SRI VASAVI ENGINEERING COLLEGE

PEDATADEPALLI, TADEPALLIGUDEM.

Certificate

This is to certify that this is a bonafide record of Practical Workdone in **Data Mining Lab** by Mr.**Chandu Neelam** bearing Roll No.**21A81A05B4** of CSE branch during the academic year 2023-2024.

No. of Experiments Done: 10

Faculty In-charge **Head of the Department**

External Examiner

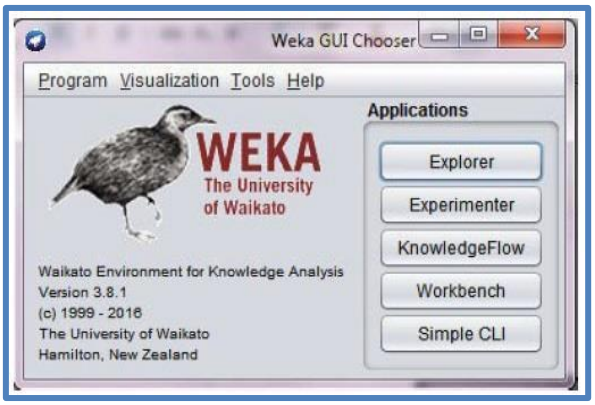
INDEX

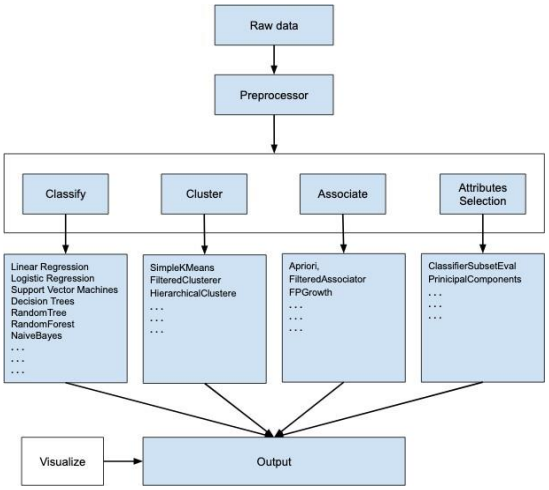
Exp No	Date	Experiment Name	Page No	Signature
0	09/08/2023	Weka Tool Introduction		
1	09/08/2023	Demonstrate Data Preprocessing on predefined Weka dataset labor.arff		
2	23/08/2023	Create a student.arff dataset and Demonstrate Data Preprocessing on it		
3	30/08/2023	Demonstrate Association rule process on predefined Weka dataset contact lenses.arff using apriori algorithm		
4	30/08/2023	Create an employee.arff dataset and demonstrate Association rule process on it using apriori algorithm		
5	13/09/2023	Demonstrate Classification process on student.arff dataset using J48 algorithm		
6	20/09/2023	Create a customer.arff dataset and demonstrate Classification process on it using J48 algorithm		
7	18/10/2023	Demonstrate Classification process on employee.arff dataset using id3 algorithm		
8	01/11/2023	Demonstrate Classification process on employee.arff dataset using Navie Bayes algorithm		
9	15/11/2023	Demonstrate Clustering process on predefined Weka dataset iris.arff using simple k-means algorithm		
10	15/11/2023	Demonstrate Clustering process on dataset student.arff using simple k-means algorithm		

Experiment 0

Weka Introduction

Weka is an Open source software provides the tools for data preprocessing, implementation of several Data Mining and Machine Learning tasks and visualization.



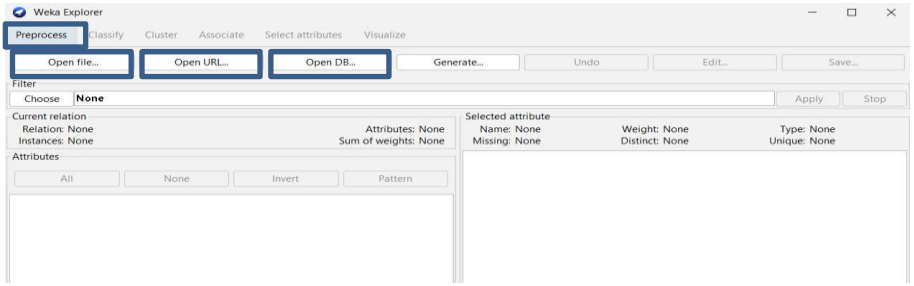


- Weka is developed by University of Waikato in New Zealand.
- "Weka" stands for the Waikato Environment for Knowledge Analysis.
- It is written using Java.
- Can download and install from https://waikato.github.io/weka-wiki/downloading_weka/
- Explorer: Gives access to all Weka's facilities using menu selection and form filling.
- Experimenter: Compares a variety of learning techniques interactively.
- KnowledgeFlow: Allows us to design configurations for streamed data processing. (Explorer holds everything in main memory).
- Workbench: Combines Explorer, Experimenter and Knowledgeflow into one application.
- Simple CLI: Simple Command Line Interface.

Weka Data Formates:

In Weka datasets can be loaded from any one of the following sources:

- Files (.arff, .csv, …)
- Web
- Database



Loading data from local file system:

Default file format for Weka is ARFF (Attribute Relation File Format).

```
@relation "employee"

@attribute id numeric

@attribute name string

@attribute contact_num numeric

@attribute dept
{HR,IT,MANAGEMENT,MAINTAINANCE}

@attribute DOB date dd-mm-yyyy

@attribute city string

@data

1,naman,1234556678,IT,02-08-2000,rjt

2,yash,1234556679,HR,04-05-2001,amd

3,kishan,1214556678,MANAGEMENT,02-11-2001,pbr

4,?,5234556678,IT,03-05-2000,amd
```

Tags:

- The @relation tag defines the name of the database.
- The @attribute tag defines the attributes.
- The @data tag starts the list of data rows each containing the comma separated fields.

Data types:

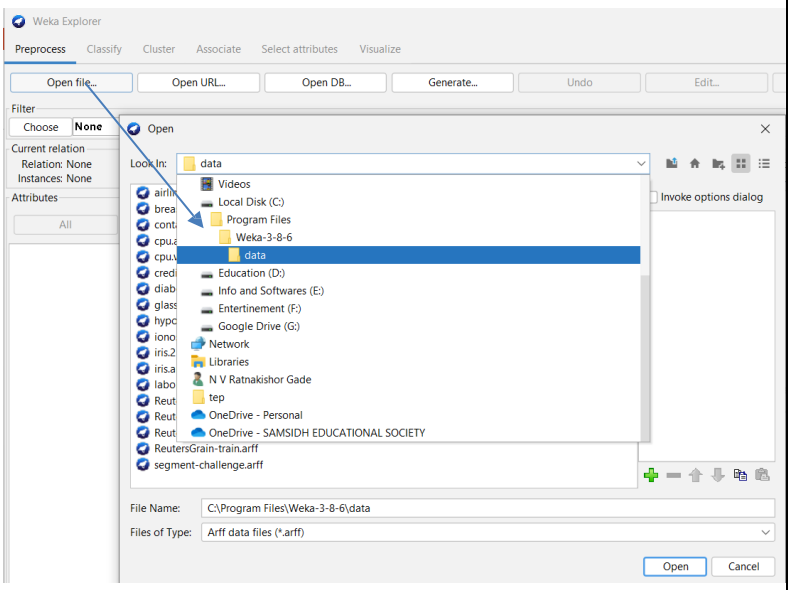
- nominal: categorical values - represented inside curly brackets.
- string : data type which accepts only string value
- numeric: used to store numbers
- date: used to store date

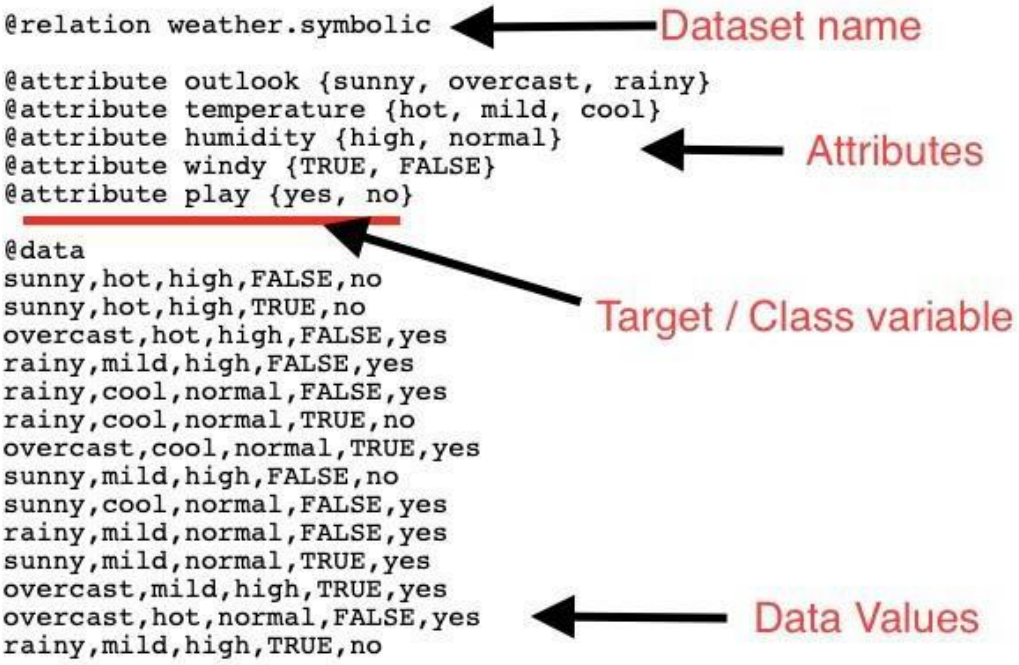
Weka is preloaded with some sample dataset files in its data folder.

Weka also supports other file formats, one can be selected while opening the file.

WEKA would give you the statistical output of the model processing. It provides you a visualization tool to inspect the data.

The various models can be applied on the same dataset. You can then compare the outputs of different models and select the best that meets your purpose.





- The @relation tag defines the name of the database.
- The @attribute tag defines the attributes.
- The @data tag starts the list of data rows each containing the comma separated fields.
- The attributes can take nominal values as in the case of outlook shown here −

  @attribute outlook (sunny, overcast, rainy)
- The attributes can take real values as in this case −

  @attribute temperature real
- You can also set a Target or a Class variable called play as shown here −

  @attribute play (yes, no)
- The Target assumes two nominal values yes or no.

Conclusion:

Experiment 1

Data Loading and Preprocessing

Aim: To demonstrate data preprocessing on predefined Weka dataset labor.arff

Tasks:

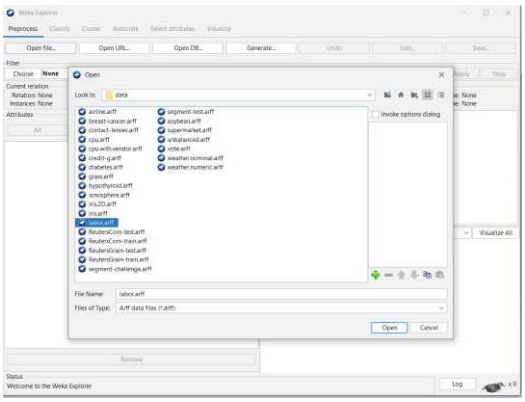
1. Load the labor.arff dataset and explore it.
2. Handle missing values if any
3. Perform Normalization/Standardization operations on numeric attributes.

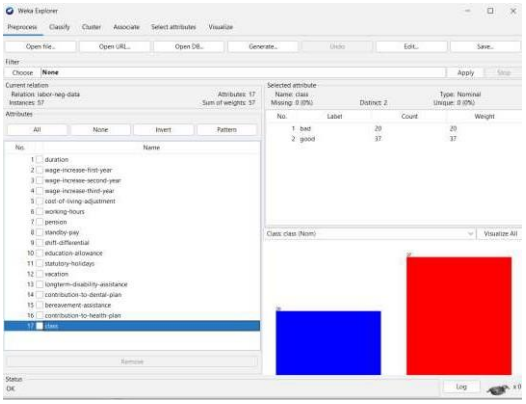
Task 1: Loading labor.arff dataset

labor.arff is built-in Weka's dataset available in its data folder.

Default path for data foalder is C:\Program Files\Weka-x-x-x\data.

Loading dataset: Explorer ➔ Preprocess ➔ Open file ➔ Weka's data folder ➔ labor.arff.





Observations:

Number of instances	
Number of attributes	
Target attribute	
Number of Target Classes	
Is it balanced dataset?	
Identify the type of attributes	
Any missing values?	

Task 2: Handling Missing Values

2.a) Removing instances with missing values

RemoveWithValues filter remove the instances that contain the given value in an attribute.

Steps:

I. Choose➔Filter➔Unsupervised➔instance➔RemoveWithValues

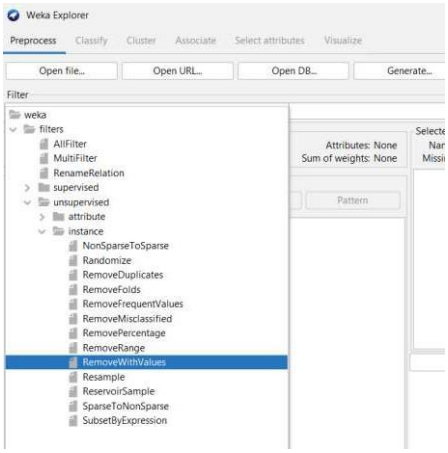
IL Click on selected filter to open its properties window.

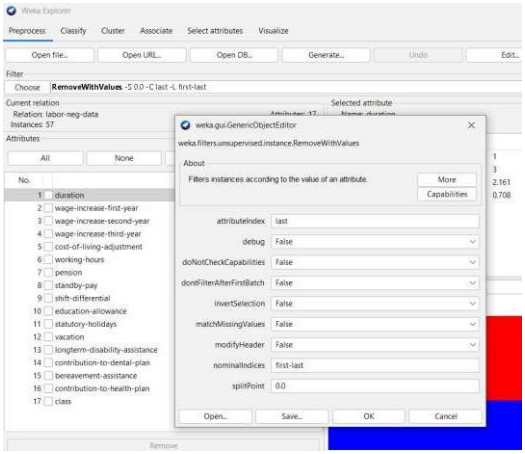
III. Enter the attribute index

IV. Select True for matchMissingValues

V. Click on ok

VI. Click on Apply





Observations:

Number of instances in the data set	
Attribute name and index selected	
Number of missing values in the selected attribute	
Number of missing values in the selected attribute	

2.b) Imputation with mean/mode values

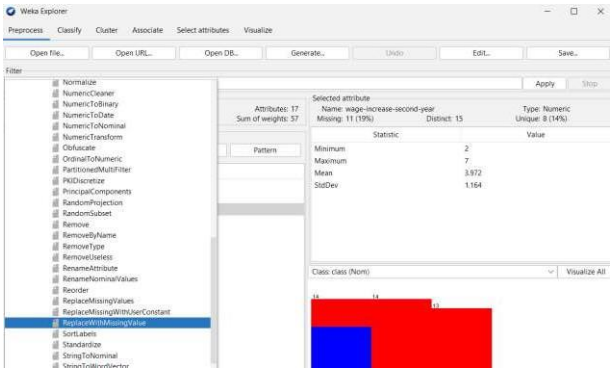
ReplaceMissingValues filter replace all missing values for nominal and numeric attributes in a dataset with the mode and mean respectively from the training data.

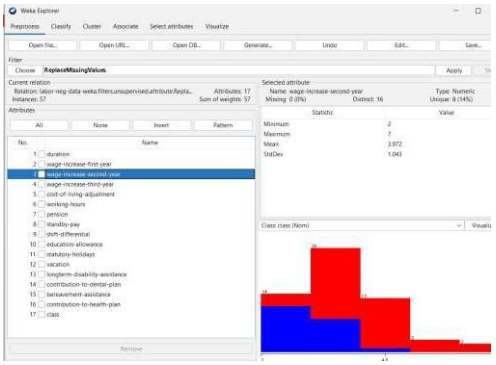
Steps:

I. Choose ➔ Filter ➔ Unsupervised ➔ Attribute ➔ ReplaceMissingValues

IL Change the properties, if required.

III. Click on Apply





Observations: Observe the missing values for all the attributes after applying the filter

Attribute	Type	Missing values are replaced with?

2.c) Imputation with User Constant

ReplaceMissingWithUserConstant filter fills the missing values with user defined constant for numeric/nominal/date attributes.

Steps:

I. Choose ➔ Filter ➔ Unsupervised ➔ Attribute ➔ ReplaceMissingWithUserConstant.

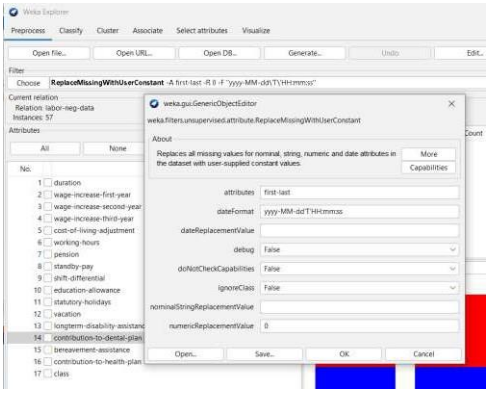
II. Open the filter properties by a click.

III. Select the attribute index

IV. Enter date/nominal/numericReplacementValue

V. Click Ok

VI. Click Apply



Observations: Note down your observations after applying filter to any attribute.

Task 3: Performing Normalization/Standardization to Numeric Attributes:

Normalization:

- Rescaling all the numeric attributes to the range Oto 1.
- Filter ➔ Unsupervised ➔ Attribute ➔ Normalize
- Default range [0, 1]
- Range can be changed to [-1, 1] by parameter values scale= 2 and translation= -1.

Standardization:

- Rescaling all the numeric attributes, such that they all have mean= 0 and SD= 1.
- Filter➔Unsupervised ➔Attribute➔Standardize

Observations:

Name of the Numeric Attribute	Actual Characteristics	Characteristics After Normalization	Characteristics After Standardization
	Min: Max: Mean: SD:	Min: Max: Mean: SD:	Min: Max: Mean: SD:
	Min: Max: Mean: SD:	Min: Max: Mean: SD:	Min: Max: Mean: SD:
	Min: Max: Mean: SD:	Min: Max: Mean: SD:	Min: Max: Mean: SD:

Conclusion:

Experiment 2

ARFF File Creation and Preprocessing

Aim: To create a student.arff dataset and Demonstrate Data Preprocessing on it

Tasks:

1. Create student.arff dataset.
2. Handle missing values if any.
3. Discretization of an attributes
4. Perform Normalization/Standardization operations on numeric attributes.

Task 1: Create student.arff dataset.

Steps:

i. Open any text editor (e.g. notepad)

ii. Using @relation, @attribute and @data tags enter the data in to the file with some missing values(?).

Attributes: Name, RollNo, Gender, Age, IQ (1 -10), score.

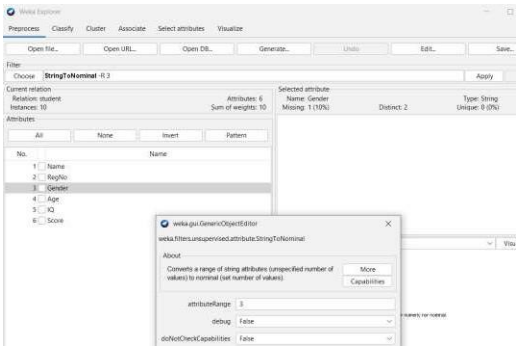
iii. Save the file as student.arff.

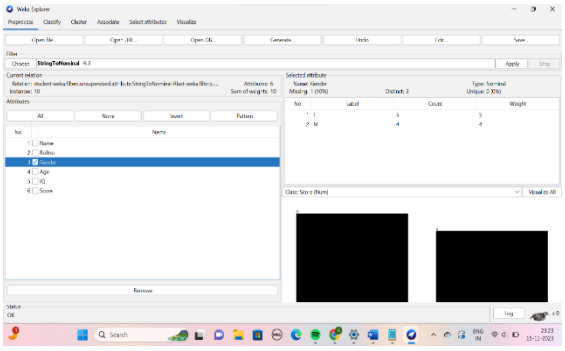
iv. Load student.arff in to Weka.

student.arff

```
@relation student
@attribute Name string
@attribute RegNo string
@attribute Gender string
@attribute Age numeric
@attribute IQ numeric
@attribute Score numeric
@data
chandu,21A81A05B4,M,20,7,80
charan,21A81A05B3,M,20,8,68
surya,21A81A05A7,F,20,?,8
Siddiq,21A81A05B0,M,40,7,80
sasi,21A81A05A8,?,39,5,53
likith,21A81A05A1,F,33,4,76
?,21A81A05A0,M,9,9,84
Bala,21A81A05C7,F,?,5,53
venkatesh,21A81A0577,F,33,6,78
rahul,21A81A0578,M,39,5,62
```

We can use *Filter* ➔ *Unsupervised* ➔ *Attribute* ➔ *StringToNominal* to convert the attribute type from string to nominal.





Note: We can make Gender attribute nominal while file creation

@attribute Gender {M, F}

Task 2: Handling Missing Values

Use *Filter* ➔ *Unsupervised* ➔ *Attribute* ➔ *ReplaceMissingValues* filter to fill missing numeric attribute values with mean and nominal values with mode

To fill the missing values in the class(target - default is last) attribute, make ignoreClass parameter True.

Observation: Observe the missing values for all the attributes after applying the filter.

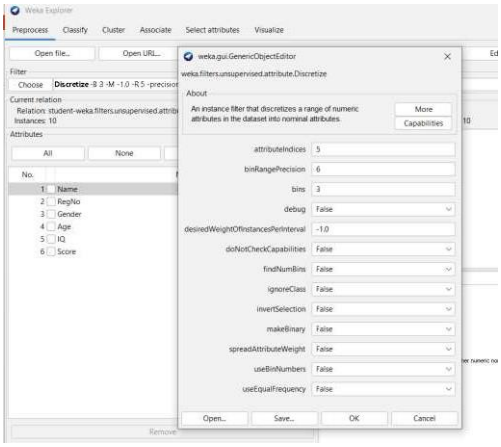
Attribute	Type	Missing values are replaced with?

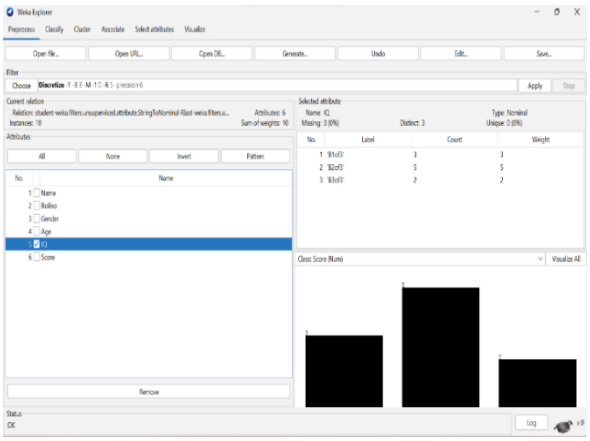
Task 3: Discretizing an attribute:

Converting a numeric attribute to a nominal attribute.

Steps:

1. Select *Filter* ➔ *Unsupervised* ➔ *Attribute* ➔ *Discretize* filter.
11. Open the filter's object editor.
111. Enter the attributeIndeces to be discretized.
1v. Enter the bins you want for the attribute.
v. Click ok
v1. Click Apply





We can use useBinNumbers parameter to change the labels to Bxofy format.

Observations:

Numeric Attribute	Number of Bins	Bin-wise Instances Count

Task 4: Performing Normalization/Standardization to Numeric Attributes:

Normalization:

Rescaling all the numeric attributes to the range 0 to 1.

Filter➔Unsupervised➔Attribute➔Normalize

Default range [0, 1]

Range can be changed to [-1, 1] by parameter values scale = 2 and translation = -1.

Standardization:

Rescaling all the numeric attributes, such that they all have mean = 0 and SD = 1.

Filter ➔ Unsupervised ➔ Attribute ➔ Standardize

Observations:

Name of the Numeric Attribute	**Actual Characteristics**	**Characteristics After Normalization**	**Characteristics After Standardization**
	Min: Max: Mean: SD:	Min: Max: Mean: SD:	Min: Max: Mean: SD:
	Min: Max: Mean: SD:	Min: Max: Mean: SD:	Min: Max: Mean: SD:

Conclusion:

Experiment 3

Association Rule Mining on contact-lenses Dataset

Aim: To demonstrate Association rule mining process on predefined Weka dataset contact-lenses.arff using Apriori algorithm.

Tasks:

1. Load contact-lenses.arff dataset and explore it.
2. Apply Apriori algorithm with default parameters.
3. Change the parameters and observe the results

Task 1: Load contact-lenses.ariff dataset and explore it.

Go to Explorer ➔ OpenFile ➔ Browse C:/Program Files/Weks-x-x-x/data ➔ open contact-lenses.arff file.

Observations:

Attribute	States and their meaning
age	1. 11. 111.
spectacle-prescript	1. 11.
astigmatism	1. 11.
tear-prod-rate	1. 11.
contact-lenses	1. 11. 111.

Task 2: Apply Apriori algorithm with default parameters

Association Rule Mining is a process that finds features which occur together or features that are correlated. Popular applications are Market Basket Analysis and Cross Marketing.

Association rules are mined out after frequent itemsets in a big dataset which can be found using algorithms such as Apriori and FP Growth.

Frequent Itemset mining mines data using support and confidence measures.

$$support\ (A \Rightarrow B) = \frac{number\ of\ instances\ containing\ both\ A\ and\ B}{total\ number\ of\ instances}$$

$$confidence\ (A \Rightarrow B) = p(B/A) = \frac{number\ of\ instances\ containing\ both\ A\ and\ B}{number\ of\ instances\ containing\ A}$$

Apriori Rule Learner in Weka implements Apriori algorithm. It iteratively reduces the minimum support from its upperBound until (i) it finds the required number of rules or the minimum support reaches lowerBound.

Default values of the some important parameters:

lowerBoundMinSupport = 0.1 (10%)

upperBoundMinSupport = 1.0 (100%)

metricType = Confidence

minMetric = 0.9 (90%)

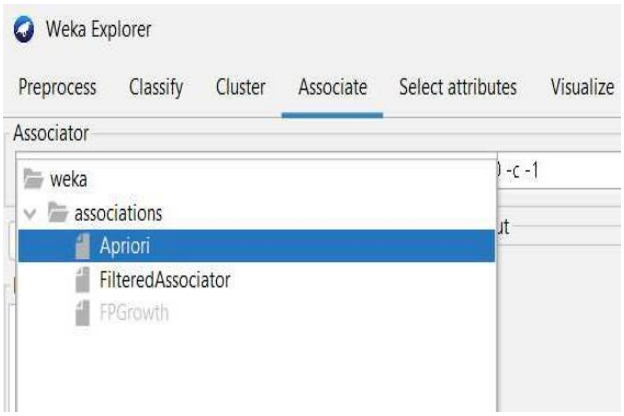
numRules = 10

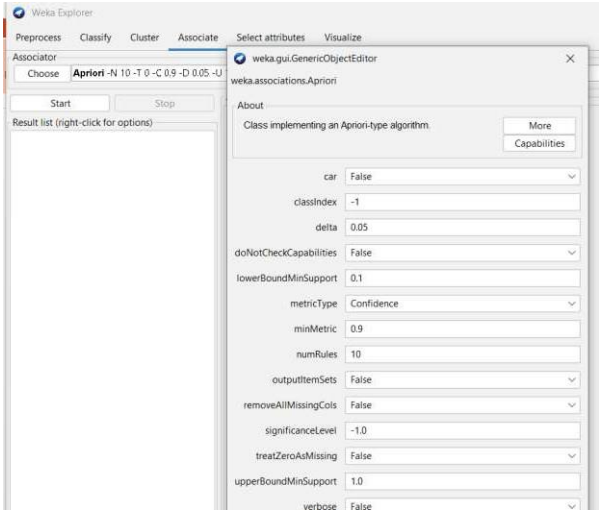
Steps:

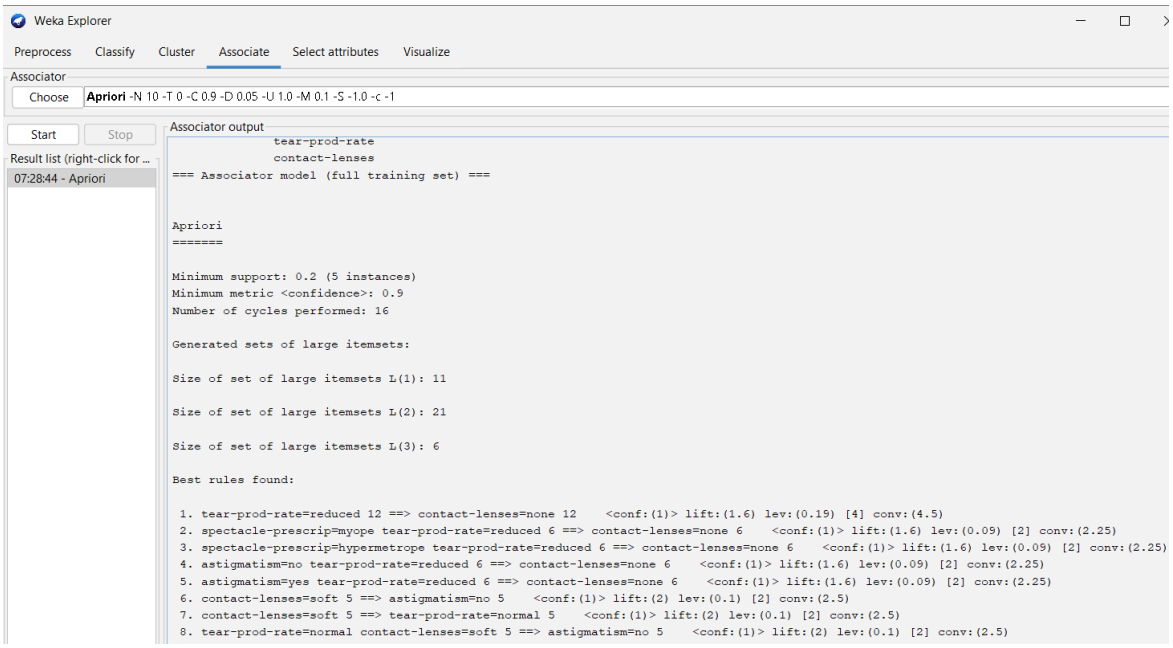
I. Select Associate ➔ Choose ➔ associations ➔ Apriori.

II. Observe the default parameter values.

III. Click on Start.







Default Parameters	**Observations**
lowerBoundMinSupport = upperBoundMinSupport = metricType = minMetric = numRules =	Minimum support = Minimum Metric <Confidence> = Number of cycles performed = Best rules found with confidence: 1. tear-prod-rate=reduced 12 ==> contact-lenses=none 12 <conf:(1)> lift:(1.6) lev:(0.19) [4] conv:(4.5) 2. spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25) 3. spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25) 4. astigmatism=no tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25) 5. astigmatism=yes tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25) 6. contact-lenses=soft 5 ==> astigmatism=no 5 <conf:(1)> lift:(2)

	lev:(0.1) [2] conv:(2.5) 7. contact-lenses=soft 5 ==> tear-prod-rate=normal 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5) 8. tear-prod-rate=normal contact-lenses=soft 5 ==> astigmatism=no 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5) 9. astigmatism=no contact-lenses=soft 5 ==> tear-prod-rate=normal 5 <conf:(1)> lift:(2) lev:(0.1) [2] conv:(2.5) 10. contact-lenses=soft 5 ==> astigmatism=no tear-prod-rate=normal 5 <conf:(1)> lift:(4) lev:(0.16) [3] conv:(3.75).

Task 3: Apply Apriori algorithm with required parameters

Observations: Change the default parameter values and perform the experiment

Parameters	Observations
lowerBoundMinSupport = upperBoundMinSupport = metricType = minMetric = numRules =	Minimum support = Minimum Metric <Confidence> = Number of cycles performed = Best rules found with confidence: 1. tear-prod-rate=reduced 12 ==> contact-lenses=none 12 <conf:(1)> lift:(1.6) lev:(0.19) [4] conv:(4.5) 2. spectacle-prescrip=myope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25) 3. spectacle-prescrip=hypermetrope tear-prod-rate=reduced 6 ==> contact-lenses=none 6 <conf:(1)> lift:(1.6) lev:(0.09) [2] conv:(2.25)

Conclusion:

Experiment 4

Association Rule Mining on Employee Dataset

Aim: To create an employee.arff dataset and demonstrate Association rule process on it using apriori algorithm

Tasks:

1. Create employee.arff dataset and load it into Weka.
2. Apply Apriori algorithm with default parameters.
3. Change the parameters and observe the results

Task 1: Create employee.arff dataset and load it.

Create employee.arff with following categorical attributes and load it in to Weka.

Attribute	States
Designation	i. Manager ii. Developer iii. Tester
Beneficiary	i. Yes ii. No
GPF	i. Yes ii. No
Salary	i. Low ii. Medium iii. High
CreditRating	i. Poor ii. Fair iii. Excellent
BankLoan	i. Yes ii. No

```
@relation Employee
@attribute Designation {Manager, Developer, Tester}
@attribute Beneficiary {yes,no}
@attribute GPF {yes, no}
@attribute Salary {low, medium, high}
@attribute CreditRating {poor, fair, good}
@attribute BankLoan {yes, no}
@data
Developer, yes, yes, medium, fair, no
Tester,no, no, high, good, no
Manager, yes, no, medium, fair, no
Developer, yes, no, high, fair, yes
Developer, yes, no, high, fair,no
Developer, yes, yes, high, fair, yes
Manager, yes, no, low, poor,no
Manager, yes, yes, medium, fair,yes
Manager, yes, no, high, good, no
Developer, no, no, medium, good, no
Tester, yes, no, medium, fair, no
Developer, yes, no, medium, fair, yes
```

Task 2: Apply Apriori algorithm with default parameters

Association Rule Mining is a process that finds features which occur together or features that are correlated. Popular applications are Market Basket Analysis and Cross Marketing.

Association rules are mined out after frequent itemsets in a big dataset which can be found using algorithms such as Apriori and FP Growth.

Frequent Itemset mining mines data using support and confidence measures.

$$support\ (A \Rightarrow B) = \frac{number\ of\ instances\ containing\ both\ A\ and\ B}{total\ number\ of\ instances}$$

$$confidence\ (A \Rightarrow B) = p(B/A) = \frac{number\ of\ instances\ containing\ both\ A\ and\ B}{number\ of\ instances\ containing\ A}$$

Apiori Rule Leamer in Weka implements Apriori algorithm. It iteratively reduces the minimum support from its upperBound until (i) it finds the required number of rules or the minimum support reaches lowerBound.

Default values of the some important parameters:

lowerBoundMinSupport = 0.1 (10%)

upperBoundMinSupport = 1.0 (100%)

metricType = Confidence

minMetric = 0.9 (90%)

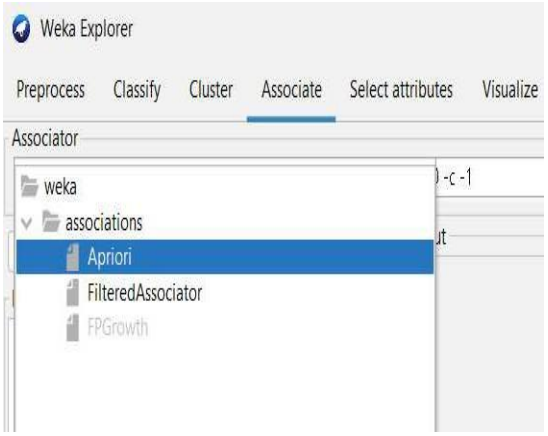
numRules = 10

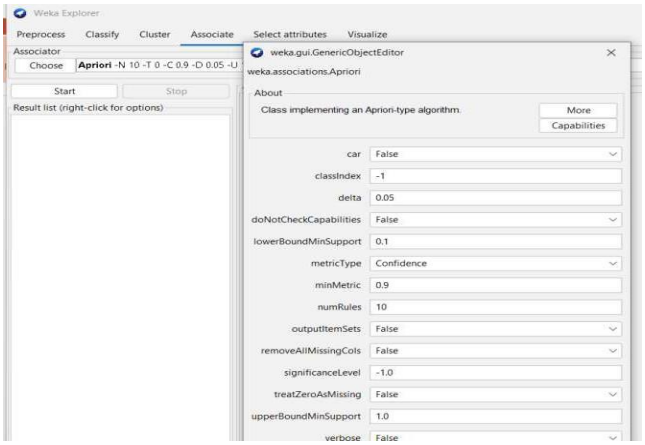
Steps:

I. Select Associate ➔ Choose ➔ associations ➔ Apriori.

II. Observe the default parameter values.

III. Click on Start.





Observations:

Default Parameters	Observations
lowerBoundMinSupport = upperBoundMinSupport = metricType = minMetric = numRules =	Minimum support = Minimum Metric <Confidence> = Number of cycles performed = Best rules found with confidence: 1. CreditRating=good 5 ==> GPF=no 5 <conf:(1)> lift:(1.25) lev:(0.07) [0] conv:(1) 2. CreditRating=good 5 ==> BankLoan=no 5 <conf:(1)> lift:(1.36) lev:(0.09) [1] conv:(1.33) 3. Designation=developer CreditRating=fair 5 ==> Beneficiary=yes 5 <conf:(1)> lift:(1.36) lev:(0.09) [1] conv:(1.33) 4. Salay=medium CreditRating=fair 5 ==> Beneficiary=yes 5 <conf:(1)> lift:(1.36) lev:(0.09) [1] conv:(1.33) 5. CreditRating=good BankLoan=no 5 ==> GPF=no 5 <conf:(1)> lift:(1.25) lev:(0.07) [0] conv:(1) 6. GPF=no CreditRating=good 5 ==> BankLoan=no 5 <conf:(1)> lift:(1.36) lev:(0.09) [1] conv:(1.33) 7. CreditRating=good 5 ==> GPF=no BankLoan=no 5 <conf:(1)> lift:(1.5) lev:(0.11) [1] conv:(1.67) 8. Designation=manager 4 ==> Beneficiary=yes 4 <conf:(1)> lift:(1.36) lev:(0.07) [1] conv:(1.07)

Task 3: Apply Apriori algorithm with required parameters

Observations: Change the default parameter values and perform the experiment

Parameters	Observations
lowerBoundMinSupport =	Minimum support =
upperBoundMinSupport =	Minimum Metric <Confidence> =
metricType =	Number of cycles performed =
minMetric =	Best rules found with confidence:
numRules =	1. BankLoan=no 11 ==> GPF=no 10 <conf:(0.91)> lift:(1.14) lev:(0.08) [1] conv:(1.1) 2. CreditRating=fair 9 ==> Beneficiary=yes 8 <conf:(0.89)> lift:(1.21) lev:(0.09) [1] conv:(1.2) 3. GPF=no 12 ==> BankLoan=no 10 <conf:(0.83)> lift:(1.14) lev:(0.08) [1] conv:(1.07)

Conclusion:

Experiment 5

Classification Using J48

Aim: To demonstrate Classification process on iris.arff dataset using j48 algorithm with percentage split.

Tasks:

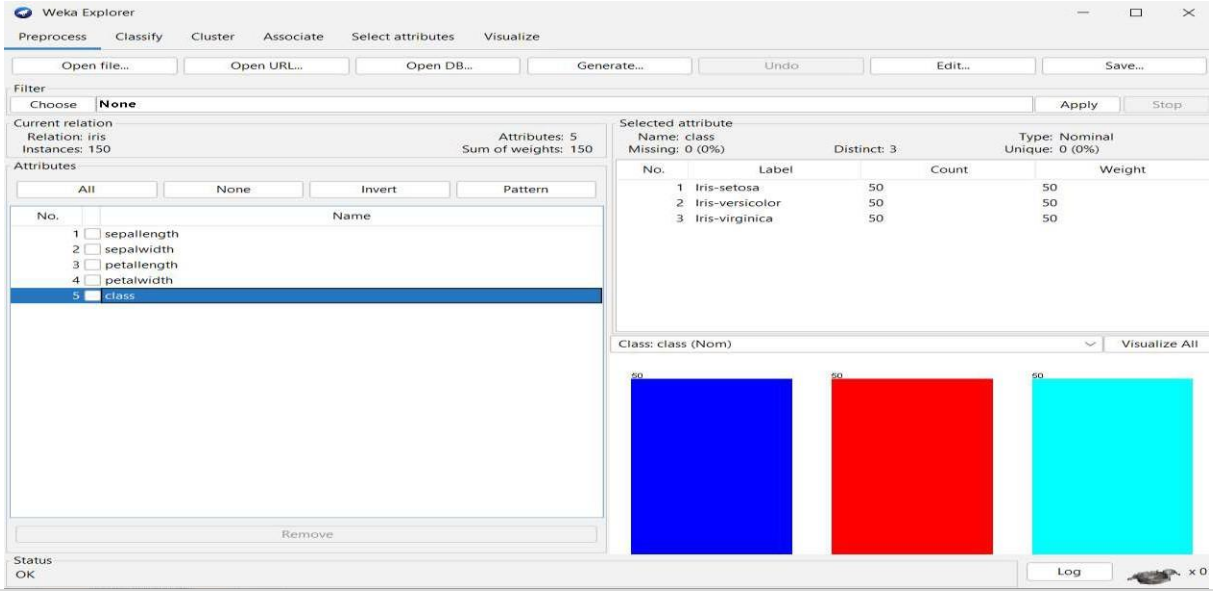
1. Load iris.arff dataset and explore it.
2. Build a classification model using J48 algorithm with percentage split.
3. Make predictions on new data.

Task 1: Load iris.arff dataset and explore it.

Load iris.arff from the Weka's data folder.

Observations:

Attribute	**Type**	**Range / States**
Number of Instances:		



Task 2: Build a classification model using 148 algorithm with percentage split.

Classification is a process of determining the class (state) of the given instance. Examples:

Determining Play or Not play based on weather conditions.

Determining the digit (0 - 9) given the image pixel data.

Determining the Spam or Not-spam based on mail text.

148 is Weka's Java implementation of the C4.5 algorithm. It can generate pruned or unpruned tress with both nominal and numerical attributes for classification.

Percentage Split:

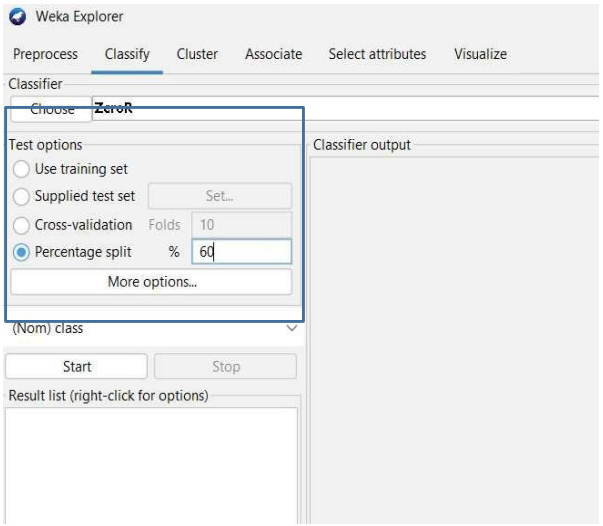
Splits the data into training and test subsets.

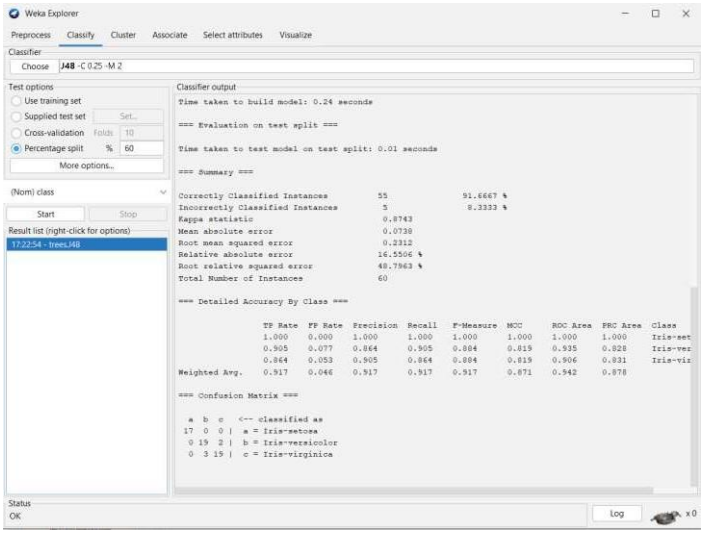
Training set is used to build the model.

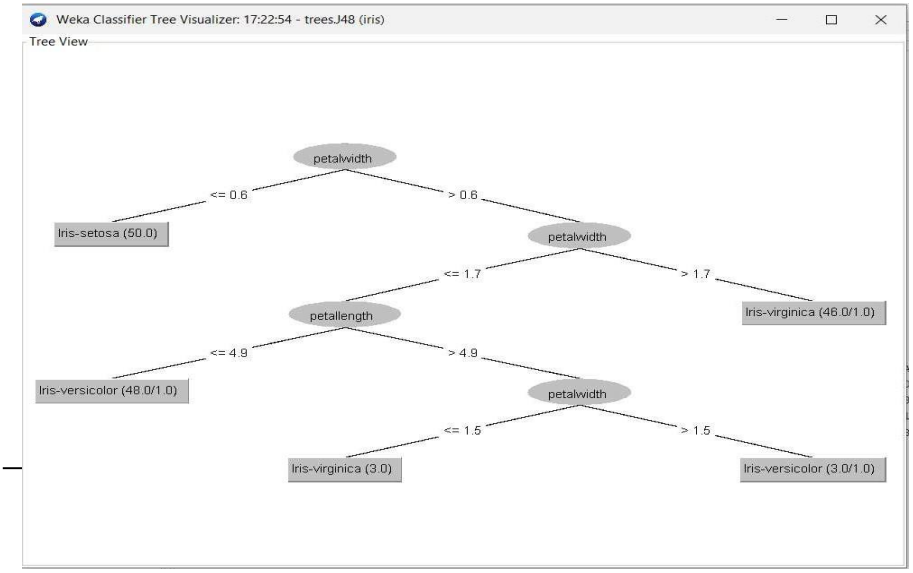
Test set is used to evaluate the model performance.

Steps to build the model:

1. Click on Classify and select Percentage split with required training percentage under Test options group.
2. Select Choose ➔ classifiers ➔ trees ➔ 148.
3. Clock on Start.
4. Right click on the model and click on Visualize tree







Observations:

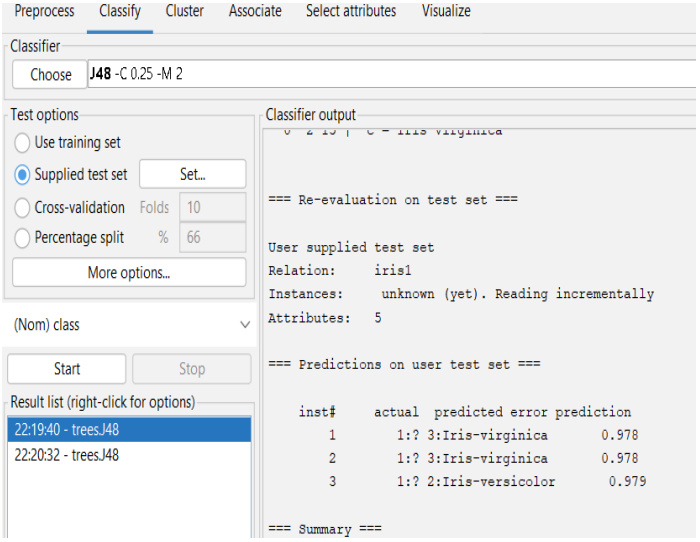
- Total number of instances:
- Correctly classified instances:
- Incorrectly classified instances:
- Accuracy:
- Calculation of Accuracy from Confusion Matrix:

Task 3: Make predictions on new data

Steps:

1. Create an ARFF file with unlabeled (use ? in the place of class label) instances.
11. On the "Classify" tab, select the "Supplied test set" option in the "Test options" pane.
111. Click the "Set" button, click the "Open file" button on the options window and select the new dataset.
1v. Click the "More options... " button and for the "Output predictions" option click the "Choose" button and select "PlainText".
v. Right click on the model in the "Results list" pane and Select "Re-evaluate model on current test set".

```
@relation iris1
@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute petalwidth numeric
@attribute class {Iris-setosa,Iris-versicolor,Iris-virginica}
@data
4.5,3.0,4.1,1.0,?
5.6,4.2,6.4,2.5,?
7.8,4.4,1.5,2.2,?
```



Observations:

Instance No.	Predicted class

Conclusion:

Experiment 6

Classification Using J48

Aim: To demonstrate Classification process on StudentResult.arff dataset using j48 algorithm with cross-validation

Tasks:

1. Create StudentResult.arff dataset and load it into Weka.
2. Build a classification model using J48 algorithm with k-fold cross validation.
3. Make predictions on new data.

Task 1: Create StudentResult.arff dataset and load it into Weka.

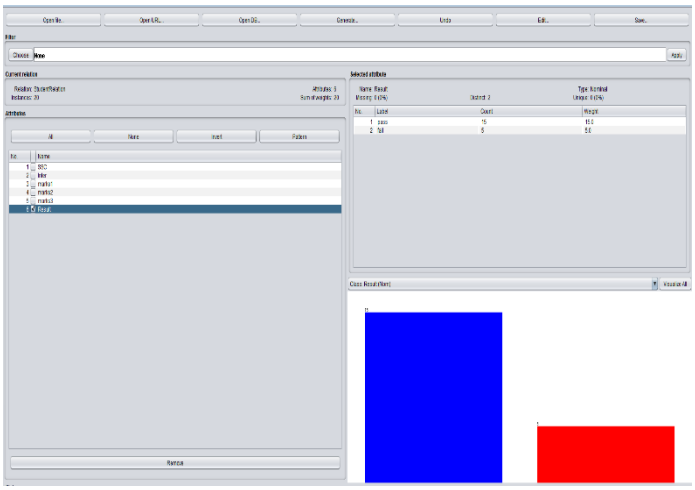
Create StudentResults.arff with following attributes and load it into Weka

Attribute	Type
SSC	Nominal with states First, Second, Third.
Inter	Nominal with states First, Second, Third.
Marksl	Numeric
Marks2	Numeric
Marks3	Numeric
Result	Nominal with states Pass & Fail.

StudentRelation.arff

```
@relation StudentRelation
@attribute SSC {first,second,third}
@attribute Inter {first,second,third}
@attribute marks1 numeric
@attribute marks2 numeric
@attribute marks3 numeric
@attribute Result {pass, fail}

@data
first, second, 55, 57, 62 pass
first, first, 63, 63, 55, pass
first, first, 65, 69, 66, pass
first, first, 76, 77, 82, pass
second, third, 32, 43, 23, fail
first, first, 67, 76, 57, pass
second,second,56,54,45,pass
Second, second, 56, 65,57, pass
third, third, 34,23,12, fail
Second, third, 23,34, 23, fail
Second, first, 65, 64, 56, pass
first, first, 65, 66, 67, pass
third, third, 45,82,23, fail
first, first, 56, 63, 73, pass
second,first,65,56, 57, pass
second,first,77,79, 80, pass
second,third,45,68, 35, fail
third,second,65,54, 67, pass
first,first,78,79,73,pass
second,first,78,77,76,pass
```



Task 2: Build a classification model using J48 algorithm with k-fold cross validation Classification is a process of determining the class(state) of the given instance.

Examples:

Determining Play or Not play based on weather conditions.

Determining the digit (0 - 9) given the image pixel data.

Determining the Spam or Not-spam based on mail text.

J48 is Weka's Java implementation of the C4.5 algorithm. It can generate pruned or unpruned tress with both nominal and numerical attributes for classification.

K-fold Cross Validation is a resampling procedure used to evaluate data mining models on a limited data set. It's process is

I. Split the input dataset into K groups

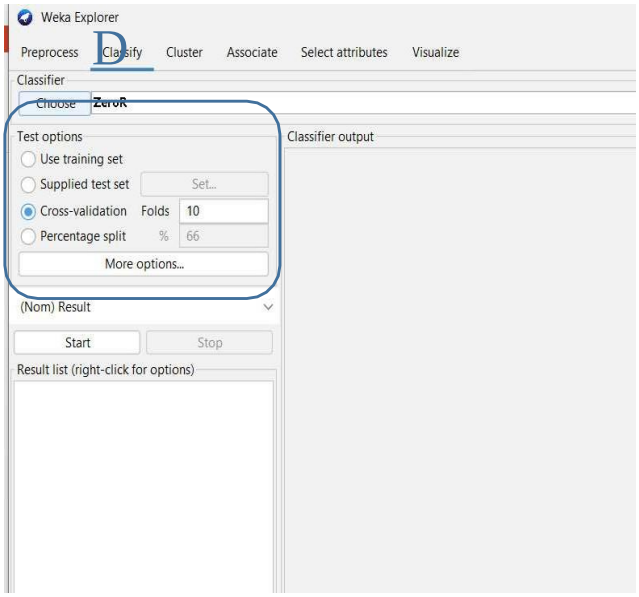
IL For i from 1 to k

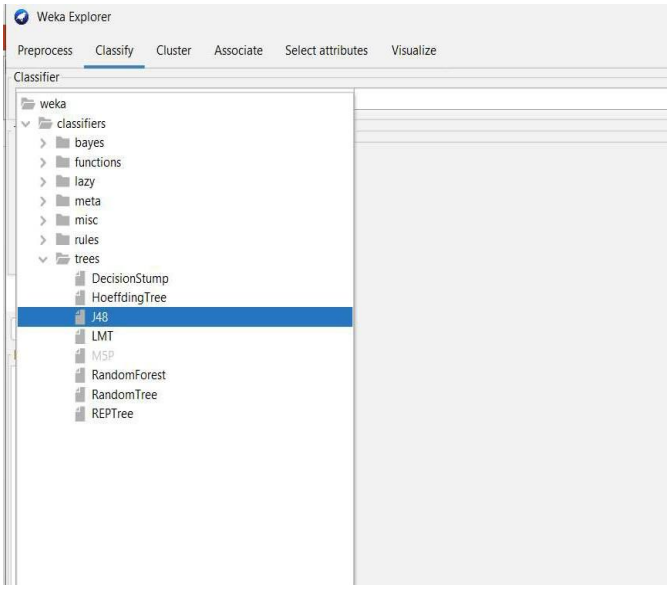
- Take i^{th} group as test dataset.

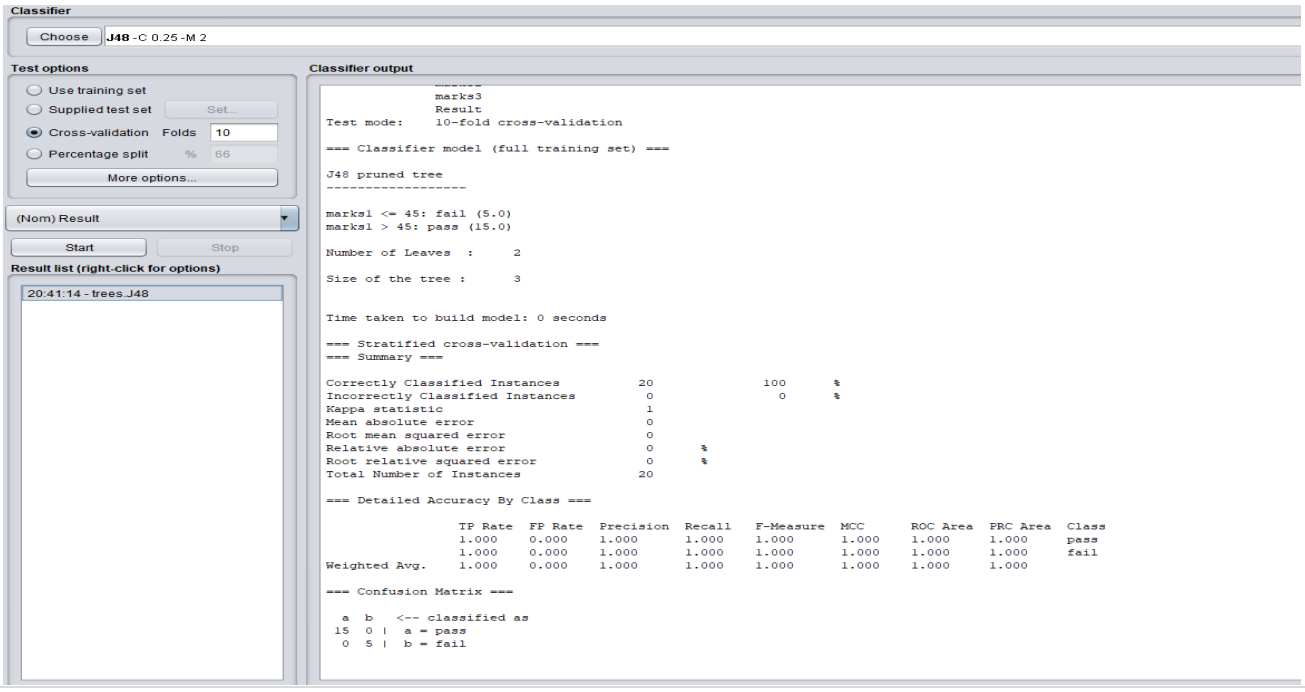
- Use remaining K-1 groups as training dataset.
- Fit the model using training set and evaluate its performance on test set.

Steps to build the model:

1. Click on Classify and select Cross-validation with default 10 folds under Test options group.
2. Select Choose ➔ classifiers ➔ trees ➔ J48.
3. Clock on Start.
4. Right click on the model and click on Visualize tree



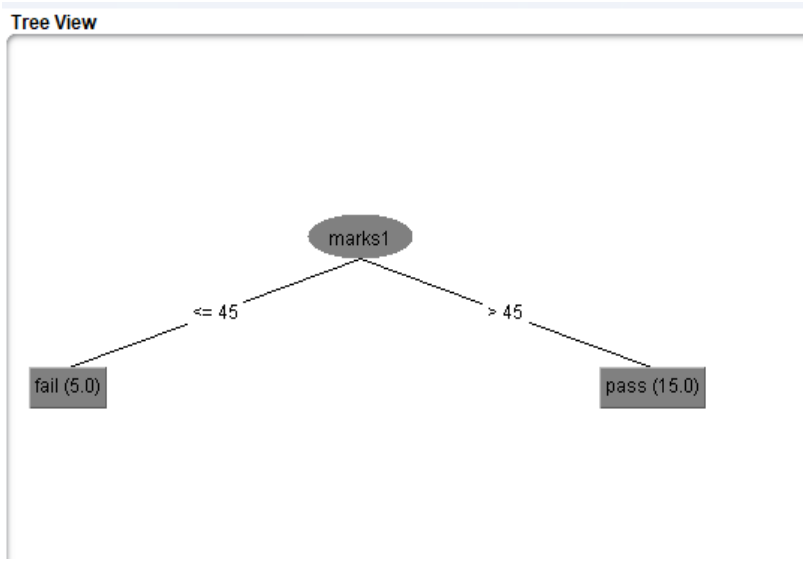




Observations:

- Total number of instances:
- Correctly classified instances:
- Incorrectly classified instances:
- Accuracy:
- Calculation of Accuracy from Confusion Matrix:

- Tree generated by J48 algorithm for the given dataset is:



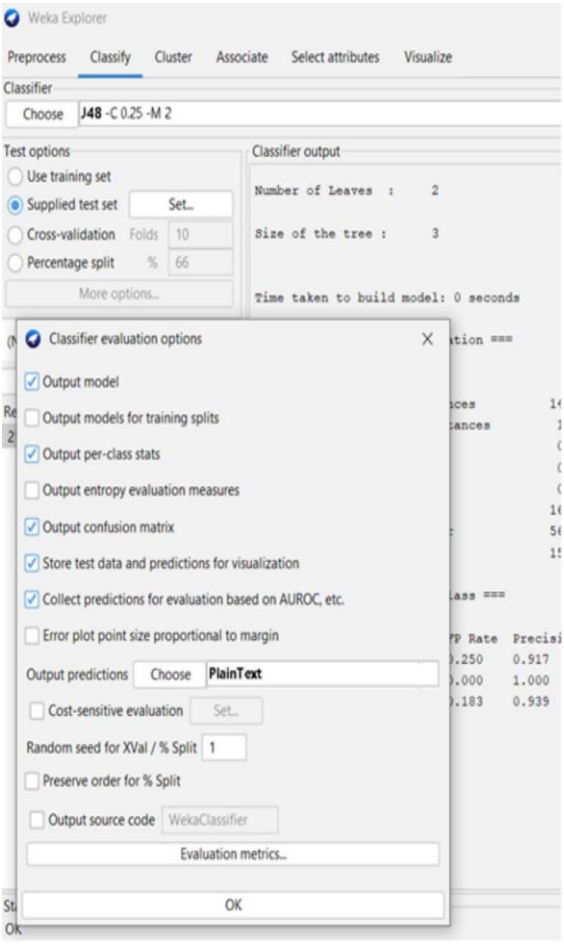
Task 3: Make predictions on new data

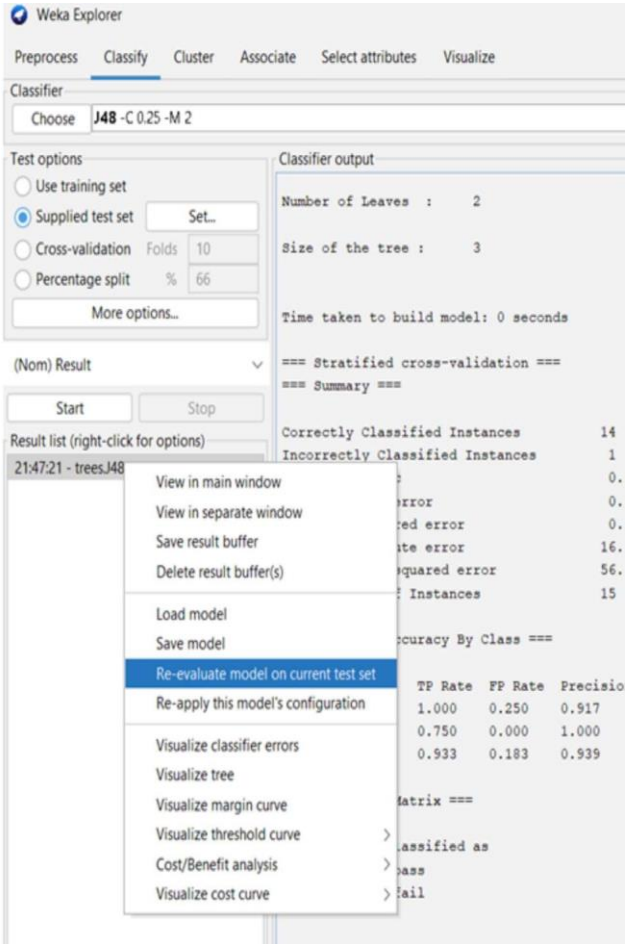
Steps:

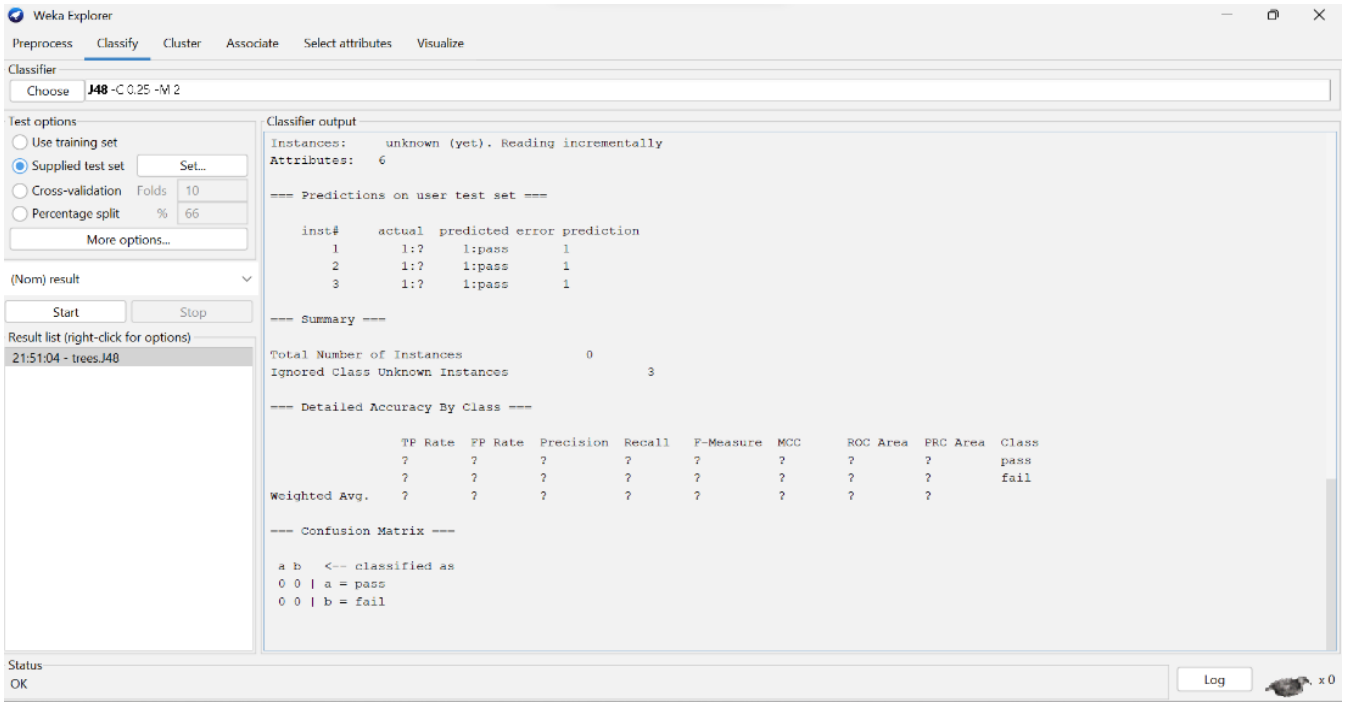
1. Create an ARFF file with unlabeled (use? in the place of class label) instances.
11. On the "Classify" tab, select the "Supplied test set" option in the "Test options" pane.
111. Click the "Set" button, click the "Open file" button on the options window and select the new dataset.
1v. Click the "More options... " button and for the "Output predictions" option click the "Choose" button and select "PlainText".
v. Right click on the model in the "Results list" pane and Select "Re-evaluate model on current test set".

StudentRelationl.arff

```
@relation StudentRelation
@attribute SSC {first,second,third}
@attribute Inter {first,second,third}
@attribute marks1 numeric
@attribute marks2 numeric
@attribute marks3 numeric
@attribute Result {pass, fail}
@data
first,first,76,67,75,?
second,third,32,43,32,?
first,first,65,36,37,?
```







Observations:

Instance No.	Predicted class

Conclusion:

Experiment 7

Classification Using ID3

Aim: To demonstrate Classification process on contact-lenses.arff dataset using ID3 algorithm with cross-validation.

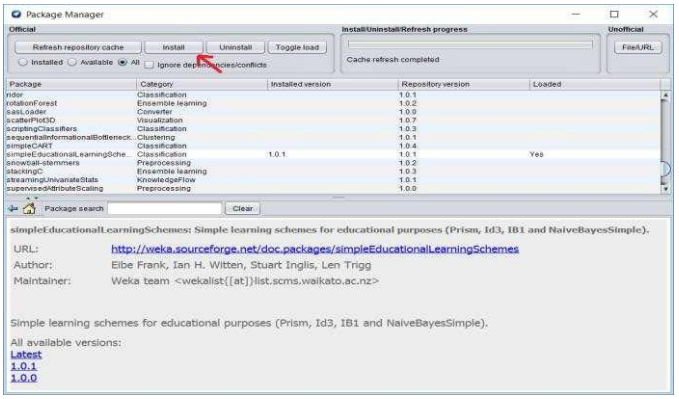
Tasks:

1. Install required package for ID3
2. Load contact-lenses.arff dataset
3. **Build** a classification model using ID3 algorithm with k-fold cross-validation.
4. Make predictions on new data.

Task 1: Install required package for ID3

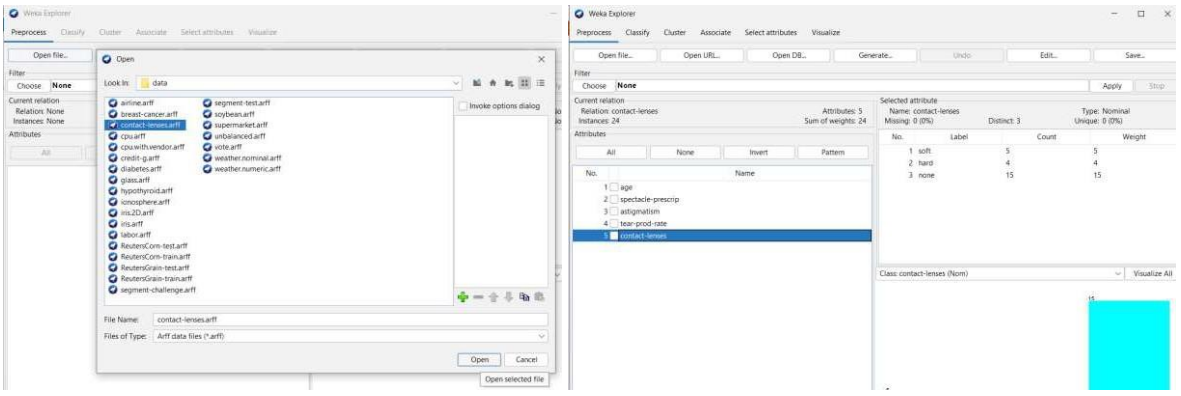
Select GUI Chooser➔Tools➔Package Manager

Search, select & install *simpleEducationalLearningSchemes* package to get ID3 classifier under trees group.



Task 2: Load contact-lenses.arff dataset

Load contact-lenses.arff from the Weka's data folder.



Task 3: Build a classification model using ID3 algorithm with k-fold cross validation.

Classification is a process of determining the class (state) of the given instance.

ID3 stands for Iterative Dichotomiser 3 and is named such because the algorithm iteratively (repeatedly) dichotomizes(divides) features into two or more groups at each step. ID3 uses a top-down greedy approach to build a decision tree. ID3 is only used for classification problems with nominal features only.

K-fold Cross Validation is a resampling procedure used to evaluate data mining models on a limited data set. It's process is

I. Split the input dataset into K groups

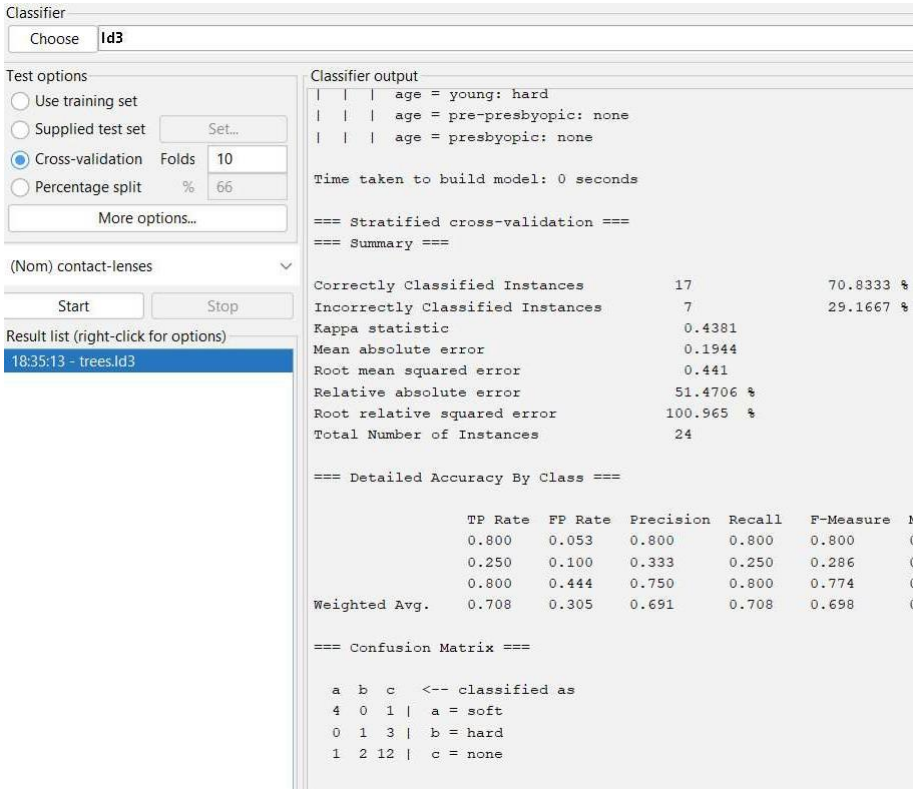
IL For i from 1 to k

- Take i^{th} group as test dataset.
- Use remaining K-1 groups as training dataset.
- Fit the model using training set and evaluate its performance on test set.

Steps to build the model:

1. Click on Classify and select Cross-validation with some number folds under Test options group.
2. Select Choose ➔ classifiers ➔ trees ➔ ID3.
3. Clock on Start.

   Note that we can't visualize ID3 tree.



Observations:

- Total number of instances:
- Correctly classified instances:
- Incorrectly classified instances:
- Accuracy:
- Calculation of Accuracy from Confusion Matrix:

Task 4: Make predictions on new data

Steps:

1. Create an ARFF file with unlabeled (use? in the place of class label) instances.

11. On the "Classify" tab, select the "Supplied test set" option in the "Test options"

pane.

m. Click the "Set" button, click the "Open file" button on the options window and select the new dataset.

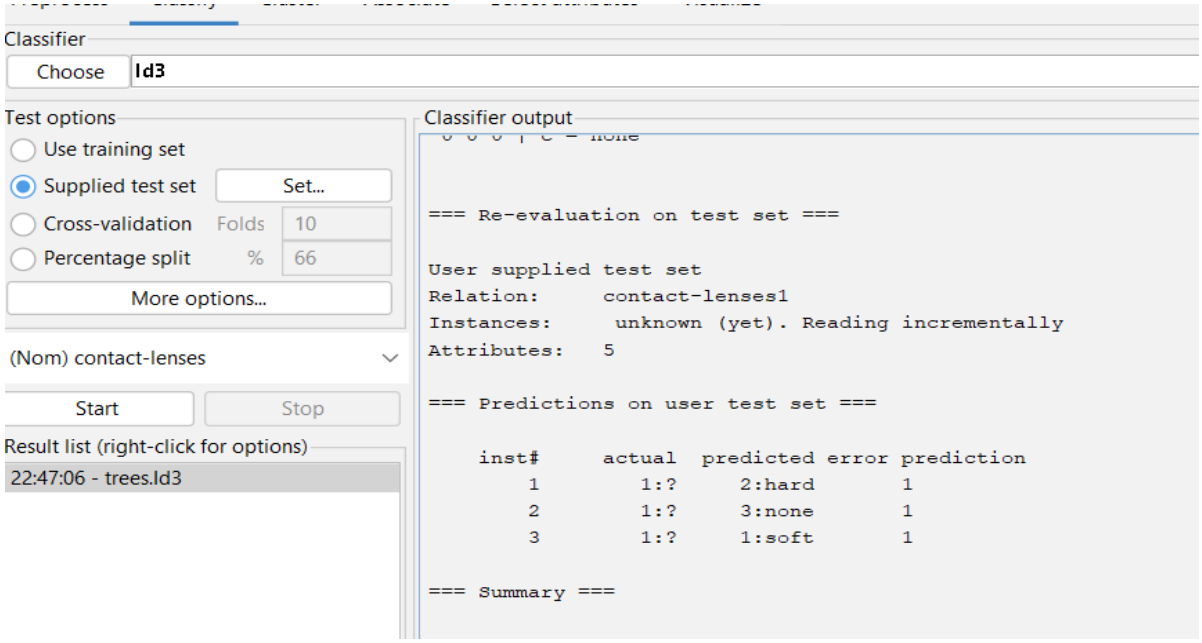
1v. Click the "More options... " button and for the "Output predictions" option click the "Choose" button and select "PlainText".

v. Right click on the model in the "Results list" pane and Select "Re-evaluate model on current test set".

contact-lensesl.arff

```
@relation contact-lenses!
@attribute age {young, pre-presbyopic, presbyopic}
@attribute spectacle-prescrip {myope, hypermetrope}
@attribute astigmatism {no, yes}
@attribute tear-prod-rate {reduced, normal}
@attribute contact-lenses {soft, hard, none}

@data
Pre-presbyopic,myope,no,normal,?
presbyopic,hypermetrope,no,reduced,?
young,myope,yes,normal,?
```



Observations:

Instance No.	Predicted class

Conclusion:

Experiment 8

Classification Using Na'ive Bayes

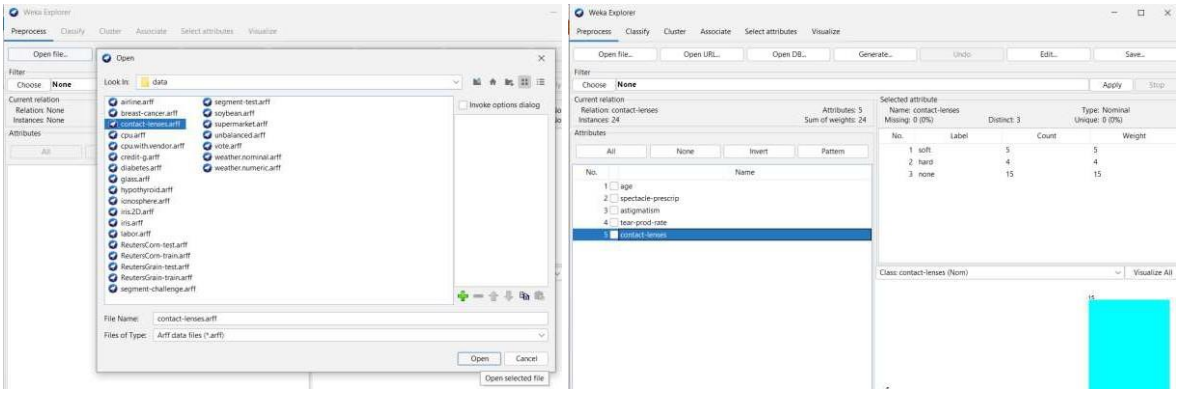
Aim: To demonstrate Classification process on contact-lenses.arff dataset using Na'ive Bayes algorithm with cross-validation.

Tasks:

1. Load contact-lenses.arff dataset
2. Build a classification model using Na'ive Bayes algorithm with k-fold cross-validation.
3. Make predictions on new data.

Task 1: Load contact-lenses.arff dataset

Load contact-lenses.arff from the Weka's data folder.



Task 2: Build a classification model using Na'ive Bayes algorithm with k-fold cross validation.

Classification is a process of determining the class (state) of the given instance.

Naive Bayes classifier is a popular machine learning algorithm based on Bayes' theorem. It is a probabilistic algorithm used for classification tasks. The "naive" in its name comes from the assumption that the features used to describe an instance are mutually independent, given the class label.

Assume the instances with n attributes and class label y E {ci, $c_{2, \ldots,}$ ed. Then the prediction $y = \arg\max_{ci}\{P(cdX)\}$. Where $P(cdX) =$ TTJ=l $P(xjlci)$-

K-fold Cross Validation is a resampling procedure used to evaluate data mining models on a limited data set. It's process is

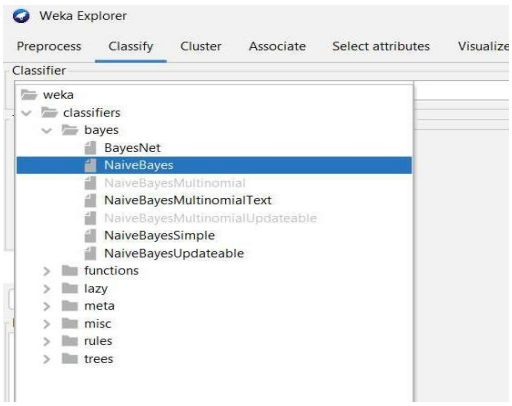
I. Split the input dataset into K groups

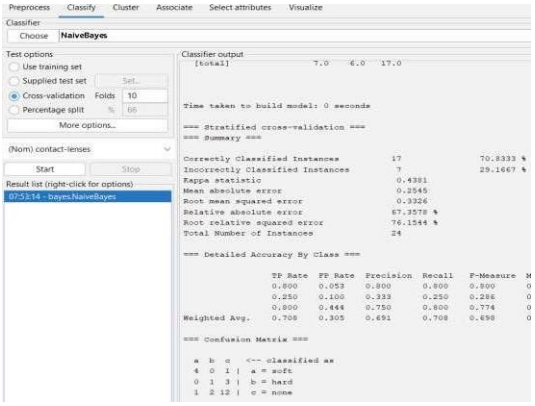
II. For i from 1 to k

- Take i^{th} group as test dataset.
- Use remaining K-1 groups as training dataset.
- Fit the model using training set and evaluate its performance on test set.

Steps to build the model:

1. Click on Classify and select Cross-validation with some number folds under Test options group.
2. Select Choose ➔ classifiers ➔ bayes ➔ Nai:ve Bayes.
3. Clock on Start.





Observations:

- Total number of instances:
- Correctly classified instances:
- Incorrectly classified instances:
- Accuracy:
- Calculation of Accuracy from Confusion Matrix:

Task 3: Make predictions on new data

Steps:

1. Create an ARFF file with unlabeled (use ? in the place of class label) instances.
11. On the "Classify" tab, select the "Supplied test set" option in the "Test options" pane.
111. Click the "Set" button, click the "Open file" button on the options window and select the new dataset.

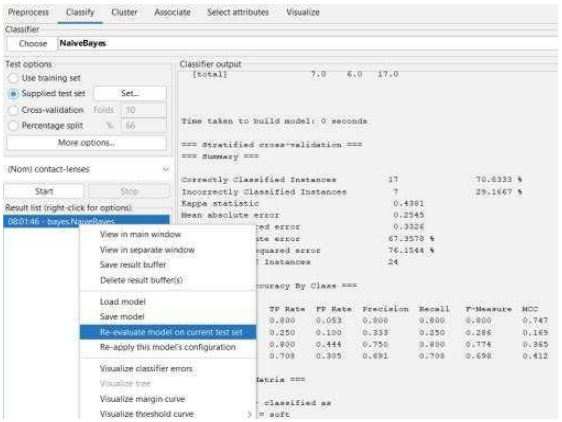
1v. Click the "More options... " button and for the "Output predictions" option click the "Choose" button and select "PlainText".

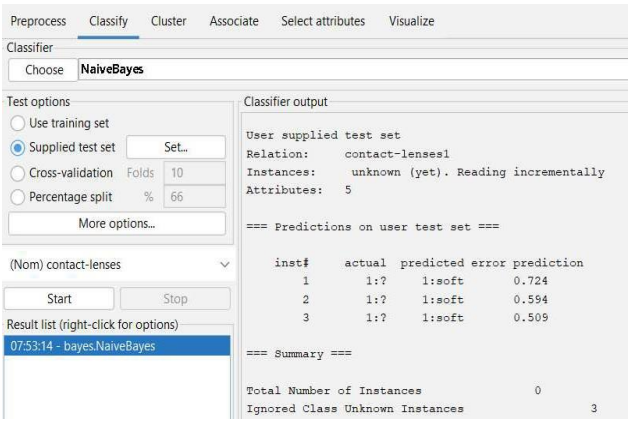
v. Right click on the model in the "Results list" pane and Select "Re-evaluate model on current test set".

Contact-lensesl.arff

```
@relation contact-lenses1
@attribute age {young, pre-presbyopic, presbyopic}
@attribute spectacle-prescrip {myope, hypermetrope}
@attribute astigmatism {no, yes}
@attribute tear-prod-rate {reduced, normal}
@attribute contact-lenses {soft, hard, none}

@data
Pre-presbyopic,myope,no,normal,?
presbyopic,hypermetrope,no,reduced,?
young,myope,no,normal,?
```





Observations:

Instance No.	Predicted class

Conclusion:

Experiment 9

K-Means Clustering

Aim: To demonstrate Clustering process on iris.arff dataset using K-means algorithm.

Tasks:

1. Load iris.arff dataset and remove target attribute.
2. Cluster the dataset using k-means algorithm.
3. Visualize the results.

Task 1: Load iris.arff dataset and remove target attribute.

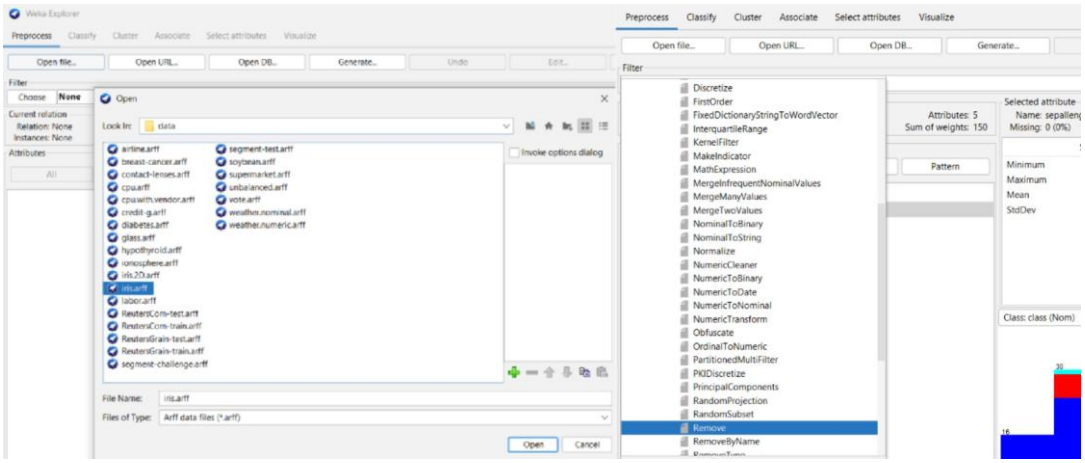
Load iris.arff from the Weka's data folder.

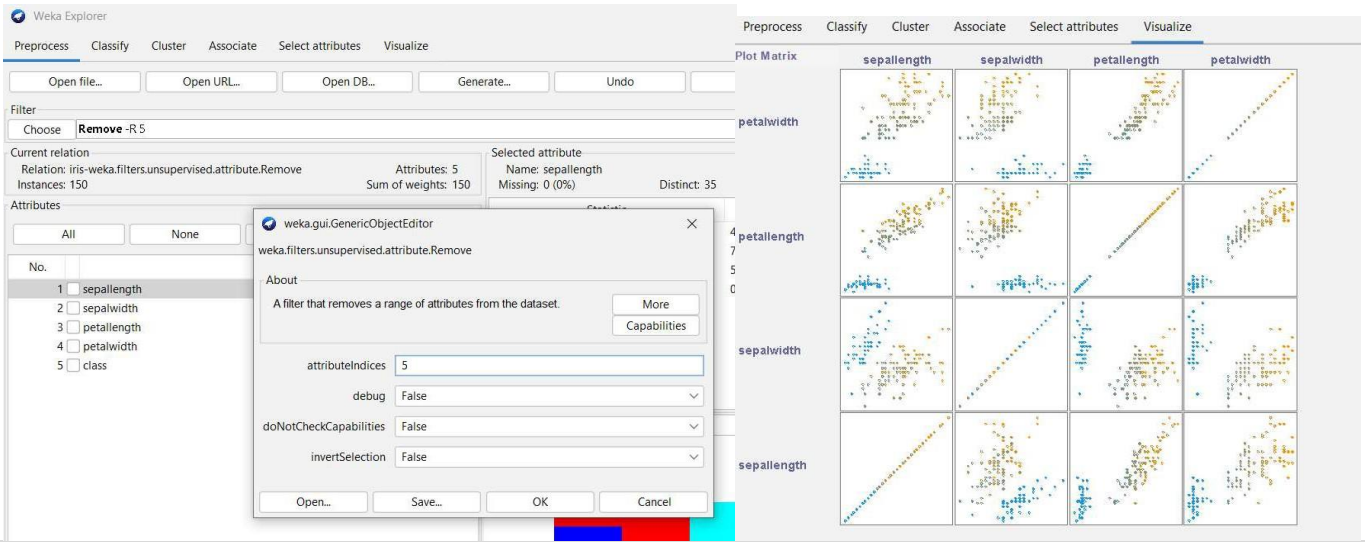
Select Choose ➔ filter ➔ unsupervised ➔ attribute ➔ Remove.

Click on filter and enter target attribute index.

Click on Apply.

Click on Visualize to observe the number of clusters that dataset may have.





Task 2: Cluster the dataset using k-means algorithm.

A clustering is an unsupervised learning technique to find groups of similar instances in the entire dataset.

K-means clustering:

I. Randomly K initial cluster centers are chosen.

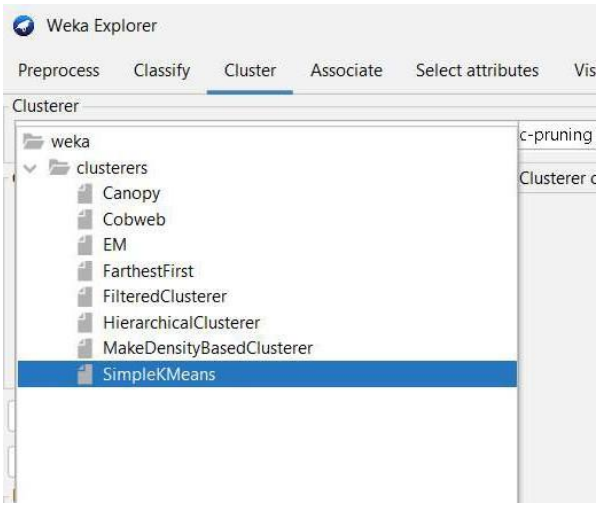
II. All instances are assigned to closest cluster centers according to the Euclidean distance measure.

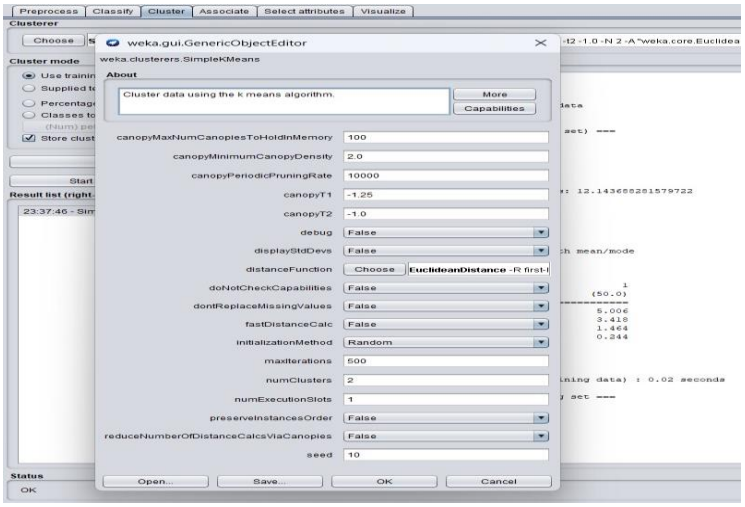
III. Cluster centers are updated based on the mean values of all instances in their respective clusters.

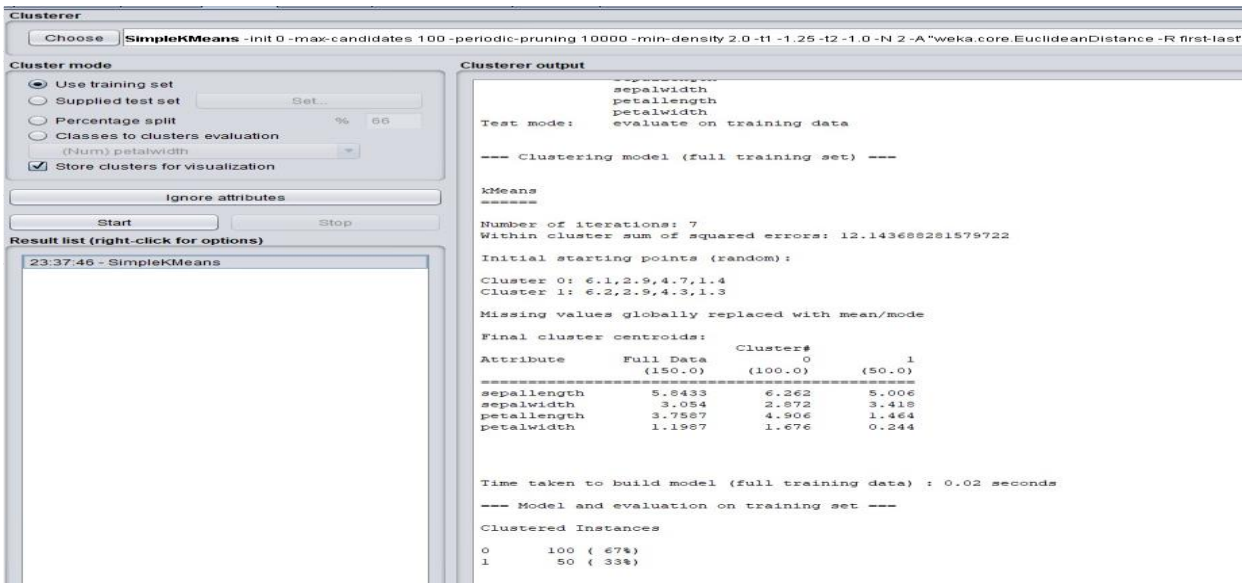
IV. Steps ii and iii are repeated until cluster centers are stabilized.

Steps:

- Open Explorer➔Cluster window.
- Select the *Use training set* option in *Cluster mode* pane.
- Check the *Store clusters for visualization* option.
- Select Choose ➔ Clusters ➔ SimpleKMeans.
- Click on SimpleKMeans and set the number of clusters.
- Click on Start.







Observations:

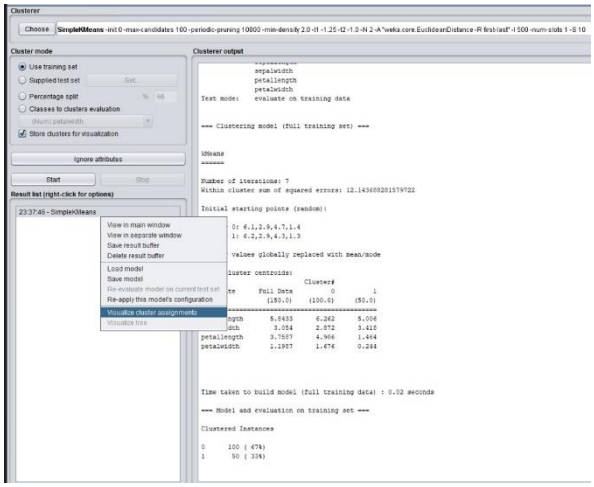
1. Number oflterations:
2. Starting random cluster centroids:

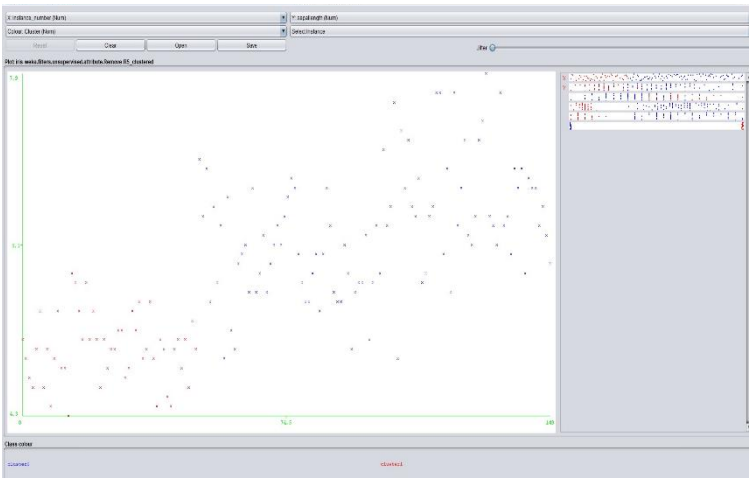
3. Final cluster centroids:

4. Number of instances in each cluster:

Task 3: Visualizing the clusters

- Right click on the clustering model in Result list.
- Click on Visualize cluster assignment.
- Save the result in arff format.





Iris clustered.arff

```
@relation iris-weka.filters.unsupervised.
attribute.Remove-R5_clustered

@attribute Instance_number numeric
@attribute sepallength numeric
@attribute sepalwidth numeric
@attribute petallength numeric
@attribute petalwidth numeric
@attribute Cluster {cluster0,cluster1}

@data
0,5.1,3.5,1.4,0.2,cluster1
1,4.9,3,1.4,0.2,cluster1
2,4.7,3.2,1.3,0.2,cluster1
3,4.6,3.1,1.5,0.2,cluster1
4,5,3.6,1.4,0.2,cluster1
5,5.4,3.9,1.7,0.4,cluster1
6,4.6,3.4,1.4,0.3,cluster1
7,5,3.4,1.5,0.2,cluster1
8,4.4,2.9,1.4,0.2,cluster1
9,4.9,3.1,1.5,0.1,cluster1
10,5.4,3.7,1.5,0.2,cluster1
11,4.8,3.4,1.6,0.2,cluster1
12,4.8,3,1.4,0.1,cluster1
13,4.3,3,1.1,0.1,cluster1
14,5.8,4,1.2,0.2,cluster1
15,5.7,4.4,1.5,0.4,cluster1
16,5.4,3.9,1.3,0.4,cluster1
17,5.1,3.5,1.4,0.3,cluster1
18,5.7,3.8,1.7,0.3,cluster1
19,5.1,3.8,1.5,0.3,cluster1
20,5.4,3.4,1.7,0.2,cluster1
```

Conclusion:

Experiment 10

Clustering an User's Dataset using K-Means algorithm

Aim: To demonstrate Clustering process on user dataset using K-means algorithm.

Tasks:

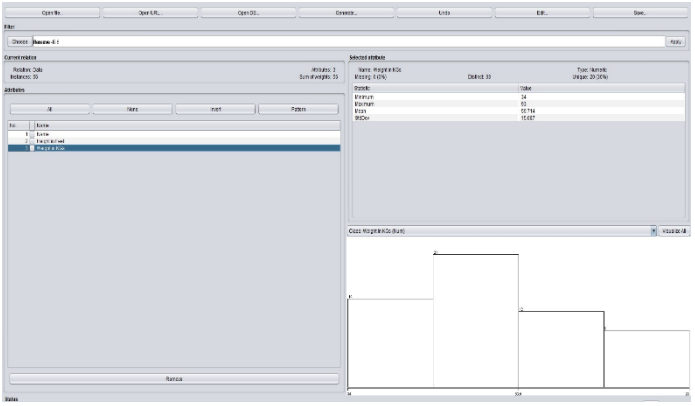
1. Load B-data Responses.csv dataset and remove target attribute.
2. Cluster the dataset using k-means algorithm.
3. Visualize the results.

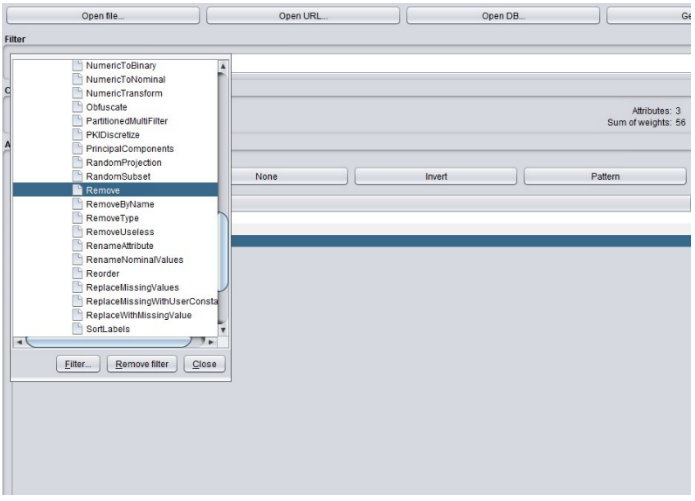
Task 1: Load data.csv dataset and remove target attribute.

- Load data.csv from the device.
- Select Choose ➔ filter ➔ unsupervised ➔ attribute ➔ Remove.
- Click on filter and enter target attribute index.
- Click on Apply.
- Click on Visualize to observe the number of clusters that dataset may have.

Data.csv

```
Name,Height in Feet,Weight in KGs
Sunny ,5.11,76
Abc,5.4,53
Gvs,5.4,54
J Narasimha,6.1,93
Ram babu,5.7,92
madhu,5.2,54
Tejaswi ,5.1,55
Satya Pavan,5.5,90
Siddiq,5.7,83
Naidu,5.5,80
Anuu,5.5,58
Likhith,5.9,65
Sherlock Holmes,5.6,62
Virat,5.9,69
Kakashi ,6,80
Ammu,5,38
Sita,5,65
Abhi,5.7,55
Nazriya,5,45
Paddu,5.3,55
Tarak,5.7,56
Nandhu,5.9,68
Rohith,5.8,85
bhavya,5.1,41
Dhanush,5.7,71
Hasini,5.3,52
Varshini ,5.2,50
Erisa,5,34
```





Task 2: Cluster the dataset using k-means algorithm.

A clustering is an unsupervised learning technique to find groups of similar instances in the entire dataset.

K-means clustering:

I. Randomly K initial cluster centers are chosen.

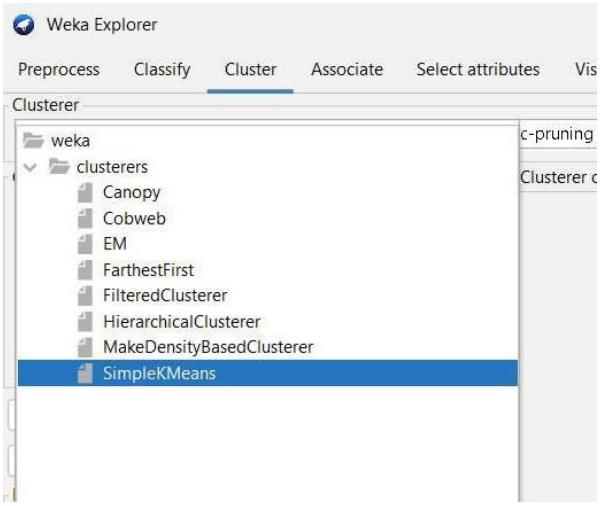
II. All instances are assigned to closest cluster centers according to the Euclidean distance measure.

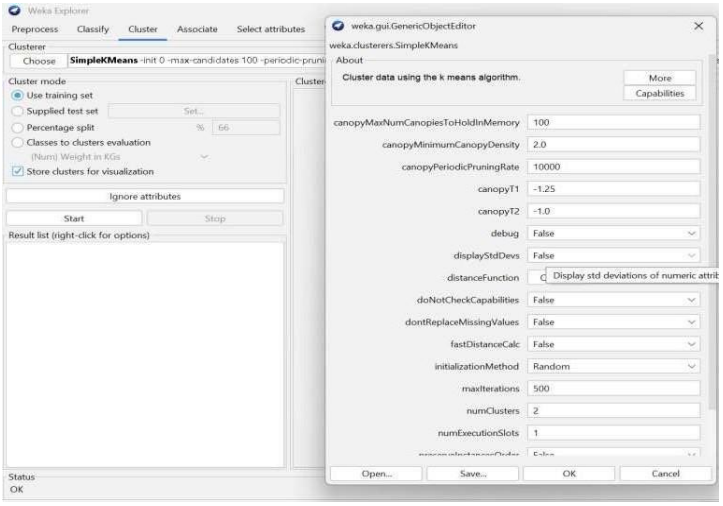
III. Cluster centers are updated based on the mean values of all instances in their respective clusters.

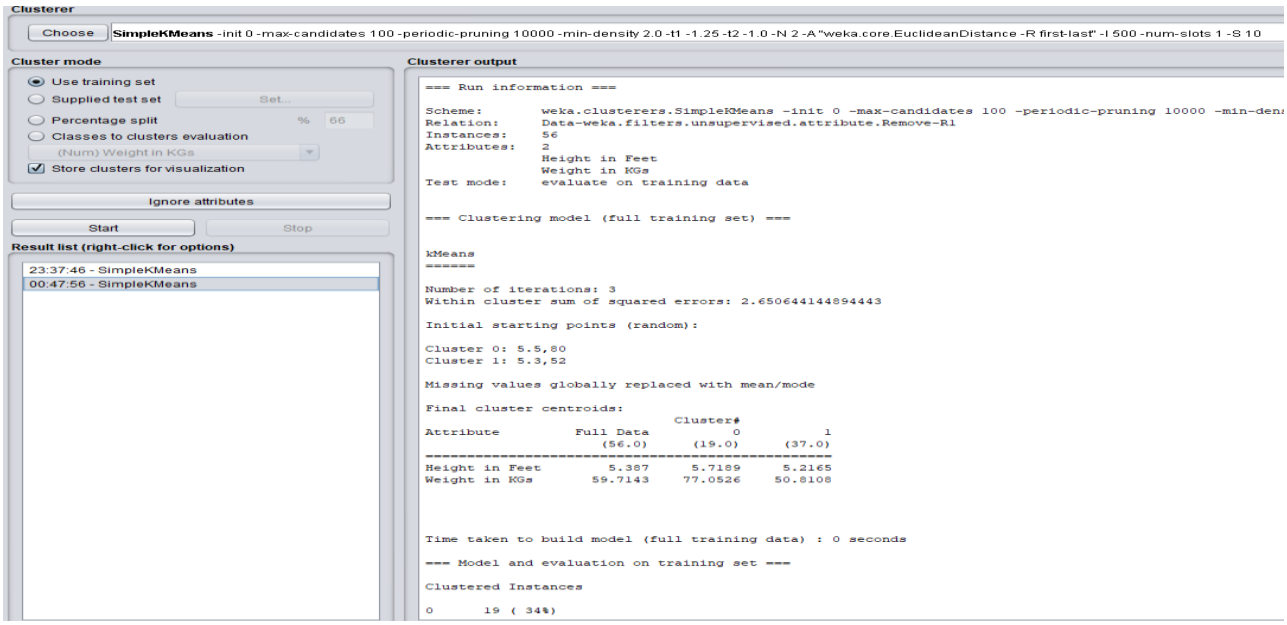
IV. Steps ii and iii are repeated until cluster centers are stabilized.

Steps:

- Open Explorer ➔ Cluster window.
- Select the *Use training set* option in *Cluster mode* pane.
- Check the *Store clusters for visualization* option.
- Select Choose ➔ Clusters ➔ SimpleKMeans.
- Click on SimpleKMeans and set the number of clusters.
- Click on Start.







Observations:

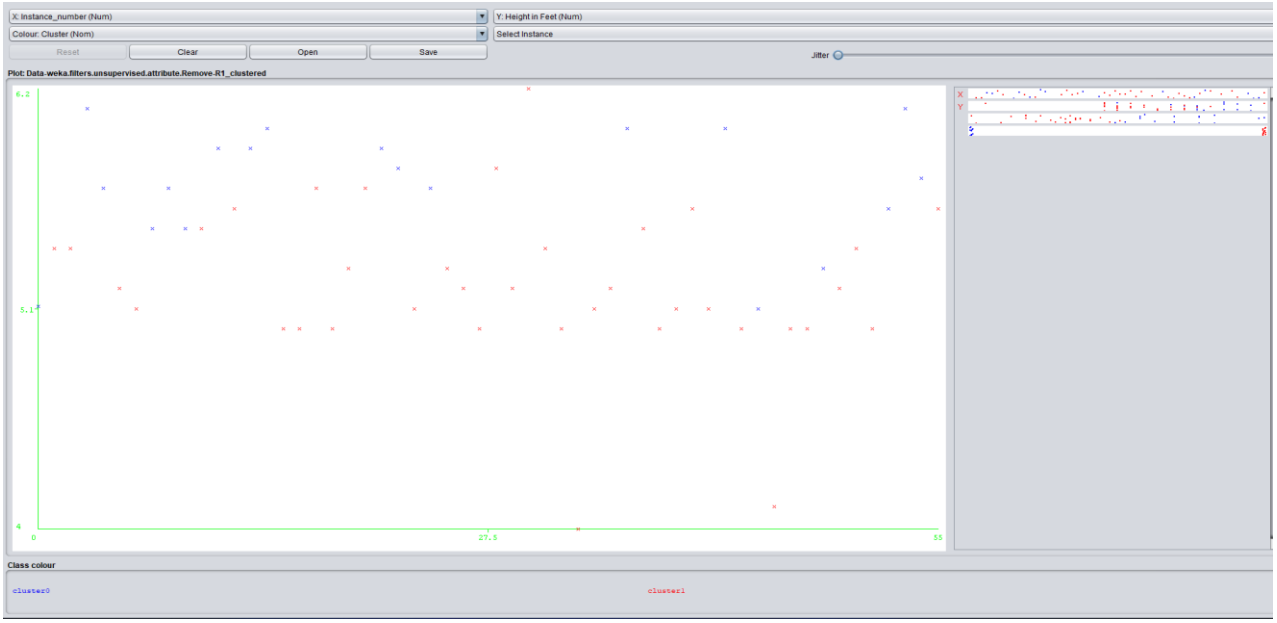
1. Number oflterations:
2. Starting random cluster centroids:

3. Final cluster centroids:

4. Number of instances in each cluster:

Task 3: Visualizing the clusters

- Right click on the clustering model in Result list.
- Click on Visualize cluster assignment.
- Save the result in arff format.



Expt 10.arff

```
@relation Data-weka.filters.unsupervised.attribute
.Remove-R1_clustered

@attribute Instance_number numeric
@attribute 'Height in Feet' numeric
@attribute 'Weight in KGs' numeric
@attribute Cluster {cluster0,cluster1}

@data
0,5.11,76,cluster0
1,5.4,53,cluster1
2,5.4,54,cluster1
3,6.1,93,cluster0
4,5.7,92,cluster0
5,5.2,54,cluster1
6,5.1,55,cluster1
7,5.5,90,cluster0
8,5.7,83,cluster0
9,5.5,80,cluster0
10,5.5,58,cluster1
11,5.9,65,cluster0
12,5.6,62,cluster1
13,5.9,69,cluster0
14,6,80,cluster0
15,5,38,cluster1
16,5,65,cluster1
17,5.7,55,cluster1
18,5,45,cluster1
19,5.3,55,cluster1
20,5.7,56,cluster1
```

Conclusion: