Journal of Mathematical Modeling
Vol. -, No. -, 2026, pp. -. Research Article

JMM

Enhancing support vector machine performance through boundary-focused covariance matrix learning

Afsaneh Pourmoezi, Ali Tavakoli, Mostsfa Eslami*

Department of Applied Mathematics, University of Mazandaran, Babolsar, Iran
Email(s): Afsaneh.pourmoezi@gmail.com, a.tavakoli@umz.ac.ir, mostafa.eslami@umz.ac.ir

Abstract. Support Vector Machine (SVM) has proven effective in various classification tasks by focusing on maximizing the margin between classes. However, standard SVM often fails to consider the underlying geometric structure of data near the decision boundary, particularly in scenarios where class overlap or noise occurs. This paper proposes a new method that focuses on boundary-critical samples, those near the decision boundary, by integrating their covariance matrix into the learning process. The principal eigenvector of this covariance matrix is then utilized to guide the classifier towards the most informative regions of the data, enabling the classifier to better capture the local geometry. Importantly, this modification is confined to the objective function, ensuring the convexity of the optimization problem is preserved. Experimental results across various datasets, including linearly and non-linearly separable ones, demonstrate that the proposed method provides competitive or slightly improved classification performance.

Keywords: Support vector machine, boundary samples, covariance matrix, eigenvector.
AMS Subject Classification 2010: 68T10, 68T01.

1 Introduction

Data classification remains one of the fundamental problems in machine learning and pattern recognition [17, 32, 33]. Among the various techniques developed for this task, Support Vector Machines (SVMs) [18, 26] have proven to be highly effective due to their solid theoretical foundation and practical performance. By constructing a hyperplane that maximizes the margin between different classes [11], SVMs have achieved success in diverse applications including medical diagnosis [5], financial forecasting [16], and face recognition [22].

However, standard SVM formulations assume that all training data are equally reliable and relevant, which may not hold in real-world scenarios where datasets often contain noisy labels, overlapping re-

*Corresponding author
Received: 14 May 2025/ Revised: 28 September 2025/ Accepted: 13 October 2025
DOI: 10.22124/jmm.2025.30656.2746

gions, and non-linear class boundaries. For such instances, the application of the same treatment to all samples yields suboptimal or unstable classifiers. To overcome such disadvantages, a number of SVM extensions have been suggested [27]. Fuzzy SVMs [24] offer confidence-based weights for samples to decrease the influence of outliers and noisy instances. Other methods like Margin Distribution Optimization [13], Localized Multiple Kernel Learning [15], and Robust SVMs [2, 19, 34] aim to optimize the margin behavior around problematic decision boundaries or emphasize regional shapes of the data. Despite these advances, most contemporary methods still employ global statistics of the data or margin-based rescaling rather than modeling sample geometry directly in conjunction with the decision boundary. In this work, we propose a new approach that extends the standard *SVM* by focusing directly on boundary-critical samples, i.e., those along the separating hyperplane, and by constructing a covariance matrix from this subset, the approach incorporates local geometry information into the optimization procedure. With this, the model can adequately estimate the decision boundary shape and orientation, especially in cases of boundary noise or class overlap. Convexity is preserved by modifying only the objective function while retaining the computation efficiency benefits of traditional SVMs. We propose a new formulation that makes use of the covariance structure of boundary-critical samples to enhance the sensitivity of the model to informative regions. Throughout the paper, we denote the norm-2 vector by $\|\cdot\|$.

The rest of this paper is organized as follows: Section 2 introduces the theoretical background of SVMs. Our proposed method is presented in Section 3. In this section, the details of the method along with the corresponding algorithm are described. In Section 4, several examples with synthetic and real data in different dimensions are applied to our proposed method, standard SVM and some other baseline methods, and their accuracy is examined. Finally, Section 5 concludes the study and outlines future research directions.

2 Theoretical background of SVM

The SVM is powerful supervised learning models designed to construct an optimal hyperplane that separates different classes with the widest possible margin [4]. By concentrating on a subset of critical training samples, known as support vectors, SVM achieves robust generalization capabilities, even in high-dimensional feature spaces [8, 12, 31].

Given a set of labeled training samples (x_i, y_i), where $x_i \in \mathbb{R}^d$ and $y_i \in \{-1, +1\}$, the standard (hard-margin) SVM formulation seeks to solve the following primal optimization problem:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1, \quad \forall i. \end{aligned} \tag{1}$$

Here, w represents the normal vector to the hyperplane, and b is the bias term that determines the hyperplane's position in the feature space.

In practical applications where perfect linear separability is not achievable, slack variables ξ_i are

introduced, leading to the soft-margin SVM:

$$\begin{aligned} \min_{w,b,\xi} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi_i \\ \text{subject to} \quad & y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i, \\ & \xi_i \geq 0, \quad \forall i, \end{aligned} \tag{2}$$

where $C > 0$ is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the classification error.

From a geometric standpoint, SVM aims to maximize the distance between the separating hyperplane and the nearest samples from each class. These closest samples, termed *support vectors*, exclusively determine the orientation and position of the hyperplane [8].

The margin, defined as the perpendicular distance between the hyperplane and the support vectors, can be expressed as:

$$\text{Margin} = \frac{2}{\|w\|}. \tag{3}$$

Thus, minimizing $\|w\|^2$ is directly equivalent to maximizing the margin, leading to improved generalization on unseen data.

To tackle non-linearly separable data, SVM leverages the so-called *kernel trick*, wherein data is implicitly mapped into a higher-dimensional feature space via a kernel function $K(x_i, x_j)$ [6, 29]. This allows the hyperplane to perform linear separation in the transformed space without explicitly computing the mapping [1, 3, 21].

Common kernel functions include:

- **Linear kernel:** $K(x_i, x_j) = x_i^T x_j$,
- **Polynomial kernel:** $K(x_i, x_j) = (x_i^T x_j + r)^d$,
- **Radial Basis Function (RBF) kernel:** $K(x_i, x_j) = \exp(-\gamma\|x_i - x_j\|^2)$.

Employing appropriate kernels enables SVM to model highly complex and non-linear decision boundaries effectively.

Ultimately, SVM offers a principled framework for classification by focusing on maximizing the margin while maintaining computational efficiency through reliance on support vectors and kernel methods.

3 Proposed method

In this section, we introduce a novel modification to the standard SVM aimed at enhancing classification accuracy by focusing on boundary-critical samples.

Standard SVM utilizes all training samples to define the separating hyperplane. However, many samples, particularly those far from the decision boundary, contribute little to the classification task and may introduce noise. Concentrating on samples near the boundary offers a more efficient and discriminative model.

Let the dataset be denoted by $\{(x_i, y_i)\}_{i=1}^n$, where $x_i \in \mathbb{R}^d$ is a feature vector and $y_i \in \{-1, 1\}$ is its corresponding class label.

Let $C^+ = \{x_i \mid y_i = +1\}$ and $C^- = \{x_i \mid y_i = -1\}$ denote the sets of samples belonging to class $+1$ and class -1, respectively. We denote the mean of data points in class $+1$ and class -1 by μ^+ and μ^-, respectively.

To identify samples near the decision boundary, we extract a subset of boundary-critical samples from both classes based on their proximity to the opposing class mean. We introduce a hyperparameter r (with $0 < r \leq 1$) that defines the proportion of samples to select from each class. For each sample $x_i \in C^+$, we compute its distance to the class -1 mean as

$$d_i^{(+\to-)} = \|x_i - \mu^-\|.$$

Let $\pi^{+\to-}$ be the permutation that sorts the samples in C^+ according to the ascending order of $d_i^{(+\to-)}$. Then, the closest $r \cdot 100\%$ of class $+1$ samples to the class -1 mean are selected as:

$$S^{+\to-} = \left\{ x_{\pi^{+\to-}(i)} \;\middle|\; i = 1, 2, \ldots, \lceil r \cdot |C^+| \rceil \right\}.$$

Similarly, for each sample $x_j \in C^-$, we compute the distance to the class $+1$ mean as

$$d_j^{(-\to+)} = \|x_j - \mu^+\|.$$

Let $\pi^{-\to+}$ be the permutation that sorts the samples in C^- in ascending order of $d_j^{(-\to+)}$. Then, the closest $r \cdot 100\%$ of class -1 samples to the class $+1$ mean are defined as:

$$S^{-\to+} = \left\{ x_{\pi^{-\to+}(j)} \;\middle|\; j = 1, 2, \ldots, \lceil r \cdot |C^-| \rceil \right\}.$$

Finally, we define the set of boundary-critical samples as

$$S_v = S^{+\to-} \cup S^{-\to+}. \tag{4}$$

To determine an appropriate separating hyperplane, in our proposed method, we consider the data scatter. Let the selected set of m samples be represented as

$$\mathbf{x}_v^{(i)} \in \mathbb{R}^d, \quad i = 1, 2, \ldots, m,$$

where each vector $\mathbf{x}_v^{(i)}$ denotes the i-th sample in a d-dimensional feature space.

The sample scatter (covariance) matrix is then given by

$$\Sigma_v = \frac{1}{m-1} \sum_{i=1}^{m} \left(\mathbf{x}_v^{(i)} - \bar{\mathbf{x}}_v \right) \left(\mathbf{x}_v^{(i)} - \bar{\mathbf{x}}_v \right)^\top,$$

where

$$\bar{\mathbf{x}}_v = \frac{1}{m} \sum_{i=1}^{m} \mathbf{x}_v^{(i)},$$

is the mean vector of the samples. Here, each term $\left(\mathbf{x}_v^{(i)} - \bar{\mathbf{x}}_v \right) \left(\mathbf{x}_v^{(i)} - \bar{\mathbf{x}}_v \right)^\top$ is a $d \times d$ matrix representing the contribution of the i-th sample to the overall scatter.

To obtain the eigenvectors of the covariance matrix Σ_v, we formulate the eigenvalue equation as

$$\Sigma_v \mathbf{v}_k = \lambda_k \mathbf{v}_k, \quad k = 1, 2, \ldots, d,$$

where

- λ_k is the k-th eigenvalue, representing the variance of the data along the direction $\mathbf{v}_k$,
- $\mathbf{v}_k$ is the corresponding eigenvector, indicating the principal direction of variation.

The eigenvector associated with the largest eigenvalue specifies the direction of maximum data variance and can be utilized to determine the position and orientation of the separating hyperplane.

3.1 Proposed SVM objective function

The separating hyperplane in SVM is fundamentally influenced by samples from opposing classes that lie close to each other. To better capture this critical region near the decision boundary, we focus on the set X_v of boundary-critical samples that are closest across classes. The direction of maximum dispersion within these samples is particularly informative for constructing a robust classifier.

By Singular Value Decomposition (SVD), one can show that the data points with normal distributions X_v lie inside of an ellipsoid whose greatest diagonal is the direction of maximum variance [14]. Moreover, we know that the eigenvector of covariance matrix corresponding to boundary points is in the same direction as the diagonals of the ellipsoid. In the process of generating principal components in Principal Component Analysis (PCA) [20], we also observe it [25].

Let the data points are inside of an ellipsoids whose diagonals are in the direction of eigenvectors $V = \{v_1, v_2, \ldots, v_d\}$ of covariance matrix Σ_v. Suppose that the separating hyperplane is almost parallel to the hyperplane formed by the eigenvectors $V \backslash \{v_i\}$, that means the eigenvector v_i is almost parallel to w and orthogonal to the given hyperplane. In other words, the vector w is derived by

$$\begin{aligned} w^* &= \arg\min_w \sum_{\substack{j=1 \\ j\neq i}}^{d} |w^T v_j| = \arg\min_w \frac{1}{2} \sum_{\substack{j=1 \\ j\neq i}}^{d} (w^T v_j)^2 = \arg\min_w \frac{1}{2} \sum_{\substack{j=1 \\ j\neq i}}^{d} (w^T v_j)(v_j^T w) \\ &= \arg\min_w \frac{1}{2} \sum_{\substack{j=1 \\ j\neq i}}^{d} w^T v_j v_j^T w = \arg\min_w \frac{1}{2} w^T B w, \end{aligned}$$

where $B = \sum_{\substack{j=1 \\ j\neq i}}^{d} v_j v_j^T$ is a symmetric, positive semi-definite matrix, ensuring the convexity of the objective function.

To address concerns about completely replacing the standard SVM objective, we instead propose a joint formulation that preserves the margin-maximizing term while incorporating the alignment with the principal boundary direction. The modified problem becomes:

$$\begin{aligned} \min_w \quad & \frac{1}{2}\|w\|^2 + \lambda w^T B w, \\ \text{subject to} \quad & y_i \left(w^T x_i + b\right) \geq 1, \quad \forall i, \end{aligned} \tag{5}$$

where $\|w\|^2$ promotes a large margin as in standard SVM, while $w^T B w$ tries to find the normal vector w such that $\langle w, v \rangle$ is minimized. Here, $\lambda > 0$ is a regularization parameter that controls the trade-off between maximizing the margin and minimizing $\langle w, v \rangle$.

Remark 1. *If the boundary-critical samples are almost equally distributed over all the features, the resulting separating hyperplane of the new approach will be much like that of the classical SVM. This implies that in these instances, the basic SVM already identified the dominant structure, and our method converges to the same solution.*

When data is not perfectly separable, slack variables $\xi_i \geq 0$ are introduced to allow violations, leading to the following problem:

$$\begin{aligned} \min_{w,\{\xi_i\}} \quad & \frac{1}{2}\|w\|^2 + \lambda w^T B w + C\sum_{i=1}^{n} \xi_i, \\ \text{subject to} \quad & y_i\left(w^T x_i + b\right) \geq 1 - \xi_i, \quad \forall i, \\ & \xi_i \geq 0, \quad \forall i. \end{aligned} \tag{6}$$

This formulation ensures that the model benefits from the geometric margin guarantees of classical SVM, while also aligning the decision boundary with the local structure captured by the top principal directions of the boundary-critical region.

To formalize the method, the following algorithm outlines the procedure for training the SVM with the proposed modification.

Algorithm 1: SVM with Boundary-Critical Samples

1: **Input:** Dataset $\{(x_i', y_i)\}_{i=1}^n$, Class labels $y_i \in \{-1, 1\}$.
2: **Step 1:** Use PCA (if needed) to reduce the dimensionality of the dataset and create a new dataset $\{(x_i, y_i)\}_{i=1}^n$.
3: **Step 2:** Compute the class means μ^+ and μ^- for Class $+1$ and Class -1, respectively.
4: **Step 3:** Select the set of boundary-critical samples S_v by (4).
5: **Step 4:** Construct matrix X_v from the selected boundary-critical samples.
6: **Step 5:** Compute the covariance matrix Σ_v.
7: **Step 6:** Extract d eigenvectors V of Σ_v.
8: **Step 7:** Find $v_i \in V$ such that v_i is almost orthogonal to separable hyperplane generated by standard SVM.
9: **Step 8:** Form $B = \sum_{\substack{j=1 \\ j\neq i}}^{d} v_j v_j^T$.
10: **Step 9:** Solve the proposed SVM objective Problem (5) to obtain the optimal weight vector w and bias b.
11: **Output:** Optimal hyperplane parameters w and bias b.

3.2 Discussion

The main innovation of the proposed method is its focus on critical-samples near the decision boundary, rather than utilizing the entire training dataset. By selecting samples near the boundary, the influence of noise or irrelevant points is reduced, ensuring that the classifier focuses on the important informational aspects of the data.

In addition, the use of the covariance matrix allows the local geometric structure around the decision boundary to be taken into account in the model, rather than simply maximizing the margin between classes in the entire dataset.

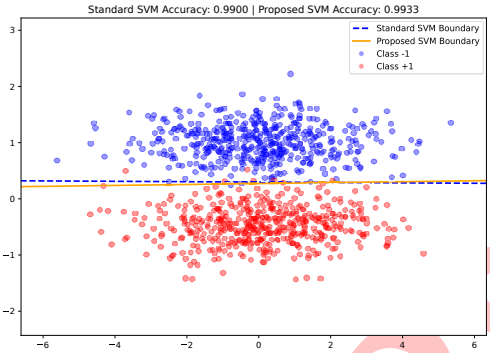

Figure 1: Classification in which the data of the two classes are on either side of the maximum dispersion.

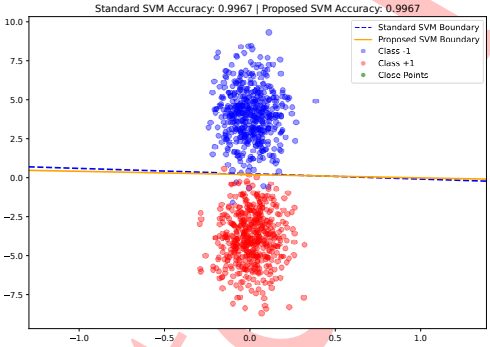

Figure 2: Classification in which the data of the two classes are on either side of the minimum dispersion.

In the case where the classes are located on both sides of the direction of maximum dispersion(s), the proposed method performs effectively by considering the eigenvector(s) (Figure 1). However, if the classes are on both sides of the minimum dispersion, the proposed method will give a separating plane almost the same as the standard SVM (Figure 2).

This is because the criterion of the proposed method is to find a normal vector that is as perpendicular (or close to perpendicular) to the direction of dispersion as possible. One note that the separating hyperplane is ultimately determined by the problem's constraints. The objective function tries to find a separating plane with the largest margin. If the dispersion is distributed almost equally in different directions, the resulting hyperplanes from the proposed method and the standard *SVM* are expected to be almost the same (Figure 3). From another perspective, one can first obtain the separating hyperplane from the standard SVM model and then consider the points whose distance from this plane is less than a small threshold as boundary samples (see Example 6). This method is conceptually more robust, since the definition of the boundary region is directly based on the model structure, not on the class means, which may not be a good representation in asymmetric data.

It is well known that if a dataset is normally distributed, then the data will contain inside of an ellipsoid, where the diameters of the ellipse are the eigenvectors of the covariance matrix of the dataset [23]. If the dispersions of the boundary data are the same in different directions, then the covariance matrix becomes approximately proportional to the identity matrix, i.e., $\Sigma \propto I$. In such a case, almost any vector can act as an eigenvector, since the data points are inside a spherical shape. In this case, it is better

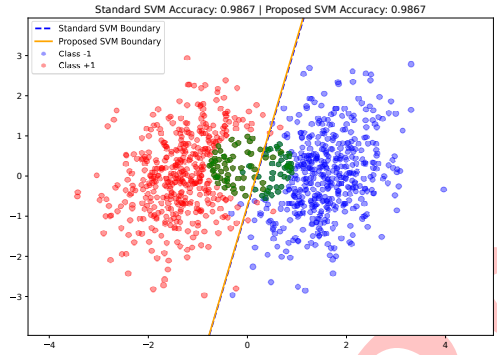

Figure 3: Classification in which dispersion is distributed almost equally in different directions for boundary points.

to reduce the percentage of boundary-critical samples or consider data from only one class to find the appropriate eigenvector.

Based on empirical observations and experiments, the proportion of boundary-critical samples is denoted by the hyperparameter r (where $0 < r \leq 1$). This parameter can be adjusted depending on the nature and complexity of the dataset, allowing flexibility in sample selection.

As discussed previously, the boundary data tend to lie within an ellipsoidal region, and therefore, the proposed method continues to perform reliably under such conditions. If the data are not distributed across a large number of clusters, then removing outliers can make this method perform well even when the data do not follow identical distributions. Nevertheless, there is another approach that always works, though at a higher computational expense. In this approach, one can first obtain the separating hyperplane from the standard SVM solution and then select the samples whose distance from this hyperplane is below a small threshold as boundary-critical samples. Example 6 illustrates this alternative approach.

Examples presented in the numerical results section demonstrate the effectiveness of the proposed method. Furthermore, by evaluating different values of r in the range of 0.01 to 0.25 for both synthetic and real datasets, it was observed that the model's performance remains relatively stable with respect to this hyperparameter. This analysis justifies the selected value of r in the experiments.

4 Experiments and results

To evaluate the effectiveness of the proposed method, a series of experiments are conducted on synthetic datasets with different feature dimensions. Both linearly separable and non-linearly separable cases are considered to analyze model performance under various scenarios. In addition to the synthetic datasets, the a9a dataset from the UCI repository, which is widely used to evaluate linear and non-linear SVM classifiers [9], as well as the Rice dataset [10, 28], are also used to evaluate the proposed method in real-world conditions.

All codes related to data analysis and graph generation are written using Python programming language. These codes use reliable libraries such as NumPy, pandas, matplotlib, scikit-learn, scipy, cvxpy. All tests are performed on an ASUS VivaBook (X513EQN) laptop with the following specifications: Intel Core i7 processor, 8GB RAM, 512GB SSD internal memory, and Windows 10 operating system.

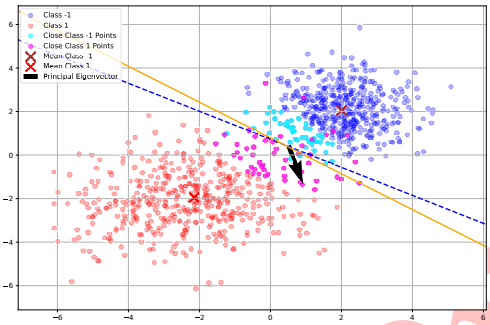

Figure 4: Decision boundaries and principal eigenvector for two-feature linear dataset

Data sets with two, three, and six features are generated. In non-linear scenarios, complex and non-linear decision boundaries are created by adding Gaussian noise to the data.

To comprehensively evaluate the performance of the proposed method, we compared it with two well-established baseline approaches in addition to the standard SVM: the ν-support vector machine and the Fuzzy SVM. The ν-SVM [30] offers a principled way to directly control the model complexity and error rate via the ν parameter, making it a robust geometric baseline close to classical SVM frameworks. On the other hand, the Fuzzy SVM is specifically designed to handle ambiguous or imprecise labels by incorporating Fuzzy membership values that fit well with the nature of uncertainty considered by our method. Therefore, comparing these two methods provides meaningful insights into how our approach performs relative to geometric margin-based and uncertainty-aware classification techniques.

For a more comprehensive evaluation, comparisons are made using 10-fold validation and the average classification accuracy along with its standard deviation is reported as a measure of the stability of the models' performance. Also, two different types of kernel functions were used to examine the behavior of the models under non-linear conditions.

In the evaluation process, the adjustment parameter λ is considered in the range $[0.1, 2]$ to examine the effect of aligning the weight vector with the dominant direction of the boundary. Also, in all experiments, the hyperparameter r is fixed at 0.1. In some experiments, to further evaluate the effect of this hyperparameter, its value is varied within the range 0.1 to 0.25, and the corresponding impact on the accuracy and stability of the proposed method was analyzed.

To ensure fairness and reproducibility across all experiments, the regularization parameter C is fixed at 1 for all models. In the non-linear settings, the polynomial kernel is used with a fixed degree of 3, the RBF kernel with $\gamma = 1$, and $\nu = 0.5$ for ν-SVM. These hyperparameters are selected uniformly for all models and datasets without additional tuning.

Example 1: In the first example, a synthetic dataset with two features and two classes is generated, where each class contains 500 samples with different variances (variance of class -1: 1.0, variance of class 1: 2.0), as shown in Figure 4. Purple points represent class -1, and red points represent class 1. The points highlighted in cyan correspond to the boundary samples selected with $r = 0.1$. Specifically, these include the samples of class 1 that are closest to the mean of class -1 (red cross), and the samples of class -1 that are closest to the mean of class 1 (brown cross). The main eigenvector, corresponding to the largest eigenvalue of the covariance matrix computed from these boundary samples, is shown in black in Figure 4. Figure 4 illustrates the two-feature dataset with the corresponding boundary points for both classes and the special eigenvector derived from the covariance of the selected boundary data.

Table 1: Comparison of classification accuracy (%) and standard deviation for different methods

Method	Mean Accuracy	Standard Deviation	Training Time(s)
Standard SVM	98.60	$\pm$ 0.0066	0.0021
Fuzzy SVM	98.50	$\pm$ 0.0055	0.0025
ν-SVM	97.60	$\pm$ 0.0058	0.0042
Proposed SVM	98.70	$\pm$ 0.0040	0.0032

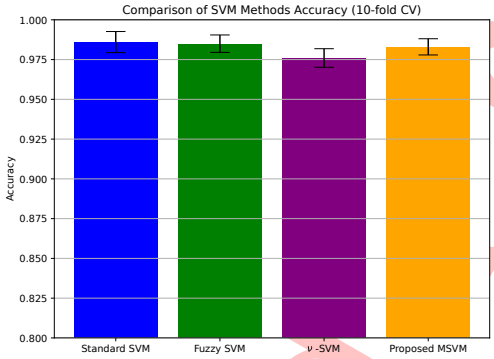

Figure 5: Comparison of mean classification accuracy and standard deviation for four methods: standard SVM, Fuzzy SVM, ν-SVM, and the proposed SVM

To compare the performance of the proposed method with baseline methods, classification accuracy and standard deviation values obtained from 10-fold cross-validation are reported in Table 1. The results cover four different methods: standard SVM, Fuzzy SVM, ν-SVM, and the proposed SVM. Each value represents the average accuracy along with its standard deviation, allowing both predictive performance and model stability to be assessed. According to the Table 1, the proposed method with an accuracy of 98.70% and a standard deviation of 0.0040 has shown the best performance in terms of accuracy and stability between iterations.

In comparison, the standard SVM and Fuzzy SVM methods have achieved accuracies of 98.60% and 98.50%, respectively, but have larger standard deviations, indicating higher fluctuations in their performance compared to the proposed method. The ν-SVM method has the lowest performance in terms of both accuracy (97.60%) and stability (standard deviation 0.0058).

These results show that the proposed method is able to generate a more accurate and stable decision boundary by using local statistical information in the boundary region. Also, its smaller standard deviation indicates greater robustness to training data changes, which is of great importance in practical applications.

As shown in Table 1, although the proposed SVM method requires slightly more training time compared to standard and Fuzzy SVM variants, the increase is marginal and well justified by its superior accuracy and robustness. The modest additional computational cost arises mainly from solving the convex optimization problem with the boundary-focused regularization term. This overhead remains minimal in practice, especially considering modern solver efficiencies and the critical performance gains achieved. Therefore, the proposed approach strikes an excellent balance between computational complexity.

Figure 5 shows a visual comparison of the classification accuracy of the proposed method with other methods.

Table 2 shows the classification accuracies of four SVM-based methods across different hyperpa-

Table 2: Classification accuracy (%) for different hyperparameter values in various SVM methods. Bold numbers indicate the best accuracy per method.

Method	Parameter Value	Mean Accuracy (%)
Standard SVM (C)	0.5	98.50
	0.8	98.50
	1.0	98.60
Fuzzy SVM (C)	0.5	98.50
	0.8	98.50
	1.0	98.50
Proposed SVM (C)	0.5	98.60
	0.8	98.64
	1.0	98.70
ν-SVM (ν)	0.3	97.60
	0.5	97.60
	0.7	97.40

Table 3: Sensitivity of classification accuracy (%) with respect to the value of r for the selected boundary-critical samples using 10-fold cross-validation (fixed $\lambda = 1.0$)

Percentage of Boundary Samples	Mean Accuracy	Standard Deviation
0.01	97.80	$\pm$ 0.0068
0.05	98.10	$\pm$ 0.0063
0.10	98.70	$\pm$ 0.0040
0.15	98.20	$\pm$ 0.0054
0.20	98.20	$\pm$ 0.0054
0.25	98.20	$\pm$ 0.0054

rameter values. The parameters tested include the regularization parameter C for Standard SVM, Fuzzy SVM, and the proposed SVM, as well as the parameter ν for ν-SVM. All results are averaged over 5-fold stratified cross-validation on the synthetic dataset. As seen in Table 2, the highest accuracies for standard SVM and proposed SVM are achieved at $C = 1.0$ while Fuzzy SVM performed consistently across all tested C values. The ν-SVM achieved its best result at $\nu = 0.5$. Overall, the proposed SVM slightly outperforms the other methods, demonstrating the effectiveness of incorporating the principal eigenvector regularization term. The results also indicate that the models are robust to small changes in hyperparameters within the tested ranges.

The effect of varying the hyperparameter r (representing the proportion of selected boundary samples) on the classification accuracy, measured using 10-fold cross-validation, is reported in Table 3. As can be seen in Table 3, the classification accuracy initially improves with increasing the hyperparameter r and reaches its maximum value of 98.70% at $r = 0.10$, with the lowest standard deviation of 0.0040. After that, the accuracy remains around 98.20% and does not change noticeably.

This behavior indicates that using an appropriate proportion of boundary samples (particularly around $r = 0.10$) provides sufficient and effective statistical information for learning, while higher values of r offer limited additional benefit and may increase computational complexity. Moreover, the low standard deviation in this region demonstrates better stability of the model performance against variations in the training data.

To evaluate the sensitivity of the proposed method to the regularization parameter λ, we varied its value from 0.1 to 2.0. This parameter controls the influence of the principal direction regularization term in the objective function. Table 4 reports the classification accuracy and standard deviation obtained via 10-fold cross-validation for each value.

Table 4: Sensitivity of classification accuracy (%) to the regularization parameter λ in the proposed SVM method (fixed $r = 1.0$)

Lambda (λ)	Mean Accuracy	Standard Deviation
0.1	98.10	$\pm$ 0.0054
0.4	98.00	$\pm$ 0.0063
0.7	98.00	$\pm$ 0.0063
1.0	98.70	$\pm$ 0.0072
1.3	98.00	$\pm$ 0.0063
1.6	97.90	$\pm$ 0.0083
1.9	97.70	$\pm$ 0.0090

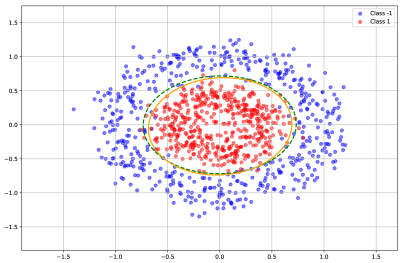

Figure 6: Decision boundaries and principal eigenvector for two-feature non-linear dataset

As can be seen in Table 4, the classification accuracy remains relatively stable for values less than $\lambda = 1.0$ and reaches its maximum value of 98.70% with a relatively low standard deviation at $\lambda = 1.0$. After this point, increasing the value of λ causes a gradual decrease in accuracy and an increase in the standard deviation.

This trend shows that introducing a regularization penalty for the main direction of the data in the objective function helps improve the performance of the model, but making the weight of this penalty too large may cause an inappropriate change in the direction of the decision boundary and reduce the generalization power of the model. Therefore, the choice of $\lambda = 1.0$, which is used in all experiments, provides a good balance between maintaining a large margin and being consistent with the data structure.

Example 2: In the second example, two non-linear two-feature datasets with 500 synthetic samples are created to evaluate further the proposed method for two classes with different variances. Class variance minus one = 0.5 and Class variance plus one = 0.1 for the first dataset (Figure 6) and Class variance minus one = 0.4, and class variance one = 0.9 for the second dataset (Figure 7). In Figures 6 and Figure 7, the blue points are class minus one, the red points are class one, the green dashed curve is the standard SVM separator, and the orange curve is the proposed SVM separator. Figure 6 shows the standard and proposed SVM implementations with the RBF kernel (defined in Section 2.3). Figure 7 shows the standard and proposed SVM implementations with the cubic polynomial kernel (defined in Section 2.3). After evaluating the models on the non-linear two-feature synthetic dataset, classification results for two different kernels — RBF and Polynomial (degree 3) — are reported. Table 5 summarizes the accuracies achieved by both the standard SVM and the proposed method.

The results clearly show that the proposed method consistently outperforms the standard SVM, albeit slightly, under both kernel settings. Particularly with the RBF kernel, the proposed method improved

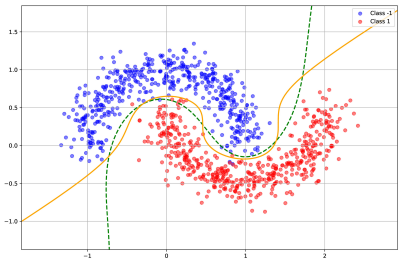

Figure 7: Decision boundaries and principal eigenvector for two-feature non-linear dataset

Table 5: Comparison of standard and proposed SVM models on a two-feature non-linear dataset using different kernels

Kernel	Standard SVM Accuracy (%)	Proposed SVM Accuracy (%)
RBF	95.33	95.67
Polynomial (degree 3)	99.33	99.67

classification accuracy from 95.33% to 95.67%, while with the Polynomial kernel of degree 3, it achieved a slight improvement from 99.33% to 99.67%. These findings highlight the robustness and adaptability of the proposed method across different kernel types.

Example 3: In this example, we created a synthetic dataset with three features, 500 samples per class, and linear separability. The variance for each class is different: the variance for class -1 was 0.9 and the variance for class 1 was 2.3.
The numerical results of four different approaches are compared, including the standard SVM, ν-SVM, Fuzzy SVM, and the proposed method. The performance of the models is evaluated using 10-fold cross-validation, and the average classification accuracy along with the standard deviation and training time for each method are reported in Table 6. As observed in Table 6, the proposed SVM achieves the highest mean accuracy with comparable stability (lowest standard deviation) compared to the other approaches. Although Fuzzy and ν-SVM also perform well, their accuracies are slightly lower than those of the standard and proposed SVM models.

To evaluate whether the observed differences in classification performance among the four methods are statistically significant, pairwise Paired t-tests are conducted based on the cross-validation results. The results are summarized in Table 7. These results demonstrate that the proposed SVM significantly outperforms Fuzzy SVM and ν-SVM. Importantly, its improvement over the standard SVM is not statistically significant (p-value > 0.05). This indicates that the proposed method should be regarded as a competitive alternative to the standard SVM, achieving comparable accuracy while offering additional robustness to boundary noise and asymmetric data distributions.

Example 4: To provide a more comprehensive evaluation of the classifiers on real-world data, we applied all four methods to the Rice dataset after performing preprocessing and PCA. To reduce computational complexity and potential overfitting, PCA is applied to project the data into a 6-dimensional subspace. The transformation retained approximately 99.9% of the total data variance, indicating that the most significant information was preserved during the dimensionality reduction process [7]. The classification performance is measured using three metrics: Accuracy, F1-score, and AUC-ROC, using

Table 6: Comparison of classification accuracy (%) and standard deviation for different methods (10-fold cross validation)

Method	Mean Accuracy (%)	Standard Deviation	F1-score (%)
Standard SVM	99.40	$\pm$ 0.0066	99.36
Fuzzy SVM	98.90	$\pm$ 0.0114	98.88
ν-SVM	98.30	$\pm$ 0.0149	98.25
Proposed SVM	99.42	$\pm$ 0.0065	99.38

Table 7: Paired t-test results between different SVM variants

Comparison	t	p-value	Significance
Fuzzy SVM vs Proposed SVM	-3.082	0.0131	Significant
ν-SVM vs Proposed SVM	-2.602	0.0287	Significant
Standard SVM vs Proposed SVM	2.3510	0.0985	Not Significant

10-fold cross-validation. The results are reported in Table 8.

The results in Table 8 indicate that all four methods achieve very similar performance on the Rice dataset. The proposed SVM attains the highest mean Accuracy (92.79%) and F1-score (92.83%), while standard SVM and Fuzzy SVM show nearly identical Accuracy and F1-score values. The ν-SVM exhibits slightly lower performance, but the differences across all metrics are minimal. Overall, these results suggest that all methods are comparably effective for this classification task, with no single method demonstrating a clear advantage.

To assess the sensitivity of the proposed model to the regularization alignment parameter λ, experiments are performed using values ranging from 0.1 to 2 with a fixed boundary sample ratio (10%). The classification accuracy and standard deviation over 10-fold cross-validation are reported in Table 9.

As shown in Table 9, the accuracy of the proposed method remains stable across a wide range of λ values. The highest accuracy (92.79%) is observed for $\lambda = 0.4$ and $\lambda = 1.0$, with only minor fluctuations in other values. This robustness confirms that the model is not highly sensitive to the choice of λ within this range.

To evaluate the impact of the selected boundary samples on model performance, we varied the hyperparameter r, which defines the proportion of closest samples from each class to the mean of the opposite class, in the range of 0.01 to 0.25, while keeping $\lambda = 1.0$ fixed. Table 10 presents the results in terms of classification accuracy and stability.

Table 10 shows that the choice of the hyperparameter r, i.e., the proportion of boundary samples, has a limited effect on the model's classification accuracy. The highest accuracy is achieved at $r = 0.10$, while performance remains nearly the same for other values. This indicates that the proposed model can generalize effectively even with a relatively small subset of discriminative samples from the boundary region.

Example 5: To further validate the proposed method on real-world data, we conducted experiments on the well-known a9a dataset (Adult dataset from UCI), which contains 32,561 training and 16,281 test instances with 123 features.

All methods are evaluated using 5-fold cross-validation on the entire a9a dataset. The comparison includes four classifiers: standard SVM, the proposed method, Fuzzy SVM, and ν-SVM. The average accuracy and standard deviation over five folds are reported in Table 11.

As shown in the table above, the proposed SVM requires a longer training time compared to the standard SVM and Fuzzy SVM methods. This increase in training time is due to the additional compu-

Table 8: Comparison of average accuracy, F1-score, and AUC-ROC (%) with standard deviations over 10-fold cross-validation on the Rice dataset

Method	Accuracy (%)	F1-score (%)	AUC-ROC (%)
Standard SVM	92.70 ± 1.00	92.75 ± 1.00	97.92 ± 0.00
Fuzzy SVM	92.70 ± 1.00	92.77 ± 1.00	97.90 ± 1.00
ν-SVM	92.55 ± 1.00	92.51 ± 1.00	97.86 ± 0.00
Proposed SVM	92.79 ± 1.00	92.83 ± 1.00	97.92 ± 0.00

Table 9: Effect of λ on accuracy (%) of the proposed model over 10-fold cross-validation on the Rice dataset (fixed $r = 0.1$)

λ	Accuracy (%)	Standard Deviation
0.1	92.73	$\pm$ 0.0116
0.4	92.79	$\pm$ 0.0115
0.7	92.76	$\pm$ 0.0120
1.0	92.79	$\pm$ 0.0112
1.3	92.73	$\pm$ 0.0116
1.6	92.70	$\pm$ 0.0112
1.9	92.76	$\pm$ 0.0120

Table 10: Effect of boundary data percentage on accuracy (%) of the proposed model over 10-fold cross-validation on the Rice dataset (fixed $\lambda = 1.0$)

Boundary Percentage	Accuracy (%)	Standard Deviation
0.01	92.70	$\pm$ 0.0116
0.05	92.70	$\pm$ 0.0116
0.1	92.79	$\pm$ 0.0112
0.15	92.72	$\pm$ 0.0112
0.20	92.72	$\pm$ 0.0112
0.25	92.72	$\pm$ 0.0112

tational steps involved in the proposed model, which includes convex optimization utilizing the principal boundary eigenvector.

However, the accuracy of the proposed model is slightly better than the other methods, with a similar standard deviation, indicating its stable performance.

In Table 12, by setting ($\lambda = 1.5$), the accuracy of the proposed method is better than the standard SVM. Moreover, considering the standard deviation of the mean accuracies, the proposed method demonstrates slightly better stability compared to the standard SVM.

We conducted experiments to compare the classification accuracy of four different methods: standard SVM, the Proposed Method, Fuzzy SVM, and ν-SVM. Each method is evaluated using multiple values of its key hyperparameters: C for SVM-based methods and ν for ν-SVM. The experiments are performed using 5-fold cross-validation on the dataset, and the mean accuracies for each parameter setting are recorded.

Following the results shown in Table 13, it is clear that both standard SVM and the proposed Method provide stable and high accuracy across the tested parameter values, indicating their robustness and suitability for this dataset. Fuzzy SVM shows slightly lower performance, which may be due to the weighting scheme applied to the samples. On the other hand, ν-SVM exhibits significantly lower accuracy for all tested (ν) values, suggesting that the current parameter settings may not be optimal for this dataset or that ν-SVM is less effective under these conditions.

Table 11: Comparison of classification accuracy (%) and standard deviation for different methods on the a9a dataset using 5-fold cross-validation

Method	Mean Accuracy (%)	Standard Deviation	Training Time(s)
Standard SVM	83.89	$\pm$ 0.0084	30.42
Fuzzy SVM	82.65	$\pm$ 0.0079	26.87
v-SVM	40.75	$\pm$ 0.0575	4.45
Proposed SVM	83.90	$\pm$ 0.0083	35.25

Table 12: Comparison of classification accuracy (%) and standard deviation for different methods on the a9a dataset using 5-fold cross-validation and $\lambda = 1.5$

Method	Mean Accuracy (%)	Standard Deviation
Standard SVM	83.89	$\pm$ 0.0084
Fuzzy SVM	82.65	$\pm$ 0.0079
v-SVM	40.75	$\pm$ 0.0575
Proposed SVM	83.92	$\pm$ 0.0083

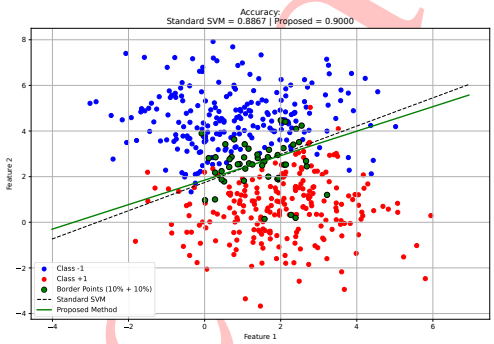

Figure 8: Boundary-critical points is selected based on distance to standard SVM decision boundary.

Example 6: In this example, the goal is to evaluate the effectiveness of the proposed boundary point selection strategy. Unlike the previous examples, where 10% of the samples closest ($r = 0.10$) to the mean of the opposite class are considered as boundary-critical points, here the boundary-critical points are defined geometrically based on the vertical distance from the separating hyperplane. Specifically, data whose vertical distance from the hyperplane obtained from the standard SVM is less than a certain small threshold (such as $\delta = 0.2$) are selected as boundary-critical points.

Using this new subset, the proposed model is retrained via 10-fold cross-validation, and its performance is compared with the original method. The results showed that the accuracy of the model did not decrease significantly, indicating the stability and robustness of the proposed method with respect to the type of boundary point definition (see Figure 8).

The results of the ten-fold validation are shown in Table 14.

The accuracy of the proposed method with this new boundary-critical points selection method is still higher than standard.

Table 13: Classification accuracy for different methods and parameter settings on the dataset

Method	Parameter	Mean Accuracy
Standard SVM	0.5	0.8388
	0.8	0.8388
	1.0	0.8389
Proposed Method	0.5	0.8389
	0.8	0.8387
	1.0	0.8390
Fuzzy SVM	0.5	0.8254
	0.8	0.8254
	1.0	0.8265
v-SVM	0.1	0.4015
	0.3	0.4033
	0.5	0.4175

Table 14: Selection of 10% percent of boundary-critical points based on the alternative method

Method	Mean Accuracy (%)	Standard Deviation
Standard SVM	88.67	$\pm$ 0.0120
Proposed SVM	90.67	$\pm$ 0.0100

5 Conclusion

In this study, a novel framework was proposed to enhance the performance of SVMs by redefining the objective function based on the geometric structure of data near the decision boundary. By focusing on critical samples that play a significant role in defining the separating hyperplane, the proposed approach effectively captured the underlying distribution of the data with greater precision.

Experimental results suggest that utilizing the principal eigenvector extracted from a carefully selected subset of samples can enhance classification performance in certain scenarios, particularly in non-linear settings and high-dimensional datasets. This highlights the importance of leveraging local boundary information rather than relying solely on the global distribution of all training samples.

The principal advantage of the proposed method lies in its ability to adapt to complex data structures and to reinforce decision boundaries against noise and overlapping samples. From a theoretical perspective, this approach provides a promising pathway toward geometry-aware learning models. Practically, the method shows shows promise in addressing challenges in real-world classification tasks.

Future research directions include extending the proposed method to multi-class problems, investigating its adaptability with various types of kernel functions, and applying it to complex real-world datasets, such as those encountered in medical diagnosis or financial forecasting.

Acknowledgement

The authors sincerely thank the anonymous reviewer for their valuable comments and suggestions, which helped improve the quality of this work.

References

[1] M.A. Nanda, K.B. Seminar, D. Nandika, A. Maddu, *A comparison study of kernel functions in the support vector machine and its application for termite detection*, Inf. **9** (2018) 5.

[2] A. Alhudhaif, *A non-linear optimization based robust attribute weighting model for the two-class classification problems*, PeerJ Comput. Sci. **9** (2023) e1598.

[3] N. Amaya-Tejera, M. Gamarra, J.I. Velez, E. Zurek, *A distance-based kernel for classification via support vector machines* , Front. Artif. Intell. **7** (2024) 1287875.

[4] P. Arabie, L. Hubert, G. De Soete, *Clustering and Classification,* World Science, 1996.

[5] L. Auria, R.A. Moro, *Support vector machines (SVM) as a technique for solvency analysis*, (2008). [Unpublished/working paper].

[6] C.J. Burges, *A tutorial on support vector machine for pattern recognition*, Data Min. Knowl. Discov. **2** (1998) 955–974.

[7] L.J. Cao, K.S. Chua, W.K. Chong, H.P. Lee, Q.M. Gu, *A comparison of PCA, KPCA and ICA for dimensionality reduction in support vector machine*, Neurocomputing **55** (2003) 321–336.

[8] J. Cervantes, F. Garcia-Lamont, L. Rodriguez-Mazahua, A. Lopez, *A comprehensive survey on support vector machine classification: Applications, challenges and trends*, Neurocomputing **408** (2020) 189–215.

[9] C.C. Chang, C.J. Lin, *LIBSVM: a library for support vector machines*, ACM Trans. Intell. Syst. Technol. **2** (2011) 27.

[10] I. Cinar, M. Koklu, *Classification of rice varieties using artificial intelligence methods*, Int. J. Intell. Syst. Appl. Eng. **7** (2019) 188–194.

[11] C. Cortes, V. Vapnik, *Support-vector networks*, Mach. Learn. **20** (1995) 273–297.

[12] N. Cristianini, J. Shawe-Taylor, *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press, 2000.

[13] A. Garg, D. Roth, *Margin distribution and learning*, Proc. 20th Int. Conf. Mach. Learn. (2003) 210–217.

[14] G.H. Golub, C.F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore, 1989.

[15] M. Gonen, E. Alpaydin, *Localized multiple kernel learning*, Proc. 25th Int. Conf. Mach. Learn. (2008) 352–359.

[16] G. Guo, S.Z. Li, K. Chan, *Face recognition by support vector machines*, Proc. 4th IEEE Int. Conf. Automat. Face Gesture Recogn. (2000) 196–201.

[17] E. Hafezieh, A. Tavakoli, M. Matinfar, *Strategies for disease diagnosis by machine learning techniques*, J. Math. Model. **11** (2023) 441–450.

[18] M.A. Hearst, S.T. Dumais, E. Osuna, J. Platt, B. Scholkopf, *Support vector machines*, IEEE Intell. Syst. **13** (1998) 18–28.

[19] R. Hu, X. Zhu, Y. Zhu, J. Gan, *Robust SVM with adaptive graph learning*, World Wide Web **23** (2020) 1945–1968.

[20] I.T. Jolliffe, *Principal component analysis for special types of data*, Springer, New York, 2002.

[21] S.A. Kasnavi, M. Aminafshar, M.M. Shariati, N.E.J. Kashan, M. Honarvar, *The effect of kernel selection on genome wide prediction of discrete traits by support vector machine*, Gene Rep. **11** (2018) 279–282.

[22] G. Khyathi, K.P. Indumathi, J. Hasin, S. Siluvai, G. Krishnaprakash, *Support vector machines: a literature review on their application in analyzing mass data for public health*, Cureus **17** (2025) e77169.

[23] D.P. Kroese, Z. Botev, T. Taimre, R. Vaisman, *Data Science and Machine Learning: Mathematical and Statistical Methods*, Chapman and Hall/CRC, 2019.

[24] C.F. Lin, S.D. Wang, *Fuzzy support vector machines*, IEEE Trans. Neural Netw. **13** (2002) 464–471.

[25] A. Mackiewicz, W. Ratajczak, *Principal components analysis (PCA)*, Comput. Geosci. **19** (1993) 303–342.

[26] D.A. Pisner, D.M. Schnyer, *Support Vector Machine, in: Machine Learning*, Academic Press, 2020.

[27] A. Pourmoezi, M. Eslami, A. Tavakoli, *A new insight on the model of support vector machine*, Comput. Sci. Eng. **4** (2025) 297–308.

[28] Rice (Cammeo and Osmancik) [Dataset]. (2019). UCI Machine Learning Repository. https://doi.org/10.24432/C5MW4Z.

[29] B. Scholkopf, A. Smola, K.R. Muller, *Nonlinear component analysis as a kernel eigenvalue problem*, Neural Comput. **10** (1998) 1299–1319.

[30] B. Scholkopf, A.J. Smola, R.C. Williamson, P.L. Bartlett, *New support vector algorithms*, Neural Comput. **12** (2000) 1207–1245.

[31] J.A. Sidey-Gibbons, C.J. Sidey-Gibbons, *Machine learning in medicine: a practical introduction*, BMC Med. Res. Methodol. **19** (2019) 64.

[32] M.H.N. Solhdar, M.J. Solahdar, S. Eskandarix, *An intrusion detection system with a parallel multilayer neural network*, J. Math. Model. **9** (2021) 437–450.

[33] L. Zhou, S. Pan, J. Wang, A.V. Vasilakos, *Machine learning on big data: Opportunities and challenges*, Neurocomputing **237** (2017) 350–361.

[34] F. Zhu, N. Ye, W. Yu, S. Xu, G. Li, *Boundary detection and sample reduction for one-class support vector machines*, Neurocomputing **123** (2014) 166–173.