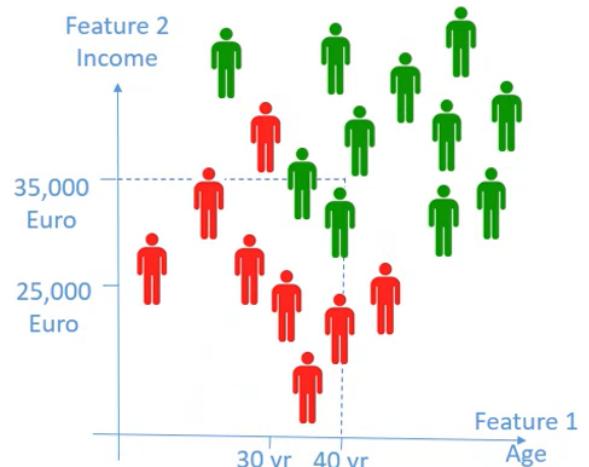
- How could we reduce the computational effort?
 - We can reduce the number of objects in the training set by
 - **sampling the training set**



Essa dipende dal numero di oggetti nel training set, quindi possiamo campionare l'insieme di addestramento:

Lazy learners

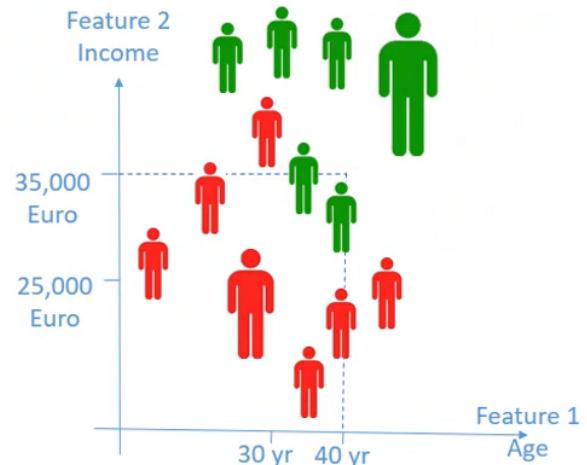
- How could we reduce the computational effort?
 - We can reduce the number of objects in the training set by
 - **sampling the training set**



oppure rimpiazzare il numero di oggetti con qualche rappresentante, si può fare ad esempio usando algoritmi di clustering:

Lazy learners

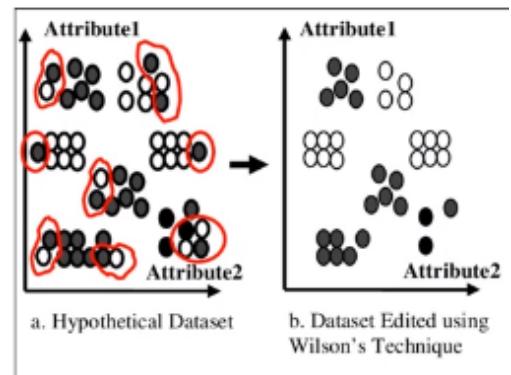
- How could we reduce the computational effort?
 - We can reduce the number of objects in the training set by
 - sampling the training set
 - Replace a number of objects with a representative



Oppure usando dei metodi di editing quindi cercando di rimuovere delle istanze del training set che appaiono in realtà inutili. Un esempio di tecnica di editing è quella che vediamo nello pseudocodice:

Lazy learners

- How could we reduce the computational effort?
 - We can reduce the number of objects in the training set by
 - **Editing Methods: remove training tuples that prove to be useless.**



PREPROCESSING

```

A: For each example  $o_i$  in the dataset  $O$ ,
  1: Find the K-Nearest Neighbors of  $o_i$  in  $O$  (excluding  $o_i$ )
  2: Label  $o_i$  with the class associated with the largest number of
     examples among the K nearest neighbors (breaking ties randomly)
B: Edit Dataset  $O$  by deleting all examples that were misclassified
    in step A.2
  
```

CLASSIFICATION RULE:

Classify new example q using K-NN rule with the edited subset O_r

Per ogni campione nel data set si cercano i K nearest neighbors e si va ad applicare il classificatore KNN quindi si associa a quel campione una classe usando il classificatore KNN. Se la classe associata al classificatore coincide con la classe dell'oggetto, l'oggetto viene preservato. Se la classe predetta dal classificatore non coincide con la classe effettiva dell'oggetto, l'oggetto viene rimosso. La giustificazione di questa rimozione è che se effettivamente quell'oggetto non è classificato correttamente attraverso il KNN vuol dire che probabilmente quel punto è un punto rumoroso all'interno di oggetti appartenenti ad un'altra classe. Questo è il motivo per cui gli oggetti nella immagine sono cerchiati in rosso, sono rimossi in quanto appartengono a zone in cui ci sono oggetti che appartengono all'altra classe e non alla classe associata con l'oggetto stesso. Questa tecnica è benefica sia dal punto di vista di riduzione della complessità che dell'accuratezza del classificatore in quanto va a rimuovere oggetti considerati rumorosi.

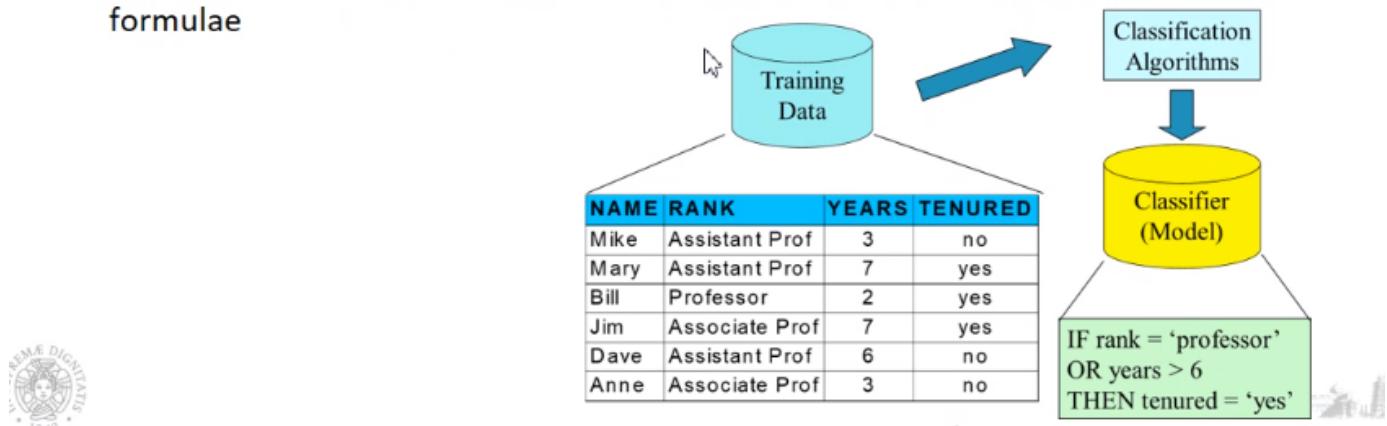
Si può migliorare il KNN passando attraverso la costruzione di un modello. I modelli di classificazione sono costruiti usando un algoritmo di addestramento a partire dai dati di addestramento.

Vediamo un semplice esempio di modello. Dall'insieme di dati che ho nell'insieme di addestramento riesco ad astrarre una singola regola che mi forma il classificatore. Bisogna discriminare i valori "yes" e "no" per la classe "professore di ruolo" sulla base della regola estratta dal training set: ovvero se il "rank" è professor OPPURE gli anni sono maggiori di 6 allora il professore è di ruolo. Questa semplice regola la astraggo dai dati nel training set.

Eager learners – A two Steps Process

First Step

- **Model construction (learning step or training phase):** a classifier is built describing a set of predetermined classes or concepts by exploiting the training set.
 - The model is represented as classification rules, decision trees, or mathematical formulae



Quando mi arriva una nuova istanza, mi basta applicare la regola e posso classificare la nuova istanza o nella classe "Yes" o nella classe "No". Mi costruisco un modello che sarà l'unica cosa usata in fase di classificazione ed ecco perchè questi classificatori vengono chiamati "**Eager learners**".

In questo semplice esempio sono riuscito a produrre la regola semplicemente guardando i pochi dati in possesso, con tanti dati non possiamo generare il modello con questa semplicità quindi serviranno degli algoritmi di apprendimento. Il grande vantaggio che ho però è che durante la fase di

classificazione posso usare il modello risparmiando tempo computazionale.

Eager learners – A two Steps Process

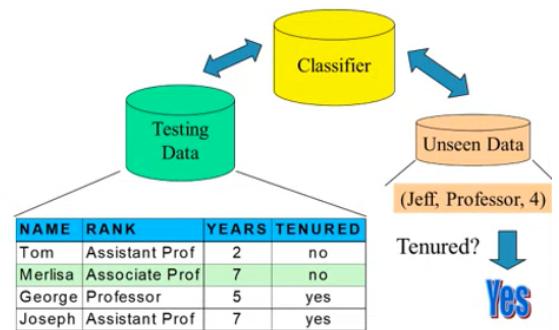
Second Step

- **Model usage:** for classifying future or unknown objects

- Estimate accuracy of the model

- The known label of test sample is compared with the classified result from the model
 - **Accuracy rate** is the percentage of test set samples that are correctly classified by the model (valid if the dataset is not imbalanced)
 - **Test set** is independent of the training set
 - Why do you use a test set (overtraining)?

- If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known



SEM & DIGN.

18/02/2023

Cerchiamo dunque di capire quale sia il passaggio tra il classificatore che abbiamo analizzato nell'ultima lezione in modo dettagliato, il KNN, e il classificatore che analizzeremo invece durante questa lezione. Nel primo caso sfruttiamo direttamente il training set, che ci dà la possibilità di ottenere dei buoni risultati in termini di accuratezza ma ci pone il problema dei tempi di esecuzione direttamente percepibili dall'utente. Il KNN deve calcolare tutte le distanze tra l'oggetto incognito e gli oggetti nell'insieme di addestramento, andare a vedere quali sono gli oggetti più vicini all'oggetto incognito e ragionando con un approccio a maggioranza, assegnare la classe che è quella maggioritaria tra gli oggetti più vicini. Se il numero di oggetti nel training set è elevato ci vorrà un tempo elevato.

Ci sono degli approcci per ridurre il numero di oggetti ma non possiamo ridurlo troppo dato che perderemmo di accuratezza.

! I **lazy learners (studenti pigri)** non costruiscono un modello ma sfruttano direttamente l'insieme di addestramento.

Negli **eager learners (studenti desiderosi di imparare)** INVECE viene costruito un modello. In fase di classificazione ci dimentichiamo del training set ed usiamo solamente il modello per eseguire la classificazione.

Il **modello** è un astrazione dell'insieme di addestramento, dunque ci aspettiamo che tramite il modello i tempi si riducano. Nell'ultima parte della lezione precedente abbiamo visto come sia possibile generare un modello semplice.

La classificazione funzionerà in questo modo:

quando arriverà un oggetto incognito con i valori corrispondenti alle 3 caratteristiche analizzate, il classificatore userà la regola dedotta dal training set per andare ad assegnare l'oggetto alla classe

predetta.

⚠ L'oggetto in verde fa eccezione, in questo caso gli anni sono maggiori di 6 ma la classe ottenuta è NO. Ovviamente avendo costruito il modello sul training set, può accadere che gli oggetti nuovi si differenzino da ciò che abbiamo in forma di conoscenza nel training set. E' normale che durante la fase di classificazione si commettano degli errori.

Eager learners – A two Steps Process

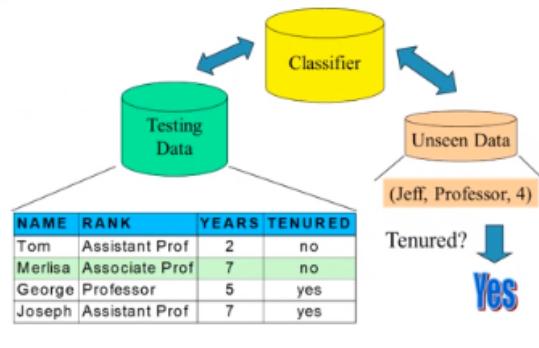
Second Step

- **Model usage:** for classifying future or unknown objects

- **Estimate accuracy of the model**

- The known label of test sample is compared with the classified result from the model
- **Accuracy rate** is the percentage of test set samples that are correctly classified by the model (valid if the dataset is not imbalanced)
- **Test set** is independent of the training set
- Why do you use a test set (overtraining)?

- If the accuracy is acceptable, use the model to classify data tuples whose class labels are not known



Questo insieme di dati (**testing data**) è l'insieme di dati usato per verificare il comportamento del classificatore. Si tratta di un insieme di dati non usato in fase di addestramento ma costruito come un training set.

❓ Perchè non usiamo nuovamente il training set per verificare il classificatore?

Avendo addestrato il classificatore con il training set, ci aspettiamo che su quell'insieme il classificatore si comporti abbastanza bene, vogliamo invece verificare che il classificatore sia in grado di generalizzare e predire. Come vediamo nello schema nella immagine, gli eager learners hanno una fase di produzione del modello e una fase di test.

❓ Come facciamo a valutare un classificatore?

Quello che normalmente usiamo per valutare un classificatore è quella che in letteratura è detta accuracy rate, ovvero la *percentuale di campioni nell'insieme di test che sono classificati correttamente dal modello*. Questa metrica è usata per valutare la bontà dei classificatori. Ma bisogna associare questo rate anche al tipo di dati che vengono processati (se sono bilanciati o meno ad esempio!)...

Supponiamo di avere un problema di classificazione binario (ad esempio la classe dei SI e dei NO). Se le due classi sono **bilanciate** (uguale probabilità di ottenere le 2 classi), l'accuracy rate è ottimo per misurare l'accuratezza. Se non sono bilanciate, avremo un alta probabilità di ottenere una classe e una bassa probabilità di ottenere l'altra. Questo significa che se prendiamo un insieme di test, abbiamo una probabilità ad esempio di trovare istanze della classe minoritaria pari al 5% e un 95% di probabilità di trovare istanze di classe maggioritaria. Perchè l'accuratezza potrebbe non essere una metrica corretta in caso di sbilanciamento?

Supponiamo di aver costruito un classificatore in modo molto "semplice e stupido": **in uscita da sempre la classe maggioritaria**. Nel caso in cui la percentuale di istanze di classe maggioritaria sia del 95% e quella minoritaria sia del 5%, l'accuracy rate è pari al 95%, in quanto sbaglia tutte le istanze di classe minoritaria, ma nel 95% dei casi classifica correttamente. Il 95% effettivamente non è male, ma nel caso specifico il classificatore non sta funzionando bene, in quanto ignora completamente la classe minoritaria. La classe minoritaria in problemi di cybersecurity corrisponde alla classe di attacchi, ci aspettiamo che effettivamente questi problemi siano rappresentati da una minoranza delle istanze a disposizione. Quindi avremo un classificatore che in realtà non riesce a riconoscere quella che per noi è la classe rilevante (pur se minoritaria), anche se l'accuratezza è elevata. **Dunque quando abbiamo a che fare con dati sbilanciati, l'accuratezza non è una metrica da considerare per valutare le prestazioni dei classificatori.**

L'essere capace di generalizzare ci dice anche che il classificatore non soffre del problema dell'**overtraining**, ovvero un classificatore che ha appreso veramente bene l'insieme di addestramento, ma talmente tanto da non riuscire a generalizzare. Quel classificatore non sarà dunque in grado di predire la classe nel caso di oggetti sconosciuti.

? Come identificare l'**overtraining**?

Prendo il training set in fase di addestramento e costruisco il classificatore. Poi faccio 2 valutazioni:

1. classifico il training set, ovvero applico il classificatore per classificare le istanze del training set
2. poi applico il classificatore per classificare le istanze dell'insieme di test
calcolo l'accuratezza nei 2 casi e la confronto. Se le 2 accuratezze sono simili, non avrò overtraining. Se la prima accuratezza è molto più alta della seconda, mi trovo probabilmente in un caso di overtraining.

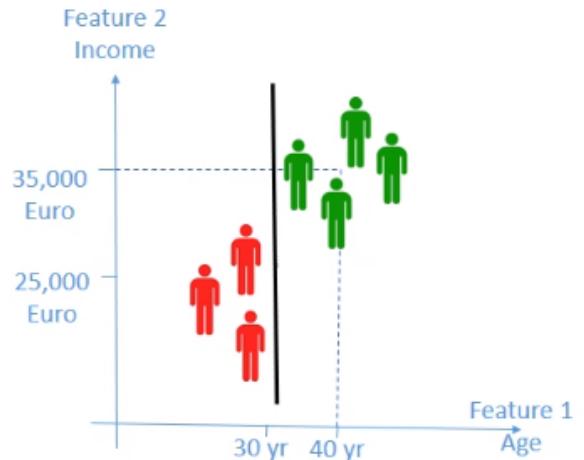
Eager learners

Per capire come questi modelli vengono costruiti andremo a parlare di 2 classificatori molto conosciuti in letteratura.

Negli ultimi anni il problema della classificazione è stato risolto usando la tecnologia, dunque macchine più potenti, con maggiore capacità di memorizzare dati e una grande mole di dati a disposizione. Il primo approccio trasmesso in modo intuitivo è quello seguente:

Eager learners

- Eager learners
- Simple model
 - Separation line



supponiamo di avere un insieme di addestramento rappresentato in figura, dunque la classe di oggetti rossi e quella di oggetti verdi e vogliamo apprendere un modello per classificare. Un modello banale potrebbe essere quello di usare una **linea di separazione per dividere i due oggetti**.

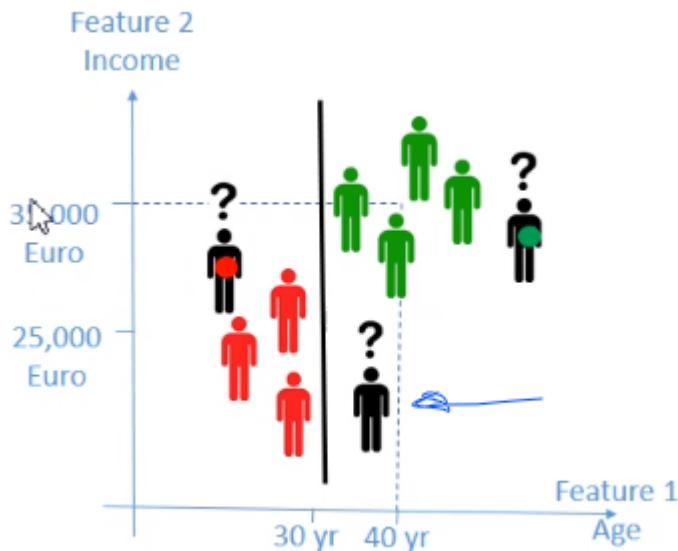
In fase di classificazione se l'oggetto incognito si trova a destra verrà classificato come verde, altrimenti come rosso:



Questo sembra essere un modo di agire corretto ma ha un grosso problema...

! in quanto le 2 classi devono essere effettivamente separabili! Su questo ci torneremo più avanti.

La linea inoltre che abbiamo tracciato è parallela all'asse y che effettivamente separa i 2 gruppi, vediamo cosa potrebbe accadere però se arriva l'oggetto incognito indicato in blu nell'immagine:



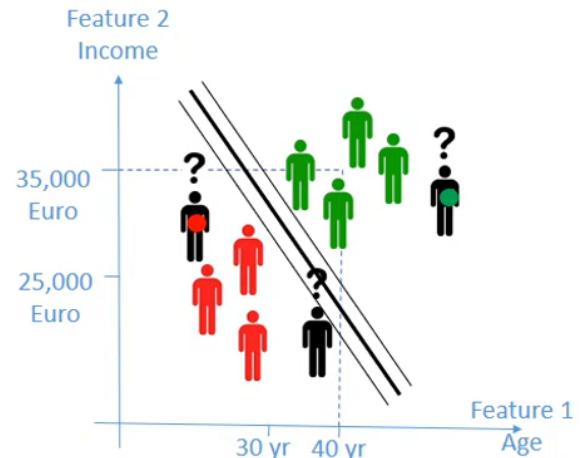
sembra più prossimo alla classe rossa che alla classe verde. **Applicando la strategia di classificazione precedente però verrà classificato come verde!** 🤦

Support Vector Machines

Qui è sbagliato come questa linea di separazione è stata tracciata, infatti mi aspetterei di andare a tracciare la linea di separazione che tende a massimizzare la distanza tra la linea di separazione e gli oggetti di classe rossa e massimizzare la distanza tra la linea di separazione e gli oggetti della classe verde. Ovvero la linea di separazione deve essere tale da massimizzare queste distanze:

Support Vector Machines

- **Support Vector Machine**
 - The shortest distance from a hyperplane to one side of its margin is equal to the shortest distance from the hyperplane to the other side of its margin, where the “sides” of the margin are parallel to the hyperplane.



Questo mi consente di applicare un classificatore che si chiama **Support Vector Machine** in quanto usa come supporto dei vettori di punti che sono più prossimi alla linea di separazione.

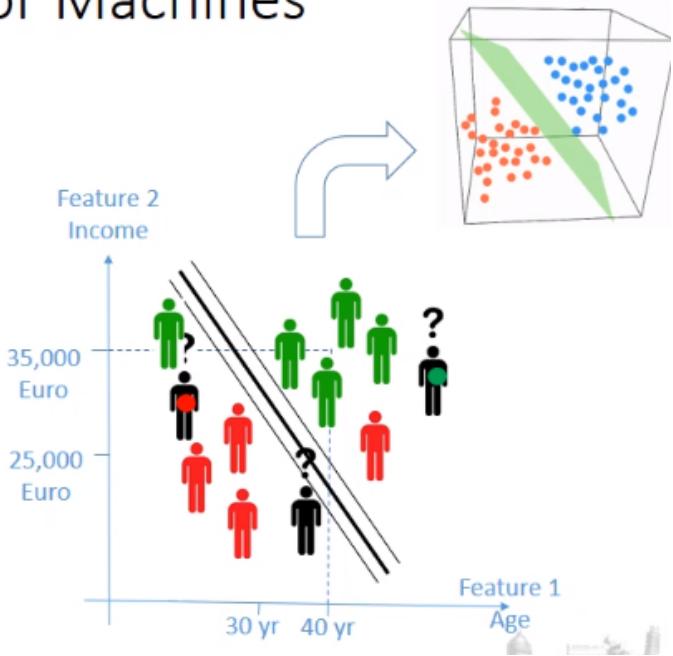
💡 Questa idea mi consente una maggiore generalizzazione ma non risolve il problema menzionato prima!

Ovvero la linea di separazione o l'iperpiano di separazione se parlo di oggetti definiti su più dimensioni funziona se posso assumere che le 2 classi sono linearmente separabili. Se gli oggetti non sono separabili nello spazio attuale, si definisce uno spazio con più dimensioni fittizie e si definisce nello

spazio con più dimensioni un iper-piano separatore:

Support Vector Machines

- **Support Vector Machine**
 - **Easy if the data are linearly separable;** otherwise it uses a nonlinear mapping to transform the original training data into a higher dimension
 - With the new dimension, it searches for the linear optimal separating hyperplane (i.e., “decision boundary”)



visto che stiamo creando un artificio matematico, quello che riusciamo a classificare nello spazio a più dimensioni non è detto che sia classificato correttamente nello spazio originale!.

Il compito del SVM è quello di **trovare l'iperpiano ottimale** in grado di separare le classi nello spazio definito.

Ci aspettiamo di non poter ottenere una separazione al 100%, ma questo classificatore fornisce una buona classificazione.

L'iperpiano di separazione dividerà lo spazio in due semispazi, vediamo dove si posiziona l'oggetto incognito e lo classifichiamo con la classe associata a quel sottospazio. Le SVM si usano per risolvere problemi di classificazione binaria, dunque dove abbiamo solo 2 classi da individuare.

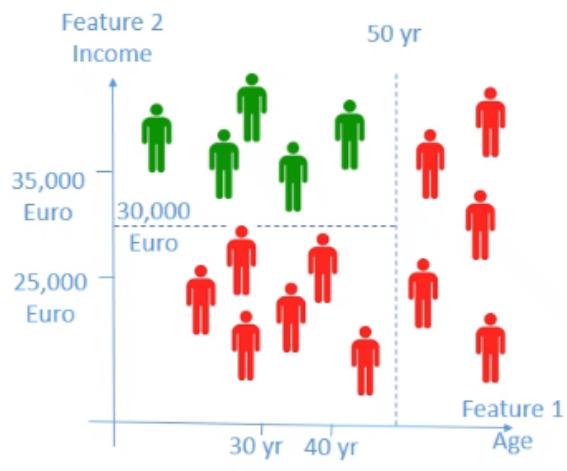
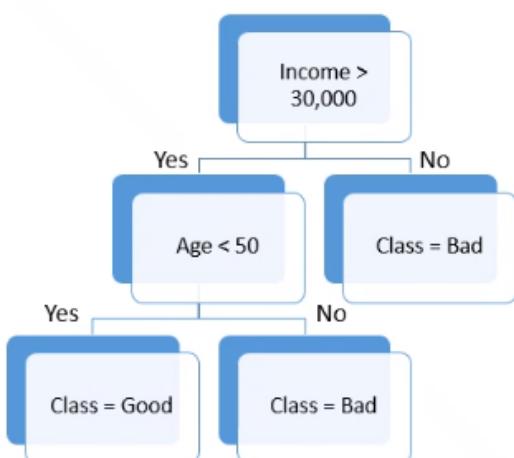
Nel caso di **multiclassi** dovremmo costruire n classificatori con n pari al numero di classi in esame. Ogni classificatore classificherà una classe rispetto a tutte le altre considerate insieme. La SVM può essere usata in caso di multiclassi ma è necessario costruire un numero di classificatori pari al numero delle classi e addestrare questi classificatori in modo tale che possano riconoscere la classe rispetto a tutte le altre.

Decision Trees

Questo classificatore funziona anche per problemi multclasse direttamente. Si chiama albero di decisione in quanto il classificatore è un albero:

Decision Trees

- Decision Tree



in ogni nodo dell'albero si testa il valore di un attributo che descrive gli oggetti e a seconda del valore del test si prende un ramo di uscita. Il *primo nodo* che si chiama *radice dell'albero* testa il valore della variabile `reddito`. Se questo valore è maggiore di 30000 allora si prende il ramo `Yes` che ci porta ad un altro nodo in cui si testerà il valore della variabile `età`. In base al valore, si prenderà il ramo corrispondente e si troverà la classificazione desiderata.

❓ Come possiamo strutturare questo albero?

Nel caso dell'insieme di addestramento rappresentato sulla destra è facile da ottenere. Tutti i punti di classe verde corrispondenti alla classe `Good` sono all'interno del semipiano contraddistinto dall'avere valori di `reddito` maggiori di 30000 e valori di `età` inferiori a 50. Infatti questo semipiano è proprio rappresentato da un primo nodo che verifica il valore di `reddito` maggiore di 30000 e un secondo nodo in cascata che verifica l'`età` minore di 50.

Questo albero di decisione è stato possibile costruirlo avendo a che fare solo con 2 variabili. **Se abbiamo tante variabili, abbiamo bisogno di qualche algoritmo che ci consenta di decidere la struttura dell'albero di decisione.** Qui abbiamo usato `reddito` come attributo su cui fare il test della radice dell'albero. Ma chi ci dice quale attributo usare come primo attributo? Come scegliere la caratteristica radice dell'albero e le caratteristiche degli altri nodi? La scelta dell'attributo ottimale ci consente di avere un numero ridotto di nodi e un'ottima accuratezza. Per apprendere un albero di decisione dobbiamo poter avere una strategia per scegliere gli attributi su cui andremo a fare i test nei singoli nodi e andiamo a scegliere questi attributi cercando di raggiungere una **partizione pura**. Nell'esempio di prima la nostra divisione in semipiani che avevamo fatto era stata fatta considerando di andare a raggruppare in questi semipiani degli oggetti appartenenti alla stessa classe. Quando si riesce a realizzare questa divisione stiamo effettivamente costruendo partizioni pure.

Dunque una partizione è detta **pura** se contiene solo oggetti appartenenti alla stessa classe.

❓ Perchè vogliamo individuare partizioni pure? I nuovi oggetti che arriveranno durante la classificazione si posizioneranno in semipiani in cui avremo la totalità di istanze appartenenti ad una classe dunque ci aspettiamo che questo ci consenta di classificare correttamente l'istanza incognita. Abbiamo selezionato `reddito` maggiore di 30000 che ci consente di isolare una partizione pura se `reddito` è minore di 30000, dall'altro se `reddito` è maggiore di 30000 abbiamo usato l'`età` per ottenere un ulteriore partizione pura.

Quindi quando vado a costruire l'albero di decisione devo trovare una strategia che mi consenta di scegliere gli attributi ad ogni nodo con l'idea di andare verso una partizione nuova. Ovviamente però questa strategia deve avere anche delle condizioni di terminazione, cioè devo potermi fermare nella generazione dei nodi quando si verificano le condizioni che ora vedremo.

Decision Tree Learning

- **Decision Tree Learning**

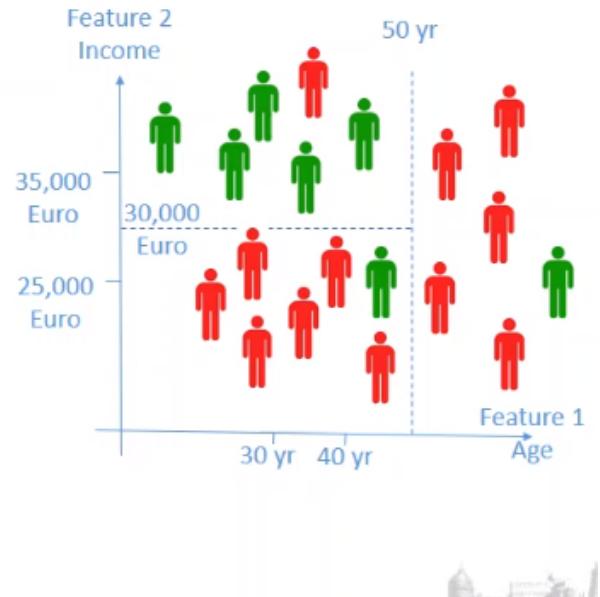
- Choice of the decision variable on the nodes
 - Search for pure partition
- Termination condition

- **Decision Tree Learning (design parameters)**

- Depth of the tree
- Pre-pruning versus post-pruning

- **Decision Tree Learning (Characteristics)**

- Simplicity
- Interpretability



Quando attivo un albero di decisione non devo fissare dei parametri.

❗ Se l'albero è **tropppo profondo** ho il rischio di isolare dei sottopiani molto specifici, che mi selezionano delle piccole parti dello spazio, ovvero posso avere un **overtraining**, si specializza troppo, e quando riceve dei punti differenti predice male la classe in quanto non riesce a generalizzare. Se ho il rischio di overtraining devo lavorare sull'algoritmo di apprendimento affinchè l'albero non diventi troppo profondo oppure devo procedere a fare un post-pruning ovvero una potatura dell'albero.

- **Pre-pruning** : cerco di fermare la crescita dell'albero in modo tale che non diventi troppo profondo, per esempio posso richiedere che per generare un nodo nuovo nell'albero di decisione devo avere almeno un numero fissato di oggetti sul nodo precedente
- **Post-pruning** : posso pensare di tagliare parti del sottoalbero in modo tale da ridurre la profondità dell'albero ed aumentare la sua capacità di generalizzazione

Gli alberi di decisione sono interpretabili in quanto possono essere espressi in forma di regole.

❓ Come funziona un algoritmo di apprendimento di un albero?

Decision Tree Learning

- **Basic algorithm is greedy**

Follows the problem solving heuristic of making the locally optimal choice at each stage with the hope of finding the global optimum

- Tree is constructed in a **top-down recursive divide-and-conquer manner**
- At start, all the training examples are at the root
- Attributes are categorical (if continuous-valued, they are discretized in advance)
- Examples are partitioned recursively based on selected attributes
- Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

L'algoritmo segue un approccio **euristico** cercando di fare la scelta ottimale dell'attributo che vogliamo usare su ogni singolo nodo. L'algoritmo cerca di selezionare in modo ottimale ma locale l'attributo.

Questi sono approcci **greedy** in quanto fanno un ottimizzazione **locale** ad ogni passo dell'algoritmo.

L'albero è costruito in modo ricorsivo partendo dalla radice.

Si parte da tutti gli insiemi di addestramento contenuti nella radice dell'albero, dunque sceglio il primo attributo ottimale (introducendo una qualche metrica che ci consente di prendere questa scelta), ovvero quello che ci consente di dividere il training set in sottoinsiemi ognuno relativo ad un valore dell'attributo che abbiamo scelto. Nell'esempio precedente tutte le istanze maggiori o superiori al valore di **reddito** 30000. Se questi 2 sottoinsiemi dell'insieme di addestramento raggiungono una partizione pura, possiamo fermarci in quanto basta quel singolo attributo per classificare. Altrimenti dobbiamo ricercare un secondo attributo. Ecco perchè l'approccio è **euristico e ricorsivo**.

Il nostro target è avere delle partizioni pure ma dobbiamo introdurre delle regole per fermare la ricorsione:

Decision Tree Learning

- **Conditions for stopping partitioning**

- All samples for a given node belong to the same class
 - There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
 - There are no samples left
-
- Strategy for selecting the attributes: to create partitions at each branch as pure as possible.
 - A partition is pure whether all the tuples in it belong to the same class

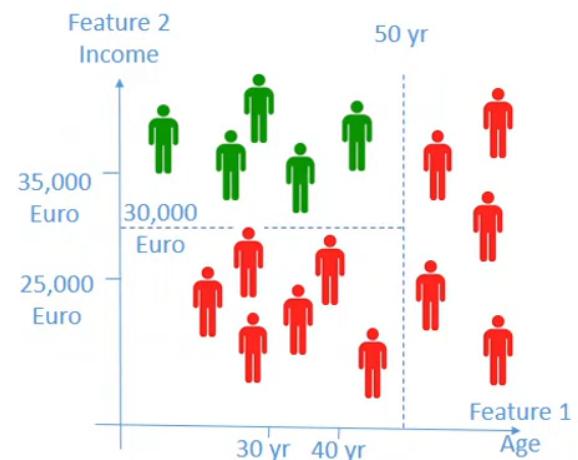
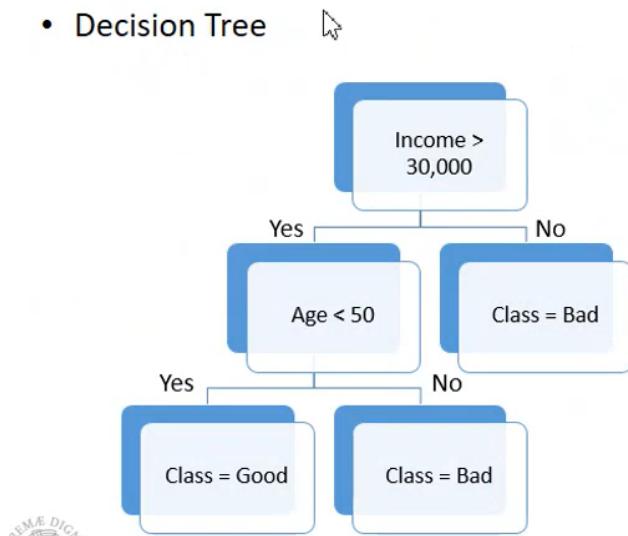
- Se tutti i campioni per un dato nodo sono in una partizione pura ovvero appartengono alla stessa classe allora possiamo fermare la ricorsione
- Abbiamo sfruttato tutti gli attributi ma non siamo riusciti a trovare la partizione pura, dunque dobbiamo fermarci con un sottoinsieme di dati che non formano una partizione pura. Dunque il nodo a cui ci siamo fermati viene considerata una **foglia** e sarà identificato con la classe corrispondente alla maggioranza delle istanze contenute nel nodo
- Non abbiamo più esempi da partizionare

? Come facciamo in modo matematico ad andare verso partizioni pure?

Ci serve una metrica per decidere tra tutti gli attributi considerati quello che mi consente di avere una partizione pura. Tornando all'esempio banale analizzato in precedenza:

Decision Trees

- Decision Tree



when I choose `reddito` I don't create a pure partition because in reality it's true that if $\text{reddito} \leq 30000$ the partition is pure, but if $\text{reddito} \geq 30000$ the partition we get is not pure! So how do I choose `reddito` instead of `età`? **come decido quale attributo mi porta di più verso una partizione pura?** Or in other words, how do I choose an attribute that leads me to more pure partitions? From a mathematical point of view, how do I choose an attribute that has the largest information gain?

Per poter trovare questa metrica però dobbiamo necessariamente introdurre il concetto di **entropia**:

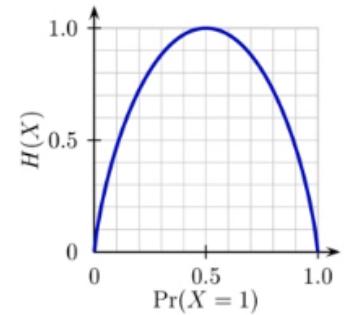
Decision Tree Learning

Choice of the decision attribute

- Let D a training set of class-labelled objects
- Suppose that the class label attribute has m distinct values defining m distinct classes, C_i (for $i=1,..,m$).
- Select the attribute with the highest information gain
- Let p_i be the probability that an arbitrary tuple in D belongs to class C_i , estimated by $|C_{i,D}|/|D|$
- Expected information (entropy) needed to classify a tuple in D :

$$Info(D) = - \sum_{i=1}^m p_i \log_2(p_i)$$

- Observe: p_i is the probability of class i in D



In questo caso nell'immagine abbiamo m classi distinte e vogliamo trovare l'attributo ottimale. Partiamo dall'insieme di addestramento e calcoliamo l'entropia di informazione definita dalla formula nella immagine.

p_i è la probabilità che un'istanza nel training set appartenga alla classe i -esima e può essere stimata andando a contare gli elementi che appartengono alla classe i -esima nel training set e dividendo per il numero totale di istanze.

Supponiamo di avere una partizione pura dunque che il training set abbia istanze solo appartenenti a una classe, ad esempio alla classe i -esima, la probabilità per questa classe è uguale a 1 e la p_i per le altre classi è 0. Dunque calcolando l'entropia avremmo (supponendo che la classe i -esima sia la prima e che $\log_2(0) = 0$):

$$Info(D) = -1 \log_2(1) - \sum_{i=2}^m p_i \log_2(p_i) = 0 - \sum_{i=2}^m p_i \log_2(p_i) = 0$$

Abbiamo in altre parole che **quando la partizione è pura** il valore dell'**entropia** è minimo e pari a **0**.

Se tutte le classi hanno uguale probabilità, il valore dell'**entropia** è **massimo**. Facendo riferimento al grafico a campana in alto ci riferiamo al caso di un problema a 2 classi, ovvero se la probabilità di una classe aumenta diminuisce quella dell'altra. Il valore massimo di entropia $H(X)$ si ha quando la probabilità è 0.5, ovvero quando abbiamo uguale probabilità di avere le 2 classi. **La entropia massima si ha con uguale probabilità di avere istanze appartenenti alle classi.**

Dunque quando l'entropia è vicina a 0 siamo in presenza di partizione pura, se l'entropia è molto alta siamo in presenza di una partizione in cui le istanze appartengono con equa probabilità alle differenti classi.

Vogliamo dunque selezionare l'attributo che ci consente di ridurre il più possibile l'entropia, ovvero selezionare quell'attributo che ci consente di avere il **guadagno** maggiore in termini di informazione che ci è dato dalla differenza dall'entropia iniziale **Info(D)** e quella che qui è definita come **Info_A(D)** ovvero l'entropia che troviamo usando l'attributo specifico:

Decision Tree Learning

Choice of the decision attribute

- Suppose that we split D into v partitions $\{D_1, \dots, D_v\}$ on some attribute A having v distinct values $\{a_1, \dots, a_v\}$
- Information needed (after using A to split D into v partitions) to classify D:

$$Info_A(D) = \sum_{j=1}^v \frac{|D_j|}{|D|} \times Info(D_j)$$

Cardinality of the set of objects in D with value for A equal to a_j

where D_j contains all the tuples in D with value a_j for A

$$Gain(A) = Info(D) - Info_A(D)$$



Information gained by branching on attribute A



Andiamo a dividere l'insieme di addestramento in v sottoinsiemi dove ogni sottoinsieme corrisponde a uno specifico valore dell'attributo A che stiamo considerando. Dunque se l'attributo A ha 3 valori: basso, medio e alto, prenderemo tutte le istanze che nel training set hanno valore basso, tutte quelle con valore medio e tutte quelle con valore alto e partizioniamo l'insieme di addestramento iniziale con questi sottoinsiemi. Per ogni sottoinsieme andiamo a calcolare l'entropia del sottoinsieme $Info(D_j)$ e la moltiplichiamo per un **peso** che è dato da:

$$\frac{|D_j|}{|D|} = \frac{\text{cardinalità dell'insieme degli oggetti in } D \text{ con valore di } A \text{ uguale ad } A_j}{\text{cardinalità totale dell'insieme di addestramento}}$$

Questo rapporto ci dà la probabilità che un'istanza abbia un valore A_j per un attributo A.

Sommiamo questi prodotti per tutti i possibili valori di A e otteniamo $Info_A(D)$.

Avendo diviso l'insieme di addestramento in v valori ed avendo che ogni sottinsieme trovato sia puro, $Info(D_j)$ sarebbe per ogni valore A_j uguale a 0, portandoci ad avere $Info_A(D) = 0$. Dunque avremmo il guadagno massimo. Ovvero se selezionassimo un attributo che una volta partizionato l'insieme di addestramento mi restituirebbe solo partizioni pure, avremmo il guadagno massimo e quindi il risultato desiderato. Difficilmente questo accadrà nella pratica, ma noi vorremo selezionare tra tutti gli attributi quello che ci consente di avere il guadagno maggiore ovvero che ci porta più velocemente verso la partizione pura.

Esempio:

Decision Tree Learning

■ Class P: buys_computer = "yes"

■ Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14} \log_2 \left(\frac{9}{14}\right) - \frac{5}{14} \log_2 \left(\frac{5}{14}\right) = 0.940$$

age	p _i	n _i	I(p _i , n _i)
<=30	2	3	0.971
31...40	4	0	0
>40	3	2	0.971

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31...40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31...40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31...40	medium	no	excellent	yes
31...40	high	yes	fair	yes
>40	medium	no	excellent	no

In questo esempio le istanze hanno 4 attributi (age,income,student,credit_rating) con possibili valori e abbiamo due classi, `buys_computer` con valori "no" e "yes".

Come prima cosa dobbiamo calcolare l'entropia sull'insieme di addestramento, andando ad usare la formula definita all'inizio. Ci serve calcolare la probabilità che ci siano istanze di classe "yes" nel training set e la probabilità che ci siano istanze di classe "no" nel training set. Per calcolare questa probabilità andiamo a contare quante istanze all'interno della classe "yes" ci sono nel training set e quante invece della classe "no" ci sono nel training set.

Per classe "yes" abbiamo 9 istanze dunque probabilità 9/14, mentre per la classe "no" ne abbiamo 5, dunque 5/14.

Calcoliamo l'entropia che sarà 0.940. Questo valore è molto alto in quanto abbiamo sia istanze di classe no che istanze di classe yes e il numero di queste istanze è abbastanza bilanciato. A questo punto dobbiamo decidere quale attributo usare nel nodo radice, ovvero il primo attributo che formerà l'albero di decisione. Per farlo dobbiamo testare tutti gli attributi che abbiamo a disposizione per decidere quale attributo ci garantisce il guadagno di informazione maggiore. Dobbiamo calcolare $Info_A(D)$ sostituendo ad A di volta in volta `income`, `student`, `age` e `credit_rating`. Vediamo il calcolo quando valutiamo `age`, per gli altri attributi il calcolo sarà simile.

Dobbiamo calcolare la Info per ogni possibile valore di age, il primo che consideriamo è `age` ≤ 30 .

Quante istanze abbiamo con valori ≤ 30 ? 5. Dunque la probabilità sarà 5/14. Queste 5 istanze formano una partizione di istanze con valore di age ≤ 30 . L'entropia di questa partizione è rappresentata con $I(2,3)$. Se andiamo a vedere le istanze caratterizzate da valore di `age` ≤ 30 , abbiamo 3 istanze con valore di classe "no" e 2 istanze con valore di classe "yes". La probabilità di avere classe "yes" è 2/5 mentre quella di avere classe "no" è 3/5. Stesso calcolo bisogna ripetere per gli altri possibili valori di age (tra 31 e 40 e maggiore di 40). Il `Gain(age)` risulta essere quello più alto dunque decidiamo di usare nel primo nodo dell'albero la variabile `age`.

L'albero di decisione avrà nel nodo radice l'attributo age e il test sui possibili valori di age ci porterà a

$$\begin{aligned} Info_{age}(D) &= \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) \\ &\quad + \frac{5}{14} I(3,2) = 0.694 \end{aligned}$$

$\frac{5}{14} I(2,3)$ means "age ≤ 30 " has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

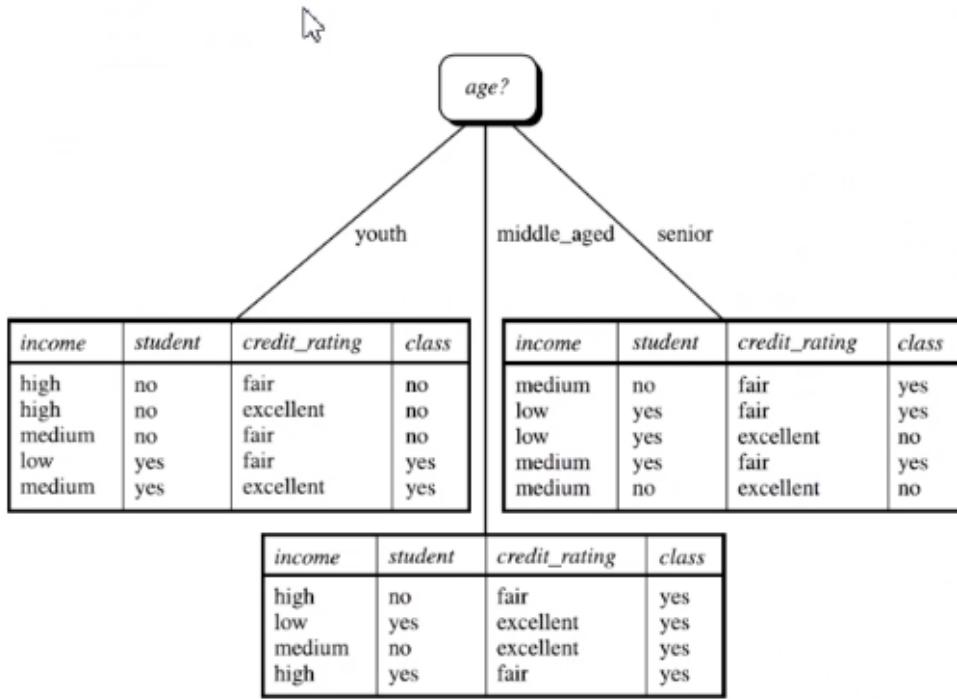
$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit_rating) = 0.048$$

ottere una partizione pura:

Decision Tree Learning



Una delle condizioni di terminazione era proprio l'ottenimento di una partizione pura, dunque non c'è più bisogno di dividere questo sottoinsieme dell'insieme di addestramento in base ad altri attributi, ma possiamo direttamente fermarci e andare ad associare a questo nodo la classe "yes".

❓ Cosa succede per gli altri valori?

Qui siamo arrivati ad avere una partizione che non è pura, possiamo allora riapplicare quello già fatto sostituendo a D non più il training set ma un sottoinsieme costituito dalle istanze con valore di age pari a "youth". La stessa cosa per age senior. Cerchiamo di risuddividere il sottoinsieme youth e senior sfruttando gli altri attributi ma procedendo con lo stesso approccio. Ovvero andando a vedere quale è l'attributo tra quelli rimanenti che ci fornisce il miglior guadagno.

Alla fine arriveremo ad avere un albero di decisione in grado di classificare tutti gli oggetti:

Decision Tree Learning Scheme

Algorithm: `Generate_decision_tree`. Generate a decision tree from the training tuples of data partition, D .

Input:

- Data partition, D , which is a set of training tuples and their associated class labels;
- $attribute_list$, the set of candidate attributes;
- $Attribute_selection_method$, a procedure to determine the splitting criterion that “best” partitions the data tuples into individual classes. This criterion consists of a $splitting_attribute$ and, possibly, either a $split-point$ or $splitting_subset$.

Output: A decision tree.

NCCS

Decision Tree Learning Scheme

Method:

- (1) create a node N ;
- (2) if tuples in D are all of the same class, C , then
 - (3) return N as a leaf node labeled with the class C ;
 - (4) if $attribute_list$ is empty then
 - (5) return N as a leaf node labeled with the majority class in D ; // majority voting
 - (6) apply `Attribute_selection_method`($D, attribute_list$) to find the “best” $splitting_criterion$;
 - (7) label node N with $splitting_criterion$;
 - (8) if $splitting_attribute$ is discrete-valued and
 - multiway splits allowed then // not restricted to binary trees
 - (9) $attribute_list \leftarrow attribute_list - splitting_attribute$; // remove $splitting_attribute$

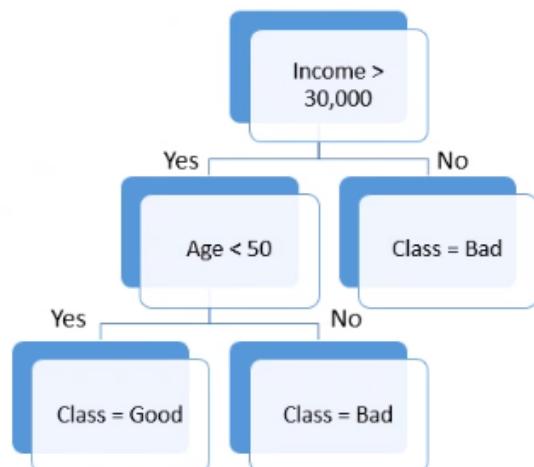
L'albero di decisione generato è **interpretabile** in quanto l'albero di decisione può essere rappresentato come insieme di regole:



Decision Tree: Interpretability

- **Interpretability**

- IF Income > 30,000 AND Age < 50 THEN
Class = Good
- IF Income > 30,000 AND Age >= 50
THEN Class = Bad
- IF Income <= 30,000 THEN Class = Bad



M&E DIGITAL

basta infatti percorrere l'albero dalla radice ad ogni foglia. Ovvero quando arriva un oggetto da classificare, l'oggetto incognito è caratterizzato dai suoi valori per gli attributi. Partendo dalla radice, si prende il valore dell'attributo dell'oggetto incognito e vedo se devo andare nel ramo di destra o sinistra. Ogni percorso dalla radice alla foglia è una regola, ad esempio nell'albero della immagine, se partiamo dalla radice `income` maggiore di 30000 fino ad arrivare alla classe `good` generiamo una prima regola.

⚠ Il problema potrebbe essere quello di avere troppi rami e nodi e quindi avere sottoparti dello spazio troppo piccole e quindi che l'albero non sia in grado di generalizzare. Il primo approccio è il **pre-pruning** ovvero decido di fermare la generazione dell'albero anche se non ho raggiunto le condizioni di terminazione prima illustrate:

Overfitting and Tree Pruning

Overfitting: An induced tree may overfit the training data

- Too many branches, some may reflect anomalies due to noise or outliers
- Poor accuracy for unseen samples

Two approaches to avoid overfitting

- **Prepruning:** Halt tree construction early - do not split a node if this would result in the goodness measure (typically, information gain, Gini index, etc.) falling below a threshold
 - Difficult to choose an appropriate threshold

Oppure genero l'albero finché posso e poi provo a vedere se posso tagliare qualcosa andando a

migliorare l'accuratezza (tramite il **post-pruning**):

Overfitting and Tree Pruning

- **Postpruning:** Remove branches from a “fully grown” tree — get a sequence of progressively pruned trees
 - Removes subtrees from a “fully grown” tree. A subtree at a given node is pruned by removing its branches and replacing it with a leaf. The leaf is labeled with the most frequent class among the subtree being replaced.
 - Use a set of data different from the training data to decide which is the “best pruned tree”

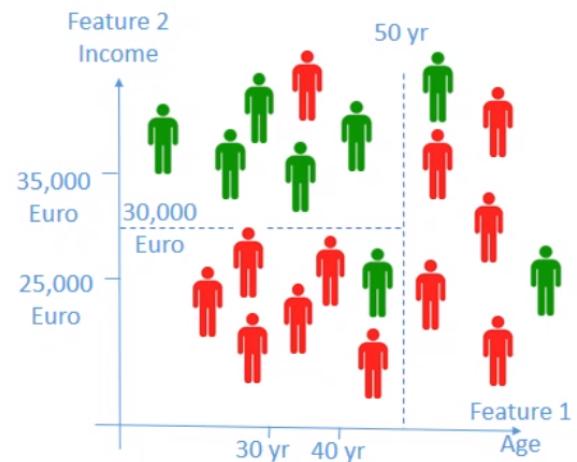
Quello che normalmente accade è quello di generare un albero molto completo e dettagliato e poi procedere con un post-pruning per cercare di rendere l'albero con meno foglie e rami e quindi più adatto alla generalizzazione.

Nella prossima lezione vedremo come valutare le performance dei classificatori e quindi capire come un algoritmo esistente è migliore rispetto ad un altro.

11/03/2023

Performance Evaluation

- **How is the performance of a classifier evaluated?**
 - Use of a test set: set of labelled objects not used in the training set
 - Most intuitive metrics
 - Classification accuracy
- $$\frac{\text{Number of correctly classified objects}}{\text{Total number of objects}}$$



❓ Dati alcuni classificatori che posso aver implementato, come posso decidere il classificatore migliore per il mio problema?

Per poterlo fare è necessario introdurre delle metriche e misurarle per poter dire quale soluzione è migliore.

Abbiamo detto che per valutare i classificatori utilizzo un insieme di test.

E' fatto come l'insieme che utilizzo per il training ma ovviamente dato che voglio testare la capacità di generalizzazione del classificatore **l'insieme di test non deve essere stato usato per l'addestramento**.

Tipicamente accade che ho un insieme di dati etichettato e uso parte di questo insieme di dati per il training e parte per il test. Quello che mi viene dato è un insieme di dati progettato. Ovviamente quando andiamo a provare il nostro classificatore sul test set, non possiamo aspettarci un accuratezza del 100% da parte del classificatore. Quello che ci viene in mente per misurare la classificazione è usare il concetto di **accuratezza**.

L'accuratezza è un rapporto tra il numero di oggetti classificati correttamente e il numero totale di oggetti. Dunque si tratta della percentuale di oggetti nel test set classificati correttamente.

⚠ In realtà bisogna fare attenzione su cosa effettivamente possiamo avere come risultati dell'accuratezza. ⚠

Supponiamo di avere un problema a 2 classi, **rossa** e **verde**. Abbiamo realizzato un classificatore e questo classificatore ha un accuratezza del 95%.

❓ Se abbiamo questa informazione siamo "contenti" oppure no?

Supponiamo che il problema sia **fortemente sbilanciato**, ovvero ho una classe che è quella maggioritaria che ha il 95% di istanze, e l'altra classe che è quella minoritaria che ha solo il 5% delle istanze. Facendo un classificatore parecchio stupido che rispondesse sempre con la classe maggioritaria, otterrei una accuratezza del 95%.

Questo ci porta a concludere che in caso di problemi in cui abbiamo dataset sbilanciati l'accuratezza non ci soddisfa come metrica in quanto mette insieme le 2 classi senza farci distinzioni, ovvero andando a contare il numero di oggetti classificati correttamente non distingue tra le 2 classi e dunque usando solo l'accuratezza non riusciamo a capire se esiste una differenza tra le 2 classi. Il problema possiamo risolverlo andando oltre l'accuratezza e andando a usare la **MATRICE DI CONFUSIONE**.

Per semplicità supponiamo di avere solo 2 classi ma può essere generalizzata ad m classi.

Performance Evaluation

- Confusion matrix

Actual class\Predicted class	C_1	$\neg C_1$
C_1	True Positives (TP)	False Negatives (FN)
$\neg C_1$	False Positives (FP)	True Negatives (TN)

Example of Confusion Matrix:

Actual class\Predicted class	buy_computer = yes	buy_computer = no	Total
buy_computer = yes	6954	46	7000
buy_computer = no	412	2588	3000
Total	7366	2634	10000

- o predette come appartenenti alla stessa classe
- o predette come appartenenti ad un'altra.

Andando a vedere dunque la cella indicata come **true positive**, questo corrisponde alle istanze che appartengono alla classe C_1 e sono state effettivamente classificate come appartenenti alla classe C_1 .

Nella matrice quello indicato come **false negatives** sono istanze appartenenti alla classe C_1 ma classificate erroneamente come appartenenti alla classe $\neg C_1$, ecco perché sono dei false negative.

I **false positive** invece sono istanze appartenenti a $\neg C_1$ che sono erroneamente classificate come non appartenenti alla classe $\neg C_1$.

Infine i **true negative** sono istanze appartenenti alla classe $\neg C_1$ classificate correttamente come appartenenti alla classe $\neg C_1$.

Qui abbiamo un esempio di matrice di confusione. La matrice di confusione ideale risultante da una classificazione, dovrebbe essere una matrice **diagonale**, dovrebbe avere valori diversi da 0 lungo la diagonale principale. Dovrebbe essere un classificatore ideale. Nella realtà avviene che le altre celle non diagonali in realtà hanno valori differenti da 0. La matrice ci dice esattamente per ogni istanza appartenente a una classe per cosa in realtà viene confusa. Sappiamo che istanze appartenenti alla classe **Yes** sono confuse con la classe **No**. Ad esempio ci dice che 412 istanze appartenente alla classe **No** sono confuse per la classe **Yes**. Dunque ci dà una informazione abbastanza completa di cosa accade. Guardando la matrice di confusione ci saremmo accorti subito che il classificatore non funzionava correttamente, nel caso introdotto all'inizio del 95%.

L'accuratezza non ci soddisfa, la matrice di confusione ci dà una bella informazione ovvero ci dice come il classificatore si sta comportando però voglio confrontare differenti classificatori ovvero confrontare diverse matrici, il quale però non è banale, in quanto una matrice potrebbe un valore di true positive maggiore e un valore di true negative inferiore, dunque non ragionando su un singolo valore, diventa difficile effettuare la valutazione dei classificatori. Si può cercare di rappresentare la matrice di confusione usando delle formule riassuntive dell'informazione all'interno della matrice di confusione.

Performance Evaluation

- Classification accuracy (balanced dataset)

$$\frac{TP + TN}{ALL}$$

- Imbalanced dataset

- Sensitivity (TPR)

$$\frac{TP}{P}$$

- Specificity

$$\frac{TN}{N}$$

- Precision

$$\frac{TP}{TP + FP}$$

- Recall

$$\frac{TP}{P}$$



- Precision:** è il rapporto tra i true positive e i true positive più i false positive. Esprime quanti positivi considerati come positivi dal nostro classificatore sono realmente positivi. Esprime il rapporto tra quelle che sono realmente istanze positive rispetto a tutte quelle classificate come positive dal classificatore (sia nel caso in cui classifica bene che nel caso in cui classifica male). **Ci dice se ci sono istanze negative che sono state erroneamente classificate come positive.**
- Recall:** sono i reali positivi, ovvero effettivamente le istanze effettivamente positive, dunque la recall ci dice quante delle istanze effettivamente positive il nostro classificatore è in grado di riconoscere. Ovviamente se la recall è 1 vuol dire che tutte le istanze positive sono state classificate correttamente. Avere recall uguale a 1 non ci dice quante istanze il nostro classificatore sta classificando come positive. Questa informazione ci è data dalla precision.

Dobbiamo aspettarci che entrambe siano più possibile vicine a 1.

Torniamo al problema iniziale, ovvero confrontare i classificatori. Usando precision e recall potremmo avere un classificatore migliore in termini di precisione e un altro in termini di recall. Risolviamo il problema rimettendo insieme precision e recall ovvero tramite la **F-measure**:

Performance Evaluation

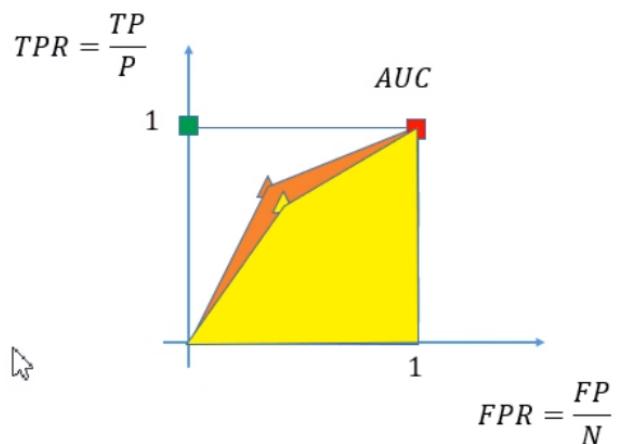
- Classification accuracy (balanced dataset)

$$\frac{TP + TN}{ALL}$$

- Imbalanced dataset

• Sensitivity (TPR)
$\frac{TP}{P}$
• Specificity
$\frac{TN}{N}$

• Precision
$\frac{TP}{TP + FP}$
• Recall
$\frac{TP}{P}$



F-mesure:

$$F_\beta = \frac{(1 + \beta^2) \times precision \times recall}{\beta^2 \times precision + recall}$$

ovvero una media geometrica di precision e recall. Il parametro β serve a pesare maggiormente la precision e recall.

Un altro modo per trovare un unico valore per confrontare i classificatori si può usare la **AUC (Area Under Curve)** ovvero l'area sotto la curva.

Per definire la AUC dobbiamo fare riferimento al piano Rock i cui assi sono rappresentati da TPR e FPR.

Il FPR (false positive rate) è il rapporto tra i falsi positivi (FP) ed N ovvero il rapporto tra quelli che il nostro classificatore considera come positivi anche se sono istanze di classe negativo e il numero totale dei negativi.

Il TRP (true positive rate) è il rapporto tra i true positive (TP) e P che equivale alla Recall.

💡 Un classificatore ideale avrà TPR pari a 1 ovvero tutti i positivi vengono classificati correttamente. L'FPR invece vorremmo che FP sia uguale a 0 ovvero non vorremmo avere falsi positivi. Dunque FPR sarà uguale a 0.

Il nostro classificatore ideale corrisponderebbe al quadratino verde nel piano Rock. Ovviamente quando implementiamo il classificatore troveremo ad esempio come risultato della classificazione, i triangolini in arancione e giallo ovvero un possibile classificatore che ha un valore di TPR diverso da 1 e un valore di FPR diverso da 0. Se connettiamo il valore ottenuto dal classificatore in termini di FPR e TPR ai punti (0,0) e (1,1) otteniamo delle curve che sono curve rappresentative del nostro classificatore.

Possiamo calcolare l'area sotto la curva, che corrisponde all'area in giallo per il classificatore giallo e in arancione per quello arancione (anche se coperta da quella gialla e poco visibile nell'immagine).

Dato che il nostro classificatore ideale è quello disegnato in verde, più è grande l'area sotto la curva più il classificatore che consideriamo si avvicina a quello ideale.

Calcolare la AUC ci porta a introdurre una metrica unica per valutare un classificatore.

Non è che abbiamo lo stesso rischio con l'accuratezza?

In realtà NO perchè l'accuratezza non prende in considerazione il fatto che stiamo riconoscendo correttamente istanze di classe positive o negative, mentre nella AUC questo è preso in considerazione in quanto quando calcolo l'area della curva uso TPR e FPR i quali prendono in considerazione i falsi positivi e i positivi reali. Se avessi il mio classificatore con un alto numero di falsi positivi avrei che il classificatore sarebbe molto spostato verso il quadratino rosso dunque significherebbe l'AUC molto più piccola.

L'AUC (così come la F-measure) sono una metrica che posso usare per confrontare 2 classificatori anche nel caso di insiemi sbilanciati.

La matrice di confusione può essere introdotta per M classi considerando istanze appartenenti a una classe e vedo come quelle istanze sono classificate correttamente come appartenenti alla classe oppure non correttamente come appartenenti alle altre classi. Se considero ogni riga, ciascuna di esse corrisponde a una istanza di una classe e mi fa vedere quelle istanze di una classe come sono classificate dal classificatore nelle varie classi. Posso calcolare anche il TP, TN, FP, FN per ogni classe considerando le istanze appartenenti alla classe ad esempio la classe C_i come positive e tutte le altre istanze, ovvero non appartenenti alla classe C_i come negative.

University of Tiso

Francesco Marcellon

Performance Evaluation Multi-class problems

- Confusion matrix

		Estimated Emotion							
		Anger	Boredom	Disgust	Fear	Happiness	Sadness	Neutral	Emotion Recog. Rate
True Emotion	Anger	19	0	2	0	3	0	0	79.2%
	Boredom	1	8	1	1	0	1	7	42.1%
	Disgust	0	1	6	0	1	0	3	54.5%
	Fear	1	3	2	7	2	0	1	43.8%
	Happiness	3	0	3	2	5	0	2	33.3%
	Sadness	0	0	0	0	0	14	0	100.0%
	Neutral	0	5	1	0	0	0	13	68.4%
HMM Recog. Rate		79.2%	47.1%	40.0%	70.0%	45.5%	93.3%	50.0%	

- Compute TP_i , TN_i , FP_i and FN_i with reference to a class C_i , considering all the other classes as negative
- Consequently, all the other metrics (precision, recall, etc.) are defined for the single class
- Metrics for the classifier can be obtained by averaging the metrics for the classes



$2 + 3 = 5$ sono i falsi negativi, sulle righe sono i falsi negativi

17/03/2023

Queste metriche ci aiutano ad avere un'idea di come poter confrontare due classificatori tra di loro.

Performance Evaluation

Statistical Evidence

Evaluating classifier accuracy

- A unique execution of different classifiers cannot allow concluding that one classifier is better than another
- In one trial, by chance one classifier could have an accuracy higher than the other
- We need to perform different trials. But how?
- We have only a labelled dataset.

Ma non ci basta fare un'unica prova vedendo la AUC o l'accuratezza su un unico test perché può accadere che in un'unica esecuzione possiamo essere particolarmente fortunati o sfortunati, ovvero avere un classificatore che si comporta particolarmente bene su quell'insieme di test avendolo addestrato con l'insieme di addestramento che avevamo. Dunque un unico esperimento potrebbe non essere sufficiente. Dobbiamo cercare di avere una distribuzione di valori ovvero ripetere l'esecuzione un certo numero di volte.

❓ Come possiamo fare avendo un unico insieme di addestramento e di test?

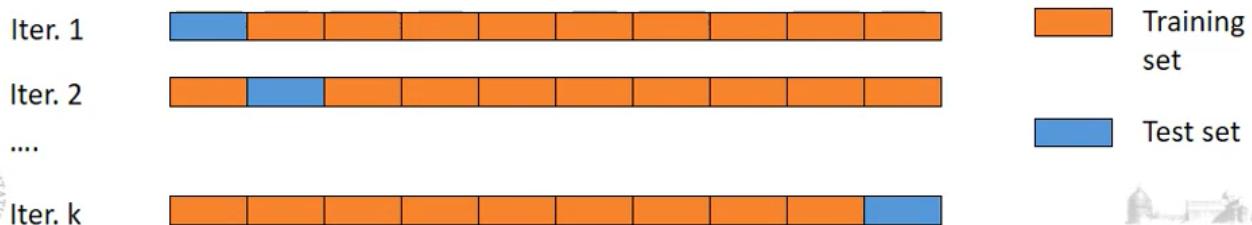
Nella pratica succede che abbiamo a disposizione un insieme di dati etichettato dunque questo insieme che tipicamente ci viene fornito da chi ci dà l'incarico di analizzare i dati ci fornisce un insieme di dati etichettati dunque non li divide in un insieme di addestramento e in un insieme di test. Questo lavoro viene svolto da "noi". Partiamo dall'insieme di dati etichettato e lo scomponiamo in un insieme di addestramento e in un insieme di test.

Performance Evaluation

Cross-Validation

Cross-validation (k-fold, where k = 10 is most popular)

- Randomly partition the data into k mutually exclusive subsets, each approximately equal size
- At i-th iteration, use D_i as test set and others as training set
- Leave-one-out: k folds where $k = \#$ of tuples, that is, only a sample is left out at a time for the test set. Suitable for small sized data.
- *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data



Possiamo dunque crearcirci più insiemi di addestramento e di test. Il metodo più popolare è quello che va sotto il nome di **cross validation**. Prendiamo un insieme di dati e lo dividiamo in k fold che contengono istanze uniche all'interno dell'insieme di addestramento. Nella figura stiamo usando un $k = 10$. Ripetiamo per 10 volte (in questo caso) il processo seguente: usiamo 9 fold per l'addestramento e il fold rimanente per il test. Ripetiamo 10 volte in quanto ogni volta usiamo un fold diverso per il test.

In questo modo testiamo il classificatore sempre su dati diversi. Questo approccio ci consente di generare 10 valori della metrica che stiamo considerando. Se vogliamo confrontare 2 classificatori usiamo una K-fold cross validation che in generale ha come valore di $k = 10$ e generiamo due distribuzioni di valori corrispondenti ai 2 classificatori.

Estimating Confidence Intervals: Null Hypothesis

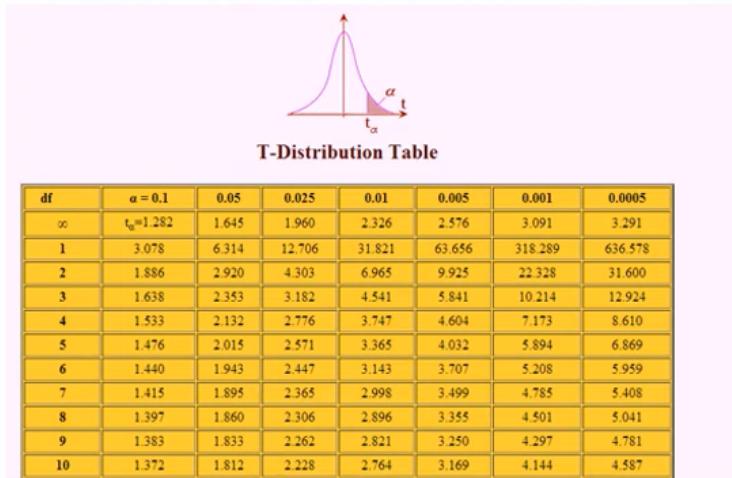
- Perform 10-fold cross-validation
- **Assumptions:** the samples (for instance, classification accuracy) are normally distributed within each group (output of each classifier) and the variances of the two populations are not reliably different
- Use t-test (or Student's t-test)
 - evaluate the differences in means between two groups
 - **Null Hypothesis:** the two distributions of accuracy for C_1 and C_2 are the same
- If we can reject null hypothesis, then
 - we conclude that the difference between C_1 and C_2 is statistically significant
- Choose the model with lower error rate
- No assumption: Wilcoxon test

Per capire se i 2 classificatori sono differenti, dobbiamo capire se queste 2 distribuzioni sono diverse oppure sono la stessa distribuzione. Se provo statisticamente che sono differenti, se una distribuzione ha un valore medio diverso dall'altro e in particolare stiamo usando come metrica l'accuratezza, posso concludere con evidenza statistica che un classificatore è migliore dell'altro. Voglio verificare se date le 2 distribuzioni queste possono essere considerate diverse. Per fare questo esistono diversi test statistici, quello più conosciuto è il **T-Test o Student's T-test** che assume che le distribuzioni dei valori siano distribuzioni gaussiane. Questo test valuta le differenze tra le medie nei 2 gruppi e valuta **l'ipotesi nulla** ovvero che le due distribuzioni di accuratezza per C_1 e C_2 siano la stessa distribuzione. Se questa ipotesi nulla può essere rifiutata allora si può concludere che esiste una differenza tra le due distribuzioni dal punto di vista statistico e scegliere il modello con l'accuratezza più alta o l'errore più basso come modello da usare. Quando applichiamo il T-test facciamo l'ipotesi che i campioni seguono una distribuzione normale o gaussiana e che la varianza delle 2 popolazioni non siano molto differenti. Questa ipotesi sulla normalità dobbiamo farla quando usiamo il T-test, esistono altri test statistici che non ci obbligano nel fare ipotesi sul tipo di distribuzione come ad esempio il Wilcoxon-test.

? Come applichiamo il t-test?

Estimating Confidence Intervals: Null Hypothesis

- Fix the confidence value α (typically $\alpha = 0.05$)
- Compute t and analyze the table of t-distribution: if $t > t_\alpha$ the null hypothesis is rejected and therefore the performance of the two classifiers is statistically different



df	$\alpha = 0.1$	0.05	0.025	0.01	0.005	0.001	0.0005
∞	$t_0 = 1.282$	1.645	1.960	2.326	2.576	3.091	3.291
1	3.078	6.314	12.706	31.821	63.656	318.289	636.578
2	1.886	2.920	4.303	6.965	9.925	22.328	31.600
3	1.638	2.353	3.182	4.541	5.841	10.214	12.924
4	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	1.476	2.015	2.571	3.365	4.032	5.894	6.869
6	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	1.372	1.812	2.228	2.764	3.169	4.144	4.587

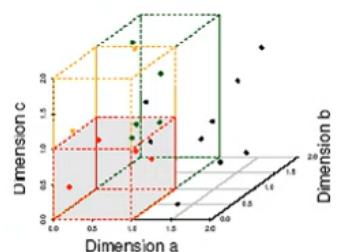
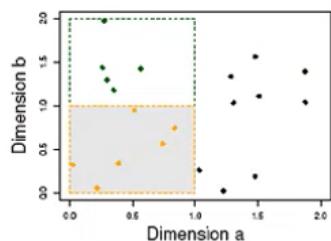
Calcoliamo il valore di T e lo confrontiamo con dei valori tabulati. In particolare deve accadere che il valore di T calcolato sia maggiore di quello in tabella, se questo accade possiamo concludere che l'ipotesi nulla può essere rifiutata e quindi che le 2 distribuzioni non sono la stessa distribuzione. Le tabelle chiedono di andare a individuare il grado di libertà ovvero la colonna df , nel caso specifico di una 10-cross validation è uguale al numero di ripetizioni -1 ovvero uguale a 9. L' α è il valore di confidenza che vogliamo avere nel risultato statistico che produciamo. *Più è basso il suo valore più è alta la confidenza sul rifiuto dell'ipotesi nulla* in quanto l' α indica la probabilità che l'ipotesi nulla sia vera. Tipicamente si sceglie un α uguale a 0.05.

Confrontando l'accuratezza o l'AUC potremmo trovarci ad avere delle distribuzioni molto ampie e dunque per caso potremmo trovarci ad avere che due valori medi sono differenti ma in realtà le due distribuzioni si sovrappongono, dunque il fatto che si abbia differenza nei valori medi potrebbe essere solo una conseguenza del caso.

I problemi reali della classificazione

Real Problems

- Many challenges
 - **Scalability:** millions of objects
 - **Computational cost** in learning and classification
 - **Dimensionality:** hundreds of features
 - Curse of dimensionality: all objects appear to be far from each other
 - **Imbalanced datasets**
 - Most methods tend to learn the majority class
 - **Interpretability**
 - Most applications require classifiers which are able to explain how they infer the result



Deep Learning

- Representation learning methods that
 - use a **cascade of multiple layers of nonlinear processing units** for feature extraction and transformation. Each successive layer uses the output from the previous layer as input.
 - learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts.



Raw representation



- Age **35**
- Weight **65**
- Income **23 k€**
- Children **2**
- Likes sport **0.3**
- Likes reading **0.6**
- Education **high**
- ... **...**



Higher-level representation

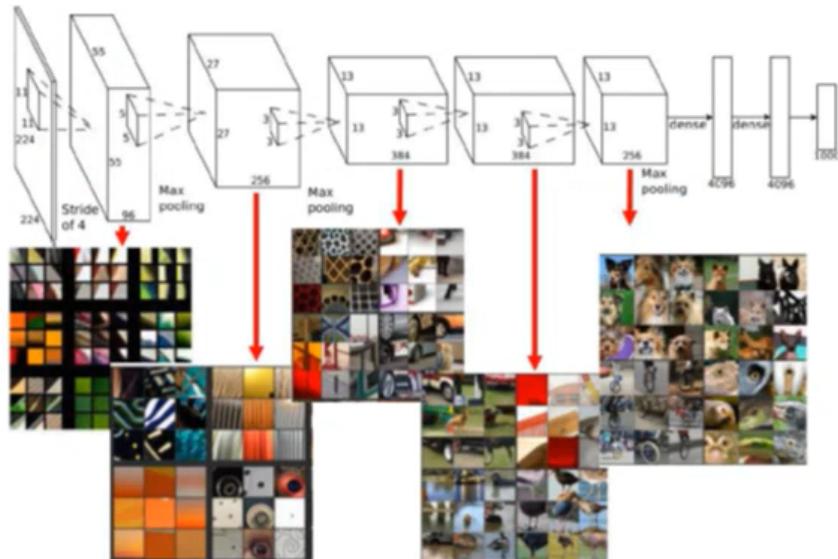
- Young parent **0.9**
- Fit sportsman **0.1**
- High-educated reader **0.8**
- Rich obese **0.0**
- ... **...**

Oggi si parla molto di deep-learning che sostanzialmente è una cascata di livelli multipli di unità di processazione non lineari connessi tra di loro. Questi sistemi hanno molti parametri che devono essere appresi e in qualche modo ben adattati al problema che vogliamo risolvere. Se abbiamo tanti parametri che vogliamo addestrare abbiamo bisogno di tanti dati, quindi per poter far apprendere questi sistemi abbiamo bisogno di tanti dati. Motivo per cui non tutti i problemi hanno una soluzione nel deep-learning.

Questo è un esempio di deep-learning in cui apprendiamo le immagini:

Deep Learning

- Multiple levels of abstraction

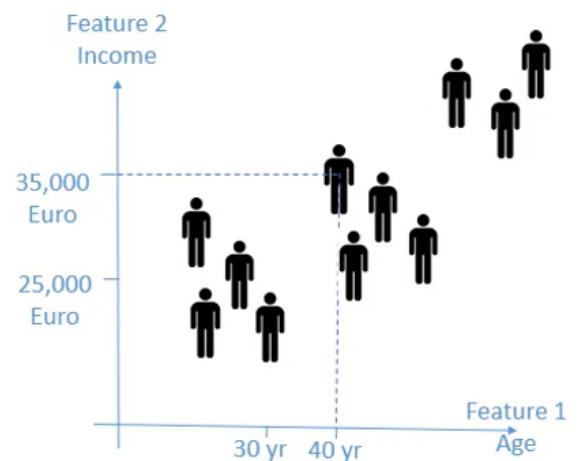


Clustering

La differenza sostanziale tra clustering e classificazione è che nella classificazione si partiva da un insieme di oggetti etichettati con la classe di appartenenza e cercavamo di apprendere un modello in modo tale che un nuovo oggetto potesse essere classificato. **Nel clustering invece partiamo da un insieme di dati non etichettato!** Non abbiamo alcuna ground truth o descrizione dell'oggetto tramite una label, se non le sue features. Lo scopo che ci prefiggiamo è raggruppare questi oggetti in qualche modo. Ovvero trovare degli insiemi, dei cluster di oggetti che sono molto simili tra di loro, che possono essere raggruppati. Non sappiamo in realtà se quei gruppi che stiamo creando sono gruppi che dal punto di vista dell'applicazione hanno un senso, ma vogliamo cercare di creare gruppi coesi:

Clustering

- **Objective**
 - **Find structure in the data**
 - **Group objects into clusters**
 - similar (or related) to one another within the same group
 - dissimilar (or unrelated) to the objects in other groups
 - **Unsupervised learning:** no predefined classes

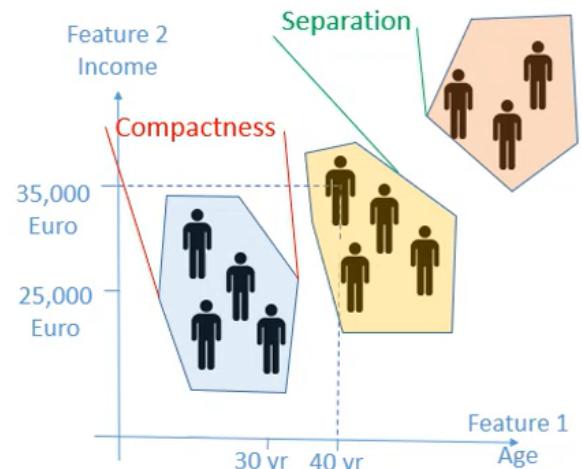


in pratica vuol dire che dobbiamo prima di tutto cercare di definire un'idea di somiglianza tra gli oggetti e una volta che abbiamo fissato come calcolare la somiglianza tra gli oggetti, applicare degli algoritmi che ci consentono di mettere nello stesso cluster oggetti simili e fare in modo che oggetti non simili siano su cluster diversi. Dunque raggruppare oggetti che sono simili tra di loro e mettere oggetti diversi su cluster diversi.

Questo è un esempio di cosa può fare un algoritmo di clustering, generando gruppi coesi e compatti e che siano separati tra di loro:

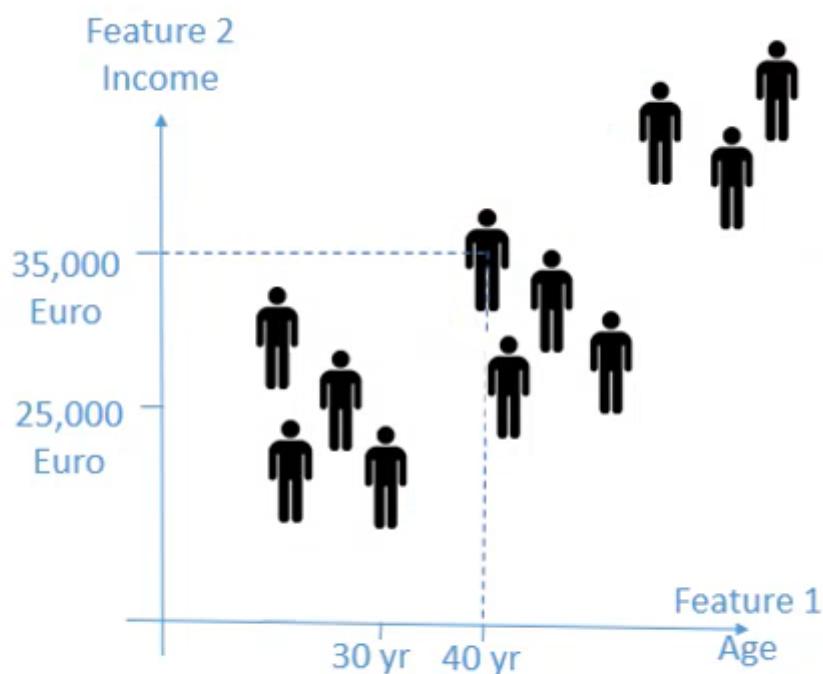
Clustering

- **Objective**
 - **Find structure in the data**
 - **Group objects into clusters**
 - similar (or related) to one another within the same group
 - dissimilar (or unrelated) to the objects in other groups
 - **Unsupervised learning:** no predefined classes

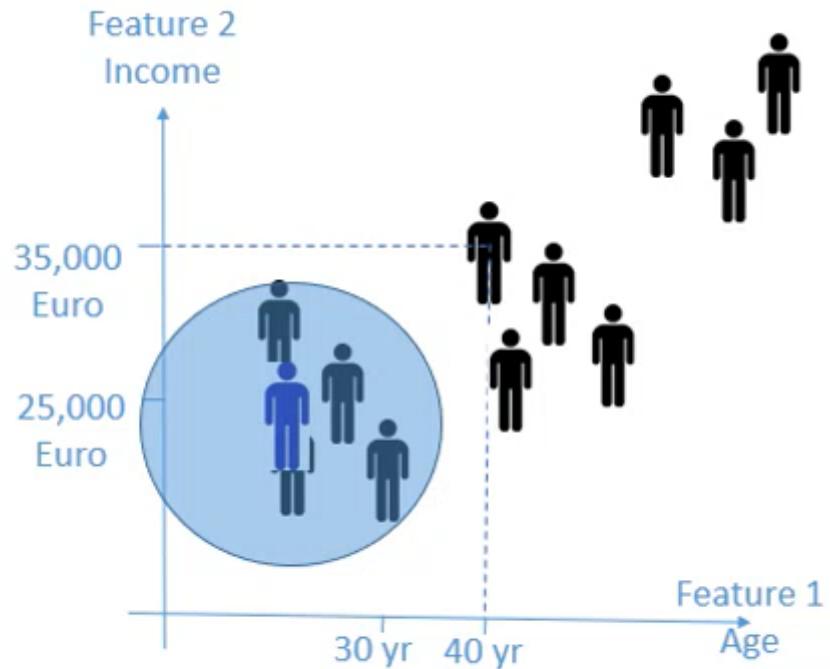


Uno degli algoritmi molto popolare e tra i primi proposti in letteratura è il **K-means**

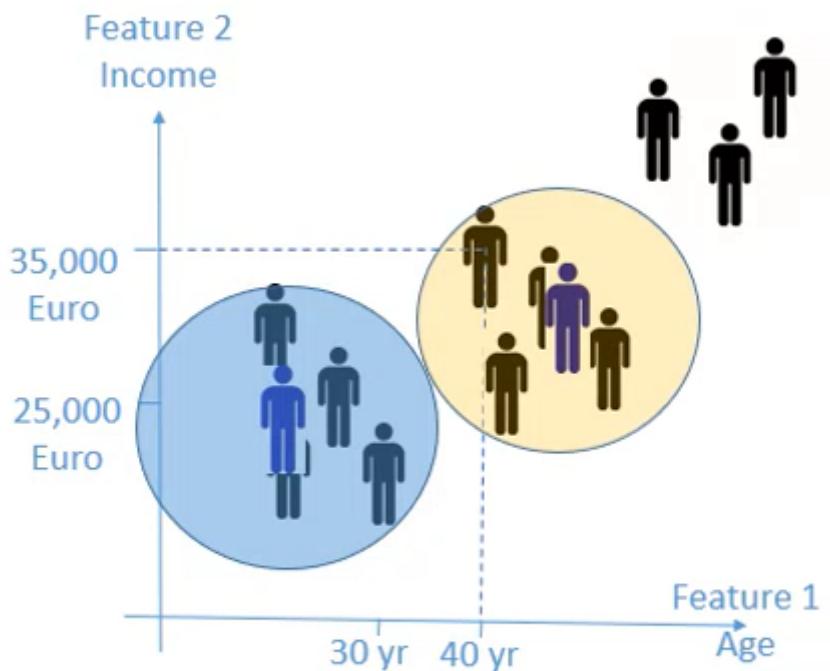
Definisce un prototipo per ogni cluster e va a minimizzare in modo iterativo la distanza tra il prototipo e i punti che appartengono al cluster:



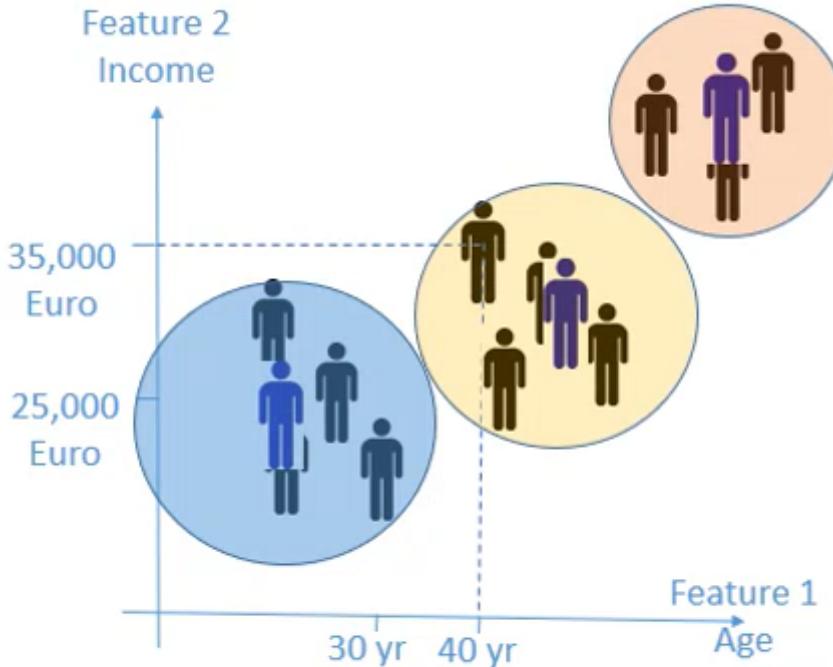
$K = 3$



$K = 3$



$K = 3$



$$K = 3$$

questo processo iterativo va a minizzare la funzione presentata nell'immagine:

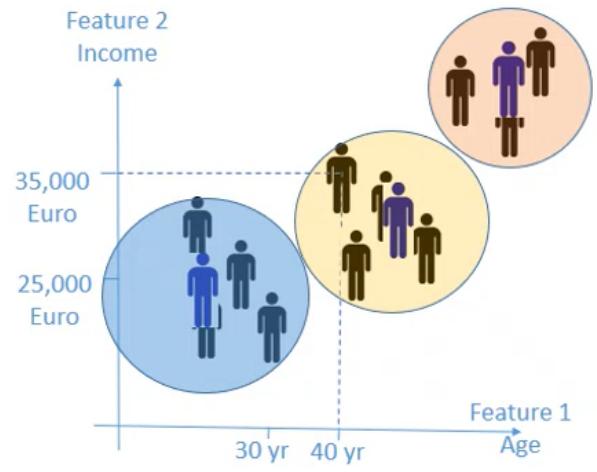
Clustering: K-means

- Find k subgroups that form compact and well-separated clusters
 - Each subgroup is represented by a prototype c_i
- Minimize iteratively

$$E = \sum_{i=1}^k \sum_{o \in C_i} dist(o, c_i)^2$$

where

$$c_i = \frac{1}{m_i} \sum_{o \in C_i} o$$



questa funzione è la sommatoria per $i = 1 \dots k$ con k uguale al numero di cluster della distanza tra ogni oggetto appartenente al cluster c_i rispetto al prototipo del cluster stesso.

! Dunque quello che cerca di fare questo algoritmo è cercare di minimizzare la distanza tra oggetti appartenenti a un cluster e il prototipo del cluster. Questo ovviamente rende il cluster compatto ma implicitamente aumenta anche la separazione tra i cluster.

➤ Il prototipo del cluster viene calcolato con la media presentata in basso nell'immagine.



L'algoritmo funziona in questo modo:

Clustering: K-means

- Pseudo Code

Algorithm: k-means. The k -means algorithm for partitioning, where each cluster's center is represented by the mean value of the objects in the cluster.

Input:

- k : the number of clusters,
- D : a data set containing n objects.

Output: A set of k clusters.

Method:

- (1) arbitrarily choose k objects from D as the initial cluster centers;
- (2) **repeat**
- (3) (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
- (4) update the cluster means, that is, calculate the mean value of the objects for each cluster;
- (5) **until** no change;



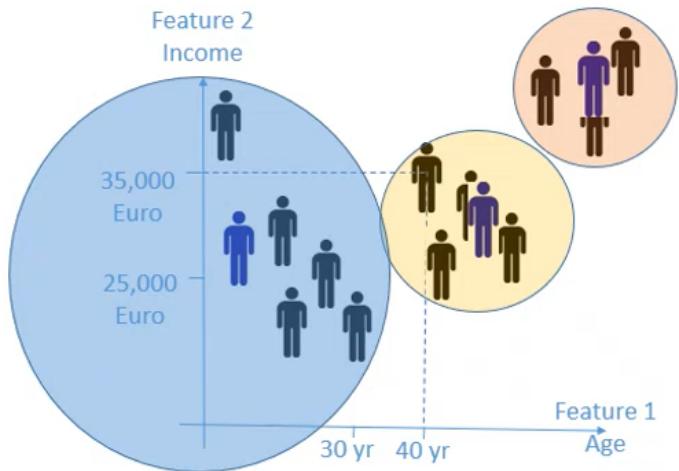
dobbiamo fissare prima di tutto il valore K ovvero dobbiamo dire all'algoritmo quanti cluster vogliamo. Poi l'algoritmo sceglie arbitrariamente K oggetti dall'insieme come prototipi iniziali e ripete le azioni seguenti. Assegna ogni oggetto al cluster a cui è più vicino e aggiorna i prototipi del cluster. Ripete queste azioni fin tanto che non ci sono più spostamenti di un oggetto tra un cluster e un altro. Il risultato finale è avere dei cluster di oggetti con la proprietà di aver minimizzato la distanza tra l'oggetto appartenente al cluster e il suo prototipo.

Questo algoritmo ha un aspetto su cui soffermarci, nel clustering non conosciamo nulla sugli oggetti eccetto la loro descrizione tramite le caratteristiche, quindi **come possiamo fissare il K ?**

Clustering: means

- **K-means**

- Sensitive to outliers
- An object with an extremely large and different value may substantially distort the distribution of the data and therefore the centroid



$K = 3$



? Può la funzione di costo E venirci in aiuto per minimizzare il valore di K?

- Minimize iteratively

$$E = \sum_{i=1}^k \sum_{o \in C_i} dist(o, c_i)^2$$

where



$$c_i = \frac{1}{m_i} \sum_{o \in C_i} o$$

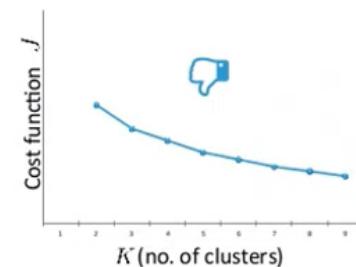
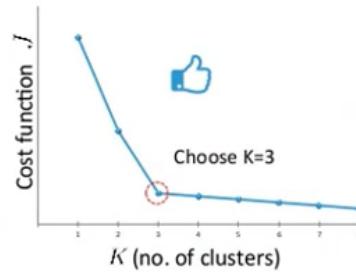
Potrei pensare di lavorare in questo modo: fissare $K = 2$ eseguire l'algoritmo K-means e vedere quale è il valore finale della funzione di costo E, e continuare così fin tanto che non trovo il K tale per cui ho il valore minore di E.

Ovviamente all'aumentare di K la funzione di E diminuisce in ogni caso, e quindi quando $K = \text{numero di oggetti}$ abbiamo una distanza tra l'oggetto e il prototipo del cluster pari a 0 in quanto ci sarà a tendere un cluster formato da ogni singolo oggetto. L'approccio proposto dunque non può funzionare, quindi questa soluzione per trovare il valore di K non ci soddisfa.

Si potrebbe allora pensare di usare un approccio euristico:

Clustering

- **Elbow method** to determine the number of clusters
 - From k=2 to a predefined maximum value
 - Apply the clustering algorithm
 - Calculate the sum of within-cluster variances $\text{var}(k)$
 - Plot the curve of $\text{var}(k)$ with respect to k
 - Choose k as the first or most significant turning point of the curve



Vado ad analizzare qual'è l'andamento della funzione di costo all'aumentare di K, se effettivamente esistono dei cluster naturali, quel profilo che dovrei ottenere è quello che vedo nella figura in alto nella slide, ovvero avere un repentino calo nella funzione di costo fino ad arrivare ad un valore di K per cui in realtà comincio ad avere un calo più lieve.

Ad esempio se parto con un insieme di dati che dovrebbe avere effettivamente 3 cluster, nel momento in cui divido in 2 cluster ho effettivamente una grossa diminuzione in quanto sto creando dei cluster

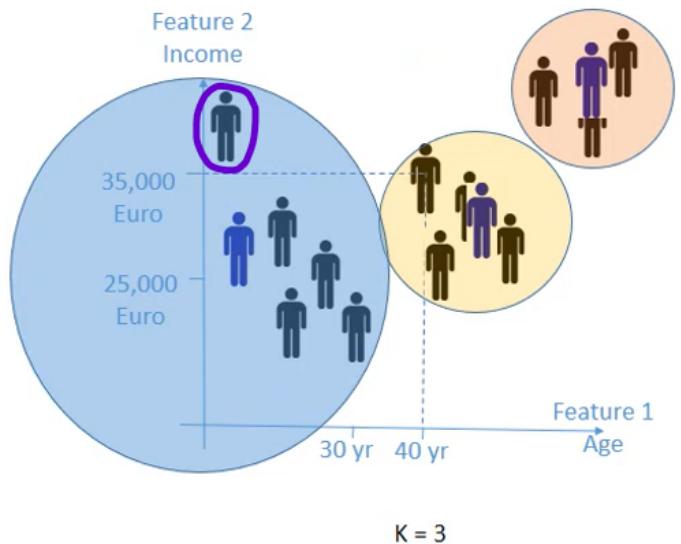
che effettivamente esistono. Quando incomincio ad avere poca diminuzione della funzione di costo? Quando in realtà vado a spezzare un cluster naturale, ovvero la distribuzione dei punti all'interno del cluster è all'incirca uniforme quindi separando il cluster non ho dei grandi vantaggi in termini di funzione di costo, **ovviamente decrescerà ma non in maniera così forte!**

Se ottengo un profilo come quello in alto a destra, posso andare a individuare il numero ottimale di cluster andando a prendere il valore di K corrispondente al gomito (**elbow**) della funzione. Se invece all'aumentare di K ho il profilo come quello presentato nel grafico sottostante ho una situazione diversa in quanto **probabilmente non esistono dei cluster naturali** in quanto è come se avessimo dei punti egualmente distribuiti nello spazio, quindi eseguendo il K-means con un valore di K i cluster vengono individuati ma in realtà sono dei cluster forzati e non esistono realmente nell'insieme di dati.

Sensibilità agli outliers

Clustering: means

- **K-means**
 - Sensitive to outliers
 - An object with an extremely large and different value may substantially distort the distribution of the data and therefore the centroid



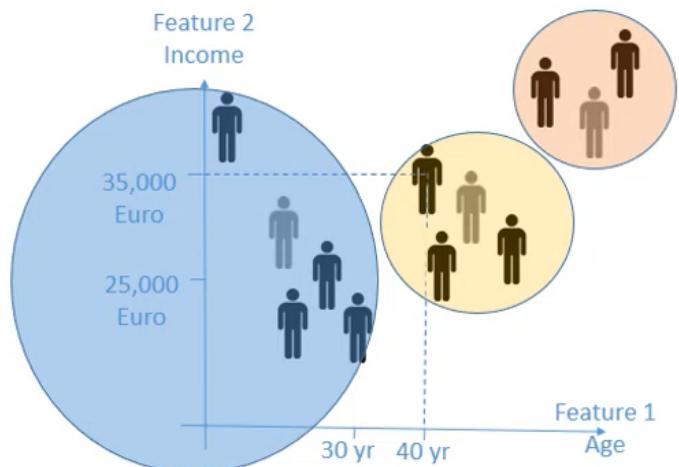
L'omino cerchiato in viola è abbastanza distante dagli altri dunque possiamo considerarlo come un outlier. Quando eseguo il K-means, tutti gli oggetti che sono nello spazio in ogni caso vengono inseriti in cluster, dunque se eseguo il K-means con $K = 3$ ottengo all'incirca i cluster definiti nella slide.

Questo vuol dire che l'oggetto che è molto diverso dagli altri viene effettivamente inserito in un cluster e l'effetto che questo comporta nel cluster è che siccome il prototipo del cluster è calcolato come media degli oggetti all'interno del cluster avrà come effetto che l'outlier trascina in qualche modo verso di sé il prototipo e dunque il prototipo del cluster viene spostato un pochino verso l'outlier ovviamente se questo è molto lontano lo spostamento è molto marcato.

Possiamo mitigare questo problema usando un algoritmo leggermente diverso dal K-means detto **K-medoids**:

Clustering: K-medoids

- **K-medoids:** prototype of a cluster is the most centrally located object in the cluster
 - Less sensitive to outliers
 - Can manage also categorical values

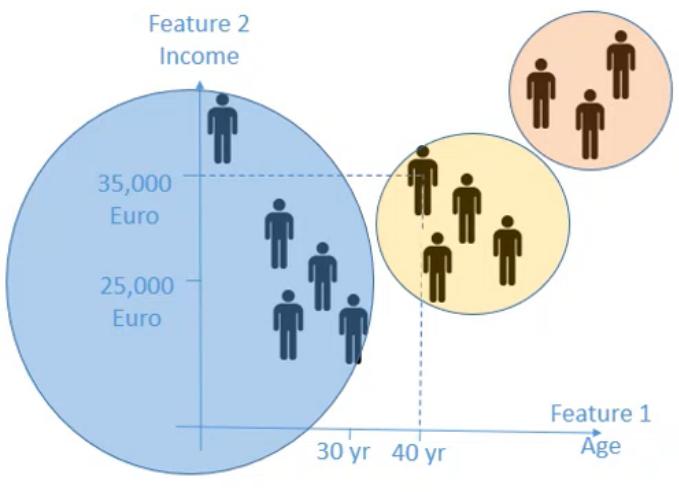


la differenza è che fondamentalmente non si usa la media come abbiamo visto prima ma uso un oggetto del data set come prototipo abbiamo detto infatti che nel K-means il prototipo è la media degli oggetti appartenenti al cluster quindi può essere un oggetto del dataset ma tipicamente è un oggetto virtuale, in realtà non è un oggetto del dataset. In questo caso invece il prototipo è un oggetto del dataset. Ci riduce il problema in quanto se il prototipo è un oggetto del dataset, quell'effetto di trascinamento del prototipo verso l'outlier è ovviamente mitigato, in quanto in ogni caso quel prototipo viene scelto tra gli oggetti che sono caratterizzanti del cluster e quindi si ritrova tra quelli oggetti tipici del cluster.

Sia il K-means che il K-medoids producono [cluster di forma sferica](#):

Clustering

- K-means and k-medoids drawbacks
 - Only spherical clusters
 - Need to fix the number of clusters before executing the clustering algorithm
- Is it possible to determine the number of clusters?
 - There exist some empirical methods



la ragione è nella scelta del tipo di distanza che viene usata nel K-means o K-medoids classico.

Quando calcoliamo infatti la distanza euclidea ogni dimensione ha lo stesso peso.

Se le singole dimensioni venissero pesate in modo diverso daremmo vita a un cluster di forma **ellissoidale**. Nel caso comunque del K-means e del K-medoids i cluster che vengono generati hanno una forma prefissa che viene prefissa dal tipo di distanza che stiamo usando, nel K-means classico questa distanza essendo euclidea ci porta a cluster sferici, nel caso di K-means con distanza prefissa ci porterebbero a dei cluster ellisoidali.

In ogni caso qualsiasi sia la distanza usata, quando viene usato il K-means andiamo a produrre **cluster convessi**, che siano essi sferifici o ellisoidali.

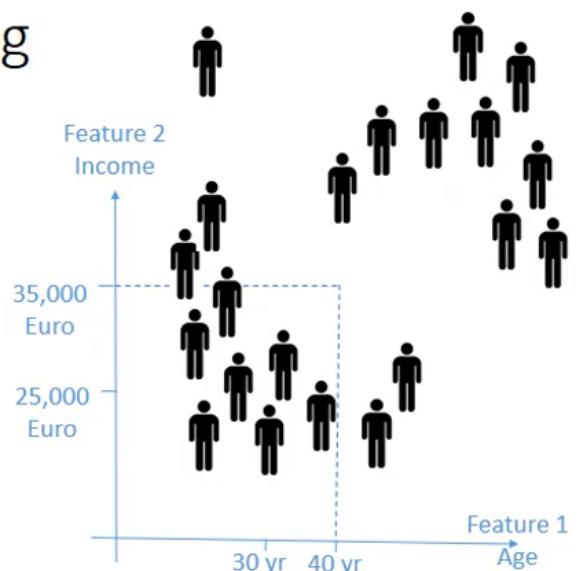
Questi cluster convessi sono nel numero stabilito prima di eseguire il K-means.

dunque una volta fissato K e fissato il tipo di distanza da usare (tipicamente euclidea) eseguiamo l'algoritmo e tutti gli oggetti che fanno parte dell'insieme di dati vengono collocati in qualche cluster. Il K-means fa parte di quegli algoritmi di cluster partizionali in quanto partizionano l'insieme di dati in un numero finito di cluster che è stabilito all'esecuzione come parametro dell'algoritmo stesso.

Esiste invece una famiglia di algoritmi di clustering basati sulla densità:

- **Density-based clustering**
 - Clusters correspond to high density zones
 - One of the most famous clustering algorithms: Density-Based Spatial Clustering of Applications with Noise (DBSCAN)
 - Discover clusters of arbitrary shape
 - Handle noise
 - Only one scan of the dataset

Clustering



© 2018 DataCamp

l'idea infatti è quella di creare dei cluster andando ad analizzare la densità dei punti all'interno dello spazio in particolare individuando delle zone dense di oggetti. Le zone dense di oggetti sono separate da zone a bassa densità di punti. Dunque le zone dense sono cluster di oggetti e le zone non dense sono quelle che separano i cluster tra di loro.

L'algoritmo più conosciuto basato sulla densità è il **DBSCAN** il quale permette di scoprire cluster di forma arbitraria seguendo le densità.

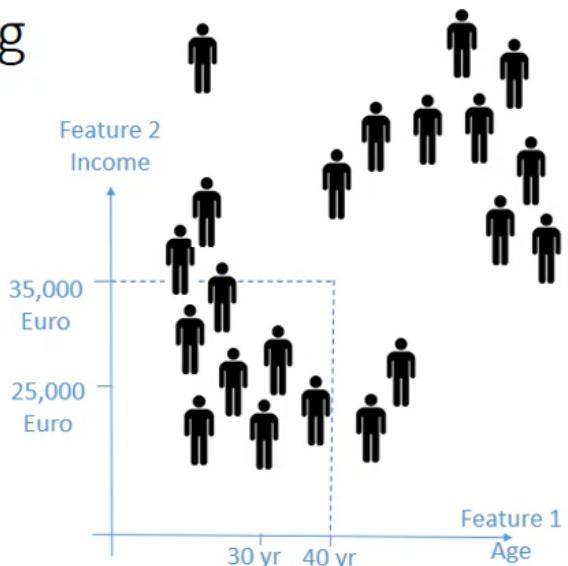
Inoltre permette di gestire il rumore (gli outlier) e non necessita di tante scansioni del dataset.

Per essere eseguito richiede che vengano inizializzati solo 2 parametri (non è necessario inizializzare il

numero di cluster):

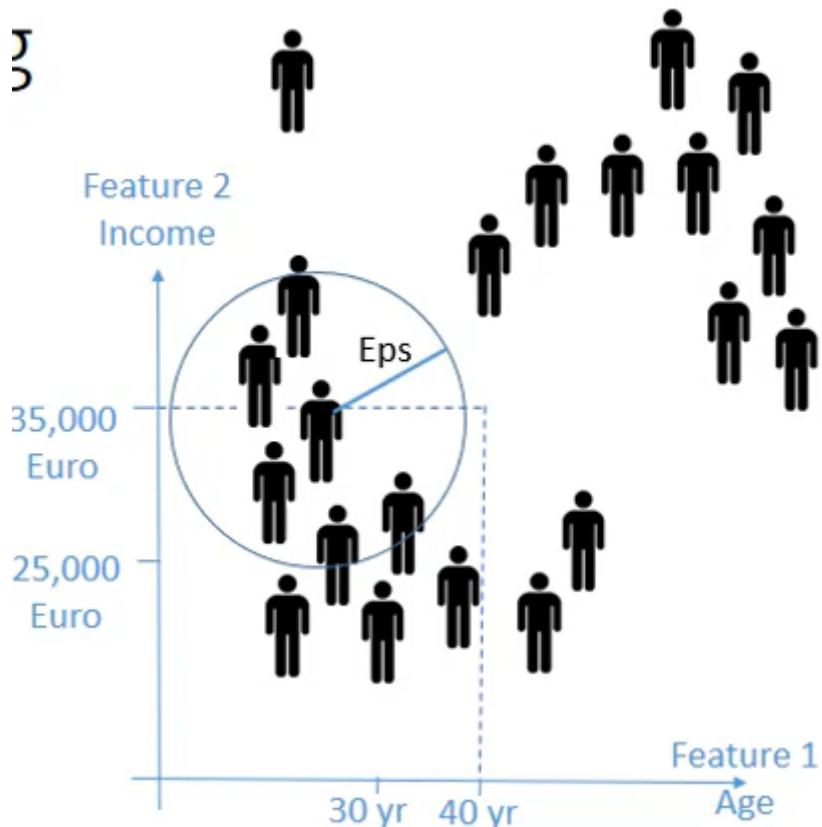
Clustering

- DBSCAN parameters
 - **Eps:** maximum radius of the neighbourhood of an object
 - **MinPts:** Minimum number of points in an Eps-neighbourhood of that object
- Eps and MinPts determine our concept of density in the data set



- **Eps:** raggio massimo per andare ad individuare il vicinato di un oggetto
- **MinPts:** il numero minimo di punti nel vicinato di un oggetto

tramite questi 2 parametri riusciamo a fissare la nostra idea di densità in quanto viene espressa come numero minimo di punti nel vicinato di un oggetto diviso il volume del vicinato dell'oggetto:



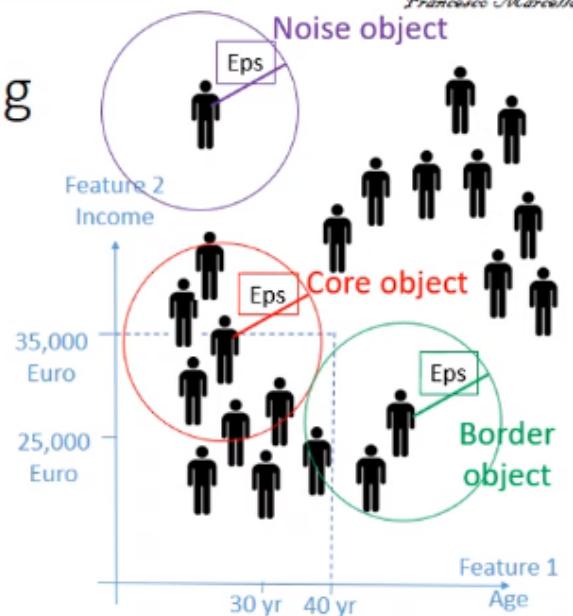
01/04/2023

Un oggetto viene detto **core** se ha almeno MinPts all'interno del suo vicinato mentre un oggetto è detto **bordo** se all'interno del suo vicinato ha un numero di oggetti inferiori al MinPts. Un oggetto è detto

noise (outlier) se non è né un oggetto core né un oggetto bordo.

Clustering

- DBSCAN preliminaries
 - An object is a **core object** if it has more than MinPts points within Eps
 - A **border object** has fewer than MinPts within Eps, but it is in the neighbourhood of a core object
 - A **noise object** is any object that is not a core object or a border object

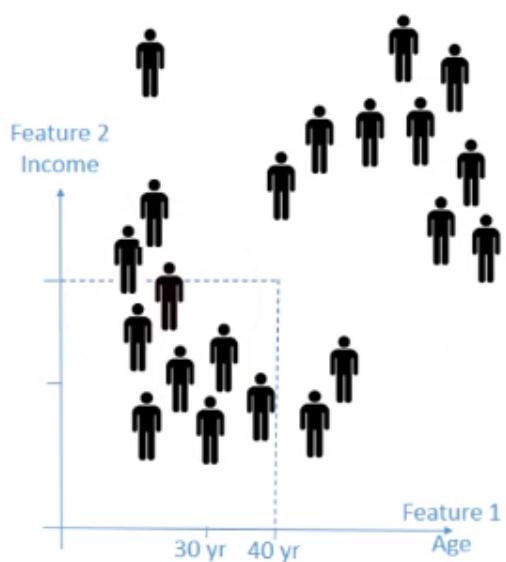


L'oggetto bordo si trova al bordo del cluster e in qualche modo è l'oggetto che è tra la zona densa che individua il cluster e la zona non densa che separa il cluster.

La strategia adottata dal DBSCAN è rappresentata nelle seguenti 2 immagini:

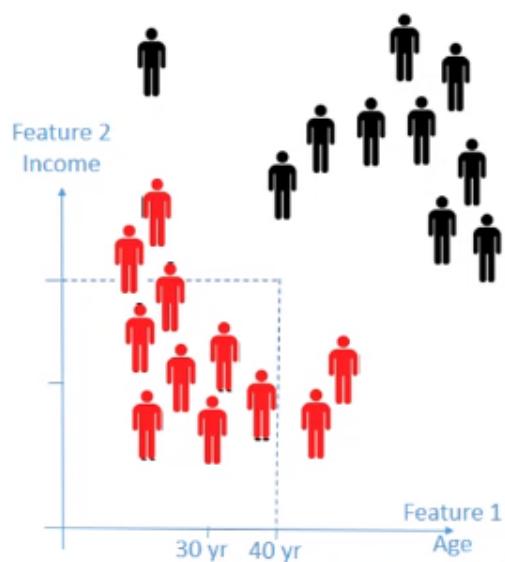
Clustering

- DBSCAN strategy
 - If two core objects are close enough (within the Eps distance of one another) are put in the same cluster
 - Any border object that is close enough to a core object is put in the same cluster as the core object
 - Noise objects are discarded



Clustering

- **DBSCAN strategy**
 - If two core objects are close enough (within the Eps distance of one another) are put in the same cluster
 - Any border object that is close enough to a core object is put in the same cluster as the core object
 - Noise objects are discarded



Due oggetti **core** che sono abbastanza vicini tra di loro vengono inclusi nello stesso cluster e vengono anche inseriti nello stesso cluster tutti gli oggetti nel vicinato di questi oggetti core. Tutto questo fin quando non si raggiungono degli oggetti **bordo** a questo punto il processo termina e si va a costruire un nuovo cluster estrando tra gli oggetti rimanenti un oggetto a caso.

Come parametri per l'esecuzione del DBSCAN abbiamo l'epsilon e il min points:

Clustering

- **DBSCAN pseudo-code**

Algorithm: DBSCAN: a density-based clustering algorithm.

Input:

- D : a data set containing n objects,
- ϵ : the radius parameter, and
- $MinPts$: the neighborhood density threshold.

Output: A set of density-based clusters.

Clustering

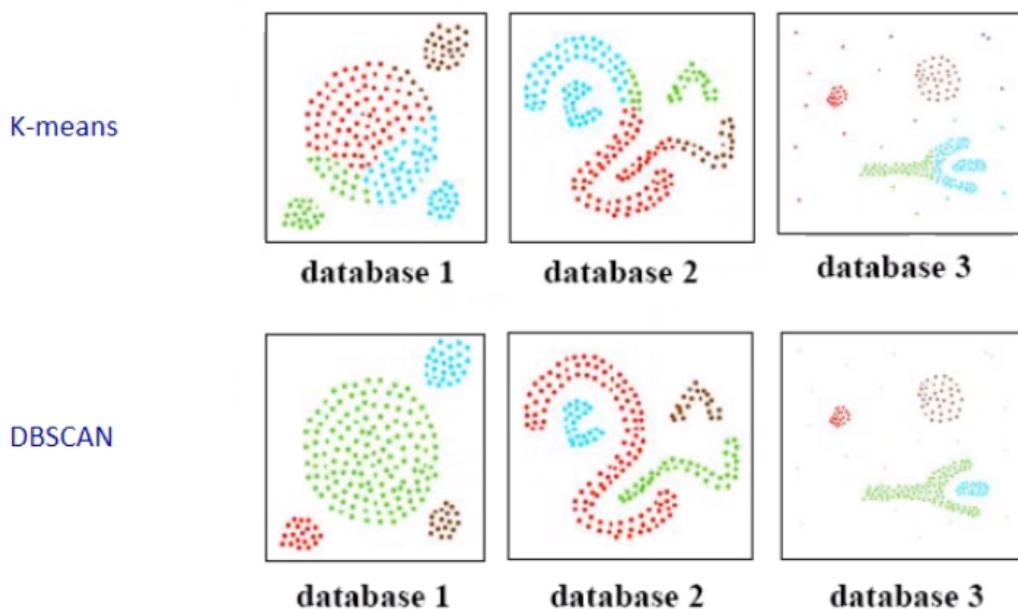
Method:

- (1) mark all objects as **unvisited**;
- (2) **do**
- (3) randomly select an unvisited object p ;
- (4) mark p as **visited**;
- (5) **if** the ϵ -neighborhood of p has at least $MinPts$ objects
- (6) create a new cluster C , and add p to C ;
- (7) let N be the set of objects in the ϵ -neighborhood of p ;
- (8) **for** each point p' in N
- (9) **if** p' is **unvisited**
- (10) mark p' as **visited**;
- (11) **if** the ϵ -neighborhood of p' has at least $MinPts$ points,
 add those points to N ;
- (12) **if** p' is not yet a member of any cluster, add p' to C ;
- (13) **end for**
- (14) output C ;
- (15) **else** mark p as **noise**;
- (16) **until** no object is **unvisited**;

Si estrae a caso un oggetto e si vede se è un punto core, se lo è si inizia un cluster e si considerano tutti gli oggetti all'interno del vicinato che stiamo considerando. Se sono oggetti core vengono inseriti nel cluster e continuiamo a inserirli fin tanto che non troviamo oggetti che non sono più core, quegli oggetti formeranno fattivamente il bordo del cluster. Se l'oggetto estratto a caso non è core viene inserito negli oggetti **outlier**. Non è detto che questi oggetti siano degli outlier alla fine del processo del DBSCAN perchè potrebbero essere inseriti al termine dell'algoritmo come oggetti bordo di qualche cluster.

Alla fine abbiamo dei cluster stabiliti dalla densità più degli outlier. Il grande vantaggio del DBSCAN è che in uscita non avremo solo dei cluster ma abbiamo anche la possibilità di individuare degli outlier, ovvero oggetti non inseriti in alcun cluster.

Clustering: Density-based vs. Partitional



❓ Quanto il risultato finale dell'esecuzione del DBSCAN dipende dai parametri che stiamo usando?

Ovvero quanto questi risultati sono sensibili alla scelta dei parametri?

Il K-means è sensibile alla scelta di K ma a prescindere dalla scelta i cluster che individuava erano sferici.

Nel secondo dataset di K-means c'è un cluster abbastanza grande e poi ci sono 3 cluster più piccoli. Si nota che i cluster non sono sferici né tantomeno convessi.

Per il dataset 3 abbiamo una situazione per cui ci sono dei punti considerati outlier e poi ci sono dei cluster sia sferici che concavi. Nel primo dataset per il K-means con $K = 4$, in quanto 4 erano i cluster che ci aspettavamo, in realtà genera dei cluster che non rispecchiano i cluster naturali all'interno del dataset. Questo perché il K-means tende a creare dei cluster sferici e cerca di creare cluster che sono all'incirca della stessa grandezza.

Se consideriamo il dataset 2 del K-means, siamo molto lontani dai cluster naturali dell'insieme di dati, questo perché il K-means genera cluster sferici. Il database 3 è l'ennesima dimostrazione di questa problematica.

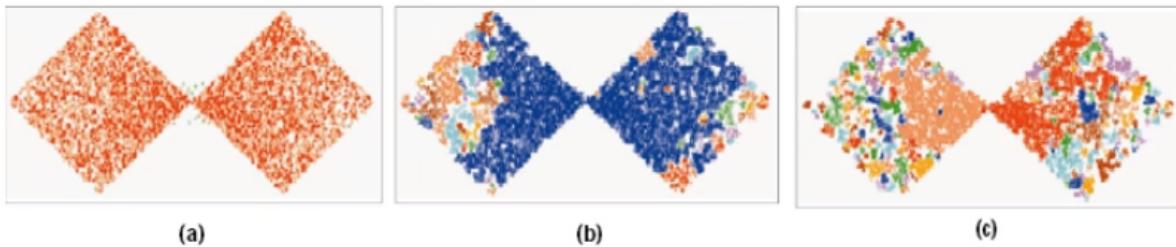
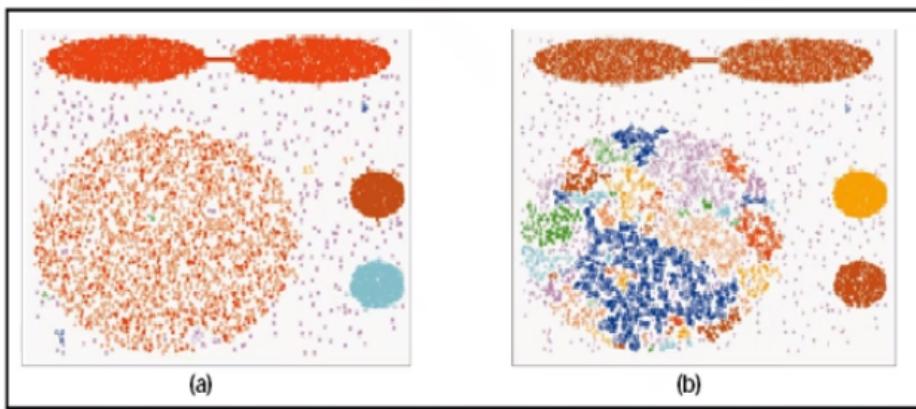
Se eseguiamo il DBSCAN, ad esempio vedendo le uscite per il dataset 1, vengono correttamente individuati i 4 cluster naturali, stessa cosa per il dataset 2 e dataset 3.

❓ Dunque il DBSCAN è sempre l'algoritmo da usare? **Dipende.**

Quando abbiamo zone a densità diverse, DBSCAN non funziona bene. I due parametri eps e minpts ci fissano la densità per l'intero spazio, ma se nello spazio abbiamo densità diversa dei punti, DBSCAN in zone altamente dense individua tanti cluster mentre in zone poche dense individua tutti gli oggetti come outlier. Il DBSCAN è fortemente sensibile alla scelta dei parametri.

Clustering: Sensitive to Parameters

Figure 8. DBScan results for DS1 with MinPts at 4 and Eps at (a) 0.5 and (b) 0.4.



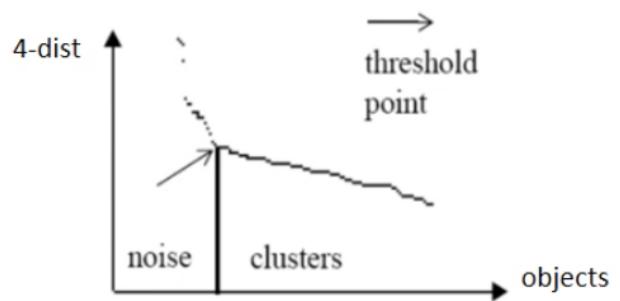
Passando dalla figura (a) alla figura (b), abbassando il valore di eps da 0.5 a 0.4 vuol dire andare a impostare una densità maggiore, in quanto lasciamo inalterato il numero minimo di punti ma diminuiamo il volume considerato in quanto diminuiamo eps ed è come se stessimo cercando una densità maggiore per individuare i cluster. Cambia in quella zona in cui abbiamo una densità di punti ma non così densa.

Il DBSCAN è fortemente dipendente dalla combinazione di eps e minpts e inoltre fissa una nostra idea di densità che è valida per l'intero spazio. C'è un modo per fissare i valori di epsilon e minpts nel modo ottimale?

Un possibile approccio euristico descritto di seguito **vale solo se possiamo assumere che all'interno dello stesso spazio ci sia la stessa densità:**

Clustering: how to determine Eps and MinPts

- For a given MinPts we define a function MinPts-dist , mapping each point to the distance from its MinPts -th nearest neighbor.
- We sort the object in descending order of their MinPts-dist values: the plot of this function gives some hints concerning the density distribution.
- If we choose an arbitrary object o , set the parameter Eps to $\text{MinPts-dist}(o)$, all points with an equal or smaller MinPts-dist value will be core points.

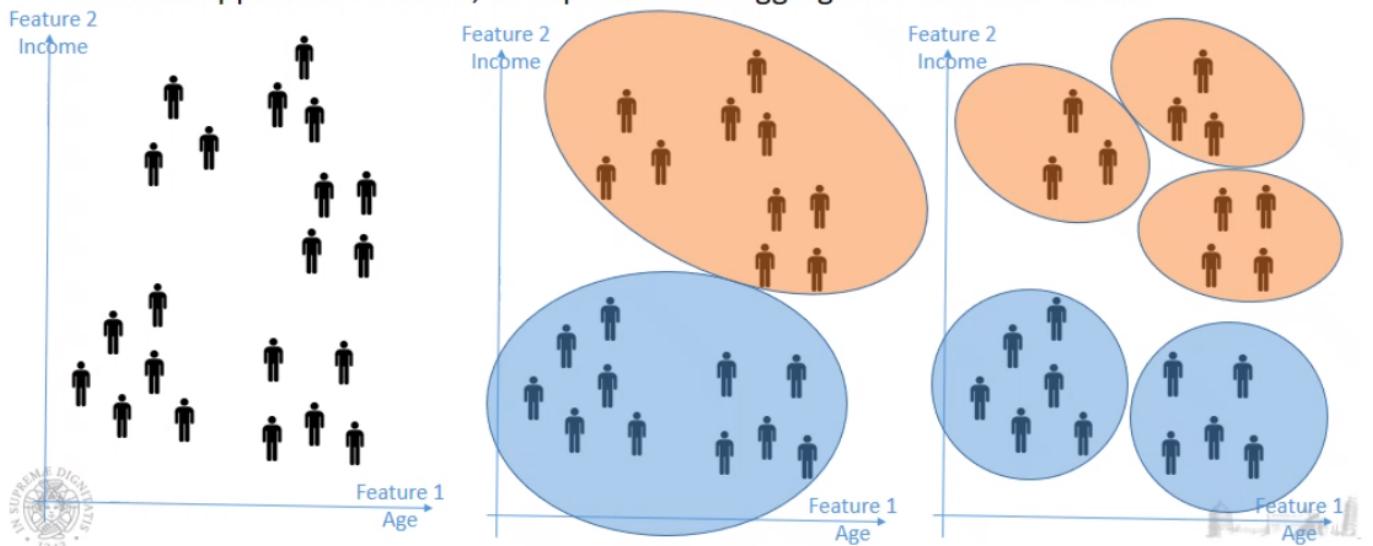


Fissiamo la distanza pari ad esempio a 4 e quindi andiamo a determinare i vicini di un oggetto prendendo il 4°. Produciamo poi un plot di tutte queste distanze ordinate in modo decrescente. Nelle ascisse abbiamo gli oggetti mentre nelle ordinate abbiamo le distanze di ogni oggetto rispetto al suo 4° vicino.

Noi possiamo scegliere un valore per eps tale per cui ci sono pochi punti che hanno un minPts distance maggiore del valore che scegliamo. Se prendiamo il valore individuato dalla freccia per quanto riguarda eps siamo sicuri che tutti i punti che vanno dall'oggetto individuato dalla freccia fino a tutti gli oggetti sulla destra sono sicuramente dei punti core. Questo perché stiamo individuando attraverso la minPts distance la distanza rispetto all'oggetto del minPts vicino. Se quella distanza è minore di eps siamo sicuri si tratti di un oggetto core. Quello individuato dalla freccia è l'oggetto core e tutti gli oggetti sulla destra sono core. Quelli sulla sinistra della freccia non è detto che siano outlier, potrebbero essere board. In ogni caso questo plot ci indica come scegliere una epsilon distance in modo tale da avere un numero limitato di outlier.

Hierarchical Clustering

- In some application domain, multiple levels of aggregation can be desirable



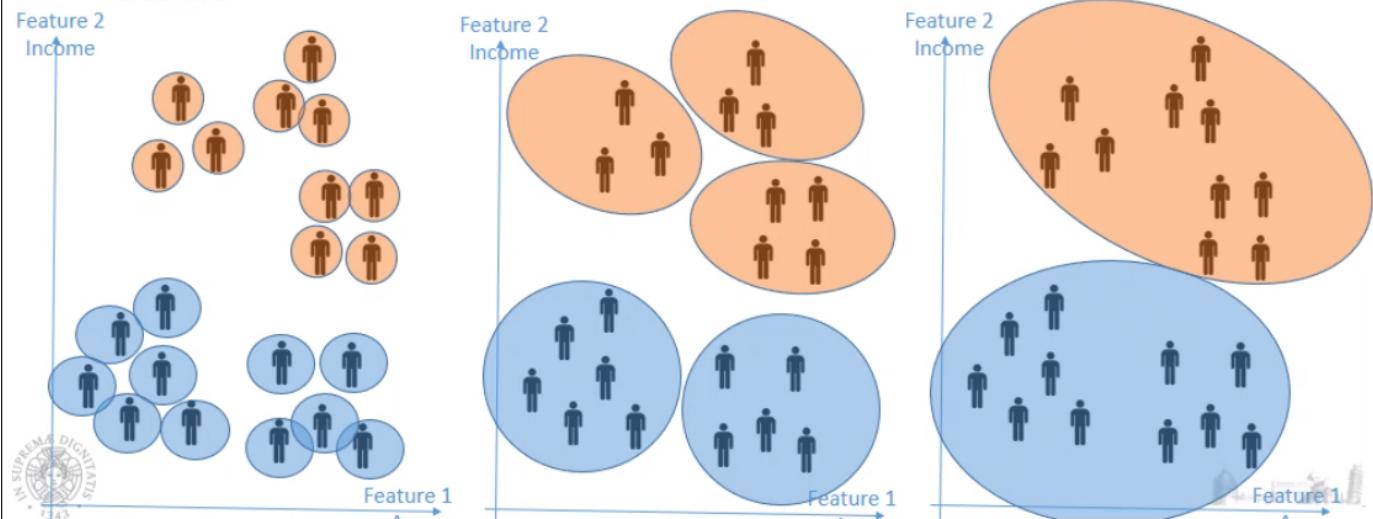
Questa introduzione fatta sugli algoritmi di clustering basati sulla densità ci ha fatto capire come questi algoritmi approcciano il problema basandosi sulla densità. I primi algoritmi visti invece come il K-means che sono algoritmi partizionali, minimizzano una funzione di costo trovando una partizione. Esiste poi un'altra tipologia di algoritmi di clustering, detti **gerarchici**, permettono di raggruppare i punti in diversi livelli. In altre parole facciamo vedere che l'insieme di punti sulla sinistra potrebbe essere raggruppato in modo grossolano al centro e potrebbe essere rifinito con il clustering sulla destra. Non ci danno solo una partizione piatta, ma ci danno una partizione che in realtà sono più partizioni perché formano una gerarchia.

Esistono 2 tipi di approccio per gli algoritmi gerarchici:

- agglomerativo:**

Hierarchical Clustering

- Agglomerative approaches:** clusters are merged if, for instance, an object in C1 and an object in C2 form the minimum distance between any two objects from different clusters



Si parte dall'avere un cluster per ogni oggetto e poi si agglomerationo.

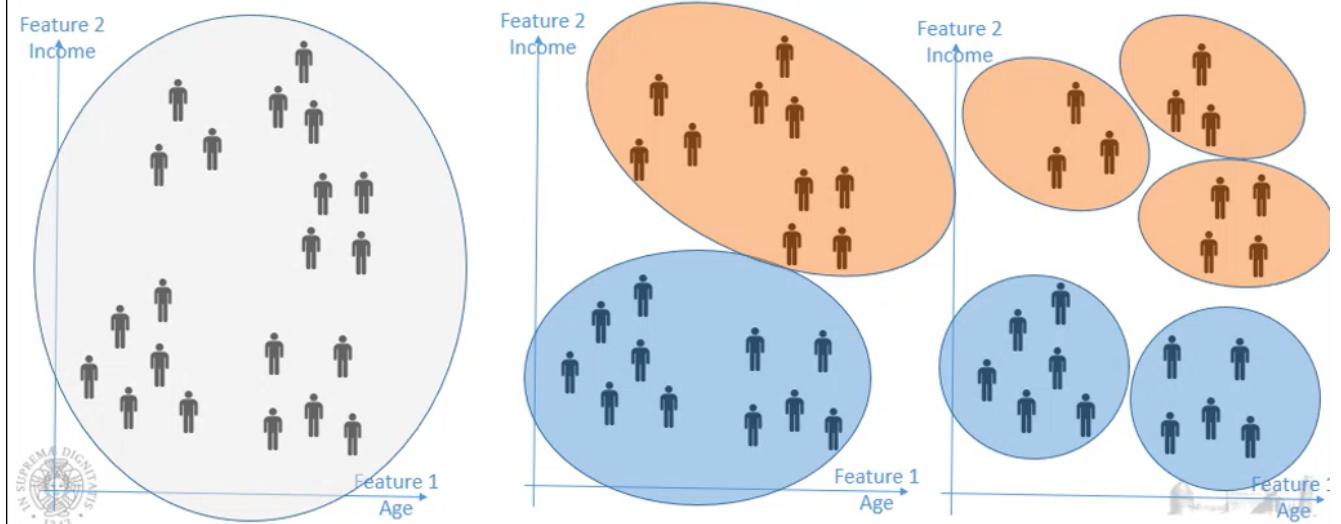
Avremo dunque un primo livello nella figura in cui sono stati agglomerati tutti i piccoli cluster formati da un oggetto, formando 3 cluster in alto e 2 in basso. Poi questi vengono agglomerati in ulteriori 2 cluster complessivi.

Esistono vari modi per mettere insieme i cluster a vari livelli, l'idea è di partire da un cluster per un singolo modello e via via raggruppare i cluster.

- **divisivo:**

Hierarchical Clustering

- **Divisive approaches:** a cluster is split according to some principle, e.g., the maximum Euclidean distance between the closest neighboring objects in the cluster



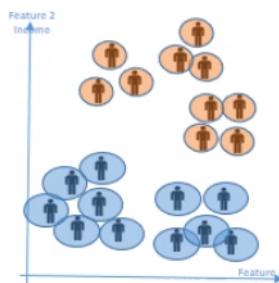
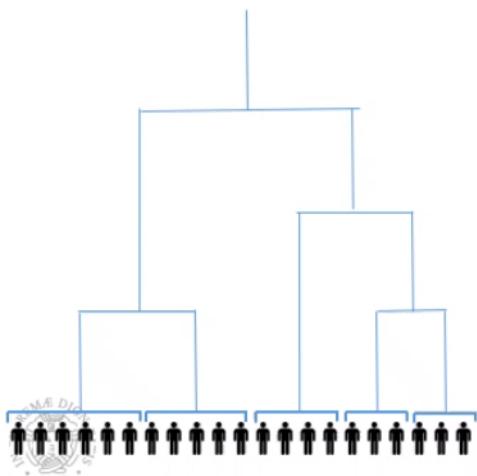
L'altro approccio è divisivo, si parte da un unico cluster e via via partizioniamo.

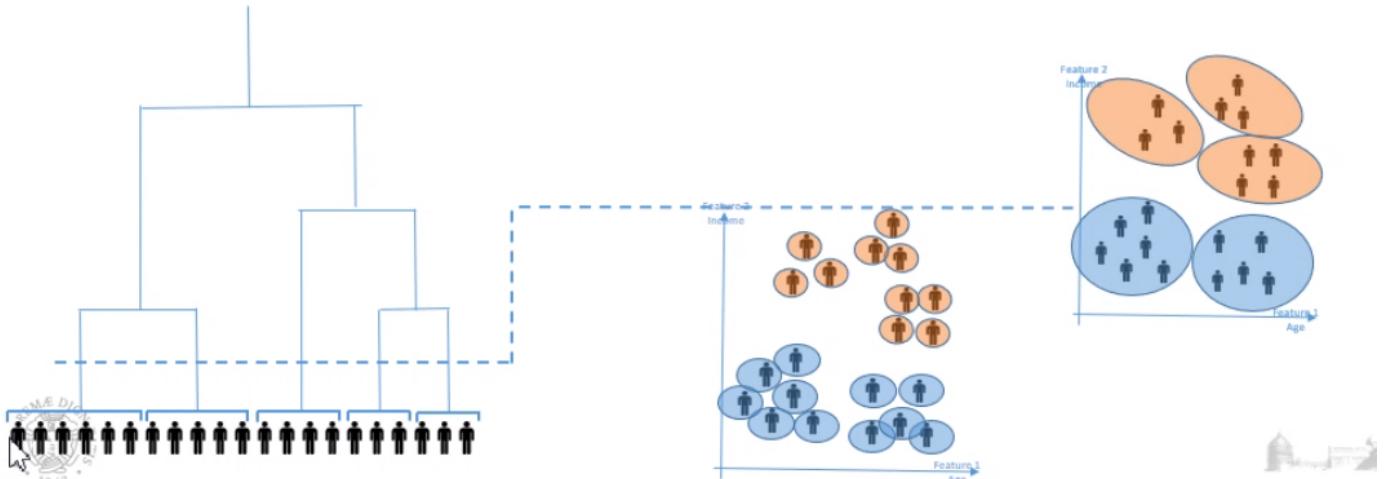
La gerarchia di cluster si rappresenta tramite una struttura dati detta **dendogramma**:

Si tratta di un diagramma ad albero utile a rappresentare le relazioni tra cluster su diversi livelli.

Hierarchical Clustering: Agglomerative

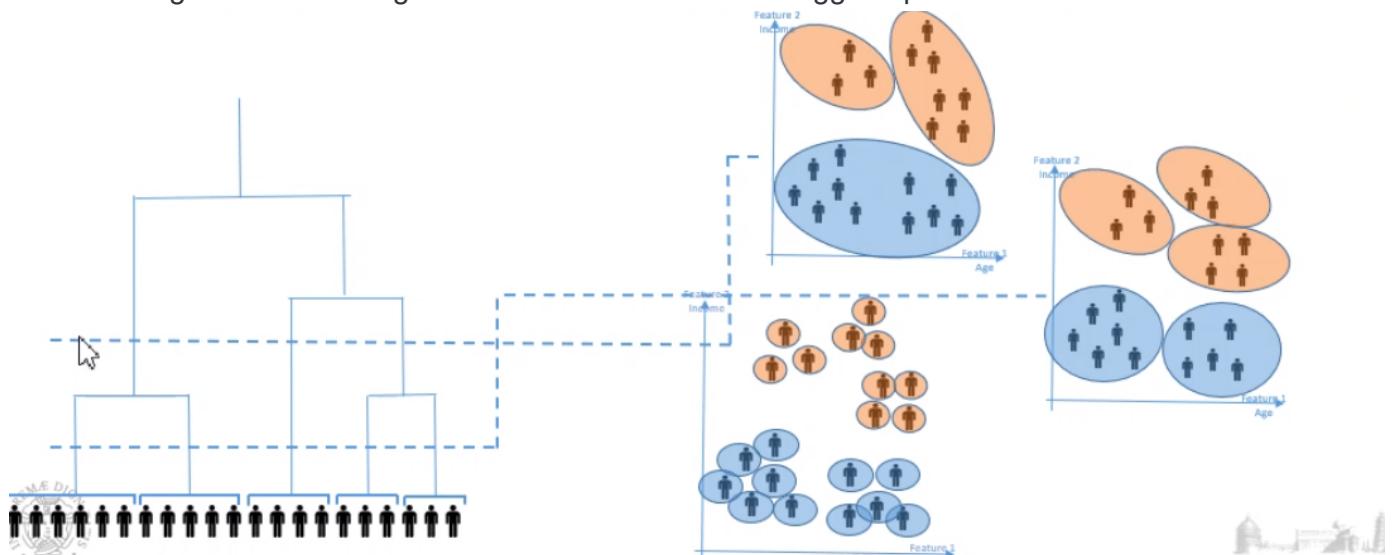
- **Dendogram:** tree diagram used to illustrate the arrangements of the clusters



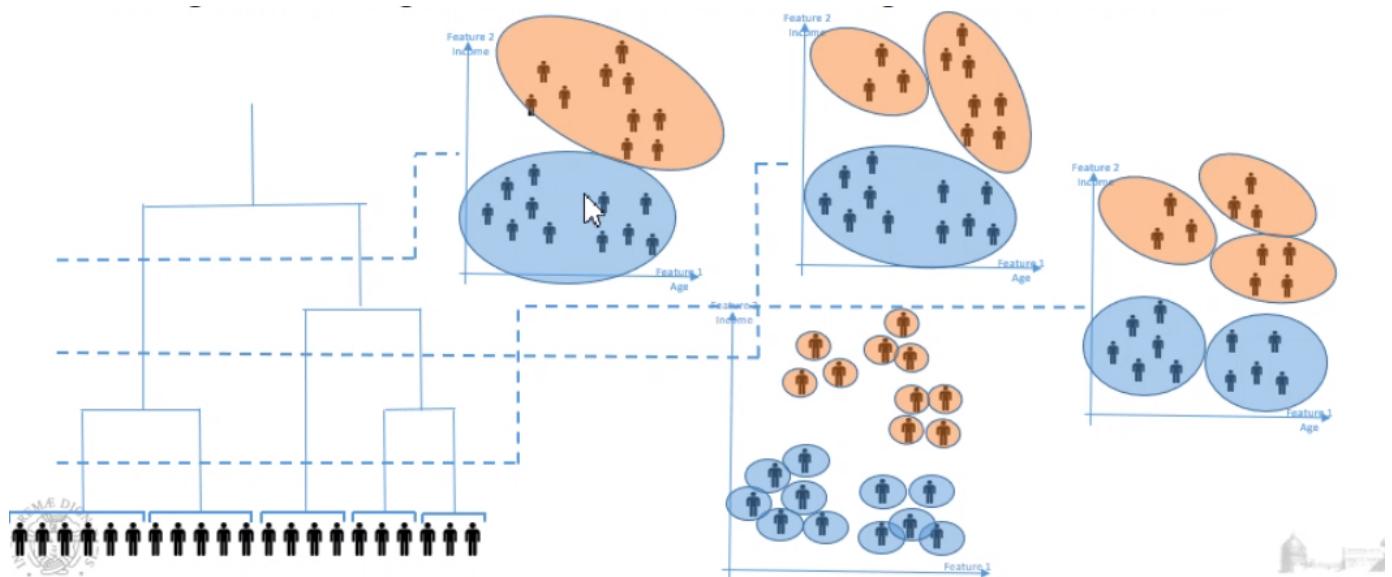


In fondo ci sono tutti gli oggetti dell'insieme di dati e fa vedere come questi oggetti vengono raggruppati nell'insieme di cluster, se tagliassimo il dendogramma lungo la linea tratteggiata otterremmo i cluster in figura:

se invece tagliassimo il dendogramma nella nuova linea tratteggiata più in alto:



otterremmo un unico cluster che racchiude quelli celesti precedenti mentre otterremo 2 cluster arancioni e così via:



Questo dendogramma rappresenta tutte le relazioni che esistono tra cluster più grandi e più piccoli. Questo ci dà parecchie informazioni su come si organizzano gli oggetti facendoci capire se abbiamo oggetti simili tra di loro oppure no.

La cosa interessante negli algoritmi gerarchici è che non abbiamo parametri particolari da settare. L'altro aspetto da considererare è che in ogni caso i cluster che vengono generati tagliando il dendogramma sono cluster sferici.

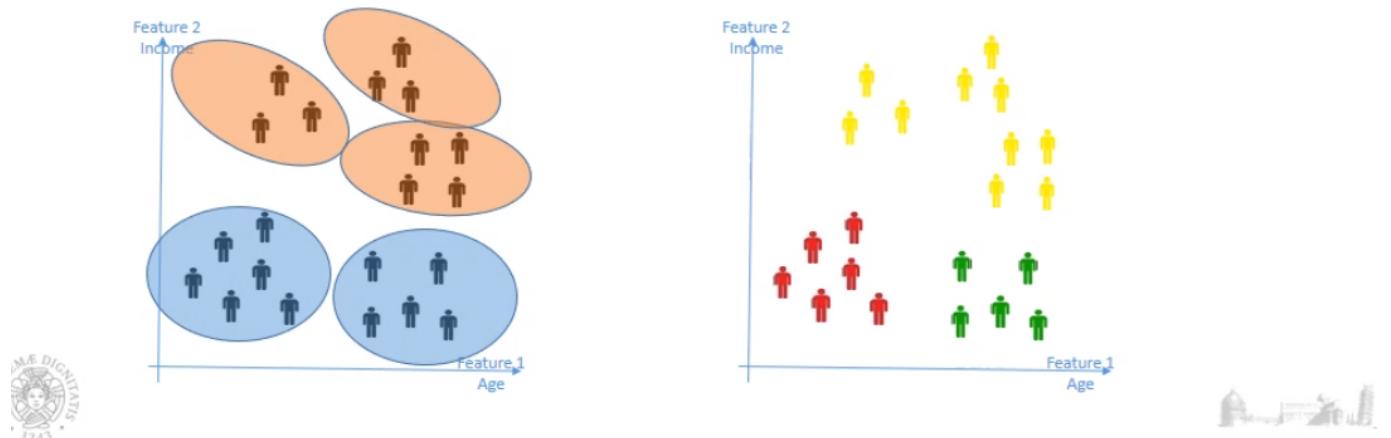
❓ Come possiamo valutare il risultato prodotto da un algoritmo di clustering?

Quando abbiamo parlato della classificazione abbiamo introdotto delle metriche per confrontare gli algoritmi tra loro. Vorremo anche qui un qualcosa di analogo. La complicazione rispetto al caso della classificazione è che qui non abbiamo una ground truth, ovvero nella classificazione sapevamo esattamente cosa ottenere. Con problemi di clustering non sappiamo quale possa essere il risultato migliore. Non sappiamo quali oggetti dovrebbero essere messi insieme.

Il metodo **estrinseco** prevedono di avere una ground truth. Vogliamo sperimentare degli algoritmi di clustering. Possiamo prendere degli insiemi di dati che sono etichettati, di cui conosciamo se appartengono alla stessa classe e dunque dovranno appartenere allo stesso cluster. Sfruttiamo l'informazione della classificazione per valutare se il nostro algoritmo di clustering sta lavorando bene, vedendo se i punti che appartengono alla stessa classe vengono inseriti nello stesso cluster. Se questo accade significa che l'algoritmo sta lavorando bene.

Clustering: Which metrics?

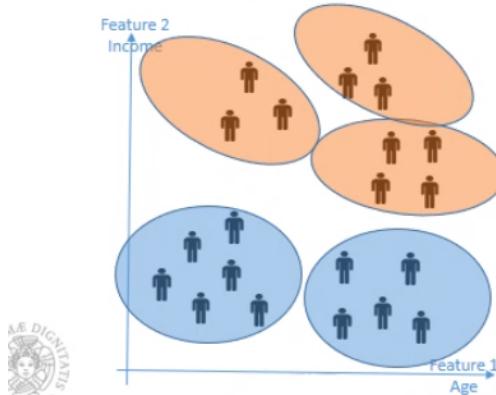
- Extrinsic vs. intrinsic methods
 - **Extrinsic: supervised, i.e. the ground truth is available**
 - Compare results achieved by a clustering algorithm against the ground truth



Quando affrontiamo però un problema reale nessuno ci dà delle label. Per valutare nel caso reale algoritmi di clustering non possiamo sfruttare la ground truth ma possiamo individuare delle metriche che descrivono in numeri cosa dovrebbe essere un buon cluster:

Clustering: Which metrics?

- Extrinsic vs. intrinsic methods
 - Intrinsic: unsupervised, i.e. the ground truth is unavailable
 - Evaluate the goodness of a clustering by considering how well the clusters are separated, and how compact the clusters are (Silhouette coefficient)



$$a(\mathbf{o}) = \frac{\sum_{\mathbf{o}' \in C_i, \mathbf{o} \neq \mathbf{o}'} dist(\mathbf{o}, \mathbf{o}')}{|C_i| - 1}$$

$$b(\mathbf{o}) = \min_{C_j; 1 \leq j \leq k, j \neq i} \left\{ \frac{\sum_{\mathbf{o}' \in C_j} dist(\mathbf{o}, \mathbf{o}')}{|C_j|} \right\}$$

$$s(\mathbf{o}) = \frac{b(\mathbf{o}) - a(\mathbf{o})}{\max\{a(\mathbf{o}), b(\mathbf{o})\}}$$

dovrebbe essere compatto e il più possibile separato dagli altri cluster dunque le metriche **intrinseche** usano il concetto stesso di cluster.

Una metrica proposta è il **coefficiente Silhouette (CS)**. Questo viene calcolato per ogni oggetto, si calcola poi la media del coefficiente calcolato per ogni oggetto su tutti gli oggetti dell'insieme di dati. Se b è molto più grande di a , abbiamo che il massimo tra a e b è b e quindi il CS è circa uguale a 1. Se a è molto più grande di b , $b - a$ è negativo e quindi il CS è circa uguale a -1. **Questo CS varia dunque tra -1 e 1.**

a è il valor medio delle distanze calcolate tra l'oggetto \mathbf{o} che sto considerando e tutti gli oggetti \mathbf{o}' che appartengono allo stesso cluster di \mathbf{o} .

? Perchè nella formula di **a**, al denominatore, viene sottratto alla cardinalità un 1?

Perchè non stiamo considerando \mathbf{o} stesso.

? **a** (\mathbf{o}) cosa ci rappresenta?

Rappresenta la **compattezza** del cluster in quanto se **a** è piccolo tutti gli oggetti che appartengono allo stesso cluster di \mathbf{o} sono vicini ad \mathbf{o} .

? **b** (\mathbf{o}) cosa ci rappresenta?

Dentro le parentesi graffe di **b** c'è la media delle distanze tra l'oggetto \mathbf{o} e un oggetto \mathbf{o}' che appartengono a un cluster C_j che è differente dal cluster a cui appartiene \mathbf{o} .

Cosa ci sta valutando? Valuta la **separazione** tra il cluster a cui appartiene \mathbf{o} e il cluster più vicino.

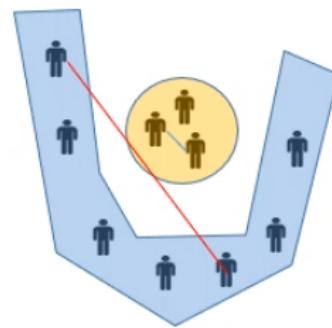
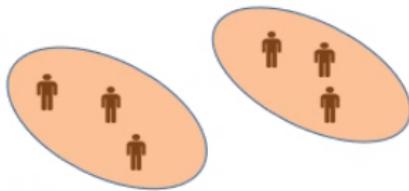
Se **a** è molto più piccolo di **b**, il cluster a cui \mathbf{o} appartiene è molto compatto e ben separato dagli altri cluster quindi quando **s (\mathbf{o})** tende a 1 siamo nella situazione migliore.

Nel caso opposto in cui invece **s (\mathbf{o})** tende a -1, accade che il valore di **a** in realtà è molto grande, vuol dire che **a** non è compatto e **b** avendo un valore piccolo ci indica che qualche cluster non è il

cluster a cui c_i appartiene ma che in realtà ha punti molto vicini ad c_j . $s(\text{c}_i)$ che tende a -1 dunque ci indica che quel cluster non è una buona partizione.

Il CS è adeguato e funziona bene quando possiamo assumere che tutti i cluster siano convessi.

Is the Silhouette coefficient appropriate?



Convex versus Concave

Se non lo sono non funziona più bene. Se i cluster non sono convessi, in questo caso se andiamo a considerare quello celeste che è **concavo**, mediamente la distanza tra un oggetto nel cluster celeste e quelli nel cluster stesso è mediamente più alta della distanza media tra l'oggetto e quelli nel cluster giallo.

⚠ Questo deriva dal fatto che il cluster che stiamo considerando è concavo e non convesso!
Sfortunatamente non esistono metriche adeguate per queste situazioni.

Questi coefficienti che si basano su compattezza e separazione funzionano bene per algoritmi di clustering come il K-means ovvero che producono cluster convessi, non funzionano bene ad esempio con il DBSCAN che può produrre cluster concavi.

Graph Clustering

Graph Clustering

- **Community detection:** objects are linked to each other
 - **vertices are individuals or organizations, and the links are interdependencies between the vertices,** representing friendship, common interests, or collaborative activities



L'analisi dei grafi permette di rappresentare le relazioni che esistono nelle reti sociali, rappresentando attraverso grafi, potremmo rappresentare le persone come nodi e le loro relazioni sociali con degli archi trovando se esistono dei gruppi di nodi che sono fortemente connessi.

Graph Clustering

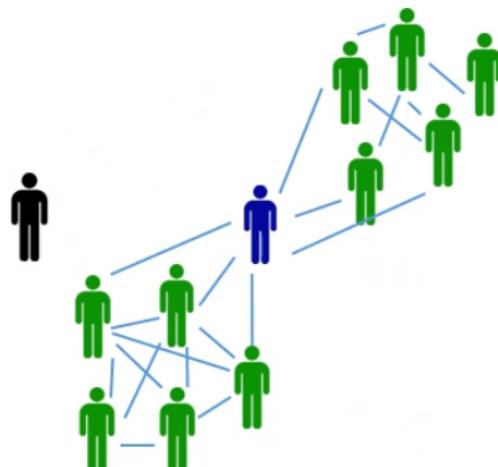
- **Objective**
 - Determine groups of nodes strongly connected among them and weakly connected to the others
 - Groups identify for instance individuals with common interests or special relationship (families, cliques, terrorist cells)



Graph Clustering

- **Cliques, hubs and outliers**

- Individuals in a tight social group, or **clique**, know many of the same people, regardless of the size of the group
- Individuals who are **hubs** know many people in different groups but belong to no single group. Politicians, for example bridge multiple groups
- Individuals who are **outliers** reside at the margins of society. Hermits, for example, know few people and belong to no group



Esistono tipicamente 3 tipologie di nodi:

- **clique**: nodi effettivamente all'interno di cluster
- **hub**: oppure nodi che appartengono a tanti gruppi ma che non appartengono a nessuno, indicati in **blu** in figura
- **outlier**: nodi che risiedono ai margini delle relazioni

Graph Clustering

- Scan algorithm

- Structural similarity

$$\sigma(v, w) = \frac{|\Gamma(v) \cap \Gamma(w)|}{\sqrt{|\Gamma(v)| |\Gamma(w)|}}$$

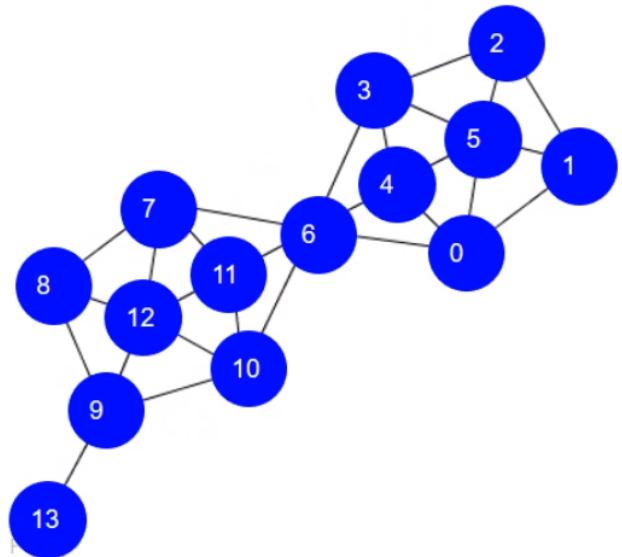
$$\Gamma(u) = \{v | (u, v) \in E\} \cup \{u\}$$

- the ϵ -Neighborhood of v is defined as:

$$N_\epsilon(v) = \{w \in \Gamma(v) | \sigma(v, w) \geq \epsilon\}$$

- v is a core vertex if and only if

$$CORE_{\epsilon, \mu}(v) \Leftrightarrow |N_\epsilon(v)| \geq \mu$$



Per determinare clique, hub e outlier esistono 2 tipi di algoritmi:

- quelli che si basano sulla seguente considerazione: si introduce una funzione di somiglianza tra nodi applicando poi algoritmi di clustering tradizionali.
- definiscono un algoritmo basandosi direttamente sul grafo come ad esempio lo **Scan algorithm**. Questo algoritmo è simile al DBSCAN ma ovviamente la densità non si misura con ϵ e $minpts$

ma fissando quanto un nodo sia collegato con altri nodi.

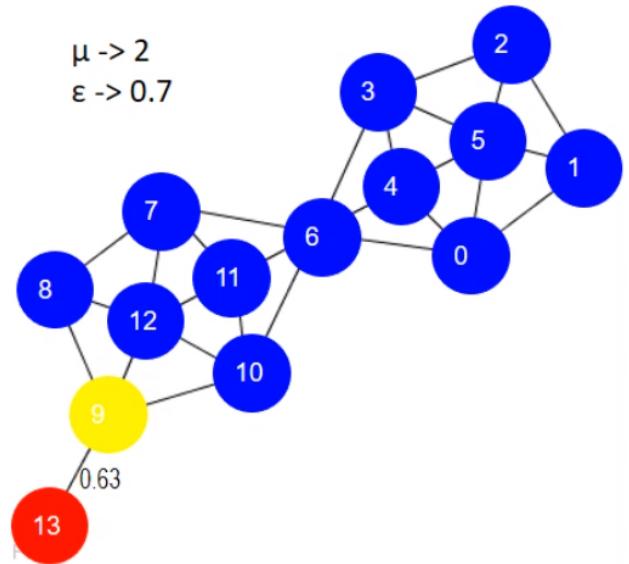
Graph Clustering

- Scan algorithm
 - Structural similarity is large for members of a clique and small for hubs and outliers
 - Two parameters:
 - μ and ϵ

$$\Gamma(13) = \{9\} \cup \{13\}$$

$$\Gamma(9) = \{13, 8, 12, 10\} \cup \{9\}$$

$$\sigma(9,13) = \frac{|\{13, 8, 12, 10, 9\} \cap \{9, 13\}|}{\sqrt{|\{13, 8, 12, 10, 9\}| |\{9, 13\}|}} = \frac{2}{\sqrt{10}} = 0.63$$

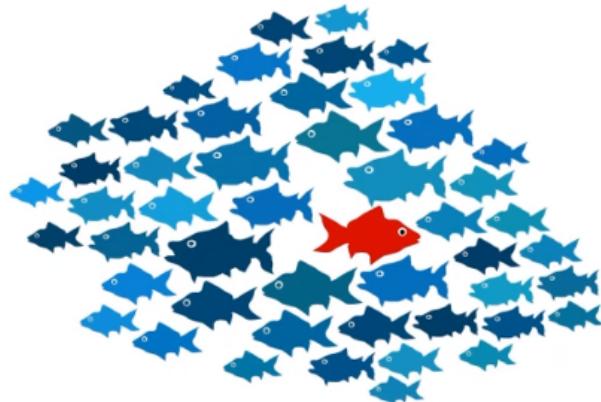


15/04/2023

Un outlier è un oggetto che devia significativamente dagli altri oggetti che possiamo definire normali.

What are outliers?

- **Outliers:** Data objects that deviate significantly from the normal objects as if it were generated by a different mechanism
 - Unusual credit card purchase, anomalous values collected from sensors on board an equipment, Cristiano Ronaldo, ...
 - Outlier detection vs. novelty detection: early stage, outlier, but later merged into the model

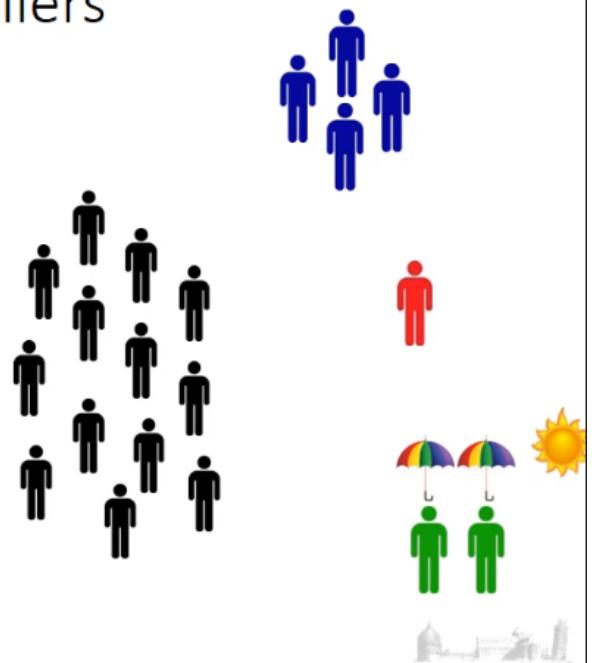


Supponiamo di dover gestire gli acquisti che vengono fatti attraverso le carte di credito da un certo numero di persone e vorremmo capire se sta avvenendo un determinato acquisto anomalo.

L'individuazione di possibili anomalie di questi outlier è qualcosa di rilevante in ambito cybersecurity.

Types of Outliers

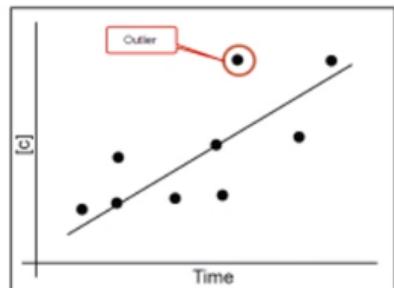
- **Global outlier:** Object significantly deviates from the rest of the other objects
 - Ex. Intrusion detection in computer networks
- **Contextual outlier:** Object significantly deviates from the rest of the other objects based on a selected context
 - 10^0 C in Pisa (outlier in July, normal in March)
- **Collective outlier:** a subset of data objects collectively deviates from the whole data set, even if the individual data objects may not be outliers



- **Global outlier:** oggetto molto differente rispetto agli altri oggetti. L'oggetto rosso è distante da tutti gli altri oggetti. Con tecniche di individuazione degli outlier globali potremmo vedere se c'è un qualche tentativo di intrusione in una rete di computer ad esempio.
- **Contextual outlier:** rispetto a quello globale è un outlier che accade in determinate condizioni. Se parliamo di una temperatura di 10° a Pisa, questa in un periodo invernale è normale ma in un periodo estivo è un outlier relativo al contesto estivo. La vera difficoltà nell'individuarli consiste nel definire le variabili che descrivono il contesto e il comportamento.
- **Collective outlier:** non abbiamo un singolo oggetto ma abbiamo tipicamente un piccolo gruppo di oggetti che deviano significativamente dal comportamento degli altri oggetti. Determinarli non è semplice in quanto alcune tecniche non possono essere usate. Se volessimo usare un algoritmo di clustering come il DBSCAN, se abbiamo un piccolo gruppo di oggetti quel piccolo gruppo viene individuato come cluster e non come outliers.

Challenges of Outlier Detection

- Modeling normal objects and outliers properly
 - Hard to enumerate all possible normal behaviors in an application
 - The border between normal and outlier objects is often a gray area
- Application-specific outlier detection
 - Choice of distance measure among objects and the model of relationship among objects are often application-dependent
 - E.g., clinic data: a small deviation could be an outlier; while in marketing analysis, larger fluctuations



Cerchiamo di modellare come sono gli oggetti normali e poi andiamo a vedere quanto ogni oggetto devia dal comportamento normale. Nella slide abbiamo un certo numero di oggetti i quali possono essere modellati attraverso un approssimazione lineare ai minimi quadrati. Questa linea è il modello che ci rappresenta gli oggetti normali. Se calcolando la distanza di tutti gli oggetti rispetto a questa linea vediamo che esistono oggetti molto distanti da questa linea, allora questi saranno degli outlier. Questi sono gli approcci che vengono normalmente usati, qui vengono usati delle linee come approssimanti.

⚠ Andando a individuare un outlier in questi termini devo andare a definire una soglia ovvero quando effettivamente se un valore supera la soglia viene considerato un outlier. Non è semplice determinare una soglia se non ho grandi conoscenze del dominio nel quale mi muovo.

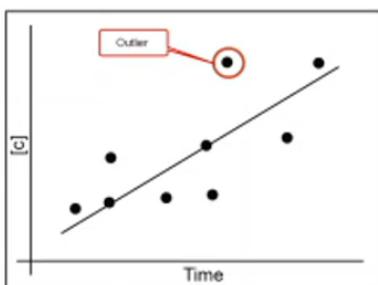
❓ Come possiamo distinguere un outlier dal rumore?

University of Pisa

Francesco Marcellon

Challenges of Outlier Detection

- Handling noise in outlier detection
 - Noise may distort the normal objects and blur the distinction between normal objects and outliers. It may help hide outliers and reduce the effectiveness of outlier detection
- Understandability
 - Understand why these are outliers: Justification of the detection
 - Specify the degree of an outlier: the unlikelihood of the object being generated by a normal mechanism



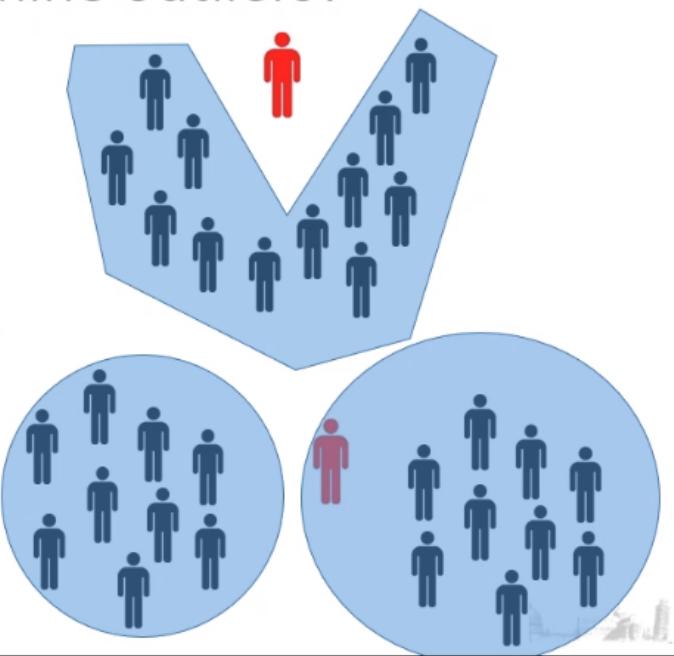
Se siamo interessati a un outlier detection, applicare un filtro di smoothing può impattare quello che abbiamo all'interno del segnale stesso. Applicare dei filtri di smoothing dunque potrebbe non farci individuare correttamente degli outlier.

! Inoltre è importante poter giustificare perché uno specifico oggetto è un outlier.

How to determine outliers?

- **Clustering-based approaches**

- An object is a global outlier if does not belong to any cluster
 - **Apply for instance DBSCAN:** objects which are not within a cluster are outliers
- An object is a global outlier if there is a large distance between the object and its closest cluster c
 - **Apply for instance k-means and partition objects into clusters.** If $\text{dist}(o,c)/\text{avg_dist}(c)$ is large, likely o is an outlier



Commenteremo un paio di tecniche di cui una si basa sulle distanze calcolando la prossimità di punti vicini al punto che stiamo analizzando mentre l'altra è basata sulle tecniche di clustering.

Se applico un algoritmo di clustering, questo raggruppa oggetti che sono simili, dunque un oggetto che non viene inserito in nessun cluster di default è un outlier in quanto non è simile ad altri oggetti raggruppabili. L'idea è quindi quella di andare ad applicare il DBSCAN. Questa tecnica ha un problema fondamentale: la scelta di epsilon e min points. Se non scegliamo questi 2 parametri in modo corretto potremmo avere il problema di tanti outlier o di pochi outlier. Questo vuol dire che l'individuazione dell'outlier dipende fortemente dalla scelta. L'altro problema che abbiamo quando usiamo il DBSCAN è l'avere densità diverse all'interno dello spazio di ricerca. Se basiamo la scelta dei parametri sulla base della zona densa rischiamo nella zona meno densa di avere molti outlier, e viceversa.

In realtà si può anche usare un altro algoritmo di clustering che è il K-means. Quando lo abbiamo descritto abbiamo detto che per poterlo eseguire richiede di fissare il numero K di cluster che vogliamo ottenere. Una volta fissato K tutti i punti vengono inseriti. Di default tutti gli oggetti vengono posizionati in un cluster. Come facciamo allora a individuare se ho degli outlier? Si calcola la distanza tra ogni oggetto e il prototipo del cluster e confrontare la distanza media tra gli oggetti e il prototipo del cluster. Se un oggetto ha una media molto diversa dalla media delle distanze degli altri oggetti quell'oggetto probabilmente era un outlier. E' finito in un cluster perchè è così che funziona il K-means, ma sarebbe dovuto essere individuato come outlier.

Questi 2 approcci sono esempi di approcci basati su algoritmi di clustering. Quindi vado a usare degli algoritmi che nascono per tutt'altro scopo.

Questi approcci in qualche modo soffrono del problema menzionato prima ovvero che potremmo avere zone a densità differenti. Nell'esempio nell'immagine abbiamo zone di punti con densità differenti:

How to determine outliers?

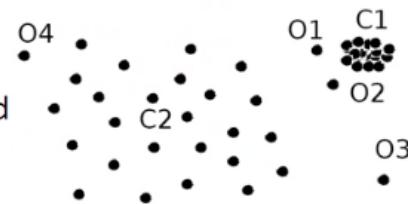
- **Density-based approaches**

- **Local outliers**

- Outliers comparing to their local neighborhoods, instead of the global data distribution
 - In Fig., o1 and o2 are local outliers to C1, o3 is a global outlier, but o4 is not an outlier. However, proximity-based clustering cannot find that o1 and o2 are outliers (e.g., comparing with O4).

- **Intuition: The density around an outlier object is significantly different from the density around its neighbors**

- **Method: Use the relative density of an object against its neighbors as the indicator of the degree of the object being outlier**



l'oggetto O3 appare come outlier in quanto è piuttosto distante dal cluster C2 e dal cluster C1, dunque sembra essere un punto molto isolato e potrebbe essere un outlier globale. O1 e O2 non sono molto distanti dal cluster C1 ma se isoliamo la zona dello spazio in cui abbiamo O1 e O2 sono in realtà degli outlier in quanto la zona ha una densità di punti molto alta e quindi O1 e O2 sono degli outlier locali in quanto diventano outlier quando cominciamo ad analizzare la zona in cui si trovano. O1 e O2 hanno una distanza da C1 simile a quella di O4 con C2. Se basassimo le nostre considerazioni dunque semplicemente sulla distanza dovremmo dedurre che anche O4 è un outlier. Ma O4 intuitivamente non sembra un outlier perché è in una zona a bassa densità ovvero O4 sembra parte del cluster C2. Quando individuiamo gli outlier dovremmo pensare a cosa c'è intorno al punto che stiamo osservando.

La densità intorno a un outlier è significativamente diversa rispetto a quella dei suoi vicini. L'intuizione che ci consente di risolvere il problema descritto prima è quella di provare a considerare la densità intorno al punto che stiamo considerando rispetto alla densità intorno ai punti a lui vicini.

Nel caso di O1 la densità che ho intorno a O1 è bassa in quanto i punti più vicini sono in C1. Se considero invece i vicini di O1, la densità per quei punti è molto alta dunque O1 è un outlier in quanto la densità intorno a O1 è differente rispetto alla densità dei punti che sono in C1. O4 ha una densità simile alla densità dei suoi vicini ovvero O4 si trova in una zona a densità bassa che però è comune in quella zona dunque O4 *non è effettivamente un outlier*. Questa condizione ci consente di gestire spazi con densità differenti.

❓ Come implementarla a livello matematico?

Dobbiamo trovare il modo di lavorare con densità relative al singolo oggetto e confrontare queste densità con quelle dei suoi vicini.

How to determine outliers?

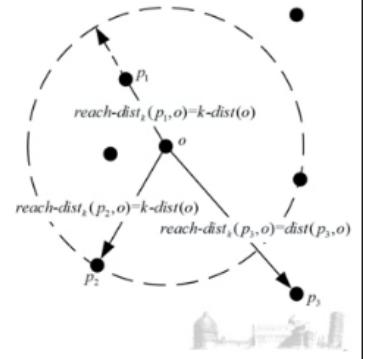
- **Density-based approaches**

- **k-distance of an object o , $\text{dist}_k(o)$:** distance between o and its k -th NN
- **k-distance neighborhood of o , $N_k(o) = \{o' \mid o' \text{ in } D, \text{dist}(o, o') \leq \text{dist}_k(o)\}$**
- $N_k(o)$ could be bigger than k since multiple objects may have identical distance to o
- **Reachability distance from o' to o :**

$$\text{reachdist}_k(o \leftarrow o') = \max\{\text{dist}_k(o), \text{dist}(o, o')\}$$

- **Local reachability density of o**

$$\text{lrd}_k(o) = \frac{\|N_k(o)\|}{\sum_{o' \in N_k(o)} \text{reachdist}_k(o' \leftarrow o)}$$



Il **Local Outlier Factor** ci consente di dire per ogni oggetto quanto questo è probabile che sia un outlier.

Introduciamo il concetto di **K-distance** ovvero la distanza tra l'oggetto \square e il k -esimo oggetto più vicino.

Inoltre introduciamo il concetto di **vicinato K-distance di o** ovvero tutti i punti \square' tali che \square' è a una distanza da \square inferiore o uguale alla distanza \square .

Questo vicinato può contenere un numero superiore di punti a k in quanto potremmo avere che alcuni punti sono ad uguale distanza. Introduciamo la distanza di raggiungibilità tra \square' e \square come la distanza massima tra $\text{dist}_k(o)$ e la distanza effettiva tra \square e \square' .

Il **Local Reachability Distance** è invece il rapporto tra il numero di oggetti contenuti nel vicinato $N_k(o)$ diviso la somma delle reachability distance per tutti gli oggetti \square' che sono nel vicinato $N_k(o)$.

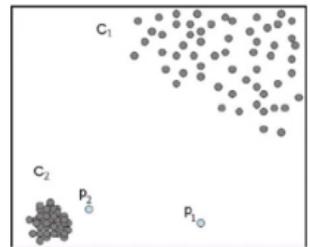
Possiamo definire quello che in letteratura è chiamato il LOF:

How to determine outliers?

- **Density-based approaches**

- **LOF (Local outlier factor)** of an object o is the average of the ratio of local reachability of o and those of o 's k -nearest neighbors

$$LOF_k(o) = \frac{\sum_{o' \in N_k(o)} \frac{lrd_k(o')}{lrd_k(o)}}{\|N_k(o)\|} = \sum_{o' \in N_k(o)} lrd_k(o') \cdot \sum_{o' \in N_k(o)} \text{reachdist}_k(o' \leftarrow o)$$



- The lower the local reachability density of o , and the higher the local reachability density of the k NN of o , the higher LOF
- This captures a local outlier whose local density is relatively low comparing to the local densities of its k NN



corrisponde alla media dei rapporti della local reachability di \square con quelli degli \square' che si trovano nel suo vicinato.

La formula è la sommatoria per tutti gli \square' che si trovano nel vicinato $N_k(o)$ del rapporto tra la local reachability density degli \square' e la local reachability density di \square .

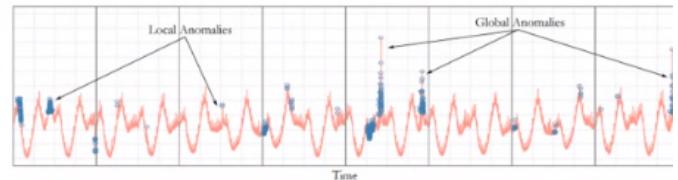
Può accadere che queste densità lrd per gli oggetti nel vicinato e per l'oggetto \square sono simili dunque il LOF è piuttosto basso. Se invece la densità negli \square' è più alta di quella in \square allora il valore di LOF cresce molto. Questa situazione in cui la densità dei vicini è più alta di quella di \square significa che allora \square si trova in una zona che non ha vicino dei punti dunque il suo vicinato è poco denso rispetto a quello che vedono i suoi vicini. **Questo vuol dire che l'oggetto è un outlier.** Se andiamo a vedere la figura nella slide, nel caso di p_1 questo è un outlier senza dubbi, infatti il vicinato di p_1 consiste dei punti in C_2 . La densità di p_1 è molto bassa rispetto alla densità dei punti in C_1 . Ma anche p_2 è un outlier, ovviamente la densità di p_2 è un po più alta rispetto a p_1 ma in ogni caso è molto più bassa rispetto alla lrd che contraddistingue i suoi vicini. Questa formulazione del LOF ci risolve il problema di individuare gli outlier in zone differenti con densità differenti.

L'idea di densità si adatta automaticamente alle zone che stiamo prendendo in considerazione.

! Per ogni punto abbiamo un valore del LOF, decidere se quel punto è un outlier dipende da una **soglia** ma a differenza di prima evitiamo il problema delle zone con densità differenti. Quindi abbiamo si ancora un parametro per decidere se un oggetto è un outlier ma ci siamo liberati dal problema di doverci confrontare con zone a densità differente.

Why outlier detection in industry?

- **Anomaly detection:** anomalous data can be connected to some kind of problem or rare event such as structural defects, malfunctioning equipment
- **Condition-Based Maintenance:** Any machine, whether it is a rotating machine (pump, compressor, gas or steam turbine, etc.) or a non-rotating machine (heat exchanger, distillation column, valve, etc.) will eventually reach a point of poor health. That point might not be that of an actual failure or shutdown, but one at which the equipment is no longer acting in its optimal state.

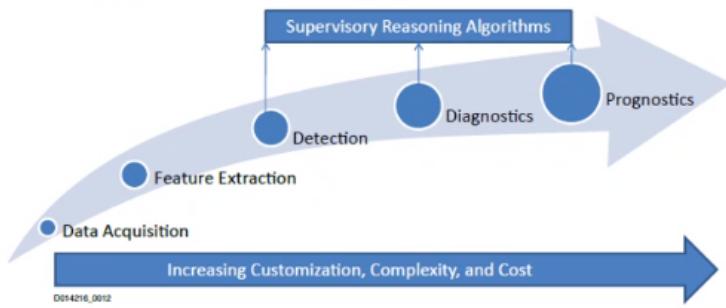


E' interessante in problematiche industriali in quanto consente di vedere se c'è un'anomalia nel funzionamento di macchinari industriali oppure negli accessi a un server oppure nel traffico di rete di un'azienda.

La condition based maintenance: quando si acquista un'auto il concessionario programma delle manutenzioni (il tagliando). Perchè si fa manutenzione programmata? Perchè hanno visto che esistono dei componenti sono soggetti ad usura. A volte questa manutenzione potrebbe non essere necessaria, dunque l'idea di manutenzione basata sulle condizioni è quella di monitorare l'auto per capire quando serve fare manutenzione, ovvero capire se stiamo andando verso un funzionamento pericoloso oppure no e addirittura prevedere possibili anomalie.

Why outlier detection in industry?

- **Prognostic:** predicting the time at which a system or a component will no longer perform its intended function
- Prognostics predicts the future performance of a component by assessing the extent of deviation or degradation of a system from its expected normal operating conditions.



Frequent pattern analysis

Frequent Pattern

- Pattern

- a set of items (milk and bread which appear together in a transaction data set)
- A subsequence (buy first a PC, then a digital camera and then a memory card)
- A substructure refers to different structural forms (subgraphs, subtrees)
- Frequent pattern: a pattern that occurs frequently in a data set



Esistono vari tipi di pattern, quello su cui ci soffermeremo è l'insieme di items. Nel caso di un supermercato sono i prodotti che vengono venduti. Questi insiemi di items sono quello che noi acquistiamo quando andiamo al supermercato. La lista di prodotti che abbiamo acquistato finisce nella transazione. La lista di transazioni corrisponde agli acquisti fatti da un cliente in un supermercato. La lista è un insieme di item, dunque per ogni transazione è un insieme di item acquistati da una persona. Quando parliamo di frequent pattern parliamo di insiemi di items che vengono acquistati dai clienti del supermercato stesso. Il nostro insieme di dati è fatto da tante transazioni in cui ogni transazione corrisponde allo scontrino degli insiemi di dati.

Siamo interessati ad analizzare queste transazioni in quanto vorremmo capire se ci sono insiemi di prodotti che vengono acquistati frequentemente insieme da un buon numero di clienti. Se 2 o più prodotti vengono acquistati insieme da tanti clienti. Se scopro che 3 prodotti vengono acquistati dalla maggior parte dei clienti insieme, posso pensare di fare una campagna pubblicitaria per il primo prodotto, attraiendolo però anche sugli altri 2 prodotti.

In cybersecurity ci aiuta a individuare dei pattern frequenti nel comportamento degli utenti verso specifiche applicazioni che stiamo monitorando dunque possiamo usare queste tecniche per modellare cosa è normalmente frequente. I pattern frequenti di cui abbiamo parlato ora sono insiemi di prodotti o item. Esistono altre tipologie di pattern che sono sequenze. Quando il cliente arriva al supermercato compra una lista di prodotti, l'ordine all'interno di quei prodotti non è significativo perché dipende da come la cassiera prende i prodotti dal nastro. Quindi queste non sono sequenze ma insiemi di prodotti. Se lo stesso cliente va una prima volta al supermercato e compra dei prodotti poi ci ritorna comprando altri prodotti, in questo caso genera una **sequenza**.

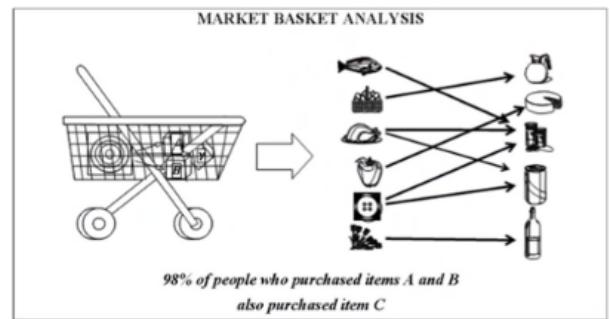
❓ Perchè siamo interessati alle sequenze?

Può accadere che il nostro cliente abbia comprato la prima volta un PC e poi la seconda sia tornato a prendere una videocamera digitale. Se questa sequenza si ripete spesso diventa una sequenza interessante per chi vende, in quanto potrò fare una promozione adeguata per attrarre subito il cliente a comprare il PC + videocamera. Quello che vogliamo individuare sono possibili sequenze **FREQUENTI**.

Ci soffermeremo sugli insiemi di item e andremo a capire come funziona l'algoritmo più conosciuto ovvero l'algoritmo **A-priori**.

Frequent Pattern

- **Motivation: Finding inherent regularities in data**
 - What products were often purchased together? - Beer and diapers?!
 - What are the subsequent purchases after buying a PC?
 - What kinds of DNA are sensitive to this new drug?
 - Can we automatically classify web documents?
- **Applications:** Basket data analysis, cross-marketing, catalog design, sale campaign analysis, Web log (click stream) analysis, and DNA sequence analysis



ci consente di estrarre tutti i pattern frequenti una volta fissata una soglia ovvero fissiamo una soglia per esempio vogliamo estrarre tutti i pattern verificati da almeno il 30% dei clienti. Come i clienti vanno a comprare i prodotti e l'analisi se ci sono insiemi di prodotti frequentemente acquistati dai clienti va sotto il nome di *Market Basket Analysis*.

Dovendo analizzare tutte le possibili combinazioni dei prodotti diventa difficile in quanto le combinazioni esplodono. Non possiamo permetterci di generare tutte le possibili combinazioni di prodotti ma dobbiamo pensare ad un approccio che ci consente di individuare questi pattern in modo efficiente.

A-priori algorithm

Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq
- end



Cart	MILK	BUTTER	COLA	BREAD	STEAK
●	MILK	BUTTER	COLA	BREAD	STEAK
●	MILK	BUTTER	COLA	BREAD	STEAK
●	MILK	BUTTER		BREAD	STEAK
●			COLA	BREAD	STEAK
	MILK		COLA	BREAD	STEAK

Nella parte destra sono rappresentate le transazioni che corrispondono alle righe per ciascun cliente.

Le colonne rappresentano i prodotti. Il terzo cliente non ha acquistato Coca-Cola ad esempio. Con [click](#) indichiamo gli item set che generiamo. Ad esempio vogliamo analizzare latte e burro insieme come item set per capire se è frequente. Possiamo dire che è frequente quando esiste un numero di transazioni in cui latte e burro compaiono insieme. Questo itemset deve comparire in un numero di transazioni superiori ad una **soglia** che viene fissata. Questa soglia in letteratura viene chiamata **minSupp** perché tiene conto del supporto dell'itemset, ovvero dato dal numero di transazioni in cui quell'itemset appare. Come funziona l'algoritmo a priori. Per prima cosa ci si pone il problema di capire quali sono gli 1-itemsets frequenti, ovvero l'insieme di item composto da un unico prodotto:

Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

C_1



- Fix a minimum frequency min_freq.

- Find all the frequent 1-itemsets (L_1)**



- For ($k=1$; L_k is not empty; $k++$) do

begin



- C_{k+1} = candidates generated from L_k

- for each transaction t in database do

- increment the count of all candidates in C_{k+1} that are contained in t



- L_{k+1} = candidates in C_{k+1} with min_freq

end



Dobbiamo individuare quali sono gli item che sono frequenti. Per fare ciò abbiamo il minSupp o anche

detto **min_freq** e lo poniamo a 4. Questo min_sup ci fissa la soglia per cui possiamo dedurre che un itemset è frequente, dunque ha almeno quel numero di transazioni in cui effettivamente appare. Si prendono tutti gli item e si va a vedere se sono frequenti. Bisogna fare una scansione del nostro database delle transazioni e contare in quante transazioni questo 1-itemset appare. Milk appare nella prima nella seconda nella terza e nell'ultima transazione, appare 4 volte e dato che min_freq è 4 risulta essere frequente. Per il burro, non è frequente in quanto $3 < 4$:

Frequent Pattern Analysis

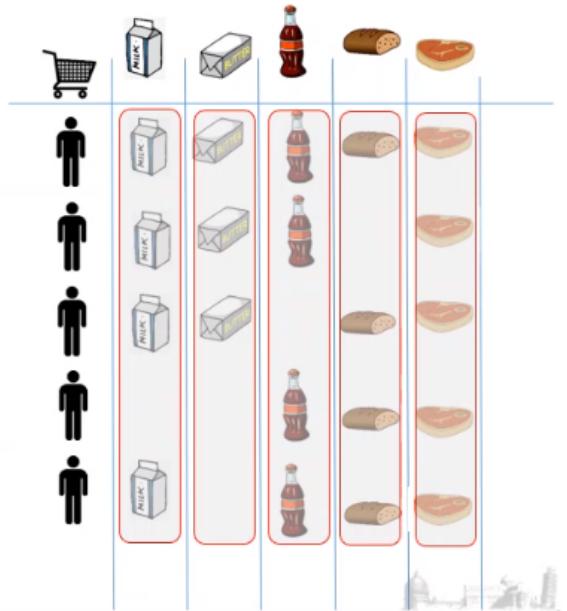
- The A-Priori Algorithm

$\text{min_freq} = 4$

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq .
- Find all the frequent 1-itemsets (L_1)**
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq
 - end



Sia coca cola che pane che carne sono frequenti.

Dopo aver eseguito la prima azione dell'algoritmo abbiamo che L_1 contiene latte cocacola pane e carne:

Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq .
- Find all the frequent 1-itemsets (L_1)**
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq
 - end



C_1	Freq	L_1
	4	
	3	
	4	
	4	
	5	

Ci dice di prendere tutti questi 1-itemset frequenti e combinarli in modo tale da generare tutte le possibili combinazioni di itemset composti da 2 item. Prendo dunque gli item frequenti combinandoli



con altri item. Ad esempio prendo latte e lo combino con coca-cola pane e carne e così via:

Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq
 - end



latte e burro non vengono combinati in quanto burro non è frequente. Ovvero non combino latte con il burro per la proprietà a priori la quale mi dice che: ogni sottoinsieme non vuoto di un item set frequente deve essere frequente.

Why not?



Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq
 - end



Se ho che burro non è frequente è inutile combinarlo in quanto anche combinandolo con un item frequente non posso generare itemset frequenti.

Se per caso generassimo un itemset dobbiamo essere sicuri che tutti i sottoinsiemi di quell'itemset sono frequenti.

Why not?



A-priori property: Any nonempty subset of a frequent itemset must also be frequent



Frequent Pattern Analysis

- The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq

end



Why not?



A-priori property: Any nonempty subset of a frequent itemset must also be frequent



Why not?



In C_2 abbiamo messo latte e cocacola ma non abbiamo messo cocacola e latte. L'ordine non conta, a noi interessa la sola presenza ma non l'ordine. Quando andiamo al supermercato il fatto che latte preceda cocacola è una casualità. Una volta generato i possibili pattern di ordine 2 dobbiamo fare quanto fatto prima con i pattern di ordine 1. Ovvero verificare se esistono transazioni in cui compaiono insieme i 2 item che costituiscono l'itemset di ordine 2. Per ogni transazione vado a verificare tutti i pattern di ordine 2 e la loro frequenza:

Frequent Pattern Analysis

- The A-Priori Algorithm

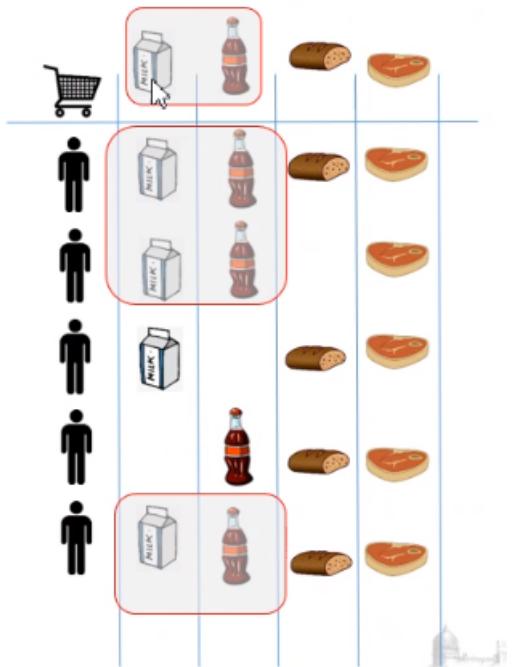
min_freq = 4

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do
 - begin
 - C_{k+1} = candidates generated from L_k
 - for each transaction t in database do
 - increment the count of all candidates in C_{k+1} that are contained in t
 - L_{k+1} = candidates in C_{k+1} with min_freq

end



Frequent Pattern Analysis

The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do


```
begin
        •  $C_{k+1}$  = candidates generated from  $L_k$ 
        • for each transaction t in database do
          • increment the count of all candidates in
             $C_{k+1}$  that are contained in t
        •  $L_{k+1}$  = candidates in  $C_{k+1}$  with min_freq
      end
```



Definisco il contenitore L_2 a seguito delle frequenze calcolate.

Continuo con C_3 :

Frequent Pattern Analysis

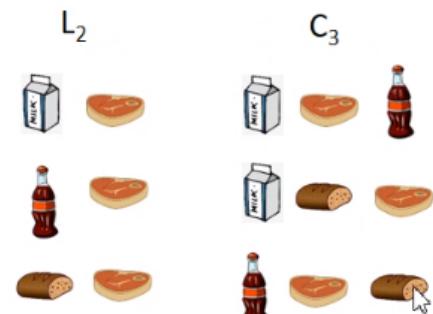
The A-Priori Algorithm

C_k : Candidate itemset of size k

L_k : frequent itemset of size k

- Fix a minimum frequency min_freq.
- Find all the frequent 1-itemsets (L_1)
- For ($k=1$; L_k is not empty; $k++$) do


```
begin
        •  $C_{k+1}$  = candidates generated from  $L_k$ 
        • for each transaction t in database do
          • increment the count of all candidates in
             $C_{k+1}$  that are contained in t
        •  $L_{k+1}$  = candidates in  $C_{k+1}$  with min_freq
      end
```



Do we need to count in the dataset?

Remember the A-priori property!



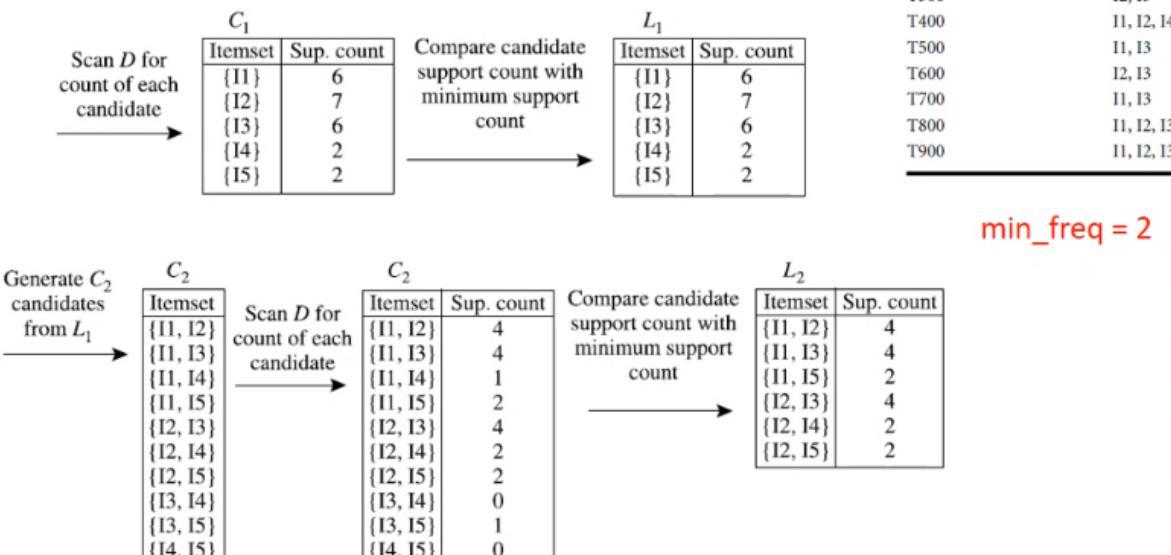
prendiamo latte carne e cocacola, c'è bisogno che io vada a contare in quante transazioni appare oppure posso dire che non è frequente? Posso usare la priorità a priori, in quanto affinchè un itemset sia frequente, devono essere frequenti tutti i sottoinsiemi per poter essere l'intero itemset frequente. Ad esempio l'itemset C_3 (LATTE,CARNE,COCACOLA) non è frequente perché in L_2 non esiste un itemset frequente corrispondente a (LATTE,COCACOLA)

Per comprendere meglio la frequent pattern analysis presentiamo il seguente esempio con una

`min_freq = 2.`

Il TID indica l'ID della transazione presente nel nostro database.

Frequent Pattern Analysis



Frequent Pattern Analysis

- (a) Join: $C_3 = L_2 \bowtie L_2 = \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\}$
- $$\bowtie \{\{I1, I2\}, \{I1, I3\}, \{I1, I5\}, \{I2, I3\}, \{I2, I4\}, \{I2, I5\}\}$$
- $$= \{\{I1, I2, I3\}, \{I1, I2, I5\}, \{I1, I3, I5\}, \{I2, I3, I4\}, \{I2, I3, I5\}, \{I2, I4, I5\}\}.$$
- (b) Prune using the Apriori property: All nonempty subsets of a frequent itemset must also be frequent. Do any of the candidates have a subset that is not frequent?
- The 2-item subsets of $\{I1, I2, I3\}$ are $\{I1, I2\}$, $\{I1, I3\}$, and $\{I2, I3\}$. All 2-item subsets of $\{I1, I2, I3\}$ are members of L_2 . Therefore, keep $\{I1, I2, I3\}$ in C_3 .
 - The 2-item subsets of $\{I1, I2, I5\}$ are $\{I1, I2\}$, $\{I1, I5\}$, and $\{I2, I5\}$. All 2-item subsets of $\{I1, I2, I5\}$ are members of L_2 . Therefore, keep $\{I1, I2, I5\}$ in C_3 .
 - The 2-item subsets of $\{I1, I3, I5\}$ are $\{I1, I3\}$, $\{I1, I5\}$, and $\{I3, I5\}$. $\{I3, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I1, I3, I5\}$ from C_3 .
 - The 2-item subsets of $\{I2, I3, I4\}$ are $\{I2, I3\}$, $\{I2, I4\}$, and $\{I3, I4\}$. $\{I3, I4\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I3, I4\}$ from C_3 .
 - The 2-item subsets of $\{I2, I3, I5\}$ are $\{I2, I3\}$, $\{I2, I5\}$, and $\{I3, I5\}$. $\{I3, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I3, I5\}$ from C_3 .
 - The 2-item subsets of $\{I2, I4, I5\}$ are $\{I2, I4\}$, $\{I2, I5\}$, and $\{I4, I5\}$. $\{I4, I5\}$ is not a member of L_2 , and so it is not frequent. Therefore, remove $\{I2, I4, I5\}$ from C_3 .
- (c) Therefore, $C_3 = \{\{I1, I2, I3\}, \{I1, I2, I5\}\}$ after pruning.

Una volta generati gli itemset con una frequenza di 2 elementi, come possiamo generare i candidati dell'itemset con 3 elementi?

Tramite l'operazione di JOIN: si parte dall'itemset di ordine 2 contenuto nel primo L_2 e si trovano nel secondo L_2 l'itemset di ordine 2 che ha il primo item uguale e il secondo item differente. Con questo approccio genero l'itemset con 3 elementi $I1, I2, I3$ ad esempio.

TID	List of item IDs
T100	I1, I2, I5
T200	I2, I4
T300	I2, I3
T400	I1, I2, I4
T500	I1, I3
T600	I2, I3
T700	I1, I3
T800	I1, I2, I3, I5
T900	I1, I2, I3

$\min_freq = 2$



Frequent Pattern

	L_2	Ass. Rule	Supp	Conf	Lift
• Association rule $(I_1, I_2) \rightarrow I_3$		\Rightarrow	4	1	1.25
• Support (reflects usefulness) Supp $((I_1, I_2) \rightarrow I_3) = freq(I_1, I_2, I_3)$		\Rightarrow	4	0.8	0.8
• Confidence (reflects certainty) Conf $((I_1, I_2) \rightarrow I_3) = \frac{freq(I_1, I_2, I_3)}{freq(I_1, I_2)}$		\Rightarrow	4	1	1.67
• Lift (reflects correlation) Lift $((I_1, I_2) \rightarrow I_3) = \frac{freq(I_1, I_2, I_3)}{freq(I_1, I_2) freq(I_3)}$		\Rightarrow \Rightarrow	4	0.8	0.8



Dunque esiste un insieme di prodotti frequenti. Quando questi pattern sono frequenti posso generare le regole associative. Si considerano un sottoinsieme di item dell'itemset frequente e il restante dell'item sulla destra:

se I_1, I_2, I_3 è un pattern frequente posso dire che $I_1, I_2 \Rightarrow I_3$ ovvero se i clienti acquistano I_1, I_2 allora acquisteranno anche I_3 . Le regole associative come quella indicata poco fa sono caratterizzate da 2 metriche principali: il **supporto** e la **confidenza**.

- Il supporto è il numero di transazioni per cui quella regola è verificata ovvero in altre parole il supporto coincide con la frequenza dell'item set I_1, I_2, I_3 .
- La confidenza invece riflette la certezza della regola ovvero ci dice quanto sia probabile che acquistando I_1 e I_2 il cliente acquisti anche I_3 . Supponendo di avere una confidenza di 1 significa che tutti i clienti che hanno comprato I_1 e I_2 hanno comprato anche I_3 . ⚠️ Avere una confidenza 1 potrebbe non indicarci questa forte dipendenza tra ciò che ho nell'antecedente e ciò che ho nel conseguente. Supponiamo che I_3 sia presente in tutte le transazioni, avremo che la frequenza di I_1, I_2, I_3 è uguale a quella di I_1 e I_2 , perchè I_3 è presente in tutte le transazioni. Abbiamo si la confidenza uguale a 1, ma non perchè acquistando I_1 e I_2 allora si acquista anche I_3 **ma perchè I_3 viene acquistato sempre**.

Per tener conto di questa anomalia della confidenza in letteratura è stata introdotta la metrica **lift** la quale riflette la correlazione ovvero ci dice che antecedente e conseguente sono correlate. La definizione del LIFT può essere espressa in termini di probabilità ed è la probabilità congiunta di I_1 , I_2 e I_3 diviso il prodotto della probabilità di avere l'antecedente per la probabilità di avere il conseguente.

Avere lift uguale 1 significa che antecedente e conseguente sono indipendenti e dunque non c'è

una correlazione tra l'aver acquistato I₁ e I₂ e l'aver acquistato anche I₃.

Frequent Pattern Analysis

Kulczynski ($I_1 \rightarrow I_2$) = $\frac{1}{2} \left(\frac{\text{freq}(I_1, I_2)}{\text{freq}(I_1)} + \frac{\text{freq}(I_1, I_2)}{\text{freq}(I_2)} \right)$

Association rule: $I_1 \rightarrow I_2$

2 × 2 Contingency Table for Two Items

	<i>milk</i>	$\overline{\text{milk}}$	Σ_{row}
<i>coffee</i>	<i>mc</i>	\overline{mc}	<i>c</i>
$\overline{\text{coffee}}$	$\overline{m}\overline{c}$	$\overline{m}\overline{c}$	\overline{c}
Σ_{col}	<i>m</i>	\overline{m}	Σ

Null-transactions w.r.t. m and c

Null-invariant

Data

Set	<i>mc</i>	\overline{mc}	$m\overline{c}$	$\overline{m}\overline{c}$	χ^2	lift	all_conf.	max_conf.	Kulc.	cosine
D_1	10,000	1000	1000	100,000	90557	9.26	0.91	0.91	0.91	0.91
D_2	10,000	1000	1000	100	0	1	0.91	0.91	0.91	0.91
D_3	100	1000	1000	100,000	670	8.44	0.09	0.09	0.09	0.09
D_4	1000	1000	1000	100,000	24740	25.75	0.5	0.5	0.5	0.5
D_5	1000	100	10,000	100,000	8173	9.18	0.09	0.91	0.5	0.29
D_6	1000	10	100,000	100,000	965	1.97	0.01	0.99	0.5	0.10

Subtle: They disagree

Weka e progetto finale

The Weka GUI chooser

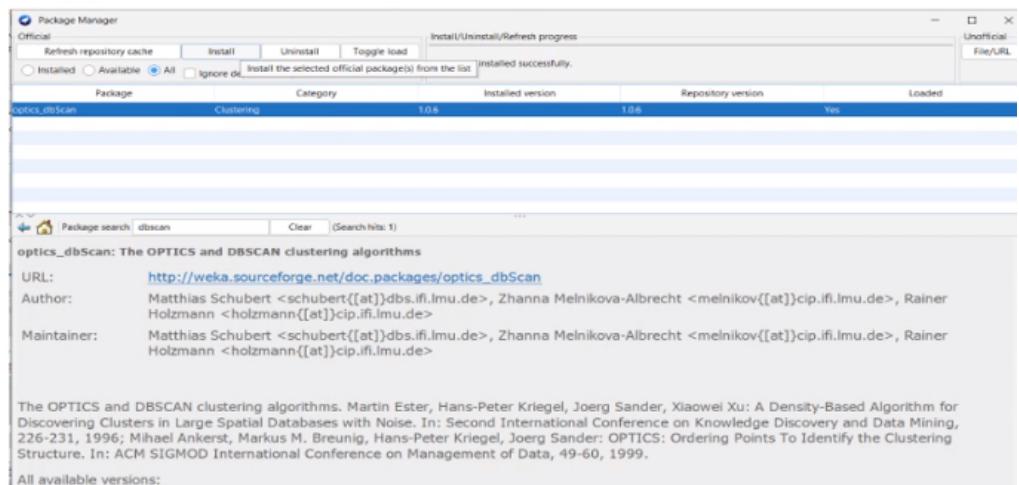


è possibile installare ulteriori algoritmi rispetto a quelli di default preinstallati con Weka:

The Weka GUI chooser

Package Manager:

- Select all
- Search for optics_dbSan
- Install it

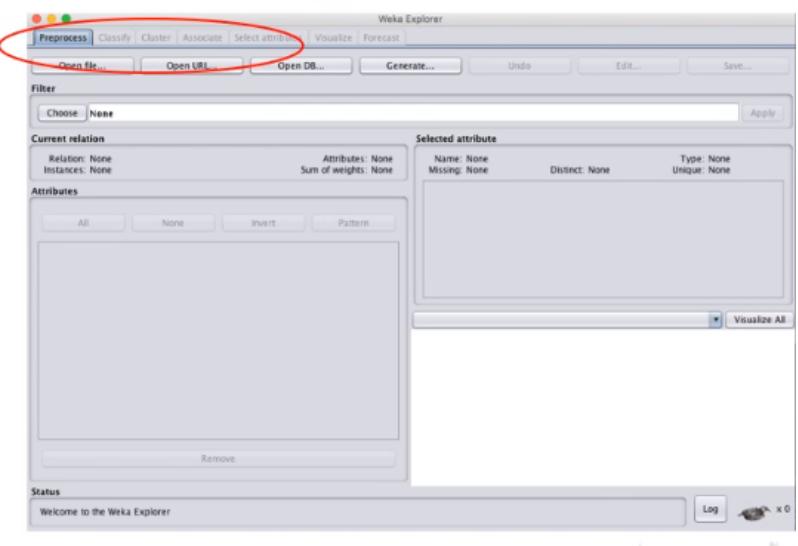


ad esempio possiamo installare il package optics_dbscan e Smote.

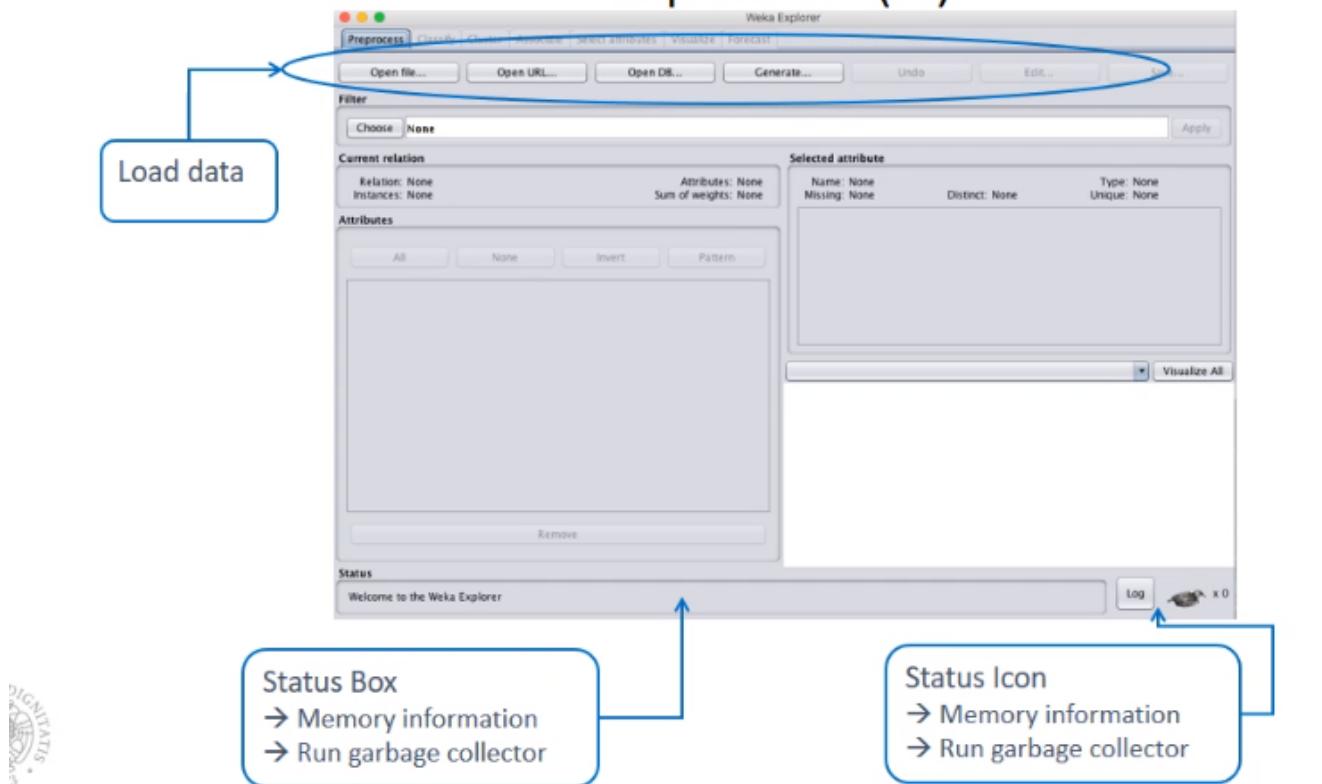
Smote è un algoritmo è un algoritmo per fare ribilanciamento dei dati.

Explorer (1)

- **Preprocess:** Choose and modify the data being acted on.
- **Classify:** Train and test learning schemes that classify or perform regression.
- **Cluster:** Learn clusters for the data.
- **Associate:** Learn association rules for the data.
- **Select attribute:** Select the most relevant attributes in the data.
- **Visualize:** View an interactive 2D plot of the data.



Explorer (2)



Il formato con cui vengono gestiti i dati in weka è il formato ARFF.

Loading Data

- **Open File:** Brings up a dialog box allowing you to browse for the data file on the local file system. You can read files in a variety of formats: WEKA's ARFF format, CSV format, etc.
- **Open URL:** Asks for a Uniform Resource Locator address for where the data is stored.
- **Open DB:** Reads data from a database.
- **Generate:** Enables you to generate artificial data from a variety of DataGenerators.



The .ARFF format

- **ARFF files** have two distinct sections: **Header and Data**.
- The **Header section** contains the name of the relation, a list of the attributes (the columns in the data), and their types.
- The **Data section** starts with the @data declarations and contains all the instances of the dataset .
- The ARFF format has the following types of attributes:
 - **Numeric**
 - **Nominal** (represented by the set of values they can take on, enclosed in curly braces)
 - **String**
 - **Date**



```
*ARFF file for the weather data
@relation weather

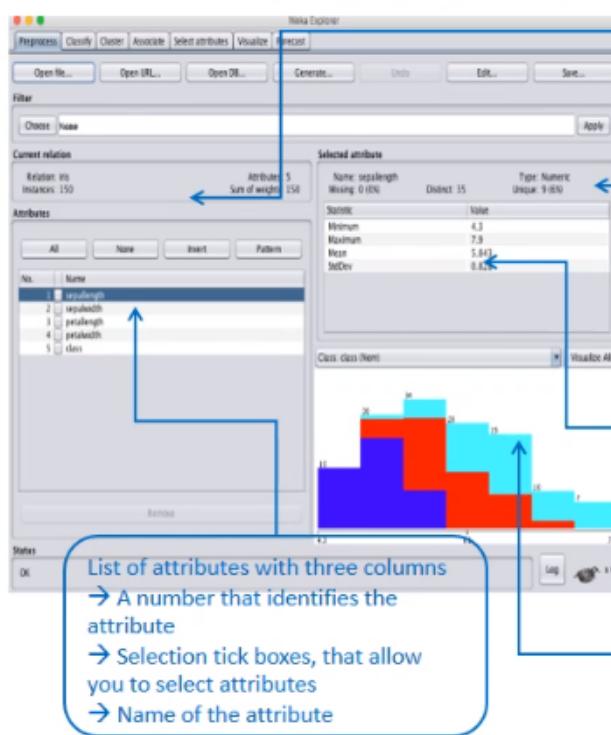
@attribute outlook {sunny, overcast, rainy}
@attribute temperature numeric
@attribute humidity numeric
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}
@
@ 14 instances
@
@data
sunny,85,85, FALSE,no
sunny,80,90, TRUE,no
overcast,83,86, FALSE,yes
rainy,70,96, FALSE,yes
rainy,68,80, FALSE,yes
rainy,65,70, TRUE,no
overcast,64,65, TRUE,yes
sunny,72,95, FALSE,no
sunny,69,70, FALSE,yes
rainy,75,80, FALSE,yes
sunny,75,70, TRUE,yes
overcast,72,90, TRUE,yes
overcast,81,75, FALSE,yes
rainy,71,91, TRUE,no
```



outlook è un attributo nominale ad esempio, mentre per temperature si tratta di un attributo numerico.

Una volta caricato l'insieme di dati compaiono un pò di informazioni per capire come è fatto l'insieme di dati:

Information on the data



Name of the relation, number of instances, number of attributes.

Information on the selected attribute:

- Name and type of attribute
- Number of instances in the data for which this attribute is missing
- Number of different values that the data contains for this attribute
- Number of instances in the data having a value for this attribute that no other instances have.

→ Nominal attributes: the list consists of each possible value for the attribute along with the number of instances that have that value.

→ Numeric attributes: the list gives four statistics describing the distribution of values in the data

List of attributes with three columns

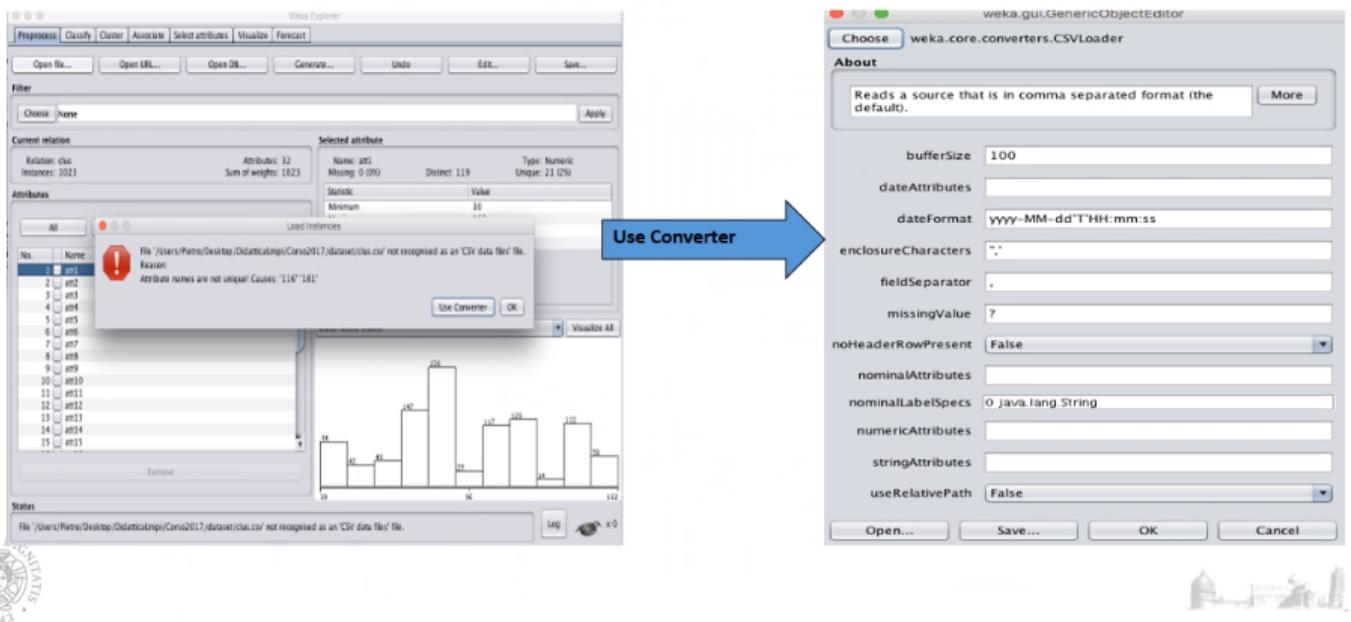
- A number that identifies the attribute
- Selection tick boxes, that allow you to select attributes
- Name of the attribute

Coloured histogram, colour-coded according to the attribute chosen as the Class using the box above the histogram

Converters to ARFF(1)

- ARFF files are not the only format one can load, but all files that can be converted with Weka's "core converters". The following formats are currently supported:
 - C4.5
 - CSV
 - libsvm
 - binary serialized instances
 - XRFF
 - text files in folders
- If Weka cannot load the data, it tries to interpret it as an ARFF. If that fails, it pops up a box from which the user can select the converter.

Converters to ARFF(2)



Nel menu visualizzare è possibile vedere alcune caratterizzazioni del dataset stesso, in particolare quando visualizziamo visualize nel menu in cima compaiono delle visualizzazioni in formato scatter plot del dataset stesso.

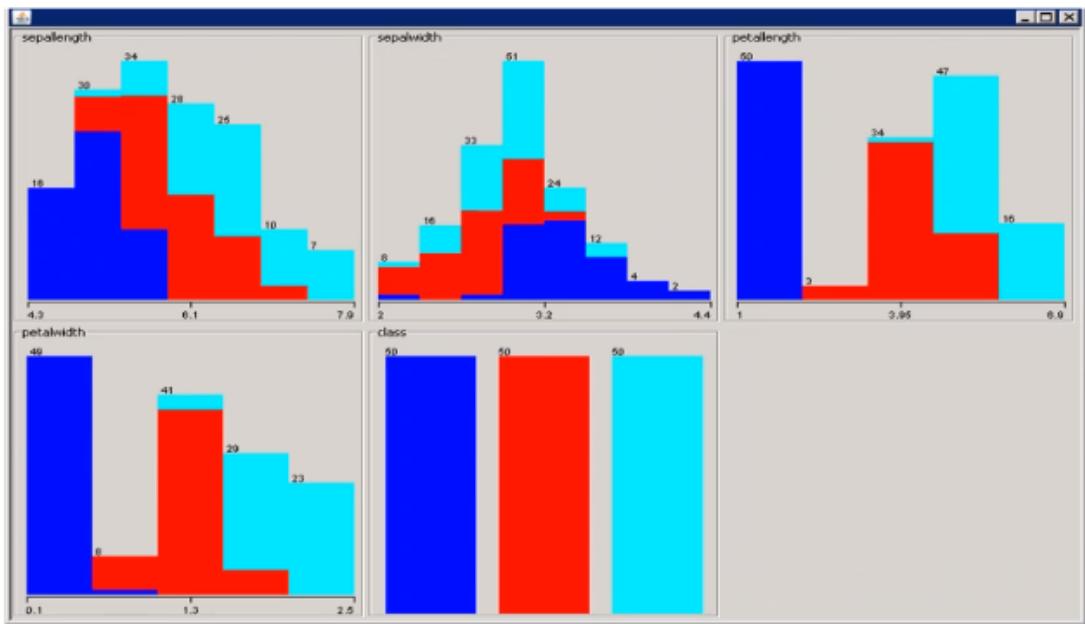
Visualizing Data (Scatter Plot)

- Select the Visualize Section



Visualizing Data (Scatter Plot)

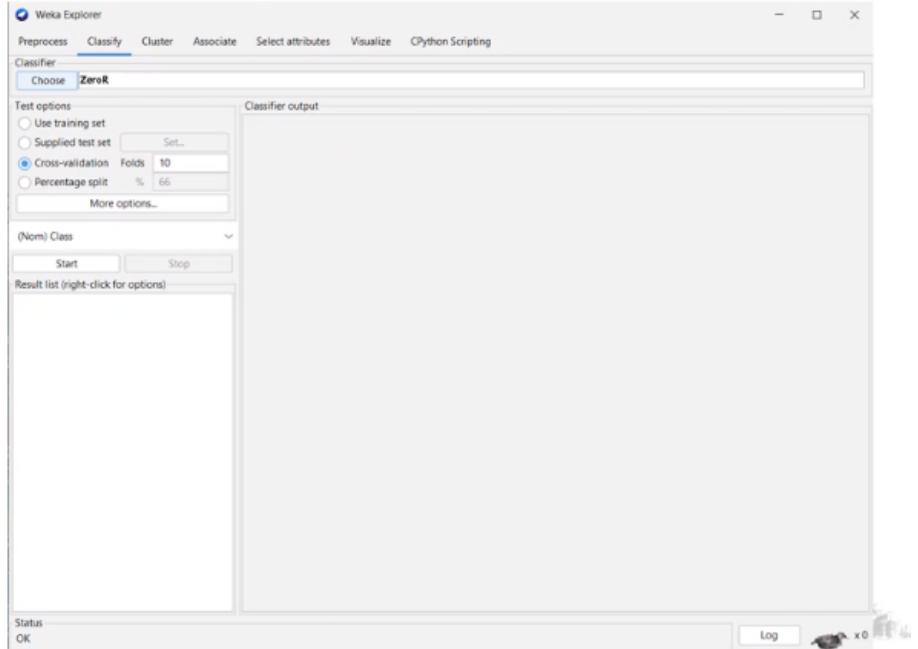
- Click Visualize All



Una volta elaborato e analizzato la distribuzione dei dati possiamo iniziare a lavorare sugli algoritmi:

Classification

- Select Classify
- Push button Choose
- Choose the classifier you desire to employ
- Select cross-validation
- Push button Start

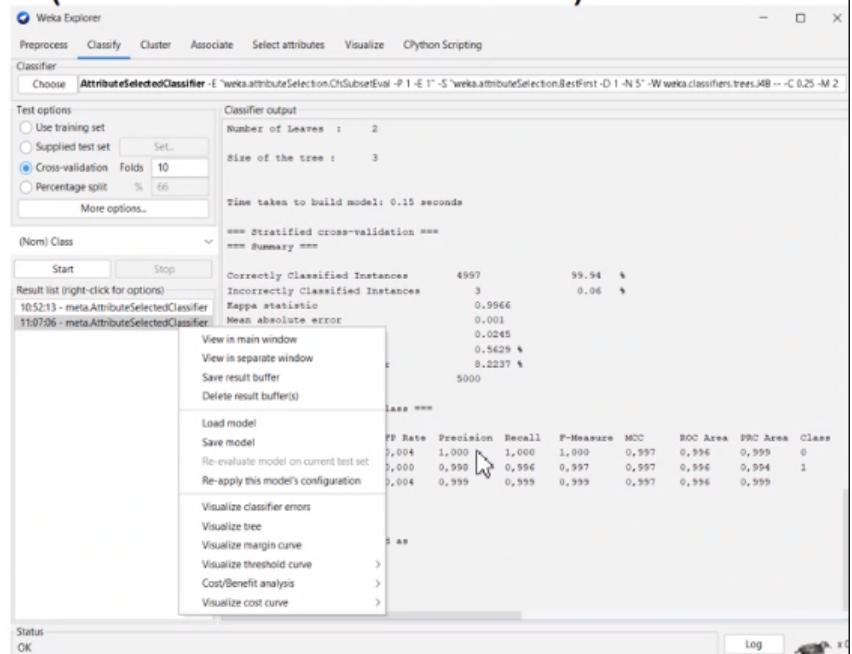


Per eseguire algoritmi di classificazione sul dataset dobbiamo usare il menu Classify.

Noi useremo la cross-validation con un numero di fold pari a 10.

Classification (Result Visualization)

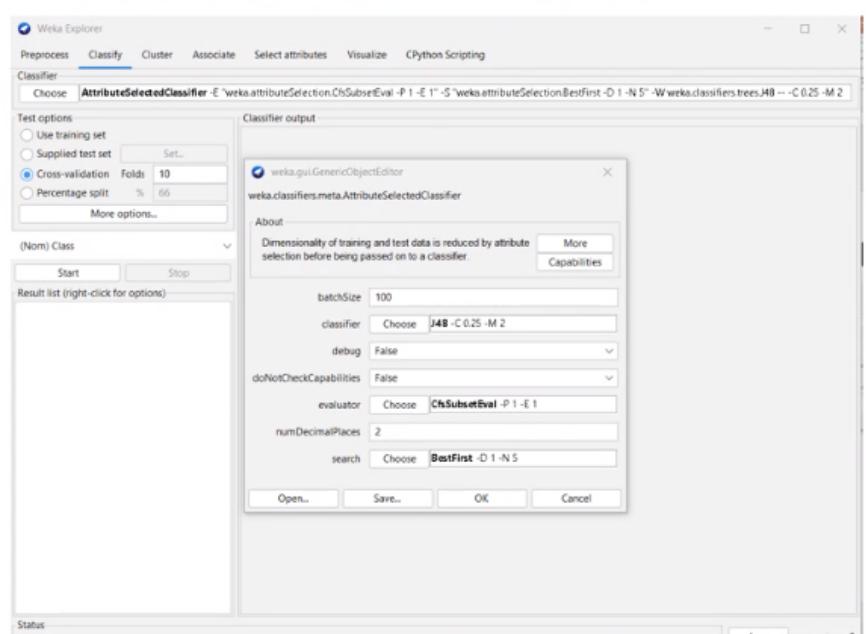
- Select Classify
- Push button Choose
- Choose the classifier you desire to employ
- Select cross-validation
- Push button Start
- Select the execution you would like to visualize in Result list (right-click for options)
- Visualize classifier errors



Se volessimo ad esempio effettuare una selezione delle caratteristiche oppure ribilanciare l'insieme dei dati non può essere fatto in preprocessaizone ma va fatto in cross-validation in quanto va fatto sul training set. Weka quando esegue la cross validation si genera in automatico il training set e il validation set per 10 volte se effettuiamo una 10-cross validation:

Classification with selected features

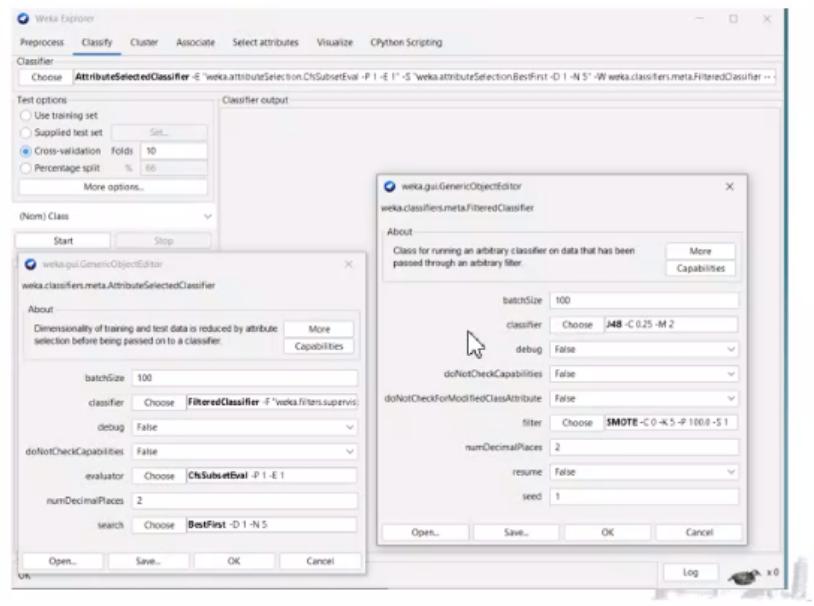
- Select Classify
- Push button Choose
- Choose meta
- Choose AttributeSelectedClassifier
- Configure Attribute Selector



Si può anche ribilanciare il dataset:

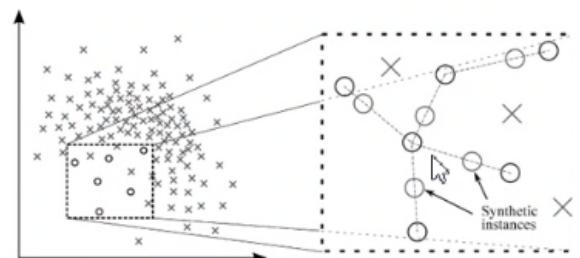
Classification with selected features and rebalance

- Select Classify
- Push button Choose
- Choose meta
- Choose AttributeSelectedClassifier
- For the classifier, choose FilteredClassifier
- Configure Filter



Classification with Imbalance datasets

- **SMOTE: Synthetic Minority Over-sampling Technique**
- The minority class is over-sampled by taking each minority class sample and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors.
- Depending upon the amount of over-sampling required, neighbors from the k nearest neighbors are randomly chosen.
- For instance, if the amount of over-sampling needed is 200%, only two neighbors from the five nearest neighbors are chosen and one sample is generated in the direction of each.



IBk (k nearest neighbor) + J48 (albero di decisione binario)

Per il clustering:

- Select Cluster
- Push button Choose
- Choose clustering algorithm



Clustering

Weka Explorer

Preprocess Classify Cluster Associate Select attributes Visualize CPython Scripting

Clusterer Choose SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance" -R first-last" -l 500 -num-slots 1

Cluster mode
 Use training set
 Supplied test set Set...
 Percentage split % 66
 Classes to clusters evaluation (Num) X2
 Store clusters for visualization
Ignore attributes Start Stop

Result list (right-click for options)
104658 - SimpleKMeans

Cluster output

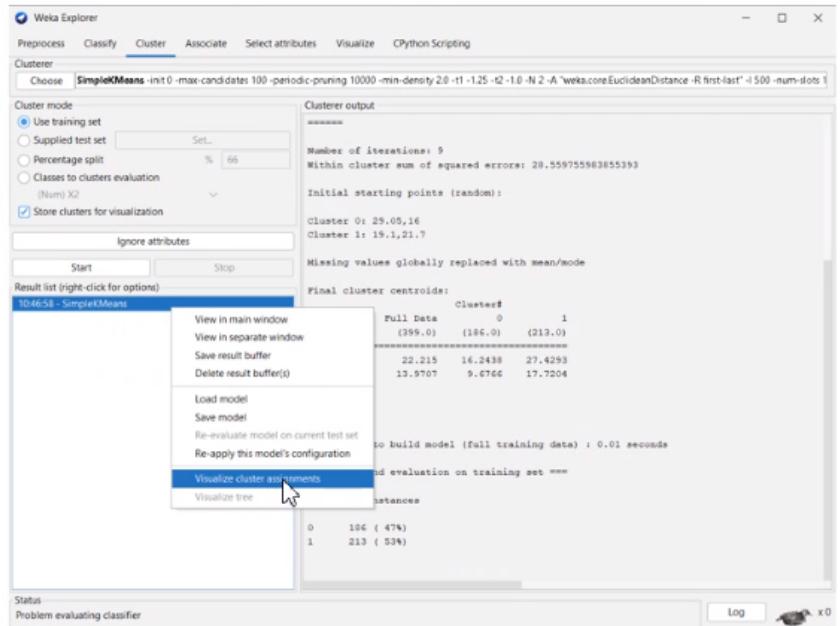
```
=====
Number of iterations: 9
Within cluster sum of squared errors: 28.559755903855393
Initial starting points (random):
Cluster 0: 29.05,16
Cluster 1: 19.1,21.7
Missing values globally replaced with mean/mode
Final cluster centroids:
Cluster#
Attribute Full Data 0 1
(X1) (399.0) (186.0) (213.0)
=====
X1 22.215 16.2430 27.4293
X2 13.9707 9.6766 17.7204

Time taken to build model (full training data) : 0.01 seconds
*** Model and evaluation on training set ***
Clustered Instances
0 106 ( 47%)
1 213 ( 53%)
```

Status Problem evaluating classifier Log x 0

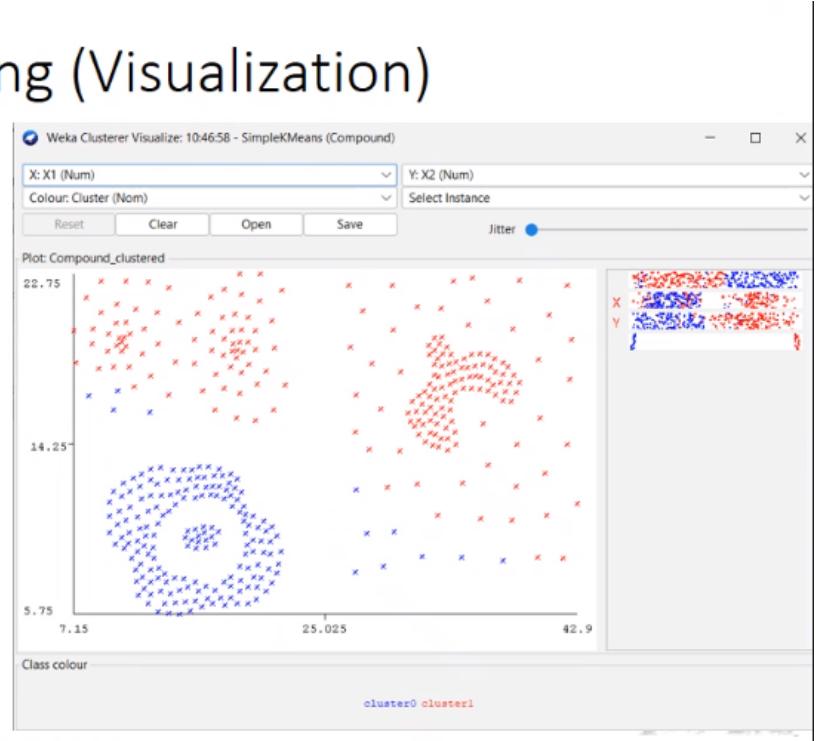
Clustering (Visualization)

- Select Cluster
- Push button Choose
- Choose clustering algorithm
- Push button Start
- Select the execution you would like to visualize in Result list (right-click for options)
- Visualize cluster assignments



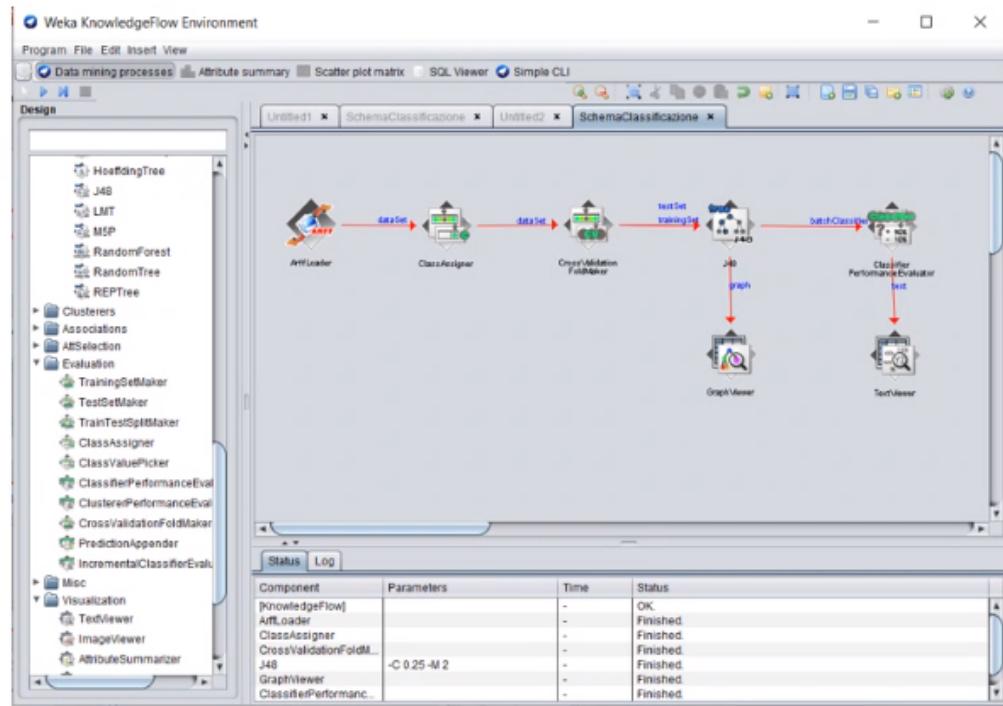
Clustering (Visualization)

- Select Cluster
- Push button Choose
- Choose clustering algorithm
- Push button Start
- Select the execution you would like to visualize in Result list (right-click for options)
- Visualize cluster assignments



Knowledge Flow

Knowledge Flow Classification example (1)



ClassAssigner consente di specificare quale attributo va considerato come attributo classe da identificare.

Knowledge Flow Classification example (2)

Setting up a flow to load an ARFF file and perform a cross-validation using J48 (decision tree)

1. Create a source of data (DataSources tab - ARFFLoader)
2. Connect it to a ARFF file (right click over the ARFFLoader icon - Configure)
3. Specify which attribute is the class (Evaluation tab – ClassAssigner)
4. Connect the ArffLoader to the ClassAssigner (right click over the ArffLoader, select the dataSet under Connections and link with the ClassAssigner component with a left click)
5. Specify which column is the class (right click over the ClassAssigner - choose Configure)
6. Add a CrossValidationFoldMaker component (Evaluation)
7. Connect the ClassAssigner to the CrossValidationFoldMaker (right click over ClassAssigner, select dataSet, left click over CrossValidationFoldMaker)



Assignment

Assignment

Find the best parameter combinations for the classification, clustering and association rule problems described in the next slides.

Prepare two slides for classification and clustering problems.

- First slide, tested algorithms with parameters used
- Second slide, best obtained results

Prepare one slide for association rule problem: obtained results

Submit the slides two days before examination date by using the form

<https://docs.google.com/forms/d/e/1FAIpQLSdSUBHP-qGI0Z-OCwOI7uSHyHOKCOQOGvoJC097adRTIOgaTQ/viewform>



During the examination, I will ask you some explanations on the obtained results.



Il file da caricare va nominato con il nome e cognome.

Classification problem

The datasets contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions.

The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions. For the sake of simplicity, we consider a **stratified sampling with 152 frauds out of 85442 transactions (dataset creditCard.arff)**.

It contains only numerical input variables which are the result of a PCA transformation. Features V1, V2, ... V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependent cost-sensitive learning.

Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.



Usare il dataset creditCard.arff

Usiamo un dataset campionato in cui ci sono 85442 istanze di cui queste 152 sono fraudolente.

Si tratta di un dataset FORTEMENTE sbilanciato.

A. Susto et al. / Journal of... 2017

Classification problem

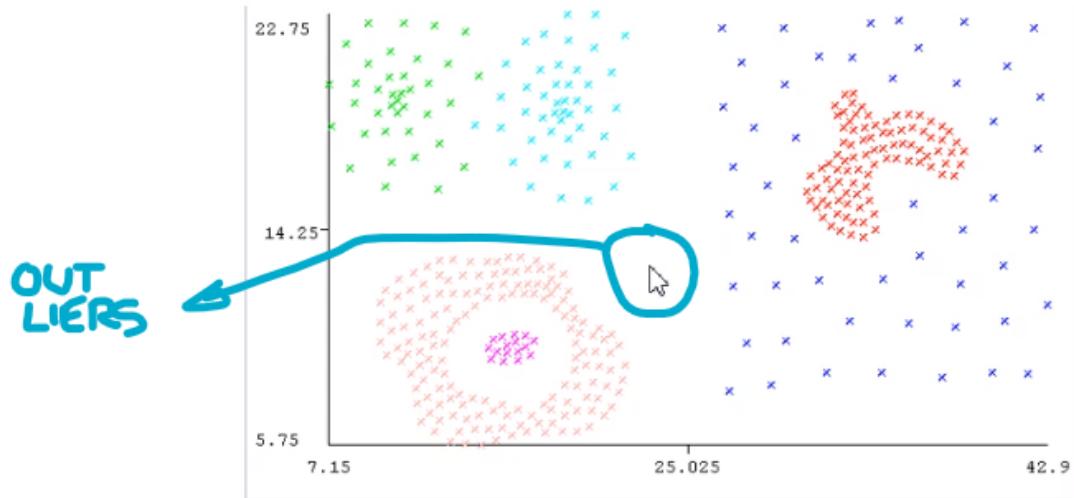
You have to apply a **10-fold cross validation** by using **two types of classifiers** (J48 and Ibk), without and with rebalancing (Smote) and without and with feature selection. In one slide, you have to report the parameters you used and in the second slide you have to report the results you obtained.

M&D/Cs

Per il clustering usiamo il seguente dataset:

Clustering problem

Dataset compoundClustering.arff



Clustering problem

You have to apply **two types of clustering algorithms** (k-means and DBScan). In one slide, you have to report the parameters you used and in the second slide you have to report the results you obtained.

e bisogna cercare la combinazione migliore in modo tale che i cluster che vengono generati corrispondano ai cluster che vediamo

Per l'association rule:

Association rule (optional)

Dataset supermarket.arff

You have to execute the a-priori algorithm with minsupp = 0.1 and confidence = 0.7. In one slide, you have to show the results you obtained.