Knowledge-Based Systems 163 (2019) 749–761

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Weighted nearest neighbors feature selection

Peter Bugata, Peter Drotár *

Department of Computers and Informatics, Technical University of Kosice, Letná 9, Košice, Slovakia

ARTICLE INFO

Article history:
Received 15 March 2018
Received in revised form 17 September 2018
Accepted 3 October 2018
Available online 9 October 2018

Keywords:
Feature selection
k-nearest neighbors
Stochastic gradient descent
Euclidean distance
High-dimensional data

ABSTRACT

Huge amounts of data are pervasive in many domains and applications. Unfortunately, high-dimensional data are tightly associated with the curse of dimensionality, a phenomenon that adversely affects many data mining algorithms. Therefore, it is desirable to reduce the dimensionality of the data through preprocessing techniques such as feature selection (FS). Although FS is frequently perceived as a preprocessing technique, in some domains, such as bioinformatics, it is of paramount importance for identifying relevant attributes, and therefore, provides answers to the investigated research question. In this paper, we propose a novel supervised FS method based on k-nearest neighbors algorithm. In particular, we use distance and attribute weighted k-nearest neighbors with gradient descent as an iterative optimization algorithm for finding the function minima. The new method is compared with the state-of-the-art FS algorithms using eight artificial and twelve high-dimensional real-world datasets. The experimental results indicate that the proposed algorithm is able to identify the relevant features and shows the highest prediction performance for all four considered prediction algorithms.

1. Introduction

The storage and processing of high-dimensional data has become a ubiquitous activity in many emerging applications. In areas such as DNA microarray analysis, text and document classification, social media services, and high-resolution images processing, data of extremely high dimensionality [1–3] are produced. As indicated by data in the UCI machine learning repository, dataset sizes grew from thousands of features in the 1990s to several millions of features in 2010 [1]. Datasets of extreme dimensions are becoming an inevitable part of machine learning pipelines and methods that can deal with these data are thus strongly required.

The number of acquired and stored features in datasets is growing; however, many of these features are irrelevant or redundant [3,4]. These features are not only useless in the process of knowledge discovery, but also increase the dimensionality of the Euclidean space in which data points are embedded. Usually, the sample size remains considerably behind the feature number, causing the curse of dimensionality phenomenon to occur and a resultant decrease in classification accuracy [5,6]. Unfortunately, information about which features are useful and which are not, is a priori unknown. One of the frequently used approaches for combatting this problem is dimensionality reduction, which is an important preprocessing step in machine learning and data mining. The aim is to significantly reduce the dimensionality of the original high-dimensional feature space. With the increased dataset dimensionality sizes that are currently experienced, dimensionality reduction has gained even greater significance, because it provides advantages, such as reduced storage needs, shorter classifier training time, and better visualization possibilities, as well as avoiding over-fitting [7].

There are two means of reducing dimensionality: feature extraction (FE) and feature selection (FS). FE reduces dimensionality through mapping a high-dimensional space to a new low-dimensional feature space. The most widely used methods probably are principal component analysis (PCA) [8], non-linear dimensionality reduction (NLDR) [9], and linear discriminant analysis (LDA) [10], and new methods appear regularly [11]. Although FE methods are more successful in terms of finding the optimal solution of a problem, the drawback of FE is that the new feature space has no physical meaning for interpretation. Additionally, FE algorithms can break down because of their high computational complexity [12].

FS methods utilize certain criteria to evaluate the quality of features. The highest ranking features are selected for further processing and the remaining ones are eliminated. Obviously, different evaluation criteria yield different results and select different feature subsets. The advantage of FS is that the features are not transformed to a different space, and therefore, their relationship to the underlying data pattern is preserved. Several approaches exist for FS: filter, wrapper, and embedded. The computationally simplest are univariate filter methods. Their main advantages are simplicity and low computational cost. Although they neglect the relationship between features, in some scenarios they provide

* Corresponding author.
E-mail address: peter.drotar@tuke.sk (P. Drotár).

https://doi.org/10.1016/j.knosys.2018.10.004
0950-7051/

results competitive with other, more complex, algorithms [13]. The filter FS is independent of the classifier and applies the evaluation criteria to select features. A classifier is then built using only the selected features. The filters that are more sophisticated than univariate are the multivariate filter FS methods, which take into account also the relationship between features. These are very widely used because of their simplicity and performance. The most frequently used representatives of this category are the well-established minimal-redundancy-maximal-relevance (mRMR) algorithm [14] and Relief [15]. Additionally, more new FS algorithms are being proposed to enhance the performance of these state-of-the-art techniques in various domains [16].

Wrapper FS methods are more tightly coupled with the classifier. These methods utilize the performance of a particular classifier to evaluate feature subsets using a search strategy. It is the utilization of a search algorithm to find suitable feature subsets that renders these methods computationally demanding. In particular, when applied to high-dimensional datasets their time complexity quickly becomes unsupportable. Recently, several attempts, most of which constituted methods based on evolutionary computation [17], have been made to improve search strategies. Although these methods achieved a degree of success, they still present challenges and need to be investigated further to solve issues such scalability.

The apparent disadvantage of wrapper approach is computational overhead needed for evaluation of each candidate feature set by executing the learning algorithm. Moreover the wrapper FS results are classifier dependent and sometimes tend to overfit the data. On the other hand, the filter FS methods provides better computational complexity than the wrapper methods, however they completely ignore the interaction with classifier [18].

Another relatively new approach is ensemble FS [19]. In this case, multiple base selectors (usually filter FS models) are combined to solve the problem. The objective is to introduce diversity and increase the stability of the FS process, because the weaknesses of single selectors are overcome by collective decisions.

FS is a very active area of research and many interesting papers on the subject regularly appear. Reviews of the topic have been provided in many excellent review papers, such as [16–18,20–22].

The concept of k-nearest neighbors has been already utilized for feature selection in several different ways. Li et al. [23] proposed kNN based method as an alternative to unstable random forest. It randomly selects the subset of features and use these features to train the kNN classifier. The prediction accuracy score is then assigned to each feature included in particular subset. The resulting feature scores are the mean values of all assigned accuracy scores. In similar fashion, Park and Kim [24] proposed FS method using ensemble of kNN classifiers that iteratively looks for significant features. These methods utilize the kNN classifier as a black box and do not explore the kNN algorithm. In contrast, authors of [25] proposed the clever construction of distance matrix and provided the wrapper based FS with embedded kNN classifier. The attribute weighted kNN for feature selection is presented in [26]. The proposed algorithm is limited only to the regression data and the cost function can become discontinuous in this case. The continuous and differentiable cost function is necessary for the correct function of gradient descent algorithm.

In this paper we propose novel FS technique based on k-nearest neighbors algorithm utilizing gradient descent method. The proposed method is suitable for high dimensionality small sample size data (HDSSS) and regression datasets. HDSSS data represent one of the main challenges of current feature selection research since the extremely high dimension has negative influence on reliability of statistical analysis. The weighted kNN shown that it is able to outperform other state-of-the-art FS methods especially for these high dimensional-low sample datasets.

The rest of the paper is organized as follows. Section 2 describes the proposed FS method together with the implementation details. In Section 3, we first evaluate the performance of the proposed method on synthetic data and then compare the influence of the FS methods on the classification performance. Finally, in Section 4 we present our conclusions and outline some future research directions.

2. k-nearest neighbors feature selection

In this chapter, we describe the proposed FS approach. The method is based on the k-nearest neighbors (kNN) technique that employs gradient descent as the iterative optimization procedure for searching function minima.

2.1. Weighted k-nearest neighbors

The kNN algorithm is frequently used to solve classification and regression tasks [6]. The value of the target variable in the test set is determined based on the values of the target variable of its nearest neighbors in the training set. Two parameters need to be set: the natural number k, representing the number of nearest neighbors, and the distance definition. In the case of a classification task, the class of the testing sample is selected according to the majority voting of the nearest neighbors (i.e., the predicted value is equal to the class that has the majority in nearest neighbor sample set). When solving the regression task, the value of the predicted target variable is a real number obtained by the arithmetic mean of the nearest neighbors target variables.

In this paper, if not stated otherwise, the more general case is assumed, i.e., a regression task. Let $\{F_1, F_2, \ldots, F_m\}$ be the set of m numerical predictor variables, $\{x_1, x_2, \ldots, x_n\}$ be the set of n observations, and $\mathbf{y} = (y_1, y_2, \ldots, y_n)$ be the target variable. Then, if we denote the observation in the test set by x_i and the set of indices of k nearest neighbors of sample x_i by N_i^k, the prediction of target variable p_i is given by

$$p_i = \frac{1}{k} \sum_{j \in N_i^k} y_j. \tag{1}$$

kNN is a prototype method based on object similarity. The algorithm searches the k observations in the training set that are most similar to the observation in the testing set. To express similarity, a distance metric is used. The distance definition depends on the type of variables in the dataset. In the case of continuous variables, usually the Euclidean metric is selected. In general, distance is required to fulfill the properties given in [27]. A detailed definition of distances for different types of datasets can be found in [28].

In the conventional kNN technique, all nearest neighbors are equally relevant for predicting the target variable. However, we can assume that observations that are closer to each other are more similar, and thus should more strongly influence the prediction of the target variable. The kNN can be extended so that the voting output of the nearest neighbors is weighted; closer observations have greater weights. This approach is called distance-weighted kNN.

Neighbor weights are given by the transformation of the distance realized through the evaluation function $w(\cdot)$. Then, Eq. (1) for prediction p_i can be rewritten as

$$p_i = \frac{1}{\sum_{j \in N_i^k} w(d_{ij})} \sum_{j \in N_i^k} w(d_{ij}) \cdot y_j, \tag{2}$$

where d_{ij} denotes the distance between observations x_i and x_j and $w(d_{ij})$ is its value after transformation through function $w(\cdot)$. The function $w : \Re_0^+ \rightarrow \langle 0, 1 \rangle$ satisfies the following properties:

- $w(0) = 1$,
- Function $w(\cdot)$ is decreasing (the greater distance means a lower function value)
- $\lim_{d\to\inf} w(d) = 0$.

Another modification of kNN is attribute weighted kNN. Every attribute in $\{F_1, F_2, \ldots, F_m\}$ is assigned a weight that determines the usefulness of the attribute for the determination of the target variable. Feature weighting can be expressed through the Hadamard product as

$$\mathbf{v} \circ \mathbf{x}_i = (v_1 \cdot x_{i1}, v_2 \cdot x_{i2}, \ldots, v_m \cdot x_{im}), \tag{3}$$

where $\mathbf{v} = (v_1, v_2, \ldots, v_m)$ is a weighting vector and $\mathbf{x_i} = (x_{i1}, x_{i2}, \ldots, x_{im})$ is any observation from the dataset.

2.2. Proposed approach: weighted k-nearest neighbors

The proposed FS method, weighted k-nearest neighbors (WkNN-FS), takes advantage of both weighting schemes: distance and attribute weighting. The underlying assumption is that the target variable is most accurately determined by the most relevant attributes of the most nearest neighbors.

The feature weights are used to determine the distance between different observations. If $\mathbf{v}$ is a weighting vector and d_{ij} is the distance between two arbitrary observations $\mathbf{x_i}$ and $\mathbf{x_j}$, then the weighted distance of two observations is defined as

$$d_{ij}(\mathbf{v}) = d(\mathbf{v} \circ \mathbf{x_i}, \mathbf{v} \circ \mathbf{x_j}). \tag{4}$$

This distance definition is used to choose the nearest neighbors. However, a change in the weights can result in a change in the nearest neighbors. The predicted value of target variable $\mathbf{y}$ for the ith observation for weighting vector $\mathbf{v}$ and evaluation function w is given by the weighted average of the target variable of other observations:

$$p_i(\mathbf{v}) = \frac{1}{\sum_{j\neq i} w(d_{ij}(\mathbf{v}))} \sum_{j\neq i} w(d_{ij}(\mathbf{v})).y_j. \tag{5}$$

The prediction accuracy is measured by the difference between predicted value $p_i(\mathbf{v})$ and the actual value of the target variable y_i. The estimation error is expressed by *loss function* $l(y_i, p_i(\mathbf{v}))$, where $l : \Re \times \Re \to \Re_0^+$.

We define the *cost function* as a mean error for the entire dataset as

$$C(\mathbf{v}) = \frac{1}{n} \sum_{i=1}^{n} l(y_i, p_i(\mathbf{v})). \tag{6}$$

The model defined in Eq. (5) allows the regression task to be solved, with the error being the function of the attribute weights. Therefore, the goal is to find the weight vector that yields the smallest error. This is equivalent to solving an optimization task

$$\mathbf{v^{opt}} = \underset{\mathbf{v}\in\Re^m}{\operatorname{argmin}}\, C(\mathbf{v}). \tag{7}$$

The resulting vector $\mathbf{v^{opt}}$ determines the importance of features. The most important features according to WkNN are assigned the highest weights. The algorithm described by pseudocode is provided in Algorithm 1. The method implementation in Python is freely available at github.[1]

[1] https://github.com/bugatap/WkNN-FS.

Algorithm 1 Weighted nearest neighbors feature selection.

set *distance function, distance evaluation function, loss function*
set *learning rate* η, *number of iterations = I (stop condition), min. gradient norm = g (stop condition), negative weights flag, gradient normalization flag*
initialize v ($\mathbf{v} \leftarrow 0$ **or** $\mathbf{v} \leftarrow 1/m$)
repeat
  increment *number of iterations*
  calculate distance matrix considering feature weights $d_{ij}(\mathbf{v})$
  calculate $w(d_{ij}(\mathbf{v}))$ using distance evaluation function $w(\cdot)$
  calculate predictions vector $p_i(\mathbf{v})$
  calculate error using loss function $l(y_i, p_i(\mathbf{v}))$
  calculate gradient-vector of partial derivations of cost function with current weights
  if *gradient normalization flag* = `True` **then** normalize gradient
  end if
  update weights $\mathbf{v}' = \mathbf{v} - \eta * gradient$
  if *negative weights flag* = `True` **then** zero out negative weights
  end if
until *number of iterations* $< I$ **or** *min. gradient norm* $< g$
sort features according **v**
select the top N features **or** features with non-zero weights

2.3. Optimal weights

The FS algorithm attempts to find the vector $\mathbf{v} = (v_1, v_2, \ldots, v_m)$ that minimizes the prediction error. In the proposed method, we consider only continuous and differentiable cost functions $C : \Re^m \to \Re_0^+$, defined in Eq. (6). Since the cost function is continuous and differentiable, we can utilize a gradient descent algorithm to find its minima. Gradient descent is an iterative optimization algorithm that utilizes the gradient of the function to find its minima.

The gradient of function C for weight vector $\mathbf{v}$ is computed in each step of the algorithm and the new vector $\mathbf{v}$ is determined by taking steps proportional to the negative of the gradient. To obtain the gradient, the partial derivations of function C with respect to each weight v_l, for $l \in \{1, 2, \ldots, m\}$, should be calculated as

$$\frac{\partial C(\mathbf{v})}{\partial v_l} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial l(y_i, p_i(\mathbf{v}))}{\partial v_l} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial l(y_i, p_i(\mathbf{v}))}{\partial p_i(\mathbf{v})} \cdot \frac{\partial p_i(\mathbf{v})}{\partial v_l}. \tag{8}$$

To solve Eq. (8), we need to calculate the partial derivation of predicted value $p_i(\mathbf{v})$ with respect to v_l. Therefore, by derivation of Eq. (5)

$$\begin{aligned}\frac{\partial p_i(\mathbf{v})}{\partial v_l} = {} & \frac{-1}{\left[\sum_{k\neq i} w(d_{ik}(\mathbf{v}))\right]^2} \cdot \sum_{k\neq i} w'(d_{ik}(\mathbf{v})) \cdot \frac{\partial d_{ik}(\mathbf{v})}{\partial v_l} \\ & \cdot \sum_{j\neq i} w(d_{ij}(\mathbf{v}))y_j \\ & + \frac{1}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \cdot \sum_{j\neq i} w'(d_{ij}(\mathbf{v})) \cdot \frac{\partial d_{ij}(\mathbf{v})}{\partial v_l} \cdot y_j \end{aligned} \tag{9}$$

is obtained. This can be simplified to

$$\begin{aligned}\frac{\partial p_i(\mathbf{v})}{\partial v_l} = {} & \frac{-1}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \cdot \sum_{k\neq i} w'(d_{ik}(\mathbf{v})) \cdot \frac{\partial d_{ik}(\mathbf{v})}{\partial v_l} \cdot p_i(\mathbf{v}) \\ & + \frac{1}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \cdot \sum_{k\neq i} w'(d_{ik}(\mathbf{v})) \cdot \frac{\partial d_{ik}(\mathbf{v})}{\partial v_l} \cdot y_k. \end{aligned} \tag{10}$$

Next, by a further update we obtain

$$\frac{\partial p_i(\mathbf{v})}{\partial v_l} = \frac{1}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \cdot \sum_{k\neq i} w'(d_{ik}(\mathbf{v})) . \frac{\partial d_{ik}(\mathbf{v})}{\partial v_l} . (y_k - p_i(\mathbf{v})), \quad (11)$$

where w' denotes the first derivation of the distance evaluation function w. Eqs. (8) and (11) define the determination of the gradient of C in general. The distance function, its evaluation function and loss function and their derivatives are to be substituted for more specific computation.

2.4. Method parametrization: distance function, distance evaluation function, and loss function

The proposed FS method has three degrees of freedom. It allows various definitions of distance and distance evaluation functions, and also the selection of different loss functions. In this section, we propose specific methods that can be used for FS and present the rationale behind the selection of the particular function.

2.4.1. Loss function

The loss function is used to express the prediction error in a regression task. The value of the loss function decreases when the value of the predicted variable is close to the actual value of the target variable. The loss function equals zero if the predicted value of the target variable is the same as the value of the target variable. The natural choice for the loss function is the *absolute loss function* (l_1)

$$l_1(y_i, p_i(\mathbf{v})) = |y_i - p_i(\mathbf{v})| . \quad (12)$$

The partial derivation of this error function with respect to v_l can be calculated simply as

$$\frac{\partial l_1(y_i, p_i(\mathbf{v}))}{\partial v_l} = sgn(p_i(\mathbf{v}) - y_i) \frac{\partial p_i(\mathbf{v})}{\partial v_l} . \quad (13)$$

The drawback of the absolute loss function is that it is non differentiable for points that satisfy $y_i - p_i(\mathbf{v}) = 0$. This problem can be solved by using the *square loss function* (l_2)

$$l_2(y_i, p_i(\mathbf{v})) = (y_i - p_i(\mathbf{v}))^2 . \quad (14)$$

Thus, we have partial derivation with respect to v_l

$$\frac{\partial l_2(y_i, p_i(\mathbf{v}))}{\partial v_l} = 2(p_i(\mathbf{v}) - y_i) \frac{\partial p_i(\mathbf{v})}{\partial v_l} . \quad (15)$$

The square loss function is differentiable at each point of its domain. Its disadvantage is that it is sensitive to outliers, because the square of the error is utilized in Eq. (14). Therefore, the resulting cost function is influenced by the large errors introduced by outliers.

Additionally, we consider also the *Huber loss function* (l_δ), which is differentiable and robust to outliers. For *delta* $>$ 0 and pair $(y_i, p_i(\mathbf{v}))$, the Huber loss function is defined as

$$l_\delta(y_i, p_i(\mathbf{v})) = \begin{cases} \frac{1}{2}(y_i - p_i(\mathbf{v}))^2, & \text{if } |y_i - p_i(\mathbf{v})| \leq \delta, \\ \delta |y_i - p_i(\mathbf{v})| - \frac{1}{2}\delta^2, & \text{otherwise.} \end{cases} \quad (16)$$

Then, the partial derivation is

$$\frac{\partial l_\delta(y_i, p_i(\mathbf{v}))}{\partial v_l} = \begin{cases} (p_i(\mathbf{v}) - y_i) . \frac{\partial p_i(\mathbf{v})}{\partial v_l}, & \text{if } |y_i - p_i(\mathbf{v})| \leq \delta, \\ \delta . sgn(p_i(\mathbf{v}) - y_i) . \frac{\partial p_i(\mathbf{v})}{\partial v_l}, & \text{otherwise.} \end{cases} \quad (17)$$

2.4.2. Distance measure and its evaluation function

In [28], the definitions of several distance measures for different types of datasets were provided. Our method is aimed at datasets with numerical predictor variables, and therefore, we focus on measures for these types of data. The suitable choice in this case is Euclidean distance (L2). The Euclidean distance of two observations x_i and x_k as a function of $\mathbf{v}$ is defined as

$$d_{ik}(\mathbf{v}) = L2(\mathbf{v} \circ \mathbf{x_i}, \mathbf{v} \circ \mathbf{x_k}) = \sqrt{\sum_{j=1}^{n} v_j^2 . (x_{ij} - x_{kj})^2}, \quad (18)$$

with its partial derivation with respect to v_l being

$$\frac{\partial d_{ik}(\mathbf{v})}{\partial v_l} = \frac{v_l . (x_{il} - x_{kl})^2}{\sqrt{\sum_{j=1}^{n} v_j^2 . (x_{ij} - x_{kj})^2}} = \frac{v_l . (x_{il} - x_{kl})^2}{d_{ik}(\mathbf{v})} \quad (19)$$

Distance d in the proposed WkNN-FS method is transformed using evaluation function w that satisfies the properties stated in Section 2.1. We assume the evaluation function in general form to be

$$w(d) = e^{-cd^\alpha}, \text{ where } c, \alpha \in \Re^+ . \quad (20)$$

It can be shown that this function satisfies the required criteria. For $\alpha = 2$, the function is equivalent to the radial basis function (RBF) frequently employed as an RBF kernel in support vector machine (SVM) classifiers. The derivation of Eq. (20) can be calculated simply as

$$w'(d) = (e^{-cd^\alpha})' = -c\alpha . d^{\alpha-1} w(d), \quad (21)$$

Thus, for $\alpha = 2$ and $c = 1$ we have $w'(d) = (e^{-d^2})' = -2d . e^{-d^2} = -2d . w(d)$. Substituting the derivation of the evaluation function into Eq. (11), we have

$$\frac{\partial p_i(\mathbf{v})}{\partial v_l} = \frac{2v_l}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \sum_{k\neq i} w(d_{ik}(\mathbf{v})) \cdot (x_{il} - x_{kl})^2 . (p_i(\mathbf{v}) - y_k). \quad (22)$$

Finally, we have $C(\mathbf{v})$, which represents the arithmetic mean of square error for each observation in the form

$$\frac{\partial C(\mathbf{v})}{\partial v_l} = \frac{1}{n} \sum_{i=1}^{n} \frac{\partial l(y_i, p_i(\mathbf{v}))}{\partial v_l} = \frac{2}{n} \sum_{i=1}^{n} (p_i(\mathbf{v}) - y_i) \frac{\partial p_i(\mathbf{v})}{\partial v_l} . \quad (23)$$

Eqs. (22) and (23) define the derivation of the gradient of function C. If in any iteration of the algorithm v_l becomes equal to zero, partial derivation of $C(\mathbf{v})$ with respect to v_l also equals zero and as a result weight v_l would retain its zero value for all following iterations. This unwanted issue can be avoided by using the square root of weights as weight features. Distance $d_{ik}(\mathbf{v})$ after this modification becomes

$$d_{ik}(\mathbf{v}) = L2(\sqrt{\mathbf{v}} \circ \mathbf{x_i}, \sqrt{\mathbf{v}} \circ \mathbf{x_k}) = \sqrt{\sum_{j=1}^{n} v_j . (x_{ij} - x_{kj})^2} . \quad (24)$$

After this update, Eq. (22) can be written as

$$\frac{\partial p_i(\mathbf{v})}{\partial v_l} = \frac{1}{\sum_{k\neq i} w(d_{ik}(\mathbf{v}))} \sum_{k\neq i} w(d_{ik}(\mathbf{v})) \cdot (x_{il} - x_{kl})^2 \cdot (p_i(\mathbf{v}) - y_k). \quad (25)$$

The goal of this modification is to remove the term $2v_l$ from the right hand side of Eq. (22), since it makes the partial derivation of $C(\mathbf{v})$ become zero in cases where $v_l = 0$. This prevents the algorithm from freezing in zero weight.

3. Numerical experiments

We evaluated the performance of the proposed approach from two important aspects of FS: its ability to identify the features correlated with the target variable and its influence on predictor

Table 1
Characteristics of artificial datasets used in this study.

Dataset name	Acronym	Number of samples	Number of features	Relevant features
Linear regression	Reg	200	500	5
Friedman	Fri	200	500	5
Madelon	Mad	200	500	5
MadelonHD	MHD	150	15,000	15
Lin. regression 5k	Reg5k	5000	500	5
Friedman 5k	Fri5k	5000	500	5
Madelon 5k	Mad5k	5000	500	5
MadelonHD 5k	MHD5k	5000	15,000	15

accuracy. To obtain a quantitative measure for the ability to identify relevant features, we applied the *index of success* measure on artificial datasets. Artificial data were used, because in real data it is difficult to ultimately determine which features are relevant for the target variable.

In the second part of our experiments, four frequently used classifiers were employed to evaluate the extent to which the proposed methods improve the prediction performance. In this case, twelve publicly available high-dimensional datasets were used.

In the experiments, four different versions of the proposed WkNN-FS were considered. As a loss function, we compared the square loss and Huber loss function, for the distance measure we employed Euclidean distance in all the considered alternatives, and for the distance evaluation function we chose the exponential function (exp) $w(d) = e^{-d}$ and RBF $w(d) = e^{-d^2}$. The specific combination of parameters is always indicated in brackets. We included for comparison also seven frequently used FS methods, reliefF [29], mRMR with mutual information criterion [14], f-score [30], l1-SVM [31], chi-squared (chi2) based FS, wrapper FS based on kNN, and Random kNN FS (RkNN) [23] as a baseline for the newly proposed methods.

3.1. Index of success on artificial datasets

The advantage of artificial datasets is that the underlying pattern in the data is a priori known, and therefore, we could exactly evaluate the success of the FS method in terms of selecting the relevant features. We used four regression problems and four classification problems, of which two were high-dimensional datasets. The basic characteristics of the datasets are provided in Table 1. There are two groups of datasets, one containing hundreds of samples, and second containing 5000 samples. This is to evaluate algorithms behavior in different conditions.

Our primary interest was to evaluate the accuracy of the FS methods in terms of selecting the features that are relevant to the target variable. The quality of the selection process was evaluated by the index of success. The index of success ($Suc.$) is defined as

$$Suc. = \left[\frac{R_s}{R_t} - \alpha\frac{I_s}{I_t}\right], \tag{26}$$

Table 3
Characteristics of the real-world datasets used in this study.

Dataset [Source]	No. of samples	No. of features	No. Class 0	No. of Class 1
Alon [32]	62	2,000	40	22
Burczynski [33]	127	22,283	85	42
Chowdary [34]	104	22,283	62	42
Chin [35]	118	22,215	43	75
Golub [36]	72	7,129	47	25
Gordon [37]	181	12,533	94	87
Pomeroy [38]	60	7,128	39	21
Singh [39]	102	12,600	52	50
Tian [40]	173	12,625	36	137
Reg01 [41,42]	89	5,787	–	–
Reg02 [41,42]	76	5,144	–	–
Reg03 [41,42]	133	5,787	–	–

where R_s is the number of selected relevant features, I_s is the number of selected irrelevant features, R_t is the total number of relevant features, and I_t represents the total number of irrelevant features [43]. The term $\alpha = \min\{\frac{1}{2}, \frac{R_t}{I_t}\}$ is used to enhance that the inclusion of irrelevant features is preferred to the omission of relevant ones. The measure *Suc.* captures a method's capability to correctly identify significant features relevant for the computation of the target variable.

To evaluate $Suc.$, the rate of the number of features returned by FS should be determined. We followed the approach proposed in [43] and set the rate of the number of returned features as 3% of all features for a particular dataset. If all the relevant features were selected at first, we set $Suc. = 1$.

The Linear Regression dataset and Friedman datasets constituted the regression task. The Linear Regression dataset was generated by applying a random linear regression model with $N_{inf} = 5$ nonzero regressors to the well-conditioned, centered, Gaussian input with unit variance and some Gaussian centered noise with an adjustable scale. The Friedman dataset was generated according to the rule for the Friedman 1 dataset. The Madelon and MadelonHD datasets were generated according to the same rule, and only some of the parameters were changed to achieve a high dimensionality for the MadelonHD dataset. These classification tasks are given by the clusters of points normally distributed about the vertices of an $N_{inf} = 5$ ($N_{inf} = 15$ for MadelonHD) dimensional hypercube with sides of length 2*2 and four clusters are assigned to each class. All the datasets were obtained from *scikit-learn* dataset generators [44].

3.2. Index of success results

The $Suc.$ results for the evaluated FS methods are presented in Table 2. The best score for each dataset is emphasized by bold font.

For small sample synthetic datasets, the highest average $Suc. = 0.85$ rate was achieved by the new WkNN-FS (l_2,exp), which outperformed other methods on three out of the four artificial datasets. WkNN-FS (l_2,exp) was followed by two variations of the

Table 2
Index of success for different feature selection methods on eight artificial datasets.

Method	Reg	Fri	Mad	MHD	*mean*	Reg5k	Fri5k	Mad5k	MHD5k	*mean* (5*k*)
reliefF	0.40	0.99	0.60	0.40	0.60	0.80	**1.00**	**1.00**	0.93	0.93
f-score	0.60	0.80	0.60	0.47	0.62	**1.00**	0.99	0.80	0.53	0.83
mRMR/mutual info	0.60	0.80	0.40	0.40	0.55	0.99	**1.00**	0.80	0.53	0.83
chi2	0.80	0.80	0.40	0.40	0.60	0.99	**1.00**	0.80	0.60	0.85
l1-SVM	**1.00**	0.80	0.40	0.13	0.58	**1.00**	0.80	0.80	0.80	0.85
RkNN	0.60	0.80	0.60	0.40	0.60	0.80	**1.00**	0.80	0.53	0.78
kNN-wrapper	0.80	**1.00**	**1.00**	0.13	0.73	**1.00**	**1.00**	**1.00**	0.99	**1.00**
WkNN-FS (l_2,exp)	**1.00**	0.80	**1.00**	**0.60**	**0.85**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
WkNN-FS (l_2,rbf)	**1.00**	0.80	0.99	0.53	0.83	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
WkNN-FS (l_δ,exp)	0.99	0.80	**1.00**	0.40	0.80	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
WkNN-FS (l_δ,rbf)	**1.00**	0.80	0.99	0.53	0.83	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**

Table 4
Influence of feature selection methods on F_1 score for nine high-dimensional datasets. Classification problems. Selected 30 features.

		noFS	reliefF	f-score	mRMR	chi2	l1-SVM	RkNN	kNN wrapper	WkNN-FS (l_2,exp)	WkNN-FS (l_2,rbf)	WkNN-FS (l_δ,exp)	WkNN-FS (l_δ,rbf)
	NB	0.79 ± 0.20	0.76 ± 0.12	0.92 ± 0.08	0.94 ± 0.06	0.91 ± 0.07	**1.00 ± 0.00**	0.97 ± 0.05	0.86 ± 0.12	0.99 ± 0.03	0.97 ± 0.06	**1.00 ± 0.00**	0.99 ± 0.03
	SVC	0.84 ± 0.23	0.93 ± 0.07	0.97 ± 0.05	0.97 ± 0.06	0.93 ± 0.13	**1.00 ± 0.00**	**1.00 ± 0.00**	0.94 ± 0.06	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Burczynski	RF	0.89 ± 0.15	0.86 ± 0.14	0.95 ± 0.07	0.97 ± 0.06	0.94 ± 0.11	0.94 ± 0.07	0.97 ± 0.06	0.91 ± 0.12	**0.99 ± 0.04**	**0.99 ± 0.04**	**0.99 ± 0.04**	**0.99 ± 0.04**
	kNN	0.69 ± 0.19	0.78 ± 0.22	0.96 ± 0.06	0.94 ± 0.11	0.78 ± 0.32	**1.00 ± 0.00**	0.91 ± 0.18	0.94 ± 0.08	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.80 ± 0.07	0.83 ± 0.07	0.95 ± 0.02	0.96 ± 0.02	0.89 ± 0.07	0.99 ± 0.02	0.96 ± 0.03	0.91 ± 0.03	0.99 ± 0.01	0.99 ± 0.01	**1.00 ± 0.00**	0.99 ± 0.01
	NB	0.89 ± 0.06	0.91 ± 0.06	0.92 ± 0.07	0.92 ± 0.05	0.92 ± 0.07	0.97 ± 0.03	0.96 ± 0.04	0.62 ± 0.16	**0.98 ± 0.04**	0.94 ± 0.04	**0.98 ± 0.04**	0.92 ± 0.06
	SVC	0.89 ± 0.06	0.93 ± 0.06	0.93 ± 0.05	0.93 ± 0.04	0.91 ± 0.05	**0.99 ± 0.03**	0.96 ± 0.04	0.93 ± 0.06	**0.99 ± 0.03**	0.97 ± 0.03	0.98 ± 0.04	0.96 ± 0.04
Chin	RF	0.92 ± 0.05	0.92 ± 0.06	0.93 ± 0.06	0.93 ± 0.05	0.92 ± 0.05	**0.95 ± 0.05**	0.93 ± 0.05	0.94 ± 0.06	**0.95 ± 0.04**	0.93 ± 0.05	**0.95 ± 0.05**	0.92 ± 0.06
	kNN	0.87 ± 0.05	0.92 ± 0.06	0.93 ± 0.06	0.92 ± 0.05	0.92 ± 0.05	0.93 ± 0.04	0.93 ± 0.05	0.93 ± 0.06	0.99 ± 0.03	0.99 ± 0.02	0.98 ± 0.04	**1.00 ± 0.00**
	mean	0.89 ± 0.02	0.92 ± 0.01	0.93 ± 0.01	0.92 ± 0.01	0.92 ± 0.00	0.96 ± 0.02	0.94 ± 0.01	0.86 ± 0.13	**0.98 ± 0.02**	0.96 ± 0.03	0.97 ± 0.01	0.95 ± 0.03
	NB	0.85 ± 0.06	0.90 ± 0.06	0.89 ± 0.04	0.91 ± 0.07	0.89 ± 0.04	0.91 ± 0.07	0.90 ± 0.04	0.38 ± 0.17	**0.93 ± 0.04**	0.89 ± 0.05	0.89 ± 0.04	0.89 ± 0.04
	SVC	0.88 ± 0.01	0.91 ± 0.02	0.91 ± 0.03	0.91 ± 0.03	0.91 ± 0.04	**0.95 ± 0.03**	**0.95 ± 0.04**	0.91 ± 0.02	0.95 ± 0.03	0.90 ± 0.02	**0.95 ± 0.03**	0.91 ± 0.03
Tian	RF	0.88 ± 0.01	0.89 ± 0.04	0.91 ± 0.04	**0.92 ± 0.04**	0.91 ± 0.04	0.91 ± 0.03	0.91 ± 0.03	0.88 ± 0.02	**0.92 ± 0.02**	0.89 ± 0.03	0.91 ± 0.03	0.90 ± 0.03
	kNN	0.88 ± 0.01	0.89 ± 0.01	0.89 ± 0.02	0.90 ± 0.02	0.89 ± 0.01	0.88 ± 0.01	0.88 ± 0.01	0.91 ± 0.06	0.94 ± 0.03	0.95 ± 0.05	0.92 ± 0.03	**0.97 ± 0.03**
	mean	0.88 ± 0.02	0.90 ± 0.01	0.90 ± 0.01	0.91 ± 0.01	0.90 ± 0.01	0.91 ± 0.02	0.91 ± 0.03	0.77 ± 0.23	**0.93 ± 0.01**	0.91 ± 0.03	0.92 ± 0.02	0.92 ± 0.03
	NB	0.88 ± 0.10	0.97 ± 0.05	0.97 ± 0.05	0.96 ± 0.07	0.93 ± 0.11	**0.99 ± 0.04**	0.97 ± 0.05	0.55 ± 0.32	0.97 ± 0.07	0.97 ± 0.07	0.97 ± 0.05	0.97 ± 0.07
	SVC	0.95 ± 0.07	0.97 ± 0.05	0.97 ± 0.05	0.97 ± 0.05	0.95 ± 0.07	0.95 ± 0.07	0.97 ± 0.05	0.55 ± 0.33	0.99 ± 0.03	**1.00 ± 0.00**	0.98 ± 0.04	**1.00 ± 0.00**
Chowdary	RF	0.95 ± 0.08	**0.97 ± 0.05**	**0.97 ± 0.05**	**0.97 ± 0.05**	0.95 ± 0.09	**0.97 ± 0.05**	**0.97 ± 0.05**	0.96 ± 0.06	**0.97 ± 0.05**	0.96 ± 0.06	**0.97 ± 0.05**	0.96 ± 0.06
	kNN	0.85 ± 0.14	0.95 ± 0.09	0.99 ± 0.03	0.91 ± 0.10	0.85 ± 0.12	**1.00 ± 0.00**	0.97 ± 0.05	0.96 ± 0.10	0.99 ± 0.03	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.91 ± 0.04	0.97 ± 0.01	**0.98 ± 0.01**	0.96 ± 0.03	0.92 ± 0.04	**0.98 ± 0.02**	0.97 ± 0.00	0.76 ± 0.20	**0.98 ± 0.01**	**0.98 ± 0.02**	**0.98 ± 0.01**	**0.98 ± 0.02**
	NB	0.98 ± 0.06	0.94 ± 0.10	0.94 ± 0.10	0.94 ± 0.10	0.96 ± 0.09	0.99 ± 0.04	**1.00 ± 0.00**	0.85 ± 0.12	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	SVC	0.62 ± 0.34	0.97 ± 0.07	0.97 ± 0.07	0.97 ± 0.07	0.97 ± 0.07	0.99 ± 0.04	0.98 ± 0.06	0.91 ± 0.14	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Golub	RF	0.98 ± 0.06	0.98 ± 0.06	0.94 ± 0.09	0.96 ± 0.08	0.94 ± 0.09	0.96 ± 0.08	0.98 ± 0.06	0.95 ± 0.11	**1.00 ± 0.00**	0.96 ± 0.08	0.98 ± 0.06	0.98 ± 0.06
	kNN	0.32 ± 0.32	0.88 ± 0.17	0.90 ± 0.17	0.90 ± 0.17	0.84 ± 0.18	0.90 ± 0.17	0.98 ± 0.06	0.98 ± 0.06	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.72 ± 0.28	0.94 ± 0.04	0.93 ± 0.02	0.94 ± 0.03	0.93 ± 0.05	0.96 ± 0.04	0.98 ± 0.01	0.92 ± 0.05	**1.00 ± 0.00**	0.99 ± 0.02	**1.00 ± 0.00**	**1.00 ± 0.00**
	NB	0.98 ± 0.04	**1.00 ± 0.00**	0.98 ± 0.03	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	0.98 ± 0.04	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	SVC	0.98 ± 0.04	0.99 ± 0.02	0.99 ± 0.02	**1.00 ± 0.00**	0.99 ± 0.02	**1.00 ± 0.00**	0.99 ± 0.02	0.98 ± 0.03	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Gordon	RF	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	**1.00 ± 0.00**	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02
	kNN	0.86 ± 0.03	0.96 ± 0.03	0.98 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	**1.00 ± 0.00**	**1.00 ± 0.00**	0.98 ± 0.03	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.95 ± 0.05	0.99 ± 0.01	0.99 ± 0.01	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	0.98 ± 0.01	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	NB	0.59 ± 0.14	0.78 ± 0.18	0.81 ± 0.19	0.83 ± 0.17	0.83 ± 0.17	**0.86 ± 0.13**	**0.86 ± 0.13**	0.61 ± 0.09	0.85 ± 0.13	0.82 ± 0.20	0.85 ± 0.13	0.77 ± 0.13
	SVC	0.53 ± 0.38	0.80 ± 0.16	0.80 ± 0.16	0.83 ± 0.17	0.83 ± 0.17	0.83 ± 0.13	0.83 ± 0.13	0.84 ± 0.14	**0.88 ± 0.17**	0.84 ± 0.18	**0.88 ± 0.17**	0.85 ± 0.16
Alon	RF	0.69 ± 0.28	0.81 ± 0.16	0.81 ± 0.16	0.81 ± 0.16	0.84 ± 0.16	0.81 ± 0.17	**0.87 ± 0.11**	0.65 ± 0.26	0.77 ± 0.15	0.79 ± 0.16	0.76 ± 0.30	0.80 ± 0.16
	kNN	0.10 ± 0.30	0.72 ± 0.29	0.76 ± 0.15	0.73 ± 0.13	0.80 ± 0.18	0.40 ± 0.35	0.82 ± 0.17	0.74 ± 0.15	**0.97 ± 0.10**	**0.97 ± 0.10**	**0.97 ± 0.10**	**0.97 ± 0.10**
	mean	0.48 ± 0.22	0.77 ± 0.04	0.79 ± 0.02	0.80 ± 0.04	0.82 ± 0.02	0.72 ± 0.19	0.84 ± 0.02	0.71 ± 0.09	**0.87 ± 0.07**	0.86 ± 0.07	0.86 ± 0.07	0.85 ± 0.07
	NB	0.46 ± 0.32	0.64 ± 0.28	0.81 ± 0.29	0.86 ± 0.18	0.77 ± 0.23	**0.98 ± 0.06**	0.74 ± 0.31	0.67 ± 0.13	0.79 ± 0.30	0.81 ± 0.14	0.84 ± 0.14	0.85 ± 0.16
	SVC	0.00 ± 0.00	0.66 ± 0.24	0.80 ± 0.31	0.66 ± 0.30	0.46 ± 0.31	**0.98 ± 0.06**	0.76 ± 0.29	0.67 ± 0.29	0.92 ± 0.17	0.91 ± 0.14	0.88 ± 0.15	0.87 ± 0.31
Pomeroy	RF	0.05 ± 0.15	0.66 ± 0.24	0.72 ± 0.29	0.65 ± 0.27	0.61 ± 0.36	0.72 ± 0.30	**0.73 ± 0.28**	0.52 ± 0.37	0.53 ± 0.38	0.57 ± 0.33	**0.73 ± 0.30**	0.65 ± 0.29
	kNN	0.05 ± 0.15	0.27 ± 0.33	0.79 ± 0.29	0.65 ± 0.33	0.07 ± 0.20	0.81 ± 0.30	0.73 ± 0.28	0.81 ± 0.21	0.98 ± 0.06	**1.00 ± 0.00**	0.97 ± 0.10	**1.00 ± 0.00**
	mean	0.14 ± 0.19	0.56 ± 0.17	0.78 ± 0.03	0.71 ± 0.09	0.48 ± 0.26	**0.87 ± 0.11**	0.74 ± 0.01	0.66 ± 0.10	0.80 ± 0.17	0.82 ± 0.16	0.85 ± 0.09	0.84 ± 0.12
	NB	0.71 ± 0.08	0.94 ± 0.10	0.94 ± 0.09	0.95 ± 0.09	0.93 ± 0.08	0.95 ± 0.09	0.95 ± 0.06	0.80 ± 0.12	0.94 ± 0.09	0.94 ± 0.09	0.95 ± 0.08	**0.97 ± 0.08**
	SVC	0.88 ± 0.10	0.94 ± 0.09	0.93 ± 0.08	0.95 ± 0.06	0.95 ± 0.09	0.96 ± 0.08	0.97 ± 0.06	0.91 ± 0.10	0.96 ± 0.09	0.98 ± 0.04	0.94 ± 0.08	**0.98 ± 0.04**
Singh	RF	0.92 ± 0.09	0.95 ± 0.09	0.93 ± 0.08	**0.96 ± 0.09**	0.95 ± 0.09	**0.96 ± 0.09**	0.94 ± 0.09	0.93 ± 0.08	0.95 ± 0.09	0.94 ± 0.09	0.95 ± 0.09	0.95 ± 0.09
	kNN	0.82 ± 0.10	0.95 ± 0.09	0.94 ± 0.09	0.95 ± 0.06	0.94 ± 0.09	0.98 ± 0.05	0.98 ± 0.05	0.91 ± 0.10	**1.00 ± 0.00**	**1.00 ± 0.00**	0.98 ± 0.05	**1.00 ± 0.00**
	mean	0.83 ± 0.08	0.94 ± 0.01	0.93 ± 0.01	0.95 ± 0.00	0.94 ± 0.01	0.96 ± 0.01	0.96 ± 0.02	0.89 ± 0.05	0.96 ± 0.02	0.96 ± 0.03	0.96 ± 0.02	**0.97 ± 0.02**
WTL										4/4/1	2/6/1	5/3/1	4/3/2

Table 5
Influence of feature selection methods on F_1 score for nine high-dimensional datasets. Classification problems. Selected 60 features.

		noFS	reliefF	f-score	mRMR	chi2	l1-SVM	RkNN	kNN wrapper	WkNN-FS (l_2,exp)	WkNN-FS (l_2,rbf)	WkNN-FS (l_δ,exp)	WkNN-FS (l_δ,rbf)
	NB	0.79 ± 0.20	0.77 ± 0.12	0.90 ± 0.11	0.96 ± 0.06	0.86 ± 0.12	0.94 ± 0.10	0.99 ± 0.04	0.87 ± 0.10	**1.00 ± 0.00**	0.98 ± 0.04	**1.00 ± 0.00**	**1.00 ± 0.00**
	SVC	0.84 ± 0.23	0.95 ± 0.07	0.96 ± 0.06	0.97 ± 0.06	0.97 ± 0.06	0.99 ± 0.04	**1.00 ± 0.00**	0.95 ± 0.06	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Burczynski	RF	0.89 ± 0.15	0.90 ± 0.10	0.94 ± 0.09	0.97 ± 0.06	0.93 ± 0.11	0.91 ± 0.13	0.96 ± 0.07	0.88 ± 0.15	**1.00 ± 0.00**	0.99 ± 0.04	0.99 ± 0.04	0.99 ± 0.04
	kNN	0.69 ± 0.19	0.76 ± 0.31	0.93 ± 0.06	0.92 ± 0.13	0.77 ± 0.32	0.99 ± 0.04	0.90 ± 0.18	0.92 ± 0.09	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.80 ± 0.07	0.84 ± 0.08	0.93 ± 0.02	0.96 ± 0.02	0.88 ± 0.08	0.96 ± 0.03	0.96 ± 0.03	0.90 ± 0.03	**1.00 ± 0.00**	0.99 ± 0.01	**1.00 ± 0.01**	**1.00 ± 0.01**
	NB	0.89 ± 0.06	0.91 ± 0.07	0.92 ± 0.06	0.92 ± 0.05	0.91 ± 0.07	0.95 ± 0.05	0.96 ± 0.04	0.64 ± 0.17	**0.98 ± 0.04**	0.93 ± 0.05	**0.98** ± 0.04	0.93 ± 0.04
	SVC	0.89 ± 0.06	0.92 ± 0.05	0.93 ± 0.04	0.93 ± 0.05	0.93 ± 0.04	**0.99 ± 0.03**	0.95 ± 0.05	0.92 ± 0.06	0.98 ± 0.04	0.96 ± 0.05	0.98 ± 0.04	0.98 ± 0.03
Chin	RF	0.92 ± 0.05	0.91 ± 0.05	0.93 ± 0.05	0.93 ± 0.05	0.93 ± 0.05	0.93 ± 0.05	0.93 ± 0.05	0.92 ± 0.07	0.94 ± 0.04	0.93 ± 0.05	**0.95 ± 0.04**	0.93 ± 0.06
	kNN	0.87 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.05	0.92 ± 0.06	0.93 ± 0.05	0.92 ± 0.05	0.99 ± 0.03	0.99 ± 0.02	0.98 ± 0.04	**1.00 ± 0.00**
	mean	0.89 ± 0.02	0.91 ± 0.01	0.92 ± 0.01	0.93 ± 0.01	0.92 ± 0.01	0.95 ± 0.03	0.94 ± 0.01	0.85 ± 0.12	**0.97 ± 0.02**	0.95 ± 0.03	**0.97 ± 0.01**	0.96 ± 0.03
	NB	0.85 ± 0.06	0.90 ± 0.05	0.88 ± 0.06	0.92 ± 0.07	0.88 ± 0.07	**0.96 ± 0.04**	0.91 ± 0.04	0.54 ± 0.16	**0.96 ± 0.03**	0.92 ± 0.05	0.92 ± 0.04	0.92 ± 0.02
	SVC	0.88 ± 0.01	0.93 ± 0.02	0.94 ± 0.03	0.93 ± 0.04	0.92 ± 0.04	**0.98 ± 0.02**	0.93 ± 0.04	0.91 ± 0.01	0.96 ± 0.03	0.87 ± 0.05	0.96 ± 0.03	0.93 ± 0.03
Tian	RF	0.88 ± 0.01	0.90 ± 0.02	0.90 ± 0.04	**0.92 ± 0.04**	**0.92 ± 0.04**	0.90 ± 0.03	0.91 ± 0.03	0.89 ± 0.02	0.91 ± 0.02	0.89 ± 0.02	0.91 ± 0.03	0.89 ± 0.01
	kNN	0.88 ± 0.01	0.89 ± 0.02	0.90 ± 0.02	0.90 ± 0.02	0.88 ± 0.01	0.88 ± 0.01	0.88 ± 0.01	0.90 ± 0.05	0.99 ± 0.01	**1.00 ± 0.01**	0.97 ± 0.02	**1.00 ± 0.01**
	mean	0.88 ± 0.02	0.91 ± 0.01	0.91 ± 0.02	0.91 ± 0.01	0.90 ± 0.02	0.93 ± 0.04	0.91 ± 0.02	0.81 ± 0.16	**0.95 ± 0.03**	0.92 ± 0.05	0.94 ± 0.02	0.93 ± 0.04
	NB	0.88 ± 0.10	0.97 ± 0.05	0.97 ± 0.05	**0.98 ± 0.06**	0.97 ± 0.07	0.94 ± 0.08	0.97 ± 0.05	0.35 ± 0.27	0.95 ± 0.07	0.97 ± 0.07	0.95 ± 0.07	0.97 ± 0.07
	SVC	0.95 ± 0.07	0.97 ± 0.05	0.97 ± 0.05	0.98 ± 0.04	0.97 ± 0.05	0.98 ± 0.06	0.97 ± 0.05	0.22 ± 0.32	0.97 ± 0.07	0.98 ± 0.04	0.97 ± 0.07	**0.99 ± 0.04**
Chowdary	RF	0.95 ± 0.08	**0.97 ± 0.05**	0.96 ± 0.08	**0.97 ± 0.05**	0.96 ± 0.06	**0.97 ± 0.05**	**0.97 ± 0.05**	0.95 ± 0.07	**0.97 ± 0.05**	**0.97 ± 0.05**	**0.97 ± 0.05**	**0.97 ± 0.05**
	kNN	0.85 ± 0.14	0.96 ± 0.08	0.97 ± 0.05	0.96 ± 0.07	0.83 ± 0.13	**1.00 ± 0.00**	0.96 ± 0.08	0.93 ± 0.10	0.99 ± 0.03	0.99 ± 0.03	0.99 ± 0.03	**1.00 ± 0.00**
	mean	0.91 ± 0.04	0.97 ± 0.01	0.97 ± 0.01	0.97 ± 0.01	0.93 ± 0.06	0.97 ± 0.02	0.97 ± 0.01	0.62 ± 0.33	0.97 ± 0.01	**0.98 ± 0.01**	0.97 ± 0.01	**0.98 ± 0.01**
	NB	0.98 ± 0.06	0.96 ± 0.09	0.94 ± 0.10	0.96 ± 0.09	0.96 ± 0.09	0.91 ± 0.15	**1.00 ± 0.00**	0.83 ± 0.11	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	SVC	0.62 ± 0.34	0.97 ± 0.07	0.97 ± 0.07	0.95 ± 0.08	0.97 ± 0.07	0.97 ± 0.07	0.98 ± 0.06	0.80 ± 0.16	0.98 ± 0.06	0.98 ± 0.06	0.98 ± 0.06	0.98 ± 0.06
Golub	RF	**0.98 ± 0.06**	**0.98 ± 0.06**	**0.98 ± 0.06**	0.96 ± 0.08	0.96 ± 0.08	0.95 ± 0.11	0.96 ± 0.08	0.91 ± 0.14	**0.98 ± 0.06**	**0.98 ± 0.06**	**0.98 ± 0.06**	**0.98 ± 0.06**
	kNN	0.32 ± 0.32	0.91 ± 0.14	0.93 ± 0.16	0.90 ± 0.17	0.90 ± 0.17	0.64 ± 0.26	0.98 ± 0.06	0.89 ± 0.20	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.72 ± 0.28	0.95 ± 0.02	0.95 ± 0.02	0.94 ± 0.03	0.94 ± 0.03	0.86 ± 0.13	0.98 ± 0.02	0.86 ± 0.05	**0.99 ± 0.01**	**0.99 ± 0.01**	**0.99 ± 0.01**	**0.99 ± 0.01**
	NB	0.98 ± 0.04	**1.00 ± 0.00**	0.99 ± 0.02	**1.00 ± 0.00**	**1.00 ± 0.00**	0.97 ± 0.04	**1.00 ± 0.00**	0.97 ± 0.05	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	SVC	0.98 ± 0.04	0.99 ± 0.02	**1.00 ± 0.00**	0.99 ± 0.02	**1.00 ± 0.00**	0.98 ± 0.03	0.99 ± 0.02	0.97 ± 0.03	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
Gordon	RF	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	**1.00 ± 0.00**	0.99 ± 0.02
	kNN	0.86 ± 0.03	0.95 ± 0.05	0.99 ± 0.02	0.99 ± 0.02	0.99 ± 0.02	0.92 ± 0.04	**1.00 ± 0.00**	0.99 ± 0.02	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.95 ± 0.05	0.98 ± 0.02	0.99 ± 0.00	**1.00 ± 0.00**	**1.00 ± 0.00**	0.97 ± 0.03	**1.00 ± 0.00**	0.98 ± 0.01	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	NB	0.59 ± 0.14	0.81 ± 0.19	0.80 ± 0.17	0.84 ± 0.16	0.74 ± 0.22	0.87 ± 0.14	0.83 ± 0.17	0.67 ± 0.67	0.85 ± 0.14	**0.88 ± 0.15**	0.83 ± 0.12	0.72 ± 0.18
	SVC	0.53 ± 0.38	0.83 ± 0.17	0.71 ± 0.28	0.81 ± 0.16	0.86 ± 0.16	0.83 ± 0.30	0.73 ± 0.29	0.77 ± 0.18	0.74 ± 0.30	**0.88 ± 0.17**	**0.88 ± 0.17**	0.79 ± 0.16
Alon	RF	0.69 ± 0.28	0.81 ± 0.16	0.76 ± 0.15	0.81 ± 0.16	**0.84 ± 0.16**	0.83 ± 0.17	0.81 ± 0.16	0.61 ± 0.26	0.71 ± 0.28	0.76 ± 0.15	0.76 ± 0.30	0.77 ± 0.15
	kNN	0.10 ± 0.30	0.61 ± 0.32	0.76 ± 0.15	0.70 ± 0.28	0.71 ± 0.29	0.33 ± 0.34	0.72 ± 0.29	0.76 ± 0.17	0.97 ± 0.10	**1.00 ± 0.00**	0.97 ± 0.10	0.97 ± 0.10
	mean	0.48 ± 0.22	0.77 ± 0.09	0.76 ± 0.03	0.79 ± 0.06	0.79 ± 0.05	0.71 ± 0.22	0.77 ± 0.05	0.70 ± 0.07	0.82 ± 0.10	**0.88 ± 0.09**	0.86 ± 0.08	0.81 ± 0.09
	NB	0.46 ± 0.32	0.74 ± 0.30	0.75 ± 0.33	**0.87 ± 0.21**	0.68 ± 0.26	0.78 ± 0.30	0.76 ± 0.30	0.60 ± 0.25	0.85 ± 0.16	0.74 ± 0.30	0.81 ± 0.30	0.78 ± 0.31
	SVC	0.00 ± 0.00	0.59 ± 0.31	0.78 ± 0.31	0.75 ± 0.19	0.52 ± 0.29	0.83 ± 0.17	0.73 ± 0.28	0.62 ± 0.24	**0.88 ± 0.15**	0.78 ± 0.30	0.85 ± 0.19	0.85 ± 0.30
Pomeroy	RF	0.05 ± 0.15	0.69 ± 0.26	0.72 ± 0.29	0.65 ± 0.27	0.63 ± 0.35	0.52 ± 0.37	**0.73 ± 0.28**	0.45 ± 0.38	0.66 ± 0.27	0.65 ± 0.24	**0.73 ± 0.30**	0.66 ± 0.27
	kNN	0.05 ± 0.15	0.30 ± 0.38	0.67 ± 0.31	0.76 ± 0.17	0.07 ± 0.20	0.42 ± 0.36	0.75 ± 0.29	0.78 ± 0.20	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**	**1.00 ± 0.00**
	mean	0.14 ± 0.19	0.58 ± 0.17	0.73 ± 0.04	0.76 ± 0.08	0.47 ± 0.24	0.64 ± 0.17	0.74 ± 0.01	0.61 ± 0.12	**0.85 ± 0.12**	0.79 ± 0.13	**0.85 ± 0.10**	0.82 ± 0.12
	NB	0.71 ± 0.08	0.94 ± 0.10	0.93 ± 0.08	0.94 ± 0.09	0.94 ± 0.09	**0.95 ± 0.08**	**0.95 ± 0.09**	0.75 ± 0.10	0.94 ± 0.10	0.93 ± 0.09	0.94 ± 0.09	0.92 ± 0.10
	SVC	0.88 ± 0.10	0.94 ± 0.09	0.94 ± 0.09	0.95 ± 0.06	0.94 ± 0.09	0.95 ± 0.08	0.95 ± 0.06	0.91 ± 0.11	**0.97 ± 0.07**	**0.97 ± 0.05**	0.95 ± 0.07	**0.97 ± 0.05**
Singh	RF	0.92 ± 0.09	0.94 ± 0.09	0.93 ± 0.08	0.94 ± 0.09	0.93 ± 0.08	0.93 ± 0.08	0.94 ± 0.09	0.94 ± 0.09	0.95 ± 0.09	0.95 ± 0.09	**0.96 ± 0.06**	0.95 ± 0.09
	kNN	0.82 ± 0.10	0.94 ± 0.06	0.94 ± 0.09	0.96 ± 0.06	0.95 ± 0.09	0.99 ± 0.03	0.97 ± 0.06	0.95 ± 0.07	0.99 ± 0.03	**1.00 ± 0.00**	0.98 ± 0.05	0.99 ± 0.03
	mean	0.83 ± 0.08	0.94 ± 0.00	0.93 ± 0.00	0.95 ± 0.01	0.94 ± 0.01	0.95 ± 0.02	0.95 ± 0.01	0.89 ± 0.08	**0.96 ± 0.02**	**0.96 ± 0.03**	**0.96 ± 0.02**	**0.96 ± 0.03**
WTL										7/2/0	6/2/1	7/2/0	7/2/0

Table 6
Influence of feature selection methods on MAE score for three high-dimensional datasets. Regression problems. Selected 30 features.

		noFS	reliefF	f-score	mRMR	chi2	l1-SVM	RkNN	kNN wrapper	WkNN-FS (l_2,exp)	WkNN-FS (l_2,rbf)	WkNN-FS (l_δ,exp)	WkNN-FS (l_δ,rbf)
	Lasso	1.10 ± 0.39	0.64 ± 0.19	0.60 ± 0.12	0.57 ± 0.16	0.65 ± 0.14	0.55 ± 0.17	0.47 ± 0.12	0.51 ± 0.13	0.50 ± 0.16	**0.44 ± 0.11**	0.50 ± 0.26	0.57 ± 0.15
	SVR	0.47 ± 0.10	0.72 ± 0.14	0.60 ± 0.13	0.54 ± 0.16	0.56 ± 0.12	0.45 ± 0.12	0.38 ± 0.09	0.39 ± 0.08	0.35 ± 0.10	0.42 ± 0.11	**0.33 ± 0.06**	0.37 ± 0.07
Reg01	RF	0.45 ± 0.11	0.62 ± 0.18	0.60 ± 0.12	0.51 ± 0.14	0.49 ± 0.16	0.48 ± 0.13	0.41 ± 0.09	0.42 ± 0.10	0.37 ± 0.10	0.42 ± 0.09	**0.36 ± 0.11**	0.42 ± 0.10
	kNN	0.78 ± 0.13	0.70 ± 0.18	0.65 ± 0.11	0.72 ± 0.16	0.64 ± 0.17	0.69 ± 0.12	0.43 ± 0.07	0.38 ± 0.10	0.33 ± 0.16	0.41 ± 0.14	**0.32 ± 0.12**	0.37 ± 0.14
	mean	0.70 ± 0.27	0.67 ± 0.04	0.61 ± 0.02	0.58 ± 0.08	0.59 ± 0.06	0.54 ± 0.09	0.42 ± 0.03	0.42 ± 0.05	0.39 ± 0.07	0.42 ± 0.01	**0.38 ± 0.07**	0.43 ± 0.08
	Lasso	0.39 ± 0.10	0.60 ± 0.14	0.41 ± 0.08	0.47 ± 0.20	0.47 ± 0.16	0.34 ± 0.11	0.35 ± 0.10	0.39 ± 0.08	0.33 ± 0.08	**0.24 ± 0.09**	**0.24 ± 0.08**	0.34 ± 0.10
	SVR	0.38 ± 0.14	0.41 ± 0.08	0.43 ± 0.08	0.43 ± 0.14	0.41 ± 0.11	0.30 ± 0.12	0.29 ± 0.06	0.31 ± 0.08	0.30 ± 0.10	**0.25 ± 0.08**	0.26 ± 0.09	0.26 ± 0.08
Reg02	RF	0.31 ± 0.07	0.43 ± 0.10	0.38 ± 0.09	0.31 ± 0.07	0.30 ± 0.05	0.29 ± 0.09	0.25 ± 0.04	0.22 ± 0.07	0.25 ± 0.08	0.22 ± 0.06	0.21 ± 0.07	**0.20 ± 0.08**
	kNN	0.51 ± 0.11	0.57 ± 0.11	0.49 ± 0.08	0.45 ± 0.11	0.44 ± 0.08	0.34 ± 0.08	0.32 ± 0.10	0.18 ± 0.03	0.20 ± 0.08	**0.15 ± 0.06**	0.17 ± 0.06	0.17 ± 0.07
	mean	0.40 ± 0.07	0.50 ± 0.08	0.43 ± 0.04	0.42 ± 0.06	0.41 ± 0.07	0.32 ± 0.03	0.30 ± 0.04	0.27 ± 0.08	0.27 ± 0.05	**0.22 ± 0.04**	**0.22 ± 0.03**	0.24 ± 0.06
	Lasso	1.12 ± 0.24	0.53 ± 0.10	0.64 ± 0.11	0.54 ± 0.07	0.65 ± 0.11	**0.45 ± 0.08**	0.47 ± 0.06	0.63 ± 0.09	0.53 ± 0.08	0.57 ± 0.07	0.53 ± 0.11	0.56 ± 0.10
	SVR	0.54 ± 0.08	0.54 ± 0.10	0.65 ± 0.11	0.53 ± 0.08	0.64 ± 0.11	0.43 ± 0.07	0.44 ± 0.07	0.59 ± 0.08	0.43 ± 0.08	0.52 ± 0.07	**0.40 ± 0.05**	0.51 ± 0.06
Reg03	RF	0.57 ± 0.06	0.58 ± 0.06	0.64 ± 0.11	0.53 ± 0.08	0.64 ± 0.11	**0.50 ± 0.06**	**0.50 ± 0.08**	0.56 ± 0.11	0.54 ± 0.10	0.55 ± 0.08	0.51 ± 0.07	0.54 ± 0.08
	kNN	0.61 ± 0.06	0.58 ± 0.07	0.70 ± 0.07	0.56 ± 0.04	0.70 ± 0.07	0.48 ± 0.04	0.47 ± 0.04	0.52 ± 0.08	0.42 ± 0.05	0.39 ± 0.09	**0.38 ± 0.05**	0.40 ± 0.07
	mean	0.71 ± 0.24	0.56 ± 0.02	0.66 ± 0.03	0.54 ± 0.02	0.66 ± 0.02	0.47 ± 0.02	0.47 ± 0.03	0.57 ± 0.04	0.48 ± 0.05	0.51 ± 0.07	**0.46 ± 0.07**	0.50 ± 0.06
WTL										1/1/1	1/1/1	3/0/0	1/0/2

Table 7
Influence of feature selection methods on MAE score for three high-dimensional datasets. Regression problems. Selected 60 features.

		noFS	reliefF	f-score	mRMR	chi2	l1-SVM	RkNN	kNN wrapper	WkNN-FS (l_2,exp)	WkNN-FS (l_2,rbf)	WkNN-FS (l_δ,exp)	WkNN-FS (l_δ,rbf)
	Lasso	1.10 ± 0.39	0.61 ± 0.16	0.62 ± 0.12	0.55 ± 0.19	0.68 ± 0.20	0.59 ± 0.14	0.51 ± 0.13	0.55 ± 0.14	0.51 ± 0.18	0.51 ± 0.15	**0.48 ± 0.16**	0.54 ± 0.14
	SVR	0.47 ± 0.10	0.63 ± 0.16	0.59 ± 0.12	0.47 ± 0.17	0.55 ± 0.14	0.47 ± 0.09	0.39 ± 0.07	0.41 ± 0.10	**0.32 ± 0.10**	0.38 ± 0.08	0.34 ± 0.06	0.39 ± 0.09
Reg01	RF	0.45 ± 0.11	0.61 ± 0.15	0.59 ± 0.11	0.43 ± 0.13	0.49 ± 0.15	0.45 ± 0.10	0.41 ± 0.09	0.42 ± 0.09	**0.38 ± 0.11**	0.42 ± 0.11	**0.38 ± 0.11**	0.41 ± 0.11
	kNN	0.78 ± 0.13	0.65 ± 0.16	0.59 ± 0.13	0.69 ± 0.17	0.68 ± 0.15	0.66 ± 0.10	0.50 ± 0.10	0.39 ± 0.10	0.29 ± 0.14	0.30 ± 0.12	**0.26 ± 0.12**	0.39 ± 0.13
	mean	0.70 ± 0.27	0.62 ± 0.02	0.60 ± 0.01	0.53 ± 0.10	0.60 ± 0.08	0.54 ± 0.08	0.45 ± 0.05	0.44 ± 0.08	0.37 ± 0.08	0.40 ± 0.07	**0.36 ± 0.08**	0.43 ± 0.06
	Lasso	0.39 ± 0.10	0.47 ± 0.09	0.49 ± 0.21	0.59 ± 0.31	0.54 ± 0.16	0.39 ± 0.18	0.35 ± 0.09	0.40 ± 0.06	0.35 ± 0.13	0.26 ± 0.06	**0.25 ± 0.09**	0.33 ± 0.14
	SVR	0.38 ± 0.14	0.40 ± 0.09	0.40 ± 0.06	0.43 ± 0.12	0.44 ± 0.09	0.32 ± 0.11	0.30 ± 0.09	0.33 ± 0.10	0.29 ± 0.11	**0.26 ± 0.09**	0.27 ± 0.08	0.29 ± 0.09
Reg02	RF	0.31 ± 0.07	0.44 ± 0.09	0.33 ± 0.07	0.33 ± 0.06	0.31 ± 0.05	0.30 ± 0.08	0.26 ± 0.06	0.22 ± 0.07	0.23 ± 0.08	0.21 ± 0.09	0.21 ± 0.07	**0.19 ± 0.06**
	kNN	0.51 ± 0.11	0.57 ± 0.07	0.44 ± 0.07	0.47 ± 0.11	0.45 ± 0.08	0.39 ± 0.09	0.33 ± 0.09	0.19 ± 0.05	0.14 ± 0.06	0.16 ± 0.06	**0.13 ± 0.06**	0.17 ± 0.06
	mean	0.40 ± 0.07	0.47 ± 0.06	0.42 ± 0.06	0.46 ± 0.09	0.44 ± 0.08	0.35 ± 0.04	0.31 ± 0.03	0.29 ± 0.08	0.25 ± 0.08	**0.22 ± 0.04**	**0.22 ± 0.05**	0.25 ± 0.07
	Lasso	1.12 ± 0.24	0.55 ± 0.08	0.62 ± 0.12	0.54 ± 0.10	0.64 ± 0.11	0.51 ± 0.06	**0.47 ± 0.06**	0.64 ± 0.12	0.52 ± 0.12	0.59 ± 0.08	0.50 ± 0.06	0.56 ± 0.13
	SVR	0.54 ± 0.08	0.57 ± 0.08	0.62 ± 0.12	0.49 ± 0.09	0.64 ± 0.11	0.43 ± 0.08	0.42 ± 0.06	0.55 ± 0.06	0.43 ± 0.07	0.52 ± 0.06	**0.39 ± 0.05**	0.49 ± 0.06
Reg03	RF	0.57 ± 0.06	0.58 ± 0.07	0.65 ± 0.12	0.53 ± 0.07	0.64 ± 0.11	**0.51 ± 0.06**	0.51 ± 0.07	0.54 ± 0.12	0.53 ± 0.09	0.57 ± 0.09	0.52 ± 0.07	0.54 ± 0.08
	kNN	0.61 ± 0.06	0.56 ± 0.08	0.68 ± 0.15	0.57 ± 0.05	0.70 ± 0.07	0.51 ± 0.05	0.48 ± 0.06	0.51 ± 0.10	0.37 ± 0.06	0.37 ± 0.10	**0.34 ± 0.05**	0.42 ± 0.10
	mean	0.71 ± 0.24	0.56 ± 0.01	0.64 ± 0.03	0.53 ± 0.03	0.66 ± 0.03	0.49 ± 0.03	0.47 ± 0.03	0.56 ± 0.05	0.46 ± 0.07	0.51 ± 0.09	**0.44 ± 0.07**	0.50 ± 0.06
WTL										3/0/0	2/0/1	3/0/0	2/0/1

kNN-based FS method, WkNN-FS (l_2,rbf) and WkNN-FS (l_δ,rbf). These methods were able to identify more than one half of the relevant features in the most challenging MadelonHD dataset, which is better than the best results achieved by the state-of-the-art algorithms, f-score, reliefF, and mRMR. The fourth best performing method is again another implementation of WkNN-FS, in particular WkNN-FS (l_2,exp), yielding the highest *Suc*. score on the Madelon datasets. We can conclude that on artificial datasets, the performance of all forms of the proposed WkNN-FS is the same as or better than that of the compared state-of-the-art methods.

For high sample datasets (denoted 5k) the results are more balanced. The availability of higher number of samples allowed algorithms to better learn the underlying pattern in data so the *Suc*. rates are higher than *Suc*. rates for small sample datasets. All WkNN-FS together with kNN-wrapper were able to successfully determine all relevant features.

Regarding the parameters of WkNN, based on the presented results the choice of the loss function and distance evaluation function appears to influence the performance of the proposed FS method. The methods utilizing the square loss function were more successful in terms of identifying the relevant features for small sample datasets (Reg, Fri, Mad, MHD); however, the difference is only 10%. However, independently of the parameter choice the WkNN-FS algorithm performed better than the other compared methods.

3.3. Influence of weighted k-nearest neighbors feature selection on prediction performance on real-world datasets

The *Suc*. rate presented in the previous section gives good insight into the performance of the FS techniques. Basically, the goal of FS is to choose the relevant, and only the relevant, features. Another important aspect of FS is the influence of the selection of a subset on the prediction performance of the classifier.

Since the majority of datasets for classification problems discussed in this section are datasets with a class imbalance, we used the F_1 score to measure the prediction performance of the classifier. The F_1 score summarizes the balance between precision and recall and is defined as $F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$. Additionally to F_1 we calculated also the accuracy of prediction. These results are provided in Online supplementary material.

The prediction performance on regression datasets is measured using mean absolute error (MAE) and root mean squared error (RMSE). Both metrics can range from zero to ∞ and are indifferent to direction errors. RMSE has property of penalizing large errors more. In the paper we show results for MAE metric, the results for RMSE are provided in Online supplementary material.

Synthetic data have the advantage of being generated according to some clearly defined rule. This is also their major limitation: since the data are synthetically generated they do not possess the natural flaws of real data. Unfortunately, in real data we cannot definitely and with confidence declare which features are relevant for class label prediction and which are not. Therefore, for the real data we evaluated the prediction performance in terms of the F_1 score and MAE. Since FS is frequently used as a preprocessing step in high-dimensional scenarios, we selected twelve high-dimensional publicly available datasets. The description of these is given in Table 3, together with relevant data resources. Three of these datasets constitute the regression problem. Nine are binary (two class classification tasks) or were converted to binary format according to the rule described in a particular source paper. We believe that the results are also applicable for multi-class scenarios, since multi-class datasets can be decomposed into multiple binary classification tasks through the divide and conquer approach.

As the predictor, we employed four algorithms, each based on a different underlying concept. The four utilized classifiers were the Gaussian naive Bayes (NB) (replaced by Lasso regressor for regression tasks), SVM classifier/regression (SVC/SVR) with RBF kernel, the ensemble predictor Random Forest (RF) with decision trees as the base estimators, and kNN ($k = 21$). The underlying algorithms are described well in the literature; see, e.g., [6,45].

To evaluate the F_1 score and MAE, each method was used to select the 30 and 60 most important features that were fed to the classifier input. We used 10-fold stratified cross-validation to validate the results.

3.3.1. Prediction performance results

In order to evaluate the performance of the proposed algorithms, again seven FS methods, f-score, ReliefF, mRMR, chi2, l1-SVM, RkNN, and kNN-wrapper were compared. The popular machine learning methods kNN, SVM, RF, and NB/Lasso, implemented in the *scikit-learn* module, were used in this study to induce predictors. For each dataset, we ran all eleven FS algorithms and obtained the F_1 scores for classification and MAE for regression tasks. The F_1 prediction scores are presented in Table 4 for 30 selected features and in Table 5 for 60 selected features. The results are graphically compared in Figs. 1 and 2. We selected always the highest F_1 score from four classifiers (NB, SVC, RF, kNN). The results for regression tasks are summarized in Table 6 for the best 30 features and in Table 7 for extended set of 60 features. We provide the individual results for each predictor for all twelve datasets, together with the *mean* F_1 and MAE score as the average of the four applied predictors for every dataset. Moreover, WTL (win/tie/loss) represents the number of datasets for which the mean F_1 score (respectively MAE for regression problems) (average of the kNN, SVM, RF, and NB/Lasso classifier scores) of the predictors induced by the corresponding WkNN-FS method is higher than (or equal to or lower than) that of the predictor induced based on the feature sets selected by the best classical FS method. Only mean scores are included in WTL statistics, not results of individual predictors. Note that the WkNN-FS methods are handicapped in this comparison, since the result of a particular WkNN is compared always with the best performing classical FS method (reliefF, f-score, or mRMR, chi2, l1-SVM, RkNN, and kNN-wrapper) on a particular dataset.

Table 4 shows that all versions of WkNN-FS outperform conventional methods. This is clearly demonstrated by the WTL statistics, where the most successful WkNN-FS(l_δ,exp) is mostly the same as or better than any conventional FS method. The only exception is Pomeroy dataset where it performs slightly worse than l1-SVM method. This is in alignment with the results obtained on artificial data, where WkNN-FS methods exhibit higher detection rates. However, WkNN-FS(l_δ,exp) demonstrated lower *Suc*. rates than the other WkNN-FS methods, although the differences are not very notable. The second most successful WkNN-FS(l_2,exp) also shows the best performance on the artificial data. It outperformed the conventional methods on four datasets, tied with them on four, and performed worse than they did on only one dataset. However, closer investigation reveals that the dataset on which WkNN-FS(l_2,exp) performed worse than the conventional method (l1-SVM in this case) is the Pomeroy dataset. The low mean F_1 score is the result of the low performance of the RF classifier in conjunction with WkNN-FS(l_2,exp) on this dataset. This is in contrast with the results of the kNN classifier, the induction of which yields result competitive to result of l1-SVM. Therefore, even in this case, WkNN(l_2,exp) is able to achieve the same score as conventional methods.

Increasing the number of selected features to 60 even further enhance the advantage of WkNN over conventional methods. The results are shown in Table 5. Only in one case (Tian dataset) l1-SVM outperforms WkNN(l_2,rbf), otherwise the proposed WkNN algorithms yield same or better F_1 score than conventional methods.

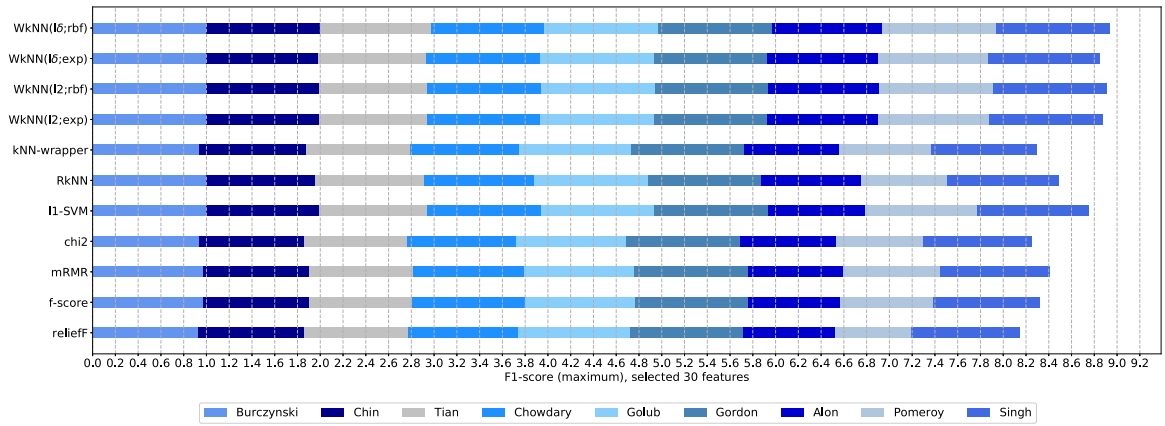

Fig. 1. Influence of different FS methods on max F_1 score on high-dimensional datasets. The highest F_1 score of the four classifiers is taken. Selected 30 features.

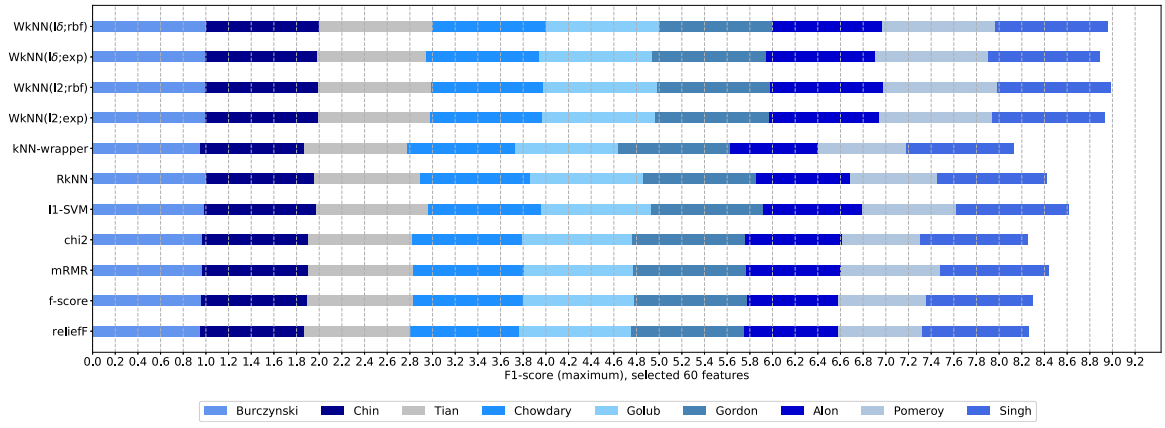

Fig. 2. Influence of different FS methods on max F_1 score on high-dimensional datasets. The highest F_1 score of the four classifiers is taken. Selected 60 features.

We performed similar experiments on three regression tasks from Table 3. We used the Lasso instead of NB classifier for regression tasks. MAE for all FS methods using 30 most relevant features is summarized in Table 6. For this three datasets, results are balanced with RkNN, kNN-wrapper, WkNN(l_2,exp), WkNN(l_2,rbf), and WkNN(l_δ,rbf) achieving very similar results. The other conventional methods provide higher error rates. On the other hand, the WkNN (l_δ,exp) outperforms all other methods on all three datasets.

Similarly, as in the case of classification tasks, increasing number of selected features to 60 helped WkNN methods. WkNN(l_2,exp), WkNN(l_2,rbf), and WkNN(l_δ,rbf) perform slightly better than RkNN and kNN-wrapper.

To obtain statistical validation of the results in our experiments, we used Friedman test and Bonferroni–Dunn post-hoc test as recommended in [46]. We compared performance of methods on datasets from Table 3. The null hypothesis (there is no difference between FS methods) can be rejected after Friedman test. The p-values and z-scores of Bonferroni–Dunn post-hoc test for comparison of WkNN methods with other methods are presented in Table 8. The z-value was used to calculate the corresponding p-value from the table of normal distribution $\mathcal{N}(0, 1)$. The adjusted p-value was then calculated for 4*7 tests. For comparison of WkNNs and kNN-wrapper, reliefF, chi2, and f-score we can safely reject the null hypothesis (there is no difference between two methods) with significance level $\alpha = 0.05$. For WkNN(l_δ,exp) we can also reject null hypothesis for the case of 60 selected features in comparison with l1-SVM and mRMR. The RkNN did not show up as statistically different from WkNN methods, but if we consider average rank for 30 selected features: kNN-wrapper (9.3750), reliefF (9.2500), chi2 (8.7500), f-score (8.4167), mRMR (7.3333), RkNN (5.2917), l1-SVM (4.8333), WkNN(l_2,rbf) (3.8750), WkNN(l_δ,rbf) (3.5833), WkNN(l_2,exp), (3.0000), WkNN-FS(l_δ,exp) (2.2917) and for 60 selected features: kNN-wrapper (9.2917), reliefF (8.9167), chi2 (8.7917), f-score (8.1667), l1-SVM (7.0417), mRMR (6.6250), RkNN (5.7500), WkNN(l_2,rbf) (3.2917), WkNN(l_δ,rbf) (3.2083), WkNN(l_2,exp) (2.6250), WkNN-FS(l_δ,exp) (2.2917) we can see that RkNN ranks behind the WkNN methods.

In general, all the newly proposed methods demonstrated a better performance in terms of the F_1 and MAE score than the conventional FS methods evaluated in this study. The choice of distance evaluation function $w(d) = e^{-d}$ provided a higher F_1 score and MAE in most cases. As in the case of synthetic data, we can notice that the choice of the loss and the distance evaluation function influenced the performance of WkNN-FS to some extent, although not crucially, and all the considered implementations of WkNN-FS demonstrated very competitive results. The best prediction results were achieved by WkNN(l_δ,exp) algorithm.

4. Conclusions

FS is an important dimensionality reduction technique regularly used in many machine learning and pattern recognition application domains. This paper presented a new algorithm for supervised FS, namely, the weighted k-nearest neighbors FS. As its name suggests, it is based on the principle of k-nearest neighbors algorithm and relies on gradient descent to find optimal weights. Numerical experiments were conducted using two types of data: eight synthetically generated datasets and twelve real-world high-dimensional datasets. The experimental results for the synthetic datasets show that WkNN-FS effectively identifies the relevant

Table 8
Bonferroni–Dunn test for the comparison of WkNN-FS with other feature selection methods. Displayed is p-value and z-score in brackets for 30 and 60 selected features.

		kNN wrapper	reliefF	chi2	f-score	l1-SVM	mRMR	RkNN
kNN-FS	30	1.36E−03 (4.06)	2.02E−03 (3.97)	8.9E−03 (3.60)	2.22E−02 (3.35)	1.00 (0.71)	0.30 (2.55)	1.00 (1.05)
(l_2,rbf)	60	2.62E−04 (4.43)	9.13E−04 (4.15)	1.36E−03 (4.06)	8.9E−03 (3.60)	0.16 (2.77)	0.39 (2.46)	1.00 (1.82)
WkNN-FS	30	5.29E−04 (4.28)	7.98E−04 (4.19)	3.8E−03 (3.82)	1.00E−02 (3.57)	1.00 (0.92)	0.16 (2.77)	1.00 (1.26)
(l_δ,rbf)	60	1.97E−04 (4.49)	6.97E−04 (4.22)	1.05E−03 (4.12)	7.00E−03 (3.66)	0.13 (2.83)	0.32 (2.52)	1.00 (1.88)
WkNN-FS	30	7.00E−05 (4.71)	1.10E−04 (4.62)	6.08E−04 (4.25)	1.77E−03 (4.00)	1.00 (1.35)	3.84E−02 (3.20)	1.00 (1.69)
(l_2,exp)	60	2.4E−05 (4.92)	9.4E−05 (4.65)	1.47E−04 (4.55)	1.19E−03 (4.09)	3.10E−02 (3.26)	8.78E−02 (2.95)	0.59 (2.31)
WkNN-FS	30	5E−06 (5.23)	8E−06 (5.14)	5.2E−05 (4.77)	1.7E−04 (4.52)	1.00 (1.87)	5.50E−03 (3.72)	0.75 (2.22)
(l_δ,exp)	60	7E−06 (5.17)	2.8E−05 (4.89)	4.4E−05 (4.80)	4.01E−04 (4.34)	1.26E−02 (3.51)	3.84E−05 (3.20)	0.30 (2.55)

features. The results for the high-dimensional real-world datasets prove that WkNN-FS can effectively reduce dimensionality and obtain the best average detection rate in terms of the F_1 score and MAE.

This study showed that WkNN-FS is an effective FS method; however, it offers several open possibilities for further research. Here, we considered only the two alternatives of loss function and distance evaluation function, but there are other options that could be considered that may further boost the performance of the algorithm. In addition, only the Euclidean distance was considered, but for some data other distance functions may be more suitable. Moreover, the flexible implementation of WkNN-FS makes it a very good candidate for application to imbalanced data. Clever adjustment of the weighting vector can contribute to the successful utilization of the modified WkNN-FS for imbalanced data and to the still quite unexplored area of FS for imbalanced data. FS is an important area of research and therefore there are many additional directions in which further research can be continued.

Acknowledgment

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-16-0211.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.knosys.2018.10.004.

References

[1] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, Recent advances and emerging challenges of feature selection in the context of big data, Knowl.-Based Syst. 86 (Supplement C) (2015) 33–45, http://dx.doi.org/10.1016/j.knosys.2015.05.014, URL http://www.sciencedirect.com/science/article/pii/S0950705115002002.

[2] K. Yu, X. Wu, W. Ding, J. Pei, Scalable and accurate online feature selection for big data, ACM Trans. Knowl. Discov. Data 11 (2) (2016) http://dx.doi.org/10.1145/2976744, 16:1–16:39.

[3] L. Gao, J. Song, X. Liu, J. Shao, J. Liu, J. Shao, Learning in high-dimensional multimedia data: the state of the art, Multimedia Syst 23 (3) (2017) 303–313, http://dx.doi.org/10.1007/s00530-015-0494-1.

[4] D. Dembélé, A flexible microarray data simulation model, Microarrays 2 (2) (2013) 115–130, http://dx.doi.org/10.3390/microarrays2020115, URL http://www.mdpi.com/2076-3905/2/2/115.

[5] G. Hughes, On the mean accuracy of statistical pattern recognizers, IEEE Trans. Inform. Theory 14 (1) (1968) 55–63, http://dx.doi.org/10.1109/TIT.1968.1054102.

[6] J.H. Friedman, R. Tibshirani, T. Hastie, The Elements of Statistical Learning, Springer, USA, 2009.

[7] I. Guyon, S. Gunn, M. Nikravesh, L. Zadeh, Feature Extraction, Foundations and Applications, Springer Heidelberg, 2006.

[8] I. Jolliffe, Principal Component Analysis, Springer-Verlag, New York, USA, 1986.

[9] S.T. Roweis, L.K. Saul, Nonlinear dimensionality reduction by locally linear embedding, Science 290 (5500) (2000) 2323–2326, http://dx.doi.org/10.1126/science.290.5500.2323, arXiv:http://science.sciencemag.org/content/290/5500/2323.full.pdf, URL http://science.sciencemag.org/content/290/5500/2323.

[10] A.M. Martinez, A.C. Kak, Pca versus lda, IEEE Trans. Pattern Anal. Mach. Intell. 23 (2) (2001) 228–233, http://dx.doi.org/10.1109/34.908974.

[11] X. Zhao, F. Nie, S. Wang, J. Guo, P. Xu, X. Chen, Unsupervised 2d dimensionality reduction with adaptive structure learning, Neural Comput. 29 (5) (2017) 1352–1374.

[12] J. Yan, B. Zhang, N. Liu, S. Yan, Q. Cheng, W. Fan, Q. Yang, W. Xi, Z. Chen, Effective and efficient dimensionality reduction for large-scale and streaming data preprocessing, IEEE Trans. Knowl. Data Eng. 18 (3) (2006) 320–333, http://dx.doi.org/10.1109/TKDE.2006.45.

[13] P. Drotar, J. Gazda, Z. Smekal, An experimental comparison of feature selection methods on two-class biomedical datasets, Comput. Biol. Med. 66 (2015) 1–10, http://dx.doi.org/10.1016/j.compbiomed.2015.08.010, URL http://www.sciencedirect.com/science/article/pii/S0010482515002917.

[14] H. Peng, F. Long, C. Ding, Feature selection based on mutual information: criteria of max-dependency, max-relevance, and min-redundancy, IEEE Trans. Pattern Anal. Mach. Intell. 27 (8) (2005) 1226–1239.

[15] K. Kira, L.A. Rendell, A practical approach to feature selection, in: Proceedings of the ninth international workshop on Machine learning, in: ML92, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992, pp. 249–256.

[16] J.C. Ang, A. Mirzal, H. Haron, H.N.A. Hamed, Supervised, unsupervised, and semi-supervised feature selection: a review on gene selection, IEEE/ACM Trans. Comput. Biol. Bioinform. 13 (5) (2016) 971–989, http://dx.doi.org/10.1109/TCBB.2015.2478454.

[17] B. Xue, M. Zhang, W.N. Browne, X. Yao, A survey on evolutionary computation approaches to feature selection, IEEE Trans. Evol. Comput. 20 (4) (2016) 606–626, http://dx.doi.org/10.1109/TEVC.2015.2504420.

[18] Y. Li, T. Li, H. Liu, Recent advances in feature selection and its applications, Knowl. Inf. Syst. 53 (3) (2017) 551–577, http://dx.doi.org/10.1007/s10115-017-1059-8.

[19] B. Seijo-Pardo, I. Porto-Daz, V. Boln-Canedo, A. Alonso-Betanzos, Ensemble feature selection: Homogeneous and heterogeneous approaches, Knowl.-Based Syst. 118 (2017) 124–139, http://dx.doi.org/10.1016/j.knosys.2016.11.017, URL http://www.sciencedirect.com/science/article/pii/S0950705116304749.

[20] N. Spolaor, M.C. Monard, G. Tsoumakas, H.D. Lee, A systematic review of multi-label feature selection and a new method based on label construction, Neurocomputing 180 (2016) 3–15.

[21] P.Y. Lee, W.P. Loh, J.F. Chin, Feature selection in multimedia: The state-of-the-art review, Image Vis. Comput. 67 (2017) 29–42.

[22] K. Shima, N. pour Hossein, N. Bahareh, Multilabel feature selection: A comprehensive review and guiding experiments, Wiley Interdiscip. Rev.: Data Min. Knowl. Discovery 8 (2018) e1240, http://dx.doi.org/10.1002/widm.1240.

[23] S. Li, E.J. Harner, D.A. Adjeroh, Random knn feature selection - a fast and stable alternative to random forests, BMC Bioinformatics 12 (1) (2011) 450, http://dx.doi.org/10.1186/1471-2105-12-450.

[24] C.H. Park, S.B. Kim, Sequential random k-nearest neighbor feature selection for high-dimensional data, Expert Syst. Appl. 42 (5) (2015) 2336–2342, http://dx.doi.org/10.1016/j.eswa.2014.10.044, URL http://www.sciencedirect.com/science/article/pii/S095741741400668X.

[25] A. Wang, N. An, G. Chen, L. Li, G. Alterovitz, Accelerating wrapper-based feature selection with K-nearest-neighbor, Knowl.-Based Syst. 83 (2015) 81–91, http://dx.doi.org/10.1016/j.knosys.2015.03.009, URL http://www.sciencedirect.com/science/article/pii/S0950705115001033.

[26] A. Navot, L. Shpigelman, N. Tishby, E. Vaadia, Nearest neighbor based feature selection for regression and its application to neural activity, in: Advances in Neural Information Processing Systems, Vol. 18, MIT Press, 2006, pp. 995–1002.

[27] D. Hand, H. Mannila, P. Smyth, Principles of Data Mining, MIT Press, Cambridge, USA, 2001.

[28] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, J. Artif. Int. Res. 6 (1) (1997) 1–34.

[29] M. Robnik-Šikonja, I. Kononenko, Theoretical and empirical analysis of Relieff and RReliefF, Mach. Learn. 53 (1) (2003) 23–69, http://dx.doi.org/10.1023/A:1025667309714.

[30] B. Weir, Estimating f-statistics: A historical view, Philos. sci. 79 (5) (2012) 637–643.

[31] I. Guyon, A. Elisseeff, An introduction to variable and feature selection, J. Mach. Learn. Res. 13 (7) (2015) e1002195, http://dx.doi.org/10.1371/journal.pbio.1002195.

[32] U. Alon, N. Barkai, D. Notterman, K. Gish, S. Ybarra, D. Mack, A. Levine, Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays, Proc. Natl. Acad. Sci. 96 (12) (1999) 6745–6750.
[33] M.E. Burczynski, R.L. Peterson, N.C. Twine, K.A. Zuberek, B.J. Brodeur, L. Casciotti, V. Maganti, P.S. Reddy, A. Strahs, F. Immermann, W. Spinelli, U. Schwertschlag, A.M. Slager, M.M. Cotreau, A.J. Dorner, Molecular classification of Crohn's disease and ulcerative colitis patients using transcriptional profiles in peripheral blood mononuclear cells, J. Mol. Diagnostics 8 (1) (2006) 51–61.
[34] D. Chowdary, J. Lathrop, J. Skelton, K. Curtin, T. Briggs, Y. Zhang, J. Yu, Y. Wang, A. Mazumder, Prognostic gene expression signatures can be measured in tissues collected in RNAlater preservative, J. Mol. Diagnostics 8 (1) (2006) 31–39.
[35] K. Chin, S. DeVries, J. Fridlyand, P.T. Spellman, R. Roydasgupta, W.-L. Kuo, A. Lapuk, R.M. Neve, Z. Qian, T. Ryder, F. Chen, H. Feiler, T. Tokuyasu, C. Kingsley, S. Dairkee, Z. Meng, K. Chew, D. Pinkel, A. Jain, B.M. Ljung, L. Esserman, D.G. Albertson, F.M. Waldman, J.W. Gray, Genomic and transcriptional aberrations linked to breast cancer pathophysiologies, Cancer Cell 10 (6) (2006) 529–541.
[36] T.R. Golub, D.K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, H. Coller, M.L. Loh, J.R. Downing, M.A. Caligiuri, C.D. Bloomfield, E.S. Lander, Molecular classification of cancer: class discovery and class prediction by gene expression monitoring, Science 286 (5439) (1999) 531–537.
[37] G.J.G. Gordon, R.V.R. Jensen, L.-L.L. Hsiao, S.R.S. Gullans, J.E.J. Blumenstock, S.S. Ramaswamy, W.G.W. Richards, D.J.D. Sugarbaker, R.R. Bueno, Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and Mesothelioma, Cancer Res. 62 (17) (2002) 4963–4967.
[38] S.L. Pomeroy, P. Tamayo, M. Gaasenbeek, L.M. Sturla, M. Angelo, M.E. McLaughlin, J.Y.H. Kim, L.C. Goumnerova, P.M. Black, C. Lau, J.C. Allen, D. Zagzag, J.M. Olson, T. Curran, C. Wetmore, J.A. Biegel, T. Poggio, S. Mukherjee, R. Rifkin, A. Califano, G. Stolovitzky, D.N. Louis, J.P. Mesirov, E.S. Lander, T.R. Golub, Prediction of central nervous system embryonal tumour outcome based on gene expression, Nature 415 (6870) (2002) 436–442.
[39] Y.N. Singh, S.K. Singh, A.K. Ray, Bioelectrical signals as emerging biometrics: Issues and challenges, ISRN Signal Process. 2012 (1) (2012) 136–151, http://dx.doi.org/10.5402/2012/712032.
[40] E. Tian, F. Zhan, R. Walker, E. Rasmussen, Y. Ma, B. Barlogie, J.D. Shaughnessy Jr., The role of the Wnt-signaling antagonist DKK1 in the development of osteolytic lesions in multiple Myeloma, New Engl. J. Med. 349 (26) (2003) 2483–2494.
[41] O. Demir-Kavuk, H. Riedesel, E.-W. Knapp, Exploring classification strategies with the CoEPrA 2006 contest, Bioinformatics 26 (2010) 603–609, http://dx.doi.org/10.1093/bioinformatics/btq021.
[42] Coepra 2006: Comparative evaluation of prediction algorithms, http://www.coepra.org (Accessed: 19.05.18).
[43] V. Bolón-Canedo, N. Sánchez-Maroño, A. Alonso-Betanzos, A review of feature selection methods on synthetic data, Knowl. Inf. Syst. 34 (3) (2013) 483–519.
[44] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, E. Duchesnay, Scikit-learn: Machine learning in Python, J. Mach. Learn. Res. 12 (2011) 2825–2830.
[45] R.O. Duda, P.E. Hart, D.G. Stork, Patern Classification, Willey, New York, USA, 2000.
[46] J. Demšar, Statistical comparisons ofclassifiers over multiple data sets, J. Mach. Learn. Res. 7 (2006) 1–30.