Data mining

Data mining

- La maggior parte delle aziende dispone di enormi basi di dati contenenti dati di tipo operativo
- queste basi di dati costituiscono una potenziale miniera di utili informazioni
- nei tool DBMS e DW attuali ci sono poche possibilità di estrarre conoscenza
- data mining: scoperta di pattern in data set (di grandi dimensioni)
- tali pattern devono essere
  - -validi
  - -precedentemente sconosciuti
  - -potenzialmente utili
  - -comprensibili

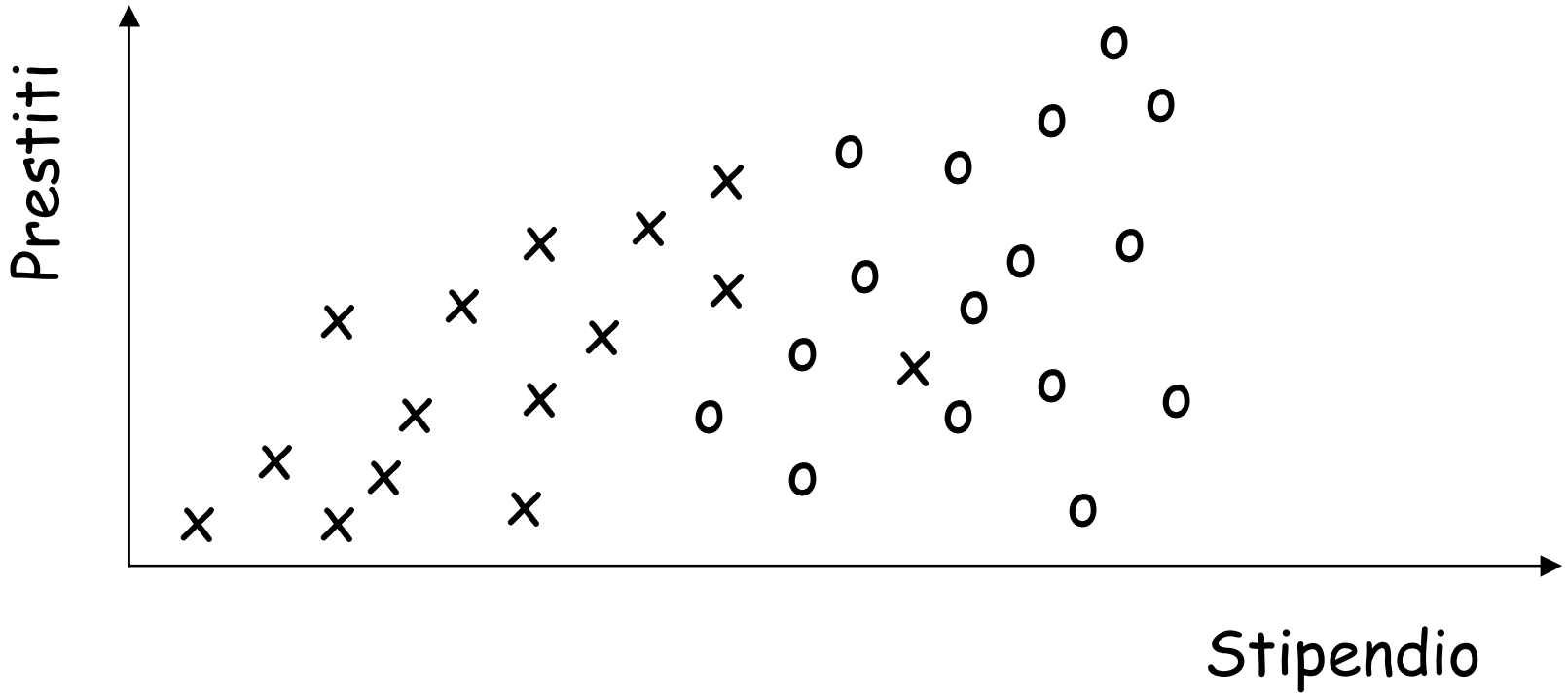
Data mining - Esempio

record delle vedite:

id transazione	id cliente	prodotti comprati
tran1	cust33	p2, p5, p8
tran2	cust45	p5, p8, p11
tran3	cust12	p1, p9
tran4	cust40	p5, p8, p11
tran5	cust12	p2, p9
tran6	cust12	p9

- Tendenza: i prodotti p5, p8 vengono spesso comprati insieme
- Tendenza: al cliente 12 piace il prodotto p9

Data Mining - Esempio

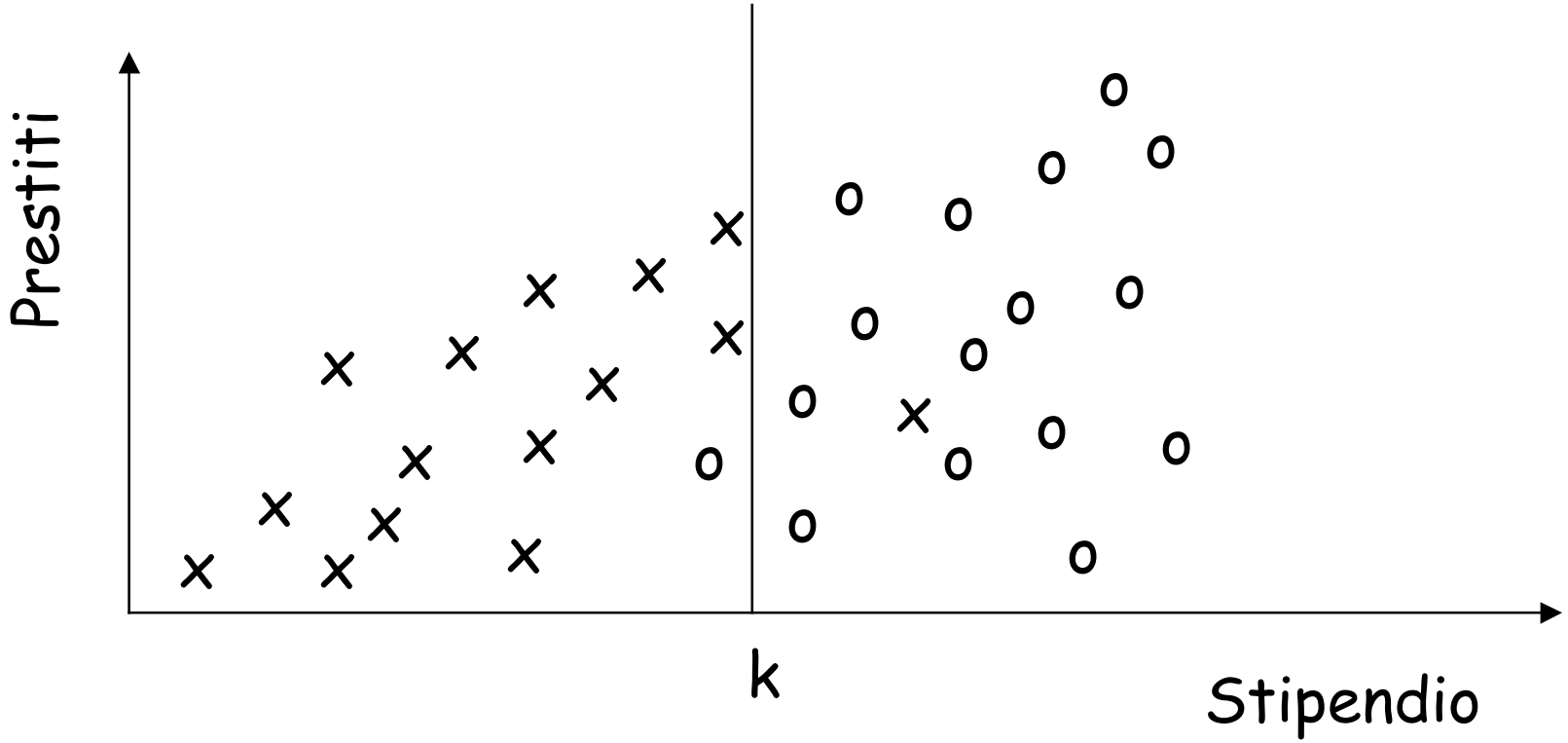


Persone che hanno ricevuto un prestito dalla banca:
x: persone che hanno mancato la restituzione di rate
o: persone che hanno rispettato le scadenze

Knowledge Discovery

- Un processo di KD si basa sui seguenti elementi:
  - *Dati*: insieme di informazioni contenute in una base di dati o data warehouse
  - *Pattern*: espressione in un linguaggio opportuno che descrive in modo succinto le informazioni estratte dai dati
    - regolarità
    - informazione di alto livello

Esempio



IF stipendio < k THEN mancati pagamenti

Caratteristiche dei pattern

- *Validità*: i pattern scoperti devono essere validi su nuovi dati con un certo grado di certezza
  - Esempio: spostamento a destra del valore di k porta riduzione del grado di certezza
- *Novità*: misurata rispetto a variazioni dei dati o della conoscenza estratta
- *Utilità*
  - Esempio: aumento di profitto atteso dalla banca associato alla regola estratta
- *Comprensibilità*: misure di tipo
  - sintattico
  - semantico

Applicazioni

- rilevazione di frodi (ambiti telecomunicazioni, bancario e compagnie di carte di credito)
- approvazione di prestiti e crediti
- analisi degli acquisti (market basket data analysis)
- segmentazione dei clienti
- applicazioni finanziarie
- profilazione dei clienti
- commercio elettronico

Marketing

- Analisi delle vendite
  - ➪associazioni (correlazioni) tra vendite di prodotti
  - ➪birra e pannolini
- Profilazione dei clienti
  - ➪il data mining può dire quali tipi di clienti comprano quali prodotti
- Identificazione dei requisiti dei clienti
  - ➪identificare i prodotti migliori per i diversi clienti
  - ➪predire quali fattori attireranno nuovi clienti

Analisi Corporativa

- Finanze
  - ⇨ analisi e predizione dei flussi di denaro
- Risorse
  - ⇨ riassumere e confrontare le risorse con i costi
- Concorrenza
  - ⇨ confrontarsi con gli altri concorrenti aggregando i dati allo stesso livello

Rilevazione di frodi

- Frodi di compagnie di assicurazioni auto
  - ⇨ l'estrazione di regole di associazione può permettere di rilevare gruppi di persone che simulano incidenti per guadagnare sull'assicurazione

- Riciclaggio di denaro
  - ⇨ dal 1993, l'agenzia statunitense del tesoro per la rilevazione di crimini finanziari usa tecniche di data mining per rilevare transazioni di denaro sospette

Altre applicazioni

- Squadre sportive
  - ⇨ i New York Knicks usano il data mining per ottenere un vantaggio competitivo
- Astronomia
  - ⇨il California Institute of Technology e l'Osservatorio di Palomar hanno scoperto 22 quasar con l'aiuto del data mining
- Banche
  - ⇨la Security Pacific/Bank of America usa il data mining come supporto per le decisioni sui prestiti commerciali e per prevenire le frodi

Il data mining è all'intersezione di molti campi

- AI
- machine learning
- acquisizione di conoscenza
- statistica
- visualizzazione di dati
- reti neurali
- basi di dati
- data mining

Dati residenti in memoria principale

Data residenti in memoria secondaria

Tecniche di data mining

- Regole di associazione
- Clustering
- Classificazione
- Pattern matching in sequenze
- Scoperta di valori disallineati (outliers)
- Mining di testo/immagini

Sfide

- Scalare le tecniche esistenti (bisogno di velocità…) : le vecchie tecniche sono in grado di gestire insiemi di dati residenti in memoria
- Identificare applicazioni del data mining (success stories: ad esempio rilevazione di intrusioni)
- Nuove tecniche e nuovi tipi di data mining

Esempi

- Regole di associazione: il 99% della gente che acquista pannolini acquista anche birra
- Classificazione: le persone sotto i 25 anni che guadagnano meno di 25K sono cattivi creditori
- Sequenze simili: i DNA di *A* e *B* sono simili
- Rilevazione di valori disallineati (ouliers): questa connessione è un attacco

Due stili di data mining

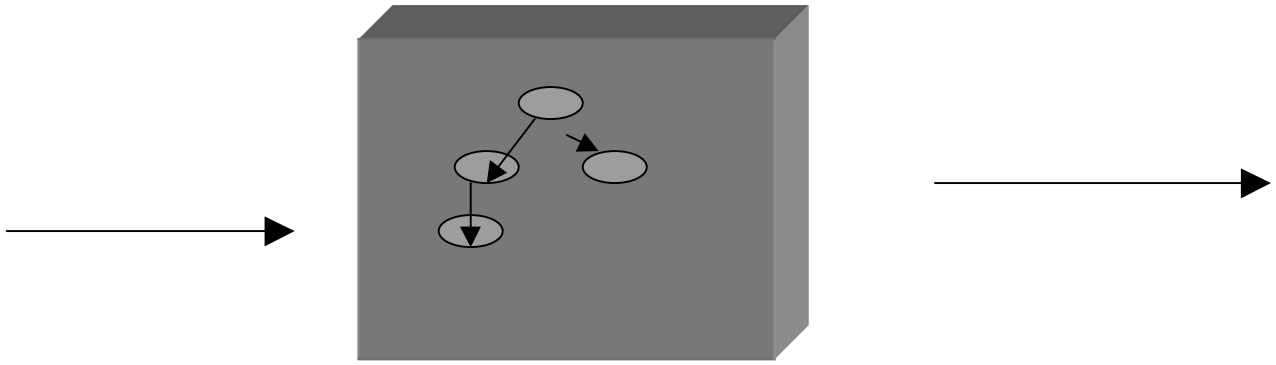
- Data mining diretto: top-down, usato quando sappiamo cosa stiamo cercando (un esempio è la modellazione predittiva)
- Data mining non diretto: bottom-up, "lascia i dati liberi di parlare"
- trova i pattern e lascia all'utente il compito di determinare se siano o meno importanti

- non sono mutuamente esclusivi
- entrambi richiedono l'intervento umano

Data Mining diretto

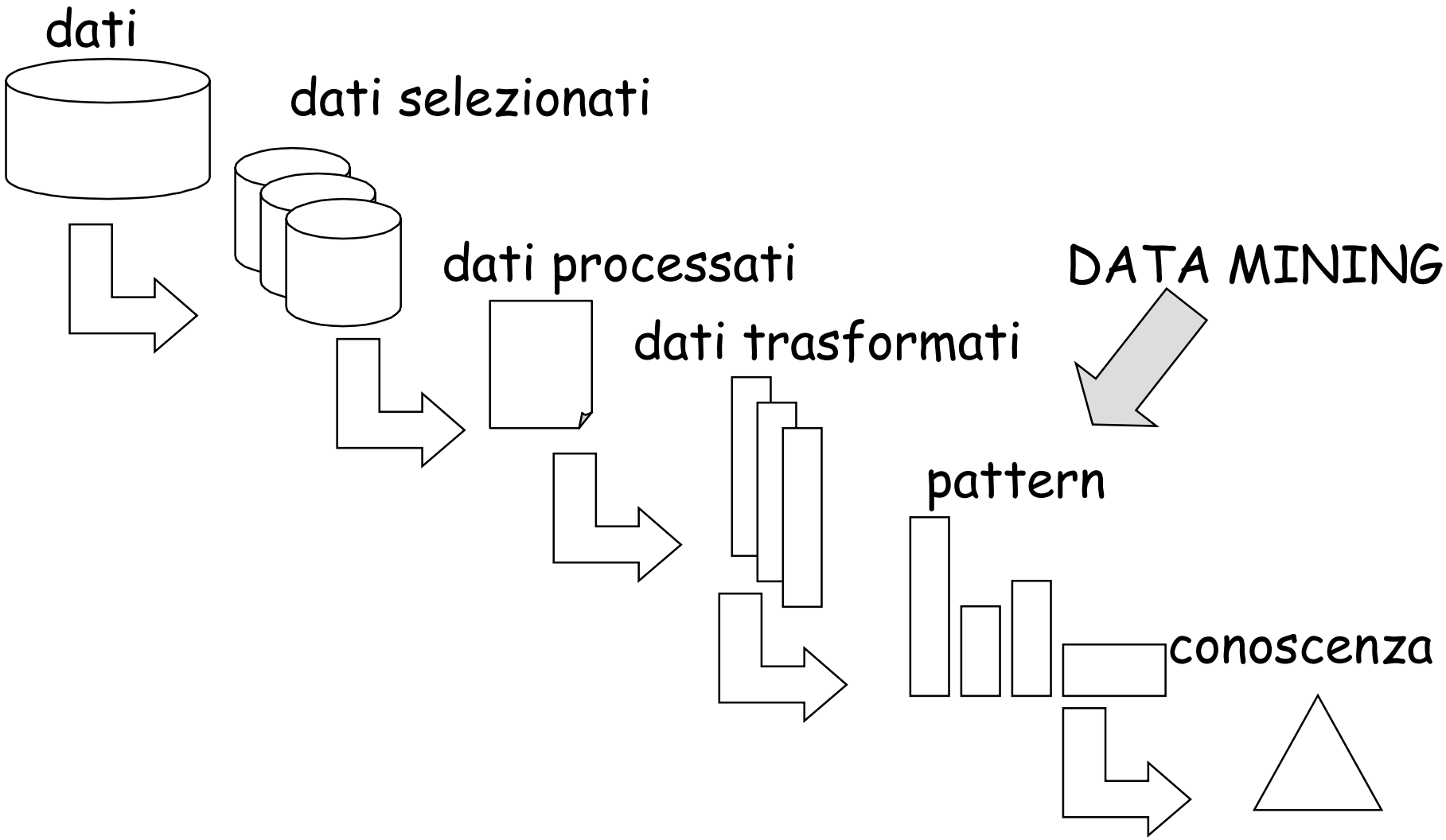
- Modellazione predittiva:
  - Chi è più facile risponda alla nostra campagna?
  - Quale macchina è più facile che fallisca?
  - Quali clienti ci lasceranno entro 6 mesi?

Data Mining non diretto



- Vogliamo sapere cosa sta succedendo: come è il modello viene fuori con le risposte
- es.: segmentazione, vogliamo usare un albero di decisione per scoperire un importante segmento di clienti
  - estrazione di regole di associazione

Processo di estrazione



Processo di estrazione

- Il processo di estrazione in genere parte da insiemi di dati eterogenei
- deve garantire adeguata efficienza, ipotizzando che i dati risiedano su memoria secondaria
- deve essere scalabile
- deve associare misure di qualità ai pattern estratti
- deve permettere di applicare criteri diversificati di estrazione

Estrazione di regole di associazione da basi di dati di grosse dimensioni

Regole di associazione

- Market basket data: il cestino del "supermarket" contiene {pane, latte, birra, pannolini…}
- Si vogliono trovare regole che correlano la presenza di un insieme di prodotti X con un altro insieme Y
  - es: X = pannolini, Y= birra, $X \Rightarrow Y$ con *confidenza* 98%
  - eventualmente vincolato: es, considerando solo clienti uomini

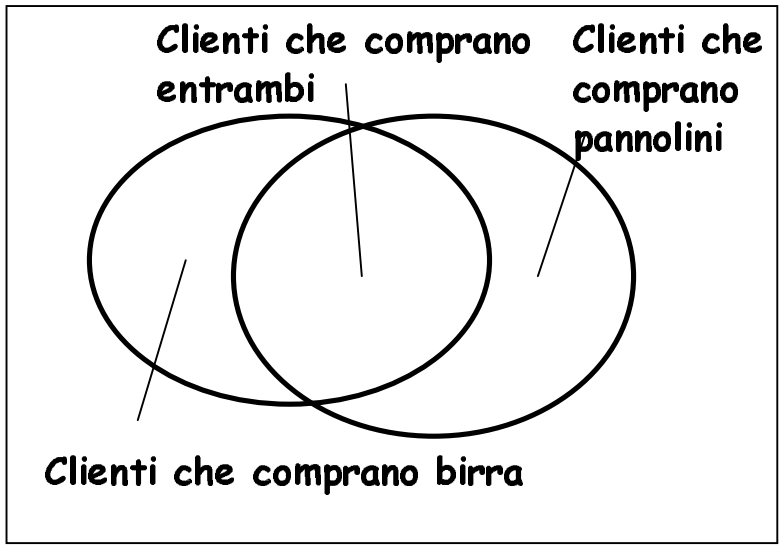
Applicazioni

- Market basket analysis: dimmi come posso migliorare le mie vendite associando promozioni agli insiemi di prodotti "best seller"
- Marketing: "le persone che comprano questo libro comprano anche …"
- Rilevazione di frodi: una richiesta di mutua è sempre collegata a una richiesta di visita a un dottore nello stesso giorno
  Pianificazione degli scaffali: dati i prodotti "best seller," come organizzo gli scaffali del mio supermercato?

Regole di associazione: concetti base

- Dati:
  (1) una base di dati di transazioni,
  (2) ogni transazione è una lista di prodotti (comprati dai clienti in una visita)
- Trova: tutte le regole che correlano la presenza di un insieme di prodotti con quella di un altro insieme di prodotti
  - es: il 98% della gente che acquista gomme e autoaccessori ottiene anche servizi automobilistici
- Applicazioni
  - $* \Rightarrow$ *Accordi di manutenzione* (cosa il negozio dovrebbe fare per incrementare gli accordi di manutenzione)
  - *Piccoli elettrodomestici* $\Rightarrow *$ (quali altri prodotti dovrebbe vendere il negozio)
  - Promozioni (buoni sconto, …) "attaccate" nella vendita diretta

Misure per le regole: supporto e confidenza



Trova tutte le regole $X \& Y \Rightarrow Z$ con supporto e confidenza sopra un certo valore minimo

- **supporto**, *s*, probabilità che una transazione contenga $\{X \cup Y \cup Z\}$
- **confidenza**, *c*, probabilità condizionale che una transazione che contiene $\{X \cup Y\}$ contenga anche Z

ID Transazione	Prodotti
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Sia il supporto minimo 50% e la confidenza minima 50%, si ha

$A \Rightarrow C$ (50%, 66.6%)
$C \Rightarrow A$ (50%, 100%)

Estrazione di regole di associazione - un esempio

ID Transazione	Prodotti
2000	A,B,C
1000	A,C
4000	A,D
5000	B,E,F

Supporto minimo 50%
Confidenza minima 50%

Frequent Itemset	Supporto
{A}	75%
{B}	50%
{C}	50%
{A,C}	50%

Per la regola $A \Rightarrow C$:

supporto = supporto($\{A \cup C\}$) = 50%

confidenza = supporto($\{A \cup C\}$)/supporto($\{A\}$) = 66.6%

Principio **apriori**:

ogni sottoinsieme di un frequent itemset deve essere frequent

Estrazione dei Frequent Itemset: il passo chiave

- Trova i *frequent itemset*: gli insiemi di prodotti che hanno supporto maggiore del supporto minimo
  - un sottoinsieme di un frequent itemset deve essere anch'esso un frequent itemset
    - cioè, se {*AB*} è un frequent itemset, sia {*A*} che {*B*} devono essere frequent itemset
  - trova iterativamente i frequent itemset con cardinalità da 1 a *k (k*-itemset*)*
- Usa i frequent itemset per generare le regole di associazione

Decomposizione del problema

Due fasi:

• Generare tutti gli itemset il cui supporto è sopra una certa soglia. Chiamiamo tali itemset ***grandi (o caldi)***.
(Tutti gli altri itemset sono *piccoli.)*
Come? Generare tutte le combinazioni? (esponenziale!)
(***DIFFICILE***)

• Per un certo itemset grande

$$Y = I_1 I_2 \ldots \quad I_k \qquad k >= 2$$

Generare (al più *k* regole) $X \Rightarrow I_j$ $X = Y - \{I_j\}$

confidenza = $c \leq$ supporto(Y)/ supporto (X)

In questo modo, si ha una soglia *c* e si decide quali regole tenere (***FACILE***)

Esempi

Tid	Items
1	{a,b,c}
2	{a,b,d}
3	{a,c}
4	{b,e,f}

Assumiamo s = 50 % e c = 80 %

Supporto minimo: 50 % $\Rightarrow$ itemset {a,b} e {a,c}

Regole: ~~a $\Rightarrow$ b con supporto 50 % e confidenza 66.6 %~~

~~a $\Rightarrow$ c con supporto 50 % e confidenza 66.6 %~~

c $\Rightarrow$ a con supporto 50% e confidenza 100 %

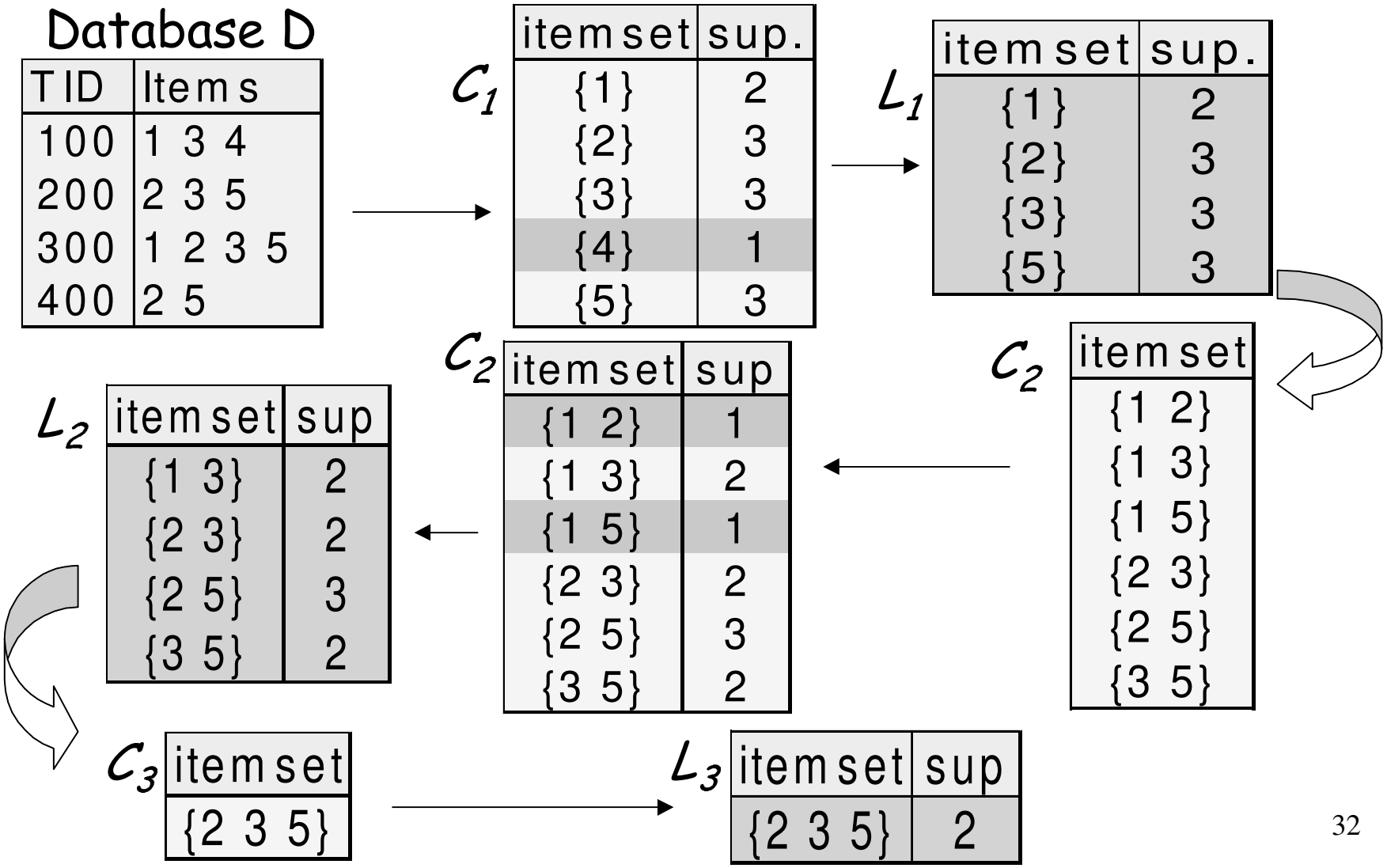
b $\Rightarrow$ a con supporto 50% e confidenza 100%

Algoritmo Apriori

- Passo di Join: C_k viene generato combinando L_{k-1} con se stesso
- Passo di Prune: ogni (k-1)-itemset che non è frequente non può essere un sottoinsieme di un frequent k-itemset
- Pseudo-codice:

C_k: itemset candidati di dimensione k
L_k : frequent itemset di dimensione k
L_1 = {item frequenti};
for (k = 1; L_k !=∅; k++) **do**
begin
C_{k+1} = candidati generati da L_k;
for each transazione t nel database **do**
incrementa il contatore di tutti i candidati in C_{k+1} che sono contenuti in t
L_{k+1} = candidati in C_{k+1} con supporto ≥ min_support
end
return $\cup_k L_k$;

Algoritmo Apriori - esempio



Algoritmo Apriori - esempio

Regole estratte:

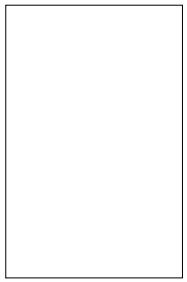
$1 \Rightarrow 3$

$2 \Rightarrow 5$

$5 \Rightarrow 2$

$2\ 3 \Rightarrow 5$

$3\ 5 \Rightarrow 2$



Algoritmo Apriori - esempio

- Supporto = 70% (3 transazioni su 4)
- relazione:

Transid	custid	date	item	qty
111	201	5/1/99	pen	2
111	201	5/1/99	ink	1
111	201	5/1/99	milk	3
111	201	5/1/99	juice	6
112	105	6/3/99	pen	1
112	105	6/3/99	ink	1
112	105	6/3/99	milk	1
113	106	5/10/99	pen	1
113	106	5/10/99	milk	1
114	201	6/1/99	pen	2
114	201	6/1/99	ink	2
114	7/19/00	6/1/99	juice	4

Algoritmo Apriori - esempio

- Level 1:
  - L1: {pen} 1, {ink} 3/4, {milk} 3/4
- Level 2:
  - C2: {pen, ink}, {pen,milk}, {ink, milk}
  - L2: {pen,ink} 3/4, {pen, milk} 3/4
- Level 3:
  - C3: nessuno

- Regole estratte:
  - ink $\Rightarrow$ pen
  - milk $\Rightarrow$ pen

Algoritmo Apriori - Estensioni

- In molti casi, gli item sono organizzati gerarchicamente

- il supporto di un itemset può solo aumentare se un item viene rimpiazzato con un suo antenato nella gerarchia

Algoritmo Apriori - Estensioni

- Supponendo di avere informazioni anche per gli item generalizzati, si possono calcolare le regole nel modo usuale

Transid	custid	date	item	qty
111	201	5/1/99	stationery	3
111	201	5/1/99	beverage	9
112	105	6/3/99	stationery	2
112	105	6/3/99	beverage	1
113	106	5/10/99	stationery	1
113	106	5/10/99	beverage	1
114	201	6/1/99	stationery	4
114	201	6/1/99	beverage	4

- Per esercizio: provare a calcolare le nuove regole di associazione

Algoritmo Apriori - Estensioni

- Determinazione regole di associazione nel contesto di sottoinsiemi di dati, che soddisfano determinate condizioni
  - se una penna è acquistata da un certo cliente, allora è probabile che lo stesso cliente comprerà anche latte
    - si considerano solo gli acquisti di un certo cliente
- pattern sequenziali
  - tutti gli item acquistati da un certo cliente in una certa data definiscono un itemset
  - gli itemset associati ad un cliente possono essere ordinati rispetto alla data, ottenendo una sequenza di itemset (pattern sequenziale)
  - il problema è determinare tutti i pattern sequenziali con un certo supporto

Algoritmo Apriori - Efficienza

- Il "core" dell'algoritmo Apriori:
  - Usa i frequent (k – 1)-itemset per generare i frequent k-itemset candidati
  - Usa scansioni del database e il pattern matching per raccogliere i count per gli itemset candidati
- Il collo di bottiglia di *Apriori* è la generazione dei candidati
  - insiemi di candidati enormi:
    - 10^4 frequent 1-itemset generano 10^7 2-itemset candidati
    - per scoprire un frequent pattern di dimensione 100, es. $\{a_1, a_2, \ldots, a_{100}\}$, si devono generare $2^{100} \approx 10^{30}$ candidati
  - scansioni multiple del database:
    - sono necessarie (n +1) scansioni, dove n è la lunghezza del pattern più lungo

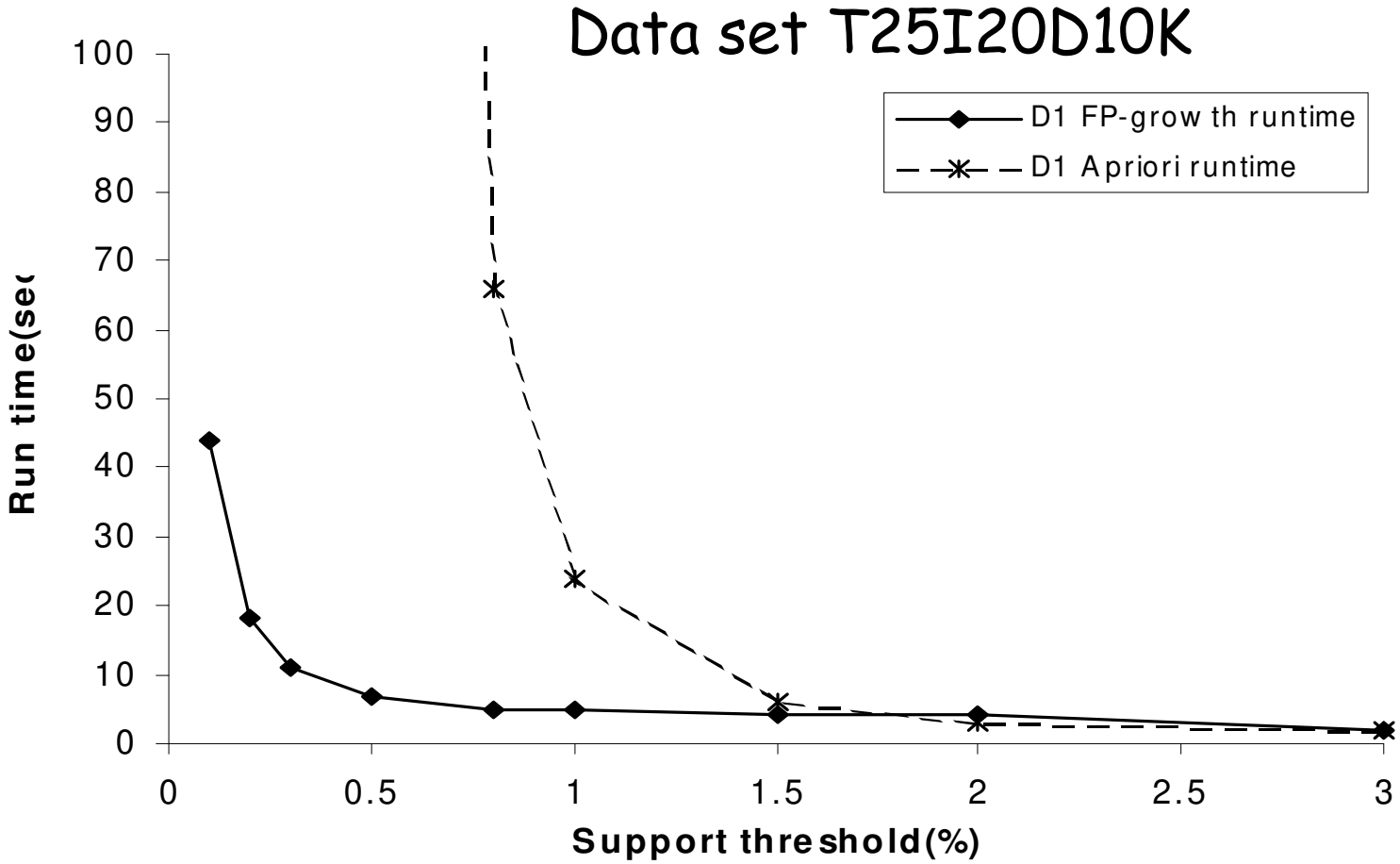
Estrazione dei Frequent Pattern senza generazione dei candidati

- Si comprime un database di grosse dimensioni in una struttura compatta, detta Frequent-Pattern tree (FP-tree)
  - struttura molto condensata, ma completa per frequent pattern mining
  - permette di evitare costose scansioni del database
- Un metodo efficiente di estrazione dei frequent pattern basato sugli FP-tree
  - metodologia divide-and-conquer: si decompongono i task di estrazione in task più piccoli
  - si evita la generazione dei candidati: solo test su sotto-database

FP-growth vs Apriori: Efficienza

- Gli studi delle prestazioni mostrano che
  - FP-growth è di un ordine di magnitudine più veloce di Apriori
- Motivazioni
  - No generazione dei candidati, no test dei candidati
  - Usa una struttura dati compatta
  - Elimina scansioni ripetute del database
  - Le operazioni base sono il counting e la costruzione dell'FP-tree

FP-growth vs Apriori: Scalabilità con la soglia di supporto



Classificazione & Previsione

Il problema della classificazione

- Date:
  - Tuple a ognuna delle quali è assegnato un class level
- Sviluppare un modello per ogni classe
  - esempio:
  - buon creditore : (età in [25,40]) AND (reddito > 50K) AND (stato civile = CONIUGATO)
- Applicazioni:
  - approvazione di crediti (buono, cattivo)
  - collocazione di negozi (buono, medio, sbagliato)
  - situazioni di emrgenza (emergenza, non-emergenza)

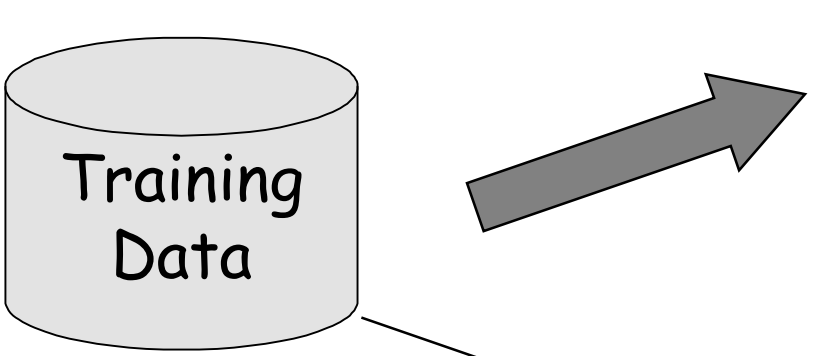
Classificazione vs Previsione

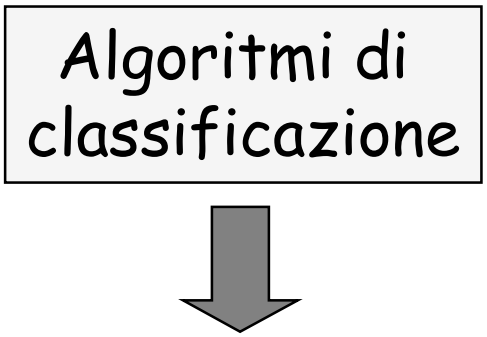
- Classificazione:
  - prevede etichette di classi in categorie
  - classifica i dati (costruisce un modello) basandosi sul training set e i valori (etichette di classi) di un attributo di classificazione e usa tale modello per classificare i nuovi dati
- Previsione:
  - modella funzioni a valori continui, prevede cioè valori sconosciuti o mancanti
- Applicazioni tipiche
  - approvazione di crediti
  - vendita mirata
  - diagnosi mediche
  - analisi dell'efficacia di trattamenti

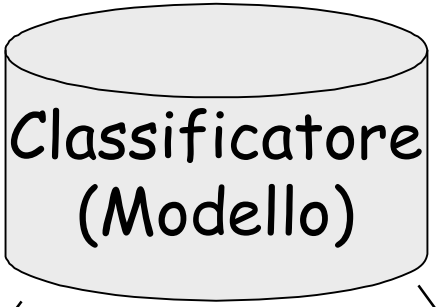
Classificazione
Un processo in due passi

- **Costruzione del modello**: che descrive un insieme di classsi predeterminate
  - ogni tupla/campione è assunta appartenere a una classe predefinita, come determinato dal class label attribute
  - insieme di tuple utilizzati per la costruzione del modello: training set
  - il modello è rappresentato da regole di classificazione, alberi di decisione, o formule matematiche
- **Uso del modello**: per classificare nuovi oggetti
- Stima dell'accuratezza del modello
  - l'etichetta conosciuta del campione di test è confrontata con il risultato della classificazione prodotto dal modello
  - il tasso di accuratezza è la percentuale dei campioni nel test set che sono classificati correttamente dal modello
  - il test set deve essere indipendente dal training set, altrimenti si verifica l'over-fitting

Processo di classificazione
Costruzione del modello



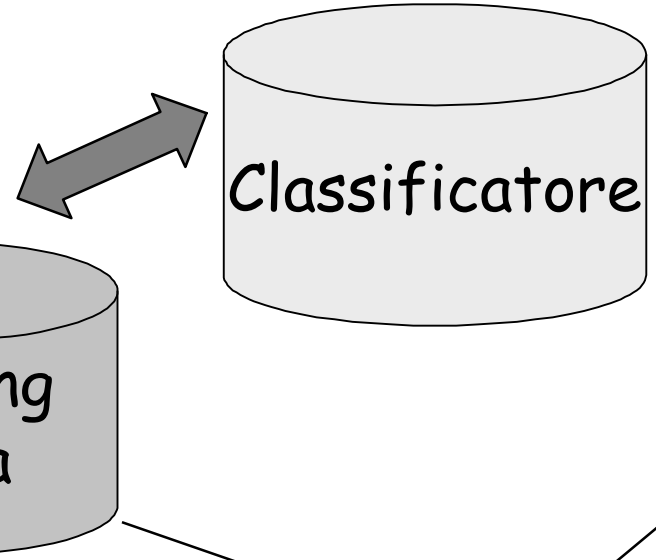


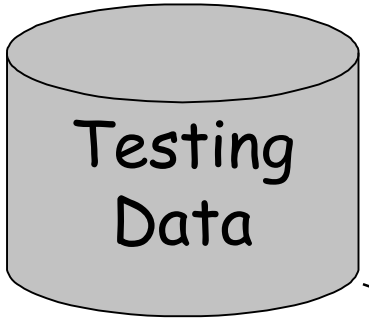


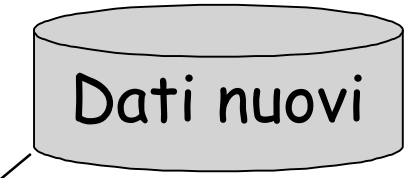
NOME	QUALIFICA	ANNI	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

IF qualifica = 'professor'
OR anni > 6
THEN tenured = 'yes'

Processo di classificazione
Uso del modello nelle previsioni



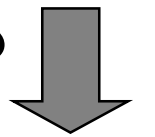




(Jeff, Professor, 4)

NOME	QUALIFICA	ANNI	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Tenured?



Yes

Apprendimento (Learning) Supervisionato vs Non supervisionato

- Apprendimento supervisionato (classificazione)
  - Supervisione: i dati di training (osservazioni, misure, etc.) sono associati ad etichette che indicano la classe dei dati
  - i nuovi dati sono classificati basandosi sul training set
- Apprendimento non supervisionato (clustering)
  - le classi di etichette dei dati di training non sono note
  - dato un insieme di misure, osservazioni, etc. si vuole stabilire l'esistenza di classi o cluster nei dati

Misure per valutare gli approcci di classificazione

- **Accuratezza predittiva**
  - l'abilità del modello di prevedere correttamente la class label di dati nuovi o non precedentemente esaminati
- **Velocità**
  - i costi computazionali coinvolti nel generare e utilizzare il modello
- **Robustezza**
  - l'abilità del modello di fare previsioni corrette in presenza di dati rumorosi o con valori mancanti
- **Scalabilità**
  - l'abilità di costruire il modello efficientemente date grandi quantità di dati
- **Interpretabilità**
  - il livello di comprensione che il modello fornisce

Approcci di classificazione

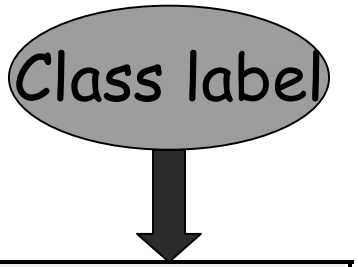
- **Alberi di decisione**
- Classificazione Bayesiana
- Bayesian belief networks
- Reti neurali
- Classificatori K-nearest neighbor
- Case-based reasoning
- Algoritmi genetici
- Rough Sets
- Logica Fuzzy

Classificazione per induzione sugli alberi di decisione

Alberi di decisione

- Alberi di decisione
  - una struttura ad albero simile ai flow-chart
  - i nodi interni denotano un test su un attributo
  - i rami rappresentano il risultato del test
  - i nodi foglia rappresentano le etichette di classi
- generazione degli alberi di decisione in due fasi
  - costruzione dell'albero
    - all'inizio, tutti gli esempi di training sono nella radice
    - gli esempi vengono partizionati ricorsivamente basandosi sugli attributi selezionati
  - pruning dell'albero
    - si identificano e rimuovono rami che riflettono rumore o outliers (valori errati)
- uso degli alberi di decisione: classificazione di campioni sconosciuti
  - si testano i valori degli attributi del campione sull'albero di decisione

Training Dataset

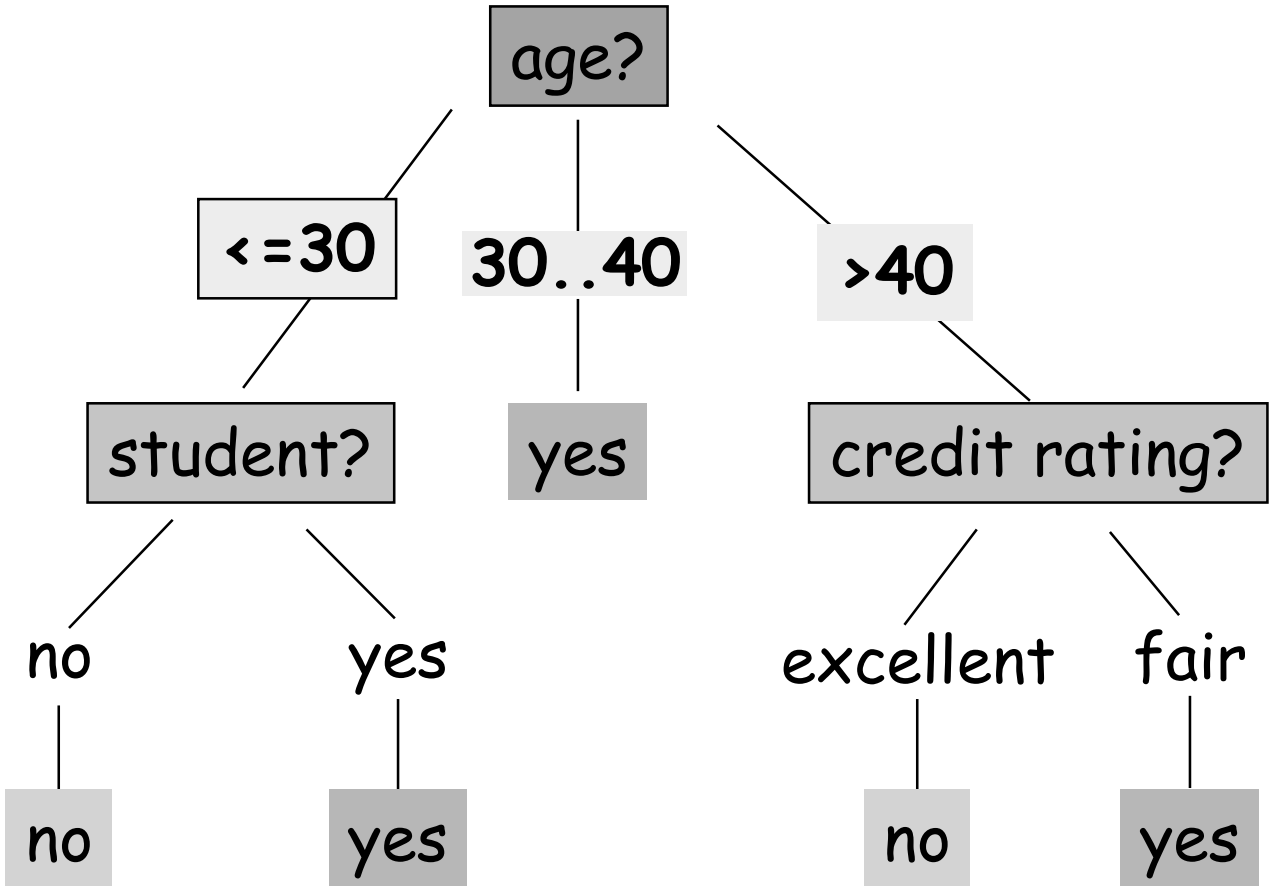


Esempio che segue Quinlan's ID3

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
30…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

Quinlan's ID3 è un algoritmo per generare alberi di decisione

Output: un albero di decisione per "*buys_computer*"



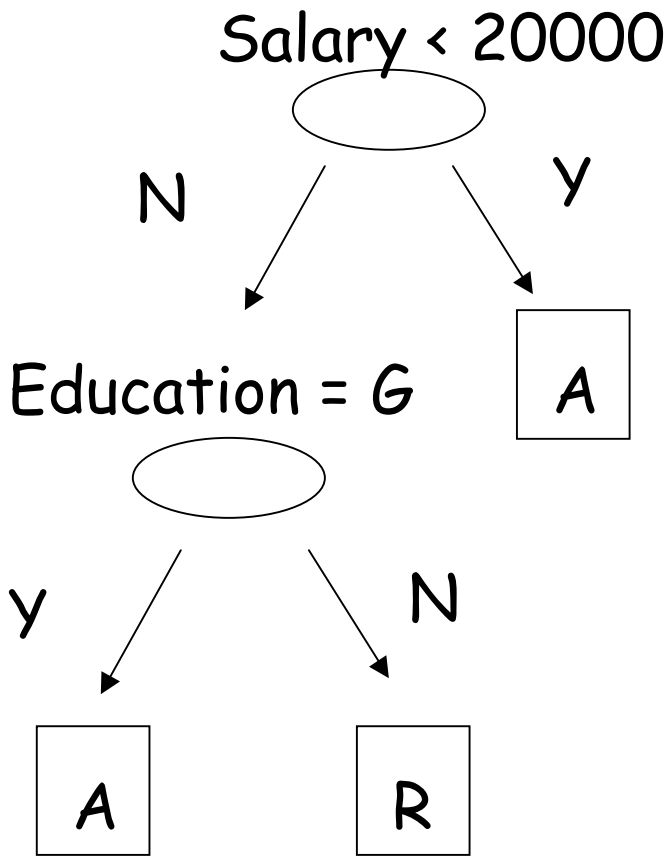
Algoritmo per induzione sugli alberi di decisione

- Algoritmo base (un algoritmo greedy)
  - l'albero è costruito in modo top-down ricorsivo divide-and-conquer
  - all'inizio, tutti gli esempi di training sono nella radice
  - gli attributi sono di tipo categoriale (se a valori continui, sono stati discretizzati in precedenza)
  - gli esempi sono partizionati ricorsivamente basandosi sugli attributi selezionati
  - gli attributi di test sono selezionati sulla base di euristiche o misure statistiche (es, information gain)
- Condizioni per fermarsi nel partizionamento
  - tutti i campioni in un certo nodo appartengono alla stessa classe
  - non ci sono più attributi per un ulteriore partizionamento – per classificare la foglia si usa majority voting
  - non ci sono più campioni non classificati

Alberi di decisione: un esempio

Training set

Salary	Education	Class
10000	HS	R
40000	C	A
15000	C	R
75000	G	A
18000	G	A



Alberi di decisione

- Vantaggi:
  - veloce
  - regole facili da interpretare
  - dati altamente dimensionali
- Svantaggi:
  - no correlazioni

Alberi di decisione: algoritmi proposti

- Machine learning:
  - ID3 (Quinlan 86)
  - C4.5 (Quinlan 93)
  - CART (Breiman, Friedman, Olshen, Stone, Classification and Regression Trees 1984)
- Database:
  - SLIQ (Metha, Agrawal and Rissanen, EDBT96)
  - SPRINT (Shafer, Agrawal, Metha, VLDB96)
  - Rainforest (Gherke, Ramakrishnan, Ghanti VLDB98)

Alberi di decisione

- Trovare l'albero migliore è NP-Hard (problema di complessità esponenziale)
- siamo interessati ad algoritmi che non fanno backtracking (non rimettono mai in discussione una decisione precedente)
- assumiamo di avere un test con *n* risposte che partiziona **T** nei sottoinsiemi $T_1, T_2, \ldots, T_n$

  se il test deve essere valutato senza esplorare le successive dimensioni dei T_i, l'unica informazione disponibile come guida è la distribuzione delle classi in T e nei suoi sottoinsiemi

Algoritmi per gli alberi di decisione

- Fase di costruzione:
  - suddivide ricorsivamente i nodi usando l'attributo e il valore su cui si ha la migliore suddivisione per il nodo
- Fase di pruning:
  - alberi più piccoli (anche se imperfetti) ottengono migliore accuratezza predittiva
  - si potano ricorsivamente i nodi foglia per evitare l'over-fitting

Predictor variables (attributi)

- Numericamente ordinati: i valori sono ordinati e possono essere rappresentati sulla linea reale (es. stipendio)
- Categoriali: assumono valori da un insieme finito su cui non c'è alcun ordinamento naturale (es. colore)
- Ordinali: assumono valori da un insieme finito i cui valori possiedono un ordinamento chiaro, ma le distanze tra loro sono ignote (es. scala di preferenza: buono, medio, cattivo)

Suddivisioni binarie

Partizionamento ricorsivo (binario)

- suddivisione monovariante su un attributo numericamente ordinatio o ordinale X

  $X <= c$

- su categoriale X $X \in A$
- combinazione lineare su numerico

  $\sum a_i X_i <= c$

  c e A sono scelti in modo da massimizzare la separazione

Un po' di probabilità ...

- Supponiamo che le classi in cui vogliamo classificare gli elementi del training set S siano *C1*, …, *Cn*
- **|S|** = # casi totali
- freq(*Ci*,S) = # casi in S che appartengono a *Ci*

- Prob("questo caso appartiene a *Ci*") = freq(*Ci*,S)/|S|
- informazione trasportata =

  log (freq(*Ci*,S)/|S|)
- entropia = informazione attesa = info(*C1*, …, *Cn*) =

  $\Sigma_{i=1}^{n}$ (freq(*Ci*,S)/|S|) log (freq(*Ci*,S)/|S|)

Information gain nell'induzione su alberi di decisione

- Assumiamo che usando l'attributo *A* un insieme S sarebbe partizionato negli insiemi $\{S_1, S_2, \ldots, S_v\}$ e consideriamo per semplicità due sole classi Ci (n=2): P (positivi) e N (negativi)
  - se S_i contiene p_i esempi di *P* e n_i esempi di *N*, l'entropia, o l'informazione attesa per classificare gli oggetti in tutti in sottoalberi S_i è

$$E(A) = \sum_{i=1}^{v} \frac{p_i + n_i}{p + n} I(p_i, n_i)$$

- L'informazione di codifica che si guadagnerebbe effettuando branching su *A*

$$Gain(A) = I(p, n) - E(A)$$

Selezione di attributo attraverso il calcolo dell'Information Gain

- Classe P: buys_computer = "yes"
- Classe N: buys_computer = "no"
- I(p, n) = I(9, 5) =0.940
- Calcoliamo l'entropia per *age*:

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
30…40	4	0	0
>40	3	2	0.971

$$E(age) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694$$

quindi

$$Gain(age) = I(p,n) - E(age) = 0.246$$

analogamente

$$Gain(income) = 0.029$$

$$Gain(student) = 0.151$$

$$Gain(credit\_rating) = 0.048$$

Estrazione di regole di classificazione dagli alberi

- Rappresentano la conoscenza sotto forma di regole IF-THEN
- viene creata una regola per ogni cammino dalla radice a una foglia
- ogni coppia attributo-valore lungo un acmmino form un congiunto
- i nodi foglia contengono la previsione della classe
- le regole sono più facili da capire per gli umani
- Esempi

  IF *age* = "<=30" AND *student* = "*no*" THEN *buys_computer* = "*no*"

  IF *age* = "<=30" AND *student* = "*yes*" THEN *buys_computer* = "*yes*"

  IF *age* = "31...40" THEN *buys_computer* = "*yes*"

  IF *age* = ">40" AND *credit_rating* = "*excellent*" THEN *buys_computer* = "*yes*"

  IF *age* = "<=30" AND *credit_rating* = "*fair*" THEN *buys_computer* = "*no*"

Overfitting

- Gli alberi di decisione possono crescere in modo tale che c'è una foglia per ogni esempio nel training set
- Estremi:
  - Overfitted: "qualsiasi cosa non ho ancora visto, non posso classificarla"
  - Troppo generale: "se è verde, è un albero"

Evitare l'overfitting

- L'albero generato può descrivere troppo accuratamente i dati di training
  - troppi rami, alcuni possono riflettere anomalie dovute a rumore o outliers
  - la conseguenza è una bassa accuratezza per nuovi campioni
- Due approcci per evitare l'overfitting
  - Prepruning: fermare la costruzione dell'albero presto — non suddividere un nodo se questo farebbe scendere la misura di qualità sotto una soglia
    - è però difficile scegliere la soglia opportuna
  - Postpruning: si eliminano rami da un albero "fully grown" — si ottiene una sequenza di alberi progressivamente potati
    - si utilizza un insieme di dati diversi da quelli di training per decidere qual è il "miglior albero potato"

Alberi di decisione scalabili Metodi di induzione

- SLIQ (EDBT'96 — Mehta et al.)
  - costruisce un indice per ogni attributo e solo la lista delle classi e la lista dell'attributo corrente risiedono in memoria
- SPRINT (VLDB'96 — J. Shafer et al.)
  - costruisce una struttura dati di lista di attributi
- PUBLIC (VLDB'98 — Rastogi & Shim)
  - integra suddivisione dell'albero e potatura dell'albero: si ferma prima nella costruzione dell'albero
- RainForest (VLDB'98 — Gehrke, Ramakrishnan et al.)
  - separa gli aspetti di scalabilità da i criteri che determinano la qualità dell'albero
  - costruisce una lista AVC (attributo, valore, class label)

Cluster Analysis

Cos'è la Cluster Analysis?

- Cluster: una collezione di oggetti
  - Simili l'un l'altro all'interno dello stesso cluster
  - Dissimili dagli oggetti negli altri cluster
- Cluster analysis
  - Raggruppamento di un insieme di oggetti in cluster
- Il clustering è classificazione non supervisionata: non ci sono classi predefinite
- Applicazioni tipiche
  - Come un tool stand-alone per ottenere informazioni sulla distribuzione dei dati
  - Come un passo di preprocessing per altri algoritmi

Applicazioni generali del Clustering

- Riconoscimento di pattern
- Analisi di dati spaziali
  - creare mappe tematiche in GIS clusterizzando su feature spaziali
  - rilevare spatial cluster e giustificarli in data mining spaziali
- Image Processing
- Scienze economiche (specialmente ricerche di mercato)
- WWW
  - classificazione di documenti
  - clustering di dati di Weblog per scoprire gruppi di pattern di accesso simili

Esempi di applicazioni del Clustering

- Marketing: scoprire gruppi distinti nelle basi di clienti e usare questa conoscenza per sviluppare programmi di marketing personalizzati
- Land use: identificazione di aree di utilizzo della terra simile in un database di osservazione della terra
- Assicurazione: identificazione gruppi di assicurati con richiesta di rimborso media alta
- City-planning: identificazione di gruppi di case in base al tipo di case, valore e collocazione geografica
- Studio di terremoti: gli epicentri dei terremoti osservati devono essere clusterizzati lungo le faglie continentali

Clustering

- Un buon metodo di clustering produce cluster di alta qualità con
  - alta similarità intra-class
  - bassa similarità inter-class
- La qualità del risultato del clustering dipende sia dalla misura di similarità utilizzata che dal metodo e dalla sua implementazione
- La qualità di un metodo di clustering è anche misurata dalla sua abilità di scoprire alcuni o tutti i pattern nascosti

Requisiti del Clustering in Data Mining

- Scalabilità
- Abilità di gestire diversi tipi di attributi
- Scoperta di cluster con forma arbitraria
- Requisiti minimi di conoscenza del dominio per determinare i parametri di input
- Abilità di gestire rumore e valori disallineati
- Insensibilità all'ordine dei record in input
- Alta dimensionalità
- Incorporazione di vincoli specificati dall'utente
- Interpretabilità e usabilità

Clustering: Metriche

- Metrica di dissimilarità/similarità: la similarità è espressa in termini di una funzione di distanza, che è tipicamente una metrica: $d(i, j)$
- dati rappresentati sotto forma di:
  - matrice dati (n oggetti per p variabili): il valore che ogni oggetto assume per ogni variabile
  - matrice di dissimilarità (n oggetti per n oggetti): memorizza le distanze per tutte le coppie di oggetti
- funzioni di distanza: distanza euclidea, distanza di Manhattan, distanza di Minkowski
- le definizioni delle funzioni di distanza sono in genere molto diverse per variabili scalate su intervalli, booleane, categoriali, ordinali e rapporti
- spesso si assegnano dei pesi alle diverse variabili basandosi sulla semantica dei dati e delle applicazioni

Misure di qualità del Clustering

- c'è una funzione "qualità" separata che misura la "bontà" di un cluster
- è difficile definire "abbastanza simili" o "abbastanza buono"
- la risposta è tipicamente altamente soggettiva

Principali approcci di Clustering

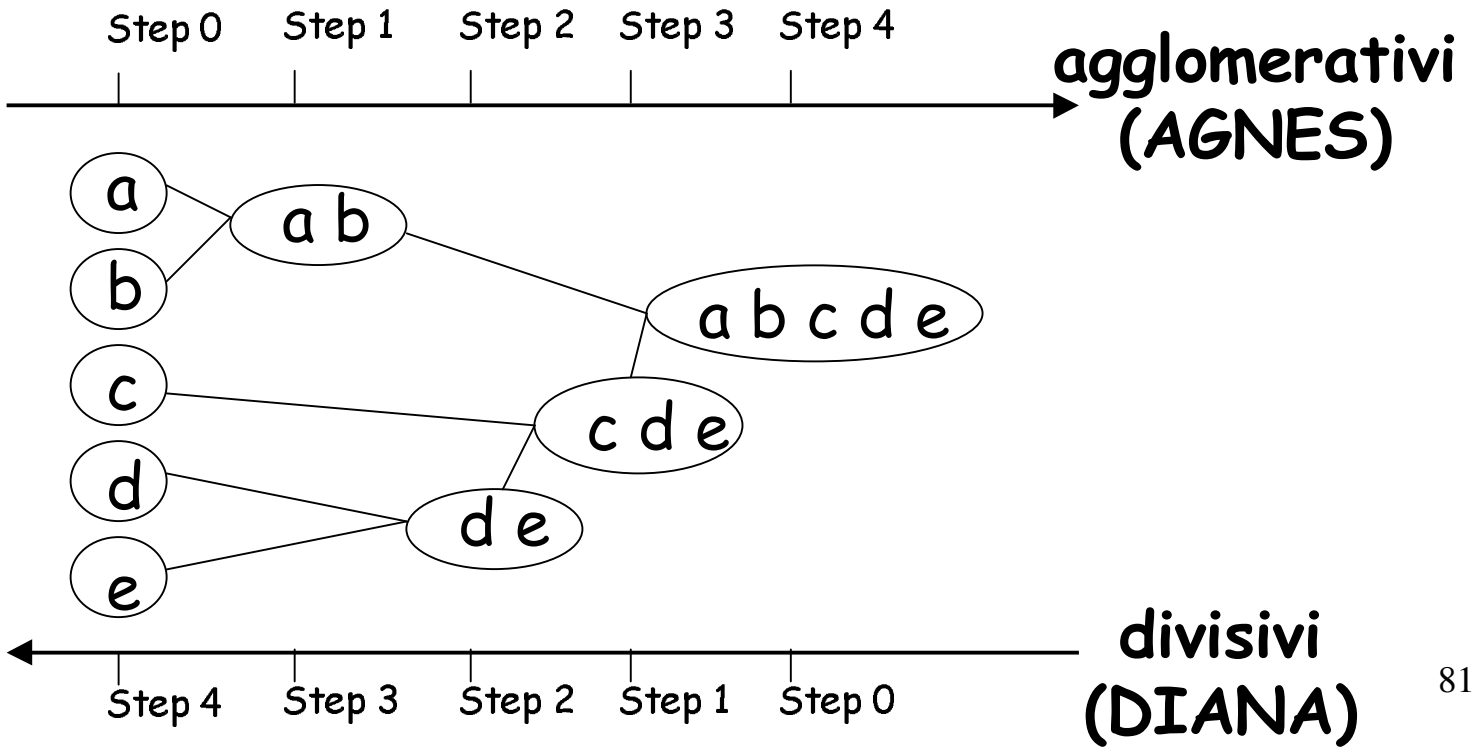
- Algoritmi di partizionamento: si costruiscono le varie partizioni e poi le si valuta sulla base di qualche criterio
- Algoritmi gerarchici: si crea una decomposizione gerarchica dell'insieme dei dati (o oggetti) usando un qualche criterio
- Density-based: basati su funzioni di connectività e densità
- Grid-based: basati su una struttura a livelli di granularità multipla
- Model-based: si ipotizza un modello per ogni cluster e l'idea è trovare la migliore corrispondenza di tale modello con ogni altro

Algoritmi di partizionamento: concetti di base

- Metodi di partizionamento: costruiscono un partizionamento di un database ***D*** di ***n*** oggetti in un insieme di ***k*** cluster
- Dato un *k*, si trova una partizione di *k cluster* che ottimizzi il criterio di partizionamento scelto
  - ottimo globale: enumera esaustivamente tutte le partizioni
  - metodi euristici: algoritmi *k-means* e *k-medoids*
  - *k-means* (MacQueen'67): ogni cluster è rappresentato dal crentro del cluster
  - *k-medoids* o PAM (Partition around medoids) (Kaufman & Rousseeuw'87): ogni cluster è rappresentato da uno degli oggetti nel cluster

Clustering gerarchico

- Usa la matrice distanza come criterio di clustering
- questo metodo non richiede il numero ***k*** di cluster come input, ma ha bisogno di una condizione di terminazione



Metodi di Clustering Density-Based

- Clustering basato su densità (local cluster criterion), come il density-connected points
- caratteristiche principali:
  - scopre cluster di forma arbitraria
  - gestisce il rumore
  - una sola scansione
  - ha bisogno dei parametri di densità come condizione di terminazione
- metodi:
  - DBSCAN: Ester, et al. (KDD'96)
  - OPTICS: Ankerst, et al (SIGMOD'99).
  - DENCLUE: Hinneburg & D. Keim (KDD'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)

Metodi di Clustering Grid-Based

- Usano strutture dati a griglia multirisoluzione
- metodi:
  - STING (a STatistical INformation Grid approach): Wang, Yang e Muntz (1997)
  - WaveCluster: Sheikholeslami, Chatterjee e Zhang (VLDB'98)
  - CLIQUE: Agrawal, et al. (SIGMOD'98)
  - Self-Similar Clustering: Barbará & Chen (2000)

Metodi di Clustering Model-Based

- Cercano di ottimizzare la corrispondenza tra i dati e qualche modello matematico
- Approcci statistici e AI
  - Conceptual clustering
    - una forma di clustering in machine learning
    - produce uno schema di classificazione per un insieme di oggetti non etichettati
    - trova descrizioni caratteristiche per ogni concetto (classe)
  - COBWEB (Fisher'87)
    - un metodo semplice e popolare di conceptual learning incrementale
    - crea un clustering gerarchico sotto forma di un classification tree
    - ogni nodo si riferisce ad un concetto e contiene una descrizione probabilistica di tale concetto