A Data Mining Tool For Producing Characteristic Classifications in the Legal Domain

Stephen Lawrence Dale and Trevor Bench-Capon
LIAL - Legal Informatics At Liverpool,
Department of Computer Science,
The University of Liverpool,
Liverpool,
L69 7ZF, UK.

.

Abstract

Data Minining is a relatively new term for the discovery of knowledge in large data sets. Essentially the idea is to extract relationships from a mass of data. In this paper we will suggest that opportunities for data mining exist in the legal domain, and we will begin by suggesting some of the uses to which it might be put. We then briefly discuss some of the constraints that must be placed on a data mining tool in the legal domain. The bulk of the paper is concerned with a particular data mining tool, which is designed to identify the characteristics of typical members of a legal category. We describe the theory of the tool, and discuss some potential pitfalls with respect to its application. We then work through a simple example to illustrate how the tool operates, and briefly present the results from a more complicated example. Finally we draw some conclusions and make some suggestions for future work.

1. Introduction

Many organisations hold large sets of data from which they believe useful information can be extracted. The process of extraction is termed "data mining" or "knowledge discovery". Such large data sets do exist in the field of law, particularly where cases are processed in a routine adminstrative manner, such as many welfare benefits, and we see applications for data mining particularly in the following areas:

1) *Monitoring adherence to guidelines*: in areas such as sentencing, guidelines are issued which are supposed to govern the way in which decisions are made. Since the guidelines may require a number of factors to be weighed, and there may be discretion within the guidelines, it can be difficult to see whether they are being in general complied with. The idea here is to extract the relationships from data relating to such decisions to see to what extent the discovered relationships reflect the guidelines.
2) *Investigation of legal theories*: some jurisprudes attempt to explain legal decisions in a particular area from a particular theoretical standpoint. See for example Edwards (1994). Often support for such views is anecdotal and debatable. Data mining a relevant set of past decisions would allow us to see whether the predicted relationships emerge.
3) *Validation of elicited knowledge*: often the rules elicited from an expert during knowledge acquisition will be incomplete, or possibly even incorrect. Data mining could be used to confirm the existence of the posited rules, or to suggest refinements.
4) *First cut knowledge elicitation*: alternatively data mining could be used to produce an initial set of rules which could then be inspected and refined by an expert. We have placed this - perhaps the most obvious application - last, since we believe that usually there will be prior opinion or theory as to what the relationships are, and this can be used to focus the data mining process.

2. Data Mining in Law

Almost any technique developed for machine learning can be applied to data mining as well. There are, however, some constraints imposed by the legal domain, which make particular demands on the tool.

First the output must be transparent. In some typical data mining applications - identification of likely purchasers of a product, for example - the classification itself is all that matters. In contrast the applications cited above require that we extract the principles used to classify. Some techniques which are successful in other domains, such as neural networks, do not provide the principles used to classify in an easily accessible form.

Second the technique must be resilient to noise. In any legal dataset there will be "rogue" decisions: either because they are genuinely aberrant, or because there were special features of the case which are not fully reflected in the recorded features. It is important that such decisions do not hide the relationships we wish to uncover.

In other words we require our data mining tool to

1) Produce symbolic descriptions
2) Describe characteristic members of the class of interest, so that we focus on the core of the concept without being distracted by special cases.

Limiting ourselves to descriptions of typical members does not harm any of the four applications described in section 1. In none of these are we using the tool to classify individual cases: rather we are trying to find a broad

picture. Thus in monitoring and investigation of legal theories, we have to accept that there will be special cases - what matters is is whether the guidelines are in general followed, and whether the theory correctly identifies the main run of cases. In validation and elicitation, we are using the tool in conjuction with an expert, who is responsible for providing special cases and refining the boundaries of the classification.

In this paper we describe a simple data mining tool which has the characteristics given above. The tool is adapted from the work of Stephan (1996).

3. Theory of the Tool.

In this section we describe the theory underlying the technique we have implemented. We begin by discussing the case of boolean data, where the principles are clearest, and then describe how the technique extends to continous data and enumerated data types.

3.1 Basic Operation

Let us suppose that we have a data base of cases, with their outcomes. Let us suppose also, for the moment, that all the attributes of the cases are booleans. What we are interested in is:

1) whether any combinations of particular values for particular attributes provide necessary conditions for a given outcome;
2) whether any combinations of particular values for particular attributes provide sufficient conditions for a given outcome.

Suppose that it is a necessary condition that attribute A have value V for the outcome O. This means that outcome O is a sufficient condition for attribute A to have value V. Now if we select from the database all records with outcome O, all of them will have value V for attribute A.

Suppose that it is a sufficient condition for outcome O that attribute A have value V. Now if we select all the records from the database with outcome NOT-O, none of them will have value V for attribute A. That is V will be the complement of the values possessed by A in the selection with respect to the values A can take. In the case of a boolean, if A is true is a sufficient condition for outcome O, the value of A for all records with outcome NOT-O will be false.

Note that if the necessary condition is a conjunction of several attributes, all will be identified from the records with outcome O. Similarly, if the sufficient condition is a disjunction of several attributes all will be identified by considering the records with outcome NOT-O.

The picture is more complicated, however, if a necessary condition is formed by a disjunction of two or more attributes, or a sufficient condition is formed by the conjunction of two or more attributes. For example if `A v B` is a necessary condition, `A` may be either true or false in the selection of records with outcome O, with `B` either true or false in case that `A` is true, and with `B` true in case that `A` is false. Similarly with a sufficient condition of `A & B`, `A` will be constrained to false in records with outcome NOT-O only in cases where `B` is true.

If we are to detect these combinations, therefore we need to further partition the data into cases where there is a given outcome, *and* a given value for a particular attribute, and examine what constraints, if any, are placed on the values of the other attributes in these partitions. In the above example where `A v B` is a necessary condition for outcome O, the partition with O and `A` false, will contain only records where `B` is true. Similarly the partition with outcome O and `B` false will contain only records with `A` true.

If we create these additional partitions, fixing the value of each attribute in turn, we will be able to detect any necessary and sufficient conditions formed by pairs of attributes. If we further partition these partitions by fixing the values of a second attribute we can find all necessary conditions formed by a disjunction of three attributes and sufficient conditions formed by a conjunction of three attributes. Repeated partitioning in this way will find necessary and sufficient conditions of any length. This, of course, supposes that we start from a sufficiently large data set to allow the repeated partitioning to generate partitions with enough records to give meaningful results. We will return to this point in 4.3.

3.2 Continuous Values for Attributes

Suppose now that the attributes are not boolean, but continuous values. Here we will be trying to find some range of values for the attribute which contribute to a necessary or sufficient condition for the outcome. Now what we will be interested in is the maximum and minimum values for attributes in particular partitions. Suppose we have an attribute (call it "age") which in the database as a whole can take values from 0 to 100. If in the selection with outcome O it occupies only the range 65 to 100, we can deduce that `age >= 65` is a necessary condition for outcome O. Obviously we will be interested only in cases where the range so produced represents a significant curtailment of the original range. Similar considerations applied to the set of records with outcome NOT-O suggest that we can find ranges which indicate sufficient conditions, by taking the complement of the range of values which appear for the attribute in that partition with respect to the full range it exhibits in the whole database. Repeated partitioning, in the manner described above, will also find combinations of such attributes with other continuous attributes, with booleans and with a mixture of the two.

It is important to note that the range delivered may not contain the exact boundary, since this relies on a record with the precise borderline value being present in the database. It will, however, deliver a *safe* range, in the sense that the range will be too narrow rather than too wide. Since, however, our aim is to detect the characteristics of *typical* members, the potential absence of the borderline cases is perhaps not crucial. Moreover, given a large data set, we may expect the borderline values to be present in practice.

Where we need to partition on a continuous attribute we must select some threshold value to partition on. We will discuss how we might arrive at such a threshold in Section 4.4.

3.3 Enumerated Values for Attributes.

In the case where an attribute can take one of a enumerated set of values, we will be interested in the subset of the possible values which appears in the selection with outcome O and in the complement of the subset which appears in the selection with outcome NOT-O. If either of these subsets are significantly smaller than the original set, the attributes will be interesting. In this case also we can repeatedly partition on such attributes, by selecting a subset of the possible vallues to act as the criterion for partitioning. We will discuss how such partitioning subsets may be chosen in 4.4.

3.4 Other Attributes

Some attributes will have values which are not boolean, do not form a numeric ranges, and are not taken from an enumerated set of reasonable size. Names and addresses are an obvious example. It is unlikely that we will find any meaningful relationships featuring such items using our technique, and so they are best stripped from the data at the outset. This is not to say that such attributes may not have significance: for example, post code is often significant in data mining applications. Our technique is not, however, useful for such attributes. Arguably, such attributes will have less impact in the legal domain than in a marketing application. In some particular cases, however, for example if a legal theory was making use of socio-economic status, of which postcode is a good indicator, the use of our technique might well be inappropriate. As with any statistically based tool, it is important to be aware of any particular features of the problem which may affect the choice of the tool to use.

4 Some problems with the Above Technique.

There are a number of questions that need to be asked about the above technique. In this section we will discuss some of them, and indicate our solutions.

4.1 Computational Complexity.

If we are to partition on every combination of attribute, so that we can find disjunctions forming necessary and conjunctions forming sufficient conditions irrespective of their length, we will need to generate a very large number of partitions. For n attributes the first partitioning will generate 2n partitions, the next 4n(n-1) partitions, and so on. Clearly this is worse than order n factorial, and hence infeasible for large n.

We hypothesise, however, that interesting conditions will be conjunctions/disjunctions of relatively few terms. If this is so, we need not produce every partition, but only say the first m levels, giving all disjunctive/conjunctive combinations of m+1 variables. For this the computation is of order n^m. For all realistic expected values of n (say less than 100) this makes computation feasible for combinations of reasonable size, perhaps up to 6 or 7. Of course if n is small, m can be larger, and if we are prepared to employ more

computation time m can be increased. Diminishing returns are, however, likely to set in, especially if we believe that the the irreducible disjunctions/conjunctions are likely to be relatively short.

4.2 Sample Sizes.

Clearly the method relies on having a representative sample of cases in a given partition: otherwise false positives will proliferate. Suppose that the chance of an attribute having the value true were 0.9: then in a sample of 10, the probability that all examples should be true by chance is 0.35; in a sample of 20, this falls to 0.12; and in a sample of 50, to 0.005. However, even on a sample of 10, if there were an even chance of A being either true or false, the probability of a false positive would be no more than 0.00098. Since we can can discover the relative numbers of trues and falses for a given attribute by examining the database we can determine the the likelihood of false positive for any given sample size. This means that we can accompany our report of a relationship with a measure of our confidence that it is not spurious. If desired we can impose a threshold level of confidence and only report examples which meet a given standard of confidence.

4.3 Partitioning and sample size.

The sample size will be radically affected by the amount of partitioning that has occurred. Obviously each time we partition we reduce the sample size, and some partitions will have too few records to form a meaningful sample, or even no records. This enforces a limit on the partitioning - so that we don't go below a sensible sample size - and provides the effective constraint on the level of partitioning. In practice the partitions will reduce to too small a size before the complexity of repeated partitioning becomes a problem. As we suggested above, however, our belief is that significant relationships will combine relatively few attributes, in which case this limit on partitioning will not result in missing any significant relationships.

4.4 Choice of Thresholds for Continuous and Enumerated Values.

When partitioning on a boolean we have no choice - true will go into one partition and false into the other. If, however, the attribute can take a numeric range, we must decide where the line will be drawn - there is no reason to suppose that the midpoint of the range will be the correct place. However for the purposes of the technique it does not matter if the ranges chosen for partitioning do not cover the whole range the attribute can take: in the case of an attribute in range 0 - 100 with a "correct" threshold of, say, 65, partitioning on 0 to 10 for failure and 90 to 100 for success will serve as well as finding the exact range. Our approach therefore is to adopt conservative thresholds giving small safe ranges as the basis for partitioning, even though this means losing data from the middle of the range. The crucial thing is that the range chosen for partitioning does not include the threshold. The one caveat that must be applied is that the partition must contain enough examples to represent a sample of meaningful size. We therefore apply a range from the smallest/biggest point up/down to a value that will give a sensibly sized partition. This will, of course, miss attributes which should be of interest where the range we are trying to find is a band in the middle of the range of the attribute: it is, however, difficult to see how such a band could be found without prior knowledge, and the technique is probably not appropriate for these cases. Note, however, that in some potential applications, we will have a theory or an expect to guide us. Here a range from the middle may be suggested, and we can use that as our partition.

Similar considerations apply to attributes with an enumerated set of values. Here we should partition on any single value that gives a sensibly sized partition. If we have expert advice, it may be possible to order the enumerated values in some way (e.g. colours could be ordered as in the spectrum). If this is so we can get larger partitions by taking a subset from either end of the order. Provided, however, the dataset is large enough relative to the number of enumerated values, we believe that best results are obtained by partitioning on individual values.

An alternative treatment of attributes with enumerated values is to assign a boolean attribute for each of the potential values. Thus instead of colour withe values {red, green, blue} we have colour-red, colour-green, and colour-blue, and in each record one of these will be true and the other two false. This move is often made when neural networks, and may well be apprpriate for our technique also.

4.5 Noise

The technique as it stands is extremely sensitive to noise. A single aberrant decision, (or a typing error in the database) may cause the apparent range of values that an attribute can take in a given partition to extend across the whole range, thus completely obliterating the significance of that attribute. Clearly, since we have no

assurance that our data is clean and consistent, we need to take account of this. Our approach again is based on that of Stephan (1996).

At every point within the range there will be a certain number of "hits". Let us therefore plot a cumulative frequency diagram from the bottom (B) of the range to the top (T). Let us assume that the "hits" include noise so that the real bottom of the range is some R , where B < R < T. Provided that there are relatively few noise "hits" compared with true "hits", the frequency of hits in B to R will be significantly less than the hits in R-T. Examination of the frequency diagram will therefore show a significant discontinuity at point R, allowing us to regard hits in range B to R as noise. Of course, this makes certain assumptions about the distribution of terms across the range in the database as a whole, but these assumptions can be confirmed by comparing the cumulative frequency diagram with one drawn for the complete database.

This analysis can be performed once an attribute has been identified as being of interest. In order to so identify it, however, we need to consider the partition *without* outliers. Determining the minimum and maximum values for attributes in the partition is therefore performed not on the whole partition, but on the records in the partition which do not have extreme values for the attribute under consideration. The number of disregarded values can be set at a level according to the amount of noise anticipated. Note that the selection of records for consideration is performed with respect to attributes individually: thus an outlying value on one attribute will not prevent a record being considered with respect to its other attributes. Once an attribute is flagged as interesting, this can be confirmed by plotting the cumulative frequency as described above.

By calling the outlying examples "noise". we have implied that they represent mistakes, either in the decision or in the data. But there is another way of looking such examples, which is especially appropriate in the legal domain. It may be that the concept is not appropriately defined by necessary and sufficient conditions at all, but rather there are special cases to which the concept is quite properly applied. Consider basketball players. A typical basketball player is over 6 foot tall, but there are professional players as short as five foot six. Suppose that in our database we have 1000 people, ranging in height from five foot to six foot six of whom 100 are basketball players. Of the basketball players, 98 are above six foot, one is five foot six and one is five foot ten. The technique would identify "height greater than five foot six" as a necessary condition. Thus to get all the basketball players we need a range of twelve inches. But cutting this to eight inches - a reduction of a third - still gives us 99% of the players, and cutting it to six inches - a reduction of 50% - still gives us 98% coverage. The idea here is that there is a trade off between the extent of the range and the coverage the range gives. If a substantial reduction in the range gives only a small loss of coverage, we may consider the reduction useful. At some point the trade off will become adverse, however, and at this point we may consider that we have identified a condition describing a "typical" member. Stephan herself prefers this interpretation of outlying examples, calling the condition so arrived at a "characteristic description". With reference to the legal domain, where open textured concepts abound, we may also prefer to interpret the results in this way, seeing the refined description as a means of homing in on the "core" of the concept.

5. Empirical results

The technique described above has been implemented in C. Essentiially there are three modules: one to partition the data, one to identify the maxima and minima of ranges, and one to produce cumulative frequency ploits for interesting attributes.

5.1 A simple example

To illustrate the technique consider the simple example of retirement age in UK Social Security law. A person is retired if they are a man over 70, a woman over 65, a man over 65 who has elected to retire, or a woman over 60 who has elected to retire. We can produce a data set by generating records with a random number between 0 and 100 for age, a random 0 or 1 to represent sex, and a random 0 or 1 to represent whether they have elected to retire. We call these attributes `age`, `sex` and `elected` respectively. A fourth attribute, `retired`, will be set to 1 if they are considered retired and to 0 if they are not.

We now construct two partitions of the data set, according to the value of the fourth attribute. We now get a report for the partition with retired = 1.

```
Retired = 1
feature     min   max
age          60   100
```

for the partition with retired = 0

```
Retired = 0
feature     min   max
age          0    69
```

Note that `sex` and `elected` are not reported since their range has not been constrained.

This identifies `age` as an interesting attribute: with `age>=60` a necessary condition and `age>69` as a sufficient condition for being retired.

Now we partition each of these partitions on `age>80`, `age<20`, `sex = 0` and `sex=1`, and `elected=0` and `elected=1`. Two partitions will be empty: those with `retired = 1` and `age < 20`, and with `retired = 0` and `age > 80`. Where all the data has gone into one partition, we know that that partition will yield nothing of interest not discovered previously. The remaining reports for `retired = 1` are:

```
Retired = 1 and sex = 0
feature    min  max
age         60    100

Retired = 1 and sex = 1
feature    min  max
age         65   100

Retired = 1 and elected = 1
feature   min  max
age       60   100

Retired = 1 and elected = 0
feature   min  max
age        65   100
```

Of these two show features of interest: in the case of men, `age >= 65` is a necessary condition, and in the case of those who have not elected to retire, `age>65` is again a necessary condition. The other two do not strengthen conditions already found. In the rest of our description such reports will be supressed.

For `retired = 0`, the following reports are produced:

```
Retired = 0 and sex = 0
feature    min  max
age          0  64

Retired = 0 and sex = 1
feature    min  max
age         0    69

Retired = 0 and elected = 0
feature    min  max
age         0   69

Retired = 0 and elected = 1
feature    min  max
 age        0   64
```

In two cases we have identified further sufficient conditions: for a woman, or for someone who has elected to retire, it is a sufficient condition that `age>64`. In the other two cases, again no extra information is given.

At this point we could record the rules governing retirement as a prolog program - using only the sufficient conditions:

```
retired(X):- age(X,A),A > 69.
retired(X):- sex(X,f),age(X,A),
             age > 64.
```

```
retired(X):- elected(X),age(X,A),
             age > 64.
```

Only one sufficient condition is now missing: that where a woman has elected to retire. This can be found by a further level of partitioning, where we will receive the report:

```
Retired = 0 sex 0 elected = 1
feature     min  max
age          59    100
```

This allows us to add the final clause to our Prolog program:

```
retired(X):- sex(X,f),elected(X),
             age(X,A),A > 59.
```

5.2 Adding Noise

We then added noise, extending the data set by giving 6% of the records the incorrect value for retired. Now the reports were not on the basis of the whole partitions, but instead the top and bottom 5% of values were ignored. This safe over-estimate of noise meant that all the noise records were excluded from consideration, and the same reports were generated, except that the maximum values which had been 100 were now 98 and the minimum values which had been 0 were now 2. The booleans were, of course, unaffected. At the first level of partitioning this meant `age` was interesting, and so the cumulative frequency diagram in Figure 1 was produced.

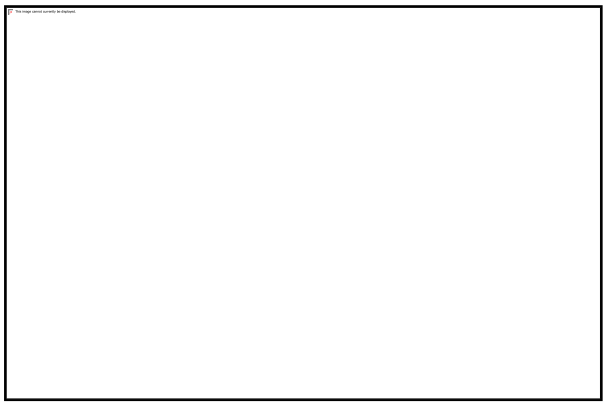

Figure 1

Note that there are in fact three discontinuities, the curve steepening at age = 60 where retired women are counted, and again at age = 65, when all women and men who have elected to retire are included, and again at age = 70, when everyone is included. This is in itself interesting, since it identifies the three significant age points: analysis of the various finer grained partitions eventually produces only a single appropriate discontinuity in each: at age = 60 for retired women, at age 65 for women who have not elected to retire and men who have, and at age = 70 for men who have elected to retire.

5.3 A Second Example

The above simple example illustrates the technique. For a second example we will consider the data set used in Bench-Capon (1993) to examine properties of neural networks.

This example was based on a fictional welfare benefit paid to pensioners to defray expenses for visiting a spouse in hospital. The conditions were:

1) The person should be of pensionable age (60 for a woman, 65 for a man)
2) The person should have paid contributions in four out of the last five relevant contribution years
3) The person should be a spouse of the patient
4) The person should not be absent from the UK

5) The person should have capital resources not amounting to more than £3,000
6) If the relative is an in-patient the hospital should be within a certain distance: if an out-patient, beyond that distance.

These conditions represent a range of typical condition types: (3) and (4) are Boolean necessary conditions, (5) is a threshold on a continuous variable representing a necessary condition, and (2) relates five variables, only four of which need be satisfied, effectively giving rise to a disjunctive necessary condition. (1) and (6) are perhaps the most interesting since the relevance of a variable depends on the value of another: in (1) sex is relevant only for ages between 60 and 65, and in (6) the effect of the distance variable depends on the Boolean saying whether the patient is an in-patient or an out-patient. We have therefore 12 attributes which can contribute to satisfaction of the conditions.

The data sets were generated from a LISP program. The data set consisted of 50% satisfying cases, where the outcome of each condition was generated randomly within the range which would satisfy the relevant condition, and other values generated randomly across the full range. The other cases were generated so that an equal number would specifically fail on each of the conditions, other values being generated at random. A number of "noise" attributes with no effect on the outcome were also included: in the experiments there were 52 such noise attributes, giving a total of 64 input factors.

The first level partition, on whether or not the benefit was payable, gave four necessary conditions:

1) The spouse attribute was constrained to 1;

2) The absence attribute was constrained to 0;

3) Age was constrained to > = 60;

4) Capital was constrained to <= 3000.

The next level of parttioning of those qualifying for benmefit identified the rest of the necessary conditions:

1) Partioning on sex constrained age to >=65 in the case of males;

2) Partioning on in-patient contrained distance to be <50 if true and >= 50 if false;

3) Partitioning on each of the contribution conditions = false, constrained all of the other four contribution conditions to true.

Thus two levels of partitioning successfully identified all the necessary conditions. To obtain the sufficient condition it is necessary for *all* the necessary conditions to be satisfied. This was not discovered, as too deep a partioning was required.

6. Concluding Remarks

In this paper we have described a simple but effective tool for data mining large legal data sets so as to extract symbolic descriptions of the characteristics members of some class of interest.

Of course, what is needed now is for the tool to be evaluated on some real data set., since this is the only way to become convinced that the various assumptions made about distributions and in particular the distribution of noise can be confirmed.

The particular appeal of this technique lies in its treatment of special cases, whether noise, or genuinely special. In law it is often useful to identify the core of a concept, without being misled by the fascination of the hard cases which give rise to leading decisions.

We believe that data mining will become an increasingly important topic in the legal domain, and that the simple technique described may provide an interesting contribution.

References

Edwards, L., (1994) *Modelling Law Using a Feminist Theoretical Perspective*, in Proceedings of the Fourth National Conference on Law, Computers and Artificial Intelligence, EUCLID, Exeter.

Stephan, V., (1996) *Extracting Symbolic Objects From Relational Databases*, in Proceedings of the Seventh International Workshop on Database and Expert Systems Applications, IEEE Computer Society Press, Los Alamitos, CA., pp 514-519.