Clustered Partial Linear Regression

Luís Torgo

LIACC - FEP
email : ltorgo@ncc.up.pt
WWW : http://www.ncc.up.pt/~ltorgo

Joaquim Pinto da Costa

LIACC – DMA/FC
email : jpcosta@ncc.up.pt

R. Campo Alegre, 823, 2º - 4150 PORTO
PORTUGAL
Phone : (+351) 22 607 8830 Fax : (+351) 22 600 3654

Abstract. This paper presents a new method that deals with a supervised learning task usually known as multivariate regression. The main distinguishing feature of this new technique is the use of a clustering method to obtain sub-sets of the training data before the learning phase. After this "resampling" process a different regression model is fitted to each found cluster. We call the resulting method *clustered partial linear regression*. Predictions using this technique are preceded by a cluster membership query for each test case. The cluster membership probability of a test case is used as a weight in an averaging process that calculates the final prediction. This averaging process involves the predictions of the regression models associated to the clusters for which the test case may belong. We have tested this general multistrategy approach using several regression techniques and we have observed significant accuracy gains in several data sets. We have also compared our method to bagging that also uses an averaging process to obtain predictions. This experiment showed that the two methods are significantly different. Finally, we present a comparison of our method with several state-of-the-art regression methods.

Keywords : Regression, Clustering, Multistrategy learning, Multiple models.

1 Introduction

This paper describes *clustered partial linear* models. This is a new method for addressing multivariate regression problems. Multivariate regression is a supervised learning task that can be loosely defined as the search for a model of the relationship between a target continuous variable and a set of other input variables (attributes, features). The technique we describe deals with this problem using a multistrategy approach. In an initial stage a set of samples of the available training data is obtained using a clustering method. This step is motivated by the assumption that on reasonably complex domains it should be easier to model sub-sets of "similar" training cases than

to try to fit a single model to all data. After this initial resampling phase we fit a regression model to each of the found clusters. Although this general two-stage schema can be applied to any regression method (and even to other supervised learning tasks), in this paper we concentrate our description on *partial linear* regression models (*e.g.* [12, 17]). Still, we also report some results using other regression techniques. Partial linear models belong to the class of semiparametric approaches that integrate parametric with non-parametric techniques. In the case of partial linear models, a standard least squares linear polynomial (*e.g.* [8]) is integrated with a kernel smoother [14, 18]. The main motivation behind these models is to retain as much as possible the comprehensibility of linear polynomials, while trying to improve their accuracy by adding a smoothing component that compensates, on a query-base, for the local inadequacies of the linearity assumption of first order polynomials.

The use of *clustered partial linear* models for predicting the target value of a test case also involves two stages. In a first step we collect a set of cluster membership values that represent the probability of the test case belonging to each cluster. Using these probabilities and the predictions of the regression models in each cluster we calculate the final prediction of our model through a weighed average of these individual predictions.

This paper is organized as follows. The next section describes partial linear regression that is the basic technique that we use within our clustered approach to regression. Section 3 presents clustered partial linear models. In Section 4 we describe a series of experiments with these models. A further analysis of clustered partial linear regression is given in Section 5. Finally, we present the main conclusions of this work.

2 Partial Linear Models

Partial linear regression [12, 17] is a semiparametric technique that integrates a linear polynomial with a kernel smoothing component. A prediction for a query case using these models is obtained by summing the value predicted by the linear model with the value resulting from smoothing over the residuals (errors) of the linear model in the neighbouring training points. The more inadequate the linear model is to the given training sample the larger the importance of the smoothing component. In the extreme case where the linear component perfectly fits the training data, a partial linear model reduces to a standard least squares linear polynomial.

Given a data set, $\{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n}$, where $\mathbf{x}_i$ is a vector of attribute values, a linear regression model of the form $Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_a X_a$, can be obtained using a least squares error criterion. This consists of finding the vector of parameters $\boldsymbol{\beta}$ that minimizes the sum of the squared error, *i.e.* $(\mathbf{Y} - \mathbf{X}\beta)'(\mathbf{Y} - \mathbf{X}\beta)$, where $\mathbf{X}'$ denotes the transpose of matrix $\mathbf{X}$. After some matrix algebra the minimization of this expression with respect to $\boldsymbol{\beta}$ leads to the following set of equations, usually referred to as the *normal equations* (*e.g.* [8]),

$$(\mathbf{X}'\mathbf{X})\beta = \mathbf{X}'\mathbf{Y}\,. \tag{1}$$

The parameter values can be obtained solving the equation,

$$\beta = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y} \tag{2}$$

where $\mathbf{X}^{-1}$ denotes the inverse of matrix $\mathbf{X}$.

As the inverse matrix does not always exists this process suffers from numerical instability. A better alternative [15] is to use a set of techniques known as *Singular Value Decomposition* (*SVD*), that can be used to find solutions of systems of equations with the form $\mathbf{X}\beta = \mathbf{Y}$.

A kernel smoother [14, 18] can be seen as a form of lazy learner [1] that delays learning till prediction time. Given a query point $\mathbf{x}_q$, a prediction is obtained using the following expression,

$$ks(\mathbf{x}_q) = \frac{1}{SKs}\sum_{i=1}^{n} K\left(\frac{d(\mathbf{x}_i, \mathbf{x}_q)}{h}\right) \times y_i \tag{3}$$

where,

$d(.)$ is the distance function between two instances;
$K(.)$ is a kernel function;
h is a bandwidth value;
$<\mathbf{x}_i, y_i>$ is a training instance;
and SKs is the sum of the weights of all training cases, *i.e.*

$$SKs = \sum_{i=1}^{n} K\left(\frac{d(\mathbf{x}_i, \mathbf{x}_q)}{h}\right).$$

This formula is a weighed average over the target values of the training cases that are nearer to the query point. The notion of neighborhood implies the definition of a metric over the multi-dimensional space defined by the input variables and of a distance function between any two cases in this space. The bandwidth size, h, defines the size of the neighborhood that "enters" the weighed average. The kernel function, $K(.)$, provides a smoothing effect, giving more "importance" to nearer training cases. Many different variants of all these "parameters" of kernel smoothing are described in the literature (*e.g.* [2]).

Partial linear models integrate a linear polynomial with a kernel smoother applied on the residuals (errors) of the polynomial. The role of the kernel smoother is to provide an estimate of the error of the linear polynomial for the particular query case under consideration. This estimated error is then added to the linear polynomial prediction giving the predicted value of the partial linear model. Formally, this can be described by,

$$pl(\mathbf{x}_q) = \beta\mathbf{x}_q - \frac{1}{SKs}\sum_{i=1}^{n} K\left(\frac{d(\mathbf{x}_i, \mathbf{x}_q)}{h}\right) \times e_i \quad (4)$$

where,

e_i is the error of the linear model in case $\langle \mathbf{x}_i, y_i \rangle$, given by $e_i = \beta\mathbf{x}_i - y_i$.

Thus, to obtain a prediction for a query case using a partial linear model, we start by obtaining the predicted value of the linear polynomial, $\beta\mathbf{x}_q$. We then calculate the error of this linear polynomial in the training cases that are nearer to the query point, $\mathbf{x}_q$. Using these errors we obtain a kernel prediction of the error for the query case. Finally, this predicted error is added to the initial value predicted by the linear polynomial giving the prediction of the partial linear model.

Compared to linear polynomials, partial linear models have significant advantages in terms of predictive accuracy when the domain is non-linear. Contrary to kernel smoothers, partial linear models have some degree of comprehensibility due to the use of a linear polynomial. However, as they also incorporate a kernel component (that is not comprehensible), they are less understandable than linear regression. In effect, the polynomial component of partial linear models can only be regarded as a rough description of the true surface approximated by these models. The accuracy of this description is proportional to the linearity of the domain under study. In highly non-linear domains the predominance of the kernel "corrections" is so high that the linear polynomial is a very poor "explanation" of the predictions of the partial linear model.

3 Clustered Partial Linear Models

The regression method we propose integrates a clustering technique with a partial linear model. The key idea is to obtain partial linear models for clusters of data instead of fitting a single model to all data.

The first step of our methodology consists of obtaining a clustering of the data. For this purpose we have used system AUTOCLASS C[1] [5, 6]. The main motivation for this choice was the fact that AUTOCLASS C provides the features that we need to implement our method, namely, cluster membership probabilities and automatic choice of the number of clusters. As the goal of this clustering stage is to find groups of "similar" training cases, we have not used the information about the target variable values in this task. This means that AUTOCLASS C only receives information on the input variables. This system outputs the number of found clusters and attaches to each training case the probability of belonging to each of the clusters. Using this information we create a set of training samples, one for each cluster. If a training case has some probability of belonging to more than one cluster it is included in the respective training samples. This means that there may exist some degree of overlap between these training samples.

[1] The program is freely available at http://ic-www.arc.nasa.gov/ic/projects/bayes-group/autoclass/.

After this clustering-based resampling of the training data, we fit a partial linear model to each of the samples. This consists of obtaining a least squares linear polynomial for each cluster, because the kernel component does not involve any "training". Thus, clustered partial linear models consist of a set of partial linear models built in different clusters of the data. If the clustering algorithm is able to produce a symbolic representation of each cluster it is possible to consider clustered partial linear models as a single model. In effect, the symbolic representation of each cluster can be seen as a condition for applying the respective partial linear model, which means that we can look at a clustered partial linear model as a set of rules of the form "IF <cluster representation> THEN <partial linear model>". Even if the clustering algorithm does not produce such symbolic representation of the clusters, we could use any classification algorithm to produce such description, as it is done for instance in ABACUS [9]. This later system can be related to our approach although it follows a different methodology. ABACUS starts by using an equation discovery method to derive equations that can summarise the given data. If more than one equation is necessary the original data is divided accordingly. After this step a classification system is used to discriminate among the obtained divisions of the data. The found descriptions can be regarded as pre-conditions to the application of the previously found equations. Our system, on the contrary, starts by finding relevant clusters of the data irrespectively of the models that are going to be used, and only in a second stage models are obtained for each cluster. In effect, a similar approach to ours is slightly outlined as a future research direction in the above mentioned paper by Falkenhainer and Michalski [9]. Still, other distinguishing features of our work include the use of strongly overlapping clusters and also the use of a probabilistic cluster membership function when making predictions about query cases, which leads to the combination of the values predicted by different models.

Regarding predictions using clustered partial linear models they are obtained as follows. Given a query case, $\mathbf{x}_q$ we obtain the probabilities of belonging to each of the clusters. For each cluster with membership probability higher than zero, the respective partial linear model is used to obtain a prediction for the query case. These predictions are then averaged to obtain the final predicted value using the following formula,

$$CPL(\mathbf{x}_q) = \sum_{k=1}^{j} \left(P_k(\mathbf{x}_q) \times pl_k(\mathbf{x}_q) \right) \tag{5}$$

where,

j is the number of clusters;

$P_k(\mathbf{x}_q)$ is the probability of the query case belonging to cluster k;

$pl_k(\mathbf{x}_q)$ is the prediction of the partial linear model of cluster k (*c.f.* Eq. 4).

4 Experimental Analysis

4.1 An Illustrative Example

This section describes an example of running an implementation of our clustered partial linear models on a particular domain in order to provide a better intuition of its behaviour. We have used the *Abalone* data set, which concerns the task of trying to predict the age of abalone (actually the number of rings that determines this age), based on a series of biometrics of these animals (see Table 1 for full data set details). Running AUTOCLASS C on the 4177 cases we obtain 7 clusters with 1299, 1205, 875, 261, 224, 157 and 156 cases, respectively. This division is obtained by assigning each case to its most probable cluster. As AUTOCLASS C may also give lower probabilities that a case belongs to other cluster, and in these situations we have decided to allocate the same instance to all the alternative clusters (*c.f.* Section 3), the final 7 training sets have 2279, 2838, 2691, 1871, 537, 182 and 188 cases, which gives an idea of the degree of overlap between the training sets. The next step is to develop a partial linear model for each of these training sets. As a partial linear model includes a non-symbolic component (the kernel) we can only provide some examples of part of the models (*i.e.* the linear regression component of the partial linear model). In spite of this lack of descriptive precision we can try to compare some of the models obtained for the built training sets (the found clusters), and the model it would be obtained if the clustering phase was not carried out (*i.e.* using all training data). For instance, for the first training set (2279 cases) the model is :

N.Rings = 7.94 - 5.82*Length + 20.8*Height + 10.3*Whole - 18.4*Shucked - 10.9*Viscera + 10.3*Shell

For the second training set (2838 cases) the obtained model is :

N.Rings = 2.74 - 4.55*Length + 13.4*Diameter + 26.7*Height + 14.9*Whole - 24.6*Shucked - 17.3*Viscera

For the 6th training set (182 cases) we get :

N.Rings = 4.26 - 17.2*Length + 33.4*Diameter

Finally, using all training data (4177 cases) we would get the following single model:

N.Rings = 2.97 + 11.8*Diameter + 9.8*Height + 9.49*Whole - 20.3*Shucked - 10.3*Viscera + 8.73*Shell

The diversity of the models provides an indication of the differences between the found clusters and also the full training data. This can also be confirmed looking at some of the statistics output by AUTOCLASS C, like the cross entropy between each

cluster and the full training data, that gives a measure of the difference between two probability distributions (higher values indicate further divergence from the probability distribution of all training data) :

Cluster	Divergence Value
0	4.17e+01
1	4.36e+01
2	4.48e+01
3	4.99e+01
4	5.15e+01
5	5.49e+01
6	5.62e+01

As we will show in the next section this clustering based resampling of the data is advantageous in terms of predictive accuracy.

Table 1. The used data sets.

Data Set	Main characteristics
Housing	506 case; 13 continuous variables. Predicting housing values in Boston.
Abalone	4177 cases; 7 cont. vars.; 1 nominal var. Predicting the age of abalone.
Elevators	8752 cases; 40 cont. vars. Aircraft control problem (prediction of elevators level).
F	3000 cases; 5 cont. vars. Artificial domain with marked clusters of data points.
Kinematics	8192 cases; 8 cont. vars. Robot arm control problem.
Computer	8192 cases; 22 cont. vars. Prediction of CPU activity level in a computer network.
Computer (small)	8192 cases; 12 cont. vars. Simplified version of the previous data set.
Telecomm	15000 cases; 49 cont. vars. Telecommunications problem.

4.2 Comparative Experiments

In this section we describe a series of experiments with clustered partial linear models. The experiments we report are carried out using the data sets shown in Table 1. With respect to the experimental methodology all reported results are averages of five repetitions of a 10-fold Cross Validation experiment. Significance of the differences in mean squared error (MSE) are asserted through paired *t*-tests.

The first experiment we report was designed to answer the following question:

- *Is there a significant difference in accuracy between applying the models in clustered training samples or to all available training data?*

We have tested our clustering-based method with three different types of regression models: partial linear models; linear regression models; and regression trees. For each of these trials we have compared the "clustered" variant, obtained using the methodology described in Section 3, with the alternative of simply applying the regression model to all available training data. The results are shown in Table 2. Significant wins (99% confidence) of the "clustered" versions are marked with "+" signs, while the opposite appears with "-" signs.

Table 2. The advantages of clustering the data.

	Partial Linear Models			*Linear Regression*			*Regression Trees*		
Data Set	Clustered	All data		Clustered	All data		Clustered	All data	
Housing	13.04	16.94	+	14.89	22.83	+	25.42	20.13	
Abalone	2.28	4.75	+	2.22	5.00	+	5.55	5.41	
Elevators	5.61	15.07	+	9.95	37.23	+	9.73	17.38	+
F	1.56	3.45	+	5.99	28.96	+	4.31	7.56	+
Kinematics	0.010	0.012	+	0.030	0.041	+	0.032	0.039	+
Computer	24.38	22.07	-	152.89	220.89	+	8.21	12.35	+
Computer (small)	13.98	13.93	-	80.30	241.16	+	7.65	14.36	+
Telecomm	89.86	53.72	-	631.51	823.97	+	76.71	61.90	-

These experiments confirm the advantages of pre-clustering the data as we propose. In effect, with all three regression methods (that are quite different) there is a general trend towards a significant gain in predictive accuracy. However, clustering the given training data is just one of the distinguishing factors of our methodology. Another difference to the non-clustered methods, is the averaging of the predictions of different models. Regarding this issue our methodology resembles bagging [3], were predictions are obtained by averaging over a set of models obtained using different bootstrap samples of the given data. According to Breiman [3] bagging is expected to give good results when the base prediction method is sensible to small perturbations on the learning set, as it is the case of regression trees. However, that is not the case of partial linear models that are quite robust to these variations, as the results of Table 3 confirm. This table shows the results of a comparison between partial linear models and regression trees with their respective "bagged" versions. The results confirm Breiman's statement. The "bagged" models were obtained using 50 bootstrap replicates of the data. Significant wins of the bagged versions are marked with '+' signs.

Table 3. Bagging regression models.

	Partial Linear Models			*Regression Trees*		
Data Set	Single	Bagged		Single	Bagged	
Housing	16.94	17.12		20.13	13.02	+
Abalone	4.75	4.68	+	5.41	4.63	+
Elevators	15.07	15.11		17.38	10.18	+
F	3.45	3.41		7.56	4.04	+
Kinematics	0.012	0.012		0.039	0.024	+
Computer	22.07	22.05		12.35	8.20	+
Computer (small)	13.93	14.14	-	14.36	9.75	+
Telecomm	53.72	54.56	-	61.90	37.08	+

The results of these experiments with bagging show that the accuracy advantages of clustered partial linear models that were observed in Table 2, can only be caused by the effects of clustering the training data. In effect, the results of Table 3, indicate that there is nothing to gain with averaging over several partial linear models.

Having shown the advantages of a pre-clustering of the training sample, it remains an open question whether the results of clustered partial linear models are good when compared to other existing approaches to regression. The following experiment has the goal of answering this question. We have compared clustered partial linear models with several state-of-the-art regression systems, namely, CUBIST [http://www.rule-quest.com], MARS [11] and a bagged version of CART [4]. The results of this experiment are shown in Table 4. Significant wins of clustered partial linear models are marked with '+' signs.

Table 4. Clustered partial linear regression (CPL) versus other approaches.

Data Set	CPL	CUBIST		MARS		BaggedCART	
Housing	13.04	14.24		18.32	+	13.02	
Abalone	2.28	4.67	+	4.54	+	4.63	+
Elevators	5.61	12.91	+	6.04		10.18	+
F	1.56	13.64	+	19.95	+	4.04	+
Kinematics	0.010	0.027	+	0.036	+	0.024	+
Computer	24.38	6.49	-	10.22	-	8.20	-
Computer (small)	13.98	9.71	-	14.00		9.75	-
Telecomm	89.86	91.51		??[2]		37.08	-

Clustered partial linear models achieved quite competitive accuracy in most domains. Some results are particularly outstanding, namely in the *Abalone*, *F* and *Kinematics*

[2] The version we have of MARS (3.6) gives a "segmentation fault" on this data set.

domains. Moreover, in the case of *Abalone* and *F*, the excellent scores are clearly caused by the clustering step because neither the single models nor the "bagged" versions obtain similar results (*c.f.* Tables 2 and 3). However, the results obtained on the two *Computer* domains and in the *Telecomm* application are a bit disappointing. A possible cause of these results is the complete inadequacy of linear polynomials to these domains, which can be confirmed by the results of Table 2. Although partial linear models include a smoothing component that could overcome this mismatch, there are situations were this is not possible. In effect, the lack of symmetry near the boundaries of the input space causes well known difficulties to kernel smoothers [13]. The extrapolation capabilities of linear polynomials may also lead to "wild" predictions near these boundaries. These two factors together may explain the poor performance on these domains. This explanation is consistent with the fact that clustered regression trees (that do not have such difficulties) do not achieve such disappointing results (*c.f.* Table 2). Thus, we claim that these poor results are caused by the lack of adequacy of the base regression models to the domains and not by any difficulty of our proposed methodology.

5 Discussion

The main motivation behind our proposal is the hypothesis that modeling a small set of nearby cases is easier than modeling a larger sample of instances. Although we do not provide any theoretical proof of this hypothesis we have collected a series of experimental results that we claim to provide good indications to support this. Moreover, there is a strong relation between this hypothesis and the way lazy learners proceed. In effect, this type of techniques can be seen as obtaining a local model around the neighborhood of each query case. These methods are able to easily capture the local regularities of the regression surface and thus achieving excellent accuracy. Our methodology differs from these approaches in that the local neighborhoods are pre-defined at the start through a clustering method instead of being a function of each query case. This has large computational advantages over "pure" lazy learners. Moreover, as there is a fixed (and usually small) set of pre-defined clusters we can obtain a comprehensible model of the data, which is not the case of lazy learners.

In spite of the promising accuracy results our methodology also has some disadvantages. The more noticeable is the increment of computation time when compared to applying a regression model to all training data. This additional cost is caused by the clustering step. AUTOCLASS C has many parameters that could be explored in order to try to improve this speed issue. Still, clustering is always a heavy task and its weight in the overall computation time of clustered partial linear models will always be high.

As we have already mentioned there are some relations of our method with multiple model approaches like bagging [3]. These relations have to do with the construction of different models based on possibly overlapping samples of cases and in averaging the predictions of these models as a form to obtain predictions for test cases. However, the method used to obtain the individual samples is totally different. In bagging a bootstrap random sampling process is used to obtain samples with the

same size as the original training sample. In our methodology the samples are not obtained randomly and are usually smaller than the original training sample. The type of resampling carried out by our clustering step changes the distribution of the cases in the original sample, which is not the case with bagging. From this perspective, our method is related to boosting [16, 10], where the same distribution change is done through a system of weights. However, contrary to boosting our method is not sequential and thus it is possible to construct the individual models in parallel.

Devogelaere *et al.* [7] describe a related approach to regression. Their GAdC system performs a genetic algorithm driven clustering of the training data. The evaluation function that drives the genetic algorithm-based search for the clusters includes several factors like cluster distance penalty, prediction error, etc. This means that, unlike our method, GAdC uses information of the regression accuracy to guide the search for clusters. Within each cluster, GAdC uses a kind of kernel model to obtain predictions. Another difference of our work is the probabilistic approach to cluster membership that leads to overlap of clusters and also to the averaging of different clustered models.

The method described in this paper can also be related to typical partition-based learners, like tree-based models or rule-based systems. These methods also partition the given sample in a set of local regions and fit some kind of model within each of these regions. However, there are some fundamental differences to our method. The more important is the criteria used to form the partitions. Regression trees (*e.g.* [4]), for instance, search for regions of low variance in the target variable. Our method does not use any information regarding the target variable to obtain the partitions. On the contrary the partitions are obtained based on information concerning the input variables. Other differences to these approaches are the possible overlap of the regions, the probabilistic approach to decide on which cluster to "place" a test case, and the averaging over different regions in prediction tasks.

In the near future we intend to extend our experimental evaluation of the method, with the goal of clearly understanding what are the key factors affecting its results. We also intend to further experiment with tuning of the clustering step. Finally, based on the poor results of clustered partial linear models in some domains, that we claim to be caused by the regression model, we intend to explore the possibility of using different regression models for each cluster.

6 Conclusions

We have described an approach to multivariate regression whose main distinguishing feature is the use of a clustering algorithm to obtain samples of the available training data. These samples are then modeled individually through partial linear models. Predictions using the resulting clustered partial linear models are obtained by averaging over the models in the clusters for which the membership probability of the test cases is higher than zero.

We have tried this clustering-based approach to regression with three different types of regression models. With all three methods we have observed a similar pattern of results, showing the advantages of pre-clustering the training data.

We have compared our method to bagging that also uses averaging over multiple models. The results show that there are significant differences between the two methods, with some clear advantages of our approach.

Compared to existing state-of-the-art regression approaches our method achieved quite competitive results in the tested domains. The accuracy in some domains can be considered quite outstanding, deserving further analysis.

References

1. Aha,D. : *Lazy Learning*. Kluwer Academic Publishers, 1997.
2. Atkeson,C., Moore,A., Schaal,S. : Locally Weighted Regression. *Artificial Intelligence Review*, **11**, 11-73, 1997.
3. Breiman, L.: Bagging predictors. In *Machine Learning*, **24** (2). Kluwer Academic Publishers, 1996.
4. Breiman,L. , Friedman,J.H., Olshen,R.A. and Stone,C.J.: *Classification and Regression Trees*, Wadsworth Int. Group, Belmont, California, USA, 1984.
5. Cheeseman,P., Kelly,J., Self,M., Stutz,J. : Autoclass: a bayesian classification system. In Proceedings of the 5th International Conference on Machine Learning. Morgan Kaufmann, 1988.
6. Cheesman,P., Stutz,J. : Bayesian classification (AUTOCLASS): theory and results. In Advances in Knowledge Discovery. AAAI Press, 1995.
7. Devogelaere,D., Van Bael,P., Rijckaert,M. : Regression through genetic algorithm driven clustering. In Proceeding of the 7th European Congress on Intelligent Techniques & Soft Computing (EUFIT'99), 1999.
8. Drapper,N., Smith,H. : *Applied Regression Analysis* (2nd edition). John Wiley & Sons, 1981.
9. Falkenhainer,B., Michalski,R. : Integrating quantitative and qualitative discovery in the ABACUS system. In *Machine Learning, an artificial intelligence approach*, vol. III. Kodratoff,Y. and Michlaski,R. (eds.). Morgan Kaufmann, 1990.
10. Freund,Y., Shapire,R. : Experiments with a new boosting algorithm. In Proceedings of the 13th International Conference on Machine Learning. Morgan Kaufmann, 1996.
11. Friedman,J. (1991) : Multivariate adaptive regression splines (with discussion). *Annals of Statistics*, **19** : 1-141.
12. Hardle, W. : *Applied Nonparametric Regression*. Econometric Society Monographs. Cambridge University Press, 1990.
13. Hastie,T., Loader,C. : Local regression : automatic kernel carpentry. *Statistical Science*, **8**, 120-143, 1993.
14. Nadaraya,E.A. : On estimating regression. *Theory of Probability and its Applications*, **9**, 141-152, 1964.
15. Press,W., Teukolsky,S., Vetterling,W., Flannery,B. : *Numerical receipes in C*. Cambridge University Press, 1992.
16. Shapire,R. : The strength of weak learnability. *Machine Learning*, **5**, 197-227. Kluwer Academic Publishers, 1990.
17. Spiegelman,C.H. : Two techniques for estimating treatment effects in the presence of hidden variables: adaptive regression and a solution to Reiersol's problem. Unpublished Ph.D. Thesis, Northwestern University, Dept. of Mathematics, 1976.
18. Watson, G.S. : Smooth regression analysis. *Sankhya: The Indian Journal of Statsitics*, Series A, **26**, 359-372, 1964.