International Journal of Bio-Science and Bio-Technology
Vol.7, No.4 (2015), pp.89-108
http://dx.doi.org/10.14257/ijbsbt.2015.7.4.10

A Review of Cancer Classification Software for Gene Expression Data

Tan Ching Siang[1],Ting Wai Soon[1],Shahreen Kasim[2], Mohd Saberi Mohamad[1]*,Chan Weng Howe[1],Safaai Deris[1],Zalmiyah Zakaria[1], Zuraini Ali Shah[1] and Zuwairie Ibrahim[3]

[1]Artificial Intelligence and Bioinformatics Research Group, Faculty of Computing, UniversitiTeknologi Malaysia, 81310 Skudai, Johor.*
[2]Faculty of Computer Science and Information Technology, UniversitiTun Hussein Onn Malaysia, 86400 Parit Raja, BatuPahat, Johor.
[3]Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang, 26600 Pekan, Pahang.
kyotcs@yahoo.com, shahreen@uthm.edu.my,ivan647389@hotmail.com, saberi@utm.my,whchan2@live.utm.my, safaai@utm.my, zalmiyah@utm.my, aszuraini@utm.my, zuwairie@ump.edu.my*

Abstract

Microarray technology provides a way for researchers to measure the expression level of thousands of genes simultaneously in a single experiment. Due to the increasing amount of microarray data, the field of microarray data analysis has become a major topic among researchers. One of the examples of microarray data analysis is classification. Classification is the process of determining the classes for samples. The goal of classification is to identify the differentially expressed genes so that these genes can be used to predict the classes for new samples. In order to perform the tasks of classification of microarray data, classification software is required for effective classification and analysis of large-scale data. This paper reviews numerous classification software applications for gene expression data. In this paper, the reviewed software can be categorized into six supervised classification methods: Support Vector Machine, K-Nearest Neighbour, Neural Network, Linear Discriminant Analysis, Bayesian Classifier, and Random Forest.

Keywords*: Cancer Classification, Gene Expression Data, Microarray,Supervised Classification Methods, Bioinformatics, Artificial Intelligence*

1. Introduction

Microarray technology allows the expression level of large amount of genes to be monitored simultaneously in a single experiment [1]. Nowadays, the analysis of gene expression data has become a major topic among bioinformaticians, biostatisticians, clinicians, and scientists. This is because analysis of gene expression data has allowed the discovery of hidden information that provides biological knowledge. However, gene selection and classification are needed for the analysis and interpretation of such data.

Cancer classification is a process of determining whether a patient does or does not have cancer. The goal of classification is to identify informative genes that can be used to predict the classes for new testing samples. With the increasing volume of data generated by modern biomedical studies, software is required for effective classification and analysis of large-scale data. Bioinformatics has emerged as a discipline in which there is emphasis on analysing the large-scale data [2]. Bioinformatics is dedicated to the discovery and

ISSN: 2233-7849 IJBSBT

implementation of software that facilitates and eases the understanding of biological processes. InSrinivasan *et al.*[3],the most general definition of bioinformatics in addressing biological problems is discussed.

Most biomedical researchers are looking for appropriate software which is not only can achieve high prediction accuracy but also includes a user friendly design in order to ease the implementation. Moreover, such softwareisvery useful if the source code is available. In addition, the software should be up-to-date with the related information to make sure that it is competitive with other software.

In this paper, the classification software applications for six supervised classification methodsare reviewed. The six supervised classification methods include the Support Vector Machine (SVM), K-Nearest Neighbour (KNN), Neural Network (NN), Bayesian Classifier, Linear Discriminant Analysis (LDA), and Random Forest (RF). Furthermore, the sources of the software and web-based applications are listed as well.

2. Software for Support Vector Machine (SVM)

2.1 LIBSVM

LIBSVM, a library for SVMs, was developed by Chang and Lin [4]. The main purpose of developing this software was to help users implementing SVM. This package supports three main learning tasks: classification, regression, and estimation of probability. For classification, it supports binary and multi-class classification. It also includes various formulations of SVM such as c-classification, v-classification, ∈-regression, and v-regression. Other features include support for cross-validation for performance measurement, model selection, and solving of unbalanced data using weighted SVM. It is mainly implemented in C++ and Java but there are many extensions such as R, MATLAB, Python, and Perl that have been developed by Chang and Lin and others. Moreover, it also provides different kernel settings such as linear, polynomial, and radial basis functions. This package is mainly for Windows and Linux.

2.2 SVMlight

SVMlight was developed by Joachims [5] by using the idea of Vapnik [6] to solve regression, classification, and ranking problems. SVMlight has a fast optimization algorithm and can be used for high-dimensional datasets. It is mainly implemented in C. This software is also efficient in handling large amounts of support vectors. Besides that, it provides two performance measurement methods based on error rate, recall, and precision. They are leave-one-out cross-validation and XiAlpha-estimates. The kernel settings that are supported by the software are radial basis function and polynomial. The SVMs can be trained with cost models as well. This software is mainly for Solaris, Windows, Linux, and Cygwin.

2.3. SVMTorch

SVMTorch was developed by Collobert and Bengio [7] for solving both classification and regression problems. The classification part is similar to SVMlight but the difference is that this software is adopted to solve large-scale regression problems. The decomposition algorithm used for regression is based on the idea of Osuna *et al.* [8]. This extension enables more than 20000 examples to be solved. This software is also faster in terms of computational time compared to previous proposed SVM software application, *Nodelib*that was developed by Flake and Lawrence [9]. The computational time of SVMTorch in processing both small and large datasets is less than that of *Nodelib*. As a result, this software has shown computational time improvements compared to *Nodelib*. Besides that, this software adds two new features for classification, multiclass

classification, and sparse data. This software also enables the format of binary files to be converted into the format of sparse files. It is implemented in C++ and is mainly for Solaris, Linux, and Windows.

2.4 mySVM

mySVM was developed by Ruping [10] by using the SVM that was introduced by Vapnik [11]. This software uses the same optimization algorithm as in SVMlight. It is mainly used for classification and regression. Besides, this software can also be applied for estimation of distribution and pattern recognition. The additional feature of this software is that users can use a variety of file formats as its inputs, such as sparse and dense formats. Besides, it offers different kernel settings such as dot, polynomial, neural, ANOVA, user-defined kernel, sum of other different kernels, and the product of other different kernels. The language for this software is C++ and it is available for Windows as well as Unix. mySVM is also being developed and implemented in the Java environment as JmySVM. JmySVM is part of the RapidMiner software. Please refer to http://rapid-i.com/content/view/181/190/ for more details about RapidMiner

2.5 Weka-LibSVM (WLSVM)

Weka [12] is data-mining software that includes numerous machine learning algorithms. This software is applicable for solving many problems such as classification, regression, data preprocessing, and clustering. It is implemented in a Java environment. SVM is one of the machine learning algorithms in Weka. The implementation of SVM is based on LIBSVM. LIBSVM offers several methods such as one-class SVM, R-SVM, R-SVM, and others. The combination of Weka and LIBSVM is named asWeka-LibSVM (WLSVM) [13]. This software offers different types of kernel settings such as linear, polynomial, sigmoid, and radial basis function. This software can be applied directly to datasets or using Java code to call the functions. It is mainly implementedon Windows, Mac OS, and Linux.

2.6 BSVM

BSVM was developed by Hsu and Lin [14] for solving multi-class classification and regression problems by using a decomposition algorithm [15]. Bound-constrained formulation is used to solve both regression and one-to-one multi-class classification. Besides that, Crammer and Singer's formulation is used to solve multi-class classification. After that, Crammer and Singer's formulation with squared hinge (L2) loss is added to solve the multi-class classification in BSVM version 2.08. In BSVM, simple working set selection is used to achieve faster convergences for the difficult cases. Different kernel settings are offered by this software such as polynomial, radial basis function, and so on. It is implemented in C++ and mainly for Linux and Windows.

2.7 TinySVM

TinySVM was developed by Bindings and Link [16] to solve classification and regression problems. It is widely used in many areas in the real-world such as recognition of hand-writing. It supports c-classification and c-regression. Besides that, it can handle datasets with large amounts of training samples and feature dimensions. The number can be several tens of thousands and hundreds of thousands for training samples and feature dimensions, respectively. The optimization algorithm is similar to that of SVMlight and this allows it to perform faster in handling binary classes compared to SVMlight. It also provides leave-one-out and XiAlpha-estimates. It is implemented in C++ with OO style andmainly for Unix and Windows.

2.8 *SVM in R (e1071)*

SVM was implemented in R by Karatzoglou and Meyer [17]. This R package of SVM is mainly for regression, one-class classification, and classification problems. It is the implementation of LIBSVM in an R environment. It also supports different formulations of SVM such as c-classification, v-classification, spoc-classification, and ∈-regression. This package is updated from timetotime. The newest version of e1071 is maintained by Meyer *et al.* [18]. A variety of kernel settings are supported such as linear, polynomial, and radial basis function. Apart from that, a variety of new features such as clustering, imputing missing values, cross-validation methods for performance measurement, Naïve Bayes classifiers, and others have been added to this package. The package can be installed on Linux, Mac OS, and Windows platforms.

2.9 *LSVM*

LSVM stands for Lagrangian Support Vector Machine,which was developed by Mangasarian and Musican [19]. It is a fast SVM that is implemented in MATLAB and beable to classify datasets that containinga huge number of patterns efficiently. This software takes only 34 minutes on a Pentium II 400 Mhz desktop to finish the task of classifying a dataset with over two million points and 10 features. A simple iterative approach is used to train the SVM in order to classify the patterns after training. Besides, LSVM can also be used for solving nonlinear classification problems. This software supports different kernel settings such as linear kernel, cubic kernel, and quadratic kernel. From the comparison between LSVM and SVMlight in terms of computational time, LSVM is comparable to or even better than SVMlight. This software is also easier to code since only a few lines of codes are involved. The MATLAB code for LSVM can be obtained from http://jmlr.org/papers/volume1/mangasarian01a/html/node3.html.

2.10 PyML

PyML [20] stands for machine learning written in a Python environment. It is a framework that contains numerous classification and regression methods. SVM is one of the classifiers that are included in PyML. PyML can be used for data preprocessing and normalization as well. Apart from that, it also provides feature selection methods, performance measurement of classifiers based on cross-validation, ROC curves and error rates, different kernel settings, multiclass classification methods, and other classifiers. For multiclass classification, one-against-one and one-against-rest are supported. The delimited and sparse file formats are supported by this software. It can be installed on Linux and Mac OS platforms.

2.11 PSVM

PSVM is also known as Proximal Support Vector Machine and was developed by Fung and Managasarian [21]. Its implementation differs from standard SVM because the latter classifies the samples to either one of the classes. In the case of PSVM, the samples are classified to the nearest two parallel planes. PSVM was developed to solve two-class classification as well as the problem of unbalanced classes using nonlinear PSVM. This software is written in MATLAB for two versions of PSM: linear and nonlinear. Both versions can be downloaded at the same link. Besides, different kernel settings are offered to execute nonlinear PSVM. The kernel settings are radial basis function and square Gaussian.

2.12 MSVMpack

MSVMpack is a software package that mainly used for solving multi-class classification. It was developed by Lauer and Guermeur [22]. This package contains four

multi-class classification methods that were proposed previously. The purpose of developing this package was to include all four multi-class classification methods in a package in order to ease theimplementation. The purpose of developing this package was to include all four multi-class classification methods [23-26] in a single package for easy. This package also supports different kind of kernel settings such as linear, Gaussian radial basis function, and polynomial. Instead of standard kernel settings, it also supports custom kernels. Other features of this package are data normalization, algorithms for the selection of the model, and use of C API to ease the integration in other programs. This package also offers aWeb server that can be used by users to access the functions through an Internet connection. This package is mainly for Linux and MacOS.

2.13 Summary of SVM Software

Table 1 and 2 show a summary and resources of SVM software respectively.

Table 1. A Summary for SVM Software

No	Software	Author/Year	Language	Features
1	LIBSVM	Chang and Lin [4]	C++ and Java	- Various SVM formulations - Performing well in multi-class classification - Solving regression problems - Estimation of probability - Solving unbalanced data using weighted SVM - Supporting of cross validation - Selection of model - Different kernel settings - Extensions: MATLAB, R, Python, Perl and others
2	SVMlight	Joachims [5]	C	- Has optimization algorithm - Solving regression, classification and ranking problems - Can be used for high-dimensional datasets - Efficient in handling large amount of support vectors - Using LOOCV and XiAlpha-estimates of error rate, precision and recall for performance measurement - Supporting various kernel settings
3	SVMTorch	Collobert and Bengio [7]	C++	- Similar to SVMlight - Solving large-scale of regression problem - Multiclass - Sparse data
4	mySVM	Ruping [10]	C++	- Solving classification and regression problems - Has optimization algorithm - Can use variety of input file

				formats - Different kernel settings
5	Weka-LibSVM (WLSVM)	Hall *et al.* ; EL-Manzalawy and Honavar [12, 13]	Java	- Similar to LIBSVM - Different types of kernel settings - Can be directly applied or using Java code to call the functions to process the datasets
6	BSVM	Hsu and Lin [14,15]	C++	- Solving multi-class classification and regression - Bound formulation is used to solve one-to-one multi-class classification and regression problems - Crammer and Singer's formulation is used to solve multi-class classification
7	TinySVM	Bindings *et al.* [16]	C++ with OO style	- Solving classification and regression problems - Dealing with large number of training samples and feature dimension - Has optimization algorithm which is similar with SVMlight - Providing LOOCV and XiAlpha-estimates
8	SVM in R (e1071)	Karatzoglou *et al.* [17]	R	- The implementation of LIBSVM in R - Solving clustering, classification, and regression problems - Different formulations of SVM - Different kernel settings - Imputing missing values, cross validation for performance measurement -Including Naïve Bayes classifier
9	LSVM	Mangasarian and Musicant [19]	MATLAB	- Iterative approach is used to train SVM - A fast SVM in classifying datasets with huge number of patterns - Different kernel settings
10	PyML	Ben-Hur [20]	Python	- Numerous of classifiers including SVM - Can be used for data preprocessing and normalization - Other features such as

				feature selection, performance measurement methods, different kernel settings, and multi-class classification
11	PSVM	Fung and Mangasarian [21]	MATLAB	- Linear and nonlinear PSVM - Solving the classification problems of two-class and unbalanced classes - Kernel settings are provided for nonlinear PSVM
12	MSVMpack	Lauer and Guermeur [22]	C	- Solving multi-class classification - Including four multi-class classification methods that were proposed previously - Various kernel settings and custom kernels - Data normalization - Selection of model - C API for the ease of integration in other programs - A web server for accessing function of package through internet

Table 2. Sources for SVM Software

No	Software	Sources
1	LIBSVM	http://www.csie.ntu.edu.tw/~cjlin/libsvm/
2	SVMlight	http://svmlight.joachims.org/
3	SVMTorch	http://www.cs.cmu.edu/afs/cs.cmu.edu/project/learn-43/lib/photoz/.g/mmp/trees/SVM/
4	mySVM	http://www-ai.cs.uni-dortmund.de/SOFTWARE/MYSVM/index.html
5	Weka-LibSVM (WLSVM)	http://www.cs.iastate.edu/~yasser/wlsvm/
6	BSVM	http://www.csie.ntu.edu.tw/~cjlin/bsvm/
7	TinySVM	http://chasen.org/~taku/software/TinySVM/
8	SVM in R (e1071)	http://cran.r-project.org/web/packages/e1071/index.html
9	LSVM	http://research.cs.wisc.edu/dmi/lsvm/
10	PyML	http://sourceforge.net/projects/pyml/
11	PSVM	http://research.cs.wisc.edu/dmi/svm/psvm/
12	MSVMpack	http://www.loria.fr/~lauer/MSVMpack/MSVMpack.html

2. Software for K-Nearest Neighbour (KNN)

3.1 Mayday Software

Mayday was developed originallyby Gehlenborg [27]. It is software used for microarray data analysis, storage, and visualization [28]. It also offers a variety of plug-ins

such as connection to an R environment and different types of data mining methods. Besides, the classification methods from the Weka library [29] are also used as plug-ins for this software. Mayday software is Web-based classification software that uses Java in implementation. There are a variety of classifiers in Mayday classification software such as KNN, decision tree, and SVM. Other features such as feature selection and data preprocessing are offered by this software too. It can be used through network access or downloads from the Mayday website. In Battke *et al.* [30], Mayday software was rewritten to make it more efficient for future use and developments. An example of a plug-in is the addition of new clustering methods. The software is mainly for Linux, Mac OS, and Windows.

3.2 *kknn*

The package 'kknn' is the implementation of a weighted KNN classifier in an R environment. It was developed originallyby Schliep and Hechenbichler [31] based on the idea of Hechenbichler and Schliep [32]. This package is updated from timetotime and its newest version was maintained by Schliep and Hechenbichler [33]. This package is used for classification, clustering, and regression. In it, Minkowski distance is used for KNN. This package also provides a few datasets for testing, such as Iris and Ionosphere. Cross-validation is offered to assess the performance of KNN. This package is mainly for Linux, Mac OS, and Windows.

3.3 knnGarden

knnGarden is an R package for classification using multi-distance-based KNN. The newest version of this package is maintained by Wei *et al.* [34]. A few functions are provided in this package, such as filling in missing observations in the dataset and two versions of KNN. The versions are KNN with Mahalanobis distance and versatile distance. Function knnMCN is for KNN with Mahalanobis Distance while function knnVCN is for KNN with Versatile Distance. These functions are based on the idea of Venables and Ripley [35]. This package is for Mac OS, Linux, and Windows.

3.4 Weka-KNN

Weka [12] is data mining software that provides numerous machine learning algorithms. A KNN classifier is included as one of the machine learning methods in Weka. As a result, users can use Weka-KNN to perform the implementation of KNN for their researches. The file formats arff and csv are supported by this software. A cross-validation procedure is offered for the performance evaluation as well. Users can choose the number of folds for implementing the cross-validation procedure. This software is implemented in a Java environment.

3.5 rknn

The package 'rknn' is the implementation of Random KNN (RKNN) in an R environment for classification and regression with variable selection [36]. The Random KNN is based on the idea of Li [37]. A parallel version of the package 'rknn' was proposed by Harner *et al.* [38]. The newest version of this package was maintained by Li [39]. Random KNN is used for classification as well as regression. Random KNN can also be used for selecting important features using an RKNN-FS algorithm in order to solve high-dimensional datasets [40]. The implementation of RKNN-FS for feature selection and Random KNN for classification gives superior results compared to RF. Besides, this package offers extra functions such as normalization of data, feature selection, and performance measurement methods. This package can be installed on Linux, MacOS, and Windows.

3.6 ArrayMinerClassMaker

ArrayMiner [41] is software used for the analysis of gene expression data. It can be used for clustering and classification. This software is divided into two parts: Clustering and ClassMaker. For classification, ArrayMinerClassMaker is used. ClassMakeris mainly used for class prediction for new samples. Two classification methods including KNN and a voting method are available for ClassMaker. Other features are offered by this software such as cross-validation for performance measurement, train-and-test evaluation, multi-class classification, and others. The csv file format is mainly used in this software. For more details, please refer to http://www.optimaldesign.com/ArrayMiner/ClassMarker.htm. This software is implemented in Java and available for MacOS and Windows.

3.7 BRB-Array Tools

BRB-ArrayTools [42] is integrated software that is used for the visualization and analysis of microarray data. This software can be used for clustering, classification, and feature selection. KNN is one of the classifiers in this software. This software can be installed as a plug-in for Excel. As a result, Excel users can easily use this software to implement KNN for classification through the menu bar in Excel. Apart from that, this software provides functions such as class prediction, class discovery, normalization, cluster analysis, graphical presentation, and performance measurement such as cross-validation. The input of this software is usually in the form of Excel spreadsheets. Over 100 published microarray datasets are provided by this software. The analysis tools are written in Java, C, R, and Fortran languages. As a result, users can use all the analysis tools in Excel.

3.8 Summary ofKNN Software

Table 3 and 4 show the summary and sources of KNN software respectively.

Table 3. A Summary for K-Nearest Neighbor Software

No	Software	Author/Year	Language	Features
1	Mayday software	Gehlenborg [27]	Java	- A variety of classifiers including KNN - Feature selection methods are provided - Data preprocessing
2	kknn	Schliep and Hechenbichler [31]	R	- Weighted KNN for classification, clustering and regression - Few datasets are provided - Cross validation for performance measurement
3	knnGarden	Wei *et al.* [34]	R	- KNN for multi-based distance - Filling missing observations
4	Weka-KNN	Hall *et al.* [12]	Java	- Implementation of KNN in Weka - Cross validation is supported

5	rknn	Li *et al.* [40]	R	- Random KNN for classification and regression - Providing functions of data normalization, feature selection, performance measurement and others
6	ArrayMinerClassMaker	Falkenauer and Marchand [41]	Java	- Can be used for clustering and classification - cross validation for performance measurement -Multi-class classification
7	BRB-ArrayTools	Simon *et al.* [42]	C, R, Fortran, and Java	- Can be used for feature selection, clustering and classification - Addition of performance measurement methods such as cross validation

Table 4.Sources for K-Nearest Neighbor software

No	Software	Sources
1	Mayday Software	http://www-ps.informatik.uni-tuebingen.de/mayday/wp/?page_id=8
2	kknn	http://cran.r-project.org/web/packages/kknn/
3	knnGarden	http://cran.r-project.org/web/packages/knnGarden/index.html
4	Weka-KNN	http://www.ibm.com/developerworks/apps/download/index.jsp?contentid=494038&filename=os-weka3-Example.zip&method=http&locale=
5	rknn	http://cran.r-project.org/web/packages/rknn/index.html
6	ArrayMinerClassMaker	http://www.optimaldesign.com/ArrayMiner/ArrayMinerDownload.html
7	BRB-ArrayTools	http://linus.nci.nih.gov/BRB-ArrayTools.html

4. Software for Neural Networks (NN)

4.1 Pattern Classification Program (PCP)

PCP [43] is open-source software for a variety of machine learning methods. This software is mainly used for solving classification problems. The examples of classification methods are Multi-layer Perceptron, SVM, and KNN. The advantages of this software are that it offers a variety of classification, gene selection, gene extraction, and performance measurement methods. A cross-validation procedure is offered for performance evaluation. Dimension reduction methods are also provided such as Singular Value Decomposition (SVD) and Principal Component Analysis (PCA). Tab-delimited

text files are used as input files by PCP. This software is implemented in C and C++ languages and is available for Windows, Cygwin, and Linux.

4.2 nnet

The package 'nnet' is implements NN in an R environment for classification and class prediction based on the idea of Ripley [44] as well as Venables and Ripley [35]. It is an R package that was maintained by Ripley [45]. This package is mainly for feed-forward single-hidden-layer NNs with a single hidden layer. This package offers multinomial log-linear models as well. The functions include training the NN and predicting new examples using the trained NN. This software is available for Linux, Mac OS, and Windows.

4.3 neuralnet

'neuralnet' was originally developed by Fritsch and Gunther [46]. NN is implemented in an R environment as the R package 'neuralnet' [47]. The package focuses on the supervised learning algorithms and the newest version of this package is maintained by Fritsch and Gauenther [48]. This package was built for training multi-layer perceptrons in the analysis of regression. This package offers the use of backpropagation and resilient backpropagation algorithms. As a result, users can use either backpropagation or resilient backpropagation. Besides that, there is a plot function for visualizing the results of NN. The visualization provides a better understanding of outputs for users. This package is mainly for Linux, Windows, and Mac OS.

4.4 pnn

'pnn' is an R package for Probabilistic Neural Network (PNN) based on the idea of Specht [49]. This package is mainly used to solve the problems of automatic learning. It can perform well even when the number of known observations is small. This package was maintained by Chasset [50]. In this package, four main functions and a dataset are provided. The functions are 'learn', 'smooth', 'perf', and 'guess'. The learn function is used for training a new PNN with new training data. The smooth function is used to set the parameters of smoothing, the perf function to evaluate the performance of the PNN, and the guess function to predict the class for new samples. This package can be installed on Linux, Windows, and Mac OS.

4.5 RSSNS

RSSNS is an R package for NNsthat usesthe Stuttgart Neural Network Simulator (SNNS) [51]. The newest version of RSNNS is maintained byBergmeir and Benitez [52]. Basically, there are a variety of NNsthat are implemented in SNNS. This library is implemented in R to form the RSNNS package. As a result, a comprehensive analysis and visualization of NNs are offered by this package. Examples of NNs are multi-layer perceptron and radial basis function NNs. This package is available for Windows, Linux, and Mac OS.

4.6 Summary of Neural Networks Software

Table 5 and 6 show the summary and sources of NN software respectively.

Table 5.A Summary for Neural Networks Software

No	Software	Author/Year	Language	Features
1	PCP	Buturovic [43]	C and C++	- Can be used for classification, gene selection, gene extraction - Performance measurement using cross validation
2	nnet	Ripley [45]	R	- Supervised learning method for classification - Using Feed-forward NN with single hidden layer
3	neuralnet	Fritsch and Gunther [46]	R	- Using of backpropagation to train NN - Plotting function is provided for visualizing results of NN
4	pnn	Chasset [50]	R	- R package for PNN - Perform well even the number of known observations is small - Providing four main functions, *learn*, *smooth*, *perf*, and *guess*

Table 6. Sources for Neural Networks Software

No	Software	Sources
1	PCP	http://pcp.sourceforge.net/
2	nnet	http://cran.r-project.org/web/packages/nnet/index.html
3	neuralnet	http://cran.r-project.org/web/packages/neuralnet/index.html
4	pnn	http://cran.r-project.org/web/packages/pnn/
5	RSNNS	http://cran.r-project.org/web/packages/RSNNS/index.html

5. Bayesian Classifier Software

5.1 Iterative Bayesian Model Averaging

Yeung *et al.* [53] proposed an Iterative Bayesian Model Averaging method for gene expression data. Traditional Bayesian Model Averaging (BMA) cannot deal with high-dimensional data. However, this method uses a backward elimination technique to eliminate the uninformative genes based on a rank-ordered list of the genes and then applies the traditional BMA algorithm. An R package for Iterative BMA was developed by Yeung and Raftery [54] for binary classification. The function 'iterateBMAglm.train'in this package is for the selection of relevant genes by iteratively implementingthe BMA algorithm from the BMA package. Moreover, the data should consist of two class types. A list of selected genes and models of the training data, the classification error, and the Brier Score of the test set are returned. This software is available for Bioconductor in an R framework.

5.2 Full Bayesian Network Classifier

Su and Zhang [55] proposed a Full Bayesian Network Classifier. They used conditional probability tables to represent the gene independence, and decision trees were generated based on the conditional probability tables. The motivation for this method was to overcome the limitations of the structure learning in Bayesian Network learning. The idea is that, instead of using the structure to represent variable independence, the method

uses a full Bayesian Network as the structure of the targets. The software for this method was also built by Su and Zhang for implementation in a Weka environment. In addition this software is efficient in terms of both training time and classification time compared to other methods in the Wekaenvironment. TheWekaframeworkcan be downloaded from http://www.cs.waikato.ac.nz/ml/weka/downloading.html.

5.3 Bayesian Trans-Dimensional Sampling

Lamnisos *et al.* [56] proposed a Bayesian gene selection and classification based on a trans-dimensional sampling technique for gene expression data. This method implements a new form of reversible jump sampling which is called as trans-dimensional Markov chains. However this new form of reversible jump sampling was implemented with joint updating of the model and the auxiliary variables. By using the new form of reversible jump sampling, it could avoid slow mixing in the chain because the auxiliary variables are correlated with the model parameters. In addition, a general model that splits the addition–deletion move and combines local moves with more global moves is developed. This is used for tuning the parameters in order to obtain a more acceptable fall in rates. Moreover, this technique allows the dimensions space to be changed in order to deal with high-dimensional gene expression data. This software is available for MATLAB users.

5.4 Bayesian Stochastic Search Variable Selection

Jun and Su [57] proposed Bayesian stochastic search variable selection (SSVS) with a new generalized posterior probability distribution to overcome the high-dimensional problem that exists in gene expression data. They used SSVS as an efficient sampling-based technique and proposed a new generalized posterior probability distribution to deal with high-dimensional gene expression data. Moreover a generalized singular g-prior is used to overcome the singularity problem of the covariance matrix involved in the g-prior distribution. However this singular g-prior has been found to be effective in overcoming the statistical problem of a big number of genes in a small sample size. In addition, the SSVS algorithm uses the simulation-based Markov Chain Monte Carlo (MCMC) technique. However, this software is available for MATLAB users.

5.5 Naïve Bayes Classifier

Rosen *et al.* [58] developed a Web-based software by using a Naïve Bayes Classifier technique for classification of meta-genomic data. Meta-genomic data are gene expression data recovered from environmental samples. This software is used in taxonomic classification in order to assess the exact information of an organism. Two functions of the Naïve Bayes Classifier are introduced. The first function is the novice functionality, while the second is the expert functionality. These two functions allow users to choose the parameters for execution of this classification software. These parameters include the upload file, genome list, and number of read length. The output file contains a summary of the matches and score distribution. This software enables users to easily analyse the taxonomic composition of the input datasets with a convenient user interface.

5.6 Summary of Bayesian Classifier Software

Table 7 and 8 show the summary and sources of Bayesian Classifier software respectively.

Table 7. A Summary for Bayesian Classifier Software

No	Software	Author/year	Language	Feature
1	Iterative Bayesian Model Averaging	Yeung *et al.* [53]	R	- Using backward elimination technique to discard uninformative gene - Applying Bayesian Model Averaging algorithm
2	Full Bayesian Network Classifier	Su and Zhang [55]	Java	- Decision trees are generated from conditional probability tables
3	Bayesian Trans-dimensional Sampling	Lamnisos *et al.* [56]	MATLAB	- Using trans-dimensional Markov chains as a reversible jump sampling technique
4	Bayesian Stochastic Search Variable Selection	Jun *et al.* [57]	MATLAB	- Using SSVS technique and a new generalized posterior probability distribution to deal with dimensional data
5	Naïve Bayes Classifier	Rosen *et al.* [58]	PHP	- Using meta-genomic data in taxonomic classification

Table 8. Sources for Bayesian Classifier Software

No	Software	Sources
1	Iterative Bayesian Model Averaging	http://www.bioconductor.org/packages/2.11/bioc/html/iterativeBMA.html
2	Full Bayesian Network Classifier	http://www.cs.unb.ca/profs/hzhang/FBC.rar
3	Bayesian Trans-dimensional Sampling	http://www2.warwick.ac.uk/fac/sci/statistics/staff/academicresearch/steel/steel_homepage/software/transsup.zip
4	Bayesian Stochastic Search Variable Selection	http://www.sta.cuhk.edu.hk/xysong/geneselection/
5	Naïve Bayes Classifier	http://nbc.ece.drexel.edu/newJob.php

6. Software for Linear Discriminant Analysis (LDA)

6.1 Regularized LDA

Guo *et al.* [59] proposed a modified LDA by using the idea of the 'nearest shrunken centroid' to deal with high-dimensional data. This technique shrinks the genes to the class centroids by a threshold in order to reduce the effect of noisy genes. However each gene in the group of centroidis shrunken individually with an assumption that these genes are independent of each other. Moreover this method employs the SVD trick to construct the matrix inversion in order to facilitate the computation complexity. However this method needs to optimize two parameters for a two-dimensional grid and the computation involves a large matrix manipulation; hence this method is computationally less efficient. This method performs well even though the number of classes is big. It is available for R packages.

6.2 Sparse Discriminant Analysis

Traditional LDAperforms well only on datasets of low-dimensional space. However, Clemmensen *et al.* [60] proposed a sparse discriminant analysis method to deal with high-

dimensionality. The sparseness criterion allows the gene selection and LDA classification processes to be performed simultaneously; therefore it can deal with gene expression data. This method is an extension of LDA to high-dimensional data as a result of the discriminant vectors involving only a subset of genes. Moreover this method is based on an optimal scoring framework from LDA. In addition, this method is also extended to execute sparse discrimination via mixtures of Gaussian distributions. This method is available for R package and MATLAB users.

6.3 Robust Regularized *LDA*

Gschwandtner *et al.* [61] proposed a robust technique for regularized discriminant analysis for a large number of variables with small sample size. This method is a combination of regularization and a robust technique to solve for data containing outliers and noisy genes. Moreover this method is executed using a sparse estimation of the inverse covariance matrix. The sparseness is manipulated by a penalty parameter. The outliers are handled by a robustness parameter which identifies the amount of observations using a maximum likelihood function. This method is available as an R package. The object of the class 'rrlda' is returned, which can be used for class prediction. The class prediction is based on the estimated inverse covariance matrix and the mean of each group of objects.

6.4. Summary ofLDA Software

Table 9 and 10 show the summary and sources of LDA software respectively.

Table 9. A Summary for Linear Discriminant Analysis Software

No	Software	Author/year	Language	Feature
1	Regularized LDA	Guo *et al.* [59]	R	- Using nearest shrunken centroid technique to reduce the effect of noisy genes.
2	Sparse Discriminant Analysis	Clemmensen *et al.* [60]	R, MATLAB	- Performing gene selection and LDA classification processes simultaneously by using the sparseness criterion.
3	Robust Regularized LDA	Gschwandtner *et al.* [61]	R	- Combination regularization and robust technique to overcome outlier and noisy genes.

Table 10. Sources for Linear Discriminant Analysis Software

No	Software	Sources
1	Regularized LDA	http://cran.r-project.org/web/packages/rda/index.html
2	Sparse Discriminant Analysis	http://cran.r-project.org/web/packages/sparseLDA/index.html
		http://www2.imm.dtu.dk/pubdb/p.php?5671
3	Robust Regularized LDA	http://cran.r-project.org/web/packages/rrlda/index.html

7. Random Forest (RF)

7.1 Backward Elimination Random Forest

Díaz-Uriarte and Alvarez [62] proposed a gene selection and classification using the backward elimination technique for RF. The backward elimination technique is performed to iteratively discard the uninformative genes based on variables' importance scores. The variable importance score for each gene is calculated by RF, which is used for gene ranking. Then 20% of the bottom genes are removed using the iterative elimination algorithm. These elimination processes are repeated until the subset of genes contains only two genes. Then the smallest subset with the lowest out-of-bag error rate is selected for classification. The small subset with informative genes will improve the classification accuracy. This method is available as an R package. Moreover, this method is developed for Web-based application with a user-friendly interface called GeneSrF. The GeneSrF application is built using Python language with parallel computing for gene selection and classification [63].

7.2 Online Random Forest

Saffari *et al.* [64] extended the traditional offline RF to an online mode RF. This allowed the offline RF to learn from online training data samples. Moreover the online RF only has small memory requirements because the sample does not need to be stored. In addition a large amount of available data can be exploited in online mode. Furthermore, this method can overcome the multiclass problem without using ordinary binary decompositions. Such binary decompositions have some limitationswith regard to computational complexity and unbalanced classes of datasets. In order to operate RF in online mode, an adaptation was made for bagging and random decision trees in online mode. This method is available in C++ programming language by using ATLAS or LAPACK subroutines.

7.3 cforest

Hothorn *et al.* [65] proposed a new technique of RF for gene selection and classification. Unlike the traditional RF algorithm, which uses CART classification trees to build the forest, this new method uses a conditional inference theory to build the classification tree. The conditional inference trees are suited to each of the bootstrap samples of the learning sample. This method can be found in the 'cforest' function in the R add-on package 'party'. This method is a computational toolbox for recursive partitioning. However the recursive partitioning is constructed by means of conditional distribution of linear statistics of the permutation test framework. The permutation test framework is used to seek the optimal binary split for response variables. This method aims at providing an iterative partitioning laboratory assembly for building tree-based regression and classification models.

7.4 Guided Regularized Random Forest

Deng and Runger [66] proposed a modified RF algorithm called Guided Regularized RF. This new algorithm uses preliminary variable importance scores to guide the gene selection process of Regularized RF. These preliminary variable importance scoresare calculated from RF, after being normalized. Then these variable importance scores are used to guide the gene selection process in Regularized RF. However, the gene selection process is stabilized by evaluation of the genes by the training data at each node. Because of the importance scores are calculated from RF based on all trees in all the training data, Guided Regularized RF performs better than Regularized RF. Guided RF Forest was built

by Deng and Runger and it is available as an R package. Moreover, Regularized RF can be implemented in this package.

7.5 Big Random Forest

RF has been extended to deal with high-dimensional data by using a parallel version. This version was developed for handling datasetsthat are too large to be processed. By using parallel programming, the implementation of RF can be processed in multiple machines. The forests can be created in parallel in two stages. In the first stage, trees are grown in parallel in a single machine using foreach. In the second stage, multiple forests are created in parallel on multiple machines and then all of the forests are converged into one forest. For big datasets, the stored data, middle-level computations, and some outputs are first cached on the disk. This allows RF to be built on considerably large datasets without hitting the RAM limit; therefore it can avoid excessive virtual memory swapping by the operating system. Big RF was developed by Lim *et al.* [67]and it is available as an R package.

7.6 Summary of Random Forest Software

Tables 11 and 12 show the summary and sources of RF software respectively.

Table 11. A Summary for Random Forest Software

No	Software	Author/year	Language	Feature
1	Backward Elimination Random Forest	Díaz-Uriarte and Alvarez [62]	R, Python	- Using backward elimination technique to iteratively discard non-informative genes.
2	Online Random Forest	Saffari *et al*. [64]	C++	- Bagging and random decision trees are generated in online mode.
3	cforest	Hothorn *et al.* [65]	R	- Using a conditional inference theory to build the classification tree
4	Guided Regularized Random Forest	Deng *et al.* [66]	R	- Using preliminary variable important scores to guide the gene selection process in order to stabilize the gene selection process.
5	Big Random Forest	Lim *et al.* [67]	R	- Handling large dataset by using parallel programming.

Table 12. Sources for Random Forest Software

No	Software	Sources
1	Backward Elimination Random Forest	http://cran.r-project.org/web/packages/varSelRF/index.html
		http://genesrf.bioinfo.cnio.es
2	Online Random Forest	http://www.ymer.org/research/files/online-forest/OnlineForest-0.11.tar.gz
3	*c*forest	http://cran.r-project.org/web/packages/party/index.html
4	Guided Regularized Random Forest	http://cran.r-project.org/web/packages/RRF/index.html
5	Big Random Forest	http://cran.r-project.org/web/packages/bigrf/index.html

8. Conclusion

Recently, a number of powerful software and Web-based softwareapplications have been developed for classification of gene expression data. In this paper, we present a comprehensive review of software for six different types of supervised classification methods. The methods are Support Vector Machine (SVM), K-Nearest Neighbour (KNN), Neural Networks (NN), Linear Discriminant Analysis (LDA), Bayesian Classifier, and Random Forest (RF). A total of 37 software applications are discussed in this paper. Despite the availability of many software applications for classification of gene expression data, most of them still have some limitationswith regard to statistical aspects, computational performance,and user-friendliness. Therefore, there is a need to develop better software. The software should include numerous classification methods in order to provide a platform for users to choose which method they are going to use. Apart from that, the software should provide a user-friendly environment.

Acknowledgements

We would like to thank the Universiti Teknologi Malaysia for supporting this research through a GUP research grant (Grant number: Q.J130000.2507.05H50) and a Matching grant (Grant number: Q.J130000.3007.00M27).Thisresearch was also funded by RACE research grant (Grant number: 1447) from the Malaysian Ministry of Education.

References

[1] L.-J. Zhang, Z.-J. Li and X. Hu, "A Hybrid Gene Selection Method for Cancer Classification", **(2007)**.
[2] A. Bhattacharya, "Bioinformatics: From molecules to systems. J. Biosci, vol.32, no.5, **(2007)**, pp.807.
[3] N. Srinivasan, R. Sowdhamini and A. Bhattacharya, "Computational biology: More than just a set of techniques", J.Biosci, vol.32, **(2007)**, pp.1-2.
[4] C. C. Chang and C. J. Lin, LIBSVM: A Library for Support Vector Machines, **(2001)**.
[5] T. Joachims, SVMlight: Support Vector Machine, **(1999)**.
[6] V. N. Vapnik, "The Nature of Statistical Learning Theory", Springer-Verlag New York, Inc., **(1995)**.
[7] R. Collobert, S. Bengio, "SVMTorch: Support Vector Machines for Large-Scales Regression Problems", J. Machine Learning Research, vol.1, **(2001)**, pp.143-160.
[8] E. Osuna, R. Freund and F. Girosi, "An improved training algorithm for support vector machines", In: Neural Networks for Signal Processing [1997] VII. Proceedings of the 1997 IEEE Workshop, **(1997)**.
[9] G. W. Flake and S. Lawrence, "Efficient SVM Regression Training with SMO", Machine Learning, vol.46, no.1-3, **(2001)**, pp.271–290.
[10] S. Ruping, mySVM-Manual, **(2000)**.
[11] V. N. Vapnik, "Statistical learning theory", **(1998)**.
[12] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, "The WEKA Data Mining Software: An Update", *SIGKDD Explorations*, vol.11, no.10-18, **(2009)**.
[13] E. Y. Manzalawy and V. Honavar, "WLSVM: Integrating LibSVM into Weka Environment", **(2005)**.
[14] C.-W. Hsu and C. J. Lin, "A Comparison of Methods for Multiclass Support Vector Machines", IEEE Transactions on Neural Networks, vol.13, **(2002)**, pp.415-425.
[15] C.-W. Hsu, N. Cristianini and C.-J. Lin, "A Simple Decomposition Method for Support Vector Machine", In Machine Learning, Netherlands: Kluwer Academic Publishers, **(2002)**, pp.291-314.
[16] L. Bindings, R. Link and F. Optimization, TinySVM: Support Vector Machines, **(2002)**.
[17] A. Karatzoglou, D. Meyer and K. Hornik, "Support Vector Machines in R. *J. Statistical Software*, vol.15, **(2006)**, pp.1-28.
[18] D. Meyer, E. Dimitriadou, K. Hornik, A. Weingessel and F. Leisch, Package 'e1071', **(2013)**.
[19] O. L. Mangasarian, D. R. Musicant, Lagrangian Support Vector Machine. J. Machine Learning Research, vol.1, **(2001)**, pp.161-177.
[20] A. Ben-Hur, PyML-machine learning in Python, **(2008)**.
[21] G. Fung, S. Ben-David and O. L. Managasarian, "Multicategory Proximal Support Vector Machine Classifiers", In Machine Learning, Netherlands: Springer Science, **(2005)**, pp.77-97.
[22] F. Lauer and Y. Guermeur, "MSVMpack: A Multi-Class Support Vector Machine Package", J. Machine Learning Research, vol.12, **(2011)**, pp.2293-2296.
[23] J. Weston and C. Watkins, "Multi-class support vector machines", In: Technical Report CSD-TR-98-04, **(1998)**.

[24] K. Crammer and Y. Singer, "On the algorithmic implementation of multiclass kernel-based vector machines", J. Machine Learning Research, vol.2, **(2001)**, pp.265-292.
[25] Y. Lee, Y. Lin and G. Wahba, "Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data", J. American Statistical Association, vol.99, **(2004)**, pp.67-81.
[26] Y. Guermeur and E. Monfrini, "A quadratic loss multi-class SVM for which a radius-margin bound applies", J. Informatica, vol.22, **(2011)**, pp.73-96.
[27] N. Gehlenborg, "MAYDAY: Microarray Data Analysis, **(2003)**.
[28] J. Dietzsch, N. Gehlenborg and K. Nieselt, "Mayday-a microarray data analysis workbench", J. Bioinformatics, vol.22, **(2006)**, pp.1010-1012.
[29] I. H. Witten and E. Frank, "Data Mining: Practical Machine Learning Tools and Techniques", San Francisco: Morgan Kaufmann, **(2005)**.
[30] F. Battke, S. Symons and K. Nieselt K. Mayday-integrative analytics for expression data. *J. BMC Bioinformatics*. 11, 1-10 (2010).
[31] Schliep K, Hechenbichler K. The kknn Package. (2006). ftp://ftp.auckland.ac.nz/pub/software/CRAN/doc/packages/kknn.pdf.
[32] Hechenbichler K, Schliep K. Weighted k-nearest-neighbor techniques and ordinal classification. In: *Discussion Paper 399, SFB 386*, (2004).
[33] Schliep K, Hechenbichler K. Package 'kknn'. (2013). http://cran.r-project.org/web/packages/kknn/kknn.pdf.
[34] Wei B, Yang F, Wang X, Ge Y. Package 'knnGarden'. (2013). http://cran.r-project.org/web/packages/knnGarden/knnGarden.pdf.
[35] Venables WN, Ripley BD. Modern Applied Statistics with S Statistics and Computing. Springer, (2002).
[36] Harner EJ, Li S, Adjeroh DA. rknn: an R Package for Random KNN Classification and Regression with Variable Selection. (2012). http://biostat.mc.vanderbilt.edu/wiki/pub/Main/UseR-2012/105-Harner.pdf.
[37] Li S. Random KNN Modeling and Variable Selection for High Dimensional Data. In: *PhD thesis. West Virginia University*, (2009).
[38] Harner EJ, Li S, Adjeroh DA. rknn: an R Package for Parallel Random KNN Classification with Variable Selection. (2013). http://www.edii.uclm.es/~useR-2013/abstracts/files/115_useR2013_prknn.pdf.
[39] Li S. Package 'rknn'. (2013). http://cran.r-project.org/web/packages/rknn/rknn.pdf.
[40] Li S, Harner EJ, Adjeroh DA. Random KNN feature selection-a fast and stable alternative to Random Forests. *J. BMC Bioinformatics*. 12, 450-460 (2011).
[41] Falkenauer E, Marchand A. Using k-Means? Consider ArrayMiner. *Proceedings of the 2001 International Conference on Mathematics and Engineering Techniques in Medicine and Biological Sciences (METMBS'2001), Las Vegas, Nevada, USA*, (2001).
[42] Simon R, Lam A, Li MC, Ngan M, Menenzes S, Zhao Y. Analysis of Gene Expression Data Using BRB-ArrayTools. *J. Cancer Informatics*. 3, 11-17 (2007).
[43] Buturovic LJ. PCP: a program for supervised classification of gene expression profiles. *J. Bionformatics*. 22, 245-247 (2006).
[44] Ripley B. Pattern Recognition and Neural Networks. Cambridge, (1996).
[45] Ripley B. Package 'nnet'. (2013). http://cran.r-project.org/web/packages/nnet/nnet.pdf.
[46] Fritsch S, Gunther F. neuralnet: Training of Neural Networks. *R Foundation for Statistical Computing*. (2008).
[47] Gunther F, Fritsch S. neuralnet: Training of Neural Networks. *The R J*. 2, 30-38 (2010).
[48] Fritsch S, Gauenther F. Package 'neuralnet'. (2013). http://cran.r-project.org/web/packages/neuralnet/neuralnet.pdf.
[49] Specht DF. Probabilistic neural networks. *Neural networks*. 3, 109-118 (1990).
[50] Chasset P-O. Package 'pnn'. (2013). http://cran.r-project.org/web/packages/pnn/pnn.pdf.
[51] Bergmeir C, Benitez JM. Neural Networks in R Using the Stuttgart Neural Network Simulator: RSNNS. *J. Statistical Software*. 46, 1-26 (2012).
[52] Bergmeir C, Benitez JM. Package 'RSNNS'. (2013) http://cran.r-project.org/web/packages/RSNNS/RSNNS.pdf.
[53] Yeung K, Bumgarner R, Raftery AE. Bayesian Model Averaging: Development of an Improved Multi-Class, Gene Selection and Classification Tool for Microarray Data. *J. Bioinformatics*. 21, 2394-2402 (2005).
[54] Yeung KY, Raftery A, Painter I. The Iterative Bayesian Model Averaging (BMA) algorithm. (2009). http://www.bioconductor.org/packages/ 2.11/bioc/manuals/iterativeBMA/man/iterativeBMA.pdf.
[55] Su L, Zhang H. Full Bayesian Network Classifiers. *Proceedings of the Twenty-Third International Conference on Machine Learning*. 897-904 (2006).
[56] Lamnisos D, Griffin JE, Steel MFJ. Transdimensional sampling algorithms for Bayesian variable selection in classification problems with many more variables than observation. *J. Computational and Graphical Statistics*. 18, 159-612 (2009).

[57] Jun AY, Yuan SX. Bayesian variable selection for disease classification using gene expression data. *J. Bioinformatics*. 26, 215-222 (2009).
[58] Rosen GL, Reichenberger ER, Rosenfeld AM. NBC: the Naïve Bayes Classification of metagenomic reads. *J. Bioinformatics*. 27, 127-129 (2011).
[59] Guo Y, Hastie T, Tibshirani R. Regularized linear discriminant analysis and its application in microarray. *J. Biostatistics*. 8, 86-100 (2007).
[60] Clemmensen L, Hastie T, Ersboell K. Sparse discriminant analysis. In: *Technical report, IMM, Technical University of Denmark*, (2008).
[61] Gschwandtner M, Filzmoser P, Croux C, Haesbroeck G. A Robust Approach to Regularized Discriminant Analysis. (2011). http://www.stat.tugraz.at/Statistiktage2011/Gschwandtner.pdf.
[62] Díaz-Uriarte R, Alvarez DAS. Gene selection and classification of microarray data using random forest. *J. BMC Bioinformatics*. 7(3), (2006).
[63] Díaz-Uriarte R. GeneSrF and varSelRF: a web-based tool and R package for gene selection and classification using random forest. *J. BMC Bioinformatics*. 8, (2007).
[64] Saffari A, Leistner C, Santner J, Godec M Bischof H. On-line Random Forests. *3rd IEEE ICCV Workshop on On-line Computer Vision*. (2009).
[65] Hothorn T, Hornik K, Strobl C, Zeileis A. party: A Laboratory for Recursive Partytioning. (2011) http://cran.stat.auckland.ac.nz/web/packages/party/vignettes/party.pdf.
[66] Deng H, Runger G. Gene Selection with Guided Regularized Random Forest. In: *Technical Report*, (2012)
[67] Lim A, Breiman L, Cutler A. Big Random Forest: Classification and Regression Forests for Large datasets. (2013). http://cran.rproject.org/ web/packages/bigrf/bigrf.pdf.

