Nearest Convex Hull Classification

Georgi I. Nalbantov[1,2], Patrick J. F. Groenen[2], and Jan C. Bioch[2]

[1] ERIM, Erasmus University Rotterdam
[2] Econometric Institute, Erasmus University Rotterdam
nalbantov@few.eur.nl

Econometric Institute Report EI 2006-50

Abstract. Consider the classification task of assigning a test object to one of two or more possible groups, or classes. An intuitive way to proceed is to assign the object to that class, to which the distance is minimal. As a distance measure to a class, we propose here to use the distance to the convex hull of that class. Hence the name Nearest Convex Hull (NCH) classification for the method. Convex-hull overlap is handled through the introduction of slack variables and kernels. In spirit and computationally the method is therefore close to the popular Support Vector Machine (SVM) classifier. Advantages of the NCH classifier are its robustness to outliers, good regularization properties and relatively easy handling of multi-class problems. We compare the performance of NCH against state-of-art techniques and report promising results.

1 Introduction

There are many approaches to the classification task of separating two or more groups of objects on the basis of some shared characteristics. Existing techniques range from Linear Discriminant Analysis (LDA), Quadratic Discriminant Analysis (QDA) and Binary Logistic Regression to Decision Trees, Neural Networks, Support Vector Machines (SVM), etc. Many of those classifiers make use of some kind of a distance metric (in some n-dimensional space) to derive classification rules. Here, we propose to use another such classifier, called Nearest Convex Hull (NCH) classifier.

As the name suggests, the so-called hard-margin version of the NCH classifier assigns a test object $\mathbf{x}$ to that group of training objects, which convex hull is closest to $\mathbf{x}$. This involves solving an optimization problem to find the distance to each class. Algorithms for doing so have been proposed in the literature under the general heading of finding the minimum distance between convex sets (see, e.g., [14] and [2]). We confer also to [10] for a more general discussion on distance-based classification. Existing off-the-shelf algorithms however cannot be directly applied for classification tasks where a mixture of a soft-margin and a hard-margin approaches is required. In the separable, hard-margin case, a problem arises if $\mathbf{x}$ lies inside the convex hulls of two or more groups, since its distance to these convex hulls is effectively equal to zero and the classification of $\mathbf{x}$ is undetermined. To deal with this problem, we introduce a soft-margin version of the NCH classifier, where convex-hull overlap between $\mathbf{x}$ and a given class is penalized linearly. The difference with the soft-margin SVM approach lies in the requirement that the soft approach is applied to all data points except the test point $\mathbf{x}$. As an alternative solution to convex-hull overlaps, one could map the training data from the original space into a higher-dimensional space where convex-hull overlap can be avoided. A combination of both approaches is also possible.

The linear (and not, for example, quadratic) penalization of the errors gives rise to the robustness-to-outliers property of NCH. Another advantage of NCH in terms of computational speed arises in the context of multi-class classification tasks. This occurs because only same-class objects are considered in the estimation of a (soft) distance to a convex hull, and not the whole data set. The decision surface of the NCH classifier is not explicitly computed because the classification process for each test point is independent of the classification process for other test points. That is why the classification process is instance-based in nature. In sum, the NCH method can be considered as a type of instance-based large-margin classifier.

The paper is organized as follows. First we provide some intuition behind the NCH classifier and a formal definition of it. Next, we discuss the technical aspects of the classifier – derivation and implementation. Finally, we show some experimental results on popular data sets and then conclude.

2 Nearest Convex Hull Classifier: Definition and Motivation

At the outset, consider a binary data set of positive and negative objects $\langle I^+, I^- \rangle$ from $\mathbb{R}^n$. Formally, the task is to separate the two classes of objects with a decision surface that performs well on a test data set. This task is formalized as finding a (target) function $f : \mathbb{R}^n \rightarrow \{-1, 1\}$ such that f will classify correctly unseen observations. The extension to the multi-class case is straightforward. The decision rule of the NCH classifier is the following: *a test point* $\mathbf{x}$ *should be assigned to that class, which convex hull is closest to* $\mathbf{x}$.

Let us consider the so-called separable case where the classes are separable by a hyperplane and draw an intuitive comparison between NCH and the popular SVM classifier. See Figure 1 for an illustrative binary classification example. Panels (a) and (c) refer to SVM classification, and Panels (b) and (d) refer to NCH classification. In SVM classification, the target function is a hyperplane of the form $\mathbf{w}'\mathbf{x} + b = 0$, where $\mathbf{w}$ is a vector of coefficients and b in the intercept. The SVM hyperplane $\mathbf{w}^{*\prime}\mathbf{x} + b^* = 0$ (denoted as h_{SVM}) is the one that separates the classes with the widest margin, where a margin is defined as the distance between a (separating) hyperplane and the closest point to it from the training data set. In terms of Figure 1, Panel (a), the width of white band is equal to twice the margin, which is shown in Panel (c). The closest point to h_{SVM} is defined to lie on the hyperplane $\mathbf{w}^{*\prime}\mathbf{x} + b^* = 1$ if this point is positively labeled,

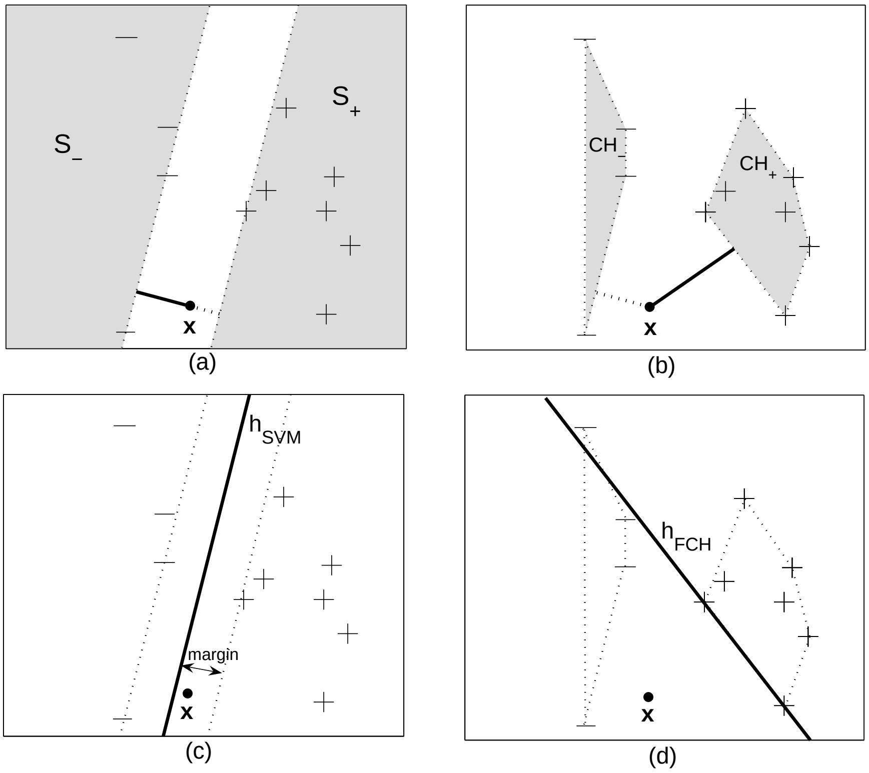

Fig. 1. Classification of a test point $\mathbf{x}$ with SVM in Panels (a) and (c), and NCH in Panels (b) and (d) on a binary data set. In Panel (a), the white band has the largest possible width, which is equal to twice the margin, shown in Panel (c). The points to the left and to the right of the band form shaded sets S_- and S_+, respectively. Test point $\mathbf{x}$ receives label $+1$ since it is farther from S_- than S_+. In Panel (b) point $\mathbf{x}$ is classified as -1 since it is farther from the convex hull of the positive points, CH_+, than from the convex hull of the negative points, CH_-.

or on the hyperplane $\mathbf{w}^{*\prime}\mathbf{x} + b^* = -1$ if this closest point is negatively labeled. For all points $\mathbf{x}$ that lie outside the margin it holds that either $\mathbf{w}^{*\prime}\mathbf{x} + b^* < -1$ or $\mathbf{w}^{*\prime}\mathbf{x} + b^* > 1$. The former set of points is defined as S_-, and the latter set of points is defined as S_+. For any test point $\mathbf{x}$, the SVM classification rule can be formulated as follows: a test point $\mathbf{x}$ should be classified as -1 if it is farther away from set S_+ than from set S_-; otherwise $\mathbf{x}$ receives label $+1$.

It has been argued (see, e.g., [4], [14]) that SVM classification searches for a balance between empirical error (or, the goodness-of-fit over the training data) and complexity, where complexity is proxied by the distance between sets S_+ and S_- (that is, twice the margin). In the separable case at hand, the empirical error of h_{SVM} is zero since it fits the data perfectly. Also, complexity and margin width are inversely related: the larger the margin, the lower the associated complexity. The balance between empirical error and complexity can intuitively be approached from an instance-based viewpoint as well. In this case, complexity is imputed in the classification of each separate test object/instance. Thus, the larger the distance from a test object $\mathbf{x}$ to the farther one of the two sets S_+ and S_-, the lower the complexity associated with the classification of $\mathbf{x}$.

The NCH classifier can also be considered from a fit-versus-complexity standpoint. Let us denote by CH_+ and CH_- the set of points that form the convex hulls of the positive and negative objects, respectively (see Figure 1, Panel (b)). Somewhat similarly to SVM, in NCH classification one considers the distance to the farther one of the two convex hulls CH_+ and CH_- as a proxy for the complexity associated with the classification of $\mathbf{x}$. Quite interestingly, this distance is always as big as or bigger than the distance from $\mathbf{x}$ to the farther of sets S_+ and S_-. This property holds since the convex hull of the $+1$ (-1) points is a subset of S_+ (S_-), as can be seen in Figure 1. Therefore, if one considers the distance to the farther-away convex hull as a proxy for complexity associated with the classification of $\mathbf{x}$, then NCH classification is characterized by a lower complexity than SVM classification. However, the fit over the training data of NCH may turn out to be inferior to SVM in some cases. Let h_{FCH} denote the hyperplane that is tangent to the farther-away convex hull of same-class training data points, and is perpendicular to the line segment that represents the distance between $\mathbf{x}$ and this convex hull, as in Figure 1, Panel (d). Thus, the distance between $\mathbf{x}$ and h_{FCH} equals the distance between $\mathbf{x}$ and the farther convex hull. Effectively, in NCH classification $\mathbf{x}$ is classified using h_{FCH}. Notice that by definition h_{FCH} separates without an error either the positive or the negative observations, depending on which convex hull is farther from $\mathbf{x}$. Thus, h_{FCH} is not guaranteed to have a perfect fit over the whole data set that consists of both positive and negative points, as illustrated in Figure 1, Panel (d). As a consequence, it is not clear a priory whether NCH or SVM will strike a better balance between fit and complexity in the classification of a given point $\mathbf{x}$: there is a gain for NCH coming from decreased complexity (in the form of an increased distance) vis-a-vis SVM on the one hand, accompanied by a potential loss arising from a possible increased empirical error of h_{FCH} over the whole training data set, on the other.

NCH has the property that the extent of proximity to a given class is determined without taking into consideration objects from other classes. This property contrasts with the SVM approach, where the sets S_+ and S_- are not created independently of each other. A similar parallel can be drawn between LDA and QDA methods. In LDA, one first determines the Mahalanobis distances from $\mathbf{x}$ to the centers of the classes using a common pooled covariance matrix and then classifies $\mathbf{x}$ accordingly. In QDA, one uses a separate covariance matrix for each class. Analogically, the NCH classifier first determines the Euclidean distance from $\mathbf{x}$ to the convex hulls of each of the classes and then classifies $\mathbf{x}$ accordingly. In sum, loosely speaking one may think of the shift from SVM to NCH as resembling the shift from LDA to QDA.

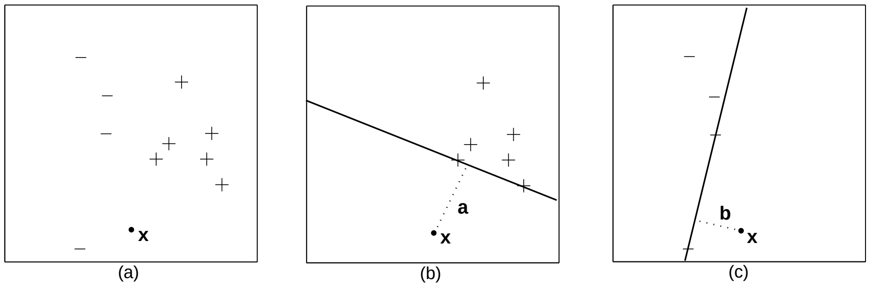

Fig. 2. Classification of a test point $\mathbf{x}$ with NCH on the binary data set in Panel (a) in two steps. At stage one (Panel (b)), a test point $\mathbf{x}$ is added to a data set that contains only the positive class, and the distance $\mathbf{a}$ from $\mathbf{x}$ to the convex hull of this class is computed. At stage two (Panel (c)), $\mathbf{x}$ is added to a data set that contains only the negative class, and the distance $\mathbf{b}$ from $\mathbf{x}$ to the convex hull of this class is computed. If $\mathbf{a} > \mathbf{b}$ ($\mathbf{a} < \mathbf{b}$), then $\mathbf{x}$ is assigned to the negative (positive) class.

3 Estimation

3.1 Separable case

Consider a data set of l objects from k different groups, or classes. Let l_k denote the number of objects in the k^{th} class. According to NCH, a test point $\mathbf{x}$ is assigned to that class, to which the distance is minimal. In the separable case, the distance to a class is defined as the distance to the convex hull of the objects from that class. The algorithm for classifying $\mathbf{x}$ can be described as follows (see Figure (2)): first, compute the distance from $\mathbf{x}$ to the convex hull of each of the k classes; second, assign to $\mathbf{x}$ the label of the closest class. Formally, to find the distance from a test point $\mathbf{x}$ to the convex hull of the nearest class, the following quadratic optimization problem has to be solved for each class k:

$$\begin{aligned} \min_{\mathbf{w}_k, b_k} \quad & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k \\ \text{such that} \quad & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0,\ i = 1, 2, \ldots, l_k \\ & -(\mathbf{w}_k'\mathbf{x} + b_k) = 1 \end{aligned} \tag{1}$$

The distance between hyperplane $\mathbf{w}_k'\mathbf{x} + b_k = 0$ and $\mathbf{x}$ is defined as $1/\sqrt{\mathbf{w}_k'\mathbf{w}_k}$ by the last constraint of (1). This distance is maximal when $\frac{1}{2}\mathbf{w}_k'\mathbf{w}_k$ is minimal. At the optimum, it represents the distance from $\mathbf{x}$ to the convex hull of class k. The role of the first l_k inequality constraints is to ensure that the hyperplane classifies correctly each point that belongs to class k. Effectively, for each of the k classes, the l_k same-class objects are assigned label 1, and the test point is assigned label -1. Eventually, $\mathbf{x}$ is assigned to that class to which the distance is minimal, that is, which corresponding value for the objective function in (1) is maximal.

3.2 Nonseparable case

Optimization problem (1) can be solved for each k only if the test point lies outside the convex hull of each class k. A further complication arises if some of the convex hulls overlap. Then a test point could lie simultaneously in two or more convex hulls and its classification label would be undetermined. To cope with these situations, so-called slack variables can be introduced, similarly to the SVM approach. Consequently, the nonseparable version of optimization problem (1) that has to be solved for each class k becomes:

$$\begin{array}{rl} \min\limits_{\mathbf{w}_k, b_k, \boldsymbol{\xi}} & \frac{1}{2}\mathbf{w}_k'\mathbf{w}_k + C\sum\limits_{i=1}^{l_k}\xi_i \\ \text{such that.} & \mathbf{w}_k'\mathbf{x}_i + b_k \geq 0 - \xi_i,\ \ \xi_i \geq 0,\ \ i = 1, 2, \ldots, l_k \\ & -(\mathbf{w}_k'\mathbf{x} + b_k) = 1. \end{array} \tag{2}$$

Note that in (2) the points that are incorrectly classified are penalized linearly via the term $\sum_{i=1}^{l_k}\xi_i$. If one prefers a quadratic penalization of the classification errors, then the sum of squared errors $\sum_{i=1}^{l_k}\xi_i^2$ should be substituted for $\sum_{i=1}^{l_k}\xi_i$ in (2). One can go even further and extend the NCH algorithm in a way analogical to LS-SVM ([7]) by imposing in (2) that constraints $\mathbf{w}_k'\mathbf{x}_i + b_k \geq 0 - \xi_i$ hold as equalities, on top of substituting $\sum_{i=1}^{l_k}\xi_i^2$ for $\sum_{i=1}^{l_k}\xi_i$.

Each of the k (primal) optimization problems pertaining to (2) can be expressed in dual form[1] as:

$$\begin{array}{rl} \max\limits_{\boldsymbol{\alpha}} & \alpha_{l_k+1} - \frac{1}{2}\sum_{i,j=1}^{l_k+1}\alpha_i\alpha_j y_i y_j(\mathbf{x}_i'\mathbf{x}_j) \\ \text{such that} & 0 \leq \alpha_i \leq C,\ \ i = 1, 2, \ldots, l_k,\ \text{ and }\ \sum_{i=1}^{l_k+1} y_i\alpha_i = 0, \end{array} \tag{3}$$

where the α's are the Lagrange multipliers associated with the respective k^{th} primal problem. Here α_{l_k+1} is the Lagrange multiplier associated with the equality constraint $-(\mathbf{w}_k'\mathbf{x} + b_k) = 1$. In each problem $y_i = 1, i = 1, 2, \ldots, l_k$ and $y_{l_k+1} = -1$. The advantage of the dual formulation (3) is that different *Mercer* kernels can be employed to replace the inner product $\mathbf{x}_i'\mathbf{x}_j$ in (3) in order to obtain nonlinear decision boundaries, just like in the SVM case. Three popular kernels are linear $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \mathbf{x}_i'\mathbf{x}_j$, polynomial of degree d $\kappa(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i'\mathbf{x}_j + 1)^d$ and the Radial Basis Function (RBF) kernel $\kappa(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma \parallel \mathbf{x}_i - \mathbf{x}_j \parallel^2)$, where the manually-adjustable γ parameter determines the proximity between $\mathbf{x}_i$ and $\mathbf{x}_j$.

A total of k NCH optimization problems have to be solved to determine the class of any test point $\mathbf{x}$. This property provides for the fact that the NCH decision boundary is in general implicit and nonlinear, even in case the original data is not mapped into a higher-dimensional space via a kernel. Figure 3 demonstrates that this property does not hold in general for Support Vector Machines,

[1] The derivation of the dual problem resembles the one used in SVM (see, e.g., [4]).

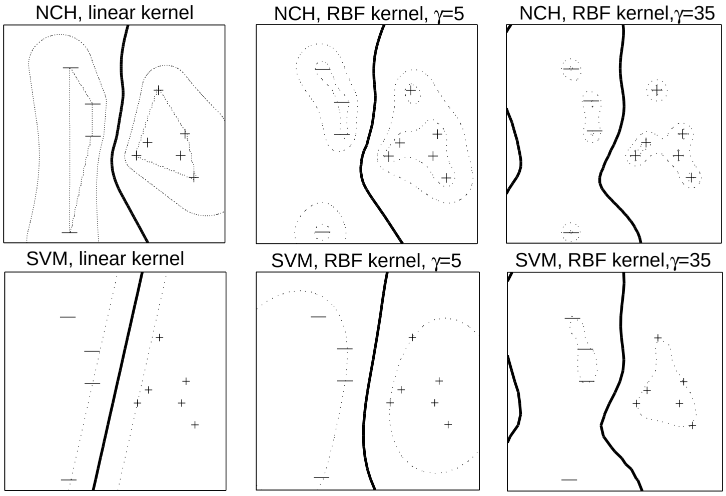

Fig. 3. Decision boundaries for NCH and SVM using the linear and RBF kernels on a linearly separable data set. The dashed contours for the NCH method are iso-curves along which the ratio of the distances to the two convex hulls is constant.

for instance. This figure also illustrates that the NCH decision boundary appears to be less sensitive to the choice of kernel and kernel parameters than the respective SVM boundary.

Technically speaking, in case the convex hulls do not overlap, NCH could be solved using the standard SVM optimization formulation (see, e.g., [14], [4]). In this case one searches for the widest margin between each of the k classes and a test point $\mathbf{x}$. This margin represents the distance from $\mathbf{x}$ to the convex hull of the k^{th} class. The class for which the margin is smallest is the winning one. The standard nonseparable-case SVM formulation cannot however be automatically applied to the nonseparable NCH case, since the equality constraint in (2) will not be satisfied in general.

4 Experiments on Some UCI and SlatLog Data Sets

The basic optimization algorithm for Nearest Convex Hull classification (3) is implemented via a modification of the freely available LIBSVM software ([5]). We tested the performance of NCH on several small- to middle-sized data sets that are freely available from the SlatLog and UCI repositories ([12]) and have been analyzed by many researchers and practitioners (e.g. [3], [8], [9], [13] and others): *Sonar*, *Voting*, *Wisconsin Breast Cancer* (W.B.C.), *Heart*, *Australian Credit Approval* (A.C.A.), and *Hepatitis* (Hep.). Detailed information on these data sets can be found on the web sites of the respective repositories.

We compare the results of NCH to those of several state-of-art techniques: Support Vector Machines (SVM), Linear and Quadratic Discriminant Analysis

Table 1. Leave-one-out accuracy rates (in %) of the Nearest Convex Hull classifier as well as some standard methods on several data sets. Rbf, 2p and lin stand for Radial Basis Function, second-degree polynomial and linear kernel, respectively

	NCH	NCH	NCH	SVM	SVM	SVM								
	rbf	2p	lin	rbf	2p	lin	NB	LR	LDA	QDA	MLP	*k*NN	DS	C4.5
Sonar	**91.4**	90.4	88.0	88.9	82.2	80.8	67.3	73.1	75.5	74.9	81.3	86.5	73.1	71.2
Voting	95.9	85.5	95.9	96.5	96.3	96.8	90.3	96.5	95.9	94.2	94.9	93.3	95.9	**97.0**
W.B.C.	**97.4**	97.1	97.3	97.0	96.9	96.9	96.0	96.1	96.0	91.4	95.0	97.0	92.4	95.3
Heart	**85.6**	82.6	84.1	**85.6**	81.1	85.6	83.0	83.7	83.7	81.5	78.9	84.4	76.3	75.2
A.C.A.	86.4	85.4	86.1	**87.4**	79.9	87.1	77.1	86.4	85.8	85.2	84.8	85.9	85.5	83.8
Hep.	85.2	84.5	84.5	**86.5**	86.5	86.5	83.2	83.9	85.8	83.9	79.4	85.8	79.4	80.0

(LDA and QDA), Logistic Regression (LR), Multi-layer Perceptron (MLP), *k*-Nearest Neighbor (*k*NN), Naive Bayes classifier (NB) and two types of Decision Trees – Decision Stump (DS) and C4.5. The experiments for the NB, LR, MLP, *k*NN, DS and C4.5 methods have been carried out with the WEKA learning environment using default model parameters, except for *k*NN. We refer to [15] for additional information on these classifiers and their implementation. We measure model performance by the leave-one-out (LOO) accuracy rate. Because we aim at comparing several methods, LOO seems to be more suitable than the more general *k*-fold cross-validation (CV), because it always yields one and the same error rate estimate for a given model, unlike the CV method (which involves a random split of the data into several parts).

Table 1 presents performance results for all methods considered. Some methods, namely *k*NN, NCH and SVM, require tuning of model parameters. In these cases, we report only the highest LOO accuracy rate obtained by performing a grid search for tuning the necessary parameters. Overall, the NCH classifier performs quite well on all data sets, and achieves best accuracy rates on three data sets. SVM also perform best on three data sets. The rest of the techniques show relatively less favorable and more volatile results. For example, the C4.5 classifier performs best on the *Voting* data set, but achieves rather low accuracy rates on two other data sets – *Sonar* and *Heart*. Note that not all data sets are equally easy to handle. For instance, the performance variation over all classifiers on the *Voting* and *Breast Cancer* data sets is rather low, whereas on the *Sonar* data set it is quite substantial.

5 Conclusion

We have introduced a new technique that can be considered as a type of an instance-based large-margin classifier, called Nearest Convex Hull classifier (NCH). NCH assigns a test observation to the class, which convex hull is closest. Convex-hull overlap is handled via the introduction of slack variables and/or kernels. NCH induces an implicit and generally nonlinear decision surface between the classes. One of the advantages of NCH is that an extension from binary to multi-class classification tasks can be carried out in a straightforward way. Others are its alleged robustness to outliers and good generalization qualities. A potential weak point of NCH, which also holds for SVM, is that it is not clear a priori which type of kernel and what value of the tuning parameters should be used. Furthermore, we do not address the issue of attribute selection and the estimation of class-membership probabilities. Further research could also concentrate on the application of NCH in more domains, on faster implementation suitable for analyzing large-scale data sets, and on the derivation of theoretical test-error bounds.

References

1. Ben-Hur, A., Horn, D., Siegelmann, H.T., Vapnik V.: Support Vector Clustering. Journal of Machine Learning Research **2** (2001) 125–137
2. Bennett, K., Bredensteiner, E.: Duality and Geometry in SVM Classifiers. In Proceddings of the 17th International Confefence on Machine Learning, Morgan Kaufmann, San Francisco, CA, (2000) 57–64
3. Breiman, L.: Bagging predictors. Machine Learning **24** (1996) 123–140
4. Burges, C.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery **2** (1998) 121–167
5. Chang, C.C., Lin, C.J.: LIBSVM: a library for support vector machines (2006) Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.
6. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines. Cambridge University Press (2000)
7. van Gestel, T.V., Suykens, J.A.K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., Moor, B.D., Vandewalle, J.: Benchmarking least squares support vector machine classifiers. Machine Learning **24** (2004) 5–32
8. King, R.D., Feng, C., Sutherland, A.: STATLOG: comparison of classification algorithms on large real-world problems. Applied Artificial Intelligence **9(3)** (1995) 289–334
9. Lim, T., Loh, W., Shih, Y.: A comparison of prediction accuracy, complexity, and training time for thirtythree old and new classification algorithms. Machine Learning **40** (1995) 203–228
10. von Luxburg, U., Bousquet, O.: Distance–Based Classification with Lipschitz Functions. Journal of Machine Learning Research **5** (2004) 669–695
11. Nalbantov, G.I., Bioch, J.C., Groenen, P.J.F.: Instance-based classification with support hyperplanes. Econometric Institute technical report, Erasmus University Rottedam (to appear)
12. Newman, D., Hettich, S., Blake, C., Merz, C.: UCI Repository of machine learning databases (1998) `http://www.ics.uci.edu/~mlearn/MLRepository.html` University of California, Irvine, Dept. of Information and Computer Sciences.
13. Perlich, C., Provost, F., Simonoff, J.S.: Tree induction vs. logistic regression: a learning-curve analysis. Journal Of Machine Learning Research **4** (2003) 211–255
14. Vapnik, V.N.: The Nature of Statistical Learning Theory. Springer-Verlag New York, Inc. (1995) 2nd edition, 2000.
15. Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools and techniques. Morgan Kaufman, San Francisco (2005) 2nd edition.