CSCE 474/874: Introduction to Data Mining
Spring 2025

Assignment No. 2 February 1, 2025

Documentation of Contributions

Please use this form to document the contributions of each team member to the group effort. You may either sign electronically or sign and scan it. In either case, you must submit this document along with all other elements of the assignment.

Furthermore, you confirm that the work submitted is entirely your own and that you understand the consequences of plagiarism as specified in the Course Syllabus and UNL CSE Academic Integrity Policy.

Name	Contribution %	Contributions	Signature & Date
Dada Zhang	**50%**	- **Working on data analysis using Python and WeKa** - **Writing report** - **Build repo** - **Discussion with team member**	
Nick Montemarano	**50%**	- **Help develop python code** - **Implement comparison testing** - **Writing report** - **Discussion**	

Task 1. We started this assignment with writing Apriori algorithm in Python using Google Colab. The python file is included in the zip file.

1. **Dataset and preprocessing.**
   Load dataset (**vote.arff**) to Goole Colab and do conversion (e.g., dataframe and transaction) for further Apriori algorithm. We deleted the last column (Class) of vote.arff dataset due to its records.

2. **Determine the frequent sets and generate association rules.**
   There are multiple ways to determine the frequent sets in Python and we finally customized a function of Apriori algorithm with inputs such as transactions, minimum support, and minimum confidence.

   Inputs: the minimums for support and confidence are 0.3 and 0.6.

Dataset – test.arff. We load the test dataset to the program and find the frequent item sets and association rules. The results were saved in text format. See "**frequent_itemsets.txt**" and "**association_rules.txt**" files.

Task 2. Plot runtime of Apriori algorithm and number of rules as a function of "minimum support." The plots using "vote.arff" dataset are shown in Appendix.

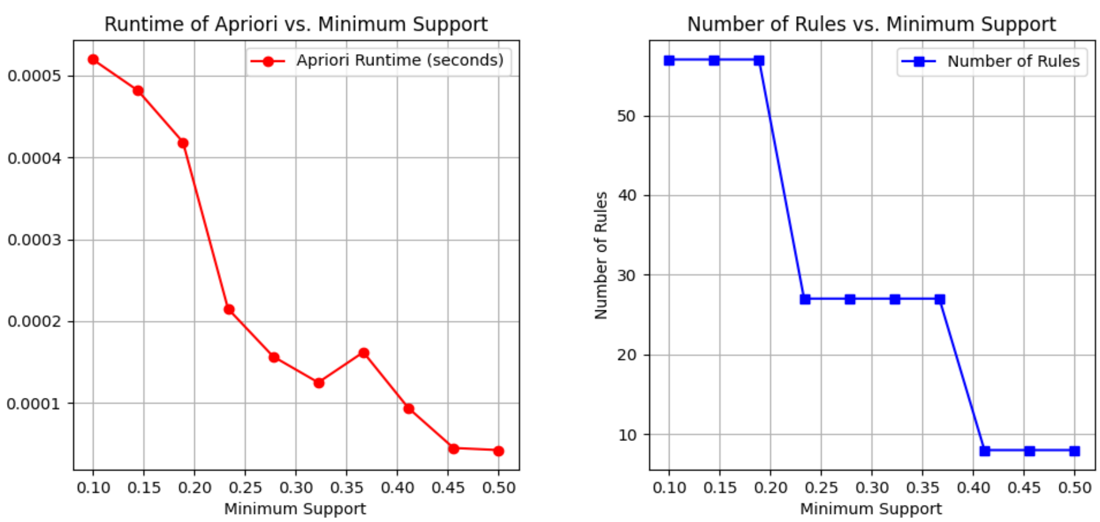


Figure 1. Plot of runtime and number of rules vs. minimum support using test.arff dataset.

Task 3. Derive association rules in Weka and compare them to the results derived from the program.

Parameter setting. First, import data into Weka Explorer: Click Preprocess – Open file – test.arff (or select vote.arff for practice). Then, click Associate – Apriori – set up parameters to generate association rules (See Figure 2).

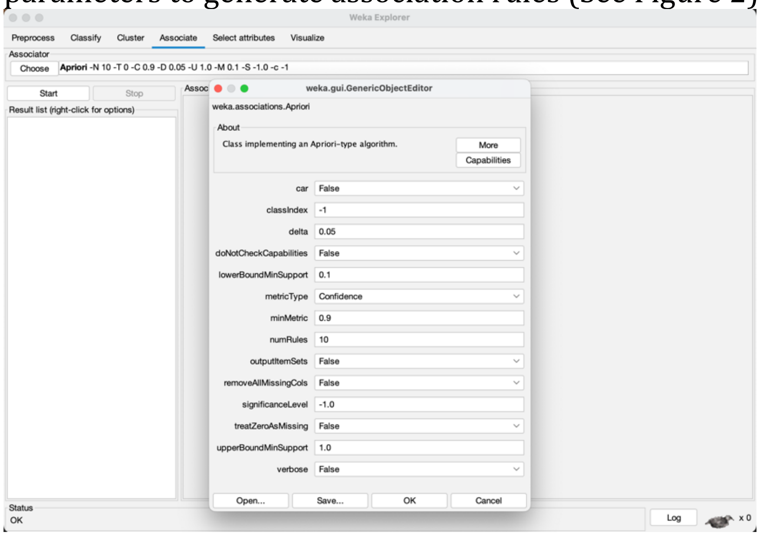


Figure 2 Setting parameters for Aprior Association in Weka.

Since we used the minimums for support and confidence are 0.3 and 0.6 in Python program, then we used the same values in Weka, see Figure 3 for details.

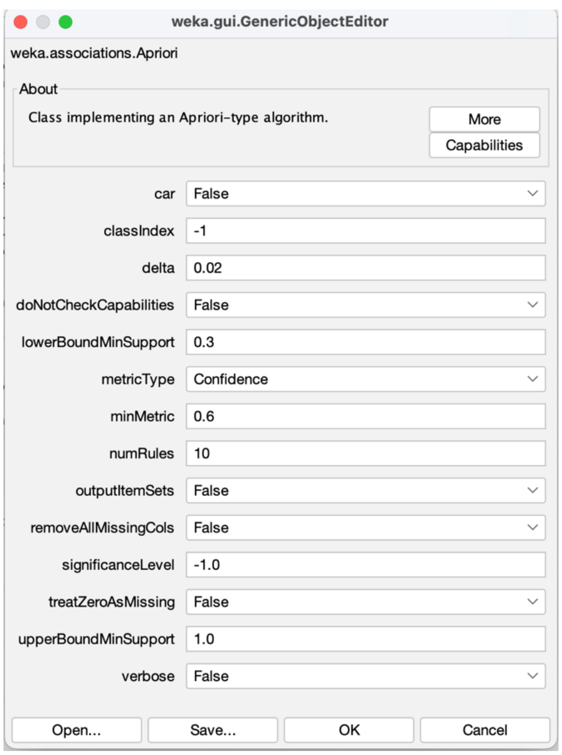


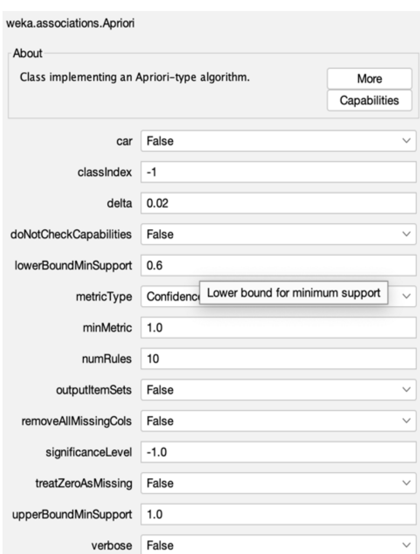
Figure 3. Parameter setting in Weka.

Output and comparison. We keep using the "test.arff" dataset in Weka software, and the results are shown in Figure 4. Comparing the results from Python and Weka, the results are same.

```
Associator output
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.6 -D 0.05 -U 1.0 -M 0.3 -S -1.0 -c -1
Relation:     test-1
Instances:    5
Attributes:   6
              Bread
              Milk
              Diaper
              Beer
              Eggs
              Coke
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.7 (3 instances)
Minimum metric <confidence>: 0.6
Number of cycles performed: 6

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 8

Size of set of large itemsets L(3): 2

Best rules found:

 1. Eggs=n 4 ==> Milk=y 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 2. Milk=y 4 ==> Eggs=n 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 3. Coke=n 3 ==> Bread=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 4. Beer=y 3 ==> Diaper=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 5. Bread=y Eggs=n 3 ==> Milk=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 6. Bread=y Milk=y 3 ==> Eggs=n 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 7. Diaper=y Eggs=n 3 ==> Milk=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 8. Milk=y Diaper=y 3 ==> Eggs=n 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 9. Milk=y 4 ==> Bread=y 3    <conf:(0.75)> lift:(0.94) lev:(-0.04) [0] conv:(0.4)
10. Bread=y 4 ==> Milk=y 3    <conf:(0.75)> lift:(0.94) lev:(-0.04) [0] conv:(0.4)
```

Figure 4. Association output in Weka using test.arff dataset. Inputs are shown in Figure 3.

If we want to match the results of "test_output.rtf" file, the inputs and results are shown below.




```
Apriori
=======

Minimum support: 0.6 (3 instances)
Minimum metric <confidence>: 1
Number of cycles performed: 20

Generated sets of large itemsets:

Size of set of large itemsets L(1): 6

Size of set of large itemsets L(2): 8

Size of set of large itemsets L(3): 2

Best rules found:

 1. Eggs=n 4 ==> Milk=y 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 2. Milk=y 4 ==> Eggs=n 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 3. Coke=n 3 ==> Bread=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 4. Beer=y 3 ==> Diaper=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 5. Bread=y Eggs=n 3 ==> Milk=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 6. Bread=y Milk=y 3 ==> Eggs=n 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 7. Diaper=y Eggs=n 3 ==> Milk=y 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
 8. Milk=y Diaper=y 3 ==> Eggs=n 3    <conf:(1)> lift:(1.25) lev:(0.12) [0] conv:(0.6)
```

Appendix

Output using "vote.arff" dataset.

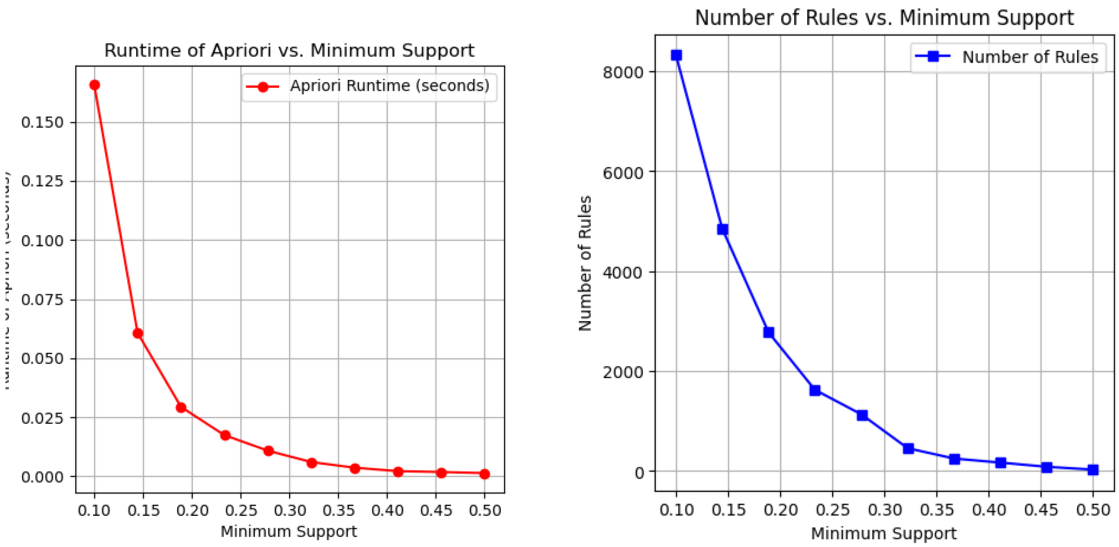


Figure 5. Plot of runtime and number of rules vs. minimum support using vote.arff dataset

The top 10 results from Weka and Python were summarized in Figures 6-7.

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.6 -D 0.02 -U 1.0 -M 0.3 -S -1.0 -c -1
Relation:     vote-weka.filters.unsupervised.attribute.ReplaceMissingValues
Instances:    435
Attributes:   17
              handicapped-infants
              water-project-cost-sharing
              adoption-of-the-budget-resolution
              physician-fee-freeze
              el-salvador-aid
              religious-groups-in-schools
              anti-satellite-test-ban
              aid-to-nicaraguan-contras
              mx-missile
              immigration
              synfuels-corporation-cutback
              education-spending
              superfund-right-to-sue
              crime
              duty-free-exports
              export-administration-act-south-africa
              Class
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.56 (244 instances)
Minimum metric <confidence>: 0.6
Number of cycles performed: 22

Generated sets of large itemsets:

Size of set of large itemsets L(1): 12

Size of set of large itemsets L(2): 7

Size of set of large itemsets L(3): 1

Best rules found:

 1. aid-to-nicaraguan-contras=y 257 ==> export-administration-act-south-africa=y 254    <conf:(0.99)> lift:(1.15) lev:(0.08) [33] conv:(9.16)
 2. anti-satellite-test-ban=y 253 ==> export-administration-act-south-africa=y 250    <conf:(0.99)> lift:(1.15) lev:(0.08) [33] conv:(9.01)
 3. adoption-of-the-budget-resolution=y 264 ==> export-administration-act-south-africa=y 259    <conf:(0.98)> lift:(1.14) lev:(0.08) [32] conv:(6.27)
 4. physician-fee-freeze=n 258 ==> Class=democrat 253    <conf:(0.98)> lift:(1.6) lev:(0.22) [94] conv:(16.61)
 5. physician-fee-freeze=n export-administration-act-south-africa=y 251 ==> Class=democrat 246    <conf:(0.98)> lift:(1.6) lev:(0.21) [91] conv:(16.16)
 6. physician-fee-freeze=n 258 ==> export-administration-act-south-africa=y 251    <conf:(0.97)> lift:(1.13) lev:(0.07) [29] conv:(4.6)
 7. physician-fee-freeze=n Class=democrat 253 ==> export-administration-act-south-africa=y 246    <conf:(0.97)> lift:(1.13) lev:(0.07) [29] conv:(4.51)
 8. export-administration-act-south-africa=y Class=democrat 255 ==> physician-fee-freeze=n 246    <conf:(0.96)> lift:(1.63) lev:(0.22) [94] conv:(10.38)
 9. Class=democrat 267 ==> export-administration-act-south-africa=y 255    <conf:(0.96)> lift:(1.11) lev:(0.06) [26] conv:(2.93)
10. physician-fee-freeze=n 258 ==> export-administration-act-south-africa=y Class=democrat 246    <conf:(0.95)> lift:(1.63) lev:(0.22) [94] conv:(8.21)
```

Figure 6. Results of association in Weka.

index	Antecedent	Consequent	Support	Confidence
0	{'crime'}	{'superfund-right-to-sue'}	0.459770115	0.75471698
1	{'superfund-right-to-sue'}	{'crime'}	0.459770115	0.85470086
2	{'water-project-cost-sharing'}	{'superfund-right-to-sue'}	0.356321839	0.63786008
3	{'superfund-right-to-sue'}	{'water-project-cost-sharing'}	0.356321839	0.66239316
4	{'education-spending'}	{'religious-groups-in-schools'}	0.365517241	0.92982456
5	{'religious-groups-in-schools'}	{'export-administration-act-south-africa'}	0.517241379	0.795053
6	{'export-administration-act-south-africa'}	{'religious-groups-in-schools'}	0.517241379	0.60321716
7	{'el-salvador-aid'}	{'physician-fee-freeze'}	0.388505747	0.74449339
8	{'physician-fee-freeze'}	{'el-salvador-aid'}	0.388505747	0.95480226
9	{'immigration'}	{'crime'}	0.337931034	0.65919283
10	{'religious-groups-in-schools'}	{'superfund-right-to-sue'}	0.47816092	0.73498233
11	{'superfund-right-to-sue'}	{'religious-groups-in-schools'}	0.47816092	0.88888889
12	{'crime'}	{'education-spending'}	0.374712644	0.61509434
13	{'education-spending'}	{'crime'}	0.374712644	0.95321637
14	{'crime'}	{'physician-fee-freeze'}	0.4	0.65660377
15	{'physician-fee-freeze'}	{'crime'}	0.4	0.98305085
16	{'immigration'}	{'religious-groups-in-schools'}	0.354022989	0.69058296
17	{'el-salvador-aid'}	{'export-administration-act-south-africa'}	0.383908046	0.73568282
18	{'superfund-right-to-sue'}	{'physician-fee-freeze'}	0.356321839	0.66239316
19	{'physician-fee-freeze'}	{'superfund-right-to-sue'}	0.356321839	0.87570622
20	{'education-spending'}	{'superfund-right-to-sue'}	0.337931034	0.85964912
21	{'superfund-right-to-sue'}	{'education-spending'}	0.337931034	0.62820513
22	{'religious-groups-in-schools'}	{'el-salvador-aid'}	0.487356322	0.74911661
23	{'el-salvador-aid'}	{'religious-groups-in-schools'}	0.487356322	0.93392071
24	{'el-salvador-aid'}	{'water-project-cost-sharing'}	0.324137931	0.62114537
25	{'immigration'}	{'export-administration-act-south-africa'}	0.450574713	0.87892377
26	{'education-spending'}	{'physician-fee-freeze'}	0.324137931	0.8245614
27	{'physician-fee-freeze'}	{'education-spending'}	0.324137931	0.79661017
28	{'crime'}	{'export-administration-act-south-africa'}	0.471264368	0.77358491
29	{'el-salvador-aid'}	{'superfund-right-to-sue'}	0.434482759	0.83259912
30	{'superfund-right-to-sue'}	{'el-salvador-aid'}	0.434482759	0.80769231

Figure 7. An example of Python output.

- **The output for the top 10 rules based on confidence for python were :**
    - **{'duty-free-exports', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.352, Confidence: 1.000)**
    - **{'anti-satellite-test-ban', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.487, Confidence: 1.000)**
    - **{'aid-to-nicaraguan-contras', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.520, Confidence: 1.000)**
    - **{'aid-to-nicaraguan-contras', 'anti-satellite-test-ban'} -> {'export-administration-act-south-africa'} (Support: 0.515, Confidence: 1.000)**
    - **{'immigration', 'aid-to-nicaraguan-contras'} -> {'export-administration-act-south-africa'} (Support: 0.310, Confidence: 1.000)**
    - **{'duty-free-exports', 'anti-satellite-test-ban'} -> {'export-administration-act-south-africa'} (Support: 0.340, Confidence: 1.000)**
    - **{'immigration', 'anti-satellite-test-ban'} -> {'export-administration-act-south-africa'} (Support: 0.308, Confidence: 1.000)**
    - **{'aid-to-nicaraguan-contras', 'anti-satellite-test-ban', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.460, Confidence: 1.000)**
    - **{'handicapped-infants', 'anti-satellite-test-ban', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.303, Confidence: 1.000)**

- {'mx-missile', 'anti-satellite-test-ban', 'adoption-of-the-budget-resolution'} -> {'export-administration-act-south-africa'} (Support: 0.418, Confidence: 1.000)

- **These differ from the Weka results however many of the same rules are generated for both top ten, just with slightly different confidence values. It seems Weka's apriori algorithm found strong relations with physician-fee-freeze which the python algorithm did not. The python algorithm also seems to have overall stronger confidence values.**

Reference:

A priori: Frequent itemsets via the Apriori algorithm. https://rasbt.github.io/mlxtend/user_guide/frequent_patterns/apriori/#apriori-frequent-itemsets-via-the-apriori-algorithm

Apriori Algorithm. https://www.geeksforgeeks.org/apriori-algorithm/