

“TreeLiker – Graphical User Interface”



TREELIKER

USER MANUAL

Intelligent Data Analysis Research Lab

Department of Cybernetics

Faculty of Electrical Engineering

Czech Technical University in Prague (CTU)

June 2012

CONTENTS

1	Introduction.....	4
2	Detailed Functional Description	4
2.1	Input Module.....	5
2.2	Template Module	5
2.3	Pattern search Module.....	5
2.4	Found Patterns Module.....	5
2.5	Training Module	5
3	Installation.....	5
3.1	Installation Guide	5
3.2	Running the Machine learning tool.....	6
4	Graphic User Interface.....	6
4.1	Menu Bar	6
4.1.1	File	6
4.1.1.1	New Project.....	7
4.1.1.2	Load Project.....	7
4.1.1.3	Save Project.....	8
4.1.1.4	Exit.....	9
4.1.2	Help.....	9
4.2	Input Module.....	9
4.2.1	Welcome Window	9
4.2.2	Input Tab.....	11
4.2.2.1	Add new dataset.....	12
4.2.2.2	Edit dataset.....	12
4.2.2.3	Delete dataset.....	12
4.3	Template Module	13
4.4	Pattern Search Module.....	14
4.4.1.1	Search for Relational Patterns.....	15
4.4.2	Found Patterns Module.....	17
4.5	Training Module	19
4.5.1	Start Training	19

1 INTRODUCTION

TreeLiker-GUI is a simple application providing access to fast algorithms for work with complex structured data in relational form. The data can, for example, describe large organic molecules such as proteins or groups of individuals such as social networks or predator-prey networks etc. The algorithms included in TreeLiker-GUI are unique in that, in principle, they are able to search a given set of relational patterns exhaustively – thus guaranteeing that if some good pattern capturing an important feature of the problem exists, it will be found. In experiments with real-life data, the algorithms were shown to be able to construct complete non-redundant sets of patterns for chemical datasets involving several thousand molecules as well as for datasets from genomics or proteomics.

The included relational learning algorithms are tailored towards so-called tree-like features for which some otherwise very hard sub-problems (NP-hard) become tractable. The problem of finding a complete set of informative features remains hard also for tree-like features, however, we were able to develop algorithms for tree-like features which scale well for problems of real-life scale. Currently, the machine learning algorithms integrated in TreeLiker-GUI include implementations of relational learning algorithms HiFi and RelF and Poly in an intuitive GUI.

The three algorithms were described in the following papers:

RelF: Ondřej Kuželka and Filip Železný. Block-Wise Construction of Tree-like Relational Features with Monotone Reducibility and Redundancy. *Machine Learning*, 83, 2011

Poly: Ondřej Kuželka, Andrea Szabóová, Matěj Holec and Filip Železný. Gaussian Logic for Predictive Classification. *ECML/PKDD 2011: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases* (this paper described a restricted version of Poly)

HiFi: Ondřej Kuželka and Filip Železný. HiFi: Tractable Propositionalization through Hierarchical Feature Construction. *Late Breaking Papers, the 18th International Conference on Inductive Logic Programming*, 2008

TreeLiker-GUI uses WEKA.

WEKA: Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, Ian H. Witten (2009); The WEKA Data Mining Software: An Update; *SIGKDD Explorations, Volume 11, Issue 1*.

The panel-based philosophy of TreeLiker-GUI is also inspired by WEKA.

2 DETAILED FUNCTIONAL DESCRIPTION

The application is composed of six main modules: the input module, the template module, the pattern search module, the found patterns module and the training module. The structure and descending order of the different modules are the following:

2.1 INPUT MODULE

This module allows the user to select the dataset directories or the specific files that should be used as input data. The user can add as many datasets as desired.

2.2 TEMPLATE MODULE

The template module permits the user to introduce the template specifying the language bias that should be used in the execution of the algorithms of the application.

2.3 PATTERN SEARCH MODULE

The pattern search module enables the user to construct relational patterns for the datasets selected in the Input Module. The language bias is taken from the Template Module

2.4 FOUND PATTERNS MODULE

This module uses the results provided by the pattern search module. It shows the structural patterns that were found.

2.5 TRAINING MODULE

This module allows the user to train a classifier based on the patterns generated in the Pattern Search Module. The available classifiers are Zero Rule, SVM with Radial Basis Kernel, J48 Decision Tree, One Rule, Ada-boost, Simple Logistic Regression, Random Forest, L2-Regularized Logistic Regression and Linear SVM. The results of training are shown in the result list, where the user can choose one of them and display it.

3 INSTALLATION

3.1 INSTALLATION GUIDE

To be able to run the Machine Learning Tool, follow the next steps:

- 1) Download and install the Java SE Runtime Environment (JRE). The version needed is the **JRE6**.

The latest version of JRE can be downloaded from the Java website:

<http://www.oracle.com/technetwork/java/javase/downloads/>

3.2 RUNNING THE MACHINE LEARNING TOOL

To run the Machine Learning Tool just run the file start.bat (or start.sh if you are working on Linux) and the program will start automatically.

Figure #1: Welcome Window



4 GRAPHIC USER INTERFACE

4.1 MENU BAR

Provides access to file options and also provides the help option of the application.

Figure #2: Menu Bar



4.1.1 FILE

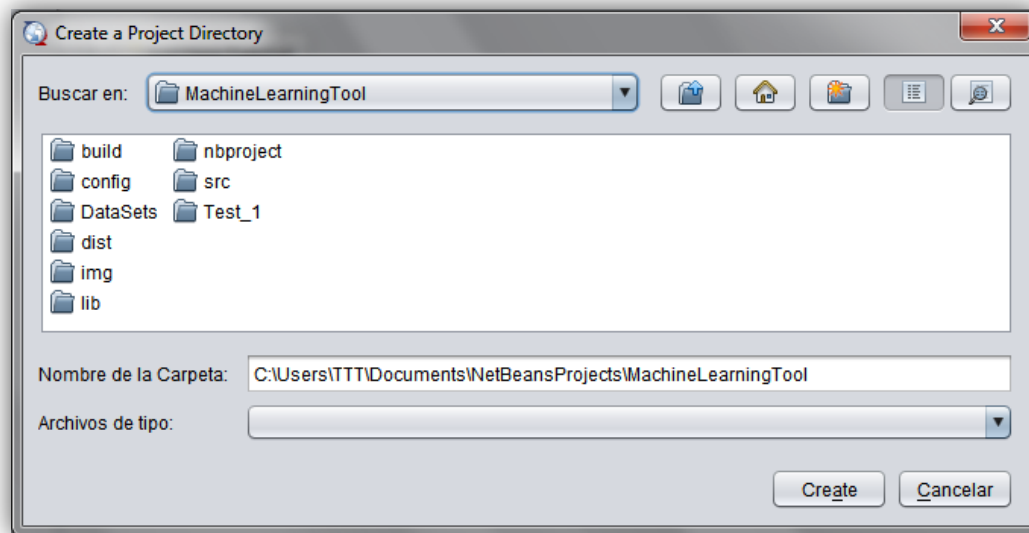
It has the following options:

4.1.1.1 NEW PROJECT

You can create a new TreeLiker project; in this case, it creates a new directory.

1. On the menu bar, click the **File** option.
2. Click the **New Project** option.
3. On the **Create a Project Directory** window, select the desired project location.

Figure #3: Create a Project Directory Window



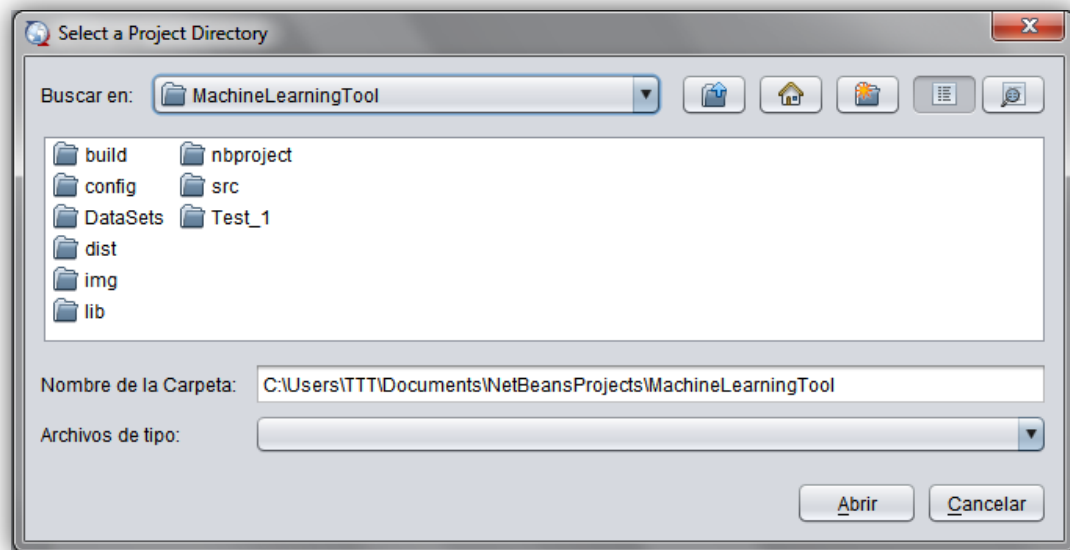
4. Type the name of the project.
5. If you want to continue, click **Create**. If you do not want to, click **Cancel** or close the window.

4.1.1.2 LOAD PROJECT

You can load an existing Machine Learning Tool project.

1. On the menu bar, click the **File** option.
2. Click the **Load Project** option.
3. On the **Select a Project Directory** window, search the project location.

Figure #4: Select a Project Directory Window



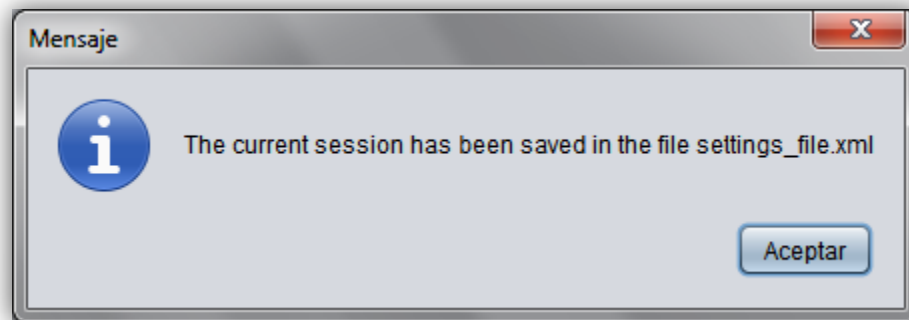
4. Select a project by clicking on the file or directory.
5. If you want to continue with the loading, click **Open**. If you do not want to, click **Cancel** or close the window.

4.1.1.3 SAVE PROJECT

You can save the changes done in the current Machine Learning Tool project.

1. On the menu bar, click the **File** option.
2. Click the **Save Project** option.
3. Accept on the message window that appears, or just close it.

Figure #5: Save Project Message

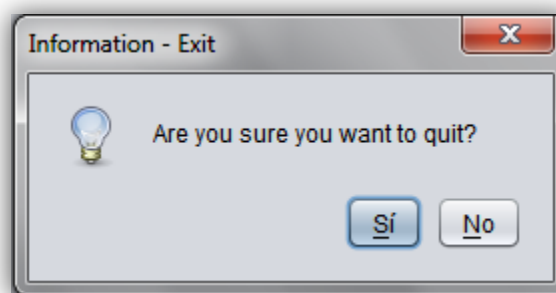


4.1.1.4 EXIT

Exit the Machine Learning Tool application.

1. On the menu bar, click the **File** option.
2. Click the **Exit** option.
3. If you want to continue with the exit, accept when the **Information – Exit** message window appears. If you do not want to, deny it or close the window.

Figure #6: Information - Exit Message



4.1.2 HELP

It has the following option:

4.2 INPUT MODULE

You can enter Machine Learning Tool application and indicate the dataset directories or files.

4.2.1 WELCOME WINDOW

You can enter the Machine Learning Tool application, by loading an existing project or creating a new one.

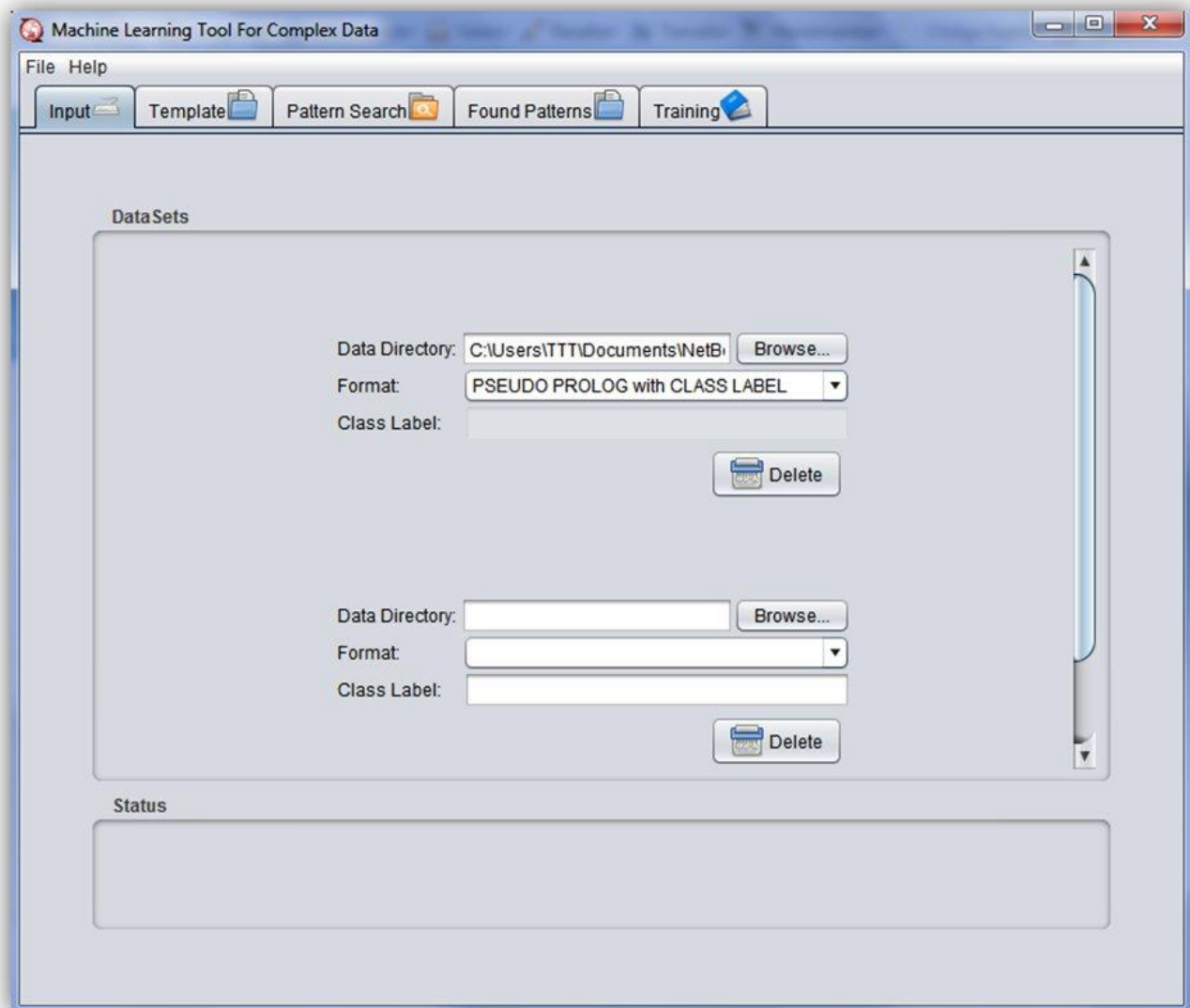
Figure #8: Welcome Window



4.2.2 INPUT TAB

You can select the datasets that are going to be used by the other Modules.

Figure #11: Input Tab



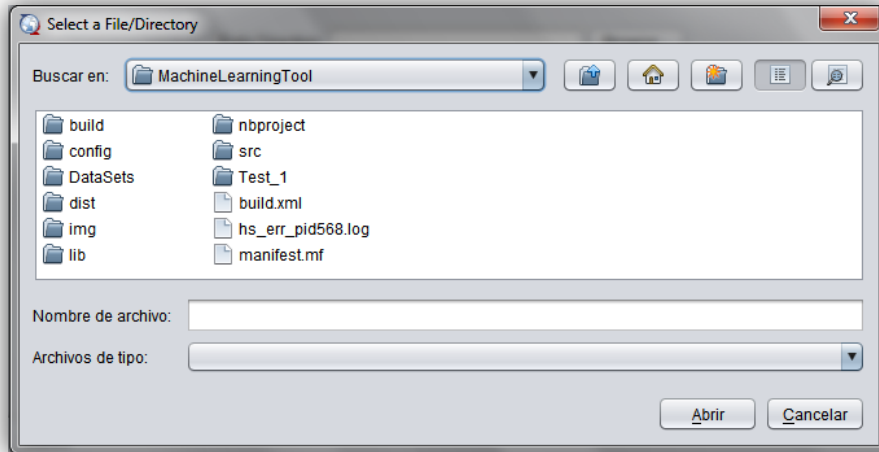
Note: There are some restrictions on this tab. An error occurs if you do not fulfill them, and it is going to appear as an error message in the status panel. They are the following:

1. The data directory/file must be selected for every dataset.
2. The format of all datasets must be selected.
3. The class-labels of all datasets must be set if the selected format does not contain information about class-labels of the individual examples).
4. The selected dataset cannot be empty.
5. The format of the selected dataset must be correct.

4.2.2.1 ADD NEW DATASET

1. On the **Input** tab, click the **Add New Dataset** button.
2. Click the **Browse...** button to search for the **Dataset's Directory**.
3. On the **Select a Directory** window, select the dataset directory location.

Figure #12: Select a Project Directory window



4. Select a dataset by clicking on the directory.
5. If you want to continue with the selection, click **Open**. If you do not want to, click **Cancel** or close the window.
6. Select the **Format** of the dataset
7. Write the **Class Label** for the dataset selected.

4.2.2.2 EDIT DATASET

1. On the **Input** tab, locate the **dataset** you want to edit.
2. You can edit the following parameters:
 - a. Dataset's Directory:
 - i. Follow the step 2 to 5 indicated in the [Add New DataSet](#) section.
 - b. Format:
 - i. Select the new **Format** for the dataset selected.
 - c. Class Label:
 - i. Write the new **Class Label** for the dataset selected.

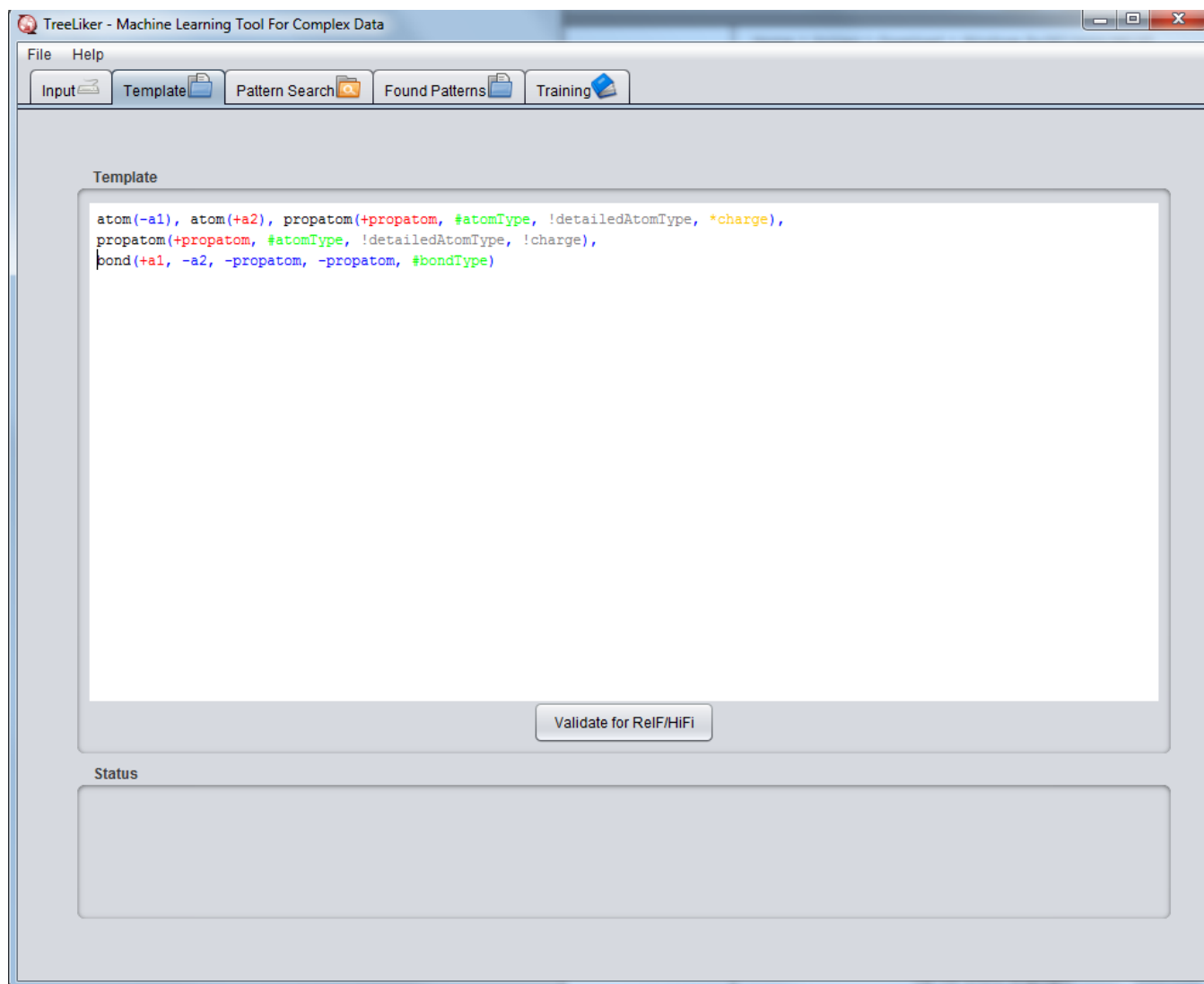
4.2.2.3 DELETE DATASET

1. On the **Input** tab, locate the **dataset** you want to delete.
2. Click the **Delete** button.

4.3 TEMPLATE MODULE

Allows the user to enter a template and validates the correctness of it. See the document describing the language bias specification using templates for a description of the template-based language bias

Figure #13: Template Tab



Note: There are some restrictions on this tab. An error occurs if you do not fulfill them, and it is going to appear an error message in the status panel. They are the following:

1. If the template text is empty.
2. Specifies if there are syntax errors in the given template.
3. Indicates if there's not an input variable for a specific output variable.
4. Indicates if there are cycles in the template.

Note: The components of each literal of a template can be distinguished by their color:

- i. Output variables (Blue)
- ii. Input variables (Red)
- iii. Constants (Green)
- iv. Ignored variables (Gray)
- v. Aggregation variables (Orange)

4.4 PATTERN SEARCH MODULE

The pattern search module enables the user to construct relational patterns for the datasets selected in the Input Module. The language bias is taken from the Template Module

Figure #15: Pattern Search Tab – ReLF settings

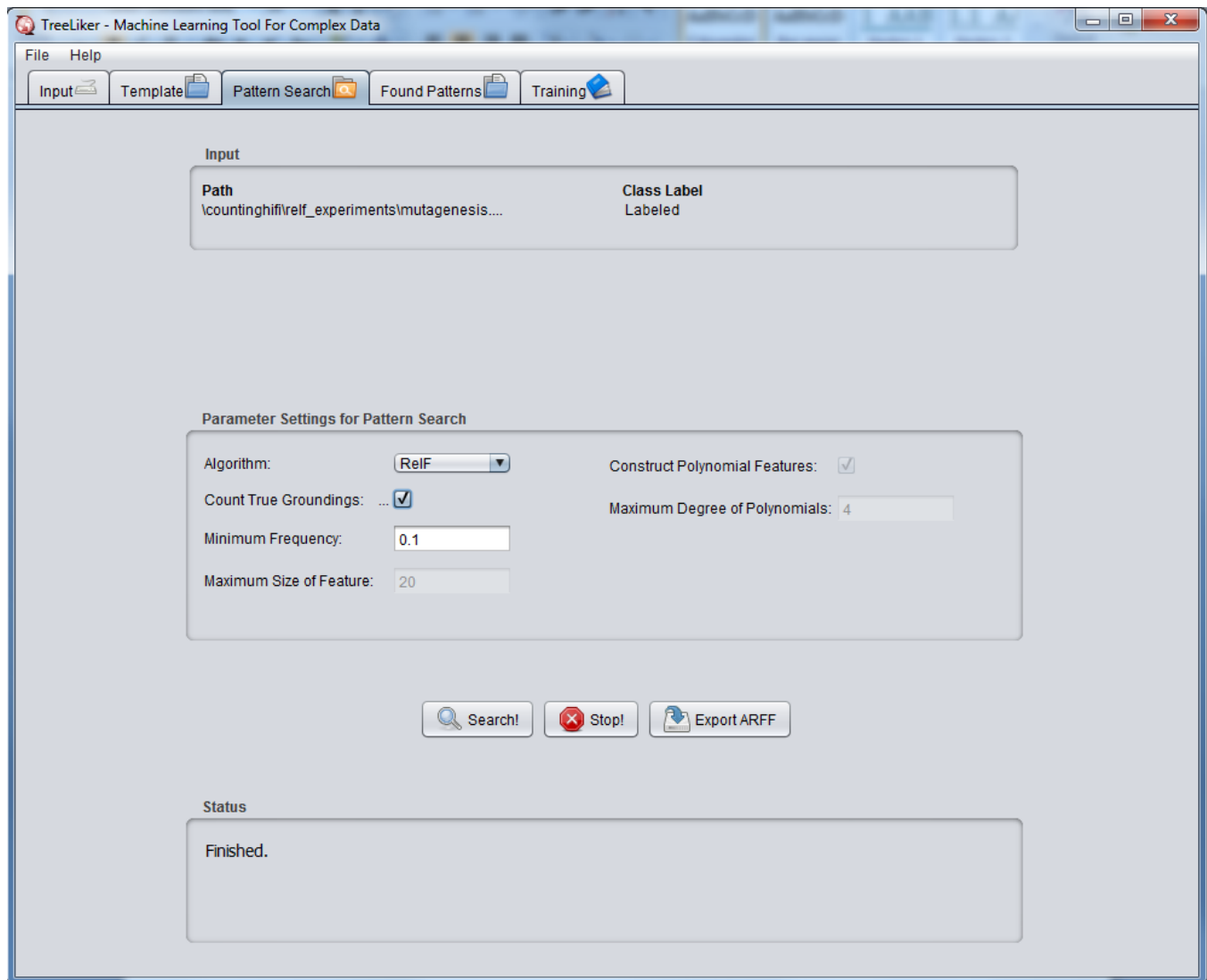
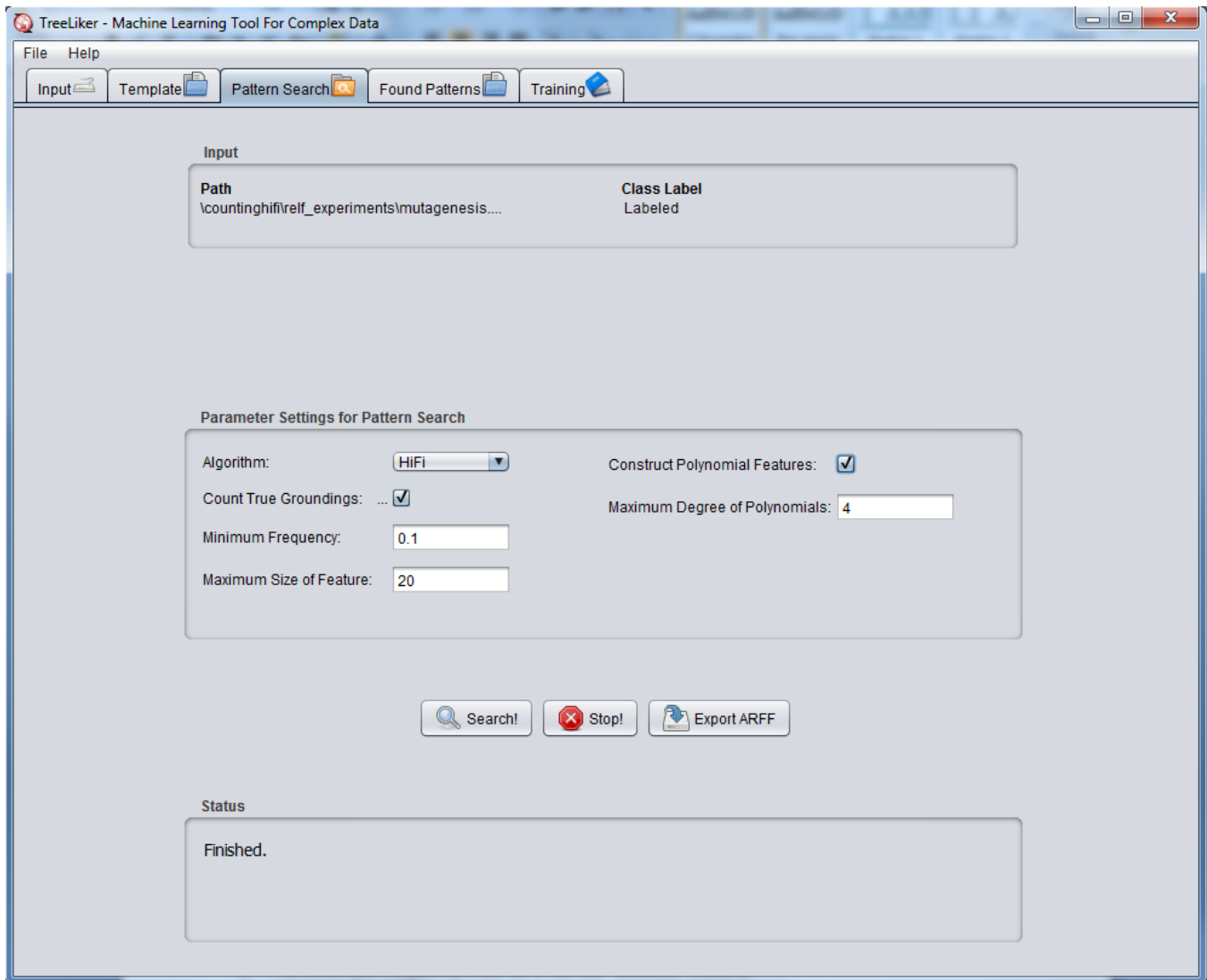


Figure #16: Pattern Search Tab – HiFi settings



Note: There are some restrictions on this tab. An error occurs if you do not fulfill them, and it is going to appear an error message in the status panel. They are the following:

1. Select datasets to start the Pattern Search.
2. Establish a valid template.
3. The minimum-frequency must be a number between 0 and 1.
4. All the parameters have to be a number greater than or equal to 0.

4.4.1.1 SEARCH FOR RELATIONAL PATTERNS

You can start the Pattern Search selecting the desired algorithm to run, followed by the necessary parameters according to the selected algorithm.

Note: If you try to run the Pattern Search, but all the parameters are the same as the last time you ran it, there is going to appear a message in the status panel indicating that the pattern search has already been done.

If the selected algorithm corresponds to RelF you have to determine the following parameters:

1. **Minimum Frequency** (a number from 0 to 1)
2. Check the **Count True Groundings** checkbox if you want to use it, uncheck it if you do not want to.

If the selected algorithm corresponds to RelF/HiFi you have to determine the following parameters:

1. **Minimum Frequency** (a number from 0 to 1)
2. **Maximum size of features** (a positive integer)
3. Check the **Count True Groundings** checkbox if you want to use it, uncheck it if you do not want to.
4. Check the **Construct Polynomial Features** checkbox if you want HiFi to construct multivariate polynomial aggregation features.
5. Select the maximum degree of the multivariate polynomial aggregation features.

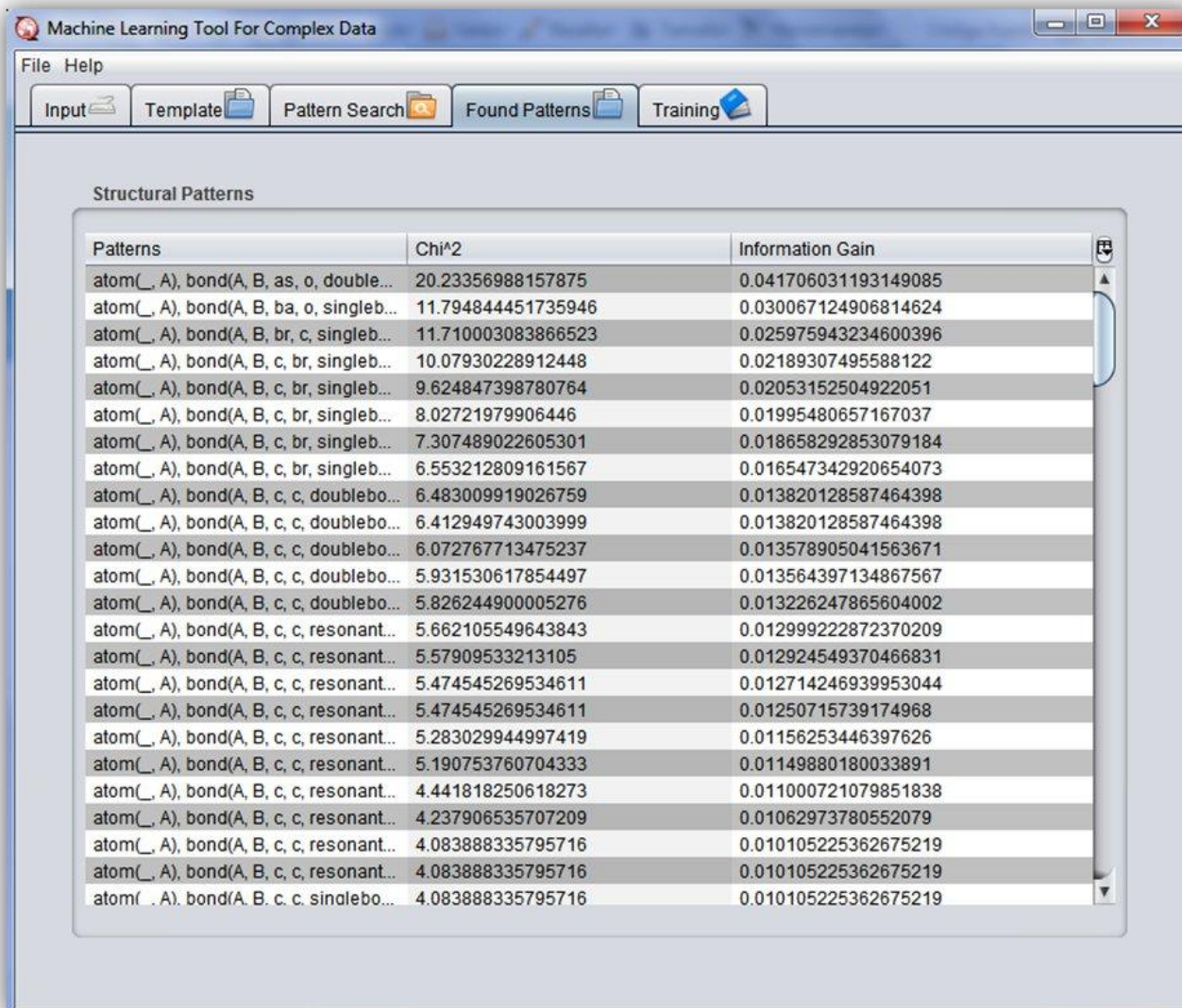
Click the **Search!** button to start the Pattern Search.

Note: The information about what the Pattern Search algorithm is doing is going to appear in the status panel, as well as error messages if something wrong happens.

4.4.2 FOUND PATTERNS MODULE

Shows the relational patterns found with the pattern search algorithm.

Figure #17: Found Structural Patterns Tab.



Patterns	Chi^2	Information Gain
atom(, A), bond(A, B, as, o, double...	20.23356988157875	0.041706031193149085
atom(, A), bond(A, B, ba, o, singleb...	11.794844451735946	0.030067124906814624
atom(, A), bond(A, B, br, c, singleb...	11.710003083866523	0.025975943234600396
atom(, A), bond(A, B, c, br, singleb...	10.07930228912448	0.02189307495588122
atom(, A), bond(A, B, c, br, singleb...	9.624847398780764	0.02053152504922051
atom(, A), bond(A, B, c, br, singleb...	8.02721979906446	0.01995480657167037
atom(, A), bond(A, B, c, br, singleb...	7.307489022605301	0.018658292853079184
atom(, A), bond(A, B, c, br, singleb...	6.553212809161567	0.016547342920654073
atom(, A), bond(A, B, c, c, doublebo...	6.483009919026759	0.013820128587464398
atom(, A), bond(A, B, c, c, doublebo...	6.412949743003999	0.013820128587464398
atom(, A), bond(A, B, c, c, doublebo...	6.072767713475237	0.013578905041563671
atom(, A), bond(A, B, c, c, doublebo...	5.931530617854497	0.013564397134867567
atom(, A), bond(A, B, c, c, doublebo...	5.826244900005276	0.013226247865604002
atom(, A), bond(A, B, c, c, resonant...	5.662105549643843	0.012999222872370209
atom(, A), bond(A, B, c, c, resonant...	5.57909533213105	0.012924549370466831
atom(, A), bond(A, B, c, c, resonant...	5.474545269534611	0.012714246939953044
atom(, A), bond(A, B, c, c, resonant...	5.474545269534611	0.01250715739174968
atom(, A), bond(A, B, c, c, resonant...	5.283029944997419	0.01156253446397626
atom(, A), bond(A, B, c, c, resonant...	5.190753760704333	0.01149880180033891
atom(, A), bond(A, B, c, c, resonant...	4.441818250618273	0.011000721079851838
atom(, A), bond(A, B, c, c, resonant...	4.237906535707209	0.01062973780552079
atom(, A), bond(A, B, c, c, resonant...	4.083888335795716	0.010105225362675219
atom(, A), bond(A, B, c, c, resonant...	4.083888335795716	0.010105225362675219
atom(, A), bond(A, B, c, c, singlebo...	4.083888335795716	0.010105225362675219

Note: If you want to order the results according to the Patterns, the Chi Square or Information Gain, you can click on the titles of each column (Patterns, Chi Square or Information Gain).

Additionally, the user interface provides a button (Top Right) with options to hide columns or to adjust each column to the size of the content.

Figure #18: Found Structural Patterns Tab with option panel.

Machine Learning Tool For Complex Data

File Help

Input Template Pattern Search Found Patterns Training Prediction

Structural Patterns

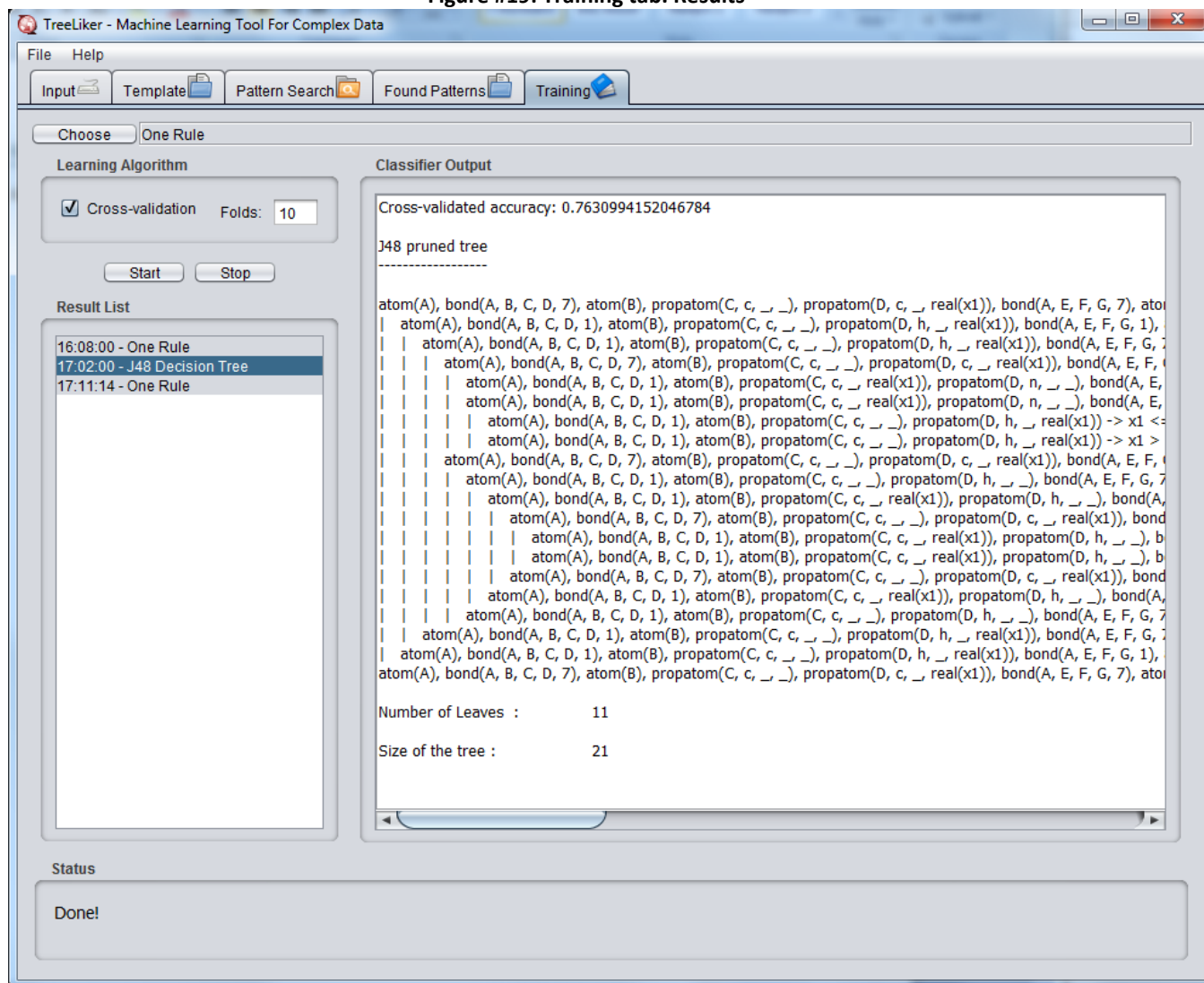
Patterns	Chi ²	Information Gain
atom(, A), bond(A, B, c, c, resonant...	20.23356988157875	
atom(, A), bond(A, B, c, c, doubleb...	11.794844451735946	
atom(, A), bond(A, B, c, c, doubleb...	11.710003083866523	
atom(, A), bond(A, B, c, c, doubleb...	10.07930228912448	
atom(, A), bond(A, B, c, c, doubleb...	9.624847398780764	
atom(, A), bond(A, B, c, c, doubleb...	8.02721979906446	
atom(, A), bond(A, B, c, br, singleb...	7.307489022605301	
atom(, A), bond(A, B, c, br, singleb...	6.553212809161567	0.016547342920654073
atom(, A), bond(A, B, c, br, singleb...	6.483009919026759	0.013820128587464398
atom(, A), bond(A, B, c, br, singleb...	6.412949743003999	0.013820128587464398
atom(, A), bond(A, B, c, br, singleb...	6.072767713475237	0.013578905041563671
atom(, A), bond(A, B, c, br, singleb...	5.931530617854497	0.013564397134867567
atom(, A), bond(A, B, c, br, singleb...	5.826244900005276	0.013226247865604002
atom(, A), bond(A, B, c, br, singleb...	5.662105549643843	0.012999222872370209
atom(, A), bond(A, B, c, br, singleb...	5.57909533213105	0.012924549370466831
atom(, A), bond(A, B, c, br, singleb...	5.474545269534611	0.012714246939953044
atom(, A), bond(A, B, c, br, singleb...	5.474545269534611	0.01250715739174968
atom(, A), bond(A, B, c, br, singleb...	5.283029944997419	0.01156253446397626
atom(, A), bond(A, B, c, br, singleb...	5.190753760704333	0.01149880180033891
atom(, A), bond(A, B, c, br, singleb...	4.441818250618273	0.011000721079851838
atom(, A), bond(A, B, c, br, singleb...	4.237906535707209	0.01062973780552079

☒ Patterns
☒ Chi²
☒ Information Gain
☐ Desplazamiento Horizontal
☐ Compactar Todas las Columnas
☐ Compactar la Columna Seleccionada

4.5 TRAINING MODULE

Allows the user to train classifiers and evaluate them using cross-validation.

Figure #19: Training tab. Results



Note: There are some restrictions on this tab. An error occurs if you do not fulfill them, and it is going to appear an error message in the status panel. They are the following:

1. Always select a classifier.
2. If Cross-validation is selected, indicate the number of folds.
3. The number of folds has to be an integer number greater than 1.
4. Can't have more folds than instances.

4.5.1 START TRAINING

1. Click the **Training** tab.
2. Check the **Choose** button to select a classifier.
3. You can choose to use cross-validation or not.
 - a. If you want to use **Cross-validation**:
 - i. Check the **Cross-validation** checkbox to enable the **Folds** text field.
 - ii. Write the number of folds in the **Folds** text field.
 - b. If you do not want to use **Cross-validation**:
 - i. Uncheck the **Cross-validation** checkbox.
4. Click the **Start** button; the results are going to appear in the **Classifier Output** text area.

Note: if you want to visualize results for a different item on the result list, just select the name by clicking on it or using the up and down keys.
5. If you want to delete one of the searches you can select it on the result list and press the back-space key to erase it from the list.