International Journal of Maps in Mathematics

*Volume **9**, Issue **1**, 2026, Pages:**141–159***

E-ISSN: 2636-7467

www.simadp.com/journalmim

FS-GKM: A CLUSTERING-DRIVEN FEATURE SELECTION FRAMEWORK FOR ENHANCED SUPERVISED LEARNING PERFORMANCE

DUYGU SELIN TURAN *

Abstract. Feature selection is a fundamental process in machine learning that involves identifying and selecting the most informative features from a dataset to be used in model training. This step is crucial, as it can significantly enhance model performance, mitigate overfitting, reduce computational complexity, and improve the interpretability of the resulting models. By eliminating irrelevant or redundant features, feature selection facilitates the development of more efficient and accurate predictive models. Therefore, the development of new feature selection algorithms is of great importance. In this study, a novel feature selection algorithm—Feature Selection using Global k-Means (FS-GKM)—is proposed. In this study, a novel feature selection algorithm—Feature Selection using Global k-Means (FS-GKM)—is proposed. The method leverages the global k-means clustering algorithm to group features based on their similarity. Subsequently, the algorithm assesses the discriminative power of features by analyzing class density distributions within each cluster. This clustering-based evaluation enables the identification of features that contribute most significantly to class separability. To validate the effectiveness of the FS-GKM method, a comprehensive experimental analysis was conducted using 13 benchmark datasets. Each dataset was subjected to dimensionality reduction through 13 different feature selection techniques, and the resulting feature subsets were evaluated using four distinct classifiers. The proposed FS-GKM algorithm achieved superior performance in 40 out of 52 comparative cases, demonstrating its robustness and effectiveness across diverse scenarios.

Keywords: Feature selection, Classification, Clustering, Preprocessing.

MSC (2020): 68T05, 68T09, 68T10.

1. Introduction

Feature selection plays a vital role in machine learning and data processing by significantly improving the efficiency, scalability, and precision of predictive models, particularly when dealing with large-scale or high-dimensional datasets. By identifying and retaining the most informative features while eliminating irrelevant or redundant ones, feature selection reduces model complexity and computational burden, which is especially critical in text

Received: 2025.11.24 *Revised: 2026.01.08* *Accepted: 2026.01.18*

* Corresponding author

Duygu Selin Turan (duygu.selin.balli@ege.edu.tr) ⋄ ORCID: 0000-0001-9881-6013

mining and vector space modeling applications [26]. This reduction not only improves classification accuracy but also mitigates overfitting, shortens training time, and enhances model interpretability. From a broader perspective, feature selection can be viewed as a search or optimization problem, which becomes increasingly challenging as data dimensionality and volume grow, as highlighted in big data bioinformatics studies [30]. Moreover, in complex real-world systems such as industrial control systems and network intrusion detection, effective feature selection is essential to handle high-dimensional traffic data, remove redundancy, and improve detection efficiency and robustness [7]. Consequently, feature selection is not merely a preprocessing step but a fundamental component for building robust, generalizable, and computationally efficient models capable of handling large and complex datasets.

Several approaches have been developed for feature selection, including filter methods, wrapper methods, and embedded methods. Filter methods evaluate each feature individually, typically using statistical measures, but often fail to consider interactions between features. Wrapper methods assess subsets of features based on model performance and tend to yield higher accuracy; however, they are computationally expensive. Embedded methods perform feature selection during the model training process itself—examples include LASSO and decision trees—offering a good balance between computational efficiency and predictive performance. Nevertheless, each technique comes with its own limitations, such as the overfitting risk in wrapper methods or limited flexibility in filter-based approaches [16].

A variety of feature selection techniques have been proposed in the literature. RELIEF ranks features based on their ability to distinguish between instances of different classes, taking feature interactions into account. While effective for complex datasets, it can be computationally intensive due to repeated nearest-neighbor searches [15]. ANOVA (Analysis of Variance) assesses the importance of features by comparing the means across different classes. It ranks features based on the variance ratio, making it suitable for identifying features that contribute significantly to class separation [25].

The FARNeMF (Feature Adaptive Robust Non-Negative Matrix Factorization) algorithm is designed for feature selection and dimensionality reduction, particularly in high-dimensional datasets. By accounting for noise and outliers, FARNeMF ensures a more robust selection of features. It extends Non-Negative Matrix Factorization (NMF) to enhance the efficiency of feature extraction [12]. Principal Component Analysis (PCA) is another widely used technique for dimensionality reduction in high-dimensional datasets. It transforms data into a set of uncorrelated principal components, simplifying the dataset while retaining most of its variance. Although PCA does not directly select features, it effectively reduces redundancy and enhances computational efficiency while preserving critical information [1].

The Laplacian Score is a filter-based method that evaluates feature importance based on its ability to preserve locality, independent of a learning algorithm. It can be applied in both supervised and unsupervised settings, and has demonstrated superior performance compared to data variance and the Fisher score in several experiments [11]. Recursive Feature Elimination (RFE) improves model performance by iteratively removing the least important features based on model-assigned importance scores. This process continues until only the most informative features are retained, making RFE particularly useful for high-dimensional classification and regression tasks [17].

GBNRS is a novel feature selection approach based on rough set theory, specifically designed to handle continuous data without the need for membership functions or parameter tuning. By incorporating an adaptive neighborhood radius selection mechanism, GBNRS improves attribute reduction and outperforms the state-of-the-art NRS algorithm in classification accuracy [31]. GRRS (Granular-Rectangular Rough Set) offers another innovative approach that avoids distance metrics altogether. Instead, it divides space into granular-rectangular regions and constructs the neighborhood radius based on hierarchical spatial relationships. This method enhances accuracy and efficiency, especially in cases where traditional distance measures are ineffective [32].

To identify a small subset of genes from large-scale gene expression data collected via DNA microarrays, Guyon et al. proposed a gene selection method based on Support Vector Machines (SVM) in combination with RFE. Their approach aims to improve classification performance while also identifying genes with biological relevance to cancer. Experimental results show that their method outperforms traditional correlation-based techniques [10].

The Generalized Fisher Score extends the traditional Fisher score by jointly selecting features that maximize a lower bound of the Fisher criterion. This results in a mixed-integer programming problem, which is solved using a cutting plane algorithm that alternates between multivariate ridge regression and projected gradient descent. The method demonstrates improved performance over both the standard Fisher score and other state-of-the-art approaches [8].

ILFS (Infinite Latent Feature Selection) introduces a robust probabilistic, graph-based algorithm that ranks features by modeling all possible subsets as paths on a graph. By treating feature relevance as a latent variable in a PLSA-inspired generative model, it avoids the combinatorial explosion of subset evaluation. Extensive benchmarking shows that ILFS outperforms eleven state-of-the-art methods across diverse datasets [21].

The method proposed in [29] addresses high-dimensional generalized linear models with Lipschitz loss functions by establishing a non-asymptotic oracle inequality for empirical risk minimization under a LASSO penalty. This penalty is applied to the model coefficients, normalized by their empirical norm. The method is demonstrated through logistic regression, density estimation, classification with hinge loss, and least squares regression tasks. Infinite Feature Selection (Inf-FS) models feature subsets as paths in a graph, where edges encode pairwise feature relationships. It employs matrix power series and Markov chain principles to evaluate paths of arbitrary lengths, ranking features based on relevance and redundancy. An unsupervised variant is also proposed, and experiments show that Inf-FS outperforms 18 state-of-the-art methods across a variety of benchmark datasets [22].

This paper introduces the FS-GKM method, which is based on the Global k-Means clustering algorithm. The primary objective of FS-GKM is to identify the most discriminative features within a dataset. In this approach, the instances in the dataset are kept fixed, and each feature is individually evaluated by clustering its values using the global k-means algorithm. This algorithm addresses the sensitivity of standard k-means to the initial selection of cluster centers. The number of clusters k is set equal to the number of classes in the dataset. After clustering, the class distributions within the resulting clusters are analyzed. A feature

is retained if any cluster contains a number of instances from a specific class exceeding a predefined threshold; otherwise, it is discarded.

The remainder of this paper is organized as follows: Section 2 reviews related work in the field. Section 3 provides a detailed description of the proposed FS-GKM method. Section 4 presents the datasets used in the experiments, outlines the four classification algorithms and performance metrics employed, and discusses the experimental results. Finally, Section 5 concludes the paper with a summary of findings and suggestions for future research.

2. Related Work

Categories of feature selection methods are presented in this section.

2.1. **Filter Methods.** Filter methods evaluate the importance of features based on statistical measures independently of any machine learning model. These methods are computationally efficient and scalable, making them ideal for large datasets. For instance, ANOVA (Analysis of Variance) is a commonly used filter method that assesses the relationship between categorical target variables and numerical features by analyzing variance. Other examples include correlation coefficients, which measure linear relationships, and chi-square tests, which evaluate the independence of categorical features. While filter methods are fast and easy to implement, they often overlook interactions between features, limiting their effectiveness in complex datasets. Research suggests that filter methods are best suited for preliminary feature selection before applying more advanced techniques [9].

2.2. **Wrapper Methods.** Wrapper methods take a distinct approach by using a machine learning model to assess the performance of various feature subsets. These methods are iterative and can be computationally costly, as they involve training and evaluating models multiple times. A popular example is RFE (Recursive Feature Elimination), which systematically eliminates the least important features based on model coefficients or feature importance scores. Other wrapper methods include forward selection, which adds features one at a time, and backward elimination, which removes them sequentially. Although wrapper methods often lead to better model performance by accounting for feature interactions, they are not suitable for very large datasets due to their computational demands. Research indicates that wrapper methods are most effective when sufficient computational resources are available and optimizing model performance is a priority [14].

2.3. **Embedded Methods.** Embedded methods integrate feature selection directly into the model training process, offering greater efficiency compared to wrapper methods. These techniques are specific to the model being used and often leverage regularization to penalize less important features. For example, Lasso Regression (L1 Regularization) shrinks the coefficients of less relevant features to zero, effectively performing feature selection. Similarly, tree-based models such as decision trees and random forests naturally rank features based on their importance in splitting nodes. Embedded methods offer a good balance between computational efficiency and model performance, as they account for feature interactions during training. However, their reliance on particular algorithms can limit their adaptability to other models. Research indicates that embedded methods are especially beneficial for high-dimensional datasets, where computational efficiency is a key consideration [27].

2.4. **Unsupervised Methods.** Unsupervised methods are applied when there is no target variable, making them ideal for exploratory data analysis and dimensionality reduction. These techniques focus on transforming or selecting features based on their inherent properties rather than their relationship with a target. A key example is PCA (Principal Component Analysis), which reduces dimensionality by generating new, uncorrelated variables (principal components) that capture the maximum variance in the data. Other unsupervised methods include Independent Component Analysis (ICA) and clustering-based feature selection. While unsupervised methods effectively reduce noise and redundancy, they may not always align with the features' predictive power in supervised tasks. Research suggests that PCA is especially useful for preprocessing high-dimensional data before applying supervised learning methods [13].

2.5. **Hybrid Methods.** Hybrid methods combine the advantages of both filter and wrapper techniques to strike a balance between computational efficiency and model performance. These approaches typically use filter methods to initially reduce the feature space, then apply wrapper methods for further refinement. For instance, genetic algorithms can be guided by filter-based scores to identify the optimal feature subset. Another strategy is ensemble feature selection, where results from multiple methods are combined to enhance robustness. Hybrid methods are particularly valuable for large datasets, where pure wrapper methods would be too computationally expensive. However, they can be complex to implement and require careful parameter tuning. Research indicates that hybrid methods hold great potential for feature selection, particularly in fields like bioinformatics and text mining [24].

2.6. **Domain Knowledge-Based Methods.** Domain knowledge-based methods rely on the insights of experts to identify features known to be significant in a particular field. These approaches often involve feature engineering, where new features are constructed based on domain-specific knowledge. For instance, in healthcare, factors like patient age, medical history, and laboratory test results might be chosen based on clinical expertise. While these methods can produce highly interpretable and contextually relevant features, they can also be subjective and may not easily transfer across different domains. Furthermore, they necessitate collaboration between domain specialists and data scientists, which can be resource-intensive. Research suggests that domain knowledge-based methods are most effective when integrated with data-driven techniques, ensuring both practical relevance and statistical robustness [18].

2.7. **Stability-Based Methods.** Stability-based methods focus on identifying features that remain consistently important across different subsets of data or models. These techniques aim to reduce variability in feature selection results, which can be influenced by random fluctuations in the data. One example is stability selection, which uses subsampling and regularization to identify features consistently selected in multiple iterations. Another approach is bootstrap aggregating (bagging), which combines feature importance scores from multiple bootstrap samples. Stability-based methods are especially beneficial for enhancing the reliability of feature selection in high-dimensional datasets. However, they can be computationally demanding and may require careful parameter tuning. Research suggests that stability-based

methods are a robust feature selection approach, particularly when reproducibility is crucial [20].

3. The Proposed Algorithm

In this section, a new clustering-based approach for feature selection is proposed. The core idea of the proposed algorithm is to cluster the features using the global k-means algorithm and select those that effectively distinguish classes based on class densities within the clusters.

3.1. **The Global k-Means Algorithm.** Various algorithms have been developed to address the clustering problem, with the incremental k-means algorithm representing a notable improvement over the traditional k-means method [19]. Unlike classical k-means, which is highly sensitive to the initial choice of cluster centers and often converges to local minima, the incremental k-means algorithm incrementally constructs clusters by systematically evaluating candidate centers. This stepwise approach improves stability and increases the likelihood of reaching a globally optimal solution. Moreover, its ability to dynamically update clusters as new data points are introduced makes it particularly suitable for large-scale and evolving datasets. The pseudocode for the incremental k-means algorithm—given a training set $X^1, \ldots, X^k$ and a predefined number of clusters k—is presented in 1 [28]. The pseudocode of the global k-means algorithm is presented in Algorithm 1.

Algorithm 1 Global k-means

Input: Data set $x_1, \ldots, x_m$, Number of clusters k
Output: Cluster centers $x^1, \ldots, x^k$

1: Initialize the first cluster center as $c^1 \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$;
2: Set $t \leftarrow 1$;
3: **while** termination condition $t > k$ is not met **do**
4: Set $p \leftarrow 1$;
5: **while** $p \leq m$ **do**
6: Run the k-means algorithm with initial centers $(c^1, \ldots, c^t, x_p)$;
7: **if** the clustering result improves upon the previous one **then**
8: Update cluster centers: $x^i \leftarrow c^i$ for $1 \leq i \leq t-1$ and $x^t \leftarrow c^{t+1}$;
9: **end if**
10: $p \leftarrow p + 1$;
11: **end while**
12: Update $c^i \leftarrow x^i$ for $1 \leq i \leq t$ and increment $t \leftarrow t + 1$;
13: **end while**
14: **Return** final cluster centers $x^1, \ldots, x^k$;

The global k-means algorithm aims to obtain an optimal clustering configuration at each iteration. It starts by calculating the mean of all data points to establish the first cluster center, initializing $t = 1$. In each subsequent iteration, t is incremented, and every data point is evaluated as a potential candidate for the tth cluster center. During Step 7, the algorithm seeks to minimize the total distance between data points and their assigned centers, retaining the configuration that yields the lowest overall error for the current t-cluster setup. This iterative process continues until the desired number of clusters is reached, at which point the algorithm converges to a final clustering solution.

3.2. **Feature Selection Using the Global k-Means Algorithm.** The FS-GKM algorithm is introduced in this section. It consists of two main stages. In the first stage, each feature is clustered, and the class distributions within the resulting clusters are analyzed. In the second stage, the discriminative power of each feature is evaluated against a predefined threshold. Features exceeding this threshold are deemed informative and included in the selected feature set; those falling below are discarded. The pseudocode of the FS-GKM algorithm is presented in Algorithm 2.

Algorithm 2 Feature Selection Using the Global k-Means Algorithm

Input: Data set D, feature set $A = \{a_1, a_2, \ldots, a_m\}$, threshold r, number of clusters k
Output: Selected feature set B
Initialize $B \leftarrow \emptyset$
for $i \leftarrow 1$ to m **do**
  Cluster the ith feature into k clusters using Algorithm 1
  for $j \leftarrow 1$ to k **do**
    for $s \leftarrow 1$ to k **do**
      Count the number of data points in cluster j that belong to class s, denote this as *count*
      if *count* $\geq r$ **then**
        $B \leftarrow B \cup \{a_i\}$
      end if
    end for
  end for
end for
Return the selected feature set B

The algorithm takes as input an $m \times n$ data set D, the feature set A, a threshold r used to evaluate class distribution within clusters, and the number of clusters k. Initially, the selected feature set B is empty. Each feature is clustered into k clusters, where k corresponds to the number of classes in the data set. The class distribution within each cluster is then analyzed. If, for any cluster, the number of data points belonging to a class exceeds the threshold r, the corresponding feature is added to the selected set B.

3.3. **Complexity of the FS-GKM Algorithm.** The time complexity of the global k-means algorithm is generally higher than that of standard k-means, typically estimated as $O(n \times k^2 \times m)$, where n denotes the number of data points, k is the number of clusters, m represents the number of features per data point, and t is the number of iterations per k-means run. This increased complexity arises from the algorithm's iterative procedure of adding one cluster center at a time and exhaustively evaluating each data point as a potential new centroid. At each step, a full k-means clustering is performed to optimize cluster assignments, significantly increasing computational cost compared to standard k-means. Although the global k-means offers advantages such as reduced sensitivity to initial centroid placement and a higher likelihood of reaching a global optimum rather than local minima, it does so at the expense of considerably higher computational demands.

In the FS-GKM algorithm, the global k-means is executed m times, where m is the number of features. However, in each run, only a single feature is considered (i.e., dimensionality is 1). Therefore, the time complexity of FS-GKM can be approximated as $O(n \times m \times t \times k^4)$.

TABLE 4.1. Brief description of the datasets

Dataset	Number of Instances	Number of Features	Number of Classes
iris	150	4	3
hepatitis	155	19	2
wine	178	12	3
heart1	270	13	2
iono	351	34	2
horse	368	23	2
wdbc	569	30	2
credit	690	15	2
german	1000	19	2
digits	1797	64	10
wifiLocalization	2000	8	4
madelon	2000	500	2
htru2	17898	8	2

4. EXPERIMENTS

In this section, we present a comparison of the proposed algorithm with other widely used or state-of-the-art feature selection algorithms on 13 benchmark datasets. The datasets were selected from the UCI Machine Learning Repository [6]. A description of these datasets is provided in Table 4.1. Thirteen feature selection algorithms were used for comparison: PCA [1], RFE [17], ANOVA [25], FARNeMF [12], GBNRS [31], CFS [10], Fisher [8], ILFS [21], LASSO [29], Laplacian [29], Inf-FS [22], GRRS-N, and GRRS-1 [32]. We utilized the results obtained by these algorithms as presented in [32]. The FS-GKM algorithm, CFS, Fisher, ILFS, Laplacian, LASSO, Inf-FS, RFE, PCA, and ANOVA were implemented in Python using the Feature Selection Code Library (FSLib) and scikit-learn. The FARNeMF algorithm was run with the neighborhood radius varying from 0.01 to 0.5 in increments of 0.01. The GBNRS algorithm was executed ten times, and the optimal result was retained. The GRRS algorithm was executed with the neighborhood radius varying from 1 to 5 in increments of 1. For the proposed algorithm under test, the threshold value r was considered with increments of 10, ranging from 50 to 90, and the optimal result was retained. For the global k-Means algorithm in FS-GKM, the number of clusters k was set equal to the number of classes in each dataset.

4.1. **Classifiers and Performance Measure.** The FS-GKM algorithm was evaluated using four classifiers: SVM, CART, KNN, and MLP. Support Vector Machines (SVM) is a supervised learning algorithm primarily used for classification tasks, aiming to identify the optimal hyperplane that maximizes the margin between data points of different classes in high-dimensional spaces. A notable advantage of SVM is its ability to handle non-linear data efficiently through the kernel trick, which transforms the data into a higher-dimensional space to facilitate separation. However, SVM can be computationally demanding, especially with large datasets, which may limit its suitability for time-sensitive applications [4].

Classification and Regression Trees (CART) is a decision tree algorithm that partitions data into subsets by evaluating feature values, constructing a tree-like structure applicable to both classification and regression. Its main strengths lie in simplicity and interpretability, as the resulting tree can be easily visualized and understood, making the decision-making process transparent. However, CART models are prone to overfitting when the tree grows too deep and may struggle with noisy data or imbalanced class distributions [2].

TABLE 4.2. Accuracy results for all threshold values on the credit dataset

Threshold Value	Accuracy (SVM)	Accuracy (CART)	Accuracy (MLP)
50	0.84	0.77	0.77
60	0.86	**0.86**	**0.80**
70	0.86	0.81	0.77
80	**0.87**	0.84	0.77
90	0.84	0.83	0.78

Multilayer Perceptron (MLP) is a type of artificial neural network composed of multiple layers of interconnected neurons, trained using the backpropagation algorithm. It excels at modeling complex patterns and relationships in data, making it particularly effective for large datasets. However, MLPs require substantial computational resources during training, are susceptible to overfitting, and demand careful tuning of hyperparameters, which can be complex and time-consuming [23].

K-Nearest Neighbors (KNN) is a simple, instance-based algorithm commonly used for both classification and regression. It classifies a data point based on the majority class among its k closest neighbors in the feature space. The main advantage of KNN lies in its simplicity and effectiveness in low-dimensional spaces with minimal training. However, KNN can be computationally intensive during prediction, especially with large datasets, due to the need to compute distances to all training instances [5].

Accuracy, a widely used metric for evaluating machine learning algorithms, was employed to compare the algorithms in this study. It is calculated as the ratio of correctly predicted instances to the total number of instances, as shown in equation 4.1. In this context, True Positive (TP) refers to instances that are correctly predicted as positive, False Positive (FP) refers to instances that are incorrectly predicted as positive, True Negative (TN) refers to instances that are correctly predicted as negative, and False Negative (FN) refers to instances that are incorrectly predicted as negative. TP and TN indicate correct predictions, while FP and FN represent incorrect predictions [3].

$$acc = \frac{\text{TP+TN}}{\text{TP+FP+FN+TN}} \times 100 \tag{4.1}$$

Higher accuracy indicates better model performance and greater reliability.

4.2. **Selection of the Threshold Value for FS-kM and the Number of Nearest Neighbors for KNN.** The FS-GKM algorithm is sensitive to the input threshold value r. Depending on this threshold, the selected feature set B varies. Therefore, FS-GKM was executed for threshold values $r = 50, 60, 70, 80$, and 90. For each threshold value, a feature set B is obtained, resulting in five different B sets. Classification is then performed using all these sets, and the best result among them is considered the final accuracy. The best result among all threshold values was considered the final accuracy. The accuracy results for all threshold values on the credit dataset are given in Table 4.2. The best results are highlighted in bold.

This procedure was repeated for all datasets, and the highest accuracy values were obtained at a threshold of $r = 50$ for the German, Heart1, HTRU2, Madelon, WDBC, and Iris datasets; at $r = 60$ for the Horse and Wine datasets; at $r = 70$ for the Iono, WiFiLocalization, and

TABLE 4.3. Accuracy results of the KNN algorithm for different k values on the credit dataset

Number of Nearest Neighbors k	Threshold Value r				
	50	60	70	80	90
1	0.67	0.65	0.66	0.71	0.74
2	0.67	0.71	0.65	0.66	0.72
3	0.70	0.70	0.68	0.73	0.77
4	0.74	0.74	0.71	0.73	0.75
5	0.72	0.77	0.71	0.75	0.79
6	0.74	0.75	0.72	0.74	0.79
7	0.71	0.73	0.75	0.76	0.79
8	0.72	0.74	0.73	0.78	0.79
9	0.70	0.72	0.72	0.77	**0.80**
10	0.71	0.73	0.73	0.77	0.79

Digits datasets; and at $r = 80$ for the Hepatitis and Credit datasets. The classifiers were run on the datasets obtained from the features selected using the specified threshold values.

The performance of the k-Nearest Neighbors (KNN) algorithm depends heavily on the choice of k, the number of nearest neighbors used for classification or regression. A small k can make the model sensitive to noise and outliers, resulting in overfitting, while a large k smooths the decision boundary and may cause underfitting. Selecting an appropriate k balances bias and variance. In this study, k was tested for values from 1 to 10, and the best accuracy was retained, highlighted in bold in Table 4.3.

4.3. **Computational Results.** Computational experiments are carried out on a laptop with Intel(R) Core(TM) i7-8550U CPU 1.80 GHz and RAM 8 GB. To illustrate the performance of the FS-GKM algorithm, the performance of the algorithm is compared with other 13 feature selection algorithms presented in Section IV using three classifiers in Subsection 4.1. Accuracy as a performance measure in Subsection 4.1 is used for comparison. The results are given in Tables 4.4, 4.5, 4.6, 4.7, 4.9, 4.10, 4.11 and 4.12. In the tables, we report the accuracy values and the best results obtained by the algorithms are highlighted using bold font.

In [32], the first 10 feature selection algorithms listed in Tables 4.4, 4.5, 4.6, 4.7 were applied to the 10 datasets given in the same table. The SVM, MLP, Cart, and KNN classifiers were applied on the reduced datasets. The classification accuracy values from [32] are listed in Tables 4.4, 4.5, 4.6, 4.7.

The SVM classifier under consideration five-fold cross-validation was then executed on the new datasets, consisting of the features selected by the feature selection methods for each dataset, and the accuracy values are shown in the Table 4.4. The accuracy values in the table show that, with the exception of the two datasets named Horse and Madelon, the FS-GKM algorithm outperformed all others in the remaining datasets.

The Cart classifier under consideration five-fold cross-validation was executed on the reduced datasets, consisting of the features selected by the feature selection methods for each dataset, and the accuracy values are shown in the Table 4.5. With the exception of the Hepatitis and Madelon datasets, the best accuracy value was obtained by running the CART classifier on the reduced datasets using the FS-GKM algorithm.

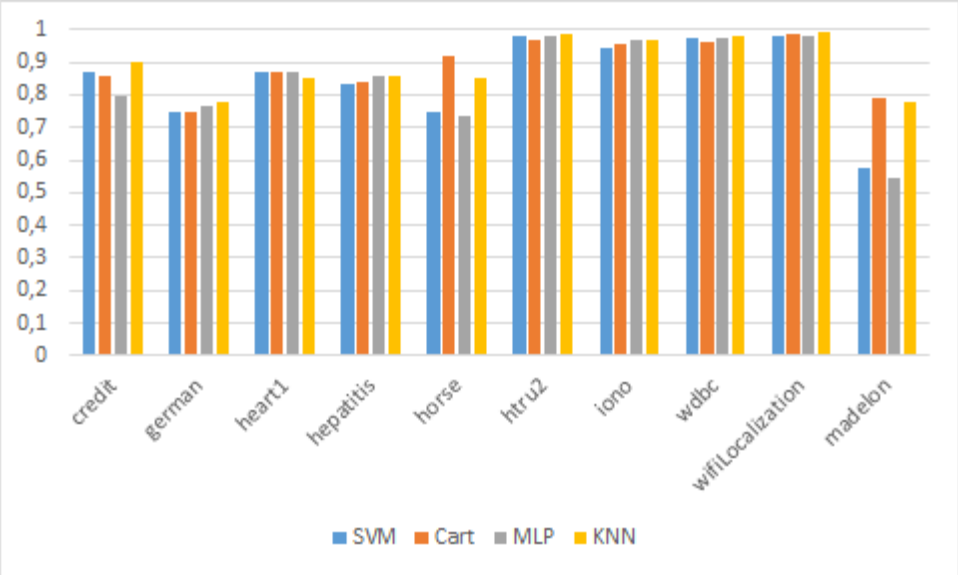

FIGURE 1. Comparison of classifiers for the FS-GKM

The MLP classifier was evaluated using five-fold cross-validation on the reduced datasets, which were composed of features selected by the feature selection methods for each dataset. The corresponding accuracy values are presented in Table 4.6. When the data in the table is examined, the FS-GKM algorithm has overperformed in 70% of the problems.

The KNN classifier was assessed using five-fold cross-validation on the reduced datasets, which consisted of features selected by the feature selection methods for each dataset. The resulting accuracy values are shown in Table 4.7. Upon reviewing the data in the table, it is evident that the FS-GKM algorithm outperformed the others in 80% of the cases.

The graph of accuracy values with four different classifiers, obtained by classifying the reduced datasets using the FS-GKM algorithm, is shown in Figure 1. According to the graph, KNN achieved the highest accuracy in 80% of the cases.

Since the proposed algorithm generates different reduced datasets for different threshold values, it is sensitive to the threshold. Figure 2 visualizes the percentage decrease in the number of features in the dataset for various threshold values. Upon examining the graph, it is evident that as the threshold value increases, the number of features decreases.

In addition to the feature selection methods in [32], the FS-GKM algorithm was compared with the ANOVA, PCA, and RFE feature selection methods, using SVM, MLP, and CART classifiers, on the digits, wine, iris, and wdbc datasets from the Table 4.1. For the ANOVA, RFE, and PCA feature selection algorithms, the number of features to be selected was specified as input. To determine this, the FS-GKM algorithm was first run on the datasets with threshold values of 50, 60, 70, 80, and 90. Based on the result with the highest accuracy, the number of features selected by the FS-GKM algorithm was then used as input for the ANOVA, RFE, and PCA algorithms. The number of features of the original datasets and the number of features selected after the FS-GKM are presented in Table 4.8.

The SVM, MLP, and CART classifiers were trained and tested on the obtained reduced datasets using a 33% percentage split, and accuracy values were obtained. The results

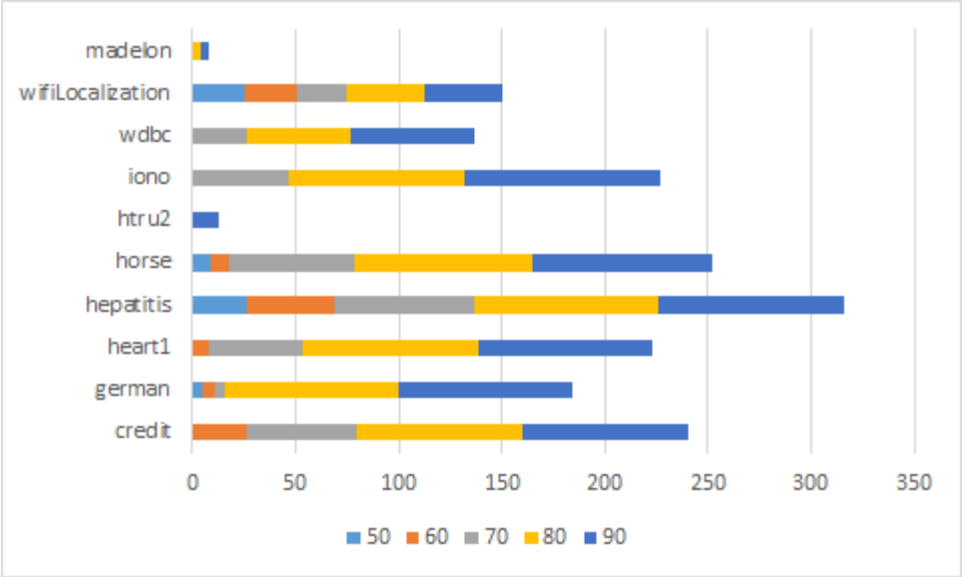

FIGURE 2. Comparison of the number of features for different threshold values

obtained in this way are presented in Tables 4.9, 4.10, 4.11 and 4.12, with the best accuracy values highlighted in bold.

According to Tables 4.9, 4.10, 4.11, and 4.12, using the reduced datasets obtained with the FS-GKM algorithm, the best accuracy was achieved in 1 out of 3 cases for the digits dataset, in 2 out of 3 cases for the wdbc dataset, and in all cases for the wine and iris datasets.

5. CONCLUSION

In this paper, a novel feature selection algorithm, FS-GKM, is introduced. The algorithm partitions the instances in the dataset into clusters based on the number of classes, using the global k-means clustering algorithm. If the class distributions within the resulting clusters exceed a predefined threshold, the feature is considered discriminative, and these features are included in the reduced feature set. Since the proposed algorithm is sensitive to the threshold, it is run with different threshold values, and the best results are retained. To illustrate the effectiveness of the method, 13 benchmark datasets were reduced using 14 different feature selection algorithms, including FS-GKM. Then, SVM, KNN, CART, and MLP classifiers were trained and tested on both the original and reduced datasets. Based on the accuracy values obtained from the training and testing results, it was observed that the datasets reduced with the FS-GKM algorithm showed better performance in approximately 77% of the cases.

Although the effectiveness of the method has been demonstrated through computational experiments, it suffers from sensitivity to the threshold value. As future work, it is planned to explore whether better results can be achieved by using different clustering methods, and to develop an approach for determining the threshold value provided as input to the method.

TABLE 4.4. Accuracy Results of Feature Selection Algorithms Using SVM

Dataset	original	FARNeMF	GBNRS	GRRS-N	GRRS-1	CFS	Fisher	ILFS	Laplacian	LASSO	Inf-FS	FS-GKM
credit	0.855 ± 0.022	0.855 ± 0.021	0.855 ± 0.021	0.855 ± 0.022	0.855 ± 0.022	0.659 ± 0.020	0.629 ± 0.022	0.855 ± 0.022	0.651 ± 0.022	0.855 ± 0.022	0.629 ± 0.022	**0.868 ± 0.037**
german	0.708 ±0.008	0.716 ± 0.017	0.713 ± 0.017	0.721 ± 0.003	0.709 ± 0.008	0.700 ± 0.006	0.727 ± 0.011	0.730 ± 0.005	0.700 ± 0.010	0.702 ± 0.009	0.727 ± 0.011	**0.750 ± 0.033**
heart1	0.742 ± 0.121	0.742 ±0.090	0.734 ± 0.116	0.792 ± 0.069	0.792 ± 0.076	0.786 ± 0.081	0.793 ± 0.078	0.745 ± 0.073	0.742 ± 0.128	0.789 ± 0.078	0.793 ± 0.078	**0.870 ± 0.025**
hepatitis	0.794 ± 0.017	0.793 ± 0.011	0.793 ± 0.011	0.793 ± 0.011	0.793 ± 0.017	0.794 ± 0.017	0.794 ± 0.017	0.793 ± 0.017	0.794 ± 0.017	0.794 ± 0.017	0.794 ± 0.017	**0.834 ± 0.094**
horse	0.809 ± 0.028	0.809 ± 0.028	0.809 ± 0.028	0.809 ± 0.028	**0.810 ± 0.028**	0.649 ± 0.028	0.622 ± 0.057	0.665 ± 0.028	0.613 ± 0.028	0.657 ± 0.053	0.622 ± 0.057	0.750 ± 0.146
htru2	0.974 ± 0.003	0.973 ± 0.003	0.974 ± 0.002	0.976 ± 0.002	0.976 ± 0.002	0.947 ± 0.002	0.975 ± 0.002	0.947 ± 0.003	0.908 ± 0.003	0.976 ± 0.002	0.975 ± 0.002	**0.982 ± 0.002**
iono	0.866 ± 0.015	0.889 ± 0.033	0.854 ± 0.020	0.889 ± 0.030	0.889 ± 0.044	0.900 ± 0.025	0.903 ± 0.047	0.858 ± 0.045	0.843 ± 0.038	0.761 ± 0.042	0.903 ± 0.047	**0.943 ± 0.046**
wdbc	0.952 ± 0.017	0.957 ± 0.020	0.963 ± 0.019	0.963 ± 0.020	0.956 ± 0.006	0.942 ± 0.015	0.926 ± 0.017	0.947 ± 0.017	0.937 ± 0.018	0.909 ± 0.017	0.926 ± 0.017	**0.974 ± 0.013**
wifiLocalization	0.972 ± 0.004	0.974 ± 0.008	0.974 ± 0.008	0.976 ± 0.010	0.976 ± 0.010	0.972 ± 0.020	0.972 ± 0.011	0.972 ± 0.004	0.976 ± 0.010	0.975 ± 0.010	0.972 ± 0.011	**0.983 ± 0.007**
madelon	0.598 ± 0.021	0.616 ± 0.022	0.620 ± 0.026	0.616 ± 0.021	0.610 ± 0.021	0.590 ± 0.026	**0.625 ± 0.015**	0.620 ± 0.010	0.612 ± 0.023	0.615 ± 0.017	0.615 ± 0.017	0.578 ± 0.026

TABLE 4.5. Accuracy Results of Feature Selection Algorithms Using Cart

Dataset	original	FARNeMF	GBNRS	GRRS-N	GRRS-1	CFS	Fisher	ILFS	Laplacian	LASSO	Inf-FS	FS-GKM
credit	0.785 ± 0.018	0.807 ± 0.018	0.816 ± 0.026	0.829 ± 0.018	0.838 ± 0.025	0.744 ± 0.041	0.613 ± 0.035	0.794 ± 0.030	0.701 ± 0.020	0.755 ± 0.022	0.607 ± 0.016	**0.860 ± 0.125**
german	0.684 ±0.043	0.689 ± 0.032	0.680 ± 0.030	0.684 ± 0.019	0.713 ± 0.024	0.697 ± 0.006	0.705 ± 0.013	0.706 ± 0.020	0.656 ± 0.023	0.710 ± 0.019	0.706 ± 0.019	**0.745 ± 0.047**
heartl	0.664 ± 0.084	0.711 ±0.079	0.731 ± 0.058	0.714 ± 0.060	0.731 ± 0.063	0.742 ± 0.080	0.667± 0.030	0.745 ± 0.073	0.671 ± 0.076	0.789 ± 0.079	0.670 ± 0.078	**0.870 ± 0.091**
hepatitis	0.720 ± 0.042	0.811 ± 0.066	0.830 ± 0.097	0.839 ± 0.028	0.832 ± 0.062	**0.845 ± 0.053**	0.768 ± 0.035	0.819 ± 0.062	0.799 ± 0.077	0.794 ± 0.053	0.768 ± 0.035	0.839 ± 0.094
horse	0.793 ± 0.027	0.784 ± 0.051	0.817 ± 0.013	0.820 ± 0.038	0.837 ± 0.044	0.698 ± 0.045	0.614 ± 0.057	0.684 ± 0.049	0.668 ± 0.065	0.636 ± 0.014	0.625 ± 0.062	**0.917 ± 0.172**
htru2	0.967 ± 0.002	0.968 ± 0.003	0.967 ± 0.003	0.967 ± 0.002	0.969 ± 0.002	0.937 ± 0.003	0.963 ± 0.002	0.957 ± 0.003	0.924 ± 0.005	0.967 ± 0.003	0.969 ± 0.002	**0.970 ± 0.002**
iono	0.886 ± 0.038	0.928 ± 0.037	0.891 ± 0.015	0.931 ± 0.030	0.929 ± 0.017	0.920 ± 0.032	0.926 ± 0.039	0.901 ± 0.038	0.897 ± 0.040	0.857 ± 0.037	0.923 ± 0.039	**0.957 ± 0.059**
wdbc	0.928 ± 0.019	0.949 ± 0.015	0.943 ± 0.009	0.953 ± 0.018	0.961 ± 0.023	0.947 ± 0.019	0.940 ± 0.015	0.928 ± 0.026	0.921 ± 0.030	0.912 ± 0.016	0.940 ± 0.010	**0.965 ± 0.027**
wifiLocalization	0.960 ± 0.024	0.963 ± 0.022	0.964 ± 0.021	0.966 ± 0.014	0.969 ± 0.011	0.957 ± 0.028	0.957 ± 0.029	0.965 ± 0.023	0.964 ± 0.015	0.967 ± 0.011	0.955 ± 0.027	**0.988 ± 0.022**
madelon	0.756 ± 0.016	0.785 ± 0.012	0.590 ± 0.026	**0.820 ± 0.034**	0.799 ± 0.023	0.744 ± 0.017	0.761 ± 0.024	0.758 ± 0.016	0.794 ± 0.015	0.752 ± 0.028	0.796 ± 0.024	0.793 ± 0.018

TABLE 4.6. Accuracy Results of Feature Selection Algorithms Using MLP

Dataset	original	FARNeMF	GBNRS	GRRS-N	GRRS-1	CFS	Fisher	ILFS	Laplacian	LASSO	Inf-FS	FS-GKM
credit	0.866 ± 0.023	**0.872 ± 0.026**	0.865 ± 0.023	0.861 ± 0.019	0.869 ± 0.025	0.750 ± 0.046	0.641 ± 0.025	0.857 ± 0.018	0.762 ± 0.029	0.865 ± 0.022	0.629 ± 0.021	0.798 ± 0.033
german	0.742 ±0.013	0.755 ± 0.021	0.748 ± 0.013	0.749 ± 0.021	0.760 ± 0.023	0.702 ± 0.016	0.752 ± 0.028	0.744 ± 0.005	0.725 ± 0.022	0.724 ± 0.014	0.750 ± 0.027	**0.765 ± 0.037**
heart1	0.756 ± 0.099	0.756 ±0.120	0.762 ± 0.101	0.792 ± 0.069	0.792 ± 0.076	0.762 ± 0.108	0.789± 0.078	0.745 ± 0.061	0.745 ± 0.068	0.789 ± 0.078	0.793 ± 0.078	**0.870 ± 0.040**
hepatitis	0.825 ± 0.048	0.839 ± 0.058	0.813 ± 0.032	0.825 ± 0.039	0.852 ± 0.028	0.833 ± 0.035	0.813 ± 0.036	0.813 ± 0.053	0.820 ± 0.062	0.825 ± 0.035	0.800 ± 0.042	**0.859 ± 0.094**
horse	0.801 ± 0.043	0.817 ± 0.026	0.820 ± 0.018	0.826 ± 0.013	**0.829 ± 0.017**	0.684 ± 0.036	0.649 ± 0.075	0.677 ± 0.045	0.641 ± 0.026	0.649 ± 0.041	0.654 ± 0.071	0.733 ± 0.113
htru2	0.979 ± 0.002	0.979 ± 0.002	0.979 ± 0.002	0.979 ± 0.001	0.980 ± 0.003	0.955 ± 0.001	0.978 ± 0.002	0.974 ± 0.002	0.928 ± 0.001	0.979 ± 0.003	0.978 ± 0.002	**0.981 ± 0.002**
iono	0.914 ± 0.028	0.917 ± 0.018	0.900 ± 0.019	0.897 ± 0.039	0.912 ± 0.029	0.918 ± 0.025	0.923 ± 0.025	0.903 ± 0.032	0.889 ± 0.034	0.872 ± 0.033	0.923 ± 0.032	**0.971 ± 0.043**
wdbc	0.972 ± 0.011	0.968 ± 0.022	0.971 ± 0.019	0.970 ± 0.021	0.972 ± 0.009	0.961 ± 0.012	0.945 ± 0.011	0.949 ± 0.007	0.958 ± 0.015	0.933 ± 0.017	0.942 ± 0.015	**0.976 ± 0.024**
wifiLocalization	0.976 ± 0.014	0.977± 0.013	0.980 ± 0.009	0.979 ± 0.003	0.980 ± 0.006	0.975 ± 0.016	0.973 ± 0.022	0.979 ± 0.010	0.979 ± 0.004	0.976 ± 0.014	0.975 ± 0.024	**0.981 ± 0.006**
madelon	0.559 ± 0.030	0.715 ± 0.012	0.630 ± 0.026	**0.769 ± 0.015**	0.737 ± 0.016	0.571 ± 0.029	0.715 ± 0.023	0.658 ± 0.012	0.757 ± 0.012	0.595 ± 0.020	0.755 ± 0.012	0.543 ± 0.015

TABLE 4.7. Accuracy Results of Feature Selection Algorithms Using KNN

Dataset	original	FARNeMF	GBNRS	GRRS-N	GRRS-1	CFS	Fisher	ILFS	Laplacian	LASSO	Inf-FS	FS-GKM
credit	0.871 ± 0.021	0.863 ± 0.022	0.856 ± 0.032	0.876 ± 0.031	0.860 ± 0.028	0.723 ± 0.015	0.634 ± 0.021	0.839 ± 0.020	0.737 ± 0.020	0.855 ± 0.032	0.634 ± 0.021	**0.898 ± 0.036**
german	0.712 ±0.014	0.741 ± 0.029	0.727 ± 0.016	0.734 ± 0.018	0.739 ± 0.029	0.657 ± 0.017	0.738 ± 0.049	0.732 ± 0.020	0.680 ± 0.020	0.686 ± 0.019	0.738 ± 0.049	**0.780 ± 0.037**
heart1	0.760 ± 0.092	0.758 ±0.097	0.762 ± 0.096	0.775 ± 0.062	0.782 ± 0.076	0.775 ± 0.080	0.738± 0.074	0.734 ± 0.092	0.748 ± 0.092	0.649 ± 0.097	0.738 ± 0.074	**0.852 ± 0.074**
hepatitis	0.799 ± 0.080	0.848 ± 0.037	0.857 ± 0.052	0.851 ± 0.041	0.858 ± 0.028	0.851 ± 0.017	0.781 ± 0.043	0.839 ± 0.048	0.831 ± 0.062	0.805 ± 0.080	0.781 ± 0.043	**0.859 ± 0.094**
horse	0.777 ± 0.056	0.801 ± 0.032	0.845 ± 0.034	**0.861 ± 0.019**	**0.861 ± 0.019**	0.657 ± 0.050	0.640 ± 0.047	0.649 ± 0.062	0.638 ± 0.048	0.640 ± 0.061	0.640 ± 0.047	0.850 ± 0.182
htru2	0.977 ± 0.002	0.977 ± 0.002	0.9977 ± 0.002	0.978 ± 0.001	0.979 ± 0.002	0.950 ± 0.001	0.977 ± 0.002	0.972 ± 0.001	0.929 ± 0.001	0.978 ± 0.001	0.977 ± 0.002	**0.986 ± 0.002**
iono	0.848 ± 0.045	0.911 ± 0.039	0.889 ± 0.057	0.928 ± 0.030	0.926 ± 0.028	0.903 ± 0.033	0.914 ± 0.043	0.903 ± 0.032	0.857 ± 0.050	0.871 ± 0.050	0.914 ± 0.043	**0.971 ± 0.054**
wdbc	0.966 ± 0.015	0.973 ± 0.020	0.966 ± 0.024	0.966 ± 0.022	0.970 ± 0.007	0.959 ± 0.011	0.950 ± 0.015	0.954 ± 0.013	0.956 ± 0.020	0.917 ± 0.017	0.950 ± 0.020	**0.979 ± 0.028**
wifiLocalization	0.975 ± 0.005	0.975± 0.003	0.976 ± 0.006	0.977 ± 0.006	0.977 ± 0.004	0.975 ± 0.030	0.975 ± 0.058	0.975 ± 0.027	0.977 ± 0.014	0.976 ± 0.027	0.975 ± 0.058	**0.993 ± 0.010**
madelon	0.574 ± 0.025	0.750 ± 0.018	0.579 ± 0.027	**0.900 ± 0.004**	0.891 ± 0.013	0.569 ± 0.035	0.737 ± 0.010	0.637 ± 0.022	0.840 ± 0.015	0.571 ± 0.032	0.824 ± 0.015	0.780 ± 0.033

TABLE 4.8. Comparison of the number of features

Datasets	The number of features of original dataset	The number of features of reduced dataset
iris	4	2
wine	19	7
wdbc	30	23
digits	64	30

TABLE 4.9. Accuracy results for the dataset digits

	Feature Selection Methods			
Classifiers	ANOVA	RFE	PCA	FS-GKM
SVM	**0.97**	0.96	**0.97**	0.84
MLP	0.97	**0.98**	**0.98**	**0.98**
Cart	0.84	**0.87**	0.84	0.76

TABLE 4.10. Accuracy results for the dataset wine

	Feature Selection Methods			
Classifiers	ANOVA	RFE	PCA	FS-GKM
SVM	**1**	**1**	0.96	**1**
MLP	0.13	0.70	0.89	**0.93**
Cart	0.94	0.91	0.91	**0.96**

TABLE 4.11. Accuracy results for the dataset iris

	Feature Selection Methods			
Classifiers	ANOVA	RFE	PCA	FS-GKM
SVM	**1**	0.98	**1**	**1**
MLP	**1**	**1**	**1**	**1**
Cart	**1**	0.98	**1**	**1**

TABLE 4.12. Accuracy results for the dataset wdbc

	Feature Selection Methods			
Classifiers	ANOVA	RFE	PCA	FS-GKM
SVM	**0.98**	0.97	0.96	**0.98**
MLP	0.95	**0.97**	0.96	0.95
Cart	**0.94**	0.93	0.93	**0.94**

Acknowledgments. The authors would like to thank the referee for some useful comments and their helpful suggestions that have improved the quality of this paper.

Conflict of interest. The authors declare no potential conflict of interests.

References

[1] Abdi, H., & Williams, L. J. (2010). Principal component analysis. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4), 433–459.

[2] Breiman, L., Friedman, J., Olshen, R. A., & Stone, C. J. (1986). *Classification and Regression Trees*. Wadsworth & Brooks/Cole.

[3] Chicco, D., & Jurman, G. (2020). The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation. *BMC Genomics*, 21(6).

[4] Cortes, C., & Vapnik, V. (1995). Support vector networks. *Machine Learning*, 20(3), 273–297.

[5] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21–27.

[6] Dua, D., & Graff, C. (2019). UCI Machine Learning Repository. https://archive.ics.uci.edu/ml/index.php

[7] Fang, Y., Yao, Y., Lin, X., Wang, J., & Zhai, H. (2024). A feature selection based on genetic algorithm for intrusion detection of industrial control systems. *Computers & Security*, 139, 103675.

[8] Gu, Q., Li, Z., & Han, J. (2012). Generalized fisher score for feature selection. *arXiv preprint*, arXiv:1202.372.

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

[10] Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46, 389–422.

[11] He, X., Cai, D., & Niyogi, P. (2005). Laplacian score for feature selection. In *Advances in Neural Information Processing Systems* (Vol. 18).

[12] Hu, Q., Zhao, H., & Yu, D. (2008). Efficient symbolic and numerical attribute reduction with neighborhood rough sets. *Pattern Recognition and Artificial Intelligence*, 21(6), 732–738.

[13] Jolliffe, I. T. (2002). *Principal Component Analysis*. Springer.

[14] Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence*, 97, 273–324.

[15] Kira, K., & Rendell, L. A. (1992). The feature selection problem: Traditional methods and a new algorithm. In *Proceedings of the Tenth National Conference on Artificial Intelligence* (pp.129–134).

[16] Kuzudisli, C., Bakir-Gungor, B., Bulut, N., Qaqish, B., & Yousef, M. (2023). Review of feature selection approaches based on grouping of features. *PeerJ*, 11, e15666.

[17] Liu, H., & Yu, L. (2005). Toward integrating feature selection algorithms for classification and clustering. *IEEE Transactions on Knowledge and Data Engineering*, 17(4), 491–502.

[18] Liu, H., & Motoda, H. (2012). *Feature Selection for Knowledge Discovery and Data Mining*. Springer.

[19] MacQueen, J. (1967). Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability* (pp. 281–297).

[20] Meinshausen, N., & Bühlmann, P. (2010). Stability selection. *Journal of the Royal Statistical Society: Series B*, 72(4), 417–473.

[21] Roffo, G., Melzi, S., Castellani, U., & Vinciarelli, A. (2017). Infinite latent feature selection: A probabilistic latent graph-based ranking approach. In *Proceedings of the IEEE International Conference on Computer Vision* (pp.1398–1406).

[22] Roffo, G., Melzi, S., Castellani, U., Vinciarelli, A., & Cristani, M. (2020). Infinite feature selection: A graph-based feature filtering approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(12), 4396–4410.

[23] Rumelhart, D. E., Hinton, G. E., & Williams, R. J. (1986). Learning representations by back-propagating errors. *Nature*, 323(6088), 533–536.

[24] Saeys, Y., Inza, I., & Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *Bioinformatics*, 23(19), 2507–2517.

[25] Shayestegan, M., Kohout, J., Verešpejová, L., Chovanec, M., & Mareš, J. (2024). Comparison of Feature Selection and Supervised Methods for Classifying Gait Disorders. *IEEE Access*, 12, 17876–17894.

[26] Sridharan, K., & Sivakumar, P. (2018). A systematic review on techniques of feature selection and classification for text mining. *International Journal of Business Information Systems*, 28(4), 504-518.

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

[28] Turan, D. S., & Ordin, B. (2025). The incremental SMOTE: A new approach based on the incremental k-means algorithm for solving imbalanced data set problem. *Information Sciences*, 711, 122103.

[29] Van de Geer, S. A. (2008). High-dimensional generalized linear models and the lasso.

[30] Wang, L., Wang, Y., & Chang, Q. (2016). Feature selection methods for big data bioinformatics: A survey from the search perspective. *Methods*, 111, 21-31.

[31] Xia, S., Zhang, H., Li, W., Wang, G., Giem, E., & Chen, Z. (2022). GBNRS: A novel rough set algorithm for fast adaptive attribute reduction in classification. *IEEE Transactions on Knowledge and Data Engineering*, 34(3), 1231–1242.

[32] Xia, S., Wu, S., Chen, X., Wang, G., Gao, X., Zhang, Q., Giem, E., & Chen, Z. (2022). GRRS: Accurate and efficient neighborhood rough set for feature selection. *IEEE Transactions on Knowledge and Data Engineering*, 35(9), 9281–9294.

(D. S. Turan) Ege University, Faculty of Science, Department of Matehmatics, Bornova, 35040, Izmir, Türkiye