A) Two types of candidate and frequent itemsets were implemented ‘F(k-1)*F(k-1)’ and ‘F(k-1)*F(1)’

**In addition to this , Brute force method is also implemented which doesn’t filter the candidate sets on the basis of min support count and return all possible itemsets.**

Here is the result of running the data sets on dataset ‘car.data’
[http://archive.ics.uci.edu/ml/datasets/Car+Evaluation](http://archive.ics.uci.edu/ml/datasets/Car+Evaluation)

Minimum Support value used: 0.4

| Dataset Name | Number of Candidates Itemsets by Brute force | Number of Candidates Itemsets by F(k-1)*F(1) | Number of Candidates Itemsets by F(k-1)*F(k-1) | Number of Frequent Itemsets by Brute force | Number of Frequent Itemsets by F(k-1)*F(1) | Number of Frequent Itemsets by F(k-1)*F(k-1) |
|---|---|---|---|---|---|---|
| car.data | 560 | 42 | 23 | 13 | 13 | 13 |

Output Screenshot Command prompt:

```
Total number of frequent itemsets generated: 13
Number of candidate sets generated by Brute Force :  560
Number of candidate sets generated by F(k-1)*F(1) : 42
Number of candidate sets generated by F(k-1)*F(k-1) : 23
```

B)

| Dataset | Number of rows | Number of Candidates Itemsets by F(k-1)*F(1) | Number of Candidates Itemsets by F(k-1)*F(k-1) | Number of frequent Itemsets |
|---|---|---|---|---|
| car.data (min support= 0.2) | 1728 | 389 | 126 | 13 |
| car.data (min support= 0.5) | 1728 | 19 | 15 | 7 |
| car.data (min support= 0.6) | 1728 | 6 | 6 | 4 |
| flare.data2(min support= 0.2) | 1066 | 646 | 142 | 111 |

| | | | | |
|---|---|---|---|---|
| flare.data2(min support= 0.4) | 1066 | 49 | 28 | 31 |
| flare.data2(min support= 0.6) | 1066 | 4 | 4 | 7 |
| nursery.data(min support= 0.2) | 12960 | 587 | 327 | 39 |
| nursery.data(min support= 0.4) | 12960 | 6 | 6 | 4 |
| nursery.data(min support= 0.5) | 12960 | 3 | 3 | 3 |

From the observations, we can easily conclude that by F(k-1)*F(k-1) fares better than F(k-1)*F(1) because the latter generates more unnecessary candidates than the former.

Logic: F(k-1)*F(1) is considerably more inefficient that F(k-1)*F(k-1) because it considers those sets also who have a subset which is infrequent. Ideally, for every candidate k-itemset that survives the pruning step, every item in the candidate must be contained in at least k - 1 of the frequent (k - 1)-itemsets (*Reference: Text book*)

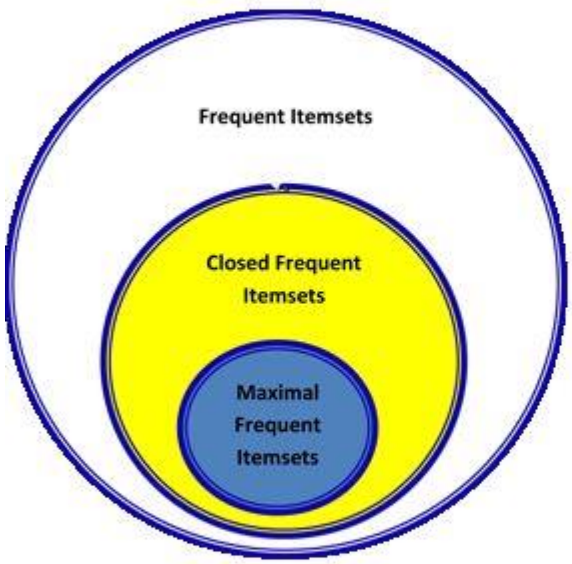
***Also, one observation is that, If we keep the minimum support count at a higher value like 70-80%, the we observe similar values for the number of frequent itemsets generated by both algorithms.***

C) Below is the Observation:

| Dataset | Number of rows | Number of Maximal Itemsets | Number of Closed Itemsets | Number of frequent Itemsets |
|---|---|---|---|---|
| car.data (min support= 0.2) | 1728 | 23 | 58 | 58 |
| car.data (min support= 0.5) | 1728 | 5 | 7 | 7 |
| car.data (min support= 0.6) | 1728 | 4 | 4 | 4 |
| flare.data2(min support= 0.2) | 1066 | 8 | 68 | 111 |

| | | | | |
|---|---|---|---|---|
| flare.data2(min support= 0.4) | 1066 | 3 | 21 | 31 |
| flare.data2(min support= 0.6) | 1066 | 1 | 5 | 7 |
| nursery.data(min support= 0.2) | 12960 | 27 | 39 | 39 |
| nursery.data(min support= 0.4) | 12960 | 4 | 4 | 4 |
| nursery.data(min support= 0.5) | 12960 | 3 | 3 | 3 |

In conclusion, closed and maximal frequent itemsets are subsets of frequent itemsets but maximal frequent itemsets are a more compact representation because it is a subset of closed frequent itemsets. The diagram shows the relationship between these three types of itemsets. Closed frequent itemsets are more widely used than maximal frequent itemset because when efficiency is more important that space, they provide us with the support of the subsets so no additional pass is needed to find this information.




**Reference** :
http://www.hypertextbookshop.com/dataminingbook/public_version/contents/chapters/chapter002/section004/blue/page002.html

D) Below is the required observation:

| Dataset | Number of rows | Number of Rules by confidence based pruning | Number of rules by brute force(without pruning) | Number of frequent Itemsets |
|---|---|---|---|---|

| | | | | |
|---|---|---|---|---|
| car.data (min support= 0.2 , min confidence =0.2) | 1728 | 83 | 2147500028 | 13 |
| car.data (min support= 0.4 min confidence =0.5) | 1728 | 12 | 62 | 7 |
| car.data (min support= 0.3 min confidence =0.3) | 1728 | 36 | 32768 | 4 |
| flare.data2(min support= 0.2 min confidence =0.2) | 1066 | 70 | 1168231628800 | 111 |
| flare.data2(min support= 0.4 min confidence =0.2) | 1066 | 23 | 5122 | 31 |
| flare.data2(min support= 0.6 min confidence =0.2) | 1066 | 5 | 6 | 7 |
| nursery.data(min support= 0.1 min confidence =0.1) | 12960 | 277 | 2658455991569831745807614120560697340 | 39 |
| nursery.data(min support= 0.2 min confidence =0.2) | 12960 | 26 | 8190 | 4 |
| nursery.data(min support= 0.25 min confidence =0.01) | 12960 | 6 | 6 | 3 |

It can be easily inferred from the above table that confidence based pruning sufficiently scores over brute force over the complexity and efficiency

E) Below is the require observation:

| Dataset | Number of rows | Top 10 Association Rules by confidence pruning with their confidence measures. | Conclusion |
|---|---|---|---|
| car.data (min support= 0.2 , min confidence =0.01) | 1728 | ['2', 'vhigh'] => ['unacc'] (0.925925925926)<br>['2', 'low'] => ['unacc'] (0.887037037037)<br>['2'] => ['unacc'] (0.877314814815)<br>['2', 'med'] => ['unacc'] (0.864197530864)<br>['2', 'high'] => ['unacc'] (0.855555555556)<br>['vhigh'] => ['unacc'] (0.809523809524)<br>['small'] => ['unacc'] (0.78125)<br>['med', 'vhigh'] => ['unacc'] (0.779069767442)<br>['2'] => ['med'] (0.75)<br>['4'] => ['med'] (0.75) | The combination of the top 10 rules observed is found in maximum in the dataset. |

| | | | |
|---|---|---|---|
| car.data (min support= 0.3 min confidence =0.25) | 1728 | ['2'] => ['unacc'] (0.877314814815)<br>['2', 'med'] => ['unacc'] (0.864197530864)<br>['vhigh'] => ['unacc'] (0.809523809524)<br>['2'] => ['med'] (0.75)<br>['4'] => ['med'] (0.75)<br>['2', 'unacc'] => ['med'] (0.738786279683)<br>['low'] => ['unacc'] (0.72962962963)<br>['unacc'] => ['med'] (0.717355371901)<br>['low', 'med'] => ['unacc'] (0.708333333333)<br>['high'] => ['med'] (0.688888888889) | The min support and confidence count is increased, hence the list is shortened and those survive who exceed a confidence value of 0.25 |
| car.data (min support= 0.4 min confidence =0.3) | 1728 | ['2'] => ['unacc'] (0.877314814815)<br>['low'] => ['unacc'] (0.72962962963)<br>['unacc'] => ['med'] (0.717355371901)<br>['high'] => ['med'] (0.688888888889)<br>['low'] => ['med'] (0.688888888889)<br>['med'] => ['unacc'] (0.66975308642)<br>['high'] => ['unacc'] (0.653703703704)<br>['unacc'] => ['low'] (0.651239669421)<br>['unacc'] => ['2'] (0.626446280992)<br>['unacc'] => ['high'] (0.58347107438) | The combination of the top 10 rules observed is found in maximum in the dataset. |
| flare.data2(min support= 0.2 min confidence =0.2) | 1066 | ['1', 'H', 'S', 'X'] => ['0'] (1.0)<br>['0', 'H', 'S', 'X'] => ['1'] (1.0)<br>['0', '1', 'S', 'X'] => ['H'] (1.0)<br>['0', '1', 'H', 'S'] => ['X'] (1.0)<br>['1'] => ['0'] (0.999061913696)<br>['1', '2'] => ['0'] (0.999027237354)<br>['1', '3'] => ['0'] (0.998113207547)<br>['1', '2', '3'] => ['0'] (0.99797979798)<br>['1', '2', 'C'] => ['0'] (0.99593495935)<br>['1', '2', 'D'] => ['0'] (0.995815899582)<br>['0', '1', 'R'] => ['2'] (0.98623853211) | The combination of the top 10 rules observed is found in maximum in the dataset. |
| flare.data2(min support= 0.4 min confidence =0.3) | 1066 | ['1', 'H', 'X'] => ['0'] (1.0)<br>['0', 'H', 'X'] => ['1'] (1.0)<br>['1'] => ['0'] (0.999061913696)<br>['1', '2'] => ['0'] (0.999027237354)<br>['1', '3'] => ['0'] (0.998113207547)<br>['1', '2', '3'] => ['0'] (0.99797979798)<br>['1'] => ['2'] (0.96435272045)<br>['0', '1'] => ['2'] (0.964319248826)<br>['0', '1', 'H'] => ['X'] (0.94301994302)<br>['1', '3'] => ['2'] (0.933962264151) | The combination of the top 10 rules observed is found in maximum in the dataset. |
| flare.data2(min support= 0.6 min confidence =0.2) | 1066 | ['1', '2', 'X'] => ['0'] (1.0)<br>['0', '2', 'X'] => ['1'] (1.0)<br>['1'] => ['0'] (0.999061913696)<br>['1', '2'] => ['0'] (0.999027237354)<br>['1', '3'] => ['0'] (0.998113207547)<br>['1', '2', '3'] => ['0'] (0.99797979798)<br>['1'] => ['2'] (0.96435272045)<br>['0', '1'] => ['2'] (0.964319248826)<br>['1', '3'] => ['2'] (0.933962264151)<br>['0', '1', '3'] => ['2'] (0.933837429112) | The combination of the top 10 rules observed is found in maximum in the dataset. |

| | | | |
|---|---|---|---|
| nursery.data(min support= 0.1 min confidence =0.1) | 12960 | ['great_pret', 'priority'] => ['convenient'] (0.686274509804)<br>['priority', 'recommended'] => ['convenient'] (0.679933665008)<br>['priority', 'problematic'] => ['convenient'] (0.677356656948)<br>['priority'] => ['convenient'] (0.671420083185)<br>['pretentious', 'priority'] => ['convenient'] (0.669193045029)<br>['priority', 'slightly_prob'] => ['convenient'] (0.668806161746)<br>['1'] => ['convenient'] (0.666666666667)<br>['2'] => ['convenient'] (0.666666666667)<br>['3'] => ['convenient'] (0.666666666667)<br>['complete'] => ['convenient'] (0.666666666667) | The combination of the top 10 rules observed is found in maximum in the dataset. |
| nursery.data(min support= 0.2 min confidence =0.2) | 12960 | ['priority'] => ['convenient'] (0.671420083185)<br>['great_pret'] => ['convenient'] (0.666666666667)<br>['nonprob'] => ['convenient'] (0.666666666667)<br>['not_recom'] => ['convenient'] (0.666666666667)<br>['pretentious'] => ['convenient'] (0.666666666667)<br>['problematic'] => ['convenient'] (0.666666666667)<br>['recommended'] => ['convenient'] (0.666666666667)<br>['slightly_prob'] => ['convenient'] (0.666666666667)<br>['usual'] => ['convenient'] (0.666666666667)<br>['critical'] => ['convenient'] (0.571428571429) | The combination of the top 10 rules observed is found in maximum in the dataset. |
| nursery.data(min support= 0.1 min confidence =0.01) | 12960 | ['great_pret', 'priority'] => ['convenient'] (0.686274509804)<br>['priority', 'recommended'] => ['convenient'] (0.679933665008)<br>['priority', 'problematic'] => ['convenient'] (0.677356656948)<br>['priority'] => ['convenient'] (0.671420083185)<br>['pretentious', 'priority'] => ['convenient'] (0.669193045029)<br>['priority', 'slightly_prob'] => ['convenient'] (0.668806161746)<br>['1'] => ['convenient'] (0.666666666667)<br>['2'] => ['convenient'] (0.666666666667)<br>['3'] => ['convenient'] (0.666666666667)<br>['complete'] => ['convenient'] (0.666666666667) | The combination of the top 10 rules observed is found in maximum in the dataset. The combination of the top 10 rules observed is found in maximum in the dataset. |

Important Observation: 1) very high buying rate and 2 doors are unacceptable cars

2) great pret and health priority is convenient which is very obvious.

F) Below is the observation:

| Dataset | Number of rows | Top 10 Association Rules by confidence pruning with their confidence measures. | Conclusion |
|---|---|---|---|
| car.data (min support= 0.2 , min Lift =1) | 1728 | ['2', 'vhigh'] => ['unacc'] (1.32231404959)<br>['high', 'unacc'] => ['2'] (1.30878186969)<br>['med', 'unacc'] => ['2'] (1.29032258065)<br>['2', 'low'] => ['unacc'] (1.2667768595)<br>['unacc'] => ['2'] (1.25289256198)<br>['2'] => ['unacc'] (1.25289256198)<br>['2', 'med'] => ['unacc'] (1.23415977961)<br>['2', 'high'] => ['unacc'] (1.22181818182)<br>['low', 'unacc'] => ['2'] (1.21573604061)<br>['unacc'] => ['vhigh'] (1.15608028335) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |
| car.data (min support= 0.3 min Lift =0.85) | 1728 | ['med', 'unacc'] => ['2'] (1.29032258065)<br>['unacc'] => ['2'] (1.25289256198)<br>['2'] => ['unacc'] (1.25289256198)<br>['2', 'med'] => ['unacc'] (1.23415977961)<br>['unacc'] => ['vhigh'] (1.15608028335)<br>['vhigh'] => ['unacc'] (1.15608028335)<br>['unacc'] => ['low'] (1.04198347107)<br>['low'] => ['unacc'] (1.04198347107)<br>['low', 'med'] => ['unacc'] (1.01157024793)<br>['high'] => ['2'] (1.0) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |
| car.data (min support= 0.4 min Lift =0.66) | 1728 | ['unacc'] => ['2'] (1.25289256198)<br>['2'] => ['unacc'] (1.25289256198)<br>['unacc'] => ['low'] (1.04198347107)<br>['low'] => ['unacc'] (1.04198347107)<br>['med'] => ['unacc'] (0.956473829201)<br>['unacc'] => ['med'] (0.956473829201)<br>['unacc'] => ['high'] (0.933553719008)<br>['high'] => ['unacc'] (0.933553719008)<br>['med'] => ['high'] (0.918518518519)<br>['high'] => ['med'] (0.918518518519) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |
| flare.data2(min support= 0.2 min Lift =1) | 1066 | ['0', '1', 'S', 'X'] => ['H'] (3.03703703704)<br>['0', '1', 'H', 'S'] => ['X'] (2.23949579832)<br>['0', '1', 'X'] => ['H'] (2.11188920012)<br>['0', '1', 'H'] => ['X'] (2.11188920012)<br>['0', '1', '2', 'H'] => ['X'] (2.09639702526)<br>['0', '1', '2', 'X'] => ['H'] (2.03162523254)<br>['0', '1', 'H', 'X'] => ['S'] (1.77363282105)<br>['0', '1', 'H'] => ['S'] (1.67257112185)<br>['0', '1', 'S'] => ['H'] (1.67257112185)<br>['0', '1', 'X'] => ['S'] (1.23334551212)<br>['0', '1', 'S'] => ['X'] (1.23334551212)) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |

| | | | |
|---|---|---|---|
| flare.data2(min support= 0.4 min Lift =0.65) | 1066 | ['0', '1', '2'] => ['O'] (1.03797468354)<br>['0', '1', 'O'] => ['2'] (1.03696498054)<br>['1', '2', 'X'] => ['0'] (1.00093896714)<br>['0', '2', 'X'] => ['1'] (1.0)<br>['1', '2'] => ['0'] (0.999965291098)<br>['0', '1'] => ['2'] (0.999965291098)<br>['1', '3'] => ['0'] (0.999050403047)<br>['0', '1'] => ['3'] (0.999050403047)<br>['1', '2', '3'] => ['0'] (0.998916868213)<br>['1', '3'] => ['2'] (0.968486161075)<br>['1', '2'] => ['3'] (0.968486161075) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |
| flare.data2(min support= 0.1 min Lift =0.85) | 1066 | ['0', '1', '2', 'O', 'X'] => ['B'] (7.25170068027)<br>['0', '1', '2', 'S', 'X'] => ['H'] (3.03703703704)<br>['0', '1', '2', 'A'] => ['I'] (2.48032828496)<br>['0', '1', 'A'] => ['I'] (2.43439627969)<br>['0', '1', '2', 'I'] => ['A'] (2.43439627969)<br>['0', '1', '2', 'X'] => ['B'] (2.40067716584)<br>['0', '1', '2', 'D'] => ['I'] (2.2897087086)<br>['0', '1', '2', 'I'] => ['D'] (2.28012833743)<br>['0', '1', '2', 'H', 'S'] => ['X'] (2.23949579832)<br>['0', '1', '2', 'B'] => ['X'] (2.20902646773)<br>['0', '1', '2', 'B', 'O'] => ['X'] (2.2055640438) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent.. |
| nursery.data(min support= 0.1 min Lift =0.25) | 12960 | ['very_crit'] => ['spec_prior'] (1.87685459941)<br>['spec_prior'] => ['very_crit'] (1.87685459941)<br>['spec_prior'] => ['great_pret'] (1.5)<br>['great_pret'] => ['spec_prior'] (1.5)<br>['critical', 'priority'] => ['spec_prior'] (1.48517828972)<br>['spec_prior'] => ['critical'] (1.25900805426)<br>['critical'] => ['spec_prior'] (1.25900805426)<br>['inconv', 'priority'] => ['spec_prior'] (1.25880182853)<br>['convenient', 'critical'] => ['spec_prior'] (1.25741839763)<br>['priority', 'spec_prior'] => ['critical'] (1.22784150156)<br>['spec_prior'] => ['problematic'] (1.21958456973)<br>['problematic'] => ['spec_prior'] (1.21958456973) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |
| nursery.data(min support= 0.2 min confidence =0.1) | 12960 | ['priority'] => ['convenient'] (1.00713012478)<br>['convenient'] => ['priority'] (1.00713012478)<br>['great_pret'] => ['convenient'] (1.0)<br>['convenient'] => ['great_pret'] (1.0)<br>['nonprob'] => ['convenient'] (1.0)<br>['convenient'] => ['nonprob'] (1.0)<br>['not_recom'] => ['convenient'] (1.0)<br>['convenient'] => ['not_recom'] (1.0)<br>['pretentious'] => ['convenient'] (1.0)<br>['convenient'] => ['pretentious'] (1.0) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |

| | | | |
|---|---|---|---|
| nursery.data(min support= 0.3 min confidence =0.01) | 12960 | ['priority'] => ['convenient'] (1.00713012478)<br>['convenient'] => ['priority'] (1.00713012478)<br>['priority'] => ['inconv'] (0.989304812834)<br>['inconv'] => ['priority'] (0.989304812834)<br>['convenient'] => ['critical'] (0.857142857143)<br>['critical'] => ['convenient'] (0.857142857143) | Now those rules are prominent whose right hand side or the rule consequent is higher because confidence ignores the rule consequent. |

Important Observation: 1) very high buying rate and 2 doors are unacceptable cars

2) great pret and health priority is convenient which is very obvious.

## <u>Observation and Consequence: (Lift vs Confidence)</u>

We can see that the rules generated by the confidence and lift measures are different because the confidence measure ignores the support of the itemset appearing in the rule consequent while the Lift is obtained by dividing the confidence of the rule by the support of the right hand side of the rule or rule consequent.

In layman terms, we can think that ignoring the number of times an item appears as a rule consequent can play a part in efficiency since, if an item is appearing more than the others as a rule consequent then this phenomena is ought to be observed.

Lift is not down-ward closed and does not suffer from the rare item problem. Also lift is susceptible to noise in small databases. Rare itemsets with low counts (low probability) which per chance occur a few times (or only once) together can produce enormous lift values.

Range: [0,∞][0,∞] (1 means independence)

### The other outputs are generated in different output files kept in observations folder.

***One full run of program over car.data gives the following collective output with min support as 0.5 , min confidence of 0.2 and min Lift of 1:***

**Starting Association Rule Mining.........................................................................**

**Distinct Items in the dataset: ---------> ['vhigh', '2', 'small', 'low', 'unacc', 'med', 'high', 'big', '4', 'more', '3', '5more', 'acc', 'vgood', 'good']**

********************  There are a total of 15 item(s) and  1728 transaction(s)  ********************

********************  Working on generation of frequent itemsets  ********************

**print frequent itemsets {1: [[['2'], 0.5], [['4'], 0.5], [['high'], 0.625], [['low'], 0.625], [['med'], 0.75], [['unacc'], 0.7002314814814815]], 2: [[['med', 'unacc'], 0.5023148148148148]]}**

**Total number of frequent itemsets generated: 7**

**Number of candidate sets generated by F(k-1)*F(1) : 19**

**Number of candidate sets generated by F(k-1)*F(k-1) : 15**

******************** Working on generating rules by confidence pruning ********************

******************** confidence pruned rules generatation completed ********************

******************** Writing output for confidence based rules started ********************

**Following confidence based pruned rules are generated:**

**['unacc'] => ['med'] (0.717355371901)**

**['med'] => ['unacc'] (0.66975308642)**

**2 confidence based pruned association rules generated**

******************** Working on generating rules by Lift pruning ********************

******************** Lift pruned rules generatation completed ********************

******************** Writing output for lift based rules started ********************

**Following Lift based pruned rules are generated:**

**0 Lift based pruned association rules generated**

******************** Generating Association rules completed ********************

******************** Generating Closed Itemsets ********************

******************** writing output ********************

**Closed sets are:------------->**

**['med'], 0.75**

**['unacc'], 0.700231481481**

**['high'], 0.625**

**['low'], 0.625**

**['med', 'unacc'], 0.502314814815**

**['2'], 0.5**

['4'], 0.5

7 Number of Closed sets generated

****************** Generating maximal Itemsets ******************

****************** writing output ******************

Maximal sets are:------------->

['high'], 0.625

['low'], 0.625

['med', 'unacc'], 0.502314814815

['2'], 0.5

['4'], 0.5

5 Number of maximal sets generated

**Datasets used:**

http://archive.ics.uci.edu/ml/datasets/Molecular+Biology+%28Splice-junction+Gene+Sequences%29

http://archive.ics.uci.edu/ml/datasets/Car+Evaluation

http://archive.ics.uci.edu/ml/datasets/Solar+Flare

http://archive.ics.uci.edu/ml/datasets/Nursery

References: 1) http://michael.hahsler.net/research/association_rules/measures.html

2)http://www.hypertextbookshop.com/dataminingbook/public_version/contents/chapters/chapter002/section004/blue/page002.html

3) http://www.wikipedia.org.

4) https://github.com/kissghosts/

5) http://www.ncbi.nlm.nih.gov/pubmed/8796672