Margin-based Refinement for Linear Discriminant Analysis

Helene Dörksen and Volker Lohweg

Abstract For the two classes supervised learning problem, we present a refinement method for increasing the classification accuracy of an initial separating hyperplane in the feature space $\mathbb{R}^d$. The main idea corresponds to dimensionality reduction of, e.g. *LDA* separation, however not in its original form $\mathbb{R}^d \to \mathbb{R}$ but rather as dimensionality reduction $\mathbb{R}^d \to \mathbb{R}^j$ for some $j < d$ and $j > 1$. The method combines discriminant and margin-based properties of the separation. Due to efficiency reasons, we define rules for fast calculation of the refinement. Furthermore, we discuss theoretical fundamentals of our method and show its high performance by cross-validation tests on datasets from the *UCI Machine Learning Repository* with different numbers of features and objects. Due to the margin-based origin, the method is suitable for not well-balanced datasets. Cross-validation tests for not well-balanced data are given as well.

Helene Dörksen · Volker Lohweg
inIT – Institute Industrial IT, Ostwestfalen-Lippe University of Applied Sciences, Lemgo, Germany
✉ helene.doerksen@th-owl.de
✉ volker.lohweg@th-owl.de

Archives of Data Science, Series A
(Online First)
KIT Scientific Publishing
Vol. 4, No. 1, 2018

DOI 10.5445/KSP/1000085951/12
ISSN 2363-9881

1 Introduction

Creating an accurate classification rule is often one crucial part of machine learning tasks. In many cases, dimensionality reduction methods are able to contribute to the solution of the task (cf. (Guyon et al (2006)). Combining classifiers (Kuncheva (2004)) is another powerful technique to solve this problem.

We present a dimensionality reduction technique, which is able to increase the classification performance of some initial separating hyperplane by combining discriminant and margin-based properties. Specifying the application scenario, we concentrate on *Linear Discriminant Analysis* (*LDA*) (Fisher (1936)) and on the situation, where the boundaries of the classes might be lying relatively close to each other. As a logical intuitive workflow, the experts will fit a separating hyperplane exactly between boundaries of the classes. They rather will prefer a geometrical margin-based separation provided by, e.g., a linear *Support-Vector-Machine* (*SVM*) (Vapnik and Chervonenkis (1974); Vapnik (1998)), and not a statistics-based separation by *LDA*. Therefore, the statistical strengths of *LDA* will be neglected, however, possible overfitting drawbacks of *SVM* will be taken into account. The method, which we present in this paper, contributes to the discussed situation. It combines both classifiers and profits from the statistical nature of *LDA* and the margin-like strengths of *SVM*.

Our method is based on the *LDA* classification post-refined by a margin-based optimisation. The goal is to improve the initial *LDA* classification behaviour on the boundaries of classes. The procedure is related to the method described in (Dörksen and Lohweg (2017)), which is originally designed for *SVM* classification. Differently to the *SVM* refinement from (Dörksen and Lohweg (2017)), we define rules for the fast margin-based *LDA* refinement. Our method concentrates on the value of the margin of *LDA* classifier, which will be increased. In addition, by the rules we are looking for the features in the lower-dimensional spaces with more powerful discriminative properties as in the original space.

We prove several statements regarding theoretical fundamentals of the refinement method as well as examine it experimentally on datasets from the UCI (2017) Machine Learning Repository. For many examples we illustrate, that our method has higher generalisation ability and outperforms initial *LDA* as well as *SVM* results.

2 Approach

This section is organised as follows. In the first part we recall the theoretical foundations of *LDA* and *SVM*, which are required for our refinement method. In the second part we describe the approach and in the third part we define the rules for its fast implementation.

2.1 Foundations of LDA and SVM Classification

We consider the classification task for two classes $\mathbf{x}^+ \in \mathbf{T}^+$ and $\mathbf{x}^- \in \mathbf{T}^-$, where $\mathbf{x}^+, \mathbf{x}^-$ are subsets of objects. Objects are row vectors $\mathbf{x} \in \mathbb{R}^d$ with $\mathbf{x} = (x_1, \ldots, x_d)$ in the d-dimensional feature space $\mathbf{X} \subset \mathbb{R}^d$. *Feature vectors* $\mathbf{f}_1^+, \ldots, \mathbf{f}_d^+$ and $\mathbf{f}_1^-, \ldots, \mathbf{f}_d^-$ for classes $\mathbf{T}^+$ and $\mathbf{T}^-$ are defined as follows:

$$\mathbf{f}_i^+ := \{x_i \mid \text{ for all } x_i \in \mathbf{x}^+\},\ \mathbf{f}_i^- := \{x_i \mid \text{ for all } x_i \in \mathbf{x}^-\},\ i = 1, \ldots, d.$$

Feature vectors $\mathbf{f}_i^+$ resp. $\mathbf{f}_i^-$ are column vectors of the lengths corresponding to the number of objects in classes $\mathbf{T}^+$ resp. $\mathbf{T}^-$.

We assume that a linear combination of features $x_1, \ldots, x_d$ (or a projection of an object $\mathbf{x}$) is given as:

$$h(\mathbf{x}) = \langle \mathbf{a}, \mathbf{x} \rangle = \sum_{i=1}^{d} a_i x_i, \tag{1}$$

where $\mathbf{a} = (a_1, \ldots, a_d)$ and $a_i \in \mathbb{R}$, $i = 1, \ldots, d$. With some scalar (so-called *bias*) $c \in \mathbb{R}$, the rule for the *linear classification*, w.r.t. Equation (1) is the following:

$$\mathbf{x} \in \mathbf{T}^+, \text{ if } h(\mathbf{x}) \geq c \quad \text{and} \quad \mathbf{x} \in \mathbf{T}^-, \text{ if } h(\mathbf{x}) < c. \tag{2}$$

In the geometrical interpretation, the equality $h(\mathbf{x}) = c$ represents a hyperplane in $\mathbb{R}^d$, which separates the classes $\mathbf{T}^+$ and $\mathbf{T}^-$.

First classifier in our investigations is *LDA*. It provides a classification hyperplane by solving an optimisation task with respect to the so-called *Fisher's linear discriminant* (Fisher (1936)). Maximizing the discriminant corresponds to the searching for the direction, which maximizes the projected class means

while minimizing the variance of the classes in this direction. Without loss of generality, in the theoretical part of our work we do not concentrate on the original value of *Fisher's linear discriminant*, but rather on the almost equivalent distance metric d (Guyon et al (2006)), which we name also *discriminant* in our framework. Distance d relies on the both functions *mean* $\mu : \mathbb{R}^n \to \mathbb{R}$ and *variance* $\sigma^2 : \mathbb{R}^n \to \mathbb{R}$, though for a vector $\mathbf{v} = (v_1, \ldots, v_n)$ holds:

$$\mu(\mathbf{v}) = \frac{1}{n}\sum_{k=1}^{n} v_k, \quad \sigma^2(\mathbf{v}) = \frac{1}{n}\sum_{k=1}^{n} (v_k - \mu(\mathbf{v}))^2 .$$

We set $\mu_{+i} := \mu(\mathbf{f}_i^+)$, $\mu_{-i} := \mu(\mathbf{f}_i^-)$ and $\sigma_{+i}^2 := \sigma^2(\mathbf{f}_i^+)$, $\sigma_{-i}^2 := \sigma^2(\mathbf{f}_i^-)$. The distance d between classes w.r.t. single feature vectors $\mathbf{f}_i^+$ and $\mathbf{f}_i^-$ is following:

$$d(\mathbf{f}_i^+, \mathbf{f}_i^-) := \frac{(\mu_{+i} - \mu_{-i})^2}{\sigma_{+i}^2 + \sigma_{-i}^2}. \tag{3}$$

The extention of the above definition depends on the linear combinations of feature vectors. For a vector $\mathbf{a} = (a_1, \ldots, a_d)$, we receive the corresponding linear combinations of feature vectors:

$$\mathbf{f}^+(\mathbf{a}) := a_1 \cdot \mathbf{f}_1^+ + \ldots + a_d \cdot \mathbf{f}_d^+ \quad \text{and} \quad \mathbf{f}^-(\mathbf{a}) := \quad a_1 \cdot \mathbf{f}_1^- + \ldots + a_d \cdot \mathbf{f}_d^-. \tag{4}$$

The terms $\mathbf{f}^+(\mathbf{a})$ resp. $\mathbf{f}^-(\mathbf{a})$ are column vectors of the same lengths as $\mathbf{f}_i^+$ resp. $\mathbf{f}_i^-$. Within settings

$$\mu_+ := \mu\left(\mathbf{f}^+(\mathbf{a})\right), \ \mu_- := \mu\left(\mathbf{f}^-(\mathbf{a})\right), \ \sigma_+^2 := \sigma^2\left(\mathbf{f}^+(\mathbf{a})\right), \ \sigma_-^2 := \sigma^2\left(\mathbf{f}^-(\mathbf{a})\right),$$

the distance metric d is defined for linear combinations of feature vectors as follows:

$$d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right) = \frac{(\mu_+ - \mu_-)^2}{\sigma_+^2 + \sigma_-^2}. \tag{5}$$

Thus, the metric d is based on statistical characteristics of the feature vectors and their linear combinations. Further, *LDA* corresponds to the solution of the optimisation task:

$$\max_{\mathbf{a}} d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right) \text{ for } \mathbf{a} \in \mathbb{R}^d.$$

Regarding scaling and translation, the discriminant d has the following properties:

1. Scaling: For an enlarging or shrinking of $\mathbf{a}$ with respect to $s \in \mathbb{R}$, $s \neq 0$, which is $s \cdot \mathbf{a} = (s \cdot a_1, \ldots, s \cdot a_d)$, holds:

$$d\left(\mathbf{f}^+(s \cdot \mathbf{a}), \mathbf{f}^-(s \cdot \mathbf{a})\right) = |s| \cdot d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right). \tag{6}$$

2. Translation: For each translation vector $\mathbf{t} = (t_1, \ldots, t_d) \in \mathbb{R}^d$ and linear combinations of the translations of feature vectors, i.e.
$a_1 \cdot \left(\mathbf{f}_1^+ + t_1\right) + \ldots + a_d \cdot \left(\mathbf{f}_d^+ + t_d\right) =: [\mathbf{f}^+ + \mathbf{t}](\mathbf{a})$ resp.
$a_1 \cdot \left(\mathbf{f}_1^- + t_1\right) + \ldots + a_d \cdot \left(\mathbf{f}_d^- + t_d\right) =: [\mathbf{f}^- + \mathbf{t}](\mathbf{a})$ holds:

$$d\left([\mathbf{f}^+ + \mathbf{t}](\mathbf{a}), [\mathbf{f}^- + \mathbf{t}](\mathbf{a})\right) = d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right). \tag{7}$$

Our second classifier is *SVM*. The *SVM* provides a geometrical solution for the separating hyperplane between classes by searching for the largest *margin* ρ, which represents the distance between boundaries of the classes. We illustrate in Figure 1 the geometrical interpretation of *margin* for a simple example in a two-dimensional feature space. For the separable case in canonical form (Schölkopf and Smola (2002)) the *margin*

$$\rho = \frac{2}{\|\mathbf{a}\|} = \frac{2}{\sqrt{a_1^2 + \ldots + a_d^2}} \tag{8}$$

is maximized. Within labels $y = 1$ for all $\mathbf{x}^+ \in \mathbf{T}^+$ and $y = -1$ for all $\mathbf{x}^- \in \mathbf{T}^-$, it is equivalent to the solution of the problem:

$$\begin{aligned} &\min \frac{1}{2}\|\mathbf{a}\|^2 \\ &\text{subject to } y_i\left(\langle \mathbf{a}, \mathbf{x}_i \rangle - c\right) \geq 1 \\ &\text{with } \mathbf{x}_i \in \{\mathbf{x}^+, \mathbf{x}^-\},\ \forall i. \end{aligned} \tag{9}$$

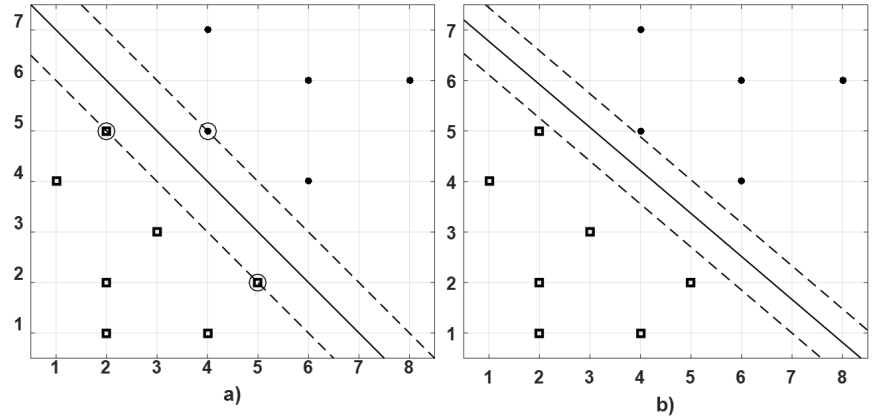

Figure 1: Two classes □ and • are considered. In a) the *SVM* classification solution with $\mathbf{a}_{SVM} = (-1, -1)$ is shown by the solid line. The so-called *support vectors* (more details found in (Schölkopf and Smola (2002)) are indicated by ◯. The distance from the classification boundary to the *support vectors* is equal to $\rho_{SVM}/2$. From formula (10) $\rho_{SVM} = 2/\sqrt{2} \approx 1.41$. The discriminant value (4) for $\mathbf{a}_{SVM}$ is $d_{SVM} \approx 3.20$. For the same classes, in b) the *LDA* classification solution with $\mathbf{a}_{LDA} = (-1.9480, -2.2868)$ is shown by the solid line. Here, the discriminant value (4) is $d_{LDA} \approx 4.68$ and $\rho_{LDA} \approx 0.67$. Dashed lines in a) and b) represent all points having the distances $\rho_{SVM}/2$ resp. $\rho_{LDA}/2$ to the classification boundaries.

For the non-separable case *slack variables* ξ_i with $\xi_i \geq 0$, $\forall i$ are defined. Slack variables store the deviation from the margin ρ in order to relax the constraints (Alpaydin (2010)). A *soft margin* classifier for a non-separable case is the solution of the problem:

$$\begin{aligned} &\min \frac{1}{2}\|\mathbf{a}\|^2 + C\sum_i \xi_i \\ &\text{subject to } y_i\left(\langle \mathbf{a}, \mathbf{x}_i\rangle - c\right) \geq 1 - \xi_i \\ &\text{with } \mathbf{x}_i \in \{\mathbf{x}^+, \mathbf{x}^-\},\ \xi_i \geq 0,\ \forall i, \end{aligned} \tag{10}$$

where the constant $C > 0$ determines the trade-off between margin maximization and training error minimization (Schölkopf and Smola (2002)). In our investigations, without loss of generality, we apply formula (10) for the calculation of the margin value ρ for both separable and non-separable cases.

2.2 Refinement of *LDA*

The workflow of the refinement approach consists of three basic steps presented below.

i. *Calculation of the initial separation hyperplane with LDA and the corresponding margin ρ_{LDA} resp. Equation* (10) *based on initial parameters* $\mathbf{a}_{LDA}$.

ii. *Dimensionality reduction of the feature space based on initial parameters* $\mathbf{a}_{LDA}$ *of the LDA hyperplane.*

iii. *Calculation of the SVM separation parameters* $\mathbf{b}_{SVM}$ *in reduced feature space, such that for the re-calculated margin of SVM hyperplane* ρ_{REF} *in the original space (resp. Equation* (19) *below) holds:*

$$\rho_{LDA} \leq \rho_{REF}. \tag{11}$$

We remind here that we apply formula (10) for the calculation of the margin value for both separable and non-separable cases. To illustrate the steps of the workflow, we assume that a classification rule in Equation (2) is calculated with *LDA*. The margin of the initial rule is $\rho_{LDA} = 2/\|\mathbf{a}\|$. The original *LDA* projection is a linear mapping $h : \mathbb{R}^d \to \mathbb{R}$ into a one-dimensional space with:

$$h(\mathbf{x}) = h(x_1, \ldots, x_d) = \sum_{i=1}^{d} a_i x_i. \tag{12}$$

However, there are further *LDA* initiated linear projections $g : \mathbb{R}^d \to \mathbb{R}^j$ into lower dimensional spaces $\mathbf{U}$ with $1 < j < d$. We define:

Definition 1 Let $g : \mathbb{R}^d \to \mathbb{R}^j$ be a linear mapping into the feature spaces $\mathbf{U} \subset \mathbb{R}^j$ of dimensions $j = 2, \ldots, d-1$:

$$g(x_1, \ldots, x_d) = \begin{pmatrix} \sum\limits_{i \in I_1} a_i x_i \\ \vdots \\ \sum\limits_{i \in I_j} a_i x_i \end{pmatrix} := \begin{pmatrix} u_1 \\ \vdots \\ u_j \end{pmatrix}, \tag{13}$$

where for $k = 1, \cdots, j$ holds $I_k \subset \{1, \cdots, d\}$.

If all I_k are non-empty disjoint subsets of indices with the property that:

$$\bigcup_{k=1}^{j} I_k = \{1, \cdots, d\} \quad \text{and} \quad I_k \cap I_l = \emptyset \quad \text{for all} \quad l \neq k, \tag{14}$$

then g is called *refinement mapping*.

Due to the definition above, it is clear that $j < d$, i.e. g is a mapping, which leads to a dimensionality reduction of the initial feature space. In the refinement mapping g all possible $a_i x_i$'s, $i = 1, \ldots, d$ of h appear in some u_k, $k = 1, \ldots, j$ and each $a_i x_i$ appears only once. In the feature space $\mathbf{U}$ we calculate parameters for the new *SVM* hyperplane as:

$$\tilde{h}(\mathbf{u}) = \tilde{h}(u_1, \ldots, u_j) = \sum_{i=1}^{j} b_i u_i. \tag{15}$$

Thus, for $b_1, \ldots, b_j \in \mathbb{R}$ and scalar $\tilde{c}$ the rule for the linear classification in the feature space $\mathbf{U}$ is the following:

$$\mathbf{x} \in \mathbf{T}^+, \text{ if } \sum_{i=1}^{j} b_i u_i \geq \tilde{c} \quad \text{and} \quad \mathbf{x} \in \mathbf{T}^-, \text{ if } \sum_{i=1}^{j} b_i u_i < \tilde{c}. \tag{16}$$

The margin of *SVM* in the space $\mathbf{U}$ is obviously $\rho_{SVM} = 2/\|\mathbf{b}\|$. It is clear, that the direct comparison of ρ_{LDA} and ρ_{SVM} makes no sense. We have to re-calculate the margin of $\tilde{h}$ in the original space. Obviously, the formula is:

$$\rho_{REF} = \frac{2}{\sqrt{b_1^2 \sum_{i \in I_1} a_i^2 + \cdots + b_j^2 \sum_{i \in I_j} a_i^2}}. \tag{17}$$

Example 1 We illustrate the dimensionality reduction principle of a projection g on the dataset *iris* (found in UCI (2017) Machine Learning Repository or in MATLAB R2016b as *fisheriris*). The sample has four features and three classes with 50 objects in each class. The classes are: *'setosa'*, *'versicolor'* and *'virginica'*. We consider the classes *'versicolor'* and *'virginica'* and calculate the *LDA* solution (by MATLAB R2016b code *classify* with *coef* as a structure array containing coefficients describing the boundary between classes; *coef* consists of the linear term and a constant term. The linear term corresponds to

the parameters of the linear combination h in Equation (1), the constant term corresponds to the bias c in Equation (2). The calculated projection h and the bias c for the dataset are the following:

$$h(\mathbf{x}) = 3.5563\, x_1 + 5.5786\, x_2 - 6.9701\, x_3 - 12.3860\, x_4$$
$$\text{and} \quad c = -16.6631 \tag{18}$$

and the classification accuracy for the boundary between classes is 97%. We are interested in a more powerful classification rule with respect to the classification accuracy. Thus, we construct the linear projection g for $j = 2$, $I_1 = \{1\}$ and $I_2 = \{2, 3, 4\}$, i.e.

$$g(\mathbf{x}) = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} = \begin{pmatrix} 3.5563\, x_1 \\ 5.5786\, x_2 - 6.9701\, x_3 - 12.3860\, x_4 \end{pmatrix} \tag{19}$$

Figure 2 shows the classes in the feature space $\mathbf{U} \subset \mathbb{R}^2$.

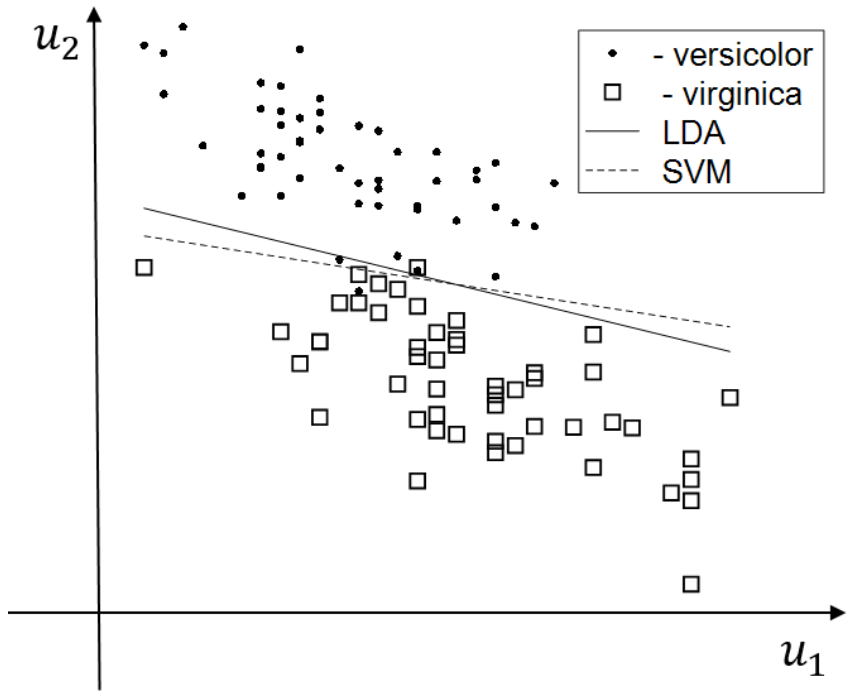

Figure 2: *LDA* initiated two-dimensional feature space **U** and *iris* sample projected into it are illustrated. The original *LDA* boundary corresponds to the solid line: $u_1 + u_2 = c$.

In the feature space **U** we are able to increase the classification accuracy till 98% if we now apply the *SVM* method. Based on *SVM*, the new classification rule is the following:

$$\mathbf{x} \in \mathbf{T}^{+}, \text{ if } b_1\, u_1 + b_2\, u_2 \geq \tilde{c} \quad \text{and} \quad \mathbf{x} \in \mathbf{T}^{-}, \text{ if } b_1\, u_1 + b_2\, u_2 < \tilde{c}, \tag{20}$$

where $b_1 = 0.6074$, $b_2 = 0.9579$ and $\tilde{c} = -24.0495$. It can be easily seen, that the hyperplane $b_1\, u_1 + b_2\, u_2 = \tilde{c}$ in the space **U** is equivalent to the hyperplane

$$\tilde{h}(\mathbf{x}) = b_1\, a_1\, x_1 + b_2 \cdot (a_2\, x_2 + a_3\, x_3 + a_4\, x_4) = \tilde{c} \tag{21}$$

in the original space **X** and, in general, hyperplanes $h(\mathbf{x}) = c$ and $\tilde{h}(\mathbf{x}) = \tilde{c}$ are different, i.e. they lead to different classification results. The margins of $h(\mathbf{x})$ and $\tilde{h}(\mathbf{x})$ are comparable. They are resp. $\rho_{LDA} = 0.1276$ and $\rho_{REF} = 0.1353$. Thus, we were able to find a new separating hyperplane with larger margin and higher classification accuracy as the initial *LDA*.

Some properties of linear mappings g, which transform the original d-dimensional feature space $\mathbf{X} \subset \mathbb{R}^d$ into lower-dimensional feature space $\mathbf{U} \subset \mathbb{R}^j$ for $j = 2, \ldots, d-1$, are described in proposition (1):

Proposition 1. *Assume a linear projection* $h(\mathbf{x})$ *from Equation* (1) *and* refinement mappings $g(\mathbf{x})$ *are considered. It holds:*

1. *Classification rules for* $h(\mathbf{x})$ *w.r.t.* (2) *and for* $\sum_{i=1}^{j} b_i u_i$ *are equivalent, whether* $b_i = 1$ *for all* $i = 1, \ldots, j$ *and* $c = \tilde{c}$.
2. *There are* $\binom{d}{2}$ *different linear mappings, which transform the original* d*-dimensional feature space* **X** *into the* $d-1$*-dimensional feature space* **U**.
3. *The number of different transformations of the original* d*-dimensional feature space* **X** *into the two-dimensional feature space* **U** *is:*

$$\sum_{j=1}^{d/2-1} \binom{d}{j} + \frac{1}{2}\binom{d}{d/2}, \text{ if } d \text{ is even}; \tag{22}$$

$$\sum_{j=1}^{(d-1)/2} \binom{d}{j}, \text{ if } d \text{ is odd}. \tag{23}$$

Proof. 1. It follows from the definition of g:

$$\sum_{i=1}^{j} b_i u_i = \sum_{i=1}^{j} u_i = \sum_{i=1}^{d} a_i x_i, \tag{24}$$

i.e. the statement is true.

2. With different nomenclature, this statement is related to (Dörksen and Lohweg (2014, Prop. 1)).
3. The statement is related to (Dörksen and Lohweg (2014, Prop. 2)).

□

2.3 Rules for Fast Refinement

The calculation of all possible non equal mappings g might be a crucial task in the case the dimension of the feature space $\mathbf{X}$ is large. As one example, the calculation time of all $d-1$-dimensional feature spaces $\mathbf{U}$ is $O(d^2)$ and of two-dimensional feature spaces is $O(2^d)$ (Dörksen and Lohweg (2014, Prop. 3)). From that point of view, we are interested not in the best classification solution through all possible feature spaces $\mathbf{U}$. We are rather interested in rules for the fast searching for such $\mathbf{U}$, where we expect to receive higher classification accuracy as in the initial space $\mathbf{X}$. One possible rule, called *Min/Max rule* is described in (Dörksen and Lohweg (2017)). *Min/Max rule* finds the $d-1$-dimensional feature space $\mathbf{U}$ within the time that depends linearly on the number of features. We adopt this rule as follows:

Min/Max Rules for Margin-based Refinement

1. Find the set I of two indices k_1 and k_2 from $\{1,\cdots,d\}$ such that $a_{k_1}^2 + a_{k_2}^2 = \sum_{i\in I} a_i^2$ is minimal (for *Min Rule*) or maximal (for *Max Rule*), i.e. choose from $|a_1|,\cdots,|a_d|$ two with minimal or maximal values, respectively.
2. For the set I define the linear mapping $g : \mathbb{R}^d \to \mathbb{R}^{d-1}$ such that

$$g(\mathbf{x}) = (u_1,\ldots,u_{d-1})^T$$

with $u_1 = \sum\limits_{i\in I} a_i x_i$. The entries for u_i, $i \neq 1$ are all remaining $a_i x_i$'s with $1 \leq i \leq d$ and $i \notin I$.

Proof. By the selection $\sum_{i\in I} a_i^2$ with only two elements in I we expect that the refinement for a_i's, $i \notin I$ is marginal, i.e. $b_j \approx 1$, for $j \neq 1$. In that case the margin value ρ_{REF} depends mainly on both $\sum_{i\in I} a_i^2$ and b_1^2. Thus, the margin ρ_{REF} represented in Equation (19) will become large whether $\sum_{i\in I} a_i^2$ or b_1^2 (or both) are small. The influence of b_1 is not possible, but it is possible to select the appropriate a_i's. We deduce that $|a_i|$'s with small values strongly enlarge the margin. This fact leads to the *Min Rule*. For the *Max Rule*, the argumentation for the selection of the large $|a_i|$'s is the opposite. By the margin optimisation within *Max Rule*, we degrade the contribution of $|a_i|$'s with large values and increase the margin. □

For these rules the time complexity of the dimensionality reduction is $O(d)$. The restriction of these rules is that only mappings $g : \mathbb{R}^d \to \mathbb{R}^{d-1}$ which reduce the dimension by 1 are considered. By this restriction, the performance of the refinement might be marginal for several data sets. To overcome this situation, we are interested in such mappings $g : \mathbb{R}^d \to \mathbb{R}^j$ for arbitrary j.

We concentrate on *LDA* fundamentals and corresponding solution, i.e. $\mathbf{a} = \mathbf{a}_{LDA}$. The discriminant value d from (4) is related to the discriminative power of feature vectors, i.e. higher d means higher discriminative power of the separation. Our idea is to construct mappings $g : \mathbb{R}^d \to \mathbb{R}^j$, such that the feature vectors in the space $\mathbf{U}$ have higher discriminative power than the feature vectors in the original space $\mathbf{X}$. Within a set I of indices from $\{1, \cdots, d\}$, corresponding vector $\mathbf{a}_I$ and the setting:

$$\mathbf{f}^+(\mathbf{a}_I) := \sum_{i\in I} a_i \mathbf{f}_i^+ \quad \text{and} \quad \mathbf{f}^-(\mathbf{a}_I) := \sum_{i\in I} a_i \mathbf{f}_i^-,$$

this idea leads to the following rule:

Discriminant Rule

1. Find a set I of indices from $\{1, \cdots, d\}$ such that for the corresponding a_i's, $i \in I$ holds:

$$d\left(\mathbf{f}^+(\mathbf{a}_I), \mathbf{f}^-(\mathbf{a}_I)\right) \geq d\left(\mathbf{f}_i^+, \mathbf{f}_i^-\right) \text{ for all } i \in I \tag{25}$$

2. If $I \neq \emptyset$ define the *refinement mapping* $g : \mathbb{R}^d \to \mathbb{R}^j$.

Proof. By defining the projection g we establish that the discriminative power of the feature vector combination is larger or equal than discriminative power of

each single feature vector of the set I, i.e. features in the space $\mathbf{U}$ have higher or equal discriminative power as features in the original space $\mathbf{X}$. □

We remark that the weights $\mathbf{a}_I$ for the feature vectors of the subset I correspond to the weights of the initial classification. The discriminant $d\left(\mathbf{f}^+(\mathbf{a}_I), \mathbf{f}^-(\mathbf{a}_I)\right)$ is calculated on these weights and does not need a re-optimisation.

The limitation of the presented *discriminant rule* arises by the question how to find such a set I with property (27). A combinatorial solution of this problem is unacceptable for large d. One possible technique to find the set I is the combination of the *discriminant rule* and *Min/Max rule*. It is simply based on the finding mapping $g : \mathbb{R}^d \to \mathbb{R}^j$ iteratively by the *Min/Max Rule* and testing property (27) in each iteration. The time complexity of this rule is $O(d \log d)$. For the pseudocode, see algorithm 1:

Algorithm 1 Discriminant Rule with Min/Max

1. Initiate $I = \emptyset$.
2. Choose the index k corresponding to $|a_k|$ with minimal/maximal value (within iteration of this step do not consider the $|a_k|$ which are already in I).
3. If property (27) is true, then add k to the set I and go to 2 again. Otherwise, go to 4.
4. If $I \neq \emptyset$ define *refinement mapping* $g : \mathbb{R}^d \to \mathbb{R}^j$.

Finally, we describe one rule, which is based on finding such subsets I of the feature vectors, that for the *LDA* solution $\mathbf{a} = \mathbf{a}_{LDA}$ holds:

$$d\left(\mathbf{f}^+(\mathbf{a}_I), \mathbf{f}^-(\mathbf{a}_I)\right) > d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right). \tag{26}$$

Also in this case, the weights $\mathbf{a}_I$ for the features of the subset I correspond to the weights of the initial classification. The discriminant $d\left(\mathbf{f}^+(\mathbf{a}_I), \mathbf{f}^-(\mathbf{a}_I)\right)$ is calculated on these weights and does not need re-optimisation.

Under the assumption that feature vectors are normally distributed (i.e. in our denotations it means $\mathbf{f}_i^+ \sim \mathcal{N}(\mu_{+i}, \sigma_{+i}^2)$ and $\mathbf{f}_i^- \sim \mathcal{N}(\mu_{-i}, \sigma_{-i}^2)$ for all

$i = 1, \ldots, d$), for their linear combination the following theorem is well-known from mathematical statistics:

Theorem 1 Assume that variables $X_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$, $1 \leq i \leq d$ are normally independent identically distributed. Then for their linear combination Y within parameters $a_i \in \mathbb{R}$, $i = 1, \ldots, d$ and some scalar $c \in \mathbb{R}$ holds:

$$Y = \sum_{i=1}^{d} a_i X_i + c \sim \mathcal{N}(\mu, \sigma^2), \text{ with } \mu = \sum_{i=1}^{d} a_i \mu_i + c, \; \sigma^2 = \sum_{i=1}^{d} a_i^2 \sigma_i^2. \quad (27)$$

Without loss of generality, we consider our classes in standardised form: The feature vectors in the class $\mathbf{T}^+$ have values $\mu_{+_i} = 0$ and $\sigma_{+_i}^2 = 1$ for all $i = 1, \ldots, d$. All feature vectors of the class $\mathbf{T}^-$ can be transposed respectively. Their means and standard deviations are μ_{-_i} and $\sigma_{-_i}^2$ for all $i = 1, \ldots, d$. We assume that all feature vectors are normally independent identically distributed. According to Theorem 1 it holds for the linear combination of feature vectors $\mathbf{f}_i^+$, $i = 1, \ldots, d$ from the class $\mathbf{T}^+$ within parameters $a_i \in \mathbb{R}^d$, $i = 1, \ldots, d$ and some scalar $c \in \mathbb{R}$:

$$\mathbf{f}^+(\mathbf{a}) = \sum_{i=1}^{d} a_i \mathbf{f}_i^+ + c \sim \mathcal{N}(\mu_+, \sigma_+^2), \quad (28)$$

with

$$\mu_+ = \sum_{i=1}^{d} a_i \mu_{+_i} + c = c \quad \text{and} \quad \sigma_+^2 = \sum_{i=1}^{d} a_i^2 \sigma_{+_i}^2 = \sum_{i=1}^{d} a_i^2. \quad (29)$$

The linear combination of the feature vectors in the class $\mathbf{T}^-$ distributed respectively $\mathcal{N}(\mu_-, \sigma_-^2)$, where

$$\mu_- = \sum_{i=1}^{d} a_i \mu_{-_i} + c \quad \text{and} \quad \sigma_-^2 = \sum_{i=1}^{d} a_i^2 \sigma_{-_i}^2. \quad (30)$$

We insert means and standard deviations into formula (4). It follows:

$$d\left(\mathbf{f}^+(\mathbf{a}), \mathbf{f}^-(\mathbf{a})\right) = \frac{\left(c - \left(\sum_{i=1}^{d} a_i \mu_{-_i} + c\right)\right)^2}{\sum_{i=1}^{d} a_i^2 + \sum_{i=1}^{d} a_i^2 \sigma_{-_i}^2} = \frac{\left(\sum_{i=1}^{d} a_i \mu_{-_i}\right)^2}{\sum_{i=1}^{d} a_i^2 (1 + \sigma_{-_i}^2)}. \quad (31)$$

Assume, that the subset I would be constructed by eliminating one single feature j, i.e. $I = \{1,\dots,d\} \setminus j$. Thus, we are interested in the relation between $d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right)$ and $d\left(\mathbf{f}^+(\mathbf{a}_I),\mathbf{f}^-(\mathbf{a}_I)\right)$. It holds:

$$d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right) = \frac{\left(\sum\limits_{i\in I} a_i\mu_{-_i} + a_j\mu_{-_j}\right)^2}{\sum\limits_{i\in I} a_i^2(1+\sigma_{-_i}^2) + a_j^2(1+\sigma_{-_j}^2)} \leq \frac{\left(\sum\limits_{i\in I} a_i\mu_{-_i} + a_j\mu_{-_j}\right)^2}{\sum\limits_{i\in I} a_i^2(1+\sigma_{-_i}^2)}. \tag{32}$$

Without loss of generality, we assume $a_j\mu_{-_j} \neq 0$. Further, assume that for the single terms of the nominator $\left(\sum\limits_{i\in I} a_i\mu_{-_i} + a_j\mu_{-_j}\right)^2$ in formula (34) it is valid:

$$\operatorname{sign}\sum_{i\in I} a_i\mu_{-_i} \neq \operatorname{sign} a_j\mu_{-_j} \quad \text{and} \quad \left|\sum_{i\in I} a_i\mu_{-_i}\right| > \left|a_j\mu_{-_j}\right|. \tag{33}$$

Within the assumption (**??**), it follows that eliminating the term $a_j\mu_{-_j}$ will increase the nominator, i.e. for the subset I it is true:

$$d\left(\mathbf{f}^+(\mathbf{a}_I),\mathbf{f}^-(\mathbf{a}_I)\right) > d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right).$$

Now we are able to prove the following statement:

Lemma 1. *Assume two classes* $\mathbf{T}^+$ *and* $\mathbf{T}^-$ *given in standardised form have normally independent identically distributed feature vector. Assume all* $a_i\mu_{-_i} \neq 0$*,* $i = 1,\dots,d$*. Further, within* $\operatorname{sign}\alpha = 1$ *for* $\alpha > 0$ *and* $\operatorname{sign}\alpha = -1$ *for* $\alpha < 0$*, for subsets* I_1 *and* I_2 *with* $I_1 \cup I_2 = \{1,\dots,d\}$ *and* $I_1 \cap I_2 = \emptyset$ *holds:*

$$\operatorname{sign}\sum_{i\in I_1} a_i\mu_{-_i} \neq \operatorname{sign}\sum_{i\in I_2} a_i\mu_{-_i}. \tag{34}$$

If $\left|\sum_{i\in I_1} a_i\mu_{-_i}\right| > \left|\sum_{i\in I_2} a_i\mu_{-_i}\right|$*, then* $d\left(\mathbf{f}^+(\mathbf{a}_{I_1}),\mathbf{f}^-(\mathbf{a}_{I_1})\right) > d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right)$*. Otherwise, i.e. if* $\left|\sum_{i\in I_1} a_i\mu_{-_i}\right| < \left|\sum_{i\in I_2} a_i\mu_{-_i}\right|$*, then* $d\left(\mathbf{f}^+(\mathbf{a}_{I_2}),\mathbf{f}^-(\mathbf{a}_{I_2})\right) > d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right)$*.*

Proof. The statement of the lemma is true by the fact that the value of $d\left(\mathbf{f}^+(\mathbf{a}),\mathbf{f}^-(\mathbf{a})\right)$ can be increased by enlarging the value of the nominator or reducing the value of the denominator. □

Based on the considerations above, we define the sign rule in algorithmus 2:

Algorithm 2 Sign Rule

1. Consider the classes in standardised form: the feature vectors of the class $\mathbf{T}^+$ have values $\mu_{+i} = 0$ and $\sigma_{+i}^2 = 1$ for all $i = 1, \ldots, d$. All feature vectors of the class $\mathbf{T}^-$ are transposed respectively.
2. Define subsets I_1 and I_2 as follows: $I_1 = \{i \mid a_i \mu_{-i} > 0\}$ and $I_2 = \{i \mid a_i \mu_{-i} < 0\}$. Thus, sign $\sum_{i \in I_1} a_i \mu_{-_i} = 1 \neq$ sign $\sum_{i \in I_2} a_i \mu_{-_i} = -1$.
3. If $\sum_{i \in I_1} a_i \mu_{-_i} > \left| \sum_{i \in I_2} a_i \mu_{-_i} \right|$, then define $I := I_1$. Otherwise, if $\sum_{i \in I_1} a_i \mu_{-_i} < \left| \sum_{i \in I_2} a_i \mu_{-_i} \right|$, then define $I := I_2$.
4. If $I \neq \emptyset$ define *refinement mapping* $g : \mathbb{R}^d \rightarrow \mathbb{R}^j$.

We remark, that the *Sign Rule* is constructed from the analysis of discriminative characteristics of feature vectors combinations and it is not margin-based. Therefore, the refinement within this rule will not neccesserily lead to the increasing of the margin. Nevertheless, in the next section, we will present some examples where this rule is working for the margin well.

3 Experimental Results

In this section we present experimental results of the application of the *Discriminant Rule with Min/Max* and *Sign Rule* to datasets from the Machine Learning Repository (UCI (2017)). The refinement results for these datasets indicate high performance. The results of the application of stand-alone *Min/Max Rules for Margin-based Refinement* to several samples are found in (Dörksen and Lohweg (2017)).

For the considered samples, the feature vectors in one class are standardised to have values $\mu_i = 0$ and $\sigma_i^2 = 1$ for all $i = 1, \ldots, d$; all feature vectors of another class are transformed respectively. Without loss of generality, the initial margin is standardised $\rho_{initial} = 2$ before refinement. Further, we define the

$Balance \geq 1$ of the sample as the number of objects $\#Obj(class)$ of one class divided by the number of the objects of the other, i.e.

$$Balance = \begin{cases} \frac{\#Obj(\mathbf{T}^+)}{\#Obj(\mathbf{T}^-)}\text{, if } \#Obj\,(\mathbf{T}^+) \geq \#Obj\,(\mathbf{T}^-)\,, \\ \frac{\#Obj(\mathbf{T}^-)}{\#Obj(\mathbf{T}^+)}\text{, otherwise.} \end{cases}$$

For the well-balanced samples is $Balance = 1$, not well-balanced samples have $Balance > 1$.

We present the classification rates in terms of accuracies (in %) and t_{K-1}-*statistics* of *K-fold cross-validation (cv) paired* t *test* (Alpaydin (2010)) for comparing two classification algorithms. For the tests, the complete dataset is divided randomly into K equalsized parts. To generate each pair, one of the K parts is keeped out as the training set. The remaining $K-1$ parts form the validation set. Doing this K times, K pairs are formed. Thus, the training sets consist of 10% and validation sets consist of 90% of the sample. For *K-fold cross-validation (cv) paired* t *test*, two classification algorithms are trained K times on the same sets. The error percentages of the classifiers on the corresponding validation sets are recordered as p_i^1 and p_i^2 for $i = 1, \ldots, K$.

If the two classification algorithms have the same error rates, then we expect them to have the same mean, or equivalently, that for the difference of their means holds $\mu(p_i^1 - p_i^2) = 0$. That leads to the following hypotesis test w.r.t. $p_i = p_i^1 - p_i^2$:

$$H_0 : \ \mu(p_i^1 - p_i^2) = 0 \text{ vs. } H_1 : \ \mu(p_i^1 - p_i^2) \neq 0$$

Within values

$$\mu = \frac{\sum_{i=1}^{K} p_i}{K}, \quad \sigma^2 = \frac{\sum_{i=1}^{K} (p_i - \mu)^2}{K-1},$$

the t_{K-1}-statistic is defined as:

$$t_{K-1} \sim \frac{\sqrt{K} \cdot \mu}{\sigma^2}.$$

The larger the value of the t_{K-1}-statistic is, the more it is likely that the algorithms have different error rates. E.g. for t_9: If $|t_9| > 2.26$, then we reject the hypothesis that the algorithms have the same error rate with 97.5% confidence. In our tests, we calculate t_{K-1}-statistic for comparing the initial classifier with

its refinement. In addition to the value t_{K-1}, in Table 1 the information about samples and classification calculations are following: The value $\#Feat = d$ is the dimention of the considered feature space (i.e. number of features). The number of all objects is:

$$\#Obj = \#Obj\left(\mathbf{T}^{+}\right) + \#Obj\left(\mathbf{T}^{-}\right).$$

The *LDA* hyperplane is calculated within MATLAB 2016b function *classify*, the *SVM* hyperplane within *svmtrain*. The refinement is performed by Algorithm 1 and Algorithm 2. The accuracy of *LDA* classification is denoted by $LDA(\%)$, the accuracy of refinement is $REF(\%)$. The refinement rule is indicated by *Min*, *Max* or *Sign*. The margin of the classification hyperplane after refinement is marked by ρ_{REF} (to be compared with standardised $\rho_{LDA} = 2$). In addition, the last column in Table 1 indicates the accuracy of the stand-alone *SVM*.

Table 1: Results of *K-fold cv paired* t *test* ($K = 10$) for refinement within initial *LDA*. In columns, overall accuracies, margin ρ_{REF} as well as t_{K-1}-statistics for benchmarking are given. Refinement rule is indicated by Min, Max or Sign. For initial margin is valid $\rho_{LDA} = 2$ for all samples. All samples are standardised with respect to one of the classes, such that for one class holds: $\mu_i = 0$ and $\sigma_i^2 = 1,\ \forall i = 1, \ldots, d$. For comparison, last column indicates accuracy of the stand-alone *SVM*. *NA (not available)* stands for the samples, where the *SVM* solution was not delivered by function *svmtrain*..

Dataset	#Feat	#Obj	Balance	*LDA* (%)	*REF* (%)	ρ_{REF}	t_{K-1}	*SVM* (%)
Breast Cancer	9	683	1.86	94.88	95.61 Max	542.10	3.37	94.13
Splice	60	2423	2.15	81.30	86.66 Min	3.03	14.25	NA
Banknote Authent.	4	1372	1.25	97.27	98.62 Sign	4.02	6.68	96.06

The results in Table 1 indicate clearly, that due to the refinement the margin values are increased and the classification rates are improved. Furthermore, according to the t_{K-1}-statistics, the refinement performs better as initial *LDA*. Even though the rules are announced for *LDA*, our method can be started with linear *SVM* as initial feature combination. In Table 2 we illustrate the refinement results for this case. The denotations in Table 2 are similar to such from Table 1 with the different initial classification hyperplane and corresponding accuracy indicated by *SVM* (%). The last column in Table 2 represents the accuracies of the stand-alone *LDA*.

Table 2: Results of *K-fold cv paired* t *test* for refinement witin initial *SVM*. Table denotations are similar to such from Table 1. Last column indicates accuracy of the stand-alone *LDA*. *NA (not available)* stands for the samples, where the *LDA* solution was not delivered by function *classify*.

Dataset	#Feat	#Obj	Balance	*SVM* (%)	*REF* (%)	ρ_{REF}	t_{K-1}	*LDA* (%)
CNAE	333	240	1.00	79.31	85.23 Min	7.56	2.48	NA
Heart	13	270	1.25	76.91	78.11 Min	3.14	2.79	73.95
Indian Liver	10	579	2.51	61.05	62.28 Max	61.30	2.41	59.73
Plan. Relax	12	182	2.5	53.96	54.81 Max	62.76	2.94	NA
Ecoli	6	220	1.86	95.45	96.08 Sign	9.33	3.35	NA

Also in this scenario, t_{K-1}-statistics illustrate, that the refinement outperforms the initial *SVM*.

Finally, we show some results on samples which are not well-balanced. Accuracy as an exclusive measure is not sufficient here. Scores based on true positives and false negatives of each class are suitable for such problems. *F-measures* are calculated as scores based on true positives and false negatives for each class, where:

$$F(class) = \frac{2\,TP(class)}{2\,TP(class) + FP(class) + FN(class)}, \tag{35}$$

and *TP, FP, FN* are resp. true positives, false positives, and false negatives. The value of *F-measure* lies in the interval $[0, 1]$, i.e. with respect to the classification accuracy in %, the value 0 corresponds to 0% resp. 1 to 100%. Table 3 represents overall *F-measures* for *K-fold cross-validation*. Due to small sizes of single classes, not all tests are executable with $K = 10$. For that reason, several tests were executed with $K = 5$ or $K = 3$.

Table 3: For *K-fold cross-validation*, overall *F-measures* for initial *LDA* or *SVM* and for the corresponding refinement are presented. *F-measure* is based on true positives and false negatives of each class.

Dataset	#Feat	#Obj	Balance	$F(\mathbf{T}^+)$ / $F(\mathbf{T}^-)$ *Initial*	$F(\mathbf{T}^+)$ / $F(\mathbf{T}^-)$ *REF*
Fertility ($K = 5$)	9	100	7.33	0.15 / 0.75 *LDA*	0.17 / 0.84 *Min*
Hepatitis ($K = 3$)	16	80	5.15	0.32 / 0.76 *LDA*	0.38 / 0.83 *Max*
Splice ($K = 10$)	60	2423	2.15	0.72 / 0.86 *SVM*	0.77 / 0.88 *Min*
Lung Cancer ($K = 5$)	56	19	1.11	0.52 / 0.36 *SVM*	0.57 / 0.43 *Sign*

The results of Table 3 illustrate, that the refinement is a powerfull method for the not well-balanced samples.

4 Conclusion and Outlook

We presented a refinement method which is able to increase the accuracy of classification and leads to a higher generalisation ability. Our method is based on dimensionality reduction and combines discriminant and margin-based properties of the separation between classes. Due to the relatively low time complexity $O(d \log d)$ (where d is the number of the considered features) of the *Discriminant Rule with Min/Max* as well as $O(d)$ of *Sign Rule*, we suggest to test the rules for their performance by classifier design within any hyperplane.

Our future investigations regarding this topic will consider research activities for the definition of other rules for fast refinement, their suitability to large data sets as well as possibilities of the incorporation of the refinement fundamentals into one single step. Furthermore, applicability of the method in the context of *Feature Extraction* and *Feature Selection* tasks will be analysed.

References

Alpaydin E (2010) Introduction to Machine Learning, 2nd edn. The MIT Press, Cambridge. ISBN: 978-0-262012-43-0.

Dörksen H, Lohweg V (2014) Combinatorial refinement of feature weighting for linear classification. In: Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA). DOI: 10.1109/ETFA.2014.7005106.

Dörksen H, Lohweg V (2017) Margin-based Refinement for Support-Vector-Machine Classification. In: Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods (ICPRAM), vol. 1. DOI: 10.5220/0006115502930300.

Fisher RA (1936) The use of multiple measurements in taxonomic problems. Annals of Eugenics 7:179–188. DOI: 10.1111/j.1469-1809.1936.tb02137.x.

Guyon I, Gunn S, Nikravesh M, Zadeh LA (2006) Feature Extraction: Foundations and Applications. Studies in Fuzziness and Soft Computing, Springer, New York. ISBN: 35-4035-487-5, DOI: 10.1007/978-3-540-35488-8.

Kuncheva LI (2004) Combining Pattern Classifiers: Methods and Algorithms. Wiley-Interscience. DOI: 10.1002/0471660264.

Schölkopf B, Smola AJ (2002) Learning with kernels: Support vector machines, regularization, optimization, and beyond. Adaptive computation and machine learning, MIT Press, Cambridge. ISBN: 978-0-262194-75-4.

UCI (2017) Machine Learning Repository. URL: `https://archive.ics.uci.edu/ml/datasets.html`.

Vapnik VN (1998) Statistical Learning Theory, 1st edn. Wiley, London. ISBN: 978-0-471030-03-4.

Vapnik VN, Chervonenkis A (1974) Theory of Pattern Recognition (in Russian). Nauka, Moscow.