VIT®
BHOPAL
www.vitbhopal.ac.in

VIT BHOPAL UNIVERSITY

School of Computing Science and Engineering

Bhopal-Indore Highway, Kothrikalan, Sehore
Madhya Pradesh - 466114

CSA3006 DATA MINING AND DATA WAREHOUSING

REG.NO : 23BAI10359

NAME : Kislay Anand

BRANCH : CSE(AI & ML)

SEMESTER: Winter Semester 2025-26

INDEX

EXP.NO: 01	**Exploring WEKA and Building an anonymous Data Warehouse**
DATE: 09.01.25	

AIM

To implement Decision Tree learning using WEKA and generate a decision tree model for classification.

PROCEDURE

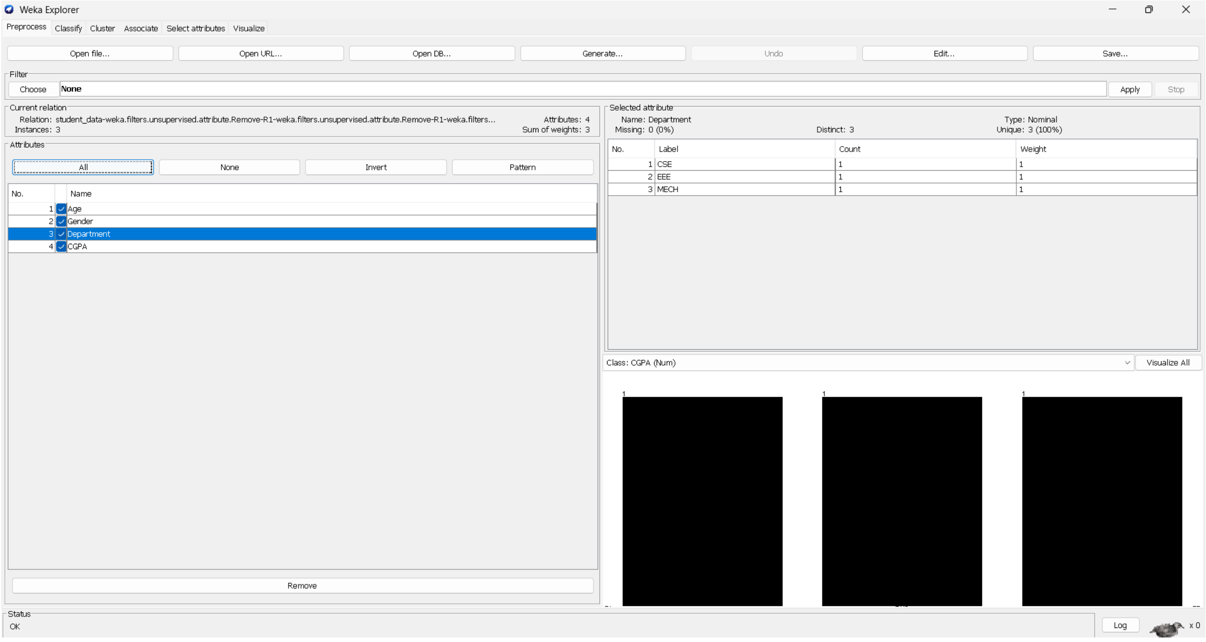
1. Open WEKA and click on Explorer.
2. Click Open file and load the dataset (.csv or .arff).
3. Go to the Preprocess tab and check the attributes.
4. Remove unwanted or sensitive attributes if needed.
5. Go to the Classify tab.
6. Click on Choose → trees → J48 (Decision Tree algorithm).
7. Select the class attribute (target variable).
8. Click on Start to run the classifier.
9. Observe the generated:
    - Decision tree
    - Accuracy
    - Confusion matrix

Save the model if required.

INPUT

A dataset containing attributes and class labels.
Input (anonymous_student_data.arff):

- Age
- Gender
- Department
- CGPA



Filter: None

Current relation
Relation: student_data-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.unsupervised.attribute.Remove-R1-weka.filters... Attributes: 4
Instances: 3 Sum of weights: 3

Attributes

No.		Name
1	☑	Age
2	☑	Gender
3	☑	Department
4	☑	CGPA

Selected attribute
Name: Department Type: Nominal
Missing: 0 (0%) Distinct: 3 Unique: 3 (100%)

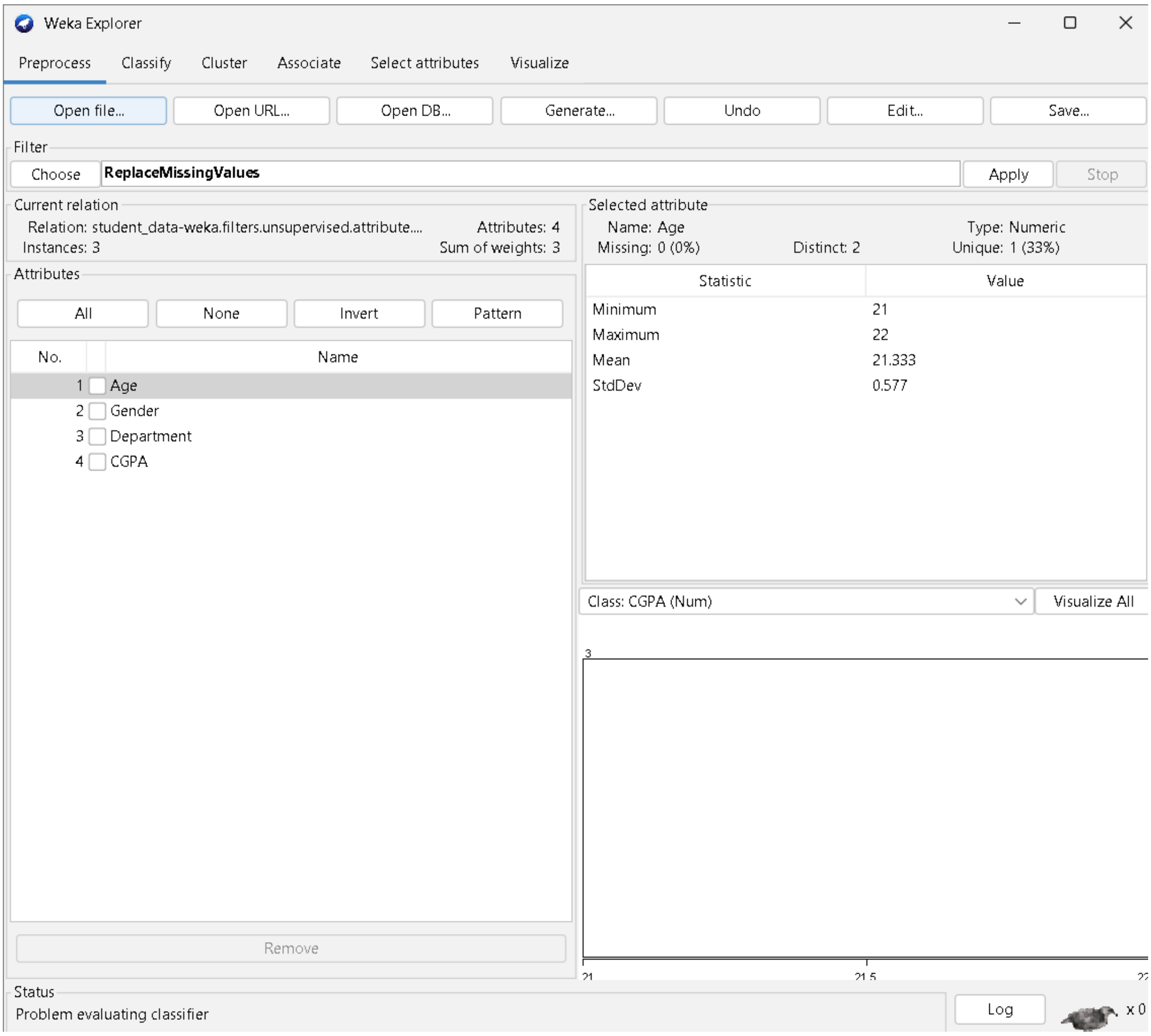
No.	Label	Count	Weight
1	CSE	1	1
2	EEE	1	1
3	MECH	1	1

Status
OK

OUTPUT

- A generated **Decision Tree model**
- Classification accuracy
- Confusion matrix
- Correctly and incorrectly classified instances

(WEKA displays all these after clicking Start)



Filter: ReplaceMissingValues

Current relation

Relation: student_data-weka.filters.unsupervised.attribute.... Attributes: 4
Instances: 3 Sum of weights: 3

Selected attribute

Name: Age Type: Numeric
Missing: 0 (0%) Distinct: 2 Unique: 1 (33%)

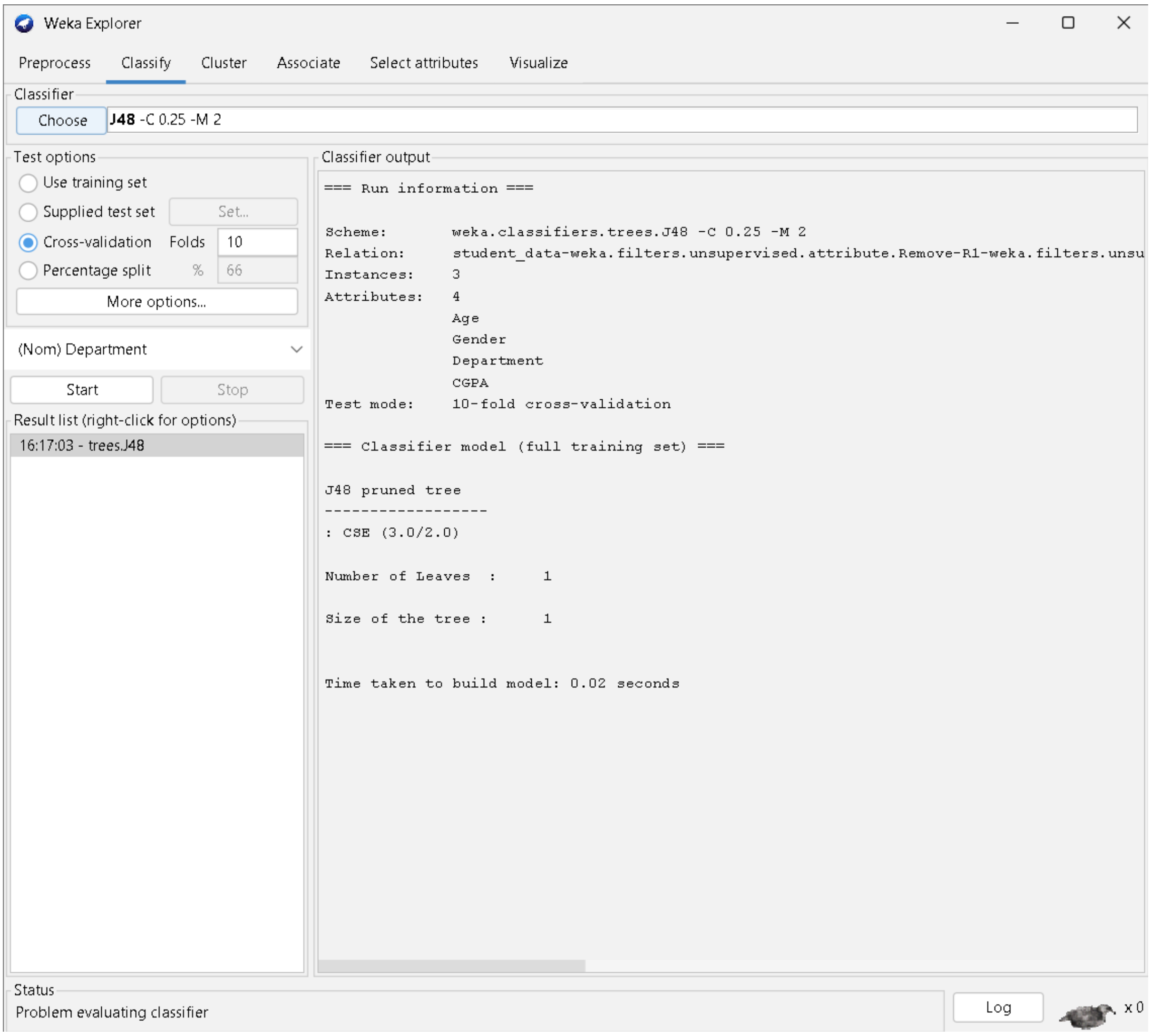
Statistic	Value
Minimum	21
Maximum	22
Mean	21.333
StdDev	0.577

No.		Name
1	□	Age
2	□	Gender
3	□	Department
4	□	CGPA

Class: CGPA (Num)

Status

Problem evaluating classifier



J48 -C 0.25 -M 2

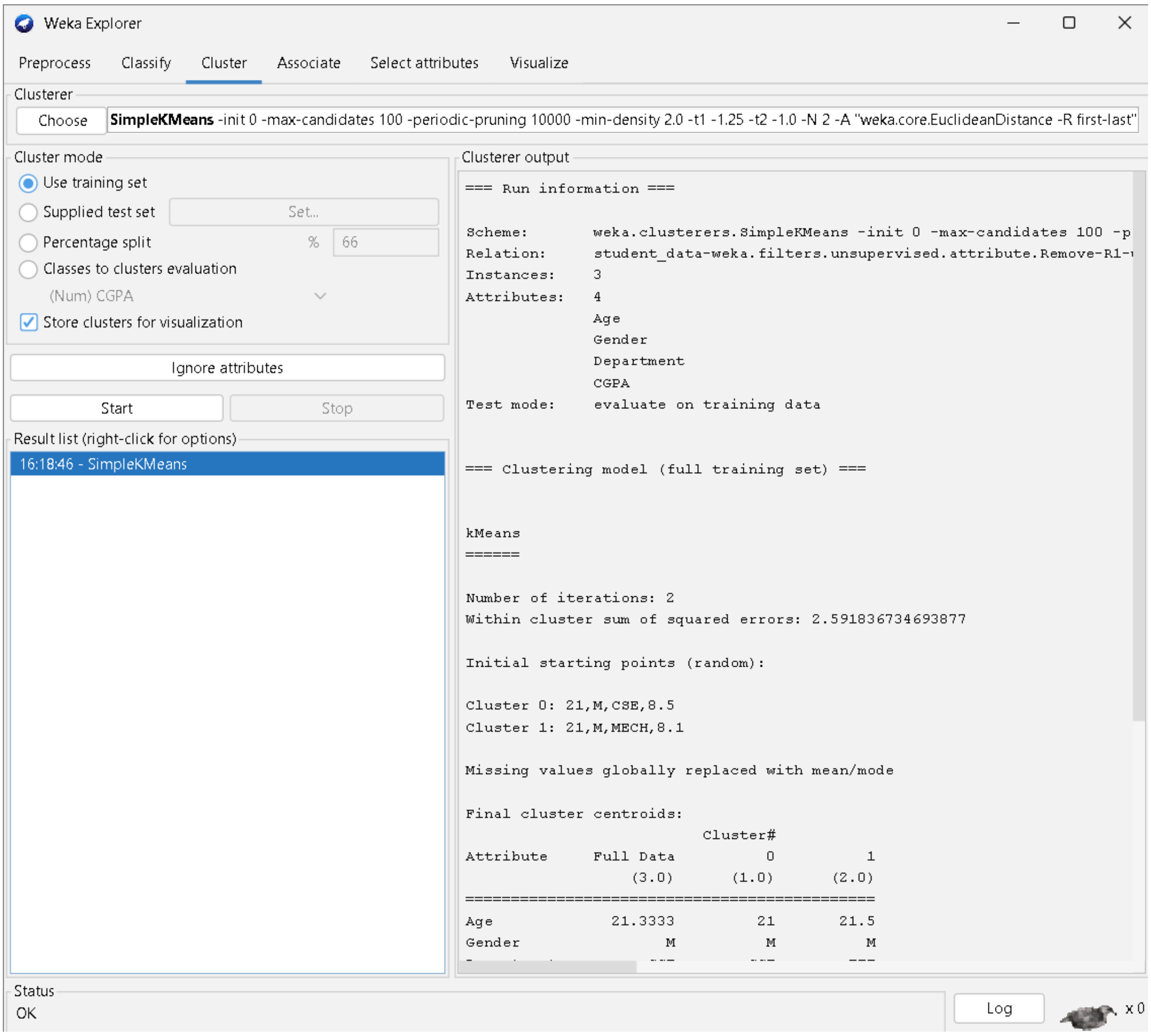
○ Use training set
● Cross-validation Folds 10

(Nom) Department

16:17:03 - trees.J48

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     student_data-weka.filters.unsupervised.attribute.Remove-R1-weka.filters.unsu
Instances:    3
Attributes:   4
              Age
              Gender
              Department
              CGPA
Test mode:    10-fold cross-validation

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

J48 pruned tree
------------------
: CSE (3.0/2.0)

Number of Leaves  :     1

Size of the tree :      1


Time taken to build model: 0.02 seconds
```

Problem evaluating classifier



Clusterer

SimpleKMeans -init 0 -max-candidates 100 -periodic-pruning 10000 -min-density 2.0 -t1 -1.25 -t2 -1.0 -N 2 -A "weka.core.EuclideanDistance -R first-last"

Cluster mode

- ● Use training set
- ○ Supplied test set
- ○ Percentage split % 66
- ○ Classes to clusters evaluation
- (Num) CGPA
- ☑ Store clusters for visualization

Result list (right-click for options)

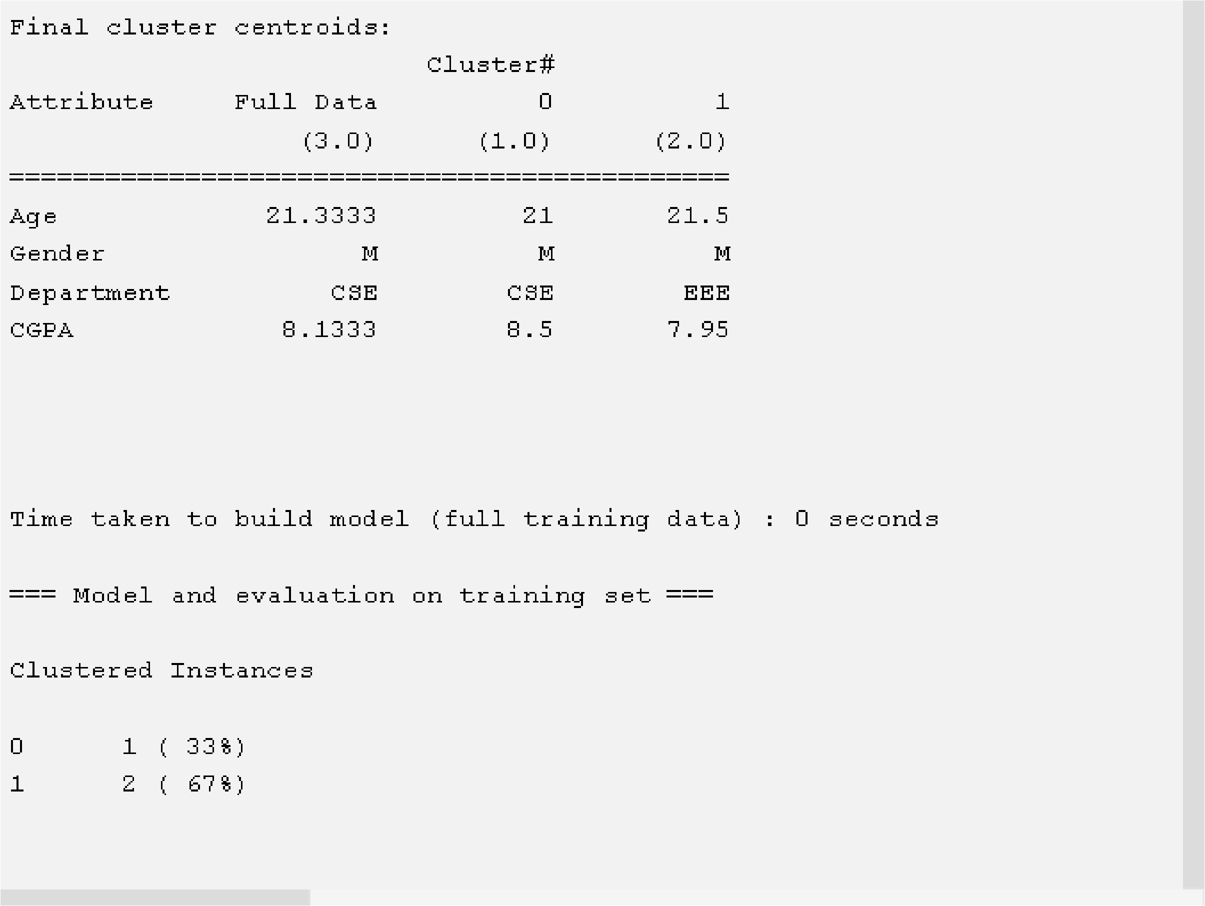
16:18:46 - SimpleKMeans

Clusterer output

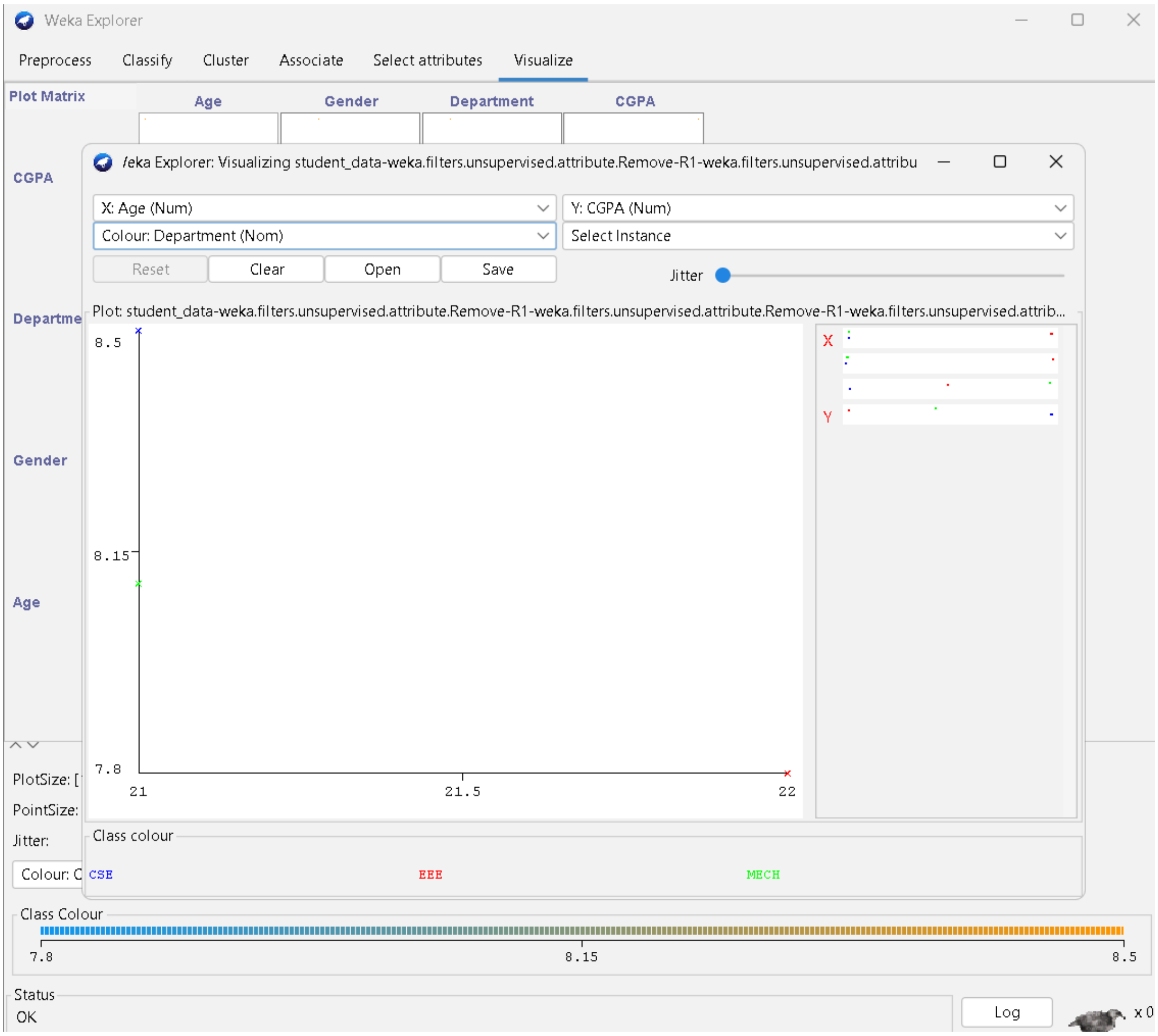
```
=== Run information ===

Scheme:       weka.clusterers.SimpleKMeans -init 0 -max-candidates 100 -p
Relation:     student_data-weka.filters.unsupervised.attribute.Remove-R1-
Instances:    3
Attributes:   4
              Age
              Gender
              Department
              CGPA
Test mode:    evaluate on training data


=== Clustering model (full training set) ===


kMeans
======

Number of iterations: 2
Within cluster sum of squared errors: 2.591836734693877

Initial starting points (random):

Cluster 0: 21,M,CSE,8.5
Cluster 1: 21,M,MECH,8.1

Missing values globally replaced with mean/mode

Final cluster centroids:
                          Cluster#
Attribute      Full Data          0          1
                   (3.0)      (1.0)      (2.0)
==============================================
Age              21.3333         21       21.5
Gender                 M          M          M
```

Status

OK



```
Final cluster centroids:
                          Cluster#
Attribute     Full Data          0          1
                  (3.0)      (1.0)      (2.0)
=============================================
Age             21.3333         21       21.5
Gender                M          M          M
Department          CSE        CSE        EEE
CGPA             8.1333        8.5       7.95




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

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

Clustered Instances

0       1 ( 33%)
1       2 ( 67%)
```



RESULT

Thus, the Decision Tree learning algorithm (J48) was successfully implemented using WEKA and a classification model was generated from the given dataset.

EXP.NO: 02	**Implementation of several data pre-processing tasks on datasets.**
DATE:	

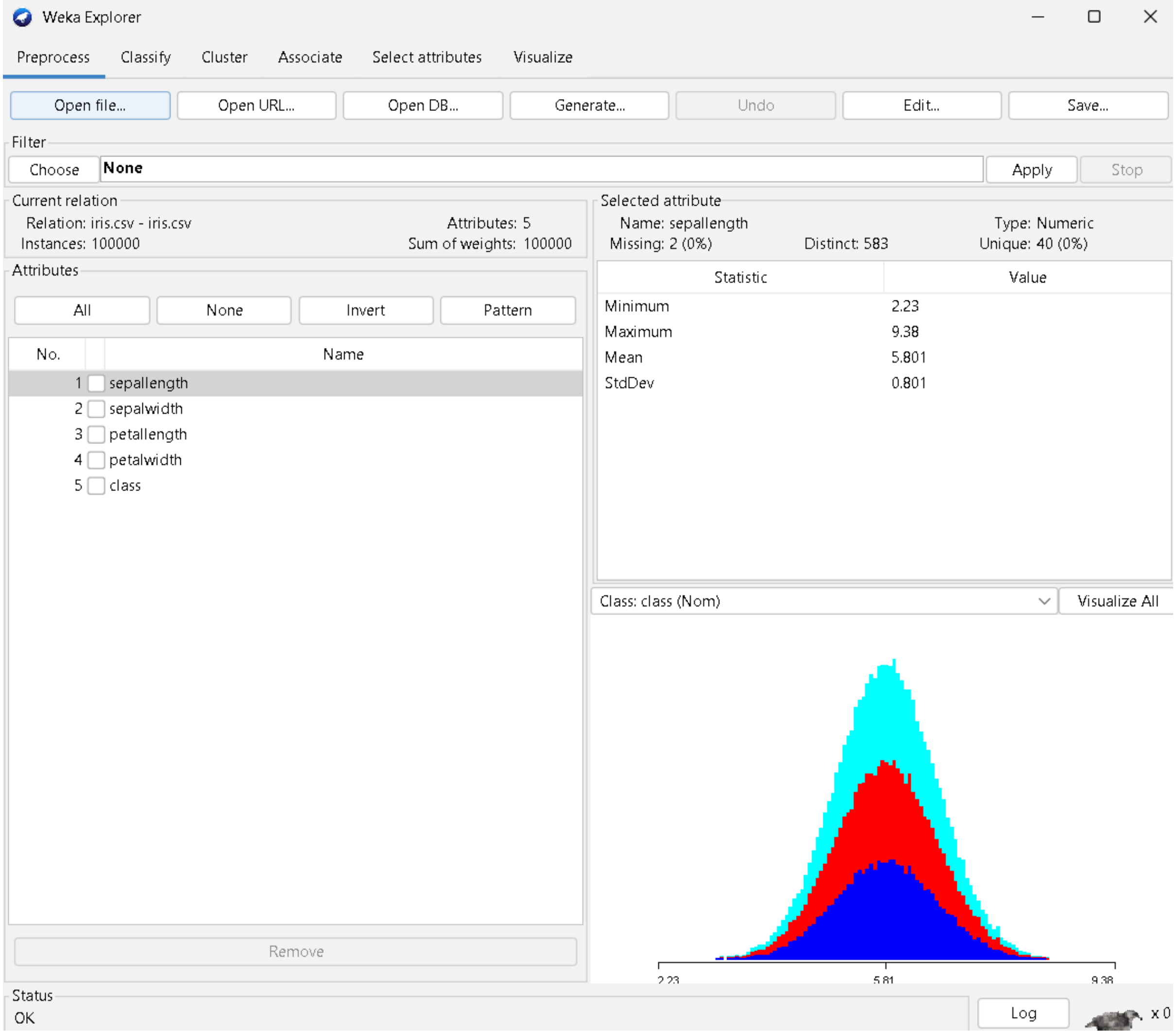
AIM

To perform **data pre-processing** on a given dataset using the **WEKA tool**, including missing value handling, normalization, discretization, attribute selection, and attribute removal.

PROCEDURE

1. Opened the **WEKA application** and selected **Explorer** mode.

2. Loaded the given dataset using the **Open file** option in the Preprocess tab.



Filter: None

Relation: iris.csv - iris.csv
Attributes: 5
Instances: 100000
Sum of weights: 100000

No.	Name
1	□ sepallength
2	□ sepalwidth
3	□ petallength
4	□ petalwidth
5	□ class

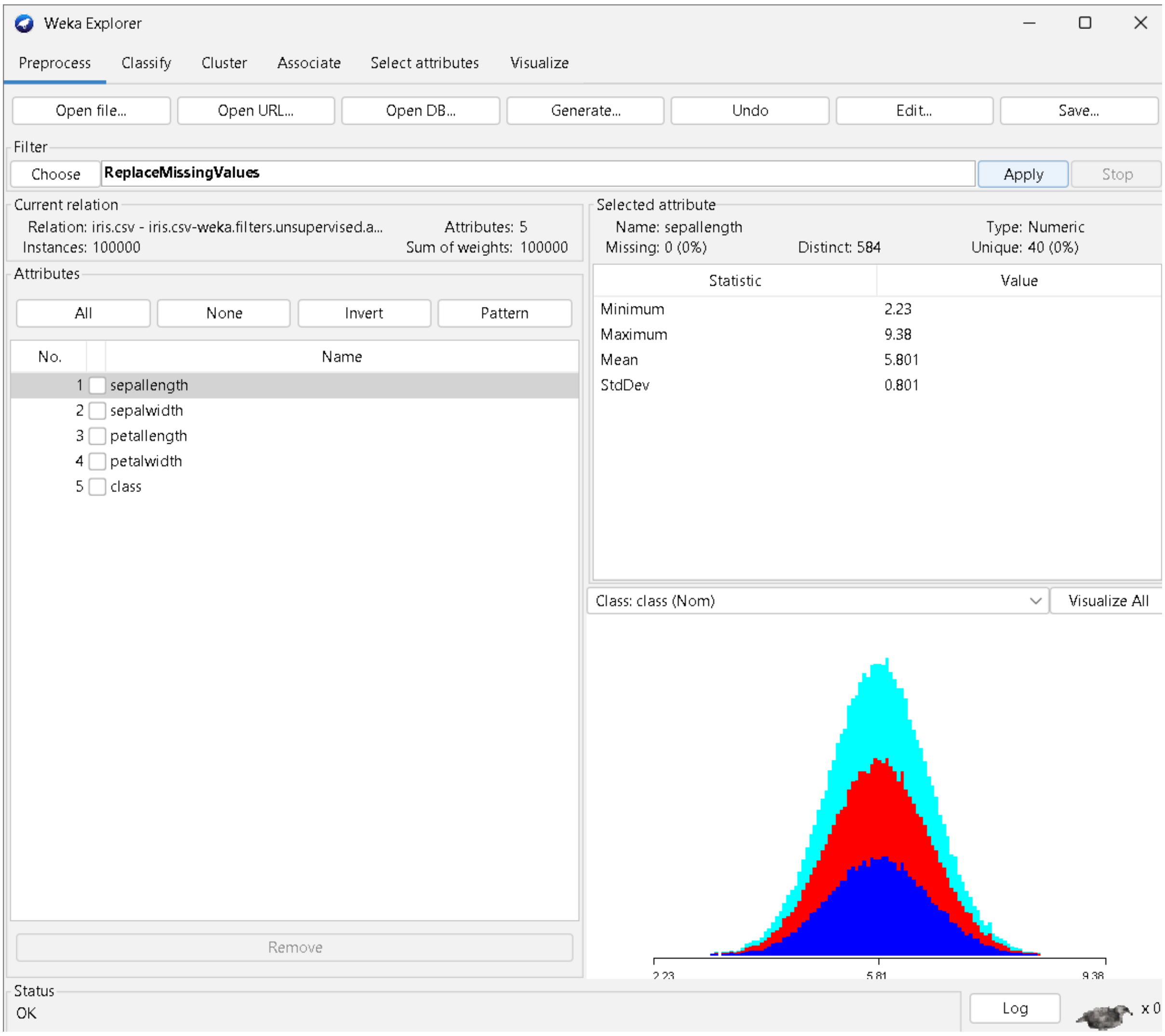
Name: sepallength
Type: Numeric
Missing: 2 (0%)
Distinct: 583
Unique: 40 (0%)

Statistic	Value
Minimum	2.23
Maximum	9.38
Mean	5.801
StdDev	0.801

Class: class (Nom)

Status: OK

3. Performed **missing value handling** using the ReplaceMissingValues filter.



Filter: ReplaceMissingValues

Current relation
Relation: iris.csv - iris.csv-weka.filters.unsupervised.a... Attributes: 5
Instances: 100000 Sum of weights: 100000

Attributes

No.	Name
1	☐ sepallength
2	☐ sepalwidth
3	☐ petallength
4	☐ petalwidth
5	☐ class

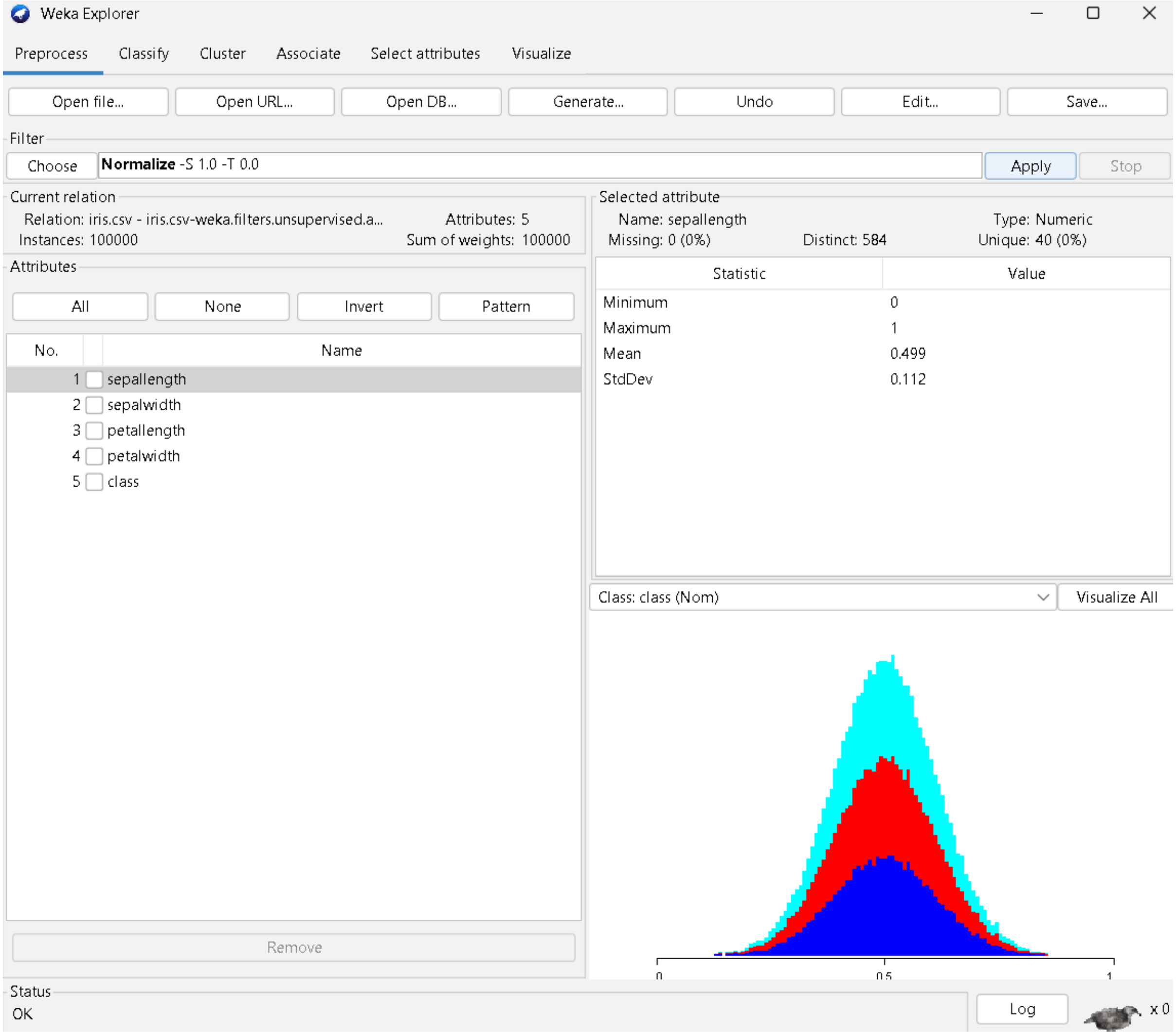
Selected attribute
Name: sepallength Type: Numeric
Missing: 0 (0%) Distinct: 584 Unique: 40 (0%)

Statistic	Value
Minimum	2.23
Maximum	9.38
Mean	5.801
StdDev	0.801

Class: class (Nom)

Status
OK

4. Applied **normalization** to scale numeric attributes using the Normalize filter.



Filter

Normalize -S 1.0 -T 0.0

Current relation

Relation: iris.csv - iris.csv-weka.filters.unsupervised.a... Attributes: 5

Instances: 100000 Sum of weights: 100000

Attributes

No.		Name
1	□	sepallength
2	□	sepalwidth
3	□	petallength
4	□	petalwidth
5	□	class

Selected attribute

Name: sepallength Type: Numeric

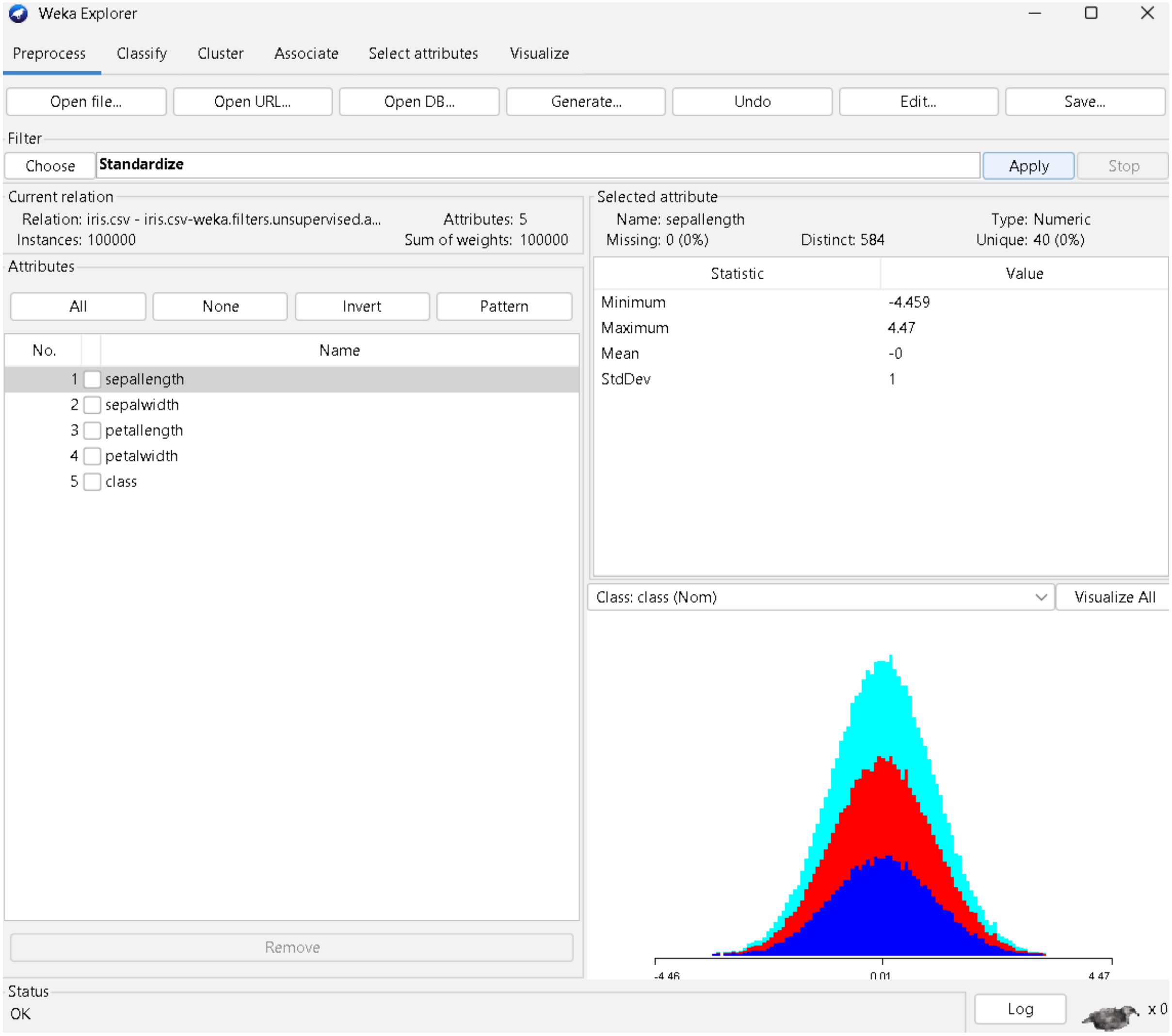
Missing: 0 (0%) Distinct: 584 Unique: 40 (0%)

Statistic	Value
Minimum	0
Maximum	1
Mean	0.499
StdDev	0.112

Class: class (Nom)

Status

OK



Filter

Standardize

Current relation

Relation: iris.csv - iris.csv-weka.filters.unsupervised.a...   Attributes: 5

Instances: 100000   Sum of weights: 100000

Attributes

No.		Name
1	□	sepallength
2	□	sepalwidth
3	□	petallength
4	□	petalwidth
5	□	class

Selected attribute

Name: sepallength   Type: Numeric

Missing: 0 (0%)   Distinct: 584   Unique: 40 (0%)

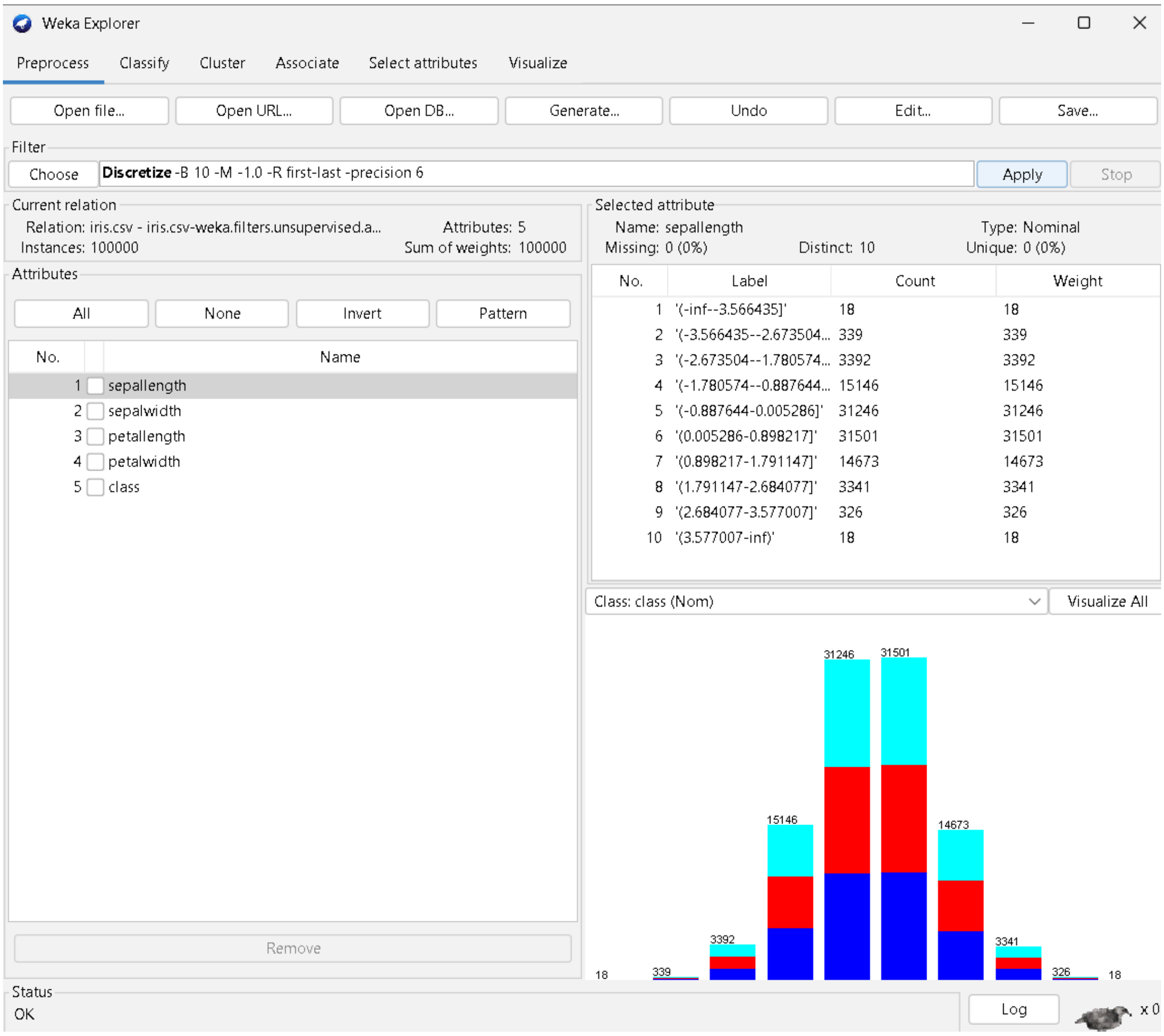
Statistic	Value
Minimum	-4.459
Maximum	4.47
Mean	-0
StdDev	1

Class: class (Nom)

Status

OK

5. Converted numeric attributes into categorical values using the Discretize filter.



Filter

Discretize -B 10 -M -1.0 -R first-last -precision 6

Current relation

Relation: iris.csv - iris.csv-weka.filters.unsupervised.a... Attributes: 5

Instances: 100000 Sum of weights: 100000

Attributes

No.		Name
1	□	sepallength
2	□	sepalwidth
3	□	petallength
4	□	petalwidth
5	□	class

Selected attribute

Name: sepallength Type: Nominal

Missing: 0 (0%) Distinct: 10 Unique: 0 (0%)

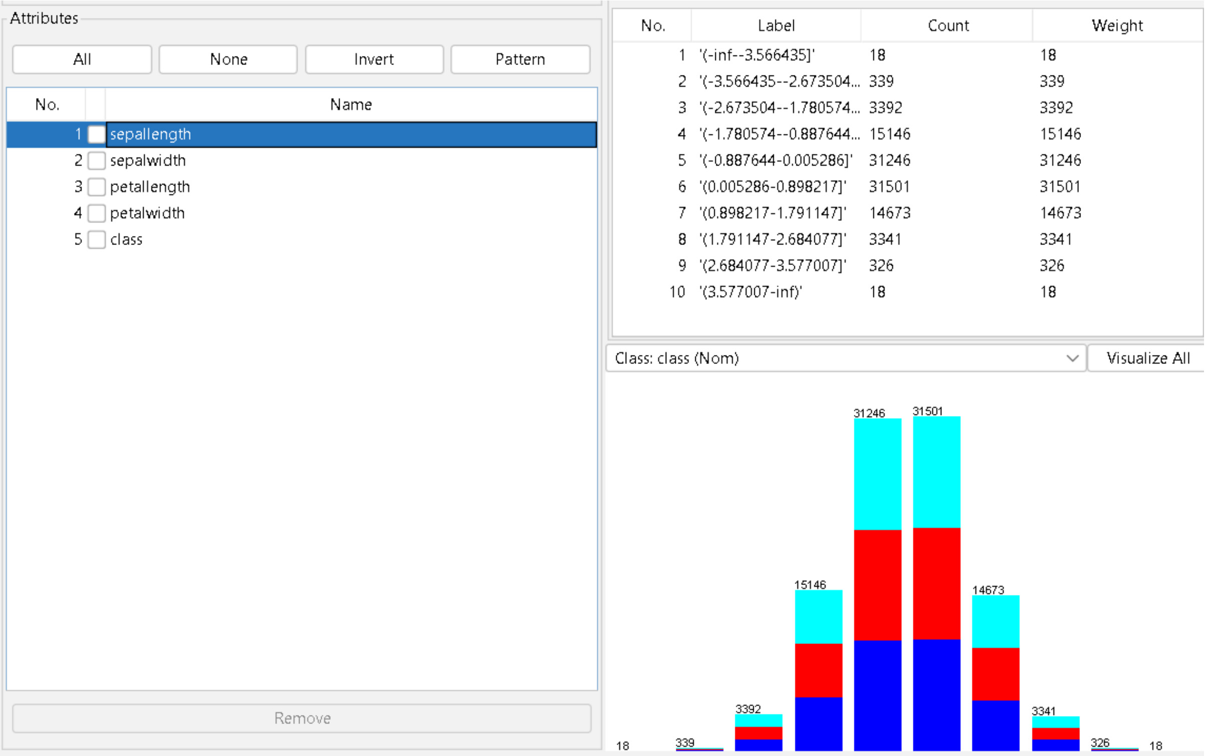
No.	Label	Count	Weight
1	'(-inf--3.566435]'	18	18
2	'(-3.566435--2.673504...	339	339
3	'(-2.673504--1.780574...	3392	3392
4	'(-1.780574--0.887644...	15146	15146
5	'(-0.887644-0.005286]'	31246	31246
6	'(0.005286-0.898217]'	31501	31501
7	'(0.898217-1.791147]'	14673	14673
8	'(1.791147-2.684077]'	3341	3341
9	'(2.684077-3.577007]'	326	326
10	'(3.577007-inf)'	18	18

Class: class (Nom)

Status

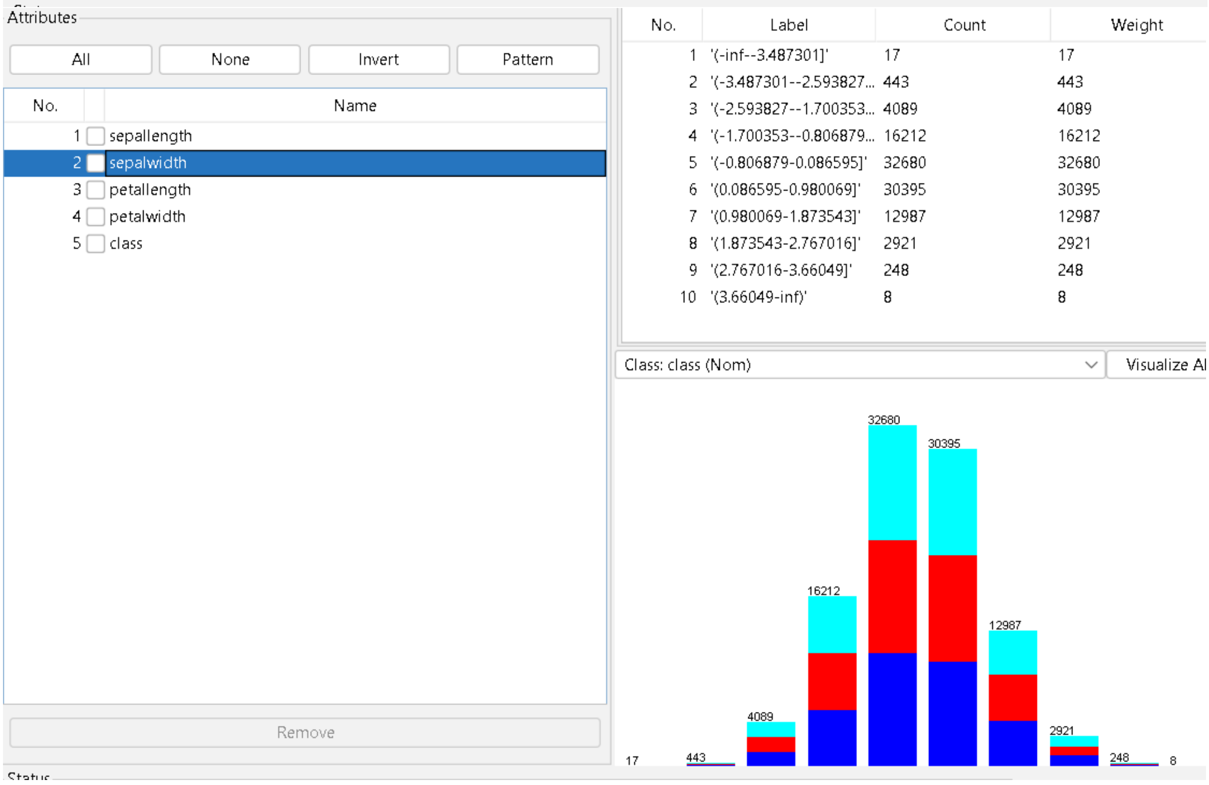
OK

6. Selected relevant attributes using the **Select Attributes** tab with suitable evaluators.



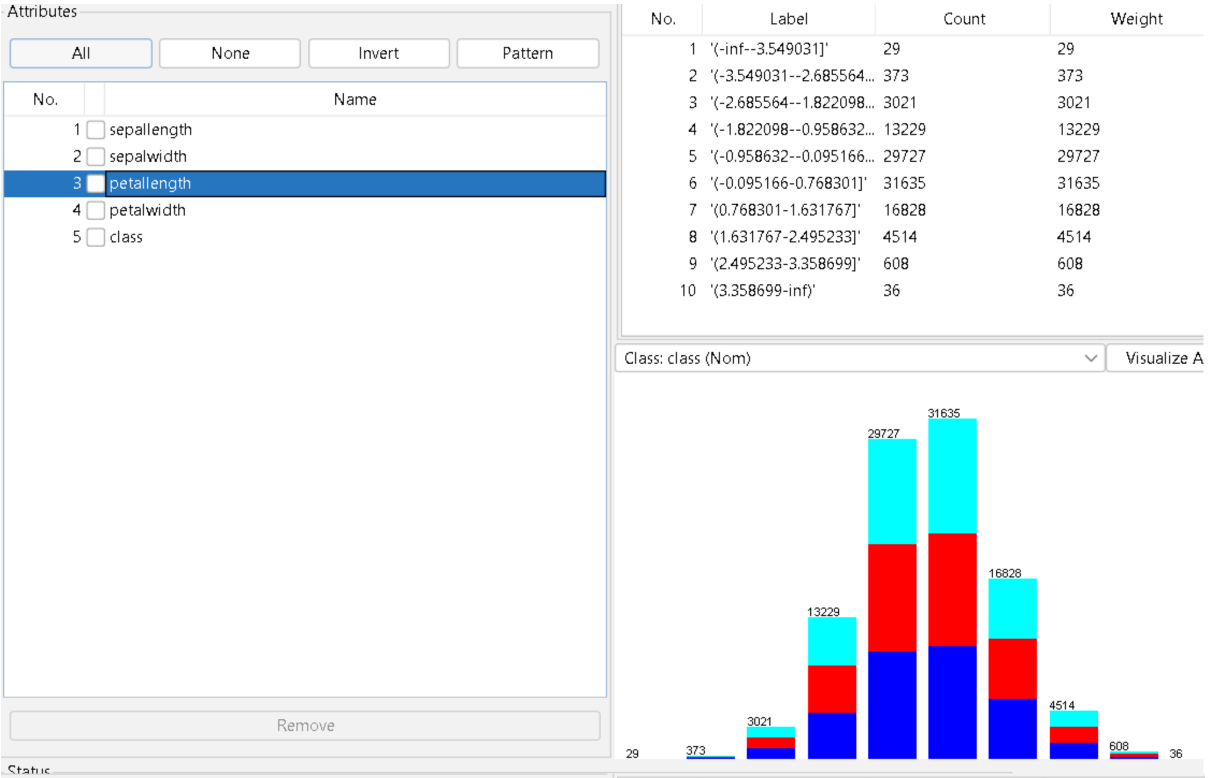
No.		Name
1	□	sepallength
2	□	sepalwidth
3	□	petallength
4	□	petalwidth
5	□	class

No.	Label	Count	Weight
1	'(-inf--3.566435]'	18	18
2	'(-3.566435--2.673504...	339	339
3	'(-2.673504--1.780574...	3392	3392
4	'(-1.780574--0.887644...	15146	15146
5	'(-0.887644-0.005286]'	31246	31246
6	'(0.005286-0.898217]'	31501	31501
7	'(0.898217-1.791147]'	14673	14673
8	'(1.791147-2.684077]'	3341	3341
9	'(2.684077-3.577007]'	326	326
10	'(3.577007-inf)'	18	18



No.		Name
1	□	sepallength
2	□	sepalwidth
3	□	petallength
4	□	petalwidth
5	□	class

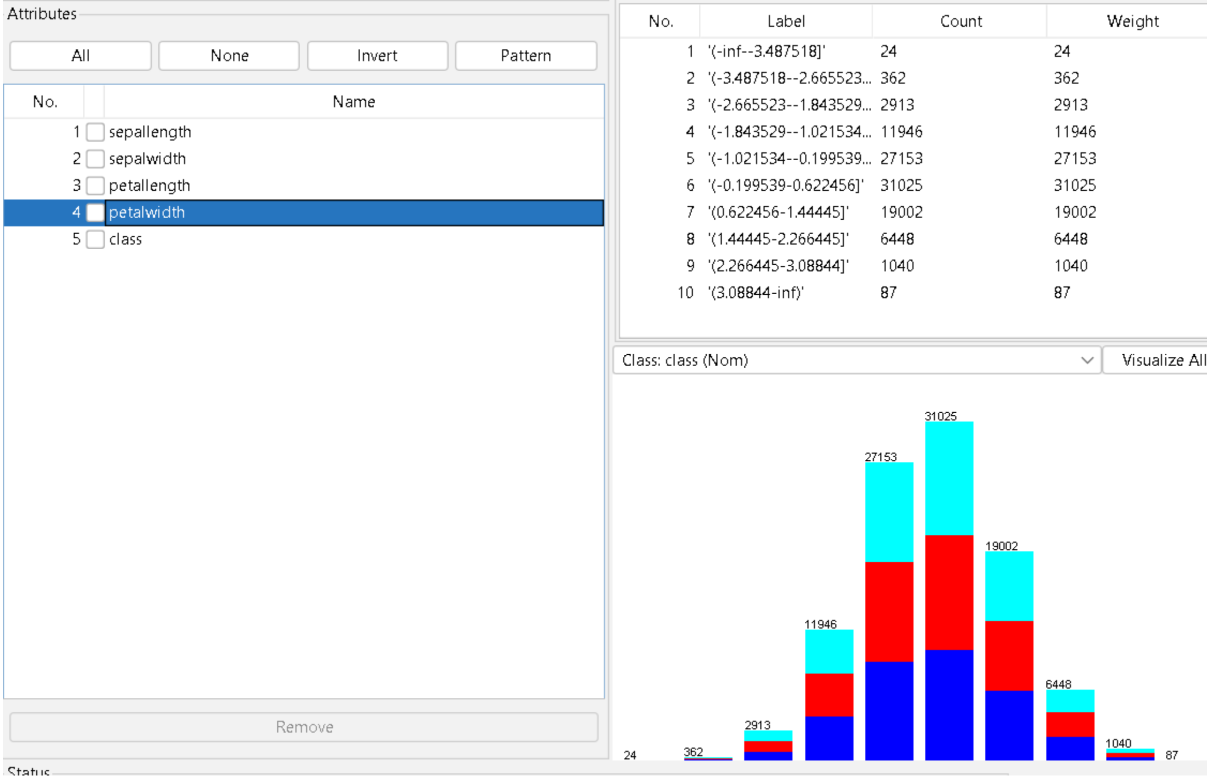
No.	Label	Count	Weight
1	'(-inf--3.487301]'	17	17
2	'(-3.487301--2.593827...	443	443
3	'(-2.593827--1.700353...	4089	4089
4	'(-1.700353--0.806879...	16212	16212
5	'(-0.806879-0.086595]'	32680	32680
6	'(0.086595-0.980069]'	30395	30395
7	'(0.980069-1.873543]'	12987	12987
8	'(1.873543-2.767016]'	2921	2921
9	'(2.767016-3.66049]'	248	248
10	'(3.66049-inf)'	8	8



No.	Name
1	□ sepallength
2	□ sepalwidth
3	□ petallength
4	□ petalwidth
5	□ class

No.	Label	Count	Weight
1	'(-inf--3.549031]'	29	29
2	'(-3.549031--2.685564...	373	373
3	'(-2.685564--1.822098...	3021	3021
4	'(-1.822098--0.958632...	13229	13229
5	'(-0.958632--0.095166...	29727	29727
6	'(-0.095166-0.768301]'	31635	31635
7	'(0.768301-1.631767]'	16828	16828
8	'(1.631767-2.495233]'	4514	4514
9	'(2.495233-3.358699]'	608	608
10	'(3.358699-inf)'	36	36

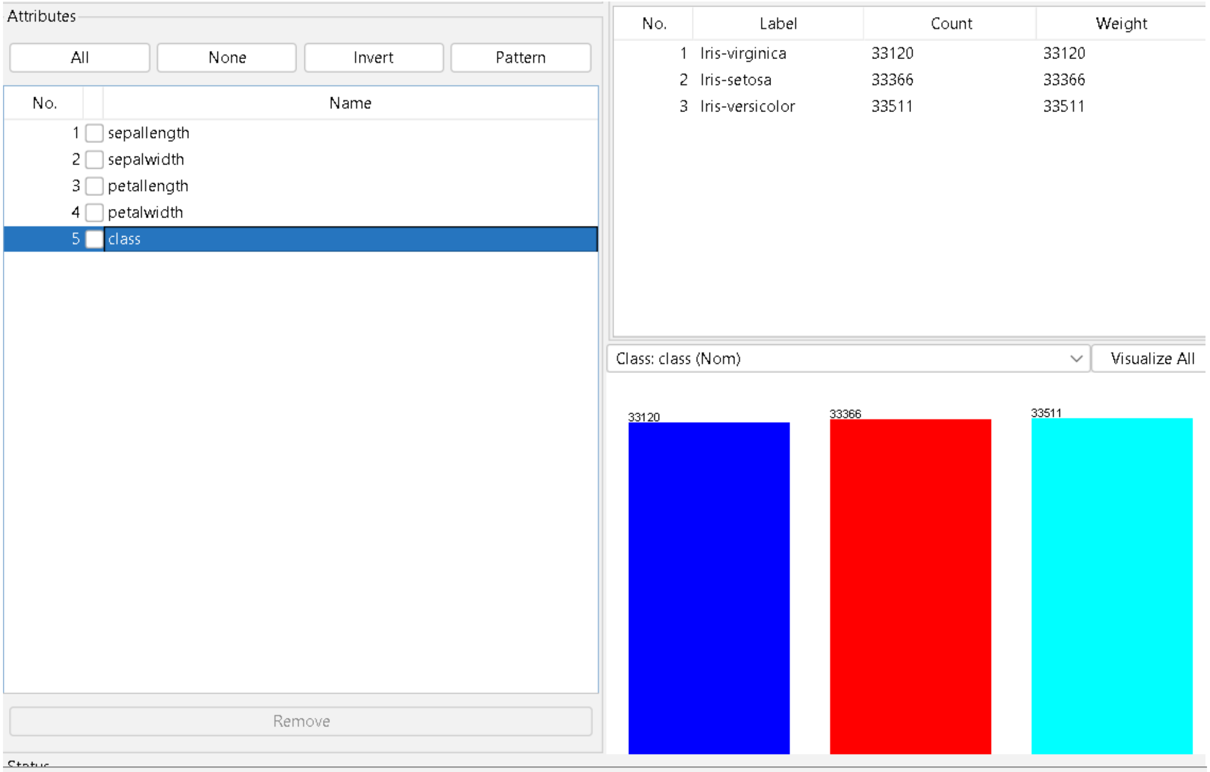
Class: class (Nom)



No.	Name
1	□ sepallength
2	□ sepalwidth
3	□ petallength
4	□ petalwidth
5	□ class

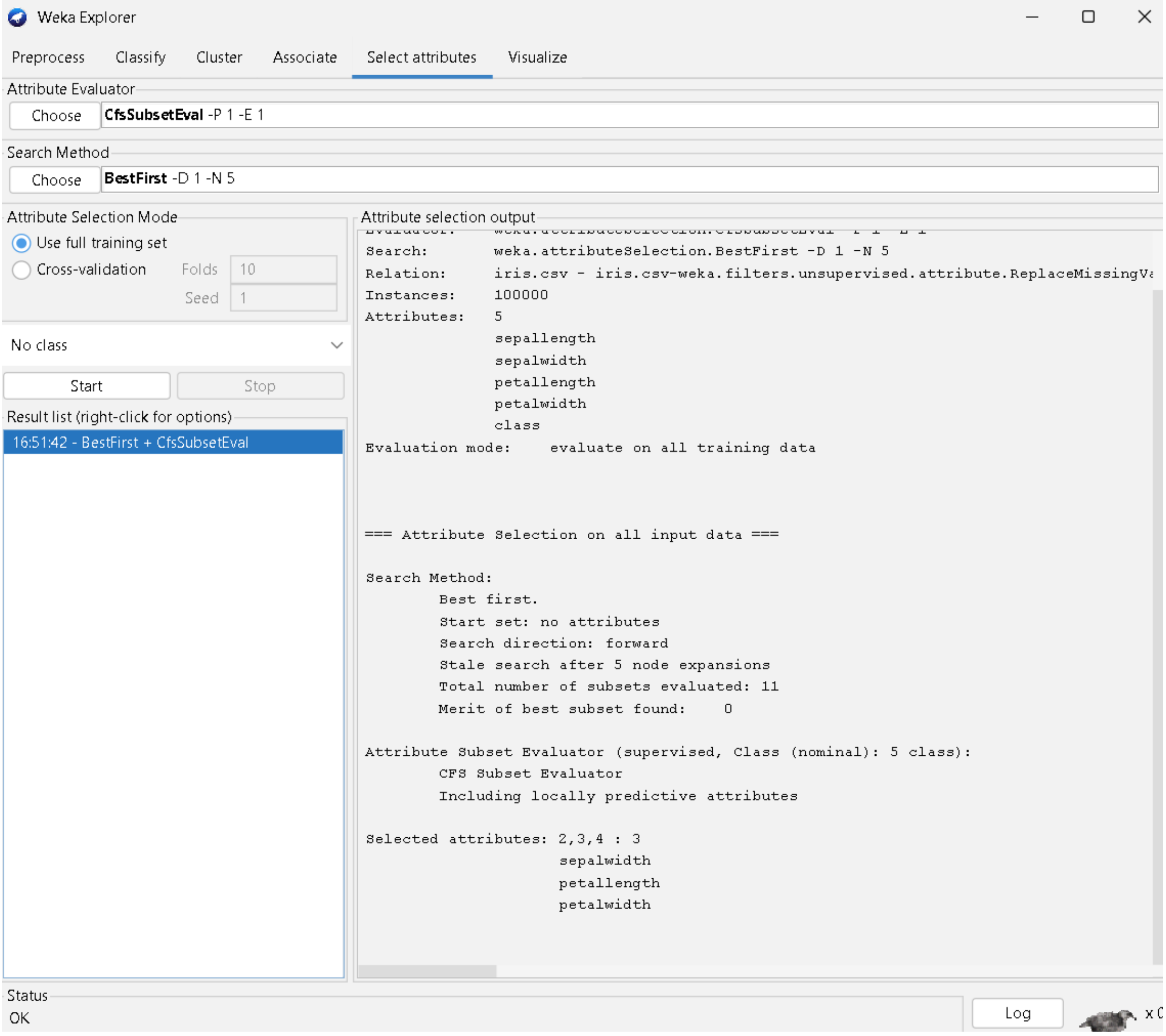
No.	Label	Count	Weight
1	'(-inf--3.487518]'	24	24
2	'(-3.487518--2.665523...	362	362
3	'(-2.665523--1.843529...	2913	2913
4	'(-1.843529--1.021534...	11946	11946
5	'(-1.021534--0.199539...	27153	27153
6	'(-0.199539-0.622456]'	31025	31025
7	'(0.622456-1.44445]'	19002	19002
8	'(1.44445-2.266445]'	6448	6448
9	'(2.266445-3.08844]'	1040	1040
10	'(3.08844-inf)'	87	87

Class: class (Nom)



No.	Name
1	□ sepallength
2	□ sepalwidth
3	□ petallength
4	□ petalwidth
5	□ class

No.	Label	Count	Weight
1	Iris-virginica	33120	33120
2	Iris-setosa	33366	33366
3	Iris-versicolor	33511	33511



Attribute Evaluator: **CfsSubsetEval** -P 1 -E 1

Search Method: **BestFirst** -D 1 -N 5

Attribute Selection Mode
● Use full training set
○ Cross-validation   Folds 10   Seed 1

No class

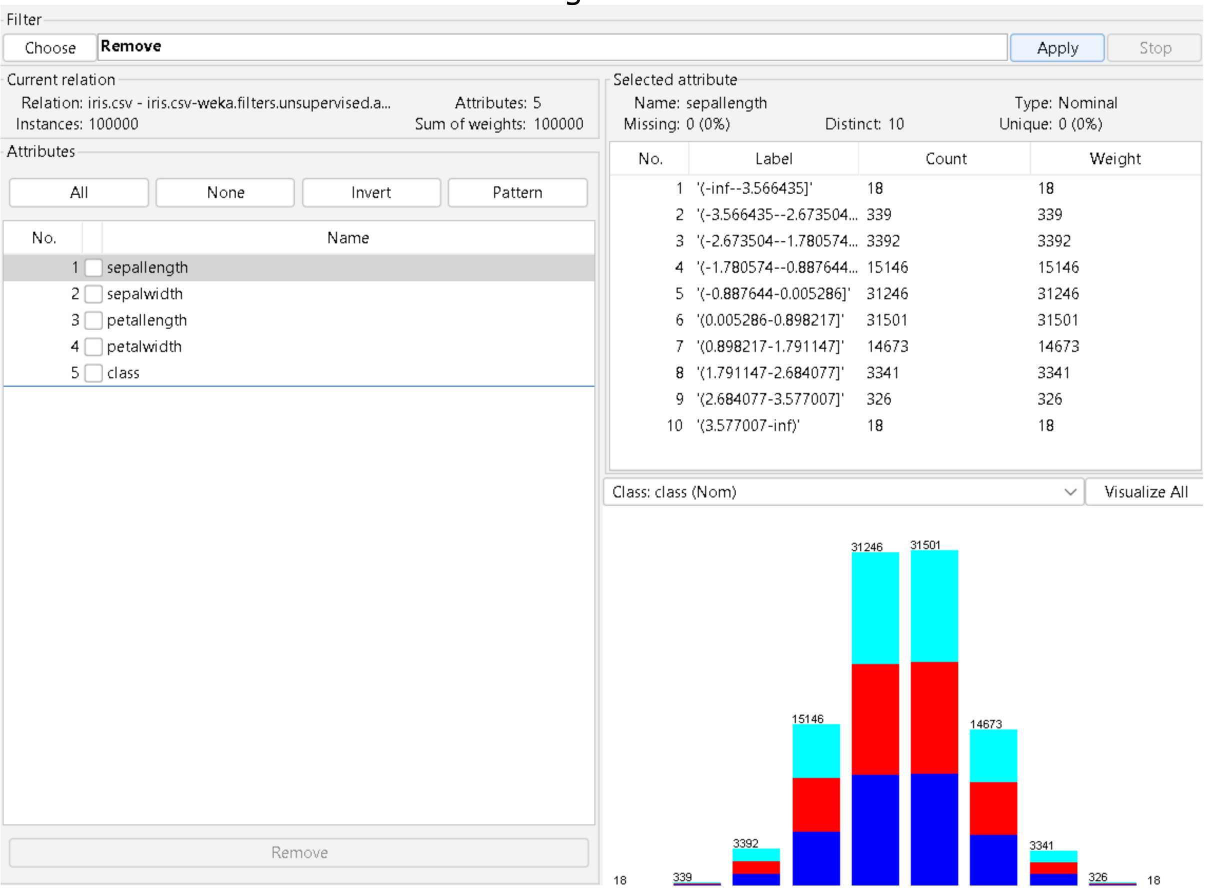
Result list (right-click for options)
16:51:42 - BestFirst + CfsSubsetEval

Attribute selection output

```
Evaluator:    weka.attributeSelection.CfsSubsetEval -P 1 -E 1
Search:       weka.attributeSelection.BestFirst -D 1 -N 5
Relation:     iris.csv - iris.csv-weka.filters.unsupervised.attribute.ReplaceMissingVa
Instances:    100000
Attributes:   5
              sepallength
              sepalwidth
              petallength
              petalwidth
              class
Evaluation mode:     evaluate on all training data



=== Attribute Selection on all input data ===

Search Method:
        Best first.
        Start set: no attributes
        Search direction: forward
        Stale search after 5 node expansions
        Total number of subsets evaluated: 11
        Merit of best subset found:     0

Attribute Subset Evaluator (supervised, Class (nominal): 5 class):
        CFS Subset Evaluator
        Including locally predictive attributes

Selected attributes: 2,3,4 : 3
                     sepalwidth
                     petallength
                     petalwidth
```

Status
OK

7. Removed unwanted attributes using the Remove filter.



Filter: Remove

Current relation

Relation: iris.csv - iris.csv-weka.filters.unsupervised.a... Attributes: 5

Instances: 100000 Sum of weights: 100000

Selected attribute

Name: sepallength Type: Nominal

Missing: 0 (0%) Distinct: 10 Unique: 0 (0%)

Attributes

No.	Name
1	□ sepallength
2	□ sepalwidth
3	□ petallength
4	□ petalwidth
5	□ class

No.	Label	Count	Weight
1	'(-inf--3.566435]'	18	18
2	'(-3.566435--2.673504...	339	339
3	'(-2.673504--1.780574...	3392	3392
4	'(-1.780574--0.887644...	15146	15146
5	'(-0.887644-0.005286]'	31246	31246
6	'(0.005286-0.898217]'	31501	31501
7	'(0.898217-1.791147]'	14673	14673
8	'(1.791147-2.684077]'	3341	3341
9	'(2.684077-3.577007]'	326	326
10	'(3.577007-inf)'	18	18

8. Observed the changes after each step and captured screenshots.

INPUT

The input is a dataset in **CSV format** containing multiple attributes with missing values, numeric values, and irrelevant attributes.
iris.csv

OUTPUT

The output is a **pre-processed dataset** in which:

- Missing values are handled
- Data is normalized
- Numeric attributes are discretized
- Important attributes are selected
- Unnecessary attributes are removed

The cleaned dataset is suitable for further data mining tasks.
iris.arff(preprocessed)

RESULT

Thus, data pre-processing tasks such as **missing value handling, normalization, discretization, attribute selection, and attribute removal** were successfully performed using the **WEKA tool**, resulting in a clean and well-structured dataset.

EXP.NO: 03	Implementation of association rule mining on data sets
DATE: 06.02.25	

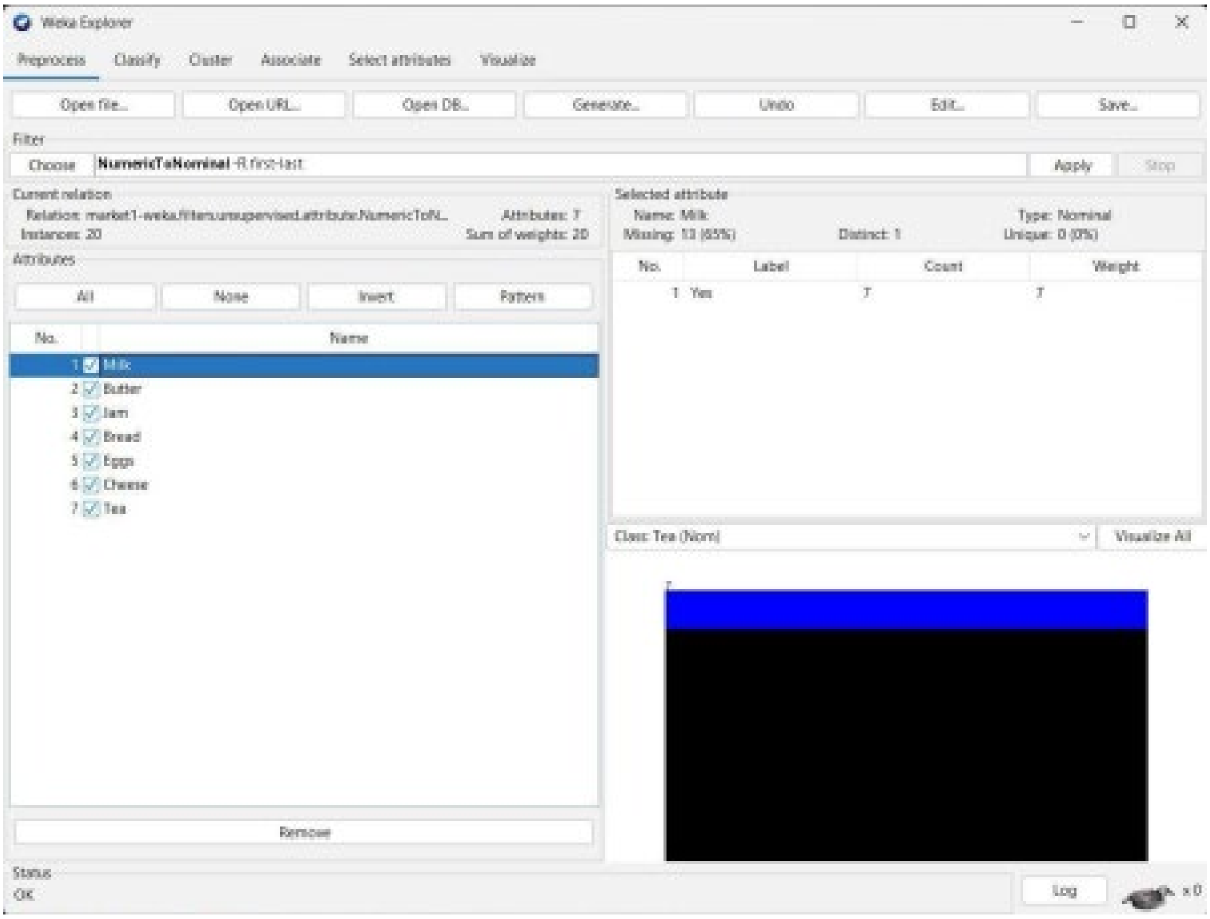
AIM

Implementation of association rule mining on data sets.

PROCEDURE

☦ Collect the transactional dataset containing items such as Milk, Bread, Butter, etc.
☦ Open the dataset in Excel and remove unnecessary empty columns. ☦ Replace all missing values (NaN) with **No**.
☦ Ensure that all attributes contain only **Yes/No** values.
☦ Save the cleaned file in **CSV format** (e.g., market_cleaned.csv). ☦ Open **WEKA GUI Chooser**.
☦ Click on **Explorer**.
☦ Go to the **Preprocess** tab.
☦ Click **Open File** and load the cleaned dataset.
☦ If required, convert attributes to nominal using: ○ Filter → Unsupervised → Attribute → NumericToNominal ☦ Click **Apply** to activate the filter.
☦ Go to the **Associate** tab.
☦ Click **Choose** and select **Apriori** algorithm.
☦ Click on the Apriori name to set parameters.
☦ Set:
○ Minimum Support = 0.2 ○ Minimum Confidence = 0.6 ○ Number of Rules = 10 ☦ Click **OK**.
☦ Click **Start** to run the algorithm.
☦ Observe the generated association rules in the output window. ☦ Note down support, confidence, lift, leverage, and conviction values. ☦ Analyze the strongest rules based on these measures.

INPUT



Filter: **NumericToNominal** -R first-last

Current relation
Relation: market1-weka.filters.unsupervised.attribute.NumericToN... Attributes: 7
Instances: 20 Sum of weights: 20

Attributes

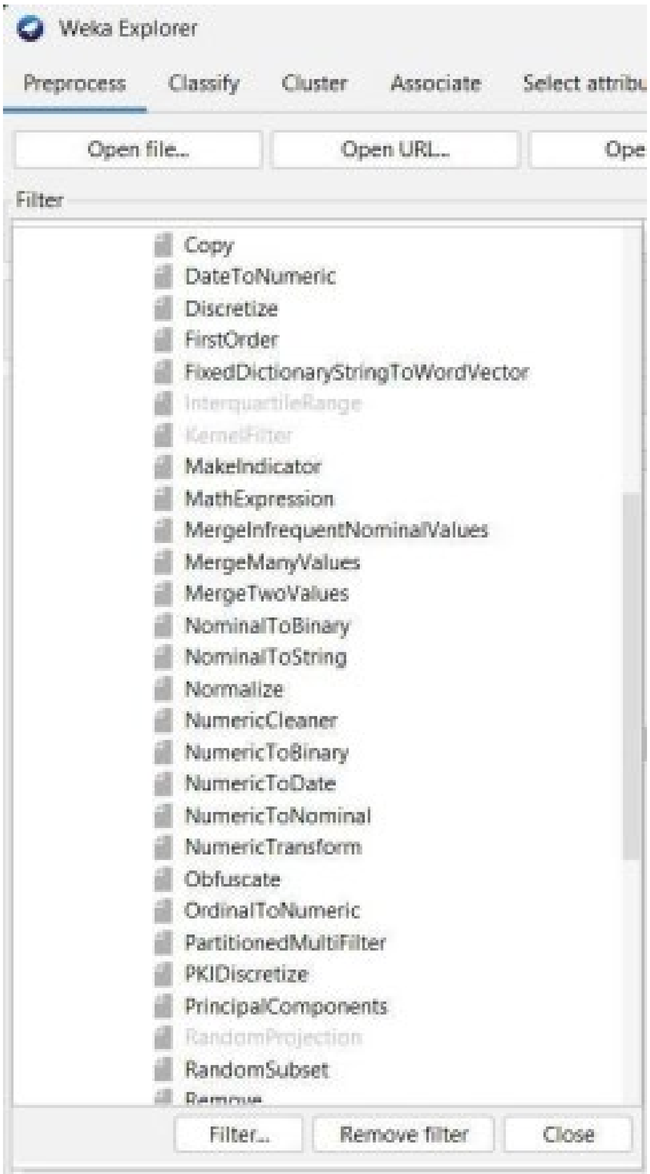
No.		Name
1	☑	Milk
2	☑	Butter
3	☑	Jam
4	☑	Bread
5	☑	Eggs
6	☑	Cheese
7	☑	Tea

Selected attribute
Name: Milk Type: Nominal
Missing: 13 (65%) Distinct: 1 Unique: 0 (0%)

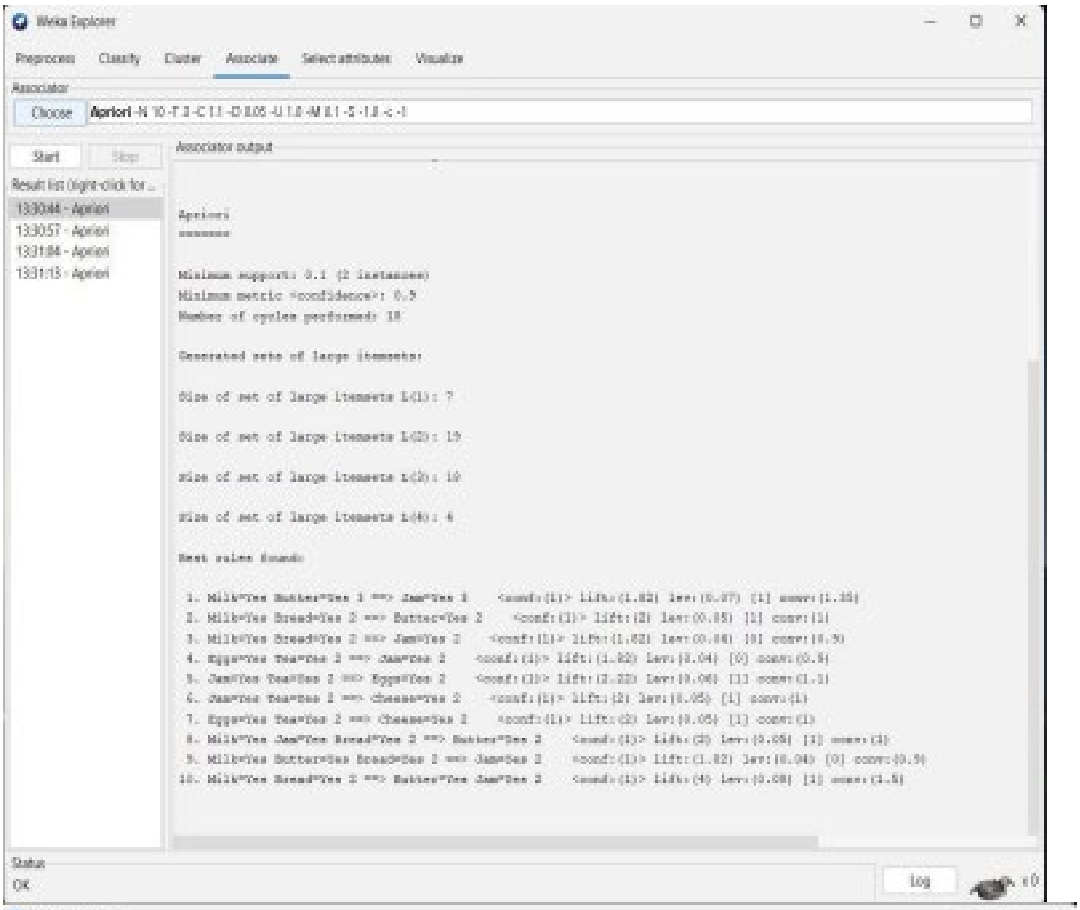
No.	Label	Count	Weight
1	Yes	7	7

Class: Tea (Nom)

Status
OK

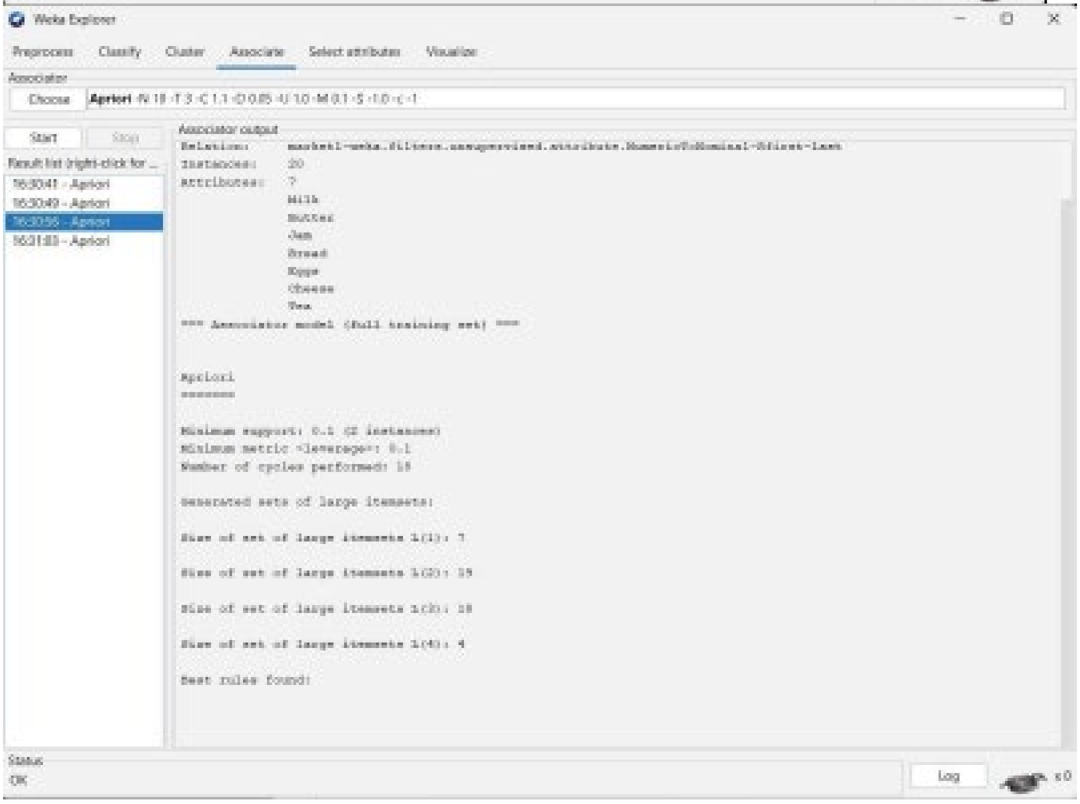

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attribu
Open file...
Open URL...
Ope
Filter
Copy
DateToNumeric
Discretize
FirstOrder
FixedDictionaryStringToWordVector
InterquartileRange
KernelFilter
MakeIndicator
MathExpression
MergeInfrequentNominalValues
MergeManyValues
MergeTwoValues
NominalToBinary
NominalToString
Normalize
NumericCleaner
NumericToBinary
NumericToDate
NumericToNominal
NumericTransform
Obfuscate
OrdinalToNumeric
PartitionedMultiFilter
PKIDiscretize
PrincipalComponents
RandomProjection
RandomSubset
Remove
Filter...
Remove filter
Close

OUTPUT



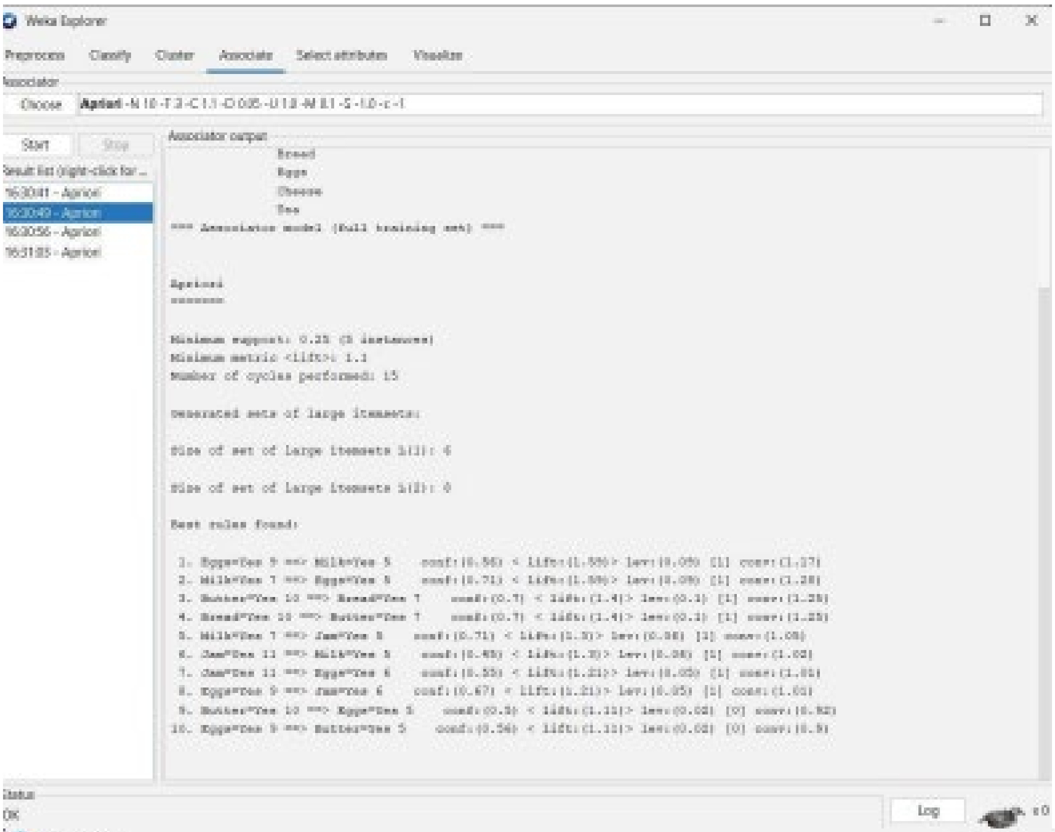
Apriori -N 10 -T 3 -C 1.1 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
Apriori
=======

Minimum support: 0.1 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 7

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 10

Size of set of large itemsets L(4): 4

Best rules found:

 1. Milk=Yes Butter=Yes 3 ==> Jam=Yes 3    <conf:(1)> lift:(1.82) lev:(0.07) [1] conv:(1.35)
 2. Milk=Yes Bread=Yes 2 ==> Butter=Yes 2    <conf:(1)> lift:(2) lev:(0.05) [1] conv:(1)
 3. Milk=Yes Bread=Yes 2 ==> Jam=Yes 2    <conf:(1)> lift:(1.82) lev:(0.08) [0] conv:(0.5)
 4. Eggs=Yes Tea=Yes 2 ==> Jam=Yes 2    <conf:(1)> lift:(1.82) lev:(0.04) [0] conv:(0.9)
 5. Jam=Yes Tea=Yes 2 ==> Eggs=Yes 2    <conf:(1)> lift:(2.22) lev:(0.06) [1] conv:(1.1)
 6. Jam=Yes Tea=Yes 2 ==> Cheese=Yes 2    <conf:(1)> lift:(2) lev:(0.05) [1] conv:(1)
 7. Eggs=Yes Tea=Yes 2 ==> Cheese=Yes 2    <conf:(1)> lift:(2) lev:(0.05) [1] conv:(1)
 8. Milk=Yes Jam=Yes Bread=Yes 2 ==> Butter=Yes 2    <conf:(1)> lift:(2) lev:(0.05) [1] conv:(1)
 9. Milk=Yes Butter=Yes Bread=Yes 2 ==> Jam=Yes 2    <conf:(1)> lift:(1.82) lev:(0.04) [0] conv:(0.5)
10. Milk=Yes Bread=Yes 2 ==> Butter=Yes Jam=Yes 2    <conf:(1)> lift:(4) lev:(0.08) [1] conv:(1.5)
```



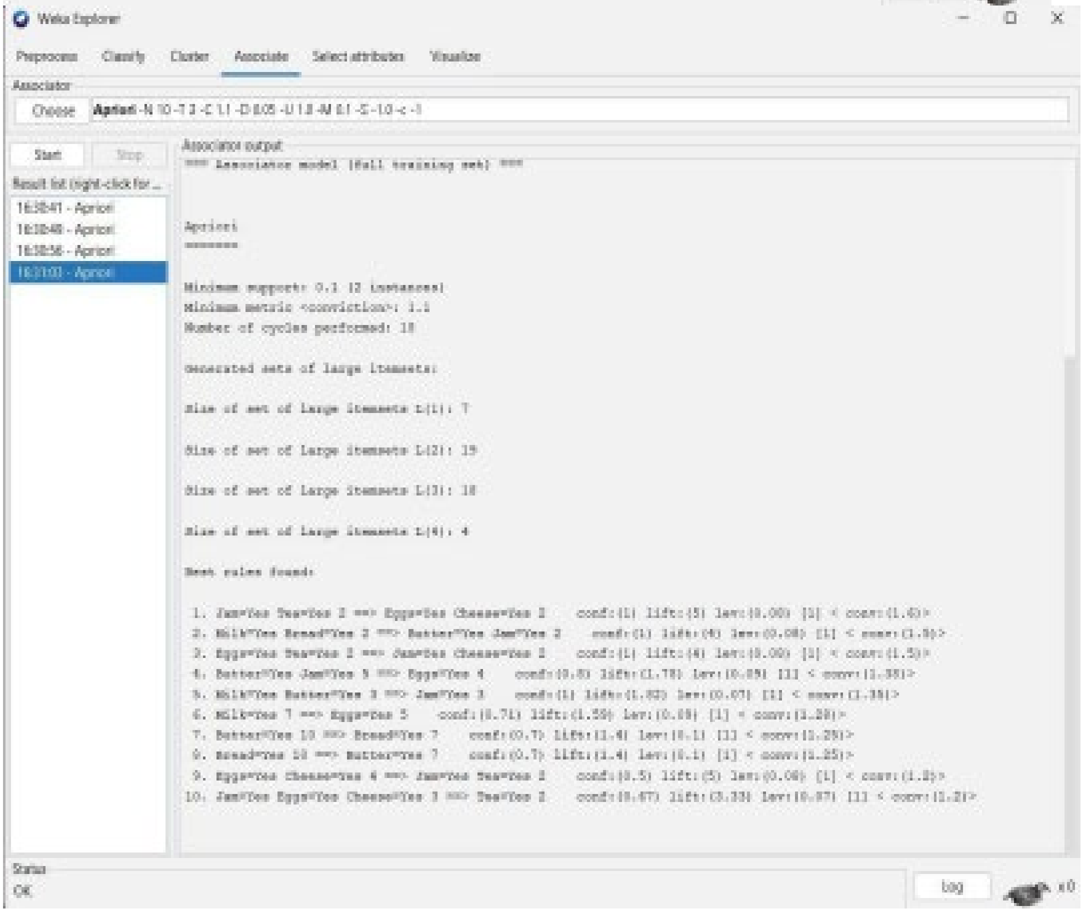
Apriori -N 18 -T 3 -C 1.1 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
Relation:     market1-weka.filters.unsupervised.attribute.NumericToNominal-Rfirst-last
Instances:    20
Attributes:   7
              Milk
              Butter
              Jam
              Bread
              Eggs
              Cheese
              Tea
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.1 (2 instances)
Minimum metric <leverage>: 0.1
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 7

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 10

Size of set of large itemsets L(4): 4

Best rules found:
```



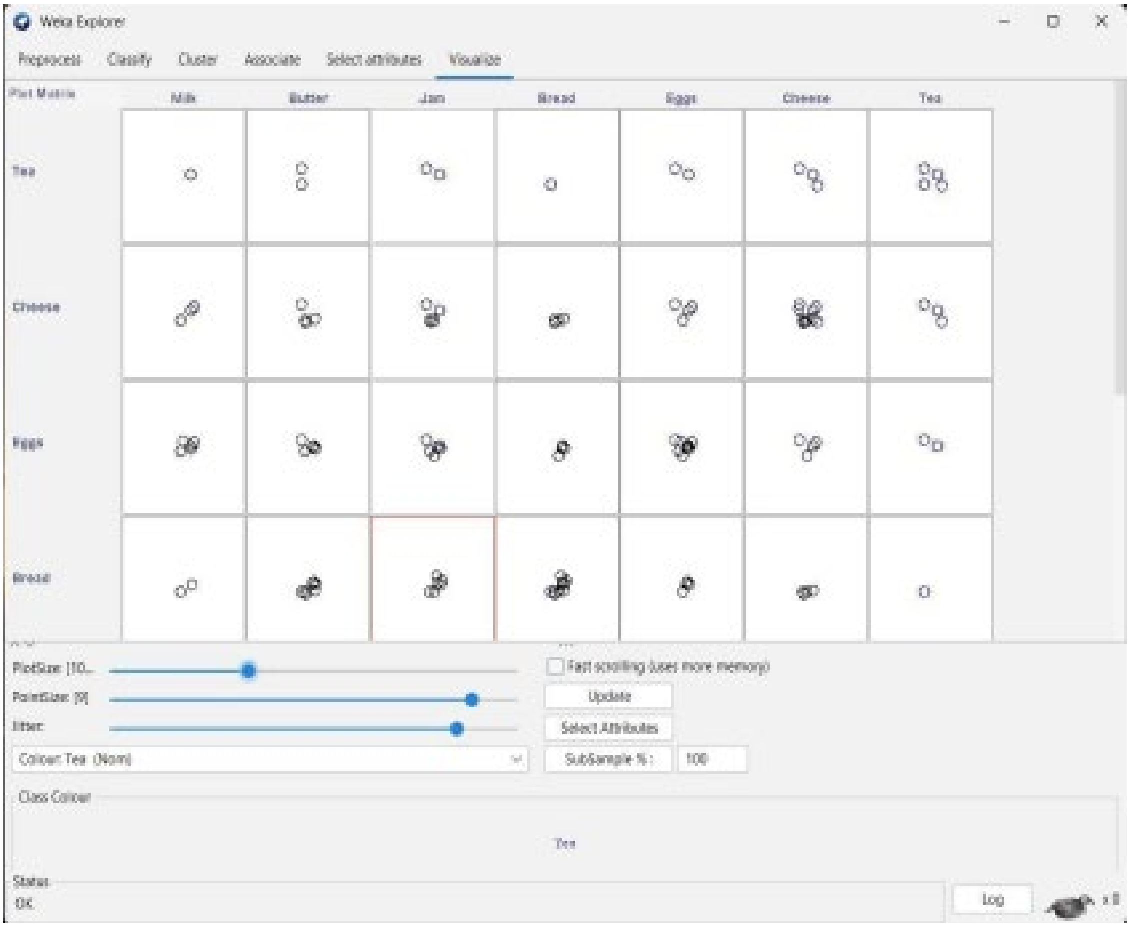
Apriori -N 10 -T 1 -C 1.1 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
              Bread
              Eggs
              Cheese
              Tea
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.25 (5 instances)
Minimum metric <lift>: 1.1
Number of cycles performed: 15

Generated sets of large itemsets:

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

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

Best rules found:

 1. Eggs=Yes 9 ==> Milk=Yes 5    conf:(0.56) < lift:(1.59)> lev:(0.09) [1] conv:(1.27)
 2. Milk=Yes 7 ==> Eggs=Yes 5    conf:(0.71) < lift:(1.59)> lev:(0.09) [1] conv:(1.28)
 3. Butter=Yes 10 ==> Bread=Yes 7    conf:(0.7) < lift:(1.4)> lev:(0.1) [1] conv:(1.25)
 4. Bread=Yes 10 ==> Butter=Yes 7    conf:(0.7) < lift:(1.4)> lev:(0.1) [1] conv:(1.25)
 5. Milk=Yes 7 ==> Jam=Yes 5    conf:(0.71) < lift:(1.3)> lev:(0.06) [1] conv:(1.05)
 6. Jam=Yes 11 ==> Milk=Yes 5    conf:(0.45) < lift:(1.3)> lev:(0.06) [1] conv:(1.02)
 7. Jam=Yes 11 ==> Eggs=Yes 6    conf:(0.55) < lift:(1.21)> lev:(0.05) [1] conv:(1.01)
 8. Eggs=Yes 9 ==> Jam=Yes 6    conf:(0.67) < lift:(1.21)> lev:(0.05) [1] conv:(1.01)
 9. Butter=Yes 10 ==> Eggs=Yes 5    conf:(0.5) < lift:(1.11)> lev:(0.02) [0] conv:(0.92)
10. Eggs=Yes 9 ==> Butter=Yes 5    conf:(0.56) < lift:(1.11)> lev:(0.02) [0] conv:(0.9)
```



Apriori -N 10 -T 3 -C 1.1 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.1 (2 instances)
Minimum metric <conviction>: 1.1
Number of cycles performed: 18

Generated sets of large itemsets:

Size of set of large itemsets L(1): 7

Size of set of large itemsets L(2): 19

Size of set of large itemsets L(3): 18

Size of set of large itemsets L(4): 4

Best rules found:

 1. Jam=Yes Tea=Yes 2 ==> Eggs=Yes Cheese=Yes 2    conf:(1) lift:(5) lev:(0.08) [1] < conv:(1.6)>
 2. Milk=Yes Bread=Yes 2 ==> Butter=Yes Jam=Yes 2    conf:(1) lift:(4) lev:(0.08) [1] < conv:(1.5)>
 3. Eggs=Yes Tea=Yes 2 ==> Jam=Yes Cheese=Yes 2    conf:(1) lift:(4) lev:(0.08) [1] < conv:(1.5)>
 4. Butter=Yes Jam=Yes 5 ==> Eggs=Yes 4    conf:(0.8) lift:(1.78) lev:(0.09) [1] < conv:(1.38)>
 5. Milk=Yes Butter=Yes 3 ==> Jam=Yes 3    conf:(1) lift:(1.82) lev:(0.07) [1] < conv:(1.35)>
 6. Milk=Yes 7 ==> Eggs=Yes 5    conf:(0.71) lift:(1.59) lev:(0.09) [1] < conv:(1.28)>
 7. Butter=Yes 10 ==> Bread=Yes 7    conf:(0.7) lift:(1.4) lev:(0.1) [1] < conv:(1.25)>
 8. Bread=Yes 10 ==> Butter=Yes 7    conf:(0.7) lift:(1.4) lev:(0.1) [1] < conv:(1.25)>
 9. Eggs=Yes Cheese=Yes 4 ==> Jam=Yes Tea=Yes 2    conf:(0.5) lift:(5) lev:(0.08) [1] < conv:(1.2)>
10. Jam=Yes Eggs=Yes Cheese=Yes 3 ==> Tea=Yes 2    conf:(0.67) lift:(3.33) lev:(0.07) [1] < conv:(1.2)>
```



PlotSize: [10...

PointSize: [9]

Jitter:

Colour: Tea (Nom)

SubSample % : 100

Status

OK

RESULT

After applying the Apriori algorithm on the transactional dataset using WEKA, several association rules were generated.
The discovered rules show relationships among different products such as Milk, Bread, Butter, Cheese, etc.

- The Apriori algorithm successfully generated frequent itemsets.
- Several strong association rules were obtained.
- Items like Milk, Bread, Butter, and Cheese showed strong relationships. • Rules with high support, confidence, and lift were considered important.
- The results help understand customer purchasing behavior.
- These rules can be used for product placement and marketing strategies.

EXP.NO: 04	**Implementation of classification techniques: Naïve Baye's, and SVM on data sets**
DATE:	

AIM

To implement and analyse Naïve Bayes and Support Vector Machine (SVM) classification techniques on a given dataset using WEKA.

PROCEDURE

1. Open **WEKA GUI Chooser**.
2. Click on **Explorer**.
3. Go to the **Preprocess** tab.
4. Click **Open file** and load the dataset (e.g., breast-cancer.arff).
5. Verify that the **class attribute** is correctly selected (usually the last attribute).
6. Switch to the **Classify** tab.

For Naïve Bayes:

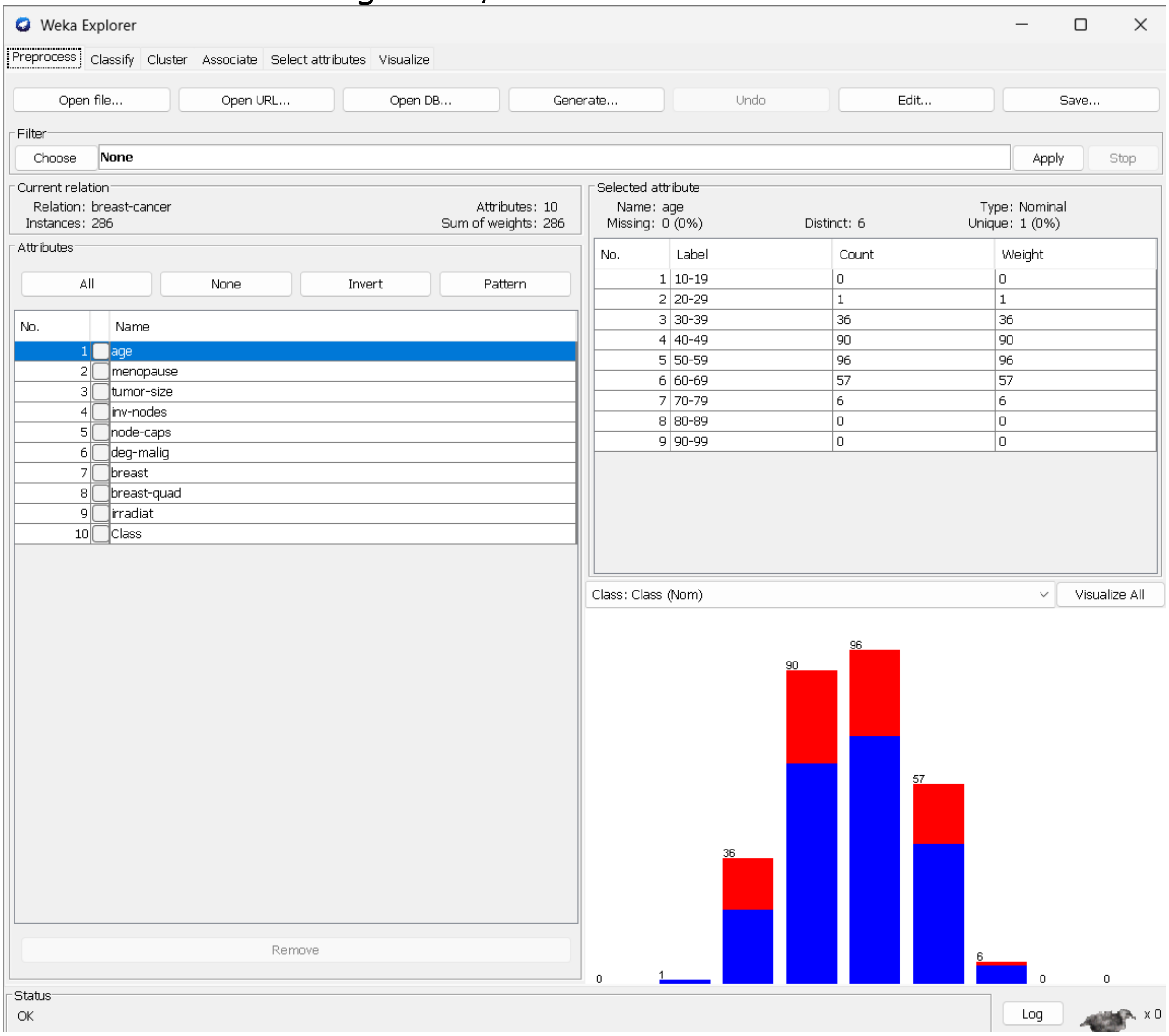
7. Click **Choose** → classifiers → bayes → NaiveBayes.
8. Select **10-fold cross validation** under Test options.
9. Click **Start**.

For SVM:

10. Click **Choose** → classifiers → functions → SMO.
11. (Optional) Select kernel type (Polynomial / RBF).
12. Choose **10-fold cross validation**.
13. Click **Start**.

INPUT

- Dataset: breast-cancer.arff
- Number of attributes: Depends on dataset
- Class attribute: Diagnosis / Class label



Filter: None

Current relation
Relation: breast-cancer
Attributes: 10
Instances: 286
Sum of weights: 286

Attributes

No.	Name
1	age
2	menopause
3	tumor-size
4	inv-nodes
5	node-caps
6	deg-malig
7	breast
8	breast-quad
9	irradiat
10	Class

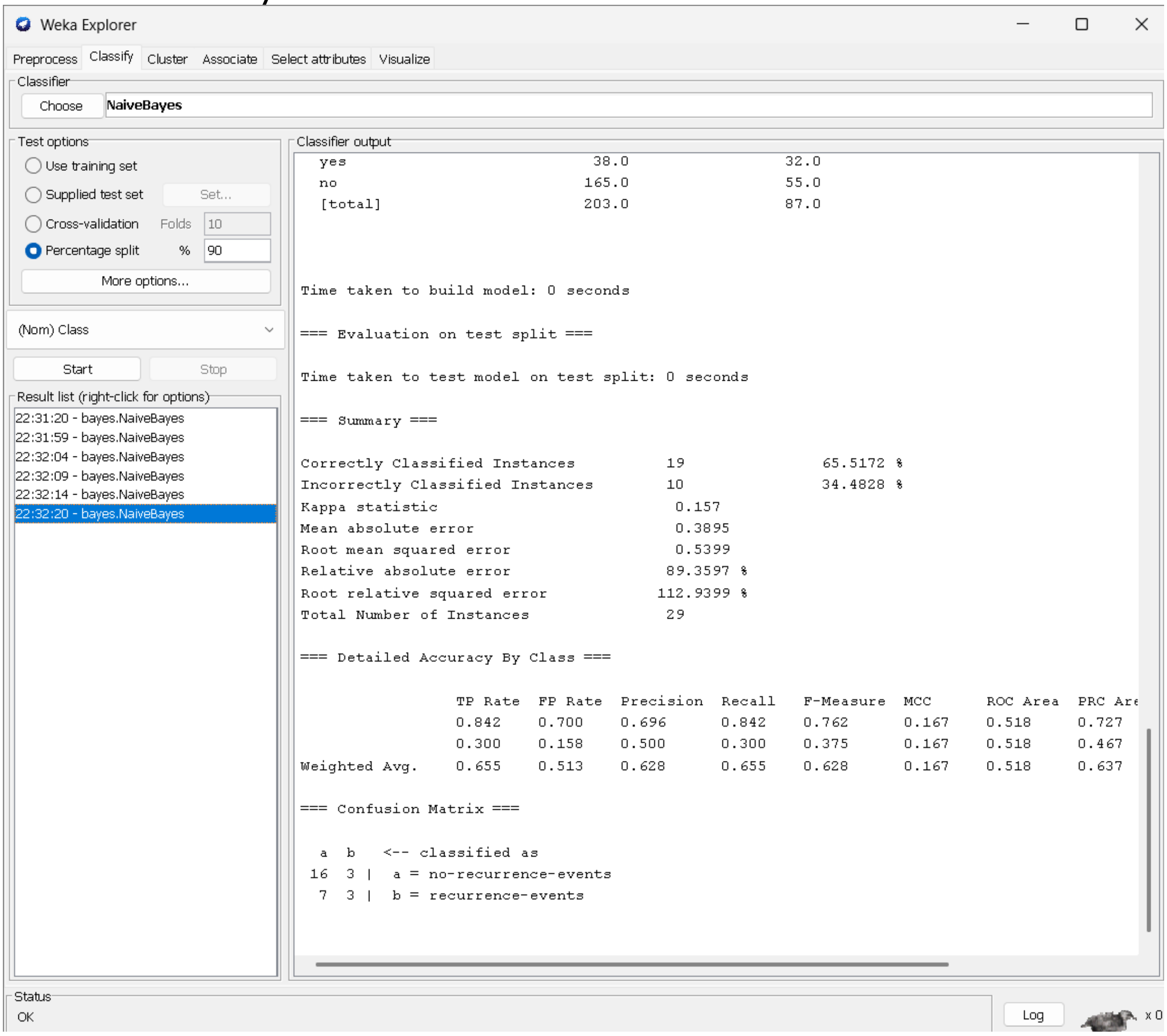
Selected attribute
Name: age
Type: Nominal
Missing: 0 (0%)
Distinct: 6
Unique: 1 (0%)

No.	Label	Count	Weight
1	10-19	0	0
2	20-29	1	1
3	30-39	36	36
4	40-49	90	90
5	50-59	96	96
6	60-69	57	57
7	70-79	6	6
8	80-89	0	0
9	90-99	0	0

Class: Class (Nom)

Status
OK

- Classification algorithms:
    - Naïve Bayes



NaiveBayes

○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 90

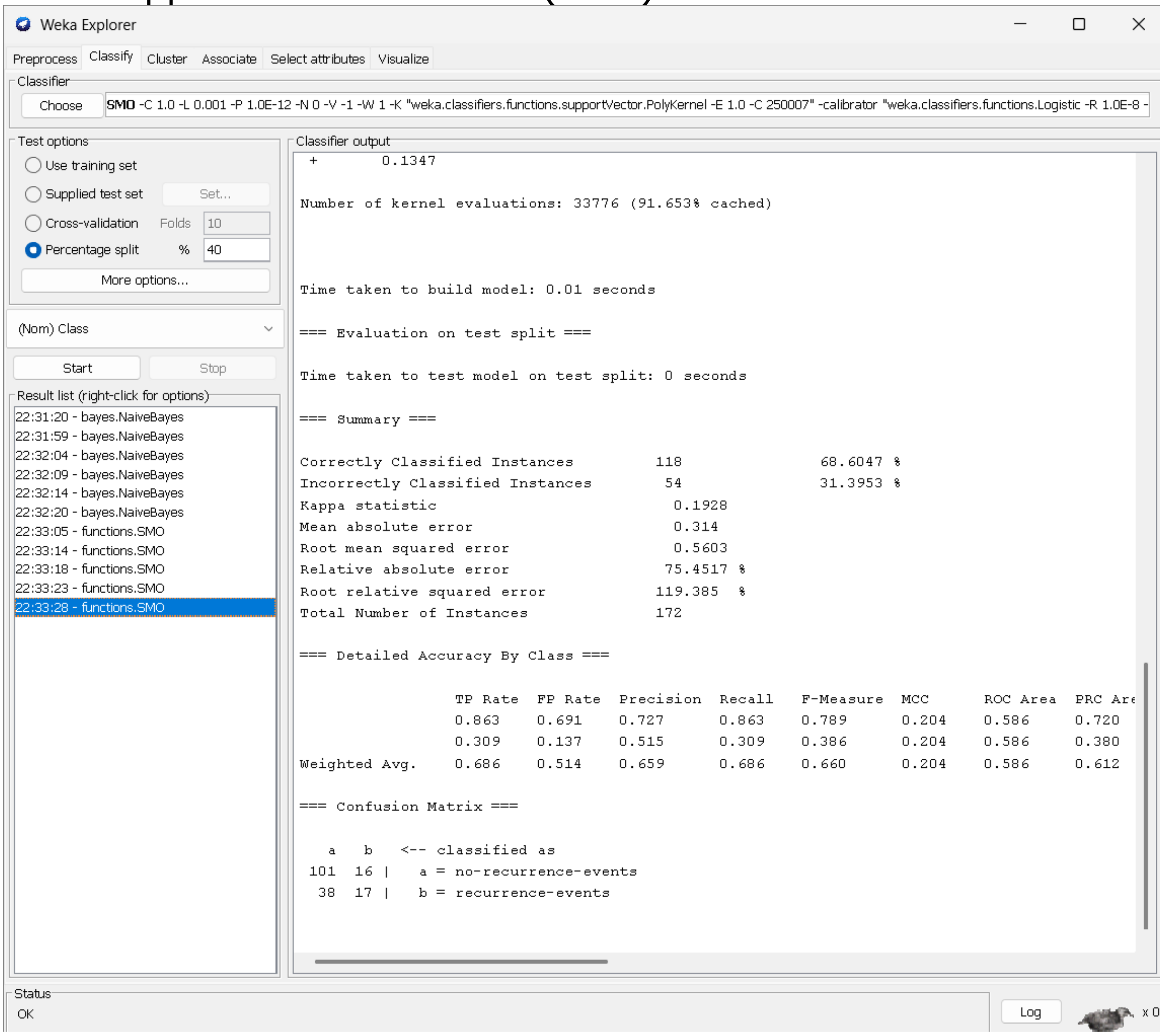
(Nom) Class

22:31:20 - bayes.NaiveBayes
22:31:59 - bayes.NaiveBayes
22:32:04 - bayes.NaiveBayes
22:32:09 - bayes.NaiveBayes
22:32:14 - bayes.NaiveBayes
22:32:20 - bayes.NaiveBayes

```
  yes                           38.0                 32.0
  no                           165.0                 55.0
  [total]                      203.0                 87.0



Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances          19               65.5172 %
Incorrectly Classified Instances        10               34.4828 %
Kappa statistic                          0.157
Mean absolute error                      0.3895
Root mean squared error                  0.5399
Relative absolute error                 89.3597 %
Root relative squared error            112.9399 %
Total Number of Instances               29

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Are
                 0.842    0.700    0.696      0.842    0.762      0.167    0.518     0.727
                 0.300    0.158    0.500      0.300    0.375      0.167    0.518     0.467
Weighted Avg.    0.655    0.513    0.628      0.655    0.628      0.167    0.518     0.637

=== Confusion Matrix ===

  a  b   <-- classified as
 16  3 |  a = no-recurrence-events
  7  3 |  b = recurrence-events
```

OK

- Support Vector Machine (SMO)



Classifier

SMO -C 1.0 -L 0.001 -P 1.0E-12 -N 0 -V -1 -W 1 -K "weka.classifiers.functions.supportVector.PolyKernel -E 1.0 -C 250007" -calibrator "weka.classifiers.functions.Logistic -R 1.0E-8 -

Test options

○ Use training set
○ Supplied test set
○ Cross-validation Folds 10
● Percentage split % 40

(Nom) Class

Result list (right-click for options)

```
22:31:20 - bayes.NaiveBayes
22:31:59 - bayes.NaiveBayes
22:32:04 - bayes.NaiveBayes
22:32:09 - bayes.NaiveBayes
22:32:14 - bayes.NaiveBayes
22:32:20 - bayes.NaiveBayes
22:33:05 - functions.SMO
22:33:14 - functions.SMO
22:33:18 - functions.SMO
22:33:23 - functions.SMO
22:33:28 - functions.SMO
```

Classifier output

```
 +       0.1347

Number of kernel evaluations: 33776 (91.653% cached)



Time taken to build model: 0.01 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances         118               68.6047 %
Incorrectly Classified Instances        54               31.3953 %
Kappa statistic                          0.1928
Mean absolute error                      0.314
Root mean squared error                  0.5603
Relative absolute error                 75.4517 %
Root relative squared error            119.385  %
Total Number of Instances              172

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Are
                 0.863    0.691    0.727      0.863    0.789      0.204    0.586     0.720
                 0.309    0.137    0.515      0.309    0.386      0.204    0.586     0.380
Weighted Avg.    0.686    0.514    0.659      0.686    0.660      0.204    0.586     0.612

=== Confusion Matrix ===

   a   b   <-- classified as
 101  16 |   a = no-recurrence-events
  38  17 |   b = recurrence-events
```

Status

OK

OUTPUT

- Classification accuracy
- Confusion matrix
- Precision, Recall, F-measure
- Error rate
- ROC area (for SVM)

RESULT

The Naïve Bayes and Support Vector Machine classifiers were successfully implemented using WEKA.

It was observed that:

- Naïve Bayes produced faster results with reasonable accuracy.
- SVM (SMO) achieved higher classification accuracy with lower error rate.

Thus, SVM performs better for complex datasets, while Naïve Bayes is efficient for quick predictions.