Statistical Applications in Genetics and Molecular Biology

Volume 3, *Issue* 1 2004 *Article* 33

PLS Dimension Reduction for Classification with Microarray Data

Anne-Laure Boulesteix*

*Department of Statistics, University of Munich, anne-laure.boulesteix@stat.uni-muenchen.de

PLS Dimension Reduction for Classification with Microarray Data*

Anne-Laure Boulesteix

Abstract

Partial Least Squares (PLS) dimension reduction is known to give good prediction accuracy in the context of classification with high-dimensional microarray data. In this paper, the classification procedure consisting of PLS dimension reduction and linear discriminant analysis on the new components is compared with some of the best state-of-the-art classification methods. Moreover, a boosting algorithm is applied to this classification method. In addition, a simple procedure to choose the number of PLS components is suggested. The connection between PLS dimension reduction and gene selection is examined and a property of the first PLS component for binary classification is proved. In addition, we show how PLS can be used for data visualization using real data. The whole study is based on 9 real microarray cancer data sets.

KEYWORDS: partial least squares, feature extraction, variable selection, boosting, gene expression, discriminant analysis, supervised learning

*I thank the two reviewers for their interesting comments, which helped me to improve this manuscript. I also thank Gerhard Tutz, Korbinian Strimmer and Joe Whittaker for critical comments and discussion, Klaus Hechenbichler for providing the R program for AdaBoost and Jane Fridlyand for providing the pre-processed NCI data set.

1 Introduction

The output of n microarray experiments can be summarized as a $n \times p$ data matrix, where p is the number of analyzed genes. p is always much larger than the number of experiments n. An important application of microarray technology is tumor diagnosis, i.e. class prediction. High-dimensionality makes the application of most classification methods difficult, if not impossible. To overcome this problem, one can either extract a small subset of interesting variables (gene selection) or construct m new components which summarize the original data as well as possible, with $m < p$ (dimension reduction).

Gene selection has been studied extensively in the last few years. The most commonly used gene selection procedures are based on a score which is calculated for all genes individually. Then the genes with the best scores are selected. These methods are often denoted as univariate gene selection. Several selection criteria have been used in the literature, e.g. the t statistic (Hedenfalk et al., 2001), Wilcoxon's rank sum statistic (Dettling and Bühlmann, 2003) or Ben Dor's combinatoric 'TNoM' score (Ben-Dor et al., 2000). When using a test statistic as criterion, it is useful to adjust the p-values with a multiple testing procedure (Dudoit et al., 2003). The main advantages of gene selection are its simplicity and interpretability. Gene selection procedures output a list of relevant genes which can be experimentally analyzed by biologists. Moreover, univariate gene selection is generally quite fast.

The scores mentioned in the previous paragraph are all based on the association of individual genes with the classes. Interactions and correlations between genes are omitted, although they are of great interest in system biology. For illustration, let us consider three genes A, B and C. A relevance score like the t statistic might tell us: gene A is more relevant than gene B and gene B is more relevant than gene C for classification. Now suppose we want to select two of these three genes to perform classification. The t statistic does not tell us if it is better to select A and B, A and C or B and C. A few sophisticated procedures intend to overcome this problem by selecting optimal subsets with respect to a given criterion instead of ranking the genes. Bo and Jonassen (2002) look for relevant pairs of genes, whereas Li et al. (2001) want to find optimal gene subsets via genetic algorithms. However, these methods generally suffer from overfitting: the obtained gene subsets might be optimal for the training data, but they do not perform as well on independent test data. Moreover, they are based on computationally intensive iterative algorithms and thus very difficult to interpret and implement.

Dimension reduction is a wise alternative to variable selection in order to overcome this dimensionality problem. It is also denoted as feature extraction. Unlike gene selection, such methods use all the genes included in the data set. The whole data are projected onto a low-dimensional space, thus allowing a graphical representation. The new components often give information or hints about the data's intrinsic structure, although there is no standard concept and procedure to do this. Dimension reduction is sometimes criticized for its lack of interpretability, especially for applied scientists who often need more concrete answers about individual genes. In this paper, we show that PLS dimension reduction is tightly connected to gene selection.

Dimension reduction methods for classification can be categorized into linear and nonlinear, supervised and unsupervised methods. Intuitively, supervised methods, i.e. methods which use the class information of the observations to construct new components, should be preferred to unsupervised methods, which work only 'by chance' in 'good' data sets (Nguyen and Rocke, 2002). Since nonlinear methods are generally computationally intensive and lack robustness, they are not recommended for microarray data analysis. To our knowledge, the only well-established supervised linear dimension reduction method working even if $n < p$ is the Partial Least Squares method (PLS). PLS is a linear method in the sense that the new components are linear combinations of the original variables. However, the coefficients defining the new components are not linear. Another approach denoted as between-group analysis has been proposed by Culhane et al. (2002), but it turns out that it is strongly related to PLS. Principal component analysis (Ghosh, 2002; Kahn et al., 2001) is an unsupervised method: its goal is to find uncorrelated linear transformations of the original variables which have high variance. As an unsupervised method, it is inappropriate for classification. Sufficient dimension reduction for classification is reviewed in Dennis and Lee (1999) and applied to microarray data in Chiaromonte and Martinelli (2001). Sufficient dimension reduction is a supervised approach: the goal is to find components which summarize the predictor variables such that the class and the predictor variables are independent given the new components. This method cannot be applied if $p > n$. A few other dimension reduction methods for classification are reviewed in Hennig (2004). Some of them, such as discriminant coordinates or the Bhattacharyya distance approach cannot be applied if $p > n$. The mean/variance difference coordinates approach is introduced in Young et al. (1987). It can theoretically be applied if $p > n$, but it requires the eigendecomposition of a $p \times p$ empirical covariance matrix, which is not recommended when $p >> n$. To our knowledge, PLS is the only fast supervised dimension reduction method which can handle a huge number of predictor variables.

It is known that PLS dimension reduction can be used for classification problems in the context of microarray data analysis (Nguyen and Rocke, 2002; Huang and Pan, 2003). However, these papers do not include any extensive comparative study of classification methods. Moreover, they treat the PLS technique as a 'black box' which is only meant to improve classification accuracy, without concern for the components themselves. In this paper, two aspects of PLS dimension reduction are examined. First, its classification performance is compared with the classification performance of top-ranking methods which have already been studied in the literature. Second, the connection between PLS dimension reduction and gene selection is examined.

In recent years, aggregation methods such as bagging (Breiman, 1996) and boosting (Freund, 1995) have been extensively analyzed. They lead to spectacular improvements of prediction accuracy when they are applied to classification problems. In microarray data analysis, accuracy improvement is also observed (Dettling and Bühlmann, 2003; Dudoit et al., 2002). So far, aggregating methods have been applied with weak and unstable classifiers such as stumps or classification trees. To our knowledge, boosting has never been used with dimension reduction techniques. In this paper, we apply a classical boosting algorithm (AdaBoost) in the framework of PLS dimension reduction.

The paper is organized as follows. PLS dimension reduction and boosting are introduced in section 2. In Section 3, the data are introduced and a few examples of data visualization using PLS dimension reduction are given. Classification results using PLS, PLS with boosting and various other methods are presented in section 4. In section 5, the connection between PLS and gene selection is studied and an interesting property of the first PLS component is proved in the case of binary responses.

In the following, $X_1, \dots, X_p$ denote the continuous predictors (genes) and $\mathbf{x} = (X_1, \dots, X_p)^T$ the corresponding random vector. $\mathbf{x}_i = (x_{i1}, \dots, x_{ip})^T$ for $i = 1, \dots, n$ denote independent identically distributed realizations of the random vector $\mathbf{x}$. Each row of the $n \times p$ data matrix $\mathbf{X} \in \mathbb{R}^{n \times p}$ contains a realization of $\mathbf{x}$.

2 Dimension reduction and classification with PLS

2.1 Outline of the method

Suppose we have a learning set $\mathcal{L}$ consisting of observations whose class is known and a test set $\mathcal{T}$ consisting of observations whose class has to be predicted. The data matrices corresponding to $\mathcal{L}$ and $\mathcal{T}$ are denoted as $\mathbf{X}_L$ and $\mathbf{X}_T$, respectively. The vector containing the classes of the observations from $\mathcal{L}$ is denoted as $\mathbf{Y}_L$. A classification method can be formalized as a function δ of $\mathbf{X}_L$, $\mathbf{Y}_L$ and the vector of predictors $\mathbf{x}_{new,i}$ corresponding to the ith observation from the test set:

$$\begin{array}{rlcl} \delta(., \mathbf{X}_L, \mathbf{Y}_L): & \mathbb{R}^p & \rightarrow & \{1, \dots, K\} \\ & \mathbf{x}_{new,i} & \rightarrow & \delta(\mathbf{x}_{new,i}, \mathbf{X}_L, \mathbf{Y}_L). \end{array}$$

In this section, we describe briefly the function δ which is discussed in the paper. From now on, it is denoted as δ_{PLS}. δ_{PLS} consists of two steps.

The first step is dimension reduction, which finds m appropriate linear transformations $Z_1, \dots, Z_m$ of the vector of predictors $\mathbf{x}$, where m has to be chosen by the user (this topic is discussed in Section 2.3). In the whole paper, $\mathbf{a}_1, \dots, \mathbf{a}_m$ denote the $p \times 1$ vectors which are used to construct the linear transformations $Z_1, \dots, Z_m$:

$$\begin{aligned} Z_1 &= \mathbf{a}_1^T \mathbf{x}, \\ \dots &= \dots, \\ Z_m &= \mathbf{a}_m^T \mathbf{x}. \end{aligned}$$

In this paper, the vectors $\mathbf{a}_1, \dots, \mathbf{a}_m$ are determined using the SIMPLS algorithm (de Jong, 1993), which is one of the variants of PLS dimension reduction. The SIMPLS algorithm is introduced in Section 2.2. The linear transformations $Z_1, \dots, Z_m$ are denoted as new components, for consistency with the PLS literature.

The second step is linear discriminant analysis using the new components $Z_1, \dots, Z_m$ as predictor variables. Linear discriminant analysis is described in Section 4. One could use another classification method such as logistic regression. However, logistic regression is known to give worse results for some specific data configurations. For example, logistic regression does not perform well when the different classes are completely or quasi-completely separated by the predictor variables, as claimed by Nguyen and Rocke (2002). Since this configuration is quite common in microarray data, logistic regression is not a good choice. Linear discriminant analysis, which is not recommended when the number of predictor variables is large (see Section 4), performs well when applied to a small number of approximately normally distributed PLS components.

The procedure to predict the class of the observations from $\mathcal{T}$ using $\mathcal{L}$ can be summarized as follows.

1. Determine the vectors $\mathbf{a}_1, \dots, \mathbf{a}_m$ using the SIMPLS algorithm (see Section 2.2) on the learning set $\mathcal{L}$. If $\mathbf{A}$ denotes the $p \times m$ matrix containing the vectors $\mathbf{a}_1, \dots, \mathbf{a}_m$ in its columns, the matrix $\mathbf{Z}_L$ of new components for the learning set is obtained as

$$\mathbf{Z}_L = \mathbf{X}_L \mathbf{A}. \tag{1}$$

2. Compute the matrix $\mathbf{Z}_T$ of new components for the test data set as

$$\mathbf{Z}_T = \mathbf{X}_T \mathbf{A}. \tag{2}$$

3. Predict the class of the observations from $\mathcal{T}$ by linear discriminant analysis, using $Z_1, \dots, Z_m$ as predictor variables. The classifier is built using only $\mathbf{Z}_L$.

This two-step approach is applied to microarray data by Nguyen and Rocke (2002). In this paper, we use the SIMPLS algorithm by de Jong (1993), which can be seen as a generalization for multicategorical response variables of the algorithm used by Nguyen and Rocke (2002). The SIMPLS algorithm is presented in the next section.

2.2 The SIMPLS algorithm

Partial Least Squares (PLS) is a wide family of methods originally developed as a multivariate regression tool in the context of chemometrics (Martens and Naes, 1989). PLS regression was later studied by statisticians (Stone and Brooks, 1990; Garthwaite, 1994; Frank and Friedman, 1993). An overview of the history of PLS regression is given in Martens (2001). PLS regression is especially appropriate to predict a univariate or multivariate continuous response using a large number of continuous predictors. The underlying idea of PLS regression is to find uncorrelated linear transformations of the original predictor variables which have high covariance with the response variables. These linear transformations can then be used as predictors in classical linear regression models to predict the response variables. Since the p original variables are summarized into a small number of relevant new components, linear regression can be performed even if the number of original variables p is much larger than the number of available observations. The different PLS algorithms differ in the definition of the linear transformations. Here, the focus is on the SIMPLS algorithm, because it can handle both univariate and multivariate variables.

If Y is a binary response, it can be treated as a continuous response variable, since PLS regression does not require any distributional assumption. However, if Y is a multicategorical variable, it cannot be treated as a continuous response variable. The problem can be circumvented by dummy-coding. The multicategorical random variable Y is transformed into a K-dimensional random vector $\mathbf{y} \in \{0,1\}^K$ as follows:

$$\begin{aligned} y_{i1} &= 1 \quad \text{if } Y_i = k, \\ y_{ik} &= 0 \quad \text{otherwise,} \end{aligned}$$

where $\mathbf{y}_i = (y_{i1}, \dots, y_{iK})^T$ denotes the ith realization of $\mathbf{y}$. In the following, $\mathbf{y}$ denotes the random variable Y if Y is binary ($K = 2$) or the K-dimensional random vector as defined above if Y is multicategorical ($K > 2$).

The SIMPLS algorithm proposed by de Jong (1993) computes the vectors $\mathbf{a}_1, \dots, \mathbf{a}_m$ defined as follows.

Definition 1 *Let $\hat{COV}$ denote the empirical covariance computed from the available data set. $\mathbf{a}_1$ and $\mathbf{b}_1$ are the unit vectors maximizing $\hat{COV}(\mathbf{a}_1^T\mathbf{x}, \mathbf{b}_1^T\mathbf{y})$. For all $j = 2, \dots, m$, $\mathbf{a}_j$ and $\mathbf{b}_j$ are the unit vectors maximizing $\hat{COV}(\mathbf{a}_j^T\mathbf{x}, \mathbf{b}_j^T\mathbf{y})$ subject to the constraint $\hat{COV}(\mathbf{a}_j^T\mathbf{x}, \mathbf{a}_i^T\mathbf{x}) = 0$ for all $i = 1, \dots, j-1$.*

In words, the SIMPLS algorithm computes linear transformations of $\mathbf{x}$ and linear transformations of $\mathbf{y}$ which have maximal covariance, under the constraint that the linear transformations of $\mathbf{x}$ are mutually uncorrelated. In PLS regression, a multivariate regression model is then built using $\mathbf{y}$ as multivariate response variable and $\mathbf{a}_1^T\mathbf{x}, \dots, \mathbf{a}_m^T\mathbf{x}$ as predictors, hence the name PLS regression. The regression coefficients for each response variable and each original variable are also output by the SIMPLS algorithm. However, they are not used in this paper, since we use the SIMPLS algorithm for dimension reduction only: our focus is on the new components $Z_1, \dots, Z_m$, which are then used in linear discriminant analysis.

The predictor variables as well as the response variables have to be centered to have zero mean before running the SIMPLS algorithm. The `R` library `pls.pcr` includes an implementation of the SIMPLS algorithm, which is used in this paper. Except the number of PLS components, which is discussed in Section 2.3, PLS dimension reduction with SIMPLS does not involve any free parameter, which makes it very simple to use. To illustrate PLS dimension reduction, let us consider the following data matrix $\mathbf{X}$:

$$\begin{array}{ccccc} X_1 & X_2 & X_3 & X_4 & X_5 \\ 1 & 5 & 4 & 4 & 3 \\ 2 & 9 & 3 & 2 & 6 \\ 5 & 6 & 7 & 2 & 7 \\ 3 & 1 & 2 & 4 & 3 \end{array}$$

and the vector of classes

$$\mathbf{Y}^T = (1 \;\; 1 \;\; 2 \;\; 2).$$

After centering $\mathbf{Y}$ and the columns of $\mathbf{X}$, the SIMPLS algorithm is applied with e.g. $m = 2$. One obtains:

$$\begin{array}{rcl} \mathbf{a}_1^T & = & (1.77 \quad -4.86 \quad 0.53 \quad 0.76 \quad -0.82) \\ \mathbf{a}_2^T & = & (2.31 \quad 3.01 \quad 3.02 \quad -1.79 \quad 3.45) \end{array}$$

The matrix of new components is obtained as

$$\mathbf{Z} = \mathbf{X}\mathbf{A},$$

where $\mathbf{A}$ is the 5×2 matrix containing $\mathbf{a}_1$ and $\mathbf{a}_2$ in its columns:

$$\begin{array}{rr} Z_1 & Z_2 \\ -0.20 & -0.46 \\ -0.71 & 0.13 \\ 0.33 & 0.76 \\ 0.58 & -0.43 \end{array}.$$

As can be seen from the matrix $\mathbf{Z}$, Z_1 seems to separate the two classes very well. Z_2, which is uncorrelated with Z_1, seems to be less relevant. It indicates that $m = 1$ might be a sensible choice in this case. With less trivial data, the second PLS component is often relevant for the classification problem. It is often difficult to choose the right number m of PLS components to use for classification. In the following section, we adress the problem of the choice of m.

2.3 Choosing the number of components

There is no widely accepted procedure to determine the right number of PLS components. Here, we propose to use a simple method based on cross-validation. Suppose we have a learning set $\mathcal{L}$ and a test set $\mathcal{T}$. Only the learning set $\mathcal{L}$ is used to choose m. The following procedure is repeated N_{run} times: the classifier δ_{PLS} is built using only $\alpha\%$ of the observations from $\mathcal{L}$ and applied to the remaining observations, with m taking successively different values. For each of the N_{run} runs, the error rate is computed using only the remaining observations from $\mathcal{L}$. After N_{run} runs, the mean error rate over the N_{run} runs is computed for each value of m. For a more precise description of the mean error rate, see Section 4.1. The value of m minimizing the mean error rate is then used to predict the class of the observations from $\mathcal{T}$. In the following, it is denoted as m_{opt}. In our analysis, we set α to 0.7 for consistency with Section 4 and $N_{run} = 50$, which seems to be a good compromise between computation time and estimation accuracy. It seems that m_{opt} does not depend highly on the parameters α and N_{run}.

When the procedure described above is used to choose the number of PLS components, the classification method consisting of PLS dimension reduction and linear discriminant analysis does not involve any free parameter. Since boosting is known to improve classification accuracy in many situations, we suggest applying a boosting strategy to this classification method. Boosting is briefly introduced in the following section.

2.4 Boosting

Bagging and boosting consist of building a simple classifier using successively different bootstrap samples. In bagging, the bootstrap samples are based on the unweighted bootstrap and the predictions are made by majority voting. In boosting, the bootstrap samples are built iteratively using weights that depend on the predictions made in the last iteration. An early study focusing on statistical aspects of boosting is Schapire et al. (1998). A classifier based on a learning set $\mathcal{L}$ containing n_L observations is represented in section 2.1 as a function of the p-dimensional vector of predictors $\mathbf{x}_{new,i}$:

$$\begin{array}{rccl} \delta(.,\mathbf{X}_L,\mathbf{Y}_L): & \mathbb{R}^p & \rightarrow & \{1,\ldots,K\} \\ & \mathbf{x}_{new,i} & \rightarrow & \delta(\mathbf{x}_{new,i},\mathbf{X}_L,\mathbf{Y}_L). \end{array}$$

In boosting, perturbed learning sets $\mathcal{L}_1,\ldots,\mathcal{L}_B$ are formed adaptively by drawing from the learning set $\mathcal{L}$ at random, where the probability of an observation to be selected in $\mathcal{L}_k$ depends on the prediction made by $\delta(.,\mathbf{X}_{L_{k-1}},\mathbf{Y}_{L_{k-1}})$. Observations which are uncorrectly classified by $\delta(.,\mathbf{X}_{L_{k-1}},\mathbf{Y}_{L_{k-1}})$ have greater probability to be selected in $\mathcal{L}_k$.

The discrete AdaBoost procedure was proposed by Freund (1995). In the first iteration, the weights are initialized to $w_1 = \cdots = w_{n_L} = 1/n_L$. In the following we show the k-th step of the algorithm as described by Tutz and Hechenbichler (2004).

Discrete AdaBoost algorithm

1. - Based on the resampling probabilities $w_1, \dots, w_{n_L}$, the learning set $\mathcal{L}_k$ is sampled from $\mathcal{L}$ with replacement.
   - The classifier $\delta(., \mathbf{X}_{L_k}, \mathbf{Y}_{L_k})$ is built.
2. The learning set $\mathcal{L}$ is run through the classifier $\delta(., \mathbf{X}_{L_k}, \mathbf{Y}_{L_k})$ yielding an error indicator $\epsilon_i = 1$ if the i-th observation is classified incorrectly and $\epsilon_i = 0$ otherwise.
3. With $e_k = \sum_{i=1}^{n_L} w_i \epsilon_i$, $b_k = (1 - e_k)/e_k$ and $c_k = \log(b_k)$ the resampling probabilities are updated for the next step by

$$w_{i,new} = \frac{w_i b_k^{\epsilon_i}}{\sum_{j=1}^{n_L} w_j b_k^{\epsilon_j}} = \frac{w_i \exp\left(c_k \epsilon_i\right)}{\sum_{j=1}^{n_L} w_j \exp\left(c_k \epsilon_j\right)}$$

After B iterations the aggregated voting for observation $\mathbf{x}_{new}$ is obtained by

$$\arg\max_j \left(\sum_{k=1}^{B} c_k I(\delta(x, \mathbf{X}_{L_k}, \mathbf{Y}_{L_k}) = j)\right)$$

In this paper, we propose to apply the AdaBoost algorithm with $\delta = \delta_{PLS}$ with different numbers of components. To our knowledge, boosting has never be used in the context of dimension reduction. In the whole study, we use 9 real microarray cancer data sets which are introduced in the following section.

3 Data

3.1 Data sets

Colon: The colon data set is a publicly available ’benchmark’ gene expression data set which is extensively described in Alon et al. (1999). The data set contains the expression levels of 2000 genes for 62 patients from two classes. 22 patients are healthy patients and 40 patients have colon cancer.

Leukemia: This data set is introduced by Golub et al. (1999) and contains the expression levels of 7129 genes for 47 ALL-leukemia patients and 25 AML-leukemia patients. It is included in the `R` library `golubEsets`. After data preprocessing following the procedure described in Dudoit et al. (2002), only 3571 variables remain. It is easy to achieve excellent classification accuracy on this data set, even with quite trivial methods as described in the original paper by Golub et al. (1999).

Prostate: This data set gives the expression levels of 12600 genes for 50 normal tissues and 52 prostate cancer tissues. We threshold the data and filter genes as described in Singh et al. (2002). The filtering step leaves us with 5908 genes.

Breast cancer (ER+/ER-): This data set gives the expression levels of 7129 genes for 46 breast cancer patients from which 23 have status ER+ and 23 have status ER-. It is presented in West et al. (2002).

Carcinoma: This data set comprises the expression levels of 7463 genes for 18 normal tissues and 18 carcinomas. We standardize each array to have zero mean and unit variance. For an extensive description of the data set, see Notterman et al. (2001).

Lymphoma: The data set presented by Alizadeh et al. (2000) comprises the expression levels of 4026 genes for 62 patients from 3 different classes (B-CLL, FL and DLBCL). The missing values are inputed as described in Dudoit et al. (2002) using the function `pamr.inpute` from the `R` library `pamr` (Tibshirani et al., 2002).

SRBCT: This gene expression data set is presented in Kahn et al. (2001). It contains the expression levels of 2308 genes for 83 Small Round Blue Cells Tumor (SRBCT) patients belonging to one of the 4 tumor classes: Ewing family of tumors (EWS), non-Hodgkin lymphoma (BL), neuroblastoma (NB) and rhabdomyosarcoma (RMS).

Breast cancer (BRCA): This breast cancer data set contains the expression levels of 3227 genes for breast cancer patients with one of the three tumor types: sporadic, BRCA1 and BRCA2. It is described in Hedenfalk et al. (2001). The data are preprocessed as described in Simon et al. (2004).

NCI: This dataset comprises the expression levels of 5244 genes for 61 patients with 8 different tumor types: 7 breast, 5 central nervous system, 7 colon, 6 leukemia, 8 melanoma, 9 non-small-cell-lung-carcinoma, 6 ovarian, 9 renal Ross et al. (2000). The data are preprocessed as described in Dudoit et al. (2002).

In this next section, some of these data sets are visualized graphically using PLS dimesnion reduction.

3.2 Data Visualization via PLS dimension reduction

An advantage of PLS dimension reduction is the possibility to visualize the data by graphical representation. For instance, one can plot the second PLS component against the first PLS component using different colors for each class. As a visualization method, PLS might be useful for applied researchers who need simple graphical tools. In the following, we give a few concrete examples and show briefly and qualitatively that PLS dimension reduction can outline relevant cluster structures.

Suppose we have to analyse a data set with a binary response. One of the classes, e.g. class 2, consists of 2 subclasses: 2a and 2b. In the following, we try to interpret the PLS components in terms of clusters. For example, the first PLS component may discriminate between class 1 and class 2a and the second PLS component between class 1 and class 2b. In order to illustrate this point, we perform PLS dimension reduction on the whole prostate data set. We also cluster the observations from class 2 into two subclasses 2a and 2b using the k-means algorithm on the original variables $X_1, \ldots, X_p$. For the k-means clustering, we set the maximal number of iterations to 10. As can be seen from Figure 1, the first PLS component separates almost perfectly class 1 and class 2b, whereas the second PLS component separates almost perfectly class 1 and class 2a. Thus, the two PLS components can be interpreted in terms of clusters. A similar result can be obtained with the breast cancer data. We perform PLS dimension reduction on the whole breast cancer data set and cluster the observations from class 2 into 2a and 2b using the k-means algorithm on $X_1, \ldots, X_p$. The first and the second PLS components are reprensented as a scatterplot in Figure 2. We observe that the first PLS component can separate class 1 from class 2 perfectly. The second PLS component separates only 1 and 2a from 2b. Similar results are observed for the carcinoma and the leukemia data. Thus, for 4 of 5 data sets with binary class, the PLS components can be easily interpreted in terms of clusters.

However, in our examples, we do not know whether the subclasses 2a and 2b are biologically interpretable: they are only the output of the k-means clustering algorithm. Thus, we also perform the same analysis on the lymphoma data set, for which three biologically interpretable classes are known. Patients with tumor type DLBCL are assigned to class 1, B-CLL to class 2a and FL to class 2b. PLS dimension reduction is performed as if the class were binary. As can be seen from Figure 3, the first PLS discriminates between class 1 and class 2, whereas the second PLS discriminates between class 2a and classes 1 and 2b.

As a conclusion, we recommend the PLS technique as a visualization tool, because it can outline relevant cluster structures. As can be seen from the figures presented in this section, the PLS components can be used to predict the class of new observations. The next section is dedicated to the classification method δ_{PLS} consisting of PLS dimension reduction and linear discriminant analysis.

4 Classification results on real microarray data

4.1 Study design

For each data set, 200 random partitions into a learning data set $\mathcal{L}$ containing n_L observations and a test data set $\mathcal{T}$ containing the $n - n_L$ remaining observations are generated. This approach for evaluating classification methods was used in one of the most extensive comparative studies of classification meth-

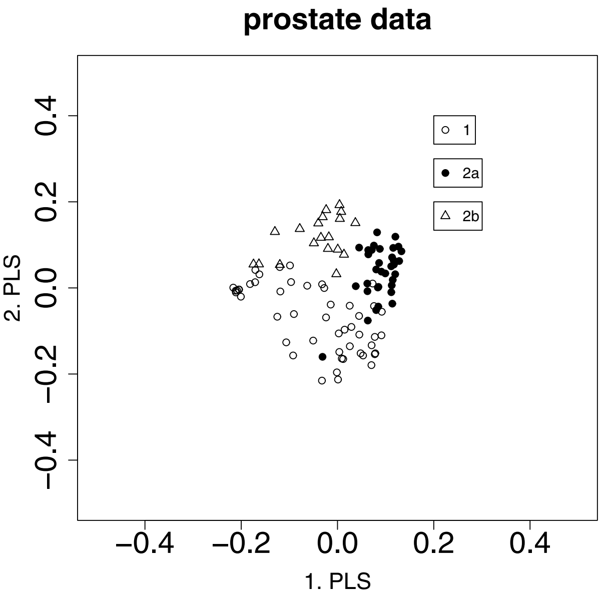

Figure 1: First and second PLS components for the prostate data

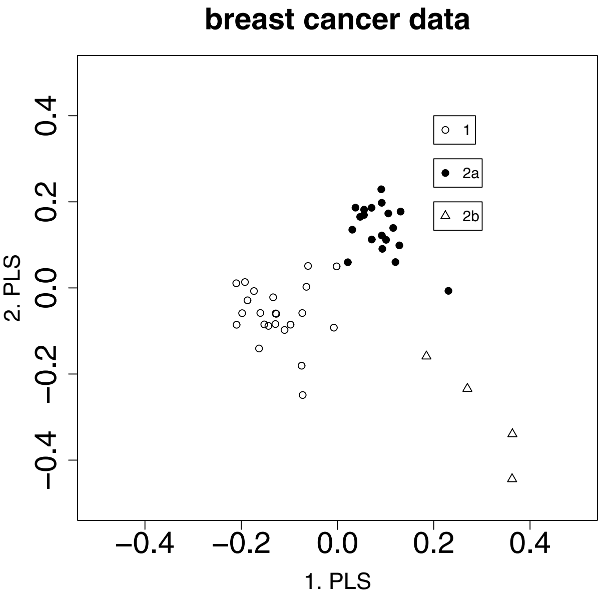

Figure 2: First and second PLS components for the breast cancer data

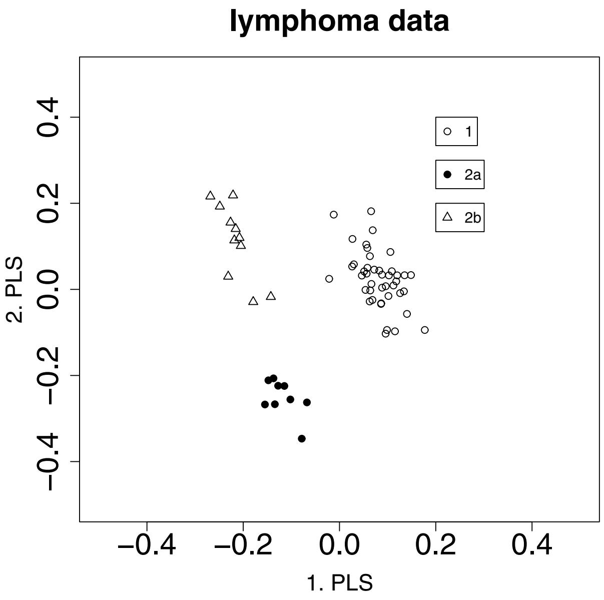

Figure 3: First and second PLS components for the lymphoma data with 2 classes

ods for microarray data (Dudoit et al., 2002). It is believed to be more reliable than leave-one-out cross-validation (Braga-Neto et al., 2004). We fix the ratio n_L/n at 0.7, which is a usual choice. For each partition $\{\mathcal{L}, \mathcal{T}\}$, we predict the class of the observations from $\mathcal{T}$ using δ_{PLS} with successively 1,2,3,4,5 PLS components for the data sets with a binary response. We also use the discrete AdaBoost boosting algorithm based on the classifier $\delta = \delta_{PLS}$ with 1,2,3 PLS components. For data sets with multicategorical responses, we use 1,2,3,4,5,6 PLS components for the lymphoma and BRCA data, 1,2,3,4,5,6,8,10 for the SRBCT data and 1,5,10,15,20 components for the NCI data.

For each approach and for each number of components, the mean error rate over the 200 partitions is computed using only the test set. Let $n_{\mathcal{T}_k}$ denote the number of observations in the test set $\mathcal{T}_k$, $\mathcal{L}_1, \ldots, \mathcal{L}_{200}$ denote the 200 learning sets and $\mathcal{T}_1, \ldots, \mathcal{T}_{200}$ the 200 corresponding test sets. For a given approach, a given number of components and a given partition, $\hat{Y}_i$ denotes the predicted class of the ith observation of the test set. The mean error rate MER over the 200 partitions is given by

$$MER = \frac{1}{200} \sum_{k=1}^{200} \frac{1}{n_{\mathcal{T}_k}} \sum_{i=1}^{n_{\mathcal{T}_k}} I(\hat{Y}_i \neq Y_i), \tag{3}$$

where I is the standard indicator function ($I(A) = 1$ if A is true, $I(A) = 0$ otherwise).

The results are summarized in Tables 1 and 2.

For each partition $\{\mathcal{L}_k, \mathcal{T}_k\}$, the optimal number of PLS components m_{opt} is estimated following the procedure described in section 2.3 and the error rate of δ_{PLS} with m_{opt} PLS components is computed. The corresponding mean error rate over the 200 random partitions is given in Table 1 (last column). The candidate numbers of components used to determine m_{opt} by cross-validation are also given in the table for each data set. For the data sets with a binary response, m_{opt} is chosen from $1, 2, 3, 4, 5$. For data sets with a multicategorical response (except the NCI data), m_{opt} is chosen from $1, 2, 3, 4, 5, 6, 7, 8, 9, 10$. For the NCI data set, which has much more classes, m_{opt} is chosen from $1, 5, 10, 15, 20$.

For comparison, the mean error rate obtained with some of the best classification methods for microarray data is also computed. The first one is nearest-neighbor classification based on 5 neighbors (5NN). This method can be summarized as follows. For each observation from the test set, the 5 closest observations ('neighbors') in the learning set are found and the observation is assigned to the class which is most common among those k neighbors. Closeness is measured using a specified distance metric. The most common distance metric, which we use here, is the euclidean distance metric. Nearest-neighbor classification is implemented in the R library `class`. This method is known to achieve good classification accuracy with microarray data (Dudoit et al., 2002).

The second method is linear discriminant analysis (LDA), which is also known to give good classification accuracy (Dudoit et al., 2002). A short description of linear discriminant is given in the following. Suppose we have p predictor variables. The random vector $\mathbf{x} = (X_1, \dots, X_p)^T$ is assumed to a multivariate normal distribution within class k ($k = 1, \dots, K$) with mean $\boldsymbol{\mu}_k$ and covariance matrix $\boldsymbol{\Sigma}_k$. In linear discriminant analysis, $\boldsymbol{\Sigma}_k$ is assumed to be the same for all classes: for all k, $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$. Using estimates $\hat{\boldsymbol{\mu}}_k$ and $\hat{\boldsymbol{\Sigma}}$ in place of $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}$, the maximum-likelihood discrimant rule assigns the ith new observation $\mathbf{x}_{new,i}$ to the class

$$\delta(\mathbf{x}_{new,i}) = \arg\min_k (\mathbf{x}_{new,i} - \hat{\boldsymbol{\mu}}_k)\hat{\boldsymbol{\Sigma}}^{-1}(\mathbf{x}_{new,i} - \hat{\boldsymbol{\mu}}_k)^T. \tag{4}$$

This approach is usually denoted as linear discriminant analysis, because $\delta(\mathbf{x}_{new,i})$ is a linear function of the vector $\mathbf{x}_{new,i}$. In our study, it does not perform as well as 5NN, SVM and PAM, probably because the estimation of the inverse of $\hat{\boldsymbol{\Sigma}}$ is not robust when the number of variables is two large. Thus, the classification results using linear discriminant analysis are not shown.

The third method is Support Vector Machines (SVM). This method is used by Furey et al. (2000) and seems to perform well on microarray data. The idea is to find a separating hyperplan which separates the classes as well as possible in an enlarged predictor space. This leads to a complex optimization problem in high dimension. In our study, the optimal hyperplan is determined using the function `svm` from the `R` library `e1071` with the default parameter settings.

A short overview of NN, LDA and SVM is given in Hastie et al. (2001). These three methods require preliminary gene selection. The gene selection is performed by ranking genes according to the BSS/WSS-statistic, where BSS denotes the between-group sum of squares and WSS the within-group sum of squares. For gene j the BSS/WSS-statistic is calculated as

$$BSS_j/WSS_j = \frac{\sum_{k=1}^K \sum_{i:Y_i=k} (\hat{\mu}_{jk} - \hat{\mu}_j)^2}{\sum_{k=1}^K \sum_{i:Y_i=k} (x_{ij} - \hat{\mu}_{jk})^2},$$

where $\hat{\mu}_j$ is the sample mean of X_j and $\hat{\mu}_{jk}$ is the sample mean of X_j within class k, for $k = 1, \dots, K$. The genes with the highest BSS/WSS-statistic are selected. There is no well-established rule to choose the number of genes to select, which is a major drawback of classification methods requiring gene selection. In this study, we decide to use 20 or 50 genes for data sets with a binary response and 100 and 200 genes for data sets with a multicategorical response. The results obtained using other numbers of genes turn out to be similar or worse. Moreover, these numbers are in agreement with similar studies found in the literature (Dudoit et al., 2002).

At last, we apply a recent method called 'prediction analysis of microarray'(PAM) which was especially designed for high-dimensional microarray data (Tibshirani et al., 2002). To our knowledge, it is the only fast classification method beside PLS which can be applied to high-dimensional data without gene selection. PAM is based on shrunken centroids. The user has to choose the shrinkage parameter Δ. The number of genes used to compute the shrunken centroids depends on Δ. A possible choice is $\Delta = 0$: all genes are used to compute the centroids. Tibshirani et al. (2002) propose to select the best value of Δ by cross-validation: the classification accuracy is evaluated by leave-one-out cross-validation for a set of 30 values of Δ. The value of Δ minimizing the number of misclassifications is chosen. In our study, we try successively both approaches: $\Delta = 0$ (denoted as PAM) and $\Delta = \Delta_{opt}$ (denoted as PAM-opt), where Δ_{opt} is determined by leave-one-out cross-validation as described in Tibshirani et al. (2002). The PAM method as well the choice of Δ by cross-validation are implemented in the `R` library `pamr` (Tibshirani et al., 2002).

The table of results contains only the error rates obtained with 5NN, SVM, PAM and PAM-opt, because the classification accuracy with LDA was found to be comparatively bad for all data sets. The number of selected genes is specified for each method: for example, 'SVM-20' stands for Support Vector Machines with 20 selected genes.

The classification results obtained with δ_{PLS}, 5NN, SVM and PAM are presented in the next section, where as the results obtained with boosting are discussed in Section 4.3.

4.2 Classification accuracy of δ_{PLS}

The classification results using the PLS-based approach δ_{PLS} are summarized in Table 1. The data sets with a binary response can be divided in two groups. For the leukemia and carcinoma data, the classification accuracy does not depend highly on the number of PLS components. It seems that subsequent components are only noise. On the contrary, the error rate is considerably reduced by using more than one component for the colon, prostate and breast cancer data. The improvement is rather dramatic for the prostate data. Thus, it seems that for data sets with low error rates (leukemia, carcinoma), the classes are optimally separated by one component, whereas subsequent components are useful for data sets with high error rates (prostate, colon, breast cancer).

PLS dimension reduction is very fast because it is based on linear operations with small matrices. The proposed procedure is much faster than the standard approach consisting of selecting a gene subset and building a classifier on this subset. For the lymphoma data and the SRBCT data, $K - 1$ seems to be the minimum number of PLS components required to obtain a good classification accuracy. It is noticeable that δ_{PLS} can also perform very well on data sets with many classes ($K = 8$ for the NCI data).

As can be seen from Table 1, the number of components giving the best classification accuracy is not the same for all data sets. When our procedure to determine the number of useful PLS components is used for each partition $(\mathcal{L}, \mathcal{T})$, the classification accuracy turns out to be quite good. In Figure 4, histograms of m_{opt} over the 200 random partitions are represented for each data set. These histograms agree with Table 1. For instance, the most frequent value

Colon	1	2	3	4	5		m_{opt}
$(K = 2)$	0.136	0.114	0.119	0.143	0.147		0.124
Leukemia	1	2	3	4	5		m_{opt}
$(K = 2)$	0.020	0.028	0.03	0.030	0.028		0.024
Prostate	1	2	3	4	5		m_{opt}
$(K = 2)$	0.366	0.140	0.076	0.081	0.077		0.078
Breast cancer	1	2	3	4	5		m_{opt}
$(K = 2)$	0.14	0.110	0.104	0.106	0.103		0.110
Carcinoma	1	2	3	4	5		m_{opt}
$(K = 2)$	0.025	0.021	0.022	0.024	0.023		0.024
Lymphoma	1	2	3	4	5	6	m_{opt}
$(K = 3)$	0.037	0.0003	0.002	0.001	0.004	0.003	0.004
SRBCT	1	2	3	4	6	10	m_{opt}
$(K = 4)$	0.343	0.200	0.056	0.027	0.009	0.003	0.003
BRCA	1	2	3	4	5	6	m_{opt}
$(K = 3)$	0.468	0.348	0.310	0.268	0.285	0.303	0.0304
NCI	1	5	10	15	20		m_{opt}
$(K = 8)$	0.715	0.338	0.293	0.318	0.325		0.329

Table 1: Mean error rate over 200 random partitions with PLS

Colon	5NN-20	5NN-50	$SVM - 20$	$SVM - 50$	PAM	PAM-opt
$(K = 2)$	0.182	0.19	0.134	0.139	0.143	0.130
Leukemia	5NN-20	5NN-50	$SVM - 20$	$SVM - 50$	PAM	PAM-opt
$(K = 2)$	0.034	0.039	0.038	0.05	0.022	0.046
Prostate	5NN-20	5NN-50	$SVM - 20$	$SVM - 50$	PAM	PAM-opt
$(K = 2)$	0.119	0.124	0.086	0.085	0.370	0.099
Breast cancer	5NN-20	5NN-50	$SVM - 20$	$SVM - 50$	PAM	PAM-opt
$(K = 2)$	0.117	0.123	0.100	0.093	0.120	0.147
Carcinoma	5NN-20	5NN-50	$SVM - 20$	$SVM - 50$	PAM	PAM-opt
$(K = 2)$	0.020	0.021	0.024	0.029	0.036	0.096
Lymphoma	5NN-100	5NN-200	$SVM - 100$	$SVM - 200$	PAM	PAM-opt
$(K = 3)$	0.014	0.003	0.038	0.019	0.013	0.042
SRBCT	5NN-100	5NN-200	$SVM - 100$	$SVM - 200$	PAM	PAM-opt
$(K = 4)$	0.012	0.0052	0.010	0.014	0.046	0.069
BRCA	5NN-100	5NN-200	$SVM - 100$	$SVM - 200$	PAM	PAM-opt
$(K = 3)$	0.378	0.318	0.588	0.581	0.331	0.396
NCI	5NN-100	5NN-200	$SVM - 100$	$SVM - 200$	PAM	PAM-opt
$(K = 8)$	0.394	0.366	0.466	0.452	0.316	0.296

Table 2: Mean error rate over 200 random partitions with classical methods

of m_{opt} for the colon data is 2. It can be seen in Table 1 that the best classification accuracy is obtained with 2 PLS components for the colon data.

Some of the classical methods tested in this paper also perform well, especially SVM and PAM. SVM performs slightly better than PAM for most data sets. However, a pitfall of SVM is that it necessitates gene selection in practice, although not in theory. On the whole, the PLS-based method presented in this paper performs at least as good as the other methods for most data sets. More specifically, PLS performs better than the other methods for the colon, the prostate data, the SRBCT and the BRCA data. It is (approximately) as good as PAM and better than SVM and 5NN for the leukemia data, as good as SVM and better than PAM and 5NN for the breast cancer data, as good as 5NN and better than PAM and 5NN for the carcinoma data and the lymphoma data, and a bit worse than PAM-opt but much better than 5NN and PAM for the NCI data. Each of the three tested methods (5NN,SVM,PAM) performs much worse than PLS for at least two data sets. PLS is the only method which ranges among the two best methods for all data sets. This accuracy is not reached at the expense of computational time, except if one performs many cross-validation runs for the choice of the number of components. The problem of the choice of the number of components is one of the major drawbacks of the PLS approach. This problem is partly solved by the procedure based on cross-validation, but this procedure is computationally intensive and not optimal. Another inconvenient of the PLS approach which is often mentioned in the statistical literature is that it is based on an algorithm rather than on a theoretical probabilistic model, like LDA or PAM. However, PLS is a fast and efficient method which never fails to give a good to excellent classification accuracy for all the studied data sets. Since the best number of components can be estimated by cross-validation, the method does not involve any 'free' parameter like the number of selected genes for SVM or 5NN.

Boosting does not improve the classification obtained with δ_{PLS} in most cases. However, the results are interesting because they indicate a qualitative similarity between boosting and PLS. This topic is discussed in the next section.

4.3 Classification accuracy of discrete AdaBoost with $\delta = \delta_{PLS}$

4.3.1 Real Data

In this section, we compute the mean classification error rate over 50 random partitions using the AdaBoost algorithm with $\delta = \delta_{PLS}$ and $B = 30$. $B = 30$ turns out to be a sensible choice for all data sets, because the classification accuracy remains constant after approximately 20 iterations. The results are represented in Figure 5 (top) for the prostate data. Boosting can reduce the error rate when one or two PLS components are used. However, the classification accuracy of δ_{PLS} with three PLS components is not improved by boosting. It can be

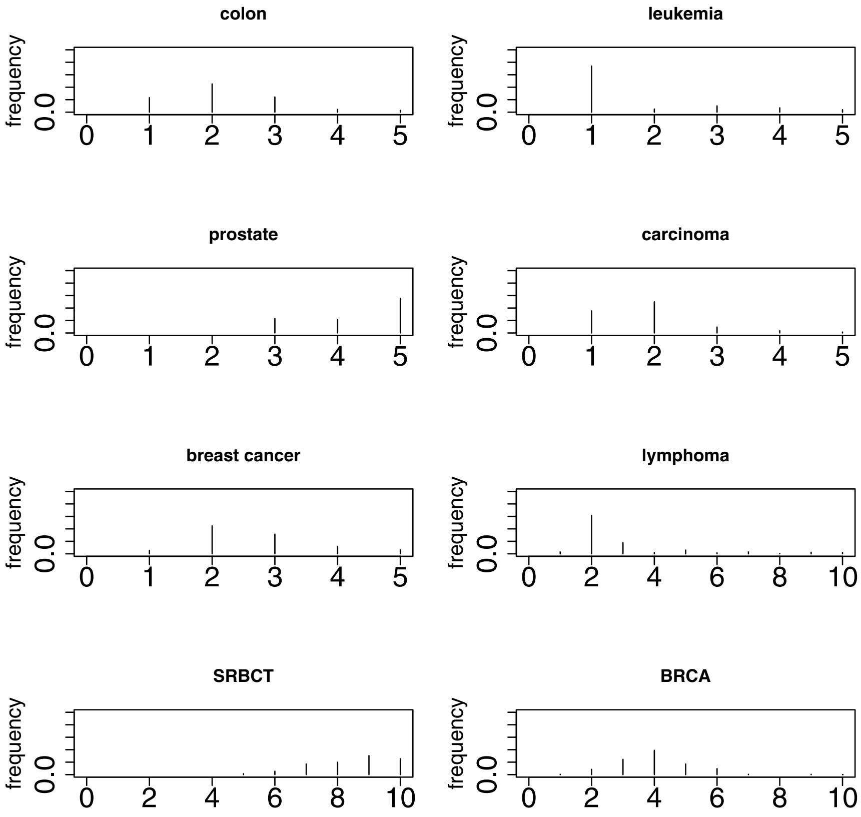

Figure 4: Histogram of the estimated optimal number of components for different data sets.

	$B = 1$	$B = 2$	$B = 3$	$B = 4$	$B = 5$
PLS 1	0.80	-0.74	0.79	-0.74	0.60
PLS 2	-0.48	0.63	-0.35	0.58	-0.30
PLS 3	0.03	0.00	-0.00	0.00	0.14
PLS 4	-0.06	-0.01	-0.03	-0.02	-0.19

Table 3: Correlations between 4 PLS components and the 5 first PLS components with boosting (prostate data)

seen from Table 1 that the best classification accuracy for δ_{PLS} is reached with three PLS components: the fourth and fifth PLS components do not improve the classification accuracy. Thus, with a fixed number m of PLS components, boosting improves the classification accuracy if and only the $(m+1)$th PLS component also does.

In order to examine the connection between boosting and PLS, we perform PLS dimension reduction on the whole prostate data set. We also run the AdaBoost algorithm with $\delta = \delta_{PLS}$ (with 1 component) and compute the empirical correlations between the four first PLS components and the first component obtained at each boosting iteration. The results are shown for 5 boosting iterations in Table 3. The first component at each boosting iteration is strongly correlated with the first and the second PLS component, but not with the subsequent components. This statement agrees with the classification accuracy results: it can be seen from Figure 5 (top) that the classification accuracy obtained by boosting with one component equals approximately the classification accuracy of δ_{PLS} with two components.

Thus, both the classification results and the study of the correlations suggest a similarity between the PLS components obtained in subsequent boosting iterations and the subsequent PLS components obtained when δ_{PLS} is used without boosting. The same can be observed with the multicategorical responses. Here we focus on the SRBCT data, but the study of other data sets yields similar results. The mean error rate of δ_{PLS} with boosting is depicted in Figure 5 (bottom) for different numbers of PLS components. As for the prostate data, boosting reduces the error rate when one or two PLS components are used, but not when three PLS components are used. As can be seen from Table 1, three is the minimal number of components required to obtain good classification accuracy. Thus, with a fixed number m of PLS components, boosting improves the classification accuracy if and only the $(m+1)$th PLS component also does.

The similarity between PLS and boosting can be intuitively and qualitatively explained as follows. In this paragraph, 'boosting' stands for 'boosting of δ_{PLS} with one component'. At iteration k in boosting, an observation is either in or out of the learning set, and the probability depends on how the observation was classified at iteration $k-1$. The observations which are misclassified at iteration $k-1$ have higher probability to be selected in the learning set at iteration

k. At each iteration, the error rate in the learning set is expected to decrease, since the algorithm focuses on 'problematic' observations. In practice, the PLS components computed at subsequent iterations have low correlations with the PLS component computed at the first iteration. The PLS component computed at the first iteration has high covariance with the class in the whole learning set, whereas the PLS components computed at subsequent iterations have high covariance with the class in particular learning sets where observations which are uncorrectly predicted by the first PLS component are over-representated.

Let us consider δ_{PLS} without boosting, but with several PLS components. For the computation of each PLS component, all the observations remain in the learning set, but the mth PLS component is uncorrelated with the $m-1$ first PLS components. Thus, observations which are correctly predicted by the $m-1$ first PLS components do not participate as much in the construction of the mth PLS component as the observations which are uncorrectly predicted. In conclusion, both algorithms (boosting and PLS with several components) focus on observations or directions which have been neglected in the previous runs (for boosting) or components (for PLS). The theoretical connection between boosting and PLS could be examined in future work in a probabilistic framework.

4.3.2 Simulated Data

In simulations, we examine the effect of boosting on the classification accuracy for multicategorical data. For the generation of simulated data, the number of classes K is set successively to $K = 3$ and $K = 4$ and the number of observations in each class is set to 30 for the learning sets. The test sets contain 100 observations for each class, in order to improve the accuracy of the estimation of the error rate. To limit the computation time, the number of predictor variables p is set to $p = 200$. Similar results can be obtained with different values of n and p. Each class k is separated from the other classes by a group of 10 genes. The K groups of relevant genes are distinct, which is a simplifying but realistic hypothesis. For each class k, the 10 relevant genes are assumed to have the following conditional distributions:

$$\begin{aligned} X|Y = k &\sim \mathcal{N}(\mu = 0, \sigma = 1) \\ X|Y \neq k &\sim \mathcal{N}(\mu = 1, \sigma = 1), \end{aligned}$$

where $\mathcal{N}(\mu, \sigma)$ denotes the normal distribution with mean μ and standard deviation σ.

For $K = 3$ and $K = 4$ successively, we generate 50 learning data sets $\{\mathcal{L}_1, \dots, \mathcal{L}_{50}\}$ and 50 test data sets $\{\mathcal{T}_1, \dots, \mathcal{T}_{50}\}$ as follows. First, the K groups of 10 relevant genes are drawn within each class from the conditional distributions given above. The remaining genes are drawn from the standard normal distribution for all classes. For each pair $\{\mathcal{L}_k, \mathcal{T}_k\}$ ($k = 1, \dots, 50$), δ_{PLS} with boosting ($B = 30$) for 1,2,3 components is used to predict the classes of the

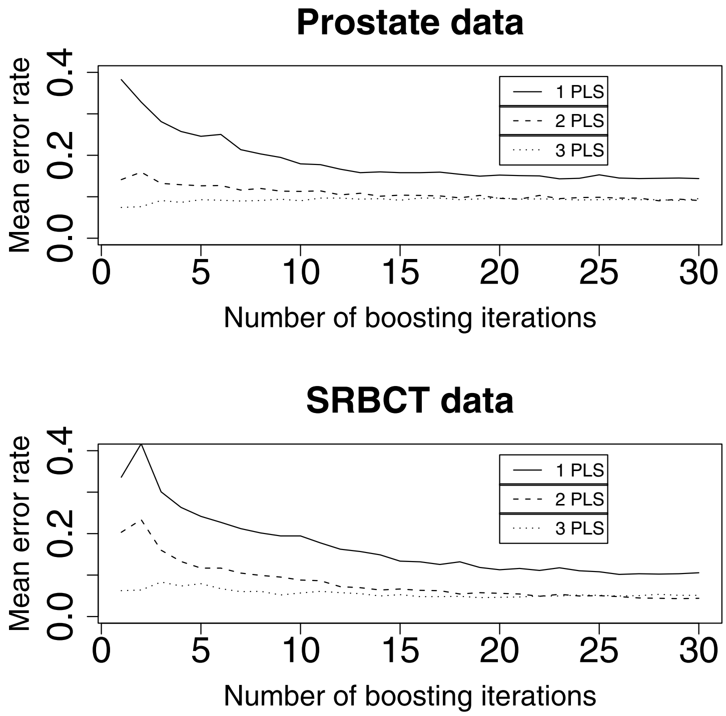

Figure 5: Mean error rate over 50 random partitions with AdaBoost and δ_{PLS} with different numbers of PLS components for the prostate data (top) and the SRBCT data (bottom)

	1	2	3
K=3	0.328	0.077	0.113
K=4	0.504	0.283	0.104

Table 4: Mean error rate over 50 simulated learning sets and test sets with δ_{PLS} for different numbers of PLS components.

observations from $\mathcal{T}_k$. The mean error rate over the 50 runs is then computed at each boosting iteration.

The results are depicted in Figure 6 for $K = 3$ (top) and $K = 4$ (bottom). As can be seen from Figure 6, boosting improves the classification accuracy of δ_{PLS} if and only if less than $K - 1$ components are used. It seems that using boosting with a larger number of components can even decrease the classification accuracy. For comparison, the classification accuracy of δ_{PLS} without boosting is given in Table 4 for different numbers of PLS components. The best classification accuracy is achieved with $K-1$ PLS components for both $K = 3$ and $K = 4$. Thus, the similarity between boosting and PLS which is observed for real data can also be observed for simulated data: for a given number m of PLS components, boosting improves the classification accuracy if and only if the $(m + 1)$th PLS component also does.

In the following section, we show a connection between the first PLS component and gene selection: the squared coefficient in the first PLS component can be seen as a score of relevance for single genes (see section 4 for more details). 'Boosted gene selection' might be an interesting application of boosting with PLS: we suggest that selecting the top-ranking genes at each boosting iteration might improve the classification accuaracy of classifiers based on small gene subsets, although the study of this topic would be beyond the scope of this paper.

5 PLS and gene selection

Biologists often want statisticians to answer questions such as 'which genes can be used for tumor diagnosis ?'. Thus, gene selection remains an important issue and should not be neglected. Dimension reduction is sometimes wrongly described as a black box which looses the information about single genes. In the following, we will see that PLS is strongly connected to gene selection.

In this section, only binary responses are considered: Y can take values 1 and 2. We denote as $\mathbf{Y}_C = (Y_{C1}, \ldots, Y_{Cn})^T$ the vector obtained by centering $\mathbf{Y} = (Y_1, \ldots, Y_n)^T$ to have zero mean:

$$\begin{aligned} Y_{Ci} &= -n_2/n \quad \text{if } Y_i = 1, \\ &= \quad n_1/n \quad \text{if } Y_i = 2, \end{aligned}$$

where n_1 and n_2 are the numbers of observations in class 1 and 2, respectively.

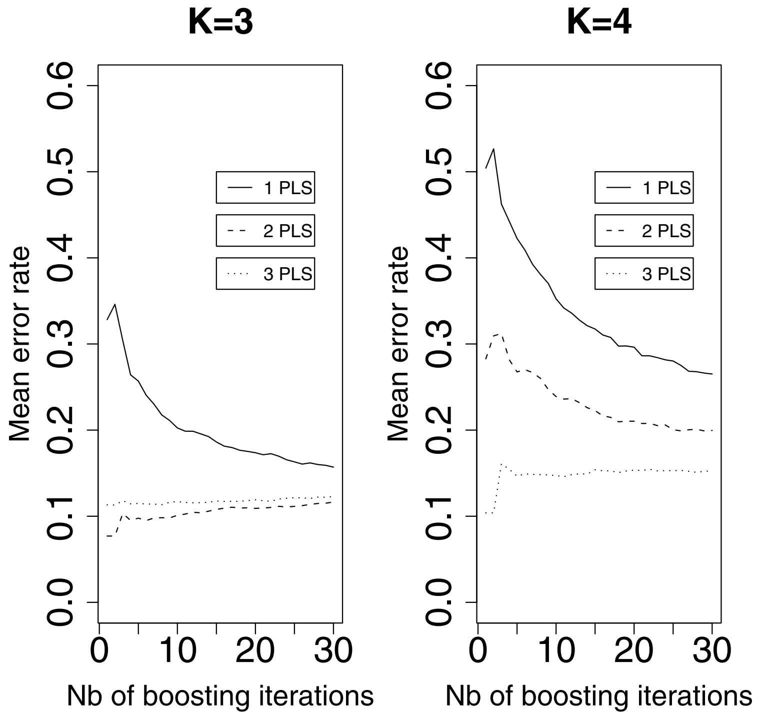

Figure 6: Mean error rate over 50 simulated learning sets and test sets with AdaBoost and δ_{PLS} with different numbers of PLS components for simulated data for $K = 3$ (left) and $K = 4$ (right)

To perform PLS dimension reduction, it is not necessary to scale each column of the data matrix $\mathbf{X}$ to unit variance. However, the first PLS component satisfies an interesting property with respect to gene selection if $\mathbf{X}$ is scaled. In this section, the columns of the data matrix $\mathbf{X}$ are supposed to be have been scaled to unit variance and, as usual in the PLS framework, centered to zero mean. $\mathbf{a} = (a_1, \ldots, a_p)^T$ denotes the $p \times 1$ vector defining the first PLS component as calculated by the SIMPLS algorithm.

A classical gene selection scheme consists of ordering the p genes according to BSS_j/WSS_j and selecting the top-ranking genes. For data sets with a binary response, we argue that a_j^2 can also be seen as a scoring criterion for gene j and we prove that the ordering of the genes obtained using BSS_j/WSS_j is the same as the ordering obtained using a_j^2.

Theorem 1 *If $K = 2$, there exists a strictly monotonic function f such that*

$$BSS_j/WSS_j = f(a_j^2),$$

for $j = 1, \ldots, p$.

Proof. From the SIMPLS algorithm, we get

$$\mathbf{a} = c_1 \cdot \mathbf{X}^T \mathbf{Y}_C,$$

where c_1 is a scalar. For $j = 1, \ldots, p$,

$$a_j = c_1 \cdot \sum_{i=1}^{n} x_{ij} Y_{Ci}.$$

It leads to

$$\begin{array}{rcl} a_j & = & c_1 \cdot (-(n_2/n) \sum_{i:Y_i=1} x_{ij} + (n_1/n) \sum_{i:Y_i=2} x_{ij}) \\ a_j^2 & = & c_1^2 \cdot (n_1 n_2/n)^2 (\hat{\mu}_{j2} - \hat{\mu}_{j1})^2 \end{array}$$

For $K = 2$,

$$\begin{array}{rcl} BSS_j & = & n_1(\hat{\mu}_{j1} - \hat{\mu}_j)^2 + n_2(\hat{\mu}_{j2} - \hat{\mu}_j)^2 \\ & = & n_1((n\hat{\mu}_{j1} - n_1\hat{\mu}_{j1} - n_2\hat{\mu}_{j2})/n)^2 + n_2((n\hat{\mu}_{j2} - n_2\hat{\mu}_{j2} - n_1\hat{\mu}_{j1})/n)^2 \\ & = & (n_1 n_2^2/n^2 + n_2 n_1^2/n^2)(\hat{\mu}_{j2} - \widehat{\mu_{j1}})^2 \\ & = & c_2 a_j^2, \end{array}$$

where c_2 is a positive constant which does not depend on j. $BSS_j + WSS_j$ is proportional to the sample variance of X_j. Since the variables $X_1, \ldots, X_p$ all have equal sample variance, there exists a constant c_3 which is independent of j such that

$$\begin{array}{rcl} BSS_j/WSS_j & = & \frac{BSS_j}{c_3 - BSS_j} \\ & = & \frac{c_2 a_j^2}{c_3 - c_2 a_j^2}. \end{array}$$

□

As a consequence, the first PLS component calculated by the SIMPLS algorithm can be used to order and select genes and the ordering is the same as the ordering produced by one of the most widely accepted selection criteria. As an illustration, the BSS/WSS ratio can be computed for the 2000 genes of the colon data set. For the 5 first genes, one obtains:

$$1.069 \cdot 10^{-2}, \;\; 3.979 \cdot 10^{-5}, \;\; 6.439 \cdot 10^{-3}, \;\; 2.431 \cdot 10^{-3}, \;\; 9.492 \cdot 10^{-4}.$$

The coefficients of these 5 genes for the first PLS component are

$$9.280 \cdot 10^{-5}, \;\; -5.691 \cdot 10^{-6}, \;\; 7.217 \cdot 10^{-5}, \;\; -4.444 \cdot 10^{-5}, \;\; 2.779 \cdot 10^{-5}.$$

As can be seen from these partial results, the ordering of the genes produced by the BSS/WSS ratio is the same as the ordering produced by the absolute value of the coefficient for the first PLS component. For the colon data, the 5 top-ranking genes are gene 493 (Hsa.37937), gene 377 (Hsa.36689), gene 249 (Hsa.8147), gene 1635 (Hsa.2097) and gene 1423 (Hsa.1832).

Up to a constant, the BSS/WSS-statistic equals the F-statistic which is used to test the equality of the means within different groups. Since BSS_j/WSS_j is obtained by a strictly monotonic transformation of a_j^2, a_j^2 can be seen as a test statistic itself. We prove that the SIMPLS algorithm can be used as a gene selection procedure which is exactly equivalent to the procedure based on the BSS/WSS ratio or on the F-statistic. This method tend to be sensitive to outliers, which are common in microarray data. Moreover, it does not incorporate interactions and correlations between genes, as all univariate criteria. However, it is one of the most widely used criteria for gene selection and seems to perform well in most cases (Dudoit et al., 2002). We claim that one should rather use the first PLS component than the BSS/WSS ratio because it is faster to compute.

6 Discussion

In this paper, several aspects of PLS dimension reduction for classification are examined. First, PLS is compared to several other classification methods which are known to give excellent classification accuracy. To our knowledge, this work is the first extensive comparison study including PLS. The classifier δ_{PLS} turns out to be the best one in terms of classification accuracy for most of the data sets. Another advantage is its computational efficiency. Even if PLS dimension reduction is originally designed for continuous regression, it can be successfully applied to classification problems. To determine the optimal number of PLS components, a simple cross-validation procedure is proposed. The reliability of this procedure is quite good, although not perfect. An aggregation strategy (AdaBoost) was used in the hopes of improving the classification accuracy, because aggregation methods are known to be very effective in reducing the error rate on independent test data. The conclusion is that boosting does not improve the classification accuracy of PLS, except in some special cases. The second topic of this paper is gene selection. We show that the first PLS component can be used for gene selection and prove that the proposed procedure is equivalent to a well-known gene selection procedure found in the literature. Thus, the information on single genes does not get lost through PLS dimension reduction. Moreover, we claim that PLS dimension reduction can be used as a visualization tool. Contrary to principal component analysis, PLS is a supervised procedure which uses the information about the class of the observations to construct the new components. Unlike sufficient dimension reduction and related methods, PLS can handle all the genes simultaneously and performs gene selection intrinsically. In a word, PLS is a very fast and competitive tool for classification problems with high-dimensional microarray data as regards to prediction accuracy, feature selection and visualization. In future work, one could examine the theoretic connection between PLS and boosting, as well as the use of boosting in gene selection. Since the best classification accuracy is often reached with more than one PLS component, the subsequent PLS components could also be used to perform a refined gene selection. One could also try to improve the procedure to choose the number of components. It seems that cross-validation is appropriate, but a more sophisticated cross-validation scheme could maybe improve the classification performance of our PLS-based approach.

References

Alizadeh, A. A., Eisen, M. B., Davis, R. E., Ma, C., Lossos, I. S., Rosenwald, A., Boldrick, J. C., Sabet, H., Tran, T., Yu, X., Powell, J. I., Yang, L., Marti, G. E., Moore, T., Hudson, J., Lu, L., Lewis, D. B., Tibshirani, R., Sherlock, G., Chan, W. C., Greiner, T. C., Weisenburger, D. D., Armitage, J. O., Warnke, R., Levy, R., Wilson, W., Grever, M. R., Byrd, J. C., Botstein, D., Brown, P. O., Staudt, L. M., 2000. Distinct types of diffuse large b-cell lymphoma identified by gene expression profiling. Nature 403, 503–511.

Alon, U., Barkai, N., Notterman, D. A., Gish, K., Ybarra, S., Mack, D., Levine, A., 1999. Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays. PNAS 96, 6745–6750.

Ben-Dor, A., Bruhn, L., Friedman, N., Nachman, I., Schummer, M., Yakhini, Z., 2000. Tissue classifiction with gene expression profiles. Journal of Computational Biology 7, 559–584.

Bo, T. H., Jonassen, I., 2002. New feature subset selection procedures for classification of expression profiles. Genome Biology 3, R17.

Braga-Neto, U., Hashimoto, R., Dougherty, E. R., Nguyen, D. V., Carroll, R. J., 2004. Is cross-validation better than resubstitution for ranking genes ? Bioinformatics 20, 253–258.

Breiman, L., 1996. Bagging predictors. Machine Learning 24, 123–140.

Chiaromonte, F., Martinelli, J., 2001. Dimension reduction strategies for analyzing global gene expression data with a response. Mathematical Biosciences 176, 123–144.

Culhane, A. C., Perriere, G., Considine, E., Gotter, T., Higgins, D., 2002. Between-group analysis of microarray data. Bioinformatics 18, 1600–1608.

de Jong, S., 1993. SIMPLS: An alternative approach to partial least squares regression. Chemometrics and Intelligent Laboratory Systems 18, 251–253.

Dennis, R. C., Lee, H., 1999. Dimension reduction in binary response regression. Journal of the American Statistical Association 94, 1187–1200.

Dettling, M., Bühlmann, P., 2003. Boosting for tumor classification with gene expression data. Bioinformatics 19, 1061–1069.

Dudoit, S., Fridlyand, J., Speed, T. P., 2002. Comparison of discrimination methods for the classification of tumors using gene expression data. Journal of the American Statistical Association 97, 77–87.

Dudoit, S., Shaffer, J. P., Boldrick, J. C., 2003. Multiple hypothesis testing in microarray experiments. Statistical Science 18, 71–103.

Frank, I. E., Friedman, J. H., 1993. A statistical view of some chemometrics regression tools. Technometrics 35, 109–135.

Freund, Y., 1995. Boosting a weak learning algorithm by majority. Information and Computation 121, 256–285.

Furey, T. S., Cristianini, N., Duffy, N., Bednarski, D. W., Schummer, M., Haussler, D., 2000. Support vector machine classification and validation of cancer tissue samples using microarray expression data. Bioinformatics 16, 906–914.

Garthwaite, P. H., 1994. An interpretation of partial least squares. J.Amer.Stat.Assoc. 89, 122–127.

Ghosh, D., 2002. Singular value decomposition regression modelling for classification of tumors from microarray experiments. Proceedings of the Pacific Symposium on Biocomputing 98, 11462–11467.

Golub, T., Slonim, D. K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J. P. ., Coller, H., Loh, M. L., Downing, J., Caligiuri, M. A., Bloomfield, C. D., Lander, E. S., 1999. Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. Science 286, 531–537.

Hastie, T., Tibshirani, R., Friedman, J. H., 2001. The elements of statistical learning. Springer-Verlag, New York.

Hedenfalk, I., Duggan, D., Chen, Y., Radmacher, M., Bittner, M., Simon, R., Meltzer, P., Gusterson, B., Esteller, M., Raffeld, M., Yakhini, Z., Ben-Dor, A., Dougherty, E., Kononen, J., Bubendorf, L., Fehrle, W., Pittaluga, S., Gruvberger, S., Loman, N., Johannsson, O., Olsson, H., Wilfond, B., Sauter, G., Kallioniemi, O.-P., Borg, A., Trent, J., 2001. Gene-expression profiles in hereditary breast cancer. N Engl J Med 344, 539–548.

Hennig, C., 2004. Symmetric, asymmetric, and robust linear dimension reduction for classification. Journal of Computational and Graphical Statistics (forthcoming).

Huang, X., Pan, W., 2003. Linear regression and two-class classification with gene expression data. Bioinformatics 19, 2072–2078.

Kahn, J., Wei, J. S., Ringner, M., Saal, L. H., Ladanyi, M., Westermann, F., Berthold, F., Schwab, M., Antonescu, C. R., Peterson, C., Meltzer, P. S., 2001. Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks. Nature Medicine 7, 673–679.

Li, L., Weinberg, C. R., Darden, T. A., Pedersen, L. G., 2001. Gene selection for sample classification based on gene expression: study of sensitivity to choice of parameters of the ga/knn method. Bioinformatics 17, 1131–1142.

Martens, H., 2001. Reliable and relevant modelling of real world data: a personal account of the development of PLS regression. Chemometrics and Intelligent Laboratory Systems 58, 85–95.

Martens, H., Naes, T., 1989. Multivariate Calibration. Wiley, New York.

Nguyen, D., Rocke, D. M., 2002. Tumor classification by partial least squares using microarray gene expression data. Bioinformatics 18, 39–50.

Notterman, D. A., Alon, U., Sierk, A. J., Levine, A. J., 2001. Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays. Cancer Research 61, 3124–3130.

Ross, D. T., Scherf, U., Eisen, M. B., Perou, C. M., Spellman, P., Iyer, V., Jeffrey, S. S., de Rijn, M. V., Waltham, M., Pergamenschikov, A., Lee, J. C. F., Lashkari, D., Shalon, D., Myers, T. G., Weinstein, J. N., Botstein, D., Brown,

P. O., 2000. Systematic variation in gene expression patterns in human cancer cell lines. Nature Genetics 24, 227–234.

Schapire, R., Freund, Y., Bartlett, P., Lee, W., 1998. Boosting the margin: A new explanation for the effectiveness of voting methods. Annals of Statistics 26, 1651–1686.

Simon, R. M., Korn, E. L., McShane, L. M., Radmacher, M. D., Wright, G. W., Zhao, Y., 2004. Design and Analysis of DNA Microarray Investigations. Springer-Verlag, New York.

Singh, D., Febbo, P. G., Ross, K., Jackson, D. G., Manola, J., Ladd, C., Tamayo, P., Renshaw, A. A., D'Amico, A. V., Richie, J. P., Lander, E. S., Loda, M., Kantoff, P. W., Golub, T. R., Sellers, W. R., 2002. Gene expression correlates of clinical prostate cancer behavior. Cancer Cell 1, 203–209.

Stone, M., Brooks, R. J., 1990. Continuum regression: cross-validated sequentially constructed prediction embracing ordinary least squares, partial least squares and principal component regression. J.R.Statist.Soc.B 52, 237–269.

Tibshirani, R., Hastie, T., Narasimhan, B., Chu, G., 2002. Diagnosis of multiple cancer types by shrunken centroids of gene expression. PNAS 99, 6567–6572.

Tutz, G., Hechenbichler, K., 2004. Aggregating classifiers with ordinal response structure. Journal of Statistical Computation and Simulation (forthcoming).

West, M., Blanchette, C., Dressman, H., Huang, E., Ishida, S., Spang, R., Zuzan, H., Olson, J., Marks, J., Nevins, J., 2002. Predicting the clinical status of human breast cancer using gene expression profiles. PNAS 98, 11462–11467.

Young, D. M., Marco, V. R., Odell, P. L., 1987. Quadratic discrimination: some results on optimal low-dimensional representation. Journal of Statistical Planning and Inference 17, 307–319.