

Action Rules



Group 3

Jagadeesh Sudhakariah

Karthik Rangaraj

Katy Mitchell

Aravind Telidevara

Keerthana Sridhar

Krupa Chand

What Are Action Rules?

- An action rule is a rule extracted from an information system that describes a transition of objects from one state to another with respect to a decision attribute.

A rule can be defined as:

$$r = [\underbrace{a1,1 \wedge a2,1 \wedge \dots \wedge ap,1}_{\text{stable part}}] \wedge [\underbrace{b1,1 \wedge b2,1 \wedge \dots \wedge bq,1}_{\text{flexible part}}] \rightarrow d1$$

Stable attribute: attribute's value does not change

Flexible attributes: attribute's value can change

- Actions can be constructed from two rules extracted previously from the same database. These two rules describe two different decision classes. The goal is to re-classify objects from one of these classes into the other.
- Action rules describe knowledge about possible actions associated with objects which are hidden in a decision system.

Project Description

Extraction of Classification rules using LERS:

- The software required for the project was installed.
- The java code was run to obtain the classification rules using the LERS algorithm.
- The program returns the certain, possible and marked values after executing the LERS algorithm.

Reclassification of rules:

- ARAS generates all the classification rules using the LERS strategy.
- Clusters are built around the objects that are to be reclassified by the user by considering a decision attribute from the rules.

A Graphical user interface has been developed to ease the process of accepting input, producing the outputs, setting the support, confidence, initial, end values and selecting the decision attribute using Java.

Dataset

- **Car:** The car dataset consists of 2 files, the data file and the attribute file. The car_data.txt file contains the data on which the algorithm runs in order to calculate rules whereas the car_names.txt file contains the attribute names of the data present in car_data.txt file. The contents of these files can be seen below:

car_data.txt

```
vhhigh,vhigh,2,2,small,low,unacc  
vhhigh,vhigh,2,2,small,med,unacc  
vhhigh,vhigh,2,2,small,high,unacc
```

car_names.txt

```
buying  
maint  
doors  
persons  
lug_boot  
safety  
class
```

- **Mammographic:** Similarly, the mammographic dataset has 2 files whose contents can be seen below:

mam_data.txt

```
5,67,3,5,3,1  
4,43,1,1,?,1  
5,58,4,5,3,1
```

mam_names.txt

```
BI-RADS assessment  
Age  
Shape  
Margin  
Density  
Severity
```

Car Dataset Evaluation

The screenshot displays an IDE with two windows. The left window shows the source code for `MainApp.java`, which is a JavaFX application. The right window is a dialog titled "Extractor: Action Rules" for configuring data extraction parameters.

MainApp.java Code:

```
package com.uncc.kdd;

import javafx.application.Application;

public class MainApp extends Application
{
    @Override
    public void start(Stage pStage) throws Exception
    {
        Parent head = FXMLLoader.Load(getClass().getResource("actionrule.fxml"));
        pStage.setTitle("Extractor: Action Rules");
        pStage.setScene(new Scene(head, 680, 600));
        pStage.show();
    }

    public static void main(String[] args)
    {
        Launch(args);
    }
}
```

Extractor: Action Rules Dialog:

- Data set File Path: C:\Users\KA...
- Attribute File Path: C:\Users\KAR...
- Output File Path: C:\Users\KAR...
- Delimiter:
- Proceed only after loading fi...
- Minimum Support:
- Decision Attribute:
- Minimum Confidence (%):
- Stable Attribute:
- Initial value:
- End value:
-
- Output will be generated in ...
- Output will be generated in ...

Console Output:

```
MainApp (1) [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (11-Nov-2020, 1:29:57 PM)
Data File Input
file path C:\Users\KARTHIK NARAYAN\Desktop\group3_project\input files\car_data.txt
Attribute File Input
file path C:\Users\KARTHIK NARAYAN\Desktop\group3_project\input files\car_names.txt
Output file location
file path C:\Users\KARTHIK NARAYAN\Desktop\group3_project\input files
Load Inputs button clicked...
Input Files Loaded Successfully!
```

Analysis/Results

The screenshot displays an IDE with two main panels. The top-left panel shows the source code for `MainApp.java`, which is part of the `com.uncc.kdd` package. The code imports `javafx.application.Application` and defines a `MainApp` class that extends `Application`. The `start` method loads an `FXMLLoader` with the resource `actionrule.fxml`, sets the stage title to "Extractor: Action Rules", and shows the stage with a scene of size 680x600. The `main` method calls `Launch` with the arguments.

The top-right panel shows the Outline view, listing the `com.uncc.kdd` package and the `MainApp` class with its methods: `start(Stage) : void` and `main(String[]) : void`.

The bottom panel shows the Console output, which includes the following text:

```
<terminated> MainApp (1) [Java Application] C:\Program Files\Java\jre1.8.0_261\bin\javaw.exe (11-Nov-2020, 1:29:57 PM)

$$ Computing Action Rules! $$

$$ ACTION RULES $$

(class: acc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 36 ##### CONFIDENCE: 26.97 percent
(safety = high, class: acc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 36 ##### CONFIDENCE: 26.97 percent
(safety = med, class: acc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 36 ##### CONFIDENCE: 26.97 percent
(safety = med, class: unacc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 36 ##### CONFIDENCE: 26.67 percent
(lug_boot = big, safety = med, class: unacc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 36 ##### CONFIDENCE: 25.00 percent
(buying: low --> vhigh, class = acc ) --> (maint: vhigh --> low) ##### SUPPORT: 16 ##### CONFIDENCE: 33.33 percent
(buying: low --> vhigh, lug_boot = big, class = acc ) --> (maint: vhigh --> low) ##### SUPPORT: 16 ##### CONFIDENCE: 33.33 percent
(buying: low --> vhigh, safety = high, class = acc ) --> (maint: vhigh --> low) ##### SUPPORT: 16 ##### CONFIDENCE: 33.33 percent
(lug_boot = big, safety = high, class: acc --> vgood ) --> (maint: vhigh --> low) ##### SUPPORT: 16 ##### CONFIDENCE: 26.67 percent
(lug_boot = big, safety = med, class: acc --> good ) --> (maint: vhigh --> low) ##### SUPPORT: 16 ##### CONFIDENCE: 44.44 percent
(safety = high, class: acc --> vgood ) --> (maint: vhigh --> low) ##### SUPPORT: 23 ##### CONFIDENCE: 27.88 percent
(safety = high, class: unacc --> vgood ) --> (maint: vhigh --> low) ##### SUPPORT: 24 ##### CONFIDENCE: 26.67 percent
```

Results generated as out.md

cmake -D CMAKE_BUILD_TYPE=RELEASE \ Untitled-1

out (2).md

home > jagu-secure > Downloads > Telegram Desktop > out (2).md > # Computing LERS! > ## Iteration 1 > ### CER

1 # Computing LERS!

2

3 ## Iteration 1

4

5 ### CERTAIN RULES

6

7

8

9 | RULE | SUPPORT | CONFIDENCE |

10 | -- | -- | -- |

11 ### POSSIBLE RULES

12

13

14 | RULE | SUPPORT | CONFIDENCE |

15 | -- | -- | -- |

16 | (buying - high) --> maint - high | 108 | 25.00 % |

17 | (buying - high) --> maint - low | 108 | 25.00 % |

18 | (buying - high) --> maint - vhigh | 108 | 25.00 % |

19 | (buying - high) --> maint - med | 108 | 25.00 % |

20 | (buying - low) --> maint - high | 108 | 25.00 % |

21 | (buying - low) --> maint - low | 108 | 25.00 % |

22 | (buying - low) --> maint - vhigh | 108 | 25.00 % |

23 | (buying - low) --> maint - med | 108 | 25.00 % |

24 | (buying - vhigh) --> maint - high | 108 | 25.00 % |

25 | (buying - vhigh) --> maint - low | 108 | 25.00 % |

26 | (buying - vhigh) --> maint - vhigh | 108 | 25.00 % |

27 | (buying - vhigh) --> maint - med | 108 | 25.00 % |

28 | (buying - med) --> maint - high | 108 | 25.00 % |

Preview out (2).md

Computing LERS!

Iteration 1

CERTAIN RULES

RULE	SUPPORT	CONFIDENCE
------	---------	------------

POSSIBLE RULES

RULE	SUPPORT	CONFIDENCE
(buying - high) --> maint - high	108	25.00 %
(buying - high) --> maint - low	108	25.00 %
(buying - high) --> maint - vhigh	108	25.00 %
(buying - high) --> maint - med	108	25.00 %
(buying - low) --> maint - high	108	25.00 %
(buying - low) --> maint - low	108	25.00 %
(buying - low) --> maint - vhigh	108	25.00 %
(buying - low) --> maint - med	108	25.00 %
(buying - vhigh) --> maint - high	108	25.00 %
(buying - vhigh) --> maint - low	108	25.00 %
(buying - vhigh) --> maint - vhigh	108	25.00 %
(buying - vhigh) --> maint - med	108	25.00 %
(buying - med) --> maint - high	108	25.00 %
(buying - med) --> maint - low	108	25.00 %

Conclusion

- LERS-type algorithm was implemented in order to construct action rules from a single classification rule.
- Relations representing rules produced by LERS strategy are marked.
- The overall complexity of the algorithm was decreased when LERS is used as the pre-processing module for ARAS .
- A Graphical user interface was created and operations were performed on it to obtain the action rules output in a text and md file.

References

1. Raś, Zbigniew W., Elżbieta Wyrzykowska, and Hanna Wasyluk. "ARAS: Action rules discovery based on agglomerative strategy." International Workshop on Mining Complex Data. Springer, Berlin, Heidelberg, 2007.
2. Im, Seunghyun, and Zbigniew W. Raś. "Action rule extraction from a decision table: ARED." International Symposium on Methodologies for Intelligent Systems. Springer, Berlin, Heidelberg, 2008.
3. Tzacheva, Angelina A., and Zbigniew W. Raś. "Action rules mining." International Journal of Intelligent Systems 20.7 (2005): 719-736.

Thank you