Chapter #

VARIABLE SELECTION IN INDUSTRIAL DATASETS USING PARETO GENETIC PROGRAMMING

Guido Smits[2], Arthur Kordon[1], Katherine Vladislavleva[2], Elsa Jordaan[2] and Mark Kotanchek[3]

[1]*The Dow Chemical Company*, Freeport, TX

[2]*Dow Benelux*, Terneuzen, The Netherlands

[3]*The Dow Chemical Company*, Midland, MI

Abstract: This chapter gives an overview, based on the experience from the Dow Chemical Company, of the importance of variable selection to build robust models from industrial datasets. A quick review of variable selection schemes based on linear techniques is given. A relatively simple fitness inheritance scheme is proposed to do nonlinear sensitivity analysis that is especially effective when combined with Pareto GP. The method is applied to two industrial datasets with good results.

Key words: Genetic programming; Symbolic Regression; Variable Selection; Pareto GP

1. INTRODUCTION

Many industrial applications are based on high-dimensional multivariate data. The dominant approach for data analysis in this case is dimensionality reduction by Principal Component Analysis (PCA) and building linear models with projections to latent structures by means of Partial Least Squares (PLS) (Eriksonn *et al*, 2001). This approach, however, has two key

issues: (1) the model interpretation is difficult and (2) it is limited to linear systems. One approach to extend this to nonlinear systems is to use neural networks. The variable selection algorithm in this case is based on gradually reducing the number of inputs until an optimal structure is obtained (Saltelli *et al*, 2001). However, this process is coupled with the hidden layer structure selection and requires high quality data sets. One of the unique features of Genetic Programming (GP) is its built-in mechanism to select the variables that are related to the problem during the simulated evolution and to gradually ignore variables that are not. In this way, a different type of nonlinear variable selection can be used for dimensionality reduction that could be appropriate for industrial data analysis. This idea was explored in (Gilbert *et al*, 1998) for variable selection from a spectral data set with 150 variables. Only between 6 and 9 variables were selected in the GP-derived predictive rules. Other applications can be found in (Francone *et al*, 2004, RML Technologies, 2002 and Johnson *et al*, 2000)

An approach for GP-based variable selection with emphasis on multi-objective Pareto-front GP will be described in the chapter. The organization is as follows. The generic issue of dealing with high dimensional spaces is addressed in Section 2. Section 3 gives a short overview of the linear techniques for variable selection. The proposed method for variable selection using Pareto GP is discussed and illustrated with synthesized data for re-discovering of Newton's Law of gravity in Section 4. The method is demonstrated with two successful industrial applications, described in Section 5.

2. THE CURSE OF DIMENSIONALITY

In modeling projects, the assumption is implicitly made that we know the "true" inputs to a given problem and that the reference data set feature vectors are defined in the space of these "true" inputs. In practice, many times one has to select the relevant inputs from a possibly large set of candidate inputs i.e. input selection is an integral part of the modeling problem. All too often, people don't worry too much about this input selection problem. They build models using all the available inputs thinking that, in a magical way, the modeling system will figure out which inputs are relevant and which are not. To build robust models it is essential to limit the number of inputs to an absolute minimum for a number of reasons which all have to do with the so-called "curse of dimensionality".

The goal of a data-driven modeling problem is to estimate an unknown function based on a finite number of samples. Because we only have a finite

number of samples available, there will always be an infinite number of possible functions that can be selected and that will interpolate the data equally well. To come up with a unique solution it is necessary to impose some kind of constraints on possible solutions. In the absence of first-principle constraints that can be obtained from a understanding of the physics behind the problem, these constraints are often defined in terms of the smoothness of the function in a given neighborhood around a data point. The accuracy of the function estimation obviously depends on having enough samples within that neighborhood. If the dimensionality of the problem is increased, there are basically two options to have sufficient data points within such a local neighborhood. First, one can try to collect sufficient samples to get this high density, something which is very often not possible. Second, one can increase the size of this 'local' neighborhood, but this is at the expense of imposing stronger (possible incorrect) constraints on the problem solution. This is the essence of the "curse of dimensionality".

The properties of high dimensional spaces often appear counter-intuitive because our experience is limited to low-dimensional spaces (Cherkassky, 1998). For example, objects like a hypercube have an increasing ratio of surface area to volume with increasing dimensionality. Following are four properties of high dimensional spaces that contribute to the problem:

- *Samples sizes with the same data density increase exponentially with dimensionality.* If ρ is the reference data density in one dimension ρ^d is the equivalent density in d dimensions.
- *An increasingly large radius is needed to enclose a given fraction of data points in higher dimensional space.* For example, the edge length of a hypercube which encloses a given fraction of samples p is given by:

$$el_d(p) = p^{1/d}$$

Figure 1 shows the corresponding graph for up to dimensionality 20 for fractions of 5, 10 and 20 %. Notice that high edge lengths (>0.75) are needed very quickly. An edge length of 1 would result in a hypercube that covers the entire space.

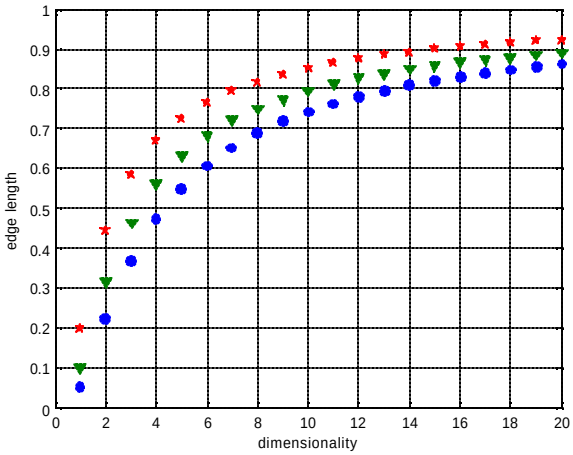

Figure #-1. Edge length needed for a hypercube to include 5%, 10% and 20% of the data.

- *In higher dimensional spaces, almost every point is closer to an edge than to another point.* Or, in other words, in higher dimensional spaces extrapolation is the norm rather than the exception. For a sample size *n*, the expected distance between data points sampled from a uniform distribution in the unit hypercube is given by:

$$D(d,n) = \frac{1}{2}\left(\frac{1}{n}\right)^{1/d}$$

- *Almost every point is an outlier in its own projection on the line defined by the prediction point and the origin.* The expected location of this prediction point is $\sqrt{d-1/2}$. The remaining points will follow a standard normal distribution with mean zero and standard deviation one since the other points are unrelated to the direction of the projection. For example when d=10, the expected value of the prediction point is 3.1 standard deviations away from the center of the training data. In this sense this point can be considered to be an outlier of the training data.

These properties of high dimensional spaces have serious consequences for building models based on a limited number of samples. The higher the dimensionality of the space the more likely it is that we will not have the data points we need to make a local estimate. Also, the higher the

dimensionality, the more we have to resort to extrapolation instead of interpolation to make predictions. For these reasons, it is essential to limit the number of inputs to a data driven model to an absolute minimum. In the next sections we will describe how this can be done using linear techniques but also how this can be done effectively using genetic programming.

3. VARIABLE SELECTION USING LINEAR TECHNIQUES

Before we discuss the reduction of input dimensionality using genetic programming we will quickly review how this can be achieved for linear models. Suppose we try to build a model of the form: $\mathbf{Y} = \mathbf{X}.\mathbf{b} + \mathbf{e}$ where X and Y are matrices with the inputs and the observations, $\mathbf{b}$ is a vector of parameters and e is the vector of errors. The least squares solution of b that minimizes $\mathbf{e}`.\ \mathbf{e}$ is obtained from $\mathbf{b} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{Y}$ irrespective of any distribution properties of the errors. A large number of statistical procedures are available to select the best subset of inputs to use in the linear regression equation (Draper and Smith, 1981). Examples of these procedures are: (1) all possible regressions, (2) best subset regressions, (3) backward elimination, (4) stepwise regression, (5) ridge regression, (6) principal components regression, (7) latent root regression, (8) stagewise regression etc. Some of these procedures, like principal components and latent root regression are specific to linear methods and do not have an immediate analogue to apply in non-linear modelling. Most of the other procedures rely on some sort of significance test (e.g. the partial F-test) to decide which variable to keep or to discard depending on the specific procedure. The use of these significance tests automatically implies an assumption about the underlying distribution of the errors. Frequently these are assumed to be normally distributed. The preferred procedures are either or a combination of the backward elimination and stepwise regression procedures. Although theoretically the all possible regression procedure would be the best, in practice this is only feasible for a limited number of possible inputs.

The backward elimination method starts from a regression equation containing all variables. At each iteration the variable with the lowest partial F-test value is compared to a preselected significance level and is eliminated whenever the significance is lower than the preselected value. The procedure stops when no more variables can be found that meet this criterion. The stepwise regression method attempts to achieve the same

result by working in the other direction, i.e. to insert variables into the equation as long as they meet certain significance criteria (see Draper and Smith, 1981] for more details on these procedures). These procedures usually work fine within a linear framework but one has to realize that there are many possibilities for any variable selection scheme to go wrong whenever the data set being used is not balanced (in the sense that not all input dimensions are properly represented) or some of the variables are related to other unmeasured latent variables.

4. VARIABLE SELECTION USING PARETO GENETIC PROGRAMMING

4.1 Fitness inheritance in the total population

As mentioned earlier, one of the potential applications of symbolic regression via genetic programming is sensitivity analysis of nonlinear problems with a potentially large set of candidate input variables. These kinds of problems are frequently encountered in the chemical processing industry. Sensitivity analysis is also called the problem of feature selection in machine learning terminology. Many problems in the chemical industry are of this type. There usually are a large number of measurements available at a plant, many of which are redundant or not relevant to the problem that one tries to solve.

Engineering knowledge about the problem is usually the first step to try and narrow down the number of inputs. Sensitivity analysis generates a ranking of all the input variables in terms of how important they are in modeling a certain unknown process. In linear problems the sensitivity of an input variable is related to the derivative of the output with respect to that variable. In nonlinear problems, however, the derivative becomes a local property and has to be integrated over the entire input domain to qualify as a sensitivity. Since this approach is not really practical in a genetic programming context we've opted to relate the sensitivity of a given input variable to its fitness in the population of equations. The reasoning is that important input variables will be used in equations that have a relatively high fitness. So the fitness of input variables is related to the fitness of the equations they are used in. There is, however, a question with respect to credit assignment i.e. what portion of the fitness goes to what variable in the equation. The easiest approach is to distribute the credit (the fitness of the equation) equally over all variables present. A complicating factor is that probably not every variable is equally important in a given equation. In

addition, most equations in a genetic programming population are not parsimonious and possess chunks of inactive code (a good description of the problem of 'bloat' can be found in Banzhaf *et al*, 1998). Variables that are present in these chunks of inactive code do not contribute to the final fitness of the equation but still obtain some credit for being part of that equation. There is no direct solution for this problem on the individual equation level but still reliable answers can be obtained provided we evaluate a large number of equations. Again the reasoning is simple, if a given input variable is absolutely essential to solve the problem, it must be present in the high fitness equations. Other nonessential variables will be present in both low-fitness and high-fitness equations so their fitness will be closer to the average fitness over all equations. More important variables will obtain more credit and will have a fitness that exceeds this average value. So provided the population size is large enough we can take the fitness of each equation in the population, distribute this fitness in equal amounts over the input variables present in that equation and sum all these contributions for each input variable over the entire population. An improved version of this, at the expense of a little bit of extra computation, is doing the same but instead of just using the equations in the population also include every sub equation in each of these equations. The extra computational step will considerably improve the statistics of the input variable fitnesses since now the number of equations is equal to the *total number of nodes in every equation-tree in the population* rather than the population size itself.

4.2 A simple example

As a simple example we'll try to rediscover Newton's Law of gravitation. This states that any two objects attract one another gravitationally. The attractive force depends linearly on the mass of each object (doubling the mass doubles the force) and inversely on the square of the distance between the two objects:

$$F = -g.\frac{m_1.m_2}{r^2}$$

g is the gravitational constant which is just a number to match up the results of the equation with our system of measurement. The '-' sign indicates that the force is attractive. A synthetic dataset with 50 patterns was generated where the two masses are random numbers in the range [0,1] and the radius is a real number in the range [1,2]. An additional 50 inputs with

random numbers in the range [0,1] were added. These extra inputs are just ”noise” variables and make the task of discovering Newton’s law progressively harder since part of the problem now is to discover the ”true” variables x_1, x_2 and x_3 in the total set.

From figure 2, which shows the correlation coefficient from each of the input variables to the force, the output variable that needs to be predicted, we see that the masses (variables 1 and 2) have a relatively high absolute correlation to the force but the others (the distance and the random variables) cannot be easily distinguished from each other.

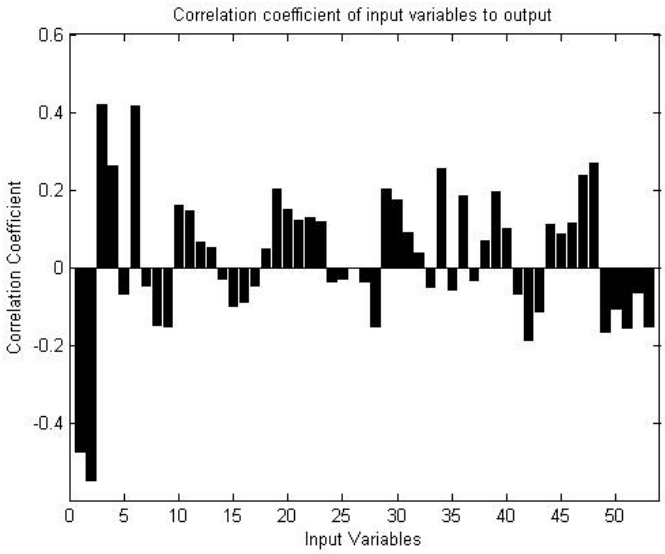

Figure #-2. Correlation coefficients of inputs relative to the output for Newton's problem

Next we’ll apply genetic programming to do a nonlinear sensitivity analysis. The particular version of Genetic Programming we use is called Pareto GP and is described in (Smits and Kotanchek, 2004). In Pareto GP an archive is used to store equations that are at or near the Pareto border of fitness versus some equation complexity measure. This archive is maintained during a run. All the equations in the next generation are obtained either by mutation of existing equations in the archive or by crossover between members of the archive and the previous population. A typical run consists of a number of different cascades. At the start of a new cascade a new population is generated from scratch but, since the archive is maintained, good solutions appear very quickly again in the population. A cascade usually has a fixed number of generations. The final result is the set

of the equations in the archive, which represents the Pareto front of fitness versus complexity. When we apply genetic programming to do a nonlinear sensitivity analysis to the augmented Newton dataset as described earlier, the picture shown in Figure 3 emerges. The important variables are identified very quickly and then stabilize in sensitivity. The unimportant variables die out relatively fast after an initial period and final fitness values become very small.

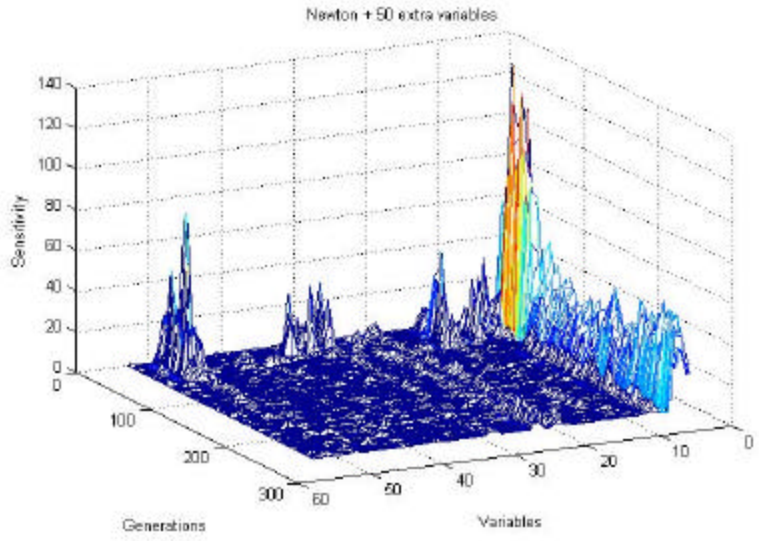

Figure #-3. Evolution of fitness for all variables in the Newton problem. Notice that the important variables on the right-hand side are identified very quickly

4.3 Fitness inheritance in the Pareto front only

We already mentioned that when variables accumulate fitness from the entire population there is a chance that we get somewhat of a distorted picture because unimportant variables that happen to be part of equations with a relatively high fitness will also pick up fitness from those. To compensate for this we introduced a modification where variables only accumulate fitness from equations that reside in the archive. Since the archive contains the Pareto front of all of the high fitness equations relative to their complexity this modifications is expected to make the variable

selection more robust. In the next section with two industrial applications we will show that this is indeed the case.

5. APPLICATIONS

Variable selection and dimensionality reduction is critical for developing parsimonious empirical models from industrial data sets. One of the key application areas of symbolic regression models, generated by GP is inferential sensors (Kordon *at al*, 2003). This type of empirical models predicts difficult-to-measure process variables (outputs), such as NOx emissions, polymer properties, biomass, etc., with easy-to-measure sensors, such as temperatures, flows, and pressures (inputs). Usually model development begins with the broadest possible selection of input sensors that process engineers think may influence the output.

The proposed method for variable selection will be illustrated in two applications of inferential sensors development on (1) a data set with middle-sized dimensionality (8 inputs and 251 data points) and on (2) a high-dimensional data set of 23 inputs and 7000 data points.

5.1 Variable selection on middle-sized industrial data

The inferential sensor in this application predicts emissions from process variables. The correlation coefficients of the eight potential inputs relative to the emissions (the output) are shown in Figure 4. For this problem it is difficult to satisfy the regulatory requirements of 7.5% error with a linear model and so a nonlinear solution is needed.

Pareto GP was used for variable selection and nonlinear model generation. The results from the variable selection are shown in Figures 5-8. The results are based on 5 independent runs of 10 cascades with 50 generations. The average sensitivities with their standard deviations for each input, as defined in the previous section, for two population sizes of 100 and 1000 are shown in each figure. The sensitivities in Figures 5 and 6 are based on all models in the population at the last generation and the sensitivities in Figures 7 and 8 are based on the models in the archive at the last generation of the Pareto GP evolution.

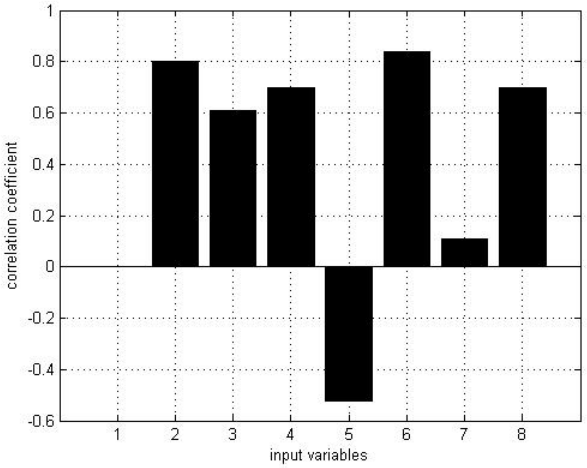

Figure #-4. Correlation coefficients of process inputs relative to emissions

The last sensitivity analysis is a better foundation for variable selection because it is based on the high-quality potential models which are the breeding source for the Pareto front. In principle, the most sensitive inputs (x2, x5, x6, and x8) have been consistently selected in all cases but the difference is clearer with the archive selection in Figures 7 and 8. For comparison, a linear variable selection, based on PCA-PLS model with two principal components, is shown in Figure 9. The inputs ranking is represented by a Variable Importance in the Projection (VIP, described in Eriksonn et al, 2001). Variables with VIP > 1 are treated as important.

One of the differences between the linear and the GP-based variable selection is that input x5 is insignificant from the linear point of view (which is supported by the low correlation coefficient of -0.5 in Figure 4). However, it is one of the most significant inputs, according to the nonlinear sensitivity analysis and process experts. The experts also selected two models for the final implementation, which included the four most influential inputs from the GP variable selection – x2, x5, x6, and x8. The correlation coefficient of the selected models is 0.93 and 0.94, much higher than the linear option and within the regulatory limits. The application details are given in (Kordon *at al*, 2003).

Table #-1. Sensitivity analysis of models in the population at the last generation (mean and standard deviation over five independent runs)

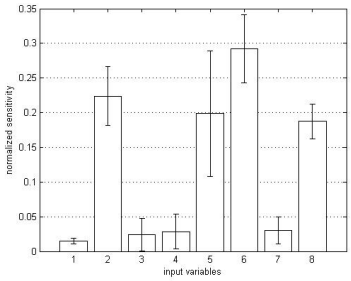

Figure #-5. Population size 100

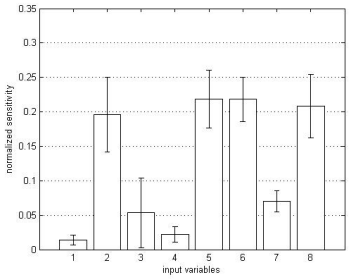

Figure #-6. Population size 1000

Table #-2. Sensitivity analysis of models in the archive at the last generation (mean and standard deviation over five independent runs)

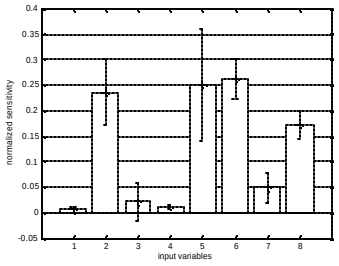

Figure #-7. Population size 100

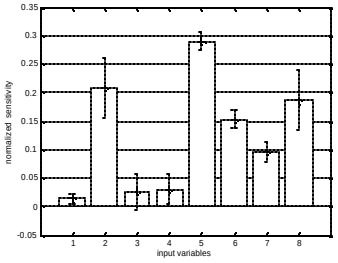

Figure #-8. Population size 1000

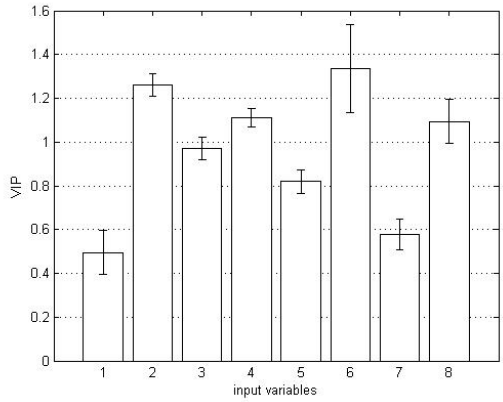

Figure #-9. Variable importance in the projection (VIP) of the 8 inputs based on a two principal components PCA-PLS model of the emissions soft sensor

5.2 Variable selection on high dimensional industrial data

The inferential sensor in the high dimensional application predicts propylene concentration. This application illustrates the scale-up performance of the proposed method on an industrial problem with a much larger search space. The results from the GP sensitivity analysis are shown in Figure 10,11 and the results from the corresponding linear variable ranking are shown in Figure 12.

In this case the difference between the linear and nonlinear variable selection is significant. The GP-based sensitivity analysis identifies four clear winners – inputs x4, x6, x8, and x21 (see Figures 10,11 whereas the linear variable ranking suggests 12 important variables with VIP > 1 – inputs x4, x5, x6, x8, x9, x19, x11, x12, x14, x15, x20, and x22 (see Figure 12). The proposed reduction of the search space, based on the linear ranking is much less and an important variable, input x21 is missing. For the final implementation an ensemble of four models has been designed. The selected models from the process expert included the four inputs, based on the GP sensitivity analysis, i.e., inputs x4, x6, x8, and x21, and input x11, recommended in a backup model from the experts. The application details are given in (Jordaan *at al*, 2004).

Table #-2. Sensitivity analysis of models in the archive at the last generation (mean and standard deviation over five independent runs)

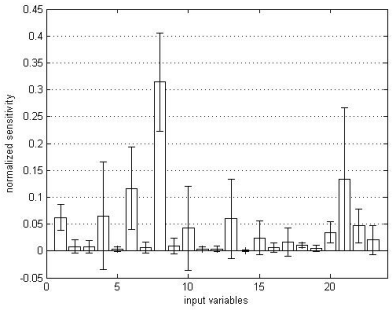

Figure #-10. Population size 100

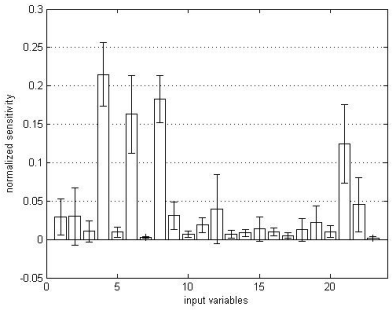

Figure #-11. Population size 1000

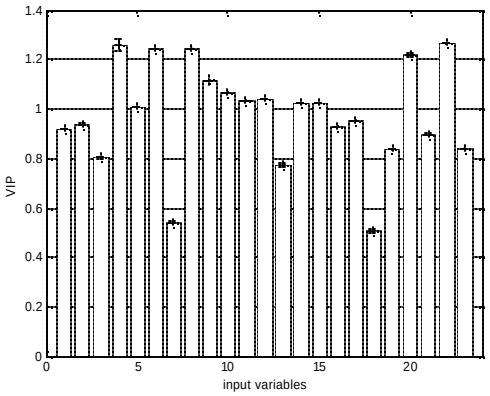

Figure #-12. Variable importance in the projection (VIP) of the 23 inputs based on a four principal components PCA-PLS model of the propylene soft sensor

6. SUMMARY

Sensitivity analysis using Genetic Programming and more specifically Pareto GP has been used successfully in many industrial applications to do nonlinear variable selection. It has been observed that the results are quite

consistent and often allow for a considerable reduction in the feature space before final models are built. This, in general, leads to more robust models. The variable sensitivity is accomplished through a relatively simple fitness inheritance scheme that imposes little additional overhead in terms of computational effort.

References

Banzhaf, W., Nordin, P., Keller, R, and Francone, F. (1998). *Genetic Programming: An Introduction*, San Francisco, CA: Morgan Kaufmann.

Cherkassky V, Mulier, F., 1998, "Learning from data, Concepts, Theory and Methods", Wiley Interscience, ISBN 0-471-15493-8.

Draper, N. R. and Smith, H. (1981) *Applied Regression Analysis, Second Edition*, New York, NY: Wiley.

Eriksson, L., Johansson, E., Wold, N., and Wold, S. (2001). *Multi and Megavariate Data Analysis: Principles and Applications*, Umea, Sweden, Umetrics Academy.

Francone, F. *et al* (2004). *Discrimination of Unexploded Ordnance from Clutter Using Linear Genetic Programming*, Genetic and Evolutionary Computation Conference, Late Breaking Papers.

Gilbert, R.J., Goodacre, R., Shann, B., Taylor, J., Rowland, J.J. and Kell, D.B., *Genetic Programming-Based Variable Selection for High-Dimensional Data*, in J.R.Koza et al., editors, Genetic Programming 1998: Proceedings of the Third Annual Conference (GP-98), Madison, WI 22-25 July 1998, Morgan Kaufmann, San Fransisco, CA.

Ohnson, H.E, Gilbert, R.J., Winson, M.K., Goodacre, R., Smith, A.R., Rowand, J.J., Hall, M.A. and Kell, D.B. *Explanatory Analysis of he Metabolome Using Genetic Programming of Simple, Interpretable Rules*, in Genetic Programming and Evolvable Machines, Vol 1 (2000)

Jordaan, E., Kordon, A., Smits, G., and Chiang L. (2004). Robust Inferential Sensors based on Ensemble of predictors generated by Genetic Programming, In *Proceedings of PPSN 2004*, pp. 522-531, Birmingham, UK.

Kordon, A., Smits, G., Kalos, A., and Jordaan**,** E.(2003). Robust Soft Sensor Development Using Genetic Programming, In *Nature-Inspired Methods in Chemometrics***,** (R. Leardi-Editor), Amsterdam: Elsevier

Kotanchek, M, Smits, G. and Kordon, A. (2003). Industrial Strength Genetic Programming, In *Genetic Programming Theory and Practice,pp 239-258*, R. Riolo and B. Worzel (Eds), Boston, MA:Kluwer.

Koza, J. (1992). Genetic Programming: On the Programming of Computers by Means of Natural Selection, Cambridge, MA: MIT Press.

RML Technologies, Inc. (2002) *Discipulus Owner's Manual*.

Saltelli A., Chan K., and Scott E. (2001). Sensitivity Analysis, Baffins Lane, Chichester, UK: Wiley.

Smits, G. and Kotanchek. (2004), Pareto -Front Exploitation in Symbolic Regression, *Genetic Programming Theory and Practice, pp 283-300*, R. Riolo and B. Worzel (Eds), Boston, MA:Kluwer.