ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.

Instance and Feature Weighted k-Nearest-Neighbours Algorithm

Gabriel Prat and Lluís A. Belanche

Computer Science Department - Universitat Politècnica de Catalunya
Omega Building, Jordi Girona 1-3, 08034, Barcelona - Spain

Abstract. We present a novel method that aims at providing a more stable selection of feature subsets when variations in the training process occur. This is accomplished by using an instance-weighting process –assigning different importances to instances– as a preprocessing step to a feature weighting method that is independent of the learner, and then making good use of both sets of computed weigths in a standard Nearest-Neighbours classifier. We report extensive experimentation in well-known benchmarking datasets as well as some challenging microarray gene expression problems. Our results show increases in stability for most subset sizes and most problems, without compromising prediction accuracy.

1 Introduction

The *feature subset selection* (FSS) problem has been studied for many years by the statistical as well as the machine learning communities. However, the *stability* of the FSS process has been relatively neglected in the literature until very recently –see e.g. [1, 2]. Previous research aimed at quantifying stability, rather than enhancing it, leading to the development of stability measures [3]; few works address the explicit *improvement* of such stability, notably [2].

In previous work, we studied methods aimed at providing a more stable selection of feature subsets when variations in the training process occur [4], in a way that is independent of the learner or the specific FSS algorithm. We argue here that it is possible that the classification ability of different features varies across the feature space: for some subset of the data we should use a certain set of features, while for some other subset another set of features results in a better classification accuracy; conversely, the instances may contribute differently to the importance of features. Our objective is therefore to foster the study of possible synergies between both tasks to ultimately develop workable learning algorithms. In this paper we present a method that combines the weighting of instances with the feature weighting process into a more effective doubly-weighted Nearest-Neighbours classifier. We report performance in a series of experiments, using well-known benchmarking datasets and some challenging microarray gene expression problems. Our results show improvements in FSS stability for most subset sizes and problems, without compromising prediction accuracy.

2 Preliminaries

Let $D = \{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$ be a training data set of length N, each instance $\mathbf{x}_n \in \mathbb{R}^d$ with its corresponding class label t_n. The *margin* of an instance with

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

respect to a hypothesis (a classification rule, in this case) measures the confidence of the classifier when making its prediction [5]. In particular, the *hypothesis margin* of $\mathbf{x}$ is the distance between the hypothesis and the closest hypothesis that assigns an alternative label to $\mathbf{x}$. For 1-Nearest-Neighbours, the hypothesis margin of an instance $\mathbf{x}$ to a set of data points D is given by [6]:

$$\theta_D(\mathbf{x}) = \frac{1}{2}\Big(\|\mathbf{x} - m(\mathbf{x})\| - \|\mathbf{x} - h(\mathbf{x})\| \Big) \tag{1}$$

where $m(\mathbf{x})$ and $h(\mathbf{x})$ are the **near hit** and **near miss** of $\mathbf{x}$: those instances in D nearest to $\mathbf{x}$ with the same and with a different class label, respectively. RELIEF is a filter algorithm that uses the hypothesis-margin concept in eq. (1) to assess the importance of each feature in a dataset D as the accumulated influence that each feature has in computing the margin of every instance in D [7]. In particular, RELIEVEDF is a deterministic feature ranking algorithm that depends on a user-defined parameter l. The algorithm picks one instance at a time and computes the hypothesis margin of each feature independently, accumulating the feature-wise distances to the l nearest hits and l nearest misses.

SIMBA is a more recent feature weighing algorithm that assigns weights to features based on their contributions to the hypothesis margins of the instances [5]. Since better generalization is expected if instances have larger margins, one should favour features that *contribute* more to these margins. Instances $\mathbf{x}$ achieving highly positive $\theta_D(\mathbf{x})$ present good modeling behavior (being far from misses and close to hits), while those with highly negative $\theta_D(\mathbf{x})$ become outlying ones (surrounded by misses and far from hits). The presence or absence of these latter instances in a training sub-sample is therefore a source of unstability.

In the Margin-based Instance Weighting (MBIW) method, an instance $\mathbf{x} \in \mathbb{R}^d$ can be mapped to $\mathbf{x}'$ according to $x'_j = |x_j - m(\mathbf{x})_j| - |x_j - h(\mathbf{x})_j|$ [8]. The larger the value of x'_j, the more feature j contributes to the margin of instance $\mathbf{x}$; thus $\mathbf{x}'$ captures the local profile of feature relevance. To compute an overall relevance for $\mathbf{x}$, the average over all margin vectors is taken, very much as RELIEF does; then the weight of an instance $\mathbf{x}$ is given by:

$$\omega(\mathbf{x}) = \frac{1/\overline{dist}(\mathbf{x}')}{\sum_{i=1}^{N} 1/\overline{dist}(\mathbf{x}'_i)}, \qquad \text{where } \overline{dist}(\mathbf{x}') = \frac{1}{N-1} \sum_{i=1, \mathbf{x}'_i \neq \mathbf{x}'}^{N-1} \|\mathbf{x}' - \mathbf{x}'_i\| \tag{2}$$

3 Combining Instance and Feature Weighting

An important problem with the hypothesis-margin concept defined in eq. (1) is the presence of noise. By this we mean every aspect in the data that is specific of the particular training sample (i.e., it is not a *regularity* of the problem). This may affect both instances (outliers), or features (redundant or irrelevant), and will certainly mislead the margin calculus of an instance. The proposed method extends SIMBA to incorporate the instance weights obtained with the MBIW method into the feature weights, to influence the way SIMBA behaves.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

In this paper, the MBIW method is executed first and the weights are handed over to SIMBA. However, our framework is quite flexible and one could also consider the other way around. We tested two different versions:

Normal: unmodified SIMBA algorithm (all instances drawn randomly)

Sample: base instance selection on the probability distribution given by the learned instance weights

Order: sort instances by decreasing weight, and base the iteration order directly on the resulting order (no randomness)

We call the methods SIMBAMBIW: SIMBA with Margin Based Instance Weighting (pseudo-code is shown in **Algorithm 1**). Note the use of the **weighted norm** of a vector $\mathbf{z}$ as $\|\mathbf{z}\|_{\mathbf{w}} = \sqrt{\sum_{i=1}^{d} w_i^2 z_i^2}$. Using this combined strategy, features can be ranked according to their importance (using the $\mathbf{w}$ weights), and at the same time favouring stability due to the ω weights.

Algorithm 1: SIMBAMBIW (D)

Compute instance weights ω using eq. (2)
$\mathbf{w} \leftarrow (1, 1, \ldots, 1)$; `// Initialize feature weights`
for $n \leftarrow 1$ **to** N **do**
  if *strategy is* ***order*** **then**
    let $\mathbf{x}$ be the instance ranked in position n according to ω
  else if *strategy is* ***sample*** **then**
    draw an instance $\mathbf{x}$ from D, according to the distribution $\omega / \|\omega\|_1$
  else
    let $\mathbf{x}$ be the nth instance of a random permutation of D
  end
  calculate $m(\mathbf{x})$ and $h(\mathbf{x})$ with respect to $D \setminus \{\mathbf{x}\}$ using $\|\cdot\|_{\mathbf{w}}$
  for $i \leftarrow 1$ **to** d **do**
    $\Delta_i \leftarrow \frac{1}{2}\left(\frac{(x_i - (m(\mathbf{x}))_i)^2}{\|\mathbf{x} - m(\mathbf{x})\|_{\mathbf{w}}} - \frac{(x_i - (h(\mathbf{x}))_i)^2}{\|\mathbf{x} - h(\mathbf{x})\|_{\mathbf{w}}}\right) w_i$
  end
  $\mathbf{w} \leftarrow \mathbf{w} + \omega(\mathbf{x})\boldsymbol{\Delta}$
end
$\mathbf{w} \leftarrow \mathbf{w}^2 / \left\|\mathbf{w}^2\right\|_\infty$ where $(\mathbf{w}^2)_i := (w_i)^2$

4 Experimental Work

This section provides empirical evaluation of the proposed method. We test it to verify its real applicability in three groups of problems: a selection of 15 datasets from the UCI machine learning repository, the five problems used in the FSS

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

UCI datasets			
problem	d	C	N
Diabetes	8	2	768
Glass	10	6	214
Heart	13	2	20
Ionosphere	34	2	351
Landsat	36	6	6,435
LSVT Voice	309	2	126
Mammogram	65	2	86
Musk	168	2	6,598
Parkinsons	23	2	197
Pop Failures	18	2	540
SpectF	44	2	267
Sonar	60	2	208
Vehicle	18	4	946
Waveform	21	3	5,000
Wdbc	10	2	699

Microarray datasets			
problem	d	C	N
Breast cancer	24,481	2	97
Colon tumour	2,000	2	62
GCM	16,063	14	190
Leukemia	7,129	2	72
Lung cancer	12,533	2	181
Prostate cancer	12,600	2	136

NIPS Challenge datasets			
problem	d	C	N
Arcene	10,000	2	200
Dexter	20,000	2	600
Dorothea	100,000	2	1,150
Gisette	5,000	2	7,000
Madelon	500	2	2,600

Table 1: Dataset descriptions: d, C, N are the number of features, classes and instances, respectively.

challenge organized during the NIPS'2003 conference and six widely-used cancer microarray data –Table 1. The stability of an algorithm in selecting a subset of k features out of the initial full feature size d over a batch of M runs can be evaluated using the *Kuncheva index* (KI), defined as in [1]:

$$\mathrm{KI}\left(\mathcal{E}(k)\right) = \frac{2}{M(M-1)} \sum_{i=1}^{M-1} \sum_{j=i+1}^{M} \frac{|S_i(k) \cap S_j(k)| - (k^2/d)}{k - (k^2/d)}$$

where $S_i(k)$ is the subset of selected features of length k in the i-th run, and $\mathcal{E} = \{S_1, S_2, ..., S_M\}$ is the set containing all the retrieved feature subsets. KI values are bounded in $[-1, 1]$, with 1 corresponding to the maximum stability. The experimental setup consists of the two nested cross-validation loops: for every fold and repetition of the outer cross-validation loop, two feature-weighting processes are conducted with the same instances: one with the original SIMBA algorithm and one with our modified version taking instance weights into account. The KI is computed for every subset length at every partition loop and then averaged over the 10 times. Once the features have been obtained we test the obtained feature weights using a modified k-NN classifier that accepts both instance and feature weights, recording prediction accuracy on the leftout test parts. We use these weights to perform an **inner** 5x2-fold cross-validation with the purpose of estimating the prediction error of each classifier. This error is then computed for each fold to compare the feature sets selected by SIMBAMBIW.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

The modified k-NN classifier –shown in **Algorithm 2**– uses the feature weights to influence the distance calculation between two instances. Instead of using a majority voting as the original k-NN does to compute the label of the test instance, it uses the instance weights to give more relevant instances more influence in the voting –line 9 in the algorithm. By using an algorithm that accepts feature weights we overcome the need of finding a suitable feature set given the resulting weights of the process, as we did in our previous paper [4]. If we wanted to use the traditional version of k-NN at this point, we would have to decide a size s for the selected feature set, order the features according to their weights and keep the first s, or else use a classifier to perform a costly search in wrapper mode.

Algorithm 2: Instance and Feature Weighted k-Nearest Neighbours

Input : Training set $D = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, constant k, instance weights ω, feature weights $\mathbf{w}$, new instance $\mathbf{x}^*$ to be classified

Output: Class prediction for $\mathbf{x}^*$

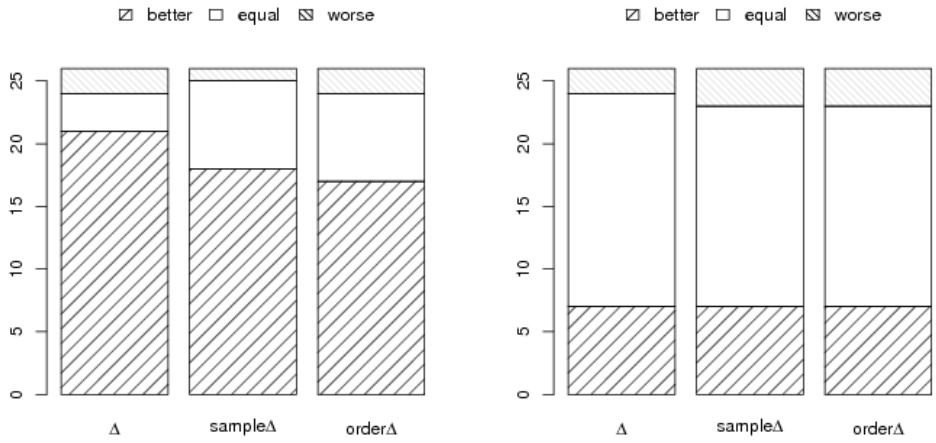
1 Initialize all $c_i \in \mathcal{C}$ to 0 ; `// C is the set of class labels`
2 **foreach** $\mathbf{x}_n \in D$ **do**
3 $\quad d_n \leftarrow \|\mathbf{x}_n - \mathbf{x}^*\|_{\mathbf{w}}$
4 **end**
5 Sort $\mathbf{d}$ in descending order
6 $D_k \leftarrow$ nearest k instances according to $\mathbf{d}$
7 **foreach** $x_n \in D_k$ **do**
8 $\quad$ let k be the class of $\mathbf{x}_n$
9 $\quad c_k \leftarrow c_k + \omega_n$
10 **end**
11 **return** $\arg\max_i c_i$

In Fig. 1 we see the number of problems (including UCI, NIPS and microarray) for which the modified versions of the FSS algorithm had better/equal/worse stability results, and the number of problems which the classification error of the resulting feature sets was better/equal/worse. We see that both modifications lead to more (or equally) stable results most of the time. In fact, SimbaMIW is only significantly less stable than standard Simba in one single case (the NIPS Madelon dataset using the 'sample' version). Very importantly, predictive errors are similar to those of more unstable versions.

5 Conclusions

The present work has introduced SimbaMBIW, a new method for improving the stability of feature subset selection algorithms, which draws upon previous algorithmic work on feature weighting and hypothesis margins for instances. Our strategy uses a double set of weights, one for the features and another one for the

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8.
Available from http://www.i6doc.com/en/.



(a) Stability **(b)** Classification error

Fig. 1: Number of problems where SimbaMiw was better/equal/worse than standard Simba regarding stability and classification error.

instances. Its suitability has been assessed using data from three different environments: microarray gene expression data, real-world and synthetic datasets. The present work offers a number of interesting avenues for further research. We are interested in quantifying and improving *prediction* stability: the ability of a classifier in labelling each instance coherently (independently of its correctness); there are also alternative ways to combine the weights: specifically, the instance weights can be updated at each iteration, given that the feature weights are re-computed, which would lead to a synergetic process.

References

[1] L I. Kuncheva. A stability index for feature selection. In *IASTED International Conference on Artificial Intelligence and Applications*, pp. 390–395, 2007.

[2] Y. Saeys, T. Abeel, Y. Peer. Robust feature selection using ensemble feature selection techniques. In *ECML-PKDD*, pages 313–325. Springer-Verlag, 2008.

[3] P. Somol, J. Novovičová. Evaluating stability and comparing output of feature selectors that optimize feature subset cardinality. *IEEE Trans. on PAMI*, 32(11):1921–39, 2010.

[4] G. Prat, and Ll. Belanche. Improved stability of feature selection by combining instance and feature weighting. In M. Bramer and M. Petridis (eds), *Research and Development in Intelligent Systems XXXI*, pp. 35–49. Springer, 2014.

[5] R.G. Bachrach, A, Navot, N. Tishby. Margin based feature selection - theory and algorithms. In *Intl. Conf. on Machine Learning (ICML)*, pages 43–50, 2004.

[6] K. Crammer, R.G. Bachrach, A. Navot, N. Tishby. Margin Analysis of the LVQ Algorithm. In *Advances in NIPS 2002*, pages 462–469, 2002.

[7] K. Kira, L. Rendell. The feature selection problem: Traditional methods and a new algorithm. pp. 129–134, Cambridge, USA, 1992

[8] Y. Han, L. Yu. A Variance Reduction Framework for Stable Feature Selection. In *ICDM*, pp. 206–215, 2010