

## A random subspace based conic functions ensemble classifier

Emre ÇİMEN\* 

Computational Intelligence and Optimization Laboratory (CIOL),  
Department of Industrial Engineering, Eskişehir Technical University, Eskişehir, Turkey

Received: 14.11.2019

Accepted/Published Online: 11.04.2020

Final Version: 29.07.2020

**Abstract:** Classifiers overfit when the data dimensionality ratio to the number of samples is high in a dataset. This problem makes a classification model unreliable. When the overfitting problem occurs, one can achieve high accuracy in the training; however, test accuracy occurs significantly less than training accuracy. The random subspace method is a practical approach to overcome the overfitting problem. In random subspace methods, the classification algorithm selects a random subset of the features and trains a classifier function trained with the selected features. The classification algorithm repeats the process multiple times, and eventually obtains an ensemble of classifier functions. Conic functions based classifiers achieve high performance in the literature; however, these classifiers cannot overcome the overfitting problem when it is the case data dimensionality ratio to the number of samples is high. The proposed method fills the gap in the conic functions classifiers related literature. In this study, we combine the random subspace method and a novel conic function based classifier algorithm. We present the computational results by comparing the new approach with a wide range of models in the literature. The proposed method achieves better results than the previous implementations of conic function based classifiers and can compete with the other well-known methods.

**Key words:** Random subspace, classification, linear programming, ensemble learning, conic functions

### 1. Introduction

Classification is a task that aims to obtain a model priorly by training a function with observed data and then apply the learned function to unseen test data in order to label test data points to the correct category. In most of the cases, algorithms require to solve an optimization problem to get the classification function. The number of features in the dataset is always a subject that a researcher should consider while selecting the right algorithm for an application. When the training data set has  $m$  data points from  $\mathbb{R}^n$  and if  $m \gg n$ , the data space becomes denser. Under this situation, a large scale optimization algorithm becomes suitable to use. In such a case, one can use data elimination methods for irrelevant data points or can divide the training data into smaller subsets to build smaller optimization problems. Generally, it is a good case to have sufficient valuable information to construct an accurate classifier.

There are many real-life applications of which data is hardy to collect, and the data feature size is large. For example, in biotechnology-related problems, collecting many samples can be laborious, expensive, or impossible. On the other hand, feature size (e.g., DNA, RNA, gene features, etc.) is generally high dimensional. Machine learning helps researchers in disease diagnosis (cancer, heart diseases, etc.), agriculture [1], drug development, and viral infection (SARS-CoV, COVID-19) studies. Since these researches try to bring solutions, hunger, pandemics, and other life-threatening diseases, they are essential to be targeted. Another machine

\*Correspondence: [ecimen@eskisehir.edu.tr](mailto:ecimen@eskisehir.edu.tr)

learning problem faces with high dimensionality is image processing. Although it is easier to collect samples in image processing, we may need to have thousands of features to represent an image sample. In text processing studies also the dimensionality is very high related to sample size. Even though there are efficient methods to represent document feature space, still high dimensionality remains as a problem. In this case, data space becomes sparse, and learning algorithms suffer from this sparsity. As a result of the *curse of dimensionality*, classifiers overfit and cannot reach the same test accuracy with training accuracy. In other words, classification error becomes high.

There are 3 components of classification error: bias, variance, and noise. Every classification model has some assumptions. For example, a linear classifier assumes the border between classes is linear. However, it may not be a real case; thus, this situation causes some error. Even we have sufficient data, the classifier will produce incorrect decisions in some parts of the data space. Namely, the classification model will have a bias. The models have strong assumptions about the problem have a high bias. On the other hand, random selections of the training data cause to have different models after training. However, since there is only one the real boundary, we would expect to have similar models at the end of the training. When the data size is not sufficient, dimensionality is high, and the classification model is complex then it should be; different trained models will fit the training data perfectly, but these models will make inconsistent predictions for a text sample. Namely, the classification model has a high variance. overfitting and model variance is closely related. There is a trade-off between bias and variance, and it is affected by every design choice of the model. Aggarwal [2] provides a more detailed explanation about classification errors.

As stated above, high dimensionality causes overfitting and high model variance. To overcome the overfitting problem, frequently, data is tried to be represented in a less dimensional space through dimensionality reduction methods. Besides dimensionality reduction approaches, another way to reduce bias and variance related errors is to use ensemble learning methods. Ensemble learning approaches increase the prediction accuracy by combining the results from multiple classifiers [2]. There are various ensemble learning approaches, and random subspace-based ensemble learning is one of them. Details about dimensionality reduction and ensemble learning methods are given in the following section.

Any classification method can be chosen as the base algorithm of an ensemble learning approach. Polyhedral conic functions (PCF) based classification algorithms are robust and accurate classifiers in the literature. However, when the feature size to sample size ratio is high, PCF based classification algorithms overfit to training data and perform poorly on the test dataset. Thus, the overfitting problem becomes a barrier to apply PCF based classifiers to real-life problems. The motivation of this study is to develop a method that overcomes the overfitting problem and maximizes the ability of generalization of PCF based classifiers. Therefore, in this study we develop a classification algorithm that combines the random subspace ensemble learning method and PCF (or CF) based classifiers. No study aims to solve overfitting problem of PCFs yet. The proposed method helps to fill this gap in the literature. The first novelty of the study is the presented algorithm called random subspace based conic functions ensemble classifier (RS-CF). The second novelty is the used regularization method and it has not been applied conic function based classifiers before. We provide the efficiency of the method on a wide range of problems such as image, arrhythmia, voice, text, and disability classification. This study provides a new method to researchers who use PCFs (or CFs) for classification and have relatively less sample size related to feature size.

Organization of the rest of the paper is as follows. We present existing work about the dimensionality reduction methods, ensemble learning methods, and CF-based classifiers in Subsection 1.1. In Section 2, we

explain the theoretical background of random subspace method and CF-based classifiers. We present the proposed random subspace ensemble classifier based on conic functions in Section 3. In Section 4, we give experimental results in comparison with the well-known algorithms in the literature. Finally, we provide conclusions. One can reach to notations from Table 1.

**Table 1.** Notation table.

Notation	Meaning
Capital boldface letters ( <b>A</b> )	2-d matrices (datasets)
Lower case boldface letters ( <b>v</b> )	Vectors
Lower case italic letters ( <i>s</i> )	Scalars
$\ \cdot\ _p$	$p$ -norm of a vector
$\langle \cdot, \cdot \rangle$	Dot product
$ \mathbf{A} $	Cardinality of a set <b>A</b>

### 1.1. Existing work

In this section, we discuss current work and provide a literature review about the methods that help to prevent overfitting, such as dimensionality reduction methods and ensemble learning methods. Additionally, we give applications and existing algorithms of the PCF-based classifiers used in this study.

Dimensionality reduction aims to represent data samples within a less dimensional space. The objective is to minimize inevitable information loss. We can group dimensionality reduction methods into 3 categories: feature selection, dimensionality reduction with axis rotation, and dimensionality reduction with type transformation [2].

Feature selection methods aim to select the best subset of features. A good feature selection algorithm minimizes the relevant information loss while representing the input data with fewer features. Filter methods [3, 4], wrapper methods [5, 6] and embedded methods [7] are various groups of feature selection algorithms. While filter algorithms use ranking techniques to order the features, wrapper methods solve an optimization model by using classification accuracy as an objective function. Chandrashekar et al. provide a survey on feature selection methods. [8].

Dimensionality reduction with axis rotation techniques transforms input space into another space, which is more significant and more discriminative for classification. This transformation can be supervised or unsupervised. One of the most well-known unsupervised dimensionality reduction algorithm, principal component analysis (PCA) [9], uses an orthogonal transformation to transform the input data linearly. The resulting features are the linear combinations of the original features. Singular value decomposition (SVD) is another linear transformation method [10]. It is closely related to PCA and also more general than PCA. SVD provides the same basis vectors and data transformation as PCA for data sets in which the mean of each attribute is 0 [2]. An application of SVD to text-domain is latent semantic analysis (LSA) [11]. LSA assumes that words with similar meanings occur in similar pieces of text. On the other hand, linear discriminant analysis (LDA, sometimes called Fisher's linear discriminant) is a supervised linear dimensionality reduction method, thus takes class labels into account [12]. Transformation can be nonlinear too. Autoencoders [13] are an efficient neural network-based unsupervised tools to make nonlinear transformations. Khalid et al. presented a detailed survey of dimensionality reduction methods in [14].

One can reduce dimensionality with type transformation as well. In such a case, the data represented with a more complex type is transformed into a less complex one, e.g., time series to multidimensional. Discrete wavelet transformation methods are in this group.

As dimensionality reduction, ensemble learning methods also aim to reduce overfitting and improve test performance of the classifiers. The well-known methods are bagging, random subspace methods, random forest, and boosting. Bagging (aka bootstrap aggregating) creates different training sets with independently drawn bootstrapped samples, and trains a classifier on each of them. Then, the final decision is made via a majority voting (classification) or averaging (regression). Bagging methods aim to reduce variance. Random subspace methods [15] (aka feature bagging) are ensemble learning approaches that use a random subset of the entire feature set for each learner. Random subspace methods are adaptable to any classification approaches such as support vector machines, nearest neighbors, and decision trees. A particular version of decision trees, which is a version of the random subspace related method and decision trees, is a very well-known random forest algorithm [16, 17]. The generalization error of random forest classifiers depends on the strength of the individual trees in the forest [17]. Boosting weights each training sample, and then trains different classifiers with the use of these sample weights. Based on classifier performance, the boosting method updates sample weights. More precisely, while weights of the incorrectly classified samples are increased, weights of the correctly classified samples are reduced. Thus, the selected base classification method gives more attention to incorrectly classified samples. Finally, the boosting method creates an ensemble of classifiers. One of the most well-known boosting methods is AdaBoost (adaptive boosting)[18].

Besides dimensionality reduction and ensemble learning methods, there are several recent studies in the literature to prevent the overfitting problem with various points of view. Xiong et al. use a vibrating mechanism to prevent overfitting [19]. Their method is a continuous version of dropout and can achieve a certain  $L_1$  regularization. Authors show that their approach achieves better classification accuracy without using  $L_1$  regularization or dropout. Werpachowski et al. detect overfitting via adversarial examples, and their model correctly indicates the overfitting of the trained model to the training set [20]. Feldman et al. extend developments in understanding overfitting due to adaptive data analysis to multiclass prediction problems [21]. They also claim that multiclass prediction problems are significantly more robust to overfitting when reusing a test (or holdout) dataset. Wu et al. present a novel regularizer to reduce kernel redundancy in a deep CNN model and prevent overfitting [22]. Salman et al. proposed a consensus-based classification algorithm that enabled them to avoid overfitting and significantly improve classification accuracy, especially when the number of training samples is limited [23].

The classifier used in this study is polyhedral conic functions (PCFs) that are presented by Gasimov and Ozturk [24]. Since then, various classification algorithms developed by using PCFs [25–40]. Bagirov et al. developed a piece-wise linear classifier based on polyhedral conic and max-min separabilities [26]. Oztruk et al. presented an incremental version of a piece-wise linear classifier based on polyhedral conic separation [27]. Oztruk and Ciftci used clustering as a preprocessing and built nonlinear decision boundaries with PCFs [28]. Cimen et al. presented incremental conic functions (ICF) algorithm for large scale classification problems [34, 35]. ICF algorithm provides a generalization of PCFs and applies smart steps to boost training speed in large scale problems. Most of the PCF based classification algorithms require to solve linear programming (LP) models, and that makes PCF based classifiers possible to run on larger data sets. Due to their short test time, one can use PCF based classifiers on real-time problems such as gesture recognition [31]. Furthermore, success

of PCF based classifiers has shown on arrhythmia classification [29], object detection [30] and text classification [33].

## 2. Theoretical background

In this section, we explain theoretical background about the random subspace method and conic functions to make the proposed method more understandable and clear. The random subspace method and conic functions are two main nested parts in the proposed method.

### 2.1. Random subspace method

The random subspace method is introduced by Ho to improve the generalization accuracy of complex decision forests [15]. The method is an ensemble construction technique and iterates multiple loops. In each iteration, selects a small number of features from the input feature space and then train a classifier with the selected subset of the features. In contrast to the sample bagging (bootstrap aggregating), features are bagged in the random subspace method. Essentially, this process yields a bunch of classifiers by training them with various projections of input space to the other lower-dimensional spaces. Ensemble construction with a random subspace method is given in Algorithm 1. The training set  $\mathbf{T}$  is given in Equation 1.  $y^f$  is the class label of  $\mathbf{t}^f$  where  $\nu$  is the number of classes in Equation 2.

$$\mathbf{T} = \{\mathbf{t}^1, \dots, \mathbf{t}^e\}, \quad \mathbf{t}^f \in \mathbb{R}^n, \quad f \in F = \{1, \dots, e\} \quad (1)$$

$$\mathbf{y} = \{y^1, \dots, y^e\}, \quad y^f \in \{1, \dots, \nu\}, \quad f \in F = \{1, \dots, e\} \quad (2)$$

---

#### Algorithm 1 Random subspace method.

---

- 0: **Input:** Training set  $\mathbf{T} \in \mathbb{R}^{e \times n}$ , number of classifiers to be trained is  $m$ , dimension of subspaces is  $\tilde{n}$ .
  - Output:** Ensemble of classifiers  $G = \{g_1, \dots, g_m\}$ .
  - 1: **for**  $i = 1$  to  $m$ :
  - 2: Select a random subset of  $1, \dots, n$  with  $\tilde{n}$  elements.
  - 3: Obtain a subset  $\tilde{\mathbf{T}}_i$  of training set  $\mathbf{T}$ , where  $\tilde{\mathbf{T}}_i \in \mathbb{R}^{e \times \tilde{n}}$ .
  - 4: Train classifier  $g_i$  with the set  $\tilde{\mathbf{T}}_i$ .
  - 5: **end for**
- 

Once the ensemble of classifiers are obtained, a data point  $\mathbf{x} \in \mathbb{R}^n$  can be classified via calculating the average class probabilities or majority voting. The variable  $q_{iy} \in \{0, 1\}$ , equals 1 if the classifier  $i$  assigns the data point to the  $y^{th}$  class where  $y \in \{1, \dots, \nu\}$ . The probability of the data point  $\mathbf{x}$  to be predicted in the class  $y$  represented as  $pr_y$  and can be calculated as in Equation 3. Predicted class label can be found as in Equation 4.

$$pr_y = \frac{\sum_{i=1}^m q_{iy}}{m}, \quad \forall y \in \{1, \dots, \nu\} \quad (3)$$

$$\hat{y} = \underset{y}{argmax} \{pr_y\} \quad (4)$$

We predict class label  $\hat{y}$  by assigning the data point  $\mathbf{x}$  to the class with the maximum probability. Although most classification methods suffer from the *curse of dimensionality*, the random subspace method

takes advantage of high dimensionality. Opposite to the *Occam's razor*, generalization accuracy improves as the complexity of classifier increases [15].

## 2.2. Conic functions for classification

Conic functions to separate 2 sets of data is presented by Gasimov and Ozturk [24]. A conic function,  $g : \mathbb{R}^n \rightarrow \mathbb{R}$ , is defined as in Equation 5. In this equation,  $\mathbf{x}, \mathbf{w}, \mathbf{c} \in \mathbb{R}^n$  and  $\xi, p, \gamma \in \mathbb{R}$ , where  $\mathbf{x}$  is the data point,  $\mathbf{w}$  is the aim,  $\mathbf{c}$  is the vertex point and  $\gamma$  is the bias. We summarize parameters of a PCF in Table 2. In [24], the authors use a special case of this definition when  $p = 1$  and named the function polyhedral conic functions due to these type of functions have at most  $2^n$  sides in  $\mathbb{R}^n$ . Essentially, separation border between 2 data sets correspond to a level set of a PCF and this border is formed via intersection of utmost  $2^n$  half-spaces. Level set of a PCF is convex polyhedron. Figure 1 interprets the graph and level set of a PCF when the feature size is 2 of a data point. One can also see a decision boundary constructed by a PCF for a binary classification task from Figure 1. We label a data point as blue class or red class according to data point's value at that PCF. More specifically if a data point lays in the blue region, it gets a negative value at the PCF and gets a positive value, otherwise. Cimen et al. use  $p = 2$  norm to obtain circular separation borders [34].

$$g_{(\mathbf{w}, \xi, \gamma, \mathbf{c})}(\mathbf{x}) = \langle \mathbf{w}, (\mathbf{x} - \mathbf{c}) \rangle + \xi \|\mathbf{x} - \mathbf{c}\|_p - \gamma \quad (5)$$

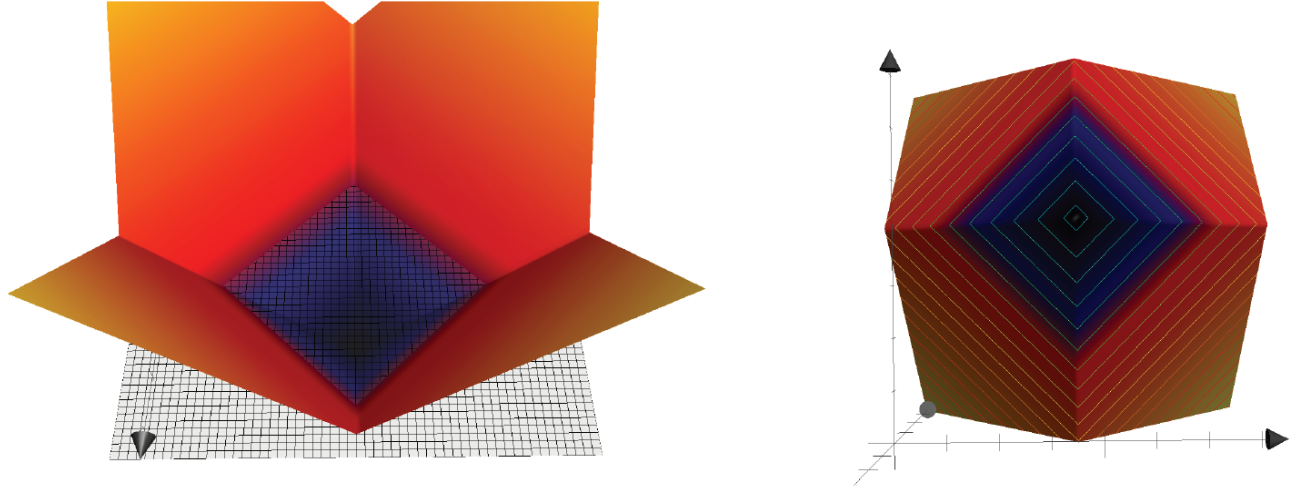
**Table 2.** Parameters of a PCF.

Parameter	Meaning
$\mathbf{x}$	Data point
$\mathbf{w}$	Aim
$\mathbf{c}$	Vertex point
$\xi$	Measure of folding of a PCF
$\gamma$	Bias
$p$	Norm
$n$	Feature size

Assume that a multiclass classification problem is converted into a binary classification problem where class  $y^*$  is the target. Then, training set  $\mathbf{T}$  is split into 2 sets;  $\mathbf{A}$  (target class) and  $\mathbf{B}$  (rest) as in Equation 6 and Equation 7. One way to obtain a PCF that separates  $\mathbf{A}$  and  $\mathbf{B}$ , is solving a linear programming (LP) model as given in Equation 8.

$$\mathbf{a}^h \in \mathbf{A} \subset \mathbf{T} \quad h \in H = \{f | y_f = y^*, f = 1, \dots, e\} \quad (6)$$

$$\mathbf{b}^j \in \mathbf{B} \subset \mathbf{T} \quad j \in J = \{f | y_f \neq y^*, f = 1, \dots, e\} \quad (7)$$



**Figure 1.** Graph and level set of a PCF in  $\mathbb{R}^2$  where  $z$  dimension is PCF value.

$$\begin{aligned}
 \min \quad & \frac{1}{|\mathbf{A}|} \sum_{h \in H} \theta_h + \frac{1}{|\mathbf{B}|} \sum_{j \in J} \psi_j \\
 \text{s.t.} \quad & \langle \mathbf{w}, (\mathbf{a}^h - \mathbf{c}) \rangle + \xi \|\mathbf{a}^h - \mathbf{c}\|_1 - \gamma \leq \theta_h \quad \forall h \in H \\
 & -\langle \mathbf{w}, (\mathbf{b}^j - \mathbf{c}) \rangle - \xi \|\mathbf{b}^j - \mathbf{c}\|_1 + \gamma \leq \psi_j \quad \forall j \in J \\
 & \xi, \gamma \geq 1 \\
 & \theta_h \geq 0 \quad \forall h \in H \\
 & \psi_j \geq 0 \quad \forall j \in J
 \end{aligned} \tag{8}$$

In the LP model in Equation 8, a constraint is added to the model for each data point in sets  $\mathbf{A}$  and  $\mathbf{B}$ . On obtained the PCF, it is expected that all the data points belong to set  $\mathbf{A}$ ,  $\mathbf{a}^h \in \mathbf{A}$ , takes a positive value and all the data points belong to set  $\mathbf{B}$ ,  $\mathbf{b}^j \in \mathbf{B}$ , takes a negative value. That's the reason of opposite signs in the constraints.  $\theta_h$  and  $\psi_j$  are the classification errors of the corresponding data points and obviously, the objective function aims to minimize weighted sum of the classification errors. It should be stated that the center  $\mathbf{c} \in \mathbb{R}^n$  is calculated as mean value of set  $\mathbf{A}$ , prior to the LP solution. By this way, linearity is satisfied in the model. Once the parameters  $\mathbf{w}$ ,  $\xi$  and  $\gamma$  are obtained as a solution of the LP, any data point  $\mathbf{x} \in \mathbb{R}^n$  in the test set can be classified as  $\mathbf{A}$  if  $g(\mathbf{w}, \xi, \gamma, \mathbf{c})(\mathbf{x}) < 0$  or classified as  $\mathbf{B}$ , otherwise.

### 3. Proposed method: random subspace based conic functions ensemble classifier (RS-CF)

In this study, we present an ensemble classification method based on conic functions. This method consists of multiple steps and in each step, our algorithm repeats the following processes: Firstly, training data is represented into fewer dimensional space by selecting dimensions randomly. Then, the corresponding LP model is solved to get the classifier parameters for each target class. At the end of these repeated steps, an ensemble of conic classifiers is obtained.

With this study, it is expected to reduce the overfitting problem of conic classifiers, when the dimensionality is high. In order to reach this goal, we get to benefit from the random subspace approach. The reason we



---

**Algorithm 2** A random subspace ensemble classifier based on conic functions.

---

- 0: **Input:** Training set  $\mathbf{T} \in \mathbb{R}^{e \times n}$ , number of classifiers to be trained for each class is  $m$ , dimension of subspaces is  $\tilde{n}$ , class labels  $y \in \{1, \dots, \nu\}$ .  
**Output:** Ensemble of classifiers  $G = \{g_{11}, \dots, g_{1m}, \dots, g_{\nu 1}, \dots, g_{\nu m}\}$  and selected random feature subsets  $S = \{S_1, \dots, S_m\}$ .  
 1: **for**  $i = 1$  to  $m$ :  
 2: Select a random subset of features  $\{1, \dots, n\}$  with  $\tilde{n}$  elements.  
 3: Hold selected random subset of features as  $S_i$ .  
 4: Project training set  $\mathbf{T}$  into  $S_i$  and get projection as  $\tilde{\mathbf{T}}_i \in \mathbb{R}^{e \times \tilde{n}}$ .  
 5: **for**  $y = 1$  to  $\nu$ :  
 6: Split  $\tilde{\mathbf{T}}_i$  into  $\tilde{\mathbf{A}}_i$  and  $\tilde{\mathbf{B}}_i$  where:

$$\tilde{\mathbf{a}}^h \in \tilde{\mathbf{A}}_i \subset \tilde{\mathbf{T}}_i \quad h \in H = \{f | y_f = y, f = 1, \dots, e\} \quad (9)$$

$$\tilde{\mathbf{b}}^j \in \tilde{\mathbf{B}}_i \subset \tilde{\mathbf{T}}_i \quad j \in J = \{f | y_f \neq y, f = 1, \dots, e\} \quad (10)$$

- 7: Calculate  $\mathbf{c}_{yi} \in \mathbb{R}^{\tilde{n}}$  as centroid of  $\tilde{\mathbf{A}}_i$ .  
 8: Solve the LP model to obtain  $\tilde{\mathbf{w}}_{yi}^1, \tilde{\mathbf{w}}_{yi}^2, \xi_{yi}, \gamma_{yi}$  parameters of  $g_{yi}$ :

$$\begin{aligned} \min \quad & \frac{1}{|\tilde{\mathbf{A}}_i|} \sum_{h \in H} \theta_h + \frac{1}{|\tilde{\mathbf{B}}_i|} \sum_{j \in J} \psi_j + \lambda (\|\tilde{\mathbf{w}}_{yi}^1\|_1 + \|\tilde{\mathbf{w}}_{yi}^2\|_1 + \xi_{yi} + \gamma_{yi}) \\ \text{s.t.} \quad & \langle (\tilde{\mathbf{w}}_{yi}^1 - \tilde{\mathbf{w}}_{yi}^2), (\tilde{\mathbf{a}}^h - \mathbf{c}_{yi}) \rangle + \xi_{yi} \left\| \tilde{\mathbf{a}}^h - \mathbf{c}_{yi} \right\|_1 - \gamma_{yi} \leq \theta_h \quad \forall h \in H \\ & -\langle (\tilde{\mathbf{w}}_{yi}^1 - \tilde{\mathbf{w}}_{yi}^2), (\tilde{\mathbf{b}}^j - \mathbf{c}_{yi}) \rangle - \xi_{yi} \left\| \tilde{\mathbf{b}}^j - \mathbf{c}_{yi} \right\|_1 + \gamma_{yi} \leq \psi_j \quad \forall j \in J \\ & \xi_{yi}, \gamma_{yi} \geq 1 \\ & \tilde{\mathbf{w}}_{yi}^1, \tilde{\mathbf{w}}_{yi}^2 \geq \mathbf{0} \\ & \theta_h \geq 0 \quad \forall h \in H \\ & \psi_j \geq 0 \quad \forall j \in J \end{aligned}$$

- 9: **end for**  
 10: **end for**
- 

want to solve this problem is that the effectiveness of conic function based classifiers on high dimensional data is low. The overfitting drawback of current conic function based classifier methods prevents the application of these approaches to some real-life problems. In Algorithm 2, developed random subspace-based conic functions ensemble classifier method is given.

Algorithm 2 differs from previous implementations of conic function based methods, in terms of model creation and the LP model. Algorithm 2 requires to solve  $(m \times \nu)$  LP models. By this way,  $m$  conic functions are obtained for each class. In step 2 of Algorithm 2, active  $\tilde{n}$  dimensions are randomly selected from feature set  $\{1, \dots, n\}$ . In step 4, the training set  $\mathbf{T}$  is projected into that particular subspace and  $\tilde{\mathbf{T}}_i \in \mathbb{R}^{e \times \tilde{n}}$  is obtained. Following to that process,  $\tilde{\mathbf{T}}_i$  is split into  $\tilde{\mathbf{A}}_i$  and  $\tilde{\mathbf{B}}_i$ , where data points of  $\tilde{\mathbf{A}}_i$  are in  $\tilde{\mathbf{T}}_i$  and their labels are equal to  $y$ . Oppositely, class labels of the data points of  $\tilde{\mathbf{B}}_i$ , do not equal to  $y$ . Essentially, this part is none other than one-vs-all approach for multiclass classification. Once the target class  $\tilde{\mathbf{A}}_i$ , and the rest  $\tilde{\mathbf{B}}_i$  are get, then the corresponding LP problem is solved to get  $\tilde{\mathbf{w}}_{yi}^1, \tilde{\mathbf{w}}_{yi}^2, \xi_{yi}, \gamma_{yi}$  parameters of classifier  $g_{yi}$ . Here,



$g_{yi}$  stands for the classifier of  $y^{th}$  class in  $i^{th}$  random subspace selection. When all iterations are completed, ensemble of classifiers  $G = \{g_{11}, \dots, g_{1m}, \dots, g_{\nu 1}, \dots, g_{\nu m}\}$  are obtained.

The LP model in step 8 of Algorithm 2 consists a constraint for each data point in  $\tilde{\mathbf{A}}_i$  and  $\tilde{\mathbf{B}}_i$ , in addition to the sign constraints. The first constraint group is all  $\mathbf{a}^h \in \tilde{\mathbf{A}}_i$  and  $\theta_h$  is the classification error for the corresponding data point.  $\theta_h$  is equal to 0, as long as related data point remains at the negative side of the PCF. Oppositely, in the second group of the constraints, the LP model tries to force all  $\mathbf{b}^j \in \tilde{\mathbf{B}}_i$  to be at the positive side of the PCF.  $\psi_j$  is the classification error for  $\mathbf{b}^j$  and it is equal to 0 if  $\mathbf{b}^j$  has a positive value in the PCF.

The objective function of the LP has 2 parts. The first part,  $\frac{1}{|\tilde{\mathbf{A}}_i|} \sum_{h \in H} \theta_h + \frac{1}{|\tilde{\mathbf{B}}_i|} \sum_{j \in J} \psi_j$ , is basically sum of classification errors in the training. Clearly, this part is balanced with the cardinality of  $\tilde{\mathbf{A}}_i$  and  $\tilde{\mathbf{B}}_i$ . The second part,  $\|\tilde{\mathbf{w}}_{yi}^1\|_1 + \|\tilde{\mathbf{w}}_{yi}^2\|_1 + \xi_{yi} + \gamma_{yi}$ , is the  $l_1$  norm regularization term [41], which is not seen in previous implementations of PCF based methods.  $\lambda$  is the tuning parameter between sum of classification errors and regularization term.

One can notice the difference between PCF definition in Equation 5-8 and the LP model in Algorithm 2. Such a redefinition as in Equation 11 is needed not to violate linearity in the regularization part in step 8 of Algorithm 2. Otherwise, it would be needed to use absolute value of  $\tilde{\mathbf{w}} \in \mathbb{R}^{\tilde{n}}$  because of  $\tilde{\mathbf{w}}$  is unconstrained in sign and that makes the model is not solvable by an LP model. In order to overcome this barrier,  $\tilde{\mathbf{w}}$  is rerepresented as in Equation 11 and one can see, thanks to Equation 12,  $l_1$  norm of the vector does not violate the linearity in the objective function. Additionally, it should be noted that the objective function will always force the  $\tilde{\mathbf{w}}_{yi}^1$  or  $\tilde{\mathbf{w}}_{yi}^2$  to be 0.

$$\tilde{\mathbf{w}} = \tilde{\mathbf{w}}^1 - \tilde{\mathbf{w}}^2; \quad \tilde{\mathbf{w}}^1, \tilde{\mathbf{w}}^2 \geq \mathbf{0} \quad (11)$$

$$\begin{aligned} \|\tilde{\mathbf{w}}\|_1 &= \|\tilde{\mathbf{w}}^1\|_1 + \|\tilde{\mathbf{w}}^2\|_1 = \sum_{d=1}^{\tilde{n}} \tilde{w}_d^1 + \tilde{w}_d^2 \\ \tilde{\mathbf{w}}^1 &= \{\tilde{w}_1^1, \dots, \tilde{w}_n^1\}; \quad \tilde{\mathbf{w}}^2 = \{\tilde{w}_1^2, \dots, \tilde{w}_n^2\} \end{aligned} \quad (12)$$

Prediction of any test point,  $\mathbf{x}$  is made according to majority voting as given the Algorithm 3. This computation is repeated for each data point in the test set.

$$\hat{y} = \underset{y}{\operatorname{argmax}} \{pr_y\} \quad (13)$$

In step 5 of Algorithm 3,  $\tilde{\mathbf{x}}_i$  is classified as class  $y$  by  $g_{yi}$ , if  $g_{yi}(\tilde{\mathbf{x}}_i) < 0$  is satisfied. In step 6, probabilities of corresponding classes are found and the predicted class label  $\hat{y}$  is found in Equation 13.

#### 4. Experimental results

In this section, training/testing accuracy and training/testing times of the proposed method are presented in comparison with support vector machines (SVM) [42], random forest [16, 17], AdaBoost [43], extreme learning machine (ELM)[44], single polyhedral conic function and  $k$ -means based polyhedral conic functions [28] classifiers. Such a comparison is essential to present the advantages and disadvantages of the study. SVM and random forest are selected for comparison due to being 2 of the most widely used classifiers. Another important point of comparison with random forest and AdaBoost is that they are ensemble learning-based

**Algorithm 3** Class label prediction.

---

0: **Input:** Ensemble of classifiers  $G = \{g_{11}, \dots, g_{1m}, \dots, g_{\nu 1}, \dots, g_{\nu m}\}$ , selected subsets  $S_i$  where  $i \in \{1, \dots, m\}$  and test point  $\mathbf{x} \in \mathbb{R}^n$ .  
**Output:** Class label  $\hat{y}$  and classification probabilities  $\mathbf{pr}$  of test point  $\mathbf{x}$ .

1: Set class probability vector  $\mathbf{pr} = \{pr_1, \dots, pr_\nu\} = \mathbf{0}$

2: **for**  $i = 1$  to  $m$ :

3: Project  $\mathbf{x}$  into  $S_i$  and obtain  $\tilde{\mathbf{x}}_i \in \mathbb{R}^{\tilde{n}}$ .

4: **for**  $y = 1$  to  $\nu$ :

5: **if**  $g_{yi}(\tilde{\mathbf{x}}_i) < 0$ :

6:  $pr_y = pr_y + (1/m)$

7: **end if**

8: **end for**

9: **end for**

---

methods. Extreme learning machine represents neural network based classifiers in this comparison.  $k$ -means based polyhedral conic functions method and single PCF can be shown as 2 of the most successful classifiers that use conic functions. We present results of classifiers when principal component analysis (PCA) [9] and linear discriminant analysis (LDA) [12] are applied to datasets to reduce dimensionality.

The hyper-parameters of the classifiers are optimized with grid search and the best test accuracy is selected among the obtained results. The  $\lambda$  of the proposed method is chosen from  $\{0.01, 0.05, 0.1, 0.2\}$ ,  $\tilde{n}$  is chosen from  $data\ set\ feature\ size \times \{0.1, 0.2, 0.3\}$  and the number of classifiers in the ensemble ( $M$ ) is chosen from  $\{5, 10, 20\}$ . Gaussian radial basis kernel (RBF) is used while training the SVM.  $\gamma$  of the RBF kernel is chosen from  $\{\text{auto calculation}, 2^{-5}, 2^{-3}, 2^{-1}, 1, 2^1, 2^3, 2^5\}$ . The cost of the SVM is searched in  $\{1, 2^3, 2^5\}$ . Random forest number of estimators are selected among  $\{5, 10, 20, 30\}$ . The cluster size of the  $k$ -Means PCF method is searched in  $\{1, 3, 5\}$ . The number of principal components selected among  $data\ set\ feature\ size \times \{0.1, 0.2, 0.3, 0.4\}$ . AdaBoost's number of estimators are selected from  $\{5, 10, 20, 30, 50, 100\}$ . 5-fold cross validation procedure is used in the tests.

We implement the proposed algorithm in MATLAB and Python 3.7. One can reach source codes from CIOL laboratory GitHub page<sup>1</sup>. We solve linear programming models with the Gurobi package [45] in Python and with MATLAB's linear programming solver. Tests are carried out on a computer with Intel (R) Core (TM) i7 CPU running at 2.5 GHz and 16 GB RAM.

7 benchmark data sets are selected from UCI ML data set repository [46]. Statlog data set [46] consists of multispectral values of pixels in  $3 \times 3$  neighborhoods in a satellite image, and the classification associated with the central pixel in each neighborhood. SCADI data set [47] contains measurements of 70 children with physical and motor disability based on ICF-CY. The aim in arrhythmia [48] data set is to distinguish between the presence and absence of cardiac arrhythmia and to classify it in 1 of the 16 groups. The classes in the data set are converted to 2 classes as healthy and patient. LSVT voice rehabilitation data set [49] features to correspond to speech signals, and the aim is to classify whether voice rehabilitation treatment leads to phonations considered acceptable or unacceptable. Parkinson's disease (PD) data set [50] consists of speech signal features of healthy controls and patients. CNAE-9 [51, 52] is a highly sparse (99.2%) data set containing 1080 documents of free text business descriptions of Brazilian companies. TTC-3600 Turkish text categorization data set [53]

---

<sup>1</sup>CIOL (2020). A random subspace based conic functions ensemble classifier source code [online]. Website <https://github.com/emrecimen/CIOL-ConicFunctionsBasedClassifiers> [accessed 13 April 2020].

consists of documents including 600 news/texts from 6 classes: economy, culture-arts, health, politics, sports, and technology. The number of features, classes, and data points in the data sets can be found in Table 3. We choose these datasets because their number of features/number of samples ratio are high.

**Table 3.** Description of the datasets.

Data set	# Data points	# Features	# Classes
Statlog	6435	36	6
SCADI	70	206	7
Arrhythmia	452	279	2
LSVT	126	309	2
PD	756	754	2
CNAE-9	1080	857	9
TTC-3600	3600	4814	6

Training and testing times are provided in Tables 4 and 5. The proposed method requires more training time than other methods. It is because an LP is solved recursively  $m \times \nu$  (number of classifiers in the ensemble  $\times$  number of classes) times. Random forest algorithm has an advantage on the other methods in terms of training and testing time. AdaBoost and ELM require very short time for testing. Combining a classifier with PCA shortens its test time. However since computing principal components takes time, it increases training time in some tests. All methods' testing times are less than 2 min.

Training and testing accuracies are presented in Tables 6 and 7. SVM and random forest have an overfitting problem in all data sets. Similarly,  $k$ -Means PCF overfits in all data sets except Statlog. ELM, AdaBoost, and the RS-CF methods are more robust to overfitting. However, AdaBoost fails on CNAE-9, and ELM fails on TTC-3600. Although combining classifiers with PCA to get fewer dimensions has some improvements on overfitting, it does not solve this problem.

Out of 7 sets, in the 2 sets SVM + PCA, in the 1 set RF, in the 2 sets ELM + PCA, in the 1 set AdaBoost and in the 3 sets the proposed method RS-CF achieve the best performance. No method dominates the others. Although PCA does not prevent overfitting of classifiers, it is clear that PCA provides some improvement in the test performances. Contrary to PCA, LDA could not help to improve test performances. Furthermore, it could not prevent overfitting in most of the tests. Especially in the TTC-3600 dataset, LDA combined classifiers performed poorly. AdaBoost fails on CNAE-9 but achieves consistent results in the other test sets. Similarly, ELM performs poorly on TTC-3600 but achieves the best results in the 2 of the sets. RF never fails but can achieve the best test accuracy only in 1 of the tests. RS-CF achieves the best accuracy on 3 of the sets and lays in the first or second place on 5 of the tests. RS-CF achieves better accuracy than the previous implementations of conic function based classifiers  $k$ -Means PCF and single PCF. On 6 of the tests, RS-CF achieves better accuracy than  $k$ -Means PCF and single PCF. One can see graphs of training and test accuracies can in Figures 2 and 3.

## 5. Conclusions

High dimensionality in the data has always been a problem when the number of samples is not enough, and the number of features/number of samples ratio is high. This problem may arise in many real-life problems, e.g., medicine, biology, physics. In this case, the feature space cannot be well-represented, and almost all of the

**Table 4.** Training times.

	Statlog	SCADI	Arrhythmia	LSTV	PD	CNAE-9	TTC-3600
<b>SVM</b>	2.11	0.01	0.20	0.03	1.24	1.80	333
<b>SVM + PCA</b>	1.44	0.01	0.04	0.01	0.16	0.73	186
<b>SVM + LDA</b>	2.54	0.05	0.16	0.06	0.87	1.48	91.9
<b>RF</b>	1.24	0.03	0.21	0.09	0.81	0.31	4.35
<b>RF + PCA</b>	1.22	0.13	0.19	0.20	0.28	1.00	12.3
<b>RF + LDA</b>	1.39	0.17	0.21	0.18	0.75	1.44	72.2
<b>k-means PCF</b>	54.7	1.26	2.34	11.8	183	8.78	484
<b>k-means PCF + PCA</b>	43.8	1.22	1.36	1.54	42.9	22.7	43.0
<b>k-means PCF + LDA</b>	99.8	0.48	1.00	0.29	2.07	35.0	131
<b>single PCF</b>	23.7	1.88	3.19	7.36	110	5.48	384
<b>single PCF + PCA</b>	14.5	1.25	1.07	0.87	35.6	7.61	1093
<b>single PCF + LDA</b>	26.8	0.42	0.60	0.29	1.33	9.08	92.8
<b>ELM</b>	3.91	0.17	0.49	0.15	0.52	0.91	16.6
<b>ELM + PCA</b>	0.64	0.09	0.18	0.15	0.68	0.84	197
<b>ELM + LDA</b>	0.67	0.11	0.21	0.17	2.04	4.02	5069
<b>AdaBoost</b>	0.40	0.13	0.19	1.63	18.7	0.56	43.0
<b>AdaBoost + PCA</b>	0.11	0.04	1.40	0.39	1.31	4.54	222
<b>AdaBoost + LDA</b>	0.42	0.37	0.23	0.06	0.71	1.97	81.1
<b>Proposed (RS-CF)</b>	476	2.91	9.44	2.85	38.3	112	22526

**Table 5.** Test times.

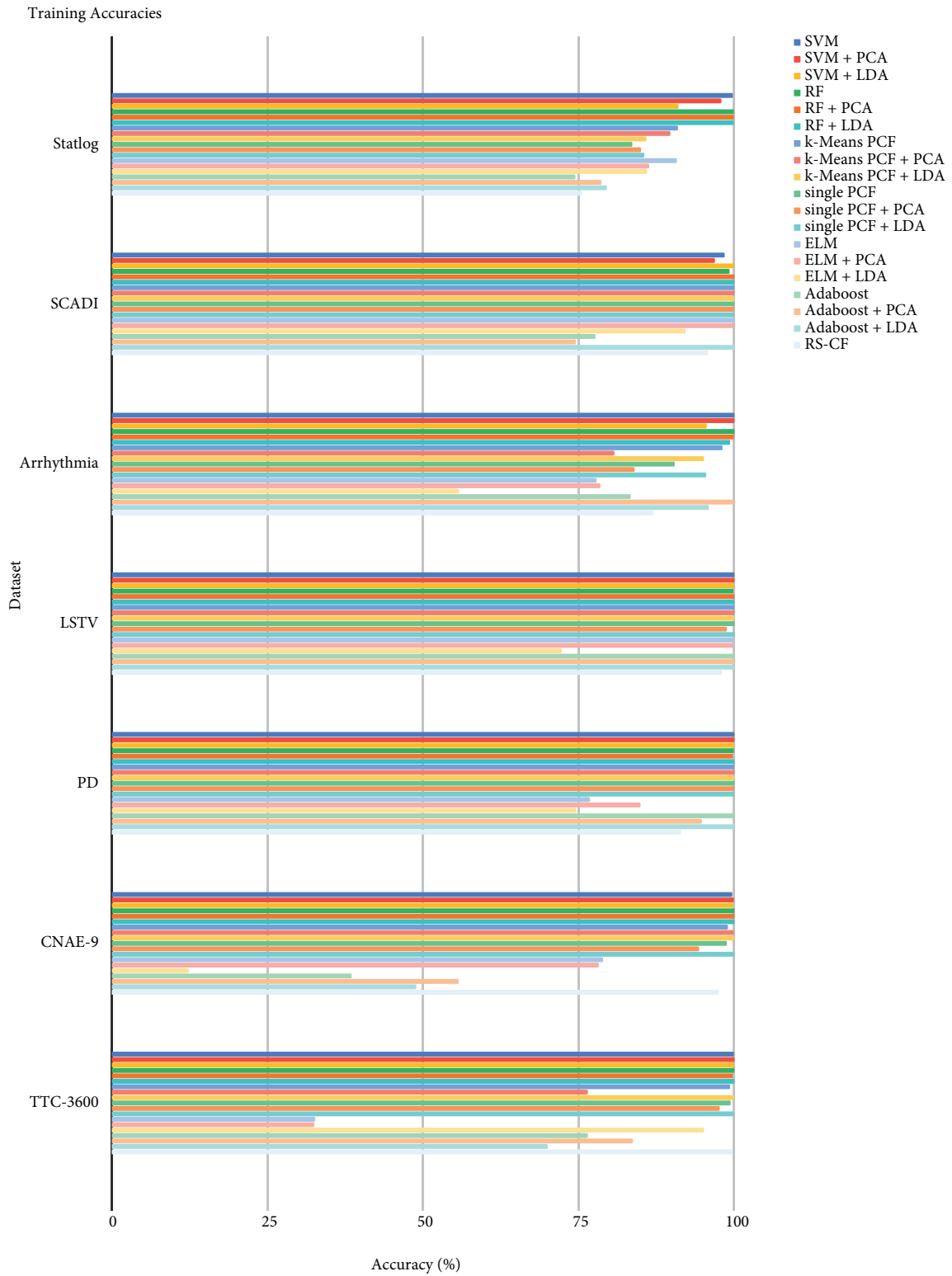
	Statlog	SCADI	Arrhythmia	LSTV	PD	CNAE-9	TTC-3600
<b>SVM</b>	0.50	0.01	0.05	0.01	0.31	0.39	50.0
<b>SVM + PCA</b>	0.27	0.01	0.01	0.01	0.03	0.15	25.5
<b>SVM + LDA</b>	0.27	0.01	0.01	0.01	0.02	0.04	0.35
<b>RF</b>	0.04	0.01	0.02	0.01	0.02	0.03	0.19
<b>RF + PCA</b>	0.03	0.01	0.01	0.02	0.01	0.03	0.06
<b>RF + LDA</b>	0.05	0.02	0.01	0.02	0.04	0.04	0.25
<b>k-means PCF</b>	14.7	0.07	0.23	0.04	0.37	1.89	73.5
<b>k-means PCF + PCA</b>	18.5	0.05	0.15	0.01	0.27	3.00	2.48
<b>k-means PCF + LDA</b>	2.46	0.02	0.07	0.02	0.12	0.66	1.71
<b>single PCF</b>	1.74	0.08	0.11	0.02	0.21	0.82	26.5
<b>single PCF + PCA</b>	4.56	0.06	0.06	0.01	0.05	0.54	1.68
<b>single PCF + LDA</b>	0.61	0.01	0.04	0.01	0.05	0.18	0.70
<b>ELM</b>	0.17	0.01	0.06	0.01	0.06	0.05	1.15
<b>ELM + PCA</b>	0.03	0.01	0.02	0.01	0.01	0.02	0.56
<b>ELM + LDA</b>	0.04	0.01	0.01	0.01	0.01	0.04	0.05
<b>AdaBoost</b>	0.02	0.02	0.02	0.08	0.15	0.05	2.48
<b>AdaBoost + PCA</b>	0.01	0.01	0.08	0.04	0.05	0.05	0.87
<b>AdaBoost + LDA</b>	0.02	0.04	0.01	0.01	0.01	0.06	0.45
<b>Proposed (RS-CF)</b>	43.0	0.19	0.59	0.16	1.01	18.1	84.9

**Table 6.** Training accuracies.

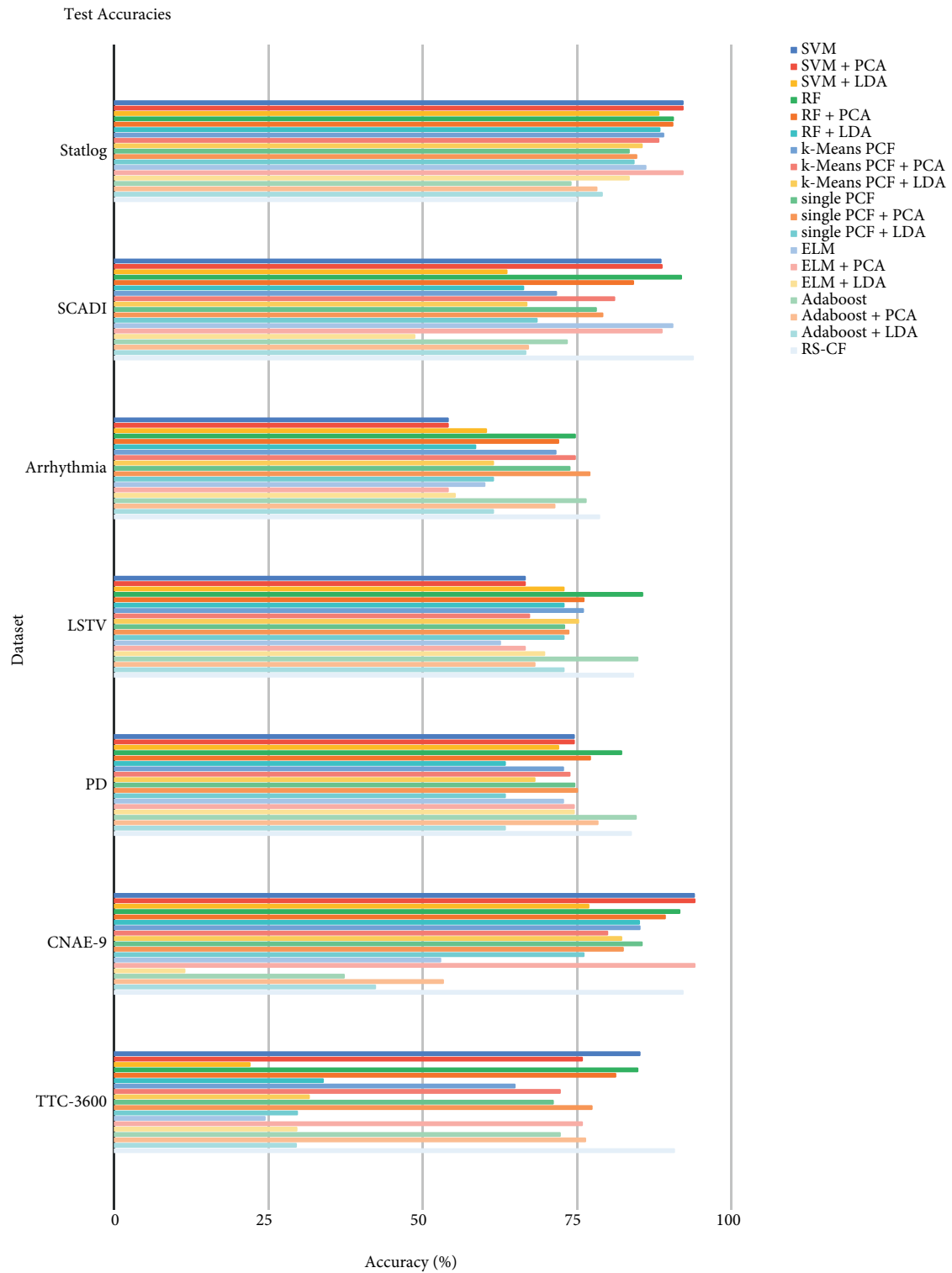
	Statlog	SCADI	Arrhythmia	LSTV	PD	CNAE-9	TTC-3600
SVM	99.76	98.45	100.0	100.0	100.0	99.70	99.88
SVM + PCA	97.92	96.87	100.0	100.0	100.0	99.93	99.97
SVM + LDA	91.03	100.0	95.52	100.0	100.0	99.81	99.98
RF	99.96	99.19	100.0	99.80	99.90	100.0	99.98
RF + PCA	99.88	100.0	99.88	100.0	99.76	100.0	99.76
RF + LDA	99.93	100.0	99.34	100.0	100.0	100.0	99.98
k-means PCF	90.97	100.0	98.06	100.0	100.0	98.96	99.27
k-means PCF + PCA	89.70	100.0	80.70	100.0	100.0	99.91	76.49
k-means PCF + LDA	85.90	100.0	95.08	100.0	100.0	100.0	99.98
single PCF	83.64	100.0	90.43	100.0	100.0	98.77	99.38
single PCF + PCA	85.00	100.0	83.96	98.81	100.0	94.35	97.65
single PCF + LDA	85.53	100.0	95.47	100.0	100.0	99.98	99.94
ELM	90.74	100.0	77.87	99.80	76.82	78.94	32.71
ELM + PCA	86.28	100.0	78.43	99.80	84.95	78.22	32.52
ELM + LDA	85.96	92.19	55.81	72.23	74.60	12.31	95.15
AdaBoost	74.42	77.72	83.35	100.0	100.0	38.54	76.49
AdaBoost + PCA	78.61	74.56	100.0	100.0	94.74	55.69	83.68
AdaBoost + LDA	79.48	100.0	95.90	100.0	100.0	48.89	69.97
Proposed (RS-CF)	75.52	95.71	87.00	98.01	91.50	97.50	99.62

**Table 7.** Test accuracies.

	Statlog	SCADI	Arrhythmia	LSTV	PD	CNAE-9	TTC-3600
SVM	92.24	88.69	54.20	66.69	74.60	94.07	85.28
SVM + PCA	<b>92.26</b>	88.85	54.20	66.67	74.60	<b>94.17</b>	75.92
SVM + LDA	88.32	63.71	60.40	72.95	72.10	76.95	22.14
RF	90.66	91.94	74.79	<b>85.72</b>	82.27	91.76	84.92
RF + PCA	90.57	84.23	72.14	76.18	77.27	89.35	81.36
RF + LDA	88.46	66.41	58.64	72.95	63.50	85.19	33.95
k-means PCF	89.11	71.79	71.68	76.09	72.89	85.28	65.03
k-means PCF + PCA	88.29	81.14	74.77	67.40	73.90	80.00	72.36
k-means PCF + LDA	85.65	66.92	61.51	75.29	68.26	82.31	31.70
single PCF	83.51	78.21	73.89	73.04	74.70	85.65	71.22
single PCF + PCA	84.76	79.24	77.19	73.75	75.13	82.59	77.50
single PCF + LDA	84.29	68.59	61.51	72.95	63.50	76.20	29.80
ELM	86.20	90.60	60.17	62.71	72.88	52.96	24.53
ELM + PCA	<b>92.26</b>	88.85	54.20	66.67	74.60	<b>94.17</b>	75.92
ELM + LDA	83.50	48.83	55.31	69.82	74.60	11.57	29.69
Adaboost	74.06	73.46	76.57	84.92	<b>84.66</b>	37.41	72.36
Adaboost + PCA	78.29	67.18	71.47	68.22	78.44	53.42	76.42
Adaboost + LDA	79.19	66.80	61.51	72.95	63.49	42.41	29.64
Proposed (RS-CF)	75.10	<b>93.94</b>	<b>78.76</b>	84.18	83.86	92.22	<b>90.86</b>



**Figure 2.** Training accuracies.



**Figure 3.** Test accuracies.



classification methods struggle with the overfitting problem. Feature selection/extraction algorithms represent data with lower dimensions to overcome this problem. Another approach is ensemble learning, which trains multiple classifiers. In this method, a projection of input feature space into a random subspace is used to train each classifier.

PCF based classifier methods have shown to be successful classifiers in the literature with applications. However, current versions of PCF based classifiers have the overfitting problem when the feature size is close or higher than the number of data points. In this study, a random subspace ensemble classifier based on conic functions (RS-CF) is introduced. We aimed to bring a novel solution to the overfitting problem of conic function based classifiers. We evaluated the performance of the proposed method in comparison with the wide range of other methods presented in the literature. SVM, RF, AdaBoost, ELM,  $k$ -Means PCF, single PCF classifiers are used for the benchmark. We applied PCA and LDA for dimensionality reduction and used reduced versions of data sets with benchmark classifiers. We show that the proposed method has achieved the best results more than other methods, and it can compete with the other methods. Moreover, the proposed method overperforms  $k$ -Means PCF and single PCF, which are the previous implementation of conic functions based classifiers. In future research, we plan to apply the proposed method to real-life applications in medicine and psychology.

## Acknowledgment

The author would like to thank the editor and anonymous reviewers. Their valuable evaluations and suggestions improved the paper significantly. The author also thanks Dr. Gürkan Öztürk (Department of Industrial Engineering, Eskişehir Technical University, Eskişehir, Turkey) for his comments prior to the submission of the paper.

## References

- [1] Wang H, Cimen E, Singh N, Buckler E. Deep learning for plant genomics and crop improvement. *Current Opinion in Plant Biology* 2020; 54: 34-41.
- [2] Aggarwal C C. *Data Mining: The Textbook*. New York, USA: Springer Publishing Company, 2015.
- [3] Roffo G, Melzi S, Cristani M. Infinite feature selection. In: 2015 IEEE International Conference on Computer Vision (ICCV); Santiago, Chile; 2015. pp. 4202-4210.
- [4] Roffo G, Melzi S. Ranking to learn: Feature ranking and selection via eigenvector centrality. In: Appice A, Ceci M, Loglisci C, Masciari E, Ras ZW (editors). *New Frontiers in Mining Complex Patterns*. Cham, Switzerland: Springer International Publishing, 2016.
- [5] Zhang Y, Wang S, Phillips P, Ji G. Binary pso with mutation operator for feature selection using decision tree applied to spam detection. *Knowledge-Based Systems* 2014; 64: 22-31.
- [6] Xuan P, Guo MZ, Wang J, Wang CY, Liu XY et al. Genetic algorithm-based efficient feature selection for classification of pre-mirnas. *Genetics and Molecular Research: GMR* 2011; 10 (2): 588-603.
- [7] Blum A L, Langley P. Selection of relevant features and examples in machine learning. *Artificial Intelligence* 1997; 97 (1): 245-271.
- [8] Chandrashekar G, Sahin F. A survey on feature selection methods. *Computers & Electrical Engineering* 2014; 40 (1): 16-28.
- [9] FRS K P. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* 1901; 2 (11): 559-572.

- [10] Golub G H, Reinsch C. Singular Value Decomposition and Least Squares Solutions. Berlin-Heidelberg, Germany: Springer, 1971.
- [11] Dumais S T. Latent semantic analysis. *Annual Review of Information Science and Technology* 2004; 38 (1): 188-230.
- [12] Fisher R A. The use of multiple measurements in taxonomic problems. *Annals of Eugenics* 1936; 7 (2): 179-188.
- [13] Vincent P, Larochelle H, Bengio Y, Manzagol P A. Extracting and composing robust features with denoising autoencoders. In: *Proceedings of the 25th International Conference on Machine Learning, ICML '08*; New York, NY, USA; 2008; pp. 1096-1103.
- [14] Khalid S, Khalil T, Nasreen S. A survey of feature selection and feature extraction techniques in machine learning. In: *2014 Science and Information Conference*; London, UK; 2014. pp. 372-378.
- [15] Ho T K. The random subspace method for constructing decision forests. *IEEE Transactions on Pattern Analysis & Machine Intelligence* 1998; 20: 832-844.
- [16] Ho T K. Random decision forests. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*; Quebec, Canada; 1995. pp. 278-282.
- [17] Breiman L. Random forests. *Machine Learning* 2001; 45 (1): 5-32.
- [18] Freund Y, Schapire R E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences* 1997; 55 (1): 119-139.
- [19] Xiong J, Zhang K, Zhang H. A vibrating mechanism to prevent neural networks from overfitting. In: *2019 15th International Wireless Communications Mobile Computing Conference (IWCMC)*; Tangier, Morocco; 2019. pp. 1737-1742.
- [20] Werpachowski R, György A, Szepesvari C. Detecting overfitting via adversarial examples. In: *Advances in Neural Information Processing Systems (NIPS 2019)*; Vancouver, Canada; 2019. pp. 7856-7866.
- [21] Feldman V, Frostig R, Hardt M. The advantages of multiple classes for reducing overfitting from test set reuse. *arXiv.org* 2019; arXiv:1905.10360.
- [22] Wu B, Liu Z, Yuan Z, Sun G, Wu C. Reducing overfitting in deep convolutional neural networks using redundancy regularizer. In: *Artificial Neural Networks and Machine Learning–ICANN 2017*; Alghero, Italy; 2017. pp. 49-55.
- [23] Salman S, Liu X. Overfitting mechanism and avoidance in deep neural networks. *arXiv.org* 2019; arXiv:1901.06566.
- [24] Gasimov R N, Ozturk G. Separation via polyhedral conic functions. *Optimization Methods and Software* 2006; 21 (4): 527-540.
- [25] Ozturk G. A new mathematical programming approach to solve classification problems. PhD, Eskişehir Osmangazi University, Institute of Science, Eskişehir, Turkey, 2007. (in Turkish).
- [26] Bagirov A M, Ugon J, Webb D, Ozturk G, Kasimbeyli R. A novel piecewise linear classifier based on polyhedral conic and max–min separabilities. *TOP* 2013; 21 (1): 3-24.
- [27] Ozturk G, Bagirov A M, Kasimbeyli R. An incremental piecewise linear classifier based on polyhedral conic separation. *Machine Learning* 2015; 101 (1): 397-413.
- [28] Ozturk G, Ciftci M T. Clustering based polyhedral conic functions algorithm in classification. *Journal of Industrial and Management Optimization* 2015; 11 (3): 921-932.
- [29] Cimen E, Ozturk G. Arrhythmia classification via k-means based polyhedral conic functions algorithm. In: *2016 International Conference on Computational Science and Computational Intelligence*; Las Vegas, USA; 2016. pp. 798-802.
- [30] Cevikalp H, Triggs B. Polyhedral conic classifiers for visual object detection and classification. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*; Hawaii, USA, 2017. pp. 4114-4122.
- [31] Cimen E. Gesture Recognition with Polyhedral Conic Functions based Classifiers. MS, Graduate School of Science, Anadolu University, Eskişehir, Turkey, 2013 (in Turkish).

- [32] Uylas Sati N. A binary classification approach based on support vector machines via polyhedral conic functions. Celal Bayar University Journal of Science 2016; 12 (2): 135-149.
- [33] Uylas Sati N, Ordin B. Application of the polyhedral conic functions method in the text classification and comparative analysis. Scientific Programming 2018; 2018: 1-11.
- [34] Cimen E, Ozturk G, Gerek O N. Incremental conic functions algorithm for large scale classification problems. Digital Signal Processing 2018; 77: 187-194.
- [35] Cimen E, Ozturk G, Gerek O N. Icf: An algorithm for large scale classification with conic functions. SoftwareX 2018; 8: 59-63.
- [36] Cimen E, Ozturk G. O-pcf algorithm for one-class classification. Optimization Methods and Software 2019; 0 (0): 1-15.
- [37] Ozturk G, Cimen E. Polyhedral conic kernel-like functions for svms. Turkish Journal of Electrical Engineering & Computer Sciences 2019; 27: 1172-1180.
- [38] Cimen E. Optimization based predictive methods for large scale data. PhD, Eskişehir Technical University, Graduate School of Sciences, Eskişehir, Turkey, 2018.
- [39] Dordinejad G G, Cevikalp H. Cone vertex estimation in polyhedral conic classifiers. In: 25th Signal Processing and Communications Applications Conference (SIU); Antalya, Turkey; 2017. pp. 1-4.
- [40] Ozturk G, Ceylan G. Max margin polyhedral conic function classifier. In: 2016 International Conference on Computational Science and Computational Intelligence (CSCI); Las Vegas, USA; 2016. pp. 1395-1396.
- [41] Zhu J, Rosset S, Hastie T, Tibshirani R. Inormm support vector machines. In: Proceedings of the 16th International Conference on Neural Information Processing Systems, NIPS'03; Vancouver, Canada; 2003. pp. 49-56.
- [42] Cortes C, Vapnik V. Support-vector networks. Machine Learning 1995; 20 (3): 273-297.
- [43] Zhu J, Zou H, Rosset S, Hastie T. Multi-class adaboost. Statistics and Its Interface 2009; 2: 349-360.
- [44] Huang G, Zhou H, Ding X, Zhang R. Extreme learning machine for regression and multiclass classification. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 2012; 42 (2): 513-529.
- [45] Gurobi Optimization I. Gurobi optimizer reference manual, 2016.
- [46] Lichman M. UCI machine learning repository, 2013.
- [47] Zarchi M, Bushehri S F, Dehghanizadeh M. Scadi: A standard dataset for self-care problems classification of children with physical and motor disability. International Journal of Medical Informatics 2018; 114: 81-87.
- [48] Guvenir H A, Acar B, Demiroz G, Cekin A. A supervised machine learning algorithm for arrhythmia analysis. In: Computers in Cardiology; Lund, Sweden; 1997. pp. 433-436.
- [49] Tsanas A, Little M A, Fox C, Ramig L O. Objective automatic assessment of rehabilitative speech treatment in parkinson's disease. IEEE Transactions on Neural Systems and Rehabilitation Engineering 2014; 22 (1): 181-190.
- [50] Sakar C O, Serbes G, Gunduz A, Tunc HC, Nizam H et al. A comparative analysis of speech signal processing algorithms for parkinson's disease classification and the use of the tunable q-factor wavelet transform. Applied Soft Computing 2019; 74: 255-263.
- [51] Ciarelli P M, Oliveira E. Agglomeration and elimination of terms for dimensionality reduction. In: Proceedings of the 2009 Ninth International Conference on Intelligent Systems Design and Applications, ISDA '09; Washington, DC, USA; 2009. pp. 547-552.
- [52] Ciarelli P M, Salles E O T, Oliveira E. An evolving system based on probabilistic neural network. In: 11th Brazilian Symposium on Neural Networks; Sao Paulo, Brazil; 2010. pp. 182-187.
- [53] Kilinc D, Ozcift A, Bozyigit F, Yildirim P, Yucalar F et al. Ttc-3600: A new benchmark dataset for turkish text categorization. Journal of Information Science 2017; 43 (2): 174-185.