

# Comparison of different weighting schemes for the $k$ NN classifier on time-series data

Zoltan Geler<sup>1</sup>, Vladimir Kurbalija<sup>2</sup>, Miloš Radovanović<sup>2</sup>, Mirjana Ivanović<sup>2</sup>

<sup>1</sup> Faculty of Philosophy, University of Novi Sad

Dr Zorana Đinđića 2, 21000 Novi Sad, Serbia, gellerz@gmail.com

<sup>2</sup> Department of Mathematics and Informatics, Faculty of Sciences, University of Novi Sad

Trg D. Obradovića 4, 21000 Novi Sad, Serbia, {kurba,radacha,mira}@dmi.uns.ac.rs

## Corresponding author:

Zoltan Geler

Faculty of Philosophy

Dr Zorana Đinđića 2

21000 Novi Sad

Serbia

+381 21 4853918

**Abstract.** Many well-known machine-learning algorithms have been applied to the task of time-series classification, including decision trees, neural networks, support vector machines, and others. However, it was shown that the simple 1-nearest neighbor (1NN) classifier, coupled with an elastic distance measure like Dynamic Time Warping (DTW), often produces better results than more complex classifiers on time-series data, including  $k$ -nearest neighbor ( $k$ NN) for values of  $k > 1$ . In this article we revisit the  $k$ NN classifier on time-series data by considering 10 classic distance-based vote weighting schemes in the context of Euclidean distance, as well as four commonly used elastic distance measures: DTW, Longest Common Subsequence (LCS), Edit Distance with Real Penalty (ERP), and Edit Distance on Real sequence (EDR). Through experiments on the complete collection of UCR time-series datasets we confirm the view that the 1NN classifier is very hard to beat. Overall, for all considered distance measures, we found that variants of the Dudani weighting scheme produced the best results.

**Keywords:** Time series, classification, 1-nearest neighbor,  $k$ -nearest neighbor, weighted  $k$ -nearest neighbor, elastic distance measures

## 1. Introduction

Time series embody one form of sequential data where the data is ordered by a time parameter [36]. Each element of a time series describes the phenomenon under examination at a specific point in time. Depending on whether the observations were carried out continuously or at regular intervals we can differentiate continuous and discrete time series [5]. In this paper, we consider the simplest form of discrete time series whose elements are uniformly spaced real numbers.

Time series are used for storage, display and analysis of data across a wide range of different domains, including various areas of science, medicine, economics, ecology, telecommunications and meteorology [12, 22, 36]. Esling and Agon [12] emphasize that in almost every scientific field, measurements are performed over time and that the collected data can be organized in the form of

time series with the aim of extracting meaningful knowledge. For finding new and useful information from the sets of collected data we can rely on methods from statistical analysis and modeling, data mining and machine learning [36].

While research in statistical modeling techniques has a longer history [36], the need to process increasing volumes of data has heightened the interest for studying different task types of temporal data mining: indexing, classification, clustering, prediction, segmentation, anomaly detection and others [10, 21]. The area of interest of this work is primarily related to time-series classification.

In recent years, there is a growing interest for research in different aspects of time-series classification [18, 24, 48, 63, 64]. The possibility of applying many well-known machine learning techniques was investigated in this field. These techniques include: decision trees [52], neural networks [43], support vector machines [61], first order logic rules [51], Bayesian classifiers [46] and others. However, it was shown that the simple nearest-neighbor (1NN) approach often produces better results than the mentioned more complex classifiers for the time-series data [63].

The nearest neighbor algorithm is probably one of the most esteemed algorithms in data mining [60]. It is based on the following very simple idea: unknown samples are placed into the class of their nearest neighbors [8]. The majority voting  $k$ -nearest neighbor ( $k$ NN) rule generalizes this concept by finding the  $k$  nearest neighbors and choosing the class that is most frequent among them [14]. The distance-weighted  $k$ -nearest neighbor rule proposed by Dudani [11] assigns weights to the neighbors according to their distance from the unclassified sample: greater significance is given to closer neighbors.

Since finding the nearest neighbors constitutes the core idea behind the  $k$ NN rule, one of the most essential questions of its implementation is related to the selection of an appropriate distance measure. In the domain of time series, several different distance measures are applied for comparing data sequences. The most commonly used and most frequently investigated time-series distance measures are Euclidean distance [13], Dynamic Time Warping (DTW) [3], Longest Common Subsequence (LCS) [59], Edit Distance with Real Penalty (ERP) [7], and Edit Distance on Real sequence (EDR) [6].

Several different methods for assigning weights to the nearest neighbors are proposed in the literature [11, 19, 20, 35, 37, 39, 44, 65]. Generally, each paper that presents a new way of computing weights reports the superiority of the new method compared to some previous solutions. Several of these papers ([19, 20, 65]) compare various weighting schemes using a relatively small number of datasets (commonly 12) from the UCI machine learning repository [2]. The conclusions are usually based on comparing classification results using Euclidean distance.

The aim of this study is to compare different weighting schemes from the above mentioned papers in the domain of time series. Our research is motivated by the view that the simple 1NN rule gives better results than the majority voting  $k$ NN rule [10]. We will investigate whether the proposed weighting schemes can produce higher classification accuracies than the 1NN classifier and reexamine the view that the simple 1NN classifier is very hard to beat [63]. In order to achieve this objective our research encompasses the majority of all publicly available, labeled time-series datasets in the world which are provided by the UCR Time Series Repository [25]. Moreover, our examinations encompass the five most commonly used time-series distance measures (Euclidean

distance and the unconstrained versions of DTW, LCS, ERP and EDR) and provide statistical support to the results obtained.

The rest of this paper is organized as follows. The required background knowledge about the various weighting schemes of the weighted  $k$ -nearest neighbor rule is presented in Section 2. The time-series similarity measures of interest are also described in Section 2. Section 3 outlines our Framework for Analysis and Prediction (FAP) [28] which was used to carry out the experiments. A summary of the used datasets is also given in Section 3. The experiments and their results are presented in Section 4. Section 5 concludes the paper and discusses possible directions of future work.

## 2. Background and related work

### 2.1. The weighted $k$ -nearest neighbor rule

A time series  $Q$  of length  $n$  can be viewed as a sequence  $(q_1, q_2, \dots, q_n)$  of real numbers which describes the change of the observed phenomenon at equidistant points in time [7, 12, 66]. The task of classification is to determine the class (label) of an unclassified time series  $S$  based on a given training set of pre-classified time series  $T = \{(T_1, T_1^C), (T_2, T_2^C), \dots, (T_n, T_n^C)\}$  [12, 21, 36]. In the remainder of this paper we denote the set of classes assigned to the elements of the training set with  $C$ , i.e.

$$C = \{T_1^C, T_2^C, \dots, T_n^C\}$$

where  $T_i^C$  denotes the class of time series  $T_i$ .

According to Ratanamahatana et al. [49], the simple nearest neighbor rule (1NN) [8] is one of the most popular time-series classification methods. The class of a new sequence is determined by the closest (most similar) member of the training set. In [63], Xi et al. have shown that the combination of the 1NN rule and the DTW distance measure is a very competitive classifier compared to other more complex methods. Cover and Hart [8] proved that the asymptotic misclassification rate  $R$  of the 1NN rule satisfies the following condition:

$$R^* \leq R \leq R^* \left( 2 - R^* \frac{|C|}{|C| - 1} \right)$$

where  $R^*$  is the Bayesian probability of error.

One of the first formulations and analyses of the  $k$ -nearest neighbor rule originates from Fix and Hodges [14]. Let  $C = \{c_1, c_2\}$  and let the training set  $T$  contain  $n_i$  representatives of class  $c_i$  ( $i = 1, 2$ ). A new time series  $Q$  is labeled with class  $c_1$  if  $k_1/n_1 > k_2/n_2$ , where  $k_i$  denotes the number of training examples of class  $c_i$  which are among the  $k = k_1 + k_2$  nearest neighbors of  $Q$ .

The majority voting  $k$ -nearest neighbor rule ( $k$ NN) algorithm is a natural extension of the 1NN rule: a new series  $Q$  is labeled with the class that is most frequent among the  $k$  nearest neighbors of  $Q$  inside the training set  $T$ . The choice of the class can be formally written as follows:

$$Q^C = \arg \max_{c \in C} \left\{ \sum_{i=1}^k E(c = T_i^C) \right\} \quad (1)$$

where  $T_i^C$  denotes the class of the  $i$ -th nearest neighbor, and  $E(\cdot)$  is an indicator function that returns 1 if its argument is true and 0 otherwise. Ties are broken arbitrarily.

Wu et al. [60] highlight a number of issues related to the choice of  $k$ . If  $k$  is too small, the  $k$ NN rule can be sensitive to noise points. If  $k$  is too large, the closest neighbors can include many different classes. Another issue may arise from the equality of the neighbors in the process of majority voting regardless of their distance from the query object. This can be addressed by weighting the votes of the neighbors in accordance with their distance. If we denote the weight of the  $i$ -th nearest neighbor with  $w_i$ , Eq. (1) can be adjusted in the following way:

$$Q^C = \arg \max_{c \in C} \left\{ \sum_{i=1}^k w_i E(c = T_i^C) \right\} \quad (2)$$

The first weighted voting method for the  $k$ NN rule was proposed by Dudani in [11] (henceforth denoted Dudani). In this approach weights are taken from the interval  $[0,1]$ . The closest neighbor is weighted with 1, the farthest with 0 and the others are scaled between by the linear mapping defined in Eq. (3), where  $d_i$  denotes the distance between the query sequence  $Q$  and the  $i$ -th of the nearest neighbors.

$$w_i = \begin{cases} \frac{d_k - d_i}{d_k - d_1} & d_k \neq d_1, \\ 1 & d_k = d_1. \end{cases} \quad (3)$$

Dudani [11] has also suggested two additional alternative weighting schemes: the inverse distance weight (Inverse, Eq. (4)) and the rank weight (Rank, Eq. (5)).

$$w_i = \frac{1}{d_i} \quad (4)$$

$$w_i = k - i + 1 \quad (5)$$

Instead of the inverse distance we may rely on the inverse of the squared distance [35, 39, 60] (ISquared). In both of these cases there is a possibility of division by zero. This is usually solved by adding a small constant  $\varepsilon$  (we have used 0.001 in our experiments) to the denominator as in Eq. (6).

$$w_i = \frac{1}{d_i^2 + \varepsilon} \quad (6)$$

The weighting function in Eq. (3) excludes the  $k$ -th neighbor from the voting process in the situation when  $d_k \neq d_1$  since  $d_k - d_i = 0$  for  $i = k$ . Macleod et al. [37] provide a generalization of Dudani's weighting function by introducing two new parameters:  $s \geq k$  and  $\alpha \geq 0$  (Macleod, Eq. (7)).

Through them we can overcome this shortcoming. When  $s = k$  and  $\alpha = 0$ , Eq. (7) becomes the

original weighting scheme proposed by Dudani. From the several combinations of these parameters, which have been investigated in [37], we will use  $s = k$  with  $\alpha = 1$ .

$$w_i = \begin{cases} \frac{(d_s - d_i) + \alpha(d_s - d_1)}{(1 + \alpha)(d_s - d_1)} & d_s \neq d_1, \\ 1 & d_s = d_1. \end{cases} \quad (7)$$

In [44], Pao et al. have used the Fibonacci sequence as the weighting function (Fibonacci, Eq. (8)). They have compared this scheme with the three weighting methods defined by Dudani (linear mapping - Eq. (3), inverse distance - Eq. (4), and the rank weight - Eq. (5)) in the field of recognizing emotions from Mandarin speech. Beside the majority and weighted voting rules their study has also included two other variations of the  $k$ NN classifier: *Categorical Average Patterns* [57] and *Weighted Discrete kNN* [45]. They have found that the Fibonacci weighting function outperforms the others in all of the examined classifiers.

$$\begin{aligned} w_i &= w_{i+1} + w_{i+2} \\ w_k &= w_{k-1} = 1 \end{aligned} \quad (8)$$

Gou et al. [20] have introduced a weighting scheme calculated as the reciprocal value of the neighbors' rank (Uniform, Eq. (9)), and a weighting function (DualU, Eq. (10)) based on Dudani's linear mapping (Eq. (3)). They have compared the classification accuracies of the majority voting  $k$ NN classifier and its weighted voting forms based on these three methods of computing weights. The experiments were conducted using artificially generated data and real data selected from the UCI machine learning repository [2] with numeric attributes only. Their conclusion is that the combined weighting in Eq. (10) surpasses the other examined classifiers.

$$w_i = \frac{1}{i} \quad (9)$$

$$w_i = \begin{cases} \left( \frac{d_k - d_i}{d_k - d_1} \right) \frac{1}{i} & d_k \neq d_1, \\ 1 & d_k = d_1. \end{cases} \quad (10)$$

In [65] Zavrel has matched another weighting scheme against the linear mapping (Eq. (3)) and the inverse distance (Eq. (4)) weighting functions proposed by Dudani, as well as against the 1NN classifier and the majority voting rule. This scheme (Zavrel) is based on the exponential function as shown in Eq. (11) where  $\alpha$  and  $\beta$  are constants determining the slope and the power of the exponential decay function, respectively. The survey covered 12 datasets from the UCI machine learning repository [2] and one additional linguistic dataset supplied by the author. Zavrel has found that the weighted voting can improve  $k$ NN's accuracy and that Dudani's linear mapping (Eq. (3)) is superior to the other classifiers examined in his study. In our inquiry we will select  $\alpha = \beta = 1$ , similarly as in Zavrel's paper.

$$w_i = e^{-\alpha d_i^\beta} \quad (11)$$

The dual distance-weighted function (DualD) depicted with Eq. (12) was presented by Gou et al. in [19]. This novel weighted  $k$ NN rule extends Dudani's linear mapping (Eq. (3)): it weights the closest and the farthest neighbors the same way as the linear mapping, but assigns smaller values to those

between them. The authors have compared its classification accuracy with the accuracies of the 1NN, the  $k$ NN and Dudani's linear-mapping based weighted  $k$ NN rule. The dual distance-weighted function has performed better with each of the 12 sets from the UCI machine learning repository [2] which were used in the experiments.

$$w_i = \begin{cases} \left( \frac{d_k - d_i}{d_k - d_1} \right) \left( \frac{d_k + d_1}{d_k + d_i} \right) & d_k \neq d_1, \\ 1 & d_k = d_1. \end{cases} \quad (12)$$

## 2.2. Time-series distance measures

The main task in the process of classification of a new time series  $Q$  using the nearest neighbor rule is to find  $k$  time series from the training set that are closest to  $Q$ . The distance between two time series  $Q = (q_1, q_2, \dots, q_n)$  and  $S = (s_1, s_2, \dots, s_m)$  is defined using a proximity measure - a function that returns the nonnegative distance  $d(Q, S)$  between them [12]. A distance *metric* is a distance measure that for every time series  $Q, S$  and  $X$  satisfies the following conditions:

1. **Reflexivity:**  $d(Q, S) = 0$  if and only if  $Q = S$ ,
2. **Symmetry:**  $d(Q, S) = d(S, Q)$ ,
3. **Triangle inequality:**  $d(Q, S) \leq d(Q, X) + d(X, S)$ .

In the rest of this section distance measures used in our experiments will be briefly introduced.

**Euclidean distance.** The most common distance measure in time-series data mining is probably the Euclidean distance [1, 13, 62]. Assuming that  $Q$  and  $S$  are of the same length  $n$ , we can think of them as points in  $n$ -dimensional space. In this manner we will be able to calculate their distance relying on the differences between the corresponding elements of the sequences as in Eq. (13).

$$d(Q, S) = \sqrt{\sum_{i=1}^n (q_i - s_i)^2} \quad (13)$$

The advantage of Euclidean distance is that it is very easy to compute and to understand. Furthermore, it fulfills the above mentioned conditions to be a distance metric and therefore it can be used for indexing time series in databases [13]. There are, however, some disadvantages, too: the sequences must have the same number of points (can be avoided by interpolation to equal length [50]), it is sensitive to shifting and scaling along the  $y$ -axis (can be precluded by normalizing the series [9, 17]), and it is also sensitive to distortions and shifting along the time axis [26].

**Dynamic Time Warping (DTW).** Euclidean distance is based on linear aligning of related points of time series (Fig. 1 (a)): the  $i$ -th point of the first series is paired with the  $i$ -th point of the second one. The assessment of the distance can be improved by warping the time axis of one or both of the sequences (Fig. 1 (b)). One of the most popular distance measures based on non-linear aligning is the Dynamic Time Warping [3, 26, 63].

DTW searches for an optimal warping path in the matrix of distances between the components of the time series  $Q$  and  $S$  (Fig. 1 (c)). A warping path  $W = w_1, w_2, \dots, w_k$  (where  $\max(n, m) \leq k \leq$

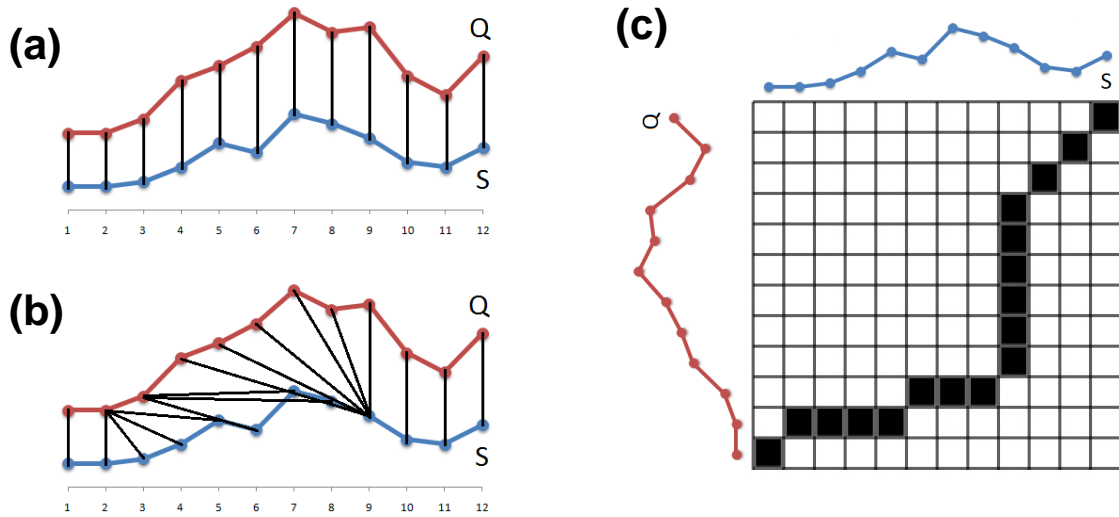
$m + m - 1$ ) represents a sequence of adjacent cells from the matrix. Each element of a warping path is of the form  $w_i = (x_i, y_i)$  (where  $x_i$  denotes a row of the matrix and  $y_i$  denotes a column of the matrix) and the warping path must satisfy the following constraints:

- **Boundary condition** - the first and the last element of the warping path are in diagonally opposite corners of the matrix:  $w_1 = (1,1)$ ,  $w_k = (n, m)$ .
- **Continuity condition** - matrix cells denoted by adjacent elements of the warping path must be neighbors:  $x_{i+1} - x_i \leq 1$ ,  $y_{i+1} - y_i \leq 1$ .
- **Monotonicity condition** - the warping path must be monotonically non-decreasing along the time axis:  $x_{i+1} - x_i \geq 0$ ,  $y_{i+1} - y_i \geq 0$ .

From the set of all possible warping paths we are seeking for the optimal one, which minimizes the warping cost (the sum of the cells that constitute the warping path). This can be found using dynamic programming, as recursively defined by Eq. (14).

$$D(i, j) = \begin{cases} 0 & i = j = 0, \\ \infty & i = 0, j > 0 \text{ or } i > 0, j = 0, \\ d(q_i, s_j) + \min \begin{cases} D(i-1, j-1) \\ D(i-1, j) \\ D(i, j-1) \end{cases} & i, j \geq 1. \end{cases} \quad (14)$$

where  $d(q_i, s_j)$  is the distance between  $q_i$  and  $s_j$ . The distance between  $Q$  and  $S$  is then defined as  $DTW(Q, S) = D(n, m)$ . Euclidean distance can be seen as a special warping path which contains only the diagonal elements on the distance matrix, and which is defined if the time series are of the same length.



**Fig. 1 (a)** Linear aligning of time series in case of the Euclidean distance, **(b)** Non-linear aligning of DTW, **(c)** Warping path inside the warping matrix

**Longest Common Subsequence (LCS).** This approach calculates the distance between two sequences relying on a variation of the edit distance technique - a well known method in the field of string processing. The basic idea is to express the similarity of the time series based on the length of their longest common subsequence [59]. The length of the LCS can be computed using dynamic programming based on the recursive definition in Eq. (15). The condition  $q_i = s_j$  is usually too strong

for time series and it is often replaced with a parameterized condition  $|q_i - s_j| \leq \epsilon$ , where  $0 < \epsilon < 1$ . The dissimilarity  $LCS(Q, S)$  between  $Q$  and  $S$  is calculated according to Eq. (16) as presented in [49] and [9].

$$L(i, j) = \begin{cases} 0 & i = 0 \text{ or } j = 0, \\ 1 + L(i - 1, j - 1) & i, j > 0 \text{ and } q_i = s_j, \\ \max(L(i - 1, j), L(i, j - 1)) & i, j > 0 \text{ and } q_i \neq s_j. \end{cases} \quad (15)$$

$$LCS(Q, S) = \frac{n + m - 2L(n, m)}{n + m} \quad (16)$$

**Edit Distance with Real Penalty (ERP).** Ding et al. in [10] refer to distance measures that utilize linear aligning between the points of time series as *lock-step* measures (Euclidean distance and other forms of the  $L_p$  norm for  $p \geq 1$ ). One of their main advantages is that they represent distance metrics and thus they can easily be applied for indexing in databases. However, the fixed mapping between the points makes them sensitive to noise and to time shiftings. *Elastic* measures like DTW and LCS address these issues by allowing one-to-many (DTW) and one-to-many/one-to-none (LCS) mappings [10]. Since neither DTW nor LCS satisfy the triangle inequality [55], they are non-metric distance measures.

In [7] Chen and Ng propose the ERP distance function as a combination of the  $L_1$  norm and the elastic distance measures. To handle local time shiftings it calculates the real penalty between non-gap elements using  $L_1$  distance. The distance for gaps is computed based on a constant value denoted by  $g$  (the default value is 0) in the definition of this measure (Eq. (17)). ERP is a distance metric, but it is sensitive to noise [6].

$$E(i, j) = \begin{cases} \sum_{k=1}^j |s_k - g| & i = 0, \\ \sum_{k=1}^i |q_k - g| & j = 0, \\ \min \begin{cases} |q_i - s_j| + E(i - 1, j - 1) \\ |q_i - g| + E(i - 1, j) \\ |s_j - g| + E(i, j - 1) \end{cases} & \text{otherwise} \end{cases} \quad (17)$$

**Edit Distance on Real sequence (EDR).** EDR [6] is an elastic distance measure based on edit distance which has been developed with the aim to improve the accuracy of LCS in the case when the time series contains similar sub-sequences with gaps of different sizes between them. EDR is robust to noise, outliers and local time shifting. In contrast to LCS, EDR assigns penalties according to the lengths of the gaps, but EDR also does not represent a distance metric. This measure is defined as follows:



$$E(i, j) = \begin{cases} j & i = 0, \\ i & j = 0, \\ \min \begin{cases} subcost(i, j) + E(i - 1, j - 1) \\ 1 + E(i - 1, j) \\ 1 + E(i, j - 1) \end{cases} & otherwise. \end{cases} \quad (18)$$

The subcost in Eq. (18) is calculated by the following formula:

$$subcost(i, j) = \begin{cases} 0 & |q_i - s_j| \leq \varepsilon, \\ 1 & |q_i - s_j| > \varepsilon. \end{cases}$$

## 2.3 Motivation for the evaluation

In several significant research contributions in the field of time-series data mining [10, 48, 58, 63], the simple 1NN classifier is selected as one of the most accurate classifiers, demonstrating comparable and even superior performance than many more complex classification techniques. Xi et al. [63] highlight that in this domain the simple 1NN classifier (in combination with DTW) is very hard to beat.

The majority voting  $k$ -nearest neighbor rule ( $k$ NN) generalizes the idea of the 1NN rule by taking into account not one, but  $k$  nearest neighbors. As an attempt to improve classification accuracy, several different methods for assigning weights to the nearest neighbors are proposed in the literature [11, 19, 20, 35, 37, 39, 44, 65]. However, these papers are not directly related to the domain of time series and every newly introduced weighting scheme is compared only to a few other ones. Furthermore, the experiments are commonly based on a small number of datasets (usually from the UCI machine learning repository [2]), the examinations are performed exclusively in combination with Euclidean distance and the results are not supported by statistical tests.

To the best of our knowledge, our work represents the most comprehensive analysis of different weighting schemes for the  $k$ NN classifier in the field of time-series data mining. It includes the five most commonly used distance measures (Euclidean distance, DTW, LCS, ERP and EDR), a large number of weighting functions and the majority of all publicly available, labeled time-series datasets in the world (provided by the UCR Time Series Repository [25]). In addition, we also provide statistical support to the obtained results.

## 3. The Framework for Analysis and Prediction and the datasets

Time-series data are generated and utilized in many domains of science, economics and medicine, and the amount of data from which we need to extract valuable information is continuously increasing. This has led to a growing interest of studying different fields of temporal data analysis [5, 9, 10, 12, 21, 49]. We can distinguish between two distinct field of time series research [9, 36]: statistical analysis and modeling on one side, and the data mining and machine learning approach on the other side.

According to Das and Gunopulos [9] the statistical approach is mainly interested in identifying patterns, trend analysis, seasonality and forecasting. On the other hand, temporal data mining is focused on database management and on research tasks like indexing, classification, clustering,

prediction, data representation and distance measurements [10, 12, 36]. Ratanamahatana et al. [49] and Das and Gunopulos [9] emphasize that the methods studied by statisticians are of little furtherance for researchers of time series-data mining. As a consequence, it has become increasingly important to develop new methodologies for different task types of temporal data mining and to investigate and enhance the existing ones. However, the new solutions are usually separately implemented and described in different publications where the comparison is often based only on a very limited number of case studies.

Motivated by these considerations we have developed an open source, multipurpose and multifunctional library for researchers and practitioners in the field of time-series data mining. Our Framework for Analysis and Prediction (FAP)[28] is written in Java and it is designed to be a free and extensible software package which will cover all of the main tasks of temporal data mining and analysis. The incitement behind this framework is to support and alleviate the investigation and comparison of different techniques utilized in this domain.

In its latest version, beside the implementation of the most commonly used distance measures described in Section 2.2, FAP contains several others including different forms of the  $L_p$  norm, Time Warp Edit Distance (TWED) [38], Spline [29], Swale [42] and the Canberra distance [27]. The elastic distance measures (DTW, LCS, ERP, EDR, TWED) are implemented in three ways: without constraining the warping window, using the Sakoe-Chiba band [53], and employing the Itakura parallelogram [23]. Along with the nearest neighbor rule and the majority voting NN rule, our library incorporates all of the different weighting schemes outlined in Section 2.1. Among the methods for testing the accuracy of classifiers, FAP currently supports stratified k-fold cross validation (SCV), leave-one-out (LOO) and the holdout method, with the support for training/testing method of error classification estimation. There are also several classes implementing pre-processing transformations including scaling, shifting, min-max normalization, z-score normalization, decimal scaling and linear equiscaling. Various time-series representations are also supported: Piecewise Linear Approximation (PLA), Piecewise Aggregate Approximation (PAA), Adaptive Piecewise Constant Approximation (APCA), Symbolic Aggregate Approximation (SAX) and Spline.

The Framework for Analysis and Prediction has been already successfully employed within various research domains including: investigation of the influence of global constraints on distance measures [31, 32], developing a distributed distance matrix generator based on agents [40, 41], mining time series in the psychological domain [30, 34] and time-series analysis in the neurology domain [33]. FAP might be applied in other domains too, for example, signal processing [56] or image processing [54].

The experiments performed in this research are executed on 46 datasets from [25] (Table 1), which includes the majority of all publicly available, labeled time-series datasets in the world. The length of time series varies from 24 to 1882 (column  $L$ ), depending of the dataset. The number of time series per dataset varies from 56 to 9236 (column  $S$ ) and the number of classes varies from 2 to 50 (column  $C$ ). Column  $ID$  contains the labels assigned to datasets that are used in the tables with detailed results of the experiments presented in the Appendix of this paper.

In the case of LCS and EDR the distance also depends on the matching threshold  $\epsilon$ . The value of this parameter was determined as proposed in [6]. Let  $\sigma(T)$  denote the standard deviation over the points of time series  $T$ . The matching threshold is equal to  $(\max \sigma(T_i))/4$  (for all the time series  $T_i$

in a dataset). The obtained values of  $\epsilon$  are presented in the corresponding columns of Table 1 (the same values are used for LCS and EDR as in [6]). The aim of our work is not to show how to pick optimal values for the threshold parameter  $\epsilon$ , but to investigate different extensions of the 1NN classifier. The  $\epsilon$  values listed in Table 1 work reasonably well for kNN classification (see Table 2), therefore we use them in all our experiments.

ID	Dataset	S	L	C	$\epsilon$	ID	Dataset	S	L	C	$\epsilon$
1	50words	905	270	50	0.249537	24	mallat	2400	1024	8	0.249878
2	adiac	781	176	37	0.249289	25	medicalimages	1141	99	10	0.248734
3	beef	60	470	5	0.024453	26	motes	1272	84	2	0.248507
4	car	120	577	4	0.249783	27	noninvasivefatalecg_thorax1	3765	750	42	0.249833
5	cbf	930	128	3	0.249022	28	noninvasivefatalecg_thorax2	3765	750	42	0.249833
6	chlorineconcentration	4307	166	3	0.249246	29	oliveoil	60	570	4	0.084183
7	cinc_ecg_torso	1420	1639	4	0.249924	30	osuleaf	442	427	6	0.250294
8	coffee	56	286	2	2.984938	31	plane	210	144	7	0.256236
9	cricket_x	780	300	12	0.548431	32	sonyaiborobotsurface	621	70	2	0.248208
10	cricket_y	780	300	12	0.564973	33	sonyaiborobotsurfaceii	980	65	2	0.248069
11	cricket_z	780	300	12	0.532931	34	starlightcurves	9236	1024	3	0.249879
12	diatomsizereduction	322	345	4	0.249637	35	swedishleaf	1125	128	15	0.249022
13	ecg200	200	96	2	0.248695	36	symbols	1020	398	6	0.249686
14	ecgfivedays	884	136	2	0.249079	37	synthetic_control	600	60	6	0.247908
15	faceall	2250	131	14	0.249044	38	trace	200	275	4	0.249545
16	facefour	112	350	4	0.249643	39	twoleadecg	1162	82	2	0.248471
17	fish	350	463	7	0.306026	40	twopatterns	5000	128	4	0.249022
18	gun_point	200	150	2	0.249165	41	uwavegesturelibrary_x	4478	315	8	0.249603
19	haptics	463	1092	5	0.249886	42	uwavegesturelibrary_y	4478	315	8	0.249603
20	inlineskate	650	1882	7	0.249934	43	uwavegesturelibrary_z	4478	315	8	0.249603
21	italypowerdemand	1096	24	2	0.244736	44	wafer	7164	152	2	0.249176
22	lighting2	121	637	2	0.249804	45	wordssynonyms	905	270	25	0.249537
23	lighting7	143	319	7	0.249608	46	yoga	3300	426	2	0.249706

**Table 1.** Characteristics of the UCR datasets used in the experiments

## 4. Experimental results

In Section 2.1 we have seen that a number of different extensions of the nearest-neighbor rule are proposed in the literature in order to improve the accuracy of the classification. Assertions about the superiority of a new weighting scheme are often founded on comparing classification errors considering a small number of datasets (commonly from the UCI machine learning repository [2]). Moreover, in all of these experiments the distance between objects is determined solely by Euclidean distance. In the domain of time series, however, besides this lock-step measure several other elastic measures are also in use (the most widely used are briefly outlined in Section 2.2). In this section we will report the results of our investigation of various NN classifiers on a large number of datasets from the UCR Time Series Repository [25] (Table 1) considering both the Euclidean and the unconstrained elastic distance measures. These datasets encompass various different domains, including biology, astronomy, robotics, medicine, etc. In addition to classification accuracies, we will also provide statistical support to our findings.

In these experiments, the accuracy of classification is obtained by 10 runs of stratified 10-fold cross-validation (SCV10x10) using the best value of parameter  $k$  obtained in the range from 1 to 30 by stratified 9-fold cross-validation (SCV1x9) on the respective training sets from the folds. Within each of the 10 runs, the datasets are randomly divided into 10 stratified disjoint folds of approximately equal size. The testing is then performed through 10 iterations: one (not yet selected) fold is used for testing, and the other nine for finding the best  $k$  value using SCV1x9. This approach ensures that the testing and training subsets are completely separated. Detailed results are shown in the Appendix of this paper.

	Euclidean distance			DTW			LCS			ERP			EDR		
	Win	Error	k	Win	Error	k	Win	Error	k	Win	Error	k	Win	Error	k
1NN	9	0.1595		15	0.1404		10	0.1611		15	0.1294		6	0.1620	
kNN	6	0.1605	2.99	7	0.1394	2.80	6	0.1529	4.71	7	0.1298	2.80	4	0.1556	4.84
Inverse	5	0.1593	3.55	9	0.1373	3.97	10	0.1491	6.14	7	0.1274	3.35	5	0.1505	6.28
ISquared	6	0.1586	4.11	10	0.1372	4.88	7	0.1503	5.94	9	0.1253	3.70	9	0.1489	7.22
Rank	6	0.1601	4.11	8	0.1399	3.97	6	0.1542	6.38	8	0.1288	3.96	4	0.1559	6.53
Fibonacci	6	0.1585	3.94	8	0.1378	4.27	8	0.1547	5.19	11	0.1263	4.16	8	0.1564	5.16
Dudani	11	0.1571	6.58	9	0.1369	5.90	6	0.1481	8.78	13	0.1249	6.45	9	0.1488	9.16
Macleod	6	0.1601	3.44	7	0.1397	3.37	4	0.1520	5.50	7	0.1283	3.35	4	0.1526	5.60
DualD	10	0.1567	6.87	19	0.1359	6.56	19	0.1474	9.47	14	0.1242	6.72	13	0.1480	9.75
Zavrel	10	0.1587	5.16	12	0.1380	5.10	6	0.1522	5.10	12	0.1285	3.42	10	0.1510	6.76
Uniform	8	0.1570	6.25	12	0.1362	7.43	7	0.1550	8.44	9	0.1258	6.66	10	0.1560	8.65
DualU	13	0.1571	13.22	11	0.1369	13.98	12	0.1485	15.56	17	0.1254	14.49	9	0.1497	16.24

**Table 2.** Comparison the average accuracies of different NN classifiers for the five most commonly used time-series distance measures

The average classification errors of the examined NN classifiers and the average values of the parameter  $k$  are presented in Table 2. The best results are marked with symbol ●, and the weakest ones with symbol ○. Column *Win* denotes the number of datasets for which the given  $k$ NN classifier gave the smallest classification error in comparison to the other classifiers. With respect to this category the best result is achieved by the dual distance-weighted  $k$ NN rule defined by Eq. (12) in Section 2.1 (for DTW DTW, LCS and EDR), the DualU weighting scheme (Euclidean distance and ERP).

From Table 2 we can notice that in terms of average classification accuracy the simple nearest neighbor classifier underperforms most of the other forms of the  $k$ NN classifier in case of all considered distance measures. On the other hand, the best result is always obtained using the dual distance-weighting scheme. It is also worth to notice that the differences between the best and worst average results are not particularly big: 0.0038 (Euclidean distance), 0.0045 (DTW), 0.0137 (LCS) , 0.0055 (ERP), 0.0140 (EDR).

By comparing the average values of parameter  $k$ , it is evident that (in order to achieve the best accuracy) the largest number of neighbors is required by the DualU weighting scheme (Eq. (10)): the average value is greater than 13 for all investigated distance measures. The required set of the closest objects is smallest in case of the majority voting NN rule - the mean value is less than 5 for all of the five measures.

In the rest of this section we will describe the results obtained for each distance measure in more detail. We will provide a summary of comparison of the simple nearest-neighbor rule and the other versions of the  $k$ NN classifier in terms of the number of datasets for which they achieved better and worse classification accuracies (Tables 3, 6, 480, 431 and 495). The average differences of the classification accuracies between the simple nearest-neighbor rule and the other considered classifiers are also listed in these tables.

In order to see whether there is a statistically significant difference between the analyzed classifiers, we counted the statistically significant wins and losses according to the corrected resampled t-test [4] at significance level 0.001, for each classifier. These results are presented in Tables 1052, 1271, 1600, 1185 and 1842. In addition, we compared the average error rates across all datasets using the Wilcoxon sign-rank test [15] adjusted by the one-step Bonferroni scheme [16]. The obtained adjusted  $p$  values are shown in Tables 5, 8, 40, 58 and 49. The rows and columns of these tables are sorted by overall average error rates (cf. Table 2), with the upper triangle of the symmetrical matrix of  $p$  values shown, thus enabling the reader to easily inspect the  $p$  values for methods to which a given method is supposedly superior to (reading by rows) or inferior to (reading by columns).

#### 4.1. Euclidean distance

From Table 3 we can see that, compared to the simple nearest neighbor rule, among all of the observed classifiers, the Dudani, the DualD and the DualU weighting schemes gave better accuracies for the largest number of datasets (28, 28 and 25, respectively) and that the  $k$ NN classifier and the Rank weighting scheme gave the largest number of datasets with higher classification error than 1NN (26 and 25). The largest number of statistically significant wins (Table 4), and the smallest number of losses were produced by the Dudani weighting scheme. The DualD and the DualU schemes provide the second and the third best results in this comparison too. We can also notice that the majority voting  $k$ -nearest neighbor and 1NN classifiers have the smallest number of statistically significant wins.

It can be seen in Table 5 that according to the adjusted sign-rank test the DualD scheme exhibits significant superiority to all schemes except Uniform, DualU and Dudani. On the other hand, the error rates of Dudani are judged to be significantly better than the other schemes (except DualD, DualU, Zavrel and Unifrom). The DualU scheme also performed strongly according to this test, exhibiting very low  $p$  values against most methods with lower average error rates, with the exceptions being Dudani and Zavrel. In all, it can be said that both statistical testing methodologies agree that DualD, Dudani and DualU are the best weighting schemes to use in combination with Euclidean distance, and that Uniform is the fourth best behind the mentioned three.

	$k$ NN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
Better than 1NN	11	13	16	11	18	<b>28</b>	14	<b>28</b>	19	19	25
Average diff.	0.0092	0.0102	0.0100	<b>0.0118</b>	0.0063	0.0083	0.0094	0.0084	0.0078	0.0096	0.0054
Worse than 1NN	<b>26</b>	24	24	25	22	15	24	15	19	22	12
Average diff.	0.0056	0.0050	0.0049	0.0061	0.0030	<b>0.0082</b>	0.0067	0.0072	0.0056	0.0030	0.0017

**Table 3.** Comparison of  $k$ NN with 1NN in case of Euclidean distance

	Dudani	DualD	DualU	Uniform	ISquared	Zavrel	Fibonacci	Inverse	Macleod	Rank	kNN	NN
Wins	97	98	72	53	36	45	35	20	20	21	10	17
Losses	9	12	30	26	33	51	50	41	47	57	80	88
Wins-Losses	88	86	42	27	3	-6	-15	-21	-27	-36	-70	-71

**Table 4.** Statistically significant wins and losses counts for different NN classifiers, across all datasets, in case of Euclidean distance

	ERROR	0.1567	0.1570	0.1571	0.1571	0.1585	0.1586	0.1587	0.1593	0.1595	0.1601	0.1601	0.1605
ERROR		DualD	Uniform	DualU	Dudani	Fibonacci	ISquared	Zavrel	Inverse	NN	Rank	Macleod	kNN
0.1567	DualD		0.70191	1.00000	1.00000	0.04547	0.01305	0.16922	0.01100	0.15400	0.02063	0.00236	0.00595
0.1570	Uniform			1.00000	0.49978	0.74422	0.30360	1.00000	0.34616	1.00000	0.46096	0.32428	0.08732
0.1571	DualU				1.00000	0.03214	0.06143	1.00000	0.00384	0.05153	0.00287	0.01322	0.00194
0.1571	Dudani					0.02226	0.03275	0.61199	0.00595	0.24021	0.02964	0.00434	0.00400
0.1585	Fibonacci						1.00000	1.00000	0.57312	1.00000	0.38958	0.58591	1.00000
0.1586	ISquared							1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.1587	Zavrel								1.00000	1.00000	1.00000	1.00000	0.22457
0.1593	Inverse									1.00000	1.00000	1.00000	0.11749
0.1595	NN										1.00000	1.00000	0.47544
0.1601	Rank											1.00000	1.00000
0.1601	Macleod												1.00000
0.1605	kNN												

**Table 5.** Bonferroni-adjusted  $p$  values for the pairwise Wilcoxon sign-rank test of the differences in average error rates across the datasets, in case of Euclidean distance

## 4.2. Dynamic Time Warping (DTW)

The situation in the case of DTW is similar as in the case of Euclidean distance. From Table 6 we can see, that the kNN classifier has generated the smallest number of datasets (11) with better results than the 1NN classifier, and the DualU and DualD classifiers the largest number (24). The second best result was obtained by the Zavrel method (23 datasets with better accuracies than the simple nearest-neighbor rule). This is reflected also in the number of statistically significant wins and losses (Table 7): after 1NN, the majority voting  $k$ -nearest neighbor rule has the second worst ratio of wins and losses (-60). The largest number of wins (108) was produced by the DualD scheme followed by the Dudani method (88 wins). Interestingly, according to the corrected resampled t-test the DualU weighting scheme did not rank among the best ones.

According to the results of the adjusted sign-rank test (Table 8), the DualD scheme outperforms all of the other classifiers (except DualU, Zavrel and to a lesser extent 1NN). In contrast to the Euclidean distance, the Dudani method did not produce low  $p$  values against most of the weighting schemes with lower average error rates (except Macleod and Rank). The Wilcoxon sign-rank test does not seem to confirm the wins-losses result of the Inverse scheme from Table 7, either: it exhibited higher  $p$  values against all other classifiers (among the ones with lower average error rates). Again, both of the statistical tests affirm that in combination with the DTW distance measure the DualD weighting scheme is the best choice. However, the results in column NN clearly indicate that the 1NN classifier is very hard to beat (especially in case of DTW).

	kNN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
Better than 1NN	11	19	20	13	17	22	14	<b>24</b>	23	16	<b>24</b>
Average diff.	<b>0.0206</b>	0.0167	0.0148	0.0184	0.0140	0.0150	0.0179	0.0150	0.0106	0.0186	0.0087
Worse than 1NN	<b>24</b>	20	19	23	20	17	<b>24</b>	15	14	21	14
Average diff.	0.0069	0.0080	0.0071	0.0087	0.0051	0.0090	0.0085	<b>0.0091</b>	0.0086	0.0043	0.0023

**Table 6.** Comparison of kNN with 1NN in case of DTW

	DualD	Dudani	Inverse	Zavrel	Uniform	DualU	ISquared	Macleod	Fibonacci	Rank	kNN	NN
Wins	108	88	60	73	60	54	51	29	30	29	23	27
Losses	14	15	31	50	40	46	51	52	57	58	83	135
Wins-Losses	94	73	29	23	20	8	0	-23	-27	-29	-60	-108

**Table 7.** Statistically significant wins and losses counts for different NN classifiers, across all datasets, in case of DTW

ERROR	0.1359	0.1362	0.1369	0.1369	0.1372	0.1373	0.1378	0.1380	0.1394	0.1397	0.1399	0.1404
ERROR	DualD	Uniform	DualU	Dudani	ISquared	Inverse	Fibonacci	Zavrel	kNN	Macleod	Rank	NN
0.1359	DualD	0.03479	1.00000	0.00457	0.25074	0.03199	0.01462	0.66578	0.00460	0.00016	0.00025	0.34188
0.1362	Uniform		1.00000	0.56725	1.00000	1.00000	1.00000	1.00000	1.00000	0.21505	0.41112	1.00000
0.1369	DualU			1.00000	1.00000	1.00000	0.29673	1.00000	0.13473	0.09454	0.05906	0.23230
0.1369	Dudani				0.73143	0.11718	0.37950	1.00000	0.15009	0.00059	0.00062	1.00000
0.1372	ISquared					1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.1373	Inverse						1.00000	1.00000	1.00000	1.00000	1.00000	1.00000
0.1378	Fibonacci							1.00000	1.00000	0.57177	0.54839	1.00000
0.1380	Zavrel								1.00000	1.00000	1.00000	0.17366
0.1394	kNN									1.00000	1.00000	1.00000
0.1397	Macleod										1.00000	1.00000
0.1399	Rank											1.00000
0.1404	NN											

**Table 8.** Bonferroni-adjusted  $p$  values for the pairwise Wilcoxon sign-rank test of the differences in average error rates across the datasets, in case of DTW

### 4.3. Longest Common Subsequence (LCS)

In comparison with the nearest neighbor rule, by the number of datasets with smaller classification error (Table 9), the best ranked classifiers are Dudani and DualD (33). The second best result was produced by the DualU scheme (32). The smallest difference (24) was found in the case of the kNN rule. The largest number of statistically significant wins (Table 10), and the smallest number of losses were produced by the DualD weighting scheme. Like in the case of DTW, it is followed by the Dudani method. The third place was taken by the DualU scheme. The worst ratio of wins and losses was produced by the simple nearest neighbor rule. It is followed by the kNN classifier, similarly to the previous distance measures.

The results of the Bonferroni-adjusted Wilcoxon sign-rank test (Table 11) support the findings of the corrected resampled t-test (Table 10): the DualD weighting scheme surpasses all of the other analyzed methods (except DualU). Analogous to the results for Euclidean distance, in terms of the obtained  $p$  values, the Dudani method outmatched all of the weighting schemes with lower average error rates (except DualU and to a lesser extent Inverse). On the other hand, the DualU method did not achieved low  $p$  values against most of the methods with lower average error. Based on the

findings of the statistical tests we come to a similar conclusion as in the case of the Euclidean distance and the DTW distance measure: DualD and Dudani are the best weighting schemes to use in combination with LCS.

	kNN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
Better than 1NN	24	29	29	25	31	<b>33</b>	27	<b>33</b>	27	28	32
Average diff.	0.0214	0.0226	0.0211	0.0176	0.0110	0.0214	0.0220	<b>0.0226</b>	0.0212	0.0129	0.0188
Worse than 1NN	<b>19</b>	14	14	18	10	11	17	11	16	14	7
Average diff.	0.0073	0.0074	0.0083	0.0070	0.0045	0.0101	0.0105	<b>0.0105</b>	0.0103	0.0057	0.0030

**Table 9.** Comparison of kNN with 1NN in case of LCS

	DualD	Dudani	DualU	Inverse	Uniform	ISquared	Fibonacci	Macleod	Zavrel	Rank	kNN	NN
Wins	148	130	86	72	64	46	61	52	38	41	27	30
Losses	12	18	59	47	53	44	67	63	58	80	94	200
Wins-Losses	136	112	27	25	11	2	-6	-11	-20	-39	-67	-170

**Table 10.** Statistically significant wins and losses counts for different NN classifiers, across all datasets, in case of LCS

ERROR	0.1474	0.1481	0.1485	0.1491	0.1503	0.1520	0.1522	0.1529	0.1542	0.1547	0.1550	0.1611
ERROR	DualD	Dudani	DualU	Inverse	ISquared	Macleod	Zavrel	kNN	Rank	Fibonacci	Uniform	NN
0.1474	DualD	0.03805	0.98764	0.01725	0.00317	0.00025	0.00002	0.00023	0.00100	0.03607	0.03347	0.00176
0.1481	Dudani		1.00000	0.30537	0.01950	0.00037	0.00066	0.00059	0.00272	0.11719	0.15683	0.00267
0.1485	DualU			1.00000	0.59575	0.22113	0.10611	0.00274	0.03469	1.00000	1.00000	0.00007
0.1491	Inverse				1.00000	0.27745	1.00000	0.13919	0.32943	1.00000	1.00000	0.06862
0.1503	ISquared					0.39280	0.18368	0.00921	0.14131	1.00000	1.00000	0.06143
0.1520	Macleod						1.00000	0.92880	1.00000	0.85687	0.51696	0.63697
0.1522	Zavrel							0.52932	1.00000	0.21667	0.07439	0.32222
0.1529	kNN								0.68922	0.02644	0.02246	1.00000
0.1542	Rank									0.43511	0.21642	0.44756
0.1547	Fibonacci										1.00000	0.00402
0.1550	Uniform											0.03629
0.1611	NN											

**Table 11.** Bonferroni-adjusted  $p$  values for the pairwise Wilcoxon sign-rank test of the differences in average error rates across the datasets, in case of LCS

#### 4.4. Edit distance with Real Penalty (ERP)

The results in Table 12 show, the largest number of datasets, for which we detected better classification accuracies than for the 1NN rule, was produced by the DualU (28), the DualD (24) and the Dudani (23) weighting schemes, analogously as in the previous cases. The simple 1NN rule has the smallest number of statistically significant wins. As in the case of the DTW and LCS distances, the best difference between wins and losses was achieved by the DualD weighting scheme (Table 13). It is followed by the Dudani and the Uniform methods.

Based on the results of the Bonferroni-adjusted Wilcoxon sign-rank test in Table 14, we can see that the DualD scheme outperforms almost all of the other methods: it exhibits higher  $p$  values only against the Dudani and the DualU schemes. We can also notice, that the method proposed by Dudani performed strongly, too. With respect to the average classification accuracy the ISquared



scheme precedes every other method (except DualD and Dudani), but it did not produce lower  $p$  values against the schemes with lower average error rates. Overall, both of the statistical tests indicate that DualD and Dudani methods represent the best choices in combination with ERP, too.

	kNN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
Better than 1NN	13	15	17	15	17	23	15	24	23	19	<b>28</b>
Average diff.	0.0120	0.0160	<b>0.0169</b>	0.0142	0.0123	0.0142	0.0145	0.0138	0.0029	0.0131	0.0069
Worse than 1NN	<b>26</b>	24	23	23	19	16	24	14	11	20	8
Average diff.	0.0066	0.0061	0.0044	<b>0.0080</b>	0.0035	0.0073	0.0070	0.0066	0.0025	0.0041	0.0010

**Table 12.** Comparison of kNN with 1NN in case of ERP

	DualD	Dudani	Uniform	DualU	ISquared	Fibonacci	Inverse	Rank	Macleod	KNN	Zavrel	NN
Wins	90	97	58	73	53	41	36	35	32	19	35	17
Losses	5	14	28	48	31	41	37	43	41	72	99	127
Wins-Losses	85	83	30	25	22	0	-1	-8	-9	-53	-64	-110

**Table 13.** Statistically significant wins and losses counts for different NN classifiers, across all datasets, in case of ERP

ERROR	0.1242	0.1249	0.1253	0.1254	0.1258	0.1263	0.1274	0.1283	0.1285	0.1288	0.1294	0.1298
ERROR	DualD	Dudani	ISquared	DualU	Uniform	Fibonacci	Inverse	Macleod	Zavrel	Rank	NN	kNN
0.1242	DualD	1.00000	0.00566	1.00000	0.00289	0.01062	0.00005	0.00004	0.28407	0.00002	0.08383	0.00001
0.1249	Dudani		0.02670	1.00000	0.01359	0.04331	0.00065	0.00072	0.48936	0.00010	0.17017	0.00006
0.1253	ISquared			0.92880	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	1.00000	0.45573
0.1254	DualU				1.00000	1.00000	0.70659	0.40467	0.01427	0.60988	0.00038	0.02961
0.1258	Uniform					1.00000	0.85263	1.00000	1.00000	1.00000	1.00000	0.09378
0.1263	Fibonacci						1.00000	0.68710	1.00000	1.00000	1.00000	0.01152
0.1274	Inverse							1.00000	1.00000	1.00000	1.00000	1.00000
0.1283	Macleod								1.00000	1.00000	1.00000	1.00000
0.1285	Zavrel									1.00000	0.30165	1.00000
0.1288	Rank										1.00000	1.00000
0.1294	NN											1.00000
0.1298	kNN											

**Table 14.** Bonferroni-adjusted  $p$  values for the pairwise Wilcoxon sign-rank test of the differences in average error rates across the datasets, in case of ERP

#### 4.5. Edit Distance on Real sequence (EDR)

Based on the numbers shown in Table 15 and Table 16 it is evident that this distance measure does not differ significantly from the previous ones. The number of datasets for which the various extensions of the 1NN classifier achieved better results is in range from 20 (kNN) to 36 (DualU). Following DualU, the second best results were obtained by the DualD and Dudani (33) schemes. The number of datasets with accuracies worse than with 1NN is not larger than 24. The largest number of statistically significant wins and the smallest number of losses are produced by the DualD weighting scheme, followed by the Dudani and the ISquared methods. Again, the simple 1NN rule has the worst ratio of wins and losses.

Similarly as in the case of the other distance measures, from the results of the adjusted sing-rank test in Table 17, we can see that the DualD weighting scheme produced low  $p$  values against most of

the other methods except Dudani, DualU and Zavrel. We can also notice that the Dudani scheme exhibited low  $p$  values against a number of weighting schemes with lower average error rates except the DualU and Zavrel methods and to a lesser extent the ISquared and Uniform schemes. In comparison with the ERP distance, according to this test the ISquared scheme performed a bit better. Based on these results we can highlight the DualD and the Dudani schemes as the best choices also with the EDR distance measure.

	kNN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
Better than 1NN	20	27	29	24	30	33	26	33	29	31	36
Average diff.	0.0248	0.0240	0.0234	0.0179	0.0105	0.0208	0.0217	0.0219	0.0182	0.0113	0.0159
Worse than 1NN	24	19	17	20	12	12	19	12	10	11	4
Average diff.	0.0083	0.0063	0.0044	0.0074	0.0048	0.0065	0.0069	0.0066	0.0022	0.0068	0.0009

**Table 15.** Comparison of kNN with 1NN in case of EDR

	DualD	Dudani	ISquared	DualU	Inverse	Zavrel	Uniform	Macleod	Fibonacci	Rank	KNN	NN
Wins	142	126	98	99	77	96	75	53	62	43	31	11
Losses	6	14	37	56	49	76	72	76	87	97	136	207
Wins-Losses	136	112	61	43	28	20	3	-23	-25	-54	-105	-196

**Table 16.** Statistically significant wins and losses counts for different NN classifiers, across all datasets, in case of EDR

	ERROR	0.1480	0.1488	0.1489	0.1497	0.1505	0.1510	0.1526	0.1556	0.1559	0.1560	0.1564	0.1620
ERROR		DualD	Dudani	ISquared	DualU	Inverse	Zavrel	Macleod	kNN	Rank	Uniform	Fibonacci	NN
0.1480	DualD		0.30385	0.02309	0.39242	0.00285	0.43467	0.00008	0.00000	0.00016	0.03207	0.00595	0.00103
0.1488	Dudani			0.39788	1.00000	0.03899	1.00000	0.00048	0.00001	0.00109	0.20613	0.09163	0.00225
0.1489	ISquared				1.00000	0.12900	1.00000	0.10232	0.00100	0.03629	1.00000	1.00000	0.06492
0.1497	DualU					1.00000	1.00000	0.10931	0.00143	0.04376	1.00000	0.43299	0.00000
0.1505	Inverse						1.00000	1.00000	0.00152	0.40647	1.00000	1.00000	0.28412
0.1510	Zavrel							0.62916	0.02226	0.18958	1.00000	1.00000	0.00197
0.1526	Macleod								0.00033	1.00000	1.00000	1.00000	0.48969
0.1556	kNN									0.00494	0.01100	0.30302	1.00000
0.1559	Rank										1.00000	1.00000	1.00000
0.1560	Uniform											1.00000	0.00969
0.1564	Fibonacci												0.03085
0.1620	NN												

**Table 17.** Bonferroni-adjusted  $p$  values for the pairwise Wilcoxon sign-rank test of the differences in average error rates across the datasets, in case of EDR

## 4.6 Summary

Observing the average classification accuracy, in the case of all of the considered distance measures, the best results are obtained with the dual distance-weighting scheme defined by Gou at al. [19] (Eq. (12)). The worst average classification accuracy is produced by the simple nearest neighbor rule (except for Euclidean distance and ERP). It is worth noting that the differences between the best and worst average results are not particularly big ( $<0.02$ ).

It is evident that there are some noticeable differences between the analyzed distance measures at the level of individual datasets. This is summarized in Table 18. In this table, the datasets for which

all of the observed extensions of the 1NN classifier (the unweighted  $k$ NN and all of its weighed versions) gave better classification accuracies are denoted with symbol ●. On the other hand, symbol ○ denotes the datasets for which none of the extensions has outperformed the simple nearest neighbor rule.

The obtained results show some resemblances between the distance measures, too. We can clearly identify two groups of datasets whose members exhibit similar characteristics. The first such group consists of the *haptics*, *italypowerdemand*, *noninvasivefatalecg\_thorax1*, *noninvasivefatalecg\_thorax2*, *uwavegesturelibrary\_x*, *uwavegesturelibrary\_y* and *uwavegesturelibrary\_z* datasets. For each dataset of this group, almost every distance measure gave better results with the unweighted and the weighted  $k$ NN than with the 1NN classifier. On the other hand, in case of the *cinc\_ecg\_torso* dataset every distance measure gave better (or equal) accuracies using the 1NN classifier compared to the unweighted and the weighted  $k$ NN (except for DTW).

Dataset	L2	DTW	LCS	ERP	EDR	Dataset	L2	DTW	LCS	ERP	EDR
50words		●				mallat		●	●		●
adiac	○	○				medicalimages	●		●		●
beef		○				motes	●	○			
car		○				noninvasivefatalecg_thorax1	●	●	●	●	●
cbf		○	●	○	●	noninvasivefatalecg_thorax2	●		●	●	●
chlorineconcentration	○	○	●	○	●	oliveoil				○	
cinc_ecg_torso	○		○	○	○	osuleaf					
coffee			○	○		plane		○			
cricket_x						sonyaiborobotsurface		○	○	○	
cricket_y						sonyaiborobotsurfaceii	○				
cricket_z						starlightcurves		●	●		●
diatomsizereduction	○	○	●	○	●	swedishleaf			●		
ecg200			○			symbols					
ecgfivedays		○	○		○	synthetic_control	○	●	●	●	●
faceall	○		●			trace	○	○		○	
facefour			○	○	○	twoleadecg		○	●	○	●
fish			●		●	twopatterns		○	○	○	●
gun_point			●	○	●	uwavegesturelibrary_x	●	●	●		●
haptics	●	●	●	●	●	uwavegesturelibrary_y	●	●	●	●	●
inlineskate			●			uwavegesturelibrary_z	●	●	●	●	●
italypowerdemand	●	●	●	●		wafer	○		○	○	
lighting2			○		○	wordssynonyms		●			○
lighting7		●	●	●		yoga				○	

**Table 18.** Comparison of different distance measures at the level of individual datasets

In order to better understand the importance of the first neighbor, we have counted how many times was  $k = 1$  selected as the optimal value in the process of tuning parameter  $k$  over all datasets. For every examined variation of the  $k$ NN classifier, the optimal value of parameter  $k$  was selected 4600 times (100 times by the SC1x9 algorithm for each of the 46 examined datasets). The obtained results in percentages are shown in Table 19. We can see that  $k = 1$  represents the best solution very often, especially in case of the majority voting  $k$ -nearest neighbor rule. Optimal solutions

requiring a larger number of neighbors most frequently occur in case of the DualD, Dudani and the DualU weighting schemes. We can also notice that  $k = 1$  was much less frequently chosen in case of LCS and EDR than in case of the other distance measures.

	kNN	Inverse	ISquared	Rank	Fibonacci	Dudani	Macleod	DualD	Zavrel	Uniform	DualU
<b>L2</b>	<b>61.07</b>	52.48	46.50	56.63	51.93	24.89	54.54	23.57	40.43	44.28	27.24
<b>DTW</b>	<b>60.22</b>	44.09	38.37	55.07	51.65	33.35	52.65	31.52	41.87	42.91	30.17
<b>LCS</b>	<b>35.28</b>	25.33	22.67	30.52	29.83	18.20	27.17	17.85	28.96	24.48	17.24
<b>ERP</b>	<b>56.63</b>	49.43	44.59	52.04	48.33	30.96	50.43	30.37	48.48	41.50	30.15
<b>EDR</b>	<b>36.07</b>	25.83	21.67	31.72	29.46	14.22	27.50	13.46	28.85	21.83	14.13

**Table 19.** Percentage of SCV10x10 folds for which  $k = 1$  was selected as the optimal value over all datasets.

Through a series of detailed experiments presented in this work we have confirmed the view that the simple 1NN is very hard to beat [63]. When observing the number of statistically significant wins and losses, the best results are achieved by the distance-weighted scheme defined by Dudani [11] (Eq. (3)) in case of Euclidean distance, and the dual distance-weighting scheme defined by Gou et al. [19] (Eq. (12)) in case of the unconstrained elastic distance measures. Both the corrected resampled t-test and the adjusted Wilcoxon sing-rank test results support the DualD and the Dudani weighting methods as the best choices in combination with all of the five discussed time-series distance measures.

## 5. Conclusions and future work

In the last decade classification has been intensively investigated in the field of time-series data mining [18, 24, 48, 63, 64]. Among the considerable number of proposed techniques, the simple nearest neighbor classifier and the Dynamic Time Warping distance measure were shown to be one of the best combinations [63]. To improve the accuracy of classification, the majority voting  $k$ -nearest neighbor rule generalizes the idea of the 1NN rule by taking into account not one but  $k$  nearest neighbors. The next step in investigating the nearest neighbor rule is assigning different weights to the neighbors.

Several different weighting schemes are proposed in the literature [11, 19, 20, 35, 37, 39, 44, 65]. They have been examined exclusively in combination with the Euclidean distance and they were either not tested in the domain of time series, or tested only in a very limited extent. Furthermore, the results were not supported by statistical tests. In this paper, we have, through a detailed analysis, compared a wide variety of nearest-neighbor weighting schemes in combination with the five most commonly used time-series distance measures based on the largest set of freely available labeled time-series datasets [25]. Based on the obtained results we can conclude that, for a significant number of datasets, both the majority voting  $k$ NN and the distance distance-weighted  $k$ -nearest neighbor rule can produce better accuracies than the 1NN classifier. On the other hand, for a notable number of datasets they cannot outperform this simple classifier. In both cases the differences are not overwhelming. Among the analyzed schemes the best performances were obtained with the dual distance-weighting scheme defined by Gou et al. [19] (Eq. (12)) and with the weighting function defined by Dudani [11] (Eq. (3)). These findings were also confirmed by the

corrected resampled t-test and the Wilcoxon sing-rank test adjusted by the one-step Bonferroni scheme.

Since the elastic measures (DTW, LCS, ERP and EDR) generally provide more accurate classification accuracies compared to non-elastic measures, it would be interesting to check the influence of different weighting schemes on constrained versions of these measures. The major drawback of these measures is performance, since they are all based on quadratic complexity algorithms. The introduction of global constraints significantly speeds-up the computation process [31] and in some cases (DTW) even improves the classification accuracies [32]. Furthermore, due to the high dimensionality of time-series data, it would be interesting to investigate the interaction of the hubness phenomenon [48] with different *k*NN weighting methods and the behavior of the hubs-based weighting scheme [47].

## Acknowledgments

The authors would like to thank Eamonn Keogh for collecting and making available the UCR time series datasets, as well as everyone who contributed data to the collection, without whom the presented work would not have been possible. V. Kurbalija, M. Radovanović and M. Ivanović thank the Serbian Ministry of Education, Science and Technological Development for support through project no. OI174023, “Intelligent Techniques and their Integration into Wide-Spectrum Decision Support.”

## References

1. Agrawal, R. et al.: Efficient similarity search in sequence databases. Proceedings of the 4th International Conference on Foundations of Data Organization and Algorithms. pp. 69–84 Springer Berlin Heidelberg (1993).
2. Bache, K., Lichman, M.: UCI Machine Learning Repository, <http://archive.ics.uci.edu/ml>, (2013).
3. Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. KDD workshop. pp. 359–370 Seattle, WA (1994).
4. Bouckaert, R., Frank, E.: Evaluating the Replicability of Significance Tests for Comparing Learning Algorithms. In: Dai, H. et al. (eds.) Advances in Knowledge Discovery and Data Mining SE - 3. pp. 3–12 Springer Berlin Heidelberg (2004).
5. Brockwell, P.J., Davis, R.A.: Introduction to Time Series and Forecasting. Springer (2002).
6. Chen, L. et al.: Robust and fast similarity search for moving object trajectories. Proceedings of the 2005 ACM SIGMOD international conference on Management of data. pp. 491–502 ACM, New York, NY, USA (2005).
7. Chen, L., Ng, R.: On The Marriage of Lp-norms and Edit Distance. Proceedings of the Thirtieth international conference on Very large data bases. pp. 792–803 VLDB Endowment (2004).

8. Cover, T., Hart, P.: Nearest neighbor pattern classification. *Inf. Theory, IEEE Trans.* 13, 1, 21–27 (1967).
9. Das, G., Gunopulos, D.: Time Series Similarity and Indexing. In: Ye, N. (ed.) *The Handbook of Data Mining*. pp. 279–304 Lawrence Erlbaum Associates (2003).
10. Ding, H. et al.: Querying and mining of time series data: experimental comparison of representations and distance measures. *Proc. VLDB Endow.* 1, 2, 1542–1552 (2008).
11. Dudani, S.A.: The Distance-Weighted k-Nearest-Neighbor Rule. *Syst. Man Cybern. IEEE Trans.* 6, 4, 325–327 (1976).
12. Esling, P., Agon, C.: Time-series Data Mining. *ACM Comput. Surv.* 45, 1, 12:1–12:34 (2012).
13. Faloutsos, C. et al.: Fast subsequence matching in time-series databases. *ACM SIGMOD Rec.* 23, 2, 419–429 (1994).
14. Fix, E., Hodges, J.L.: Discriminatory Analysis - Nonparametric Discrimination: Consistency Properties, (1951).
15. García, S. et al.: A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability. *Soft Comput.* 13, 10, 959–977 (2009).
16. García, S., Herrera, F.: An Extension on “Statistical Comparisons of Classifiers over Multiple Data Sets” for all Pairwise Comparisons. *J. Mach. Learn. Res.* 9, 2677–2694 (2008).
17. Goldin, D.Q., Kanellakis, P.C.: On similarity queries for time-series data: Constraint specification and implementation. In: Montanari, U. and Rossi, F. (eds.) *Principles and Practice of Constraint Programming — CP ’95 SE - 9*. pp. 137–153 Springer Berlin Heidelberg (1995).
18. Górecki, T., Luczak, M.: Using derivatives in time series classification. *Data Min. Knowl. Discov.* 26, 2, 310–331 (2013).
19. Gou, J. et al.: A New distance-weighted k-nearest neighbor classifier. *J. Inf. Comput. Sci.* 9, 6, 1429–1436 (2012).
20. Gou, J. et al.: A Novel Weighted Voting for K-Nearest Neighbor Rule. *J. Comput.* 6, 5, 833–840 (2011).
21. Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2006).
22. Hand, D.J. et al.: *Principles of Data Mining*. MIT Press (2001).
23. Itakura, F.: Minimum prediction residual principle applied to speech recognition. *Acoust. Speech Signal Process. IEEE Trans.* 23, 1, 67–72 (1975).
24. Jeong, Y.-S. et al.: Weighted dynamic time warping for time series classification. *Pattern Recognit.* 44, 9, 2231–2240 (2011).

25. Keogh, E. et al.: The UCR Time Series Classification/Clustering Homepage: [www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/), citeulike-article-id:2139261, (2011).
26. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. *Knowl. Inf. Syst.* 7, 3, 358–386 (2005).
27. Kotsiantis, S.B.: Supervised Machine Learning: A Review of Classification Techniques. *Proceedings of the 2007 Conference on Emerging Artificial Intelligence Applications in Computer Engineering: Real Word AI Systems with Applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*. pp. 3–24 IOS Press, Amsterdam, The Netherlands, The Netherlands (2007).
28. Kurbalija, V. et al.: A Framework for Time-Series Analysis. In: Dicheva, D. and Dochev, D. (eds.) *Artificial Intelligence: Methodology, Systems, and Applications SE - 5*. pp. 42–51 Springer Berlin Heidelberg (2010).
29. Kurbalija, V. et al.: Case-based curve behaviour prediction. *Softw. Pract. Exp.* 39, 1, 81–103 (2009).
30. Kurbalija, V. et al.: Matching Observed with Empirical Reality—What you see is what you get? *Fundam. Informaticae*. 129, 1, 133–147 (2014).
31. Kurbalija, V. et al.: The Influence of Global Constraints on DTW and LCS Similarity Measures for Time-Series Databases. In: Dicheva, D. et al. (eds.) *Third International Conference on Software, Services and Semantic Technologies S3T 2011 SE - 10*. pp. 67–74 Springer Berlin Heidelberg (2011).
32. Kurbalija, V. et al.: The influence of global constraints on similarity measures for time-series databases. *Knowledge-Based Syst.* 56, 49–67 (2014).
33. Kurbalija, V. et al.: Time-Series Analysis in the Medical Domain: A Study of Tacrolimus Administration and Influence on Kidney Graft Function. *Comput. Biol. Med.* 50, 19–31 (2014).
34. Kurbalija, V. et al.: Time-series Mining in a Psychological Domain. *Proceedings of the Fifth Balkan Conference in Informatics*. pp. 58–63 ACM, New York, NY, USA (2012).
35. Larose, D.T.: *Discovering Knowledge in Data: An Introduction to Data Mining*. Wiley (2005).
36. Laxman, S., Sastry, P.S.: A survey of temporal data mining. *Sadhana*. 31, 2, 173–198 (2006).
37. Macleod, J.E.S. et al.: A Re-Examination of the Distance-Weighted k-Nearest Neighbor Classification Rule. *Syst. Man Cybern. IEEE Trans.* 17, 4, 689–696 (1987).
38. Marteau, P.-F.: Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching. *Pattern Anal. Mach. Intell. IEEE Trans.* 31, 2, 306–318 (2009).
39. Mitchell, T.M.: *Machine Learning*. McGraw-Hill, Inc., New York, NY, USA (1997).
40. Mitrovic, D. et al.: Agent-Based Distributed Computing for Dynamic Networks. *Inf. Technol. Control*. 43, 1, 88–97 (2014).

41. Mitrović, D. et al.: Distributed Distance Matrix Generator Based on Agents. Proceedings of the 2Nd International Conference on Web Intelligence, Mining and Semantics. pp. 40:1–40:6 ACM, New York, NY, USA (2012).
42. Morse, M.D., Patel, J.M.: An efficient and accurate method for evaluating time series similarity. Proceedings of the 2007 ACM SIGMOD international conference on Management of data. pp. 569–580 ACM, New York, NY, USA (2007).
43. Nanopoulos, A. et al.: Feature-based Classification of Time-series Data. *Int. J. Comput. Res.* 10, 49–61 (2001).
44. Pao, T.-L. et al.: A Comparative Study of Different Weighting Schemes on KNN-Based Emotion Recognition in Mandarin Speech. In: Huang, D.-S. et al. (eds.) *Advanced Intelligent Computing Theories and Applications. With Aspects of Theoretical and Methodological Issues SE* - 101. pp. 997–1005 Springer Berlin Heidelberg (2007).
45. Pao, T.-L. et al.: Emotion Recognition and Evaluation of Mandarin Speech Using Weighted D-KNN Classification. Proceedings of the 17th Conference on Computational Linguistics and Speech Processing, ROCLING. Association for Computational Linguistics and Chinese Language Processing (ACLCLP), Taiwan (2005).
46. Pavlovic, V. et al.: Time-series classification using mixed-state dynamic Bayesian networks. *Computer Vision and Pattern Recognition*, 1999. IEEE Computer Society Conference on. pp. 609–615 (1999).
47. Radovanović, M. et al.: Hubs in Space: Popular Nearest Neighbors in High-Dimensional Data. *J. Mach. Learn. Res.* 11, 2487–2531 (2010).
48. Radovanović, M. et al.: Time-series classification in many intrinsic dimensions. *Proc. 10th SIAM Int. Conf. on Data Mining (SDM)*. pp. 677–688 (2010).
49. Ratanamahatana, C. et al.: Mining Time Series Data. In: Maimon, O. and Rokach, L. (eds.) *Data Mining and Knowledge Discovery Handbook SE* - 51. pp. 1069–1103 Springer US (2005).
50. Ratanamahatana, C.A., Keogh, E.: Three myths about dynamic time warping data mining. *Proceedings of SIAM International Conference on Data Mining (SDM'05)*. pp. 506–510 (2005).
51. Rodríguez, J. et al.: Learning First Order Logic Time Series Classifiers: Rules and Boosting. In: Zighed, D. et al. (eds.) *Principles of Data Mining and Knowledge Discovery SE* - 29. pp. 299–308 Springer Berlin Heidelberg (2000).
52. Rodríguez, J.J., Alonso, C.J.: Interval and Dynamic Time Warping-based Decision Trees. *Proceedings of the 2004 ACM Symposium on Applied Computing*. pp. 548–552 ACM, New York, NY, USA (2004).
53. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. *Acoust. Speech Signal Process. IEEE Trans.* 26, 1, 43–49 (1978).
54. Shi, T. et al.: Application of grid-based k-means clustering algorithm for optimal image processing. *Comput. Sci. Inf. Syst.* 9, 4, 1679–1696 (2012).



55. Skopal, T., Bustos, B.: On Nonmetric Similarity Search Problems in Complex Domains. *ACM Comput. Surv.* 43, 4, 34:1–34:50 (2011).
56. Stojanovic, R. et al.: Optimization and implementation of the wavelet based algorithms for embedded biomedical signal processing. *Comput. Sci. Inf. Syst.* 10, 1, 502–523 (2013).
57. Takigawa, Y. et al.: Pattern Classification Using Weighted Average Patterns of Categorical k-Nearest Neighbors. *Proc. of the 1th International Workshop on Camera-Based Document Analysis and Recognition*. pp. 111–118 (2005).
58. Tomašev, N., Mladenović, D.: Nearest Neighbor Voting in High Dimensional Data: Learning from Past Occurrences. *Comput. Sci. Inf. Syst.* 9, 2, 691–712 (2012).
59. Vlachos, M. et al.: Discovering similar multidimensional trajectories. *Proceedings 18th International Conference on Data Engineering*. pp. 673–684 *IEEE Comput. Soc* (2002).
60. Wu, X. et al.: Top 10 algorithms in data mining. *Knowl. Inf. Syst.* 14, 1, 1–37 (2008).
61. Wu, Y., Chang, E.Y.: Distance-function Design and Fusion for Sequence Data. *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*. pp. 324–333 *ACM, New York, NY, USA* (2004).
62. Wu, Y.-L. et al.: A Comparison of DFT and DWT Based Similarity Search in Time-series Databases. *Proceedings of the Ninth International Conference on Information and Knowledge Management*. pp. 488–495 *ACM, New York, NY, USA* (2000).
63. Xi, X. et al.: Fast time series classification using numerosity reduction. *Proceedings of the 23rd international conference on Machine learning*. pp. 1033–1040 *ACM, New York, NY, USA* (2006).
64. Ye, L., Keogh, E.: Time Series Shapelets: A New Primitive for Data Mining. *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. pp. 947–956 *ACM, New York, NY, USA* (2009).
65. Zavrel, J.: An Empirical Re-Examination of Weighted Voting for k-NN. *Proceedings of the 7th Belgian-Dutch Conference on Machine Learning*. pp. 139–148 (1997).
66. Zhang, H. et al.: A Non-parametric Wavelet Feature Extractor for Time Series Classification. In: Dai, H. et al. (eds.) *Advances in Knowledge Discovery and Data Mining SE - 71*. pp. 595–603 *Springer Berlin Heidelberg* (2004).

## Appendix

Tables 0, 1, 2, 3 and 4 contain the classification errors and the values of parameter  $k$  obtained for the analyzed similarity measures (Euclidean distance, DTW, LCS, ERP and EDR). Due to lack of space, the values reported in the tables in the Appendix are shown rounded to three decimal places.

ID	1NN error	kNN error	$k$	Inverse error	$k$	ISquared error	$k$	Rank error	$k$	Fibonacci error	$k$	Dudani error	$k$	Macleod error	$k$	DualD error	$k$	Zavrel error	$k$	Uniform error	$k$	DualU error	$k$
1	0.298	0.301	1.40	0.301	1.66	0.300	2.93	0.302	1.87	0.298	4.84	0.293	6.71	0.302	1.63	<b>0.292</b>	<b>7.24</b>	0.299	4.76	0.299	3.68	0.299	10.36
2	<b>0.315</b>	0.320	1.52	0.323	1.80	0.326	2.43	0.322	2.11	0.321	5.17	0.317	5.73	0.320	1.56	0.316	6.40	0.322	1.66	0.320	6.29	0.317	14.79
3	0.487	0.540	8.06	0.522	9.94	0.495	7.71	0.542	8.49	0.485	4.48	0.558	9.54	0.557	7.16	0.543	9.36	0.555	7.25	<b>0.483</b>	<b>10.38</b>	0.492	12.43
4	0.222	0.229	3.07	0.225	3.25	0.222	3.48	0.228	4.00	0.216	4.76	<b>0.204</b>	<b>6.13</b>	0.225	3.43	0.205	6.16	0.205	4.08	0.215	7.07	0.212	11.18
5	0.011	0.008	11.44	0.007	11.71	0.007	12.90	0.006	16.29	0.012	4.93	0.006	15.79	0.007	12.12	0.006	15.89	<b>0.006</b>	<b>13.54</b>	0.010	13.32	0.010	17.21
6	<b>0.002</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>2.11</b>
7	<b>0.001</b>	0.001	1.02	0.001	1.07	0.002	1.18	0.001	1.23	0.001	1.06	0.002	2.73	0.002	1.22	0.002	2.75	<b>0.001</b>	<b>1.04</b>	0.001	1.29	<b>0.001</b>	<b>1.98</b>
8	0.100	0.108	1.12	0.115	1.52	0.127	2.01	0.114	1.24	0.113	1.81	0.114	2.11	0.119	1.40	0.114	2.52	0.102	2.44	0.105	3.91	<b>0.099</b>	<b>6.41</b>
9	0.331	0.331	1.00	0.331	1.00	0.332	1.08	0.331	1.00	0.331	1.12	0.330	3.51	0.331	1.00	0.332	3.97	0.331	1.12	0.331	1.12	<b>0.328</b>	<b>13.32</b>
10	0.344	0.347	2.00	0.346	3.08	0.343	3.93	0.350	3.20	0.341	7.10	0.344	6.31	0.350	2.66	0.343	7.18	0.344	1.20	<b>0.338</b>	<b>8.10</b>	<b>0.338</b>	<b>18.17</b>
11	0.331	0.331	1.00	0.331	1.00	0.332	1.06	0.331	1.00	0.331	1.06	0.331	3.38	0.331	1.00	0.332	3.44	0.331	1.38	0.331	1.10	<b>0.328</b>	<b>14.61</b>
12	<b>0.001</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>
13	0.100	0.103	2.78	0.096	4.08	0.089	4.11	0.105	3.85	0.096	3.91	0.092	5.51	0.099	3.89	0.089	5.63	<b>0.085</b>	<b>4.18</b>	0.092	5.57	0.093	13.94
14	0.005	0.006	1.02	0.006	1.08	0.006	2.03	0.006	1.07	0.006	1.06	<b>0.005</b>	<b>2.92</b>	0.006	1.05	0.005	3.01	0.007	2.90	0.006	3.42	0.006	5.79
15	<b>0.040</b>	<b>0.040</b>	<b>1.00</b>	<b>0.040</b>	<b>1.00</b>	<b>0.040</b>	<b>1.00</b>	<b>0.040</b>	<b>1.00</b>	<b>0.040</b>	<b>1.00</b>	0.041	2.37	<b>0.040</b>	<b>1.00</b>	0.041	2.53	0.041	1.19	<b>0.040</b>	<b>1.00</b>	0.040	7.88
16	0.057	0.066	1.33	0.066	1.38	0.071	2.13	0.065	1.45	0.064	1.85	0.071	3.32	0.068	1.38	0.070	3.32	<b>0.055</b>	<b>3.33</b>	0.065	4.31	0.059	6.41
17	0.168	0.173	3.01	0.167	3.86	0.165	4.28	0.167	5.24	<b>0.153</b>	<b>6.35</b>	0.162	8.83	0.167	3.96	0.162	9.35	0.170	4.35	0.156	6.71	0.160	17.80
18	0.053	0.058	1.42	0.058	2.27	0.062	2.91	0.057	2.44	0.057	2.77	0.052	4.48	0.058	2.29	<b>0.050</b>	<b>4.50</b>	0.058	2.19	0.058	4.33	0.054	4.01
19	0.590	0.531	17.25	0.527	17.58	<b>0.522</b>	<b>18.73</b>	0.523	21.18	0.558	9.05	0.529	24.92	0.523	17.94	0.528	24.61	0.544	17.84	0.529	24.40	0.565	26.52
20	0.478	0.478	1.06	0.481	1.47	0.482	2.06	0.478	1.09	0.482	2.25	0.476	4.36	0.478	1.18	0.475	4.56	<b>0.474</b>	<b>6.62</b>	0.482	2.58	0.479	9.94
21	0.033	0.031	8.36	0.029	8.34	0.029	9.85	<b>0.028</b>	<b>10.29</b>	0.032	6.55	0.029	11.85	0.029	8.55	0.029	12.42	0.030	8.49	0.029	13.93	0.031	21.48
22	0.258	0.273	2.18	0.269	2.66	0.273	2.96	0.273	3.05	0.260	3.73	<b>0.241</b>	<b>5.84</b>	0.268	2.98	0.249	6.36	0.255	7.38	0.259	5.24	0.258	13.14
23	0.354	0.362	2.75	0.356	3.10	0.354	3.28	0.360	3.63	0.365	4.96	0.361	5.46	<b>0.352</b>	<b>3.06</b>	0.363	5.36	0.365	1.81	0.368	3.76	0.354	10.63
24	0.016	0.016	1.10	0.016	1.39	0.016	1.64	0.016	1.32	0.016	1.57	<b>0.015</b>	<b>5.04</b>	0.016	1.38	0.015	5.73	0.017	2.40	0.016	2.84	<b>0.015</b>	<b>14.71</b>
25	0.228	0.228	2.78	0.222	3.70	0.223	4.84	0.225	4.78	0.225	7.93	<b>0.211</b>	<b>8.50</b>	0.224	3.78	0.212	9.32	0.222	6.80	0.224	15.34	0.225	22.67
26	0.079	0.064	13.90	0.058	14.60	0.057	16.80	0.058	20.23	0.074	9.16	0.058	27.90	0.059	15.22	<b>0.057</b>	<b>28.47</b>	0.061	24.58	0.060	27.09	0.077	21.68
27	0.157	0.156	4.28	0.153	5.52	0.149	6.61	0.152	7.06	0.150	9.52	0.147	10.30	0.152	5.67	0.147	10.80	<b>0.145</b>	<b>15.87</b>	0.147	15.25	0.148	27.47
28	0.095	0.092	4.36	0.089	4.69	0.086	6.35	0.089	6.28	0.088	9.29	0.086	9.86	0.088	4.79	<b>0.086</b>	<b>10.89</b>	0.089	11.73	0.086	15.83	0.088	25.87
29	0.117	<b>0.112</b>	<b>2.02</b>	0.117	2.44	0.118	2.64	0.117	2.91	0.117	2.73	0.115	3.76	0.118	2.36	0.115	3.89	0.115	2.24	<b>0.112</b>	<b>5.84</b>	0.115	10.95
30	0.342	0.347	1.29	0.350	1.51	0.352	2.38	0.350	1.71	0.349	2.59	0.345	4.74	0.350	1.59	<b>0.339</b>	<b>5.11</b>	0.344	8.57	0.349	5.02	0.340	14.13
31	0.028	0.032	4.28	0.031	7.45	<b>0.026</b>	<b>9.39</b>	0.031	5.84	0.029	1.39	0.031	12.00	0.030	5.12	0.032	11.74	0.029	8.98	0.032	5.48	0.028	5.14
32	0.014	0.015	1.38	0.016	1.76	0.015	1.93	0.015	1.88	0.015	1.69	<b>0.014</b>	<b>3.84</b>	0.016	1.86	0.014	3.96	0.015	2.59	0.015	2.02	0.014	4.07
33	<b>0.014</b>	0.015	1.20	0.016	1.30	0.016	1.66	0.015	1.39	0.015	1.39	0.016	3.67	0.016	1.30	0.015	3.73	0.014	2.35	0.016	1.57	0.014	11.42
34	0.111	0.112	1.60	0.110	3.48	0.110	3.41	0.112	3.66	0.110	4.39	<b>0.108</b>	<b>5.54</b>	0.110	3.48	0.109	5.95	0.110	3.47	0.108	5.02	0.109	22.38
35	0.177	0.177	1.00	0.177	1.00	0.177	1.00	0.177	1.00	0.177	1.09	<b>0.174</b>	<b>4.08</b>	0.177	1.00	0.174	4.41	0.177	1.00	0.177	1.00	0.179	13.13
36	0.035	0.039	2.14	0.038	3.76	0.037	4.00	0.038	3.52	0.038	2.39	0.034	6.47	0.038	3.47	0.034	7.03	0.035	2.70	0.039	4.87	<b>0.031</b>	<b>18.27</b>
37	<b>0.077</b>	0.079	1.34	0.079	1.34	0.079	1.34	0.079	1.57	0.079	1.58	0.081	2.83	<b>0.079</b>	<b>1.34</b>	0.082	2.99	0.079	1.40	0.079	1.71	0.080	10.88
38	<b>0.119</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.00</b>	0.120	1.24	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.39</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.00</b>	<b>0.119</b>	<b>1.15</b>
39	0.004	0.005	3.02	0.005	4.30	0.005	5.61	0.005	5.61	0.004	3.76	0.005	7.33	0.005	4.16	0.005	7.80	0.005	4.08	<b>0.004</b>	<b>4.47</b>	0.004	9.01
40	0.013	0.013	1.08	0.013	1.08	0.013	1.13	0.013	1.12	0.012	3.29	0.011	6.12	0.013	1.08	0.011	6.33	<b>0.011</b>	<b>4.05</b>	0.012	3.41	0.012	18.30
41	0.228	0.222	3.02	0.221	3.40	0.222	3.74	0.222	4.72	0.220	9.35	<b>0.218</b>	<b>6.39</b>	0.222	3.49	0.219	6.92	0.227	5.75	0.222	6.01	0.219	19.32
42	0.275	0.270	4.17	0.266	6.11	0.263	8.74	0.266	7.00	0.263	8.35	0.261	9.81	0.268	5.07	0.261	10.79	0.273	12.95	0.258	16.08	<b>0.257</b>	<b>28.98</b>
43	0.288	0.286	3.24	0.285	4.89	0.283	5.99	0.285	5.78	0.281	10.61	0.278	8.44	0.285	4.95	<b>0.277</b>	<b>8.74</b>	0.286	7.84	0.281	8.66	0.279	24.49
44	<b>0.001</b>	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	0.001	1.08	<b>0.001</b>	<b>1.00</b>	<b>0.001</b>	<b>1.00</b>	0.001	1.19	<b>0.001</b>	<b>1.00</b>	0.001	1.19	0.001	1.12	0.001	1.24	<b>0.001</b>	<b>1.21</b>
45	0.284	0.288	1.46	0.288	1.56	0.289	2.35	0.289	1.78	0.286	4.07	0.279	5.96	0.289	1.60	<b>0.278</b>	<b>6.19</b>	0.286	3.69	0.286	3.03	0.287	10.01
46	0.057	0.057	1.00	0.057	1.06	0.058	1.20	0.057	1.00	0.057	1.09	0.057	3.64	0.057	1.06	0.057	4.08	0.058	1.53	0.058	1.20	<b>0.056</b>	<b>11.92</b>

**Table 20.** Classification errors and the values of parameter  $k$  obtained for Euclidean distance

ID	1NN		kNN		Inverse		ISquared		Rank		Fibonacci		Dudani		Macleod		DualD		Zavrel		Uniform		DualU	
	error		error	k	error	k	error	k	error	k	error	k	error	k	error	k	error	k	error	k	error	k	error	k
1	0.279		0.266	3.00	0.261	3.95	0.264	5.18	0.267	4.31	0.263	8.74	0.262	6.13	0.267	3.27	<b>0.260</b>	<b>7.34</b>	0.268	6.17	0.266	6.92	0.265	21.77
2	<b>0.334</b>		0.337	1.33	0.343	2.12	0.343	2.20	0.342	2.47	0.337	4.30	0.338	6.31	0.341	1.81	0.336	6.82	0.339	1.53	0.341	5.39	0.335	14.88
3	<b>0.468</b>		0.542	5.42	0.513	7.90	0.512	4.58	0.542	5.78	0.485	3.13	0.543	6.44	0.542	5.34	0.540	7.18	0.545	5.46	0.470	10.08	0.475	11.59
4	<b>0.245</b>		0.253	1.27	0.269	3.01	0.261	5.53	0.263	1.93	0.259	2.98	0.258	4.29	0.265	2.06	0.256	5.02	0.264	2.48	0.260	4.09	0.251	9.62
5	<b>0.000</b>		<b>0.000</b>	<b>1.14</b>	<b>0.000</b>	<b>1.14</b>	<b>0.000</b>	<b>1.14</b>	<b>0.000</b>	<b>1.21</b>	<b>0.000</b>	<b>1.21</b>	<b>0.000</b>	<b>1.21</b>	<b>0.000</b>	<b>1.14</b>	<b>0.000</b>	<b>1.21</b>	<b>0.000</b>	<b>1.14</b>	<b>0.000</b>	<b>1.21</b>	<b>0.000</b>	<b>1.57</b>
6	<b>0.003</b>		<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.48</b>
7	<b>0.015</b>		<b>0.015</b>	<b>1.00</b>	<b>0.015</b>	<b>1.00</b>	0.015	1.34	<b>0.015</b>	<b>1.00</b>	<b>0.015</b>	<b>1.00</b>	0.016	1.89	<b>0.015</b>	<b>1.00</b>	0.015	2.43	0.015	2.56	<b>0.015</b>	<b>1.00</b>	0.015	2.73
8	<b>0.074</b>		0.076	1.02	0.087	1.24	0.088	1.68	0.085	1.15	0.085	1.24	0.082	1.16	0.083	1.11	0.087	1.35	<b>0.074</b>	<b>1.00</b>	0.087	1.89	0.076	1.70
9	0.177		0.181	1.55	0.181	1.76	0.181	2.65	0.180	1.80	0.179	4.05	0.177	4.12	0.181	1.66	0.174	4.49	<b>0.173</b>	<b>3.87</b>	0.178	6.91	0.175	15.99
10	0.170		0.172	1.43	0.164	3.08	0.167	6.59	0.171	2.43	0.170	4.98	0.163	6.05	0.172	2.06	<b>0.162</b>	<b>6.98</b>	0.168	5.94	0.174	10.08	0.169	16.77
11	0.173		0.175	1.28	0.176	1.60	0.179	3.28	0.176	1.65	0.177	2.77	0.175	4.05	0.177	1.40	0.173	4.97	<b>0.173</b>	<b>4.84</b>	0.180	4.59	0.176	7.70
12	<b>0.004</b>		<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>	<b>0.004</b>	<b>1.00</b>
13	0.170		0.183	3.04	0.165	4.02	0.157	4.99	0.183	4.07	0.176	3.37	0.156	5.30	0.182	3.86	<b>0.155</b>	<b>5.93</b>	0.164	4.80	0.174	3.86	0.168	12.70
14	<b>0.008</b>		0.009	1.28	0.010	2.01	0.010	2.20	0.009	1.61	0.009	1.87	0.009	2.81	0.009	1.68	0.009	2.82	0.008	1.24	0.009	2.23	0.009	7.02
15	0.022		0.022	1.00	0.023	1.19	0.024	1.85	0.022	1.04	0.022	1.37	0.021	4.07	0.022	1.06	0.021	4.60	<b>0.020</b>	<b>4.48</b>	0.022	1.47	0.021	15.33
16	0.055		0.073	3.05	0.072	3.62	0.071	3.65	0.070	4.28	0.064	1.91	0.070	5.85	0.070	3.58	0.071	7.18	0.055	1.00	<b>0.050</b>	<b>11.08</b>	0.057	12.51
17	0.198		0.205	3.01	0.188	3.82	<b>0.185</b>	<b>4.28</b>	0.204	4.82	0.202	4.48	0.193	6.07	0.191	3.54	0.190	6.98	0.189	3.59	0.201	9.58	0.196	13.46
18	0.085		0.092	2.30	0.092	4.67	<b>0.080</b>	<b>5.89</b>	0.090	4.67	0.091	2.41	0.084	6.43	0.091	3.91	0.085	7.01	0.094	3.92	0.091	7.75	0.088	9.96
19	0.584		0.534	16.72	0.532	16.63	0.535	16.18	0.530	18.61	0.552	9.88	0.526	17.92	0.533	16.64	<b>0.524</b>	<b>19.29</b>	0.538	20.64	0.531	22.63	0.561	26.71
20	0.453		0.455	2.46	<b>0.419</b>	<b>13.60</b>	0.428	15.81	0.466	6.60	0.459	5.72	0.460	10.29	0.469	4.92	0.449	12.35	0.447	9.16	0.457	11.23	0.455	20.14
21	0.046		0.040	5.58	0.039	6.17	<b>0.036</b>	<b>7.11</b>	0.037	8.28	0.038	6.28	0.039	12.15	0.038	6.68	0.038	12.54	0.040	6.27	0.039	13.28	0.039	23.92
22	0.106		0.110	1.20	0.116	1.65	0.115	1.59	0.111	1.43	0.113	1.78	0.114	2.68	0.118	1.63	0.111	2.76	0.106	1.90	0.116	2.66	<b>0.106</b>	<b>6.98</b>
23	0.278		0.243	4.58	0.239	5.61	0.257	7.56	0.250	6.59	0.238	6.80	0.232	9.19	0.242	5.55	<b>0.226</b>	<b>10.44</b>	0.267	4.10	0.228	13.57	0.253	21.96
24	0.013		0.013	2.66	0.011	3.80	0.011	4.66	0.012	4.75	0.012	5.53	0.011	7.79	0.011	3.94	0.011	8.39	0.011	4.04	0.012	8.09	<b>0.010</b>	<b>23.15</b>
25	0.198		0.207	2.72	0.190	8.33	0.187	12.48	0.199	6.04	0.195	7.68	0.195	9.30	0.199	5.08	0.193	11.98	0.187	18.02	<b>0.186</b>	<b>20.70</b>	0.192	25.97
26	<b>0.046</b>		0.046	1.02	0.047	1.70	0.049	4.91	0.046	1.20	0.047	1.87	0.047	2.23	0.046	1.22	0.047	3.02	0.047	3.55	0.048	3.29	0.047	9.07
27	0.192		0.179	5.32	0.177	5.93	0.177	6.85	0.176	8.83	0.177	10.49	0.171	12.35	0.175	6.67	<b>0.170</b>	<b>14.18</b>	0.175	6.34	0.175	18.21	0.182	28.84
28	0.121		0.123	4.37	0.119	6.75	0.116	10.03	0.117	8.55	0.115	8.67	0.113	11.52	0.117	7.02	<b>0.113</b>	<b>12.84</b>	0.118	6.20	0.114	21.28	0.116	25.95
29	0.117		0.118	1.94	0.118	2.17	0.120	1.99	0.122	2.56	0.105	3.28	0.123	3.60	0.120	2.16	0.123	3.60	0.120	1.99	<b>0.103</b>	<b>4.27</b>	0.118	10.11
30	0.291		0.293	1.13	0.307	4.32	0.293	6.58	0.303	2.52	0.296	3.54	0.301	6.46	0.303	2.22	0.297	7.21	<b>0.288</b>	<b>5.96</b>	0.297	7.08	0.294	13.61
31	<b>0.000</b>		<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>
32	<b>0.025</b>		0.026	1.08	0.026	1.11	0.026	1.11	0.026	1.16	0.026	1.15	0.027	2.79	0.026	1.22	0.027	2.77	0.027	1.38	0.026	1.30	<b>0.025</b>	<b>3.23</b>
33	0.027		0.029	1.28	0.029	2.01	0.029	2.16	0.029	1.61	0.029	1.87	0.028	3.68	0.029	1.74	0.028	3.76	<b>0.025</b>	<b>3.49</b>	0.029	2.49	0.028	10.43
34	0.066		0.061	4.01	0.059	4.81	0.060	4.63	0.060	5.77	0.060	8.16	<b>0.057</b>	<b>8.06</b>	0.059	4.81	0.057	8.52	0.061	4.69	0.059	11.90	0.059	27.50
35	0.182		0.183	2.96	0.184	3.11	0.184	3.03	0.183	4.23	0.179	7.75	0.175	6.40	0.183	3.29	<b>0.175</b>	<b>6.45</b>	0.185	3.17	0.182	5.84	0.179	19.08
36	0.018		0.019	1.36	<b>0.016</b>	<b>2.65</b>	0.017	3.01	0.019	1.54	0.019	1.81	0.017	4.71	0.019	1.70	0.017	5.26	0.018	1.32	0.019	2.08	0.017	11.01
37	0.008		0.005	4.06	0.005	4.20	0.005	4.46	0.005	5.30	0.004	3.99	0.006	7.92	0.005	4.33	0.006	7.74	0.007	4.44	<b>0.004</b>	<b>4.47</b>	0.006	24.52
38	<b>0.000</b>		<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>
39	<b>0.001</b>		0.002	2.46	0.001	3.96	<b>0.001</b>	<b>4.48</b>	<b>0.001</b>	<b>1.90</b>	<b>0.001</b>	<b>1.06</b>	<b>0.001</b>	<b>2.32</b>	0.001	2.90	<b>0.001</b>	<b>1.91</b>	0.002	2.56	<b>0.001</b>	<b>1.41</b>	<b>0.001</b>	<b>1.00</b>
40	<b>0.000</b>		<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>
41	0.248		0.224	6.55	0.225	7.19	0.228	8.68	0.223	9.05	0.231	9.96	<b>0.221</b>	<b>12.86</b>	0.223	7.14	0.222	14.25	0.230	16.77	0.227	18.86	0.232	28.85
42	0.340		0.293	6.78	0.294	11.48	0.298	15.75	0.291	11.73	0.304	13.36	0.289	15.40	0.292	8.41	<b>0.289</b>	<b>17.82</b>	0.301	17.64	0.291	24.84	0.302	29.65
43	0.303		0.283	6.50	0.279	7.45	0.279	10.57	0.279	8.88	0.281	10.48	0.276	11.68	0.281	6.98	<b>0.274</b>	<b>13.04</b>	0.278	22.99	0.276	18.74	0.280	28.99
44	0.006		0.006	1.00	0.006	1.02	0.006	1.13	0.006	1.00	0.006	1.00	0.006	2.55	0.006	1.00	0.006	2.53	0.006	1.33	0.006	1.00	<b>0.006</b>	<b>6.69</b>
45	0.266		0.255	2.83	0.251	3.84	0.252	4.92	0.257	4.24	0.249	8.41	0.250	6.16	0.257	3.13	<b>0.247</b>	<b>7.24</b>	0.253	6.11	0.254	7.70	0.250	22.71
46	0.060		0.060	1.00	0.061	2.15	0.060	2.82	0.060	1.00	0.061	1.24	0.059	4.02	0.060	1.09	<b>0.059</b>	<b>4.39</b>	0.060	1.38	0.061	1.53	0.059	12.30

**Table 21.** Classification errors and the values of the parameter  $k$  obtained for DT

ID	1NN error	kNN error k	Inverse error k	ISquared error k	Rank error k	Fibonacci error k	Dudani error k	Macleod error k	DualD error k	Zavrel error k	Uniform error k	DualU error k
1	0.177	0.183 1.97	0.185 2.63	0.184 2.91	0.184 2.83	0.177 4.87	<b>0.172 6.05</b>	0.184 2.68	0.174 6.34	0.183 2.32	0.178 4.99	0.174 17.12
2	0.855	0.820 21.93	<b>0.809 26.32</b>	0.818 23.15	0.825 26.08	0.854 6.07	0.811 25.47	0.816 22.79	0.810 26.13	0.818 21.70	0.857 13.64	0.813 24.70
3	0.530	0.575 8.11	<b>0.530 8.25</b>	0.548 5.67	0.550 9.25	0.533 3.65	0.575 10.29	0.573 7.67	0.585 9.51	0.587 8.61	0.552 5.66	0.530 5.56
4	0.192	0.195 2.81	0.185 4.37	0.186 3.46	0.193 3.71	0.182 4.34	0.175 6.37	0.204 3.32	<b>0.170 7.26</b>	0.196 3.17	0.186 5.15	0.182 14.74
5	0.001	<b>0.001 2.74</b>	<b>0.001 2.74</b>	<b>0.001 2.74</b>	<b>0.001 3.61</b>	<b>0.001 3.61</b>	0.001 4.86	0.001 2.99	0.001 4.86	<b>0.001 2.74</b>	<b>0.001 3.61</b>	0.001 7.40
6	0.155	0.129 3.63	<b>0.106 16.89</b>	0.127 4.07	0.127 5.88	0.128 10.28	0.113 8.34	0.123 4.83	0.110 10.49	0.127 4.11	0.128 12.18	0.117 10.10
7	<b>0.002</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>	<b>0.002 1.21</b>	<b>0.002 1.00</b>	<b>0.002 1.21</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>	<b>0.002 1.00</b>
8	<b>0.072</b>	0.086 1.12	0.081 3.32	0.087 1.45	0.088 1.35	0.088 1.84	0.082 2.55	0.095 1.38	0.078 2.91	0.091 1.37	0.085 4.61	0.074 5.23
9	0.191	0.195 1.60	0.198 2.57	0.197 3.44	0.199 2.64	0.194 7.93	0.188 6.84	0.199 2.67	<b>0.188 7.70</b>	0.197 1.94	0.193 6.50	0.190 13.97
10	0.166	0.166 1.84	0.170 2.51	0.167 4.21	0.169 3.51	0.164 4.64	0.162 7.03	0.171 2.33	0.162 7.93	0.167 1.98	0.164 9.89	<b>0.159 17.76</b>
11	0.195	0.197 2.01	0.198 3.33	0.194 4.44	0.197 4.23	0.193 6.08	0.182 6.60	0.196 3.47	<b>0.181 7.20</b>	0.198 2.94	0.194 6.28	0.191 15.33
12	0.043	0.016 8.40	0.016 8.25	0.016 8.21	0.017 10.77	0.032 5.47	0.013 7.44	0.016 8.12	0.013 7.52	0.016 8.25	0.029 12.46	<b>0.013 8.03</b>
13	<b>0.112</b>	0.119 1.24	0.121 1.55	0.125 1.70	0.121 1.74	0.119 1.63	0.124 2.93	0.125 1.78	0.121 2.98	0.121 1.53	0.123 2.09	0.113 8.41
14	<b>0.005</b>	0.006 1.20	0.006 1.62	0.006 1.70	0.006 1.70	0.006 1.33	0.006 2.18	0.006 1.67	0.006 2.34	0.006 1.55	0.006 1.67	0.006 3.22
15	0.017	0.016 3.18	0.015 3.38	0.015 3.81	0.016 4.61	0.015 5.70	<b>0.014 8.87</b>	0.016 4.24	0.015 10.30	0.016 3.40	0.015 9.07	0.016 24.01
16	<b>0.009</b>	0.026 3.72	0.025 4.07	0.018 7.42	0.025 4.83	<b>0.009 1.00</b>	0.010 1.13	0.023 4.45	0.010 1.16	0.026 3.76	0.010 2.65	<b>0.009 1.00</b>
17	0.191	0.177 6.85	<b>0.148 9.27</b>	0.170 7.18	0.166 9.15	0.164 7.18	0.158 11.10	0.165 7.72	0.158 12.87	0.169 7.14	0.166 14.57	0.169 19.24
18	0.048	0.045 8.32	0.037 10.09	<b>0.036 12.28</b>	0.043 11.49	0.043 5.62	0.041 16.80	0.043 10.13	0.043 17.86	0.044 8.32	0.039 10.47	0.041 19.13
19	0.549	0.520 8.98	0.507 8.62	0.511 9.82	0.513 10.47	0.509 7.80	0.504 16.25	0.516 9.55	<b>0.499 17.35</b>	0.517 8.59	0.506 21.17	0.527 25.19
20	0.406	0.404 4.14	0.389 9.32	0.392 7.92	0.399 6.11	0.393 5.66	0.385 8.58	0.392 4.22	<b>0.377 11.47</b>	0.397 4.24	0.396 9.28	0.390 21.67
21	0.048	0.044 3.64	0.044 4.24	0.043 4.71	0.040 5.52	<b>0.039 6.52</b>	0.045 9.71	0.040 4.87	0.045 10.27	0.042 4.33	0.040 7.54	0.044 19.82
22	<b>0.150</b>	0.166 1.94	0.181 2.58	0.183 3.77	0.178 2.84	0.161 1.36	0.185 4.95	0.182 3.12	0.188 5.61	0.177 2.47	0.172 4.97	0.165 7.09
23	0.292	0.277 5.03	0.279 5.73	0.270 7.20	0.285 6.79	0.269 5.73	0.280 9.90	0.280 5.89	0.274 11.00	0.276 5.29	0.268 15.66	<b>0.265 18.77</b>
24	0.033	0.026 15.69	0.025 17.90	0.025 20.27	0.026 19.14	0.027 7.05	0.025 22.48	0.026 16.16	0.025 22.62	0.026 15.07	<b>0.024 19.21</b>	0.026 26.88
25	0.243	0.238 3.96	0.228 4.33	0.229 6.60	0.233 7.15	0.230 8.02	0.222 10.35	0.224 5.01	<b>0.218 12.13</b>	0.227 4.33	0.230 13.78	0.229 23.86
26	0.018	0.020 2.52	0.018 5.07	0.017 6.03	0.019 4.63	0.018 4.87	0.016 8.71	0.018 4.54	<b>0.015 9.77</b>	0.017 4.76	0.018 7.99	0.018 18.93
27	0.182	0.155 6.86	0.150 9.37	0.153 7.32	0.152 9.34	0.154 11.00	0.149 14.23	0.150 7.93	<b>0.149 16.24</b>	0.153 7.10	0.151 23.00	0.156 28.03
28	0.134	0.101 10.05	0.097 13.95	0.100 10.06	0.099 15.22	0.111 10.65	0.095 20.41	0.099 10.88	<b>0.094 22.70</b>	0.100 10.15	0.103 25.43	0.107 21.47
29	0.833	<b>0.623 21.64</b>	<b>0.623 21.64</b>	<b>0.623 21.64</b>	0.743 23.79	0.833 1.00	<b>0.623 21.64</b>	<b>0.623 21.64</b>	<b>0.623 21.64</b>	<b>0.623 21.64</b>	0.833 1.00	<b>0.623 21.64</b>
30	0.139	0.148 1.65	0.143 2.81	0.141 3.04	0.145 3.00	<b>0.138 4.97</b>	0.141 6.08	0.144 2.87	0.140 6.37	0.144 2.73	0.139 6.32	0.140 13.67
31	0.004	<b>0.003 2.70</b>	<b>0.003 2.70</b>	<b>0.003 2.70</b>	<b>0.003 3.55</b>	<b>0.003 3.55</b>	<b>0.003 4.63</b>	<b>0.003 2.71</b>	<b>0.003 4.63</b>	<b>0.003 2.70</b>	<b>0.003 3.55</b>	0.004 11.69
32	<b>0.085</b>	0.088 1.80	0.089 2.81	0.089 2.84	0.086 2.90	0.087 2.65	0.090 5.45	0.089 2.77	0.088 5.66	0.089 2.81	0.089 4.67	0.086 10.17
33	0.072	0.072 2.76	0.063 4.09	0.063 4.06	0.070 4.89	0.064 5.59	0.064 8.56	0.064 4.07	<b>0.063 9.13</b>	0.064 4.11	0.065 6.65	0.067 13.81
34	0.098	0.090 3.02	0.090 4.21	0.091 4.01	0.090 5.08	0.091 5.62	0.089 7.31	0.090 4.08	<b>0.089 8.18</b>	0.091 4.08	0.091 7.49	0.091 25.92
35	0.102	0.098 2.88	0.101 3.45	0.094 3.30	0.097 4.37	0.095 6.02	0.099 8.14	0.096 3.95	0.096 8.97	0.096 3.42	0.095 5.95	<b>0.093 23.69</b>
36	0.017	0.019 2.54	0.018 5.93	0.018 5.63	0.018 3.42	0.017 3.34	0.016 6.39	0.018 3.56	0.016 7.09	0.018 3.16	<b>0.015 4.07</b>	0.017 14.93
37	0.064	0.045 9.43	0.045 9.74	0.045 9.81	<b>0.038 11.84</b>	0.046 7.02	0.043 20.09	0.043 11.58	0.045 20.59	0.045 9.74	0.044 13.29	0.056 24.28
38	<b>0.001</b>	<b>0.001 1.00</b>	<b>0.001 1.02</b>	<b>0.001 1.00</b>	<b>0.001 1.00</b>	<b>0.001 1.00</b>	0.002 1.16	0.003 1.10	0.002 1.16	<b>0.001 1.00</b>	<b>0.001 1.00</b>	<b>0.001 1.28</b>
39	0.003	0.002 4.40	0.002 4.09	0.002 4.07	0.002 4.66	0.002 4.21	0.001 6.37	0.002 4.10	0.001 6.18	0.002 4.09	0.002 4.76	<b>0.001 7.09</b>
40	<b>0.000</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>	<b>0.000 1.00</b>
41	0.204	0.193 3.49	0.194 4.04	0.193 4.42	0.193 5.54	0.192 8.03	0.192 8.49	0.193 4.36	<b>0.191 8.74</b>	0.193 3.96	0.194 7.17	0.195 26.68
42	0.273	0.248 5.32	0.246 6.61	0.244 7.52	0.244 8.10	0.248 11.45	0.239 12.14	0.246 6.43	<b>0.237 13.63</b>	0.246 5.93	0.242 20.84	0.253 29.57
43	0.262	0.252 3.99	0.251 5.44	0.249 6.34	0.249 6.43	0.244 8.81	0.244 9.72	0.250 5.49	<b>0.243 10.93</b>	0.249 4.99	0.244 11.99	0.245 29.06
44	<b>0.002</b>	0.002 1.04	0.002 1.23	0.002 2.08	0.002 1.08	0.002 1.06	0.002 1.96	0.002 1.12	0.002 2.08	0.002 1.05	0.002 1.33	<b>0.002 3.28</b>
45	0.168	0.173 1.76	0.174 2.25	0.173 2.80	0.173 2.57	0.171 3.96	0.169 5.77	0.174 2.53	0.169 6.35	0.173 2.10	0.170 4.23	<b>0.167 15.02</b>
46	0.069	0.072 1.98	0.065 6.11	0.066 4.59	0.069 4.70	0.065 8.44	0.061 7.15	0.069 4.25	<b>0.059 7.36</b>	0.068 3.77	0.064 8.24	0.060 15.22

Table 22. Classification errors and the values of the parameter  $k$  obtained for LCS

ID	1NN error	kNN error	k	Inverse error	k	ISquared error	k	Rank error	k	Fibonacci error	k	Dudani error	k	Macleod error	k	DualD error	k	Zavrel error	k	Uniform error	k	DualU error	k
1	0.236	0.240	1.74	0.242	1.89	0.237	2.67	0.242	2.40	0.236	4.76	0.230	7.19	0.243	1.97	0.231	7.43	0.234	2.69	0.235	4.75	<b>0.228</b>	<b>22.93</b>
2	0.333	0.332	2.67	0.333	3.04	0.334	3.82	0.330	4.23	0.329	10.29	<b>0.321</b>	<b>7.35</b>	0.334	3.04	0.323	8.46	0.332	4.60	0.324	9.71	0.329	21.82
3	0.508	0.580	7.14	0.552	9.68	0.500	8.77	0.595	7.56	0.518	5.03	0.575	9.17	0.572	7.71	0.568	8.42	0.520	2.36	0.493	8.56	<b>0.490</b>	<b>17.92</b>
4	0.195	0.165	3.52	0.162	3.41	<b>0.156</b>	<b>3.77</b>	0.159	4.51	0.166	5.18	0.160	7.22	0.165	3.54	0.161	7.35	0.195	1.56	0.173	5.16	0.188	14.32
5	<b>0.000</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>
6	<b>0.005</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	<b>0.005</b>	<b>1.00</b>	0.005	1.10	<b>0.005</b>	<b>1.00</b>	0.005	4.64
7	<b>0.002</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.10</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.10</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>	<b>0.002</b>	<b>1.00</b>
8	<b>0.118</b>	0.140	1.16	0.151	1.38	0.149	1.64	0.140	1.24	0.136	1.60	0.139	2.17	0.153	1.36	0.125	2.23	<b>0.118</b>	<b>1.00</b>	0.141	2.62	0.122	4.06
9	0.202	0.204	1.16	0.204	1.23	0.204	1.35	0.204	1.43	0.205	2.73	<b>0.201</b>	<b>5.26</b>	0.204	1.36	0.201	5.74	0.202	2.38	0.208	4.36	<b>0.201</b>	<b>14.81</b>
10	0.172	0.175	1.19	0.178	1.57	0.180	2.09	0.173	1.60	0.178	3.39	0.173	5.70	0.176	1.67	0.173	6.05	<b>0.171</b>	<b>3.09</b>	0.179	4.70	0.172	14.33
11	0.208	0.210	1.20	0.211	1.41	0.212	2.12	0.211	1.51	0.211	2.53	<b>0.206</b>	<b>5.65</b>	0.211	1.48	0.206	6.15	0.207	2.47	0.212	6.80	0.208	12.30
12	<b>0.003</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>	<b>0.003</b>	<b>1.00</b>
13	0.136	0.128	2.16	0.130	2.64	0.131	2.73	0.130	2.93	0.139	2.83	0.131	4.82	0.129	2.59	0.130	5.18	<b>0.122</b>	<b>3.47</b>	0.143	4.61	0.129	12.64
14	0.007	0.008	1.16	0.008	1.21	0.008	1.30	0.008	1.36	0.008	1.42	0.007	2.72	0.008	1.24	0.007	2.74	<b>0.007</b>	<b>1.69</b>	0.008	1.69	<b>0.007</b>	<b>11.70</b>
15	0.010	0.010	1.30	0.011	2.01	0.011	2.16	0.010	1.71	0.011	2.38	0.010	5.77	0.011	1.59	0.010	6.13	<b>0.010</b>	<b>4.50</b>	0.011	2.77	0.010	13.98
16	<b>0.019</b>	0.021	1.20	0.023	1.66	0.020	1.73	0.023	1.66	<b>0.019</b>	<b>1.00</b>	0.025	2.92	0.021	1.36	0.020	2.03	<b>0.019</b>	<b>1.00</b>	0.021	4.08	<b>0.019</b>	<b>1.00</b>
17	0.127	0.130	3.70	0.124	4.99	0.126	5.86	0.126	5.83	0.114	7.48	0.120	9.91	0.124	5.10	0.120	10.45	0.127	2.56	<b>0.112</b>	<b>10.21</b>	0.113	22.18
18	<b>0.015</b>	0.021	2.98	0.021	3.96	0.021	4.58	0.022	4.47	0.017	2.65	0.016	6.07	0.022	3.75	0.017	6.16	0.016	4.69	0.019	5.25	0.016	10.10
19	0.546	0.521	13.79	0.511	14.07	0.507	15.19	0.511	15.37	0.517	8.92	0.499	19.48	0.510	13.92	<b>0.497</b>	<b>19.83</b>	0.546	2.20	0.506	23.36	0.535	23.65
20	0.394	0.404	4.76	0.356	9.23	<b>0.344</b>	<b>11.22</b>	0.392	10.15	0.380	8.64	0.352	11.80	0.388	6.90	0.345	13.41	0.394	1.60	0.383	12.35	0.384	25.39
21	0.047	0.041	7.62	0.040	8.01	0.040	7.85	0.039	9.59	0.042	6.04	0.038	12.80	0.039	7.91	<b>0.037</b>	<b>13.39</b>	0.038	8.84	0.040	13.14	0.043	21.00
22	0.119	0.135	2.80	0.131	3.58	0.131	3.62	0.139	4.28	0.124	3.73	0.123	5.65	0.130	3.61	0.124	5.92	0.119	1.76	0.130	5.92	<b>0.117</b>	<b>12.85</b>
23	0.279	0.262	5.80	0.244	6.01	<b>0.235</b>	<b>6.52</b>	0.248	7.75	0.252	5.44	0.245	10.11	0.237	6.25	0.243	10.76	0.270	2.88	0.250	10.16	0.254	21.27
24	0.007	0.007	1.12	0.007	1.28	0.007	1.29	0.007	1.22	0.007	1.24	0.007	3.79	0.007	1.21	<b>0.007</b>	<b>4.02</b>	0.007	1.72	0.007	1.33	0.007	10.27
25	0.192	0.195	2.19	0.192	3.64	0.191	5.44	0.194	4.58	0.191	7.12	<b>0.182</b>	<b>8.27</b>	0.193	3.74	0.184	9.60	0.189	12.56	0.188	13.81	0.187	24.36
26	0.028	0.031	1.60	0.030	2.95	0.030	4.21	0.030	3.24	0.029	3.94	0.028	7.21	0.030	3.13	0.027	8.39	0.027	4.91	0.029	6.34	<b>0.027</b>	<b>20.80</b>
27	0.165	0.161	4.77	0.159	4.96	0.158	5.03	0.157	7.19	0.159	9.15	0.156	12.74	0.160	6.03	<b>0.156</b>	<b>12.85</b>	0.163	10.81	0.157	16.84	0.159	27.67
28	0.097	0.084	8.21	0.081	8.85	0.080	10.21	0.079	12.89	0.085	9.73	0.077	19.75	0.080	10.31	<b>0.076</b>	<b>20.25</b>	0.092	13.08	0.082	26.11	0.090	28.65
29	<b>0.115</b>	0.123	1.31	0.125	1.56	0.125	1.67	0.122	1.59	0.123	1.82	0.120	2.58	0.125	1.60	0.120	2.70	0.125	1.58	0.125	2.28	<b>0.115</b>	<b>3.04</b>
30	0.253	0.261	2.02	0.261	2.60	0.263	2.69	0.263	3.06	0.252	5.72	0.261	4.89	0.262	2.60	0.260	5.14	0.253	1.12	<b>0.248</b>	<b>6.82</b>	0.251	19.10
31	<b>0.000</b>	0.000	2.12	0.000	2.12	0.000	2.12	<b>0.000</b>	<b>2.73</b>	<b>0.000</b>	<b>1.12</b>	<b>0.000</b>	<b>3.80</b>	0.000	2.29	<b>0.000</b>	<b>3.85</b>	<b>0.000</b>	<b>1.79</b>	<b>0.000</b>	<b>3.04</b>	<b>0.000</b>	<b>1.64</b>
32	<b>0.018</b>	0.019	1.30	0.019	1.47	0.019	1.47	0.020	1.66	0.019	1.54	0.019	2.77	0.019	1.51	0.019	2.61	0.020	1.83	0.019	2.02	0.019	5.40
33	0.018	0.018	1.68	0.018	2.92	0.018	3.08	0.018	3.32	0.018	3.07	0.018	6.48	0.018	2.93	0.018	6.84	0.018	9.43	0.018	7.22	<b>0.018</b>	<b>16.66</b>
34	0.100	0.100	2.62	0.096	4.05	0.096	4.04	0.097	5.21	0.096	4.98	<b>0.095</b>	<b>6.97</b>	0.096	4.06	0.095	7.27	0.100	3.29	0.095	5.13	0.095	24.88
35	0.097	0.100	1.44	0.100	1.74	0.100	1.88	0.100	2.10	0.098	3.40	0.095	5.80	0.101	1.79	0.095	5.89	<b>0.093</b>	<b>4.98</b>	0.097	3.51	0.095	16.76
36	0.020	0.021	2.52	0.021	3.41	0.021	3.40	0.021	3.89	0.021	3.37	0.018	7.92	0.020	3.19	<b>0.018</b>	<b>8.21</b>	0.020	1.14	0.021	4.95	0.019	15.50
37	0.024	0.014	4.23	0.014	4.91	0.015	4.84	0.014	6.00	<b>0.011</b>	<b>4.72</b>	0.014	8.12	0.014	5.08	0.014	8.16	0.016	6.25	0.013	7.46	0.017	20.42
38	<b>0.085</b>	<b>0.085</b>	<b>1.00</b>	<b>0.085</b>	<b>1.00</b>	0.087	1.13	<b>0.085</b>	<b>1.00</b>	<b>0.085</b>	<b>1.00</b>	0.085	1.22	<b>0.085</b>	<b>1.00</b>	<b>0.085</b>	<b>1.25</b>	0.085	1.22	0.085	1.72	0.086	1.22
39	<b>0.002</b>	0.002	1.26	0.002	1.31	0.002	1.43	0.002	1.46	<b>0.002</b>	<b>1.12</b>	0.002	1.69	0.002	1.33	0.002	1.88	0.002	1.42	0.002	1.44	<b>0.002</b>	<b>1.31</b>
40	<b>0.000</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>	<b>0.000</b>	<b>1.00</b>
41	0.222	0.212	3.57	0.212	3.57	0.212	3.53	0.212	4.84	0.212	8.28	<b>0.212</b>	<b>8.37</b>	0.213	3.68	0.212	8.40	0.223	3.47	0.212	6.78	0.213	25.48
42	0.296	0.274	4.98	0.273	5.86	0.272	6.62	0.269	8.73	0.273	10.90	<b>0.266</b>	<b>13.47</b>	0.272	6.39	0.267	13.99	0.294	4.38	0.268	21.70	0.278	29.19
43	0.274	0.264	4.73	0.259	5.63	0.258	6.39	0.256	7.79	0.257	11.35	0.253	11.57	0.257	6.40	<b>0.253</b>	<b>11.84</b>	0.272	5.39	0.257	15.31	0.262	28.43
44	<b>0.002</b>	0.002	1.30	0.002	1.44	0.002	2.16	0.002	1.72	0.002	1.48	0.002	3.84	0.002	1.48	0.002	4.10	<b>0.002</b>	<b>1.52</b>	0.002	2.05	<b>0.002</b>	<b>5.99</b>
45	0.223	0.223	1.96	0.224	1.99	0.215	2.72	0.223	2.45	0.222	4.20	0.216	6.53	0.224	1.99	0.217	6.65	0.219	3.02	0.221	4.35	<b>0.214</b>	<b>22.38</b>
46	<b>0.035</b>	<b>0.035</b>	<b>1.00</b>	<b>0.035</b>	<b>1.00</b>	<b>0.035</b>	<b>1.00</b>	<b>0.035</b>	<b>1.00</b>	<b>0.035</b>	<b>1.00</b>	0.037	1.98	<b>0.035</b>	<b>1.00</b>	0.037	2.26	0.036	2.06	<b>0.035</b>	<b>1.00</b>	0.036	10.43

**Table 23.** Classification errors and the values of the parameter  $k$  obtained for ERP

ID	1NN			kNN			Inverse			ISquared			Rank			Fibonacci			Dudani			Macleod			DualD			Zavrel			Uniform			DualU		
	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k	error	error	k			
1	0.159	0.160	1.08	0.162	0.162	1.62	0.161	0.161	1.59	0.161	0.161	1.41	0.162	0.162	3.35	0.161	0.161	6.61	0.162	0.162	1.47	0.161	0.161	7.01	0.159	0.159	1.52	0.160	0.160	4.67	0.158	0.158	13.29			
2	0.854	0.822	21.38	0.809	0.809	26.59	0.808	0.808	26.47	0.824	0.824	24.83	0.853	0.853	5.29	0.809	0.809	26.12	0.816	0.816	22.65	0.809	0.809	26.52	0.805	0.805	26.66	0.855	0.855	13.57	0.811	0.811	24.99			
3	0.532	0.587	9.80	0.533	0.533	9.53	0.528	0.528	6.52	0.560	0.560	11.65	0.548	0.548	2.78	0.577	0.577	9.55	0.560	0.560	8.91	0.582	0.582	9.69	0.545	0.545	3.95	0.548	0.548	6.43	0.532	0.532	5.55			
4	0.193	0.211	2.65	0.188	0.188	5.12	0.186	0.186	7.32	0.207	0.207	3.45	0.173	0.173	3.92	0.182	0.182	6.44	0.191	0.191	3.75	0.180	0.180	7.54	0.193	0.193	1.52	0.177	0.177	5.33	0.188	0.188	14.64			
5	0.005	0.001	4.54	0.001	0.001	4.06	0.001	0.001	4.06	0.001	0.001	5.37	0.000	0.000	4.42	0.001	0.001	8.68	0.001	0.001	4.34	0.001	0.001	8.68	0.002	0.002	10.70	0.000	0.000	6.74	0.001	0.001	20.75			
6	0.156	0.130	3.45	0.105	0.105	20.12	0.105	0.105	22.97	0.129	0.129	5.63	0.130	0.130	10.74	0.115	0.115	7.68	0.125	0.125	4.26	0.113	0.113	9.39	0.115	0.115	12.47	0.130	0.130	12.11	0.118	0.118	9.91			
7	0.002	0.002	1.02	0.002	0.002	1.02	0.002	0.002	1.02	0.002	0.002	1.03	0.002	0.002	1.00	0.002	0.002	1.06	0.002	0.002	1.02	0.002	0.002	1.06	0.002	0.002	1.00	0.002	0.002	1.00	0.002	0.002	1.00			
8	0.043	0.043	1.00	0.052	0.052	1.89	0.047	0.047	1.36	0.043	0.043	1.00	0.048	0.048	1.12	0.045	0.045	1.17	0.043	0.043	1.00	0.047	0.047	1.23	0.043	0.043	1.08	0.047	0.047	2.16	0.043	0.043	1.16			
9	0.198	0.201	1.34	0.202	0.202	2.00	0.203	0.203	2.64	0.201	0.201	1.66	0.201	0.201	5.40	0.192	0.192	5.52	0.199	0.199	2.10	0.194	0.194	5.85	0.196	0.196	3.71	0.199	0.199	7.27	0.197	0.197	17.70			
10	0.176	0.180	1.32	0.179	0.179	1.50	0.182	0.182	2.22	0.180	0.180	1.64	0.178	0.178	4.80	0.177	0.177	5.76	0.180	0.180	1.53	0.177	0.177	6.44	0.173	0.173	3.12	0.175	0.175	7.00	0.171	0.171	16.53			
11	0.204	0.209	1.87	0.209	0.209	2.33	0.203	0.203	3.37	0.209	0.209	2.64	0.204	0.204	6.00	0.199	0.199	6.06	0.208	0.208	2.43	0.200	0.200	6.50	0.204	0.204	2.55	0.203	0.203	8.26	0.201	0.201	16.65			
12	0.043	0.024	6.94	0.018	0.018	7.39	0.016	0.016	8.04	0.023	0.023	8.76	0.032	0.032	5.44	0.013	0.013	7.69	0.021	0.021	6.77	0.013	0.013	7.71	0.013	0.013	8.25	0.029	0.029	10.43	0.013	0.013	8.03			
13	0.118	0.124	1.22	0.124	0.124	1.91	0.126	0.126	2.09	0.124	0.124	1.58	0.125	0.125	1.81	0.117	0.117	4.63	0.125	0.125	2.02	0.115	0.115	4.83	0.111	0.111	3.61	0.126	0.126	2.12	0.121	0.121	10.78			
14	0.004	0.005	1.82	0.005	0.005	2.05	0.005	0.005	2.70	0.005	0.005	1.94	0.004	0.004	1.24	0.005	0.005	2.64	0.005	0.005	2.12	0.005	0.005	2.76	0.004	0.004	1.22	0.004	0.004	2.42	0.004	0.004	2.77			
15	0.009	0.010	1.81	0.009	0.009	2.23	0.010	0.010	2.28	0.010	0.010	2.32	0.009	0.009	3.59	0.009	0.009	5.37	0.010	0.010	1.75	0.009	0.009	5.40	0.009	0.009	4.15	0.009	0.009	4.19	0.009	0.009	15.50			
16	0.009	0.027	4.38	0.027	0.027	4.63	0.016	0.016	12.61	0.027	0.027	4.15	0.009	0.009	1.00	0.013	0.013	2.41	0.027	0.027	4.12	0.011	0.011	1.87	0.009	0.009	1.00	0.012	0.012	4.27	0.009	0.009	1.00			
17	0.188	0.175	7.14	0.155	0.155	9.61	0.157	0.157	11.48	0.165	0.165	9.42	0.168	0.168	6.63	0.169	0.169	12.41	0.164	0.164	7.76	0.162	0.162	13.91	0.166	0.166	11.46	0.167	0.167	12.89	0.170	0.170	20.35			
18	0.053	0.052	7.00	0.044	0.044	10.34	0.040	0.040	10.68	0.050	0.050	10.50	0.051	0.051	4.18	0.051	0.051	16.14	0.052	0.052	8.86	0.047	0.047	17.26	0.052	0.052	6.18	0.049	0.049	9.31	0.049	0.049	18.66			
19	0.562	0.518	10.87	0.509	0.509	10.34	0.504	0.504	14.44	0.514	0.514	15.14	0.519	0.519	8.29	0.498	0.498	18.89	0.515	0.515	12.39	0.500	0.500	20.24	0.555	0.555	3.24	0.518	0.518	21.35	0.538	0.538	24.50			
20	0.417	0.418	4.71	0.389	0.389	10.65	0.384	0.384	11.27	0.408	0.408	7.19	0.396	0.396	6.92	0.398	0.398	10.53	0.411	0.411	5.35	0.384	0.384	12.72	0.418	0.418	1.96	0.401	0.401	11.73	0.400	0.400	22.71			
21	0.035	0.036	3.26	0.037	0.037	3.74	0.038	0.038	4.40	0.035	0.035	5.70	0.034	0.034	5.11	0.037	0.037	8.93	0.036	0.036	5.68	0.037	0.037	9.58	0.034	0.034	7.03	0.033	0.033	8.85	0.034	0.034	20.15			
22	0.191	0.229	4.40	0.219	0.219	5.85	0.215	0.215	6.37	0.224	0.224	7.27	0.202	0.202	2.17	0.208	0.208	8.85	0.219	0.219	6.16	0.206	0.206	9.61	0.195	0.195	1.70	0.227	0.227	7.76	0.192	0.192	8.43			
23	0.285	0.315	3.84	0.312	0.312	5.05	0.290	0.290	8.65	0.305	0.305	6.87	0.291	0.291	4.51	0.285	0.285	13.19	0.310	0.310	5.41	0.285	0.285	13.77	0.287	0.287	1.81	0.282	0.282	15.16	0.283	0.283	19.38			
24	0.032	0.027	8.95	0.029	0.029	9.03	0.030	0.030	11.28	0.027	0.027	12.11	0.027	0.027	7.48	0.027	0.027	15.43	0.027	0.027	9.55	0.028	0.028	16.44	0.032	0.032	4.18	0.025	0.025	18.14	0.027	0.027	26.63			
25	0.243	0.240	3.05	0.233	0.233	4.33	0.230	0.230	5.08	0.239	0.239	6.12	0.233	0.233	8.54	0.222	0.222	10.03	0.235	0.235	4.69	0.221	0.221	11.25	0.221	0.221	11.20	0.228	0.228	16.59	0.230	0.230	23.71			
26	0.020	0.022	2.86	0.018	0.018	4.25	0.020	0.020	4.67	0.020	0.020	4.83	0.016	0.016	5.32	0.020	0.020	8.27	0.021	0.021	4.26	0.019	0.019	8.99	0.020	0.020	3.13	0.017	0.017	6.27	0.017	0.017	22.63			
27	0.182	0.149	7.58	0.148	0.148	8.24	0.148	0.148	11.46	0.148	0.148	10.52	0.156	0.156	11.14	0.146	0.146	14.14	0.147	0.147	8.42	0.146	0.146	16.48	0.145	0.145	24.65	0.151	0.151	23.15	0.156	0.156	28.29			
28	0.134	0.103	10.12	0.099	0.099	17.32	0.095	0.095	22.83	0.100	0.100	15.93	0.111	0.111	12.16	0.096	0.096	20.48	0.100	0.100	12.16	0.094	0.094	23.41	0.099	0.099	25.32	0.103	0.103	26.64	0.107	0.107	19.07			
29	0.833	0.623	21.64	0.623	0.623	21.64	0.623	0.623	21.64	0.743	0.743	23.79	0.833	0.833	1.00	0.623	0.623	21.64	0.623	0.623	21.64	0.623	0.623	21.64	0.623	0.623	21.64	0.833	0.833	1.00	0.623	0.623	21.64			
30	0.146	0.150	1.49	0.148	0.148	2.40	0.150	0.150	2.69	0.150	0.150	2.46	0.143	0.143	4.94	0.145	0.145	5.59	0.150	0.150	2.25	0.146	0.146	5.89	0.146	0.146	1.58	0.142	0.142	6.26	0.143	0.143	17.66			
31	0.004	0.003	2.70	0.003	0.003	2.70	0.003	0.003	2.70	0.003	0.003	3.55	0.003	0.003	3.55	0.003	0.003	4.95	0.003	0.003	2.71	0.003	0.003	4.95	0.004	0.004	3.10	0.003								