Expert Systems with Applications 38 (2011) 10018–10024

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

General framework for class-specific feature selection

Bárbara B. Pineda-Bautista, J.A. Carrasco-Ochoa, J. Fco. Martínez-Trinidad *

Computer Science Department, National Institute of Astrophysics, Optics and Electronics, Luis Enrique Erro No. 1 Sta María Tonanzintla, Puebla, CP 72840, Mexico

ARTICLE INFO

Keywords:
Class-specific feature selection
Feature selection
Supervised classification
Classifier ensemble

ABSTRACT

Commonly, when a feature selection algorithm is applied, a single feature subset is selected for all the classes, but this subset could be inadequate for some classes. Class-specific feature selection allows selecting a possible different feature subset for each class. However, all the class-specific feature selection algorithms have been proposed for a particular classifier, which reduce their applicability. In this paper, a general framework for using any traditional feature selector for doing class-specific feature selection, which allows using any classifier, is proposed. Experimental results and a comparison against traditional feature selectors showing the suitability of the proposed framework are included.

1. Introduction

Feature selection is a commonly used process in pattern recognition and machine learning, wherein a subset of the features available from the data is selected for applying a classifier or learning algorithm. In this way, the selected subset contains features that most contribute to accuracy; and the remaining, unimportant features, are discarded.

Since, the class labels are given in supervised classification problems, it is natural that we want to keep only such features that are related to or lead to these classes and that allow to build a good classifier. Therefore, determining which features must be selected for describing instances in a supervised classification problem is very important, because an accurate classification of new instances depends on a good feature set. In practice, the most relevant features for a supervised classification problem are not known a priori. For this reason, large amounts of features are usually incorporated. Unfortunately, some features can be redundant or irrelevant, which could affect the classification accuracy (Dash & Liu, 1997; John, Kohavi, & Pfleger, 1994).

Feature selection has been widely studied (Almuallim & Dietterich, 1991; Fukunaga, 1990; Guyon, 2003; Kittler, 1978; Kohavi & John, 1997) and a great amount of algorithms (Al-Ani, 2009; Polat & Güneş, 2009; Sivagaminathan & Ramakrishnan, 2007; Wei-Chou, Shian-Shyong, & Tzung-Pei, 2008) for solving this problem has been proposed.

Feature selection has been widely used for eliminating redundant or irrelevant features, and it can be done in two ways: feature selection for all classes (traditional) and class-specific feature selection. In traditional feature selection, a single feature subset is selected for discriminating among all the classes in a supervised classification problem. In this context, there are two kinds of algorithms, wrappers and filters. Wrappers use a classifier, as a black box, to score feature subsets according to their accuracy. On the other hand, filters select feature subsets as a pre-processing step, independently of the chosen classifier.

In contrast, class-specific feature selection algorithms select a subset of features (possibly different) for each class. In this kind of algorithms different approaches have been proposed (Fu & Wang, 2002, 2005; Nanni, 2006; Oh, Lee, & Suen, 1999) which are strongly related to the use of a particular classifier.

In this work, a general framework for class-specific feature selection, which can use any traditional feature selector for selecting a possible different subset for each class of a supervised classification problem, is proposed. Our framework proposes to use the one-against-all class binarization method for transforming a c-class problem into c binary problems, one for each class, where the instances of a class are used as positive examples, and all other instances as negatives. For doing class-specific feature selection, traditional feature selectors are applied over these binary problems, thus, the feature subset selected for each binary problem is assigned to the class from which this problem was constructed. In order to classify new instances our framework proposes to use a classifier ensemble, where, for each class, a classifier is trained using the whole training set, but using the feature subset assigned to the class. Finally, our framework applies an ad hoc decision rule for combining classifier outputs. Preliminary results of our work were published in Pineda-Bautista, Carrasco-Ochoa, and Martínez-Trinidad (2009), where we presented some ideas of how to use traditional feature selection algorithms for solving class specific feature selection and as a first result, we tested these ideas using filter algorithms.

This paper is organized as follows: Section 2 describes related work; in Section 3, the proposed framework for class-specific

* Corresponding author. Tel.: +52 222 266 31 00x8325; fax: +52 222 266 31 52.
E-mail addresses: barpin@inaoep.mx (B.B. Pineda-Bautista), ariel@inaoep.mx (J.A. Carrasco-Ochoa), fmartine@inaoep.mx (J. Fco. Martínez-Trinidad).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.02.016

feature selection is presented; the experimental results are shown and commented in Section 4; finally, in Section 5, our conclusions and some directions for future work, are exposed.

2. Related work

Most of the algorithms in the literature about feature selection solve the traditional feature selection (Guyon, 2003; Liu & Motoda, 2008), i.e., these algorithms find a single feature subset for discriminating among all the classes in a supervised classification problem. Another important approach for solving feature selection problems is the class-specific feature selection. In this approach, in order to discriminate a class from the remaining classes, the algorithms find a subset of features (possibly different) for each class. In this section, we describe some works following this approach.

In Baggenstoss (1999) the author proposed using class-specific features, but for feature extraction. Some other works (Baggenstoss, 2004; Baggenstoss & Beierholm, 2004; Baggenstoss & Niemann, 2000) followed this work. Although these works are focused in feature extraction, they are the basis for works dealing with the class-specific feature selection problem.

In Oh et al. (1999), different features are selected for each class. In this work, for each class, the separation capacity of each individual feature is evaluated, and all features are sorted in descending order according to this capacity, then $d/2$ features are selected, where d is the number of original features. Since a different set of features is selected for each class, a modular neural network classifier is proposed, which consists of c sub-networks, one for each class of the c-class problem. Each sub-network produces only one output, and a new instance is assigned to the class corresponding to the maximum output value. Since, in Oh et al. (1999), the same number of features is selected for each class, some irrelevant features could be included in some classes, and in some other classes, some relevant features could be excluded.

In Fu and Wang (2002, 2005) a different feature subset is selected for each class, and a novel RBF (Radial Based Function) classifier is proposed. These works are based on RBF neural networks, which have a set of hidden units, each one used for identifying one class, therefore a subset of hidden units can be used for discriminating a class from the others. For identifying a feature subset for each class, a genetic algorithm (GA) determines a feature mask for the hidden unit subset associated to each class. Even though the number of selected features could be different for each class, the feature selection process is very expensive and it depends on the proposed classification rule. Additionally, in the experiments, only three databases (Glass, Thyroid and Wine) from the UCI repository (Merz & Murphy, 1998) were tested.

In Nanni (2006), a feature selection method for 2-class problems was introduced. In this method, each class is divided in clusters, and then a subset of features is selected for each cluster. The same number of features is selected for all the clusters in the same class, but this number can be different for each class. The number of clusters and the number of features to be selected for each class are parameters that must be provided by the user. For selecting features for a cluster, features are ranked according to a separability measure based on the scalar Mahalanobis distance, and the features corresponding to the highest values of this measure are retained. For classification, a classifier is trained for each cluster, using only the feature subset selected for that cluster. Each classifier measures the similarity between a new instance and the instances of the cluster for which it was trained. A new instance is assigned to the class of the cluster where the maximum similarity is reached. Since the number of clusters and the number of features to be selected for each cluster are unknown parameters, a big amount of experiments must be done in order to find good values for these parameters.

For improving classification accuracy, classifier ensembles have been used. These ensembles follow the idea of selecting different feature subsets for the same supervised classification problem. In Skurichina and Duin (2005) and Silva and Fred (2007) this kind of ensembles are presented, but although a different feature subset is selected for each classifier, this subset is used for all the classes.

3. General framework for class-specific feature selection

In the previous section, we can appreciate that all the class-specific feature selection algorithms are strongly related to the use of a particular classifier. However, it is desirable that we could apply class-specific feature selection independently of the classifier used in the classification stage, as occurs when we apply a filter algorithm; or using any classifier that guide the selection, as occurs when we apply a wrapper algorithm.

Therefore, in order to allow the use of any traditional feature selector for class-specific feature selection, and using any classifier (the one desired by the user) the class-specific feature selection framework proposed in this work consists of four stages: class binarization, class balancing, class-specific feature selection, and classification (see Fig. 1). The classification stage is not a part of the selection process, but for taking advantage of class-specific feature selection, it is necessary to define a new classification stage, since conventional classifiers only allow using a single feature subset.

3.1. Class binarization

In class-specific feature selection, the goal is to select a feature subset that allows discriminating a class from the remaining classes. Therefore, in the first stage of our framework, we propose to uses the one-against-all class binarization (Fürnkranz, 2002) in order to transform a c-class problem into c binary problems. For each class w_i, $i = 1,\ldots,c$; a binary problem $\langle w_i, \Omega_i \rangle$ where $\Omega_i = \bigcup_{\substack{j=1 \\ j \neq i}}^{c} w_j$, is created, i.e., for each binary problem the instances of the class w_i are used as positive examples, and the instances of all other classes are used as negative examples (see Fig. 2).

3.2. Class balancing process

It is well known that when the one-against-all class binarization technique is used, the generated binary problems could be imbalanced (Japkowicz & Stephen, 2002). In order to avoid that this imbalance affects the selection, we propose to balance the classes applying an oversampling method, before applying a conventional feature selector on a binary problem. In the literature, there are several methods for oversampling. Some of the most used are random oversampling (Hulse, Khoshgoftaar, & Napolitano, 2007), SMOTE (Chawla, Bowyer, Hall, & Kegelmeyer, 2002), and Borderline-SMOTE (Han, Wang, & Mao, 2005). However, after testing these methods, oversampling by repeating training instances (see Fig. 2) got the best results. Therefore, we propose to use this oversampling method after the class binarization step.

For oversampling by repeating training instances, for each class w_i, $i = 1,\ldots,c$; $\beta_i = |\Omega_i| - |w_i|$ is computed, where $|w_i|$ is the number of instances in class w_i; and $|\Omega_i|$ is the number of instances in the remaining classes. If $\beta_i > 0$ the classes will be balanced by repeating instances in the class w_i until the number of instances in w_i and Ω_i are the same.

3.3. Class-specific feature selection

After binarization, we have c different binary problems. For each binary problem, features are selected using a traditional feature selector, and the selected features are assigned to the class from

Fig. 1. General framework stages for class-specific feature selection.

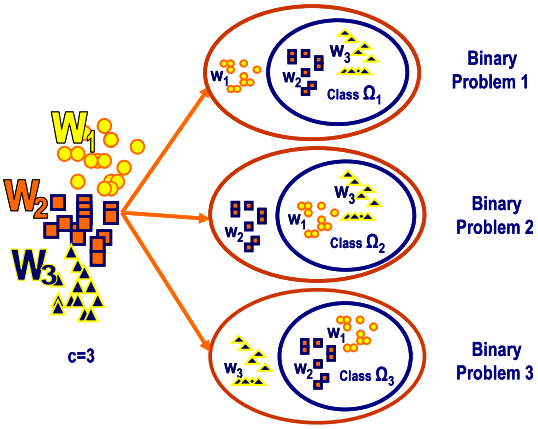

Fig. 2. Example of one-against-all class binarization for a 3-class supervised classification problem.

which the binary problem was constructed. In this way, c possible different feature subsets are obtained, one for each class of the original c-class supervised classification problem. In this stage, it is possible to use a different traditional feature selector for each binary problem, however in our experiments we used the same selector for all the classes.

At the end of this stage (see Fig. 3), the feature selection (class-specific feature selection) has properly finished. However, as for each class a possible different subset of features has been selected, it is very important to define how to use these subsets for classification. In the next stage, this point is addressed.

3.4. Classification

Since for doing class-specific selection, we have transformed a multiclass problem in a set of binary problems. At first glance, the straightforward way for using each subset of features associated to each class is to follow a multi-classification scheme training a classifier for each binary problem and integrating the decisions of the classifiers. However, following this approach we would be solving a problem different from the one originally formulated. It is important to highlight that the set of features associated to a class, in theory, is the best subset found that allows discriminating the objects of this class from the objects in the other classes. Therefore in the classification stage of our framework, for each class w_i, a classifier e_i is trained for the original multi-class problem (i.e., the instances in the training set for the classifier e_i maintain their original class) but taking into account only the selected features for the class w_i. In this way, we will have a classifier ensemble $E = \{e_1, \ldots e_c\}$. When a new instance O is classified through the ensemble, its original dimensionality d must be reduced to the dimensionality d_i used by the classifier e_i $i = 1,2,\ldots,c$; due to each classifier will assign a class to O the following decision rule is applied:

1. If a classifier e_i gives as output the class w_i, i.e., the same class for which the features (used for training e_i) were selected; then the class w_i is assigned to O. If there is a tie (two or more classifiers give as output w_i), the class of O is assigned through majority vote among all classifiers. If the tie continues then the class of O will be the majority class among the tied classes.
2. If no classifier gives as output the class for which the features (used for training e_i) were selected; the class of O is assigned through majority vote. If there is a tie then the class of O will be the majority class among the tied classes.

In Fig. 4, the classification process using class-specific feature selection is depicted.

4. Experimental results

In order to show the effectiveness of the proposed general framework for class-specific feature selection, some experiments over 15 databases from the UCI dataset repository (Merz & Murphy, 1998), see Table 1, with different number of instances, features, and classes, were done.

Since the classification step proposed for taking advantage of class-specific feature selection allows to use any classifier for the ensemble, we evaluated the performance of feature subsets, selected by our framework, using the following classifiers: Naive Bayes, k nearest neighbors (kNN) using $k = 1$ and $k = 3$, C4.5, and multilayer perceptron (MLP).

In our experiments, the following feature selection alternatives were compared:

1. Class-specific feature selection (proposed method).
2. Feature selection for all the classes (traditional feature selection).
3. Using all the features (do not apply feature selection).

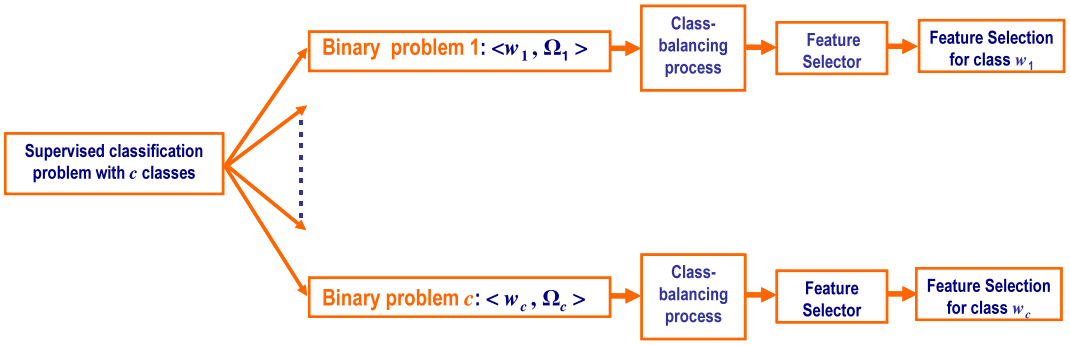

Fig. 3. The result of the third stage of our framework is the class-specific feature selection.

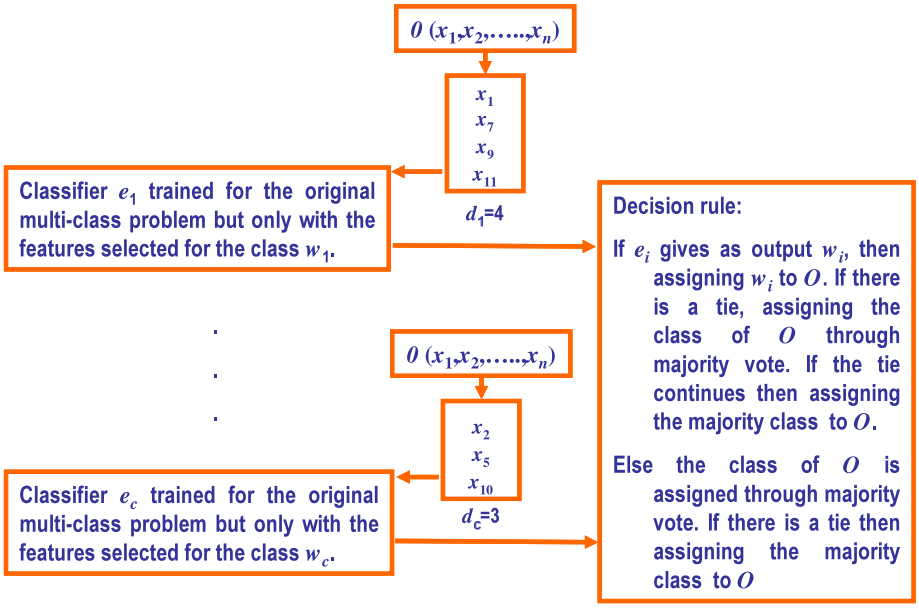

Fig. 4. Classification process using class-specific feature selection.

Table 1
Databases used in the experiments.

Databases	Classes	Objects	Features
Annealing	5	798	38
Bridges	7	106	11
Dermatology	6	366	32
E. coli	8	336	8
Echocardiogram	3	132	11
Glass	6	214	9
Iris	3	150	4
Nursery	5	12,960	8
Optdigits	10	3823	64
Page-blocks	5	5473	10
Postoperative	3	90	8
Segment	7	210	19
Thyroid gland	3	215	5
House-votes-84	2	435	16
Wine	3	178	13

For alternatives 2 and 3, the classifiers were directly used since it is equivalent to apply the classification stage of our method with the same feature subset for all the classes. For all the experiments, we used 10 folds cross validation, reporting the average classification error.

Given that our method can use any traditional feature selection method as base for doing class-specific feature selection, we made experiments using both filter and wrapper methods. In the following sections, we show the experiments for each one of these approaches.

4.1. Experiments using filter methods

For testing the proposed framework for class-specific feature selection, using filter feature selectors, we use four methods implemented in WEKA (Witten & Frank, 2005). These methods consist of an evaluator, to measure the quality of a subset of features, and a search method. For our experiments we used as evaluators: CfsSubsetEval (e_1) and ConsistencySubsetEval (e_2); and as search methods: BestFirst (b_1) and GeneticSearch (b_2). Thus, all the combinations of these evaluators with these search methods, in total four traditional filter feature selectors (see Table 2), were used.

Table 2
Filter feature selectors used for testing the proposed framework.

Feature selector	Evaluator	Search method
e1b1	CfsSubsetEval	BestFirst
e1b2	CfsSubsetEval	GeneticSearch
e2b1	ConsistencySubsetEval	BestFirst
e2b2	ConsistencySubsetEval	GeneticSearch

In Fig. 5, the average classification errors, over all the databases, for the different classifiers, following the three feature selection alternatives, and using the filter feature selectors, are reported. From this figure, we can see that in all cases the average classification error reached using the feature subsets obtained by our framework was smaller than the one obtained applying traditional feature selection. In addition, most of the cases, the average classification error reached using the feature subsets obtained by our framework was also smaller than using all the features (do not apply feature selection).

Finally, in the Table 3, we show, for all the classifiers, the average classification error of the four feature selectors (see Table 2) applying: our method, traditional feature selection and without feature selection. In this table, we can see in bold letter that, for all the tested classifiers, the proposed method always got the best results (the smallest classification error rates).

4.2. Experiments using wrapper methods

In this experiment, we use four wrapper feature selectors, implemented in WEKA (Witten & Frank, 2005). These methods consist of an evaluator, to measure the quality of a subset of features, and a search method. For our experiments we used as evaluators: ClassifierSubsetEval (e_3) and WrapperSubsetEval (e_4); and as search methods: BestFirst (b_1) and GeneticSearch (b_2). Thus, all the combinations of these evaluators with these search methods, in total four traditional wrapper feature selectors (see Table 4), were used.

In Fig. 6, the average classification errors, over all the databases, for the different classifiers, following the three feature selection alternatives, and using the wrapper feature selectors, are reported.

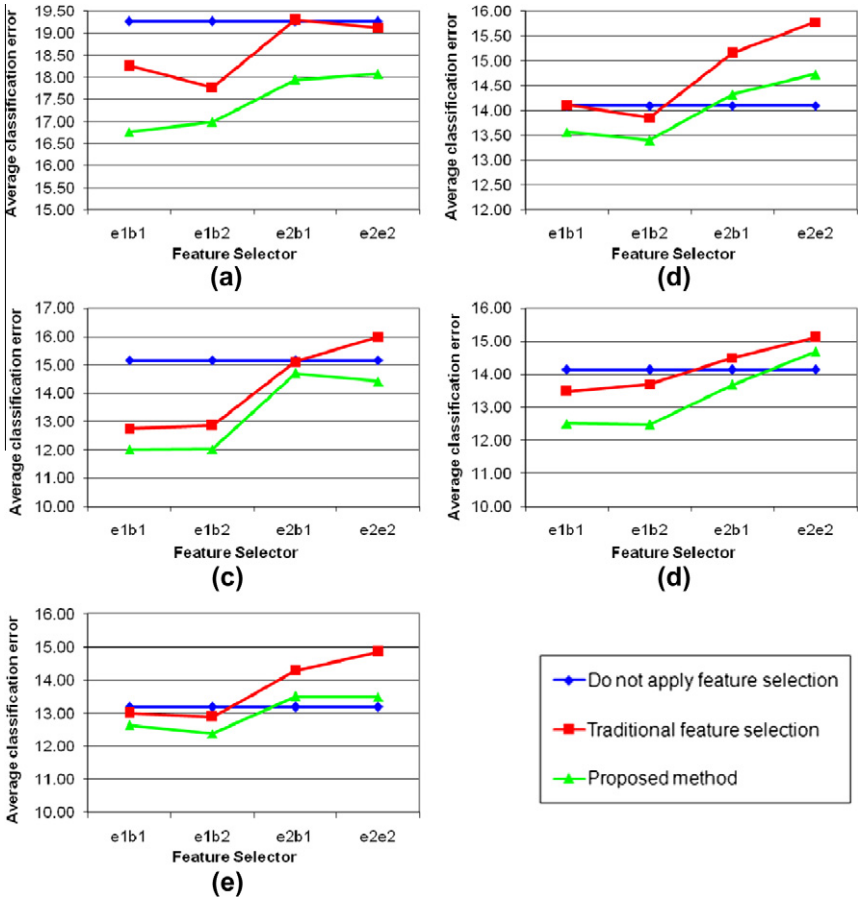

Fig. 5. Average classification error for (a) Naive Bayes, (b) C4.5, (c) *k*NN with $k = 1$, (d) *k*NN with $k = 3$, and (e) MLP; following the three feature selection alternatives, using filter selectors.

Table 3
Average classification error for all the classifiers using the three different filter feature selection alternatives.

Feature selection variant	Naive Bayes	C4.5	*k*NN $k = 1$	*k*NN $k = 3$	MLP
Do not apply feature selection	19.27	14.09	15.14	14.15	13.19
Traditional feature selection	18.61	14.72	14.18	14.20	13.76
Proposed framework	**17.44**	**14.00**	**13.29**	**13.34**	**13.01**

From this figure, we can see that, in most cases the average classification error reached using the feature subsets obtained by our framework was smaller than the one obtained applying traditional feature selection and using all the features (do not apply feature selection).

Finally, in the Table 5, we show, for all the classifiers, the average classification error of the four feature selectors (see Table 2) applying: our method, traditional feature selection and without feature selection. In this table, the smallest classification error rates appear in bold letter; we can see that, the feature subsets obtained by the proposed framework got better results (smaller classification error rates) for most of the cases (except for C4.5) than those obtained using traditional feature selection. Nevertheless, when wrapper methods were used, using all the available features (do not apply feature selection) was the best alternative for *k*NN using $k = 1$ and MLP.

Summarizing, from the average results shown in Tables 3 and 5, we can see that when a filter approach was used, the feature subsets obtained by the proposed framework got better results than those obtained applying traditional feature selection, for all the classifiers. In addition, when a wrapper approach was used, the feature subsets obtained by the proposed framework got better results than those obtained applying traditional feature selection in 4 from 5 classifiers.

Table 4
Wrapper feature selectors used for testing the proposed framework.

Feature selector	Evaluator	Search method
e3b1	ClassifierSubsetEval	BestFirst
e3b2	ClassifierSubsetEval	GeneticSearch
e4b1	WrapperSubsetEval	BestFirst
e4b2	WrapperSubsetEval	GeneticSearch

5. Conclusions

In this paper, a general framework for class-specific feature selection was proposed. For taking advantage of class-specific feature selection, a classification scheme based on classifier ensembles, as well as a novel decision rule for classifying new instances were also proposed.

One of the main characteristics of the proposed framework is that it allows using any traditional feature selector for choosing

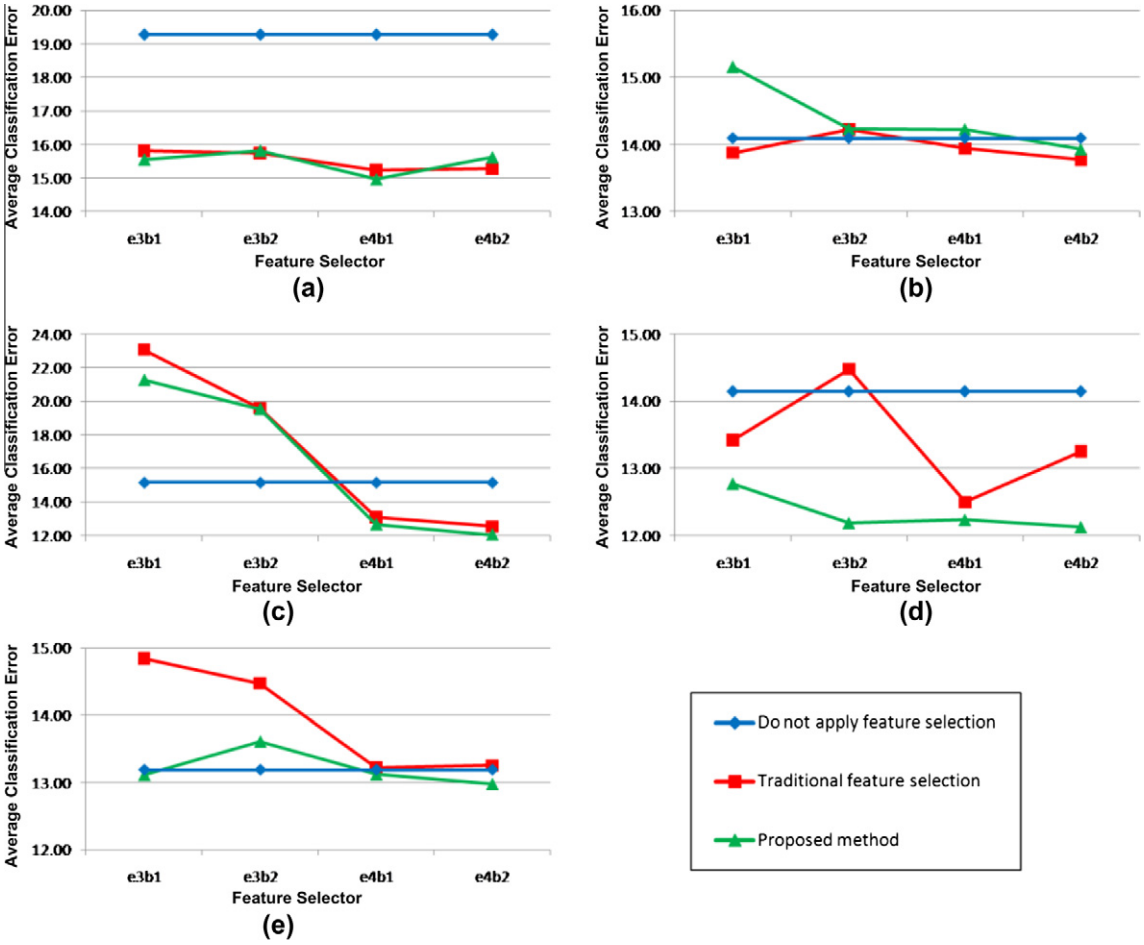

Fig. 6. Average classification error for (a) Naive Bayes, (b) C4.5, (c) *k*NN with $k = 1$, (d) *k*NN with $k = 3$, and (e) MLP; following the three feature selection alternatives, using wrapper selectors.

Table 5
Average classification error for all the classifiers using the three different wrapper feature selection alternatives.

Feature selection variant	Naive Bayes	C4.5	*k*NN $k = 1$	*k*NN $k = 3$	MLP
Do not apply feature selection	19.27	14.09	**15.14**	14.15	**13.19**
Traditional feature selection	15.52	**13.95**	17.07	13.41	13.95
Proposed framework	**15.49**	14.39	16.37	**12.33**	13.21

the features that better describe or characterize each specific class in a supervised classification problem. Additionally, due to conventional classifiers, usually work using the same set of features for all the classes and our method produces a possible different subset of features for each class; the proposed framework includes a classification phase for taking advantage of class-specific feature selection. This classification step allows using any classifier through an ensemble; in contrast to other class-specific selectors, which are designed for using a particular classifier.

Based on our experimental results, we can conclude that, usually, applying traditional feature selection allows getting better results than using all the available features. However, in most of the cases, applying class-specific feature selection, using the proposed framework, allows getting better results than applying traditional feature selection. The experiments also show that the proposed framework is suitable to apply both filter and wrapper methods for solving the problem of class-specific feature selection. Additionally, unlike previous works, our framework allows to do class-specific feature selection for any classifier.

As future work, we are going to look for other classification schemes, which would allow reaching better classification results when using class-specific feature selection.

References

Al-Ani, A. (2009). A dependency-based search strategy for feature selection. *Expert Systems with Applications, 36*(10), 12392–12398.

Almuallim, H., & Dietterich, T. G. (1991). Learning with many irrelevant features. In *Proceedings of the ninth national conference on artificial intelligence* (pp. 547–552).

Baggenstoss, P. M. (1999). Class-specific feature sets in classification. *IEEE Transactions on Signal Processing, 47*(12), 3428–3432.

Baggenstoss, P. M. (2004). Class-specific classifier: Avoiding the curse of dimensionality. *IEEE Aerospace and Electronic Systems Magazine, 19*, 37–52.

Baggenstoss, P. M., & Beierholm, T. (2004). Speech music discrimination using class-specific features. In *Proceedings of the 17th international conference on pattern recognition* (pp. 379–382).

Baggenstoss, P. M., & Niemann, H. (2000). A theoretically optimal probabilistic classifier using class-specific features. In *Proceedings of international conference on pattern recognition* (pp. 767–772).

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321–357.

Dash, M., & Liu, H. (1997). Feature selection for classification. *Intelligent Data Analysis, 1*, 131–156.

Fu, X., & Wang, L. (2002). A GA-based novel RBF classifier with class dependent features. In *Proceedings of the congress on evolutionary computation* (pp. 1890–1894).

Fukunaga, K. (1990). *Statistical pattern recognition* (2nd ed.). San Diego, CA: Academic Press.

Fürnkranz, J. (2002). Round robin classification. *Journal of Machine Learning Research, 2*, 721–747.

Fu, X., & Wang, L. (2005). *Data mining with computational intelligence*. Springer.

Guyon, I. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157–1182.

Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning. In *Proceedings of the international conference on intelligent computing* (pp. 878–887).

Hulse, J. V., Khoshgoftaar, T. M., & Napolitano, A. (2007). Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on machine learning* (pp. 935–942).

Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis, 6*(5), 429–450.

John, G. H., Kohavi, R., & Pfleger, K. (1994). Irrelevant features and the subset selection problem. In *Proceedings of the eleventh international conference on machine learning* (pp. 21–129).

Kittler, J. (1978). Feature set search algorithms. *Pattern Recognition and Signal Processing*, 41–60.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97*, 273–324.

Liu, H., & Motoda, H. (2008). *Computational methods of feature selection*. Chapman & Hall/CRC.

Merz, C. J., & Murphy, P. M. (1998). UCI repository of machine learning databases. University of California at Irvine, Department of Information and Computer Science. Available from: <http://ftp.ics.uci.edu/pub/machine-learning-databases/>.

Nanni, L. (2006). Cluster-based pattern discrimination: A novel technique for feature selection. *Pattern Recognition Letters., 27*, 682–687.

Oh, I. S., Lee, J. S., & Suen, C. Y. (1999). Analysis of class separation and combination of class-dependent features for handwriting recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 21*, 1089–1094.

Pineda-Bautista, B., Carrasco-Ochoa, J. A., & Martínez-Trinidad, J. F. (2009). Taking advantage of class-specific feature Selection. In *Proceedings of the international intelligent data engineering and automated learning conference* (pp. 1–8).

Polat, K., & Güneş, S. (2009). A new feature selection method on classification of medical datasets: Kernel F-score feature selection. *Expert Systems with Applications, 36*(7), 10367–10373.

Silva, H., & Fred, A.L.N. (2007). Feature subspace ensembles: A parallel classifier combination scheme using feature selection. In *Proceedings of the international workshop on multiple classifier systems* (pp. 261–270).

Sivagaminathan, R. K., & Ramakrishnan, S. (2007). A hybrid approach for feature subset selection using neural networks and ant colony optimization. *Expert Systems with Applications, 33*(1), 49–60.

Skurichina, M., & Duin, R. P. W. (2005). Combining feature subsets in feature selection. In *Proceedings of the international workshop on multiple classifier systems* (pp. 165–175).

Wei-Chou, C., Shian-Shyong, T., & Tzung-Pei, H. (2008). An efficient bit-based feature selection method. *Expert Systems with Applications, 34*(4), 2858–2869.

Witten, I. H., & Frank, E. (2005). *Data mining: Practical machine learning tools and techniques* (2nd ed.). San Francisco: Morgan Kaufmann.