big data and cognitive computing

Article

Supervised Density-Based Metric Learning Based on Bhattacharya Distance for Imbalanced Data Classification Problems

Atena Jalali Mojahed [1], Mohammad Hossein Moattar [2,*] and Hamidreza Ghaffari [1]

[1] Department of Computer Engineering, Ferdows Branch, Islamic Azad University, Ferdows, Iran; ajalalia@gmail.com (A.J.M.); hamidghaffary53@yahoo.com (H.G.)

[2] Department of Computer Engineering, Mashhad Branch, Islamic Azad University, Mashhad, Iran

* Correspondence: moattar@mshdiau.ac.ir

Abstract: Learning distance metrics and distinguishing between samples from different classes are among the most important topics in machine learning. This article proposes a new distance metric learning approach tailored for highly imbalanced datasets. Imbalanced datasets suffer from a lack of data in the minority class, and the differences in class density strongly affect the efficiency of the classification algorithms. Therefore, the density of the classes is considered the main basis of learning the new distance metric. It is possible that the data of one class are composed of several densities, that is, the class is a combination of several normal distributions with different means and variances. In this paper, considering that classes may be multimodal, the distribution of each class is assumed in the form of a mixture of multivariate Gaussian densities. A density-based clustering algorithm is used for determining the number of components followed by the estimation of the parameters of the Gaussian components using maximum a posteriori density estimation. Then, the Bhattacharya distance between the Gaussian mixtures of the classes is maximized using an iterative scheme. To reach a large between-class margin, the distance between the external components is increased while decreasing the distance between the internal components. The proposed method is evaluated on 15 imbalanced datasets using the k-nearest neighbor (KNN) classifier. The results of the experiments show that using the proposed method significantly improves the efficiency of the classifier in imbalance classification problems. Also, when the imbalance ratio is very high and it is not possible to correctly identify minority class samples, the proposed method still provides acceptable performance.

Keywords: imbalanced data classification; distance metric learning; Bhattacharya divergence; class density estimation

check for updates

Citation: Jalali Mojahed, A.; Moattar, M.H.; Ghaffari, H. Supervised Density-Based Metric Learning Based on Bhattacharya Distance for Imbalanced Data Classification Problems. *Big Data Cogn. Comput.* **2024**, *8*, 109. https://doi.org/10.3390/bdcc8090109

Academic Editor: Fabrizio Marozzo

Received: 23 July 2024
Revised: 21 August 2024
Accepted: 3 September 2024
Published: 4 September 2024

1. Introduction

Machine learning (ML) allows computers to learn without being explicitly programmed. Many machine learning and pattern recognition methods require calculating the distance between data points, often utilizing the Euclidean distance metric [1,2], as developing a new and suitable distance metric for the data is challenging. Currently, distance metric learning (DML) is a significant aspect of machine learning, where a machine learns a novel distance metric according to the input patterns' characteristics. This new metric can enhance the effectiveness of classification algorithms that rely on the distance metric.

Supervised learning is a category of learning algorithms. In this type of learning, the algorithm has access to labeled data. The goal of supervised distance metric learning is to train a new distance metric that brings the data points with the same label closer together and separates the points with different labels. In the learning process, according to Figure 1, the most appropriate distance matrix is extracted for each dataset using the distance metric learning algorithm. Then, the data are mapped to a new space based on the new distance metrics, and after that, various classification algorithms can be applied to the mapped data.

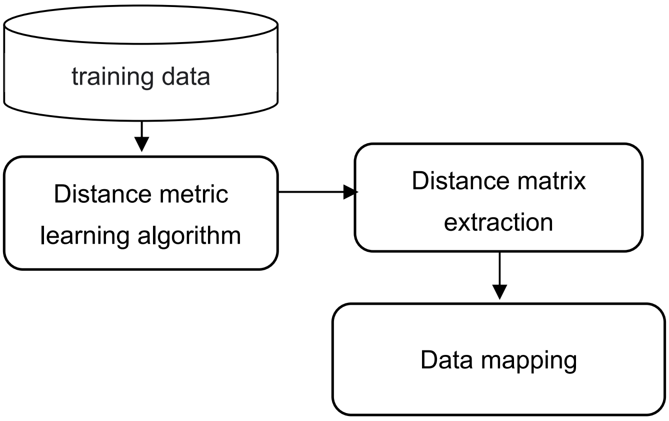

training data
Distance metric learning algorithm
Distance matrix extraction
Data mapping

Figure 1. The general process of supervised distance metric learning.

One of the important issues in the field of data mining and machine learning is the problem of classifying imbalanced datasets. An imbalanced distribution of classes in datasets occurs when the number of observations in the minority class (positive class) is much less than in the majority class (negative class). The existence of rare or very expensive samples in the real world creates imbalanced datasets [3,4]. On the other hand, traditional classification algorithms such as the k-nearest neighbor (KNN) algorithm and support vector machine (SVM) do not work well on imbalanced datasets because they do not consider the quality of the data space and the imbalance ratio. Therefore, when classifying imbalanced datasets, these algorithms tend towards the majority class and consider the minority class samples as noise or outliers. Consequently, the probability of misclassifying the minority class compared to the majority class increases, and the accuracy of their performance on these samples is very low. Additionally, when the imbalance ratio is very high, it becomes challenging to identify the minority class [5,6].

The challenge of classifying imbalanced datasets is usually encountered in scenarios where anomaly detection is crucial, such as medical diagnosis (diagnosing rare diseases) [7], fraud detection in the banking system [8], prediction of natural disasters like earthquakes, face recognition, text classification [9], error detection [10], and anomaly detection [11]. Consequently, research in the field of imbalanced dataset classification has gained significant attention in recent years [12]. Given that in real-world applications, the primary objective is often to identify rare cases, it is imperative to develop a model capable of accurately classifying the minority class.

In this article, a new distance metric is proposed for imbalanced data classification. The difference in class density strongly affects the efficiency of classification algorithms. Therefore, the local density of classes is the primary basis for learning the new approach. In the proposed method, the density between classes is the main criterion for learning the distance metric. Imbalanced datasets suffer from a lack of data in the minority class, and the disparity between the density of the minority class and the majority class significantly impacts the efficiency of the k-nearest neighbor classification algorithm.

On the other hand, it is possible that the data of one class are composed of several densities. That is, the data are a combination of several normal distributions with different parameters, each having distinct means and variances. In the proposed method, for identifying the number of normal distributions, the distribution of classes is assumed to be a mixture of Gaussians. The number of components is determined using the DBSCAN density-based clustering algorithm. To accurately identify the Gaussian components of the Gaussian mixture model (GMM) probabilistic models, the parameters are estimated and updated separately using the maximum a posteriori (MAP) estimator. Subsequently, the distance between the densities of Gaussian components is maximized using the Bhattacharya distance and an iterative optimization algorithm to achieve a large between-class margin.

The rest of this article is organized as follows: In Section 2, works related to the topic of distance metric learning are reviewed. In Section 3, the proposed model is presented, which is based on training the appropriate distance metric for the imbalanced datasets to enhance the efficiency of classification algorithms. Section 4 reports the results of experiments on several imbalanced datasets. The conclusion and future works are outlined in Section 5.

2. Related Works

2.1. Background and Definitions

Distance metric learning emerged in 2003 in Mr. Xing's article with the introduction of Mahalanobis distance [13]. Unlike Euclidean distance, Mahalanobis distance also considers the correlation between features and is represented by Equation (1).

$$d_M(x,x') = \sqrt{(x-x')^T\ M(x-x')} \tag{1}$$

where x and x' are two random vectors from the same distribution with covariance matrix M. Matrix M is a symmetric positive semi-definite (PSD) matrix with d × d dimensions [14]. If M is an identity matrix, Equation (1) will represent the Euclidean distance. Since M is PSD, d_M has the following properties.

$$d_M(x,x') \geq 0$$
$$d_M(x,x) = 0$$
$$d_M(x,x') = d_M(x',x)$$
$$d_M(x,x'') \leq d_M(x,x') + d_M(x',x'')$$

Since matrix M can be written as $M = L^TL$, the linear transformation of the data is performed with the transfer matrix L in the Mahalanobis distance.

$$d_M(x,x') = \sqrt{(x-x')^T L^T L(x-x')}$$
$$d_M(x,x') = \sqrt{(Lx-xL')^T(Lx-Lx')}$$

In general, the research conducted in the field of distance metric learning can be divided into the following categories [15].

a. Pairwise cost-based approaches

Weinberger et al. [16,17] introduced the large margin nearest neighbor (LMNN) algorithm, a supervised distance metric designed to improve the k-nearest neighbor classifier. LMNN works by reducing the distance between k-nearest neighbors that share the same label (target neighbors), thus increasing their separation from samples of different classes and creating a larger margin. However, a notable limitation of this algorithm is its focus on optimizing intra-class distances, meaning the initially designated target neighbors do not change during training. As a result, the performance of LMNN is highly dependent on the initial selection of these target neighbors.

Zadeh et al. [18] presented the geometric mean metric learning (GMML) algorithm to improve the accuracy of k-nearest neighbor classification. In their approach, samples are divided into two sets: S for similar point pairs and D for dissimilar point pairs. The algorithm computes a similarity matrix for S and a dissimilarity matrix for D. It then finds the shortest line connecting the inverses of the second-moment matrix of similar points to the second-moment matrix of dissimilar points, with the midpoint of this line defined as the geometric mean. This method aims to achieve a global minimum in a strongly convex objective function, focusing on minimizing the total distance among all similar points.

Ying et al. [19] developed the distance metric learning algorithm with eigenvalue optimization (DML_eig) to improve the accuracy of k-nearest neighbor classification. Their method involves dividing samples into two sets: S, which contains pairs of similar points, and D, which includes pairs of dissimilar points. They refined the objective function from Xing et al. that seeks to maximize the distances between dissimilar pairs while keeping the sum of squared distances between similar pairs within an upper limit. The DML_eig algorithm optimizes this process by identifying the largest eigenvector in each iteration, which helps maximize the minimum squared distances between dissimilar pairs while adhering to the constraints for similar pairs. However, this approach requires eigen decomposition of a matrix at each iteration, resulting in significant time consumption during convergence.

Nguyen et al. [20] introduced distance metric learning through maximization of the Jeffrey divergence (DMLMJ) to enhance the accuracy of k-nearest neighbor classification. In their approach, they divided samples into two groups: S, comprising k similar neighbors, and D, consisting of k dissimilar neighbors. They calculated the covariance matrices for both sets and derived their eigenvectors and eigenvalues. The objective was to learn a linear transformation that maximizes the Jeffrey divergence between the two multivariate Gaussian distributions.

b. Probabilistic approaches

In this group of methods, researchers aim to maximize the probability that a point is similar to its neighbors while minimizing classification error based on k-nearest neighbors in the image space. A significant disadvantage of these methods is that their equations are non-convex, leading to convergence at a local maximum and increased computational complexity. Davis et al. [21], in a method known as information theoretic metric learning (ITML), categorized all samples into two sets: classmates and non-classmates. They sought to train and optimize the Mahalanobis distance by minimizing the log-determinant divergence. This method employs concepts from Bergman's information theory and optimization.

c. Boost-Like methods

Boost-like methods try to train a new metric in each step with a linear combination of the sub-metrics of the previous steps, which have weak constraints in their formula. In other words, by adding weaker learners, strong learners are produced. Chang et al. [22] introduced a boosting algorithm for supervised learning of the Mahalanobis distance metric called BoostMDM. In this method, a cost function is defined, which is repeatedly reduced in each step. In each iteration, the metric matrix is combined with what was learned in the previous step. In this method, the entire sample space is used for training.

d. Hybrid approaches

This category explores the integration of distance metric learning with other models, such as online learning, to enhance system efficiency. These methods are adaptable to large datasets and can continuously update metrics with current data.

Zhong et al. [23] introduced the scalable large margin online metric learning (SLMOML) method, which employs log-determinant divergence to maintain proximity between two trained Mahalanobis distance metrics. Using the Hinge loss function, SLMOML creates a large margin between different samples and connects passive learning with Bergman imaging, achieving global convergence. The initial Mahalanobis matrix remains positive semi-definite (PSD) throughout the process.

Liu et al. [24] proposed LM-KNN, addressing label prediction in multi-label problems where each example can have multiple related tags, such as in document classification. To reduce the cost and increase the speed of tag prediction for unseen samples, they utilized a distance metric to identify relationships between labels, enabling the separation of distinct tags. Rather than relying on costly optimization techniques, LM-KNN employs a KNN solution to predict labels in the transformed space.

e. Deep metric learning approaches

In this category, distance metric learning is combined with deep learning to increase the efficiency of the system for managing non-linear and massive data [25–27]. In this type of learning, instead of Mahalanobis distance, neural networks are used to create a new feature space with high discrimination power. Cao et al. [28] used a deep neural network to reduce the intra-class distance, increase the extra-class distance, and improve the performance of classification methods such as KNN.

2.2. Learning Distance Metric in Imbalanced Applications

Wang et al. [29] introduced the iterative metric learning (IML) method to address imbalanced data in classification, consisting of three steps. First, the LMNN algorithm iteratively transforms training samples to better align with the test data space. Second, the distance between test samples and training samples is calculated to select the closest training samples, thereby reducing the training sample size. Finally, training samples from each iteration are compared to those from the previous iteration to retain the most similar ones, creating a more stable neighborhood space for test samples.

Feng et al. [30] proposed the distance metric by balanced KL-divergence (DMBK), which develops a new distance metric suitable for imbalanced datasets. They assumed a Gaussian density with uniform covariance across classes and used Kullback–Leibler divergence to measure the distance between class distributions. To effectively handle samples from both minority and majority classes, they utilized the logarithm of the geometric mean of the normalized Kullback–Leibler deviation, solving the optimization problem using ascending gradients and iterative operations.

Gautheron et al. [31] introduced imbalanced metric learning (IML), which creates a distance metric to improve performance on imbalanced datasets. Unlike the traditional Mahalanobis formula, which minimizes loss across all pairs without considering labels, IML employs two loss functions: one for same-label pairs, aiming to reduce their distance below one, and another for different-label pairs to increase their distance beyond one plus a margin.

Yan et al. [32] presented the deep metric framework with border-line-margin loss (DMFBML), combining a new distance metric with a neural network to minimize intra-class distance while maximizing extra-class distance in overlapping class regions, thereby improving classification accuracy for minority class samples in imbalanced datasets. Lastly, the authors of [33] proposed a method to extract desirable features and reduce undesirable ones in imbalanced datasets, which could enhance distance metric learning and facilitate the transfer of data to a new feature space with higher discrimination power. This paper proposes a method that simultaneously selects and extracts features through a cost-sensitive optimization problem. The feature extraction phase focuses on reducing error and maintaining geometric relationships between data using a manifold learning optimization problem. In the feature selection phase, a cost-sensitive optimization problem is used to minimize the upper limit of the generalization error. The combined optimization problem is solved by adding a cost-sensitive term to balance the classes without manipulating the data.

The authors of [34] introduced DFSVM, a novel method for imbalanced classification that combines deep learning with fuzzy support vector machines. DFSVM begins by utilizing a deep neural network to generate an embedding representation of the data, trained with triplet loss to strengthen the similarities within classes while maximizing differences between classes. To address the challenges posed by imbalanced data distribution, an oversampling technique is applied in the embedding space, focusing on feature and center distances to create diverse new samples and mitigate overfitting. Finally, a fuzzy support vector machine (FSVM) with cost-sensitive learning serves as the classifier, assigning higher misclassification costs to minority class samples to enhance overall classification performance.

3. The Proposed Method

Since most distance metric learning algorithms are designed for balanced datasets, it is essential to introduce additional algorithms that cater to imbalanced datasets. In this article, we present a novel distance metric called DMLdbIm, specifically developed for imbalanced datasets. The aim of DMLdbIm is to establish an appropriate data space that enhances the efficiency of distance-based classification algorithms.

In imbalanced datasets, the disparity between the densities of the minority and majority classes is extremely pronounced. Consequently, during classification, the minority class may be overlooked, leading to a significant reduction in the classifier's effectiveness for that class. The proposed algorithm emphasizes the density of the minority class, employing a new distance metric designed to bring samples of the minority class closer together while maximizing their distance from samples of the majority class in the data space. This approach aims to enhance the classification algorithm's performance on the minority class. The overall structure of the proposed DMLdbIm method is illustrated in Figure 2. The subsequent sections will discuss the various phases of the proposed model.

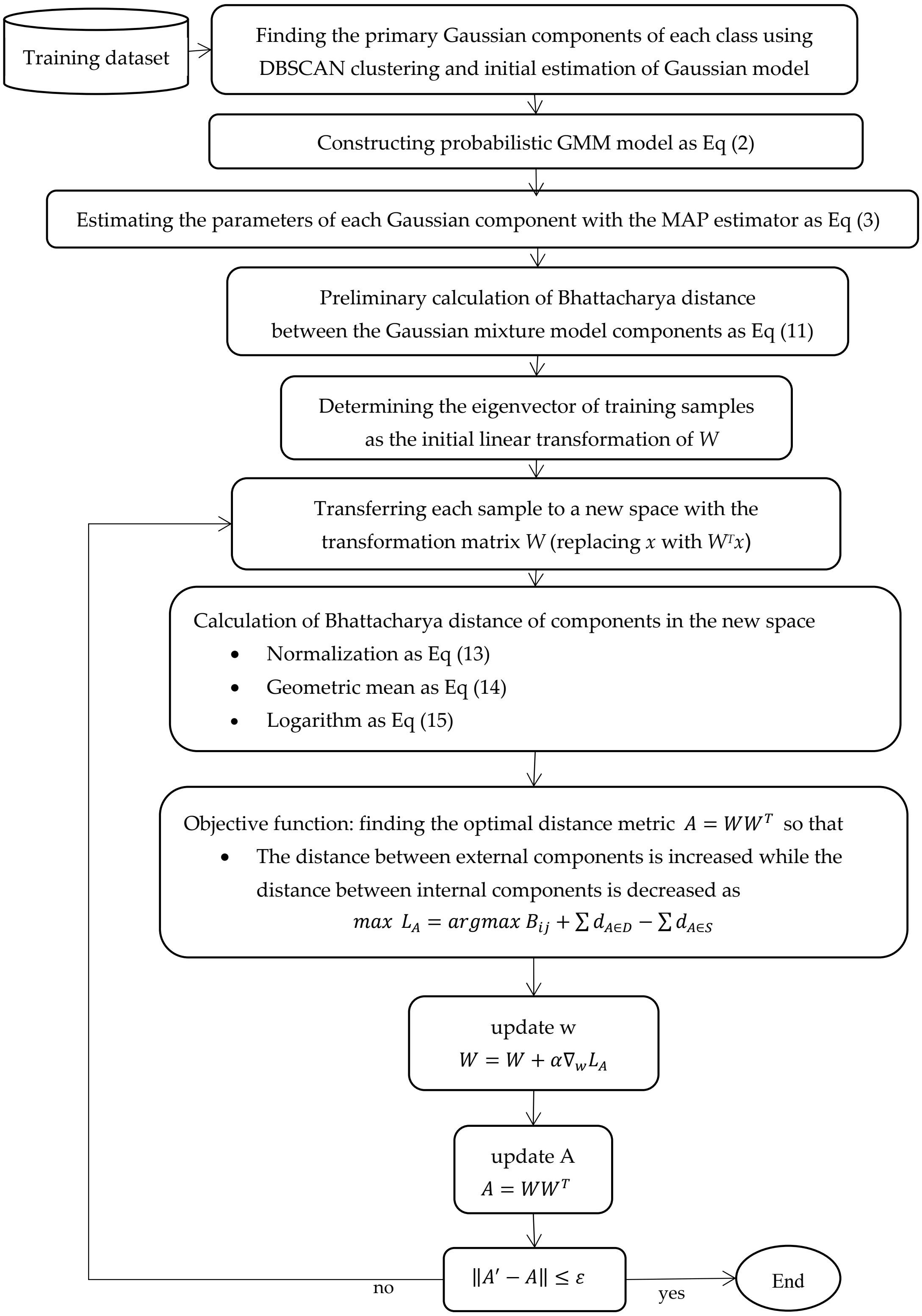

Training dataset
Finding the primary Gaussian components of each class using DBSCAN clustering and initial estimation of Gaussian model
Constructing probabilistic GMM model as Eq (2)
Estimating the parameters of each Gaussian component with the MAP estimator as Eq (3)
Preliminary calculation of Bhattacharya distance between the Gaussian mixture model components as Eq (11)
Determining the eigenvector of training samples as the initial linear transformation of W
Transferring each sample to a new space with the transformation matrix W (replacing x with $W^T x$)
Calculation of Bhattacharya distance of components in the new space
Normalization as Eq (13)
Geometric mean as Eq (14)
Logarithm as Eq (15)
Objective function: finding the optimal distance metric $A = WW^T$ so that
The distance between external components is increased while the distance between internal components is decreased as
$max\ L_A = argmax\ B_{ij} + \sum d_{A \in D} - \sum d_{A \in S}$
update w
$W = W + \alpha \nabla_w L_A$
update A
$A = WW^T$
$\|A' - A\| \leq \varepsilon$
no
yes
End

Figure 2. The general structure of the proposed DMLdbIm model.

3.1. The Proposed Model Construction Method

The proposed approach includes 5 phases.

3.1.1. The First Phase: Estimating the Density of the Classes

In the proposed method, a new distance metric is developed based on the density differences between classes. This approach is particularly relevant for imbalanced datasets, where the disparity in density between the minority and majority classes is often significant. Additionally, it is possible for the data of a single class to consist of multiple populations; that is, the data may be represented as a combination of several normal distributions, each with distinct parameters, including different means and variances. The proposed method aims to identify the number of normal distributions present in the data. For example, in a medical dataset, the number of patients with a rare disease may be considerably lower than that of those who are unaffected. By employing the new distance metric, we can analyze the density differences between affected and unaffected patients. If the affected class comprises multiple subgroups with varying characteristics, we can model this combination using multivariate Gaussian models. The distribution of different classes is represented as a mixture of Gaussian distributions with varying covariances. Consequently, if the sample distribution of each class adheres to several Gaussian models, the relevant parameters for each Gaussian can be estimated. If $X = \{x_1,\ldots,x_n\}$ denotes the set of training data, then the multivariate Gaussian density function can be expressed as shown in Equation (2).

$$g\left(X;\mu,\sum\right) = \frac{1}{2\pi^{\frac{D}{2}}\left|\sum\right|^{\frac{1}{2}}}\exp\left(-\frac{1}{2}(X-\mu)^T\sum\nolimits^{-1}(X-\mu)\right) \tag{2}$$

$$\mu = \frac{1}{n}\sum\nolimits_{i=1}^{n} x_i$$

$$\sum = \frac{1}{n}\sum\nolimits_{i=1}^{n} (x_i-\mu)(x_i-\mu)^T$$

In Equation (2), D is the dimensionality (the number of features in the dataset), π is the weight (percentage of the data in one component), μ is the mean vector, and $\sum$ is the covariance matrix. The covariance matrix should be symmetric and positive semi-definite (PSD). This phase includes the following steps:

a. Estimating the initial values of GMM parameters

In the proposed method, the first step is to determine whether the distribution of samples from each class conforms to several Gaussian models, followed by making initial estimates of the weights, means, and covariance parameters. To achieve this, a common clustering method is employed. Given that the focus of the proposed method is on the density of imbalanced datasets, the DBSCAN algorithm is selected. This method clusters data based on the density of data points, enabling it to identify clusters with complex shapes and recognize data points that do not belong to any cluster (outliers). However, when the data are intertwined and lack clear boundaries, DBSCAN may struggle to detect clusters accurately. In such cases, GMMs, which are probabilistic models, can be utilized to identify these types of clusters. In this step, the parameters, including weights, means, and initial covariances of the components, are calculated for each cluster identified by DBSCAN.

b. Maximum a posteriori (MAP) parameter re-estimation

In many applications, the maximum likelihood estimator is employed to estimate the covariance of class distributions. However, this method performs poorly with imbalanced datasets, where the number of samples in the minority class is significantly small. As a result, the estimate of the covariance matrix is typically elliptical and deviates from the true covariance matrix, which substantially diminishes classification accuracy. In the referenced article [26], a solution to this challenge is proposed by jointly estimating the covariance matrix for all classes, under the assumption that all classes share the same covariance. However, in real-world applications, the variances of class distributions are often unequal.

Therefore, this research addresses the challenge using the maximum a posteriori (MAP) estimator, implemented in the following two steps.

First step: The probability of the training vectors to be in the components of the initial mixture is computed by the posterior probability in Equation (3).

$$\Pr\left(i\middle|X_t,\theta_{prior}\right) = \frac{w_i g(X_t|\mu_i,\sum_i)}{\sum_{k=1}^{K} w_k g(X_t|\mu_k,\sum_k)} \tag{3}$$

In the expression $\Pr\left(i\middle|X_t,\theta_{prior}\right)$, the probability that a data point X_t originates from the ith Gaussian component is denoted, and θ_{prior} represents initial parameters. MAP involves using prior knowledge or "prior beliefs" about the model parameters. This approach allows leveraging historical information to improve parameter estimation. Using prior information can help prevent overfitting. A GMM without prior evidence may become overly sensitive to training data. $g(X_t|\mu_i,\sum_i)$ is the Gaussian density function that is expressed in Equation (2). A Gaussian mixture model is a weighted sum of K Gaussian densities as given by the equation, $\sum_{k=1}^{K} w_k g(X_t|\mu_k,\sum_k)$, where K is the number of clusters, X_t is a D-dimensional continuous-valued data vector, and w_k, k = 1, . . ., K, are the mixture weights. Utilizing MAP allows for more effective adjustments of cluster weights based on prior evidence, aiding in better identification and separation of clusters.

Then, the weight, mean, and variance parameters are calculated using Equations (4)–(6).

$$weight: \quad w_i = \sum_{t=1}^{T} \Pr\left(i\middle|X_t,\theta_{prior}\right) \tag{4}$$

$$mean: \quad E_i(X) = \frac{1}{n_i}\sum_{t=1}^{T} \Pr\left(i\middle|X_t,\theta_{prior}\right) X_t \tag{5}$$

$$variance: \quad E_i(X^2) = \frac{1}{n_i}\sum_{t=1}^{T} \Pr\left(i\middle|X_t,\theta_{prior}\right) X_t^2 \tag{6}$$

Second step: The previous estimates must be formulated as Equations (7)–(9) to ensure compatibility with the ith component.

$$\hat{w}_i = [\frac{\alpha_i^w n_i}{T} + (1-\alpha_i^w) w_i]\gamma \tag{7}$$

$$\hat{\mu}_i = \alpha_i^m E_i(X) + (1-\alpha_i^m)\mu_i \tag{8}$$

$$\hat{\delta}_i^2 = \alpha_i^v E_i\left(X^2\right) + (1-\alpha_i^v)\left(\alpha_i^2 - \mu_i^2\right) - \hat{\mu}_i^2 \tag{9}$$

Each Gaussian component is composed of the following parameters: a mean μ, which defines its center; covariance Σ, which defines its width; and mixing probability π, which defines how big or small the Gaussian function will be. If we have a dataset composed of N = 1000 three-dimensional points (D = 3), then x will be a 1000 × 3 matrix. μ will be a 1 × 3 vector, and Σ will be a 3 × 3 matrix. Adjustment coefficients $\left\{\alpha_i^w, \alpha_i^m, \alpha_i^v\right\}$ balance the new and old estimates. The γ normalization factor is applied to the weight of all of the mixtures so that the sum of the weights becomes 1. For each mixture and each parameter, the coefficients of adaptation (i.e., $\alpha_i^\rho\big)$ are defined in the form of Equation (10). According to its value, the effect of new data in estimating parameters is changed.

$$\alpha_i^\rho = \frac{n_i}{n_i + r^\rho} \tag{10}$$

The related factor r^ρ controls the number of new samples that must be observed in each mixture component before the new parameters replace the old ones. MAP can effectively estimate the means and covariances of the Gaussian distributions, especially when dealing with limited or noisy data.

3.1.2. The Second Phase: Calculating the Distance between the Gaussian Components Using the Bhattacharya Distance

The Bhattacharya distance [30] measures the symmetric difference between two probability distributions. For imbalanced data, the Bhattacharyya distance is a better choice than other divergences because it is symmetrical and its calculation method is less sensitive to the differences in the number of samples in each class and focuses more on the degree of overlap between the distributions. For $P_i = (X; \mu_i, \sum_i)$ and $P_j = \left(X; \mu_j, \sum_j\right)$ as two Gaussian distributions, the Bhattacharya distance between the two distributions is in the form of Equation (11).

$$B_{ij} = \frac{1}{8}(\mu_i - \mu_j)^T \sum^{-1}(\mu_i - \mu_j) + \frac{1}{2}\ln\left(\frac{\det \sum}{\sqrt{\det \sum_i \ \det \sum_j}}\right) \tag{11}$$

In Equation (11), the variance is equal to Equation (12):

$$\sum = \frac{\sum_i + \sum_j}{2} \tag{12}$$

3.1.3. The Third Phase: The Process of Learning the Proposed Distance Metric

To train the new distance metric (matrix A), one needs to calculate and update the transfer matrix (linear transformation), such as W, so that $A = WW^T$. Therefore, W is initialized with the eigenvector of all training samples, then each sample x is replaced with $W^T x$, and the Bhattacharya distance between the two distributions in the new space is calculated. The Bhattacharya distance is normalized between the Gaussian components according to Equation (13) to describe the difference between the components of different classes.

$$E^A_{i_k j_l} = \frac{q_{i_k} q_{j_l} B_A\left(p_{i_k} || p_{j_l}\right)}{\sum_{1 \le m \le c_i} \sum_{1 \le n \le c_j} q_m q_n B_A(p_m || p_n)} \tag{13}$$

where q_{i_k} is the number of samples of the k-th component of the i-th class, p_{i_k} represents the Gaussian pdf of the k-th component of the i-th class, and c_i denotes the number of components in the ith class. Then, according to Equation (14), the geometric mean is taken until the samples from different components are separated in a balanced way.

$$A^* = argmax\left(\prod_{1 \le i_k < j_l \le z} E^A_{i_k j_l}\right)^{\frac{1}{z}} \tag{14}$$

where z is the number of components. Equation (14) is maximized when the E^A values in both desired components are equal. This equation aims to diminish the influence of majority class components while enhancing the impact of minority class components. Consequently, the optimal distance metric matrix A can account for the imbalanced distribution of components. To further achieve this balance and reduce the effect of majority class components while increasing the influence of minority class components, the logarithm of the geometric mean is computed, as indicated in Equation (15).

$$A^* = argmax \ \log\left(\prod_{1 \le i_k < j_l \le z} E^A_{i_k j_l}\right)^{\frac{1}{z}} \tag{15}$$

To enhance the discrimination power of A and achieve a larger margin in Equation (15), the between-class distance must be increased, while the within-class distance should be decreased. To ensure that the proposed distance metric learning method is both appropriate and optimal, the following two constraints are added to the objective function in Equation (16).

$$\max L_A = argmax\ \log\Big(\prod_{1 \le i_k < j_l \le z} E^{A}_{i_k j_l}\Big)^{\frac{1}{z}} + \lambda_1 \sum_{1 \le i_k < j_l \le N} \theta_{i_k j_l} d_A\big(x_{i_k}, x_{j_l}\big) - \lambda_2 \sum_{1 \le i < j \le N} (1-\theta_{ij}) d_A\big(x_i, x_j\big) \tag{16}$$

$$\text{S.t.} \quad A \ge 0$$

$$g(A) = \sum_{1 \le i,j \le N} (1-\theta_{ij}) d_A^2\big(x_i, x_j\big) \le 1$$

In Equation (16), L_A is the objective function, and N is the total number of samples in the training dataset. When $\theta_{ij} = 0$, it means both samples have the same label. λ_1 and λ_2 are regularization parameters that can be adjusted experimentally. Matrix A should be PSD so that it can be decomposed into A = WW^{T}. Therefore, the constraint A $\ge$ 0 is considered. So that the distance between samples of the same class does not fall below a certain value, the limit $g(A)$ is set. The proposed distance metric learning method treats the classes in the imbalanced dataset equally and avoids the tendency of the learning algorithm towards the majority class.

3.1.4. The Fourth Phase: The Optimization Process of the Proposed Objective Function

Since it is difficult to find the distance metric matrix A directly, we try to find the transformation matrix W. First, the eigenvectors of all training samples are considered as the initial value of W. Then, according to Equation (17), to update W, an incremental gradient of L_A is calculated with respect to W to obtain the local maximum value compared to the current solution by stepping in the positive direction of the gradient.

$$W_{new} = W_{old} + \alpha \nabla_w L_A \ , \qquad \alpha = 0.1\ (step\ size) \tag{17}$$

When matrix W is updated, A is set as WW^T. The above operation is repeated until the distance metric matrix A converges.

3.2. Evaluation

In the proposed DMLdbIm method, according to Figure 3, the datasets are mapped to the new space based on the obtained distance metrics, and then traditional classification algorithms are applied to determine the labels of the test samples.

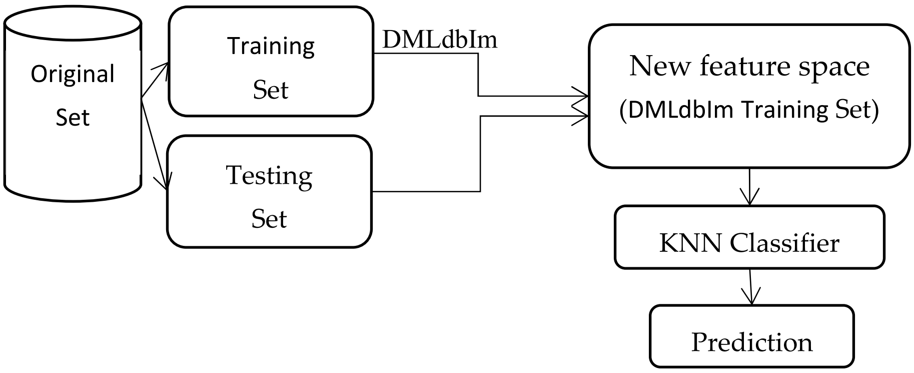


Figure 3. Evaluation of the proposed DMLdbIm model.

4. Experiments

4.1. Specifications of the Datasets

In this research, we utilized 13 imbalanced datasets from KEEL (URL (accessed on 22 July 2024) https://sci2s.ugr.es/keel/imbalanced.php) [31] and 2 imbalanced datasets from UCI (URL (accessed on 22 July 2024) https://archive.ics.uci.edu/ml/index.php) [32], each exhibiting different imbalance ratios, for the evaluation process. The details of these datasets are presented in Table 1. In the experiments, all datasets were transformed into two-class problems using the One-Against-All (OAA) method. For each dataset, we specified the number of samples, the number of features, the number of classes, the number of minority class samples, the distribution of classes, and the imbalance rate (the ratio of the number of majority class samples to the number of minority class samples). Evaluation on these datasets was conducted using 5-fold cross-validation [35].

Table 1. The specifications of 15 evaluation datasets including the size of the datasets, class distributions, and imbalance ratios.

	Dataset	No. of Samples	No. of Features	No. of Minority Sample	Distribution of Classes	Imbalance Ratio
1	Heart (uci)	270	13	120	(55.56,44.44)	1.25
2	WDBC (uci)	569	30	212	(62.74,37.26)	1.68
3	Pima	768	8	268	(65.16,34.84)	1.87
4	Glass0	214	9	70	(67.32,32.68)	2.06
5	Ecoli1	336	7	81	(75.9,24.1)	3.14
6	Ecoli2	336	7	55	(83.63,16.37)	5.1
7	newthyroid1	215	5	35	(83.72,16.28)	5.14
8	Glass6	214	9	29	(86.45,13.55)	6.37
9	Ecoli3	336	7	35	(89.58,10.42)	8.6
10	yeast-2_vs_4	514	8	51	(90.08,9.92)	9.08
11	yeast-1_vs_7	459	7	30	(93.46,6.54)	14.3
12	winequality-red-8_vs_6	656	11	18	(97.26,2.74)	35.44
13	winequality-red-8_vs_6-7	855	11	18	(97.89,2.11)	46.5
14	winequality-white-3-9_vs_5	1482	11	25	(98.31,1.69)	58.28
15	winequality-red-3_vs_5	691	11	10	(98.55,1.45)	68.1

4.2. Evaluation Criteria

In the evaluation stage, the confusion matrix, as shown in Table 2, is utilized. In this matrix, TP (true positive) represents the number of samples accurately classified as belonging to the positive class. TN (true negative) indicates the number of samples correctly classified as belonging to the negative class. FP (false positive) refers to the number of samples incorrectly classified as belonging to the positive class. FN (false negative) denotes the number of samples mistakenly classified as belonging to the negative class.

Table 2. Confusion matrix [36] as the basis for performance comparisons.

		Predicted	
		Positive	**Negative**
Actual	Positive	TP	FN
	Negative	FP	TN

The classification accuracy is obtained from Equation (18).

$$acc = \frac{TP + TN}{TP + FP + TN + FN} \tag{18}$$

To evaluate the effectiveness of classification algorithms on imbalanced datasets, accuracy is not an appropriate criterion, as models tend to favor the majority class, resulting in inflated overall accuracy. Instead, it is more beneficial to focus on the number of positive samples identified from the minority class. For this purpose, recall (sensitivity) and precision criteria are used, as they account for the types of errors (false positives or false negatives) that the model makes. Recall, as defined in Equation (19), compares the number of positive samples to all truly positive samples, while precision, according to Equation (20), compares the number of positive samples to all samples diagnosed as positive. Finally, the F1 measure is utilized as the harmonic mean of these two criteria, as indicated in Equation (21), with the objective of maximizing the F1 value [37,38].

$$Recall = \frac{TP}{TP + FN} \tag{19}$$

$$Precision = \frac{TP}{TP + FP} \tag{20}$$

$$F1 = 2 * \frac{\text{Recall} \times Precision}{\text{Recall} + Precision} \tag{21}$$

4.3. Evaluation of the Proposed DMLdbIm Method

In this article, similar to many existing studies, the k-nearest neighbor classifier is used as the classification method. This classifier is applied to the primary dataset with the Euclidean distance metric and is compared against several methods, including DMBK [26], LMNN [16,17], ITML [21], IML [27], DMLMJ [20], GMML [18], DML_eig [19], and the proposed DMLdbIm method. Below is a brief introduction to each of the compared methods:

- LMNN: this method ensures that, for each training example, its k-nearest neighbors of the same class (target neighbors) are closer than examples from other classes.
- ITML: The objective of this method is to minimize the differential relative entropy between two multivariate Gaussian distributions. It utilizes LogDet regularization to minimize (or maximize) the distance between examples of the same (or different) classes.
- GMML: the goal of this method is to minimize the total distances among similar points.
- DML_eig: this method aims to maximize the minimum squared distance between dissimilar pairs while keeping an upper bound on the total squared distance for similar pairs.
- DMLMJ: the objective here is to learn a linear transformation that maximizes the Jeffrey divergence between two distributions derived from local pairwise constraints.
- IML: this method positions samples of the same class at distances of less than one and samples of different classes at distances greater than one, aiming to reduce the negative effects of class imbalance in the dataset.
- DMBK: the goal of DMBK is to learn a linear transformation that maximizes the logarithm of the geometric mean of the normalized Kullback–Leibler divergence between distributions that share the same covariance Gaussian density.
- DMLdbIm: This proposed method aims to identify components of multivariate Gaussian density with varying covariances for each class. It seeks to maximize the Bhattacharya distance between the Gaussian mixtures of different classes, increasing the distance between external components while decreasing the distance between internal components to create a wide margin between classes.

In the evaluation stage, the parameters for the objective function outlined in Equation (16) are set according to the values specified in Table 3.

Table 3. Initial parameters of the proposed approach.

Name	Value
λ_1	0.00001
λ_2	0.00001
α	0.1
Tol	1×10^{-3}

In Table 4, the F1 measure is used for comparison. The F1 score is computed as the average of 10 runs of five-fold cross-validation on each dataset. Initially, the number of neighbors in the KNN classification is set to 1. The rank of each method on each dataset is indicated in parentheses. The Mean row presents the average efficiency of each method, while the Average Rank row indicates their performance ranking. In the evaluation stage, a method with a higher F1 measure and a lower rank is considered to perform better.

Table 4. Comparison of F1 measures for the 1NN classifier using the proposed DMLdbIm method and other methods on 15 standard datasets. The numbers in parentheses indicate the rank of each approach on the respective dataset. The last two rows show the average performance and average rank of the approaches, providing a useful basis for comparison.

Dataset	Euclidean	ITML	LMNN	DML-eig	GMML	DMLMJ	DMBK	IML	DMLdbIm
Heart	75.15 (6)	77.33 (2)	75.09 (7)	75.06 (8)	73.14 (9)	76.65 (3)	76.08 (4)	75.83 (5)	**77.41** (1)
Wdbc	91.51 (8)	90.21 (9)	91.94 (7)	93.17 (5)	92.28 (6)	**95.56** (1)	94.15 (3)	93.75 (4)	94.81 (2)
Pima	56.34 (4)	54.25 (9)	55.55 (7)	54.75 (8)	57.77 (2)	**58.07** (1)	55.98 (6)	56.3 (5)	56.55 (3)
Glass0	67.61 (6)	63.5 (9)	64.63 (8)	74.1 (4)	68.62 (5)	**75.92** (1)	75.17 (2)	67.15 (7)	74.30 (3)
Ecoli1	73.56 (6)	73.7 (4)	67.9 (9)	69.68 (8)	72.33 (7)	**78.32** (1)	73.58 (5)	74.61 (2)	74.51 (3)
Ecoli2	71.43 (5)	70.28 (8)	71.35 (6)	77.1 (4)	68.65 (9)	**87.12** (1)	78.26 (3)	70.37 (7)	83.93 (2)
Newthyroid1	87.07 (9)	90.25 (8)	**96.77** (1)	93.79 (5)	93.34 (6)	92.52 (7)	94.61 (3)	93.82 (4)	95.96 (2)
Glass6	76.06 (8)	78.53 (5)	78.16 (7)	81.99 (3)	78.33 (6)	70.73 (9)	84.45 (2)	80.32 (4)	**88.72** (1)
Ecoli3	50 (8)	52.98 (6)	54.8 (3)	52.99 (5)	50.4 (7)	58.73 (2)	54.32 (4)	45.66 (9)	**60.74** (1)
Yeast-2_vs_4	69.39 (9)	73.79 (5)	74.45 (4)	70.77 (8)	71.17 (7)	75.54 (2)	75.08 (3)	71.66 (6)	**78.27** (1)
Yeast-1_vs_7	26.09 (7)	23.04 (8)	33.15 (6)	36.51 (4)	37.07 (3)	NAN (9)	39.85 (2)	34.15 (5)	**47.79** (1)
winequality-red-8_vs_6	NA	14.81 (5)	NA	32.12 (3)	25.11 (4)	NA	38.99 (2)	NA	**44.44** (1)
winequality-red-8_vs_6-7	NA	17.25 (3)	NA	25.00 (2)	*12.45* (5)	NA	NA	*16.19* (4)	**40.00** (1)
winequality-white-3-9_vs_5	NA	22.13 (2)	NA	20.00 (3)	*11.57* (5)	NA	NA	*17.07* (4)	**33.33** (1)
winequality-red-3_vs_5	NA	NA	NA	25.00 (2)	*10.78* (3)	NA	NA	NA	**40.00** (1)
Mean	49.61	53.47	50.91	58.80	54.86	51.27	56.03	53.12	**66.05**
AverageRank	6.53	5.8	5.8	4.8	5.6	3.93	3.66	5.06	**1.6**

As shown in Table 4, the proposed approach outperforms most other methods across several datasets. However, the DMLMJ approach has outperformed the proposed method in some cases. DMLMJ is likely to excel over DMLdbIm when dealing with simpler, linearly separable data distributions, where computational efficiency is crucial and where a linear transformation is sufficient to achieve good separability. In contrast, DMLdbIm performs better in complex, multimodal distributions with varying covariances across classes, where capturing intricate internal structures and maximizing class separation in a non-linear feature space are essential. The choice between the two depends on the complexity of the data and the need for either linear or non-linear separability.

The results also indicate that when class imbalance is low, some other methods may perform better than the proposed approach, occasionally placing it in the second or third position. However, as the number of samples in the minority class decreases and imbalance increases, causing existing methods to lose efficiency, the proposed approach maintains its effectiveness. It continues to correctly identify minority class samples, showing an efficiency of about 40%, while some methods, such as Euclidean, LMNN, DMLMJ, and even the DMBK method, fail to function at all, indicated by NA (not available).

On the other hand, when the classifier has less complex decision boundaries (such as with KNN when k > 1), the proposed approach has outperformed the other methods by a wider margin. In Table 5, the F1 measure for 3NN is presented. As seen in this table, by increasing the number of neighbors to 3, the proposed DMLdbIm method shows better performance than other methods, and when the imbalance rate increases, although the efficiency of the algorithm decreases, it still shows better efficiency and more stability than other methods. As shown, the proposed approach surpasses the other methods in 14 cases, with an average rank of 1.4, whereas for the 1NN case (as reported in Table 4), it achieved 9 better results with an average rank of 1.6. These observations confirm the insights from the previous paragraph and provide valuable guidance for potential applications of the proposed approach.

One of the main reasons our method performs better than others is that it considers the data to be a combination of several different distributions with varying means and variances. This approach is particularly effective for classes composed of multiple subgroups with distinct characteristics. Additionally, we use MAP to estimate the parameters of the Gaussian model, which proves to be more effective even when the number of data points in the minority class is small.

Table 6 shows the F1 measure for the 5NN classifier. As Table 6 shows, the efficiency of the proposed method is reduced in comparison with Table 5, but it generally performs better than the other methods on the highly imbalanced datasets and has a better ranking. It also can be concluded that DMLdbIm, IML, and DML_eig are particularly effective in scenarios where class structure and distribution intricacies play a significant role. DMLdbIm excels in complex, multimodal datasets where class distributions involve varying covariances. By focusing on maximizing the Bhattacharya distance between Gaussian mixtures, it effectively separates classes with intricate internal structures. IML is specifically designed to handle class imbalance, outperforming other methods in datasets where class proportions are skewed. Its approach of positioning same-class samples closer and different-class samples farther apart makes it highly effective in maintaining classification performance, even when minority classes are underrepresented. DML_eig, on the other hand, shines in situations where ensuring a wide separation between dissimilar pairs is critical, especially when it is essential to prevent class overlap. By maximizing the minimum squared distance between dissimilar pairs, DML_eig is particularly useful in maintaining clear decision boundaries in datasets with closely spaced classes.

From the other approaches, which had lower performance, LMNN is highly effective in cases where the primary goal is to ensure that k-nearest neighbors of the same class are closer than those of different classes, making it ideal for nearest-neighbor classification tasks with well-defined class clusters. ITML, with its focus on minimizing differential relative entropy and employing LogDet regularization, excels in cases where balancing the distance between similar and dissimilar classes is essential, particularly when the data follow a Gaussian distribution. GMML is best suited for situations where the goal is to minimize the total distances among similar points, making it ideal for tasks that require clustering or grouping data points within the same class while ensuring that similar points remain close. These approaches tend to work well in more traditional classification tasks where the data are less complex and more homogeneous.

Table 5. Comparison of F1 measures for the 3NN classifier using the proposed DMLdbIm method and other methods on 15 standard datasets. The numbers in parentheses indicate the rank of each approach on the respective dataset. The last two rows show the average performance and average rank of the approaches, providing a useful basis for comparison.

Dataset	Euclidean	ITML	LMNN	DML-eig	GMML	DMLMJ	DMBK	IML	DMLdbIm
Heart	72.07 (9)	73.53 (8)	74.31 (7)	77.14 (3)	75.30 (5)	75.80 (4)	*78.23* (2)	74.99 (6)	**79.06** (1)
Wdbc	92.61 (7)	92.27 (9)	92.64 (6)	*94.19* (5)	92.58 (8)	95.37 (2)	*95.31* (3)	94.99 (4)	**97.67** (1)
Pima	58.47 (7)	58.48 (6)	59.57 (4)	58.94 (5)	58.38 (8)	57.04 (9)	60.42 (3)	60.93 (2)	**63.27** (1)
Glass0	70.8 (8)	70.77 (9)	73.02 (5)	**76.12** (1)	71.55 (7)	74.73 (3)	73.01 (6)	73.25 (4)	75.67 (2)
Ecoli1	73.68 (6)	74.33 (5)	72.28 (8)	74.62 (4)	73.11 (7)	80.38 (3)	81.71 (2)	71.31 (9)	**82.76** (1)
Ecoli2	85.60 (4)	81.68 (6)	80.41 (7)	84.95 (5)	78.28 (9)	86.27 (3)	86.68 (2)	80.24 (89)	**89.88** (1)
Newthyroid1	90.51 (9)	**97.15** (1)	95.95 (4)	94.56 (5)	96.37 (3)	93.80 (7)	92.73 (8)	97.14 (2)	94.42 (6)
Glass6	74.88 (8)	75.92 (7)	76.99 (6)	81.57 (2)	74.04 (9)	79.51 (4)	79.87 (3)	77.25 (5)	**83.33** (1)
Ecoli3	53.12 (7)	49.32 (9)	55.22 (5)	*51.19* (8)	53.82 (6)	59.27 (2)	58.77 (3)	58.08 (4)	**60.97** (1)
Yeast-2_vs_4	75.56 (8)	77.85 (5)	78.99 (2)	74.76 (9)	78.12 (4)	76.35 (7)	78.38 (3)	77.32 (6)	**80.56** (1)
Yeast-1_vs_7	22.22 (8)	31.79 (5)	28.49 (6)	*36.61* (4)	21.76 (9)	*39.21* (2)	*39.04* (3)	26.87 (7)	**46.38** (1)
winequality-red-8_vs_6	NA	5.88 (4)	NA	22.22 (2)	8.00 (3)	NA	NA	NA	**39.39** (1)
winequality-red-8_vs_6-7	NA	NA	NA	NA	NA	NA	NA	NA	**28.58** (1)
winequality-white-3-9_vs_5	NA	NA	NA	22.00 (2)	15.78 (3)	NA	NA	3.21 (4)	**28.57** (1)
winequality-red-3_vs_5	NA	NA	NA	NA	NA	NA	NA	NA	**36.71** (1)
Mean	51.30	52.59	52.52	56.59	53.13	54.51	54.94	53.03	**65.80**
Average Rank	6.33	5.53	4.93	3.93	5.66	4	3.46	4.66	**1.4**

Figure 4 shows the average F1 measure of the proposed DMLdbIm method compared to other methods for overall performance comparison. In general, the proposed method outperforms all other methods. Additionally, as observed, the DML-eig approach ranks second on average for the reasons previously mentioned.

Figure 5 shows the accuracy, precision, recall, and F1 criteria of the evaluated approaches when the 3NN classifier is applied to the Heart dataset. As Figure 5 shows, the proposed DMLdbIm method has higher performance in terms of accuracy and precision compared to other methods, and because it has less FP error, negative samples are correctly identified in most cases. This situation arises from a strict classification threshold, which prioritizes accurate positive predictions while potentially missing some true positives, thus lowering recall. Additionally, it shows that the approach is designed to address class imbalance by focusing on precision, so it might effectively reduce false positives but at the cost of capturing fewer positive instances overall. This trade-off suggests that while the method is accurate in its positive classifications, it may not cover all positive cases, reflecting a balance between precision and recall.

Table 6. Comparison of F1 measures for the 5NN classifier using the proposed DMLdbIm method and other methods on 15 standard datasets. The numbers in parentheses indicate the rank of each approach on the respective dataset. The last two rows show the average performance and average rank of the approaches, providing a useful basis for comparison.

Dataset	Euclidean	ITML	LMNN	DML-eig	GMML	DMLMJ	DMBK	IML	DMLdbIm
Heart	75.44 (8)	75.72 (7)	79.1 (3)	**79.94** (1)	77.60 (6)	73.48 (9)	78.14 (5)	79.63 (2)	78.62 (4)
Wdbc	95.16 (4)	92.12 (9)	94.32 (8)	94.98 (5)	95.32 (3)	94.43 (7)	94.58 (6)	95.67 (2)	**96.58** (1)
Pima	57.52 (8)	61.05 (3)	57.32 (9)	60.48 (5)	57.65 (6)	57.53 (7)	60.75 (4)	**61.78** (1)	61.08 (2)
Glass0	71.28 (7)	66.24 (9)	71.75 (4)	**75.85** (1)	70.45 (8)	71.54 (5)	73.83 (3)	71.46 (6)	74.10 (2)
Ecoli1	69.44 (7)	69.69 (6)	65 (8)	**83.12** (1)	70.52 (5)	82.27 (2)	78.72 (4)	63.62 (9)	79.99 (3)
Ecoli2	85.96 (7)	83.01 (9)	**91.34** (1)	86.30 (6)	85.21 (8)	88.45 (3)	86.36 (5)	87.02 (4)	88.74 (2)
Newthyroid1	84.73 (9)	89.28 (6)	97.24 (2)	88.03 (8)	92.16 (4)	88.96 (7)	90.92 (5)	**98.92** (1)	92.27 (3)
Glass6	74.1 (9)	82.9 (5)	86.37 (3)	85.28 (4)	89.39 (2)	79.32 (7)	78.06 (8)	**91.2** (1)	79.87 (6)
Ecoli3	55.69 (9)	57.82 (8)	60.78 (7)	63.25 (3)	62.56 (4)	60.94 (6)	62.17 (5)	64.65 (2)	**65.43** (1)
Yeast-2_vs_4	75 (8)	69.33 (9)	76.34 (5)	78.01 (2)	75.9 (6)	75.45 (7)	77.25 (4)	**78.24** (1)	77.38 (3)
Yeast-1_vs_7	NA	14.42 (6)	3.83 (8)	29.39 (3)	23.38 (5)	30.12 (2)	27.14 (4)	10.77 (7)	**35.39** (1)
winequality-red-8_vs_6	NA	NA	NA	NA	NA	NA	NA	NA	**28.57** (1)
winequality-red-8_vs_6-7	NA	NA	NA	NA	NA	NA	NA	NA	**25** (1)
winequality-white-3-9_vs_5	NA	NA	NA	25 (2)	7.4 (4)	NA	NA	10.29 (3)	**27.14** (1)
winequality-red-3_vs_5	NA	1.19 (3)	NA	NA	NA	NA	NA	9.36 (2)	**31.08** (1)
Mean	49.62	50.85	52.22	56.64	53.83	53.49	53.86	54.84	**62.74**
Average Rank	6.53	5.93	4.73	3.26	4.6	5	4.4	3	**2.13**

Figures 6–19 illustrate the detailed performance of various algorithms on different datasets using the 3NN classifier. As shown in Figure 6, the proposed approach significantly outperforms the other methods on the WDBC dataset, indicating its superiority. We hypothesize that the multimodal representation of the classes contributes to this improved performance.

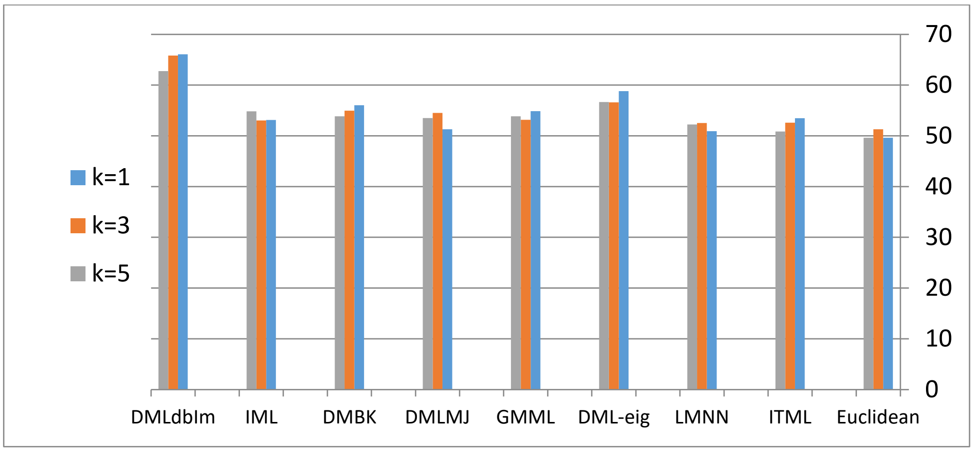
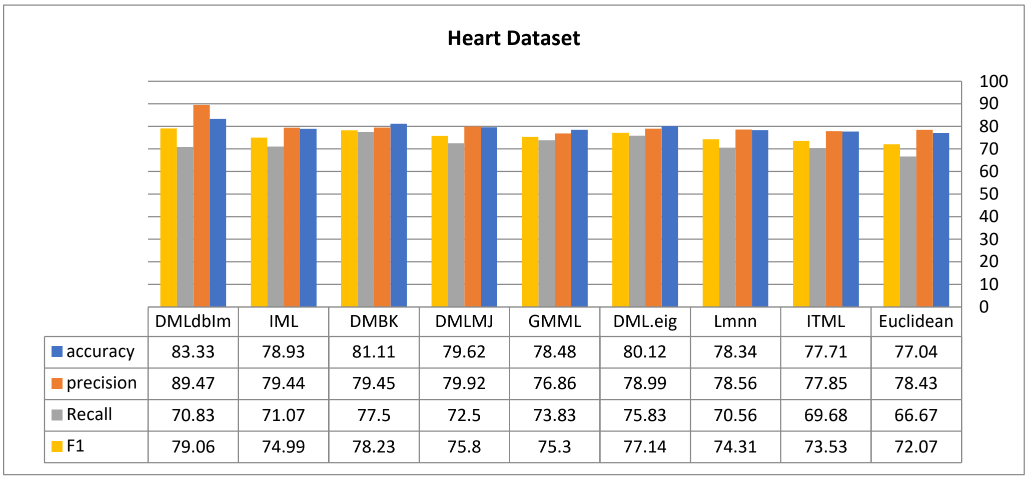
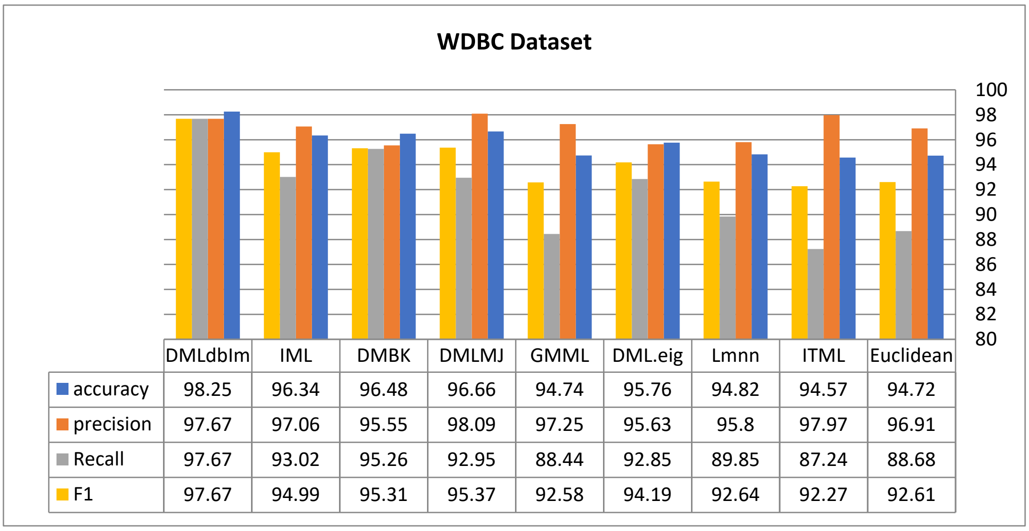


Figure 4. Comparison of the mean F1 measure for different neighborhood sizes in the KNN algorithm. k represents the number of nearest neighbors used. Blue denotes k = 1, while red and gray denote k = 3 and k = 5, respectively.



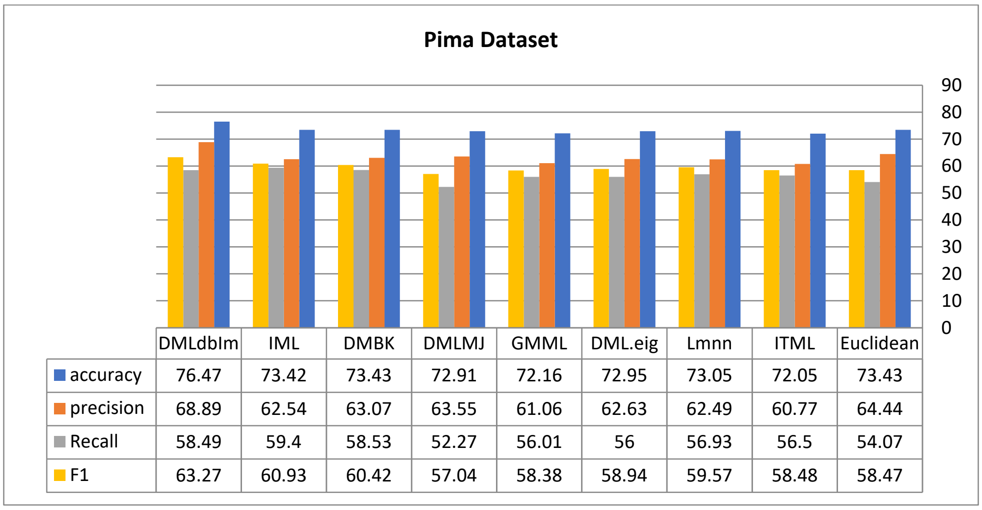
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	83.33	78.93	81.11	79.62	78.48	80.12	78.34	77.71	77.04
🟧 precision	89.47	79.44	79.45	79.92	76.86	78.99	78.56	77.85	78.43
◼ Recall	70.83	71.07	77.5	72.5	73.83	75.83	70.56	69.68	66.67
🟨 F1	79.06	74.99	78.23	75.8	75.3	77.14	74.31	73.53	72.07

Figure 5. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Heart dataset.



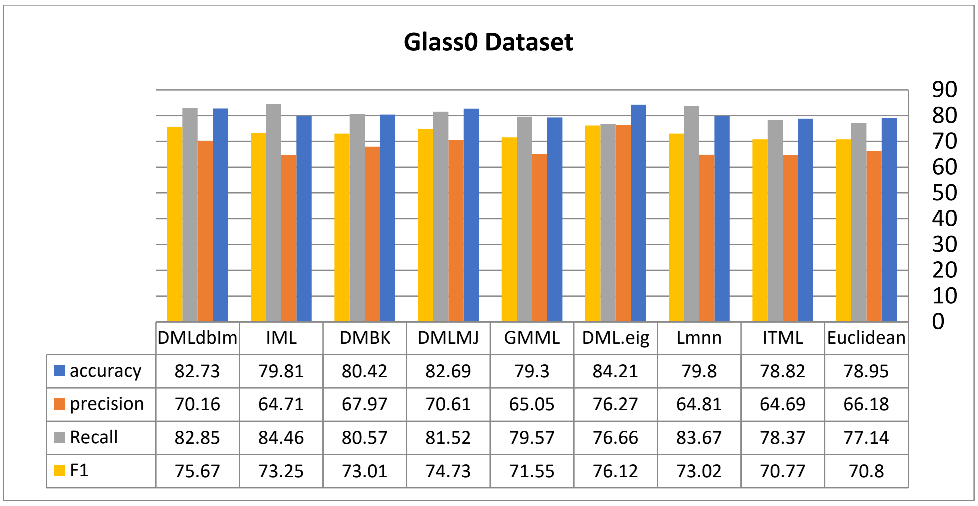
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	98.25	96.34	96.48	96.66	94.74	95.76	94.82	94.57	94.72
🟧 precision	97.67	97.06	95.55	98.09	97.25	95.63	95.8	97.97	96.91
◼ Recall	97.67	93.02	95.26	92.95	88.44	92.85	89.85	87.24	88.68
🟨 F1	97.67	94.99	95.31	95.37	92.58	94.19	92.64	92.27	92.61

Figure 6. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on WDBC dataset.



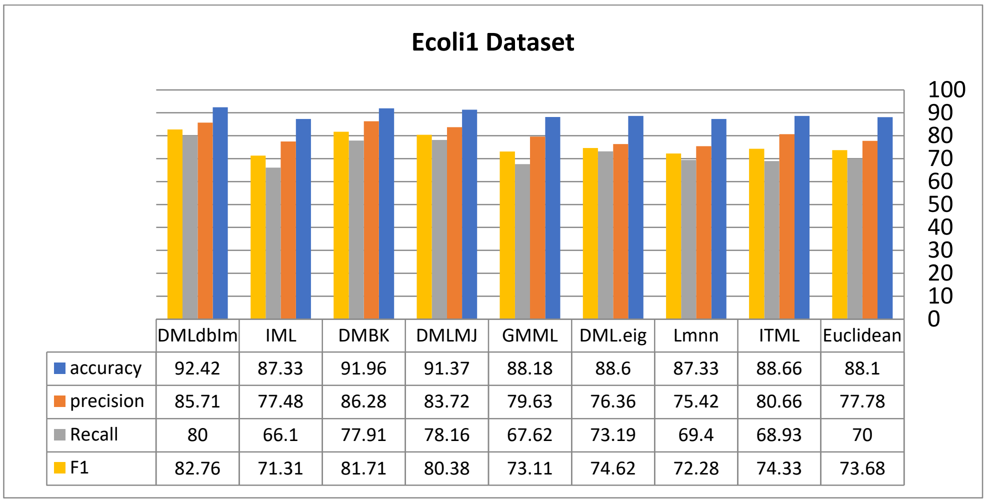
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	76.47	73.42	73.43	72.91	72.16	72.95	73.05	72.05	73.43
🟧 precision	68.89	62.54	63.07	63.55	61.06	62.63	62.49	60.77	64.44
■ Recall	58.49	59.4	58.53	52.27	56.01	56	56.93	56.5	54.07
🟨 F1	63.27	60.93	60.42	57.04	58.38	58.94	59.57	58.48	58.47

Figure 7. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Pima dataset.



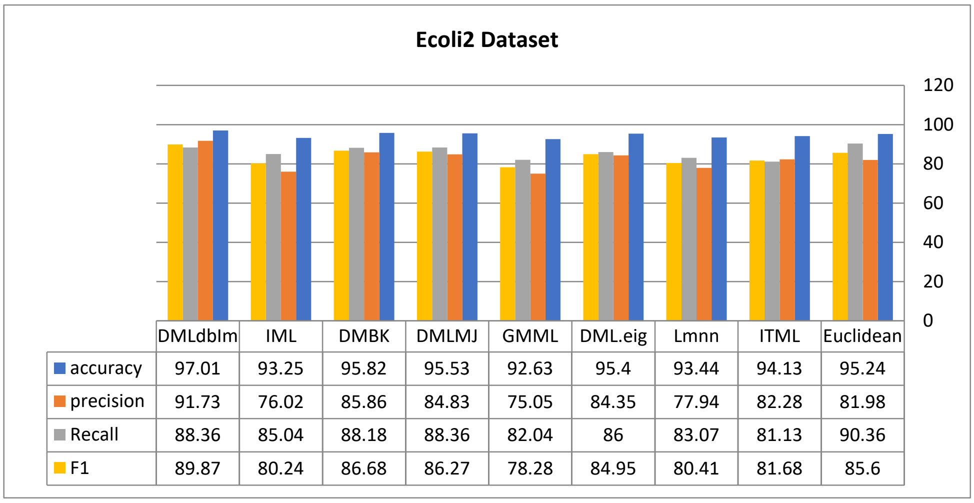
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	82.73	79.81	80.42	82.69	79.3	84.21	79.8	78.82	78.95
🟧 precision	70.16	64.71	67.97	70.61	65.05	76.27	64.81	64.69	66.18
■ Recall	82.85	84.46	80.57	81.52	79.57	76.66	83.67	78.37	77.14
🟨 F1	75.67	73.25	73.01	74.73	71.55	76.12	73.02	70.77	70.8

Figure 8. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Glass0 dataset.



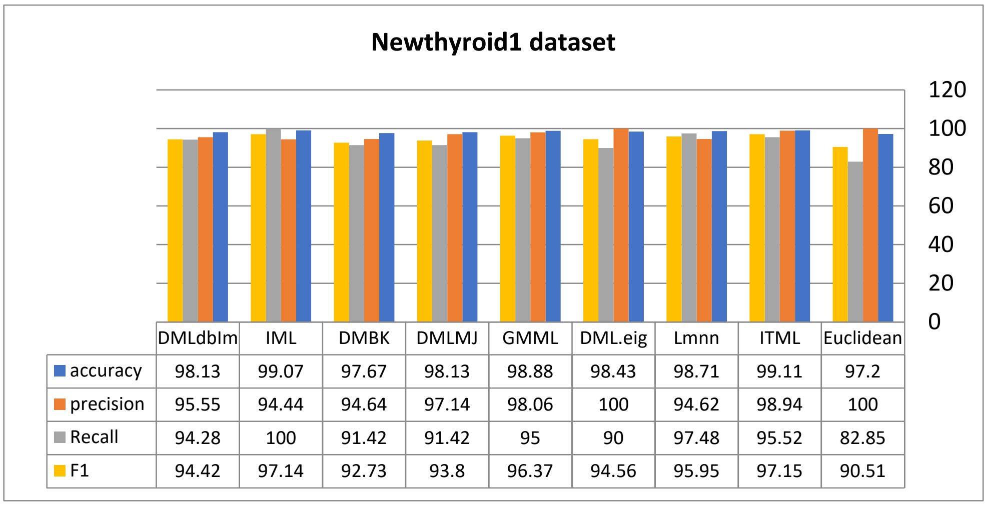
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	92.42	87.33	91.96	91.37	88.18	88.6	87.33	88.66	88.1
🟧 precision	85.71	77.48	86.28	83.72	79.63	76.36	75.42	80.66	77.78
■ Recall	80	66.1	77.91	78.16	67.62	73.19	69.4	68.93	70
🟨 F1	82.76	71.31	81.71	80.38	73.11	74.62	72.28	74.33	73.68

Figure 9. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Ecoli1 dataset.



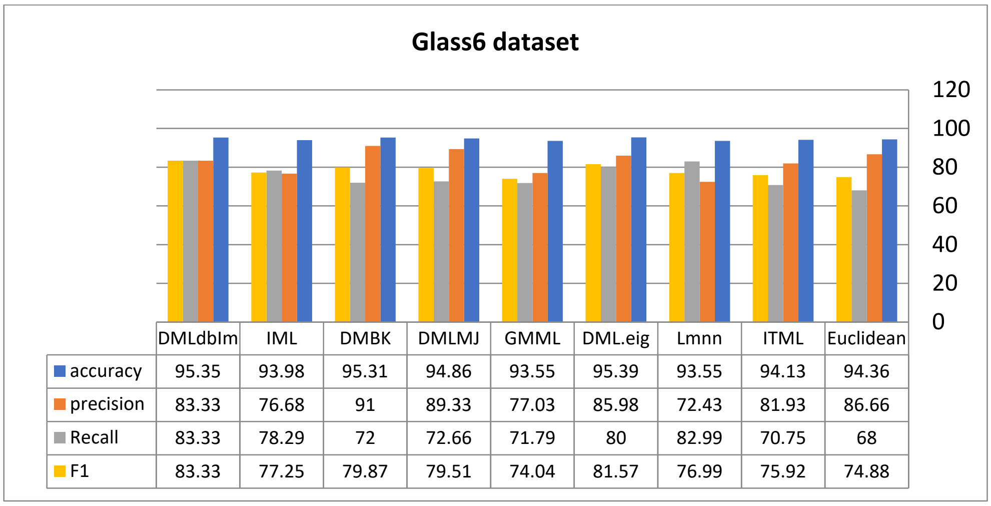
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
accuracy	97.01	93.25	95.82	95.53	92.63	95.4	93.44	94.13	95.24
precision	91.73	76.02	85.86	84.83	75.05	84.35	77.94	82.28	81.98
Recall	88.36	85.04	88.18	88.36	82.04	86	83.07	81.13	90.36
F1	89.87	80.24	86.68	86.27	78.28	84.95	80.41	81.68	85.6

Figure 10. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Ecoli2 dataset.



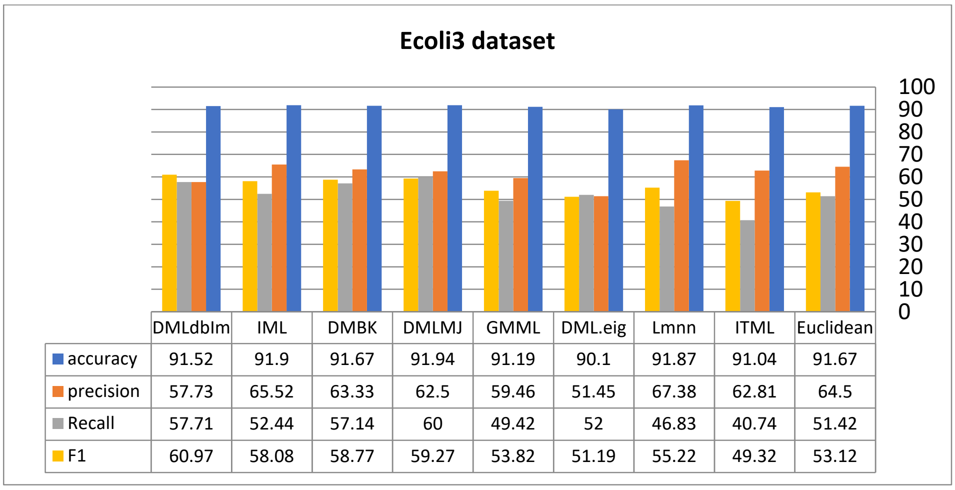
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
accuracy	98.13	99.07	97.67	98.13	98.88	98.43	98.71	99.11	97.2
precision	95.55	94.44	94.64	97.14	98.06	100	94.62	98.94	100
Recall	94.28	100	91.42	91.42	95	90	97.48	95.52	82.85
F1	94.42	97.14	92.73	93.8	96.37	94.56	95.95	97.15	90.51

Figure 11. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Newthyroid1 dataset.



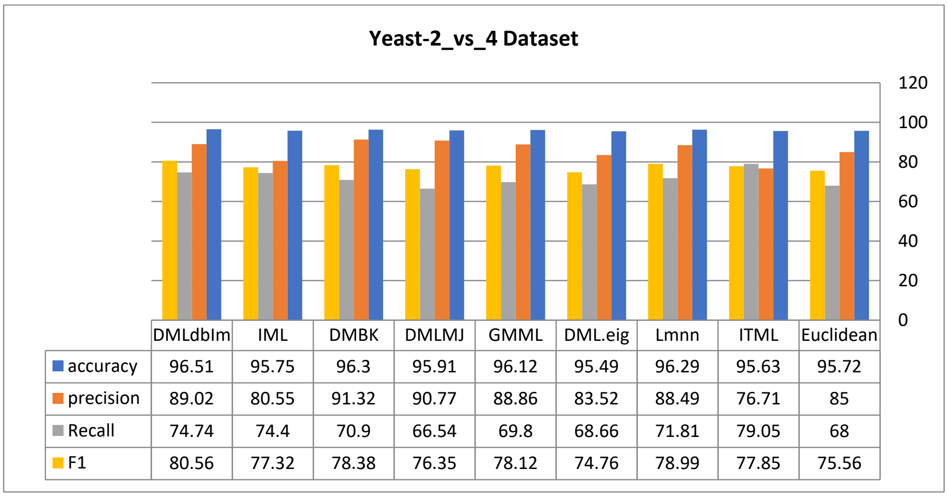
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
accuracy	95.35	93.98	95.31	94.86	93.55	95.39	93.55	94.13	94.36
precision	83.33	76.68	91	89.33	77.03	85.98	72.43	81.93	86.66
Recall	83.33	78.29	72	72.66	71.79	80	82.99	70.75	68
F1	83.33	77.25	79.87	79.51	74.04	81.57	76.99	75.92	74.88

Figure 12. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Glass6 dataset.



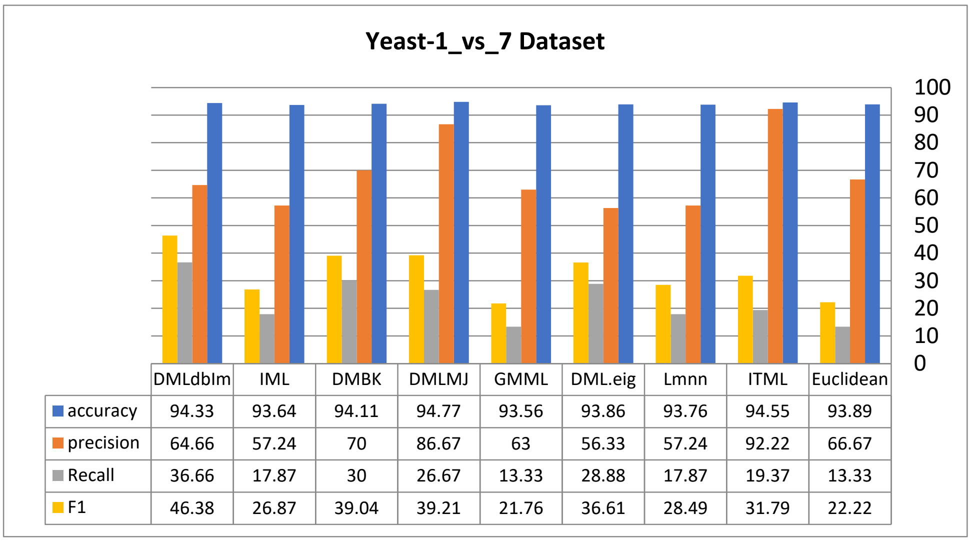
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	91.52	91.9	91.67	91.94	91.19	90.1	91.87	91.04	91.67
🟧 precision	57.73	65.52	63.33	62.5	59.46	51.45	67.38	62.81	64.5
⬜ Recall	57.71	52.44	57.14	60	49.42	52	46.83	40.74	51.42
🟨 F1	60.97	58.08	58.77	59.27	53.82	51.19	55.22	49.32	53.12

Figure 13. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Ecoli3 dataset.



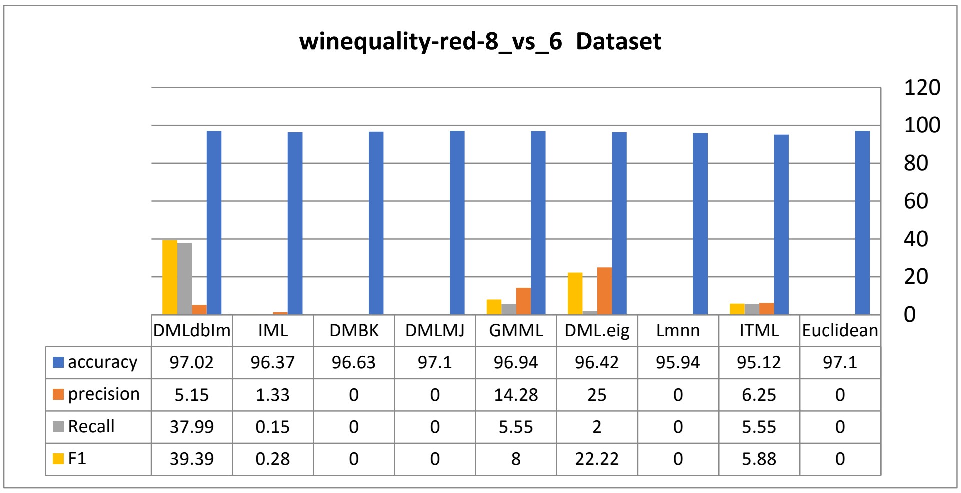
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	96.51	95.75	96.3	95.91	96.12	95.49	96.29	95.63	95.72
🟧 precision	89.02	80.55	91.32	90.77	88.86	83.52	88.49	76.71	85
⬜ Recall	74.74	74.4	70.9	66.54	69.8	68.66	71.81	79.05	68
🟨 F1	80.56	77.32	78.38	76.35	78.12	74.76	78.99	77.85	75.56

Figure 14. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Yeast-2_vs_4 dataset.



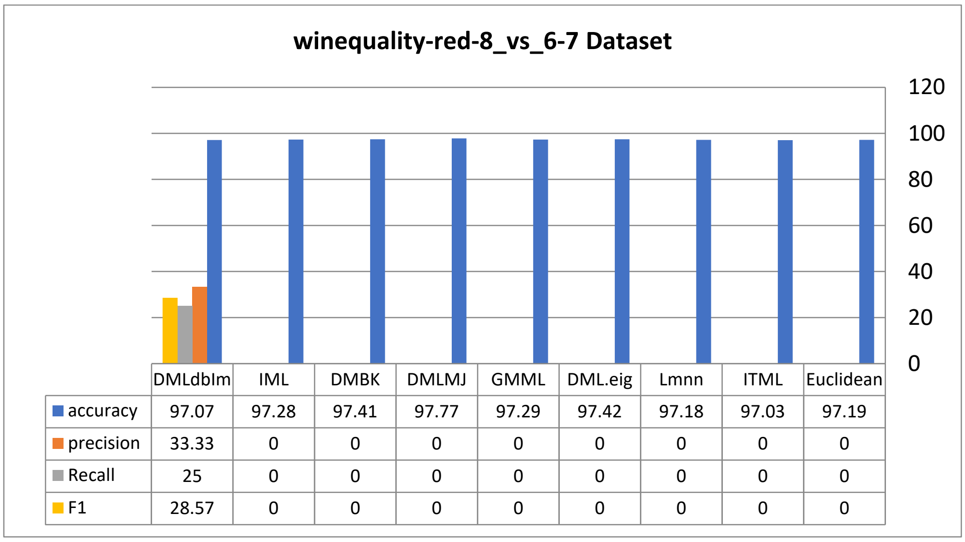
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	94.33	93.64	94.11	94.77	93.56	93.86	93.76	94.55	93.89
🟧 precision	64.66	57.24	70	86.67	63	56.33	57.24	92.22	66.67
⬜ Recall	36.66	17.87	30	26.67	13.33	28.88	17.87	19.37	13.33
🟨 F1	46.38	26.87	39.04	39.21	21.76	36.61	28.49	31.79	22.22

Figure 15. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on Yeast-1_vs_7 dataset.



	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
■ accuracy	97.02	96.37	96.63	97.1	96.94	96.42	95.94	95.12	97.1
■ precision	5.15	1.33	0	0	14.28	25	0	6.25	0
■ Recall	37.99	0.15	0	0	5.55	2	0	5.55	0
■ F1	39.39	0.28	0	0	8	22.22	0	5.88	0

Figure 16. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on winequality-red-8_vs_6 dataset. As seen, the other approaches actually fail in recognizing the minority class.



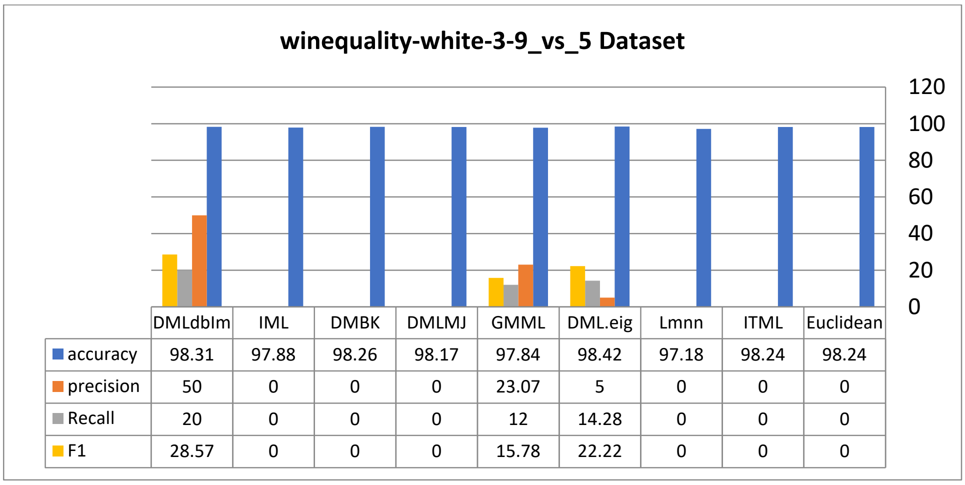
	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
■ accuracy	97.07	97.28	97.41	97.77	97.29	97.42	97.18	97.03	97.19
■ precision	33.33	0	0	0	0	0	0	0	0
■ Recall	25	0	0	0	0	0	0	0	0
■ F1	28.57	0	0	0	0	0	0	0	0

Figure 17. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on winequality-red-8_vs_7 dataset. As seen, the other approaches actually fail in recognizing the minority class.

Similar observations are noted in Figure 7 and with the Pima dataset. The proposed approach again shows a significant improvement over other methods, though precision remains considerably higher than recall. This reinforces the previous findings. However, the lower recall could be a drawback in medical datasets, possibly due to a high imbalance ratio. To address this issue, oversampling techniques could be employed to reduce the imbalance and potentially improve recall.

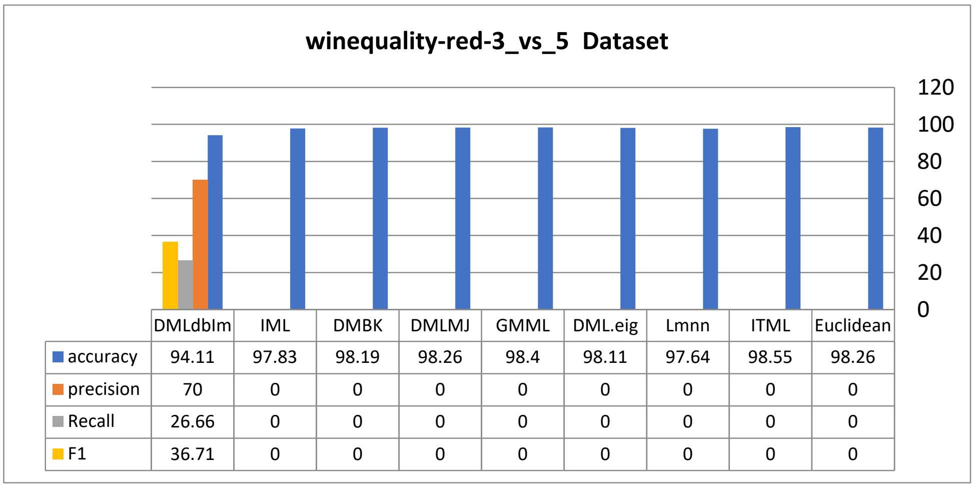
The average superiority of the proposed approach is also evident in Figures 8 and 9. Comparing these figures, the proposed approach surpasses the other methods by a wider margin in Figure 9, which is attributed to the higher imbalance ratio of the Ecoli1 dataset. This demonstrates the approach's effectiveness in handling highly imbalanced problems. The same observations can be seen in Figure 10 and on the Ecoli2 dataset, which has an imbalance ratio of 5.1.

Figure 11 suggests that the proposed approach fails in comparison with other approaches. Besides the aforementioned reasons, this observation may be due to the prior assumptions on the parameters, especially the number of components.



	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	98.31	97.88	98.26	98.17	97.84	98.42	97.18	98.24	98.24
🟧 precision	50	0	0	0	23.07	5	0	0	0
⬜ Recall	20	0	0	0	12	14.28	0	0	0
🟨 F1	28.57	0	0	0	15.78	22.22	0	0	0

Figure 18. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on winequality-white-3-9_vs_5 dataset. As seen, the other approaches actually fail in recognizing the minority class.



	DMLdbIm	IML	DMBK	DMLMJ	GMML	DML.eig	Lmnn	ITML	Euclidean
🟦 accuracy	94.11	97.83	98.19	98.26	98.4	98.11	97.64	98.55	98.26
🟧 precision	70	0	0	0	0	0	0	0	0
⬜ Recall	26.66	0	0	0	0	0	0	0	0
🟨 F1	36.71	0	0	0	0	0	0	0	0

Figure 19. Comparison of the accuracy, precision, recall, and F1 of the evaluated approaches with 3NN classifier on winequality-red-3_vs_5 dataset. As seen, the other approaches actually fail in recognizing the minority class.

Figures 12–14 suggest the better performance of the proposed approach on the Glass6, Ecoli3, and Yeast-2_vs_4 datasets. However, the performance superiority of the proposed approach is more visible in Figures 15–19, which are categorized under highly imbalanced problems. Even for winequality-red-8_vs_6, winequality-red-8_vs_7, winequality-white-3-9_vs_5, and winequality-red-3_vs_5, which are very highly imbalanced with imbalance ratios of 35.44, 46.5, 58.28, and 68.1, respectively, most of the previous approaches have failed to provide results. In contrast, the proposed approach has achieved acceptable outcomes, demonstrating both its robustness against high imbalance and its applicability in such problems.

As seen in Figures 5–19, when the imbalance ratio increases, methods that ignore the minority class still have higher accuracy. This is why the accuracy criterion is not suitable for evaluating the efficiency of classification algorithms on imbalanced datasets. However, the proposed DMLdbIm method is still able to identify minority class samples in cases of higher imbalance ratios. In Table 7, the mean and rank of accuracy, precision, recall, and F1 measures of different methods for 3NN classification are given. According to Table 7, the proposed DMLdbIm method generally has higher accuracy, precision, recall, and F1 and better ranking than other methods.

Table 7. Comparison of average accuracy, precision, recall, and F1 of 3NN classifier with the proposed DMLdbIm method and other methods on 15 standard datasets.

Evaluation Criteria	Euclidean	ITML	LMNN	DML-eig	GMML	DMLMJ	DMBK	IML	DMLdbIm
Average Accuracy	91.40	91.28	91.31	92.04	90.98	92.37	92.31	91.51	**92.83**
Average Precision	57.57	58.87	55.67	61.56	58.57	60.44	59.23	55.53	**73.25**
Average Recall	48.7	50.18	51.36	54.33	50.96	52.19	53.45	55.8	**62.3**
Average F1 Measure	51.30	52.59	52.52	56.59	53.13	54.51	54.94	53.03	**65.80**
Accuracy Average Rank	5.53	6.13	6.26	4.6	6.46	3.33	3.4	5.73	**3.13**
Precision Average Rank	4.4	4.8	5.8	4.53	5.26	3.01	3.53	5.53	**3**
Recall Average Rank	6.26	5.66	4.66	4.4	5.4	4.53	3.6	3.46	**2.13**
F1 Average Rank	6.33	5.53	4.93	3.93	5.66	4	3.46	4.66	**1.4**

4.4. Comparison with Deep Learning

In the continuation of the experiments, we compared our proposed method with another method that uses a deep neural network on an unbalanced dataset. Wang et al. [34], discussed in the related works section, present an imbalance classification method based on deep learning and fuzzy support vector machine, named DFSVM. This method first uses a deep neural network to obtain an embedding representation of the data. The deep neural network is trained using triplet loss to enhance similarities within classes and differences between classes. In Table 8, the F1 measure for the 3NN is shown.

Table 8. Comparison of F1 measure of 3NN classifier for the proposed DMLdbIm method and DFSVM method on evaluation datasets.

#	Dataset	No. of Samples	No. of Feature	Imbalance Ratio	DFSVM	DMLdbIm
1	Glass1	214	9	1.82	62.9	**64.52**
2	Glass6	214	9	6.38	83.1	**90.91**
3	Yeast1vs7	459	8	14.3	47.6	**47.79**
4	Yeast3	1484	8	8.1	**78.1**	67.74
5	Yeast6	1484	8	41.4	50.1	**83.33**
6	Ecoli0147vs2356	336	7	10.56	71.6	**86.15**
7	Ecoli01vs235	244	7	9.17	80.9	**90.91**
8	Ecoli0267vs35	224	7	9.18	76.4	**80.23**
9	Vehcle3	846	18	2.99	73.7	**81.53**
10	Pageblocks0	5472	10	8.79	**80.5**	79.21
		Average			63.29	**77.23**

As can be seen in Table 8, the proposed DMLdbIm method performs better than the DFSVM method in most cases, demonstrating the effectiveness of the approach even when compared to a recent deep learning-based method. Table 8 shows that the deep learning-based approach is very sensitive to the amount of available data and has superior performance compared to the proposed approach only when a good amount of data are available. In other cases, the proposed approach has outperformed the DFSVM approach by a very wide margin.

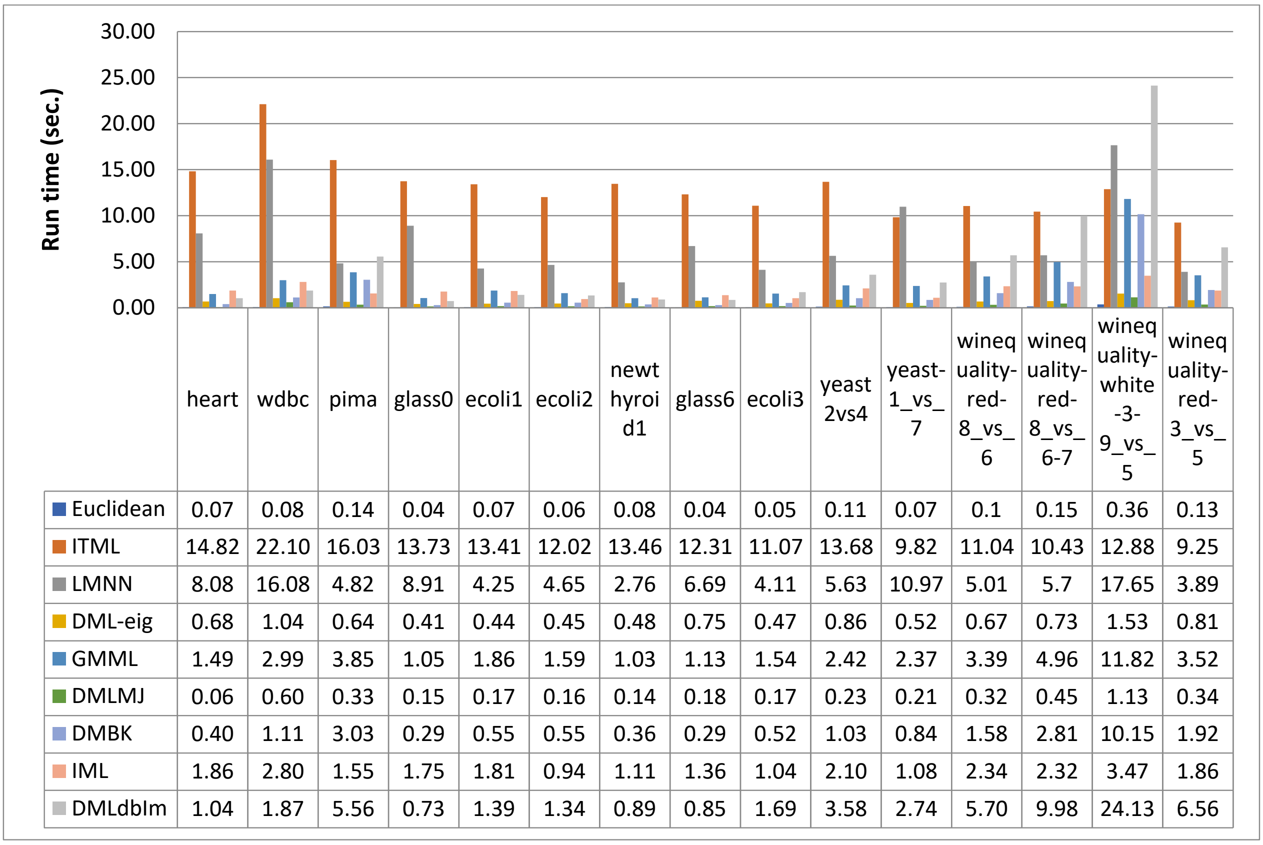
4.5. Computational Complexity Analysis

The computational complexity of the proposed DMLdbIm method is analyzed in this section. The approach starts with DBSCAN clustering. With N datapoints, the time complexity of the DBSCAN algorithm is O(NlogN). Additionally, the time complexity of GMM construction is O(N.K.D^3), where K represents the number of Gaussian components. Subsequently, the covariance matrices are updated with a time complexity of O(K.N.D^2). The generalized eigenvalue decomposition of the covariance matrices can be performed in O(D^3).

Following the previously described phases, the optimization phase of L_A commences. This phase involves calculating the Bhattacharya distance between two probability distributions with a computational complexity of O(K^2.C^2), where C represents the number of classes. This is followed by the computation of L_A, which has a complexity of O(N^2). If we assume there are I iterations in the optimization process, the overall computational complexity for this phase will be O(I.(K^2.C^2 + N^2)). Given that N is significantly larger than both K and C, the computational complexity can be simplified to O(I.N^2).

With the aforementioned steps in mind, the overall time complexity of the proposed method is O(NlogN + N.K.D^3 + K.N.D^2 + D^3 + I.N^2). Since the first five steps are executed only once, their computational complexity can be considered negligible. Consequently, we can conclude that the computational complexity of the proposed method is O(I.N^2), which is primarily influenced by the number of data points. This complexity is comparable to that of some earlier works, such as DMLMJ [20], which has a complexity of O(N^2.D + k.N^3), where k denotes the number of k-nearest neighbors.

In Figure 20, the execution time of different methods on various datasets is given in seconds. All methods were implemented in MATLAB R2015b (64-bit) on a personal laptop with an Intel Core i5 processor (2.30 GHz) and 4 GB of main memory, running the Windows 7 operating system.



	heart	wdbc	pima	glass0	ecoli1	ecoli2	newthyroid1	glass6	ecoli3	yeast2vs4	yeast-1_vs_7	winequality-red-8_vs_6	winequality-red-8_vs_6-7	winequality-white-3-9_vs_5	winequality-red-3_vs_5
■ Euclidean	0.07	0.08	0.14	0.04	0.07	0.06	0.08	0.04	0.05	0.11	0.07	0.1	0.15	0.36	0.13
■ ITML	14.82	22.10	16.03	13.73	13.41	12.02	13.46	12.31	11.07	13.68	9.82	11.04	10.43	12.88	9.25
■ LMNN	8.08	16.08	4.82	8.91	4.25	4.65	2.76	6.69	4.11	5.63	10.97	5.01	5.7	17.65	3.89
■ DML-eig	0.68	1.04	0.64	0.41	0.44	0.45	0.48	0.75	0.47	0.86	0.52	0.67	0.73	1.53	0.81
■ GMML	1.49	2.99	3.85	1.05	1.86	1.59	1.03	1.13	1.54	2.42	2.37	3.39	4.96	11.82	3.52
■ DMLMJ	0.06	0.60	0.33	0.15	0.17	0.16	0.14	0.18	0.17	0.23	0.21	0.32	0.45	1.13	0.34
■ DMBK	0.40	1.11	3.03	0.29	0.55	0.55	0.36	0.29	0.52	1.03	0.84	1.58	2.81	10.15	1.92
■ IML	1.86	2.80	1.55	1.75	1.81	0.94	1.11	1.36	1.04	2.10	1.08	2.34	2.32	3.47	1.86
■ DMLdbIm	1.04	1.87	5.56	0.73	1.39	1.34	0.89	0.85	1.69	3.58	2.74	5.70	9.98	24.13	6.56

Figure 20. The average execution time of different distance metric learning methods (in seconds) with 3NN as the classifier.

As Figure 20 suggests, although the proposed approach is expected to have a high computational cost compared to some other approaches, the computational speed of the proposed DMLdbIm method is several times faster than commonly used methods such as ITML and LMNN.

5. Conclusions

This article proposes a new distance metric learning approach called DMLdbIm, which is most efficient on highly imbalanced datasets. In the proposed method, the distribution of classes is assumed to be in the form of a mixture of Gaussians. This assumption is based on the possibility that the data are composed of several normal distributions with different parameters, each having a different mean and variance. The proposed approach aims to increase the discrimination power of the learned metric by increasing the between-class distance of the external Gaussian components and decreasing the distance between the internal ones. For this purpose, MAP estimation is used to calculate the parameters of the components, even when the number of samples in a class is very small.

In the experiments, the proposed DMLdbIm method shows better performance compared to other distance metric learning methods in increasing the efficiency of the k-nearest neighbor classifier, especially when the imbalance ratio increases. In these cases, when other methods are not effective at all, the proposed method provides acceptable performance. Therefore, in fields where accurately identifying exceptional cases is highly important, this capability is extremely valuable, for example, in detecting low-prevalence cancers, identifying real financial fraud (where mistakes can lead to huge financial losses), and detecting cybercrimes in the virtual space. Additionally, the approach has a higher speed than some commonly used methods. In the future, one may plan to use non-linear and kernel-based variations of the proposed distance metric learning model. Also, since deep metric learning is a relatively new field in the literature, it would be worthwhile to implement the proposed approach with a deep learning architecture. Deep neural networks can find complex patterns in data, which helps in identifying scarce samples. Deep models with retraining capabilities can adapt well to scarce data.

Author Contributions: A.J.M. implemented the approach, performed the analyses, and also prepared the initial draft. M.H.M. proposed the original idea, supervised the process, and proofread the manuscript. H.G. is the advisor and helped with the completion of the manuscript. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The source code and all necessary data for this article are shared via the link below: (accessed on 22 July 2024) https://github.com/Mohammad-H-Moattar/Supervised-density-based-metric-learning-.git.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

1. Russell, S.J.; Norvig, P. *Artificial Intelligence: A Modern Approach*; Pearson Education Limited: Kuala Lumpur, Malaysia, 2016.
2. Duin, R.P.; Tax, D.M.J. Statistical pattern recognition. In *Handbook of Pattern Recognition and Computer Vision*; World Scientific Pub Co Inc: Hackensack, NJ, USA, 2005; pp. 3–24.
3. He, H.; Ma, Y. (Eds.) *Imbalanced Learning: Foundations, Algorithms, and Applications*; John Wiley & Sons: Hoboken, NJ, USA, 2013.
4. He, H.; Garcia, E.A. Learning from imbalanced data. *IEEE Trans. Knowl. Data Eng.* **2008**, *21*, 1263–1284.
5. Ali, A.; Shamsuddin, S.M.; Ralescu, A.L. Classification with class imbalance problem: A review. *Int. J. Adv. Soft Comput. Its Appl.* **2015**, *7*, 176–204.
6. Nguyen, G.H.; Bouzerdoum, A.; Phung, S.L. Learning pattern classification tasks with imbalanced datasets. In *Pattern Recognition*; InTech: Houston, TX, USA, 2009.
7. Mazurowski, M.A.; Habas, P.A.; Zurada, J.M.; Lo, J.Y.; Baker, J.A.; Tourassi, G.D. Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Netw.* **2008**, *21*, 427–436. [CrossRef] [PubMed]

8. Wei, W.; Li, J.; Cao, L.; Ou, Y.; Chen, J. Effective detection of sophisticated online banking fraud on extremely imbalanced data. *World Wide Web* **2013**, *16*, 449–475. [CrossRef]
9. Li, Y.; Sun, G.; Zhu, Y. Data imbalance problem in text classification. In Proceedings of the Information Processing (ISIP), 2010 Third International Symposium on Information Processing, Qingdao, China, 15–17 October 2010; IEEE: Piscataway, NJ, USA; pp. 301–305.
10. Zhu, Z.B.; Song, Z.H. Fault diagnosis based on imbalance modified kernel Fisher discriminant analysis. *Chem. Eng. Res. Des.* **2010**, *88*, 936–951. [CrossRef]
11. Tavallaee, M.; Stakhanova, N.; Ghorbani, A.A. Toward credible evaluation of anomaly-based intrusion-detection methods. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2010**, *40*, 516–524. [CrossRef]
12. Kotsiantis, S.; Kanellopoulos, D.; Pintelas, P. Handling imbalanced datasets: A review. *GESTS Int. Trans. Comput. Sci. Eng.* **2006**, *30*, 25–36.
13. Xing, E.P.; Jordan, M.I.; Russell, S.J.; Ng, A.Y. Distance metric learning with application to clustering with side-information. In *Advances in Neural Information Processing Systems*; Mit Pr: Cambridge, MA, USA, 2003; pp. 521–528.
14. Bellet, A.; Habrard, A.; Sebban, M. *Metric Learning*; Synthesis Lectures on Artificial Intelligence and Machine Learning; Springer: Berlin/Heidelberg, Germany, 2015; Volume 9, pp. 1–151.
15. Li, D.; Tian, Y. Survey and experimental study on metric learning methods. *Neural Netw.* **2018**, *105*, 447–462. [CrossRef]
16. Weinberger, K.Q.; Blitzer, J.; Saul, L.K. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems*; Mit Pr: Cambridge, MA, USA, 2006; pp. 1473–1480.
17. Weinberger, K.Q.; Saul, L.K. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.* **2009**, *10*, 207–244.
18. Zadeh, P.; Hosseini, R.; Sra, S. Geometric mean metric learning. In Proceedings of the International Conference on Machine Learning, PMLR, New York City, NY, USA, 19–24 June 2016; pp. 2464–2471.
19. Ying, Y.; Li, P. Distance metric learning with eigenvalue optimization. *J. Mach. Learn. Res.* **2012**, *13*, 1–26.
20. Nguyen, B.; Morell, C.; De Baets, B. Supervised distance metric learning through maximization of the Jeffrey divergence. *Pattern Recognit.* **2017**, *64*, 215–225. [CrossRef]
21. Davis, J.V.; Kulis, B.; Jain, P.; Sra, S.; Dhillon, I.S. Information-theoretic metric learning. In Proceedings of the 24th international conference on Machine learning, Corvallis, OR, USA, 17–24 June 2007; ACM: New York, NY, USA; pp. 209–216.
22. Chang, C.C. A boosting approach for supervised Mahalanobis distance metric learning. *Pattern Recognit.* **2012**, *45*, 844–862. [CrossRef]
23. Zhong, G.; Zheng, Y.; Li, S.; Fu, Y. SLMOML: Online Metric Learning With Global Convergence. *IEEE Trans. Circuits Syst. Video Technol.* **2018**, *28*, 2460–2472. [CrossRef]
24. Liu, W.; Tsang, I.W. Large Margin Metric Learning for Multi-Label Prediction. In Proceedings of the AAAI, Austin, TX, USA, 25–30 January 2015; Volume 15, pp. 2800–2806.
25. Kaya, M.; Bilge, H.Ş. Deep metric learning: A survey. *Symmetry* **2019**, *11*, 1066. [CrossRef]
26. Suárez, J.L.; García, S.; Herrera, F. A tutorial on distance metric learning: Mathematical foundations, algorithms, experimental analysis, prospects and challenges. *Neurocomputing* **2021**, *425*, 300–322. [CrossRef]
27. Ghojogh, B.; Ghodsi, A.; Karray, F.; Crowley, M. Spectral, Probabilistic, and Deep Metric Learning: Tutorial and Survey. *arXiv* **2022**, arXiv:2201.09267.
28. Cao, X.; Ge, Y.; Li, R.; Zhao, J.; Jiao, L. Hyperspectral imagery classification with deep metric learning. *Neurocomputing* **2019**, *356*, 217–227. [CrossRef]
29. Wang, N.; Zhao, X.; Jiang, Y.; Gao, Y. Iterative Metric Learning for Imbalance Data Classification. In Proceedings of the 2018 International Joint Conference on Artificial Intelligence IJCAI, Stockholm, Sweden, 13–19 July 2018; pp. 2805–2811.
30. Feng, L.; Wang, H.; Jin, B.; Li, H.; Xue, M.; Wang, L. Learning a Distance Metric by Balancing KL-Divergence for Imbalanced Datasets. *IEEE Trans. Syst. Man Cybern. Syst.* **2018**, *49*, 2384–2395. [CrossRef]
31. Gautheron, L.; Habrard, A.; Morvant, E.; Sebban, M. Metric learning from imbalanced data with generalization guarantees. *Pattern Recognit. Lett.* **2020**, *133*, 298–304. [CrossRef]
32. Yan, M.; Li, N. Borderline-margin loss based deep metric learning framework for imbalanced data. *Appl. Intell.* **2022**, *53*, 1487–1504. [CrossRef]
33. Fattahi, M.; Moattar, M.H.; Forghani, Y. Improved cost-sensitive representation of data for solving the imbalanced big data classification problem. *J. Big Data* **2022**, *9*, 1–24. [CrossRef]
34. Wang, K.F.; An, J.; Wei, Z.; Cui, C.; Ma, X.H.; Ma, C.; Bao, H.Q. Deep learning-based imbalanced classification with fuzzy support vector machine. *Front. Bioeng. Biotechnol.* **2022**, *9*, 802712. [CrossRef] [PubMed]
35. UCI Machine Learning Repository. Available online: https://archive.ics.uci.edu/ml/index.php (accessed on 22 July 2024).
36. Navarro, J.R.D.; Noche, J.R. Classification of Mixtures of Student Grade Distributions Based on The Gaussian Mixture Model Using The Expectation-Maximization Algorithm. 2003. Available online: https://www.researchgate.net/publication/2922541_Classification_of_Mixtures_of_Student_Grade_Distributions_Based_on_the_Gaussian_Mixture_Model_Using_the_Expectation-Maximization_Algorithm (accessed on 22 July 2024).

37. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the KDD'96: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, Portland, OR, USA, 2–4 August 1996*; Volume 96, pp. 226–231.
38. Bhattacharyya, A. On a measure of divergence between two statistical populations defined by their probability distributions. *Bull. Calcutta Math. Soc.* **1943**, *35*, 99–109.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.