

A comparison of filters and wrappers methods for feature selection methods in supervised classification

Edgar Acuña¹

Department of Mathematics, University of Puerto Rico at Mayaguez, Mayaguez, PR 00680, Puerto Rico

Abstract

In this paper we carry out an empirical comparison of the performance of filter and wrapper procedures for feature selection in supervised classification. The filters considered are the RELIEF, Las Vegas Filter, and FINCO a new procedure that is being introduced here. Among the wrappers we considered sequential forward selection, sequential backward selection and the sequential floating forward selection. The classifier used for the wrappers is one based on kernel density estimation. Both types of procedures are compared according to their percentages of features selected and their effect in the misclassification error rate of a kernel density estimate classifier. The comparison is carried out in twelve datasets coming from the Machine Learning Database Repository at the University of California, Irvine.

Keywords. Feature selection, Wrappers, Filters, Supervised classification, Kernel density classifiers

1 Introduction

The dimensionality problem in machine learning and/or statistical modeling still remains as a very important problem to be investigated in spite of the fact that computers are becoming more powerful everyday. The classification task is more conveniently done with few features for two reasons: A saving of computing time and an easy interpretation of the model. The goal of feature selection is to choose a small subset of features such that the recognition rate of the classifier does not decrease significantly. The feature selection

¹ *E-mail address: edgar@cs.uprm.edu*

methods depend on the way that the subsets are generated and on the evaluation function used to measure the quality of the subset under examination. The generation procedure can be of three types: complete, heuristic and random. The evaluation function can be one of these: a distance measure, an information measure, a dependence measure, a consistency measure, and the misclassification error rate. Dash and Liu (1997) established 15 categories of feature selection procedures, based on the generation procedure of the subsets and the evaluation function used to compare them. In this paper we compare the performance of six feature selection procedures: The RELIEF, Las Vegas Filter, FINCO, Sequential forward selection, Sequential backward selection and, Sequential floating forward selection. The first three are considered filter methods because they do not use a classifier to select the feature, whereas the last three require a classifier and they are named wrappers. The classifier used in this paper is one based on the product kernel density estimator of the class conditional density (Acuna, et al. , 2002). Our comparison is carried out in twelve datasets available in the Machine Learning Repository at the University of California, Irvine. The number of features of these datasets varies from 4 to 60. In section 2 we discuss the filter methods. Wrappers are discussed in section 3. The experimental methodology used is described in section 4 and, in section 5 we present the conclusions of our experimental study.

2 Filter methods

These methods do not require the use of a classifier to select the best subset of features. They use general characteristics of the data to evaluate features. In this paper we considered three filter methods: the RELIEF, Las Vegas Filter (LVF) and a new procedure introduced by us called FINCO. We will describe briefly each of them.

a) The RELIEF. This method was introduced for a two class problem by Kira and Rendell in 1992. The general idea of the method is to chose the features that distinguish most between classes. In a two class problem, initially all the p features of the dataset D have relevance weight W_j for $j = 1, \dots, p$, equal to zero. Then at each step of an iterative processs an instance \mathbf{x} is chosen randomly from D and the weights are updated according to the distance of \mathbf{x} to its *Nearmiss* and *NearHit*. The *Nearmiss* is the instance in the dataset D that is closest to \mathbf{x} but that belongs to the other class. The *NearHit* is the instance in D that is closest to \mathbf{x} and it is in the same class. The updating formula of W_j is given by

$$W_j = W_j - \text{diff}(x_j, \text{Nearhit}_j)^2 + \text{diff}(x_j, \text{Nearmiss}_j)^2$$

where x_j is the j -th component of \mathbf{x} , and the function *diff* computes the distance between the values of a feature for two given instances. For nominal and binary features, *diff* is either 1 (the values are different) or 0 (the values are equal). For ordinal and continuous features, *diff* is the difference of the values of the feature for the two instances normalized by the range of the feature. The process is repeated M times, where M is a predefined parameter, usually it is taken as the number of instances in D . At the end the best subset will include the features with relevance greater than a threshold fixed beforehand. The RELIEF algorithm appears in Fig. 1.

Input: D =Dataset, p = number of features in D , M : number of instances randomly drawn, Threshold= τ .

1. $T = \phi$, T is the subset containing the features selected

2. Initialize all weights, W_j ($j=1, \dots, p$), to zero

3. For $i=1$ to M

{ Choose at random an instance x in D .

Find its *NearHit* and *NearMiss*

For $j=1$ to p

$$W_j = W_j - \text{diff}(x_j, \text{nearHit}_j)^2 + \text{diff}(x_j, \text{nearMiss}_j)^2$$

}

4. For $j=1$ to p

If $W_j > \tau$ then append f_j to T

5. Output: T .

Fig 1. The RELIEF algorithm.

The RELIEF works well in datasets containing mixed type of features, that can be noisy and correlated too. Its time complexity is linear in the number of features as well as in the number of instances. However, the unclear choice of the threshold as well as of the parameter M is a disadvantage of the RELIEF. The method was extended to multiclass problems by Kononenko (1994) and Kononenko et al. (1997). The new algorithm was named RELIEF-F. In this

case one *Nearmiss* is found for every class distinct to the class containing \mathbf{x} and the distance from \mathbf{x} to each *Nearmiss* is weighted according to the proportion of instances in each class. The updating formula of the relevance weights is as follows:

$$W_j = W_j - \text{diff}(x_j, \text{Nearhit})^2 + \sum_{C \neq \text{class}(x_j)} \frac{P(C)}{1 - P(\text{class}(x_j))} \text{diff}(x_j, \text{Nearmiss}(C))^2$$

In this paper we have used both algorithms.

b) The LVF. It was introduced by Liu and Setiono (1996). LVF uses a random generation of subsets and an inconsistency measure as evaluation function. Two instances of a dataset D are inconsistent if they have the same feature values but different classes. The inconsistency measure of a given subset of features T relative to a dataset D is defined as

$$\text{Inconsistency}(T, D) = \frac{\sum_{i=1}^K |D_i| - |M_i|}{N} \quad (1)$$

where $|D_i|$ is the number of occurrences of the i -th feature value combination on T , K is the number of the distinct combinations of features values on T , $|M_i|$ is the cardinality of the class to which belong the majority of instances on the i -th feature value combination, and N is the number of instances in the dataset D . The algorithm requires an inconsistency threshold close or equal to zero, which is fixed beforehand. Any candidate subset having an inconsistency greater than the threshold is rejected. The random generation process stops when the maximum number of subsets to be generated is reached. This number is another prefixed parameter of the algorithm. The LVF method is suitable for datasets having only nominal variables, but can be applied to datasets with continuous features after a discretization process, such as the method of Fayyad and Irani based on partitioning by minimization of the class information entropy with Minimal Description Length Principle as an stopping criteria (Dougherty, et al. , 1995). In this paper we have applied a simple equal width intervals discretization method based on the Scott's formula (see Venables and Ripley (1997), p. 169)) to estimate the width interval. The LVF algorithm is shown in Fig. 2.

c) FINCO. In this paper we introduce a new feature selection method called FINCO (it stands by Forward and Inconsistency). FINCO uses a sequential forward generation of subsets along with the inconsistency measure used in the LVF method. The best subset of features T is initialized as the empty set and in each step we add to T the feature that gives the lowest inconsistency rate along with the features already included in T . The process continues until the inconsistency rate given by T and each of the features not yet selected is less than a predefined inconsistency threshold. A feature entering to T can not be removed from it. Continuous features need to be discretized before

applying FINCO. The FINCO algorithm appears in Fig. 3.

In order to avoid the nesting problem a floating forward selection may be applied, but we did not consider it in this work.

Input: D =Dataset, p = number of features in D , S = set of all the features in D , MaxTries = maximum number of trials, Threshold= δ .

$C_{best} = p, S_{best} = S$

For $i=1$ to MaxTries

{ S_i = subset of S randomly selected.

$C_i = \text{card}(S_i)$

If ($C_i < C_{best}$)

{ If $\text{Inconsistency}(S_i, D) \leq \delta$ then $\delta_{best} = \text{Inconsistency}(S_i, D)$

$S_{best} = S_i, C_{best} = C_i$

}

If ($C_i = C_{best}$)

{If $\text{Inconsistency}(S_i, D) < \delta_{best}$ then $S_{best} = S_i$ }

}

Output: S_{best} : The best subset of selected features.

Fig 2. The LVF algorithm.

3 Wrapper methods

These methods use the misclassification error rate of a given classifier as the evaluation function. A large discussion of wrappers can be found in Kohavi and John (1997). In this paper we have used a classifier based on kernel density estimation which is described later in this section. The misclassification error

rate is estimated by a 10-fold cross-validation technique. We have only considered wrapper methods where the subsets are generated heuristically. There are three main approaches: Sequential Forward selection (SFS), Sequential Backward selection (SBS), and the Sequential Floating Forward selection (SFFS).

Input: D=Dataset, p=number of features in D, S=set of all the features in D, Threshold= δ .

Initialization:

Let $k=0$ and $T_k = \phi$ (T_k : Subset of features selected until the k-th step).

Inclusion step:

For $k=1$ to p . Do

$$x^+ = \arg \min_{x \in S-T_k} Incons(T_k + x)$$

where, $S-T_k$:Subset of features not yet selected.

$Incons(T_k+x)$: Inconsistency level using the features in T_k along with feature x .

Thus, x^+ : is the most important feature with respect to T_k

If $Incons(T_k+x^+) \leq Incons(T_k)$ and $Incons(T_k+x^+) > \delta$ then let $T_{k+1}=T_k+x^+$ and $k:=k+1$. Go to the inclusion step.

Termination:

If either $Incons(T_{k+1}) > Incons(T_k)$ or $Incons(T_{k+1}) \leq \delta$

Output: T_k : Subset of selected features.

Fig 3. The FINCO algorithm.

In the SFS the best subset of features T is initialized as the empty set and in each step we add to T the feature that gives the highest correct classification rate (CCR) along with the features already included in T . The process continues until the CCR of the classifier built using the features in T and each of the features not yet selected does not increase. In the SBS the best

subset of features T is initialized as the set containing all the features and in each step we remove from T the feature that gives the highest CCR if it was excluded from T . Thus the worst feature with respect to T is removed. The process continues until the CCR excluding from T each of the features not yet removed decreases. The methods mentioned suffer of the nesting effect. This means that a feature that is included (removed) in some step of the iterative process can not be excluded (included) in a later step.

The Sequential floating forward selection (SFFS) method was introduced by Pudil et al. (1994) to deal with the nesting problem. The best subset of features T is initialized as the empty set and at each step first a new feature is added to T as in the SFS method but after that one searches for features that can be removed from T as in the SBS method until the CCR decreases. The iterations continue until none new variable can be added to T because the CCR does not increase. The SFFS algorithm is shown in Fig 4.

The kernel density estimator classifier. From a Bayesian point of view, an object with measurement vector \mathbf{x} is assigned to the class j^* such that $j^* = \operatorname{argmax}_{1 \leq j \leq J} \hat{f}(j/\mathbf{x})$. In this paper the posterior estimates are obtained indirectly via the class conditional density $f(\mathbf{x}/j)$ using Kernel density estimators. Formally for a given class j and a random sample $\mathbf{X}_1, \mathbf{X}_2, \dots, \mathbf{X}_n$ of the p -dimensional random vector \mathbf{X} with continuous components, the product kernel estimate of the class conditional density at the point \mathbf{x} is given by

$$\hat{f}(\mathbf{x}) = \frac{1}{nh_1h_2\dots h_p} \sum_{i=1}^n \prod_{v=1}^p K\left(\frac{x_v - \mathbf{X}_{iv}}{h_v}\right) \quad (2)$$

where the kernel K will be usually a symmetric unimodal density function, for instance the normal density, and h_v represents the bandwidth of the v -th predictor. In this paper we have used kernel density estimation with both fixed and adaptive bandwidth. The standard kernel density estimator uses a fixed bandwidth obtained by assuming that the class conditional density is a multivariate normal. That is, $h_v = s_v(4/n(p+2))^{1/(p+4)}$, ($v=1, \dots, p$) where p is the number of predictors, n is the number of instances of a given class, and s_v is the standard deviation of the v -th predictor. In the adaptive kernel, the bandwidth varies from one point to another, and its value will depend on the concentration level of data in the neighborhood of the point. The basic idea of this approach is to combine the standard kernel density estimation with the nearest neighbor approach. The adaptive kernel used is of the form

$$\hat{f}_{adapt}(\mathbf{x}) = \frac{1}{nh_1h_2\dots h_p} \sum_{i=1}^n \frac{1}{\lambda_i^p} \prod_{v=1}^p K\left(\frac{x_v - \mathbf{X}_{iv}}{\lambda_i h_v}\right) \quad (3)$$

where $\lambda_i = (\hat{f}(\mathbf{X}_i)/g)^{-\alpha}$. Here g is the geometric mean of $\hat{f}(\mathbf{X}_i)$, a positive

pilot estimation of $f(\mathbf{X}_i)$ and, $0 \leq \alpha \leq 1$. We have also considered kernel density estimators for categorical predictors: binary, nominal and ordinal. In the first case, we have followed the Aitchison and Aitken's proposal and in the last two cases the Titterington's proposal (see Acuña et al. (2002)).

Input: D=Dataset, p=number of features in D, S= set of all features in D.

Initialization:

T_2 = Subset of selected features after applying SFS twice. Set $k=2$.

Step 1: Inclusion. Find

$$x^+ = \arg \max_{x \in S - T_k} CCR(T_k + x)$$

Thus, x^+ is the most important feature with respect to T_k

Let $T_{k+1} = T_k + x^+$ and, $k := k+1$

Step 2: Conditional exclusion. Find

$$x^- = \arg \max_{x \in T_k} CCR(T_k - x)$$

Thus, x^- is the less important feature in T_k

If $CCR(T_k - x^-) > CCR(T_k)$ then let $T_{k-1} = T_k - x^-$ and $k := k-1$.

Go to step 2. Otherwise go to step 1.

Termination:

If $CCR(T_{k+1}) \leq CCR(T_k)$.

Output: T_k =Subset of selected features.

Fig 4. The sequential floating forward selection algorithm.

4 Experimental Methodology

The comparisons were carried out using 12 datasets coming from the Machine Learning Database Repository at UCI. In Table 1 we show a summary of the characteristics of the 12 datasets used in this paper. In the *Glass* dataset we

Table 1

Datasets used in this paper. The (*) mark indicates that some features in these datasets have not been considered in our experiment.

Dataset	Instances	Classes	Features			
			C	B	N	0
<i>Iris</i>	150	3	4	-	-	-
<i>Sonar</i>	208	2	60	-	-	-
<i>Glass*</i>	214	6	9	-	-	-
<i>Heartc</i>	303	2	5	3	3	2
<i>Bupa</i>	345	2	6	-	-	-
<i>Ionosphere*</i>	351	2	34	-	-	-
<i>Crx</i>	690	2	6	4	5	-
<i>Breastw</i>	699	2	9	-	-	-
<i>Diabetes</i>	768	2	8	-	-	-
<i>Vehicle</i>	846	4	18	-	-	-
<i>German</i>	1000	2	7	2	10	1
<i>Segment*</i>	2310	7	19	-	-	-

have not considered 3 features (6,8, and 9) because in one of the classes they take only one value and this fact makes impossible to compute our KDE classifier. In the *Ionosphere* dataset we have discarded features 1 and 2, actually the feature 2 assumes the same value in both classes and feature 1 assumes only one value in one of the classes. In the *Segment* dataset the feature 3 assumes the same value for all classes so it is totally irrelevant, features 4 and 5 assume, with very few exceptions, only one value in some classes, hence we have not considered those three features.

For all the selection methods, except FINCO, the setup was as follows: The experiment was repeated 10 times for datasets with less than 10 features otherwise it was repeated 20 times. The size of the the best subset was determined by averaging the number of features selected in each repetition, and the best subset was built using the features with the highest frequencies. FINCO was done only one time since it does not involve randomness. In the RELIEF

Table 2
Percentages of features selected for wrappers with KDE classifiers and Filters

Dataset	Wrappers (Standard kernel)			Wrappers (Adaptive kernel)			Filters		
	SBS	SFS	SFFS	SBS	SFS	SFFS	RELIEF	LVF	FINCO
<i>Iris</i>	0.750	0.750	0.500	0.750	0.500	0.250	0.750	0.750	0.750
<i>Sonar</i>	0.933	0.183	0.117	0.950	0.150	0.100	0.617	0.267	0.067
<i>Glass</i>	0.500	0.500	0.500	0.833	0.500	0.333	0.667	0.667	0.667
<i>Heartc</i>	0.846	0.385	0.308	0.615	0.462	0.385	0.692	0.462	0.462
<i>Bupa</i>	0.667	0.667	0.500	0.833	0.500	0.500	0.500	0.667	0.667
<i>Ionosphere</i>	0.531	0.219	0.156	0.844	0.188	0.156	0.656	0.250	0.156
<i>Crx</i>	0.667	0.267	0.200	0.733	0.333	0.200	0.600	0.467	0.400
<i>Breastw</i>	0.667	0.444	0.333	0.778	0.556	0.444	0.556	0.444	0.333
<i>Diabetes</i>	0.750	0.500	0.500	0.625	0.375	0.625	0.375	0.500	0.375
<i>Vehicle</i>	0.667	0.444	0.333	0.778	0.333	0.333	0.555	0.389	0.278
<i>German</i>	0.850	0.250	0.200	0.800	0.250	0.250	0.650	0.550	0.350
<i>Segment</i>	0.625	0.438	0.375	0.313	0.438	0.375	0.687	0.375	0.312
AVERAGE	0.704	0.421	0.335	0.738	0.382	0.308	0.629	0.482	0.401

method, the relevance threshold, a value close to zero, was chosen after repeating the experiment and observing a gap on the relevance weights of the features. The value of the threshold depends on the dataset being used. The number of instances to chose randomly for updating the relevance weights was taken equal to the number of instances in the dataset. In LVF and FINCO we first computed the inconsistency measure of the dataset including all the features and this value was taken as the inconsistency threshold in each dataset. The number of iterations in the LVF was chosen between 200 and 5000 depending on the number of features available in the dataset. Table 2 shows the percentages of features selected with each of the six methods and both kernel density estimate (KDE) classifiers: the standard kernel (fixed bandwidth) and the adaptive kernel. Notice that clearly SBS selects the highest number of features with both type of kernels whereas SFFS selects the smallest number of features. Table 3 shows the misclassification error rate before feature selection and the ratios after/before feature selection of misclassification error for five methods using the standard KDE classifier. Table 4 shows the same information but using the adaptive KDE classifier. The SBS method was excluded in Tables 3 and 4 because besides selecting a large number of features it also took a long time to select the best subset.

Table 3

Cross-validation errors before feature selection and ratios after/before of the CV errors for wrappers and filters using the standard KDE classifier

Datasets	Error before FS	Wrappers		Filters		
		SFS	SFFS	RELIEF	LVF	FINCO
<i>Iris</i>	3.53	1.014	1.069	1.014	1.014	1.014
Sonar	17.18	0.722	1.156	0.623	0.945	1.317
Glass	44.57	0.818	0.815	1.053	0.919	0.862
Heartc	22.30	0.852	0.842	0.868	1.003	0.954
Bupa	40.75	0.882	0.887	0.798	0.898	0.798
Ionosphere	10.93	0.539	0.599	0.727	0.868	0.933
Crx	18.93	0.938	0.748	1.026	1.018	0.923
Breastw	3.64	0.947	0.992	1.087	0.972	0.999
Diabetes	25.38	0.975	0.914	1.049	0.977	0.930
Vehicle	35.15	0.840	0.910	0.937	1.098	1.057
German	28.71	0.884	0.731	0.919	1.009	0.930
Segment	15.86	0.207	0.238	0.488	0.583	0.328
AVERAGE		0.801	0.825	0.882	0.942	0.920

5 Concluding Remarks

SFFS and FINCO select the lowest number of features. Also it seems that the type of kernel used it does not have much effect on the size of the best feature subset. Furthermore the computation of a wrapper with an adaptive kernel can take much more time. As expected a wrapper method reduces more the misclassification error than a filter method. In general, all the feature selection procedures considered reduce the misclassification error in most of the datasets. FINCO seems to be biased towards small feature subsets particularly if the number of instances is small relative to the number of features as it occurs in the *Sonar* dataset. The discretization method required for LVF and FINCO may affect their performance. On average, RELIEF reduces more the misclassification error rate than LVF and FINCO, furthermore is computed more faster. FINCO and LVF have similar performance with respect to misclassification error reduction, but FINCO is computed faster. RELIEF selects more features than FINCO and LVF since only takes care of the irrelevant features but not of the redundant ones. On average SFS and SFFS reduce the misclassification error in a similar amount, but SFFS is computed faster. A

Table 4

Cross-validation errors before feature selection and ratios after/before of the CV errors for wrappers and filters using the adaptive KDE classifier

Datasets	Error before FS	Wrappers		Filters		
		SFS	SFFS	RELIEF	LVF	FINCO
<i>Iris</i>	4.47	0.825	0.895	0.825	0.825	0.825
Sonar	16.37	0.955	1.197	0.752	0.939	1.392
Glass	35.46	0.903	1.011	0.986	0.987	1.100
Heartc	21.56	0.846	0.863	0.902	1.037	0.948
Bupa	37.51	0.916	0.912	0.870	0.994	0.870
Ionosphere	10.33	0.885	1.001	0.968	1.301	1.019
Crx	17.83	0.761	0.825	0.970	0.838	0.885
Breastw	3.94	0.822	0.902	0.805	0.837	0.944
Diabetes	26,26	0.911	0.927	0.983	0.966	0.930
Vehicle	36.74	0.879	0.833	0.920	1.041	1.014
German	28.28	0.890	0.934	1.125	1.112	0.949
Segment	13.34	0.326	0.345	0.627	0.574	0.400
AVERAGE		0.826	0.887	0.894	0.954	0.939

large number of classes slows down the computation of the filters.

We have written several S-Plus (Version 6.0) functions to carry out all the procedures discussed in this paper. The functions were tested in a DELL workstation with a dual processor PENTIUM Xeon running at 933MHz, and they are available in www.math.uprm.edu/~edgar/research.html.

6 Acknowledgment

This work was supported by the Office of Naval Research grant N00014-00-1-0360 and by the PRECISE group of the Engineering School at the UPR-Mayaguez funded by NSF grant EIA 99-77071. The author wants to thank Luis Daza for some help in programming as well as Frida Coaquira for a careful checking of the results.

References

- ACUÑA, E, ROJAS, A., and COAQUIRA, F. (2002). The effect of feature selection on combining classifiers based on kernel density estimates. In K.Jajuga, A. Sokodowski, H.-H. Bock (Eds). Classification, Clustering and Data Analysis. Springer-Verlag, Berlin, 161-168.
- BLAKE, C.L. and MERTZ, C.J. (1998). UCI Repository of machine learning databases [<http://www.ics.uci.edu/~mlearn/MLRepository.html>]. Irvine, CA: University of California, Department of Information and Computer Science.
- DASH, M. and LIU, H. (1997). Feature Selection for classification. Intelligent Data Analysis I. 131-156.
- DOUGHERY, J., KOHAVI, R., and SAHAMI, M. (1995). Supervised and unsupervised discretization of continuous features. Proceedings of the Twelfth International Conference in Machine Learning, Morgan Kaufmann. San Francisco, 194-202.
- HALL, M.A. (2000) Feature Selection for Discrete and Numeric Class Machine Learning. Proc. Seventeenth International conference on Machine Learning, San Francisco, CA, Morgan. Kaufmann, 359-366.
- KIRA, K. and RENDEL, L. (1992). The Feature Selection Problem : Traditional Methods and a new algorithm. Proc. Tenth National Conference on Artificial Intelligence, MIT Press, 129-134.
- KOHAVI, R and JOHN, G. H. (1997). Wrappers for feature subset selection. Artificial Intelligence Journal, 97, 1-2 273-324.
- KONONENKO, I., SIMEC, E., and ROBNIK-SIKONJA, M. (1997). Overcoming the myopia of induction learning algorithms with RELIEFF. Applied Intelligence Vol7, 1, 39-55.
- KONONENKO, I. (1994). Estimating Attributes: Analysis and Extension of Relief. In F. Bergadano y L. D.Raedt, eds. Proc. seventh European Conference on Machine Learning, 171-182.
- LIU, H. and SETIONO, R. (1996). A probabilistic approach to feature selection-a filter solution. Proc. of the thirteenth International Conference of Machine Learning, 319-337.
- PUDIL, P., FERRI, J., NOVOVICOVÁ, J. and KITTLER, J. (1994). Floating search methods for feature selection with nonmonotonic criterion function. 12th International Conference on Pattern Recognition, 279-283.
- VENABLES, W.N. and RIPLEY, B.D. (1997). Modern Applied Statistics with S-PLUS. Second Edition. Springer-Verlag, New York.