Network Traffic Classification based on Class Weight based K-NN Classifier (CWK-NN)

Mohamad Hijazi School of Arts and Sciences Lebanese International University Nabatieh, Lebanon
MohamadOsamaHijazi@Gmail.com

Jawad Khalife
School of Arts and Sciences
Lebanese International University
Beirut, Lebanon
jkhalife.khalife@liu.edu.lb

Jesus Diaz Verdejo School of IT and Telecom. Eng. University of Granada
Granada, Spain
jedv@ugr.es

Hussein Al-ghor Faculty of Technology Lebanese University Saida,
Lebanon
Hussein.ghor@ul.edu.lb

Abstract— **Network traffic identification is the first and most important step in network management and security. Numerous methods introduced by researchers. One solution depends on processing both packet header and payload, which is costly from both time and processing aspects. Another solution depends on the flow of statistical information such as packet header length, flow duration. Blind classifiers are not accurate yet very fast and do not violate privacy. Machine learning fills the gap between accuracy and time by using the blind classifier method and comparing the results with grand truth then adapting and increasing the accuracy. K-NN is used widely for its effectiveness and simplicity. However, a major drawback of K-NN is its dependency on the training set, being a lazy classification algorithm with no classification model to build. In this work, we aim first at assessing the KNN algorithm in traffic classification. Then we introduce a new deficiency, related to the training samples distribution in the n-dimensional space we measure and propose an enhancement for K-NN adapting to the new problem and outperforming native K-NN classifier. We weight the classes, not the instance, based on the intersections of class clusters in the dataset. Finally, we propose a new Class Weight based K-NN Classifier (CWK-NN), an enhanced K-NN algorithm that can easily adapt to the newly explored training set deficiency.**

Keywords— K-NN, weighted K-NN, traffic classification, computer network, traffic identification, training dataset

I. INTRODUCTION

The ability to identify network applications is centric to many network management and security tasks, including quality of service assignment, traffic engineering, content-dependent pricing, resource allocation, traffic shaping, and others. With the proliferation of applications, many of them using different kinds of obfuscation, traditional port-based classification has long become obsolete.

Numerous methods were proposed for traffic classification as in [1] in the last decade. These methods have different characteristics at many levels, including the analyzed input, the applied techniques, and the classified target objects. Deciding upon which classification features to use is a strategic choice for any traffic classifier. Ideally, Deep packet inspection as in [2] and [3], or DPI, evaluates the data part and the header of a packet that is transmitted through an inspection point. DPI goes beyond examining IP packet headers, therefore, it raises many privacy concerns and is not applicable when the traffic is encrypted or tunneled. However, DPI techniques are considered in the literature as the most accurate techniques and are used therefore as reference classifiers to build the Ground Truth or reference results.

On the other hand, blind classifiers do not inspect the payload and have the potential ability to deal with these obstacles, at the expense of an acceptable sacrifice in accuracy. However, less accurate, the so-called blind methods are preferred in most environments because they guarantee the users' privacy, have the potential to classify encrypted communications and usually require less computational power.

Most of these techniques are based on traffic attributes at the network and transport layers, such as packet sizes and inter-arrival times. Due to the problem dimensionality, Machine Learning (ML) techniques can be used in the classification context. ML classification is considered an instance of supervised learning, i.e., learning where a training set of

correctly identified observations is available. An algorithm that implements classification, especially in a concrete implementation, is known as a classifier. In machine learning, the observations are often known as instances, the explanatory variables are termed features (grouped into a feature vector), and the possible categories to be predicted are classes. The term "classifier" sometimes also refers to the mathematical function, implemented by a classification algorithm that maps input data (traffic attributes in the traffic classification case) to a category (Application type like `HTTP, NTP, DNS…etc.`).

There is a relevant research activity in network traffic classification, employing different Machine learning approaches. Among them, K-NN was used in several papers [4-7] due to its simplicity and zero-training time, as will be shown in the next sections.

Cover and Hart first proposed K-NN (K Nearest Neighbor) algorithm in 1968. The algorithm itself is simple and effective, which is a typical lazy or instance-based learning algorithm. It is not essential for the classifier to use the training set to train and the training time complexity is 0. With K-NN, the function is only approximated locally and all computation is deferred until classification. The principle of K nearest neighbor classification algorithm (K-neighborhood) is that if the K samples in the eigenvalue space most closed to the sample to be identified almost belong to the same class, we determine this sample also belongs to such a class. The key difference between classification and regression tree in [8] is that in classification, the task is predicting a discrete class label while in regression, the task consists of predicting a continuous quantity.

In section two, we included the most significant papers available in the literature. In section three, we explained the known knn problems which is assessing knn in classifying the datasets in each application. In section four we introduced the new problem and we proposed and applied a solution. In section 5 we will describe briefly the Experimental design (for the lack of space). And provide the results of this work. In section 6 there is a conclusion.

II. RELATED WORK

A large number of traffic classification approaches exist in the literature, some are based on deep packet inspection, which discloses the packet payload and raises privacy concerns, while others rely on discriminative traffic attributes at the network and transport layers, such as packet sizes and inter-arrival times. This emergent research needs to arise since IANA Port-based classification in [9] has become obsolete for more than one decade. Machine learning techniques have been widely used in the literature, especially the K Nearest Neighbor due to its simplicity, zero-time for training, and adequacy for general classification contexts.

In [10] authors reduced the flow attributes used in classification by introducing a relationship between flows in order to derive an expanding vector of flow which represents the attributes of the classes. Their algorithm outperforms naïve base classifier with 3.2% of decrease in error rate and achieved 99% of accuracy.

Known DPI methods were enhanced in [11] by introducing a methodology that automatically learns signatures for application. Their method enhances known DPI by identifying new applications that old dpi cannot identify, has higher accuracy, handles variation in applications and adapts to changing network traffic without user intervention.

Many papers in the literature used K-NN for traffic classification and attempted to enhance the native algorithm by addressing key challenges. For example, to select the most appropriate K value for K-NN, authors in [7] used Dynamic K-NN, distance weighted K-NN, clustering the results, and building a model. Dynamic K-NN is responsible for selecting the most appropriate K value for each dataset. In the training phase, K value is selected as well as calculating the weight of each sample, then having clusters in order to build a model. In the test phase, they calculate the distance between the center of each cluster and the test point X to be classified. After finding the nearest cluster, the weighted distance between X and the rest of the cluster samples is calculated. After determining the class of X they add it to the training dataset

after calculating its weight. As a result, they preserved the same accuracy but were able to decrease the time to classify an instance by 50%.

III. K-NN CHALLENGES

A. Algorithm Challenges

1) Adjusting the K value

The K value has a major impact in determining the nature of the instance to be classified, selecting a small value of K means that noise will have a higher influence on the result and a large value make it computationally expensive as explained in [12]. .The only way to validate a K-NN model is by the error on test data. As The authors in [13] applied the manual way of K value Selection. They also applied other solutions related to the dataset modelling. They outperformed old K-NN with performance increase ranging from 0.01 to 0.02 of average misclassification cost.

2) Distance Method Selection

The distance function plays a major role in K-NN classification accuracy. Researches constantly enhance distance functions and add a new attribute to increase accuracy. To determine the proximity of two objects in the feature space, any distance type method can be applied. (Euclidean, Manhattan, Minkowski, etc.) yet researcher must test different distance types for each dataset application . In [14] author introduced a new parameter for the original distance method enhanced K-NN by focusing on enhancing the distance measurement method. In [14] the authors introduced a new variable to the distance rule that made it dynamic. This variable represents the radius of a sphere centered on the new object and excluding all samples that have a distinct label, the distance between an unknown point X (to be classified) and samples that have the same "nature" is less than 1 while the distance between X and the different elements is 1. They tested the new algorithm on five different Datasets and values of K ranging from 1 to 50, with Manhattan, Euclidean distance methods used, they achieved lowering error rates up to 6%.

B. Dataset Challenges

1) Outlier Instances

An outlier is a data point that differs significantly from other observations. An outlier may be due to variability in the measurement or it may indicate the experimental error; the latter are sometimes excluded from the data set. Outliers, when found in the training dataset, can decrease the classification accuracy of K-NN voting.

Finding such exception has received much attention in the data mining field. For example, one of the solutions used in [15] is to partition the dataset into several clusters and then in each cluster, to calculates the Kth nearest neighborhood for the object to find outliers.

Another Solution is weighting each Instance which is quantifying each Instance contribution in the Classification process. And adding that number to the voting process.

Authors in [16] used fuzzy K-NN to enhance accuracy f score and g mean of pulsar selection algorithm. They a special weighting algorithm in the training phase, where they compute the relationship between the features and the class membership. Nearest neighbors are given high membership value to the class they are near from. However, the samples that are far from a class will be given a small membership value to this class. The proposed algorithm outperforms other machine learning algorithms in accuracy metric values starting from 0.01 to 0.04, G means metric from 0.03 to 0.06, and decreased the FPR value by 0.04 .

2) Class Imbalance Problem

A drawback of the basic "majority voting" classification occurs when the class distribution is skewed. That is, examples of a more frequent class tend to dominate the prediction of the new example because they tend to be common among the k nearest neighbors due to their large number.

One way to overcome this problem is to weight the classification, taking into account the distance from the test point to each of its k nearest neighbors. The class

(or value, in regression problems) of each of the k nearest points is multiplied by a weight proportional to the inverse of the distance from that point to the test point.

Another Intersecting application of weighting is Introduced by authors in [4] used the sigmoid function to weight training Instances based on Instances distribution in the dataset.

WKS algorithm boost the weight of instances surrounded by "friendly neighbors" and the decreases the weight if the instance is surrounded mostly with different class members. They were able to decrease the error rate that ranged from 0.007 to 0.025 for 10 different datasets. We find that their work was the most significant to our algorithm so we implemented their algorithm in the experimental setup we built and compared the results between the native, WKS, and CWK-NN.

IV. Proposed Algorithm (Class Weight K-NN)

As mentioned previously, the dataset may have various deficiencies, most commonly, outliers (samples with errors) and, imbalance in the dataset classes (some classes are predominant over the others). To address these deficiencies, weighing samples (outliers problem), or clustered samples (imbalance problem) are among the most common solutions found in the literature.

To explore and understand further KNN training set deficiencies, let us consider the ideal training set for KNN, as derived from the way KNN classifies best any unknown sample.

In a very ideal training-set situation, space is made of one representative sample for each class so that any unknown point will fall in the n-dimensional (n is the number of features or attributes) "exactly" (i.e. coincides) in the location where only one of the class samples exist.

In real training-sets, however, the location of the unknown point and existing sample can rarely coincide. They might rather be close enough for KNN to be able to derive the classification decision through majority voting. When each class has many samples, KNN voting described earlier will make the assumption about the eventual class of the unknown point.

As such, in a less or near-ideal training-set situation, an unknown point may fall then within the zone in the n-dimensional space where only members of the same class surround it, in this case, KNN voting will be more reliable since the unknown point shall undoubtedly belong to the same class of its surrounding neighbors. Following the same reasoning, the closer the samples are to the class centroid (i.e. higher density near the centroid), the more reliable KNN decision shall be.

To illustrate this concept, Figure 1 shows, the distribution of labeled training points. The set of labels belonging to each class (Classes 1, 2 and 3), are clustered into circles for n=2 or spheres for n=3. The point labeled with X is the unknown point. Class cluster radius is defined by the standard deviation of the training samples distances from the centroid, and the distance between two clusters is the distance between their centroids. For example, D12 denotes the distance between centroid C1 of class 1 and C2 of class 2

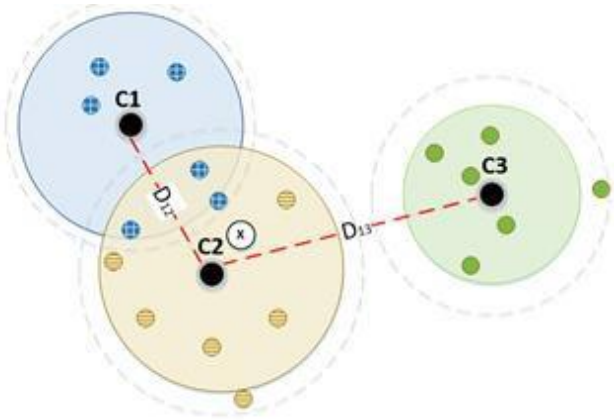

Fig. 1. real training set samples' distribution

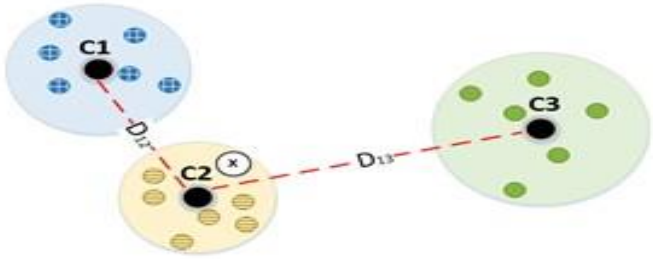

Fig. 2. ideal distribution for a KNN-based classifier

Fig 2. Shows a near-ideal distribution where the clusters are disjoint and do not intersect at all, mimicking the near ideal situation. In this case, a test point falling into only one of the class clusters will lead to a highly deterministic KNN voting decision since all nearest neighbors to X will have the same label, class 2 in this case.

In the case where unknown points fall out of the radius of any of the existing clusters, KNN voting decision still applies however with less reliability. In other words, the points falling in the intersections or between zones are those for which KNN decision will be controversial (i.e. not all of the nearest neighbors have the same class label), which will decrease the overall reliability of KNN classification decisions when compared to the ideal training-set distribution in Fig 2.

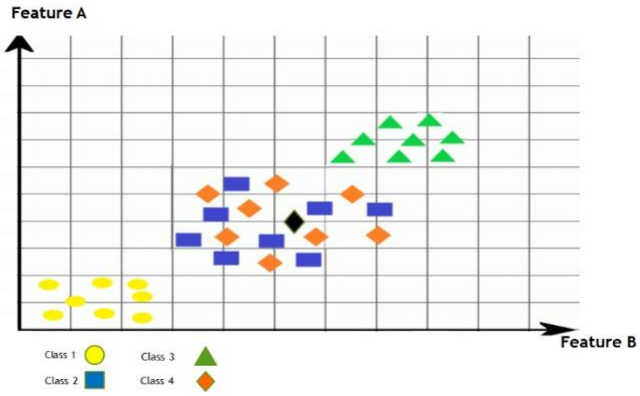

Fig. 3. Real dataset classes distribution

To highlight more on this aspect and for simplicity, in Fig3. we introduce a dataset having only four classes. Each instance has only two attributes. Please notice that the classes are balanced yet two classes is intersecting.

Consider the reliability of the KNN classification decision when classifying the instances that belong to collide classes. The reliability of KNN decisions is very weak. yet for the classes not intersecting the reliability is supposed to increase evidently for training sets having the distribution similar to class one and class 3. Since the reliability of KNN decisions for points, falling inside the clusters will be higher than those falling outside any of the existing clusters or at the intersections. Except for the differences in the neighbors' distances considerations, the reliability of the classification decision shall be quite the same for points falling outside any of the existing clusters or at the intersections in both cases.

Therefore, when the circles representing class clusters are more distant and dense as in Fig2, KNN classification is supposed to become more reliable, at least for the unknown points falling within class clusters. On the contrary, having too close cluster centroids with lower density (i.e. more spread distribution) will result in additional intersections between clusters, which will decrease the reliability of KNN classification for the unknown points falling on the intersecting zones between classes (e.g. C2 and C3 in Fig 1.).

In other words, when class clusters have no or fewer intersections, the training dataset shall become closer to the "near-ideal distribution" in Fig 2. . Thus leading for better classification results.

Despite the fact that imbalanced sets are likely to have class clusters intersections, having one characteristic (i.e. class imbalance or class intersections) does not necessarily imply the other:

- A dataset can be imbalanced but classes do not intersect
- A dataset can be balanced but classes do intersect

The latter case, depicted in Fig 1. is the most relevant to our approach where we measure the new KNN training set deficiency related to relative points' distribution within the n-dimensional space rather than

their relative numbers. To the best of our knowledge, "Class clusters intersections" is not explored yet as a deficiency for KNN training sets, nor was it regarded in separate from the imbalanced class problem.

Up to this level, we have theoretically explained and discussed what we think to be ideal for KNN training set distribution and speculated about its impact on KNN classification. However, most of the presented assumptions have to be proven through experimentation.

For this reasons, we attempt next to measure and quantify the value of "class clusters intersections" inside KNN training sets. Then, we attempt to derive a new KNN classifier in order to experimentally demonstrate the importance and impact of considering of class intersections or training points distribution in the training set on KNN classification.

A. Weighting Classes with CWKNN

A Class Weight based KNN Classifier or CWKNN is simply a KNN classifier that takes into consideration the samples distribution or class intersections described earlier. As mentioned, intersections should be expressed in terms of newly defined class weights. The WCKNN algorithm is a two-phase process that includes the training and the testing described as follows:

1- Calculate centroids of each class.
2- Calculate the variance instance of each class.
3- Calculate the distance between centroids.
   if the distance is larger than the sum of two radiuses the classes does not intersect and the weight of the two classes remain the same. Else if the distance between centroids is smaller than the sum of radiuses there is an intersection and the weights of the classes must be decreased.
4- Assign the weight calculated for all instances.

Training phase:

For each class we add two instances the first instance is

The centroid, and second instance represents the avarge distribution of the class in the n dimensions (where n is the number of attributes).

The centroid is the instance core instance having the average value of all attributes. The radius is the distance between centroid and the instance having the variance of value for all attributes. Thus it's the allowed normal margin for class distribution in the n dimensional space. All the instances outside the threshold is considered as outliers and they are not significant comparing to the majority of the class instances. Distances between centroids are accordingly calculated. To formalize how CWKNN operates in the training phase refers to simply formalize how class weights are calculated. The main idea of setting the class weights is that classes who do not intersect have less error margin. We boost the weight of all of Instances having the same class. However, classes with intersections should have decreased weights.

Let M be the total number of classes in the dataset

For each class i, find the distance between the class centroid and all other centroids.

Equation 1

$$W = 1 + e^{-i}$$

Equation 2

$$= \sum_{=1}^{M} a$$

$$a_{ij} = \begin{cases} 0, & \text{if distance}(i,j) \leq (\mathbf{r_i} + \mathbf{r_j}),\ \ j \neq 0 \\ \frac{1}{M} & \text{if distance}(i,j)\ > (\mathbf{r_i} + \mathbf{r_j}),\ \ j \neq 0 \\ \frac{1}{M} & \text{if distance}(i,j)\ = 0,\ \ j = i \end{cases}$$

$$\epsilon [0 , \frac{1}{M}]$$

$$A \quad \epsilon [\frac{1}{M} , 1]$$

$$W \epsilon [1, \quad (1 + e^{-1})]$$

Wi: Weight of class i in what concerns class intersections.

aij is a discrete value indicating if there is an intersection between classes i and j.

Ai is a value measuring the level of intersections between class i and the rest of classes.

As shown in equation 1, if Ai, denoting the number of class i's intersections with other classes, increases then Wi should decrease.

Ai has a minimal value of $\frac{1}{M}$ (since each class intersects with itself at least)--when the class is disjoint from all other classes-- and a maximum value for Ai=1 when theoretically it intersects with all other classes.

This mechanism ensures that the interval of Wi is. Intersecting classes are not thus explicitly punished, but rather, they are not rewarded higher weights.

Test phase: each training sample belonging to class i is assigned the weight of the class. The weighted voting of K neighbors is combined, and the weight of each neighbor is the function of the class to which the neighbor

the training phase.

As per equation 1, the class weight decreases exponentially when the value of the number of intersections increases towards M. The contributions of the classification for the training sample belonging to a good class is regulated thus by the weight value when the numbers of intersections get considerably higher.

V. DATASET AND RESULTS

One of the objectives of this work, as previously stated, is to assess and to enhance KNN in network traffic classification. The completion of these objectives requires the accomplishment of some additional objectives related to building the experimental setup.

In this experiment we used a server that has 16 GB of ram , 2 Intel(R) Xeon(R) 2.66 GHz processors with 4 cores each. With Ubuntu 18.01 OS to capture network traffic and converting the flows into a dataset. We captured network flows by using tcp dump. then we used nDPI to detect the real nature of the flow traffic and to set the flow attributes.

was collected the flow , over an extended period of time spanning around 3 days and totaling around 1.2 GB of real traffic. we have excluded the encrypted traffic.

The 3 days span of collecting the data was due to our desire to collect, a large dataset for testing and classification and then we were obliged to use a part of the data collected 51752 labeled flow for the lack of resources. The part was selected randomly. Knn classification algorithm stores all training dataset in the in the RAM.

After obtaining the dataset we have used weka open source software to add wks and our algorithm.

To validate the obtained Result we have used many other datasets from the weka library. We will only include the results from three dataset for the lack of time.

Dataset	Number of clases	Number of attributes	Number of instances
Network flow	7	59	51752
Indian diabetes	2	8	768
Balance scale	3	4	628

Fig. 4. Datasets general information.

As mentioned, there is no optimal conditions applicable for all dataset so we have ran several experiments to adjust the optimal k value and distance method per dataset. Then we applied wks and CWK-NN enhancement.

The dataset was randomized we used 0 as a random seed. Then we split the dataset by 2/3 ratio for training and the remaining for testing.

Dataset	**True positive rate**		
	K-NN	***WKS***	***CWK-NN***
Network flow	98.548%	98.367%	98.721%
Indian diabetes	61.67%	64.05%	69.02%
Balance scale	77.74%	78.18%	78.93%

Fig. 5. comparison between the native knn, wks, and the proposed algorithm in the best k value and distance method per dataset.

VI. CONCLUSION

In this work, we have explored and measured a new deficiency, related to KNN training samples distribution in the n-dimensional space and classes' intersections. We weighted classes accordingly in the training set. Then, we suggested class weighting scheme, then we derived a newly proposed Class Weight based KNN Classifier (CWKNN), an enhanced KNN algorithm that can account for sample distributions or classes intersections in KNN training sets.

Furthermore, we have assessed KNN algorithm in traffic classification contexts through an experimental test-bed that we build for this purpose.

Our tests on real captured and Internet-based datasets showed up top 7% enhancements in the classification results, compared to the native and samples' weighted KNN approaches.

Moreover, and as per our testbed, we showed that K=3 and Euclidean distance type can lead to more than 98% of overall Tp in computer traffic classification context.

VII. REFRENCES

[1] J. Khalife J. Verdejo, and A. Hajjar A multilevel taxonomy and requirements for an optimal traffic-classification model [Journal]. - [s.l.] : Int. Journal of Network Management , 2014. - 2 : Vol. 24.

[2] J. Khalife J. Verdejo, and A. Hajjar On the Performance of OpenDPI in Identifying P2P Truncated traffic [Book]. - [s.l.] : JOURNAL OF NETWORKS, 2013. - Vols. VOL. 8, NO. 1.

[3] digitalguardian digitalguardian [Online] // digitalguardian. - digitalguardian, 2019. - https: //digitalguardian.com/blog/what-deep-packet-inspection-how-it-works-use-cases-dpi-and-more.

[4] Min Zhang Min Qi , *, Kang Sun , YuJun Niu , Longxiang Shi New Classification Algorithm WKS Based on Weight [Conference] // 2017 IEEE 19th International Conference on e-Health Networking, Applications and Services (Healthcom). - Dalian, China : IEEE, 2017.

[5] Hou Kaihu Zhan, Hui ,Zhou Zhou A Weighted KNN Algorithm Based on Entropy [Journal]. - Kunming, china : IEEE, 2018.

[6] Shweta Taneja Charu Gupta, Kratika Goyal , Dharna Gureja Enhanced K-Nearest Neighbor Algorithm Using Information Gain and Clustering [Conference] // 2014 Fourth International Conference on Advanced Computing & Communication Technologies. - [s.l.] : IEEE, 2014.

[7] Kumar Nath Syed,M.Rahman,Akram Salah An Enhancement of k-Nearest Neighbor Classification Using Genetic Algorithm [Journal]. - 2018.

[8] machinelearningmastery machinelearningmastery [Online] // machinelearningmastery. - machinelearningmastery, 2018. - https: //machinelearningmastery.com/classification-versus-regression-in-machine-learning/.

[9] IANA IANA Port Numbers. [Online] // IANA . - IANA, 2013. - 01 2013. - http: //www.iana.org/assignments/ port-numbers.

[10] Lei Dinga Jun Liua, Tao Qina, Haifei Lib Internet Traffic Classification Based on Expanding Vector of Flow [Journal]. - China : IEEE.

[11] Alok Tongaonkarî Ruben Torres, Marios Iliofotou, Ram Keralapura, Antonio Nucci Towards self adaptive network traffic classification [Journal]. - [s.l.] : IEEE, 2014.

[12] pseudo_teetotaler stack overflow [Online] // stack overflow. - 2019. - https: //stackoverflow.com/questions/33884325/ideal-k-value-in-KNN-for-classification.

[13] Zhang Shichao Cost-Sensitive KNN Classification [Journal]. - [s.l.] : IEEE, 2019.

[14] Jigang Wang Predrag Neskovic,Leon N. Cooper Improving nearest neighbor rule with a simple adaptive Distance measure [Conference] // International Conference on Natural Computation. - [s.l.] : ICNC 2006: Advances in Natural Computation, 2006.

[15] P. Yang B. Huang KNN Based Outlier Detection Algorithm in Large Dataset [Conference] // 2008 International Workshop on Education Technology and Training & 2008 International Workshop on Geoscience and Remote Sensing. - Shanghai : Research Gate, 2008.

[16] TM Mohamed Pulsar Selection Using Fuzzy knn Classifier [Conference] // Future Computing and Informatics Journal (2018). - [s.l.] : IEEE, 2018.