ELSEVIER

Artificial Intelligence 116 (2000) 1–16

Artificial Intelligence

www.elsevier.com/locate/artint

Unsupervised stratification of cross-validation for accuracy estimation

N.A. Diamantidis[a,1], D. Karlis[b,2], E.A. Giakoumakis[a,*]

[a] *Athens University of Economics and Business, Informatics Department, 76 Patission St., Athens 10434, Greece*

[b] *Athens University of Economics and Business, Department of Statistics, 76 Patission St., Athens 10434, Greece*

Received 7 October 1997; received in revised form 25 September 1998

Abstract

The rapid development of new learning algorithms increases the need for improved accuracy estimation methods. Moreover, methods allowing the comparison of several different learning algorithms are important for the performance evaluation of new ones. In this paper we propose new accuracy estimation methods which are extensions of the k-fold cross-validation method. The methods proposed construct cross-validation folds deterministically instead of using the random sampling approach. The deterministic construction of folds is performed using unsupervised stratification by exploiting the distribution of instances in the instance space. Our methods are based either on the one-center approach or on clustering procedures. These methods attempt to construct more representative folds, therefore reducing the bias of the resulting estimator. At the same time, our methods allow direct comparisons between the performance of learning algorithms in different experiments, since no randomness is present. A simulation experiment examining the performance of the proposed methods is reported, depicting their behavior in a variety of situations. The new methods reduce mainly the bias of the estimator. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Machine learning; Inductive learning; Cross-validation; Accuracy estimation; Clustering

[*] Corresponding author. Email: mgia@aueb.gr.
[1] Email: nad@aueb.gr.
[2] Email: karlis@stat-athens.aueb.gr.

0004-3702/00/$ – see front matter © 2000 Elsevier Science B.V. All rights reserved.
PII: S0004-3702(99)00094-6

1. Introduction

A large amount of research in machine learning has been devoted to inductive learning, with the goal of constructing classification rules from a set of observations. In inductive learning, a learning algorithm processes a set of N instances $x_1, x_2, \ldots, x_N$ collectively called the training set $\boldsymbol{x}$. Each instance is described by an attribute-valued vector along with a class denoted by $C(x_i)$. For example, in medical diagnosis the attributes can be clinical characteristics of the patient, while the class can be the diagnosis for this patient. Suppose that there are s distinct classes with values, $y_1, \ldots, y_s$ and that $C(x_i) = y_j$ $(i = 1, \ldots, N,\ j = 1, \ldots, s)$ implies that the ith instance belongs to the class with value y_j. In our example, the classes are the different possible diagnoses. The problem of inductive learning is to construct a classifier based on the currently available training set.

The performance of a classifier can be measured by its accuracy, which expresses the degree of success in classifying correctly new instances. The *true accuracy* θ of a classifier is defined as the probability of correctly predicting the class for a randomly selected instance. It can be computed as the ratio of correctly classified instances to the total number of instances when the classifier is tested on the population. However, in the real world, samples are always finite and in many cases their size is relatively small. This makes reliable estimation of the true, but unknown, accuracy of a learning algorithm an important but difficult task.

The most obvious method for estimation of the true accuracy is to use the initial dataset for both training and testing the classifier. This estimator of accuracy is called *apparent accuracy*. Apparent accuracy $\hat{\theta}_c$ is an optimistic estimator, since it favors classifiers that overfit the data. In practice, accuracy estimates are made by constructing disjoint training and test sets using sampling methods.

Reliable estimates of classification accuracy are important, not only for estimating the true accuracy of a classifier, but also for finding the best classifier from a set of competitive ones (model selection). There is no universal learning algorithm giving the best performance in all possible learning situations [24]. The development of inductive learning algorithms is more concentrated on designing algorithms that can deal with real world applications. In this direction, collections of datasets related to real world situations (for example, the UCI repository of machine learning databases [18]), are being continuously used to compare learning algorithms. Researchers often use the results of these comparisons in order to draw conclusions on the superiority of one algorithm over another. Therefore, objective methods for estimating accuracy are needed.

Three main methods have been applied for accuracy estimation purposes:

(a) holdout with random resampling,
(b) cross-validation, and
(c) bootstrap.

All these methods are based on random sampling techniques and, thus, randomness may affect their behavior, especially with respect to the comparison of learning algorithms in different experiments. In this paper we propose deterministic approaches for k-fold cross-validation that construct representative rather than random folds. We refer to these methods as *unsupervised stratification of cross-validation*, because they exploit the distribution of instances in the instance space in an unsupervised manner, ignoring the class. The principle for constructing representative folds in unsupervised stratification is to direct similar instances to different folds. With these methods we attempt to reduce the effects of using less instances for training. Furthermore, the proposed methods are deterministic, i.e., given the dataset, unsupervised stratification always constructs the same folds. Hence, unsupervised stratification can be used as a tool by researchers for comparing learning algorithms in different experiments, as an alternative to predefined holdout test sets or specific cross-validation folds.

The rest of this paper is organized as follows: In Section 2 we review some methods for accuracy estimation and examine the factors that influence their performance. In Section 3 we present the idea of unsupervised stratification of cross-validation. Section 4 presents a method for unsupervised stratification based on the notion of the center of the instance space. A method of unsupervised stratification based on clustering procedures is presented in Section 5. An experimental comparison of the new methods to random cross-validation ones is reported in Section 6. Finally, some concluding remarks can be found in Section 7.

2. Accuracy estimation methods

In this paper we focus on the application of accuracy estimation to classification problems. A supervised learning algorithm constructs a classifier, in order to predict the class of unlabelled instances. Classifiers can be decision trees [2,21], production rules [5], or classification rules such as nearest neighbor and naive Bayes [27]. Several methods have been proposed for calculating an estimator $\hat{\theta}$ of the true accuracy θ. Generally, the *bias* of an estimator $\hat{\theta}$ is defined as $\mathit{Bias} = E[\hat{\theta}] - \theta$, where E denotes expectation. The *variance* of the method is defined as $\mathit{Var}(\hat{\theta}) = E[(\hat{\theta} - E[\hat{\theta}])^2]$. The bias expresses how the estimator overestimates or underestimates the true accuracy, while the variance expresses the variability of the estimator. An estimator is reliable if it has low bias and variance. In practice, an appropriate tradeoff between bias and variance is a more realistic target. Squared error can be decomposed to bias plus variance, a well-known result in statistics [22]. Similar decompositions apply to zero-one loss functions appearing in classification problems [15,16].

A simple method for accuracy estimation is *holdout with random resampling*. In this method the dataset is randomly partitioned in two disjoint subsets of $p \cdot N$ and $(1 - p) \cdot N$ instances (typically $p = 1/2$ or $p = 2/3$). The first subset serves as the training set and the second as the test set. The classifier is built using the training set and it is tested on the test set. This procedure is repeated for a number of times and the mean accuracy is computed. The drawback of this method is that it makes inefficient use of data, since typically a relatively large proportion of the instances is used for testing. The resulting estimates can be highly pessimistic.

Cross-validation [26] attempts to resolve this drawback by successively removing some instances from the initial set, treating them as a test set. In *k-fold cross-validation*, the dataset is randomly partitioned into k disjoint blocks (the *folds*), of (approximately) equal size d $(d \approx N/k)$. The learning algorithm runs k times. In the ith run, the ith training set is formed by the initial dataset without the ith fold, while the test set is formed using the ith fold alone. Let $\hat{\theta}_{(i)}$ be the ratio of correctly classified instances to the total number of tested instances in the ith run. The estimator $\hat{\theta}_k$ of the accuracy for the k-fold cross-validation method is calculated as $\hat{\theta}_k = \sum_{i=1}^{k} \hat{\theta}_{(i)}/k$. Note that complete cross-validation implies the construction of all possible folds containing d instances, making the evaluation computationally intractable even for small sample sizes.

A variation of cross-validation is *stratified cross-validation*, where the class distribution in each fold is approximately the same as in the initial dataset. In general, cross-validation is appropriate for both accuracy estimation and model selection [2,3,23]. However, it usually produces pessimistic accuracy estimations, because in each iteration only a subset of the instances is used for training. Typically, the bias is smaller than in the holdout with random resampling method. Pessimism in accuracy estimation occurs if for a specified algorithm and dataset, more training instances improve classification performance. Theoretical studies and empirical evaluation show that this does not always happen [13,24].

The tradeoff between bias and variance in cross-validation depends on the number of folds [13,27]. A small number of folds typically results in small variance and large bias. Since in each run a much smaller portion of examples is used for training, accuracy estimation is more pessimistic. On the other hand, for larger numbers of folds, bias becomes lower but variance increases. For example, when applying the *leave one out* method in which each fold contains only one instance, the estimate is almost unbiased but its variance is relatively high. Empirical studies show that stratified cross-validation gives better estimates for both bias and variance [13]. Finally, multiple runs of cross-validation produce more stable estimations as they reduce the variance [13].

Another widely used technique is the *bootstrap* method [8]. In bootstrap, each training set is formed by randomly drawing instances from the initial dataset with replacement. The classifier constructed using the ith training set is tested on the remaining instances, resulting in an estimate $\hat{\theta}_i$. This is repeated b times. The 0.632 bootstrap estimator $\hat{\theta}_B$ is defined as [7]:

$$\hat{\theta}_B = \frac{1}{b}\sum_{i=1}^{b}\bigl(0.632\hat{\theta}_i + 0.368\hat{\theta}_c\bigr),$$

where $\hat{\theta}_c$ is the apparent accuracy defined earlier.

Weiss and Kulikowski [27] suggested using the bootstrap method for small datasets because it makes better use of the data. However, the 0.632 bootstrap estimator favors overfitting classifiers (e.g., perfect memorizers) and it is not suitable for model selection [13]. Efron and Tibshirani [9] proposed new estimators to deal with this form of bias. They combined the leave one out with the bootstrap method to obtain smoother estimates. They also proposed reducing the bias of the 0.632 bootstrap estimator by using the 0.632+ estimator which is defined as

$$\hat{\theta}_{B+} = \frac{1}{b}\sum_{i=1}^{b}\bigl(\hat{w}\hat{\theta}_i + (1-\hat{w})\hat{\theta}_c\bigr),$$

where $\hat{w}$ is a weight that ranges from 0.632 to 1, estimated from the dataset so as to reduce the bias. This estimator provides a better balance between the pessimistic $\hat{\theta}_i$ estimator and the optimistic $\hat{\theta}_c$ estimator.

3. Representative partitioning for cross-validation

The cross-validation methods presented previously were based on randomly constructed folds. This has two disadvantages. The first is due to the fact that cross-validation is used by researchers for comparing inductive algorithms. Even though cross-validation may give good results, the randomness in partitioning the dataset results in different estimates between runs. Thus, the comparison of different learning experiments becomes difficult. The second disadvantage is that the folds can be quite unrepresentative, consequently biasing the estimator. The deterministic variations of cross-validation described below exploit the instance space and attempt to direct similar instances to different folds. We refer to these methods as unsupervised stratification, to distinguish them from stratification based on class distribution, referred to as *supervised stratification* in the rest of this paper.

The hypothetical two-class learning problem shown in Fig. 1 illustrates the idea behind unsupervised stratification. In the two-dimensional instance space shown, the points representing the instances clearly belong to two regions A and B. Consider now a case of two-fold cross-validation, in which the first fold mostly contains instances from region A while the second fold mostly contains instances from region B. In this scenario it is possible to get pessimistic results. In the first run the learning algorithm is trained mainly from the instances of region B and tested on many instances from region A. Similarly, in the second run the learning algorithm is trained on many instances from region A and tested on many instances from region B. The estimates produced in this case will be pessimistic if we make the (plausible) assumption that when a test case is closer to the training set the probability of a correct prediction is higher [7]. An alternative scenario for the example above is to sample an (approximately) equal number of instances from both regions to form the cross-validation folds. From this example we can see that a more objective partitioning method is to direct similar instances to different folds. Application of this principle to the example of Fig. 1 results in more balanced folds, containing instances from both regions, which are expected to give better estimates.

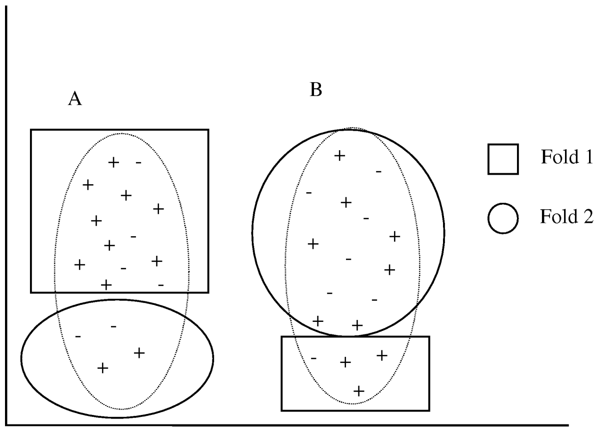

Fig. 1. A hypothetical two-dimensional learning problem. Instances belong to two regions A and B (dotted lines). The partitioning shown is unrepresentative and hence it may give unreliable results.

The objective of directing similar instances to different folds is to reduce the pessimistic effects caused by the removal of instances from the dataset. At the same time each of the k classifiers is tested on as many different testing situations as possible. When an instance is used for testing, we expect that we have left as many instances from the same region as possible for training. Thus, the effect of the absence of this instance is reduced, and consequently the bias is also reduced. In addition, with this method each fold contains distant instances. Hence, each classifier is tested on different testing situations.

The application of a systematic procedure to the construction of folds provides the ability to construct folds deterministically rather than randomly. We will describe algorithms for such partitions in the next sections.

4. Ordering instances using a single center

In order to discover regions having similar instances in the instance space, we need to define a *similarity measure*. The similarity measure used in this paper is a variation of Euclidean distance proposed by Aha et al. [1]. For n attributes the similarity between two instances x_i and x_j is defined as

$$\mathit{Sim}(x_i, x_j) = -\sqrt{\sum_{t=1}^{n} f(x_{it}, x_{jt})},$$

where x_{it} is the value for attribute t of the ith instance. The function $f(x_{it}, x_{jt})$ is defined for continuous attributes as $f(x_{it}, x_{jt}) = (x_{it} - x_{jt})^2$ while for discrete attributes it is defined as

$$f(x_{it}, x_{jt}) = \begin{cases} 0 & \text{if } x_{it} = x_{jt}, \\ 1 & \text{if } x_{it} \neq x_{jt}. \end{cases}$$

Missing attribute values are presumed to be maximally different. Continuous attributes are normalised to the [0,1] interval so that all attributes will have the same effect on the computation of similarity. Other similarity functions can also be used.

A first approach to the deterministic ordering of instances is to order them according to their similarity to the center of the instance space. In this approach we assume that instances close to the center of the instance space are themselves close. For discrete attributes the value with maximum frequency is considered the attribute value of the center. For continuous attributes the center attribute value is the mean of the known attribute values of all instances. Fig. 2 shows this method (1C-CV).

In the method shown in Fig. 2, instances with close similarities to the center are directed to different folds. Fig. 3 shows graphically the distribution of similar instances to different folds.

The single center approach is the fastest method for unsupervised stratification. The algorithm makes two passes over the data, one to find the attribute values for the center of the instance space and a second to sort instances according to their similarity to the center. Therefore, the running time for the procedure is $\mathrm{O}(N \log N)$.

```
Find the Center of the instance space
Sort instances according to their similarity to the Center
For each instance $x_{(i)}$ (in order)
   $m = i \bmod k$
   if $m = 0$ then $m = k$
   Assign instance $x_{(i)}$ to fold m
End For
```

Fig. 2. Ordering instances using a single center. The 1C-CV method.

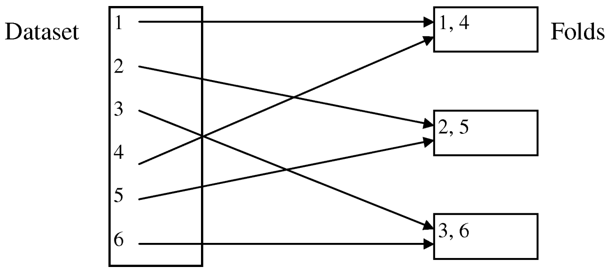

Fig. 3. Assignment of similar instances to different folds. Instances are sorted according to their similarity to the center of the instance space.

5. Ordering instances with clustering methods

The implicit assumption of the single center approach is that instances having identical similarities to the center are themselves close. Obviously this assumption does not hold in general, especially for datasets with continuous attributes. Thus, we need to find regions in the instance space where the instances are themselves close. Such a search for regions with similar instances arises also in cluster analysis [10]. Methods developed for cluster analysis can be applied to unsupervised stratification, in order to find a better approximation of the true similarity between instances. The aim of clustering is to form groups of instances, where the instances in each group are similar, and at the same time there is small similarity between clusters. We describe two approaches commonly used in cluster analysis, hierarchical clustering and K-means clustering, and discuss their application to unsupervised stratification.

5.1. Hierarchical clustering

Hierarchical clustering starts by treating each instance as a cluster and iteratively merges the two most similar clusters until a single cluster remains. Two critical issues arise in hierarchical clustering. The first is choosing a method for forming clusters and the second is choosing the number of clusters which better represents the structure of the data.

Several measures of similarity between two clusters have been proposed, such as *single linkage* (the similarity of the two closest instances), *complete linkage* (the similarity of the two furthest instances), and *average linkage*. In the context of our work, we implemented the *average linkage between groups* method. The definition of similarity for two clusters g_1 and g_2 in this method is defined as:

$$\mathit{ClustSim}(g_1, g_2) = \frac{\sum_{x_i \in g_1, x_j \in g_2} \mathit{Sim}(x_i, x_j)}{\mathit{size}(g_1) \cdot \mathit{size}(g_2)},$$

where $\mathit{size}(g_1)$ and $\mathit{size}(g_2)$ denote the number of instances assigned to each cluster. The average linkage between groups method leads to the construction of clusters having spherical form. This method is suitable for discovering regions that contain instances that are themselves close.

On the other hand, there is no standard statistical procedure to determine the optimal number of clusters. The decision is highly based on the subjective judgment of the researcher analyst or on heuristic procedures. In our work we adopt the criterion introduced in [20]. This criterion examines the distribution of the *clustering coefficient,* which is the similarity between the clusters merged in each repetition of the clustering procedure. As the number of clusters decreases, the value of the coefficient also decreases. The optimal number of clusters is given by the stage in which the obtained coefficient is smaller than a critical value derived from the distribution of the coefficient. For N instances we have $N - 1$ stages of hierarchical clustering. The number of clusters is determined by the first stage j $(1 \leqslant j \leqslant N - 2)$ in which

$$\alpha_{j+1} < \bar{\alpha} - 2 \cdot s_\alpha,$$

where α_j is the coefficient at stage j; $\bar{\alpha}$ and s_α are the mean and the standard deviation of the coefficients. The value of the coefficient in stage $j + 1$ indicates that the last merging of clusters is unrepresentative, because α_{j+1} lies in the lower tail of the distribution. If there is no stage satisfying the inequality above, we can assume either that there is no structure in the data, or that the single cluster solution represents better the structure of the data. Other criteria for choosing the optimal number of clusters and experimental comparisons between them can be found in [4,12,20].

Hierarchical clustering requires $N - 1$ iterations until we reach one cluster. Since the similarity of all pairs of instances must be considered, the running time of the method is $\mathrm{O}(N^2)$. Moreover, the similarity matrix of instances has space complexity $\mathrm{O}(N^2)$. Therefore, hierarchical clustering is suitable only for relatively small datasets.

5.2. K-means clustering

In K-means clustering, the number of clusters is provided by the user. K-means clustering partitions the data into a specified number of clusters in an iterative manner. The method is performed in three steps:

Step 1. Select the K initial centers.
Step 2. Assign each instance to the cluster with the closest center.
Step 3. Refine cluster centers. If there are significant changes to the cluster centers repeat from Step 2.

```
/*$C_r, C_f, C_h$ denote cluster centers*/
For $f = 1$ to K
     $C_f = x_f$
End For
For $i = K + 1$ to N
     $C_r$ = the closest center to $x_i$
     $S_1 = Sim(x_i, C_r)$
     /*compute the similarity $S_2$ between the two closest centers*/
     $S_2 = \max\{Sim(C_f, C_h), f = 1, \ldots, K, h = 1, \ldots, K, f \neq h\}$
     /*find the largest similarity between the center $C_r$ and all
     other centers*/
     $S_3 = \max\{Sim(C_r, C_f), f = 1, \ldots, K, f \neq r\}$
     if $(S_1 < S_2)$ or $(S_1 < S_3)$ then
        $C_r = x_i$
     End if
End For
```

Fig. 4. Selection of initial centers for K-means clustering.

There are several variations on K-means clustering. For example, the selection of initial centers (Step 1) can be random or guided by more sophisticated methods. In Step 3 the cluster centers are refined. New cluster centers are computed from the instances assigned to each cluster. This can be performed when an instance is assigned to a cluster or after the end of each repetition. In our work we applied a modified version of the method used by the SPSS statistical package [25]. In Step 1 the initial centers are selected by the procedure shown in Fig. 4. The aim of the procedure shown in Fig. 4 is the selection of distant instances for the initial centers.

In Step 3 the cluster centers are refined after the end of each repetition. The clustering procedure finishes when the largest change in the similarity of cluster centers is lower than 2% of the largest similarity between the cluster centers of the previous iteration.

Since K-means clustering processes the raw data there is no need for a similarity matrix, but we need to make more than one pass over the data. The running time of the clustering procedure is $O(KN)$. Cutting et al. [6] propose fast algorithms for K-means clustering for large datasets. The drawback of K-means clustering is that the number of clusters has to be defined by the user. We can estimate the number of clusters by sampling a fraction of instances from the dataset and applying hierarchical clustering using the heuristic procedure described above.

5.3. *Ordering instances using clusters*

After the application of any of the clustering procedures above to the dataset, we have the instance space partitioned into clusters. The next step is to order the clusters. The clusters are sorted according to the similarity between cluster centers and the center of the first

```
For each cluster
     Find the Center of the cluster
     Sort the instances of the cluster according to their
     similarity to the Cluster Center
End For
Sort Cluster according to their similarity to the first Cluster
For each instance g (in order)
     For each instance $x_{(g,i)}$ in the cluster g
         m = i mod k
         if m = 0 then m = k
         Assign instance $x_{(g,i)}$ to fold m
     End For
End For
```

Fig. 5. Ordering instances with clustering. The CL-CV method.

cluster. After the clusters have been ordered, we can sort the instances within each cluster. Instances are sorted according to their similarity to the cluster center. Suppose that $x_{(g,i)}$ denotes the ith ordered instance in the gth ordered cluster. For each cluster we distribute the instances in folds in the same manner as in the ordering procedure of the single center method. Fig. 5 shows this method (CL-CV).

Sorting the clusters has small computational cost (when $K \ll N$). The additional computational cost of sorting the instances within the clusters is $\mathrm{O}(KN \log N)$. Finally, note that if the clustering procedure reduces to a single cluster solution, then the resulting ordering is identical to the one produced using the single center approach.

6. Experimental comparison

In the previous section we introduced two new methods for deterministic ordering of instances for k-fold cross-validation. For the experimental comparison of these methods with random cross-validation (R-CV) and random supervised stratification of cross-validation (RS-CV) we followed a methodology similar to the one in [13]. In order to compare different accuracy estimation methods Kohavi experimented with datasets from the UCI repository. The datasets used were Breast Cancer, Chess, Hypothyroid, Mushroom, Soybean Large and Vehicle. Kohavi tested the C4.5 learning algorithm [21] that constructs decision tree classifiers and naive Bayes classification. Cross-validation runs were performed on subsets of the entire datasets. Using the learning curves of C4.5 and naive Bayes, the number of instances in each subset was chosen at points where the learning curve was not flat. The comparison of the accuracy estimation methods on datasets that do not have flat learning curves shows the difference of these methods more clearly. In our work, the same datasets were used for the comparison of the cross-validation methods, plus the Letter Recognition dataset from the UCI repository. The latter was chosen in order to test the application of the methods on larger datasets. For the first six datasets we used the sample sizes reported in [13] while for the Letter Recognition the sample size was chosen according to the learning curve shown in Fig 6.

Table 1 shows the sample size for each dataset. Given that these sample sizes typically lead to pessimistic estimates, we are interested to find the less pessimistic methods. The method giving the highest accuracy is the less pessimistic one.

The methodology used can be described as follows: Each accuracy estimation method ran 50 times on different samples of the size shown in Table 1. The mean accuracy and standard deviation of the runs were computed. For the clustering procedure we used K-means clustering. The number of clusters for the first six of the datasets was estimated by sampling 50 times a subset of 100 instances and choosing the number of clusters using the hierarchical clustering method. We used the same procedure for the Letter Recognition dataset but with subsets containing 200 instances. Table 2 shows the number of clusters chosen for each dataset, computed by averaging the number of clusters from each run. The standard deviations of the estimations are also presented.

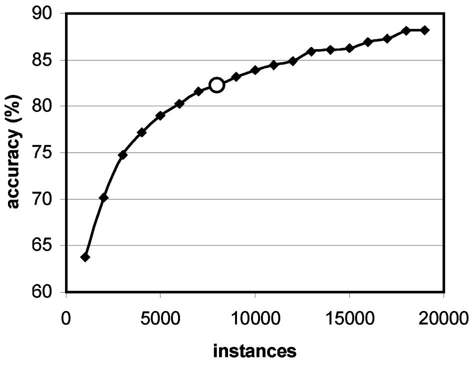

Fig. 6. The learning curve for C4.5 for the Letter Recognition dataset. Each point on the learning curve is the mean accuracy of the decision tree on the instances not found in the training set. The mean accuracy is computed over 10 runs. The empty circle indicates the dataset size used for experimentation. The learning curve was generated by the MLC++ *LearnCurve* utility [14].

Table 1
Datasets and sample sizes for experimentation

Dataset	Classes	Attributes	Discrete	Continuous	Total size	Sample size
Breast Cancer	2	9	0	9	699	50
Chess	2	36	36	0	3196	900
Hypothyroid	2	25	18	7	3163	400
Mushroom	2	22	22	0	8124	800
Soybean Large	19	35	35	0	683	100
Vehicle	4	18	0	18	846	100
Letter Recognition	26	16	0	16	20000	8000

Table 2
Estimated number of clusters. Rounded values were used in the experiment

Dataset	Number of clusters
Breast Cancer	4.18 ± 0.825
Chess	2.86 ± 0.880
Hypothyroid	4.16 ± 1.251
Mushroom	6.06 ± 0.711
Soybean Large	4.78 ± 1.183
Vehicle	4.42 ± 0.859
Letter Recognition	8.44 ± 0.907

Although the Breast Cancer, Soybean Large and Vehicle datasets permit hierarchical clustering (because of their size), we applied K-means clustering in order to get results comparable to those of other datasets. Given the number of clusters, we applied the four accuracy estimation methods (R-CV, RS-CV, 1C-CV, CL-CV) to 50 samples from each dataset using the C4.5 algorithm as the learning algorithm.

Table 3 shows the mean accuracy and standard deviation of the cross-validation methods for 2, 5, 10 and 20 folds.

Table 3 highlights the least pessimistic methods and the methods with the smallest variance for each dataset and each fold. Given the seven datasets and four cases of folds for each dataset, we need to compare all methods for both bias and variance. In Table 4 we summarize the results.

Inspection of Table 4 reveals some interesting features of unsupervised stratification. It is clear that the clustering method reduces the bias. The differences in variance are not significant enough to lead to conclusions on which method outperforms the rest. The methods proposed improve the variance compared to random cross-validation. However, random supervised stratification of cross-validation seems to slightly outperform the clustering method. The one center method does not improve significantly the estimates compared to random cross-validation. This method requires less computational effort, but it may lead to sub-optimal results when the instance space contains singularities. This method can be used as a tool for the deterministic construction of folds with low computational time. In contrast, the clustering method is more reliable, but it is computationally demanding.

Some other important points concerning the clustering method are:

(1) The clustering method gives better results for a moderate number of folds. When the number of folds is large (close to the leave one out method), it is expected that the effect of unsupervised stratification is smaller.

(2) The clustering method produces better estimates for datasets with low variance. Unsupervised stratification improves mainly the bias. Therefore, for datasets with low estimator variance the bias reduction is more important.

Table 3
Mean accuracy and standard deviation of the C4.5 algorithm for the four variations of cross-validation. Bold indicates the best results for a specified combination of fold and dataset

Dataset/method	2-folds		5-folds		10-folds		20-folds	
Breast Cancer	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	89.92	4.91	90.60	4.26	90.36	4.51	90.52	4.63
RS-CV	**90.40**	**3.79**	91.20	**3.73**	90.76	4.48	**90.97**	4.79
1C-CV	90.28	4.54	**91.40**	4.38	91.20	**4.10**	90.42	5.10
CL-CV	90.04	5.12	91.36	4.15	**91.36**	4.17	**90.97**	**4.60**
Chess	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	**96.76**	**0.70**	**97.72**	0.47	97.93	0.45	98.08	**0.42**
RS-CV	96.69	**0.70**	97.69	0.50	97.95	**0.43**	98.07	0.50
1C-CV	96.73	0.96	97.66	**0.46**	97.93	0.51	98.05	0.49
CL-CV	96.52	0.86	97.63	0.65	**97.97**	0.47	**98.12**	0.47
Hypothyroid	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	97.92	0.72	98.27	0.74	98.31	0.75	**98.44**	0.71
RS-CV	97.94	0.77	98.29	0.63	98.42	**0.62**	98.41	0.69
1C-CV	98.00	0.76	**98.39**	0.60	98.36	0.73	**98.44**	**0.68**
CL-CV	**98.02**	**0.70**	98.26	**0.56**	**98.45**	0.67	98.43	0.71
Mushroom	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	98.91	0.48	99.14	0.40	99.26	0.38	99.31	**0.40**
RS-CV	98.86	0.49	99.13	0.42	99.30	**0.35**	99.36	0.42
1C-CV	98.94	0.46	**99.23**	**0.37**	99.31	0.37	99.36	**0.40**
CL-CV	**98.98**	**0.39**	99.22	0.39	**99.35**	0.38	**99.37**	0.42
Soybean Large	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	55.32	7.09	64.54	5.58	67.84	5.10	69.06	5.45
RS-CV	**60.40**	**6.46**	67.16	**5.06**	69.26	5.20	69.06	5.57
1C-CV	56.60	6.85	66.56	5.89	68.66	5.48	70.14	4.99
CL-CV	58.72	7.07	**67.40**	5.10	**69.70**	**4.47**	**70.72**	**4.84**
Vehicle	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	**52.10**	6.42	57.26	5.41	58.76	6.77	58.10	6.85
RS-CV	51.98	6.38	57.40	5.30	58.94	6.66	**58.64**	**6.08**
1C-CV	50.72	**6.01**	**58.10**	**5.17**	57.94	6.85	58.58	6.14
CL-CV	51.46	6.69	57.74	6.90	**58.98**	**6.49**	58.46	6.48
Letter Recognition	Acc	Stdv	Acc	Stdv	Acc	Stdv	Acc	Stdv
R-CV	**77.22**	1.04	80.73	0.48	81.64	0.54	82.04	0.51
RS-CV	77.21	1.03	**80.86**	0.54	**81.68**	**0.45**	82.08	**0.47**
1C-CV	77.09	1.17	80.82	0.53	81.67	0.47	82.05	0.55
CL-CV	77.10	**0.58**	80.82	**0.48**	81.63	0.51	**82.11**	0.49

Table 4
Summary of the results. Each entry shows the number of times a method gave the best results out of the 28 cases. The first column does not add up to 28 because for many cases the method producing the best result for bias and variance differs. The last two columns do not add up to 28 because there are draws

Method	Best results for both bias and variance	Best results for bias	Best results for variance
R-CV	1	5	3
RS-CV	4	6	11
1C-CV	3	5	7
CL-CV	6	14	9

(3) If we have a large instance space (i.e., there are many attributes) with a small number of instances, then the clustering methods may give better estimates. In such cases it is more likely to find clusters of instances. A typical example of a dataset with many attributes from the experiment above is the Soybean Large one. The clustering method clearly gives better estimates. On the other hand, if we have a small number of attributes and many instances, then the clustering methods have less effect, since it is less likely to have instances belonging to well separated clusters.

(4) The clustering method can mainly improve the estimates for relatively small datasets. There is some evidence that estimates can also be improved for larger datasets. Application of the clustering method to the Letter Recognition dataset improved the variance for small numbers of folds (2 or 5) and the bias for 20 folds. However, for very large datasets, we expect the proposed methods to be less beneficial. Since there is plenty of data, we can employ enough data for testing without significantly affecting the training phase. In the case of large datasets (e.g., data mining) random cross-validation can give reliable estimates. Moreover, sampling methods can have a different goal: to construct efficiently an accurate classifier from a subset of the data, as opposed to accuracy estimation. In this case, the role of sampling methods is to find a training set much smaller than the whole dataset and to construct an accurate classifier, with a low running time [11,17,21,28].

The above points show the potential advantages of the clustering method. These points also serve as a guide for researchers and data analysts concerning the application of unsupervised stratification.

Finally, we must point out that, as in random cross-validation, the proposed methods suffer from the law of generalization performance [23,24]. This has to be taken into account when the methods are applied for model selection. When using cross-validation for model selection (including the newly proposed methods), there is a form of inductive bias in the same way as in learning algorithms [19]. As a result, using cross-validation for model selection cannot guarantee that we will select the best algorithm for all learning situations. On the other hand the assumption made by unsupervised stratification is that a test instance is more likely to be classified correctly when it is closer to the training set. This is a plausible assumption for real world applications [7], making the proposed method applicable to a wide range of learning situations. Similar assumptions made by supervised stratification seem to hold in practice [13].

7. Conclusions

The development of new and the evolution of known learning algorithms increases the need for estimation methods which can produce more reliable accuracy estimates. In this paper we presented new methods for the construction of folds for cross-validation in an objective manner without using random sampling from the initial dataset. Moreover, the accuracy estimates among different experiments are comparable since there is no randomness. The two approaches proposed for the deterministic construction of folds are the one center method and the clustering one. Our methods result in more representative allocation of observations into folds, and hence they reduce the bias.

The clustering based algorithm produces better results, but it requires more computational time. Experimental comparisons illustrated the performance of the new algorithms for several distinct datasets. The results of the experiments showed that unsupervised stratification can improve the estimates, especially in learning situations where

(a) we use a moderate number of folds,
(b) cross-validation produces estimates with small variance, or
(c) there is a large number of attributes relative to the number of instances.

Acknowledgements

The authors would like to thank the two anonymous reviewers for providing helpful comments and suggestions.

References

[1] D. Aha, D. Kibler, M. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.
[2] L. Breiman, J. Friedman, R. Olshen, C. Stone, Classification and Regression Trees, Wadsworth, Belmont, CA, 1984.
[3] L. Breiman, P. Spector, Submodel selection and evaluation in regression. The X-random case, Internat. Statist. Rev. 60 (1992) 291–319.
[4] P. Cheesman, J. Stutz, Bayesian classification (AutoClass): Theory and results, in: U.M. Fayyad, G. Piatesky-Shapiro, P. Smyth, R. Uthurusamy (Eds.), Advances in Knowledge Discovery and Data Mining, MIT Press, Cambridge, MA, 1996, pp. 153–180.
[5] P. Clark, R. Boswell, Rule induction with CN2: Some recent improvements, in: Proc. 5th European Working Session on Learning, Springer, Berlin, 1991, pp. 151–163.
[6] D.R. Cutting, J.O. Pedersen, D.R. Karger, J.W. Tukey, Scatter/Gather: A cluster-based approach to browsing large document collections, in: Proc. 15th Annual International ACM/SIGIR Conference, 1992, pp. 318–329.
[7] B. Efron, Estimating the error rate of a prediction rule: Improvement on cross-validation, J. Amer. Statist. Assoc. 78 (1983) 316–330.
[8] B. Efron, R. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall, London, 1993.
[9] B. Efron, R. Tibshirani, Cross-validation and the bootstrap: Estimating the error rate of a prediction rule, Technical Report 477, Stanford University, 1995. Available at: http://utstat.toronto.edu/tibs/research.html.

[10] B.S. Everitt, Cluster Analysis, Halsted Press, London, 1980.

[11] J. Furnkranz, Integrative windowing, J. Artificial Intelligence Res. 8 (1998) 129–164.

[12] A. Hardy, On the number of clusters, Comput. Statist. Data Anal. 23 (1996) 83–96.

[13] R. Kohavi, Wrappers for performance enhancement and oblivious decision graphs, Ph.D. Thesis, Department of Computer Science, Stanford University, 1995. Available at: http://robotics.stanford.edu/ronnyk/.

[14] R. Kohavi, D. Sommerfield, J. Dougherty, Data mining using MLC++: A machine learning library in C++, in: Tools with Artificial Intelligence, IEEE Computer Society Press, 1996, pp. 234–245.

[15] R. Kohavi, D. Wolpret, Bias plus variance decomposition for zero–one loss functions, in: Proc. 13th International Conference on Machine Learning, Bari, Italy, Morgan Kaufmann, San Mateo, CA, 1996, pp. 274–283.

[16] E.B. Kong, T.G. Dietterich, Error-correcting output coding corrects bias and variance, in: Proc. 12th International Conference on Machine Learning, Tahoe City, CA, Morgan Kaufmann, San Mateo, CA, 1995, pp. 313–321.

[17] D.D. Lewis, J. Catlett, Heterogeneous uncertainty sampling for supervised learning, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, Morgan Kaufmann, San Mateo, CA, 1994, pp. 148–156.

[18] C.J. Merz, P.M. Murphy, UCI repository of machine learning databases, http://www.ics.uci.edu/MLRepository.html, University of California, Department of Information and Computer Science, Irvine, CA, 1998.

[19] T. Mitchell, The need for biases in learning generalizations, in: J. Shavlik, T.G. Dietterich (Eds.), Readings in Machine Learning, Morgan Kaufmann, San Mateo, CA, 1990, pp. 184–192.

[20] R. Mojena, Hierarchical grouping methods and stopping rules: An evaluation, The Computer Journal 20 (1977) 359–363.

[21] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann, San Mateo, CA, 1993.

[22] V. Rohatgi, Statistical Inference, Willey, New York, 1984.

[23] C. Schaffer, Selecting a classification method by cross-validation, Machine Learning 13 (1993) 135–143.

[24] C. Schaffer, A conservation law for generalization performance, in: Proc. 11th International Conference on Machine Learning, New Brunswick, NJ, Morgan Kaufmann, San Mateo, CA, 1994, pp. 259–267.

[25] SPSS Manuals, Professional statistics 6.1, SPSS inc., 1994.

[26] M. Stone, Cross-validatory choice and assessment of statistical predictions, J. Roy. Statist. Soc. 36 (1974) 111–147.

[27] S. Weiss, C. Kulikowski, Computer Systems that Learn, Morgan Kaufmann, San Mateo, CA, 1991.

[28] Y. Yang, Sampling strategies and learning efficiency in text categorization, in: Proc. AAAI Spring Symposium on Machine Learning in Information Access, AAAI Press, 1996, pp. 88–95.