Iterated Probabilistic Weighted $k$-Nearest Neighbor Algorithm*

J. M. Martínez-Otzeta and B. Sierra

Dept. of Computer Science and Artificial Intelligence
University of the Basque Country
P.O. Box 649. E-20080 San Sebastián. Spain
e-mail: {ccchmaotj, ccpsiarb}@si.ehu.es
Website: http://www.sc.ehu.es/ccwrobot

Abstract. We report on the application of a new version of the $k$-NN algorithm named Iterated Probabilistic Weighted $k$-NN algorithm (IPW-$k$-NN). This algorithm classifies new cases based on the probability distribution each case has to belong to each class. These probabilities are computed according to the $k$ Nearest Neighbors, as a way of measure the typicalness of a given case with regard to every class.

Keywords. Supervised Classification, Nearest Neighbor, $k$-NN, Machine Learning, Pattern Recognition

1 Introduction

The $k$-NN classification method [2] assigns to an unclassified case the class resulting from a voting of the $k$ nearest cases among a set of previously classified cases.

We present a new voting method which takes into account the fact that not all the cases in the database are typical representatives of the class they belong to (i.e., they could have some degree of exceptionality). This new method gives to each point in the training database a measure of its typicality regarding its neighbors and their typicality as well.

2 A new approach: the Iterated Probabilistic Weighted $k$ Nearest Neighbor Method (IPW-$k$-NN)

Our method is a new version of the Probabilistic Weighted $k$ Nearest Neighbor Method (Sierra and Lazkano [5]). The authors make use of Bayesian Networks to estimate the probabilities, while we take into account the nearest neighbors to do the same task. The bigger the probability a neighbor has to belong to class $\theta$, the bigger will be the contribution this neighbor will make to case’s probability.

* This work was supported by the University of the Basque Country.
of belonging to that class \( \theta \). Several iterations are made so farther neighbors could influence this probability computation. The number of iterations is given by a parameter, namely \( \beta \). Value of previous computation is taken into account, being the total result a linear combination of the old value and the new one. An \( \alpha \) parameter drives the program about how much counts the relative weight of these two components in the new probability value. The new proposed method, IPW-\( k \)-NN is shown in its algorithmic form in Figure 1.

```
begin IPW-k-CNN
    As input we have the samples file TR, containing \( n \) cases \((x_i, \theta_i), i = 1, ..., n\), the value of \( k, \alpha \) and \( \beta \) and a new case \((y, \theta)\) to be classified
    \( \theta \) ranges over \( m \) classes
FOR each case \((x_i, \theta_i)\) in TR DO
    BEGIN
        Search the \( k \) Nearest Neighbors of \((x_i, \theta_i)\) in TR \( \cdot \) \((x_i, \theta_i)\) and store their TR index in \( \text{Neigh}_{i,j}, j = 1, ..., k \)
        Initialize the probability array \( P_{ij} \) associated to the case \( i \) as follows:
        if \( \theta_i = j \) then \( P_{ij} = 1 \) otherwise \( P_{ij} = 0 \)
    END
FOR \( \beta \) iterations DO
    BEGIN
        FOR each case \((x_i, \theta_i)\) in TR DO
            BEGIN
                Modify the associated probability array \( P_{ij} \) as follows:
                \( P'_{ij} = \left( \sum_{j=1}^{k} P_{\text{Neigh}_{i,j}} \right)/k; P_{ij} = \alpha P'_{ij} + (1 - \alpha) P_{ij} \)
            END
        END
    BEGIN
        Search the \( k \) Nearest Neighbors of \((y, \theta)\) in TR
        Reset the weights of all existing classes \( WC_i = 0 \)
    FOR each of the \( k \)-NN \((x_k, \theta_k)\) DO
        BEGIN
            FOR each class \( i \) actualize its weight \( WC_i \) as follows:
            \( WC_i = WC_i + P_{ki} \)
        END
        Output the class \( \theta_i \) with greatest weight \( WC_i \)
    end IPW-k-CNN
```

Fig. 1. The pseudo-code of the Iterated Probabilistic Weighted \( k \)-Nearest Neighbor Algorithm.

3 Experimental Results

3.1 Datasets

Ten databases have been used to test our algorithm. All but one have been obtained from the **UCI Machine Learning Repository** [1]. **Nettalk** database was
obtained from MLC++ main repository [4]. All databases have been randomly split ten times into two sets: a training set and a testing set. Size and other characteristics of these sets are given in Table 1.

<table>
<thead>
<tr>
<th>Domain</th>
<th>Training cases</th>
<th>Test cases</th>
<th>Num. of classes</th>
<th>Num. of attributes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diabetes</td>
<td>691</td>
<td>77</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Heart</td>
<td>243</td>
<td>27</td>
<td>2</td>
<td>13</td>
</tr>
<tr>
<td>Pima</td>
<td>691</td>
<td>77</td>
<td>2</td>
<td>8</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>315</td>
<td>36</td>
<td>2</td>
<td>34</td>
</tr>
<tr>
<td>Monk2</td>
<td>383</td>
<td>44</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>Wine</td>
<td>160</td>
<td>18</td>
<td>3</td>
<td>13</td>
</tr>
<tr>
<td>ZOO</td>
<td>94</td>
<td>10</td>
<td>7</td>
<td>16</td>
</tr>
<tr>
<td>Letters</td>
<td>18,600</td>
<td>2,000</td>
<td>26</td>
<td>16</td>
</tr>
<tr>
<td>Nettalk</td>
<td>7,229</td>
<td>7,242</td>
<td>324</td>
<td>203</td>
</tr>
<tr>
<td>Shuttle</td>
<td>43,500</td>
<td>14,500</td>
<td>7</td>
<td>9</td>
</tr>
</tbody>
</table>

In order to give a real perspective of applied methods, we use 10-Fold Cross-validation in all experiments [6]. Parameter $\beta$ (number of iterations) ranges from 1 to 10 and parameter $\alpha$ (relative weights of the two terms in expression 1) from 0 to 1, in steps of width 0.01.

3.2 $k$-NN vs. IPW-$k$-NN

We have tested our new method against the classical $k$-NN. Tables showing IPW-$k$-NN performance against $k$-NN, over the ten databases are given.

According to the data shown in Table 2, the new proposed approach IPW-$k$-NN algorithm outperforms the result obtained by the standard $k$-NN in three of the used databases (Diabetes, Heart and Pima), while in seven of them obtains the same accuracy for several values of $\alpha$ and $\beta$.

<table>
<thead>
<tr>
<th>Database</th>
<th>$k$-NN Best</th>
<th>$k$-NN Mean</th>
<th>IPW-$k$-NN Best</th>
<th>IPW-$k$-NN Mean</th>
<th>$\alpha$</th>
<th>$\beta$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Diabetes</td>
<td>75.71±0.35</td>
<td>75.71±0.24</td>
<td>75.61±0.34</td>
<td>75.61±0.34</td>
<td>0.65</td>
<td>0.57</td>
</tr>
<tr>
<td>Heart</td>
<td>68.5±10.95</td>
<td>71.48±10.48</td>
<td>70.64±10.34</td>
<td>70.64±10.34</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Pima</td>
<td>75.11±0.604</td>
<td>75.38±0.604</td>
<td>75.38±0.604</td>
<td>75.38±0.604</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>86.62±0.57</td>
<td>86.62±0.57</td>
<td>86.62±0.57</td>
<td>86.62±0.57</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Monk2</td>
<td>66.38±0.86</td>
<td>66.65±0.86</td>
<td>66.65±0.86</td>
<td>66.65±0.86</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Wine</td>
<td>75.95±0.86</td>
<td>75.95±0.86</td>
<td>75.95±0.86</td>
<td>75.95±0.86</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>ZOO</td>
<td>98.69±0.04</td>
<td>98.69±0.04</td>
<td>98.69±0.04</td>
<td>98.69±0.04</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Letter</td>
<td>95.3±0.11</td>
<td>95.3±0.11</td>
<td>95.3±0.11</td>
<td>95.3±0.11</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Nettalk</td>
<td>95.1±2.97</td>
<td>98.5±2.97</td>
<td>98.5±2.97</td>
<td>98.5±2.97</td>
<td>1.00</td>
<td>0.98</td>
</tr>
<tr>
<td>Shuttle</td>
<td>95.85±0.02</td>
<td>95.85±0.02</td>
<td>95.85±0.02</td>
<td>95.85±0.02</td>
<td>1.00</td>
<td>0.98</td>
</tr>
</tbody>
</table>

In table 2 it is shown the best result obtained by $k$-NN, along with the corresponding $k$ value and the mean of $k$-NN performance in the range [1,10].
In the right half of the table it is shown the best performance of IPW-\(k\)-NN, along with corresponding values of \(k\), \(\alpha\) and \(\beta\). Mean(1) is the mean of IPW-\(k\)-NN performance over \(k\) in the range [1,10], for the given values of \(\alpha\) and \(\beta\). Mean(2) is the mean of IPW-\(k\)-NN performance over a interval of eleven values of \(\alpha\), centered in the given value. Only interval values in the range [0.00,1.00] are tested.

4 Conclusion and Further Results

In this work we have developed and tested a new distance based algorithm: Iterated Probabilistic Weighted \(k\) Nearest Neighbor (IPW-\(k\)-NN). We have shown that improvements over classic \(k\)-NN are reached for some values of \(\alpha\) and \(\beta\) and for some databases.

Not surprisingly, when the best result of \(k\)-NN holds for \(k = 1\), no improvement is achieved, given that the classification of a case relies in just a neighbor instead of a voting among several ones. So it is necessary to change the class of that only neighbor to change the voting result. When a voting among more than a neighbor is needed, our algorithm behaves better.

Further research involves a characterization of \(\alpha\) and \(\beta\) values for which improvement holds, as well as another distance apart from Euclidean, as Mahalanobis, which takes into account dependences among variables; we are also thinking about making experiments with this new approach in order to perform Feature Subset Selection [3] that will allow it to outperform the results obtained by using all the predictors variables.

References


