Package 'RoughSetKnowledgeReduction'

August 29, 2016

Type Package

Title Simplification of Decision Tables using Rough Sets

Version 0.1

Date 2012-03-13

Author Alber Sanchez

Maintainer Alber Sanchez <a.sanchez@uni-muenster.de>

Description Rough Sets were introduced by Zdzislaw Pawlak on his book ``Rough Sets: Theoretical Aspects of Reasoning About Data". Rough Sets provide a formal method to approximate crisp sets when the set-element belonging relationship is either known or undetermined. This enables the use of Rough Sets for reasoning about incomplete or contradictory knowledge. A decision table is a prescription of the decisions to make given some conditions. Such decision tables can be reduced without losing prescription ability. This package provides the classes and methods for knowledge reduction from decision tables as presented in the chapter 7 of the aforementioned book. This package provides functions for calculating the both the discernibility matrix and the essential parts of decision tables.

License MIT + file LICENSE

Depends methods

Collate DecisionTable.R ConditionReduct.R DiscernibilityMatrix.R ValueReduct.R

NeedsCompilation no

Repository CRAN

Date/Publication 2014-12-18 09:11:32

R topics documented:

`rs-package` *Simplification of Decision Tables using Rough Sets*

Description

Rough sets theory can be applied to reduce knowledge from decision tables. This package includes a few S4 classes for doing so.

Details

Package:	rs
Type:	Package
Version:	0.1
Date:	2012-03-13
License:	GPL (>= 2)
Depends:	methods

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

`checkConsistency` *CHECK CONSISTENCY*

Description

It checks if the rules in a Decision Table object are consistent or inconsistent. A couple of rules are inconsistent if they have the same conditions and different decision; if they have the same decision they are consistent; if they have different conditions no matter the decision they are consistent.

Usage

```
checkConsistency(object)
```

Arguments

object	A Decision Table object

Value

It returns a boolean vector indicating which rules are inconsistent or contradictory in the decision table given. It is a summary of the consistency matrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,computeConsistencyMatrix

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,
1,2,1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
ruleConsistencyDT <- new(Class="DecisionTable",decisionTable = exampleMatrix1)
ruleConsistencyResults <- checkConsistency(ruleConsistencyDT)
```

---

checkConsistency-methods

Methods for Function checkConsistency

Description

Methods for function checkConsistency

Methods

signature(object = "DecisionTable") This method reports which rules in a Decision Table object are consistent. A rule in a Decision Table object is consistent if there is no other rule with the same conditions and a different decision. This method does not specify which pair of rules are consistent or not, for obtaining that information use computeConsistencyMatrix.

`classifyDecisionTable` *CLASSIFY DECISION TABLE*

Description

It applies the Value Reduct object rules to a Decision Table object. It returns an object with new decisions for the rules in the Decision Table object.

Usage

```
classifyDecisionTable(object, decisionTable)
```

Arguments

`object`	A Value Reduct object
`decisionTable`	A Decision Table object

Value

It returns a Decision Table object which rules have the same conditions of input DT object but the rule decisions of the Value Reduct rules where they match.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`ValueReduct-class`,`DecisionTable-class`

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,
1,2,1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,
0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt1 <- new(Class="DecisionTable",decisionTable = exampleMatrix1)
dt2 <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
vr1 <- simplifyDecisionTable(dt1)
dt3 <- classifyDecisionTable(vr1,dt2)#It classifies dt2 with the rules obtained from dt1
dt3 <- removeDuplicatedRulesDT(dt3)
```

`classifyDecisionTable-methods`
Methods for Function `classifyDecisionTable`

Description

Methods for function `classifyDecisionTable`

Methods

`signature(object = "ValueReduct")` This method uses the reduced rules in a Value Reduct object for classifying the rules in a Decision Table object. It returns a Decision Table object with the same original rule conditions but with the decision of the Value Reduct object which apply.

`computeConsistencyMatrix`
COMPUTE CONSISTENCY MATRIX

Description

It computes the consistency matrix of a decision table object. A Consistency Matrix object is made of each rule consistency check against the other rules in a Decision Table object. A couple of rules are inconsistent if they have the same conditions and different decision; if they have the same decision they are consistent; if they have different conditions no matter the decision they are consistent.

Usage

```
computeConsistencyMatrix(object)
```

Arguments

`object`	A Decision Table object

Value

It returns a boolean diagonal matrix indicating inconsistency between rules. It must be interpreted by columns.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,checkConsistency

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,1,
2,1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
conMatDT <- new(Class="DecisionTable",decisionTable = exampleMatrix1)
conMat <- computeConsistencyMatrix(conMatDT)
```

---

computeConsistencyMatrix-methods

Methods for Function computeConsistencyMatrix

Description

Methods for function computeConsistencyMatrix

Methods

signature(object = "DecisionTable") This method checks the consistency between each possible pair of rules in a Decision Table object. For any pair of rules, they are consistent if the same conditions imply the same decision. The method checkConsistency is a summary of this method, reporting if a rule is consistent but it is not specific about the pair of rules tested.

---

computeCore *COMPUTE CORE*

Description

It computes the core conditions of a Decision Table object using a Discernibility Matrix object.

Usage

```
computeCore(object)
```

Arguments

object	A Discernibility Matrix object

Value

It returns a numeric vector indicating the columns ids which are the core of the Decision Table object from which the Discernibility Matrix object was created.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DiscernibilityMatrix-class,computeDiscernibilityMatrix,findFirstConditionReduct,findSmallestReductFam

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
disMatDT <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dm <- computeDiscernibilityMatrix(disMatDT)
core <- computeCore(dm)
```

---

computeCore-methods *Methods for Function* computeCore

Description

Methods for function computeCore

Methods

signature(object = "DiscernibilityMatrix") This method computes the core of a Decision Table object; the core is computed using the discernibility matrix method. The core a set of conditions which are present in all the condition reducts of a decision table; in other words the core is the intersections of the conditions of all the condition reducts in a decision table.

computeDiscernibilityMatrix
COMPUTE DISCERNIBILITY MATRIX

Description

It computes the Discernibility Matrix object of a Decision Table object. The Discernibility Matrix object is made of rule condition differences on a Decision Table object.

Usage

```
computeDiscernibilityMatrix(object)
```

Arguments

object	A Decision Table object

Value

It returns an object of the class DiscernibilityMatrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,DiscernibilityMatrix-class,computeCore

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
disMatDT <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dm <- computeDiscernibilityMatrix(disMatDT)
```

computeDiscernibilityMatrix-methods
Methods for Function computeDiscernibilityMatrix

Description

Methods for function computeDiscernibilityMatrix

Methods

signature(object = "DecisionTable") This method computes the discernibility matrix of a Decision Table object. A Discernibility Matrix object is a 3 dimension boolean array where each element represents if there is a difference in the same condition of each pair of rules in a Decision Table object. A discernibility matrix is useful for calculating the core of a decision table.

computeSupportConsistency
COMPUTE SUPPORT CONSISTENCY

Description

It computes the support and consistency of the rules in the Value Reduct object. For each rule in the Value Reduct object, support is the number of decision table rules to which the value reduct rule conditions apply divided by the number of rules in the decision table object. For each rule in the Value Reduct object, consistency is the number of rules to which the value reduct condition and decision applies divided by the number of rules of the Decision Table object to which the value reduct rule conditions apply.

Usage

```
computeSupportConsistency(object, decisionTable)
```

Arguments

object	A Value Reduct object
decisionTable	A Decision Table object

Value

It returns a numeric matrix which contains the Value Reduct object representation and the support and consistency values of each rule.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ValueReduct-class,classifyDecisionTable

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dtUnique <- removeDuplicatedRulesDT(dt)
cr <- new(Class="ConditionReduct",decisionTable = dtUnique,columnIds=c(1,2,4,5))
cr <- removeDuplicatedRulesCR(cr)
vr <- computeValueReduct(cr)
vr <- removeDuplicatedRulesVR(vr)
mat <- computeSupportConsistency(vr,dt)
print(mat)
```

---

computeSupportConsistency-methods

Methods for Function computeSupportConsistency

Description

Methods for function computeSupportConsistency

Methods

signature(object = "ValueReduct") Support and Consistency are measures of the fitness of a rule respect to a Decision Table object. Support is the ability of a rule to classify the rules in a Decision Table object and Consistency is the correctness of the rule in the Decision Table object. For a single rule, Support counts the number of rules in the Decision Table with the same conditions divided by the total number of rules and Consistency is the number of times the rule, including its decision is found in the decision table.

computeValueReduct *COMPUTE VALUE REDUCT*

Description

It computes the Value Reduct object of the Condition Reduct object. In other words, it removes the superfluous conditions of each rule in the Condition Reduct object.

Usage

```
computeValueReduct(object)
```

Arguments

object	A Condition Reduct object

Value

It returns an object of type ValueReduct.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class,ValueReduct-class,classifyDecisionTable

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
vr <- computeValueReduct(cr)
```

computeValueReduct-methods

Methods for Function computeValueReduct

Description

Methods for function computeValueReduct

Methods

signature(object = "ConditionReduct") This method takes a Condition Reduct object and reduces its rules at the condition level and returns a Value Reduct object.

conditionReduct *CONDITION REDUCT*

Description

User friendly constructor of an instance of the class Condition Reduct. Objects of this class can be created by the user or by objects of the class Decision Table.

Usage

```
conditionReduct(theDecisionTable, theColumnIds)
```

Arguments

theDecisionTable	A decision table object
theColumnIds	A numeric vector representing the column Ids of the decision table which conform the reduct. The decision Id columns is needed, which is always the last column.

Value

It returns a Condition Reduct object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr1 <- conditionReduct(dt,c(1,2,4,5))
isConditionReduct(cr1) == TRUE# Test if cr1 actually is a condition reduct of dt
cr2 <- findFirstConditionReduct(dt)# Gets the first found condition reduct in dt
listCr1 <- findSmallestReductFamilyFromCore(dt)# Gets a set of the least condition reducts of dt
listCr2 <- findAllReductsFromCore(dt)# Gets all the reducts from dt
```

ConditionReduct-class *Class* "ConditionReduct"

Description

A condition reduct is a decision table where the superfluous conditions have been removed. This object can be created by the users, but it suggested its creation by the use of the methods provided by a Decision Table object.

Objects from the Class

Objects can be created by calls of the form `new("ConditionReduct", decisionTable, columnIds)`. A decisionTable is a numeric matrix where each row is a rule. The matrix last column is the decision of the rules and the remaining columns are rule conditions. The columnIds is a numeric vector with the position of the column which conform the condition reduct. This object can be created by the users, but it is preferred its creation by the use of the methods provided by a Decision Table object.

Slots

`decisionTable`**:** Object of class `"DecisionTable"`

`columnIds`**:** Object of class `"numeric"`

Methods

computeValueReduct `signature(object = "ConditionReduct")`: ...

getColumnIds `signature(object = "ConditionReduct")`: ...

getConditionReduct `signature(object = "ConditionReduct")`: ...

getConditionReductDecisionTable `signature(object = "ConditionReduct")`: ...

initialize `signature(.Object = "ConditionReduct")`: ...

isConditionReduct `signature(object = "ConditionReduct")`: ...

print `signature(x = "ConditionReduct")`: ...

removeDuplicatedRulesCR `signature(object = "ConditionReduct")`: ...

show `signature(object = "ConditionReduct")`: ...

Note

This is not a complete implementation of Rough Set theory; instead it is just the application of the theory to decision table simplification also known as knowledge reduction.

Author(s)

Alber Sanchez

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`DecisionTable` `DiscernibilityMatrix` `ValueReduct`

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr1 <- conditionReduct(dt,c(1,2,4,5))
isConditionReduct(cr1) == TRUE# Test if cr1 actually is a condition reduct of dt
cr2 <- findFirstConditionReduct(dt)# Gets the first found condition reduct in dt
listCr1 <- findSmallestReductFamilyFromCore(dt)# Gets a set of the least condition reducts of dt
listCr2 <- findAllReductsFromCore(dt)# Gets all the reducts from dt
```

---

decisionTable *DECISION TABLE*

Description

User friendly constructor of an instance of the class Decision Table.

Usage

```
decisionTable(theDecisionTable)
```

Arguments

`theDecisionTable`
: A numeric matrix representing a decision table

Value

It returns a Decision Table object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,
1,2,1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
```

DecisionTable-class *Class* "DecisionTable"

Description

A decision table is a set of rules with the same number of conditions and only one decision.

Objects from the Class

Objects can be created by calls of the form new("DecisionTable", decisionTable). A decisionTable is a numeric matrix where each row is a ruleThe matrix last column is the decision of the rules and the remaining columns are rule conditions.

Slots

decisionTable: Object of class "matrix"

Methods

checkConsistency signature(object = "DecisionTable"): ...

computeConsistencyMatrix signature(object = "DecisionTable"): ...

computeDiscernibilityMatrix signature(object = "DecisionTable"): ...

findAllReductsFromCore signature(object = "DecisionTable"): ...

findFirstConditionReduct signature(object = "DecisionTable"): ...

findSmallestReductFamilyFromCore signature(object = "DecisionTable"): ...

getCondition signature(object = "DecisionTable"): ...

getDecision signature(object = "DecisionTable"): ...

getDecisionTable `signature(object = "DecisionTable")`: ...

getRule `signature(object = "DecisionTable")`: ...

initialize `signature(.Object = "DecisionTable")`: ...

print `signature(x = "DecisionTable")`: ...

removeDuplicatedRulesDT `signature(object = "DecisionTable")`: ...

show `signature(object = "DecisionTable")`: ...

simplifyDecisionTable `signature(object = "DecisionTable")`: ...

Note

This is not a complete implementation of Rough Set theory; instead it is just the application of the theory to decision table simplification also known as knowledge reduction.

Author(s)

Alber Sanchez

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`DiscernibilityMatrix` `ConditionReduct` `ValueReduct`

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,
1,2,1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
```

---

discernibilityMatrix *DISCERNIBILITY MATRIX*

Description

Objects of this class are not meant to be directly created by users; instead, they are created by the objects of the class Discernibility Matrix.

Usage

```
discernibilityMatrix(theDiscernibilityMatrix)
```

Arguments

`theDiscernibilityMatrix`
: A boolean 3 dimension array

Value

It returns a Discernibility Matrix object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DiscernibilityMatrix-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dm <- computeDiscernibilityMatrix(dt)
```

DiscernibilityMatrix-class

Class "DiscernibilityMatrix"

Description

A discernibility matrix identifies the differences in condition values for each pair of rules in a decision table. Its main function is to help in the calculation of the core of the decision table rules. Objects of this class are not meant to be built by users, instead they should be created using a Decision Table object and the method computeDiscernibilityMatrix.

Objects from the Class

Objects can be created by calls of the form `new("DiscernibilityMatrix", discernibilityMatrix)`. Objects of this class are not meant to be built by users, instead they should be created using a Decision Table object and the method computeDiscernibilityMatrix.

Slots

`discernibilityMatrix`: Object of class `"array"`

Methods

computeCore `signature(object = "DiscernibilityMatrix")`: ...

getDiscernibilityMatrix `signature(object = "DiscernibilityMatrix")`: ...

initialize `signature(.Object = "DiscernibilityMatrix")`: ...

print `signature(x = "DiscernibilityMatrix")`: ...

show `signature(object = "DiscernibilityMatrix")`: ...

Note

This is not a complete implementation of Rough Set theory; instead it is just the application of the theory to decision table simplification also known as knowledge reduction.

Author(s)

Alber Sanchez

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`DecisionTable` `ConditionReduct` `ValueReduct`

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dm <- computeDiscernibilityMatrix(dt)
```

---

findAllReductsFromCore

FIND ALL REDUCTS FROM CORE

Description

It computes all the condition reduct of a Decision Table object taking as a starting point the core conditions of a Decision Table object.

Usage

```
findAllReductsFromCore(object)
```

Arguments

object	A Decision Table object

Value

It returns a list of ConditionReduct objects representing all the reducts found in the Decision Table object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,findFirstConditionReduct,findSmallestReductFamilyFromCore,computeCore

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
listCr <- findAllReductsFromCore(dt)
```

---

findAllReductsFromCore-methods

Methods for Function findAllReductsFromCore

Description

Methods for function findAllReductsFromCore

Methods

signature(object = "DecisionTable") This method seeks all the condition reducts in a Decision Table object. For doing this, it uses the core as an starting point and adds condition combinations until all the conditions in the Decision Table object has been added.

findFirstConditionReduct
FIND FIRST CONDITION REDUCT

Description

Of the many possible condition reducts in a Decision Table, it returns the first found.

Usage

```
findFirstConditionReduct(object)
```

Arguments

object	A Decision Table object

Value

It returns one condition reduct object with the least number of conditions. This reduct belongs to the family of the smallest reducts of the decision table. It may be the only one.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,findSmallestReductFamilyFromCore,findAllReductsFromCore,computeCore

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
c2 <- findFirstConditionReduct(dt)
```

`findFirstConditionReduct-methods`
Methods for Function `findFirstConditionReduct`

Description

Methods for function `findFirstConditionReduct`

Methods

`signature(object = "DecisionTable")` This method returns the smallest first found condition reduct in a Decision Table object. For doing this, it uses the core as a starting point and adds condition combinations until it obtains a condition reduct.

`findSmallestReductFamilyFromCore`
FIND SMALLEST REDUCT FAMILY FROM CORE

Description

It returns a set of condition reducts found in a Decision Table object. The Condition Reduct objects returned have the same number of conditions which is the smallest number of condition on a reduct for the given Decision Table object.

Usage

```
findSmallestReductFamilyFromCore(object)
```

Arguments

`object`	A Decision Table object

Value

It returns a list of Condition Reduct objects representing the smallest reducts, all of them with the same number of conditions.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class,findFirstConditionReduct,findAllReductsFromCore,computeCore

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
listCr <- findSmallestReductFamilyFromCore(dt)
```

---

findSmallestReductFamilyFromCore-methods

Methods for Function findSmallestReductFamilyFromCore

Description

Methods for function findSmallestReductFamilyFromCore

Methods

signature(object = "DecisionTable") This method returns a list of Condition Reduct objects found in a Decision Table object. All the Condition Reduct objects have the same number of conditions and at the same time are the condition reducts with the least number of conditions possible. For doing this, the method uses the core as a starting point and adds condition combinations until it obtains a condition reduct.This method returns a list of Condition Reduct objects found in a Decision Table object. All the Condition Reduct objects have the same number of conditions and at the same time are the condition reducts with the least number of conditions possible. For doing this, the method uses the core as an starting point and adds condition combinations until it obtains a condition reduct.

---

getColumnIds *GET COLUMN IDS*

Description

Accessor method for the column ids which compose a slot of a Condition Reduct object.

Usage

```
getColumnIds(object)
```

Arguments

object	A Condition Reduct object

Value

It returns the numeric vector column Ids.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- findFirstConditionReduct(dt)
cids <- getColumnIds(cr)
```

---

getColumnIds-methods *Methods for Function* getColumnIds

Description

Methods for function getColumnIds

Methods

signature(object = "ConditionReduct") This method returns the column ids of the conditions of a Decision Table object which are thought of being part of a condition reduct. To be sure if it really is a condition reduct the isConditionReduct method could be used.

getCondition *GET CONDITION*

Description

Method for obtaining the conditions of the rules in a Decision Table object.

Usage

```
getCondition(object)
```

Arguments

object	A Decision Table object

Value

It returns the conditions of the decision table as a numeric matrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,1,2,
1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
condDt <- getCondition(dt)
```

getCondition-methods *Methods for Function* getCondition

Description

Methods for function getCondition

Methods

signature(object = "DecisionTable") This method returns the conditions of all rules in a Decision Table object.

getConditionReduct *GET CONDITION REDUCT*

Description

Accessor method for a slot of a Condition Reduct object which returns a numeric matrix representing the object.

Usage

```
getConditionReduct(object)
```

Arguments

object	A Condition Reduct object

Value

It returns the Condition Reduct object inflated as a numeric matrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- findFirstConditionReduct(dt)
getCr <- getConditionReduct(cr)
```

---

getConditionReduct-methods

Methods for Function `getConditionReduct`

Description

Methods for function `getConditionReduct`

Methods

`signature(object = "ConditionReduct")` This method returns a numeric matrix as a representation of a Condition Reduct object.

---

getConditionReductDecisionTable

GET CONDITION REDUCT'S DECISION TABLE

Description

Accessor method for obtaining the Decision Table object of a slot of a Condition Reduct object.

Usage

```
getConditionReductDecisionTable(object)
```

Arguments

`object`	A Condition Reduct object

Value

It returns the Decision Table object of the Condition Reduct object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,3,4,5))
crdt <- getConditionReductDecisionTable(cr)
```

---

getConditionReductDecisionTable-methods

Methods for Function getConditionReductDecisionTable

Description

Methods for function getConditionReductDecisionTable

Methods

signature(object = "ConditionReduct") This method returns the Decision Table object of a Condition Reduct object.

---

getDecision *GET DECISION*

Description

Method for obtaining the decision of the rules in a Decision Table object.

Usage

```
getDecision(object)
```

Arguments

object	A Decision Table object

Value

It returns the decision of the Decision Table object as a numeric vector.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,1,2,
1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
desDt <- getDecision(dt)
```

---

getDecision-methods *Methods for Function* getDecision

Description

Methods for function getDecision

Methods

signature(object = "DecisionTable") This method returns a numeric vector as a representation of the decision of the rules in a Decision Table object.

---

getDecisionTable *GET DECISION TABLE*

Description

Accessor method for obtaining the numeric matrix which represents a slot of Decision Table object.

Usage

```
getDecisionTable(object)
```

Arguments

object	A Decision Table object

Value

It returns the Decision Table object as a numeric matrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,1,2,
1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
dtMat <- getDecisionTable(dt)
```

---

getDecisionTable-methods

Methods for Function getDecisionTable

Description

Methods for function getDecisionTable

Methods

signature(object = "DecisionTable") This method returns a numeric matrix as a representation of a Decision Table object.

getDiscernibilityMatrix *GET DISCERNIBILITY MATRIX*

Description

Accessor method for obtaining the boolean array which represents a slot of a Discernibility Matrix object.

Usage

```
getDiscernibilityMatrix(object)
```

Arguments

object	A Discernibility Matrix object

Value

It returns the Decision Table object as a boolean array of 3 dimensions.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DiscernibilityMatrix-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dm <- computeDiscernibilityMatrix(dt)
dmArray <- getDiscernibilityMatrix(dm)
```

getDiscernibilityMatrix-methods

Methods for Function getDiscernibilityMatrix

Description

Methods for function getDiscernibilityMatrix

Methods

signature(object = "DiscernibilityMatrix") This method returns a 3 dimension boolean array representing a Discernibility Matrix object.

getRule *GET RULE*

Description

Method for obtaining a rule of a Decision Table object as a numeric vector.

Usage

```
getRule(object, ruleIndex)
```

Arguments

object	A Decision Table object
ruleIndex	A numeric vector made of the row indexes of the rules wanted

Value

It returns a subset of rules as numeric matrix; each rule is a row.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable-class

Examples

```
exampleMatrix1 <- matrix(c(1,0,2,1,1,2,2,0,0,1,0,1,0,2,1,1,2,
1,0,0,2,0,1,1,2,1,1,2,0,1,1,0,0,2,1,2,1,1,2,1),ncol = 5)
dt <- decisionTable(exampleMatrix1)
ruleIndex <- c(1,2,4,7,8)
ruleSet <- getRule(dt,ruleIndex)
```

---

getRule-methods *Methods for Function* getRule

Description

Methods for function getRule

Methods

signature(object = "DecisionTable") This method returns a set of rules from a Decision Table object.

---

getValueReduct *GET VALUE REDUCT*

Description

Accessor method for obtaining a numeric matrix representation of a Value Reduct object.

Usage

```
getValueReduct(object)
```

Arguments

object	A Value Reduct object

Value

It returns the value reduct as a numeric matrix.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ValueReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
vr <- computeValueReduct(cr)
getVr <- getValueReduct(vr)
```

---

getValueReduct-methods

Methods for Function getValueReduct

Description

Methods for function getValueReduct

Methods

signature(object = "ValueReduct") This method returns a numeric matrix representing a Value Reduct object.

---

getValueReductConditionReduct

GET VALUE REDUCT'S CONDITION REDUCT

Description

Accessor method for obtaining a Condition Reduct object of a slot of a Value Reduct object.

Usage

```
getValueReductConditionReduct(object)
```

Arguments

object	A Value Reduct object

Value

It returns the condition reduct object of the Value Reduct object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ValueReduct-class,ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
vr <- computeValueReduct(cr)
vrcr <- getValueReductConditionReduct(vr)
```

---

getValueReductConditionReduct-methods

Methods for Function getValueReductConditionReduct

Description

Methods for function getValueReductConditionReduct

Methods

signature(object = "ValueReduct") This method returns the Condition Reduct object of a Value Reduct object.

---

initialize-methods

Methods for Function initialize

Description

Methods for function initialize

Methods

signature(.Object = "ConditionReduct") Constructor for a Condition Reduct object.

signature(.Object = "DecisionTable") Constructor for a Decision Table object.

signature(.Object = "DiscernibilityMatrix") Constructor for a Discernibility Matrix object.

signature(.Object = "ValueReduct") Constructor for a Value Reduct object.

isConditionReduct *IS CONDITION REDUCT*

Description

It tests if a Condition Reduct object is a condition reduct of its Decision Table object.

Usage

```
isConditionReduct(object)
```

Arguments

object	A Condition Reduct object

Value

It returns a boolean indicating if the Condition Reduct object is a condition reduct of the Decision Table object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr1 <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
cr2 <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,5))
isConditionReduct(cr1) == TRUE
isConditionReduct(cr2) == FALSE
```

isConditionReduct-methods *Methods for Function* isConditionReduct

Description

Methods for function isConditionReduct

Methods

signature(object = "ConditionReduct") This method returns a boolean indicating if a Condition Reduct object actually is a condition reduct of its Decision Table object.

print-methods *Methods for Function* print

Description

Methods for function print

Methods

signature(x = "ConditionReduct") It prints a Condition Reduct object.

signature(x = "DecisionTable") It prints a Decision Table object.

signature(x = "DiscernibilityMatrix") It prints a user friendly Discernibility Matrix object.

signature(x = "ValueReduct") It prints a Value Reduct object.

removeDuplicatedRulesCR *REMOVE DUPLICATED RULES FROM CONDITION REDUCT*

Description

It returns a new Conditions Reduct object without the Decision Table object rules which are duplicated in the Condition Reduct object.

Usage

```
removeDuplicatedRulesCR(object)
```

Arguments

object	A Condition Reduct object

Value

It returns a Condition Reduct object without duplicated rules in its Decision Table object from the condition reduct perspective.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dtUnique <- removeDuplicatedRulesDT(dt)
cr <- new(Class="ConditionReduct",decisionTable = dtUnique,columnIds=c(1,2,4,5))
cr <- removeDuplicatedRulesCR(cr)
```

---

removeDuplicatedRulesCR-methods

Methods for Function removeDuplicatedRulesCR

Description

Methods for function removeDuplicatedRulesCR

Methods

signature(object = "ConditionReduct") This method removes the duplicated rules of a Condition Reduct object. For accomplishing this, the method removes rules from the Decision Table of the Condition Reduct object which are duplicated in the column ids that makes the condition reduct.

removeDuplicatedRulesDT *REMOVE DUPLICATED RULES FROM DECISION TABLE*

Description

It returns a new Decision Table object without duplicated rules.

Usage

```
removeDuplicatedRulesDT(object)
```

Arguments

object	A Decision Table object

Value

It returns a Decision Table object without duplicated rules.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

ConditionReduct-class

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dtUnique <- removeDuplicatedRulesDT(dt)
```

`removeDuplicatedRulesDT-methods`
Methods for Function `removeDuplicatedRulesDT`

Description

Methods for function `removeDuplicatedRulesDT`

Methods

`signature(object = "DecisionTable")` This method removes the duplicated rules in a Decision Table object.

`removeDuplicatedRulesVR`
REMOVE DUPLICATED RULES FROM VALUE REDUCT

Description

It returns a new Value Reduct object without duplicated rules.

Usage

```
removeDuplicatedRulesVR(object)
```

Arguments

`object`	A Value Reduct object

Value

It returns a Value Reduct object without duplicated rules.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`ValueReduct-class`

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
dtUnique <- removeDuplicatedRulesDT(dt)
cr <- new(Class="ConditionReduct",decisionTable = dtUnique,columnIds=c(1,2,4,5))
cr <- removeDuplicatedRulesCR(cr)
vr <- computeValueReduct(cr)
vr <- removeDuplicatedRulesVR(vr)
```

---

removeDuplicatedRulesVR-methods

Methods for Function removeDuplicatedRulesVR

Description

Methods for function removeDuplicatedRulesVR

Methods

signature(object = "ValueReduct") This method removes the duplicated rules in a Value Reduct object.

---

show-methods

Methods for Function show

Description

Methods for function show

Methods

signature(object = "ConditionReduct") It shows the first ten rules and conditions of a Condition Reduct object.

signature(object = "DecisionTable") It shows the first ten rules and conditions of a Decision Table object.

signature(object = "DiscernibilityMatrix") It shows the first ten rows and columns of a Discernibility Matrix object.

signature(object = "ValueReduct") It shows the first ten rules and conditions of a Value Reduct object.

`simplifyDecisionTable` *SIMPLIFY DECISION TABLE*

Description

It returns a Value Reduct object which is the smallest and first found on the Decision Table object.

Usage

```
simplifyDecisionTable(object)
```

Arguments

`object`	A Decision Table object

Value

It returns a Value Reduct computed from the first condition reduct found in the decision table.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`DecisionTable-class`

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
vr <- simplifyDecisionTable(dt)
```

simplifyDecisionTable-methods

Methods for Function `simplifyDecisionTable`

Description

Methods for function `simplifyDecisionTable`

Methods

`signature(object = "DecisionTable")` This method returns a Value Reduct object. It is a shortcut for finding the first smallest condition reduct and after the value reduct.

valueReduct *VALUE REDUCT*

Description

Objects of this class are not meant to be directly created by users; instead, they are created by the objects of the class Condition Reduct.

Usage

```
valueReduct(theConditionReduct, theValueReduct)
```

Arguments

`theConditionReduct`	A Condition Reduct object
`theValueReduct`	A numeric matrix representing a value reduct

Value

It returns a Value Reduct object.

Author(s)

Alber Sanchez <alber.sanchez@uni-muenster.de>

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

`ValueReduct-class`

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,
0,0,0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
vr <- computeValueReduct(cr)
```

ValueReduct-class *Class* "ValueReduct"

Description

Objects of this class are not meant to be created by users, instead a Condition Reduct object and the method computeValueReduct should be used. A value reduct is a condition reduct where the superfluous conditions of each rule has been removed.

Objects from the Class

Objects can be created by calls of the form `new("ValueReduct", conditionReduct, valueReduct)`. Objects of this class are not meant to be created by users, instead a Condition Reduct object and the method computeValueReduct should be used.

Slots

`conditionReduct`**:** Object of class `"ConditionReduct"`

`valueReduct`**:** Object of class `"matrix"`

Methods

classifyDecisionTable `signature(object = "ValueReduct")`: ...

computeSupportConsistency `signature(object = "ValueReduct")`: ...

getValueReduct `signature(object = "ValueReduct")`: ...

getValueReductConditionReduct `signature(object = "ValueReduct")`: ...

initialize `signature(.Object = "ValueReduct")`: ...

print `signature(x = "ValueReduct")`: ...

removeDuplicatedRulesVR `signature(object = "ValueReduct")`: ...

show `signature(object = "ValueReduct")`: ...

Note

This is not a complete implementation of Rough Set theory; instead it is just the application of the theory to decision table simplification also known as knowledge reduction.

Author(s)

Alber Sanchez

References

Pawlak, Zdzislaw 1991 *Rough Sets: Theoretical Aspects of Reasoning About Data* Dordrecht: Kluwer Academic Publishing.

See Also

DecisionTable DiscernibilityMatrix ConditionReduct

Examples

```
exampleMatrix2 <- matrix(c(1,1,0,1,1,2,2,0,0,0,1,1,1,2,0,0,0,
0,0,0,2,1,0,0,1,2,2,2,1,1,0,0,2,2,2),ncol = 5)
dt <- new(Class="DecisionTable",decisionTable = exampleMatrix2)
cr <- new(Class="ConditionReduct",decisionTable = dt,columnIds=c(1,2,4,5))
vr <- computeValueReduct(cr)
```

Index