Do We Need Whatever More than k-NN?

Mirosław Kordos[1], Marcin Blachnik[2], Dawid Strzempa[1]

[1] University of Bielsko-Biała, Department of Mathematics and Computer Science, Bielsko-Biała, Willowa 2, Poland;
[2] Silesian University of Technology, Electrotechnology Department, Katowice, Krasinskiego 8, Poland;

Abstract. Many sophisticated classification algorithms have been proposed. However, there is no clear methodology of comparing the results among different methods. According to our experiments on the popular datasets, k-NN with properly tuned parameters performs on average best. Tuning the parametres include the proper k, proper distance measure and proper weighing functions. k-NN has a zero training time and the test time can be significantly reduced by prior reference vector selection, which needs to be done only once or by applying advanced nearest neighbor search strategies (like KDtree algorithm). Thus we propose that instead of comparing new algorithms with an author's choice of old ones (which may be especially selected in favour of his method), the new method would be rather compared first with properly tuned k-NN as a gold standard. And based on the comparison the author of the new method would have to aswer the question: "Do we really need this method since we already have k-NN?"

1 Introduction

K-nearest-neighbor classification was developed from the need to perform discriminant analysis when reliable parametric estimates of probability densities are unknown or difficult to determine. In an unpublished US Air Force School of Aviation Medicine report in 1951, Fix and Hodges introduced a non-parametric method for pattern classification that has since become known the k-nearest neighbor (k-NN) [1].
Since k-NN classification is one of the most fundamental and simple classification methods, it should be one of the first choices for a classification study when there is little or no prior knowledge about the distribution of the data [13]. Our paper investigates if k-NN should be not only the first but in most cases also the only classification algorithm applied to a given problem.
An approach to comparison of the accuracy of various classification algorithms depends on the author of the given paper. In general many authors try to show that their method is the best by comparing it with other methods that performs poorer on the given data. Frequently on of the compared algorithms is k-NN, however with k=1, rarely k=3. As our experiments shown, a good choice for k is rather any number between 10 and 20 than between 1 (or even 3), what changes the k-NN performance dramatically. The fact is rarely mentioned by the authors. There can be two reasons of that: first, they are unaware of this, because they do not perform the experiments themselves, but rather take the results from other sources and second, they are aware of this, but deliberately

use 1-NN (sometimes 3-NN) in order to do better in comparison with k-NN. A proper selection of k is most important. The weighing schemes mentioned below also improve classification results but in a lesser degree then setting a reasonable k value.
The next issue is that the vectors that lie closer to the test vector should be paid more attention to by applying rather a weighted k-NN that the raw form of k-NN. The simplest weighting system is to make the weights inversely proportional do the distance between the given point and the test point. However, other schemes may be used as well.
And finally we should take into account that different features have different prediction ability measured e.g. by feature ranking. Some classifications, as neural networks or decision trees have already in various ways embedded the feature weighting mechanism. Since the raw k-NN does not, it is advised to add the weighing scheme while determining the class of the test vector.
Another great impact on the accuracy of kNN classifier has appropriate reference vector selection. In many real problem storing whole data matric is difficult, and considering distance calculation which has $O(n^2)$ complexity and large memory requirements may radically restrict kNN's usability, unless some means are undertaken as discussed in the following sections.
The k-NN algorithm is really much simpler than many other methods and on a whole spectrum of datasets performs very well in comparison to them. Thus, we propose to establish o gold standard of comparing any new classification algorithm to k-NN with optimally tuned parameters. To enable this task in an easy way we are currently creating a web page at www.kordos.com/knn, where the user can submit any data set for classification and compare the result of theri algorithm to that of (almost) optimally tuned k-NN.

2 k-Nearest Neighbors Algorithm

The Basis of k-NN In k-NN a vector is classified by a majority voting of its neighbors, and assigned to the class most common among its k nearest neighbors. For example in 5-NN if two nearest neighbors belong to class A and three to class B, the vector is assigned to class B. If k = 1, then the vector is simply assigned to the class of its nearest neighbor. The same method can be used for regression, by simply assigning the output value of a vector to be the average of the values of its k nearest neighbors.
Both in classification and regression it can be useful to weight the contributions of the neighbors, so that the nearer neighbors contribute more to the average than the more distant ones. K-NN is a lazy algorithm; it has a zero training time, but all the calculations are performed at the predictions time. For big datasets that can be long if the distance is to be calculated to each training vector. However, the simple solution is to use tree based search strategies instead of typically done linear search. The most common choices are KDTree or ballTree algorithms. Another solution, however not so smart as the previous one, is to cluster the data first (what needs to be done only once) and then perform k-NN twice. Once on the cluster centers to find k nearest clusters and the second one only on the vectors within the k nearest clusters (the second run is needed only if vectors in the k nearest clusters belong to different classes).

Distance Matrices The distance between two vectors can be calculated using different distance measures. Most frequently Minkovski distance is used:

$$d_{x,y} = \sqrt[\lambda]{\sum_{i=1}^{f} (|x_i - y_i|)^{\lambda}} \quad (1)$$

Where d_x, d_y is the distance between the vector x and y in f-dimensional feature space. For the exponent $\lambda = 1$, the above becomes the Manhattan norm, for $\lambda = 2$ Euclidean norm and for $\lambda = \infty$ Chebyshev norm. We performed some experiments to determine the distance measure influence on k-NN classification accuracy.

Weighting We discuss two possible weighting schemes: one based on the distance and one on the feature ranking. In the case of the distance-based weighting, the class of a given vector y is determined by the greatest sum W_c of weighed distances of k nearest neighbors from class c. In the simplest case the weight can be proportional to the inverse of the distance between the two vectors (eq. 2-left). In the experiments, we however used the exponential weighting scheme (eq. 2-right).

$$W_c = \sum_{i=1}^{n} d_{x_i,y}^{-1} \qquad W_c = \sum_{i=1}^{n} 2^{-0.2 d_{xi,y}} \quad (2)$$

In the weighting by future ranking, first a correlation between each single feature and class (or output value in case of regression) is calculated and then the distance in each feature dimension is multiplied by this correlation (eq. 3a).

$$d_{x,y} = \sqrt[\lambda]{\sum_{i=1}^{f} (|c_i(x_i - y_i)|)^{\lambda}} \quad (3)$$

not the only possible weighting by feature, but this is the simplest one and that one we used in the experiments. The aim of that weighting is to limit the noise that can be contained in the less related features.
As the experiments showed using the distance-based weighing improves the results and using also the feature-based weighting improves the results even more. However, both improvements are relatively small.

Limitations of k-NN The limitations of k-NN are results of the shape of decision borders produced by the algorithm, that is turning the corners and a tendency to suppress a narrow passages, as shown in the figures below. That is however not a frequent case in a real world datasets. To prevent these effects one can increase the exponent in the Minkovski distance measure, but that however can lead to an overall decrease in accuracy (see the experiment section), because only the distance in one feature direction will have a practical influence on the decision making.

Another option is to utilize Mahalanobis distance function, or other data-dependent matrices like probabilistic measures (VDM or MRM) that depends on data distribution. These last two metrics can be also utilized in case of nominal attributes, or in non homogeneous datasets that consists of diferent feature types. [22]

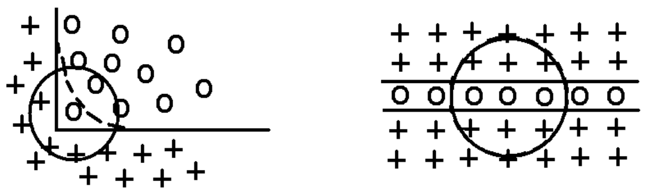

Fig. 1. Limitations of k-NN algorithm

3 Experimental Results

We performed experiments with k-NN on the popular benchmark datasets from the UCI Machine Learning repository [3]: Iris, Appendicitis, WBC, Diabetes and Ionosphere. We compared the following algorithms: standard k-NN, k-NN weighted by distance [4], k-NN weighted by distance and feature, MLP [5], CART [6], C4.5 [7], RBF [5], LVQ [8], LDA [9], FDA [10], SMLP [11], SSV [12], IncNet [13], FSM [14], SVM [15], Naive Bayes [16]. There are algorithms that on a particular dataset performs slightly better than k-NN weighted by distance and feature, however first, the difference is very little and second there is no other algorithms than on average performs as well as k-NN. We performed the experiments only with k-NN and presented the other algorithm results as reported either by their authors or by the comparison projects [17–19].

4 Conclusions

As the experimental results shows in most cases the simple k-NN is not only sufficient but also one of the best and most universal algorithm.
So do we need whatever more than k-NN? The obvious answer is yes, because as no free lunch theorem says there is no best method that would beat all the others, but as our results proved, when k-NN is correctly tuned its accuracy can be comprehensive to other even much more sophisticated algorithms. Of course k-NN is not suitable for all possible problems, specially for very large datasets and real time prediction processes it requires some preprocessing of the data (see section 2) to reduce its computational complexity and memory requirements.
Another problem is the flexibility of k-NN. It may easily overfit the data, so tuning k-NN classifier requires good accuracy prediction methodology that would overcome

[!h]

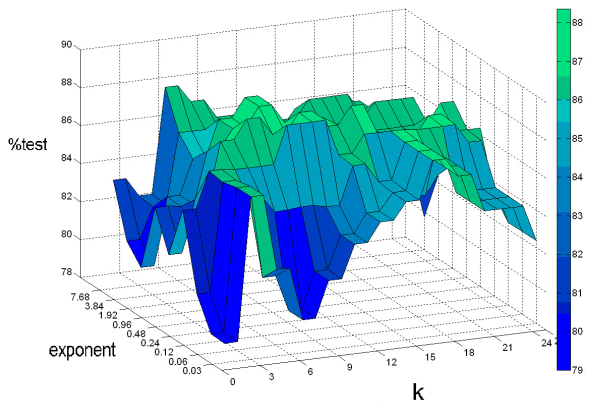

Fig. 2. Influence of the number of neighbor (k) and the power in the Minkovski distance metrics on the classification accuracy of unweighted k-NN in 10-fold crossvalidation on the Appendicitis dataset.

Table 1. Classification accuracy on the Wisconsin Breast Cancer (WBC) dataset.

algorithm	test	source
5-NN	97.4	this work, [1]
5-NN weighted by distance	97.3	this work, [1]
5-NN weighted by distance and attribute	97.6	this work
C4.5	94.7	[17]
LVQ	96.6	[17]
FDA	96.7	[17]
SSV	96.3	[12]
CART	93.5	[17]
LDA	96.0	[17]
MLP+BP	96.7	[17]
SVM lin, opt C	96.7	[17]
incNet (3000 epochs, 40 neurons)	97.1	[13]
SMLP	97.1	[11]

such limitation like bootstrapping or cross validation. We believe that the conclusion that derives from our experiments is that the k-NN algorithm should become a reference model to all other algorithms developed at any research groups. We have begun creating a simple web page at www.kordos.com/knn that allows fine tuning of the k-NN classifier and performing experiments on any uploaded data. Results of the experiments can then be used for comparison with other methods.

[!h]

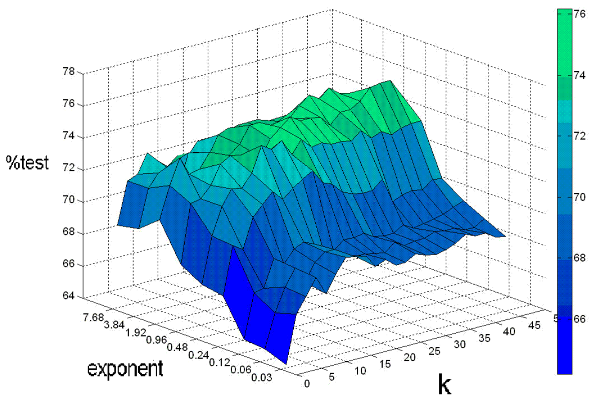

Fig. 3. Influence of the number of neighbor (k) and the power in the Minkovski distance metrics on the classification accuracy of unweighted k-NN in 10-fold crossvalidation on the Diabetes dataset.

Table 2. Classification accuracy on the Appendicitis dataset.

algorithm	test	source
11-NN	88.4	this work, [1]
11-NN weighted by distance	88.9	this work, [1]
11-NN weighted by distance and attribute	89.4	this work
SMLP	88.2	[20]
Nad've Bayes	83.0	[17]
SVM	88.1	[18]
SSV	87.8	[12]
CART	84.9	[18]
SMLP	88.2	[11]
FSM	87.6	[14]
incNet (1100 epochs, 30 neurons)	90.1	[13]
MLP+BP	85.8	[19]

Table 3. Classification accuracy on the Diabetes dataset.

algorithm	test	source
23-NN	76.2	this work, [1]
23-NN weighted by distance	76.6	this work, [1]
23-NN weighted by distance and attribute	77.0	this work
SVM, Gauss, C, sigma opt	77.4	[20]
RBF	75.7	[17]
LVQ	76.0	[17]
CART	74.7	[17]
C4.5	73.0	[17]
SSV	74.8	[12]
MLP+BP	75.2	[19]

Table 4. Classification accuracy on the Iris dataset.

algorithm	test	source
13-NN	96.7	this work, [1]
13-NN weighted by distance	96.7	this work, [1]
13-NN weighted by distance and attribute	96.8	this work
Naive Bayes	97.3	[17]
NEFCLASS	96.7	[18]
trainable fuzzy system	96.0	[12]

Table 5. Classification accuracy on the Ionosphere dataset.

algorithm	test	source
13-NN	96.7	this work, [1]
13-NN weighted by distance	96.8	this work, [1]
13-NN weighted by distance and attribute	97.1	this work
IB3	96.7	[21]
C4.5	94.9	[17]
SVM	93.2	[17]
CART	88.9	[17]
FSM	92.8	[14]
MLP+BP	96.0	[18]

References

1. E. Fix, J. L. Hodges, "Discriminatory analysis, nonparametric discrimination: Consistency properties", USAF School of Aviation Medicine, Randolph Field, Texas, 1951.
2. $http : //www.scholarpedia.org/article/K - nearest_neighbor$
3. UCI Machine Learning Repository, $http : //archive.ics.uci.edu/ml$
4. D. Strzempa, "Internet System for Data Classification" (in Polish), MSc Thesis, The Silesian University of Technology, Katowice 2008, available at $http : //www.ath.bielsko.pl/~mkordos/mgr/ds2008.pdf$
5. R. O. Duda, et. al., "Pattern Classification", New York, Wisley, 2001
6. L. Breiman et. al. "Classification and Regression Trees", Wadsworth, CA, 1984
7. J. R. Quinlan, "C4.5: Programs for Machine Learning", Morgan Kaufmann, 1993
8. T. Kohonen, "Statistical Pattern Recognition Revisited", Elsevier, 1990
9. R. Schalkoff, "Pattern Recognition: Statistical, Structural and Neural Approaches", Wiley, 1992
10. R. A. Fisher, "The use of multiple measurements in taxonomic problems", Wiley, 1950
11. M. Kordos, "Search-based Algorithms for Multilayer Perceptrons", PhD Thesis, The Silesian University of Technology, Gliwice 2005, available at $http : //www.fizyka.umk.pl/~kordos/pdf/MKordos - PhD.pdf$
12. K. Grabczewski, "Application of SSV Criterion for generating classification rules" (in Polish), PhD Thesis, Nicholaus Copernicus University, Torun 2003
13. N. Jankowski, "Ontogenic Neural Networks for Medical Data Classification" (in Polish), PhD Thesis, Nicholaus Copernicus University, Torun 1999
14. Torun 1999 R. Adamczak, "Neural networks application for experimental data classification" (in Polish), PhD Thesis, Nicholaus Copernicus University, Torun 2001
15. B. Schlkopf, A. J. Smola, "Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond", MIT Press, 2001
16. D. Michie, D. J. Spiegelhalter, C. C. Taylor, "Machine Learning, neural and statistical classification", Elis Horwood, London 1994
17. B. Ster, A Dobnikar, "Neural Networks in medical diagnosis: Comparison with other methods", EANN '96, pp. 427-430, 1996
18. F. Zarndt, "A comprehensive case study: An examination of machine learning and connectionists algorithms", MSc Thesis, Department of Computer Science, Brigham Young University, 1995
19. S. M. Weiss, I. Kapouleas, "An empirical comparison of pattern Recognition, neural nets and machine learning classification methods", Reading in Machine Learning, Morgan Kauffman Publ, CA, 1990
20. $http : //www.fqs.pl/business_intelligence/products/ghostminer/product_overview$
21. $http : //www.is.umk.pl/projects/datasets.html$
22. Blachnik M, Duch W, Wieczorek T, Probabilistic distance measures for prototype-based rules. Proc. of the 12th Int. Conference on Neural Information Processing (ICONIP'2005), Taipei, Taiwan, pp. 445-450