Description of EGPC (v1.0)

EGPC is a multi-class classifier based on genetic programming and majority voting. The main features of EGPC are that:

- It runs in command line interface (CLI) and graphical user interface (GUI) modes;
- It can be used for binary and multi-class classification;
- It can handle microarray gene expression data as well as UCI machine learning (ML) databases (with no missing values);
- It can handle numeric, nominal and Boolean (converted to numbers) features;
- It can evolve rules with arithmetic and/or logical functions;
- It can handle training subsets constructed by fixed or random split of data; and
- It can handle training and validation data stored in two separate files provided that they have the same number of features and the features are in same order.

1. Data format:

EGPC can handle two formats of (tab, colon, space, and semicolon delimited) text files: Microarray and UCI ML data files. In Microarray format, the first row contains the labels of the samples and the other rows contain the gene expression values of different genes in different samples. One file: BreastCancer.txt in the examples is in this format.

In UCI ML format, the first column contains the labels of the samples and the other columns contain the values of features in different samples/instances. Two files: Monk.txt and WCBreast.txt in the examples are in this format.

In either form, the labels of samples must be numeric values starting from 0, i.e. for binary classification, the labels of samples will be either 0 or 1 while for multi-class classification, the labels will be from 0 to (*c-1)* where *c* is the number of classes of samples in the data file.

2. Training and test subsets constructions:

The training and the test data can be in one file or in two separate files. If the data are in two separate files, the split of whole data is fixed. While the data are in one file, the training and test subsets can be constructed randomly or by fixed split of the data. In the case of random split, the information about training subset is entered by combining the training sizes of different classes, delimited by colon (:), into a single string. Let us give an example of this. Suppose a data set consists of three classes of samples, and the number of samples of the classes is 8, 8, and 6, respectively. If the training subset consists of 50% of the samples, the string containing information about the training data would be 4:4:3. However, in the case of the fixed split of the samples stored in one file, a file containing the indexes of the training samples should be passed to the program. See *BreastTrainIndex.txt* for an example of the fixed split of *BreastCancer.txt* data. Note that the index of the first sample is 1.

3. Features/Attributes/Genes:

EGPC can handle numeric, nominal and Boolean features. If any feature is in nominal format, convert it to numeric format. For example, if the values of a nominal feature is *Sunny, Cloudy, Rainy, and Snowing*, convert these values to 0, 1, 2, and 3, respectively. The identifiers for these features are as follows:
N: Numeric; L: Nominal; and B: Boolean. If a range of the features are of same type, they can be indicated by <start index>:<end index>. For example, if the first 30 features of a data set are numeric, indicate it by *N1:30*. For nominal features, there is one extra parameter, the maximum value of the nominal feature, which follows the <end index> separated by colon. For example, L1:30:4. For multi-type features, indicate each type and range by the above notations and separate each other by colon. An example may be *N1:30:L31:50:4:B51:60.*

Each feature in the data file is represented by an `X' followed by the feature number. For example, X1314 represents the feature 1314 of a data set.

4. Functions:

EGPC can handle the following functions :{ +,-,*, /, SQR, SQRT, LN, EXP, SIN, COS, AND, OR, NOT, =, <>, >, <, >=, <=}. If all the features are numeric, one can use all the above functions, only arithmetic functions: {+,-,*, /, SQR, SQRT, LN, EXP, SIN, COS}, or only logical and Boolean functions: {AND, OR, NOT, =, <>, >, <, >=, <=}. Note here that since all the features are numeric, only {AND, OR, NOT} can NOT be used.
If all the features are either nominal or Boolean, we recommend using logical and Boolean functions only. However, EGPC would be able to handle the combinations as nominal features are converted to numeric values (by the user) before running the software.
Protected executions of some functions:

- *SQRT (Y)=0* if *Y* is negative;
- *Y/Z=1* if *Z* is 0;
- *EXP(Y)=EXP(Max(-10000, Min(Y,20))*; *Y* is bounded in [-10000,20];
- *LN(Y)=0* if *Y=0*; otherwise *LN(Y)=LN(|Y|)* and *Y* is bounded in [-1.0E100, 1.0E100];
- *SIN (Y)* and *COS(Y)*: *Y* is bounded in [-10000, 10000].

5. Ensemble size:

The number of single rules or sets of rules that participate in majority voting is indicated here as ensemble size. The minimum ensemble size is 3. For binary classification, we recommend an odd value for ensemble size.

6. Evolved rules (S-expressions):

The expression of a rule is called S-expression. The predicted class of a rule is determined depending on the output of the rule:

Boolean output (an S-expression consists of logical or logical+arithmetic functions):
- Binary classification:
  - IF (S-expression) THEN 0 ELSE 1.
- Multi-class classification:
  - IF (S-expression) THEN *TargetClass* ELSE *Other*.

Real-valued output (an S-expression consists of arithmetic functions only):
- Binary classification:
  - IF (S-expression>=0) THEN 0 ELSE 1.
- Multi-class classification:
  - IF (S-expression>=0) THEN *TargetClass* ELSE *Other*.

Therefore, the slice point for real-valued output is 0.

7. Default settings of different genetic programming (GP) parameters:

The default settings of some GP parameters in EGPC are as follows:
Maximum number of nodes in a GP tree=100;
Maximum initial depth=6;
Initial population generation method: ramped half-and-half
Maximum crossover depth=7;
Selection method for crossover: greedy-over;
Reproduction probability=0.1;
Crossover probability=0.9;
Mutation probability=0.1;
Termination criteria: fitness=1.0 or maximum number of generations has passed;
Regeneration type: elitism (elite size=1)

8. Output files:

EGPC classification software will create three files containing rules, accuracy and gene frequency information in the working directory (from where the software is run). If the name of a data file is *Example.txt*, EGPC software will create three files named as *RulesExample.txt*, *AccuracyExample.txt*, and *GeneFreqExample.txt*. However, EGPC preprocessing software will create two files containing preprocessed data (default: *DataOut.txt*) and cross-reference indexes (*CrossRefIdx.txt*).

9. Four example files included with this software bundle:

Monk's problem (Monk.txt): It has been downloaded from http://www.ics.uci.edu/~mlearn/MLRepository.html. It is a binary classification problem, and consists of 6 nominal features (values: 1-4), and total 556 samples divided into 124 training and 432 test samples. We have put the 432 independent test samples into the file

MonkValid.txt that are used as a validation file for the problem. The data are in UCI ML format. It is a simple problem; the target concept is: IF ((X1 = X2) OR (X5 = 1)) THEN 1 ELSE 0.

Wisconsin breast cancer data (WCBreast.txt): It has been downloaded from http://www.ics.uci.edu/~mlearn/MLRepository.html. It consists of 30 numeric features, and a total of 569 samples. It is a binary classification problem. To use as an example, we randomly split this data set into training and test subset into 1:1 ratio.

Breast cancer data (BreastCancer.txt): It is a microarray data file consisting of the gene expressions values of 3226 preprocessed genes across 22 samples. It is a 3-class classification problem. The classes of the samples are BRAC1, BRAC2 or Sporadic. All the features are numeric. The training subset consists of the fixed split of this data set, and the indexes of the training samples are stored in *BreastTrainIndex.txt*. The numbers of training and test samples are 17 and 5, respectively. The number of BRAC1, BRAC2 and Sporadic samples in the training and test subsets are {6, 6, 5}, and {2, 2, 1}, respectively. The original data set is available at http://research.nhgri.nih.gov/microarray/NEJM_Supplement/.

Brain cancer data (BrainPre.txt): It is a microarray data file consisting of expressions values of 12625 genes across 50 samples. It is a binary classification problem. This data file is provided as an example file for data preprocessing. The original data file is available at http://www-genome.wi.mit.edu/cancer/pub/glioma.

10. Execution of the software:

The EGPC software is implemented in Java programming language and available as a jar (Java Archive) executable file. Three jar files: *EGPCpre.jar*, *EGPCcom.jar*, and *EGPCgui.jar* are available with this software bundle. *EGPCpre.jar* is for preprocessing of microarray data in CLI mode; *EGPCcom.jar* and *EGPCgui.jar* are for data classification and important features` identification in CLI and GUI modes, respectively. *EGPCgui.jar* has also interfaces for data preprocessing in GUI mode.

- **Execution of EGPCpre.jar in CLI mode:**

To run the program from command prompt, type:

java [-Xmx<heap size>] **-jar EGPCpre.jar** [arguments...]

Command line arguments and formats:

-Xmx<heap size>: maximum heap size; some data sets may require higher heap size.
Example: -Xmx512m (m or M for mega byte).

-f <input file>: input data file name (with path if not on the current working directory); <input file> must be provided.

-o <output file>: output file name (with path if not on the current working directory); default: DataOut.txt.

-p <l:h:d:f>: preprocessing parameters; l=lower threshold, h-higher threshold, d=difference, f=fold change.
-n <normalization info>: normalization info; for log normalization type G with the base like G10 or Ge while for linear normalization type La:b where a:b is the range.
-h <header info>: header info; G: first column contains genes IDs; S: first row contains samples IDs; GS or SG for both.

Example:
java -jar EGPCpre.jar -f "DataFile/BrainPre.txt" -o BrainPro.txt -p 20:16000:100:3 -n Ge -h GS

- **Execution of EGPCcom.jar in CLI mode:**

To run the program type:
java [-Xmx<heap size>] **-jar EGPCcom.jar** [arguments...]

Command line arguments and formats:
-Xmx<heap size>: maximum heap size; some data sets may require higher heap size. Example: -Xmx512m (m or M for mega byte).
-u: UCIML format; default (if it is omitted) is Microarray format.
-d <data file>: data file name (with path if not on the current working directory); <datafile> must be provided.
-v <validation file>: validation file name (with path if not on the current working directory); if it is not provided, the training information must be provide under the "–t" below.
-s <sample size>: number of samples; must be provided.
-a <feature size>: number of features; must be provided.
-A <feature info>: feature information; default is that all features are numeric. See above "Features/Genes" for details about feature information.
-t <training info>: training subset information; the training information can be either the filename (with path if not on the current working directory) containing the indexes of the training samples or the training size of each type of sample delimited by colon like 179:106.
-c <classes>: number of classes; default is 2.
-m <ensemble size>: ensemble size; default is 3.
-F <functions>: functions to be used; functions are delimited by colon (:) and the default functions are "+:-:/:*:sqr:sqrt". Note here that the functions' string must be within double quotation (" ").
-p <population size>: population size; default is 1000.
-g <max gen>: maximum number of generations; default is 50.
-r <max run>: number of trials or repetition; default is 20.

- **Execution of EGPC on the first three example files from command prompt:**

1. Monk problem:

java -jar EGPCcom.jar -u -d "DataFile/Monk.txt" -v "DataFile/MonkValid.txt" -s 556 -a 6 -A L1:6:4 -F ">:<:=:<>:AND:OR:NOT:>=:<=" -m 3 -c 2 -r 3

2. Wisconsin Breast cancer (WCBreast.txt):
java -jar EGPCcom.jar -u -d "DataFile/WCBreast.txt" -s 569 -t 179:106 -a 30 -A N1:30 -F "+:-:*:/:SQRT:SQR:>:<:=:<>:AND:OR:NOT:>=:<=" -m 3 -c 2 -r 3

3. Breast cancer (BreastCancer.txt):
java -jar EGPCcom.jar -d "DataFile/BreastCancer.txt" -s 22 -a 4434 -t "DataFile/BreastTrainIndex.txt" -A N1:4434 -F "+:-:/:*:SQRT:SQR" -m 3 -c 3 -r 3

- **Execution of EGPCgui.jar in GUI mode:**

Go to the command prompt and type:
java [-Xmx<heapsize>] **-jar EGPCgui.jar**

Two snapshots of the software in GUI mode are shown in Figs. 1 and 2. The first page tells about the software. *Preprocess Data* page is for data preprocessing. The data filename and values of different parameters are entered from *Run EGPC* page. While EGPC is being executed on a data file, the number of correct predictions and the accuracies by single rules or sets of rules as well as those by the EGPC can be viewed on *View Accuracy* page. The more frequently selected genes/features are displayed in the page *Feature Ranking*. Here also three output files for rules, accuracy and gene frequency are created.

The previously mentioned three examples can also be run in graphical user interface mode. Note here that you must put the example files under the subdirectory “DataFile/” in the current working directory; otherwise EGPCgui.jar would generate errors. If you try to execute EGPCgui.jar by double clicking on it, it may not work.

11. Contact:

Should you have any question or found any bug, contact:

- Hitoshi Iba: iba@iba.k.u-tokyo.ac.jp
- Topon Kumar Paul: toponpaul@gmail.com

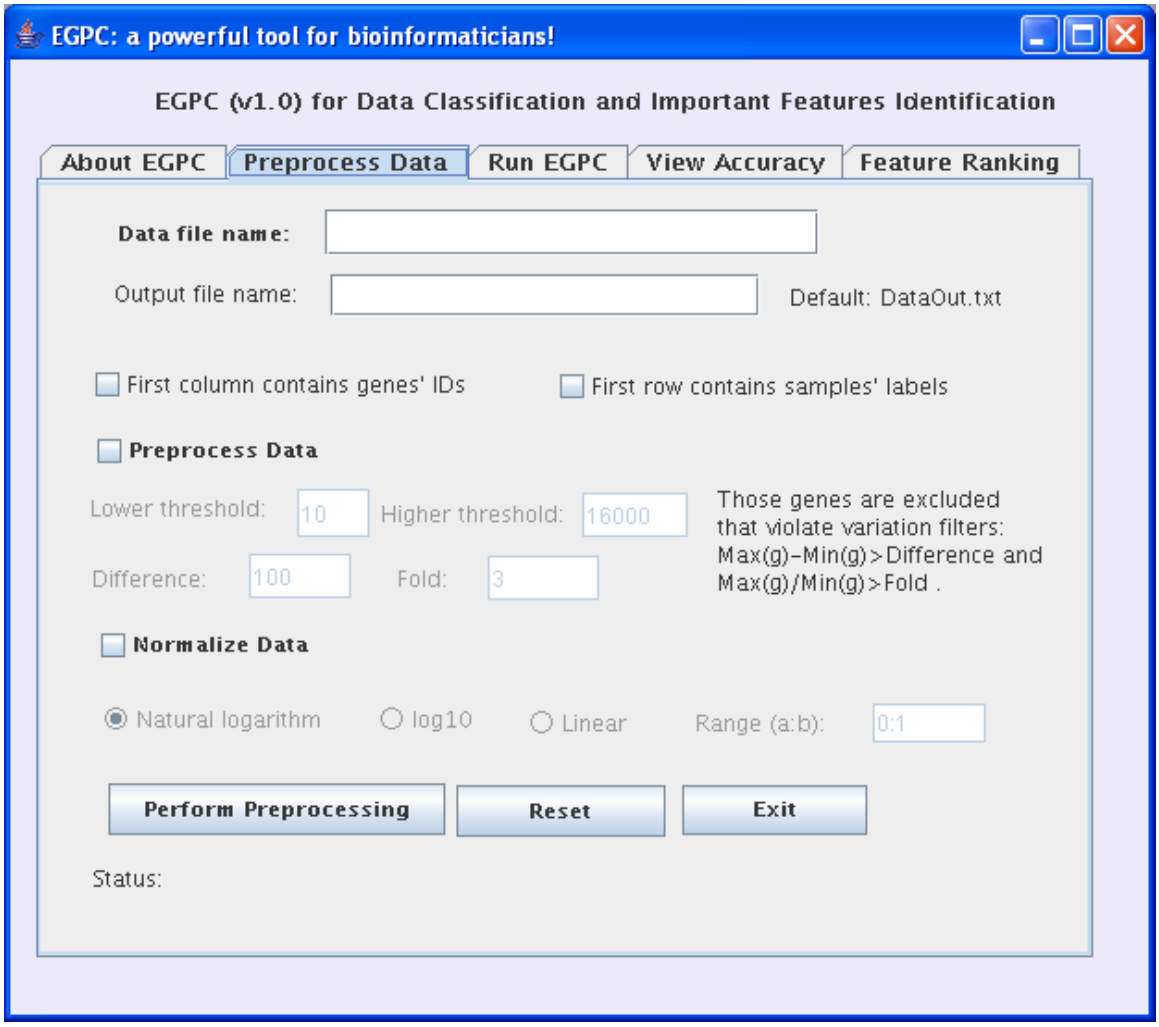

Figure 1. A screen shot of GUI of EGPC (Preprocess Data)

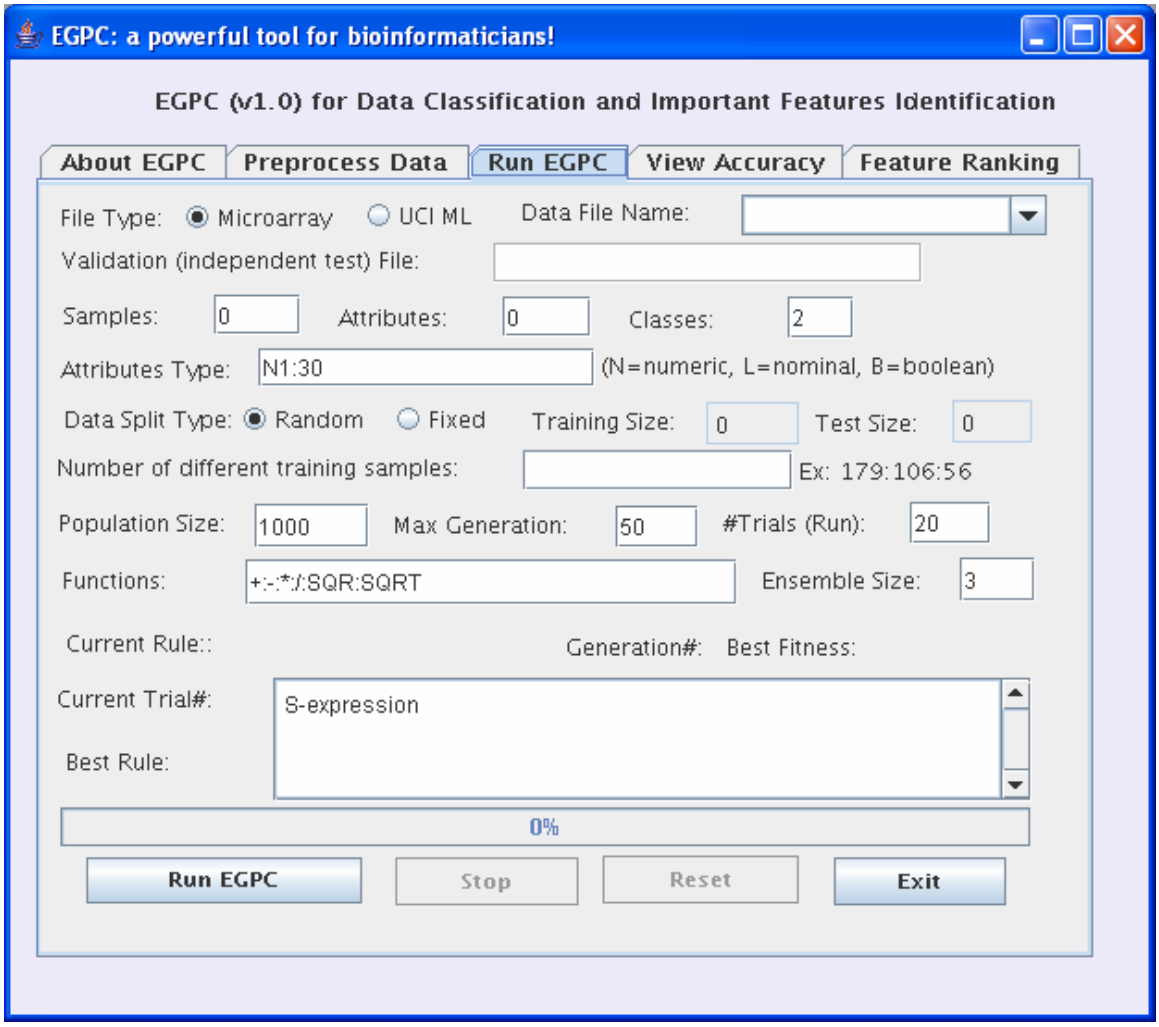

Figure 2. A screen shot of GUI of EGPC (Run EGPC)