HyperClique Pattern Discovery

Data Mining Assignment - 3

Anchit Gupta - 22 October 2019

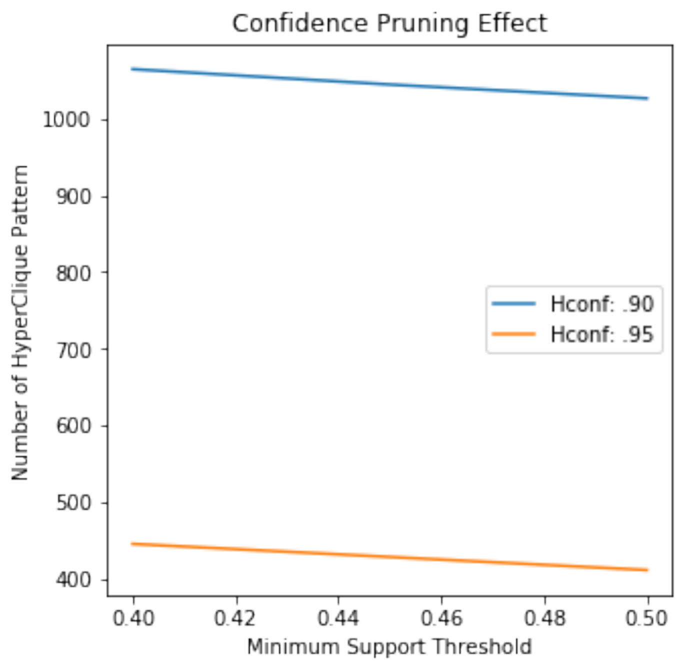
Introduction

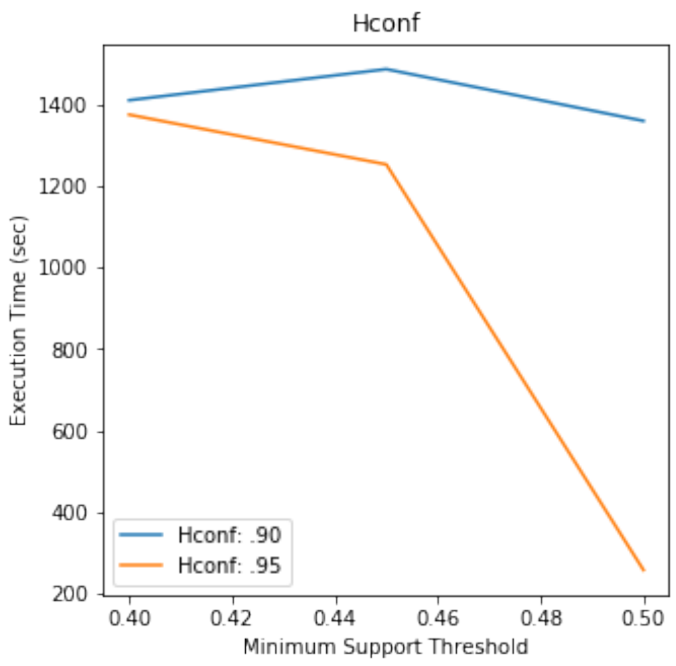
The Hyperclique Pattern Discovery algorithm is one of the most effective algorithm in order to find out the best possible candidates from the datasets, and it's so effective that it can work well even in when the support values are low.

In the given datasets I have experimented on the **pumsb** dataset from out of four datasets which are given to us. Reason for choosing the pubs dataset is that it can be easily verified as in the paper the researchers have taken many examples and plotted graphs of this pubs dataset.

Insight

In this task I have used the used minimum h-confidence as 0.93 and 0.95 to calculate hyper clique patterns. These are the different patters which are of different sizes.





Graphs Plotted: (1) Graph showing theNumber of patterns generated by hyperclique miner algorithm

(2) Graph showing the execution time of hyperclique miner algorithm

In the first graph it can be clearly seen that the decreasing the value of the H-confidence leads to more number of hyper clique patterns in the dataset.
While the second graph is showing the time taken to execute on different H-confidence value for the given values of the threshold. As the number of patterns in the higher H-confidence the pruning of possible candidates is more resulting the lesser time to process as number of left possible candidates are left less.

```
H-conf  0.93
Support  0.4
Size:  1 Patterns: 71
Size:  2 Patterns: 160
Size:  3 Patterns: 345
Size:  4 Patterns: 319
Size:  5 Patterns: 142
Size:  6 Patterns: 26
Size:  7 Patterns: 1
Size:  8 Patterns: 0
Overall Size:  1064
```

```
Support  0.45
Size:  1 Patterns: 63
Size:  2 Patterns: 153
Size:  3 Patterns: 341
Size:  4 Patterns: 318
Size:  5 Patterns: 142
Size:  6 Patterns: 26
Size:  7 Patterns: 1
Size:  8 Patterns: 0
Overall Size:  1044
```

```
Support  0.5
Size:  1 Patterns: 52
Size:  2 Patterns: 147
Size:  3 Patterns: 340
Size:  4 Patterns: 318
Size:  5 Patterns: 142
Size:  6 Patterns: 26
Size:  7 Patterns: 1
Size:  8 Patterns: 0
Overall Size:  1026
```

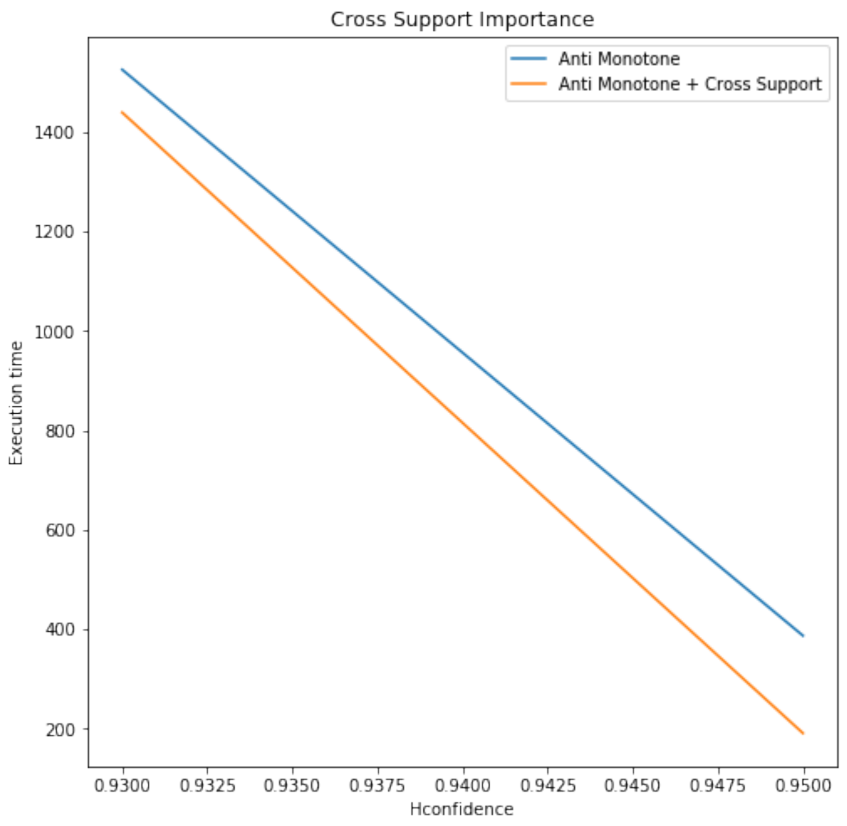
```
H-conf  0.95
Support  0.4                            Support  0.45
Size:  1 Patterns: 71                   Size:  1 Patterns: 63
Size:  2 Patterns: 129                  Size:  2 Patterns: 122
Size:  3 Patterns: 143                  Size:  3 Patterns: 141
Size:  4 Patterns: 80                   Size:  4 Patterns: 80
Size:  5 Patterns: 21                   Size:  5 Patterns: 21
Size:  6 Patterns: 2                    Size:  6 Patterns: 2
Size:  7 Patterns: 0                    Size:  7 Patterns: 0
Overall Size:  446                      Overall Size:  429
```

```
Support  0.5
Size:  1 Patterns: 52
Size:  2 Patterns: 117
Size:  3 Patterns: 140
Size:  4 Patterns: 80
Size:  5 Patterns: 21
Size:  6 Patterns: 2
Size:  7 Patterns: 0
Overall Size:  412
```

Insight - The Effect of the Cross Support Pruning

The Hyper Clique pattern algorithm follows the property of anti-monotone and cross-support pruning. So, in this I try to check whether the performance of the Hyper Clique algorithm how much affected due to removing of cross support property from it.

For this experiment I run my algorithm on the different input scheme to optimised considering my machine configurations. Here I have taken H-Confidence value 0.93 and 0.95 and single support 0.95 to check how much more time it take to run the for to generate the Hyper Clique Patterns and what's the difference in number of patterns.



Graph showing the comparison between the time taken by algorithm to run with and without cross support property.

Here, is can be seen that graph is tends to meet near the top left corner, well it's due to the fact that the cross support in that case not helping very much but as you see in the right bottom the difference is nearly 200 seconds which when done on more lesser support can leads to higher number.

Algorithm

So, although the research paper provided us the algorithm how we can calculate the hyper clique patterns from the given dataset. But this section I am going to explain the approach that I have used in this task.

Steps performed to generate the output

1. Firstly I imported the pumsb data using the pandas library and convert it into the pandas dataframe.
2. Then I get all the transaction data into the numpy array and find out the unique elements form the transaction dataset.
3. Then create the dataset of asymmetric type of the given data.
4. Using this asymmetric dataframe I calculate the frequency of items and support of individual items.
5. Then,

```
    for loop through the all minimum h-confidences:
        for loop through the all minimum support:
            get all the individual items from dataset whose minimum support satisfies:
                for loop through size = 2 to unique items length:
                    get cross support patterns then prune, merge and sort them
                    according to the support, cross support and anti-monotone
                    property:
```

6. Then using the results obtained in step 5 generate the graphs.
7. (To verify the importance of cross support) Again the step 5 but without the cross support property.