Journal of Intelligent Information Systems, 9, 57–81 (1997)

Feature Subset Selection within a Simulated Annealing Data Mining Algorithm*

JUSTIN C. W. DEBUSE jcwd@sys.uea.ac.uk
VICTOR J. RAYWARD-SMITH vjrs@sys.uea.ac.uk
Computing Department, School of Information Systems, University of East Anglia, Norwich NR4 7TJ, UK

Abstract. An overview of the principle feature subset selection methods is given. We investigate a number of measures of feature subset quality, using large commercial databases. We develop an entropic measure, based upon the information gain approach used within ID3 and C4.5 to build trees, which is shown to give the best performance over our databases. This measure is used within a simple feature subset selection algorithm and the technique is used to generate subsets of high quality features from the databases. A simulated annealing based data mining technique is presented and applied to the databases. The performance using all features is compared to that achieved using the subset selected by our algorithm. We show that a substantial reduction in the number of features may be achieved together with an improvement in the performance of our data mining system. We also present a modification of the data mining algorithm, which allows it to simultaneously search for promising feature subsets and high quality rules. The effect of varying the generality level of the desired pattern is also investigated.

Keywords: Feature subset selection, data mining, simulated annealing

1. Introduction

Feature subset selection (Devijver and Kittler, 1982, John et al., 1994, Pei et al., 1995) is a data mining enhancement technique which reduces the number of fields in the database to a highly predictive subset. This process may improve the performance of the data mining algorithm to be used, in terms of speed, accuracy and simplicity. In addition to this, the identification of features which do not need to be stored, collected or bought may bring financial savings. $Filter$ feature subset selection occurs as a pre-processing phase, selecting the most powerfully predictive fields before the data mining algorithm to be used is applied to the database. $Wrapper$ feature subset selection techniques (Kohavi and Sommerfield, 1995, Liu and Setiono, 1996a) select fields whilst applying the data mining algorithm to the data. Many feature subset selection algorithms contain two key components:

- A measure of the quality of a set of features. This should take into account some measure of the predictive power of the fields, as well as the size of the feature subset. The measure of predictive power used should correlate closely with the performance of the data mining algorithm to be used.

* This research was sponsored by the Teaching Company Scheme under grant no. GR/K 39875

- A search strategy to find the best feature subset as defined by this measure. If the database contains many fields and/or records then an exhaustive evaluation of all possible feature subsets will be computationally infeasible.

  A search strategy which tests combinations of fields rather than each feature individually will generally give superior results. This is because fields which are not powerfully predictive in isolation may prove to be so when used in combination with others. Conversely, a field which is powerfully predictive in isolation may be inferior to a group of individually weaker features. Such fields may impair the overall accuracy of some data mining algorithms, as demonstrated in (John et al., 1994). ID3 was applied to the problem of learning the Boolean concept $(A0 \wedge A1) \vee (B0 \wedge B1)$. An additional, uniformly random feature was introduced, which matched the class label of the records 75% of the time. The random feature was used at the root of the tree, resulting in a tree which was less accurate than that formed without this feature. C4.5 (Quinlan, 1993) and CART (Breiman et al., 1984) were found to suffer from the same problem.

The quality measure and search strategy used must both be computationally efficient if the technique is to be feasible. An important benefit of feature subset selection is the improved execution speed of the data mining algorithm on the reduced problem; this is clearly nullified if the technique takes more time to execute than it saves.

The potential improvement in the performance by the removal of features may seem counter-intuitive, since statistical decision theory has shown that the probability of classification error should decrease as the number of features increases. However, this property generally only holds for infinitely large databases, upon which "overfitting" is impossible. For real, finite databases, data mining systems will tend to overfit the data, producing rules which incur severe accuracy degradation when applied to unseen data.

We consider the application of feature subset selection techniques to classification algorithms within this study. Such algorithms produce class descriptions for a database of preclassified objects. All fields within the database will be described as $input$ except for the class field, which shall be referred to as the $output$. Feature subset selection algorithms are thus applied only to the input fields within a database.

Within this paper, examples of feature subset quality measures and search strategies are given in sections 2 and 3 respectively. Some methods which may be used to establish the quality of the final solution produced by a feature subset selection algorithm are presented in section 4. The simulated annealing based data mining algorithm to which feature subset selection is to be applied and the commercial databases used are described in sections 5 and 6 respectively. Section 7 describes the feature subset selection algorithms which are investigated within this study; the results of the experiments undertaken are presented in section 8.

2. Feature subset quality measures

A number of potential measures exist for determining the predictive power of a feature subset, including the following, some of which are described in (Devijver and Kittler, 1982).

2.1. Single feature quality measures

The following feature quality measures only measure the quality of a single feature.

- Statistical techniques.

  An example of such a technique is the Chaid (ANGOSS, 1995, Kass, 1980), or $cluster$ tree induction algorithm, which ranks the fields within the database at each node in the tree, in order to determine which should form the test at the node. The ranking therefore gives a measure of the relative field qualities.

  The rankings are formed in the implementation used within the KnowledgeSeeker package (ANGOSS, 1994) by attempting to split the database into subsets using each field in turn; the fields are ordered by the statistical significance (determined by a χ^2 test) of the splits which they form.

  The $exhaustive$ (ANGOSS, 1995) tree induction algorithm is essentially the same as the $cluster$ method, differing principally in its halting condition. Rather than stopping when the statistical similarity of the most similar adjacent pair of categories is less than some pre-determined level, the algorithm continues until only two categories exist. The split with the greatest statistical significance over all the iteratively generated splits is then applied to the field, provided its significance is not less than the pre-determined level.

- Information theoretic techniques.

  An information gain based measure, used by ID3 (Quinlan, 1983, Quinlan, 1986) and C4.5 (Quinlan, 1993), is an example of such a technique. For the databases which we consider, which contain d records with Boolean output fields, the entropy over the whole database is defined as

  $$entropy_{output} = -(p\log_2 p + q\log_2 q),$$

  where p and q represent the probability of a randomly selected output value being true or false respectively. Now let us consider the entropy for a binary split. We define v and w to be the number of records in the set above and below the split boundary respectively. Let p_v and q_v denote the probability of the output field having value true and false respectively for records above the split boundary; p_w and q_w are similarly defined for records below the split boundary. The entropy for a binary split is then (Quinlan, 1993, Shannon and Weaver, 1963)

  $$\begin{aligned} entropy_{split} &= v/d \times (-p_v \times \log_2 p_v - q_v \times \log_2 q_v) \\ &+ w/d \times (-p_w \times \log_2 p_w - q_w \times \log_2 q_w). \end{aligned} \quad (1)$$

  The information gained by a binary split is (Quinlan, 1993)

  $$\textit{information gain} = entropy_{output} - entropy_{split}.$$

For each field, every possible binary split is attempted and tested; the highest information gain achieved in this way is recorded as the field quality. The split point should be defined using the same numeric representation as the data mining algorithm for which the features are to be selected. This means that the measure only evaluates binary splits which are possible using the data mining algorithm, which increases the likelihood of a close correlation between the feature quality measure and the performance of the system.

- Class separability techniques.
  - Interclass distance. A measure of distance between pairs of values within the field to be tested must firstly be defined. For example, Euclidean interclass distance (Devijver and Kittler, 1982) is defined for two objects a and b of different classes (i.e. with different output values) with values a_v and b_v for the tested field as
    $$\delta(a, b) = [(a_v - b_v)^2]^{1/2}.$$
    The average pairwise distance between objects in different classes thus gives a measure of the separability of the two classes; this separability gives a measure of the quality of the feature. If all objects in the database have very similar values for a feature then the average pairwise distance between objects belonging to different classes will be low, since the field is not highly predictive.
  - Probabilistic distance. This form of measure allows the extent of overlap between classes to be taken into account. For example, consider two classes, A and B, each with equal probability of occurrence within the database. If, for all feature values whose probability of occurrence given that the class is B is greater than zero, the probability of occurrence given that the class is A is zero then there is no overlap between the classes. Conversely, if for all feature values, the probability of occurrence given that the class is B equals the probability of occurrence given that the class is A then the two classes are completely overlapping.
    A probabilistic distance based measure, using a normalised distance between means, is implemented within the Tooldiag (Rauber, 1996) package. For a database with Boolean valued output, where the probabilities of a random record having output value 'true' and 'false' are denoted p and q, this is defined in the following way. For a field F, the mean and standard deviation for the class 'true' are denoted m_T and s_T respectively. Similarly, the mean and standard deviation values for the field of class 'false' are denoted by m_F and s_F respectively. The quality of the field by this measure is then defined as:
    $$Quality = p \times q \times [1 - \frac{(s_T + s_F)^2}{(m_T - m_F)^2}].$$
    If the two classes overlap completely then m_T and m_F will be equal, giving a maximally negative quality. Conversely, if the means are not equal and the standard deviations are both zero then the field will have a maximally positive quality, since the classes are disjoint. Between these two extremes, the measure will favour fields which maximise the difference between the means and minimise the standard deviations.

2.2. Multiple feature quality measures

The following feature quality measures may be applied to feature subsets of any size.

- Estimated minimal error probability. This is used within wrapper feature subset selection algorithms. The classifier is trained with the feature subset using the training portion of the database alone. The error rate of the classifier on new data must then be evaluated. This clearly cannot be performed using the testing portion of the database, since this could no longer be used as unseen test data. Instead, an estimate of the expected error rate for the classifier if presented with unseen data must be made using the training database alone. Potential error estimation methods include n-fold cross validation (Breiman et al., 1984, Weiss and Kulikowski, 1991) and bootstrapping (Weiss and Kulikowski, 1991).
- Interclass distance, as previously described for single features, may also be applied to multiple features. A measure of distance between pairs of feature vectors within the database, rather than field values, must be defined. For two k-dimensional feature vectors a and b (resulting from a feature subset of cardinality k), belonging to different classes, Euclidean interclass distance (Devijver and Kittler, 1982) is defined as

$$\delta(a,b) = [\sum_{j=1}^{k}(a_j - b_j)^2]^{1/2}.$$

  Again, the average pairwise distance between objects in different classes gives a measure of the separability of the two classes; this separability is used to measure the quality of the feature subset. If the set of features is powerfully predictive then most pairs of objects belonging to different classes will have significantly different values for every field. Conversely, if most pairs of objects which belong to different classes have very similar values for each feature within the subset then the subset will have very little (if any) predictive power.
- Probabilistic distance may be applied to multiple features in addition to single features as described previously. Again, the extent of overlap between classes gives a measure of the quality of the feature subset. Using the example of two classes, A and B again, consider a database within which as many objects belong to class A as to class B. If, for every pattern vector with probability greater than zero of occurrence given that the class is B, the probability of occurrence is zero given that the class is A then there is no overlap between the classes. Conversely, if for all pattern vectors, the probability of occurrence given that the class is B equals the probability of occurrence given that the class is A then the two classes are completely overlapping.

3. Feature subset search strategies

A variety of techniques to find the best feature subsets exist, including the following:

- Best features (Devijver and Kittler, 1982). A measure of the predictive power of a single feature, such as those described in section 2, is first used to obtain a score for each feature individually; the features are then ranked by the resulting scores. The feature subset is produced by selecting the best k fields by this ranking. Although simple, this method has the disadvantage of potentially missing subsets of features which, although weak in isolation, are powerfully predictive in combination.

- Sequential forward selection (SFS) (Devijver and Kittler, 1982, Michael and Lin, 1973, Whitney, 1971). This procedure simply greedily adds one feature at a time to the current feature subset. At each stage, the feature included within the subset is that which maximises the score which the constructed subset achieves when tested using some measure of predictive power. Initially, all fields are tested in isolation and the best is chosen for the feature subset of size one. For the feature subset of size two, the addition of every feature in turn within the feature subset of size one is examined; the feature whose inclusion results in the subset with greatest score is included. This process continues until the desired number of features have been included within the subset.

  The feature subset produced is likely to be superior to that yielded by the 'best features' method, since statistical dependence between features is taken into account. However, the algorithm never removes features from the subset, even if they become superfluous after the addition of other fields. It has also been shown that this algorithm has the potential to select the worst possible set of features (Cover and VanCampenhout, 1977).

  Variations of SFS include sequential backward selection (SBS) (Devijver and Kittler, 1982), which works in the opposite direction, discarding features one at a time from the full set. Both methods also exist in forms where more than one feature is added or deleted at each stage.

- Branch-and-bound (Chang, 1973, Devijver and Kittler, 1982, Narendra and Fukunaga, 1977). For any pair of feature subsets A, B, where $A \subset B$, a measurement is used for which the predictive power of B will be greater than or equal to that of A. This property allows many feature subsets to be rejected without evaluation. The approach requires the definition of a maximum size k for the feature subset; the best subset of cardinality $\leq k$ is then produced.

  The algorithms works by building a search tree, for which each level l will contain $F - l$ features, for a database containing F fields; the tree will have a maximum depth of $F - k$. As the search progresses, branches of the tree will be explored up to level $F - k$; the predictive power of the best feature subset at this level is stored as b. Any nodes in the tree with predictive power less than b can be removed, since their children will all have at best equal power. The algorithm examines the least powerfully predictive successors to a node in the tree first, since they are the most likely to be rejected.

  This method has the disadvantage of assuming that the performance of the data mining system will not degrade as the size of the feature set increases. Unfortunately this is rarely the case. For example, C4.5 has been shown to give inferior error rates when irrelevant features are introduced (Thrun et al., 1991). Any measurement of predictive

power which is suitable for the the branch-and-bound algorithm would not give a greater score to the feature set before the inclusion of the irrelevant features. Another serious shortcoming is the limitation on dimensionality (Foroutan and Sklansky, 1987).

Table 1. A small example database

Field x	Field y	Field z	Class
2	1	1	T
1	2	1	T
2	2	1	F
2	2	2	F
2	2	2	F

The small example in table 1 allows us to demonstrate how the best possible feature subset may be missed by some approaches. For this database, Class = (Field(x) = 1 $\vee$ Field(y) = 1). The predictive power of each field is measured using function 1 in section 2.1. The lowest entropy over all binary splits on a field is thus taken as a measure of its predictive power. The most powerfully predictive fields will have the lowest entropy; ideally, the entropy should be at its minimal value of zero. In such cases, knowledge of the value of the field allows us to predict without error the class of the object.

Using this measure of predictive power, fields x, y and z have scores 1.85, 1.85 and 0.55 respectively. This means that approaches such as SFS and 'best features' would select field z to form a feature subset of size one. Such a subset would clearly be the best of its size. However, to form a subset containing two features, these methods would include either field x or y along with z. Rules or trees produced using such subsets would achieve no entropy reduction from such an inclusion. The optimal subset of size two, containing fields x and y, allows the production of rules or trees for which the measure of predictive power has value zero. Such a subset would be found by using a method such as SBS. However, the SBS method would produce a non-optimal feature subset of size one.

4. Evaluating the results of the feature subset selection mechanism

For most of the feature subset selection methods which have been described, the set of features which are produced will not necessarily be optimal with respect to the classification performance of the data mining algorithm to which they are applied. Indeed, the worst possible set of features may in fact be produced.

In most cases, the feature subset which is selected will be neither the best nor the worst possible, but somewhere between the two extremes. In such situations, the simplest test of the quality of the subset is to compare the performance of the data mining system using all fields to that achieved using the selected subset. However, such a test may prove infeasible for large databases with very large numbers of fields. A possible alternative to this approach is to make use of the definitions of relevance described in (John et al., 1994). These describe the relevance of a field F, where the set S contains all fields except F, and the value assignment to all features in S is denoted by s, as follows.

1. The field is defined to be strongly relevant iff there exists

   - some value f for field F,
   - some class c,
   - some value assignment s_i to all features in S,

   where

   $p(F = f, s = s_i) > 0$ and

   $p(Class = c \mid F = f, s = s_i) \neq p(Class = c \mid s = s_i)$.

   The removal of such a field would always cause a loss in prediction accuracy.

2. The field is defined to be weakly relevant iff it is not strongly relevant and there exists some S' (a subset of S, for which the value assignment to all features is denoted by s') for which there exists

   - some value f for field F,
   - some class c,
   - some value assignment s'_i to all features in S',

   where

   $p(F = f, s' = s'_i) > 0$ and

   $p(Class = c \mid F = f, s' = s'_i) \neq p(Class = c \mid s' = s'_i)$.

   A weakly relevant field therefore contributes to classification accuracy in some circumstances.

3. If the field neither strongly or weakly relevant then it is defined to be irrelevant.

A measure of the proportion of strongly relevant, weakly relevant and irrelevant features within the subset may therefore be used to establish its quality. Some examples of alternative quality measures, based upon principles such as inconsistency and entropy, may be found in (Liu and Setiono, 1996b, Koller and Sahami, 1996).

5. Simulated annealing

5.1. The standard approach

As described previously in (de la Iglesia et al., 1996), let us assume that we have a flat file, D, of d records and that all fields have known values for each $r \in D$. We will refer to these records as *defined records*. Therefore, D consists of a set of records specifying values for the attributes $A_1, A_2, \ldots, A_n$ over the domains $Dom_1, Dom_2, \ldots, Dom_n$, respectively. We are asked to find a rule of the form $\alpha \Rightarrow \beta$ which appears to hold for some records in this database. For simplicity, α, the precondition of the rule can be interpreted as a conjunction of the form

$$(A_{\delta_1} \in v_{\delta_1}) \wedge (A_{\delta_2} \in v_{\delta_2}) \wedge \ldots \wedge (A_{\delta_i} \in v_{\delta_i})$$

with $v_{\delta_1} \subseteq Dom_1, v_{\delta_2} \subseteq Dom_2 \ldots, v_{\delta_i} \subseteq Dom_i$. The predictive part of the rule, β, can take the form of $(A_j \in v_j)$, where $v_j \subseteq Dom_j$.

We use $\alpha(r)$ to denote that α is true for record r. Associated with any rule $\alpha \Rightarrow \beta$ are three sets of records,

$$\begin{aligned} A &= \{r \mid \alpha(r)\},\ B = \{r \mid \beta(r)\} \text{ and} \\ C &= \{r \mid \alpha(r) \wedge \beta(r)\} = A \cap B. \end{aligned}$$

We can then define the integer values a, b and c by

$$a = |A|,\ b = |B| \text{ and } c = |C|.$$

Figure 1 illustrates the situation.

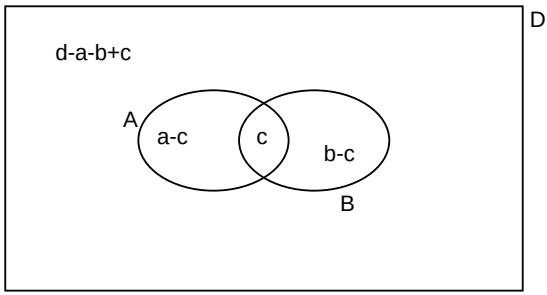

Figure 1. Venn diagram for a rule in a database of defined records.

Remembering that d denotes the size of the database, one can define various ratios between these values which measure properties of the rule. The three ratios which are of greatest interest for measuring the quality of our rules are as follows:

$App(\alpha \Rightarrow \beta) = \frac{a}{d}$	APPLICABILITY of the rule,
$Acc(\alpha \Rightarrow \beta) = \frac{c}{a}$	ACCURACY of the rule,
$Int(\alpha \Rightarrow \beta) = \frac{b}{d}$	INTEREST of the rule.

The SA based approach, using the toolkit SAmson (Mann, 1995), has been previously described in (de la Iglesia et al., 1996). Our measure of solution quality for the simulated annealing algorithm is described briefly here.

Rules produced by the data mining package must maximise c, since this represents the number of records for which the rule fired correctly. The number of records for which the rule fired incorrectly $(a - c)$ must clearly be minimised. We also wish to minimise $(b - c)$, since this represents the number of records for which the rule did not fire. However, the size of b is frequently fixed, since the output value of our rules may be predetermined. This is the case for many problems, and in particular, for those used in this paper. We therefore wish to maximise $\lambda c - a$, for some $\lambda \in \mathcal{R}$.

This allows us to control the type of rule produced using a single parameter. The value of λ determines the generality of rules produced; increasing λ gives greater applicability with reduced accuracy and vice versa.

We are interested in rules that have both high accuracy and applicability. However, the data mining projects which we undertake are for commercial clients, who generally wish to determine the value of β for themselves. We will therefore assume the value of β to be fixed. We can then construct more complex rules with preconditions in Disjunctive Normal Form (DNF) by discovering a number of rules, $\alpha_1 \Rightarrow \beta, \alpha_2 \Rightarrow \beta, \ldots, \alpha_k \Rightarrow \beta$, all with preconditions of the above format and combining them to obtain a single rule $\alpha_1 \vee \alpha_2 \vee \ldots \vee \alpha_k \Rightarrow \beta$. For a rule with a given level of applicability, we may measure its quality by comparing its accuracy to the maximum which may be achieved, which is $Min\{\frac{b}{a}, 1\} = Min\{\frac{Int}{App}, 1\}$. When the applicability of a rule does not exceed its interest, its maximum accuracy is clearly 100%.

Our key accuracy measurement is then the improvement gained over that which may be achieved by just predicting the specified output value for all records. For example, given a specified output value of 'false', a rule which predicts this with 70% accuracy and high applicability is of high quality if the proportion of records within the database with output value 'false' is only 10%.

The package developed was used to produce a simple single rule. After the first rule was produced, the records classified by that rule were removed, and the process started again. If a second rule was found, then the preconditions of the first and second rule were OR-ed together, to form a rule whose precondition was in DNF, and so on. However, the results shown in subsequent sections refer to simple rules, that is, they are the result of one run of the SA. The output value of rules produced was forced to be 'false' for the first and 'true' for the second case study. The fitness was evaluated using λ values of 1.41 and 55 for the first and second problems respectively. The adequate λ values for each problem were found after some experimentation, and depend on the required level of accuracy/applicability desired in the rules. In particular, for databases where the output value required is an exception, the λ values need to be considerably larger to allow for a greater difference between a and c. Experiments were also performed using a range of λ values; the effect of these on rule accuracy/applicability is documented in section 8.2.

As described in (Rayward-Smith et al., 1995), the precondition of a rule was represented as the conjunction of a set of Gray coded upper and lower limits (one pair for each input field). Rules were therefore of the form

if
$l_1 \leq x_1 \leq u_1$ **and**
$l_2 \leq x_2 \leq u_2$ **and**
...
$l_n \leq x_n \leq u_n$
then y,

where k is the number of input fields and l_i and u_i represent lower and upper limits for input field i.

The rules produced by the SA were simplified before being tested against the appropriate databases. This process simply removed any inequalities within the rule which excluded

less than a threshold minimum percentage of the database. For each rule, the threshold value was raised to the highest value possible without reducing the fitness value.

A neighbourhood of a solution was generated by selecting a bit at random and inverting it. An initial temperature of 10 was used within a Lundy and Mees (Lundy and Mees, 1986) cooling schedule, with a Beta value of 0.9 and 20 proposed moves spent at each temperature step. Non-monotonicity was introduced within the cooling schedule by means of a threshold parameter. This works by returning the temperature to half its value at the point of the last temperature rise when the threshold percentage of this value, set to 15, is reached. The temperature value at the point of the last temperature rise is considered to be the initial temperature if no rises have yet been performed. This scheme was found to give superior results to a monotonic schedule.

For each feature subset used, in addition to the full complement, five runs of the SA were performed using these parameters. The rule with the greatest fitness value produced was then tested on the appropriate database. The testing involved obtaining accuracy, applicability and fitness values for the rule over both the training and testing portions of the database.

5.2. Incorporating feature subset selection into the SA algorithm

Our simulated annealing algorithm may be modified so that it searches simultaneously for promising feature subsets and high quality rules. This is done by adding a penalty to the $\lambda c - a$ function, which is used to measure the quality of rules. The new quality measure used, for a rule which makes use of g features, is $\lambda c - a - \gamma g, \gamma \in \mathcal{R}$. A range of γ values was experimented with, with the SA algorithm being run five times with each. The best rule produced using this modification of the SA algorithm for the two databases is reported in section 8.

5.3. Thermostatistical persistency

Thermostatistical persistency (Chardaire et al., 1995) (TP) is an extension of simulated annealing, designed to improve its performance. It was originally designed to solve optimisation problems, within which all variables were binary. Over every temperature step, the mean value of every variable was computed. Any variable with mean below a pre-determined lower threshold was fixed at zero and excluded from the search. Similarly, variables with mean value above a pre-determined upper threshold were fixed at one and excluded from the search. The number of iterations spent at the next temperature step was then reduced by the same proportion as the number of non fixed variables.

The aim of this method is thus to make 'obvious' decisions in the optimisation process early on, then concentrate upon more difficult choices later. The technique was adapted for our problem by creating a binary array, with one bit for every input field within the database. At every proposed move, the bit for a field was set to 0, unless one of its limits had been moved and the resulting rule accepted. The mean value of each bit was measured over a number of proposed moves. At the end of this time period, any fields with mean

binary value below some threshold minimum were fixed at their current value and excluded from the search. The search then continued, with both the number of proposed moves spent at each temperature step and the number of proposed moves over which the means were measured set to their initial value multiplied by m/n, where m out of the total n input fields remained unfixed.

For both databases, a range of experiments were performed. The number of proposed moves over which the mean binary values were initially computed was set to a range of values; for each of these, the value of the threshold used to fix fields was also experimented with. The best rule produced from the experiments performed for each database is reported in section 8.

6. Case studies

The two databases described in the subsequent section contained information from which rules explaining customer behaviour were to be induced. The company to which the data belonged sought rules of sufficiently high quality to give them a commercial advantage.

The databases had several input fields, of type continuous or discrete with total ordering (integer). The single output field in both cases was of Boolean type.

6.1. Problem one

The database for the first problem, described previously in (de la Iglesia et al., 1996), contained 32,378 records, each with 17 input fields. The database was split at random into a separate testing and training set; the resulting databases are shown in table 2.

Table 2. Databases for problem one

Database	No. of records	Percentage with output 'false'
Original	32,378	70.0
Training	16,311	70.3
Testing	16,067	69.7

The company was interested only in finding rules to describe the output 'false'. For this problem, unless the applicability of a rule exceeds 70.3% on the training portion of the database, the maximum accuracy of a rule on this data is 100%.

6.2. Problem two

This database contained 29,645 records, each having 66 input fields. The company was interested in rules which predicted an output value of 'true', despite this being the case for only 2.2% of the database. As before, the database was split at random into a separate testing and training set, shown in table 3.

Table 3. Databases for problem two

Database	No. of records	Percentage with output 'true'
Original	29,645	2.23
Training	14,911	2.29
Testing	14,734	2.17

For this database, any rule with applicability which does not exceed 2.29% on the training portion of the database will have a maximum accuracy of 100%. All rules with greater applicability will have a maximum accuracy of b/a. For example, a rule with 50% applicability will have a maximum accuracy of $\frac{342}{7456} = 4.59\%$.

7. The feature subset selection algorithms

The majority of the experiments undertaken made use of the simple 'best features' search strategy, since this was computationally simple and thus suitable for large databases. A number of single feature quality measures were used within this approach; the best of these was then used within a multiple feature quality measure. Finally, an exhaustive search strategy was investigated using a multiple feature quality measure based upon the best of the single feature quality measures.

7.1. Best features approach

The 'best features' search strategy for finding the best feature subset was initially used, since this had extremely low computational complexity and therefore was suitable for application to large databases. The algorithm was initially used with quality measures which examined each feature individually. The best of these was then used within a measure which examined groups of features to determine their quality.

7.1.1. Single feature quality measures

In order to determine the best measurement to be used within the algorithm a comparative study of several was undertaken, using the database of the first case study. Each measure was applied to every field individually, rather than sets of fields. The following measures were investigated:

1. The information gain measure, described in section 2.1.
2. Euclidean interclass distance, as described in section 2.1.
3. The 'cluster' method used in the commercially available package KnowledgeSeeker (Biggs et al., deVille, 1990), as described in section 2.1.

4. The ‘exhaustive’ method used within KnowledgeSeeker, as described in section 2.1.
5. A probabilistic distance based measure, using a normalised distance between means, as described in section 2.1.

Each measure was used to determine the quality of every feature in the database individually. The rankings are shown in table 4.

Table 4. Feature quality rankings

Field number	Ranking				
	1	2	3	4	5
1	13	6	12	12	13
2	8	4	9	8	8
3	12	8	10	10	14
4	6	14	8	4	7
5	16	10	7	6	17
6	9	1	13	13	11
7	7	3	6	5	5
8	1	2	1	9	4
9	5	5	5	7	3
10	11	12	14	14	9
11	4	7	2	3	6
12	3	16	4	2	2
13	15	15	15	15	10
14	10	9	11	11	12
15	14	13	16	16	15
16	2	11	3	1	1
17	17	17	17	17	16

These results show that the measures used differ quite widely in their ranking of features. We may, however, establish which of these suits our data mining algorithm best by empirical analysis. The measure for which the algorithm gives the best rules, for a given subset size, will clearly be the most suitable.

The SA package was run five times using only the best field by each of the above measures. The experiments were repeated using the best two fields, and again using the best three. The fitness of each best rule is displayed in table 5. Figures in brackets refer to results achieved using the test database. It is important to note that they are included for reference only and were not used to determine the relative quality of the measures. This is because such use would potentially involve the ‘unseen’ data in the quality measure selection process and thus effectively render it ‘seen’.

The same experiments were performed using the database for the second problem, yielding the results shown in table 6.

The results indicate that the information gain based method is probably the most appropriate for our data mining algorithm. For the first database, rules of superior quality may be produced when two fields are used rather than with all present (these results are shown in detail in section 8.1). In both cases, the method produced superior results to all other approaches. Moreover, the simple ‘best features’ search which we have used can clearly deliver good results, whilst being computationally less expensive than any other approach.

Table 5. Problem one fitness values

Quality measure	Fitness		
	Best field only	Best two fields	Best three fields
1	968.9(968.9)	1203.42(1125.52)	1203.42(1125.52)
2	340.05(246.14)	973.57(967.72)	988.52(959.08)
3	968.9(968.9)	968.9(968.46)	1203.42(1125.52)
4	889.54(774.05)	889.54(774.05)	965.5(955.8)
5	889.54(774.05)	889.54(774.05)	1018.17(823.86)

Table 6. Problem two fitness values

Quality measure	Fitness		
	Best field only	Best two fields	Best three fields
1	4942(3582)	5088(4726)	5107(4749)
2	3942(2854)	4243(2400)	4265(2426)
3	3899(2811)	3899(2811)	4942(3582)
4	3899(2811)	3899(2811)	3899(2811)
5	4942(3582)	5088(4726)	5088(4726)

7.1.2. The multiple feature quality measure

Despite producing good results, even the best of the single feature quality measures does not cater for interactions between fields. For example, in the extreme case of the most predictive field f_1 being duplicated as f_2, we would consider f_1 and f_2 to be the first and second best fields respectively. However, f_2 will clearly offer no enhancement in predictive power when used in conjunction with f_1, despite being as powerful in isolation.

To deal with this potential deficiency, we investigate a modified information gain approach. This makes use of the information gain measure described previously, but examines fields in groups rather than individually. It works by building a decision tree of a limited size and assigning a score to each field used within it, based upon the quality of the test within which it is used and the number of records to which it applies. The score for each field is computed using the following algorithm.

```
1.  initialise all field score values to equal zero;
2.  while total number of different fields used within the tree
3.  < required feature subset size do
4.  begin
5.      for every node in the tree without children do
6.      begin
7.          for every field within the database do
8.              measure the information gain for every binary split which is
9.              possible using the same representation scheme as the SA based
10.             data mining algorithm;
```

```
11.             use the best binary split as the test at the node and generate its
12.             two children;
13.             let the information gain for this split = gain_node;
14.             let the number of records to which the node applies = d_node;
15.             let score_node = gain_node × d_node / d;
16.             add score_node to score value for field used within split at node
17.         end
18.     end
```

This scheme is based upon a simplified form of the ID3 or C4.5 decision tree building process, halting when the tree contains a sufficient number of different fields to form the required size of feature subset. The information gain at each node will clearly only measure the decrease in uncertainty of the output value caused by the inclusion of the field within the rule. This means that nodes which apply to low entropy portions of the database can, at best, only give a relatively low information gain. The scheme will therefore tend to favour nodes which apply to areas of the database which have high entropy and partition it into areas within which this is considerably lower. The weighting of these values by the proportion of the database to which the node applies will tend to favour nodes which give a high information gain and apply to a large number of records.

If, as with our case studies, feature subsets of several sizes are to be investigated, then this approach is used to generate the largest of these first. This will produce a ranking from which smaller subsets are produced.

7.2. Exhaustive search

The results which we obtained suggested that a subset of size two may be sufficiently large to give comparable or superior performance to that achieved using the full feature set. This lead us to experiment with an approach which exhaustively tested every possible feature subset of size two. For every possible pair of fields, the following test was undertaken:

- For the first of the two features, the information gain for every possible binary split (using the same representation scheme as the SA algorithm) was recorded. This process was repeated for the second feature.
- The information gain for the best binary split on each of the two fields was compared. Let us denote the field for which the best binary split gave the highest information gain as f_{best}; the other field is denoted by f_{worst}.
- The best binary split was used to partition the database into two subsets, which we will refer to as X and Y. This split will clearly be made using the field f_{best}.
- For subset X, the information gain for every possible binary split (using the same representation scheme as the SA algorithm) using the field f_{worst} was computed. The binary split for which the information gain was highest was used to divide X into two. This process was repeated for subset Y.

- The information gain achieved by splitting the database into the resulting four subsets was measured. This was achieved by treating the four subsets as though they were created by forming a split at the root of a tree on a discrete variable with four possible values.

  We will label the four subsets L, M, N and O, which contain l, m, n and o records respectively. For every subset I, p_I and q_I denote the probability of the output field of its members having value true and false respectively. The entropy for the four subsets is therefore (Quinlan, 1993):

$$\begin{aligned} entropy_{subsets} = \quad & l/r \quad \times(-p_L \times \log_2 p_L - q_L \times \log_2 q_L) + \\ & m/r \quad \times(-p_M \times \log_2 p_M - q_M \times \log_2 q_M) + \\ & n/r \quad \times(-p_N \times \log_2 p_N - q_N \times \log_2 q_N) + \\ & o/r \quad \times(-p_O \times \log_2 p_O - q_O \times \log_2 q_O). \end{aligned}$$

  The information gain is then (Quinlan, 1993)

$$\textit{information gain} = entropy_{output} - entropy_{subsets}.$$

All possible feature pairs within the database were therefore ranked by their resulting information gain value and the best selected.

This approach was applied to the the two databases. Results produced by this method are documented, where a different subset to that produced using the standard information gain based technique is found, in section 8.

8. Experimental results

The fitness, accuracy and applicability of the rules produced are presented in the following sections for both fixed and multiple λ value experiments. Rules reported as 'default' are the most general possible, having an applicability of 100%. The results for the standard best features method refer to the feature subsets produced by the standard information gain approach; rules produced by the modified best features method refer to subsets produced by the modified information gain approach. The results reported as 'exhaustive pair' were produced using the best feature pair over all within the database (see section 7.2 for a description of this method); no results are shown where the pair was the same as that produced by the standard information gain approach. In all tables shown, TP refers to rules produced using thermostatistical persistency; MSA refers to the modified SA algorithm. Rows marked "Si" refer to the results produced for the standard method for i fields. Similarly, Rows marked "Mi" refer to results achieved using the modified method for i fields. Rows marked "all" refer to results produced using the full set of features.

8.1. Single λ value experiments

The first set of experiments were performed using λ values of 1.41 and 55 for the first and second problems respectively.

The best rule produced over five runs using the SA algorithm for each of the feature subsets selected by our algorithm was tested on the appropriate databases. The fitness, accuracy and applicability for both the training and testing databases are shown in tables 7 and 8, for the first and second problem respectively. The figures in the column marked '#' denote the total number of features which were used within the rule. Figures in brackets refer to results produced using the test database. The applicability of all of the rules produced for the first case study is less than the interest, so the maximum possible accuracy is 100%. For the second case study, the applicability of all of the rules produced exceed their interest; their maximum possible accuracies are therefore tabulated.

Table 7. Results for problem one

Method	#	Fitness	Accuracy (%)	Applicability (%)
All	2	1195.61(1133.36)	80.8(80.4)	52.5(52.6)
S1	1	968.9(968.9)	78(78.2)	59.7(59)
S2	2	1203.42(1125.52)	82.1(81.7)	46.8(46.3)
S3	2	1203.42(1125.52)	82.1(81.7)	46.8(46.3)
S4	2	1089.41(1045.7)	82(81.6)	42.6(43.3)
M1	1	968.9(968.9)	78(78.2)	59.7(59)
M2	2	1203.42(1125.52)	82.1(81.7)	46.8(46.3)
M3	3	1204.86(1124.55)	82.2(81.7)	46.6(46.1)
M4	3	1204.86(1124.55)	82.2(81.7)	46.6(46.1)
MSA	5	1225.21(1134.52)	81.7(81)	49.4(49.7)
TP	8	1223.55(1132.35)	81.7(81)	49.2(49.4)

Our approach, in its original form, produced a feature subset containing two features from which the SA could produce rules of comparable quality to those found when all features were present. Interestingly, these two features were also used within the rule which was produced using all features. Our approach did not appear to suffer greatly from interactions between fields; in modified form, only a very slight improvement in rule quality was achieved, for subsets containing three or more fields. The results produced using thermostatistical persistency were of good quality, although the rule produced required the largest number of features. The technique also suffered the disadvantage of being sensitive to parameter changes. The modified simulated annealing algorithm produced better results than all other approaches.

The results for the second problem exhibit a similar pattern to that displayed for the first. If the fitness values over the entire database are compared, a subset containing only two features will allow the production of rules of higher quality than when all of the features are present. Again, the modification to our feature subset selection algorithm

Table 8. Results for problem two

Method	#	Fitness	Accuracy (%)	Applicability (%)	Max accuracy (%)
All	38	5851(2812)	3.1(2.45)	55.6(55.2)	4.12(3.92)
S1	1	4942(3582)	2.98(2.66)	52.1(52.2)	4.4(4.15)
S2	2	5088(4726)	3.04(2.98)	50.7(50)	4.52(4.33)
S3	3	5107(4749)	3.05(2.99)	50.6(49.9)	4.53(4.34)
S4	3	5107(4749)	3.05(2.99)	50.6(49.9)	4.53(4.34)
M1	1	4942(3582)	2.98(2.66)	52.1(52.2)	4.4(4.15)
M2	2	5088(4726)	3.04(2.98)	50.7(50.1)	4.52(4.33)
M3	3	5117(4787)	3.07(3.02)	49.8(49.3)	4.6(4.4)
M4	4	5127(4795)	3.08(3.02)	49.7(49.2)	4.61(4.4)
MSA	4	5286(4625)	3.13(3)	49(48.5)	4.68(4.46)
TP	27	5288(4423)	3.13(3)	49(48.4)	4.68(4.48)

gave only slight advantages when subsets containing three or more features were produced. Thermostatistical persistency gave good results, but used more features than any other feature subset selection mechanism. The modified simulated annealing algorithm produced results comparable with the best of all other approaches. Larger γ values tended to produce rules containing fewer features, since each had a greater associated penalty. If the value of γ was set to a suitably high level then the number of features used within the best rule decreased from four (for the best rule over all γ values) to two. Interestingly, this rule was the same as that produced using the best feature subset of size two.

8.2. Multiple λ value experiments

The same experiments as were performed using single λ values were conducted using a range of λ values, with the exception of thermostatistical persistency, due to the extensive parameter experimentation required. In order to allow such numbers of experiments to be run, a 10% random sample of the training database for each of the two case studies was used. For the first case study, a training database containing 1695 records was used, of which 71.3% had an output value of 'false'. For the second problem, the training database contained 1484 records; 2.16% of these had an output value of 'true'.

Changing the value of λ controls the balance between accuracy and applicability in the rules produced. Figures 2 and 3 illustrate this process for case studies one and two respectively. The accuracy and applicability levels achieved on the testing database for the best rule produced from all approaches at each λ value are plotted.

The experimental results for the first and second case study are tabulated in tables 9 and 10 respectively. The fitnesses achieved on the 10% sample of the training database are given; figures in brackets show fitness when applied to the testing database. It should be noted that all feature subsets were produced using the 10% sample of the training portion of the database.

Tables 11 and 12 show the time (in CPU seconds) taken to produce the best rule in each of the multiple λ value experiments; the rows are labelled in the same way as for

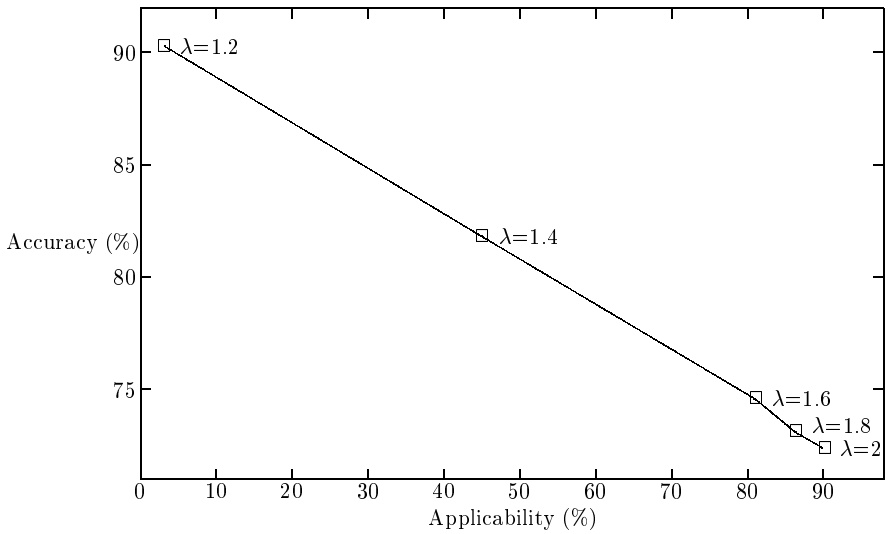

Figure 2. Case study one

tables 9 and 10. For both databases, the results illustrate the reduction in computational expense which may be achieved by reducing the number of features in the database. If the best feature alone is used then less than 15% of the time taken using all of the features is required for the first case study. The effect is even more pronounced for the second case study, where single feature experiments take less than 5% of the time required when using all of the features. Such substantial time savings would allow the data mining algorithm to be applied to databases which, if all features were present, would require an infeasible amount of time to data mine.

As the value of λ is modified for a given database, the best pattern with respect to the fitness function is likely to change, with lower values favouring higher accuracies. The best feature subsets with respect to each of these patterns are therefore in turn also likely to change, both in terms of the fields which they contain and their size. A feature subset selection method which is superior to all others for small λ values will therefore not necessarily also be superior for large values. This is illustrated by the results for the first database. The best four fields selected by the standard feature subset selection method gave the best results only for λ values of 1.2 and 1.4. The best four fields selected by the modified feature subset selection method only gave the best results for λ values of 1.6 and 2.0, whilst none of the feature selection methods tested gave an improvement over the full feature set for a λ value of 1.8. There was also a general downward trend in proportional improvement given over the full feature set with increasing values of λ, from a best of 836% at $\lambda = 1.2$ to 0.1% at $\lambda = 2$. The decreased improvement yielded at larger λ values is further illustrated by the γ fitness function results. A combination of large λ and γ values lead to the production of rules containing no inequalities, since the penalty incurred by the addition of an inequality exceeded the improvement in fitness which it yielded.

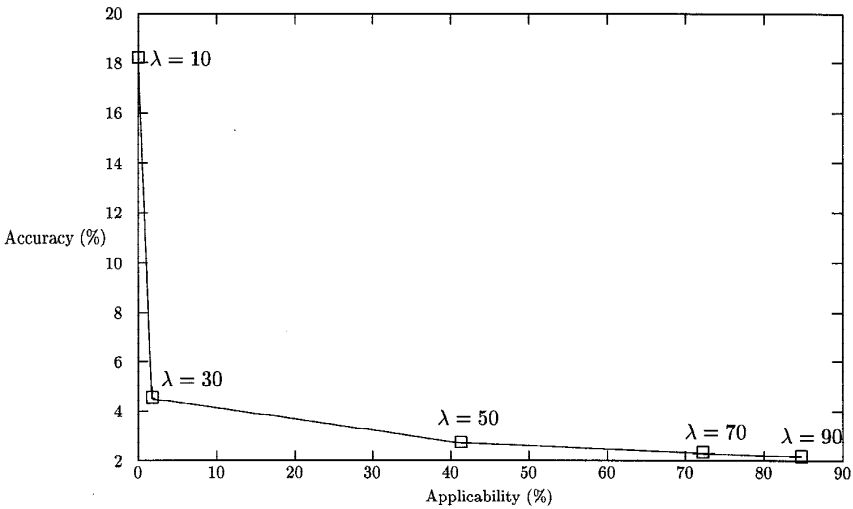

Figure 3. Case study two

9. Conclusions

Our data mining approach does not attempt to produce a classification for the whole database and then search within this for interesting patterns. Instead, it searches for interesting patterns directly. As a consequence of this, it is likely to be able to give high quality results using feature subsets of very small size, as our results have shown. Provided a measure of subset quality is used which has strong correlation with the performance of our data mining algorithm, a simple algorithm may be used to produce them. This simplicity allows the approach to be applied to large databases, containing considerable numbers of fields.

The feature subset selection technique used yields several important advantages. The reduction in problem size for the SA algorithm allows us to tackle much larger databases than would be feasible otherwise. In databases with very large numbers of fields, the problem of overfitting may also be substantially reduced. Even when this is not the case, rules are produced which may be understood more readily. The identification of fields which are powerfully predictive represents substantial knowledge in itself for the company. Indeed, such a discovery is viewed by them as a useful 'data prospecting' process, within which broad, 'headline information' is rapidly revealed. Perhaps of greatest commercial significance is the identification of fields which have little or no predictive power. Such fields will often have been bought in by the company; any which may be safely discarded

Table 9. Fitness values for case study one

Method	λ				
	1.2	1.4	1.6	1.8	2.0
All	4.8(5)	116(905.2)	296.2(2387.2)	519(4739.4)	741(6464)
S1	10.2(-11.2)	124.4(904.2)	299.4(2454.6)	516.2(4434.8)	738(6464)
S2	27.2(-11.2)	148.4(1051.8)	299.4(2454.6)	516.2(4434.8)	738(6464)
S3	27.2(-11.2)	148.4(1051.8)	299.4(2454.6)	516.2(4434.8)	738(6464)
S4	33.4(41.8)	148.4(1051.8)	301.8(2467.8)	518.4(4445.4)	740(6470)
M1	10.2(-11.2)	124.4(904.2)	299.4(2454.6)	516.2(4434.8)	738(6464)
M2	27.2(-11.2)	148.4(1051.8)	299.4(2454.6)	516.2(4434.8)	738(6464)
M3	36.4(1.2)	148.4(1051.8)	302.8(2514)	516.2(4434.8)	740(6465
M4	36.4(1.2)	147.6(1034.6)	302.8(2514)	516.2(4434.8)	740(6471)
γ = 0.1	2.0(2.8)	89.2(713.6)	294.6(2467.2)	512.2(4404.6)	743(6382)
γ = 1	7.8(4.6)	134.6(898)	299.6(2422.4)	521.6(4405.8)	744(6448)
γ = 3	8.4(-184.8)	124.4(963.6)	294.6(2467.2)	520.2(4417.8)	742(6406)
γ = 5	1.2(17.4)	93(678.4)	303.8(2501.6)	517.4(4428.2)	739(6421)
γ = 7	1.2(17.4)	91.2(747.8)	303.2(2475.4)	502.8(4230.2)	739(6431)
γ = 10	8.4(-184.8)	101.6(719.6)	294.6(2467.2)	500.8(4229.8)	736(6413)
γ = 15	10.4(-95.8)	148.4(1051.8)	301.2(2463.4)	499.4(4224.2)	default
γ = 20	2.4(6.4)	91.2(747.8)	298.4(2467)	default	default
γ = 25	7.8(22.8)	147.6(1034.6)	274.6(2314)	516.2(4434.8)	default
γ = 30	7(43)	148.4(1051.8)	296.8(2466)	default	default

Table 10. Fitness values for case study two

Method	λ				
	10	30	50	70	90
All	0(0)	0(0)	923(63)	1520(1833)	2150(6223)
S1	0(0)	60(-198)	402(118)	979(6028)	1619(11328)
S2	18(-8)	58(-13875)	460(22)	1026(5384)	1650(13785)
S3	20(-186)	179(-842)	560(292)	1167(5159)	1807(10699)
S4	23(-74)	194(-915)	611(510)	1229(4545)	1859(9986)
M1	0(0)	60(-198)	402(118)	979(6028)	1619(11328)
M2	18(-8)	58(-13875)	460(22)	1026(5384)	1650(13785)
M3	18(-1)	171(-169)	568(507)	1068(4273)	1668(10707)
M4	18(-12)	135(-110)	568(507)	1068(4272)	1668(10707)
Exhaustive pair	0(0)	93(85)	310(1504)	873(5537)	1473(10537)
γ = 10	11(-184)	185(-126)	663(-105)	1445(2572)	2055(6958)
γ = 20	-2(-9)	156(-103)	868(321)	1411(3395)	2057(9042)
γ = 30	8(-126)	55(-32)	454(926)	1364(4303)	1951(9329)
γ = 40	0(0)	157(48)	680(187)	1375(3890)	1971(8902)
γ = 50	9(-1)	55(-7)	769(789)	1315(3798)	1879(10016)
γ = 60	7(-39)	71(-74)	518(1248)	1189(5618)	1848(9854)
γ = 70	7(-20)	96(94)	471(2157)	1189(5618)	1845(10696)
γ = 80	6(-12)	53(-49)	642(571)	1191(6435)	1822(9616)
γ = 90	0(0)	74(83)	629(596)	1135(5658)	1787(10768)
γ = 100	9(9)	20(-272)	575(714)	1119(6092)	1619(11328)

will bring an instant financial saving. At the very least, every field brings an associated collection and storage cost, which may be saved if it proves superfluous.

Table 11. Times for case study one

Method	λ				
	1.2	1.4	1.6	1.8	2.0
All	117	112	113	112	112
S1	17	16	15	16	16
S2	24	22	22	23	22
S3	29	28	29	30	29
S4	36	35	34	36	36
M1	17	16	15	16	16
M2	24	22	22	23	22
M3	28	31	28	28	28
M4	37	35	35	35	35
$\gamma = 0.1$	117	117	114	121	116
$\gamma = 1$	116	115	115	111	111
$\gamma = 3$	120	119	119	117	116
$\gamma = 5$	116	116	116	116	114
$\gamma = 7$	114	113	113	113	113
$\gamma = 10$	120	119	115	114	116
$\gamma = 15$	116	116	116	116	116
$\gamma = 20$	116	120	113	113	114
$\gamma = 25$	117	114	112	115	115
$\gamma = 30$	113	113	113	113	114

Table 12. Times for case study two

Method	λ				
	10	30	50	70	90
All	358	355	368	366	365
S1	15	16	16	15	14
S2	19	18	19	19	19
S3	26	26	25	25	25
S4	30	32	32	30	30
M1	15	16	16	15	14
M2	19	18	19	19	19
M3	23	23	24	24	23
M4	31	29	29	29	29
Exhaustive pair	18	16	17	17	18
$\gamma = 10$	361	365	364	367	365
$\gamma = 20$	373	365	365	368	365
$\gamma = 30$	358	359	359	359	360
$\gamma = 40$	374	393	360	360	361
$\gamma = 50$	365	361	366	361	360
$\gamma = 60$	364	372	371	375	361
$\gamma = 70$	368	374	369	362	362
$\gamma = 80$	370	363	362	363	363
$\gamma = 90$	399	397	373	365	362
$\gamma = 100$	367	362	361	360	376

A modification of our approach to take into account interactions between fields has been compared to the original scheme. Although improvements are made in some experiments

using this method, for the databases which we use they are often marginal. We have also investigated a modification which exhaustively tests all feature subsets of size two, so that fields which are powerfully predictive only in combination may be selected. For the variable λ value experiments on the second database, this approach gave improvements over other feature pairs for all but one of the experiments. However, in all other cases, the best feature pair proved to be the same as that produced by our standard feature subset selection method.

Thermostatistical persistency appears to produce good results as a feature subset selection mechanism. However, its sensitivity to parameter values means that significant experimentation may have to be undertaken before its full benefits are realised.

Despite the improvements in solution quality which the feature subset selection methods have yielded, the performance of our data mining algorithm is not dependent upon such preprocessing in many situations. We have demonstrated that, by simultaneously searching for high quality rules and promising feature subsets, a modification of our algorithm produces results of equal or superior quality to all of the other methods, for the fixed λ values used. The approach does not give such a consistent advantage when λ values are varied, although for the second database the results are superior for all but one of the experiments undertaken.

The results produced using varying λ values have illustrated the effect of pattern generality upon the feature subset selection technique. Patterns at different levels of generality, as our results suggest, may contain different sets of features and therefore demand alternative feature subset selection methods. For all but one of the experiments undertaken, at least one of our feature subset selection methods gave improved results over the full feature complement. We must therefore conclude that, in addition to being tailored to the type of data mining algorithm being used, a feature subset selection mechanism must also be appropriate for the level of generality of the desired pattern.

References

ANGOSS: 1994, KnowledgeSEEKER 3.11.05 on-line help.

ANGOSS: 1995, *KnowledgeSEEKER for Windows Version 3.0 User's Guide*, Toronto, Canada.

Biggs, D., de Ville, B. and Suen, E.: 1991, A method of choosing multiway partitions for classification and decision trees, *Journal of Applied Statistics* **18**(1), 49–62.

Breiman, L., Friedman, J. H., Olshen, R. A. and Stone, C. J.: 1984, *Classification and Regression Trees*, Wadsworth and Brooks, Monterey, CA.

Chang, C. Y.: 1973, Dynamic programming as applied to feature selection in pattern recognition systems, *IEEE Trans. Syst. Man and Cybernet.* **3**, 166–171.

Chardaire, P., Lutton, J. L. and Sutter, A.: 1995, Thermostatistical persistency: A powerful improving concept for simulated annealing algorithms, *European Journal of Operational Research* **86**.

Cover, T. M. and Van Campenhout, J. M.: 1977, On the possible orderings in the measurement selection problem, *IEEE Trans. Sys. Man Cybern.* **7**, 651–661.

de la Iglesia, B., Debuse, J. C. W. and Rayward-Smith, V. J.: 1996, Discovering knowledge in commercial databases using modern heuristic techniques, *in* E. Simoudis, J. W. Han and U. Fayyad (eds), *Proc. of the Second Int. Conf. on Knowledge Discovery and Data Mining (KDD-96)*, pp. 44–49.

de Ville, B.: 1990, Applying statistical knowledge to database analysis and knowledge base construction, *Proceedings of the Sixth IEEE Conference on Artificial Intelligence Applications*, IEEE Computer Society, Washington.

Devijver, P. A. and Kittler, J.: 1982, *Pattern Recognition: a Statistical Approach*, Prentice-Hall International, London.

Foroutan, I. and Sklansky, J.: 1987, Feature selection for automatic classification of non-gaussian data, *IEEE Trans. Sys. Man Cybern.* **17**, 187–198.

John, G. H., Kohavi, R. and Pfleger, K.: 1994, Irrelevant features and the subset selection problem, *in* W. W. Cohen and H. Hirsh (eds), *Machine Learning: Proceedings of the Eleventh International Conference*, Morgan Kaufmann, San Francisco, pp. 121–129.

Kass, G. V.: 1980, An exploratory technique for investigating large quantities of categorical data, *Appl. Statist.* **29**, 119–127.

Kohavi, R. and Sommerfield, D.: 1995, Feature subset selection using the wrapper method: overfitting and dynamic search space topology, *in* U. M. Fayyad and R. Uthurusamy (eds), *Proc. of the First Int. Conf. on Knowledge Discovery and Data Mining*, AAAI Press, pp. 192–197.

Koller, D. and Sahami, M.: 1996, Toward optimal feature selection, *in* (Saitta, 1996).

Liu, H. and Setiono, R.: 1996a, Feature selection and classification - a probabilistic wrapper approach, *Proc. of the 9th Int. Conf. on Industrial and Engineering Applications of AI and Expert Systems* pp. 419–424.

Liu, H. and Setiono, R.: 1996b, A probabilistic approach to feature selection - a filter solution, *in* (Saitta, 1996).

Lundy, M. and Mees, A.: 1986, Convergence of an annealing algorithm, *Mathematical programming* **34**, 111–124.

Mann, J. W.: 1995, X-SAmson v1.0 user manual, *School of Information Systems, University of East Anglia*.

Marill, T. and Green, D. M.: 1963, On the effectiveness of receptors in recognition systems, *IEEE Trans. Inform. Theory* **9**, 11–17.

Michael, M. and Lin, W. C.: 1973, Experimental study of information measures and inter-intra class distance ratios on feature selection and ordering, *IEEE Trans. Systems Man Cybernet.* **3**, 172–181.

Narendra, P. M. and Fukunaga, K.: 1977, A branch and bound algorithm for feature subset selection, *IEEE Transactions on Computers* pp. 917–922.

Pei, M., Goodman, E. D., Punch, W. F. and Ding, Y.: 1995, Genetic algorithms for classification and feature extraction, *Proc. of the Classification Society Conf.*.

Quinlan, J. R.: 1983, Learning efficient classification procedures and their application to chess end games, *in* R. S. Michalski, J. G. Carbonell and T. M. Mitchell (eds), *Machine learning: An artificial intelligence approach*, Morgan Kaufmann, San Mateo, CA.

Quinlan, J. R.: 1986, Induction of decision trees, *Machine Learning* **1**.

Quinlan, J. R.: 1993, *C4.5: Programs for Machine Learning*, Morgan Kaufmann.

Rauber, T. W.: 1996, The tooldiag package. Available electronically from: `http://www.uninova.pt/~tr/home/tooldiag.html`.

Rayward-Smith, V. J., Debuse, J. C. W. and de la Iglesia, B.: 1995, Using a genetic algorithm to data mine in the financial services sector, *in* A. Macintosh and C. Cooper (eds), *Applications and Innovations in Expert Systems III*, SGES Publications, pp. 237–252.

Saitta, L. (ed.): 1996, *Proc. of the 13th Int. Conf. on Machine Learning*.

Shannon, C. E. and Weaver, W.: 1963, *The Mathematical Theory of Communication*, University of Illinois Press, Urbana.

Thrun, S., Bala, J., Bloedorn, E., Bratko, I., Cestnik, B., Cheng, J., De Jong, K., Dzeroski, S., Fahlman, S. E., Fisher, D., Hamman, R., Kaufman, K., Keller, I. Kononenko, I., Kreuziger, J., Michalski, R. S., Mitchell, T., Pachowicz, P., Reich, Y., Vafaie, H., Van de Welde, W., Wenzel, W., Wnek, J. and Zhang, J.: 1991, The MONK's problems - a performance comparison of different learning algorithms, *Carnegie Mellon University CMU-CS-91-197* .

Weiss, S. M. and Kulikowski, C. A.: 1991, *Computer systems that learn : classification and prediction methods from statistics, neural nets, machine learning and expert systems*, Morgan Kaufmann, San Francisco.

Whitney, A.: 1971, A direct method of nonparametric measurement selection, *IEEE Trans. Comput.* **20**, 1100–1103.