CS5600 - Data Mining

Assignment 2

Frequent Pattern Mining

Group 3

Kritik Agarwal	CS23MTECH11009
Darpan Gaur	CO21BTECH11004
Hari Priyanka Allam	CC23M24P100001
Maloth David	CS21BTECH11035

Exercise 1

Used smpf algorithms Apriori, FPGrowth_itemsets, and Eclat to compute frequent itemsets for retail1.txt and retail2.txt datasets.

Minimum support threshold (0.05%, 1%, 2%, 3%, 5%, 7%).

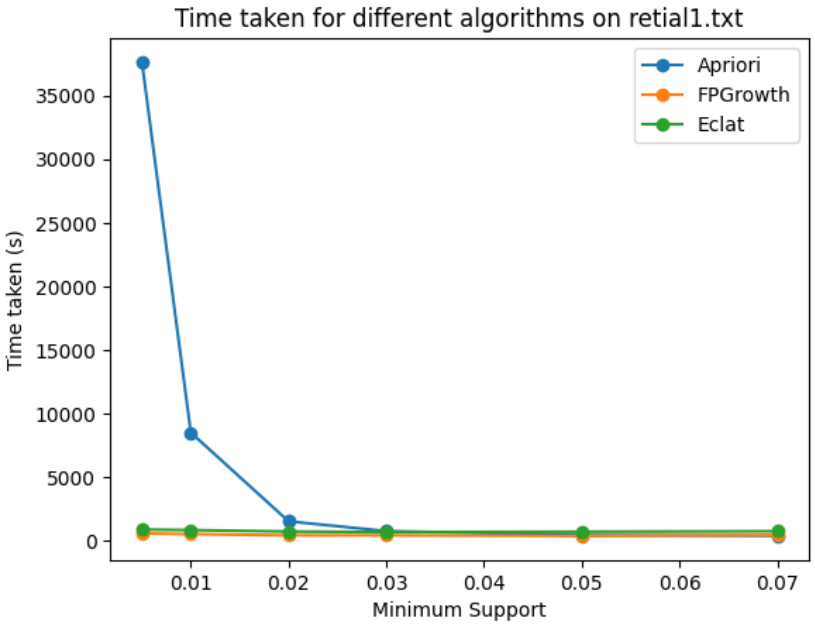

Figure 1: Minimum support vs Computational time for retail1.txt

Observations: retail1.txt

- For minsup ($< 5\%$), Apriori performs worst, while FPGrowth_itemsets performs best, as Apriori use join and prune technique which is computationally expensive.
- As minsup increases performance of Apriori improves.

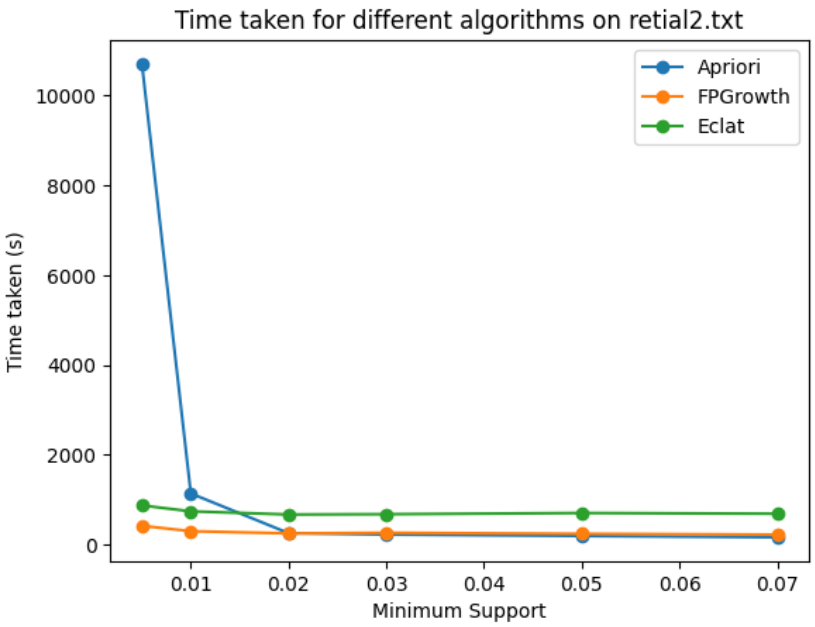

Figure 2: Minimum support vs Computational time for retail2.txt

Observations: retail2.txt

- Apriori performs worst for minsup (0.05%, 1%) and FPGrowth performs best.
- Eclat performs worst for minsup ($> 2\%$), while FPGrowth and Apriori performance are comparable, as with large dataset there is less increament in time with Apriori as compared to Eclat and FPGrowth.

Exercise 2

Time taken by all three algorithms is almost similar for all the cases.

Algorithm		**minsup = 0.05%**	**minsup = 1%**	**minsup = 2%**
FPGrowth	Time	491	430	419
	Itemsets	396	140	45
FPClose	Time	491	419	415
	Itemsets	5	5	5
FPMax	Time	470	451	413
	Itemsets	5	5	5

Table 1: Time and Itemsets on retail1.txt

Algorithm		**minsup = 0.3%**	**minsup = 0.5%**	**minsup = 1%**
FPGrowth	Time	393	360	366
	Itemsets	24	10	5
FPClose	Time	415	379	348
	Itemsets	5	5	5
FPMax	Time	415	381	338
	Itemsets	5	5	5

Table 2: Time and Itemsets on retail1.txt (continued)

Algorithm		**minsup = 0.05%**	**minsup = 1%**	**minsup = 2%**
FPGrowth	Time	418	290	246
	Itemsets	580	159	55
FPClose	Time	443	331	254
	Itemsets	13	13	13
FPMax	Time	455	325	246
	Itemsets	13	13	13

Table 3: Time and Itemsets on retail2.txt

Algorithm		**minsup = 0.3%**	**minsup = 0.5%**	**minsup = 0.7%**
FPGrowth	Time	244	220	224
	Itemsets	32	16	13
FPClose	Time	243	228	229
	Itemsets	13	13	13
FPMax	Time	233	233	224
	Itemsets	13	13	13

Table 4: Time and Itemsets on retail2.txt (continued)

Exercise 3

Frequent 1-itemsets	Frequent 2-itemsets	Frequent 3-itemsets
[1126], [2375], [2183], [1097], [1084], [1394], [644], [2316], [2363], [1142], [349], [1816], [1943], [2096], [1017], [593], [209], [298], [881], [508], [1456], [1182], [358], [114], [1534], [2339], [1131], [1324], [2395], [762], [324], [1680], [574], [1046], [181], [461], [1834], [2202], [2400], [1336], [2060], [225], [720], [754], [1012], [1407], [547], [1605], [1235], [781], [2398], [945], [2399], [189], [1264], [1721], [1215], [308], [2129], [90], [426], [1582], [479], [116], [1594], [860], [489], [1989], [2027], [2428], [983], [2231], [2380], [708], [1862], [2352], [1165], [21], [1617], [304], [355], [99], [1995], [2407], [530], [2232], [1825], [396], [1603]	[1534, 1603], [1097, 1126], [90, 1534], [225, 1336], [116, 426], [225, 720], [720, 1603], [1097, 2183], [426, 479], [21, 1534], [114, 1534], [1534, 1943], [426, 1534], [114, 1943], [99, 1534], [1943, 2232], [99, 1943], [1126, 2183], [479, 1534], [1816, 1834], [1394, 1989], [349, 1142], [1680, 1721], [90, 1943], [1084, 1126], [1084, 2183], [1142, 2363], [754, 1012], [720, 1336], [349, 2363], [1084, 1097], [225, 508], [1534, 2232], [1534, 1582], [225, 1215], [116, 479]	[90, 1534, 1943], [1534, 1943, 2232], [426, 479, 1534], [99, 1534, 1943], [1084, 1126, 2183], [225, 720, 1336], [1097, 1126, 2183], [116, 426, 479], [349, 1142, 2363]

Table 5: Frequent Itemsets

Observations of MSApriori algorithm:

- Frequent 1-itemsets show which individual products or items are most common.Frequent 2-itemsets reveal associations between items that are commonly purchased or used together.Frequent 3-itemsets provide deeper insights into combinations of items that co-occur, potentially useful for market basket analysis or identifying relationships in the data.

- The size of the frequent itemsets grows progressively from single items to pairs and then to triplets.The frequency count diminishes as the size of the itemset increases, which is typical in frequent itemset mining because it becomes rarer for larger sets to meet the minimum support.
- The frequent 2-itemsets and 3-itemsets provide valuable insight into item relationships. These associations can be useful for creating recommendations or identifying products that tend to be purchased together in a retail context.

Exercise 4

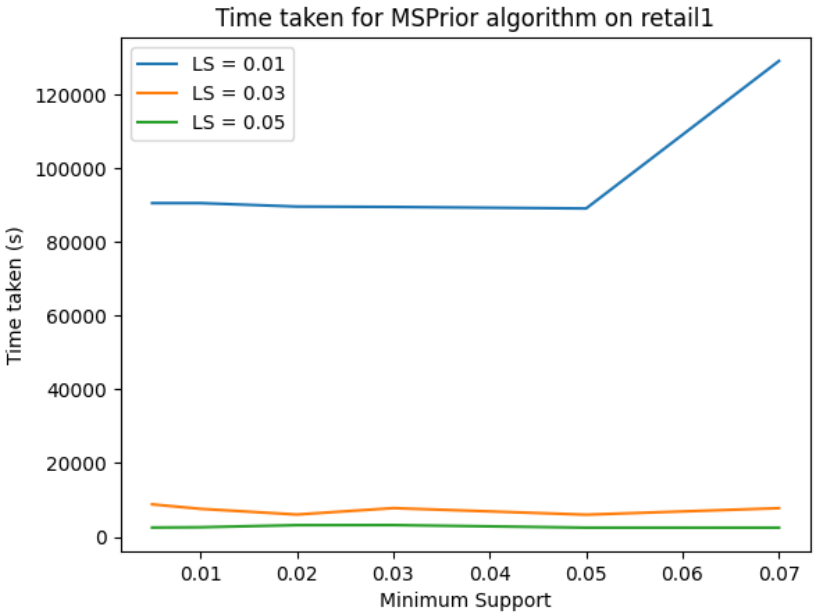

Figure 3: Time taken for MSPrior algorithm on retail1.txt

Observations of MSApriori algorithm from SPMF library: retail1.txt

- The highest execution times are observed with LS = 0.01 across all minsup values, with times exceeding 89,000 seconds and reaching a peak of 129,007 seconds at minsup = 7%.
- Lower LS values consistently resulted in longer execution times, indicating that a smaller LS results in more granular and intensive processing.
- With LS = 0.03 and 0.05, the execution times were significantly lower (e.g., 2,550–8,833 seconds for LS = 0.03 and 2,517–3,199 seconds for LS = 0.05).

LS values		**minsup = 0%**	**minsup = 1%**	**minsup = 2%**
0.1	Time(s)	90481	90480	89546
	Itemsets	140	140	140
0.03	Time(s)	88833	7617	6072
	Itemsets	24	24	24
0.05	Time(s)	2550	2628	3199
	Itemsets	10	10	10

Table 6: Time and Itemsets on retail1.txt

LS values		**minsup = 3%**	**minsup = 5%**	**minsup = 7%**
0.01	Time(s)	89441	89022	129007
	Itemsets	140	140	140
0.03	Time(s)	7814	379	348
	Itemsets	24	24	24
0.05	Time(s)	3216	2521	2517
	Itemsets	10	10	10

Table 7: Time and Itemsets on retail1.txt (continued)

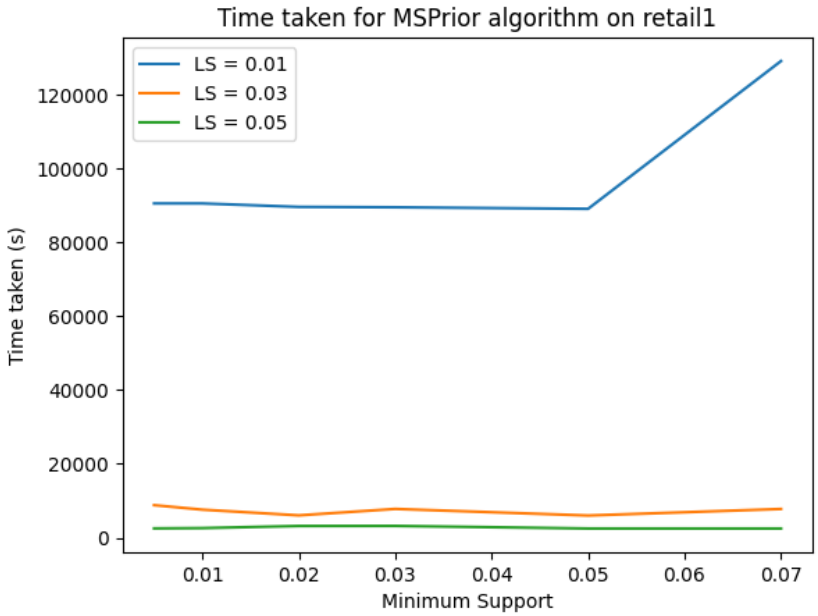

Figure 4: Time taken for MSPrior algorithm on retail2.txt

Observations of MSApriori algorithm from SPMF library: retail2.txt

- Similar trends were observed, with LS = 0.01 taking the longest time, though the absolute values were lower than retail1.
- The execution times for LS = 0.03 and 0.05 were consistently shorter, ranging between 1,141 *2,585 seconds for LS = 0.03 and 923* 1,165 seconds for LS = 0.05.

LS values		**minsup = 0%**	**minsup = 1%**	**minsup = 2%**
0.1	Time(s)	20400	18800	16296
	Itemsets	159	159	159
0.03	Time(s)	1141	1380	1231
	Itemsets	32	32	32
0.05	Time(s)	1165	1088	1061
	Itemsets	16	16	16

Table 8: Time and Itemsets on retail2.txt

LS values		**minsup = 3%**	**minsup = 5%**	**minsup = 7%**
0.01	Time(s)	15609	16120	14904
	Itemsets	159	159	159
0.03	Time(s)	1407	1275	2585
	Itemsets	32	32	32
0.05	Time(s)	1059	923	1128
	Itemsets	16	16	16

Table 9: Time and Itemsets on retail2.txt (continued)

Number of Itemsets:

- Across both datasets, LS = 0.01 consistently found the most itemsets (140 for retail1 and 159 for retail2), implying more comprehensive mining at a lower threshold.
- LS = 0.03 found fewer itemsets (24 for retail1, 32 for retail2), and LS = 0.05 identified the least (10 for retail1, 16 for retail2).

- The count of itemsets remained consistent across varying minsup values for each LS, showing stability in the number of frequent itemsets found.
- The results indicate that lower LS values lead to longer execution times and more frequent itemsets, reflecting a more thorough analysis of the dataset. However, this comes with a significant computational cost.
- Higher LS values (e.g., 0.03 and 0.05) result in shorter execution times and fewer itemsets, suggesting that these configurations may be more suitable for applications requiring faster processing at the cost of finding fewer itemsets.

Exercise 5

Part (a)

The top five itemsets based on their support values are as follows:

- (`1534`, `1943`): 29,159 occurrences
- (`1816`, `1834`): 20,265 occurrences
- (`225`, `1215`): 18,970 occurrences
- (`1394`, `1989`): 15,134 occurrences
- (`1534`, `1582`): 14,136 occurrences

The `cannot_be_together` constraint was applied to these itemsets, ensuring that none of them appear together in the final results. Additionally, a `must_have` constraint was specified, requiring that any frequent itemset must contain at least one of the following items: {1816, 225, 1394, 1534}.

Part (b)

After applying the `must_have` constraint, the following frequent 1-itemsets were obtained:

- {1394}
- {1816}
- {1534}
- {225}

The total number of frequent itemsets obtained under this constraint was 4.

Part (c)

Retail1 Dataset

- **SPMF MS-Apriori**:
  - Frequent itemsets count: 140
  - Maximum memory usage: 833.05 MB
  - Total time: 87.16 seconds
- **Custom MS-Apriori without Constraints**:
  - Frequent itemsets count: 120
- **Difference in Itemset Counts (SPMF - No Constraints)**: 20

Retail2 Dataset

- **SPMF MS-Apriori**:
  - Frequent itemsets count: 159
  - Maximum memory usage: 276.53 MB
  - Total time: 17.60 seconds
- **Custom MS-Apriori without Constraints**:
  - Frequent itemsets count: 128
- **Difference in Itemset Counts (SPMF - No Constraints)**: 31

Part (d)

The execution times for the MS-Apriori algorithm under different configurations were as follows:

- **No Constraints**: 0.08 seconds
- **With Constraints**: 0.09 seconds
- **SPMF with Constraints as Post-Filtering**: 18.97 seconds

Frequent Itemsets Obtained

Frequent 1-itemsets: {9}, {19}, {32}, {31}, {36}, {37}, {38}, {39}, {41}, {45}, {48}, {49}, {60}, {65}, {78}, {79}, {101}, {89}, {110}, {117}, {123}, {147}, {161}, {170}, {175}, {179}, {185}, {201}, {225}, {237}, {242}, {249}, {258}, {255}, {264}, {270}, {271}, {286}, {301}, {310}, {338}, {413}, {438}, {475}, {479}, {522}, {533}, {548}, {589}, {592}, {604}, {677}, {740}, {783}, {824}, {956}, {1004}, {1146}, {1327}, {1393}, {2238}, {2958}, {3270}, {10515}, {12925}, {13041}, {14098}, {15832}, {16010}, {16217}.

Frequent 2-itemsets: Examples include {39, 438}, {48, 101}, {48, 270}, among others.

Frequent 3-itemsets: Examples include {38, 39, 170}, {39, 48, 475}, among others.

The number of itemsets meeting the post-filtering constraints using SPMF was 21.

Conclusion

The analysis highlights that applying constraints within MS-Apriori significantly affects the frequency and count of resulting itemsets. Execution times for custom MS-Apriori with and without constraints remained low, while post-filtering using SPMF required notably higher computational time.