UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Supervised and Experiential Learning

A rule-based classifier

RULES: A Simple Rule Extraction System

Written by:

Walzthöny, Eric

—

Friday 22nd April, 2022

1 Introduction

Different inductive algorithms have been proposed in the area of Machine Learning (ML), especially in the area of expert systems. The majority of these systems enabled the extraction of knowledge in a suitable format, these include the CLS, ID3, and ID4 amongst others. In this assignment, the focus will be RULES, by Pham and Aksoy (1994). This algorithm starts with forming simple rules with one condition, and then build rules with two, three or more conditions. Any given rule is only added if all the examples presented are classified correctly with it. The termination of the algorithm is when all examples used are covered by a rule. An implementation that Pham and Aksoy used, is the check for irrelevant conditions, which means that given a rule, if it can classify the training examples with a simpler rule with less conditions, it is added. Since there are combinations of rules of different attributes, when an attribute no longer has any unclassified examples, it will not be considered to form a new rule, thus reducing the number of attributes decrease in every iteration.

2 Methods

The algorithm was set up according to the flow diagram depicted in the original paper "RULES: A Simple Rule Extraction System", it is also showed in the image below:

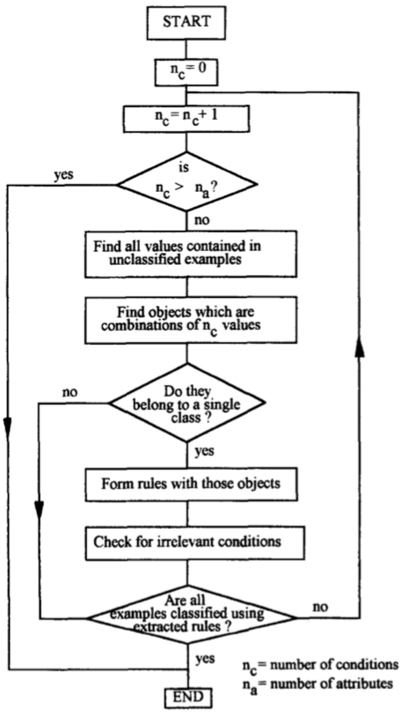

Figure 1: Figure 1. RULES flow chart from the original paper

Additionally, the data was separated into training and testing with a 70/30 split, (70 for testing, 30 for training).

2.1 Dataset

The implementation of this was done in python, using an OOP approach, building classes for the different steps in the pipeline to prepare the data. The datasets used are from the UCI ML Repository, they are the following:

- Breast Cancer (Features: 286, Attributes: 9)
- Cars (Features: 1728, Attributes: 6)
- Agaricus Lepiota (Features: 8124, Attributes: 22)

2.2 Pre-Processing

During the preprocessing different things were done:

- Read the column names from the data file.
- Clean the column names and attach to the original data.
- Convert each categorical attribute into numerical.
- Save the mapping of each categorical-numerical.
- Ensure no missing values are in the data (drop if any).

Each of these are done by a specific class. The Cleaner class is the one that takes the .names data and reads the text until where the attribute names are described. It then cleans and returns a mapping of the abbreviations and column names. On the other hand, the Processor class takes care of mapping from and categorical to numerical and backwards. It also saves the original mapping, such that they are self-contained.

3 Results

For the results, three different tables will be shown which contain the Rule, number of attributes covered, instances covered in the training phase (train coverage), instances covered during the testing phase (test coverage) and the precision of the rule.

These tables are going to be divided according to the size of the dataset: small, medium and large.

3.1 Small

The breast cancer dataset contains attributes regarding demographic data, such as age-group, the specific hormonal period they were in. The target is to classify the diagnosis into recurrent and non-recurrent events.

The small dataset, breast cancer, had the following results as seen in Table 1 below. This table shows a subset of 20 rules, in which 12 rules cover over 50 percent of the data. The values which are highlighted, are the total coverage (train and test) whilst the last value is the average precision of the rules. We can see that when using random generated splits, the number of rules differ, as well as the precision and coverage. Especially if the specific combination is not present in the dataframe.

RULE	Num Attributes	Instances Covered (Train)	Instances Covered (Test)	Precision
IF 3 = 0-4 THEN no-recurrence-events	3	2.70%	1.01%	80.00%
IF 1 = 50-59 AND 3 = 40-44 AND 4 = 0-2 THEN no-recurrence-events	4	3.60%	3.01%	91.67%
IF 1 = 50-59 AND 3 = 10-14 THEN no-recurrence-events	5	4.50%	7.23%	80.00%
IF 3 = 20-24 AND 8 = left_up THEN no-recurrence-events	7	6.31%	10.84%	66.67%
IF 6 = 2 AND 8 = left_up THEN no-recurrence-events	11	9.91%	6.63%	36.63%
IF 3 = 10-14 AND 4 = 6-8 THEN recurrence-events	1	0.90%	0.60%	100.00%
IF 1 = 30-39 AND 6 = 2 THEN no-recurrence-events	5	4.50%	1.20%	72.22%
IF 1 = 30-39 AND 2 = premeno THEN recurrence-events	6	5.41%	7.23%	66.67%
IF 3 = 10-14 THEN no-recurrence-events	3	2.70%	3.61%	100.00%
IF 1 = 40-49 AND 8 = right_low THEN no-recurrence-events	2	1.80%	1.20%	83.33%
IF 1 = 50-59 AND 3 = 50-54 THEN no-recurrence-events	1	0.90%	2.41%	100.00%
IF 1 = 60-69 AND 3 = 25-29 THEN no-recurrence-events	1	0.90%	1.81%	91.67%
IF 2 = premeno AND 8 = central THEN no-recurrence-events	1	0.90%	1.20%	100.00%
IF 2 = ge40 AND 6 = 1 THEN no-recurrence-events	7	6.31%	3.61%	100.00%
IF 1 = 50-59 AND 3 = 30-34 AND 7 = right THEN recurrence-events	2	1.80%	0.90%	100.00%
IF 1 = 30-39 THEN no-recurrence-events	1	0.90%	0.30%	100.00%
IF 1 = 60-69 AND 3 = 40-44 THEN recurrence-events	1	0.90%	1.10%	72.22%
IF 1 = 40-49 AND 6 = 1 AND 9 = no THEN no-recurrence-events	5	4.50%	2.50%	50.00%
IF 1 = 50-59 AND 2 = premeno AND 8 = right_up THEN recurrence-events	1	0.90%	3.01%	50.00%
IF 1 = 40-49 AND 6 = 1 THEN recurrence-events	1	0.90%	1.20%	100.00%
Total		61.24%	60.60%	82.05%

Figure 2: Experimental Results for RULES on small dataset

Here the maximum accuracy that is achieved with a single rule is about 45 percent accuracy. Which is then increased subsequently by increasing the number of attributes used to make the rules. Now, when the training size is reduced, the number of combinations is much easier to find, such that with 11 rules it is able to achieve over 65 percent accuracy.

```
rules._predict()
✓ 0.2s
IF 3 = 25-29  THEN no-recurrence-events Coverage: 72 | Pct-Coverage: 28.80 | Precision 100.0
IF 1 = 50-59  THEN no-recurrence-events Coverage: 114 | Pct-Coverage: 45.60 | Precision 100.0
IF 3 = 15-19  THEN no-recurrence-events Coverage: 31 | Pct-Coverage: 12.40 | Precision 100.0
IF 2 = ge40 AND 3 = 30-34  THEN no-recurrence-events Coverage: 21 | Pct-Coverage: 8.40 | Precision 100.0
IF 2 = premeno AND 3 = 20-24  THEN no-recurrence-events Coverage: 39 | Pct-Coverage: 15.60 | Precision 100.0
IF 5 = yes  THEN recurrence-events Coverage: 23 | Pct-Coverage: 9.20 | Precision 100.0
IF 4 = 3-5  THEN recurrence-events Coverage: 6 | Pct-Coverage: 2.40 | Precision 100.0
IF 1 = 60-69  THEN no-recurrence-events Coverage: 31 | Pct-Coverage: 12.40 | Precision 100.0
IF 1 = 40-49  THEN no-recurrence-events Coverage: 54 | Pct-Coverage: 21.60 | Precision 100.0
IF 3 = 30-34  THEN no-recurrence-events Coverage: 5 | Pct-Coverage: 2.00 | Precision 100.0
IF 1 = 30-39  THEN recurrence-events Coverage: 12 | Pct-Coverage: 4.80 | Precision 100.0
The final accuracy on the test dataset is: 65.60%
```

Figure 3: Example of the output for small training size

We can see that with a reduction in the training size it is able to iterate over the possible combinations much faster than with larger training sizes (approx 0.2s). Now, when training with more examples, such as having the original 70/30 training-testing split, the accuracy increases to over 85 percent.

3.2 Medium

The medium dataset, Cars, we want to be able to predict the class or the "condition" the car is in. For this it would need to use different results. In the figure below it shows that the following rules had a good coverage of the rules during the training but poor during the testing. This can be mitigated by performing different experiments with various trials in order to see how it affects the performance of the model.

RULE	Num Attributes	Instances Covered (Train)	Instances Covered (Test)	Precision
IF 1 = high AND 2 = 4 AND 3 = 2 AND 4 = med THEN low	1	6.40%	1.30%	66.67%
IF 1 = low AND 2 = 2 AND 3 = 2 AND 5 = low THEN vhigh	1	22.67%	12.66%	100.00%
IF 1 = high AND 2 = 3 AND 4 = small THEN low	1	12.20%	4.50%	100.00%
IF 1 = med AND 6 = good THEN low	5	27.67%	2.91%	100.00%
IF 1 = high AND 3 = 4 AND 4 = big AND 5 = low THEN vhigh	1	11.20%	9.01%	100.00%
IF 1 = med AND 2 = 2 AND 3 = 2 THEN vhigh	1	0.58%	11.20%	100.00%
Total		80.72%	41.58%	94.45%

Figure 4: Example of the output for medium training size

Even though the model had a good coverage the final accuracy of the model was about 78 percent when using mentioned training and splitting method.

3.3 Large

The large dataset, albeit took long to run, it managed to get a relative good results, which can be seen in the following table:

RULE	Instances Covered (Train)	Instances Covered (Test)	Precision
IF odor = y THEN p	22.67%	18.36%	66.67%
IF population = y THEN e	12.28%	6.53%	100.00%
IF odor = f THEN p	8.25%	12.93%	100.00%
IF stalk-shape = t THEN e	3.23%	4.21%	100.00%
IF odor = s THEN p	6.50%	7.14%	100.00%
Total	52.93%	49.17%	93.33%

Figure 5: Example of the output for lower training size

Due to the large waiting time for the brute force combination, there had to be a reduction in the training size, and the results which are seen are for a 30/70 training-testing. Albeit the training-testing split, it showed a good coverage of the rules. The overall accuracy of the model was at 58 percent.

4 Conclusion

In conclusion, this assignment gave a good insight into the inner workings of rule based classification systems. Especially, their effect on different dataset sizes, as the computing time increases directly to the number of unique values in each attribute, and the number of attributes. The timings for the small dataset are much lower than that for the large dataset. Albeit, the rule based classifiers are a good method for classification tasks as well as expert systems, the inner workings of which needs to be adapted such that the rule extraction is optimized.