OPTIMIZATION ERUDITORUM
Volume 1 (2024), No. 2, 88–100
https://doi.org/10.69829/oper-024-0102-ta02
Tulipa Opera Scholarum

A POLYHEDRAL CONIC FUNCTIONS BASED EMBEDDED FEATURE SELECTION METHOD

REFAIL KASIMBEYLI[1,2,*] AND OZNUR AY[1]

[1] *Department of Industrial Engineering, Faculty of Engineering, Eskisehir Technical University, Eskisehir 26555, Türkiye*
[2] *UNEC Mathematical Modeling and Optimization Research Center, Azerbaijan State University of Economics (UNEC), Istiqlaliyyat Str. 6, Baku, 1001, Azerbaijan*

Abstract. In this paper, a polyhedral conic functions based embedded feature selection method is proposed. The original PCF algorithm developed for classification, is reformulated such that, both feature selection and classification are performed simultaneously. The proposed algorithm is tested on some currently available real world data sets and compared with other well known feature selection and classification algorithms. It is shown that classifiers obtained by the new method, have better test accuracies on some test problems, exhibit comparable prediction performance on all data sets tested, and increase the generalization performance.

Keywords. Feature selection, polyhedral conic function, embedded methods, mathematical programming, classification, data mining.

1. Introduction

Feature selection is an important machine learning task in the context of supervised learning. In supervised learning framework, it refers to select a subset of attributes (*features*) that will maximize the prediction accuracy (or generalization ability) and classify the data using selected subset. Due to feature selection, higher learning performance (or prediction accuracy), lower computational cost, lower storage cost and more interpretable models are obtained in classification problems.

Feature selection methods can be categorized into three main types; filter, wrapper and embedded methods [11, 22, 15, 17]. In filter methods, feature selection and machine learning algorithm are done separately. First, the most relevant subset of features is selected by using a statistical measure, then learning algorithm is performed on the subset. Since there is no interaction between feature selection and machine learning algorithm, filter methods can be seen as a data preprocessing step, which reduces the dimensionality [11]. In literature, based on the chosen ranking criteria (statistical measures), many methods have been developed, such as Relief, Fisher score and information gain based.

In wrapper methods, machine learning algorithm is used as a black box [11]. These methods only require prediction accuracy to evaluate the relevance of features. They can be combined with any machine learning algorithm, since they do not depend on the special structure of the learning algorithm [16]. The most common approaches for the wrapper methods are recursive feature elimination, sequential feature selection and genetic algorithms.

In embedded methods, machine learning algorithm and feature selection parts can not be separated from each other, since it is decided, which feature to be selected, when the model is constructed. In other words, feature selection and machine learning tasks are performed simultaneously. Thus, the machine

*Corresponding author.
E-mail address: rkasimbeyli@eskisehir.edu.tr (Refail Kasimbeyli), oznuray26@gmail.com (Oznur Ay)
2020 Mathematics Subject Classification: 65K05, 90C25.
Accepted: September 18, 2024.

learning algorithm plays a significant role in embedded methods [16]. The embedded methods are usually developed by adding a sparsity term to the original problem with a trade-off parameter. For example, Least Absolute Shrinkage and Selection Operator (LASSO) [23] and 1-norm SVM [24] use l_1 norm as a sparsity term, while Bradley and Mangasarian use l_0 norm in feature selection concave (FSV) approach [5].

In this study, an embedded feature selection method, based on polyhedral conic functions (PCF) algorithm [9] (see also [1]), is proposed. This algorithm is based on the nonlinear seoaration theorem established by Kasimbeyli [13] (see also [14]).

Our goal is to achieve a more efficient classifier by adding a feature selection mechanism to the original PCF algorithm and to improve the generalization ability. The objective function of the PCF algorithm is extended with a sparsity term. This term is introduced as an approximation to zero norm of the normal vector of optimal separating surface. Additionally, two penalty parameters are added to balance the classification and the feature selection parts of the objective function. We also propose a modified version of the new FSV method, where the modified version of the PCF algorithm, is used. A comparison between the two approaches is presented. The successive linearization approach given in [5] is adapted to solve the new extended feature selection problems. Both versions of the proposed algorithm are tested on real world data sets. The obtained results are interpreted and a comparison between the performances of different methods, is presented.

The rest of the paper is organized as follows. In Section 2, the new embedded feature selection method is explained. In this section we also review the feature selection concave method (in subsection 2.1) and the polyhedral conic functions method (in subsection 2.2). In Section 3, computational results and interpretations related to the proposed method, are given. Finally, some conclusions of this study are discussed in Section 4.

2. Polyhedral Conic Functions (PCF) Based Embedded Feature Selection (FSV) Method

In this section, we introduce the embedded feature selection method and its modified version, by using the idea of the PCF based classification algorithm. This is an iterative method which sequentially generates polyhedral conic functions which separate some part of the given dataset from the other one, and simultaneously performs the feature selection. The algorithm finishes, when the complete set of PCFs are generated which separate the whole dataset, possibly with some allowed error tolerance.

The method combines the classification and the feature selection algorithms, and consists of an outer and an inner loops. The main focus of the work, is to combine the feature selection procedure and the PCF algorithm by extending the objective function of PCF algorithm, with a penalizing term. To achieve this, we examine the embedded feature selection methods and for this purpose, use the feature selection concave approach of [5], which is combined with the PCF algorithm. We adapt the sparsity term and the solution procedure that was used in FSV, to the new extended method.

For better understanding the new embedded method, we first briefly explain the original FSV algorithm, and the original PCF based classification algorithm, give their important properties, advantages and drawbacks, and explain how these features are taken into account in the new method.

2.1. Feature selection concave (FSV).

Feature selection concave approach is based on the robust linear programming (RLP) method given in [4]. The RLP approach serves to separate two (linearly non separable) disjoint finite point sets $A = \{a^i \in \mathbb{R}^n : i \in I\}$ and $B = \{b^j \in \mathbb{R}^n : j \in J\}$ in $\mathbb{R}^n$, where $I = \{1, ..., m\}$ and $J = \{1, ..., p\}$. This method aims to find an optimal separating surface in the form of a hyperplane $\{x : w^T x = \gamma\}$, by minimizing the number of miss-classified points. The problem which generates such an hyperplane, is formulated as follows:

$$\min_{w,\gamma,y,z} \quad \frac{e^T y}{m} + \frac{e^T z}{p}$$

subject to

$$\begin{aligned} -w^T a^i + \gamma + 1 \leq y_i, \forall i \in I, \\ w^T b^j - \gamma + 1 \leq z_j, \forall j \in J, \\ y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ w = (w_1, ..., w_n) \in \mathbb{R}^n, \gamma \in \mathbb{R}, \end{aligned} \tag{2.1}$$

where e is the vector of ones in an appropriate size. The feature selection is applied to the normal vector w of this hyperplane. For this, a penalizing term is added to the objective function of the above problem, which suppresses as many of the components of the normal vector w, as possible. Then, the corresponding components of w which are equal to zero, are removed. The penalizing term is defined as an approximation to l_0-norm (representing the number of non-zero elements of w), by a smooth function whose gradient can be computed and can be used to perform a gradient descent direction [16]. This approximation is described as a concave exponential function in [5]:

$$t(v, \alpha) = e - \mathbf{e}^{-\alpha v}, \alpha > 0,$$

where $v = (v_1, ..., v_n) \in \mathbb{R}^n$, $\mathbf{e}^{-\alpha v} = (\mathbf{e}^{-\alpha v_1}, \ldots, \mathbf{e}^{-\alpha v_n})$, $\mathbf{e}$ is the base of the natural logarithm and the parameter $\alpha > 0$ controls the steepness of the objective function [16]. Then the feature selection model is formulated in the following form:

$$\min_{w,\gamma,y,z,v} \quad (1-\lambda)\Big(\frac{e^T y}{m} + \frac{e^T z}{p}\Big) + \lambda e^T (e - \mathbf{e}^{-\alpha v})$$

subject to

$$\begin{aligned} -w^T a^i + \gamma + 1 \leq y_i, \forall i \in I, \\ w^T b^j - \gamma + 1 \leq z_j, \forall j \in J, \\ -v_t \leq w_t \leq v_t, t = 1, \ldots, n, \\ y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ w = (w_1, ..., w_n), v = (v_1, ..., v_n) \in \mathbb{R}^n, \gamma \in \mathbb{R}. \end{aligned} \tag{2.2}$$

The obtained mathematical program (2.2) is known as Feature Selection Concave (FSV), which minimizes the weighted sum of the classification error defined by the term $\left(\frac{e^T y}{m} + \frac{e^T z}{p}\right)$ and the number of non-zero elements of the normal vector w, defined by the term $e^T(e - \mathbf{e}^{-\alpha v})$. The parameter $\lambda \in [0, 1)$ balances these two objectives in the model and serves to maximize the generalization performance. For $\lambda = 0$, the model (2.2) becomes robust linear programming problem (2.1). The case $\lambda = 1$ does not lead to a significant result, because the components of the normal vector w are selected regardless to classification. Therefore, the parameter λ is searched over the range $[0, 1)$ using the cross-validation method until some meaningful solution (w, γ) is obtained.

Since the problem (2.2) is known to be NP-hard [16], the successive linear approximation algorithm proposed by [5] is used for solving this problem. The following section briefly explains this algorithm.

2.1.1. *Successive linear approximation (SLA) algorithm for solving FSV.* Choose some value for the balancing parameter $\lambda \in [0, 1)$, set $\alpha = 5$ (note that this value for α was suggested by [5]), and start with a random set of decision variables $(w^0, \gamma^0, y^0, z^0, v^0)$. Set $k = 0$ and determine $(w^{k+1}, \gamma^{k+1}, y^{k+1}, z^{k+1}, v^{k+1})$ by solving the linear program:

$$\min_{w,\gamma,y,z,v} \quad (1-\lambda)\Big(\frac{e^T y}{m} + \frac{e^T z}{p}\Big) + \lambda\alpha(\mathbf{e}^{-\alpha v^k})^T (v - v^k)$$

subject to

$$\begin{aligned} -w^T a^i + \gamma + 1 \leq y_i, \forall i \in I, \\ w^T b^j - \gamma + 1 \leq z_j, \forall j \in J, \\ -v_t \leq w_t \leq v_t, t = 1, \ldots, n \\ y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ w = (w_1, ..., w_n), v = (v_1, ..., v_n) \in \mathbb{R}^n, \gamma \in \mathbb{R}. \end{aligned} \tag{2.3}$$

Stop if,

$$(1-\lambda)\Big(\frac{e^T(y^{k+1}-y^k)}{m} + \frac{e^T(z^{k+1}-z^k)}{p}\Big) + \lambda\alpha(\mathbf{e}^{-\alpha v^k})^T(v^{k+1}-v^k) = 0. \tag{2.4}$$

Remark 2.1. It is shown in [5] [6, Theorem 2.2] that, the SLA algorithm terminates finitely at a stationary point which satisfies the minimum principle necessary optimality condition for problem (2.3) and leads to a sparse w with good generalization properties.

2.2. **Polyhedral conic functions (PCF) algorithm.** This subsection gives brief description on the background of the proposed method. The starting point for our investigation is the PCF algorithm developed in [9]. This is a supervised classification algorithm developed for separating two disjoint finite point sets in $\mathbb{R}^n$.

We begin by recalling the definition of the polyhedral conic functions.

Definition 2.2. [9, Definition 2.2.] A function $g : \mathbb{R}^n \to \mathbb{R}$ is called polyhedral conic, if its graph is a cone and for every $\alpha \in \mathbb{R}$, all its sublevel sets $S_\alpha = \{x \in \mathbb{R}^n : g(x) \leq \alpha\}$ are polyhedrons.

The polyhedral conic functions $g_{(w,\xi,\gamma,a)} : \mathbb{R}^n \to \mathbb{R}$ used in the PCF algorithm, are defined as follows:

$$g_{(w,\xi,\gamma,a)}(x) = w'(x-a) + \xi\|x-a\|_1 - \gamma, \tag{2.5}$$

where $w, a \in \mathbb{R}^n, \xi, \gamma \in \mathbb{R}, w'x = w_1x_1 + ... + w_nx_n$ is a scalar product of w and x, $\|x\|_1 = \mid x_1 \mid +...+ \mid x_n \mid$ is an l_1-norm of the vector $x \in \mathbb{R}^n$.

The graph of $g_{(w,\xi,\gamma,a)}(x)$, is a cone with vertex at $(a, -\gamma) \in \mathbb{R}^n \times \mathbb{R}$. The PCF algorithm aims to separate data points by using the sublevel sets of the polyhedral conic functions sequentially constructed at each iteration.

The PCF algorithm is explained below.

Let A and B be two given finite point sets in $\mathbb{R}^n$:

$$A = \{a^i \in \mathbb{R}^n : i \in I\}, B = \{b^j \in \mathbb{R}^n : j \in J\}, \;\; I = \{1, ..., m\}, J = \{1, ..., p\}.$$

PCF Algorithm.

Initialization Step: Let $l = 1, I_l = I, A_l = A, e \in \mathbb{R}^m$ be the vector of ones and go to Step 1.

Step 1: Let a^l be an arbitrary points of A_l. Solve subproblem (P_l):

$$(P_l) \;\; \min \Big(\frac{e^T y}{m}\Big)$$

subject to

$$w^T(a^i - a^l) + \xi\|a^i - a^l\|_1 - \gamma + 1 \leq y_i, \;\; \forall i \in I_l, \tag{2.6}$$

$$-w^T(b^j - a^l) + \xi\|b^j - a^l\|_1 + \gamma + 1 \leq 0, \;\; \forall j \in J, \tag{2.7}$$

$$y = (y_1, ..., y_m) \in \mathbb{R}^m_+, w \in \mathbb{R}^n, \xi \in \mathbb{R}_+, \gamma \geq 1. \tag{2.8}$$

Let $w^l, \xi^l, \gamma^l, y^l$ be a solution of (P_l). Let

$$g_l(x) = w^l(x - a^l) + \xi^l \|x - a^l\|_1 - \gamma^l \tag{2.9}$$

and go to Step 2.

Step 2: Let $I_{l+1} = \{i \in I_l : g_l(a^i) + 1 > 0\}$, $A_{l+1} = \{a^i \in A_l : i \in I_{l+1}\}$. If $A_l \neq \emptyset$ set $l = l + 1$ and go to Step 1. Otherwise go to Step 3.

Step 3: Define the function $g(x)$ as follows:

$$g(x) = \min_l \ g_l(x) \tag{2.10}$$

where L is the number of PCF's generated, and stop.

At each iteration l, the algorithm chooses an arbitrary element a^l from the set A_l and calculates the set (w^l, ξ^l, γ^l), by solving subproblem (P_l). Due to the constraints (2.6)-(2.8), the algorithm generates a polyhedral conic function g_l whose sublevel set contains as many points from the set A_l as possible, and contains no points from the set B (due to constraint (2.7)).

It is proved in [9, Theorem 2.3] that, the PCF algorithm terminates in finite number of iterations and the function g generated by the algorithm and defined in (2.10), separates the given two sets A and B in the following sense:

$$g(a) \leq 0, \ \ \forall a \in A, \tag{2.11}$$

$$g(b) > 0, \ \ \forall b \in B. \tag{2.12}$$

Remark 2.3. It is reported in different studies that the PCF algorithm gives competitive classification accuracies [3, 2, 7, 19]. The advantage of this algorithm is that, it uses polyhedral sets, constructed in an optimal way, as an intersection of 2^n halfspaces at every iteration. These halfspaces are generated by a single function called a polyhedral conic function. Because of this reason, the PCF algorithm guarantees efficient, easy and exact separation for arbitrary two finite point sets. The main drawback of the algorithm is the generalization property. The training accuracy of the PCF algorithm is always 100%, which is a result of the constraint (2.7), which prevents the entering of elements of B to a sublevel set of the separating PCF. Therefore the resulting classifier provides a 100% separation between the sets A and B which leads to overfitting, and results in undesired gap between the training and the test accuracies. To overcome this drawback, in the new algorithm, the right hand side of the constraint (2.7) is changed. Like the constraint (2.6), we put a nonnegative decision variable z_j for every $j \in J$ (instead of zero) at the right hand side, and add a new term to the objective function, which consists of sums of these variables. By this way, the algorithm will not prevent the entering of elements of B to a sublevel set of the separating PCF in optimal way. Moreover, this term has been assigned some "weight coefficient" so that the decision maker can control the effect of this term to the resulting "classification quality".

Remark 2.4. The vertex of the separating polyhedral conic set, generated as a sublevel set of the PCF at every iteration l of the PCF algorithm, is determined by a pair of elements a^l and γ^l. The element a^l is chosen randomly from the set A_l, and "optimal" value of the decision variable γ^l, is calculated by the algorithm. As it was pointed out in the relevant paper [9], the choice of element a^l may affect the number of separated elements, the total number of separating functions generated (and hence the total number of subproblems solved), and the classification accuracy. A version of PCF algorithm, called "The Modified PCF (M-PCF)", is proposed in the same study [9], whose main advantages can be summarized as follows:

- a more effective vertex selection,

- a lesser number of subproblems solved and a lesser number of PCFs generated, and
- a higher classification accuracy result.

In this study, M-PCF algorithm is also utilized with a simple preprocessing step, and a detailed comparison between the two versions of the proposed feature selection method, is provided.

The following section presents the PCF based feature selection method.

2.3. **The embedded feature selection method based on PCF (FS-PCF).** This section presents the embedded feature selection method and its modified version. The operation principle of the method can be briefly summarized as follows: it generates at each iteration a polyhedral conic function by using the solutions of subproblems (2.13)-(2.14) below. Then the data sets are updated using the same procedure as in PCF algorithm. In the new algorithm, the constraint related to the set B is changed by letting some elements of B to be contained in the sublevel set of PCF generated (in an optimal way). This situation increases the generalization performance of the algorithm.

Assume that A and B are two given sets in $\mathbb{R}^n$;

$$A = \{a^i \in \mathbb{R}^n : i \in I\},\ I = \{1, ..., m\}, B = \{b^j \in \mathbb{R}^n : j \in J\},\ J = \{1, ..., p\}.$$

Then the proposed algorithm operates as follows.

FS-PCF Algorithm
Initial Step: Let $l = 1, I_l = I, A_l = A, J_l = J, B_l = B$ and let e be a vector of ones in appropriate size. Select the values of the parameters L and C from the set $P = \{10^i : i = -4, -3, ..., 0, ..., 3, 4\}$, and go to Step 1.
Step 1: Select an arbitrary point c from A_l and set $k = 1$. Solve the following subproblem (P_l):

$$(P_l)\ \ \min\ \frac{e^T y}{m} + \frac{e^T z}{p}$$

subject to

$$\begin{aligned} w^T(a^i - c) + \xi\|a^i - c\|_1 - \gamma + 1 \le y_i,\ \ \forall i \in I_l, \\ -w^T(b^j - c) - \xi\|b^j - c\|_1 + \gamma + 1 \le z_j,\ \ \forall j \in J_l, \\ y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ w = (w_1, ..., w_n) \in \mathbb{R}^n, \xi \in [0, \infty), \gamma \in [1, \infty). \end{aligned} \tag{2.13}$$

Let (w, ξ, γ, y, z) be an optimal solution of (P_l). Assign $(w^{lk}, \xi^{lk}, \gamma^{lk}, y^{lk}, z^{lk}, a^l) = (w, \xi, \gamma, y, z, c)$ and determine the vector $v^{lk} = (v_1^{lk}, ..., v_n^{lk}) \in \mathbb{R}^n$ randomly.

Step 2: Solve the subproblem (P_{lk}) given below:

$$(P_{lk})\ \ \min\ \frac{e^T y}{m} + L\Big(\frac{e^T z}{p}\Big) + C\alpha(\mathbf{e}^{-\alpha v^{lk}})(v - v^{lk})$$

subject to

$$\begin{aligned} &w^T(a^i - a^l) + \xi\|a^i - a^l\|_1 - \gamma + 1 \leq y_i, \ \ \forall i \in I_l, \\ &-w^T(b^j - a^l) - \xi\|b^j - a^l\|_1 + \gamma + 1 \leq z_j, \ \ \forall j \in J_l, \\ &-v_t \leq w_t \leq v_t, t = 1, \ldots, n, \\ &y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ &w = (w_1, ..., w_n), v = (v_1, ..., v_n) \in \mathbb{R}^n, \xi \in [0, \infty), \gamma \in [1, \infty). \end{aligned} \tag{2.14}$$

Let $(w^{l,k+1}, \xi^{l,k+1}, \gamma^{l,k+1}, y^{l,k+1}, z^{l,k+1}, v^{l,k+1})$ be an optimal solution of subproblem (2.14). If

$$\frac{e^T(y^{l,k+1} - y^{lk})}{m} + L\Big(\frac{e^T(z^{l,k+1} - z^{lk})}{p}\Big) + C\alpha(\mathbf{e}^{-\alpha v^{lk}})(v^{l,k+1} - v^{lk}) = 0 \tag{2.15}$$

let

$$g_l(x) = w^{l,k+1}(x - a^l) + \xi^{l,k+1}\|x - a^l\|_1 - \gamma^{l,k+1} \tag{2.16}$$

and go to Step 3, otherwise assign $k = k + 1$ and repeat Step 2.
Step 3: Update the sets A_l and B_l as follows:

$$\begin{aligned} &I_{l+1} = \{i \in I_l : g_l(a^i) + 1 > 0\}, A_{l+1} = \{a^i \in A_l : i \in I_{l+1}\}, \\ &J_{l+1} = \{j \in J_l : g_l(b^j) + 1 > 0\}, B_{l+1} = \{b^j \in B_l : j \in J_{l+1}\}. \end{aligned}$$

Set $l = l + 1$. If $A_l \neq \emptyset$ go to Step 1, otherwise go to Step 4.
Step 4: Define the function $g(x)$ as follows:

$$g(x) = \min_l g_l(x) \tag{2.17}$$

and stop.

2.4. **Discussion on the FS-PCF algorithm.** Steps 1-3 of the FS-PCF algorithm, can be considered as an outer loop, where the sets A_l and B_l are updated and the new initial solution and the resulting polyhedral conic classifier for the new updated sets (of iteration l) is calculated. The algorithm terminates if the current set A_l becomes empty. By [9, Theorem 2.3], the number of iterations required for termination, is finite. Step 2 of the FS-PCF algorithm can be considered as an inner loop, where the optimal classifier, that is function g_l is generated by updating the index k and iteratively solving the subproblem (P_{lk}) (see (2.14)) till the stop criteria (2.15) is satisfied. The subproblem (2.14) and the stop criteria (2.15) are very similar (almost the same) to the problem (2.3) and the stop criteria (2.4), respectively. Hence, based on the analysis given in Remark 2.1, we can conclude that, the sequence of solutions $\{(w^{l,k}, \xi^{l,k}, \gamma^{l,k}, y^{l,k}, z^{l,k}, v^{l,k})\}$, $k = 1, 2, ...$, generated in this step (for every l), converges (even the number of iterations required for satisfying the stop criteria, is finite), and leads to the desired solution for a sparse w with good generalization property.

Any point of set A_l, misclassified in a current iteration, will be correctly classified in some other iteration. In other words, the point remains outside of the sublevel set of the function generated according to the selected vertex point a^l in some iteration, will fall into the sublevel set of a function generated in one of further iterations. Therefore, in the objective function of the problem P_{lk}, the error term related to the set A_l, is not penalized.

We now describe another version of the proposed feature selection method, using the so-called **modified PCF approach (FS-MPCF)** with the following preprocessing step.
Vertex Evaluation Procedure: Solve the subproblem (P_l) given below for every $l \in I = \{1, ..., m\}$ and every $a^l \in A$.

$$(P_l) \quad \min \ \frac{e^T y}{m} + \frac{e^T z}{p} \tag{2.18}$$

subject to

$$\begin{aligned} w^T(a^i - a^l) + \xi\|a^i - a^l\|_1 - \gamma + 1 \leq y_i, \forall i \in I, \\ -w^T(b^j - a^l) - \xi\|b^j - a^l\|_1 + \gamma + 1 \leq z_j, \forall j \in J, \\ y = (y_1, ..., y_m) \in \mathbb{R}^m_+, z = (z_1, ..., z_p) \in \mathbb{R}^p_+, \\ w = (w_1, ..., w_n) \in \mathbb{R}^n, \xi \in [0, \infty), \gamma \in [1, \infty). \end{aligned} \tag{2.19}$$

Let $(w^l, \xi^l, \gamma^l, y^l, z^l)$ be an optimal solution of (P_l), corresponding to the element $a^l \in A_l$, and let N_l be the number of elements of set A_l, which fall into the sublevel set of the polyhedral conic function $g_l(x) = w^l(x - a^l) + \xi^l\|x - a^l\|_1 - \gamma^l$ generated for this optimal solution. Sort all the elements a^l of set A in descending order with respect to the numbers N_l. After this preprocessing step, the element of A_l which corresponds to the maximal number N_l over all elements of A_l, will be selected as a point c in the initial step of FS-PCF algorithm. By this way, in the initial solution, the element c will not be selected arbitrarily, but the element with maximum performance, will be selected.

3. Computational Results and Comparisons

In this section, the FS-PCF Algorithm is tested on real world data. For this purpose, some currently available moderate sized and relatively large scale data sets from the UCI [8] machine learning repository, are used. The moderate sized data sets are used to compare the results obtained by FS-PCF and FS-MPCF, with the results obtained using RLP and FSV. The relatively large scale data sets are used to compare the proposed algorithm with other classifiers. The moderate sized data sets are the BUPA Liver Disorders (BUPA Liver), the Wisconsin Breast Cancer Diagnosis (WBCD), the Wisconsin Breast Cancer Prognosis (WBCP), the Ionosphere, the Cleveland Heart Disease (Cleveland Heart), the Pima Indians Diabetes (Pima Diabetes). Stratified tenfold cross-validation is applied to these data sets, and the relatively large scale data sets are divided into train and test parts using the values given in Table 1.

We chose the values of the weighting parameters L and C from set $P = \{10^i : i = -4, -3, ..., 0, ..., 3, 4\}$, which led to the best predictive accuracy on each data set by using grid search technique [12]. We started the parameter selection with the set $P' = \{10^i : i = -7, -6, ..., 0, ..., 6, 7\}$. Since no meaningful results were obtained in the two extremes of this set, we decided to narrow the range. Besides, we set the parameter $\alpha = 5$ as it is suggested by [5].

RLP, FSV, FS-PCF and FS-MPCF methods are implemented by using [10] tool and Python programming language [21]. All the reported results of these methods are performed on a computer with processor 3.5 GHz and 16 GB of RAM. In addition, stratified tenfold cross-validation technique is utilized from the scikit-learn machine learning package [20].

The results for test and train accuracies, obtained by these methods are presented in Table 2. Bold numbers refer to the best predictive accuracy of each row. As can be seen from the results presented in Table 2, the FS-PCF and FS-MPCF algorithms achieve the best test accuracy in four of the six used data sets. Their results are also acceptable in comparison with other algorithms for all data sets. Both versions of the proposed algorithms overcome the problem of overfitting, when we compare with the results obtained using PCF algorithm [9].

Test set accuracy results on relatively large scale data sets using different classification algorithms are presented in Table 3. The results of all classifiers except the FS-PCF are gathered from the article [18]. We tried to choose different types of machine learning algorithms such as: a probability based classifier

TABLE 1. Description of data sets

Data sets	Train/test	No. of features	No. of classes	No. of samples
BUPA liver	Tenfold	6	2	345
WBCD	Tenfold	9	2	683
WBCP	Tenfold	32	2	194
Ionosphere	Tenfold	34	2	351
Cleveland heart	Tenfold	13	2	297
Pima diabetes	Tenfold	8	2	768
Spambase (SB)	3682/919	57	2	4601
Landsat satellite image (LSI)	4435/2000	36	6	6435
DNA	2000/1186	180	3	3186
Isolet (ISO)	6238/1559	617	26	7797
Optical recognition of handwritten digits (HD)	3823/1797	64	10	5620

TABLE 2. Tenfold cross-validation results obtained using the proposed feature selection algorithms and the other algorithms (average training accuracy [%], average test accuracy [%]).

	RLP		FSV		FS-MPCF		FS-PCF	
Data sets	Train	Test	Train	Test	Train	Test	Train	Test
1 BUPA Liver	68.5	66.4 $\pm$ 5.61	66.6	64.6 $\pm$ 5.29	81.2	**69.0** $\pm$ 6.73	81.2	68.4 $\pm$ 4
2 WBCD	97.5	97.2 $\pm$ 1.90	97.2	96.2 $\pm$ 2.68	100	**97.5** $\pm$ 4.11	99.7	97.1 $\pm$ 2
3 WBCP	89.4	77.5 $\pm$ 8.91	69.4	67.9 $\pm$ 8.99	89.5	**81.1** $\pm$ 5.25	85.2	80.9 $\pm$ 4
4 Ionosphere	94.8	87.3 $\pm$ 7.28	86.5	83.5 $\pm$ 4.50	93.0	92.6 $\pm$ 6.36	93.3	**92.9** $\pm$ 6
5 Cleveland Heart	85.1	**82.5** $\pm$ 5.95	83.2	80.2 $\pm$ 6.36	92.4	80.7 $\pm$ 6.11	91.8	80.8 $\pm$ 5
6 Pima Diabetes	76.6	75.5 $\pm$ 5.13	76.5	**75.7** $\pm$ 3.56	80.8	73.4 $\pm$ 4.96	81.4	74.2 $\pm$ 4

Naive Bayes (with kernel) (NB), a rule based classifier PART, a support vector machine classifier Linear LibSVM, a decision tree classifier J48 and Logistic Regression (Logistic). Weighting parameters L and C are determined on relatively large scale data sets for the FS-PCF algorithm by applying the grid search technique in the same set P with moderate sized data sets. The results of other classifiers are reported applying WEKA with the default parameter values in the relevant study [18].

Based on the results presented in Table 3, we can conclude that, the FS-PCF algorithm achieved the best predictive accuracy on Landsat Satellite Image data set than all other classifiers compared. Moreover, the test accuracies of the FS-PCF algorithm on Spambase and DNA data sets are close to the results of classifiers that achieved the best accuracy.

TABLE 3. Test set accuracies obtained using different classifiers for large scale data sets.

Data sets Classifiers	SB	LSI	DNA	ISO	OD
NB (kernel)	76.17	82.10	93.34	84.22	90.32
PART	91.40	85.25	91.06	82.81	89.54
LibSVM (LIN)	90.97	85.05	93.09	96.02	96.55
J48	92.93	85.35	92.50	83.45	85.75
Logistic	92.06	83.75	88.36	-	92.21
FS-PCF	92.06	90.50	93.17	90.19	92.49

3.1. **Finiteness of the number of iterations, and the number of selected features.** This subsection presents the computational results obtained on 6 moderate sized data sets, which are related to the number of iterations performed in Step 2 of the FS-PCF algorithm, and the average number of features selected during these iterations. The convergence of the Successive Linear Approximation Algorithm used in Step 2 of the FS-PCF algorithm, has been emphasized in section 2.4. The computational results showed that, Step 2 terminates in 2 to 5 iterations for all test problems used. Table 4 summarizes the average number of iterations performed in Step 2, and the average number original problem features selected by the classifiers generated by the algorithm. All numbers presented are average numbers over $10-$folds. It is remarkable that all the features were suppressed by the algorithm for the "Ionosphere" data set.

We have also collected the obtained results for every-fold and every PCF generated (for every iteration of the outer loop), related to WBCP data set in Table 5.

TABLE 4. Average number of iterations in the inner loop of the FS-PCF algorithm and the average number of selected features.

Data sets	Average number of iterations performed in the inner loop	Average number of features selected by the classifier
1 BUPA Liver (6)	2.01	5.97
2 WBCD (9)	4.05	0.75
3 WBCP (32)	4.31	13.10
4 Ionosphere (34)	2.00	0.00
5 Cleveland Heart (13)	2.08	12.66
6 Pima Diabetes (8)	2.81	7.84

3.2. **Detailed comparison between two versions of the proposed method.** It is reported in [9] that, on some data sets, M-PCF algorithm gives better prediction accuracy than the PCF algorithm. However, as it can be seen in Tables 4 and 5, no significant improvement is observed applying this algorithm, except for two data sets. Note that the use of M-PCF algorithm may become more expensive especially in the cases, if the data set under consideration contains a huge number of samples. Because the implementation of this algorithm requires to perform the vertex evaluation procedure, by solving the problem (2.18)-(2.19) for every sample of the set, in order to determine the element a_l with maximum performance. Instead, the PCF algorithm chooses this element randomly. Since, the modified procedure requires dramatically more training time without any guarantee on the improvement of other performance metrics, we can conclude that it may be not reasonable to apply the modified procedure for data sets with huge number of samples.

TABLE 5. The number of iterations performed in the inner loop and the number of selected features for every fold obtained on WBCP data set.

	1. PCF		2. PCF		3. PCF		4. PCF		5. PCF		6. PCF	
Folds	Iter	Features	Iter	Features	Iter	Features	Iter	Features	Iter	Features	Iter	Features
1. Fold	4	17	4	17	4	17	6	9	3	5	-	-
2. Fold	4	15	4	15	4	17	6	16	2	0	-	-
3. Fold	4	19	4	16	5	18	3	14	4	2	-	-
4. Fold	5	13	4	14	4	17	7	15	5	8	3	4
5. Fold	5	16	6	17	6	19	4	17	4	6	3	3
6. Fold	3	17	4	16	4	16	3	14	4	4	2	0
7. Fold	5	15	4	20	4	15	8	14	3	4	-	-
8. Fold	7	19	4	17	4	17	8	9	3	3	-	-
9. Fold	4	17	3	16	4	18	5	15	2	1	-	-
10. Fold	4	19	3	18	4	20	4	17	8	4	-	-

TABLE 6. Average tenfold training accuracy [%], average tenfold test accuracy [%]

	FS-MPCF		FS-PCF	
Data sets	Training accuracy	Test accuracy	Training accuracy	Test accuracy
1 BUPA Liver	81.19	69.00	81.19	68.40
2 WBCD	100.00	97.52	99.70	97.09
3 WBCP	89.46	81.12	85.23	80.93
4 Ionosphere	93.00	92.58	93.32	92.85
5 Cleveland Heart	92.39	80.74	91.81	80.76
6 Pima Diabetes	80.84	73.44	81.40	74.20

TABLE 7. Training times and average number of PCF generated

	FS-MPCF		FS-PCF	
Data sets	Training time (sec)	Average number of PCFs generated	Training time (sec)	Average number of PCFs generated
1 BUPA Liver	22.41	111	6.00	154
2 WBCD	109.41	40	4.68	39
3 WBCP	32.73	6	3.16	8
4 Ionosphere	109.47	200	34.73	201
5 Cleveland Heart	22.04	34	3.67	61
6 Pima Diabetes	146.73	191	75.19	338

4. CONCLUSION

The paper presents a new feature selection method which is based on the combination of polyhedral conic functions algorithm with feature selection concave approach. The new method uses polyhedral conic surfaces, instead of hyperplanes used in FSV approach. The modified PCF algorithm is also adapted to the proposed method with a simple preliminary step. It is shown that the new method allows to diminish the problem of overfitting, which is characteristic for the PCF algorithm, and increases the generalization performance. Classifiers obtained by the new methods exhibit feature suppression and

have better test accuracies on four from six moderate sized test problems, and on one from six relatively large scale data sets.

Future work may include further analysis of the benefits of separation, using polyhedral conic surfaces.

Statements and Declarations

The authors declare that they have no conflict of interest, and the manuscript has no associated data.

References

[1] M. Acar and R. Kasimbeyli. A polyhedral conic functions based classification method for noisy data. *Journal of Industrial and Management Optimization*, 17:3493–3508, 2021.

[2] A. M. Bagirov, R. Kasimbeyli, G. Öztürk, and J. Ugon. Piecewise linear classifiers based on nonsmooth optimization approaches. In T. Rassias, C. Floudas, and S. Butenko, editors, *Optimization in Science and Engineering*, pages 1–32. Springer, New York, 2014.

[3] A. M. Bagirov, J. Ugon, D. Webb, G. Ozturk, and R. Kasimbeyli. A novel piecewise linear classifier based on polyhedral conic and max-min separabilities. *TOP*, 21(1):3–24, 2013.

[4] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1(1):23–34, 1992.

[5] P. S. Bradley and O. L. Mangasarian. Feature selection via concave minimization and support vector machines. In J. W. Shavlik, editor, *International Conference on Machine Learning*, volume 98 of *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 82–90, San Francisco, 1998. Morgan Kaufmann Publishers Inc, United States.

[6] P. S. Bradley, O. L. Mangasarian, and W. Street. Feature selection via mathematical programming. *INFORMS Journal on Computing*, 10(2):209–217, 1998.

[7] H. Cevikalp and B. Triggs. Polyhedral conic classifiers for visual object detection and classification. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition*, pages 261–269, Honolulu, 2017. Institute of Electrical and Electronics Engineers, United States.

[8] D. Dheeru and E. Karra Taniskidou. UCI machine learning repository, 2017.

[9] R. N. Gasimov and G. Ozturk. Separation via polyhedral conic functions. *Optimization Methods and Software*, 21(4):527–540, 2006.

[10] I. Gurobi. Optimization: Gurobi optimizer reference manual (2018). *URL http://www.gurobi.com*, 2016.

[11] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[12] C.-W. Hsu, C.-C. Chang, and C.-J. Lin. A practical guide to support vector classification. *Department of Computer Science, National Taiwan University*, pages 1–16, 2003.

[13] R. Kasimbeyli. A nonlinear cone separation theorem and scalarization in nonconvex vector optimization. *SIAM Journal on Optimization*, 20(3):1591–1619, 2010.

[14] R. Kasimbeyli and M. Karimi. Separation theorems for nonconvex sets and application in optimization. *Operations Research Letters*, 47(6):569–573, 2019.

[15] V. Kumar and S. Minz. Feature selection. *SmartCR*, 4(3):211–229, 2014.

[16] T. N. Lal, O. Chapelle, J. Weston, and A. Elisseeff. Embedded methods. In I. Guyon, M. Nikravesh, and L. Gunn, S.and Zadeh, editors, *Feature Extraction: Foundations and Applications*, volume 207, pages 137–165. Springer, Berlin, Heidelberg, 2006.

[17] J. Neumann, C. Schnörr, and G. Steidl. Combined SVM-based feature selection and classification. *Machine Learning*, 61:129–150, 2005.

[18] G. Ozturk, A. M. Bagirov, and R. Kasimbeyli. An incremental piecewise linear classifier based on polyhedral conic separation. *Machine Learning*, 101(1-3):397–413, 2015.

[19] G. Ozturk and M. T. Ciftci. Clustering based polyhedral conic functions algorithm in classification. *Journal of Industrial & Management Optimization*, 11(3):921–932, 2015.

[20] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[21] G. Rossum. Python reference manual. Technical report, Amsterdam, Netherlands, 1995.

[22] J. Tang, S. Alelyani, and H. Liu. Feature selection for classification: A review. *Data Classification: Algorithms and Applications*, page 37, 2014.

[23] R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, 58(1):267–288, 1996.

[24] J. Zhu, S. Rosset, R. Tibshirani, and T. J. Hastie. 1-norm support vector machines. In S. Thrun, L. K. Saul, and B. Schölkopf, editors, *Proceedings of the 16th International Conference on Neural Information Processing Systems*, volume 16, pages 49–56, Whistler British Columbia Canada, 2003. MIT Press, Cambridge, United States.