Dealing with noisy data in the context of k-NN Classification

Stefanos Ougiaroglou
stoug@uom.gr

Georgios Evangelidis
gevan@uom.gr

Department of Applied Informatics
School of Information Sciences, University of Macedonia
156 Egnatia St, GR-54006 Thessaloniki, Greece

ABSTRACT

Like many other classifiers, k-NN classifier is noise-sensitive. Its accuracy highly depends on the quality of the training data. Noise and mislabeled data, as well as outliers and overlaps between data regions of different classes, lead to less accurate classification. This problem can be dealt with by adopting either a large k value or by pre-processing the training set with an editing algorithm. The first strategy involves trial-and-error attempts to tune the value of k, while the second strategy constitutes a time-consuming pre-processing step. This paper discusses and compares these two strategies and reveals their advantages and drawbacks.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

Keywords

k-NN classification, editing, noisy data

1. INTRODUCTION

Classification is an important data mining task that has attracted the attention of both academia and industry. Classification algorithms (or classifiers) attempt to assign new items to a set of predefined classes on the basis of some training data, i.e., a set of already classified items. Classifiers can be either eager or lazy (or instance-based). While all classifiers aim for accurate predictions, they differ on how they work. An eager classifier pre-processes the training set and builds a model that is then used to classify new items. Lazy classifiers do not build any model, but instead, they classify new items by examining the available training data. Essential role for the efficiency and the effectiveness of all classifiers plays the size and the "quality" of the training set. Here, we focus on the effectiveness and on the "quality" of the training set.

k-Nearest Neighbours (k-NN) classifier [1] is an extensively used lazy classifier. It is simple and easy to implement and can be exploited in many domains. Also, it is analytically tractable and, for $k = 1$ and unlimited items, the error rate is asymptotically never worse than twice the minimum possible, which is the Bayes rate [1]. k-NN classifier uses the training set whenever a new item has to be classified. It classifies an item x by searching in the training set and retrieving the k nearest items (neighbours) to x according to a distance metric. Then, x is classified to the most common class among the classes of its k nearest neighbours. This class is determined via nearest neighbours voting.

Like many other classifiers, k-NN classifier is noise sensitive. Accuracy depends on the "quality" of the training set. Noise and mislabeled data, as well as overlaps between data areas of different classes, affect accuracy. This problem can be dealt with by adopting one of the following strategies: (i) application of an editing algorithm that pre-processes the training set in order to remove the irrelevant items, or, (ii) usage of a large k value that extends the examined neighbourhood and, thus, can smooth out the impact of noise.

This paper attempts to evaluate the advantages and drawbacks of the two strategies. The contribution is an experimental study that compares the strategies and reveals the strategy that should be adopted in certain circumstances.

The rest of the paper is organized as follows. Section 2 shows how a large value for k renders k-NN classifier noise-tolerant. Section 3 concerns editing. Section 4 presents the experimental study that compares the strategies. Finally, Section 5 concludes the paper.

2. NOISE AND THE VALUE OF K

The accuracy achieved by k-NN classifier depends on the selection of the value of k. The value of k that achieves the highest accuracy depends on the dataset used. Its determination implies tuning via trial-and-error pre-processing tasks. Although the determination of k can not follow any rule and the "best" k may be different for different datasets, usually, larger k values are appropriate for datasets with noise since they examine larger neighbourhoods. However, large k values fail to clearly define the boundaries between distinct classes. Small k values render the classifier more noise sensitive. Thus, in cases of training sets that contain noise, classification may be less accurate. In other words, the value of k defines the size of the examined neighbourhood. Practically, the larger the number of neighbours, the lower the impact of noisy data in determining the correct

BCI '15 September 2-4, 2015, Craiova, Romania
© 2015 ACM. ISBN 978-1-4503-3335-1/15/09...$15.00
DOI: 10.475/123_4

class label for a new item.

In cases of training sets with high level of noise, the determination of the value of k that achieves the highest accuracy is a difficult task that involves costly and time-consuming trial-and-error pre-processing. Certainly, the higher the level of noise in the training set is, the larger value for k is required, and thus, the more trial-and-error procedures are needed. We note that even the best k value may not be optimal. This happens because k-NN classifier uses a unique k value for all new items. Different k values may be optimal for different data areas. Consequently, heuristics for dynamic determination of k [6] can be adopted.

3. NOISE REMOVAL THROUGH EDITING

Editing aims to improve accuracy by improving the quality of the training set. To achieve this, editing algorithms try to remove noise, outliers and mislabeled items and smooth the decision boundaries between classes. Ideally, an editing task pre-processes the training set and builds an edited set without overlaps between the classes. Then, k-NN classifier searches for nearest neighbours in the edited set. Certainly, the k value of a k-NN classifier that uses an edited set (instead of the original training set) should also be tuned through a trial-end-error procedure. However, one expects that this procedure will be less time consuming since the data will be noise-free and a relatively small value for k will achieve the best possible classification accuracy.

Several editing algorithms have been proposed and can be found in the literature. Here, we present Wilson's Edited Nearest Neighbor (ENN) rule [12] that is the reference editing algorithm and constitutes the base of all other editing algorithms. In addition, ENN-rule is the algorithm that we used in our experimentation (see Section 4).

Algorithm 1 presents ENN-rule. Initially, the edited set (ES) is set to be equal to the training set (TS) (line 1). For each item x of TS, ENN-rule searches in TS and retrieves its k nearest neighbours (line 3). If x is misclassified by the majority vote of these neighbours, it is removed from ES (lines 4–7). ENN-rule considers misclassified items to be noise and, thus, they are removed. ENN-rule must compute all distances between the training items.

Algorithm 1 ENN-rule

```
Input: $TS, k$
Output: $ES$
1: $ES \leftarrow TS$
2: for each $x \in TS$ do
3:     $NNs \leftarrow$ find the $k$ nearest to $x$ neighbors in $TS - \{x\}$
4:     $majorClass \leftarrow$ find the most common class of $NNs$
5:     if $x_{class} \neq majorClass$ then
6:         $ES \leftarrow ES - \{x\}$
7:     end if
8: end for
9: return $ES$
```

Like k-NN classifier, ENN-rule uses the k parameter. Its determination is an issue that should also be addressed. [13, 3] consider $k = 3$ to be a typical setting. This is adopted in many papers (e.g. [7]). However, in some cases, researchers determine the value of k that achieves the best performance through trial-and-error (e.g., [11]). In [12], the impact of k is discussed in detail. Moreover, in [4], a large number of k values are evaluated. It turns out that the best k value depends on the dataset at hand and should be determined by considering the item distribution.

All other editing algorithms either extends or is based on the idea of ENN-rule. For instance, All-kNN [10] is a variation of ENN-rule that iteratively executes ENN-rule with different k values. Multiedit [2] divides the training set into random subsets. Then, it applies ENN-rule over each item of each one subset but searching for the one nearest neighbours in another subset. Repeated ENN (RENN) rule [10] is quite similar to All-kNN. RENN-rule iteratively applies ENN-rule until each item's majority of k nearest items have the same class. EENProb and ENNth [11] retrieve the k nearest neighbors and, then, perform editing based on probability estimations. In [4], another variation of ENN-rule is presented, where an item enters the edited set, only if all its k nearest items have the same class with it (distance ties increase the value of k). Sanchez et al. presented two editing procedures that use geometric information provided by proximity graphs [9]. k-NCN editing and its iterative version [7] are variations of ENN-rule that use the k nearest centroid neighbourhood classifier [8]. EHC [5] does not based on ENN-rule. It adopts a clustering procedure that finds homogeneous clusters and removes thes single item clusters.

4. EXPERIMENTATION

4.1 Experimental setup

The two strategies were compared using eight datasets distributed by the KEEL repository[1] and summarized in Table 1. Apart from the original versions of the datasets, we built and used new versions of them by artificially adding noise. For the datasets that include more than two classes (LR, PD, LS and YS), we used four versions, each with a different level of noise: 0%, 10%, 30%, 50%. The first version corresponds to the original dataset and the rest are its artificially built versions. The noise was added by setting the class of the 10% or 30% or 50% of the training items to a randomly chosen different class label. The other four datasets (MGT, PH, BN and PM) have only two classes. Therefore, they cannot afford high level of noise (the addition of noise strengthens the opposite class). Thus, we did not build versions with 50% of noise.

For editing purposes we used ENN-rule [12]. Based on [13, 3], the value of k of ENN-rule was set to be 3. For each version of each dataset, ENN-rule pre-processed the training set and built an edited set. Then, k-NN classifier used the edited set in order to classify new items. The comparison of the two strategies was implemented by executing k-NN classifier over the training set and the edited set and varying the value of k (in most cases from 1 to 50). All implementations were coded in C. We used the Euclidean distance as the distance metric. Also, we adopted a five-fold cross validation schema.

4.2 Comparison of the two strategies

Figure 1 presents the results. It presents eight diagrams. Each one corresponds to a dataset. The Y-axis measures the accuracy achieved while the X-axis indicates the corresponding value of k. Each diagram includes two curves for each version of the dataset. The black curve represents

[1] http://sci2s.ugr.es/keel/datasets.php

Table 1: Datasets description

dataset	Size	Attr.	Classes
Letter Recognition (LR)	20000	16	26
Magic G. Telescope (MGT)	19020	10	2
Pen-Digits (PD)	10992	16	10
Landsat Satellite (LS)	6435	36	6
Phoneme (PH)	5404	5	2
Banana (BN)	5300	2	2
Yeast (YS)	1484	8	10
Pima (PM)	768	8	2

the measurements corresponding to the k-NN classifier that uses the original training set (TS) (without noise removal), while the grey curve represents measurements corresponding to the edited set (ES). A notation like "ES-10%" means that the specific curve corresponds to the edited set built by the version with 10% noise. By examining the diagrams, we make the following observations:

(i) Almost in all cases, the k-NN classifier that uses the original training set can achieve higher accuracy than the one that uses the edited set. This is not true only in the cases of MGT, BN and PH with 30% extra noise. But, for MGT and BN, it is evident that the k-NN classifier that uses the original training set with an even larger value for k can eventually achieve higher accuracy than the one that uses the edited set. This does not seem to be achieved in the case of PH where the strategy of editing is clearly preferable. Therefore, we can conclude that if the major goal is to achieve the highest possible accuracy, editing should be avoided. Instead, an extensive trial-and-error procedure should be used on the training set to tune the value of k.

(ii) A k-NN classifier that follows the application of an editing algorithm avoids costly and time-consuming trial-and-error procedures. Figure 1 shows that the best possible accuracy is achieved with a relatively small value for k (i.e., $k < 10$). This is not true only in the case of the PM dataset with 30% extra noise where a larger value for k should be adopted. Therefore, we can conclude that when one wants to avoid lengthy trial-and-error procedures, one should use an editing algorithm as a pre-processing step.

(iii) For noise-free datasets, like LR, LS and PD, $k = 1$ should be adopted. A larger value for k may harm accuracy.

(iv) Finally, all diagrams confirm that the higher the level of noise, the larger the value of k that should be adopted.

5. CONCLUSIONS

We compared two strategies for dealing with noisy data in the context of k-NN classification. The first uses an editing algorithm. The second one tries to avoid the impact of noise by adopting a large value for k. Our experimentation illustrated that when the major goal is the highest possible accuracy one should adopt the latter approach. However, this implies costly trial-and-error procedures for tuning the large value for k. On the other hand, one can avoid the trial-and-error procedures by adopting an editing algorithm and achieving slightly lower accuracy.

An interesting direction for future work would be the development of a non-parametric adaptive classification model that, for each new item, automatically uses the appropriate number of nearest neighbours depending on the "quality" of the data region that surrounds the new item.

6. REFERENCES

[1] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Trans. Inf. Theor.*, 13(1):21–27, Sept. 2006.

[2] P. A. Devijver and J. Kittler. On the edited nearest neighbor rule. In *Proceedings of the Fifth International Conference on Pattern Recognition.* The Institute of Electrical and Electronics Engineers, 1980.

[3] M. García-Borroto, Y. Villuendas-Rey, J. A. Carrasco-Ochoa, and J. F. Martínez-Trinidad. Using maximum similarity graphs to edit nearest neighbor classifiers. In *Proceedings of the 14th Iberoamerican Conference on Pattern Recognition: Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications*, CIARP '09, pages 489–496, Berlin, Heidelberg, 2009. Springer-Verlag.

[4] K. Hattori and M. Takahashi. A new edited k-nearest neighbor rule in the pattern classification problem. *Pattern Recognition*, 33(3):521 – 528, 2000.

[5] S. Ougiaroglou and G. Evangelidis. EHC: Non-parametric editing by finding homogeneous clusters. In C. Beierle and C. Meghini, editors, *Foundations of Information and Knowledge Systems*, volume 8367 of *Lecture Notes in Computer Science*, pages 290–304. Springer, 2014.

[6] S. Ougiaroglou, A. Nanopoulos, A. N. Papadopoulos, Y. Manolopoulos, and T. Welzer-Druzovec. Adaptive k-nearest-neighbor classification using a dynamic number of nearest neighbors. In *Proceedings of the 11th East European conference on Advances in databases and information systems*, ADBIS'07, pages 66–82, Berlin, Heidelberg, 2007. Springer-Verlag.

[7] J. S. Sánchez, R. Barandela, A. I. Marqués, R. Alejo, and J. Badenas. Analysis of new techniques to obtain quality training sets. *Pattern Recogn. Lett.*, 24(7):1015–1022, Apr. 2003.

[8] J. Sánchez, F. Pla, and F. Ferri. On the use of neighbourhood-based non-parametric classifiers. *Pattern Recognition Letters*, 18(1113):1179 – 1186, 1997.

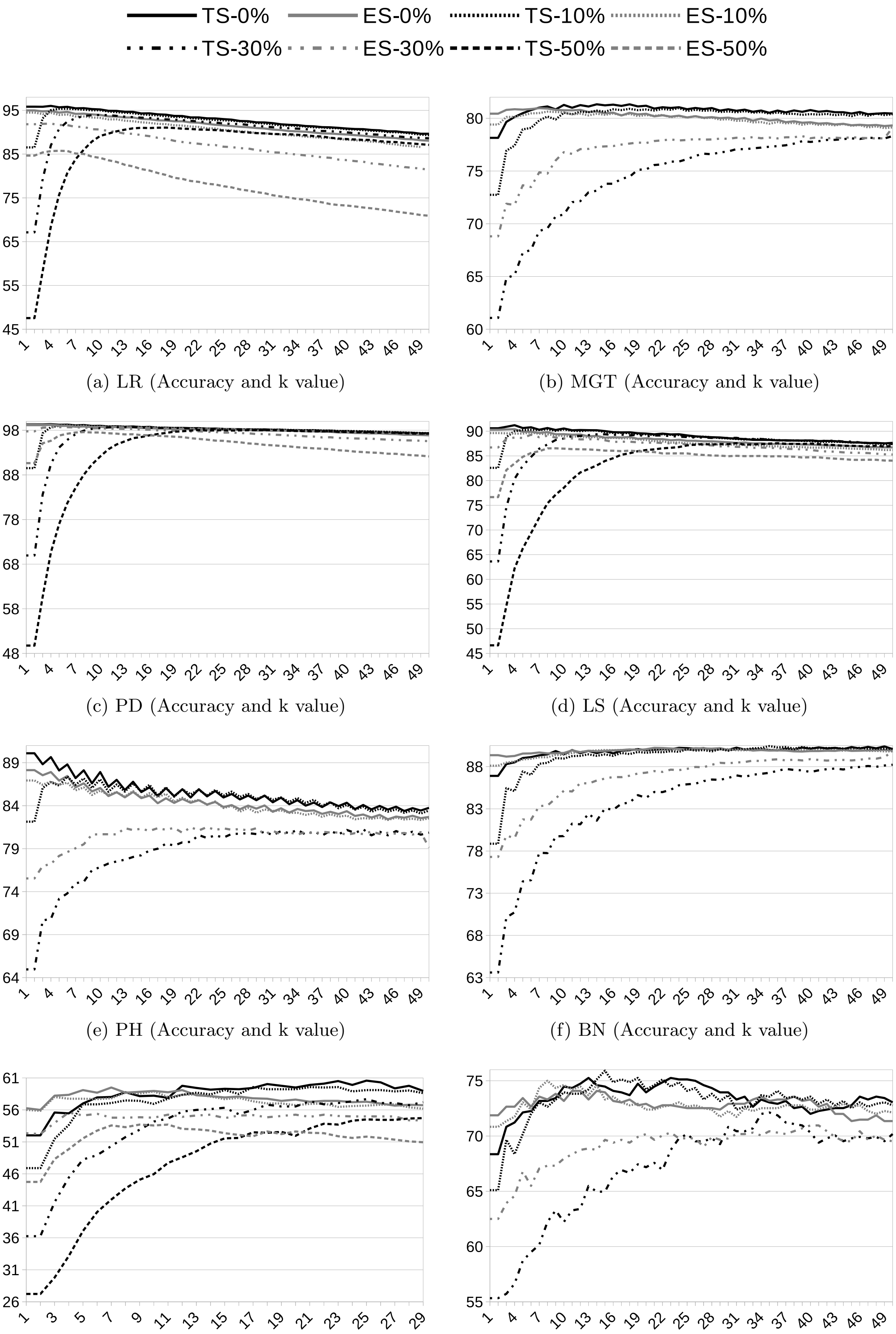
[9] J. Sánchez, F. Pla, and F. Ferri. Prototype selection for the nearest neighbour rule through proximity graphs. *Pattern Recognition Letters*, 18(6):507 – 513, 1997.

[10] I. Tomek. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6:448–452, 1976.

[11] F. Vázquez, J. S. Sánchez, and F. Pla. A stochastic approach to wilson's editing algorithm. In *Proceedings of the Second Iberian conference on Pattern Recognition and Image Analysis - Volume Part II*, IbPRIA'05, pages 35–42, Berlin, Heidelberg, 2005. Springer-Verlag.

[12] D. L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE trans. on systems, man, and cybernetics*, 2(3):408–421, July 1972.

[13] D. R. Wilson and T. R. Martinez. Reduction techniques for instance-basedlearning algorithms. *Mach. Learn.*, 38(3):257–286, Mar. 2000.



(a) LR (Accuracy and k value)

(b) MGT (Accuracy and k value)

(c) PD (Accuracy and k value)

(d) LS (Accuracy and k value)

(e) PH (Accuracy and k value)

(f) BN (Accuracy and k value)

(g) YS (Accuracy and k value)

(h) PM (Accuracy and k value)

Figure 1: Experimental results