Compression-Based Evaluation of Partial Determinations

Bernhard Pfahringer and **Stefan Kramer**
Austrian Research Institute for Artificial Intelligence
Schottengasse 3, A-1010 Vienna, Austria
{bernhard,stefan}@ai.univie.ac.at

Abstract

Our work tackles the problem of finding partial determinations in databases and proposes a compression-based measure to evaluate them. Partial determinations can be viewed as generalizations of both functional dependencies and association rules, in that they are *relational* in nature and may have *exceptions*. Extending the measures used for evaluating association rules, namely *support* and *confidence*, to partial determinations leads to a few problems. We therefore propose a measure based on the *minimum description length* (MDL) principle to remedy this problem. We assume the hypothetical task of transmitting a given *database* as efficiently as possible. The new measure estimates the compression achievable by transmitting partial determinations instead of the original data. It takes into account both the complexity and the correctness of a given partial determination, thus avoiding overfitting especially in the presence of noise. We also describe three different kinds of search using the new measure. Preliminary empirical results in a few boolean domains are favorable.

1 Introduction

Recently, researchers in KDD have paid much attention to the search for functional dependencies (Mannila & Räihä, 1994) and association rules (Agrawal & Srikant, 1994; Mannila et al., 1994). *Functional dependencies* are essentially relational and do not allow for the possibility of exceptions. Therefore, algorithms searching for functional dependencies would not find a dependency even if there was only one exception to the rule.

For instance, an algorithm might induce the following functional dependency from a train schedule database:

$$\{Direction\} \rightarrow \{Minutes\}$$

This means that the direction of a train determines the time when the train is leaving every hour. If there was a different schedule at night, this kind of algorithm would usually not find such a dependency.

On the contrary, *association rules* not only allow for the possibility of exceptions, but are essentially probabilistic. This is a useful feature for the search especially in large databases where exceptions are likely to arise. However, association rules are propositional and therefore limited in their expressiveness.

For instance, in a supermarket database the association rule

$$\{Bread, Butter\} \Rightarrow \{Milk\}$$

might be found. It says, that customers that purchase bread and butter also purchase milk. Clearly, the statement will not be true for all customers that purchase bread and butter. Anyway, the information is useful even if the association rule holds only for 80% of the customers that satisfy the left-hand side.

Our work deals with a form of knowledge that is more expressive than the propositional association rules, but also well-suited for real-world data, since exceptions are possible. According to (Russell, 1989), we will call those non-strict functional dependencies *partial determinations*. Partial determinations can be viewed as generalizations of both functional dependencies and association rules, in that they are *relational* in nature and may have *exceptions*.

Unfortunately, the measures evaluating association rules, *support* and *confidence*, are not suited for partial determinations. We therefore propose an alternative, compression-based measure for evaluating partial determinations in databases, and describe its use in search.

In section 2 we define partial determinations, in section 3 we introduce the compression-based measure, in section 4 we report on empirical results, in section 5 we survey related work, and finally in section 6 we draw conclusions.

2 Partial Determinations

Before we define partial determinations, we will present the definition of functional dependencies according to (Mannila & Räihä, 1994):

Given a relation schema (i.e. a set of attributes) R, a functional dependency over R is an expression $X \rightarrow Y$, where $X, Y \subseteq R$. To define the semantics of such expressions, let r be a relation (a table) over R,

i.e. a set of tuples over R. We write $Dom(A)$ for the domain of an attribute $A \in R$ and $Dom(X)$ for the domain of a set of attributes $X \subseteq R$. The projection of a tuple t on a set of attributes X, denoted $t[X]$, is defined as the restriction of t on X (here $X \subseteq R$). Now $X \rightarrow Y$ holds in r, denoted $r \models X \rightarrow Y$, if all tuples $u, v \in r$ with $u[X] = v[X]$ also satisfy $u[Y] = v[Y]$.

Partial determinations are generalizations of functional dependencies. They are expressions of the form $X \rightarrow_d Y$. The index d is a number. The set of attributes X will be referred to as LHS, the left-hand side of the partial determination, and Y will be referred to as RHS, the right-hand side.

Semantically, $X \rightarrow_d Y$ holds in r, denoted $r \models X \rightarrow_d Y$, if d is the *determination factor* $d(X, Y)$ as defined by (Russell, 1989). The determination factor is the probability that two randomly chosen tuples have the same values of Y, provided they have the same values of X. Note that $d(X, Y)$ is defined without regard to a particular mapping from $Dom(X)$ to $Dom(Y)$.

Corresponding to a partial determination, we define a mapping $pd_{X \rightarrow_d Y}$ in the following way. The domain of the mapping is defined by the LHS and the range is defined by the RHS of the partial determination. The tuples in r determine the mapping itself: For all tuples u that are equal under the projection X, we call the most frequently occurring $u[Y]$ the majority tuple. $pd_{X \rightarrow_d Y}$ maps tuples of $Dom(X)$ to their respective majority tuples in $Dom(Y)$. A tuple $u \in r$ is called an *exception* to the mapping $pd_{X \rightarrow_d Y}$, if $u[Y] \neq pd_{X \rightarrow_d Y}(u[X])$. Depending on the number of exceptions, the partial determination is more or less "functional".

In the following, the notion of a *partial determination* is used for both the statement $X \rightarrow_d Y$ and the corresponding function $pd_{X \rightarrow_d Y}$. This is not problematic, since there is a correspondence between the referents.

3 Evaluating Partial Determinations

Searching for partial determinations in databases requires a function measuring their goodness. Next, we will show why *support* and *confidence*, the measures of association rules, can not be generalized for this purpose.

3.1 Support and Confidence

For simplicity, we will define *support* and *confidence* using the same notation as for functional dependencies. Let $R = \{I_1, I_2, ..., I_m\}$ be a relation schema with attributes over a binary domain $\{0, 1\}$. (Agrawal & Srikant, 1994) call those attributes *items*.) Let $r = \{t_1, t_2, ..., t_n\}$ be a relation over the relation schema R. An association rule is an expression $X \Rightarrow Y$, where $X \subset R, Y \subset R$, and $X \cap Y = \emptyset$. If $W \subseteq R$ and $t[A] = 1$ for all $A \in W$, we write $t[W] = \bar{1}$. The rule $X \Rightarrow Y$ holds in r with *confidence* c if $c\%$ of the tuples in r for which $t[X] = \bar{1}$ also $t[Y] = \bar{1}$. The rule $X \Rightarrow Y$ has *support* s in r if $t[XY] = \bar{1}$ for $s\%$ of the tuples in r .

Generalizing those measures for partial determinations leads to four different problems:

1. When we consider partial determinations, the tuples can be divided into those which satisfy the rule and those which are exceptions. Here, *support* is the percentage of tuples that satisfy the rule, and *confidence* is 100% minus the percentage of exceptions in r. So *support* and *confidence* are no different measures when applied to partial determinations.
2. When another attribute is added to the LHS of the dependency, the evaluation automatically gets "better" or, in the worst case, remains the same. The reason is that the number of exceptions will decrease in most cases, and at worst will remain constant.
3. Conversely, when adding another RHS-attribute, the evaluation automatically gets "worse" or, in the best case, remains the same. The reason for this is that the number of exceptions will increase in most cases, and at best will remain constant.
4. The measure does not take into account the complexity of the mapping, that is, the information needed to encode the relationship between the domain and the range. Thus, the partial determination just needs to have enough attributes in the LHS in order to obtain a high *support*. Obviously, this may cause overfitting of the data.

In order to avoid these problems, we propose a compression-based measure based on the so-called *Minimum Description Length* (MDL) principle (Rissanen, 1978). This principle allows taking into account a theory's accuracy and its complexity simultaneously. The key idea is to measure the joint cost of coding a theory and of coding the data in terms of that theory. A good introduction to MDL can be found in (Pednault, 1991). In this paper we only deal with partial determinations ranging over boolean attributes and having a single consequent.

3.2 A New Compression-Based Measure for Partial Determinations

We assume the hypothetical task of transmitting a given *database* as efficiently as possible. If we can find a good partial determination for a given attribute, transmitting this partial determination instead of the raw data values can improve efficiency considerably. The MDL principle allows us to estimate the degree of compression achievable by using a given partial determination.

The MDL principle tries to measure both the simplicity and the accuracy of a particular theory (in our setting: partial determination) in a common currency, namely in terms of the number of bits needed for encoding theory and data. (Cheeseman 90) defines the

message length of a theory (called *model* in his article) as:

```
Total message length =
  Message length to describe the model +
  Message length to describe the data,
                  given the model.
```

This way a more complex theory will need more bits to be encoded, but might save bits when encoding more data correctly. The theory with the *minimal* total message length is also the *most probable* theory explaining the data (Rissanen, 1978). Actually we don't really need to encode, we just need a formula estimating the number of bits needed if we were to encode a theory and data in terms of that theory.

So we have to define an encoding for partial determinations that will allow the hypothetical receiver of a transmission to reconstruct the original attribute values. Therefore we have to specify which attributes are used in the LHS, which mapping is used to compute the value of the RHS-attribute, and which tuples in the relation contradict the determination (the exceptions). The exact definition of the coding cost of a partial determination for boolean attributes is given in figure 1.

Total cost (1) is the sum of the cost for selecting the attributes (the *theory*) and for defining their mapping and for encoding the exceptions to the mapping (encoding the *data* in terms of the *theory*).

The selection of attributes (2) is defined by specifying which of all possible attributes are used.

For encoding the mapping only the $2^{|UsedAttrs|}$ different mapping values need to be encoded. This can be done by specifying which of all of these values are "true" (3).

Additionally we have to "correct" the mapping by specifying which examples (tuples in the relation) are assigned a wrong value by the mapping (4).

For estimating the cost of encoding the selection of E elements out of N possible elements (5,6,7) we just use the theoretical entropy-based bound provided by (Shannon & Weaver, 1949). As we do not have to really encode the data, using this theoretical bound makes sense.[1] Alternatively, selection could also be coded and estimated as described in (Quinlan 1993). The advantage of the entropy-based estimate is its efficient computability (only 2 logarithms need to be computed versus $2 * E$ logarithms).

Now according to the MDL principle the one partial determination which minimizes the above cost-function, i.e. the one with the smallest bit-cost (the most *compressive* determination) is the most probable determination given the tuples of the relation.

[1] Especially as there are coding methods known (e.g. *arithmetic coding*), that can actually achieve this bound.

3.3 Bounds for the Compression-Based Measure

We can deduce two bounds useful for searching for partial determinations. First of all the maximal number of LHS-attributes of a useful partial determination is a function of the number of tuples in a given relation (BOUND1):

$$|LHS_{PD}| \quad < \quad log_2|Tuples|$$

This follows from the fact that encoding the raw values of the RHS-attribute (without using a partial determination PD) would yield a bitstring of an approximate size of $|Tuples|$. Now a partial determination PD using $log_2|Tuples|$ LHS-attributes would define a mapping with a truth-table containing $2^{log_2|Tuples|} = |Tuples|$ entries. Additionally we have to encode the selected attributes. Therefore we would get:

$$cost_{MDL}(PD) > cost_{MDL}(\emptyset)$$

Using such a partial determination would be more costly than directly encoding the data.

The second bound is a lower bound for the cost achievable when specializing (i.e. adding attributes to the LHS of) a particular determination PD_N. Assume that this determination uses N attributes and the resulting mapping has p bits "on" and n bits "off". If we add an attribute A_{N+1} to this partial determination, we get BOUND2:

$$\begin{aligned} cost_{MDL}(PD_{N+1}) \quad > \quad & cost_{choose}(N+1, |AllAttrs|) \\ + \quad & cost_{choose}(min(p,n), 2^{N+1}) \end{aligned}$$

This is because we have to encode one more attribute, the resulting mapping will in the - regarding coding cost - best case still have only $min(p,n)$ bits "on" or "off" respectively, and in the best case the number of exceptions will be zero.

This BOUND2 also tells us that specializing a total determination (a determination already having zero exceptions) will not yield a better determination, as its respective cost will be strictly larger.

3.4 Rule-Interest Measures

Using the gain in coding cost ($cost_{MDL}(\emptyset) - cost_{MDL}(PD)$) as the actual evaluation function satisfies the three principles for rule-interest measures stated by (Piatetsky-Shapiro, 1991). They were originally formulated for propositional rules and can easily be reformulated for partial determinations:

1. If the attributes of the left-hand side are statistically independent of the attributes of the right-hand side, the value of the rule-interest function shall be zero.
2. If the support gets bigger, the value of the rule interest function shall increase.
3. If the number of tuples in the database increases and the support stays constant, the value of the rule interest function shall decrease.

$$cost_{MDL}(PD) = cost(LHS) + cost(Mapping) + cost(Exceptions) \quad (1)$$

$$cost(LHS) = cost_{choose}(|UsedAttrs|, |AllAttrs|) \quad (2)$$

$$cost(Mapping) = cost_{choose}(|TrueEntries|, |AllEntries|) \quad (3)$$

$$cost(Exceptions) = cost_{choose}(|Exceptions|, |AllExamples|) \quad (4)$$

$$cost_{choose}(E, N) = N * entropy(E, N) \quad (5)$$

$$entropy(E, N) = -(plog(E/N) + plog(1 - E/N)) \quad (6)$$

$$plog(P) = \begin{cases} 0 & \text{if P=0} \\ P * log_2(P) & \text{otherwise} \end{cases} \quad (7)$$

Figure 1: The Definition of the Coding Cost of a Partial Determination

The first condition is satisfied by MDL-based measures because of the way they are derived from probability theory (Rissanen, 1978). Empirically, this is also evident in experiments on random data. The second condition is obviously true for our measure, since the number of exceptions decreases. The third condition is trivially satisfied as well, because the tuples that are added can only be exceptions if the support stays the same.

4 Empirical Results

For a preliminary empirical evaluation of the above formula we have done a few experiments on mostly artificial boolean data sets with different levels of attribute noise. As enumerating and evaluating all possible subsets of attributes up to a size given by BOUND1 is clearly out of the question for realistically sized data sets, one must rely on search to provide reasonable candidates for evaluation. For this purpose we have implemented three different search strategies:

1. Hill-Climbing: may fail to find a good partial determination, especially if the first attributes chosen during search are irrelevant.
2. Depth-Bounded Backtracking: BOUND2 allows to prune some of the search space, but unfortunately for such small sets of attributes BOUND2 is rarely ever effective, because for small sets exception costs tend to dominate the total coding costs. Bounded depth-first search will of course fail to find partial determinations more complex than the bound. On the other hand, if it fails to find any partial determination, we know that no small determination exists. And if we can afford to search up to a depth equal to BOUND1, we will either find the best partial determination or prove its non-existence.
3. Stochastic Search: This third strategy tries to be both more complete than hill-climbing and to be less costly than bounded depth-first search. This strategy works as follows: N times do randomly generate attribute subsets of a size not exceeding BOUND1. If such a random subset has a smaller bit-cost than the empty set, this subset is already a useful partial determination by itself. But due to its random generation it might contain more attributes than necessary. Therefore to further improve this partial determination, we perform an unbounded depth-first search on this subset, which will return the optimal partial determination possible for this subset. The global effect of this search strategy is that especially in the presence of attribute noise it tends to find better partial determinations at still reasonable runtime costs.

We have conducted a few experiments using purely artificial boolean domains to investigate the effects of attribute noise. Table 1 summarizes search results for a particular boolean relation involving 15 attributes:

$$\begin{aligned} a2 &= or(a0, a1) \\ a7 &= or(a3, a4, a5, a6) \\ a10 &= parity(a8, a9) \\ a15 &= parity(a11, a12, a13, a14) \end{aligned}$$

This particular experiment used 5000 randomly drawn examples. Attribute noise was set to 10%, i.e. with a probability of 10% each attribute value was switched from 0 to 1 or vice versa. Due to space reasons we

Attr	S	Gain	Exc	T	Det
a2	HC	0	30	20	
	SS	1099	17	23	0,1
	DB_3	1099	17	70	0,1
a7	HC	0	15	20	
	SS	116	13	31	2,3,4,5,6,9,10,11,13
	DB_3	0	15	70	
a10	HC	0	50	20	
	SS	939	25	31	8,9
	DB_3	939	25	70	8,9
a15	HC	0	49	20	
	SS	317	34	20	6,10,11,12,13,14
	DB_3	0	49	70	

Table 1: Partial determinations found for selected boolean attributes. Search methods are hill-climbing (HC), stochastic search (SS), and depth-bounded backtracking to level three (DB_3). Gain is the gain in number of bits relative to the empty set, exceptions (Exc) are percentages, runtimes (T) are in seconds, and Det are the LHS-attributes of the respective determination.

only report results for the RHS-attributes. All determinations found really have less exceptions than the empty set. For each type of search we list the determination found (if any), the gain (i.e. $cost_{MDL}(\emptyset) - cost_{MDL}(PD)$) expressed in number of bits, the number of exceptions of the determination in percentages when applying the determination's mapping to the tuples, and the runtimes in seconds.

Hill-climbing fails to find any determinations. Stochastic search finds determinations for all attributes in reasonable time. Bounded depth-first search takes more time than both other methods without finding better solutions. For OR4 and PARITY4, and a depth bound of three, this was to be expected. Only one out of all results possibly shows signs of overfitting (SS for A7).

Table 2 summarizes the results of searching for partial determinations in the "natural" voting domain[2]. This domain is in principle a boolean domain, but does have missing values. Therefore we introduced an additional boolean attribute for each original attribute specifying whether the value is known or not. The binary class information (*democrat* or *republican*) is provided as attribute 0. So we have a total of 33 boolean attributes and 300 examples. Due to space reasons we only report results for a few prototypical RHS-attributes. From the table we see that gain and exceptions correlate as expected, namely that for larger gains we also get fewer exceptions. Complete search only finds slightly better solutions at a pronouncedly larger time cost. Both incomplete methods sometimes fail (see e.g. A20 or A22). But note that the following well-known simple determination between attributes A0 and A8 is found by all three methods:

$$A8_{PhysicianFreezeDecision} \rightarrow A0_{PartyMembership}$$

[2]VOTING is one of the databases available at the UC Irvine Machine Learning Repository

5 Related Work

In this section we briefly describe how our work relates to the closest work found in the literature.

- (Russell, 1989) introduced the term "partial determination" and described a method for evaluating partial determinations with respect to given facts. Briefly, Russell estimates the proportion of tuples in the database for which the determination holds through sampling. In other words, he estimates nothing else but the *confidence* of the dependency given a set of tuples. Therefore the same argument as for *support* and *confidence* applies.
- (Shen, 1991) describes an algorithm that searches for three kinds of regularities in a large knowledge base. One of those regularities are determinations, and they are restricted to those having a single attribute in the left-hand side. Since Shen is not looking for more complex dependencies, there is no need to avoid overfitting the data. Furthermore, the determinations of interest are like association rules in that they have binary attributes. The algorithm generates such simple determinations and returns them if the support is bigger than the counter-support and if a statistical test suggests their significance.
- (Schlimmer, 1993) proposes an algorithm that returns every "reliable" partial determination with a complexity lower than a user-defined threshold. The reliability measure is supposed to measure the "functional degree" of the map given *subsequent* data. Schlimmer argues that this is better than functions that measure the functional degree of the map given the *current* data. The reliability measure is used for pruning: If a determination is not reliable, one can conclude that all determinations with a more complex domain will not be reliable and need not be considered. The algorithm performs a complete search in the space of partial determinations up to a user-defined complexity threshold, but it does not avoid overfitting the data, since it does not have a penalty for overly complex dependencies.

  We recognize two problems with this particular reliability measure:

  1. The reliability measure only evaluates the *domain* of the partial determinations. Consequently, it does not consider how the tuples from the domain map to the tuples from the range. Clearly, not every two functions with the same domain and the same range, but different mappings, should have the same values of the evaluation function. As a further consequence, the measure does not take into account how the projections of the tuples to the attributes from the right-hand side are distributed.

Search	Attr	Gain	Exc	T	Det	Attr	Gain	Exc	T	Det
HC	0	194	5	6	8	2	19	28	8	6,12,23
SS		194	5	15	8		18	30	14	6,16
DB_3		194	5	57	8		19	28	52	6,12,23
HC	12	87	14	6	10,24,26	14	104	15	7	16
SS		79	16	14	10,26		104	15	15	16
DB_3		87	14	52	10,24,26		106	13	52	0,10,16
HC	20	1	37	8	4,32	22	0	35	12	
SS		0	50	14			11	27	14	0,14
DB_3		1	37	52	4,32		20	24	52	0,4,18

Table 2: Partial determinations found for some attributes of the voting domain.

2. The algorithm does not avoid overfitting the data, since it does not have a penalty for overly complex dependencies. We believe that algorithms searching for more complex dependencies have to cope with the problem of overfitting.

6 Further Work and Conclusion

We plan to extend this work by experiments with other search strategies, e.g. genetic algorithms and combinations of search algorithms. Experiments with large real-world databases will have to be the next step. This will require generalizing the measure for multi-valued attributes. More importantly, we will investigate how knowledge can be included in order to obtain meaningful dependencies. [Klemettinen et al., 1994] show how hierarchies of attributes can be used to select rules from large sets of discovered association rules. Alternatively, hierarchies could be utilized by operators *during the search* for dependencies. Furthermore, we will investigate determinations having more than one RHS-attribute.

In summary, we presented a new compression-based measure to evaluate partial determinations and its use for search. Partial determinations are a useful form of knowledge since they are more expressive than association rules but also allow for exceptions. The possibility of exceptions makes this kind of dependency interesting to search for in real-world data. In contrast to other measures of partial determinations known to the authors, the MDL-based function avoids overfitting the data. The usefulness of the approach is supported by experimental evidence.

Acknowledgements This research is sponsored by the Austrian *Fonds zur Förderung der Wissenschaftlichen Forschung (FWF)* under grant number P10489-MAT. Financial support for the Austrian Research Institute for Artificial Intelligence is provided by the Austrian Federal Ministry of Science, Research, and Arts. We would like to thank Gerhard Widmer for valuable discussions.

References

Agrawal R. and Srikant R.: Fast Algorithms for Mining Association Rules. Proceedings of the 20^{th} VLDB Conference, Santiago, Chile, 1994.

Cheeseman P.: On Finding the Most Probable Model, in Shrager J., Langley P.(eds.): *Computational Models of Discovery and Theory Formation*, Morgan Kaufmann, Los Altos, CA, 1990.

Klemettinen M., Mannila H., Ronkainen P., Toivonen H., and Verkamo A.I.: Finding Interesting Rules from Large Sets of Discovered Association Rules. Proceedings of the Third International Conference on Information and Knowledge Management, ACM Press, 1994.

Mannila H., Toivonen H., and Verkamo A.I.: Efficient Algorithms for Discovering Association Rules. Proceedings of the AAAI-94 Workshop on Knowledge Discovery in Databases, 1994.

Mannila H. and Räihä K.-J.: Algorithms for Inferring Functional Dependencies From Relations. *Data & Knowledge Engineering* 12 (1994) 83-99, 1994.

Pednault E.P.D.: Minimal Length Encoding and Inductive Inference. In: *Knowledge Discovery in Databases*, G. Piatetsky-Shapiro and W.J. Frawley (eds.), AAAI Press, 1991.

Piatetsky-Shapiro G.: Discovery, Analysis, and Presentation of Strong Rules. In: *Knowledge Discovery in Databases*, G. Piatetsky-Shapiro and W.J. Frawley (eds.), AAAI Press, 1991.

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

J. Rissanen: Modeling by Shortest Data Description. In: *Automatica*, 14:465-471, 1978.

Russell S.J.: *The Use of Knowledge in Analogy and Induction*. Pitman Publishing, London, 1989.

Schlimmer J.C.: Efficiently Inducing Determinations: A Complete and Systematic Search Algorithm that Uses Optimal Pruning. Proceedings of the 10^{th} International Conference on Machine Learning, 1993.

Shannon C.E. and Weaver W.: *The Mathematical Theory of Communication*, University of Illinois Press, 1949.

Shen W.-M.: Discovering Regularities from Large Knowledge Bases. Proceedings of the 8^{th} International Workshop on Machine Learning, 1991.