Parameter-Free Conglomerate nearest Neighbor Classifier Using Mass-Ratio-Variance Outlier Factors

Patcharasiri Fuangfoo and Krung Sinapiromsaran*

***Abstract*—Classification is one important area in machine learning that labels the class of an instance via a classifier from known-class historical data. One of the popular classifiers is k-NN, which stands for "k-nearest neighbor" and requires a global parameter k to proceed. This global parameter may not be suitable for all instances. Naturally, each instance may situate on different regions of clusters such as an interior instance placed inside a cluster, a border instance placed on the outskirts, an outer instance placed faraway from any cluster, which requires a different number of neighbors. To automatically assign a different number of neighbors to each instance, the concept of scoring from the anomaly detection research is desired. The Mass-ratio-variance Outlier Factor, MOF, is selected as the scoring scheme for the number of neighbors of each instance. MOF gives the highest score to an instance placed very far from any cluster and the lowest score to an instance surrounded by other instances. This leads to the proposed classifier called the conglomerate nearest neighbor classifier, which does not require any parameter assigning the appropriate number of neighbors to each instance ordered by MOF. Experimental results show that this classifier exhibits similar accuracy to the k-nearest neighbor algorithm with the best k over the synthesized datasets. Six UCI datasets, the QSAR dataset, the German dataset, the Cancer dataset, the Wholesale dataset, the Haberman dataset, and the Glass3 dataset are used in the experiment. This method outperforms two UCI datasets, Wholesale and Glass3, and displays similar performance with respect to these six UCI datasets.**

***Index Terms*—Classification, conglomerate nearest neighbor, *k*-nearest neighbor, and mass-ratio-variance**

I. Introduction

The main task of classification in machine learning is to identify a class for an unknown-class instance via a classifier. The classifier is built from historical label-class data, called a training dataset, using the classification algorithm. The most attractive classifier is the k-nearest neighbor (k-NN), which identifies an unknown-class instance by the majority voting classes from k- nearest neighbors of that instance in a training dataset. The prediction algorithm requires the number of nearest neighbors to extract their classes for comparison, where neighbors are defined by some similarity measure [1].

Even though this k-nearest neighbor algorithm is simple and only requires a single global parameter k, it has two weaknesses [2]

1) It requires the whole collection of instances from a training dataset to search for k-nearest neighbors, and
2) The global parameter k may not be suitable for all instances in different regions such as interior instances, border instances, and outer instances since they all exhibit different density.

Many studies have attempted to solve the first problem using a structured search tree. Their strategies are to gather similar instances in a partition and find the representative instance in that partition to compare with a new instance. Junior Medjeu Fopa et al. propose the parameter-free KNN method for rating prediction called *free*KNN. It dynamically selected an appropriate number of neighbors depending on the user and the item to be rated [3]. Some researchers propose an improved k-nearest neighbor algorithm denoted as Dk-NN, using dynamic k instead of a single value of k [4], and some researchers varying k from 1, 3, 5, …, $\sqrt{n}$ and use the majority voting of all classes to label an instance [2].

Applying the k -nearest neighbor algorithm using a single fixed k value may not be appropriate for interior instances, border instances, and outer instances. Therefore, selecting the k value according to the density of each instance would give better results. To identify these types of instances, a scoring algorithm is desired.

An anomalous score is designed to give high scores to abnormal instances, or outer instances. These anomalies appear in the dataset and do not conform to any well-defined notion of normal instance behavior [5]. An interior instance that densely appears in a cluster is categorized as a normal instance, whereas an instance that is isolated and distant from other instances is usually categorized as an anomalous instance. Note also that a border instance should be assigned score higher than a normal instance since it places at the outskirts of the cluster.

A mass-ratio-variance outlier factor (MOF) [6] uses the density concept by utilizing the mass-ratio between a pair of data points, which is defined as the variance of the mass-ratio distribution from the considered instance against other instances. The large variance is associated with outliers, whereas the small variance is associated with a normal data point.

This research automatically assigns different odd-values from 1 for anomalous instances to $\sqrt{n}$ or normal instances placing in the center of a cluster according to their density scores from MOF without any user parameter setting while the performance of this algorithm has little effect on the accuracy. It can be used in streaming data, where the appropriate number of nearest neighbors at one window, a collection of time-ordered instances during some interval of times, may not be appropriate for the next window, and searching for the appropriate number of nearest neighbors at any window is prohibitive. Only parameter-free classification algorithm is suitable in this situation.

The rest of this paper is organized as follows. The next section briefly describes related work. Section III details the

Manuscript received January 3, 2023; revised January 19, 2023; accepted March 12, 2023.

The authors are with Chulalongkorn University, Bangkok, Thailand. E-mail: 6470121923@student.chula.ac.th (P.F.)

*Correspondence: krung.s@chula.ac.th (K.S.)

conglomerate nearest neighbor classifier and its algorithm. Section IV is the experimental results, and the last section concludes this paper and describes the future work.

II. Related Work

This section states all materials used in this research. The k-nearest neighbor algorithm is reviewed included its weakness. The improved k-NN algorithms from other researchers are also surveyed. Moreover, the concept of outlier scores is covered, which is a mass-ratio-variance outlier factor (MOF) used to assign different numbers of nearest neighbors for instances in a dataset.

A. k-Nearest Neighbor

The k-Nearest Neighbor algorithm [7, 8] is based on the concept that instances from the same class will form a dense cluster. So, it is possible to predict a class label for an unclassified instance by considering the class of instance close to it. k-NN finds the k closest instances to the query instance and determines its class by locating the class label that appears most frequent. In a machine learning term, k-NN algorithm is lazy since there is no training phase only the testing phase is performed, and all training instances are kept for the testing phase. Several improved k-NN algorithms employ weighting schemes that change the voting influence and distance measurements of a dataset to produce more accurate results. The effectiveness of k-NN has been questioned due to its large storage requirements, and lack of a principled method for choosing k.

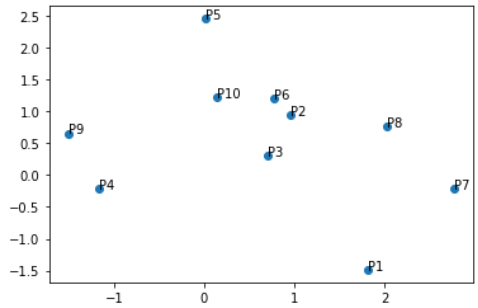
The effectiveness of the k-NN algorithm is influenced by the choice of k. Two problems listed below are the disadvantages of the k-NN algorithm.

- If a noise is present for 1NN in the area where the query instance is located, the class of the query instance can be defined in terms of this noise. This could be resolved with $k > 1$.
- When a class, or a portion of a class, is defined by a subset having the number of elements smaller than k. Most border instances will be misclassified. This problem could be resolved by lowering k.

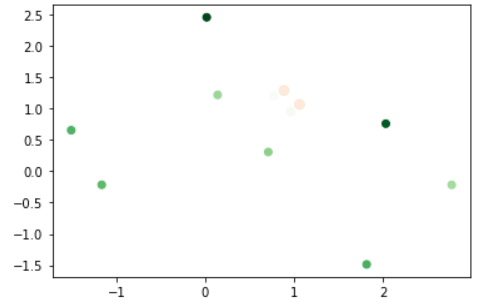
In 2014, Ahmad Basheer Hassanat et al. [2] proposed multi-classifiers with ensemble learning using the same nearest neighbor rule. This classifier will be indicated by MkNN in this paper. MkNN builds multiple k-NN classifiers starting from 1 to the integral square root of the number of instances in the training set. Then it labels a class of an instance using majority rule from these k-NN classifiers, i.e., the class with the highest number of votes from 1-NN, 3-NN, 5-NN, …, $\sqrt{n}$-NN will be chosen. The result shows that their classifier outperforms the traditional k-NN using a single value of k. The reason for choosing the largest number of neighbors as the square root of the training set comes from their experiments and if k is large then the algorithm requires an extensive computation time.

B. Outlier Score

In 2021, a parameter-free mass-ratio-variance outlier factor (MOF) [6] is an outlier score that is the variance of the mass-ratio distribution of the computed instance. Specifically, the density of an instance is first calculated, and then compared with that of its neighboring instances. This comparison yields an outlier score that indicates the degree to which the instance deviates from the norm in terms of its density. In this method, normal instances and their neighbors have similar densities, whereas outliers have densities that differ significantly from those of their neighbors. By evaluating the density of an instance related to that of its neighbors, this approach provides an effective means of detecting outliers in a large dataset. The mass-ratio of other instances is defined as the ratio of the number of instances within the sphere of the distance from this computed instance to that of other instances.



(a) sample dataset



(b) MOF scoring

Fig. 1. Example of a mass-ratio-variance outlier factor (MOF).

To understand MOF, consider the original dataset in Fig. 1. (a) and the colored dataset using MOFs in Fig. 1. (b), where the shaded color corresponds to values of MOFs. Instances P2 and P6 in Fig. 1. (b) have the lightest shade since they are placed inside the cluster. Instances P4 and P9 have the darker shade since they are both at the border of the cluster. Instance P5 has the darkest shade since it is the farthest away from any cluster.

III. Conglomerate Nearest Neighbor Classifier

The proposed conglomerate nearest neighbor classifier requires two phases of learning, (1) the training phase for assigning the number of nearest neighbors to all instances and (2) the testing phase for determining the class of an instance.

A. Proposed Method

The pseudo-code for two-phases conglomerate nearest neighbor algorithm is demonstrated in subsection B of this section. During the training phase, the conglomerate nearest neighbor algorithm determines the maximum number of nearest neighbors of this dataset by finding the odd integer less than or equal to the square root of the number of instances in each class, called K. MOF is calculated for each class and partition according to the odd integer from 1 to K

with the number of nearest neighbors assigned to each instance. Note that if an instance is placed in the interior, surrounded by other instances, the number of assigned neighbors is high in order to increase the accuracy of predicting this instance and if it rests on the border, then it is wise to use a small number of neighbors to reduce misclassification and if it places far away from any cluster, then it is wise to use the least number of neighbors, i.e., 1.

To determine the number of neighbors based on MOF, the range of MOFs for each class obtained from the training dataset, from the smallest to the largest values, is divided evenly into the greatest integer less than or equal to square root of the number of given class instances (k_c), i.e., k = 1 is assigned to the highest MOF range, k = 3 is assigned to the next to the highest MOF range and so on until the last lowest MOF range uses $k = k_c$.

To predict a class of unknown instance, x, in the testing phase, the conglomerate nearest neighbor algorithm first finds the closet instance, x_c,. Then it extracts MOF of x_c and uses it to determine the number of used neighbors, k_x. Lastly, the class of x is determined by the majority class among k_x nearest neighbors of x.

B. Pseudo-Code of the Conglomerate Nearest Neighbor Algorithm

The following is pseudo-code of conglomerate nearest neighbor algorithm.

Input: the training dataset, and unknown instance x
Output: class label of unknown instance

1) For each class, calculate the maximum of number of neighbor (k_c) = the greatest integer less than or equal to square root of the number of training dataset.
2) For each class, calculate the MOF score for every instance in the training dataset.
3) Divide equally the MOF score range into k_c ranges and k = 1 is assigned to the highest MOF range, k = 3 is assigned to the next to the highest MOF range and so on until the last lowest MOF range uses $k = k_c$.
4) Find the nearest instances of the training dataset according to a distance metric.
5) Use the number of neighbors according to that nearest neighbor.
6) Resulting Class = most frequent class label of the k nearest instances.

IV. Experimental Results

The conglomerate nearest neighbor algorithm is implemented on the google colaboratory using Python programming language. The accuracy from the synthesized datasets and UCI datasets will be used to compare the performance of each algorithm, which are the best k of the k-NN algorithm and the MkNN algorithm.

A. Synthesized Dataset

These synthesized datasets are generated using make_circles from scikit-learn to make two circles, the large circle represents one class, and the embedded small circle represents another class. The generated instances of each class have the same standard deviation and scaling factor. Table I summarizes the properties of each dataset with the number of instances (#Inst), the number of class 0 (#c0) in blue and the number of class 1 (#c1) in red. The synthesized data is presented graphically, with Fig. 2 depicting the first dataset having the number of instances in class c0 having 5 times the number of instances in class c1. Fig. 3 displaying the second dataset having the number of instances in class c0 having 2.5 times the number of instances in class c1. Fig. 4 exhibiting the third dataset having the number of instances in class c0 having approximately 1.67 times the number of instances in class c1. Fig. 5 illustrating the fourth dataset having the number of instances in class c0 having 1.25 times the number of instances in class c1 and Fig. 6 showcasing the fifth dataset having the same number of instances for both classes.

TABLE I: Synthesized Datasets Used

No.	#Inst	#c0	#c1
1	600	500	100
2	700	500	200
3	800	500	300
4	900	500	400
5	1000	500	500

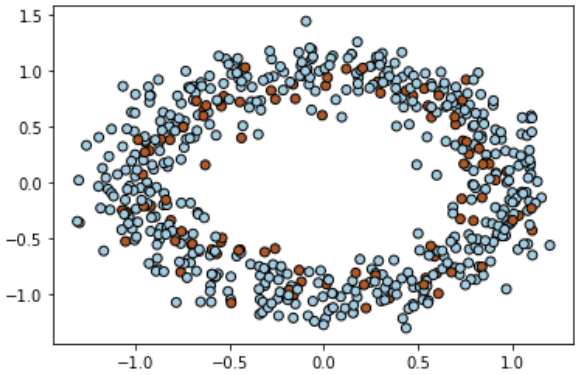


Fig. 2. Plot of synthesized data for dataset no. 1.

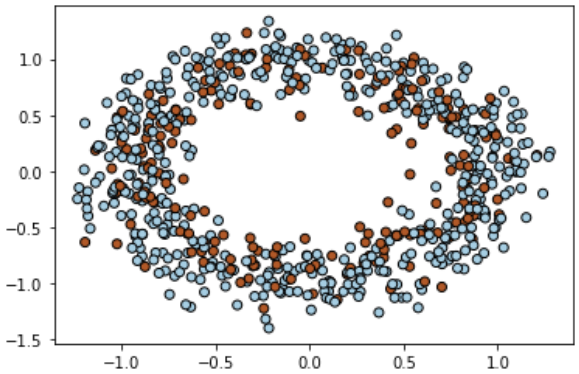


Fig. 3. Plot of synthesized data for dataset no. 2.

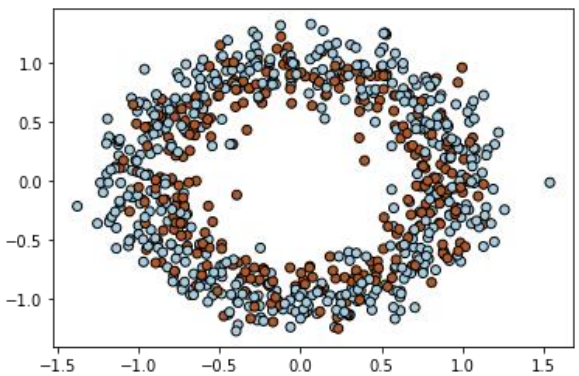


Fig. 4. Plot of synthesized data for dataset no. 3.

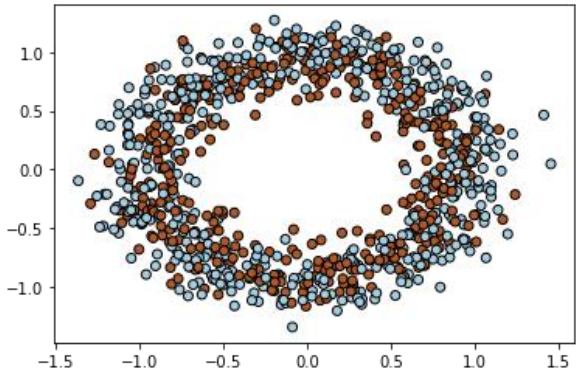


Fig. 5. Plot of synthesized data for dataset no. 4.

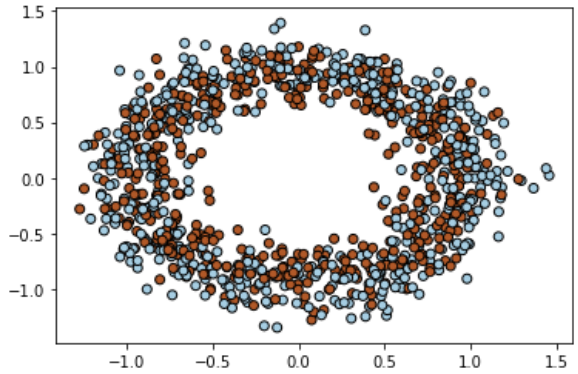


Fig. 6. Plot of synthesized data for dataset no. 5.

Table II shows the accuracy of each dataset, and Fig. 7 shows the bar graph representing performance of k-NN, MkNN, and Conglomerate NN. The proposed algorithm exhibits better performance on datasets no.1 and no.2 than k-NN and dataset no.4 shows that it has a better accuracy than MkNN.

Considering dataset no. 1 and no. 2 when the number of instances from one class is at least two time higher than the number of instances from another class, the Conglomerate NN exhibits better performance than that of k-NN since the Conglomerate NN uses different number of neighbors based on its density whereas k-NN uses only one value of neighbors for all instances. There is not much difference between the performance of MkNN and Conglomerate NN regardless of the number of instances.

TABLE II: THE RESULTS OF THE PROPOSED CLASSIFIER COMPARED TO OTHER CLASSIFIERS-ACCURACY IS THE AVERAGE OF 50 RUNS

No.	k-NN	MkNN	Conglomerate NN
1	0.7912	0.8292	0.8256
2	0.6727	0.7032	0.7023
3	0.6615	0.6501	0.6459
4	0.6080	0.5830	0.5845
5	0.5995	0.5746	0.5695

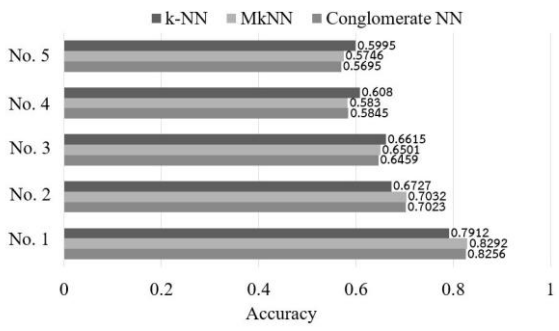


Fig. 7. Average accuracy for synthesized datasets from 50 trials.

B. UCI Dataset

This part covers the results from six real-world datasets from UCI repository, which are the QSAR dataset, the German dataset, the Cancer dataset, the Wholesale dataset, the Haberman dataset and the Glass3 dataset, as shown in Table III. Each dataset contains only two classes. The "Name" column is the dataset name, the "#Inst" column shows the number of instances and the "#Att" column shows the number of attributes.

The results in Table IV show the accuracy for each classifier and are represented in the bar graph in Fig. 8. The proposed method was compared with two other algorithms, the best k of the k-NN algorithm and the MkNN algorithm. The proposed algorithm outperforms both k-NN and MkNN on the Wholesale dataset and the Glass3 dataset. It has a better accuracy than k-NN on the German dataset and it is better than MkNN on the Haberman dataset. There is not much difference between the performance of MkNN and Conglomerate NN.

TABLE III: DESCRIPTION OF THE DATA SETS USED

No.	Name	#Inst	#Att
1	QSAR	1055	41
2	German	1000	24
3	Cancer	699	10
4	Wholesale	440	6
5	Haberman	306	3
6	Glass3	214	9

TABLE IV: THE RESULTS OF THE PROPOSED CLASSIFIER COMPARED TO OTHER CLASSIFIERS-ACCURACY IS THE AVERAGE OF 50 RUNS

Name	k-NN	MkNN	Conglomerate nearest neighbor
QSAR	0.8252	0.8005	0.7679
German	0.6994	0.7064	0.6998
Cancer	0.9746	0.9649	0.9639
Wholesale	0.6645	0.6818	0.7002
Haberman	0.7592	0.7416	0.7444
Glass3	0.9207	0.9200	0.9211

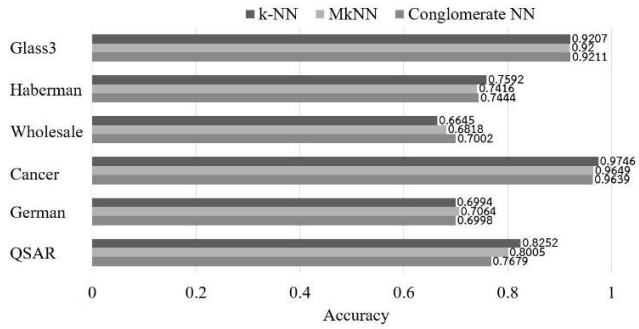


Fig. 8. Average accuracy for UCI dataset from 50 trials.

V. CONCLUSION

This paper proposes the conglomerate nearest neighbor algorithm with no parameter. Different nearest neighbor assignments for each instance come from MOF, which is adapted from density of instances in a dataset. From the experiments with the synthesized datasets, it has a similar performance to the original k-NN with the best k and MkNN.

For synthesized datasets as two class circles, three algorithms, the k-NN algorithm, the MkNN algorithm, and the conglomerate nearest neighbor algorithm, could identify the class of an unknown instance in the testing set with similar accuracy except the conglomerate NN has better

performance than k-NN when the data is imbalance. For real-world datasets, the conglomerate nearest neighbor can predict the class of an unknown instance inclusive of k-NN. Moreover, the conglomerate nearest neighbor has higher accuracy than k-NN in the German dataset, the Wholesale dataset, and the Glass3 dataset.

Without any parameter, the conglomerate nearest neighbor demonstrates comparable performance with k-NN, whereas k-NN will need to determine the optimal parameter to achieve the best result. Moreover, the conglomerate has similar performance to MkNN.

As part of future work, the current work is currently adding steps to determine the optimal number of neighbors in the training phase and selecting the number of neighbors for the new instance via its nearest neighbor. Nonetheless, the conglomerate nearest neighbor can be improved during the testing phase such as selecting the number of neighbors from the largest value among the number of nearest neighbors from each class. This conglomerate nearest neighbor could be extended to solve a multi-class classification problem with some categorical attributes.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Patcharasiri Fuangfoo conducted the research, analyzed, and wrote the paper; Krung Sinapiromsaran originated the concept, analyzed, and revised the paper; All authors had approved the final version of the paper.

Acknowledgment

Patcharasiri Fuangfoo thanks the Applied mathematics and computational science, department of mathematics and computer science, Faculty of Science, Chulalongkorn University for publication support.

References

[1] A. A. Torres-Garc á, C. A. Reyes-Garc á, L. Villase ñor-Pineda, and O. M. Montoya, "Biosignal processing and classification using computational learning and intelligence principles," *Algorithms, and Applications*, London U.K.: Academic Press, 2022, ch. 6, p. 117.

[2] A. B. Hassanat, M. A. Abbadi, G. A. Altarawneh, and A. A. Alhasanat, "Solving the problem of the K parameter in the KNN classifier using an ensemble learning approach," *(IJCSIS) International Journal of Computer Science and Information Security*, vol. 12, no. 8, pp. 33–39, Aug. 2014.

[3] J. M. Fopa, M. Gueye, S. Ndiaye, and H. Naacke, "A parameter-free KNN for rating prediction," *Data & Knowledge Engineering*, vol. 142, Nov. 2022.

[4] X. F. Zhong, S. Z. Guo, L. Gao, H. Shan, and J. H. Zheng, "An improved k-NN classification with Dynamic k," *International Conference on Machine Learning and Computing (ICMLC)*, pp. 211–216, Feb. 2017.

[5] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Computing Surveys*, vol. 41, no. 3, pp. 15–15:58, July 2009.

[6] P. Changsakul, S. Boonsiri, and K. Sinapiromsaran, "Mass-ratio-variance based Outlier Factor," presented at 18th International Joint Conference on Computer Science and Software Engineering (JCSSE), 2021.

[7] S. B. Kotsiantis, I. D. Zaharakis and P. E. Pintelas, "Machine learning: a review of classification and combining techniques," *Artificial Intelligence Review*, vol. 26, pp. 159–190, 2006.

[8] S. B. Kotsiantis, "Supervised machine learning: A review of classification techniques," *Informatica*, vol. 31, no. 3, pp. 249–268, 2007.