Engineering Science and Technology, an International Journal 28 (2022) 101011

HOSTED BY

Contents lists available at ScienceDirect

Engineering Science and Technology, an International Journal

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

Full Length Article

Quad division prototype selection-based k-nearest neighbor classifier for click fraud detection from highly skewed user click dataset

Deepti Sisodia *, Dilip Singh Sisodia

Department of Computer Science & Engineering, National Institute of Technology, Raipur, Chhattisgarh 492010, India

ARTICLE INFO

Article history:
Received 31 October 2020
Revised 5 May 2021
Accepted 23 May 2021
Available online 04 June 2021

Keywords:
Under-sampling
Fraudulent publishers
Nearest-neighbors
K-NN
Quad division
Class Imbalance
Prototype selection

ABSTRACT

In online advertising, the user-clicks dataset based fraudulent publishers' classification models exhibit poor performance due to high skewness in class distribution of the publishers. The nearest-neighbor based classification techniques are popularly used to reduce the impact of class skewness on performance. The Nearest-Neighbor techniques use Prototype Selection (PS) methods to select promising samples before classifying them for reducing the size of training data. Although Nearest-Neighbor techniques are simple to use and reduce the negative impact of the loss of potential information, they suffer from higher storage requirements and slower classification speed when applied on datasets with skewed class distributions. In this paper, we propose a Quad Division Prototype Selection-based k-Nearest Neighbor classifier (QDPSKNN) by introducing quad division method for handling uneven class distribution. The quad-division divides the data into four quartiles (groups) and performs controlled under-sampling for balancing class distribution. It reduces the size of the training dataset by selecting only the relevant prototypes in the form of nearest-neighbors. The performance of QDPSKNN is evaluated on Fraud Detection in Mobile Advertising (FDMA) user-click dataset and fifteen other benchmark imbalanced datasets to test its generalizing behaviour. The performance is also compared with one baseline model (k-NN) and four other prototype selection methods such as NearMiss-1, NearMiss-2, NearMiss-3, and Condensed Nearest-Neighbor. The results show improved classification performance with QDPSKNN in terms of precision, recall, f-measure, g-mean, reduction rate and execution time, compared to existing prototype selection methods in the classification of fraudulent publishers as well as on other benchmark imbalanced datasets. Wilcoxon signed ranked test is conducted to demonstrate significant differences amid QDPSKNN and state-of-the-art methods.

1. Introduction

Advancement and accessibility of online technology have prompted companies to market their products using web advertising. In addition, online advertising provides benefits to ad-network, advertisers, and publishers [1]. An ad-network like Google serves as an intermediary or broker between the advertiser and publisher by providing a platform for advertisements to be placed on the websites of publishers. Advertisements generate revenue for advertisers, who pay for displaying advertisements on other websites, since a user who clicks on these ads is directed to their site. The publisher is a website like blogs or news websites that displays advertisements and is charged by the ad-network as per the number of generated clicks on the ads while revenue is taken from the advertiser [2–5]. However, revenue earned by publishers on every generated click can induce click fraud, which refers to the malicious clicks generated to earn more revenue and deplete the competitor's budget [6–8].

The generation of hundreds of millions of clicks per day by the users results in massive datasets, which hinders the identification of click fraud. Moreover, only a small percentage of the total number of generated clicks is fraudulent, which results in dataset imbalance [9], i.e., one class of dataset comprises a higher number of samples than the other [10]. In such cases, learning methods fail to appropriately represent the data characteristics [11]. Miscellaneous data comprising highly uneven distribution in the classes aggravates the issue of fraudster identification. Detecting click fraud generally relies on machine learning models [12], which use an enormous number of features on large datasets. Thus, training models and their use for lookups tend to be costly. The issue of data imbalance can be resolved by utilizing widely used data sampling strategies like oversampling methods that oversample the

* Corresponding author.
E-mail address: dsisodia.phd2017.cse@nitrr.ac.in (D. Sisodia).
Peer review under responsibility of Karabuk University.

https://doi.org/10.1016/j.jestch.2021.05.015
2215-0986/

under-represented class or under sampling methods that under samples the over-represented class. However, oversampling the minority instances may lead to overfitting and may slow-down the learning process. While the use of prototype selection under sampling methods may improve the efficiency, which reduces the size of the training set by performing controlled under sampling that selects specified samples from the dataset [13]. Some well-trained near-neighbor based machine learning models can be utilized to effectively deal with imbalanced datasets during classification. However, near-neighbor methods like k-NN suffers from two major issues during classification: a) Need for huge memory space, to store all instances and their distances to neighbor instances for the classification [14,15]. b) Lower classification speed, as the search of all samples towards classifying the test instance, slows down the classification process [16]. To further improve the effectiveness of classification, prototype selection methods discussed in references [17–23], are often employed that select the samples to retain instead of deleting them from the dataset. Thus, utilizing Prototype Selection, which is a type of data reduction method, helps in modeling k-NN towards saving memory resources and speeding up the processing time [14,18].

The present study proposes Quad Division Prototype Selection based k-NN classifier (QDPSKNN) that reduces the size of data by performing controlled under-sampling on majority instances. Controlled under sampling allows to select specified prototypes (nearest neighbors) from the majority data. The designed method selects prototypes using the k-nearest neighbor rule. QDPSKNN first divides the majority training data into four quartiles (groups) Q_1 to Q_4 where each quartile or group comprises 25% of entire majority instances. k-NN is then applied individually to each group to select specified prototypes, where k-NN measures the distance between the query instance (test sample) and all the majority instances in each group. Query instance is chosen by taking median of the samples in each group. Thereafter, k-instances are selected as promising prototypes from each group equivalent to the number of minority instances to make the dataset balanced. Unlike the conventional under sampling approaches which balances the dataset by deleting the majority instances, QDPSKNN performs under sampling by selecting promising prototypes for data balancing. QDPSKNN is highly effective because it reduces the negative impact of the loss of significant information while increasing the positive influence of data cleaning in the under-sampling process. It effectively extracts informative prototypes as nearest-neighbors, enhancing the prediction of the designed method. The following are the key contributions of the work:

- In this work, a Quad Division Prototype Selection-based k-Nearest Neighbor classifier (QDPSKNN) is proposed, which performs controlled under-sampling by selecting prototypes based on k-nearest neighbor rule for balancing skewed class distribution in the user-click dataset.
- The performance of QDPS based k-NN strategy is evaluated on Fraud Detection in Mobile Advertising (FDMA 2012), a raw user-click data of online advertising so as to classify fraudulent publishers.
- To evaluate the effectiveness of QDPSKNN, a fair comparison is performed by conducting extensive experiments on fifteen other benchmark imbalanced datasets to test its generalizing behavior.
- The effectiveness of the QDPSKNN is also compared with one baseline model (k-NN) and four other prototype selection methods, namely NearMiss-1, NearMiss-2, NearMiss-3, and Condensed Nearest Neighbor (CNN). Its efficiency is demonstrated in handling the challenging issues encountered in the classification of fraudulent publishers.
- Wilcoxon signed ranked test is conducted to demonstrate significant differences amid QDPSKNN and state-of-the-art methods.

The remaining text of this paper is organized under the following sections. Related works on prototype selection methods for controlled under-sampling are discussed in detail in section 2. The conventional k-NN model is discussed in section 3. The proposed methodology is elaborated in section 4, while necessary evaluation measures are presented in section 5. The experimental results are discussed in detail in section 6, while the discussion is presented in section 7. The paper is concluded in section 8.

2. Related works

Data reduction methods [23] facilitate the execution of data-mining techniques by improving the datasets. Datasets after reduction comprise the properties of factual knowledge of the actual dataset. Data reduction methods can be divided into two main categories: prototype generation and prototype selection. PS approaches elects a set of training samples from the original dataset as a prototype, and handles real data efficiently using less memory. This approach use data from the actual training set by finding border instances and discarding noisy samples [24]. Data reduction using k-NN can alleviate the consumption of memory and exacerbates the process of classification. A generalized mean distance (GMD) based k-NN model (GMDKNN) [25] utilizes the multi-GMD and nested-GMD, which computes the multi-local mean vector of the query instance in each class by opting class specified k-closest neighbors. The k-GMD's are computed correspondingly using the obtained per class k-local mean vectors, which are then further used towards designing the categorical GMD. Another k-NN model based on local mean representation (LMRKNN) [26] first selects the categorical nearest neighbors of a query instance to compute the respective categorical k-local mean vectors (k-LMV). The query instance is then represented using the categorical k-LMV's linear combination. Thereafter, the class specified representation-based distances among the query instance and the categorical k-LMV are chosen for determining the query instance's class. Weighted representation and weighted local mean representation based two k-NN rules [27] are designed with an aim of improving the classification performance of k-NN. The designed methods consider the localities of categorical k-neighbors and computes the adaptive weights of categorical k-NN and the multi local mean vectors for classification. Integrating the course to fine classification with the k-NN (CFKNNC) [28] is an optimal k-NN strategy that choses less amount of training instances closer to the test instance from the actual training instances. The method then determines the k-training instances similar of the test instance by computing a representation-based distance which performs more accurate classification than the conventional k-NN. Toward selecting the relevant subset from the actual dataset, the CNN technique was the first prominent condensation approach [29]. It creates a subset of samples that can correctly classify the original data set, utilizing the 1-Nearest-Neighbor rule. It retains the class border instances and discards internal instances as miss-classified instances are placed closer to the decision boundary. A geometric median (GM) [30] based enhanced PS method has been introduced for speeding up the election method, which is a data reduction approach for k-NN that utilizes GM as a class prototype. The method reduces the summation of the distances to the rest of the samples. Novel algorithms for dimensionality reduction and prototype selection on multi-label data has been designed for which CNN has been extended along with an exten-

sion of the Class Augmented PCA(CA-PCA) for multi-label data [31]. The NearMiss approaches [32,33] conducts under-sampling by discarding the samples from over-represented class on the basis of their distances among one another. The first experiment was carried out with "NearMiss-1" (NM-1), which elects the more numbered class samples close to less numbered class samples. The approach chooses majority class instances while their mean distance to three nearest minority class instances is smallest. The second experimentation is performed with "NearMiss-2" (NM-2), which selects the higher numbered class instances, where average distances to three furthermost less figured class instances are the smallest. The next approach was the "NearMiss3" (NM-3) that selects specified nearest higher-numbered class instances for every less numbered class instance. The results of the experiments demonstrate the superior performance of the NearMiss-2 approach. NM-1, NM-2, NM-3, and "Most Distant" [34] were k-NN under-sampling approaches discussed in. Rather than utilizing the complete set of examples of over-represented class, a smaller subset of such instances is elected to result in less imbalanced data. The NM-1 elects those over-represented instances whose mean distance to the three nearest under-represented class instances are the smallest, while NM-2 elects the over-represented instances with their mean distance to the three farthest under-represented class instances are the most minor. NM-3 differs with NM-1 and NM-2 by selecting specified nearest over-represented instances for every under-represented class instance to warranty that each less numbered samples are surrounded by highly numbered instances. At last, 'most distant' approach elects those out-numbered class instances with the highest mean distance to the 3- nearest less-numbered class samples. The results demonstrated superiority of NM-2, corresponding to the rest of the under-sampling approaches towards skewed learning. NM approaches are highly effective in cleaning the decision surface by maximizing the distance amidst minority and majority samples.

The k-NN methods and data reduction techniques discussed in the literature outperforms conventional k-NN but suffers with higher computational complexity due to the selection of border and internal samples. Several prototype selection methods elect each of the instances as per the score of its closest neighbor. The process of searching the closest neighbors on a vast dataset consumes much time. Our proposed approach overcomes the issues by speeding-up the process of prototype selection by choosing only promising prototypes from four equal groups of majority instance-set using the Quad Division method.

3. Nearest-Neighbor based prototype selection methods

3.1. Existing k-Nearest-Neighbor model

In this section, we will discuss the modeling of k-NN [33], and the problems associated with imbalanced classification. k-NN functioning, shown in Fig. 2(a) includes determination of the distance amidst the query instance and the rest of the instances of the dataset, election of specific k-instances closer to the query point, and assignment of the most frequent label through majority voting for classification. Let the labeled training set (f_i, c_i), where$i = 1, \cdots, p$ and $c_i \in \{z_1, z_2, \cdots\cdots, z_p\}$. c_i indicate the class label while $z_1, z_2, \cdots\cdots, z_p$ represents different classes of c_i. k-NN method searches a set of k-prototypes from the data which are nearest to the query instance f_t using any distance measure like the Euclidean distance, Manhattan distance etc., and computes the label of query instance as per the class predominance in the neighborhood.

The working of k-NN for the given query instance f_t using the majority voting[3536] is expressed as follows:

$$c_t = z \in \{z_1, z_2, \ldots, z_p\} \overset{\arg max}{\sum_{f_i \in M(f_t,k)}} E(c_i, z) \tag{1}$$

where $c_t$denotes predicted class towards the query instance (f_t), z represents the set of different classes $z_1, z_2, \cdots\cdots, z_p$ and p indicates the total number of classes.

$M(f_t, k) = set\ of\ k - prototypes\ nearest\ to\ f_t$

$$E(c_i, z) = \begin{cases} 1\ if\ c_i = z \\ 0 \quad else \end{cases} \tag{2}$$

where, $E(c_i, z)$ represents an indicator-function which returns 1 for true condition otherwise returns 0 for false condition.

Equation-1 could also be expressed as

$$c_t = \arg max \left\{ \sum_{f_i \in M(f_t,k)} E(c_i, z_1), \sum_{f_i \in M(f_t,k)} E(c_i, z_2), \cdots\cdots, \sum_{f_i \in M(f_t,k)} E(c_i, z_p) \right\} \tag{3}$$

$$c_t = \arg max \left\{ \sum_{f_i \in M(f_t,k)} \frac{E(c_i, z_1)}{k}, \sum_{f_i \in M(f_t,k)} \frac{E(c_i, z_2)}{k}, \cdots\cdots, \sum_{f_i \in M(f_t,k)} \frac{E(c_i, z_p)}{k} \right\} \tag{4}$$

and it is known that

$$x(z_j)_{(f_t,k)} = \sum_{f_i \in M(f_t,k)} \frac{E(c_i, z_j)}{k} \tag{5}$$

where, $x(z_j)_{(f_t,k)}$ shows the probability of occurring of j^{th} class in the neighbor of f_t. Thus, Eq. (4) turns as

$$c_t = \arg max \left\{ x(z_1)_{(f_t,k)}, x(z_2)_{(f_t,k)}, \cdots .., x(z_n)_{(f_t,k)} \right\} \tag{6}$$

Equation (6) shows that standard k-NN employs prior probabilities towards calculating query instance class.

Given the mathematical formulation of traditional k-NN, we will now discuss the disadvantages of the k-NN towards dealing with the unbalanced dataset. Fig. 2 (a) shows a sample scenario of binary skewed class distribution, where the majority of the instances (represented as green plus sign) belong to the class "A," and the minority samples (served as blue minus sign) belong to class "B." The query instance is shown as a red plus sign. It is observed from the figure with $k = 7$; standard k-NN will classify the query instance towards the majority class A. But if the method takes into account the skewed distribution of classes around the neighborhood of the query instance (dotted-square region), then the query instance will be classified to minority class B [37]. In order to overcome the above reported issues of k-NN, we proposed QDPSKNN which is briefly discussed in Section 4.

3.2. Other baseline prototype selection methods for controlled under-sampling

For uneven class distribution in the dataset, we investigated four other prototype selection methods that select instances to 'keep' for controlled under-sampling.

3.2.1. NearMiss

NearMiss [33,38–40] belongs to a family of under-sampling which aims in balancing the distribution of classes by randomly alienating majority class samples. When samples of two distinct classes are closer to each other, these methods eliminate the samples of majority class towards increasing the spaces amid two classes, which aids in the process of classification. Near-neighbor

based NearMiss methods first searches the distances amid all the majority and minority class samples. Thereafter, n majority samples that having the smallest-distance to the instances in the minority class are selected. For k minority class samples, these near-neighbor based methods results in k × n majority class samples. The working of near-neighbor based different versions of NearMiss methods (NearMiss-1, NearMiss-2, NearMiss-3) is pictorially depicted in Fig. 2(b-d) and discussed as follows.

- NearMiss-1: elects the instances of the outnumbered class whose average distance to 3 nearest less-frequent class instances is the smallest.
- NearMiss-2: elects the instances of the outnumbered class whose average distance to 3 farthermost less-frequent class instances is the smallest.
- NearMiss-3: elects a specified nearest instance of the outnumbered class for every instance of the less-frequent class.

3.2.2. Condensed Nearest-Neighbor

CNN is designed for reducing the data samples towards classification using k-NN. It elects a set of prototypes p from the training samples in such a manner that k-NN with p can classify the instances exactly as k-NN does with the complete dataset [29]. The basic working of CNN is pictorially depicted in Fig. 2(e).

4. Proposed method

4.1. Quad-Division prototype selection based k-NN (QDPSKNN) for controlled Under-sampling

This section presents the principle of quad-division based k-nearest neighbor classifier QDPSKNN for controlled under sampling, the basic working is shown in Fig. 2(f). Dataset is usually segmented into quartiles, deciles, and percentiles [41]. 'Quartiles' further segment the data into 4 parts; 'deciles' segment the data into ten parts; and 'percentiles'- segment the data into hundred parts [42]. However, segmenting the data into numerous parts may increase classification complexity; thus, the majority data is segmented using the quad division process. The proposed method divides the majority data into four equal quartiles or groups $Q_1 - Q_4$where each group comprises 25% of the entire instances as shown in Fig. 1 [43–45].

Dividing the majority data into groups gauges the spread of values below and above the mean. Also, makes it easier to analyze the outliers in the dataset [46]. The proposed method is based on the k-NN presumption that observations with similar characteristics tend to have similar outcomes. Variations are made on the k-NN algorithm towards selecting the promising prototypes. The modified k-NN is expressed as follows-

The complete dataset D is a set of majority and minority class instances and can be expressed as:

$$D = \{f_1, f_2, \cdots ..., f_p\} \tag{7}$$

$$D = d_{mn} + d_{mj} \tag{8}$$

where, p is the total number of instances of entire dataset D, $d_{mn}$is the set of minority instances and d_{mj} is the set of majority instances of D and can be defined as:

$$d_{mj} = \{f_1, f_2, ..., f_n\} \tag{9}$$

$$d_{mn} = \{f_{n+1}, f_{n+2}, \cdots, f_p\} \tag{10}$$

The majority dataset $d_{mj}$is then divided into four quartiles or group $Q_1 - Q_4$using the Equations 11–13.

$$Q_1 = \left(\frac{1(n+1)}{4}\right)^{th} \text{value} \tag{11}$$

$$Q_2 = \left(\frac{2(n+1)}{4}\right)^{th} \text{value} \tag{12}$$

$$Q_3 = \left(\frac{3(n+1)}{4}\right)^{th} \text{value} \tag{13}$$

The query instance or test instance (f_t) is then chosen by computing the median of instances in each group $Q_1 - Q_4$. Since the median separates the higher half of instances from the lower half, it is a highly reliable tool to measure the data in uneven distributions. It is also less sensitive towards outliers. Query instance is obtained by computing the median from the instances of a quartile as:

$$Median = \begin{cases} \frac{n}{2} if\ n\ is\ even \\ \frac{n+1}{2} if\ n\ is\ odd \end{cases} \tag{14}$$

where n is the number of instances in a quartile or group.

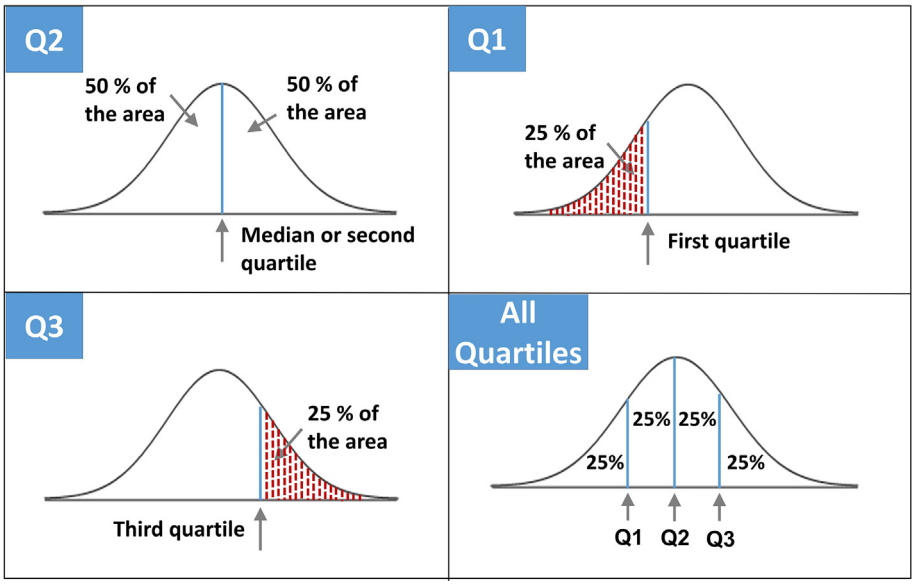

Fig. 1. Dividing the majority data into Quartiles.

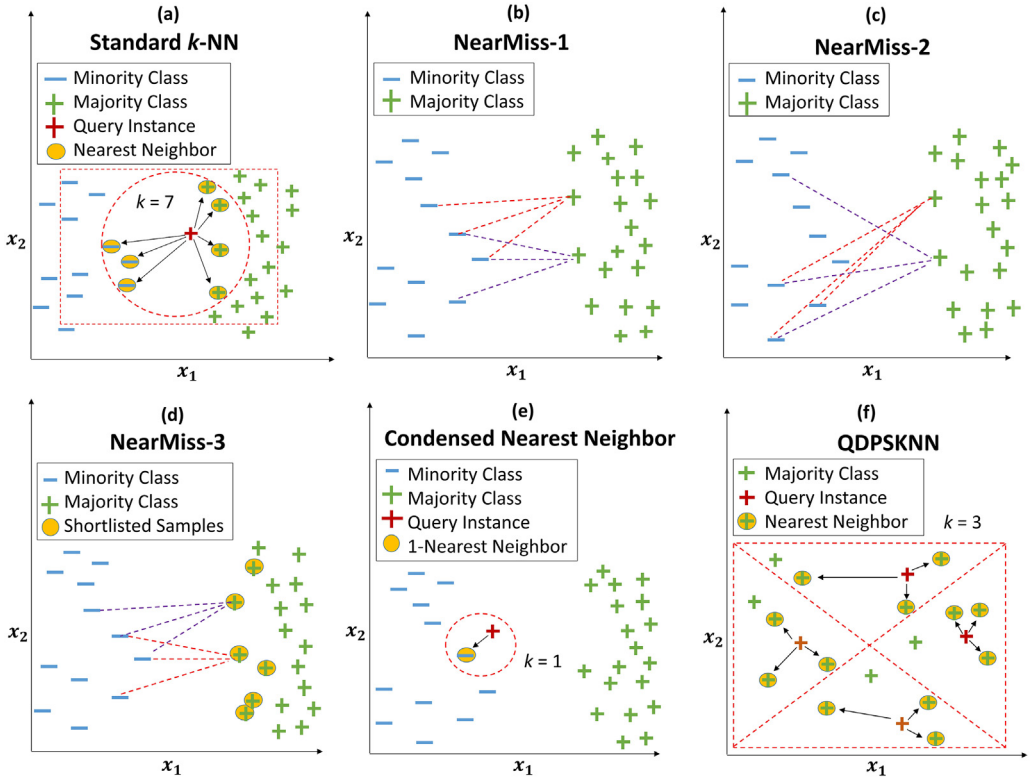

Fig. 2. Intuitive explanation of existing PS methods and proposed QDPSKNN for controlled under-sampling.

The method then searches for a set of promising relevant k-prototypes from each group based on Euclidean distance measure between query instance f_t and instances of each group f_{Qi}, $i = 1, 2, 3$ as:

$$dist(f_t, f_{Qi}) = \sqrt{\sum_{i=1}^{n} (f_t - f_{Qi})^2} \tag{15}$$

The proposed approach then selects the promising prototypes (nearest neighbors) pp from $Q_1 - Q_4$as

$$pp_1 \in Q_1, pp_2 \in Q_2, pp_3 \in Q_3, pp_4 \in Q_4 \tag{16}$$

The promising prototypes pp obtained from $Q_1 - Q_4$ are then aggregated to create $d_{mj}{}'$ such that the number of samples of $d_{mj}{}'$ are less than the number of samples of d_{mj}as

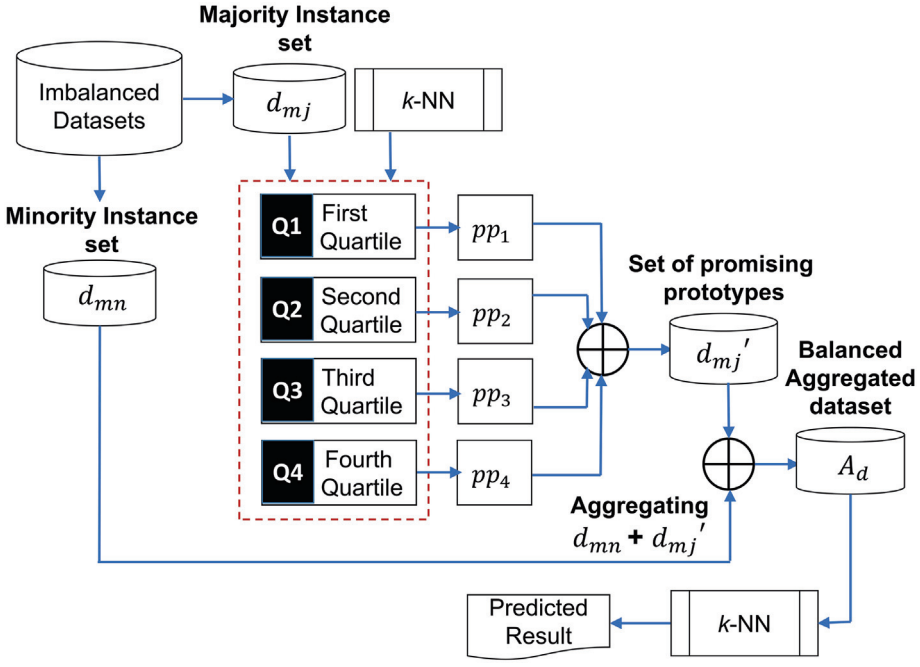

Fig. 3. Workflow of proposed QDPSKNN method.

$$d'_{mj} = \{pp_1 + pp_2 + pp_3 + pp_4\} \quad (17)$$

$$\left|d'_{mj}\right| < \left|d_{mj}\right|, d'_{mj} \in d_{mj}$$

where, $d_{mj}{}'$ =a set of significant prototypes selected from each group of d_{mj} where the number of observations are defined by the user.

The proposed method then aggregates the promising nearest-neighbors of $d_{mj}{}'$ with minority dataset$d_{mn}$and generates a balanced aggregated dataset $A_d$of reduced size as:

$$A_d = d_{mj}{}' + d_{mn} \quad (18)$$

Finally, for classification, conventional k-NN is modeled on a new aggregated dataset $A_d$as:

$$F = KNeighborsClassifier(A_d, k) \quad (19)$$

where, k is the number of nearest neighbors. Key benefits of the proposed method are as follows-

- Unlike the conventional k-NN, where the cost of distance computation between query instance and rest of all dataset samples is higher, QDPSKNN, being an under-sampling strategy overcomes the issue by computing the distance only between each group's query instance and majority instances and thus lowers the computational cost.
- Unlike the conventional under sampling approaches which deletes the instances from the majority set to make the dataset balanced that results in loss of information, QDPSKNN selects the relevant promising prototypes to balance the skewed class distribution, thus, enhances the classification procedure. As a consequence, it is more efficient and scalable.
- Another advantage of the proposed method over the standard k-NN is that for f_p samples of the minority datasets d_{mn}, the proposed method selects $\frac{f_p}{4}$ samples from each group, equivalent to the number of minority instances to balance the dataset. In comparison, standard k-NN selects f_n instances, having the smallest distance to that of the query instance from the entire dataset.

Below is the pseudo-code of QDPSKNN for controlled under-sampling. Fig. 3 shows the visual representation of the workflow of proposed method, while the abbreviations used to define proposed method are discussed in Table 1.

Pseudo-code: Quad-Division Prototype Selection based *k*-NN (QDPSKNN)

Input:
D = dataset
d_{mj} = set of majority instances of dataset$D$
d_{mn} = set of minority instances of dataset$D$
Q_i = Quartile
f_t = query instance
f_{Qi} = instances of each group
pp = promising prototypes
$d_{mj}{}'$ = a set of promising prototypes obtained from quartiles
Output:
A_d = aggregated dataset // balanced dataset by aggregating promising nearest-neighbors of $d_{mj}{}'$ with minority datasetd_{mn}
Method:
1. Loading the D which is a set of majority and minority class instances$D \leftarrow \{d_{mn}, d_{mj}\} D \leftarrow \{f_1, f_2, \cdots .., f_p\}$

Table 1
. Abbreviations used in proposed method.

Symbols	Meaning	Symbols	Meaning
D	dataset	f_t	query instance
d_{mj}	set of majority instances of dataset$D$	f_{Qi}	instances of each group
d_{mn}	set of minority instances of dataset$D$	pp	promising prototypes
Q_i	Quartile	$d_{mj}{}'$	a set of promising prototypes obtained from quartiles
A_d	aggregated dataset	c_i	Class label
f_i	Instance of dataset$D$	c_t	predicted class
z	set of different classes$z_1, z_2, \cdots\cdots, z_p$	$M(f_t, k)$	*set of k – prototypes nearest to* f_t

Table 2
Summarized details of imbalanced benchmark datasets.

Datasets	Type	Features	Instances	Classes	MV	IR
FDMA2012 [D1]	Multiclass	103	2523	03	No	17.1
Wine [D2]	Multiclass	13	178	03	No	1.5
Balance [D3]	Multiclass	4	625	03	No	5.88
New Thyroid [D4]	Multiclass	5	215	03	No	4.84
Thyroid [D5]	Multiclass	21	720	03	No	36.9
Pageblocks [D6]	Multiclass	10	548	05	No	164
Glass [D7]	Multiclass	9	214	07	No	8.44
Ecoli [D8]	Multiclass	7	336	08	No	71.5
Ecoli2 [D9]	Binary	7	336	02	No	5.46
Pima [D10]	Binary	8	768	02	No	1.87
Ecoli3 [D11]	Binary	7	336	02	No	8.6
Glass6 [D12]	Binary	9	214	02	No	6.38
Glass1 [D13]	Binary	9	214	02	No	1.82
Iris [D14]	Binary	4	150	02	No	2.0
Haberman [D15]	Binary	3	306	02	No	2.78
Wisconsin [D16]	Binary	9	683	02	No	1.86

Note: MV- Missing Values, IR-Imbalance Ratio

* (*continued*)

Pseudo-code: Quad-Division Prototype Selection based *k*-NN (QDPSKNN)

2. Splitting the majority and minority instances from the$Dd_{mj} = \{f_1, f_2, \cdots, f_n\} d_{mn} = \{f_{n+1}, f_{n+2}, \cdots, f_p\}$
3. Dividing majority dataset d_{mj}into four equal quartiles. Let Q_i be the i^{th} quartile.$Q_i = \left(\frac{i(n+1)}{4}\right)^{th} value,\ i = 1, 2, 3$
4. Obtaining query instance $f_t$by computing the median from the instances of each quartile$Q_i Median = \begin{cases} \frac{n}{2} if\ n\ is\ even \\ \frac{n+1}{2} if\ n\ is\ odd \end{cases}$
5. Searching the promising relevant *k*-prototypes from $Q_i$based on Euclidean distance between query instance $f_t$and all instances of each groupf_{Qi} $dist(f_t, f_{Qi}) = \sqrt{\sum_{i=1}^{n} (f_t - f_{Qi})^2}$
6. Selecting the promising prototypes (nearest neighbors) *pp* from$Q_i pp_i \in Q_i$
7. Aggregating the *pp* to create a set of$d_{mj}{}' = \sum_{i=1}^{4} pp_i \left|d_{mj}{}'\right| < \left|d_{mj}\right|, d_{mj}{}' \in d_{mj}$
8. Generating a balanced aggregated dataset A_d by aggregating the promising nearest-neighbors of $d_{mj}{}'$ with minority dataset$d_{mn} A_d = d_{mj}{}' + d_{mn}$

5. Evaluation measures

Precision, recall, and f-measure are often used to appropriately monitor the performance of classification in distinct classes in order to achieve an optimum classification (balanced). In this work, precision is the key measure of confidence as it focuses on singling out illegitimate clicks while minimizing the chances of predicting legitimate clicks as fraud ones [1].

- *Precision*[47,48]: It is defined as the percentage of actual false clicks among all the clicks predicted to be a fraud.

$$Precision = \frac{fraudsters\ identified\ correctly}{fraudsters\ identified\ correctly + legitimates\ wrongly\ labeled\ as\ fraudsters}$$

- *Recall*[49]: It is defined as the percentage of fraudulent clicks correctly identified by the model from the fraudulent clicks.

$$Recall = \frac{fraudsters\ identified\ correctly}{fraudsters\ identified\ correctly + fraudsters\ wrongly\ labeled\ as\ legitimates}$$

- *F1 score* [50]- It computed the weighted-average of precision and recall into a single metric.

$$F1 = 2 * \frac{precision * recall}{precision + recall}$$

- *G-mean*- It indicates how well the model performs at the threshold, where the true positive rate and true negative rate are equal. Maximizes true positive and true negative rate respectively.

$$GM = \sqrt{True\ Positive\ rate * True\ Negative\ rate}$$

where, TP = True Positive indicates the fraudulent clicks predicted by the model as fraudulent. TN = True Negative indicates genuine clicks predicted by the model as normal. FP = False Positive indicates genuine clicks predicted by the model as fraudulent. FN = False Negative indicates the fraudulent clicks predicted by the model as normal.

Other than the metrics discussed above, one more performance metrics is used in this work towards comparing the performance of QDPS based *k*-NN with other prototype selection methods. The purpose of prototype selection methods is the reduction of storage requirements. We thus evaluated the reduction rate in this work.

- *Reduction Rate*: It is the percentage of the size of the reduced training dataset for the size of the whole training dataset [30]. It is defined as:

$$Reduction\ Rate = 1 - \frac{(size(R_s))}{(size(T_r))}$$

where$size(R_s)$ = samples of the reduced training set and $size(T_r)$ = samples of the whole training set. The value of the reduction rate should range from 0 to 1. The higher the reduction rate, the training set will be better reduced by the method, and less will be the memory and classification time.

- *Execution Time*- It is the overall time spent on designing a complete model.

6. Experimental results

The experiments were carried out on a personal system with an Intel(R) Core(TM) i5-7200U CPU running at 2.70 GHz with 12 GB RAM with a Windows-10 operating system. The proposed method was implemented in Python version 3.7 with libraries including numpy[51], scipy[52], sklearn[53], imblearn[54] and pandas[55] from the Anaconda package[56]. QDPSKNN was designed for controlled under-sampling to generalize well with uneven class distributions encountered in modeling the online advertising user-click data for the classification of fraudsters. The training and testing of the used methods on all datasets was implemented via 10-fold cross-validation.

6.1. Datasets

In this empirical study, total sixteen datasets were employed for evaluating the effectiveness of proposed QDPSKNN, where user clicks dataset is sourced from FDMA [57] and fifteen other standard

Table 3
Parameters tuned for baseline method k-NN and state-of-the-art prototype selection methods.

Methods	Parameters
NearMiss-1	random_state = 0, sampling strategy='not minority', version = 1
NearMiss-2	random_state = 0, sampling strategy='not minority', version = 2
NearMiss-3	random_state = 0, sampling strategy='not minority', version = 3
CNN	random_state = 0, sampling strategy='not minority', n_neighbors = 1
k-NN	n_neighbors = 5, p = 2, random_state = 0

Table 4
Performance comparison of QDPS, k-NN and other PS methods based on Average Precision and Recall.

DS	Precision						Recall					
	QDPSKNN	k-NN	NM1	NM2	NM3	CNN	QDPS	k-NN	NM1	NM2	NM3	CNN
D1	75.1	30.9	48.6	63.9	46.3	28.6	70.1	28.1	45.3	59.4	41.7	24.6
D2	90.2	70.9	72.9	69.2	45.1	37.3	84.7	65.3	70.1	64.2	41.5	32.7
D3	74.4	57.8	29.4	70.1	47.2	46.7	70.4	55.9	23.4	66.2	39.2	41.2
D4	72.3	97.7	88.8	96.2	90.4	74.6	66.3	91.0	85.7	94.4	88.8	70.3
D5	88.5	30.5	72.2	80.5	35.7	34.9	82.4	27.1	69.8	76.1	30.8	29.4
D6	86.4	56.7	78.6	79.6	52.9	51.6	80.3	50.1	75.2	74.2	48.9	48.1
D7	89.6	84.2	70.3	82.4	64.1	60.4	84.1	80.5	65.3	78.4	60.1	57.4
D8	76.1	80.1	65.4	90.6	72.3	77.1	71.3	75.1	62.5	85.4	68.4	73.5
D9	82.9	79.1	80.9	76.3	64.9	77.0	78.9	73.2	76.4	74.2	61.3	72.2
D10	89.4	63.8	79.3	81.4	60.1	58.5	85.4	60.5	76.1	77.5	58.3	55.9
D11	93.5	90.8	64.2	58.8	45.2	58.3	90.4	86.7	61.8	55.2	40.3	55.6
D12	93.6	91.2	75.7	87.5	100	68.7	90.8	87.5	73.8	83.3	98.4	66.1
D13	75.7	72.7	64.5	71.8	70.1	56.5	70.5	69.2	61.3	68.7	66.5	53.9
D14	83.5	69.8	83.0	77.2	70.2	48.2	79.4	63.8	80.4	73.6	65.1	45.6
D15	75.4	62.5	69.8	72.1	62.3	38.4	70.4	58.3	64.6	67.4	58.4	35.1
D16	91.5	49.5	60.3	78.2	44.5	44.2	86.3	44.5	75.3	55.1	41.3	40.6
AV	**83.6**	**68.0**	**69.0**	**77.2**	**60.7**	**53.8**	**78.9**	**63.6**	**66.7**	**72.1**	**56.8**	**50.1**
SE	**0.02**	**0.05**	**0.04**	**0.02**	**0.04**	**0.03**	**0.02**	**0.05**	**0.04**	**0.03**	**0.05**	**0.03**

Note: DS- Datasets, AV- Average and SE- Standard Error

imbalanced benchmark datasets collected from the KEEL dataset repository [58]. Table 2 represents the summarized detail of highly imbalanced datasets comprising less proportion of positive samples. The datasets significantly differ in data size consisting of hundreds to thousands of samples, number of classes, features and imbalance ratio. The datasets do not contain any noise or missing values as stated in Table 2. The performance of datasets is assessed utilizing metrics, namely- precision, recall, f-measure, g-mean, reduction rate and execution time.

6.2. Classification performance

Table 3 represents the list of parameters tuned in experimenting. The modeling procedure was conducted separately for all the measures. All the employed methods were executed once only as there is no change in their performance due to fixed parameters during classification. PS methods were run using training instances for creating the reduced dataset. Table 4 and Table 5 show the performance of k-NN, proposed QDPSKNN, and state-of-the-art PS methods, respectively, based on average precision, recall, f-measure, and g-mean. The performance was summarized by computing the average and standard error. The average, which represents the significance of a set of not-equal values, was computed to summarize an amount of data into a single value. The standard error represents the standard-deviation of sample means, and defines the accuracy of the mean of a sample from a population that is likely to be compared to the true population-mean. The smaller the standard error, the more representative the sample will be of the overall population. It is calculated as,

$Se = \frac{\sigma}{\sqrt{n}}$, where σ = standard deviation and n = number of observations

Results from Table 4-5 show that the proposed method achieved the best precision score on thirteen datasets out of a total of 16 datasets, while NearMiss-2 achieved the second-best precision score on nine datasets out of a total of 16 datasets. Fig. 4 (a-b) graphically represents the summarized performances of all methods based on computed average and standard error of precision, recall, f-measure and g-mean on sixteen datasets namely FDMA2012, New Thyroid, Ecoli2, Balance, Thyroid, Ecoli3, Glass6, Glass1, Wine, Pima, Iris, Haberman, Wisconsin, Pageblocks, Glass, and Ecoli. Similarly, the results demonstrate the effectiveness of the proposed method with other PS methods in handling skewness in class distribution. QDPSKNN method achieved superior performance with the lowest standard error compared to the other methods on all datasets.

6.3. Comparison of reduction rate and execution time of QDPS and other PS methods

The prototype selection methods are mainly used to reduce the storage requirements and increase the classification speed; thus, reduction rate and execution time are evaluated, respectively. The reduction rate denotes the percentage of the reduced size of the training dataset for the size of the whole training dataset. In comparison, the execution time evaluates the amount of time spent in building the model. Usually, PS methods obtain higher reduction rates corresponding to average learning performance, we also compared the reduction rate and execution time of QDPSKNN with the rest of the prototype selection methods. Table 6 represents the reduction rate and execution time based on average and standard error. The proposed method received the highest average reduction rate of 77.1% obtained from all the datasets with the standard error of 0.04%, which represents highly informative samples.

The reduction rate of 93.9% was achieved on the FDMA dataset utilizing the Quad-Division method, which divides the dataset into four equal groups and selects promising prototypes from each of the groups and balances the dataset appropriately. NM2 has obtained the second-highest average reduction score, which balances the dataset by computing the minimum distance among majority instances and three farthest minority samples. NM3 has the third-lowest reduction score compared to the proposed method and NM2, as it keeps the samples of majority class whose average distance to the three nearest and three farthest minority class instances are the smallest. CNN obtained the fourth-highest reduction score, which keeps only those instances which are closer to the decision boundary. NM1 received the fifth lowest reduction scores among all the PS methods as it balances the dataset by computing the minimum distance among majority instances and three nearest minority samples. The execution time of the proposed method with baseline model, and existing PS methods are represented in Table 6 for sixteen imbalanced datasets, where the average execution time of QDPSKNN was found to be the smallest among all datasets. Fig. 4 (c-d) pictorially shows the summarized performances of all employed methods based on computed aver-

Table 5
Performance comparison of QDPS, k-NN and other PS methods based on F-measure and G-mean.

DS	F-Measure						G-mean					
	QDPSKNN	k-NN	NM1	NM2	NM3	CNN	QDPS	k-NN	NM1	NM2	NM3	CNN
D1	72.5	29.4	46.9	61.6	43.9	26.4	73.3	29.9	47.5	62.1	44.5	27.2
D2	87.4	68.0	71.5	66.6	43.2	34.8	88.1	68.4	72.1	67.2	43.9	35.6
D3	72.3	56.8	26.1	68.1	42.8	43.8	73.1	57.2	26.8	68.6	43.6	44.6
D4	69.2	94.2	87.2	95.3	89.6	72.4	69.9	94.7	87.8	95.8	90.2	73.1
D5	85.3	28.7	71.0	78.2	33.1	31.9	86.1	29.1	71.6	78.8	33.8	32.7
D6	83.2	53.2	76.9	76.8	50.8	49.8	84.0	53.7	77.5	77.4	51.5	50.5
D7	86.8	82.3	67.7	80.4	62.0	58.9	87.5	82.7	68.4	80.9	62.7	59.6
D8	73.6	77.5	63.9	87.9	70.3	75.3	74.4	78.0	64.5	88.5	70.9	75.9
D9	80.9	76.0	78.6	75.2	63.0	74.5	81.6	76.5	79.2	75.7	63.7	75.3
D10	87.4	62.1	77.7	79.4	59.2	57.2	88.1	62.5	78.3	79.9	59.8	57.9
D11	91.9	88.7	63.0	56.9	42.6	56.9	92.6	89.1	63.6	57.5	43.3	57.6
D12	92.2	89.3	74.7	85.3	99.2	67.4	92.9	89.7	75.3	85.9	99.8	68.1
D13	73.0	70.9	62.9	70.2	68.3	55.2	73.8	71.3	63.5	70.7	68.9	55.9
D14	81.4	66.7	81.7	75.4	67.6	46.9	82.1	67.1	82.3	75.9	68.2	47.6
D15	72.8	60.3	67.1	69.7	60.3	36.7	73.6	60.8	67.7	70.2	60.9	37.4
D16	88.8	46.9	67.0	64.6	42.8	42.3	89.6	47.3	68.0	66.1	43.5	43.1
AV	**81.2**	**65.7**	**67.7**	**74.5**	**58.7**	**51.9**	**81.9**	**66.1**	**68.4**	**75.1**	**59.3**	**52.3**
SE	**0.02**	**0.05**	**0.04**	**0.03**	**0.05**	**0.03**	**0.02**	**0.05**	**0.04**	**0.03**	**0.04**	**0.03**

Note: DS- Datasets, AV- Average and SE- Standard Error

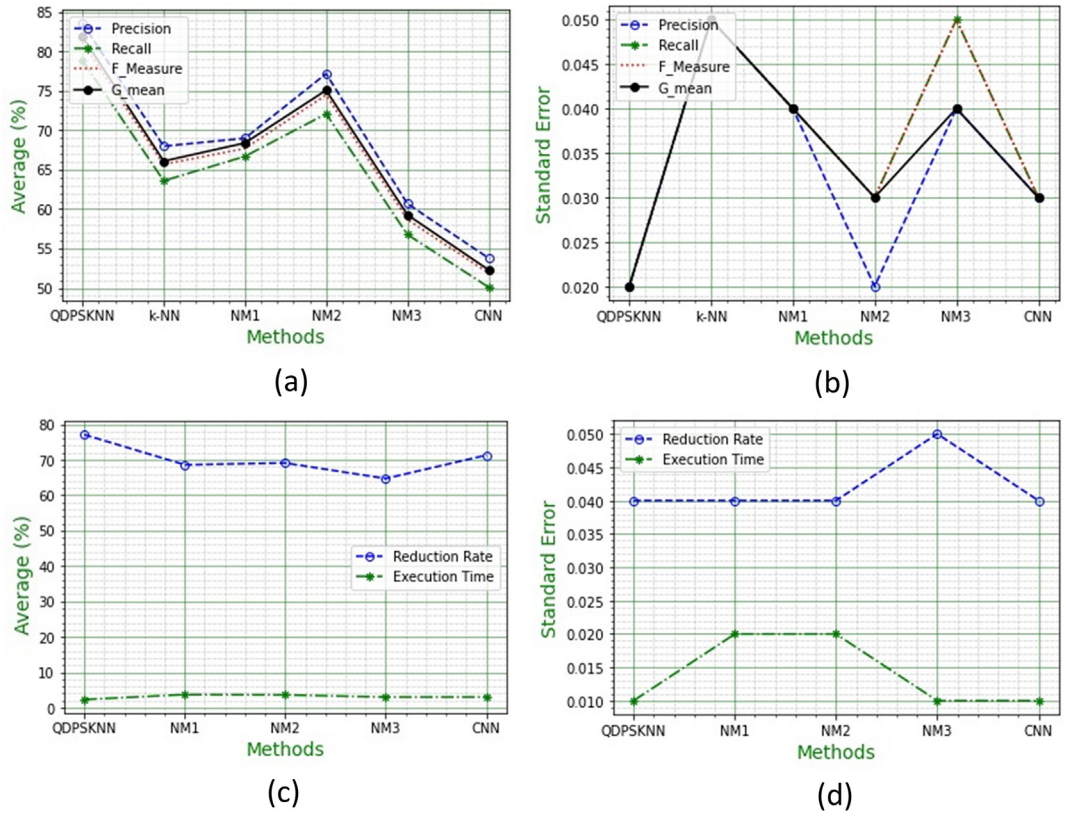

Fig. 4. Summarized performances of proposed and state-of-the-art methods on all sixteen datasets based on average and standard error of (a-b) precision, recall, f-measure and g-mean, (c-d) reduction rate and execution time.

age and standard error of reduction rate and execution time on all sixteen datasets.

6.4. Statistical testing using Wilcoxon Signed Rank test

To evaluate significant differences amid the proposed method QDPSKNN and state-of-the-art methods, Wilcoxon Signed-Rank test [59] was employed in this work. At first, pair-wise comparisons were performed between the methods. The *p*-value was then computed per comparison which demonstrates the lowest-level significance of a hypothesis (h = 0.05) which leads to rejection. Results of Wilcoxon Signed-Rank Test is shown in Table 7 which is computed in terms of precision value illustrated in Table 4. The term h(0.05) discussed in Table 7 represents null-hypothesis at a significance level of 5%, the term R + represents the positive summation of ranks for the data-sets which indicates that the 1st algorithm performs superior than the 2nd. The term R- demonstrates the negative summation of ranks for the data-sets which indicates that the 2nd algorithm performs superior than the 1st. The results obtained through Precision based Wilcoxon test shows the promising performance of QDPSKNN over rest of the methods employed in experimentation.

Table 6
Comparison of Reduction rate (%) and Execution time (seconds) of QDPS with other PS methods.

DS	QDPSKNN		CNN		NM3		NM1		NM2	
	RR	ET	RR	ET	RR	ET	RR	ET	RR	ET
D1	93.9	11.4	80.7	30	90.5	30.7	81.9	21.1	91.9	21
D2	91.9	1.3	78.6	1.3	60.7	1.2	58.1	1.2	58.1	1.3
D3	88.9	2.7	70.8	2.6	74.4	2.2	69.1	2.3	80.3	2.4
D4	74.3	2	68.4	2.8	76.4	2.1	67.4	2.4	76.8	2.3
D5	93.6	2.5	89.2	2.9	90.3	2.6	91.2	2.7	92.1	2.8
D6	79.5	1.2	77.6	1.3	73.1	1.4	78.1	1.5	79.3	1.4
D7	77.9	2.3	71.0	2.4	73.7	2.1	72.9	2.3	72.9	2.2
D8	47.4	1.3	48.1	1.4	43.4	1.3	34.5	1.3	50.5	1.3
D9	55.2	1.1	53.3	1.2	44.3	1.2	19.1	1.2	50.1	1.2
D10	40.2	2.6	32.1	2.9	30.2	2.8	30.2	2.8	38.2	2.8
D11	85.6	1	74.2	1.2	80.3	1	80.1	1.1	73.8	1
D12	85.7	1.3	84.6	1.4	72.5	1.3	87.6	1.5	71.2	1.4
D13	82.4	2.4	69.7	2.8	80.5	2.6	62.3	2.5	79.2	2.1
D14	90.3	1.8	70.2	2.1	89.7	2.4	73.1	2.1	89.3	1.9
D15	75.3	1.2	68.6	1.3	65.6	1.1	68.7	1.2	72.3	1.3
D16	70.7	1.3	61.2	1.2	59.6	1.2	60.3	1.4	65.4	1.3
AV	**77.1**	**2.3**	**68.6**	**3.7**	**69.1**	**3.6**	**64.7**	**3.0**	**71.3**	**3.0**
SE	**0.04**	**0.01**	**0.04**	**0.02**	**0.04**	**0.02**	**0.05**	**0.01**	**0.04**	**0.01**

Note: DS- Datasets, AV- Average and SE- Standard Error, RR = Reduction Rate and ET = Execution Time

Table 7
Results of Wilcoxon Signed rank test based on Precision.

Methods Compared	$R+$	$R-$	p-value	$h(0.05)$
QDPSKNN vs. k-NN	120	16	0.0023364	Rejected
QDPSKNN vs. NM1	341	10	0.0000764	Rejected
QDPSKNN vs. NM2	293	19	0.0183569	Rejected
QDPSKNN vs. NM3	317	45	0.00001	Rejected
QDPSKNN vs. CNN	344	34	0.0000003	Rejected

7. Discussion

For controlled under-sampling, quad division prototype selection-based k-NN is designed to handle the issues of skewness in class distribution towards detecting click fraud in online advertising. The proposed QDPS based nearest neighbor classifier is mainly designed to improve the classification performance of fraudulent publishers from highly skewed user-click dataset.

The procedure of Nearest-Neighbor based methods takes too much time towards searching the prototypes in an entire dataset. Besides, towards storing samples in the training dataset, they store irrelevant samples as well, which degrades the performance. Compared with standard k-NN and other PS methods, QDPSKNN performs more accurately on the imbalanced dataset as it reduces the time need to search for the promising prototypes. By keeping only, the promising nearest-neighbors or prototypes in the training set, it automatically reduces the storage requirements and time needed towards generalization. Not many computations are required by the proposed method to merge instances to generate final balanced reduced-sized dataset before the classification. It lowers the training set size while increasing the classification speed. It also provides a higher reduction rate with lesser execution time corresponding to average learning performance. The present research work performs an empirical study of QDPSKNN with other existing PS methods towards reducing the size of highly imbalanced datasets and was empirically analyzed over small and medium-sized datasets.

8. Conclusion

In this work, a controlled under sampling method QDPSKNN was designed by selecting prototypes using quad-division, which focuses on balancing uneven class distribution on user click data in the classification of fraudsters. Variations were made on the existing k-NN algorithm towards selecting the promising prototypes. At first, the majority and minority data are separated, the proposed method then reduces the data size by performing controlled under-sampling only on majority instances. It divided the majority training data into four equal groups and applied k-NN on each group to select promising prototypes equivalent to the number of minority instances. To test the generalization behavior of QDPSKNN, its performance was also compared with the existing PS methods, namely NearMiss-1, NearMiss-2, NearMiss-3, and CNN, which selected samples to 'keep', instead of reducing samples from the dataset on fifteen benchmark imbalanced datasets. Results showed that rebalance methods that 'keep' the instances shown in Fig. 2(b), 2(c), and 2(d) alters the sampling in the presence of marginal outliers and thus performed poorly than the proposed method. The method shown in Fig. 2(e) is criticized as it elects random samples, which results in the retention of unessential instances and retention of internal samples instead of boundary instances. The designed methodology does not require complex computations towards searching promising nearest-neighbors from each group or finding samples close to decision boundaries. It enhances the classification performance and provides a comparable reduction and precision rate. As shown in Fig. 2(f), promising and relevant instances are chosen from each group for balancing the dataset and overcome the limitation of existing prototypes selection methods. The comparison of the effectiveness of QDPSKNN with the existing PS methods, revealed that it is significantly more appropriate based on precision score, reduction rate, and execution time. The only limitation of the proposed algorithm is that the proposed method is not empirically evaluated over real-time large-sized datasets. The proposed algorithm is only empirically evaluated on small and medium-sized benchmark datasets which are already preprocessed. If experiments are performed with real-time large-sized datasets compris-

ing missing value and outliers, results may vary. The proposed method is unsuitable for datasets comprising, outliers, missing values and noisy samples. The proposed method is inherently dependent on k-NN which is sensitive towards outliers and noise in the dataset that needs manual imputation towards the missing-values and removing the outliers. Thus, evaluating the effectiveness of the proposed method on such datasets may degrade the performance of classification. In the future, one can work towards reducing the issues related to processing huge real-world datasets with QDPSKNN method.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] L. Song, X. Gong, X. He, R. Zhang, A. Zhou, Multi-Stage Malicious Click Detection on Large Scale Web Advertising Data, in: First Int. Work. Big Dyn. Distrib. Data (BD3), Italy, 2013: pp. 67–72.

[2] C.M.R. Haider, A. Iqbal, A.H. Rahman, M.S. Rahman, An ensemble learning based approach for impression fraud detection in mobile advertising, J. Netw. Comput. Appl. 112 (2018) 126–141, https://doi.org/10.1016/j.jnca.2018.02.021.

[3] L. Zhang, Y. Guan, Detecting click fraud in pay-per-click streams of online advertising networks, in: Proc. - 28th Int. Conf. Distrib. Comput. Syst. ICDCS 2008, IEEE Computer Society, 2008: pp. 77–84. https://doi.org/10.1109/ICDCS.2008.98.

[4] H.S. Khraim, A.A. Alkrableih, The Effect of Using Pay Per Click Advertisement on Online Advertisement Effectiveness and Attracting Customers in E-marketing Companies in Jordan, Int. J. Mark. Stud. 7 (2015) 180–189, https://doi.org/10.5539/ijms.v7n1p180.

[5] D. Berrar, Random forests for the detection of click fraud in online mobile advertising, in: Proc. 2012 Int. Work. Fraud Detect. Mob. Advert. (FDMA), Singapore, 2012: pp. 1–10. http://berrar.com/resources/Berrar_FDMA2012.pdf.

[6] V. Dave, S. Guha, Y. Zhang, Measuring and fingerprinting click-spam in ad networks, in: Proc. ACM SIGCOMM 2012 Conf. Appl. Technol. Archit. Protoc. Comput. Commun., 2012: p. 175. https://doi.org/10.1145/2377677.2377715.

[7] H. Xu, D. Liu, A. Koehl, H. Wang, A. Stavrou, Click fraud detection on the advertiser side, in: 19th Eur. Symp. Res. Comput. Secur. Wroclaw, 2014: pp. 419–438.

[8] M. Kantardzic, C. Walgampaya, B. Wenerstrom, O. Lozitskiy, S. Higgins, D. King, Improving Click Fraud Detection by Real Time Data Fusion, in: IEEE Int. Symp. Signal Process. Inf. Technol. Ajman, UAE, IEEE Computer Society, 2008: pp. 69–74. https://doi.org/10.1109/ISSPIT.2008.4775655.

[9] D. Sisodia, D.S. Sisodia, Data Sampling Strategies for Click Fraud Detection Using Imbalanced User Click Data of Online Advertising : An Empirical Review Data Sampling Strategies for Click Fraud Detection Using Imbalanced User Click Data of Online Advertising : An Empirical Review, IETE Tech. Rev. (2021) 1–10, https://doi.org/10.1080/02564602.2021.1915892.

[10] D.S. Sisodia, N.K. Reddy, S. Bhandari, Performance Evaluation of Class Balancing Techniques for Credit Card Fraud Detection, in 2017, IEEE Int. Conf. Power, Control. Signals Instrum. Eng IEEE 2017 (2017) 2747–2752.

[11] D.S. Sisodia, U. Verma, Distinct Multiple Learner-Based Ensemble Smotebagging (ML-ESB) Method For Classification Of Binary Class Imbalance Problems, Int. J. Technol. 10 (4) (2019) 721, https://doi.org/10.14716/ijtech.v10i410.14716/ijtech.v10i4.1743.

[12] D. Sisodia, D.S. Sisodia, Gradient boosting learning for fraudulent publisher detection in online advertising, Data Technol. Appl. 55 (2) (2021) 216–232, https://doi.org/10.1108/DTA-04-2020-0093.

[13] Y. Tang, Y.-Q. Zhang, N.V. Chawla, S. Krasser, Correspondence SVMs Modeling for Highly Imbalanced Classification, IEEE Trans. Syst. Man. Cybern. 39 (2009) 281–288, https://doi.org/10.1109/TSMCB.2008.2002909.

[14] W. Kasemtaweechok, Chatchai and Suwannik, Prototype Selection for k-Nearest Neighbors Classification Using Geometric Median, in: Proc. Fifth Int. Conf. Network, Commun. Comput., 2016: pp. 140–144.

[15] F. Garcia, Salvador and Derrac, Joaquin and Cano, Jose and Herrera, Prototype Selection for Nearest Neighbor Classification : Taxonomy and Empirical Study, IEEE Trans. Pattern Anal. Mach. Intell. 34 (2012) 417–435.

[16] T.R. Wilson, D Randall and Martinez, Reduction Techniques for Instance-Based Learning Algorithms, Mach. Learn. 38 (2000) 257–286.

[17] Y.B. Fernandez Hernandez, R. Bello, Y. Filiberto, M. Frias, L. Coello Blanco, Y. Caballero, An Approach for Prototype Generation based on Similarity Relations for Problems of Classification, Comput. Y Sist. 19 (1) (2015), https://doi.org/10.13053/cys-19-1-2053.

[18] J.J. Valero-Mas, J. Calvo-Zaragoza, J.R. Rico-Juan, Jose J and Calvo-Zaragoza, Jorge and Rico-Juan, On the suitability of Prototype Selection methods for kNN classification with distributed data, Neurocomputing. 203 (2016) 150–160, https://doi.org/10.1016/j.neucom.2016.04.018.

[19] D.B. Skalak, Prototype Selection for Composite Nearest Neighbor Classifiers, Doctoral Dissertation, Department Of Computer Science, University of Massachusetts at Amherst, May 1997, pp. 1-279, n.d.

[20] S. Ougiaroglou, G. Evangelidis, Fast and accurate k-nearest neighbor classification using prototype selection by clustering, in: 2012 16th Panhellenic Conf. Informatics, 2012: pp. 168–173.

[21] R.M.O. Cruz, R. Sabourin, G.D.C. Cavalcanti, Analyzing different prototype selection techniques for dynamic classifier and ensemble selection, in: 2017, Int. Jt. Conf Neural Networks, IEEE 2017 (2017) 3959–3966.

[22] E. Pękalska, R.P.W. Duin, P. Paclík, Prototype selection for dissimilarity-based classifiers, Pattern Recognit. 39 (2) (2006) 189–208, https://doi.org/10.1016/j.patcog.2005.06.012.

[23] J.L.G.-G.R.-G.G. Herrera, Data Reduction for Big Data, in: Big Data Preprocessing, Springer, 2020: pp. 81–99.

[24] J.S. Sánchez, High training set size reduction by space partitioning and prototype abstraction, Pattern Recognit. 37 (7) (2004) 1561–1564.

[25] H.Y. Jianping Gou, Hongxing Ma, Weihua Ou, Shaoning Zeng, Yunbo Rao, A generalized mean distance-based k-nearest neighbor classifier, Expert Syst. Appl. 115 (2019) 356–372. https://doi.org/10.1016/j.eswa.2018.08.021.

[26] J. Gou, W. Qiu, Z. Yi, Y. Xu, Q. Mao, Y. Zhan, Yong Xu, Qirong Mao, A local mean representation-based K-nearest neighbor classifier, ACM Trans. Intell. Syst. Technol. 10 (3) (2019) 1–25, https://doi.org/10.1145/3319532.

[27] J. Gou, W. Qiu, Z. Yi, X. Shen, Y. Zhan, W. Ou, Xiangjun Shen, Yongzhao Zhan, Locality constrained representation-based K-nearest neighbor classification, Knowledge-Based Syst. 167 (2019) 38–52, https://doi.org/10.1016/j.knosys.2019.01.016.

[28] H.L. Yong Xu, Qi Zhu, Zizhu Fan, Minna Qiu, Yan Chen, Coarse to fine K nearest neighbor classifier, Pattern Recognit. Lett. 34 (2013) 980–986. https://doi.org/10.1016/j.patrec.2013.01.028.

[29] P. Hart, The condensed nearest neighbor rule, IEEE Trans. Inf. Theory. 14 (1968) 515–516, https://doi.org/10.1109/TIT.1968.1054155.

[30] C. Kasemtaweechok, W. Suwannik, Adaptive geometric median prototype selection method for k-nearest neighbors classification, Intell. Data Anal. 23 (4) (2019) 855–876, https://doi.org/10.3233/IDA-184190.

[31] V.S. Devi, S.A. Kuruvilla, R. Aparna, Prototype Selection and Dimensionality Reduction on Multi-Label Data, in: Proc. 7th ACM IKDD CoDS 25th COMAD, 2020: pp. 195–199.

[32] Y.-M. Chyi, Classification analysis techniques for skewed class distribution problems, Master Thesis, Department of Information Management, National Sun Yat-Sen University, 2003.

[33] M.A. Arefeen, S.T. Nimi, M.S. Rahman, Neural Network Based Undersampling Techniques, ArXiv Prepr. ArXiv1908.06487. (2019) 1–8. http://arxiv.org/abs/1908.06487.

[34] I. Mani, I. Zhang, kNN approach to unbalanced data distributions: a case study involving information extraction, in: Proc. Work. Learn. from Imbalanced Datasets, 2003: pp. 1–7.

[35] Y. Li, X.J. Zhang, Improving k Nearest Neighbor with Exemplar Generalization for Imbalanced Classification, in: Pacific-Asia Conf. Knowl. Discov. Data Min., 2011: pp. 321–332. https://doi.org/10.1007/978-3-642-20847-8.

[36] S. Liu, Wei and Chawla, Class confidence weighted knn algorithms for imbalanced data sets, in: Pacific-Asia Conf. Knowl. Discov. Data Min., 2011: pp. 345–356.

[37] H. Dubey, V. Pudi, Class Based Weighted K-Nearest Neighbor over Imbalance Dataset, in: Pacific-Asia Conf. Knowl. Discov. Data Min., 2013: pp. 305–316.

[38] S.J. Yen, Y.S. Lee, Under-sampling approaches for improving prediction of the minority class in an imbalanced dataset, Lect. Notes Control Inf. Sci. 344 (2006) 731–740, https://doi.org/10.1007/11816492_89.

[39] W. Feng, W. Huang, J. Ren, Class imbalance ensemble learning based on the margin theory, Appl. Sci. 8 (2018) 1–7, https://doi.org/10.3390/app8050815.

[40] A. More, Survey of resampling techniques for improving classification performance in unbalanced datasets, ArXiv Prepr. ArXiv1608.06048. 10000 (2016) 1–7. http://arxiv.org/abs/1608.06048.

[41] D.G. Altman, J.M. Bland, Statistics Notes: Quartiles, quintiles, centiles, and other quantiles, Bmj. 309 (1994) 996–996. https://doi.org/10.1136/bmj.309.6960.996.

[42] Segmenting Data: Quartiles, Deciles, Percentiles - Data Science Career Options, (n.d.). https://datasciencecareeroptions.com/resources-category/data-science-fundamentals/segmenting-data/ (accessed August 9, 2020).

[43] J. Nicholas, Introduction to Descriptive Statistics, Mathematics Learning Centre, University of Sydney, 1990. https://doi.org/10.1016/b978-0-12-800852-2.00008-0.

[44] C.V.K. Veni, T.S. Rani, Quartiles based UnderSampling(QUS): A Simple and Novel Method to increase the Classification rate of positives in Imbalanced Datasets, in: 9th Int. Conf. Adv. Pattern Recognition, ICAPR 2017, IEEE, 2018: pp. 121–126. https://doi.org/10.1109/ICAPR.2017.8593202.

[45] E. Langford, Quartiles in elementary statistics, J. Stat. Educ. 14 (2006) 1–21, https://doi.org/10.1080/10691898.2006.11910589.

[46] J. Žerovnik, D. Rupnik Poklukar, Elementary methods for computation of quartiles, Teach. Stat. 39 (3) (2017) 88–91, https://doi.org/10.1111/test.v39.310.1111/test.12133.

[47] A. Tharwat, Classification Assessment Methods, Appl. Comput. Informatics. 17 (1) (2021) 168–192, https://doi.org/10.1016/j.aci.2018.08.003.

[48] D.M.W. Powers, Evaluation: From Precision, Recall and F-Measure To Roc, Informedness, Markedness & Correlation, J. Mach. Learn. Technol. 2 (2011) 37–63. https://doi.org/10.1.1.214.9232.

[49] M. Sokolova, G. Lapalme, A systematic analysis of performance measures for classification tasks, Inf. Process. Manag. 45 (4) (2009) 427–437, https://doi.org/10.1016/j.ipm.2009.03.002.
[50] R.P. Indola, N.F.F. Ebecken, On extending F-measure and G-mean metrics to multi-class problems, in: Sixth Int. Conf. Data Mining, Text Min. Their Bus. Appl. UK, 2005: pp. 25–34. https://doi.org/ISSN 1743-3517.
[51] NumPy Reference — NumPy v1.19 Manual, (n.d.). https://numpy.org/doc/stable/reference/ (accessed August 22, 2020).
[52] Documentation — SciPy.org, (n.d.). https://www.scipy.org/docs.html (accessed August 22, 2020).
[53] scikit-learn: machine learning in Python — scikit-learn 0.23.2 documentation, (n.d.). https://scikit-learn.org/stable/ (accessed August 22, 2020).
[54] scikit-learn, (n.d.). https://pypi.org/project/imblearn/ (accessed August 22, 2020).
[55] pandas documentation — pandas 1.1.1 documentation, (n.d.). https://pandas.pydata.org/docs/ (accessed August 22, 2020).
[56] Anaconda package lists — Anaconda documentation, (n.d.). https://docs.anaconda.com/anaconda/packages/pkg-docs/ (accessed August 22, 2020).
[57] R. Oentaryo, E.-P. Lim, M. Finegold, D. and others Lo, Detecting Click Fraud in Online Advertising : A Data Mining Approach, J. Mach. Learn. Res. 15 (2014) 99–140. Doi: 10.1145/2623330.2623718.
[58] A.F. Ernández, J.L. Uengo, J.D. Errac, KEEL Data-Mining Software Tool : Data Set Repository, Integration of Algorithms and Experimental Analysis Framework, J. Mult. Log. Soft Comput. 17 (2011) 255–287.
[59] F. WILCOXON, Individual comparisons of grouped data by ranking methods., J. Econ. Entomol. 39 (1946) 269. https://doi.org/10.1093/jee/39.2.269.