# NLC: A Measure Based on Projections*

Roberto Ruiz, José C. Riquelme, and Jesús S. Aguilar-Ruiz

Departamento de Lenguajes y Sistemas,
Universidad de Sevilla
Avda. Reina Mercedes S/N.
41012 Sevilla, España
{rruiz,riquelme,aguilar}@lsi.us.es

**Abstract.** In this paper, we propose a new feature selection criterion. It is based on the projections of data set elements onto each attribute. The main advantages are its speed and simplicity in the evaluation of the attributes. The measure allows features to be sorted in ascending order of importance in the definition of the class. In order to test the relevance of the new feature selection measure, we compare the results induced by several classifiers before and after applying the feature selection algorithms.

## 1 Introduction

The selection of relevant features is a central problem in machine learning. If a relevant feature is removed, the measure of the remaining features will deteriorate. In order to identify relevant attributes, we need to address what a good feature is for classification. Without defining the *goodness* of a feature or features, it does not make sense to talk about best or optimal features. The algorithms evaluate the attributes based on general characteristics of the data.

Feature Selection can be viewed as a search problem, where each state in the search space specifies a subset of the possible features. The need for evaluation is common to all search strategies.

In this paper, we propose a new feature selection criterion not based on calculated measures between attributes, or complex and costly distance calculations. This criterion is based on a unique value called NLC. It relates each attribute with the label used for classification. This value is calculated by projecting data set elements onto the respective axis of the attribute (ordering the examples by this attribute), then crossing the axis from the beginning to the greatest attribute value, and counting the Number of Label Changes (NLC) produced.

* This work has been supported by the Spanish Research Agency CICYT under grant TIC2001-1143-C03-02, and Junta Andalucia under coordinated action ACC-1021-TIC-2002.

V. Mařík et al. (Eds.): DEXA 2003, LNCS 2736, pp. 907–916, 2003.

## 2 Related Work

Feature selection algorithms use different evaluation functions. Functions are based in criterions to measure the relevance of the attributes. There are several taxonomies of these evaluation measures in previous work, depending on different criterions: Langley [9] group evaluation functions into two categories: filter and wrapper. Blum y Langley [3] provide a classification of evaluation functions into four groups, depending on the relation between the selection and the induction process: embedded, filter, wrapper, weight. Another different classification, Doak [5] and Dash [4] provide a classification of evaluation measure based on their general characteristics more then in the relation with the induction process. The classification realized by Dash, separate five different types of measures: distance, information, dependence, consistency y accuracy. Feature Selection can be viewed as a search problem, where each state in the search space specifies a subset of the possible features. The need for evaluation is common to all search strategies. In general, attribute selection algorithms perform a search through the space of feature subsets, and must address four basic issues affecting the nature of the search: 1) Starting point: forward and backward, according to whether it began with no features or with all features. 2) Search organization: exhaustive or heuristic search. 3) Evaluation strategy: wrapper or filter. 4) Stopping criterion: a feature selector must decide when to stop searching through the space of feature subsets. A predefined number of features are selected, a predefined number of iterations reached. Whether or not the addition or deletion of any feature produces a better subset, we also stop the search, if an optimal subset according to some evaluation function is obtained.

## 3 Feature Evaluation

### 3.1 Observations

To discover main idea of the algorithm we base on the data sets IRIS and WINE, because of the easy interpretation of their two-dimensional projections.

In Figure 1(a) it is possible to observe that if the projection of the examples is made on the ordinate axis we can not obtain intervals where any class is a majority. Nevertheless, for the Petalwidth attribute it is possible to appreciate some intervals where the class is unique: [0,0.6] for Setosa, [1.0,1.3] for Versicolor and [1.8,2.5] for Virginica. This is because when projecting the examples on this attribute the number of label changes is minimum. For example, it is possible to verify that for Petalwidth the first label change takes place for value 1 (setosa to Versicolor), the second in 1.3 (Versicolor to Virginica). There are other changes later and the last one is in 1.8.

In Figure 1(b) the same conclusion is reached with data set WINE. We analyze the projection of data set elements onto C8 and C7 attributes. We identify intervals where one class is a majority when crossing the abscissas axis from the beginning to the greatest attribute value: [0,1] for class 3, [1.5,2.3] for

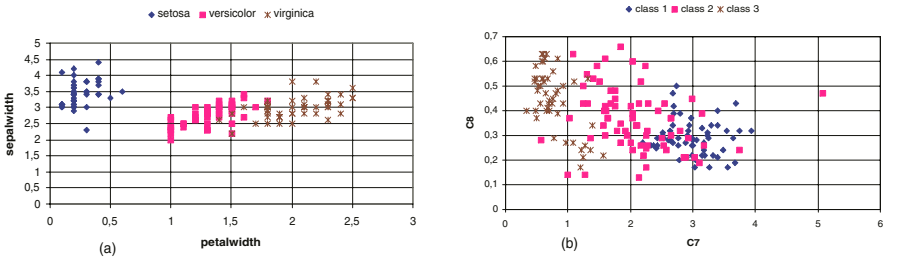


**Fig. 1.** (a) IRIS. Representation of Attributes *Sepalwidth-Petalwidth* (b) WINE. Representation of Attributes *C8-C7*

class 2 and [3,4] for class 1. Nevertheless, for C8 attribute on the ordinate axis it is not possible to observe any intervals where the class is unique.

We conclude that it will be easier classify by attributes with the smallest number of label changes. If the attributes are in ascending order according to the NLC, we obtain a ranking list with the better attributes from the point of view of the classification, in Iris this would be: Petalwidth 16, Petallength 19, Sepallenth 87 and Sepalwidth 120. This result agrees with what is common knowledge in data mining, which states that the width and length of petals are more important than those related to sepals.

Classifying IRIS with C4.5 by Sepalwidth only, we obtain 59% accuracy and by Petalwitdth 95%. The attributes used in Figure 1(b) are the first and the last on the ranked list, with a NLC value of 43 and 139 respectively. Applying the classifier C4.5, we obtain 80.34% accuracy by C7 and 47.75% by C8.

### 3.2 Definitions

**Definition 1:** An *example* e ∈ E is a tuple formed by the Cartesian product of the value sets of each attribute and the set C of labels. We define the operations *att* and *lab* to access the attribute and its label (or class): att: E x N → A and lab: E → C, where N is the set of natural numbers.

**Definition 2:** Let the *universe* U be a sequence of example from E. We will say that a database with n examples, each of them with m attributes and one class, forms a particular universe. Then U=<u[1],...,u[n]> and as the database is a sequence, the access to an example is achieved by means of its position. Likewise, the access to j-th attribute of the i-th example is made by att(u[i],j), and for identifying its label lab(u[i]).

**Definition 3:** An *ordered projected sequence* is a sequence formed by the projection of the universe onto the i-th attribute. This sequence is sorted out in ascending order.

**Definition 4:** A partition in *constant subsequences* is the set of subsequences formed from the ordered projected sequence of an attribute in such a way as to maintain the projection order. All the examples belonging to a subsequence have the same class and every two consecutive subsequences are disjointed with respect to the class.

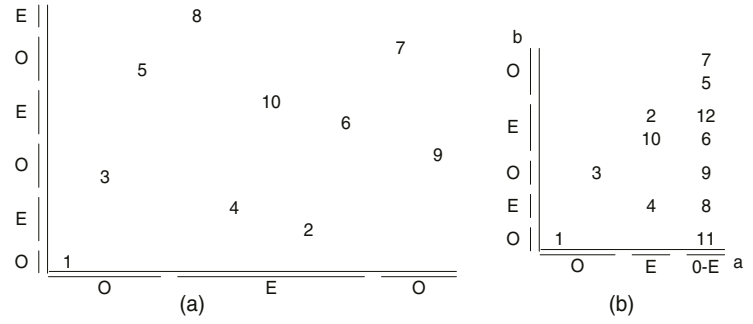


**Fig. 2.** Data set with (a) ten and (b) twelve elements and two classes

**Definition 5:** A *subsequence of the same value* is the sequence composed of the examples with identical value from the i-th attribute within the ordered projected sequence. This situation can be originated in continuous variables, and it will be the way to deal with the discrete variables.

**Definition 6:** two examples are *inconsistent* if they match except for the class label.

### 3.3 Description

The algorithm is based on this basic principle: to count the label changes of examples projected onto each feature. If the attributes are in ascending order according to the NLC we will have a list that defines the priority of selection, from greater to smaller importance.

Before formally exposing the algorithm, we will explain in more detail the main idea. Let us consider the situation depicted in Figure 2(a), with ten elements numbered and two labels (O-odd numbers and E-even numbers): the projection of the examples on the abscissas axis produces three constant subsequences {O,E,O} corresponding to the examples {[1,3,5][8,4,10,2,6][7,9]}. Identically, with the projection on the ordinates axis we can obtain six constant subsequences {O,E,O,E,O,E} formed by the examples {[1][2,4][3,9][6,10][5,7][8]}. We check that the first attribute has two label changes and the second one has five. Applying our hypothesis, the first attribute is more relevant than the second one, because it has a smaller NLC.

## 4 Algorithm

The algorithm is very simple and fast (Table 1). It has the capacity to operate with continuous and discrete variables as well as with databases which have two classes or multiple classes. For each attribute, the training-set is ordered (QuickSort [7], this algorithm is O(n log n), on average and we count the NLC throughout the ordered projected sequence.

**Table 1.** Main Algorithm

```
Input: E training (N examples, M attributes)
Output: E reduced (N examples, K attributes)
  for each attribute A_i ∈ 1..M
    QuickSort(E,i)
    NLC_i ← NumberChanges(E,i)
  NLC Attribute Ranking
  Select the k first
```

**Table 2.** NumberChanges function

```
Input: E training (N examples, M attributes), i
Output: number of label changes
 for each example e_j ∈ E with j in 1..N
    if att(u[j],i) ∈ subsequence of the same value
      changes = changes + ChangesSameValue()
    else
      if lab(u[j]) <> lastLabel)
        changes = changes + 1
 return(changes)
```

Applying the algorithm to the example of the Figure 2(b) we obtain the ordered projected sequences:

$$\{1, 3, 4, 10, 2, 11, 8, 9, 6, 12, 5, 7\}$$

$$\{1, 11, 4, 8, 3, 9, 10, 6, 2, 12, 5, 7\}$$

and the partitions:

$$\{[1, 3][4, 10, 2][11, 8, 9, 6, 12, 5, 7]\}$$

$$\{[1, 11][4, 8][3, 9][10, 6, 2, 12][5, 7]\}$$

The elements' projections onto the first attribute produce two constant subsequences and one subsequence of the same value with different labels. The elements' projections onto the second attribute produces five constant subsequences.

NumberChanges considers whether we deal with different values from an attribute, or with a subsequence of the same value (this situation can be originated in continuous and discrete variables). In the first case, it compares the present label with the last one. Whereas in the second case, where the subsequence is of the same value, it counts the maximum possible changes by means of the function *ChangesSameValue*.

In the attribute represented on the ordinates axis ($b$) in Figure 2(b), we see several subsequences of the same value with the same label, then, we deal with constant subsequence, and the result is four label changes (NLC=4). In the attribute on the abscissas axis ($a$), the first two partitions are constant

subsequences, and the third is a subsequence of the same value with two labels. Therefore, we consider the maximum possible NLC.

In the previous case, we have the subsequence [11,8,9,6,12,5,7] where four elements are class O (odd) y three class E (even). There are two reasons for counting the maximum NLC: first, we want to penalize the attribute in these inconsistency situations; and second, we want to avoid ambiguities that could be produced depending on the elements order after algorithm QuicSort is applied. For example, in different independent executions, we could obtain these situations: [E,E,E,O,O,O,O], [E,E,O,O,O,O,E], [O,O,E,E,O,O,E],... with NLC equal to 1, 2 and 3 respectively. *ChangesSameValue* returns 5, the maximum. The situation is: [O,E,O,E,O,E,O]. This can be obtained with low cost. It can be deduced counting the class' elements in the subsequence without resorting the elements.

We conclude that the attribute $b$ with four NLC is more relevant that the attribute $a$ with seven NLC.

## 5 Experiments

In this section we compare the quality of selected attributes by the NCL measure with the selected attributes by the other two methods: Information Gain (IG) [10] and the ReliefF method [8]. IG has been chosen because it is the more popular concept and it is used more when you want to evaluate the relevance of an attribute. And the ReliefF method has been chosen because it is widely referenced in other papers. The ReliefF method is a version of the Relief method by Kononenko, wich permits attributes with missing values and multiclass problems. The quality of each selected attribute was tested by means of three classifiers: the Naive Bayes [6], C4.5 [10] and 1-NN [1]. The implementation of the induction algorithms and the others selectors was done using the Weka library[1] and the comparison was performed with eighteen databases of the University from California Irvine [2]. The data sets were chosen with few missing values.

The process followed to test the quality of the attributes selected with the NCL measure was the following. For all original data sets, we obtained the accuracy using the three classifiers and the size of the decision trees induced by C4.5. We obtained the same measures after applying each selector algorithm, recording the number of attributes selected.

To asses the obtained results, two paired t statistical tests with a confidence level of 95% were realized

In order to establish the number of attributes in each case, we obtain a ranked list of features with the three method and we use the learning curve to observe the effect of added features. Starting with one feature (the most relevant one first) and gradually adding next most relevant feature one by one, we calculate its accuracy rate. We select the set of attributes with the best accuracy. Applying a different classifier, we obtain a different set.

[1] http://www.cs.waikato.ac.nz/ ml

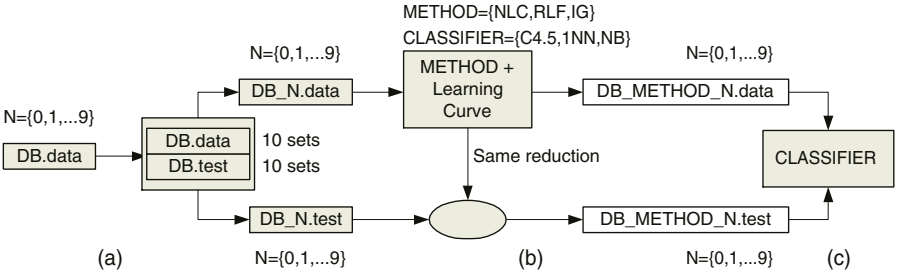


**Fig. 3.** Cross-validation process

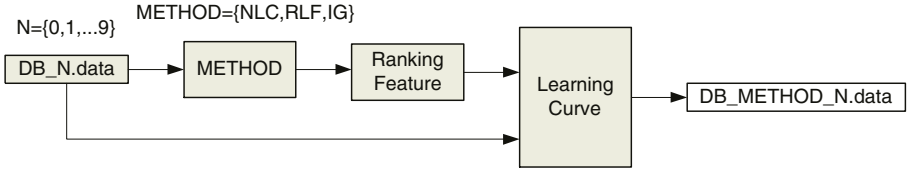


**Fig. 4.** Reduction method: feature ranking and learning curve

For each database (DB), the measures were estimated taking the mean of a ten fold cross validation. A ten-fold cross-validation is performed by dividing the data into ten blocks of cases that have an approximately similar size, and for each block in turn, testing the model constructed from the remaining nine blocks on the unseen cases in the hold-out block (Figure 3(a)). The same folds were used for each algorithm training-sets.

Each reducing method was given a training set (DB_N.data) consisting of 90% of the available data, from which it returned a subset DB_METHOD_N (Figure 3(b)), where METHOD is one of NLC, RLF, IG, N is a value in 0,1,...,9 and includes a classifier to obtain the learning curve (Figure 4). We use the same classifier that we are going to classify the test set. For example, from DB_1.data we would obtain DB_IG_1.data by applying the IG method. The remaining 10% of the unseen data (DB_N.test) was also reduced (DB_METHOD_N.test) (Figure 3(b)) and tested on the instances of DB_METHOD_N.data using a classifier (Figure 3(c)). For example, we obtain iris_ig_1.data by applying the IG method to the iris_1.data file generated by the cross validation. Afterwards, we use iris_ig_1.data to classify iris_ig_1.test by means of the nearest neighbor technique. When we deal with the learning curve, we also apply 1NN (Figure 4).

As a further comparison, another widely-used learner, C4.5 and NB, was run on these data sets (Figure 3(c)). For example, after reducing iris_3.data with NLC, iris_nlc_3.data was generated, it was given as input to C4.5 and the decision tree generated was used to classify the iris_nlc_3.test file (test files are reduced too).

If we consider all the possible results that we get using the original data, the three selection methods (NLC, RLF and IG) and the three classifiers (C4.5, 1NN and NB) with eighteen data sets taking the mean of a 10-fold cross validation,

**Table 3.** Accuracy obtained with *C4.5*, *1NN* and *naive Bayes*, selecting the subset with the best accuracy

| Data | C4.5 | | | | | | 1NN | | | | | | NB | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C4.5 | NLC | 1 | 2 | 3 | RLF | IG | 1NN | NLC | 1 | 2 | 3 | RLF | IG | NB | NLC | 1 | 2 | 3 | RLF | IG |
| anne | 98.6 | 98.4 | | | | 98.4 | 98.0 | 99.3 | 99.0 | | | | 98.9 | 98.9 | 86.3 | 89.2 | ○ | | • | 90.0 | 92.4 |
| bala | 78.4 | 78.4 | | | | 78.4 | 78.4 | 86.9 | 86.9 | | | | 86.9 | 86.9 | 88.8 | 88.8 | | | | 88.8 | 88.8 |
| germ | 71.1 | 73.8 | ○ | ○ | | 70.5 | 74.5 | 72.4 | 69.7 | | | | 70.8 | 71.0 | 74.8 | 75.5 | | | | 73.4 | 74.7 |
| diab | 76.7 | 75.1 | | | | 75.9 | 75.5 | 70.9 | 68.2 | | | | 66.7 | 68.5 | 76.2 | 75.6 | | | | 75.8 | 76.5 |
| glas | 69.2 | 70.1 | | | | 71.0 | 67.3 | 70.5 | 71.9 | • | | | 75.1 | 74.2 | 45.8 | 56.6 | ○ | ○ | | 47.7 | 51.0 |
| gla2 | 77.8 | 79.6 | | | | 79.2 | 79.0 | 79.2 | 78.3 | • | • | | 89.0 | 89.0 | 61.9 | 69.9 | ○ | ○ | | 64.4 | 69.9 |
| h-st | 78.5 | 83.7 | ○ | | | 80.4 | 85.2 | 74.4 | 79.3 | ○ | | • | 79.6 | 83.3 | 84.1 | 81.9 | | | • | 85.2 | 84.4 |
| iono | 88.6 | 89.7 | | • | | 92.6 | 91.2 | 86.6 | 87.2 | | • | | 91.4 | 89.2 | 83.2 | 84.6 | | | | 89.7 | 87.2 |
| iris | 94.0 | 92.7 | | | | 94.0 | 92.7 | 95.3 | 93.3 | | | | 93.3 | 93.3 | 95.3 | 92.7 | • | | | 94.0 | 92.7 |
| kr-v | 99.5 | 99.3 | | | | 99.5 | 99.5 | 96.5 | 97.4 | ○ | | ○ | 98.3 | 96.9 | 88.0 | 90.4 | ○ | | • | 94.0 | 90.4 |
| lymp | 78.3 | 74.9 | | | | 77.7 | 76.3 | 79.7 | 77.0 | | | | 83.0 | 77.8 | 83.8 | 83.8 | | | | 81.8 | 83.8 |
| segm | 97.0 | 96.9 | | | | 96.8 | 96.9 | 97.0 | 97.1 | | | | 97.1 | 97.1 | 80.0 | 87.2 | ○ | | | 87.2 | 87.2 |
| sona | 72.6 | 73.5 | | | | 75.0 | 72.2 | 85.5 | 84.6 | | | | 81.3 | 87.5 | 67.8 | 73.5 | | | | 74.9 | 73.5 |
| spli-2 | 94.4 | 94.0 | | | | 94.0 | 94.2 | 73.9 | 90.0 | ○ | | | 90.0 | 90.0 | 95.3 | 96.1 | | | | 96.3 | 95.8 |
| vehi | 73.5 | 72.6 | | | | 72.7 | 73.7 | 70.1 | 70.7 | | | | 71.2 | 69.3 | 44.3 | 44.3 | | | • | 48.5 | 44.3 |
| vowe | 82.9 | 81.7 | | | | 82.7 | 81.0 | 99.4 | 99.2 | | | | 99.0 | 99.2 | 66.1 | 68.7 | ○ | | | 69.2 | 69.1 |
| wave | 76.6 | 77.6 | | | | 78.1 | 77.4 | 73.8 | 79.1 | ○ | | | 78.2 | 79.1 | 80.0 | 80.7 | | | • | 81.4 | 80.7 |
| zoo | 93.1 | 94.1 | | | | 93.1 | 92.1 | 96.0 | 97.0 | | | | 95.0 | 95.0 | 95.0 | 95.0 | | | | 91.0 | 92.1 |

we get two hundred and eighteen results ((1+3) × 3 × 18 = 216). Now we are going to analyze these results to obtain some conclusions about the performance of the different methods.

Table 3 shows a summary of the results of the classification using C4.5, 1NN and NB. Table shows how often each method performs significantly better (denoted by ○) or worse (denoted by •) than data without reduction (column named 1), and better or worse than ReliefF (RLF) and Information Gain (IG) (column 2 and 3 respectively). Then, we obtain fifty four results comparing NLC with data without reduction (18 data sets × 3 classifiers = 54) and one hundred and eight comparing NLC with RLF and IG (18 data sets × 3 classifiers × 2 = 108). NLC measure is better than data without reduction in twelve of the fifty four cases, and in forty one are equal and only in one is worse than data without reduction. Furthermore, in four of the one hundred and eight cases, the set of attributes selected by the NLC measure yields better accuracy than the two other methods. In ninety three they are equal, and in eleven they are worse than the other.

We select the set of attributes with the best accuracy. Applying each classifier, we obtain a different set. Therefore, we get three set of attributes for each reduction method for each data set. We obtain the percentage of the original features retained and we calculate the average over the eighteen data sets (nine results). All the methods are between 50% and 60% of the original features.

The experiments show that by applying NLC, the knowledge attained in the original training file is conserved into the reduced training file, and the

**Table 4.** Accuracy obtained with *C4.5*, *1NN* and *naive Bayes*, selecting the three first attributes of the ranked list

| Data | C4.5 | | | | | 1NN | | | | | NB | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | C4.5 | NLC | 1 | 2 | RLF | IG | 1NN | NLC | 1 | 2 | RLF | IG | NB | NLC | 1 | 2 | RLF | IG |
| anneal | 98.6 | 89.9 | • | | 92.5 | 90.6 | 99.3 | 91.1 | | | 91.6 | 91.6 | 86.3 | 84.0 | • | • | 90.0 | 88.9 |
| balance | 78.4 | 69.4 | | | 69.4 | 69.4 | 86.9 | 68.0 | | | 69.6 | 67.7 | 88.8 | 74.2 | | | 73.6 | 73.3 |
| g_credit | 71.1 | 70.9 | | | 71.5 | 72.2 | 72.4 | 61.2 | • | • | 70.6 | 70.6 | 74.8 | 71.6 | | | 71.3 | 73.9 |
| diabetes | 76.7 | 74.6 | | | 74.7 | 75.1 | 70.9 | 69.8 | | | 66.7 | 69.3 | 76.2 | 77.1 | | | 76.4 | 76.8 |
| glass | 69.2 | 71.9 | | | 67.7 | 65.4 | 70.5 | 66.3 | | | 66.3 | 64.5 | 45.8 | 53.8 | ○ | | 45.8 | 49.1 |
| glass2 | 77.8 | 81.4 | | | 77.9 | 81.4 | 79.2 | 77.9 | | | 83.4 | 77.9 | 61.9 | 68.7 | | | 63.8 | 68.7 |
| heart-s | 78.5 | 72.6 | | • | 73.3 | 85.2 | 74.4 | 67.8 | • | • | 73.3 | 84.8 | 84.1 | 74.8 | | • | 74.8 | 79.6 |
| ionosphe | 88.6 | 80.4 | • | • | 88.3 | 90.0 | 86.6 | 84.3 | | | 85.7 | 88.6 | 83.2 | 78.3 | | • | 83.5 | 86.6 |
| iris | 94.0 | 94.0 | | | 94.0 | 94.0 | 95.3 | 95.3 | | | 95.3 | 95.3 | 95.3 | 95.3 | | | 95.3 | 95.3 |
| kr-vs | 99.5 | 90.4 | | | 90.4 | 90.4 | 96.5 | 90.4 | | | 90.4 | 90.4 | 88.0 | 90.4 | | | 90.4 | 90.4 |
| lymph | 78.3 | 77.7 | | | 79.7 | 77.7 | 79.7 | 75.7 | | | 83.7 | 75.0 | 83.8 | 75.0 | | | 80.3 | 72.3 |
| segment | 97.0 | 90.5 | ○ | ○ | 85.6 | 85.5 | 97.0 | 91.8 | ○ | ○ | 85.3 | 88.5 | 80.0 | 77.0 | ○ | ○ | 72.5 | 64.3 |
| sonar | 72.6 | 68.8 | | | 70.2 | 70.7 | 85.5 | 73.1 | | | 66.8 | 70.2 | 67.8 | 73.1 | | | 70.6 | 70.6 |
| splice-2 | 94.4 | 80.5 | | | 81.4 | 80.8 | 73.9 | 80.2 | | | 81.2 | 80.6 | 95.3 | 79.6 | | | 81.1 | 80.7 |
| vehicle | 73.5 | 53.5 | • | • | 62.4 | 61.4 | 70.1 | 53.7 | | | 56.0 | 57.7 | 44.3 | 40.4 | | | 42.8 | 40.8 |
| vowel | 82.9 | 69.1 | | | 70.6 | 72.1 | 99.4 | 79.6 | | | 80.1 | 82.8 | 66.1 | 57.9 | | | 56.0 | 58.7 |
| waveform | 76.6 | 66.2 | | | 64.5 | 65.3 | 73.8 | 57.0 | | | 56.3 | 56.4 | 80.0 | 65.1 | | | 66.1 | 64.9 |
| zoo | 93.1 | 84.2 | ○ | | 72.3 | 85.2 | 96.0 | 83.2 | ○ | | 71.3 | 87.2 | 95.0 | 84.2 | ○ | | 71.3 | 84.2 |

dimensionality of data is reduced significantly. We obtain similar results with the other method, but needing much more time.

It is very interesting to compare the speed of attribute selection techniques. We measured the time taken in milliseconds to select the ranking of attributes. NLC is an algorithm with a very short computation time. NLC takes 792 milliseconds in reducing 18 data sets whereas ReliefF takes 566 seconds and IG 2189 milliseconds. We obtain the percentage of reduction time of each data sets, and we calculate the average. NCL reduce the computational cost to the 99% of the time needed by ReliefF and 50% of the time needed by IG.

In order to compare the first attributes in the ranking list of each method, we obtain Table 4 where the data sets are reduced to the first three attributes of each ranking list. we observe the accuracy for each reduction method applying the three classifiers. We obtain similar results with the three methods. Table shows how often each method performs significantly better or worse than ReliefF (RLF) and Information Gain (IG) (column 1 and 2 respectively). In ten of the one hundred and eight cases, the set of attributes selected by the NLC measure yields better accuracy than the two other methods. In eighty they are equal, and in fourteen they are worse than the other.

## 6 Conclusions

In this paper we present a deterministic attribute selection criterion. The main advantages are its speed and simplicity in the evaluation of the attributes. The

measure allows features to be sorted in ascending order of relevance. A considerable reduction of the number of attributes is produced. It is not based on calculated measures between attributes, or complex and costly distance calculations. The computational cost is lower than other methods $O(m \times n \times \log n)$.

We conclude that by applying NLC, the knowledge attained in the original training file is conserved into the reduced training file, and the dimensionality of data is reduced significantly. We obtain similar results with the other method, but needing much more time.

## References

1. Aha, D., Kibler, D., & Albert, M. Instance-based learning algorithms. Machine Learning, 6, 37–66 (1991).
2. Blake, C., & Merz, E. K. Uci repository of machine learning databases (1998).
3. Blum, A., & Langley, P. Selection of relevant features and examples in machine learning. Artificial Intelligence, pp. 245–271 (1997).
4. Dash, M., & Liu, H. Feature selection for classification. Intelligent Data Analisys, 1 (1997).
5. Doak, J. An evaluation of search algorithms for feature selection. Technical Report, Los Alamos National Laboratory (1994).
6. Duda, R., & P. Hart. Pattern classification and scene analysis. John Willey and Sons (1973).
7. Hoare, C. A. R. Quicksort. Computer Journal, 5, 10–15 (1962).
8. Kononenko, I. Estimating attributes: Analysis and estensions of relief. European Conference on Machine Learning, pp. 171–182 (1994).
9. Langley, P. Selection of relevant features in machine learning. Procs. Of the AAAI Fall Symposium on Relevance, pp. 140–144 (1994).
10. Quinlan, J. Induction of decision trees. Machine Learning, 1, 81–106 (1986).