Supervised learning of microarray gene expression data

CHONG SHOU

Program of Computational Biology and Bioinformatics, Yale University

ABSTRACT

We compare several supervised learning methods to Support Vector Machine method to expression microarray data. Four different kernel functions for SVM are tested as well as Decision Tree, Neural Networks and Naïve Bayes methods. Techniques used in SVM training by *Brown et al.* are used and the results are compared. Best performed SVMs are then used to predict functional roles for previously un-annotated ORFs based on their expression data, and some of them are verified by recent available experimental results.

INTRODUCTION

DNA microarray

A DNA microarray is a collection of microscopic DNA spots. Commonly, each spot is a single strand DNA sequence, representing a single gene, printed on a glass by covalent attachment. Quantitative measurements of the concentration of DNA sequences in a solution could be obtained by DNA-DNA or DNA-RNA hybridization under high-stringent conditions and fluorescence based detection. Thus, biologists could use DNA microarrays to measure the expression level of thousands of interested genes in a single experiment. This is called the expression profile, which gives expression levels under a certain biological condition for a whole set of genes. Expanding this by testing expression levels under a series of biological conditions could generate a matrix of data that monitors expression profiles through time.

Data description

Systematic experiments of gene expression profiles of yeast *S. cerevisiae* are performed by using DNA microarrays. 6221 genes with clear ORFs (Open Reading Frames) are tested in the experiments, and 81 biological conditions are used for diverse expression profiles, including different time points after exposing the organism to a certain environment. One biological condition is used as a reference condition, so that expression levels under all other conditions could be presented by the ratio of the actual expression level to the reference expression level. Thus, the whole dataset is a 6221 by 80 matrix, with each data point representing the expression level ratio of a certain gene under a certain condition. Biological meaning of the ratio represents the regulation of gene expression. If the ratio is larger than 1, then it is up-regulated compared to the reference condition; if it is smaller than 1, then it is down-regulated.

Instead of using the ratio, which is always larger than zero, normalized logarithm is used for analysis suggested by *Eisen et al*. It then makes down-regulated expressions with ratio in (0,1) to $(-\infty,0)$, and up-regulated expressions with ratio in $(1,+\infty)$ to $(0,+\infty)$, in order to maintain symmetry. Define X_i to be the logarithm of the ratio of expression level E_i for gene X in experiment i to the expression level R_i of gene X in the reference state, normalized so that the expression vector $\overrightarrow{X} = (X_1,\ldots,X_{80})$ has Euclidean length 1: $X_i = \dfrac{\log(E_i/R_i)}{\sqrt{\sum_{j=1}^{80}\log^2(E_j/R_j)}}$.

The natural basis for organizing gene expression data is that genes involved in the same biological process or carrying out closely related functions should have similar patterns of expression. Thus, genes could be clustered according to the similarity of expression patterns, and a function-unknown gene with a similar expression pattern with a function-known gene, could be predicted with certain confidence to be involved in the same biological process or to have a related function.

Supervised learning

Having samples that we will try to learn the pattern from, there are two general classes of learning methods. *Eisen et al.* used unsupervised learning method to cluster the microarray expression data. Without any biological knowledge of the function or biological process labels of those genes, they did clustering analysis solely based on similarities between two samples.

Although unsupervised clustering seems to successfully uncover some underlying gene clusters, the biological significance of those clustering are not clear. Thus, it is always useful to incorporate known biological knowledge into the learning process. Here, we use the biological knowledge to label the samples for the training set, and building our models upon that.

METHODS

We will use several supervised learning methods in this paper applying to this particular dataset, and compare the performances of each method.

Brown et al. suggested that SVM outperforms other supervised learning methods on this dataset. Here, we will try to reproduce the result they provided, and also try some other supervised learning methods to compare the performance. This paper does not aim to find a best model for this dataset, yet to understand the nature of those commonly used supervised learning methods.

We have downloaded the whole microarray expression dataset from the Stanford website (http://rana.stanford.edu/clustering). Format is adjusted to fit our analysis. 2467 genes with good annotations are selected to form the training set as *Eisen et al.* suggested. Among the rest 3754 genes, 186 genes without clear expression profiles are discarded, which results in a 3568-gene test set.

Class labels for the training set are obtained from GO (Gene Ontology, http://www.geneontology.org). Six classes are selected for classification and prediction. Four are biological processes: respiration, tricarboxylic acid cycle (TCA), meiosis and proteolysis; two are cellular components: histone and ribosome. Classes are selected such that biologically, the possibility of being classified to two or more classes is small.

Additional dataset pre-processing is required for performing SVM. Individual dataset is needed for each of the six classes, so that the genes within the class will be labeled positive, and all others negative. Thus, the SVM method could be evaluated for each class.

Three-fold cross validation is used for model evaluation. The training set is split into three parts, and models are training using every combination of two parts, and making the prediction for the third part. Model evaluation is based on the fitness of predicted labels and the actual labels.

Other supervised learning methods are also used to test their performance, which include Decision tree, Naïve Bayes, and Neural networks. Principle component analysis is also performed, trying to reduce the dimension of feature space.

The performance of a learning method is evaluated by the index of savings, as suggested by *Brown et al*. We define the cost function by $C(\mathrm{M})=fp(M)+2\cdot fn(M)$, where $fp(M)$ and $fn(M)$ are the numbers of false positives and false negatives for method *M*. Also define $\mathrm{C}(N)$ to be the cost value such that all samples are classified as negative. And define save function $S(M)$ as $\mathrm{S}(M)=C(N)-C(M)$. Thus, larger value of the save function indicates a better model.

Predictions of unknown functions for the test set are made by SVM alone. Top predictions are examined in detail for biological verification.

RESULTS

1. SVM (Support Vector Machines)

We use the software GIST for performing SVM in this paper, developed by *Noble et al.* (http://bioinformatics.ubc.ca/gist/). It is installed in Linux system and performed SVM training locally.

Four kernel functions are used for SVM: linear kernel $K(X,Y)=\vec{X}\cdot\vec{Y}+1$, 2-power polynomial kernel $K(X,Y)=\left(\vec{X}\cdot\vec{Y}+1\right)^2$, 3-power polynomial kernel $K(X,Y)=\left(\vec{X}\cdot\vec{Y}+1\right)^3$ and radial kernel $K(X,Y)=\exp\left(-\sigma\,\|\,X-Y\,\|^2\right)$. However, for each class, samples labeled to this class are only a small fraction of the whole dataset, which leads to an imbalance in the number of positive and negative training examples. Thus, the positive samples are more likely to be considered as noise, which cause the SVM to make incorrect classifications. *Brown et al.* suggested adding to the diagonal of the kernel matrix a constant so that avoiding the positive sample to be considered as noises. For positive samples X_i, $K'(i,i)=K(i,i)+\lambda\left(n^+/N\right)$; for negative samples X_i, $K'(i,i)=K(i,i)+\lambda\left(n^-/N\right)$, where *N* is the total number of samples, and n^+ and n^- are the number of positive samples and negative samples, respectively. λ is set to 0.1.

We first use all the samples in the training set without cross validation to train the SVM, in order to check the fitness of the kernel functions. And the models are then used to the test set for prediction.

Table 1: Fitness of kernel functions to the dataset.

Class	Method	FP	FN	TP	TN	S(M)
TCA	Linear SVM	8	0	14	2445	20
	Power-2 SVM	0	0	14	2453	28
	Power-3 SVM	0	0	14	2453	28
	Radial SVM	0	0	14	2453	28
Proteolysis	Linear SVM	29	0	8	2430	-13
	Power-2 SVM	0	0	8	2459	16
	Power-3 SVM	0	0	8	2459	16
	Radial SVM	0	0	8	2459	16
Meiosis	Linear SVM	123	0	17	2327	-89
	Power-2 SVM	0	0	17	2450	34
	Power-3 SVM	0	0	17	2450	34
	Radial SVM	0	0	17	2450	34
Respiration	Linear SVM	199	0	35	2233	-129
	Power-2 SVM	5	0	35	2427	65
	Power-3 SVM	0	0	35	2432	70
	Radial SVM	0	0	35	2432	70
Histone	Linear SVM	261	1	26	2179	-209
	Power-2 SVM	0	0	27	2440	54
	Power-3 SVM	0	0	27	2440	54
	Radial SVM	0	0	27	2440	54
Ribosome	Linear SVM	118	2	170	2177	222
	Power-2 SVM	10	0	172	2285	334
	Power-3 SVM	2	0	172	2293	342
	Radial SVM	0	0	172	2295	344

Table 1 indicates the training fitness of kernel function to the dataset. And Power-3 and Radial kernels are both good performers.

We then used Power-3 polynomial kernel to test the performance of SVM by 3-fold cross validation.

Table 2: Prediction performance comparison of Power-3 SVM to *Brown et al.* (Meiosis class are not used by *Brown et al.*)

Class	Power-3 SVM	FP	FN	TP	TN	S(M)
TCA	*Chong*	2	13	1	2451	0
	Brown et al.	4	9	8	2446	12
Proteolysis	*Chong*	0	8	0	2459	-6
	Brown et al.	3	8	27	2429	51
Meiosis	*Chong*	4	17	0	2456	-4
	Brown et al.	/	/	/	/	/
Respiration	*Chong*	18	30	5	2414	-8
	Brown et al.	6	8	22	2431	38
Histone	*Chong*	5	27	0	2435	-5
	Brown et al.	0	2	9	2456	18
Ribosome	*Chong*	33	41	131	2262	229
	Brown et al.	7	3	118	2339	229

The performance of SVM is not very good for sparse positive-sample datasets, even if we have modified traditional kernel function and kernel matrix. For the class Ribosome, it has 172 positive labels within 2467 samples, the faction of positive samples is larger, thus the performance of SVM is the best. However, other classes do not have a large positive sample fraction, which makes the performance not satisfied.

2. Decision Tree

Decision tree method is a classic classification method. We use the function J48 in software Weka to build the model (http://www.cs.waikato.ac.nz/ml/weka/). 3-fold cross validation is also performed to the training set. The tree ended up with 44 leaves.

Table 3: Confusion matrix of 3-fold cross validation of Decision Tree method.

Classified as →	TCA	Proteolysis	Meiosis	Respiration	Histone	Ribosome	Other
TCA	**1**	0	1	3	0	0	9
Proteolysis	0	**0**	0	0	0	0	8
Meiosis	0	0	**1**	0	0	0	16
Respiration	0	0	0	**1**	0	2	32
Histone	0	0	0	0	**0**	0	27

Ribosome	0	0	0	0	0	**114**	58
Other	9	3	3	13	1	35	**2130**

Still, the class Ribosome with larger fraction of positive labels is better predicted than other classes. And compared to SVM, Decision tree method is not a good performer, either.

3. Naïve Bayes

Naïve Bayes is a simple probabilistic classifier based on applying Bayes' theorem with naïve independence assumptions. It has the form as the following:

$p(\mathrm{C} \mid F_1,\ldots,F_n) = \frac{1}{Z} p(C) \prod_{i=1}^{n} p(F_i \mid C)$, where Z is a constant scaling factor based on the features F_i, $i \in (1,n)$; and C is the dependent class variable. Thus the classifier is defined to maximize this conditional probability.

Naïve Bayes has an advantage of dealing with high dimension feature space, since it assumes that features are independent, and the conditional probability is just a product concerning all features. Model is trained in Weka, using NaiveBayes function.

Table 4: Confusion matrix of 3-fold cross validation of Naïve Bayes method.

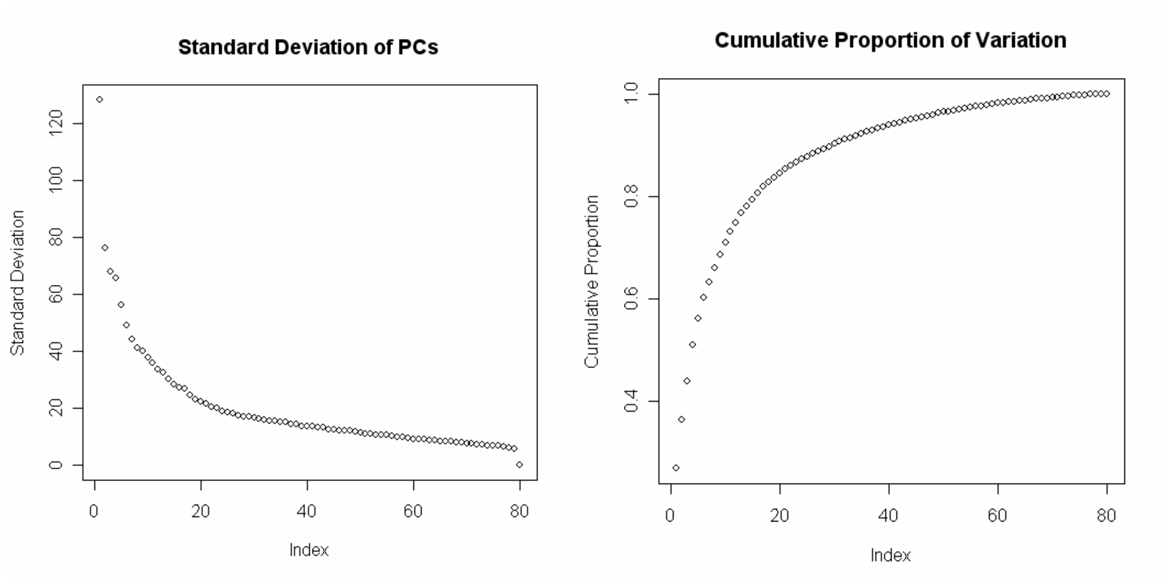
Classified as →	TCA	Proteolysis	Meiosis	Respiration	Histone	Ribosome	Other
TCA	**9**	0	0	2	0	0	3
Proteolysis	0	**0**	0	0	1	0	7
Meiosis	0	0	**5**	1	4	0	7
Respiration	6	0	1	**10**	2	2	16
Histone	0	0	4	1	**7**	0	15
Ribosome	0	0	0	36	0	**120**	16
Other	22	8	76	159	365	92	**1472**

Classification by Naïve Bayes has more true positives while also has more false positives, compared to SVM and Decision Tree methods.

4. Feature selection and PCA (Principle Component Analysis)

Because the dimension of the feature space is high, we try to reduce the dimension by feature selection. Greedy forward and backward stepwise feature selection is performed to the training set. We use Weka data preprocessing function, and choose the supervised attribute selection, greedy stepwise method to perform the task. 22 features are selected out of 80 features.

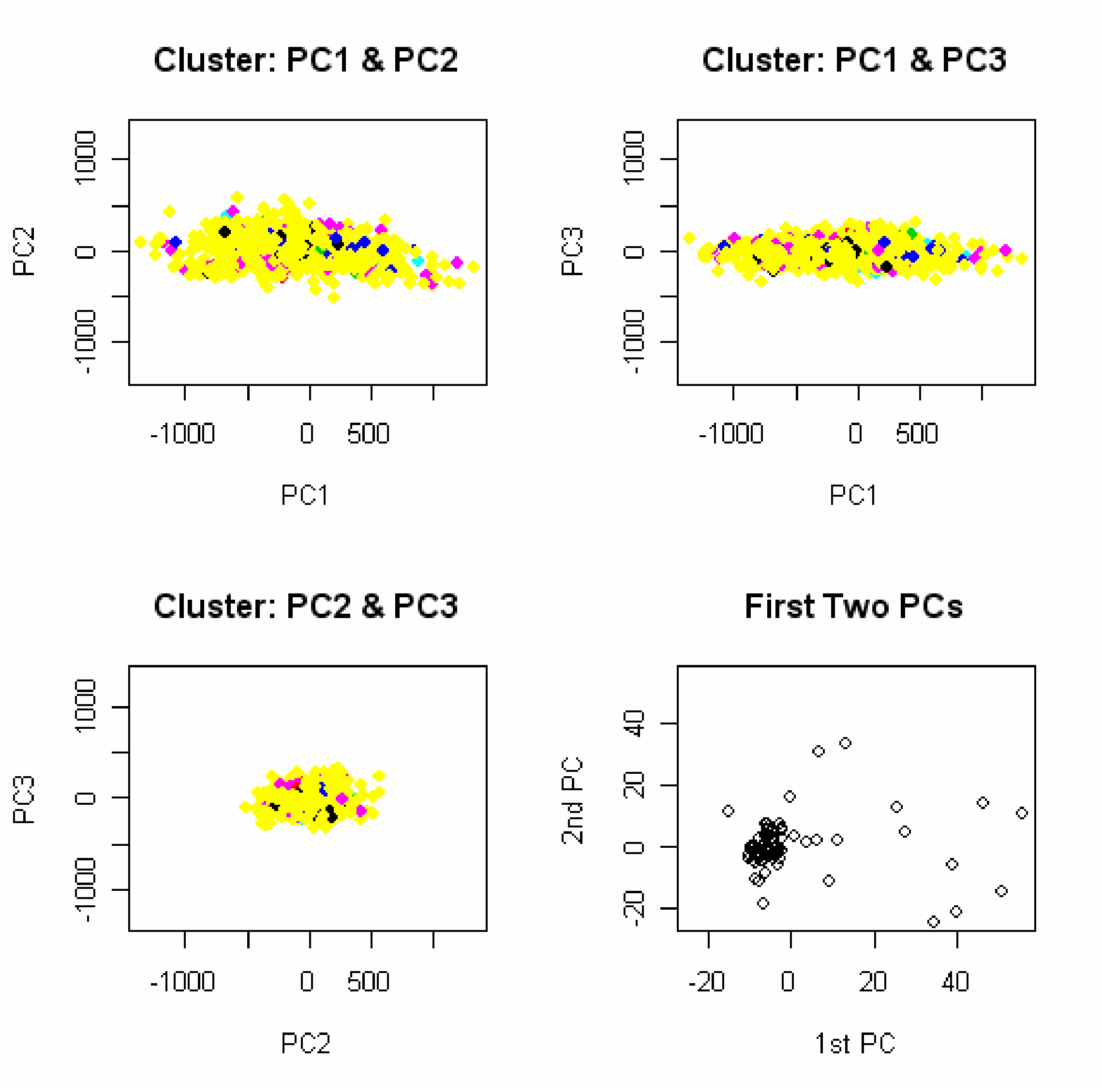
Figure 1: PCA: Standard deviation of principle components and cumulative proportion of variation of ranked principle components.



In fact, 80 conditions used in the original dataset are of different time points when the organism is exposed to 8 different environments. And within the same environment, the conditions may have some redundancy, and depend on its previous time point condition in a certain extent. Thus, feature selection could reduce the dimension of the feature space while maintaining a majority of variation.

We also tried to use PCA to the whole dataset, using the “svd” function in R. Seen from Figure 1, first 30 PCs are responsible for about 90% of variation of the dataset. And the standard deviation tends to be linearly decreasing after 30 PCs.

Figure 2: Class spread according to pairs of three top-ranked principle components, and comparison of top two principle components.



We also looked into the first two PCs, and try to identify some useful patterns indicating the differences between them. Shown in Figure 2, one dense area is at the (0,0) centered area, indicating the majority of PC values are similar in two PCs. However, some other features in PC do have large variations, indicating large variations within the data points for these features.

Colored plots are shown, indicating how data points of 7 classes (6 functional classes and 1 class for all other functions) spread according to the pairs of first 3 PCs. Yellow points are labeled with “other functions”. The problem remains that the labeled points are only a few with respect to the great amount of data points. Thus the pattern is not clear.

5. Neural Networks

We also tried Neural Networks method to the dataset. Because the learning time is extremely long if using the whole dataset with 80 features, we only use 22 features that are selected after feature selection. The Neural Network is training by the function "Multilayer Perceptron" in Weka, which uses Backpropagation algorithm.

The network does not have feedback loops, and it has many layers during the information flow. Each neuron is a linear combination of the 22 features, and it has a threshold for output. Within iteration, the algorithm calculates the error using mean-square error, and tries to minimize the error according to the labels. The confusion matrix after 3-fold cross validation is shown in Table 5.

Table 5: Confusion matrix of 3-fold cross validation of Neural Networks method.

Classified as →	TCA	Proteolysis	Meiosis	Respiration	Histone	Ribosome	Other
TCA	**5**	0	0	1	0	0	8
Proteolysis	0	**0**	0	0	0	0	8
Meiosis	0	0	**0**	0	0	0	17
Respiration	5	0	1	**1**	2	6	23
Histone	0	0	0	0	**0**	1	26
Ribosome	0	0	0	1	0	**137**	34
Other	8	1	2	12	2	29	**2140**

The performance of Neural networks is close to that of decision tree. It is not capable to find a small fraction of positive samples out of an extremely large sample space, e.g. for the class Proteolysis. However, it fits well if the positive sample size is large enough to improve the prediction stability and accuracy, e.g. for the class Ribosome.

DISCUSSIONS

1. Models evaluation

We first tried to reproduce the result by *Brown et al.* of SVM model, however, our model is not good enough to identify small fraction of positive samples out of the large dataset. Table 2 shows that for the Histone class in *Brown et al.*, 11 samples are labeled as positive out of 2467 samples in total. The SVM model successfully identified 9 samples while missing only 2 samples, and without any false positives. The reason might be that we are missing some important parameters that should be adjusted, which did not implied by the original paper.

Among the four supervised learning methods that we used for this analysis, none of them successfully identified the majority of positive samples when the positive sample size is small. SVM is a little better than the other three methods, Naïve Bayes, Decision Tree and Neural Networks in finding more true positives while maintaining a relatively low false positive rate. Naïve Bayes method is better to find positive samples, however, it also increases false positives while reduce false negatives. Decision tree and Neural networks are neither good performers for this dataset.

Principle component analysis does not discover clustering of classes. Figure 2 shows that the positive data points are primarily spread out in the dataset; although principle components give the data points a normalized shape.

2. Function prediction

3-power polynomial kernel and radial kernel SVM models trained by the whole training set are used to predict function labels for the test set. Table 6 lists the genes that are predicted to be class members by both SVMs, and the objective function value from both models are larger than 0.1. For Ribosome class, the threshold is set to be 1.0 to select top hits. The objective function measures the Soft Margin distance, which indicates the goodness of separation.

Table 6: Predicted functional classifications of ORFs

Class	Gene ID	Gene Name	Description
TCA	YOR135C	*IRC14*	Decreased metabolite accumulation (glycogen)
	YPR002W	*PDH1*	Mitochondrial protein that participates in respiration, induced by diauxic shift
Meiosis	YOL131W		Putative protein of unknown function
	YHR157W	*REC104*	Protein involved in early stages of meiotic recombination; required for meiotic crossing over

Respiration	YPR020W	*ATP20*	Subunit of the mitochondrial F1F0 ATP synthase, which is a large enzyme complex required for ATP synthesis
	YGR182C		Dubious ORF unlikely to encode a protein
	YBL100C		Exhibits growth defect on a non-fermentable (respiratory) carbon source
	YDR077W	*SED1*	Major stress-induced structural GPI-cell wall glycoprotein in stationary-phase cells, associates with translating ribosomes, possible role in mitochondrial genome maintenance
Ribosome	YLR062C	*BUD28*	Dubious ORF, 98% ORF overlaps RPL22A, which codes protein component of the large (60S) ribosomal subunit
	YGL102C		Dubious ORF, overlaps 3' end of essential RPL28 gene encoding a large subunit ribosomal protein
	YLL044W		Dubious ORF, partially overlaps RPL8 coding for protein in large ribosomal subunit
	YLR339C		Dubious ORF, partially overlaps the essential gene RPP0, which is a conserved ribosomal protein
	YDR417C		Dubious ORF
	YNL119W	*NCS2*	Protein involved in invasive growth; ribosomal biogenesis and assembly
	YLR076C		Dubious ORF, partially overlaps the essential gene RPL10 encoding the ribosomal protein L10
	YFR031C-A	*RPL2A*	Protein component of the large ribosomal subunit

Histone and Proteolysis classes do not have any outstanding predictions. And for the listed prediction, some of them have definitive annotations, which exactly match the predicted classification, such as YHR157W; some others has possible functions that matches our prediction, such as YLR062C; and the others do not have any useful annotations, which could be investigated later in detail by experiments, such as YGR182C.

3. Future Direction

The main problem in this analysis is that the positive samples are only a small fraction of the whole dataset. This prohibits supervised learning methods to be trained efficiently with high accuracy for prediction. However, we could incorporate un-supervised learning methods, such as k-nearest neighbors clustering method, into our supervised learning methods. Semi-supervised learning is an emerging category of learning methods that typically deals with a small amount of labeled data with a large amount of unlabeled data. It is possible to use un-supervised clustering methods to find density based clusters first, and then use known annotations to label a few positive samples, and modify previous clustering results to accommodate the labels. This could be of great practical value to our dataset.

CONCLUSIONS

We have demonstrated that support vector machine can classify genes into some functional categories based on experimental microarray expression data, although the accuracy is not satisfied compared to the result from *Brown et al.* High power polynomial and radial kernel SVMs are best performers than linear or lower power polynomial kernel SVMs, Decision Tree, Naïve Bayes and Neural Networks methods. Prediction by 3-power polynomial SVM model gives reasonable results for the test set. Some predictions are verified by recent experiment results of their function labels, and some of our top predictions still do not have experimental supports, which could be valuable for experimental designs.

Acknowledgements

I thank Professor Martin Schulz and Professor Mark Gerstein for their suggestions and comments, and I also thank Jiang Du for his help on technique issue using R.

References

1. Eisen,M.,Spellman,P.,Brown, P.& Botstein, D.(1998) *Proc.Natl.Acad.Sci.USA* **95**, 14863-14868
2.Brown,M.,Grundy,W.,Lin,D.,Cristianini,N.,Sugnet,C.,Furey,T.,Ares,M.&Haussler,D.(2000)*Proc. Natl.Acad.Sci.USA* **97**, 262-267
3. Noble,W.& Pavlidis,P., (2006) Gist—Support vector machine and kernel principal components analysis software toolkit, Version 2.3
4. WikiPedia, http://en.wikipedia.org/wiki/