A Statistical Method for Determining Importance of Variables in an Information System

Witold R. Rudnicki[1], Marcin Kierczak[2], Jacek Koronacki[3], and Jan Komorowski[1,2]

[1] ICM, Warsaw University, Pawinskiego 5a, Warsaw Poland
W.Rudnicki@icm.edu.pl
http://www.icm.edu.pl/~rudnicki/
[2] Uppsala University, The Linnaeus Centre for Bioinformatics Husargatan 3, Uppsala Sweden
[3] Institute of Computer Science, Polish Academy of Sciences, J.K. Ordona 21, Warsaw, Poland

Abstract. A new method for estimation of attributes' importance for supervised classification, based on the random forest approach, is presented. Essentially, an iterative scheme is applied, with each step consisting of several runs of the random forest program. Each run is performed on a suitably modified data set: values of each attribute found unimportant at earlier steps are randomly permuted between objects. At each step, apparent importance of an attribute is calculated and the attribute is declared unimportant if its importance is not uniformly better than that of the attributes earlier found unimportant. The procedure is repeated until only attributes scoring better than the randomized ones are retained. Statistical significance of the results so obtained is verified. This method has been applied to 12 data sets of biological origin. The method was shown to be more reliable than that based on standard application of a random forest to assess attributes' importance.

1 Introduction

Application of computer programs to decision support or classification of data dates back to the 1970's. Such problems can be formally presented in the form of a decision system which consists of a set of objects O, each of the objects being described by P different attributes, $X_1, X_2, \ldots, X_P$, and a decision attribute Y not equal to any of the attributes whose value may be unknown.

An expert system or, more narrowly, a classifier can be defined as a function $F(X_1, \ldots, X_P) \to Y$.

The first generation of expert systems was designed by human experts whose knowledge was explicitly coded by the *if* **A** *then* **B** rules [1]. The systems could cope with all examples whose decision attribute value could have been predicted by the experts, but were unable to cope with new (unseen) examples with properties not earlier predicted by the experts. Applicability of such systems

S. Greco et al. (Eds.): RSCTC 2006, LNAI 4259, pp. 557–566, 2006.

was limited to simple cases with a small number of attributes. Multi-dimensional problems with complex properties based on uncertain data could not be handled by this rational approach. For a detailed discussion see [1].

Starting in the 1980-ies, machine learning and statistical methods, such as neural networks,decision trees, and many others, were popularized with the aim to address several of the limitations of expert systems [2,3,4,5]. These methods, often collectively called inductive learning, generate models from examples with known Y, which can then be applied to unseen cases. They can successfully cope with high-dimensional problems, albeit the achieved improvement comes with a price.

While the first generation expert systems were hardly tractable for anything but simple and small domains, the newer methods are often even less amenable to human understanding. Some of them are more or less like a black box, where one throws in the description of the object and by a process that is hidden to human inspection the outcome decision comes out automatically. Neural networks are most notorious in this respect, but even several rule-based methods, such as Bayesian networks [6] and in certain circumstances methods based on rough set theory [7,8,9,10], generate decision functions which are not easy to analyse. Such functions may be comprised of thousands, if not hundreds of thousands, of simple rules which are connected by complex, non-linear logical relations. Although the problems that are very complex are often likely to be described by very complex models, we should not give up the possibility of gaining insight into the structure of the generated model. There exist several approaches to obtaining a better legibility of the models. One well-known idea is to make models less exact (cf. Ziarko's approach and approximate attributes in rough sets), which avoids over-fitting and generates simpler models with often higher performance of the model on the unseen examples. A similar idea is to use dynamic reducts that sample the space of examples and allow finding the most important attributes. Another approach to obtaining legible models from large rule sets is rule tuning (see e.g. Ågotnes *et al* [11,12] which often provides a very significant reduction of the cardinality of the rule set and, sometimes, an improvement of the classification quality due to a generalization algorithm used in rule tuning. Yet another approach is the use of templates to discover local models [13,14]. Finally, in the rough set model approach of Kowalczyk, a small subset of attributes is selected using various heuristics and user knowledge to generate simple models.

Unfortunately, for problems with a very high dimension where domain knowledge is not yet available, for instance in functional genomics and other areas of modern molecular biology and medicine, other approaches have to be applied. Interestingly, biomedical researchers are often interested in learning which of the attributes are the important ones. Only later, the researchers investigate classifiers that may be generated using these attributes. Thus, within such a framework, the first task is to identify the most important attributes. This problem is particularly acute for high-dimensional data of biological origin, where the number of attributes X_i can be of order of thousands.

Recently, a new classifier, actually comprised of an ensemble of decision trees, the so-called random forest (RF) has been proposed by Breiman [15]. The RF's classification ability is comparable to, if not better than that of the best methods available, e.g. boosting [16]. In addition, RF offers two features which improve significantly, and in a very natural way, our understanding of a classification problem under scrutiny. These are:

- the assessment of the importance of the contributions of the attributes to the final prediction,
- the assessment of the interactions between the attributes.

In the present study, we show the limits of the importance estimation as originally proposed by Breiman and present a method that aims at discerning the truly most important attributes for classification and in this respect improves significantly upon the original approach of Breiman.

2 The Method

The process of determining whether a given attribute contributes significantly to the final prediction or not, is based on multiple application of RFs, and utilization of the estimate of importance generated by each RF.

Random Forests. Random forest is a classification method that combines results from an ensemble of many, say M, decision trees built on bootstrap samples drawn with replacement from the original training sample. Each bootstrap sample is of the same size, say N, as the original sample. Drawing with replacement guarantees that roughly 1/3 of elements from the original sample are not used in each bootstrap sample (indeed, note that the probability of not drawing a particular element is $(1 - 1/N)^N \approx e^{-1}$). For each tree in the forest, elements of the original sample not used to grow this tree are called out-of-bag or oob elements for the tree.

Assume that each element (object) in the training sample is given as a vector of P attributes. At each stage of tree building, i.e. for each node of any particular tree in the forest, p attributes out of all P attributes are randomly selected, where $p << P$ (say, $p = \sqrt{P}$), and the best split on these p attributes is used to split the data in the node. Each tree is grown to the largest extent possible, i.e. there is no pruning. In this way, RF consisting of M trees is constructed. Classification of each (new) object is made by simple voting of all trees.

Estimation of Attribute Importance. For any k-th attribute, proceed in the following way. In every tree in the forest, put down its oob objects and count the number of votes cast for the correct class. Then randomly permute the values of attribute k in the oob objects, put these objects down the tree and count the number of votes cast for the correct class. Subtract the latter number of votes from that obtained for the original oob data. The average of this difference over

all trees in the forest constitutes the raw importance score for attribute k. The normalized score, i.e. the so-called z-score by reference to the standard normal case, is obtained as usual, by dividing the raw score by its standard error.

Apparent Importance of Noisy Attributes. Applying the standard normal theory when calculating p-values corresponding to z-scores obtained is well justified by the observed low correlations between scores for individual trees for a number of data sets (cf. [15]). At the same time, however, we have observed that even if all the attributes are completely random and therefore by design not related to the decision attribute, relatively strong correlations between some attributes and the decision attribute may appear by chance. Such attributes are then singled out by the RF and the classifier effectively based on them is built. As a result, these noisy attributes get z-scores which suggest strong, non-random correlation with the decision attribute. The simulation experiment was performed, that indeed confirmed this hypothesis.

Highest z-score of Permuted Attribute. In order to discern properties which significantly and truly, not by chance, contribute to correct predictions, we propose to find by simulation means how high a z-score of a noisy attribute can get on the average. This can be done by estimating the mean and variance of the distribution of the Highest z-score of Permuted Attribute (HZPA). To this end, we introduce controlled noise into the system by randomly permuting values of the selected attributes, retaining the other attributes untouched and generating bootstrap samples of objects modified in the way described.

The Algorithm. We apply this idea in an iterative 'self-consistent' way into the algorithm which generates statistically validated set of important attributes, with average z-score higher than the average z-score of the HZPA.

Our method is coded as a bash script, which calls RandomForest program and two additonal C programs used to generate permuted data sets and perform statistical computations and selection of the variables. The pseudocode of the algorithm is as follows:

```
RunRandomForests()    // Initial run
SetZlimit()           // Set the z-score which will be
                      // considered provisionally important
BuildNonRandomList()  // build an initial list of provisionally
                      // important attributes
while ( !Glob_Self_Cons )
do                                  // Main Loop
  while ( !Loc_Self_Cons )
  do                                // Loop for current Zlimit
    for  (i=1;i<=NSTEP;i++)         // NSTEP repetitions
    do
      ShuffleAttributes             // Create data set with permuted
                                    // unimportant attributes
```

```
      RunRandomForests              // single Random Forest run
    done
    ComputeStatistics               // Computes z-score for all
                                    // important attributes
                                    // and finds AvHZPA
    BuildNewNonRandomList           // Only attributes with
                                    // z-score higher than current
                                    // Zlimit are on the list
    Loc_Self_Cons=CompLists()       // CompLists() returns TRUE
                                    // if old and new list of important
                                    // attributes are identical
  done
  Glob_Self_Cons=ChckGlobCon()      // ChckGlobCon() returns TRUE
                                    // when AvHZPA is lower than current
                                    // Zlimit
  Zlimit=Zlimit+Delta               // Increase current Zlimit
done
```

At the first step, the classifier is used with input vectors consisting of all attributes; z-scores are computed for all attributes, and attributes which have z-score higher than some predefined level are provisionally considered as important, while the remaining ones are considered unimportant. We start with the threshold level equal to 1.0.

The second step consists in running several random forests. Each time the values of the attributes identified in the previous step as unimportant are randomly permuted and values of the important attributes remain unchanged. For each run of RF and all attributes, z-scores are computed, and the average (over all RFs) z-scores for all important attributes are obtained as well. Moreover, for each run of RF, the highest z-score is found among those for the permuted attributes (i.e. HZPA is found), and its average over all RF runs, AvHZPA, is determined for later use. Note that the permuted attribute with the highest z-score can prove different for different RF runs, since they are run on different bootstrap samples.

Attributes which have average scores higher than the current threshold are considered to be temporarily important, and attributes which have average scores lower than the threshold are irreversibly considered to be unimportant. This second step is repeated at each fixed threshold level until all the attributes considered temporarily important have average z-scores higher than the current threshold. After this condition is satisfied, we have a set of temporarily important attributes which we consider "self-consistent at the current threshold level".

Once the self-consistence at a given level has been achieved, in the third step of the procedure, the check is performed if the current threshold level allows one to distinguish the attributes that carry real information from those that do not. If the threshold level is higher than AvHZPA, we conclude that full self-consistence has been reached and the iterative procedure is finished. Otherwise, the threshold is increased and the procedure for reaching self-consistence at this higher threshold level is repeated.

Finally (this step is not included in the pseudocode above), once full self-consistence has been reached, a statistical test of significance is performed for conclusive importance of attributes found important in the third step of the procedure. This test rests on repeating the second step of the procedure, but with a much higher number of iterations (actually, we increase the number of iterations to 1000, while NSTEP was set at 40).

Note that the average score of the non-permuted attributes and the average of the HZPA are obtained from sums of conditionally independent variables, where independence comes from random permutations of the attributes deemed unimportant (the experiment is conditioned on the sample and the fixed values of the attributes deemed important). Therefore, if the number of iterations is sufficiently large, the averages can be assumed to be normally distributed. As the test of significance, a simple one-sided t-test is used, namely the test for equality of two means against the alternative that the mean of z-scores of an attribute tested for importance is higher than the mean of HZPA. We consider the attribute conclusively important if the null hypothesis is rejected at 0.001 significance level. A large number of iterations makes the test sufficiently powerful.

Summarizing, it is indeed a tall order for an attribute to be designated conclusively important. First, full self-consistence requires that the candidates for such designation have average z-scores higher than AvHZPA. And second, an even more stringent requirement is placed in the procedure's final step, namely that the final significance test can be passed by only these attributes whose true average z-score has a chance to be lower than AvHZPA with probability only 0.001, the AvHZPA being obtained on the basis of all attributes conclusively designated unimportant and comprised of the highest scores for each run in the final step.

Additionally, given, say, I attributes designated conlusively important, we generate the distribution of the classification error for the system built on I randomly selected attributes, not including any of the I important attributes determined by the algorithm. We then check if the classification result obtained for the conlusively important attributes is likely to be drawn from the generated distribution.

Computational Complexity. Our algorithm is an overlay superimposed on the original random forest, which calls the original program several times in the iterative fashion. Therefore the computational complexity of the whole algorithm depends both on the computational complexity of the random forest and that of our extension.

Two aspects of the computational complexity should be taken into account - dependence of the number of elementary operations on the number of samples and that on the number of attributes.

Obviously, the complexity of the random forest is of the same order as the complexity of building an individual tree, which is $P^{1/2}Nlog(N)$.

Regarding our extension, it is easily seen that its complexity is independent of the number of samples. On the other hand, dependence of the number of elementary operations on the number of attributes depends on data under

scrutiny. Indeed, the number of iterations depends on the observed importance of attributes. For two limit cases - when an attribute is finally important or is deemed conclusively unimportant in the initial run - the number of iterations of the feature selection algorithm is not affected by the number of attrubutes. However, in the worst case scenario, when an attribute is deemed provisionally important, an additional round of iterations may be necessary to find that it is conclusively unimportant. Therefore, while in the best case the whole algorithm's complexity due to the number of attributes is that of the random forest, i.e., it is of order $P^{1/2}$, in the worst case it is of order $P^{3/2}$. Consequently, the overall complexity of the whole algorithm achieves order $P^{1/2}N \log N$ or $P^{3/2}N \log N$ in the worst case.

3 Data

The algorithm presented in the previous section was applied to twelve data sets of biological origin. The number of objects in the data sets varies between 319 and 820, and the number of attributes for all datasets is 202 including two-valued decision attribute, with the exception of dataset No. 8, where the number of all attributes is 183. Each attribute can take up to twenty categorical values, but usually this number is smaller. For categorical attributes, the device suggested by Theorem 4.5 of [3] was applied to ensure high performance of the classifier. Biologically, each object is a sequence of the HIV protein, and the decision attribute tells, whether virus carrying protein coded with this sequence is, or isn't, resistant to one of the antiviral drugs. Biological implications of our findings will be published elswhere. The data can be accessed at the following URL: http://www.icm.edu.pl/~rudnicki/RoughSets/data/

4 Results and Discussion

The algorithm described is used to find the attributes that contribute significantly to the final prediction. In Table 1, results of the algorithm are compared with those obtained by direct application of the Breiman's approach to finding important attributes. In that approach the random forest is run first with all the attributes, then only the attributes with 'high' z-scores are retained, and finally the forest is run again using only these attributes. In our implementation, z-scores larger than 3 were considered 'high'. Consequently, the attributes with z-scores higher than 3 in the second run are conclusively declared important when using the Breiman's approach.

In the majority of cases classification error is low, and in all cases it is significantly lower than percent error of the random classifier (data not shown).

One may notice that in all cases we found less attributes than suggested by the application of the Breiman algorithm and the assumption that z-score higher than 3 implies importance of an attribute. Interestingly, in all cases, the AvHZPA is significantly higher than 3 and varies considerably between data sets; indeed, it varies between 5.3 and 8.7. Therefore it is impossible to build an *'a priori'*

Table 1. Summary of results for all datasets. The following entries are in the succesive rows: number of objects in each data set (OB), number of important attributes using the method developed in the current study IA (C), number of important attributes obtained using the Breiman approach IA (B), AvHZPA for each data set (AvHZPA) and percent error of the classifier (%ERR).

Data	1	2	3	4	5	6	7	8	9	10	11	12
OB	356	354	353	355	319	354	749	675	820	721	737	767
IA (C)	7	14	20	19	14	15	17	23	30	6	7	7
IA (B)	21	32	39	31	24	23	52	42	59	49	48	58
AvHZPA	8.1	8.3	6.4	5.6	7.0	5.2	6.8	7.3	6.7	8.0	8.7	7.6
%ERR	4.4	11.0	13.9	8.7	24.5	11.6	4.9	18.8	13.9	15.6	26.0	22.6

analytical model of the HZPA distribution and inference has to be based on Monte Carlo-like approach, e.g. as presented in this report.

Accordingly, in Table 2, example results for the final t-test are summarized. Two variables which passed the initial test, have the value of the t-statistic lower than the threshold, set at 3, and consequently they *fail the verification test.*

It is interesting to note that only two variables had z-scores higher than HZPA for all 1000 iterations. Even rather highly scoring attributes had in some iterations scores smaller than HZPA. For example, attribute # 112 had score lower than HZPA in 7 cases out of 1000, despite that its average z-score was almost two times as high as the average of the HZPA.

The results of our study suggest that a single run of the RF classifier, and in particular the attribute importance analysis, may be subject to siginficant random fluctuations generated by spurious correlations between important and unimportant attributes.

Within our approach, this issue has been addressed by multiple application of RFs with randomly permuted values of attributes found unimportant, proper use of the estimates of attributes' importance generated by each RF, and a final test of significance of the results. When looking for important attributes, neither arbitrary selection of the limiting z-score, above which the attribute is considered important, nor (even more artificial) a priori selection of the number of important attributes is needed. Such arbitrary decisions have been replaced by an objective statistical procedure based on comparisons of z-scores for original attributes with the HZPA. Only the attributes which in many bootstrap samples score significantly higher than any attribute which is unimportant by design, can be conjectured to be important.

A related problem has been studied by Gediga and Duentsch within the rough set framework [17,18] several years ago. They have shown limited applicability of statistical methods in assesing the rule importance. Instead, they introduce the notion of casual dependencies in information systems and provide arguments that approximate reducts cannot be applied to measure quality of a model in certain cases.

Their results do not apply to our approach, since the methodology presented here is developed towards minimizing, to any desirable level, the error of the

Table 2. The results for the final step of the algorithm. Sixteen provisionally important attributes were tested using one thousand replications. Subsequent columns represent attribute number (*Attribute*), average z-score (*Z*) over 1000 iterations, standard deviation (*SDev(Z)*) of the mean z-score, average rank (*Rank*) in the importance ranking, standard deviation of the rank (*SDev(R)*), value of *t*-statistic (*t*) and the number of instances, when given attribute had higher score than an AvHZPA (*Inst*), respectively.

Attribute	Z	SDev(Z)	Rank	SDev(R)	t	Inst
2	29.11	0.05	2.000	0.000	60.7	1000
4	11.01	0.04	10.20	0.07	8.3	874
28	10.66	0.05	11.04	0.1	6.8	824
35	9.57	0.04	13.42	0.07	3.9	735
36	14.24	0.03	4.58	0.03	18.7	974
38	9.28	0.03	14.16	0.06	3.2	702
67	**8.40**	**0.03**	**15.62**	**0.05**	**0.4**	**578**
77	12.16	0.03	7.71	0.05	12.3	918
79	11.48	0.04	9.22	0.08	9.6	896
112	16.52	0.03	3.05	0.007	25.8	993
145	64.50	0.09	1.000	0.000	141.4	1000
169	13.40	0.03	5.50	0.04	15.8	959
171	11.10	0.04	10.05	0.08	8.4	858
176	13.09	0.05	6.35	0.07	13.8	945
179	**9.03**	**0.04**	**14.48**	**0.07**	**2.2**	**661**
189	11.02	0.04	10.22	0.07	8.4	867
AvHZPA	**8.28**	**0.08**	**14.78**	**0.11**	–	–

second kind (that is to minimize the number of false positives), whereas the approach of Gediga and Duentsch pertains to minimization of the error of the first kind (minimizing the number of false negatives).

Acknowledgments

The authors acknowledge funding from the EU grant HPRI-CT-2001-00153, and Wallenberg Foundation. Computations were performed at ICM, Warsaw University, grant G26-11.

References

1. Cowell, R.G., Dawid, A.P., Lauritzen, S.L., Spiegelharter, D.J. (1999) *Probabilistic networks and expert systems*. Springer-Verlag, New York.
2. Bishop, C.M. (1996) *Neural Networks for Pattern Recognition*. Clarendon Press, Oxford
3. Breiman, L., Friedman, J.H., Olshen, R.A. and Stone, C.J. (1984) *Classification and Regression Trees*. Wadsworth International Group, Monterey, Ca.
4. Ripley, B.D. (1996) *Pattern Recognition and Neural Networks*. Cambridge University Press, Cambridge

5. Hastie, T., Tibshirani, R. and Friedman, J. (2001) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, New York
6. Pearl, J. (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, San Francisco.
7. Pawlak, Z. (1981) Information systems theoretical foundations, *Inf. Syst.* **6**, 205–218. Rough Set Theory
8. Komorowski, J., Oehrn, A, Skowron, A. (2002). ROSETTA Rough Sets. In *Handbook of Data Mining and Knowledge Discovery*, W. Klsgen and J. Zytkow (eds.), pp. 554–559, Oxford University Press.
9. Bazan, J. and Szczuka, M. (2001). RSES and RSESlib A collection of tools for rough set computations. In *Proc. of RSCTC'2000, LNAI 2005*, pp 106–113, Springer-Verlag, Berlin.
10. Pawlak, Z. (1991) *Rough Sets. Theoretical Aspects of Reasoning about Data*. Kluwer Academic Publishers. Rough Set Theory
11. Ågotnes T., Komorowski H. J. and Løken T. Taming Large Rule Models in Rough Set Approaches. In Zytkow J. M. and Rauch J. (Eds.) *Principles of Data Mining and Knowledge Discovery, Third European Conference, PKDD '99, Proceedings*. LNCS **1704**, 193–203
12. Makosa E. Rule Tuning, *MSc Thesis*, The Linnaeus Center for Bioinformatics, Uppsala University, 2005.
13. Nguyen H. S., Nguyen S. H. (1998). Pattern extraction from data. *Fundamenta Informaticae* **34**, 129–144.
14. Nguyen H. S., Skowron A. and Synak P. (1998). Discovery of data patterns with applications to decomposition and classfification problems. In: L. Polkowski and A. Skowron (eds.), *Rough Sets in Knowledge Discovery 2: Applications, Case Studies and Software Systems*. Physica-Verlag, 55–97.
15. Breiman, L. Random Forests, *Machine Learning* **45** (2001), 5–32. Also see the bibliography at: http://www.stat.berkeley.edu/~breiman/RandomForests/cc_papers.htm
16. Freund, Y. and Schapire, R. (1996) Experiments with a new boosting algorithm, *Machine Learning: Proceedings of the Thirteenth International Conference*, Morgan Kauffman, San Francisco, 148–156. Also see the bibliography at: http://www.cs.princeton.edu/~schapire/boost.html
17. Duentsch I. and Gediga G. (1998). Uncertainty Measures of Rough Set Prediction. *Artif. Intell.* **106**, 109–137.
18. Duentsch I. and Gediga G. (1997). Statistical evaluation of rough set dependency analysis. *Int. J. Hum.-Comput. Stud.* **46** 589–604.