Not Being at Odds with a Class: A New Way of Exploiting Neighbors for Classification

Myriam Bounhas¹ and Henri Prade² and Gilles Richard³

Abstract. Classification can be viewed as a matter of associating a new item with the class where it is the least at odds w.r.t. the other elements. A recently proposed oddness index applied to pairs or triples (rather than larger subsets of elements in a class), when summed up over all such subsets, provides an accurate estimate of a global oddness of an item w.r.t. a class. Rather than considering all pairs in a class, one can only deal with pairs containing one of the nearest neighbors of the item in the target class. Taking a step further, we choose the second element in the pair as another nearest neighbor in the class. The oddness w.r.t. a class computed on the basis of pairs made of two nearest neighbors leads to low complexity classifiers, still competitive in terms of accuracy w.r.t. classical approaches.

1 Introduction
Several classification methods rely on the idea that a new item should be classified in the class w.r.t. which it appears to be the least at odds. Logical proportions [7] are Boolean expressions that link four Boolean variables through equivalences between similarity or dissimilarity indicators pertaining to pairs of these variables. Among logical proportions, the heterogeneous ones [7] provide a natural basis to build a global oddness measure of an item w.r.t. a multiset, by cumulating the oddness index w.r.t. different triples of elements in the multiset for the different features [3]. This has suggested the simple procedure of classifying a new item x into the class C which minimizes this global oddness measure. Moreover, it has been noticed that the oddness measure of an item w.r.t. a triple can be generalized to a subset of any size, as well as to numerical features [3]. We investigate here the use of oddness measures based on selected pairs, since our aim is to show that we can keep on with the same idea of oddness while lightening the computational cost. Doing so, we preserve the accuracy while further reducing the complexity by constraining the set of considered pairs. The idea is to constrain the choice of the two elements in the pairs. We study the options of using pairs including one nearest neighbour, and then two nearest neighbors [4].

2 Oddness of an item w.r.t. a multiset of values
In order to build an oddness index for a Boolean x w.r.t. a multiset of Boolean values \{a_i | i \in [1,n]\}, we look for a formula \(F(a_1,\ldots,a_n, x)\) holding iff \(a_1 = \ldots = a_n = 0\) and \(x = 1\), or if \(a_1 = \ldots = a_n = 1\) and \(x = 0\). The oddness \(\text{odd}(\{a_i | i \in [1,n]\},x)\) of a Boolean x w.r.t. a multiset of Boolean values \(a_i | i \in [1,n]\) can be summarized as:

\[\neg(\bigvee_{i\in[1,n]} a_i \equiv x) \land \neg(\bigwedge_{i\in[1,n]} a_i \equiv x)\] (1)

It is clear that odd holds true only when the value of x is seen as being at odds among the other values: x is the intruder in the multiset of values. In the case \(n = 2\), \(\text{odd}(a_1, a_2, x)\) is 0 if and only if the value of x is among the majority value in the multiset \(a_1, a_2\). When \(n = 3\), \(\text{odd}(a_1, a_2, a_3, x)\) does not hold true only in the situations where there is a majority among values in \(a_1, a_2, a_3\), and x belongs to this majority (e.g. \(\text{odd}(0, 1, 0), 0 = 0\), or when there is no majority at all (e.g. \(\text{odd}(0, 1, 1), 0 = 0\)).

Extension to real values is quite straightforward. Assuming that numerical features are renormalized between 0 and 1, we use the standard definitions of the logical connectives in the [0,1]-valued Łukasiewicz logic [8]. Then, a graded counterpart to formula (1) is:

\[\min(|x - \max(a_1,\ldots,a_n)|,|x - \min(a_1,\ldots,a_n)|)\] (2)

A natural extension \(\text{odd}(\{\overline{a_i} | i \in [1,n]\}, \overline{x})\) to vectors with \(m\) features is to consider the sum componentwise of the odd values computed via expression (1) or (2) as:

\[\sum_{i=1}^{m} \text{odd}(\{\overline{a_i} | i \in [1,n]\}, \overline{x^i})\] (3)

where \(x^j\) is the j-th component of \(\overline{x}\) and the \(a_i's\) are the j-th components of the vectors \(\overline{a_i}\) respectively. If \(\text{odd}(\{\overline{a_i} | i \in [1,n]\}, \overline{x}) = 0\), it means that no feature indicates that \(\overline{x}\) behaves as an intruder among the \(a_i's\). On the contrary, high values of \(\text{odd}(\{\overline{a_i} | i \in [1,n]\}, \overline{x})\) means that, on many features, \(\overline{x}\) appears as an intruder.

3 Global oddness measure for classification
Given a set \(C = \{\overline{a_i} | i \in [1,n]\}\) of vectors gathering examples of the same class, one might think of computing \(\text{odd}(C, \overline{x})\) as a way of evaluating how much \(\overline{x}\) is at odds w.r.t. C. An immediate classification algorithm would be to compute \(\text{odd}(C, \overline{x})\) for every class and to allocate to \(\overline{x}\) the class which minimizes this number. Nevertheless, this number is not really meaningful when the size of C is large. Indeed, we have to be careful because then \(\{\overline{a_i} | i \in [1,n]\}\) is summarized by two vectors made respectively by the minimum and the maximum of the feature values among the examples of C (due to expressions (2) and (3)). These two vectors have high chance to be fictitious in the sense that, usually, they are not elements of C. Approximating our knowledge of the set C using only the maximal ranges of the feature values over the members of the set seems very crude. An idea is then to consider small subsets S of the class C, then compute \(\text{odd}(S, \overline{x})\) and finally add all these atomic oddness indice to get a global measure of oddness of \(\overline{x}\) w.r.t. C. This approach leads to the following initial formula: \(\sum_{S \subseteq C, |S|=n} \text{odd}(S, \overline{x})\). To take into account the relative size of the different classes, it is fair to introduce a normalization factor and our final definition is:

\[\text{Odd}_n(C, \overline{x}) = \frac{1}{\binom{n}{k}} \sum_{S \subseteq C, |S|=n} \text{odd}(S, \overline{x})\]
A classification algorithm follows, where we allocate to a new item \( x \) the class \( C \) minimizing \( Odd_d(C, \bar{x}) \) for a given \( n \), thus defining a family of classifiers also denoted \( Odd_d \).

**Algorithm 1** Oddness-based algorithm

**Input:** a training set \( T \) of examples \((C, cl(T))\)

a new item \( \bar{x} \),

an integer \( n \),

Partition \( T \) into sets \( C \) of examples with the same label \( c \).

for each \( C \) do

Compute \( Odd_d(C, \bar{x}) \) for subsets of size \( n \).  

end for

\[ cl(\bar{x}) = \arg \min_{c} Odd_d(C, \bar{x}) \]

return \( cl(\bar{x}) \)

4 Experiments

The experimental study is based on 15 datasets taken from the U.C.I. machine learning repository [6], applying standard 10 folds cross-validation technique. On these datasets, we compare classifiers \( Odd_1, Odd_2, Odd_3 \) accuracy to the one of state-of-the-art classifiers C4.5, SVMs, JRip and IBk (shown in Table 1). Our first experiments

<table>
<thead>
<tr>
<th>Datasets</th>
<th>C4.5</th>
<th>SVM</th>
<th>JRip</th>
<th>IBk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Balance</td>
<td>0.78</td>
<td>0.87</td>
<td>0.89</td>
<td>0.90</td>
</tr>
<tr>
<td>Car</td>
<td>0.95</td>
<td>0.91</td>
<td>0.87</td>
<td>0.91</td>
</tr>
<tr>
<td>Monk1</td>
<td>0.99</td>
<td>0.75</td>
<td>0.10</td>
<td>0.98</td>
</tr>
<tr>
<td>Monk2</td>
<td>0.95</td>
<td>0.67</td>
<td>0.73</td>
<td>0.64</td>
</tr>
<tr>
<td>Monk3</td>
<td>0.90</td>
<td>0.10</td>
<td>0.10</td>
<td>0.99</td>
</tr>
<tr>
<td>Spect</td>
<td>0.81</td>
<td>0.81</td>
<td>0.83</td>
<td>0.80</td>
</tr>
<tr>
<td>Voting</td>
<td>0.96</td>
<td>0.96</td>
<td>0.95</td>
<td>0.93</td>
</tr>
</tbody>
</table>

lead to rather poor performances due to the huge number of subsets considered in each class, giving them equal importance, while a lot of them blur the accuracy of oddness measure through the summation.

In our first experiments, considering the average accuracy on all datasets, \( Odd_2 \) is a better performer than \( Odd_1 \) and \( Odd_3 \). We then focus on \( Odd_2 \), trying to privilege subsets including elements of particular interest such as nearest neighbors in the classes. Since selecting one element of a pair as a nearest neighbor of the item \( x \) in the target class leads to good accuracy rates, we also consider the option of taking the second element as another nearest neighbor in the class, with the normalization factor chosen accordingly as \( \frac{1}{2} \), leading to \( Odd_2(NN, NN) \). This is also beneficial from a complexity viewpoint.

In Tables 2 and 3, we provide classification results respectively for \( Odd_2(NN, Std) \) (using only one nearest neighbour) and \( Odd_2(NN, NN) \) (using two nearest neighbours) for different values of \( k \) (\( k \) being the number of nearest neighbours used).

In the last column of Table 2, we assign respectively a positive ‘+’, negative ‘-’ or neutral ‘.’ mark if the \( Odd_2(NN, NN) \) is respectively better, worse or similar to \( Odd_2(NN, Std) \) for \( k = 15 \). This comparison shows that the two classifiers have similar efficiency for most datasets, except for Monk2 where \( Odd_2(NN, NN) \) outperforms not only \( Odd_2(NN, Std) \), but also SVM and IBk.

We can also note that \( Odd_2(NN, NN) \) performs more or less in the same way as the best known algorithms. Especially we compared this classifier to IBk for \( k = 15 \) using the Wilcoxon Matched-Pairs Signed-Ranks [5]. We get a p-value of 0.024 which shows that \( Odd_2(NN, NN) \) is significantly better than IBk.

5 Conclusion

In this paper, we suggest a new way to evaluate the oddness of an item w.r.t. a class. Since using subsets of pairs (Odd_2) provides better accuracy results than singletons (Odd_1) or triples (Odd_3), we further investigate this option by filtering candidate pairs. We first choose one item in a pair as a nearest neighbour, then two elements in a pair as nearest neighbours in the class. Experiments show that we are still competitive with state of the art classifiers (k-NN, SVM) while having drastically decreased complexity. These results suggest that it may be beneficial to consider pairs of nearest neighbours and then to minimize an oddness measure, which departs from the k-NN view dealing with nearest neighbours in isolation.

REFERENCES