Task 1

FREQUENT ITEMSETS

A frequent itemset has a support value which is greater than the minimum support value. The support value of an itemset is its occurrence compared to the number of transactions.

When minsup = 0.15:
0.15 * 5 = 0.75, 0.75 < 1 (if an itemset has at least one occurrence, it is a frequent itemset)

There were 6 unique values in the transactions [“A”, “B”, “C”, “D”, “E”, “F”]. From here, there are 63 combinations and thus 63 possible candidate itemsets. Out of these 63 possible itemsets, 35 had an appearance in at least one of the 5 transactions. This means that there were 35 frequent itemsets when minsup is 0.15.

FREQUENT RULES

From the 35 frequent itemsets, there were 190 possible association rules using (2^k -2 for every frequent itemset). This was by obtained by finding subsets of itemsets X and Y such that X and Y are non-empty sets and that the sets can be expressed in the form X implies Y and Y implies X. The reason for taking subsets is because the anti-monotone property ensures that all subsets of a frequent itemset will be a frequent itemset.

The confidence of a rule is obtained by doing P(X and Y) / (Y) where itemset X implies itemset Y. A frequent rule must have a confidence value higher than 0.80 when minconf = 0.80. A total of 97 rules met this requirement meaning that there were 97 total frequent rules.

Task 2

Implementation of the Apriori algorithm:

For convenience, each element in the transactions have been converted into a string number. For example, the item “laundry needs” corresponds to a “0” and the item “dairy foods” corresponds to a “1”. By doing this, it was easier to perform operations. From here, every unique value in all the transactions were taken and were pruned into a list such that the pruned list had k=1 frequent items.

For my implementation pruning took place at the end of the while loop, rather than the start of the while loop. Pruning was done by getting the non-frequent itemsets in size-k and removing supersets of these size-k non-frequent itemsets in any k+1 candidate itemsets. This meant that the pruned list would be sent to the top of the list through the while loop.

From a k-size candidate list of itemsets, the support of each candidate was counted. I did this by using a dictionary which had an item as a key and the transactions which it appeared in as the value. Thus, the support of an itemset was the number of common transactions where all the items had appeared in. Because this was always constant, I also created a dictionary would store this information for each itemset because it could be a subset of a possible k+1 candidate itemset. Only the itemsets which support met the minsup requirement were added to the results list. Furthermore, only the itemsets which met the minsup requirement were used to generate k+1 candidate itemsets. This process continued as the value of k increased until there was no more k+1 candidate itemsets possible.

From here, the possible rules were generated using combinations in itertools. There were to be 2^k-2 possible rules for each frequent itemset because the rules are asymmetrical. From here, each candidate rule was compared to the standards of confidence and lift. If the confidence of a rule was greater than the minconf and the lift was greater than the minlift. The rule could be considered a frequent rule.

Task 3

Output was produced with minsup = 0.15, minconf = 0.80 and minlift =0
There were a total of 97 rules produced and 35 frequent itemsets.
Please view the output at the bottom of my Jupyter notebook

Task 4

Interesting discovery #1.
The item “bread and cake” appears in 17/17 appearances as the Y itemset for frequent rules at minsup= 0.30 and minconf = 0.80, minlift = 0. This is likely because of the “bread and cake” appears most out of all the transactions in the supermarket.csv file. Because confidence is calculated though $\sigma(\mathrm{X} \cup \mathrm{Y}) / \sigma(X)$, a frequent rule will have a high confidence value if the size of set Y is big because this ensures the difference in the numerator and denominator is big. As predicted, the “bread and cake” item is appearances in 3330 unique transactions.

Interesting discovery #2.
15/17 frequent rules at minsup= 0.30 and minconf = 0.80, minlift = 0 are rules with 3 elements. Only (*total = high -> bread and cake*) and *(margarine -> bread and cake*) are the two frequent rules with 2 elements. The lack of size 2 rules suggests that there are many overlaps between the individual items with support > 0.30, such that the $\sigma(\mathrm{X} \cup \mathrm{Y})$ is not much different to $\sigma(\mathrm{X})$. This difference in $\sigma(\mathrm{X} \cup \mathrm{Y})$ and $\sigma(\mathrm{X})$ when $\mathrm{X} \cup \mathrm{Y}$ is size three must be greater than when $\mathrm{X} \cup \mathrm{Y}$ is size two.

Task 5

Minrelativesup

Due to the anti-monotonic property of support, the support of a superset cannot exceed the support of a subset. This is a foundational concept in the Apriori algorithm, thus the introduction of the minrelativesup variable will influence the generation of itemsets and association rules. The minrelativesup variable does not influence the anti-monotonic property of support when minrelativesup < 1. This is because the minrelativesup variable is obtained from the following fraction: support of current k-itemset / maximum support of size(k-1) subset. For the minrelativesup variable to be less than one, maximum support of size(k-1) subset must be greater than the support of current k-itemset. This is consistent with the anti-monotonic property of support because the support of a subset if always greater than the support of a superset.

When the minrelativesup variable is greater than 1, there will be no frequent itemsets and therefore no association rules when k > 2. For the fraction to equal a value greater than 1, the denominator must be greater than the numerator. This means that for an itemset to be a frequent itemset when minrelativesup > 1: support of k-itemset must be greater than the maximum support of size(k-1) subset. This is a contradiction to the anti-monotonic property of support. In this case, the only possible frequent itemsets can be k=1 or k=2 and frequent rules k=2.

As such, the minrelativesup variable does influence the generation of frequent itemsets and frequent rules.

When minsup = 0.15, minconf = 0.80, minlift = 0 and minrelativesup = 0: There are 2066 frequent itemsets and 955 frequent rules

When minsup = 0.15, minconf = 0.80, minlift = 0 and minrelativesup = 0.1: There are 465 frequent itemsets and 15 frequent rules

When minsup = 0.15, minconf = 0.80, minlift = 0 and minrelativesup = 0.5: There are 423 frequent itemsets and 6 frequent rules

As noticeable the increase in minrelativesup causes a decrease in the number of frequent itemsets and number of frequent rules. However, a high value of minrelativesup does not cause 0 frequent itemsets and 0 frequent rules because the minrelativesup only affects k>2. This means that there can be frequent itemsets from k=1 and k=2 and frequent rules from k=2.

Message to marker: Thank you for marking my assignments this semester. Grades aside, I thoroughly enjoyed the assignments. They were definitely challenging but I am proud to see how much I have accomplished through this semester. Another big thanks to the COMPSCI361 staff for engaging me into Machine Learning. This is definitely something that interests me and something I wish to do as a career as I enjoy puzzles and challenges. Thank you for inspiring me.

- Byung Joe Kim