Feature Subset Selection for Support Vector Machines by Incremental Regularized Risk Minimization

Holger Fröhlich, Andreas Zell
Center for Bioinformatics Tübingen (ZBIT)
University of Tübingen
72076 Tübingen
Germany
{holger.froehlich,andreas.zell}@informatik.uni-tuebingen.de

***Abstract*— In this paper we present a novel feature selection algorithm for SVMs which works by decreasing the regularized risk in an iterative manner by using a combination of a backward elimination procedure together with an exchange algorithm. It is applicable to linear as well as to nonlinear problems. We test this new algorithm on toy and real life data sets and show its good performance in comparision to state-of-the-art feature selection methods.**

I. Introduction

A. Overview

In many pattern classification tasks we are confronted with the problem, that we have a very high dimensional input space and we want to find out the combination of the original input features which contribute most to the classification. Supposed we want to classify cells whether they are cancer cells or not based upon their gene expressions. Surely on one hand we want to have a combination of genes as small as possible. On the other hand we want to get the best possible performance of the learning machine.

Let us assume we have a training data set $\mathcal{D} = \{(\mathbf{x_i}, y_i)|\mathbf{x_i} \in \mathcal{X}, y_i \in \mathcal{Y}, i = 1, ..., n\}$ (drawn i.i.d. from some unknown probability distribution $P(\mathbf{x}, y)$) where $\mathcal{X}$ is a vector space of dimension d and $\mathcal{Y} = \{+1, -1\}$. Then the problem of feature selection can be formally addressed in the following two ways [13]:

1) Given $m \ll d$, find out the m features that give the smallest expected generalization error; or
2) Given a maximum allowable generalization error γ, find the smallest number m of features.

It can be shown that problem 2 is NP-complete [4].

Unlike e.g. Gaussian Processes in regression, Support Vector Machines (SVMs) do not offer the opportunity of an automated relevance detection and hence algorithms for feature selection play an important role. In the literature two general approaches are known to solve the feature selection problem: The filter approach and the wrapper approach [7]: In a filter method feature selection is performed as a preprocessing step to the actual learning algorithm, i.e. before applying the classifier to the selected feature subset. Features are selected with regard to some predefined relevance measure which is independent of the actual generalization performance of the learning algorithm. This can mislead the feature selection algorithm [1, 7]. Wrapper methods, on the other hand, train the classifier system with a given feature subset as an input and return the estimated generalization performance of the learning machine as an evaluation of the feature subset. This step is repeated for each feature subset taken into consideration.

B. Feature selection as capacity control

A goal of every classifier f is to mimimize the expected generalization error (or risk) over all possible patterns drawn from the unknown distribution $\mathcal{P}(\mathbf{x}, y)$ (see e.g. [10])

$$R[f] = \int_{\mathcal{X}\times\mathcal{Y}} \ell(\mathbf{x}, y, f(\mathbf{x}))d\mathcal{P}(\mathbf{x}, y) \tag{1}$$

with ℓ being some loss-function. However, we cannot compute this quantity as we do not know $\mathcal{P}$. On the other hand it is a crucial insight of Statistical Learning Theory [10] that minimizing the empirical risk (or training error)

$$R_{\text{emp}}[f] = \frac{1}{n}\sum_{i=1}^{n} \ell(\mathbf{x_i}, y_i, f(\mathbf{x_i})) \tag{2}$$

does not guarantee a minimum of $R[f]$. Thus instead we minimize the regularized risk [9]

$$R_{\text{reg}}[f] = R_{\text{emp}}[f] + \lambda\Omega[f] \tag{3}$$

which is an upper bound on $R[f]$. In the case of SVMs one usually chooses $\Omega[f] = \frac{1}{2}||\mathbf{w}||^2$ where $||\mathbf{w}||$ is inverse to the size of the margin and $\|\cdot\|$ is the 2-norm. That means that by maximizing the margin between the two classes +1 and -1 we are minimizing our regularized risk and hence a bound on the true risk.

In SVMs this idea is exactly implemented by solving the dual quadratic program [3, 8]

$$\begin{aligned} &\min_\alpha W^2(\alpha) = \tfrac{1}{2}\textstyle\sum_{i,j=1}^{n} \alpha_i\alpha_j y_i y_j \mathrm{k}(\mathbf{x_i}, \mathbf{x_j}) - \sum_{i=1}^{n} \alpha_i \\ &\text{subj. to } 0 \le \alpha_i \le C,\ i = 1, ...n, \text{ and } \textstyle\sum_{i=1}^{n} \alpha_i y_i = 0 \end{aligned} \tag{4}$$

where k is a kernel function and the constant C regularizes the trade-off between training error and margin maximization.

To perform feature selection we wish to minimize our regularized risk. Hence we should increase the margin between classes +1 and -1. In this way feature selection can be viewed as controling the capacity of the underlying classifier.

One way of doing so is the Recursive Feature Elimination (RFE) algorithm [6]. Let α^* be the solution of (4) with regard to the current feature subset, and let $\mathbf{x^{-t}}$ denote that feature t has been removed from pattern $\mathbf{x}$. Assuming that the set of support vectors does not change significantly when eliminating just one feature, then RFE removes the r features for which the change in margin

$$DW^2 = \left| \sum_{i,j=1}^{n} \alpha_i^* \alpha_j^* y_i y_j \mathrm{k}(\mathbf{x_i}, \mathbf{x_j}) - \sum_{i,j=1}^{n} \alpha_i^* \alpha_j^* y_i y_j \mathrm{k}(\mathbf{x_i^{-t}}, \mathbf{x_j^{-t}}) \right| \quad (5)$$

$(t = 1, ..., d)$ is smallest. Usually r is set to half of the number of existing features. This procedure is repeated until the desired number m of features has been reached. (This refers to problem 1.) As an output of the algorithm we receive a ranking of all features according to the time of their removal and the measure DW^2.

II. Our algorithm

RFE is a powerful and fast feature selection algorithm, but as it uses a greedy strategy to perform backward elimination it can lead to suboptimal solutions. In our algorithm we wish to combine the speed of RFE as a feature ranking algorithm with a method to further improve accuracy of the classifier. Our basic idea is as follows: Given some ranking of all features, we can divide our features in a set S of m features which are used for our classifier and a set R of $d-m$ features which are the removed features. However, there might be features in R which should be combined with some of S to further improve our accuracy, i.e. reduce our regularized risk. If we view our set R as a queue, then naturally the first η features are those which should be tested first to improve our performance. Hence we add them to our set S. Afterwards we remove the η worst features from S according to the RFE criterion (5). These features are then put at the end of the queue. For each feature subset S we calculate the regularized risk. If our regularized risk did not change significantly any more (e.g. less than 10^{-5} in 5 steps in a row), we assume the algorithm to be converged. This is usually achieved after a few loops. We then resort the queue by performing RFE and restart the whole algorithm. If again we converge to the same solution, we stop, otherwise we restart the algorithm.

It is clear that the the inner loop of the algorithm converges after $c \ll d$ steps, because otherwise (if it would not converge) there would always have to be a significant improvement of the regularized risk whenever η features from R are added to S. This would mean that RFE ranked all features exactly in the wrong direction. On the other hand it is clear, that the better the orignal RFE ranking is, the faster convergence will occur.

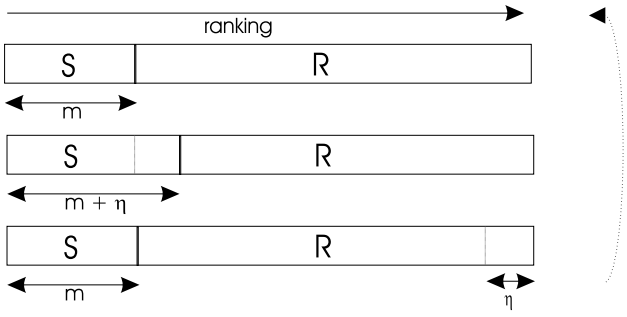

Fig. 1. Basic idea of the IRRM algorithm

The reason, why we do not resort the queue after each step is, that changing just a few features from the queue will not change our ranking significantly. Hence we would put almost the same features at the beginning of our queue as those which we have removed before. Additionally, note that a resorting after each step would impose an inacceptable high computational burden.

The details of the algorithm, which we call **Incremental Regularized Risk Minimization (IRRM)**, are given below:

Algorithm 1 IRRM algorithm

```
perform RFE
S = set of selected features
R = set of removed features (queue)
t = 0
repeat
  S_old = S
  repeat
    R_reg^old = R_reg
    compute R_reg for features in S
    if R_reg^old < R_reg
      restore old S
    C = η highest ranked features
    from R
    S ← S ∪ C
    R ← R − C
    remove η features from S
    according to (5)
    put removed features at end of
    queue R
  until convergence
  resort queue R by means of RFE
  t ← t + 1
until S == S_old AND t > 1
return best solution S*
```

We empirically tested η-values of m, $\frac{m}{2}$, $0.1m$ and 1 and found $\eta = 1$ to perform best. Thus the following results refer to this situation.

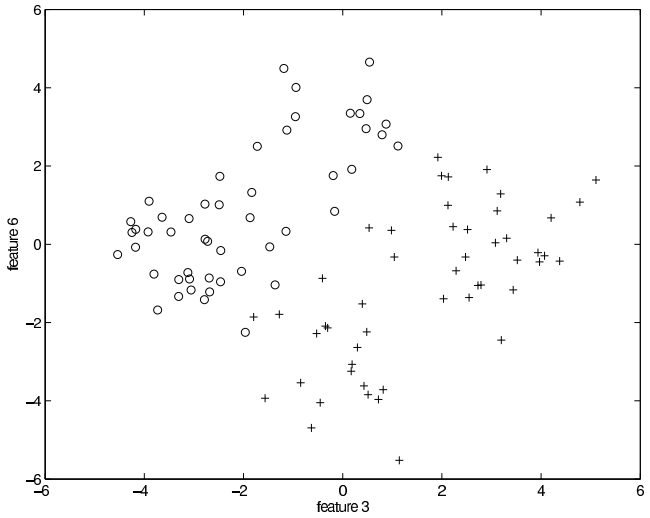

Fig. 2. Linear toy problem: class separation between classes +1 (blue crosses) and -1 (red circles) induced by features 3 and 6

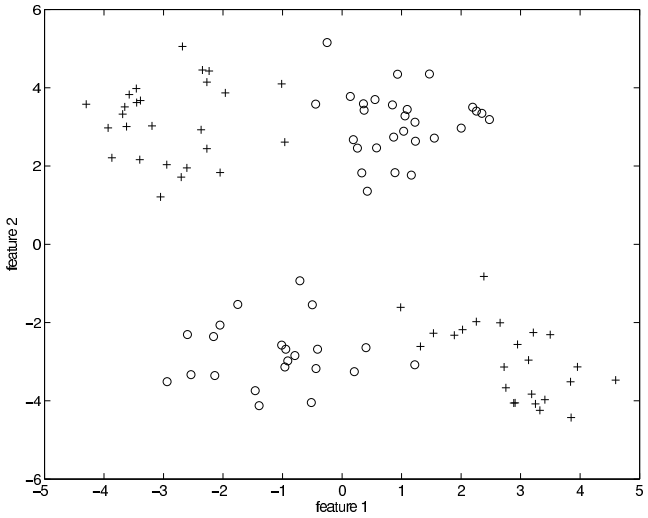

Fig. 3. Nonlinear toy problem: class separation between classes +1 (blue crosses) and -1 (red circles) induced by features 1 and 2

III. Experiments

A. Data Sets

We compare our method to feature selection based on mutual information (e.g. [2]), RFE [6] and (where possible) to the ℓ_2-AROM algorithm [12]. RFE and ℓ_2-AROM are wrapper methods which are especially designed for SVMs. Usually ℓ_2-AROM cannot handle nonlinear problems, but on linear problems Weston et al. showed that it could perform very well.

We investigated the following two artificial and two real life data sets:

a) Linear toy problem: We created 500 training and 10000 independent test points with 202 features following the same method as described in [13]: Six of the 202 features were relevant, but still have inner redundancy. The probability of classes $y = +1$ and $y = -1$ was equal. The first three features X_1, X_2, X_3 were drawn as $X_i = y\mathcal{N}(i, 1)$ and the second three features X_4, X_5, X_6 were drawn as $X_i = \mathcal{N}(0, 1)$ with a probability of 0.7, otherwise the first three were drawn as $X_i = \mathcal{N}(0, 1)$ and the second three as $X_i = y\mathcal{N}(i-3, 1)$. Note that features 3 and 6 are the most important features (fig. 2). The rermaining features are randomly drawn from $\mathcal{N}(0, 20)$.

b) Nonlinear toy problem: We created 500 training and 10000 independent test points with 52 features again following [13]: Two dimensions of 52 were relevant (fig. 3). The probability of classes $y = +1$ and $y = -1$ was equal. If $y = -1$, then the first two features X_1, X_2 are drawn from $\mathcal{N}((-\frac{3}{4}, -3)^T, \mathbf{I})$ or $\mathcal{N}((\frac{3}{4}, 3)^T, \mathbf{I})$ with equal probability. If $y = 1$ then X_1, X_2 are drawn with equal probability from $\mathcal{N}((3, -3)^T, \mathbf{I})$ or $\mathcal{N}((-3, 3)^T, \mathbf{I})$. The remaining features are randomly drawn from $\mathcal{N}(0, 20)$.

c) Lymphoma data set: In the lymphoma problem [5] the gene expression of 96 samples is measured with microarrays to give 4026 features, 61 of the samples are malignant and 35 are labelled "otherwise".

d) Hia data set: The HIA (Human Intestinal Absorption) data set [11] consists of the description of 196 molecules based on 2934 features which were calculated from the 3D structure of the molecules. The molecules are divided into 2 classes "high oral bioavailability" and "low oral bioavailability".

Preparations: All features for the data sets were normalized to mean 0 and standard deviation 1.

For the nonlinear toy problem a polynomial kernel of degree 2 with soft margin parameter $C = 10000$ was taken. For the HIA data set we chose a RBF kernel of width $\sigma = 256$ and parameter $C = 110$. For all other data sets we used a linear kernel with $C = 10000$.

B. Results

Figure 4 and 5 show the test errors on the 10000 independent test points for the linear and the nonlinear toy problem depending on the number of training points which are randomly subsampled from the set of all 500 training points. Each subsampling was repeated 30 times, and the shown test errors are the averages over these 30 trials.

In both cases, for the linear as well as for the nonlinear problem, our algorithm shows the overall best performance. On the linear problem, for more than 30 training points our algorithm is clearly superior to all other methods, and for the nonlinear problem its error rate is always below that of the other methods.

Tables I and II show the cross-validation errors for the real life data sets. On the Lymphoma data set **IRRM** shows the overall best performance. The results show that even with a very low number of features it gives good results.

On the HIA data set our our algorithm works very well, too. It induced the best model of all methods with 16.84% cross-validation error and using just 50 features.

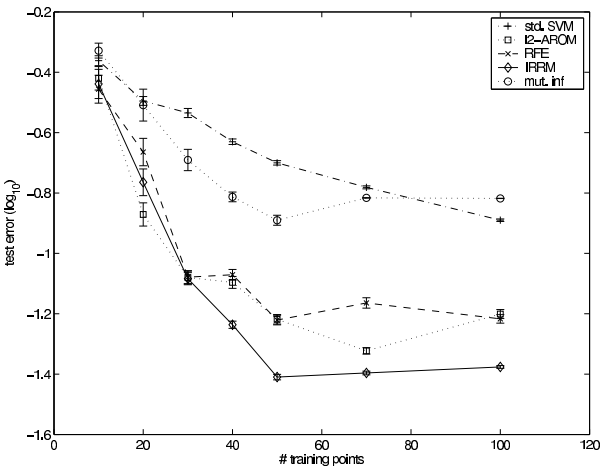

Fig. 4. Linear toy problem: test error in dependency of number of training points

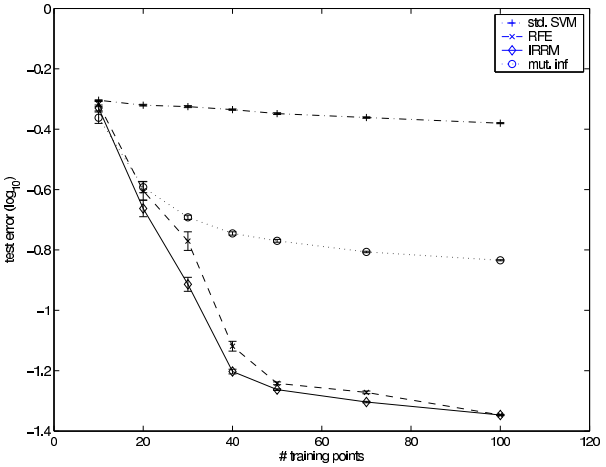

Fig. 5. Nolinear toy problem: test error in dependency of number of training points

TABLE I

LYMPHOMA DATA SET: 8-FOLD CROSS-VALIDATION ERROR $\pm$ STANDARD ERROR (%). THE STANDARD SVM ACHIEVED $28.12\% \pm 3.84\%$.

#features	**IRRM**	ℓ_2-AROM	RFE	mut. inf.
10	4.17 ± 1.58	11.46 ± 4.44	7.29 ± 1.89	11.46 ± 3.84
20	5.21 ± 2.19	12.5 ± 3.15	6.25 ± 3.05	6.25 ± 2.08
50	3.13 ± 1.53	7.29 ± 2.46	4.17 ± 2.23	3.13 ± 1.53
100	2.08 ± 1.36	2.08 ± 1.36	3.13 ± 1.53	4.17 ± 2.22
250	2.08 ± 1.36	2.08 ± 1.36	2.08 ± 1.36	3.13 ± 2.19

TABLE II

HIA DATA SET: 7-FOLD CROSS-VALIDATION ERROR $\pm$ STANDARD ERROR (%). THE STANDARD SVM ACHIEVED $19.38\% \pm 3.47\%$.

#features	**IRRM**	ℓ_2-AROM	RFE	mut. inf.
10	25 ± 3.21	-	24.49 ± 2.86	40.82 ± 2.8
20	19.39 ± 3.1	-	19.39 ± 3	39.8 ± 2.64
50	$\mathbf{16.84 \pm 3.4}$	-	19.9 ± 3	23.47 ± 2.31
100	18.88 ± 3.19	-	17.35 ± 3.16	19.39 ± 3
250	18.88 ± 2.43	-	18.37 ± 3.25	19.39 ± 3.56

IV. CONCLUSION

We have presented a new feature selection algorithm for SVMs which works by incrementally decreasing the regularized risk. It is a combination of a backward elimination and an exchange algorithm. It is fully applicable to linear as well as to nonlinear problems. Our algorithm shows a good performance on toy data and real life data which is at least comparable to state-of-the art methods. On toy data we demonstrated that it is stable against a low number of training points.

It is also worth to mention that our algorithm still has a moderate compution time of $\mathcal{O}(k \cdot (c + \log_2 d))$ SVM trainings where d is the number of input features and $c \ll d$ the number of steps needed for convergence. In our experiments the factor k was always less than 5.

ACKNOWLEDGEMENTS

We thank Altana Pharma for providing us with the HIA data set and Jörg Wegner for computing the corresponding features.

REFERENCES

[1] A. L. Blum and P. Langley. Selection of Relevant Features and Examples in Machine Learning. *Artificial Intelligence*, 97(12):245 – 271, 1997.

[2] B. Bonnlander and A. Weigend. Selecting input variables using mutual information and nonparametric density estimation. In *Proc. 1994 Int. Symp. on Artificial Neural Networks*, pages 42 – 50, 1994.

[3] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273 – 297, 1995.

[4] S. Davies and S. Russel. NP-Completeness of Searches for Smallest Possible Feature Sets. In *Proc. of the 1994 AAAI Fall Symposion on Relevance*, pages 37 – 39, 1994.

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

[6] I. Guyon, J. Weston, S. Barnhill, and V. Vapnik. Gene Selection for Cancer Classification using Support Vector Machines. *Machine Learning*, 46:389 – 422, 2002.

[7] R. Kohavi and G. John. Wrappers for Feature Subset Selection. *Artificial Intelligence*, 97(12):273 – 324, 1997.

[8] B. Schölkopf, C. Burges, and V. Vapnik. Extracting support data for a given task. In U. N. Fayyad and R. Uthurusamy, editors, *First Int. Conf. for Knowledge Discovery and Data Mining*, Menlo Park, 1995. AAAI Press.

[9] B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[10] V. Vapnik. *Statistical Learning Theory*. John Wiley and Sons, New York, 1998.

[11] M. D. Wessel, P. C. Jurs, J. W. Tolan, and S. M. Muskal. Prediction of Human Intestinal Absorption of Drug Compounds from Molecular Structure. *J. Chem. Inf. Comput. Sci.*, 38:726 – 735, 1998.

[12] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero-norm with linear models and kernel methods. *JMLR special Issue on Variable and Feature Selection*, 3:1439 – 1461, 2002.

[13] J. Weston, S. Mukherjee, O. Chapelle, M. Pontil, T. Poggio, and V. Vapnik. Feature selection for SVMs. In S. Solla, T. Leen, and K.-R. Müller, editors, *Adv. in Neural Inf. Proc. Syst. 13*. MIT Press, 2001.