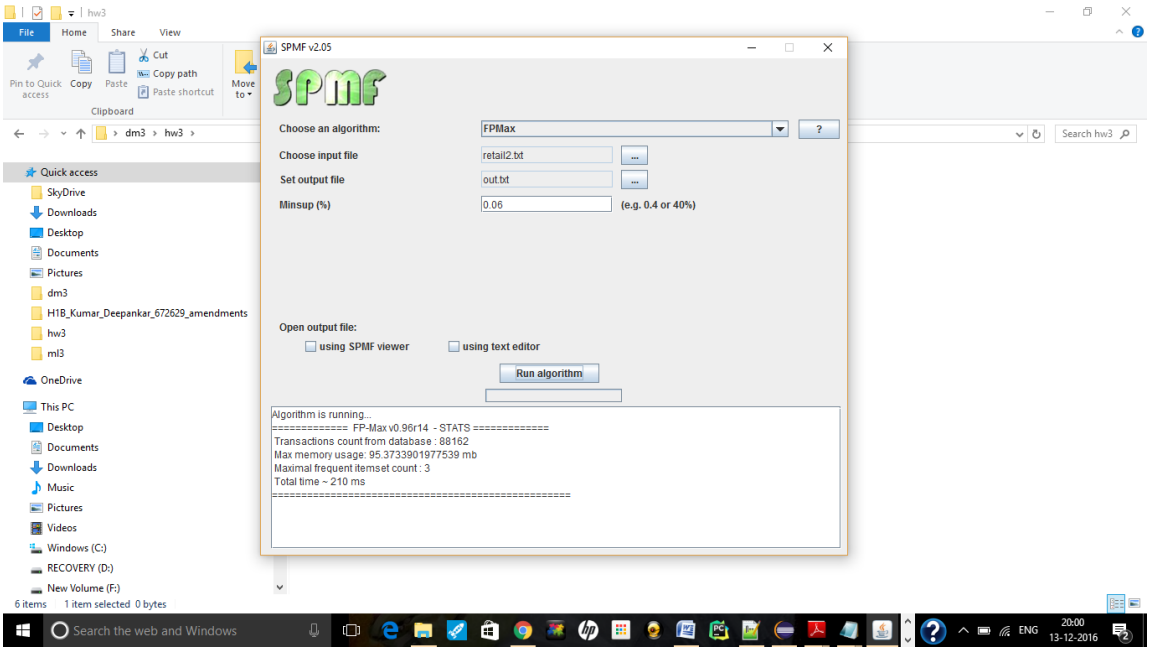
# Exercise 1

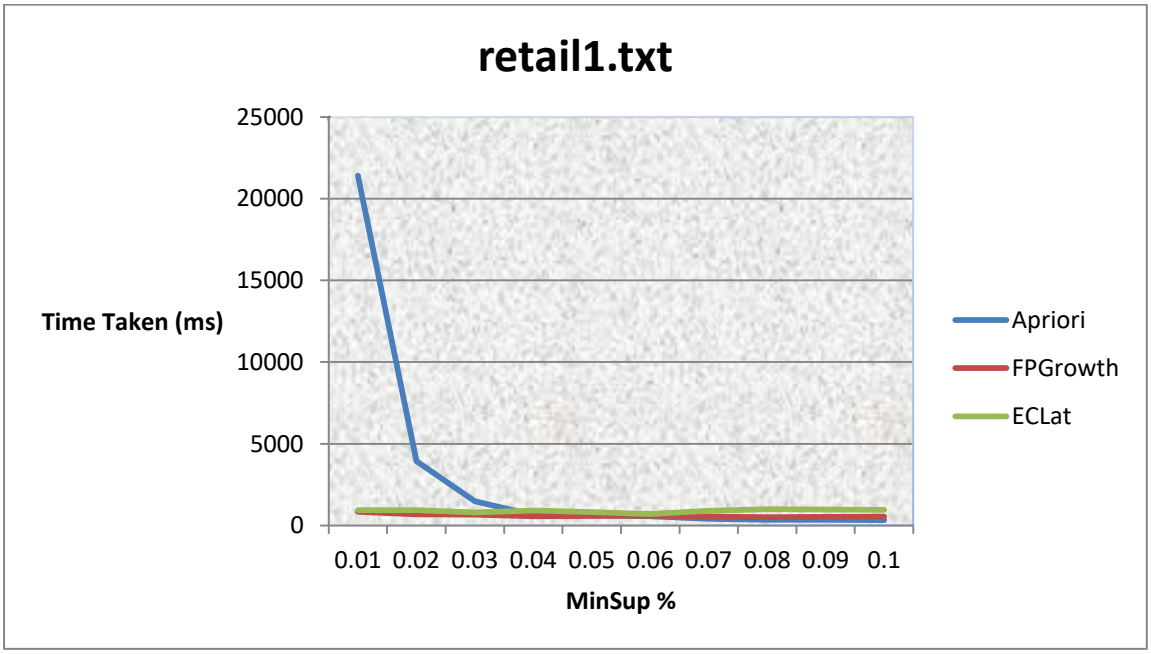
**Smpf.jar** is used.

The total time and Frequent item set counts are logged into Excel and the Excel charts are pre pared.



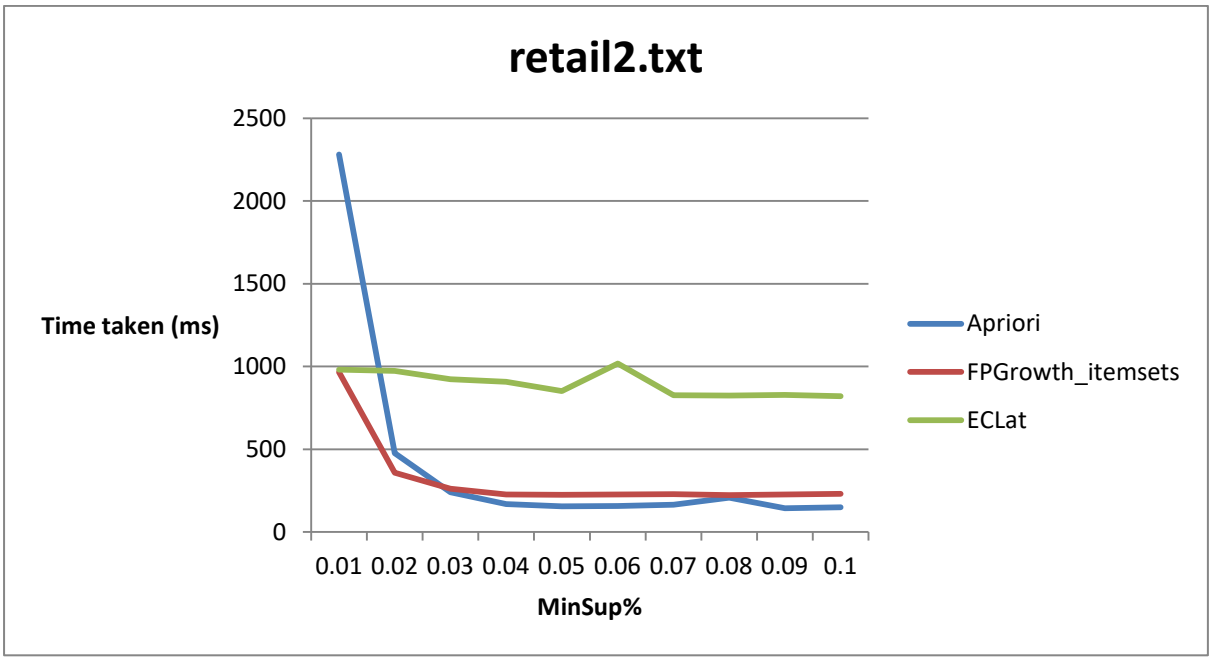

Minimum Support is varied from 0.01 to 0.10 is step of 0.01 and the frequent item sets are obtained and plotted against.

For Dataset retail1.txt



Minimum Support is varied from 0.01 to 0.10 is step of 0.01 and the frequent item sets are obtained and plotted against.

For Dataset retail2.txt




From the plots it is evident that for low value of Minimum Support Apriori algorithm doesn't perfom well. But at relative high value of minimum support the performance remains same.

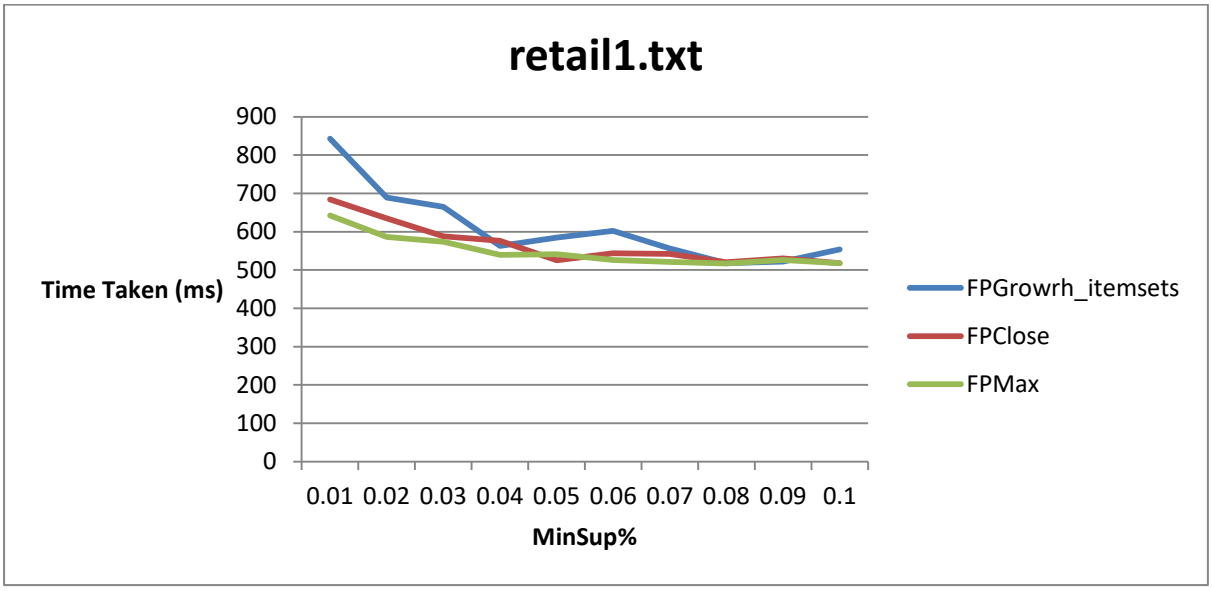
For relative high value of minimum support all three algorithms performance remain similar.

But for dataset retail2.txt Apriori algorithm performs better.

## Exercise 2

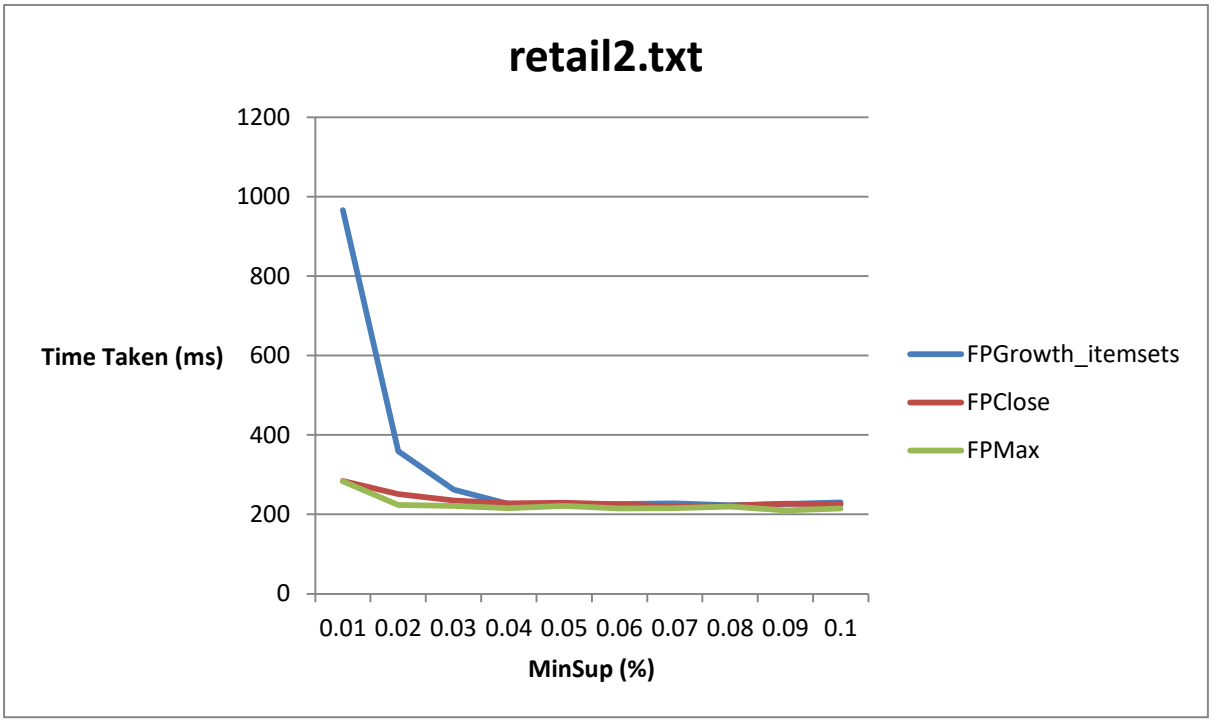
Minimum Support is varied from 0.01 to 0.10 is step of 0.01 and the frequent item sets are obtained and plotted against.

For Dataset retail1.txt




| | min-sup = 0.5% | min-sup = 1% | min-sup = 2% | min-sup = 5% |
|---|---|---|---|---|
| FP-growth | 674 ms | 625 ms | 588 ms | 538 ms |
| | 396 | 140 | 45 | 10 |
| FP Close | 679 ms | 633 | 595 | 540 ms |
| | 323 | 131 | 45 | 10 |
| FPMax | 681 ms | 662 ms | 577 ms | 626 ms |
| | 175 | 80 | 33 | 8 |

For Dataset retail2.txt




| | min-sup = 0.5% | min-sup = 1% | min-sup = 2% | min-sup = 5% |
|---|---|---|---|---|
| FP-growth | 350 ms | 257ms | 235ms | 221ms |
| | 580 | 159 | 55 | 16 |
| FP Close | 372 ms | 270 ms | 236 ms | 215 ms |
| | 580 | 159 | 55 | 16 |
| FPMax | 383 ms | 264 ms | 229 ms | 223 ms |
| | 284 | 78 | 20 | 4 |

FPClose and FPMax perform better than FPGrowth at low level of support.

## Exercise 3 & 4

Please find DM folder for java implementation of MS-Apriori algorithm.

## Exercise 5

**a)**

Top 5 itemsets with maximum support are:
1084 1126 2183 #SUP: 6396
426 479 1534 #SUP: 6598
225 1816 1834 #SUP: 6959
90 1534 1943 #SUP: 7294
349 1142 2363 #SUP: 9505

Cannot be together sets
(1084 1126 2183), (426 479 1534),( 225 1816 1834), (90 1534 1943),
(349 1142 2363)
Must have set:
(90, 225, 349, 1084, 426)
The above item set is obtained taking item with least min id from the 5 itemsets and then sorting in ascending order.

**b)**

The parameters used for the custom implementation of MSApriori Algorithm in Ex 3 are:

$$\delta = 0.5, \varphi = 5\%, LS = .01$$

The must have set is (90, 225, 349, 1084, 426)
The number of obtained frequent itemsets is 12.
Level1
---
90 #SUP: 18695
225 #SUP: 51911
349 #SUP: 11697
426 #SUP: 21461
1084 #SUP: 17037
Level 2
---
116 426 #SUP: 5951
1142 349 #SUP: 9505
2363 349 #SUP: 9505
479 426 #SUP: 9752
2183 1084 #SUP: 7561

Level 3:
---
116 479 426 #SUP: 5951
1142 2363 349 #SUP: 9505

**c)**

The parameters used for the custom implementation of MSApriori Algorithm in Ex 4 are:

$$\delta = 0.5, \varphi = 5\%, LS = .01$$

**Number of Frequent Itemsets obtained are 105**

Ran MSApriori algorithm of SPMF with
Beta = 0.5 and LS =.01 Here is the output:

========== MSAPRIORI - STATS ==========
The algorithm stopped at level 4, because there is no candidate
**Frequent itemsets count : 121**
Maximum memory usage : 1086.2515029907227
mb Total time ~ 62787 ms

**d)**

| Algorithm | Without Filtering | With Filtering(item constraint) |
|---|---|---|
| Custom Implementation | 17639 ms | 17041 ms |
| SPMF Implementation | 62787 ms | |