The Role of Domain Knowledge in Data Mining

Sarabjot S. Anand, David A. Bell and John G. Hughes
School of Information and Software Engineering
Faculty of Informatics
University of Ulster (Jordanstown)
Northern Ireland
E-Mail : ss.anand , da.bell , jg.hughes@ulst.ac.uk
Tel: 44-1232-365131 ext. 6671
Fax: 44-1232-366068

Abstract

The ideal situation for a Data Mining or Knowledge Discovery system would be for the user to be able to pose a query of the form "Give me something interesting that could be useful" and for the system to discover some useful knowledge for the user. But such a system would be unrealistic as databases in the real world are very large and so it would be too inefficient to be workable. So the role of the human within the discovery process is essential. Moreover, the measure of what is meant by "interesting to the user" is dependent on the user as well as the domain within which the Data Mining system is being used.

In this paper we discuss the use of domain knowledge within Data Mining. We define three classes of domain knowledge: Hierarchical Generalization Trees (HG-Trees), Attribute Relationship Rules (AR-rules) and Environment-Based Constraints (EBC). We discuss how each one of these types of domain knowledge is incorporated into the discovery process within the EDM (Evidential Data Mining) framework for Data Mining proposed earlier by the authors [ANAN94], and in particular within the STRIP (Strong Rule Induction in Parallel) algorithm [ANAN95] implemented within the EDM framework. We highlight the advantages of using domain knowledge within the discovery process by providing results from the application of the STRIP algorithm in the actuarial domain.
Keywords: Data Mining, Knowledge Discovery in Databases, Domain Knowledge, Evidence-based discovery

1 Introduction

Data stored in computers is growing in volume very rapidly indeed with data being collected in scientific as well as business domains. Though the collection and generation of data is on the increase (e.g. from the Human Genome Project [FASM94]), techniques for using the data in beneficial ways are slow to develop. Most of the data being collected is never analyzed and ends up on archive files, which are perhaps never to be re-opened. The realization that hidden within these masses of data is useful knowledge which can

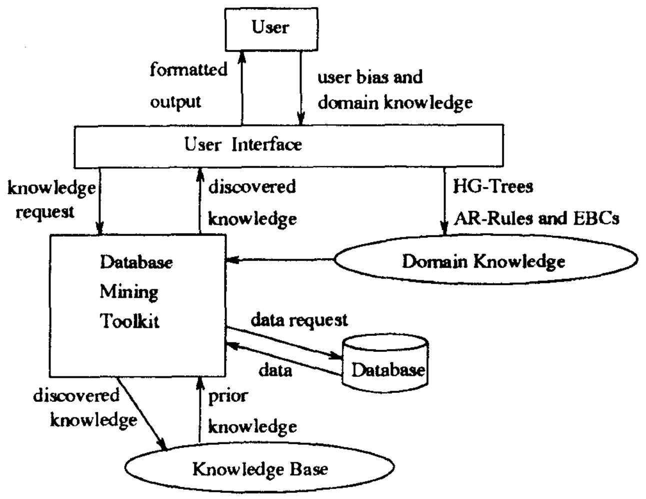

Figure 1: A Model for Data Mining or Knowledge Discovery in Databases

be of great use within the domain in which the data was collected lead to a great amount of effort and resources being channelled into Knowledge Discovery in Databases or Data Mining [FRAW91, PIAT91, PIAT93].

The ideal situation for knowledge discovery would be where the discovery process is not biased by the user in any way as this would lead to a pure form of discovery where the user isn't really aware of what may be discovered. But such a discovery system is not a viable option as the amount of knowledge that can be discovered from a data set far exceeds the amount of data. In fact this would effectively entail the use of another tool to discover knowledge, this time from the large amounts of knowledge discovered by a Data Mining tool. A more realistic discovery tool has within it a role for the human. Figure 1 shows our model for a Data Mining tool.

The user provides two kinds of information:

1. Domain Knowledge
2. Bias Information

Domain Knowledge consists of information about the data that is already available either through some other discovery process or from a domain expert. We can classify domain knowledge into three classes: Hierarchical Generalization Trees (HG-Trees), Attribute Relationship Rules (AR-rules) and Environment Based Constraints (EBC). HG-

Trees represent generalizations of attribute values that may be of interest to the user, e.g. a date field can be generalized into decades. This has two advantages: firstly, patterns involving these fields may be more visible when using the generalized attribute values, and secondly they may be more useful to the user. For example, in the actuarial domain a rule of the kind

if man is born on the 1st of October, 1956
his policy will lapse

is less useful than a rule of the type

if man is in his late 30's then his policy will lapse.

AR-rules can clearly take many forms. A good example is that of integrity constraints defined within the design of the database or information already known with high certainty to the user. Such information can be used to constrain the search space of the discovery process [BELL93]. EBCs are constraints put on the discovery process by the discovery environment. For example, in a database of spatial information one of the images may have been recorded with a very skew angle on the object. When processing the database the discovery process must take this information into account. Another EBC may be a method for measuring the interestingness of a rule discovered. We discuss these three types of domain knowledge in greater detail in section 3 and describe how each type of domain knowledge can be incorporated into the discovery process within the EDM framework for Data Mining proposed earlier by the authors [ANAN94].

Bias Information provided by the user consists of information like the attributes of interest within the data, which attributes are of interest as antecedents and which as consequents and support and uncertainty bounds requirements for rules discovered. These are refered to by Agrawal et. al. as Syntactic and Support constraints [AGRA93].

The rest of the paper is organised as follows: In section 2 we define the Evidential Data Mining (EDM) framework for Data Mining proposed earlier by the authors. In section 3 we discuss the three kinds of domain knowledge and describe, via examples, how they are incorporated into the discovery process within EDM. In section 4 we discuss different techniques, including domain knowledge, used by the STRIP (Strong Rule Induction in Parallel) algorithm for pruning the search and rule space, and section 5 illustrates how the incorporation of domain knowledge affected the knowledge discovered within an actuarial application using the STRIP Algorithm implemented within EDM.

2 EDM: The Evidential Data Mining Framework

EDM is a framework for Data Mining based on Evidence Theory. EDM consists of two main parts: a data and knowledge representation part and the discovery operators. Data stored in databases is considered to be evidence supporting difference pieces of knowledge. This data is represented in the form of mass functions as defined below:

$$m : 2^{A_1} \times 2^{A_2} \times \ldots 2^{A_n} \longrightarrow [0,1]$$

where the $A_i s$ are the frames of discernment [1] of each of the attributes in the table e.g. the tuple

$$< Morrison, 14, IndiaSt., Belfast, NULL >$$

is represented as

$$m < \{Morrison\}, \{14\}, \{IndiaSt.\}, \{Belfast\}, A_5 >= s_i$$

where s_i is the ratio of the number of occurrences of the tuple to the total number of tuples.

The example mass function above highlights one of the advantages of using an Evidence Theory approach to Data Mining. Evidence Theory allows an expression of ignorance which is an intuitive way of treating NULL or missing values in databases. In the above example the missing value for the fifth attribute in the tuple is represented as the whole frame of discernment for the attribute i.e. A_5. By this ignorance representation mechanism, Evidence Theory provides an accurate representation of the data stored in databases. As more evidence is collected the discovery process transfers its present belief associated with ignorance to rules supported by the evidence. For a discussion of the advantages of using Evidence Theory for Data Mining the reader in referred to [ANAN94]

A rule discovered within the framework has three values associated with it - representing the uncertainty, support and interestingness, using a generalization of Piatetsky-Shapio's measure for interestingness [PIAT91a], of the rule. These values allow the pruning of the rule space thereby protecting the user from being inundated with rules.

Rules are represented in the form of rule mass functions defined below:

$$M : 2^A \times 2^C \longrightarrow [0,1] \times [0,1] \times [-1,1]$$

satisfying

1. $M(< Y, \emptyset >) = (0,0,0), \forall Y \subseteq A$
2. $\sum_{X \subseteq C} M[1](< Y, X >) = 1, \forall Y \subseteq A$
   $\sum_{Y \subseteq A, X \subseteq C} M[2](< Y, X >) = 1$
   $0 \leq \sum_{X \subseteq C} M[3](< Y, X >) \leq n(Y)/n$

where, A is the frame of discernment of Antecedents
and C is the frame of discernment of Consequents
and n(Y) is the number of tuples satisfying the antecedent and n is the number of tuples in the data from which the rule has been discovered
and M[1], M[2] and M[3] represent the uncertainty, support and interestingness associated with the rule, respectively, that are the components of the 3-tuple that M maps rules to.

The M[1] component corresponds directly with the definition of the mass function within Dempster-Shafer Evidence Theory.

The discovery process within EDM consists of a series of operators on the mass functions and rule mass functions defined above. The operators are classified into: Combination, Domain, Induction, Statistical and Updating operators depending on the function performed by each operator. The Induction operator collates evidence supporting different rules from an individual sample. The Combination operators combine evidence supporting different rules collected (using Induction operators) from different samples of the database to give an overall belief in each of the rules supported by the database. The domain operators are the topic of our discussion in this paper. Each Domain operator incorporates a different type of domain knowledge as we will see in section 3 below. These include the conventional Evidence Theory operators: Coarsening and Discounting. The Coarsening operator incorporates HG-Trees type knowledge, a new operator called the ar_filter operator incorporates AR-rules and the Discounting operator incorporates EBC knowledge. The role of the Update operators is to keep the discovered knowledge and the data consistent after updates have

[1] A frame of discernment here is a mutually exclusive and exhaustive set of propositions of interest (although it can be generalised to any boolean algebra)

been made to the data. The Statistical operators provide statistical functionality required within the discovery process.

3 Domain Knowledge in EDM

Domain Knowledge is a useful way of constraining or pruning the search space for a discovery process, enhancing the performance of the system. In Data Mining users are often not aware of what they are looking for, and therefore it is important for a Data Mining system to be able to function independently of any help from the user. However, if the user does have some background information, i.e Domain Knowledge, the discovery system should be able to use it.

The Domain Operators in EDM allow the incorporation of Domain Knowledge into the discovery process. Within EDM, at present, we use three types of Domain Knowledge:

1. Attribute Relationship (AR) rules
2. Hierarchical Generalization (HG) trees
3. Environment-Based Constraints (EBC)

Domain Knowledge works in two ways within a discovery system. Firstly, domain knowledge can be used to constrain the search space and the rule space of the discovery process e.g. AR-rules (discussed in further detail in section 3.1). Secondly, domain knowledge can make patterns in the data more visible [HAN94] e.g. using HG-Trees we can generalize the data thus bringing out patterns that were not visible at less coarse grained data levels (see section 3.2).

Mallen et. al. [MALL95] use two types of Domain Knowledge in their Knowledge Discovery system, CUPID: Generalization Trees (that correspond to our HG Trees) and Intentional or Constructed Attributes. In EDM Intentional Attributes are assumed to be defined in the virtual data view [MAGI95] and are not considered as part of Domain Knowledge.

In this section we discuss what we consider as constituting Domain Knowledge and how we incorporate it into the discovery process.

3.1 Attribute Relationships Rules (AR-rules)

This type of domain knowledge consists of rules, referred to as AR-rules, that the domain expert provides about the data or rules that have been discovered using another discovery techniques. Within databases we already have some AR-rules in the form of integrity constraints that were defined at the design stage of the database. These can be incorporated within the discovery process using techniques discussed in this section. AR-rules are also available from a number of other sources. For example, in a personnel database the domain expert may provide the following domain knowledge

if (income > 45000*) then*
job = {Manager, Managing Director, Executive Director}
or
if (Car_Allowance = Yes) then
job = {Salesman, Branch Director}
or
job = {System Analyst, Programmer, IT Manager}

The first AR-rule specifies that the job specification for anyone with an income of more than 45000 pounds per annum is either a Manager, Managing Director or Executive Director. It may be possible that there are people with other job specifications that receive a salary of over 45000 but they are not of interest and are pruned off using the *ar_filter* operator as shown below. The second AR-rule states that if a person is receiving a car allowance then his/ her job specification must be either Salesman or Branch Manager. Both these rules specify a subset of the domain of the attribute *job* that is of interest whenever the Antecedent of the rule is true. The third AR-rule specifies a subset of the domain of attribute *job* which is of interest no matter what the Antecedent may be. This last AR-rule is a generalization of the Syntactic Constraint suggested by Agrawal et. al. [AGRA93] as not only can we specify attributes of interest but also subsets of the domain of the attribute of interest.

Such Domain Knowledge can be represented in the form of rule mass functions (see section 2) as follows:

$m_1(<\{income > 45000\}>,<\{Manager, Managing\ Director, Executive\ Director\}>) = (1, 0, 0)$
and
$m_2(<\{Car\_Allowance = Yes\}>,<\{Salesman, Branch\ Director\}>) = (1, 0, 0)$
and
$m_3(<>,<\{System\ Analyst, Programmer, IT\ Manager\}) = (1, 0, 0)$

We now discuss how AR-rules of the kind described above can be incorporated within the discovery process. A new domain operator known as the *ar_filter* operator, δ_m (m is a rule mass function representing the AR-rules to be incorporated into the discovery process), is used to incorporate AR-rules into the discovery process. The *ar_filter* operator is a unary operator on a mass function and takes as its parameter a rule mass function representation of the AR-rules. For all components of the mass function being operated upon, the *ar_filter* operator intersects the consequent fields with the corresponding fields in the mass function component with the same antecedent field values.

Let us consider a relation in a Housing database *R(Name, Age, Rent, St_address, Post_Code, No_of_Bedrooms, No_of_Inhabitants)*.

Suppose an expert in the local housing agency provides the following domain information:

if post_code = BT7 then No_of_Bedrooms = 3, 4 or 5
if post_code = BT9 then No_of_Bedrooms = 1 or 2
if no_of_inhabitants > 6 *then No_of_Bedrooms = 4, 5, 6 or 7*
Post_Codes of Interest are BT7, BT9, BT11 and BT1

This domain information can be stored in the form of rule mass functions as follows:

$m_1(<\{BT7\}>,<\{3,4,5\}>) = (1,0,0)$
$m_1(<\{BT9\}>,<\{1,2\}>) = (1,0,0)$
$m_2(<\{i|(i > 6 \wedge i \in \Theta_{No\_Of\_Inhabitants})\}>,<\{4, 5, 6, 7\}>) = (1, 0, 0)$
$m_3(<>,<\{BT7, BT9, BT11, BT1\}>) = (1,0,0)$

Given an Antecedent Set a rule mass function is defined as described in section 2. In the domain knowledge above the first two rules have the same Antecedent set and, therefore, form components of the same rule mass function. The third and fourth pieces of domain knowledge do not have the same Antecedent specification and therefore form separate rule mass functions. Based on the three rule mass functions defined above we can define three *ar_filter* operators $\delta_{m1}, \delta_{m2}, \delta_{m3}$.

Now suppose we have the following mass function from the Housing database:

$m(\{Morris\}, \{34\}, \{250\}, \{4\ Canterbury\ Street\}, \{BT7\}, \Theta_{No\_of\_Bedrooms}, \{7\}) = .2$
$m(\{Harvard\},\{36\},\{150\},\{15\ Durham\ Street\}, \{BT3\}, \{3\}, \{4\}) = .2$
$m(\{O'Malley\},\{22\},\{100\},\{6\ Falls\ Road\}, \{BT11\}, \Theta_{No\_of\_Bedrooms}, \{7\}) = .2$

m({Reid}, {45}, {175}, {4 Newtownards Road}, {BT9}, {4}, {8}) = .2

m({Croft},{78},{300}, {14 Cairo Street}, {BT6}, {4}, {6}) = .2

Applying the *ar_filter* operator δ_{m1}, the new mass function has the 1st component changed to

δ_{m_1} *m({Morris}, {34}, {250}, {4 Canterbury Street}, {BT7}, {3,4,5}, {7}) = .2*

Now applying δ_{m2} we get[2]

$\delta_{m_2}\delta_{m_1}$ *m({Morris}, {34}, {250}, {4 Canterbury Street}, {BT7}, {4,5}, {7}) = .2*

$\delta_{m_2}\delta_{m_1}$ *m({O'Malley}, {22}, {100}, {6 Falls Road}, {BT11}, {4,5,6,7}, {7}) = .2*

The other components remain unchanged. Thus, using AR-rules the search space is pruned from all possible values of the *No_of_Bedrooms* attribute to the values $\{4, 5\}$ and $\{4, 5, 6, 7\}$ for the 1st and 3rd components, respectively, of the mass function. Notice that a transition has taken place from total ignorance, $\Theta_{No\_of\_Bedrooms}$, to some knowledge $\{4, 5\}$ and $\{4, 5, 6, 7\}$ respectively.

Now applying δ_{m_3} we get

$\delta_{m_3}\delta_{m_2}\delta_{m_1}$ *m({Morris}, {34}, {250}, {4 Canterbury Street}, {BT7}, {4,5}, {7}) = .2*

$\delta_{m_3}\delta_{m_2}\delta_{m_1}$ *m({Harvard}, {36}, {150}, {15 Durham Street}, ∅, {3}, {4}) = .2*

$\delta_{m_3}\delta_{m_2}\delta_{m_1}$ *m({O'Malley}, {22}, {100}, {6 Falls Road}, {BT11}, {4,5,6,7}, {7}) = .2*

$\delta_{m_3}\delta_{m_2}\delta_{m_1}$ *m({Reid}, {45}, {175}, {4 Newtownards Road}, {BT9}, {4}, {8}) = .2*

$\delta_{m_3}\delta_{m_2}\delta_{m_1}$ *m({Croft}, {78}, {300}, {14 Cairo Street}, ∅, {4}, {6}) = .2*

The *ar_filter* operator is commutative as the set operation of intersection is commutative. However, AR-rules of the type represented by m_3 must be applied after all AR-Rules with a specific Antecedent specification have been applied.

Now the mass function can be used as evidence to discover rules by the STRIP algorithm.

3.2 Hierarchical Generalization Trees (HG- Trees)

Very often data stored in databases needs to be generalized in order to get meaningful or useful information [HAN94]. For example, finding rules that associate the date of birth of a customer with the car he drives is less useful than finding rules that associate an age bracket with a car type.

An HG-Tree can have a number of levels of coarseness (each represented as a different level in the HG-Tree). Figure 2 shows an example HG-Tree.

Such Domain Knowledge is stored and incorporated into the discovery process by using evidential coarsening operators. A Coarsening operator [GUAN91, GUAN92] is defined as the mapping below:

$$coar : 2^{\Theta} \longrightarrow 2^{\Omega}$$

where Θ is the frame of discernment at a lower level of the HG-Tree and Ω is the frame of discernment at a higher level of the HG-Tree

Thus for the example in Figure 2 one level of the coarsening operator would be as shown below:

$coar(\{BT3, BT4, BT5, BT6\}) = East$

$coar(\{BT9, BT10\}) = Malone$

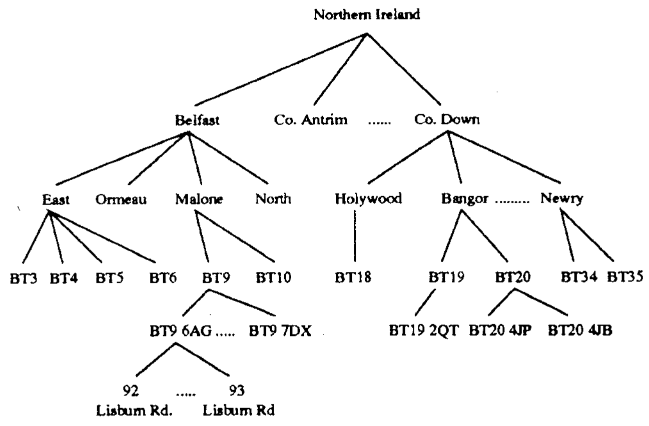

Figure 2: Example HG-Tree

$coar(\{BT7, BT8\}) = Ormeau$

$coar(\{BT14, BT15\}) = North$

For each pair of levels of coarseness a coarsening operator can be defined as above but care must be taken during the generalization process not to over-generalize, as this causes unacceptable loss of information. Thus, the construction of HG-Trees is a very important part of the discovery process. Han et. al. have shown how HG-Trees can be generated dynamically [HAN94a].

The coarsening operator is a unary operator on mass functions. It replaces the data values from a lower level of the HG-Tree, appearing in the mass function, with the corresponding value at a higher level of the HG-Tree.

3.3 Environment-Based Constraints (EBC)

When combining evidence from different sources a user might want to bias the result towards one of the pieces of evidence, perhaps due to the user having greater faith in its source. This could be due to information available to the domain expert that is not available from the data. For example, in a Spatial Database application when combining evidence of the existence of volcanoes on Mars it would clearly be desirable to discount the evidence collected from images of lower resolution or images taken from a more skew angle [BELL94].

The Discounting operator [GUAN93] allows you to specify degrees of confidence in the different sources of evidence and incorporates it into the reasoning framework. Given a degree of confidence, α, the Discounting operator transfers some of the belief in each of the propositions within the mass function to ignorance. The new mass function obtained is defined on the same frame of discernment as the first mass function - the only difference is that the new mass function associates a certain amount of belief with ignorance and its belief in each of the propositions supported by the earlier mass function is reduced by a factor determined by the degree of confidence, α. The sources of evidence may be different domain experts or different geographically distributed, heterogeneous databases. The common knowledge representation used within EDM for evidence from different sources means that we can deal with such evidence in a uniform way.

For example, let us consider a geographically distributed database from which knowledge is being discovered. Information that the data in the database at location 1 is not

[2]Notice the fact that the (1,0,0) mass simplifies our illustration here. In general, for AR-rules, the mass (x,0,0) can be associated with rules, where x lies in the interval [0,1].

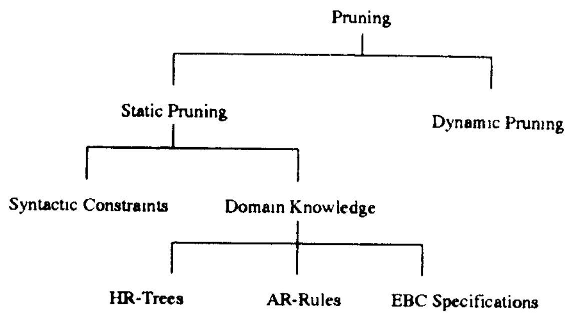

Figure 3: Classification of Search Space Pruning Techniques

as reliable as the data stored at location 2 needs to be incorporated into the discovery process to get correct results. Now suppose we can quantify the reliability of the data at location 1 as being 80% as reliable as the data at location 2, the degree of confidence in the rule mass function from location 1 can be set to 0.8 and the rule mass function can be discounted based on that degree of confidence. After discounting the rule mass functions from location 1 the discounted rule mass function can be combined with the rule mass function from location 2.

Such factors are referred to as Environment-Based Constraints (EBC) within EDM. While AR-rules and HG-Trees are useful in pruning the search space and rule space, EBCs are used to discover more accurate knowledge.

4 Pruning Strategies in the STRIP Algorithm

In a Data Mining system there are two different kinds of pruning:

1. Pruning of the Search Space
2. Pruning of the Rule Space

4.1 Pruning the Search Space

Figure 3 shows our classification of methods for pruning the search space. There are two main classes of methods for pruning the search space: Static Pruning and Dynamic Pruning. Static Pruning reduces the search space before the discovery algorithm is applied on the data. The static pruning techniques include incorporation of Domain Knowledge - AR-rules and HG-Trees and Syntactic Constraints [AGRA93]. We have already discussed in the previous section how Domain Knowledge can be used to prune the search space. Defining syntactic constraints reduces the dimensionality of the search space by selecting a subset of attributes of interest from the set of all attributes. This is very important as most discovery algorithms can achieve a linear scalability with respect to number of tuples but are non-linear with respect to the number of attributes in the data set. Dynamic Pruning takes place during the execution of the discovery algorithm. As knowledge is discovered, based on the discovered knowledge parts of the search space that have been covered or already disregarded by the algorithm are pruned.

4.2 Pruning the Rule Space

The rule space is constrained by associating different measures with a rule and then setting minimum acceptable values for them. In section 2 we specified three different measures that are commonly associated with rules. These were the uncertainty, support and interestingness or utility of the rule. Minimum and maximum limits on uncertainty, support and interestingness can be specified and used to prune the rule space. Also specifying AR-rules and antecedent attributes of interest reduces the rule space.

5 The Effect of Domain Knowledge on STRIP

In this section we discuss the effect of using domain knowledge in a Data Mining application in the actuarial domain. The data set had 28 attributes of which we picked 9 attributes that were of particular interest. Table 1 shows the attributes of interest along with the number of distinct values for each of the attributes.

Attribute#	Attribute Name	No. of Distinct Values
8	Ren_ind	2
11	Cover	3
14	Car_group	22
15	Car_age	33
16	Age_of_driver	85
17	Sex	2
18	Driver	6
21	Installments	2
22	Ncdren	10

Table 1: Attributes of Interest

Three of the attributes - Age of the driver, Age of the car and Car group - had HG-Tree type domain knowledge available. Table 2 shows the domain knowledge used by STRIP for the particular application.

```
Age_of_driver User_interval
START
gt 16 le 25 young
gt 25 le 50 middle
gt 50 le 75 old
gt 75 le 100 grand
END
Car_age User_interval
START
ge 0 le 5 New
gt 5 le 10 Middle
gt 10 le 35 Old
END
Car_group User_interval
START
gt 0 le 7 Small
gt 7 le 14 Medium
gt 14 le 22 Large
END
```

Table 2: Domain Knowledge

Apart from the domain knowledge, the Support Constraint was specified at 0.1

As mentioned earlier, domain knowledge is useful in the discovery process in two ways. Firstly, domain knowledge of the HG-Tree type can be useful in generalizing attribute values and bringing out patterns that were not visible otherwise. Secondly, domain knowledge can be used to constrain the search space as well as the rule space. We first show how using HG-Tree type domain knowledge can make patterns more visible.

The first run of the STRIP algorithm was without using any domain knowledge, and with discovered rules being constrained to those with antecedent attributes Attibute#11, Attribute#11 and 17, Attribute#11,17 and 18 and Attribute# 11, 17, 18, 21. We found the following results:

5 rules were discovered with Ren_ind in the consequent

0 rules were discovered with Age_of_driver in the consequent

0 rules were discovered with Car_age in the consequent

0 rules were discovered with Car_group in the consequent

2 rules were discovered with Ncdren in the consequent

The domain knowledge pertaining to the Car_age attribute (see Table 2) was then applied and we found that now there was 1 rule discovered with attribute Car_age in the Consequent. On using domain knowledge for the Age_of_driver and Car_group 2 rules with Car_group in the Consequent and 3 rules with Age_of_driver in the Consequent were discovered. Table 3 shows some example rules found by the STRIP algorithm.

if Cover = 1 and Sex = 79 then mta = 78 and Car_group = Medium and Age_of_driver = middle and Installments = 78 and Ncdren = 5 and proten = 78 and ncdold = 5 and prevprem = 0 with uncertainty = 1.000000 support = 0.001000 interestingness = 0.000999
if Cover = 3 and Sex = 70 and Driver = 83 then currsur = 0 and Age_of_driver = middle and proten = 78 and prevsur = 0 and prevprem = 0 with uncertainty = 1.000000 support = 0.004500 interestingness = 0.004306
if Cover = 1 and Sex = 70 and Driver = 78 then mta = 89 and currsur = 0 and Car_group = Small and Age_of_driver = middle and proten = 78 and prevsur = 0 and prevprem = 0 with uncertainty = 1.000000 support = 0.002500 interestingness = 0.002491
if Cover = 1 and Sex = 70 and Driver = 69 then Ren_ind = 82 and prevprem = 0 with uncertainty = 1.000000 support = 0.002500 interestingness = 0.002494

Table 3: Examples of Rules Discovered by STRIP using Domain Knowledge

HG-Tree type domain knowledge can also be used to constrain the rule space. The domain knowledge shown in Table 2 can be used to reduce the number of rules discovered. We used the same data and domain knowledge in the above discovery by STRIP but changed the syntactic constraint to Antecedent attributes Age_of_driver, Car_group and Car_age. This reduced the number of possible rules to be discovered from 67251 rules to 79 rules. An example rule is given in Table 4. Without using domain knowledge we would be swamped with rules with a small support.

if Car_group = Small and Car_age = New and Age_of_driver = young then age_of_pol = 1 and Ren_ind = 76 and currsur = 0 and Installments = 78 and proten = 78 and prevprem = 0 with uncertainty = 1.000000 support = 0.002500 interestingness = 0.002493

Table 4: Constraining the Rule Space using Domain Knowledge

The rule and search space can be constrained by using domain knowledge in the form of AR-rules as discussed in section 3.1.

6 Conclusions

In this paper we have discussed the importance of the role of the human in the discovery process. In particular we have discussed the advantage of using domain knowledge within the discovery process. We classified domain knowledge into Hierarchical Generalization Trees (HG-Trees), Attribute Relationship Rules (AR-rules) and Environment Based Constraints (EBC). Domain Knowledge can affect the discovery process within a Data Mining system in two ways. Firstly, it can make patterns more visible by generalizing the attribute values, and secondly, it can constrain the search space as well as the rule space.

We have discussed how each class of domain knowledge is incorporated into the discovery process within Evidential Data Mining (EDM) framework using different domain operators. We have also discussed the techniques other than domain knowledge used by the STRIP algorithm to prune the search space and the rule space.

Using an illustrative example in an actuarial domain, we showed how knowledge discovered by the application of the STRIP algorithm was affected by the use of domain knowledge provided in the form of HG-Trees. We also indicated how the use of AR-rules could further constrain the search space as well as the rule space.

7 Acknowledgements

The authors wish to thank the Northern Ireland Industrial Research and Technology Unit for funding the work presented in this paper. The Northern Ireland Housing Executive, ICL Ltd. and Kainos Software Ltd. are the industrial partners in the project.

8 References

[AGRA93] R . Agrawal, T . Imielinski, A . Swami Database Mining : A Performance Perspective IEEE Transactions on Knowledge and Data Engineering, Special Issue on Learning and Discovery in Knowledge - Based Systems, 1993.

[ANAN94] S.S. Anand, D.A. Bell, J.G. Hughes A General Framework for Database Mining Based on Evidence Theory, Internal Report, School of Information and Software Engineering, University of Ulster, November, 1994.

[ANAN95] S. S. Anand, C. M. Shapcott, D. A. Bell, J. G. Hughes Evidential Techniques for Parallel Database Mining, Proc. of the Conference on High Performance Computing and Networking, Springer-Verlag, May, 1995.

[BELL93] D. A. Bell From Data Properties to Evidence, IEEE Transactions on Knowledge and Data Engineering, Vol. 5, No. 6, Special Issue on Learning and Discovery in Knowledge - Based Databases, December, 1993.

[BELL94] D.A. Bell, S.S. Anand, C.M. Shapcott Database Mining in Spatial Databases, International Workshop on Spatio-Temporal Databases, 1994.

[FASM94] K. H. Fasman, A. J. Cuticchia, D. T. Kingsbury The GDB Human Genome Data Base anno 1994, Nucleic Acids Research, Vol. 22, No. 17, Pg. 3462 - 2469, 1994.

[FRAW91] W. J. Frawley, G. Piatetsky-Shapiro, C. J. Matheus Knowledge Discovery in Databases: An Overview, Knowledge Discovery in Databases Pg. 1 - 27 AAAI/MIT Press 1991.

[GUAN91] J.W. Guan, D.A. Bell Evidence Theory and its Applications vol. 1 North-Holland, 1991.

[GUAN92] J.W. Guan, D.A. Bell Evidence Theory and its Applications vol. 2 North-Holland, 1992.

[GUAN93] J.W. Guan, D.A. Bell Discounting and Combination Operations in Evidential Reasoning, Proc. of the Conference on Uncertainty in Artificial Intelligence, 1993.

[HAN94] J. Han Towards Efficient Induction Mechanisms in Database Systems, Theoretical Computer Science 133, Elsevier, Pg. 361 - 385, 1994.

[HAN94a] J. Han, Yongjian Fu Dynamic Generation and Refinement of Concept Hierarchies for Knowledge Discovery in Databases, AAAI Workshop in Knowledge Discovery in Databases, 1994.

[MAGI95] I. C. Magill, M. G. Tan, S. S. Anand, D. A. Bell, J. G. Hughes Database Mining in the Northern Ireland Housing Executive, IEE Colloquium on Knowledge Discovery in Databases, February, 1995.

[MALL94] J. Mallen, M. A. Bramer Utilising Domain Knowledge in Inductive Knowledge Discovery, IEE Colloquium on Knowledge Discovery in Databases, February, 1995.

[PIAT91] G. Piatetsky-Shapiro, W. J. Frawley (eds.) Knowledge Discovery in Databases AAAI/MIT Press 1991.

[PIAT91a] G. Piatetsky-Shapiro Discovery, Analysis and Presentation of Strong Rules, Knowledge Discovery in Databases Pg. 229 - 248 AAAI/MIT Press 1991.

[PIAT93] G. Piatetsky-Shapiro, W. J. Frawley (ed.) Working Notes of the AAAI Workshop on Knowledge Discovery in Databases, Washington D. C., 1993.