1991

Management of regression-model data

Rowe, Neil C.

Monterey, California. Naval Postgraduate School

Data and Knowledge Engineering, 6 (1991), 349-363.
http://hdl.handle.net/10945/36565

Management of regression-model data

Neil C. Rowe

Department of Computer Science

Code 52 Naval Postgraduate School Monterey, CA 93943

Abstract

Statisticians are becoming increasingly ambitious in the size and complexity of the data populations they are analyzing. Database support is becoming increasingly important to them, and in particular, they are demanding archiving of analysis results for possible later use, a facility not generally provided in statistical packages. We discuss the key database issues of managing regression-data models, one such analysis result, and we propose data structures including multiple partial indexes to support model-inference methods. We suggest a role for inference methods analogous to those used in artificial intelligence (especially inheritance inferences), to provide a generalization of the techniques of analysis of variance and covariance. A key feature of our approach is the quantification of inferences to resolve "multiple inheritance" conflicts between models applicable to a new situation.

The work reported herein was partially supported by the Foundation Research Program of the Naval Postgraduate School with funds provided by the Chief of Naval Research. The views and conclusions contained in this document are those of the author and should not be interpreted as representative of the official policies of the Navy or the U. S. Government. Frank Olken suggested the topic, and Jim Dolby, Bill Gale, Klaus Meyer-Wegener, and Daryl Pregibon provided helpful comments. This paper appeared in *Data and Knowledge Engineering, 6* (1991), 349-363. The equations were redrawn in 2008.

Key phrases: statistical computing, statistical models, regression, databases, knowledge representation, inheritance, evidence combination, statistical analysis--automatic, estimation, analysis of variance.

1. Introduction

In statistical analysis of a large database, statistical users rarely pick study problems at random, but overlap their tracks. When a user is working on something closely related to a previous analysis, they want and need to know. Good database design can help. In particular, statistical models could be stored from previous analyses, indexed on the sets and attributes involved. The models could be presented to the statistical user for selection; or if there are too many closely related models, they could be summarized automatically for the user.

Analysis scripts or logs, the record of all the actions an analyst did, are valuable to store when analysts do complex activities [7,11]. However, these scripts can take much space, must be filtered for errors, require a good deal of expertise to use, and are hard to index and find redundancies among. We prefer in this paper to investigate humbler, smaller, and more flexible units of statistical analysis, records of models tested on the data. If script files are still desired, they can be constructed mainly as lists of pointers to these model analysis results.

There seem to be four potential applications of the idea of retrieving previously stored regression models for analysis of a same database. The first (and main subject of this paper) is to a "statistician's assistant" that helps statisticians doing intensive analysis of a database. If different people work on the same database or at different times, they may unwittingly duplicate effort if records are not kept of what has been done so far. People tend to underestimate the probability of encountering related things, as illustrated by the familiar problem of two people in a group having the same birthday. But even when a statistician has found a regression model on a similar set, their work is not necessarily done; variables may need to be excluded or additional variables included, and additional transformations of variables may need to be introduced or additional functional combinations. Knowing the similar set's model greatly simplifies the search for the new model since it provides a starting point. There are 2^k linear regression models on improper subsets of only k variables, so it can require a good deal of effort to do a blind search for the best one [15]. Nonlinear models require even more searching. So prior knowledge of models can help.

Stored regression models secondly provide an archive of statistical results. This is important when a series of analyses is interrupted and later resumed, as a major way of preserving the state of the analysis. Models can provide a way of communication between research groups. Such models can contain only standard formulae and thus can be preferred to scripts containing institutional and training biases in analysis order.

A third application is to partly or fully automatic analysis of data, perhaps in the manner of the pioneering work of [4]. If a computer is to do simple statistical analyses on its own, perhaps to find anomalies to bring to the attention of a statistician, it must have some way of choosing a model to test the data against. A good choice would be a model that worked well on a related set.

A fourth application is to "antisampling" techniques for estimating values of statistics on a database [20]. Antisampling is like an opposite of sampling, obtaining its estimates from reasoning about previously computed statistics rather than samples of the data; it can be much faster than sampling when data is kept in secondary storage. Knowledge of regressions on related sets can be very helpful in making estimates on some set, providing vital correlational information about attributes. For instance, summary tables prepared by a statistical agency dealing with one attribute can be implicitly extended to correlated attributes, given a description of the correlation, without the need to publish the (mostly) redundant data, thereby extending their coverage for a fixed number of book or computer pages.

We will discuss only regression models in this paper because of their relative simplicity and ubiquitousness in data analysis [13,16]. Some recent work [18] has made important progress on identifying additional, intuitive knowledge statisticians use in doing regressions, as opposed to what is recorded about regressions in textbooks. By including this information along with regression parameters, we believe we can provide a comprehensive representation of regression models.

2. Storage and retrieval of regression models

We first consider what is necessary to store and access regression models on ordinal data, in a way analogous to what [3] did for simulation models. We first consider physical design, since performance efficiency is important, and a variety of work has already addressed the closely related problem of conceptual schema design for statistical databases [8,24].

2.1. Information necessary about a regression

Figure 1 summarizes what we propose to keep about a regression in secondary storage. We assume that in computing the regression, basic statistics on the set and attributes are also found that are necessary for the regression calculations. These include the size of the set and its means, standard deviations, maxima, and minima on the attributes involved. That is, we have simple "database abstract" information about sets, as in our work on antisampling [20]. We could also include a code describing the type of distribution on each attribute, if it is not Gaussian (a reasonable default), and this may be useful if there are many different kinds of distributions present in the data. However, this information is partly suggested by the mean, standard deviation, maximum, and minimum, and may thus be redundant.

Normal regression description:
1. size of set on which the regression was done
2. for each numeric attribute:
a. mean for the set
b. standard deviation for the set
c. maximum for the set
d. minimum for the set
e. type of data distribution (uniform, Gaussian, Poisson, etc.)
3. for each regression on the set:
a. a formula describing the fit curve
b. the mean of the residuals
c. the standard error of the residuals
d. the maximum positive residual
e. the minimum negative residual
4. A list of data items that are exceptions to the model for this set (and are thus omitted in the above calculations)

Alternatively, a note that a disqualification condition applies may be stored instead for the set or a regression. The disqualification flags are:

a. the set contains too few items to analyze
b. data values are missing for some items
c. weights on some data items are negative
d. the dependent variable has two values or one value
e. the distribution of a variable involved in the regression is "extremely" skewed
f. the residuals show systematic trend with order of consideration
g. the residuals show some other systematic trend
h. one or more points are unduly influential (their omission affects the best regression fit significantly)
i. nonstatistical (semantic) disqualification
j. the fit is too poor

Figure 1: Information to be stored about the regressions on a set

---

For purposes of this paper a regression is any formula for approximating the value of some numeric attribute of the data from values of other numeric attributes of the data, which can be represented as a string of codes; a separate in-core lexicon can define the codes. For instance, if we are interested in the relationship between the age of a person and their annual income we could store a regression formula of a function of one variable approximating the curve of income versus age, and separately store codes indicating global names for the age and income attributes. There need only be one code for each numeric attribute, one for each arithmetic operation, plus parentheses, for a total of only a few bits. (Lagged correlations as in [4] can be put into this form by defining new attributes whose values are time-lagged values of other attributes.)

To describe the fit of each regression, we must store some characterization of the distribution of the residuals. Assuming this is a good regression (we want to record poor regressions too, but not their details--see below), the residuals will often not show any systematic tendencies, and should have a Gaussian distribution. We can then characterize them by a mean and standard derivation (the mean deviation is zero for least-squares fitting, but we wish to handle many different fits, especially robust ones). With finite sets we can also calculate absolute fits, and some of the inferences we discuss involve these bounds, so we will assume they are stored too.

Some outliers may be difficult to fit to a regression model. So it is important to allow listing of arbitrary "exceptions" to a model, as a list of item codes. The codes can be defined in an "exception table", so unusual items that frequently must be taken as exceptions to models need only be coded once. For instance, a regression curve for income versus age would be greatly distorted by including millionaires, so these could be removed from a set of interests and noted as specific exceptions, then stored in separate exception files.

Logical criteria necessary for a regression to make sense [2], many of which are exploited in the REX expert system [18] for help in doing regressions, can also be stored. They also are listed in Figure 1. When one of these criteria does not hold, we should store a code for it instead of a regression model. A regression can also be nonmeaningful for non-statistical reasons--understanding of the real world represented by the data may say that certain attributes can't possibly be related, and thus any correlation found is spurious. In general, "negative information" that certain regressions won't work for certain sets and attributes is almost as useful for analysis guidance as positive information.

The statistical analysis system S [1] includes some additional information in their regression summaries, information that might also be included in archived models. But the information in Figure is all that is necessary to do effective retrieval of models.

Since previously-developed regression models need not be examined very often, if there are many such models (the situation we wish to address), they should be put in the secondary or tertiary storage. As we will see, the storage required for a model is not large, and the information about regressions on the same set should usually fit on the same storage page. So we assume the time to retrieve all regression models on some set will be about the time to retrieve a page.

2.2. Record format

We have then two kinds of regression data: information specific to a set and information specific to a regression. The former can be stored as a header [25, section 3-6], for all regressions on a set; it can be a set size followed by a list of quintuples of the form <attribute, mean, standard deviation, minimum, maximum>, one for each attribute in a regression. The header can then be followed by a chain of pointers (within the page most of the time) to regression records, each containing a coded string for the regression formula (with a special code to mark its end), the mean error, the standard error, the largest (signed) residual, the smallest, and a bit indicating whether any disqualification conditions hold for this regression. If that bit is on, it is followed by codes for disqualification conditions; if it is off, it is followed by a codes for exceptions (outliers) to this model. We expect this information will typically be 20 bytes per regression. But a variety of compression techniques can further decrease this space [25, section 14-3], as well as the powerful "lack-of-knowledge" inference we discuss in section 3.3. We postpone until Section 5 discussion of an access scheme for regression records.

3. Supporting inference on regression models

Inference is the systematic derivation of knowledge from other knowledge. To help, a system for management of regression models can use inference to find matches to models for similar situations to those under study, to give a starting model for analysis. A good starting model can save a statistician much time, especially a complicated nonlinear model. Effective inference can also save storage space by eliminating the need to save similar models.

We propose to retrieve such similar models by inference automatically whenever a statistician user begins analysis of a data set. Whenever the user specifies a data set, an independent process can retrieve related stored analyses, in parallel with the statistician user's analytical operations on their set. Since usually the statistician must do analytical thinking during analysis, there is usually free time available for retrievals from secondary or tertiary storage. Then the retrievals can be succinctly summarized for the statistician, allowing querying of further information about them.

3.1. The kinds of inheritance

Inheritance is perhaps the most important kind of inference in artificial intelligence [23, ch.4], and an important part of "common-sense" knowledge that humans possess. It is important in database design and in new approaches to software engineering [5]. This is most helpful for nonlinear regression models, since there are so many of them possible and a good starting guess can greatly simplify the search for a good one. Unfortunately when statistics are concerned, inheritance is rarely obligatory [19], but the phenomena that occur are so similar that we feel it fair to use the same term.

Suppose we have found and validated some regression for some attributes on some set of data. Without reasons to the contrary, we would expect the same regression to hold on a subset. This is "downward inheritance" in the literature of knowledge representation in artificial intelligence, the commonest inheritance direction. It makes sense for regression models, since if you found a strong correlation with few outliers on a superset, no new outliers are going to appear within the set, though the correlation may get a bit weaker if the set items are not representative of the superset items. Downward inheritance is particularly good for absolute bounds on residuals, since a maximum residual for a superset is an upper bound on the maximum residual for the same regression fit on the set. Downward inheritance is also the main idea of analysis of variance and covariance. As an example, if we know a good nonlinear regression formula relating income to age for residents of Santa Clara County, California, then that formula is a good starting point for a regression model for programmers in that county.

But analysts may not be accommodating enough to analyze only large sets on a database. They may instead examine small sets or samples, to check trends; then if they find a strong correlation, check if it holds in some superset containing the set. This is "upward inheritance". Upward-inheritance inferences are not as good as downward inferences because one tries to reason about the tendencies of a batch of unknown data items.

However, it does provide a useful starting point for analysis. An example would be inferring an income-age regression for Santa Clara County residents from the results of a detailed study on programmers in that county.

A third kind of inheritance happens between "sibling" sets, subsets of some single "parent" set, and is called "lateral inheritance" [6]. Important examples are sets representing partitions of something in time or space. For instance, we may use a regression model for last week's data for this week's; or if we may use a model for a neighboring geographical location for this location. Guarantees are rare with this sort of inheritance, as opposed to downward and upward. An example would be applying a regression model for Santa Clara County programmers to Santa Cruz programmers since the counties are adjacent.

A fourth kind of inheritance is a combination of downward and lateral inheritance that improves on both, what we call "diagonal inheritance" [19]. If we know some property of a superset and some but not all sibling sets, we may be able to "subtract out" the properties of the known siblings from the property of the superset to get a better idea about the other siblings. For instance, if the residuals of one sibling on a model that fitted a superset well (that is, the residuals of that model on the superset were normal) were themselves non-normal in some way, then we expect some other sibling to be non-normal in the other direction to "cancel out" the non-normality in building the superset. As an example, suppose we have U.S. Census information about the regression between income and age for Santa Clara County residents, and we have conducted a study ourselves of programmers in that county, and we discover that programmer salaries are higher on the average for each age. Then it is reasonable to infer that the regression model for non-programmer service workers in the county shows a lower income on the average than the county average for all professions, with the amount of difference inversely proportional to the relative size of non-programmer service personnel to programmer personnel.

A fifth kind of inheritance tries to generalize about three known models simultaneously, what might be called "analogical" inheritance. For instance, suppose we wish to do lateral inheritance from some set A to some set B, A and B defined as disjoint partitions on some attribute. Suppose we have tested models for A, B, and some subset S of A formed by intersecting A with some other set I. Then a good guess for a model on a set R formed by intersecting B with I might bear the same relationship (be analogous) to S as the model for B does to A. So if the model on B is formed by adding or deleting a term to the model for A, the model for R might be formed by adding or deleting the same term to the model for S. For instance, if the model on A is $y=x+2z$, the model on B is $y=x$, and the model on S is $y=x+2z+\sin(t)$, then by analogy a good model on R ought to be $y=x+\sin(t)$. As a more practical example, suppose Santa Clara County workers show a sharp downturn at age 65 on their age versus income regression formula, and a similar downturn is exhibited for programmers in the county although their average incomes remain 3000 dollars higher; and suppose San Diego County residents show a less dramatic downturn at age 65; then it is reasonable to guess that the regression model for San Diego County programmers averages 3000 dollars above San Diego County in general, but shows a less dramatic downturn at age 65. Analogical is a weaker kind of inheritance than the others, and requires searching for three sets having a specific relationship, but permits inference of new models.

In all these inheritance types, specific exceptions to the inheritance may be given. This has been a key point in artificial-intelligence work [14]. For instance, it is a good idea to exclude professions with highly atypical age-income relationships from summary data for a county, such as people in professional sports.

3.2. The function of inheritance: exact values, estimates, and bounds

What happens under inheritance to our properties of regression models from Figure 1 is summarized in Figure 2. The conventional meaning of inheritance is that values are copied; the disqualification flags do generally inherit this way either upward or downward. For example, if a set is missing data values for some attributes, any superset will too; and if a set has two or fewer distinct values for some attribute, then any subset does too. But with most other properties of regression models, inheritance will mean a way of estimating the property; a "good" estimate is one close to the actual value for numeric properties, and the value most of the time for nonnumeric properties. A standard error can quantify the inheritance strength, as we will discuss in section 4.1. Since we are dealing with finite data sets, we can often provide absolute bounds on a numeric estimate too. These can be found more often than exact values and estimates, are often much easier to compute since they require no distributional assumptions, and can be used to rule out definitely-inappropriate statistical models.

Statistic or flag	Inheritance downward	Inheritance upward	Inheritance laterally	Inheritance diagonally
set size	bound	no	estimate	bound
mean	estimate	estimate	estimate	estimate
stnd. deviation	estimate	estimate	estimate	estimate
maximum	bound	no	no	bound
minimum	bound	no	no	bound
distrib. type	estimate	estimate	estimate	estimate
regression fit	exact	estimate	estimate	exact
mean error	estimate	no	no	estimate
standard error	estimate	no	no	estimate
fit error	no	no	no	no
maximum error	bound	no	no	bound
minimum error	bound	no	no	bound
exceptions	estimate	estimate	no	estimate
too few items?	exact	no	no	exact
missing values?	estimate	exact	no	estimate
neg. weights?	estimate	exact	no	estimate
too few values?	exact	no	no	exact
skewed values?	estimate	estimate	estimate	estimate
resid. order trend?	exact	estimate	no	exact
other resid. trend?	exact	estimate	no	exact
influences?	estimate	estimate	no	estimate
non-statistical disqualification?	exact	no	no	exact
weak fit?	estimate	estimate	estimate	estimate

Key: "exact" means the exact value is copied from one set to another
"estimate" means the value is a good estimate for the second set
"bound" means the value is either an upper or lower bound on the value for the second set.

Figure 2: Characterization of inheritance of the components (parameters and flags) of a regression description

3.3. Intra-set inferences

An inheritance-like phenomenon takes place between different regression models on the same set. For every model, we can derive weaker models on subsets of these attributes by deleting terms from the regression model and increasing the residuals appropriately. This means that models that are subsets of other models are always comparable, and it only makes sense to store the most complex models well-supported by the data. This mechanism might be called "regression-variable upward inheritance" since it is reasoning from a more specific model to a more general model, and we can also have a "regression-variable lateral inheritance" where we guess that the model on some new attributes has a similar (but not identical) form to the model on some old attributes. For instance, if attribute A correlates linearly with attribute B, and we intuitively feel some new attribute C is like attribute A, then we expect that attribute C correlates linearly with attribute B as well. Or if we have a model of income versus age for Santa Clara County, we could guess since income is related to taxes that a similar shape, although perhaps flattened, applies to the regression of taxes versus age.

Another inference important in artificial intelligence is transitivity: if A relates to B, and B relates to C, then A relates to C for certain kinds of relationships. Something like transitivity arises for regression models when two models on different variables apply to the same set. Then sometimes a third model can be inferred, by doing functional composition of the two models; various formulae can be used to estimate parameters of the new models. For instance, if the regression formula for income versus age for Santa Clara County programmers is I=f A), and the regression of income versus taxes for American citizens is $T = a + bI + cI^2$, a good estimate of the formula for taxes versus age is $T = a + bf(A) + cf^2(A)$.

3.4. The lack-of-knowledge inference

Storing all the information of section 2 for every possible regression done may be burdensome. Even if it is kept in secondary storage with 20 bytes per regression, we may be pressed for storage when regressions are done in part automatically by the computer. We can use inheritance to help save space, with a trick often used by humans and important in many artificial-intelligence programs: the lack-of-knowledge inference [10]. People usually store facts that are in some sense "surprising" or unpredictable. That way, when a person is trying to retrieve a fact about something they know they experienced, if they can't find (remember) it, the fact must be unsurprising. When something is unsurprising, its properties can be inferred by analogies to similar things.

So we can omit information for regressions when approximately such information inherits from another regression, and expect use of the inherited value as a default. (We must add something like a bit string for each regression indicating missing items.) We expect this will be possible often once a large number of regression models have been accumulated, and it is particularly good for information that doesn't usually change much between set and subset, like the mean of the data values on some attribute. However, there are many kinds of inheritance, some of which may conflict in their recommendations, so we will restrict the lack-of-knowledge inference to downward inheritance from a uniquely-defined parent. The parent could be for every set just the data universe, the set of all data items in the database; or for sets defined as the intersection (conjunction) of conditions, we can inherit from the set that is smallest of the sets intersected. For instance, if no regression model for income versus age for Santa Clara County is stored in a statistical-analysis database whereas other regression models are stored for that same set, we could assume that the best model is just the one for all Californians, the only simple immediate superset. On the other hand, if there is no such regression model for Santa Clara County programmers, we can inspect statistics on programmers as well as on Santa Clara County, and use the model on the smaller of the two sets if both have models, or the smallest superset if neither have models.

3.5. In summary

Different kinds of inference methods may be used to infer regression models and their parameters from indirect evidence. This is important for regression-model management because it means that not as much information need be stored explicitly, and what information that remains can be more "intelligent" and irredundant, a better "database abstract" [20].

4. Quantifying inheritance of regression models

Inheritances can be quantified, and this is important to their utility. The strength of the inheritance inference can be quantified, or numeric parameters inherited can themselves be quantified. These quantizations assume that sets are like random samples with respect to one another. This may not be the case, but it does provide a "reasonable guess" regression model for some new statistical study, and there is no easy approach otherwise.

4.1. Assessing the strength of an inheritance

Exact-values and bounds derived from inheritance cannot be argued, but estimates, the most common type of statistical inheritance, can be roughly quantified as to our certainty about the value derived. From a statistical point of view this means finding standard errors of estimates in using statistics of related sets. If we can approximate the relationship between target and related set as a sampling process in either direction, we can use sampling theory for this.

Let us first consider downward inheritance, the most generally useful kind. As we mentioned in section 3.3, an inheritance is stronger the closer in size the two sets are. This is explained by sampling theory for cases where the set is a random sample with respect to the superset [9, p.24], when the mean of the set is a mean of the superset with a standard error of $\tau = \sigma\sqrt{(1/n) - (1/N)}$ where n is the size of the set, N the size of the superset, and σ the standard deviation of the set for the attribute on which the mean is computed. If n is not known, an estimate with a log-linear model may do (that is, you multiply ratios of set size to universe size, to get the ratio of the intersection to the universe size), or a more detailed model. Or absolute bounds on the target set size can be derived by a variety of means as discussed in [21].

Sampling theory also says the standard deviation of the set will be approximately the standard deviation of the superset, when n is not too small. Analogously, we expect the standard error of the regression fit for a set (item 3c in Figure 1) to be approximately the standard error of the fit for the same formula on the superset if n is not too small. (And we expect the vertical offset of the regression fit of a set (item 3b in Figure 1), if nonzero, to be the offset of the superset, with the same error.) For instance, if there are 50,000 programmers in Santa Clara County and 1,000,000 programmers in the United States, and the age-income formula for the United States programmers has a standard error of 5000 dollars, the additional error introduced in going from the superset to the set is approximately σ/222, which is insignificant when added in a sum of squares to the formula fit σ for the intrinsic error of the regression formula itself.

Errors in upward inheritance are identical to those for downward, though we need an additional assumption that the distribution of the superset and the set are the same. If not, we can still reach conclusions by reasoning about samples of the known superset distribution. For instance, if we have only done a survey of

Santa Clara County programmers, it is premature to think that the regression models we found apply to programmers in general; but if we found near-identical formulas for twenty randomly chosen counties in the United States, then that is good supporting evidence that the average of those formulae applies to programmers in the United States.

Lateral inheritance can be modeled as the cascade of an upward and downward inheritance, with total variance the sum of the variances. Or we may have statistics on the average amount by which regression parameters differed among sibling sets. If we know from our understanding of the real world modeled by the data that certain classes of siblings should show more inter-sibling similarity than others, we should tabulate separately for each class. For instance, for a weather records database we may believe that sets representing neighboring geographical localities on the same day will differ less than sets representing a series of days at the same locality.

Diagonal inheritance can occur when a superset and a set have near-identical models, and infers that that same model applies to a different subset of the superset. The reasoning and error are like downward inheritance, except when the known sibling set is large and its characteristics are either very close to that of the superset or very different from the superset. If they are very close, that increases the uncertainty about the model for the unknown set, since the known set cannot generally explain the standard error of the superset. On the other hand, if the known set is atypical of its superset, that should increase our level of confidence that the model for the superset fits well as a model for the unknown set, since something more typical must counterbalance the atypical behavior of the known set.

Analogical inheritance is hard to quantify, but in any event it should be weaker than the other kinds. It can also provide negative evidence for the other inheritances, when they would make a poor analogy to the inheritance of a closely related pair.

Some our our regression-record information is nonnumeric: the flags. To quantify them, though it is not usually necessary, we have no choice but to use certainty factors or probabilities. Much work in artificial intelligence addresses this [23, chapter 8] and many of its techniques can be borrowed. For downward inheritance we can, for instance, ask humans to give subjective likelihoods that some property of a set will be held by a subset, and then code in a table the average of these curves over a number of humans. Or we can tabulate for random set-superset pairs how often inheritance holds for some particular flag.

4.2. Assessing the strength of a transitivity

Situations suitable for transitivity inferences when two distinct overlapping models are given for the same set directly are not likely. Transitivity is more likely to occur with inheritance of one or both of the models. Calculation of the fit of the composed model is a classic problem in composition of functions of random variables. We do not discuss the details of this here because it is highly specific to the functions involved. As for the nonnumeric regression information in transitivity, it is very simple: no model with a disqualification condition can be transitively composed with another model.

4.3. Multiple inheritance

Multiple inheritance is the problem of deciding among conflicting values provided by inheritance; Figure 3 gives an example. A good manager of regression models should be able to suggest the most appropriate model for some new set and some of its attributes given such conflicting advice. There is no general-purpose theory for multiple inheritance in artificial intelligence, but a variety of special-case solutions can be used, in the manner of [12]. The abovementioned quantifications of inheritance weight influence.

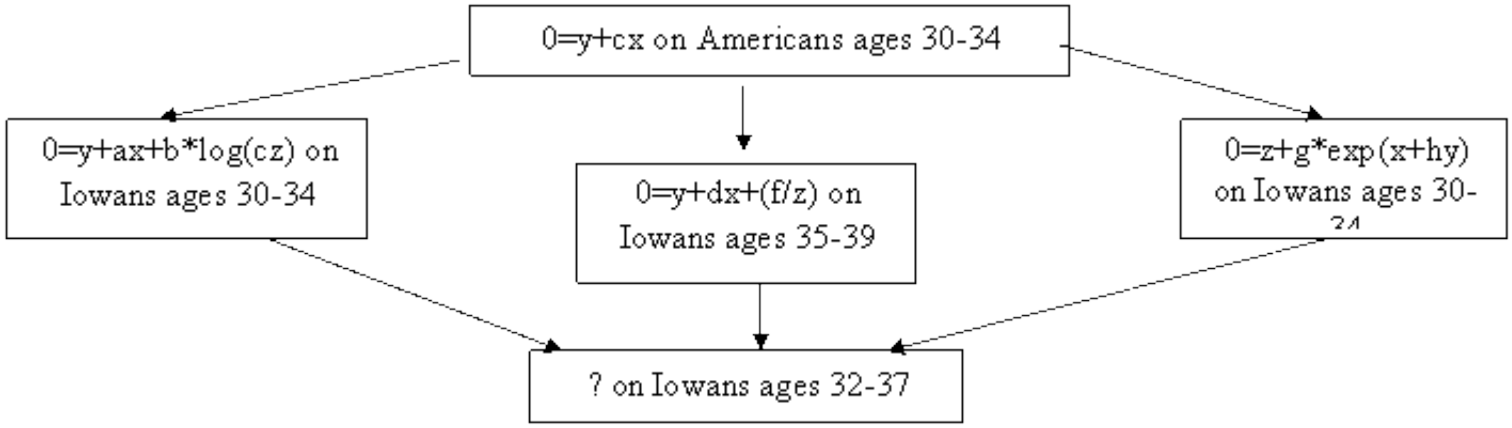
0=y+cx on Americans ages 30-34
0=y+ax+b*log(cz) on Iowans ages 30-34
0=y+dx+(f/z) on Iowans ages 35-39
0=z+g*exp(x+hy) on Iowans ages 30-34
? on Iowans ages 32-37

Figure 3: Multiple inheritance example.

Analysis of variance and analysis of covariance are classic techniques for addressing some of this problem. Given a model (not necessarily a regression) for a set, they provide a way to create submodels for subsets, where submodels generally differ only in an offset term added or multiplied into the general model. This is a limited form of downward model inference. We would like to allow a more general class of inferences, however, because analysis of data over a long period of time, or uncoordinated multi-person analysis, can provide a much richer set of models than one-time analysis of variance and covariance could provide. But necessarily our methods will be more intuitive and heuristic than the latter.

4.3.1. Methods for multiple inheritance of significantly-different models

If regression formulae (item 3a in Figure 1) on the closely related sets are significantly different, there are three approaches. First, some of the formulae may form a "family" in which some are simpler versions of others created by deleting variables. This can happen for sets in which the influence of the deleted variables on the model is small. We can then use the most complex model (or one of the most complex, if there are more than one) for the set we need a model for. For instance, to infer a regression model for Santa Clara County programmers, we can look at the models for Santa Clara County residents and for programmers in general; if the forms of the curves are similar except that there aren't many old programmers and the income formula does not show a downturn at age 65 for them, then that is like a missing variable, and we should use the curve form for Santa Clara County residents as our starting point, not the curve for programmers.

This sounds easy, but there is one problem: the models may require rearrangement to match. For instance, the models $y = ae^{bx} + c\sin(dz)$ and $x = g\log(hy)$ form a family more easily recognized when the second is rewritten as $y = (1/h)e^{(x/g)}$. (The fit of the rearranged formula can often be approximated from the original fit and other information.) Standard symbolic-algebra systems will do this, but we do not need their full power. We can just identify for every possible regression formula (and they are usually not too complicated), and for every possible variable, formula manipulations that will help make that variable the dependent one. Trial and error can be used when more than one manipulation applies.

But if the most complex model is not unique, or the models do not form a "family", then a "weighted average" of them may be a reasonable result of multiple inheritance. Assuming independence, we could take a weighted average of the functions defining the fits, with weights the reciprocals of the expected estimate error, computed by the second formula in section 4.1. Better yet, we can model each set as a Gaussian probability distribution about its fit curve with the specified distributions of the variables (item 2e in Figure 1), weight each, and take the product of the distribution heights, to get an unnormalized distribution representing the expected distribution of the result; the best regression fit can be found to it. This can be done if the best regression curves for Santa Clara County and for programmers are of different shapes below age 65; then if there are 500,000 Santa Clara County residents, and 1,000,000 programmers, then the Santa Clara model should be weighted approximately 1.4 times as much as the programmer model.

The above methods for multiple inheritance require independence assumptions that may be invalid. A third alternative is computing absolute bounds on the regression variables over all possible distributions, after removing exceptional cases, and reasoning about the overlap of those bounds. Items 3d and 3e in Figure 1, the maximum and minimum residuals, define a region within which (non-exceptional) data points must lie. We can also know bounds on the variables from univariate attribute statistics (items 2c and 2d in Figure 1). Previous work with univariate bounds alone [21,22] has convinced us of their power. For instance, income versus age for Federal employees does not always keep pace with that for other employees; so by superimposing the absolute-bounded regions for Federal employees in Santa Clara County and programmers in Santa Clara County we may be able to get a tighter absolute bound on a model of income versus age for Federal Santa Clara County programmers, by fitting a curve through the center of this narrow region.

If the target set is the intersection of other sets about which we know regression information on the same variables, the feasible region for the intersection set is the intersection of the feasible regions for the intersected sets. This may give a narrower region, through which a regression formulae can be more easily fitted. This works best when the regression formulae are greatly different for each of the intersected sets, so it is more robust than the methods of the last section. We may even see that the intersection region is null, in which case we can say the intersection of the sets must be empty without needing to retrieve any data. Even if the intersection region is nonnull, it may be small enough that histogram information about the number of values lying in certain ranges for the entire database could be used to decide if there are too few points in the region to make it worth studying. For instance, if we know at a particular company that there are no programmers over 40, then our regression model need only truncate the model for programmers in Santa Clara County at age 40.

We can further prune an intersection region if we know other "overlapping" regressions (those with a common attribute) on the same intersected sets. For instance, if our interest is in the regression of Y on X for some set intersection, then if we also know a regression of Z on X and bounds on Z, we can infer bounds on X from the second regression that may be tighter than those from the first regression. For the example of Santa Clara County programmers, let Y be income, X be age, and Z can be taxes paid. In general, we can set up a nonlinear programming problem on variables representing values of attributes, and iteratively solve using standard techniques for feasible ranges for each variable. This becomes a linear programming problem when the regression functions are all linear.

4.3.2. Multiple inheritance for nonnumeric attributes

So far we have discussed only multiple inheritance of numeric quantities. For flags on intersected sets, we can quantify confidence in an inheritance by a three-valued logic with values "exact value", "estimate", and "unknown". If any of the values inherited is exact, the result is exact; otherwise if any of the values inherited is an estimate, the result is an estimate; otherwise the result is unknown. For instance, if the residuals of a regression fit on programmers show a systematic trend with age, then the residuals on Santa Clara County programmers will also show a systematic trend, regardless of the existence or nonexistence of a systematic trend for Santa Clara County in general; and if programmers have highly skewed values of age, while Santa Clara County residents do not show any skewness, then Santa Clara County programmers should show skewed values of age. The type of distribution (item 2e in Figure 1) inherits only as an estimate, and is handled by the methods in section 4.3.1.

As for exceptions to a model (item 4 in Figure 1), they do not inherit in the sense we have been discussing. But we can infer things about them. Assuming each exception is represented by a code unique within the entire database, exceptions to the models of two sets intersected are likely only drawn from items in the intersection of their exception lists. For instance, a few programmers in small companies may be millionaires and need to be removed as exceptions to age-income regression before it is done; other millionaires in Santa Clara County can also be identified; then the exceptional programmers in Santa Clara County are almost surely in the intersection of the two exception sets. But the best regression model on the intersection set may be different, perhaps now explaining the previous exceptions; or model may be the same, but exceptions may now be unexceptional due to the larger standard deviation in the intersection set. Similarly, the exception list of the union of sets is likely the union of the exception lists of the sets.

5. An index data structure for inference

Related models can require much work to find, and the choice of the index data structures is critical. A single index on set (say, ordered by the alphanumeric sorting of the conjunctive normal form of the set description in propositional logic) is not efficient, because related sets will only accidentally appear near one another, so usually the whole index must be searched to find all the sets similar to a given set. Hashing of models is not likely to work for the same reason.

We propose instead multiple partial indexes. This is predicated on the observation that most statistical studies restrict only a fraction of the attributes (variables) in the database, and that within a single study, only a few attribute restrictions are varied after the initial selection of a data universe. This suggests that the attributes restricted in the set modeled, as well as the attributes in the model, should be the access keys for regression models. So we propose for each attribute to store, in an array, pointers to the regression models for sets restricting values on that attribute, and separately in another array, pointers to all models using the attribute in their formula. (We could also keep arrays for all models referencing a particular statistic (e.g. median), but those arrays would typically be much larger.) Every model indexed for a set should be retrieved.

Figure 4 gives an example. The five upper boxes represent stored regression models previously studied. Five partial indexes are shown below: two on set-restricting attributes, and three on attributes referred to in models. Each index is an array of model addresses, sorted in increasing order. As an example use, if a statistician asks for the system to infer or inherit a model of X versus Y on Iowans ages 20-29, then all the indexes are invoked except for the one on Z.

Model	Location	Ages	Address
y=2.5*log(x)	Iowans	30-34	1
y=2.4*x+1.8*z	Americans	35-39	2
z=-1.5*x	Iowans	any	3
z=-1.9*x	Americans	any	4
y=3*log(x)+1	Americans	20-29	5

State restriction: 1, 3
Age restriction: 1, 2, 5
Variable x in mode: 1, 2, 3, 4, 5
Variable y in model: 1, 2, 5
Variable z in model: 2, 3, 4

Figure 4: Example of five regression models.

If some array is too big, we can just omit it (arrays just facilitate access, and are not required). Or we can subdivide it and index the subparts. For instance, if many regression models correspond to sets that restrict the age attribute, then we can subdivide the age array into subarrays for age ranges whose lower bound is less than 40 and for ranges whose lower bound is at least 40. Subarrays can be recursively subdivided, and balance maintained in the manner of B-trees. (Hierarchies of nonnumeric-attribute values can be defined by type hierarchies; hierarchies of sets defined on range-restricted attributes can be defined by diagonalization mappings.) There is a tradeoff between the size and complexity of the index and the length of the arrays, which can be analyzed and optimized mathematically. Hopefully, with the many kinds of arrays, each will be short.

To find the regression models most closely related to a given set, we first identify the attributes on which that set restricts, and the attributes necessary (not just desirable) in the model. We use this to identify the relevant arrays and subarrays, using subarray indexes as necessary. We then intersect the corresponding pointer sets to stored regressions, which is straightforward to do in one pass through every relevant array if they are kept sorted. Any models in the intersection are high-potential, and we should retrieve and examine them carefully to quantify their closeness (section 4.1 explains how), and do multiple inheritance if there are more than one. If the intersection is empty, we can intersect of all but one pointer set; or all but two, and so on, until we find at least one model that qualifies. For the previous example of a model on X versus Y for Iowans ages 20-29 in Figure 4, we can first intersect the first four index arrays to get model numeral one; if we are not satisfied with this model and want additional help in constructing a new model, we can find items that belong to three of the four arrays, which then gives us models 2 and 5 to additionally explore. Of these three models, information from the first can inherit to the model we want to construct by lateral inheritance and from model

5 by downward (a stronger) inheritance, but model 2 provides no help, as can be seen as soon as the exact restrictions defining its set are examined.

6. Conclusion

As databases continue to increase in size, the possibilities for statistical analysis become seemingly endless. Database support is necessary to handle the associated bookkeeping. Inference is a key to making this possible, providing a way to find and use similar previous analyses. We have proposed a general such approach for an important class of analyses, the building and testing of regression models.

7. References

[1] R. A. Becker and J. M. Chambers. S--An interactive environment for data analysis and graphics. Wadsworth, Belmont, CA, 1984.

[2] D. Belsey, E. Kuh, and R. E. Welsch. *Regression diagnostics: identifying influential data and sources of collinearity.* Wiley, New York, 1980.

[3] R. W. Blanning. Issues in the design of relational model management systems. Proceedings of the National Computer Conference, 1983, 395-401.

[4] R. L. Blum. *Discovery and Representation of Causal Relationships From a Large Time-Oriented Clinical Database: The RX Project.* Lecture Notes in Medical Informatics, volume 19, Springer-Verlag, New York, 1982.

[5] A. Borgida, J. Mylopoulos, and H. K. T. Wong. Generalization/Specialization as a basis for software specification. In *On Conceptual Modeling: Perspectives From Artificial Intelligence, Databases, and Programming Languages,* M. L. Brodie, J. Mylopoulos, and J. W. Schmidt, eds. Brodie, M. L., Mylopoulos, J., and Schmidt, J. W., eds., Springer-Verlag, New York, 1984, 87-117.

[6] J. G. Carbonell. Default reasoning and inheritance mechanisms on type hierarchies. Proceedings of the Workshop on Data Abstraction, Databases, and Conceptual Modeling, Pingree Park, Co., June 1980, 107-109.

[7] D. B. Carr, P. J. Cowley, M. A. Whiting, and W. L. Nicholson. Organizational tools for data analysis experiments. Proceedings of the Statistical Computing Section, American Statistical Association, 1984.

[8] P. Chan and A. Shoshani. SUBJECT: a directory driven system for organizing and accessing large statistical databases. Proceedings of the Seventh International Conference on Very Large Databases, Cannes France, 1981, 553-563.

[9] W. G. Cochran. *Sampling Techniques, third edition.* Wiley, New York, 1977.

[10] A. Collins, E. H. Warnock, N. Aiello, and M. L. Miller. Reasoning from incomplete knowledge. In *Representation and Understanding: Studies in Cognitive Science,* D. G. Bobrow and A. Collins, eds., Academic Press, New York, 1975, 383-415.

[11] Cowley, P. J., Carr, D. B., and Nicholson, W. L. Experiences with a data management analysis prototype. In *Computer science and statistics: Proceedings of 18th symposium on the interface,* Fort Collins, Colorado, (March 1984), 147-150.

[12] G. Curry, L. Baer, D. Lipkie, and B. Lee. Traits: an approach to multiple-inheritance subclassing. Proceedings of the SIGOA Conference on Office Information Systems, Philadelphia Pa., June 1982, 1-9.

[13] W. A. Draper and H. Smith. *Applied Regression Analysis, 2nd edition.* Wiley, New York, 1981.

[14] S. E. Fahlman, D. S. Touretzky, and W. van Roggen. Cancellation in a parallel semantic network. Proceedings of the Seventh International Joint Conference on Artificial Intelligence, Vancouver, B.C., August 1981, 257-263.

[15] Furnavil, G. N. and Wilson, R. W. Regressions by leaps and bounds. *Technometrics, 16,* 4 (November 1974), 499-511.

[16] F. Mosteller and J. W. Tukey. *Data Analysis and Regression.* Addison-Wesley, Reading Mass., 1977.

[17] Oldford, R. W. and Peters, S. C. Statistically sophisticated software and DINDE. In *Computer science and statistics: Proceedings of the 18th symposium on the interface,* Fort Collins, Colorado (March 1986), 160-167.

[18] D. Pregibon and W. Gale. REX: An Expert System for Regression Analysis. Proceedings of the 6th COMPSTAT Symposium, Prague, 1984, 242-248.

[19] N. C. Rowe. Inheritance of statistical properties. Proceedings of the National Conference, American Association for Artificial Intelligence, Pittsburgh Pa., August 1982, 221-224.

[20] N. C. Rowe. Antisampling for estimation: an overview. *IEEE Transactions on Software Engineering, SE-11,* 10 (October 1985), 1081-1091.

[21] N. C. Rowe. Absolute bounds on set intersection and union sizes from distribution information. *EEE Transactions on Software Engineering, SE-14*, 7 (July 1988), 1033-1048.

[22] N. C. Rowe. Absolute bounds on the mean and standard deviation of transformed data for constant-derivative transformations. *SIAM Journal of Scientific and Statistical Computing, 9*, 6 (November 1988), 1098-1113.

[23] N. C. Rowe. *Artificial Intelligence through Prolog.* Englewood Cliffs, N.J.: Prentice-Hall, 1988.

[24] S. Y. W. Su. SAM*: a semantic association model for corporate and scientific-statistical databases. *Information Sciences, 29,* 1983, 151-199.

[25] G. Wiederhold. *Database Design, 2nd edition.* McGraw-Hill, New York, 1983.

Go up to paper index