Day4-Data Mining Experiments

1. Customer Segmentation using Clustering

Aim:

To segment supermarket customers using clustering based on income and spending score.

Algorithm:

1. Collect customer data

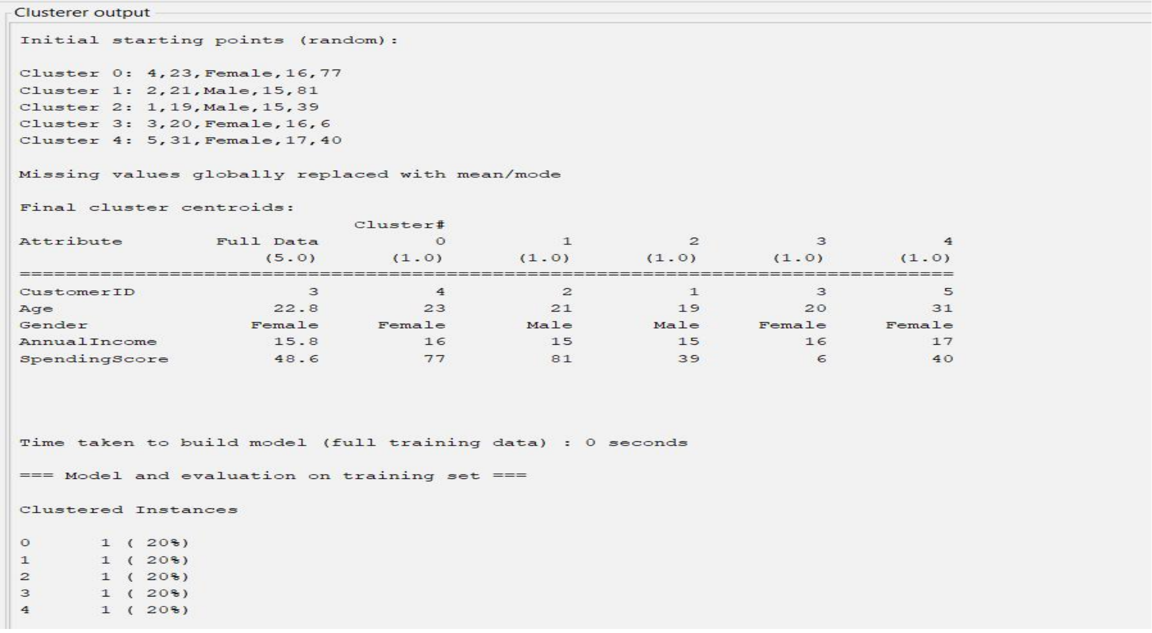
2. Select income and spending score

3. Apply K-Means

4. Choose k clusters

5. Analyze clusters

Output:



```
Initial starting points (random):

Cluster 0: 4,23,Female,16,77
Cluster 1: 2,21,Male,15,81
Cluster 2: 1,19,Male,15,39
Cluster 3: 3,20,Female,16,6
Cluster 4: 5,31,Female,17,40

Missing values globally replaced with mean/mode

Final cluster centroids:
                             Cluster#
Attribute        Full Data          0          1          2          3          4
                     (5.0)      (1.0)      (1.0)      (1.0)      (1.0)      (1.0)
=================================================================================
CustomerID               3          4          2          1          3          5
Age                   22.8         23         21         19         20         31
Gender              Female     Female       Male       Male     Female     Female
AnnualIncome          15.8         16         15         15         16         17
SpendingScore         48.6         77         81         39          6         40




Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       1 ( 20%)
1       1 ( 20%)
2       1 ( 20%)
3       1 ( 20%)
4       1 ( 20%)
```

2. Employee Data Clustering using K-Means

Aim:

To perform clustering on employee data using K-Means.

Algorithm:

1. Load CSV file

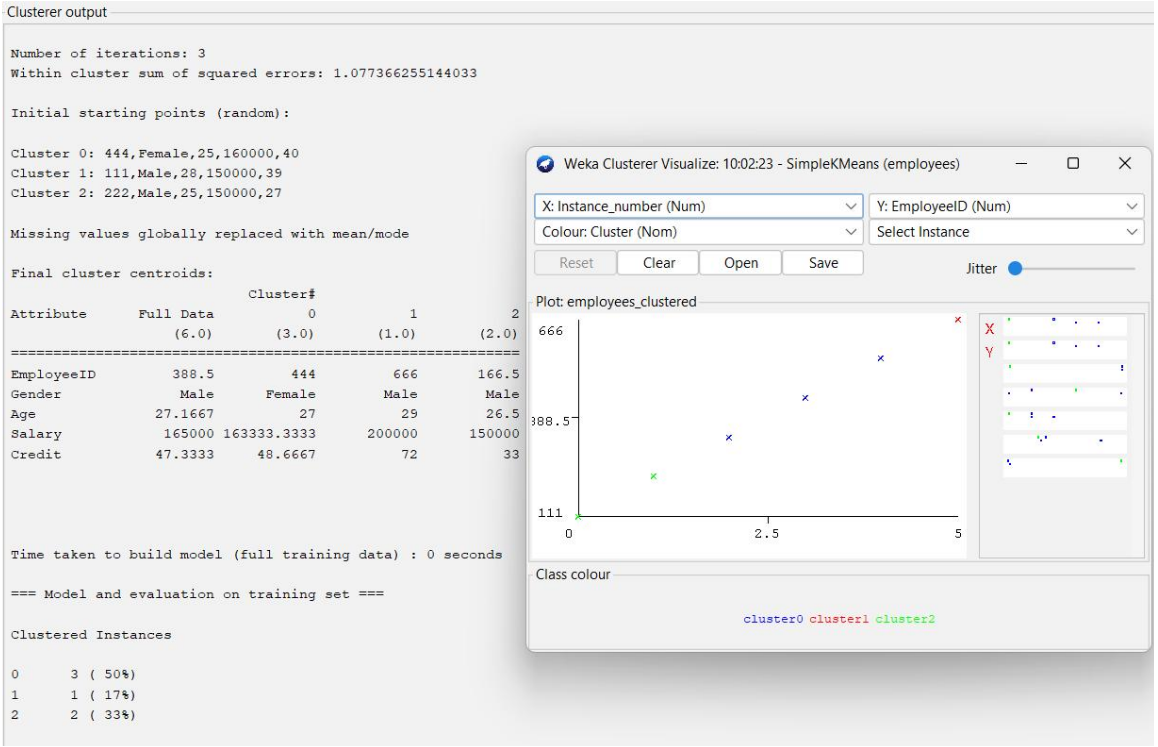
2. Convert to ARFF

3. Apply K-Means

4. Change cluster size

5. Visualize output

Output:



Clusterer output

```
Number of iterations: 3
Within cluster sum of squared errors: 1.077366255144033

Initial starting points (random):

Cluster 0: 444,Female,25,160000,40
Cluster 1: 111,Male,28,150000,39
Cluster 2: 222,Male,25,150000,27

Missing values globally replaced with mean/mode

Final cluster centroids:
                             Cluster#
Attribute      Full Data            0            1            2
                   (6.0)        (3.0)        (1.0)        (2.0)
===============================================================
EmployeeID         388.5          444          666        166.5
Gender              Male       Female         Male         Male
Age              27.1667           27           29         26.5
Salary            165000  163333.3333       200000       150000
Credit           47.3333      48.6667           72           33




Time taken to build model (full training data) : 0 seconds

=== Model and evaluation on training set ===

Clustered Instances

0      3 ( 50%)
1      1 ( 17%)
2      2 ( 33%)
```

Weka Clusterer Visualize: 10:02:23 - SimpleKMeans (employees)

X: Instance_number (Num)

Y: EmployeeID (Num)

Colour: Cluster (Nom)

Select Instance

3. Naive Bayes Classification

Aim:

To classify categorical data using Naive Bayes and compare with SVM.

Algorithm:

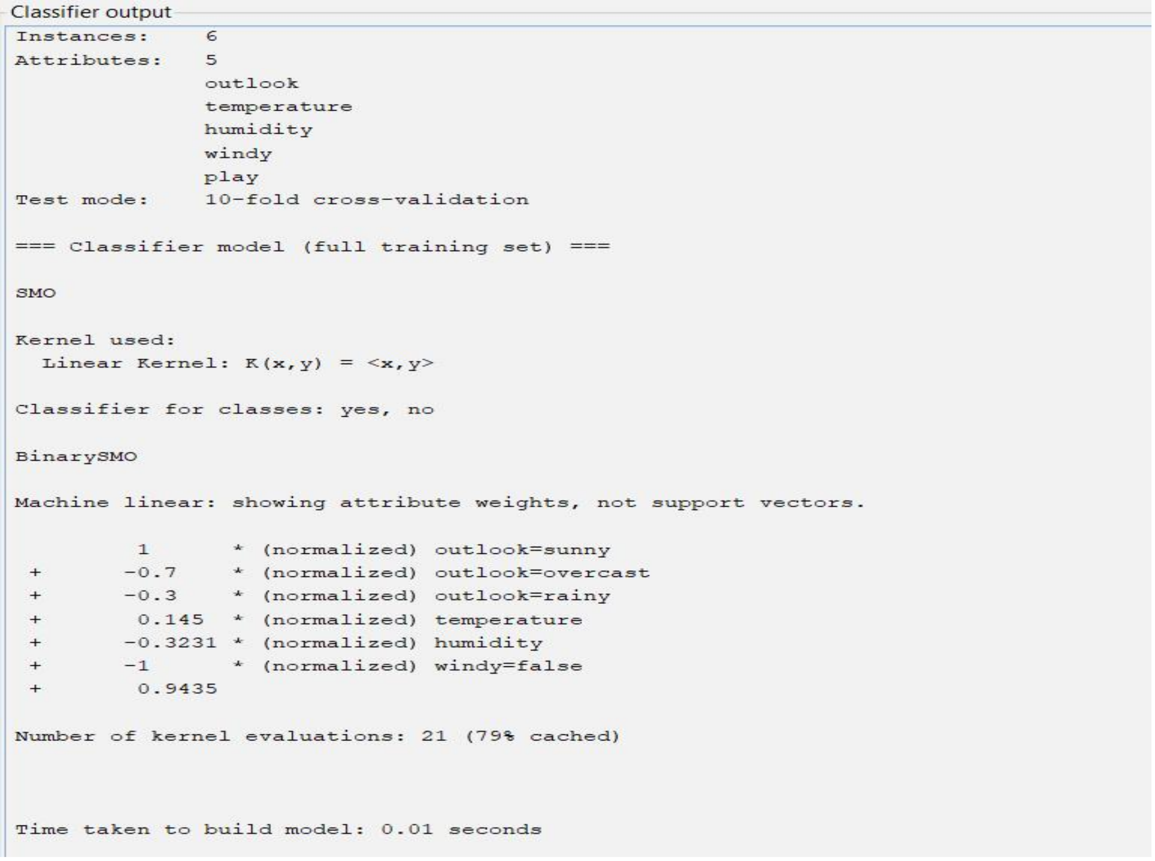
1. Load dataset

2. Apply Naive Bayes

3. Apply SVM

4. Compare accuracy

Output:



```
Instances:     6
Attributes:    5
               outlook
               temperature
               humidity
               windy
               play
Test mode:     10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: yes, no

BinarySMO

Machine linear: showing attribute weights, not support vectors.

         1       * (normalized) outlook=sunny
 +      -0.7     * (normalized) outlook=overcast
 +      -0.3     * (normalized) outlook=rainy
 +       0.145   * (normalized) temperature
 +      -0.3231 * (normalized) humidity
 +      -1       * (normalized) windy=false
 +       0.9435

Number of kernel evaluations: 21 (79% cached)



Time taken to build model: 0.01 seconds
```

4. Vegetarian Count

Aim:

To find number of vegetarian and non-vegetarian persons.

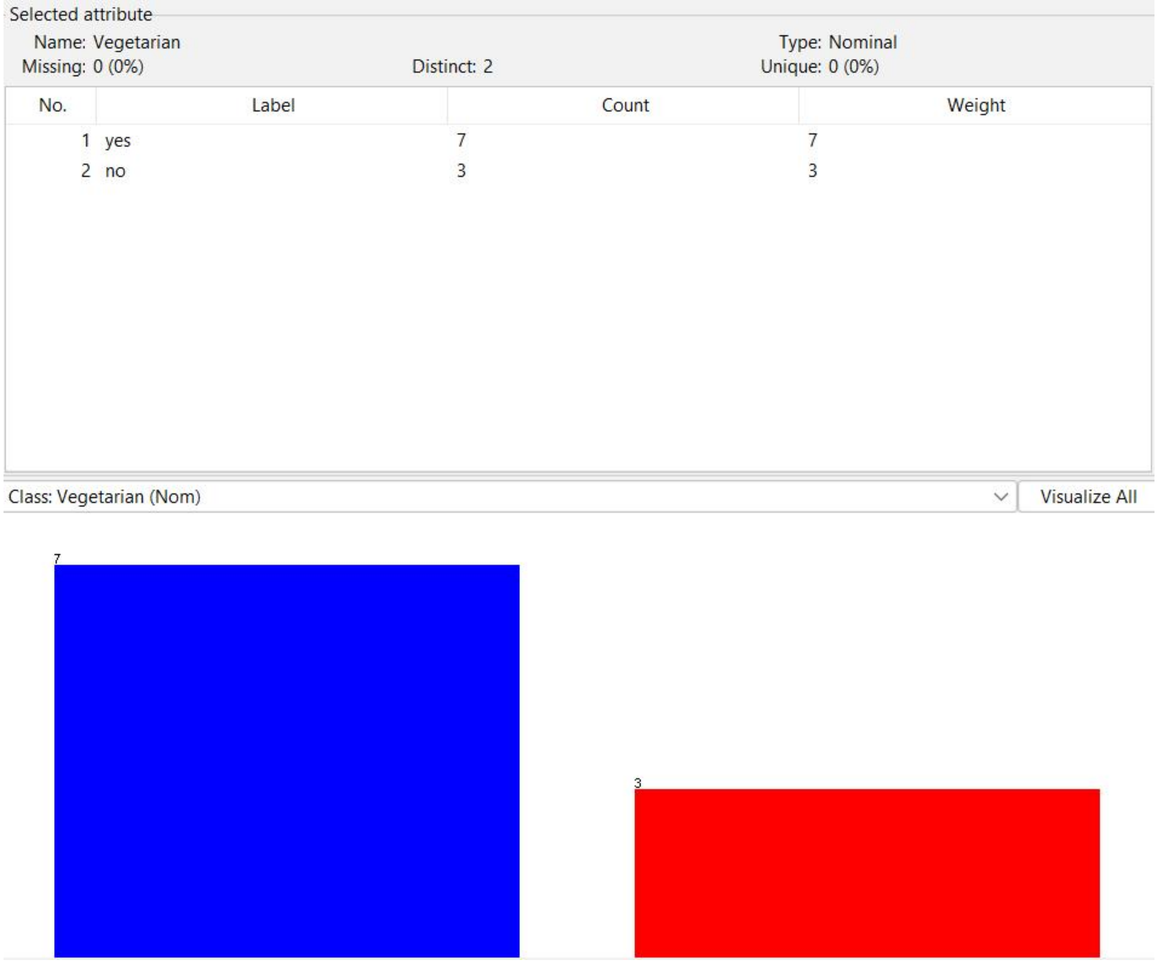
Algorithm:

1. Load data

2. Count yes and no

3. Compare totals

Output:



Selected attribute

Name: Vegetarian Type: Nominal

Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	yes	7	7
2	no	3	3

Class: Vegetarian (Nom)

5. Scatter Plot

Aim:

To plot scatter graph for mobile sales data.

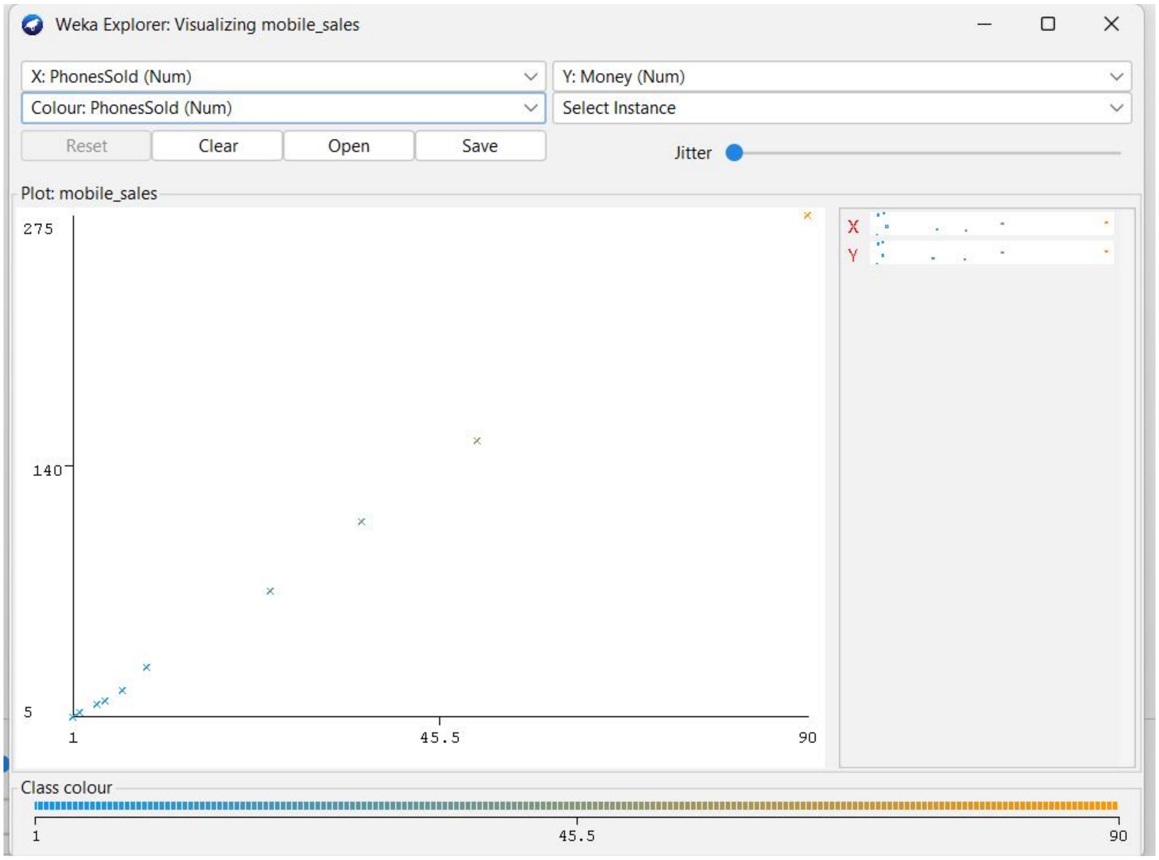
Algorithm:

1. Load x,y data

2. Choose scatter plot

3. Plot graph

Output:



Weka Explorer: Visualizing mobile_sales

X: PhonesSold (Num)

Y: Money (Num)

Colour: PhonesSold (Num)

Select Instance

6. FP-Growth Algorithm

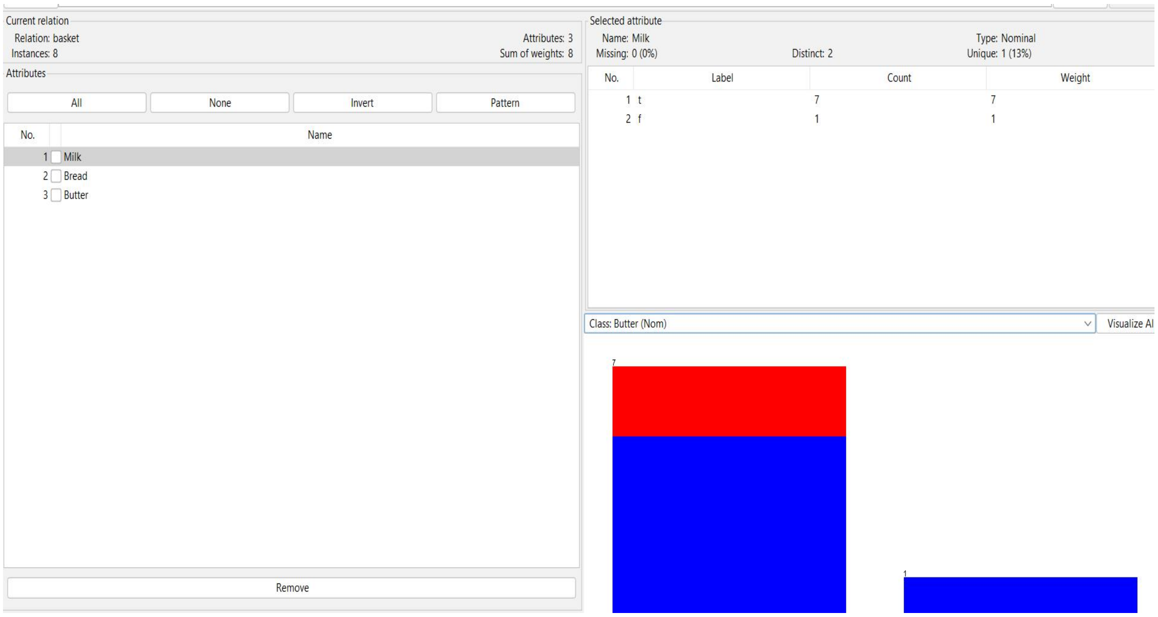
Aim:

To generate association rules using FP-Growth.

Algorithm:

1. Load transactions
2. Set min support
3. Build FP-tree
4. Generate rules

Output:



Current relation

Relation: basket | Attributes: 3

Instances: 8 | Sum of weights: 8

Selected attribute

Name: Milk | Type: Nominal

Missing: 0 (0%) | Distinct: 2 | Unique: 1 (13%)

No.	Label	Count	Weight
1	t	7	7
2	f	1	1

Attributes

No.	Name
1	□ Milk
2	□ Bread
3	□ Butter

Class: Butter (Nom)

7. Diabetes Prediction

Aim:

To predict diabetes using Decision Tree and compare with SVM.

Algorithm:

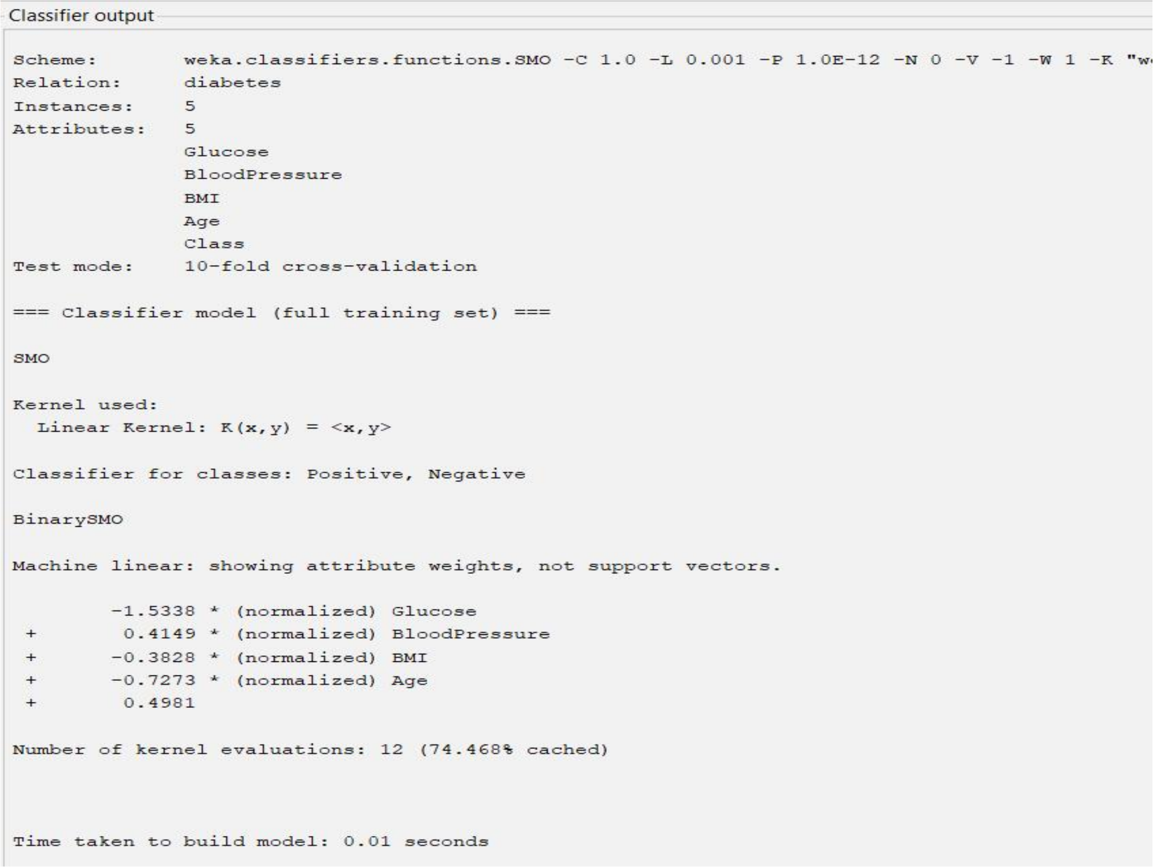
1. Load dataset

2. Apply Decision Tree

3. Apply SVM

4. Compare accuracy and F1

Output:



```
Scheme:       weka.classifiers.functions.SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "w
Relation:     diabetes
Instances:    5
Attributes:   5
              Glucose
              BloodPressure
              BMI
              Age
              Class
Test mode:    10-fold cross-validation

=== Classifier model (full training set) ===

SMO

Kernel used:
  Linear Kernel: K(x,y) = <x,y>

Classifier for classes: Positive, Negative

BinarySMO

Machine linear: showing attribute weights, not support vectors.

        -1.5338 * (normalized) Glucose
 +       0.4149 * (normalized) BloodPressure
 +      -0.3828 * (normalized) BMI
 +      -0.7273 * (normalized) Age
 +       0.4981

Number of kernel evaluations: 12 (74.468% cached)



Time taken to build model: 0.01 seconds
```

8. Data Binning

Aim:

To partition data using equal-width, equal-frequency and clustering.

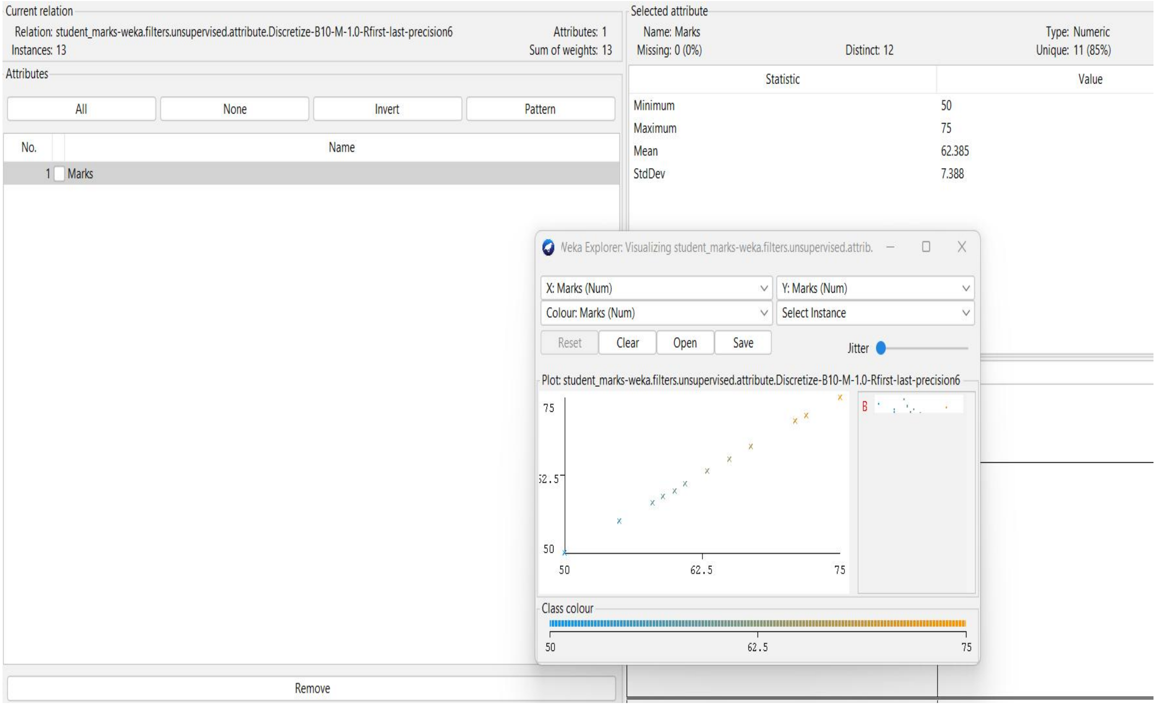
Algorithm:

1. Load marks

2. Apply binning methods

3. Plot histogram

Output:



Current relation

Relation: student_marks-weka.filters.unsupervised.attribute.Discretize-B10-M-1.0-Rfirst-last-precision6 Attributes: 1

Instances: 13 Sum of weights: 13

Selected attribute

Name: Marks Type: Numeric

Missing: 0 (0%) Distinct: 12 Unique: 11 (85%)

Statistic	Value
Minimum	50
Maximum	75
Mean	62.385
StdDev	7.388

No.	Name
1	□ Marks

Weka Explorer: Visualizing student_marks-weka.filters.unsupervised.attrib.

X: Marks (Num)

Y: Marks (Num)

Colour: Marks (Num)

Plot: student_marks-weka.filters.unsupervised.attribute.Discretize-B10-M-1.0-Rfirst-last-precision6

9. Decision Tree Dataset

Aim:

To create dataset and generate rules from decision tree.

Algorithm:

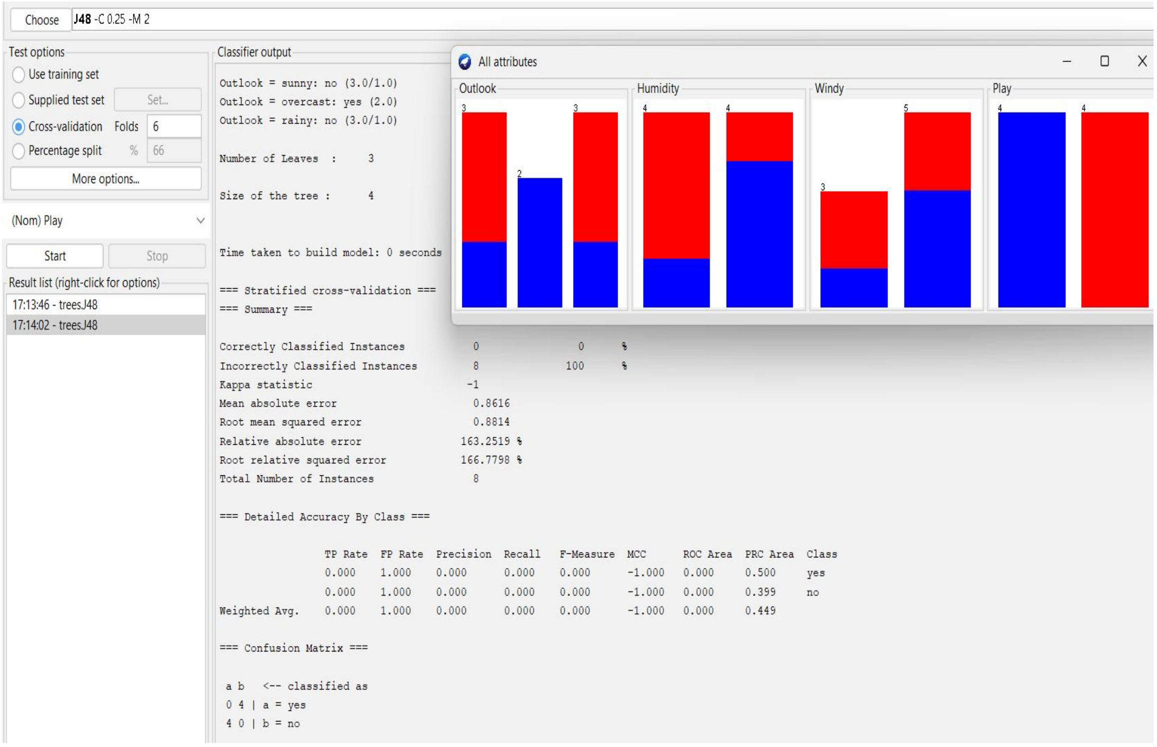
1. Create ARFF

2. Build tree

3. Generate rules

4. Plot confusion matrix

Output:



J48 -C 0.25 -M 2

○ Use training set
○ Supplied test set
● Cross-validation Folds 6
○ Percentage split % 66

(Nom) Play

17:13:46 - trees.J48
17:14:02 - trees.J48

```
Outlook = sunny: no (3.0/1.0)
Outlook = overcast: yes (2.0)
Outlook = rainy: no (3.0/1.0)

Number of Leaves  :     3

Size of the tree :      4


Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances           0                0      %
Incorrectly Classified Instances         8              100      %
Kappa statistic                         -1
Mean absolute error                      0.8616
Root mean squared error                  0.8814
Relative absolute error                163.2519 %
Root relative squared error            166.7798 %
Total Number of Instances                8

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.000    1.000    0.000      0.000    0.000      -1.000   0.000     0.500     yes
                 0.000    1.000    0.000      0.000    0.000      -1.000   0.000     0.399     no
Weighted Avg.    0.000    1.000    0.000      0.000    0.000      -1.000   0.000     0.449

=== Confusion Matrix ===

 a b   <-- classified as
 0 4 | a = yes
 4 0 | b = no
```

10. Apriori vs FP-Growth

Aim:

To compare Apriori and FP-Growth algorithms.

Algorithm:

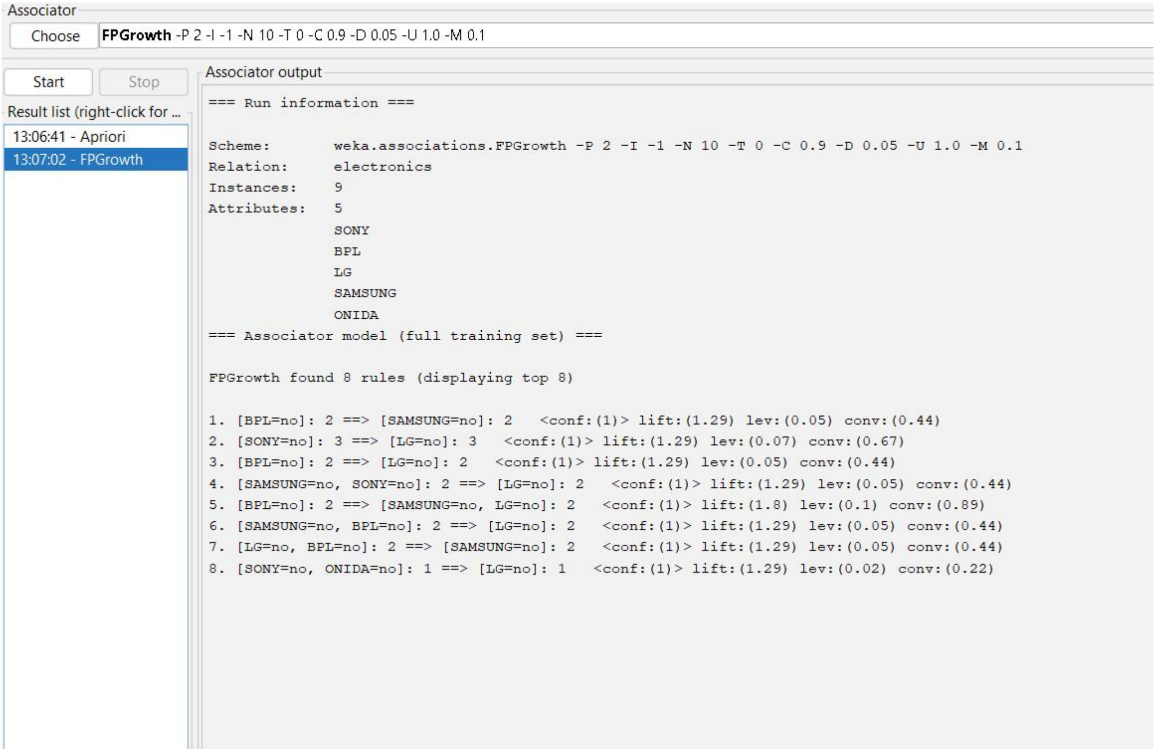
1. Load transaction data

2. Apply Apriori

3. Apply FP-Growth

4. Compare rules

Output:



FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1

```
=== Run information ===

Scheme:       weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation:     electronics
Instances:    9
Attributes:   5
              SONY
              BPL
              LG
              SAMSUNG
              ONIDA
=== Associator model (full training set) ===

FPGrowth found 8 rules (displaying top 8)

1. [BPL=no]: 2 ==> [SAMSUNG=no]: 2   <conf:(1)> lift:(1.29) lev:(0.05) conv:(0.44)
2. [SONY=no]: 3 ==> [LG=no]: 3   <conf:(1)> lift:(1.29) lev:(0.07) conv:(0.67)
3. [BPL=no]: 2 ==> [LG=no]: 2   <conf:(1)> lift:(1.29) lev:(0.05) conv:(0.44)
4. [SAMSUNG=no, SONY=no]: 2 ==> [LG=no]: 2   <conf:(1)> lift:(1.29) lev:(0.05) conv:(0.44)
5. [BPL=no]: 2 ==> [SAMSUNG=no, LG=no]: 2   <conf:(1)> lift:(1.8) lev:(0.1) conv:(0.89)
6. [SAMSUNG=no, BPL=no]: 2 ==> [LG=no]: 2   <conf:(1)> lift:(1.29) lev:(0.05) conv:(0.44)
7. [LG=no, BPL=no]: 2 ==> [SAMSUNG=no]: 2   <conf:(1)> lift:(1.29) lev:(0.05) conv:(0.44)
8. [SONY=no, ONIDA=no]: 1 ==> [LG=no]: 1   <conf:(1)> lift:(1.29) lev:(0.02) conv:(0.22)
```

11. Normalization Techniques

Aim:

To normalize data using different normalization methods.

Algorithm:

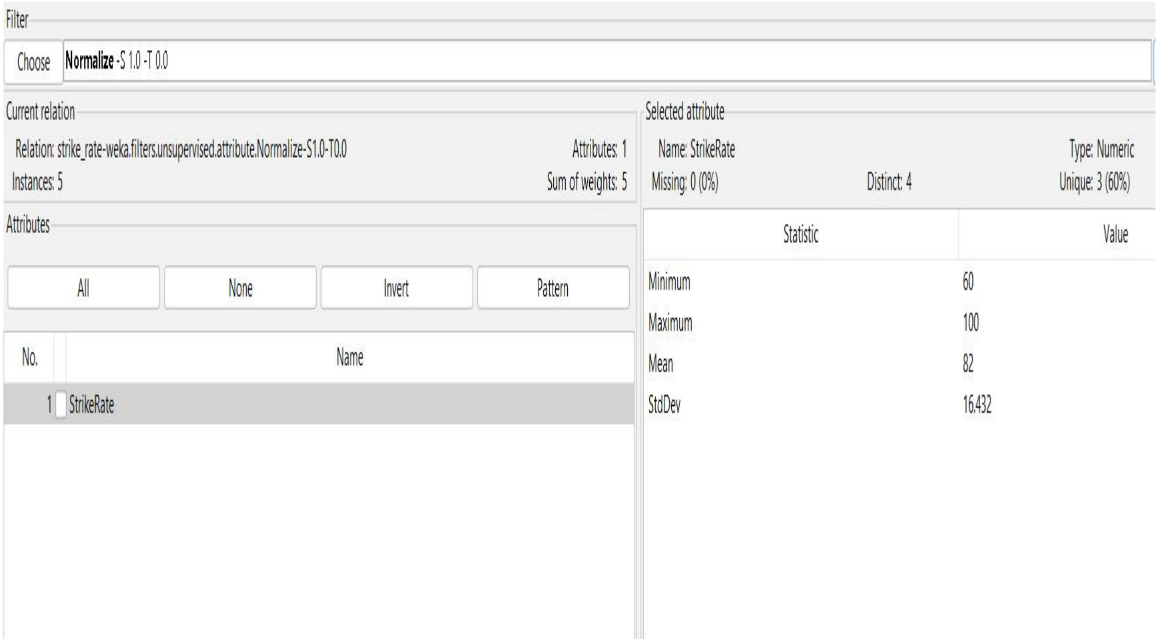
1. Load data

2. Apply min-max

3. Apply z-score

4. Apply decimal scaling

Output:



Normalize -S 1.0 -T 0.0

Relation: strike_rate-weka.filters.unsupervised.attribute.Normalize-S1.0-T0.0 Attributes: 1
Instances: 5 Sum of weights: 5

Name: StrikeRate Type: Numeric
Missing: 0 (0%) Distinct: 4 Unique: 3 (60%)

No.	Name
1	StrikeRate

Statistic	Value
Minimum	60
Maximum	100
Mean	82
StdDev	16.432

12. Mean and Variance Calculation

Aim:

To calculate variance and standard deviation for given data.

Algorithm:

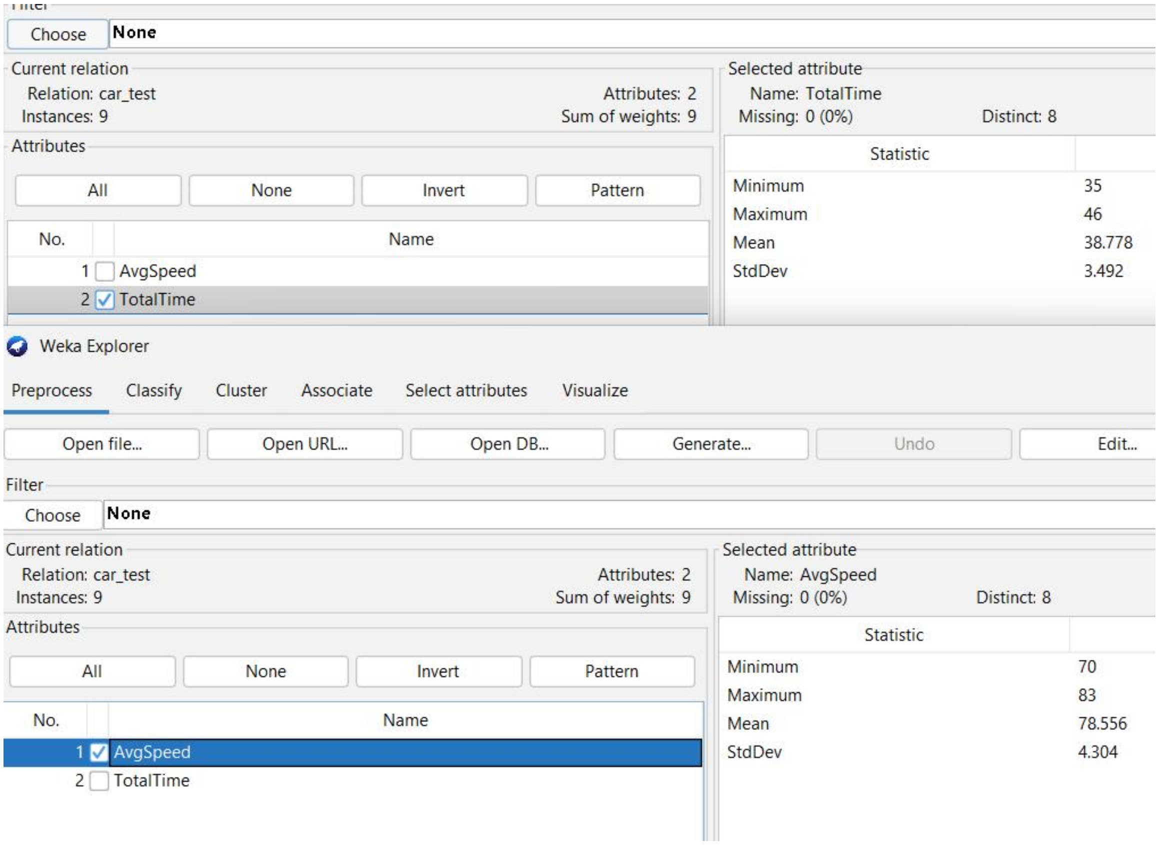
1. Load data

2. Compute mean

3. Compute variance

4. Compute standard deviation

Output:



Filter: None

Current relation
Relation: car_test
Instances: 9
Attributes: 2
Sum of weights: 9

Attributes

No.		Name
1	□	AvgSpeed
2	☑	TotalTime

Selected attribute
Name: TotalTime
Missing: 0 (0%)
Distinct: 8

Statistic	
Minimum	35
Maximum	46
Mean	38.778
StdDev	3.492

Filter: None

Current relation
Relation: car_test
Instances: 9
Attributes: 2
Sum of weights: 9

Attributes

No.		Name
1	☑	AvgSpeed
2	□	TotalTime

Selected attribute
Name: AvgSpeed
Missing: 0 (0%)
Distinct: 8

Statistic	
Minimum	70
Maximum	83
Mean	78.556
StdDev	4.304

13. Frequent Itemset Mining (Apriori & FP-Growth)

Aim:

To find frequent itemsets and generate association rules using Apriori and FP-Growth.

Algorithm:

1. Load transaction dataset

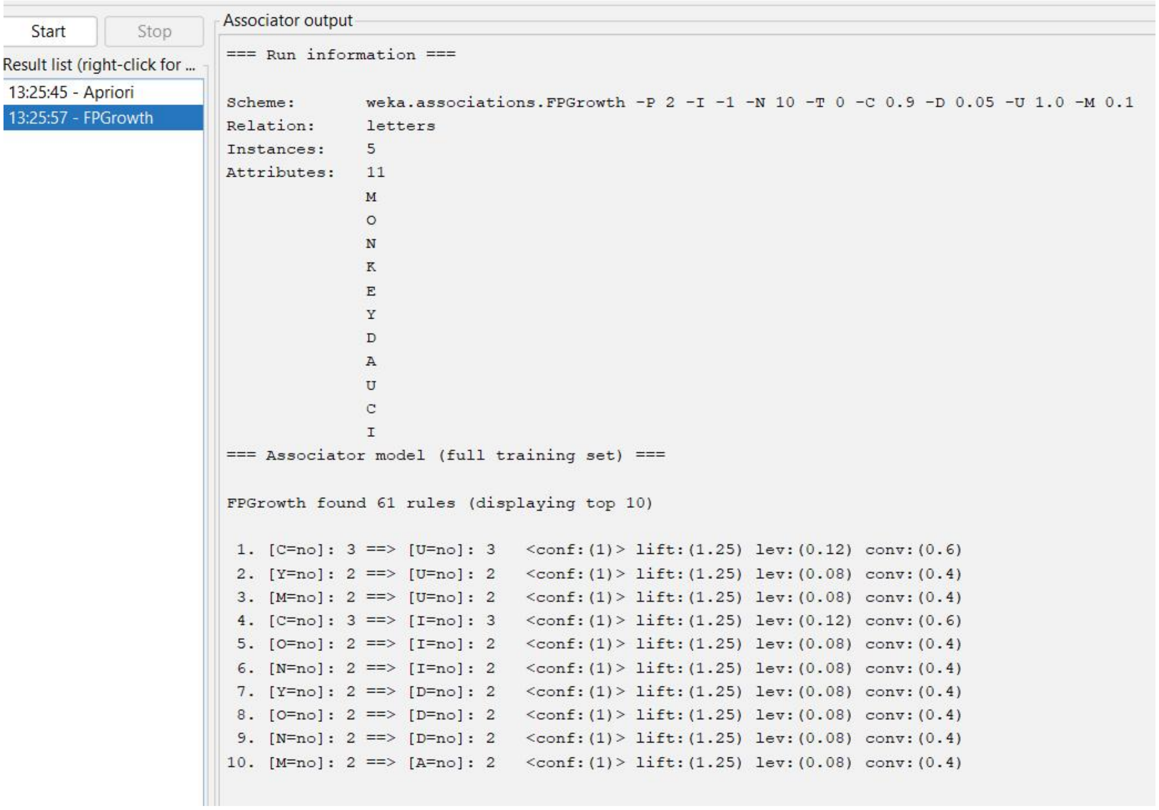
2. Set minimum support and confidence

3. Apply Apriori

4. Apply FP-Growth

5. Compare frequent itemsets and rules

Output:

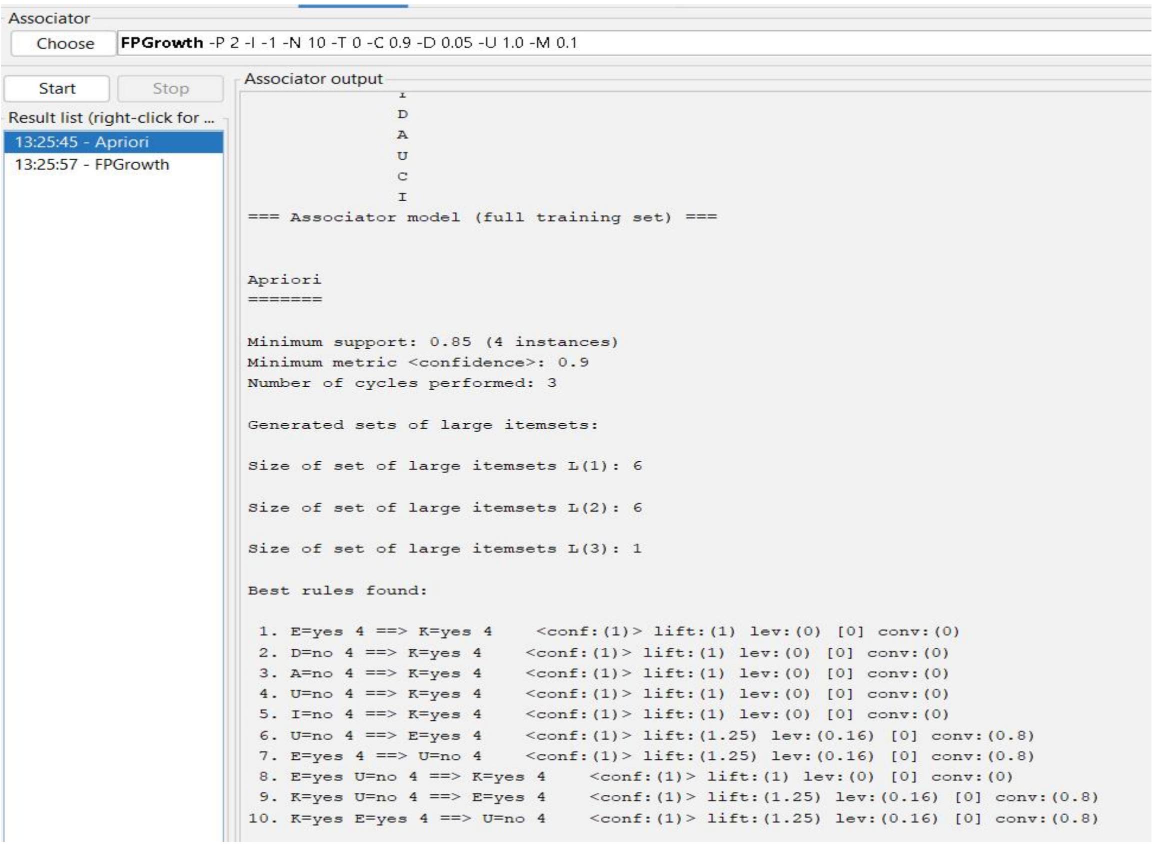


```
=== Run information ===

Scheme:       weka.associations.FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1
Relation:     letters
Instances:    5
Attributes:   11
              M
              O
              N
              K
              E
              Y
              D
              A
              U
              C
              I
=== Associator model (full training set) ===

FPGrowth found 61 rules (displaying top 10)

 1. [C=no]: 3 ==> [U=no]: 3   <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
 2. [Y=no]: 2 ==> [U=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 3. [M=no]: 2 ==> [U=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 4. [C=no]: 3 ==> [I=no]: 3   <conf:(1)> lift:(1.25) lev:(0.12) conv:(0.6)
 5. [O=no]: 2 ==> [I=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 6. [N=no]: 2 ==> [I=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 7. [Y=no]: 2 ==> [D=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 8. [O=no]: 2 ==> [D=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
 9. [N=no]: 2 ==> [D=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
10. [M=no]: 2 ==> [A=no]: 2   <conf:(1)> lift:(1.25) lev:(0.08) conv:(0.4)
```



FPGrowth -P 2 -I -1 -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1

```
               I
               D
               A
               U
               C
               I
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.85 (4 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 3

Generated sets of large itemsets:

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

Size of set of large itemsets L(2): 6

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

Best rules found:

 1. E=yes 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 2. D=no 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 3. A=no 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 4. U=no 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 5. I=no 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 6. U=no 4 ==> E=yes 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 7. E=yes 4 ==> U=no 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
 8. E=yes U=no 4 ==> K=yes 4    <conf:(1)> lift:(1) lev:(0) [0] conv:(0)
 9. K=yes U=no 4 ==> E=yes 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
10. K=yes E=yes 4 ==> U=no 4    <conf:(1)> lift:(1.25) lev:(0.16) [0] conv:(0.8)
```