



**République Algérienne Démocratique et populaire**  
**Ministère de l'Enseignement Supérieur et de la Recherche**  
**Scientifique**

**Université des Sciences et de la Technologie Houari Boumediene**  
**Faculté d'Electronique et Informatique**  
**Département Informatique**

# Rapport Partie 2

## Projet Data Mining

### Techniques de Datamining

**Réalisé par :**

- |                        |               |
|------------------------|---------------|
| • ADMANE Hocine        | 171731054926. |
| • HABCHI Lydia         | 161731064864. |
| • DJAOUADI Yamina      | 161631080116. |
| • TEBBANIMOHAMED WALID | 171731088266. |
| • SEDKAOUIAMINE        | 161731026140. |

Master spécialité : SII.

Année universitaire : 2021/2022.

## Table des matières

### Introduction.

#### A. Prétraitement des données.

##### A.1 Normalisation des données.

##### A.2 Discrétisation des données.

#### B. Extraction des motifs fréquents, des règles d'association et des règles de corrélation.

##### B.1 Algorithme Apriori.

##### B.2 Algorithme Eclat.

##### B.3 Algorithme d'extraction des règles d'association.

##### B.4 Algorithme d'extraction de règles de corrélation.

#### C. Classification supervisée des instances du dataset.

##### C.1 Programmer les algorithmes de classification.

###### C.1.1 Algorithme Naïve Bayésienne.

###### C.1.2 Algorithme KNN.

##### C.2 Construire la matrice de confusion.

##### C.3 Comparaison des deux algorithmes.

### Conclusion

## Introduction :

Le but de la deuxième partie est de mettre en pratique les techniques de data mining vues en cours à savoir :

- L'intégration, la sélection et la transformation des données qui comprend :
  - La normalisation des données + le but (mettre en relation les attributs entre eux, effectuer les mêmes opérations et obtenir des résultats cohérents et compatibles).
  - La discrétisation + le but (éliminer les valeurs redondantes).
- La détermination des motifs fréquents et des règles d'association et de corrélation + but (déterminer les préférences et les relations/comбинаisons).
- La classification des données + but (extraire les connaissances).

## A. Prétraitement des données :

### A.1 Normalisation des données :

L'intérêt d'effectuer la normalisation est de changer l'intervalle des données, il y'a plusieurs méthodes de discrétisation, on va exposer deux d'entre elles, la normalisation min-max et la normalisation z-score :

- **Normalisation min-max** : elle est sous la formule suivante :

$$newV = newMin - (newMax - newMin) \frac{currentV - oldMin}{oldMax - oldMin}.$$

- **Normalisation z-score** : elle est sous la formule suivante :

$$newV = \frac{currentV - \bar{X}}{S}, \quad S = \frac{1}{n} \sum_{i=1}^n |X_i - \bar{X}|.$$

## A.2 Discrétisation des données :

La discrétisation est faite dans le but de diminuer la taille de l'ensemble de données et le simplifier et d'éliminer les redondances, on a utilisé les méthodes de discrétisation simple et ce en créant des intervalles soit en classes d'effectifs égaux ou en classes d'amplitudes égales, on décrit ces méthodes ci-dessous :

- **Discrétisation en classes d'effectifs égaux :**

1. Diviser le nombre de valeurs sur Q (le nombre d'intervalles), pour trouver le nombre de valeurs dans chaque intervalle.

2. Déterminer la position  $i$  du  $K^{\text{e}}$  élément tel que :

$$i = \frac{n}{Q} K, \quad \begin{cases} i = i + 1 & \text{si } i > 5 \\ i = i & \text{sinon} \end{cases}.$$

3. Ajouter la  $K^{\text{eme}}$  valeur se trouvant à la position  $i$  dans l'attribut aux quantiles.

4. A la fin on ajoute chaque valeur de l'attribut dans l'intervalle auquel elle appartient,

$$\text{C-à-d : } BI \leq \text{valeur} < BS.$$

$BI$  : borne inférieure,  $BS$  : borne supérieure.

- **Discrétisation en classes d'amplitudes égales :**

1. Diviser le nombre de valeurs sur Q (le nombre d'intervalles), pour trouver l'amplitude de chaque intervalle.

2. Déterminer les bornes de chaque intervalle, et la différence entre la borne inférieure et la borne supérieure de chaque intervalle est égale à l'amplitude.

3. A la fin on ajoute chaque valeur de l'attribut dans l'intervalle auquel elle appartient.

## B. Extraction des motifs fréquents, des règles d'association et des règles de corrélation :

Pour extraire les motifs fréquents, les règles d'association et les règles de corrélation d'un ensemble de données, on va utiliser 2 algorithmes dans notre projet qui sont : l'algorithme Apriori et l'algorithme Eclat, leur description est donnée ci-dessous :

## B.1 Algorithme Apriori :

On est passé par les étapes suivantes :

- initialement, scanner l'ensemble de données une fois pour obtenir les k-itemSet fréquents,  $k=1$ .
- A chaque fois on génère les candidats avec  $(k+1)$  itemSet en calculant leurs supports et on élimine à chaque génération les candidats ayant un support  $< \text{sup}_{\min}$ .
- on s'arrête quand aucun candidat n'est généré.

### Algorithme : Apriori

**Entrée :** T : Table de transactions (Matrice) ;  $\text{Min}_{\text{Sup}}$  : le support minimum ;

**Sortie :** L : l'ensemble des motifs fréquents ;

**Var**

$C_1, C_2, \dots, C_k$  : Matrice des candidats (itemsets - support) ;

$L_1, L_2, \dots, L_k$  : Tableaux des motifs fréquents (itemsets) ;

**Début**

*/\*Construction des candidats  $C_1$  (1-itemsets) \*/*

Calculer pour chaque item le nombre d'apparitions (support) ;

*/\*Réduction des candidat  $C_1$  en liste de motifs fréquents  $L_1$  \*/*

**Pour chaque candidat c de  $C_1$  faire**

**Si**  $\text{support}(c) \geq \text{Min}_{\text{Sup}}$  **Alors**  $L_1 \leftarrow L_1 \cup c$  ; **FinSi** ;

**Fait** ;

*/\*Réitération du processus avec les 2-itemsets, 3-itemsets, ..., k-itemsets\*/*

$k \leftarrow 1$  ;

**Tant que**  $L_k \neq \emptyset$  **faire**

*/\*sauvegarde de l'ensemble des motifs fréquents\*/*

$L \leftarrow L \cup L_k$  ;

$k \leftarrow k + 1$  ;

*/\*Générer toutes les combinaisons possibles formant des k-itemsets\*/*

Générer les candidats  $C_k$  à partir de la jointure de  $L_{k-1} * L_{k-1}$  ;

*/\*Extraction des k-itemsets fréquents :  $L_k$  à partir de  $C_k$  \*/*

**Pour chaque candidat c de  $C_k$  faire**

**Si**  $\text{support}(c) \geq \text{Min}_{\text{Sup}}$  **Alors**  $L_k \leftarrow L_k \cup c$  ; **FinSi** ;

**Fait** ;

**Fait** ;

**Retourner** L ;

**Fin.**

### **Avantage :**

- Simple

### **Inconvénients :**

- Coûteux en temps de calcul et en espace mémoire car on doit faire plusieurs scans sur l'ensemble de données et générer plusieurs candidats.

## **B.2 Algorithme Eclat :**

L'algorithme Eclat consiste à inverser l'ensemble de données de telle sorte à faire des vérifications verticales pour chaque item en sauvegardant toutes les instances dans lesquelles il apparait et calculer leur nombre comme étant le support de l'item puis supprimer les itemSet ayant un  $\text{sup} < \text{sup}_{\min}$ . Répéter cette opération pour chaque k-itemSet jusqu'à ce qu'il n'y ait aucun candidat généré.

## Algorithme : Eclat

**Entrée :** T : Table de transactions (Matrice) ;  $\text{Min}_{\text{Sup}}$  : le support minimum ;

**Sortie :** L : l'ensemble des motifs fréquents ;

### Var

$C_1, C_2, \dots, C_k$  : Matrice des candidats (itemsets - support);

$L_1, L_2, \dots, L_k$  : Tableaux des motifs fréquents (itemsets);

tableVerticale : matrice booléenne ;

### Début

tableVerticale  $\leftarrow$  Représentation verticale des données de transaction ;

*/\*Construction des candidats  $C_1$  (1-itemsets) \*/*

Calculer le support de chaque itemset à partir de *tableVerticale* ;

*/\*Réduction des candidats  $C_1$  en liste de motifs fréquents  $L_1$  \*/*

**Pour chaque** candidat c **de**  $C_1$  **faire**

**Si** support(c)  $\geq \text{Min}_{\text{Sup}}$  **Alors**  $L_1 \leftarrow L_1 \cup c$  ; **FinSi** ;

**Fait** ;

*/\*Réitération du processus avec les 2-itemsets, 3-itemsets, ..., k-itemsets\*/*

$k \leftarrow 1$  ;

**Tant que**  $L_k \neq \emptyset$  **faire**

$L \leftarrow L \cup L_k$  ; */\*sauvegarde de l'ensemble des motifs fréquents\*/*

$k \leftarrow k + 1$  ;

*/\*Générer toutes les combinaisons possibles formant des k-itemsets\*/*

Générer les candidats  $C_k$  à partir de la jointure de  $L_{k-1} * L_{k-1}$  ;

*/\*Calculer pour chaque itemset de  $C_k$  le nombre de transaction en commun\*/Calculer le nombre de 1 résultant de l'intersection des items le composant ;*

*/\*Extraction des k-itemsets fréquents :  $L_k$  à partir de  $C_k$  \*/*

**Pour chaque** candidat c **de**  $C_k$  **faire**

**Si** support(c)  $\geq \text{Min}_{\text{Sup}}$  **Alors**  $L_k \leftarrow L_k \cup c$  ; **FinSi** ;


**Fait** ;

**Fait** ;

**Retourner** L ;

**Fin.**

TID	Itemsets
1	a,b,c
2	a,b
3	a,c,d
4	a,b,d
5	a,c

Représentation  
  
 verticale

Itemsets	TID Set
a	1, 2, 3, 4, 5
b	1, 2, 4
c	1, 5
d	3, 4

### Avantage :

- La structure verticale de la base de transitions fait gagner du temps de calcul.

### Inconvénients :

- Reste relativement coûteux en temps de calcul et en mémoire car on doit générer plusieurs candidats.

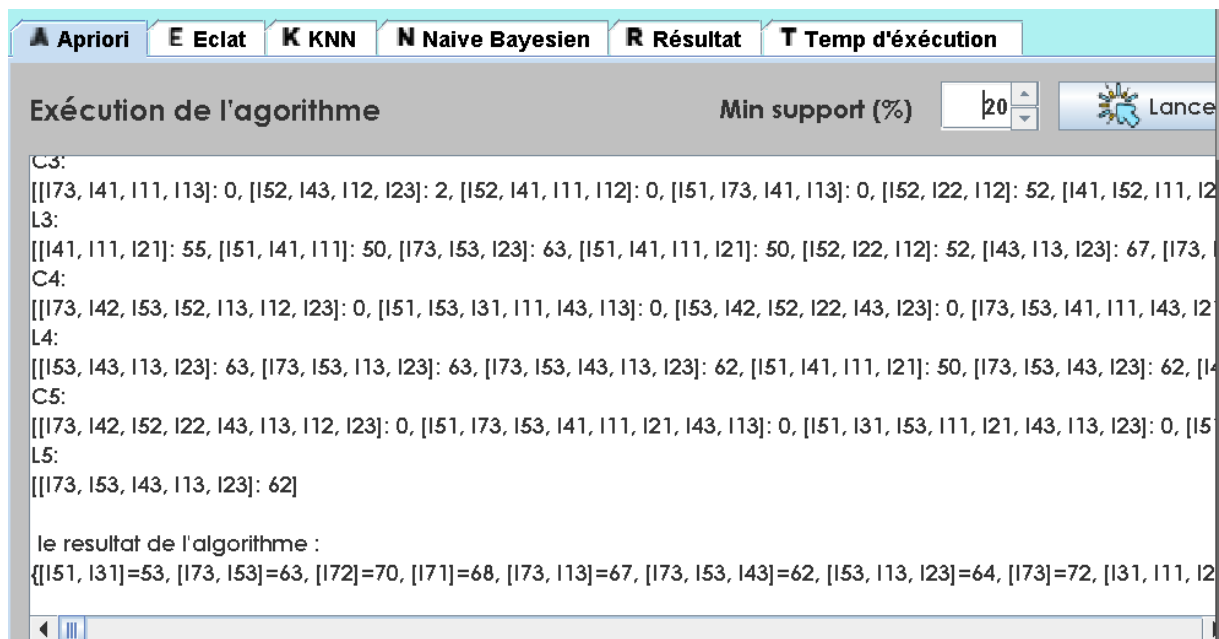
### Application sur les instances du dataSet :

- Voici les captures d'écran des résultats de l'application de l'algorithme Apriori sur les instances du Dataset avec :

**Normalisation** : Z-Score ;

**Discrétisation** : en classes d'effectifs égaux ;

**MinSup** : 20%.



**Exécution de l'agorithme** Min support (%) 20 Lance

C3:  
 [[I73, I41, I11, I13]: 0, [I52, I43, I12, I23]: 2, [I52, I41, I11, I12]: 0, [I51, I73, I41, I13]: 0, [I52, I22, I12]: 52, [I41, I52, I11, I2]

L3:  
 [[I41, I11, I21]: 55, [I51, I41, I11]: 50, [I73, I53, I23]: 63, [I51, I41, I11, I21]: 50, [I52, I22, I12]: 52, [I43, I13, I23]: 67, [I73, I

C4:  
 [[I73, I42, I53, I52, I13, I12, I23]: 0, [I51, I53, I31, I11, I43, I13]: 0, [I53, I42, I52, I22, I43, I23]: 0, [I73, I53, I41, I11, I43, I2]

L4:  
 [[I53, I43, I13, I23]: 63, [I73, I53, I13, I23]: 63, [I73, I53, I43, I13, I23]: 62, [I51, I41, I11, I21]: 50, [I73, I53, I43, I23]: 62, [I4

C5:  
 [[I73, I42, I52, I22, I43, I13, I12, I23]: 0, [I51, I73, I53, I41, I11, I21, I43, I13]: 0, [I51, I31, I53, I11, I21, I43, I13, I23]: 0, [I5

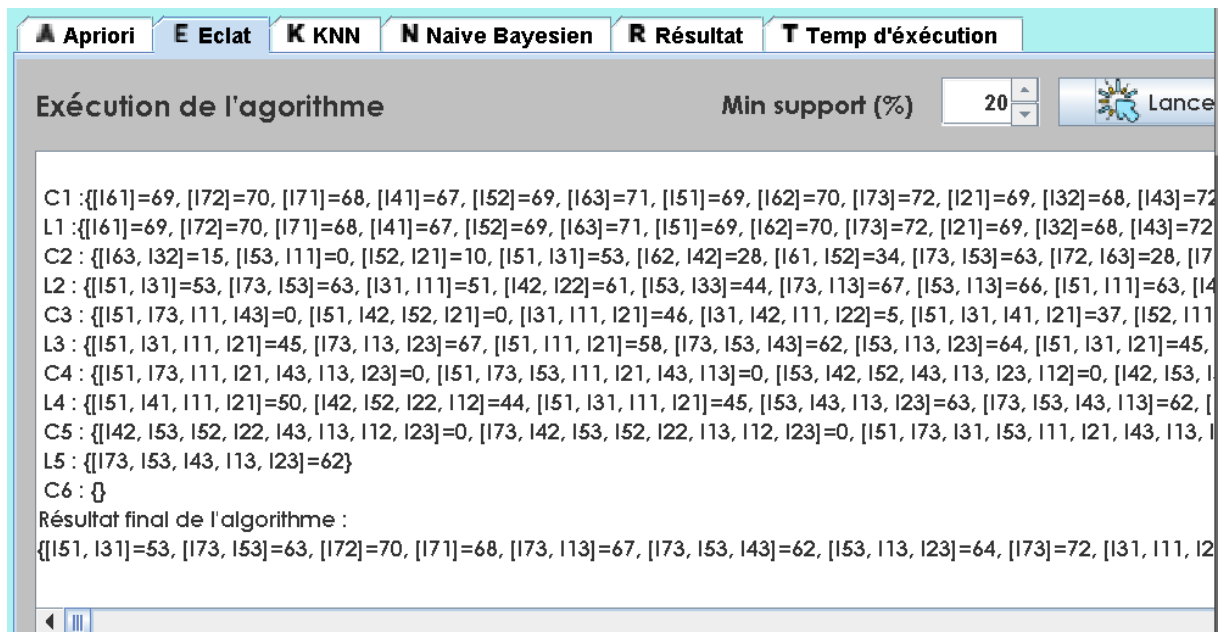
L5:  
 [[I73, I53, I43, I13, I23]: 62]

le resultat de l'algorithme :  
 {[I51, I31]=53, [I73, I53]=63, [I72]=70, [I71]=68, [I73, I13]=67, [I73, I53, I43]=62, [I53, I13, I23]=64, [I73]=72, [I31, I11, I2}

### Résultats de l'exécution de l'algorithme Apriori.



- Voici les captures d'écran des résultats de l'application de l'algorithme Eclat sur les instances du dataSet avec :



#### Résultats de l'exécution de l'algorithme Eclat.

- Diagrammes comparant le temps d'exécution des algorithmes Eclat et Apriori sur les instances du dataSet avec :

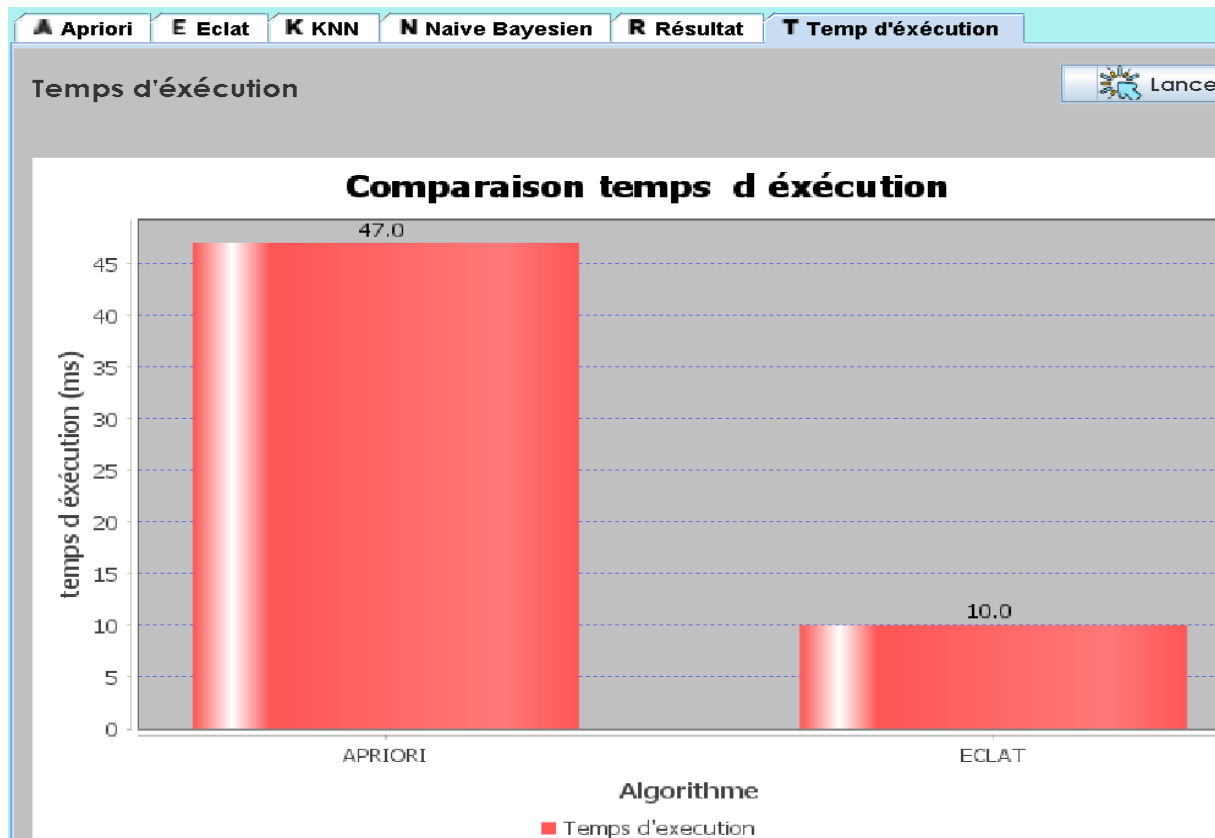


Diagramme de temps d'exécution de l'algorithme Eclat et l'algorithme Apriori.

### B.3 Algorithme d'extraction des règles d'association :

Elles permettent de déterminer la dépendance entre les items dans les transactions (les instances dans notre cas) et produire des règles, suivant des seuils définis qui sont le MinSup et MinConf, on compare le support d'une règle au MinSup, si ce dernier est  $>$  au support alors on élimine cette règle, de la même façon on compare la confiance d'une règle par rapport à MinConf et on élimine les règles qui contrent la condition. L'idée est surtout de limiter le nombre de règles produites.

**1. Support :** Le support d'une règle est le support de l'élément à droite de la règle divisée par le nombre total de transactions, et c'est un indicateur de fiabilité de la règle, voici la règle :

$$S_{X \rightarrow Y} = \frac{\text{sup}(\{X, Y\})}{N}.$$

$N$  : étant le nombre de transactions totales.

$\text{sup}(X)$  : le nombre de transactions dans lesquelles  $X, Y$  apparaissent ( $X \cup Y$ ).

**2. Confiance :** si dans une règle on a  $X$  et  $Y$ , on calcule la confiance de cette règle avec la condition : si  $X$  appartient à une transaction alors cette transaction doit aussi contenir  $Y$ . c'est un indicateur de précision de la règle :

$$C_{X \rightarrow Y} = \frac{\text{sup}(\{X, Y\})}{\text{sup}(X)}.$$

## Algorithme : extraction des règles d'association

### Entrée :

Min<sub>Sup</sub> : le support minimum ;  
Min<sub>confidence</sub> : la confiance minimale ;  
L : l'ensemble des motifs fréquents avec leurs supports ;

### Sortie :

LRS : l'ensemble des règles d'association vérifiant le support minimal ;  
LRC : l'ensemble des règles d'association vérifiant la confiance minimale ;

### Var

Items : tableaux d'items ;  
Combinations : matrice booléenne ;

### Début

#### Pour chaque itemSet de L faire

/\*décomposer l'itemSet en un ensemble d'items et les mettre dans un tableau\*/  
Items  $\leftarrow$  tous les items de itemSet ;

/\*construire toutes les combinaisons possibles de tous les items de Items (on obtient  $2^{\text{card}(\text{itemSet})}$  règles)  $\rightarrow \text{card}(\text{itemSet})$  : le nombre d'items \*/  
Combinations  $\leftarrow$  toutes les combinaisons possibles de tous les items de Items ;

/\*calcul du support pour chaque règle (combinaison) \*/

#### Pour chaque règle de combinaisons faire

/\*récupérer le support de la règle \*/  
Sup  $\leftarrow$  son support dans L ;

/\*calculer le support de la règle en divisant par le nombre total de transactions N \*/  
Sup(règle)  $\leftarrow \text{sup} / N$  ;

/\*ajouter la règle et le résultat à l'ensemble LRS si support de la règle  $\geq \text{Min}_{\text{Sup}}$  \*/  
Si (sup(règle)  $\geq \text{Min}_{\text{Sup}}$ ) alors ajouter la règle à LRS ;

/\*calculer la confiance de la règle en divisant par le support de l'élément gauche de la règle \*/  
Sup (ElementGauche)  $\leftarrow$  son support dans L ;  
Conf (règle)  $\leftarrow \text{sup} / \text{Sup} (\text{ElementGauche})$  ;

/\*ajouter la règle et le résultat à l'ensemble LRC si confiance de la règle  $\geq \text{Min}_{\text{confidence}}$  \*/  
Si (conf (règle)  $\geq \text{Min}_{\text{confidence}}$ ) alors ajouter la règle à LRC ;

Fait ;

Fait ;

Fin.

### 3. Application sur les instances du dataSet :

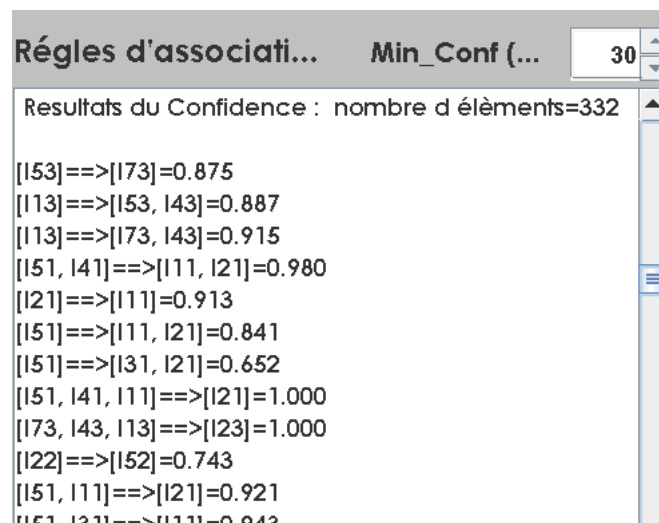
Voici les captures d'écran de quelques résultats de l'application de l'algorithme d'association sur les instances du dataSet avec :

**Normalisation** : Z-Score ;

**Discretisation** : en classes d'effectifs égaux ;

**MinSup** : 20% ;

**Minconfidence** : 30%.



**Résultats de l'exécution de l'algorithme extraction des règles d'association.**

### B.4 Algorithme d'extraction de règles de corrélation :

- Lift** : permet de calculer la corrélation entre deux éléments, il est calculé avec les supports en pourcentage :

$$Lift_{X \rightarrow Y} = \frac{sup(\{X,Y\})}{sup(X)*sup(Y)} \rightarrow \frac{C_{X \rightarrow Y}}{sup(Y)}.$$

$C_{X \rightarrow Y}$  : c'est la confiance de la règle ( $X \rightarrow Y$ ).

**Lift**  $\approx 1$  : indique que la relation est produite au hasard.

**Lift**  $> 1$  : la corrélation entre X et Y est positive.

**Lift**  $< 1$  : indique qu'il y'a une faible relation.

### Algorithme : extraction des règles de corrélation.

#### Entrée :

LRS : l'ensemble des règles d'association vérifiant le support minimal ;  
LRC : l'ensemble des règles d'association vérifiant la confiance minimale ;

#### Sortie :

LRLift : l'ensemble des règles de corrélation ( $\text{lift} > 1$ ) ;

#### Début

##### Pour chaque règle de LRS faire

/\* récupérer sa confiance à partir de **LRC** \*/

Conf (règle)  $\leftarrow$  la confiance de la règle dans LRC ;

/\* récupérer le support de l'élément à droite dans **LRS** \*/

Sup (ElémentDroite)  $\leftarrow$  le support de l'élément à droite dans LRS ;

/\* calculer le lift de la règle en divisant sa confiance par le support de **ElémentDroite** \*/

Lift (règle)  $\leftarrow$  Conf (règle) / Sup (ElémentDroite) ;

/\* ajouter la règle et le résultat à l'ensemble **LRLift** si lift de la règle  $> 1$  \*/

Si (lift (règle)  $> 1$ ) alors ajouter la règle à LRLift ;

Fait ;

Fin.

## 2. Application sur les instances du dataSet :

Voici les captures d'écran de quelques résultats de l'application de l'algorithme de corrélation sur les instances du dataSet avec :

**Normalisation** : Z-Score ;

**Discretisation** : en classes d'effectifs égaux ;

**MinSup** : 20% ;

**Minconfidence** : 30%.



nts=332

### Règles de corrélation

les règles positivement corrélées : nombre d'élève

```
[153]==>[173]=2.552  
[113]==>[153, 143]=2.958  
[113]==>[173, 143]=2.786  
[151, 141]==>[111, 121]=3.268  
[121]==>[111]=2.820  
[151]==>[111, 121]=2.802  
[151]==>[131, 121]=2.977  
[151, 141, 111]==>[121]=3.043  
[173, 143, 113]==>[123]=2.958  
[122]==>[152]=2.261  
[151, 111]==>[121]=2.802
```

**Résultats de l'exécution de l'algorithme extraction des règles de corrélation.**

## C. Classification supervisée des instances du dataset :

### C.1 Algorithme d'apprentissage supervisé :

En apprentissage supervisé, un algorithme reçoit un ensemble de données qui est étiqueté avec des valeurs de sorties correspondantes sur lequel il va pouvoir s'entraîner et définir un modèle de prédiction. Cet algorithme pourra par la suite être utilisé sur de nouvelles données afin de prédire leurs valeurs de sorties correspondantes.

Parmi les algorithmes de classification supervisés on a les deux algorithmes de Naïve Bayésienne et KNN.

#### C.1.1 La classification naïve bayésienne :

Les classificateurs de Naïve Bayes sont une famille d'algorithmes reposant sur le principe commun selon lequel la valeur d'une fonctionnalité spécifique est indépendante de la valeur de toute autre fonctionnalité. Ils nous permettent de prédire la probabilité qu'un événement se produise en fonction de conditions que nous connaissons pour les événements en question. Le nom vient du théorème de Bayes, qui peut être écrit mathématiquement comme suit:

$$P(B|A) = \frac{P(A|B)P(B)}{P(A)}$$

Avec A et B les événements et P(B) supérieure strict à 0.

- $P(A|B)$  est une probabilité conditionnelle. Plus précisément, c'est la probabilité que l'événement A se produise sachant que B l'événement s'est déjà produit.
- Idem,  $P(B|A)$  est une probabilité conditionnelle. Plus précisément, c'est la probabilité que l'événement B se produise sachant que A l'événement s'est déjà produit.
- $P(A)$  et  $P(B)$  sont les probabilités des événements A et B indépendamment les uns des autres.

#### a. Algorithme naïve bayésienne :

**P (a<sub>i</sub>)** : la probabilité d'avoir la valeur a<sub>i</sub> pour l'attribut i.

**P (a<sub>i</sub> / b<sub>j</sub>)** : la probabilité d'avoir la valeur a<sub>i</sub> pour l'attribut i, sachant qu'on a la valeur b<sub>j</sub> pour l'attribut j.

Etant donné une instance  $X = \{x_1, x_2, \dots, x_n\}$  à classer.

La classification bayésienne naïve cherche à calculer pour chaque classe C<sub>i</sub> :

**P (C<sub>i</sub> / X)** : la probabilité d'avoir une classe C<sub>i</sub> sachant une donnée X.

Avec:

$$P(C_i / X) = P(C_i) * P(x_1 / C_i) * P(x_2 / C_i) * P(x_3 / C_i) * \dots * P(x_n / C_i)$$

Puis, il faut choisir la classe C<sub>i</sub> qui maximise cette probabilité **P (C<sub>i</sub> / X)**.

$$y = \underset{j=1}{\operatorname{argmax}_{C_i}} P(C_i) * \prod_{j=1}^n P(x_j / C_i)$$

#### Remarque :

Afin d'éviter le problème de la probabilité zéro, nous pouvons utiliser l'*Estimateur de Laplace*.

Incrémenter de 1 le nombre d'instances de chaque probabilité et le nombre d'instances global.

#### b. Avantages et inconvénients de Naïve Bayes :

##### Avantage :

- C'est relativement simple à comprendre et à construire.
- Il est facile à former, même avec un petit jeu de données.
- C'est rapide.

- Il n'est pas sensible aux caractéristiques non pertinentes.

### Inconvénients :

- Il implique que chaque fonctionnalité soit indépendante, ce qui n'est pas toujours le cas.


### c. Application sur le DATASET :

Voici les captures d'écran des résultats de l'application de l'algorithme Naïve Bayésienne sur les instances du dataSet avec :

**Normalisation** : Z-Score ;

**Discretisation** : en classes d'effectifs égaux ;

20 instances.

<span>▲ Apriori</span> <span>E Eclat</span> <span>K KNN</span> <span>N Naive Bayésien</span> <span>R Résultat</span> <span>T Temp d'exécution</span>								
Exécution de l'agorithme								
nbr instan...		20	premières instances					
A	P	C	LoK	WoK	AC	LoKG	Class reel	Class pred
I12	I22	I32	I42	I52	I61	I72	Kama	Kama
I12	I22	I32	I42	I52	I61	I71	Kama	Kama
I12	I22	I33	I41	I52	I61	I71	Kama	Kama
I12	I22	I33	I41	I52	I61	I71	Kama	Kama
I13	I22	I33	I42	I53	I61	I72	Kama	Kama
I12	I22	I33	I42	I52	I61	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I72	Kama	Kama
I12	I22	I33	I42	I52	I61	I71	Kama	Kama
I13	I23	I32	I43	I53	I61	I73	Kama	Rosa
I13	I23	I33	I43	I53	I61	I73	Kama	Rosa
I12	I22	I32	I42	I52	I63	I72	Kama	Kama
I12	I22	I32	I42	I52	I61	I71	Kama	Kama
I12	I22	I33	I42	I52	I62	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I71	Kama	Kama

Résultats de l'exécution de l'algorithme Naïve Bayésienne.



### C.1.2 Algorithme des K plus proches voisins « KNN »

L'algorithme des K plus proches voisins ou K-nearest Neighbors (kNN) est un algorithme de Machine Learning qui appartient à la classe des algorithmes d'apprentissage supervisé simple et facile à mettre en œuvre qui peut être utilisé pour résoudre les problèmes de classification et de régression.

L'intuition derrière l'algorithme des K plus proches voisins est l'une des plus simples de tous les algorithmes de Machine Learning supervisé :

**Étape 1 :** Sélectionnez le nombre K de voisins.

**Étape 2 :** Calculez la distance Du point non classifié aux autres points.

**Étape 3 :** Prenez les K voisins les plus proches selon la distance calculée.

**Étape 4 :** Parmi ces K voisins, comptez le nombre de points appartenant à chaque catégorie.

**Étape 5 :** Attribuez le nouveau point à la catégorie la plus présente parmi ces K voisins.

**Étape 6 :** Notre modèle est prêt.

#### Algorithme : k-NN

**Entrée :**  $D$  : Dataset ;

$K$  : nombre de voisins à considérer ;

$Inst$  : l'instance à classifier ;

**Sortie :**  $y$  : la classe de l'instance à classifier ;

**Var**

$dist$  : tableau de  $[1..N]$  de pair (instance, distance) ; //avec  $N$  la taille de  $D = |D|$  ;

$knn$  : tableau de  $[1..k]$  d'instances ;

**Début**

**Pour chaque** instance  $X$  **de**  $D$  **faire**

$dist[X] \leftarrow$  Calculer la distance entre  $X$  et  $Inst$  ;

**Fait ;**

$dist \leftarrow$  Trier  $dist$  dans l'ordre croissant des distances ; \*

$knn \leftarrow$  Les  $k$  premières instances de  $dist$  ; \*

$y \leftarrow$  La classe dominante dans  $knn$  ;

**Retourner**  $y$  ;

**Fin.**

**Note :** \* ou bien juste récupérer dans la variable  $knn$  directement les  $k$  instances ayant une distance minimale.

### a. Exemple d'utilisation :

Dans ce rapport on s'intéresse à un exemple d'utilisation de l'algorithme des K plus proches voisins sur notre Dataset « seeds ».

Grâce à l'algorithme KNN, nous obtenons un excellent taux de bonne classification.

On peut également s'intéresser à un moyen de choisir :

L'ensemble de test (T), le K pour lequel la classification sera la meilleure ainsi que les Mesures de distance.

#### Choix de L'ensemble de test (T) :

##### Cas n°1 : 20 instances de chaque classe

A	P	C	LoK	WoK	AC	LoKG	Class reel	Class pred
I12	I22	I32	I42	I52	I61	I72	Kama	Kama
I12	I22	I32	I42	I52	I61	I71	Kama	Kama
I12	I22	I33	I41	I52	I61	I71	Kama	Kama
I12	I22	I33	I41	I52	I61	I71	Kama	Kama
I13	I22	I33	I42	I53	I61	I72	Kama	Kama
I12	I22	I33	I42	I52	I61	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I72	Kama	Kama
I12	I22	I33	I42	I52	I61	I71	Kama	Kama
I13	I23	I32	I43	I53	I61	I73	Kama	Rosa
I13	I23	I33	I43	I53	I61	I73	Kama	Rosa
I12	I22	I32	I42	I52	I63	I72	Kama	Kama
I12	I22	I32	I42	I52	I61	I71	Kama	Kama
I12	I22	I33	I42	I52	I62	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I71	Kama	Kama
I12	I22	I32	I42	I52	I62	I71	Kama	Kama

##### Cas n°2 : 25 instances de chaque classe

#### Le nombre K de voisins :

La valeur de k est un des paramètres à déterminer lors de l'utilisation de ce type de méthode.

La valeur que l'on choisit pour k va être plus critique, plus déterminante en rapport avec la performance du classificateur. On peut se permettre de considérer un plus grand nombre de voisins, sachant que plus ils diffèrent du document à classer, moins ils ont d'impact sur la prise de décision. Cependant, il demeure nécessaire de limiter le nombre de voisins pour s'en tenir à un temps de calcul raisonnable.

L'emploi de  $k$  voisins, au lieu d'un seul, assure une plus grande robustesse à la prédiction. Classiquement, dans le cas où la variable à prédire comporte deux étiquettes, ce paramètre  $k$  est une valeur impaire afin d'avoir une majorité plus facilement décidable.

Cas n°1 :  $K = 4$ ,

Exécution de l'algorithme

nbr instance : 20 | premières instances | Euclidienne | Lancer

A	P	C	LoK	WoK	AC	LoKG	Class reel	Class prédit
0.17	0.25	0.0	0.36	0.17	-1.21	0.36	Kama	Kama
0.01	0.01	0.53	-0.2	0.23	-2.19	-0.2	Kama	Kama
-0.22	-0.42	1.8	-0.89	0.24	-0.82	-0.89	Kama	Kama
-0.4	-0.55	1.29	-0.81	0.37	-1.17	-0.81	Kama	Kama
0.52	0.38	1.71	0.08	0.94	-1.91	0.08	Kama	Kama
-0.19	-0.31	1.27	-0.64	0.17	-1.01	-0.64	Kama	Kama
-0.06	-0.06	0.47	-0.17	0.0	-0.09	-0.17	Kama	Kama
-0.3	-0.41	1.06	-0.55	0.13	-0.82	-0.55	Kama	Kama
0.71	0.8	0.2	1.12	0.64	-1.35	1.12	Kama	Rosa
0.64	0.61	0.9	0.68	0.76	-1.41	0.68	Kama	Kama
0.17	0.26	-0.07	0.23	-0.05	0.69	0.23	Kama	Rosa
-0.33	-0.35	0.45	-0.5	-0.18	-1.62	-0.5	Kama	Kama
-0.38	-0.48	0.9	-0.5	-0.18	0.23	-0.5	Kama	Kama
-0.43	-0.44	0.26	-0.4	-0.32	-0.46	-0.4	Kama	Kama
-0.44	-0.45	0.18	-0.39	-0.45	-0.63	-0.39	Kama	Kama

Matrice de confusion

Nbr K-voisins : 4

Matrice du confusion :

[16, 2, 2]  
 [1, 19, 0]  
 [0, 0, 20]  
 Kama :  
 [16, 4]  
 [1, 39]  
 Rosa :  
 [19, 1]  
 [2, 38]  
 Canadian:  
 [20, 0]  
 [2, 38]

Cas n°2 :  $K = 5$ ,

Exécution de l'algorithme

nbr instance : 20 | premières instances | Euclidienne | Lancer

A	P	C	LoK	WoK	AC	LoKG	Class reel	Class prédit
0.17	0.25	0.0	0.36	0.17	-1.21	0.36	Kama	Kama
0.01	0.01	0.53	-0.2	0.23	-2.19	-0.2	Kama	Kama
-0.22	-0.42	1.8	-0.89	0.24	-0.82	-0.89	Kama	Kama
-0.4	-0.55	1.29	-0.81	0.37	-1.17	-0.81	Kama	Kama
0.52	0.38	1.71	0.08	0.94	-1.91	0.08	Kama	Kama
-0.19	-0.31	1.27	-0.64	0.17	-1.01	-0.64	Kama	Kama
-0.06	-0.06	0.47	-0.17	0.0	-0.09	-0.17	Kama	Kama
-0.3	-0.41	1.06	-0.55	0.13	-0.82	-0.55	Kama	Kama
0.71	0.8	0.2	1.12	0.64	-1.35	1.12	Kama	Rosa
0.64	0.61	0.9	0.68	0.76	-1.41	0.68	Kama	Kama
0.17	0.26	-0.07	0.23	-0.05	0.69	0.23	Kama	Rosa
-0.33	-0.35	0.45	-0.5	-0.18	-1.62	-0.5	Kama	Kama
-0.38	-0.48	0.9	-0.5	-0.18	0.23	-0.5	Kama	Kama
-0.43	-0.44	0.26	-0.4	-0.32	-0.46	-0.4	Kama	Kama
-0.44	-0.45	0.18	-0.39	-0.45	-0.63	-0.39	Kama	Kama

Matrice de confusion

Nbr K-voisins : 5

Matrice du confusion :

[15, 3, 2]  
 [0, 20, 0]  
 [0, 0, 20]  
 Kama :  
 [15, 5]  
 [0, 40]  
 Rosa :  
 [20, 0]  
 [2, 38]  
 Canadian:  
 [20, 0]  
 [2, 38]

### Cas n°3 : $K = 10$

A Apriori

E Eclat

**K KNN**

N Naive Bayesien

R Résultat

T Temp d'exécution

Exécution de l'algorithme

nbr instance


20

premières instances

▼

Euclidienne

▼

 Lancer

A	P	C	LoK	WoK	AC	LoKG	Class reel	Class prédit
0.17	0.25	0.0	0.36	0.17	-1.21	0.36	Kama	Kama
0.01	0.01	0.53	-0.2	0.23	-2.19	-0.2	Kama	Kama
-0.22	-0.42	1.8	-0.89	0.24	-0.82	-0.89	Kama	Kama
-0.4	-0.55	1.29	-0.81	0.37	-1.17	-0.81	Kama	Kama
0.52	0.38	1.71	0.08	0.94	-1.91	0.08	Kama	Kama
-0.19	-0.31	1.27	-0.64	0.17	-1.01	-0.64	Kama	Kama
-0.06	-0.06	0.47	-0.17	0.0	-0.09	-0.17	Kama	Kama
-0.3	-0.41	1.06	-0.55	0.13	-0.82	-0.55	Kama	Kama
0.71	0.8	0.2	1.12	0.64	-1.35	1.12	Kama	Rosa
0.64	0.61	0.9	0.68	0.76	-1.41	0.68	Kama	Kama
0.17	0.26	-0.07	0.23	-0.05	0.69	0.23	Kama	Kama
-0.33	-0.35	0.45	-0.5	-0.18	-1.62	-0.5	Kama	Kama
-0.38	-0.48	0.9	-0.5	-0.18	0.23	-0.5	Kama	Kama
-0.43	-0.44	0.26	-0.4	-0.32	-0.46	-0.4	Kama	Kama
-0.44	-0.45	0.18	-0.39	-0.45	-0.63	-0.39	Kama	Kama

Matrice de confusion

Nbr K-voisins

10

Matrice du confusion :

[18, 1, 1]

[0, 20, 0]

[0, 0, 20]

Kama :

[18, 2]

[0, 40]

Rosa :

[20, 0]

[1, 39]

Canadian:

[20, 0]

[1, 39]

### Remarque :

On peut avoir un conflit du choix de classe au niveau de la classification d'une instance même si on choisit un nombre impair de voisins  $k$  car dans notre cas notre Dataset est multi-classe. Pour résoudre ce conflit on a plusieurs solution parmi ces solution on a choisi de choisir une classe d'une manière aléatoire pour classifier notre instance.

## Mesures de distance :

## Cas n°1 : Euclidienne,

La distance Euclidienne est une distance géométrique dans cet espace multidimensionnel. Il est calculé comme :

$$\sum_{i=1}^n |x_i - y_i|$$

Notez que les distances Euclidiennes (et Euclidienne carré) sont calculées à partir des données brutes, et non à partir des données centrées-réduites.

## Cas n°2 : Manhattan

Distance du City-block (Manhattan). Cette distance est simplement la somme des différences entre les dimensions. Dans la plupart des cas, cette mesure de distance produit des résultats proches de ceux obtenus par la distance euclidienne simple. En revanche, notez qu'avec cette mesure, l'effet des différences simples importantes (points atypiques) est atténué (puisque ces distances ne sont pas élevées au carré). Cette distance se calcule ainsi :

$$\sqrt{\sum_{i=1}^n (x_i - y_i)^2}$$

### b. Avantages et Inconvénients de l'algorithme KNN :

#### Avantage :

- L'algorithme est simple et facile à mettre en œuvre.
- Il n'est pas nécessaire de créer un modèle, de régler plusieurs paramètres ou de formuler des hypothèses supplémentaires.
- L'algorithme est polyvalent. Il peut être utilisé pour la classification ou la régression.

#### Inconvénients :

- L'algorithme devient beaucoup plus lent à mesure que le nombre d'observation et de variables indépendantes augmente.

### c. Application sur le DATASET :

Voici les captures d'écran des résultats de l'application de l'algorithme KNN sur les instances du dataSet avec :

**Normalisation** : Z-Score ;

**Discrétisation** : en classes d'effectifs égaux ;

20 instances.

A Apriori	E Eclat	K KNN	N Naive Bayesien	R Résultat	T Temp d'exécution
-----------	---------	-------	------------------	------------	--------------------

Exécution de l'algorithme								
nbr instan...	20	premières instances	Manhattan	Lance				

A	P	C	LoK	WoK	AC	LoKG	Class reel	Class pred
0.17	0.25	0.0	0.36	0.17	-1.21	0.36	Kama	Kama
0.01	0.01	0.53	-0.2	0.23	-2.19	-0.2	Kama	Kama
-0.22	-0.42	1.8	-0.89	0.24	-0.82	-0.89	Kama	Kama
-0.4	-0.55	1.29	-0.81	0.37	-1.17	-0.81	Kama	Kama
0.52	0.38	1.71	0.08	0.94	-1.91	0.08	Kama	Kama
-0.19	-0.31	1.27	-0.64	0.17	-1.01	-0.64	Kama	Kama
-0.06	-0.06	0.47	-0.17	0.0	-0.09	-0.17	Kama	Kama
-0.3	-0.41	1.06	-0.55	0.13	-0.82	-0.55	Kama	Kama
0.71	0.8	0.2	1.12	0.64	-1.35	1.12	Kama	Rosa
0.64	0.61	0.9	0.68	0.76	-1.41	0.68	Kama	Kama
0.17	0.26	-0.07	0.23	-0.05	0.69	0.23	Kama	Kama
-0.33	-0.35	0.45	-0.5	-0.18	-1.62	-0.5	Kama	Kama
-0.38	-0.48	0.9	-0.5	-0.18	0.23	-0.5	Kama	Kama
-0.43	-0.44	0.26	-0.4	-0.32	-0.46	-0.4	Kama	Kama
-0.44	-0.45	0.18	-0.39	-0.45	-0.63	-0.39	Kama	Kama

Résultats de l'exécution de l'algorithme KNN.

## C.2 Matrice de confusion :

Une matrice de confusion est un tableau qui sert à décrire la performance d'un modèle de classification sur un ensemble de données d'essai dont les valeurs réelles sont connues. La matrice de confusion elle-même est relativement simple à comprendre, mais la terminologie qui s'y rapporte peut prêter à confusion.

Les matrices de confusion utilisent un format simple pour présenter les prévisions. Dans la matrice de confusion d'un modèle d'apprentissage automatique, les prévisions sont alignées sur la droite et les données réelles en haut. Les prévisions ou les résultats sont présentés dans les lignes, juste en dessous des données réelles. Les résultats peuvent comprendre l'indication correcte d'une valeur positive comme étant un vrai positif ou d'une valeur négative comme étant un vrai négatif, ou encore d'une valeur positive incorrecte comme un faux positif ou d'une valeur négative incorrecte comme un faux négatif.

## Application sur le DATASET :

Voici les captures d'écran de la matrice de confusion de l'algorithme KNN et Naïve Bayésienne (capturés au-dessus) avec :

**Normalisation** : Z-Score ;

**Discrétisation** : en classes d'effectifs égaux ;

20 instances.

Matrice de confusi...		
Matrice du confusion :		
[17, 2, 1]		
[0, 20, 0]		
[3, 0, 17]		
Kama :		
[17, 3]		
[3, 37]		
Rosa :		
[20, 0]		
[1, 39]		

**Résultats de la matrice de confusion de l'algorithme Naïve Bayésienne.**

Matrice de confusi...		Nbr K-voisins
		4 ▼
Matrice du confusion :		
[18, 1, 1]		
[1, 19, 0]		
[2, 0, 18]		
Kama :		
[18, 2]		
[3, 37]		
Rosa :		
[19, 1]		
[1, 39]		

**Résultats de la matrice de confusion de l'algorithme KNN.**

### C.3 Comparaison des deux algorithmes :

#### ACCURACY :

C'est le rapport entre les sujets correctement étiquetés et l'ensemble des sujets. Pour chaque classe on calcule l'Exactitude « ACCURACY » moyenne par la formule suivante :

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + TN + FP}$$

#### SENSITIVITY :

Sensitivity ou sensibilité moyenne c'est le rapport des vrais positifs au total des positifs (réels) dans les données.

Est calculé par la formule :

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

#### SPECIFICITY :

Specificity ou spécificité est le Rapport des vrais négatifs aux négatifs totaux dans les données.

Pour chaque classe on calcule SPECIFICITY moyenne par la formule :

$$\text{Specificity} = \frac{TN}{TN + FP}$$

#### PRECISION :

Rapport des prédictions correctes aux prédictions totales.

$$\text{precision} = \frac{TP}{TP + FP}$$

**RAPPEL :** pour chaque classe le RAPPEL moyenne est calculé par la formule suivante :

$$\text{recall} = \frac{TP}{TP + FN}$$

#### F-SCORE :

Pour chaque classe & F-SCORE moyenne :

Le score F1 tient compte à la fois de la précision et du rappel.

**C'est la moyenne harmonique de la précision et du rappel.**



Le score F1 est meilleur s'il existe une sorte d'équilibre entre la précision (p) et le rappel (r) dans le système.

À l'inverse, le score F1 n'est pas si élevé si une mesure est améliorée au détriment de l'autre.

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

### Application sur le DATASET :

Voici les captures de la comparaison entre l'algorithme KNN et Naïve Bayésienne (capturés au-dessus) avec :

**Normalisation** : Z-Score ;

**Discrétisation** : en classes d'effectifs égaux ;


20 instances.

A Apriori	E Eclat	K KNN	N Naive Bayésien	R Résultat	T Temp d'exécution
Classification supervisée					
			Naive Bayésien	KNN	
			Classe Kama ===== Accuracy : 0.9 Sensitivity : 0.85 Specificity : 0.925 Precision : 0.85 Rappel : 0.85 F-Score : 0.85	Classe Kama ===== Accuracy : 0.9166666666666666 Sensitivity : 0.9 Specificity : 0.9487179487179487 Precision : 0.8571428571428571 Rappel : 0.9 F-Score : 0.8780487804878048	
			Classe Rosa ===== Accuracy : 0.9833333333333333 Sensitivity : 1.0 Specificity : 1.0 Precision : 0.9523809523809523 Rappel : 1.0 F-Score : 0.975609756097561	Classe Rosa ===== Accuracy : 0.9666666666666667 Sensitivity : 0.95 Specificity : 0.975 Precision : 0.95 Rappel : 0.95 F-Score : 0.9500000000000001	
			Classe Canadian ===== Accuracy : 0.9333333333333333 Sensitivity : 0.85 Specificity : 0.9285714285714286 Precision : 0.9444444444444444	Classe Canadian ===== Accuracy : 0.95 Sensitivity : 0.9 Specificity : 0.9512195121951219 Precision : 0.9473684210526315	

Comparaison de l'algorithme KNN et Naïve Bayésienne.

▲ Apriori	E Eclat	K KNN	N Naive Bayesien	R Résultat	T Temp d'exécution
Classification supervisée					
Naive Bayesien			KNN		
Sensitivity : 1.0			Sensitivity : 0.95		
Specificity : 1.0			Specificity : 0.975		
Precision : 0.9523809523809523			Precision : 0.95		
Rappel : 1.0			Rappel : 0.95		
F-Score : 0.975609756097561			F-Score : 0.9500000000000001		
Classe Canadian =====			Classe Canadian =====		
Accuracy : 0.9333333333333333			Accuracy : 0.95		
Sensitivity : 0.85			Sensitivity : 0.9		
Specificity : 0.9285714285714286			Specificity : 0.9512195121951219		
Precision : 0.9444444444444444			Precision : 0.9473684210526315		
Rappel : 0.85			Rappel : 0.9		
F-Score : 0.8947368421052632			F-Score : 0.9230769230769231		
Résultats Naive Bayesien =====			Résultats KNN =====		
Accuracy Moyenne : 0.9388888888888888			Accuracy Moyenne : 0.9444444444444443		
Sensitivity Moyenne : 0.9			Sensitivity Moyenne : 0.9166666666666666		
Specificity Moyenne : 0.9511904761904763			Specificity Moyenne : 0.9583124869710234		
Precision Moyenne : 0.9156084656084656			Precision Moyenne : 0.9181704260651627		
Rappel Moyenne : 0.9			Rappel Moyenne : 0.9166666666666666		
F-Score Moyenne : 0.9067821994009414			F-Score Moyenne : 0.9170419011882428		

### Comparaison de l'algorithme KNN et Naïve Bayésienne.

Comparaison entre KNN et Naive Bayesien	 Comparer
<p>Comparaison entre Naive et KNN :</p> <p>L'algorithme KNN a prédit correctement plus de classe que l'algorithme Naive.</p> <p>L'algorithme KNN a une meilleure sensibilité que l'algorithme Naive.</p> <p>L'algorithme KNN a une meilleure spécificité que l'algorithme Naive.</p> <p>L'algorithme KNN a une meilleure précision que l'algorithme Naive.</p> <p>L'algorithme KNN a prédit correctement plus d'enregistrement positifs que l'algorithme Naive.</p> <p>L'algorithme KNN a une meilleur F-score que l'algorithme Naive.</p>	

### Comparaison de l'algorithme KNN et Naïve Bayésienne.

## D. Conclusion :

Ce projet nous a permis d'appliquer les connaissances acquises dans le module.

En premier, pour la phase de prétraitement on a appliqué la normalisation ainsi que la discrétisation. Ensuite on a appliqué les deux algorithmes d'extraction des motifs fréquents « Apriori et Eclat » et on a utilisé les résultats obtenus pour générer les règles d'associations et de corrélations. Et enfin, pour la classification supervisée des instances du notre dataset on a appliqué l'algorithme Naïve Bayésienne et KNN, après on a construit la matrice de confusion qui nous a permis de calculé les différents mesures pour comparer entre ces deux algorithmes de classification.