# Validation Based Modified K-Nearest Neighbor

Hamid Parvin, Hosein Alizadeh and Behrouz Minaei-Bidgoli

*Department of Computer Engineering, Iran University of Science and Technology, P.O. Box 16765-163, Narmak, Tehran, Iran*

**Abstract.** In this paper, a new classification method for enhancing the performance of K-Nearest Neighbor is proposed which uses robust neighbors in training data. The robust neighbors are detected using a validation process. This method is more robust than traditional equivalent methods. This new classification method is called Modified K-Nearest Neighbor. Inspired the traditional KNN algorithm, the main idea is classifying the test samples according to their neighbor tags. This method is a kind of weighted KNN so that these weights are determined using a different procedure. The procedure computes the fraction of the same labeled neighbors to the total number of neighbors. The proposed method is evaluated on a variety of several standard UCI data sets. Experiments show the excellent improvement in accuracy in comparison with KNN method.

**Keywords:** MKNN, KNN Classification, Modified K-Nearest Neighbor, Weighted K-Nearest Neighbor, Neighbor Validation.
**PACS:** S 07.05.Kf

## INTRODUCTION

Pattern recognition is about assigning labels to objects which are described by a set of measurements called also attributes or features. Current research builds upon foundations laid out in the 1960s and 1970s. Because pattern recognition is faced with the challenges of solving real-life problems, in spite of decades of productive research, graceful modern theories still coexist with ad hoc ideas, intuition and guessing [1].

There are two major types of pattern recognition problems: unsupervised and supervised. In the supervised category which is also called supervised learning or classification, each object in the data set comes with a pre-assigned class label. Our task is to train a classifier to do the labeling, sensibly. Most often the labeling process cannot be described in an algorithmic form. So we supply the machine with learning skills and present the labeled data to it. The classification knowledge learned by the machine in this process might be obscure, but the recognition accuracy of the classifier will be the judge of its adequacy [1]. The new classification systems try to investigate the errors and propose a solution to compensate them [2-5]. There are many classification and clustering methods as well as the combinational approaches [6-8].

K-Nearest Neighbor (KNN) classification is one of the most fundamental and simple classification methods. When there is little or no prior knowledge about the distribution of the data, the KNN method should be one of the first choices for classification. It is a powerful non-parametric classification system which bypasses the problem of probability densities completely [9]. The KNN rule classifies x by

assigning it the label most frequently represented among the K nearest samples; this means that, a decision is made by examining the labels on the K-nearest neighbors and taking a vote. KNN classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine.

In 1951, Fix and Hodges introduced a non-parametric method for pattern classification that has since become known the K-nearest neighbor rule [10]. Later in 1967, some of the formal properties of the K-nearest neighbor rule have been worked out; for instance it was shown that for K=1 and $n \to \infty$ the KNN classification error is bounded above by twice the Bayes error rate [11]. Once such formal properties of KNN classification were established, a long line of investigation ensued including new rejection approaches [12], refinements with respect to Bayes error rate [13], distance weighted approaches [14, 15], soft computing [16] methods and fuzzy methods [17, 18].

ITQON et al. in [19] proposed a classifier, TFkNN, aiming at upgrading of distinction performance of KNN classifier and combining plural KNNs using testing characteristics. Their method not only upgrades distinction performance of the KNN but also brings an effect stabilizing variation of recognition ratio; and on recognition time, even when plural KNNs are performed in parallel, by devising its distance calculation it can be done not so as to extremely increase on comparison with that in single KNN.

Alizadeh et al. in [20] proposed a new classification method for enhancing the performance of K-Nearest Neighbor which uses clustering ensemble method. This new combinational method is called Nearest Cluster Ensemble, NCE. Inspiring the traditional KNN algorithm, the main idea in their method is classifying the test samples according to their neighbor tags. First, the train set is clustered into a large number of partitions, so that any cluster expectedly includes a small number of samples. Then, the labels of cluster centers are determined using applying the majority vote between the class labels of individual members in the cluster. After that, the class label of a new test sample is determined according to the class label of the nearest cluster center. Finally, a simple majority vote is employed to aggregate the class labels of *M* classifiers. Computationally, the NCE method is faster then KNN, *K* times.

Some KNN advantages are described in follows: a) Simple to use; b) Robust to noisy training data, especially if the inverse square of weighted distance is used as the “distance” measure; and c) Effective if the training data is large. In spite of these good advantages, it has some disadvantages such as: a) Computation cost is quite high because it needs to compute distance of each query instance to all training samples; b) The large memory to implement in proportion with size of training set; c) Low accuracy rate in multidimensional data sets; d) Need to determine the value of parameter K, the number of nearest neighbors; e) Distance based learning is not clear which type of distance to use; and f) which attributes are better to use producing the best results. Shall we use all attributes or certain attributes only [21].

In this paper a new interesting algorithm is proposed which partially overcomes the low accuracy rate of KNN. Beforehand, it preprocesses the train set, computing the validity of any train samples. Then the final classification is executed using weighted KNN which is employed the validity as the multiplication factor.

The rest of this paper is organized as follows. Section II expresses the proposed algorithm which is called Modified K-Nearest Neighbor, MKNN. Experimental results are addressed in section III. Finally, section IV concludes.

## MODIFIED K-NEAREST NEIGHBOR

The main idea of the presented method is assigning the class label of the data according to *K* validated data points of the train set. In other hand, first, the validity of all data samples in the train set is computed. Then, a weighted KNN is performed on any test samples. Figure 1 shows the pseudo code of the MKNN algorithm.

```
Output_label := MKNN ( train_set , test_sample )
Begin
    For i := 1 to train_size
        Validity(i) := Compute Validity of i-th sample;
    End for;
    Output_label:=Weighted_KNN(Validity,test_sample);
    Return Output_label ;
End.
```

**FIGURE 1.** Pseudo-code of the MKNN Algorithm.

In the rest of this section the MKNN method is described in detail, answering the questions, how to compute the validity of the points and how to determine the final class label of test samples.

### Validity of the Train Samples

In the MKNN algorithm, every sample in train set must be validated at the first step. The validity of each point is computed according to its neighbors. The validation process is performed for all train samples once. After assigning the validity of each train sample, it is used as more information about the points.

To validate a sample point in the train set, the *H* nearest neighbors of the point is considered. Among the *H* nearest neighbors of a train sample *x*, validity(*x*) counts the number of points with the same label to the label of *x*. Eq. 1 is the formula which is proposed to compute the validity of every points in train set.

$$Validity(x) = \frac{1}{H}\sum_{i=1}^{H} S(lbl(x), lbl(N_i(x))) \qquad (1)$$

Where *H* is the number of considered neighbors and *lbl(x)* returns the true class label of the sample *x*. also, $N_i(x)$ stands for the *i*th nearest neighbor of the point *x*. The function *S* takes into account the similarity between the point *x* and the *i*th nearest neighbor. Eq. 2 defines this function.

$$S(a,b)=\begin{cases}1 & a=b\\ 0 & a\neq b\end{cases} \quad (2)$$

This is the standard font and layout for the individual paragraphs. The style is called "Paragraph." Replace this text with your text. The "Enter" key will take you to a new paragraph. If you need to insert a hard line break within the paragraph, please use Shift+Enter, rather than just tapping the "Enter" key.

## Applying Weighted KNN

Weighted KNN is one of the variations of KNN method which uses the *K* nearest neighbors, regardless of their classes, but then uses weighted votes from each sample rather than a simple majority or plurality voting rule. Each of the *K* samples is given a weighted vote that is usually equal to some decreasing function of its distance from the unknown sample. For example, the vote might set be equal to $1/(d_e+1)$, where $d_e$ is Euclidian distance. These weighted votes are then summed for each class, and the class with the largest total vote is chosen. This distance weighted KNN technique is very similar to the window technique for estimating density functions. For example, using a weighted of $1/(d_e+1)$ is equivalent to the window technique with a window function of $1/(d_e+1)$ if *K* is chosen equal to the total number of training samples [22].

In the MKNN method, first the weight of each neighbor is computed using the $1/(d_e+0.5)$. Then, the validity of that training sample is multiplied on its raw weight which is based on the Euclidian distance. In the MKNN method, the weight of each neighbor sample is derived according to Eq. 3.

$$W(i)=Validity(i)\times\frac{1}{d_e+0.5} \quad (3)$$

Where *W(i)* and *Validity(i)* stand for the weight and the validity of the *i*th nearest sample in the train set. This technique has the effect of giving greater importance to the reference samples that have greater validity and closeness to the test sample. So, the decision is less affected by reference samples which are not very stable in the feature space in comparison with other samples. In other hand, the multiplication of the validity measure on distance based measure can overcome the weakness of any distance based weights which have many problems in the case of outliers. So, the proposed MKNN algorithm is significantly stronger than the traditional KNN method which is based just on distance.

## EXPERIMENTAL RESULTS

This section discusses the experimental results and compares the MKNN method with original KNN algorithm.

## Data sets

The proposed method is evaluated on nine standard data sets, namely Iris, Wine, Isodata, SAHeart, Balance-scale, Bupa and Monk's problems (including three problems). None of the databases had missing values, as well as they use continuous attributes. These standard data sets which are obtained from UCI repository [23] are described as follows.

The iris database which is possibly one of the most frequently used benchmarks for evaluating classification and clustering algorithms is a well-defined problem with clear separating class boundaries. The data set contains 150 instances using three classes, where each class refers to a type of iris plant, namely *Setosa*, *Versicolour* and *Virginica*. This database uses four continuous attributes: *sepal length*, *sepal width*, *petal length* and *petal width*.

The Wine data set is the result of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines. This data set has been used with many others for comparing various classifiers. In a classification context, this is a well-posed problem with well-behaved class structures. It has three classes with 59, 71 and 48 instances. The more detail information about the wine data set is described in [24].

The Isodata set is the first test case in this study which is a two class data set and has 34 features as well as the 351 sample points.

The SAHeart data set which is obtained from www-stat.stanford.edu/ElemStatLearn is a retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. There are roughly two controls per case of CHD. Many of the CHD positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their CHD event. In some cases the measurements were made after these treatments. This data set has nine continuous features and two classes with the number of 463 instances. These data are taken from a larger dataset, described in [25].

The Balance-scale data set was generated to model psychological experimental results. Each example is classified as having the balance scale tip to the right, tip to the left, or be balanced which means three classes. The attributes are the left weight, the left distance, the right weight, and the right distance. It means this data set has four attributes. It has total 625 samples which include 49 balanced, 288 left, 288 right.

The Bupa data set is a two class data set for classification. It contains 345 data sample as well as six attributes.

In these six data sets, the instances are divided into training and test sets by randomly choosing 90% and 10% of instances per each of them, respectively. Also, all above mentioned data sets are become normalized with the mean of 0 and variance of 1, $N(0,1)$ before applying the algorithms.

The last experimented data set is Monk's problem which is the basis of a first international comparison of learning algorithms. There are three Monk's problems. The domains for all Monk's problems are the same. The second Monk's problem has added noise. For each problem, the domain has been partitioned into a train and test set. The number of Instances and attributes in all three problems are respectively, 432

and 6. These problems are two class problems. The train and test sets in all three Monk's problems are predetermined. The train sets in Monk 1, 2 and 3 are 124, 169 and 122, respectively.

## Experiments

All experiments are evaluated over 500 independent runs and the average results of these examinations are reported. In all experiments, the number of considered neighbors (the value of parameter H in Eq. 1) is set to a fraction of the number of train data which is empirically set to 10% of the train size.

Table 1 shows the results of the performance of classification using the presented method, MKNN, and traditional method, original version of KNN, comparatively.

**TABLE 1.** Comparison of recognition rate between the MKNN and KNN algorithm (%).

| | **K=3** | | **K=5** | | **K=7** | |
|---|---|---|---|---|---|---|
| | **KNN** | **MKNN** | **KNN** | **MKNN** | **KNN** | **MKNN** |
| **Monk 1** | 84.49 | **87.81** | 84.26 | **87.81** | 79.86 | **86.65** |
| **Monk 2** | 69.21 | **77.66** | 69.91 | **78.01** | 65.74 | **77.16** |
| **Monk 3** | 89.12 | **90.58** | 89.35 | **90.66** | 88.66 | **91.28** |
| **Isodata** | 82.74 | **83.52** | 82.90 | **83.32** | 80.50 | **83.14** |
| **Wine** | 80.89 | **83.95** | 83.79 | **85.76** | 80.13 | **82.54** |
| **Iris** | 95.13 | **95.50** | 95.83 | **95.90** | 95.32 | **95.51** |
| **Balance-sc** | 80.69 | **85.49** | 83.22 | **87.10** | 85.74 | **86.77** |
| **Bupa** | **63.51** | 63.30 | 60.01 | **62.52** | 60.41 | **63.29** |
| **SAHeart** | 67.51 | **69.51** | 65.59 | **69.49** | 66.21 | **69.95** |

The experiments show that the MKNN method significantly outperforms the KNN method, with using different choices of value *K*, over large variety of datasets. It is obvious that more information usually yields to more classification performance. Because of the MKNN classification is based on validated neighbors which have more information in comparison with simple class labels, it outperforms the KNN algorithm in performance.

Figure 2 investigates the effect of parameter *K* on accuracy of algorithms KNN and MKNN comparatively in four different data sets: Iris, Balance-scale, Bupa and SAHeart. The value of *K* is the odd numbers in the range of [3-15]. Although usually the MKNN method initially overwhelms the KNN algorithm, the results of two algorithms gradually close to each other by growing the value of *K*. It can be because of the larger values of *K* result in invalidity of the validity of train samples. For some data sets the KNN even dominates the MKNN method with large *K* (see Figures 2b and 2c).

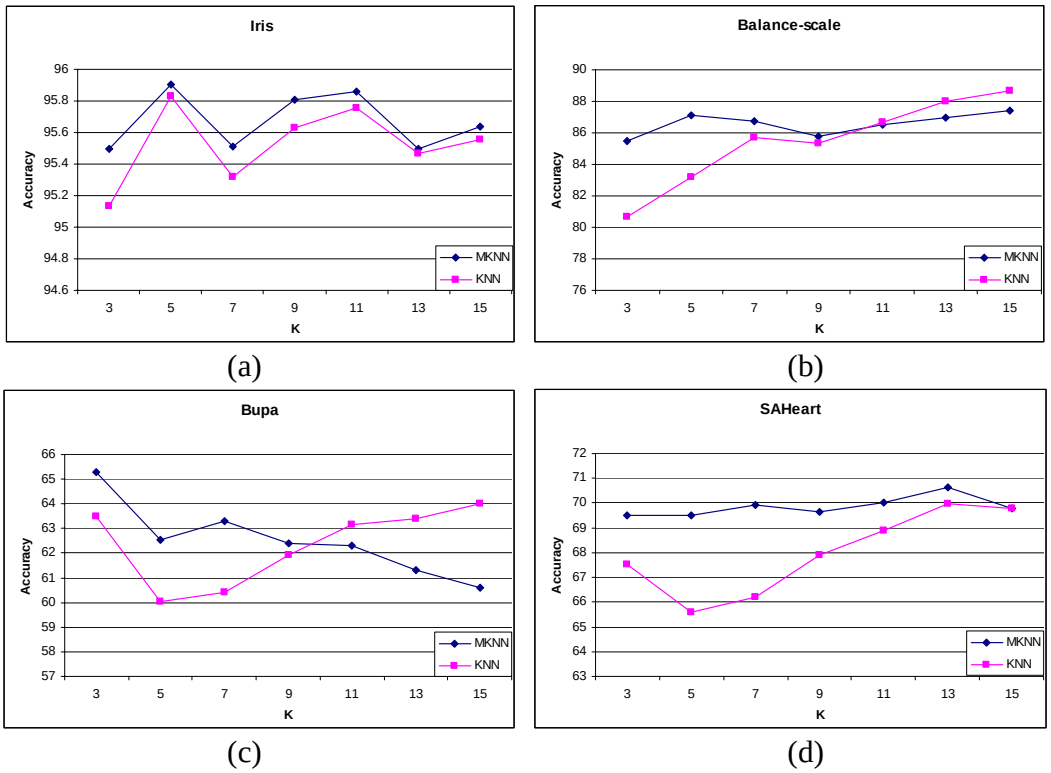


**FIGURE 2.** The effect of parameter *K* on accuracy of algorithms KNN and MKNN comparatively in four data sets (a) Iris (b) Balance scale (c) Bupa (d) SAHeart.

In addition, since computing the validity measure is executed only once in training phase of the algorithm, computationally, the MKNN method can be applied with the nigh same burden in comparison with the weighted KNN algorithm.

## CONCLUSION

In this paper, a new algorithm for improving the performance of KNN classifier is proposed which is called Modified K-Nearest Neighbor, MKNN. The proposed method which considerably improves the performance of KNN method employs a kind of preprocessing on train data. It adds a new value named “Validity” to train samples which cause to more information about the situation of training data samples in the feature space. The validity takes into accounts the value of stability and robustness of the any train samples regarding with its neighbors. Applying the weighted KNN which employs validity as the multiplication factor yields to more robust classification rather than simple KNN method, efficiently. The method evaluated on nine different benchmark tasks: Wine, Isodata, Iris, Bupa, Inosphere and three Monk’s problems. The results confirm authors' claim about its robustness and accurateness unanimously. So this method is better in noisy datasets and also in the case of outliers. Since the outliers usually gain low value of validity, it considerably yields to robustness of the MKNN method facing with outliers. The experiments on Monk 2 approve this claim.

# REFERENCES


1. L. I. Kuncheva, *Combining Pattern Classifiers, Methods and Algorithms*, New York: Wiley, 2005.
2. H. Parvin, H. Alizadeh, B. Minaei-Bidgoli and M. Analoui, "An Scalable Method for Improving the Performance of Classifiers in Multiclass Applications by Pairwise Classifiers and GA", *In Proc. of the Int. Conf. on Networked Computing and advanced Information Management by IEEE CS, (NCM08)*, Sep. 2008.
3. H. Parvin, H. Alizadeh, M. Moshki, B. Minaei-Bidgoli and N. Mozayani, "Divide & Conquer Classification and Optimization by Genetic Algorithm", *In Proc. of the Int. Conf. on Convergence and hybrid Information Technology by IEEE CS, (ICCIT08)*, Nov. 11-13, 2008.
4. H. Parvin, H. Alizadeh, B. Minaei-Bidgoli and M. Analoui, "CCHR: Combination of Classifiers using Heuristic Retraining", *In Proc. of the Int. Conf. on Networked Computing and advanced Information Management by IEEE CS, (NCM 2008)*, Korea, Sep. 2008.
5. H. Parvin, H. Alizadeh and B. Minaei-Bidgoli, "A New Approach to Improve the Vote-Based Classifier Selection", *In Proc. of the Int. Conf. on Networked Computing and advanced Information Management by IEEE CS, (NCM 2008)*, Korea, Sep. 2008.
6. H. Alizadeh, M. Mohammadi and B. Minaei-Bidgoli, "Neural Network Ensembles using Clustering Ensemble and Genetic Algorithm", *In Proc. of the Int. Conf. on Convergence and hybrid Information Technology by IEEE CS, (ICCIT08*), Nov. 11-13, 2008, Busan, Korea.
7. H. Parvin, H. Alizadeh and B. Minaei-Bidgoli, *A New Method for Constructing Classifier Ensembles*, International Journal of Digital Content: Technology and its Application, JDCTA, ISSN: 1975-9339, 2009 (in press).
8. H. Parvin, H. Alizadeh and B. Minaei-Bidgoli, *Using Clustering for Generating Diversity in Classifier Ensemble*, International Journal of Digital Content: Technology and its Application, JDCTA, ISSN: 1975-9339, 2009 (in press).
9. B.V. Darasay, *Nearest Neighbor pattern classification techniques*, Las Alamitos, LA: IEEE CS Press.
10. E. Fix, J.L. Hodges, *Discriminatory analysis, nonparametric discrimination: Consistency properties*. Technical Report 4, USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
11. Cover, T.M., Hart, P.E. Nearest neighbor pattern classification. IEEE Trans. Inform. Theory, IT-13(1):21–27, 1967.
12. Hellman, M.E. *The nearest neighbor classification rule with a reject option*. IEEE Trans. Syst. Man Cybern., 3:179–185, 1970.
13. K. Fukunaga,, L. Hostetler, *k-nearest-neighbor bayes-risk estimation*. IEEE Trans. Information Theory, 21(3), 285-293, 1975.
14. S.A. Dudani, *The distance-weighted k-nearest-neighbor rule*. IEEE Trans. Syst. Man Cybern., SMC-6:325–327, 1976.
15. T. Bailey, A. Jain, *A note on distance-weighted k-nearest neighbor rules*. IEEE Trans. Systems, Man, Cybernetics, Vol. 8, pp. 311-313, 1978.
16. S. Bermejo, J. Cabestany, *Adaptive soft k-nearest-neighbour classifiers*. Pattern Recognition, Vol. 33, pp. 1999-2005, 2000.
17. A. Jozwik, *A learning scheme for a fuzzy k-nn rule. Pattern Recognition Letters*, 1:287–289, 1983.
18. J.M. Keller, M.R. Gray, J.A. Givens, *A fuzzy k-nn neighbor algorithm*. IEEE Trans. Syst. Man Cybern., SMC-15(4):580–585, 1985.
19. K. ITQON, *Shunichi and I. Satoru, Improving Performance of k-Nearest Neighbor Classifier by Test Features*, Springer Transactions of the Institute of Electronics, Information and Communication Engineers 2001.
20. H. Alizadeh, S.K. Amirgholipour, N.R. Seyedaghaee and B. Minaei-Bidgoli, "Nearest Cluster Ensemble (NCE): Clustering Ensemble Based Approach for Improving the performance of K-Nearest Neighbor Algorithm", *11th Conf. of the Int. Federation of Classification Societies, IFCS 2009*, March 13 – 18, 2009, Dresden, Germany.
21. R. O. Duda, P. E. Hart and D. G. Stork, *Pattern Classification* , John Wiley & Sons, 2000.
22. E. Gose, R. Johnsonbaugh and S. Jost, *Pattern Recognition and Image Analysis*, Prentice Hall, Inc., Upper Saddle River, NJ 07458, 1996.
23. C.L. Blake, C.J. Merz, UCI Repository of machine learning databases:

*http://www.ics.uci.edu/~mlearn/ MLRepository.html*, 1998.
24. S. Aeberhard, D. Coomans and O. de Vel, *Comparison of Classifiers in High Dimensional Settings*, Tech. Rep. no. 92-02, Dept. of Computer Science and Dept. of Mathematics and Statistics, James Cook University of North Queensland.
25. Rousseauw et al., South African Medical Journal, 1983.