Progetto di basi di dati AREA ANALYTICS

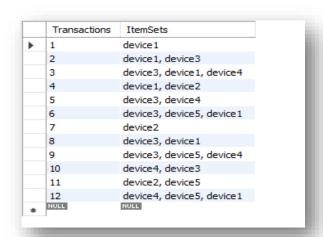
Abitudini degli utenti tramite Association rule learning

Implementazione dell'algoritmo Apriori:

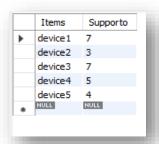
Nel file APRIORI.SQL, ci sono tutti i passaggi dell'algoritmo commentati in blocchi.

Le tabelle che servono per raccogliere i dati di transazioni e items, sono rispettivamente UTILIZZO e CANDIDATE1 create e popolate all'inizio dall'utente.

La stored procedure *APRIORI* (*IN supp double, IN conf double*) che prende in ingresso due double, che rappresentano le soglie di Supporto e di Confidenza da usare più avanti nel processo dell'algoritmo per ottenere le regole di associazione più forti. Evidenziamo adesso passo dopo passo con degli screenshots:



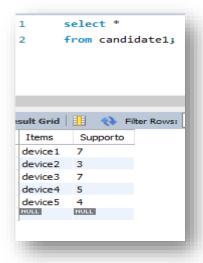
Inseriamo all'interno della tabella UTILIZZO, tutte le transazioni.



inseriamo nella tabella CANDIDATE1 l'elenco degli Items che costituiscono le transazioni.

I = { device1, device2, device3, device4, device5 }

PASSO 1: L_{k-1} ; k = 2, ... |I| L_1



dall'insieme **C**₁ a sinistra, ottengo l'insieme di large itemset **L**₁ a destra, selezionando quelli items che hanno valore di supporto maggiore o pari al valore della soglia prestabilita all'inizio dell'algoritmo.



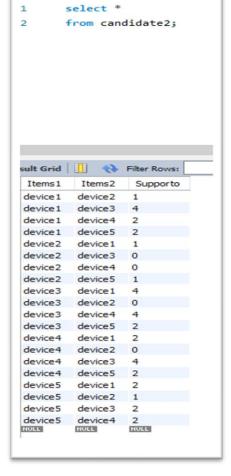
PASSO 2: $C_2 = L_1 CROSS JOIN L_1$



Effettuo il cross join per ottenere tutte le combinazioni possibili tra gli items rimasti in L_1 .

Elimino i record che contengono più di una volta lo stesso item.

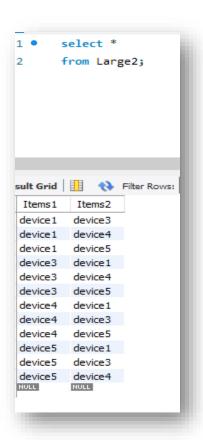
C₂ a destra .



PASSO 3 & 4

Pruning : da C₂ selezioniamo gli itemset che hanno supporto maggiore o pari al valore del Supporto prestabilito per ottenere l'insieme L₂ di 2-itemsets.

Trasferisco in L₂ l'insieme di record rimasto in C₂.



Poi si ripete dal passo1, finché non si ottiene un insieme L_k vuoto.

Effettuando diversi test di questo algoritmo con, abbiamo notato che:

- Diminuire la lunghezza massima(in termine di numero di items che costituiscono la regola) delle regole associative permette di ridurre il build time, di generare anche delle regole più semplici.
- Aumentare la soglia minima di supporto, ci permette di ridurre il build time, e di generare poche regole.
- Quando noi diamo un dataset piccolo, l'algoritmo potrebbe generare delle false regole, o meglio regole che non sono sempre vere, non capitano molto spesso.
- Per quanto riguarda lo studio delle abitudini degli utenti della smarthome, le possibilità sono tante:
 - Regole che stabiliscono quale gruppo di device uso molto spesso
 - Le impostazioni più usate su un dispositivo dato
 - Momento della giornata e quali device tende ad usare di più.

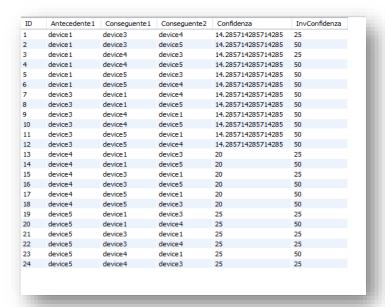
TEST: CALL APRIORI(1, 50)

Prima riga:

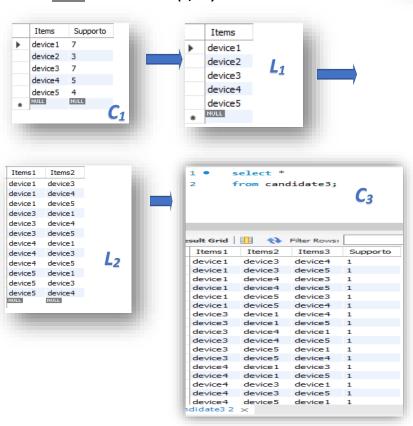
device1 -> device3, device4 14,28

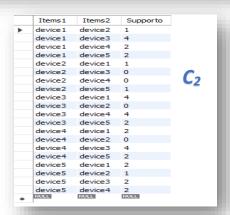
InvConfidenza rappresenta la confidenza della regola inversa a quella della riga.

device3, device4 -> device1 25



TEST: CALL APRIORI (2, 50)





Items1

NULL

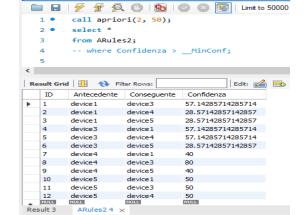
L₃

Items2

NULL

A questo punto, L_3 risulta vuoto perché nessun record ha supporto >= MinSupp quindi l'algoritmo si ferma, e considera L_2 come large itemsets da qui pescare per generare tutte le regole associative.

Items3

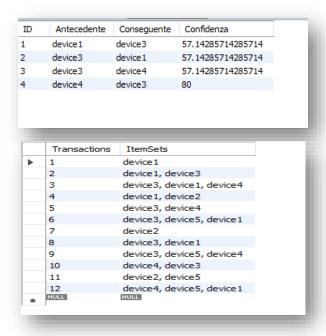


TEST: CALL APRIORI (3, 50)

in questo caso, sono già stati selezionate le regole con Confidenza > MinConf

device1 -> device3

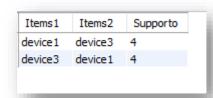
device4 -> device3: regole che si possono verificare andando a controllare la tabella delle transazioni. In particolare, si nota che nelle transazioni 3, 5, 9 e 10 device3 e device4 compaiono entrambi.



TEST: CALL APRIORI (7, 50)

Caso particolare in cui, L_2 è vuoto perché tra i record di C_2 nessun ha supporto >= 7.

A regola, per generare le regole associative, l'algoritmo dovrebbe considerare la tabella di large itemsets precedente quindi L₁, però qui abbiamo degli itemsets costituiti solamente da un item e non è



possibile generare nessuna regola con un item solo. L'algoritmo si ferma e non ci sono regole associative.