Liknon feature selection: Behind the scenes

Erinija Pranckeviciene
Vilnius University
33 PUBLICATIONS **282** CITATIONS
SEE PROFILE

Ray Somorjai
National Research Council Canada
200 PUBLICATIONS **5,334** CITATIONS
SEE PROFILE

Chapter 8

Liknon feature selection: Behind the scenes

Erinija Pranckeviciene ERINIJA.PRANCKEVICIENE(@MF.VU.LT, @GMAIL.COM)
Department of Human and Medical Genetics,
Vilnius University,
Santariskiu 2, LT-08661 Vilnius-21, Lithuania.

Ray Somorjai RAY.SOMORJAI@NRC-CNRC.GC.CA
Institute for Biodiagnostics,
National Research Council Canada,
435 Ellice Avenue, Winnipeg, MB, Canada.

Editor: Isabelle Guyon, Gavin Cawley, Gideon Dror and Amir Saffari

Abstract

Many real-world classification problems (biomedical among them) are represented by very sparse and high dimensional datasets. Due to the sparsity of the data, the selection of classification models is strongly influenced by the characteristics of the particular dataset under study. If the class differences are not appreciable and are masked by spurious differences arising because of the peculiarities of the dataset, then the robustness/stability of the discovered feature subset is difficult to assess. The final classification rules learned on such subsets may generalize poorly. The difficulties may be partially alleviated by choosing an appropriate learning strategy. The recent success of the linear programming support vector machine (Liknon) for feature selection motivated us to analyze Liknon in more depth, particularly as it applies to multivariate sparse data. The efficiency of Liknon as a feature filter arises because of its ability to identify subspaces of the original feature space that increase class separation, controlled by a regularization parameter related to the margin between classes. We use an approach, inspired by the concept of transvariation intensity, for establishing a relation between the data, the regularization parameter and the margin. We discuss a computationally effective way of finding a classification model, coupled with feature selection. Throughout the paper we contrast Liknon-based classification model selection to the related Svmpath algorithm, which computes a full regularization path.

Keywords: Feature selection, Linear programming, Margin, Transvariation intensity, Transvariation intensity function, Liknon, Regularization parameter C, Full regularization path.

8.1. Introduction

Certain (e.g., biomedical) classification problems, characterized by very sparse and high dimensional datasets, suffer from a generic difficulty: due to the sparsity, the learned classification models are strongly influenced by the characteristics of the investigated dataset. This is manifested by overfitting, caused by sample bias (Zucchini, 2000). Many feature selection strategies and methods (Guyon et al., 2006) and comparisons have been proposed in the literature (Kudo and Sklansky, 2000; Kohavi and John, 1997). Indeed, when the sample size is small and the dimensionality high, the feature selection procedure, driven by the optimization of some criterion that ensures increasing class separation, will adapt to the training data (Ambroise and McLachlan, 2002). “Too much selection can do more harm than good” (Zucchini, 2000). Even if there exist classification models that perform well without feature selection, for the interpretability of the results it is still important to determine the set of "markers" that provide good class discrimination via feature selection, whether filter, wrapper or embedded. If the dataset is "easy", i.e., the class differences are not masked by the noise in the data, one would expect this to be revealed by the validation of the feature selection procedure. Ideally, the classification error estimate will have low variance and the identities of discovered features will not vary appreciably across different random splits of the training data.

Investigations both by other researchers and by us suggest that a feature selection method based on linear programming, originally introduced by (Fung and Mangasarian, 2004), has the desired stability properties and is robust with respect to the sample bias. For a particular application, profiling of gene expression microarrays, for which the data dimensionality exceeds the available number of samples by orders of magnitude, the usefulness of the linear programming support vector machine named Liknon was demonstrated (Bhattacharyya et al., 2003). The method was investigated further and used in practical tasks of face recognition (Guo and Dyer, 2005). It was applied for classification of spectral data (Pranckeviciene et al., 2004). The Liknon feature selection, combined with other classification rules, was among the top-ranked methods in the Agnostic learning vs. Prior knowledge competition (Guyon et al., 2007a).

Useful insights have been gained (Cherkassky and Ma, 2006) on the role of the margin between the classes as an effective measure of the match between data complexity and the capacity of a learning rule. A practical capacity control of a linear rule via the structural risk minimization principle was suggested (Guyon et al., 1992). It is known that the regularization parameter C in kernel-based classification methods controls the tradeoff between maximizing the margin of the classifier and minimizing the margin errors of the training data (Igel, 2005). It is important to use an appropriate C value for an improved generalization performance of the classifier. Usually, the value of this parameter is determined by grid search and crossvalidation.

The formulation of Liknon as a linear programming problem provides a framework for a systematic search of C, computed from the training data. The role of the parameter in Liknon feature selection can be summarized as follows: the non-zero weights of the Liknon discriminant, identifying important features, correspond to those individual data dimensions, for which the absolute difference between the classes is greater than $1/C$. A transvariation intensity function, inspired by the concept of transvariation intensity (Montanari, 2004) applied to the training data, reveals how the parameter C relates quantitatively to the class separation by the margin through the ratio of class overlap over the class difference. This relation leads to an algorithm for the computation of C in Liknon, and a strategy for classification model selection. The closest method to the Liknon-based classification model selection is the Svmpath algorithm (Hastie et al., 2004), which computes the full regularization path for a given classification problem. Throughout the paper we briefly sketch the similarities and differences of the two methods.

The rest of the chapter is organized as follows. In Section 8.2, both Liknon-based and Svmpath-based feature and classification model selection are highlighted, using an artificial dataset, for which classes separate nonlinearly. The details of the primal and dual Liknon formulations are described in Section 8.3. We introduce the transvariation intensity function and analyze it in depth in Section 8.4 in relation to the linear SVM. We derive the relationship between C and the ratio of class overlap over class difference. We illustrate experimentally, that the Liknon and Svmpath algorithms have similar solutions with respect to this ratio. The algorithm for computing C in Liknon is the topic of Section 8.5. The NIPS 2003 feature selection (NIPS 2003 FS) (Guyon et al., 2006) and the recently organized Agnostic Learning versus Prior knowledge (ALvsPK) (Guyon et al., 2007a) competitions provided excellent platforms for controlled experiments with real-life datasets. In Section 8.6 Liknon feature selection is discussed in the context of these two challenges. For the classification experiments, the functions from

PRTools (Duin et al., 2004) and the publicly available Liknon Matlab script (Bhattacharyya et al., 2003) were used. The Matlab code for computing the range of C values from the training data is listed in the Appendix. For the comparison with the Svmpath method, implemented in R (Hastie et al., 2004), Liknon was also implemented in R.

8.2. Liknon and Svmpath based feature and classification model selection when classes separate nonlinearly: case study on artificial Banana dataset

In this section we give an overview of Liknon-based feature/classification model selection, contrasting it with a related Svmpath algorithm using a linear kernel. The efficiency of Liknon feature selection in finding ground truth features was demonstrated on the artificial dataset (Pranckeviciene et al., 2007), in which class separation was due to the difference in the means of two features that separated classes linearly. Liknon feature selection is also applicable where classes separate nonlinearly, but it would find only the “linear part” of feature relevance. The varying margin in the embedded feature selection by Liknon facilitates “retrieving” multivariate feature subsets separating the classes linearly. Liknon will not retrieve the features of classes made of multiple clusters such as checkerboard or concentric classes. A nonlinear boundary between classes should be such, that it can be approximated by linear boundary with some margin. After the feature selection step, other classifiers can be explored with the selected features, aiming to improve the classification performance. The approach consists of two parts: obtaining the most prominent features via Liknon, and using them with other classifiers, or using an ensemble of Liknon discriminants. The winning model is determined by the smallest classification error in k-fold crossvalidation. First, we explain the general computational procedure of the Liknon feature selection and then we study an example. We also apply the related Svmpath algorithm on the same data. To compare fairly the relative computational times of the Liknon and Svmpath methods, the Liknon feature/classification model selection method was re-coded in R (code is available from the first author upon request).

8.2.1. Computational procedure for Liknon-based feature selection

The computational paradigm for Liknon feature selection, using the training data, is k-fold crossvalidation. The number of folds k, usually 5 or 10, is determined by the sample size of the training data. In the outer loop, the data is divided into a training set `Training` and a validation set `Validation`. During classifier development, to account for the variance in the dataset, the `Training` set in the inner loop is randomly partitioned several times into two: a balanced *Training* set and the remaining *Monitoring* set. The classifier development is performed on the `Training` set. The `Validation` set is used for the assessment of the final classification model. In every data partition/split, the number `NM` of Liknon discriminants is identified, based on the *Training* set. The sequence of the `NM` increasing values of the regularization parameter C guides the search for the optimal discriminant. The optimality criterion is the balanced classification error rate (BER) of the *Monitoring* set, computed from the confusion matrix $confmat$ of the classifier:

$$confmat = \begin{pmatrix} TP & FP \\ FN & TN \end{pmatrix}, \quad BER = \frac{1}{2}\left(\frac{FP}{(FP+TP)} + \frac{FN}{(FN+TN)}\right) ,$$

where TP is the number of true positives, FP is the number of false positives, FN is the number of false negatives and TN is the number of true negatives. Increasing the number of features in the discriminant leads to a gradual adaptation to the *Training* set - overfitting, as evidenced by the rapidly decreasing training error. The *Monitoring* set is used to monitor the adaptation and find the best discriminant with the minimum monitoring BER of the particular data split.

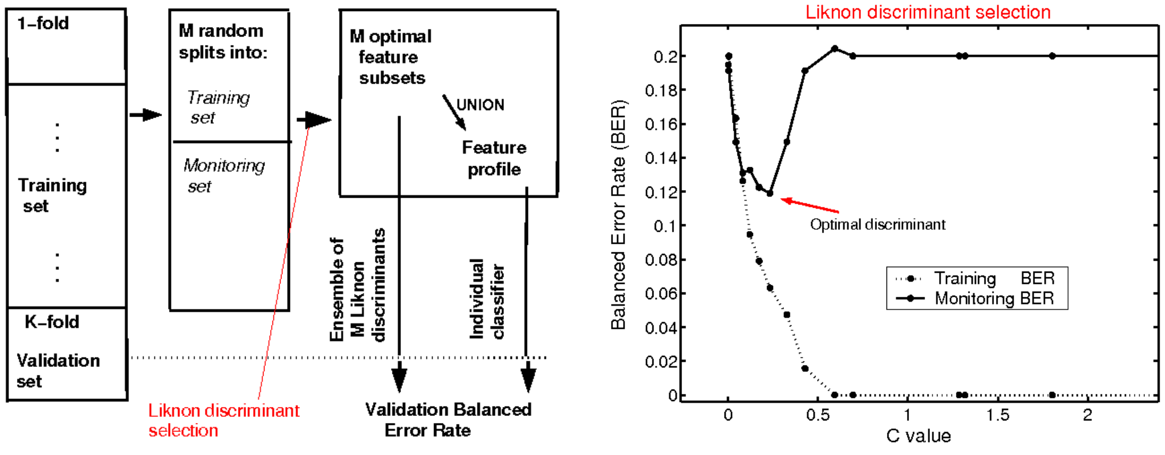

Figure 8.1: Computational scheme of Liknon-based feature/classification model selection. The left panel shows the steps in the crossvalidation procedure. The right panel shows the selection of an optimal Liknon discriminant using the monitoring set.

Every discriminant is associated with a feature subset. The number of important features is different for different discriminants. For small sample sizes for the training and monitoring sets, noisy features will occur in the model. A feature profile is created during the development, by counting the frequency of inclusion of each feature into the optimal discriminant. In the feature profile, peaking occurs for important features. Feature profiles are very important both for interpretation and for exploratory data analysis. The parameters of the presented computational procedure are the number of splits M in the inner loop and the number of Liknon discriminants `NM`. We denote the `Validation` set size as `V=V1+V2`, the balanced *Training* as `T=T1+T2` and the remaining *Monitoring* as `M=M1+M2`.

The overall scheme of the procedure is presented in Figure 8.1. The sequence of the steps in crossvalidation is shown in the left panel. In the right panel, the process of the selection of the optimal Liknon discriminant is illustrated on an artificial data set. The total number of optimization operations in the outlined procedure is $K * M * NM$, where M is the number of resamplings, NM is the number of Liknon optimizations (C values explored) and K is the number of folds.

8.2.2. Identification of useful features

We study Liknon and Svmpath using the noise-augmented Banana dataset, for which the classes separate nonlinearly. The dataset dimensionality is $D = 100$, the sample size is $N_1 + N_2 = 210 + 190$. Features 29 and 30 separate the two classes nonlinearly. The remaining 98 features are overlapping, normally distributed $N(0, 1)$. The class distribution in features containing structure and no structure is presented in Figure 8.2.

The Liknon- and Svmpath- based feature/classification model selection is performed in 5-fold crossvalidation with a single random split in the inner loop. The best solutions of both methods are determined by the minimum monitoring BER. The sizes of the data subdivision in folds are `T1+T2=84+76`, `M1+M2=83+86`, `V1+V2=43+39`, and the number of the tested Liknon models is NM=50. The computations for Liknon were continued until a stable solution

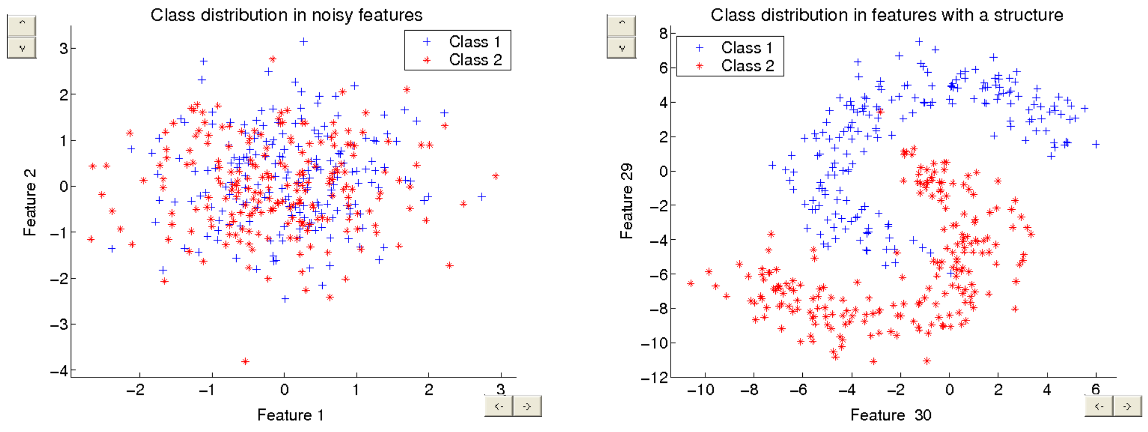

Figure 8.2: Class distribution in the Banana dataset augmented with noise. The right panel shows the nonlinear structure in features 29 and 30. The left panel shows a typical class distribution for the noisy features 1 and 2.

was reached. The feature profiles, identified in every fold, are presented in Figure 8.3 as heat maps, Liknon profiles are on the right, and Svmpath are on the left. The color-coded values of the weights of discriminants may be interpreted as indicators of feature importance. The ground truth features 29 and 30 have large weight in all folds for both algorithms.

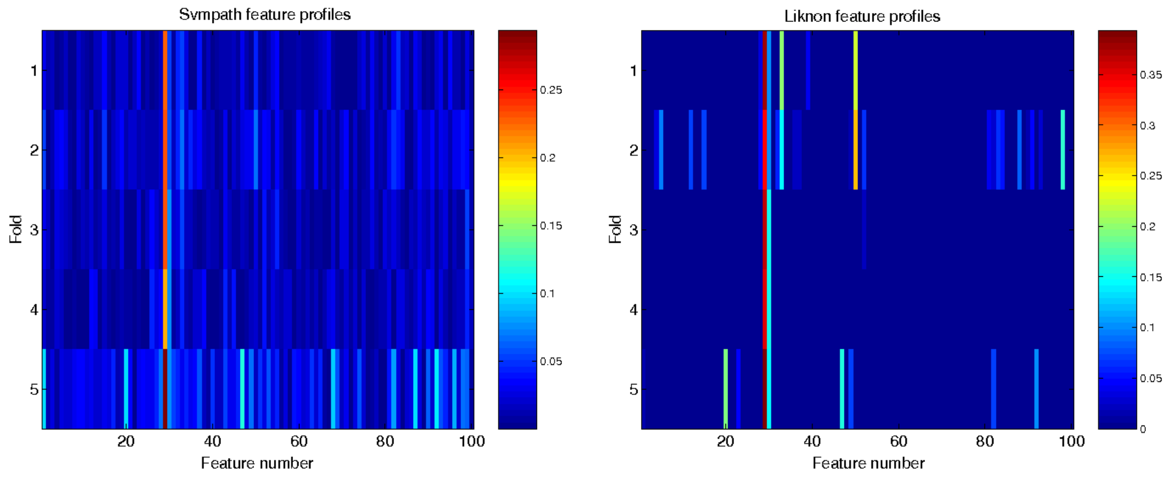

Figure 8.3: Feature profiles identified in 5 folds: Liknon on the right, Svmpath on the left.

The performances of Liknon and Svmpath methods are summarized in Table 8.1. We report the best monitoring BER, validation BER, and the validation BER of a 3nn classifier, trained using the selected features. The most important features in the Svmpath solution were those, for which the absolute value of the weight was above some threshold. Several threshold values, based on a visual examination of the Svmpath feature profiles, were explored. Time, spent by both procedures implemented in R for the computation of the best model, was estimated on a Windows-based 1.20 GHz 256 RAM PC.

Svmpath and Liknon are similar in terms of performance and Liknon's computational edge is not significant statistically. The difference is in their utility for feature selection. No large

Table 8.1: Comparison of the performances of Svmpath and Liknon.

Fold	1	2	3	4	5
Svmpath					
Monitoring BER	0.1271	0.1331	0.0888	0.0860	0.1397
Validation BER	0.1699	0.0826	0.1583	0.099	0.0489
Time (s)	15.99	17.5	16.5	18	16.99
Threshold=0.05					
3nn BER	0.0128	0.000	0.0128	0.0128	0.0385
features	3	5	4	2	23
Threshold=0.035					
3nn BER	0.0244	0.000	0.0489	0.0244	0.0489
features	8	14	6	9	38
Liknon					
Monitoring BER	0.1085	0.1266	0.0942	0.099	0.1463
Validation BER	0.1454	0.0850	0.1595	0.099	0.0620
Time (s)	6.37	12.64	14.47	9.31	8.71
3nn BER	0.0128	0.000	0.0361	0.0128	0.0244
features	6	23	3	2	9

differences are observed in monitoring and validation BER's of the best solutions of Liknon and Svmpath. None of the methods can be claimed superior in this example. The nonlinear 3nn classifier, trained on the selected feature subsets of both Liknon and Svmpath, clearly performs better in all folds on the validation sets, compared to the linear classifiers, derived from Liknon and Svmpath. Note, that the ground truth features 29 and 30 are identified in the solutions of both approaches. From the feature selection point of view, Liknon is more advantageous. Relevant features in the Liknon solution are provided by non zero weights, unlike in Svmpath. Table 8.1 shows, that the final feature subset and consequently the classification result is very sensitive to the threshold value. If the weights of the relevant features are not very distinct, choosing the threshold would present a problem, if one were to use the Svmpath solution for feature selection. Computations for Liknon with a single split in the inner loop take less time than for Svmpath. More splits in Liknon would increase the time, but still within an acceptable range. The perfect classification of the validation set in fold 2 occurs due to an accidental, split-induced data configuration. With real data, one should be cautious about a single optimistic result, which may occur just because of the overly favorable distribution of classes.

The Banana example offered several useful insights. The identified feature subsets provide information, pertinent to classification in various data "projections" in feature and sample spaces. Different classifiers perform differently in these "projections", depending on how well the feature subset represents the nature of the class separation, and the capability of the individual classifier to handle the complexity of the classification problem in the "projection". The monitoring and validation BERs of folds can be analyzed for ranking feature profiles. Liknon generates a feature profile that is optimal for linear separation. Nevertheless, we can use this feature profile as input to another classifier, realizing a nonlinear rule and gain insight into the nonlinearity of the data, yet still keeping the overall architecture simple. By processing many splits in the inner loop, we generate different feature subsets. These subsets can be accumulated into a general feature profile for another classifier, or used in the ensemble of Liknon discriminants as was done in ALvsPK challenge (Pranckeviciene et al., 2007). The justification for using this method in nonlinear separation cases still needs to be formalized and experiments need to be carried out in order to compare it with other methods, aiming to reveal possible computational or statistical advantages. In the following section the mathematical formulation of Liknon is outlined.

8.3. Liknon formulation

Liknon implements a linear rule for two-class classification:

$$y_s = \mathrm{sign}(\mathbf{x}_s \mathbf{w}^T + w_0) \quad , \tag{8.1}$$

where $\boldsymbol{x}_s = [x_s^1, \dots, x_s^D]$ are D-dimensional samples, $\boldsymbol{y} = [y_1, \dots, y_N]$ is the vector of class labels, assuming values $+1$ for the positive class and -1 for the negative class, s indexes the sample number and $N = N_1 + N_2$ is the total number of samples in the two classes. The transpose is denoted by $^{\mathbf{T}}$. The * denotes the optimal solution and the ξs are slack variables. The weight vector $\boldsymbol{w}$ is obtained by solving the optimization problem:

$$(\boldsymbol{w}^*, \dots, \xi_1^*, \dots, \xi_N^*) = \underset{(\boldsymbol{w}, \xi_1, \dots, \xi_N)}{\arg\min} \left(\|\boldsymbol{w}\|_1 + C \textstyle\sum_{s=1}^N \xi_s \right) \quad , \tag{8.2}$$
$$\text{s.t.:}$$
$$y_s \left(\boldsymbol{x}_s \boldsymbol{w}^T + w_0 \right) + \xi_s \geq 1, \;\; \xi_s \geq 0, \;\; s = 1, \dots, N \;\; .$$

The L_1 norm is $\|\boldsymbol{w}\|_1 = \sum_{f=1}^D |w_f|$. The formulation (8.2) is the same as for the linear SVM, except for the L_1 norm of the regularization term. The solution $\boldsymbol{w}^*$ is sparse. Because of sparsity, it is used for feature selection. Features, important for classification, are identified by large weights w_f^* of the vector $\boldsymbol{w}^*$. The regularization parameter C controls the level of sparseness. Formulation (8.2) is cast into a linear programming (LP) optimization problem, and $\boldsymbol{w}^*$ is obtained using an LP solver. The primal and dual LP optimization problems are related. In the primal optimization problem, the regularization parameter C appears in the cost function. In the dual it appears in the constraints.

8.3.1. The primal minimization problem

In order to present the minimization problem (8.2) in a form suitable for a general linear program solver, the variables in the objective function should be positive. Thus, every w_f variable is modeled by two non-negative variables u_f and v_f, a common practice in LP of changing the negative variables into positive ones (Arthanari and Dodge, 1981, page 32):

$$w_f = u_f - v_f, \; |w_f| = u_f + v_f \quad . \tag{8.3}$$

The pair of variables u_f, v_f, simultaneously satisfying conditions (8.3) is unique, given that only three choices are possible for the value of w_f: (i) $u_f = 0, \;\; v_f = 0, \;\;$ and $w_f = 0$; (ii) $u_f = 0, \;\; v_f \neq 0, \;\;$ and $w_f = -v_f$; (iii) $u_f \neq 0, \;\; v_f = 0, \;\;$ and $w_f = u_f$. The problem in (8.2) is reformulated in a form suitable for LP, by changing the variable w_f into the combination of v_f and u_f according to (8.3). The original formulation (8.2), after the change of variables, becomes

(Bhattacharyya et al., 2003):

$$(u_1^*,\ldots,u_D^*,v_1^*,\ldots,v_D^*,\xi_1^*,\ldots,\xi_N^*) = \underset{(u_1,\ldots,\xi_N)}{\arg\min} \left(\sum_{f=1}^D (u_f+v_f) + C\sum_{s=1}^N \xi_s\right),$$
$$\text{s.t.:}$$
$$\sum_{f=1}^D u_f y_s x_s^f - \sum_{f=1}^D v_f y_s x_s^f + y_s u_0 - y_s v_0 + \xi_s \geq 1, \quad \xi_s \geq 0, \quad u_f \geq 0, \quad v_f \geq 0 \ ,$$
$$u_0 \geq 0, \quad v_0 \geq 0 \quad f=1,\ldots,D,\ s=1,\ldots,N \ . \tag{8.4}$$

The constant C in (8.4) is the regularization parameter.

8.3.2. Duality of linear programming

The primal and dual problems of Linear Programming are related (Papadimitriou and Steiglitz, 1982):

$$\text{minimize } J_{\min}(\boldsymbol{x}) = \boldsymbol{c}\boldsymbol{x}, \quad s.t.: \ \boldsymbol{A}\boldsymbol{x} \geq \boldsymbol{b}, \quad \boldsymbol{x} \geq 0 \ . \tag{8.5}$$

$$\text{maximize } J_{\max}(\boldsymbol{z}) = \boldsymbol{b}^T\boldsymbol{z}, \ s.t.: \ \boldsymbol{A}^T\boldsymbol{z} \leq \boldsymbol{c}, \quad \boldsymbol{z} \geq 0 \ . \tag{8.6}$$

In (8.5) and (8.6), $\boldsymbol{x}$ and $\boldsymbol{z}$ denote the variables of primal and dual, $\boldsymbol{c}$ is a vector of costs and $\boldsymbol{b}$ is a vector of constraints of the primal, $J_{\min}$ and $J_{\max}$ denote the objective functions, $\boldsymbol{A}$ is a data matrix. The Liknon primal (8.4) and dual are related through the following data matrix $\boldsymbol{A}$:

$$\boldsymbol{A} = \begin{pmatrix} y_1x_1^1 & \ldots & -y_1x_1^D & y_1 & -y_1 & 1 & \ldots & 0 \\ \vdots & & \vdots & \vdots & \vdots & \ddots & & \\ \vdots & & \vdots & \vdots & \vdots & & \ddots & \\ y_Nx_N^1 & \ldots & -y_Nx_N^D & y_N & -y_N & 0 & \ldots & 1 \end{pmatrix} . \tag{8.7}$$

The costs $\boldsymbol{c}$ and the constraints $\boldsymbol{b}$ of (8.5) correspond to the costs and constraints of Liknon's primal minimization problem (8.4):

$$\boldsymbol{c} = [1_1,\ldots,1_{2D},0,0,C_1,\ldots,C_N] \quad \text{and} \quad \boldsymbol{b} = [1_1,\ldots,1_N]^T \ .$$

The variables of primal (8.5) and dual (8.6) correspond to the Liknon variables:

$$\boldsymbol{x} = [u_1,\ldots,u_D,v_1,\ldots,v_D,u_0,v_0,\xi_1,\ldots,\xi_N] \quad \text{and} \quad \boldsymbol{z} = [z_1,\ldots,z_N] \ .$$

For consistency with SVM terminology, for Liknon dual variables we use α instead of z.

8.3.3. The dual maximization problem

The Liknon dual is obtained straightforwardly from the primal. The costs $\boldsymbol{c}$ of the primal (8.5) become the constraints of the dual (8.6). Similarly, the constraints of the primal become the costs for the dual. Using the data matrix (8.7) of Liknon, its dual maximization problem (8.6) is formulated as follows:

$$(\alpha_1^*,\ldots,\alpha_N^*) = \underset{(\alpha_1,\ldots,\alpha_N)}{\arg\max} \left(\sum_{s=1}^N \alpha_s\right) \ ,$$
$$\text{s.t.:}$$
$$\pm y_1x_1^f\alpha_1 \pm \ldots \pm y_Nx_N^f\alpha_N \leq 1, \quad y_1\alpha_1 + \ldots + y_N\alpha_N = 0, \quad 0 \leq \alpha_s \leq C \ ,$$
$$s=1,\ldots,N, \quad f=1,\ldots,D \ . \tag{8.8}$$

The Liknon dual variables α are positive real numbers.

8.3.4. Optimality conditions

The optimal solutions of the primal and dual satisfy the optimality conditions for every feature $f = 1,\ldots,D$:

$$u_f^* \left(\sum_{s=1}^N y_s x_s^f \alpha_s^* - 1\right) = 0, \quad v_f^* \left(\sum_{s=1}^N y_s x_s^f \alpha_s^* + 1\right) = 0, \quad \xi_s^* \left(\alpha_s^* - C\right) = 0 \ . \tag{8.9}$$

For every sample $s = 1,\ldots,N$, the optimality conditions are:

$$\alpha_s^* \left(\sum_{f=1}^{D} y_s x_s^f (u_f^* - v_f^*) + y_s(u_0^* - v_0^*) + \xi_s^* - 1 \right) = 0 \ . \tag{8.10}$$

It is important to note that the binding constraints determine the non-zero variables of the optimal solutions. Therefore, the non-zero components w_f of $\boldsymbol{w}$ in (8.1), corresponding to the selected features, are determined solely by the constraints that become binding for u_f and v_f in (8.9).

A theoretical basis for understanding the algorithm for computing the range of C values from the training data relies on the concept of transvariation intensity.

8.4. Transvariation intensity and `margin`

Based on the concept of transvariation suggested by Gini and explained by Montanari, several class/group separation measures are defined (Montanari, 2004): transvariation area, transvariation probability and the class separation measure based on the transvariation intensity. The transvariation probability was used successfully in classification of high-dimensional data in low-dimensional projected feature spaces (Somorjai et al., 2007). We propose a modification of the transvariation intensity, to be used in our study.

8.4.1. Univariate class separation measure, based on transvariation intensity

The two classes, mapped onto a line and represented by $x_i, \ \ i = 1,\ldots,N_1$ and $x_j, \ \ j = 1,\ldots,N_2$ transvary around their corresponding mean values m_1 and m_2 if the sign of any of the $N_1 N_2$ differences $x_i - x_j$ is opposite to the sign of $m_1 - m_2$, where N_1 and N_2 are the number of samples in the two classes and the indices i and j are used to distinguish the samples x_i and x_j of the two classes. Such pair is called a transvariation and the absolute difference $|x_i - x_j|$ is its intensity. The class separation measure, based on the transvariation intensity, is defined as:

$$i_{tran} = \frac{2\sum_{i=1}^{N_1}\sum_{j=1}^{N_2} |x_i - x_j|}{\sum_{i=1}^{N_1}\sum_{j=1}^{N_2} |(x_i - m_1) - (x_j - m_2)|} \ . \tag{8.11}$$

The denominator in (8.11), divided by $N_1 N_2$, is the Gini mean difference. We will be using the expression of the denominator in (8.11), called the transvariation intensity maximum (Montanari, 2004). For the purposes of our study the sign is important. As an extension, in place of m_1 and m_2, we can take any two points, p_1 from class 1 and p_2 from class 2, or even the medians. With respect to the two arbitrary points p_1 and p_2, we can measure how well the signed interval $p_1 - p_2$ orders other points x_i and x_j of classes 1 and 2, respectively. A reference ordering on the line places the class 2 points x_j to the left, towards $-\infty$ and the class 1 points x_i to the right, towards $+\infty$. Let the pair of points (p_1, p_2) be the reference pair. For an arbitrary single pair (x_i, x_j), we compute, with respect to the reference pair (p_1, p_2), the appropriate single term in the **signed** transvariation intensity maximum of (8.11):

$$t_{ij}(p_1, p_2) = (x_i - p_1) - (x_j - p_2) = (x_i - x_j) - (p_1 - p_2) \ .$$

If the class 2 points x_j are located to the left of p_2 and the class 1 points x_i occur to the right of p_1, then the pair (x_i,x_j) is ordered by the pair (p_1,p_2) and $t_{ij}(p_1,p_2)$ is positive. If the points (x_i,x_j) are on the wrong sides of p_1 and p_2, then the $t_{ij}(p_1,p_2)$ is negative. If the ***margin*** between the classes is defined by p_1 and p_2, then the value $t_{ij}(p_1,p_2)$ is the extent by which the margin orders the pair of points (x_i,x_j). The ***margin*** is a segment on the line at a specific location, determined by the p_1 on the left and p_2 on the right. In the context of classification, the ordering may either be ***complete*** (the classes separate) or ***partial*** (there is class overlap). A complete ordering gives maximal classification accuracy. However, for the same classification accuracy, several partial orderings may exist. In margin-based classification $p_2 < p_1$. The value $t_{ij}(p_1,p_2)$ shows the amount by which the margin separates the class points.

8.4.2. Size of margin errors

In the following, $\boldsymbol{x}_s = [x_s^1,\ldots,x_s^D]$ are D-dimensional samples, the vector $\boldsymbol{y} = [y_1,\ldots,y_N]$ comprises the class labels, assuming values $+1$ for the positive class and -1 for the negative class. The indices i and j refer to samples from the positive and negative classes, respectively. $N = N_1 + N_2$ is the total number of samples in the two classes. We consider the balanced case, $N_1 = N_2$. $\boldsymbol{w}$ is a linear discriminant, onto which all multivariate points are projected. The dual variables of SVM and Liknon are denoted by α_s, assuming real, non-negative values. Let us consider the linear two-class classification rule (8.1):

$$y_s = \mathrm{sign}(\boldsymbol{x}_s\boldsymbol{w}^T + w_0) \ .$$

In margin-based classifiers, such as the Linear Support Vector Machine (SVM), the projected class points have to satisfy the margin requirements. Margin in SVM is specified by two positions: -1 for class 2 and $+1$ for class 1. Some points, when projected onto the discriminant, may fail to satisfy the margin requirement by an amount ξ, the slack variable:

$$\boldsymbol{x}_i\boldsymbol{w}^T + w_0 = 1 - \xi_i, \quad \boldsymbol{x}_j\boldsymbol{w}^T + w_0 = -1 + \xi_j \ .$$

The difference between the projections and the margin is:

$$(\mathbf{x}_i - \mathbf{x}_j)\mathbf{w}^T - (1-(-1)) = -(\xi_i + \xi_j) \ .$$

The value of $-(\xi_i+\xi_j)$ relates to $t_{ij}(p_1,p_2)$, discussed in Section 8.4.1. It indicates quantitatively how the margin $[-1 \ , \ +1]$ separates the projected points. If both ξ_i and ξ_j are positive, meaning that the projected points fall on the wrong side of the margin, then the value $-(\xi_i+\xi_j)$ is negative.

In SVM, the margin errors are defined (Scholkopf et al., 2000) as **points** with positive slack variables $\xi_i > 0$, $\xi_j > 0$. The value of the slack variable can be interpreted as the size of the margin error. In SVM, negative values of the slack variables are not considered. If the projected pair of points falls outside the margin (i.e., classified correctly), then the margin error is zero. The negative-valued slack variables would provide information on how well the margin separated the projected class points, and the value of $-(\xi_i+\xi_j)$ would be positive. Suppose we allowed a "negative size" and counted all slacks $\xi < 0$ and $\xi \geq 0$ negative and non-negative. In this case, the size of the total margin error indicated the extent by which the margin, specified by the pair $p_1 = 1$ and $p_2 = -1$ in SVM, separated the classes, when projected onto the discriminant $\boldsymbol{w}$. We can compute the size of the total margin error by summing up all projected points $\boldsymbol{x}_i$ of class 1 and $\boldsymbol{x}_j$ of class 2 and taking the difference:

$$(\sum_{i=1}^{N_1}\mathbf{x}_i - \sum_{j=1}^{N_2}\mathbf{x}_j)\mathbf{w}^T - \frac{N}{2}(1-(-1)) = -(\sum_{i=1}^{N_1}\xi_i + \sum_{j=1}^{N_2}\xi_j) \ . \tag{8.12}$$

The value $\xi_{ntot} = -(\sum_{i=1}^{N_1} \xi_i + \sum_{j=1}^{N_2} \xi_j)$ in (8.12) is quantitatively related to the class separation by the margin in the following way. If there are many points on the wrong side of the margin (bad class separation), then the size of the total margin error tends to be positive since $\xi_s > 0$ for those points, and subsequently ξ_{ntot} is negative. When the margin separates the majority of the points, the size of the total margin error tends to be negative, but ξ_{ntot} positive. For classes perfectly separated by the margin, ξ_{ntot} is strictly positive.

We explained the concept of the total margin error size, using the SVM margin $[-1\ ,\ +1]$. However, any segment can play the role of margin. For some fixed $\boldsymbol{w}$, the varying margin positions p_1 and p_2 produce varying ξ_{ntot}. Equation (8.12) represents a linear equation in two variables: the margin, given by $(p_1 - p_2)$ and the size of the total margin error, given by $-\xi_{ntot}$. There is a direct link of (8.12) to the transvariation intensity function, introduced in Section 8.4.3. The transvariation intensity function represents the quantity $-\xi_{ntot}$.

8.4.3. Transvariation intensity function

We introduce the transvariation intensity function as the signed transvariation intensity maximum (the denominator in (8.11)) of the classes with respect to the arbitrary pair of D-dimensional points, specifying two locations $\boldsymbol{p}_{i_1}$ and $\boldsymbol{p}_{j_2}$ in the D-dimensional space. For balanced classes ($N_1 = N_2 = \frac{N}{2}$), the D-dimensional vectorial expression of the transvariation intensity function in the original data space is:

$$\boldsymbol{T}_o = \frac{2}{N}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}((\boldsymbol{x}_i - \boldsymbol{p}_{i_1}) - (\boldsymbol{x}_j - \boldsymbol{p}_{j_2})) = (\sum_{i=1}^{N_1}\boldsymbol{x}_i - \sum_{j=1}^{N_2}\boldsymbol{x}_j) - \frac{N}{2}(\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2}) \ . \tag{8.13}$$

The left side of the equality represents the differences between the data points and the selected locations $\boldsymbol{p}_{i_1}$ and $\boldsymbol{p}_{j_2}$. In the linear Support Vector Machine, the optimal discriminant $\boldsymbol{w}$ is determined by a linear combination of the scaled data points:

$$\boldsymbol{w} = \sum_{i=1}^{N_1}\alpha_i\boldsymbol{x}_i - \sum_{j=1}^{N_2}\alpha_j\boldsymbol{x}_j \ . \tag{8.14}$$

By scaling the differences of the left side of (8.13), the transvariation intensity function $\boldsymbol{T}_\alpha$ is expressed in the scaled data space as:

$$\boldsymbol{T}_\alpha = \frac{2}{N}\sum_{i=1}^{N_1}\sum_{j=1}^{N_2}(\alpha_i(\boldsymbol{x}_i - \boldsymbol{p}_{i_1}) - \alpha_j(\boldsymbol{x}_j - \boldsymbol{p}_{j_2})) = (\sum_{i=1}^{N_1}\alpha_i\boldsymbol{x}_i - \sum_{j=1}^{N_2}\alpha_j\boldsymbol{x}_j) - \frac{\sum_{s=1}^{N}\alpha_s}{2}(\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2}). \tag{8.15}$$

In the formulation of SVM (Shawe-Taylor and Christianini, 2004), the constraint $\sum_{i=1}^{N_1}\alpha_i - \sum_{j=1}^{N_2}\alpha_j = 0$ allows for factoring out the multiplier $\frac{\sum_{s=1}^{N}\alpha_s}{2}$ from the left hand side of expression (8.15). The same constraint arises in the Liknon dual formulation (8.8). The vectors $\boldsymbol{T}_o$ and $\boldsymbol{T}_\alpha$ become scalars $t = \boldsymbol{T}_o\boldsymbol{w}^T$ and $t_\alpha = \boldsymbol{T}_\alpha\boldsymbol{w}^T$ when the classes are projected onto the discriminant vector $\boldsymbol{w}$. Let us project the right-hand side of (8.13) and (8.15) onto the discriminant $\boldsymbol{w}$ given by (8.14):

$$T_o\boldsymbol{w}^T = (\sum_{i=1}^{N_1}\boldsymbol{x}_i - \sum_{j=1}^{N_2}\boldsymbol{x}_j)\boldsymbol{w}^T - \frac{N}{2}((\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2})\boldsymbol{w}^T) \ , \tag{8.16}$$

and in the scaled space:

$$T_\alpha\boldsymbol{w}^T = (\sum_{i=1}^{N_1}\alpha_i\boldsymbol{x}_i - \sum_{j=1}^{N_2}\alpha_j\boldsymbol{x}_j)\boldsymbol{w}^T - \frac{\sum_{s=1}^{N}\alpha_s}{2}((\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2})\boldsymbol{w}^T) \ . \tag{8.17}$$

The dot product $d = (\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2})\boldsymbol{w}^T$ in (8.16) and (8.17) specifies the margin. For fixed discriminant $\boldsymbol{w}$, the differences between the projected class points $d_w = (\sum_{i=1}^{N_1} \boldsymbol{x}_i - \sum_{j=1}^{N_2} \boldsymbol{x}_j)\boldsymbol{w}^T$ and $d_{wa} = (\sum_{i=1}^{N_1} \alpha_i \boldsymbol{x}_i - \sum_{j=1}^{N_2} \alpha_j \boldsymbol{x}_j)\boldsymbol{w}^T$ are constant. The transvariation intensity functions (8.16) and (8.17) become linear equations in two variables: $t(d) = d_w + (\frac{N}{2})d$ and $t_a(d) = d_{wa} + (\frac{\sum_{s=1}^{N} \alpha_s}{2})d$, if the pair of data points $\boldsymbol{p}_{i_1}$ and $\boldsymbol{p}_{j_2}$ varies. In the projection, the transvariation intensity function indicates how the size of the total margin error (ξ_{ntot} described in Section 8.4.2) varies with the margin, specified by the different pairs of the projected points of the two classes.

8.4.4. Ordering the classes on the discriminant

The projected class points on the discriminant are ordered: class 2 tends to occupy the line segment towards $-\infty$ and class 1 towards $+\infty$. $t(d)$ allows the comparison of the discriminants for fixed d. Larger positive values of $t(d)$ correspond to margins that separate the two classes better. The transvariation intensity function $t(d)$ equals to zero at $d_{t(0)}$, the difference between the class centroids on the discriminant $\boldsymbol{w}$:

$$d_{t(0)} = \frac{2}{N}(\sum_{i=1}^{N_1} \boldsymbol{x}_i - \sum_{j=1}^{N_2} \boldsymbol{x}_j)\boldsymbol{w}^T \quad . \tag{8.18}$$

$t_\alpha(d)$ equals to zero at $d_{t_\alpha(0)}$, which is the difference between the centers of mass of the class points on the wrong sides of the margin on $\boldsymbol{w}$ (in SVM the margin is $[-1, \; +1]$):

$$d_{t_\alpha(0)} = \frac{2}{\sum_{s=1}^{N} \alpha_s}(\; \boldsymbol{w}\boldsymbol{w}^T \;) \quad . \tag{8.19}$$

The expression $\boldsymbol{w}$ in (8.14), when used in (8.19), gives:

$$d_{t_\alpha(0)} = \frac{2}{\sum_{s=1}^{N} \alpha_s}(\sum_{i=1}^{N_1} \alpha_i(\boldsymbol{x}_i\boldsymbol{w}^T) - \sum_{j=1}^{N_2} \alpha_j(\boldsymbol{x}_j\boldsymbol{w}^T)) \quad . \tag{8.20}$$

The points outside the margin have zero α coefficients. The remaining points are the ones that determine the class overlap on $\boldsymbol{w}$, by the amount $d_{t_\alpha(0)}$ (8.20). In the following, we will use $\boldsymbol{a}$ when referring to the difference between classes:

$$\boldsymbol{a} = (\sum_{i=1}^{N_1} \boldsymbol{x}_i - \sum_{j=1}^{N_2} \boldsymbol{x}_j) \quad . \tag{8.21}$$

The left panel of Figure 8.4 illustrates the linear SVM boundary that separates class 1 (squares) from class 2 (crosses), and the margins. The top panel on the right shows the partial class ordering on the discriminant $\boldsymbol{w}$. The right-bottom panel shows the transvariation intensity functions t and t_α of the class ordering shown on the right-top panel. The transvariation intensity functions illustrate the class separation with respect to the distances between the projected class points. The distances/margins that separate classes better, are characterized by positive values of the transvariation intensity functions. In the top-right panel, the centers of classes and the centers of the overlapping points are marked by stars. The segments - $d_{t(0)}$ and $d_{t_\alpha(0)}$ - are indicated by horizontal lines. The arrows from these lines lead to the bottom panel, where the transvariation intensity functions are depicted. The arrows show the positions of the critical segments $d_{t(0)}$ and $d_{t_\alpha(0)}$ in the bottom-right plot. In this example, the classes overlap. The points best separating the classes are swapped, hence the "margin" is negative and corresponds to the largest value of t and t_α.

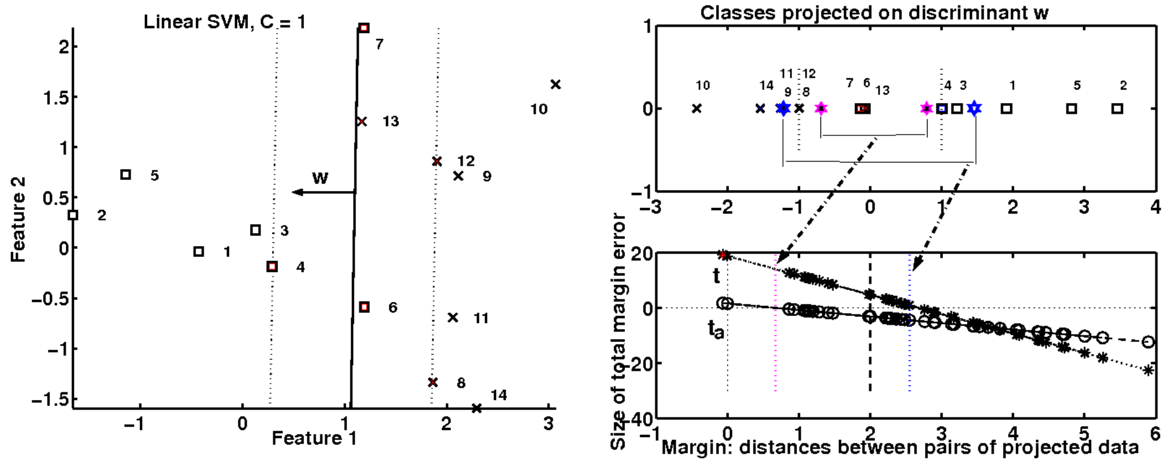

Figure 8.4: Illustration of the concepts related to the transvariation intensity functions. The left panel shows the class separation boundary (solid line) and the margins (dotted lines). The right-top panel shows the ordering of the projected class points (numbered) on the SVM discriminant. The right-bottom panel shows the transvariation functions t and t_α computed for the class ordering of the right-top panel. The asterisks (*) on t and circles (o) on t_α indicate the actual distances $(\boldsymbol{p}_{i_1} - \boldsymbol{p}_{j_2})\boldsymbol{w}^T$ between the points on the right-top panel.

8.4.5. Regularization parameter C

The class overlap (8.20) is always less than the difference between the class centroids (8.18) on the discriminant $\boldsymbol{w}$:

$$\frac{2}{\sum_{s=1}^{N} \alpha_s}(\boldsymbol{w}\boldsymbol{w}^T) \leq \frac{2}{N}(\boldsymbol{a}\boldsymbol{w}^T) \ . \tag{8.22}$$

The coefficients $\alpha_s \leq C, \ \ s = 1, \ldots, N$ are constrained by C in the original SVM formulation. Rearranging (8.22) and noting that for $\alpha_s \leq C$, the mean value $\frac{\sum_{s=1}^{N}\alpha_s}{N}$ is also less than C, we have:

$$\frac{(\boldsymbol{w}\boldsymbol{w}^T)}{(\boldsymbol{a}\boldsymbol{w}^T)} < \frac{\sum_{s=1}^{N}\alpha_s}{N} \leq C \quad \Rightarrow \quad ratio = \frac{\|\boldsymbol{w}\|}{\|\boldsymbol{a}\|\cos(\boldsymbol{w}\hat{\ }\boldsymbol{a})} \leq C \ . \tag{8.23}$$

The *ratio* in (8.23) explains how the regularization parameter C relates to the class separation. C bounds the ratio of $\|\boldsymbol{w}\|$ over $\|\boldsymbol{a}\|\cos(\boldsymbol{w}\hat{\ }\boldsymbol{a})$. $\|\boldsymbol{w}\|$ is the length of the normal to the separation boundary, related to the margin between the classes in the original space as $\frac{1}{\|\boldsymbol{w}\|}$. The expression $a_w = \|\boldsymbol{a}\|\cos(\boldsymbol{w}\hat{\ }\boldsymbol{a})$ means a projection of the class difference (8.21) onto the normal $\boldsymbol{w}$, showing the alignment of the two vectors $\boldsymbol{w}$ and $\boldsymbol{a}$. An increase in class separation is associated with the increase of $\|\boldsymbol{w}\|$ and the decrease of a_w. Thus, the increasing *ratio* indicates better separation of the classes, projected onto the solved discriminant $\boldsymbol{w}$.

Both Svmpath and Liknon produce a sequence of solutions. The sequence of the solutions improving class separation proceeds through the stages of under-fitting, optimal and over-fitting with respect to the classification performance on the independent validation set. These stages can be visualized using the monitoring BER of every solution. For visualization we use a logarithmic transformation of the ratio $\frac{\|\boldsymbol{w}\|}{a_w}$, $lr = \lg_{10}(\frac{\|\boldsymbol{w}\|}{a_w})$. Figure 8.5 illustrates the performance (monitoring BER) of the sequence of the solutions of the full regularization path and Liknon

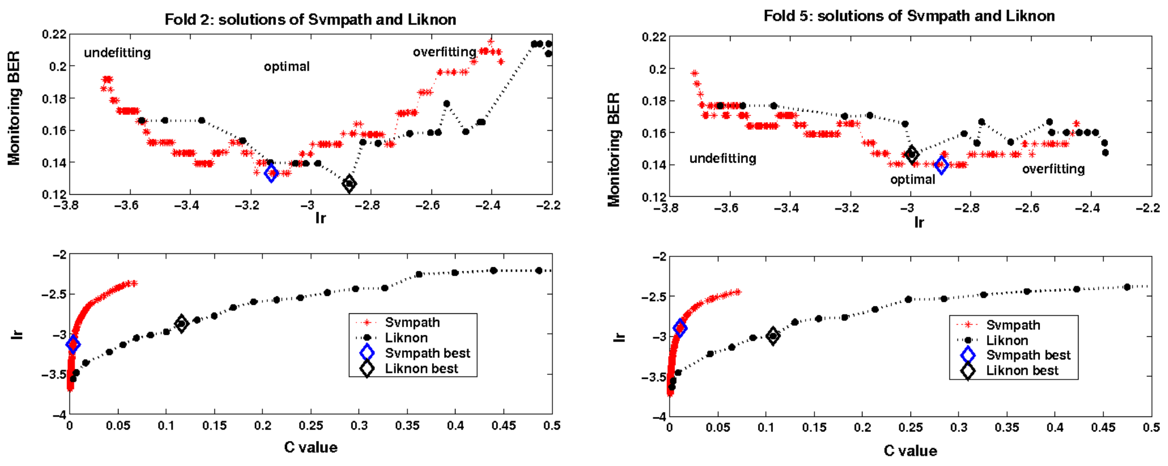

Figure 8.5: Relationship between the performance, the lr and regularization parameter C for the sequence of the solutions of the full regularization path and Liknon. We illustrate the solutions in the folds 2 and 5 of the Banana example. The best solutions are indicated by diamonds. The top panel shows the monitoring BER of every solution versus $lr = \lg_{10}(\frac{\|\boldsymbol{w}\|}{a_w})$. The bottom panel shows the plot of lr versus C.

versus the lr in folds 2 and 5 of the artificial Banana example of Section 8.2. Plots of lr versus C for every solution are shown in the left-bottom (fold 2) and the right-bottom (fold 5) panels. In the Svmpath method, C corresponds to the parameter $1/\lambda$ (Hastie et al., 2004). The solutions, producing minimum monitoring BER are indicated by diamonds. The values of the lr of the optimal solutions of both Liknon and Svmpath are similar and occur in the range of $[-3.2, -2.8]$. The increasing lr starts saturating at about -2.8, where the monitoring BER starts growing. The C values, associated with the optimal solutions of Svmpath and Liknon are not close, but the character of the dependance between the lr and C is similar for both methods. It is possible to obtain several Liknon solutions and use them for approximating the range of the optimal lr for Svmpath. The details are beyond the scope of this chapter. The explained concepts and relation (8.23) provide the basis for the derivation of the constructive algorithm in Liknon for C computation.

8.5. Liknon feature selection

Liknon simultaneously identifies a subset of useful features and a linear discriminant. Discriminants of increasing complexity in terms of large/non-zero weights are associated with small margins between classes. The margin can be decreased by increasing the value of the regularization parameter C. Here we outline the algorithm for the computation of C for Liknon.

8.5.1. The standard discriminant, given by the solution of the Liknon dual

The optimal solution $(^*)$ $\alpha^* = [\alpha_1^*, \ldots, \alpha_N^*]$ of the dual (8.8) satisfies all constraints and the optimality conditions (8.9) and (8.10). Geometrically, it also determines the direction that discriminates the classes, obtained from a linear combination of the data points. According to the optimality conditions, all coordinates of the discriminant (henceforth called the *standard discrim-*

inant) have unit length. For a set of m selected individual features $\boldsymbol{f} = (f_1, \ldots, f_m), \quad m \leq D$, corresponding to the binding constraints, the set of equations

$$e^{f_k} = \sum_{i=1}^{N_1} \alpha_i^* x_i^{f_k} - \sum_{j=1}^{N_2} \alpha_j^* x_j^{f_k} = \pm 1 \quad , \tag{8.24}$$

is satisfied. Therefore, the solution of the Liknon dual identifies the standard discriminant in the m-dimensional subspace of the selected features $\boldsymbol{f}$:

$$\boldsymbol{e} = \sum_{i=1}^{N_1} \alpha_i^* \boldsymbol{x}_i - \sum_{j=1}^{N_2} \alpha_j^* \boldsymbol{x}_j, \quad \text{and} \quad \boldsymbol{e} = [\pm 1_1 \ldots \pm 1_m], \tag{8.25}$$

which corresponds to (8.14) and can be interpreted as a "preimage" of the $\boldsymbol{w}$, normal to the optimal Liknon hyperplane. The discriminant $\boldsymbol{e}$ has constant length $\sqrt{m}$, determined by the dimensionality of the identified subspace.

8.5.2. Given vs. desired: margin control

The Liknon-selected feature subset of dimensionality m identifies some subspace. In this subspace we have $\boldsymbol{a}$ - the difference between the class vectors and the standard discriminant $\boldsymbol{e}$, given by the solution of the Liknon dual. The distance $\boldsymbol{a}\boldsymbol{e}^T$ between the classes projected onto $\boldsymbol{e}$ depends on the number of non-zero elements of the standard discriminant/number of selected features. The ratio of class overlap to the distance between classes, (8.23), in Liknon gives:

$$\frac{(\mathbf{e}\mathbf{e}^T)}{(\mathbf{a}\mathbf{e}^T)} < \frac{\sum_{s=1}^{N} \alpha_s}{N} \leq C \quad . \tag{8.26}$$

Thus, we can explain and visualize how the value of the parameter C controls the class separation by the distance, and influences the selection of the subspaces:

$$\frac{(\boldsymbol{a}\boldsymbol{e}^T)}{m} > \frac{1}{C} \quad .$$

The distance between the classes on the standard discriminant, determined by the solution of Liknon at a specific value of C, will be greater than $\frac{1}{C}$. We don't know in advance the standard discriminant, and apply (8.26) to the individual features to determine C.

8.5.3. Selection of C

Projection of the values of individual features f_k onto their respective elements of the standard discriminant $x_s^{f_k} e^{f_k}$ is merely a multiplication of $x_s^{f_k}$ by 1 or -1. The difference $d_{i,j}^{f_k} = (x_i^{f_k} - x_j^{f_k}) e^{f_k}$ of the feature values in the two classes specifies the margin. a^{f_k} is an element of (8.21). The transvariation intensity functions t and t_α can be written as:

$$t^{f_k}(d_{i,j}^{f_k}) = a^{f_k} e^{f_k} - (\tfrac{N}{2}) d_{i,j}^{f_k} \quad , \quad t_\alpha(d_{i,j}^{f_k}) = e^{f_k} e^{f_k} - (\tfrac{\sum_{s=1}^{N} \alpha_s}{2}) d_{i,j}^{f_k} \quad . \tag{8.27}$$

According to (8.18) and (8.19), and noting that $e^{f_k} e^{f_k} = 1$, the margins/segments for which t^{f_k} and t_α attain zero are:

$$d_{t^{f_k}(0)} = \tfrac{2}{N}(a^{f_k} e^{f_k}), \quad d_{t_\alpha(0)} = \tfrac{2}{\sum_{s=1}^{N} \alpha_s} \quad . \tag{8.28}$$

Relation (8.26) for an individual feature gives:

$$\frac{1}{(a^{f_k} e^{f_k})} < \frac{\sum_{s=1}^{N} \alpha_s}{N} \quad . \tag{8.29}$$

The sign of $a^{f_k} e^{f_k}$ can be ignored, because it merely indicates swapped classes. C influences the selection of the individual features by bounding the class difference:

$$|a^{f_k}| = |\sum_{i=1}^{N_1} x_i^{f_k} - \sum_{j=1}^{N_2} x_j^{f_k}| \; > \; \frac{1}{C} \quad , \tag{8.30}$$

The individual features, with a difference between classes greater than $\frac{1}{C}$, are candidates to be included in the Liknon discriminant. By arranging class differences of the individual features in descending order, we obtain the relationship $|a^{f_k}_{\max}| \geq \ldots \geq |a^{f_k}_{\min}|$. Our first approach to selecting C was setting the parameter equal to the inverse of every member of the sequence, $\frac{1}{|a^{f_k}_{\max}|} \leq \ldots \leq \frac{1}{|a^{f_k}_{\min}|}$, and solving a sequence of Liknon models (primal) with the sequence of C values so determined. However, if the number of features is large, as in microarray data, the computations become prohibitive. An algorithm for selecting fewer C values is necessary. Our second approach was to compute a histogram and select the C using the modes of the histogram. (A limitation of the histogram approach is the selection of optimal bin.)

8.5.4. Algorithm for computing C

The algorithm for computing a subset of C values is based on the transvariation intensity functions F_{trans} of the individual features. The F_{trans} are modeled by the linear equations $t^{f_k} = |a^{f_k}| - (\frac{N}{2})d$, with slopes $\frac{N}{2}$ and intercepts $|a^{f_k}|$. The variable d accounts for the distances $(x_i^{f_k} - x_j^{f_k})e^{f_k}$. The standard F_{trans}, $t_\alpha = 1 - (\frac{\sum_{l=1}^{N} \alpha_l}{2})d$ has slope $\frac{\sum_{l=1}^{N} \alpha_l}{2}$ and intercept 1. In general, the F_{trans} of the features that separate classes better are larger and they attain zero at larger d. The F_{trans} for which distances/margins separate classes better have positive values. Given the features, the increase in class separation as a function of all features for all distances d can be determined by summing the positive parts of the t^{f_k} . We call this measure t_{total}. It increases piecewise linearly as the distance d decreases. The set of NM (number of models) values of t_{total} is used for the determination of the set of distances d_s, $s = 1, \ldots, NM$, assuming the class differences $|a^s| = (\frac{N}{2})d_s$. The C_s value associated with the d_s is computed using (8.30), as $C_s = \frac{2}{Nd_s}$. Setting the C value to C_s means that we constrained the absolute class difference of the selected features to be greater than $\frac{1}{C_s}$. The smaller the value of d_s, the larger C_s, the more features are included in the Liknon discriminant.

Figure 8.6 illustrates the idea of the outlined algorithm for an artificial dataset with $D = 10$ features and $N = 100$ samples. Two features are discriminatory, the rest are fully overlapping $N(0,1)$-distributed features. The F_{trans} values of the discriminatory features are larger and attain zero at a larger d.

There are segments where t_{total} changes slowly. The more functions become positive and add to the total, the more t_{total} changes. The potentially interesting segments occur where the F_{trans} concentrate. A Matlab code for computing the range of C values is listed in the Appendix.

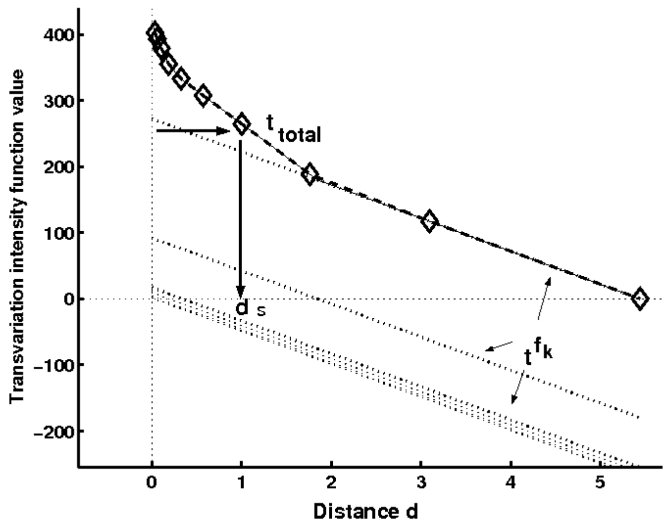

Figure 8.6: Computation of C using t_{total}. The transvariation intensity functions t^{f_k} are shown by the dotted lines. The dashed line depicts t_{total}. The diamonds indicate the positions and values of the t_{total} for which the distance d_s is determined.

8.6. Liknon feature and classification model selection on the benchmark datasets

Here we present a realistic single-run assessment of Liknon-based feature selection with respect to other feature selection methods in the NIPS 2003 FS challenge. We also present our results for the ALvsPK competition. In the ALvsPK challenge, the Liknon-based classification model selection strategy was among the top-ranked methods.

8.6.1. Liknon feature selection applied to the NIPS 2003 FS benchmark datasets

Information on the NIPS 2003 FS benchmark is on the website `www.nipsfsc.ecs.soton.ac.uk` and in publications (Guyon et al., 2006, 2007b). In the benchmark, the ground truth for features - *useful* or *probe* - is available. Probe is a "fake" feature purposefully inserted into the dataset. We tested Liknon's utility as a filter and a wrapper in a single run, including univariate feature prefiltering. The class difference (8.21) provides information about the class separation. In prefiltering we rank the individual features by the decreasing absolute value of $|a^{f_k}|$ computed for every feature and discarding a percentage of low-$|a^{f_k}|$ value features. Liknon is solved for the remainder. The ranking of the features by (8.21) to the ranking by difference in means can be compared by the fraction of the disagreeing ranks. The percentage of disagreeing ranks of the features ordered by (8.21) and by the difference between the means of the NIPS 2003 FS training data is: ARCENE - 66.90%; DEXTER - 24.61%; DOROTHEA - 59.48%; GISETTE - 3.28%; MADELON - 5.20%. The percentage of the discarded features and C were determined in several repetitions of 5-fold crossvalidation, with a single split in the inner loop. The determined parameters were used in a `final single` Liknon run on the Training set. For DOROTHEA and GISETTE, a smaller, random, balanced training sample was taken. For the Liknon wrapper, the same discriminant was used as the final classifier. For the filter, the Liknon identified feature subset was input to 3-nearest-neighbor (3nn) and subspace (subsp) classifiers. The final model - 3nn or subsp - was the one with the smaller Validation BER. Table 8.2 summarizes the experimental situation.

Table 8.2: Experimental setup for the NIPS 2003 FS benchmark.

Dataset Data origin	ARCENE mass spectra	GISETTE handwritten digits	DEXTER text	DOROTHEA drug discovery	MADELON difficult artificial
Dimensionality	10000	5000	20000	100000	500
Total probes(%)	30	50	50.3	50	96
Train	44 + 56	3000 + 3000	150 + 150	78 + 722	1000 + 1000
Validation	44 + 56	500 + 500	150 + 150	34 + 316	300 + 300
Test	310 + 390	3250 + 3250	1000 + 1000	78 + 722	600 + 600
Discarded(%)	86	95	75	99	98
C	0.0058	0.00066	0.0035	0.2369	0.016
Classifier	3nn	3nn	subsp	subsp	3nn

Table 8.3 presents the performance of the Liknon wrapper and filter with respect to the methods in the NIPS 2003 FS benchmark that performed feature selection. Mean and median test BERs of the latter allow ranking the Liknon-based method with respect to other methods. We also present the best entries, published recently (Guyon et al., 2007b). Both Liknon filter and wrapper compare favorably with the average performance of the benchmark methods, but are worse than the recent best entries. Our results indicate better performance of Liknon as a filter than as a wrapper. Liknon features, used with 3nn and subsp classifiers performed better on all datasets, except for DEXTER.

Table 8.3: Performance comparison. Test BERs

Dataset	Mean	Median	Liknon filter	Liknon wrapper	Best recent entry
ARCENE	0.2396	0.2087	0.1711	0.1962	0.1048
DEXTER	0.1170	0.0690	0.0820	0.0820	0.0325
DOROTHEA	0.2054	0.1661	0.1961	0.2011	0.0930
GISETTE	0.0823	0.0246	0.0402	0.0742	0.0111
MADELON	0.1922	0.1245	0.1133	0.3789	0.0622

Liknon was robust in detecting the ground truth. Our fraction of features and probes and those of the best recent entries are listed in Table 8.4. The fraction of features F_{feat} is the ratio of the number of the selected features to the total number of features. The fraction of probes F_{probe} is the ratio of the probes in the selected subset to the total number of selected features. The Liknon feature subsets contain fewer features and probes than the best recent entries. Useful features were consistently identified in crossvalidation and in the feature subset obtained in the final training.

The presented computational experiment highlights the potential of a simple single Liknon application to the data. First, by univariate feature filtering, using class difference (8.21) as the criterion, most of the probes were discarded. Second, an even smaller feature subset, optimal for linear separation, was obtained by solving Liknon. It is known, that some NIPS 2003 FS

Table 8.4: Fraction of features F_{feat} and probes F_{probe}

Dataset	Our F_{feat}	Our F_{probe}	Best recent F_{feat}	Best recent F_{probe}
ARCENE	0.0041	0.00	0.14	0.04
DEXTER	0.0079	0.095	0.23	0.55
DOROTHEA	$7 * (10^{-5})$	0.00	0.01	0.03
GISETTE	0.0112	0.0179	0.20	0.00
MADELON	0.02	0.00	0.04	0.00

benchmark datasets have nonlinear character. Linear classifiers such as Liknon are not optimal in such scenario. The poorer performance of the Liknon wrapper compared to the filter corroborates this fact. If the number of features is larger than the number of samples, then the number of non-zero weights in the Liknon discriminant is bounded by the number of samples, guaranteed by the nature of linear programming formulation and binding constraints. The small number of features, identified in a single Liknon run, might not be sufficient to fully capture most of the information about class separation. The feature profile, accumulated in many Liknon runs on many data splits, may overcome this limitation. Such strategy was used in the ALvsPK competition and it ranked Liknon among the top-ranked methods.

8.6.2. Liknon versus Svmpath on the NIPS2003 feature selection benchmark datasets

In this subsection the Liknon filter is contrasted with the related method of S. Rosset and J. Zhu (Rosset and Zhu, 2006) on NIPS2003 FS benchmark datasets. The method included a univariate filtering by the t-statistic and computation of the principal components (PCA) as features after prefiltering for the ARCENE and DOROTHEA datasets; Then, a regularized optimization scheme using various combinations of a loss function L and a penalty J was applied as follows.

- ARCENE: L was the Huberized hinge loss, and J was the L_2-norm penalty (linear support vector machine).
- DEXTER: L was the Huberized hinge loss, and J was the L_1-norm penalty (L_1 norm support vector machine).
- DOROTHEA: L was the Huberized hinge loss; J was the L_1-norm penalty (L_1 norm support vector machine).
- GISETTE: L was the exponential loss; J was the L_1-norm penalty.
- MADELON: L was the hinge loss, and J was the L_2-norm penalty (radial basis kernel support vector machine).

All hyper-parameters (including the number of features) were selected using 5-fold cross-validation. Liknon filter was applied to all datasets, using the same unified scheme explained in the previous subsection. The test BERs, area under the curve, AUC, number of features and probes of both approaches are summarized and contrasted in Table 8.5. These results are also available on the website `www.nipsfsc.ecs.soton.ac.uk`.

The method of S.Rosset and J.Zhu, applied to DEXTER and DOROTHEA, is comparable to the Liknon filter, since it also implements a linear SVM with L_1-norm penalty. If we compare the performances by BER and AUC, then the results of S.Rosset and J.Zhu are better on these datasets. If we compare the size and purity of the feature subsets in DEXTER and DOROTHEA,

Table 8.5: Results of S.Rosset and J.Zhu compared to Liknon-based feature/classification model selection in the NIPS2003 feature selection benchmark.

Method	Saharon Rosset and Ji Zhu			Liknon filter		
Dataset	BER	AUC	Features(Probes)	BER	AUC	Features(Probes)
ARCENE	0.1962	0.8038	3000(171)	0.1711	0.8908	41(0)
DEXTER	0.0690	0.9628	112(50)	0.0820	0.9511	158(15)
DOROTHEA	0.1569	0.8451	5207(4009)	0.1996	0.8053	7(0)
GISETTE	0.0134	0.9826	1500(0)	0.0402	0.9791	56(1)
MADELON	0.0906	0.9094	21(2)	0.1133	0.9369	10(0)

then the Liknon filter is better. In the NIPS2003 FS challenge, several methods performed better on the feature sets containing many probes. For datasets with limited number of samples, e.g., DEXTER, many fake features transform the class separation character and we actually have a different classification problem. On this feature-augmented problem some methods do better than on the original problem without probes. In general, a fair comparison is only possible if we had the reference BERs of several methods - linear and nonlinear - using a) only relevant features and b) all features. Such experiment is beyond the scope of this chapter.

8.6.3. Numerical experiments on the datasets of Agnostic Learning vs. Prior Knowledge competition

The Liknon-based classification model participated in the agnostic track of the ALvsPK challenge as a "black box", using five datasets ADA, GINA, HIVA, NOVA and SYLVA. 10-fold crossvalidation with 31 splits in the inner loop and without feature prefiltering was used to obtain an ensemble (ens) of 31 Liknon discriminants, and the profile of the identified relevant features. From the profile, a subset of features occurring more frequently than some threshold, (the thresholds were $10\%, 15\%, 20\%, \ldots, 95\%, 100\%$), was selected and used in training the following classifiers from PRTools (Duin et al., 2004): fisher linear discriminant, logistic linear classifier, quadratic classifier, subspace classifier, and 1- and 3-nearest-neighbor classifiers. On four of the five datasets, the ensembles of Liknon discriminants performed better than other rules. The dimensionality of the datasets, the sizes of the subdivisions of training data (explained in Section 8.2), the optimal threshold for the feature profile, the number of the tested Liknon models `NM`, and the winning classifier are summarized in Table 8.6.

Our results in the ALvsPK competition are presented in Table 8.7. We report our balanced test error rate (BER), the area under the receiver operating curve (AUC), and the same information for the winning entries. The ensemble of Liknon discriminants was superior for ADA, HIVA, NOVA and SYLVA. For GINA, the 3-nearest-neighbor rule, trained on the most prominent features, performed best. In Section 8.2, we analyzed a dataset, for which two features separate the classes nonlinearly. Similarly, in GINA, which is too complex for a linear classifier, Liknon identified features that represented the nature of class separation sufficiently well for a nonlinear rule. For SYLVA, among the individual rules, the quadratic discriminant performed comparably to the ensemble. The histogram-based approach of C selection worked best for HIVA. For the remaining datasets, the C selection algorithm presented in Section 8.5.4 gave the top-ranked results. All details of our results in the ALvsPK challenge have already been published elsewhere (Pranckeviciene et al., 2007).

Table 8.6: Setup and parameters of the ALvsPK challenge datasets

Dataset Data origin	ADA marketing	GINA handwritten digits	HIVA drug discovery	NOVA text	SYLVA ecology
Dimensionality	48	970	1617	16969	216
Threshold	55%	50%	20%	80%	20%
Training	600+600	350+350	100+100	400+400	400+400
Monitoring	2487+419	1237+1184	3572+34	842+94	11758+397
Validation	343+113	176+171	408+15	138+55	1351+89
NM	5	8	20	25	10
Winning classifier	ens	3nn	ens	ens	ens

In Table 8.7, for GINA, NOVA and SYLVA, our AUC is slightly better than those of the best entries. This arises because AUC accounts for the full range of operating points, whereas BER is based on a single confusion matrix and represents a single operating point of the classifier on the ROC curve. A larger value of AUC suggests, that the classifier may have improved classification performance when the prevalence (proportions) of the test samples in the two classes is different from the proportion given in the challenge datasets.

Table 8.7: Results on the ALvsPK challenge datasets: our method and the best entry in the agnostic track

Dataset	Our Test BER	Our Test AUC	Best Test BER	Best Test AUC
ADA	0.1818	0.8702	0.1660	0.9168
GINA	0.0533	0.9740	0.0339	0.9668
HIVA	0.2939	0.7589	0.2827	0.7707
NOVA	0.0725	0.9814	0.0456	0.9552
SYLVA	0.0190	0.9949	0.0062	0.9938

8.7. Discussion and Conclusions

Inspired by the work of Montanari (Montanari, 2004), we introduced a novel and key concept, the transvariation intensity function. It characterizes the univariate separation of two classes as a linear function of the distances between points from the classes. For the original and scaled data, projected onto some discriminant, it specifies the two distances important for class separation: the size of the class difference (8.18) and the size of class overlap (8.20). Their ratio elucidates the role of the regularization parameter as the controller of the class overlap in SVM and Liknon. The relationship between the regularization parameter C, the class difference and the class overlap provided a better understanding of how Liknon works, important in practical applications of the method. In Liknon, the class difference on the standard discriminant and individual selected features is constrained from below by $1/C$. By increasing the parameter C, we control the class overlap in individual features, allowing more features to be included into the optimal Liknon discriminant. Based on the total transvariation intensity function of the individual features, we proposed an algorithm for computing C values.

We demonstrated the efficiency of Liknon, combined with univariate feature filtering, in identifying smaller feature subsets of the ground truth features. Because of the properties of the linear programming method of Liknon, the number of non-zero components (selected features) in the solution of the primal $\boldsymbol{w}$ is bounded by the number of samples in the dataset. Liknon's limited, single-run performance, when compared to the best recent entries for the NIPS 2003 FS datasets, indicates that either the number of selected features in a single split is not sufficient for fully representing the nature of the class separation, or that the linear classifier cannot handle the complexity of the classification problem. As possible strategies, we suggest using an ensemble of Liknon discriminants trained on feature subsets of different sizes, or employing Liknon features as inputs to nonlinear classification rules. In the ALvsPK challenge, results based on these strategies were among those of the top-ranked methods. The main advantage of Liknon is its ability to identify stable features in high-dimensional, small sample size datasets. A possible disadvantage is the considerable computational load of processing many data splits in the inner crossvalidation loop.

Related to our work is the full regularization path algorithm (Hastie et al., 2004; Rosset and Zhu, 2006). It uses loss functions with quadratic terms and L_1-norm penalty. It allows computation of derivatives in order to obtain the sequence of solutions $\boldsymbol{w}$. The algorithm does not calculate the regularization parameter explicitly. Liknon has different formulation in terms of the loss function. Loss in Liknon is linear and the partial derivatives with respect to $\boldsymbol{w}$ would be constant. The linearity of Liknon enables formulating it as a linear programming problem and solving for $\boldsymbol{w}$ using the LP optimization method. Using such formulation we can derive how to compute the regularization parameter, but we can't compute the weights of the discriminant directly. Solving Liknon with a sequence of Cs and using a monitoring set to select the optimal solution $\boldsymbol{w}$ is similar to the procedure for the regularization path approach, but Liknon uses a **linear loss** and an L_1 penalty. These two algorithms obtain the sequence of the solutions in different ways.

We carried out numerical experiments on the artificial Banana example to compare the performances of the two methods. We also contrasted the Liknon filter with the relevant entry of the NIPS2003 FS challenge, the regularization path method used by S.Rosset and J. Zhu; it performed better than the Liknon filter. However, from the point of view of the purity and size of the selected feature subsets, Liknon was better. The regularization path method is more flexible than Liknon in learning nonlinear structures through the use of various loss functions. Liknon is "conservative" in adequately capturing the complexity of class separation, identifying only the subsets of interacting features that are optimal for linear class separation. Liknon-embedded feature selection is suited for problems for which user is interested in finding several important original features, the class differences arise due to the differences in class means, there are many correlated features and finally, the number of features is much larger than the number of the training samples.

Acknowledgments

We thank the reviewers and editors for constructive comments that helped us improve the manuscript. The first author thanks the organizers of Agnostic Learning vs. Prior Knowledge challenge for their effort in providing an excellent platform to test learning methods. The group of Prof. R.P.W. Duin is also acknowledged for making PRTools publicly available.

Appendix.

The Matlab function for computation of a sequence of NM values of C is presented in this appendix. The code implements the algorithm presented in Section 8.5.4. Given a data matrix `X` and a vector of class labels `Y`, the vector `a` of class difference is computed. A grid of `d` values is constructed out of the vector `a`. Linear transvariation functions for every feature `j` are computed as `Ftrans=[abs(a(j))-(N/2)*d]`. `Ttotal` is obtained by summing up the positive parts of the transvariation functions `Ftrans`. The logarithm of `Ttotal` is used to obtain an equally spaced grid. Using such a grid, the sequence `ds` is obtained by linear interpolation. Using the sequence `ds`, the corresponding `C` values are computed as `C=[2./(N*ds)]`. A large `C=100` is also included in the sequence.

```
function [c]=computeC(X, Y, NM)
%
% Output: c - sequence of C values
% Input: X - data matrix,  Y - class labels,  NM - number of C values
%
N = size(X,1); Na = 100;
i1 = find(Y==1); i2 = find(Y==-1);
a = sum(X(i1,:))-sum(X(i2,:));
da = (2/N)*abs(a);
d = unique([da[0:max(da)/Na:max(da)]]);
Ttotal = zeros(1,length(d));
for j = 1:length(a)
   Ftrans = [abs(a(j))-(N/2)*d];
   m = [Ftrans>0];
   Ttotal = Ttotal+Ftrans.*m;
end;
tm = max(Ttotal);tt = abs(Ttotal-tm);zeroi = find(tt==0);
if ~isempty(zeroi); Ttotal(zeroi) = []; tt(zeroi) = []; d(zeroi) = []; e
tlog = log(tt); [val,ind] = unique(tlog);
tlog = tlog(ind); d = d(ind);
t1 = min(tlog); t2 = max(tlog);
ti = [t1+(t2-t1)/NM:(t2-t1)/NM:t2];
ds = interp1(tlog,d,ti,'linear');
c = sort([2./(N*ds) 100]);
ts = tm-exp(ti);
plot(d,Ttotal,'k',ds,ts,'kd-');
return;
```

References

C.Ambroise and G.J.McLachlan. Selection bias in gene extraction on the basis of microarray gene-expression data. *PNAS*, 99(10), pages 6562-6566, 2002.

T.S. Arthanari and Y. Dodge. *Mathematical programming in statistics*. John Willey and Sons, New York, 1981.

C.Bhattacharyya, L.R.Grate, A.Rizki, D. Radisky, F.J. Molina, M.I. Jordan, M.J. Bissell and I.S. Mian. Simultaneous relevant feature identification and classification in high-dimensional spaces: application to molecular profiling data. *Signal Processing*, 83(4), pages 729-743, 2003.

V.Cherkassky and Y.Ma, Margin-based Learning and Popper's Philosophy of Inductive Learning. In M. Basu and T.K. Ho, editors. *Data complexity in pattern recognition.*. 2006.

R.P.W.Duin, P.Juszczak, P.Paclik, E.Pekalska, D.de Ridder, and D.M.J.Tax. *PRTools4 A Matlab toolbox for pattern recognition*, February, 2004.

G.Fung and O.Mangasarian. A feature selection Newton method for support vector machine classification. *Computational Optimization and Applications*, 28, pages 185–202, 2004.

G.D.Guo and C.Dyer. Learning from examples in the small sample case: face expression recognition. *IEEE Trans. on System, Man and Cybernetics - Part B*, 35(3), pages 477–488, 2005.

I.Guyon, V.Vapnik, B.Boser, L.Bottou, and S.Solla. Capacity control in linear classifiers for pattern recognition. *Proceedings of the 11th IAPR International Conference on Pattern Recognition*, IEEE, 2, pages 385–388, 1992.

I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors. *Feature extraction, foundations and applications*. Physica-Verlag, Springer, 2006.

I.Guyon, A.Safari, G.Dror, and G.Cawley. Agnostic learning vs. prior knowledge challenge. *Proccedings of International Joint Conference on Neural Networks IJCNN2007*, INNS/IEEE, Orlando Florida, pages 829–834, 2007a.

I.Guyon, J.Li, T.Mader, P.A.Pletsher, G.Schneider, and M.Uhr. Competitive baseline methods set new standards for the NIPS 2003 feature selection benchmark. *Pattern recognition letters*, 28, pages 1438–1444, 2007b.

T.Hastie, S.Rosset, R.Tibshirani, and J.Zhu. The entire regularization path for the support vector machine. *Journal of Machine Learning Research*, 5, pages 1391-1415, 2004.

C.Igel. Multi-objective model selection for support vector machines. In C.A. Coello Coello, A. Hernandez Aguirre, and E. Zitzler, editors. *Evolutionary Multi-criterion Optimization, LNCS* 3410, pages 534–546, 2005.

R.Kohavi and G.H.John. Wrappers for feature subset selection. *Artificial intelligence*, 97(1-2), pages 273–324, 1997.

M.Kudo and J.Sklansky. Comparison of algorithms that select features for pattern classifiers. *Pattern recognition*, 33(1), pages 25–41, 2000.

A.Montanari. Linear discriminant analysis and transvariation. *Journal of Classification*, 21, pages 71–88, 2004.

C.H. Papadimitriou and K. Steiglitz. *Combinatorial optimization Algorithms and Complexity*. Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

E.Pranckeviciene, R.Somorjai, R.Baumgartner, and M.Jeon. Identification of signatures in biomedical spectra using domain knowledge. *AI in Medicine*, 35(3), pages 215–226, 2005.

E.Pranckeviciene, R.Somorjai, and M.N.Tran. Feature/model selection by the Linear Programming SVM combined with State-of-art classifiers: what can we learn about the data. *Proccedings of International Joint Conference on Neural Networks IJCNN2007*, INNS/IEEE, Orlando, Florida, pages 1627–1632, 2007.

S.Rosset and J.Zhu. Sparse, flexible and efficient modelling using L_1 regularization. In I. Guyon, S. Gunn, M. Nikravesh, and L. Zadeh, editors. *Feature extraction, foundations and applications*, pages 379–398, 2006.

J.Shawe-Taylor and N.Chistianini. *Kernel methods for Pattern Analysis*. Cambridge University Press, 2004.

B.Scholkopf, R.C.Williamson, and P.Bartlet. New Support Vector Algorithms. *Neural Computation*, 12, pages 1207–1245, 2000.

R.L.Somorjai, B.Dolenko, and M.Mandelzweig. Direct classification of high-dimensional data in low-dimensional feature spaces- comparison of several classification methodologies. *Journal of Biomedical Informatics*, 40, pages 131–138, 2007.

W.Zucchini. An introduction to model selection. *Journal of mathematical psychology*, 44, pages 41–61, 2000.