

Regole associative

Regole associative

- Le regole associative descrivono correlazioni di eventi e possono essere viste come regole probabilistiche.
- Due eventi sono correlati quando sono frequentemente osservati insieme.
- Esempio: database di transazioni di vendita in un supermercato. Le regole associative in questo caso descrivono quali oggetti sono frequentemente acquistati insieme
 - Panettone=>spumante
 - Pannolini => birra

Regole associative

- Il problema della scoperta di regole associative può essere espresso come segue
- Sia $I = \{i_1, i_2, \dots, i_m\}$ un insieme di letterali chiamati **oggetti** (o **items**).
- Una **transazione** T è un insieme di oggetti tali che $T \subseteq I$. Un **database di transazioni** D è un insieme di transazioni ed è solitamente memorizzato in una tabella della forma

Identificativo della transazione	Item
----------------------------------	------

- Un **itemset** X è un set di oggetti tali che $X \subseteq I$. Si dice che una transazione T **contiene** un itemset X se $X \subseteq T$.

Regole associative

- Il **supporto** di un itemset X ($\text{supporto}(X)$) è la frazione di transazioni in D che contiene X
$$\text{supporto}(X) = \frac{\text{transazioni che contengono } X}{\text{numero totale di transazioni}}$$
- Una **regola associativa** è una implicazione della forma $X \Rightarrow Y$, dove X e Y sono itemsets e $X \cap Y \neq \emptyset$.

Confidenza e supporto

- $X \Rightarrow Y$ ha **supporto** s nel database D se e solo se una frazione pari ad s delle transazioni in D contengono $X \cup Y$:

$$s = \text{supporto}(X \Rightarrow Y) = \text{supporto}(X \cup Y)$$

- $X \Rightarrow Y$ ha **confidenza** c nel database D , se e solo se, tra tutte le transazioni che contengono X , ce n'è una frazione c che contiene anche Y :

$$c = \text{confidenza}(X \Rightarrow Y) = \frac{\text{supporto}(X \cup Y)}{\text{supporto}(X)}$$

- Confidenza e supporto possono essere indicati anche in forma percentuale

Esempi

- $1 \ \&2 \Rightarrow 3$ ha il 90% di confidenza se, quando un cliente ha comperato gli oggetti 1 e 2, nel 90% dei casi ha comperato anche 3
- $1 \ \&2 \Rightarrow 3$ ha il 20% di supporto se il 20% delle transazioni contiene 1, 2 e 3. Indica la frazione dei casi nei quali la regola si applica

Esempio

Transaction ID	Purchased Items
1	{1, 2, 3}
2	{1, 4}
3	{1, 3}
4	{2, 5, 6}

- Per supporto minimo 50% e confidenza minima 50% abbiamo le seguenti regole
- $1 \Rightarrow 3$ con supporto 50% e confidenza 66%
- $3 \Rightarrow 1$ con supporto 50% e confidenza 100%

Regole associative

- Dato un database D, il compito di scoprire le regole associative può essere riformulato come segue:
 - scoprire tutte le regole associative con almeno un minimo supporto (chiamato **minsup**) e una minima confidenza (chiamata **minconf**), dove minsup e minconf sono valori specificati dall'utente
- Il compito di scoprire regole associative può essere decomposto in due sottoproblemi:
 - Trovare tutti gli itemset che hanno supporto sopra il minimo. Tali itemset sono chiamati **itemset grandi**. Questo sottoproblema è risolto dall'algoritmo APRIORI
 - Generare tutte le regole associative con almeno la confidenza minima dall'insieme degli itemset grandi

Regole associative

- Il secondo sottoproblema è risolto dal seguente algoritmo:
 - Per ciascun itemset grande A , trova tutti i sottoinsiemi non vuoti
 - Per ciascun sottoinsieme X di A , genera la regola $X \Rightarrow (A-X)$ se e solo se il rapporto tra il supporto di A e il supporto di X è almeno minconf

Supporto: proprietà'

- Sia $X1$ un itemset e $X2$ un suo sottoinsieme proprio
- Una transazione che contiene $X1$ contiene anche $X2$
- Una transazione che contiene $X2$ non è detto che contenga anche $X1$
- Le transazioni che contengono $X2$ sono un soprainsieme di quelle che contengono $X1$
- Quindi il supporto di $X1$ è sempre inferiore o uguale a quello di $X2$
- Quindi se $X2$ non è grande, neanche $X1$ lo è
- Esempio: $\{1,2\}$ non è grande quindi $\{1,2,3\}$ non può essere grande

Supporto: proprietà'

- Però se $X1$ è grande, allora $X2$ è grande
- Ad es, se $\{1,3\}$ è grande allora $\{1\}$ e $\{3\}$ sono grandi
- Non e' pero' vero il viceversa, ovvero che se tutti i sottoinsiemi propri $X2$ di un itemset $X1$ sono grandi allora $X1$ e' grande
- Esempio: $\{1,2\}$ non e' grande, anche se tutti i suoi sottoinsiemi propri $\{1\}$ e $\{2\}$ lo sono
- La grandezza dei sottoinsiemi di $X1$ non ci da informazioni sicure su $X1$

Supporto: proprietà

- La proprietà che se X_2 non è grande, neanche X_1 lo è, ci fornisce un metodo per esplorare lo spazio dei possibili itemset
- X_1 è **più specifico** di X_2 se $X_1 \supset X_2$
- X_2 è **più generale** di X_1 se X_1 è più specifico di X_2
- Si dice che il vincolo $\text{supporto} > \text{minsup}$ è un vincolo **antimonotono** rispetto alla relazione di specificità:
 - Se il vincolo è vero per una ipotesi, è vero per tutte le generalizzazioni
 - Se il vincolo è falso per una ipotesi, è falso per tutte le specializzazioni

Algoritmo APRIORI

- Si parte dagli itemset di dimensioni minori, quelli con un solo elemento
- Si esegue uno scan del database per determinare quali sono grandi
- Ad ogni passo, si costruiscono gli itemset grandi di dimensione k usando gli itemset grandi di dimensione $k-1$ e considerando i loro soprainsiemi con un elemento in più
- Gli itemset non grandi di dimensione $k-1$ non si considerano: infatti i loro soprainsiemi non possono essere grandi
- Quindi da un passo all'altro si tengono solo gli itemset grandi

Algoritmo APRIORI

Notazione

k-itemset: un itemset con k oggetti.

L_k : insieme dei k-itemsets grandi (quelli con supporto minimo).

1. $L_1 = \{1\text{-itemsets grandi}\};$
2. per ($k=2; L_{k-1} \neq \emptyset ; k++$) fai
3. $C_k = \text{apriori-gen}(L_{k-1});$ // nuovi candidati
4. Per tutti le transazioni $t \in D$ fai
5. $C_t = \text{subset}(C_k, t);$ //candidati contenuti in t
6. Per tutti gli insiemi candidati $c \in C_t$ do
7. $c.\text{count}++;$
8. fine
9. $L_k = \{c \in C_k \mid c.\text{count} > \text{minsup}\}$
10. fine
11. $\text{Answer} = \cup_k L_k$

apriori-gen

- Due passi:
 - Passo di join
 - Passo di potatura

Passo di join

- Si assume che gli item negli itemset siano mantenuti in ordine alfabetico sulla base del loro nome
- Un k-itemset puo' essere visto come una tabella con k colonne
- Si genera un itemset di k elementi unendo due itemset grandi di k-1 elementi che hanno i primi k-2 elementi uguali (in questo modo, due sottoinsiemi di k-1 elementi sono sicuramente grandi)

Insert into C_k

Select p.item₁, p.item₂, ..., p.item_{k-1}, q.item_{k-1}

From L_{k-1} p, L_{k-1} q

Where (p.item₁=q.item₁) and and

(p.item_{k-2}=q.item_{k-2}) and (p.item_{k-1}<q.item_{k-1})

Passo di potatura

- Si eliminano gli itemset che non hanno tutti i sottoinsiemi con $k-1$ elementi grandi

Per tutti gli itemset $c \in C_k$ fai

Per tutti i $(k-1)$ -sottoinsiemi s di c fai

Se $(s \notin L_{k-1})$ allora

Cancella c da C_k

Esempio

Supp. Min.=50%=2 trans.

Database D

TID	Items
1	{1, 2, 3}
2	{1, 4}
3	{1, 3}
4	{2, 5, 6}

Scan D →

L₁

Itemset	Supp.
{1}	3
{2}	2
{3}	2

C₂

Itemset
{1,2}
{1,3}
{2,3}

Scan D →

C₂

Itemset	Supp.
{1,2}	1
{1,3}	2
{2,3}	1

L₂

Itemset	Supp.
{1,3}	2

C₃

Itemset	Supp.
---------	-------

→

L₃

Itemset	Supp.
---------	-------

Altro esempio

- Database:

ID Transazione	Items acquistati
1	{1,2,3,4}
2	{1,2,3}
3	{2,3,4}
4	{1,4}
5	{1,2,3,4}
6	{1,2,3}

- Supporto minimo = 50% = 3 transazioni
- Si esegue una scansione del database e si ottiene L1=

Itemset	Supporto
{1}	5
{2}	5
{3}	5
{4}	4

Altro esempio

- Si fa il join di ogni itemset con tutti gli altri: C2=

Itemset
{1,2}
{1,3}
{1,4}
{2,3}
{2,4}
{3,4}

- Si calcola il supporto per ogni item set: C2=

Itemset	Supporto
{1,2}	4
{1,3}	4
{1,4}	3
{2,3}	5
{2,4}	3
{3,4}	3

Altro esempio

- Si eliminano gli itemset che non sono grandi (nell'esempio nessuno): L2=

Itemset	Supporto
{1,2}	4
{1,3}	4
{1,4}	3
{2,3}	5
{2,4}	3
{3,4}	3

- Si fa il join degli itemset ottenendo C3=

Itemset
{1,2,3}
{1,2,4}
{1,3,4}
{2,3,4}

Altro esempio

- Si esegue una scansione del database e si ottiene $C3=$

Itemset	Supporto
{1,2,3}	4
{1,2,4}	2
{1,3,4}	2
{2,3,4}	3

- Si eliminano gli itemset non grandi: $L3=$

Itemset	Supporto
{1,2,3}	4
{2,3,4}	3

Altro esempio

- Si fa il join di degli itemset ottenendo $C4=$

Itemset
{1,2,3,4}

- Si eliminano gli itemset che hanno un sottoinsieme non grande: $C4=$

Itemset

- Si termina dato che $C4$ è vuoto

Altra forma del database

- Si possono apprendere regole associative anche da database della forma

Attribute₁	Attribute₂	Attribute_n
Value _{1,1}	Value _{1,2}	Value _{1,n}
....
Value _{m,1}	Value _{m,2}	Value _{m,n}

- In pratica, ogni record è considerato come una transazione e ogni possibile uguaglianza **Attributo=Valore** un item

Altra forma del database

- In questo caso una regola associativa ha la forma

$$A_1=v_{A1}, \dots, A_j=v_{Aj} \Rightarrow B_1=v_{B1}, \dots, B_k=v_{Bk}$$

- dove
 - $A_1, \dots, A_j, B_1, \dots, B_k$ sono nomi di attributi
 - $v_{A1}, \dots, v_{Aj}, v_{B1}, \dots, v_{Bk}$ sono valori tali che v_{Ai} (v_{Bh}) appartiene al dominio dell'attributo A_i (B_h).
- I database della prima forma possono essere tradotti in questa seconda forma considerando un attributo booleano per ogni possibile item.