Data Mining, ID2222 HT201

Homework Assignment 2

Frano Rajic, Ivan Stresec

November 16, 2020

1 Objective

Our objective in this homework was to implement the finding of frequent itemsets and generating association rules using the Apriori algorithm on a relatively small dataset of generated transactions.

2 Solution

Using pure Python (and some of its built-in functions) we implemented the Apriori algorithm for finding frequent itemsets and generating association rules.

2.1 Finding frequent itemsets

We created a class `FrequentItemsets` which found all frequent itemsets using the Apriori procedure for some fractional support threshold s.

Simply put, we ran an algorithm increasing the size of itemsets k starting from 1 and counted the number of times any subset of size k appeared. To achieve this, our algorithm had to make a pass over the entire dataset for every k. For $k > 1$, we ignored itemsets which contained any subset which didn't have enough support by generating a list of candidates using the previous frequent itemsets L_{k-1} and singletons L_1. Here we also took care not to generate candidates whose subsets had no support, as they could then have no support themselves either. If no candidates were found, or no found candidates had enough support, the algorithm would terminate, as frequent itemsets larger than the k of our last iteration cannot exist in those cases.

2.2 Association Rules

Much like with finding frequent itemsets, we created a class `AssociationRules` which found all association rules given some confidence threshold c, using the results of a `FrequentItemsets` object.

To determine the association rules of confidence c and with support s, the frequent items of support cs were used. For each of these itemsets, all possible $left \rightarrow right$ combinations were created and checked for these two conditions:

1. $support(left) >= s$
2. $confidence(left \rightarrow right) = support(left \cup right)/support(left) >= c$

The first condition is necessary as the preprocessed itemsets are of support at least cs.

3 Testing and evaluation

We used the dataset *T10I4D100K.dat* that was given in the homework instructions. The dataset consists of 100K generated transaction (baskets), in which below 1000 items are represented by integer hash values. To test and evaluate our solution, we used this dataset to run our algorithms and then look at the results that it gave.

3.1 Finding frequent itemsets

For a support threshold s larger than 8%, no supported itemsets were found. When using slightly lower support thresholds, between 2% and 7%, the algorithm would find only frequent singletons. Finally, when taking a standard support threshold of 1%, (a good "brick-and-mortar" value) the algorithm would give a satisfactory result, with 379 frequent singletons, 9 frequent itemsets of size 2 and 1 frequent itemset of size 3. We can see the frequent itemsets in Figure 1.

```
For k = 2:
((39, 704), 0.01107) ((704, 825), 0.01102) ((39, 825), 0.01187) ((227, 390),
 0.01049) ((789, 829), 0.01194) ((368, 829), 0.01194) ((217, 346), 0.01336)
 ((368, 682), 0.01193) ((390, 722), 0.01042)

For k = 3:
((39, 704, 825), 0.01035)
```

Figure 1: Results for the *T10I4D100K.dat* dataset with a support threshold of 1%. Frequent itemsets of size 2 and 3; inside each set of brackets there is first a tuple defining the itemset and then, separated with a comma, it's support relative to the whole dataset

Using support thresholds under 1% would result in a much larger number of frequent itemsets. For example, with a support threshold of 0.5% for itemsets of size 1, 2, 3, 4, and 5 we found 569, 342, 110, 43, and 9 frequent itemsets, respectively. This also resulted in a longer execution time of the algorithm.

3.2 Association rules

Found association rules, as well as their confidence and interest values are shown in Figure 2. For this dataset, a confidence threshold of 75% and support threshold of 1%, we have found 5 rules. All rules found are of high positive interest. For example, the rule $\{227, 390\} \rightarrow 722$ has high confidence of 86.5% and a interest of 0.806. The high interest suggests that the item 772 appears much more frequently that expected in an average basket if the items $\{227, 390\}$ are also present in the basket.

```
Determined additional itemsets with support c*s=0.0075
Association rules for c=0.75 and s=0.01
{'{227, 390} ⟶ {722} conf:0.8646329837940897 interest: 0.8061829837940897',
 '{704, 39} ⟶ {825} conf:0.9349593495934959 interest: 0.9041093495934959',
 '{704, 825} ⟶ {39} conf:0.9392014519056261 interest: 0.8966214519056261',
 '{722, 390} ⟶ {227} conf:0.8704414587332053 interest: 0.8522614587332054',
 '{825, 39} ⟶ {704} conf:0.8719460825610783 interest: 0.8540060825610784'}
```

Figure 2: Results for the *T10I4D100K.dat* dataset. Association rules found for confidence of 75% and support of 1%. Calculated interest of the rule is also shown in the output. "conf" stands for confidence.

When lowering the confidence threshold from 75% to 50%, 116 rules have been found. On the other hand, for a higher confidence threshold of 90%, only 2 rules have been found, as expected from figure 2.

4 Conclusion

This homework was a good demonstration of the market-basket analysis and using the Apriori algorithm. Concerning implementation, though we did not directly face the issue of the memory bottleneck, as we did not use very large datasets, we did face a running time bottleneck in the computation of the frequent itemsets, especially when using very low support thresholds. In comparison, generating the association rules from frequent itemsets has been much faster.