A Proximity Weighted Evidential *k* Nearest Neighbor Classifier for Imbalanced Data

Md. Eusha Kadir[1(✉)], Pritom Saha Akash[1], Sadia Sharmin[2], Amin Ahsan Ali[3], and Mohammad Shoyaib[1]

[1] Institute of Information Technology, University of Dhaka, Dhaka, Bangladesh
{bsse0708,bsse0604}@iit.du.ac.bd, shoyaib@du.ac.bd
[2] Islamic University of Technology, Gazipur, Bangladesh
sharmin@iut-dhaka.edu
[3] Independent University, Dhaka, Bangladesh
aminali@iub.edu.bd

Abstract. In k Nearest Neighbor (kNN) classifier, a query instance is classified based on the most frequent class of its nearest neighbors among the training instances. In imbalanced datasets, kNN becomes biased towards the majority instances of the training space. To solve this problem, we propose a method called Proximity weighted Evidential kNN classifier. In this method, each neighbor of a query instance is considered as a piece of evidence from which we calculate the probability of class label given feature values to provide more preference to the minority instances. This is then discounted by the proximity of the neighbor to prioritize the closer instances in the local neighborhood. These evidences are then combined using Dempster-Shafer theory of evidence. A rigorous experiment over 30 benchmark imbalanced datasets shows that our method performs better compared to 12 popular methods. In pairwise comparison of these 12 methods with our method, in the best case, our method wins in 29 datasets, and in the worst case it wins in least 19 datasets. More importantly, according to Friedman test the proposed method ranks higher than all other methods in terms of AUC at 5% level of significance.

Keywords: Classifier · Imbalanced learning · kNN · Evidence theory

1 Introduction

Classification is one of the most important tasks in machine learning. Numerous classification approaches, such as k Nearest Neighbor (kNN) [9], Decision Tree (DT), Naïve Bayes (NB), and Support Vector Machine, have been well developed and applied in many applications. However, most of the classifiers face serious trouble for imbalanced class distribution and thus learning from the imbalanced dataset is one of the top ten challenging problems in data mining research [20].

H. W. Lauw et al. (Eds.): PAKDD 2020, LNAI 12085, pp. 71–83, 2020.
https://doi.org/10.1007/978-3-030-47436-2_6

To solve class imbalance problem, various strategies have already been proposed which can be grouped into two broad categories namely data oriented and algorithm oriented approaches. Data oriented approaches use sampling techniques. In order to make dataset balanced, the sampling techniques either oversample the minority instances or select instances (under-sample) from the majority class. A sampling technique namely Synthetic Minority Over-sampling TEchnique (SMOTE) has been proposed that increases the number of minority class instances by creating artificial and non-repeated samples [4].

In contrast, algorithm oriented approaches are the modifications of traditional algorithms such as DT and kNN. The modified DTs for imbalanced classification are Hellinger Distance DT (HDDT) [5], Class Confidence Proportion DT (CCPDT) [13] and Weighted Inter-node Hellinger Distance DT (iHDwDT) [1]. These DTs use different splitting criteria while selecting a feature in split point.

kNN is one of the simplest classifiers. Despite its simplicity, kNN is considered as one of the top most influential data mining algorithms [19]. Traditional kNN finds the k closest instances from the training data to a query instance and treats all neighbors equally. Dudani has proposed a distance based weighted kNN which provides more weights to closer neighbors [8]. Another variant of kNN approach, Generalized Mean Distance based kNN (GMDKNN) [10], has been presented by introducing multi-generalized mean distance and the nested generalized mean distance. All these variants of kNN are sensitive to the majority instances and thus perform poorly for imbalanced datasets.

Considering this imbalance problem, several researchers extended kNN for imbalanced datasets [7,11,12]. In Exemplar-based kNN (kENN) [11], Li and Zhang expand the decision boundary for the minority class by identifying the exemplar minority instances. A weighting algorithm namely Class Confidence Weighted kNN (CCWKNN) has been presented in [12] where the probability of feature values given the class labels is considered as weight. Dubey and Pudi have proposed a weighted kNN (WKNN) [7] which considers the class distribution in a wider region around a query instance. The class weight for each training instance is estimated by taking the local class distributions into account.

The purpose of these existing studies is to improve the overall performance for imbalanced data. However, these methods overlook the problem of uncertainty which is prevalent in almost all datasets [18]. The reason behind this uncertainty is that the complete statistical knowledge associated with the conditional density function of each class is hardly available [6]. To address this problem, kNN has been extended using Dempster-Shafer Theory of evidence (DST) to better model uncertain data named Evidential kNN (EKNN) [6]. In EKNN, each neighbor assigns basic belief on classes based on a distance measure. Nevertheless, this approach again does not take consideration of the class imbalance problem.

To address these aforementioned problems, we propose a Proximity weighted Evidential kNN (PEkNN) classifier and make the following contributions. Firstly, we have proposed a confidence (posterior) assignment procedure on each neighbor of a query instance. Secondly, we have also proposed to use proximity of a

neighbor as a weight to discount the confidence of a neighbor. It is shown that, this weighted confidence increases the likelihood of classifying a minority class. Thirdly, DST framework is used to combine decisions from different neighbors.

2 Dempster-Shafer Theory of Evidence

Dempster-Shafer theory of evidence is a generalized form of Bayesian theory. It assigns degree of belief for all possible subsets of the hypothesis set. Let, $C = \{C_1, \ldots, C_M\}$ be a finite hypothesis set of mutually exclusive and exhaustive hypotheses. The belief in a hypothesis assigned based on a piece of evidence is ranged numerically as $[0, 1]$. A Basic Belief Assignment (BBA) is a function $m : 2^C \rightarrow [0, 1]$ which satisfies the following properties:

$$m(\emptyset) = 0 \quad \text{and} \quad \sum_{A \subseteq C} m(A) = 1 \tag{1}$$

where $m(A)$ is a degree of belief (referred as mass) which reflects how strongly A is supported by the piece of evidence. $m(C)$ represents the degree of ignorance.

Several pieces of evidence characterized by their BBAs can be fused using Dempster's rule of combination [16]. For two BBAs $m_1(.)$ and $m_2(.)$ which are not totally conflicting, the combination rule can be expressed using Eq. (2).

$$m(A) = \frac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1 - \sum_{B \cap C = \emptyset} m_1(B) m_2(C)} \quad A \neq \emptyset \tag{2}$$

where $A, B, C \in 2^C$ and $\sum_{B \cap C = \emptyset} m_1(B) m_2(C) < 1$.

For decision making, Belief, Plausibility and betting Probability (P_{bet}) are usually used. For a singleton class A, $P_{bet}(A)$ is derived in Eq. (3) where $|B|$ represents the cardinality of the element B.

$$P_{bet}(A) = \sum_{A \subseteq B} \frac{|A \cap B|}{|B|} \times m(B) \tag{3}$$

3 Proximity Weighted Evidential *k*NN (PE*k*NN)

kNN faces difficulty in imbalanced datasets as it treats all neighbors of the query instance equally and most of the neighbors will be of the majority class. To deal with this issue, the proposed algorithm attempts to provide more importance to neighbors with a higher proximity weighted confidence. Here, confidence of an instance indicates a conditional probability of that instance based on training data. Algorithms such as NB also uses conditional probability while classifying a query instance. However, the performance of NB degrades due to the poor estimation of the conditional density of the query instance associated with each class. In contrast, PEkNN computes conditional probability of neighborhood instances rather than query instance. Furthermore, as uncertainty is prevalent in

almost all datasets [18]. This is more significant for imbalanced datasets where little information is available for the minority class. To deal with this issue, PEkNN uses DST to combine the evidences provided by each neighbor.

For a new query instance (x_t), PEkNN first finds k closest neighbors according to some distance measurement (e.g. Euclidean distance). Let, $S(x_t, k)$ be the set of k closest neighbors of x_t and each member of $S(x_t, k)$ is considered as a piece of evidence which assigns mass values for each subset of C known as BBA.

Now, consider x_i as the i-th neighbor of x_t belonging to class C_q. As x_i is a piece of evidence belonging to C_q, some part of its belief will be committed to C_q. The rest of the belief can not be distributed to any other subset of C except itself. The BBA provided by x_i can be represented by Eq. (4), (5) and (6) where $0 < \beta_0 < 1$.

$$m_i(\{C_q\}) = \beta = \beta_0 \times \Psi(x_i, x_t) \tag{4}$$

$$m_i(A) = 0 \quad \forall A \in 2^C \backslash \{C, \{C_q\}\} \tag{5}$$

$$m_i(C) = 1 - \beta \tag{6}$$

Now, we will discuss about two of our intuitions. First, a piece of evidence belonging to C_q will assign a larger belief to C_q when the evidence is more reliable which we call confidence. An evidence having higher posterior probability should get more confidence than the one which is in lower posterior probability region. The second intuition is that a neighbor will assign more belief to a specific class when the neighbor and the query instance are more proximate. The function defined in Eq. (7), $\Psi(.)$ satisfies the two aforementioned intuitions where p_i is the confidence of x_i represented by the probability of class label (y_i) given x_i and $prx(x_i, x_t)$ represents the proximity between x_i and x_t.

$$\Psi(x_i, x_t) = prx(x_i, x_t) \times p_i \tag{7}$$

The procedure how PEkNN algorithm classifies a query instance is presented in Algorithm 1. The confidence assignment, proximity estimation and decision making steps are presented in detail in Sects. 3.1, 3.2 and 3.3 respectively.

3.1 Estimation of Confidence

The confidence (p_i) of an instance x_i ($x_i \in \mathbb{R}^l$) belonging to y_i is assigned in the following manner derived in Eq. (8).

$$p_i = P(y_i \mid x_i) = \frac{P(y_i) \times P(x_i \mid y_i)}{\sum_{j=1}^{M} P(C_j) \times P(x_i \mid C_j)} \tag{8}$$

where $y_i \in \{C_1, C_2, \ldots, C_M\}$, $P(C_j)$ represents the prior of C_j in training space and $P(x_i|C_j)$ represents the likelihood in Bayes' theorem. Here, two approaches of estimating class-wise Probability Density Function (PDF) is presented. First one is using Single Gaussian Model (SGM) and another one is using Gaussian Mixture Model (GMM). When PEkNN uses confidence derived from SGM, we call it sPEkNN, and mPEkNN when it uses confidence derived from GMM.

Algorithm 1: PEkNN Algorithm

Input : Training data (X), Training data labels (Y), Neighborhood size (k), Query instance (x_t)
Output: Predicted class label (y_t)

$conf$, $d_{max} \leftarrow$ `fitModel(`X, Y`)`
$s \leftarrow$ indices of k nearest neighbors of x_t
Initialize a list bba of mass values
for i = *1 to* k **do**
    $index \leftarrow s[i]$
    $confidence \leftarrow conf[index]$
    $d \leftarrow$ distance(x_t, $X[index]$)
    $proximity \leftarrow$ calculate proximity using Eq. (11) from d, d_{max}
    $bba[i] \leftarrow$ assign mass value using Eq. (4), (5), (6) and (7) from $confidence$, $proximity$
end
$m \leftarrow$ combine mass values from bba using Eq. (2)
$P_{bet} \leftarrow$ calculate betting probabilities for all classes using Eq. (3) from m
$y_t \leftarrow$ calculate decision using Eq. (12) from P_{bet}
function `fitModel(`X, Y`):`
    Initialize Array, $conf$
    $d_{max} \leftarrow 0$
    for i = *1 to* $|X|$ **do**
        $conf[i] \leftarrow$ Calculate confidence using Eq. (8) from $X[i]$, $Y[i]$
        for j = *i+1 to* $|X|$ **do**
            $d \leftarrow$ distance($X[i]$, $X[j]$)
            $d_{max} \leftarrow \max(d, d_{max})$
        end
    end
    return $conf$, d_{max}

Single Gaussian model assumes that all the features are independent and the continuous values associated with each class follow a normal distribution. Under these assumptions, the likelihood function can be represented as Eq. (9).

$$P(x) = \prod_{j=1}^{l} P(x_j) = \prod_{j=1}^{l} f(x_j; \mu_j, {\sigma_j}^2) = \prod_{j=1}^{l} \frac{1}{\sqrt{2\pi}\sigma_j} \times \exp(-\frac{(x_j - \mu_j)^2}{2\sigma_j^2}) \quad (9)$$

where x_j denotes the j-th feature of x and $f(.)$ represents the normally distributed PDF parameterized by mean (μ) and variance (σ^2).

On the other hand, GMM can also be used to estimate PDF from multivariate data. The class-wise PDF using m-component mixture model is given in Eq. (10).

$$P(x) = \sum_{i=1}^{m} \alpha_i P(x \mid Z_i) \quad (10)$$

The procedure of finding complete set of parameters $(Z_1, \ldots, Z_m, \alpha_1, \ldots, \alpha_m)$ specifying the mixture model is briefly described in [14].

3.2 Estimation of Proximity

To capture the proximity between two instances, some distance measurement can be used. The proximity between two instances (x_i and x_j) from training samples will be maximum when x_i and x_j are identical. One the other hand, it will be lowest when they are the farthest two instances in the feature space. To measure this proximity, a normalization is applied as Eq. (11) so that $prx(x_i, x_j) \in [0, 1]$. Here, d_{max} is the distance between two farthest training instances.

$$prx(x_i, x_j) = 1 - \frac{d(x_i, x_j)}{d_{max}} \tag{11}$$

3.3 Decision Making

According to Eq. (7), $\Psi(.)$ will return a larger value when a neighbor is more confident and more closer to the query instance. Now, for each of the k nearest neighbors, the BBAs are defined using Eq. (4), (5) and (6). In order to classify x_t, these BBAs are combined using DST. The betting probability (P_{bet}) for each singleton class from this combined decision will be then calculated using Eq. (3). Finally, the decision from this P_{bet} is taken using Eq. (12).

$$\hat{y} = \underset{c \in \{C_1, \ldots, C_M\}}{\arg\max} P_{bet}(c) \tag{12}$$

where c is a singleton class so that the cardinality of c is 1.

Properties of β: Value of β is bounded between 0 to 1.

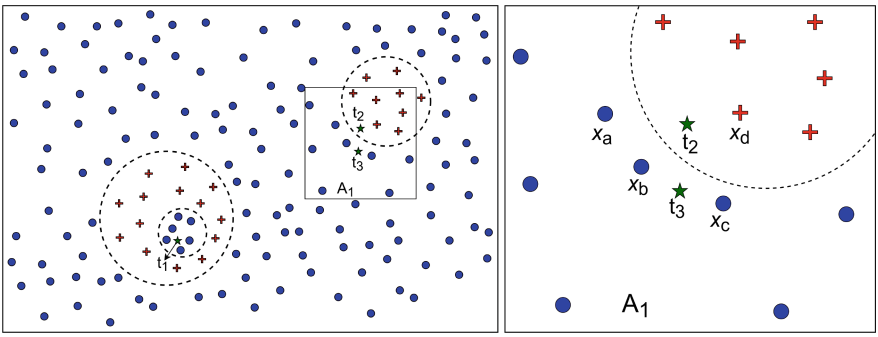
Proof. From Eq. (4), (7) and (8), it can be derived that,

$$\beta = \beta_0 \times P(y_i \mid x_i) \times prx(x_i, x_t) \tag{13}$$

Here, β_0 is a user given constant satisfying $0 < \beta_0 < 1$. The second term, $P(y_i \mid x_i)$, represents the posterior probability. The last term, $prx(x_i, x_t)$ is at most equal to 1 and at least equal to zero. As can be seen from Eq. (13), β is a product of three terms and all these terms are bounded between 0 to 1. It is sufficient to claim that, the value of β must be bounded between 0 to 1.

3.4 An Illustrative Example

Figure 1 shows the instances of a two-class imbalance problem where (+)s and (•)s represent the minority (Class-A) and majority class instances (Class-B) instances respectively. The class boundaries are represented as dotted lines and three query instances (t_1, t_2, t_3) are marked with ($\star$)s. Here, first query instance t_1 is situated in a majority class region bounded by minority instances. Both kNN and PEkNN can successfully classify t_1. Traditional algorithms such as C4.5 and NB face difficulties in this situation.



(a) Training space (b) A_1 region

Fig. 1. A synthetic imbalanced dataset

The two other query instances t_2 and t_3 associated with a region namely A_1 (see Fig. 1b). Here, for both t_2 and t_3, the four neighbors are x_a, x_b, x_c and x_d. Traditional kNN with $k = 4$, will classify both t_2 and t_3 as Class-B. PEkNN, on the other hand, considers the confidence of each neighbor. Here, x_d will provide a higher confidence compared to majority class instances (x_a, x_b and x_c). Assume, the confidence of x_a, x_b, x_c, and x_d are 0.30, 0.40, 0.30, and 0.75 respectively. And the proximity with respect to t_2 are 0.90, 0.95, 0.85 and 0.95 respectively. Then BBAs assigned by PEkNN for these neighbors are $m_a(\{B\}) = 0.2565$, $m_a(\{A, B\}) = 0.7435$, $m_b(\{B\}) = 0.3610$, $m_b(\{A, B\}) = 0.6390$, $m_c(\{B\}) = 0.2423$, $m_c(\{A, B\}) = 0.7577$ and $m_d(\{A\}) = 0.6769$, $m_d(\{A, B\}) = 0.3231$. Here, β_0 is set to 0.95. Now, combing these BBAs using DST, we get $P_{bet}(A) = 0.5325$ and $P_{bet}(B) = 0.4675$ which indicates that t_2 will be correctly classified as Class-A.

On the other hand, for the query instance t_3, the proximity of x_a, x_b, x_c and x_d are 0.85, 0.95, 0.95 and 0.85 respectively. We, therefore, get $P_{bet}(A) = 0.4661$ and $P_{bet}(B) = 0.5339$ indicating that t_3 will be classified as Class-B. Therefore, t_3 is correctly classified as a majority class instance even though the neighbors of t_2 and t_3 are same.

Instead of DST, let us reconsider simpler techniques to combine evidences such as summing and taking the maximum of the proximity weighted confidences. If we simply sum class-wise proximity weighted confidences, both t_2 and t_3 get a higher value for Class-B as three of the four neighbors belong to that class. To avoid this bias, a query can be simply classified in the class for which it gets maximum proximity weighted confidence among the neighbors. But this method does not consider the local neighborhood priors. For which, it will classify both t_2 and t_3 as minority class which is not desired. PEkNN on the other hand using the DST framework successfully classifies both query instances.

4 Experiments and Results

Dataset description, implementation details and the performance metrics followed by the results obtained from the experiments with discussion are given in the following subsections.

4.1 Dataset Description

The characteristics of the 30 benchmark datasets are shown in Table 1 which are collected from UCI machine learning repository [3] and KEEL Imbalanced Datasets [2]. Imbalance Ratio (IR) between the samples of majority class and minority class of the datasets used in these experiment are at least 1.5 and values of all the features are numeric. A dataset is highly imbalanced when the value of IR is very high.

Table 1. Descriptions of Imbalanced Datasets. Idx, #Inst, #Cl and #Ftr represent index of a dataset, number of instances, classes and features respectively.

Idx	Name	#Ftr	#Cl	#Inst	IR	Idx	Name	#Ftr	#Cl	#Inst	IR
01	Appendicitis	7	2	106	4.05	16	Shuttle-c0-vs-c4	9	2	1829	13.87
02	Ecoli1	7	2	336	3.36	17	Vehicle0	18	2	846	3.25
03	Ecoli2	7	2	336	5.46	18	Vehicle1	18	2	846	2.90
04	Ecoli3	7	2	336	8.60	19	Vehicle2	18	2	846	2.88
05	Ecoli4	7	2	336	15.80	20	Vehicle3	18	2	846	2.99
06	Glass-0-1-2-3_vs_4-5-6	9	2	214	3.20	24	Yeast-1_vs_7	7	2	459	14.30
07	Glass1	9	2	214	1.82	21	Vowel0	13	2	988	9.98
08	Glass4	9	2	214	15.46	22	Wisconsin	9	2	683	1.86
09	Glass6	9	2	214	6.38	23	Yeast-0-5-6-7-9_vs_4	8	2	528	9.35
10	Haberman	3	2	306	2.78	25	Yeast-1-2-8-9_vs_7	8	2	947	30.57
11	Ionosphere	34	2	351	1.79	26	Yeast-2_vs_8	8	2	482	23.10
12	New-thyroid1	5	2	215	5.14	27	Yeast1	8	2	1484	2.46
13	Page-blocks0	10	2	5472	8.79	28	Yeast3	8	2	1484	8.10
14	Pima	8	2	768	1.87	29	Yeast5	8	2	1484	32.73
15	Segment0	19	2	2308	6.02	30	Yeast6	8	2	1484	41.40

4.2 Implementation Details and Performance Metrics

PEkNN is benchmarked against other algorithms including traditional learning algorithms (kNN, C4.5, NB), oversampling strategy (SMOTE), recent algorithms in the kNN family (EKNN, WKNN, CCWKNN, kENN, GMDKNN) and few tree based recent algorithms for imbalanced classification (CCPDT, HDDT, iHDwDT). For PEkNN, we use $\beta_0 = 0.95$ in this experiment. For kENN, the confidence level is set 0.1 and we set $p = 1$ for GMDKNN.

We have conducted 10-fold stratified cross validation to evaluate the performance of the proposed method. The Receiver Operating Characteristic (ROC) curve [17] is widely used to evaluate imbalanced classification. We use Area Under the ROC Curve (AUC) for evaluating the classifier performance.

Table 2. Performance comparison among different algorithms on imbalanced datasets in terms of AUC (%). The best result for each dataset is in bold. SMT+kNN represents kNN followed by SMOTE sampling.

Index	C4.5	NB	kNN	GMDKNN	CCWKNN	kENN	WKNN	EKNN	CCPDT	HDDT	iHDwDT	SMT+kNN	sPEkNN	mPEkNN
01	76.94(11)	**82.78(1)**	79.58(9)	73.61(13)	75.38(12)	79.58(9)	79.79(8)	80.39(6)	71.83(14)	80.06(7)	80.88(5)	80.97(4)	82.01(2)	81.81(3)
02	81.40(14)	91.39(12)	92.89(7)	84.15(13)	92.20(11)	92.41(10)	93.00(6)	92.67(9)	94.44(3)	**94.71(1)**	94.15(4)	92.68(8)	93.95(5)	94.67(2)
03	85.57(14)	93.88(8)	**95.32(1)**	90.73(13)	94.56(6)	94.40(7)	95.30(2)	94.93(3)	91.27(11)	91.41(10)	90.83(12)	94.61(5)	93.36(9)	94.79(4)
04	76.40(13)	89.98(5)	89.83(7)	73.08(14)	88.46(10)	91.19(3)	89.08(8)	88.27(11)	90.87(4)	89.84(6)	87.76(12)	88.77(9)	92.47(2)	**94.39(1)**
05	83.59(14)	94.69(9)	94.92(5)	89.69(12)	94.84(7)	95.89(2)	94.92(5)	95.65(3)	86.57(13)	90.36(11)	92.47(10)	94.76(8)	**97.48(1)**	95.65(3)
06	91.39(13)	96.38(9)	95.82(10)	90.79(14)	96.43(8)	97.18(4)	95.41(12)	**99.12(1)**	96.59(5)	95.77(11)	96.46(7)	96.52(6)	98.26(3)	98.88(2)
07	75.68(13)	70.78(14)	82.88(5)	78.78(11)	80.75(10)	**85.84(1)**	82.70(6)	84.92(2)	83.51(4)	80.88(9)	76.39(12)	81.41(8)	83.70(3)	82.41(7)
08	76.52(14)	77.00(13)	91.52(9)	81.74(12)	90.01(11)	95.39(5)	91.52(9)	97.51(3)	95.51(4)	94.76(6)	94.76(6)	**98.77(1)**	94.01(8)	97.79(2)
09	85.88(14)	94.22(11)	95.57(7)	89.36(13)	96.13(5)	91.34(12)	95.39(8)	**98.89(1)**	96.40(4)	95.27(9)	95.27(9)	95.84(6)	97.79(3)	97.97(2)
10	54.22(13)	64.18(10)	66.43(5)	53.92(14)	68.35(3)	65.15(7)	64.31(9)	68.45(2)	66.02(6)	61.90(12)	62.10(11)	64.54(8)	**69.06(1)**	67.21(4)
11	86.87(12)	94.22(4)	91.16(10)	85.54(13)	79.69(14)	91.76(7)	91.25(9)	97.67(3)	90.90(11)	91.78(6)	91.68(8)	93.04(5)	**98.45(1)**	98.25(2)
12	93.36(14)	**100.0(1)**	99.58(7)	98.15(12)	**100.0(1)**	98.93(9)	99.81(3)	99.68(5)	97.96(13)	98.63(10)	98.36(11)	99.58(7)	99.81(3)	99.68(5)
13	91.77(14)	92.96(12)	96.11(8)	92.52(13)	96.03(9)	95.85(11)	96.16(7)	97.09(5)	98.15(3)	98.54(2)	**98.67(1)**	96.80(6)	95.94(10)	97.64(4)
14	68.92(14)	**81.75(1)**	77.93(5)	69.61(13)	77.77(8)	77.24(9)	77.93(5)	76.16(12)	76.52(11)	76.69(10)	77.89(7)	77.98(4)	81.34(2)	80.36(3)
15	97.94(14)	98.00(13)	99.94(4)	99.48(12)	99.90(6)	99.88(7)	99.95(2)	**99.98(1)**	99.59(9)	99.54(10)	99.54(10)	99.93(5)	99.77(8)	99.95(2)
16	**100.0(1)**	97.91(14)	99.59(8)	99.59(8)	99.59(8)	99.59(8)	99.59(8)	**100.0(1)**	**100.0(1)**	**100.0(1)**	**100.0(1)**	99.59(8)	**100.0(1)**	**100.0(1)**
17	91.16(13)	81.37(14)	97.80(7)	93.42(12)	96.44(11)	96.47(10)	97.93(6)	98.17(5)	**98.63(1)**	98.39(3)	98.60(2)	97.74(8)	96.52(9)	98.24(4)
18	64.51(14)	71.20(12)	83.03(3)	67.06(13)	81.40(6)	79.20(8)	83.07(2)	78.86(11)	81.71(5)	80.35(7)	78.91(10)	82.74(4)	79.04(9)	**84.38(1)**
19	94.50(13)	85.58(14)	99.21(3)	97.92(10)	98.48(9)	97.46(12)	99.20(4)	**99.64(1)**	98.95(6)	99.11(5)	98.83(7)	98.75(8)	97.56(11)	99.63(2)
20	69.22(13)	69.82(12)	80.47(6)	65.64(14)	80.34(7)	78.81(11)	82.36(2)	78.82(9)	81.04(4)	81.24(3)	80.99(5)	80.19(8)	78.82(9)	**84.08(1)**
21	96.66(14)	98.24(12)	99.84(8)	**100.0(1)**	99.84(8)	99.98(7)	**100.0(1)**	**100.0(1)**	98.33(11)	98.35(10)	97.84(13)	**100.0(1)**	**100.0(1)**	**100.0(1)**
22	92.68(14)	98.93(5)	98.67(7)	95.75(13)	96.08(12)	**99.30(1)**	98.66(8)	98.57(11)	99.07(2)	99.04(3)	98.65(9)	98.63(10)	99.01(4)	98.88(6)
23	67.60(13)	81.64(8)	79.49(10)	63.47(14)	78.15(11)	84.40(4)	77.96(12)	82.68(7)	**87.62(1)**	86.04(2)	84.30(5)	81.05(9)	85.01(3)	83.61(6)
24	69.38(14)	81.12(2)	77.51(5)	69.88(13)	74.45(9)	75.85(8)	76.85(7)	73.10(12)	74.11(10)	77.16(6)	79.30(3)	78.50(4)	**83.30(1)**	73.92(11)
25	57.41(14)	74.86(2)	68.58(9)	60.92(13)	68.74(7)	73.12(3)	68.63(8)	68.40(11)	67.06(12)	68.44(10)	71.16(5)	72.23(4)	**79.77(1)**	68.93(6)
26	70.53(14)	82.87(7)	82.43(9)	75.43(13)	82.73(8)	80.67(11)	83.43(6)	85.01(5)	85.03(4)	**88.27(1)**	81.50(10)	80.05(12)	86.95(3)	87.41(2)
27	64.93(14)	77.97(2)	74.63(6)	65.48(13)	72.75(11)	76.42(3)	73.63(10)	71.04(12)	74.24(7)	74.22(9)	75.85(5)	74.24(7)	75.96(4)	**78.61(1)**
28	84.44(14)	95.74(6)	93.78(8)	85.34(13)	93.23(12)	96.59(4)	93.65(9)	93.33(11)	96.74(2)	96.41(5)	**96.92(1)**	93.47(10)	95.64(7)	96.74(2)
29	85.70(13)	98.64(3)	96.72(10)	81.94(14)	96.27(12)	**99.23(1)**	96.78(9)	97.82(8)	98.11(5)	97.90(6)	97.85(7)	96.53(11)	99.15(2)	98.53(4)
30	72.68(14)	93.17(2)	85.25(10)	74.85(13)	84.73(12)	91.24(3)	85.08(11)	85.92(9)	89.53(4)	87.33(7)	87.76(6)	86.31(8)	**95.60(1)**	89.41(5)
Avg. Rank	13.10	7.93	6.93	12.30	8.80	6.57	6.73	6.03	6.33	6.60	7.13	6.73	4.23	**3.30**
W-T-L	29-1-0	24-0-6	23-0-7	28-1-1	23-0-7	21-0-9	21-2-7	18-3-9	19-1-10	18-1-11	22-1-7	21-1-8	Base	13-2-15
W-T-L	29-1-0	22-0-8	27-0-3	29-1-0	27-0-3	22-0-8	24-2-4	19-4-7	21-2-7	22-1-7	23-1-6	26-1-3	15-2-13	Base
Fr. Test	✓	✓	✓	✓	✓	✓	✓				✓	✓	Base	–
Fr. Test	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	–	Base

For comparison, all the classifiers are ranked on each dataset in terms of AUC, with ranking of 1 is the best. We also perform Friedman tests on the ranks. After rejecting the null hypothesis using Friedman test that all the classifiers are equivalent, a post-hoc test called Nemenyi test [15] is used to determine the performance of which classifier is significantly better than the others.

4.3 Result and Discussion

Table 2 represents the comparison of 14 classifiers over 30 imbalanced datasets. The average ranks of these classifiers indicate that kNN is better performing algorithm compared to other traditional classifiers on imbalanced datasets. Though kNN performs better than C4.5, modifications of tree based algorithms for imbalanced datasets perform better than kNN. Moreover, kNN on SMOTE sampled datasets performs slightly better than kNN without sampling.

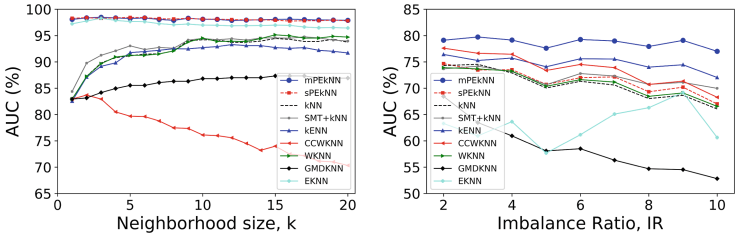
Now, if we compare kNN with its different variants, it can be observed that kENN and WKNN improve the overall performance of traditional kNN although another variant CCWKNN fails to improve the performance in most cases over the experimented datasets. Moreover, it is investigated that, the recent generalized mean based kNN approach GMDKNN performs worse than kNN on imbalanced datasets. In contrast, we can observe from Table 2 that, EKNN performs better than all other classifiers except the proposed sPEkNN and mPEkNN. It indicates that, handling uncertainty can improve the performance of kNN on imbalanced datasets. Finally, average ranks show that mPEkNN is the best performing classifier compared to others in the imbalanced datasets.

In addition, Table 2 summarizes the counts of Win-Tie-Loss (W-T-L) of sPEkNN and mPEkNN against other classifiers which indicates that mPEkNN performs better than other classifiers in most cases. From Win-Tie-Loss, it is observed that mPEkNN wins in at most 29 datasets with no loss against C4.5 and GMDKNN classifiers. In the least case, mPEkNN performs better in 19 datasets and worse in 7 datasets compared to EKNN.

The results of Friedman test (Fr. Test) with two base classifiers (sPEkNN and mPEkNN) are shown in the last two lines of the Table 2. From Friedman test with 14 classifiers and 30 datasets, we can conclude that, all the fourteen classifiers are not equivalent. After rejecting that all fourteen classifiers perform equivalent, Nemenyi test is performed to determine which classifier performs significantly better than the others. A tick(✓) sign under a classifier indicates that Nemenyi test suggests the performance of that classifier is significantly different from the base classifier in pairwise comparison at 95% confidence level. Nemenyi test states that, sPEkNN performs significantly better than all compared classifiers except EKNN, CCPDT and HDDT. More importantly, the test suggests that mPEkNN is the best performing classifier among twelve classifiers.

4.4 Effects of Neighborhood Size and Imbalance Ratio

Here, we show the effects of neighborhood size and Imbalance Ration (IR) on the performance of the proposed method compared to other kNN variants. Due to page limitations, only one dataset (Ionosphere) is used to present the comparison in terms of AUC with different the values of k ranging from 1 to 20. It is clear from Fig. 2a that sPEkNN and mPEkNN consistently perform better than the other algorithms and are less sensitive to the value of k.



(a) Effects of neighborhood size, k (b) Effects of Imbalance Ratio, IR

Fig. 2. Performance comparison among the algorithms belonging in kNN family

To visualize the effect of IR, we use a synthetic dataset of two-class problem in a two-dimensional space where instances of each class are taken from two Gaussian distributions. The characteristics of the dataset is given below where class-A is the minority class and Class-B is the majority class.

$$\eta_1^A = 0.6,\ \eta_2^A = 0.4,\ \mu_1^A = \begin{bmatrix}3\ 3\end{bmatrix}^T,\ \mu_2^A = \begin{bmatrix}-2\ -2\end{bmatrix}^T,\ \Sigma_1^A = 3I \text{ and } \Sigma_2^A = I$$
$$\eta_1^B = 0.9,\ \eta_2^B = 0.1,\ \mu_1^B = \begin{bmatrix}0\ 0\end{bmatrix}^T,\ \mu_2^B = \begin{bmatrix}4\ 3\end{bmatrix}^T,\ \Sigma_1^B = 8I \text{ and } \Sigma_2^B = I$$

Here η represents the mixture proportion and I is the identity matrix. Different datasets of 1500 samples are generated varying the class imbalance ratio ranging from 2 to 10. It is observable from Fig. 2b that, although the imbalance ratio increases, the performance of mPEkNN remains more steady compared to other kNN variants indicating less sensitivity of mPEkNN in these synthetic datasets.

5 Conclusion

This paper proposes an extended kNN algorithm to increase the performance of existing kNN by making it vigorous to imbalance class problem. In PEkNN, for a query instance, we calculate a confidence for each neighbor instance from the posterior probability of that instance which is then discounted by the proximity of that instance from the query instance. We show that this proximity weighted confidence increases the likelihood of classifying a minority class instance. To calculate the confidence we used two methods one using single Gaussian model

(sPEkNN) and other using Gaussian mixture model (mPEkNN). Results over 30 datasets provide the evidence that the proposed approach is better than twelve relevant methods in imbalanced datasets. However, one limitation of the proposed method is that we assume all the feature values as numeric. As future research direction, we have plan to extend the work for categorical features.

Acknowledgments. This research is supported by the fellowship from ICT Division, Ministry of Posts, Telecommunications and Information Technology, Bangladesh. Grant No - 56.00.0000.028.33.093.19-427; Dated 20.11.2019.

References

1. Akash, P.S., Kadir, M.E., Ali, A.A., Shoyaib, M.: Inter-node hellinger distance based decision tree. In: Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI (2019)
2. Alcalá-Fdez, J., et al.: KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. J. Multiple-Valued Logic Soft Comput. **17**(2), 255–287 (2011)
3. Bache, K., Lichman, M.: UCI machine learning repository (2013)
4. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: SMOTE: synthetic minority over-sampling technique. J. Artif. Intell. Res. **16**, 321–357 (2002)
5. Cieslak, D.A., Chawla, N.V.: Learning decision trees for unbalanced data. In: Daelemans, W., Goethals, B., Morik, K. (eds.) ECML PKDD 2008. LNCS (LNAI), vol. 5211, pp. 241–256. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87479-9_34
6. Denoeux, T.: A k-nearest neighbor classification rule based on dempster-shafer theory. IEEE Trans. Syst. Man. Cybern. **25**(5), 804–813 (1995)
7. Dubey, H., Pudi, V.: Class Based Weighted K-Nearest Neighbor over Imbalance Dataset. In: Pei, J., Tseng, V.S., Cao, L., Motoda, H., Xu, G. (eds.) PAKDD 2013. LNCS (LNAI), vol. 7819, pp. 305–316. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-37456-2_26
8. Dudani, S.A.: The distance-weighted k-nearest-neighbor rule. IEEE Trans. Syst. Man Cybern. **SMC–6**(4), 325–327 (1976)
9. Fix, E., Hodges Jr., J.L.: Discriminatory analysis-nonparametric discrimination: consistency properties. Technical report, California University, Berkeley (1951)
10. Gou, J., Ma, H., Ou, W., Zeng, S., Rao, Y., Yang, H.: A generalized mean distance-based k-nearest neighbor classifier. Expert Syst. Appl. **115**, 356–372 (2019)
11. Li, Y., Zhang, X.: Improving k nearest neighbor with exemplar generalization for imbalanced classification. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011. LNCS (LNAI), vol. 6635, pp. 321–332. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20847-8_27
12. Liu, W., Chawla, S.: Class confidence weighted kNN algorithms for imbalanced data sets. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011. LNCS (LNAI), vol. 6635, pp. 345–356. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20847-8_29
13. Liu, W., Chawla, S., Cieslak, D.A., Chawla, N.V.: A robust decision tree algorithm for imbalanced data sets. In: Proceedings of the 2010 SIAM International Conference on Data Mining, pp. 766–777. SIAM (2010)
14. McLachlan, G., Peel, D.: Finite Mixture Models. Wiley, Hoboken (2004)

15. Nemenyi, P.: Distribution-free multiple comparisons. Ph.D. thesis, Princeton University (1963)
16. Shafer, G.: A Mathematical Theory of Evidence, vol. 42. Princeton University Press, Princeton (1976)
17. Swets, J.A.: Measuring the accuracy of diagnostic systems. Science **240**(4857), 1285–1293 (1988)
18. Trafalis, T.B., Alwazzi, S.A.: Support vector regression with noisy data: a second order cone programming approach. Int. J. Gen Syst **36**(2), 237–250 (2007)
19. Wu, X., et al.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008). https://doi.org/10.1007/s10115-007-0114-2
20. Yang, Q., Wu, X.: 10 challenging problems in data mining research. Int. J. Inf. Technol. Decis. Making **5**(04), 597–604 (2006)