Effective Input Variable Selection for Function Approximation

L.J. Herrera[1], H. Pomares[1], I. Rojas[1], M. Verleysen[2], and A. Guilén[1]

[1] Computer Architecture and Computer Technology Department
University of Granada, 18071, Granada, Spain
[2] Machine Learning Group
3 Place du Levant, 1348 Louvain la Neuve, Belgium

Abstract. Input variable selection is a key preprocess step in any I/O modelling problem. Normally, better generalization performance is obtained when unneeded parameters coming from irrelevant or redundant variables are eliminated. Information theory provides a robust theoretical framework for performing input variable selection thanks to the concept of mutual information. Nevertheless, for continuous variables, it is usually a more difficult task to determine the mutual information between the input variables and the output variable than for classification problems. This paper presents a modified approach for variable selection for continuous variables adapted from a previous approach for classification problems, making use of a mutual information estimator based on the k-nearest neighbors.

1 Introduction

Input variable selection is a very important preprocessing step in any supervised or unsupervised learning problem. Having a number of irrelevant or redundant input variables can lead to overfitting and to a poor generalization of the model [3]. Furthermore in models that suffer from the curse of dimensionality in the number of input variables like grid-based fuzzy models [5], input variable selection becomes essential.

Two main trends can be followed to perform this process. Filter methods try to select the variables in a preprocess step with the only information that the I/O values bring. Wrapper methods employ the learning methodology that is going to be used, in order to select the subset of variables that brings the best performance. In both cases, there are two options to perform the "selection" of the variables subset. On the one hand it is possible to select a subset of the original variables (feature selection or input variable selection). On the other hand the initial set of input variables can be replaced by a new subset of variables that are usually obtained by linear or nonlinear transformations of the original ones (feature extraction or input variable extraction).

This paper deals with filter methods for feature selection. Filter methods have the great advantage that the model has no influence on the selected variables.

S. Kollias et al. (Eds.): ICANN 2006, Part I, LNCS 4131, pp. 41–50, 2006.

Thus they can be used as a completely separated preliminary step to the Input/Output (I/O) modelling problem. Several methodologies for input variable selection exist in the literature for both feature selection and feature extraction approaches. Principal component analysis (PCA) and kernel-PCA algorithms are examples of feature extraction methods [2,3,4]. Feature selection methods have the advantage that the meaning or understandability of the input variables of the problem is kept in the model.

For input variable selection, information theory offers a good theoretical environment for variable filtering thanks to the concepts of entropy and mutual information (MI) between variables [11]. Nevertheless, for regression problems it is a harder task to use these concepts. In regression the input and output variables take continuous values, and additional techniques have to be used to estimate the probability distribution [1]. This problem becomes even more pronounced specially when the number of data points is low comparing to the number of input variables (DNA Micro-arrays, etc.). Among the techniques to estimate the probability density functions (PDF) we can find histogram and kernel-based PDF estimators. But those estimators suffer from the curse of dimensionality and can be used for problems with a low number of variables.

A number of estimators for the entropy based on the k-nearest neighbor statistics also exist. Only recently they have been extended to the mutual information estimation by Kraskov et al [9,10]. A nice property of this estimator is that it can be used easily for sets of variables.

Using the concept of mutual information between two or more variables, a number of algorithms could be designed [1,7,8]. This paper presents a modification of the work presented in [6], adapted for continuous variables thanks to the use of the MI estimator based on the k-nearest neighbors [9]. The simulations section presents the application of the new methodology to Least Squares Support Vector Machines (LS-SVMs). It is also compared with other recent input variable selection methods presented in the literature.

The rest of the paper is organized as follows. Section 2 briefly explains the mutual information concept for continuous variables (subsection 2.1); it overviews the k-nearest neighbors estimator that is used (subsection 2.2); and finally presents the proposed algorithm for variable selection based on MI (subsection 2.3). Section 3 shortly reviews the basics of the LS-SVMs. Section 4 presents examples of application of the variable selection methodology. Section V presents the main conclusions drawn from the study.

2 Effective Feature Selection Based on the Mutual Information

In this section the basics of the proposed variable selection algorithm are presented. First the mutual information concept for continuous variables is briefly explained, followed by a k-nearest neighbors procedure to estimate it. Finally the algorithm, which is an adaptation of a previous work for discrete cases in [6], is presented.

2.1 Mutual Information

Given a single-output multiple input (MISO) function approximation or classification problem, with input variables $X = [x_1, x_2, \ldots, x_n]$ and output variable $Y = y$, the main goal of a modelling problem is to reduce the uncertainty on the dependent variable Y. According to the formulation of Shannon, and in the continuous case, the uncertainty on Y is given by its entropy defined as

$$H(Y) = -\int \mu_Y(y) \log \mu_Y(y) dy, \tag{1}$$

considering that the marginal density function $\mu_Y(y)$ can be defined using the joint PDF $\mu_{X,Y}$ of X and Y as

$$\mu_Y(y) = \int \mu_{X,Y}(x, y) dx. \tag{2}$$

Given that we know X, the resulting uncertainty of Y conditioned to known X is given by the conditional entropy, defined by

$$H(Y|X) = -\int \mu_X(x) \int \mu_Y(y|X = x) \log \mu_Y(y|X = x) dydx. \tag{3}$$

The joint uncertainty on the $[X, Y]$ pair is given by the joint entropy, defined by

$$H(X, Y) = -\int \mu_{X,Y}(x, y) \log \mu_{X,Y}(x, y) dxdy. \tag{4}$$

The mutual information (also called cross-entropy) between X and Y can be defined as the amount of information that the group of variables X provide about Y, and can be expressed as

$$I(X, Y) = H(Y) - H(Y|X). \tag{5}$$

In other words, the mutual information $I(X, Y)$ is the decrease of the uncertainty on Y once we know X. Due to the mutual information and entropy properties, the mutual information can also be defined as

$$I(X, Y) = H(X) + H(Y) - H(X|Y), \tag{6}$$

leading to

$$I(X, Y) = \int \mu_{X,Y}(x, y) \log \frac{\mu_{X,Y}(x, y)}{\mu_X(x)\mu_Y(y)} dxdy. \tag{7}$$

Thus, only the estimate of the joint PDF between X and Y is needed to estimate the mutual information between two groups of variables.

Estimating the joint probability distribution can be performed using a number of techniques. As mentioned already, histograms and kernel density estimators have been used for this purpose [1]. The next subsection will shortly review how to use a k-nearest neighbors methodology to estimate the MI.

2.2 Estimating the Mutual Information Using the *k*-Nearest Neighbors

There is extensive literature about estimators based on the k-nearest neighbors for the entropy, but it has been only recently extended to the MI [9].

Thanks to that estimator, it is possible to use sets of variables in the estimation of the MI, and thus it will allow to adapt the method presented in [6].

We define the space $Z = X, Y$ and we will use the maximum norm for any pair of points $z = (x, y)$ and $z' = (x', y')$,

$$\|z - z'\| = \max\{\|x - x'\|, \|y - y'\|\}, \tag{8}$$

although any other norm could be used. Denote by $\varepsilon(i)$ the distance from a point z_i to it is k-th nearest neighbor and by $\varepsilon_x(i)$ and $\varepsilon_y(i)$ the distances between the same points projected into the X and Y subspaces. Obviously $\varepsilon(i) = \max\{\varepsilon_x(i), \varepsilon_y(i)\}$.

We will count the number $n_x(i)$ of points x_j whose distance from x_i is strictly less than $\varepsilon(i)$, and similarly for y instead of x. The estimate for MI is then (see [9] for a proof of the convergence of this estimator)

$$\hat{I}_1(X, Y) = \psi(k) - \frac{1}{N} \sum_{i=1}^{N} [\psi(n_x(i) + 1) + \psi(n_y(i) + 1)] + \psi(N), \tag{9}$$

where ψ is the digamma function given by

$$\psi(t) = \frac{\Gamma'(t)}{\Gamma(t)} = \frac{d}{dt} \ln \Gamma(t). \tag{10}$$

Function ψ satisfies the recursion $\psi(x + 1) = \psi(t) + 1/x$ and $\psi(1) = C$ where $C = -0.5772156\ldots$ is the Euler-Mascheroni constant.

Another alternative is to replace $n_x(i)$ and $n_y(i)$ by the number of points with $\|x_i - x_j\| \leq \varepsilon_x(i)/2$ and $\|y_i - y_j\| \leq \varepsilon_y(i)/2$. The estimate for MI is then

$$\hat{I}_2(X, Y) = \psi(k) - \frac{1}{k} - \frac{1}{N} \sum_{i=1}^{N} [\psi(n_x(i)) + \psi(n_y(i))] + \psi(N). \tag{11}$$

In this paper this second estimator is used, which is the one implemented in [10]. Please check [9] for an extended explanation.

As can be noted, this MI estimator has a dependency on the value chosen for k (k-th nearest neighbor). As it is recommended in [12] for a tradeoff between variance and bias, in the examples, a mid-range value for k ($k = 6$) will be used.

2.3 Effective Variable Selection for Function Approximation Problems Using MI

The MI estimator detailed above will allow us to carry out the proposed variable selection method. It also gives the possibility of estimating the MI for groups of variables even when the number of data points we have at disposal is relatively small.

In the following it is reviewed how the MI can be used for variable selection, and it is presented the proposed method for variable selection in function approximation problems.

According to the definition of MI, $I(X, Y)$ gives the information that the group of variables X bring about Y. Any modelling problem would try to use this information and try to predict new values of Y given new values of X. As mentioned before, having unneeded variables can unnecessarily complicate the model. Furthermore the generalization capability can be decreased. Thus it is essential to select a right subset $X_G \subset X$ that comprises the same information that X has of Y. That is, we want to find a subset $X_G \subset X$ such that

$$I(X, Y) \cong I(X_G, Y). \tag{12}$$

We could directly try to evaluate $I(X_G, Y)$ for all the possible subsets X_G of X, and then select the smallest subset X'_G, whose $I(X_G, Y)$ is the highest. In this way irrelevant and redundant variables would not be selected in the optimal X'_G. This approach suggested in [13] and [1], and partially in [8] has two main drawbacks. First the number of possibilities for X_G is exponential in the number of input variables n (2^n possible subsets). Second, as the number of available data is limited, the robustness of the k-nearest neighbor MI estimator is also limited when taking into account too many input variables in X_G.

Other approaches could consider the MI of single input variables over the output variable $I(x_i, Y)$ to perform a ranking and use it as a filter to eliminate variables [7]. This approach is very good for avoiding irrelevant variables but does not consider redundant ones. For example two variables x_i and x_j can have a very high MI with respect the output variable Y, but using both of them can bring no more MI w.r.t the output variable. In this case $I(\{x_i, x_j\}, Y)$ is similar to $I(x_i, Y)$ and $I(x_j, Y)$, and thus one could be discarded.

A more robust approach would be to try selecting input variables as the MI with respect to the output variable of the selected subset increases. An iterative process would add a new variable to the current subset such that

$$I(\{X_G \cup x_i\}, Y) - I(X_G, Y), \tag{13}$$

is maximum over j. Nevertheless, as mentioned before, if the number of variables to be selected is high, the precision of the MI estimator can be lost. The results offered by the MI estimator, as exposed in [9] are optimal when the number of data points is very high, but in practice this is rarely the case.

Here it is proposed to adapt the method for discrete variables presented in [6] to function approximation problems (i.e. to continuous variables). An iterative backwards variable selection will be performed, starting from the complete set of variables X. The idea is to eliminate a variable x_i in the current selected subset $X_G \cup \{x_i\} \subset X$ if we estimate that $I(\{X_G \cup \{x_i\}\}, Y) = I(X_G, Y)$. To help in this iterative procedure, the concept of Markov blanket, adapted for this problem, will be used. We will suppose that this concept can be applied for the specific variable selection problem we deal with. The use of Markov blankets [15] implies strong conditioning between the variables. Nevertheless it will be relaxed to help us performing the variable selection.

Definition: Let M be a set of variables that do not contain x_i. We say that M is a Markov blanket for x_i if $I(\{M \cup x_i\}, X - \{M \cup x_i\}) \cong I(M, X - \{M \cup x_i\})$

Corollary: Let X_G be a subset of variables and x_i a variable in X_G. Assume that a subset M of X_G is a Markov blanket of x_i. Then $I(X_G, Y) \cong I(X_G - \{x_i\}, Y)$.

As we can see, the Markov blanket condition is stronger than the one we desire. It can be even a harder problem to find a Markov blanket of a variable in a set of variables that the variable selection problem itself. However it brings the idea on how to perform a more robust variable selection procedure. The difficult evaluation of $I(\{X_G \cup \{x_i\}\}, Y) = I(X_G, Y)$ to eliminate variables, will be transformed into estimating if x_i has a Markov blanket in X_G. Those x_i will be removed from the current X_G.

As already mentioned, calculating the Markov blanket of a variable in a set, or even trying to know if it exists is a very difficult task. Therefore it will be assumed that the Markov blanket exists, and we will derive a heuristic to guess the variables M that compose the Markov blanket of any variable x_i. The proposed algorithm is the following:

1. Calculate the MI between every two input variables $I(x_i, x_j)$
2. Starting from the complete set of input variables $X_G = X$, iterate:
   (a) For each variable x_i, let the candidate Markov blanket M_i be the set of p variables in X_G for which $I(x_i, x_j)$ is highest.
   (b) Compute for each x_i
   $$Loss_i = I(\{M_i \cup x_i\}, Y) - I(M, Y). \qquad (14)$$
   (c) Choose the x_i for which $Loss'_i$ is lowest and eliminate x'_i from X_G.

The procedure may be stopped after a fixed number of input variables are eliminated; alternatively it may be stopped when $Loss'_i$ reaches a certain threshold. This methodology is suboptimal in some aspects, but still offers a robust variable selection methodology. The Markov blanket selected for every variable is just an approximation and the number p of variables is fixed a priori. With respect to parameter p, high values can bring better chances that the pseudo-Markov blankets taken subsume real Markov blankets of the variables. However, the reliability of the MI estimator can be decreased. Considering this trade off, in general, a medium value of p should be considered. For problems with low number of data points, a lower value for p should be taken.

As we will see in the simulation section, the method proposed can outperform the other methods commented in this paper: it can therefore be a good solution for the key problem of variable selection in regression or function approximation problems.

3 Least-Squares Support Vector Machines

This section presents a brief introduction to the learning methodology used in the simulations. LS-SVMs are reformulations to standard SVMs, closely related to regularization networks and Gaussian processes but additionally emphasize and exploit primal-dual interpretations from optimization theory. LS-SVMs are a paradigm specially well suited for function approximation problems [4].

The LS-SVM model [14] is defined in its primal weight space by

$$\hat{y} = W^T \phi(X) + b, \tag{15}$$

where W^T and b are the parameters of the model, $\phi(X)$ is a function that maps the input space into a higher-dimensional feature space and X is the n-dimensional vector of inputs x_i. In Least Squares Support Vector Machines for function approximation, the following optimization problem is formulated,

$$\min_{W,b,e} J(W,e) = \frac{1}{2} W^T W + \gamma \frac{1}{2} \sum_{i=1}^{N} e_i^2, \tag{16}$$

subject to the equality constraints (inequality constraints in the case of SVMs)

$$e_i = y_i - \hat{y}_i(X_i), i = 1 \ldots N. \tag{17}$$

Solving this optimization problem in dual space leads to finding the λ_i and b coefficients in the following solution

$$\hat{y}_i = \sum_{i=1}^{N} \lambda_i K(X, X_i) + b, \tag{18}$$

where the function $K(X, X_i)$ is the kernel function defined as the dot product between the $\phi(X)$ and $\phi(X_i)$ mappings. If we consider Gaussian kernels, the width of the kernel σ_i together with the regularization parameter γ, are the hyper-parameters of the problem. Note that in the case of Gaussian kernels, the obtained model resembles Radial Basis Function Networks (RBFN), with the particularities that there is an RBF node per data point, and that overfitting is controlled by a regularization parameter instead of by reducing the number of kernels [7]. In LS-SVM, the hyper-parameters of the model are usually optimized by cross-validation.

4 Simulations

This section presents the application of the variable selection method proposed in this paper to a significant example. The MI estimator in [10] will be used in all the simulations. A LS-SVM Matlab toolbox can be found in [14]. The error measure used here is the Normalized Mean Square Error, NMSE [7].

The example considered has been taken from [7] and is a spectrometric data set coming from the food industry. This type of data form vectors with a large number of exploitable variables. Usually however, only a small subset of them is required to build a good model. The “tecator meat” data set consist of 100 spectral input variables and one output variable (the original data set has three, but we consider only the fat content). It relates to the determination of the fat content of meat samples analyzed by near infrared transmittance spectroscopy. This data set contains 172 training spectra and 43 test spectra. As in [7], the spectra are reduced to zero mean and unit variance. Also to avoid loosing information, the original mean and standard deviation are kept as two additional variables. A selection of training spectra is shown in Figure 1.

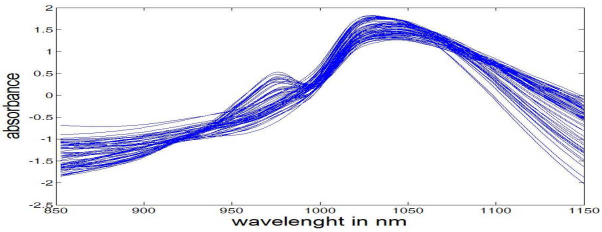

Fig. 1. A selection of the spectra from the “tecator meat” data set

In [7], first an initial subset of 16 variables is selected. Using them, a LS-SVM is optimized using cross-validation. The test NMSE obtained for this case is 0.0040. Next, all 2^{16} possible subsets of variables are tested, checking which subset of those 16 variables brings the highest MI with respect to the output variable. The optimal subset found had 8 variables. The test NMSE on the LS-SVM model is 0.0049. Note that in the comparisons presented in this section, there are some differences with the results shown in [7], since the second estimator $\hat{I}_2(X, Y)$ was used here.

Next we proceed by selecting 16 most significant variables using the approach proposed in this paper (for example using $p = 6$). The test NMSE obtained after the optimization of the LS-SVM is 0.0022. As can be seen, the initial subset selected by our method has a higher performance than the one selected by the approach in [7]. Forcing the number of variables to 8 (with parameter $p = 6$) to compare with the sub-set selected in [7], the test NMSE was 0.0024.

With respect to the value of the parameter p, similar results of NMSE with 16 variables were given by other values of p both in training and in test, showing the efficiency of the method eliminating irrelevant and redundant variables. For very low values of p, the training and test errors were even lower (test NMSE for $p = 1$ is 0.0016). It is noticeable that the variables selected for different values of p are remarkably different. This is due to the high level of redundancies that exist among the input variables and also to the low number of data points that we handle in this problem. In problems like this one, there are usually several possibilities of suboptimal subsets of variables, instead of a single optimal set.

For this problem the results obtained show that lower values of p provide better results both for training and test data sets. But during the filtering process, there are also some details that suggest discarding higher values for p. The loss function (see Eq. 14) for high values of p does not follow an increasing trend. We have a low number of data points and very high redundancies among the input variables. Thus the MI estimator can provide confusing results.

Taking $p = 1$, we will now look for a final pseudo-optimal subset of variables. As mentioned before, we could have in principle two possible stopping criteria in our algorithm. One is to specify a number of input variables to be selected. This number could be chosen according to the results of the model on the subsets of variables. In this case, the method would become a mixture of filter and wrapper. A good stopping criterion would be to select the subset that brought the best training error after the cross-validation optimization of the LS-SVM model.

Nevertheless, in order not to loose the filtering advantage of the method, a possible heuristics is to select a limit in the loss function in Eq. 14. In Figure 2 the evolution of the loss function in Eq. 14 in the iterative process is shown, for $p = 1$. From this graph we can get an idea on how much information is lost as variables are discarded in the iterative process. We saw that selecting a subset of 16 variables leads to good performance. For this value, the $Loss'$ is around 0.23. Consequently we establish a limit of 0.26, that corresponds to the next peak in the graph. From this threshold no more variables will be eliminated. Finally, the optimal subset contains 11 variables and the performances are test NMSE = 0.0016 and training NMSE = 0.0010. Other tests showed that further elimination of variables leads to a small worsening of the performances (both in training and test), increasing as more variables are discarded.

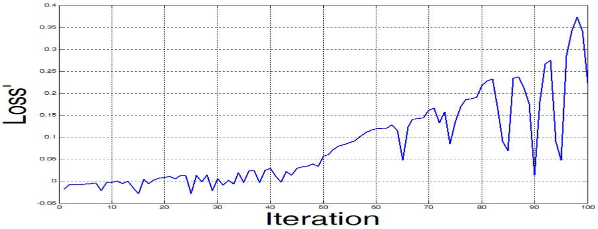

Fig. 2. Evolution of the $Loss'$ function in the run of the algorithm with $p = 1$

5 Conclusions and Further Work

In this paper it was presented an effective backward variable selection method for function approximation problems, based on the concept of MI, adapted from a previous method for classification problems. It is a robust approach compared to other ones from the literature, thanks to the use of the Markov blanket concept. As further work, we intend to design a general methodology to select the number of input variables to be discarded. Furthermore the application of the method in other domains, in particular in time series prediction, will be investigated to help solving the difficult problem of deciding which variables should intervene in the prediction model.

Acknowledgements

This work is partially supported by the Spanish CICYT Project TIN 2004-01419. This work was partially done during the stay of L. J. Herrera in the UCL, under the supervision of M. Verleysen, Research Director of the Belgian FNRS.

References

1. B.V. Bonnlander, A.S. Weigend, “Selecting input variables using mutual information and nonparametric density estimation, in Proc. of the ISANN 2004, Taiwan, 1994, pp. 42-50
2. B. Schoelkopf, A. Smola: Learning with Kernels. Cambridge, MA: MIT Press, 2002
3. S. Haykin, Neural Networks, Prentice Hall, New Jersey, 1998
4. J.A.K . Suykens, T. Van Gestel, J. De Brabanter, J. De Moor, B., Vandewalle: Least Squares Support Vector Machines, World Scientific, Singapore, 2002
5. L.J. Herrera, H. Pomares, I. Rojas, O. Valenzuela, A. Prieto: “TaSe, a Taylor Series Based Fuzzy System Model that Combines Interpretability and Accuracy”. Fuzzy Sets and Systems, vol. 153, No.3, 2005, 403-427
6. D. Koller, M. Sahami, “Toward Optimal Feature Selection”, in Proc. Int. Conf. on Machine Learning, 1996, pp. 284-292
7. F.Rossi, A. Lendasse, D. Franois, V. Wertz, M. Verleysen: “Mutual Information for the selection of relevant variables in spectrometric nonlinear modeling”, Chem. and Int. Lab. Syst., 2005, In Press
8. N. Benoudjit, D. Franois, M. Meurens, M. Verleysen: “Spectrophotometric variable selection by mutual information”, Chem. and Int. Lab. Syst., vol. 74, 2004, pp. 243-251
9. A. Kraskov, H. Stgbauer, P. Grassberger, “Estimating mutual information”, Phys.Rev.,E 69, 2004, 066138
10. http://www.klab.caltech.edu/~kraskov/MILCA/
11. T.M. Cover, J.A. Thomas, Elements of Information Theory, Wiley, New York, 1991
12. S. Harald, K. Alexander, A.A. Sergey, G. Peter, “Least dependent component analysis based on mutual information”, Phys. Rev., E 70, 2004, 066123
13. A. Sorjamaa, J. Hao, A. Lendasse, “Mutual Information and k-Nearest Neighbors Approxi-mator for Time Series Prediction”, ICANN 2005, LNCS 3697, pp. 553 - 558
14. http://www.esat.kuleuven.ac.be/sista/lssvmlab/
15. J. Pearl: Probabilistic Reasoning in Intelligent Systems. Morgan Kaufmann, CA, 1988