Rough Set Theory for Feature Selection

A sample explaining how to apply Rough Set Theory to select the most significant features from a database

Outline

Rough Set Theory

- Rough set theory was developed by Zdzislaw Pawlak in the early 1980's.
- Rough sets constitutes a sound basis for data mining as a tool to :
  - Feature selection ;
  - Discretization ;
  - Decision rule generation ;
  - Classification ;

Rough Set Theory

- Rough set theory was developed by Zdzislaw Pawlak in the early 1980's.
- Rough sets constitutes a sound basis for data mining as a tool to :
  - **Feature selection ;**
  - Discretization ;
  - Decision rule generation ;
  - Classification ;

Information System/Table

TABLE – Information System (IS)

Patients	Headache	Muscle-pain	Temperature
o_1	Yes	Yes	very high
o_2	Yes	No	high
o_3	Yes	No	high
o_4	No	Yes	normal
o_5	No	Yes	high
o_6	No	Yes	very high

- $IS = (U, C)$;
- $U = \{o_1, o_2, \ldots, o_j\}$ is a non-empty, finite set of objects called the "***universe***";
- $C = \{c_1, c_2, \ldots, c_n\}$ is a non-empty, finite set of "***condition***" attributes;

Information System/Table

TABLE – Information System (IS)

Patients	Headache	Muscle-pain	Temperature
o_1	Yes	Yes	very high
o_2	Yes	No	high
o_3	Yes	No	high
o_4	No	Yes	normal
o_5	No	Yes	high
o_6	No	Yes	very high

$\leadsto$ The Table represents an IS consisting of **3** conditional features (*Headache*, *Muscle-pain*, *Temperature*) and **6** objects ($o_1, \ldots, o_6$).

Decision System/Table

TABLE – Decision System (DS)

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

- $DS = (U, C \cup \{d\})$;
- $U = \{o_1, o_2, \ldots, o_j\}$ is a non-empty, finite set of objects called the "***universe***";
- $C = \{c_1, c_2, \ldots, c_n\}$ is a non-empty, finite set of "***condition***" attributes;
- $d \notin C$ is a distinguished attribute called "***decision***";
- The value set of d, called $\theta = \{d_1, d_2, \ldots, d_s\}$;

Decision System/Table

TABLE – Decision System (DS)

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

$\leadsto$ The Table represents a DS consisting of **3** conditional features (*Headache, Muscle-pain, Temperature*), **1** decision feature (*Flu*) having the values {*Yes, No*} and **6** objects.

Issues in the Decision Table

RST recalls that :

- The **same or indiscernible** objects may be represented several times.
- Some of the attributes may be superfluous.
- Some of attributes values may be superfluous.

Indiscernibility Relation

- The **indiscernibility relation** IND_C is an **equivalence relation** :
- This equivalence relation induces a **partitioning** of the universe.
- The partitions can be used to build new **subsets** of the universe.

$$IND_C = U/C = \{[o_j]_C | o_j \in U\} \tag{1}$$

$$[o_j]_C = \{o_i | C(o_i) = C(o_j)\} \tag{2}$$

- The **equivalence class** that includes o_j is denoted by $[o_i]_C$.

Indiscernibility Relation

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

The IND relation for some subsets of condition attributes C :

☞ **We look for the objects having the same attribute value !**

- $IND_{\{Headache\}} = \{\{o_1, o_2, o_3\}, \{o_4, o_5, o_6\}\}$
- $IND_{\{Headache, Muscle-pain\}} = \{\{o_1\}, \{o_2, o_3\}, \{o_4, o_5, o_6\}\}$

ils ont la même valeur de "headache"

ils ont le même couple de valeur pour "headache, Muscle-pain"

Indiscernibility Relation

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

The IND relation for some subsets of condition attributes C :

☞ **We look for the objects having the same attribute value !**

- $IND_{\{Headache\}} = \{\{o_1, o_2, o_3\}, \{o_4, o_5, o_6\}\}$
- $IND_{\{Headache, Muscle-pain\}} = \{\{o_1\}, \{o_2, o_3\}, \{o_4, o_5, o_6\}\}$
- $IND_C = U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$

(C all condition attributes : the 3 features) -> on prend toutes les conditions

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

algo calcul IND_C

entrée : système decision

sortie : Ensemble d'id

$IND_C = \{\}$

IS <- DS.drop(d)

group ← IS.groupby(C)

Pour g in group:

    IND_C.append(g[index])

Ex:

$IND_C = \{\}$

IS =

Patients	Headache	Muscle-pain	Temperature	
o_1	Yes	Yes	very high	gp 1
o_2	Yes	No	high	gp 2
o_3	Yes	No	high	gp 2
o_4	No	Yes	normal	gp 3
o_5	No	Yes	high	gp 4
o_6	No	Yes	very high	gp 5

ids   C_drop

$IND\_C = \{\{o_1\}, \{o_2, o_3\}, \ldots\}$

Lower & Upper Approximations

- If we have target concept :
  $X = \{o_j | Flu(o_j) = \{yes\}\} = \{o_1, o_2, o_5, o_6\}$
- We want to describe X using the subsets of the universe :
  $U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$
- The concept X cannot be defined in a *crisp manner*, due to the subset $\{o_2, o_3\}$: o_1, o_5 **and** o_6 **are singletons belonging to** X**. But** o_2 **is coupled with** o_3 **which does not belong to** X**.**
- We can approximate X by constructing two certain subsets :

1. **Lower Approximation**
2. **Upper Approximation**

Terminologies

- Let $A = (U, C)$ and let $B \subseteq C$ and $X \subseteq U$
- The **B-lower approximation** of X, denoted $\underline{B}(X)$

$$\underline{B}(X) = \{o_j | [o_j]_B \subseteq X\} \tag{3}$$

$\leadsto$ Objects in $\underline{B}(X)$ can be with certainty classified as members of X on the basis of knowledge in B.

- The **B-upper approximation** of X, denoted $\overline{B}(X)$

$$\overline{B}(X) = \{o_j | [o_j]_B \cap X \neq \emptyset\} \tag{4}$$

$\leadsto$ Objects in $\overline{B}(X)$ can be only classified as possible members of X on the basis of knowledge in B.

Terminologies

- The **B-boundary region** of X,

$$BN_B(X) = \overline{B}(X) - \underline{B}(X) \tag{5}$$

$\rightsquigarrow$ The boundary region, $BN_B(X)$, is the set of objects that can possibly, but not certainly, be classified in this way.

$\rightsquigarrow$ **A set is said to be rough if its boundary region is non-empty.**

Set approximation

$$IND_C = U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$$

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

identique

Let $X_1 = \{o_j | Flu(o_j) = \{yes\}\} = \{o_1, o_2, o_5, o_6\}$

- $\underline{C}X_1 = \{o_1, o_5, o_6\}$: **With respect to IND_C and X_1, o_2 does not belong as we are not certain that it belongs to $\underline{C}X_1$. This is because it is coupled with o_3.**

- On prend que les obj avec Flu = Yes et qui ne sont pas identique.

$IND_C = U/C = \{\{o_1\},\{o_2,o_3\},\{o_4\},\{o_5\},\{o_6\}\}$

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

identique

Let $X_1 = \{o_j \mid Flu(o_j) = \{yes\}\} = \{o_1, o_2, o_5, o_6\}$

- $\underline{C}X_1 = \{o_1, o_5, o_6\}$: **With respect to IND_C and X_1, o_2 does not belong as we are not certain that it belongs to $\underline{C}X_1$. This is because it is coupled with o_3.**

- On prend que les obj avec Flu = Yes et qui ne sont pas identique.

~~algo Lower - approximation~~

~~entrée : DS~~

~~IND_C~~

~~Sortie : Lower-appro~~

~~...~~

$X_1 = \{o_j \mid \underline{Flu(o_j) = \{\{yes\}\}}$ ↳ cond

$= \{o_1, o_2, o_3, o_6\}$

$\underline{C}X_1 = \{\}$ code python

Pour x dans X_1

Pour ind dans IND_C // ind sont des sous-liste

si x dans ind

filtre = a revoir

ind_filtrer = ?

pour x_2 dans ind_filtrer

On peut simplifier ?

↳ $\overline{C}X_i$ facile à calculer *

↳ Partir de $\overline{C}X_i$?

↳ X_i toutes les valeurs possible pour d

↳ Toutes les classes en gros.

*

$\overline{C}X_i$ = id de tous les obj tel que la décision d = condition i

(Ex : X_1 → condition "Flu = yes".)

1) On calcul $\overline{C}X_i$

2) On regarde IND_C

↳ si un obj dans $\overline{C}X_i$ est dans un groupe identique à un obj avec une condition différente, on l'enlève de $\overline{C}X_i$

3) On a trouvé $\underline{C}X_i$

comment faire ? comment l'écrire ?

$\underline{C}X_i = \overline{C}X_i$ (copy) x

↳ Pour chaque objet x dans $\overline{C}X_i$:

ind_cx = IND_C[x] // on récupère le groupe où se trouve x.

Pour tous les obj x' ≠ x dans ind_cx :

si d[x] ≠ cond :

$\underline{C}X_i$. remove(x)

break (break quelle branch !)

retourner $\underline{C}X_i$

$\boxed{\text{ça marche ?}}$

~~algo approximation_inferieur~~

Avec quoi le remplir ?

↳ elem pas ok ↳

~~X_i ← DS. groupby (d).index~~

↳ entrée

~~$\underline{C}X_i$ ← {}~~ Faut pas qu'il soit vide, juste a supp

F X_i ?

~~Pour group in X_i : // Groupe d'objet par decision~~

~~$\underline{C}X_g$ ← {}~~ inutile

~~Pour obj in group :~~

~~inc_obj ← IND_C (obj)~~

~~// on récupère le groupe indiscernable~~

~~// où se situe l'obj~~

~~Pour ~~chaque~~ obj 2 ~~dans~~ in inc_obj :~~

~~si ~~obj2[d]~~ ≠ obj[d]~~

~~// si l'obj possède un obj indiscernable~~

~~// avec une decision différente~~

~~$\underline{C}X_i$. remove (obj)~~

↳ que des indices dans IND_C

algo approximation_inferieur

Avec quoi le remplir ?

↳ elem pas ok ↳

X_i ← DS. groupby (d).index

↳ entrée

$\underline{C}X_i$ ← {} Faut pas qu'il soit vide, juste a supp

F X_i ?

Pour group in X_i : // Groupe d'objet par decision

~~$\underline{C}X_g$ ← {}~~ inutile

Pour obj in group :

inc_obj ← IND_C (obj)

// on récupère le groupe indiscernable

// où se situe l'obj

Pour ~~chaque~~ obj 2 ~~dans~~ in inc_obj :

si ~~obj2[d]~~ ≠ obj[d]

// si l'obj possède un obj indiscernable

// avec une decision différente

$\underline{C}X_i$. remove (obj)

↳ que des indices dans IND_C

algo approximation_inferieur

entrée : DS

IND_C

d

Sortie $\underline{C}X_i$

// Groupe obj par decision

X_i ← DS. groupby (d)

// type X_i = liste de tuples

// (d, obj) ~~obj ≠ d~~

$\underline{C}X_i$ ← X_i

// suppression des obj avec

// IND_C

⚠ tuple ← Pour g in X_i

[("Yes", patients) ...]

...

Pour obj in g :

// Il possède un obj identique ?

ind_c_obj = get_ind_c(obj.index)

si len(ind_c_obj) > 1

index ← Pour identique in ind_c_obj :

si DS[identique][d]

≠ obj[d]

↳ type ?

$\underline{C}X_i$. remove (obj)

possible ?

Set approximation

$$IND_C = U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$$

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

Let $X_1 = \{o_j | Flu(o_j) = \{yes\}\} = \{o_1, o_2, o_5, o_6\}$

- $\underline{C}X_1 = \{o_1, o_5, o_6\}$: **With respect to IND_C and X_1, o_2 does not belong as we are not certain that it belongs to $\underline{C}X_1$. This is because it is coupled with o_3.**
- $\overline{C}X_1 = \{o_1, o_2, o_3, o_5, o_6\}$: **Same, o_2, o_3 are included in $\overline{C}X_1$ as they can be considered as possible member of X_1.**

Tous les elt avec Flu = Yes + elt identique

Set approximation

$$IND_C = U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$$

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

Let $X_1 = \{o_j | Flu(o_j) = \{yes\}\} = \{o_1, o_2, o_5, o_6\}$

- $\underline{C}X_1 = \{o_1, o_5, o_6\}$: **With respect to IND_C and X_1, o_2 does not belong as we are not certain that it belongs to $\underline{C}X_1$. This is because it is coupled with o_3.**
- $\overline{C}X_1 = \{o_1, o_2, o_3, o_5, o_6\}$: **Same, o_2, o_3 are included in $\overline{C}X_1$ as they can be considered as possible member of X_1.**
- $BN_C(X_1) = \{o_2, o_3\}$ **(Equ. 5)** $= \overline{C}X_1 - \underline{C}X_1$

Set approximation

$$IND_C = U/C = \{\{o_1\}, \{o_2, o_3\}, \{o_4\}, \{o_5\}, \{o_6\}\}$$

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

Let $X_2 = \{o_j | Flu(o_j) = \{No\}\} = \{o_3, o_4\}$

- $\underline{C}X_2 = \{o_4\}$
- $\overline{C}X_2 = \{o_2, o_3, o_4\}$
- $BN_C(X_2) = \{o_2, o_3\}$

$\rightsquigarrow$ The decision class, "**Flu**", is **rough** since the boundary region is not empty.

Set approximation

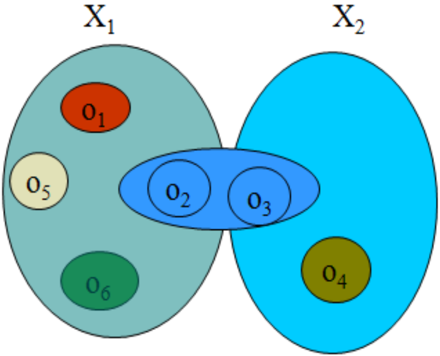

FIGURE – Set approximation

Positive region

- **Positive region** is the set of elements of U that can be uniquely classified to blocks of partion $U/\{d\}$, by means of C.

$$\boxed{POS_C\{(d)\} = \bigcup_{X \in U/\{(d)\}} \underline{C}(X)} \quad (6)$$

- $\underline{C}X_2 = \{o_4\}$
- $\underline{C}X_1 = \{o_1, o_5, o_6\}$

$\rightsquigarrow$ $POS_C\{(d)\} = \{o_1, o_4, o_5, o_6\}$ -> Union de $\underline{C}X_2$ et $\underline{C}X_1$

- **Positive region** is the set of elements of U that can be uniquely classified to blocks of partion $U/\{d\}$, by means of C.

$$POS_C\{(d)\} = \bigcup_{X \in U/\{(d)\}} \underline{C}(X) \tag{6}$$

- $\underline{C}X_2 = \{o_4\}$
- $\underline{C}X_1 = \{o_1, o_5, o_6\}$

$\rightsquigarrow$ $POS_C\{(d)\} = \{o_1, o_4, o_5, o_6\}$ -> Union de $\underline{C}X_2$ et $\underline{C}X_1$

~~algo POS_C~~
~~entrée : $\underline{C}X_i$~~
~~liste des $\underline{C}X$~~
~~return concat sublistes~~
~~↳ voir sur internet si c'est possible~~
~~* en 1 seule ligne~~

algo dependance_attributs
entrée : POS_C
U (univers)
sortie K entier
retourner $\dfrac{\text{len(POS\_C)}}{\text{len(U)}}$

- **Dependency of attributes** : the attribute d depends partially (in a degree k) on a set of attributes C.

$$k = \gamma(C, \{d\}) = \frac{|POS_C\{(d)\}|}{|U|} \tag{7}$$

$\Leftrightarrow \dfrac{\text{nbr elt } POS_c\{(d)\}}{\text{nbr elt de } U}$

* panda ~~flatten~~
~~flat_list = list(flatten(liste))~~

Positive region & Dependency of attributes

- **Dependency of attributes** : the attribute d depends partially (in a degree k) on a set of attributes C.

$$k = \gamma(C, \{d\}) = \frac{|POS_C\{(d)\}|}{|U|} \tag{7}$$

$$\Leftrightarrow \frac{\text{nbr elt } POS_c\{(d)\}}{\text{nbr elt de } U}$$

Positive region & Dependency of attributes

- **Dependency of attributes** : the attribute d depends partially (in a degree k) on a set of attributes C.

$$k = \gamma(C, \{d\}) = \frac{|POS_C\{(d)\}|}{|U|} \tag{7}$$

- $POS_C\{(d)\} = \{o_1, o_4, o_5, o_6\}$
- $U = \{o_1, \ldots, o_6\}$

Positive region & Dependency of attributes

- **Dependency of attributes** : the attribute d depends partially (in a degree k) on a set of attributes C.

$$k = \gamma(C, \{d\}) = \frac{|POS_C\{(d)\}|}{|U|} \tag{7}$$

- $POS_C\{(d)\} = \{o_1, o_4, o_5, o_6\}$ $= 4$
- $U = \{o_1, \ldots, o_6\}$ $= 6$

$\rightsquigarrow$ $k = \gamma(C, \{d\}) = 4/6$

Issues in the Decision Table

- The **same or indiscernible** objects may be represented several times.
- Some of the attributes may be superfluous.
- Some of attributes values may be superfluous.

Issues in the Decision Table

- The **same or indiscernible** objects may be represented several times.
- Some of the attributes may be superfluous.
- Some of attributes values may be superfluous.

Core and Reduct Concepts!

Definitions

- **Reduct** is a minimal subset of attributes B from C that preserves the positive region as the whole attributes set C does.

$$RED_C\{(d)\} : POS_B\{(d)\} = POS_C\{(d)\} \tag{8}$$

Definitions

- **Reduct** is a minimal subset of attributes B from C that preserves the positive region as the whole attributes set C does.

$$RED_C\{(d)\} : POS_B\{(d)\} = POS_C\{(d)\} \quad (8)$$

- **Core** is the set of indispensable attributes. It is included in every reduct.

$$CORE_C\{(d)\} \cap RED_C\{(d)\} \quad (9)$$

Feature Selection and Reduced Set

TABLE – Decision System (DS)

Patients	Headache	Muscle-pain	Temperature	Flu
o_1	Yes	Yes	very high	Yes
o_2	Yes	No	high	Yes
o_3	Yes	No	high	No
o_4	No	Yes	normal	No
o_5	No	Yes	high	Yes
o_6	No	Yes	very high	Yes

Feature Selection and Reduced Set

Needed Calculations

- $Pos_C(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H,M\}}(\{d\}) = \{o_1\}$
- $Pos_{\{H,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{M,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H\}}(\{d\}) = \emptyset$
- $Pos_{\{M\}}(\{d\}) = \emptyset$
- $Pos_{\{T\}}(\{d\}) = \{o_1, o_4, o_6\}$

on calcul Pos pour toutes les combinaisons de features et toutes les features.

H : Headache ; M : Muscle-pain ; T : Temperature ; d : for both X_1 and X_2

Feature Selection and Reduced Set

$\rightsquigarrow$ Looking for the minimal subset having the same $Pos_C(\{d\})$.

Feature Selection and Reduced Set

$\rightsquigarrow$ Looking for the minimal subset having the same $Pos_C(\{d\})$.

- $Pos_C(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H,M\}}(\{d\}) = \{o_1\}$
- $Pos_{\{H,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{M,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H\}}(\{d\}) = \emptyset$
- $Pos_{\{M\}}(\{d\}) = \emptyset$
- $Pos_{\{T\}}(\{d\}) = \{o_1, o_4, o_6\}$

Feature Selection and Reduced Set

⤳ Looking for the minimal subset having the same $Pos_C(\{d\})$.

- $Pos_C(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H,M\}}(\{d\}) = \{o_1\}$
- $Pos_{\{H,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{M,T\}}(\{d\}) = \{o_1, o_4, o_5, o_6\}$
- $Pos_{\{H\}}(\{d\}) = \emptyset$
- $Pos_{\{M\}}(\{d\}) = \emptyset$
- $Pos_{\{T\}}(\{d\}) = \{o_1, o_4, o_6\}$

⤳ There are two reducts (two generated reduced data sets) : The selected features are **H** and **T** constituting a first reduct **{Headache, Temperature}**; or **M** and **T** to have the following reduct **{Muscle-pain, Temperature}**

Feature Selection and Reduced Set

TABLE – Reduct 1 : {Muscle-pain, Temperature}

Patients	Muscle-pain	Temperature	Flu
o_1	Yes	very high	Yes
o_2	No	high	Yes
o_3	No	high	No
o_4	Yes	normal	No
o_5	Yes	high	Yes
o_6	Yes	very high	Yes

Feature Selection and Reduced Set

TABLE – Reduct 2 : {Headache, Temperature}

Patients	Headache	Temperature	Flu
o_1	Yes	very high	Yes
o_2	Yes	high	Yes
o_3	Yes	high	No
o_4	No	normal	No
o_5	No	high	Yes
o_6	No	very high	Yes

Feature Selection and Reduced Set

TABLE – Reduct 2 : {Headache, Temperature}

Patients	Headache	Temperature	Flu
o_1	Yes	very high	Yes
o_2	Yes	high	Yes
o_3	Yes	high	No
o_4	No	normal	No
o_5	No	high	Yes
o_6	No	very high	Yes

The **Core (The most important feature set to not delete from any generated reduct !)** :
{Headache, Temperature} $\cap$ **{Muscle-pain, Temperature}** $=$ **{Temperature}**

Outline

Heuristic attribute selection

How to find the reduct ?

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Limitation. . .

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Limitation. . .

1. Expensive solution to the problem ;

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Limitation. . .

1. Expensive solution to the problem ;
2. Only practical for very simple data sets ;

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Limitation. . .

1. Expensive solution to the problem ;
2. Only practical for very simple data sets ;
3. Inquire about which is the best reduct for the classification process ;

Heuristic attribute selection

How to find the reduct ?

1. Generate all possible subsets ;
2. Retrieve those with a maximum rough set dependency degree ;

Limitation. . .

1. Expensive solution to the problem ;
2. Only practical for very simple data sets ;
3. Inquire about which is the best reduct for the classification process ;

$\leadsto$ The solution to these issues is to apply a ***"heuristic attribute selection"*** method.

Heuristic attribute selection

In literature :

- QuickReduct algorithm ;
- ReverseReduct algorithm ;
- PRESET algorithm ;
- Based on genetic algorithm ;
- Using ant-based framework ;
- Using filter-based approach ;
- Under uncertain frameworks (belief function) ;

Heuristic attribute selection

The most used heuristic is the :

- QuickReduct algorithm ;
- ReverseReduct algorithm ;
- PRESET algorithm ;
- Based on genetic algorithm ;
- Using ant-based framework ;
- Using filter-based approach ;
- Under uncertain frameworks (belief function) ;

The "QuickReduct" Algorithm

The “QuickReduct” Algorithm

1. It starts off with an empty set ;

The “QuickReduct” Algorithm

1. It starts off with an empty set ;
2. The dependency of each attribute is calculated ;

The "QuickReduct" Algorithm

1. It starts off with an empty set ;
2. The dependency of each attribute is calculated ;
3. The best candidate is chosen ;

The “QuickReduct” Algorithm

1. It starts off with an empty set ;
2. The dependency of each attribute is calculated ;
3. The best candidate is chosen ;
4. A combination of best attributes is added to the reduct ;

The “QuickReduct” Algorithm

1. It starts off with an empty set ;
2. The dependency of each attribute is calculated ;
3. The best candidate is chosen ;
4. A combination of best attributes is added to the reduct ;
5. **Back to 2** ;

The “QuickReduct” Algorithm

+rapide mais petite perte de résultat.

1. It starts off with an empty set ;
2. The dependency of each attribute is calculated ;
3. The best candidate is chosen ;
4. A combination of best attributes is added to the reduct ;
5. **Back to 2** ;
6. This process continues until the **dependency of the reduct equals the consistency of the data set** ;

The "QuickReduct" Algorithm : An Example

TABLE – Decision System

$x \in U$	a	b	c	d	e	X
x_1	1	2	4	0	1	1
x_2	0	3	3	2	1	2
x_3	2	3	1	3	3	2
x_4	1	1	2	1	2	1
x_5	0	2	0	1	2	1
x_6	1	1	2	4	3	2
x_7	2	2	1	3	2	2
x_8	1	2	0	2	2	1

The “QuickReduct” Algorithm : An Example

TABLE – Decision System

$x \in U$	a	b	c	d	e	X
x_1	1	2	4	0	1	1
x_2	0	3	3	2	1	2
x_3	2	3	1	3	3	2
x_4	1	1	2	1	2	1
x_5	0	2	0	1	2	1
x_6	1	1	2	4	3	2
x_7	2	2	1	3	2	2
x_8	1	2	0	2	2	1

- $\gamma_{\{a,b,c,d,e\}}(X) = 8/8 = 1$ **(Equ. 7)**

The "QuickReduct" Algorithm : An Example

TABLE – Decision System

$x \in U$	a	b	c	d	e	X
x_1	1	2	4	0	1	1
x_2	0	3	3	2	1	2
x_3	2	3	1	3	3	2
x_4	1	1	2	1	2	1
x_5	0	2	0	1	2	1
x_6	1	1	2	4	3	2
x_7	2	2	1	3	2	2
x_8	1	2	0	2	2	1

- $\gamma_{\{a,b,c,d,e\}}(X) = 8/8 = 1$ **(Equ. 7)**

The reduct is set to the empty set :

- $R = \emptyset$

The "QuickReduct" Algorithm : An Example

*Evaluate the dependency of each attribute **(Equ. 7)** :*

- $\gamma_{\{a\}}(X) = 2/8$
- $\gamma_{\{b\}}(X) = 2/8$
- $\gamma_{\{c\}}(X) = 6/8$
- $\gamma_{\{d\}}(X) = 6/8$
- $\gamma_{\{e\}}(X) = 2/8$

The "QuickReduct" Algorithm : An Example

Evaluate the dependency of each attribute :

- $\gamma_{\{a\}}(X) = 2/8$
- $\gamma_{\{b\}}(X) = 2/8$
- $\gamma_{\{c\}}(X) = 6/8$
- $\gamma_{\{d\}}(X) = 6/8$
- $\gamma_{\{e\}}(X) = 2/8$

As $\{c\}$ *generates the highest dependency degree, it is added to the reduct :*

- $R = \{c\}$

The "QuickReduct" Algorithm : An Example

Again, we evaluate the following dependency values :

- $\gamma_{\{c,a\}}(X) = 6/8$
- $\gamma_{\{c,b\}}(X) = 6/8$
- $\gamma_{\{c,d\}}(X) = 8/8$

Stop the iterations since the generated dependency is equal to the dependency of the whole data set

Generated reduct is the following (the selected features are the following) :

- $R = \{c, d\}$