Machine Learning for Functional Genomics: Some Experiments with Supervised Learning on Microarray Data Set

T. Curk[1], B. Zupan[1,2,3], U. Petrovic[2], J. Demsar[1], G. Shaulsky[3],
L. Sacchi[4], C. Larizza[4], R. Bellazzi[4]

[1] Department of Computer Science, University of Ljubljana, Slovenia
[2] J. Stefan Institute, Ljubljana, Slovenia
[3] Department of Human and Molecular Genetics, Baylor College of Medicine, Houston, USA
[4] Dipartimento di Informatica e Sistemistica, Università di Pavia, Italy

Abstract

In the paper we study the application of various supervised machine learning techniques to induce classification models for gene function assignment based on their expression profiles. We show that a simple naïve Bayesian classifier may perform comparably to a more complex method of support vector machines, which has recently gained much attention in the field. Through experimental analysis on various data sets we also show that classification tasks considered may be hard and, despite abundance of publicly available gene expression data sets, we could not derive reliable predictors for some of gene functions modelled.

1 Introduction

In the post-genomic era, functional genomics is the discipline that attempts to assign function to all the genes in a genome. For function assignment, much of recent research in bioinformatics has focused on the analysis of DNA microarray data. A typical microarray assay provides data on the expression of a large number of genes under multiple conditions or at different time-points of some biological experiment. The data can be represented as a matrix in which rows correspond to genes, columns correspond to conditions or time-points and entries represent level of gene expression relative to that of a reference, e.g. wild-type organism under standard conditions.

At present, the most common technique for microarray data analysis is clustering,[1, 2] which is used to find groups of genes with similar expression profiles. When clustering is used for functional analysis, the prevailing idea is that genes with correlated expressions are functionally associated. Assignment of function to individual genes is done by applying the prevailing function of the genes in the shared cluster to the uncharacterized genes. Clustering is an unsupervised method and does not take into account any information on the functional labels of genes, which may well be available for a subset of genes. In this respect, supervised learning methods may be used to devise gene function prediction models. They may perform favourably to clustering,[3] as they do not rely on a particular distance matrix, may propose probabilities for functional memberships, and may also handle cases where genes belong to several functional groups. Also, evaluation mechanisms for supervised models based on random sampling are well established, and can be used to clearly asses their predictive value.

Within functional genomics, supervised methods may therefore be used to induce prediction models from functionally annotated microarray data sets. Induced models can then predict functions for genes that have not been functionally annotated, or may even detect genes with erroneous annotation. Several authors demonstrated the utility of supervised learning methods in the prediction of gene function. For instance, Brown et al.[4] compared several techniques and concluded that support vector machines performed best in building gene function prediction models for yeast. Hvidsten at al.[3] used rule-base induction approaches to perform a similar task. With the abundance of microarray data and available gene function annotations for several organisms, the utility of supervised methods

has though at best been sparse and we believe there is still a range of open issues to be studied. Those addressed in this paper include:

1. Brown et al.[4] clearly demonstrate that support vector machines (SVM) outperform a range of other machine learning methods. We hypothesize that, with some parameter tuning, other methods may perform as well and may additionally better reveal how particular input attributes are related to a functional class. For instance, a naïve Bayes method we use in our study may compute the conditional probability of outcome given the value of an attribute (see, for instance, Figure 1).
2. When there is a range of microarray data sets for the same genes but coming from an independent biological experiment there is a question whether this should be treated as a combined data set (as in Brown et. al.[4]) or used separately (choose the most informative data set for specific function, or combine separately derived classifiers in an ensemble).
3. Microarray data are often comprised of gene expression levels at distinct time points. To which extent can feature construction based on temporal analysis of expression profiles augment the existing data sets and thus enhance the performance of classifiers?

2 Data Sets

We used two sources of data for gene expression in *S. Cerevisiae* (yeast). The first data set (for convenience labelled as Brown et al.) includes 2467 genes annotated with one of six functional classes, which were derived from previously published clustering of the same data set.[1] Described through 79 attributes, gene expression measurements were taken at various time points during the diauxic shift, the mitotic cell division cycle, sporulation, and temperature and reducing shocks. Three other data sets were obtained from particular publications on yeast (http://www.transcriptome.ens. fr/ymgv/). From these, we selected eight studies (Causton_-stress, Cho_mitotic, Chu_sporulation, DeRisi_-metabolic, Gasch_stress, Hardwick_rapamycin, Spellman_cellcycle, Zhu_forkheads) each including several time series data sets. Overall, the selected 34 time series have an average 8.7 time points, yielding a total of 296 time points for all the 6957 genes. No particular normalization/filtering

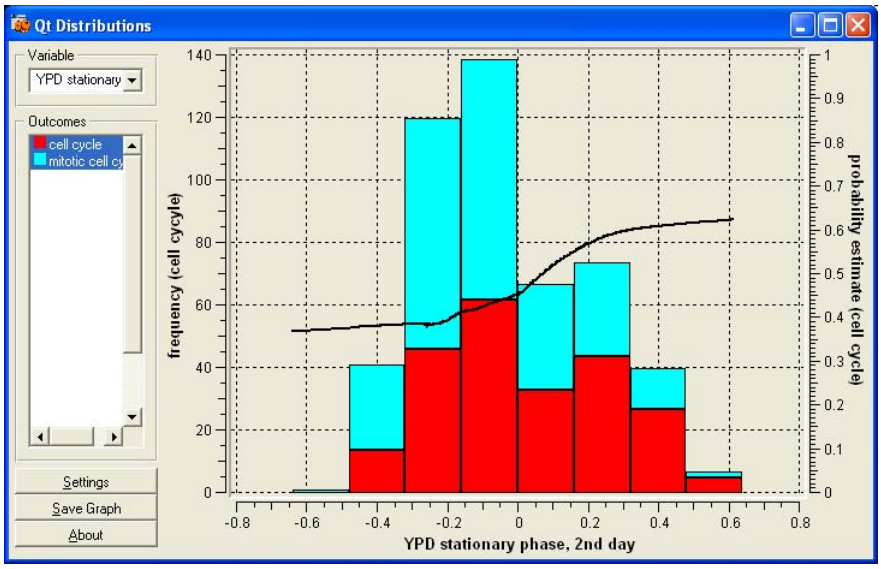

Figure 1. A histogram showing the number of genes in cell cycle and mitotic cell cycle given the value of gene expression in experiment where stationary phase cells where analyzed in different time point after 2 days of growth. Estimated conditional probability of the gene belonging to the cell cycle functional group (vs. mitotic cell cycle functional group) computed through loess estimation is also shown (smooth curve in the middle of the graph, scale on the right).

methods besides those already applied in original data sets were used. Time series data sets were then merged into a single data set.

Annotation data from the Gene Ontology Project (GO, www.geneontology.org/) were used to functionally annotate the genes. The participating geneticist (UP) selected three pairs of biological processes for which a prediction model would be potentially useful and non-trivial. The selected processes are: "cell cycle" (258 genes) vs. "mitotic cell cycle" (290 genes), "regulation of transcription" (277 genes) vs. "transcription" (264), and "response to osmotic stress" (54 genes) vs. "response to stress" (171 genes). The final three data sets contained the expression data only for the genes belonging to the selected biological processes. Note that the three problems are of the kind process vs. sub-process, for example: functional group 'mitotic cell cycle' in GO is only a part of the functional group 'cell cycle.' In case a gene in GO was annotated to both processes, the more specific (in our example 'mitotic cell cycle') was kept.

3 Methods

We report on the use of two different methods, support vector machines (*SVM*) and naïve Bayesian classifier.[5] For SVM, we have used a second-order polynomial kernel. Other parameters were set to match those from the study of Brown et al.[4] as closely as possible. Naïve Bayesian classifier, a much simpler classifier that assumes conditional independence of attributes given the class value, computes the probability of the outcome as a product of conditional class probabilities given the values of each attribute.

To assess these conditional probabilities used in naïve Bayesian Classifier, we have either used attribute discretization and an *m*-probability estimate[6] that extends the relative frequency formula and may better deal with noise (*dBayes*), or assessed probabilities directly through loess-based approximation (Figure 1). For discretization we have used an entropy and minimal description length-based algorithm by Fayyad and Irani.[7] For each attribute independently, their method finds the appropriate cut-off points such that the class (gene label) entropy within each resulting interval is minimized while balancing this with introducing as few cut-off points as possible. Such adaptive search for discretization intervals may have advantage over less robust methods (like using a constant number of cut-off points at predefined values of attributes), and may not have problems with skewed distributions. Using *m*-probability estimate helps in situations with small number of cases by adjusting conditional probabilities as estimated from the data through relative frequency towards unconditional (prior) probability determined from a complete data set. The approach can be of particular value when data is noisy. The value of the parameter m was tuned through internal 5-fold cross validation over a range of candidate values.

For some attributes, entropy-based discretization may not find any suitable cut-off point, thus reducing a continuously-valued attribute to a constant and as such removing it from the data set. This implicit attribute subset selection method was paired with explicit one for naïve Bayesian classifiers where smooth probability estimates were used as coming from loess-based approximation. The particular attribute subset selection method we have used was based on ReliefF measure of attribute relevancy,[8] where attributes with relevancy below or equal to zero were removed (*bayesFSS*).

Our data sets are composed of data from different experiments, i.e. consisting of a set of data (attribute) subsets. We hence tried to induce models from each of these subsets and combine them with product rule (multiplication of class probabilities assessed by individual models – *bayesEns* and *dBayesEns*). Also, a Temporal Abstraction clustering method[9] was applied to generate an additional attribute for each time series. The additional attribute is a label for a particular temporal pattern found in gene expression profiles. A learning set was used to find a set of candidate temporal patterns, which were then used to provide additional attributes to examples in the test set.

For comparison and as a baseline, a majority classifier (*majority*) is included in experiments. It classifies genes to the most frequent functional class from the training data set. Obviously, any useful prediction model should significantly outperform a majority classifier.

The testing schema for the learning algorithms was that of 10-fold cross validation. There, data were split to ten subsets of approximately equal size and class distribution. Iteratively, one subset was left for testing of the models that were induced

from the remaining nine subsets. Statistics (classification accuracy) were averaged across ten iterations. Since the particular version of support vector machine (libsvm, http://www.csie.ntu.edu.tw/~cjlin/libsvm/) is unable to model class probabilities, we here report only on classification accuracy.

4 Preliminary Experimental Results

Table 1 shows classification accuracies from cross validation study. Average classification accuracy (CA) and the method's rank across 10 evaluation experiments are shown. SVM performs best only (!) on the Brown et al. data set, and is outperformed by variants of naïve Bayes in three other domains. Results, though, are somehow acceptable for Transcription and Cell cycle, whereas for Stress the modelling failed.

We were interested to know the contribution of a particular data subset to the accuracy prediction within the same domain. To study the role of individual data subsets, cross validation was performed on each subset separately. Modelling with SVM and dBayes was used. Results (Table 2, Brown et al. and Cell cycle data) clearly point out that selection of a particular experiment set may well influence the performance, and that SVM (in particular) gains in appropriate selection. Notice also that while SVM performed best on a particular data set, dBayes induced more top-ranked models overall. The same was observed on the Transcription data set, while on Stress all methods failed on all data sets.

Somewhat unexpectedly, constructive induction through Temporal Abstraction did not improve the results (particular results are not reported in Tables). We cannot explain the reason for this observation and further work is needed to analyze this.

Table 1. Performance evaluation (CA, classification accuracy) on various data sets.

	Data Sets							
	Brown et al.		Cell cycle		Transcription		Stress	
Learner	CA	Rank	CA	Rank	CA	Rank	CA	Rank
SVM	0.970 ± 0.003	1.00	0.529 ± 0.007	5.15	0.575 ± 0.017	4.90	0.747 ± 0.013	3.70
bayesEns	0.955 ± 0.003	2.65	0.631 ± 0.014	1.55	0.704 ± 0.018	2.25	0.760 ± 0.006	3.15
bayesFSS	0.953 ± 0.004	2.85	0.619 ± 0.014	2.10	0.702 ± 0.018	2.30	0.697 ± 0.020	4.70
dBayes	0.946 ± 0.002	3.50	0.595 ± 0.018	3.25	0.695 ± 0.018	2.60	0.760 ± 0.006	3.15
dBayesEns	0.931 ± 0.003	5.00	0.576 ± 0.013	3.60	0.697 ± 0.022	3.00	0.760 ± 0.006	3.15
majority	0.907 ± 0.000	6.00	0.529 ± 0.001	5.35	0.512 ± 0.003	5.95	0.760 ± 0.006	3.15

Table 2. Best models on specific attribute subsets from Brown et al. data (left) and Cell cycle data (right)

Learner	Subset	CA	Rank
SVM	spo	0.941 ± 0.002	1.30
dBayes	spo	0.938 ± 0.002	1.90
dBayes	heat	0.923 ± 0.003	4.85
dBayes	cdc	0.919 ± 0.003	6.75
dBayes	diau	0.918 ± 0.005	6.65
dBayes	ddt	0.913 ± 0.006	9.90
dBayes	alpha	0.911 ± 0.005	9.80
dBayes	elu	0.909 ± 0.003	9.70
majority		0.907 ± 0.000	12.15

Learner	Subset	CA	Rank
SVM	Causton_stress_NaCl	0.615 ± 0.023	24.9
dBayes	Spellman_cellcycle cdc28	0.608 ± 0.017	20.9
dBayes	Gasch_stress 29Csorb33Cnosorb	0.604 ± 0.022	31.15
SVM	Spellman_cellcycle cdc28	0.602 ± 0.011	20.15
SVM	Gasch_stress Nitrogen	0.595 ± 0.017	27.05
…			
majority		0.529 ± 0.001	64.6

5 Discussion and Future work

The experimental results show a mixed success of building gene function prediction models from microarray data. But even with some positive results, one should exercise caution. Our models performed best on a yeast data set first used for supervised learning by Brown et al.[4] But notice that class labels for this data set came from clustering, so model prediction on this domain is actually an attempt to rediscover the distance function used for clustering. As these are in general not too complex functions (e.g. Euclidian distance), it is somehow expected for machine learning to perform reasonably well. Recently, Hvidsten et al.[3] have reported on a machine learning study where gene labels, like for the other three data sets, where obtained from Gene Ontology (GO). They use a single smaller data set, but obtain much better results than those reported in this paper. The reason may lay in particular selection of a classification problem, e.g., in particular functional labels of genes selected for modelling. Hvidsten and collaborators used functional labels from higher levels of GO. These functions are more general than those modelled in this paper, and also bear less similarity. It seems that biological problems addressed in our experiments are harder: for instance, genes in transcription and regulation of transcription may be highly co-regulated). Also, the choice of particular data set (biological experiment) is crucial, and as expected, different data sets should be considered for prediction of distinct gene functions.

The GO annotation of the yeast genome is probably the best there is for any eukaryotic genome, but it is not perfect. In addition, the 'function' annotation of genes is not always sufficient for accurate categorization and the other two annotations, 'process' and 'subcellular localization' are necessary for complete annotation. Therefore, the GO annotation data set cannot be considered as unambiguous. Similarly, microarray data are notoriously noisy so the classification that is based on them must be imperfect. It is quite possible that some of the failures reported here can be explained as the result of these ambiguities. We predict that similar attempts to correlate expression and function in other, less well annotated genomes are likely to have a higher failure rate.

A few comments are in order regarding machine learning as well. (1) Naïve Bayesian classifiers when combined with an optimization method that finds an appropriate function (m-estimate) for probability assessment seems to be doing just as well as support vector machines, while being computationally significantly less demanding. (2) Since performance on some data sets were rather poor, ensemble learning did not help. (3) Constructive induction by means of extracting temporal patterns (surprisingly) did not help. (4) Notice that while the data is coming from different sources with different experimental conditions, this should not affect the performance of particular classification methods used. Naïve Bayesian classifier, for instance, treats each attribute independently; attribute-specific conditional probabilities it uses are derived without using the data on other attributes.

The results reported here are preliminary: we have found several problems, and were surprised by the rather weak performance of some advanced machine learning methods. To find what exactly caused these problems and what can be done to alleviate them we need to further analyze these domains. In this, we are now relying on a combination of machine learning and visualization tools.

References

1 M. B. Eisen, P. T. Spellman, P. O. Brown, et al., Proc Natl Acad Sci U S A **95**, 14863 (1998).

2 V. R. Iyer, M. B. Eisen, D. T. Ross, et al., Science **283**, 83 (1999).

3 T. R. Hvidsten, A. Laegreid, and J. Komorowski, Bioinformatics **19**, 1116 (2003).

4 M. P. Brown, W. N. Grundy, D. Lin, et al., Proc Natl Acad Sci U S A **97**, 262 (2000).

5 T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning* (Springer, New York, 2001).

6 B. Cestnik, in *9th European Conference on Artificial Intelligence*, 1990), p. 147.

7 U. M. Fayyad and K. B. Irani, in *13th Int'l Joint Conf. on Artificial Intelligence* (Morgan-Kaufmann, 1993).

8 I. Kononenko, E. Simec, and M. Robnik-Sikonja, Applied Intelligence **7**, 39 (1997).

9 L. Sacchi, R. Bellazzi, C. Larizza, et al., in *IDAMAP 2003, this volume.*, 2003).