Regression by Classification

Luís Torgo

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

João Gama

email : jgama@ncc.up.pt
WWW: http://www.up.pt/~jgama

LIACC - University of Porto
R. Campo Alegre, 823 - 4150 Porto - Portugal
Phone : (+351) 2 6001672 Fax : (+351) 2 6003654
WWW : http://www.up.pt/liacc/ML

Abstract
We present a methodology that enables the use of existent classification inductive learning systems on problems of regression. We achieve this goal by transforming regression problems into classification problems. This is done by transforming the range of continuous goal variable values into a set of intervals that will be used as discrete classes. We provide several methods for discretizing the goal variable values. These methods are based on the idea of performing an iterative search for the set of final discrete classes. The search algorithm is guided by a N-fold cross validation estimation of the prediction error resulting from using a set of discrete classes. We have done extensive empirical evaluation of our discretization methodologies using C4.5 and CN2 on four real world domains. The results of these experiments show the quality of our discretization methods compared to other existing methods.
Our method is independent of the used classification inductive system. The method is easily applicable to other inductive algorithms. This generality turns our method into a powerful tool that extends the applicability of a wide range of existing classification systems.

Keywords : learning, regression, classification, discretization methods.

1. Introduction

Machine learning (ML) researchers have traditionally concentrated their efforts on classification problems. However, many interesting real world domains demand for regression tools. In this paper we present and evaluate a discretization methodology that extends the applicability of existing classification systems to regression domains. With this reformulation of regression we broaden the range of ML systems that can deal with these domains.

The idea of mapping regression into classification was originally used by Weiss & Indurkhya [19, 20] with their rule-based regression system. They used the P-class

algorithm[1] for class discretization as a part of their learning system. This work clearly showed that it is possible to obtain excellent predictive results by transforming regression problems into classification ones and then use a classification learning system. Our works is based on these results. We have oriented our research into the discretization phase as opposed to Weiss & Indurkhya's work. We do not supply a complete regression learning system like those authors did. We concentrated our research on two major goals related to the problem of class discretization. Firstly, to provide alternative discretization methods. Secondly, to enable the use of these methodologies with other classification systems. As to the first goal we were able to prove through extensive empirical evaluation on four real world domains that two of our proposed discretization methodologies outperformed the method used on the cited work. These experiments also revealed that the best methodology is dependent on both the regression domain as well as on the used classification system, thus providing strong evidence for our search-based discretization method. With respect to the second goal we have used our methodologies with CN2 [2] and C4.5 [15]. Our discretization system is easily interfaced to any other classification algorithm[2] .

The next section gives a brief overview of the steps involved in solving regression problems by means of classification inductive algorithms. We then present our discretization methodology on section 3. The experiments we have done are described on section 4. Finally we describe some future work and present the conclusions of this paper.

2. Mapping Regression Into Classification

In regression problems we are given samples of a set of independent (predictor) variables x_1, x_2, …, x_n , and the value of the respective dependent (output) variable y[3] . Our goal is to obtain a model that somehow captures the mapping $y = f(x_1, x_2, \ldots, x_n)$ based on the given samples. Classification differs from this setup in that the class is categorical instead of numerical.

Mapping regression into classification is a kind of pre-processing technique that enables us to use classification algorithms on regression problems. The use of these algorithms involves two main steps. First there is the creation of a data set with discrete classes. This step involves looking at the original continuous class values and dividing them into a series of intervals. Each of these intervals will be a *discrete* class. Every example whose output variable value lies within an interval will be assigned the respective discrete class. The second step consists on reversing the

---

1 This algorithm is historically known as the K-means method in statistics and pattern recognition.

2 We already have an interface to a linear discriminant although we still do not have experimental results.

3 For reasons of simplicity we shall use the term class instead of dependent variable from now on.

discretization process after the learning phase takes place. This will enable us to make numeric predictions from our learned *regression model*. Figure 1 shows a diagram of this process:

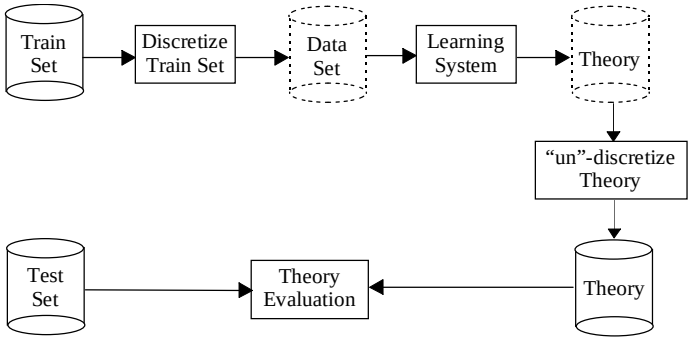

Figure 1 - Using classification algorithms on regression problems.

2.1 Methods For Splitting A Set Of Continuous Values

The key issue on a discretization process is the transformation of a set of values into a set of intervals. These intervals may then be used as discrete classes. In this section we present three methods for performing this task. All of them receive as input a set of values and the desired number of intervals.

- *Equally probable intervals (EP)*
  This strategy creates a set of N intervals with the same number of elements.
- *Equal width intervals (EW)*
  The original range of values is divided into N intervals with the same range.
- *K-means clustering (KM)*
  In this method we try to build N intervals that minimize the sum of the distances of each element of an interval to the interval's *gravity center* [4] [3]. This is basically the P-class method that is given in [20]. This method starts with the EP approximation but then tries to move the elements of each interval to contiguous intervals whenever these changes reduce the referred sum of distances.

To better illustrate these strategies we show how they group the set of values {1,3,6,7,8,9.5,10,11} assuming that we want to partition them into three intervals (N=3):

- EP gives the intervals [1 .. 6.5],]6.5 .. 9.75] and]9.75 .. 11] with each interval containing respectively the values{1,3,6}, {7,8,9.5} and {10,11}.

---

[4] We always use the median as a centrality statistic. We prefer it to the mean to avoid outliers influence.

- Using EW we get [1 .. 4.33],]4.33 .. 7.66] and]7.66 .. 11] containing the values {1,3}, {6,7} and {8,9.5,10,11}.
- Finally KM obtains the intervals [1 ..4.5],]4.5 .. 8.75] and]8.75 .. 11] grouping the values in {1,3}, {6,7,8} and {9.5,10,11}.

The *problem* of these strategies is that they assume that we know the number of intervals that is appropriate for our problem. Our experiments show that this number is dependent not only on the domain we are dealing with, but also on the classification system that will be used after the discretization process. The methodology we present in this paper overcomes this difficulty by means of an iterative search approach.

2.2 Making predictions from the learned models

After the learning phase we obtain a theory that classifies examples into one of the chosen intervals. The next step consists on using this learned theory to predict the class value of unseen instances. Given a discrete class (an interval) we want to obtain a value that will be used as our prediction. The standard procedure is to use a measure of centrality of the interval as prediction. In our experiments we use the median of the values that originated the interval.

Evaluating the accuracy of regression models

We now address the problem of evaluating the predictive power of regression models. The standard procedure used to evaluate the accuracy of a *theory* consists on testing it on unseen data. On regression the prediction error e is given by the difference between the real value y and the predicted one $\hat{y}$. This methodology is very different from the one followed in classification problems. On these tasks errors are non-metric, i.e. a prediction is either correct or incorrect. Accuracy is thus a function of the number of errors. In regression the amplitude of errors is important.

There are several statistics that somehow try to characterize the accuracy of regression models. In our experiments we have chosed to work with two of them. One gives absolute estimates of the error (MAE) while the other provides relative estimates (MAPE) :

$$MAE = \frac{\sum_i |y_i - \hat{y}_i|}{N} \qquad MAPE = \frac{\sum_i \left| \frac{y_i - \hat{y}_i}{y_i} \right| \times 100}{N}$$

There are much more possibilities each having some advantages and some disadvantages. It is out of the scope of this paper to determine which one is more adequate to a given task.

3. Iterative Class Discretization

On section 2.1 we have shown several ways of splitting a set of values into a set of intervals. These splitting methods need to know in advance the target number of

intervals (i.e. the number of discrete classes). This number is obviously dependent on the domain in state. In this section we describe an iterative search approach to solve this problem of finding the number of discrete classes to be used.

3.1 The Wrapper Approach

The goal of the class discretization process is to obtain a discrete data set that enables the classification algorithm to learn a theory that has the best possible regression accuracy. As we change the number of used classes we are changing the input to this classification system and thus varying its regression accuracy. Because of this we can easily see that the discretization process should take into account the classification system that will be used afterwards. In other terms, the used discrete classes are just a kind of parameter of the classification algorithm. The wrapper approach [8, 9] is a well known strategy which has been mainly used for feature subset selection ([8] and [10] among others) and parameter estimation [13]. Pazzani [14] also used a similar approach on feature creation which is a similar problem to ours. The use of this approach to estimate a parameter of a learning algorithm can be described by the following figure:

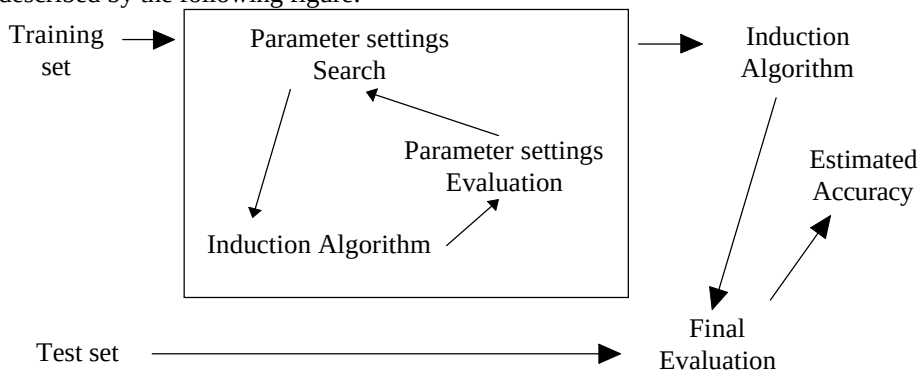

Fig. 2. The wrapper approach.

The two main components of the wrapper approach are the way how new parameter settings are generated and how they are evaluated in the context of the target learning algorithm. The basic idea is to try different parameter settings and choose the one that gives best estimated results. This best setting is the result of the wrapper process an will then be used in the learning algorithm for the real evaluation using an independent test set.

Translating this scenario to our discretization problem we basically have to find the discretization method that gives the best results. Our method tries several possible discretization settings (i.e. set of discrete classes) and chooses the one that gives the best estimated accuracy. To evaluate the candidate setups we use the well known N-fold cross validation (CV) test.

The search component of our wrapper approach consists of the process used to generate a new candidate set of classes (i.e. the search operators) and the search algorithm. We use a simple hill-climbing search algorithm coupled with a kind of lookahead mechanism to try to avoid the well-known problem of local minimum of

this algorithm. The search proceeds by trying new candidate sets of classes until a certain number (the lookahead value) of consecutive worse trials occur.

We provide two alternative ways of generating a new candidate discretization setting. Both of them can be applied to the three presented splitting strategies (section 2.1). This gives six different discretization methods that can be used to create a set of discrete classes using this wrapper approach.

3.2 Generating candidate sets of discrete classes

In this section we address the problem of modifying the current discretization setting based on its estimated predictive accuracy result. The search space consists of all possible partitions of a set of continuous values. Our system has two alternative ways of exploring the search space. Both are applicable to the 3 splitting methods that were mentioned on section 2.1:

- *Varying the number of intervals (VNI)*
  This alternative consists on trying several values of the number of intervals with the current splitting strategy. We start with a number of intervals and on each iteration of the search process we increment this number by a constant value. This is the more obvious way of improving the splitting methods presented in section 2.1.

- *Selective specialization of individual classes (SIC)*
  The second alternative is a bit more complex. The basic idea is to try to improve the previously tried set of intervals (classes). We start with any given number of intervals and during the CV-evaluation we also calculate the error estimates of each individual discrete class. The next trial is built by looking at this individual error estimates. The median of these errors is calculated. All classes whose error is above the median are specialized. The specialization consists on splitting each of these classes in two other classes. We do this by applying the current splitting method to the values within that class interval. All the other classes remain the same in the next iteration.

The next section provides an illustrative example of these two search alternatives in a discretization task.

4. The Experiments

We have conducted several experiments with four real world domains.

The main goal of the experiments was to assert each discretization methodology performance when the input conditions vary. These conditions are the regression domain, the classification learning system and the regression error measure used to evaluated the learned models. Some of the characteristics of the used data sets are summarized on Table 1. These data sets were obtained from the UCI machine learning data base repository :

Data Set	N. Examples	N. Attributes
housing	506	13
servo	167	4
auto-mpg	398	7
machine	209	6

Table 1- The used data sets.

As we already referred we have used C4.5 and CN2 in our experiments. We have used as regression accuracy measures the MAE and MAPE statistics.

We *linked* our methodologies to each of these learning algorithms obtaining two new *regression learners*. We evaluate the performance of these systems with the two statistics on the chosen domains. We have obtained this evaluation by means of a 5-fold cross validation test. On each iteration of this process we have forced our discretization system to use each of the six alternative discretization methods producing six different discrete data sets that were then given to C4.5 and CN2. We have collected a set of true prediction errors for each of the regression models learned with the six discrete data sets. The goal of these experiments was to compare the results obtained by each discretization method under different setups on the same data. The 5-CV average predictive accuracy on the "auto-mpg" data set is given on Table 2 (the small numbers in italic are the standard deviations):

		VNI			***KNI***		
		EP	***EW***	***KM***	***EP***	***EW***	***KM***
MAE	***C4.5***	2.877 *±0.333*	2.796 *±0.308*	**2.783** ***±0.299***	2.982 *±0.360*	3.127 *±0.282*	3.134 *±0.272*
	CN2	3.405 *±0.266*	4.080 *±0.654*	3.597 *±0.641*	3.311 *±0.188*	3.695 *±0.407*	**3.053** ***±0.340***
MAPE	***C4.5***	12.50 *±1.520*	12.556 *±1.980*	12.200 *±1.623*	12.072 *±2.549*	**11.600** ***±2.074***	11.996 *±1.195*
	CN2	15.189 *±1.049*	15.474 *±1.631*	15.282 *±2.235*	14.485 *±0.936*	14.930 *±1.291*	**13.871** ***±2.069***

Table 2 - Experiments with "auto-mpg".

The best score for each setup is in bold. Due to space reasons we summarize the overall results on Table 3 where we present the winning strategies for all data sets :

Set Up	**Servo**	**Machine**	**Housing**	**Auto-mpg**
C4.5 / MAE	VNI+KM	SIC+KM	VNI+EW	VNI+KM
CN2 / MAE	SIC+EP	SIC+KM	SIC+EW	SIC+KM
C4.5 / MAPE	VNI+KM	SIC+KM	SIC+KM	SIC+EW
CN2 / MAPE	SIC+EP	SIC+EW	VNI+KM	SIC+KM

Table 3 - Summary of overall results.

Table 4 gives the rank score of each strategy. The numbers in the columns of this table represent the number of times a strategy was ranked as the Nth best strategy.

The last column gives the average ranking order for each method. The methods are presented ordered by average rank (lower values are better) :

	1st	*2nd*	*3rd*	*4th*	*5th*	*6th*	*Avg. Rank*
SIC+KM	6	2	3	1	0	4	2,24
VNI+EP	0	7	3	5	1	0	2,29
VNI+KM	4	1	3	6	2	0	2,33
SIC+EW	3	0	3	2	7	1	2,90
SIC+EP	2	3	3	5	2	3	3,10
VNI+EW	1	3	1	0	3	8	3,48

Table 4 - Rank scores of the methods.

The main conclusion to draw from our experiments is the clear dependency of the best methodology on the used set up. There is no clear winning strategy on all domains. This proves the validity of our search-based approach to class discretization.

Table 3 shows that our selective specialization strategy (SIC) is most of the times (11 out of 16) one of the components of the best discretization method. Looking at the average rank results (Table 4) SIC+KM is the best followed by VNI+EP. SIC+KM is also the method that is the best more often. The two methods using EW splitting method have bad averages, nevertheless these methods sometimes are the best so they are not useless. On the contrary VNI+EP which was the second best on average was never the best strategy in all our experiments. Another interesting observation is the regularity observed on the "machine" data set (see Table 3) contrasting with the other data sets.

5. Relations To Other Work And Future Directions

We have presented a general class discretization method and evaluated it in conjunction with two classification algorithms. It is our goal to experiment with more systems with different characteristics.

Within the ML community other work exists on the area of continuous attribute discretization. This work usually performs a kind of pre-processing by trying to maximize the mutual information between the resulting discrete attribute and the classes (for instance [4] and [11]). This is a good strategy but it is applicable only when the classes are given. Ours is a very different problem, as we are determining which classes to consider.

Within the ML field some regression learning systems exist (for instance CART [1], M5 [16] and R^2 [17]) that could be used on these domains. These systems do not transform regression into classification tasks. Weiss & Indurkhya have demonstrated [20] that this transformation can obtain good results when compared the these more "classical" methods. They have done this with their rule-based regression system that learns with discrete classes. They have tested it on several domains (including the ones we have used). The results they report show that their system clearly outperforms CART, a Nearest Neighbor algorithm and the statistical method MARS [5]. These results were a key motivation for our work. They indicate that it is

possible to obtain good accuracy with classification systems on regression problems. Their system is a two step algorithm. First there is the discretization phase where they use a method that is equal to our VNI+KM method. Finally they use the resulting discrete data set with their classification system. As we did not have available their classification system we were not able to test our discretization methods together with this system. However, we have tested their discretization method with CN2 and C4.5. The experiments showed that the best method depends on both the domain as well as on the used classification system (Table 3). This fact does not enable us to definitely say that our methods are always better than the VNI+KM method. Nevertheless, these results reinforce our search-based approach that is able to chose the discretization method to use depending on both these factors. Table 4 also shows that on average both SIC+KM and VNI+EP are better then VNI+KM. This seems to indicate that these methodologies together with Weiss & Indurkhya's classification system could even get better overall regression accuracies when compared to the other "classical" regression methodologies.

One possible future improvement on our work is to try to use other search algorithms (like best-first [6], simulated annealing [18] or even genetic-based search algorithms [7]).

Another interesting research topic as to do with the inability of classification systems to take advantage of the implicit ordering of the obtained *discrete* classes. Because of this, an error has always the same cost. Unfortunately this is not suitable for the evaluation measures used for calculating the accuracy of regression models. A possible way to overcome this drawback is to use a cost matrix in the learning phase. This matrix would distinguish between the errors. This error cost information is important even in the classification scenario for several domains [12]. We could use the distance between the median of each interval as the error cost of *confusing* classes. We have already implemented this idea together with a linear discriminant that is able to use cost matrices. We do not include this work here as we still do not have experimental results.

6. Conclusions

The method described in this paper enables the use of classification systems in regression domains. Previous work [20] provided evidence for the validity of transforming regression into classification. This was oriented towards one learning algorithm. Our work enables the use of a similar transformation strategy with other classification systems. This extends the applicability of a wide range of existent inductive systems.

Our algorithm chooses the best discretization method among a set of available strategies. We estimate the prediction error of each candidate method and select the best among them. The resulting discrete classes are obtained by an iterative search procedure using the chosen discretization method. This iterative search is basically a wrapper process based on a N-fold CV evaluation that estimates the predictive error resulting from using a set of discrete classes. We have also introduced five novel methods for discrete class formation.

We have showed the validity of our search-based approach by means of extensive experiments on four real world domains. These experiments indicated that a search-based approach is necessary if we want to handle several domain/learning system/error measure scenarios. The results of the experiments also showed that some of our methods for class formation were among the best on most of the cases.

We have applied our methodology to two classification inductive systems (C4.5 and CN2). It is easy to use it with other learning algorithms. This generality turns our methodology into a powerful tool for handling regression using existing ML classification inductive systems.

References

[1]. Breiman,L. , Friedman,J.H., Olshen,R.A. & Stone,C.J. (1984): Classification and Regression Trees, Wadsworth Int. Group, Belmont, California, USA, 1984.

[2]. Clark, P. and Niblett, T. (1988) : The CN2 induction algorithm. In *Machine Learning*, **3**, 261-283.

[3]. Dillon,W. and Goldstein,M. (1984) : *Multivariate Analysis*.. John Wiley & Sons, Inc.

[4]. Fayyad, U.M., and Irani, K.B. (1993) : Multi-interval Discretization of Continuous-valued Attributes for Classification Learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence (IJCAI-93)*. Morgan Kaufmann Publishers.

[5]. Friedman, J. (1991) : Multivariate Adaptative Regression Splines. In *Annals of Statistics*, 19:1.

[6]. Ginsberg, M. (1993) : *Essentials of Artificial Intelligence*. Morgan Kaufmann Publishers.

[7]. Holland, J. (1992) : *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control and artificial intelligence*. MIT Press.

[8]. John,G.H., Kohavi,R. and Pfleger, K. (1994) : Irrelevant features and the subset selection problem. In *Machine Learning : proceedings of the 11th International Conference*. Morgan Kaufmann.

[9]. Kohavi, R. (1995) : Wrappers for performance enhancement and oblivious decision graphs. PhD Thesis.

[10]. Langley, P., and Sage, S. (1994) : Induction of selective bayesian classifiers. In *Proceedings of the 10th conference on Uncerrtainty in Artificial Intelligence*. Morgan Kaufmann Publishers.

[11]. Lee, C. and Shin, D. (1994) : A context-sensitive Discretization of Numeric Attributes for classification learning. In *Proceedings of the 11th European Conference on Artificial Intelligence (ECAI-94)*, Cohn, A.G. (ed.). John Wiley & Sons.

[12]. Michie,D., Spiegelhalter;D.J. & Taylor,C.C. (1994): *Machine Learning, Neural and Statistical Classification*, Ellis Horwood Series in Artificial Intelligence, 1994.

[13]. Mladenic, D. (1995) : Automated model selection. In *Mlnet workshop on Knowledge Level Modelling and Machine Learning*. Heraklion, Crete, Greece.

[14]. Pazzani, M.J. (1995) : Searching for dependencies in bayesian classifiers. In *Proceedings of the 5th international workshop on Artificial Intelligence and Statitics*. Ft. Laurderdale, FL.

[15]. Quinlan, J. R. (1993) : *C4.5 : programs for machine learning*. Morgan Kaufmann Publishers.

[16]. Quinlan, J.R. (1992): Learning with Continuos Classes. In *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*. Singapore: World Scientific, 1992.

[17]. Torgo, L. (1995) : Data Fitting with Rule-based Regression. In *Proceedings of the 2nd international workshop on Artificial Intelligence Techniques (AIT'95)*, Zizka,J. and Brazdil,P. (eds.). Brno, Czech Republic.

[18]. van Laarhoven,P. and Aarts,E. (1987) : Simulated annealing : Theory and Applications. Kluwer Academic Publishers.

[19]. Weiss, S. and Indurkhya, N. (1993) : Rule-base Regression. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence*, pp. 1072-1078.

[20]. Weiss, S. and Indurkhya, N. (1995) : Rule-based Machine Learning Methods for Functional Prediction. In Journal Of Artificial Intelligence Research (JAIR), volume 3, pp.383-403.