Deriving Heuristic Rules from Facts

by Hans van Thiel

"All classification, whether artificial or natural, is the arrangement of objects according to ideas."
The 19th Century American Philosopher and Logician Charles Sanders Peirce

Version 1.0, January 2007

Summary
A predecessor to this paper describes a reduction algorithm to extract distinguishing features from sets of heuristic rules. It also contains a listing of a prototype implementation in the functional programming language Haskell. This second paper builds on the first one, although it can be read separately.
The first section defines a fact model and a rule model in terms of partitions of a set. The second section treats the reduction algorithm and proves it finds all possible shortest rules. Therefore it produces a normal form representation of the rule model. The third section shows how to get the partial order, if there is one, of reduced antecedents through abduction. Entailment and overlap of underlying sets may allow further rule simplification. The fourth section treats additions to a fact model (empirical induction) and shows that reduction also works when some rules are ambiguous. The effect of new rules on a reduced normal form is summarized in two general propositions. Through semantic extrapolation , soundness and completeness of reduced rules are defined.
All examples use a data table from a classic paper by J.R. Quinlan, and the reductions were obtained using the computer program listed in the earlier paper.

key terms: categorical data analysis, multivariate data analysis, data mining, rule based knowledge, machine learning

Author Information:
E: hthiel.char@zonnet.nl
info@j-van-thiel.speedlinq.nl
W: http://j-van-thiel.speedlinq.nl

T: +31 20 6247137
Passeerdersstraat 76
1016 XZ Amsterdam
The Netherlands

Introduction

An earlier paper [THIEL] introduced an algorithm to simplify a set of rules by finding the shortest ones that could predict some consequence. It consists of a detailed step by step mathematical description and a prototype program listing in the functional programming language Haskell.

This paper builds on its predecessor by placing the algoritm in context, precisizing its scope and limits, and showing how it can be used for data mining, machine learning and logical analysis of nominal data in general. Though this paper uses the results of the first one, it can be read independently. The aim of this paper is to demonstrate a method for analyzing and optimizing rules with nominal data, starting from the ground up. All examples use the well known table from the classical paper by [Quinlan] and the reduction rules were derived with the Haskell computer program from [THIEL].

The starting point is a table like table 1 [QUINLAN]. This is not a relational table, but a representation of a set of elements where each element has exactly one value for each defined attribute. An attribute may be a binary predicate, where the value is just the indication whether the element has that predicate or not, or an attribute may have more than two possible values. (1) An attribute defines a partition over the set T, and the table represents the conjunction of those partitions.
A fact model is just a set T of elements and the conjunction of the defined attribute partitions.
Heuristic rules are rules which have no particular foundation in a (scientific) theory, but which are based solely on the observation of general patterns. Deriving rules from facts is not an entirely automatic process. From table 1, for example, it seems obvious that Fishing is the attribute to be predicted, but Weather, Temperature, Humidity or Windy could also be selected. In a logical sense there is no reason why one attribute should be the consequent and the others the antecedents, the names are just labels.
When the user has chosen the attribute of interest the table turns into a rule model, as in table 2 and table 8. A rule model can easily be verified by checking whether the antecedents for the selected consequent are all unique. If so, then the antecedent rows in the table will all predict just one consequent value. If not, then there is ambiguity. But it turns out the reduction algoritm also works with ambiguity, as is shown in the section on empirical induction.

A rule model can be transformed into another rule model with shorter rules than the original one through a reduction algoritm. It turns out this algoritm derives all possible shortes rules, and therefore the result can be considered a normal form of the original rule model.
First it is shown that any possible rule, which can either be verified or falsified in the rule model, consists of a sublist of an original list of attribute values. The reduction algoritm consists of three steps, hypothesis, falsification, and verification.
First, for each consequent attribute-value, the table is split into rules which support it and rules which do not. The starting assumption is that each supporting attribute-value will predict the consequent value on its own, and this consequent can be predicted by the or-list of those values.
Clearly this will (in general) be false and the next step consists of matching this list with each rule that does not support the consequent. For each of these rules, the values in the hypothesis that do not match that rule, will remain. The result of this falsification step is an and-list of non-matching or-lists. Using the set theoretic laws of intersection and union, this is transformed into an or-list of and-lists. This result is the collection of all rules for the consequent value, which are not contradictory. They do not occur in any of the rules which imply another consequent value.
The third step consists of a verification of those non-contradictory rules, by selecting only those and-lists, which are sublists of an original supporting rule. This is needed to reject rules which can neither be verified, nor falsified in the rule model.
Because it can be proved the reduction algoritm derives not only the shortest possible rules, but all of them, the result of the reduction is called the 'reduced normal form' of the rule model.

Rules in reduced normal form have a semantics which can easily be determined by taking the superlists of the antecedent lists. This shows that these antecedents may be partially ordered, through entailment of their undelying sets. Deriving rules between antecedents of the same consequent is called 'abduction', following Charles Sanders Peirce, who introduced the term in that sense. Abduction may be interesting in itself, because it can be used to find new rule patterns, but that is not pursued further in this paper. The partial ordering of antecedents, however, may allow further simplification of the reduced rule model. It is shown that the reduced normal form of the example from [QUINLAN] contains the same rules he discovered and that those rules cover all original rules. But other representations of the same original rule model are also possible.

The practical worth of heuristic rules is how well they fit new facts. The addition of new facts to a fact model is easily transmuted into the addition of new rules to a rule model, and this is called 'emperical induction' (3). Induction is examined with the same fact model as in the previous sections, but a different rule model. This includes an antecedent that implies either of the two consequent attribute-values (ambiguity). It turns out the reduction algoritm also works on ambiguous rule models but, as is to be expected, ambiguous rules do have an effect.
Addition of a new rule may or may not result in the addition of a new reduced rule for the same consequent attribute-value. The effect on the rules for other values than the consequent may be, be a replacement of a reduced rule by one or more longer ones, refutation of the rule altogether, or no change at all. These possible effects are summarized in two general propositions.
Though the number of possible cases may be infinite, the number of possible facts is finite, as is the number of possible rules. Each potential rule can therefore be uniquely identified and using this the semantics for a rule model in reduced normal form can be stated. It consists of all superlists of the reduced rules for each consequent attribute-value.
Using these extrapolated semantics, a group of rules for an attribute-value is defined to be sound if none of the superlists can ever occur in a rule for another attribute-value. It is defined to be complete if any additional new rule for that attribute-value is already part of the extrapolated semantics. Then, for a binary attribute, if the group of reduced rule for a predicate is sound and complete, then the group of rules for its complement is sound.
Of course soundness or completeness can never be certain for heuristic trules, because any possible fact can always turn up in the future. But these concepts and their extrapolated semantics can guide the (empirical) testing of reduced rule models.

Definitions and Terminology

Let T be a set with a number of partitions P, Q, R... defined over T. Let P1, P2... be the blocks of partition P and so on. Then the names of the partitions are attributes and the names of the blocks are attribute-values. We will use 'predicate' 'and attribute-value' as interchangeable synonyms.
Each attribute denotes the set T and each attribute-value denotes a subset of T. Therefore the logic of attribute-values (predicates) in T reduces to the (set) properties of the denoted subsets of T.

To distinguish between names of sets and the denoted sets we will use 'list', 'sublist', 'superlist,' and-list' and 'or-list' when we are talking about sets of names and operations on sets of names. The operations 'and', 'or' correspond to intersection, union of the sets denoted by their names. When referring to a list which is a row in a table, we will sometimes just use 'row'.

A 'rule' or 'implication' in T is a statement that a set in T is a subset of a set in T. The entailed set is the 'antecedent' set and the entailing set is the 'consequent' set of the rule. Either set is denoted by a list of predicates, possibly with 'and' and 'or' operators.

The conjunction of two partitions is the partition formed by the intersections of their blocks, and the disjunction of two partitions is the partition formed by the union of the blocks (of the two partitions).
A partition is a refinement of another partition if all blocks of the first are subsets of blocks of the second. A group of attributes over T 'implies' or 'predicts' an attribute over T if the conjunction of the denoted 'predicting' partitions is a refinement of the 'predicted' partition.

I: Models for Facts and Rules

A: The Fact Model

We distinguish between facts and rules. The model for facts is a set T with a number of attributes over T. Each element of T has exactly one value of each attribute and, moreover, this value is known. Then each attribute defines a partition over T and the conjunction of all partitions defines a table, whith each row a distinct list of attribute-values. An example is table 1, taken from QUINLAN.

Weather	Temperature	Humidity	Windy	Fishing
sunny	hot	high	no	bad
sunny	hot	high	yes	bad
cloudy	hot	high	no	good
rain	mild	high	no	good
rain	cool	normal	no	good
rain	cool	normal	yes	bad
cloudy	cool	normal	yes	good
sunny	mild	high	no	bad
sunny	cool	normal	no	good
rain	mild	normal	no	good
sunny	mild	normal	yes	good
cloudy	mild	high	yes	good
cloudy	hot	normal	no	good
rain	mild	high	yes	bad

Table 1: A fact model of a set with attributes. The 5 columns contain the attribute-values and the 14 rows are the and-lists of those attribute-values which have been observed. The number of possible rows is 72, but most of these have not been observed.

The set T can be infinite, but the fact model is determined by the attributes and their sizes (numbers of values). The number of possible facts is the product of the attribute sizes. For example, the 5 attributes of table 1 have 3, 3, 2, 2, and 2 values, so the number of possible facts is 72. Because the number of possible facts is always finite, each possible fact may be uniquely identified, for example, by tagging it with an integer. An actual fact model, like table 1, will denote a subset of all the possible facts.

Each row in an table may denote more than one element of the set T, and therefore have a frequency count associated with it. In our fact model, however, we just look at the presence or absence of a factual observation, and consider the number of times it occurred irrelevant.

B: The Rule Model

Stating a rule model consists of selecting a consequent attribute in a fact model. A rule model can also be represented by a table, with each antecedent list next to its consequent value. Selecting Fishing as the consequent results in the rule model of table 2, while selecting Windy results in table 8. The number of rule models which can be obtained from one fact model is the the number of defined attributes.

An actual rule model can be verified by checking whether the conjunction of antecedent attributes is a refinement of the consequent attribute. This is the case if all distinct attribute-value lists predict exactly one consequent attribute-value, as in table 2.

Weather	Temperature	Humidity	Windy	Fishing
sunny	hot	high	no	bad
sunny	hot	high	yes	
rain	cool	normal	yes	
sunny	mild	high	no	
rain	mild	high	yes	
cloudy	hot	high	no	good
rain	mild	high	no	
rain	cool	normal	no	
cloudy	cool	normal	yes	
sunny	cool	normal	no	
rain	mild	normal	no	
sunny	mild	normal	yes	
cloudy	mild	high	yes	
cloudy	hot	normal	no	

Table 2: A rule model of a set with attributes. It can be checked through observation that the conjuction of the antecedent attributes is a refinement of the consequent. Each distinct antecedent list implies either bad or good fishing.

However, a rule model may be ambiguous, as table 8 demonstrates. When Windy has been selected as consequent attribute there is an antecedent with two outcomes. If a table row should denote only one element of T this would be a contradiction, but if there are more it is an ambiguity. Nevertheless, now the conjunction of antecedent attributes is not a refinement of the consequent. Ambiguity is treated in section IV.

The number of rules in a rule model is finite. The number of possible different antecedents is the product of the number of values for each attribute. So, for table 2 this is the product of 3,3, 2, 2 which is 36.
If we allow for ambiguity each antecedent can possibly imply each consequent value, so the number of possible rules is 72. Allowing for ambiguity, the number of rules is equal to the number of facts. If we do not allow ambiguity, each possible

antecedent can have only one consequent value, and so the number of possible rules is equal to the number of possible different antecedents. This is equal to the number of possible facts divided by the number of values of the consequent attribute.

II: Reduction and Normal Form

A: Reduction

Each rule in a rule model can be written as an 'and-list' of predicates implying a consequent predicate. All the rules together for a consequent can be written as an 'or-list' of these antecedent 'and-lists'. This compounded list denotes the union of the intersection of the subsets of T denoted by the relevant attribute-values.
In table 2 it can be observed that some of the 'and-lists' (all, actually) contain redundant information.
We see, for example, that predicate Weather-cloudy implies Fishing-good, because this attribute-value does not occur in any of the rules for Fishing-bad.
We can also observe that the predicate Weather-good denotes the union of all the subsets which are denoted by the and-lists containing this predicate. A sublist of an 'and-list' denotes a superset of the set denoted by the full and-list.

A refinement of an attribute (the consequent) can be rewritten as an or-list of and-lists, where each of these and-lists is the shortest one possible and still implies a consequent attribute-value. For example, table 3 is a reduction of table 2.
We will show that such a 'reduction' is unique and also contains all the 'minimal' rules that can be observed to be true in the fact model. It is a normal form of the original rule model.

Weather	Temperature	Humidity	Wind	Fishing
sunny		high		bad
sunny	hot			
	hot		yes	
rain			yes	
sunny	mild		no	
cloudy				good
rain			no	
sunny	cool			
sunny		normal		
sunny	mild		yes	
	mild	normal		
	hot	normal		
		normal	no	
	cool		no	

Table 3: The reduced normal form of the rule model of table 2. Note that, although each of the antecedent rows for a consequent is unique, underlying sets may intersect.

Proposition 1: Any reduced and-list, which is an antecedent for some consequent attribute-value, is a sub list of an original antecedent and-list.
Proof: Any antecedent and-list is either a sublist of an antecedent list of the same attribute-value, of an antecedent list of another value of that attribute, or neither.
If it is a sublist of another attribute-value, then that sublist implies two values of the same attribute, which falsifies the reduced rule.
If the list is not a sublist of any antecedent in the rule model at all, then there is no observed antecedent for any of the possible values of the consequent attribute. There is no empirical evidence either way for such a rule.
Therefore the reduced list must be a sublist of an original antecedent.

Definition: A reduced normal form of a rule model is the or-list of all the shortest and-lists, which together imply all the values of the consequent attribute.

B: Rule Reduction Algoritm:

[THIEL] contains a mathematical description and a prototype implementation in the functional programming language Haskell of an algoritm for rule reduction. Here we provide a less formal treatment. The reduction algoritm operates on one consequent attribute-value. To obtain the normal form for the complete rule model it must be executed seperately for each consequent predicate.
There are three consecutive steps, hypothesis, falsification and verification.

1. Hypothesis

The starting hypothesis is that the selected consequent attribute-value is implied by the or-list of all attribute-values that are in in the original antecedents.
For the predicate Fishing-bad in table 2, for example, this is the or- list of 9 attribute-values, all of the 10 predicates in all of the rule model, except Weather-cloudy.

2. Falsification

The starting hypothesis is now tested against each rule which we have for the other values of the consequent attribute. Whenever we have matching predicates, we omit those from the hypothetical or-list, because those apparently do not imply our consequent predicate. By anding all the or-lists resulting from these matches, we ge a new or-list of and-lists.
This new tested hypothesis will usually be simpler than the original rule model, because we can use:

1. An and of a predicate with an and-list which already contains that predicate is just the and-list.
2. An and of an attribute-value with an and-list that contains another value of that same attribute is empty.

For example, our first rule in table 2, starting from the top down, is the original hypothesis without the first row (cloudy, hot, high, no) the second one is the original hypothesis without the second row (rain, mild, high, no) and so on. These or-lists are then anded, using the two laws of logic. The 'and' of all the or-lists is the 'or' of and-lists for Fishing-bad in table 3. (The reduction for Fishing-good is obtained by applying the algoritm to that attribute-value.)

3. Verification

The tested reductions are now checked against the original rules for the consequent attribute-value to see whether they are a sub list of one (or more). Any and- list that is not a sublist of an original one is rejected, because of proposition II.A.1.
In the example of table 2 it is found that Weather-rainy 'and' Temperature-hot is not contradictory with Fishing-good. This list, however, is not a sublist of any observed rule for Fishing-bad. Neither is it a sublist of an antecedent for Fishing-good.

Empirically, there is no evidence either way, and this hypothesis must therefore be rejected for both values of the consequent attribute.

C: Reduced Normal Form:

A reduced rule model may be viewed as a normal form of the original rule model if none of its rules can be shortened further, and if it contains all possible (valid) shortest rules.

Proposition 2: No reduced rule can be shortened further.
Proof: The starting hypothesis is an or-list of singleton attribute-values. Any reduction is constructed by the anding of non-matching or-lists and the 'and' and 'or' operations are mutually distributive. Both 'and' and 'or' are commutative and associative, so the order in which the operations are executed has no effect on the result. Therefore, if there were a shorter and-list, it would have been constructed from the starting hypothesis.

Proposition 3: The algoritm produces all possible shortest and-lists.
Proof: Each reduced and-list is a sublist of an original antecedent and-list and it cannot be shortened further.
So, if there is a reduced rule which is not discovered by the algoritm, it must consist of a real sub list of a reduced list, and also one or more attribute-values not in the original antecedent list. Moreover, this reduced list cannot be a superlist or a sublist of any other reduced and-list.
But then it cannot be a sublist of any original antecedent list, which it has to be according to proposition 1.

Therefore the result of a rule reduction is a normal form of the rule model.

III: Partial Order and Abduction

A: Semantics

The or-list of and-lists in a reduced normal form (usually) no longer denotes a refinement of the consequent attribute. Each reduced and-list covers all its superlists in the original rule model and these superlists mark the semantics of the reduced and-lists. They can easily be found by checking each reduced antecedent with all the original antecedents (for the same attribute-value). In table 4 the original antecedents of the rule model of table 2 have been tagged with an identifier, here just the row number.

Line	Weather	Temperature	Humidity	Windy	Fishing
1	sunny	hot	high	no	bad
2	sunny	hot	high	yes	
3	rain	cool	normal	yes	
4	sunny	mild	high	no	
5	rain	mild	high	yes	
6	cloudy	hot	high	no	good
7	rain	mild	high	no	
8	rain	cool	normal	no	
9	cloudy	cool	normal	yes	
10	sunny	cool	normal	no	
11	rain	mild	normal	no	
12	sunny	mild	normal	yes	
13	cloudy	mild	high	yes	
14	cloudy	hot	normal	no	

Table 4: The rule model of table 2 with each antecedent identified by its row number.

Table 5 shows the reduced normal form in table 3 with the and-lists replaced by the tagged superlists. The underlying sets are the unions of the original underlying sets. These original sets are all distinct, but the sets denoted by the reductions may have non-empty intersections, as table 5 clearly shows.

Union	Fishing
1, 2, 4	bad
1, 2,	
2	
3, 5	
4	
6, 9, 13, 14	good
7, 8, 11	
10	
10, 12	
12	
11, 12	
14	
8, 10, 11, 14	
8, 10	

Table 5: The reduced normal form of table 3 with each row representing the union of the original antecedents. Each original antecedent list is a super list of a reduced and-list.

Because the numbers of possible facts and rules are finite (though the set T may be infinite) it is also possible to tag not just the observed rules but all possible rules. Then a reduced and-list may be considered to denote, potentially, all its possible superlists and none other. This extrapolation of the semantics will be used in section IV on induction. Here we consider just the emperically observed rules, those which are in the observed rule model.

B: Partial Order and Abduction:

Using the semantics as in table 5, it is possible to find what reduced and-lists, if any, denote subsets of each other, as shown in table 6. The set of reduced and-lists may be a partially ordered set (poset).

If we discover that having two and-lists, which imply the same consequent attibute-value, one also implies the other, this is called an abduction. (2) For example, table 6 shows that:

Temperature-hot and Humidity-normal (14) => Weather-cloudy (6,9,13,14) => Fishing-good

and so on. Obviously abduction is interesting in itself because it can reveal new rules, but here we just consider the effect on the reduced rule model as a whole.

Implies	Implies	Implies	Fishing
2	1, 2	1,2,4	bad
	4		
		3,5	
	14	6,9,13,14	good
		7,8,11	
	10	10,12	
	12		
	12	11,12	
	14	8,10,11,14	
10	8,10		

Table 6: Table 5 rewritten as chains of underlying subsets which are subsets of a consequent set. Each number identifies an antecedent row in table 4 and each cell represents a reduced and-list of table 3.

Proposition 4: If there is a partial ordering of antecedents in reduced normal form, then there is exactly one representation.
Proof: A reduced normal form is the or-list of all the shortest possible and-lists of a given rule model. Each reduced antecedent list stands for the union of all its original superlists, so there is only one list of identifiers to represent it. Because each reduced and-list is uniquely identified by a list of unique individual identifiers, it can be uniquely determined whether an antecedent implies another or not, by matching the identifiers. Therefore we find all possible abductions. If there are no alternative abductions, then the partial order found is the only one.

Because of proposition 4 a representation of a reduced normal form with its partial ordering is also a normal form.

From table 5, and even more so from table 6, we can see that we do not necessarily need all the reduced rules we have found to cover all the original rules. A collection of reduced rules that covers all the original rules, as identified by the tags, is sufficient. One such representation is shown in table 7.

Antecedent	Fishing
1, 2 ,4	bad
3, 5	
6 ,9 ,13 ,14	good
7, 8, 11	
10, 12	

Table 7a: A sub collection of reductions that contains all the original antecedent and-lists 1 - 14 is sufficient to imply the consequent.

Weather	Temperature	Humidity	Wind	Fishing
sunny		high		bad
rain			yes	
cloudy				good
rain			no	
sunny		normal		

Table 7b: The same table with the antecedent and-lists of table 3.

Table 7 shows the same results J.R. Quinlan first derived from table 1 in the seminal paper (QUINLAN).Different representations of the same rule model (table 2) are also possible, as demonstrated by table 6, but this one appears to be the most efficient. Table 7a shows that the or-list of reduced and-lists denotes a refinement of the Fishing attribute, just like the original rule model.

IV: Induction

A: New Facts and Rules

In a factual model of a set T and a number of partitions over T (the attributes), adding a new fact is just the addition of new elements, each of which has exactly one value for each of the defined attributes.
If the resulting and-list of the new element (or elements) exactly matches an and-list in the fact model, then it has no effect at all. (Of course it raises the frequency count of the observed fact, but in our model this is not used.)
If the new and-list does not match any existing and-lists, then it should be added to the fact model as a new row in the table.
Because the rule model is just the fact model with one attribute selected as consequent, transformation of the new fact to a new rule is trivial. We call the addition of such new rules (or facts) to a rule model (or fact model) induction.

The fact model of table 1 is again used as example, but the rule model is a different one, with Windy as the selected consequent attribute. This rule model, however, contains an ambiguous rule, in which the same antecedent implies both yes and no for Windy.

Id	Weather	Temperature	Humidity	Fishing	Windy
1	rain	mild	high	bad	yes
2	rain	cool	normal	bad	
3	cloudy	cool	normal	good	
4	sunny	mild	normal	good	
5	cloudy	mild	high	good	
6	sunny	hot	high	bad	
7	sunny	hot	high	bad	no
8	cloudy	hot	high	good	
9	rain	mild	high	good	
10	rain	cool	normal	good	
11	sunny	mild	high	bad	
12	sunny	cool	normal	good	
13	rain	mild	normal	good	
14	cloudy	hot	normal	good	

Table 8: The rule model of the fact model of table 1, with Windy as consequent. The conjunction of Weather, Temperature, Humidity and Fishing is not a refinement of Windy. For the subset S, the one minus the ambiguous antecedent of 6 and 7, the attribute conjunction is a refinement.

If there were only one element of T for the ambiguous rule, then this would mean a a logical contradiction, but of course one antecedent list can denote more than one elements. Then the rule is not a contradiction but, rather, an ambiguity in the

rule model. It might be that another attribute, which is not in the rule model, resolves it. For example, fresh water versus salt water could make the difference between good and bad fishing for these cases.
If we leave out the two ambiguous rows, that is, consider only the subset S of T that does not contain those cases, we observe that the conjunction of attributes is a refinement of S. The reduced normal form for this rule model for the set S is shown in table 9.

Weather	Temperature	Humidity	Fishing	Windy
	cool		bad	yes
rain			bad	
		normal	bad	
cloudy	cool			
sunny	mild		good	
sunny	mild	normal		
cloudy	mild			
rain			good	no
sunny			bad	
sunny		high		
sunny	cool			
rain	mild	normal		
	hot			

Table 9: The reduced normal form for the non-ambiguous subset S (without rows 6 and 7) of table 8.

If we add the rule that Weather-sunny, Temperature-hot, Humidity-high, Fishing-bad implies Windy-yes (row 6 of table 8) to the rule model of set S we still have a refinement. For the new set S', which is a superset of S, we get the reduced normal form of table 10.

Weather	Temperature	Humidity	Fishing	Windy
	cool		bad	yes
rain			bad	
		normal	bad	
cloudy	cool			
sunny	mild		good	
sunny	mild	normal		
cloudy	mild			
sunny	hot			
	hot		bad	
rain			good	no
sunny	mild		bad	
sunny	mild	high		
sunny	cool			
rain	mild	normal		
	hot	normal		
cloudy	hot			
	hot		good	

Table 10: The reduced normal form for the non-ambiguous subset S', with row 6 but not row 7, of table 8

Comparing the tables 9 and 10 shows that adding the rule 6 does not change the existing reductions for the same attribute-value (Windy-yes), but adds two new ones. It does, however, change the rules for the other attribute-value (Windy-no).

We can just as well add rule 7 instead of rule 6 to the set S, and now the conjunctions are also a refinement of Windy. The reduced normal form for this set S" (also a superset of S) is in table 11. Comparing table 11 to table 9 shows that the two tables are equal. Adding rule 7 has no effect at all.

Weather	Temperature	Humidity	Fishing	Windy
	cool		bad	yes
rain			bad	
		normal	bad	
cloudy	cool			
sunny	mild		good	
sunny	mild	normal		
cloudy	mild			
rain			good	no
sunny			bad	
sunny		high		
sunny	cool			
rain	mild	normal		
	hot			

Table 11: The reduced normal form for the non-ambiguous subset S", with row 7 but not row 6, of table 8

Finally we can keep the ambiguity and the original set T, which results in the reduced normal form of table 12. Again, comparing with the table(s) without the rule shows that adding a rule for a predicate does not change the existing reductions for that predicate.

Weather	Temperature	Humidity	Fishing	Windy
	cool		bad	yes
rain			bad	
		normal	bad	
cloudy	cool			
sunny	mild		good	
sunny	mild	normal		
cloudy	mild			
rain			good	no
sunny	mild		bad	
sunny	mild	high		
sunny	cool			
rain	mild	normal		
	hot	normal		
cloudy	hot			
	hot		good	

Table 12: The reduced normal form for the ambiguous rule model of table 8 (with rows 6 and 7).

It turns out that the reduction algoritm also works with ambiguous rule sets. This is as as to be expected, because the ambiguous and-list will falsify the starting hypotheses for both Windy-yes and Windy-no (see section II).

The effect of adding a rule to a rule model becomes even clearer when the reduced antecedents are replaced by the union of the original rules they denote, as shown before in table 5 of section III. Using the identifying tags of table 8 for each reduction antecedent we get table 13.

Table 9 - 6, -7	Table 10 + 6, - 7	Table 11 - 6, + 7	Table 12 + 6, + 7	Windy
2	2	2	2	yes
1, 2	1, 2	1, 2	1, 2	
2	2	2	2	
3	3	3	3	
4	4	4	4	
4	4	4	4	
5	5	5	5	
	6			
	6			
9, 10, 13	9, 10, 13	9,10, 13	9,10, 13	no
11	11	7, 11	11	
11	11	7, 11	11	
12	12	12	12	
13	13	13	13	
8, 14	14	7, 8, 14	14	
	8, 14		8, 14	
	8, 14		8, 14	

Table 13: Representation of tables 9 - 12 by the unions of the original antecedents in table 8. The reduction algorithm also works for ambiguous rule models.

Adding rule 6 results in two new reductions of that rule, but adding rule 7 does not result in new reductions, because rule 7 is completely covered by existing reductions.
The last column of table 13, for the ambiguous case, shows that both rules 6 and 7 are absent from the reduced normal form. They are also absent from the first column, for the case where they have not been considered at all, but the reduced normal forms are not the same. Rule reduction also works in case of ambiguity, but an ambiguous rule will make a difference.

B: Propositions for Induction

Proposition 5: Addition of a rule for a consequent attribute-value does not change the existing reductions for that consequent, but can only add new reductions.

Proof: Suppose the new antecedent consists only of those attribute-values that are already in the existing original rules. Then the hypothesis step of the reduction algoritm (see section II) will be the same as before. Therefore the falsification step will also be the same. The verification step will only be effected if a non-falsified hypothesis, which could not be verified, will now be verified by the new rule. This is an addition.
Suppose the new antecedent contains new attribute-values. These are then added to the starting hypothetical or-list. The matching step of the falsification will result in the previous non-matching or-lists, which may or may not be appended by the new predicates. Because of the distributivity of the 'and' and 'or' operations this can only result in the addition of and-lists to the previous or-list of and-lists. These additional reductions, if any, will contain the new attribute-values.

Proposition 6: Addition of a rule for another value of the consequent attribute (which is not this one) can only result in new reductions (for this one) which are superlists of the previous reductions . If there are no superlists the reduced rule is refuted.
Proof: The original rules for the attribute-value itself are the same as before, so the hypothesis step is also the same. The falsification step consists of all the previous matches, plus a match for the new rule. This or-list is then anded to the previous and-list, which is the previous falsification result. So the new falsification result consists of superlists (including equality) of the old falsification result. Verification has three possible outcomes.
If the old and-list was not a sublist of an original rule then any superlist will also be rejected.
It is also possible that the old and-list was accepted but the superlist is not a sublist and is rejected. This particular old and-list for the attribute-value is now rejected because of the new rule for the other attribute-value.
The third possibility is that the new superlist is also a sublist of an original rule, in which case it is accepted. It will replace the old reduced rule.
If all possible superlists of the reduced rule are falsified by new rules for other attributes, then that reduced rule will not appear in the new reduced normal form at all.

These propositions explain the results from subsection A. Adding a rule for an attribute-value added a reduction rule for that value in some cases but left all existing ones. The rules for the other attribute value were in some cases replaced by longer ones with (a) specializing predicate(s), because the old reduction conflicted with the added rule for the other value of the consequent attribute.
The most interesting case is the addition of rule 7 which resulted in the same reduced normal form as before (tables 11 and 9). As the semantics of table 13 show, the new rule is covered by the existing reductions and has no effect on the reductions for the other value either.

C: Soundness and Completeness:

Though the set T might be infinite, the number of possible facts is determined by the defined attributes and finite. The number of possible rules for any rule model is, including the ambiguous ones, equal to the number of facts. The number of possible rules for each consequent attribute-value is the total number of rules, divided by the number of values (attribute size). So, the number of possible rules for Windy-yes and Windy-no is both 72 /2 = 36. In the actual rule model of table 8 these numbers are 6 and 8. In the semantics of reduced normal forms as in table 13, and earlier in table 5, only the observed rules are considered.
But if we tag not only the observed rules, but all possible rules, we can extrapolate the semantics to cover those as well. For example, the antecedent 'Weather-rain and Fishing-bad' covers all cases with all values of the other attributes, i.e. the 3 for Temperature and the 2 for Humidity, 6 in total. Only 2 of these are in table 8, the other 4 are absent.
The potential rules for a reduced normal form can be determined by tagging all possible rules, checking which are covered by each rule in the normal form, and removing redundancies.

Extrapolated rules are absent because they have not been observed and four different cases must be considered for each. The first possibility is that, through some law of the universe, the antecedent can never occur. In that case it will never contradict any reduced rule. If an antecedent is observed (in the future) it might imply an attribute-value, imply another attribute-value or imply more values of the attribute (ambiguity).

Definition: A reduced rule is sound if none of its extrapolated antecedent superlists can ever imply another value of the same attribute.

Definition: An or-list of reduced rules for an attribute-value is sound, if none of the extrapolated antecedent superlists can ever imply another value of that attribute.

Definition: An or-list of reduced rules for an attribute-value is complete, if the extrapolated antecedent superlists include all that are possible for that attribute-value.

By these definitions a group of rules for a predicate may be complete but not sound. Completeness means all possible cases are included in the extrapolated semantics, but some of these may be refuted by new rules for other values of the attribute.

Proposition 7: If the reduced rules for an attribute-value of a binary attribute are sound and complete, then the reduced rules for the other value are sound.
Proof: If the or-list of rules for an attribute is sound, then none of the possible superlists of its and-lists can be superlists of an and-list implying the other attribute. But this does not mean that none of the superlists of the other value can imply the first one, because they could denote original rules not denoted by the reduced rules for the first one. If these rules are complete then this is not possible, so the reduced rules for the second value are sound.

Of course the definitions of soundness and completeness rely on the assumption that some rules or, more precisely, some facts, will never occur. Since the set T of possible cases is infinite, observation can never guarantee that some fact will never turn up and heuristic rules can never be absolutely certain.

References

Quinlan, J.R., 'Induction of Decision Trees', Machine Learning 2, 81 – 106 (1986)
Thiel, J.M. (Hans) van, 'Pragmatic Data Mining', white paper, August 2006, available on request

Notes:

(1) In [THIEL] it is stated that lists of attribute values could have different lengths, but this turns out to be incorrect. Each element of the data set must have exactly one value defined for each attribute and, moreover, this value must be known. Whether the earlier assumption may still hold for some special cases is not clear, at this time, but in general it does not.
(2) The 19th century American logician and philosopher Charles Sanders Peirce is the originator of the term 'abduction' in this particular sense. He defined abduction as the assumption of A => B from A => C and B => C.
(3) Peirce defined 'induction' as the derivation of B => C from A => B and A => C, but we use the term for the inclusion of new empirical facts in a rule model.

