Visual Methods for Examining SVM Classifiers

Doina Caragea, Dianne Cook, Hadley Wickham, Vasant Honavar

1 Introduction

This chapter focuses on two topics studied empirically using graphics:

1. Ways to find the role different variables play in a classifier, to learn more about the data and go beyond predictive accuracy alone.
2. The variation in the support vector machine fitting when the inputs change.

The availability of large amounts of data in many application domains (e.g., bioinformatics or medical informatics) offers unprecedented opportunities for knowledge discovery in such domains. The classification community has focused primarily on building accurate predictive models from the available data. Highly accurate algorithms that can be used for complex classification problems have been designed. Although the predictive accuracy is an important measure of the quality of a classification model, for many data mining tasks understanding the model is as important as the accuracy of the model itself. Finding the role different variables play in classification provides an analyst with a deeper understanding of the domain. For example, in medical informatics applications, such an understanding can lead to more effective screening, preventive measures and therapies.

The support vector machine (SVM) algorithm [39] is one of the most effective machine learning algorithms for many complex binary classification problems (e.g., cancerous or normal cell prediction based on gene expression data [8]). In the simplest case, SVM algorithm finds a hyperplane that maximizes the margin of separation between classes. This hyperplane is defined by a subset of examples, called support vectors, which "mark" the boundary between classes. However, understanding the results and extracting useful information about class structure, such as what variables are most important for separation, is difficult. SVM is mostly used as a black box technique.

The SVM algorithm searches for "gaps" between clusters in the data, which is similar to how we cluster data using visual methods. Thus, SVM classifiers are particularly attractive to explore using visual methods. In this paper, we use dynamic visual methods, called tours [4, 14, 13], to explore SVM classifiers. Tours provide mechanisms for displaying continuous sequences of low-dimensional linear projections of data in high-dimensional Euclidean spaces. They are generated by constructing an orthonormal basis that represents a linear subspace. Tour-based methods are most appropriate for data that contain continuous real-valued variables. They are useful for understanding patterns, both linear and non-linear, in multi-dimensional data. However, because tours are defined as projections (analogous to an object shadow) rather than slices, some non-linear structures may be difficult to detect. Tours are also limited to applications where the number of variables is less than 20 because otherwise the space is too large to randomly explore within a reasonable amount of time. Hence, when we have more than 20 variables, it is important to perform some dimensionality reduction prior to applying tour methods. In classification problems, tours allow us to explore the class structure of the data, and see the way clusters are separated (linearly or not) and the shape of the clusters.

Visualization of the data in the training stage of building a classifier can provide guidance in choosing variables and input parameters for the SVM algorithm. We plot support vectors, classification boundaries, and outliers in high-dimensional spaces and show how such plots can be used to assess variable importance with respect to SVM, to complement cross-validation methods for finding good SVM input parameters and to study the stability of the SVM classifiers with respect to sampling.

Effective application of machine learning algorithms, SVM included, often requires careful choice of variables, samples and input parameters in order to arrive at a satisfactory solution. Hence, a human analyst is invaluable during the training phase of building a classifier. The training stage can be laborious and time-intensive, but once a classifier is built it can repeatedly be used on large volumes of data. Therefore, it is valuable to take the time to explore alternative variable, samples, parameter settings, plot the data, meticulously probe the data, to generate accurate and comprehensible classifiers.

Our analysis is conducted on a particular data problem, SAGE gene expression data [6], where the task is to classify cells into cancerous cells or normal cells based on the gene expression levels.

The rest of the paper is organized as follows: The **Methods** section describes the algorithms for SVM and tours, and also the aspects that we study to understand and explore the SVM model; The **Application** section illustrates how tour methods can be used to examine SVM classifiers, using a SAGE gene expression data set; The **Summary and Discussion** section summarizes our methods, describes their strengths and limitations, and presents some related work. The web page `http://people.cis.ksu.edu/∼dcaragea/vdm.html` has copies of the color figures and R code to reproduce this work.

2 Methods

2.1 Support Vector Machines

The SVM algorithm [39] is a binary classification method that takes as input labeled data from two classes and outputs a model (a.k.a., classifier) for classifying new unlabeled data into one of those two classes. SVM can generate linear and non-linear models.

Let $E = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_l, y_l)\}$, where $\mathbf{x}_i \in R^p$ and $y_i \in \{-1, 1\}$ be a set of training examples. Suppose the training data is *linearly separable.* Then it is possible to find a hyperplane that partitions the p-dimensional pattern space into two half-spaces R^+ and R^-. The set of such hyperplanes (the solution space) is given by $\{\mathbf{x}|\mathbf{x}\cdot\mathbf{w}+b=0\}$, where $\mathbf{x}$ is the p-dimensional data vector and $\mathbf{w}$ is the normal to the separating hyperplane.

SVM selects among the hyperplanes that correctly classify the training set, the one that minimizes $\|\mathbf{w}\|^2$ (subject to the constraints $y_i(\mathbf{x_i}\cdot\mathbf{w}+b)\leq 1$), which is the same as the hyperplane for which the *margin* of separation between the two classes, measured along a line perpendicular to the hyperplane, is maximized.

The algorithm assigns a weight α_i to each input point $\mathbf{x}_i$. Most of these weights are equal to zero. The points having non-zero weight are called *support vectors*. The separating hyperplane is defined as a weighted sum of support vectors. Thus, $\mathbf{w}=\sum_{i=1}^{l}(\alpha_i y_i)\mathbf{x_i}=\sum_{i=1}^{s}(\alpha_i y_i)\mathbf{x_i}$, where s is the number of support vectors, y_i is the known class for example $\mathbf{x}_i$, and α_i are the support vector coefficients that maximize the margin of separation between the two classes. The classification for a new unlabeled example can be obtained from $f_{\mathbf{w},b}(\mathbf{x})=\mathrm{sign}(\mathbf{w}\cdot\mathbf{x}+b)=\mathrm{sign}(\sum_{i=1}^{l}\alpha_i y_i(\mathbf{x}\cdot\mathbf{x_i})+b)$.

If the goal of the classification problem is to find a linear classifier for a non-separable training set (e.g., when data is noisy and the classes overlap), a set of *slack variables*, ξ_i, is introduced to allow for the possibility of examples violating the constraints $y_i(\mathbf{x_i}\cdot\mathbf{w}+b)\leq 1$. In this case the margin is maximized, paying a penalty proportional to the cost C of constraint violation, i.e., $C\sum_{i=1}^{l}\xi_i$. The decision function is similar to the one for the linearly separable problem. However, in this case, the set of support vectors consists of *bounded support vectors* (if they take the maximum possible value, C) and *unbounded* (*real*) *support vectors* (if their absolute value is smaller than C).

If the training examples are not linearly separable, the SVM works by mapping the training set into a higher dimensional *feature* space, where the data becomes linearly separable, using an appropriate kernel function k.

The SVM algorithm has several input parameters that can be varied (e.g., cost, C; tolerance in the termination criterion, ϵ; kernel function, k) and several outputs that can be studied to assess the resulting model (e.g., support vectors, separating hyperplane, variables that are important for the separation).

In this paper, we use the SVM implementation available in the R [32] package, `e1071` [17]. The SVM implementation in `e1071` is based on the LIBSVM implementation [12]. We use this implementation because the R language allows us to quickly calculate and visualise other diagnostic quantities.

2.2 Tours Methods for Visualization

Tours [4, 41, 42, 9, 16] display linear combinations (projections) of variables, $\mathbf{x'A}$ where $\mathbf{A}$ is a $p\times d(<p)$-dimensional projection matrix. The columns of A are orthonormal. Often $d=2$ because the display space on a computer screen is 2, but it can be 1 or 3, or any value between 1 and p. The earliest form of the tour presented the data in a continuous movie-like manner, but recent developments have provided guided tours [14] and manually controlled tours [13]. Here we use a $d=2$-dimensional manually-controlled tour to recreate the separating boundary between two groups in the data space. Figure 1 illustrates the tour approach.

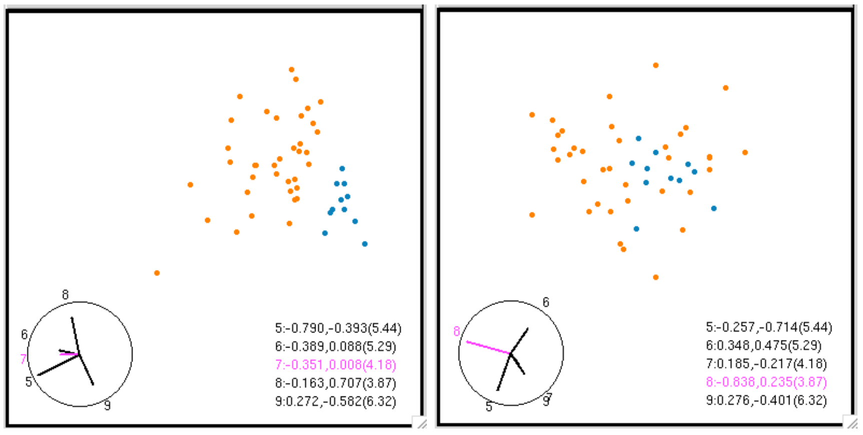

Fig. 1. Two projections from a tour of a 5-dimensional data set (variables denoted V_5,V_6,V_7,V_8 and V_9) where the two groups are separable. The left plot shows a projection where the groups are well-separated, and the plot at right shows a projection where they are not separated. The magnitude of the projection coefficients indicate variable importance, the larger the coefficient - in the direction of separation - the more important the variable. For example, the left plot shows $\frac{-0.790}{5.44}V_5 - \frac{0.389}{5.29}V_6 - \frac{0.351}{4.18}V_7 - \frac{0.163}{3.87}V_8 + \frac{0.272}{6.32}V_9$ horizontally, and $\frac{-0.393}{5.44}V_5 + \frac{0.088}{5.29}V_6 + \frac{0.008}{4.18}V_7 + \frac{0.707}{3.87}V_8 + \frac{-0.582}{6.32}V_9$ vertically.

We use the tour methods available in the data visualization software ggobi [37], and the R [32] package `rggobi` [38] that makes ggobi functionality accessible from R.

2.3 SVM and Tours

Understanding the classifier in relation to a particular data requires an analyst to examine the suitability of the method on the data, adjust the performance of the method (e.g., by appropriate choice of parameters) and adjust the data (e.g., by appropriate choice of variables used for building the classifier) as necessary to obtain the best results on the problem at hand. Tour methods can be used as exploratory tools to examine the outputs of SVM classifiers.

There are several outputs that we can examine when exploring SVM results using tours: support vectors, boundaries between classes, outliers, among others. The support vectors specify the classifier generated by the SVM algorithm. First, we observe their location in the data space and examine their importance (position) relative to the other data points. We expect to see the unbounded support vectors from each group roughly indicating the margin between the groups in some projection. The bounded support vectors should lie inside this margin.

Second, we examine the boundaries between classes in high dimensional spaces. To do this, we generate a rectangular grid of points around the data, by choosing a number of grid points between the minimum and maximum data value of each variable. For example, with two variables, 10 grid values on each axis will give $10^2 = 100$ points on a square grid, or with four variables we would have $10^4 = 10000$ points on a 4D grid. We then compute the predicted values $\mathbf{w} \cdot \mathbf{x} + b$ for each point $\mathbf{x}$ in the grid. The points that are close to the boundary will have predicted values close to 0. For two variables the boundary is a line, for three variables the boundary is a 2D plane, for four variables the boundary is a 3D hyperplane, etc. When using linear kernels with SVM, we expect to see very clear linear boundaries between the two groups. For non-linear kernels, the boundaries will be more complex.

Third, we investigate anomalies in the data, the misclassified samples and the outliers, to get insights about how these points differ from the rest of the data. The anomalies should be isolated from their group in some way. We look at a separate test set after the classifier is built from a training data set and the projection that shows the training separation is found.

The visualization of the outputs can be explored to:

1. Assess the importance of the variables based on the best projections observed;
2. Tune SVM input parameters according to the outputs observed;
3. Study the stability of the model with respect to sampling.

Assessing variable importance Real world data sets are described by many variables (e.g., for gene expression data there are commonly a lot more variables than examples). A classifier may be unreliable unless the sample size is several times as large as the number of variables [34]. Very often, a small number of the variables suffices to separate the classes, although the subset of variables may not be unique [27]. Variable selection is also useful before running tours, because the smaller the space the more rapidly it can be explored. There are many methods for variable selection [21, 22, 33, 7, 19, 26, 44] and the subsets of variables that they return can be very different. Which subset is the best? We use tours to explore and select subsets of variables than can be used to separate the data in two classes.

To do that, we first order the variables according to several criteria (e.g., PDA-PP index [26], BW index [19] and SVM variable importance [22]) and form small subsets, either by taking the best k variables according to one criterion or by combining the best variables of two or more criteria. After running SVM on the subsets formed with the variables selected, we examine the difference between results and select those subsets of variables that show the best separation between classes in some projection.

We can also assess the importance of the variables within a subset. The support vectors from each group roughly indicate a boundary between the groups in some projection. The variables contributing to the projection provide an indication of relative importance of the variables to the separation. The coefficients of the projection (elements of P) are used to examine the variable contribution.

Tuning the parameters The performance of the classifier depends on judicious choice of various parameters of the algorithm. For SVM algorithm there are several inputs that can be varied: the cost C (e.g., $C = 1$), the tolerance ϵ of the termination criterion (e.g., $\epsilon = 0.01$), the type of kernel that is used (e.g., linear, polynomial, radial or Gaussian), and the parameters of the kernel (e.g., degree or coefficients of the polynomial kernel), etc. It is interesting to explore the effect of changing the parameters on the performance of the algorithm. Visual methods can complement cross-validation methods in the process of choosing the best parameters for a particular data set. In addition, examination of the SVM results for different parameters can help understanding better the algorithm and the resulting classifiers.

Stability of the classifier Machine learning algorithms typically trade-off between the classification accuracy on the training data and the generalization accuracy on novel data. The generalization accuracy can be estimated using a separate test set or using bootstrap and cross-validation methods. All these methods involve sampling from the initial data set. It is useful to explore how the sampling process affects the classifier for the particular data at hand. This can be accomplished by studying the variation of the separation boundary, which can provide insights about the stability of the algorithm.

3 Application

3.1 Data description

We use SAGE (Serial Analysis of Gene Expression) [40] data to illustrate the visual methods described in this paper. SAGE is an experimental technique that can be used to quantify gene expression and study differences between normal and cancerous cells [45]. SAGE produces tags (10-base sequences) that identify genes (mRNA). The frequencies of these tags can be seen as a measure of the gene expression levels in the sampled cells. Different from microarray technology, SAGE does not need the sequences of the set of genes to be known. This allows for the possibility that genes related to cancer, but whose sequences or functionality have not been discovered, to be identified. However, SAGE technology is very expensive and there is not much data available.

It is believed that cancers are caused by mutations that alter the normal pattern in gene expression [45]. Genes exhibiting the greatest differences in the expression levels corresponding to normal or cancerous cells are most likely to be biologically significant. One difficulty with SAGE data, when trying to identify genes that distinguish between cancerous and normal cells, is that different samples can come from very different types of tissues (e.g., brain, breast, lung, etc.) as well as from *in vivo* and *in vitro* samples. It is known that different types of tissues are characterized by different expression patterns and they cluster together [28]. The same is believed to be true for *in vivo* and *in vitro* conditions. This makes it difficult to assert that genes whose expression levels are different in cancerous versus non-cancerous cells are indeed responsible for cancer. However, given the scarcity of the data (not too many samples from the same tissues) any findings along these directions are often interesting and potentially useful in clinical applications.

Analysis of SAGE data has received a lot of attention in the last few years. The data set used in our analysis is introduced in [6], which also provides information about the data preprocessing. It was assembled from the complete human SAGE samples (`http://www.ncbi.nlm.nih.gov/sage`) by selecting a subset of tags corresponding to a minimal transcriptome set. Our subset contains the expression values (transcripts per cell) for 822 genes found in 74 human cells. The study in [6] shows that genes with similar functionality cluster together when a strong-association-rule mining algorithm is applied.

3.2 Visualizing SVM outputs

In this section, we show how to explore the SVM outputs using tours. Suppose that the set of important variables for the analyst is $S = \{V800, V403, V535, V596\}$. We apply SVM algorithm on this data set. The results are shown in Figure 2. The two classes are colored with different colors. The support vectors (1 in one class and 3 in the other class) have larger glyphs. The left plot shows a projection where the linear separation found by SVM can be seen. The support vectors line up against each other defining the boundary between the two classes. The coefficients of the projection are also shown. The separation between the two classes is in the top left to bottom right direction, which is a combination of most of the variables.

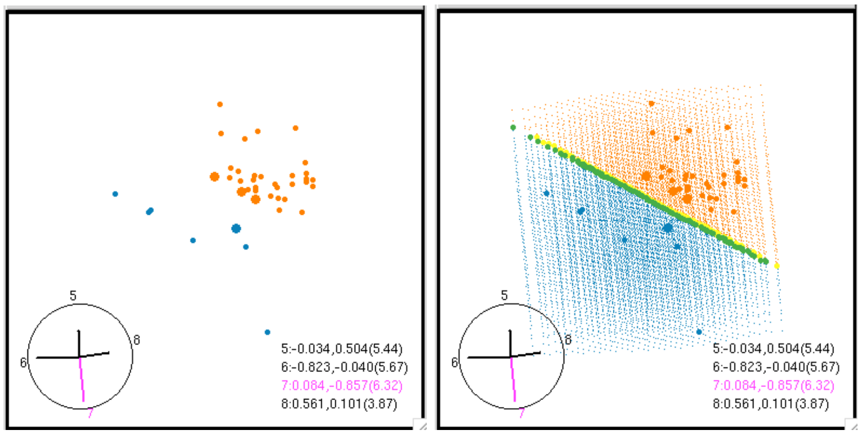

Fig. 2. SVM results for a 4-dim data set. (Left) A projection where the linear separation found by SVM can be seen. (Right) A grid over the data colored according to the class colors. The grid points that have predicted values close to 0, define a nice linear boundary between the two groups.

Using only 4 variables, it is easy to generate a grid around the data. The class of grid points can be predicted using SVM algorithm. Coloring the grid points according to the predictions allows us to see the boundary estimated by SVM. A good separation of the grid can be seen in the middle plot in Figure 2. Coloring the grid points that have predicted values close to 0 allows us to focus on the boundary between the two groups (right plot in Figure 2).

To assess the quality of the model and to find outliers, we divide the examples into training and test sets, build the model for the training data, and find the projection showing the separation between classes. We then plot the test data in the same projection to see how well it is separated, and to examine errors and outliers (Figure 3).

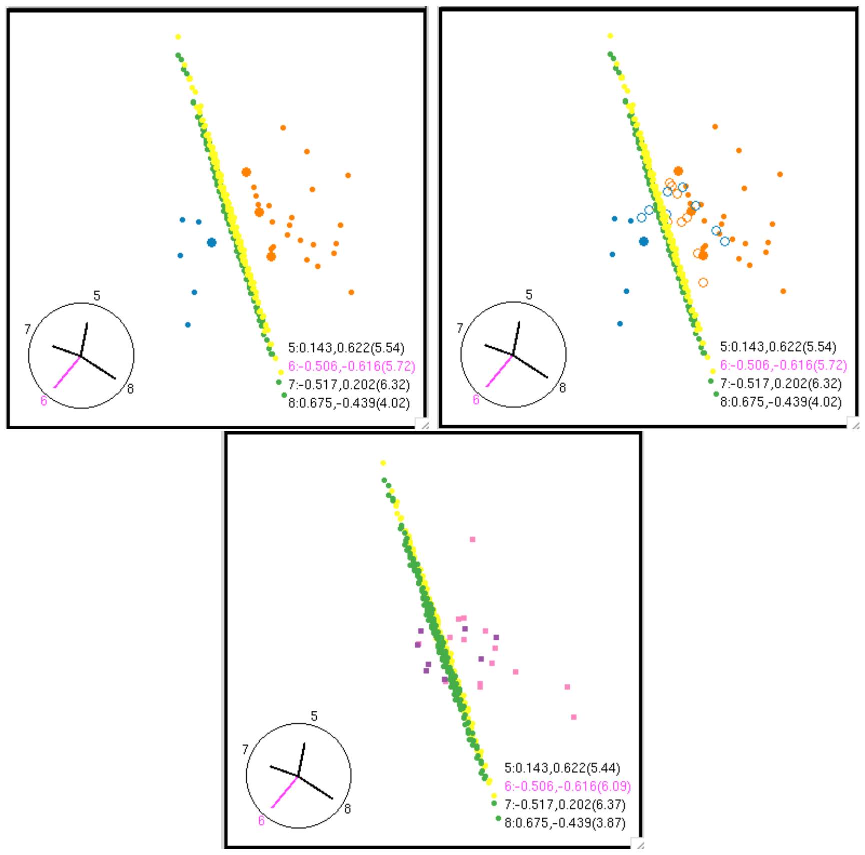

Fig. 3. A subset of the data (2/3 from each class) is used for training and the rest of the data is used for test. (Top left) A projection showing the separation for the training set. Support vectors are shown as larger solid circles. (Top Right) Bounded support vectors are also shown as open circles. (Bottom) The test set is shown with respect to the separating hyperplane found by the algorithm. We can see the errors. They belong entirely to one group (colored purple).

If we use SVM with non-linear kernels, non-linear separations can also be viewed (results presented in an expanded version of this paper [11]).

The ggobi brushing interface allows the user to shadow or show different groups of points, making it very easy to focus on different parts of the model for exploratory analysis. The ggobi main interface allows selecting different groups of variables to be shown. The ggobi identity interface allows identifying points in the plot with points in the data. See [11] for figures showing these interfaces.

The classifly R package [43] automates the process of classifier fitting and grid generation. It does this using rggobi to allow a seamless transition between classifier building in R and visualisation in GGobi. Classifly can display either the complete grid, or just the boundaries of the classification regions.

3.3 Gene Selection

To construct reliable classifiers from SAGE data, we need to select a small set of genes. As Liu et al. [27] have shown, variable selection for gene expression data usually improves the accuracy of the classifier. Variable selection is also necessary in order to make it possible to view the data using tours. As mentioned earlier the initial set has 822 variables (genes), which makes it impossible for visual analysis.

To select small sets of genes, first, data is standardized, so that each gene has mean 0 and variance 1 across samples. Three different methods, BW index [19], PDA-PP index [26] and SVM variable importance [22], are used to order the 822 variables according to their importance with respect to the criterion used by each method. The BW index of a set of genes gives the ratio of their between-group to within-group sums of squares. The PDA-PP index represents a projection pursuit index corresponding to the penalized discriminant analysis [23]. The SVM importance of the variables is determined by running SVM iteratively several times, each time the less important variable - with the smallest w_i^2 - being eliminated. The reverse order of the eliminated variables represents the order of importance [22].

The best 40 genes based on BW index are:

V721, V113, V61, V299, V176, V406, V409, V596, V138, V488, V663, V208, V165, V403, V736, V535, V70, V803, V112, V417, **V357**, V166, V761, V119, V666, V99, V398, V769, V26, V4, V55, V277, V788, V73, V45, V800, V111, V523, V94, V693.

The best 40 genes based on PDA-PP index are:

V721, **V357**, V50, V594, V559, V119, V575, V663, V523, V298, V578, V372, V817, V6, V97, V74, V299, V287, V810, V190, V655, V768, V710, V667, V390, V331, V513, V58, V661, V581, V60, V713, V509, V463, V644, V654, V799, V759, V797, V751

The best 40 genes based on SVM variable importance are:

V390, V389, V4, V596, V54, V409, V741, V398, V725, V736, V581, V263, V817, V701, V655, V165, **V357**, V157, V545, V692, V488, V70, V714, V638, V594, V195, V713, V7, V148, V150, V138, V616, V269, **V721**, V603, V208, V517, V94, V289, V424

In general, SVM takes more time to run than methods such as BW index. Therefore, we also considered the possibility of first ordering all the 822 genes according to the BW index and subsequently ordering the best 40 genes found by BW index according to the SVM variable importance. The result is shown below:

V800, V403, V535, V596, **V357**, V398, V113, V73, V119, V488, V99, V138, V736, V26, V803, V112, V693, V165, V406, V788, V277, V45, V666, V176, **V721**, V663, V417, V769, V208, V111, V523, V761, V55, V166, V94, V299, V409, V70, V61, V4

The set of genes selected by each method is quite different from the others. However, there are two genes that are on the lists of all three methods: **V721** and **V357**. Surprisingly many more genes are common for the BW and SVM gene selection methods: V596, V409, V4, V721, V138, V488, V208, V165, V736, V70, V357, V398, V94.

Given the difference in subsets of important genes, the question is: which one is the best? Not very surprisingly, different subsets of genes give comparable results in terms of classification accuracy, which makes the choice difficult. However, this is where visual methods can help. We use tours to explore different sets of genes and visualize the results of SVM algorithm on those particular subsets. This gives us an idea about how different sets of genes behave with respect to SVM algorithm.

First, to determine how many genes are needed to accurately separate the two classes, we select subsets of the 40 genes and study the variation of the error with the number of genes, using cross-validation. The initial data set is randomly divided into a training set (2/3 of all data) and a test set (1/3 of all data), then SVM is run on the training set, and the resulting model is used to classify both the training set and the test set. The errors are recorded together with the number of unbounded (real) and bounded support vectors for each run. This is repeated 100 time and the average over the measured values is calculated. Then a new variable is added and the process is repeated.

Plots for the variation in average accuracy for the training and test sets (i.e., fraction of the misclassified examples relative to the number of training and test examples, respectively), as well as for the fraction of unbounded support vectors and bounded support vectors (relative to the number of training examples) with the number of variables, are shown in Figure 4. The variables used are the best 20 SVM variables selected from the set of the best 40 BW variables. The average training error decreases with the number of variables and it gets very close to 0 when 20 variables are used. In the test error the average decreases and then starts to rise around 12 variables. There is a dip at 4 variables in both training and test error, and a plateau at 7 variables in the test error. The observed number of unbounded and bounded support vectors shows that there is a negative correlation between the two: as the number of unbounded support vector increases, the number of bounded support vectors decreases. This corresponds to our intuition: as the number of dimensions increases, more unbounded support vectors are needed to separate the data.

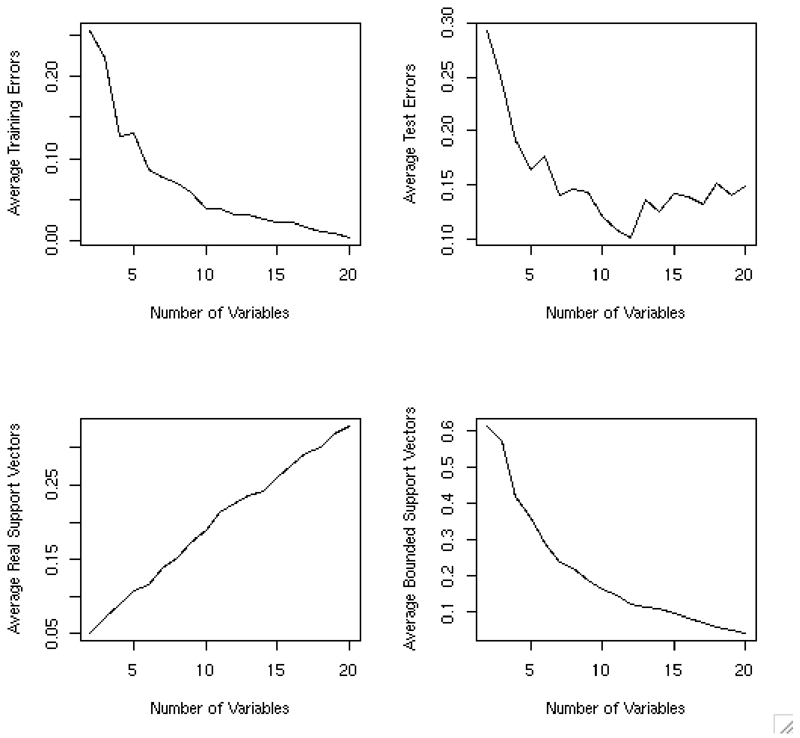

Fig. 4. (Top) The variation of the average (over 100 runs) training and test errors (i.e., fraction of the misclassified examples relative to the number of training or test examples, respectively) with the number of variables. (Bottom) The variation of the average (over 100 runs) fraction of real and bounded support vectors (relative to the number of training examples) with the number of variables.

As the tours are easier to observe when fewer variables are used, we chose to look at sets of 4 variables. Although the errors for 4 variables are slightly higher than the errors obtained using more variables, the analysis should give a good picture of class separations in the data.

We tried various combinations of 4 variable subsets of formed from the lists of most important variables. Some of these subsets gave very poor accuracy, some gave reasonably good accuracy. Table 1 shows a summary of the results obtained for three subsets of variables that give good accuracy: $S1 = \{V800, V403, V535, V596\}$ (first 4 most important SVM genes from the best 40 BW genes), $S2 = \{V390, V389, V4, V596\}$ (first 4 most important SVM genes from all 822 genes) and $S3 = \{V800, V721, V357, V596\}$ (a combination of the 2 important SVM genes from best 40 BW genes and 2 important genes for all three methods).

Table 1. Result summaries for three subsets of 4 variables: $S1 = \{V800, V403, V535, V596\}$, $S2 = \{V390, V389, V4, V596\}$ and $S3 = \{V800, V721, V357, V596\}$. The values are averaged over 100 runs.

Subset	Tr Err	Std Tr	Ts Err	Std Ts	RSV	Std RSV	BSV	Std BSV
S1	0.134	0.032	0.178	0.070	0.084	0.019	0.426	0.051
S2	0.160	0.040	0.195	0.073	0.106	0.019	0.446	0.060
S3	0.166	0.032	0.217	0.063	0.092	0.016	0.426	0.049

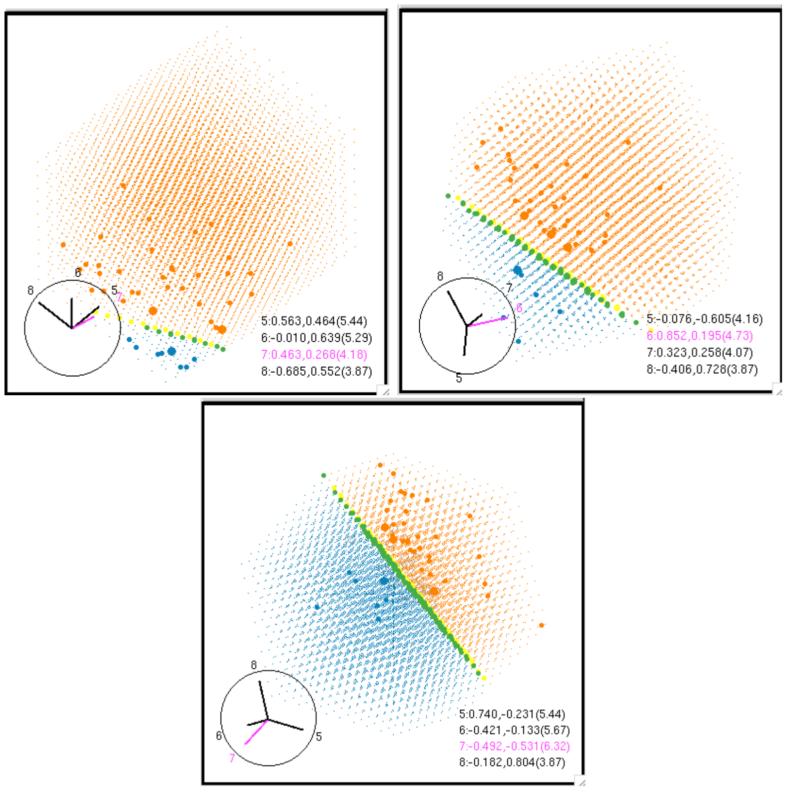

Fig. 5. Visualization of SVM results using three different subsets of the data, corresponding to three different sets of 4 variables. (Top left) Gene subset $S1 = \{V800, V403, V535, V596\}$. (Top right) Gene subset $S2 = \{V390, V389, V4, V596\}$. (Bottom left) Gene subset $S3 = \{800, V721, V357, V596\}$. Note that the subset $S2$ presents the largest separation margin, suggesting that $S2$ may be a better choice than $S1$ and $S3$.

Because the variation in the errors is not significant for the three sets of variables shown in Table 1, we looked at the results of an SVM run with each of these sets of variables (all data was used as training data). The projections where the separation found by SVM can be seen are shown in Figure 5.

Although we get similar accuracy for all three sets *S1,S2,S3*, there is some difference in the results. The set $S1$ has the smallest error, but $S2$ has a larger margin between the real support vectors, suggesting that $S2$ may be a better choice. By examining the coefficients of the projection that shows the best separation for $S2$, we observe that all variables contribute comparably to this projection, therefore we can not conclude that some are more important than others.

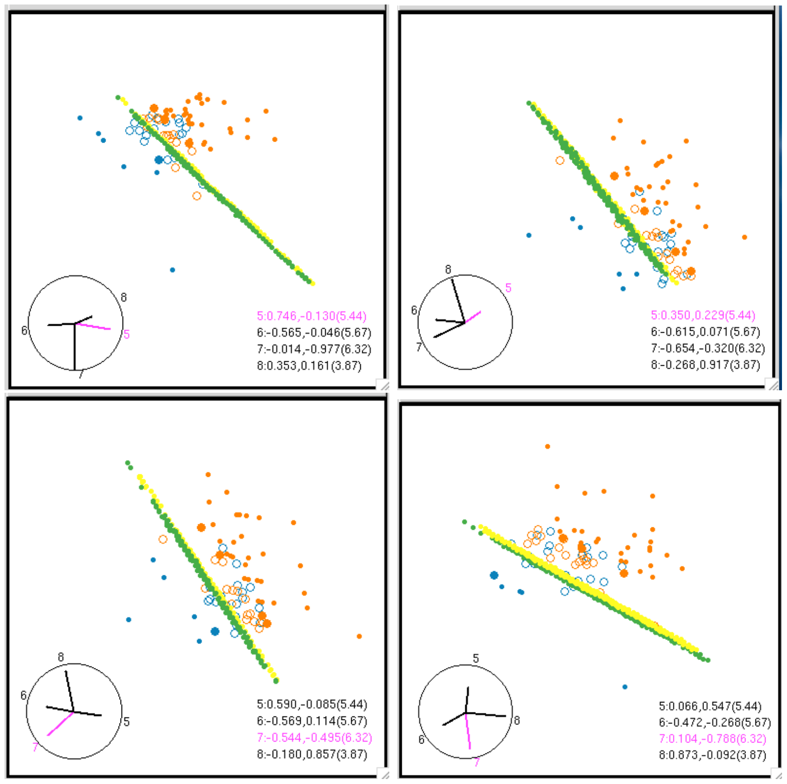

Fig. 6. Variation of the margin with the cost C. Plots corresponding to values C=1, C=0.7, C=0.5, C=0.1 are shown, from top left to bottom right. As C decreases the margin increases, as seen by the increasing distance between support vectors shown as larger solid circles.

3.4 Varying SVM input parameters

In another set of experiments we study the dependence of the margin found by the SVM algorithm on the parameter C. We ran SVM with all the data corresponding to the set $S3 = \{V800, V721, V357, V596\}$ and for each run we found a projection clearly showing the separation. Figure 6 shows the best projections when $C = 1$, $C = 0.7$, $C = 0.5$ and $C = 0.1$. Recall that C can be seen as the cost of making errors. Thus, the higher the C bound the less errors are allowed, corresponding to a smaller margin. As C decreases, more errors are allowed, corresponding to a larger margin. This can be seen in the plots, as you look from left ($C = 1$) to right ($C = 0.1$), the margin around the separating hyperplane increases. Which is the better solution?

Table 2 shows the variation of the training error (the proportion of misclassified examples relative to the number of training examples) with the parameter C. The values corresponding to the plots shown in Figure 6 are highlighted. It can be seen that the smallest error is obtained for $C = 1$, which corresponds to the plot with the smallest margin (or equivalently, the plot with the smallest number of bounded support vectors). However, based on the visual examination, we may choose to use the value $C = 0.1$, as it results in a larger margin and possibly in better generalization error.

Table 2. The dependence of the training error on the parameter C. The highlighted values correspond to the plots shown in Figure 6.

C	**1**	0.9	0.8	**0.7**	0.6	**0.5**	0.4	0.3	0.2	**0.1**
Error	**0.162**	0.148	0.148	**0.175**	0.189	**0.189**	0.175	0.202	0.202	**0.229**

3.5 Model stability

With regard to the dependence of the separation on sampling, the separating hyperplane should not vary much from one training sample to another (we might expect more variability if the data is not separable). To explore this conjecture, we ran the SVM algorithm on all examples using the variables $S3 = \{V800, V721, V357, V596\}$ and identified the bounded support vectors. Then, we removed the bounded support vectors (33 examples), obtaining a linearly separable set (containing the remaining 41). We ran SVM on samples of this set (about 9/10 points were selected for each run), found the projection showing the linear separation and kept this projection fixed for the other runs of SVM. We examined how the separation boundary between the two data sets changes. The results are shown in Figure 7. There is some variation in the separating hyperplane from sample to sample. In some samples the separating hyperplane rotated substantially from that of the first sample, as seen by the thick band of grid points.

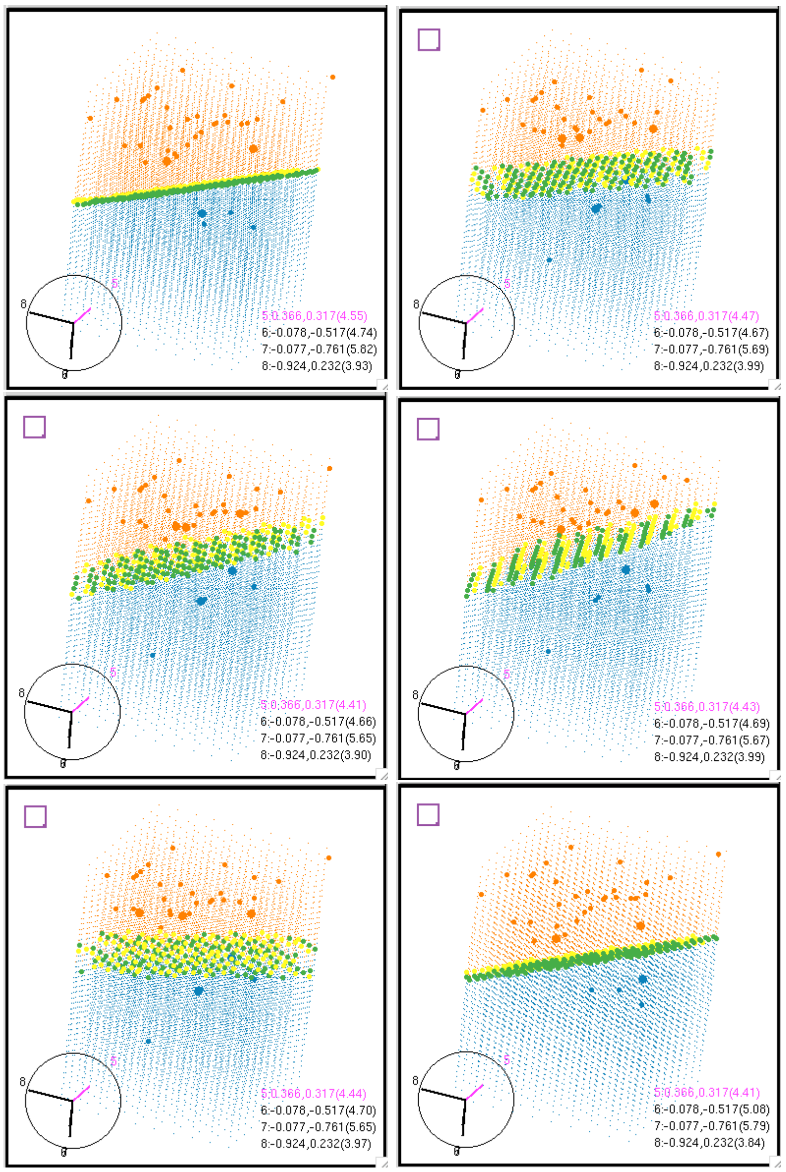

Fig. 7. We examine the variation of the separating hyperplane when sub-sampling the data. We find the projection that shows the linear separation for the first data set and we keep this projection fixed for the subsequent views. There is some variation in the separating hyperplane from sample to sample. In some samples the separating hyperplane rotated substantially from that of the first sample, as seen by the thick band of grid points.

To see how much the coefficients of the variables actually change between samples we start with the projection showing the separation for the first run, we keep this projection fixed and plot the results of the second run, then slightly rotate this second view until the best projection is found for the second run. This is shown in Figure 8. The coefficients change only a tad, with those of variable 6 changing the most.

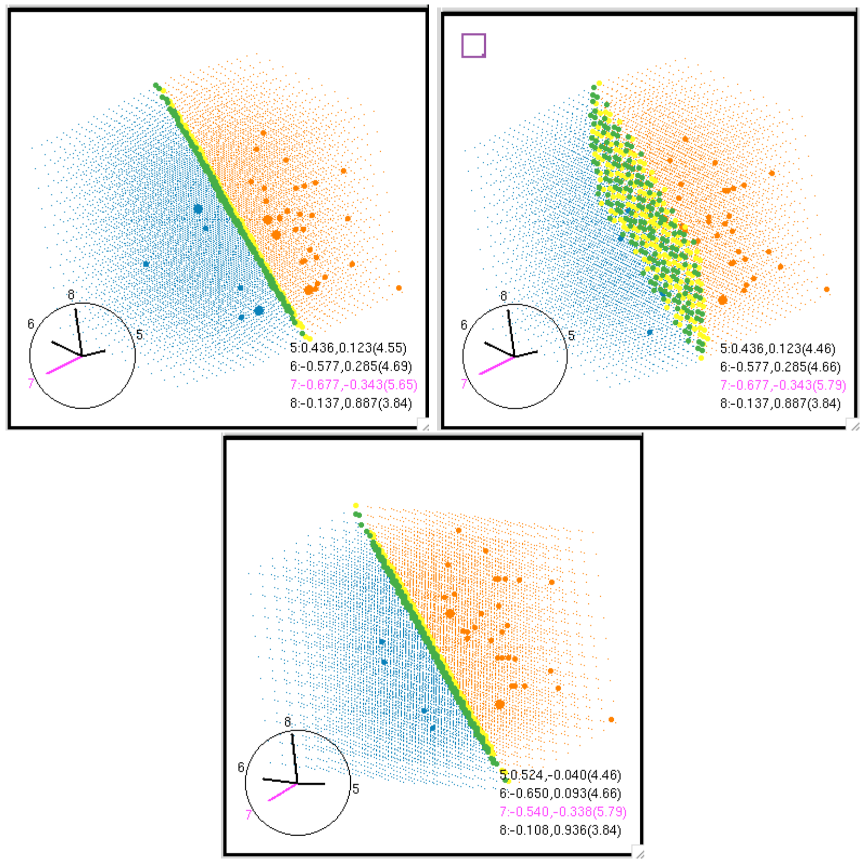

Fig. 8. Two different runs of the SVM with slightly different training data sets (9/10 points of the whole data set are selected at each run). The projection is kept fixed in the (Left) and (Middle) plots. A small rotation of the data shows the clear separation found by the second run of the SVM.

4 Summary and discussion

4.1 Summary

We have presented visual methods that can be used in association with SVM to study many aspects of the model fitting and solution. The reason for using these methods is to gain a better understanding of the model and to better characterize the fit.

We have shown how tour methods can be used to visualize and examine the position of the support vectors found by SVM relative to the other data points and anomalies in the data. They can also be used to explore boundaries between classes in high dimensional spaces. This can be done by generating a rectangular grid around the data and computing the predicted values for each point in the grid. The values close to the boundary will have predicted values close to zero. The main hindrance to drawing the boundary is that the grid of points increases in size exponentially with the number of variables. Hence, alternative ways for showing the boundary are of interest.

We have also shown how we can use visual methods in association with variable selection methods to find sets of variables that are important with respect to the separation found by the SVM algorithm. Finally, we have shown how visual methods can be used to get insights about the stability of the model found by the algorithm and to tune the parameters of the algorithm. Therefore, these methods can be used as a complement to cross-validation methods in the training phase of the algorithm.

We have illustrated the proposed methods on a publicly available SAGE gene expression data set. The implementation of these methods is available to the research community in the R[32] package `classifly`.

4.2 Discussion

The importance of the visual methods in the area of knowledge discovery and data mining (KDD) is reflected by the amount of work that has combined visualization and classification methods during the last few years [35, 20, 1, 25]. Visual methods for understanding results of several classes of classifiers have been proposed, e.g., decision tree classifiers [2, 29], Bayesian classifiers [5], neural networks [36], temporal data mining [3], etc. However, there has been relatively little work on visualizing the results of SVM algorithm in high dimensional spaces, with a few notable exceptions [29, 31, 10, 15].

Poulet [29, 31] has proposed approaches to visualizing the results of SVM. Here, the quality of the separation is examined by computing the data distribution according to the distance to the separating hyperplane and displaying this distribution as a histogram. The histogram can be further linked to other visual methods such as 2-dimensional scatter plots and parallel coordinates [24] in order to perform exploratory data analysis, e.g., graphical SVM parameter tuning or dimensionality and data reduction. These methods have been implemented in a graphical data-mining environment [30]. Similar to our methods, Poulet's approaches have been applied to visualize the results of SVM algorithm applied to bio-medical data [18].

Our previous work [10] has demonstrated the synergistic use of SVM classifiers and visual methods in exploring the location of the support vectors in the data space, the SVM predicted values in relation to the explanatory variables, and the weight vectors, **w**, in relation to the importance of the explanatory variables to the classification. We have also explored the use of SVM as a preprocessor for tour methods, both in terms of reducing the number of variables to enter into the tour, and in terms of reducing the number of instances to the set of support vectors (which is much smaller than the data set). Also in previous work [15], we have shown that using visual tools it is possible not only to visualize class structure in high-dimensional space, but also to use this information to tailor better classifiers for a particular problem.

Acknowledgements

This work was spported by National Science Foundation grants 9982341 and DMS0706949.

References

1. M. Ankerst. Report on the SIGKDD-2002 Panel the Perfect Data Mining Tool: Interactive or Automated. *SIGKDD Explorations*, 4(2), 2002.
2. M. Ankerst, C. Elsen, M. Ester, and H.-P. Kriegel. Visual Classification: An Interactive Approach to Decision Tree Construction. In *Proceedings of ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining*, San Diego, CA, 1999.
3. M. Ankerst, D. Jones, A. Kao, and C. Wang. DataJewel: Tightly Integrating Visualization with Temporal Data Mining. In *Proceedings of the ICDM Workshop on Visual Data Mining*, Melbourne, FL, 2003.
4. D. Asimov. The Grand Tour: A Tool for Viewing Multidimensional Data. *SIAM Journal of Scientific and Statistical Computing*, 6(1):128–143, 1985.
5. B. Becker, R. Kohavi, and D. Sommerfield. Visualizing the Simple Bayesian Classifier. In U. Fayyad, G. Grinstein, and A. Wierse, editors, *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann, 2001.
6. C. Becquet, S. Blachon, B. Jeudy, J. Boulicaut, and O. Gandrillon. Strong Association Rule Mining for Large Gene Expression Data Analysis: A Case Study on Human SAGE Data. *Genome Biology*, 3(12), 2002.
7. J. Bi, K. Bennett, M. Embrechts, C. Breneman, and M. Song. Dimensionality Reduction via Sparse Support Vector Machines. *Journal of Machine Learning Research*, 3, 2003.
8. M. Brown, W. Grundy, D. Lin, N. Christianini, C. Sugnet, T. Furey, M. Ares Jr., and D. Haussler. Knowledge-based Analysis of Microarray Gene Expression Data by using Support Vector Machines. Technical Report UCSC CRL-99-09, Computing Research Laboratory, USSC, Santa Cruz, CA., 1999.
9. A. Buja, D. Cook, D. Asimov, and C. Hurley. Computational Methods for High-Dimensional Rotations in Data Visualization. In C. R. Rao, E. J. Wegman, and J. L. Solka, editors, *Handbook of Statistics: Data Mining and Visualization*. Elsevier/North Holland, `http://www.elsevier.com`, 2005.

10. D. Caragea, D. Cook, and V. Honavar. Gaining Insights into Support Vector Machine Classifiers Using Projection-based Tour Methods. In *Proceedings of the Conference on Knowledge Discovery and Data Mining (KDD)*, San Francisco, CA, USA, 2001.
11. D. Caragea, D. Cook, and V. Honavar. Visual Methods for Examining Support Vector Machines results, with Applications to Gene Expression Data Analysis. Technical report, Iowa State University, 2005.
12. C.-C. Chang and C.-J. Lin. LIBSVM: A Library for Support Vector Machines, 2001. Software available at `http://www.csie.ntu.edu.tw/∼cjlin/libsvm`.
13. D. Cook and A. Buja. Manual Controls For High-Dimensional Data Projections. *Journal of Computational and Graphical Statistics*, 6(4):464–480, 1997.
14. D. Cook, A. Buja, J. Cabrera, and C. Hurley. Grand Tour and Projection Pursuit. *Journal of Computational and Graphical Statistics*, 4(3):155–172, 1995.
15. D. Cook, D. Caragea, and V. Honavar. Visualization for Classification Problems, with Examples Using Support Vector Machines. In *Proceedings of the COMPSTAT 2004, 16th Symposium of IASC*, Prague, Czech Republic, 2004.
16. D. Cook, E.-K. Lee, A. Buja, and H. Wickham. Grand Tours, Projection Pursuit Guided Tours and Manual Controls. In *Handbook of Data Visualization*. Springer, New York, NY, 2007.
17. E. Dimitriadou, K. Hornik, F. Leisch, D. Meyer, and A. Weingessel. `e1071`: Misc Functions of the Department of Statistics, TU Wien. `http://www.r-project.org`, 2006.
18. T.-N. Do and F. Poul. Incremental SVM and Visualization Tools for Bio-medical Data Mining. In *Proceedings of the European Workshop on Data Mining and Text Mining for Bioinformatics*, 2003.
19. S. Dudoit, J. Fridlyand, and T. Speed. Comparison of Discrimination Methods for the Classification of Tumors using Gene Expression Data. *Journal of the American Statistical Society*, 97(1), 2002.
20. U. Fayyad, G. Grinstein, and A. Wierse. *Information Visualization in Data Mining and Knowledge Discovery*. Morgan Kaufmann, 2001.
21. I. Guyon and A. Elisseeff. An Introduction to Variable and Feature Selection. *Journal of Machine Learning Research*, 3, 2003.
22. I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46:389–422, 2002.
23. T. Hastie, R. Tibshirani, and A. Buja. Flexible Discriminant Analysis by Optimal Scoring. *Journal of the American Statistical Association*, 89(428):1255–1270, 1994.
24. A. Inselberg and T. Avidan. The Automated Multidimensional Detective. In *Proceedings of Infovis'99*, pages 112–119, 1999.
25. D. Keim, M. Sips, and M. Ankerst. Visual Data Mining. In C. Johnson and C. Hansen, editors, *The Visualization Handbook*. Academic Press, 2005.
26. E.-K. Lee, D. Cook, S. Klinke, and T. Lumley. Projection Pursuit for Exploratory Supervised Classification. *Journal of Computational and Graphical Statistics*, 14(4):831–846, 2005.
27. H. Liu, J. Li, and L. Wong. A Comparative Study on Feature Selection and Classification Methods Using Gene Expression Profiles and Proteomic Patterns. *Genome Informatics*, 13, 2002.
28. R. T. Ng, J. Sander, and M. C. Sleumer. Hierarchical Cluster Analysis of SAGE Data for Cancer Profiling. In *BIOKDD*, pages 65–72, 2001.
29. F. Poulet. Cooperation between Automatic Algorithms, Interactive Algorithms and Visualization Tools for Visual Data Mining. In *Proceedings of*

VDM@ECML/PKDD'2002, the 2nd Int. Workshop on Visual Data Mining, Helsinki, Finland, 2002.

30. F. Poulet. Full View: A Visual Data Mining Environment. *International Journal of Image and Graphics*, 2(1):127–143, 2002.
31. F. Poulet. SVM and Graphical Algorithms: a Cooperative Approach. In *Proceedings of the Fourth IEEE International Conference on Data Mining (ICDM04)*, 2004.
32. R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2006. ISBN 3-900051-07-0.
33. A. Rakotomamonjy. Variable Selection Using SVM-based Criteria. *Journal of Machine Learning Research*, 3, 2003.
34. B. Ripley. *Pattern Recongnition and Neural Networks.* Cambridge University Press, 1996.
35. T. Soukup and I. Davidson. *Visual Data Mining: Techniques and Tools for Data Visualization and Mining.* John Wiley and Sons, Inc., 2002.
36. M. J. Streeter, M. O. Ward, and S. A. Alvarez. NVIS: An Interactive Visualization Tool for Neural Networks. In *Visual Data Exploration and Analysis VII*, volume 4302, pages 234–241, San Jose, CA, 2001.
37. D. F. Swayne, D. Temple Lang, A. Buja, and D. Cook. GGobi: Evolving from XGobi into an Extensible Framework for Interactive Data Visualization. *Computational Statistics & Data Analysis*, 43:423–444, 2003. Also see `http://www.ggobi.org`.
38. D. Temple Lang, D. Swayne, H. Wickham, and M. Lawrence. `rggobi`: An Interface between R and GGobi. `http://www.r-project.org`, 2006.
39. V. Vapnik. *The Nature of Statistical Learning Theory (Statistics for Engineering and Information Science).* Springer-Verlag, New York, NY, 1999.
40. V. E. Velculescu, L. Zhang, B. Vogelstein, and K. W. Kinzler. Serial Analysis of Gene Expression. *Science*, 270:484–487, 1995.
41. E. J. Wegman. The Grand Tour in k-Dimensions. Technical Report 68, Center for Computational Statistics, George Mason University, 1991.
42. E. J. Wegman and D. B. Carr. Statistical Graphics and Visualization. In C. R. Rao, editor, *Handbook of Statistics, Vol. 9*, pages 857–958. Elsevier Science Publishers, Amsterdam, 1993.
43. H. Wickham. `classifly`: Classify and Explore a Data Set. `http://www.r-project.org`, 2006.
44. E. P. Xing, M. I. Jordan, and R. M. Karp. Feature Selection for High-dimensional Genomic Microarray Data. In *Proceedings of the Eighteenth International Conference on Machine Learning.* Morgan Kaufmann Publishers Inc., 2001.
45. L. Zhang, W. Zhou, V. E. Velculescu, S. E. K. amd Ralph H. Hruban, S. R. Hamilton, B. Vogelstein, and K. W. Kinzler. Gene Expression Profiles in Normal and Cancer Cells. *Science*, 276(5316):1268–1272, 1997.