Implementation of RULES

Christian Reiser (20)

Algorithm

RULES-1 starts with forming simple rules with only one condition, and continues by building rules with two, three or more conditions. A rule is only added if all training examples are classified correctly according to this rule. The algorithm terminates when all training examples are covered by a rule. Before a rule is added there is an explicit check whether it contains unnecessary conditions, i. e. the training example can be classified by a simpler rule with less conditions. If an attribute value is not present any longer in the set of unclassified examples, it is not considered anymore for building a new rule. By that the number of considered attributes decreases with every iteration.
However RULES-2 was implemented which works almost in an identical fashion but instead of considering all examples at once, a single unclassified example is fixed for which a rule (with the same properties as in RULES-1) will be deduced no matter how many conditions are needed to build a rule for this example. Also all classified examples are not considered anymore. According to the RULES-2 authors this leads to a considerable speedup. Another beneficial property of RULES-2 is that the explicit check for redundant conditions is not necessary anymore, since no rules are formed in the first place that contain such conditions.

Evaluation

Three datasets from UCI ML were used

- http://archive.ics.uci.edu/ml/datasets/Congressional+Voting+Records
- http://archive.ics.uci.edu/ml/datasets/Car+Evaluation
- http://archive.ics.uci.edu/ml/datasets/Mushroom

For all datasets only 10% of the examples were used for training, so the remaining 90% could be used for testing.

Congressional Voting Records

This dataset contains the answers of individuals to simple yes/no questions. From that the preferred political party (Democrats or Republicans) should be predicted. In the following table are the three rules with highest coverage:

Rule	Coverage (training)	Coverage (test)	Precision (test)
#4 = n ⇒ democrat	46.51%	57.91%	99.12%
#3 = n ⇒ republican	41.86%	32.65%	95.31%
#10 = y ⇒ republican	6.98%	5.36%	71.43%
total	95.35%	95.92%	

With only five rules that each contain only one condition we achieve a high accuracy of 93.88% percent. Both in the training and test dataset the above three rules cover nearly every example. The first rule says that all interviewed Democrats strictly are against the freezing of physicians fees. This is not surprising at all considering that there were always major differences (that still prevail today) between Democrats and Republicans when it comes to health care topics. Overall this result shows how easy it is to find issues that represent a clear line between Democrats and Republicans.

Car evaluation

Test subjects were asked about their opinion about certain cars. Simple features like safety, number of doors, etc. were extracted from these cars which can be used to predict the subjects opinion. Below are the four rules with the highest coverage in the test dataset:

Rule	Coverage (training)	Coverage (test)	Precision (test)
#3 = 2 ⇒ unacc	26.59%	34.08%	100.00%
#5 = low ⇒ unacc	16.76%	19.74%	100.00%
#0 = high ∧ #1 = vhigh ⇒ unacc	3.47%	4.24%	100.00%
#0 = vhigh ∧ #1 = vhigh ⇒ unacc	5.20%	4.05%	100.00%
total	52.02%	62.11%	

In total 36 rules are deduced by the algorithm of which the majority has only a very small coverage. We get with only four rules a coverage of nearly ⅔ of the test dataset. Since all of these rules have a precision of 100% they seem to be reliable predictors. Noteworthy is that all four rules are criterions of exclusion. For example the first rule says that participants give cars with only two doors the lowest possible rating, while the second rule says that subjects universally disliked cars with low safety. A relatively high accuracy of 86.88% is achieved.

Mushrooms

Mushrooms are separated into the two categories edible and poisonous. Characteristics like shape, color and odor are used for prediction of that property. Below are the five rules with highest coverage in the test dataset:

Rule	Coverage (training)	Coverage (test)	Precision (test)
#5 = f ⇒ p	25.62%	26.70%	100.00%
#21 = y ⇒ e	14.04%	12.91%	100.00%
#10 = t ⇒ e	10.71%	11.94%	100.00%

#5 = y ⇒ p	6.65%	7.14%	100.00%
#5 = s ⇒ p	6.65%	7.14%	100.00%
total	63.67%	65.83	

Here we can achieve a very high accuracy of 99.64% with only 26 rules. The odor appears three times in the best-of-5 as a condition and covers thus a lot of examples: if the odor of a mushroom is pungent, fishy or spicy, according to this database it is guaranteed to be poisonous. From the second rule one can conclude that mushrooms that grow solitary are likely to be edible.

Overall can be seen that for all three datasets a high accuracy could be achieved. The cause of this might be that for every dataset it was possible to construct simple rules with only one condition that cover a lot of examples. The full set of rules with precision and coverage is stored in separate files.

Execution of the code

Python with `numpy` and `pandas` packages installed is required. To execute the algorithm the filename of the dataset must be specified as a command line argument. Since the column that contains the class label is different from dataset to dataset this can be specified as an additional command line argument. To reproduce the results from this report the following three commands should be executed:

```
python ./rules ../Data/house-votes-84.data -t 0.9 -c first -pm
python ./rules ../Data/agaricus-lepiota.data -t 0.9 -c first -pm
python ./rules ../Data/car.data -t 0.9 -c last -pm
```

The `-t` parameter specifies the percentage of the dataset that is used for testing, while `-c` specifies the column that contains the class label. `-pm` is used to print detailed metrics (precision and coverage) for each generated rule. For more details please refer to the help command.