Journal Pre-proof

EKNN: Ensemble Classifier Incorporating Connectivity and Density into *k*NN with Application to Cancer Diagnosis<!–<ForCover>Mahfouz MA, Shoukry A, Ismail MA, EKNN: Ensemble Classifier Incorporating Connectivity and Density into *k*NN with Application to Cancer Diagnosis, *Artificial Intelligence In Medicine*, doi: 10.1016/j.artmed.2020.101985</ForCover>–>

Mohamed A. Mahfouz, Amin Shoukry, Mohamed A. Ismail

PII: S0933-3657(20)31250-1

DOI: https://doi.org/10.1016/j.artmed.2020.101985

Reference: ARTMED 101985

To appear in: *Artificial Intelligence In Medicine*

Received Date: 14 December 2019

Revised Date: 2 November 2020

Accepted Date: 2 November 2020

Please cite this article as: { doi: https://doi.org/

EKNN: Ensemble Classifier Incorporating Connectivity and Density into *k*NN with Application to Cancer Diagnosis

Mohamed A. Mahfouz*[1], Amin Shoukry [1, 2], Mohamed A. Ismail[1]

[1]*Department of Computer and Systems Engineering, Faculty of Engineering, Alexandria University, Egypt*

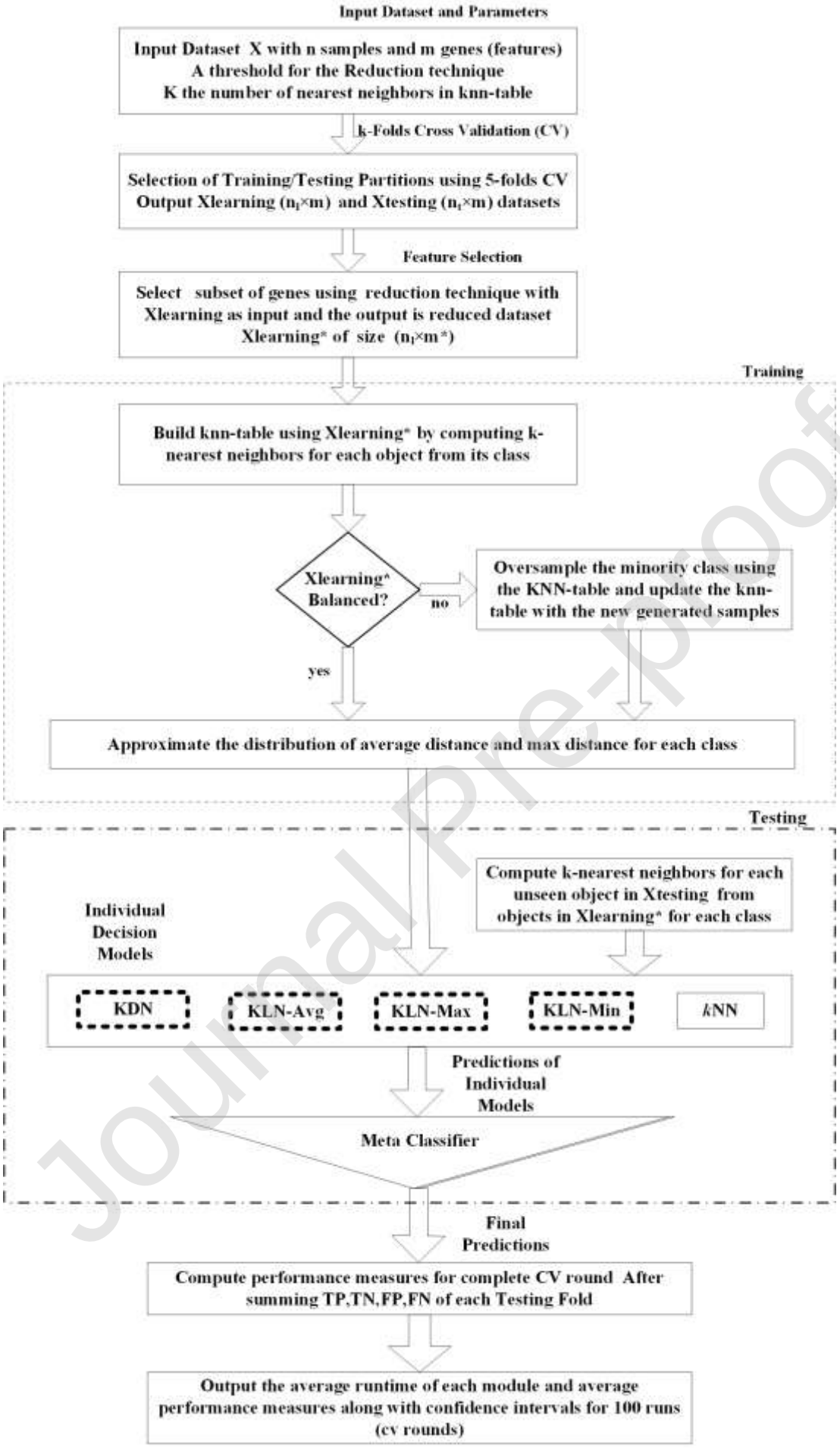
[2]*Computer Science and Engineering Dept., Egypt Japan, University of Science and Technology, Alexandria, Egypt*

[1]*E-Mails: m.a.mahfouz@gmail.com, drmaismail@gmail.com*

[2]*E-Mail:amin.shoukry@ejust.edu.eg*

Graphical abstract

1- In this paper parameters K and *k* play different roles (see Table 1).



Highlights

- This research study proposed ensemble classifier that includes four base classifiers relative to the traditional *k*NN.
- The proposed decision models are based on the density, single linkage, average linkage and complete linkage distributions of the dataset under study.
- The estimated distributions of the average linkage and complete linkage of each class are shown to be more spread out than those of other related approach in terms of the Bhattacharyya coefficient.
- Our experiments show that the proposed decision models have uncorrelated opinions. The smaller the dataset the higher the increase in performance over *k*NN.
- The use of K nearest table reduces the classification time required by the density-based decision model
- EKNN has been tested on five cancer datasets and another seven standard datasets. An improved performance has been achieved compared to several reported results of gene-based cancer detection techniques.
- The proposed ensemble targets small datasets however it can be slightly modified to work on large datasets or stream data.
- The proposed ensemble can work as a gene selector with a genetic algorithm instead of *k*NN in several related works.

Abstract

In the microarray-based approach for automated cancer diagnosis, the application of the traditional *k*-nearest neighbors *k*NN algorithm suffers from several difficulties such as the large number of genes (high dimensionality of the feature space) with many irrelevant genes (noise) relative to the small number of available samples and the imbalance in the size of the samples of the target classes. This research provides an ensemble classifier based on decision models derived from *k*NN that is applicable to problems characterized by imbalanced small size datasets. The proposed classification method is an ensemble of the traditional *k*NN algorithm and four novel classification models derived from it. The proposed models exploit the increase in density and connectivity using K^1-nearest neighbors table (KNN-table) created during the training phase. In the density model, an unseen sample *u* is classified as belonging to a class t if it achieves the highest increase in density when this sample is added to it i.e. the unseen sample can replace more neighbors in the KNN-table for samples of class t than other classes. In the other three connectivity models, the mean and standard deviation of the distribution of the average, minimum as well the maximum distance to the K neighbors of the members of each class are computed in the training phase. The class t to which *u* achieves the highest possibility of belongness to its distribution is chosen, i.e. the addition of *u* to the samples of this class produces the least change to the distribution of the corresponding decision model for class t. Combining the predicted results of the four individual models along with traditional *k*NN makes the decision space more discriminative.

With the help of the KNN-table which can be updated online in the training phase, an improved performance has been achieved compared to the traditional *k*NN algorithm with slight increase in classification time. The proposed ensemble method achieves significant increase in accuracy compared to the accuracy achieved using any of its base classifiers on Kentridge, GDS3257, Notterman, Leukemia and CNS datasets. The method is also compared to several existing ensemble methods and state of the art techniques using different dimensionality reduction techniques on several standard datasets. The results prove clear superiority of EKNN over several individual and ensemble classifiers regardless of the choice of the gene selection strategy.

Keywords: Cancer Diagnosis, Ensemble Classification, Gene Expression Analysis, Nearest Neighbors.

1. Introduction

High costs of patient data collection and complexity of the experiments often make research studies originating from single medical centres of limited samples size (small datasets) such as in microarray-based cancer diagnosis. Small datasets also exist in many other research studies in machine learning such as medical image processing, diseases diagnosis and prognosis, and outcome prediction. Such small datasets are usually imbalanced and of high dimensionality [1]. As a nonparametric classifier, *k*NN is highly influenced by the outliers in small datasets. The focus of this work is on developing an ensemble classifier based on the traditional *k*NN and individual decision models related to *k*NN that is applicable to problems characterized by small noisy imbalanced datasets without increasing the classification time.

The *k*-nearest neighbors (*k*NN) algorithm is a simple instance-based learning algorithm. Given an unseen sample *u*, the algorithm finds the *k* closest samples to *u* according to a distance metric, then *u* is assigned to the class that has the maximum number of samples in the *k* closest neighborhood of *u* (plurality voting). From this definition, it is clear that *k*NN is easy to implement, can deal with datasets having more than two classes and imposes no assumption on the input data distribution. However, traditional *k*NN lacks the scalability to manage very large datasets. Also, the *k*NN rule needs to identify the *k*-neighborhood and when it is applied to imbalanced data, there is a tendency to assign an unseen sample to the majority class label. Additionally, it is known that the plurality rule is sub-optimal when the number of labels is large and the number of examples is small. Many research studies tried to tackle the above problems facing the traditional *k*NN classifier [2]-[3, 4].

In this paper, an ensemble of *k*-nearest neighbors-based decision models (EKNN) are proposed and applied to cancer diagnosis using gene expression data. Four novel decision models namely KDN, KLN-Avg, KLN-Min and KLN-Max are proposed beside the traditional *k*NN. In the training phase, a KNN-table that holds a list of K nearest neighbors, for each sample, is maintained and statistics based on it are computed for each class. In the testing phase, both the computed statistics and the information stored in the KNN-table are used to compute the decisions of the proposed models.

Our experimental study shows that the proposed EKNN ensemble outperforms the traditional *k*NN classifier. It is able to accurately divide the samples into their respective classes by considering selected genes. While the proposed scheme increases the complexity of the training phase compared to the traditional *k*NN, the complexity of the testing phase is comparable to traditional *k*NN due to the use of the KNN-table. The training and classification times of the proposed decision models and traditional *k*NN can be highly reduced using the k-d tree data structure. The KNN-table can also help dealing with imbalanced data via up-sampling. Results show that the combined decision of various models turns out to be better compared to individual decision models. We have experimentally validated that EKNN is less sensitive to input parameters and deals better with noise than a single excellent base classifier as claimed in [5]. The higher the diversity among the base classifiers, the better the performance is [6]. The diversity between the proposed decision models seems to be data dependent but our experiments show that the proposed decision models have uncorrelated opinions especially when the performance of *k*NN degrades in case of small datasets. The Bhattacharyya distance [7] is used in this research to get insight into the distance between the assumed normal distributions. Theoretical guidelines on ensemble learning are still limited due to its complexity [8].

In this research, the performance of the proposed algorithm on several standard and cancer datasets is analyzed. The performance of EKNN on five cancer datasets is compared to neural network [9], decision tree [10], naïve Bayes [11], random forest, bagging and adaBoost with *k*NN and decision tree (DT) as the base classifiers, in addition to the traditional *k*NN [12]. Also, EKNN is compared to another two recent approaches for cancer diagnosis TC-VGC [13] and RBG-CD [14].

The remainder of this paper is organized as follows. Section 2 discusses related work. Section 3 describes the notation used in the paper as well as some background material. Section 4 discusses the proposed scheme in detail. In section 5, experimental results are discussed. Finally, Section 6 concludes the paper and highlights future research directions.

2. Related Work

2.1. kNN Relatives

The *k*NN classification rule, which is a simple majority vote, ignores the differences between training samples' qualities and treats all nearest neighbors equally. To deal with this problem, several distance-weighted voting methods have been proposed. In [2], an algorithm termed as (WKNN) has been developed, where larger weights are given to closer nearest neighbors. A fuzzy *k*NN algorithm [15] exploits the fuzzy uncertainty to enhance the performance of the traditional *k*NN algorithm, however it collects its evidence from the *k* nearest neighbors which makes the tuning of *k* tedious. The fuzzy decision rule algorithm (FRNN) [16] combines the ideas of the local information and the possibilistic approach to achieve a better performance in terms of robustness and accuracy. FRNN avoids the problem of choosing the optimal value of *k* by considering all training patterns as neighbors of different degrees derived from fuzzy typicality of the training data.

In addition, a number of adaptive distance metric nearest neighbors classification rules were developed by expressing the test sample as a linear combination of normalized training samples giving farther samples greater importance [17]. In [17], The weighted sum of the *k* nearest neighbors obtained using this method which is termed as (CFKNN) has a potential to approximate the test sample. Also, the popular (PNN) classifier [18] is a local mean-based classifier which attempts to find the pseudo nearest neighbor by combining various distances between the testing sample and its *k* nearest neighbors in each class. In [19], an algorithm termed as (MKNN) is proposed that takes into consideration the neighborhood information of the training samples to judge their closeness to the testing sample and to help making a correct classification decision. Also in [20], an improved nearest neighbor classification algorithm is proposed via fusing neighborhood information to show that the traditional nearest neighbor is one-sided and may be unconvincing. Recently, in [21], the *k*-nearest neighbors are first considered as the neighborhood information of each sample, then the *k*-general nearest neighbors (*k*GNN) of the testing sample are found to make a classification decision through a majority vote. In *k*GNN, the mutual neighborhood information of both the testing and training samples are considered and their overlapping region is used to decide whether the training samples are a *k*GNN of the testing sample. Several distance metrics have been adopted in the above algorithms, and their performance have been evaluated in [22].

In addition, the *k*NN rule usually suffers from outliers [23], [16] and [4], especially in the case of small-size training data. One of the solutions to this problem is the local mean-based *k*-nearest neighbors (LMKNN) classifier, proposed in [23]. LMKNN computes the local mean vector of the *k* nearest neighbors of the testing sample in each class to achieve robustness against outliers and improve the classification accuracy. Another related work is [24] which proposes a replacement of the majority voting system in *k*NN by combining the local mean based nearest neighbor (LMKNN) [23] and distance weighted *k*-nearest neighbors (DWKNN) [3] methods. The merging of these two methods proved to be able to improve the accuracy of the classification process. In the testing phase, LMKNN uses the closest distance to each local mean vector (center of the nearest neighbors) of each data class, which is considered effective in overcoming the negative effect of outliers [21].

A relevant study to the contribution proposed in this paper is [25]. In [25], a simple *k*NN rule termed MinKL that takes into account the labels of all of the neighbors rather than just the most common label. MinKL shows an improvement over *k*NN in several cases especially when the number of labels is large and the number of samples is small. In the proposed ensemble, the use of several rules like the MinKL make it more robust and accurate than MinKL. It should be noted that, similar to the proposed EKNN, the value of K in MinKL, LMKNN, and DWKNN is different from the original *k*NN, where the value of *k* is the number of nearest neighbors from all training data, while in these approaches the value of K is the number of nearest neighbors from each class in the training data [21].

2.2. Feature selection and classification algorithms for microarray data

An efficient and effective algorithm is necessary to extract and select features from microarray data that improve a classifier performance. Beside statistical methods for reducing the dimensionality such as principal component analysis (PCA) which is based on orthogonal projection [26], there are two main categories of commonly used techniques for reducing the high dimensionality of microarray data: filters [27] and wrappers [28]. Filter methods, such as *t*–test and signal-noise-rate (SNR) , prioritize genes according to one or more predefined metric and select the top-ranked genes. Wrapper methods have also been widely used to select feature genes from microarray data. They generate alternative feature gene subsets and select the subset with the highest classification accuracy. Genetic algorithm (GA) is combined with *k*NN in [29] to identify discriminative genes.

Other approaches for cancer diagnosis are based on computing distinguishing gene–gene pairs using correlation coefficients as in TC-VGC algorithm [13] or by computing distinguishing biclusters as in RBG-CD algorithm [14]. Both TC-VGC and RBG-CD are sensitive to their input parameters and require extensive parameter tuning but they do not need the dimension reduction step. The general conclusion is that no single classifier excel in all kinds of datasets and the dimension reduction step plays an important role in cancer diagnosis.

Convolutional neural network (CNN) is a popular NN architecture for deep learning and is frequently used for feature extraction and classification [30], [31], [32], [33], [34]. The ability of CNN to learn the characteristics of the input data reduces the required processing compared to other classification algorithms. However, CNN suffers high computational cost

and needs a lot of training data [32]. When the number of classes is big and the training data size is too small, CNN is difficult to train with high precision.

In the leak detection model proposed in [32], clustering is used to reduce the number of classes by grouping the pipes based on pressure and flow. The cluster number of the pipelines was used as the category label.

2.3. Related Ensemble methods

Several Ensembles in literature are constructed and tailored for dealing with microarray data. In [35], multiple kernels of Support Vector Machines (MK-SVM) are used to transform the problem of feature selection into a multiple parameters learning problem then a tree-like algorithm is used to extract the classification rules from the obtained support vectors. In [36], an ensemble classifier is constructed through the learning of different SVM kernels, like linear, polynomial, radial basis function (RBF), and sigmoid. The predicted results of individual models are combined through majority voting. In [37], an ensemble of decision tree classifiers is constructed on different subsets of data where kernel principal component KPCA transformation is applied to each of these subsets. The use of fine-tuned KPCA (with RBF as a kernel) allows the produced ensemble to deal with linearly inseparable data and outperforms the state-of-the-art Random Forest (RF) classifier and several other existing methods on their experimental data.

The ensemble approach is shown to be useful in dealing with small datasets as in [1], where a large number of identical neural networks in terms of their topology and neuron functions are used for prediction. Unlike our ensemble approach, the highest performing instance of the optimal neural network design in [1] is selected as the working model. Compared to the above ensembles, EKNN is simple.

2.4. Methods for dealing with Imbalanced input data

The performance of *k*NN is affected by the existing of imbalance in the input data since the majority class tends to have high classification accuracy, while the minority class tends to have low classification accuracy. Several existing methods rely on adjusting the score of unseen samples in the testing phase by giving a higher weight for minority classes [38]. However, this approach is very sensitive to its input parameter. Another approach is resizing the input dataset, resizing is done by either under-sampling the majority class or over-sampling the minority class or by doing both at the same time (hybrid) [39] until the data is balanced. The strategy that achieved the highest performance in [40] is to under-sample the negative (majority) class samples that are close to all positive samples. It is worth to note that tree ensemble classifiers such as random forests can perform well on imbalanced data without up or down sampling.

Another approach for dealing with imbalanced data is the use of a suitable performance metric. Weighted TPR-TNR [41] is a single valued metric to assess classifiers' performance that takes into account imbalanced datasets. The experimental results given in [41] show that TPR-TNR is able to rank classifiers more accurately than the methods that formulate the ranking problem as a decision making problem to be solved using any MCDM method where classifiers are the alternatives and available metrics are the criteria [42]. Using weighted metrics proved to be reliable in the case of imbalanced data or when the misclassification associated costs are different. Also, penalized learning algorithms (or cost-sensitive training) that increase the cost of classification mistakes on the minority class [37] are able to produce more reliable results.

3. PRELIMINARIES

This section provides supporting material to help the reader better understand the remaining sections. Commonly used abbreviations and symbols are listed in Table 1.

Table 1 Abbreviations and symbols used in the text

Abbreviation	Description
*k*NN	traditional k-nearest neighbors Algorithm
KDN	proposed decision model based on Density
KLN-Avg	proposed decision model based on average linkage
KLN-Max	proposed decision model based on complete linkage
KLN-Min	proposed decision model based on single linkage
EKNN	The proposed ensemble Classification Scheme
X	dataset comprising *n samples* × *m genes*
T	target label vector t=(t_1,t_2,...,t_n) corresponding to *n* samples in **X**;
X^t	dataset comprising samples of class t in **X**
n^t	number of samples in $\mathbf{X}^t$
N	total number of samples
C	total number of classes (2 for binary classes)
$I(g_i, g_j)$	absolute Pearson correlation between feature vectors representing expressions of genes g_i and g_j
$d(s_i, s_j)$	Pearson Correlation coefficient's Distance between samples s_i and s_j
x_{ij}	j^{th} feature value of the i^{th} sample in the dataset X
KNN-table	A table of size $n \times k$ cells, contains indices of K nearest neighbors fand their distances for each sample
k	number of nearest neighbors used in *kNN*
K	number of nearest neighbors from each class that are kept in the KNN-table for EKNN
$N_t(x)$	K nearest neighbors of sample *x* belonging to class *t*
α_i	KLN-Avg parameter for computing score of class *i*
β_i	KLN-Min parameter for computing score of class *i*
η_i	KLN-Max parameter for computing score of class *i*
δ	threshold used in KDN model to allow adding a score for a close but not nearest sample

3.1. Similarity between two probability distributions

The Bhattacharyya distance [7] is used in this research to measure the distance between two different probability distributions. The distance between two normal distributions $N_p(\mu_p, \sigma_p)$ and $N_q(\mu_q, \sigma_q)$ is computed as follows:

$$d = \frac{1}{4} ln\left(\frac{1}{4}\left(\frac{\sigma_p^2}{\sigma_q^2} + \frac{\sigma_q^2}{\sigma_p^2} + 2\right)\right) + \frac{1}{4}\left(\frac{\left(\mu_p - \mu_q\right)^2}{\sigma_p^2 + \sigma_q^2}\right) \quad (1)$$

Where μ_p, σ_p and μ_q, σ_q are the mean and the standard deviation of the *p*-th and *q*-th normal distributions, respectively. The Bhattacharyya distance is adopted to measure the similarity between the estimated distribution of each class using KLN-Avg, KLN-Min and KLN-Max for different choices of K. The measured distances are compared to the distances between the discrete probability distributions of each class by the MinKL algorithm [25].

The Bhattacharyya distance between two discrete probability distributions *p* and *q* over the same domain X is computed as follows:

$$d = -ln\left(\sum_{x \in X} \sqrt{p(x)q(x)}\right) \quad (2)$$

The distance d lies in $[0,\infty]$ for both the discrete and continuous cases.

3.2. Gene Expression Datasets

Gene expression is measured using DNA microarray technology by comparing gene expression of one cell inhibited under certain condition (sample c) with those of another cell maintained in normal condition (sample n) [10]. In cancer diagnosis, each sample in the input dataset is labeled either as malignant or normal in case of binary classification or otherwise is given a cancer grade. One gene expression means one feature value in the feature vector. Likewise, one feature column has multiple gene expression of a single gene on all samples. Entries of a gene expression matrix are real numbers representing gene expression of each sample.

Table 2 Gene Expression Datasets

Dataset Cancer	Ref.	type	normal/ malignant	total samples	no. features
Kentridge (Colon)	[43]	unbalanced	22/40	62	2000
GDS3257 (Lung)	[44]	unbalanced	58/49	107	2517
Notterman (Colon)	[45]	balanced	18/18	36	7457
Leukemia (Blood)	[46]	unbalanced	25/47	72	7129
CNS (Nervous)	[47]	unbalanced	21/39	60	7110

The major difficulty in the analysis of this type of data is the large number of genes (high dimensionality of the feature space) with many irrelevant genes (noise) compared to a very small number of samples. Besides, the dataset is usually imbalanced i.e. the majority class tends to have much more samples. Examples of such data are the datasets listed in Table 2 which are used in our experimental study. For these very small datasets, the traditional validation scheme in which the data is divided into fixed training and testing sets is not practical since there is no enough data to partition. Section 3.4 discusses cross validation as a solution for this problem. Also, Kentridge, Leukemia and CNS datasets are slightly imbalanced. As shown in Table 2, all listed gene expression datasets have high dimensionality ranging from 2000 to 7457 for Kentridge and Notterman datasets, respectively. The original GDS3257 (Lung) dataset has been processed using *Babelomic* tool [48] and after the filtering steps, it consists of 2517 genes before applying the dimension reduction step.

3.3. Similarity between two samples

The Euclidean, cityblock, cosine and correlation distances are the commonly used metrics with the *k*NN algorithms [22]. The most common similarity measures in the field of bioinformatics are those based on the correlation coefficient. Most of them have corresponding dissimilarity measures. The Pearson's correlation coefficient of two random variables x and y is formally defined as follows:

$$s(x,y) = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i - \bar{x}}{\sigma_x}\right)\left(\frac{y_i - \bar{y}}{\sigma_y}\right) \quad (3)$$

Where $\bar{x}, \bar{y}$ are the mean of x_i and y_i, and $\boldsymbol{\sigma}_x$, $\boldsymbol{\sigma}_y$ are their standard deviations, respectively. It is a measure of how well a straight line can be fitted to a scatter plot of *x* and *y*.

The absolute Pearson's correlation coefficient is used as a similarity metric between gene pairs in the dimension reduction step and the corresponding Pearson's distance, given in Eq. (4), measures the distance between sample pairs in both the training and classification steps:

$$d(x,y) = 1 - ABS(s(x,y)) \quad (4)$$

As the Pearson correlation coefficient falls in [-1, 1], the Pearson distance and absolute correlation lies in [0, 1].

3.4. Cross Validation (CV)

When solving a prediction problem, the resulting prediction model should be able to generalize to an independent dataset (unknown testing samples) i.e. accurately generalizes in practice. The prediction model is usually validated by partitioning the input data into two partitions (e.g. 70% for training and 30% for test). However, in our problem, the number of samples is very small and there are no enough samples to partition without losing significant modeling and testing capabilities. In this work, k-folds Cross-Validation scheme (CV) [49] is applied in testing the proposed models. The input dataset is divided into f partitions (folds), f -1 partitions participate in training, and the classes of the instances

belonging to the remaining partition are predicted by the decision model based on the training performed on the f-1 training partitions. This process is repeated k times to form a complete CV round after which the class of each sample is identified.

3.5. Performance Evaluation Metrics

The effectiveness of EKNN has been evaluated using well known performance metrics such as accuracy, sensitivity, and specificity [49] which involve TP and TN, the number of correctly classified positive and negative samples, respectively, as well as FP and FN, the number of incorrectly classified positive and negative samples, respectively.

Accuracy measures the overall effectiveness of the classification scheme. It is calculated as follows:

$$accuracy = \frac{TP + TN}{TP + FP + TN + FN} \quad (5)$$

Sensitivity measures the ability of a classifier to recognize patterns from the positive class. It can be obtained using the following equation:

$$sensitivity = \frac{TP}{TP + FN} \quad (6)$$

Specificity measures the ability of a classifier to recognize patterns from the negative class. The following equation is used to calculate specificity:

$$specificity = \frac{TN}{TN + FP} \quad (7)$$

In cancer diagnosis, sensitivity is more important than specificity since it shows how much a classifier is able to correctly identifies patients with cancer which may be treatable at this time but not later (e.g. cervical cancer). In general, a well-balanced sensitivity and specificity is accepted. AUC (Area under ROC curve) is a performance metric that combines both sensitivity and specificity. (1 - Specificity) vs. Sensitivity is plotted and the area under the curve is compared to the area under another ROC curve. AUC is between 0 and 1, the higher AUC is, the better is the performance. (1-Specificity) refers to the false positive rate. While accuracy is computed based on certain threshold, AUC can be computed for several thresholds at the same time. A good classifier should have both low false positive rate and low false negative rate.

From the definition of accuracy, high accuracy means low (FP+FN)/*n* so when there is a big difference between the operational FP and FN misclassification costs, or between the operational class frequencies compared to those in the training set then AUC is a better indicator for the performance than accuracy. Also, we used the balanced accuracy score available in [50]. It is defined as the average of recall obtained on each class. It is more meaningful in the case of imbalanced dataset.

In our experimental study the reported performance measure is the average over 30 runs. Some performance measures are reported with a 95% confidence interval to show the reliability of our estimates. After each 5-folds cross validation round the TP, TN, FP and FN of all folds are summed to get a single confusion matrix then any desired performance metric can be computed for this round.

4. Proposed Classification Scheme

The proposed EKNN scheme is a combination of feature selection strategy coupled with ensemble classification. Fig. 1 represents a top-level architecture of the EKNN, and the following sections explain its different phases in detail. The bold dashed rectangles, in fig. 1, represent our contributions.

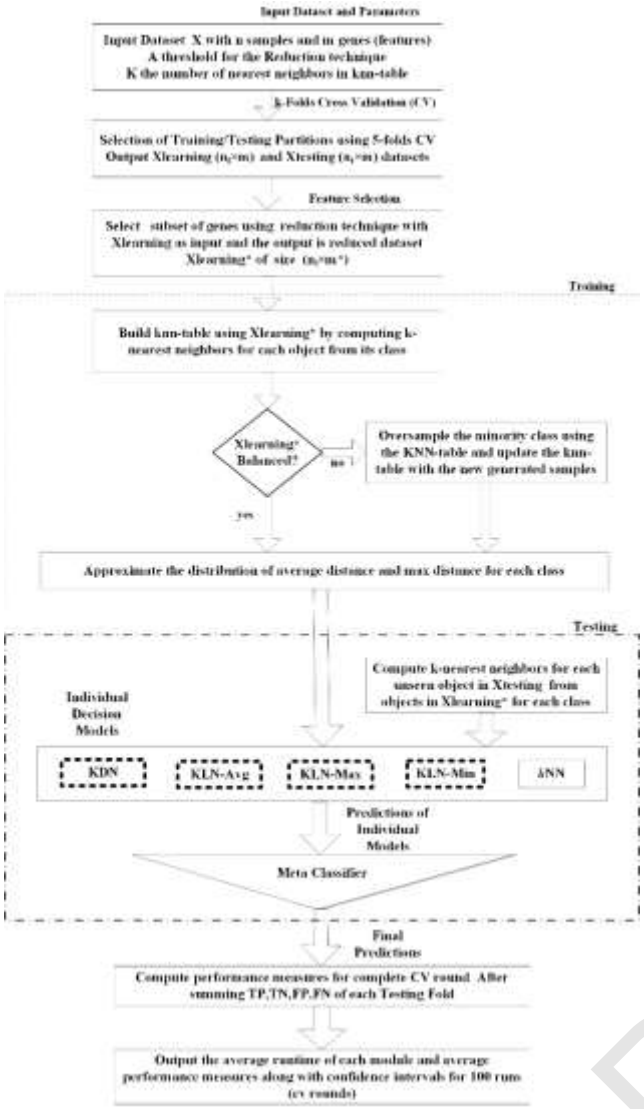


Fig. 1. Top Level Layout of the proposed scheme

Unlike traditional *k*NN algorithm in which no work is done in the training phase, in EKNN, the K-nearest neighbors table (KNN-table) is created during the training time and the necessary statistics for the proposed decision model are computed. If the input dataset is imbalanced the KNN-table can be used for identifying border samples and oversampling the minority class. Building the KNN-table helps reducing the computational complexity of the proposed decisions models as explained in the next sections. The proposed four decision models are the main contributions of this research study. As shown in Fig. 1, after selecting the training set as Xlearning and the testing set as Xtesting using 5-folds cross validation, the dimension reduction technique is applied on Xlearning to produce the reduced training set Xlearning* having the same number of samples as Xlearning but with a reduced set of features (genes in cancer diagnosis problem). Then the KNN-table is created using Xlearning*. If Xlearning* is balanced then testing can be applied, otherwise oversampling of the minority class is done on Xlearning*. Moreover, the KNN-table is updated by the newly generated samples. In order to reduce the impact of imbalanced data on the accuracy of *k*NN and the other proposed decision models, we choose the solution proposed in [51], termed as borderline-SMOTE2, it is also implemented in [50]. Information required by Borderline-SMOTE2 to oversample the minority class can be retrieved while building the KNN-table. It starts by identifying border samples of the minority class. After computing the *k* nearest neighbors of each sample in the minority class, the samples having all of their *k* nearest neighbors from the majority class are considered noise. Core samples are the samples that have less than *k*/2 nearest neighbors from majority class. Remaining samples are considered border samples. After identifying the set of border samples B, each time there is a need to generate a synthetic example, one random sample b_j in B is selected and the difference (dif_j) between it and another randomly selected sample m_i of its *m*-nearest neighbors is computed. The new generated sample equals $b_i + r\ dif_j$ where *r* is a random number between 0 and 1 if m_i belongs to the minority class, otherwise *r* lies between 0 and 0.5. Thus, the new generated examples are closer to the minority class. The KNN-table is incrementally updated by the newly generated samples along with the computed statistics. In the testing phase, the *k*-nearest neighbors of the unseen sample from each class are identified. Four decision models termed as KDN, KLN-Min, KLN-Avg and KLN-Max are proposed, in addition to the traditional *k*NN. The final decision is computed from the base decision models using stacking. For each complete round of CV, the performance metrics are computed.

4.1. Building the K-Nearest-Neighbors Table (KNN-table)

For each sample, the nearest K samples from its class are identified and sorted, in descending order, according to the similarity measure from closest to furthest. These nearest K samples can be dynamically updated by the arrival of a new sample in order to have an online classifier. For each new sample we keep its K nearest neighbors from each class along with their distances in order to reduce the computational complexity of the proposed decision model KDN, as described in the following sections. The KNN-table is computed based on Eq. (4). In order to update KNN-table dynamically, when an incoming training sample *u* arrives its similarity with every sample from its class t is computed. An entry is added with its list of top K similar samples. If the incoming sample can replace one of the nearest neighbors of previous sample x_i then the incoming sample is added to $N_t(x_i)$ instead of this farther sample. However, in order to make the proposed classifier EKNN learn from stream of data, we should also compute the required statistics dynamically. With the help of the KNN-table, in order to classify an unseen sample, we can identify neighbors of neighbors of the unseen sample for two or more levels and compute the required statistics based on these few neighbors. Dealing with stream data is out of the scope of the present work.

4.2. Computing Required Statistics from the KNN-table

Let $N_t(x_i)$ be the set of K nearest neighbors of a sample $x_i \in X^t$. Let *kdistmin*(x_i), *kdistmax*(x_i), *kdistavg*(x_i) be the minimum, maximum, and average distances from x_i to the members of $N_t(x_i)$, respectively.

Let $kdistmin_t$, $kdistmax_t$ and $kdistavg_t$ be the mean of the minimum, maximum, and average distances to the K nearest neighbors for each class *t* respectively. They are calculated as the average of *kdistmin*(x_i), *kdistmax*(x_i) and *kdistavg*(x_i) respectively. The average of *kdistmin*(x_i) for class t (for all $x_i \in X^t$) is defined as $kdistmin_t$ and computed as follows:

$$kdistmin_t = \frac{\sum_{x_i \in X^t} kdistmin(x_i)}{|X^t|} = \sum_{x_i \in X^t} \min_{x_j \in X^t} d(x_i, x_j)/|X^t| \qquad (8)$$

The average of *kdistmax*(x_i) for class t (for all $x_i \in X^t$) is defined as $kdistmax_t$ and computed as follows

$$kdistmax_t = \frac{\sum_{x_i \in X^t} kdistmax(x_i)}{|X^t|} = \sum_{x_i \in X^t} \max_{x_j \in X^t} d(x_i, x_j)/|X^t| \qquad (9)$$

The average of *kdistavg*(x_i) for class t (for all $x_i \in X^t$) is defined as $kdistavg_t$ and computed as follows

$$kdistavg_t = \sum_{x_i \in X^t} kdistavg(x_i)/|X^t| = \sum_{x_i \in X^t} (\sum_{x_j \in N_t(x)} d(x_i, x_j)/K)/|X^t| \qquad (10)$$

Assuming that *kdistmin*, *kdistmax* and *kdistavg* across samples in X follow normal distribution, the corresponding standard deviations termed as $kminsigma_t$ $kmaxsigma_t$ and $kavgsigma_t$, respectively, can be computed as follows:

$$kminsigma_t = \sqrt{\frac{1}{|X^t|-1} \sum_{x_i \in X^t} (kdistmin(x_i) - kdistmin_t)^2} \qquad (11)$$

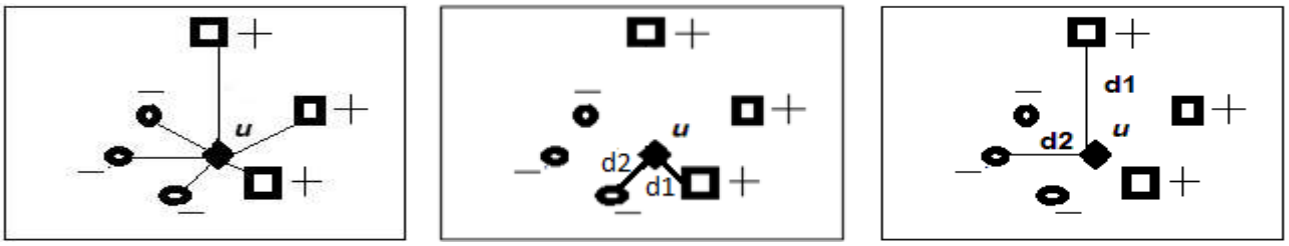
$$kmaxsigma_t = \sqrt{\frac{1}{|X^t|-1} \sum_{x_i \in X^t} (kdistmax(x_i) - kdistmax_t)^2} \qquad (12)$$

$$kavgsigma_t = \sqrt{\frac{1}{|X^t|-1} \sum_{x_i \in X^t} (kdistavg(x_i) - kdistavg_t)^2} \qquad (13)$$

If *kdistmax*(x_i), *kdistmin*(x_i) or *kdistavg*(x_i) for any $x_i \in X^t$ takes an extreme value (i.e. farther than three times the corresponding standard deviation) then x_i is removed from the computation of $kdistmax_t$, $kdistmin_t$ or $kdistavg_t$ respectively. The process may be repeated in order to reduce the effect of outliers on the computations. The rest of the data is used in computing new values for $kavgsigma_t$, $kminsigma_t$, $kmaxsigma_t$, $kdistavg_t$, $kdistmin_t$ and $kdistmax_t$.

4.3. Decision Models

As shown in Fig. 2 a), the average distance from *u* to its K nearest neighbors from each class is considered in computing $kdistavg_{+1}(u)$ and $kdistavg_{-1}(u)$. In Fig 2 b), d1 and d2 represent the $kdistmin_{+1}(u)$, $kdistmin_{-1}(u)$ for the positive and negative classes, respectively. While in Fig. 2 c) d1 and d2 represent the $kdistmax_{+1}(u)$, $kdistmax_{-1}(u)$ for the positive and negative classes, respectively. Although the nearest neighbor of *u* belongs to the positive class, the class that will be assigned to the unseen sample *u* by the proposed models will depend on the computed scores for each class using Eqs. (15), (17) and (19), respectively.



a) Average Linkage b) Single Linkage c) Complete Linkage

Fig. 2. Computed distances for unseen sample u in a) KLN-Avg b) KLN-Min and c) KLN-Max

4.3.1. Traditional k-nearest neighbors (kNN)

A sample is classified by identifying its *k* neighbors, counting the number of neighbors belonging to each class, and assigning it to the class to which the majority of the *k* neighbors belong. *k* is a positive integer, typically small. If *k* = 1, then the sample is simply assigned to the class of its nearest neighbor. In binary (two classes) classification problems, it is meaningful to choose *k* to be an odd number to avoid tied votes.

4.3.2. Proposed Decision Model based on Average Linkage Strategy (KLN-Avg)

In order to classify an unseen sample *u*, first, the sets of K nearest neighbors of *u* from each class *t* denoted $N_t(u)$ are computed along with their $kdistavg_t(u)$ as follows:

$$kdistavg_t(u) = \sum_{x_j \in N_t(u)} \frac{d(u,x_j)}{K} \quad (14)$$

The score of sample *u* for class *t* is calculated based on the modified z-score as follows:

$$\text{KLN-Avg score} = e^{-\propto_t \left| \frac{\text{kdistavg}_t(u) - \text{kdistavg}_t}{\text{kavgsigma}_t} \right|} \quad (15)$$

If KLN-Avg will be used as a single classifier then the parameters α_t for *t* = 1 to c should be fine tuned. In this case *u* is assigned to the class *t* having the maximal score. Another approach is to set α_t =1 for *t* = 1 to c and stack this single classifier. However, if the KLN-Avg is combined with another decision model using stacking then those parameters need not to be tuned (i.e. set to 1) and the scores are fed as they are to the meta-classifier.

4.3.3. Proposed decision model based on single linkage (KLN-Min)

In this model, in order to classify an unseen sample *u the* $kdistmin_t(u)$ is computed as follows:

$$kdistmin_t(u) = \min_{x_j \in N_t(u)} d(u,x_j) \quad (16)$$ The score of *u* for class *t* is calculated based on a modified z-score given as follows (using the previously computed mean and variance):

$$KLN\text{-}Min\ score = e^{-\beta_t \left| -\frac{(kdistmin_t(u) - kdistmin_t)}{kminsigma_t} \right|} \quad (17)$$

If KLN-Mins are going to be used as a single classifier then the parameters β_t for *t* = 1 to c should be fine tuned. In this case *u* is assigned to the class *t* having the maximal score. Another approach is to set β_t =1 for *t* = 1 to c and stack this single classifier. However, if the KLN-Min is combined with another decision model using stacking then those parameters need not to be tuned (i.e. set to 1) and the scores are fed as they are to the meta-classifier.

4.3.4. Proposed decision model based on complete linkage (KLN-Max)

This model follows a procedure similar to KLN-Avg above, but based on $kdistmax_t(u)$ which is computed as follows:

$$kdistmax_t(u) = \max_{x_j \in N_t(u)} d(u,x_j) \quad (18)$$

the score of a sample *u* for class *t* is calculated based on a modified z-score as follows:

$$\text{KLN-Max} = e^{-\eta_t |(kdistmax_t(u) - kdistmax_t)/kmaxsigma_t|} \quad (19)$$

If KLN-Mins are going to be used as a single classifier then the parameters η_t for *t* = 1 to c should be fine tuned. In this case *u* is assigned to the class *t* having the maximal score. Another approach is to set η_t =1 for *t* = 1 to c and stack this single classifier. However, if the KLN-Min is combined with another decision model using stacking then those parameters need not to be tuned (i.e. set to 1) and the scores are fed as they are to the meta-classifier.

4.3.5. Proposed Decision Model based on Density (KDN)

The KDN does not need the previously computed statistics. Instead, an incoming sample *u* is classified to class *t* if adding this sample to class *t* results in greater increase in the density of class t than the other classes. Let $N_t(u)$ be the set containing the K nearest neighbors of *u* from class *t*. In order to measure the increase in density for each class, the KNN-

1- In this paper parameters K and *k* play different roles (see Table 1).

table is scanned and if the incoming sample u can replace any of the K neighbors of another sample $x \in$ class t, the score of class t is incremented by one. Finally, the class having maximum score is selected. In order to decrease the computational complexity, only $N_t(u)$ for each class t is examined. In order to compute the score of class t, it is initially set to zero and is updated as follows:

for each $x_j \in N_t(u)$
  Identify the farthest sample x_h among the nearest neighbor of x_j (from the KNN-table).
  If ($d(x_u , x_j) < d(x_h, x_j)$) then
    add 1 to the score of class t
  Else if ($d(x_h, x_j)/ d(x_u , x_j)) > \delta$ then
    $[(d(x_h, x_j)/ d(x_u , x_j)) - \delta]/K$ is added to the score of class t
  endif;
endfor;

If the value of δ is set to 1 then the probability of tie scores increases. In this case the algorithm should do the same procedure on the upper level of neighbors (neighbors of neighbors) which may increase the classification time. We used 0.9 as a proper value for δ in our experiments.

4.3.6. Explanatory Example

Table 3 shows an example for computing the scores of the proposed models on a one-dimensional artificial dataset X. The dataset X has two classes positive and negative and has 10 values in each class listed in the first two columns , respectively. The value of K is chosen to be 2 so the distances to the 2-nearest neighbors from each class are computed as shown in columns 3-4 and 10-11, respectively. In columns 5-7 and 12-14 the average, minimum and maximum of the values in 3-4 and 10-11 are computed for the positive and negative classes, respectively. μ and σ in column 5 and 12 are computed using Eq. (10) and Eq. (13), respectively. Similarly, in columns 6 and 13, μ and σ are computed using Eq. (8) and Eq. (11), respectively. In columns 7 and 14, μ and σ are computed using Eq. (9) and Eq. (12), respectively. All the previous calculations are done during training time. In order to classify an unseen sample $x = 3$ the distances to its 2-NN from the positive class (which are 2 and 4) are computed in columns 3-4 as shown in the row before the last. Also, the distances to its 2-NN from the negative class (which are 1 and 5) are computed in columns 10-11. In columns 5-7 and 12-14 the average, minimum and maximum of the values in columns 3-4 and 10-11 are computed for the positive and negative classes, respectively. The scores of the positive and negative classes are computed in the last row for the KLN-Avg, KLN-Min and KLN-Max 5, 6 and 7 respectively. For the KDN model, x=3 has 2-NN from the positive class (2, 4) with distances (1, 1). x=3 is closer to x=2 than both 4 and 7 (4 and 7 are the 2-NN of x=2). Similarly, x=3 is closer to x=4 than both 2 and 7. So a total of four neighbors can be replaced by x=3 in the positive class. Similarly, a total of three neighbors can be replaced by x=3 in the negative class ($\delta = 1$). If max voting is used, x=3 will be classified as negative since KLN-Avg, KLN-Min and KLN-Max have higher scores for the negative class. However, Traditional kNN (with k =1 or 2) will assign x=3 to the positive class.

Table 3 Example to show how the score is computed for the proposed models on the Artificial Dataset

		Dist. 2-NN		Kdist			KDN Score for x=3		Dist. 2-NN		Kdist		
X^+	X^-	for X^+		avg.	min.	max.	+	-	of X^-		avg.	min.	max.
2	1	2	5	3.5	2	5	2	2	4	5	4.5	4	5
4	5	2	3	2.5	2	3	2	1	4	1	2.5	1	4
7	6	3	2	2.5	2	3	-	-	1	4	2.5	1	4
9	10	2	3	2.5	2	3	-	-	4	5	4.5	4	5
12	21	3	2	2.5	2	3	-	-	11	1	6	1	11
14	22	2	2	2	2	2	-	-	1	1	1	1	1
16	23	2	3	2.5	2	3	-	-	1	1	1	1	1
19	24	3	1	2	1	3	-	-	1	1	1	1	1
20	25	1	6	3.5	1	6	-	-	1	2	1.5	1.5	2
26	27	6	7	6.5	6	7	-	-	2	3	2.5	2.5	3
μ				2.75	2.2	3.7	-	-			2.7	1.8	3.7
Σ				1.438	1.398	1.702	-	-			1.75	1.25	3.02
for x=3		1	1	1	1	1	4	3	2	2	2	2	2
score				+ -	+ -	+ -	+ -		2	2			
				.29 .77	.42 .85	.20 .63	.57 .43						

4.4. Combining the decisions of the individual models

In this research work, the individual EKNN models are applied on a dataset X and their predictions are noted down. The scores of *k*NN, KLN-Avg, KLN-Min, KLN-Max, and KDN for the input dataset X can be represented as a table of size *n* by 5*c*. Both weighted majority voting and stacking can be used to combine the output of the individual models. In stacking, the score of the five decision models are used as inputs for another meta-classifier to learn from the output of the first layer models. Using weighted majority voting [52], weights should be determined for different EKNN decision models based on their individual performances. Naive brute-force approach can be used to find a suitable weight for each decision model such that an optimal ensemble classification results are achieved. Weights are usually computed such that their sum equals one. In our experimental results, stacking approach is used in combining the decision models. All the parameters of the proposed models are set to 1 for all classes and the scores are passed as they are to the meta classifier while if voting approach is used, the parameter α_t, β_t and η_t for each class *t*
should be tuned and a sample *u* is assigned to the class *t* having the maximal score.

5. Experimental Results and Discussion

In order to validate the performance of the proposed algorithm, it has been tested on five standard cancer datasets. After the feature selection step, the reduced gene expression matrix is fed to the algorithm as an input for ensemble classification. The output of each classification model is considered in classifying unseen samples.

5.1. Examples of Wrong Decisions of Traditional kNN Compared to EKNN

Table 4 shows several examples of false predictions of traditional *k*NN compared to the four proposed models and the ensemble classifier EKNN. For instance, the 18th, 39th and 50th samples of Kentridge dataset, in Table 4, are misclassified by the traditional *k*NN classifier. Also, the 2nd, 20th and 44th samples of GDS3257 are wrongly classified by traditional *k*NN. There is a significant improvement in the overall performance of EKNN compared to traditional *k*NN. It is evident from Table 4 that samples which were hard to classify for *k*NN are correctly classified by EKNN. This is mainly due to the ability of the proposed ensemble classifier EKNN to learn from the predictions of various decision models. In order to be able to use max voting in Table 4, instead of fine tuning the parameters of the proposed models we stacked them individually with the value one given to the parameter of each class and the output of the meta classifiers are combined using max voting. For highly predictive performance, the value of K should also be tuned separately for each proposed individual classifier. Also, the it is not necessary that all individual models participate in the final decision. Sometimes stacking subset of the proposed individual classifiers may yield a higher performance.

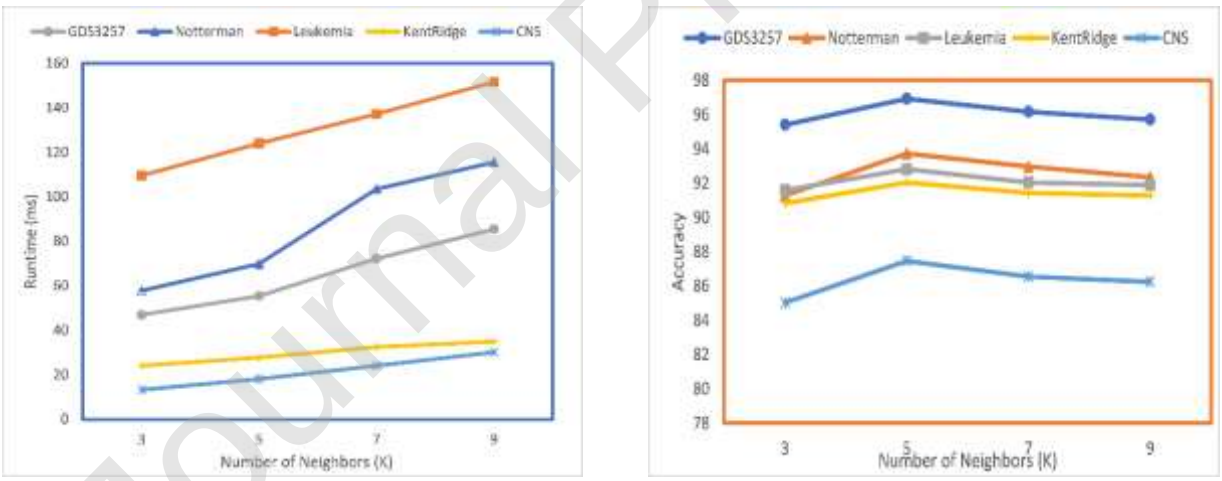
1- In this paper parameters K and *k* play different roles (see Table 1).

Table 4 Labels Assigned To Hard Samples By Various Decision Models

Dataset	Sample no.	Actual Label	Prediction					
			*k*NN	KDN	KLN-Min	KLN-Avg	KLN-Max	EKNN
Kentridge	18	-1	1	-1	1	-1	-1	-1
	39	-1	1	-1	-1	-1	1	-1
	50	-1	1	-1	1	-1	-1	-1
GDS3257	2	-1	1	-1	-1	-1	-1	-1
	20	-1	1	-1	-1	-1	1	-1
	44	-1	1	-1	-1	-1	-1	-1
Notterman	1	1	-1	1	-1	1	1	1
	19	-1	1	-1	-1	-1	-1	-1

5.2. Tuning the input parameter K for EKNN

The number of nearest neighbors K affects the performance of EKNN classifier. Therefore, in this research study, the effect of K on classification accuracy and classification time are studied. The results are shown in Fig. 3(a), which demonstrates an increase in the classification time with the increase in K. Fig. 3(b) reflects that classification accuracy increases up to K=5, and beyond this point classification performance either deteriorates or remains almost the same. Using WEKA, we found that the accuracy of traditional *k*NN increases up to *k*=3 while Notterman achieves its best accuracy for *k*=1. The proposed decision models make better usage of the nearest neighbor's information than traditional *k*NN with the help of the KNN-table.



(a) (b)

Fig. 3. Number of nearest neighbors (K) versus (a) Classification time (b) classification Accuracy of EKNN

5.3. Performance of EKNN on standard datasets

Table 5 summarizes the classification results of EKNN and compares its accuracy with each of its five base models: *k*NN, KLN-Avg, KLN-Min, KLN-Max and KDN. In addition, EKNN is compared to another four ensemble methods based on decision trees: Random Forest, Bagging and AdaBoost. The last column gives the results of an ensemble of three machine learning algorithms *k*NN, DT and NN using stacking with logistic regression as the meta-classifier. Stacking with logistic regression is also used for combining the individual models of EKNN. As shown in Table 5, EKNN outperforms *k*NN as well as all its base classifiers on the four datasets. With regards to the other ensemble classifiers, although, they

1- In this paper parameters K and *k* play different roles (see Table 1).

give quite high accuracy on several datasets, the proposed EKNN classifier is still able to outperform all of them on Ionosphere and Parkinson datasets.

Table 5 Performance Results of EKNN Compared to Individual Models and other Ensemble Methods

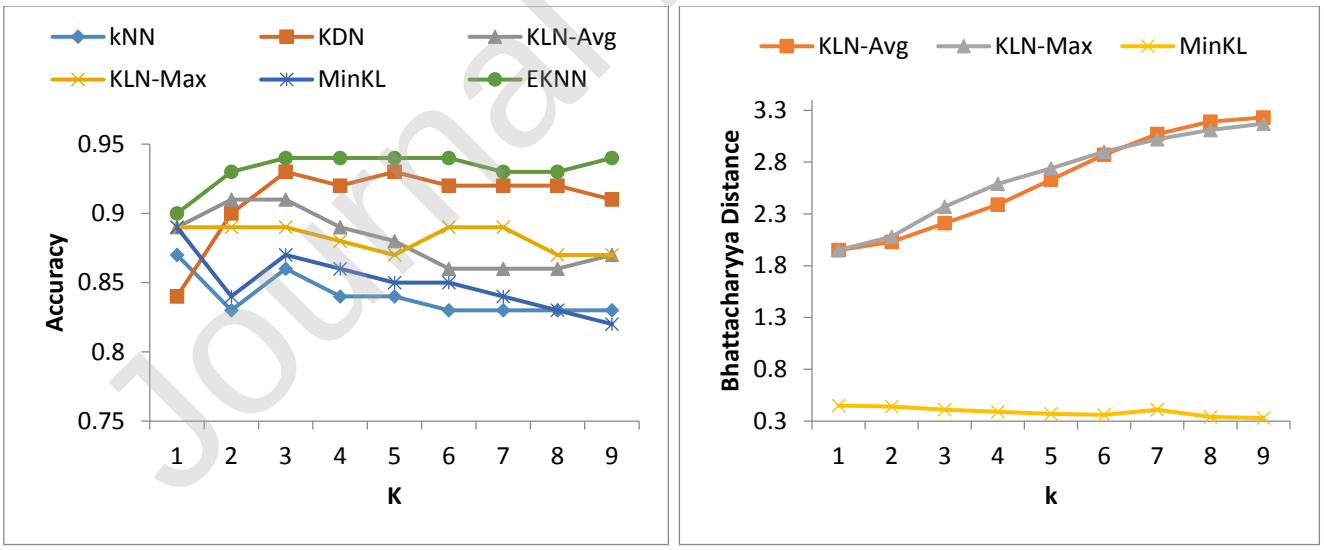
Dataset	Samples/ Features/ Classes	*k*NN	KLN-Min	KLN -Avg	KLN -Max	KDN	EKNN Stacking	Random Forest	Decision Tree Bagging	Decision Tree AdaBoost	Stacking (NN,KNN, DT)
Parkinson	195/23/2	79.82	74.67	78.31	75.62	90.31	**93.78**	92.83	82.65	85.12	92.82
Ionosphere	351/34/2	87.60	81.23	91.18	89.41	93.13	**96.48**	93.17	88.39	90.88	92.59
Transfusion	748/05/2	69.85	68.45	70.19	69.67	71.01	74.23	73.26	74.53	74.34	**77.27**
Diabetes	768/08/2	72.65	66.16	71.81	70.32	71.25	75.81	75.78	75.91	74.34	**76.17**

5.4. Effect of changing the value of K on the diversity of the distribution of KLN-Avg, KLN-Min and KLN-Max

As shown in Table 6, for the Ionosphere dataset, the performance gap between the MinKL rule and the majority rule is very small, especially for small K. This is due to the fact that the center distributions for the Ionosphere dataset are very close to the Dirac delta function in which case the MinKL reduces to the majority rule [25].

Also, the KLN-Avg and KLN-Max model perform significantly better than MinKL for all values of K. Our explanation for this increase in performance is the fact that the KLN-Avg and KLN-Max rules make the prediction based on the entire class distribution.

An important factor that affects the performance of the proposed decision models KLN-Avg and KLN-Max is the distance between their center distributions for each class. The farther they are, the better the performance of these decision models. On the other hand, the KDN model does not assume a specific distribution but only measures the increase in density. KDN is expected to perform better than the other models when the distributions of the different classes are close to each other.



(a) Accuracy (b) diversity (Bhattacharyya distance)

Fig. 4. (a) Accuracy of related algorithms compared to EKNN for different values of K (b) Distances between the distributions of the two classes of the Ionosphere dataset

Table 6 shows the distribution of the average and maximum link parameters for the two classes of the Ionosphere dataset namely p and q. For the MinKL, the probability distributions are discrete and the Bhattacharyya coefficient is computed as in Eq.(2) while for KLN-Avg and KLN-Max the distributions are continuous and Eq.(1) is used.

As shown in Table 6, the distributions of the KLN-Avg and KLN-Max on Ionosphere dataset are more spread out than those of the MinKL for the same dataset for different values of K. The KLN-Min model does not have K as a parameter. From its definition the value of K is always one (that is why it is not listed in Table 6). The Bhattacharyya coefficient on Ionosphere for KLN-Min was 0.00124. The distance between the two distributions of MinKL slightly decreases as the value of K increases while for KLN-Avg and KLN-Max it keeps increasing with the increase in K. This may explain why the best performance of EKNN is found for values of K usually larger than those for MinKL and *k*NN. However, the larger the value of K the less accurate the estimated distributions.

1- In this paper parameters K and *k* play different roles (see Table 1).

Table 6 Effect of changing the value of K on the distribution of the proposed models vs MinKL

K		KLN-Max			KLN-Avg			MinKL		
		μ	σ	d	μ	σ	D	p	q	D
1	p	0.97	0.05	1.59	0.97	0.05	1.59	0.985	0.015	0.45
	q	0.60	0.15		0.60	0.15		0.289	0.711	
3	p	0.97	0.05	2.21	0.96	0.06	2.37	0.992	0.008	0.41
	q	0.55	0.13		0.51	0.14		0.351	0.649	
5	p	0.96	0.06	2.63	0.95	0.06	2.74	0.993	0.007	0.37
	q	0.52	0.12		0.46	0.14		0.393	0.607	
7	p	0.95	0.06	3.04	0.94	0.06	3.02	0.995	0.005	0.35
	q	0.50	0.12		0.43	0.13		0.427	0.573	
9	p	0.96	0.07	3.34	0.93	0.07	3.17	0.996	0.004	0.33
	q	0.48	0.11		0.41	0.13		0.451	0.549	

5.5. Performance comparison of EKNN with its base classifiers and other Ensembles on Cancer Data

Table 7 shows the performance of EKNN using GCA [53] as the dimension reduction technique compared to its base models and three ensemble classifiers, namely AdaBoost, Bagging and Random Forest, on five cancer datasets. Decision Tree (DT) and the *k*NN are used as the base classifier for AdaBoost and Bagging. The results for the other ensemble classifiers are obtained using WEKA [12]. In table 7, Random Forest achieved the highest accuracy on two datasets while EKNN achieved the best performance compared to other ensembles on three out of five datasets in terms of accuracy.

Also, as shown in Table 7, EKNN outperform traditional *k*NN. EKNN has the highest accuracy compared to all its individual models. Ensemble classification accuracy of EKNN on Kentridge, GDS3257, Leukemia, CNS and Prostate-I are higher compared to individual models' best accuracy by 4.16%, 1.60%, 1.65%, 2.50%, and 1.30%, respectively.

Table 7 Performance Results of EKNN Compared to Individual Models and other Ensembles

Dataset	***k*NN**		KDN	KLN-Avg	KLN-Max	KLN-Min	EKNN (Stacking)			Accuracy of other Ensembles				
										Random Forest	Bagging		AdaBoost	
											*k*NN	DT	*k*NN	DT
	Acc.	*k*	Acc.	Acc.	Acc.	Acc.	Acc.	Bal. Acc.	AUC					
Kentridge	87.8	3	79.6	81.1	**89.4**	85.5	**93.12**	92.01	0.983	91.9	87.0	83.8	83.9	87.1
GDS3257	93.7	1	85.2	**97.5**	94.7	93.3	99.06	99.06	0.991	**100.0**	99.0	97.2	99.1	94.4
Leukemia	92.3	2	**97.2**	87.3	85.4	88.7	**98.81**	98.13	0.997	98.6	98.6	94.4	97.2	88.9
CNS	**76.2**	1	75.1	73.6	71.3	70.2	78.11	74.83	0.798	**88.3**	75.0	80.0	80.0	71.6
Prostate-I	93.1	1	95.3	94.0	**96.9**	95.4	**98.16**	97.31	0.976	96.3	88.2	88.2	86.8	91.9

5.6. Performance comparison of EKNN with other algorithms derived from kNN

Table 8 shows the accuracy and the ranking of the classification performance of EKNN classifier relative to the reported performance of eight *k*NN-related classifiers on three real-world datasets from UCI repository. The accuracies of the other eight classifiers in Table 8 are reported in [20]. Rank "1"is for the best and "8"is for the worst for each dataset. As shown in Table 8, the proposed EKNN method achieves the highest accuracy over two of the three real-world datasets when compared with all the other classifiers on average.

1- In this paper parameters K and *k* play different roles (see Table 1).

Table 8 Accuracy of EKNN Compared to algorithms relative to kNN

Data (UI Rep.)	samp./ att./clas.	LMKNN	MKNN	PNN	WKNN	CFKNN	FRNN	HBKNN	kGNN	EKNN
Breast	277/9/2	71.82	71.07	72.01	72.46	70.24	68.27(8)	71.80	**72.86(1)**	71.94(4)
Heart	303/13/2	79.98	77.81(8)	79.17	77.95	79.12	80.96	79.11	79.31	**81.31(1)**
Liver	345/6/2	65.21	64.13	64.72	64.42	65.78	58.85(8)	64.77	65.81	**65.87(1)**

5.7. Performance evaluation of EKNN with Different Dimension Reduction Techniques

Table 9 shows the performance of EKNN compared to the results reported in [53] for several classification techniques using two different dimensionality reduction techniques namely ACA [54] and GCA [53]. EKNN has lower accuracy relative to NN but it has the highest accuracy compared to Decision Trees, Naïve Bayes and *k*NN. The proposed approach compared to NN is simple, incremental and has less training time.

As shown in table 10, the accuracy of EKNN on Kentridge dataset is 93.3 which is the best among the other classifiers. The accuracy of NN, Decision Trees, Naïve Bayes and *k*NN are those reported in [54] using ACA. Also, EKNN achieves an accuracy of 99.06 on GDS3257 which is higher than the reported average accuracy of TC-VGC [13] which is 95.16 (using parameter *cor*=0.8-09 and average results on another input parameter termed *sig*). TC-VGC is sensitive to its two input parameters *cor* and *sig* and requires extensive parameter tuning for these parameters but it does not require the dimension reduction step. Likewise, the results prove clear superiority of EKNN over individual models regardless of the choice of gene selection strategy.

Table 9 the Performance of Different Classification Algorithms on the Leukemia Data Set

Classification Algorithm	Accuracy/ Dim. Reduction Technique	Accuracy/ Dim. Reduction Technique	Accuracy Without Dim. Reduction
Decision Trees	94.1/ACA	95.3/GCA	91.2
NN	97.1/ACA	96.2/GCA	91.2
Naïve Bayes	82.4/ACA	68.6/GCA	41.2
Random Forest	98.6/PCA	98.6/ PCA	88.9
SVM(RBF)	93.1/ PCA	93.1/ PCA	65.3
kNN	91.2/ACA	92.3/GCA	82.4
EKNN	94.7/ACA	95.91/GCA	88.4

Table 10 the Performance of Different Classification Algorithms on Kentridge Dataset

Classification Algorithm	Accuracy/ Dim. Reduction Technique	Accuracy/ Dim. Reduction Technique	Accuracy Without Dim. Reduction
Decision Trees	91.9/ACA	93.1/GCA	82.3
NN	90.3/ACA	92.3/GCA	83.9
Naïve Bayes	67.7/ACA	71.6/GCA	35.5
Random Forest	85.5/ PCA	90.3/GCA	83.9
SVM(RBF)	85.4/ PCA	87.1/GCA	80.6
kNN	83.9/ACC	87.8/GCA	79.0
EKNN	93.1/ACA	93.3/GCA	85.2

5.8. Computational complexity of EKNN

Building the KNN-table in the training phase is O($K.n^2$), where n is the number of samples and K is the number of nearest samples. The memory required for the proposed algorithm is only O($K.n$) cells. Before applying any of the individual models to classify an incoming sample, the distances from this sample to all stored samples need to be computed which is similar to *k*NN alone, this is computationally intensive, especially when the size of the training set grows but usually the number of samples is much less than number of genes in cancer datasets. Therefore, the classification time of the proposed KLN models is O(n) while for KDN, if we test only the neighbors of the incoming sample to be classified, the classification time is O(k+n)=O(n). The total computational complexity for classifying a sample using EKNN is O(n) excluding the stacking cost.

5.9. Runtime of EKNN

Table 11 shows the CPU time required during training and testing phases of EKNN compared to NN, SVM and RF. The training time corresponds to the average fit time of 5-folds CV split performed using optimal parameter values, and the testing time is the average classification (score) time of a single sample. The training time of EKNN is the sum of the time consumed by the individual models, whereas the testing time involves an overhead for stacking (meta classifier) in addition to the testing time consumed by the individual models. It should be noted that the time for reducing the dimension and estimating the number of neighbors to be kept in the KNN-table are not included here. The results in Table 11 demonstrate that the proposed EKNN technique is computationally tractable. For example, the maximum training time for the largest dataset (GDS3257 with 107 samples) is 36.4 m.s only which is much lower than the corresponding time of both RF and NN but much higher than SVM. Most of the training and testing times of the EKNN are spent in computing neighbors, the use of kd-tree [55] can further reduce both the training and testing times of EKNN using a small number of features after dimensionality reduction. As shown in Table 11, the testing time of *k*NN using kd-tree is comparable to SVM and it is about 5% of the testing time without the use of kd-tree. The proposed decision models are implemented in Python 3.8 on windows 10 and all machine learning tasks have been performed using Scikit-Learn [50] with Intel Core i5, 2.5 GHz processor and 16 GB RAM. The source code as well as some processed datasets and sample output will be available at [56].

Table 11 average runtime of EKNN compared to related algorithms (m.s)

Dataset	Sel. Genes	No. Samples	Average training time for one splite of 5-folds CV (m.s)				Average testing time for one sample (m.s)					
			SVM	RF	NN	EKNN	SVM	RF	NN	kNN	kNN (kd-tree)	EKNN
Kentridge	21	62	0.36	91.6	95.8	15.6	0.17	1.09	0.17	8.38	0.29	22.5
GDS3257	34	107	0.51	91.2	117.6	36.4	0.10	0.64	0.10	13.8	0.16	33.6
Notterman	55	36	0.38	99.2	99.6	8.6	0.14	0.88	0.13	6.95	0.43	17.4
Leukemia	50	72	0.37	94.1	128.6	20.2	0.15	0.98	0.15	9.84	0.24	25.1
CNS	84	60	0.44	93.4	102.6	14.2	0.16	1.15	0.21	8.03	0.27	21.7

6. Conclusions

In this research study, the proposed EKNN is an ensemble of the traditional *k*NN and four proposed novel classification models derived from *k*NN termed as KDN, KLN-Avg, KLN-Min and KLN-Max. From our experimental study the following can be concluded:

- The proposed decision models are able to correctly classify samples that may be wrongly classified by traditional *k*NN.
- The proposed ensemble EKNN always performs better than individual models on all the investigated datasets.
- The use of the KNN-table which is built in the training phase reduces the classification time of both EKNN and the proposed individual models.
- The KNN-table can help dealing with imbalanced data.
- The KNN-table and computed statistics in the training phase can be updated incrementally and the proposed algorithm, with slight modifications, can work on stream data however this is out of the scope of the paper.
- The training and classification time of the proposed decision models and traditional *k*NN can be highly reduced using kd-tree as shown in Table 11. This may allow EKNN to be used for large datasets.

1- In this paper parameters K and *k* play different roles (see Table 1).

- The Bhattacharyya distance gives insight into the complexity of the dataset. The larger the distance between the distributions of the different classes, for a given decision model, the lower the complexity and the higher the expected performance of the decision model. Also, it can help reduce the search space in tuning the value of K.

Therefore, we can reasonably conclude that the proposed EKNN along with a dimensionality reduction technique can help biologists in accurately predicting cancer of several body parts. The analysis of the scheme proposed in this paper suggests several directions for future work:

1. Using an ensemble of KDN, KLN-Avg, KLN-Min or KLN-Max instead of *k*NN as gene selector similar to [57].
2. Searching for a scalable technique for building the KNN-table for very large datasets.
3. In this work we stack all of the five classifiers together, however, sometimes stacking less classifiers may yield a higher performance. To tune EKNN for high predictive performance, all possible combinations of more than two classifiers (26 combinations) may be tried in a given real life application.
4. Applying EKNN to other problems that are characterized by large number of classes and small training data size such as leak detection model proposed in [32].

It has been found that by partitioning a very large dataset, most of the *k*-nearest neighbors can be found locally in each partition instead of searching the whole dataset [58]. In [58], a scalable technique for *k*NN, that relies on fuzzy clustering, has been introduced to generate a KNN-table for large datasets. In each partition, core and border samples can be approximated using rough set theory. A possible strategy is to start by partitioning Xlearning* using fuzzy clustering, computing a KNN-table for the core of each partition and another table for all border samples then merging all KNN-tables together. In our experimental study, we considered small datasets so the partitioning step was not needed.

Conflicts of Interest

The authors declare no conflict of interest.

References

[1] Shaikhina, T. and N.A. Khovanova, *Handling limited datasets with neural networks in medical applications: A small-data approach.* Artificial intelligence in medicine, 2017. **75**: p. 51-63.
[2] Dudani, S.A., *The distance-weighted k-nearest-neighbor rule.* IEEE Transactions on Systems, Man, and Cybernetics, 1976(4): p. 325-327.
[3] Batista, G. and D.F. Silva. *How k-nearest neighbor parameters affect its performance*. in *Argentine symposium on artificial intelligence*. 2009. sn.
[4] Deng, Z., et al., *Efficient kNN classification algorithm for big data.* Neurocomputing, 2016. **195**: p. 143-148.
[5] Kuncheva, L.I., *Combining pattern classifiers: methods and algorithms*. 2004: John Wiley & Sons.
[6] Kuncheva, L.I. and J.J. Rodriguez, *Classifier ensembles with a random linear oracle.* IEEE Transactions on Knowledge and Data Engineering, 2007. **19**(4): p. 500-508.
[7] Bhattacharyya, A., *On a measure of divergence between two statistical populations defined by their probability distributions.* Bull. Calcutta Math. Soc., 1943. **35**: p. 99-109.
[8] Freund, Y. and R.E. Schapire. *A desicion-theoretic generalization of on-line learning and an application to boosting*. in *European conference on computational learning theory*. 1995. Springer.
[9] Khan, J., et al., *Classification and diagnostic prediction of cancers using gene expression profiling and artificial neural networks.* Nature medicine, 2001. **7**(6): p. 673-679.
[10] Wang, Y., et al., *Gene selection from microarray data for cancer classification--a machine learning approach.* Comput Biol Chem, 2005. **29**(1): p. 37-46 DOI: 10.1016/j.compbiolchem.2004.11.001.
[11] Wu, M.-Y., et al., *Biomarker identification and cancer classification based on microarray data using laplace naive bayes model with mean shrinkage.* IEEE/ACM transactions on computational biology and bioinformatics, 2012. **9**(6): p. 1649-1662.
[12] Witten, I.H., et al., *Data Mining: Practical machine learning tools and techniques*. 2016: Morgan Kaufmann.
[13] Shin, E., et al., *TC-VGC: a tumor classification system using variations in genes' correlation.* Computer methods and programs in biomedicine, 2011. **104**(3): p. e87-e101.
[14] Mahfouz, M.A. *RBG-CD: Residue Based Genetic Cancer Diagnosis*. in *International Conference on Advanced Intelligent Systems and Informatics*. 2016. Springer DOI: 10.1007/978-3-319-48308-5_40.
[15] Keller, J.M., M.R. Gray, and J.A. Givens, *A fuzzy k-nearest neighbor algorithm.* IEEE transactions on systems, man, and cybernetics, 1985(4): p. 580-585.
[16] Sarkar, M., *Fuzzy-rough nearest neighbor algorithms in classification.* Fuzzy Sets and Systems, 2007. **158**(19): p. 2134-2152.
[17] Xu, Y., et al., *Coarse to fine K nearest neighbor classifier.* Pattern recognition letters, 2013. **34**(9): p. 980-986.
[18] Zeng, Y., Y. Yang, and L. Zhao, *Pseudo nearest neighbor rule for pattern classification.* Expert Systems with Applications, 2009. **36**(2): p. 3587-3595.
[19] Liu, H. and S. Zhang, *Noisy data elimination using mutual k-nearest neighbor for classification mining.* Journal of Systems and Software, 2012. **85**(5): p. 1067-1074.
[20] Lin, Y., et al., *A new nearest neighbor classifier via fusing neighborhood information.* Neurocomputing, 2014. **143**: p. 164-169.
[21] Pan, Z., Y. Wang, and W. Ku, *A new k-harmonic nearest neighbor classifier based on the multi-local means.* Expert Systems with Applications, 2017. **67**: p. 115-125.
[22] Medjahed, S.A., T.A. Saadi, and A. Benyettou, *Breast Cancer Diagnosis by using k-Nearest Neighbor with Different Distances and Classification Rules.* International Journal of Computer Applications, 2013. **62**(1).
[23] Mitani, Y. and Y. Hamamoto, *A local mean-based nonparametric classifier.* Pattern Recognition Letters, 2006. **27**(10): p. 1151-1159.
[24] Syaliman, K., E. Nababan, and O. Sitompul. *Improving the accuracy of k-nearest neighbor using local mean based and distance weight*. in *Journal of Physics: Conference Series*. 2018. IOP Publishing.
[25] Cheamanunkul, S. and Y. Freund. *Improved kNN Rule for Small Training Sets*. in *2014 13th International Conference on Machine Learning and Applications*. 2014. IEEE.
[26] Dai, J.J., L. Lieu, and D. Rocke, *Dimension reduction for classification with gene expression microarray data.* Statistical applications in genetics and molecular biology, 2006. **5**(1).
[27] Kohavi, R. and G.H. John, *Wrappers for feature subset selection.* Artificial intelligence, 1997. **97**(1-2): p. 273-324.
[28] Langley, P. *Selection of relevant features in machine learning*. in *Proceedings of the AAAI Fall symposium on relevance*. 1994.
[29] Backert, S., et al., *Differential gene expression in colon carcinoma cells and tissues detected with a cDNA array.* International journal of cancer, 1999. **82**(6): p. 868-874.
[30] Geng, Z., et al., *Energy optimization and prediction modeling of petrochemical industries: An improved convolutional neural network based on cross-feature.* Energy, 2020. **194**: p. 116851.
[31] Han, Y., et al., *Energy efficiency evaluation of complex petrochemical industries.* Energy, 2020: p. 117893.
[32] Hu, X., et al., *Novel leakage detection and water loss management of urban water supply network using multiscale neural networks.* Journal of Cleaner Production, 2020. **278**: p. 123611.
[33] Geng, Z., et al., *Early warning and control of food safety risk using an improved AHC-RBF neural network integrating AHP-EW.* Journal of Food Engineering. **292**: p. 110239.
[34] Geng, Z., et al., *Semantic relation extraction using sequential and tree-structured LSTM with attention.* Information Sciences, 2020. **509**: p. 183-192.
[35] Chen, Z. and J. Li. *A multiple kernel support vector machine scheme for simultaneous feature selection and rule-based classification*. in *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. 2007. Springer.
[36] Rathore, S., M. Hussain, and A. Khan, *GECC: Gene expression based ensemble classification of colon biopsies.*
[37] Lu, H., et al., *A cost-sensitive rotation forest algorithm for gene expression data classification.* Neurocomputing, 2017. **228**: p. 270-276.
[38] Tan, S., *Neighbor-weighted k-nearest neighbor for unbalanced text corpus.* Expert Systems with Applications, 2005. **28**(4): p. 667-671.
[39] Ganganwar, V., *An overview of classification algorithms for imbalanced datasets.* International Journal of Emerging Technology and Advanced Engineering, 2012. **2**(4): p. 42-47.
[40] Mani, I. and I. Zhang. *kNN approach to unbalanced data distributions: a case study involving information extraction*. in *Proceedings of workshop on learning from imbalanced datasets*. 2003.
[41] Jadhav, A.S., *A novel weighted TPR-TNR measure to assess performance of the classifiers.* Expert Systems with Applications, 2020: p. 113391.
[42] Behzadian, M., et al., *A state-of the-art survey of TOPSIS applications.* Expert Systems with applications, 2012. **39**(17): p. 13051-13069.

[43] Alon, U., et al., *Broad patterns of gene expression revealed by clustering analysis of tumor and normal colon tissues probed by oligonucleotide arrays.* Proceedings of the National Academy of Sciences, 1999. **96**(12): p. 6745-6750.
[44] Landi, M.T., et al., *Gene expression signature of cigarette smoking and its role in lung adenocarcinoma development and survival.* PloS one, 2008. **3**(2): p. e1651.
[45] Notterman, D.A., et al., *Transcriptional gene expression profiles of colorectal adenoma, adenocarcinoma, and normal tissue examined by oligonucleotide arrays.* Cancer research, 2001. **61**(7): p. 3124-3130.
[46] Golub, T.R., et al., *Molecular classification of cancer: class discovery and class prediction by gene expression monitoring.* science, 1999. **286**(5439): p. 531-537.
[47] Pomeroy, S.L., et al., *Prediction of central nervous system embryonal tumour outcome based on gene expression.* Nature, 2002. **415**(6870): p. 436-442.
[48] Al-Shahrour, F., et al., *BABELOMICS: a suite of web tools for functional annotation and analysis of groups of genes in high-throughput experiments.* Nucleic acids research, 2005. **33**(suppl 2): p. W460-W464.
[49] Hassan, M., et al., *Carotid artery image segmentation using modified spatial fuzzy c-means and ensemble clustering.* Computer Methods and Programs in Biomedicine, 2012. **108**(3): p. 1261-1276.
[50] *Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011*. Available from: https://scikit-learn.org/stable/about.html.
[51] Han, H., W.-Y. Wang, and B.-H. Mao, *Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning.* Advances in intelligent computing, 2005: p. 878-887.
[52] Littlestone, N. and M.K. Warmuth. *The weighted majority algorithm*. in *Foundations of Computer Science, 1989., 30th Annual Symposium on*. 1989. IEEE.
[53] Mahfouz, M.A. and J.A. Nepomuceno, *Graph coloring for extracting discriminative genes in cancer data.* Annals of human genetics, 2019.
[54] Au, W.-H., et al., *Attribute clustering for grouping, selection, and classification of gene expression data.* IEEE/ACM transactions on computational biology and bioinformatics, 2005. **2**(2): p. 83-101.
[55] Zhou, K., et al., *Real-time kd-tree construction on graphics hardware.* ACM Transactions on Graphics (TOG), 2008. **27**(5): p. 126.
[56] Available from: https://github.com/mamahfouz66/EKNN_Ensemble_KNN_Based_Classifier.
[57] Okun, O. and H. Priisalu, *Dataset complexity in gene expression based cancer classification using ensembles of k-nearest neighbors.* Artificial intelligence in medicine, 2009. **45**(2): p. 151-162.
[58] Mahfouz, M., *Rfknn: Rough-Fuzzy Knn for Big Data Classification.* International Journal of Advanced Research in Computer Science, 2018. **9**(2): p. 274-279 DOI: 10.26483/ijarcs.v9i2.5667.