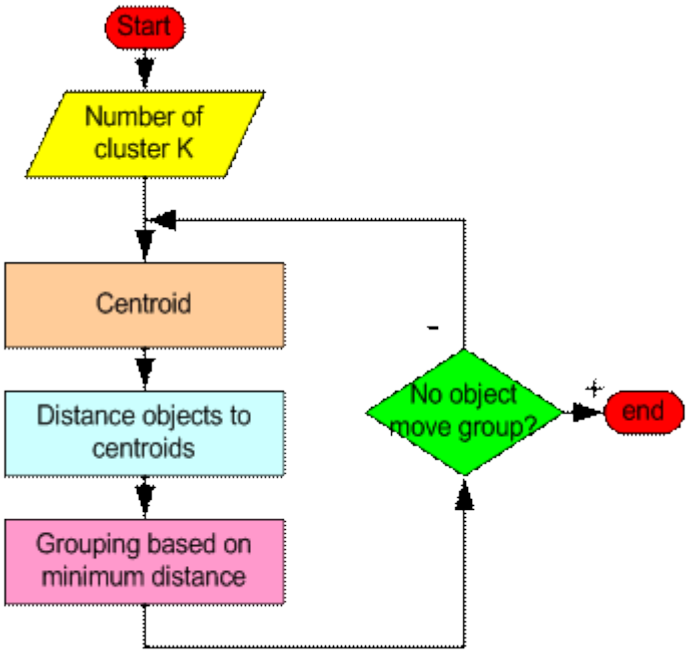
Experiment 2

Objective : To develop basic understanding of k means clustering and to find out the difference between weka and tanagra in terms of :

1. Time taken to build model
2. Clustered sum sqaure errors

INTRODUCTION TO K MEANS

K means clustering algorithm was developed by J. MacQueen (1967) and then by J. A. Hartigan and M. A. Wong around 1975.
K-means clustering is an algorithm to classify or to group objects based on attributes/features into K number of group. K is positive integer number.
The grouping is done by minimizing the sum of squares of distances between data and the corresponding cluster centroid.



k-means clustering algorithm

k-means is one of the simplest unsupervised learning algorithms that solve the well known clustering problem. The procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters) fixed apriori. The main idea is to define k centers, one for each cluster. These centers should be placed in a cunning way because of different location causes different result. So, the better choice is to place them as much as possible far away from each other. The next step is to take each point belonging to a given data set and associate it to the nearest center. When no point is pending, the first step is completed and an early group age is done. At this point we need to re-calculate k new centroids as

barycenter of the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done between the same data set points and the nearest new center. A loop has been generated. As a result of this loop we may notice that the k centers change their location step by step until no more changes are done or in other words centers do not move any more. Finally, this algorithm aims at minimizing an objective function know as squared error function given by:

$$J(V) = \sum_{i=1}^{c} \sum_{j=1}^{c_i} \left(\left\| x_i - v_j \right\| \right)^2$$

where,

'$||x_i - v_j||$' is the Euclidean distance between x_i and v_j.

'c_i' is the number of data points in i^{th} cluster.

'c' is the number of cluster centers.

Algorithmic steps for k-means clustering

Let X = $\{x_1, x_2, x_3, \ldots\ldots, x_n\}$ be the set of data points and V = $\{v_1, v_2, \ldots\ldots, v_c\}$ be the set of centers.

1) Randomly select 'c' cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..

4) Recalculate the new cluster center using:

$$v_i = (1/c_i) \sum_{j=1}^{c_i} x_i$$

where, 'c_i' represents the number of data points in i^{th} cluster.

5) Recalculate the distance between each data point and new obtained cluster centers.

6) If no data point was reassigned then stop, otherwise repeat from step 3).

ADVANTAGES

1) Fast, robust and easier to understand.

2) Relatively efficient: O(tknd), where n is # objects, k is # clusters, d is # dimension of each object, and t is # iterations. Normally, k, t, d << n.

3) Gives best result when data set are distinct or well separated from each other.

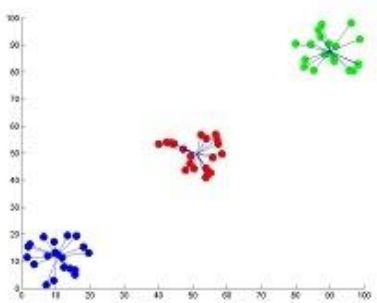


Fig I: Showing the result of k-means for *'N'* = 60 and *'c'* = 3

Disadvantages

1) The learning algorithm requires apriori specification of the number of cluster centers.

2) The use of Exclusive Assignment - If there are two highly overlapping data then k-means will not be able to resolve that there are two clusters.

3) The learning algorithm is not invariant to non-linear transformations .

4)Euclidean distance measures can unequally weight underlying factors.

5) The learning algorithm provides the local optima of the squared error function.

6) Randomly choosing of the cluster center cannot lead us to the fruitful result.

7) Applicable only when mean is defined i.e. fails for categorical data.

8) Unable to handle noisy data and outliers.

9) Algorithm fails for non-linear data set

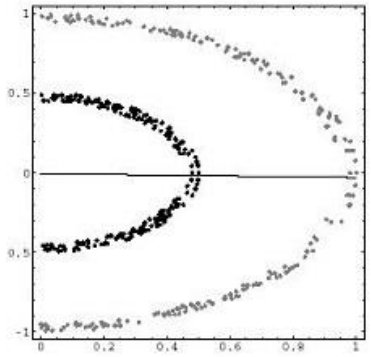


Fig II: Showing the non-linear data set where k-means algorithm fails .

K means algorithm using WEKA

This example illustrates the use of *k-means* clustering with WEKA The sample data set used for this example is based on the "bank data" available in "bank.arff" and includes 600 instances.

As an illustration of performing clustering in WEKA, we will use its implementation of the K-means algorithm to cluster the customers in this bank data set, and to characterize the resulting customer segments.

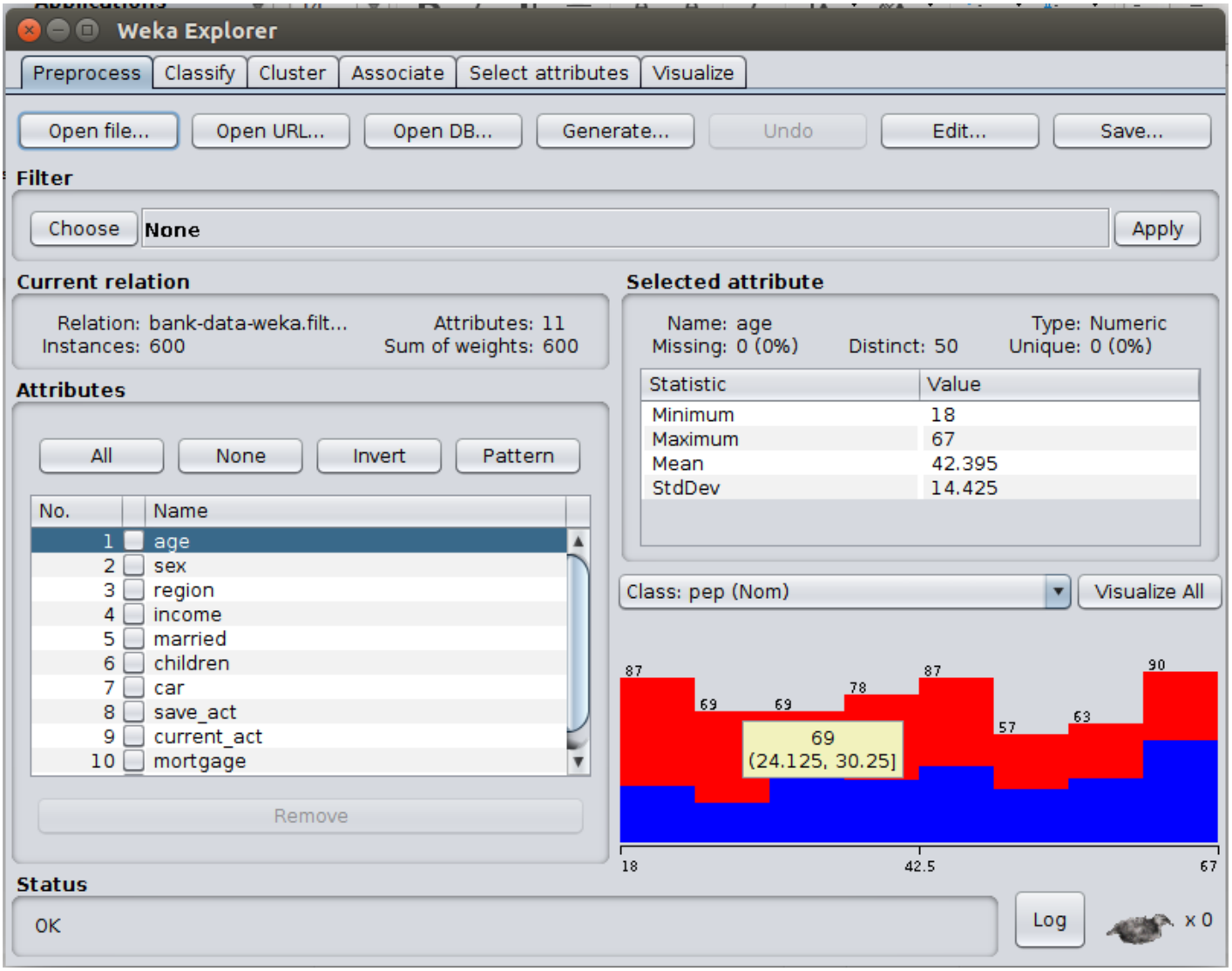


Fig1 :shows the main WEKA Explorer interface with the data file loaded.

1) Load the file .

2) To perform clustering, select the "Cluster" tab in the Explorer and click on the "Choose" button. This results in a drop down list of available clustering algorithms. In this case we select "SimpleKMeans".

3) Next, click on the text box to the right of the "Choose" button to get the pop-up window , for editing the clustering parameter.

4) In the pop-up window we enter : **number of clusters : 6**

seed : 10

```
Number of iterations: 16
Within cluster sum of squared errors: 1360.8718391528164

Initial starting points (random):

Cluster 0: 25,FEMALE,RURAL,14505.3,NO,3,NO,YES,YES,NO,NO
Cluster 1: 61,FEMALE,RURAL,22942.9,YES,2,NO,YES,YES,NO,NO
Cluster 2: 54,FEMALE,INNER_CITY,31095.6,YES,2,NO,NO,YES,NO,YES
Cluster 3: 36,FEMALE,TOWN,26920.8,YES,0,NO,NO,YES,NO,NO
Cluster 4: 42,MALE,INNER_CITY,15499.9,YES,0,YES,NO,YES,YES,YES
Cluster 5: 50,MALE,TOWN,40972.9,NO,2,YES,YES,YES,YES,YES

Missing values globally replaced with mean/mode

Final cluster centroids:
                        Cluster#
Attribute      Full Data          0          1          2          3          4          5
                 (600.0)     (72.0)    (166.0)     (71.0)     (58.0)     (99.0)    (134.0)
=============================================================================================
age                42.395    43.4444    43.7952    38.7465    37.3103     38.404    47.1791
sex                FEMALE     FEMALE     FEMALE     FEMALE     FEMALE       MALE       MALE
region         INNER_CITY      RURAL INNER_CITY INNER_CITY       TOWN INNER_CITY       TOWN
income         27524.0312  29322.789   28672.09 20239.3776 20600.8528  25720.037 33324.4929
married               YES         NO        YES        YES        YES        YES         NO
children           1.0117     2.0139     0.6265     0.6761     1.6207      0.899     0.9478
car                    NO         NO         NO         NO         NO        YES        YES
save_act              YES        YES        YES         NO         NO         NO        YES
current_act           YES        YES        YES        YES        YES        YES        YES
mortgage               NO         NO         NO         NO         NO        YES         NO
pep                    NO         NO         NO        YES         NO        YES        YES



Time taken to build model (full training data) : 0.03 seconds

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

Clustered Instances

0       72 ( 12%)
1      166 ( 28%)
2       71 ( 12%)
3       58 ( 10%)
4       99 ( 17%)
5      134 ( 22%)
```

5)After selecting “use training set” click on start and open the results in seperate window.The result window shows the centroid of each cluster as well as statistics on the number and percentage of instances assigned to different clusters.

RESULTS 1

Within cluster sum of squared errors: 1360.8718391528164
Time taken to build model (full training data) : 0.03 seconds

K means algorithm using TANAGRA

This example illustrates the use of *k-means* clustering with TANAGRA The sample data set used for this example is based on the "bank data" available in "bank.arff" and includes 600 instances.

As an illustration of performing clustering in TANAGRA, we will use its implementation of the K-means algorithm to cluster the customers in this bank data set, and to characterize the resulting customer segments.

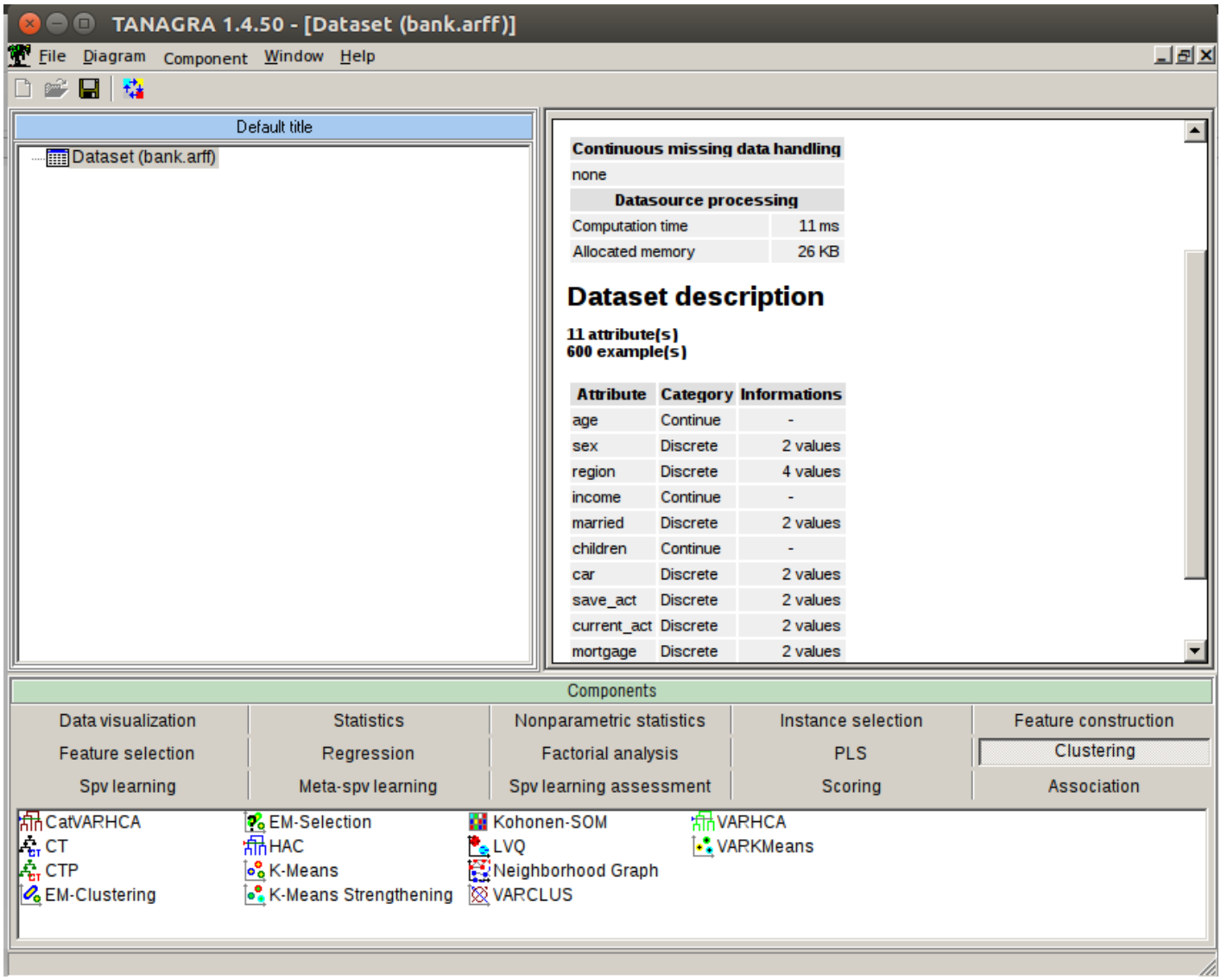


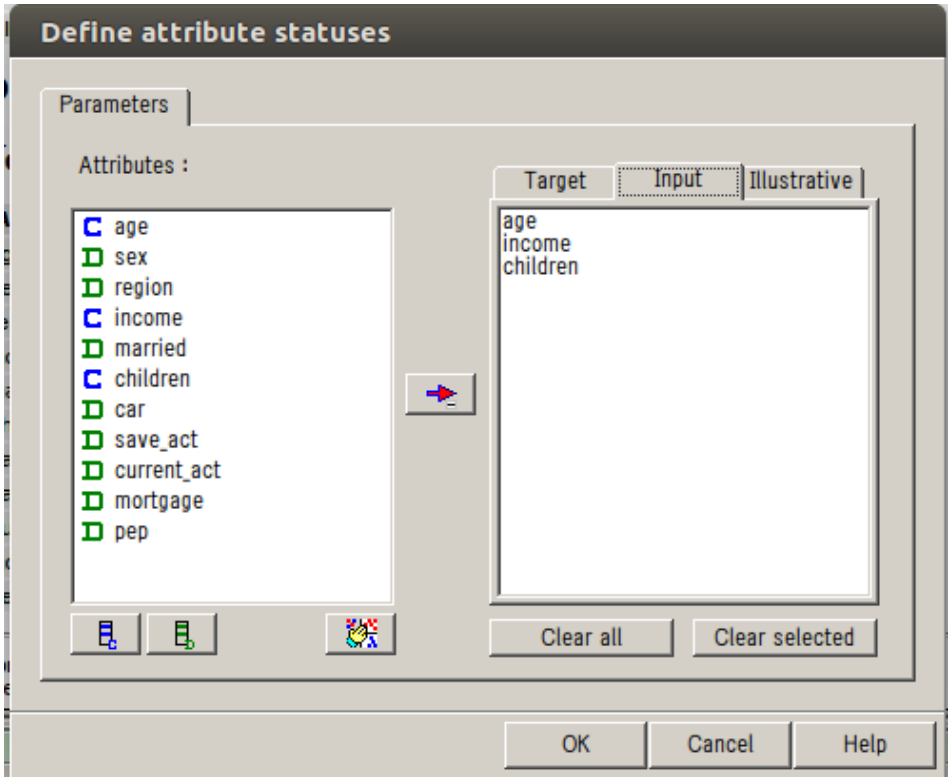
Fig1 :shows the main TANAGRA Explorer interface with the data file loaded.

2.1 Importing the dataset

1) Load the file .(.txt or .arff format).

2.2 Descriptive statistics

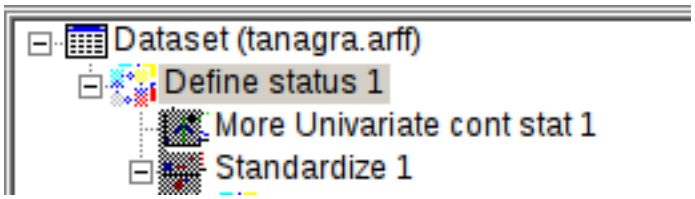
We want to obtain an overview of the main characteristics of the dataset.We add a DEFINE STATUS component into the diagram.We set all the continuous variables as INPUT. These are the active variables of the analysis i.e.they are used during the clustering process.



We add the MORE UNIVARIATE CONT STAT component (STATISTICS tab).We click on the contextual VIEW menu.

2.3 Standardising the active variable

The aim is to eliminate the discrepancy of scales between the variables . We add the STANDARDIZE component(FEATURE CONSTRUCTION tab) into the diagram. Then, we click on the VIEW menu.



2.4 K -means

We insert a new DEFINE STATUS and then insert k means from CLUSTERING tab.
Set the following parameter

K-Means 1

Parameters

K-Means parameters	
Clusters	6
Max Iteration	16
Trials	5
Distance normalization	none
Average computation	McQueen
Seed random generator	Standard

no. of clusters : 6

Results

Global evaluation

Within Sum of Squares	398.0911
Total Sum of Squares	1796.9999
R-Square	0.7785

no. of iterations :16

Cluster size and WSS

Clusters	6		
Cluster	**Description**	**Size**	**WSS**
cluster n°1	c_kmeans_1	61	55.3228
cluster n°2	c_kmeans_2	149	81.6789
cluster n°3	c_kmeans_3	85	51.2259
cluster n°4	c_kmeans_4	109	44.7132
cluster n°5	c_kmeans_5	93	87.9313
cluster n°6	c_kmeans_6	103	77.2190

distance normalisation :none

R-Square for each attempt

Number of trials	5
Trial	**R-square**
1	0.770089
2	0.777430
3	0.777005
4	0.768247
5	0.778469

Cluster centroids

Attribute	Cluster n°1	Cluster n°2	Cluster n°3	Cluster n°4	Cluster n°5	Cluster n°6
std_age_1	1.379560	-0.119040	-1.019945	-1.209713	0.606226	0.929697
std_income_1	1.971506	-0.247545	-0.809050	-1.001637	0.350956	0.601272
std_children_1	0.113064	-0.627085	1.258109	-0.653479	1.311739	-0.690903

Use GROUP CHARACTERIZATION for detailed comparisons

Computation time : 16 ms.
Created at 8/31/2016 2:44:24 AM

5)The result window shows the centroid of each cluster as well as statistics on the number and percentage of instances assigned to different clusters.

Within sum of Squares : 398.0911

Total Sum of Squares : 1796.999
BSS between sum of square : 1796.999 – 398.0911 = 1398.9079

RESULTS 2

Within cluster sum of squared errors: 1398.9079
Time taken to build model (full training data) : 0.027 seconds

CONCLUSION

Weka: Within cluster sum of squared errors: 1360.8718391528164
Time taken to build model (full training data) : 0.030 seconds

Tanagra : Within cluster sum of squared errors: 1398.9079
Time taken to build model (full training data) : 0.027 seconds

Hence for this dataset ‘Weka’ gives better result in terms of accuracy whereas ‘Tanagra’ produces faster results .

EXPERIMENT 3

To implement Hierarchical Clustering Algorithm using Weka and Tanagra.

HIERARCHICAL CLUSTERING ALGORITHM

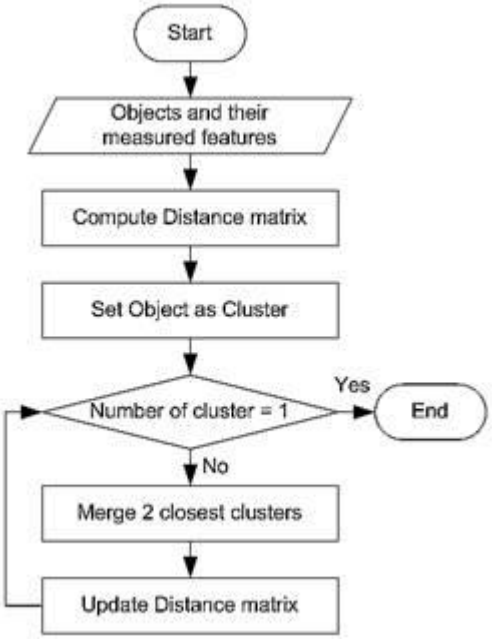
Introduction

There are two main methods of hierarchical clustering algorithm.

First method is ***agglomerative approach*** , where we start from the bottom where all the objects are and going up (*bottom up approach*) through merging of objects. We begin with each individual objects and merge the two closest objects. The process is iterated until all objects are aggregated into a single group.
Second method is ***divisive approach*** *(top down approach)* , where we start with assumption that all objects are grouped into a single group and then we split the group into two recursively until each group consists of a single object. One possible way to perform divisive approach is to first form a minimum spanning tree (e.g using Kruskal algorithm) and then recursively (or iteratively) split the tree by the largest distance.

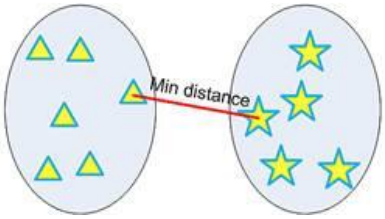
Step by step algorithm of agglomerative approach to compute hierarchical clustering is as follow:
1. Convert object features to distance matrix.
2. Set each object as a cluster (thus if we have 6 objects, we will have 6 clusters in the beginning)
3. Iterate until number of cluster is 1
    1. Merge two closest clusters
    2. Update distance matrix

The flowchart of agglomerative hierarchical clustering algorithm is given below



Numerical Example of Hierarchical Clustering

Minimum distance clustering is also called as single linkage hierarchical clustering or nearest neighbor clustering. Distance between two clusters is defined by the minimum distance between objects of the two clusters, as shown below.



For example, we have given an input distance matrix of size 6 by 6. This distance matrix was calculated based on the object features as explained in the previous section.

Dist	A	B	C	D	E	F
A	0.00	0.71	5.66	3.61	4.24	3.20
B	0.71	0.00	4.95	2.92	3.54	2.50
C	5.66	4.95	0.00	2.24	1.41	2.50
D	3.61	2.92	2.24	0.00	1.00	0.50
E	4.24	3.54	1.41	1.00	0.00	1.12
F	3.20	2.50	2.50	0.50	1.12	0.00

We have 6 objects and we put each object into one cluster (analogue to put a ball into a basket). Instead of calling them as *objects* , now we call them *clusters* . Thus, in the beginning we have 6 clusters. Our goal is to group those 6 clusters such that at the end of the iterations, we will have only single cluster consists of the whole six original objects.

In each step of the iteration, we find the closest pair clusters. In this case, the closest cluster is between cluster F and D with shortest distance of 0.5. Thus, we group cluster D and F into cluster (D, F). Then we update the distance matrix (see distance matrix below). Distance between ungrouped clusters will not change from the original distance matrix. Now the problem is how to calculate distance between newly grouped clusters (D, F) and other clusters?

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	?	4.24
B	0.71	0.00	4.95	?	3.54
C	5.66	4.95	0.00	?	1.41
D, F	?	?	?	0.00	?
E	4.24	3.54	1.41	?	0.00

That is exactly where the linkage rule comes into effect. Using single linkage, we specify minimum distance between original objects of the two clusters.

Using the input distance matrix, distance between cluster (D, F) and cluster A is computed as

$d_{(D,F)\mapsto A} = \min\left(d_{DA}, d_{FA}\right) = \min\left(3.61, 3.20\right) = 3.20$

Distance between cluster (D, F) and cluster B is

$d_{(D,F)\mapsto B} = \min\left(d_{DB}, d_{FB}\right) = \min\left(2.92, 2.50\right) = 2.50$

Similarly, distance between cluster (D, F) and cluster C is

$d_{(D,F)\mapsto C} = \min\left(d_{DC}, d_{FC}\right) = \min\left(2.24, 2.50\right) = 2.24$

Finally, distance between cluster E and cluster (D, F) is calculated as

$d_{E\to(D,F)} = \min\left(d_{ED}, d_{EF}\right) = \min\left(1.00, 1.12\right) = 1.00$

Then, the updated distance matrix becomes

Min Distance (Single Linkage)

Dist	A	B	C	D, F	E
A	0.00	0.71	5.66	3.20	4.24
B	0.71	0.00	4.95	2.50	3.54
C	5.66	4.95	0.00	2.24	1.41
D, F	**3.20**	**2.50**	**2.24**	0.00	1.00
E	4.24	3.54	1.41	**1.00**	0.00

Looking at the lower triangular updated distance matrix, we found out that the closest distance between cluster B and cluster A is now 0.71. Thus, we group cluster A and cluster B into a single cluster name (A, B).

Now we update the distance matrix. Aside from the first row and first column, all the other elements of the new distance matrix are not changed.

Dist	A,B	C	(D, F)	E
A,B	0	?	?	?
C	?	0	2.24	1.41
(D, F)	?	2.24	0	1.00
E	?	1.41	1.00	0

Using the input distance matrix (size 6 by 6), distance between cluster C and cluster (D, F) is computed as

$d_{C\to(A,B)} = \min\left(d_{CA}, d_{CB}\right) = \min\left(5.66, 4.95\right) = 4.95$

Distance between cluster (D, F) and cluster (A, B) is the minimum distance between all objects involved in the two clusters

$d_{(D,F)\mapsto(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}\right) = \min\left(3.61, 2.92, 3.20, 2.50\right) = 2.50$

Similarly, distance between cluster E and (A, B) is

$d_{E\to(A,B)} = \min\left(d_{EA}, d_{EB}\right) = \min\left(4.24, 3.54\right) = 3.54$

Then the updated distance matrix is

Min Distance (Single Linkage)

Dist	A,B	C	(D, F)	E
A,B	0	4.95	2.50	3.54
C	**4.95**	0	2.24	1.41
(D, F)	**2.50**	2.24	0	1.00
E	**3.54**	1.41	1.00	0

Observing the lower triangular of the updated distance matrix, we can see that the closest distance between clusters happens between cluster E and (D, F) at distance 1.00. Thus, we cluster them together into cluster ((D, F), E).
The updated distance matrix is given below.

Min Distance (Single Linkage)

Dist	(A,B)	C	(D, F), E
(A,B)	0.00	4.95	2.50
C	4.95	0.00	1.41
(D, F), E	**2.50**	**1.41**	0.00

Distance between cluster ((D, F), E) and cluster (A, B) is calculated as

$$d_{((D,F),E)\mapsto(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}, d_{EA}, d_{EB}\right) = \min\left(3.61, 2.92, 3.20, 2.50, 4.24, 3.54\right) = 2.50$$

Distance between cluster ((D, F), E) and cluster C yields the minimum distance of 1.41. This distance is computed as $d_{((D,F),E)\mapsto C} = \min\left(d_{DC}, d_{FC}, d_{EC}\right) = \min\left(2.24, 2.50, 1.41\right) = 1.41$. After that, we merge cluster ((D, F), E) and cluster C into a new cluster name (((D, F), E), C).
The updated distance matrix is shown in the figure below

Min Distance (Single Linkage)

Dist	(A,B)	(D, F), E),C
(A,B)	0.00	2.50
((D, F), E),C	**2.50**	0.00

The minimum distance of 2.5 is the result of the following computation

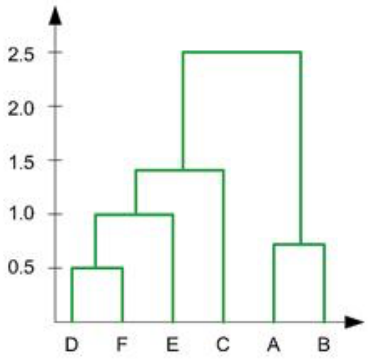
$$d_{(((D,F),E),C)\mapsto(A,B)} = \min\left(d_{DA}, d_{DB}, d_{FA}, d_{FB}, d_{EA}, d_{EB}, d_{CA}, d_{CB}\right)$$

$$d_{(((D,F),E),C)\mapsto(A,B)} = \min\left(3.61, 2.92, 3.20, 2.50, 4.24, 3.54, 5.66, 4.95\right) = 2.50$$

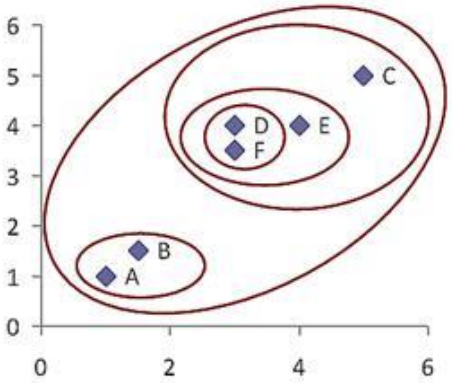
Now if we merge the remaining two clusters, we will get only single cluster contain the whole 6 objects. Thus, our computation is finished. We summarized the results of computation as follow:

1. In the beginning we have 6 clusters: A, B, C, D, E and F
2. We merge cluster D and F into cluster (D, F) at distance **0.50**
3. We merge cluster A and cluster B into (A, B) at distance **0.71**
4. We merge cluster E and (D, F) into ((D, F), E) at distance **1.00**
5. We merge cluster ((D, F), E) and C into (((D, F), E), C) at distance **1.41**
6. We merge cluster (((D, F), E), C) and (A, B) into ((((D, F), E), C), (A, B)) at distance **2.50**
7. The last cluster contain all the objects, thus conclude the computation

Using this information, we can now draw the final results of a dendrogram. The dendrogram is drawn based on the distances to merge the clusters above.



The hierarchy is given as (((D, F), E),C), (A,B). We can also plot the clustering hierarchy into XY space



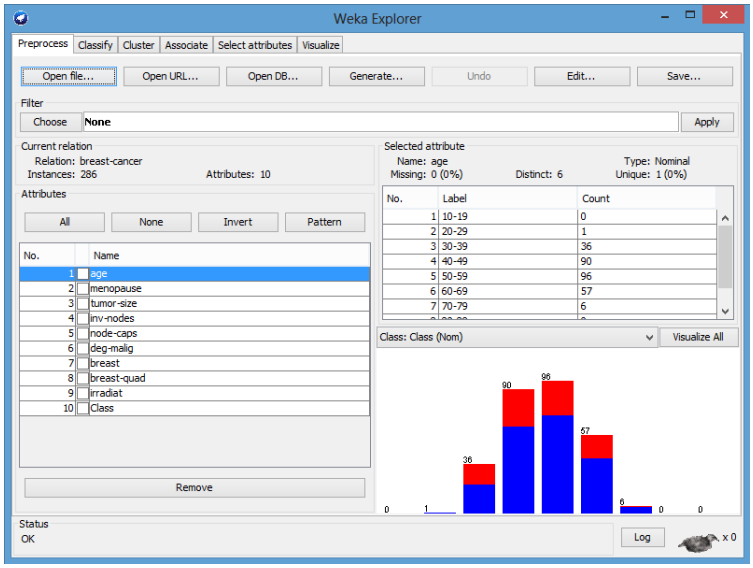
	X1	X2
A	1	1
B	1.5	1.5
C	5	5
D	3	4
E	4	4
F	3	3.5

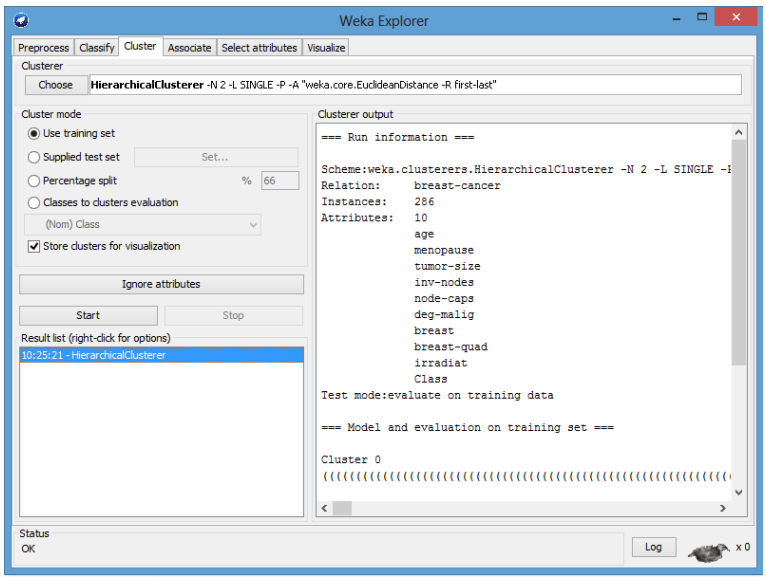
Data Set

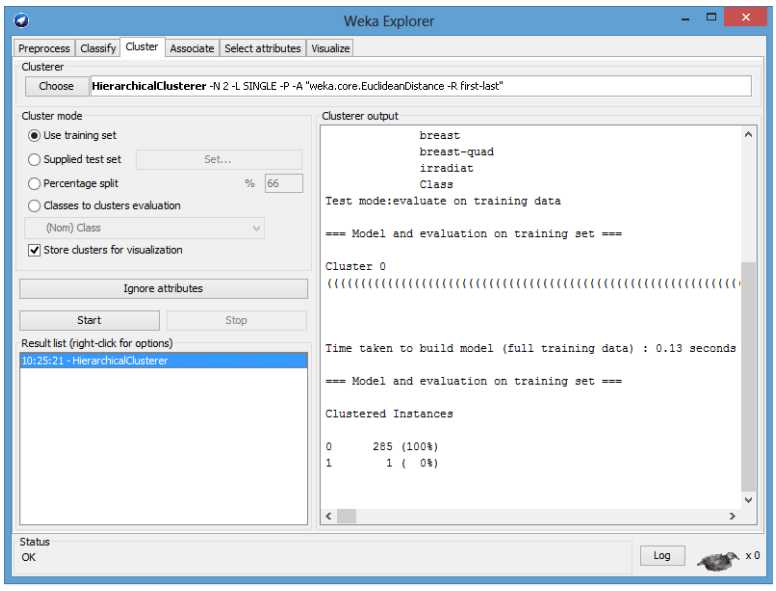
	clump	ucellsize	ucellshape	mgadhesion	sepics	bnuclei	bchromatin	normnucl	mitoses	class
1	4	2	2	1	2	1	2	1	1	begnin
2	1	1	1	1	2	1	2	1	1	begnin
3	2	1	1	1	2	1	2	1	1	begnin
4	10	6	6	2	4	10	9	7	1	malignant
5	4	1	1	1	2	1	2	1	1	begnin
6	1	1	1	1	2	1	1	1	1	begnin
7	1	1	1	1	2	1	2	1	1	begnin
8	5	1	1	1	2	1	2	1	1	begnin
9	3	1	1	1	2	1	2	1	1	begnin
10	1	1	1	1	2	4	2	1	1	begnin
11	5	3	3	2	3	1	3	1	1	begnin
12	4	2	2	1	2	1	2	1	1	begnin
13	1	1	1	1	2	1	2	1	1	begnin
14	2	1	1	1	2	1	2	1	1	begnin
15	4	1	1	1	2	1	3	2	1	begnin
16	1	1	1	1	2	1	3	1	1	begnin
17	6	1	3	2	2	1	1	1	1	begnin
18	3	1	1	2	3	4	1	1	1	begnin
19	2	1	1	1	2	1	3	1	1	begnin
20	5	5	5	8	10	8	7	3	7	malignant
21	1	1	1	1	2	1	3	1	1	begnin
22	1	1	1	1	2	1	2	1	2	begnin

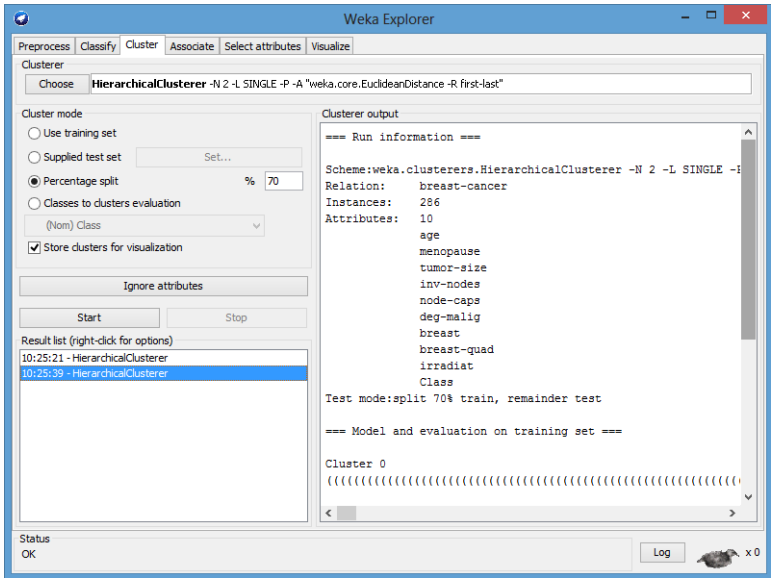
	clump	ucellsize	ucellshape	mgadhesion	sepics	bnuclei	bchromatin	normnucl	mitoses	class
23	1	1	1	1	1	1	1	1	1	begnin
24	9	8	8	9	6	3	4	1	1	malignant
25	1	1	1	1	2	1	3	1	1	begnin
26	1	1	3	1	2	1	1	1	1	begnin
27	8	7	8	5	10	10	7	2	1	malignant
28	5	2	2	2	2	1	1	1	2	begnin
29	1	1	1	1	2	1	1	1	1	begnin
30	10	10	8	6	4	5	8	10	1	malignant
31	4	1	1	3	2	1	3	1	1	begnin
32	2	1	1	1	2	1	2	1	1	begnin
33	6	1	3	1	4	5	5	10	1	malignant
34	4	1	1	2	2	1	2	1	1	begnin
35	7	4	5	10	2	10	3	8	2	malignant
36	5	1	1	1	2	1	3	1	1	begnin
37	4	1	1	1	2	1	3	1	1	begnin
38	1	3	1	2	2	2	3	3	2	begnin
39	1	1	1	1	2	4	1	1	1	begnin
40	1	1	1	1	2	1	1	1	1	begnin
41	3	1	1	1	2	1	2	1	1	begnin
42	7	8	8	7	3	10	7	2	3	malignant
43	10	4	3	6	4	10	7	8	4	malignant
44	4	2	1	1	2	1	2	1	1	begnin

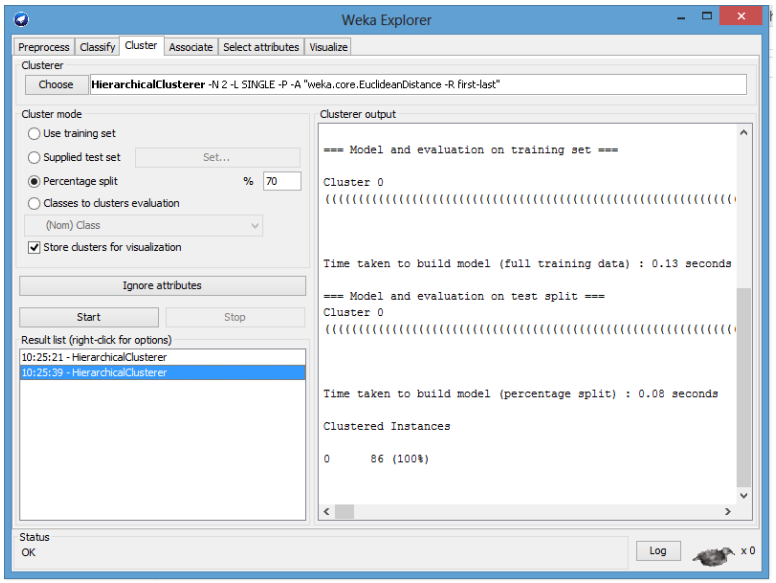
Implementation Using WEKA

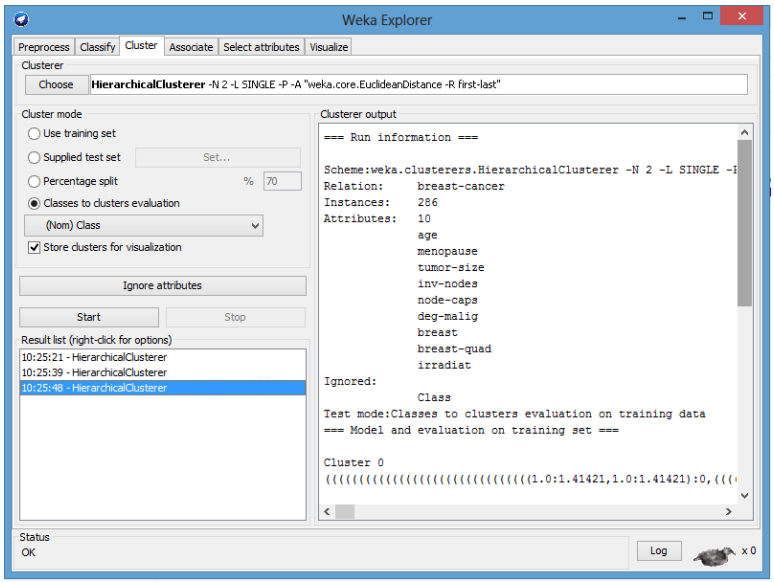

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Open file...
Open URL...
Open DB...
Generate...
Undo
Edit...
Save...
Filter
Choose
None
Apply
Current relation
Relation: breast-cancer
Instances: 286
Attributes: 10
Selected attribute
Name: age
Missing: 0 (0%)
Distinct: 6
Type: Nominal
Unique: 1 (0%)
Attributes
All
None
Invert
Pattern
No.
Name
1 age
2 menopause
3 tumor-size
4 inv-nodes
5 node-caps
6 deg-malig
7 breast
8 breast-quad
9 irradiat
10 Class
Remove
No.
Label
Count
1 10-19 0
2 20-29 1
3 30-39 36
4 40-49 90
5 50-59 96
6 60-69 57
7 70-79 6
Class: Class (Nom)
Visualize All
Status
OK
Log
x 0

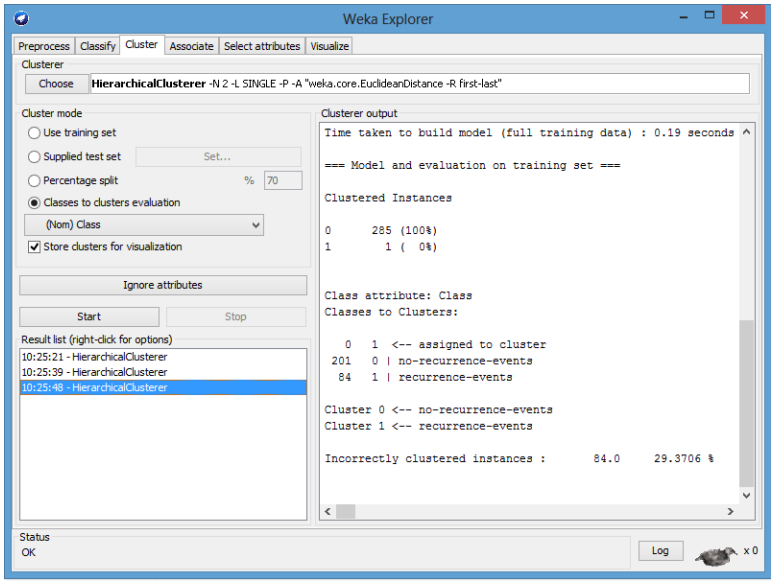

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
10:25:21 - HierarchicalClusterer
Clusterer output
=== Run information ===
Scheme:weka.clusterers.HierarchicalClusterer -N 2 -L SINGLE -
Relation: breast-cancer
Instances: 286
Attributes: 10
age
menopause
tumor-size
inv-nodes
node-caps
deg-malig
breast
breast-quad
irradiat
Class
Test mode:evaluate on training data
=== Model and evaluation on training set ===
Cluster 0
Status
OK
Log
x 0

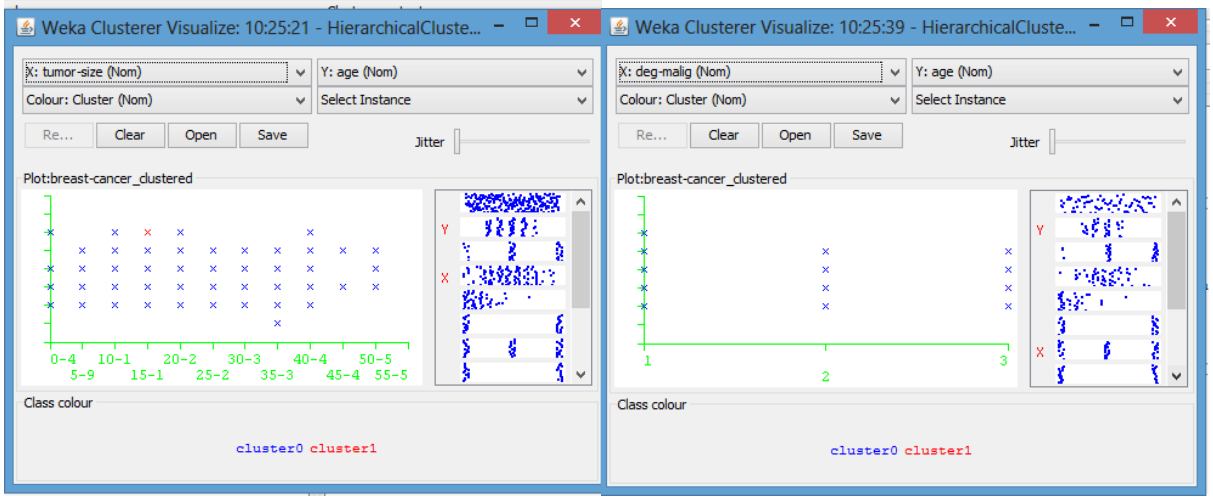

Weka Explorer
Preprocess
Classify
Cluster
Associate
Select attributes
Visualize
Clusterer
Choose
HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set
Set...
Percentage split
%
66
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start
Stop
Result list (right-click for options)
10:25:21 - HierarchicalClusterer
Clusterer output
breast
breast-quad
irradiat
Class
Test mode:evaluate on training data
=== Model and evaluation on training set ===
Cluster 0
Time taken to build model (full training data) : 0.13 seconds
=== Model and evaluation on training set ===
Clustered Instances
0 285 (100%)
1 1 (0%)
Status
OK
Log
x 0


Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 70
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
10:25:21 - HierarchicalClusterer
10:25:39 - HierarchicalClusterer
Clusterer output
=== Run information ===
Scheme:weka.clusterers.HierarchicalClusterer -N 2 -L SINGLE -
Relation: breast-cancer
Instances: 286
Attributes: 10
age
menopause
tumor-size
inv-nodes
node-caps
deg-malig
breast
breast-quad
irradiat
Class
Test mode:split 70% train, remainder test
=== Model and evaluation on training set ===
Cluster 0
Status
OK
Log
x 0

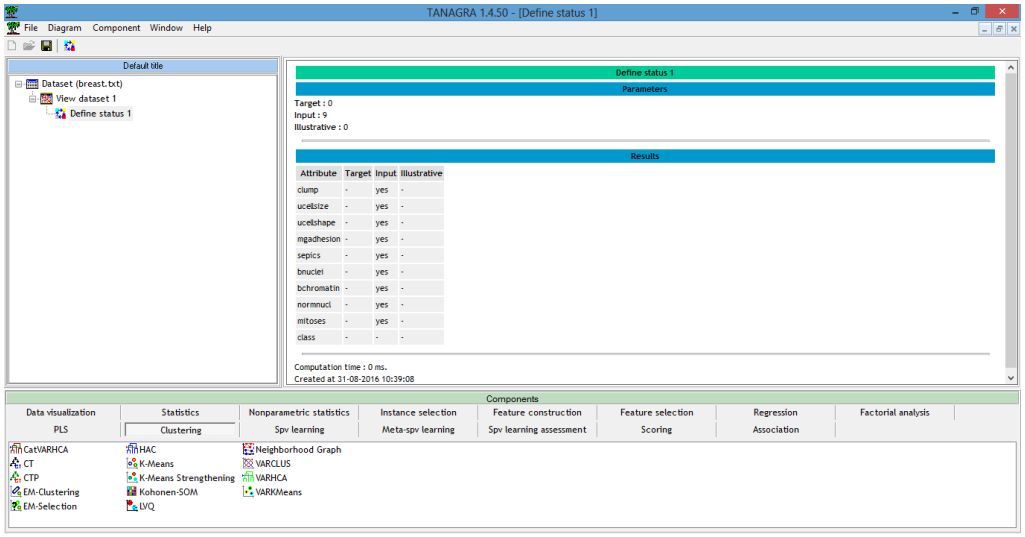

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 70
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
10:25:21 - HierarchicalClusterer
10:25:39 - HierarchicalClusterer
Clusterer output
=== Model and evaluation on training set ===
Cluster 0
Time taken to build model (full training data) : 0.13 seconds
=== Model and evaluation on test split ===
Cluster 0
Time taken to build model (percentage split) : 0.08 seconds
Clustered Instances
0 86 (100%)
Status
OK
Log
x 0

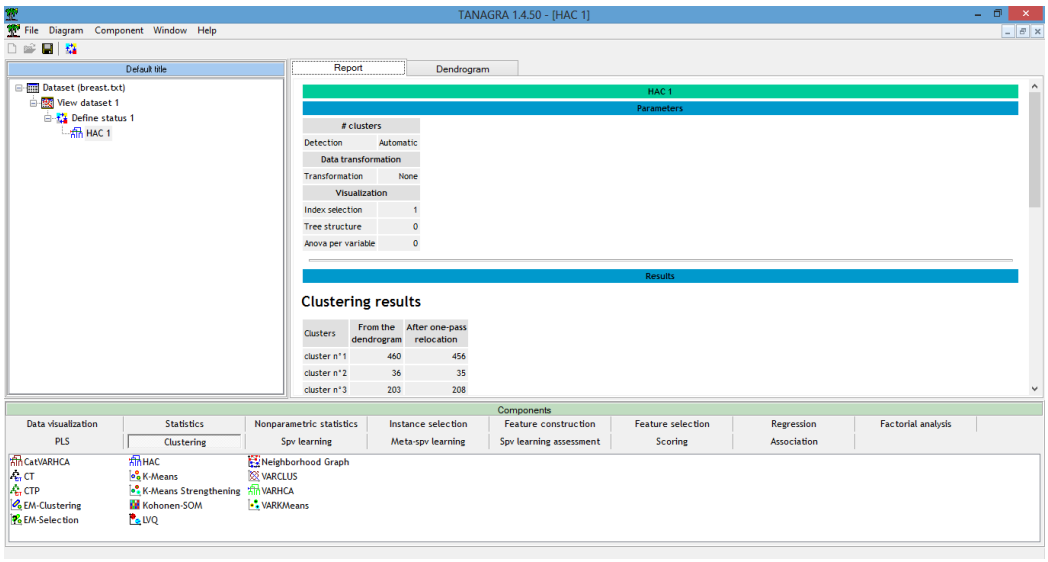

Weka Explorer
Preprocess Classify Cluster Associate Select attributes Visualize
Clusterer
Choose HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"
Cluster mode
Use training set
Supplied test set Set...
Percentage split % 70
Classes to clusters evaluation
(Nom) Class
Store clusters for visualization
Ignore attributes
Start Stop
Result list (right-click for options)
10:25:21 - HierarchicalClusterer
10:25:39 - HierarchicalClusterer
10:25:48 - HierarchicalClusterer
Clusterer output
=== Run information ===
Scheme:weka.clusterers.HierarchicalClusterer -N 2 -L SINGLE -
Relation: breast-cancer
Instances: 286
Attributes: 10
age
menopause
tumor-size
inv-nodes
node-caps
deg-malig
breast
breast-quad
irradiat
Ignored:
Class
Test mode:Classes to clusters evaluation on training data
=== Model and evaluation on training set ===
Cluster 0
(1.0:1.41421,1.0:1.41421):0,
Status
OK
Log
x 0

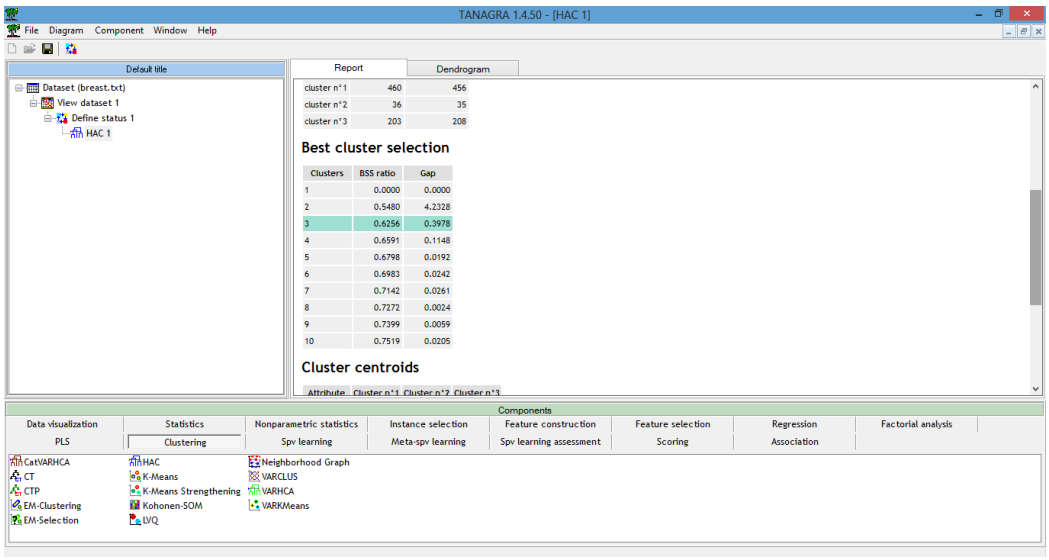


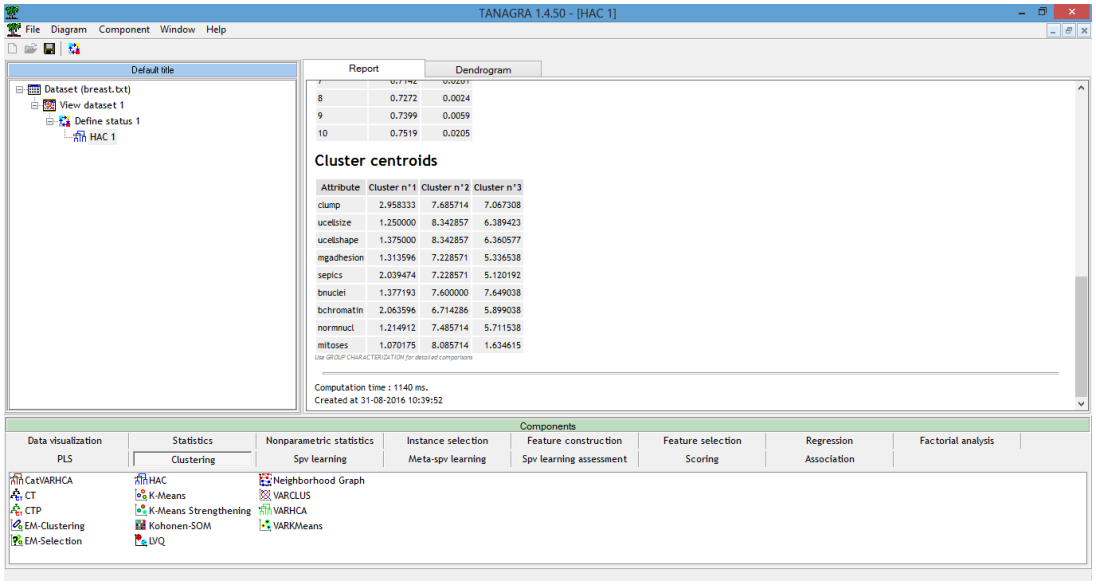


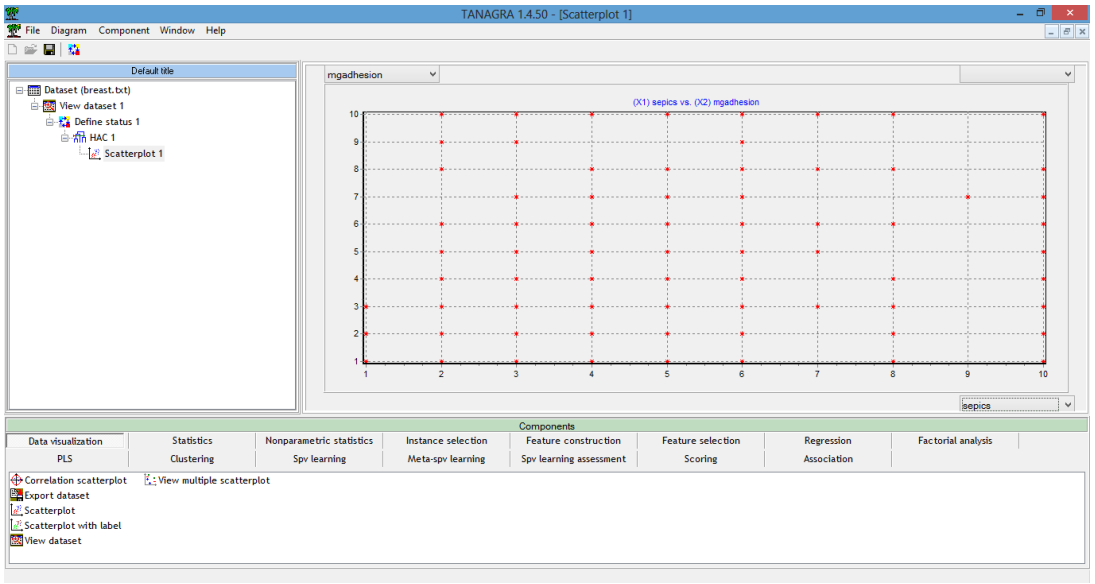
Implementation Using TANAGRA

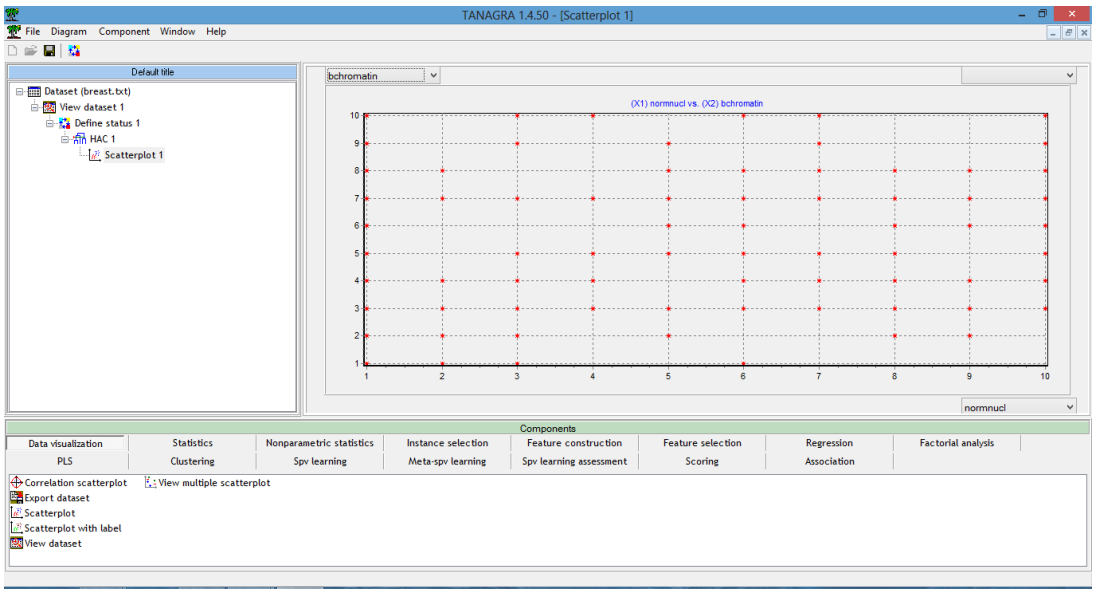

TANAGRA 1.4.50 - [Define status 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
Define status 1
Parameters
Target : 0
Input : 9
Illustrative : 0
Results
Attribute Target Input Illustrative
clump - yes -
ucellsize - yes -
ucellshape - yes -
mgadhesion - yes -
sepics - yes -
bnuclei - yes -
bchromatin - yes -
normnucl - yes -
mitoses - yes -
class - - -
Computation time : 0 ms.
Created at 31-08-2016 10:39:08
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
CatVARHCA HAC Neighborhood Graph
CT K-Means VARCLUS
CTP K-Means Strengthening VARHCA
EM-Clustering Kohonen-SOM VARKMeans
EM-Selection LVQ


TANAGRA 1.4.50 - [HAC 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
HAC 1
Report
Dendrogram
HAC 1
Parameters
clusters
Detection Automatic
Data transformation
Transformation None
Visualization
Index selection 1
Tree structure 0
Anova per variable 0
Results
Clustering results
Clusters From the dendrogram After one-pass relocation
cluster n°1 460 456
cluster n°2 36 35
cluster n°3 203 208
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
CatVARHCA HAC Neighborhood Graph
CT K-Means VARCLUS
CTP K-Means Strengthening VARHCA
EM-Clustering Kohonen-SOM VARKMeans
EM-Selection LVQ


TANAGRA 1.4.50 - [HAC 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
HAC 1
Report
Dendrogram
cluster n°1 460 456
cluster n°2 36 35
cluster n°3 203 208
Best cluster selection
Clusters BSS ratio Gap
1 0.0000 0.0000
2 0.5480 4.2328
3 0.6256 0.3978
4 0.6591 0.1148
5 0.6798 0.0192
6 0.6983 0.0242
7 0.7142 0.0261
8 0.7272 0.0024
9 0.7399 0.0059
10 0.7519 0.0205
Cluster centroids
Attribute Cluster n°1 Cluster n°2 Cluster n°3
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
CatVARHCA HAC Neighborhood Graph
CT K-Means VARCLUS
CTP K-Means Strengthening VARHCA
EM-Clustering Kohonen-SOM VARKMeans
EM-Selection LVQ


TANAGRA 1.4.50 - [HAC 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
HAC 1
Report
Dendrogram
8 0.7272 0.0024
9 0.7399 0.0059
10 0.7519 0.0205
Cluster centroids
Attribute Cluster n°1 Cluster n°2 Cluster n°3
clump 2.958333 7.685714 7.067308
ucellsize 1.250000 8.342857 6.389423
ucellshape 1.375000 8.342857 6.360577
mgadhesion 1.313596 7.228571 5.336538
sepics 2.039474 7.228571 5.120192
bnuclei 1.377193 7.600000 7.649038
bchromatin 2.063596 6.714286 5.899038
normnucl 1.214912 7.485714 5.711538
mitoses 1.070175 8.085714 1.634615
Use GROUP CHARACTERIZATION for detailed comparisons
Computation time : 1140 ms.
Created at 31-08-2016 10:39:52
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
CatVARHCA HAC Neighborhood Graph
CT K-Means VARCLUS
CTP K-Means Strengthening VARHCA
EM-Clustering Kohonen-SOM VARKMeans
EM-Selection LVQ


TANAGRA 1.4.50 - [Scatterplot 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
HAC 1
Scatterplot 1
mgadhesion
(X1) sepics vs. (X2) mgadhesion
sepics
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
Correlation scatterplot View multiple scatterplot
Export dataset
Scatterplot
Scatterplot with label
View dataset


TANAGRA 1.4.50 - [Scatterplot 1]
File Diagram Component Window Help
Default title
Dataset (breast.txt)
View dataset 1
Define status 1
HAC 1
Scatterplot 1
bchromatin
(X1) normnucl vs. (X2) bchromatin
normnucl
Components
Data visualization Statistics Nonparametric statistics Instance selection Feature construction Feature selection Regression Factorial analysis
PLS Clustering Spv learning Meta-spv learning Spv learning assessment Scoring Association
Correlation scatterplot View multiple scatterplot
Export dataset
Scatterplot
Scatterplot with label
View dataset

EXPERIMENT -4

NAIVE BAYES ALGORITHM WITH WEKA AND TANAGRA

Bayes Classification

$$P(C \mid \mathbf{X}) \propto P(\mathbf{X} \mid C)P(C) = P(X_1, \cdots, X_n \mid C)P(C)$$

Naive Bayes Classification

Assume that all input features are conditionally independent.

$$\begin{aligned} P(X_1, X_2, \cdots, X_n \mid C) &= P(X_1 \mid X_2, \cdots, X_n, C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2 \mid C) \cdots P(X_n \mid C) \end{aligned}$$

Map Classification Rule $\mathbf{x} = (x_1, x_2, \cdots, x_n)$

$$[P(x_1 \mid c^*) \cdots P(x_n \mid c^*)]P(c^*) > [P(x_1 \mid c) \cdots P(x_n \mid c)]P(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

- Algorithm: Discrete-Valued Features
  - Learning Phase: Given a training set **S** of F fe

For each target value of c_i ($c_i = c_1, \cdots, c_L$)

$\hat{P}(C = c_i) \leftarrow$ estimate $P(C = c_i)$ with examples in **S**;

For every feature value x_{jk} of each feature X_j ($j = 1, \cdots$

$\hat{P}(X_j = x_{jk} \mid C = c_i) \leftarrow$ estimate $P(X_j = x_{jk} \mid C = c$

Output: $F * L$ conditional probabilistic (genera

  - Test Phase: Given an unknown instance $\mathbf{X}' =$

"Look up tables" to assign the label c^* to $\mathbf{X}'$ i

$$[\hat{P}(a'_1 \mid c^*) \cdots \hat{P}(a'_n \mid c^*)]\hat{P}(c^*) > [\hat{P}(a'_1 \mid c) \cdots \hat{P}(a'_n \mid c)]\hat{P}(c)$$

- Example: Play Tennis

PlayTennis: training examples

Day	Outlook	Temperature	Humidity	Wind
D1	Sunny	Hot	High	Weak
D2	Sunny	Hot	High	Strong
D3	Overcast	Hot	High	Weak
D4	Rain	Mild	High	Weak
D5	Rain	Cool	Normal	Weak
D6	Rain	Cool	Normal	Strong
D7	Overcast	Cool	Normal	Strong
D8	Sunny	Mild	High	Weak
D9	Sunny	Cool	Normal	Weak
D10	Rain	Mild	Normal	Weak
D11	Sunny	Mild	Normal	Strong
D12	Overcast	Mild	High	Strong
D13	Overcast	Hot	Normal	Weak
D14	Rain	Mild	High	Strong

- Learning Phase

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=Y
Hot	2/9
Mild	4/9
Cool	3/9

Humidity	Play=*Yes*	Play=No
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=
Strong	3/9
Weak	6/9

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/$

- Test Phase
  - Given a new instance, predict its label

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*Hig*
  - Look up tables achieved in the learning phrase
  - Decision making with the MAP rule

P(Outlook=*Sunny*|Play=*Yes*) = 2/9

P(Temperature=*Cool*|Play=*Yes*) = 3/9

P(Huminity=*High*|Play=*Yes*) = 3/9

P(Wind=*Strong*|Play=*Yes*) = 3/9

P(Play=*Yes*) = 9/14

P(Outlook=*Sunny*|Pl

P(Temperature=*Cool*|

P(Huminity=*High*|Pl

P(Wind=*Strong*|Play=

P(Play=*No*) = 5/14

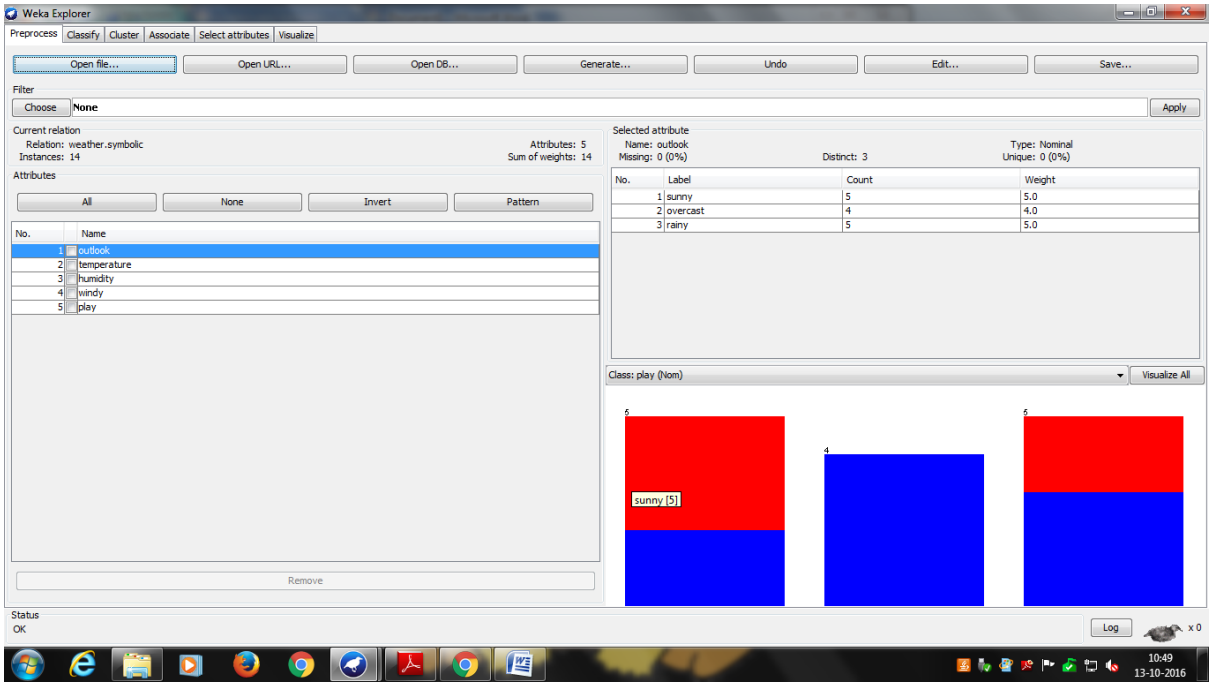
$P(Yes|\mathbf{x}') \approx$ [P(*Sunny*|Yes)P(*Cool*|*Yes*)P(*High*|Yes)P(*Strong*|*Ye*

0.0053

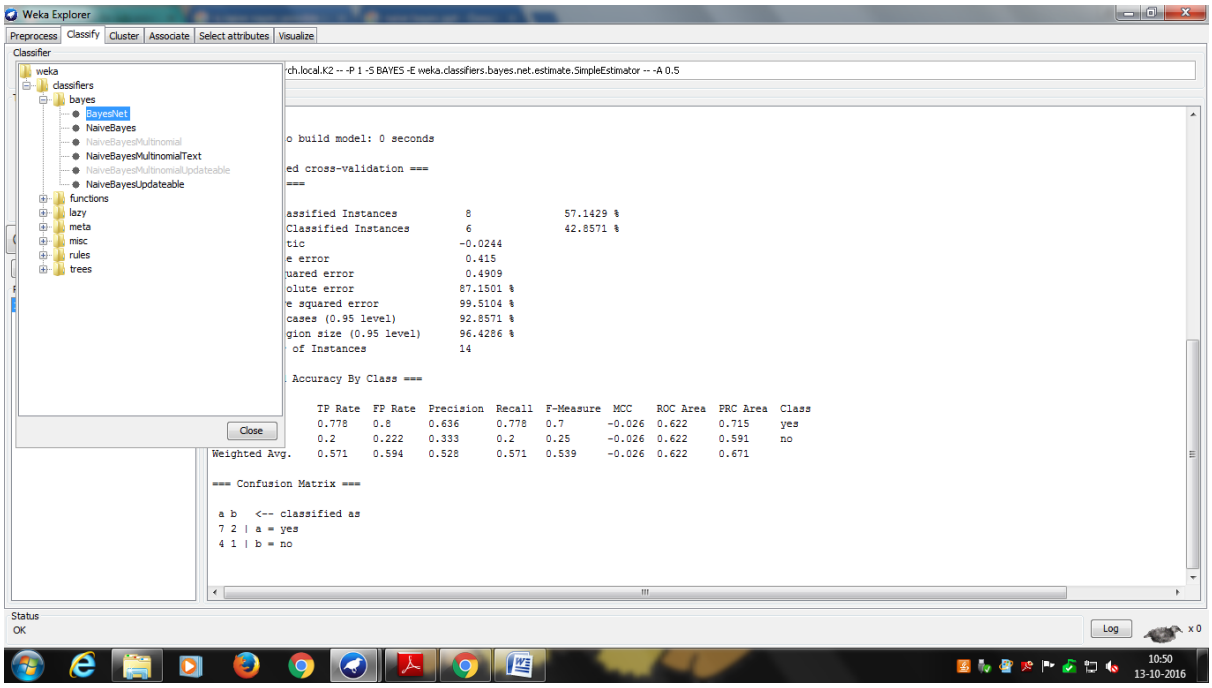
$P(No|\mathbf{x}') \approx$ [P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*

Given the fact $P(Yes|\mathbf{x}') < P(No|\mathbf{x}')$, we label $\mathbf{x}'$ to

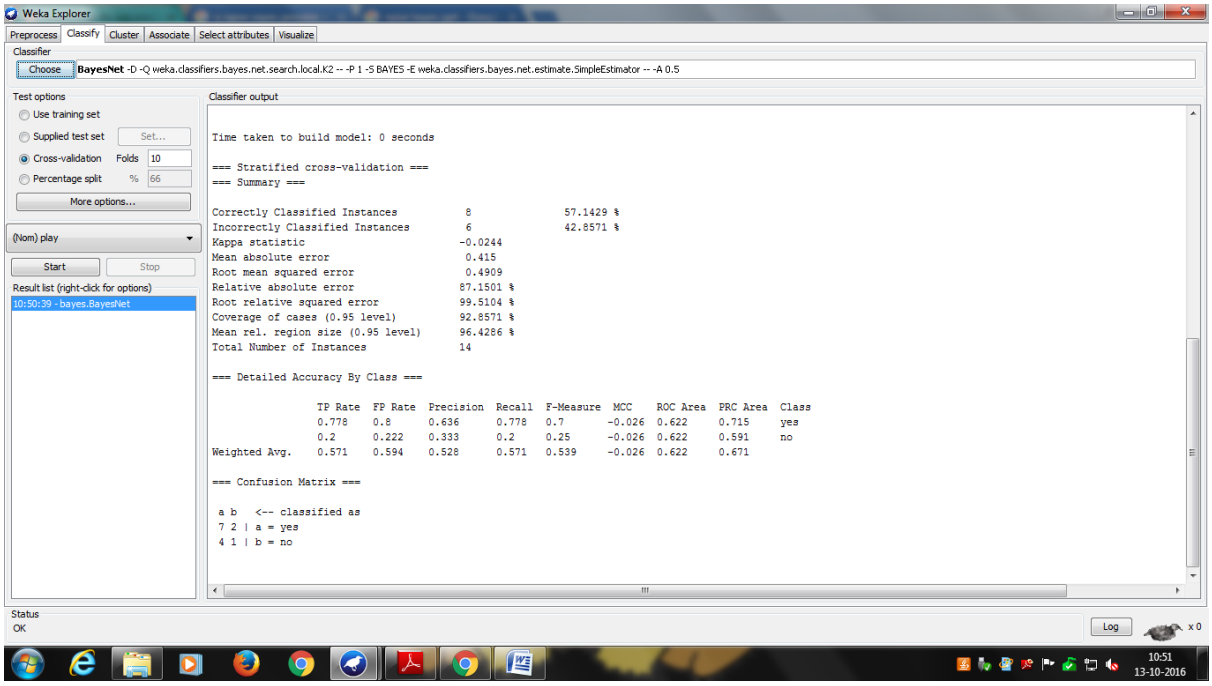
1) Weather Data Set



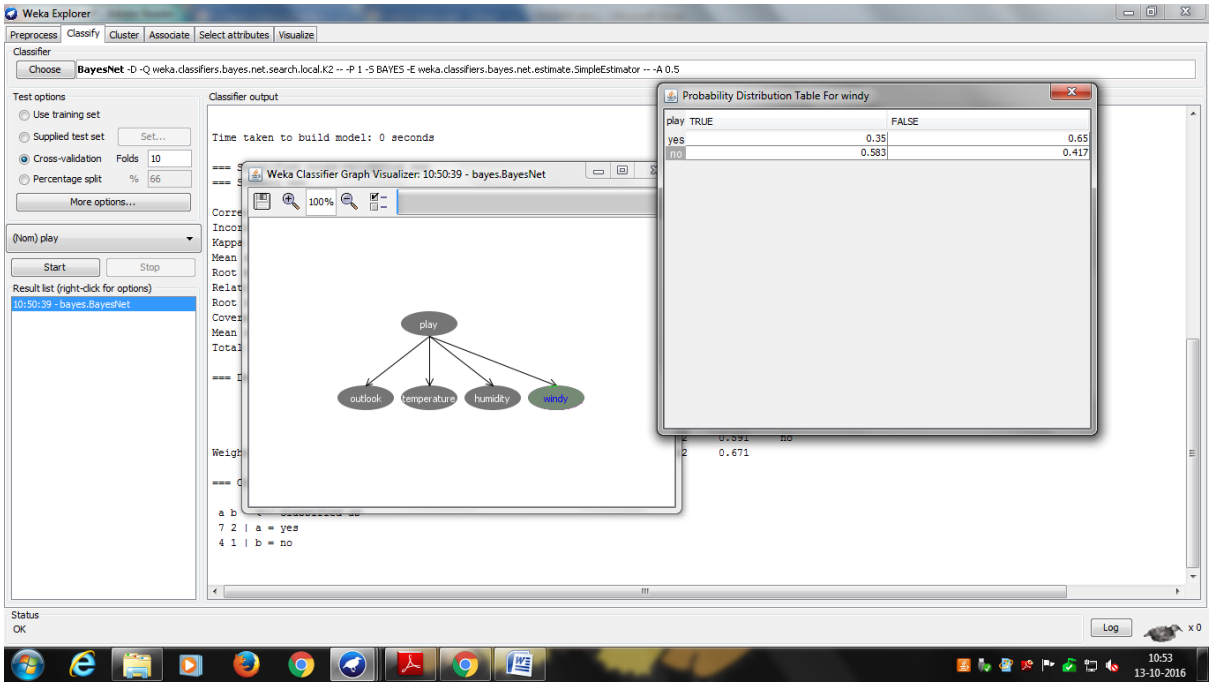
2) Applying Naive Bayes



3) NAIVE BAYES CLASSIFICATION

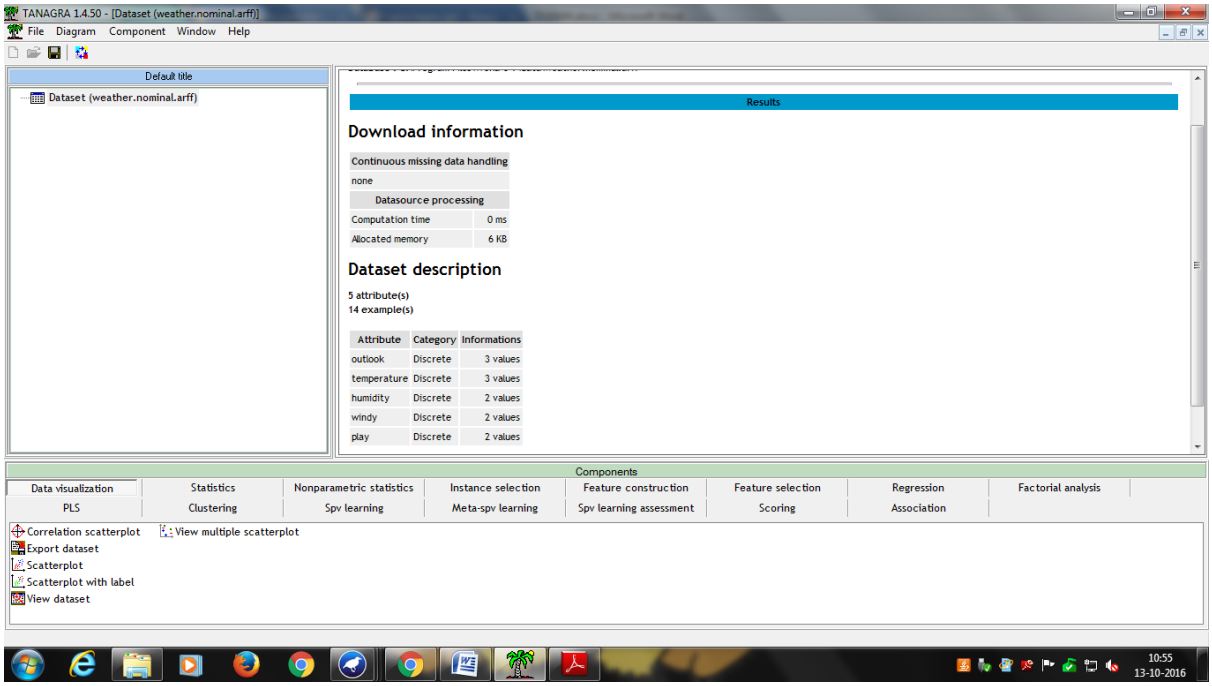


4) Visualise Graph

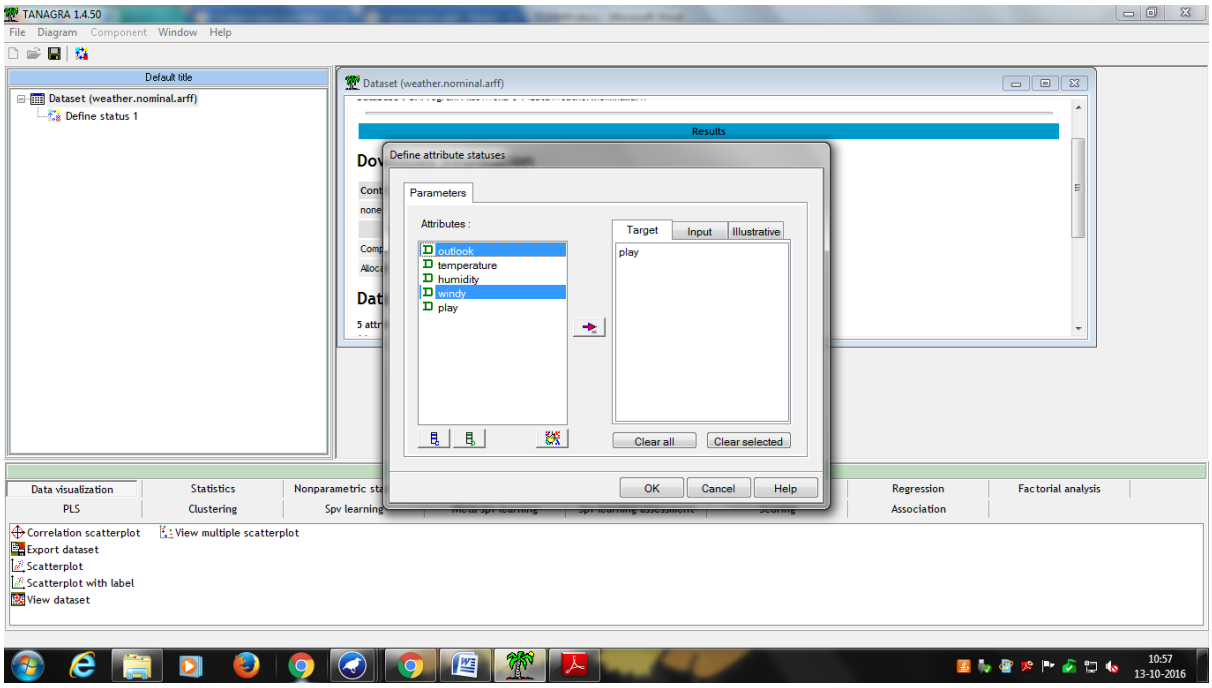


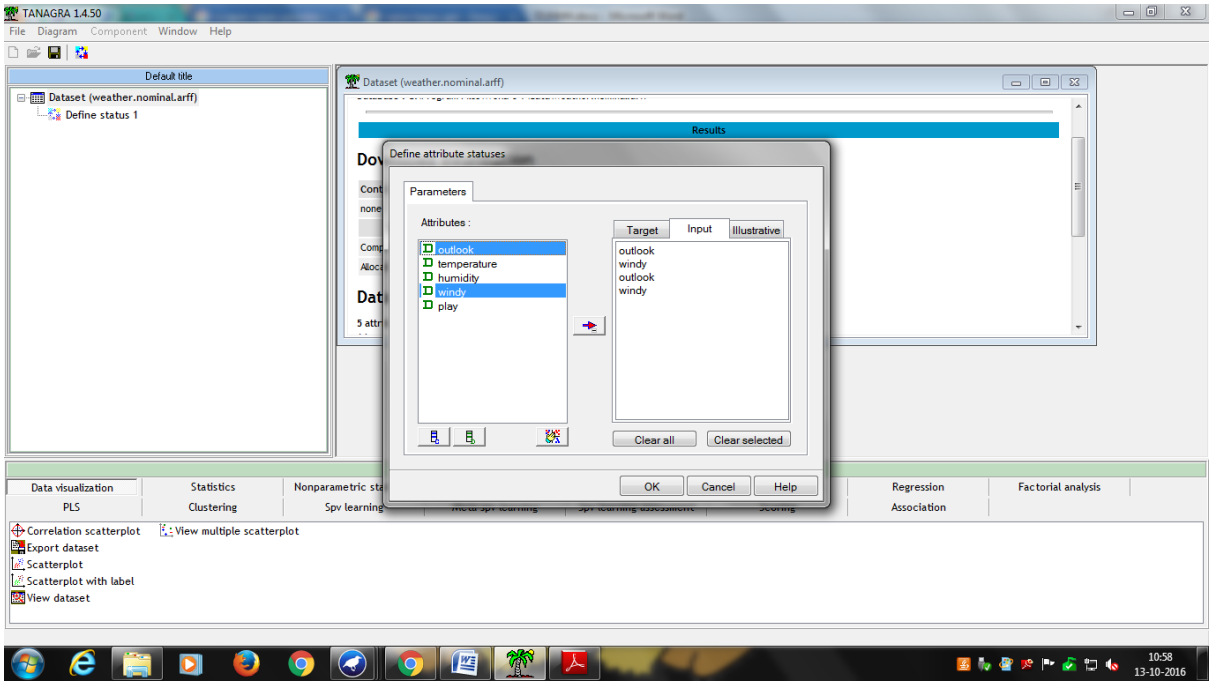
IMPLEMENTING NAIVE BAYES USING TANAGRA

1) Weather Data Set

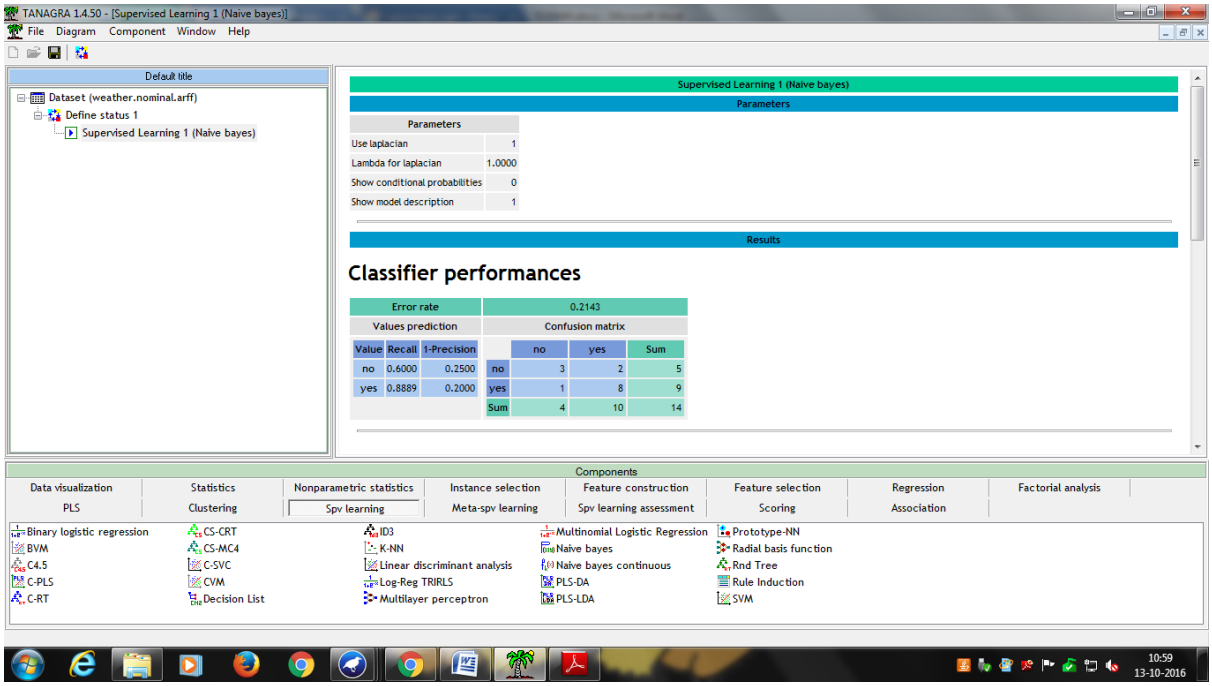


2) Applying Naive Bayes

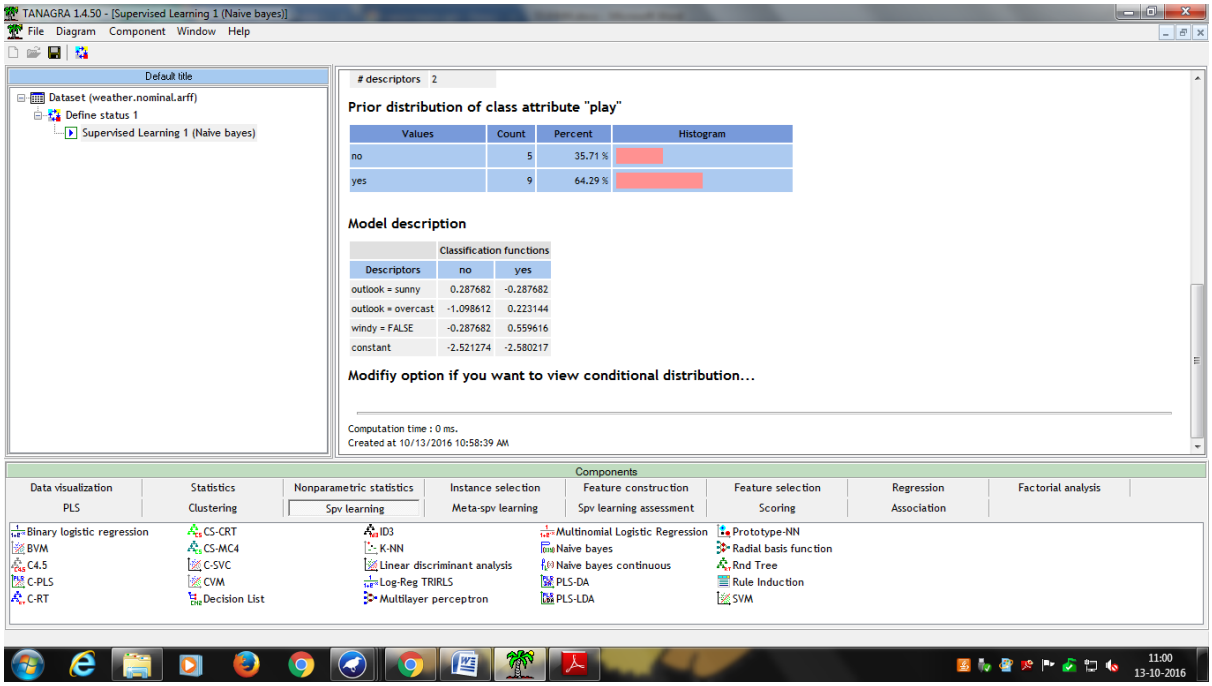




3) Naive Bayes Results



4) Model Description



COMPARISON BETWEEN WEKA AND TANAGRA RESULTS

	WEKA	TANAGRA
Time taken	0 seconds	0 milliseconds
Root mean squared error	0.4909	

EXPERIMENT-5

K – Nearest Neighbour

The ***k*-Nearest Neighbours algorithm** (or ***k*-NN** for short) is a non-parametric method used for classification and regression. In both cases, the input consists of the *k* closest training examples in the feature space. The output depends on whether *k*-NN is used for classification or regression:

- In *k-NN classification*, the output is a class membership. An object is classified by a majority vote of its neighbours, with the object being assigned to the class most common among its *k* nearest neighbours (*k* is a positive integer, typically small). If *k* = 1, then the object is simply assigned to the class of that single nearest neighbour.

- In *k-NN regression*, the output is the property value for the object. This value is the average of the values of its *k* nearest neighbours.

k-NN is a type of instance-based learning, or lazy learning, where the function is only approximated locally and all computation is deferred until classification. The *k*-NN algorithm is among the simplest of all machine learning algorithms.

Algorithm –

Here is step by step on how to compute K-nearest neighbors KNN algorithm:

1) Determine parameter K = number of nearest neighbors
2) Calculate the distance between the query-instance and all the training samples
3) Sort the distance and determine nearest neighbors based on the K-th minimum distance
4) Gather the category r of the nearest neighbours
5) Use simple majority of the category of nearest neighbors as the prediction value of the query instance

Numerical Example

We have data from the questionnaires survey (to ask people opinion) and objective testing with two attributes (acid durability and strength) to classify whether a special paper tissue is good or not. Here is four training samples

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Classification
7	7	Bad
7	4	Bad
3	4	Good
1	4	Good

Now the factory produces a new paper tissue that pass laboratory test with X1 = 3 and X2 = 7. Without another expensive survey, can we guess what the classification of this new tissue is? Fortunately, k nearest neighbor (KNN) algorithm can help you to predict this type of problem.

1. Determine parameter K = number of nearest neighbors

Suppose use K = 3

2. Calculate the distance between the query-instance and all the training samples

Coordinate of query instance is (3, 7), instead of calculating the distance we compute square distance which is faster to calculate (without square root)

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	**Square Distance to query instance (3, 7)**
7	7	
7	4	
3	4	
1	4	

3. *Sort the distance and determine nearest neighbors based on the K-th minimum distance*

X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Square Distance to query instance (3, 7)	**Rank minimum distance**	**Is it included in 3-Nearest neighbors?**
7	7		**3**	**Yes**
7	4		**4**	**No**
3	4		**1**	**Yes**
1	4		**2**	**Yes**

4. Gather the category of the nearest neighbors. Notice in the second row last column that the category of nearest neighbor (Y) is not included because the rank of this data is more than 3 (=K).

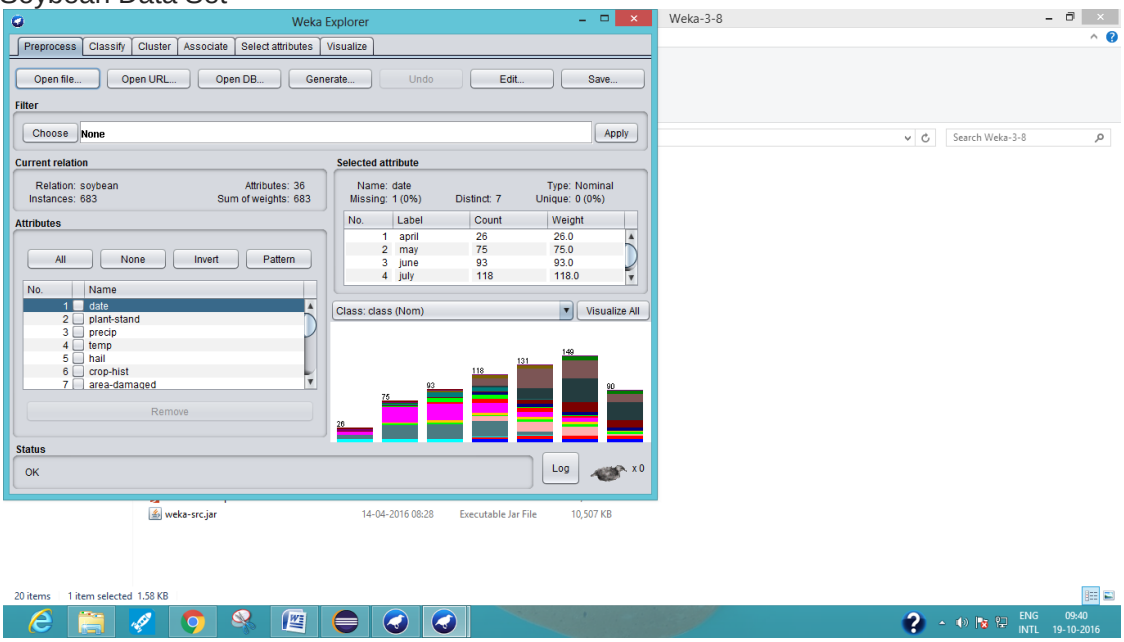
X1 = Acid Durability (seconds)	X2 = Strength (kg/square meter)	Square Distance to query instance (3, 7)	Rank minimum distance	Is it included in 3-Nearest neighbors?	**Y = Category of nearest Neighbor**
7	7		3	Yes	**Bad**
7	4		4	No	-
3	4		1	Yes	**Good**
1	4		2	Yes	**Good**

5. Use simple majority of the category of nearest neighbors as the prediction value of the query instance

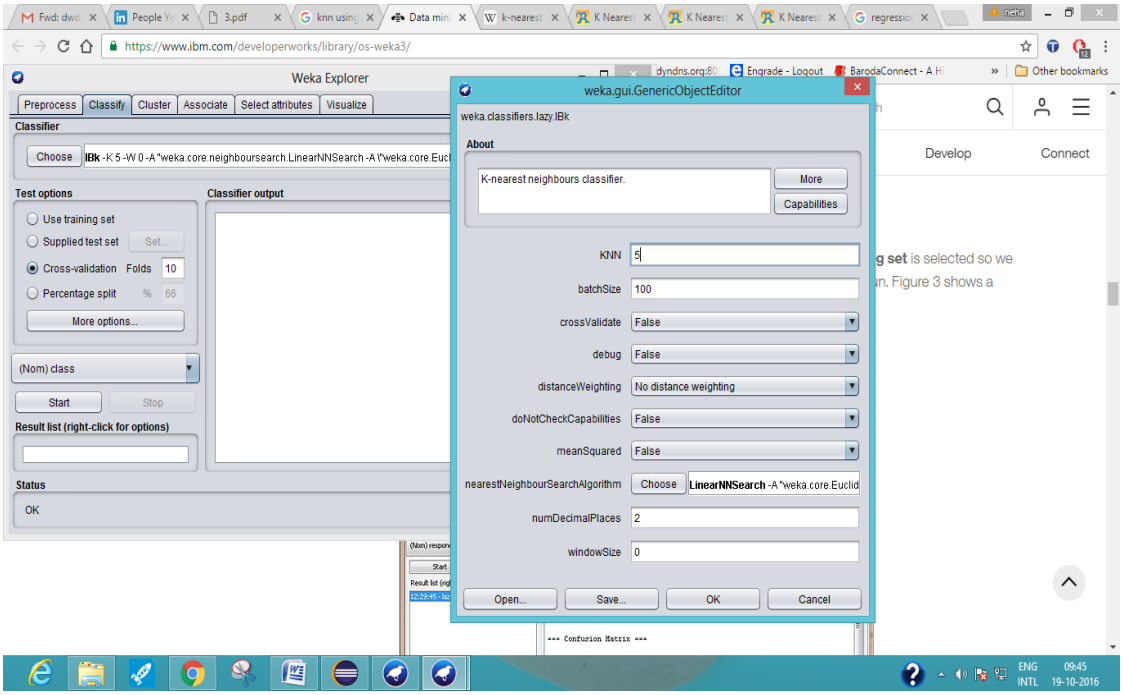
We have 2 good and 1 bad, since 2>1 then we conclude that a new paper tissue that pass laboratory test with X1 = 3 and X2 = 7 is included in **Good** category.

KNN USING WEKA

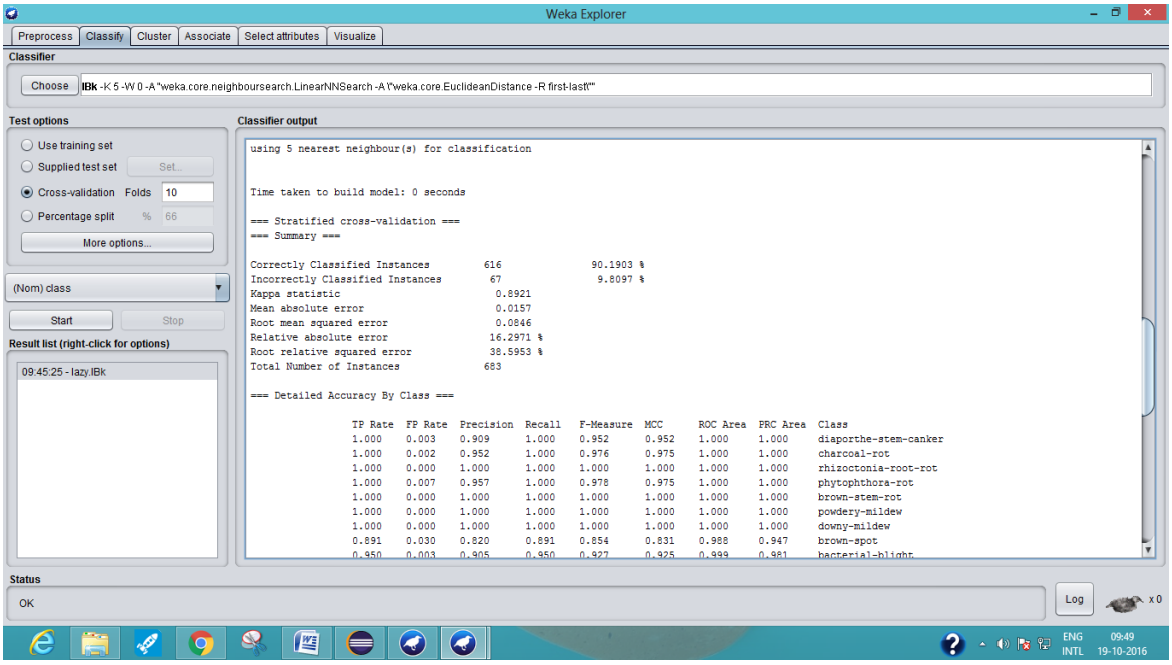
1) Soybean Data Set

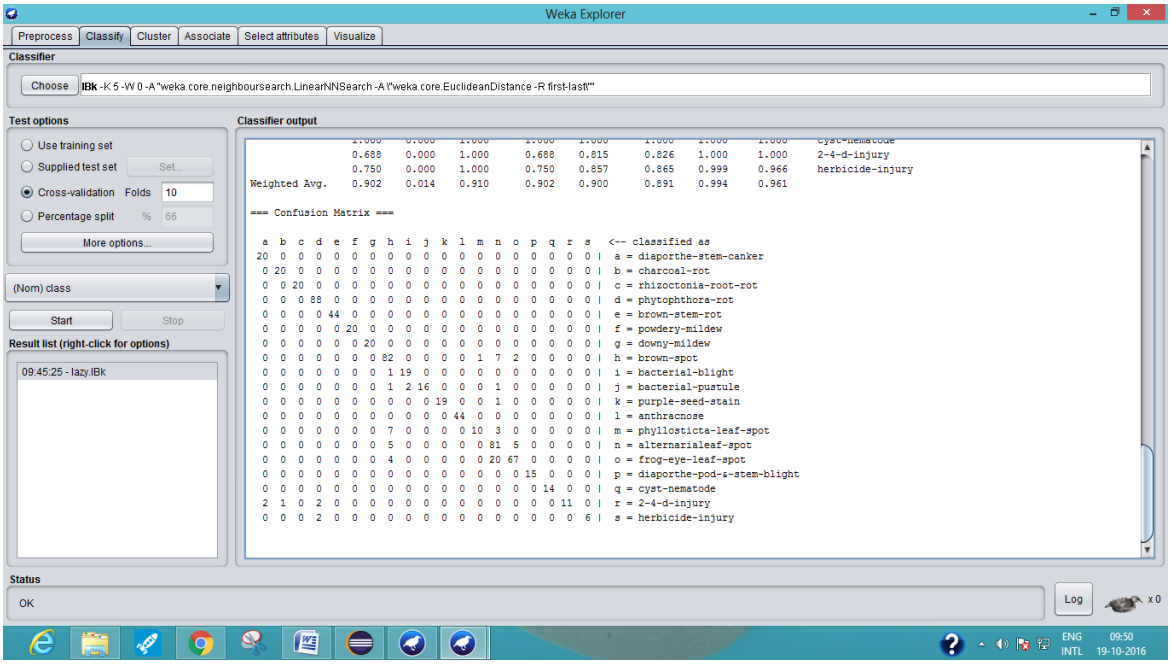


2) Applying KNN; K = 5



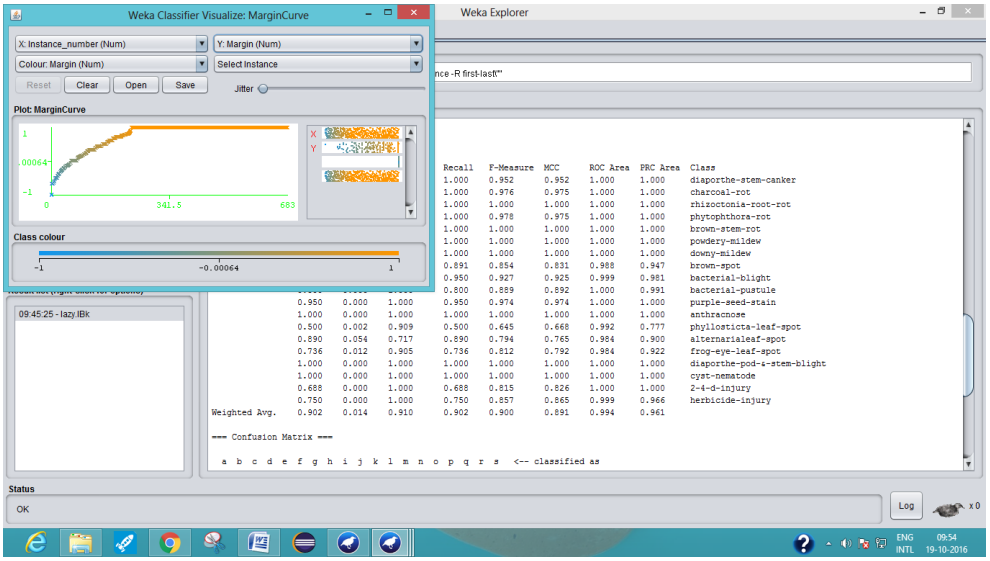
3) KNN Results





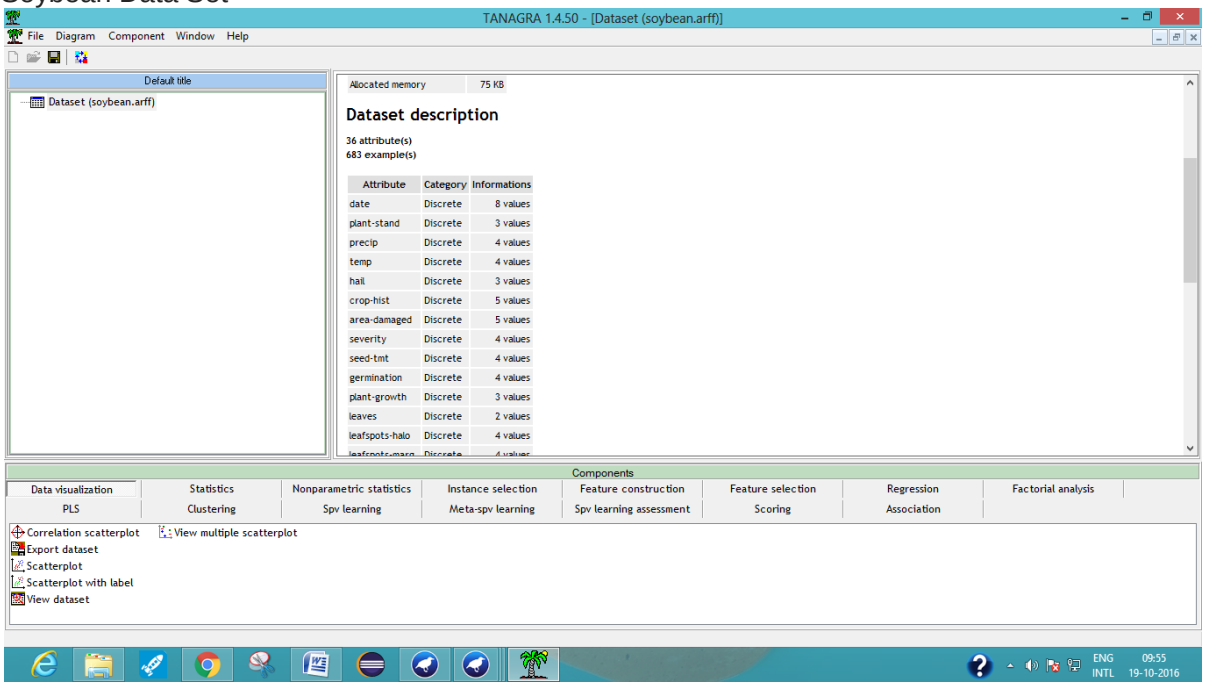
4) Visualise Margin Curve

The margin is defined as the difference between the probability predicted for the actual class and the highest probability predicted for the other classes. One hypothesis as to the good performance of boosting algorithms is that they increase the margins on the training data and this gives better performance on test data.

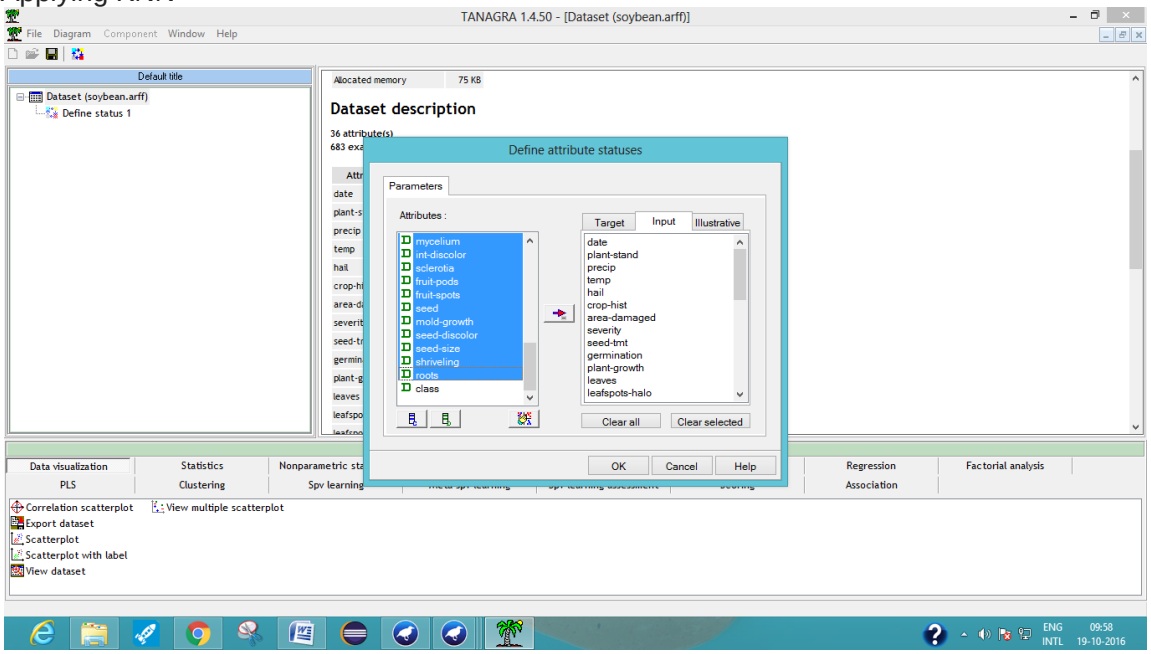


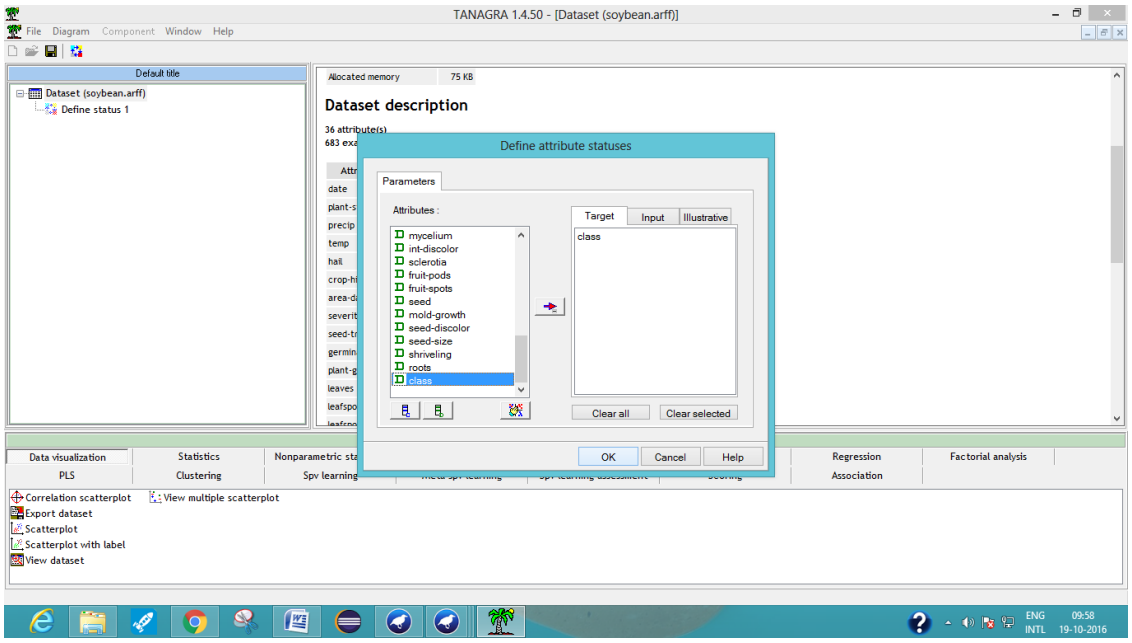
KNN using Tanagra

1) Soybean Data Set

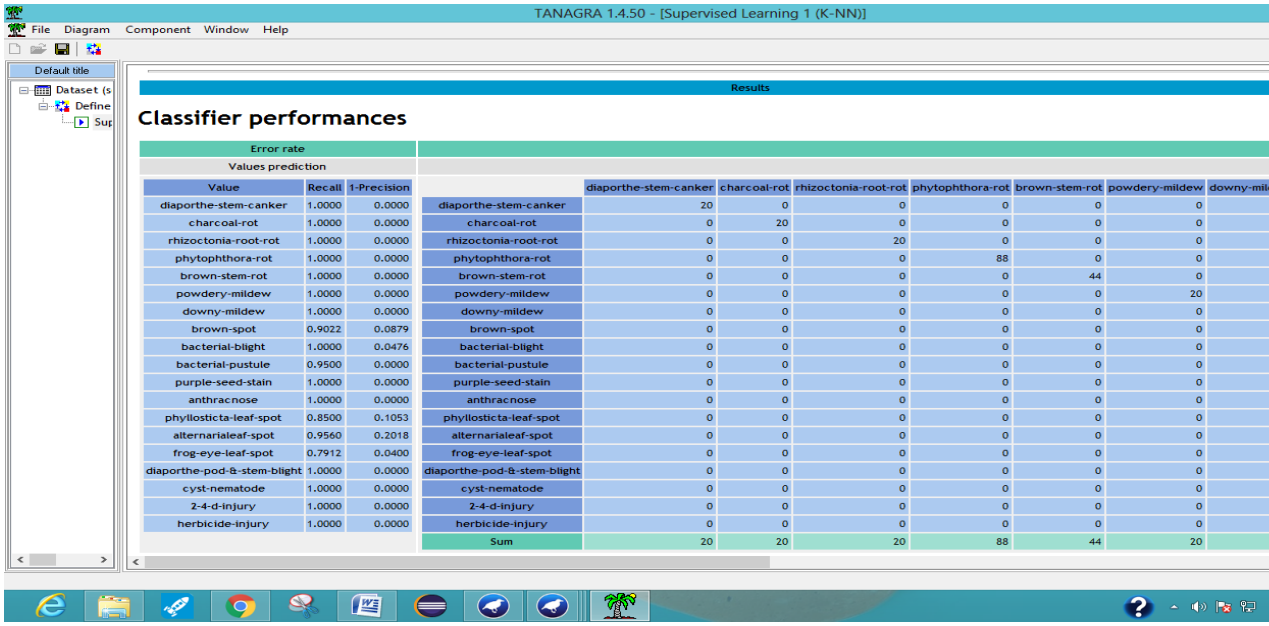


2) Applying KNN





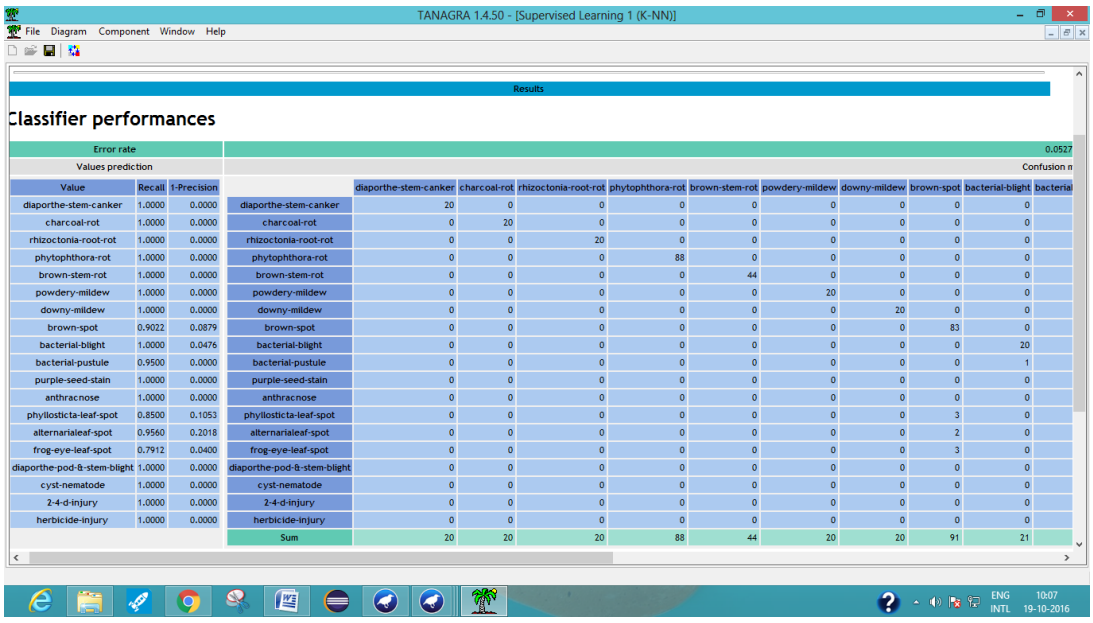
3) KNN Results – HEOM Disance



Value	Recall	1-Precision
diaporthe-stem-canker	1.0000	0.0000
charcoal-rot	1.0000	0.0000
rhizoctonia-root-rot	1.0000	0.0000
phytophthora-rot	1.0000	0.0000
brown-stem-rot	1.0000	0.0000
powdery-mildew	1.0000	0.0000
downy-mildew	1.0000	0.0000
brown-spot	0.9022	0.0879
bacterial-blight	1.0000	0.0476
bacterial-pustule	0.9500	0.0000
purple-seed-stain	1.0000	0.0000
anthracnose	1.0000	0.0000
phyllosticta-leaf-spot	0.8500	0.1053
alternarialeaf-spot	0.9560	0.2018
frog-eye-leaf-spot	0.7912	0.0400
diaporthe-pod-&-stem-blight	1.0000	0.0000
cyst-nematode	1.0000	0.0000
2-4-d-injury	1.0000	0.0000
herbicide-injury	1.0000	0.0000

	diaporthe-stem-canker	charcoal-rot	rhizoctonia-root-rot	phytophthora-rot	brown-stem-rot	powdery-mildew	downy-mil
diaporthe-stem-canker	20	0	0	0	0	0	
charcoal-rot	0	20	0	0	0	0	
rhizoctonia-root-rot	0	0	20	0	0	0	
phytophthora-rot	0	0	0	88	0	0	
brown-stem-rot	0	0	0	0	44	0	
powdery-mildew	0	0	0	0	0	20	
downy-mildew	0	0	0	0	0	0	
brown-spot	0	0	0	0	0	0	
bacterial-blight	0	0	0	0	0	0	
bacterial-pustule	0	0	0	0	0	0	
purple-seed-stain	0	0	0	0	0	0	
anthracnose	0	0	0	0	0	0	
phyllosticta-leaf-spot	0	0	0	0	0	0	
alternarialeaf-spot	0	0	0	0	0	0	
frog-eye-leaf-spot	0	0	0	0	0	0	
diaporthe-pod-&-stem-blight	0	0	0	0	0	0	
cyst-nematode	0	0	0	0	0	0	
2-4-d-injury	0	0	0	0	0	0	
herbicide-injury	0	0	0	0	0	0	
Sum	20	20	20	88	44	20	

4) KNN Results – Euclidean Distance



Value	Recall	1-Precision
diaporthe-stem-canker	1.0000	0.0000
charcoal-rot	1.0000	0.0000
rhizoctonia-root-rot	1.0000	0.0000
phytophthora-rot	1.0000	0.0000
brown-stem-rot	1.0000	0.0000
powdery-mildew	1.0000	0.0000
downy-mildew	1.0000	0.0000
brown-spot	0.9022	0.0879
bacterial-blight	1.0000	0.0476
bacterial-pustule	0.9500	0.0000
purple-seed-stain	1.0000	0.0000
anthracnose	1.0000	0.0000
phyllosticta-leaf-spot	0.8500	0.1053
alternarialeaf-spot	0.9560	0.2018
frog-eye-leaf-spot	0.7912	0.0400
diaporthe-pod-&-stem-blight	1.0000	0.0000
cyst-nematode	1.0000	0.0000
2-4-d-injury	1.0000	0.0000
herbicide-injury	1.0000	0.0000

	diaporthe-stem-canker	charcoal-rot	rhizoctonia-root-rot	phytophthora-rot	brown-stem-rot	powdery-mildew	downy-mildew	brown-spot	bacterial-blight	bacterial
diaporthe-stem-canker	20	0	0	0	0	0	0	0	0	
charcoal-rot	0	20	0	0	0	0	0	0	0	
rhizoctonia-root-rot	0	0	20	0	0	0	0	0	0	
phytophthora-rot	0	0	0	88	0	0	0	0	0	
brown-stem-rot	0	0	0	0	44	0	0	0	0	
powdery-mildew	0	0	0	0	0	20	0	0	0	
downy-mildew	0	0	0	0	0	0	20	0	0	
brown-spot	0	0	0	0	0	0	0	83	0	
bacterial-blight	0	0	0	0	0	0	0	0	20	
bacterial-pustule	0	0	0	0	0	0	0	0	1	
purple-seed-stain	0	0	0	0	0	0	0	0	0	
anthracnose	0	0	0	0	0	0	0	0	0	
phyllosticta-leaf-spot	0	0	0	0	0	0	0	3	0	
alternarialeaf-spot	0	0	0	0	0	0	0	2	0	
frog-eye-leaf-spot	0	0	0	0	0	0	0	3	0	
diaporthe-pod-&-stem-blight	0	0	0	0	0	0	0	0	0	
cyst-nematode	0	0	0	0	0	0	0	0	0	
2-4-d-injury	0	0	0	0	0	0	0	0	0	
herbicide-injury	0	0	0	0	0	0	0	0	0	
Sum	20	20	20	88	44	20	20	91	21	

COMPARISON BETWEEN WEKA AND TANAGRA

Time taken – 0, 0
Error – 0.0157, lesser
K = 5, 5

EXPERIMENT -6

Apriori Algorithm

Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by Apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket analysis.

Transactions	Items bought
T1	Item1, item2, item3
T2	Item1, item2
T3	Item2, item5
T4	Item1, item2, item5

Learning association rules basically means finding the items that are purchased together more frequently than others.
For example in the above table you can see Item1 and item2 are bought together frequently.

Original table:

Transaction ID	Items Bought
T1	{M, O, N, K, E, Y }
T2	{D, O, N, K, E, Y }
T3	{M, A, K, E}
T4	{M, U, C, K, Y }
T5	{C, O, O, K, I, E}

M – Mango

O – Oranges

N – Nintendo

K – Key-chain

E – Eggs

Y – Yo-yo

Step 1: Count the number of transactions in which each item occurs, Note 'O=Onion' is bought 4 times in total, but, it occurs in just 3 transactions.

Item	No of transactions
M	3
O	3

N	2
K	5
E	4
Y	3
D	1
A	1
U	1
C	2
I	1

Step 2: Now remember we said the item is said frequently bought if it is bought at least 3 times. So in this step we remove all the items that are bought less than 3 times from the above table and we are left with

Item	Number of transactions
M	3
O	3
K	5
E	4
Y	3

This is the single items that are bought frequently. Now let's say we want to find a pair of items that are bought frequently. We continue from the above table (Table in step 2)

Step 3: We start making pairs from the first item, like MO,MK,ME,MY and then we start with the second item like OK,OE,OY. We did not do OM because we already did MO when we were making pairs with M and buying a Mango and Onion together is same as buying Onion and Mango together. After making all the pairs we get,

Item pairs
MO
MK
ME
MY
OK
OE
OY
KE
KY
EY

Step 4: Now we count how many times each pair is bought together. For example M and O is just bought together in {M,O,N,K,E,Y}
While M and K is bought together 3 times in {M,O,N,K,E,Y}, {M,A,K,E} AND {M,U,C, K, Y}
After doing that for all the pairs we get

Item Pairs	Number of transactions
MO	1
MK	3
ME	2
MY	2
OK	3
OE	3
OY	2
KE	4
KY	3
EY	2

Step 5: Golden rule to the rescue. Remove all the item pairs with number of transactions less than three and we are left with

Item Pairs	Number of transactions
MK	3
OK	3
OE	3
KE	4
KY	3

These are the pairs of items frequently bought together.
Now let's say we want to find a set of three items that are brought together.
We use the above table (table in step 5) and make a set of 3 items.

Step 6: To make the set of three items we need one more rule (it's termed as self-join),
It simply means, from the Item pairs in the above table, we find two pairs with the same first Alphabet, so we get

- OK and OE, this gives OKE
- KE and KY, this gives KEY

Then we find how many times O,K,E are bought together in the original table and same for K,E,Y and we get the following table

Item Set	Number of transactions
OKE	3
KEY	2

While we are on this, suppose you have sets of 3 items say ABC, ABD, ACD, ACE, BCD and you want to generate item sets of 4 items you look for two sets having the same first two alphabets.

- ABC and ABD -> ABCD
- ACD and ACE -> ACDE

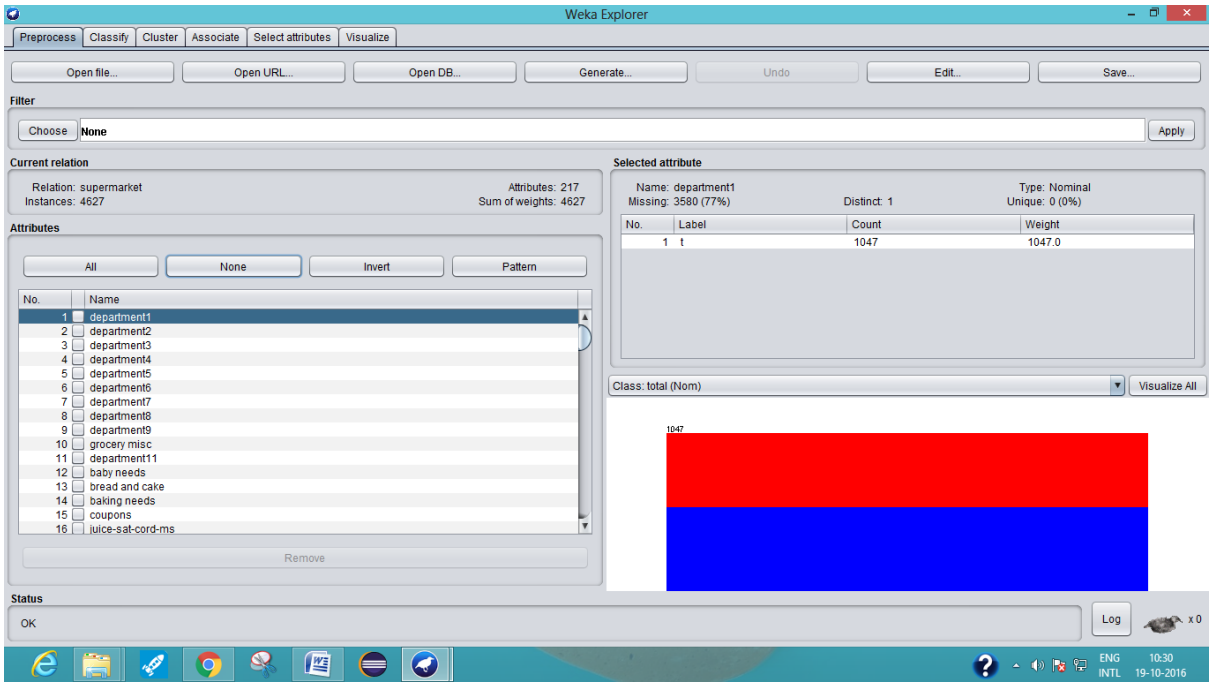
And so on … In general you have to look for sets having just the last alphabet/item different.

Step 7: So we again apply the golden rule, that is, the item set must be bought together at least 3 times which leaves us with just OKE, Since KEY are bought together just two times.

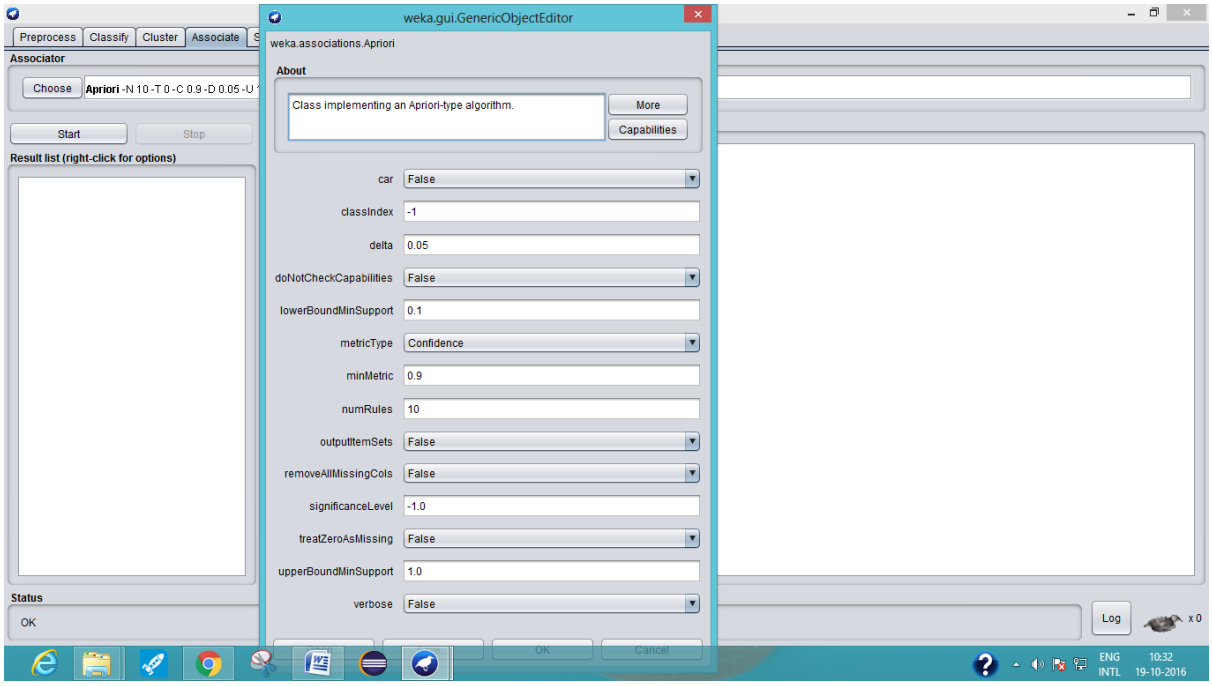
Thus the set of three items that are bought together most frequently are O,K,E.

Apriori using Weka

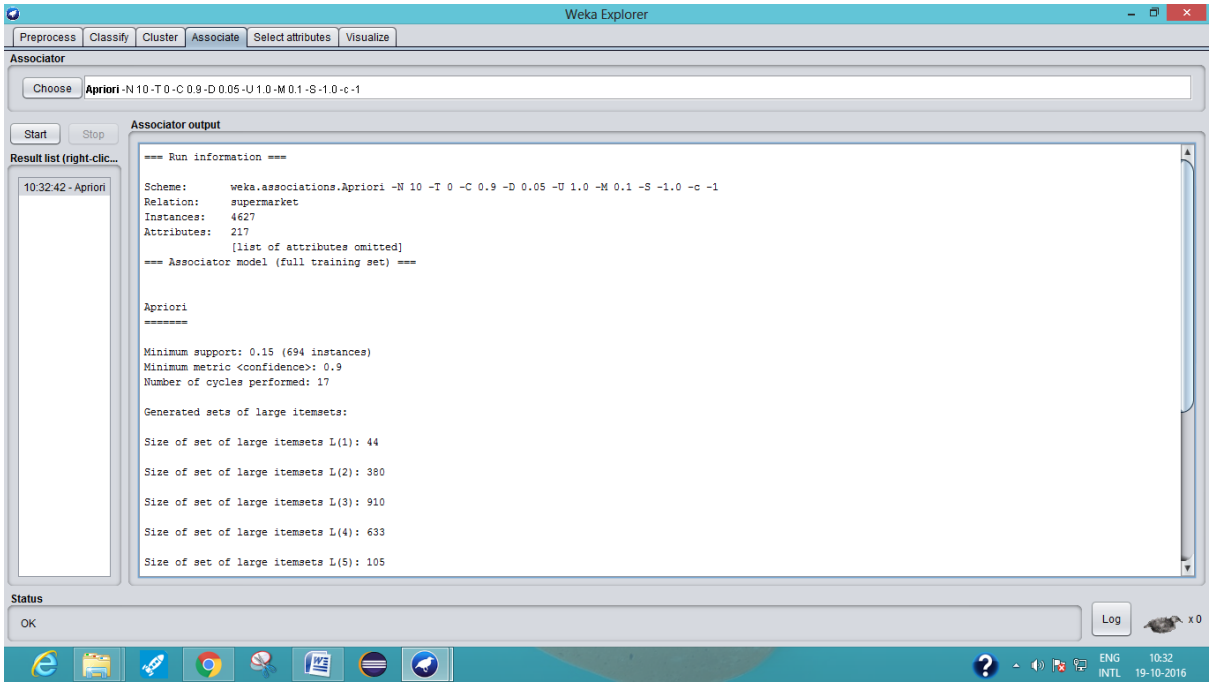
1) Supermarket Data Set



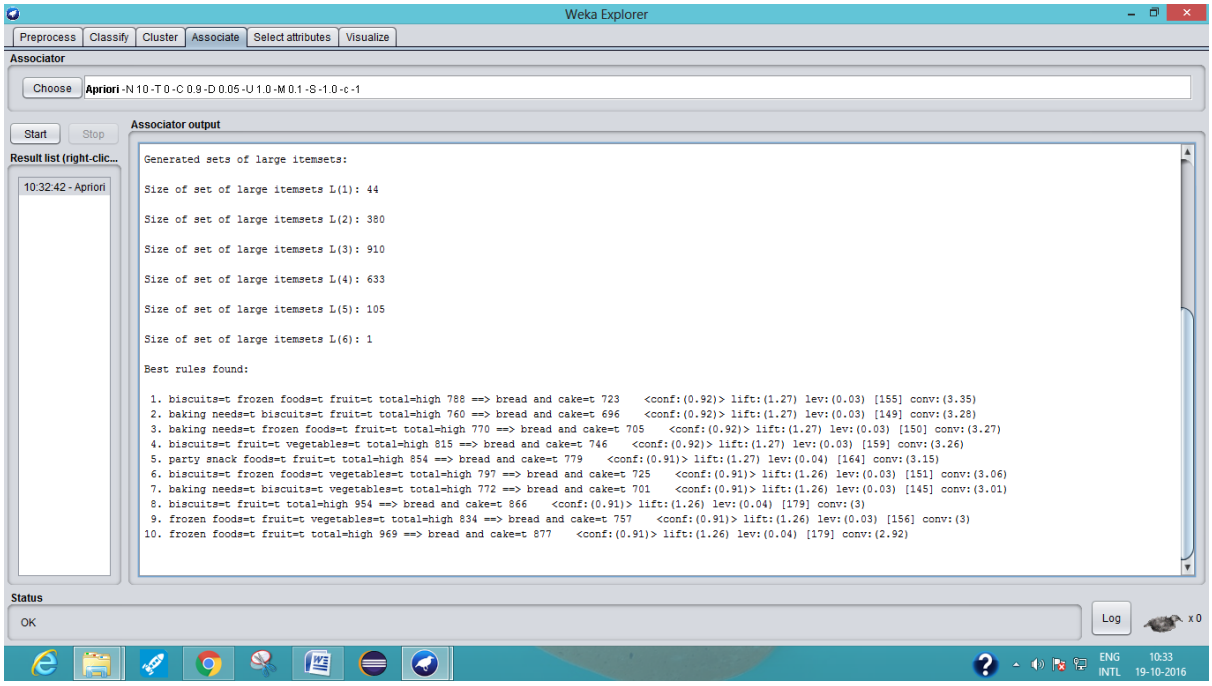
2) Applying Apriori



3) Apriori Results

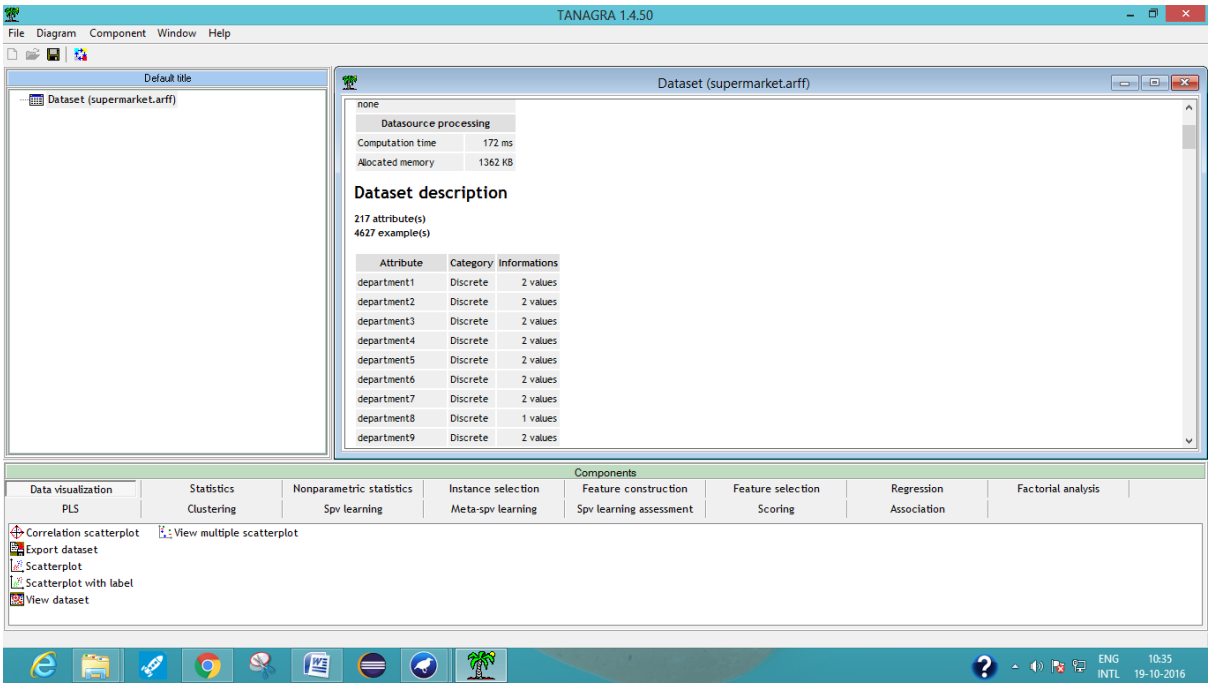


4) Items Brought together most frequently

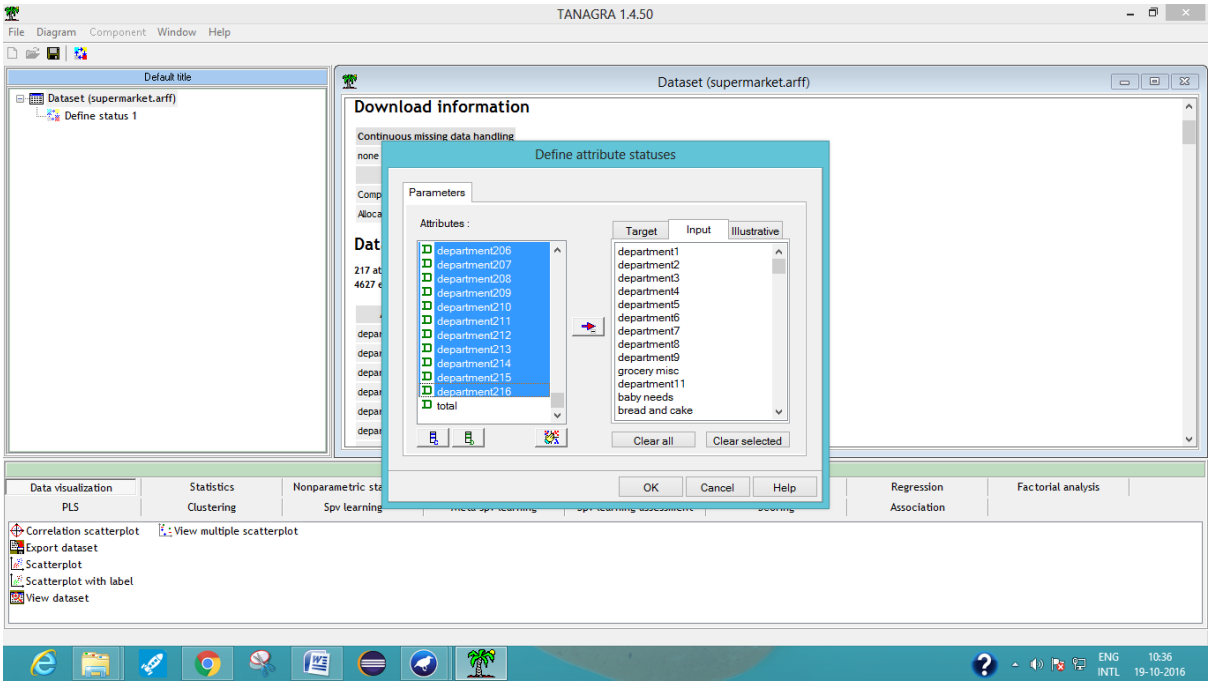


Apriori using Tanagra

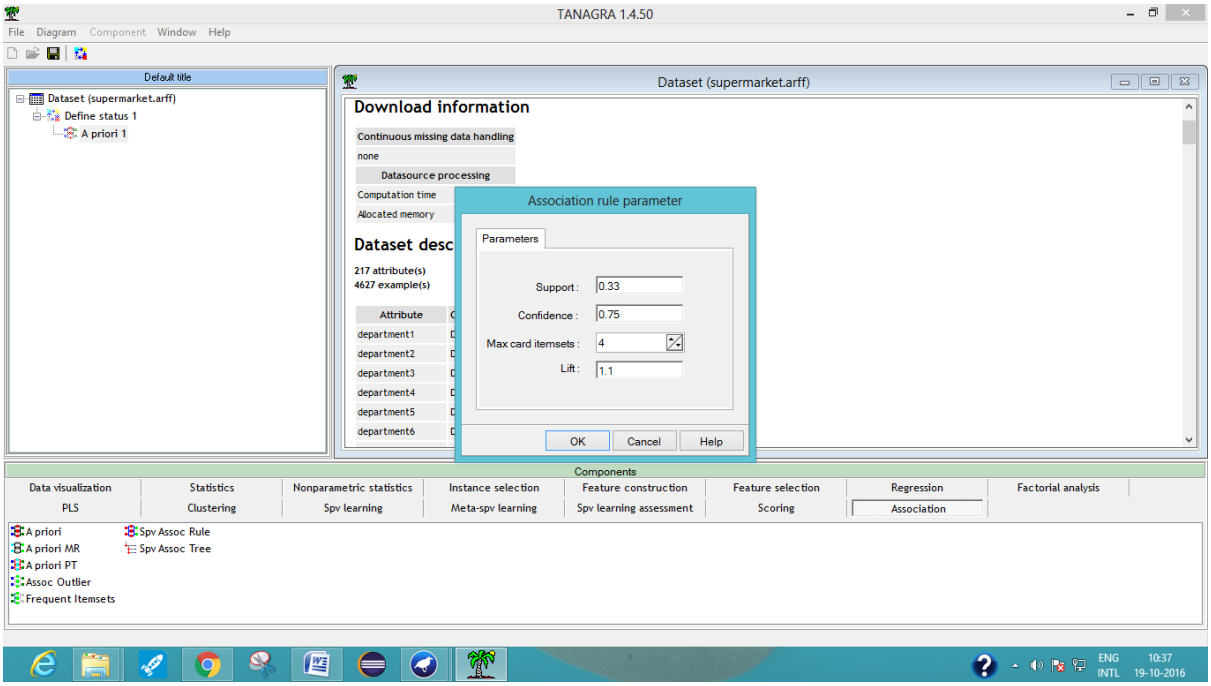
1) Supermarket Data Set



2) Applying Apriori


TANAGRA 1.4.50
File Diagram Component Window Help
Default title
Dataset (supermarket.arff)
Define status 1
Dataset (supermarket.arff)
Download information
Continuous missing data handling
none
Define attribute statuses
Parameters
Attributes :
Target
Input
Illustrative
department206
department207
department208
department209
department210
department211
department212
department213
department214
department215
department216
total
department1
department2
department3
department4
department5
department6
department7
department8
department9
grocery misc
department11
baby needs
bread and cake
Clear all
Clear selected
OK
Cancel
Help
Data visualization
Statistics
Nonparametric sta
Regression
Factorial analysis
PLS
Clustering
Spv learning
Association
Correlation scatterplot
View multiple scatterplot
Export dataset
Scatterplot
Scatterplot with label
View dataset
ENG
INTL
10:36
19-10-2016

3) Apriori


TANAGRA 1.4.50
File Diagram Component Window Help
Default title
Dataset (supermarket.arff)
Define status 1
A priori 1
Dataset (supermarket.arff)
Download information
Continuous missing data handling
none
Datasource processing
Computation time
Allocated memory
Association rule parameter
Parameters
Support :
0.33
Confidence :
0.75
Max card itemsets :
4
Lift :
1.1
OK
Cancel
Help
Dataset desc
217 attribute(s)
4627 example(s)
Attribute
department1
department2
department3
department4
department5
department6
Components
Data visualization
Statistics
Nonparametric statistics
Instance selection
Feature construction
Feature selection
Regression
Factorial analysis
PLS
Clustering
Spv learning
Meta-spv learning
Spv learning assessment
Scoring
Association
A priori
A priori MR
A priori PT
Assoc Outlier
Frequent Itemsets
Spv Assoc Rule
Spv Assoc Tree
ENG
INTL
10:37
19-10-2016

EXPERIMENT -7

Implementing ID3 using WEKA AND TANAGRA (J48 and C4.5)

Decision Tree induction is the learning of decision trees from class-labeled training tuples(instances). A decision tree is a flowchart-like tree structure, where each internal node (nonleaf node) denotes a test on attribute, each branch represents an outcome of the test, and leaf node (or terminal) holds a class label. The topmost node in a tree is the root node.

ID3 algorithm

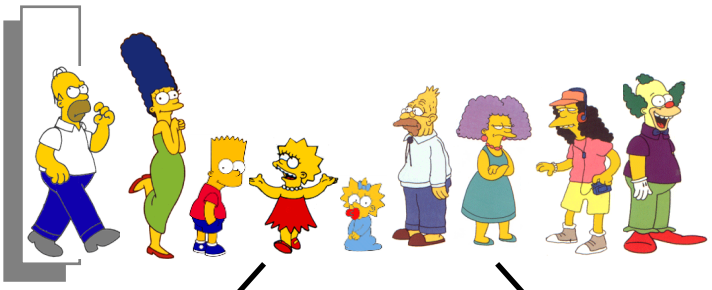
ID3 (Examples, Target_Attribute, Attributes)
Examples are the training examples. Target_Attribute is the attribute whose v
predicted by the tree. Attributes is the list of attributes which may be tested by
decision tree. Returns a decision tree that correctly classifies the given Examp

- Create a root node for the tree
- IF all examples are positive, Return the single-node tree Root, with lab
- If all examples are negative, Return the single-node tree Root, with lab
- If number of predicting attributes is empty, then Return the single nod
  with label = most common value of the target attribute in the examples
- Otherwise Begin
  - A ← The Attribute that best classifies examples
  - Decision Tree attribute for Root ← A
  - For each positive value, vi, of A,
    - Add a new tree branch below Root, corresponding to th
      vi
    - Let Examples(vi), be the subset of examples that have t
      for A
    - If Examples(vi) is empty
      - Then below this new branch add a leaf node wit
        most common target value in the examples
      - Else below this new branch add the subtree
        ID3 (Examples(vi), Target_Attribute, Attributes
- End
- Return Root

Example – The Simpsons

Person	Hair Length	Weight	Age	Cla
 Homer	0″	250	36	M
 Marge	10″	150	34	F
 Bart	2″	90	10	M
 Lisa	6″	78	8	F
 Maggie	4″	20	1	F
 Abe	1″	170	70	M
 Selma	8″	160	41	F
 Otto	10″	180	38	M
 Krusty	6″	200	45	M

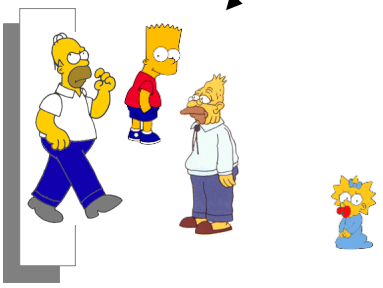
 Comic	8″	290	38	?



$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right)$$

yes

Hair Length <= 5?



$Entropy($
$(3/5)\log_2($
$(2/5)\log_2($

no

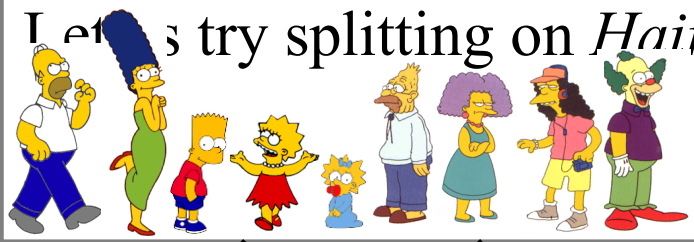
$Entropy(4\mathbf{F},5\mathbf{M}) = -(4/9)\log_2(4/9) -$
$= \mathbf{0.9911}$

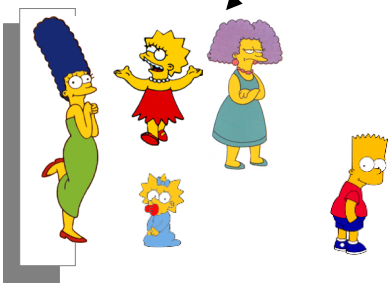
$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

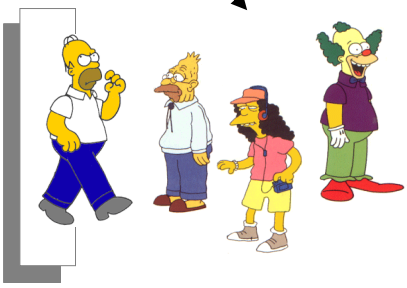
$Entropy(1\mathbf{F},3\mathbf{M}) = -(1/4)\log_2(1/4) - (3/4)\log_2(3/4)$
$= \mathbf{0.8113}$

Let's try splitting on *Hair length*

$$Entropy(S) = -\frac{p}{p+n}\log_2\left(\frac{p}{p+n}\right)$$







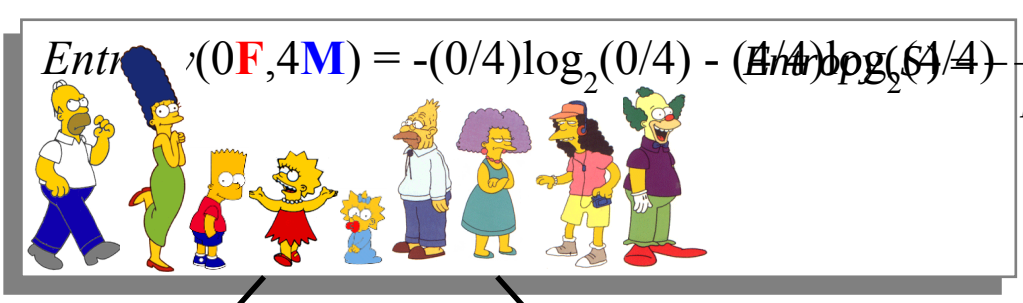
Entropy(4
(4/9)log$_2$(
(5/9)log$_2$(

Weight <= 160?

yes

no

Gain(Hair Length <= 5) = 0.9911 − (4/9 * 0.8113 + 5/9 * 0.9710) = **0.0911**

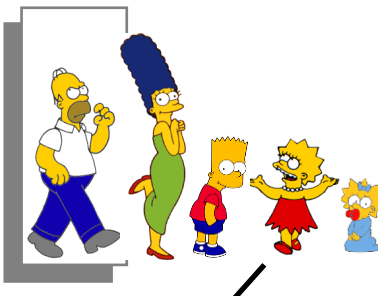


Entropy(4**F**,1**M**) = -(4/5)log2(4/5) - (1/5)log2(1/5)

$$Gain(A) = E(Current\ set) - \sum E(all\ child\ sets)$$

age <= 40?

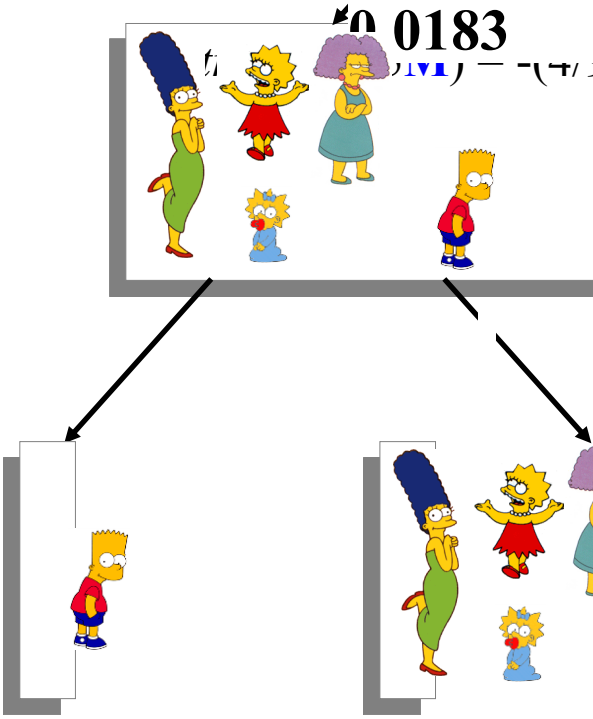
Weight <= 160?



Gain(Age

0.0183

yes



no

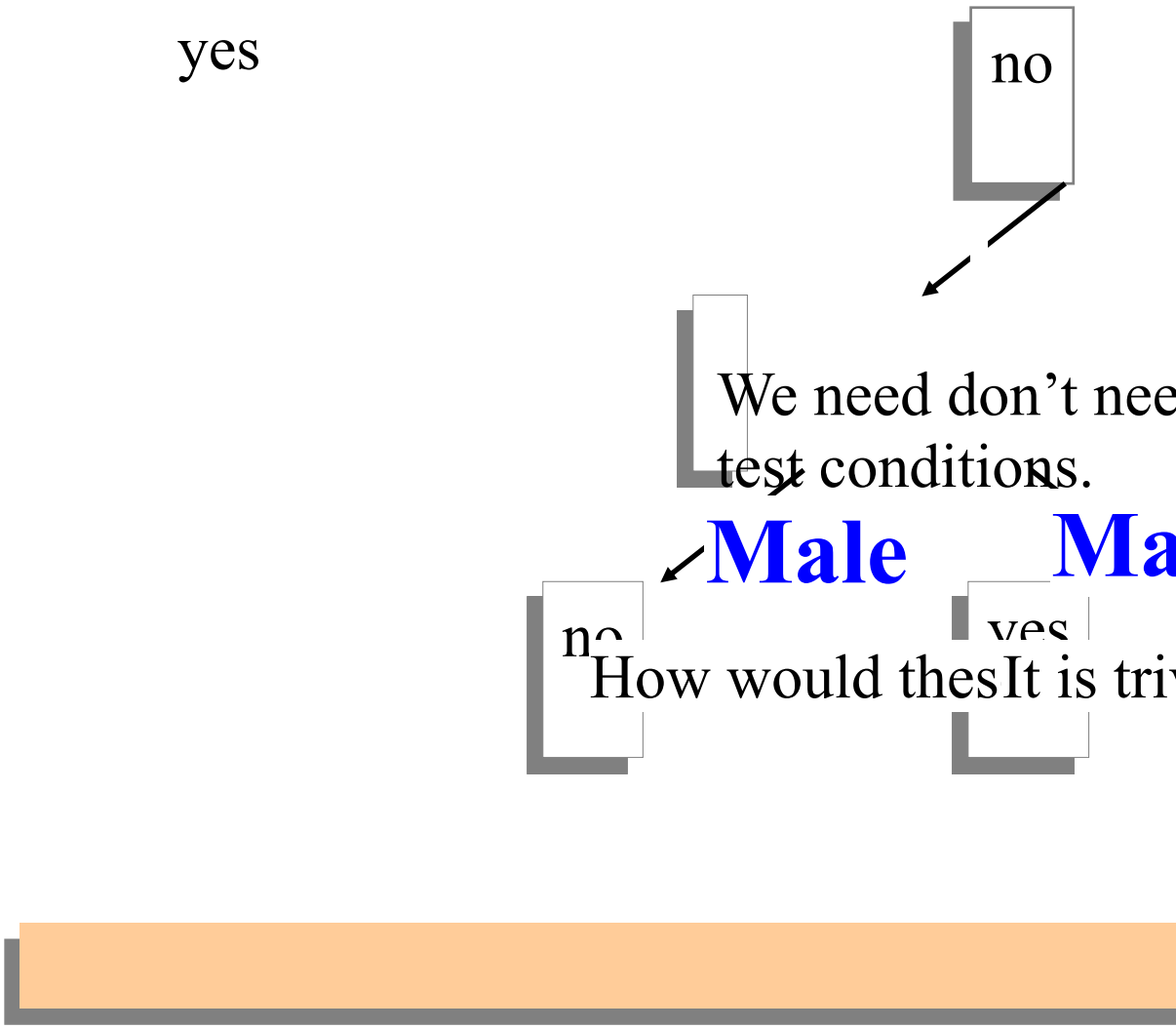
Weight <= 160?

Of the 3 features w
people who weigh
(as males), the und

This time we fi d th
we are done!

n)

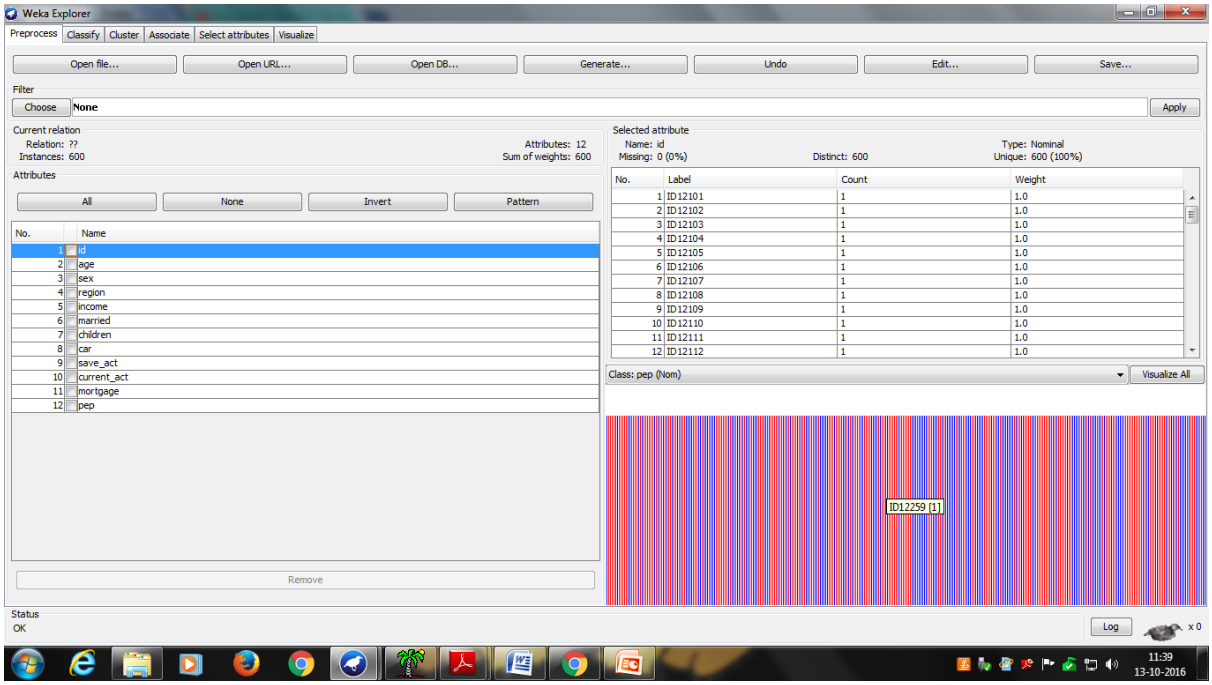
yes



IMPLEMENTING J48 USING WEKA

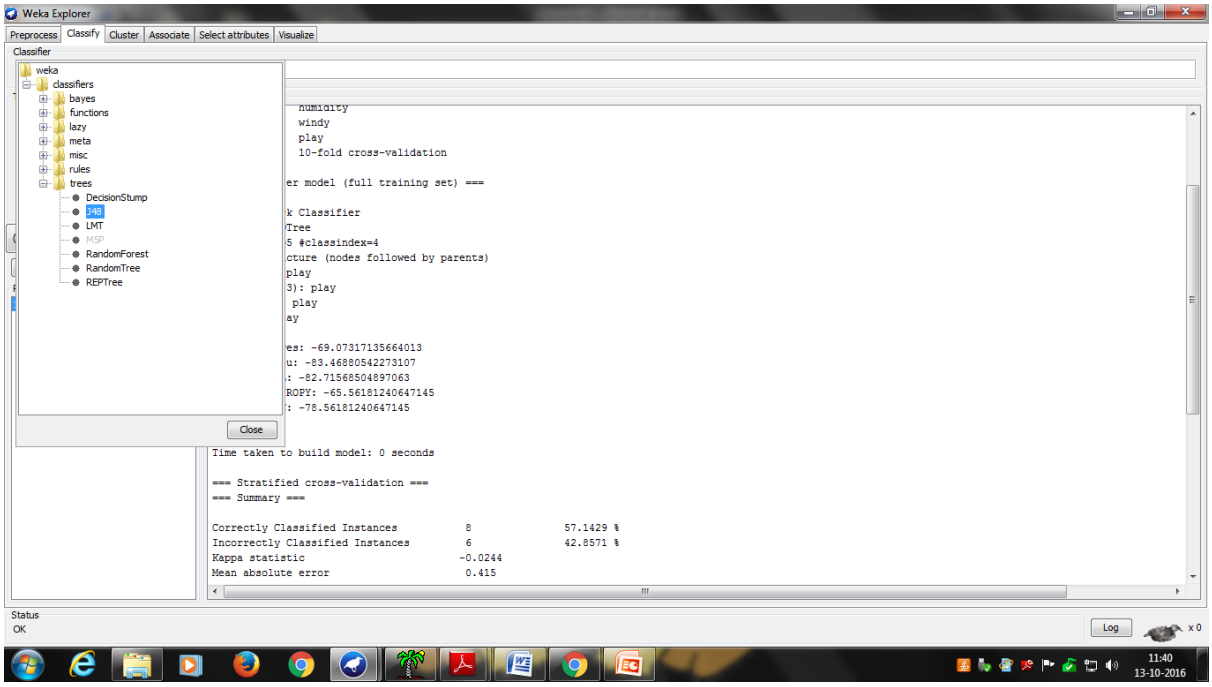
Decision tree J48 is the implementation of algorithm ID3 (Iterative Dichotomiser 3) developed by the WEKA project team.

1) Bank Data Set

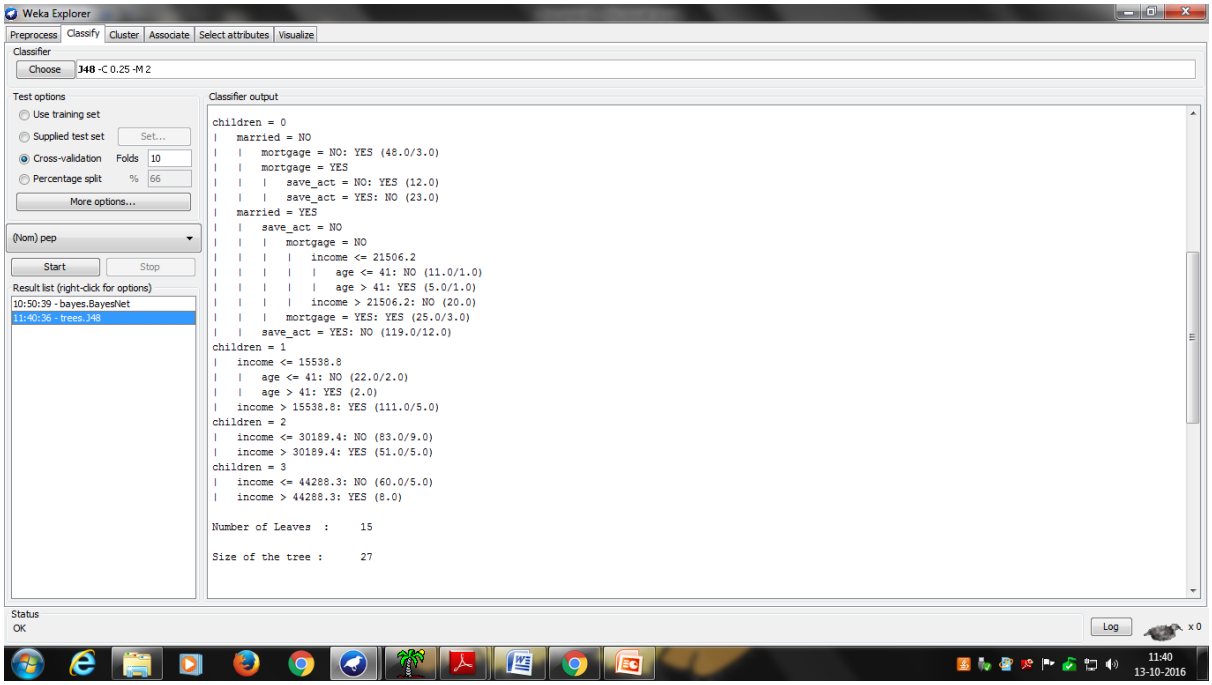


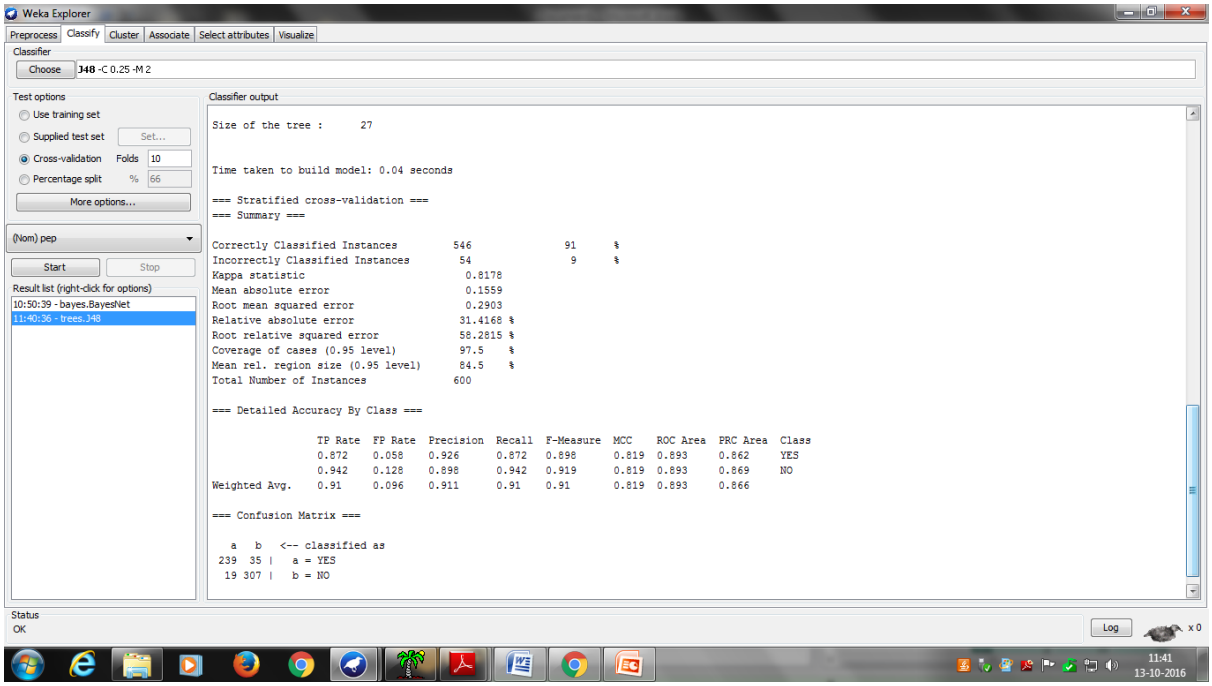
2) Implementing J48

3)

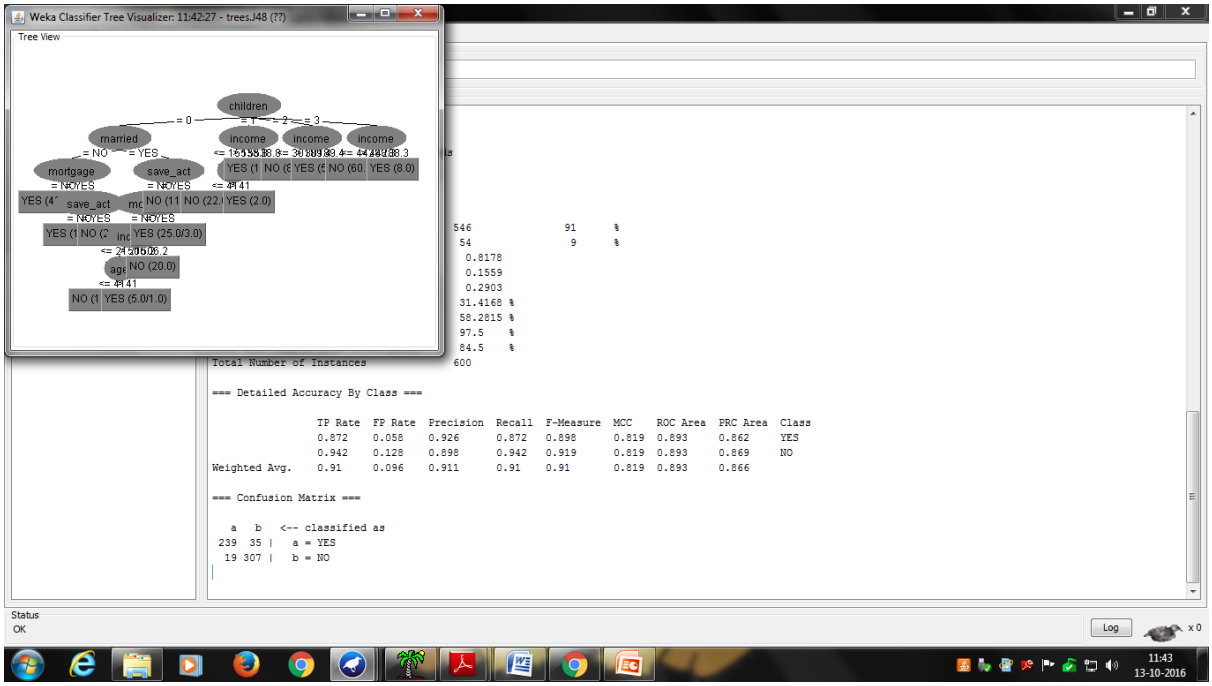


4) J48 Results



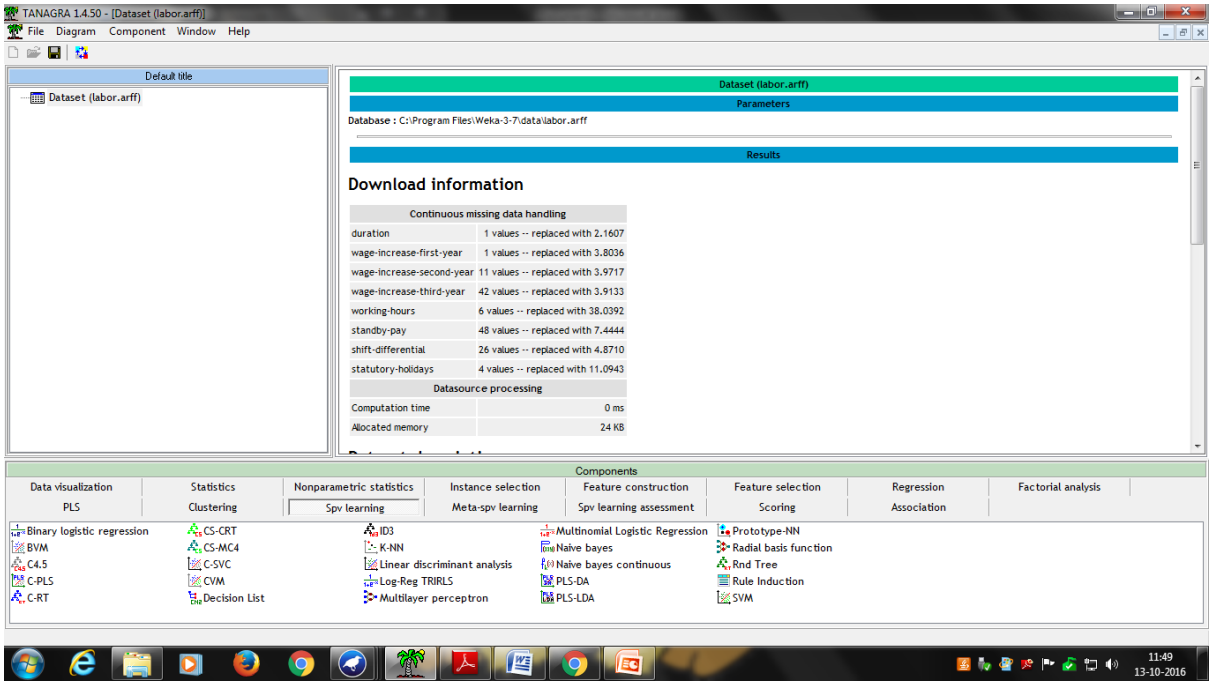


5) Visualise Results

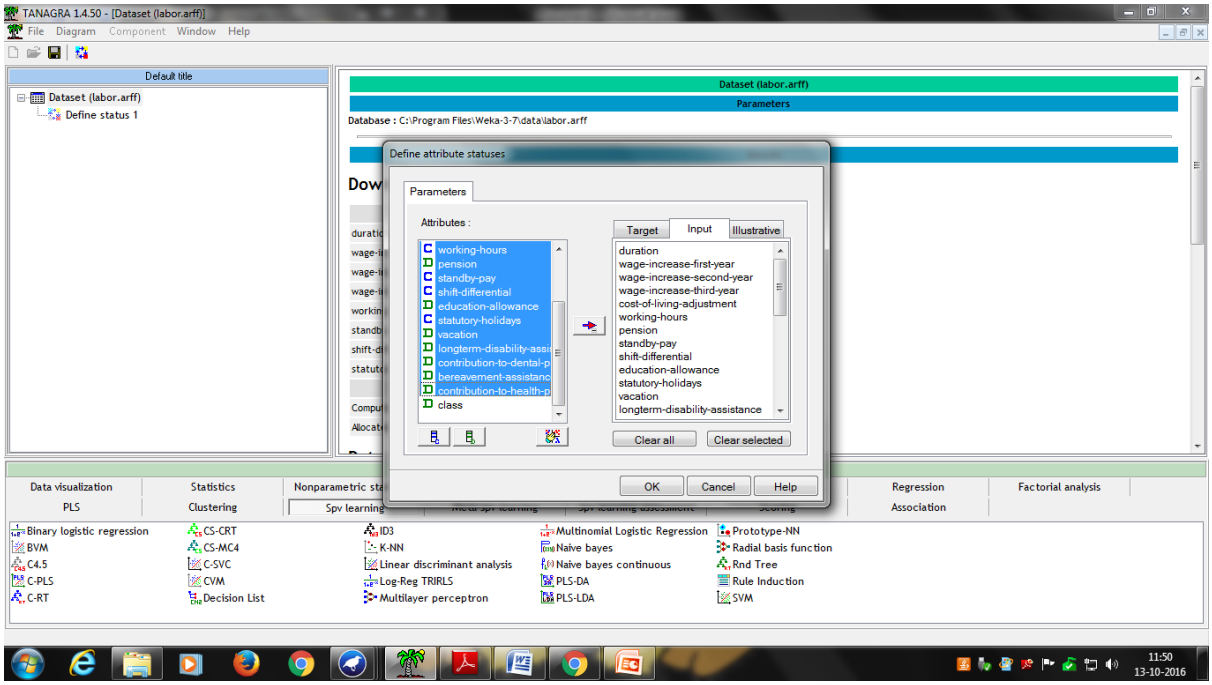


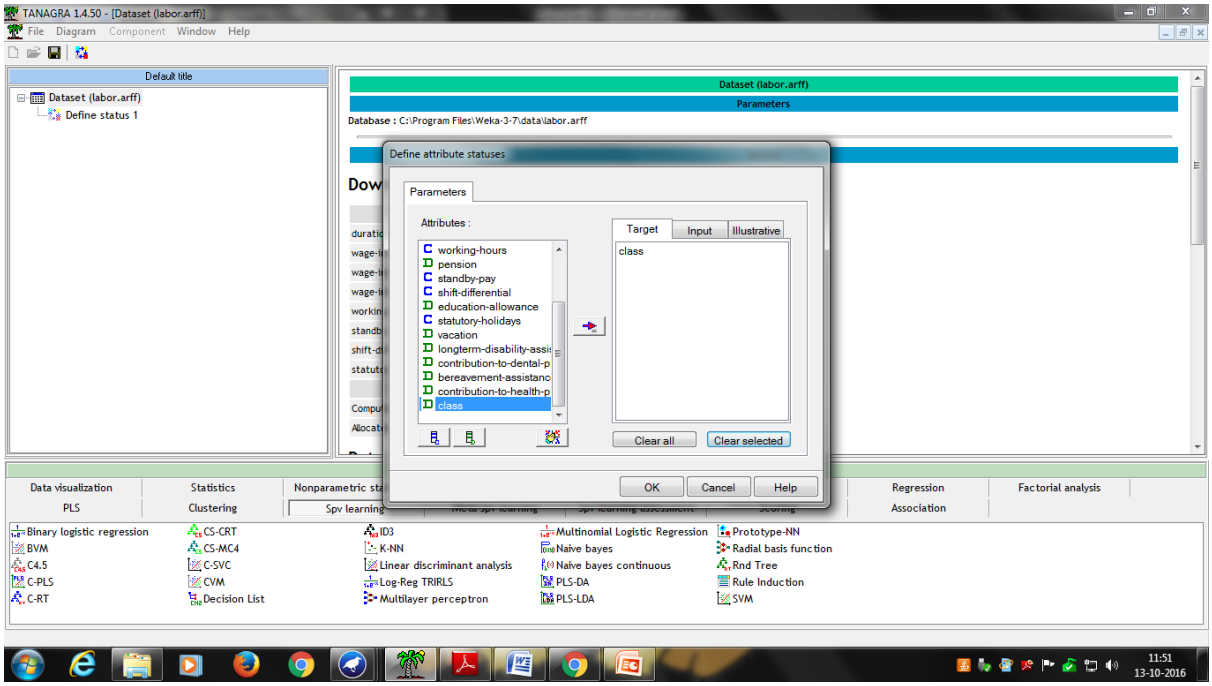
IMPLEMENT C4.5 USING TANAGRA

1) Labour Data Set

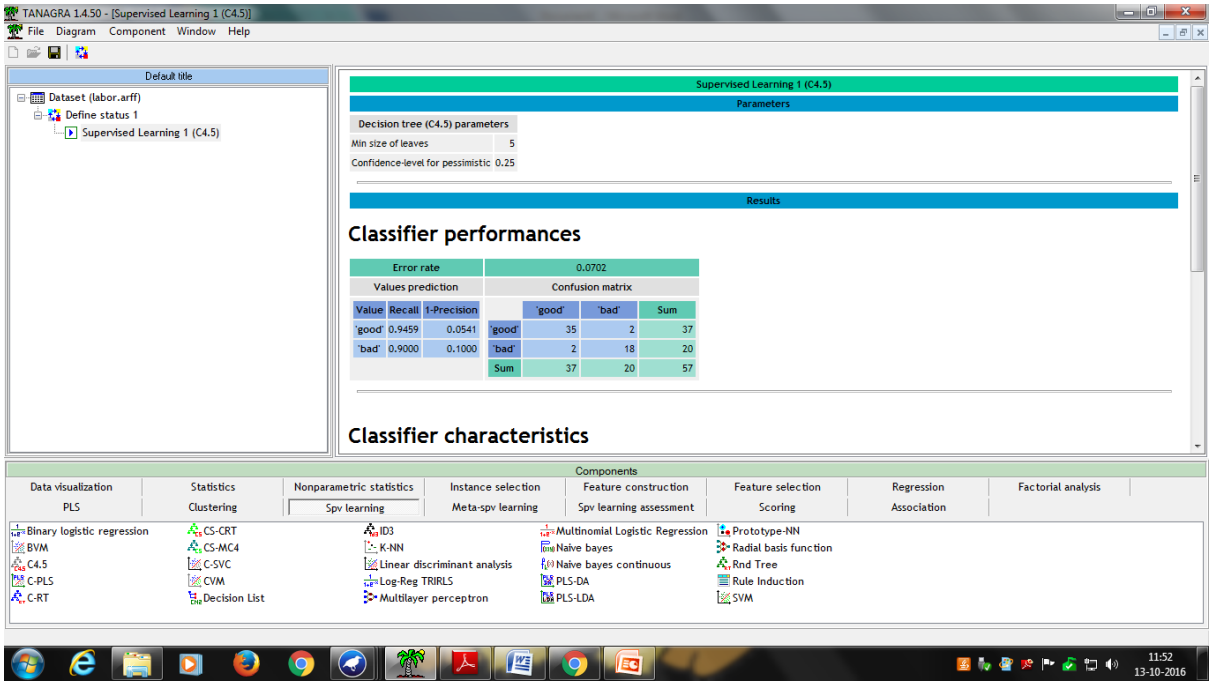


2) Implementing C4.5

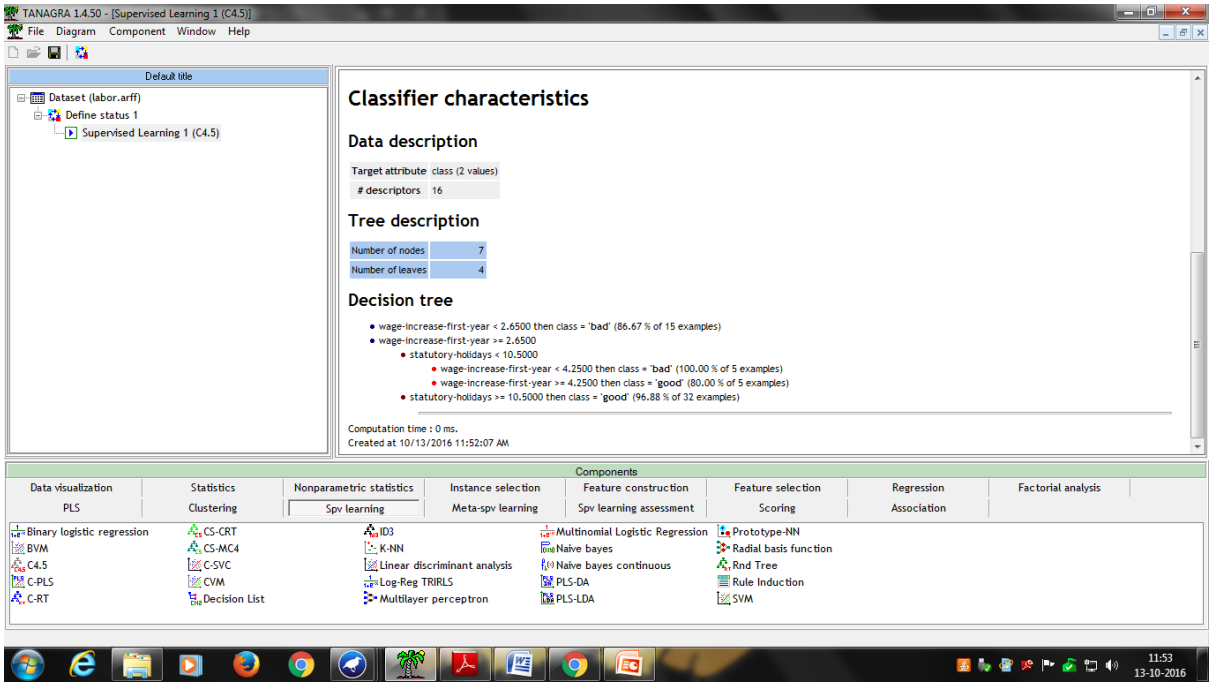




3) Tanagra C4.5 Results



4) Visualise Data



Comparison

After analyzing the results of both the tools, it is found that both are able to generate tree model in very less time. Both the tools are very efficient in generating decision trees. However, in terms of classifiers` applicability, the Weka tool is better in terms of the ability to run the classifier. However, the performance of classifier is better in Tanagra than Weka in terms of error rate. Also, Tanagra is faster than Weka in tree generation as its internal structure is organized in columns in memory. In addition, Weka tool has attained the highest performance in terms of accuracy when used with "Use Training Set" test mode than "Cross Validation" test mode followed by "Percentage Split" test mode. Through this comparative study, we conclude that Tanagra is better tool than Weka. Also, we found that c4.5 algorithm works well in decision tree induction.