

Evaluating Feature Selection for SVMs in High Dimensions

Roland Nilsson¹, José M. Peña¹, Johan Björkegren², and Jesper Tegnér¹

¹ IFM Computational Biology, Linköping University, SE58183 Linköping, Sweden,
{rolle,jmp,jespert}@ifm.liu.se

² Gustav V Research Institute, Karolinska Institute, SE17177 Stockholm, Sweden
johan.bjorkegren@ki.se

Abstract. We perform a systematic evaluation of feature selection (FS) methods for support vector machines (SVMs) using simulated high-dimensional data (up to 5000 dimensions). Several findings previously reported at low dimensions do not apply in high dimensions. For example, none of the FS methods investigated improved SVM accuracy, indicating that the SVM built-in regularization is sufficient. These results were also validated using microarray data. Moreover, all FS methods tend to discard many relevant features. This is a problem for applications such as microarray data analysis, where identifying all biologically important features is a major objective.

1 Introduction

In pattern recognition, feature selection (FS) is traditionally viewed as a pre-processing step that simplifies the task of learning classifiers, rather than a learning objective in itself [1]. On the other hand, there is currently considerable interest within the bioinformatics community to apply FS methods to discover biologically important genes (features) that are not captured by traditional statistical testing [2]. For example, in cancer research, the primary interest is to identify all cancer-related genes from microarray data [3]. This is a fundamentally different problem, because many of the biologically important features may not be needed for classification [4], and will therefore be discarded by FS methods that optimize for classification accuracy.

In addition, data dimensionality is growing rapidly: in microarray data analysis, the number of genes (features) is now on the order of 10^4 , whereas sample sizes are small, on the order of 10^2 or less. This is a problem as to the validity of existing FS methods, as they are typically tested on larger sample sizes and much lower dimensions PC. Furthermore, recent classification methods such as support vector machines (SVMs) provide built-in regularization [8], and it is not yet clear whether such methods benefit from FS (which may be viewed as a different type of regularization [9]).

To assess the applicability of FS methods to microarray data, we performed a systematic evaluation of a number of FS methods in conjunction with SVMs. To our knowledge, this study is the first systematic evaluation of feature set

accuracy, and the first to simulate high-dimensional data of the order found in microarray studies.

2 Preliminaries

2.1 Classification accuracy

Throughout, we assume that examples $(x^{(i)}, y^{(i)})$ are independent observations of the random variable pair (X, Y) with distribution $f(x, y)$ on the domain $\mathcal{X} \times \mathcal{Y}$. We will denote components of a vector x by x_i , and restrictions to a given feature set S by $x_S = \{x_i : i \in S\}$. A classifier is defined simply as a function $g(x) : \mathcal{X} \mapsto \mathcal{Y}$, predicting a category y for each observed example x . As we will consider binary classification exclusively in this paper, we will have $\mathcal{Y} = \{-1, +1\}$ throughout. For a given sample size l , a classifier is induced from data $D^l = \{(x^{(i)}, y^{(i)})\}_{i=1}^l \in (\mathcal{X} \times \mathcal{Y})^l$ by an inducer (a learning algorithm), defined as a function $I : (\mathcal{X} \times \mathcal{Y})^l \mapsto \mathcal{G}$, where \mathcal{G} is some set of possible classifiers. The optimal (Bayes) classifier g^* is defined as the one that minimizes the *risk* functional

$$R(g) = \sum_{y \in \mathcal{Y}} p(y) \int_{\mathcal{X}} 1\{g(x) \neq y\} f(x|y) dx, \quad (1)$$

where $1\{\cdot\}$ is the set indicator function [10]. In our simulations we use gaussian densities, for which $R(g)$ is easy to calculate directly from (1), and g^* is unique and can be derived analytically (see appendix B). Of course, for real data $f(x, y)$ is unknown, so $R(g)$ is not observable. However, $R(g)$ can still be estimated using cross-validation or a separate test set [10].

The functional $R(g)$ measures the performance of a specific classifier $g = I(D^l)$, induced from a particular training set D^l . However, for comparing learning algorithms we are more interested in the overall performance of the inducer I , rather than the performance of a particular g [11]. Because the training data D^l is a random variable, the performance of I is described by the distribution of $R(I(D^l))$. Interesting performance measures derived from this density include the *expected* risk

$$\rho = E_D[R(I(D^l))] \quad (2)$$

and probabilities such as $P(R(I(D^l)) > 0.5)$ as well as confidence intervals for ρ . All of these quantities are easily estimated to any desired precision in our simulations.

2.2 Feature selection accuracy

As discussed in the introduction, one may consider two different goals of feature selection, naturally leading to two different ways of assessing performance: if the accuracy of the induced classifier is the main objective, then the expected risk can be used; however, if finding relevant features is the end goal, then we should define accuracy with respect to the set of relevant features instead. Features relevance can be defined as follows [4].

Definition 1. For a given data distribution, the set of relevant features S_{REL} is defined as

$$S_{\text{REL}} = \{i : \exists S : p(y|x_i, x_S) \neq p(y|x_S)\}$$

where $p(y|x_S)$ is the conditional density of Y after observing $X_S = x_S$.

Informally, a feature is considered relevant if it carries some "information" about the target variable Y . Relevant features are either *strongly* or *weakly* relevant; the latter may be "redundant" in the sense that they are not required for optimal classification [4]. Including such features may deteriorate classification accuracy due to increasing design cost [12]. Therefore, to optimize classification accuracy, a FS method must instead find the following feature set.

Definition 2. For a given data distribution, sample size l and inducer I_S , the optimal feature set S_{OPT} is defined as the one that minimizes the expected risk

$$S_{\text{OPT}} = \arg \min_S E_{D_S} [R(I_S(D_S^l))],$$

where I_S is a suitable inducer for the data D_S^l over the feature set S .

Clearly, $S_{\text{OPT}} \subseteq S_{\text{REL}}$, since no irrelevant feature can improve the expected risk. In general, S_{OPT} may not be unique, and the minimization may require an exhaustive search among all subsets of S_{REL} , which is an NP-complete problem [13]. However, in our simulations we found that S_{OPT} was identical to the features used by the Bayes rule g^* , which is easy to determine for our data distributions (see appendix B). Therefore, for simulated data we know the true S_{REL} and S_{OPT} exactly. We then measure feature set accuracy using *precision* and *recall*.

Definition 3. For a selected feature set S , we define the precision and recall with respect to a "true" set T as

$$\text{Precision}(S) = \frac{|S \cap T|}{|S|} \quad \text{Recall}(S) = \frac{|S \cap T|}{|T|}$$

In analogue with the risk measure in the previous section, for describing the performance of a FS algorithm we consider the distribution of these measures when D^l is a random variable, and estimate the expected value of this distribution.

3 Methods

An overview of our simulation/evaluation procedure is shown in figure 1. For a given data distribution $f(x, y)$, we first take a sample D^l to be used as training data (step 1). We then perform FS and classifier (SVM) induction (step 2) as discussed in section 3.1. We calculate classifier error probabilities (steps 3,4) as described in section 2.1. For assessing the accuracy of selected feature sets, we first calculate S_{REL} and S_{OPT} for the given distribution (step 5) and then

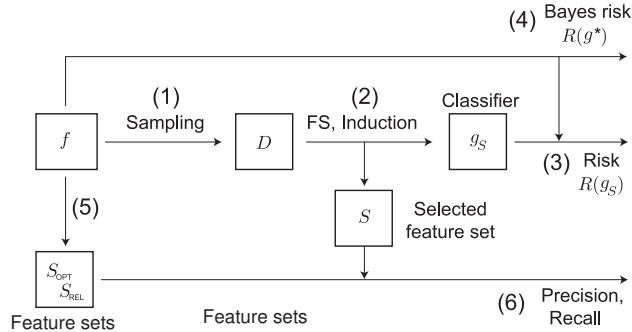


Fig. 1. A schematic view of the evaluation process.

METHOD	TYPE	OUTPUT	REF.
PC	FILTER	RANKING	[1]
WR	EMBEDDED	RANKING	[6]
RFE	WRAPPER	RANKING	[6]
LP-SVM	EMBEDDED	SET	[5]
AROM	WRAPPER	SET	[7]

Table 1. Feature selection methods tested.

measure precision and recall (step 6) as described in section 2.2, with respect to S_{REL} and S_{OPT} .

For evaluating the expected performance of a given algorithm and obtaining statistical confidence measures, this process is repeated a number of times. This yields a number of independent observations of each error measure. Standard non-parametric statistical methods can then be used to obtain correct confidence intervals for each error measure.

3.1 Feature selection methods

We chose five well-known FS methods for evaluation: Pearson Correlation (PC), SVM Naive Weight Rank (WR), Recursive Feature Elimination (RFE), Linear Programming-SVM (LPSVM) and Approximation of the z_0 -norm Minimization (AROM); their properties are briefly described in table 1. We categorize methods into the types "filter", "wrapper" or "embedded" as described by Guyon and Elisseeff [1]. Moreover, we designate methods as "ranking" or "set" depending on whether they output a ranking of features, so that one has determine the number of features $|S|$ by other means, or whether they output a set S , thus determining $|S|$ automatically (see section 4.4). For comprehensive information on these methods, we refer to each respective original paper. Initially, we also tried Radius-Margin Descent [14] and Incremental Associative Markov Blanket [15], but these methods were unfortunately too slow to permit a thorough evaluation

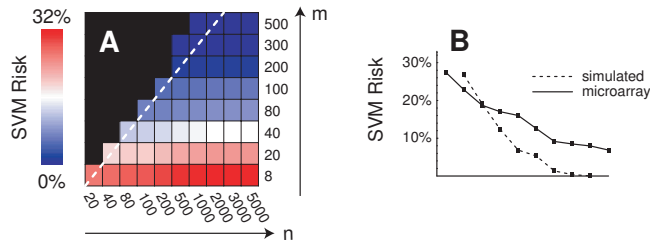


Fig. 2. A: Plot of expected SVM risk ρ vs. number of relevant features m and total number of features n . **B:** Dotted line, plot of SVM risk vs. n for simulations, corresponding to the dotted diagonal in (A). Solid line, SVM risk vs. n for microarray data. Here m is unknown but proportional to n .

and were therefore discarded. We did not consider feature *extraction* methods such as PCA, since these construct new features rather than select among existing ones [1].

Throughout, we used a linear SVM [8] as the inducer $I(D^l)$. For SVM training, we used an in-house Java implementation based on the SMO optimization scheme [16].

4 Results

We used a gaussian data distribution for all simulations. This was designed so that a subset of m features X_1, \dots, X_m were relevant to Y , while X_{m+1}, \dots, X_n were irrelevant. Of the m relevant features, $m/2$ were in the optimal feature set; further, half these ($m/4$) were *marginally* independent of Y and thus undetectable by univariate filter methods like PC. See appendix A for details. We sampled 100 training data points from this distribution and normalized to zero mean and unit variance before applying each method, following standard practise for SVMs [6]. The key parameters to the "difficulty" of the learning problems represented by this data distribution are m and n . We chose a parameter grid $8 \leq m \leq 500$, $20 \leq n \leq 5000$, with values evenly spaced on a logarithmic scale (figure 2A). For each (m, n) we repeated the simulations 100 times.

4.1 The SVM is robust against irrelevant features

The expected risk ρ for the SVM without FS on the (m, n) parameter grid is shown in figure 2A (we set the regularization parameter C to 1; see next section). We find that ρ increases slightly with n , but decreases rapidly with respect to m . Thus, more features is in general better for the SVM: as long as we can obtain a few more relevant features, we can afford to include many irrelevant ones. Therefore, improving SVM performance by FS is very difficult. In particular, an

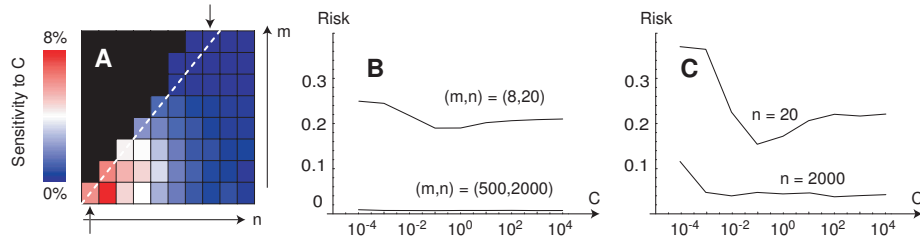


Fig. 3. Sensitivity to the SVM C -parameter. **A:** Sensitivity defined as $\max_C \hat{R} - \min_C \hat{R}$ plotted against m and n . **B:** For simulated data, detailed plot of \hat{R} against C for the cases (m, n) marked by arrows in (A), roughly corresponding to the cases in (C). **C:** For microarray data, plot of \hat{R} against C for $n = 20$ and $n = 2000$.

FS method must provide very good recall (preserve the relevant features), or SVM performance will degrade quickly.

To validate our results, we also tested the FS methods on a large microarray data set consisting of 12,600 features (genes) and 136 samples [17]. For comparison with our simulations, we first extracted the 5000 features with largest variance and then extracted random subsets of sizes $10, \dots, 5000$ from these. Although m is unknown in this case, in expectation this procedure gives a constant m/n ratio, since we draw features with equal probability. This roughly corresponds to a diagonal in figure 2A. We used random subsets of $l = 100$ samples for FS and classifier induction and estimated $\hat{R}(g)$ by the hold-out error on the remaining 36 samples. This procedure was repeated 300 times for each n . The resulting risk estimate was found to agree qualitatively with our simulations (figure 2B).

4.2 The SVM C -parameter has no influence in high dimensions

The value of the regularization parameter C has been found to strongly impact SVM classification accuracy in low dimensions [18]. In our simulations, we optimized C over a range $10^{-4}, \dots, 10^4$ for each (m, n) . We found that C was no longer important in higher dimensions (figure 3A), regardless of the value of m . At lower dimensions, $C \approx 1$ provided good performance (figure 3B), so we fixed $C = 1$ for the remainder of this study. We observed the same (and even stronger) trend for the microarray data (figure 3C). We conclude that the parameter C has virtually no impact on classification accuracy in high dimensions.

4.3 Rankings methods are comparable in high dimensions

Next, we investigated the accuracy of the feature rankings produced by PC, WR and RFE. To address this question without involving the problem of choosing $|S|$ at this stage, we measured precision and recall for $|S| = 1, \dots, n$ and visualized the results using ROC-curves (figure 4). We found that RFE outperforms WR,

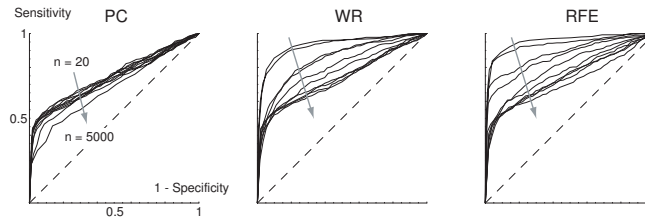


Fig. 4. ROC-curves for the PC, WR and RFE methods. Here we fixed $m = 8$ relevant features and varied $n = 20, \dots, 5000$ as indicated by grey arrows. Dashed diagonals indicate expected result for randomly selected features.

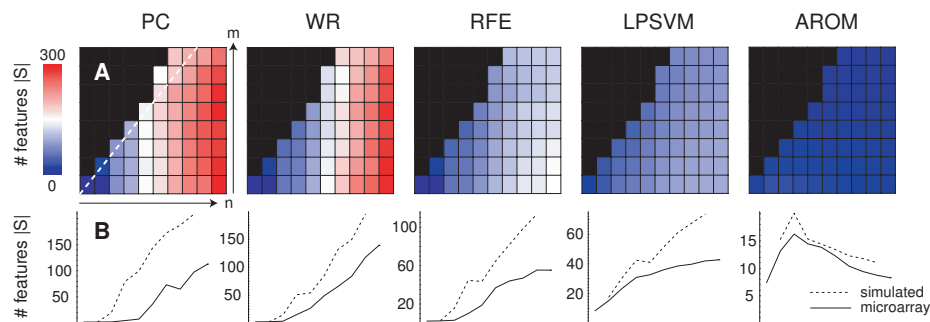


Fig. 5. A: Number of selected features $|S|$ for each (m, n) for simulated data. All plots are scaled equally to $(0, 300)$. **B:** Number of selected features $|S|$ vs. n , corresponding to the dashed diagonal in (A), for simulated and microarray data. Plots are scaled differently.

which in turn outperforms PC, in agreement with Guyon et al. [6]. This was expected, since $1/4$ of the relevant features are not detectable by PC. However, these differences diminished with increasing n . At $n = 5000$, the simpler WR method was as accurate as RFE.

4.4 Number of selected features increases with dimension

To use ranking methods in practise, a critical issue is how to determine $|S|$ (LPSVM and AROM decide this automatically by heuristics that favor small feature sets [5, 7]). A common strategy is to minimize some risk estimate $\hat{R}(g_S)$ for the classifier g_S induced using the feature set S , over a number of possible choices of $|S|$ [19]. For this purpose, we chose the radius-margin bound [20, section 10.7], which provided reasonable estimates in preliminary studies.

Overall, we found that the ranking methods tend to select more features than AROM or LPSVM (figure 5A). We also found that $|S|$ tends to *increase* with n . This can be explained by noting that with better rankings (i.e., higher

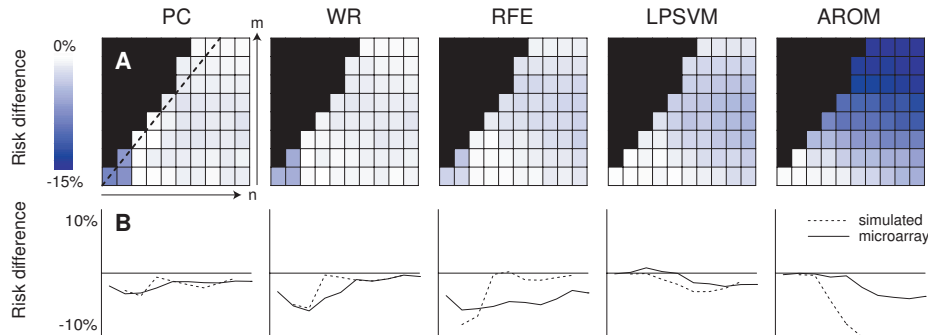


Fig. 6. A: Risk difference $\rho(g) - \rho(g_S)$, using each respective FS method to obtain S (negative means worse accuracy with FS), simulated data. **B:** Estimated risk *vs.* n , corresponding to the dashed diagonal in (A), for simulated and microarray data.

area-under-curve in figure 4), we can reach low classifier risk $R(g_S)$ for small $|S|$. Conversely, as n increases and the rankings degrade, we must choose a larger $|S|$ to optimize $R(g_S)$. Accordingly, PC chooses the largest $|S|$, and RFE the smallest. This was also verified for the microarray data (figure 5B). More surprisingly, there was also a tendency for $|S|$ to *decrease with m* (most evident for RFE). This can be understood in the same fashion: the FS problem becomes harder as m decreases, so that the rankings become more inaccurate, and a larger $|S|$ must be used. We conclude that, by selecting $|S|$ to minimize risk, we obtain methods that attempt to control recall but sacrifice precision (see also section 4.6).

LPSVM produced smaller feature sets than RFE, but otherwise exhibited the same tendencies discussed above. AROM gave the smallest $|S|$ of all methods, but dependence on n was more complicated: $|S|$ was maximal at $n \approx 100$ (similar to the data set used in the original paper [7]) and decreased for $n > 100$. We are not certain as to the cause of this behavior, but we note that AROM merely guarantees convergence to a *local* maxima of its heuristic (the zero-norm); this might be problematic in higher dimensions. Again, the simulation results were consistent with microarray data (figure 5B).

4.5 No FS method improves SVM accuracy

In principle, if $\hat{R}(g_S)$ is accurate, then optimizing this estimate over $|S|$ should at least guarantee that ranking methods do not increase classifier risk; at worst, we should reach an optimum at $|S| = n$. Our simulations verified this intuition. In figure 6A, the difference $\rho(g) - \rho(g_S)$ is close to 0 (where g is the SVM classifier without FS). We did find $\rho(g_S) > \rho(g)$ at some (m, n) for all ranking methods, but the difference was not large. The discrepancy may be because our procedure chose smaller $|S|$ in cases where $\hat{R}(g_S)$ was constant over a range of $|S|$. Overall,

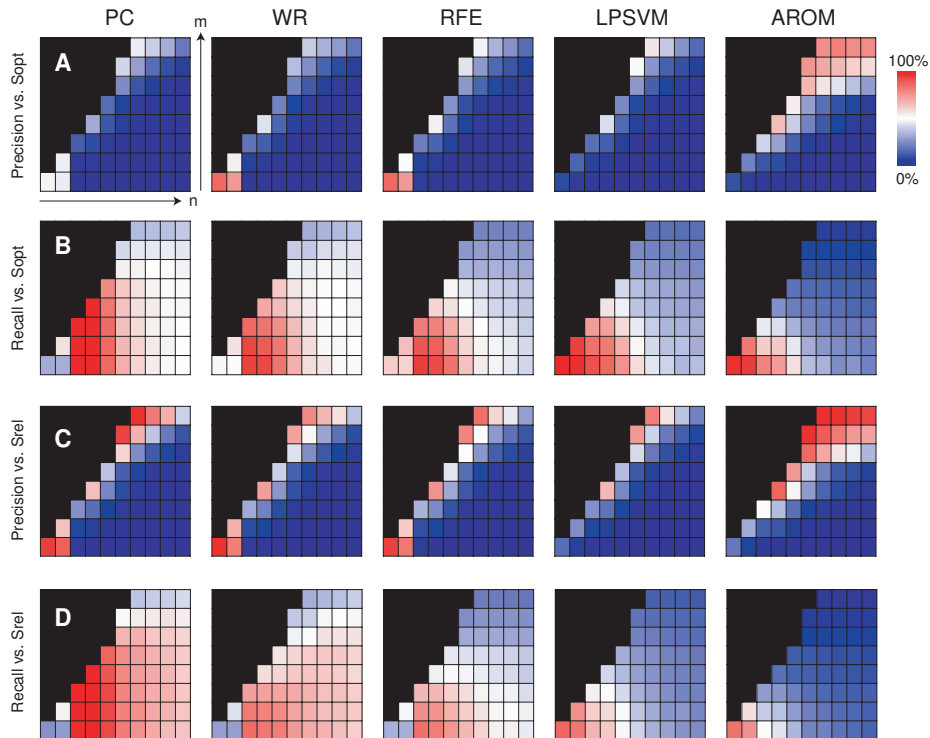


Fig. 7. Feature set accuracy measures for each FS method.

the radius-margin bound seems to be accurate, so our results concerning the ranking methods should be attributed to the rankings themselves.

LPSVM and AROM worked best around $n \approx 100$ (figure 6A,B), corresponding to the data sets used in the original publications [5, 7]. In higher dimensions however, these methods tend to increase the SVM risk. AROM performed particularly bad in this respect, increasing ρ by up to 15%, probably because it insisted on very small feature sets. Results on microarray data were similar (figure 6B) except possibly for RFE, which was less accurate on microarray data. In summary, none of the FS methods improved SVM accuracy.

4.6 Feature set accuracy

For the simulated data, we measured the accuracy of selected feature sets by precision and recall *vs.* S_{OPT} (figure 7A,B) and S_{REL} (figure 7C,D). There are interesting differences between these two feature sets (recall that S_{OPT} constitutes half of S_{REL}). Concerning recall *vs.* S_{REL} , we see that $\text{PC} > \text{WR} > \text{RFE} > \text{LPSVM} > \text{AROM}$. The filter method PC presumably performs best here since it does not distinguish between strong and weak relevance. In fact, we see that PC

selects more weakly than strongly relevant features (since it gives lower recall *vs.* S_{OPT}). In contrast, RFE, LPSVM and AROM have higher recall *vs.* S_{OPT} than *vs.* S_{REL} . All of these methods involve some form of risk optimization, and therefore target S_{OPT} . Consequently, they tend to miss — or, avoid, depending on one’s perspective — many of the weakly relevant features. WR seems to lie somewhere in-between RFE and PC in this respect.

All methods have low precision. AROM provided the best precision, but at the price of lower recall. This is natural since AROM was biased towards very small $|S|$ (figure 5). The remaining methods were comparable.

These results cannot be validated on microarray data since the true feature sets are unknown (which of course is the reason for performing simulation studies in the first place). However, given the good agreement demonstrated in the previous tests (figures 3,5,6), it is likely that the findings in this section also apply to real data.

5 Discussion

A striking trend in our results is that both classification and feature selection (FS) methods behave very differently in high *vs.* low dimensions. For example, while AROM and LPSVM have previously been found to improve classification accuracy for SVMs at lower dimensions with $m \ll n$ [5, 7], we find no such improvement at high dimensions (figure 6). Also, while the SVM C -parameter has been shown to be crucial in low dimensions, it has little or no influence in high dimensions (figure 3). Thus, we recommend that simulation studies of FS methods are performed with dimensions comparable to those found in real data.

None of the FS methods tested improved SVM classification accuracy in high dimensions. To explain this, one may consider FS as an approximate minimization of the L_0 -norm of a vector of feature weights, while the SVM minimizes the L_2 -norm [9]. From this perspective, our results on simulated and real data imply that in high dimensions, the L_2 -norm is simply the better choice. The LPSVM method on the other hand explicitly minimizes the L_1 -norm, and accordingly seems to be closer to SVM performance (figure 6B). However, one should keep in mind that sometimes small $|S|$ is an objective in itself, in which case these FS methods may still be useful. Also, other types of inducers (without built-in regularization) may benefit from FS.

In microarray data analysis, it is common to use statistical testing to control precision (often referred to as the *false discovery rate*) while maximizing recall [21], in order to obtain high quality gene (feature) sets. It is clear from figure 7 and the discussion in section 4.4 that none of the methods tested provide such control. On the contrary, optimizing classifier risk results in a behavior very different from that of statistical tests. On the other hand, the filter method PC (which provided the best recall) cannot detect multivariate relationships. A feature selection method that solves both of these problems would be most useful for microarray data analysis, as a direct multivariate analogue of the statistical methods currently available.

A Data distributions used

We used Bayesian networks [22] to represent data distributions. We used a small network of 4 nodes and simply repeated this distribution to obtain the desired number of relevant features m . The network is defined by the following local (conditional) distributions:

$$\begin{aligned}f(x_1) &= N(x_1|0, 3^2) \\f(x_2|x_1, y) &= N(x_2|y - x_1, 1) \\f(x_3|x_2) &= N(x_3|x_2, 1) \\f(x_4|x_3) &= N(x_4|x_3, 1).\end{aligned}$$

All irrelevant features were $N(x_i|0, 1)$, and we set $p(y) = 1/2$.

B Obtaining the Bayes classifier and the optimal feature set

As seen above, the family of distributions used in this paper are the gaussian distributions

$$f(x, y) = p(y)N(x|y\mu, \Sigma),$$

where $\pm\mu$ are the class-conditional expectations and Σ is the covariance matrix (equal for both classes). It is well-known that for this family the Bayes classifier is given by

$$g^*(x) = \text{sgn}(\mu^T \Sigma^{-1}x).$$

This is a hyperplane through the origin $x = 0$, and therefore the features needed by g^* are exactly those for which the corresponding component of $\mu^T \Sigma^{-1}$ is nonzero. For more details see for example Devroye et al. [10].

Acknowledgements

This work was supported by grants from the Ph.D. Programme in Medical Bioinformatics, the Swedish Research Council (VR-621-2005-4202), Clinical Gene Networks AB and Linköping University.

References

1. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. *Journal of Machine Learning Research* **3** (2003) 1157–1182
2. Dougherty, E.R.: The fundamental role of pattern recognition for the gene-expression/microarray data in bioinformatics. *Pattern Recognition* **38** (2005) 2226–2228 Editorial.

3. Golub, T.R., Slonim, D.K., Tamayo, P., Huard, C., Gaasenbeek, M., Mesirov, J., Coller, H., Loh, M., Downing, J., Caligiuri, M., Bloomfield, C., Lander, E.S.: Molecular classification of cancer: class discovery and class prediction by gene expression monitoring. *Science* **286** (1999) 531–537
4. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artificial Intelligence* **97** (1997) 273–324
5. Fung, G., Mangasarian, O.L.: A feature selection newton method for support vector machine classification. *Computational Optimization and Applications* **28** (2004) 185–202
6. Guyon, I., Weston, J., Barnhill, S., Vapnik, V.: Gene selection for cancer classification using support vector machines. *Machine Learning* **46** (2002) 389–422
7. Weston, J., Elisseeff, A., Schölkopf, B., Tipping, M.: Use of the zero-norm with linear models and kernel methods. *Journal of Machine Learning Research* **3** (2003) 1439–1461
8. Cortes, C., Vapnik, V.: Support-vector networks. *Machine Learning* **20**(3) (1995) 273–297
9. Perkins, S., Lacker, K., Theiler, J.: Grafting: Fast, incremental feature selection by gradient descent in function space. *Journal of Machine Learning Research* **3** (2003) 1333–1356
10. Devroye, L., Györfi, L., Lugosi, G.: A probabilistic theory of pattern recognition. *Applications of mathematics*. Springer-Verlag, New York (1996)
11. Dietterich, T.G.: Approximate statistical tests for comparing supervised classification learning algorithms. *Neural Computation* **10** (1998) 1895–1923
12. Jain, A.K., Waller, W.G.: On the optimal number of features in the classification of multivariate gaussian data. *Pattern Recognition* **10** (1978) 365–374
13. Davies, S., Russel, S.: NP-completeness of searches for smallest possible feature sets. In: *Proceedings of the 1994 AAAI fall symposium on relevance*, AAAI Press (1994) 37–39
14. Chapelle, O., Vapnik, V., Bousquet, O., Mukherjee, S.: Choosing multiple parameters for support vector machines. *Machine Learning* **46** (2002) 131–159
15. Tsamardinos, I., Aliferis, C.: Towards principled feature selection: Relevancy, filters and wrappers. In Bishop, C., Frey, B., eds.: *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*. (2003)
16. Keerthi, S., Gilbert, E.: Convergence of a generalized smo algorithm for svm classifier design. *Machine Learning* **46**(1-3) (2002) 351–360
17. 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.: Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell* **1** (2002) 203–209
18. Keerthi, S.S.: Efficient tuning of SVM hyperparameters using radius/margin bound and iterative algorithms. *IEEE Transactions on Neural Networks* **13**(5) (2002) 1225–1229
19. Ambrose, C., McLachlan, G.J.: Selection bias in gene extraction on the basis of microarray gene-expression data. *PNAS* **99**(10) (2002) 6562–6566
20. Vapnik, V.N.: *Statistical Learning Theory*. John Wiley and Sons, Inc. (1998)
21. Speed, T., ed.: *Statistical Analysis of Gene Expression Microarray Data*. Chapman & Hall (2003)
22. Pearl, J.: *Probabilistic reasoning in intelligent systems*. Morgan Kaufman Publishers, Inc., San Fransisco, California (1988)