

Evaluation of Machine-Learning Algorithm Ranking Advisors

Helmut Berrer¹, Iain Paterson², and Jörg Keller¹

¹ DaimlerChryslerAG, Research & Technology, FT3/AD, P.O.-Box 2360,
D-89013 Ulm, Germany

`{helmut.berrer, joerg.keller}@daimlerchrysler.com`

² Institute of Advanced Studies (IHS), Departement of Economics and Finance,
Stumpergasse 56, A-1060 Vienna, Austria
`paterson@ihs.ac.at`

Abstract. Selecting the best-suited machine-learning classification algorithms for data-mining tasks is a meta-learning activity of the METAL project. Two alternative approaches for providing a user with an 'advisor' - a ranked preference list of algorithms - have been implemented. In this paper results of experiments designed to test thoroughly the consistency of these models are presented. After considering a wide-view of how different user-profiles arise, an impartial methodology for independently comparing ranking advice is presented, which also enables overall advisor performance under two specific user-preferences - advice-horizon and time-value of advice - to be distinguished.

1 Introduction

The user of an automated assistant system for Knowledge-Discovery and Data-Mining, which is currently being developed in the ESPRIT METAL project [10], requires advice as guidance for choosing the appropriate machine-learning (ML) algorithms to accomplish the data-mining process. To have model(s) selected as being best suited is a great boon for the inexperienced user with little or no experience of commercial software packages for data-mining. But this also offers a decision assistant for a user who may already have her own preferences of methods, due to her past experience, or who may simply have her choice of software limited by financial constraints. However, a useful starting point for a choice strategy is to obtain an independent assessment - referred to as a ranking advisor - that ranks algorithms for use on a particular task. In comparison to model selection a ranking delivers not only one most-preferred algorithm for the task in hand, it delivers an ordered list of algorithms, sorted from best first to worst last. The user may of course decide not to follow the advice exactly as given. For, besides accuracy, and training and testing time, other criteria can be important for the user, e.g. level of complexity and understandability. Nakhaeizadeh et al. [11] suggested in their approach the personalisation of an evaluation of data-mining algorithms whereby criteria are quantified in the form "criterion a is x times as important as criterion b ". In this paper we consider

how user-profiles may principally differ, but we concentrate our efforts on operationalisable user-preferences that are directly related to the data made available in METAL.

In METAL two main approaches to ranking classification algorithms have been made. Brazdil et al. [2] developed a heuristic, the adjusted ratio of ratios (ARR) ranking method, based on relevant performance information, accuracy attained and time spent achieving it. We refer to this as *Ranker0*. DaimlerChrysler (DC) has developed *Ranker1* for *multi-criteria-ranking* in meta-learning, utilising data envelopment analysis (DEA), which is commonly used in economics and operations research. In DEA ranking is achieved by means of efficiency measurement, by a linear programming procedure that calculates a discrete piecewise best-practice frontier. This frontier is represented by efficient Decision Making Units (DMUs). Each DMU lies on or below the calculated efficient frontier (Charnes et al. [4]). The application of DEA as a multi-criteria metric for evaluation of data-mining algorithms was originally tested by Nakhaeizadeh et al. [11], and Jammerneegg et al. [8]. A ranking is achieved in DEA by comparing the efficiency of the DMUs. In the terminology of METAL, each DMU is a data-mining algorithm that has positive (i.e. ‘more’ means ‘better’) or negative (i.e. ‘more’ means ‘worse’) properties, also referred to as output and input components. DC implemented two software models in an integrated concept for multi-criteria-ranking of data-mining algorithms (Keller et al. [9]). These ‘*universal models*’ of Paterson [12], implemented in Ranker1, extend the idea of measuring ‘superefficiency’¹ to the more generally useful case of variable returns-to-scales (VRS), thus enabling the ranking of efficient as well as inefficient units. These models also have strong invariance properties.

1. Universal Radial: This model is unit and translation invariant for the VRS specification: input or output data may thus assume negative or zero values. It is units invariant for non-negative data in the constant return-to-scales (CRS) case.
2. Universal Additive: This model is unit and translation invariant for the VRS specification: input or output data may thus assume negative or zero values. It is units invariant for non-negative data in CRS. Unlike the universal radial model, it exhibits discontinuity in the (super)efficiency values along the weak efficient boundary; advantageous, however, is that the efficiency score includes all slacks.

For both of these models a method for normalising the efficiency scores was developed, so that inefficient units obtain an ‘inefficiency’ value less than one, weak efficient units obtain a value of one, and superefficient units obtain a value greater than one.

To establish a fair basis for comparison of the ranking advice obtained from Ranker0 and Ranker1 is an obviously desirable objective. Some ideas have been put forward: Brazdil et al. [3] and Soares et al. [15]. In this paper we present in section 4 an overall methodology for comparing advisors that can be tested

¹ First formulated by Anderson and Petersen [1].

independently of the ranking concept and which can be adjusted according to some specific user-preferences. First, however, we examine in the next section the internal consistency of the integrated nearest-neighbour (k-NN) approach to ranking using Ranker0 and Ranker1, and we consider in a wider perspective how user-profiles in the automotive industry may differ from each other (section 3).

2 Internal Consistency of the k-NN Approach

Keller et al. [9] described an integrated concept for Multi-Criteria-Ranking. No search or learning algorithm can be the best on all possible learning or optimization problems (‘no free lunch’ theorem by Wolpert et al. [16]), but it should be possible to pick out the best suited ML algorithm for a specified data set. The basic idea was that for similar data sets algorithms should behave similarly. To verify the *internal consistency* of the k-NN approach, the ‘ideal ranking’ (based on the information of the test data set and the assumption that the used ranking advisor delivers the natural/true ranking.) and the ‘recommended ranking’ (based on the information of the k-NN data sets) should be near in terms of the Spearman rank correlation coefficient for any ranking advisor.

Each data set can be described by statistical properties such as skewness, kurtosis etc. and other measures of information content, as well as basic features such as the number and kind of attributes. This ‘Metadata 1’ (MD1) is used in a 3 nearest neighbour² algorithm. The distance function $Dis(DS_j, DS_k)$ between two data sets (j and k) was influenced by the considerations of Friedman [7] and Dasarathy [6]:

$$Dis(DS_j, DS_k) = \left(\sum_{i=1}^m weight(i) \star \left| \frac{DS_j(i) - DS_k(i)}{Stdev(i)} \right|^{exponent(i)} \right)^{\frac{1}{2}}. \quad (1)$$

Each data set is described by m MD1 attributes, and in the default version equal weights and exponents(=2) are implemented. $Stdev(i)$ is the standard deviation of the i -th attribute over all data sets, and $DS_j(i)$ is the i -th MD1 attribute value of data set j .

The performance of every ML algorithm is measured by the ‘Metadata 2’ (MD2) variables, and some or all of these variables may be used to determine a ranking:

- Accuracy Rate during the Learning Phase (i.e. application to some portion of the data set)
- Accuracy Rate during the Test Phase (i.e. application to the rest of the data set)
- Duration of the Learning Phase
- Computer ‘space’ used during the Learning Phase resource requirement

² The parameter 3 for the k-NN algorithm was arbitrary chosen, and the choice of this variable is the subject of further investigations.

- Duration of the Test Phase
- Computer ‘space’ used during the Test Phase
- Hypsize the average model size for learning an algorithm

As shown in Fig. 1, a script is implemented that processes each data set in the database step by step. First a k-NN program determines from the specified MD1 attributes the 3 nearest neighbour data sets. The ranking algorithm provides an ‘ideal ranking’ on the test data set based on the specified Metadata 2.

The recommended ranking is derived by using the same MD2 variables of the 3 nearest neighbor data sets. In case of the Ranker1 an efficiency analysis of the ML algorithms is made by singly for each of the 3-NN data sets. To receive the recommended ranking based on the efficiency values, we consider two ways to combine the 3 individual results:

1. The combined efficiency value $eff(3 - NN)_j$ of an algorithm j is calculated by the arithmetic mean of the 3 results $eff(i)_j \ \forall i = 1, \dots, 3$, obtained from the analysis.

$$eff(3 - NN)_j = \sum_{i=1}^3 \frac{1}{3} * eff(i)_j . \quad (2)$$

2. The weighted efficiency value $eff(3w - NN)_j$ of an algorithm j is calculated by the weighted mean of the 3 results, obtained from the individual analysis.

$$eff(3w - NN)_j = \sum_{i=1}^3 weight(i) * eff(i)_j . \quad (3)$$

where the (standardized) weights are calculated as the inverse proportion of the k - NN distances from each data set i to the test data set.³

$$weight(i) = \frac{1/Dis(DS_i, DS_{Test})}{\sum_{i=1}^3 1/Dis(DS_i, DS_{Test})} . \quad (4)$$

As shown in Ray et al. [13] there exists many ways to evaluate the conflicts in ranking of alternatives. Here, as in Soares [14], the difference between recommended and ideal ranking was calculated by the Spearman rank correlation coefficient⁴ between two rankings R_1, R_2 .

$$S(R_{1i}, R_{2i}) = \frac{\frac{1}{n} \sum_{i=1}^n (R_{1i} - \bar{R}_1)(R_{2i} - \bar{R}_2)}{\sqrt{\frac{1}{n} \sum_{i=1}^n (R_{1i} - \bar{R}_1)^2 \frac{1}{n} \sum_{i=1}^n (R_{2i} - \bar{R}_2)^2}} . \quad (5)$$

R_{1i} is the rank of algorithm i possessed in ranking 1. \bar{R}_1 is the mean of all ranks.

The internal consistency of the k-NN approach is measured by this coefficient ($\in [-1, 1]$). Consistently high values between ideal and recommended rankings for an advisor indicate that the similarity of the data sets is well described by the k-NN based on the chosen MD1 attribute set, when processed by this ranker.

³ As long as the distances are non-zero.

⁴ A special case of the Pearson correlation coefficient for rankings.

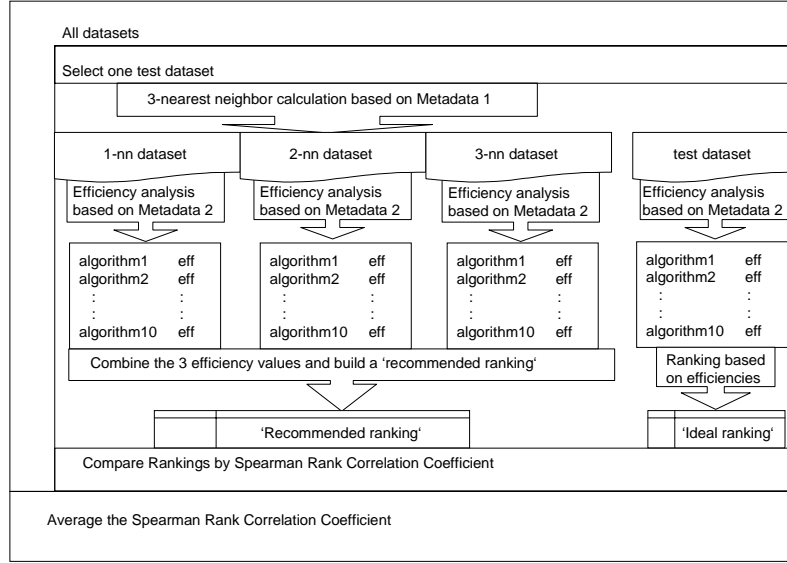


Fig. 1. Flowchart of the evaluation for the ‘recommended ranking’. The ‘ideal ranking’ on the Metadata 2 of the ‘test data set’ is compared to the ‘recommended ranking’ on the Metadata 2 derived by the data of 3 nearest neighbor data sets using Metadata 1. This procedure is repeated for every data set in the database

2.1 Experiments

Due to the fact that different ranking algorithms handle missing values in different ways, use has been made of two partial databases as well as the whole database. The detailed data were provided as results of extensive tests carried out by the METAL [10] consortium partners on ML algorithms. The databases are as follows:

- all: All data sets are used (58 data sets).
- clean10: Only the data sets with no missing values in the MD2 for all 10 algorithms are used (38 data sets).
- clean8: Due to the fact that two algorithms (clemRBFN, clemMLP) possess many missing values, this database contains all data sets with no missing values in the MD2 for all other (c50boost, c50rules, c50tree, lindiscr, ltree, mlcib1, mlcnb, ripper) algorithms (52 data sets).

To illustrate the influence of the MD1 attribute choice in the k-NN procedure, two different attribute sets⁵ are considered:

- k-NNa: Number of symbolic attributes, number of numerical attributes, number of tuples in data set, number of classes.

⁵ Equal weights and an exponential parameter of 2 are used for the distance formula.

- k-NNb: Number of symbolic attributes, number of numeric attributes, number of tuples in data set, number of classes, accuracy of default class, missing values, standard deviation of classes, tuples with missing values, mean skewness, mean kurtosis, attributes with outliers, result of M-statistic, degrees of freedom, Chi-square-distribution M-statistic, result of Sd-Ratio, Fract1, Cancor1, result of Wilks lambda, result of Bartlett statistic, Chi-square-distribution V-statistic, number of discriminant functions, class entropy, entropy of attributes, mutual information, joint entropy, equivalent number of attributes.

In the first case only basic features are chosen, in the second case all possible attributes are used in the k-NN process.

Tables 2 - 5 show the average Spearman rank correlation coefficient for the different databases. Ranker1 was executed in additive and radial formulations. A non-parametric statistical test⁶ showed that there is no significant difference in the correlation coefficients obtained. Ranker1 is able to include several inputs as well as several outputs (Acc=Accuracy, Hyp=HypSize, Test=Testtime, Total=Totaltime and Train=Traintime). For Tables 2 and 4, '3-NN' and '3w-NN' describe the above mentioned different ways to combine the individual efficiency values (Eq. 2 and 3). For Tables 3 and 5 '3-NN' and 'cross' describe two different 'recommended ranking' evaluations. In the first case only the MD2 of the 3 nearest neighbor data sets are used, in the second case all MD2 except the one of the test data set are used.

For Ranker0 the average Spearman rank correlation coefficients are always larger in the 3-NN case, which indicates that the k-NN procedures improve the consistency.

For Ranker1 the average Spearman rank correlation coefficients are generally larger in the 3w-NN case than in the 3-NN case, which indicates that the weighting procedures improve the consistency. Further, a look through the results in different rows shows that some specifications achieve a higher consistency than in the Accuracy/Totaltime case. In particular, the two input (Traintime, Testtime) and two output (Accuracy, HypSize) case shows the highest consistency of all in 10 out of 12 settings. This indicates that there are indeed gains to be made from utilising fully the multi-criteria features of Ranker1.

The average Spearman rank correlation coefficients are larger (statistically significant) for the k-NNa than for the k-NNb sets of results, which indicates that the MD1-attribute choice in the first case was better than in the second case.

To find out the 'best' MD1 attribute set and the appropriate parameter set will be the subject of further investigation.

⁶ Wilcoxon signed rank test.

3 User Profiles in Automotive Data-Mining: an Aside

For reasons of objectivity and efficiency, machine learning methods will also be used in automotive data engineering to obtain knowledge discovery and data-mining solutions. Design rules and knowledge are to be extracted from voluminous data bases. In evaluating industrial data-mining solutions a user of a meta-learning assistant will likely have personal preferences, e.g. concerning (a) time consumption; (b) accuracy; (c) complexity of model and (d) understandability. These criteria may be selected separately or in combination. DaimlerChrysler has defined a few typical user profiles, which result from a two-way classification of potential users according to their particular knowledge of machine learning and data-mining on the one hand (1), and their organisational function on the other (2). Thus user types have been discerned as in Tab. 1:

- (1A) Expert in Machine Learning and Data-Mining
- (1B) Non-Expert in Machine Learning and Data-Mining
- (2A) Scientist, researcher or developer
- (2B) Applicant in business, e.g. marketing, production, sales

and attached to the criteria (a)-(d) are preference weightings, e.g. very important (++), important (+), even (0), less (-) and never important (- -). By means of a questionnaire the following portfolio of DC user profiles was obtained from survey data:

	(1A) Expert	(1B) Non-Expert
(2A) Scientist	a/0 b/++ c/++ d/+	a/- - b/++ c/++ d/0
(2B) Applicant business	a/- b/++ c/0 d/0	a/+ b/++ c/+ d/++

Table 1. User Profiles

These user profile preferences also imply information about bounds on tolerance of factors too. A business applicant tolerates a maximum time consumption, or a scientist/researcher expects a minimum accuracy. Therefore ideally a multi-criteria ranking technique is needed, which complies with such demands. Bounds on output and input variables is consequently being implemented as a feature of Ranker1.

4 A Methodology for Evaluating Ranked Algorithm Advisors taking into account User Preferences

The aim of the METAL project is, briefly, to provide advice to potential users of data mining techniques as to which machine learning classification algorithms may be best suited to their particular task in hand. This advice is based on accumulated knowledge of the performance of various algorithms on known data sets. The advice is to be made readily available to the user by executing a script over the user data set. In its basic form the advice provides a ranking of algorithms, in most-preferred order. There are already at least two such alternative ‘advisors’, namely Ranker0 from the University of Porto, and Ranker1 from DaimlerChrysler/IHS. As mentioned previously, Ranker0 has been developed specially for the case where the performance of algorithms is described by one output - viz. *accuracy* - and one input - *time*. Ranker1, by contrast, is not limited in the number of outputs and/or inputs that may be handled. However, in order to establish the characteristics of these two rankers, it behoves us first of all to make a direct comparison of the compatibility of the advice they offer in identical situations. This section, then, outlines a methodology for comparing advisors derived from accuracy/time measurements. The term ‘methodology’ is used here instead of ‘method’ because any proper evaluation has to be predicated by the notion that users are not identical, and will express different preferences regarding their choice of ranker. Although, *a-priori*, these preferences are unknown, we adopt here the stance that the range of some of the most obvious preferences may be anticipated. Thus we wish to present results of evaluating advisors that indicates their sensitivity over a range of preference parameters.

Let us consider that an advisor is a list of (at present up to 10) algorithms, ranked by degree of recommendation, the most-highly recommended for use on a particular data set D coming first. The recommendation has been based on examining the known performance of each algorithm, in terms of its accuracy a obtained in time t on data sets *other than* D . Ranker1, for example, utilises such information from the three nearest-neighbour data sets. We wish now to test the efficacy of following such advice on data set D . For the purposes of experimental testing, we use known information on the performance of algorithms on D : algorithm i performs with accuracy a_i in time t_i , where $i = 1, \dots, 10$ is the ranked order of recommendation. It is important to note that the advisors (i.e. the order of algorithms proposed) were not obtained with knowledge of the a_i and t_i of data set D . In a real application, the user does not know a_i and t_i , nor are we able to make useful predictions of a_i and t_i for the particular data set D .

In section 2.1 of this paper we applied experimentally the ranking methods, Ranker0 and Ranker1, to the a_i and t_i data of data set D . This has yielded insights into the usefulness of the ranking method, *in terms of its internal consistency*. It shows, for example, how good the assumption of using nearest neighbours to generate a ranking is. It *does not*, however, allow us to make a direct comparison between Ranker0 and Ranker1 (or indeed any other possible ranker) because there is no independent measure of effectiveness.

We now introduce an independent measure of effectiveness, as follows. We assume that the user is interested in knowing which algorithms will yield the highest accuracy on her data set, D , and that she is willing to have this information within the first n algorithms suggested by her advisor. n may vary between 1 and 10, and is a parameter of *user-preference* that indicates how much ranking information the user is willing to take into account (the ‘*advice-horizon*’). We may imagine hypothetically that the user could perform trials on the first n algorithms listed in the advisor sequentially on her computer, and would obtain results, which are the n corresponding a_i and t_i values. We start with the case where $n = 10$. In Fig. 2 the vertical lines represent the accuracies a_i which are known at time T_i , where

$$T_i = \sum_{j=1}^i t_j . \quad (6)$$

is the cumulative lapsed time.

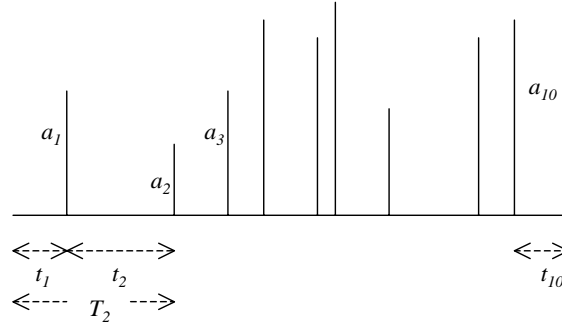


Fig. 2. Accuracy information on the time-axis.

Now the average of the ten time intervals (or ‘norm interval’) between the trials is

$$\bar{T} = T_{10}/10 . \quad (7)$$

We get information about the accuracy of algorithms at the end of each interval. So accuracy information is cumulated over time; information about an algorithm with higher accuracy is more important to the user than information about an algorithm which yields lower accuracy. However, we may regard information as ‘losing value’ the later we receive it. The situation here is somewhat analogous to a stream of cash-flows, where the present value of a future return is diminished according to the appropriate ‘time-value of money’, by discounting the cash-flow by a discount rate. Let us choose a norm discount rate of r for the norm interval. Then we define the *Discounted Information Flow* (DIF) for this discount rate as:

$$\text{DIF} = \sum_{i=1}^{10} a_i / (1 + r)^{T_i / \overline{T}} . \quad (8)$$

We may take different discount rates per norm interval, e.g. of 5 %, 10 %, ..., or even 0 % with corresponding $r = 0.05, 0.1, \dots$, and 0. Varying r indicates the sensitivity of the evaluation to this parameter. It is thus also indicative of a *user-preference* - the ‘*time-value of information*’. Low r indicates that the worth of information changes little over time, in comparison to the decrease in worth of information over time represented by higher values of r . The fact that time intervals t_i vary in length is taken into account by the exponent

$$T_i / \overline{T} . \quad (9)$$

In general, the DIF that we calculate will depend both on the user preferences for n and r . We have

$$\text{DIF}(n, r) = \sum_{i=1}^n a_i / (1 + r)^{T_i / \overline{T}} . \quad (10)$$

As mentioned above, in the real application situation, a_i and t_i are unknown and hence the ranking with the highest DIF may not be calculated in advance; likewise this function (more accurately, its inverse) is unsuitable for devising an *ex-ante* ranking procedure, as it is not clear how the measure would be aggregated over, say, 3 data sets, and, furthermore, values for n and r would need to be chosen arbitrarily. DIF, however, offers a methodology for evaluating ranking advisors *ex-post*. The value of accuracy information obtained from applying an advisor to a test data set is calculated for an information discount rate, r and an information horizon, n . The relative benefits of advisors such as Ranker0 and Ranker1 may be assessed for different user preferences, i.e. different values of discount rate and horizon. It has the added advantage of being entirely independent of these two rankers. Which of these rankers offers ‘better’ advice (in terms of DIF) for which set of user preferences is not known, *a-priori*, and will be the subject of further investigation.

Acknowledgements

The authors would like to thank Reza Nakhaeizadeh, Carlos Soares and Pavel Brazdil for fruitful discussions. The research was supported financially by the EC METAL project (ESPRIT #26.357) and DaimlerChrysler.

References

1. Anderson, P. and Petersen, N.C. (1993): A Procedure for Ranking Efficient Units in Data Envelopment Analysis, Management Science, Vol. 39, No. 10, pp.1261-1264

2. Brazdil, P. and Soares, C.: Ranking Classification Algorithms Based on Relevant Performance Information. In J. Keller and C. Giraud-Carrier, editors, *Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination*, 2000
3. Brazdil, P. and Soares, C.: A Comparison of Ranking Methods for Classification Algorithms Selection. In *Proceedings of the European Conference on Machine Learning ECML 2000*, 2000
4. Charnes, A.; Cooper, W. and Rhodes, E. (1978): Measuring the efficiency of decision making units, *European Journal of Operational Research* 2, pp. 429-444
5. Cook, W.; Kress, M. and Seiford, L.: Theory and Methodology, A general framework for distance-based consensus in ordinal ranking models, *European Journal of Operational Research* 96 (1996) 392-397
6. Dasarathy B. V.: *Nearest Neighbor (NN) Norms: NN Pattern Classification Techniques*, IEEE Computer Society Press, Los Alamitos, California, 1991
7. Friedman, J.H.: *Flexible Metric Nearest Neighbor Classification*, November 11, 1994
8. Jammerneegg, W.; Luptacik, M.; Nakhaeizadeh, G. and Schnabel, A. (1998): Ist ein fairer Vergleich von Data-Mining Algorithmen möglich? (in German, English title: A fair comparison of Data-Mining algorithms possible?) eds.: Nakhaeizadeh, G (ed.). *Data-Mining. Theoretische Aspekte und Anwendungen* (in German, English title: *Data-Mining: Theoretical Aspects and Applications*) pp 225-240, Physica Verlag
9. Keller, J.; Paterson, I. and Berrer, H.: An Integrated Concept for Multi-Criteria-Ranking of Data-Mining Algorithms. 11th European Conference on Machine Learning WS: *Meta-Learning: Building Automatic Advice Strategies for Model Selection and Method Combination* Barcelona, Catalonia (Spain) May 30th, 2000
10. EC ESPRIT METAL project # 26.357 (1998 - 2001): Partners - University of Bristol, Geneva, Vienna and Porto; DaimlerChrysler and Integral Solutions Limited; <http://www.cs.bris.ac.uk/~cgc/METAL/>
11. Nakhaeizadeh, G. and Schnabel, A. (1997): Development of Multi-Criteria Metrics for Evaluation of data-Mining Algorithms, *Third International Conference on Knowledge Discovery and Data-Mining, Proceedings*, Newport Beach, California, August 14-17, (1997), pp.37-42
12. Paterson, I.: New Models for Data Envelopment Analysis, *Measuring Efficiency Outwith the VRS Frontier*, Economics Series No. 84, July 2000, Institute for Advanced Studies, Vienna
13. Ray, T. and Triantaphyllou, E.: Procedures for the evaluation of conflicts in ranking of alternatives, *Computers & Industrial Engineering* 36 (1999) 35-44
14. Soares, C.: *Ranking Classification Algorithms on Past Performance*, Faculdade de Economia, Universidade do Porto, PhD Thesis, 1999
15. Soares, C.; Costa, J. and Brazdil, P.: A Simple and Intuitive Measure for Multicriteria Evaluation of Classification Algorithms
16. Wolpert D. and Macready W.: No free lunch theorems for search. Technical Report SFI-TR-95-02-010, The Santa Fe Institute, 1996

Annex

The following tables 2-5 are referred to in the main text.

Table 2. k-NNa: Average Spearman rank correlation coefficient for Ranker1

k-NNa			Ranker1			
			additiv		radial	
Input	Output	Data	3-NN	3w-NN	3-NN	3w-NN
Total	Acc	all	0.6694	0.6791	0.6407	0.6811
Total	Acc, Hyp	all	0.7691	0.7810	0.7404	0.7472
Total, Hyp	Acc	all	0.6015	0.6250	0.4473	0.4956
Train, Test	Acc	all	0.6825	0.7100	0.6414	0.6680
Train, Test	Acc, Hyp	all	0.7565	0.7619	0.7492	0.7612
Train, Test, Hyp	Acc	all	0.5961	0.6294	0.5494	0.5761
Total	Acc	clean10	0.6826	0.6970	0.6967	0.7509
Total	Acc, Hyp	clean10	0.8252	0.8297	0.8278	0.8383
Total, Hyp	Acc	clean10	0.6329	0.6523	0.5143	0.5590
Train, Test	Acc	clean10	0.7021	0.7177	0.6485	0.6711
Train, Test	Acc, Hyp	clean10	0.8396	0.8459	0.8223	0.8376
Train, Test, Hyp	Acc	clean10	0.6258	0.6504	0.5446	0.5526
Total	Acc	clean8	0.6886	0.7083	0.6650	0.6826
Total	Acc, Hyp	clean8	0.7326	0.7440	0.7797	0.7901
Total, Hyp	Acc	clean8	0.5650	0.5893	0.5246	0.5621
Train, Test	Acc	clean8	0.6561	0.6685	0.6125	0.6545
Train, Test	Acc, Hyp	clean8	0.8310	0.8269	0.8017	0.8278
Train, Test, Hyp	Acc	clean8	0.5604	0.5627	0.5429	0.5731

Table 3. k-NNa: Average Spearman rank correlation coefficient for Ranker0

k-NNa			Ranker0	
Input	Output	Data	cross	3-NN
Total	Acc	all	0.6305	0.6591
Total	Acc	clean10	0.7662	0.7691
Total	Acc	clean8	0.6529	0.6891

Table 4. k-NNb: Average Spearman rank correlation coefficient for Ranker1

k-NNb			Ranker1			
			additiv		radial	
Input	Output	Data	3-NN	3w-NN	3-NN	3w-NN
Total	Acc	all	0.6840	0.6833	0.6390	0.6538
Total	Acc, Hyp	all	0.7412	0.7468	0.7213	0.7247
Total, Hyp	Acc	all	0.5952	0.6107	0.4535	0.4694
Train, Test	Acc, Hyp	all	0.7455	0.7574	0.7371	0.7369
Train, Test	Acc	all	0.6714	0.6731	0.6425	0.6381
Train, Test, Hyp	Acc	all	0.5892	0.6086	0.4956	0.5062
Total	Acc	clean10	0.6628	0.6689	0.6639	0.7095
Total	Acc, Hyp	clean10	0.7904	0.7911	0.8096	0.8246
Total, Hyp	Acc	clean10	0.6443	0.6689	0.5149	0.5317
Train, Test	Acc, Hyp	clean10	0.8112	0.8319	0.8179	0.8341
Train, Test	Acc	clean10	0.6813	0.6954	0.6414	0.6602
Train, Test, Hyp	Acc	clean10	0.6348	0.6475	0.5331	0.5455
Total	Acc	clean8	0.6886	0.7005	0.6735	0.6835
Total	Acc, Hyp	clean8	0.7349	0.7376	0.7714	0.7755
Total, Hyp	Acc	clean8	0.5723	0.5755	0.5251	0.5630
Train, Test	Acc, Hyp	clean8	0.7990	0.8068	0.7875	0.8022
Train, Test	Acc	clean8	0.6676	0.6630	0.6517	0.6662
Train, Test, Hyp	Acc	clean8	0.5513	0.5435	0.5338	0.5571

Table 5. k-NNb: Average Spearman rank correlation coefficient for Ranker0

k-NNb			Ranker0	
Input	Output	Data	cross	3-NN
Total	Acc	all	0.6305	0.7162
Total	Acc	clean10	0.7662	0.8016
Total	Acc	clean8	0.6529	0.7514