Less Biased Measurement of Feature Selection Benefits

Juha Reunanen

ABB, Web Imaging Systems, P.O. Box 94, 00381 Helsinki, Finland
Juha.Reunanen@iki.fi

Abstract. In feature selection, classification accuracy typically needs to be estimated in order to guide the search towards the useful subsets. It has earlier been shown [1] that such estimates should not be used directly to determine the optimal subset size, or the benefits due to choosing the optimal set. The reason is a phenomenon called overfitting, thanks to which these estimates tend to be biased. Previously, an outer loop of cross-validation has been suggested for fighting this problem. However, this paper points out that a straightforward implementation of such an approach still gives biased estimates for the increase in accuracy that could be obtained by selecting the best-performing subset. In addition, two methods are suggested that are able to circumvent this problem and give virtually unbiased results without adding almost any computational overhead.

1 Introduction

Feature selection is the art of choosing a small yet descriptive subset of useful features from amongst a larger set of candidate features. There may be many reasons for doing this: one might, for example, wish to gain a deeper understanding of the prediction problem at hand, or simply to avoid the potentially costly measurement of all the features. Whatever the aim, it makes sense to assume that one should be able to identify the optimal feature subset size. Moreover, it is often desirable to have the possibility to estimate the increase in accuracy due to choosing an optimally sized subset instead of using all the candidate features.

2 Background

This section briefly describes some basic components required in a feature selection process: the classifier architecture, the evaluation mechanism for a single subset, and the search strategy for finding the useful subsets.

2.1 Classification

A plethora of approaches have been suggested for building automatic feature-based classifiers [see, e.g., 2]. In the context of this paper, the choice of the classifier architecture should be largely irrelevant. However, to verify the generality of the results, they are computed using two very different methods: the 1 nearest neighbor (1NN) classification rule [see, e.g., 3], which is quite popular in feature selection literature, and the C4.5 decision tree building algorithm [4].

C. Saunders et al. (Eds.): SLSFS 2005, LNCS 3940, pp. 198–208, 2006.

2.2 Cross-Validation

In order to guide the search for the optimal subset, a mechanism for determining the potential performance of a single subset is needed. This paper positions itself in the context of the *wrapper* approach [5, 6], where the subsets are evaluated using actual classifiers. A common choice is cross-validation (CV) [7], where the data available is first split into a number of folds. Then, one fold at a time is designated as the validation set, while the others are used for training. The validation set is classified using the classifier that is trained with the corresponding training set. When the errors for the different validation sets are counted up, an estimate for the classification performance using a specific subset is obtained.

The special case when the number of folds is equal to the number samples is usually called "leave-one-out cross-validation" (LOOCV).

Often, it is beneficial to retain the proportions of the different classes between all the folds. If this is enforced, the CV process is called *stratified*. As stratification is known to improve the accuracy of cross-validation [8], it is done in all the experiments of this paper.

2.3 Search Algorithms

Out of the myriad of algorithms suggested for the order in which the feature subsets should be evaluated, this paper experiments with two: Sequential Forward Selection, or SFS [9], and Sequential Floating Forward Selection, SFFS [10]. Both start the search with an empty subset, and, during one iteration, consider the insertion of each feature that still remains excluded. Out of these, the one whose addition results in the largest increase (or smallest decrease) in estimated performance is added to the current set. The difference between the algorithms is that SFFS allows backtracking during the search: after adding a feature, each feature currently selected is subjected to removal. The candidate most promising for deletion is pruned, if doing so yields a better performing subset of the corresponding size than was found previously.[1] In the experiments of this paper, the search is carried on until all the candidate features have been included. This way, the algorithms are able to propose a subset for each possible subset size.

2.4 Interpretation of the Results

Once the search algorithm together with the subset evaluation method has suggested several subsets of different sizes, the practitioner obviously wants to know how these subsets compare to each other: which subset size is the optimal one, and how much better is the optimal subset of that size compared to the full set containing all the candidate features? Answering these two questions is the essence of this paper.

3 The Problem

During the search process, the subset evaluation method, such as cross-validation, produces estimates that are used primarily to guide the search. However, these intermediate

[1] The bug fix pointed out by Somol et al. [11] is utilized in this paper.

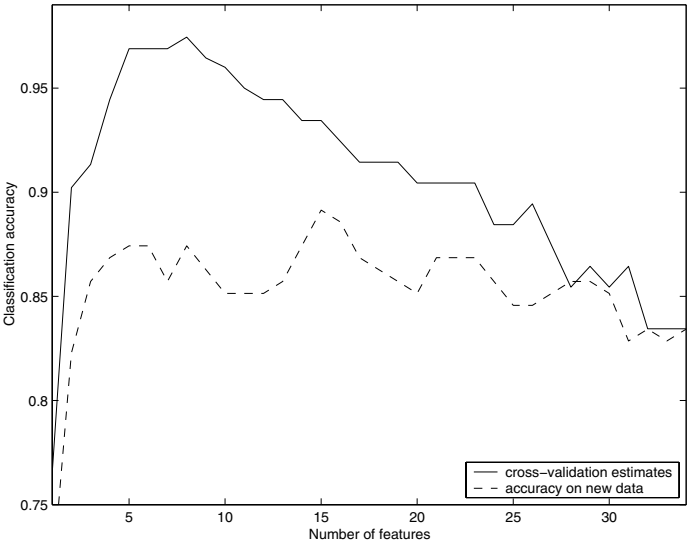

Fig. 1. Observed classification accuracies for the subsets of different sizes found by running a feature selection algorithm, as estimated with LOOCV during the search (solid line) and calculated for independent test data not seen during the search (dashed line). The data is from a single run of the experiments summarized in Table 2.

results can also be stored for later use. Once the search has finished, one could then use the same numbers to compare to each other the subsets of different sizes. An example curve drawn based on such values is shown in Fig. 1 (the solid line).

Unfortunately, when a number of such scores due to cross-validation are compared to each other to facilitate the identification of the best one, *the estimate for the winner of the comparison is no more an unbiased estimate for the accuracy of the winner*. This perhaps counterintuitive fact has been shown many times; from a pedagogical point of view, one of the most successful explanations was given by Jensen and Cohen [12].

In the context of an algorithm like SFS or SFFS, the winner subset for each size has been picked using the cross-validation values required to guide the search. This selection process renders the CV estimate for the winner of each cardinality largely useless for any later comparisons [1]. As a matter of fact, the accuracy obtained for new data tends to behave rather differently, as is shown by the dashed line of Fig. 1.

4 Existing Solutions

This section describes two methods representative of the current state of the art in determining the optimal subset size, and the performance of the best subset of that size.

4.1 Independent Test Set

It is straightforward to use an independent test set to evaluate the performances of the newly found subsets, if such a set happens to be available. On the other hand, if one has

access to more data, then one usually wants to append it to the previous dataset, in order to maximally benefit from it. In Sect. 6 of this paper, independent test sets are used to provide the ground truth, using which the approaches being tested can be compared.

4.2 Outer Loop of Cross-Validation

It has been mentioned before that an outer loop of cross-validation could be used to determine the performance of the different subsets, and hence to facilitate the choice of the optimal feature subset [1]. For example, this is what could happen: Using an outer CV loop, a researcher obtains the properly cross-validated estimates for the performance of each subset size. Then, the researcher compares the largest of these values to the estimated performance of the full set, and finds an increase of several percentage points in classification accuracy. This approach is detailed in Algorithm 1.

However, with dozens of candidate features, this method leads to overfitting on yet another level. This is because — once again — the researcher is first determining the maximum of a number of estimates, and then using that estimate.

The estimated performance for the best subset of size i found during iteration k can be thought of as a random variable $X_i^{(k)}$, realizations of which are denoted by $x_i^{(k)}$ in Algorithm 1. The random variable representing the estimate for the performance of the optimal subset (of the optimal size) is

$$\hat{X} = \max_i(\bar{X}_i) = \max_i(\frac{1}{K}\sum_{k=1}^{K} X_i^{(k)}).$$

1.	Choose a feature selection algorithm, such as SFS or SFFS.
2.	Divide all the data available into K folds.
3.	**for** $k := 1$ **to** K, (The evaluation loop.)
3.1	Create a (training) set $T^{(-k)}$, which includes the samples of all the folds, except those in the kth.
3.2	Using $T^{(-k)}$, perform a single run of the feature selection algorithm. An inner loop of cross-validation may divide $T^{(-k)}$ further.
3.3	Train classifiers using $T^{(-k)}$ and the obtained subsets of different sizes.
3.4	Test these classifiers using the samples in the kth fold, and record the performance. In the text, these estimates are referred to as $x_i^{(k)}$, $i = 1, 2, \ldots, D$, where D is the total number of candidate variables.
	end;
4.	For each subset size, determine the average of the K estimates obtained in step 3.4: $\bar{x}_i = \frac{1}{K}\sum_{k=1}^{K} x_i^{(k)}$.
5.	Out of all the subset sizes, find the one which maximizes the average score: $\hat{d} = \arg\max_i(\bar{x}_i)$. This is the estimate for the optimal subset size.
6.	The corresponding mean value, $\hat{x} = \max_i(\bar{x}_i)$, is the estimate for the maximum performance that can be attained.
7.	Perform a single run of the feature selection algorithm having all the data available.
8.	Choose the winner amongst the subsets having the size defined in step 5.

Algorithm 1. Determining the optimal subset size and the corresponding performance using an outer loop of cross-validation

A lengthy proof given by Jensen and Cohen [12, pp. 318–320] can be used almost as such to show that if every $\bar{X}_i$ is an unbiased estimate for the corresponding true performance ψ_i of a classifier built using the selected feature subset of size i, and there exists no subset size that is the optimal one in all the possible outcomes of the algorithm, then $\hat{X} = \max_i(\bar{X}_i)$ is a positively biased estimator of any ψ_i. This, in turn, makes it a positively biased estimator of the performance attainable with the best optimally sized feature subset.

5 Cross-Indexing

To obtain a truly unbiased estimate, one shall not use *any* such (probabilistic) value that was previously used to pick a certain model, or subset, from a large set of candidates. This principle is reflected in the two algorithms — called *cross-indexing A* and *B* — that this paper suggests for estimating the performance of the optimal feature subset.

Let us first discuss approach A, delineated in Algorithm 2. In step 4.1, averaged estimates for each subset size are computed much like in Algorithm 1 (step 4) but separately for each fold k, always ignoring the results obtained during the kth iteration of the evaluation loop. These estimates are then used in step 4.2 to determine the optimal subset size $\hat{d}^{(-k)}$. The corresponding performance estimate is obtained by recalling the performance estimated during iteration k of the evaluation loop for the subset having size $\hat{d}^{(-k)}$ (step 4.3). In the end, the results for the k iterations are averaged to produce the final estimates (steps 5 and 6). Next, it is shown that the positive bias of this approach, if any, is bounded by that of outer-loop CV.

1–3.	Perform steps 1–3 of Algorithm 1, including the substeps.
4.	**for** $k := 1$ **to** K, (The indexing loop.)
4.1	For each subset size, take the average of the $K-1$ estimates obtained for the *other* folds: $\bar{x}_i^{(-k)} = \frac{1}{K-1}\left(\sum_{j=1}^{k-1} x_i^{(j)} + \sum_{j=k+1}^{K} x_i^{(j)}\right)$.
4.2	Find the subset size using which maximum performance is attained: $\hat{d}^{(-k)} = \arg\max_i \left(\bar{x}_i^{(-k)}\right)$.
4.3	Record the performance for the best subset of size $\hat{d}^{(-k)}$ that was obtained during the kth iteration: $x^{(k)}_{\hat{d}^{(-k)}}$.
4.4	For comparison purposes, you can also record the performance for the *full* feature set on the kth iteration: $x_D^{(k)}$.
	end;
5.	Average all the K subset sizes obtained during the different executions of step 4.2. This average is the estimate for the optimal subset size.
6.	Average all the K performance estimates obtained in step 4.3. This average is the estimate for the performance of the best subset having the size discovered during step 5.
7–8.	Perform steps 7–8 of Algorithm 1.

Algorithm 2. The cross-indexing A algorithm. Median or other statistical descriptors could also be used instead of the average in steps 4.1, 5 and 6, if applicable.

Proposition 1. *The estimate $\hat{x}_A$ provided by Algorithm 2 is not more optimistic than the $\hat{x}$ obtained using Algorithm 1.*

Proof. The estimates provided by Algorithms 1 and 2 can be written as follows:

$$\hat{x} = \max_i \left(\frac{1}{K} \sum_{k=1}^{K} x_i^{(k)} \right) = \frac{1}{K} \sum_{k=1}^{K} x_{\hat{d}}^{(k)}, \quad \text{where}$$

$$\hat{d} = \arg\max_i \left(\frac{1}{K} \sum_{\ell=1}^{K} x_i^{(\ell)} \right) = \arg\max_i \left(\sum_{\ell} x_i^{(\ell)} \right), \quad \text{and}$$

$$\hat{x}_A = \frac{1}{K} \sum_{k=1}^{K} x_{\hat{d}^{(-k)}}^{(k)}, \quad \text{where}$$

$$\hat{d}^{(-k)} = \arg\max_i \left(\frac{1}{K-1} \left(\sum_{\ell=1}^{k-1} x_i^{(\ell)} + \sum_{\ell=k+1}^{K} x_i^{(\ell)} \right) \right) = \arg\max_i \left(\sum_{\ell \neq k} x_i^{(\ell)} \right).$$

Thus, it suffices to compare $x_{\hat{d}}^{(k)}$ and $x_{\hat{d}^{(-k)}}^{(k)}$. By definitions of $\hat{d}$ and $\hat{d}^{(-k)}$:

$$\sum_{\ell} x_{\hat{d}}^{(\ell)} \geq \sum_{\ell} x_{\hat{d}^{(-k)}}^{(\ell)} \qquad \text{and} \qquad \sum_{\ell \neq k} x_{\hat{d}^{(-k)}}^{(\ell)} \geq \sum_{\ell \neq k} x_{\hat{d}}^{(\ell)}.$$

Consequently,

$$x_{\hat{d}}^{(k)} = \textstyle\sum_\ell x_{\hat{d}}^{(\ell)} - \sum_{\ell \neq k} x_{\hat{d}}^{(\ell)} \geq \sum_\ell x_{\hat{d}^{(-k)}}^{(\ell)} - \sum_{\ell \neq k} x_{\hat{d}^{(-k)}}^{(\ell)} = x_{\hat{d}^{(-k)}}^{(k)}. \qquad \square$$

It can be observed that if $\hat{d}^{(-k)} = \hat{d}$ for all k, then the estimates provided are equal.

1–3.	Perform steps 1–3 of Algorithm 1, including the substeps.
4.	**for** $k := 1$ **to** K, (The indexing loop.)
4.1	Pick the subset size using which maximum performance was obtained on the kth execution of step 3.4: $\hat{d}^{(k)} = \arg\max_i \left(x_i^{(k)} \right)$.
4.2	Record the performance for this very subset size on all the other iterations except the kth.
4.3	Compute the average of the $K-1$ estimates obtained in step 4.2: $\tilde{x}_{\hat{d}^{(k)}}^{(-k)} = \frac{1}{K-1} \left(\sum_{\ell=1}^{k-1} x_{\hat{d}^{(k)}}^{(\ell)} + \sum_{\ell=k+1}^{K} x_{\hat{d}^{(k)}}^{(\ell)} \right)$.
4.4	For comparison purposes, you can also record the performance for the full set on all the other $K-1$ iterations.
	end;
5.	Average all the K subset sizes obtained during the different executions of step 4.1. This average is the estimate for the optimal subset size.
6.	Average all the K performance estimates obtained in step 4.3. This average is the estimate for the performance of the best subset having the size discovered during step 5.
7–8.	Perform steps 7–8 of Algorithm 1.

Algorithm 3. The cross-indexing B algorithm. Again, the statistical measure computed in steps 4.3, 5 and 6 need not be the average.

On the other hand, in cross-indexing B outlined in Algorithm 3, the estimate number k for the optimal subset size, $\hat{d}^{(k)}$, is determined in step 4.1 using the estimates obtained during the kth iteration of the evaluation loop. This estimate is then used to look up the performances for the same subset size, but for the other iterations (step 4.3).

The cross-indexing algorithms described produce only a single value for both the optimal subset size and the corresponding performance. However, those estimates that undergo averaging in steps 5 and 6 of Algorithms 2 and 3 could also be used to determine some kind of confidence intervals, or to assess the stability of the solution. Unfortunately, such attempts are outside the scope of this paper.

6 Experiments

This section compares the following four methods for determining the optimal subset size and for assessing the performance of the best subset having that size:

0. No outer loop of cross-validation at all,
1. Outer-loop CV (Algorithm 1),
2. Cross-indexing A (Algorithm 2), and
3. Cross-indexing B (Algorithm 3).

For each dataset, the optimal subset size is determined with every method. Also, the increase in accuracy that can be obtained when the optimal subset of that size is chosen, instead of the full set, is estimated. Then, a classifier is trained using the said optimal subset, and that classifier is used to classify the held-out test data. Doing the same with the full feature set and subtracting gives us the ground-truth improvement due to selecting features. This value can then be compared to the improvement predicted by the estimation approach.

The cardinality of the optimal subset as determined using method i is denoted with $\hat{d}_i$, and this value when divided by the total number of candidate features (D) and multiplied by 100% is signified by η_i. The estimated benefit due to choosing $\hat{d}_i$ features, i.e., the difference between the estimated accuracies using the optimal subset and the full set, is signified by $\delta_i^{(e)}$. On the other hand, the same difference when measured utilizing the held-out test set is denoted using $\delta_i^{(t)}$. To determine the bias of the different approaches, we need to determine the difference between these two differences: $\Delta_i = \delta_i^{(e)} - \delta_i^{(t)}$.

The smaller the absolute value of Δ_i, the smaller the bias in determining the benefits due to choosing the optimal subset found using the corresponding estimation method. Thus, from the viewpoint of this paper, the best method is signified by the smallest value of $|\Delta_i|$.

To estimate the standard deviations of the said key figures, every experiment is repeated 30 times with a different seed for the random number generator.

In the context of the 1NN classifier, the type of the inner cross-validation loop is varied: namely, LOOCV and 5-fold CV are used. However, LOOCV gets computationally too expensive with the C4.5 induction algorithm: therefore, such experiments are not done. The outer CV or cross-indexing loop always uses 5 folds.

6.1 Datasets

The datasets used in the experiments are summarized in Table 1. Each of them is publicly available at the UCI Machine Learning Repository.[2]

Table 1. The datasets used in the experiments. The number of features in the set is denoted by D. One fth of the samples are used during the search (see text). The classwise distribution of the samples in the original set is shown in the next column, and the number of training samples used (roughly the total number of samples divided by f) is given in the last column, denoted by m.

dataset	D	f	samples	m
`dermatology`	33	2	20–112 (total 366)	184
`ionosphere`	34	2	126 and 225	176
`mushroom`	112	10	3916 and 4208	813
`sonar`	60	2	97 and 111	105
`spectf`	44	2	95 and 254	175
`waveform`	40	5	1653–1692 (total 5000)	1000
`wdbc`	29	2	212 and 357	284
`wpbc`	32	2	47 and 151	99

The `mushroom` dataset in the repository has 21 categorical features for which no value is missing. In the experiments of this paper, 112 binary features are generated from them using 1-of-N coding: a categorical feature having N possible values will generate N binary features, such that the jth of these is assigned the value 1 if the sample, according to this feature, belongs to the jth category, and 0 otherwise. The `mushroom` set is chosen because it has previously expressed interesting behavior in the context of feature selection [13].

Before doing anything, each dataset is divided into the set to be used during the search, and the held-out independent test set. It is this division and all the subsequent steps that are — for each combination being tested — repeated for 30 times. The split is controlled using the parameter f (see Table 1): the dataset is first divided into f subsets, of which one is chosen as the set to be used during the search while the other $f - 1$ subsets constitute the hold-out set. Note that this is not related to any of the different levels of cross-validation: the purpose of the parameter f is just to make sure that the training sets do not get prohibitively large in those cases where the dataset happens to have a lot of samples. CV is then done during the search — potentially in two nested loops — in order to be able to guide the selection towards the useful feature subsets, and to estimate their benefits.

6.2 Results

For clarity, the figures introduced in the beginning of this section ($\hat{d}_i$, η_i, $\delta_i^{(e)}$, $\delta_i^{(t)}$ and Δ_i) are first shown in Table 2 for a single dataset using the 1NN classifier and the SFS search algorithm. Then, more results are lined up: Table 3 contains the essential

[2] http://www.ics.uci.edu/∼mlearn/MLRepository.html

Table 2. Results obtained for the `ionosphere` dataset using the 1NN classifier and the SFS algorithm. The different values of i refer to the different approaches: no outer loop of CV at all ($i = 0$), outer-loop CV as in Algorithm 1 ($i = 1$), cross-indexing A ($i = 2$), and cross-indexing B ($i = 3$). Smaller (absolute) value of $\Delta_i = \delta_i^{(e)} - \delta_i^{(t)}$ implies less observed bias, and thus a better method. The '$\pm$' signifies a single standard deviation.

inner CV	i	$\hat{d}_i$	η_i (%)	$\delta_i^{(e)}$	$\delta_i^{(t)}$	Δ_i
LOO	0	6 ± 4	19 ± 13	14 ± 3	2 ± 3	12 ± 4
	1	9 ± 5	25 ± 15	5 ± 2	2 ± 2	3 ± 3
	2	9 ± 5	25 ± 13	2 ± 3	2 ± 3	-1 ± 4
	3	8 ± 3	23 ± 9	2 ± 3	2 ± 3	-0 ± 4
5-fold	0	7 ± 4	21 ± 11	11 ± 3	2 ± 3	9 ± 3
	1	13 ± 9	39 ± 26	5 ± 3	1 ± 3	4 ± 4
	2	11 ± 6	32 ± 16	2 ± 4	1 ± 3	0 ± 5
	3	9 ± 4	26 ± 12	1 ± 3	2 ± 3	-0 ± 4

Table 3. Results like those in Table 2 but for all the datasets, as obtained using the SFS algorithm and the 1NN classifier architecture. Again, a smaller absolute value of Δ_i suggests that the estimation method is less biased, thus better.

dataset	inner CV	η_0	η_1	η_2	η_3	Δ_0	Δ_1	Δ_2	Δ_3
`dermatology`	LOO	52 ± 26	68 ± 24	68 ± 21	49 ± 15	4 ± 2	2 ± 2	0 ± 3	-0 ± 3
	5-fold	53 ± 16	69 ± 22	73 ± 16	48 ± 11	4 ± 2	2 ± 2	1 ± 2	0 ± 3
`ionosphere`	LOO	19 ± 13	25 ± 15	25 ± 13	23 ± 9	12 ± 4	3 ± 3	-1 ± 4	-0 ± 4
	5-fold	21 ± 11	39 ± 26	32 ± 16	26 ± 12	9 ± 3	4 ± 4	0 ± 5	-0 ± 4
`mushroom`	LOO	26 ± 20	69 ± 26	76 ± 21	30 ± 11	1 ± 1	0 ± 0	-0 ± 0	-1 ± 2
	5-fold	13 ± 8	60 ± 24	57 ± 24	18 ± 8	1 ± 2	0 ± 0	-0 ± 0	-2 ± 4
`sonar`	LOO	50 ± 18	66 ± 17	65 ± 16	53 ± 10	17 ± 7	5 ± 5	0 ± 6	-2 ± 7
	5-fold	38 ± 16	61 ± 18	61 ± 12	46 ± 10	18 ± 6	4 ± 6	-2 ± 7	-2 ± 7
`spectf`	LOO	32 ± 14	33 ± 23	37 ± 21	31 ± 12	18 ± 5	7 ± 6	3 ± 8	2 ± 6
	5-fold	30 ± 19	36 ± 27	35 ± 20	32 ± 13	16 ± 3	8 ± 4	3 ± 6	2 ± 5
`waveform`	LOO	52 ± 13	63 ± 17	63 ± 14	56 ± 9	6 ± 1	2 ± 2	1 ± 2	-0 ± 2
	5-fold	46 ± 11	58 ± 17	60 ± 14	53 ± 11	5 ± 2	2 ± 2	1 ± 2	0 ± 2
`wdbc`	LOO	35 ± 27	75 ± 28	78 ± 19	60 ± 15	4 ± 3	1 ± 2	0 ± 3	-0 ± 3
	5-fold	44 ± 21	68 ± 26	67 ± 20	49 ± 15	5 ± 2	2 ± 3	0 ± 4	-0 ± 3
`wpbc`	LOO	37 ± 30	77 ± 29	76 ± 22	56 ± 17	17 ± 11	6 ± 8	2 ± 10	0 ± 9
	5-fold	27 ± 20	66 ± 32	62 ± 25	48 ± 13	19 ± 10	7 ± 9	2 ± 9	1 ± 9

information of Table 2, but for all the datasets. For brevity, only η_i and Δ_i are now shown. Tables 4 and 5 report the results for the combinations C4.5/SFS and 1NN/SFFS, respectively.

6.3 Discussion

Table 2 shows us directly that for the `ionosphere` dataset, the estimates provided by method 0 — no outer loop of cross-validation at all — have a significant amount of optimistic bias, which is not a new result [1]. As expected, the straightforward outer-loop CV (approach 1) clearly lessens the problem, but does not nullify it. On the other hand, it seems that both cross-indexing methods are able to make the bias effectively vanish.

Based on Table 3, it can be readily observed that also for the other datasets, the bias incurred by cross-indexing (Δ_2 or Δ_3) is much smaller than that caused by the other approaches (Δ_0 and Δ_1). This difference really does make a difference when we want to estimate the degree of improvement (in the accuracy of a classifier) that can be attained by running a feature selection algorithm.

The `mushroom` dataset deserves some special attention. While the outer-loop CV and cross-indexing A are basically able to report perfectly unbiased accuracy results for it, those methods fail to identify the fact that virtually error-free results can be obtained with much less than half the number of features. Although the results due to cross-indexing B are slightly biased to the negative direction, it is able to report a much smaller optimal feature subset size, which already separates the classes extremely well.

In general, it can be observed that the estimates given for the size of the optimal subset are surprisingly close to each other for the outer-loop CV (method 1) and cross-indexing A (method 2) — it is just that the cross-indexing approach gives a less biased estimate for the performance. On the other hand, it appears that cross-indexing B (method 3) can identify equally performing subsets that, on the average, tend to be somewhat smaller.

Finally, Tables 4 and 5 reveal that the observations made are not too dependent on a particular choice of classifier architecture or subset selection strategy.

Table 4. Results calculated using the SFS strategy together with the C4.5 classifier

dataset	inner CV	η_0	η_1	η_2	η_3	Δ_0	Δ_1	Δ_2	Δ_3
`dermatology`	5-fold	42 ± 19	66 ± 25	61 ± 22	37 ± 13	4 ± 3	2 ± 2	1 ± 2	-0 ± 3
`ionosphere`	5-fold	34 ± 21	62 ± 33	56 ± 28	34 ± 14	6 ± 3	2 ± 2	-1 ± 3	-1 ± 3
`mushroom`	5-fold	20 ± 14	40 ± 25	38 ± 23	18 ± 10	1 ± 0	0 ± 0	0 ± 0	-0 ± 0
`sonar`	5-fold	24 ± 15	29 ± 25	35 ± 20	27 ± 12	17 ± 8	6 ± 6	0 ± 9	0 ± 8
`spectf`	5-fold	35 ± 21	61 ± 32	57 ± 25	42 ± 14	13 ± 6	4 ± 5	-0 ± 4	-1 ± 5
`waveform`	5-fold	36 ± 17	52 ± 25	52 ± 19	41 ± 12	5 ± 2	2 ± 2	-0 ± 2	-0 ± 2
`wdbc`	5-fold	30 ± 17	44 ± 30	44 ± 26	27 ± 13	4 ± 3	2 ± 2	0 ± 2	-0 ± 2
`wpbc`	5-fold	42 ± 23	21 ± 21	22 ± 19	23 ± 13	9 ± 8	2 ± 7	0 ± 7	1 ± 8

Table 5. Results for the SFFS algorithm and the 1NN classifier

dataset	inner CV	η_0	η_1	η_2	η_3	Δ_0	Δ_1	Δ_2	Δ_3
`dermatology`	LOO	34 ± 10	77 ± 25	72 ± 20	50 ± 11	6 ± 3	2 ± 2	1 ± 3	1 ± 3
	5-fold	51 ± 20	74 ± 22	72 ± 18	48 ± 10	5 ± 4	2 ± 3	1 ± 3	-0 ± 3
`ionosphere`	LOO	23 ± 11	26 ± 20	28 ± 18	23 ± 8	13 ± 5	3 ± 4	-1 ± 5	-1 ± 5
	5-fold	21 ± 9	36 ± 25	34 ± 18	28 ± 12	9 ± 4	3 ± 3	-1 ± 4	-0 ± 3
`mushroom`	LOO	7 ± 2	77 ± 27	67 ± 24	20 ± 9	4 ± 12	0 ± 0	-0 ± 0	-1 ± 1
	5-fold	9 ± 6	60 ± 26	55 ± 23	20 ± 10	2 ± 7	0 ± 0	-0 ± 0	-2 ± 6
`sonar`	LOO	26 ± 7	62 ± 26	65 ± 19	47 ± 15	21 ± 8	7 ± 7	2 ± 7	-0 ± 7
	5-fold	39 ± 15	63 ± 21	59 ± 15	43 ± 10	20 ± 7	6 ± 6	-0 ± 6	-1 ± 5
`spectf`	LOO	42 ± 13	43 ± 31	41 ± 25	33 ± 13	20 ± 6	7 ± 7	-0 ± 7	-0 ± 5
	5-fold	34 ± 18	33 ± 26	36 ± 19	33 ± 12	16 ± 4	6 ± 5	1 ± 5	1 ± 5
`waveform`	LOO	49 ± 9	55 ± 20	53 ± 13	50 ± 10	7 ± 2	2 ± 2	0 ± 2	0 ± 2
	5-fold	49 ± 14	58 ± 17	58 ± 15	52 ± 9	6 ± 2	2 ± 2	0 ± 2	-1 ± 2
`wdbc`	LOO	18 ± 7	75 ± 26	70 ± 21	47 ± 15	4 ± 3	1 ± 2	0 ± 2	-1 ± 2
	5-fold	54 ± 20	73 ± 23	71 ± 17	46 ± 13	4 ± 3	1 ± 2	-1 ± 3	-1 ± 3
`wpbc`	LOO	13 ± 6	64 ± 38	64 ± 31	50 ± 18	20 ± 9	7 ± 6	2 ± 7	1 ± 7
	5-fold	37 ± 22	61 ± 31	57 ± 22	43 ± 14	21 ± 9	9 ± 8	2 ± 8	3 ± 8

7 Summary

In feature selection, a practitioner typically needs to know the optimal feature subset size for a given dataset, and how much choosing the optimal subset of that size increases the performance.

Traditionally, cross-validation is used to give minimally biased results while using the data available as effectively as possible. However, when cross-validation is done in evaluating and comparing the models, an outer loop of cross-validation is needed for assessing the winner model. Indeed, an outer loop has been suggested for measuring the benefits due to feature selection.

In this paper, it is shown that a simple implementation of an outer cross-validation loop still gives biased estimates for the accuracy of the optimal subset as compared to the full set comprised of all the features. To tackle this problem, a new approach called "cross-indexing" is introduced in the form of two algorithms. They require practically no extra computation, nevertheless are able to give superior, virtually unbiased estimates.

References

[1] J. Reunanen. A pitfall in determining the optimal feature subset size. In *Proc. of the 4th Int. Workshop on Pattern Recognition in Information Systems (PRIS 2004)*, pages 176–185, Porto, Portugal, 2004.

[2] R. J. Schalkoff. *Pattern Recognition: Statistical, Structural and Neural Approaches*. John Wiley & Sons, Inc., 1992.

[3] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach*. Prentice–Hall International, 1982.

[4] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, 1993.

[5] G. H. John, R. Kohavi, and K. Pfleger. Irrelevant features and the subset selection problem. In *Proc. of the 11th Int. Conf. on Machine Learning (ICML-94)*, pages 121–129, New Brunswick, NJ, USA, 1994.

[6] I. Guyon and A. Elisseeff. An introduction to variable and feature selection. *Journal of Machine Learning Research*, 3:1157–1182, 2003.

[7] M. Stone. Cross-validatory choice and assessment of statistical predictions. *Journal of the Royal Statistical Society*, 36(2):111–133, 1974.

[8] R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proc. of the 14th Int. Joint Conf. on Artificial Intelligence (IJCAI-95)*, pages 1137–1143, Montreal, Canada, 1995.

[9] A. W. Whitney. A direct method of nonparametric measurement selection. *IEEE Transactions on Computers*, 20(9):1100–1103, 1971.

[10] P. Pudil, J. Novovičová, and J. Kittler. Floating search methods in feature selection. *Pattern Recognition Letters*, 15(11):1119–1125, 1994.

[11] P. Somol, P. Pudil, J. Novovičová, and P. Paclík. Adaptive floating search methods in feature selection. *Pattern Recognition Letters*, 20(11–13):1157–1163, 1999.

[12] D. D. Jensen and P. R. Cohen. Multiple comparisons in induction algorithms. *Machine Learning*, 38(3):309–338, 2000.

[13] J. Reunanen. Overfitting in making comparisons between variable selection methods. *Journal of Machine Learning Research*, 3:1371–1382, 2003.