View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by idUS. Depósito de Investigación Universidad de Sevilla

Copyright © 2002 IFAC
15th Triennial World Congress, Barcelona, Spain

ELSEVIER
IFAC
PUBLICATIONS
www.elsevier.com/locate/ifac

INPUT VARIABLE SELECTION FOR FORECASTING MODELS

Manuel R. Arahal, Alfonso Cepeda, Eduardo F. Camacho

Depto. Ingeniería de Sistemas y Automática. Universidad de Sevilla

Abstract: The selection of input variables plays a crucial role when modelling time series. For nonlinear models there are not well developed techniques such as AIC and other criteria that work with linear models. In the case of Short Term Load Forecasting (STLF) generalization is greatly influenced by such selection. In this paper two approaches are compared using real data from a Spanish utility company. The models used are neural networks although the algorithms can be used with other nonlinear models. The experiments show that that input variable selection affects the performance of forecasting models and thus should be treated as a generalization problem. *Copyright © 2002 IFAC*

Keywords: Autocorrelation, Autoregressive models, Neural networks, Time-series analysis.

1. INTRODUCTION

Sometimes a model is needed to forecast the future behavior of a time series. Past values of the time series and past or forecasted values of other variables can potentially be used as inputs of the model. When a group of (mostly) independent variables are available, variable selection has to be performed in order to avoid using variables that have little influence on the forecast and, at the same time, do not neglect useful ones.

Forward inclusion and Backward elimination by means of correlation matrix are widely used methods. Although it has been used successfully in nonlinear scenarios, it has to be noted that correlation seeks for almost linear relationships among variables. When other relationships are present, such as quadratic, the correlation test can yield wrong results.

A brute force approach can in some cases provide the best results. Either by step-wise inclusion or deletion of variables at each stage a full model has to be developed. The performance of the model is tested using some control data and allows to determine the best set of variables.

This method has a number of obvious drawbacks. First, it is very time consuming, and does not scale well with both the size/complexity of the model and the number of potential variables. Second, large amounts of control data are needed, otherwise the improvements in the variable selection procedure may not have correspondence in the later use of the model. In many situations data is scarce disallowing the use of such method.

Short term electrical load forecasting (STELF) is a task in which nonlinearities, model complexity and scarce data are combined. These problems have been addressed in (Yuan and Fine, 1998). The difference index presented there is used with some modifications in this paper and compared against the brute force approach. A comparison is carried out using actual data from a Spanish utility company. Generalization issues are tackled with special care in the experiments that simulate the use of the models under different conditions.

The algorithms for variable selection are presented next. The forecasting problem and the available data used for the comparison are shown in section 3. Following this, the results will be detailed and the conclusions presented.

2. INPUT VARIABLE SELECTION ALGORITHMS

Two algorithms for variable selection will be described here. The first one consists on step-wise inclusion of variables in neural models.

In both cases the available past data is split in two sets: one for constructing models and the other for testing their goodness. Evaluation can be seen as a function that assigns a figure of merit to each model, allowing their classification. Let us denote by $\mathscr{E}(M)$ the value of the evaluation function for model M.

Also, the set of potential variables is first pruned removing variables that have a clear quasi-linear relationship among them. Let us denote by n the number of remaining potential variables. Our problem is to decide which of the potential variables should be used as inputs in the model.

2.1 *Step-wise inclusion of variables*

This strategy adds one variable at a time to the set of selected variables. This variable is selected probing all possible models that result from the addition of any variable of the set of potential variables.

The first stage of the selection algorithm is the construction models M_1^1 to M_n^1 where de superindex denotes the number of input variables used by the model and the subindex is the variable last added to that model. The model that achieves the best performance indicates which variable should be first selected. Mathematically $v^1 = \arg\min_j \mathscr{E}(M_j^1)$. Variable v^1 is then eliminated from the set of potential variables and included in the set of selected ones.

In a second step $n-1$ models with two input variables are considered and a new variable selected.

The procedure can go on forever because theoretically a model with an extra variable can perform as well as the original model. We know however that generalization degrades when a unnecessary complexity is used.

The problem of when to stop is sometimes solved plotting the goodness of the successive models against the number of variables. Small changes are expected when including dispensable variables whereas significant improvements occur when an important variable is added. For this reason the curve has a larger slope when the first variables are added and flattens successively. The elbow of the curve has been many times signaled as the correct number of variables. As we will see in the experiments, this choice is coupled with the performance function considered.

When using neural networks as models other considerations have to be made. First of all, the effects of random initial values of weights and training stopping criterion cause networks with the same inputs and same training data to perform differently. Second, the number of nodes/connections also affect the capacity of resulting model. To avoid these problem a number of networks should be generated and used as an ensemble averaging their outputs. The networks can be trained using different algorithms and do not have to share the same structure or size.

2.2 *Index based selection*

The second algorithm to be compared here is based on a selection index that can be obtained directly from data without the need of constructing any model. The index proposed in (Yuan and Fine, 1998), is a difference-based estimator of residual variance. We will use it here with minor modifications.

Given a set of available data formed by a pairs of input vectors and target values $(\underline{x}_i, t_i)$. The input vector is composed of n different variables. Let us denote by $x_{i,j}$ the j-th component of the i-th data pair.

For each potential input variable j an indication of its usefulness can be obtained by means of index I_j. This index measures the changes in the target associated with changes in the j-th variable. To do so, all pairs $(x_{i,j}, t_i)$ of variable and target are considered and reordered so that the new pairs $(x_{p,j}, t_p)$ verify that $x_{p,j} \leq x_{q,j}$ for any $p < q$.

The index is then calculated as

$$I_j = \sum_{h=1}^{N} \left| t_{(h+1)} - t_{(h)} \right| \tag{1}$$

where N is the number of data pairs.

The way the index is used for selection is described now. Potential input variables are first divided among groups, containing each group variables that are correlated. The index is obtained for all the variables in each group. The one with lower index is considered the best representative of the group and is selected.

Proceeding in this manner, the chance of selecting redundant ("collinear") variables is ruled out.

This procedure does not construct any model, speeding up the selection process.

2.3 *Step-wise index based selection*

We propose a modification of the index based procedure. The method consists on calculating the difference index for all variables as before. Now, after the first variable has been selected it is eliminated from the set of potential input variables. A model is constructed using this one variable. The output of the model $\hat{t}^1$ is an estimation of the target variable. We use the error of the model (the residuals) as a new target: $t^2 = t - \hat{t}^1$. New difference index are then calculated for the

elements of the set of potential input variables, and the process repeats itself.

The number of models to be constructed is less than in the brute force approach outlined in section 2.1.

3. FORECASTING PROBLEM

The problem of short term load forecasting refers to the prediction of the energy demand within a horizon of 24 to 48 hours. Different approaches have been used for this problem ranging from linear to neural nets and fuzzy sets. Many papers can be found in the literature, especially in the IEEE Transactions on Power Systems (see for instance (Chow and Leung, 1997)).

In some cases (Arahal and Camacho, 2001), the hourly load forecasting problem is divided in two: the prediction of the integrated demand for a day and the normalized hourly load curve of the day. Normalized load curves are used to distribute the integrated predicted demand among the 24 hours of the day. In this paper the forecasting of the load curve of the day is not considered. The algorithms under consideration attempt to predict the total (i.e. integrated) energy demanded over 24 hours for the next two or three days.

Some notation is needed in order to present the algorithm. The hourly load l_h^d demanded at hour h of a day d is usually the variable used for electrical companies for forecasting purposes. However, the utility used as a test bed is mainly interested in predicting the integrated load for a day; that is:

$$c(j) = \sum_{h=1}^{h=24} l_h^j \tag{2}$$

This integrated load will be referred to as "daily load". The sequence of values $\{c(j)\}$ for $j = 0,1,...,n_d$ constitutes the data base of electrical load. Similar data base exists for the temperatures measured in the region where the electrical energy is supplied and for other variables of interest.

Suppose that we wish to predict the daily load for day k. Different cases appear:

(1) If k is a Monday, the prediction has to be made the previous Friday $k-3$. The advance in the forecast is $d = 3$ days.
(2) If k is a Sunday, the prediction has to be made the previous Friday $k-2$. The advance in the forecast is $d = 2$ days.
(3) If k is the day after a holiday, the prediction has to be made the working day before the holiday, like in the previous case, except in Mondays following a holiday in Friday, where the advance is $d = 4$ days.
(4) In the rest of the cases the prediction is done the previous day $k-1$. The advance in the forecast is $d = 1$ day.

From the cases above it is clear that the values of $c(j)$ are known for all $j < k-d$. These past loads can be used to generate a prediction $\hat{c}(k)$. The hat allows to distinguish between the prediction and the actual value of the load $c(k)$, which is only known the day $k+1$.

The objective of a forecaster is to produce $\hat{c}(k)$ for some days ahead (see above cases). A figure of merit often used is the prediction error defined as:

$$ep(k) = 100\frac{c(k)-\hat{c}(k)}{c(k)} \tag{3}$$

this quantity can be measured after day k and is an indication of how good the prediction was. Usually a set of past days with known load demand are used to evaluate the goodness of a predictor using the root mean squared error

$$E_{rms} = \sqrt{\frac{1}{n_d}\sum_{k=1}^{n_d} ep^2(k)} \tag{4}$$

3.1 *Data set*

The data has been supplied by a Spanish electrical company that does not wish to be further identified. It consists of hourly power demand for a vast region during several years (see (Pavón and Arahal, 2000)). In order to maintain the confidentiality of the data, the plots given in the paper correspond to a unidentified period of time and the vertical scale is normalized.

The available data set has been split in three parts (see figure 1):

- Working set.Contains four years of dayly load. It would be split in two disjoint sets, namely Construction Set (CS) used for the training of the neural networks and Validation Set (VS) used to signal when to stop the training of the networks and to select the most appropriate nets among a group of nets.
- Test set (TS). Contains one year of data which is posterior to WS that will be used to compare the different selection algorithms.

The selection of the variables of the algorithm will be carried out using just the data in WS. On the other hand, the selection algorithms will be tested using data from the TS. It has to be remarked that data from TS will not be used (or even disclosed) after the algorithms have chosen their set of input variables.

4. EXPERIMENTAL RESULTS

The variables considered as potential inputs are:

$v_1 = ctdamt$ Load of the previous day of the same type.
$v_2 = m7da$ Average over past seven days.
$v_3 = m14da$ Average over past fourteen days.

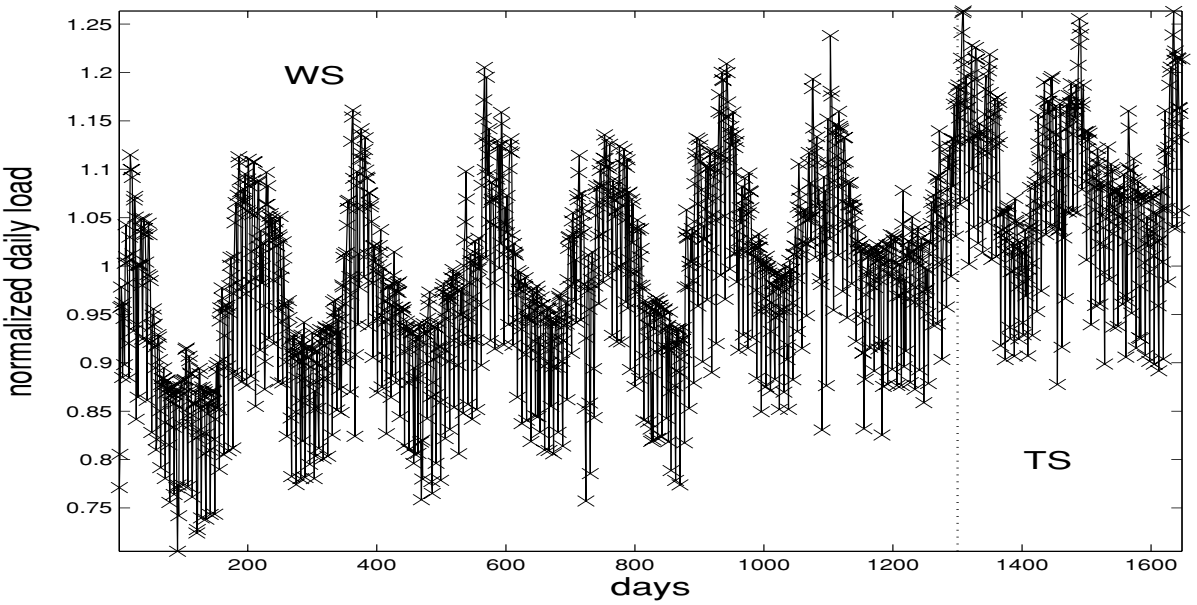

Fig. 1. Normalized electrical daily load in the working set WS=CS+VS and in the testing set TS.

$v_4 = mdaa$ Average over a week lagged a year.
$v_5 = mm2ma$ Average over a month lagged two months.
$v_6 = mm3ma$ Average over a month lagged three months.
$v_7 = mm4ma$ Average over a month lagged four months.
$v_8 = mm6ma$ Average over a month lagged six months.
$v_9 = t$ Forecasted average daily temperature.
$v_{10} = mt2da$ Average temperature over past two days.
$v_{11} = td$ Type of day: 0-holiday {1 – 7 } Sunday through Saturday.
$v_{12} = da$ Day of the year.

In order to test the selection of input variables a number of models have been obtained. Each model differs just in the input vector used. The models are neural networks of one-hidden layer with 15 nodes trained with the Levenberg-Marquardt algorithm in MATLAB.

Training is performed using just data from CS to adjust the networks parameters. The E_{rms} in the VS is monitored to stop the training iterations. The CS consists of a 20% of the data points in WS randomly selected. The VS is the remaining 80%.

Since the initial weights of the network can affect the results 50 networks are constructed instead of just one.

Denoting by J_{CS} the E_{rms} in the CS and by J_{VS} E_{rms} in the VS a new figure of merit is introduced

$$J_{WS} = 0.2J_{CS} + 0.8J_{VS} \tag{5}$$

The 15 nets that provide the smaller value of J_{WS} are just considered. The results shown in the following are an average of such 15 nets.

The number of possible variables is twelve, so it would be necessary to generate $12+11+...+2 = 78$ different models to get the complete tree of input variables relevance. This is excessively time consuming.

4.1 *Step-wise inclusion*

The next table shows the average value of J_{WS} for the best 15 nets out of 50. Each row correspond to an input variable and each column to a stage of the inclusion algorithm.

It can be seen that in the first stage (first column) all input variables are considered one by one. The smaller value appears in the first row, consequently v_1 is selected from the set of potential variables and used in subsequent stages.

In the second stage v_1 is combined one by one with the rest of potential variables. The smaller value of J_{WS} appears in the last row, corresponding to v_{11}. This is thus the second variable to be selected. It is interesting to note that, at this stage, all models yield better performance that in the previous, as expected (see section 2.1).

	1	2	3	4	5
v_1	4.94				
v_5	8.31	4.68	4.23	3.78	3.69
v_6	8.94	4.81	4.55	3.97	3.72
v_9	7.57	4.78	3.99	3.73	
v_{10}	7.09	4.69	3.97		
v_{11}	7.21	4.65			

Table 1. J_{WS} for the different models at each stage of the selection method.

Using this method the variables are selected in this order: v_1, v_{11}, v_{10}, v_9 and v_5

4.2 *Index based selection*

Using linear correlation matrix it is possible to reduce the number of variables analyzed making groups of similar variables as commented above. Doing so the

number of variables can be reduced to four $v_1 = ctdamt$, $v_4 mdaa$, $v_9 = t$ and $v_{11} = td$. These variables have probed as the best variables from their groups.

It has to be commented that this method and the next have a practical problem with variables that take discrete values such as td. For this reason da is normally selected by this method. However, in order to compare with the previous one td has been selected instead of da.

The index is calculated for each of the above pre-selected variables using data of WS. The results are shown in the next table.

v_1	v_4	v_9	v_{11}
20.64	37.58	32.96	36.03

Table 2. Values of I for the pre-selected variables.

4.3 *Step-wise index based selection*

The second index based method creates intermediate models with input variables elected via the I_j. The results are shown in the next table. Just the significant portion of the table is shown for the sake of clarity of presentation.

	1	2	3	4	5
v_1	21.2				
v_2	31.6	23.8	21.0	21.3	17.6
v_3	33.5	22.6	20.6		
v_8	34.2	23.5	21.0	20.9	17.4
v_9	32.3	22.4	20.6	20.6	
v_{11}	36.4	21.9			

Table 3. Index for the different variables at each stage of the third selection method.

In the first stage the variable with lower index is v_1 and thus is selected. Similarly the rest of variables considered are ordered yielding the sequence: v_1, v_{11}, v_3, v_9 and v_2.

5. COMPARISON

A different way of looking at the results of the the different algorithms is to test the improvements (if there is any) in the models when a new input variable is added.

In figures 2, 3 and 4 the average value of J_{CS} (dark) and J_{VS} (fair gray) is plotted for the three selection methods against the number of input variables.

According to the two first methods the best results are obtained with five input variables. The third method suggests that four variables are best.

We have to keep in mind that these selection has been performed using past data. The goodness of the models when confronted with new data has yet to be tested. To this end we disclose the TS set and use every

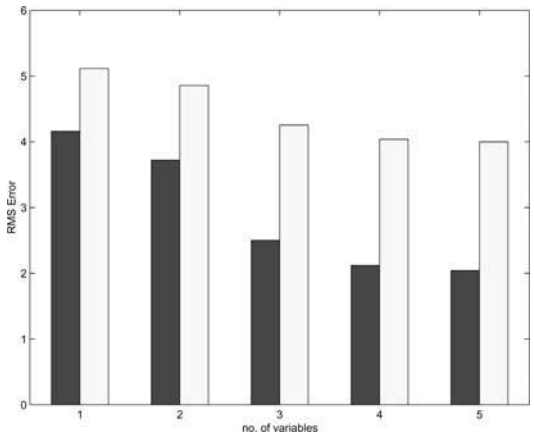

Fig. 2. J_{CS} (dark) and J_{VS} (fair gray) for the models derived of the first selection method.

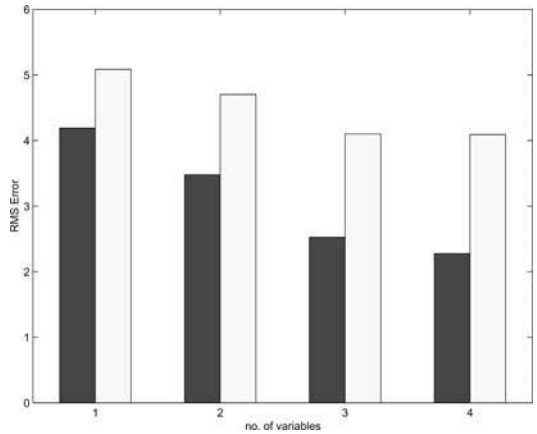

Fig. 3. J_{CS} (dark) and J_{VS} (fair gray) for the models derived of the second selection method.

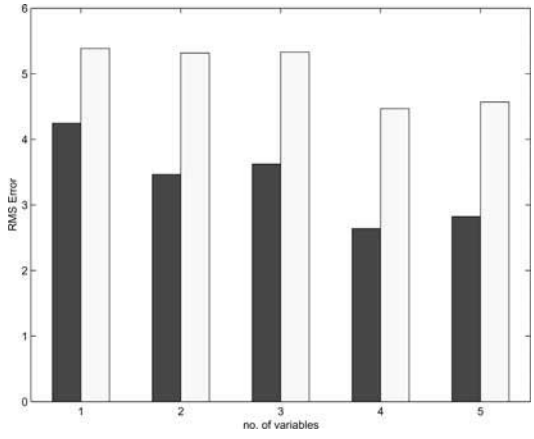

Fig. 4. J_{CS} (dark) and J_{VS} (fair gray) for the models derived of the third selection method.

model to forecast in this new conditions. This test is a though one since no additional training was performed and the TS set span one whole year in the future.

For each model, the above selected 15 nets are used with the new data set, obtaining E_{rms} in the usual way. The average of these root mean squared errors is J_{TS}, which is plotted in figure 5. The square mark

correspond to method 1, the triangle to method 2 and the circle to method 3.

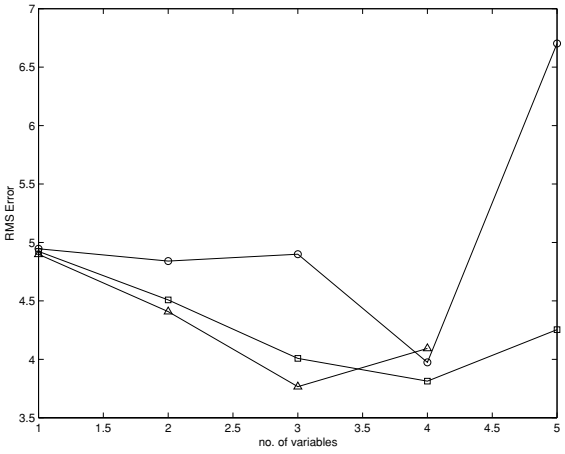

Fig. 5. J_{TS} versus the number of input variables for each model using variables selected by the first method (squares), the second (triangles) and the third method (circles).

It can be seen that care has to be taken since the best results are actually obtained with just three input variables. It is also interesting to notice that for four variables the three methods obtain very close and acceptable solutions.

A conclusion we draw in the light of this comparison is that index-based input variable selection is easy to perform and less time consuming and can (as in this case) lead to successful selection.

Also, it is worth noting that the way the neural nets have been trained and selected greatly affects generalization. At the time of closing this draft version we have yet no complete results on how this selection should be performed so as to avoid (as much as that is possible) the mismatch between $\mathscr{E}(M)$ and J_{TS}. For instance, prior to the currently used evaluation function other percentages of CS and VS where used with poorer results.

It seems that the first index-based methods avoids the overparameterization trap by using geometrical information whereas model-based selection methods need to be very careful in this respect.

6. CONCLUSIONS

Three methods for input variable selection have been compared using real data in the problem of short term load forecasting and the problems associated with each approach commented.

From the results given it is clear that input variable selection can seriously affect generalization. Moreover, it has been shown that the problem of input variable selection, neural model assessment and generalization are coupled.

Further research will be aimed at developing neural model testing procedures that would select nets that have the best generalization capabilities.

7. REFERENCES

Arahal, M.R. and E.F. Camacho (2001). Neighbor histories for short term load forecasting. *Proceedings of the European Control Conference* pp. 2796–2801.

Chow, T.W.S. and C.T. Leung (1997). Neural network based short-term load forecasting usign weather compensation. *IEEE Transactions on on Power Systems* **11**, 1736–1742.

Pavón, F. and M.R. Arahal (2000). Predicción a corto plazo de la demanda de la energía eléctrica con siconel. *(in spanish) Actas de las XXI Jornadas de Automática, CD-ROM* **1**, ja00_002.

Yuan, J.-L. and T.L. Fine (1998). Neural-network design for small training sets of high dimension. *IEEE Transactions on neural networks* **9**, 266–280.