Local Nearest Neighbor by Competition

F.J. Ferrer–Troyano * R. Giráldez J.C. Riquelme J.S. Aguilar–Ruiz
D.S. Rodríguez–Baena
Department of Computer Science, University of Sevilla
Avenida Reina Mercedes s/n, 41012 Sevilla, Spain
{ferrer,giraldez,riquelme,aguilar}@lsi.us.es

Abstract. The k–Nearest Neighbor algorithm presents as main drawbacks the selected distance and the value of the parameter k. Usually, this k value must be determined either by the user or by cross-validation. In this paper we introduce a local Nearest Neighbor algorithm that does not take this parameter from the user. Our approach evaluates different values of k that classified the *training* examples correctly and applies *k-NN* several times. With this heuristic, we propose an easy variation of the *k-NN* algorithm that improves the average accuracy given by the first *NN*.

1 Introduction

Nearest–Neighbor based algorithms have been target of experimental and theoretical studies, and practical application. With the *training* examples, the similarity function, and the method for processing missing attribute values as model itself, the basic *NN* classifier labels a new query with the label of its most similar example from those stored. Due to *NN* based algorithms are distance-based methods, normalization helps prevent attributes with initially large ranges from outweighing attributes with initially smaller ranges. Basic distances used in *NN* based algorithms are the Euclidean distance for continuous attributes and the Overlap distance for nominal attributes (both metrics were used in our experiments). To improve the accuracy with noise present in data, the *k-NN* algorithm introduces a parameter k so that for each new query q to be classified, the labels of its k nearest neighbors are observed. Then q is classified by the majority label. In case of a tie, either it can be randomly broken or by assigning that label whose average distance is the smallest. Usually k is heuristically determined by the user or by means of cross-validation [10]. Extending the classification criterion, the *k-NN*$_{wv}$ algorithms (Nearest Neighbor Weighted Voted) assign weights to the prediction made by each example. These weights are usually inversely proportional to the distance with respect to the query [4, 6] ($\frac{1}{d}$ or $1 - d$).

Therefore, the number k of examples observed and the metric used are decisive parameters. Since their introduction in the 1950's, important studies about error bounds for *NN* have been published [5, 1, 8, 9]. Researchers have also investigated new metrics [12] and new data representations [2] for improving accuracy and computational complexity. In [11] the behavior of *NN* and *k-NN* is studied in depth and experiments carried out with six synthetic data sets confirm the two following hypotheses: a) Noisy data need large values for k; b) The

*The research was supported by the Spanish Research Agency CICYT under grant TIC2001-1143-C03-02.

Figure 1: Horse Colic database. The used value of k can be determinant when the new query is a border point.

performance of *k-NN* is less sensitive to the choice of a metric. In addition, four classifiers are proposed (*Locally Adaptive Nearest Neighbor*, *local k-NN*) where the value of k can be different for each new example q to be classified. However, experiments with UCI datasets [3] show that these local Nearest Neighbor methods do not improve significantly the performance of *k-NN*. So it may be difficult to justify the added computational complexity. Nevertheless, to determine with certainty when local *NN* learners are beneficial is still an open problem. Based on a previous work [7], in this paper we introduced a local classifier that evaluates the *k-NN* algorithm several times. When a new query is near decision boundaries, the chosen *k* value can be determinant (see Figure 1). In this address, it might be possible to improve the classification accuracy by several evaluations of *k-NN* for a border-point query. In the following sections we describe *i-NN* (increasing k for the Nearest Neighbor) and several results obtained by applying the *k-NN* and *i-NN* algorithms with datasets from the UCI repository.

2 Description of the Algorithm

2.1 Approach

By means of the *k-NN* algorithm, when a new query is near decision boundaries, the assigned label can depend hardly on the value of the parameter k. At worst, the percentages of examples of each class can be similar at these regions. In such situation, the set formed by the *correct values* k_{e_i} associated with each example e_i might be either the empty set or a set formed by very high k values. That is, some examples either will not have any associated value k_{e_i} which classify it correctly or will have a so large associated value that the local adaptation of *k-NN* does not make sense with such k in these regions. So, this information can be not useful in some cases and unavailable in others. We not assume that it is possible to determine a pattern of the values of k to classify correctly the examples in overlapped regions. However, we claim that it is possible to improve the classification accuracy if the *k-NN* algorithm is evaluated several times on these regions. The idea is as simple as to give the new query more than one opportunity. Figure 1 illustrates these facts through projections on the plane of the two attributes values of the Horse-Colic database. The consequences of our approach can be argued in three cases:

- If q is a central example, the majority class might almost always be the same one for different values of k.

- If q is a noise example, either there will not be an associated k_q *correct value* or it will be large.
- If q is a border example, several evaluations of *k-NN* can avoid the classification errors.

Therefore, our decision is not to select the value for k but to find a limit k_{max} for several evaluations of *k-NN*. Due to this process must be non-parametric, k_{max} is calculated locally for each database. The method has been denominated *i-NN* (increasing the values of k for the Nearest Neighbors) because of it starts with $k = 1$ and it ends with $k = k_{max}$. In a previous work we observed that, in experiments using the same group of databases with 51 as the maximum of k, the classification accuracy given by *k-NN* becomes monotonically decreasing. So we have used such level to find the limit k_{max} and the possible values k_{min}.

2.2 The Algorithm

```
function classify-by-iNN(Set trainingSet; Query q) return Label
   fixLimit-And-RemoveOutliers(trainingSet, kmax, reducedSet)
   removeExamples-Without-CommonValue(reducedSet, q)
   reducedSet.orderBy(q)
   for(i:=1) to(kmax)
      label:=classify-by-kNN(i,q,reducedSet)
      frequencies[label]:=frecc[label]+1
   end for
return(index-With-HigherValue(frequencies))
```

Figure 2: Pseudocode of the *i-NN* algorithm.

Figure 2 contains pseudo-code of the algorithm. Let q be the query to be classified. Firstly, it is searched for each example e_i in the *training* set the minimum value $k_{min_{e_i}}$ which classifies it correctly through *k-NN* using the *training* examples. If an example has not associated a *classifier value* in the interval [1,51] then it is signed as non-classifiable. When all the examples are visited, those signed as non-classifiable are removed. Then the mean $mean_{k_{min}}$ and the standard deviation $sd_{k_{min}}$ of these $k_{min_{e_i}}$ values are calculated, so that k_{max} is assigned the odd number nearest to $\lceil mean_{k_{min}} + sd_{k_{min}} \rceil$. This process is carried out only one time for all the *test* examples to be classified. In a second step, the examples that do not share any value with q for all the attributes are removed. In order to classify the new query q, the resulting reduced set is ordered according to q. Finally, the *k-NN* algorithm is evaluated k_{max} times. In every evaluation the resulting label is increased in a vector of frequencies so that the label assigned will be the most frequent. The computational complexity of *i-NN* is:

$$\Theta(n^2 \cdot (\log(n) + 1) + n \cdot (\log(n) + m + 1));$$

where n is the number of *training* examples and m the number of attributes. In the first phase, the set *reducedSet* is generated. To do it, the $n-1$ nearest neighbors of each *training* example are ordered ($\Theta(n \cdot \log(n))$) and its *minimum k value* is calculated ($\Theta(n)$). Then the examples without an associated *minimum k value* are removed ($\Theta(n)$). In the second phase, the query q is classified ($\Theta(n \cdot (1 + \log(n)))$). The examples that have not in common any value with

Table 1: Classification accuracy obtained for a set of databases from the UCI repository by 10-folds cross-validation. Columns 1 and 4 show the rates by *k-NN* with $k = 1$ (the best value for all databases) and the best k found for each database, respectively. Columns 2 and 3 show the rates obtained by *i-NN* and *i-NN*$_{wv}$, respectively. Columns 4 and 5 show the limits for the evaluations of *i-NN* and *i-NN*$_{wv}$, respectively. The best k value obtained for *k-NN* is reported in Column 6.

Domain	*1-NN*	*i-NN*	*i-NN*$_{wv}$	*k-NN*$_{best}$	k_{iNN}	$k_{iNN_{wv}}$	k_{kNN}
Anneal	91.76	88.53	91.65	91.76	3	3	1
Audiology	75.66	71.24	77.88	75.66	7	7	1
Autos+	75.12	71.70	80.0	75.12	7	9	1
Balance-Scale+	77.76	83.04	83.04	89.76	3	3	21
Breast-Cancer+	66.78	70.28	71.68	75.52	5	5	7
Cleveland-HD+	74.92	81.19	81.19	83.17	7	5	33
Credit-Rating+	80.72	86.23	85.36	87.68	5	7	13
German-Credit	72.29	70.70	72.40	73.09	5	7	17
Glass	70.19	65.42	71.49	70.19	9	9	1
Heart-Statlog+	75.20	79.26	80.0	84.07	5	5	33
Hepatitis+	81.29	84.52	85.16	85.16	7	9	7
Horse-Colic+	67.93	70.38	73.64	70.38	7	7	7
Ionosphere	86.61	85.75	86.6	86.61	5	3	1
Iris	95.33	96.0	96.0	97.33	5	7	15
Pima-Diabetes+	71.21	74.48	74.61	75.52	9	7	17
Primary-Tumor+	35.69	43.07	39.23	43.07	15	11	29
Sonar	86.54	64.9	71.15	86.54	5	5	1
Soybean	90.92	91.65	92.24	90.92	3	3	1
Vehicle	70.06	70.68	72.58	70.06	9	9	1
Vote	92.18	92.18	92.18	93.56	1	1	5
Vowel	99.39	99.39	99.39	99.39	1	1	1
Wine	96.07	94.38	94.94	97.75	3	3	31
Wisconsin-BC	95.56	96.28	96.56	96.85	3	3	17
Zoo	97.03	97.03	97.03	97.03	1	1	1
Average	80.26	80.35	81.92	83.17	5	5	11

q are removed from *reducedSet* ($\Theta(n \cdot m)$) and then this new set is ordered according to q ($\Theta(n \cdot \log n)$). We have also carried out *i-NN*$_{wv}$ which uses *k-NN*$_{wv}$ against of *k-NN*. In this latter variant, *k-NN*$_{wv}$ is evaluated from $k = 1$ to k_{max} and the vector of frequencies is increased every time for that label whose assigned weight is the greatest. This weight is the sum of $1/d$ for all the neighbors with the same label, where d is the distance from a neighbor to q.

3 Empirical Evaluation and Discussion

To carry out and to test the method, the Euclidean and the Overlap distance were used to measure continuous and nominal attributes, respectively. The continuous attributes values were normalized in the interval [0,1]. Examples with missing-class were removed and missing attributes values were treated with the mean or mode, respectively. *i-NN* and *i-NN*$_{wv}$ were tested by 10-folds cross-validation with 24 databases from the UCI repository [3]. Both introduced methods were compared with *k-NN* using 25 different *k* values (the odd numbers in the interval [1, 51]). This limit was fixed after observing how the accuracy decreased from a value near the best k for all databases (being $k = 33$ the maximum value for two databases, *Heart-Cleveland* and *Primary-Tumor*).

In Table 1 is reported the rates obtained; the *k-NN* algorithm is included for comparison using two k values: the best k for each database (Column *k-NN*$_{best}$) and the best average-value ($k = 1$) for all databases (Column *1-NN*). The domains signed with + mean an improvement of *i-NN*$_{wv}$ regarding *1-NN* by means of t-Student statical test with $\alpha = 0.05$. It is also showed the k_{max} value used for each database and the *best k value* found for *k-NN*. We can observe

that *i-NN*$_{wv}$ obtained better accuracy than *1-NN* for 15 databases. Ten of these domains were classified with a significant improvement (about 5%). Observing Column 6 it seems correct to consider that *i-NN*$_{wv}$ can be a suitable choice when predicting the best k value for *k-NN* is not an easy and computationally fast task. That is, *i-NN* improves the rates given by *k-NN* when the best k is a high value. In addition, the average classification accuracy is kept for the most databases.

4 Conclusion and Future Work

An easy variation of the *k-NN* algorithm has been introduced and evaluated in this paper. Through *i-NN*, a new query is classified several times by *k-NN* so that the assigned label is the most frequent in all the evaluations. We have also carried out *i-NN*$_{wv}$ which evaluates *k-NN*$_{wv}$ several times against of *k-NN*. In this variant, the assigned label is that with the greatest global weight. Experiments with UCI databases show that most real domains present overlapped regions in which predicting the best k value for the NN algorithm can not be a possible task. With this approach, errors obtained when either noise is present in data or a new query is near decision boundaries can be smoothed. But classification based on geometric proximity with local adaptation is still an open problem. For further study, we are trying to predict the k values through data transformation, using prototypes and feature construction. We also studying another approach based on the *nearest enemy* instead of the nearest neighbor.

References

[1] D. W. Aha, D. Kibler, and M. K. Albert. Instance-based learning algorithms. *Machine Learning*, 6:37–66, 1991.

[2] S. Arya, D. M. Mount, N. S. Netanyahu, R. Silverman, and A. Wu. An optimal algorithm for nearest neighbor searching. In *Proceedings of 5th ACM SIAM Symposium on discrete Algorithms*, pages 573–582, 1994.

[3] C. Blake and E. K. Merz. Uci repository of machine learning databases, 1998.

[4] S. Cost and S. Salzberg. A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning*, 10:57–78, 1993.

[5] T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, IT-13(1):21–27, 1967.

[6] S.A. Dudani. The distance-weighted k-nearest-neighbor rule. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6, 4:325–327, 1975.

[7] J. S. Aguilar F. J. Ferrer and J. C. Riquelme. Nonparametric nearest neighbor with local adaptation. In *Proceedings of the* 10^{th} *Portuguese Conference on Artificial Intelligence*, Porto, Portugal, December 2001.

[8] R. C. Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11:63–91, 1993.

[9] D. Heath S. Salzberg, A. Delcher and S Kasif. Best-case results for nearest neighbor learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(6):599–610, 1995.

[10] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society B*, 36:111–147, 1974.

[11] C. Wettschereck. *A Study of Distance-Based Machine Learning Algorithms*. PhD thesis, Oregon State University, 1995.

[12] D. R. Wilson and T. R. Martinez. Improved heterogeneous distance functions. *Journal of Artificiall Intelligence Research*, 6(1):1–34, 1997.