PONDICHERRY UNIVERSITY

DEPARTMENT OF BIOINFORMATICS

PONDICHERRY-605014

M. TECH COMPUTATIONAL BIOLOGY

LAB – DATA MINING AND DATA WAREHOUSING

SUBJECT CODE: CBIO 753

Name: Pranjal Paul

Reg No.- 23310012

Semester -III

Session – 2023-2025

CERTIFICATE

Certified that this is a bonafide record of the work done by **Pranjal Paul**, Reg. No: **23310012** of 2nd year M.Tech. Computational Biology, in the lab CBIO753 – DATA MINING AND DATA WAREHOUSING during the academic year 2024-2025**.**

Faculty-in-charge	The Head
Department of Bioinformatics	Department of Bioinformatics
Pondicherry University	Pondicherry University

Submitted for the M.Tech. Practical Examination held on 16/11/2024 at Department of Bioinformatics, Pondicherry University, Pondicherry- 605014.

INDEX

EXERCISE 1

DATA PRE-PROCESSING IN WEKA

AIM:

To pre-process the raw data in Weka v.3.8.5 tool.

INTRODUCTION

Data preprocessing is the data mining technique used to transform the raw data into a useful information. The following are the steps involved in data pre-processing. They are,

- Data cleaning,
- Data transformation,
- Data reduction.

Data cleaning:

This step is required to eliminate the missing values in raw data by filling the missing values using attribute mean or the most probable value and to eliminate the noisy data by performing binning method, regression or clustering.

Data transformation:

This step transforms the data in suitable forms for data mining process. It involves normalization, attribute selection, discretization and concept hierarchy generation.

Data reduction:

This step is used to handle huge amount of data. It aims to increase storage efficiency and to reduce data storage and analysis costs. It includes data cube aggregation, attribute subset selection, numerosity reduction and dimensionality reduction.

PROCEDURE

Loading a dataset:

- Open Weka v.3.8.5 tool and enter “Explorer”
- Open file → select a dataset (airline.arff) → Open

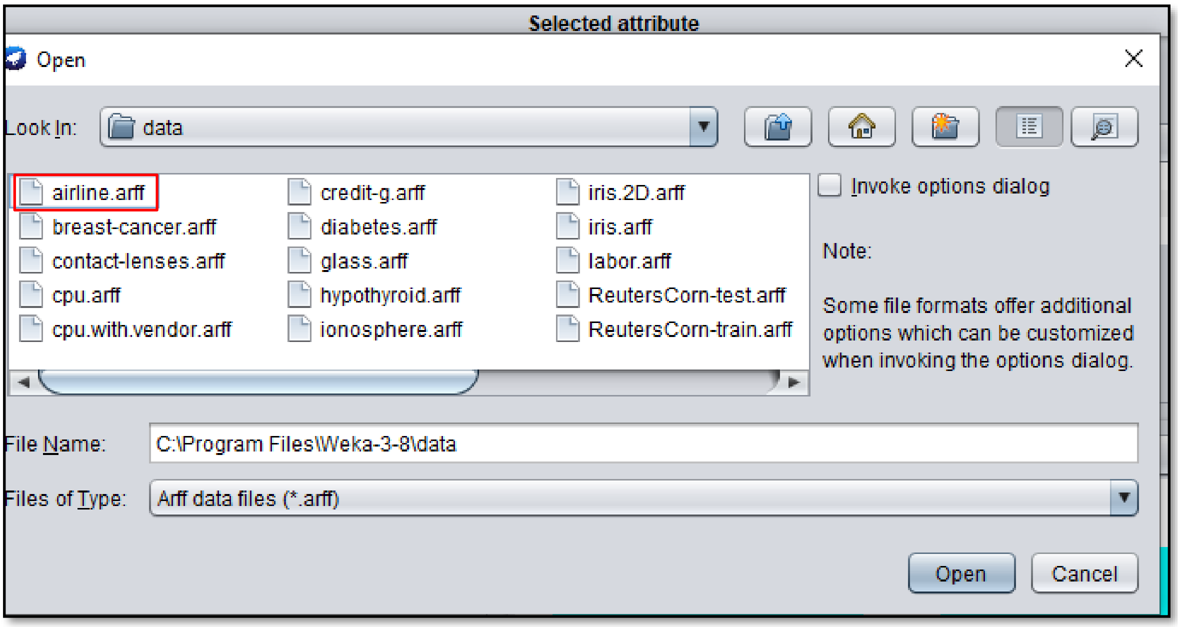


Fig. 1.1: Loading a dataset in Weka v.3.8.5 tool

Replacing Missing values:

- Dataset used: airline.arff.
- Analyze the missing value data in the dataset.
- Pre-process filter →Unsupervised →attributes →ReplaceMissingValues →Apply

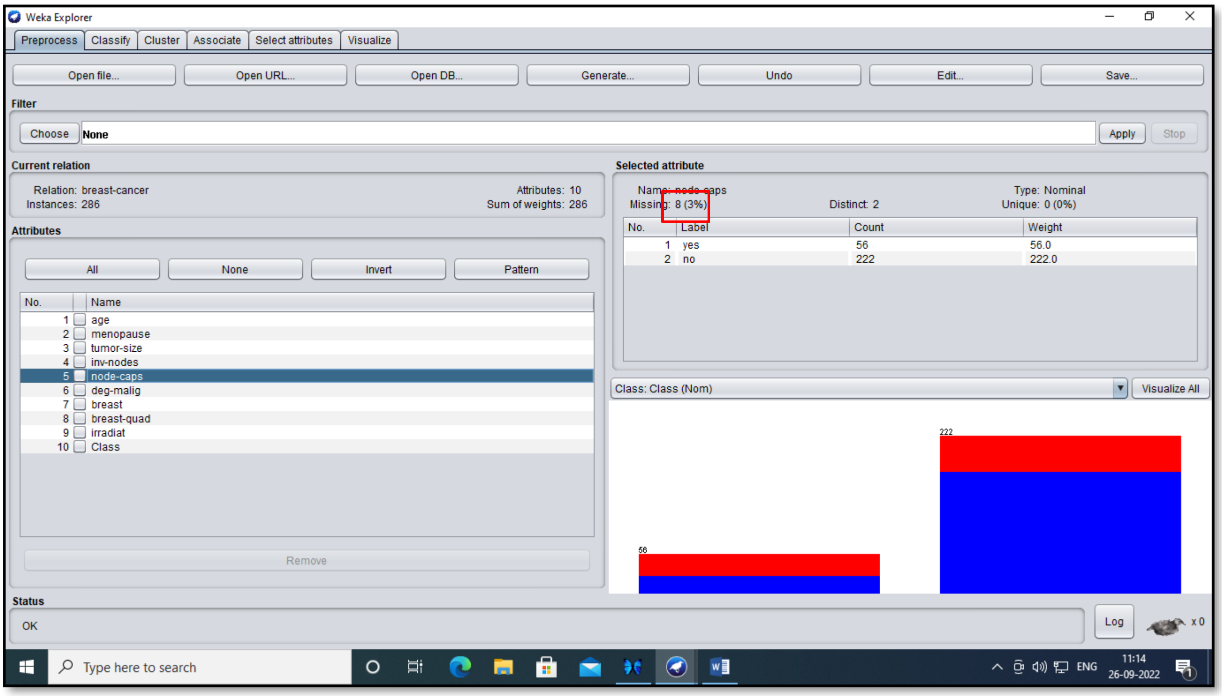


Fig. 1.2: airline.arff dataset was loaded and filter

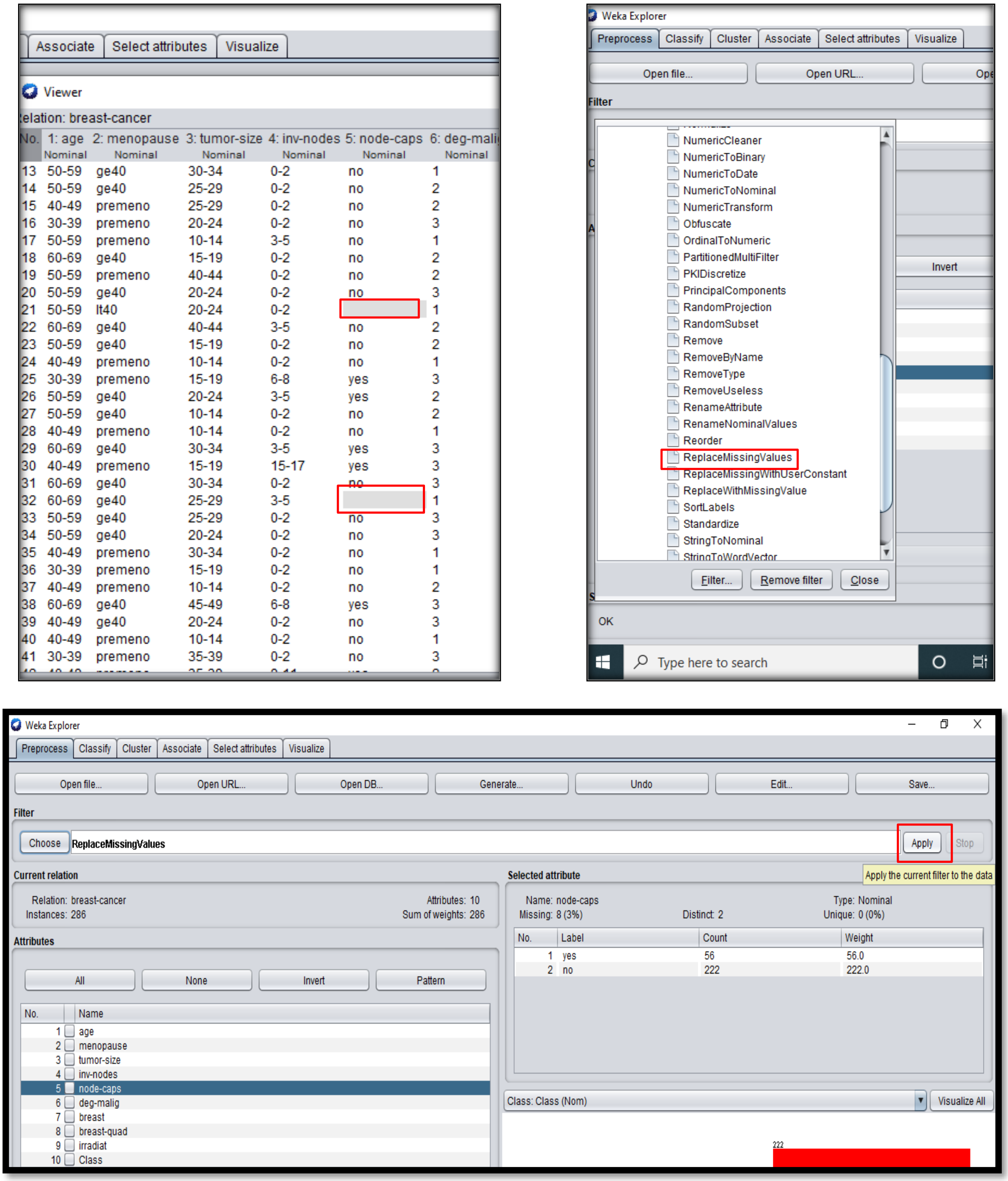
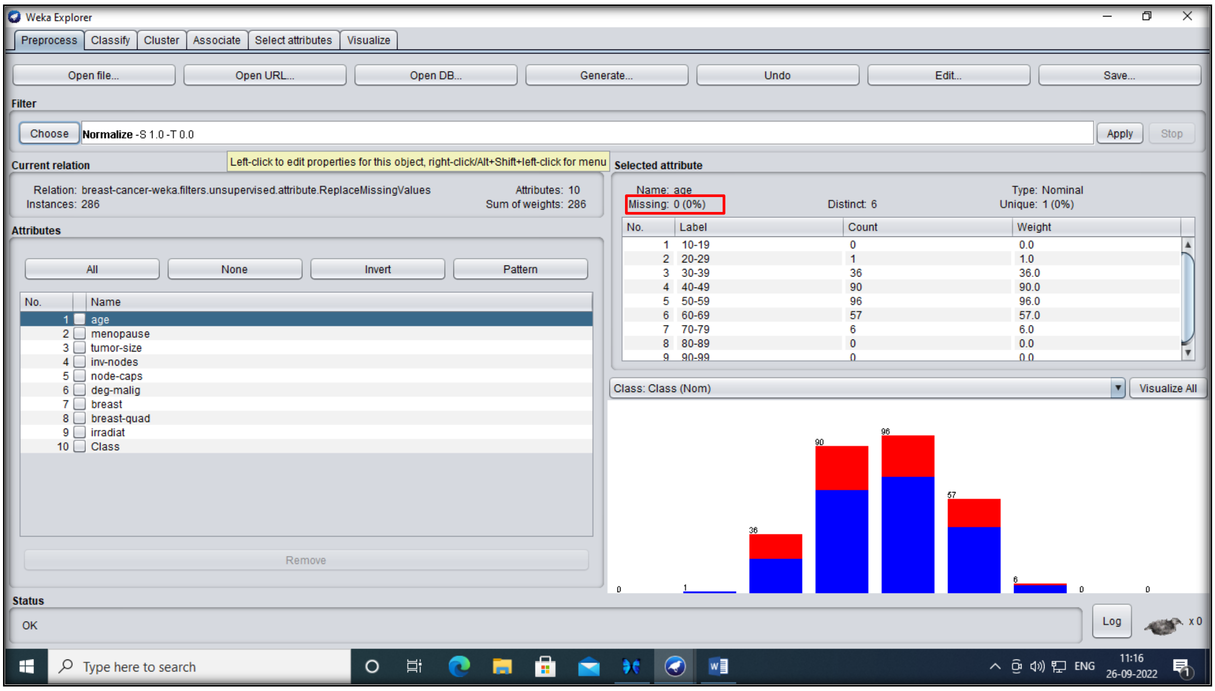
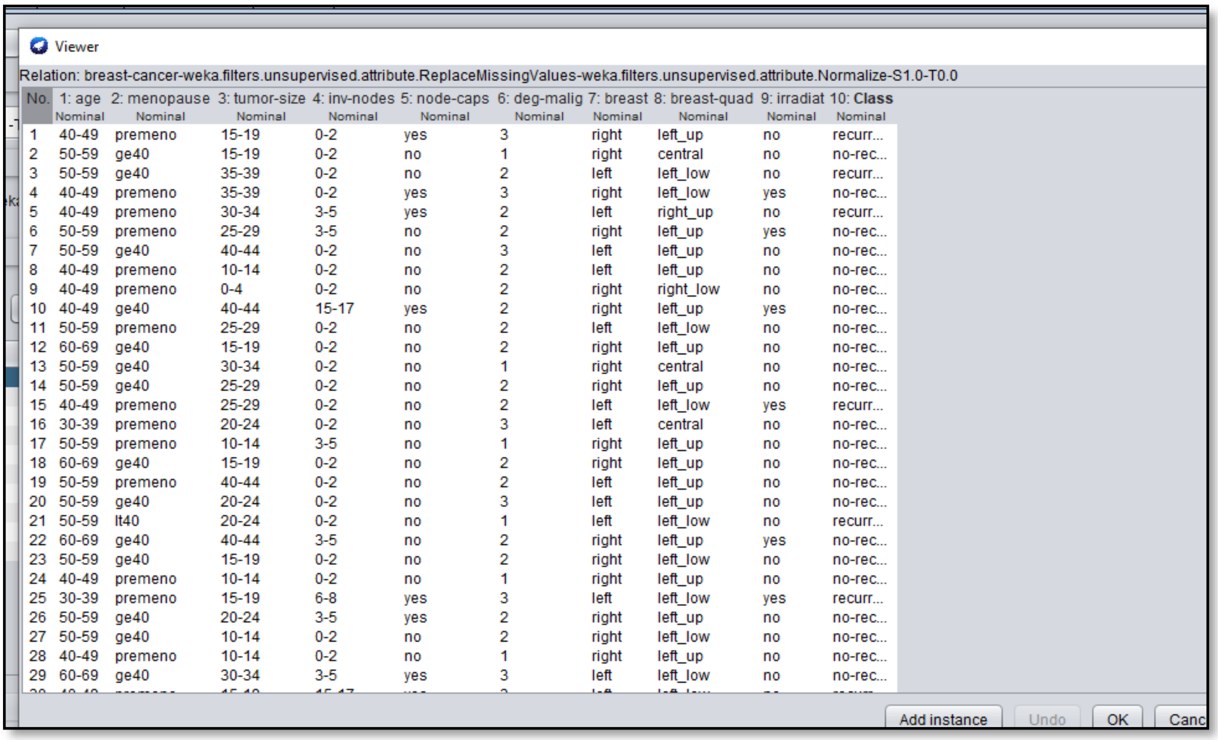


Fig 1.3: Replace missing value was applied





No.	1: age Nominal	2: menopause Nominal	3: tumor-size Nominal	4: inv-nodes Nominal	5: node-caps Nominal	6: deg-malig Nominal	7: breast Nominal	8: breast-quad Nominal	9: irradiat Nominal	10: Class Nominal
1	40-49	premeno	15-19	0-2	yes	3	right	left_up	no	recurr...
2	50-59	ge40	15-19	0-2	no	1	right	central	no	no-rec...
3	50-59	ge40	35-39	0-2	no	2	left	left_low	no	recurr...
4	40-49	premeno	35-39	0-2	yes	3	right	left_low	yes	no-rec...
5	40-49	premeno	30-34	3-5	yes	2	left	right_up	no	recurr...
6	50-59	premeno	25-29	3-5	no	2	right	left_up	yes	no-rec...
7	50-59	ge40	40-44	0-2	no	3	left	left_up	no	no-rec...
8	40-49	premeno	10-14	0-2	no	2	left	left_up	no	no-rec...
9	40-49	premeno	0-4	0-2	no	2	right	right_low	no	no-rec...
10	40-49	ge40	40-44	15-17	yes	2	right	left_up	yes	no-rec...
11	50-59	premeno	25-29	0-2	no	2	left	left_low	no	no-rec...
12	60-69	ge40	15-19	0-2	no	2	right	left_up	no	no-rec...
13	50-59	ge40	30-34	0-2	no	1	right	central	no	no-rec...
14	50-59	ge40	25-29	0-2	no	2	right	left_up	no	no-rec...
15	40-49	premeno	25-29	0-2	no	2	left	left_low	yes	recurr...
16	30-39	premeno	20-24	0-2	no	3	left	central	no	no-rec...
17	50-59	premeno	10-14	3-5	no	1	right	left_up	no	no-rec...
18	60-69	ge40	15-19	0-2	no	2	right	left_up	no	no-rec...
19	50-59	premeno	40-44	0-2	no	2	left	left_up	no	no-rec...
20	50-59	ge40	20-24	0-2	no	3	left	left_up	no	no-rec...
21	50-59	lt40	20-24	0-2	no	1	left	left_low	no	recurr...
22	60-69	ge40	40-44	3-5	no	2	right	left_up	yes	no-rec...
23	50-59	ge40	15-19	0-2	no	2	right	left_low	no	no-rec...
24	40-49	premeno	10-14	0-2	no	1	right	left_up	no	no-rec...
25	30-39	premeno	15-19	6-8	yes	3	left	left_low	yes	recurr...
26	50-59	ge40	20-24	3-5	yes	2	right	left_up	no	no-rec...
27	50-59	ge40	10-14	0-2	no	2	right	left_low	no	no-rec...
28	40-49	premeno	10-14	0-2	no	1	right	left_up	no	no-rec...
29	60-69	ge40	30-34	3-5	yes	3	left	left_low	no	no-rec...

Fig. 1.4: Missing values were removed

Normalize:

- Dataset used: weather.numeric.arff
- Pre-process → filter → Unsupervised → attributes → Normalize → Apply

Viewer

Relation: weather

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: play Nominal
1	sunny	85.0	85.0	FALSE	no
2	sunny	80.0	90.0	TRUE	no
3	overcast	83.0	86.0	FALSE	yes
4	rainy	70.0	96.0	FALSE	yes
5	rainy	68.0	80.0	FALSE	yes
6	rainy	65.0	70.0	TRUE	no
7	overcast	64.0	65.0	TRUE	yes
8	sunny	72.0	95.0	FALSE	no
9	sunny	69.0	70.0	FALSE	yes
10	rainy	75.0	80.0	FALSE	yes
11	sunny	75.0	70.0	TRUE	yes
12	overcast	72.0	90.0	TRUE	yes
13	overcast	81.0	75.0	FALSE	yes
14	rainy	71.0	91.0	TRUE	no

Fig. 1.5: Attribute - Temperature is to be normalized

Viewer

Relation: weather-weka.filters.unsupervised.attribute.Normali

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: play Nominal
1	sunny	1.0	0.64516...	FALSE	no
2	sunny	0.76190476...	0.80645...	TRUE	no
3	overcast	0.90476190...	0.67741...	FALSE	yes
4	rainy	0.28571428...	1.0	FALSE	yes
5	rainy	0.19047619...	0.48387...	FALSE	yes
6	rainy	0.04761904...	0.16129...	TRUE	no
7	overcast	0.0	0.0	TRUE	yes
8	sunny	0.38095238...	0.96774...	FALSE	no
9	sunny	0.23809523...	0.16129...	FALSE	yes
10	rainy	0.52380952...	0.48387...	FALSE	yes
11	sunny	0.52380952...	0.16129...	TRUE	yes
12	overcast	0.38095238...	0.80645...	TRUE	yes
13	overcast	0.80952380...	0.32258...	FALSE	yes
14	rainy	0.33333333...	0.83870...	TRUE	no

Fig. 1.6: Attribute - Temperature was normalized

Converting nominal to binary:

- Dataset used: weather.numeric.arff
- Pre-process→ filter → Unsupervised → attributes→ NominalToBinary → Apply

No.	1: outlook Nominal	2: temperature Numeric	3: humidity Numeric	4: windy Nominal	5: play Nominal
1	sunny	1.0	0.64516...	FALSE	no
2	sunny	0.76190476...	0.80645...	TRUE	no
3	overcast	0.90476190...	0.67741...	FALSE	yes
4	rainy	0.28571428...	1.0	FALSE	yes
5	rainy	0.19047619...	0.48387...	FALSE	yes
6	rainy	0.04761904...	0.16129...	TRUE	no
7	overcast	0.0	0.0	TRUE	yes
8	sunny	0.38095238...	0.96774...	FALSE	no
9	sunny	0.23809523...	0.16129...	FALSE	yes
10	rainy	0.52380952...	0.48387...	FALSE	yes
11	sunny	0.52380952...	0.16129...	TRUE	yes
12	overcast	0.38095238...	0.80645...	TRUE	yes
13	overcast	0.80952380...	0.32258...	FALSE	yes
14	rainy	0.33333333	0.83870	TRUE	no

Fig. 1.7: Attribute – windy is to be converted from nominal to binary values

Viewer

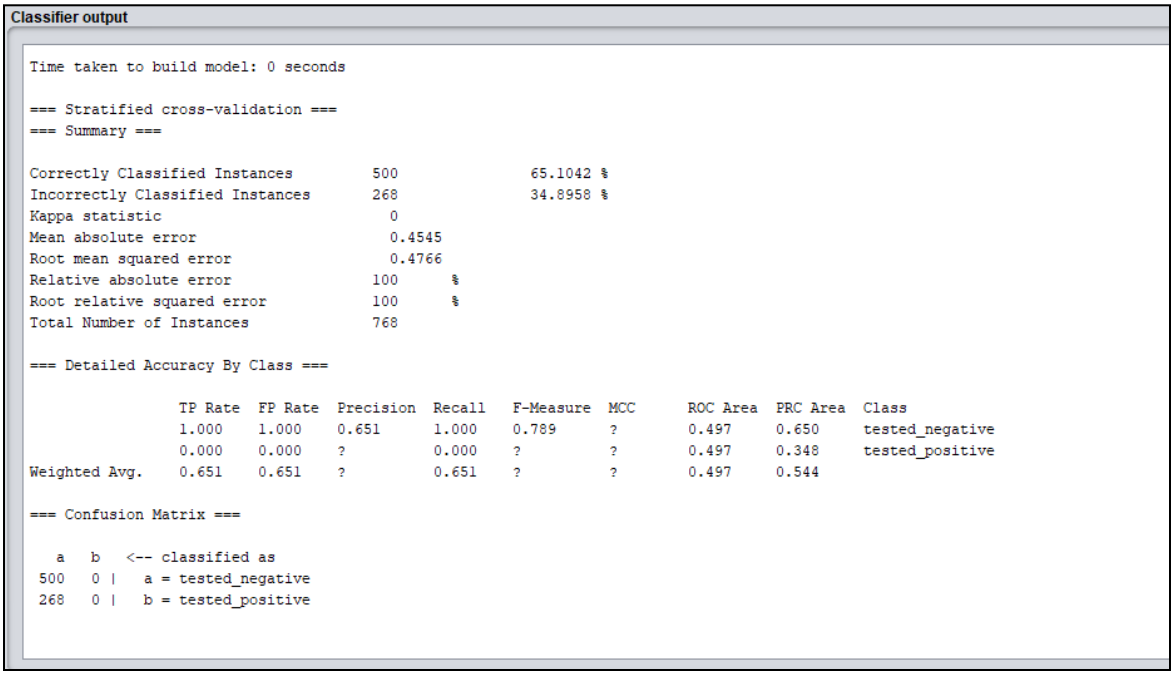
Relation: weather-weka.filters.unsupervised.attribute.Normalize-S1.0-T0.0-weka.filters.unsupervised.attribute.NominalToBinary-Rfirst-last

No.	1: outlook=sunny Numeric	2: outlook=overcast Numeric	3: outlook=rainy Numeric	4: temperature Numeric	5: humidity Numeric	6: windy=FALSE Numeric	7: play Nominal
1	1.0	0.0	0.0	1.0	0.64516...	1.0	no
2	1.0	0.0	0.0	0.76190476...	0.80645...	0.0	no
3	0.0	1.0	0.0	0.90476190...	0.67741...	1.0	yes
4	0.0	0.0	1.0	0.28571428...	1.0	1.0	yes
5	0.0	0.0	1.0	0.19047619...	0.48387...	1.0	yes
6	0.0	0.0	1.0	0.04761904...	0.16129...	0.0	no
7	0.0	1.0	0.0	0.0	0.0	0.0	yes
8	1.0	0.0	0.0	0.38095238...	0.96774...	1.0	no
9	1.0	0.0	0.0	0.23809523...	0.16129...	1.0	yes
10	0.0	0.0	1.0	0.52380952...	0.48387...	1.0	yes
11	1.0	0.0	0.0	0.52380952...	0.16129...	0.0	yes
12	0.0	1.0	0.0	0.38095238...	0.80645...	0.0	yes
13	0.0	1.0	0.0	0.80952380...	0.32258...	1.0	yes
14	0.0	0.0	1.0	0.33333333...	0.83870...	0.0	no

Fig. 1.8: Attribute – windy was converted from nominal to binary values

Correcting the Misclassify:

- Dataset used: diabetes.arff
- Classify start → Analyze Incorrectly classified Instances
- Pre-process → filter → Unsupervised → instance → RemoveMisclassified →Apply
- Classify → start → Analyze Incorrectly classified Instances



```
Classifier output

Time taken to build model: 0 seconds

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

Correctly Classified Instances         500               65.1042 %
Incorrectly Classified Instances       268               34.8958 %
Kappa statistic                          0
Mean absolute error                      0.4545
Root mean squared error                  0.4766
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              768

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    1.000    0.651      1.000    0.789      ?        0.497     0.650     tested_negative
                 0.000    0.000    ?          0.000    ?          ?        0.497     0.348     tested_positive
Weighted Avg.    0.651    0.651    ?          0.651    ?          ?        0.497     0.544

=== Confusion Matrix ===

   a   b   <-- classified as
 500   0 |   a = tested_negative
 268   0 |   b = tested_positive
```

Fig. 1.9: Incorrectly classified Instances were found to be 34.89%

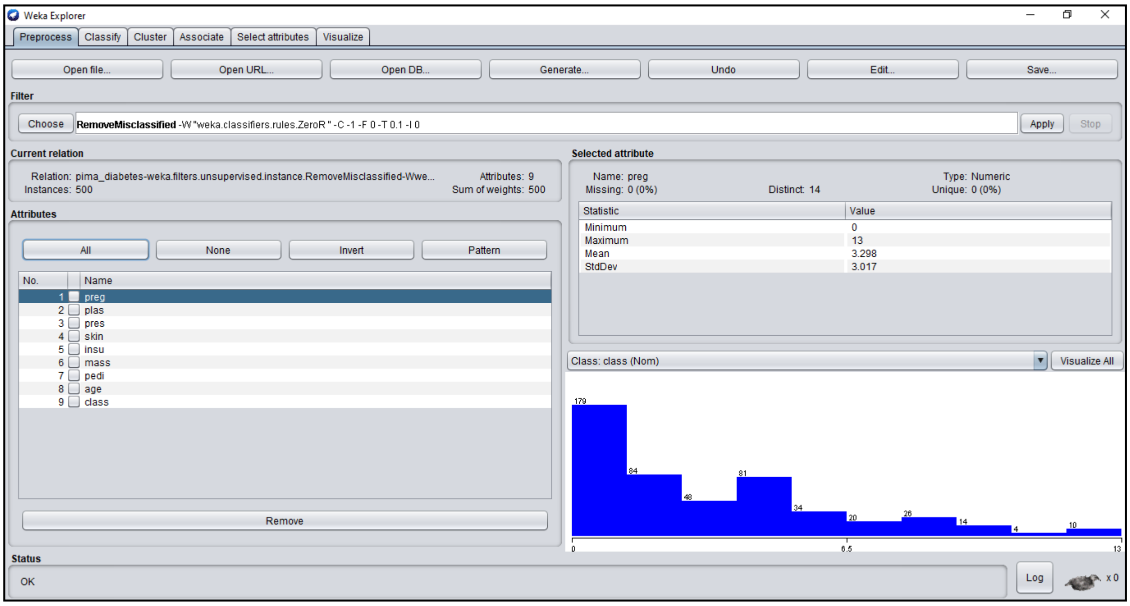


Fig. 1.10: RemoveMisclassified filter was applied

```
Classifier output

Time taken to build model: 0 seconds

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

Correctly Classified Instances         500              100      %
Incorrectly Classified Instances         0                0      %
Kappa statistic                          1
Mean absolute error                      0.0022
Root mean squared error                  0.0022
Relative absolute error                100      %
Root relative squared error            100      %
Total Number of Instances              500

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 1.000    ?        1.000      1.000    1.000      ?        ?         1.000     tested_negative
                 ?        0.000    ?          ?        ?          ?        ?         ?         tested_positive
Weighted Avg.    1.000    ?        1.000      1.000    1.000      ?        ?         1.000

=== Confusion Matrix ===

   a   b   <-- classified as
 500   0 |   a = tested_negative
   0   0 |   b = tested_positive
```

Fig. 1.11: Incorrectly classified instances were found to 0% - Misclassified removed.

Discretization:

- Dataset used: diabetes.arff
- Pre-process → filter → Unsupervised → attributes → Discretize →Apply

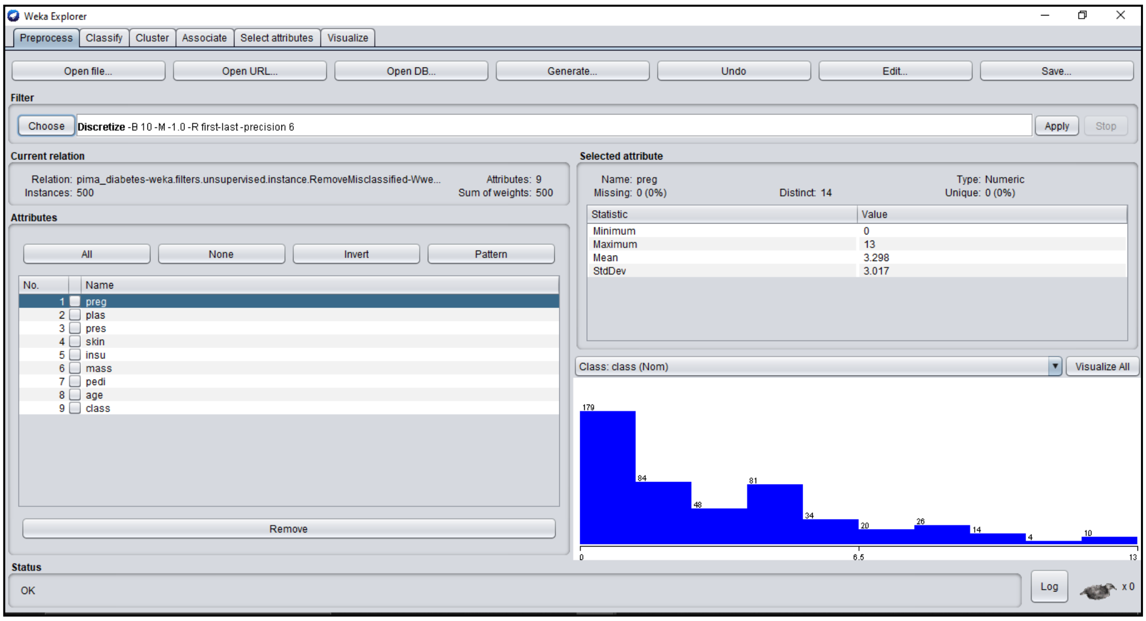


Fig. 1.12: Discretize filter was applied

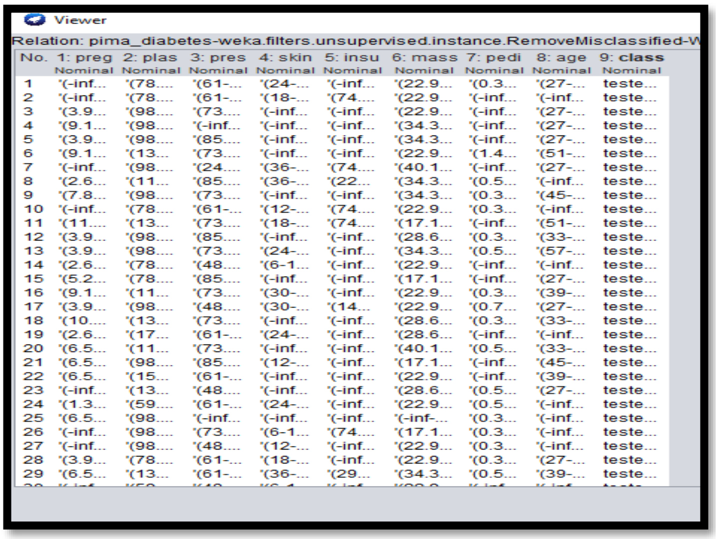


Fig. 1.13: Attributes' instances were grouped

Removing extreme value/outlier – IQR:

- Dataset used: diabetes.arff
- Pre-process→ filter→ Unsupervised→ attributes→ InterquartileRange→ Apply →Analyze Outlier
- Select outlier attribute:
- Pre-process →filter →Unsupervised →instance → RemovewithValues →Apply

  (Set attribute 10 (Column Number) in RemovewithValues i.e., Outlier)

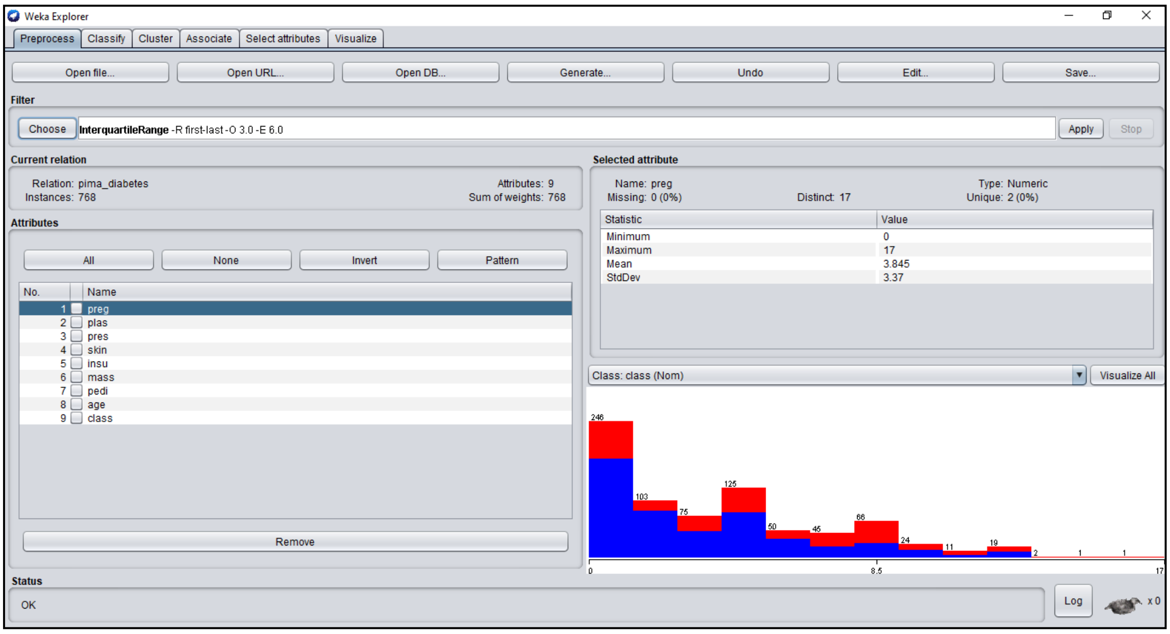
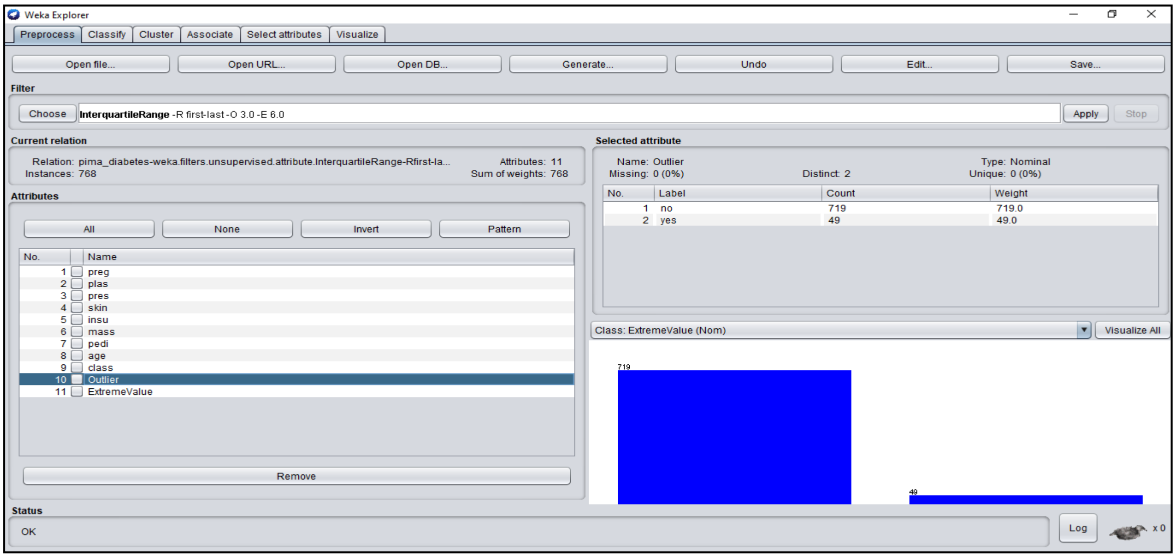
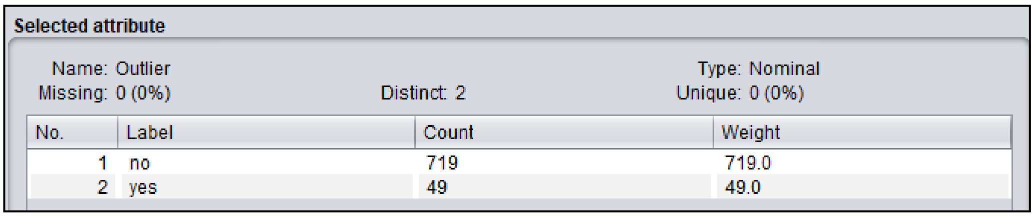


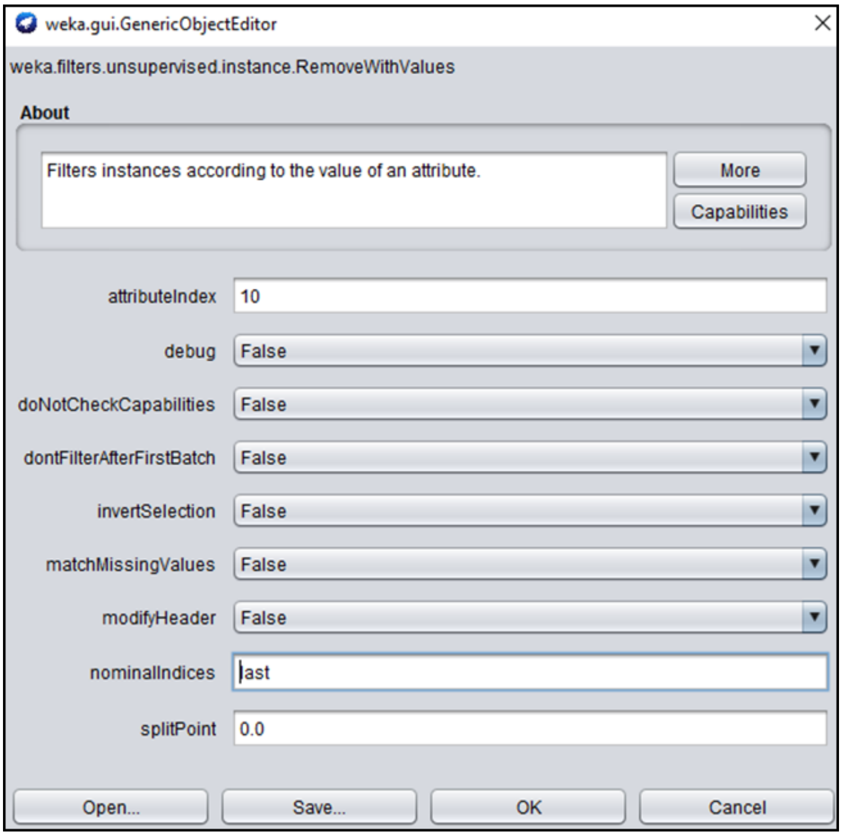
Fig. 1.14: InterquartileRange filter was applied



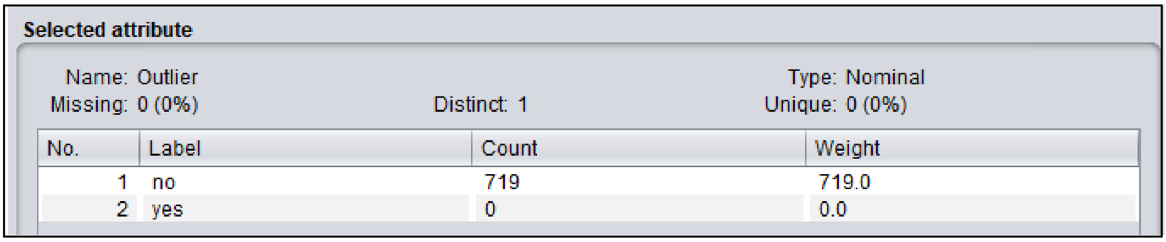
<u>**Fig. 1.15: Outlier was analyzed**</u>



<u>**Fig. 1.16: Outlier weightage – 49%**</u>



<u>**Fig. 1.17: RemoveWithValues attribute index changed to 10 i.e., Outlier attribute**</u>


Selected attribute

Name: Outlier
Missing: 0 (0%)
Distinct: 1
Type: Nominal
Unique: 0 (0%)

No.	Label	Count	Weight
1	no	719	719.0
2	yes	0	0.0

Fig. 1.18: Outlier i.e., extreme values were removed

RESULT AND INFERENCE:

The raw data from the available dataset were pre-processed in Weka v.3.8.5 tool. The results were analyzed and documented.

Date : 01/08/2024

EXERCISE 2

FEATURE SELECTION USING FILTER METHOD

AIM:

To perform the feature selection using filter method in Weka v.3.8.5 tool.

INTRODUCTION

FEATURE SELECTION:

Feature selection is the process of choosing a subset of input variables by eliminating features with little or no predictive information. It can improve the comprehensibility of the resulting classifier models and often build a model that generalizes better to unseen points and to find the correct subset of predictive features.

FILTER METHOD:

It uses a statistical measure to assign a scoring to each feature based upon the intrinsic properties of features. It is considered to be of feature independently or with regard to dependent variable.

Examples: chi – square test, variance threshold, correlation coefficient.

PROCEDURE

Loading a dataset:

- Open Weka v.3.8.5 tool and enter "Explorer"
- Open file → select a dataset (labor.arff) → Open

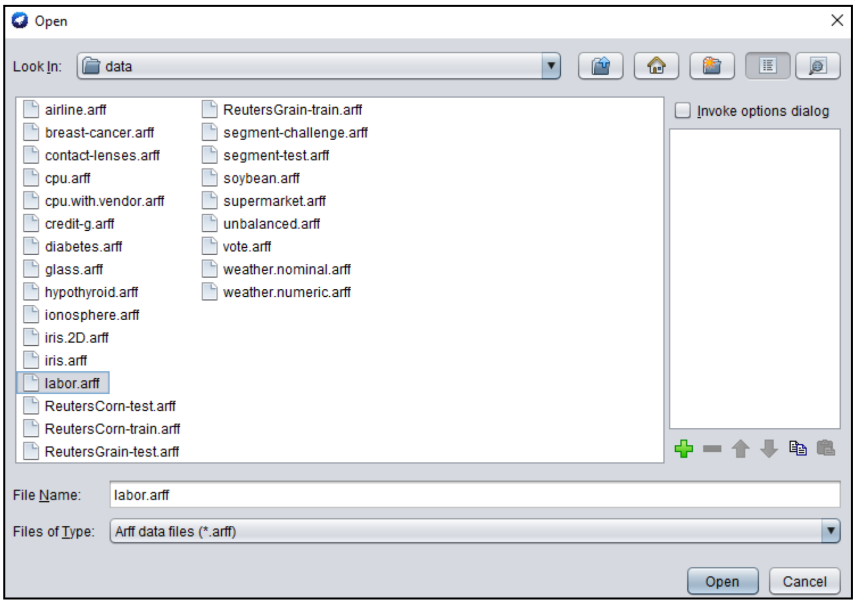
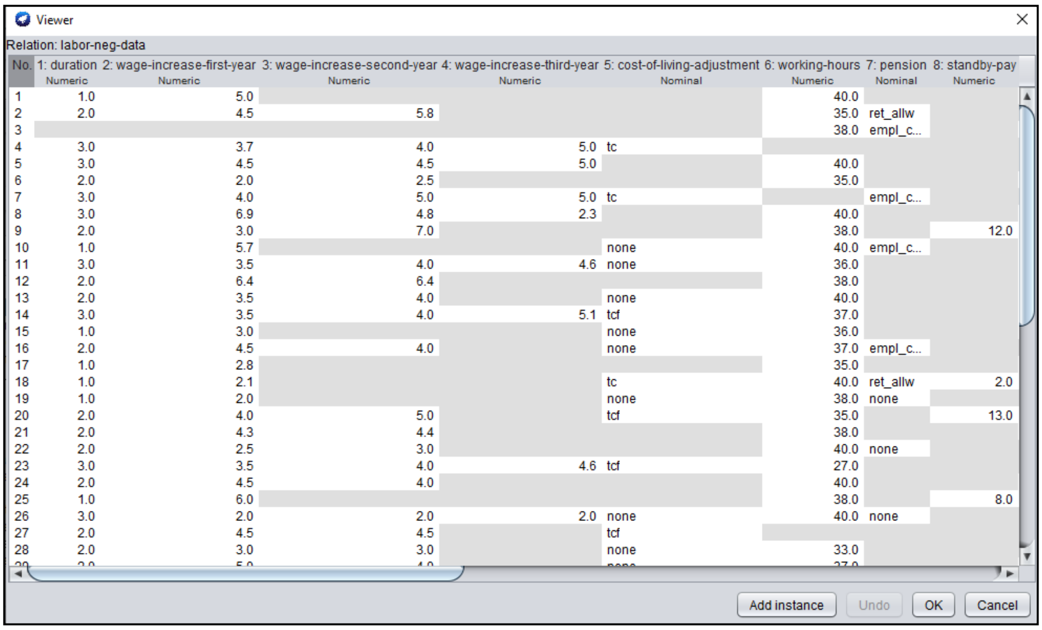


Fig. 2.1: labor.arff dataset was loaded

Preprocessing:

Replacing Missing values:

- Dataset used: labor.arff.
- Analyze the missing value data in the dataset.
- Pre-process →filter → Unsupervised →attributes →ReplaceMissingValues →Apply



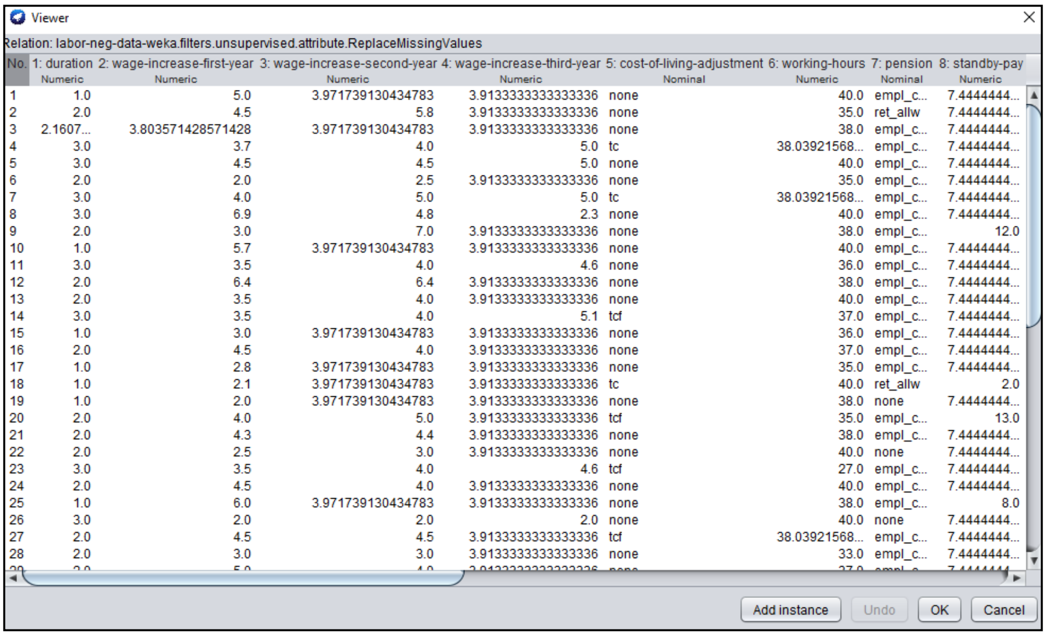
Viewer

Relation: labor-neg-data

No.	1: duration Numeric	2: wage-increase-first-year Numeric	3: wage-increase-second-year Numeric	4: wage-increase-third-year Numeric	5: cost-of-living-adjustment Nominal	6: working-hours Numeric	7: pension Nominal	8: standby-pay Numeric
1	1.0	5.0				40.0		
2	2.0	4.5	5.8			35.0	ret_allw	
3						38.0	empl_c...	
4	3.0	3.7	4.0	5.0	tc			
5	3.0	4.5	4.5	5.0		40.0		
6	2.0	2.0	2.5			35.0		
7	3.0	4.0	5.0	5.0	tc		empl_c...	
8	3.0	6.9	4.8	2.3		40.0		
9	2.0	3.0	7.0			38.0		12.0
10	1.0	5.7			none	40.0	empl_c...	
11	3.0	3.5	4.0	4.6	none	36.0		
12	2.0	6.4	6.4			38.0		
13	2.0	3.5	4.0		none	40.0		
14	3.0	3.5	4.0	5.1	tcf	37.0		
15	1.0	3.0			none	36.0		
16	2.0	4.5	4.0		none	37.0	empl_c...	
17	1.0	2.8				35.0		
18	1.0	2.1			tc	40.0	ret_allw	2.0
19	1.0	2.0			none	38.0	none	
20	2.0	4.0	5.0		tcf	35.0		13.0
21	2.0	4.3	4.4			38.0		
22	2.0	2.5	3.0			40.0	none	
23	3.0	3.5	4.0	4.6	tcf	27.0		
24	2.0	4.5	4.0			40.0		
25	1.0	6.0				38.0		8.0
26	3.0	2.0	2.0	2.0	none	40.0	none	
27	2.0	4.5	4.5		tcf			
28	2.0	3.0	3.0		none	33.0		

Add instance Undo OK Cancel

Fig. 2.2: labor.arff with missing values



Viewer

Relation: labor-neg-data-weka.filters.unsupervised.attribute.ReplaceMissingValues

No.	1: duration Numeric	2: wage-increase-first-year Numeric	3: wage-increase-second-year Numeric	4: wage-increase-third-year Numeric	5: cost-of-living-adjustment Nominal	6: working-hours Numeric	7: pension Nominal	8: standby-pay Numeric
1	1.0	5.0	3.971739130434783	3.9133333333333336	none	40.0	empl_c...	7.4444444...
2	2.0	4.5	5.8	3.9133333333333336	none	35.0	ret_allw	7.4444444...
3	2.1607...	3.803571428571428	3.971739130434783	3.9133333333333336	none	38.0	empl_c...	7.4444444...
4	3.0	3.7	4.0	5.0	tc	38.03921568...	empl_c...	7.4444444...
5	3.0	4.5	4.5	5.0	none	40.0	empl_c...	7.4444444...
6	2.0	2.0	2.5	3.9133333333333336	none	35.0	empl_c...	7.4444444...
7	3.0	4.0	5.0	5.0	tc	38.03921568...	empl_c...	7.4444444...
8	3.0	6.9	4.8	2.3	none	40.0	empl_c...	7.4444444...
9	2.0	3.0	7.0	3.9133333333333336	none	38.0	empl_c...	12.0
10	1.0	5.7	3.971739130434783	3.9133333333333336	none	40.0	empl_c...	7.4444444...
11	3.0	3.5	4.0	4.6	none	36.0	empl_c...	7.4444444...
12	2.0	6.4	6.4	3.9133333333333336	none	38.0	empl_c...	7.4444444...
13	2.0	3.5	4.0	3.9133333333333336	none	40.0	empl_c...	7.4444444...
14	3.0	3.5	4.0	5.1	tcf	37.0	empl_c...	7.4444444...
15	1.0	3.0	3.971739130434783	3.9133333333333336	none	36.0	empl_c...	7.4444444...
16	2.0	4.5	4.0	3.9133333333333336	none	37.0	empl_c...	7.4444444...
17	1.0	2.8	3.971739130434783	3.9133333333333336	none	35.0	empl_c...	7.4444444...
18	1.0	2.1	3.971739130434783	3.9133333333333336	tc	40.0	ret_allw	2.0
19	1.0	2.0	3.971739130434783	3.9133333333333336	none	38.0	none	7.4444444...
20	2.0	4.0	5.0	3.9133333333333336	tcf	35.0	empl_c...	13.0
21	2.0	4.3	4.4	3.9133333333333336	none	38.0	empl_c...	7.4444444...
22	2.0	2.5	3.0	3.9133333333333336	none	40.0	none	7.4444444...
23	3.0	3.5	4.0	4.6	tcf	27.0	empl_c...	7.4444444...
24	2.0	4.5	4.0	3.9133333333333336	none	40.0	empl_c...	7.4444444...
25	1.0	6.0	3.971739130434783	3.9133333333333336	none	38.0	empl_c...	8.0
26	3.0	2.0	2.0	2.0	none	40.0	none	7.4444444...
27	2.0	4.5	4.5	3.9133333333333336	tcf	38.03921568...	empl_c...	7.4444444...
28	2.0	3.0	3.0	3.9133333333333336	none	33.0	empl_c...	7.4444444...

Add instance Undo OK Cancel

Fig. 2.3: labor.arff without missing values

Filter method:

(i) Correlation based feature selection:

Select Attributes →choose attribute selection →CorrelationAttributeEval →set Ranker insearch method →Start →Analyze

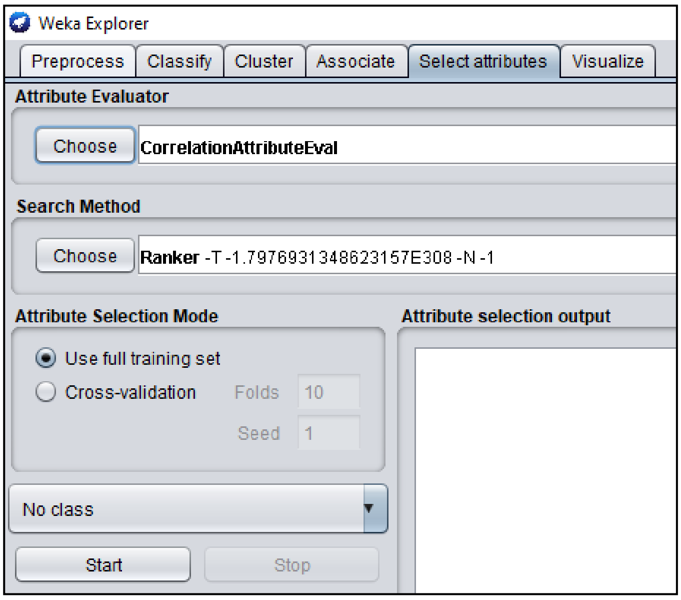


Fig. 2.4: CorrelationAttributeEval was chosen

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

Search Method:
	Attribute ranking.

Attribute Evaluator (supervised, Class (nominal): 17 class):
	Correlation Ranking Filter
Ranked attributes:
 0.6103   7 pension
 0.6054   2 wage-increase-first-year
 0.5496  13 longterm-disability-assistance
 0.5358   3 wage-increase-second-year
 0.4243  11 statutory-holidays
 0.4011   4 wage-increase-third-year
 0.386    8 standby-pay
 0.3231  16 contribution-to-health-plan
 0.3206  15 bereavement-assistance
 0.3173   6 working-hours
 0.2435   9 shift-differential
 0.1886  12 vacation
 0.1857  14 contribution-to-dental-plan
 0.1699   1 duration
 0.1072   5 cost-of-living-adjustment
 0.0492  10 education-allowance

Selected attributes: 7,2,13,3,11,4,8,16,15,6,9,12,14,1,5,10 : 16
```

Fig. 2.5: CorrelationAttributeEval – output

(ii) Information gain based feature selection:

Select Attributes→choose attribute selection→ InfoGainAttributeEval→ set Ranker in search method →Start →Analyze

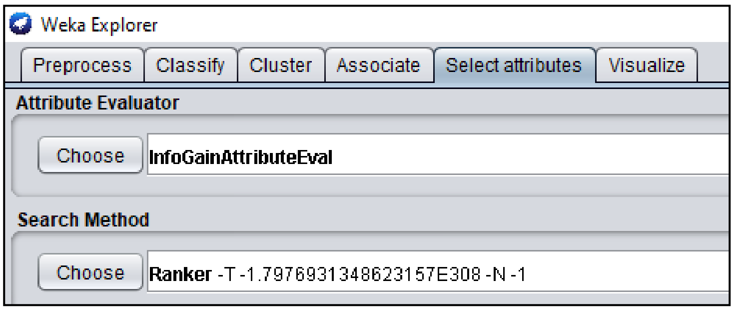
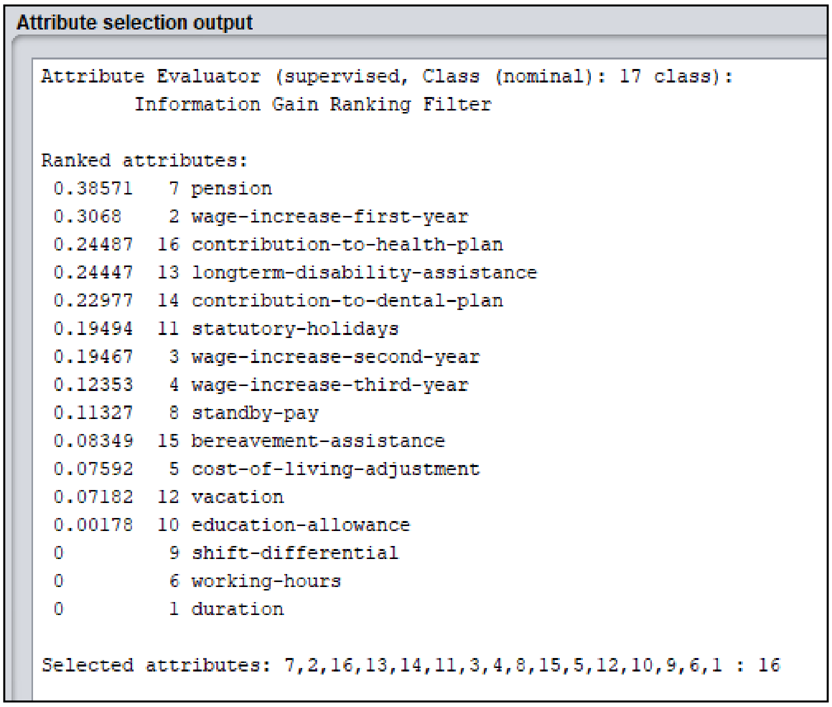


Fig. 2.6: InfoGainAttributeEval was chosen



Attribute selection output

```
Attribute Evaluator (supervised, Class (nominal): 17 class):
        Information Gain Ranking Filter

Ranked attributes:
 0.38571    7 pension
 0.3068     2 wage-increase-first-year
 0.24487   16 contribution-to-health-plan
 0.24447   13 longterm-disability-assistance
 0.22977   14 contribution-to-dental-plan
 0.19494   11 statutory-holidays
 0.19467    3 wage-increase-second-year
 0.12353    4 wage-increase-third-year
 0.11327    8 standby-pay
 0.08349   15 bereavement-assistance
 0.07592    5 cost-of-living-adjustment
 0.07182   12 vacation
 0.00178   10 education-allowance
 0          9 shift-differential
 0          6 working-hours
 0          1 duration

Selected attributes: 7,2,16,13,14,11,3,4,8,15,5,12,10,9,6,1 : 16
```

Fig. 2.7: InfoGainAttribute Eval - output

(iii) Chi-square based feature selection:

Select Attributes → choose attribute selection → ChiSquaredAttributeEval → set Ranker in search method → Start → Analyze

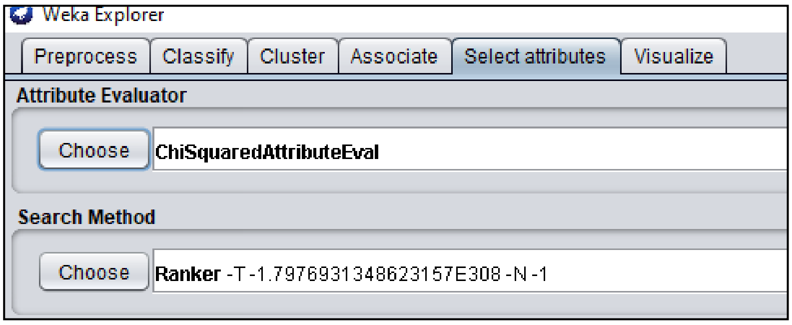


Fig. 2.8: ChiSquaredAttributeEval was chosen

```
Attribute selection output

Attribute Evaluator (supervised, Class (nominal): 17 class):
        Chi-squared Ranking Filter

Ranked attributes:
26.998     7 pension
23.7782    2 wage-increase-first-year
17.2412   16 contribution-to-health-plan
17.2163   13 longterm-disability-assistance
15.5576   11 statutory-holidays
14.6852    3 wage-increase-second-year
14.2933   14 contribution-to-dental-plan
 7.9585    8 standby-pay
 6.5555    4 wage-increase-third-year
 5.9077    5 cost-of-living-adjustment
 5.8583   15 bereavement-assistance
 5.0776   12 vacation
 0.1378   10 education-allowance
 0         6 working-hours
 0         9 shift-differential
 0         1 duration

Selected attributes: 7,2,16,13,11,3,14,8,4,5,15,12,10,6,9,1 : 16
```

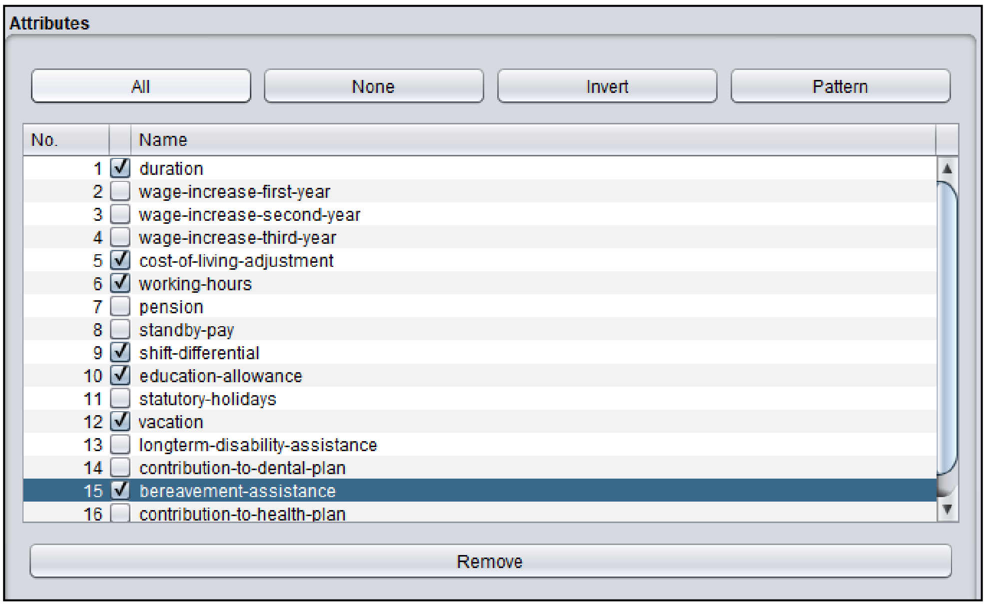
Fig. 2.9: ChiSquaredAttributeEval – output

Attribute selection

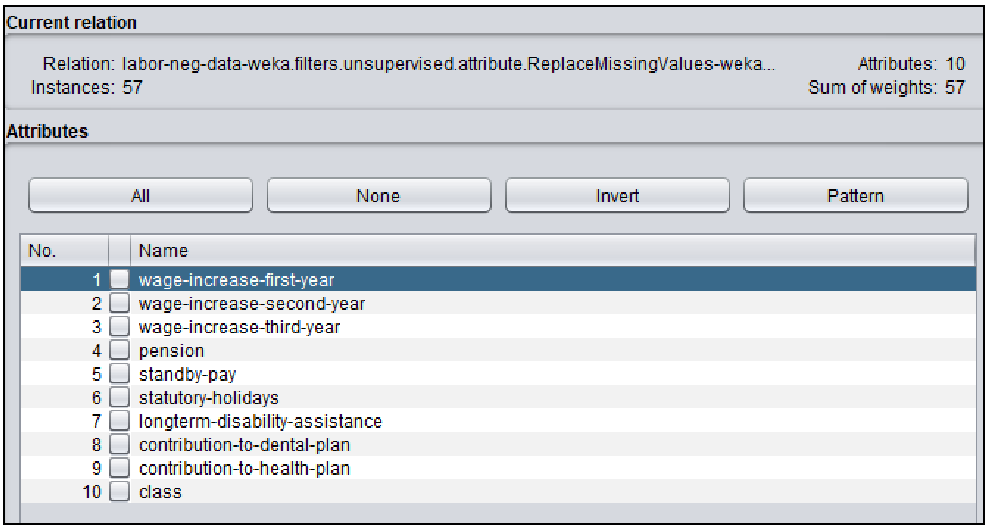
Select the best attributes by analyzing output ranks from all feature selection methods and select attributes which are not needed further. Those attributes that needs to be removed are 1,5,6,9,10,12 and 15.

Attribute removal:

Preprocess→ check the unwanted attributes →remove and save the file



<u>Fig. 2.10: Unwanted attributes were checked</u>



<u>Fig. 2.11: Unwanted attributes were removed</u>

<u>RESULT AND INFERENCE</u>:

The dataset – labor.arff was preprocessed and feature selected using filter methods (Correlation based feature selection, Information gain based feature selection and Chi-square based feature selection) in Weka v.3.8.5 tool. The best attributes were kept for further analysis while the unwanted attributes were removed.

FEATURE SELECTION USING WRAPPER METHOD

AIM

To perform the feature selection using wrapper method in Weka v.3.8.5 tool.

INTRODUCTION

FEATURE SELECTION:

Feature selection is the process of choosing a subset of input variables by eliminating features with little or no predictive information. It can improve the comprehensibility of the resulting classifier models and often build a model that generalizes better to unseen points and to find the correct subset of predictive features.

WRAPPER METHOD:

It is used as predictive model to score feature subsets. It measures the usefulness of features based on the classifier performance. It is very computationally intensive but usually provide the best performing feature set for that particular model. It has following methods, they are forward selection, backward elimination and recursive feature elimination.

PROCEDURE

- Load the pre-processed dataset – labor.arff in weka tool.
- Select Attributes→choose attribute selection→ WrapperSubsetEval→ set RandomSearch in search method →Start →Analyze
- Keep the selected attributes from the output (in Fig. 3.2) i.e., 2,3,4,6,11,12,13,15 and remove the unselected attributes from the dataset i.e., 1,5,7,8,9,10,14,16,17.

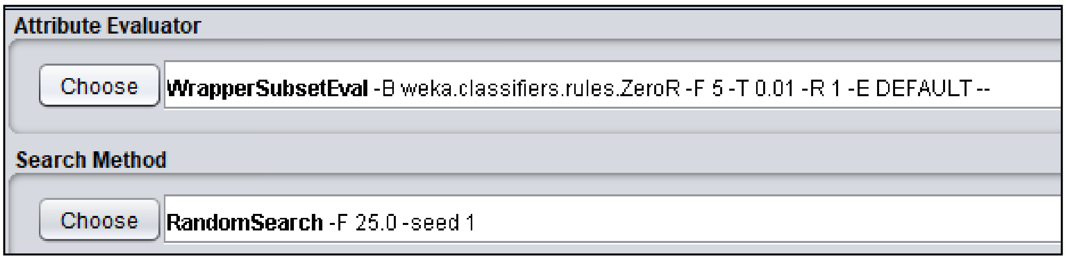


Fig: Wrapper SubsetEval and RandomSearch were selected

```
Attribute selection output

Search Method:
        Random search.
        Start set: no attributes
        Number of iterations: 16384 (25.0% of the search space)
        Merit of best subset found:    0.649

Attribute Subset Evaluator (supervised, Class (nominal): 17 class):
        Wrapper Subset Evaluator
        Learning scheme: weka.classifiers.rules.ZeroR
        Scheme options:
        Subset evaluation: classification accuracy
        Number of folds for accuracy estimation: 5

Selected attributes: 2,3,4,6,11,12,13,15 : 8
                     wage-increase-first-year
                     wage-increase-second-year
                     wage-increase-third-year
                     working-hours
                     statutory-holidays
                     vacation
                     longterm-disability-assistance
                     bereavement-assistance
```

Fig : Wrapper SubsetEval and RandomSearch – output

8 attributes were selected and other 9 attributes were removed and processed.

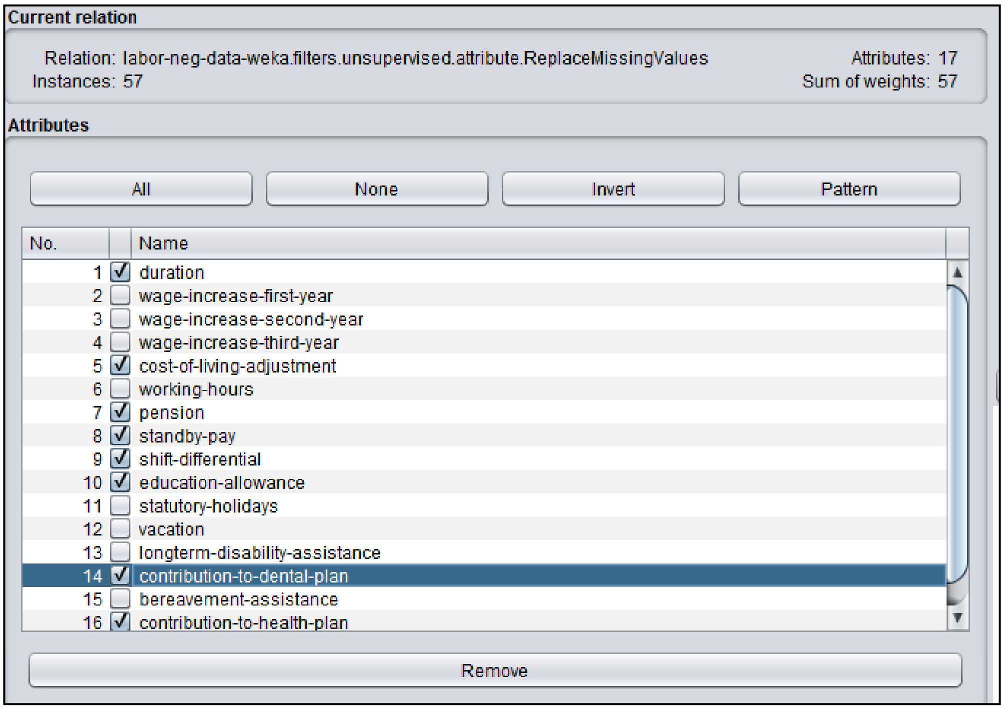


Fig : Unselected attributes -9 were checked to remove

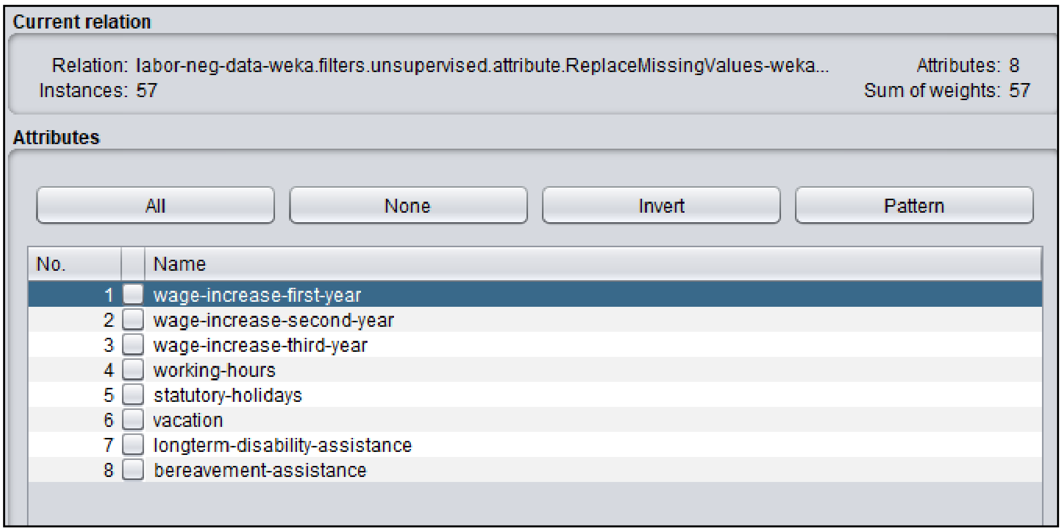


Fig : Unselected attributes were removed and retained selected attributes

RESULT AND INFERENCE:

The dataset – labor.arff was preprocessed and feature selected using wrapper method and RandomSearch in Weka v.3.8.5 tool. The best attributes were kept for further analysis while the unwanted attributes were removed.

Date: 08/08/2024

EXERCISE 3

ASSOCIATION RULE PROCESS USING APRIORI ALGORITHM

AIM:

To perform the association rule process using apriori algorithm in Weka v.3.8.5 tool.

INTRODUCTION

Association rules are created by searching data for frequent if-then patterns and using the criteria support and confidence to identify the most important relationships.

Apriori algorithm is used for finding frequent itemset in a dataset for Boolean association rule. It uses prior knowledge of frequent itemset properties and applies an iterative approach or level wise search where k-frequent itemset are used to find k+1 itemset.

Steps in Apriori Algorithm:

- Determine the support of itemset in the transactional database and select the minimum support and confidence.
- Take all supports in the transaction with higher support value than the minimum or selected support value.
- Find all the rules of these subsets that have higher confidence value than the threshold or minimum confidence.
- Sort the rules as the decreasing order of lift.

DATASET USED:

weather.nominal.arff

PROCEDURE

- Open Weka v.3.8.5 tool and enter "Explorer"
- Open file →select a dataset (weather_nominal.arff) →Open
- Analyze the missing value data in the dataset.
- Pre-process → filter →Unsupervised →attributes →ReplaceMissingValues → Apply
- Associate →choose associator →Apriori →Start

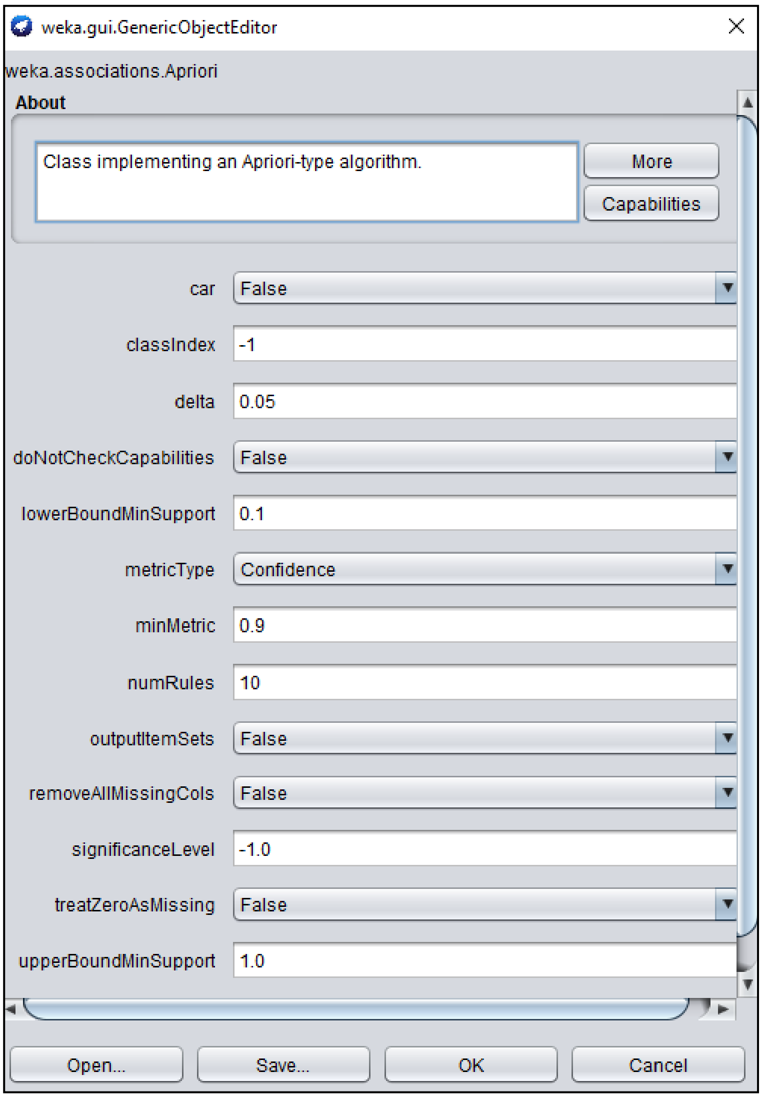


Fig. 3.1: Apriori Associator was selected

(Number of rules = 10; Minimum support = 0.1 and Minimum confidence = 0.9)

Associator output

```
Minimum support: 0.15 (2 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 17

Generated sets of large itemsets:

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

Size of set of large itemsets L(2): 47

Size of set of large itemsets L(3): 39

Size of set of large itemsets L(4): 6

Best rules found:

 1. outlook=overcast 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 2. temperature=cool 4 ==> humidity=normal 4    <conf:(1)> lift:(2) lev:(0.14) [2] conv:(2)
 3. humidity=normal windy=FALSE 4 ==> play=yes 4    <conf:(1)> lift:(1.56) lev:(0.1) [1] conv:(1.43)
 4. outlook=sunny play=no 3 ==> humidity=high 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 5. outlook=sunny humidity=high 3 ==> play=no 3    <conf:(1)> lift:(2.8) lev:(0.14) [1] conv:(1.93)
 6. outlook=rainy play=yes 3 ==> windy=FALSE 3    <conf:(1)> lift:(1.75) lev:(0.09) [1] conv:(1.29)
 7. outlook=rainy windy=FALSE 3 ==> play=yes 3    <conf:(1)> lift:(1.56) lev:(0.08) [1] conv:(1.07)
 8. temperature=cool play=yes 3 ==> humidity=normal 3    <conf:(1)> lift:(2) lev:(0.11) [1] conv:(1.5)
 9. outlook=sunny temperature=hot 2 ==> humidity=high 2    <conf:(1)> lift:(2) lev:(0.07) [1] conv:(1)
10. temperature=hot play=no 2 ==> outlook=sunny 2    <conf:(1)> lift:(2.8) lev:(0.09) [1] conv:(1.29)
```

Fig. 3.2: Apriori Associator output

RESULT AND INFERENCE:

The association rule process using Apriori algorithm was performed on weather.nominal.arff dataset in Weka v.3.8.5 tool. Total of 10 best rules were generated and it was inferred that in the first rule which is outlook=overcast => play=yes, states that the if the weather outlook is overcast then we can play.

Date: 29/08/2024

EXERCISE 4

CLASSIFICATION RULE PROCESS USING J48 ALGORITHM

AIM:

To perform the classification rule process using j48 algorithm in Weka v.3.8.5 tool.

INTRODUCTION:

Classification is a form of data analysis that can be used to construct a model which can be further used in future to predict the class label of new dataset. It is done in two step processes. They are

- Learning step
- Accuracy check

Methods in classification,

- Classification by Decision tree induction
- Bayesian classification
- Rule based classification
- Classification by backpropagation
- Support vector machines
- Associative classification

Classification by Decision tree induction: J48

In decision tree, each internal nodes represent an attribute test happening, each branch represents an ending of the test, class label is represented by each leaf node or terminal node. Given each tuple, the attribute value of the tuple is tested next to the decision tree. A path is traced from root to leaf node which holds the class prediction used for the tuple. Then this decision tree is converted to classification rules.

J48 Algorithm:

J48 considers the standardized data gain that really results in splitting the information by choosing an attribute.

Steps:

1. The leaf is labelled with a similar class if the instances belong to the similar class.
2. For each attribute, the potential data will be figured and the gain in the data will betaken from the test on the attribute.
3. Finally, the best attribute will be chosen depending upon the current selection parameter.

DATASET USED:

labor.arff

PROCEDURE:

- Open Weka v.3.8.5 tool and enter “Explorer”
- Load the pre-processed dataset – labor.arff.
- Classify→ choose classifier → trees-J48 → set 10 folds in cross validation → Start→ Analyze

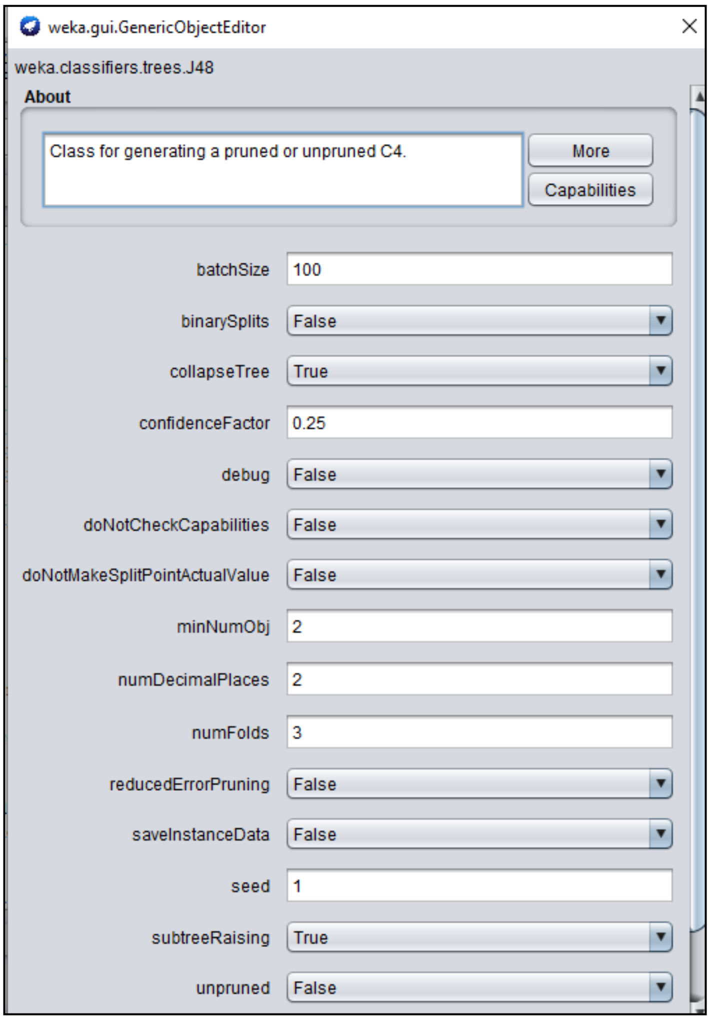


Fig. 4.1: J48 classifier was selected

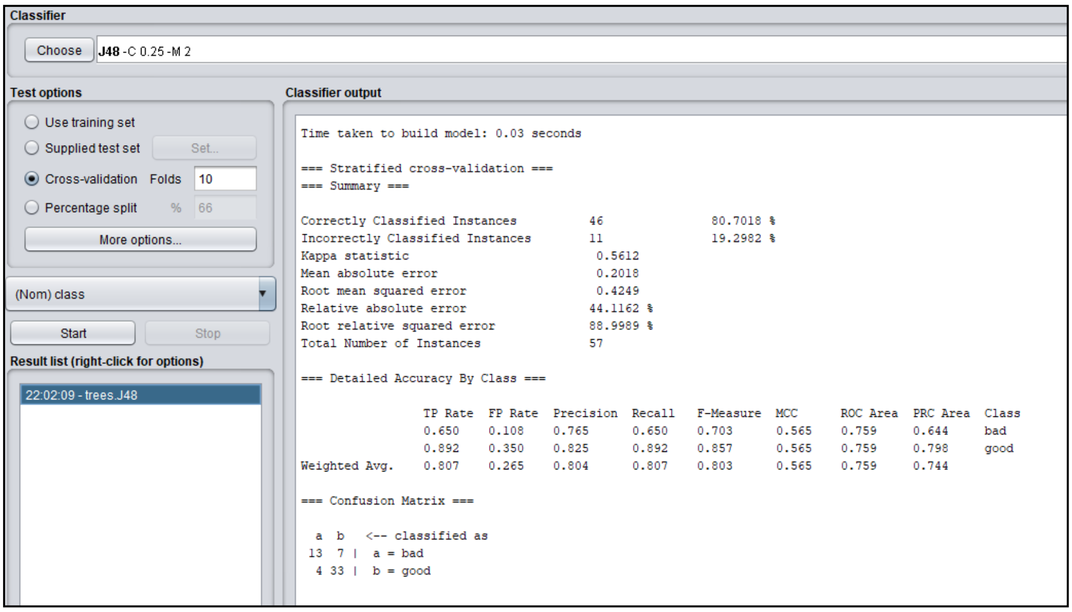


Fig. 4.2: J48 classifier output

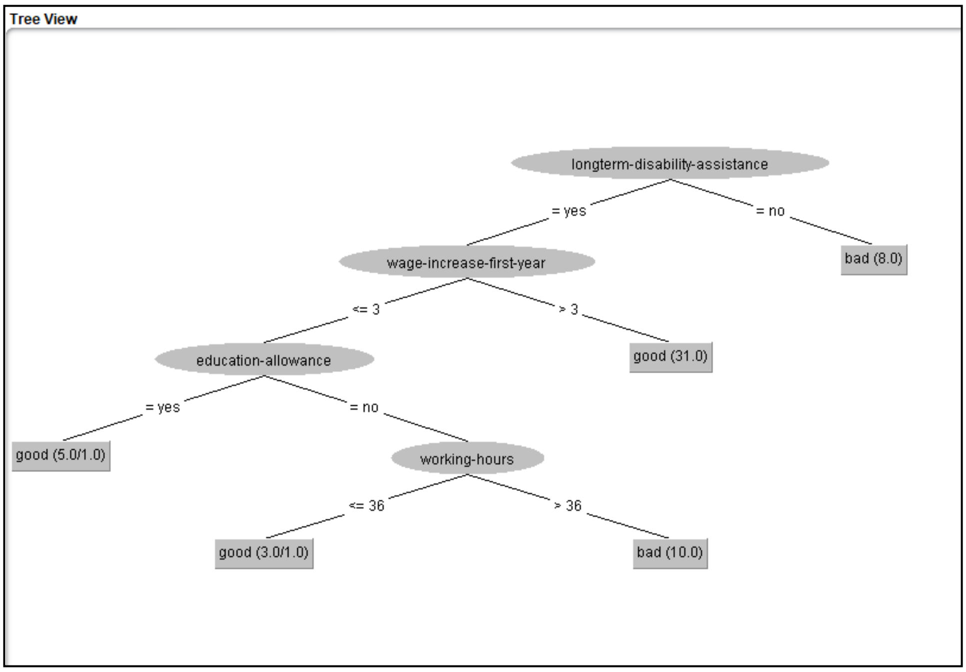


Fig. 4.3: J48 Classifier output – decision tree visualization

RESULT AND INFERENCE:

The classification rule process was performed on labor dataset using j48 algorithm in Weka v.3.8.5 tool and a decision tree was generated. From the output it was inferred that the longterm-disability-assistance was the main attribute. If a labor is getting a longterm-disability-assistance then he gets wage-increase-first-year within 3 years and if he gets education allowance, he is considered to be good.

Date: 12/09/2024

EXERCISE 5

CLASSIFICATION RULE PROCESS ON DATASET USING NAÏVE BAYES ALGORITHM

Aim:

This experiment illustrates the use of naïve bayes classifier in weka. The sample data set used in this experiment is “weather.numeric.arff” available in arff format.

Description:

The Naive Bayesian classifier is based on Bayes’ theorem with independence assumptions between predictors. A Naive Bayesian model is easy to build, with no complicated iterative parameter estimation which makes it particularly useful for very large datasets. Despite its simplicity, the Naive Bayesian classifier often does surprisingly well and is widely used because it often outperforms more sophisticated classification methods.

Algorithm:

Bayes theorem provides a way of calculating the posterior probability, P(c|x), from P(c), P(x), and P(x|c). Naive Bayes classifier assume that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence.

Likelihood Class Prior Probability

$$\underbrace{P(c\mid x)}_{\text{Posterior Probability}}=\frac{P(x\mid c)P(c)}{P(x)}$$

Posterior Probability Predictor Prior Probability

$$P(c\mid X) = P(x_1\mid c)\times P(x_2\mid c)\times\cdots\times P(x_n\mid c)\times P(c)$$

- P(c|x) is the posterior probability of class (target) given predictor (attribute).
- P(c) is the prior probability of class.
- P(x|c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

The posterior probability can be calculated by first, constructing a frequency table for each attribute against the target. Then, transforming the frequency tables to likelihood tables and finally use the Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction

$$P(x \mid c) = P(Sunny \mid Yes) = 3/9 = 0.33$$

Frequency Table		Play Golf	
		Yes	No
Outlook	Sunny	3	2
	Overcast	4	0
	Rainy	2	3

Likelihood Table		Play Golf		
		Yes	No	
Outlook	Sunny	3/9	2/5	5/14
	Overcast	4/9	0/5	4/14
	Rainy	2/9	3/5	5/14
		9/14	5/14	

$$P(x) = P(Sunny) = 5/14 = 0.36$$

$$P(c) = P(Yes) = 9/14 = 0.64$$

Posterior Probability: $P(c \mid x) = P(Yes \mid Sunny) = 0.33 \times 0.64 \div 0.36 = 0.60$

<u>Preparing Test File:</u>

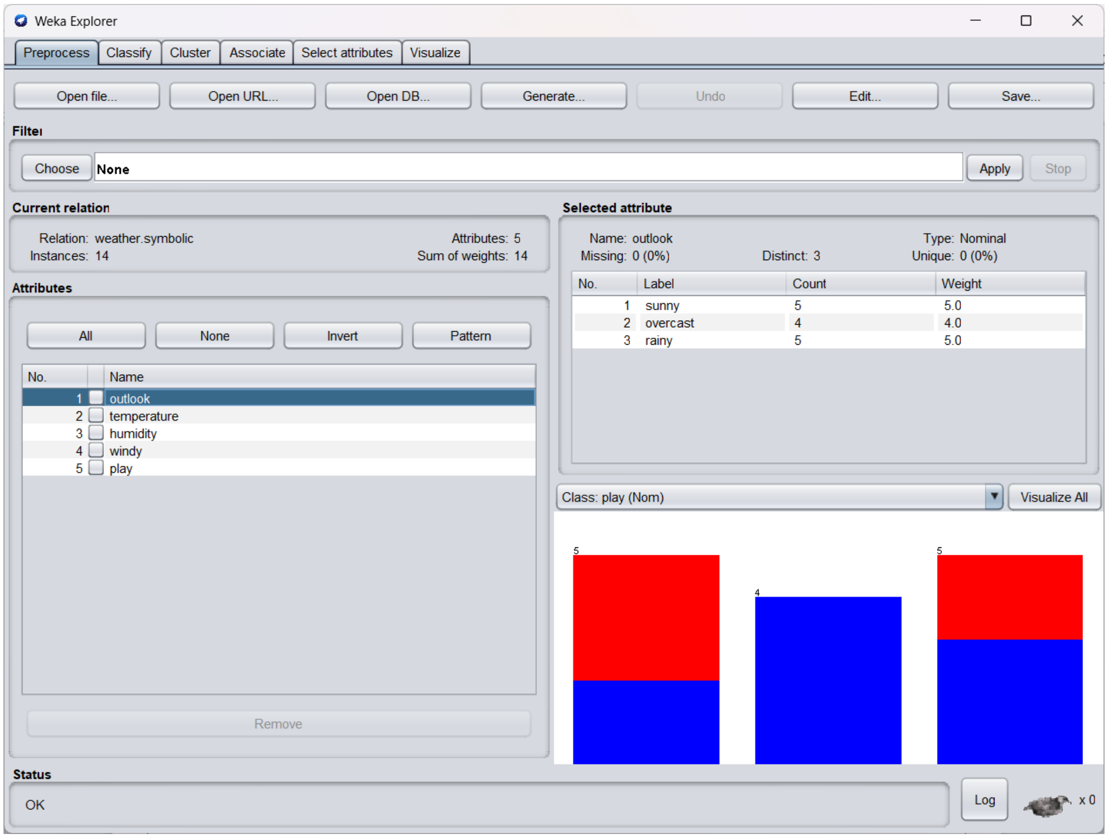
- Copy the attribute definitions from the training ARFF file into a new test ARFF file.
- Include a proper name for the relation in the test file, say @relation weather-test
- Include your test data after the @data statement. This may be a single instance (if you want to classify this instance) or a set of instances (if you want to evaluate the classifier).

<u>Example:</u>

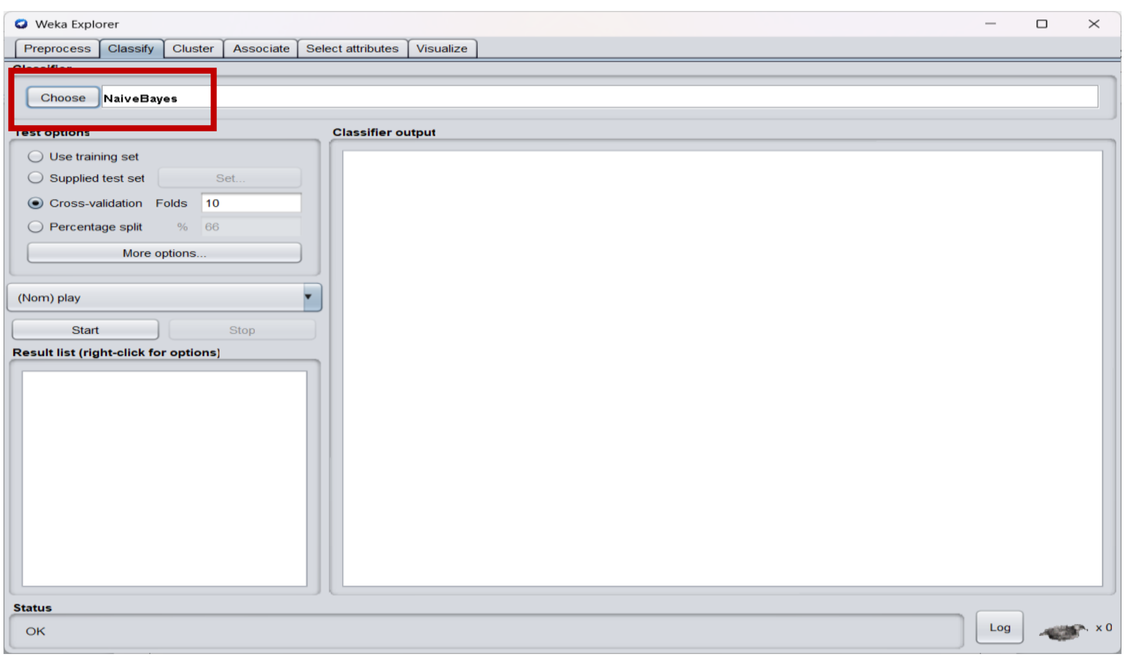
weather-test.arf

<u>Specifying Test Options:</u>

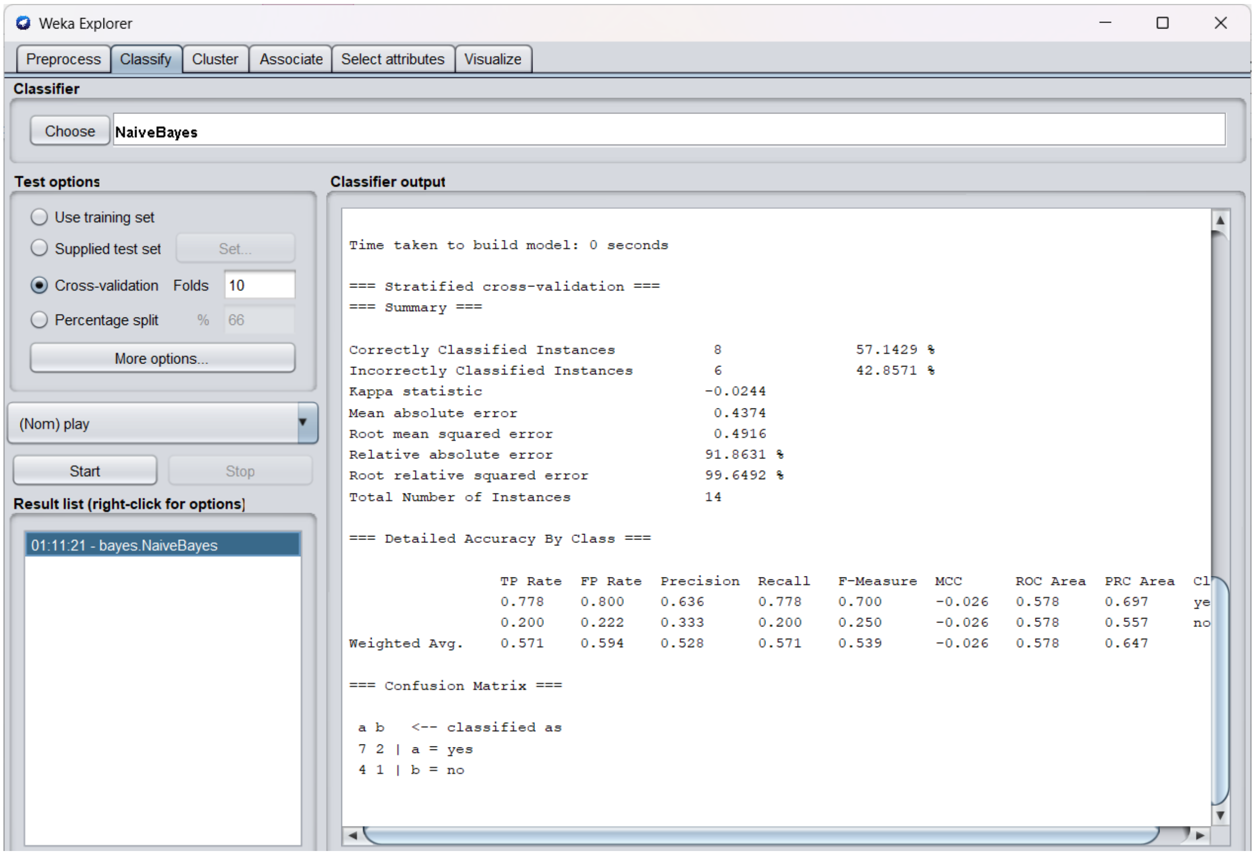
1. Get to the Weka Explorer environment and load the training file using the Preprocess mode. Try first with weather.numeric.arff.



2. Get to the Classify mode (by clicking on the Classify tab) as shown below:



3. Run the classifier you want and look at the Classifier output window. Assume you have chosen NaiveBayes, then you see the following:



Correctly Classified Instances (shown on top) tells you that your guess was correct (according to Naïve Bayes). You can see these results from the Confusion Matrix too: your instance is classified as no (b), with actual classification (your guess) no (b).

Result:

Classification rule processing on given dataset using Naïve Bayes Algorithm is done successfully.

Date: 19/09/2024

EXERCISE 6

CLASSIFICATION RULE PROCESS USING SVM ALGORITHM

AIM

To perform the classification rule process using SVM algorithm in Weka v.3.8.5 tool.

INTRODUCTION

Classification is a form of data analysis that can be used to construct a model which can be further used in future to predict the class label of new dataset. It is done in two step processes. They are

- Learning step
- Accuracy check

Methods in classification,

- Classification by Decision tree induction
- Bayesian classification
- Rule based classification
- Classification by backpropagation
- Support vector machines
- Associative classification

Support Vector Machine

Support vector machine (SVM) is a supervised learning algorithm which can be used for classification as well as regression problems. It creates the decision boundary (hyperplane) that segregates n-dimensional space into classes so that a new data point in the correct category can be placed. SVM chooses the extreme points/vectors that help in creating the hyperplane. This is called support vectors.

Require: $\boldsymbol{X}$ and $\boldsymbol{y}$ loaded with training labeled data, $\boldsymbol{\alpha} \Leftarrow 0$ or $\boldsymbol{\alpha} \Leftarrow$ partially trained SVM

$C \Leftarrow$ some value (10 for example)

repeat

  for all $\{x_i, y_i\}, \{x_j, y_j\}$ **do**

    Optimize α_i and α_j

  end for

until no changes in $\boldsymbol{\alpha}$ or other resource constraint criteria met

Ensure: Retain only the support vectors ($\boldsymbol{\alpha}_i > 0$)

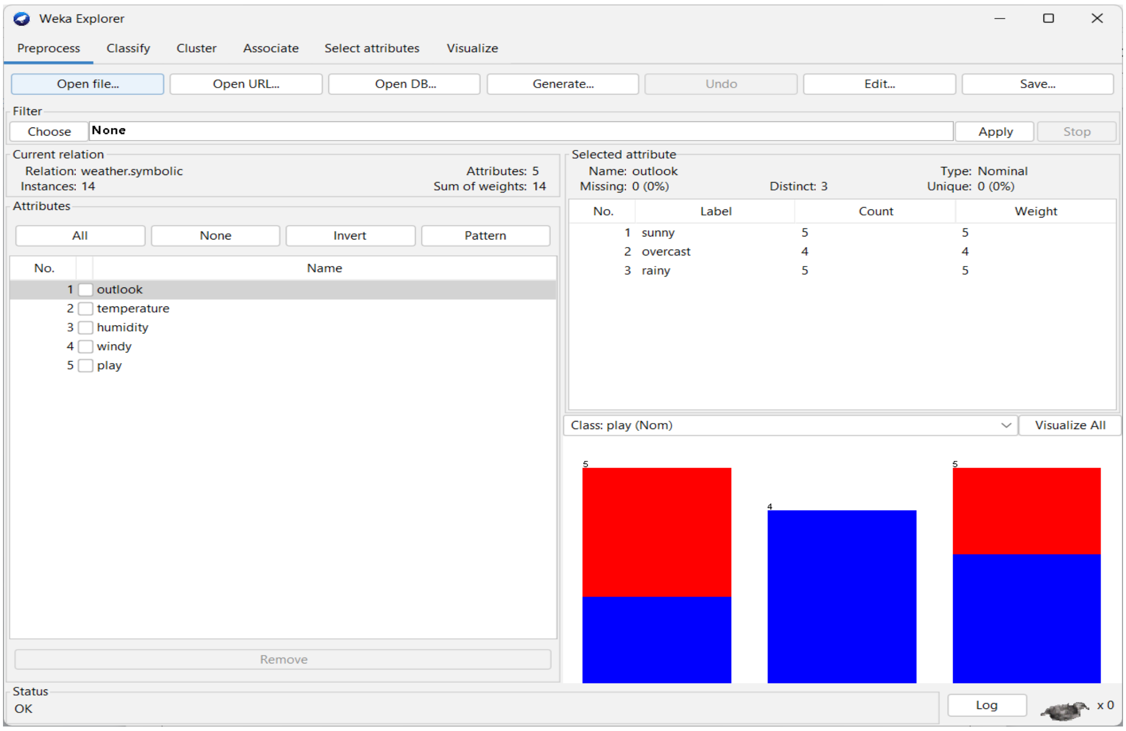
Fig. 6.1: SVM Pseudocode

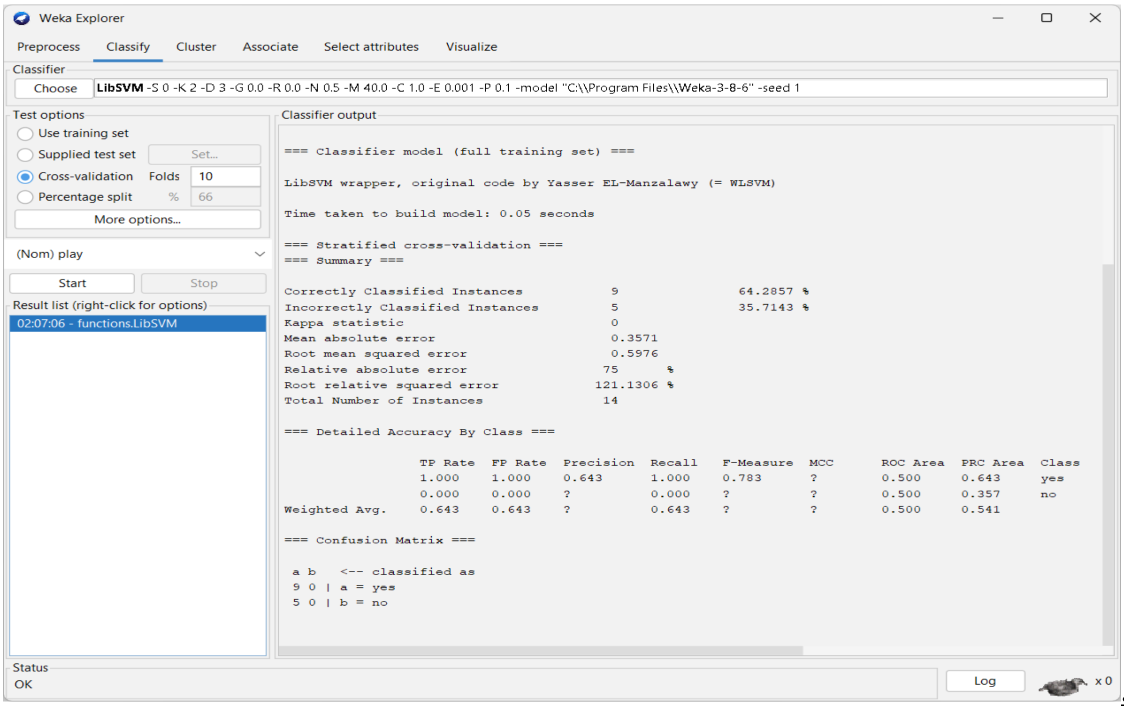
DATASET USED:

labor.arff

PROCEDURE:

- Open Weka v.3.8.5 tool and enter "Explorer"
- Load the pre-processed dataset – weather nominal.arff.
- Classify →choose classifier →LibSVM→Start→Analyze





.6.2

FIG LibSVM Classifier output

RESULT AND INFERENCE:

The classification rule process was performed with 10-fold cross validation on labor dataset using SVM algorithm (SMO) in Weka v.3.8.5 tool. From the output, it was inferred that the 91.2% of data in the dataset were correctly classified with less errors.

Date: 26/09/2024

EXERCISE:7

CLUSTERING RULE PROCESS USING SIMPLE K MEANS ALGORITHM

Aim:

To apply clustering in weka to group the similar instances in a dataset.

Cluster Analysis:

The process of grouping a set of physical or abstract objects into classes of similar objects is called Clustering. The objects are clustered or grouped based on the principle of maximizing the intraclass similarity and minimizing the interclass similarity.

Simple *k*-Means Clustering

The *k*-means algorithm takes the input parameter, *k*, and partitions a set of n objects into *k* clusters so that the resulting intracluster similarity is high but the intercluster similarity is low.

Algorithm: *k*-means. The *k*-means algorithm for partitioning where each cluster's center is represented by the mean value of objects in the dataset.

Input:

k: the number of clusters,

D: a data set containing n objects.

Output: A set of *k* clusters.

Method:

1. arbitarily choose *k* objects from d as the initial cluster centers;
2. **Repeat**
3. (re)assign each object to the cluster to which the object is the most similar, based on the mean value of the objects in the cluster;
4. Update the cluster means, i.e., calculate the mean value of the objects for each cluster;

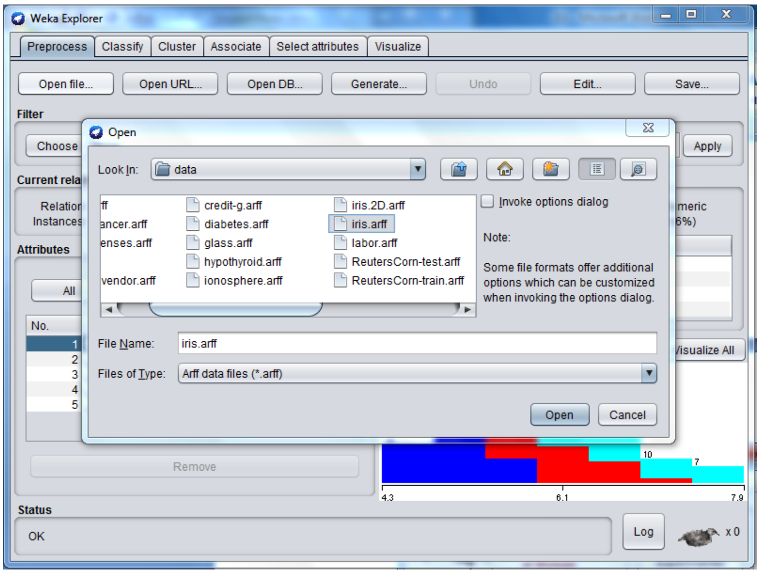
5. **Until** no change;

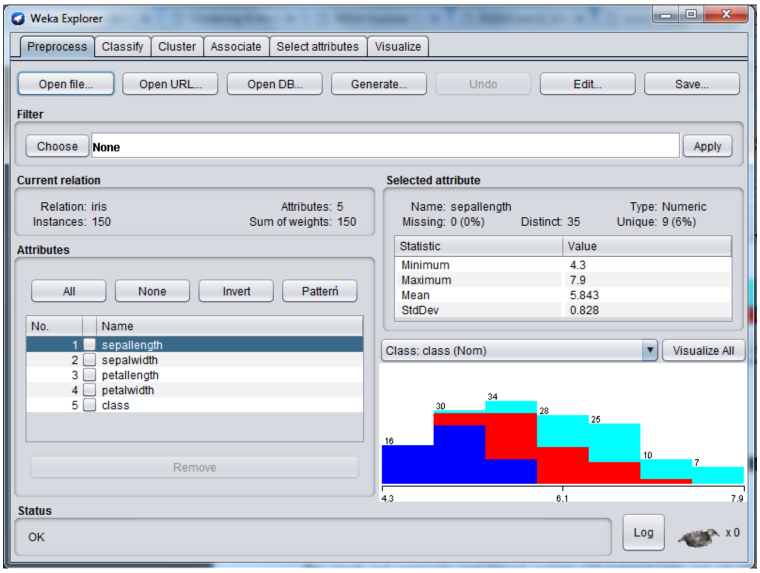
DATASET USED:

Iris data set

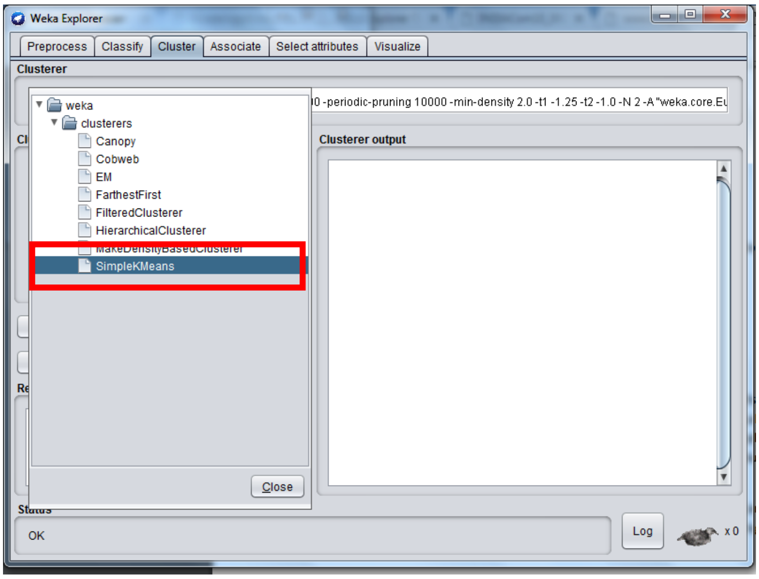
Steps for Cluster Analysis:

- **Load the Dataset**
    - Start WEKA
    - Select the Explorer
    - In the preprocess tab choose Open file and choose iris.arff file.

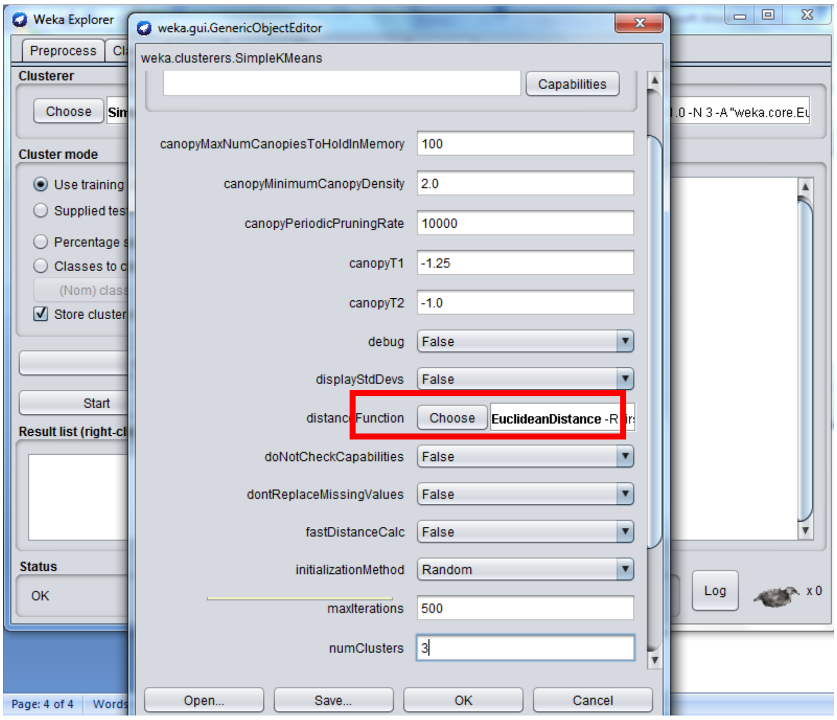




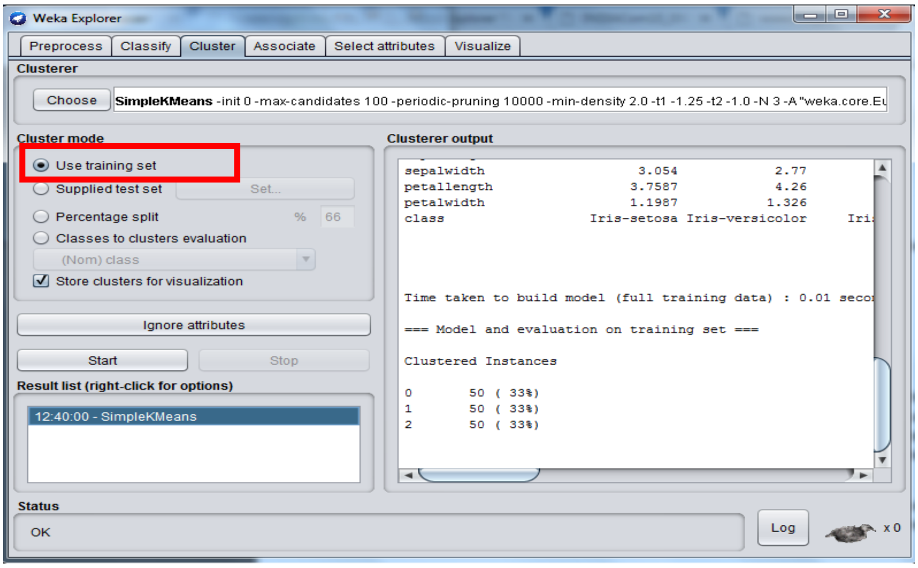
- **Clustering**
  - Select Cluster tab
  - Select the Cluster “Choose “and select simple *k*-Means



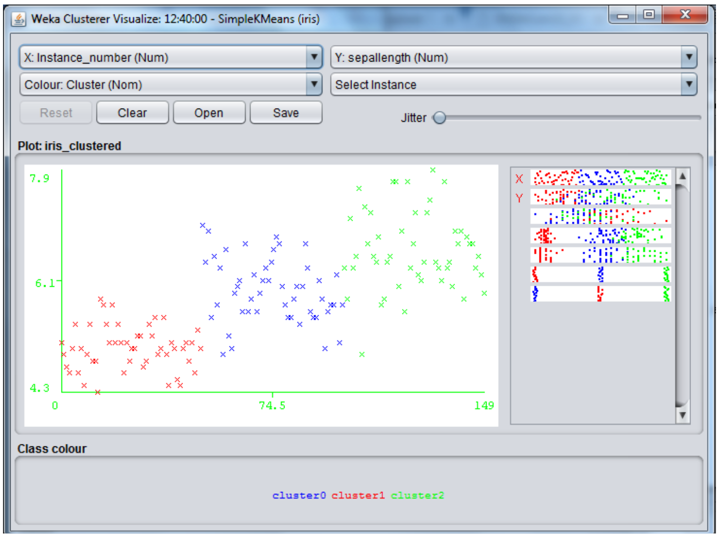
  - Click the Simple *k*-Means command box to the right of the Choose button, change the “numClusters” attribute to 3, and click the OK button



- Press Start to begin *k*-Means Clustering and evaluation.



- **Visualization**
  - Right click on the Result list and select visualize cluster assignments
  - Analyse the clusters.



Result:

Thus, Clustering rule process on dataset using simple k-mean is done successfully.