Homework 2: Discovery of Frequent Itemsets and Association Rules

ID2222 Data Mining

Homework Groups 70,
Márk Antal Csizmadia
macs@kth.se

November 2021

1 Introduction

In this document, methods and results of the discovery of frequent itemsets and association rules in massive data sets is discussed. The discussion is accompanied by the exploration of the architecture of the code base submitted alongside with this report. Firstly, the methods used in the code are reviewed, and then some results are shown and discussed.

The accompanying code base is written in Python and PySpark. Follow the `README.md` file on how to setup and run the code. The methods of the Apriori algorithm [1], mining association rules, and data utilities are factored into the `apriori.py`, `association_rules.py`, and `utils.py` files. The demonstration of the use of the code can be found in the Jupyter Notebook titled `main.ipynb`.

2 Data

The data used in the code was provided for the assignment. The dataset includes 10 000 baskets of items (with hashed item IDs) with 870 distinct items. The baskets are of varying sizes.

3 Methods and Results

3.1 Discovering Frequent Itemsets with Apriori Algorithm

For finding frequent itemsets in the data set of baskets of items, the Apriori algorithm was used. The workflow of the algorithm is shown in Figure 1:

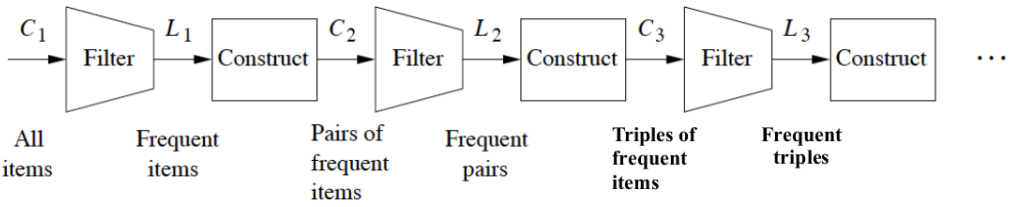

Figure 1: The workflow of the Apriori algorithm (from the lecture notes)

First, the singletons, the frequent itemsets of one item are discovered with the `get_singletons` method. This returns a list of tuples where the first elements are the frequent singletons and the second elements are their respective support. Support is the number of occurrences across the baskets. The support for the discovery of itemsets was set to 1% of the number of baskets, that is, 1000. The number of discovered frequent singletons is 375. In the second stage, using the frequent singletons, a set of candidate pairs are generated with the `construct_itemsets_apriori` method. The number of candidate pairs of items is 70125. The candidate pairs are filtered based on the support threshold (1000 occurrence) yielding 9 new frequent itemsets of size 2 (pairs of items). The frequents pairs are as follows (list of tuples where the first element is the frequent itemset and the second is its support): [({368, 829}, 1194), ({390, 722}, 1042), ({789, 829}, 1194), ({704, 825}, 1102), ({39, 704}, 1107), ({227, 390}, 1049), ({368, 682}, 1193), ({217, 346}, 1336), ({39, 825}, 1187)]. Filtering the 70125 candidate pairs down to 9 frequent itemsets of size 2 takes the most time in the code - around 10 minutes (with PySpark and Python's multiprocessing package). Using the frequent singletons and pairs, candidate triplet itemsets are generated. Importantly, a candidate itemset is valid if all of its subsets are in frequents itemsets discovered before (singletons and pairs in this case). From the 3352 candidate triplets only one triplet meets this requirement, and after filtering the discovered set of frequent triplets is [({704, 825, 39}, 1035)] where the first element of the tuple is the frequent itemset of size 3 and its support is 1035. That is, the only candidate triplet is a frequent itemset. Using the set of frequent singletons and triplets, there are 372 proposed candidate quadruplets, but all of them are pruned away there exits no such itemset with size 4, all the subsets of which are in the previously discovered frequent itemsets. Therefore, this is where the Apriori algorithm finished execution. In summary, the number of discovered frequent itemsets is 385.

3.2 Mining Association Rules from the Frequent Itemsets

The goal is to mine association rules of the form $I \to j$ where $I \cup j$ is a frequent itemset (and hence so are the itemsets I and i) and the confidence of the association rule is the ratio of the support of the frequent itemset $I \cup j$ and the support of I, that is:

$$\text{conf}\,(I \to j) = \frac{\text{support}\,(I \cup j)}{\text{support}\,(I)} \tag{3.1}$$

An association rule is regarded significant is the support of I is above the the support threshold (set to 1000 in the code) and if the confidence score is above the confidence threshold (set to 50% in the code). The association rules discovered for each of the non-singleton frequent items is shown below:

- frequent itemset: {368, 829}: none
- frequent itemset: {722, 390}: none
- frequent itemset: {789, 829}: none
- frequent itemset: {704, 825}
  - association rule: {704} → {825} with confidence=0.6143
- frequent itemset: {704, 39}
  - association rule: {704} → {39} with confidence=0.6171
- frequent itemset: {227, 390}
  - association rule: {227} → {390} with confidence=0.5770
- frequent itemset: {368, 682}: none
- frequent itemset: {217, 346}: none
- frequent itemset: {825, 39}: none
- frequent itemset: {704, 825, 39}
  - association rule: {704} → {825, 39} with confidence=0.5769
  - association rule: {704, 825} → {39} with confidence=0.9392
  - association rule: {704, 39} → {825} with confidence=0.9350
  - association rule: {825, 39} → {704} with confidence=0.8719

The association rules are mined from the frequent itemsets with the `get_association_rules` method.

4 Conclusion

The efficiency of the Apriori algorithm is amazing for discovering frequent itemsets in massive data sets, and it provides a great base for mining association rules.

References

[1] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules in large databases," in *Proceedings of the 20th International Conference on Very Large Data Bases*, ser. VLDB '94, 1994, pp. 487–499, ISBN: 1558601538.