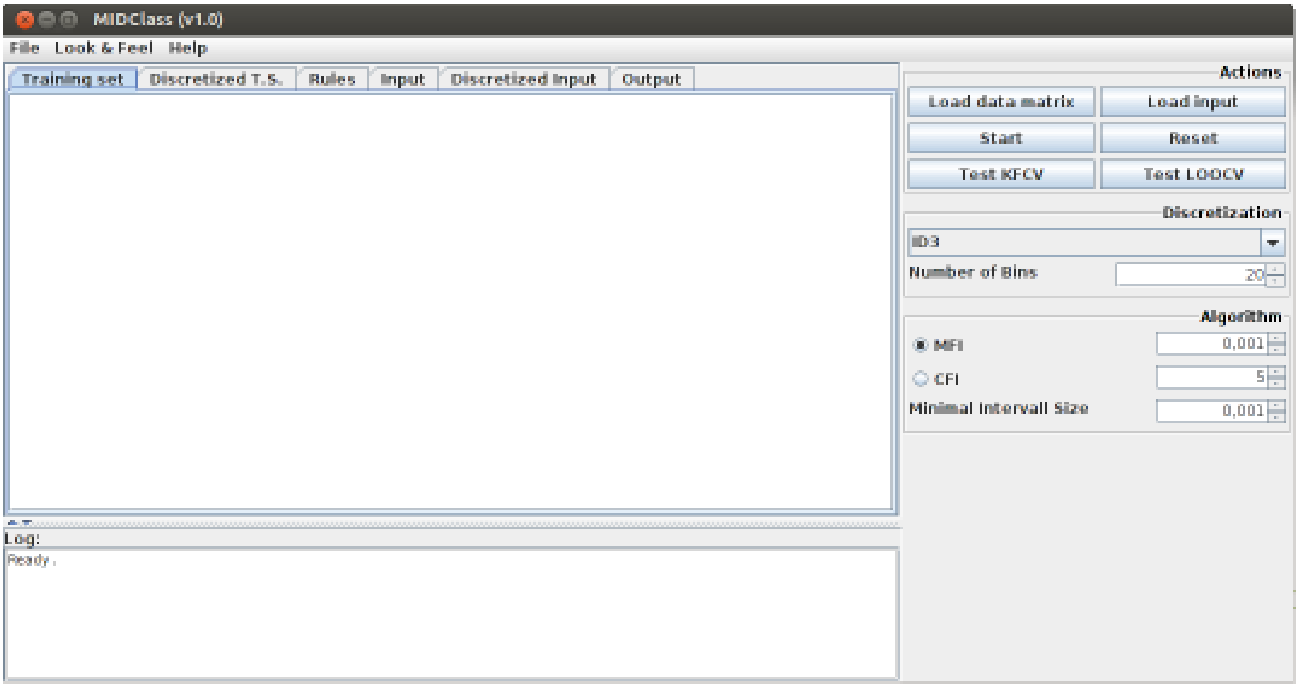
How to install and use MIDClass



ID3

Number of Bins 20

● MFI 0,001

○ CFI 5

Minimal intervall Size 0,001

Ready.

There are different ways to get MIDClass running:

- Installing locally and executing it
- Compiling the source code

Java Jar File

A **JAR** (**J**ava **AR**chive) is an archive file format typically used to aggregate many Java class files and associated metadata and resources (text, images and so on) into one file to distribute application software or libraries on the Java platform.

A JAR file allows Java runtimes to efficiently deploy a set of classes and their associated resources. The elements in a JAR file can be compressed, which, together with the ability to download an entire application in a single request, makes downloading a JAR file much more convenient than separately downloading the many uncompressed files which would form a single Java Application.

To lunch the program, users have to enter this command similar to "java -jar nameOfJarFile.jar".

Installing locally

You can download MIDClass from http://ferrolab.dmi.unict.it/midclass.html
You will get a zip file that can be installed and executed in any Linux operating systems.
The first thing to do is to uncompress the zip file into some folder. Use your preferred application for this, usually doing click or double click over it will start the application that will allow you to uncompress it.

Once uncompressed you will get a folder like MIDClass (the name of the folder could be different depending on the name chose).
That's all, you have already installed MIDClass in your computer and you can work with the graphical interface.

To run MIDClass graphical interface go inside its folder open a terminal and type:

- java -jar MIDClass.jar

Prerequisites

You need to have java installed. Go to this website to download and install; http://www.java.com/en/download/ .
You need to add path of java after set up; go here to do this: http://www.java.com/en/download/help/path.xml.
This will tell your computer where to find the java program. The current version of MIDClass are available for Linux/Unix OS and require the pre-installation of MAFIA 1.4. This package can be downloaded via:
http://himalaya-tools.sourceforge.net/Mafia/
After downloading the tar.gz file, please extract it and put its content into the folder /MIDClass/fi/Mafia-1.4.
Then, you need to compile Mafia package. To do that, please follow these steps:

1. Open a terminal and place into the folder fi/Mafia-1.4 present in to the MIDClass folder.
2. Type ‘make’ to compile the package
3. Type ‘make install’ to install the programs and any data files and documentation
4. Move the file mafia presents in MIDClass /fi/Mafia-1.4/src in MIDClass/fi

Memory configuration

MIDClass is a Java application, and by default it starts with the default memory requirements established by most of the Java applications (usually 256 Megabytes). But most probably you will need more than the default, you will realize this in the case you obtain an exception like this:

java.lang.OutOfMemoryException

There is a way to configure the memory limits for MIDClass the same way that Java does, using the option -Xmx. But this configuration should be specified through an enviroment variable called **MIDCLASS_JAVA_OPTS**. Let's see some examples of how to do this with different operating systems:

Linux

Imagine that you would like to use 1024 megabytes of memory for MIDClass, then edit the **~/.bashrc** file adding a line like this:

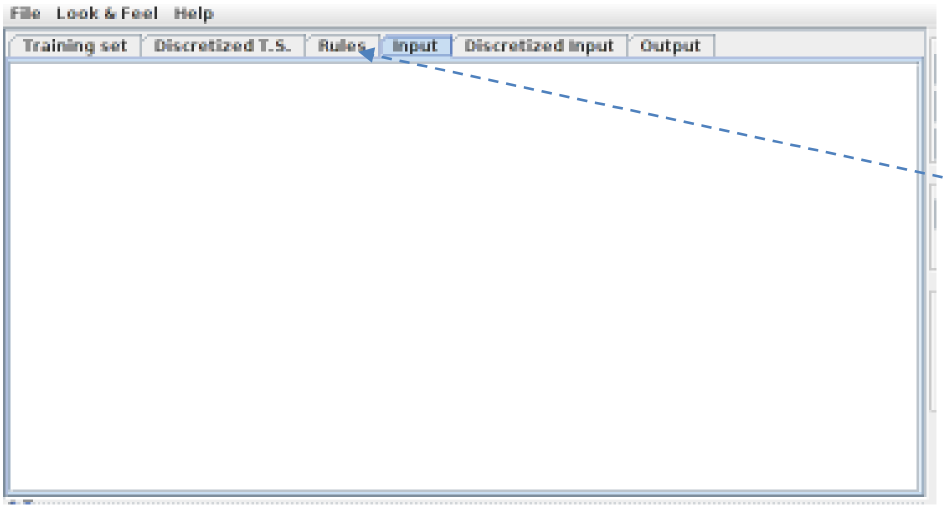
```
export MIDCLASS_JAVA_OPTS='-Xmx1024m'
```

You can also specify 2 gigabytes like this:

```
export MIDCLASS_JAVA_OPTS='-Xmx2g'
```

Starting MIDClass

Open a Terminal window, cd into the MIDClass directory, and run the command: java -jar MIDClass.jar. When running on Linux/Unix OS, make sure that you have rwx permissions for the MIDClass directory and for the directory in which your data is located. Upon running the program, the GUI appears:



The different tabs show the various steps of the algorithm, the rules and output produced.

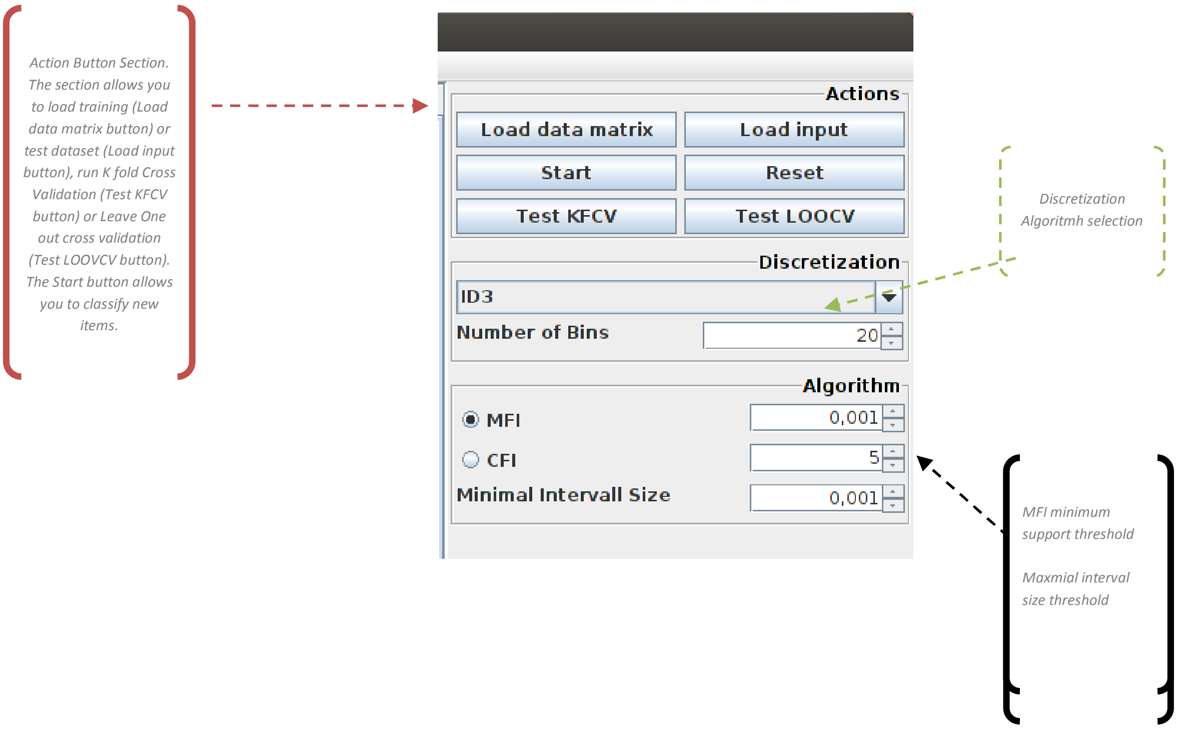
Description of Tabs

- Training Set: The tab shows the data matrix chose.
- Discretized T.S.: It shows the discretized matrix. For each gene, MIDClass discretizes the gene expression building this new matrix.
- Rules: It allows visualizing the discriminant association rules created by the algorithm.
- Input: The user can visualize the elements to be classified.
- Discretized Input: It shows the discretized input matrix.
- Output: In this tab, the program prints all input produced.

MIDClass setting and go fuction

The action buttons on the left of GUI perform all function provided by MIDClass; they are in the following paragraphs. Our algorithm has several parameters that users can set with the buttons present below the action buttons. However, we experimentally tuned the method to establish default values, with those values MIDClass outperformed all the other methods in almost all cases. Users can set:

- Discretization Function: Chose among ID3, CACC, USD, EWIB, EFIB, KMEANS, MC, ENTR, RMDL and SSD (default ID3).
- MFI Threshold: Cutoff for the maximal frequent itemsets (default 0.05),
- Minimum Interval size threshold: Allows to remove gene-intervals whose range is smaller than it (default 0.05).



Action Button Section. The section allows you to load training (Load data matrix button) or test dataset (Load input button), run K fold Cross Validation (Test KFCV button) or Leave One out cross validation (Test LOOVCV button). The Start button allows you to classify new items.

Discretization Algoritmh selection

MFI minimum support threshold

Maxmial interval size threshold

Discretization: ID3

Number of Bins: 20

Algorithm:

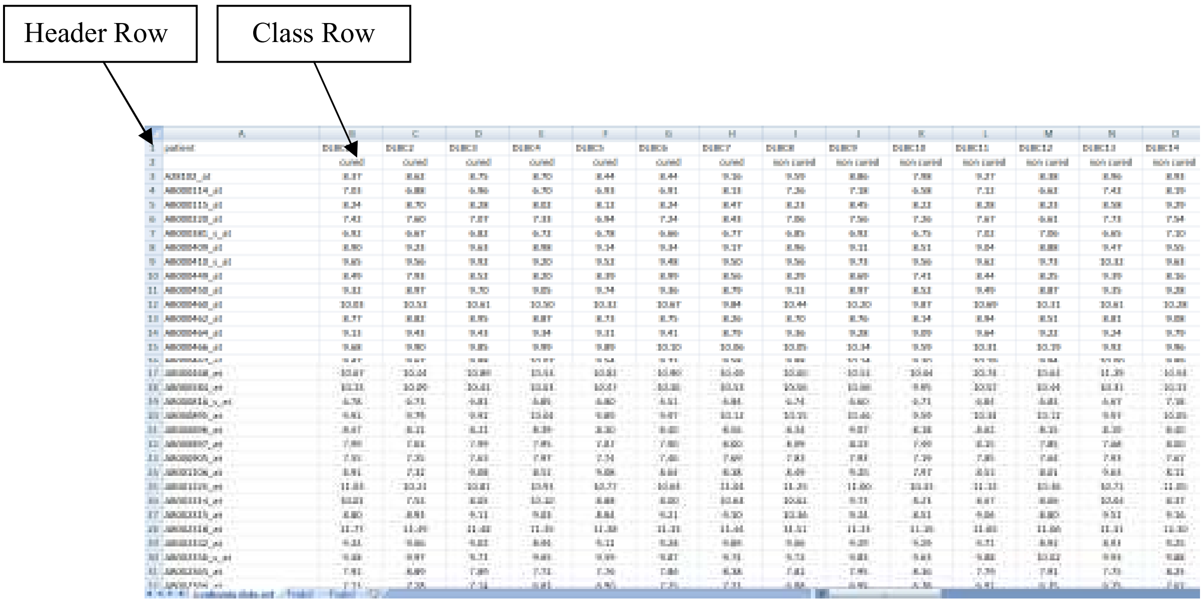
● MFI 0,001

○ CFI 5

Minimal Intervall Size: 0,001

Input Data

The input data files should be tab-delimited ASCII text files. Each row in this file represents a gene/microRNA in the dataset (except for the first two rows which are a header row and the class row), and each column represents a sample. The header row must contains for each sample a unique label as alphanumeric string, the class row must contains for each sample the belonging class as alphanumeric label. Here is an example:

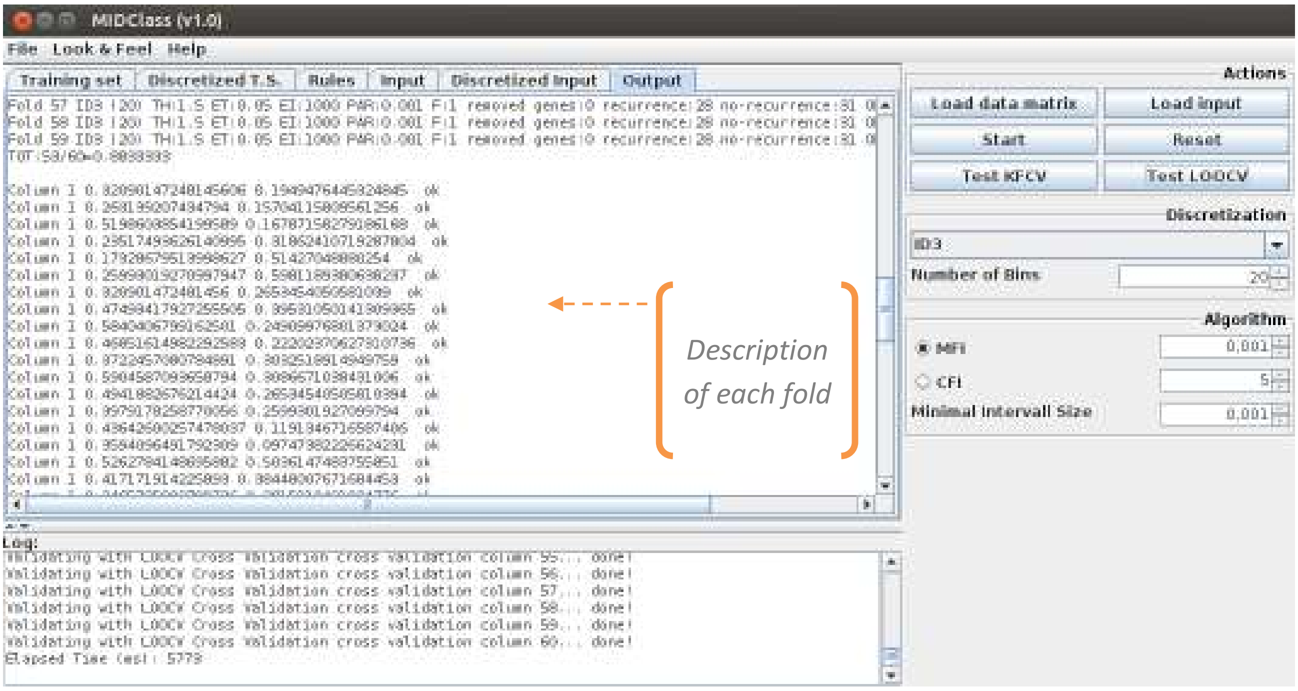


patient	DLBC1	DLBC2	DLBC3	DLBC4	DLBC5	DLBC6	DLBC7	DLBC8	DLBC9	DLBC10	DLBC11	DLBC12	DLBC13	DLBC14
	cured	cured	cured	cured	cured	cured	cured	non cured	non cured	non cured	non cured	non cured	non cured	non cured

To load tabular expression data, select: *File >> load data Matrix* or click the button labelled “Input data matrix”. The dialog box will appear allowing you to select your input file.

Leave One Out and K-Fold Cross Validation

The buttons labeled with “Test LOOCV” and “Test KFCV” provide an estimate of how accurately the classes can be predicted by MIDClass method. The cross-validated misclassification rate is computed and the results are reported. The loocv process omits one sample at a time and in k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. For each cross-validation step, the entire analysis is repeated from scratch: a predictor is constructed and applied to the sample or subsample that was omitted. The program records whether that prediction was correct or not. This is repeated, omitting all of the samples one at a time. The output table shows which samples were correctly and incorrectly predicted, and the overall cross-validated misclassification rate. The output is shown on “Output” tab and is arranged so that you can see the extent to which the different predictors agree or disagree for each specimen left out of the training set.



```
Fold 57 ID3 |20| TH:1.5 ET:0.05 EI:1000 PAR:0.001 F:1 removed genes:0 recurrence:28 no-recurrence:31 0
Fold 58 ID3 |20| TH:1.5 ET:0.05 EI:1000 PAR:0.001 F:1 removed genes:0 recurrence:28 no-recurrence:31 0
Fold 59 ID3 |20| TH:1.5 ET:0.05 EI:1000 PAR:0.001 F:1 removed genes:0 recurrence:28 no-recurrence:31 0
TOT:53/60=0.8833333

Column 1 0.8209014724814566 0.1949476445824845  ok
Column 1 0.2581392074347794 0.15704115808561256  ok
Column 1 0.5198608854198589 0.1678715827918168  ok
Column 1 0.2351749962614095 0.8186241071928780  ok
Column 1 0.1792867951398627 0.5142704888254  ok
Column 1 0.2599301927098794 0.5981189380638287  ok
Column 1 0.820901472481456 0.2653454050581099  ok
Column 1 0.4749841792725505 0.8953105014130985  ok
Column 1 0.5840406799162581 0.2490997688137902  ok
Column 1 0.4685161498229258 0.2220237062791073  ok
Column 1 0.8722457080784881 0.3032510914949758  ok
Column 1 0.5904587093658794 0.3086571038491006  ok
Column 1 0.4941882676214424 0.2653454050581039  ok
Column 1 0.3979178258770056 0.2599301927089794  ok
Column 1 0.4364260025747803 0.1191346716587405  ok
Column 1 0.3594096491792309 0.0974738222662428  ok
Column 1 0.5262784148695882 0.5036147488755851  ok
Column 1 0.4171719142258593 0.3844800767168445  ok
```

Discretization: ID3

Number of Bins: 20

Algorithm:
● MFI 0,001
○ CFI 5

Minimal Intervall Size: 0,001

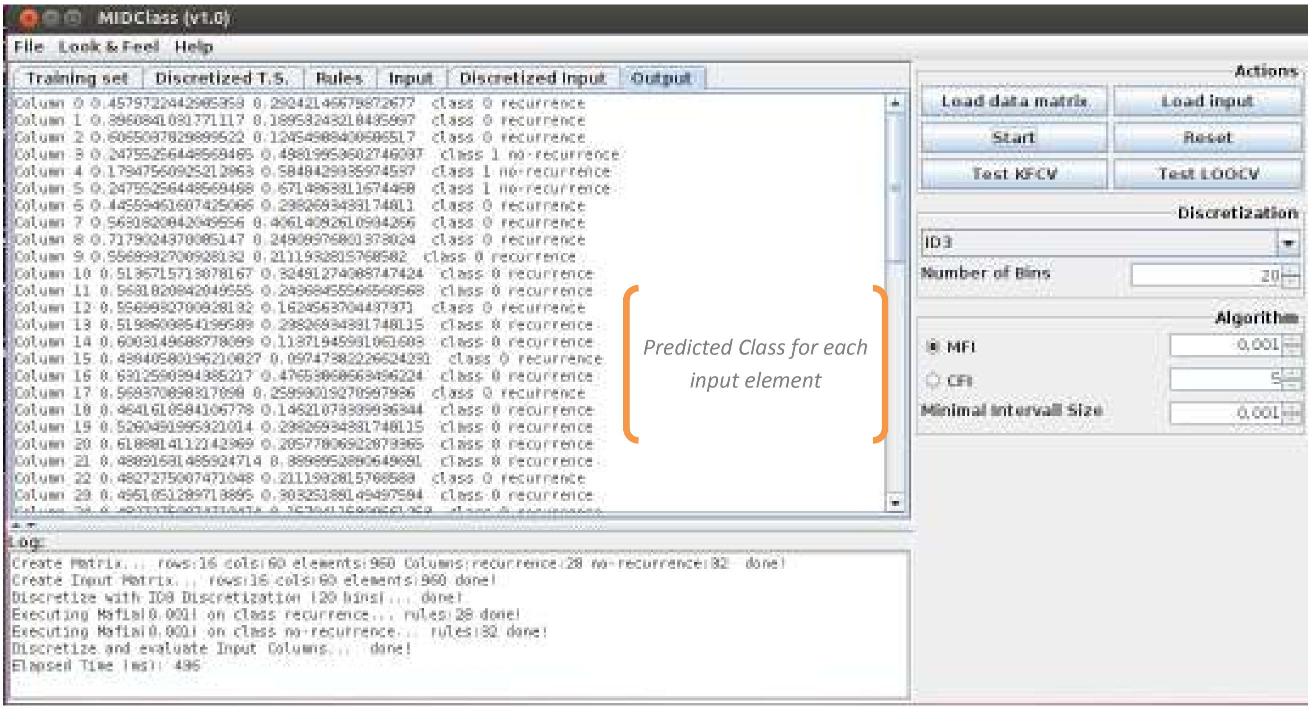
Log:

```
Validating with LOOCV Cross Validation cross validation column 55... done!
Validating with LOOCV Cross Validation cross validation column 56... done!
Validating with LOOCV Cross Validation cross validation column 57... done!
Validating with LOOCV Cross Validation cross validation column 58... done!
Validating with LOOCV Cross Validation cross validation column 59... done!
Validating with LOOCV Cross Validation cross validation column 60... done!
Elapsed Time (ms): 5779
```

Prediction for new samples

Usually, you do not have a separate set of samples that you wish to withhold from the model building process when the analysis is first done. Sometimes, however, after the model is built, you have additional samples whose classes you wish to predict using the previously developed model. We have enabled you to do this by re-building the model using the initial set of samples, and to then predict for the new samples in a combined analysis.

To classify samples not used in the model building process. The input data files should be tab-delimited ASCII text files. Each row in this file represents a gene/microRNA in the dataset (except for the first row which is a header row) and each column represents a sample. To load tabular expression data, select: File >> Load Input or click the button labeled “Load Input”. The dialog box will appear allowing you to select your input file.



```
Column 0 0.4579722442985359 0.29242146679872677  class 0 recurrence
Column 1 0.3960841031771117 0.18958324321849597  class 0 recurrence
Column 2 0.6065097829899522 0.12454989400986517  class 0 recurrence
Column 3 0.24755256448569465 0.49819953602746087  class 1 no-recurrence
Column 4 0.17947560925212863 0.5848429935974597  class 1 no-recurrence
Column 5 0.24755256448569468 0.6714869311674468  class 1 no-recurrence
Column 6 0.44559461607425066 0.2982693493174811  class 0 recurrence
Column 7 0.5631820842049556 0.40614092610984266  class 0 recurrence
Column 8 0.7179024970085147 0.24909976801373024  class 0 recurrence
Column 9 0.5569992700928132 0.2111992815768582  class 0 recurrence
Column 10 0.5136715713078167 0.32491274088747424  class 0 recurrence
Column 11 0.5681820842049555 0.24369455596560568  class 0 recurrence
Column 12 0.5569992700928132 0.1624563704437971  class 0 recurrence
Column 13 0.5198600854199589 0.29826934931748115  class 0 recurrence
Column 14 0.6003149688778099 0.11371945981061608  class 0 recurrence
Column 15 0.43940580196210827 0.09747382226624281  class 0 recurrence
Column 16 0.6312590394385217 0.47653960863496224  class 0 recurrence
Column 17 0.569370898317098 0.25999019270997936  class 0 recurrence
Column 18 0.4641610584106778 0.14621073939936344  class 0 recurrence
Column 19 0.5260491995921014 0.29826934931748115  class 0 recurrence
Column 20 0.6188814112142969 0.20577806922879365  class 0 recurrence
Column 21 0.48891681485924714 0.38998952880649681  class 0 recurrence
Column 22 0.4827275007471048 0.2111992815768588  class 0 recurrence
Column 23 0.4951051289718895 0.30325188149497594  class 0 recurrence
```

Predicted Class for each input element

Discretization: ID3
Number of Bins: 20

Algorithm
- ● MFI 0.001
- ○ CFI 5

Minimal Intervall Size: 0.001

```
Create Matrix... rows:16 cols:60 elements:960 Columns:recurrence:28 no-recurrence:32  done!
Create Input Matrix... rows:16 cols:60 elements:960 done!
Discretize with ID3 Discretization (20 bins)... done!
Executing Mafia(0.001) on class recurrence... rules:28 done!
Executing Mafia(0.001) on class no-recurrence... rules:32 done!
Discretize and evaluate Input Columns...  done!
Elapsed Time (ms): 496
```

Then click the button labeled “Start” and you will find the results in the “Output” tab.