Chipper – A Novel Algorithm for Concept Description

Ulf JOHANSSON[a,1], Cecilia SÖNSTRÖD[a], Tuve LÖFSTRÖM[a,b], Henrik BOSTRÖM[b]
[a]*University of Borås, School of Business and Informatics, Borås, Sweden*
[b]*University of Skövde, School of Humanities and Informatics, Skövde, Sweden*

Abstract. In this paper, several demands placed on concept description algorithms are identified and discussed. The most important criterion is the ability to produce compact rule sets that, in a natural and accurate way, describe the most important relationships in the underlying domain. An algorithm based on the identified criteria is presented and evaluated. The algorithm, named Chipper, produces decision lists, where each rule covers a maximum number of remaining instances while meeting requested accuracy requirements. In the experiments, Chipper is evaluated on nine UCI data sets. The main result is that Chipper produces compact and understandable rule sets, clearly fulfilling the overall goal of concept description. In the experiments, Chipper's accuracy is similar to standard decision tree and rule induction algorithms, while rule sets have superior comprehensibility.

1. Introduction

In most cases, a data mining project has its origin in a business problem, where a decision-maker or an executive requests improved support for their decisions. Depending on the type of business problem, different data mining tasks or problem types can be identified. Several taxonomies of data mining problems exist and they agree upon the most important problem types. The problem type *concept description* does not, however, appear in all taxonomies and when it is included, the definitions differ. The CRISP-DM [1] framework identifies six basic problem types in data mining:

- **Data description and summarization**, aimed at concise description of data characteristics, typically in elementary and aggregated form.
- **Segmentation**, aimed at separating data into interesting and meaningful subgroups or classes.
- **Concept descriptions**, aimed at understandable descriptions of concepts or classes.
- **Classification**, aimed at building models which assign correct class labels to previously unseen and unlabeled data items.
- **Prediction**, which differs from classification only in that the target attribute or class is continuous. Prediction is normally referred to as regression.
- **Dependency analysis**, aimed at finding a model that describes significant dependencies or associations between data items or events.

[1] Corresponding author: Ulf Johansson and Cecilia Sönströd are equal contributors to this work. Email: {ulf.johansson, cecilia.sonstrod }@hb.se.

We have earlier, see [2] and [3], argued that the CRISP-DM definition of concept description captures the essential properties of this task, since it states that the purpose of concept description “is not to develop complete models with high prediction accuracy, but to gain insights”. As noted in [3], an important implication of this is that models need not be capable of describing the whole data set. The statement that high predictive accuracy is not required is somewhat deceptive, though, since it refers only to the purpose of the model. To obtain the goal of bringing insights, models should only include relationships between data items that represent meaningful relations in the underlying domain. This entails that concept description models should have the ability to generalize well to new data from the same domain. Obviously, the model must also represent the relationships it contains in a manner that is easily interpretable. To conclude, it follows from the CRISP-DM definition and discussion of concept description that models should describe the targeted concept in an accurate and comprehensible way.

For each of the above problem types, CRISP-DM suggests several appropriate techniques. For concept description, the only two techniques mentioned are rule induction methods and conceptual clustering. Obviously, many rule induction algorithms exist, although none is specifically aimed at concept description. Examining the demands placed on a concept description model, it is clear that rule induction techniques maximizing an information gain measure for every split do not favor good concept description models. Typically, accurate decision trees tend to be fairly complex.

2. Background and related work

We have previously explored the possibility of using predictive modeling techniques for concept description, see e.g. [4][5]. The method of first building an opaque model with high predictive accuracy, typically using some sort of ensemble technique, and then using a powerful rule extraction tool to produce rules was seen to yield models that are comprehensible and have high accuracy. As mentioned above, high accuracy is not important for the purpose of producing accurate predictions, but to guarantee that the model captures general and important relationships in the data, and hence in the underlying domain.

However, when examining the rules/trees obtained in this manner, it became clear that, in the context of concept description, the hitherto used simplistic view of comprehensibility (interpreted as transparent and fairly small models) should be refined. For example, a decision tree containing a root split immediately singling out a large number of instances based on one attribute, is clearly a better description than one where the same instances are classified further down in the tree, possibly with several different splits. In [2], this discussion lead to the following break-down of comprehensibility:

- **Brevity**: The model should classify as many instances as possible using few and simple rules.
- **Interpretability**: The model should express conditions in a way that humans tend to use, i.e. without Boolean conditions or closed intervals.
- **Relevance**: Only those relationships that are general (i.e. have high accuracy) and interesting should be included in the model. What constitutes an interesting relationship is clearly domain and/or problem specific.

Using this refined view of comprehensibility as a basis, it is natural to look for alternative ways of producing rules for concept description, both regarding representation and search strategy. As far as representation is concerned, decision lists seem like an obvious alternative, since they capture the intuitive notion that once a set of instances is explained, those instances can be disregarded when considering the rest. Algorithms producing decision lists are also known as sequential covering algorithms [6], since rules are learnt one at a time. For each rule, all instances covered by this rule are removed from the data set and the next rule is learnt from the remaining instances. Several sophisticated algorithms for producing decision lists exist, and they typically use some information gain measure to decide how to refine rules in each step. Early examples of decision list algorithms include AQ [7] and CN2 [8]. More recent is RIPPER [9], based on IREP [10]. In short, RIPPER constructs its rule sets in three phases, called growing, pruning and optimization. In the growing phase, conditions are added to a rule as long as no negative examples are covered. In the pruning phase, conditions are removed based on performance on a validation set. Pruning may result in the rule covering negative examples. Furthermore, for binary problems, the majority class is the default class and all rules describe the minority class. An instance not covered by any rule is thus assigned the majority class. RIPPER is reported to be well suited to problems with uneven class distributions and scale up well to large data sets, see [9] and [11].

3. Method

In this section, the proposed algorithm, named Chipper, is first presented in detail and then the experiments are described. Chipper is a deterministic algorithm for generating decision lists consisting of simple rules. In its current state, it handles only binary problems and can thus be used for concept descriptions where the aim is to obtain a description of one class in relation to one or several others.

The basic idea is to, in every step, search for the rule that classifies the maximum number of instances using a split on one attribute. For continuous attributes, this means a single comparison using a relational operator. For nominal attributes, this is translated to a set of instances having identical values for that attribute.

Two main parameters, called *ignore* and *stop*, are used to control the rule generation process. The *ignore* parameter specifies the misclassification rate that is acceptable for each rule and can have different values for each output class. The motivation for the *ignore* parameter is that it can be used to view the data set at different levels of detail, with higher values prioritizing the really broad discriminating features of data items and with low values trying to capture more specific rules. The *stop* parameter specifies the proportion of all instances that should be covered by rules before terminating. The motivation for this parameter is that it can be used to find only the most general relationships in the data, instead of trying to find rules to cover particular instances. This parameter is also motivated by the observation in CRISP-DM that concept description models may well be partial. In effect, these two parameters control the level of "granularity" for the decision list.

When building rule sets, Chipper can be made to prioritize rules with high accuracy instead of simply maximizing the number of covered instances, by using the *prefer_accuracy* flag. If used, then for each possible rule, a score is calculated using (1) below.

```
score = #instances_covered × rule_accuracy                      (1)
```

Consequently, when *prefer_accuracy* is true, the candidate rule fulfilling the *ignore* criterion with highest score is chosen. When no more rules can be constructed, the class with the largest remaining number of instances is taken as the default class; this means that different parameter settings can produce rule sets with different default classes. The main operation of Chipper is given in pseudo code below.

```
Input: a data set with two classes
Output: a decision list
Parameters: a stop value S. For each class c, an ignore parameter $I_c$
while proportion of instances classified is smaller than S
  for each attribute a
    find $best\_split_a$;
  select the best_split covering most instances
  formulate a rule using this split
  remove all instances covered by this rule from the data set
```

The sub-procedure *find best_split*$_a$ greedily compares possible splits using one attribute and a single relational operator; i.e. for continuous attributes '<' or '>', and for categorical '='. The *ignore* parameter determines the acceptable number of misclassified instances, either as an absolute number of instances or as a proportion of remaining instances.

3.1. Data sets

Nine publicly available data sets from the UCI machine learning repository [12] were chosen for the experiments, mainly on the basis of having interpretable attributes. A summary of the characteristics of the data sets used is given in Table 1 below, where *Instances* is the total number of instances, *Cont.* is the number of continuous variables and *Cat.* is the number of categorical input variables.

Table 1: Data set characteristics

Data set	Instances	Cont.	Cat.
BUPA	345	6	0
Cleve	303	6	7
Diabetes (Pima)	768	8	0
German	1000	7	13
Hepatitis	155	6	13
Iono	351	34	0
Labor	57	8	8
Sick	3772	22	7
Votes	435	0	16

3.2. Experiments

Two main experiments were carried out to evaluate Chipper for concept description. In all experiments, standard 10-fold cross validation was used. In the first experiment, Chipper was used with parameter settings to optimize predictive power; with *ignore* at 4.5% for both classes, *stop* at 95% and *prefer_accuracy* set to true. Comparisons were carried out against the J48 tree inducer, which is based on C4.5 [13], as implemented in the data mining tool WEKA [14] and against JRIP, the WEKA implementation of RIPPER. For both these techniques, the default settings in WEKA were used.

Using the refined comprehensibility criteria above, the only aspect that is feasible to measure numerically is brevity. It is not, however, obvious how this measure should be constructed. With this is mind, the following three brevity measures were used:

- **Classification complexity** (CC), which measures the average number of tests needed to classify an instance. A low value thus signifies good brevity.
- **Top rule classification rate** (TR), which measures the proportion of instances classified by the top rule in the rule set. For this measure, a high value is desirable from a brevity perspective.
- **Brevity index** (BI_x), as an index capturing how much the x first tests in the rule set classify. This index takes values in [0, 1], where 0 indicates that all instances are classified with the first rule and 1 indicates that all instances are classified after rule number x. The index is calculated using (2) below:

$$BI_x = \frac{\left(\sum_{i=1}^{x} 2^{i-1} p_i\right) + 2^x p_{rest} - 1}{2^x - 1} \qquad (2)$$

  where p_i is the proportion of instances classified in rule (or test) i. p_{rest} is the proportion of instances not classified by the x first rules.

An inherent problem when comparing Chipper against RIPPER is that RIPPER rules contain conjunctions and that no natural translation to the Chipper representation language exists; this is handled by choosing to count either the number of tests or the number of rules, whichever is most natural for the measure at hand. For BI_x, the choice was made to use two different versions, one using number of tests and one using number of rules.

The CC measure represents the choice of evaluating the whole model, i.e. every classification made. For RIPPER, a conjunctive rule is taken to classify at the last conjunct, meaning that a top rule classifying 50 instances with a single AND is taken to classify all those 50 instances with two tests. It should be noted that using such a high stop parameter as 95 in Chipper gives a disadvantage for CC. For the TR measure, a RIPPER top rule with conjunctions is counted as a single rule, which is a rather charitable interpretation. On the other hand, this is a measure that Chipper aims to optimize, and so would be expected to perform well on. Arguably, neither CC nor TR manages to capture brevity in a satisfactory way. CC is too detailed, taking the whole model into account and making distinctions between classification by, say, rule number 10 and rule number 15. TR, on the other hand, is too blunt, since it ignores the possibility that several of the top rules together classify a large proportion of instances. Brevity index is an attempt at a more balanced measure, with the possibility of using a different level of detail. In this study, we opted for BI_3 with the simple motivation that three conditions are quite easy to grasp to a human trying to interpret a rule set, but can still capture several important relationships in the data set.

In the second experiment, the effect of different parameter settings was studied in detail for the Hepatitis data set, with the intention of illustrating how Chipper can be used for different data mining purposes. This also serves as an evaluation of Chipper's ability to fulfill the interpretability and relevance aspects of comprehensibility.

4. Results

The main results from Experiment 1 are shown below. Although the main purpose is to investigate how Chipper performs regarding accuracy compared to standard decision trees and RIPPER, the size of trees/rules produced are also reported, see Table 2. The size measure given is the number of tests in the rule set.

Table 2: Accuracy results from Experiment 1. Chipper set at 4.5-4.5-95

	J48			**RIPPER**			**Chipper**		
Data set	size	acc.	rank	size	acc.	rank	size	acc.	rank
BUPA	25	68.7%	1	4	64.6%	3	16	66.2%	2
Cleve	25	77.6%	2	6	81.5%	1	19	72.7%	3
Diabetes (Pima)	38	73.8%	3	9	76.0%	2	15	77.6%	1
German	139	70.5%	3	5	71.7%	1	19	71.1%	2
Hepatitis	20	83.9%	1	5	78.1%	3	9	80.0%	2
Iono	34	91.5%	1	2	89.7%	2	4	84.6%	3
Labor	2	73.7%	3	4	77.1%	2	4	90.0%	1
Sick	60	98.8%	1	10	98.2%	2	1	94.6%	3
Votes	5	96.3%	2	3	95.4%	3	4	96.5%	1
Average rank			1.89			2.11			2.00

As can be seen from the table, all three techniques perform similarly regarding accuracy over a number of data sets. It is, however, interesting to note that, although Chipper does not explicitly optimize overall accuracy, it still performs slightly better than RIPPER. Since the ultimate goal is comprehensibility in the context of concept description, the measures on brevity defined above are used to compare RIPPER and Chipper. In Table 3 below, all brevity measures are shown. Bold type indicates the best value for that brevity measure.

Table 3: Comprehensibility results from Experiment 1

Data set	**Chipper**					**RIPPER**				
	#tests	CC	TR	BI$_3$Test	BI$_3$Rule	#tests	CC	TR	BI$_3$Test	BI$_3$Rule
BUPA	16	6.21	0.13	**0.70**	0.70	**4**	**3.51**	**0.25**	0.79	**0.11**
Cleve	19	5.40	0.22	**0.54**	0.54	**6**	**4.75**	**0.26**	0.78	**0.29**
Diabetes (Pima)	15	**4.16**	**0.36**	**0.44**	0.44	**9**	7.24	0.23	0.80	**0.32**
German	19	6.18	**0.18**	**0.65**	0.65	**5**	**4.48**	0.17	0.85	**0.12**
Hepatitis	9	**2.51**	**0.57**	**0.31**	**0.31**	**5**	4.55	0.08	0.87	0.36
Iono	4	1.98	**0.26**	0.20	0.20	**2**	**1.74**	0.25	**0.10**	**0.11**
Labor	4	**1.75**	**0.52**	**0.17**	**0.17**	**4**	3.12	0.26	0.69	0.64
Sick	**1**	**1.00**	**1.00**	**0.00**	**0.00**	10	8.54	0.06	0.95	0.40
Votes	4	**1.59**	**0.57**	**0.13**	**0.13**	**3**	4.64	0.32	0.73	0.28
#Wins	1	**5**	**7**	**8**	4	**8**	4	2	1	**5**
Average	10.1	**3.4**	**0.42**	**0.35**	0.35	**5.3**	4.7	0.21	0.73	**0.29**

Regarding rule set size, RIPPER produces more compact rule sets, having the smallest rule set on all but one data set. However, Chipper is explicitly set at classifying at least 95% of all instances, leading to large rule sets. Looking at the different brevity measures, the overall picture is that Chipper clearly outperforms RIPPER. For CC, Chipper obtains the best brevity on 5 out of 9 data sets. For the data sets where Chipper has worse CC the difference is usually quite small, and Chipper also has the best average CC. Regarding TR, Chipper is substantially better than RIPPER, winning 7 data sets and averaging 42% of instances being classified by the top test in the rule set.

The BI_3 measure shows a similar trend when number of tests is used, with Chipper showing the best results. When using number of rules, RIPPER unsurprisingly obtains a lower BI_3 value. The results from Experiment 2 on the Hepatitis data set are shown in Table 4 below, as a summary of rule set size and accuracy.

Table 4: Results from Experiment 2. Rule set size and accuracy for Chipper on Hepatitis

	Ignore					
	2%		4.5%		6%	
Stop	acc.	#tests	acc.	#tests	acc.	#tests
65	78.7%	3	75.3%	3	78.7%	1
80	82.0%	6	78.0%	5	78.0%	3
95	81.3%	12	80.0%	9	79.3%	6

In the design of Chipper, the *Stop* parameter is meant to allow the user to set the level of detail contained in the rule set. From Table 4, it is clear that an increased *Stop* value will produce longer decision lists, which, of course, is in accordance with how this parameter is supposed to work. The *Ignore* parameter is intended as a way of controlling accuracy requirement; however, it cannot be expected to directly correlate to overall accuracy. Especially when using a low *Stop* value, and thus having quite a large proportion of instances classified by the default rule, overall accuracy does not, to a large extent, depend on individual rule accuracy. In Table 4, this is seen by comparing, for example, the accuracies obtained for *Stop* 65. Finally, it is reassuring to note that Chipper is quite consistent regarding accuracy over different parameter settings, meaning that descriptions produced overall have high generality. Figures 1 and 2 below show some sample rules obtained using different parameter settings.

```
if BILIRUBIN <= 1.3 -> DIE  105/9    if BILIRUBIN  <= 1.3 -> DIE  105/9
if ASCITES   == 1   -> LIVE 11/1     if ASCITES    == 1   -> LIVE 11/1
if SGOT      <= 69  -> DIE  16/2     if SGOT       <= 69  -> DIE  16/2
Default: DIE                         if SPLEEN PAL == 1  -> LIVE 7/1
                                     if STEROID    == 2   -> DIE  7/0
                                     if AGE        >= 56  -> LIVE  3/0
                                     Default: DIE
```

Fig. 1. Chipper rule for Hepatitis with parameters 0.06-0.06-80 (left) and 0.06-0.06-95 (right)

```
if ALBUMIN    >= 3.9 -> DIE 83/3     if ALBUMIN    >= 3.9   -> DIE   83/3
if FATIGUE    == 2   -> DIE 14/1     if FATIGUE    == 2     -> DIE   14/1
if ALK PHOSPH >= 168 -> DIE 9/1      if ALK PHOSPH >= 168   -> DIE   9/1
Default: LIVE                        if ALBUMIN    <= 2.8   -> LIVE  11/0
                                     if PROTIME    >= 63    -> DIE   5/0
                                     if BILIRUBIN  >= 3.9   -> LIVE  5/0
                                     Default: DIE
```

Fig. 2. Chipper rule for Hepatitis with parameters 0.02-0.02-65 (left) and 0.02-0.02-80 (right)

One important observation from these rules is that the *Stop* parameter provides a tool for controlling model granularity. Specifically, the left hand side rule in Fig. 2 is a high-level description of the underlying relationship. On this data set, Chipper seldom uses the same attribute twice. One can also note that for the same Ignore value, each rule set starts in the same way; a direct consequence of Chipper's deterministic search strategy. Regarding comprehensibility, the rules are all fairly short and have good brevity measures. Each rule is very simple, making the models highly interpretable.

Regarding relevance, this clearly depends on whether one class is deemed more interesting than the other and also on which attributes a decision-maker would be interested in.

5. Conclusions

In this paper, the Chipper algorithm producing decision lists for concept description has been presented and evaluated. For a concept description model, the important properties are accuracy (to ensure that the model describes general relationships in the underlying domain) and comprehensibility (to ensure that the model presents these relationships as clearly as possible). Comprehensibility can be further broken down into the three properties of brevity, interpretability and relevance. In this study, three different brevity measures were suggested; classification complexity, top rule classification rate and brevity index.

In the experimentation, Chipper was found to perform very well. More specifically, when compared against the standard decision tree technique J48 and the decision list algorithm RIPPER on a number of publicly available data sets, Chipper obtained comparable accuracies and is thus seen to describe general relationships in the data. Additionally, Chipper rule sets were in general more comprehensible than RIPPER rule sets, especially regarding brevity. In more thorough experimentation on one data set, this high accuracy was seen to be stable over different parameter settings, meaning that generality was retained even for very high-level descriptions. To conclude, the proposed Chipper algorithm clearly fulfills the concept description aim of bringing insights into the relationships and properties of a data set.

References

[1] The CRISP-DM Consortium, CRISP-DM 1.0, www.crisp-dm.org, 2000.

[2] C. Sönströd, U. Johansson and R. König, Towards a Unified View on Concept Description, *The 2007 International Conference on Data Mining (DMIN07)*, Las Vegas, NV, 2007.

[3] C. Sönströd and U. Johansson, Concept Description – A Fresh Look, *Proceedings of the 2007 International Joint Conference on Neural Networks*, IEEE Press, Orlando, FL, 2007.

[4] U. Johansson, C. Sönströd and L. Niklasson, Why Rule Extraction Matters, *8th IASTED International Conference on Software Engineering and Applications*, p. 47-42, Cambridge, MA, 2004.

[5] U. Johansson, C. Sönströd and L. Niklasson, Explaining Winning Poker – A Data Mining Approach, *5th International Conference on Machine Learning and Applications*, p. 129 – 134, Orlando, FL, 2006.

[6] J. Han and M. Kamber, *Data Mining – Concepts and Techniques 2nd ed.*, Morgan Kaufman, 2006.

[7] R. S. Michalski, On the quasi-minimal solution of the general covering problem, *Proceedings of the Fifth International Symposium on Information Processing*, p.125-128, Bled, Yugoslavia, 1969.

[8] P. Clark and T.Niblett, *The CN2 induction algorithm*, Machine Learning, 3: 261-283, 1989.

[9] W. Cohen, Fast Effective Rule Induction, *Proceeding of 12th International Conference on Machine Learning (ICML'95)*, p. 115-123, Tahoe City, CA, 1995.

[10] J. Fürnkrantz and G.Widmer, Incremental Reduced Error Pruning, *11th International Conference on Machine Learning (ICML'94)*, p. 70-77, San Mateo, CA, 1994.

[11] T. G. Dietterich, *Machine Learning Research: Four Current Directions*, AI Magazine '18, 97-136, 1997.

[12] C. L. Blake and C. J. Merz, *UCI Repository of Machine Learning Databases*, University of California, Department of Information and Computer Science, 1998.

[13] J. R. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufman, 1993.

[14] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques 2nd ed.*, Morgan Kaufman, 2005.