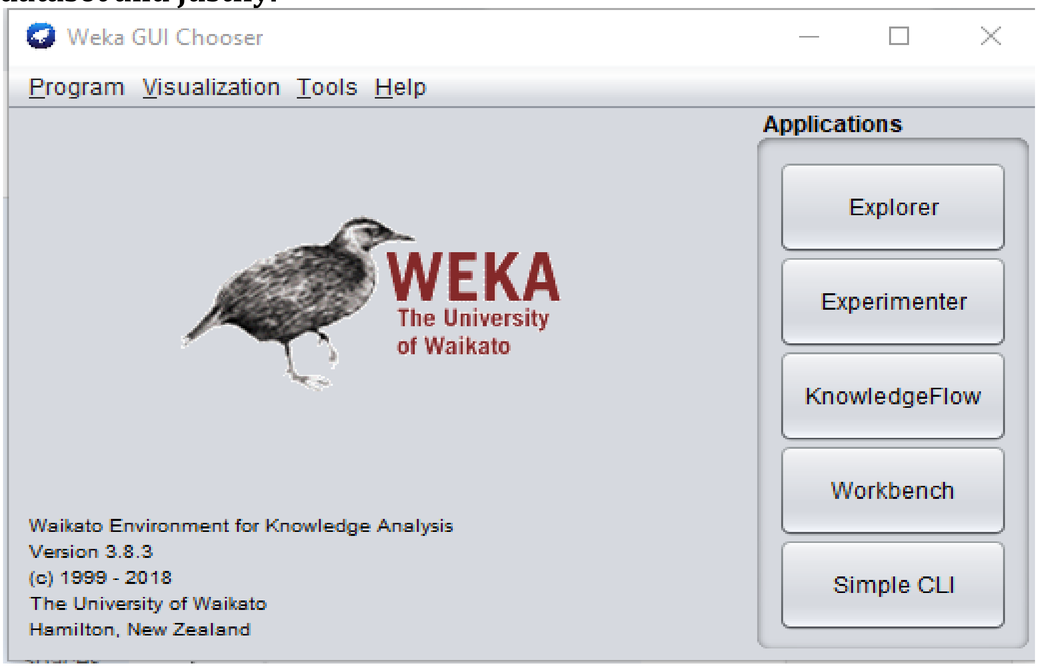
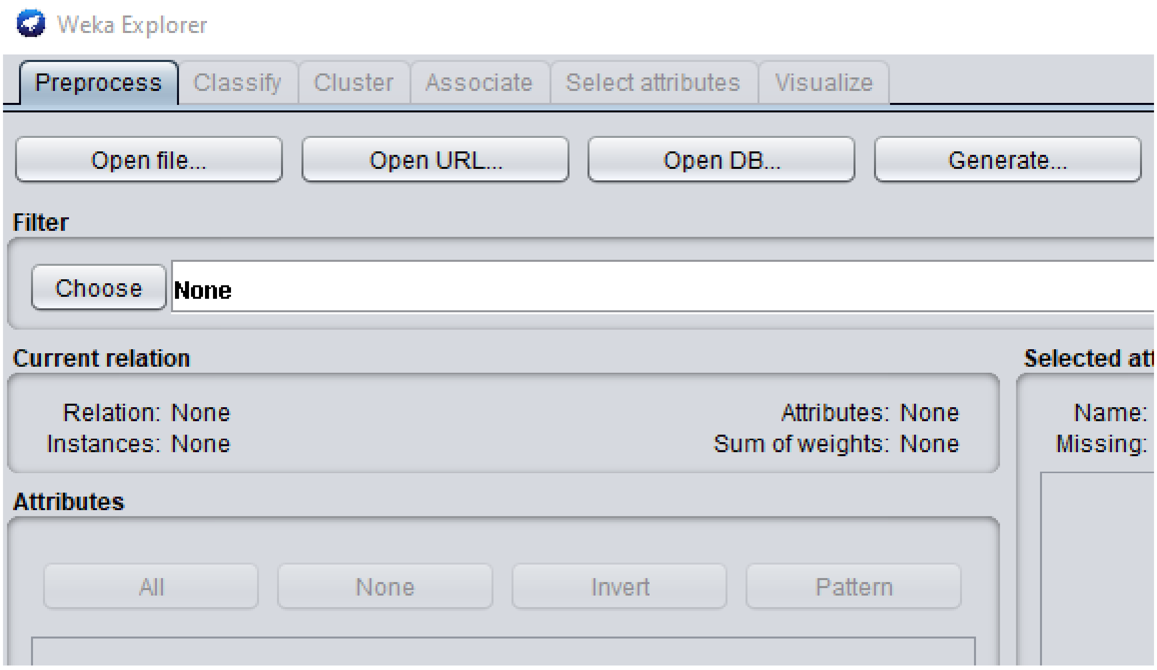
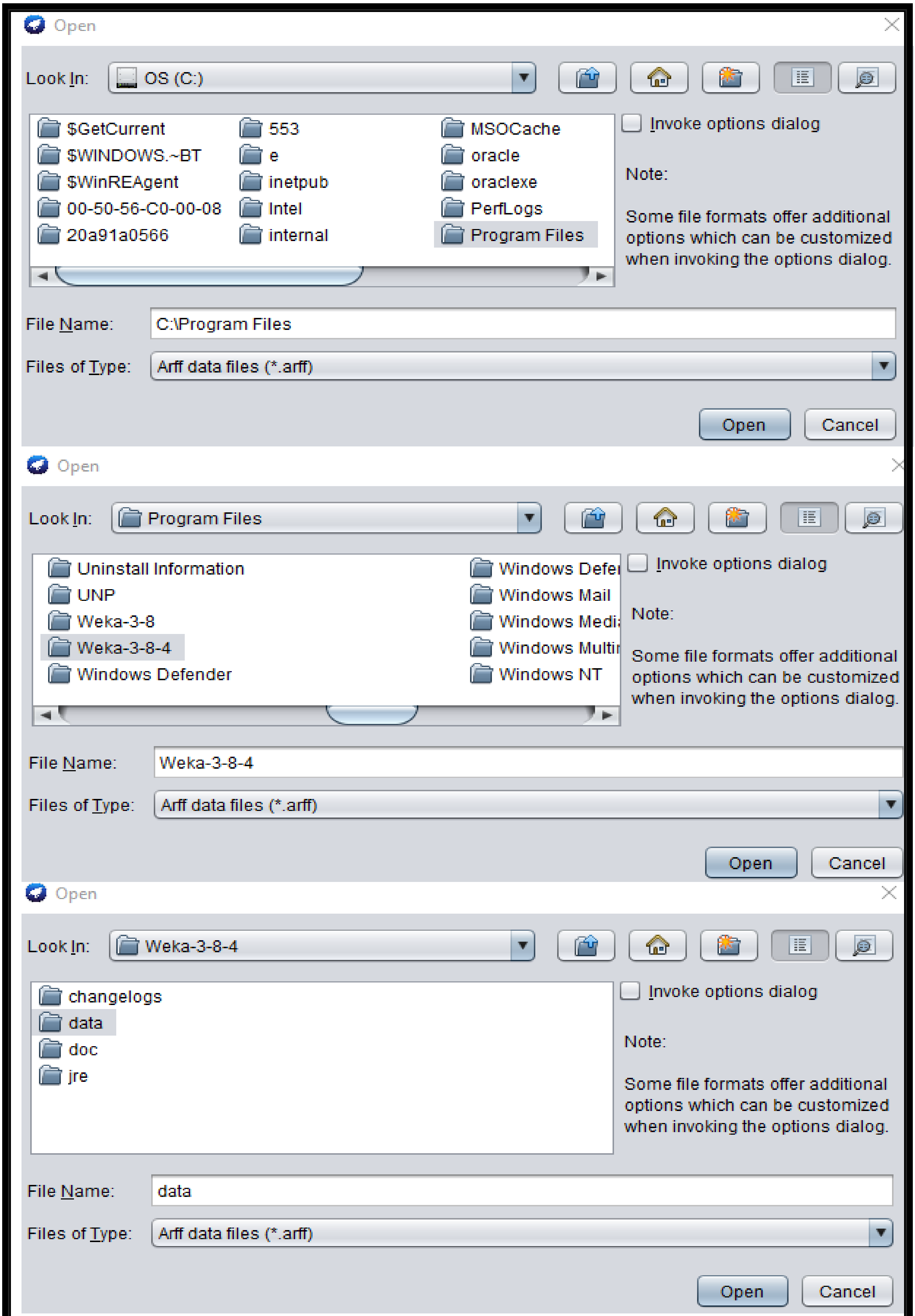
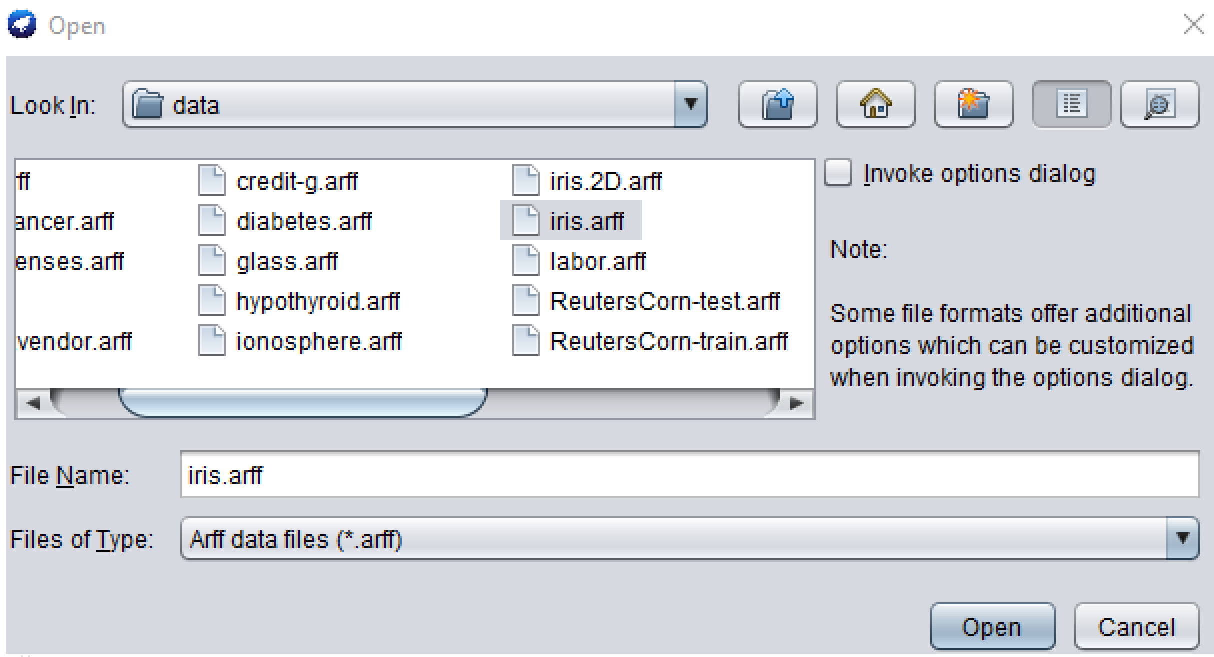
Week – 5 Demonstrate performing classification on data sets
Load each dataset into Weka and run 1d3, J48 classification algorithm.
Study the classifier output. Compute entropy values, Kappa statistic.
Extract if-then rules from the decision tree generated by the classifier, Observe the confusion matrix.
Load each dataset into Weka and perform Naïve-bayes classification and k-Nearest Neighbour classification.
Interpret the results obtained.
Plot RoC Curves
Compare classification results of ID3, J48, Naïve-Bayes and k-NN classifiers for each dataset, and deduce which classifier is performing best and poor for each dataset and justify.

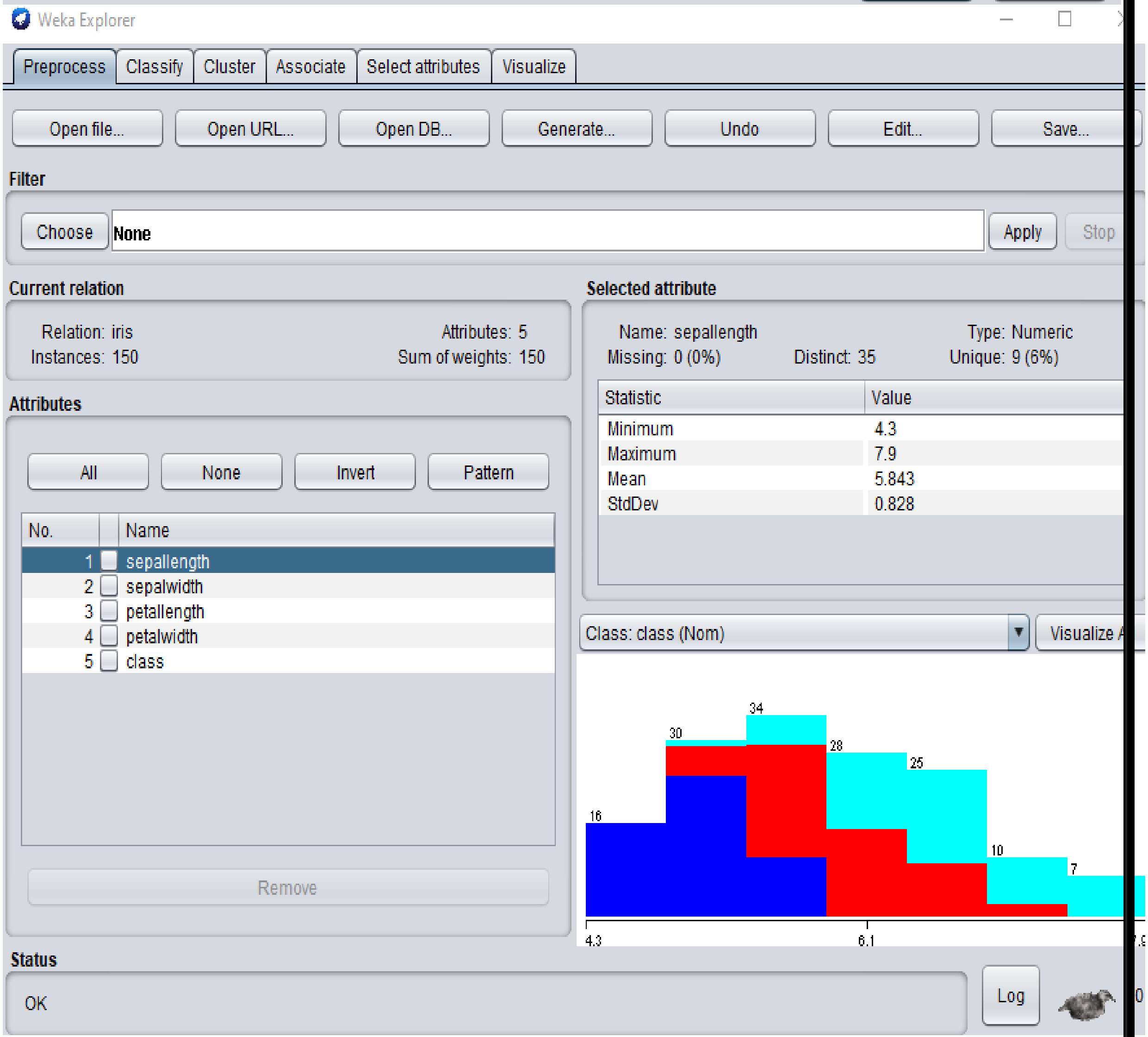


GO TO EXPLORER

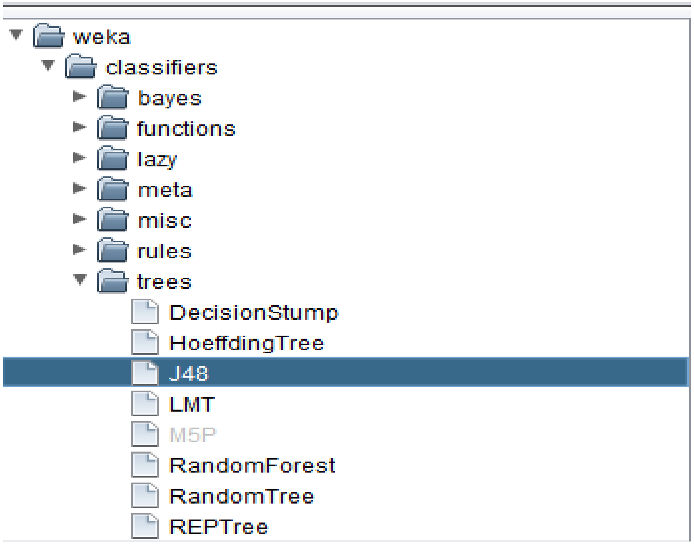


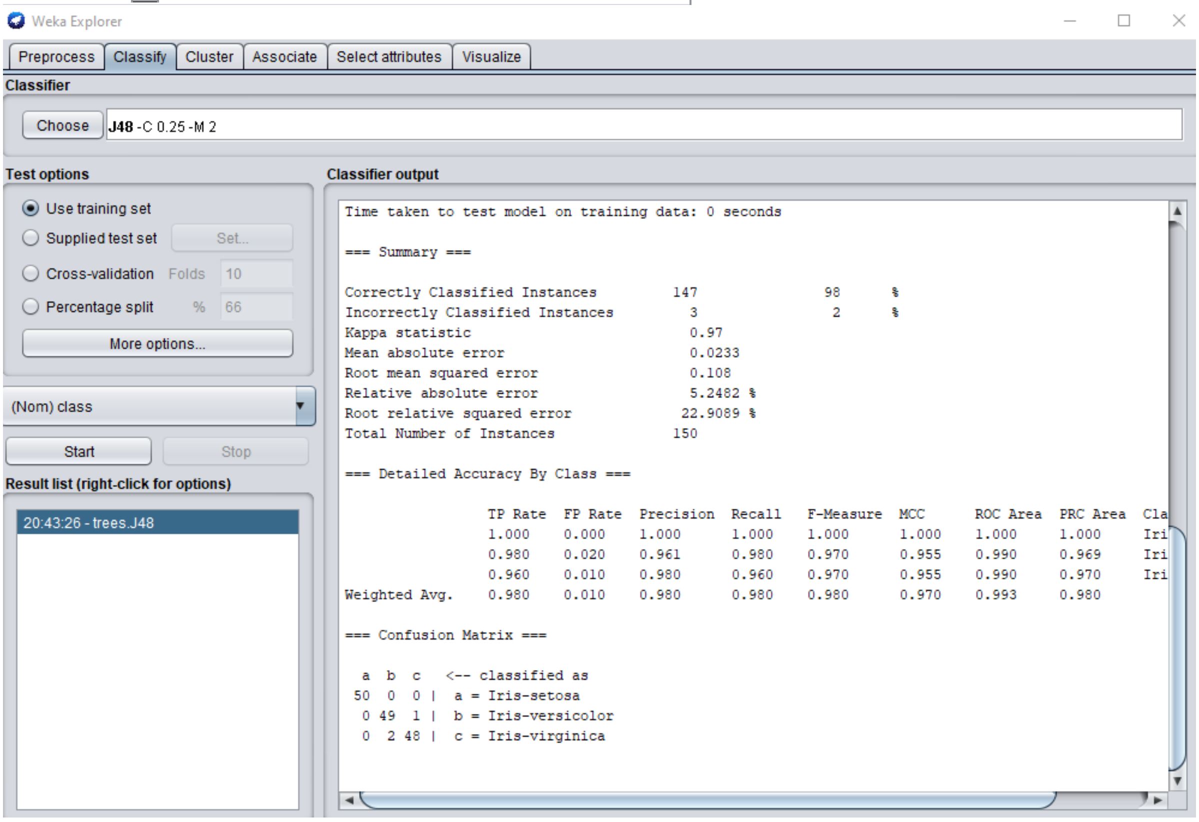

Open
Look In: OS (C:)
$GetCurrent
$WINDOWS.~BT
$WinREAgent
00-50-56-C0-00-08
20a91a0566
553
e
inetpub
Intel
internal
MSOCache
oracle
oraclexe
PerfLogs
Program Files
Invoke options dialog
Note:
Some file formats offer additional options which can be customized when invoking the options dialog.
File Name: C:\Program Files
Files of Type: Arff data files (*.arff)
Open
Cancel
Open
Look In: Program Files
Uninstall Information
UNP
Weka-3-8
Weka-3-8-4
Windows Defender
Windows Defe
Windows Mail
Windows Medi
Windows Multi
Windows NT
Invoke options dialog
Note:
Some file formats offer additional options which can be customized when invoking the options dialog.
File Name: Weka-3-8-4
Files of Type: Arff data files (*.arff)
Open
Cancel
Open
Look In: Weka-3-8-4
changelogs
data
doc
jre
Invoke options dialog
Note:
Some file formats offer additional options which can be customized when invoking the options dialog.
File Name: data
Files of Type: Arff data files (*.arff)
Open
Cancel





GO TO CLASSIFY AND CHOOSE





=== Run information ===

Scheme: weka.classifiers.trees.J48 -C 0.25 -M 2

Relation: iris

Instances: 150

Attributes: 5

sepallength
sepalwidth
petallength
petalwidth
class

Test mode: evaluate on training data

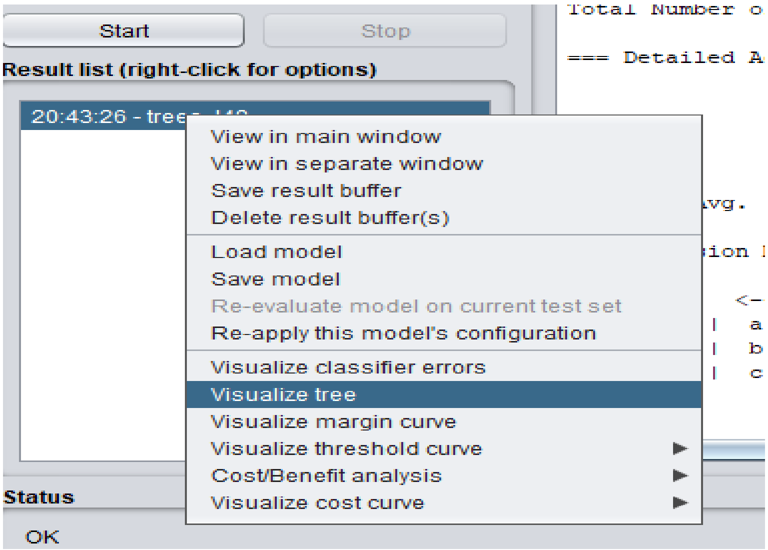
=== Classifier model (full training set) ===

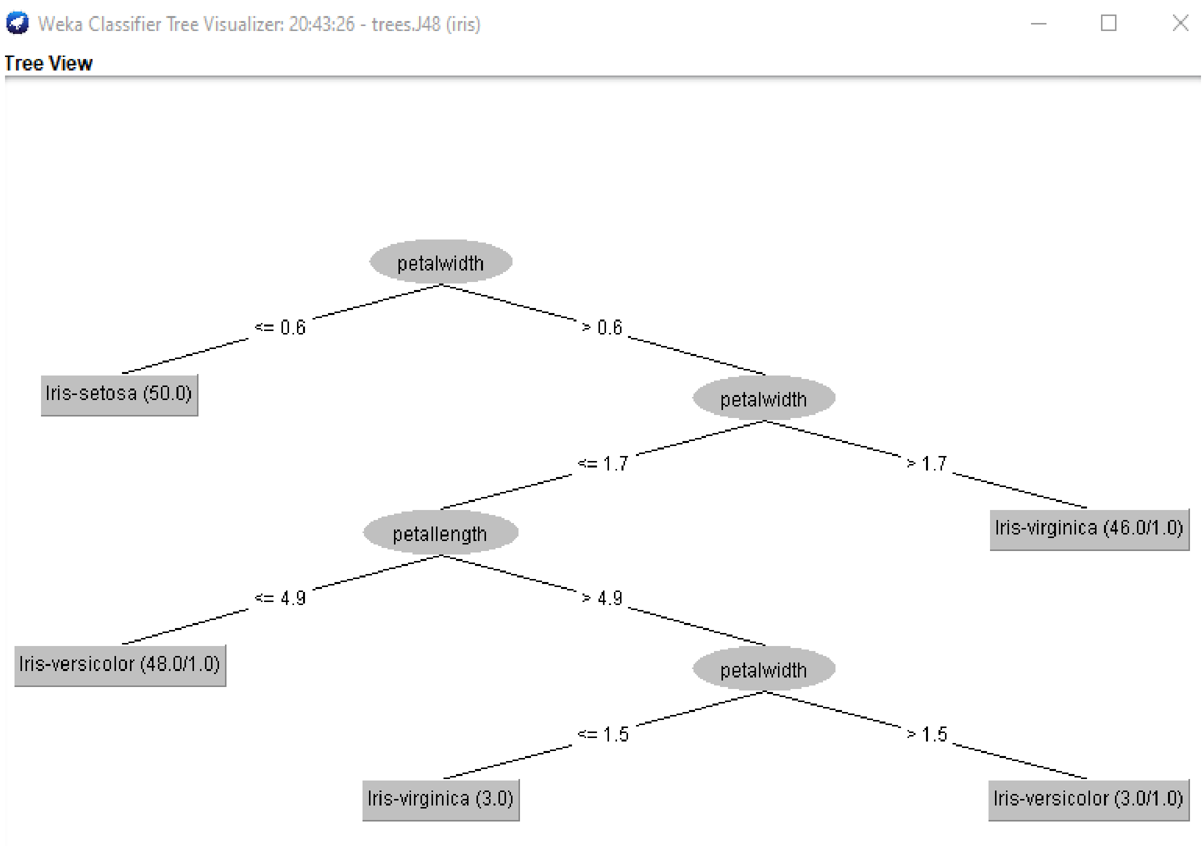
```
J48 pruned tree
------------------
petalwidth <= 0.6: Iris-setosa (50.0)
petalwidth > 0.6
|   petalwidth <= 1.7
|   |   petallength <= 4.9: Iris-versicolor (48.0/1.0)
|   |   petallength > 4.9
|   |   |   petalwidth <= 1.5: Iris-virginica (3.0)
|   |   |   petalwidth > 1.5: Iris-versicolor (3.0/1.0)
|   petalwidth > 1.7: Iris-virginica (46.0/1.0)

Number of Leaves  :       5
Size of the tree :    9

Time taken to build model: 0.01 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances         147               98      %
Incorrectly Classified Instances         3                2      %
Kappa statistic                          0.97
Mean absolute error                      0.0233
Root mean squared error                  0.108
Relative absolute error                  5.2482 %
Root relative squared error             22.9089 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area
PRC Area  Class
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000
Iris-setosa
                 0.980    0.020    0.961      0.980    0.970      0.955    0.990     0.969
Iris-versicolor
                 0.960    0.010    0.980      0.960    0.970      0.955    0.990     0.970
Iris-virginica
Weighted Avg.    0.980    0.010    0.980      0.980    0.980      0.970    0.993
0.980
```

=== Confusion Matrix ===

```
 a  b  c   <-- classified as
50  0  0 |  a = Iris-setosa
 0 49  1 |  b = Iris-versicolor
 0  2 48 |  c = Iris-virginica
```

RIGHT CLICK ON THE TREE OPTION





The kappa statistic, which takes into account chance agreement, is defined as **(observed agreement−expected agreement)/(1−expected agreement)**.

Mean Absolute Error calculates the average difference between the calculated values and actual values. It is also known as scale-dependent accuracy as it calculates error in observations taken on the same scale. It is used as evaluation metrics for regression models in machine learning. It calculates errors between actual values and values predicted by the model. It is used to predict the accuracy of the machine learning model.

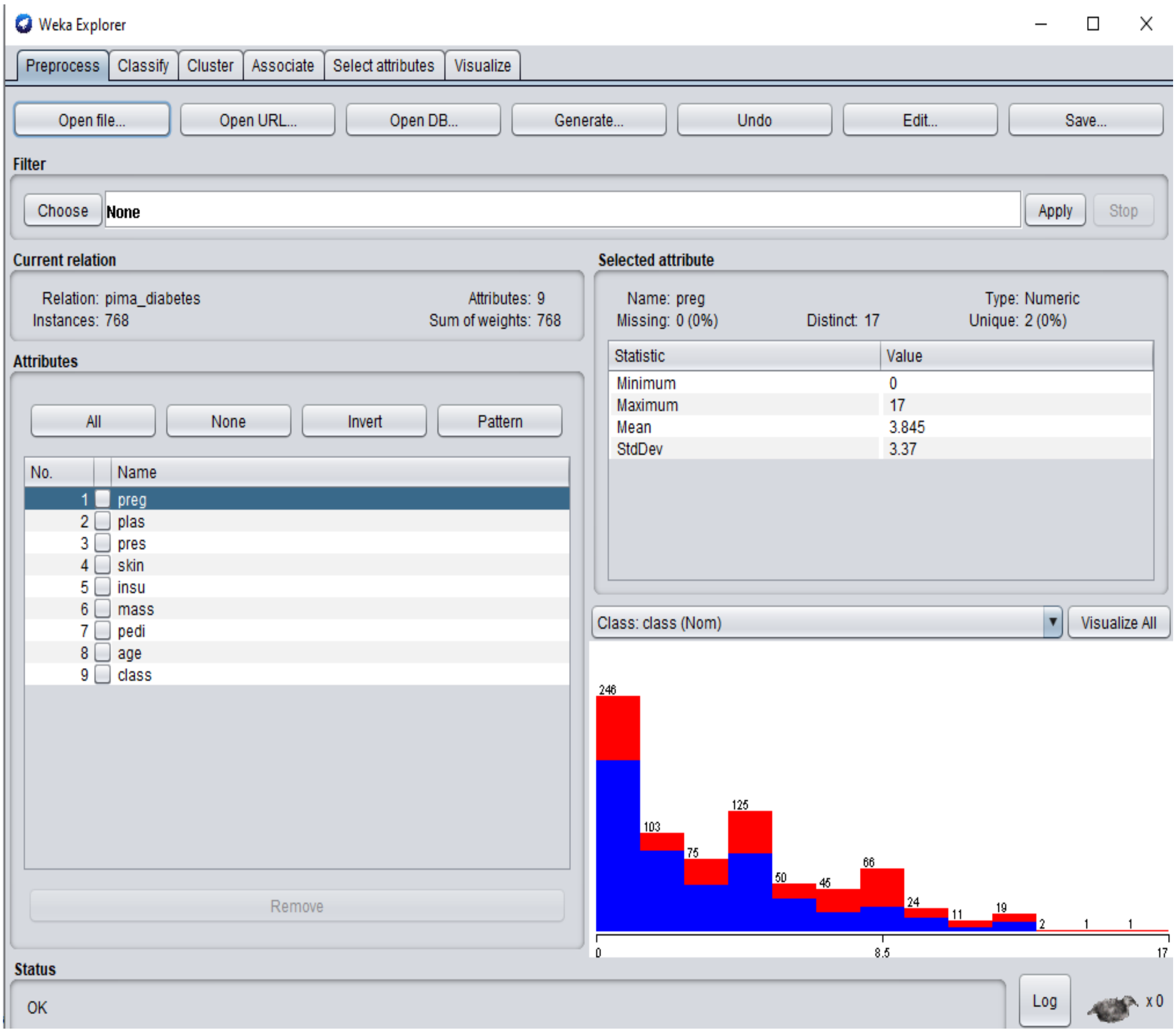
Formula:

Mean Absolute Error = $(1/n) * \sum|y_i - x_i|$

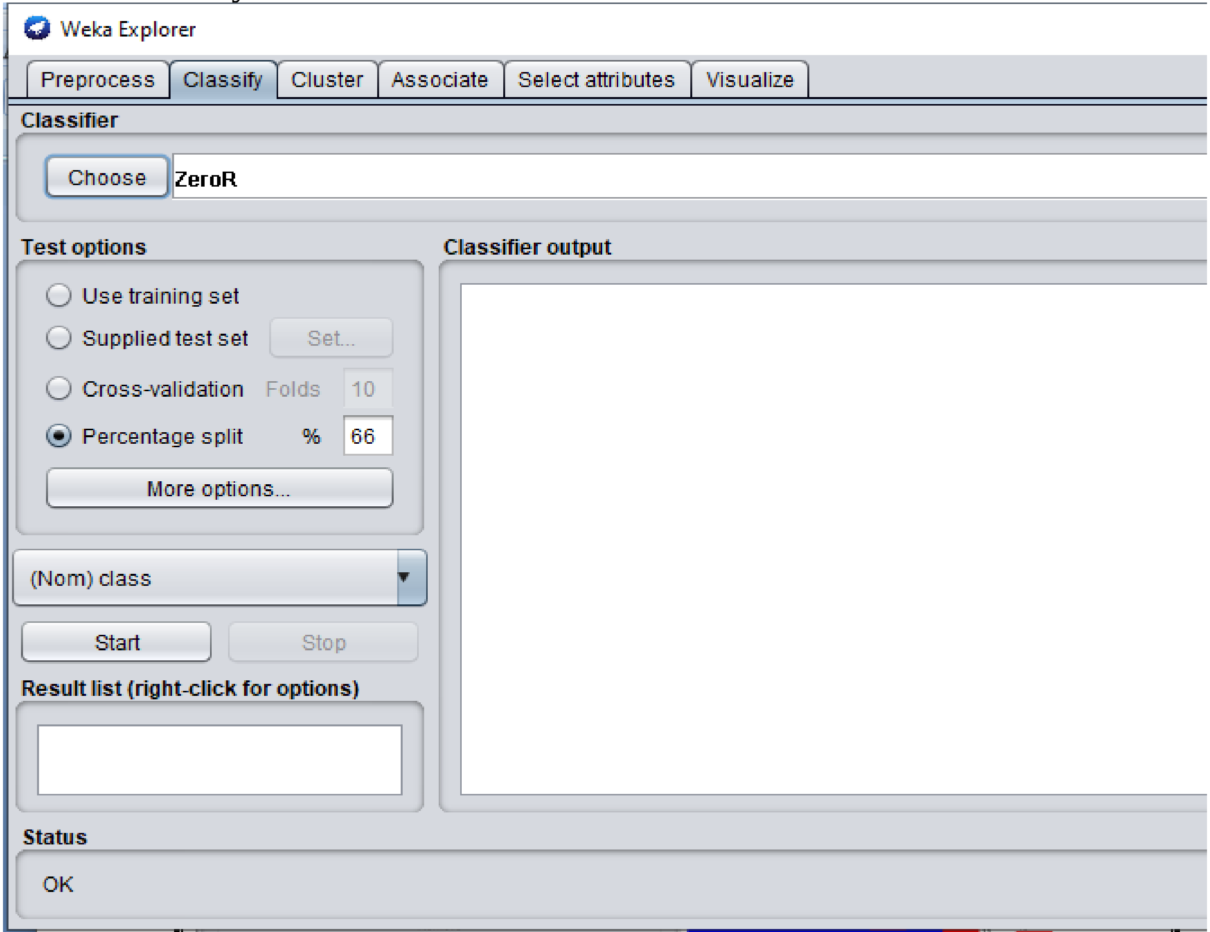
where,

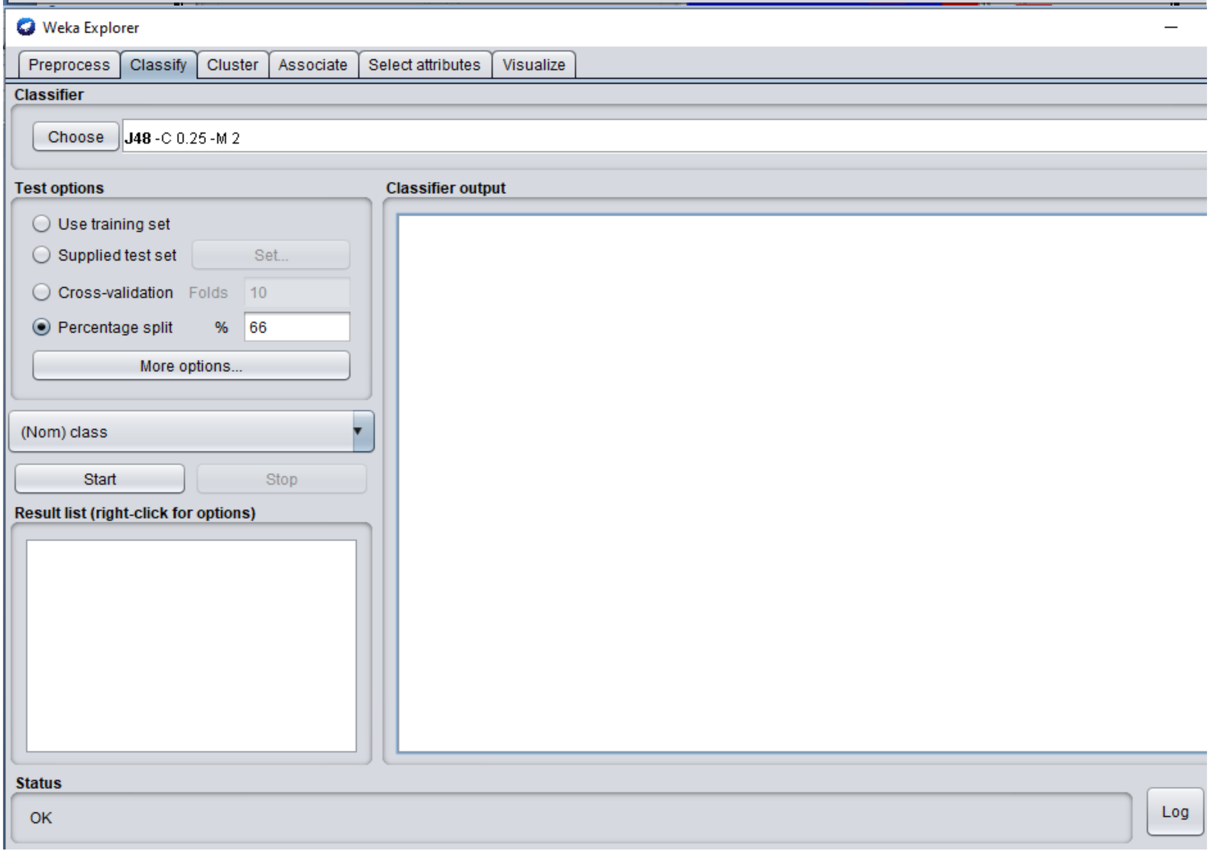
- *Σ: Greek symbol for summation*
- y_i: *Actual value for the ith observation*
- x_i: *Calculated value for the ith observation*
- *n: Total number of observations*

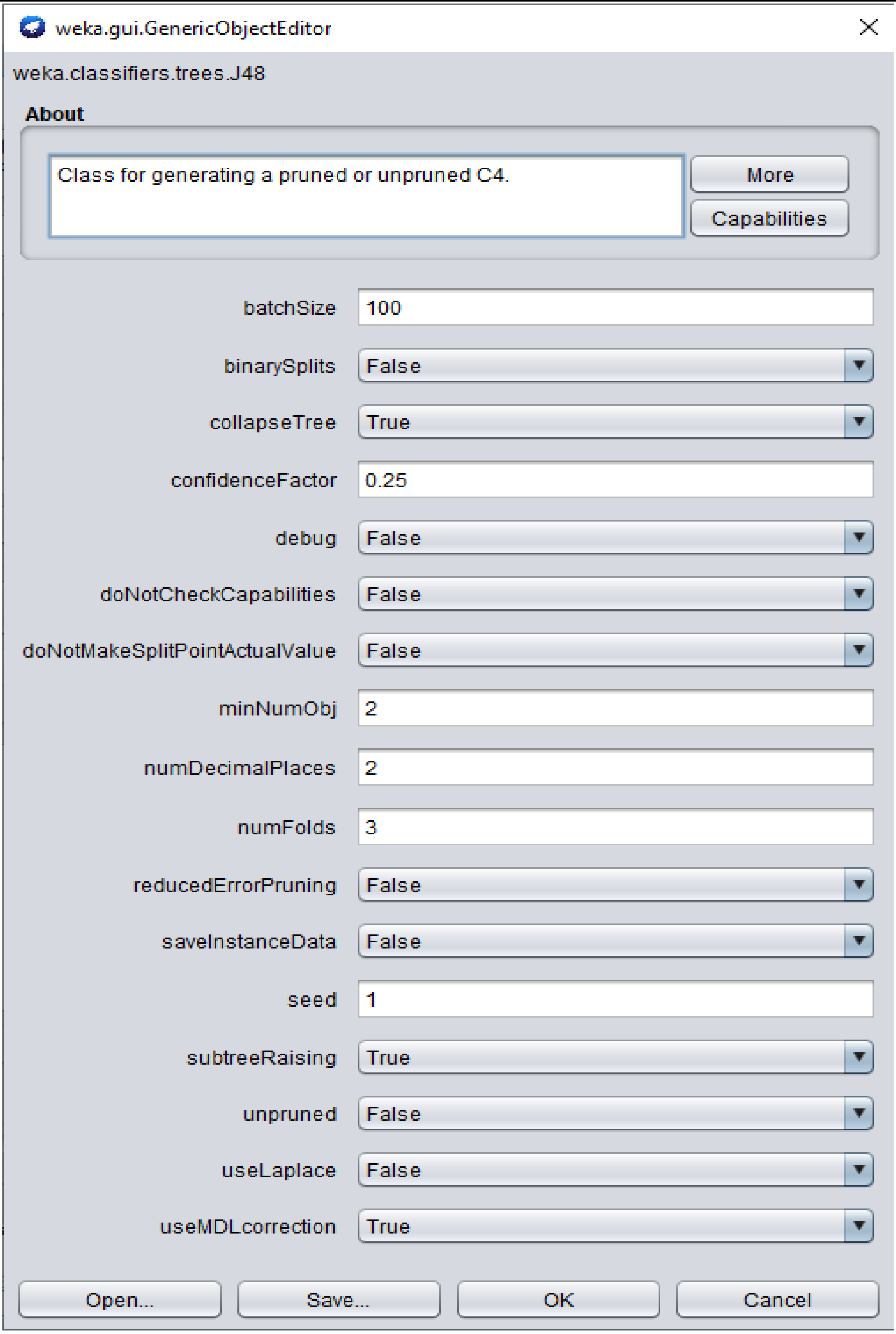
load DIABETES DATA SET



Unique attributes are not required for classification
Click on classify

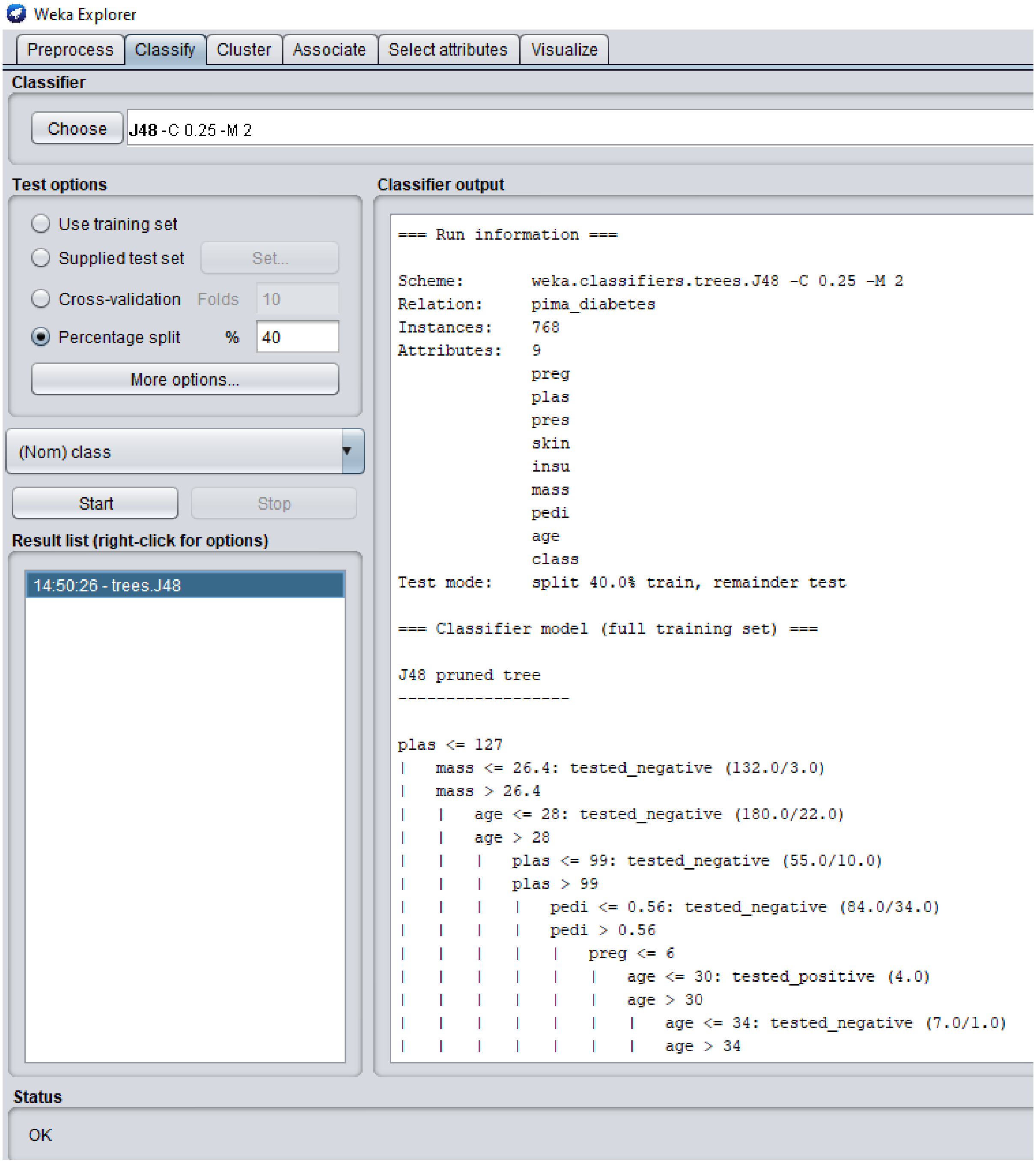






These are the parameters to change before going to classification
Percentage split: 40%
If you have 10 records ,out of that 6 is used for training data set and remaining for test data set

Click on start



Classifier output

```
|   |   |   |   pedi > 0.56
|   |   |   |   |   preg <= 6
|   |   |   |   |   |   age <= 30: tested_positive (4.0)
|   |   |   |   |   |   age > 30
|   |   |   |   |   |   |   age <= 34: tested_negative (7.0/1.0)
|   |   |   |   |   |   |   age > 34
|   |   |   |   |   |   |   |   mass <= 33.1: tested_positive (6.0)
|   |   |   |   |   |   |   |   mass > 33.1: tested_negative (4.0/1.0)
|   |   |   |   |   preg > 6: tested_positive (13.0)
plas > 127
|   mass <= 29.9
|   |   plas <= 145: tested_negative (41.0/6.0)
|   |   plas > 145
|   |   |   age <= 25: tested_negative (4.0)
|   |   |   age > 25
|   |   |   |   age <= 61
|   |   |   |   |   mass <= 27.1: tested_positive (12.0/1.0)
|   |   |   |   |   mass > 27.1
|   |   |   |   |   |   pres <= 82
|   |   |   |   |   |   |   pedi <= 0.396: tested_positive (8.0/1.0)
|   |   |   |   |   |   |   pedi > 0.396: tested_negative (3.0)
|   |   |   |   |   |   pres > 82: tested_negative (4.0)
|   |   |   |   age > 61: tested_negative (4.0)
|   mass > 29.9
|   |   plas <= 157
|   |   |   pres <= 61: tested_positive (15.0/1.0)
|   |   |   pres > 61
|   |   |   |   age <= 30: tested_negative (40.0/13.0)
|   |   |   |   age > 30: tested_positive (60.0/17.0)
|   |   plas > 157: tested_positive (92.0/12.0)

Number of Leaves  :     20

Size of the tree :      39
```

```
Number of Leaves  :     20

Size of the tree :      39


Time taken to build model: 0.03 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0.01 seconds

=== Summary ===

Correctly Classified Instances         331               71.8004 %
Incorrectly Classified Instances       130               28.1996 %
Kappa statistic                          0.3559
Mean absolute error                      0.3243
Root mean squared error                  0.4609
Relative absolute error                 70.9526 %
Root relative squared error             97.3291 %
Total Number of Instances              461

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.807    0.458    0.777      0.807    0.792      0.357    0.733     0.816     tested_negative
                 0.542    0.193    0.587      0.542    0.564      0.357    0.733     0.531     tested_positive
Weighted Avg.    0.718    0.369    0.713      0.718    0.715      0.357    0.733     0.720

=== Confusion Matrix ===

   a   b   <-- classified as
 247  59 |   a = tested_negative
  71  84 |   b = tested_positive
```

Kappa statistic:

Cohen's kappa statistic measures interrater reliability (sometimes called interobserver agreement). Interrater reliability, or precision, happens when your data raters (or collectors) give the same score to the same data item.

This statistic should only be calculated when:

Two raters each rate one trial on each sample, *or*.

One rater rates two trials on each sample.

$$k = \frac{p0 - pe}{1 - pe} = 1 - \frac{1 - p0}{1 - pe}$$

Mean Absolute Error

The Mean Absolute Error(MAE) is the average of all absolute errors.

The formula is:

$$\mathbf{MAE} = \frac{1}{n}\sum_{i=1}^{n}|x_i - x|$$

Where:

n = the number of errors,
Σ = summation symbol (which means "add them all up"),
$|x_i - x|$ = the absolute errors.

Root mean squared error:
Root mean square error or root mean square deviation is one of the most commonly used measures for evaluating the quality of predictions. It shows how far predictions fall from measured true values using Euclidean distance.
Root mean square error can be expressed as

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} \|y(i) - \hat{y}(i)\|^2}{N}},$$

where N is the number of data points, y(i) is the i-th measurement, and $\hat{y}(i)$ is its corresponding prediction.
Relative absolute error:
It is a way to measure the performance of a predictive model. It's primarily used in machine learning, data mining, and operations management. RAE is not to be confused with relative error, which is a general measure of precision or accuracy for instruments like clocks, rulers, or scales.

$$\frac{|p_1 - a_1| + \ldots + |p_n - a_n|}{|\bar{a} - a_1| + \ldots + |\bar{a} - a_n|}$$

Root relative squared error:
The Root Relative Squared Error (RRSE) is defined as the square root of the sum of squared errors of a predictive model normalized by the sum of squared errors of a simple model.
the root relative squared error E_i of an individual model i is evaluated by the equation:

$$E_i = \sqrt{\frac{\sum_{j=1}^{n} (P_{(ij)} - T_j)^2}{\sum_{j=1}^{n} (T_j - \bar{T})^2}}$$

where $P_{(ij)}$ is the value predicted by the individual model i for record j (out of n records); T_j is the target value for record j; and $\bar{T}$ is given by the formula:

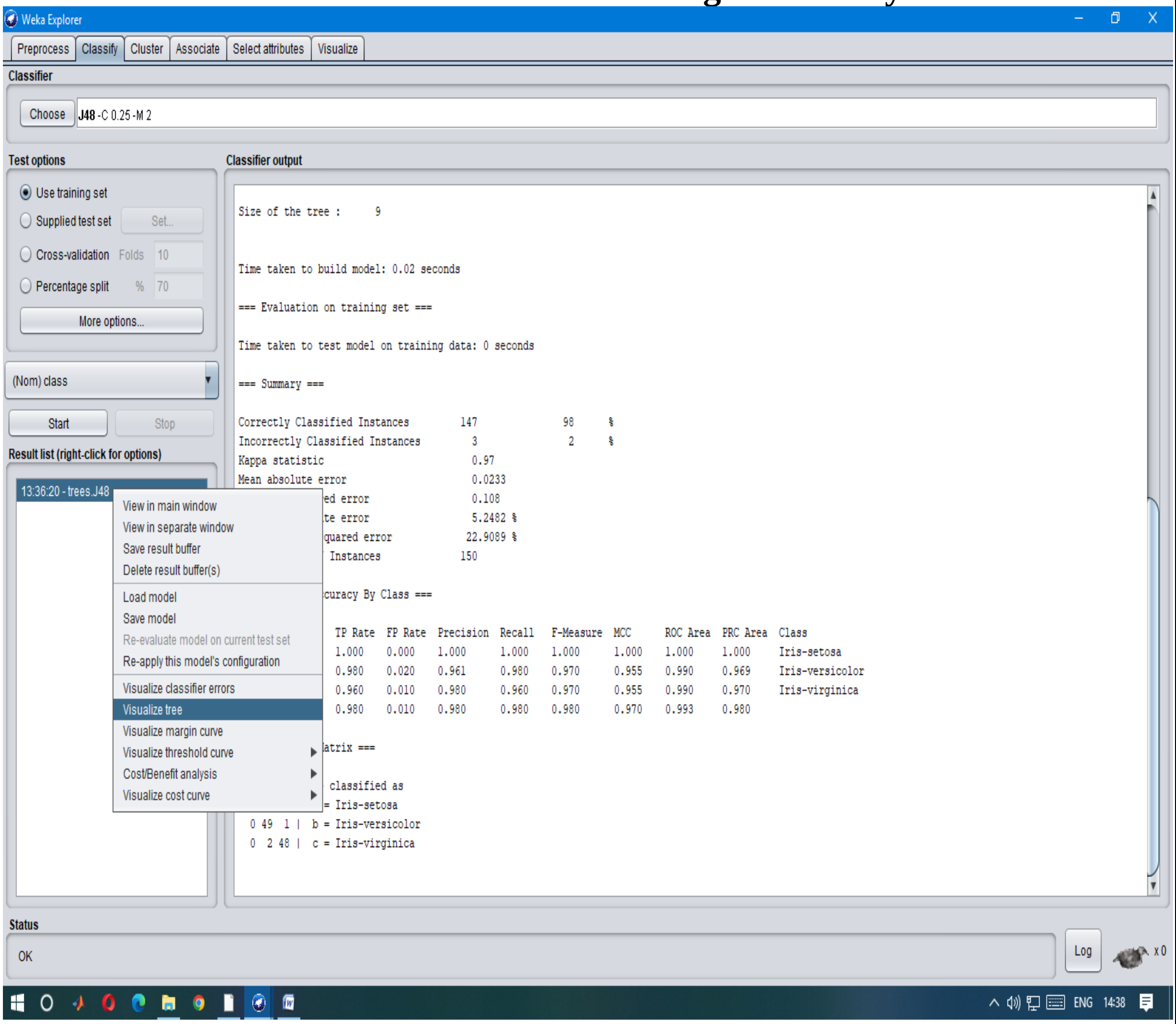
$$\bar{T} = \frac{1}{n} \sum_{j=1}^{n} T_j$$

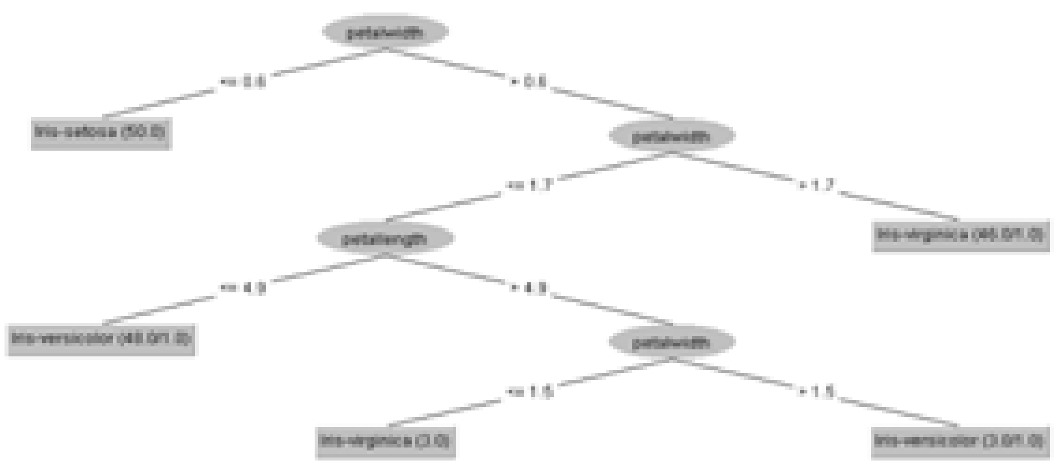
For a perfect fit, the numerator is equal to 0 and E_i = 0. So, the E_i index ranges from 0 to infinity, with 0 corresponding to the ideal.

, where:
n: represents the number of observations

y_i: represents the realized value
$\hat{y}_i$: represents the predicted value
$\bar{y}$: represents the average of the realized values

5. Extract if-then rules from the decision tree generated by the classifier.





6. Observe the confusion matrix.

```
=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 49  1 |  b = Iris-versicolor
  0  2 48 |  c = Iris-virginica
```

Precision: Appropriate when minimizing false positives is the focus.

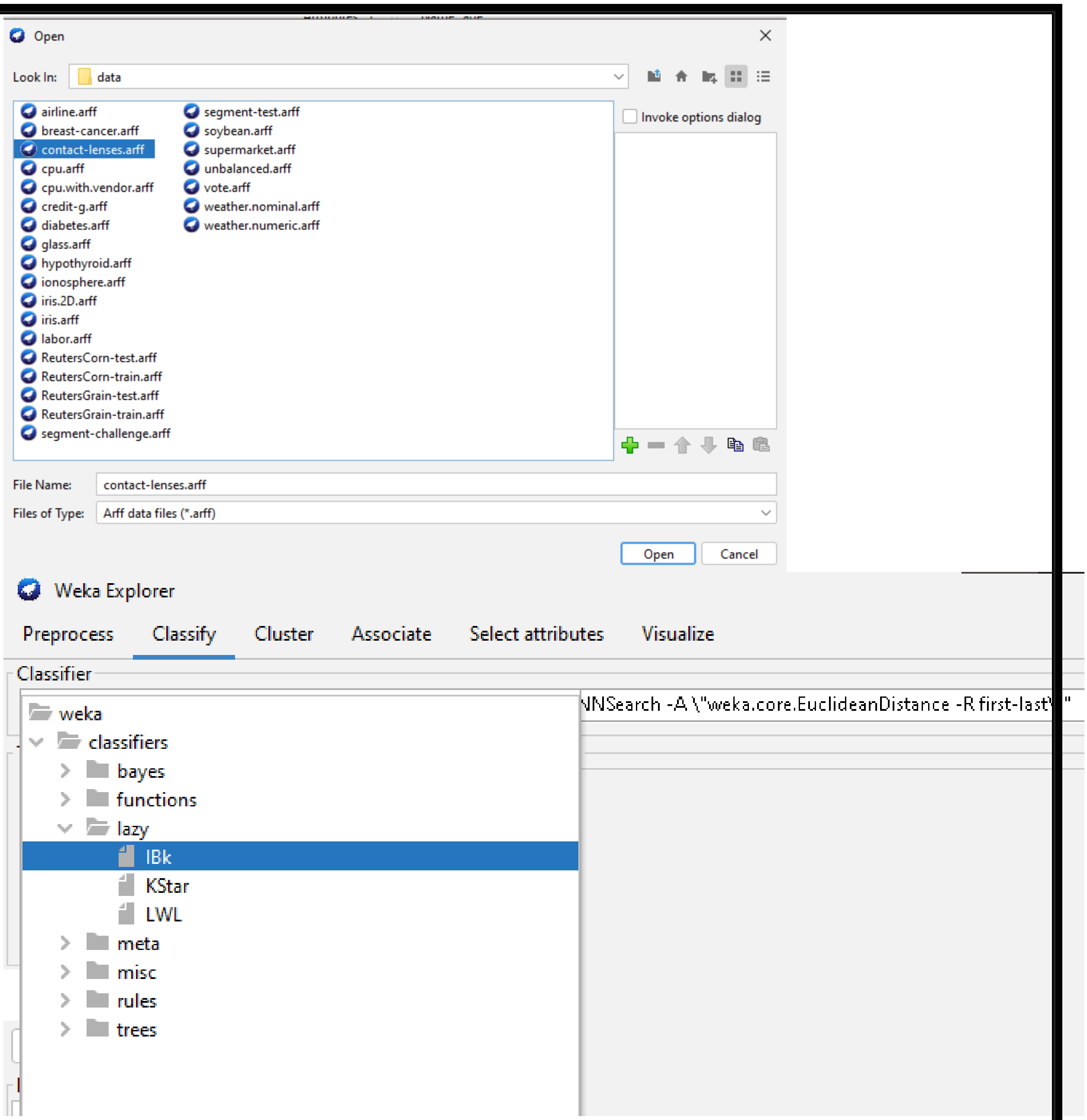
Recall: Appropriate when minimizing false negatives is the focus.

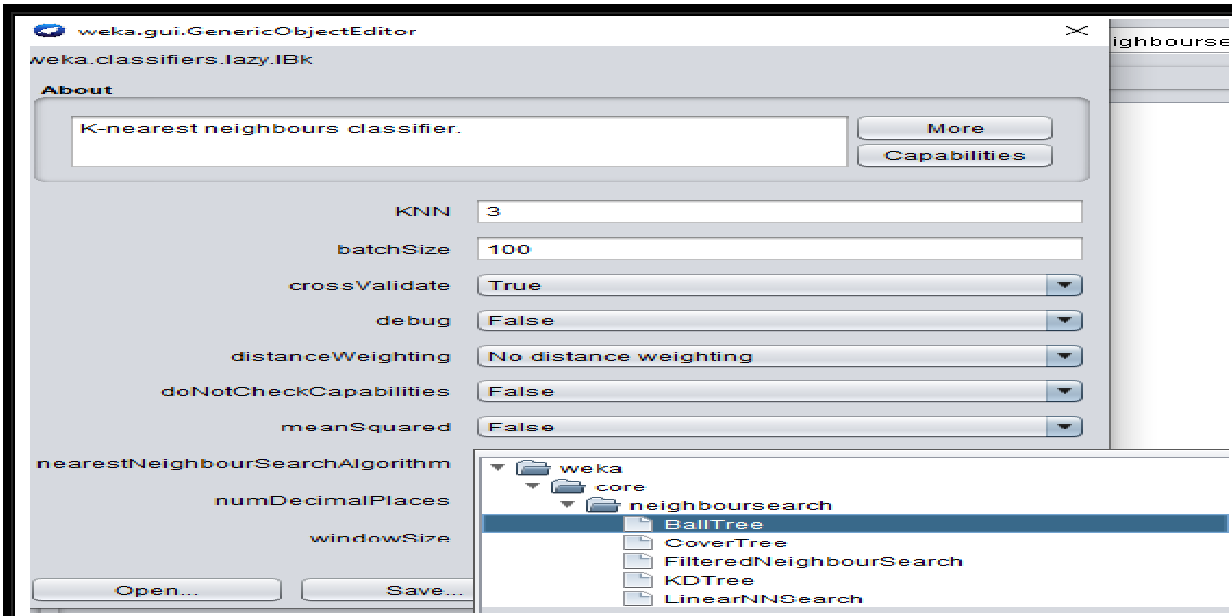
TP Rate: rate of true positives (instances correctly classified as a given class)
FP Rate: rate of false positives (instances falsely classified as a given class)
F measure is :
F-Measure = (2 * Precision * Recall) / (Precision + Recall)
MCC : it is used in machine learning as a measure of the quality of binary (two-class) classifications. It takes into account true and false positives and negatives and is generally regarded as a balanced measure which can be used even if the classes are of very different sizes
ROC(Receiver Operating Characteristics) area measurement: One of the most important values output by Weka. They give you an idea of how the classifiers are performing in general
PRC(Precision Recall) area :
Precision-recall curve. A plot of precision (= PPV) vs. recall (= sensitivity) for all potential cut-offs for a test.

Load each dataset into Weka and perform Naïve-bayes classification and k-Nearest Neighbour classification. Interpret the results obtained. Plot RoC Curves Compare classification results of ID3, J48, Naïve-Bayes and k-NN classifiers for each dataset, and deduce which classifier is performing best and poor for each dataset and justify.

K-Nearest Neighbors (KNN) is a standard machine-learning method that has been extended to large-scale data mining efforts. The idea is that one uses a large amount of training data, where each data point is characterized by a set of variables. KNN captures the idea of similarity (sometimes called distance, proximity, or closeness) with some mathematics we might have learned in our childhood— calculating the distance between points on a graph. There are other ways of calculating distance, and one way might be preferable depending on the problem we are solving. However, the straight-line distance (also called the Euclidean distance) is a popular and familiar choice. It is widely disposable in real-life scenarios since it is non-parametric, meaning, it does not make any underlying assumptions about the distribution of data (as opposed to other

algorithms such as GMM, which assume a Gaussian distribution of the given data).
Advantages & Disadvantages of KNN Algorithm
Advantages
It is very easy to understand and implement
It is an instance-based learning(lazy learning) algorithm.
KNN does not learn during the training phase hence new data points can be added with affecting the performance of the algorithm.
It is well suited for small datasets.
Disadvantages
It fails when variables have different scales.
It is difficult to choose K-value.
It leads to ambiguous interpretations.
It is sensitive to outliers and missing values.
Does not work well with large datasets.
It does not work well with high dimensions.
K nearest neighbour:
it is also called instance based learning
it's very similar to a desktop
different names of KNN
---Memory base
example
instance based
lazy learning
KNN helps us to assign label to unknown data.
APPLY KNN ON DIABETES DATA SET
2) KNN Alogrithm
Different names of KNN
k-Nearest Neighbouring
Memory-Based Reasoning
Example-Based Reasoning
Instance-Based Learning
Lazy Learning


Open
Look In: data
airline.arff
breast-cancer.arff
contact-lenses.arff
cpu.arff
cpu.with.vendor.arff
credit-g.arff
diabetes.arff
glass.arff
hypothyroid.arff
ionosphere.arff
iris.2D.arff
iris.arff
labor.arff
ReutersCorn-test.arff
ReutersCorn-train.arff
ReutersGrain-test.arff
ReutersGrain-train.arff
segment-challenge.arff
segment-test.arff
soybean.arff
supermarket.arff
unbalanced.arff
vote.arff
weather.nominal.arff
weather.numeric.arff
Invoke options dialog
File Name: contact-lenses.arff
Files of Type: Arff data files (*.arff)
Open
Cancel
Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Classifier
NNSearch -A \"weka.core.EuclideanDistance -R first-last\" "
weka
classifiers
bayes
functions
lazy
IBk
KStar
LWL
meta
misc
rules
trees



By using the Test Option as : Use Training Set

Classifer Output:

=== Run information ===

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""

Relation: contact-lenses

Instances: 24

Attributes: 5

age
spectacle-prescrip
astigmatism
tear-prod-rate
contact-lenses

Test mode: evaluate on training data

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

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances 24 100 %
Incorrectly Classified Instances 0 0 %
Kappa statistic 1
Mean absolute error 0.0494
Root mean squared error 0.0524
Relative absolute error 13.4078 %
Root relative squared error 12.3482 %
Total Number of Instances 24

=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area
PRC Area Class
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000
soft
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000
hard
1.000 0.000 1.000 1.000 1.000 1.000 1.000 1.000
none
Weighted Avg. 1.000 0.000 1.000 1.000 1.000 1.000 1.000
1.000

=== Confusion Matrix ===

a b c <-- classified as
5 0 0 | a = soft
0 4 0 | b = hard
0 15 | c = none

By using the Test Option as : Percentage Split – 60%
Classifer Output:
=== Run information ===

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""
Relation: contact-lenses
Instances: 24
Attributes: 5

age
spectacle-prescrip
astigmatism
tear-prod-rate
contact-lenses
Test mode: split 60.0% train, remainder test

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

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances	3	30	%
Incorrectly Classified Instances	7	70	%
Kappa statistic	-0.0145		
Mean absolute error	0.4301		
Root mean squared error	0.564		
Relative absolute error	97.0527 %		
Root relative squared error	103.7551 %		
Total Number of Instances	10		

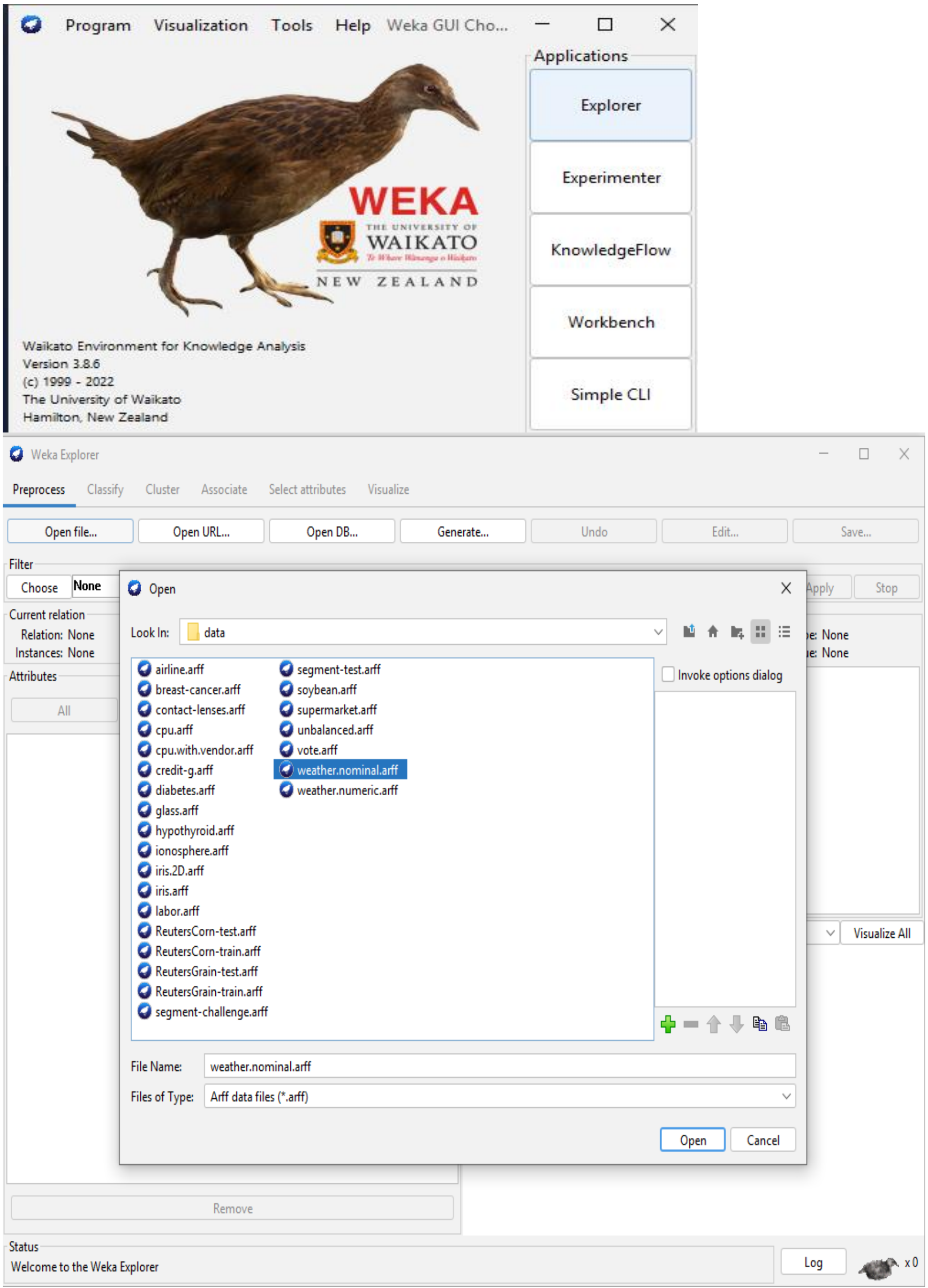
=== Detailed Accuracy By Class ===

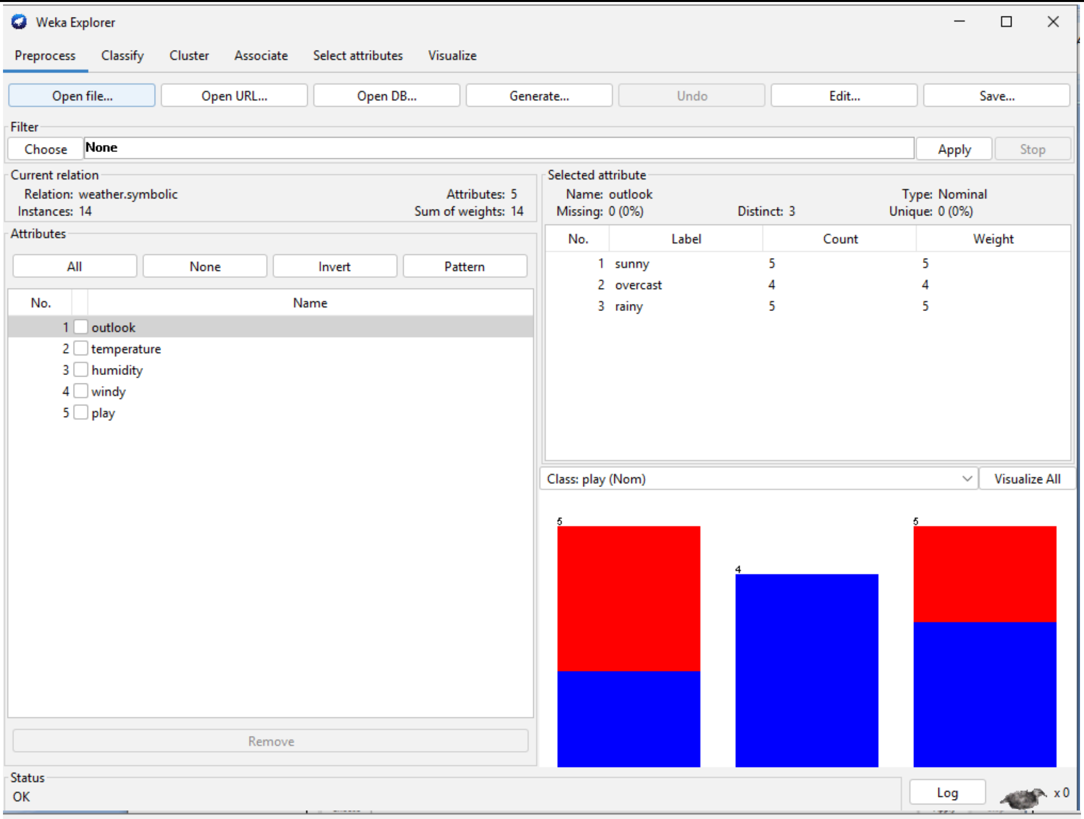
	TP Rate	FP Rate	Precision	Recall	F-Measure	MCC	ROC Area	PRC Area	Class
	0.000	0.000	?	0.000	?	?	0.920	0.886	soft
	0.000	0.333	0.000	0.000	0.000	-0.218	0.667	0.250	hard
	0.750	0.667	0.429	0.750	0.545	0.089	0.646	0.667	none
Weighted Avg.	0.300	0.300	?	0.300	?	?	0.785	0.735	

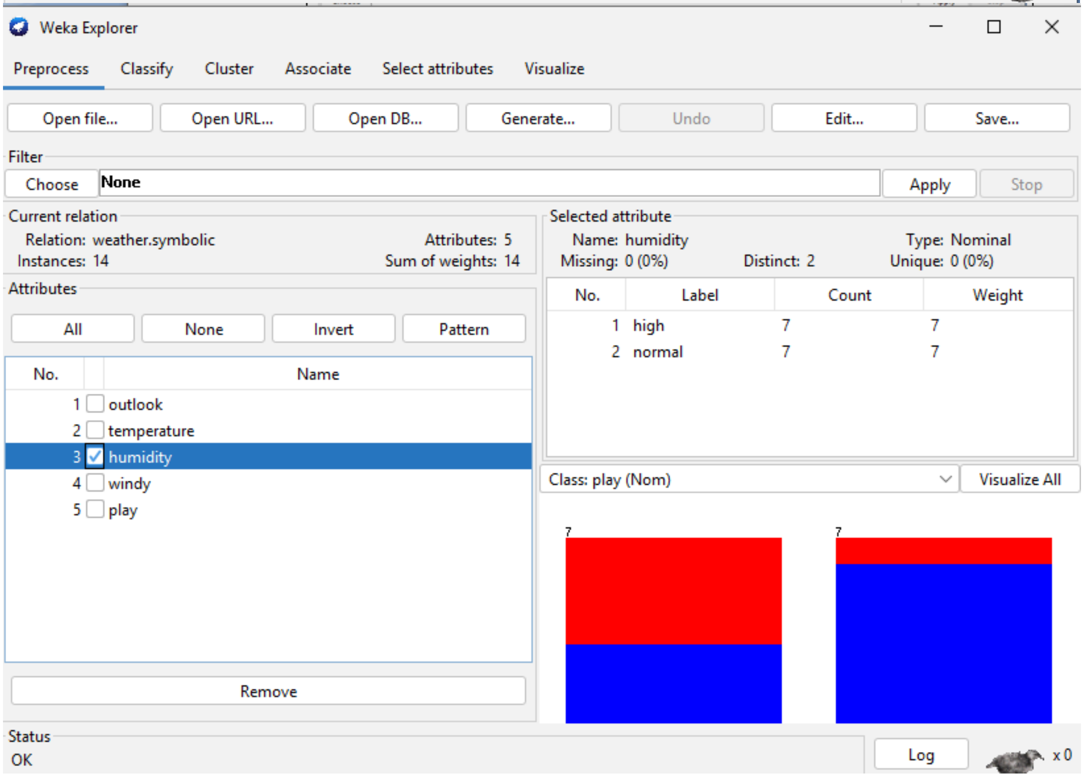
=== Confusion Matrix ===

a b c <-- classified as
0 2 3 | a = soft
0 0 1 | b = hard
0 1 3 | c = none
s

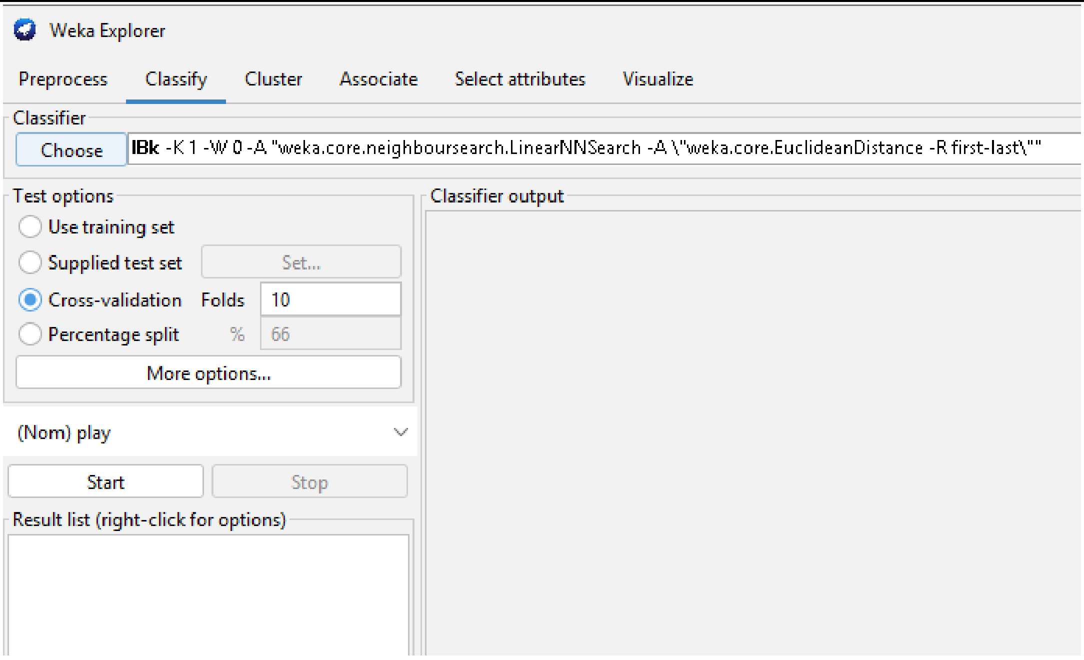
Applying KNN on weather data set



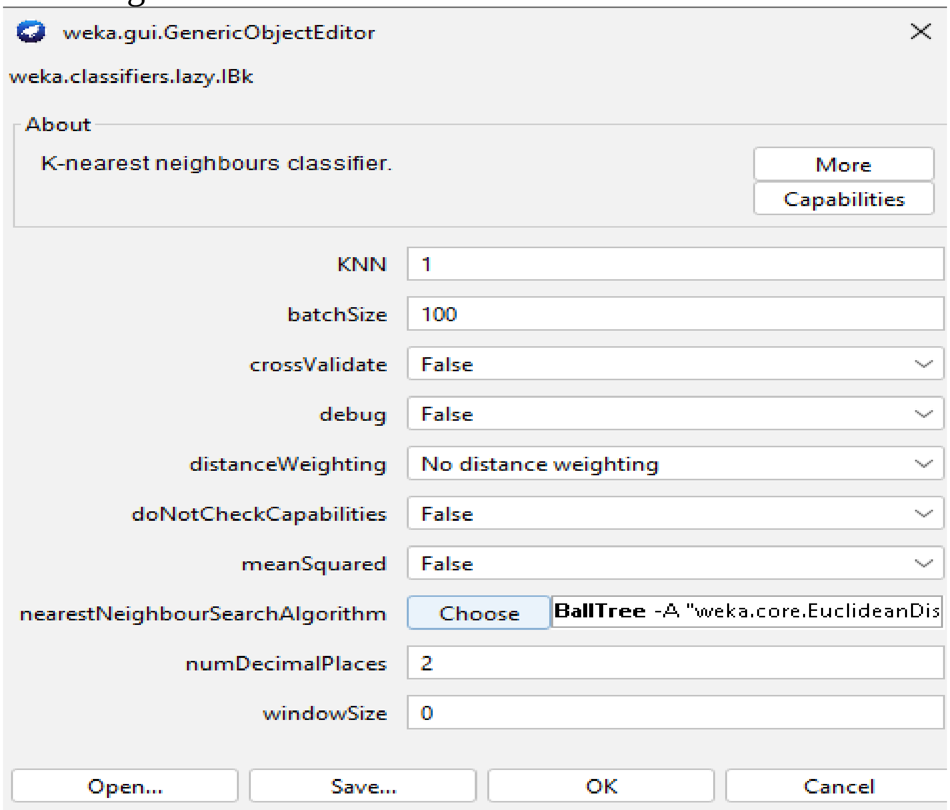




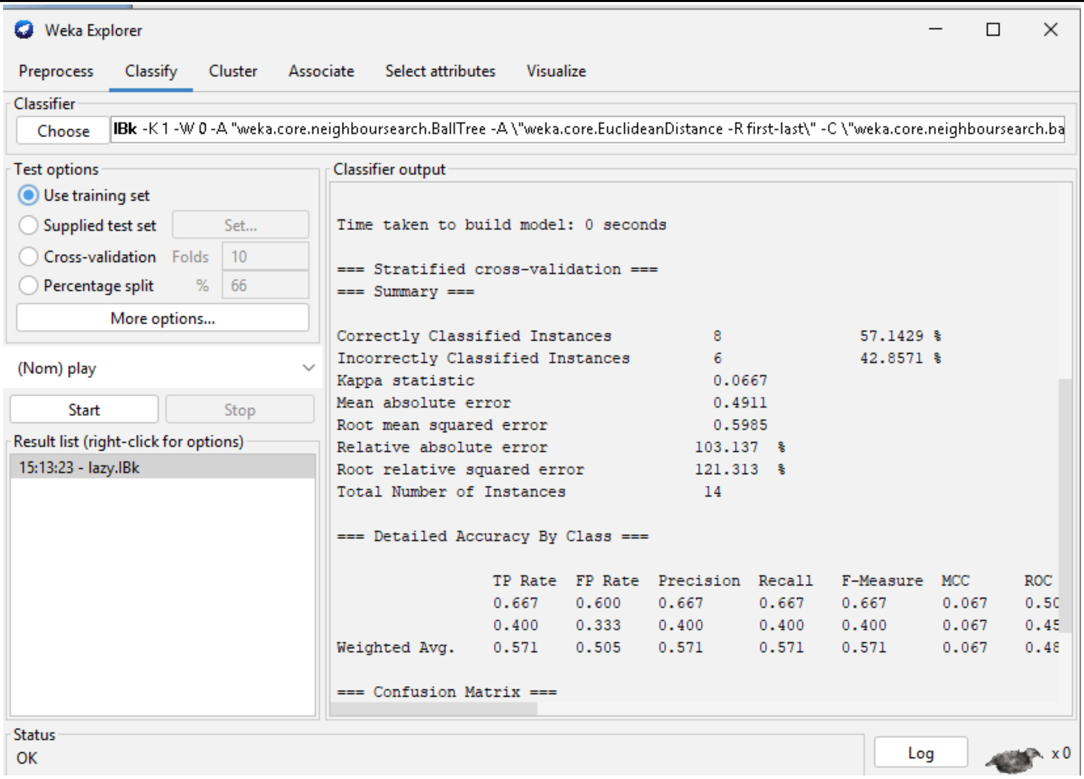
Choose test option as training set
Go to classify
Choose lazy in that select ibk



Go to properties (click on white space)
Select algorithm Ball tree



Click ok



Classifier output:
=== Run information ===

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""
Relation: weather.symbolic
Instances: 14
Attributes: 5
outlook
temperature
humidity
windy
play
Test mode: evaluate on training data

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

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on training set ===

Time taken to test model on training data: 0 seconds

=== Summary ===

Correctly Classified Instances 14 100 %
Incorrectly Classified Instances 0 0 %
Kappa statistic 1
Mean absolute error 0.0625
Root mean squared error 0.0625
Relative absolute error 13.4615 %
Root relative squared error 13.0347 %
Total Number of Instances 14

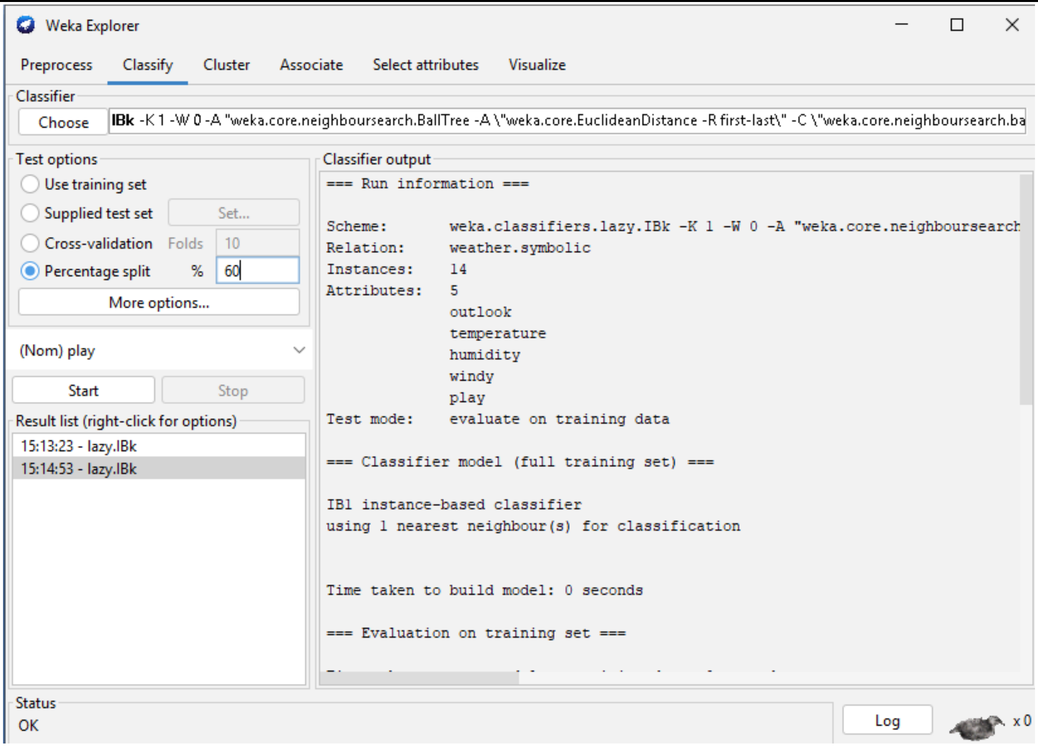
=== Detailed Accuracy By Class ===

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

=== Confusion Matrix ===

```
 a b   <-- classified as
 9 0 | a = yes
 0 5 | b = no
```

Choose test option as Percentage split:



Classifier output(after clicking start)

=== Run information ===

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""

Relation: weather.symbolic

Instances: 14

Attributes: 5

- outlook
- temperature
- humidity
- windy
- play

Test mode: split 60.0% train, remainder test

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

IB1 instance-based classifier

using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances 2 33.3333 %
Incorrectly Classified Instances 4 66.6667 %
Kappa statistic -0.5
Mean absolute error 0.5941
Root mean squared error 0.6782
Relative absolute error 127.3109 %
Root relative squared error 142.4592 %
Total Number of Instances 6

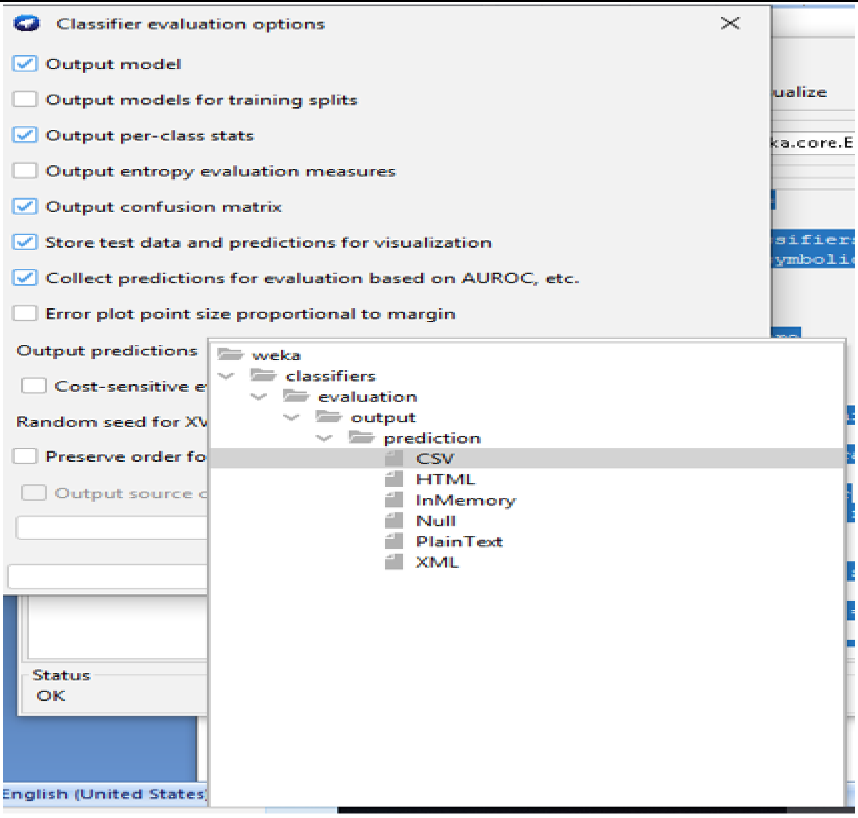
=== Detailed Accuracy By Class ===

TP Rate FP Rate Precision Recall F-Measure MCC ROC Area
PRC Area Class
0.500 1.000 0.500 0.500 0.500 -0.500 0.313 0.625
yes
0.000 0.500 0.000 0.000 0.000 -0.500 0.313 0.333
no
Weighted Avg. 0.333 0.833 0.333 0.333 0.333 -0.500 0.313
0.528

=== Confusion Matrix ===

a b <-- classified as
2 2 | a = yes
2 0 | b = no

Click on more options
Select csv



Classifier output:
=== Run information ===

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""
Relation: weather.symbolic
Instances: 14
Attributes: 5
outlook
temperature
humidity
windy
play
Test mode: split 60.0% train, remainder test

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

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification
Time taken to build model: 0 seconds

=== Predictions on test split ===

```
inst#,actual,predicted,error,prediction
1,1:yes,1:yes,,0.9
2,1:yes,2:no,+,0.9
3,1:yes,1:yes,,0.735
4,2:no,1:yes,+,0.9
5,2:no,1:yes,+,0.5
6,1:yes,2:no,+,0.9

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances         2               33.3333 %
Incorrectly Classified Instances       4               66.6667 %
Kappa statistic                       -0.5
Mean absolute error                    0.5941
Root mean squared error                0.6782
Relative absolute error              127.3109 %
Root relative squared error          142.4592 %
Total Number of Instances              6

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area
PRC Area  Class
                 0.500    1.000    0.500      0.500    0.500      -0.500   0.313     0.625
yes
                 0.000    0.500    0.000      0.000    0.000      -0.500   0.313     0.333
no
Weighted Avg.    0.333    0.833    0.333      0.333    0.333      -0.500   0.313
0.528

=== Confusion Matrix ===

 a b   <-- classified as
 2 2 | a = yes
 2 0 | b = no
```

Choose HTML in mor options

Classifier output:

```
=== Run information ===
```

Scheme: weka.classifiers.lazy.IBk -K 1 -W 0 -A "weka.core.neighboursearch.BallTree -A \"weka.core.EuclideanDistance -R first-last\" -C \"weka.core.neighboursearch.balltrees.TopDownConstructor -S weka.core.neighboursearch.balltrees.PointsClosestToFurthestChildren -N 40\""
Relation: weather.symbolic
Instances: 14
Attributes: 5
outlook
temperature
humidity
windy
play
Test mode: split 60.0% train, remainder test

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

IB1 instance-based classifier
using 1 nearest neighbour(s) for classification

Time taken to build model: 0 seconds

=== Predictions on test split ===

<html>
<head>
<title>Predictions for dataset weather.symbolic</title>
</head>
<body>
<div align="center">
<h3>Predictions for dataset weather.symbolic</h3>
<table border="1">
<tr>
<td>inst#</td><td>actual</td><td>predicted</td><td>error</td><td>prediction</td></tr>
<tr><td>1</td><td>1:yes</td><td>1:yes</td><td> </td><td align="right">0.9</td></tr>
<tr><td>2</td><td>1:yes</td><td>2:no</td><td>+</td><td align="right">0.9</td></tr>
<tr><td>3</td><td>1:yes</td><td>1:yes</td><td> </td><td align="right">0.735</td></tr>
<tr><td>4</td><td>2:no</td><td>1:yes</td><td>+</td><td align="right">0.9</td></tr>

```
<tr><td>5</td><td>2:no</td><td>1:yes</td><td>+</td><td
align="right">0.5</td></tr>
<tr><td>6</td><td>1:yes</td><td>2:no</td><td>+</td><td
align="right">0.9</td></tr>
</table>
</div>
</body>
</html>

=== Evaluation on test split ===

Time taken to test model on test split: 0 seconds

=== Summary ===

Correctly Classified Instances           2              33.3333 %
Incorrectly Classified Instances         4              66.6667 %
Kappa statistic                         -0.5
Mean absolute error                      0.5941
Root mean squared error                  0.6782
Relative absolute error                127.3109 %
Root relative squared error            142.4592 %
Total Number of Instances                6

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area
PRC Area  Class
                 0.500    1.000    0.500      0.500    0.500      -0.500   0.313     0.625
yes
                 0.000    0.500    0.000      0.000    0.000      -0.500   0.313     0.333
no
Weighted Avg.    0.333    0.833    0.333      0.333    0.333      -0.500   0.313
0.528

=== Confusion Matrix ===

 a b   <-- classified as
 2 2 | a = yes
 2 0 | b = no
```