DEV SANSKRITI
VISHWAVIDYALAYA

DEV SANSKRITI VISHWAVIDYALAYA

SESSION 2018-21

Practicle File
Of
Data Mining and Warehousing

Submitted To:
Mr. Naveen Pandey
Assisstant Professor

Submitted By:
Aniket Kumar
BCA (6th Sem.)

Department of Computer Science,
DSVV, Haridwar

INDEX

1. Draw and explain DATA WAREHOUSE ARCHITECTURE

A data warehouse is a repository that includes past and commutative information from one or multiple sources. This repository can be used by the employees of the organization for analysis, drawing insights, and future forecasting.
The fundamental concept of a data warehouse is the extract, transform and load (ETL) process:

- Extract: Gathering data from various heterogeneous sources
- Transform: Converting sub-standard data into clean, structured, and verified data that is ready to use
- Load: Loading the data into a new destination

Data Warehouse Architecture

Data warehouse architecture is a data storage framework's design of an organization. A data warehouse architecture takes information from raw sets of data and stores it in a structured and easily digestible format.

When designing a corporation's data warehouse, there are three main types of data warehouse architecture to consider:

Single-tier data warehouse architecture

The structure of a single-tier data warehouse architecture centers on producing a dense set of data and reducing the volume of data deposited. Although it is beneficial for eliminating redundancies, this architecture is not suitable for businesses with complex data requirements and numerous data streams. This is where the 2-tier and 3-tier data warehouse architecture come in as they both deal with more complex data streams.

Two-tier data warehouse architecture

In comparison, the data structure of a two-tier architecture splits the tangible data sources from the warehouse itself. Unlike a single-tier, the two-tier structure uses a system and a database server. This is most commonly used in small organizations where a server is used as a data mart. Although it is more efficient at data storage and organization, the two-tier architecture is not scalable. Moreover, it only supports a nominal number of users.

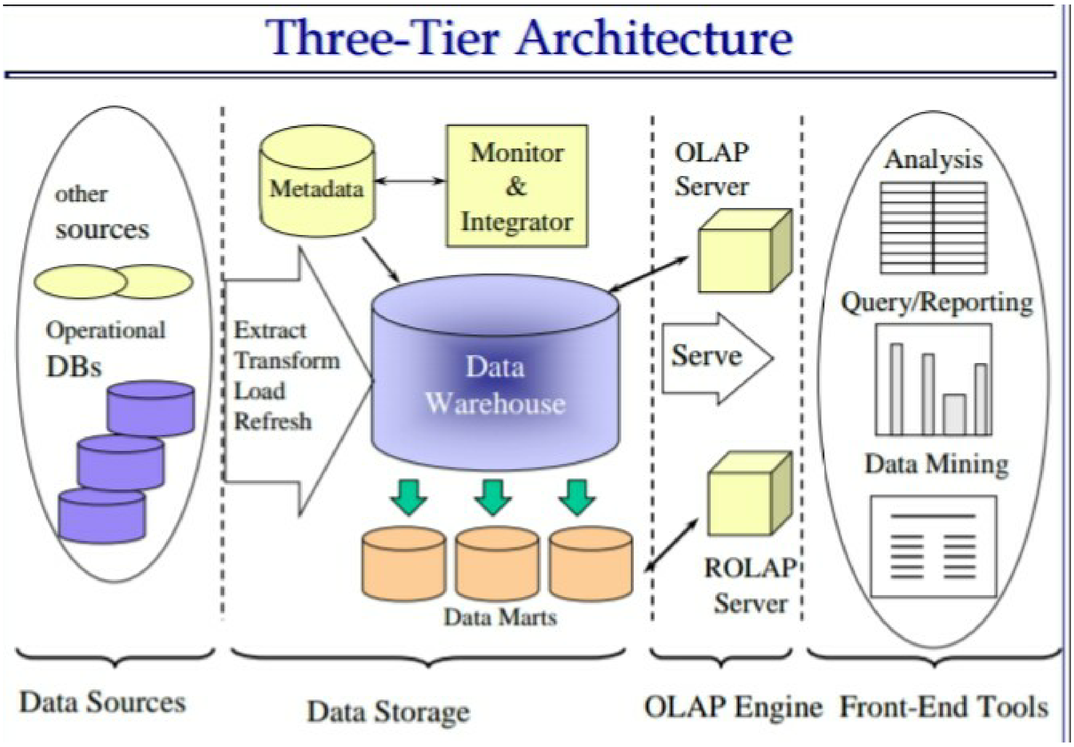
Three-tier data warehouse architecture

The three-tier data warehouse architecture is the most common type of modern DWH architecture as it produces a well-organized data flow from raw information to valuable insights.

The bottom tier typically comprises the databank server that creates an abstraction layer on data from numerous sources, like transactional databanks utilized for front-end uses.

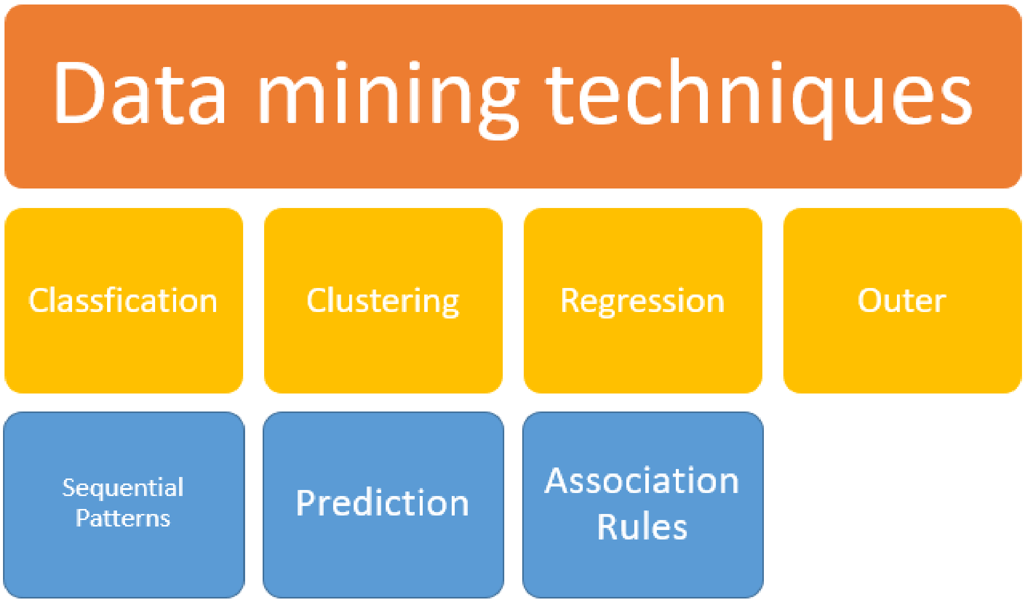
The middle tier includes an Online Analytical Processing (OLAP) server. From a user's perspective, this level alters the data into an arrangement that is more suitable for analysis and multifaceted probing. Since it includes an OLAP server pre-built in the architecture, we can also call it the OLAP-focused data warehouse.

The third and the topmost tier is the client level which includes the tools and Application Programming Interface (API) used for high-level data analysis, inquiring, and reporting.



2. Enlist and explain Data mining techniques

Following are some data mining techniques, which makes data mining very effective:



1. Tracking/Sequential patterns. One of the most basic techniques in data mining is learning to recognize patterns in your data sets. This is usually a recognition of some aberration in your data happening at regular intervals, or an ebb and flow of a certain variable over time. For example, you might see that your sales of a certain product seem to spike just before the holidays, or notice that warmer weather drives more people to your website.

2. Classification. Classification is a more complex data mining technique that forces you to collect various attributes together into discernable categories, which you can then use to draw further conclusions, or serve some function. For example, if you're evaluating data on individual customers' financial backgrounds and purchase histories, you might be able to classify them as "low," "medium," or "high" credit risks. You could then use these classifications to learn even more about those customers.

3. Association. Association is related to tracking patterns, but is more specific to dependently linked variables. In this case, you'll look for specific events or attributes that are highly correlated with another event or attribute; for example, you might notice that when your customers buy a specific item, they also often buy a second, related item. This is usually what's used to populate "people also bought" sections of online stores.

4. Outlier detection. In many cases, simply recognizing the overarching pattern can't give you a clear understanding of your data set. You also need to be able to identify anomalies, or outliers in your data. For example, if your purchasers are almost exclusively male, but during one strange week in July, there’s a huge spike in female purchasers, you’ll want to investigate the spike and see what drove it, so you can either replicate it or better understand your audience in the process.

5. Clustering. Clustering is very similar to classification, but involves grouping chunks of data together based on their similarities. For example, you might choose to cluster different demographics of your audience into different packets based on how much disposable income they have, or how often they tend to shop at your store.

6. Regression. Regression, used primarily as a form of planning and modeling, is used to identify the likelihood of a certain variable, given the presence of other variables. For example, you could use it to project a certain price, based on other factors like availability, consumer demand, and competition. More specifically, regression’s main focus is to help you uncover the exact relationship between two (or more) variables in a given data set.

7. Prediction. Prediction is one of the most valuable data mining techniques, since it’s used to project the types of data you’ll see in the future. In many cases, just recognizing and understanding historical trends is enough to chart a somewhat accurate prediction of what will happen in the future. For example, you might review consumers’ credit histories and past purchases to predict whether they’ll be a credit risk in the future.

3. Case studies on Current trends and application of Data warehouse & Data mining Techniques

Data mining concepts are still evolving and here are the latest trends that we get to see in this field −

- Application Exploration.
- Scalable and interactive data mining methods.
- Integration of data mining with database systems, data warehouse systems and web database systems.
- SStandardization of data mining query language.
- Visual data mining.
- New methods for mining complex types of data.
- Biological data mining.
- Data mining and software engineering.
- Web mining.
- Distributed data mining.
- Real time data mining.
- Multi database data mining.
- Privacy protection and information security in data mining.

Applications of Data Warehouse and Data mining techniques

Service providers

The first example of Data Mining and Business Intelligence comes from service providers in the mobile phone and utilities industries. Mobile phone and utilities companies use Data Mining and Business Intelligence to predict ‘churn’, the terms they use for when a customer leaves their company to get their phone/gas/broadband from another provider. They collate billing information, customer services interactions, website visits and other metrics to give each customer a probability score, then target offers and incentives to customers whom they perceive to be at a higher risk of churning.

Retail

Another example of Data Mining and Business Intelligence comes from the retail sector. Retailers segment customers into ‘Recency, Frequency, Monetary’ (RFM) groups and target marketing and promotions to those different groups. A customer who spends little but often and last did so recently will be handled differently to a customer who spent big but only once, and also some time ago. The former may receive a loyalty, upsell and cross-sell offers, whereas the latter may be offered a win-back deal, for instance.

E-commerce

Perhaps some of the most well -known examples of Data Mining and Analytics come from E-commerce sites. Many E-commerce companies use Data Mining and Business Intelligence to offer cross-sells and up-sells through their websites. One of the most famous of these is, of course, Amazon, who use sophisticated mining techniques to drive their, ‘People who viewed that product, also liked this’ functionality.

Supermarkets

Supermarkets provide another good example of Data Mining and Business Intelligence in action. Famously, supermarket loyalty card programmes are usually driven mostly, if not solely, by the desire to gather comprehensive data about customers for use in data mining. One notable recent example of this was with the US retailer Target. As part of its Data Mining programme, the company developed rules to predict if their shoppers were likely to be pregnant. By looking at the contents of their customers’ shopping baskets, they could spot customers who they thought were likely to be expecting and begin targeting promotions for nappies (diapers), cotton wool and so on. The prediction was so accurate that Target made the news by sending promotional coupons to families who did not yet realise (or who had not yet announced) they were pregnant!

Crime agencies

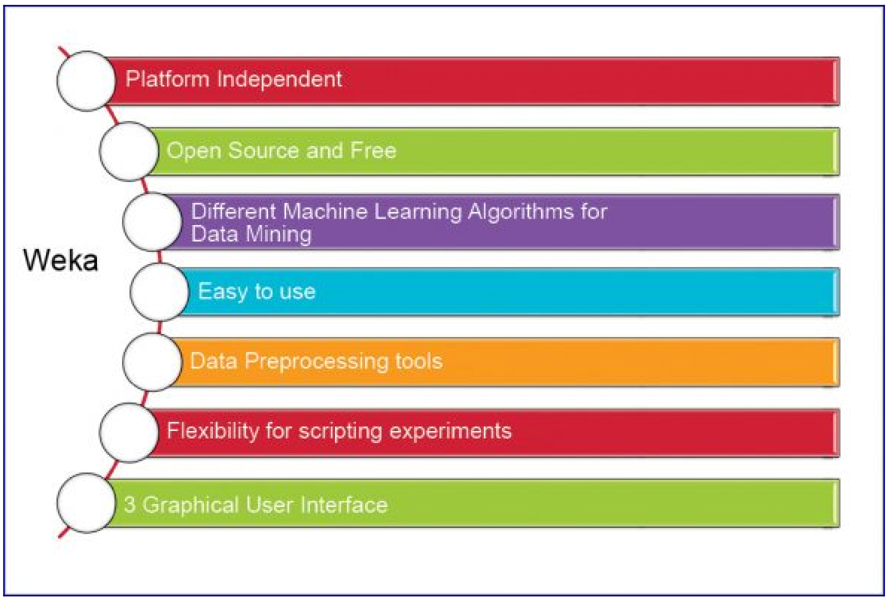
The use of Data Mining and Business Intelligence is not solely reserved for corporate applications and this is shown in our final example. Beyond corporate applications, crime prevention agencies use analytics and Data Mining to spot trends across myriads of data – helping with everything from where to deploy police manpower (where is crime most likely to happen and when?), who to search at a border crossing (based on age/type of vehicle, number/age of occupants, border crossing history) and even which intelligence to take seriously in counter-terrorism activities.

4. Brief Review of WEKA TOOL

Weka is data mining software that uses a collection of machine learning algorithms. These algorithms can be applied directly to the data or called from the Java code.
Weka is a collection of tools for:

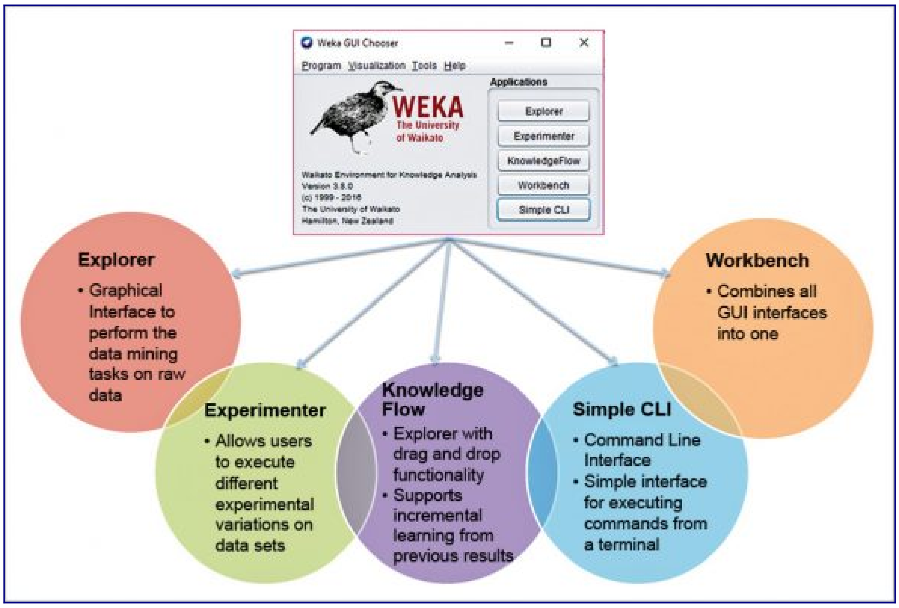
- Regression
- Clustering
- Association
- Data pre-processing
- Classification
- Visualisation

The features of Weka are shown in Figure:



Weka

- Platform Independent
- Open Source and Free
- Different Machine Learning Algorithms for Data Mining
- Easy to use
- Data Preprocessing tools
- Flexibility for scripting experiments
- 3 Graphical User Interface



Waikato Environment for Knowledge Analysis
Version 3.8.0
(c) 1999 - 2016
The University of Waikato
Hamilton, New Zealand

Explorer
- Graphical Interface to perform the data mining tasks on raw data

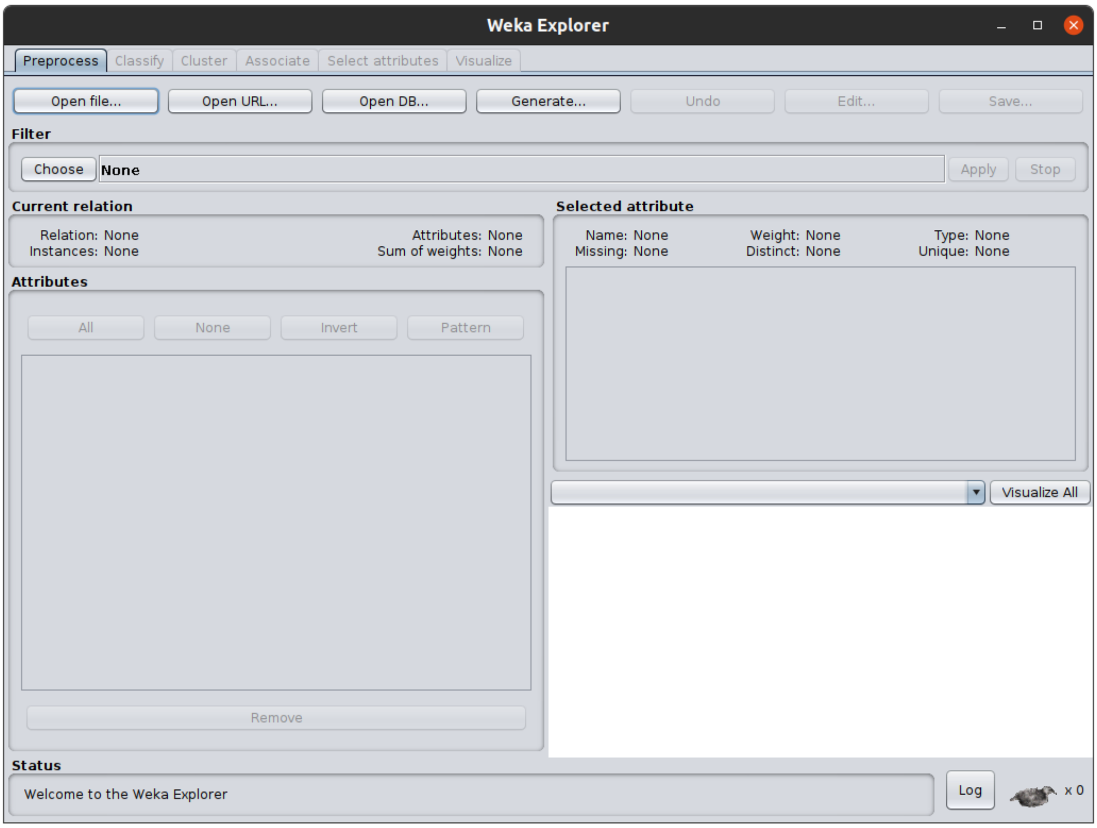
Experimenter
- Allows users to execute different experimental variations on data sets

Knowledge Flow
- Explorer with drag and drop functionality
- Supports incremental learning from previous results

Simple CLI
- Command Line Interface
- Simple interface for executing commands from a terminal

Workbench
- Combines all GUI interfaces into one

Weka Explorer



Filter: None

Relation: None
Instances: None
Attributes: None
Sum of weights: None

Name: None
Missing: None
Weight: None
Distinct: None
Type: None
Unique: None

Welcome to the Weka Explorer

The Weka Explorer is illustrated in Figure and contains a total of six tabs. The tabs are as follows.
1) *Preprocess:* This allows us to choose the data file.
2) *Classify:* This allows us to apply and experiment with different algorithms on preprocessed data files.
3) *Cluster:* This allows us to apply different clustering tools, which identify clusters within the data file.
4) *Association:* This allows us to apply association rules, which identify the association within the data.
5) *Select attributes:* These allow us to see the changes on the inclusion and exclusion of attributes from the experiment.
6) *Visualize:* This allows us to see the possible visualisation produced on the data set in a 2D format, in scatter plot and bar graph output.
The user cannot move between the different tabs until the initial preprocessing of the data set has been completed.

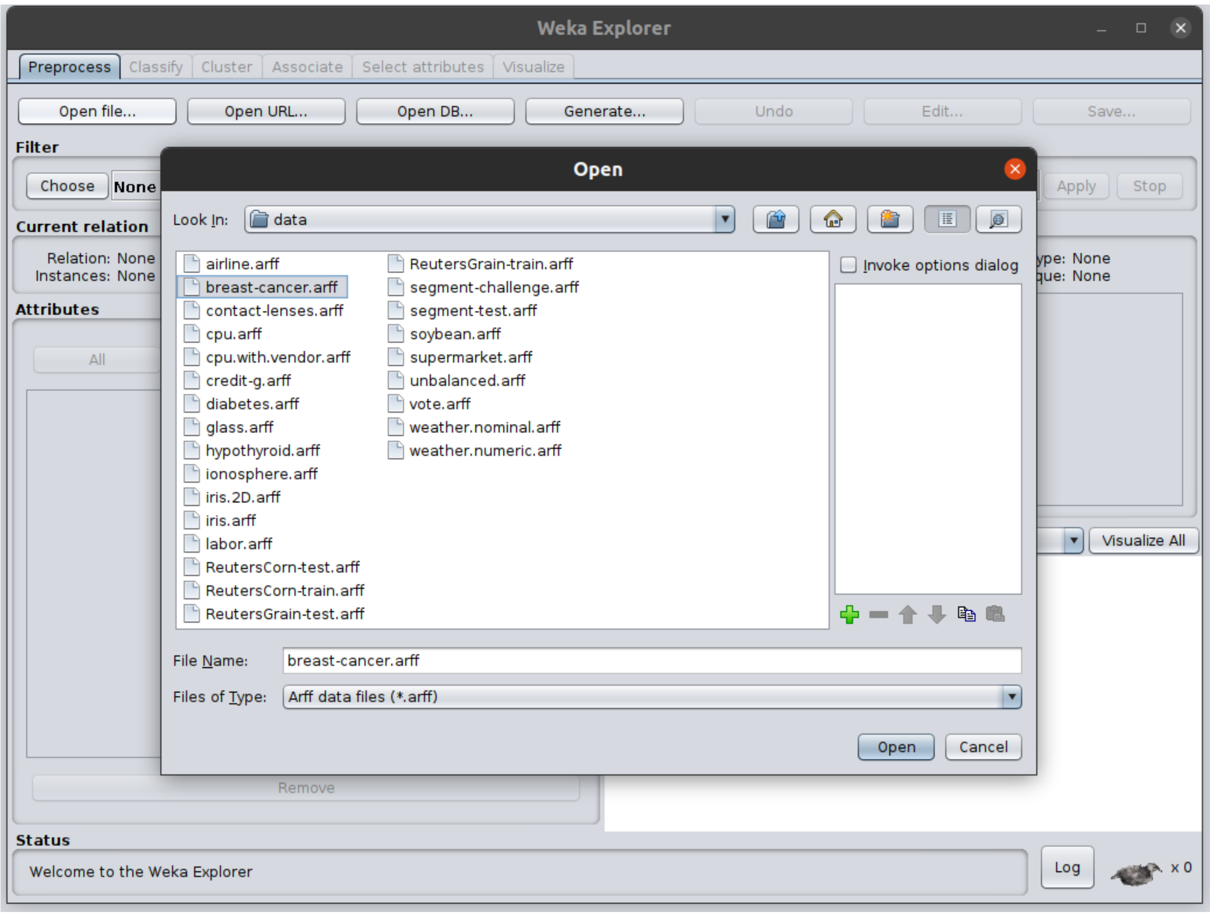
Preprocessing

Data preprocessing is a must. There are three ways to inject the data for preprocessing:

- Open File – enables the user to select the file from the local machine

- Open URL – enables the user to select the data file from different locations
- Open Database – enables users to retrieve a data file from a database source

A screen for selecting a file from the local machine to be preprocessed is shown in Figure.
After loading the data in Explorer, we can refine the data by selecting different options. We can also select or remove the attributes as per our need and even apply filters on data to refine the result.



Relation: None
Instances: None

ype: None
que: None

Look In: data

File Name: breast-cancer.arff

Files of Type: Arff data files (*.arff)

Welcome to the Weka Explorer

Classification

To predict nominal or numeric quantities, we have classifiers in Weka. Available learning schemes are decision-trees and lists, support vector machines, instance-based classifiers, logistic regression and Bayes’ nets. Once the data has been loaded, all the tabs are enabled. Based on the requirements and by trial and error, we can find out the most suitable algorithm to produce an easily understandable representation of data.

Before running any classification algorithm, we need to set test options. Available test options are listed below.
Use training set: Evaluation is based on how well it can predict the class of the instances it was trained on.

Supplied training set: Evaluation is based on how well it can predict the class of a set of instances loaded from a file.

Cross-validation: Evaluation is based on cross-validation by using the number of folds entered in the 'Folds' text field.

Split percentage: Evaluation is based on how well it can predict a certain percentage of the data, held out for testing by using the values entered in the '%' field.

To classify the data set based on the characteristics of attributes, Weka uses classifiers.

Clustering

The cluster tab enables the user to identify similarities or groups of occurrences within the data set. Clustering can provide data for the user to analyse. The training set, percentage split, supplied test set and classes are used for clustering, for which the user can ignore some attributes from the data set, based on the requirements. Available clustering schemes in Weka are k-Means, EM, Cobweb, X-means and FarthestFirst.

Association

The only available scheme for association in Weka is the Apriori algorithm. It identifies statistical dependencies between clusters of attributes, and only works with discrete data. The Apriori algorithm computes all the rules having minimum support and exceeding a given confidence level.

Attribute selection

Attribute selection crawls through all possible combinations of attributes in the data to decide which of these will best fit the desired calculation—which subset of attributes works best for prediction. The attribute selection method contains two parts.

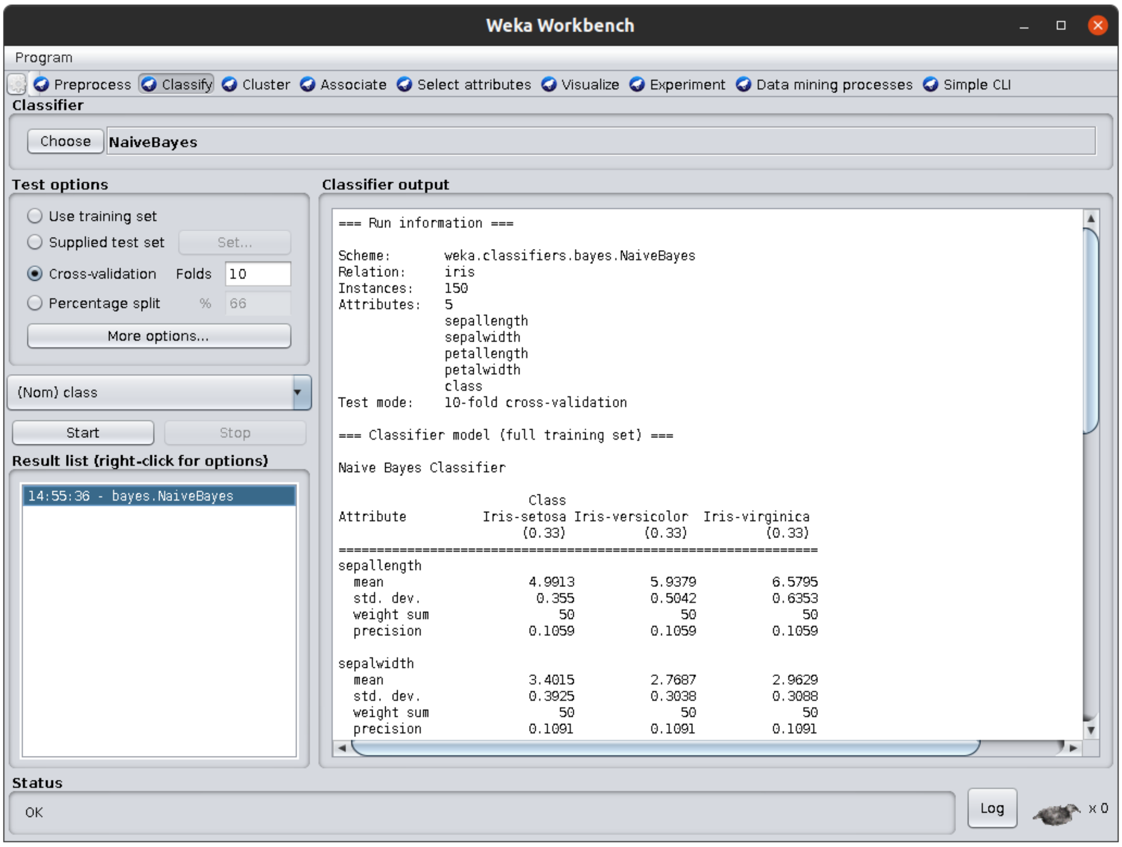
- *Search method:* Best-first, forward selection, random, exhaustive, genetic algorithm, ranking algorithm
- *Evaluation method:* Correlation-based, wrapper, information gain, chi-squared
  All the available attributes are used in the evaluation of the data set by default. But it enables users to exclude some of them if they want to.

Visualisation

The user can see the final piece of the puzzle, derived throughout the process. It allows users to visualise a 2D representation of data, and is used to determine the difficulty of the learning problem. We can visualise single attributes (1D) and pairs of attributes (2D), and rotate 3D visualisations in Weka. It has the Jitter option to deal with nominal attributes and to detect ‘hidden’ data points.

5. Practical of Classification

Steps: Open Weka > Workbench > Open File > Classify > Choose Classifier > Start **Classifier Type :** NaiveBayes



Classifier: NaiveBayes

Test options:
- ○ Use training set
- ○ Supplied test set
- ● Cross-validation Folds 10
- ○ Percentage split % 66

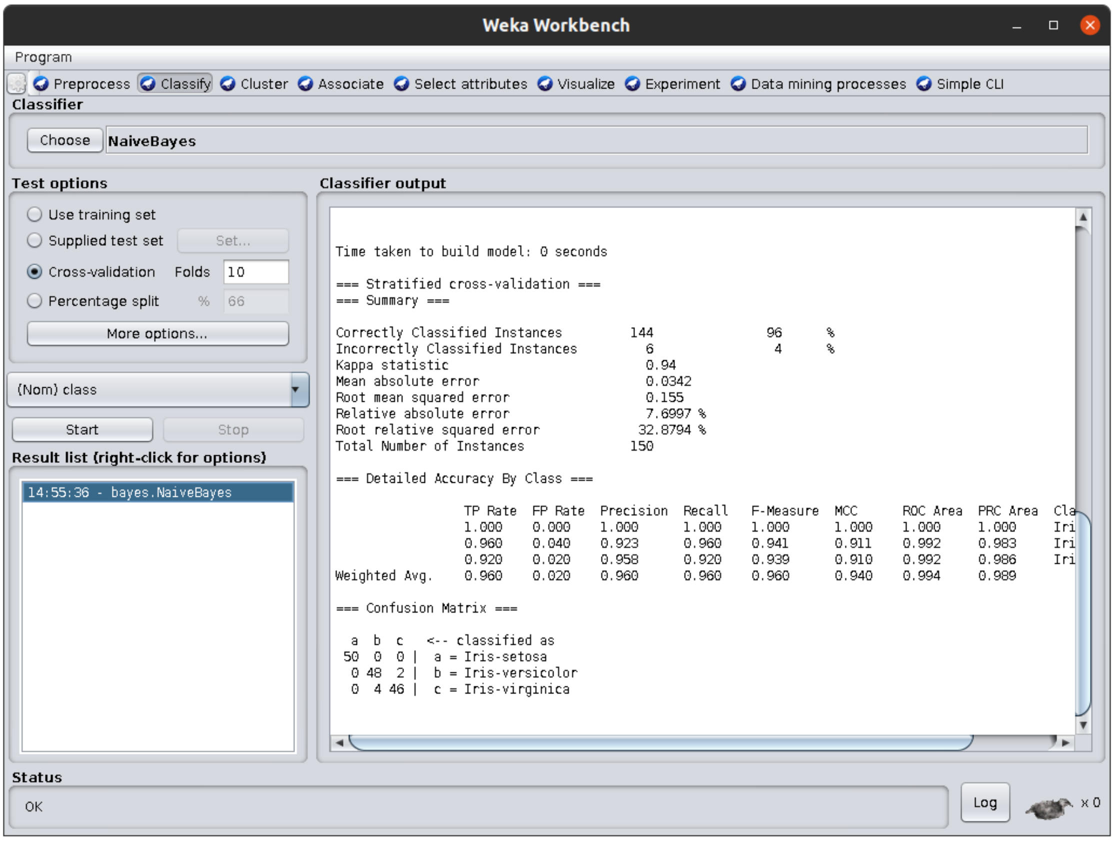
(Nom) class

Result list (right-click for options): 14:55:36 - bayes.NaiveBayes

Classifier output

```
=== Run information ===

Scheme:       weka.classifiers.bayes.NaiveBayes
Relation:     iris
Instances:    150
Attributes:   5
              sepallength
              sepalwidth
              petallength
              petalwidth
              class
Test mode:    10-fold cross-validation

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

Naive Bayes Classifier

                    Class
Attribute         Iris-setosa Iris-versicolor  Iris-virginica
                       (0.33)          (0.33)          (0.33)
==============================================================
sepallength
  mean                 4.9913          5.9379          6.5795
  std. dev.             0.355          0.5042          0.6353
  weight sum               50              50              50
  precision            0.1059          0.1059          0.1059

sepalwidth
  mean                 3.4015          2.7687          2.9629
  std. dev.            0.3925          0.3038          0.3088
  weight sum               50              50              50
  precision            0.1091          0.1091          0.1091
```

Status: OK



Classifier: NaiveBayes

Test options:
- ○ Use training set
- ○ Supplied test set
- ● Cross-validation Folds 10
- ○ Percentage split % 66

(Nom) class

Result list (right-click for options): 14:55:36 - bayes.NaiveBayes

Classifier output

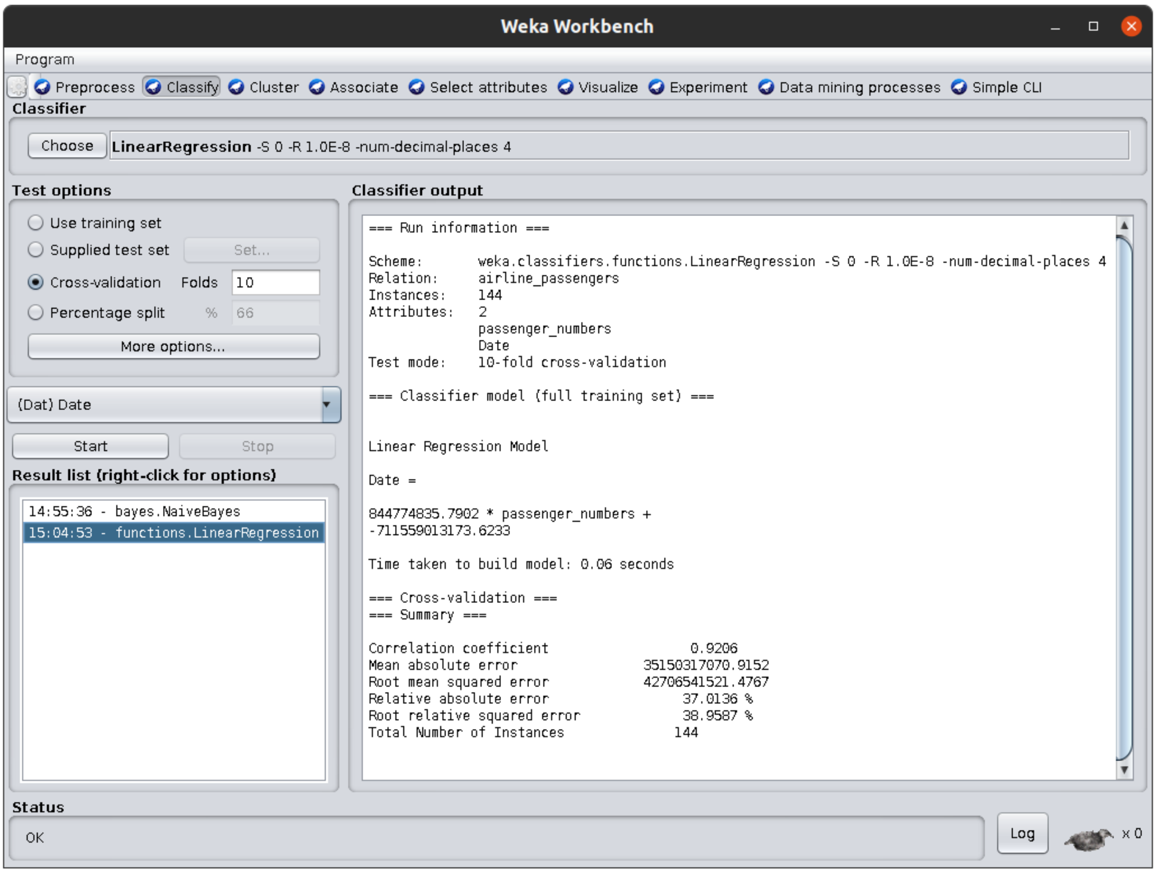
```
Time taken to build model: 0 seconds

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

Correctly Classified Instances         144               96      %
Incorrectly Classified Instances         6                4      %
Kappa statistic                          0.94
Mean absolute error                      0.0342
Root mean squared error                  0.155
Relative absolute error                  7.6997 %
Root relative squared error             32.8794 %
Total Number of Instances              150

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cla
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     Iri
                 0.960    0.040    0.923      0.960    0.941      0.911    0.992     0.983     Iri
                 0.920    0.020    0.958      0.920    0.939      0.910    0.992     0.986     Iri
Weighted Avg.    0.960    0.020    0.960      0.960    0.960      0.940    0.994     0.989

=== Confusion Matrix ===

  a  b  c   <-- classified as
 50  0  0 |  a = Iris-setosa
  0 48  2 |  b = Iris-versicolor
  0  4 46 |  c = Iris-virginica
```

Status: OK

6. Practical of Regression Analysis

Steps: Open Weka > Workbench > Open File > Classify > Choose Classifier > Start

Classifier : Linear Regression



Classifier: LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4

Test options:

○ Use training set

○ Supplied test set

● Cross-validation Folds 10

○ Percentage split % 66

(Dat) Date

Result list:

14:55:36 - bayes.NaiveBayes

15:04:53 - functions.LinearRegression

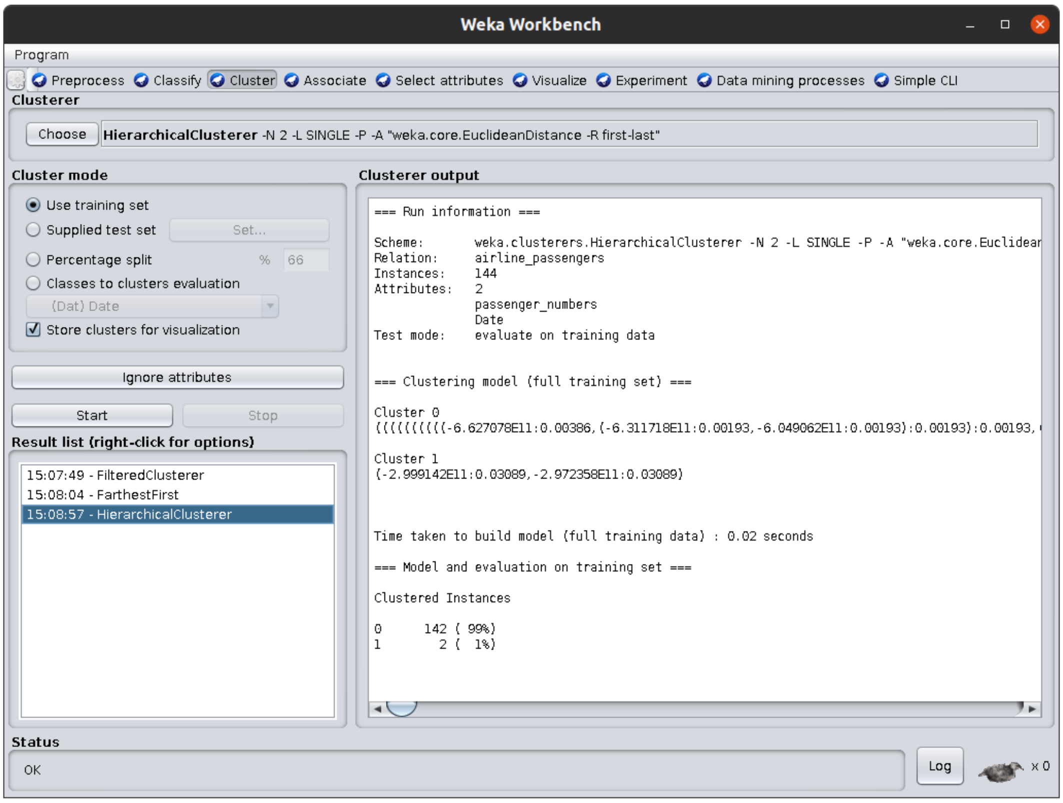
Status: OK

```
=== Run information ===

Scheme:       weka.classifiers.functions.LinearRegression -S 0 -R 1.0E-8 -num-decimal-places 4
Relation:     airline_passengers
Instances:    144
Attributes:   2
              passenger_numbers
              Date
Test mode:    10-fold cross-validation

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


Linear Regression Model

Date =

844774835.7902 * passenger_numbers +
-711559013173.6233

Time taken to build model: 0.06 seconds

=== Cross-validation ===
=== Summary ===

Correlation coefficient                  0.9206
Mean absolute error                35150317070.9152
Root mean squared error            42706541521.4767
Relative absolute error                 37.0136 %
Root relative squared error             38.9587 %
Total Number of Instances              144
```

7. Practical of Clustering

Steps: Open Weka > Workbench > Open File > Cluster > Choose Clusterer > Start

Clusterer : Hierarchical Clusterer



Clusterer: **HierarchicalClusterer** -N 2 -L SINGLE -P -A "weka.core.EuclideanDistance -R first-last"

Cluster mode

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

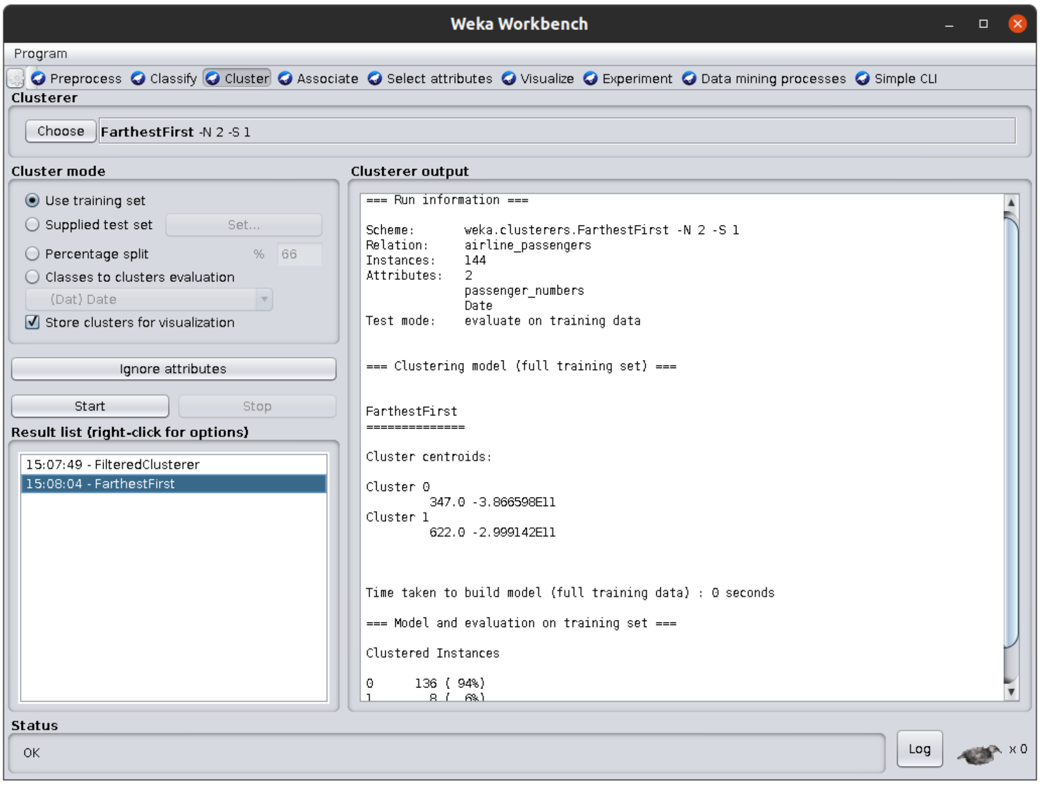
Result list (right-click for options)

- 15:07:49 - FilteredClusterer
- 15:08:04 - FarthestFirst
- 15:08:57 - HierarchicalClusterer

Clusterer output

```
=== Run information ===

Scheme:       weka.clusterers.HierarchicalClusterer -N 2 -L SINGLE -P -A "weka.core.Euclidear
Relation:     airline_passengers
Instances:    144
Attributes:   2
              passenger_numbers
              Date
Test mode:    evaluate on training data


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

Cluster 0
((((((((((-6.627078E11:0.00386,(-6.311718E11:0.00193,-6.049062E11:0.00193):0.00193):0.00193,(

Cluster 1
(-2.999142E11:0.03089,-2.972358E11:0.03089)



Time taken to build model (full training data) : 0.02 seconds

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

Clustered Instances

0      142 ( 99%)
1        2 (  1%)
```

Status: OK



Clusterer: **FarthestFirst** -N 2 -S 1

Cluster mode

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

Result list (right-click for options)

- 15:07:49 - FilteredClusterer
- 15:08:04 - FarthestFirst

Clusterer output

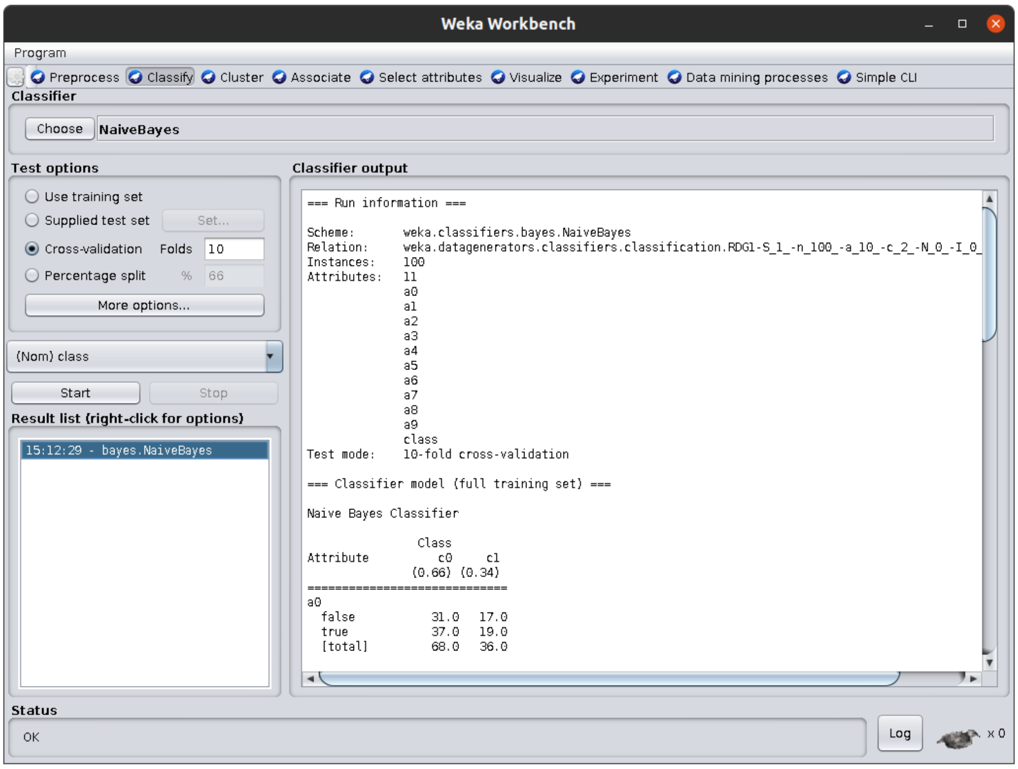
```
=== Run information ===

Scheme:       weka.clusterers.FarthestFirst -N 2 -S 1
Relation:     airline_passengers
Instances:    144
Attributes:   2
              passenger_numbers
              Date
Test mode:    evaluate on training data


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


FarthestFirst
==============

Cluster centroids:

Cluster 0
         347.0 -3.866598E11
Cluster 1
         622.0 -2.999142E11



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

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

Clustered Instances

0      136 ( 94%)
1        8 (  6%)
```

Status: OK

8. Practical of Prediction

Steps: Open Weka > Workbench > Open File > Classify > Choose Classifier > Start

Classifier : NaiveBayes



Classifier: NaiveBayes

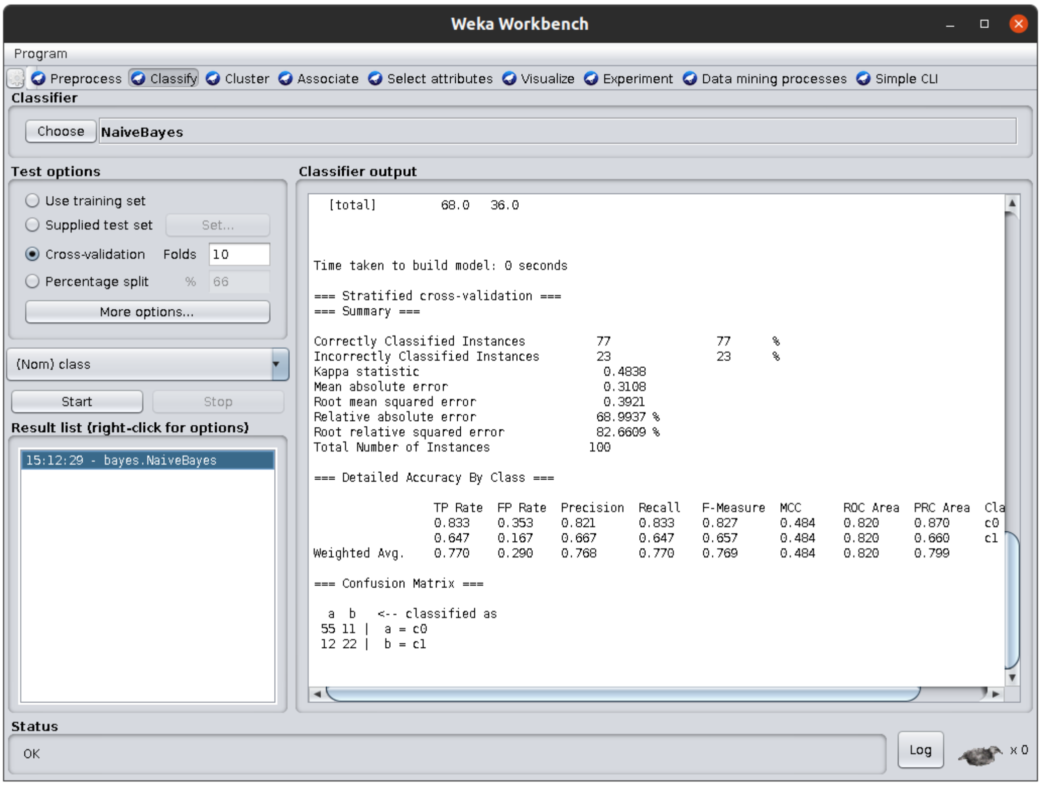
Test options:
○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) class

Result list: 15:12:29 - bayes.NaiveBayes

```
=== Run information ===

Scheme:       weka.classifiers.bayes.NaiveBayes
Relation:     weka.datagenerators.classifiers.classification.RDG1-S_1_-n_100_-a_10_-c_2_-N_0_-I_0_
Instances:    100
Attributes:   11
              a0
              a1
              a2
              a3
              a4
              a5
              a6
              a7
              a8
              a9
              class
Test mode:    10-fold cross-validation

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

Naive Bayes Classifier

               Class
Attribute         c0     c1
               (0.66) (0.34)
============================
a0
  false          31.0   17.0
  true           37.0   19.0
  [total]        68.0   36.0
```

Status: OK



Classifier: NaiveBayes

Test options:
○ Use training set
○ Supplied test set
● Cross-validation Folds 10
○ Percentage split % 66

(Nom) class

Result list: 15:12:29 - bayes.NaiveBayes

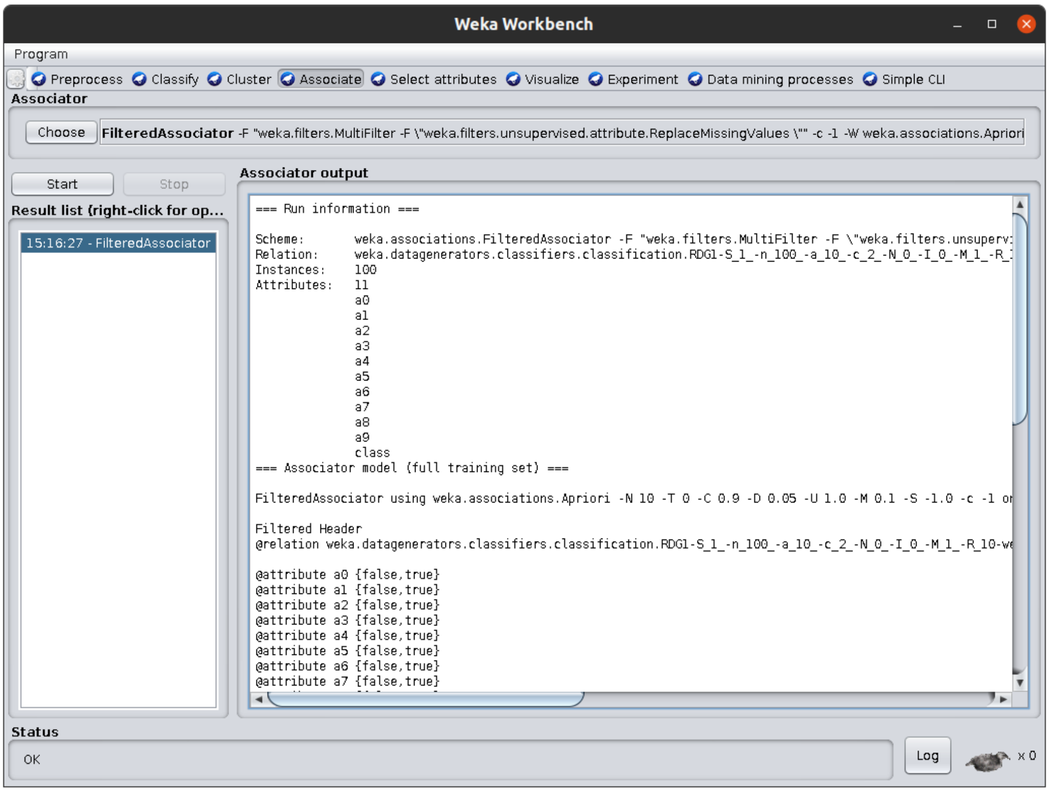
```
  [total]        68.0   36.0



Time taken to build model: 0 seconds

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

Correctly Classified Instances          77               77      %
Incorrectly Classified Instances        23               23      %
Kappa statistic                          0.4838
Mean absolute error                      0.3108
Root mean squared error                  0.3921
Relative absolute error                 68.9937 %
Root relative squared error             82.6609 %
Total Number of Instances              100

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Cla
                 0.833    0.353    0.821      0.833    0.827      0.484    0.820     0.870     c0
                 0.647    0.167    0.667      0.647    0.657      0.484    0.820     0.660     c1
Weighted Avg.    0.770    0.290    0.768      0.770    0.769      0.484    0.820     0.799

=== Confusion Matrix ===

  a  b   <-- classified as
 55 11 |  a = c0
 12 22 |  b = c1
```

Status: OK

9. Practical of Association Rules

Steps: Open Weka > Workbench > Open File > Associate > Choose Associator > Start

Associator : Filtered Associator

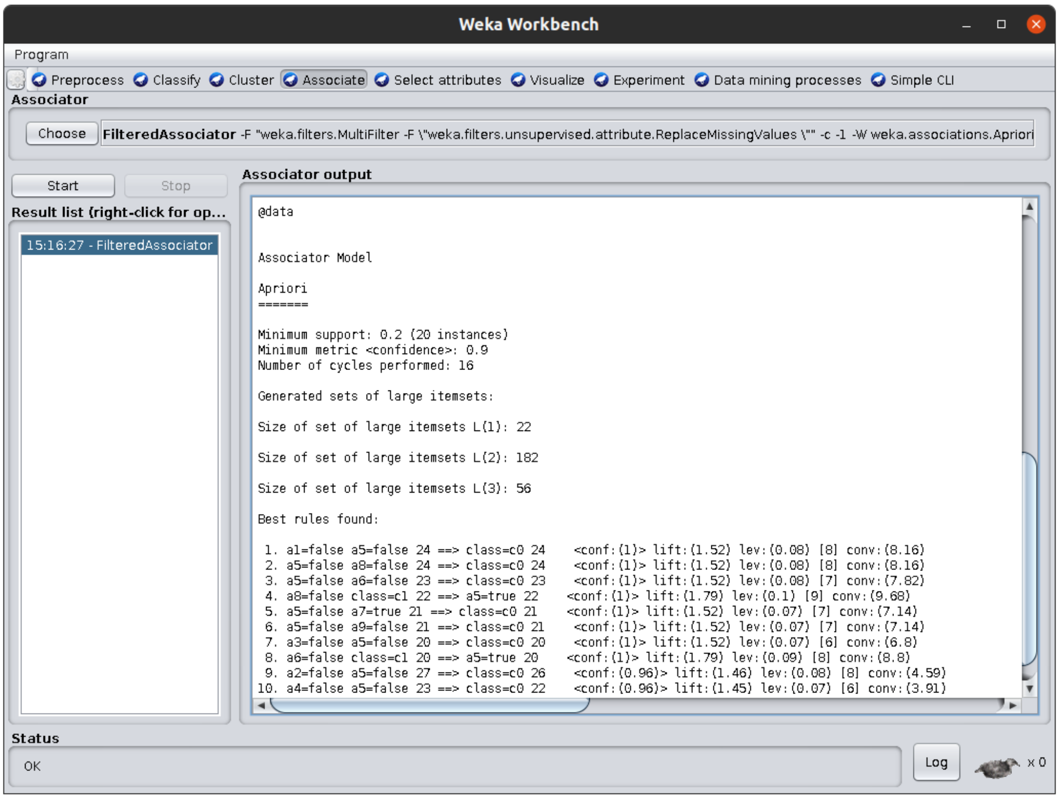


FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsupervised.attribute.ReplaceMissingValues \"" -c -1 -W weka.associations.Apriori

15:16:27 - FilteredAssociator

```
=== Run information ===

Scheme:       weka.associations.FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsuperv
Relation:     weka.datagenerators.classifiers.classification.RDG1-S_1_-n_100_-a_10_-c_2_-N_0_-I_0_-M_1_-R_
Instances:    100
Attributes:   11
              a0
              a1
              a2
              a3
              a4
              a5
              a6
              a7
              a8
              a9
              class
=== Associator model (full training set) ===

FilteredAssociator using weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1 or

Filtered Header
@relation weka.datagenerators.classifiers.classification.RDG1-S_1_-n_100_-a_10_-c_2_-N_0_-I_0_-M_1_-R_10-we

@attribute a0 {false,true}
@attribute a1 {false,true}
@attribute a2 {false,true}
@attribute a3 {false,true}
@attribute a4 {false,true}
@attribute a5 {false,true}
@attribute a6 {false,true}
@attribute a7 {false,true}
```

OK



FilteredAssociator -F "weka.filters.MultiFilter -F \"weka.filters.unsupervised.attribute.ReplaceMissingValues \"" -c -1 -W weka.associations.Apriori

15:16:27 - FilteredAssociator

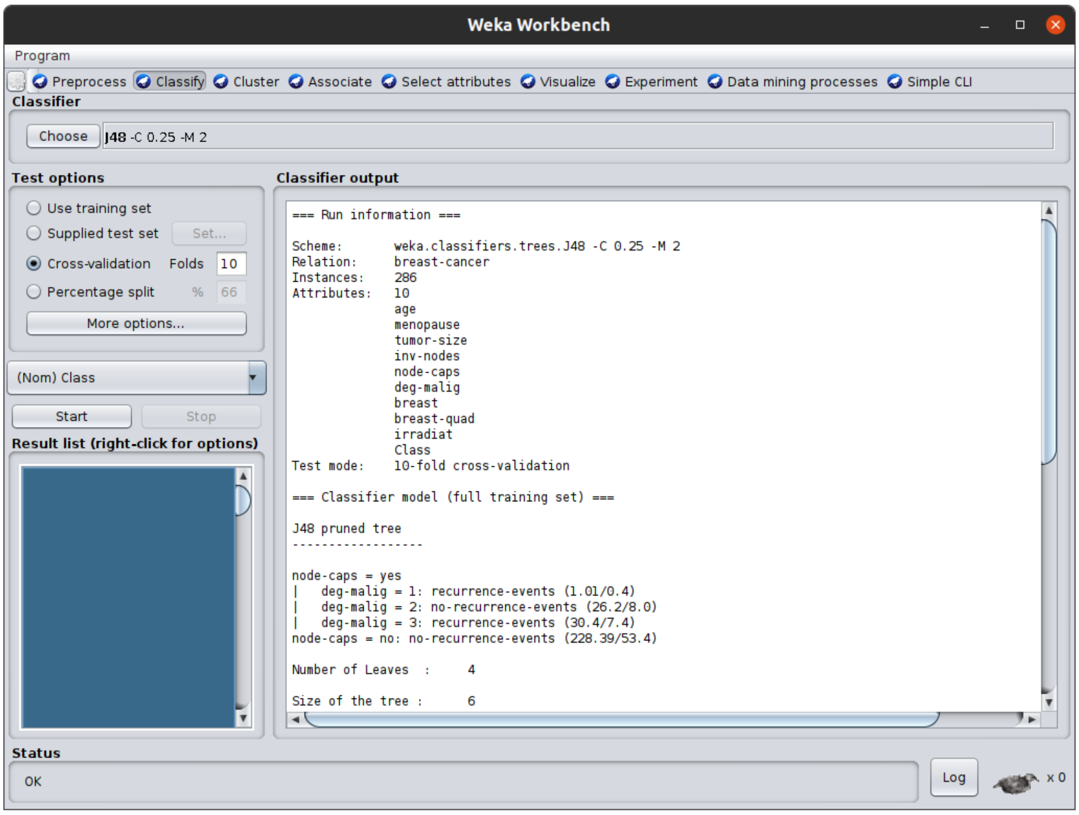
```
@data


Associator Model

Apriori
=======

Minimum support: 0.2 (20 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 16

Generated sets of large itemsets:

Size of set of large itemsets L(1): 22

Size of set of large itemsets L(2): 182

Size of set of large itemsets L(3): 56

Best rules found:

 1. a1=false a5=false 24 ==> class=c0 24    <conf:(1)> lift:(1.52) lev:(0.08) [8] conv:(8.16)
 2. a5=false a8=false 24 ==> class=c0 24    <conf:(1)> lift:(1.52) lev:(0.08) [8] conv:(8.16)
 3. a5=false a6=false 23 ==> class=c0 23    <conf:(1)> lift:(1.52) lev:(0.08) [7] conv:(7.82)
 4. a8=false class=c1 22 ==> a5=true 22   <conf:(1)> lift:(1.79) lev:(0.1) [9] conv:(9.68)
 5. a5=false a7=true 21 ==> class=c0 21   <conf:(1)> lift:(1.52) lev:(0.07) [7] conv:(7.14)
 6. a5=false a9=false 21 ==> class=c0 21    <conf:(1)> lift:(1.52) lev:(0.07) [7] conv:(7.14)
 7. a3=false a5=false 20 ==> class=c0 20    <conf:(1)> lift:(1.52) lev:(0.07) [6] conv:(6.8)
 8. a6=false class=c1 20 ==> a5=true 20   <conf:(1)> lift:(1.79) lev:(0.09) [8] conv:(8.8)
 9. a2=false a5=false 27 ==> class=c0 26    <conf:(0.96)> lift:(1.46) lev:(0.08) [8] conv:(4.59)
10. a4=false a5=false 23 ==> class=c0 22    <conf:(0.96)> lift:(1.45) lev:(0.07) [6] conv:(3.91)
```

OK

10. Practical of Outlier detection

Steps: Open Weka > Workbench > Open File > Classify > Choose Classifier > Start

Classifier : J48



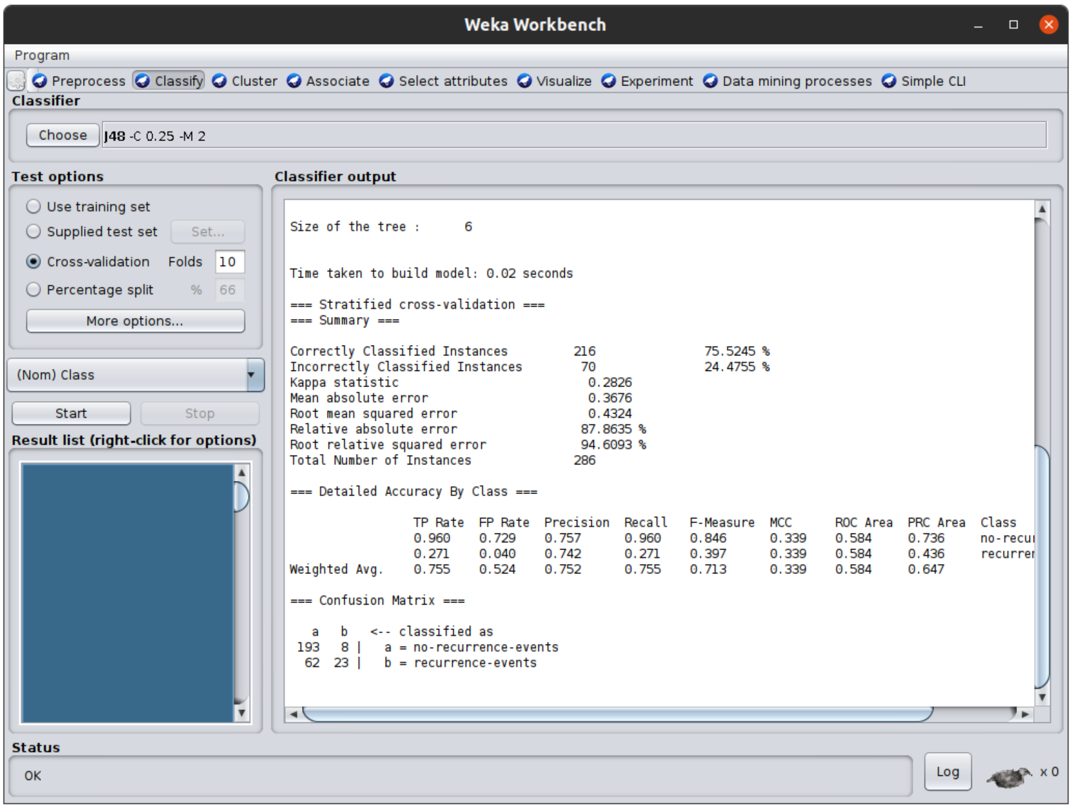
Classifier: J48 -C 0.25 -M 2

Test options: ● Cross-validation Folds 10

(Nom) Class

Status: OK

```
=== Run information ===

Scheme:       weka.classifiers.trees.J48 -C 0.25 -M 2
Relation:     breast-cancer
Instances:    286
Attributes:   10
              age
              menopause
              tumor-size
              inv-nodes
              node-caps
              deg-malig
              breast
              breast-quad
              irradiat
              Class
Test mode:    10-fold cross-validation

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

J48 pruned tree
------------------

node-caps = yes
|   deg-malig = 1: recurrence-events (1.01/0.4)
|   deg-malig = 2: no-recurrence-events (26.2/8.0)
|   deg-malig = 3: recurrence-events (30.4/7.4)
node-caps = no: no-recurrence-events (228.39/53.4)

Number of Leaves  :     4

Size of the tree :      6
```



Classifier: J48 -C 0.25 -M 2

Test options: ● Cross-validation Folds 10

(Nom) Class

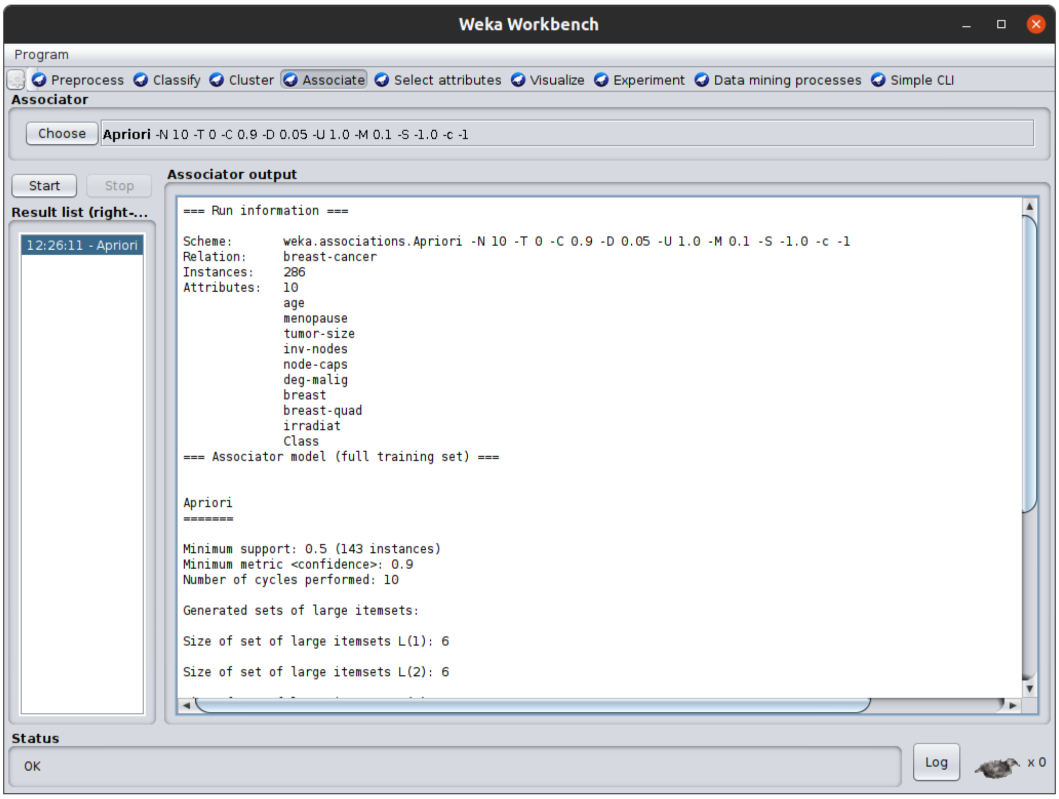
Status: OK

```
Size of the tree :      6


Time taken to build model: 0.02 seconds

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

Correctly Classified Instances         216               75.5245 %
Incorrectly Classified Instances        70               24.4755 %
Kappa statistic                          0.2826
Mean absolute error                      0.3676
Root mean squared error                  0.4324
Relative absolute error                 87.8635 %
Root relative squared error             94.6093 %
Total Number of Instances              286

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.960    0.729    0.757      0.960    0.846      0.339    0.584     0.736     no-recur
                 0.271    0.040    0.742      0.271    0.397      0.339    0.584     0.436     recurrer
Weighted Avg.    0.755    0.524    0.752      0.755    0.713      0.339    0.584     0.647

=== Confusion Matrix ===

   a   b   <-- classified as
 193   8 |   a = no-recurrence-events
  62  23 |   b = recurrence-events
```

11. Practical of Sequential Patterns

Steps: Open Weka > Workbench > Open File > Associate > Choose Associator > Start

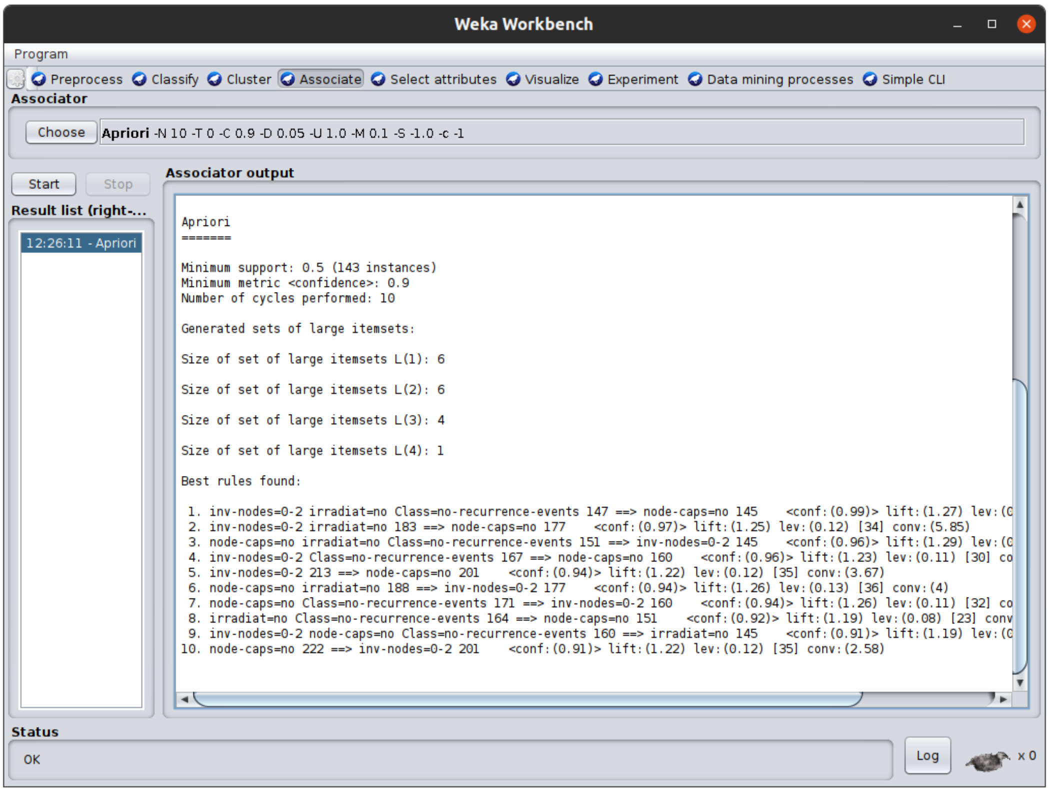
Associator : Apriori



Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
=== Run information ===

Scheme:       weka.associations.Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1
Relation:     breast-cancer
Instances:    286
Attributes:   10
              age
              menopause
              tumor-size
              inv-nodes
              node-caps
              deg-malig
              breast
              breast-quad
              irradiat
              Class
=== Associator model (full training set) ===


Apriori
=======

Minimum support: 0.5 (143 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 10

Generated sets of large itemsets:

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

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

Status: OK



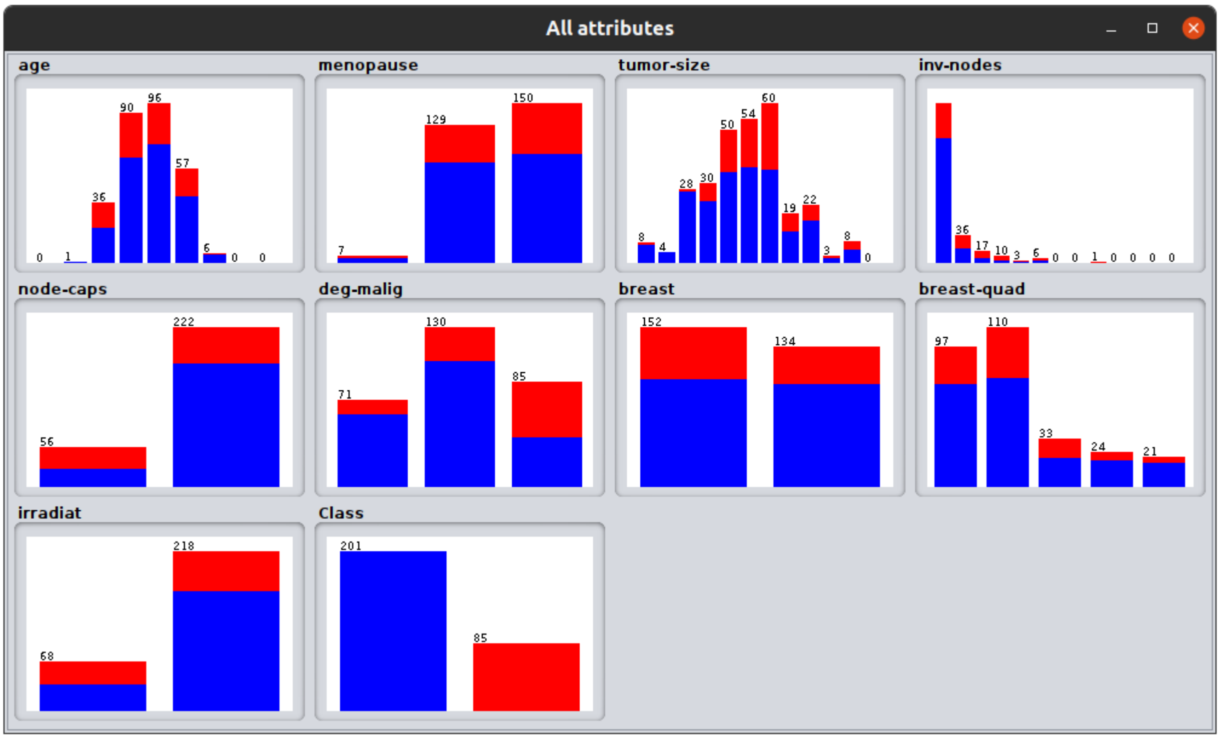
Apriori -N 10 -T 0 -C 0.9 -D 0.05 -U 1.0 -M 0.1 -S -1.0 -c -1

```
Apriori
=======

Minimum support: 0.5 (143 instances)
Minimum metric <confidence>: 0.9
Number of cycles performed: 10

Generated sets of large itemsets:

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

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

Size of set of large itemsets L(3): 4

Size of set of large itemsets L(4): 1

Best rules found:

 1. inv-nodes=0-2 irradiat=no Class=no-recurrence-events 147 ==> node-caps=no 145    <conf:(0.99)> lift:(1.27) lev:(0
 2. inv-nodes=0-2 irradiat=no 183 ==> node-caps=no 177    <conf:(0.97)> lift:(1.25) lev:(0.12) [34] conv:(5.85)
 3. node-caps=no irradiat=no Class=no-recurrence-events 151 ==> inv-nodes=0-2 145    <conf:(0.96)> lift:(1.29) lev:(0
 4. inv-nodes=0-2 Class=no-recurrence-events 167 ==> node-caps=no 160    <conf:(0.96)> lift:(1.23) lev:(0.11) [30] co
 5. inv-nodes=0-2 213 ==> node-caps=no 201    <conf:(0.94)> lift:(1.22) lev:(0.12) [35] conv:(3.67)
 6. node-caps=no irradiat=no 188 ==> inv-nodes=0-2 177    <conf:(0.94)> lift:(1.26) lev:(0.13) [36] conv:(4)
 7. node-caps=no Class=no-recurrence-events 171 ==> inv-nodes=0-2 160    <conf:(0.94)> lift:(1.26) lev:(0.11) [32] co
 8. irradiat=no Class=no-recurrence-events 164 ==> node-caps=no 151    <conf:(0.92)> lift:(1.19) lev:(0.08) [23] conv
 9. inv-nodes=0-2 node-caps=no Class=no-recurrence-events 160 ==> irradiat=no 145    <conf:(0.91)> lift:(1.19) lev:(0
10. node-caps=no 222 ==> inv-nodes=0-2 201    <conf:(0.91)> lift:(1.22) lev:(0.12) [35] conv:(2.58)
```

Status: OK

12. Data Visualization

Steps: Open Weka > Workbench > Open File > Visualize all



13. Enlist all the available tools for Data Mining

1 Xplenty

Xplenty provides a platform that has functionalities to integrate, process, and prepare data for analytics. Businesses will be able to make most of the opportunities offered by big data with the help of Xplenty and that too without investing in related personnel, hardware, and software. It is a complete toolkit for building data pipelines.

You will be able to implement complex data preparation functions through rich expression language. It has an intuitive interface to implement ETL, ELT, or a replication solution. You will be able to orchestrate and schedule pipelines through a workflow engine.

- Xplenty is the data integration platform for all. It offers the no-code and low-code options.
- An API component will provide advanced customization and flexibility.
- It has functionalities to transfer and transform data between databases and data warehouses.
- It provides support through email, chat, phone, and an online meeting.

Availability: Licensed tools.

2 Rapid Miner

Availability: Open source

Rapid Miner is one of the best predictive analysis system developed by the company with the same name as the Rapid Miner. It is written in JAVA programming language. It provides an integrated environment for deep learning, text mining, machine learning & predictive analysis.

The tool can be used for over a vast range of applications including for business applications, commercial applications, training, education, research, application development, machine learning.

Rapid Miner offers the server as both on premise & in public/private cloud infrastructures. It has a client/server model as its base. Rapid Miner comes with template based frameworks that enable speedy delivery with reduced number of errors (which are quite commonly expected in manual code writing process).

Rapid Miner constitutes of three modules, namely

1. Rapid Miner Studio: This module is for workflow design, prototyping, validation etc.
2. Rapid Miner Server: To operate predictive data models created in studio
3. Rapid Miner Radoop: Executes processes directly in the Hadoop cluster to simplify predictive analysis.

3 Orange

Availability: Open source

Orange is a perfect software suite for machine learning & data mining. It best aids the data visualization and is a component based software. It has been written in Python computing language.

As it is a component-based software, the components of orange are called ‘widgets’. These widgets range from data visualization & pre-processing to an evaluation of algorithms and predictive modeling.

Widgets offer major functionalities like

- Showing data table and allowing to select features
- Reading the data
- Training predictors and to compare learning algorithms
- Visualizing data elements etc.

Additionally, Orange brings a more interactive and fun vibe to the dull analytic tools. It is quite interesting to operate.

Data coming to Orange gets quickly formatted to the desired pattern and it can be easily moved where needed by simply moving/flipping the widgets. Users are quite fascinated by Orange. Orange allows users to make smarter decisions in short time by quickly comparing & analyzing the data.

4 Weka

Availability: Free software

Also known as Waikato Environment is a machine learning software developed at the University of Waikato in New Zealand. It is best suited for data analysis and predictive modeling. It contains algorithms and visualization tools that support machine learning.

Weka has a GUI that facilitates easy access to all its features. It is written in JAVA programming language.

Weka supports major data mining tasks including data mining, processing, visualization, regression etc. It works on the assumption that data is available in the form of a flat file.

Weka can provide access to SQL Databases through database connectivity and can further process the data/results returned by the query.

5 KNIME

Availability: Open Source

KNIME is the best integration platform for data analytics and reporting developed by KNIME.com AG. It operates on the concept of the modular data pipeline. KNIME constitutes of various machine learning and data mining components embedded together.

KNIME has been used widely for pharmaceutical research. In addition, it performs excellently for customer data analysis, financial data analysis, and business intelligence.

KNIME has some brilliant features like quick deployment and scaling efficiency. Users get familiar with KNIME in quite lesser time and it has made predictive analysis accessible to even naive users. KNIME utilizes the assembly of nodes to pre-process the data for analytics and visualization.

6 Sisense

Availability: Licensed

Sisense is extremely useful and best suited BI software when it comes to reporting purposes within the organization. It is developed by the company of same name ‘Sisense’. It has a brilliant capability to handle and process data for the small scale/large scale organizations.

It allows combining data from various sources to build a common repository and further, refines data to generate rich reports that get shared across departments for reporting.

Sisense got awarded as best BI software is 2016 and still, holds a good position.

Sisense generates reports which are highly visual. It is specially designed for users that are non-technical. It allows drag & drop facility as well as widgets.

Different widgets can be selected to generate the reports in form of pie charts, line charts, bar graphs etc. based on the purpose of an organization. Reports can be further drilled down by simply clicking to check details and comprehensive data.

7 SSDT (SQL Server Data Tools)

Availability: Licensed

SSDT is a universal, declarative model that expands all the phases of database development in the Visual Studio IDE. BIDS was the former environment developed by Microsoft to do data analysis and provide business intelligence solutions. Developers use SSDT transact- a design capability of SQL, to build, maintain, debug and refactor databases.

A user can work directly with a database or can work directly with a connected database, thus, providing on or off-premise facility.

Users can use visual studio tools for development of databases like IntelliSense, code navigation tools, and programming support via C#, visual basic etc. SSDT provides **Table Designer** to create new tables as well as edit tables in direct databases as well as connected databases.

Deriving its base from BIDS, which was not compatible with Visual Studio2010, the SSDT BI came into existence and it replaced BIDS.

8 Apache Mahout

Availability: Open source

Apache Mahout is a project developed by Apache Foundation that serves the primary purpose of creating machine learning algorithms. It focuses mainly on data clustering, classification, and collaborative filtering.

Mahout is written in JAVA and includes JAVA libraries to perform mathematical operations like linear algebra and statistics. Mahout is growing continuously as the algorithms implemented inside Apache Mahout are continuously growing. The algorithms of Mahout have implemented a level above Hadoop through mapping/reducing templates.

To key up, Mahout has following major features

- Extensible programming environment
- Pre-made algorithms
- Math experimentation environment
- GPU computes for performance improvement.

9 Oracle Data Mining

Availability: Proprietary License

A component of Oracle Advance Analytics, Oracle data mining software provides excellent data mining algorithms for data classification, prediction, regression and specialized analytics that enables analysts to analyze insights, make better predictions, target best customers, identify cross-selling opportunities & detect fraud.

The algorithms designed inside ODM leverage the potential strengths of Oracle database. The data mining feature of SQL can dig data out of database tables, views, and schemas.

The GUI of Oracle data miner is an extended version of Oracle SQL Developer. It provides a facility of direct ‘drag & drop’ of data inside the database to users thus giving better insight.

10 Rattle

Availability: Open source

Rattle is GUI based data mining tool that uses R stats programming language. Rattle exposes the statistical power of R by providing considerable data mining functionality. Although Rattle has an extensive and well-developed UI, it has an inbuilt log code tab that generates duplicate code for any activity happening at GUI.

The data set generated by Rattle can be viewed as well as edited. Rattle gives the additional facility to review the code, use it for numerous purposes and extend the code without restriction.

14. Case Study on Web Data Mining

The case study is a performance study of two classes that are using online course over the web and the web generating a mining algorithm using a variety of parameters. A number of experiments were conducted, and their results are presented in this part. The experiments were based on different parameter of interest. The parameters varied form the number of input attributes, the sample size of the class, and so on. The results of these experiments prove that the data mining algorithm are very efficient and scalable.

The clustering algorithms used

The Clustering algorithm is based on the Expectation and Maximization algorithm (Microsoft, 2003). This algorithm iterates between two steps. In the first step, called the "expectation" step, the cluster membership of each case is calculated. In the second step, called the "Maximization" step, the parameters of the models are re-estimated using these cluster memberships, which has the following major steps:

1. Assign initial means
2. Assign cases to each mean using some distance measure
3. Compute new means based on members of each cluster
4. Cycle until convergence.

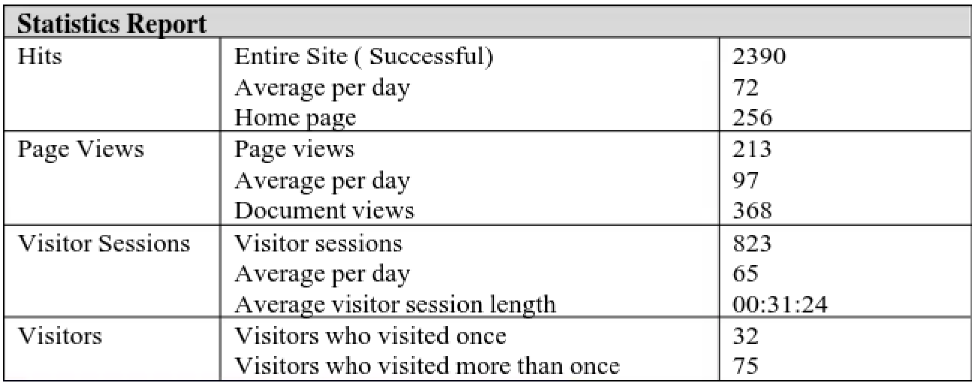
A case is assigned to each cluster with a certain probability and the means of each cluster is shifted based on that iteration. The following table shows a set of data that could be used to predict best achievement. In this study, information was generated on users that included the following list of measurements:

1. Most requested pages
2. Least requested pages
3. Top exit pages
4. Most accessed directories
5. Most downloaded files
6. New versus returning visitors
7. Summary of activity for exam period
8. Summary of activity by time increment
9. Number of views per each page

10. Page not found

The relevant measurements are viewed which implied the following:

Table 1: Statistics on Web site visits



Statistics Report		
Hits	Entire Site (Successful)	2390
	Average per day	72
	Home page	256
Page Views	Page views	213
	Average per day	97
	Document views	368
Visitor Sessions	Visitor sessions	823
	Average per day	65
	Average visitor session length	00:31:24
Visitors	Visitors who visited once	32
	Visitors who visited more than once	75

Results

This brief case study gives a look at what statistics are commonly measured on web sites. The results of these statistics can be used to alter the web site, thereby altering the next user’s experience. Table 1 displays some basic statistics that relate to frequency, length and kind of visitor. In Table 1 additional insights are gained with a breakdown of visitors by each day. This behavior might reflect variation in activity related to exams time or other issues. Monitoring and understanding visitor behavior is the first step in evaluating and improving the web site. Another relevant measurement is how many pages are viewed. This can reflect content as well as navigability. If a majority of visitors viewed only one page, it may imply that they did not find it easy to determine how to take the next step.

15. Major or Mini project Ideas on Data mining

Some major or mini project ideas on Data Mining are following:

Protecting user data in profile-matching social networks

This is one of the convenient data mining projects that has a lot of use in the future. Consider the user profile database maintained by the providers of social networking services, such as online dating sites. The querying users specify certain criteria based on which their profiles are matched with that of other users. This process has to be secure enough to protect against any kind of data breaches. There are some solutions in the market today that use homomorphic encryption and multiple servers for matching user profiles to preserve user privacy.

Sentimental analysis and opinion mining for mobile networks

This project concerns post-publishing applications where a registered user can share text posts or images and also leave comments on posts. Under the prevailing system, users have to go through all the comments manually to filter out verified comments, positive comments, negative remarks, and so on.

With the sentiment analysis and opinion mining system, users can check the status of their post without dedicating much time and effort. It provides an opinion on the comments made on a post and also gives the option to view a graph.

Automated personality classification project

The automatic system analyzes the characteristics and behaviors of participants. And after observing the past patterns of data classification, it predicts a personality type and stores its own patterns in a dataset. This project idea can be summarized as follows:

- Store personality-related data in a database
- Collect associated characteristics for each user
- Extract relevant features from the text entered by the participant
- Examine and display the personality traits
- Interlink personality and user behavior (There can be varying degrees of behavior for a particular personality type)

Such models are commonplace in career guidance services where a student’s personality is matched with suitable career paths. This can be an interesting and useful data mining projects.

ITS: Intelligent Transportation System

A multi-purpose traffic solution generally aims to ensure the following aspects:

- Transport service’s efficiency
- Transport safety
- Reduction in traffic congestion
- Forecast of potential passengers
- Adequate allocation of resources

Consider a project that uses the above system to optimize the process of bus scheduling in a city. ITS is one of the interesting data mining projects for beginners. You can take the past three years’ data from a renowned bus service company, and apply uni-variate multi-linear regression to conduct passengers’ forecasts. Further, you can calculate the minimum number of buses required for optimization in a Generic Algorithm. Finally, you validate your results using statistical techniques like mean absolute percentage error (MAPE) and mean absolute deviation (MAD).