Pruned Lazy Learning Models for Time Series Prediction

Antti Sorjamaa[1], Amaury Lendasse[1] and Michel Verleysen[2] *

1- Helsinki University of Technology - Neural Networks Research Center
P.O. Box 5400, FI-02015 - Finland
2- Université catholique de Louvain - Machine Learning Group
Place du Levant, 3, B-1348 LLN - Belgium

Abstract. This paper presents two improvements of Lazy Learning. Both methods include input selection and are applied to long-term prediction of time series. First method is based on an iterative pruning of the inputs and the second one is performing a brute force search in the possible set of inputs using a *k*-NN approximator. Two benchmarks are used to illustrate the efficiency of these two methods: the Santa Fe A time series and the CATS Benchmark time series.

1 Introduction

Time series forecasting is a challenge in many fields. In finance, one forecasts stock exchange courses or stock market indices; data processing specialists forecast the flow of information on their networks; producers of electricity forecast the load of the following day. The common point to their problems is the following: how can one analyse and use the past to predict the future?

Many techniques exist: linear methods such as ARX, ARMA, etc. [1], and nonlinear ones such as artificial neural networks [2, 3]. In general, these methods try to build a model of the process. The model is then used on the last values of the series to predict future values. The common difficulty to all methods is the determination of sufficient and necessary information for a good prediction. The nonlinear model that is used in this paper is based on a linear piecewise approximation method called the Lazy Learning [4, 5].

This method has many advantages: it is very fast and doesn't suffer from the problem of local minima (in example some neural networks give the minimum that is not a global one, but a local instead). Lazy Learning also provides good approximations for time series prediction. Input selection is a problem common to all approximators and, in this paper, we will present two improvements, which include the selection of the inputs in the Lazy Learning method itself.

The Lazy learning is presented in Section 2 and the improvements in Section 3. Finally, the results obtained with the Lazy Learning and both its improvements are summarized in Section 4.

[*] Part the work of A. Sorjamaa and A. Lendasse is supported by the project of New Information Processing Principles, 44886, of the Academy of Finland. M. Verleysen is a Senior Research Associate of the Belgian National Fund for Scientific Research.

2 Lazy Learning Model

Lazy Learning model (LL) is a linear piecewise approximation model based on PRESS statistics and recursive least squares algorithm introduced by Aha [4]. Around each sample, the *k*-nearest neighbors are determined and used to build a local linear model. This approximation model is the following:

$$\hat{y} = LL(x_1, x_2, \ldots, x_n) \qquad (1)$$

with $\hat{y}$ the approximation of the real output y and x_1 to x_n the n selected inputs. The number of inputs and the inputs itself must be determined *a priori* in LL models.

The optimization of the number of neighbors is crucial. When the number of neighbors is small, local linearity assumption is valid, but small number of data won't allow building an accurate model. On the contrary, if the number of neighbors is large, local linearity assumption is not valid anymore but the model is more accurate.

For an increasing size of the neighborhood, a leave-one-out (LOO) procedure is used to evaluate the generalization error of each different LL model [6]. LOO procedure takes each of the neighbors out one at a time. The neighbors that remain are used to build the local linear model and the error is evaluated with the data taken out. The generalization error is the average of the model error computed with each neighbor absent.

Once the generalization error for each neighborhood size is evaluated, the number of neighbors that minimizes this generalization error is selected.

The main advantages of the LL models are the simplicity of the model itself and the low computational load. Furthermore, the local model can be built only around the ones for which the approximation is requested.

Unfortunately, as mentioned above, the inputs have to be known *a priori*. Several sets of inputs have to be used to select the optimal one. The error has to be evaluated around each data point for each set of inputs. The optimal input set is the one that minimizes the generalization error. This procedure is long and reduces considerably the advantages of the Lazy Learning. In Section 3, we will present two different methods, which include the selection of the inputs in the Lazy Learning method itself.

3 Input selection methodology

3.1 Pruned Lazy Learning Model

Pruned Lazy Learning method (PLL) is a modified version of the LL that has been presented in the previous section. This method leaves out (prunes) the least important inputs in the same way than widely known backward pruning algorithm. The initial model is build according to (1) with all the available inputs. Then each input from x_1 to x_n is taken out one at a time. The basic LL is applied to each pruned input set and the generalization error is obtained. Then the pruning that minimizes the generalization error is selected and the corresponding input is permanently taken away. This operation is repeated until an optimal set of inputs is found. This optimal set of inputs is the one that minimizes the generalization error obtained with the Leave-one-Out procedure.

With this method at most *n*(*n*-1)/2 PLL models are built and evaluated. This is significantly less than the number of possible input sets that can be built (2^*n*). However it is not guaranteed that this selected input set is the optimum. This disadvantage is common to all pruning methods [2].

3.2 *K*-NN Approximator

In this method, all the 2^*n* possible input sets are built and evaluated. To avoid an unrealistic computational time, local constant models are used instead of local linear models. So the output of (1) is the arithmetic average of the outputs of the *k* nearest neighbors (with *k* selected with the LOO procedure). This is an extended version of the traditional *k*-NN, which is used for classification tasks [7]. The selected set of inputs will be correct if this approximator is able to provide an enough accurate approximation of the output. If this is the case, a comparison between the 2^*n* tested input sets, will be valid.

Function approximation capability of the *k*-NN is illustrated with a toy example. Fig. 1. represents an approximation obtained with the *k*-NN. The approximated function is *y*=sin(*x*)+sin(5**x*)+sin(15**x*)+*e* with *e* an uniform noise in [-0.5, 0.5]. The inputs that are selected by *k*-NN are afterwards used with normal LL model.

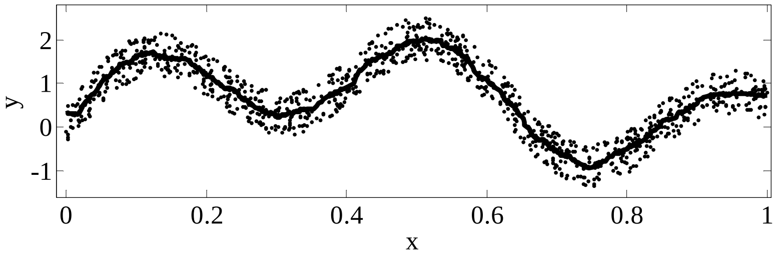


Fig. 1: Toy Example and its approximation (solid line).

4 Applications

4.1 Santa Fe A Benchmark

We illustrate the methods described in the previous sections on a standard benchmark in time-series prediction. The Santa Fe A time series [3] has been chosen mainly because of the large number of data available for the training stage (1000) as well as for the test stage (9000). The learning set is represented in Fig. 2.

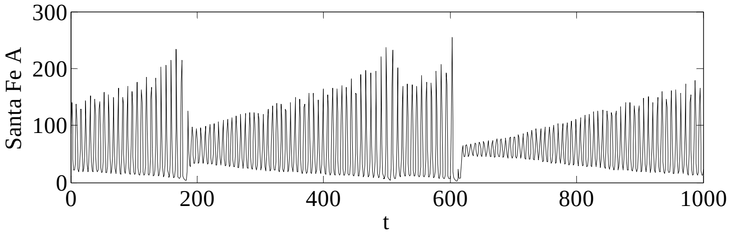


Fig. 2: Santa Fe A time series.

The results obtained with LL, PLL and *k*-NN + LL are summarized in Table 1.

	Learning		Calculation time	Test	Prediction 40 steps
	k	LOO error	Minutes	MSE	MSE
LL	56	42.32	2.58	42.0746	1765.6
LL pruned	59	19.42	13.95	20.6037	148.37
k-NN	3	57.71	33.78	53.5387	1252.1
k-NN + LL	15	33.57	0.20	31.4548	1770.1

Table 1: Results for Santa Fe A.

The selected inputs for continuous LL are (t-1, t-2, t-3), for LL pruned (t-1, t-2, t-3, t-11) and for *k*-NN (t-1, t-2, t-12). In *k*-NN + LL the inputs selected with *k*-NN are used with the Lazy Learning method for comparison.

The best method is LL pruned and it is used for a long-term prediction. Input set is selected only once and prediction is done recursively (running forecast). The results for the long-term prediction are presented in Fig. 3 and in the last column of Table 1.

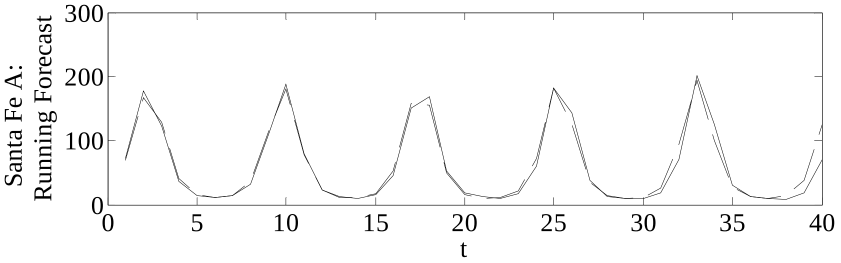


Fig. 3: Results obtained with running forecast on Santa Fe A test set.

4.2 **CATS Benchmark**

The CATS Benchmark is an artificial time series with 5000 data [8]. Within those, 100 values are missing. These missing values are divided into 5 blocks: from 981 to 1000, from 1981 to 2000, from 2981 to 3000, from 3981 to 4000 and from 4981 to 5000. Dataset from 1 to 980 is illustrated in Fig. 4.

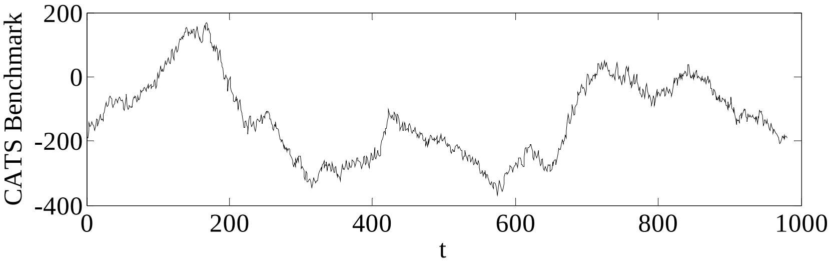


Fig. 4: CATS Benchmark time series.

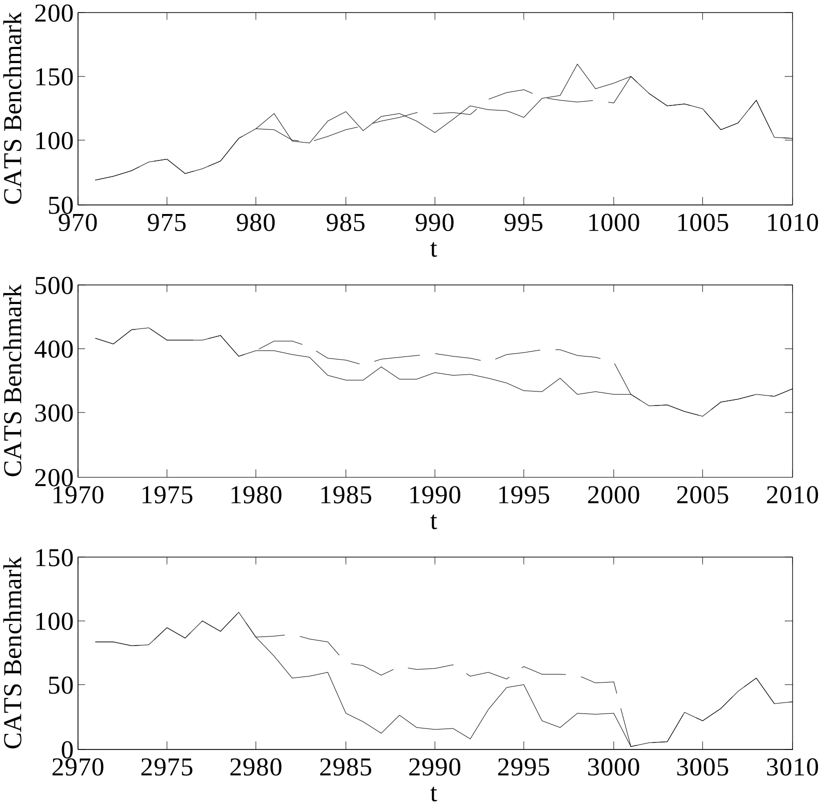
The results obtained with LL, PLL and *k*-NN + LL are summarized in Table 2.

	CATS Learning		Calculation time	CATS test
	k	LOO error	Minutes	MSE
LL	242	96.18	144.37	87.67
LL pruned	242	96.18	249.81	87.67
k-NN	9	101.60	494.15	98.41
k-NN + LL	234	99.74	3.29	92.23

Table 2: Results for CATS Benchmark.

The selected inputs are (t-1 to t-16) for continuous LL and LL pruned, and for *k*-NN (t-1, t-2, t-3, t-4, t-6, t-7, t-9, t-10, t-11, t-13, t-14, t-16).

The best method is again LL pruned and it is used to predict the 100 missing values and the results are presented in Fig. 5.



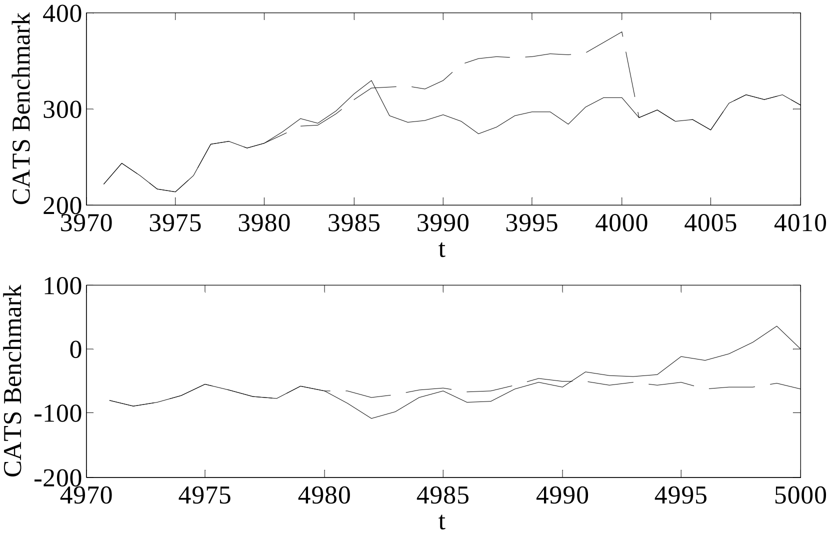


Fig. 5: CATS Benchmark: real values (solid) and approximations (dashed).

5 Conclusions

Two extensions of LL, which include input selection, have been presented and both of them provide satisfactory performance. Nevertheless, the most promising method is PLL, which is not only the most accurate but also the fastest.

A comparison between the selected sets of inputs performed by PLL and other input selection methods (for example methods which use mutual information) will be studied next.

References

[1] L. Ljung, System Identification—Theory for User, Prentice-Hall, Englewood CliPs, NJ, 1987.

[2] Cottrell, B.Y. Girard, M. Mangeas, C. Muller, Neural modeling for time series: a statistical stepwise method for weight elimination, IEEE Trans. Neural Networks 6 (6), pages 1355–1364, 1995.

[3] A.S. Weigend, N.A. Gershenfeld, Times Series Prediction: Forecasting the future and Understanding the Past, Addison-Wesley, Reading, MA, 1994.

[4] D. W. Aha, Editorial of Special Issue on Lazy Learning. Artificial Intelligence Review, vol. 11, number 1-5, pages 1-6, 1997.

[5] G. Bontempi, M. Birattari and H. Bersini, ed. I. Bratko and S. Dzeroski, Local learning for iterated time series prediction. Machine Learning :Proceedings of the Sixteenth International Conference, pages 32-38, Morgan Kaufmann Publishers, San Francisco, CA, 1999

[6] B. Efron and R. J. Tibshirani, An introduction to the bootstrap, Chapman & Hall, 1993.

[7] C. M. Bishop, Neural Networks for Pattern Recognition. New York: Oxford, 1995.

[8] A. Lendasse, E. Oja, O. Simula, M. Verleysen, Time Series Prediction Competition: The CATS Benchmark, IJCNN 2004, International Joint Conference on Neural Networks, Budapest (Hungary), 25-29 July 2004, vol. II, pp. 1615-1620.