Applied Soft Computing Journal xxx (xxxx) xxx

Contents lists available at ScienceDirect

Applied Soft Computing Journal

journal homepage: www.elsevier.com/locate/asoc

Standardized Variable Distances: A distance-based machine learning method

Abdullah Elen [a,*], Emre Avuçlu [b]

[a] Department of Computer Technologies, T.O.B.B. Vocational School of Tech. Sci., Karabuk University, Turkey
[b] Department of Computer Technologies, Vocational School of Tech. Sci., Aksaray University, Turkey

ARTICLE INFO

Article history:
Received 19 June 2020
Received in revised form 16 October 2020
Accepted 23 October 2020
Available online xxxx

Keywords:
Machine learning
Multiclass classifier
Distance-based classifier

ABSTRACT

Today, machine learning algorithms are an important research area capable of analyzing and modeling data in any field. Information obtained through machine learning methods helps researchers and planners to understand and review systematic problems of their current strategies. Thus, it is very important to work fully in every field that facilitates human life, such as early and correct diagnosis, correct choice, fully functioning autonomous systems. In this paper, a novel machine learning algorithm for multiclass classification is presented. The proposed method is designed based on the Minimum Distance Classifier (MDC) algorithm. The MDC is variance-insensitive because it classifies input vectors by calculating their distances/similarities with respect to class-centroids (average value of input vectors of a class). As it is known, real-world data contains certain proportions of noise. This situation negatively affects the performance of the MDC. To overcome this problem, we developed a variance-sensitive model, which we call Standardized Variable Distances (SVD), considering the standard deviation and z-score (standardized variable) factors. To ensure the accuracy of the SVD, we used Wisconsin Breast Cancer Original (WBCO) and LED Display Domain (led7digit) datasets, which we obtained from UCI machine learning repository, with 5-fold cross validation. It was compared and analyzed classification performance of the SVD with Decision Tree (DT), Random Forest (RF), k-Nearest Neighbor (k-NN), Multinomial Logistic Regression (MLR), Naïve Bayes (NB), Support Vector Machine (SVM), and the Minimum Distance Classifier (MDC), which are well-known in the literature. It has also been compared thirteen different studies using the same datasets over the past five years. Our results in the experimental studies have shown that the SVD can classify better than traditional and state-of-the-art methods, compared in this study. The proposed method reached over 97% classification accuracy (CACC), F-measure (FM) and area under the curve (AUC) on the WBCO dataset. On the led7digit dataset, approximately 74% CACC, 75.1% FM and 82.2% AUC scores were obtained. It has been observed that the classification scores obtained with the SVD are higher than other ML algorithms used in the experimental studies.

1. Introduction

Machine Learning (ML) is an artificial intelligence technique that makes predictions from existing data using mathematical and statistical methods for unknown situations. In this respect, ML is a natural result of the intersection of Computer Science and Statistics. Thanks to ML, computers learn to accomplish tasks such as recognize patterns, classification, prediction, clustering, etc. In addition to scientific studies, ML finds practical use in many commercial and social applications today [1–3]. The main purpose of a ML task is to be able to generalize from experience. In other words, it is to work correctly on samples that have not been encountered before, with the experience gained from old samples. Sample data are often characterized by measurable properties called features, and an ML algorithm tries to find a correlation between the outputs called "class-labels" and features [4]. One of the most important issues in ML is that input data, which will constitute the experience of the model, are not missing (not fully covering the data-space) or incorrect (wrongly measured or labeled) [5]. In such cases, the trained ML model cannot provide sufficient performance.

ML techniques are divided into four subgroups: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning. In supervised learning, input data and expected output data are provided to the ML to learn before by an expert. Thus, a relationship is established between target data and output data. As a result, a generalized model is created for solving problem. In unsupervised learning, no label data is presented to the ML and asked to configure self-introduction data and find hidden patterns [6]. In semi-supervised learning, some

* Corresponding author.
E-mail addresses: aelen@karabuk.edu.tr (A. Elen), emreavuclu@aksaray.edu.tr (E. Avuçlu).

https://doi.org/10.1016/j.asoc.2020.106855
1568-4946/

Abbreviations

ACC:	Overall accuracy
AROW:	Adaptive regularization of weights
AUC:	Area under the curve
AVG:	Average of supports
CACC:	Classification accuracy
DT:	Decision tree
FM:	F-measure or F1 score
ICA:	Individual classifier accuracy
k-NN:	K-nearest neighbor
MAX:	Maximum of Support
MDC:	Minimum distance classifier
ML:	Machine learning
MLR:	Multinomial logistic regression
MV:	Majority Voting
NB:	Naïve Bayes
OB:	Online bagging
OGD:	Online gradient descent
PAL:	Passive aggressive learning
PPV:	Positive predictive value or precision
PRO:	Product of supports
RF:	Random forest
RIS:	Ranking-based Instance Selection
ROC:	Receiver Operating Characteristics
SCW:	Soft confident weighted
S-HT:	Single Hoeffding tree
SLWNB:	Self-labeled weighted local naïve Bayes
SVD:	Standardized variable distances
SVM:	Support vector machine
TPR:	True positive rate, sensitivity, or recall
WBCO:	Wisconsin breast cancer original

output data are missing in the training dataset. A small portion of the input data is labeled, while a large portion is unlabeled. In reinforcement learning, ML is in a relationship with the dynamic environment. It gets feedback with reward and penalty method while navigating problem space to achieve a specific purpose [7].

In the literature, many studies have been done on the solution of pattern recognition and classification problems. When these studies are examined, it is seen that the suggestions proposed based on ML algorithms are more common. In this study, a new distance-based ML algorithm that can make multiclass classification is introduced. The validity of the proposed method was ensured by the 5-fold cross validation of the WBCO and led7digit datasets obtained from UCI machine learning repository. The results showed that it gives more accurate results than the classifiers used in these experimental studies. Since the proposed method has promising results, it is possible to say that it is a classifier that can be preferred by researchers in scientific studies.

This paper is organized as follows; In the first section, the definition of ML, basic concepts about it and the fields where it is used in practice are mentioned. In the second section, there are studies performed on the traditional and state-of-the-art ML algorithms in the last five years. In the third section, the MDC algorithm, which is the leading version of the proposed method, and technical concepts in its architectural structure are mentioned. In the fourth section, illustration of the proposed method with its mathematical model, pseudo-code, flowchart and are shared. Also, a visual example is provided to illustrate differences between the MDC and SVD. In the fifth section, accuracy of the proposed method with experimental studies is ensured and its performance in detail is compared with other ML algorithms. In the last section, this paper is summarized.

2. Related work

In this section, studies based on ML in the last five years are included. Singh and Singh [8] proposed an ensemble technique (called bagged NB-DT) by using the essential bootstrap aggregating method on hybridization of two base classifiers to be namely Decision Tree (DT) and Naïve Bayes (NB). It trains the basic classifier by repeating random samples from the training set and performs majority-voting of its predictions. In their experimental studies, they determined the bag size to be 100. Ding et al. [9] presented an ordering-based ensemble pruning based on induction of the DT. Their method maps the dataset and base classifiers to a new dataset where the ensemble pruning can be transformed to a feature selection problem. Zhang et al. [10] proposed an empirical analysis to investigate the possibility of strengthening a one-versus one scheme for multiclass imbalance classification problems by applying binary ensemble learning approaches. The purpose of their method is to enhance the performance of binary decomposition utilized for multiclass unbalanced problems via implementing ensemble learning paradigm. Cavalcanti and Soares [11] presented a new algorithm that can make instance selection, called Ranking-based Instance Selection (RIS), which aims to select a subset of the original training set. Their method assigns a score for each instance in the training set, depending on its relationship with all other instances. As the number of close patterns of the same class increases, the instance score also increases.

Karlos et al. [12] introduced a self-labeled weighted variant of the local classifier, called SLWNB, which uses NB as the base-classifier of the self-training scheme. It combines the self-training scheme with local learning using the NB and preprocessing weighting phase. Pham et al. [13] chose several state-of-the-art algorithms as benchmark algorithms. They include several well-known additive models i.e. Passive Aggressive learning (PAL), Soft Confident Weighted (SCW), Online Gradient Descent (OGD), Adaptive Regularization of Weights (AROW), single Hoeffding tree (S-HT) and Online Bagging (OB) with Hoeffding tree (HT). In the proposed method, Gauss random projection was used to create random matrix and size of all the down-spaces was set to $q = 2\,log2\,p$. Aridas et al. [14] introduced a new ensemble method for generating random decision forests. Their method includes a NB classification algorithm to increment the diversity of trees in the forest to increase performance in terms of classification accuracy. Krawczyk and Woźniak [15], proposed two new untrained aggregation operators called as NP-AVG and NP-MAX. It is stated that they can be used in the absence of additional learning materials to train the combination rule. They used the following classifier combination algorithms to compare the performance of their operators. Combiners that do not require training are maximum of support (MAX), majority voting (MV), product of supports (PRO) and, average of supports (AVG).

ML techniques continue to be widely applied in the field of health as in many fields [16]. Studies in the field of health are vital for humanity. Today, thanks to the developments in ML and other artificial intelligence techniques, the methods that can help diagnose breast cancer have been developed. Some studies that have been done on this subject and used in experiments to compare the performance of the SVD are as follows: Nilashi et al. [17] proposed a new knowledge-based system for classifying breast cancer disease using classification, clustering, and noise removing methods. They used the Expectation Maximization (EM) method to aggregate data in similar groups, and Classification and Regression Trees (CART) to generate the fuzzy rules to be used

for the classification of breast cancer disease. They also included Principal Component Analysis (PCA) in their method to overcome the multi-collinearity issue.

Chandrasekaran et al. [18] introduced a fully integrated common-source amplifier based on analog artificial neural network (ANN). The performance of their method with a custom non-linear activation function was tested on the WBCO dataset. Their method achieved 97% classification accuracy on this dataset. Asri et al. [19] did experimental work on the WBCO dataset for a performance comparison between the four machine learning algorithms (SVM, C4.5, NB and k-NN). According to their experimental results obtained that SVM gives the highest accuracy (97,13%) with lowest error rate. Ibeni et al. [20] presented the full Bayesian approach to assess the predictive distribution of all classes using Naïve Bayes (NB), Bayesian networks (BN), and tree-augmented naïve Bayes (TAN) classifiers. They used the WBCO dataset to compare the performance of the algorithms. According to their results, they declared that the best performance is the BN method with an accuracy of 97,2%. Das and Biswas [21] proposed an ensemble learning method for the prediction of breast cancer. Their ensemble method comprises a total of five ML methods which include RF, NB, SVM with two different kernels (RBF, Polynomial), k-NN and DT. They experimented on the WBCO dataset from UCI machine learning repository. According to the classification result, they achieved an average of 95,2% FM score.

3. MDC and statistical concepts

In this section, basic methods used in the development of the SVD are mentioned. The proposed method is modeled with inspiration from the MDC algorithm. It is an ML algorithm that can make distance-based and multiple classification. In this section, it is presented basic bases that should be known about the architecture of the SVD.

3.1. Mean Distance Classifier (MDC)

MDC is an ML algorithm that can classify each input vector in the dataset by calculating its distance from class centroids. Class centroids represent the average values of input vectors of a class and are the only parameter learned in the training phase. In the testing phase of the MDC, the distance from a test sample to the class centroids is calculated and determined as the element of the class with closest value [22]. The Euclid, Manhattan, Minkowski, Chebyshev, Hellinger, Angular and similar methods are used for distance measurement. The concept of distance here is expressed as an index of similarity. Thus, samples at a minimum distance to a class centroid will have maximum similarity. The advantages of this classifier are to become very effective in applications, to become easy for calculation and to have little feature variability. The disadvantage is that it is insensitive on varying degrees of data. This increases the risk of misclassification, especially in noisy data. The general classification principle of the MDC can be expressed as in Eq. (1). The MDC has higher error-rate when compared to other classification methods.

$$x \in \omega_i \;\; if \;\; d\left(x, z_i\right) = \arg\min_j d\left(x, z_j\right), \;\; i, j \in \{1, 2, \ldots, n\} \tag{1}$$

where x the input vector to be classified, z_i is the i-th class-centroid, n is the number of classes, d is the distance function.

3.2. Standardized variables

The standardized variable is called as the z-score. Other terms include z-values, standard scores, and normal scores. These terms can be used interchangeably as in this manuscript. The z-score measures exactly how many standard deviations are above or below the mean of a data point. The above average data point has positive standard scores, while others have negative standard scores. In Fig. 1, standard deviation and z-score values are shown as representative in normal distribution.

It is calculated by subtracting the population mean (μ_p) from a raw score (i.e. an observed value or data point) and then dividing the difference by the population standard deviation (σ_p). The process of converting a raw score into a standard score is called standardization or normalization. If μ_p and σ_p are known, a raw score x is converted into a standard score by Eq. (2).

$$z = \frac{x - \mu_p}{\sigma_p} \tag{2}$$

The absolute value of z symbolizes the distance between raw score x and μ_p in units of the standard deviation. z is negative when the raw score is below the mean, positive when above. Calculating z-value by using this formula requires μ_p and σ_p. However, knowing the true mean and standard deviation of a population is generally unrealistic, except in situations such as the standardized test in which the entire population is measured [23].

3.3. Distance measurements

Distance measurements are an important part of some ML algorithms. These measurements are often used in both supervised and unsupervised learning to calculate the similarity between data points. There are many methods for calculating the distances of a vector from the classified samples. The SVD uses distance measurements to classify input vectors, as in the k-NN and MDC algorithms. The distance measurements we used in the SVD's performance tests are as follows:

Euclidean distance: It is the most preferred distance measure in practice. Also known as "L_2 norm" or "Ruler Distance", which is an extension to the Pythagorean theorem [24]. Euclidean distance refers to square root of the sum of differences of two vectors. The Euclidean distance equation is as follows:

$$d_{Euclidean} = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{3}$$

where, n, i represents the number of samples and the index of a sample to be measured, respectively.

Minkowski distance: It is also known as "L_p norm", is a generalized metric. This family of distances includes three distance metrics (Euclidean, Manhattan and Chebyshev) that are special cases of Minkowski distance, corresponding to different values of p for power distance [24]. The Minkowski distance equation is as follows:

$$d_{Minkowski} = \sqrt[p]{\sum_{i=1}^{n} (|x_i - y_i|)^p} \tag{4}$$

where p refer to a positive number. When $p = 2$, the distance becomes the Euclidean distance, when it becomes Manhattan distance. Chebyshev distance is a variant of Minkowski distance where $p = \infty$. x_i is the i-th value in the vector x and y_i is the i-th value in the vector y.

Manhattan distance: It is also known as " L_1 norm", "Taxicab norm" or "City-block distance", which introduced by Hermann Minkowski in 19th-century [24]. This method refers to the sum of the absolute differences of two vectors. As an example of operating principle of this method, if a grid-like path is followed,

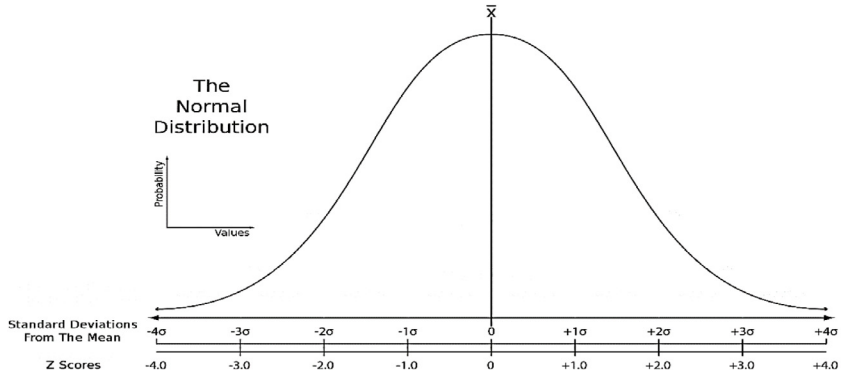

Fig. 1. Standard deviation and z-score values in normal distribution.

it calculates the distance to travel from one data point to another. The Manhattan distance equation is as follows:

$$d_{Manhattan} = \sum_{i=1}^{n} |x_i - y_i| \tag{5}$$

Chebyshev distance: It is also known as "Maximum Value", "Lagrange", and "Chessboard" distance. This method is a metric that measures maximum distance between two observed instances in the vector space. The Chebyshev distance equation is as follows:

$$d_{Chebyshev} = \max_i |x_i - y_i| \tag{6}$$

Hellinger distance: It was introduced in 1909 by Hellinger [25], it is a metric used to measure the similarity between two probability distributions [24]. This method is a metric satisfying triangle inequality. The reason for including $\sqrt{2}$ in the definition of this method is to ensure that the distance value is always between zero and one. The Hellinger distance equation is as follows:

$$d_{Hellinger} = \frac{1}{\sqrt{2}} \sum_{i=1}^{n} \left(\sqrt{x_i} - \sqrt{y_i}\right)^2 \tag{7}$$

Angular distance: It is also called cosine distance, is derived from the cosine similarity, which measures the angle between two vectors, where the cosine distance is obtained by subtracting the cosine similarity from one [24]. The Angular distance equation is as follows:

$$d_{Angular} = 1 - \left(\sum_{i=1}^{n} (x_i y_i) \Big/ \sqrt{\sum_{i=1}^{n} x_i^2} \sqrt{\sum_{i=1}^{n} y_i^2} \right) \tag{8}$$

4. The proposed method

In this section, the multiclass ML model is explained in detail with mathematical notations and pseudocodes. In addition, flowchart and pseudocodes showing the training and testing stages of the SVD are given. A visual example is presented to emphasize the classification difference between the proposed method and the MDC algorithm. As it is known, each input vector belongs to one of different classes in multiclass classifiers. The goal is to create a function that accurately predicts the class to which the vector belongs when a new input vector is given to the classifier. Eq. (9) shows the input matrix ($v_X \in R^{m \times n}$) and the output matrix ($v_Y \in R^m$) corresponding to each input vector.

$$v_X = \begin{bmatrix} x_{0,0} & \cdots & x_{0,n} \\ \vdots & \ddots & \vdots \\ x_{m,0} & \cdots & x_{m,n} \end{bmatrix}, \quad v_Y = \begin{bmatrix} y_0 \\ \vdots \\ y_m \end{bmatrix} \tag{9}$$

Where, m represents the number of samples in the data set and n represents the number of attributes. In the first stage of the proposed method, class labels are determined (uniquely) from the output vector in the dataset desired to be trained, as shown in Eq. (10).

$$\begin{aligned} &v_Y = [y_0, y_1, \ldots, y_m] \\ &c = \{\exists a \in v_Y : (P(a) \wedge \forall b \in v_Y : P(b) \rightarrow a = b)\} \end{aligned} \tag{10}$$

Where, class labels take a value in the range $c = [0, 1, \ldots, k]$, $k \in N$. The resulting class labels are indexed from 0 to k for ease of operation. This is since the datasets have various types of output vectors (integer, real number, non-numeric values, etc.). Class labels are converted to a standard data type. The pseudocode that shows the acquisition of class labels is as in Algorithm 1. The parameter of the *GetClasses*() function is the output vectors of the training set.

Algorithm 1. Get distinct elements of the outputs.

```
function GetClasses(Outputs[m])
    set result[0] ← Outputs[0];
    set counter ← 1;
    for i ← 1 to m do
        if (!result.Contains(Outputs[i])) then
            result[counter] ← Outputs[i];
            counter ← counter + 1;
        end if
    end for
    return result.Sort();
end function
```

In the next step, the mean values of the input vectors are calculated for each class label. Centroid-class vectors or centroid-class matrix ($\mu V \in R^{k \times n}$) shown in Eq. (11) are obtained.

$$\begin{aligned} \mu V &= \sum_{a=0}^{k} \sum_{j=0}^{n} \mu V[a, j] \\ &= \left\{ \sum_{i=0}^{m} \{x_{i,j}, \quad c_a = y_i \Big/ \sum_{i=0}^{m} \{1, \quad c_a = y_i \right. \end{aligned} \tag{11}$$

Where $x_{i,j}$ is the j-th attribute of the i-th sample in the input vector, y_i is the output value of the i-th sample, c_a class label, $\mu V[a, j]$ is the j-th attribute of the a-th vector of the centroid-class matrix. The pseudo-code that shows the acquisition of centroid-class vectors is as in Algorithm 2. The parameters of the *GetCentroidVectors*() function are class labels, input and output vectors of the training set, respectively.

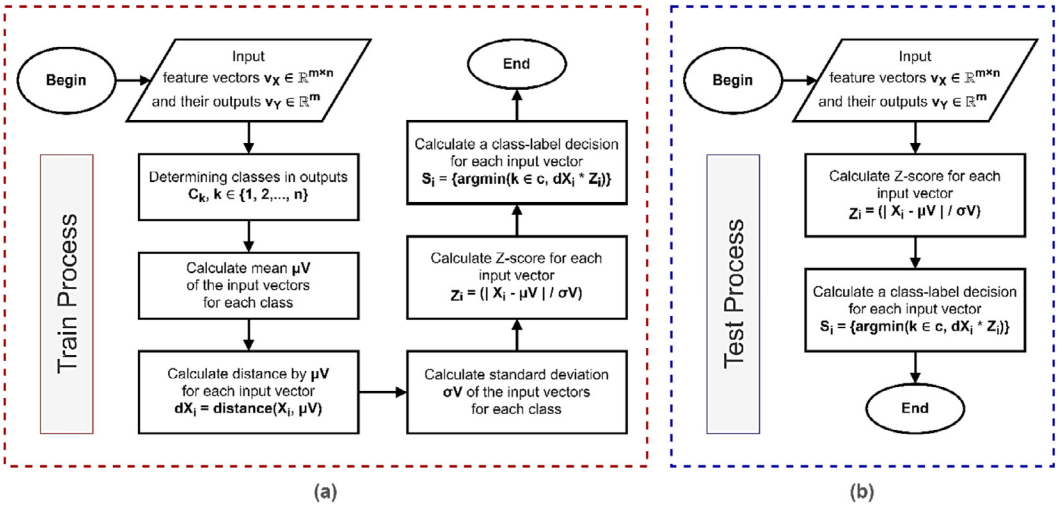

Fig. 2. Flowchart of the SVD with train and test process; (a) train process, (b) test process.

Algorithm 2. Get centroid vectors.

```
1:  function GetCentroidVectors(Class[k], Inputs[m, n], Outputs[m])
2:      set result[k, n];
3:      for a ← 0 to k do
4:          set counter ← 0;
5:          for j ← 0 to n do
6:              for i ← 0 to m do
7:                  if (Class[a] = Outputs[i]) then
8:                      result[Class[a], j] ← result[Class[a], j] + Inputs[i, j];
9:                      counter ← counter + 1;
10:                 end if
11:             end for
12:             result[Class[a], j] ← result[Class[a], j] / counter;
13:         end for
14:     end for
15:     return result;
16: end function
```

Then, the distances ($dX \in R^k$) of each input vector to the centroid class vectors are calculated as shown in Eq. (12).

$$dX = \sum_{i=0}^{k} \{dX[i] = distance(X, \mu V[i, *]),\ X | X \in v_X \tag{12}$$

Where k represents the class label and an input vector in the X dataset. One of the distance methods such as Euclidean, Manhattan, Minkowski, Chebyshev, Angular, Hellinger is used for the $distance(X, \mu V)$ function, which represents the distance measurement. In the next step, the standard deviation matrix ($\sigma V \in R^{k \times n}$) of the input vectors for each class is obtained as shown in Eq. (13).

$$\sigma V = \sum_{a=0}^{k} \sum_{j=0}^{n} \sigma V[a, j] = \left\{ \sqrt{\sum_{i=0}^{m} \{(x_{i,j} - \mu V[a, j])^2},\quad c_a = y_i \middle/ \sum_{i=0}^{m} \{1,\quad c_a = y_i \right. \tag{13}$$

The pseudo-code that shows the acquisition of standard deviation vectors is as in Algorithm 3. The parameters of the *GetStdDevVectors*() function are class labels, input and output vectors of the training set and centroid class vectors obtained from Algorithm 2.

Algorithm 3. Get standard deviation vectors.

```
1:  function GetStdDevVectors(Class[k], Inputs[m, n], Outputs[m], Mean[k, n])
2:      set result[k, n];
3:      for a ← 0 to k do
4:          set counter ← 0;
5:          set c ← Class[a];
6:          for j ← 0 to n do
7:              for i ← 0 to m do
8:                  if (c = Outputs[j]) then
9:                      result[c, j] ← result[c, j] + Pow((Inputs[i, j] - Mean[c, j]), 2);
10:                     counter ← counter + 1;
11:                 end if
12:             end for
13:             result[c, j] ← Sqrt(result[c, j] / counter);
14:         end for
15:     end for
16:     return result;
17: end function
```

In the next step, the absolute values ($Z \in R^k$) of the standard normal distribution or, more commonly known, z-score for an input vector in the dataset are calculated as in Eq. (14).

$$Z = \sum_{i=0}^{k} Z[i] = \left\{ \sum_{j=0}^{n} \frac{|X_j - \mu V[i, j]|}{\sigma V[i, j]},\quad X | X \in v_X \right. \tag{14}$$

Where, X_j represents the j-th attribute of an input vector in the dataset, n represents the number of attributes in the input vector and k represents the number of classes. The pseudo-code that shows the absolute z-score value for each class label of an input vector is as in Algorithm 4. The parameters of the *GetAbsoluteZscores*() method are class labels, centroid-class vectors, standard deviation vectors, and an input vector selected from the dataset, respectively.

In the last stage, the similarity scores of each input vector for classes defined in Eq. (3) are calculated as in Eq. (15).

$$Decision = \arg\min_{i} (i | i \in c,\quad dX[i] Z[i]) \tag{15}$$

Where $i \in c$ represents the class label. The absolute value of the z-score indicates how many standard deviations are from the mean. If a z-score equals zero, the centroid-class vector has the same value as the sample presented. The training process of the method proposed in Algorithm 5 is shown. Where, the parameters of the *Train*() method are the input vectors of the training set and their output values, respectively. As in the MDC, distance measurement is represented as a representation of the *Train*() method in the 12th line of the source-code. The *DistanceMetric*() method

Algorithm 4. Get absolute Z-scores.

```
function GetAbsoluteZscores(Class[k], Mean[k, n], StdDev[k, n], InputVector[n])
    set result[k];
    for a ← 0 to k do
        set c ← Class[a];
        for j ← 0 to n do
            result[c] ← result[c] + Abs(InputVector[j] - mean[c, j]) / stdDev[c, j];
        end for
    end for
    return result;
end function
```

has been used to express methods such as Euclid, Manhattan, Minkowski, Chebyshev, Hellinger.

Algorithm 5. Train process of the proposed method.

```
function Train(Inputs[m, n], Outputs[m])
    set result[m];
    set class[k] ← GetClasses(Outputs);
    set mean[k, n] ← GetCentroidVectors(class, Inputs, Outputs);
    set stdDev[k, n] ← GetStdDevVectors(class, Inputs, Outputs, mean);
    for i ← 0 to m do
        set score[k] ← NULL;
        for a ← 0 to k do
            set inputVector ← Inputs.SelectRow(m);
            set meanVector ← mean.SelectRow(a);
            set zscore ← GetAbsoluteZscore(class, mean, stdDev, inputVector);
            set distance ← DistanceMetric(inputVector, meanVector);
            set score[a] ← distance * zscore;
        end for
        result[i] ← score.GetIndexByVal(score.Minimum());
    end for
    return result;
end function
```

In Algorithm 6, the test process of the proposed classification method is shown. The parameters of the *Test*() method are the input vectors of the training set and their output values, class labels, centroid-class vectors obtained from Algorithm 2, and standard deviation vectors obtained from Algorithm 3.

Algorithm 6. Test process of the proposed method.

```
function Test(Inputs[m, n], Outputs[m], Class[k], Mean[k, n], StdDev[k, n])
    set result[m];
    for i ← 0 to m do
        set score[k] ← NULL;
        for a ← 0 to k do
            set inputVector ← Inputs.SelectRow(m);
            set meanVector ← Mean.SelectRow(a);
            set zscore ← GetAbsoluteZscores(Class, Mean, StdDev, inputVector);
            set distance ← DistanceMetric(inputVector, meanVector);
            set score[a] ← distance * zscore;
        end for
        result[i] ← score.GetIndexByVal(score.Minimum());
    end for
    return result;
end function
```

In the code line of the algorithm 6 and 7 expressed as "*score*.*GetIndexByVal*(*score*.*Minimum*())", the index number represented by the minimum value in the array variable named as "*score*" is found. This index number corresponds to a class label. Thus, the class to which each input vector can belong is estimated. The flowchart of the proposed method is as shown in Fig. 2. In this flowchart, the algorithm clarifies how both the training process and the test process are applied step by step.

Table 1
The simple dataset for compare the difference between the SVD and MDC visually.

	Class A		Class B		Class C	
	X-axis	Y-axis	X-axis	Y-axis	X-axis	Y-axis
Input vectors (Features of the classes A, B and C)	41	45	55	17	65	50
	39	55	49	15	73	37
	37	42	54	21	80	40
	33	47	40	18	63	39
	40	50	50	23	65	45
	44	49	42	29	76	42
	36	44	48	28	82	35
	34	39	34	19	73	39
	30	41	57	24	67	35
	25	47	37	22	72	46
	38	44	35	15	69	45
	28	40	44	25	64	37
	35	50	45	14	68	36
	47	54	46	30	78	35
	28	49	62	19	67	38
	41	45	52	26	64	42
	47	48	41	25	70	52
	41	53	35	28	77	48
	24	55	49	19	77	36
	41	47	41	20	60	48
Mean	**36,45**	**47,20**	**45,80**	**21,85**	**70,50**	**41,25**
Std. Dev.	**6786**	**4808**	**7845**	**4913**	**6295**	**5457**

4.1. Visualization and comparison: the SVD vs. the MDC

In order to compare the difference between the SVD and the MDC visually, it is offered a simple classification example as shown in Fig. 3. The simple dataset shown in Table 1 consists of three different classes (Class A, Class B, and Class C) based on two attributes (Feature 1: *X*-axis and Feature 2: Y-axis). Each class contains 20 samples, represents coordinate points on the 2D plane. The last two rows of this table show the mean and standard deviation values of the input vectors, respectively.

Accordingly, MDC and SVD classifiers were used to determine which class a point in the coordinate (42, 35) belongs to (See Table C in Fig. 3). While the MDC algorithm is based on the mean (See Table A in Fig. 3) of the input vectors in the classification process, SVD takes into account both the mean (See Table A in Fig. 3) and standard deviation (See Table B in Fig. 3) values.

In Fig. 3, Table A shows the centroid-vectors of each input label of the input vectors in the dataset, and Table B shows the standard deviation vectors of each input label of the input vectors. In Table C, the input vector to be tested by classification algorithms is given. In Table D, it shows the classification results for the sample input vector of both algorithms. When both classifiers calculate the minimum distance according to their own methods; For the sample input vector, the MDC decides "Class A" with a distance of 13.4 units, and the SVD as "Class B" with 43.27 units.

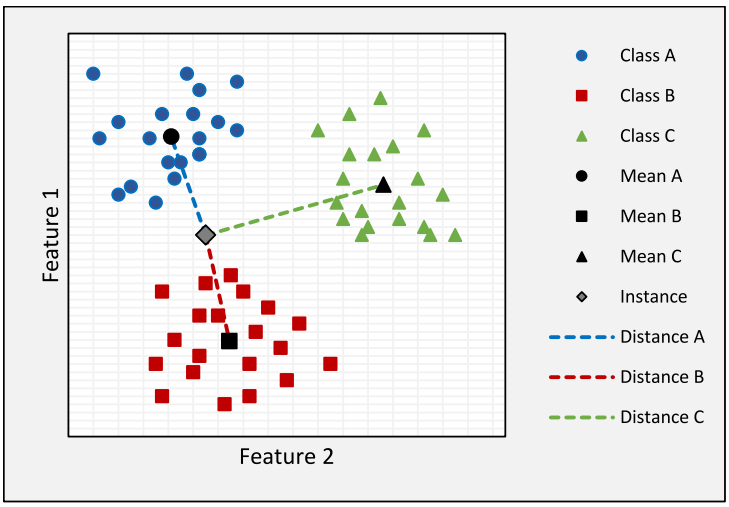

Table A. Centroid (mean) vectors by classes.

	Feature 1	Feature 2
Class A	36,45	47,20
Class B	45,80	21,85
Class C	70,50	41,25

Table B. Standard deviation vectors by classes.

	Feature 1	Feature 2
Class A	6,786	4,808
Class B	7,845	4,913
Class C	6,295	5,457

Table C. Instance of input vector for test.

Instance	Feature 1	Feature 2
Input Vector	42	35

Table D. Classification results.

Decision	MDC	SVD
Class A	13,40	44,97
Class B	13,69	43,27
Class C	29,18	165,51

Fig. 3. Visualization of classification process of the SVD on a sample dataset.

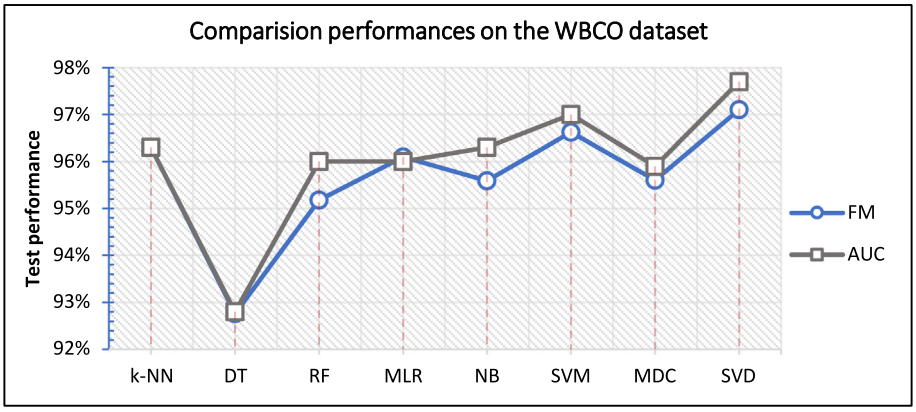

Fig. 4. Comparison test performances of the classifiers on the WBCO dataset.

Table 2
Attributes and settings of the machine learning methods.

Method	Attributes and settings
DT	Learning method is determined as "C4.5".
RF	The number of decision trees is determined as 100.
MLR	The lower bound principle (Newton–Raphson) was used for multinomial logistic regression fitting.
NB	The distribution parameter is determined as "Gaussian".
SVM	The kernel has been designated as "Gaussian".
k-NN MDC SVD	Euclidean, Manhattan, Minkowski, Chebyshev, Hellinger and Angular were used as distance measurement methods for k-NN, MDC, and SVD. The k-value of the k-NN algorithm is fixed at 3.

Table 3
Attributes of the WBCO dataset.

#	Attribute	Domain
1	Sample code number	Id number
2	Clump thickness	1–10
3	Uniformity of cell size	1–10
4	Uniformity of cell shape	1–10
5	Marginal adhesion	1–10
6	Single epithelial cell size	1–10
7	Bare nuclei	1–10
8	Bland chromatin	1–10
9	Normal nucleoli	1–10
10	Mitoses	1–10
11	Classes (benign or malignant)	2: benign, 4: malignant

5. Experimental results

In experimental studies, two different datasets were used, one for binary and one for multiple classification to ensure the accuracy of the proposed method. It was used 5-fold cross validation, which is a common model to conduct the experiment. In this model, a dataset is randomly divided into five non-reciprocal subsets with approximately the equivalent number of instances. Thus, five studies are performed for each dataset. In the training and testing of classification algorithms, 80% and 20% parts of the dataset were used, respectively. Then, by calculating their average accuracy values, the overall success rate of the algorithms was obtained. The attributes and settings of the ML algorithms used to compare performance in experimental studies are given in Table 2.

5.1. Wisconsin Breast Cancer Original (WBCO) dataset

The performance of our proposed method was first validated with the Wisconsin Breast Cancer Original (WBCO) dataset from UCI machine learning repository [26]. The WBCO dataset was chosen in this study because it is a popular in machine learning field, and classification accuracy metrics in the dataset is widely reported. The WBCO dataset contains 699 samples obtained from a breast tissue. Each record in the dataset has nine properties,

Table 4
Comparisons of performances of the methods against WBCO dataset (best values highlighted in bold).

	Method	Fold-1			Fold-2			Fold-3			Fold-4			Fold-5			Average		
		TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM
Euclidean	*k-NN*	0,968	0,953	0,96	0,974	0,979	0,976	0,984	0,984	0,984	0,958	0,963	0,96	0,926	0,937	0,932	0,962	0,963	0,963
	MDC	0,968	0,953	0,96	0,978	0,974	0,976	0,978	0,974	0,976	0,946	0,942	0,944	0,916	0,912	0,914	0,957	0,951	0,954
	SVD	0,98	0,989	0,984	0,962	0,978	0,97	0,99	0,995	0,992	0,953	0,973	0,963	0,931	0,953	0,941	**0,963**	**0,977**	**0,97**
Manhattan	*k-NN*	0,968	0,953	0,96	0,984	0,984	0,984	0,984	0,984	0,984	0,949	0,957	0,953	0,926	0,937	0,932	0,962	0,963	0,963
	MDC	0,942	0,901	0,921	0,962	0,942	0,952	0,973	0,963	0,968	0,935	0,921	0,928	0,916	0,912	0,914	0,946	0,928	0,937
	SVD	0,98	0,989	0,984	0,971	0,984	0,977	0,99	0,995	0,992	0,962	0,978	0,97	0,924	0,942	0,933	**0,965**	**0,977**	**0,971**
Minkowski	*k-NN*	0,973	0,963	0,968	0,971	0,984	0,977	0,984	0,984	0,984	0,94	0,952	0,946	0,926	0,937	0,932	0,959	0,964	0,961
	MDC	0,964	0,973	0,969	0,949	0,957	0,953	0,984	0,984	0,984	0,943	0,947	0,945	0,928	0,933	0,931	0,954	0,959	0,956
	SVD	0,98	0,989	0,984	0,962	0,978	0,97	0,99	0,995	0,992	0,953	0,973	0,963	0,923	0,947	0,935	**0,961**	**0,976**	**0,969**
Chebyshev	*k-NN*	0,962	0,942	0,952	0,964	0,973	0,969	0,984	0,984	0,984	0,94	0,952	0,946	0,919	0,927	0,923	0,954	0,956	0,955
	MDC	0,964	0,973	0,969	0,949	0,957	0,953	0,978	0,974	0,976	0,94	0,952	0,946	0,928	0,933	0,931	0,952	0,958	0,955
	SVD	0,98	0,989	0,984	0,962	0,978	0,97	0,99	0,995	0,992	0,953	0,973	0,963	0,923	0,947	0,935	**0,961**	**0,976**	**0,969**
Hellinger	*k-NN*	0,978	0,974	0,976	0,978	0,974	0,976	0,995	0,99	0,992	0,94	0,952	0,946	0,919	0,927	0,923	0,962	0,963	0,963
	MDC	0,952	0,922	0,937	0,962	0,942	0,952	0,978	0,974	0,976	0,952	0,952	0,952	0,913	0,917	0,915	0,952	0,941	0,946
	SVD	0,98	0,989	0,984	0,962	0,978	0,97	0,99	0,995	0,992	0,953	0,973	0,963	0,931	0,953	0,941	**0,963**	**0,977**	**0,97**
Average	*k-NN*	0,97	0,957	0,963	**0,974**	**0,979**	**0,976**	0,986	0,985	0,986	0,945	0,955	0,95	0,923	0,933	0,928	0,96	0,962	0,961
	MDC	0,958	0,944	0,951	0,96	0,955	0,957	0,979	0,974	0,976	0,943	0,943	0,943	0,921	0,921	0,921	0,952	0,947	0,95
	SVD	**0,98**	**0,989**	**0,984**	0,963	**0,979**	0,971	**0,99**	**0,995**	**0,992**	**0,955**	**0,974**	**0,964**	**0,926**	**0,948**	**0,937**	**0,963**	**0,977**	**0,97**

Table 5
Comparisons of performances of the methods against WBCO dataset (best values highlighted in bold).

Method	Fold-1			Fold-2			Fold-3			Fold-4			Fold-5			Average		
	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM
DT	0,947	0,911	0,929	0,927	0,931	0,929	0,952	0,952	0,952	0,924	0,936	0,93	0,886	0,911	0,898	0,927	0,928	0,928
RF	0,968	0,953	0,96	0,964	0,973	0,969	0,968	0,968	0,968	0,933	0,941	0,937	0,917	0,932	0,925	0,95	0,954	0,952
MLR	0,973	0,963	0,968	**0,973**	0,963	0,968	0,984	0,984	0,984	**0,968**	0,968	0,968	0,911	0,922	0,916	0,962	0,96	0,961
NB	0,955	0,968	0,961	0,946	0,962	0,954	0,984	0,984	0,984	0,938	0,957	0,947	**0,924**	**0,942**	**0,933**	0,949	0,963	0,956
SVM	**0,984**	0,984	**0,984**	0,971	**0,984**	**0,977**	0,984	0,984	0,984	0,955	0,968	0,961	0,917	0,932	0,925	0,962	0,97	0,966
SVD	0,98	0,989	**0,984**	0,971	**0,984**	**0,977**	**0,99**	**0,995**	**0,992**	0,962	**0,978**	**0,97**	**0,924**	**0,942**	**0,933**	**0,965**	**0,977**	**0,971**

Table 6
Comparisons of AUC scores of the classifiers on the WBCO dataset (best values highlighted in bold).

Method	Distance	Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
k-NN	*Euclidean*	0,953	0,979	0,984	0,963	0,937	0,963
DT	–	0,911	0,931	0,952	0,936	0,911	0,928
RF	–	0,947	0,973	0,989	0,962	0,927	0,96
MLR	–	0,963	0,963	0,984	0,968	0,922	0,96
NB	–	0,968	0,962	0,984	0,957	**0,942**	0,963
SVM	–	0,984	**0,984**	0,984	0,968	0,932	0,97
MDC	*Minkowski*	0,973	0,957	0,984	0,947	0,933	0,959
SVD	*Manhattan*	**0,989**	**0,984**	**0,995**	**0,978**	**0,942**	**0,977**

all values are represented as integer numbers between 1 and 10 and have been found to fluctuate especially between benign (458 patients) and malignant (241 patients) samples. The attributes of the WBCO dataset are shown in Table 3.

In the first stage of our experimental study, it was compared the classification performance of distance-based ML algorithms (k-NN, MDC and SVD) according to different distance measurement methods on the WBCO dataset. The results of this experiment are detailed in Table 4. As can be seen from this table, the average FM score of the SVD according to all distance methods is 97%. When evaluated separately according to cross validation scores, it obtained the highest FM score in other datasets except Fold-2.

Table 5 shows the classification performance of the SVD (with Manhattan distance) against five other machine learning algorithms (DT, RF, MLR, NB, and SVM). The method has achieved better performance than other algorithms.

The AUC scores of the classifiers on the WBCO dataset are compared in Table 6. According to Table 4, distance-based ML algorithms; Given the best FM scores as the k-NN (Euclidean distance), MDC (Minkowski distance) and SVD (Manhattan distance).

In Fig. 4, the average FM and AUC scores of the classification algorithms used in experimental studies were compared according to the 5-fold cross validation method on the WBCO dataset. The distance methods of distance-based classifiers are determined according to Table 6.

According to the 5-fold cross validation method of classification algorithms on the WBCO dataset, AUC scores for each subdataset are shown in Fig. 5. The distance methods of the k-NN, MDC and SVD classifiers are determined according to Table 6.

Table 7 provides a comparison of the classification performance of the SVD's traditional and state-of-the-art ML algorithms on the WBCO dataset. This table contains six different studies, including the SVD. The proposed method has the highest score with the classification accuracy (CACC) of 0.973.

In Fig. 6, ROC curves showing the test results in each dataset according to the 5-fold cross validation method of the SVD are given.

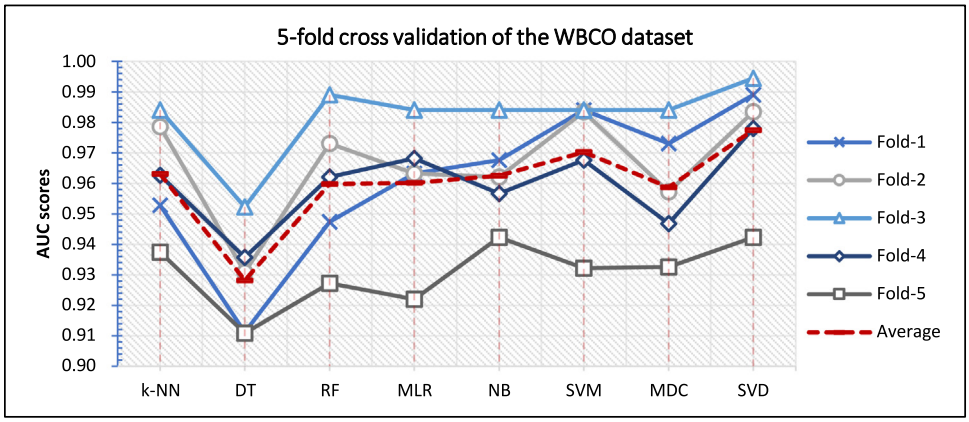

Fig. 5. AUC scores of the classifiers on the WBCO dataset.

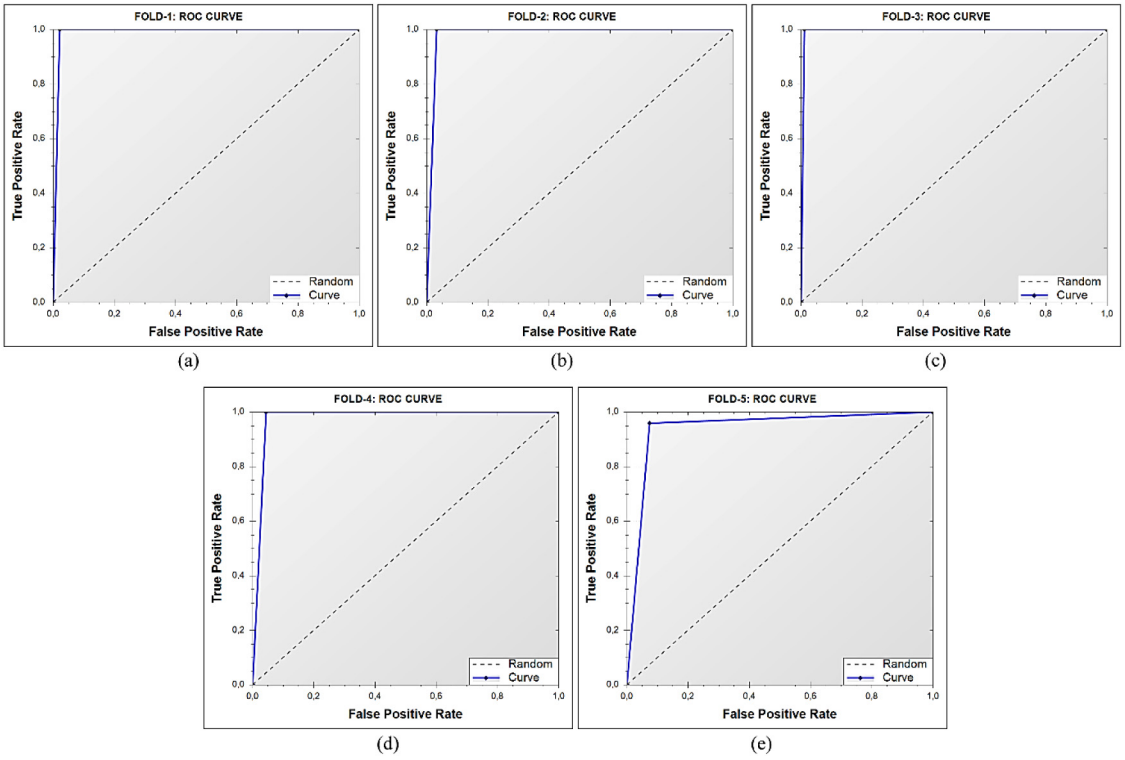

Fig. 6. ROC curves for test performance of the SVD on the WBCO dataset.

Table 7
Comparison of the SVD with traditional and state-of-the-art methods on the WBCO dataset.

Reference	Classification methods	CACC	FM	AUC
Proposed method	SVD with Manhattan distance	0,973	0,971	0,977
Nilashi et al. [17] (2017)	EM-PCA-CART-Fuzzy Rule-based	0,932	–	–
Chandrasekaran et al. [18] (2020)	ANN with custom non-linear activation function.	0,969	–	0,989
Asri et al. [19] (2016)	C4.5, SVM, NB and k-NN	– – – –	C4.5 (0,93-0,96) SVM (0,95- 0,97) NB (0,94-0,96) k-NN (0,93-0,96)	– – – –
Ibeni et al. [20] (2019)	Bayesian Networks (BN), NB, Tree-Augmented Naïve Bayes (TAN)	BAN (0,972) NB (0,959) TAN (0,962)	BAN (0,978) NB (0,968) TAN (0,971)	– – –
Das & Biswas [21] (2019)	Ensemble learning (RF, NB, SVM-RBF, SVM-Polynomial, k-NN)	–	0,952	–

Table 8
Attributes of the led7digit dataset.

#	Attribute	Domain
1	Led1	[0, 1]
2	Led2	[0, 1]
3	Led3	[0, 1]
4	Led4	[0, 1]
5	Led5	[0, 1]
6	Led6	[0, 1]
7	Led7	[0, 1]
8	Classes	{0, 1, 2, 3, 4, 5, 6, 7, 8, 9}

5.2. Led7digit dataset

The performance of our proposed method was secondly validated with the LED Display Domain (led7digit) dataset from UCI machine learning repository [27]. This simple domain contains 7 Boolean attributes, one for each light-emitting diode of a 7-segment display. The task is to determine which digit is shown in the display. In this case, each attribute value has the 10% probability of having its value inverted. This dataset is a sample of 500 instances obtained from the original data generator. The attributes of the led7digit dataset are shown in Table 8.

In the second stage of our experimental study, it was compared the classification performance of the SVD on led7digit dataset with both traditional and state-of-the-art ML algorithms as in the previous stage. Then, we analyzed the SVD in detail with the distance-based classifiers according to 5-fold cross validation method. In the last stage of our second experiment, it was evaluated the performance of the SVD with all the classifiers used in this study. Table 9 provides a comparison of classification performance of the SVD with traditional and state-of-the-art ML algorithms on the led7digit dataset.

At this stage of our experimental studies, it was compared the performances of ML algorithms that classify according to distance measurement in detail. In Table 10, the performances of the k-NN, MDC and SVD classifiers on led7digit dataset were compared according to Euclidean, Manhattan, Minkowski, Chebyshev and Hellinger distance measurement methods. According to the results obtained; The MDC classifier Euclidean has the best FM score in the distances of the Manhattan and Hellinger, while the SVD has the best FM score in the distances of the Minkowski and Chebyshev. However, according to the average of these four distances, the highest FM score belongs to the SVD classifier. In addition, the SVD achieved the highest classification score with the Manhattan distance on the Fold-1 dataset. The highest classification scores of the MDC and k-NN algorithms were obtained with Hellinger distance on Fold-1 dataset.

Apart from this, when using Angular (Cosine distance) as the distance measurement, it is seen that the average performance value of the SVD does not change, but the k-NN and MDC algorithms are close to zero. In other words, it can be said that SVD works consistently with all distance measurement methods we use in experimental studies. Classification performances of the k-NN, MDC and SVD algorithms according to angular distance measurement are shown in Table 11.

Comparison of the AUC scores of the classification algorithms according to different distance measurement methods on the Led7digit dataset is given in Table 12. The proposed method achieved the highest AUC score in all distance measurements relative to the 5-fold cross validation average. As a result, the SVD achieved a very successful classification score compared to all distance measurement methods used in experimental studies.

Fig. 7 shows the ROC curves of the distance-based classification algorithms (SVD, MDC and k-NN) according to the 5-fold cross validation method on the led7digit dataset. The distance methods of distance-based classifiers are based on the highest average AUC scores in Table 12; "Chebyshev distance" for k-NN classifier, "Minkowski distance" for the MDC classifier and "Hellinger distance" for the SVD. When it is evaluated as ROC curves, it is seen that k-NN classifier provides low performance in Fold-3 and the MDC's Fold-4 sub-dataset. On the other hand, the SVD is consistently at the average score level in the entire 5-fold cross validation series.

The average AUC scores of the distance-based classification algorithms (SVD, MDC and k-NN) according to the 5-fold cross validation method on the led7digit dataset are shown in Fig. 8.

Table 13 shows the AUC scores according to the 5-fold cross validation method of eight ML algorithms, including the SVD. The distance methods of distance-based classifiers are determined according to the highest average AUC scores in Table 12. As can be seen from the table, the highest mean AUC score belongs to the SVD.

In experimental studies, the WBCO and Led7digit datasets were used for validation of the proposed method. The results obtained for each dataset are presented in detail. It is seen that the classification performance of the SVD is higher in both datasets compared to other ML algorithms. In addition to considering the distance-based classifiers, it is seen that the SVD has the highest classification score in all distance metrics (Euclidean, Manhattan,

Table 9
Comparison of SVD with state-of-the-art methods (for led7digit dataset).

Reference	Classification methods	CACC	FM
Proposed method	SVD with Manhattan distance	0,738	0,751
Singh & Singh [8] (2019)	Bagged Naive Bayes-Decision Tree (BNBDT)	0,692	–
Ding et al. [9] (2017)	Ordering-based ensemble pruning	0,722	–
Zhang et al. [10] (2016)	One-vs-One scheme for ensemble learning	SMB-CART (0,708) Easy-BPNN (0,72) SMB-SVM (0,721)	– – –
Cavalcanti & Soares [11] (2020)	Ranking-based Instance Selection (RIS)	RIS 1 (0,743) RIS 2 (0,628) RIS 3 (0,743)	– – –
Karlos et al. [12] (2016)	Self-labeled Weighted Local Naive Bayes (SLWNB), R is training set ratio.	R %10 (0,604) R %20 (0,672) R %30 (0,686) R %40 (0,706)	– – – –
Pham et al. [13] (2017)	Random Projection and Hoeffding tree	–	0,685
Aridas et al. [14] (2016)	Ensemble method which called NBFCV (Random Forest + Naïve Bayes)	0,708	–
Krawczyk & Woźniak [15] (2016)	Untrained aggregation operators which called NP-AVG (average of support) and NP-MAX (maximum of support)	NP-AVG (0,732) NP-MAX (0,753)	–

Table 10
Test results on the led7digit dataset by distance metrics (best values highlighted in bold).

	C.V.	k-NN			MDC			SVD		
		TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM
Euclidean	Fold-1	0,775	0,764	0,769	0,812	0,794	0,803	0,8	0,786	0,793
	Fold-2	0,693	0,657	0,675	0,731	0,708	0,719	0,704	0,657	0,679
	Fold-3	0,673	0,634	0,653	0,739	0,721	0,73	0,732	0,708	0,72
	Fold-4	0,724	0,713	0,719	0,752	0,758	0,755	0,777	0,767	0,772
	Fold-5	0,718	0,719	0,718	0,739	0,739	0,739	0,734	0,745	0,739
Manhattan	Fold-1	0,775	0,764	0,769	0,829	0,81	0,819	0,813	0,795	0,804
	Fold-2	0,693	0,657	0,675	0,754	0,735	0,744	0,715	0,697	0,706
	Fold-3	0,673	0,634	0,653	0,749	0,73	0,739	0,732	0,708	0,72
	Fold-4	0,724	0,713	0,719	0,781	0,776	0,778	0,793	0,775	0,784
	Fold-5	0,718	0,719	0,718	0,77	0,754	0,762	0,734	0,745	0,739
Minkowski	Fold-1	0,775	0,764	0,769	0,826	0,805	0,816	0,8	0,786	0,793
	Fold-2	0,693	0,657	0,675	0,677	0,654	0,666	0,697	0,647	0,671
	Fold-3	0,673	0,634	0,653	0,745	0,721	0,733	0,732	0,708	0,72
	Fold-4	0,724	0,713	0,719	0,758	0,757	0,758	0,777	0,767	0,772
	Fold-5	0,718	0,719	0,718	0,708	0,721	0,714	0,734	0,745	0,739
Chebyshev	Fold-1	0,763	0,731	0,747	0,779	0,714	0,745	0,8	0,786	0,793
	Fold-2	0,662	0,614	0,637	0,638	0,537	0,583	0,697	0,647	0,671
	Fold-3	0,646	0,595	0,62	0,574	0,563	0,569	0,732	0,708	0,72
	Fold-4	0,765	0,716	0,739	0,697	0,649	0,673	0,777	0,767	0,772
	Fold-5	0,699	0,699	0,699	0,671	0,635	0,652	0,734	0,745	0,739
Hellinger	Fold-1	0,791	0,782	0,787	0,83	0,819	0,825	0,789	0,775	0,782
	Fold-2	0,693	0,657	0,675	0,722	0,705	0,714	0,675	0,636	0,655
	Fold-3	0,673	0,634	0,653	0,721	0,699	0,71	0,724	0,697	0,71
	Fold-4	0,724	0,713	0,719	0,754	0,749	0,752	0,779	0,777	0,778
	Fold-5	0,745	0,747	0,746	0,719	0,729	0,724	0,734	0,745	0,739
Averages		0,716	0,694	0,705	0,739	0,719	0,729	**0,749**	**0,733**	**0,74**

Table 11
Test results on the led7digit dataset by Angular distance metric.

	C.V.	k-NN			MDC			SVD		
		TPR	PPV	FM	TPR	PPV	FM	TPR	PPV	FM
Angular	Fold-1	0	0	NaN	0	0	NaN	0,8	0,786	0,793
	Fold-2	0,032	0,058	0,041	0,013	0,008	0,01	0,705	0,651	0,677
	Fold-3	0,008	0,086	0,014	0,002	0,014	0,004	0,723	0,699	0,711
	Fold-4	0,027	0,061	0,038	0	0	NaN	0,766	0,757	0,761
	Fold-5	0,008	0,086	0,014	0	0	NaN	0,734	0,745	0,739
AVERAGES		0,015	0,058	0,027	0,003	0,005	0,007	**0,746**	**0,727**	**0,736**

Table 12
Comparison of AUC scores according to distance metrics of the classifiers on the led7digit dataset.

Method		Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
Euclidean	**k-NN**	0,783	0,813	0,528	0,659	0,839	0,725
	SVD	0,879	0,776	0,810	0,748	0,879	**0,818**
	MDC	0,876	0,826	0,815	0,646	0,871	0,807
Manha	**k-NN**	0,783	0,813	0,528	0,659	0,839	0,725
	SVD	0,876	0,783	0,810	0,748	0,879	**0,819**
	MDC	0,863	0,772	0,754	0,698	0,858	0,789
Minkowski	**k-NN**	0,783	0,813	0,528	0,659	0,839	0,725
	SVD	0,879	0,779	0,810	0,748	0,879	**0,819**
	MDC	0,932	0,827	0,862	0,559	0,879	0,812
Chebyshev	**k-NN**	0,786	0,817	0,525	0,751	0,839	0,744
	SVD	0,879	0,779	0,810	0,748	0,879	**0,819**
	MDC	0,846	0,836	0,791	0,628	0,871	0,794
Hellinger	**k-NN**	0,780	0,813	0,528	0,659	0,834	0,723
	SVD	0,879	0,779	0,811	0,762	0,879	**0,822**
	MDC	0,878	0,783	0,761	0,642	0,919	0,797
Average	**k-NN**	0,783	**0,814**	0,527	0,678	0,838	0,728
	SVD	0,878	0,779	**0,810**	**0,751**	0,879	**0,82**
	MDC	**0,879**	0,809	0,797	0,635	**0,880**	0,8

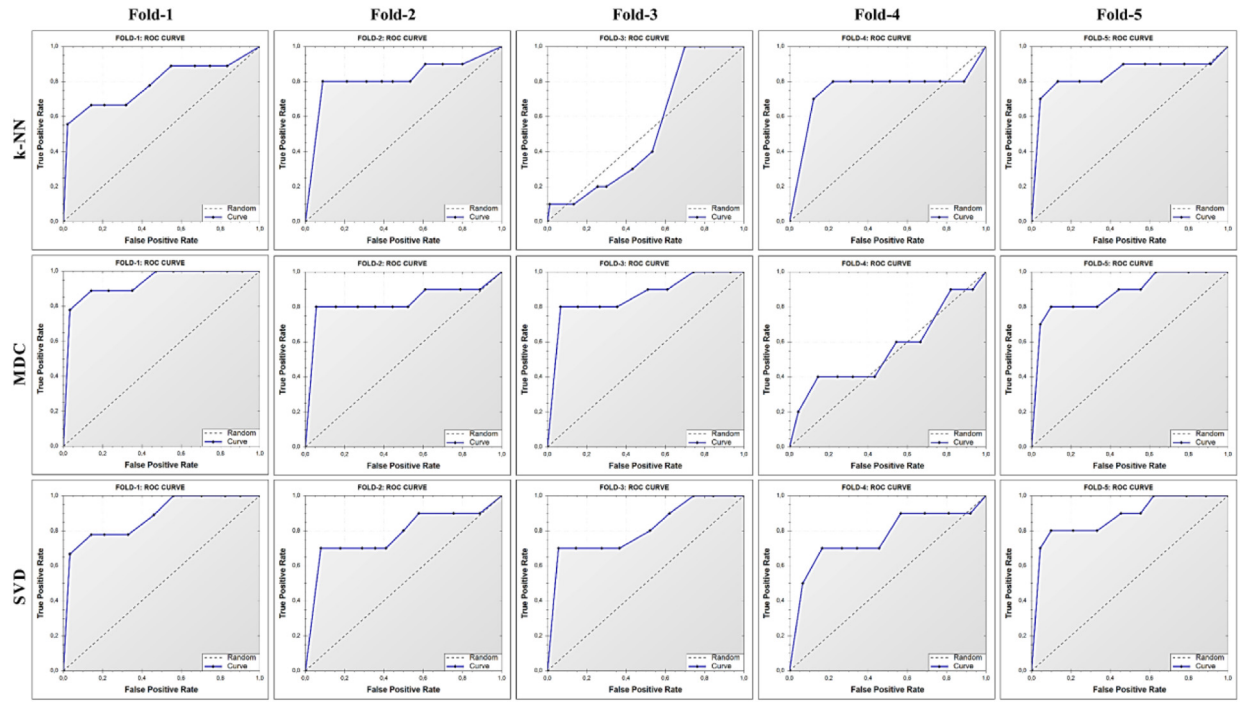

Fig. 7. ROC curves of the distance-base classifier on the led7digit dataset.

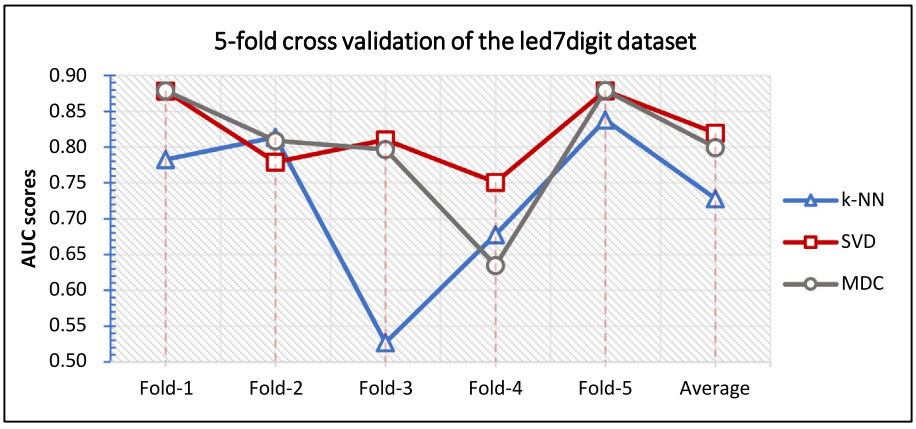

Fig. 8. AUC scores of the SVD, MDC and k-NN classifiers.

Minkowski, Chebyshev and Hellinger) for both datasets. In addition, it was observed that when Angular (Cosine) was preferred as the distance metric, there was no obvious change in the classification performance of the SVD, while the MDC and k-NN algorithms failed.

6. Discussions

The proposed method, unlike the MDC classifier, takes into account the z-score, standard deviation vectors, eliminating the variance insensitivity of the MDC, providing higher classification success. As it is known, input parameters of real-world problems are quite noisy, which negatively affects the classification success, especially in distance-based ML algorithms. Although the training data are normalized, it increases the possibility of the method to produce erroneous results due to the high variance or the feature with standard deviation value. Therefore, the performance comparisons (See Tables 4 and 10) of the distance-based classifiers (k-NN, MDC and SVD) used in experimental studies are given in detail. In addition to the higher average classification success of SVD, it is observed that the classification success of SVD in each sub-data set is generally higher than the 5-fold cross-validation model.

Run-time (CPU) analysis of ML algorithms used in experimental studies is shown in Table 14. Analyzes were performed on WBCO and Led7digit data sets according to a 5-fold cross validation model and their average values were calculated in milliseconds. According to the results, it is seen that the working time cost of the SVD is between the MDC and NB. The computer system on which runtime analysis is performed has Intel i7 3.6 GHz CPU, 8 GB system memory and 64 bit operating system (Windows 10) features.

7. Conclusions

In this study, a distance-based machine learning algorithm capable of multiple classification is proposed. This method is called SVD (*Standardized Variable Distances*). The reason why we prefer this name is mainly because we use the "Standardized Variable" method, commonly known as the z-score, in prediction model of

Table 13
Comparison of AUC scores of the classifiers on the led7digit dataset (best values highlighted in bold).

Method	Distance	Fold-1	Fold-2	Fold-3	Fold-4	Fold-5	Average
k-NN	Chebyshev	0,786	0,817	0,525	0,751	0,839	0,744
DT	–	0,781	0,671	0,797	0,63	0,92	0,760
RF	–	0,783	0,675	0,82	0,634	0,769	0,736
MLR	–	0,524	0,542	0,521	0,538	0,596	0,544
NB	–	0,863	0,802	0,836	0,501	0,902	0,781
SVM	–	0,87	0,774	0,809	0,598	0,912	0,793
SVD	Hellinger	0,879	0,779	0,811	0,762	0,879	**0,822**
MDC	Minkowski	0,932	0,827	0,862	0,559	0,879	0,812

Table 14
Run-time analysis of the ML algorithms on the WBCO and Led7digit datasets (in milliseconds).

Dataset		DT	RF	NB	MLR	SVM	k-NN	MDC	SVD
WBCO	*Train*	285	1385	2	24	40	57	1	3
	Test	1	25	1	< 1	1	16	< 1	1
Led7digit	*Train*	32	204	8	219	176	122	1	6
	Test	4	17	1	< 1	27	35	< 1	2

the SVD, inspired by the MDC algorithm. The MDC is a method that can classify an input vector by calculating its distance from class-centroids. Classification performance of the MDC is very effective in cases where feature patterns has little variability. However, variance-insensitive is a disadvantage especially for noisy feature patterns. Thanks to the proposed method, we have improved the classification performance according to MDC, considering standard deviation factors. It was ensured the accuracy of the SVD with Wisconsin Breast Cancer Original (WBCO) and LED Display Domain (led7digit) datasets obtained from UCI machine learning repository. These datasets are frequently preferred in classification algorithms and there are many scientific studies in terms of performance comparison. In experimental studies, the 5-fold cross validation technique for both datasets were used. Accordingly, seven different ML algorithms well-known in the literature with their classification performances were compared and analyzed in detail. State-of-the-art methods published in the last 5 years using the same datasets were also compared. According to our experimental results, the SVD has outperformed many traditional and state-of-the-art algorithms. The proposed method was found to be higher than other ML algorithms used in the experimental studies with a classification accuracy of over 97% on the WBCO dataset and about 74% on the led7digit dataset. The SVD may be an alternative to other ML algorithms as a new approach method.

CRediT authorship contribution statement

Abdullah Elen: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Emre Avuçlu:** Conceptualization, Investigation, Resources, Writing - review & editing.

Declaration of competing 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.

Acknowledgments

We would like to thank Dr. Eftâl Şehirli and Dr. Cemil Közkurt for their supports.

References

[1] L. Wei, Q. Zou, M. Liao, H. Lu, Y. Zhao, A novel machine learning method for cytokine-receptor interaction prediction, Comb. Chem. High Screen. 19 (2) (2016) 144–152, http://dx.doi.org/10.2174/1386207319666151110122621.

[2] W. Mao, G. Yan, L. Dong, A novel machine learning based method of combined dynamic environment prediction, Math. Probl. Eng. 2013 (2013) 1–15, http://dx.doi.org/10.1155/2013/141849.

[3] Ö.F. Ertuğrul, M.E. Tağluk, A novel machine learning method based on generalized behavioral learning theory, Neural Comput. Appl. 28 (12) (2016) 3921–3939, http://dx.doi.org/10.1007/s00521-016-2314-8.

[4] M. Mohammed, M.B. Khan, E.B.M. Bashier, Machine Learning: Algorithms and Applications, CRC Press, Boca Raton, FL, USA, ISBN: 978-1-4987-0538-7, 2016.

[5] E.A. Abdel Maksoud, S. Barakat, M. Elmogy, Medical images analysis based on multilabel classification, Mach. Learn. Bio-Signal Anal. Diagn. Imaging 20 (2019) 9–245, http://dx.doi.org/10.1016/b978-0-12-816086-2.00009-6.

[6] K. Kourou, T.P. Exarchos, K.P. Exarchos, M.V. Karamouzis, D.I. Fotiadis, Machine learning applications in cancer prognosis and prediction, Comput. Struct. Biotechnol. J. 13 (2015) 8–17, http://dx.doi.org/10.1016/j.csbj.2014.11.005.

[7] F. Doshi-Velez, J. Pineau, N. Roy, Reinforcement learning with limited reinforcement: Using Bayes risk for active learning in POMDPs, Artificial Intelligence 187–188 (2012) 115–132, http://dx.doi.org/10.1016/j.artint.2012.04.006.

[8] N. Singh, P. Singh, A novel bagged Naïve Bayes-decision tree approach for multi-class classification problems, J. Intell. Fuzzy Systems 36 (3) (2019) 2261–2271, http://dx.doi.org/10.3233/jifs-169937.

[9] S. Ding, Z. Chen, S. Zhao, T. Lin, Pruning the ensemble of ANN based on decision tree induction, Neural Process. Lett. 48 (1) (2017) 53–70, http://dx.doi.org/10.1007/s11063-017-9703-6.

[10] Z. Zhang, B. Krawczyk, S. Garcìa, A. Rosales-Pérez, F. Herrera, Empowering one-vs-one decomposition with ensemble learning for multi-class imbalanced data, Knowl.-Based Syst. 106 (2016) 251–263, http://dx.doi.org/10.1016/j.knosys.2016.05.048.

[11] G.D.C. Cavalcanti, R.J.O. Soares, Ranking-based instance selection for pattern classification, Expert Syst. Appl. (2020) 113269, http://dx.doi.org/10.1016/j.eswa.2020.113269.

[12] S. Karlos, N. Fazakis, A.-P. Panagopoulou, S. Kotsiantis, K. Sgarbas, Locally application of naive Bayes for self-training, Evolving Systems 8 (1) (2016) 3–18, http://dx.doi.org/10.1007/s12530-016-9159-3.

[13] X.C. Pham, M.T. Dang, S.V. Dinh, S. Hoang, T.T. Nguyen, A.W.-C. Liew, Learning from data stream based on random projection and hoeffding tree classifier, in: 2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA), 2017, http://dx.doi.org/10.1109/dicta.2017.8227456.

[14] C.K. Aridas, S.B. Kotsiantis, M.N. Vrahatis, Increasing diversity in random forests using naive Bayes, Artif. Intell. Appl. Innov. 7 (2016) 5–86, http://dx.doi.org/10.1007/978-3-319-44944-9_7.

[15] B. Krawczyk, M. Woźniak, Untrained weighted classifier combination with embedded ensemble pruning, Neurocomputing 196 (2016) 14–22, http://dx.doi.org/10.1016/j.neucom.2016.02.040.

[16] A. Elen, M.K. Turan, Classifying white blood cells using machine learning algorithms, Int. J. Eng. Res. Dev. 11 (1) (2019) 141–152, http://dx.doi.org/10.29137/umagd.498372.

[17] M. Nilashi, O. Ibrahim, H. Ahmadi, L. Shahmoradi, A knowledge-based system for breast cancer classification using fuzzy logic method, Telemat. Inform. 34 (4) (2017) 133–144, http://dx.doi.org/10.1016/j.tele.2017.01.007.

[18] S.T. Chandrasekaran, R. Hua, I. Banerjee, A. Sanyal, A fully-integrated analog machine learning classifier for breast cancer classification, Electronics 9 (3) (2020) 515, http://dx.doi.org/10.3390/electronics9030515.

[19] H. Asri, H. Mousannif, H.A. Moatassime, T. Noel, Using machine learning algorithms for breast cancer risk prediction and diagnosis, Procedia Comput. Sci. 83 (2016) 1064–1069, http://dx.doi.org/10.1016/j.procs.2016.04.224.

[20] W.N.L.W.H. Ibeni, M.Z.M. Salikon, A. Mustapha, S.A. Daud, M.N.M. Salleh, Comparative analysis on Bayesian classification for breast cancer problem, Bull. Electr. Eng. Inform. 8 (4) (2019) 1303–1311, http://dx.doi.org/10.11591/eei.v8i4.1628.

[21] S. Das, D. Biswas, Prediction of breast cancer using ensemble learning, in: 2019 5th International Conference on Advances in Electrical Engineering (ICAEE) Dhaka, Bangladesh, 2019, pp. 804-808, http://dx.doi.org/10.1109/ICAEE48663.2019.8975544.

[22] M. Rudrapatna, A. Sowmya, Feature weighted minimum distance classifier with multi-class confidence estimation, in: AI 2006: Advances in Artificial Intelligence, 2006, pp. 253–263, http://dx.doi.org/10.1007/11941439_29.

[23] S.-S. Kim, D. Chung, B.-K. Park, J.-B. Kim, S.-D. Lee, Visual model of human blur perception for scene adaptive capturing, in: Digital Photography, Vol. 7250, 2009, pp. 1–10, http://dx.doi.org/10.1117/12.805828.

[24] H.A. Abu Alfeilat, A.B.A. Hassanat, O. Lasassmeh, A.S. Tarawneh, M.B. Alhasanat, H.S. Eyal Salman, V.B.S. Prasath, Effects of distance measure choice on k-nearest neighbor classifier performance: A review, Big Data (2019) http://dx.doi.org/10.1089/big.2018.0175.

[25] E.E. Hellinger, Neue Begründung der Theorie quadratischer Formen von unendlichvielen Veränderlichen, J. Reine Angew. Math. (Crelle's J.) 136 (1909) 210–271, http://dx.doi.org/10.1515/crll.1909.136.210.

[26] UCI machine learning repository: Breast cancer wisconsin (original) data set (WBCO), 2020, Retrieved 10 2020, http://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Original).

[27] UCI machine learning repository: LED display domain data set (LDD), 2020, Retrieved 10 2020, http://archive.ics.uci.edu/ml/datasets/LED+Display+Domain.