An Input Variable Importance Definition based on Empirical Data Probability and Its Use in Variable Selection

Vincent Lemaire and Fabrice Clérot
France Telecom Research and Development
FTR&D/DTL/TIC
2 avenue Pierre Marzin
22307 Lannion Cedex - France
E-mail: {vincent.lemaire,fabrice.clerot}@francetelecom.com

Abstract— **Variable and feature selection have become the focus of much research in areas of application for which datasets with tens or hundreds of thousands of variables are available. We propose a new method to score subsets of variables according to their usefulness for the performance of a given model. This method is applicable on every kind of model and on classification or regression task. We assess the efficiency of the method with our results on the NIPS 2003 feature selection challenge and with an example of a real application.**

I. Introduction

Up to 1997, when a special issue on relevance including several papers on variable and feature selection was published [2], few domains explored more than 40 features. The situation has changed considerably in the past few years, notably in the field of data-mining with the availability of ever more powerful data warehousing environments.

A recent special issue of JMLR [7] gives a large overview of techniques devoted to variable selection and an introduction to variable and feature selection can be found in this special issue [6]. A challenge on feature selection has been organized during the NIPS 2003 conference to share techniques and methods on databases with up to 100000 features.

The objective of variable selection is three-fold: improve the prediction performance of the predictors, provide faster and more cost-effective predictors, and allow a better understanding of the underlying process that generated the data.

Among techniques devoted to variable selection we find filter methods, which select variables by ranking them with correlation coefficients, and subset selection methods, which assess subsets of variables according to their usefulness to a given model.

Wrapper methods [8] use the elaborated model as a black box to score subsets of variables according to their usefulness for the modeling task. In practice, one needs to define: (i) how to search the space of all possible variable subsets; (ii) how to assess the prediction performance of a model to guide the search and halt it; (iii) which predictor to use.

We propose a new method to perform the second point above and to score subsets of variables according to their predictive power for the modeling task. It relies on a definition of the variable importance as measured from the variation of the predictive performance of the model (classification or regression). The method is motivated and described in section II. Having presented the NIPS feature selection challenge in III, we compare in section IV the performance of the proposed method with other techniques on this challenge. Section V shows an example of application in a practical context and we conclude in VI.

II. Analysis of an Input Variable Influence

A. Motivation and previous works

Our motivation is to measure variable importance given a predictive model. The model is considered a perfect black box and the method has to be usable on a very large variety of models for classification (whatever the number of classes) or regression problems.

When a predictive model has been built, a question often raised in practice is ‘*What* would happen to this individual *if* this variable was set to a different value *?*’. A simple way to answer this question is to plot the variation of the output of this predictive model for this individual versus the variation of the variable [10], [9], [1].

For non-linear models the variation of the output can be non-monotonous. Hence, the influence of an input variable cannot be evaluated by a local measurement. The measurement of the difference of the output of a model with respect to the variation of an input variable provides a more global information and can be applied to discrete variables. However, the choice of the variation range should depend on the variable: too small a value has the same drawback as the partial derivatives (local information and not well suited for discrete variables), too large a value can be misleading if the function (the model) with respect to an input V is non-monotonous, or periodic.

A characteristic of the ‘what if?’ simulation is that it relies on the generalization capabilities of the model since the output of the model is calculated with values of the variables which can be away from the training set; for instance, a discrete variable can be treated as a continuous one. The ‘what if?’ simulation is extended to define causal importance and saliency measurement by Féraud et al. in [5]. Their definition however does not take into account the true interval of variation of the input variables. They propose to use a prior on the possible values of the input variables. The knowledge needed to define this prior depends on the specificities of the input variable (discrete, positive, bounded, etc). Such individual knowledge is clearly difficult and costly to obtain for databases with a large number of variables. A more automatic way than this ‘prior’ approach is needed.

A first step in this direction is given by Breiman in [3] (paper updated for the version 3.0 of the random forest) where he proposes a method which relies on the distribution of probability of the variable studied. Each example is randomly perturbed by randomly drawing another value of the studied variable among the values spanned by this variable across all examples. The performance of the perturbed set are then compared to the ‘intact’ set. Ranking variable performance differences allows to rank variable importance. This method allows to automatically determine the possible values of a variable from its probability distribution, even if perturbing every example only once does not explore the influence of the full probability distribution of the variable. Moreover, although [3] seems to restrict the method to random forests, it can obviously be extended to other models.

The method described in this article combines the definition of the ‘variable importance’ as given in Féraud et al. [5] with an extension of Breiman’s idea [3]. This new definition of variable importance both takes into account the probability distribution of the studied variable and the probability distribution of the examples.

B. Definition of the variable importance

The importance of an input variable is a function of examples I (see Figure 1) probability distribution and of the probability distribution of the considered variable (V_j).

Let us define:

- V_j the variable for which we look for the importance;
- V_{ij} the realization of the variable V_j for the example i;
- I_m the example m a vector with n components;
- f the predictive model;
- $P_{V_j}(u)$ the probability distribution of the variable V_j;
- $P_I(\nu)$ the probability distribution of examples I.

and

$$f_j(a;b) = f_j(a_1, ..., a_n; b) = f(a_1, ..., a_{j-1}, b, a_{j+1}, ..., a_n) \quad (1)$$

where a_p is the p^{th} component of the vector a.

The importance of the variable V_j (see Figure 1) is the average of the measured variation of the predictive model output when examples are perturbed according to the probability distribution of the variable V_j. The perturbed output of the model f, for an example I_i is the model output for this example but having exchanged the j^{th} component of this example with the j^{th} component of another example, k. The measured variation, for the example I_i is then the difference between the ‘true output’ $f_j(I_i;V_{ij})$ and the ‘perturbed output’ $f_j(I_i;V_{kj})$ of the model. The importance of the variable V_j is then the average of $|f_j(I_i;V_{ij}) - f_j(I_i;V_{kj})|$ on both the examples probability distribution and the probability distribution of the variable V_j. The importance of the variable V_j for the model f is then:

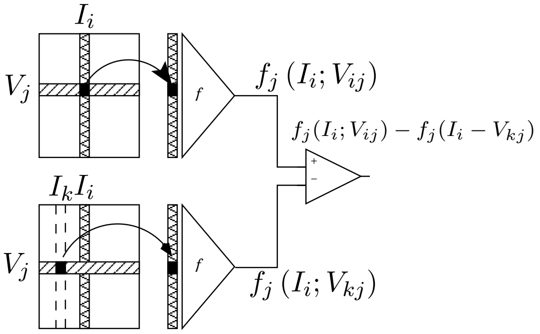
I_i
V_j
f
$f_j\left(I_i;V_{ij}\right)$
$f_j(I_i;V_{ij}) - f_j(I_i - V_{kj})$
+
−
I_kI_i
V_j
f
$f_j\left(I_i;V_{kj}\right)$

Fig. 1. Graphical representation of the random draw

$$S(V_j|f) = \iint P_{V_j}(u)du P_I(v)dv \left|f_j\left(I_i;V_{ij}\right) - f_j\left(I_i;V_{kj}\right)\right| \quad (2)$$

C. Computation

Approximating the distributions by the empirical distributions, the computation of the average of $S(V_j|f)$ would require to use all the possible values of the variable V_j for all examples available. For N examples and therefore N possible values of V_j the computation time scales as N^2 and become very long for large databases.

There are, at least, two faster heuristics to compute $S(V_j|f)$:

1) We draw simultaneously I_i and V_{kj} and compute one realization of $|f_j(I_i,V_{ij}) - f_j(I_i,V_{kj})|$. The measure of the average of $S(V_j|f)$ is then realized by means of a Kalman filter until convergence (see [11] to initialize and set the Kalman filter parameters).

2) $S(V_j|f)$ can be written:

$$S(V_j|f) = \int P_I(v)dv \int P_{V_j}(u)du \left|f_j\left(I_i;V_{ij}\right) - f_j\left(I_i;V_{kj}\right)\right| \quad (3)$$

Approximating the probability distribution of the data by the empirical distribution of the examples:

$$S(V_j|f) = \frac{1}{N}\sum_{i\in N} E\left\{\left|f_j\left(I_i;V_{ij}\right) - f_j\left(I_i;V_{kj}\right)\right|\right\} \quad (4)$$

As the variable probability distribution can be approximated using representative examples (P) of an ordered statistic:

$$S(V_j|f) = \frac{1}{N}\sum_{i\in N}\sum_{p\in P} \left|f_j\left(I_i;V_{ij}\right) - f_j\left(I_i;v_p\right)\right| \text{Prob}(v_p) \quad (5)$$

The computation can also be stopped with a Kalman filter. This method is especially useful when V_j takes only discrete values since the inner sum is exact and not an approximation.

D. Application to feature subset selection

The wrapper methodology [8] offers a simple and powerful way to address the problem of variable selection, regardless the chosen learning machine. The learning machine is considered a perfect black box and the method lends itself to off-the-shelf machine learning software packages. Exhaustive search can only be performed if the number of variables is small and heuristics are otherwise necessary. Among these, backward elimination and 'driven' forward selection which can both rely on the variable importance described above.

In backward elimination one starts with the set of all variables and progressively eliminates the least important variable. The model is re-trained after every selection step. In forward selection, as in [3], at a first step we train a model with all variables then we rank the variables using the method described in this paper and in a second step we train models where variables are progressively incorporated into larger and larger subsets according with their ranks.

Comparison between both methods will be discussed elsewhere. Hereafter we restrict the discussion to backward elimination. We note here that both methods have the appealing property of depending on one parameter only, the degradation of the performance of the model trained with the subset relatively to the best possible performance reached.

To speed up the backward elimination another parameter is added. At each step of the backward elimination we remove all variables with an importance smaller than a very low threshold (10^{-6}). With this implementation the backward elimination method has only two simple parameters, a performance threshold to define the selected subset and an importance threshold to discard variables with 'no' importance.

III. Feature Selection Challenge

A. Introduction

Asserting the performance of data-mining methods is always a difficult task. Standard 'benchmark' problems such as the databases of the UCI repository are not well-suited to investigate the properties of variable selection techniques since most of the databases include only a small number of variables.

The purpose of the NIPS 2003 workshop on feature extraction was to bring together researchers of various application domains to share techniques and methods. Organizers of the challenge[1] formatted a number of datasets for the purpose of benchmarking feature selection algorithms in a controlled manner. The data sets were chosen to span a wide variety of domains. They chose data sets that had sufficiently many examples to create a large enough test set to obtain statistically significant results. The input variables are continuous or binary, sparse or dense. All problems however are two-class classification problems. The similarity of the tasks will allow participants to enter results on all data sets to test the genericity of the algorithms.

Each dataset was split in 3 sets: training, validation and test set. Only the training labels were provided. During the development period, challengers could send classification results (on the five datasets or on only one) and received in return validation set error rate. At any time the participants could submit their final classification results. A submission was considered final if the author(s) made a simultaneous submission on the five data sets before the deadline. A very large number of submissions were made on each dataset (840 for the most tried) but there were only 136 final submissions and 56 final valid submissions (organizers kept the five better results of every challenger).

B. Datasets

We describe here very briefly the five datasets. The number of examples for each train, valid and test set are given in Table I. Manipulations of the datasets described below were performed by the organizers before the challenge.

- The task of ARCENE is to distinguish cancer versus normal patterns from mass-spectrometric data (continuous input variables). For data compression reasons organizers of the challenge thresholded the values. Before the benchmark linear SVM trained on all features had 15 % test error rate.
- The task of GISETTE is to discriminate between confusable handwritten digits: the four and the nine (sparse continuous input variables, many methods have been tried on this dataset, see `yann.lecun.com/exdb/mnist/`). The dataset was normalized so that the pixel values would be in the range [0,1] then values below 0.5 have been thresholded by the organizer to increase data sparsity. Before the benchmark linear SVM trained on all features had 3.5 % test error rate.
- The task of DEXTER is to filter texts about 'corporate acquisitions' (sparse continuous input variables, see `kdd.ics.uci.edu/databases/reuters21578/`). The order of the features and the order pattern were randomized. Before the benchmark linear SVM trained on all features had 5.8 % test error rate.
- The task of DOROTHEA is to predict which compounds bind to Thrombin (sparse binary input variables). Before the benchmark 'lambda method' trained on all features had 21 % test error rate (no linear SVM tried).
- The task of MADELON is to classify artificial data (continuous input variables) with only 5 useful features. Before the benchmark organizers of the challenge used a K-nearest method, with $K = 3$, with the 5 useful features only which gives a 10 % error rate.

Probes refer to 'random features' distributed similarly to the real features and added to every dataset. This allows organizers to rank algorithms according to their ability to filter out irrelevant features.

[1] All the informations about the challenge, the datasets, the results can be found on: `www.nipsfsc.ecs.soton.ac.uk`

TABLE I

DATA STATISTICS

Dataset	Fraction of probes	Number of Features	Training set	Validation set	Test set
Arcene	30 %	10000	100	100	700
Gisette	50 %	5000	6000	1000	6500
Dexter	50 %	20000	300	300	2000
Dorothea	50 %	100000	800	350	800
Madelon	96 %	500	2000	600	1800

IV. RESULTS AND COMPARISON OF THE NIPS 2003 CHALLENGE

A. Test conditions on the proposed method

As we wished to investigated the performance of our variable importance measurement, we chose to use a single learning machine for all datasets (no bagging, no Ada-boost, no other bootstrap method): a MLP neural network with 1 hidden layer, tangent hyperbolic activation function and stochastic back-propagation of the squared error as training algorithm. We added a regularization term active only on directions in weight space which are orthogonal to the training update [4].

For each dataset we split the training set in two sets: a training (70 %) and a validation set (30 %); the validation set of the challenge is then used as a test set. We made a single final submission before December first and we decided to keep this submission after December first (the valid submissions made before December first received the labels of the validation set, allowing a new attempt which was to be sent before December 8th). Therefore we compare below the results obtained with the proposed method with the valid results sent before December first.

The preprocessing used is only a zero-mean, unit-variance standardization. The strategy used to constitute the selected variable subset is the standard backward elimination. The subset of variables was chosen as the smallest subset allowing a performance greater than 95 % of the best performance reached during the selection process.

B. Comparison with others results

1) Comparison with baseline results: Our results compared to the baseline results, linear SVM and 'lambda method' (features selection by correlation with the target followed by Golub's classifier; see `clopinet.com/isabelle/Projects/NIPS2003/Slides/NIPS2003-Datasets.pdf`) are presented in Table II.

The results presented in Table II show that all the results obtained are included between the results of the lambda method and the linear SVM. The Fraction Of Features (FoF) is defined as the ratio of the number of used variables by the classifier to the total number of variables in the dataset and the Balanced Error Rate (BER) as the average of the error rate on positive class examples and the error rate on negative class examples.

TABLE II

TEST BALANCED ERROR RATE

Dataset	1	2	3	4	5	6
Arcene	30	1.5	-	15	30	29.65
Dexter	50	0.61	-	5.8	20	9.70
Dorothea	50	0.07	-	-	21	22.24
Madelon	96	1.6	10	-	41	16.38
Gisette	50	1.8	-	3.5	30	3.48

1: Fraction of probes of the dataset (%)
2: Fraction of features used (%)
3: K-nearest BER (%)
4: Linear SVM's BER (%)
5: Lambda method's BER (%)
6: Neural network BER (%),with the best subset of variables

Clearly restricting ourselves to a simple model with no bootstrap techniques cannot allow us to reach very good BER, particularly on databases as ARCENE where the number of example is quite small.

2) Comparison with same model using all the variables: Our results compared to the results of a neural network trained with all the variables are presented in Table III.

TABLE III

TEST BALANCED ERROR RATE

Dataset	1	2	3
Arcene	1.5	20.2	29.65
Dexter	0.61	15.1	9.70
Dorothea	0.07	30	22.24
Madelon	1.6	31.5	16.38
Gisette	1.8	4.3	3.48

1: Fraction of features used (%)
2: Neural network BER (%), with all the variables
3: Neural network BER (%), with the best subset of variables

The performance of the model trained with the subset relatively to the performance of the model trained with all variables are improved on every dataset excepted ARCENE. For this dataset it seems that the use of only 70 examples for training with no bootstrap does not allow to have good variable selection considering generalization performances.

An example of the BER obtained during the backward elimination phase is presented on the Figure 2 for GISETTE.

The BER does not increase as we remove (backward elimination) variables until a very small number of variables is reached where the BER starts to be affected and increases sharply. Note that during the first step of the backward elimination we remove all variables with 'no' importance. For GISETTE this first step removes 44 % of the variables (there are 50 % of probes in GISETTE). At the end of the backward elimination we keep 90 variables, that is 1.8 % of the features, and we only have 5.56 % of probes, that is 5 'dummy'

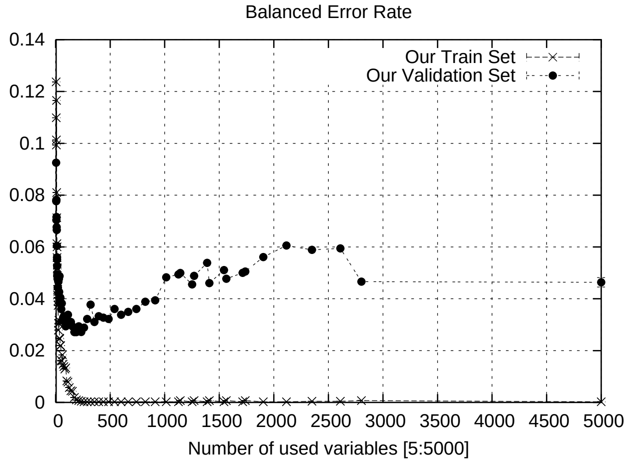

Fig. 2. Balanced Error Rate during the backward elimination for GISETTE

variables, among these 90 variables. This shows that for this database, even with a single MLP, the variable selection task has been well performed.

3) Comparison with all other valid submissions : The organizers of the challenge rated the classification results only with the BER. For methods having performance differences that are not statistically significant, the method using the smallest number of features win.

'Variable selection' is always somewhat ambiguous when the result is judged from the BER only, specially when different learning machines are used, since it is more a matter of balance between BER and FoF rather than a matter of BER only: to prefer a BER=0.1 using 50 % of features to a BER=0.12 using 10 % of the features is mostly a matter of application requirements. In some applications, one would trade some accuracy for less features, as in the real-time application we describe below for instance.

We first note that our results with a single MLP compare quite favorably with results by Amir Reza (results named 'SimpleNN' on the web site challenge) also using a single MLP and all features even if, as stated above, the use of a single learning machine without bootstrap does not lead to excellent BERs. However as can be seen below our BER results are close to the average results. The point here is just to stress that our model, although admittedly not the most adapted for accuracy on some datasets, indeed reaches a 'reasonable' BER (see also point 2) below).

What we expect from a variable selection technique is to adapt itself in such situation by removing as many features as possible. Therefore, what we can expect from the combination of our simple model and our selection technique is to keep a BER reasonably close to the average while using significantly less features on all datasets.

Below we use a representation of the results which allows a comparison of the proposed method to other methods on the five datasets. This representation has two axis (see Figure 3): the first axis of comparison is the ratio between the BER of a submitted method and our BER (BER*) and the second axis is the ratio between our FoF (FoF*) and the FoF of a submitted method. For each dataset our results are placed in the center of the figure. Each author(s) is represented with a marker symbol. A marker is placed for each method of this author(s) and for each dataset. This allows to compare the results for each dataset. This simple representation allows to define 4 classes of methods on every dataset: 1) a better BER and a better FoF; 2) a better BER but a worse FoF; 3) a better FoF but a worse BER; 4) a worse FoF and a worse BER. As authors were able to send more than one submission, authors may have more than five identical marker symbol.

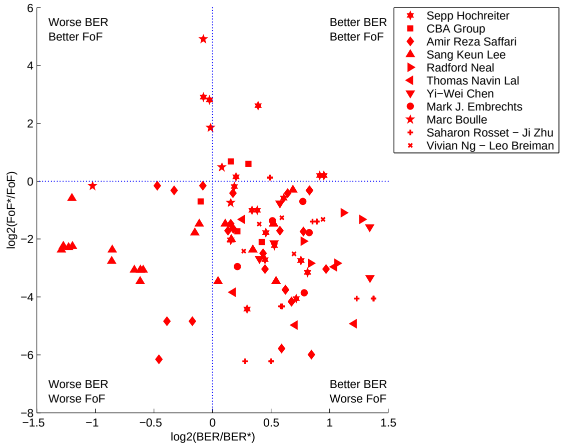

Fig. 3. Results on the test set for all the valid final submissions (56) labelled by author(s).

The figure 3 shows results of methods which never used 100 % of the variables compared to the results obtained with our method. This figure shows that compared to the proposed method:

1) No method gave better results (better BER and better FoF) on the five datasets.
2) No method obtained a significantly better BER (less than 0.8 BER* regardless of the FoF) on the five datasets.
3) Half of the authors have tried a method which gives a worse BER and a worse FoF.
4) Only 4 authors proposed methods allowing to have a better BER and less features on some datasets.
5) The proposed method, combined with backward elimination using only one neural network, selects very few variables compared with the other methods.

The points above show that the proposed variable selection technique exhibits the expected behavior by both keeping the BER to a reasonable level (better than the BER with all features, except for ARCENE as already discussed, close to the average result of the challenge) and dramatically reducing the number of features on all datasets.

V. Application to Fraud Detection

The case study is the on-line detection of the fraudulent use of a post-paid phone card. Here the 'fraud' term includes all cases which may lead to a fraudulent non-payment by the caller. The purpose is to prevent non-payments by warning the owners of phone card that the current use of their card is unusual. The original database contains 15330 individuals described with 368 inputs variables of various natures. The database contains 97 % examples which belong to the class 'not fraudulent' and 3 % which belong to class 'fraudulent'.

Using all variables in the modeling phase allows to obtain good fraudulent/non fraudulent classification performances but this model cannot be applied on-line because of computing and data extraction time constraints. It is thus necessary to reduce significantly the number of variables while keeping good performances.

The BER on the test set versus the number of variables is given in Figure 4. This figure shows that with the proposed method one can obtain the same BER with 100 variables than with 368 variables and a small degradation using 90 variables. Accepting a degradation of the performance by 10 % allows to retain only 40 variables.

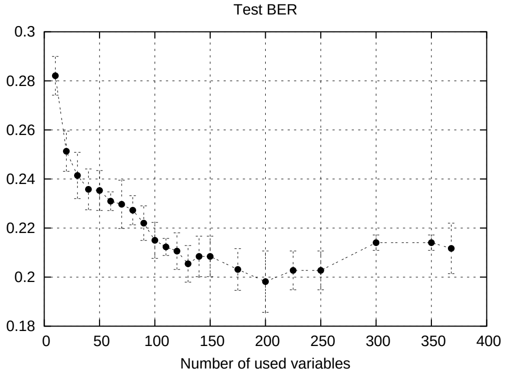

Fig. 4. BER on the test set versus the number of variables.

The classification performances are given below in the form of lift curves in Figure 5 using 40, 90 and all the variables. Regarding the variable selection method, the performances of the neural network trained with 90 variables shows a marginal degradation of the performance as compared to the neural network trained with all the 368 variables. The neural network trained with 40 variables shows no degradation of the performance for small segments of the population: the selectivity is the same up to a lift ratio of 0.6 which is a key issue for such systems where only small segments of the population can be processed in real time.

These results show that, on this real application, it is possible to obtain excellent performances with the methodology described in this paper. Moreover, it allows a much simpler interpretation of the model as it only relies on much fewer input variables but such business-oriented discussion is out of the scope of this paper.

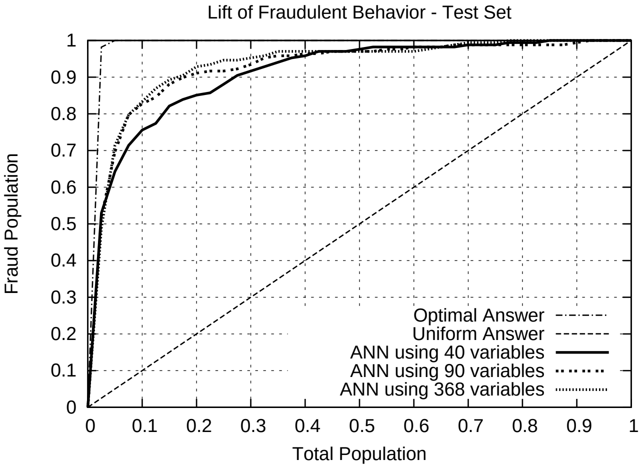

Fig. 5. Detection rate (%) of the fraudulent users obtained with different number of variables (ANN: Artificial Neural Network), given as a lift curve.

VI. Conclusions

We presented a new measure which allows to estimate the importance of each input variable of a model. This measure has no adjustable parameter, is applicable on every kind of model and for classification or regression task.

Experimental results on the NIPS 2003 feature selection challenge show that using this measure coupled with backward elimination allows to reduce considerably the number of input variables with no degradation of the modeling accuracy. Experimental results on a real application show the effectiveness of this approach.

References

[1] W. G. Baxt and H. White. Bootstrapping confidence intervals for clinical inputs variable effects in a network trained to identify the presence of acute myocardial infraction. *Neural Computation*, 7:624–638, 1995.

[2] A. Blum and P. Langley. Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97(1-2):245–271, December 1997.

[3] Leo Breiman. Random forest. *Machine Learning*, 45, 2001.

[4] Anthony N. Burkitt. Refined pruning techniques for feed-forward neural networks. *Complex System*, 1992.

[5] Raphael Féraud and Fabrice Clérot. A methodology to explain neural network classification. *Neural Networks*, 15:237–246, 2002.

[6] Isabelle Guyon and André Elisseef. An introduction to variable and feature selection. *JMLR*, 3(Mar):1157–1182, 2003.

[7] JMLR, editor. *JMLR Special Issue on Variable and Feature Selection*, volume 3(Mar). Journal of Machine Learning Research, 2003.

[8] R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence*, 97(1-2), 1997.

[9] J. Moody. *Prediction Risk and Architecture Selection for Neural Networks*. From Statistics to Neural Networks-Theory and Pattern Recognition. Springer-Verlag, 1994.

[10] A. N. Réfénes, A. Zapranis, and J. Utans. Stock performance using neural networks: A comparative study with regression models. *Neural Network*, 7:375–388, 1994.

[11] Greg Welch and Gary Bishop. SCAAT: Incremental tracking with incomplete information. In *SIGGRAPH*, Los Angeles, August 12-17 2001.