Proceedings of ECML-95 Workshop on Statistics, Machine Learning and Knowledge Discovery in Databases, 1995, Heraklion, Greece, pp 187-192

PolyAnalyst 2.0: Combination of Statistical Data Preprocessing and Symbolic KDD Technique.

Mikhail V.Kiselev
Computer Patient Monitoring Laboratory at National Research Center of Surgery, per.Abrikosovski, 2, Moscow 119874, RUSSIA
e-mail: geography@glas.apc.org

A B S T R A C T

Data mining system PolyAnalyst [Kiselev94] builds and tests hypotheses about interdependences or other regularities in data in form of functional programs constructed recursively from simpler programs starting from a certain set of elementary primitives. Experience of use of PolyAnalyst shows that its performance is limited significantly by the three following undesirable factors which are typical for many other KDD systems as well. 1. Search in very wide hypotheses spaces consumes great computation time. Most undesirable feature is strong dependence of computation time on number of attributes in explored relational database (number of independent variables). 2. Every method based on the least squares criterion is vulnerable to even small number of far outliers originated from various data collection errors while it may be quite tolerant to strong normally distributed noise added to true values. 3. There exists no good general criterion for search termination. It was found that statistical exploration of data carried out before starting the main search process can give effective cure for these difficulties. This preprocessing was realized in the stand alone utility ARNAVAC. In combination with the data mining system PolyAnalyst it was used for exploration of large databases from medicine and geophysics.

Keywords: Machine discovery, database exploration, discovery of functional relationship, noise elimination.

1. Introduction.

During last decade a number of knowledge discovery systems were created which detect structure hidden in data in form of functional dependencies between attributes and formulate them as mathematical equations or other symbolic rules. This class of systems includes BACON [Langley, Simon, Bradshaw & Zytkow 87], ABACUS [Falkenhainer & Michalski 90], FAHRENHEIT [Zytkow & Zhu 91], KDW [Piatetski-Shapiro & Matheus 91], KEPLER [Wu & Wang 89] and many others. One of most developed systems which can discover very complex and diverse equations, solves systematically problems of data error analysis and evaluates statistical significance of obtained results is 49er [Zytkow & Zembowicz 93]. KDD system PolyAnalyst [Kiselev 94; Kiselev, Arseniev & Flerov 94] designed by us discovers empirical laws in data in form of functional programs constructed from standard and user-defined functional primitives. Although the systems which discover numerical dependencies in data use diverse knowledge representation formalisms and search methods they face the same set of difficulties inherent to their approach.

A. Work of the above mentioned systems is based on search in some hypotheses spaces. If we would like to be able to discover wide class of dependences and other regularities we have to realize search in very wide spaces.Therefore computation time is the most limiting factor. Most KDD systems are designed to explore relational databases. It is known that while dependence of their computation time on the number of records (rows) is usually linear or close to linear, dependence on the number of database attributes (columns) is much stronger.

B. The natural measure of fitness of found formula expressing numerical dependence is the standard error. Nevertheless, methods based on minimization of standard error have a major drawback - they are unstable to presence of even small number of erroneous data when their distribution differs strongly from the distribution of the rest data. This kind of noise leads to great uncertainity in values of constants entering discovered empirical laws and even to impossibility to select correct form of sought dependence from many candidates. Meanwhile these methods are fairly tolerant to strong random noise added to correct values. For example, in our tests we spoiled exact functional dependence between two variables by two methods. In the first test we substituted some values of dependent variable by random values distributed uniformly in the same limits as the dependent variable. We call it **pure noise**. Percent of substituted points served as a measure of contamination. In databases this kind of noise represents the pure data collection errors. In the second test we added to correct values normally distributed random numbers centered at zero. In that test measure of impurity was ratio of dispersion of these random number to initial dispersion of dependent variable. We call this noise component **additive noise**. It models measurement errors, influence of some hidden factors, roundoff error, etc. Then our KDD system PolyAnalyst tried to discover correct dependence in these noised data. The tests show that 3% contamination in the first test causes the same effect as approx. 70% contamination in the second test.

C. For many KDD systems the question "When stop the search?" is very hard to answer. There exists no good general criterion for search termination. For most sufficiently rich formalisms there is no practical method to prove that found formula is the best possible one and further search is useless. If we have ahead infinite number of new equations not reducible to found ones the possibility always remains that some of them express precious long-awaited discovery.

Very attractive approach to solution of the problem A and partially of the problem B was proposed in [Zembowicz & Zytkow 93]. In this paper a simple, fast and general algorithm was presented which detects presence in data (not exact) functional dependence between two variables not recognizing particular form of the dependence. This neat algorithm finds pairs of inter-dependent variables even under conditions of high noise level and is used for prevention of hopeless but time consuming search for non existing dependences by 49er system. However this algorithm is not free of some drawbacks. For example, it does not perform very well in case of very unevenly distributed data; its results depend on functional transformations of dependent and independent variables (for example, $x \to x^2$). It contains three constants whose values are to be determined from domain knowledge. Beside that it does not solve the problems B and C.

In the present paper we propose more general procedure whose purpose is to solve the problems A-C. This procedure was realized as a fast data preprocessing utility running before the main machine discovery system (PolyAnalyst in our case). Our requirements to this utility were the following:

- it should include fast noise-tolerant algorithm detecting multidimensional dependences in data;
- it should be general and (ideally) not include free parameters determined by user;
- it should not depend on functional transformations applied to variables (transformations realized by reversible functions of one argument).

Beside that the other desirable feature of our algoritm would be its ability to calculate some continuous parameter characterizing degree of functional relationship. If our algorithm proves to be fast enough it can be used as a data browser which would recognize groups of variables linked by the most explicit functional relationships and prevent time-expensive search for numerical equations expressing non existing dependencies. If it will be able to recognize and discard the pure noise subset of data records it would solve problem B. At last if it will evaluate standard error characterizing detected functional relationship it would solve problem C (see Section 4). However before we begin describing the algorithm we overview very briefly PolyAnalyst system which is a consumer of the results obtained by discussed data preprocessing utility.More detailed description of PolyAnalyst can be found in [Kiselev 94].

2. PolyAnalyst: search in the space of functional programs.

PolyAnalyst analyzes data represented as a set of records consisting of one or several fields so that all records have the same format. It can discover dependence of some field on other fields and express it (if it exists) in explicit symbolic form. Beside discovery of numerical dependencies this class of problems includes also classification and problems close to grammar inference.

The kernel of data mining system PolyAnalyst is a mechanism which builds new functional programs from existing functional programs. If not to consider semantics of these programs they can be understood as abstract objects with some number of inputs (or without inputs) and 1 output. The simplest atomic functional procedures are called primitives. The set of primitives is determined by structure and properties of data to be analyzed. The set of primitives includes standard and user-defined primitives. Information in databases is accessed via special data access primitives which also can be standard and user-defined. Such an architecture gives PolyAnalyst flexibility because the user-defined data access primitives can issue SQL requests, read data files or support more sophisticated data access schemes not bound by the relational database ideology.

To build new functional programs from existing ones PolyAnalyst uses four production schemes. The simplest scheme is **functional composition**. PolyAnalyst takes one functional program (it is called producing function) and connects some its inputs with outputs of some other existing programs. The second scheme serves to realize such often used concepts as $\exists$ and $\forall$ quantors, calculation of average, integration, etc. To form new functional program using this scheme PolyAnalyst selects some existing program returning numerical or boolean values. The new program returns sum (or result of boolean OR operation) of values returned by the old procedure which correspond to all possible combinations of values of certain old procedure's arguments. The third scheme realizes analogue of **while** construction of the conventional programming languages like C or PASCAL. At last the scheme most important for discovery of the numerical laws produces functional program realizing rational expression (a polynom divided by a polynom) formed from numerical constants and some existing programs (naturally, they should return numerical values).

In fact, this mechanism realizes a simple universal programming language suitable for formalization of wide range of laws and rules which can be discovered in data. For example, **if** construction can be expressed using functional composition with the special primitive called **TF-commutator** taken as a producing function. First argument of the TF-commutator is boolean. If it has value **true** then result of the TF-commutator is equal to value of it's second argument, else it is equal to value of the third argument.

The generator of functional programs is controlled by the search strategy module. The search direction is determined in accordance with evaluation of each individual functional program carried out by the search strategy module. In case of discovery of numerical laws the main evaluation criterion is the standard error. The search process is a combination of full search (low priority component) and generalizing transformations - GT (high priority component). The GT process takes one of the best found programs (called root program) and uses it for creation of new programs with help of one of the above mentioned production schemes in all possible ways satisfying the following condition. Every derivative program should have some set of arguments such that when these arguments have certain constant values the derivative program becomes identical to the root program. This condition guarantees that the derivative programs is not worse (in terms of standard error) than the root program. In normal situations the GT process takes the most part of computation time of PolyAnalyst.

3. Discovery of multidimensional functional dependencies in relational databases.

In Introduction we formulated three problems which are common for many KDD systems using symbolic representation of obtained knowledge (they will be referred to as 'symbolic KDD systems') and showed how these problems could be solved by a fast data preprocessor detecting presence of multidimensional functional relationships in data. Unfortunately because of strict limitations on size of the article we cannot include the full formal description of preprocessor's algorithm. Instead we outline below only its basic steps.

A relational database DB can be represented as a rectangular table with N rows and $M+1$ columns. One column will be considered as a 'dependent variable'. The other M columns are 'independent variables'. Since functional relationships between real-valued variables are most important for us we restrict our consideration to databases containing real numbers only although in fact our algorithm can be extended easily to integer an cathegorical variables also. Thus, database can be represented as a finite set of points in $M+1$ -dimensional space R^{M+1}. Coordinates in R^{M+1} corresponding to the independent variables will be denoted as x_i, $i=1,...,M$; a coordinate corresponding to the dependent variable will be denoted as y.

We impose the following important limitation on the number of independent variables: $M \leq \frac{1}{2}\log_2 N$. It does not mean that considered algorithm is useless for analyzis of relational databases with the greater number of columns. In these cases the algorithm runs several times on different subsets of independent variables. Some questions concerning strategy of choise of these subsets are touched in section 4.

Presence of noise makes it impossible to determine functional dependence considering positions of individual points of DB in R^{M+1}. Instead of that, our approach is based on breaking the space of independent variable values R^M to certain set of regions $\{E_i\}$ and comparing distributions of y for each region E_i. Namely, we cut R^M by hyperplanes $x_i = const$ and take the rectangular regions formed by these hyperplanes as E_j. We call such set of regions the grid $\{E_j\}$. The hyperplanes forming the grid may be constructed by various methods. However it is important that datapoints would be distributed among the cells E_j as evenly as possible.

Consider one cell E_i. Let Y_i be a set of all values of the dependent variable for data points laying in E_i; Y be a set of all values of the dependent variable. If distributions of y for many cells differ significantly from total distribution of y it can be considered as a sign of presence of functional dependence of y on x_i. Comparing Y_i and Y we reveal a compact "core" in Y_i with distribution maximally different from Y. Such a core may mark the set of values $f(E_i)$ of a hypothetical function $y = f(x_1,...,x_M)$ expressing sought functional relationship.

This procedure is applied to each region E_i. The subset of DB consisting of all the datapoints which belong to some core will be called **FR-subpopulation** of DB; number of datapoints in FR-subpopulation divided by N will be denoted as ν_{FR}. Strength of the functional relationship showed by the FR-subpopulation can be expressed as $S_{FR} = b(N\nu_{FR}, N, A)$, where $b(k,K,p)$ is a tail area probability of the binomial distribution with the number of trials K and the event probability p, $A = \frac{1}{Nn_{cores}} \sum L_i$, where L_i is the number of datapoints from DB whose y lay inside core interval i.

Thus, we have a procedure discovering in each database DB a subset which most probably expresses functional dependence between the dependent variable y and the independent variables x_i and calculates two parameters characterizing strength of this dependence: ν_{FR} and S_{FR}. The first one determines which part of DB obeys this dependence. The second one characterizes its fuzzyness. This procedure has one free parameter H - the number of grid cells per each dimension of space of independent variables. It can be shown that the best choice of H is an integer number nearest to $N^{1/2M}$.

Our algorithms constructing grid and determining cores in the grid cells are designed so that determination of FR-subpopulation does not depend on selected parametrization of coordinate axes of R^{M+1}.

Of course, discussed procedure is more complex and therefore slower than algorithm in [Zembowicz & Zytkow 93]. However dependence of its computation time on N and M is surprisingly weak. Asymptotical dependence of computation time of our algorithm on N is $O(N \log N)$. Since M cannot be very big it is senseless to say about the asymptotical dependence on M. Really the dependence of the computation time on M is very weak because the number of grid cells does not depend on M (it is $O(\sqrt{N})$).

The discussed algorithm has one useful side effect. If we try to discover an equation expressing functional dependence of y on x_i over the FR-subpopulation then we will hardly achieve standard deviation much less than weighted average of dispersion of values belonging to cores in all cells. Here the numbers of core points in grid cells serve as weights. Thus, this value (we denote it as r_{FR}) gives an approximate estimate for the minimum possible standard error of desired symbolic equation.

4. ARNAVAC utility and its interaction with PolyAnalyst.

The algorithm described in the previous section was realized by us in the data preprocessing utility ARNAVAC. The main purpose of ARNAVAC utility is to help KDD system PolyAnalyst to solve the problems A.-C. mentioned in the Introduction .

1. ARNAVAC determines groups of variables showing strongest influence on the dependent variable using the algorithm from the previous section.

2. Simultaneously ARNAVAC discards all datapoints which do not belong to the respective FR-subpopulation thus reducing noise in the new database.

3. ARNAVAC computes value of r_{FR} for this FR-subpopulation. If PolyAnalyst analyzes a database preprocessed by ARNAVAC it stops the search when standard error of the best found empirical law reaches this value.

ARNAVAC itself can be considered as a simple and fast data mining tool. because it can build its own predictive procedure for each database. This procedure is based on the table "grid cell - predicted value of dependent variable" created by ARNAVAC. The average value of the dependent variable for points belonging to core of some cell is taken as predicted value for this cell. The table characterizing parameters of FR-subpopulation in different cells which can be printed by ARNAVAC has proven to be useful for visual preliminary evaluation of quality of data and probability to find something interesting in them.

5. Detection of the one-dimensional functional dependence in the test data.

As an illustration of behavior of our algorithm we give here the results of its use for analysis of the three following test databases with $M = 1, N = 1000$. All they contain pure functional dependence $y = \frac{x}{(x^2+1)}$ for x distributed uniformly in [0, 7] which is contaminated artificially by noise. We used two kinds of noise (see discussion in Introduction): additive noise with dispersion d_{add} and pure noise including s_{pure} percent of all datapoints (1000 in our case). Strength of additive noise is expressed conveniently by the dimensionless value s_{add} which is the ratio of d_{add} and dispersion of y calculated exactly using the formula above. We varied s_{add} and s_{pure} in these three cases and obtained the following results (Fig. 1). On the second and third graphs the points belonginfg to FR-subpopulation are marked by crosses. On the first graph the points NOT belonginfg to FR-subpopulation are marked by crosses. We use the one-dimensional dependence in our test example to make the results more interpretable and to avoid difficulties of visualization of point distributions in multi-dimensional spaces.

Results obtained by ARNAVAC and PolyAnalyst for these three cases are summarized in Table 1.

Table 1. Results of exploration of the test data with the different levels of noise (Fig.1) by ARNAVAC and PolyAnalyst

s_{add}	0.3%	30%	30%
s_{pure}	3%	10%	80%
ν_{FR}	0.952	0.883	0.058
$\ln S_{FR}$	-2870	-583.5	-20.25
r_{FR}	8.09%	29%	29.5%
Equation obtained by PolyAnalyst for whole dataset	$\frac{0.986x}{x^2 + 0.9735}$	$\frac{0.907x}{x^2 + 0.7975}$	$\frac{44.8}{x^2 + 162}$
Equation obtained by PolyAnalyst for the FR-subpopulation	$\frac{0.99995x}{x^2 + 0.9995}$	$\frac{0.986x}{x^2 + 0.958}$	$\frac{1.12x}{x^2 + 1.975}$
Standard error of the equation for the FR-subpopulation	1.52%	28.9%	39.55%

First of all we can see that relative part of the pure noise component in data 1 - ν_{FR} determined by ARNAVAC is close to its true value s_{pure}. It is seen that the algorithm described in section 3 is quite strict. It tends to consider as noise some points of the true FR-subpopulation laying near its edges: 1 - $\nu_{FR} > s_{pure}$. For this reason in case of low quality data we use a less strict modification of the algorithm. Table 1 shows that value of $\ln S_{FR}$ is a good indicator of quality of data.Close values of r_{FR} and s_{add} confirm the idea of use of r_{FR} as an estimate for standard error of best formula which could be found by PolyAnalyst. It works unsatisfactorily only in rare cases of very exact functional dependence because in these situations the discretization error inherent to the algorithm prevails over data errors. At last the equations obtained by

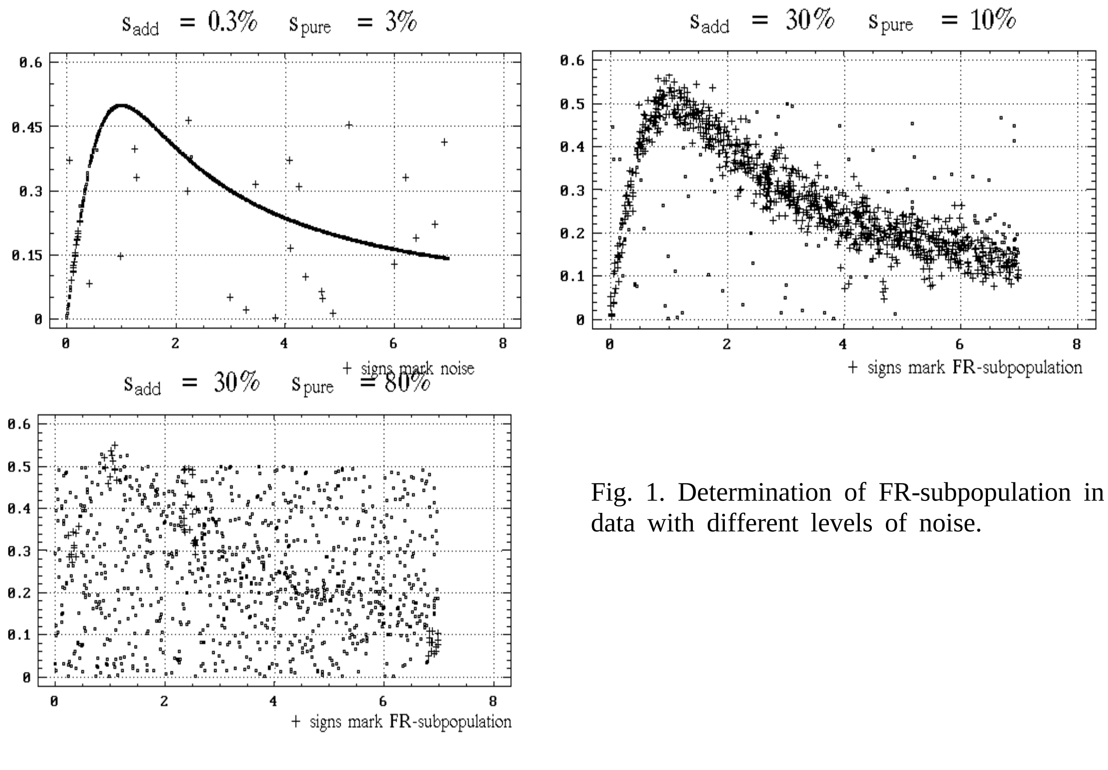

Fig. 1. Determination of FR-subpopulation in data with different levels of noise.

PolyAnalyst for unprocessed data and for FR-subpopulation show illustratively how elimination of noise increases accuracy of determination of constants entering these equations and helps to find a correct formula even in almost hopeless situations like case 3 of Fig. 1.

6. Conclusion

It is evident from this article that the data preprocessor detecting multi-dimensional functional relationships in data is a natural and necessary supplement to symbolic KDD systems. The algorithm which was proposed as a basis for such a preprocessor allows to reach also two other goals. It is able to "purify" the detected functional dependence discarding erroneous/exceptional datapoints which do not obey this dependence and to evaluate the standard error characterizing it. Three major benefits obtained from use of the preprocessor are:

- substantial decrease of computation time;
- improved stability of result obtained by symbolic KDD methods, greater accuracy of computation of constants entering discovered equations;
- good criterion for termination of search performed by symbolic KDD system.

The proposed algorithm was implemented in the data browser/preprocessor utility ARNAVAC. In combination with the data mining system PolyAnalyst it has proven to be a powerful tool for analyzis of scientific and industrial databases. High efficiency of discussed technique and its realization in ARNAVAC utility was demonstrated by theoretical evaluations, by tests on artificially generated data and by successful solution of real problems from geophysics and medicine.

REFERENCES

ARSENIEV, S.B. & KISELEV, M.V. (1991)
The Object-Oriented Approach to the Medical Real Time System Design, Proceedings of MIE-91, In: Lecture Notes in Medical Informatics, Springer-Verlag, Berlin, v.45, pp 508-512

FALKENHAINER, B.C. & MICHALSKI, R.S. (1990)
Integrating Quantitative and Qualitative Discovery in the ABACUS System In: Y.Kodratoff, R.S.Michalski, (Eds.): Machine Learning: An Artificial Intelligence Approach (Volume III). San Mateo, CA: Kaufmann, pp 153-190.

KISELEV, M.V. (1994)
PolyAnalyst - a Machine Discovery System Inferring Functional Programs, Proceedings of AAAI Workshop on Knowledge Discovery in Databases’94, Seattle, pp 237-249.

KISELEV, M.V., ARSENIEV, S.B. & FLEROV E.V. (1994)
PolyAnalyst - a Machine Discovery System for Intelligent Analysis of Clinical Data, ESCTAIC-4 Abstracts (European Society for Computer Technology in Anaesthesiology and Intensive Care), Halkidiki, Greece, p. H6.

LANGLEY, P., SIMON, H.A., BRADSHAW, G.L. & ZYTKOW, J.M. (1987)
Scientific discovery: Computational explorations of the creative processes. Cambridge, MA: MIT Press.

PIATETSKI-SHAPIRO, G. & MATHEUS, C. (1991)
Knowledge Discovery Workbench, Proceedings of AAAI Workshop on Knowledge Discovery in Databases’91, pp 11-24.

WU, Y. & WANG S. (1989)
Discovering Knowledge from Observational Data, Proceedings of IJCAI-89 Workshop on Knowledge Discovery in Databases, Detroit, MI, pp.369-377.

ZEMBOWICZ, R. & ZYTKOW J.M. (1993)
Testing the Existence of Functional Relationship in Data, Proceedings of AAAI Workshop on Knowledge Discovery in Databases’93, Washington, D.C., pp 102-111.

ZYTKOW, J.M. & ZEMBOWICZ R. (1993)
Database Exploration in Search of Regularities, Journal of Inteligent Information Systems, Vol.2, pp.39-81.

ZYTKOW, J.M. & ZHU, J. (1991)
Automated Empirical Discovery in a Numerical Space, Proceedings of the Third Chinese Machine Learning Workshop, Harbin, Peoples’ Republic of China, pp.1-11.