Université des Sciences et de la Technologie Houari Boumediene

Faculté d'Electronique et Informatique

Département Informatique

Techniques de Datamining

Réalisé par :

Manel Djehiche G01

Amira Benguega G01

M2 SII 2019/2020

Table des matières

- ✓ Clarans
  - Algorithme
  - Résultat

Conclusion

Introduction

le développement d'outils de stockage de données, les techniques de collecte de données et l'informatisation de la société ont permis la production d'une quantité énorme de données. Cependant, il est inutile de disposer de données que nous ne pouvons pas comprendre ni tirer de conclusions significatives. Nous devons donc les analyser.

Traditionnellement, les données étaient analysées à la main pour découvrir des connaissances intéressantes. Cependant, cela prend beaucoup de temps et est sujet à des erreurs, car cela peut manquer des informations importantes et il n'est tout simplement pas réaliste de le faire sur des bases de données volumineuses.

Pour résoudre ce problème, des techniques automatiques ont été conçues pour analyser les données et extraire des modèles, tendances ou autres informations utiles. C'est le but de Data Mining.

Pour effectuer le data mining, un processus en deux phases est généralement suivi (sept étapes). Ce processus est souvent appelé processus de «knowledge discovery from data»(KDD).

Première phase: prétraitement

- ✔ Nettoyage des données: les données sont nettoyées au moyen de processus tels que le remplissage des valeurs manquantes, le lissage des données bruitées ou la résolution des incohérences dans les données.
- ✔ Intégration des données: les données avec différentes représentations sont rassemblées et les conflits internes résolus.
- ✔ Transformation des données: les données sont normalisées, agrégées et généralisées.
- ✔ Réduction des données: cette étape vise à présenter une représentation réduite des données dans un entrepôt de données.
- ✔ Discrétisation des données: implique la réduction d'un nombre de valeurs d'un attribut continu en divisant la plage d'intervalles d'attribut.

Deuxième phase:

- ✔ Data mining: Cette étape consiste à appliquer des techniques de data mining (algorithmes) pour analyser les données et découvrir des modèles intéressants, ou pour extraire des connaissances intéressantes de ces données.
- ✔ Évaluer les connaissances découvertes: Cette étape consiste à évaluer les connaissances extraites des données. Cela peut être fait en termes de mesures objectives et / ou subjectives.
- ✔ Visualisation: Enfin, la dernière étape consiste à visualiser les connaissances extraites des données.

Les données du monde réel peuvent être bruyantes, incomplètes et mélangées de différentes sources. C'est là qu'intervient la phase exploratoire des données avant la phase de pré-traitement afin de comprendre nos données, de détecter les irrégularités et de découvrir des relations entre les attributs. Augmentant ainsi la qualité des résultats de l'étape de data mining.

Dans cette première phase du projet nous nous intéressons à l'étude exploratoire des données.

Objectif de projet (partie I)

Le but de cette première phase du projet est :

- ✔ Etudier le data set « Heart_stat.arff », ainsi que rédiger une description globale du sujet et détaillée des attributs.
- ✔ Développer une application en java qui permet de réaliser les fonctionnalités suivantes :
    - Lecture du benchmark, manipulation de son contenu, extraire les attributs, afficher la description et les valeurs.
    - Calculer les mesures de tendance centrale et en déduire les symétries.
    - Afficher les histogrammes pour chaque attribut.
    - Afficher la boîte à moustache pour chaque attribut et en déduire les données aberrantes.
    - Afficher les diagrammes de dispersions et en déduire les corrélations.

Description de dataset Heart_stat

L'objectif est de prédire sur la base de mesures de diagnostic si un patient a une maladie cardiaque. Plusieurs contraintes ont été placées sur la sélection de ses instances à partir d'une base de données plus grande.

Nous avons ouvrir la dataset « Heart_stat.arff » en utilisant les 3 classes de *weka* Instances,Instance et Attribute. Nous avons remarqué que notre dataset contient :

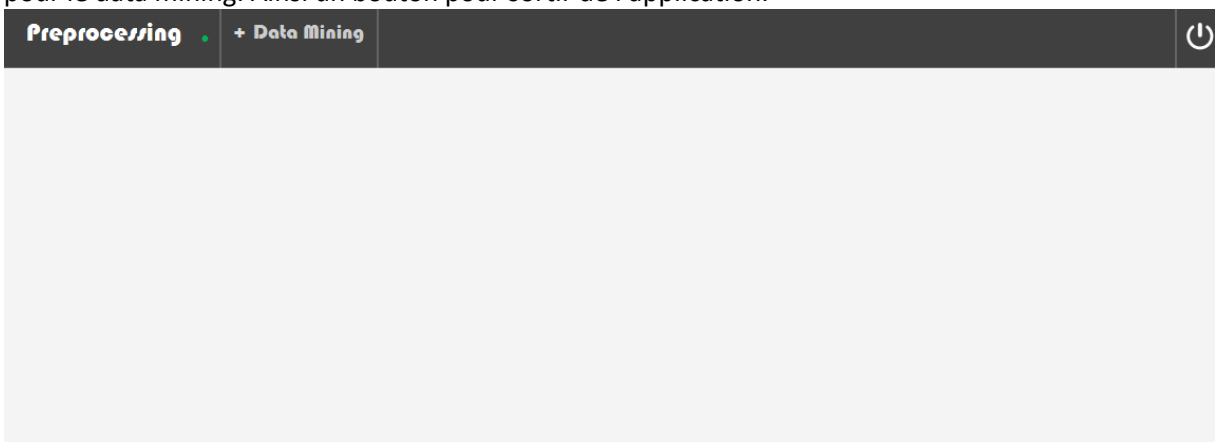
- ✔ 13 Attribut entre booléen, numérique ,nominale et ordinal.
- ✔ 269 instances

Attribut	***Type***
Âge	Numérique
Sexe	Booléen

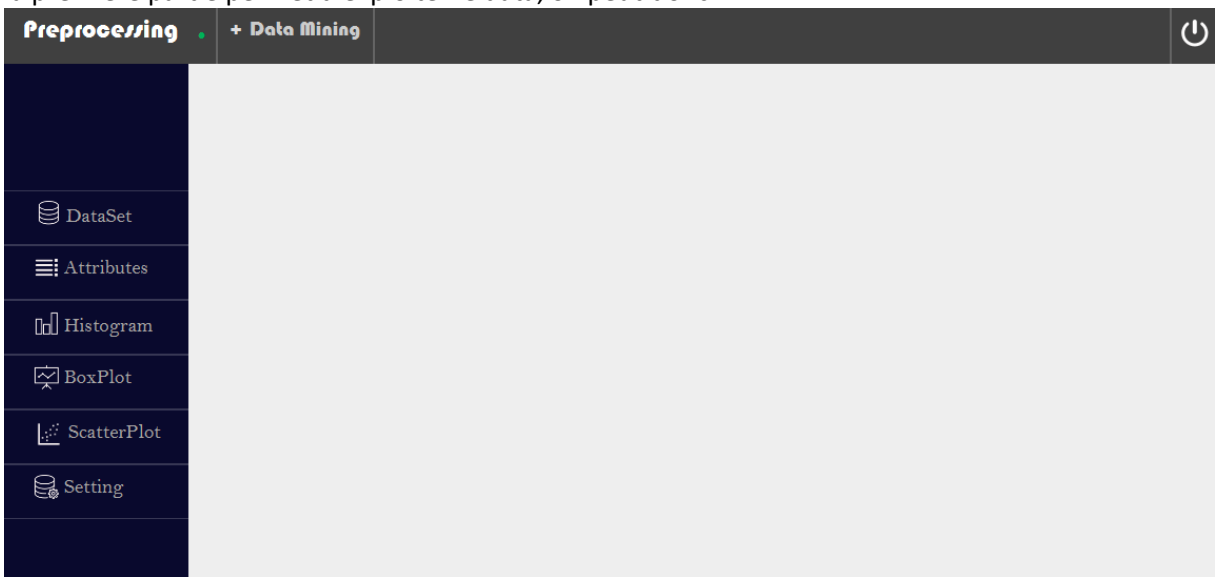
Poitrine	Ordinal
Vraie pression artérielle	Numérique
Sérum cholestérol	Numérique
Glycémie	Booléen
Résultats électro cardiographiques	Ordinal
Fréquence cardiaque maximale	Numérique
Angine induite par l'exercice	Booléen
Oldpeak	Numérique
Pente	Ordinal
Nombre de grands navires	Numérique
thal	Ordinal
Class	Nominal

Présentation de l'interface de l'application

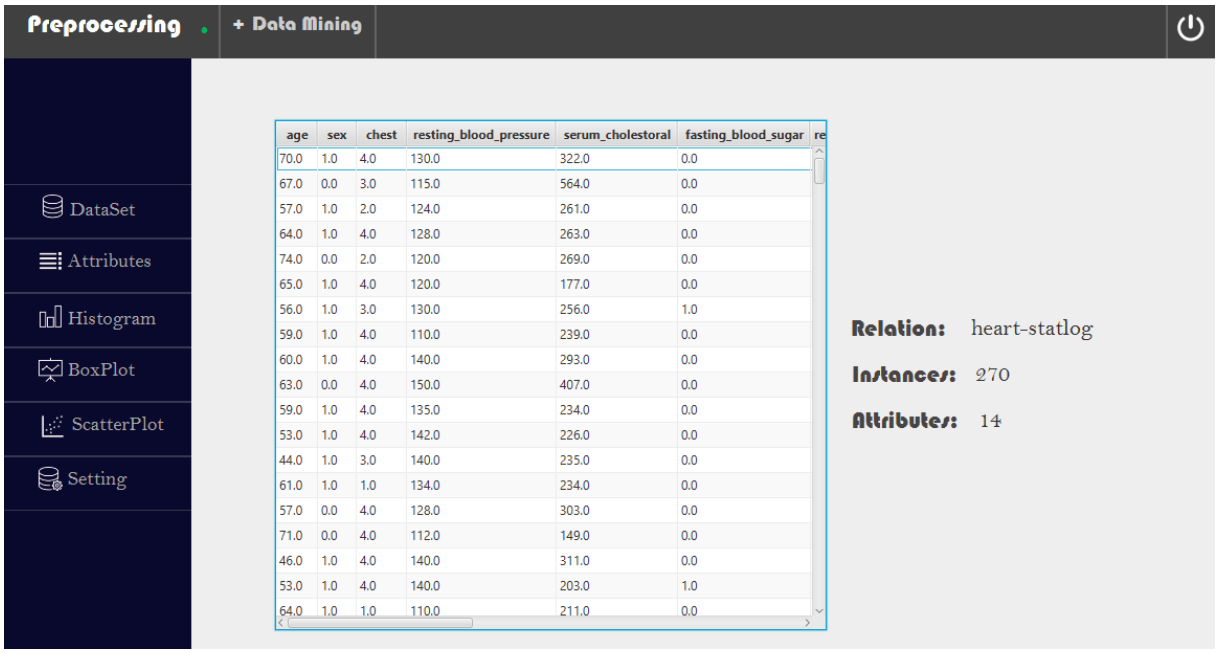
l'application est constituée de deux zones, une pour la première partie (preprocessing) la deuxième pour le data mining. Ainsi un bouton pour sortir de l'application.



la première partie permet d'exploiter le data, on peut donc:

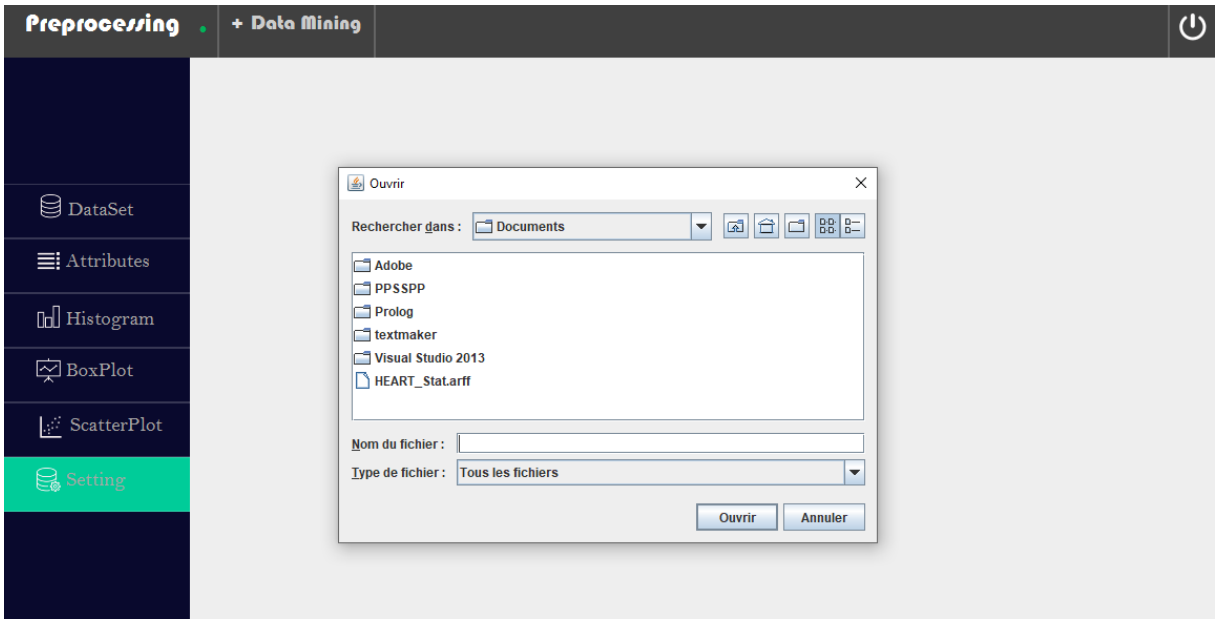


✔ afficher le dataset et quelque informations supplémentaire (bouton: dataset)



age	sex	chest	resting_blood_pressure	serum_cholestoral	fasting_blood_sugar	re
70.0	1.0	4.0	130.0	322.0	0.0	
67.0	0.0	3.0	115.0	564.0	0.0	
57.0	1.0	2.0	124.0	261.0	0.0	
64.0	1.0	4.0	128.0	263.0	0.0	
74.0	0.0	2.0	120.0	269.0	0.0	
65.0	1.0	4.0	120.0	177.0	0.0	
56.0	1.0	3.0	130.0	256.0	1.0	
59.0	1.0	4.0	110.0	239.0	0.0	
60.0	1.0	4.0	140.0	293.0	0.0	
63.0	0.0	4.0	150.0	407.0	0.0	
59.0	1.0	4.0	135.0	234.0	0.0	
53.0	1.0	4.0	142.0	226.0	0.0	
44.0	1.0	3.0	140.0	235.0	0.0	
61.0	1.0	1.0	134.0	234.0	0.0	
57.0	0.0	4.0	128.0	303.0	0.0	
71.0	0.0	4.0	112.0	149.0	0.0	
46.0	1.0	4.0	140.0	311.0	0.0	
53.0	1.0	4.0	140.0	203.0	1.0	
64.0	1.0	1.0	110.0	211.0	0.0	

✔ décrire les attributs de dataset (bouton: attribute)
✔ afficher l'histogramme (bouton: histogram)
✔ afficher le boxplot (bouton: boxPlot)
✔ afficher le scatterplot (bouton: scatterPlot)
✔ modifier le dataset (bouton: setting)



Étude exploratoire des caractéristiques descriptives des données

Calculer les paramètres de tendance centrale

Avant de calculer la médiane, la moyenne, les quartiles Q1 et Q3, IQR et le mode de chaque attribut dans notre dataset « Heart_stat » nous allons les définir :

- ✔ **Max et Min**

Max = La plus grande valeur que peut prendre un attribut.
Min = La plus petite valeur que peut prendre un attribut.

- ✔ **Moyenne**

Cas de valeurs continues : La moyenne est calculée en divisant la somme de toutes les valeurs d'un attribut par le nombre total de valeurs que peut prendre ce dernier.

$$\bar{x} = \frac{\sum_{i=1}^{N} x_i}{N} = \frac{x_1 + x_2 + \cdots + x_N}{N}$$

Cas de données groupées en compartiments : On calcule le MidRange de chaque intervalle ($si\ Intervalle = [L1,2]\ alors\ MidRange = (L1 + L2)\ /\ 2$). Puis on divise la somme des MidRanges de tous les intervalles par le nombre d'intervalles.

- ✔ **Médiane**

La médiane tend à partager un ensemble de valeurs rangées par ordre croissant en deux sous-ensembles de taille égale.

Cas de variables continues : Soit N la taille de l'ensemble des valeurs et M la médiane : $Si\ N = 2p\ alors\ M\ (p+1)\ valeur\ Si\ N = 2p+1\ alors\ M = la\ moyenne\ entre\ la\ p_e\ et\ la\ (p+1)\ valeur$

Cas de données regroupées en compartiments :

- Calculer la fréquence de chaque groupe.
- Calculer la médiane comme suit :

$$M = L1 + (N2 - \Sigma(freq)1 freqmedian) width$$

$L1 = la\ valeur\ min\ de\ l'intervalle\ L1{-}L2\ N = le\ nombre\ total\ de\ valeurs\ (Total\ fréquences)\ \Sigma(freq)1\ fréquences\ avant\ le\ groupe\ médian\ freqmedian\ fréquence\ du\ groupe\ médian\ Width = L2{-}L1$

Pour trouver le groupe médian il suffit de calculer les fréquences cumulées, trouver la valeur médiane des fréquences, puis le groupe auquel elle appartient. Ce groupe est appelé groupe ou classe médiane.

- ✔ **Mode**

Le mode représente la valeur la plus fréquente quel que soit le type de cette valeur (Numérique, nominale ou intervalle)

- ✔ **Quartiles (Q1 et Q3)**

Les quartiles sont les valeurs qui divisent un ensemble de valeurs en 4 sous-ensembles de même taille.
Pour calculer les quartiles, il faut d'abord que les valeurs de l'attribut soient rangées par ordre croissant.
Le 1er quartile Q1 est la donnée de la série qui sépare les 25 % inférieurs des données.

Le 3^{e} quartile Q3 est la donnée de la série qui sépare les 75 % inférieurs des données.

- ✔ **IQR**

IQR= Q3-Q1

Etude de Symétrie

Une distribution est dite symétrique si la moyenne, la médiane et le mode sont presque égaux ou égaux, elle est dite asymétrique sinon.

Elle est dite Asymétrique à droite (negatively skewed) si : Moyenne < Médiane<Mode

Elle est dite Asymétrique à gauche(positively skewed) si: Moyenne < Médiane<Mode

- ✔ tension artérielle au repos: Asymétrique à gauche.

Name: resting_blood_pressure
Type: Numeric

Statistic	Value
Minimum	94.0
Maximum	200.0
Mean	131.34444444444443
Median	130.0
Mode	120.0
variance	319.03705080545234
StdDiv	17.861608292800856

- ✔ cholestérol sérique: Asymétrique à gauche.

Mean	249.65925925925927
Median	245.0
Mode	204.0

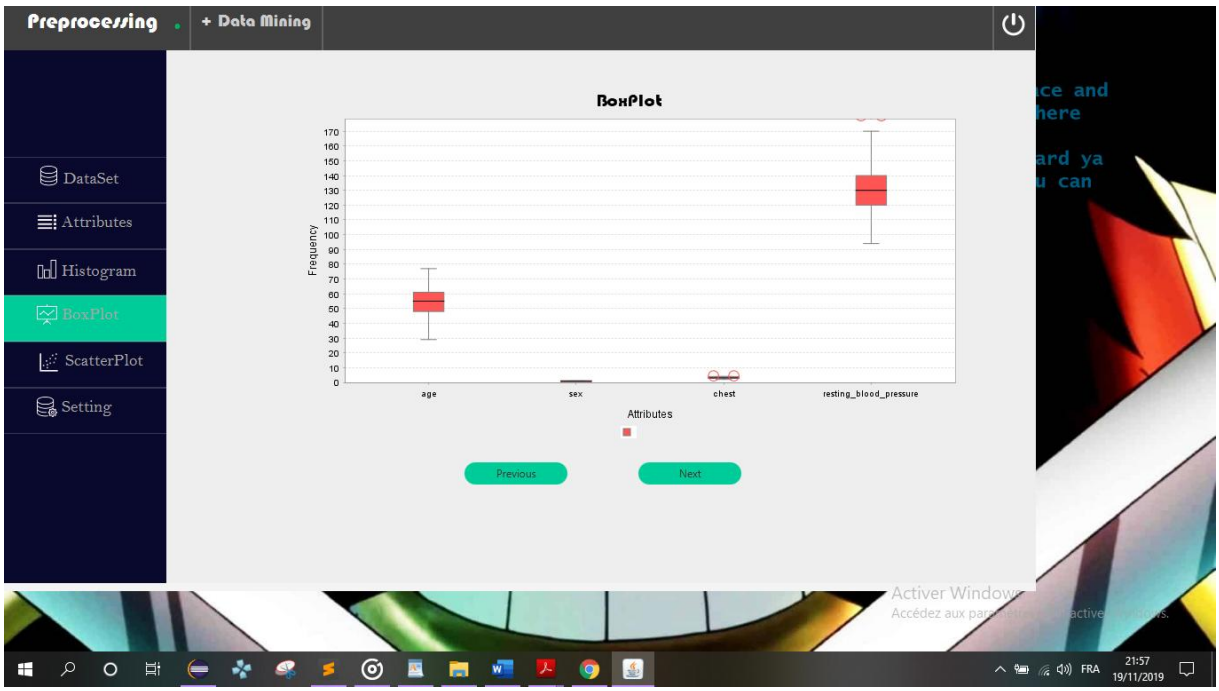
✔ fréquence cardiaque maximale atteinte: Asymétrique.

Mean	149.67777777777778
Median	153.5
Mode	142.0

les Boite à Moustache

Un box-plot est un graphique simple composé d'un rectangle duquel deux droites sortent afin de représenter certains éléments des données. Il est intéressant d'utiliser les box-plot lorsqu'on désire visualiser des concepts tels que la symétrie, la dispersion ou la centralité de la distribution des valeurs associées à une variable. Ils sont aussi très intéressants pour comparer des variables basées sur des échelles similaires et pour comparer les valeurs des observations de groupes d'individus sur la même variable.

Exemple: boîte à moustache pour les quatre premiers attributs.



D'après le boxplot, les attributs qui ne contiennent pas des valeurs aberrantes sont :

age, sex, serum cholestoral, fasting blood sugar, resting electrocardiographic results, exercise induced angina, the slope of the peak exercise ST segment, thal

les attributs qui contiennent des valeurs aberrantes sont :

chest pain type, resting blood pressure, maximum heart rate achieved, oldpeak, number of major vessels

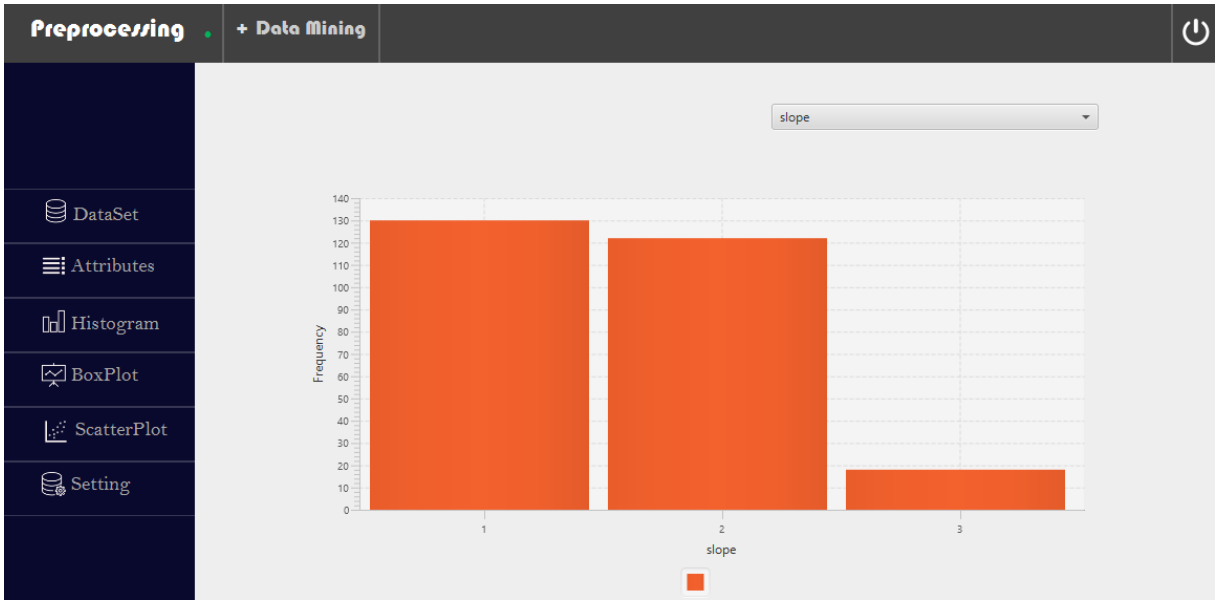
Histogramme

l'histogramme est une méthode graphique permettant de résumer la distribution d'un attribut donné.

cas 1: attribut nominal

une barre est dessinée pour chaque valeur de l'attribut. La hauteur de la barre est la fréquence de la valeur (le nombre d'instances qui ont cette valeur).

Example: Attribute = slope (prend trois valuer 1, 2, 3)



cas 2: attribut numérique

La plage de valeurs de l'attribut est partitionnée en sous-intervalles consécutives disjointes de largeur généralement égale. Pour chaque sous-intervalles, une barre est dessinée avec une hauteur qui représente le nombre total d'éléments observés dans le sous-intervalles.Example: Attribute =

Age

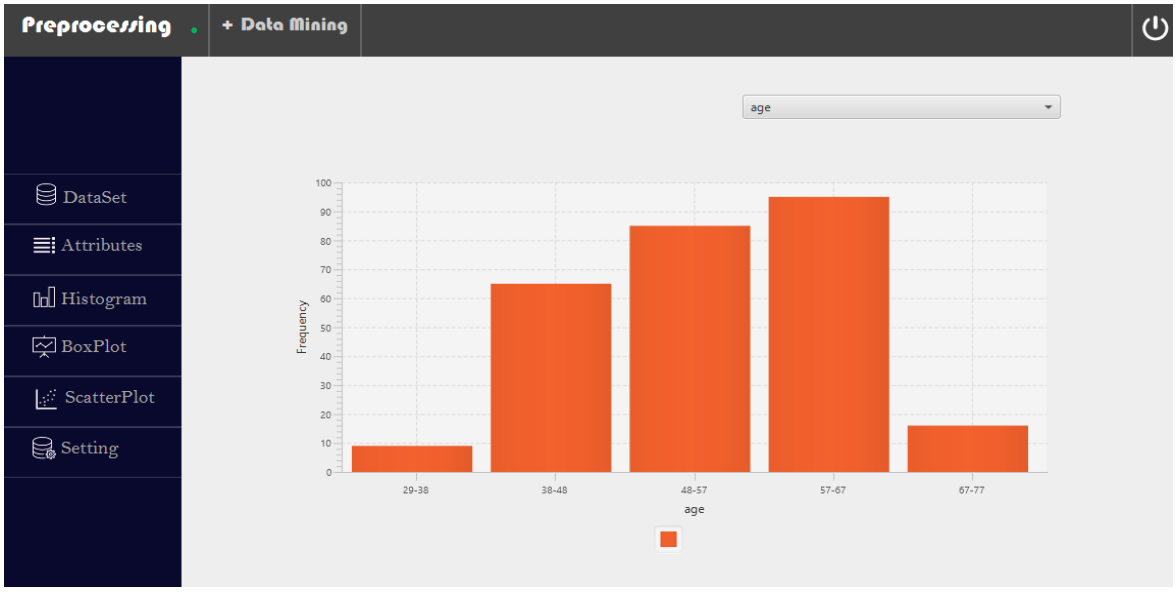


Diagramme de Dispersion

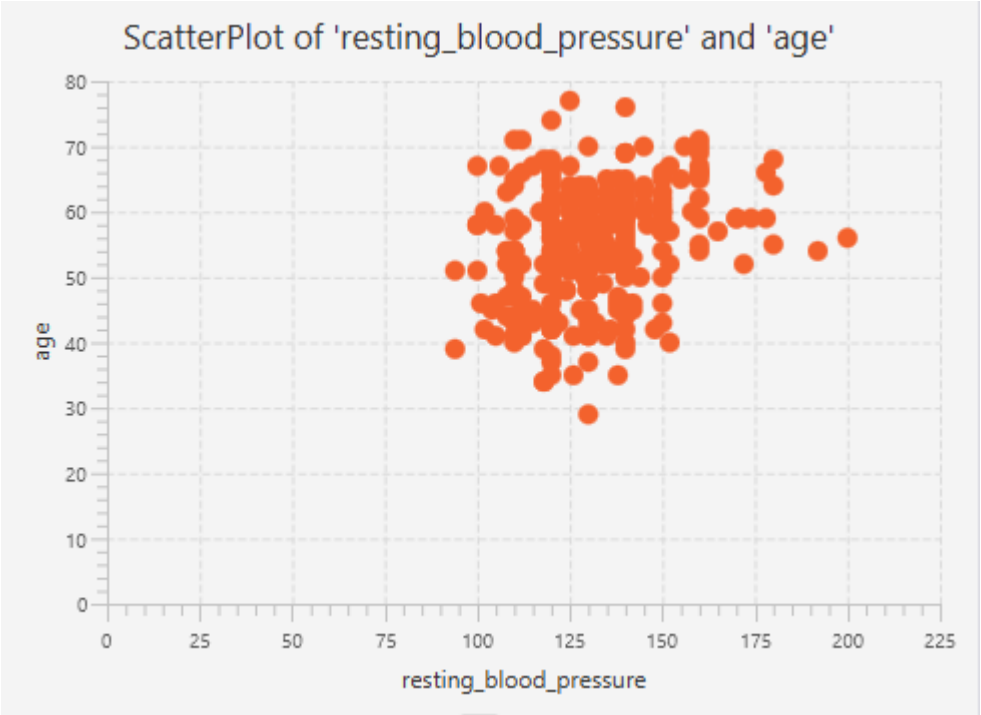
Le diagramme de dispersion ou de corrélation (ou scatter diagram en anglais) est un outil de contrôle et d'aide à la décision pour vérifier l'existence de corrélation ou d'une relation entre variables de nature quantitative.

Si les points tracés du motif sont inclinés du bas à gauche au haut à droite =>une corrélation positive.

Si le motif de tracé pointe de haut en bas à droite => une corrélation négative.

sinon => pas de corrélation.

Example: Attribute1 = resting blood pressure , Attribute2 = age.



Partie II : Les outils Datamining

Dans cette phase nous allons implémenter et appliquer quatre algorithmes de datamining sur le dataset HEART_STAT après le prétraitement, deux algorithme d'associations (Apriori et Eclat) et deux algorithme de partitionnement (K-MEDOIDS et CLARANS).A la fin nous allons faire une étude comparatif entre les ces quatre algorithme.

1. Objectif

Nous amène à nous fixer les objectifs suivants :

- ✓ Implémentation de A-PRIORI ECLAT
- ✓ Appliquer A-PRIORI et ECLAT
- ✓ Implémentation de K-MEDOIDS et CLARNS
- ✓ Appliquer K-MEDOIDS et CLARNS
- ✓ Comparaison entre les 3 algorithmes.
- ✓ Application des algorithmes sur HEART_Stat

2. Algorithme Datamining

2.1 Algorithme d'association

Structure de données

Dataset HEART_STAT

Avant de d'appliquer les algorithmes de datamining, nous devons appliquer le Binning sur les quatre attributs numériques de notre dataset. Il s'agit d'une discrétisation à fin de transformer les valeurs numériques en classe pour réduire la complexité d'exécution des algorithmes après. Nous avons choisir que les intervalles résultants vont avoir la même taille.

L'ensemble des items fréquent

Nous avons choisi d'utiliser la structure de dictionnaire sous java tell que un sous ensemble d'élément est représenter par {clé : valeur} pour diminuer la complexité lors de la génération des sous ensembles Ck, pour accéder à une fréquence d'un élément il suffit de faire dic['element'] avec une complexité o(1) à la place de parcourir tout l'ensemble o(n)

2.1.1 Apriori

Algorithme

Entré

- ✓ D : Base de donné de transaction
- ✓ Support_min : le support minimum
- ✓ Confidence_min : la confidence minimale

Sortie

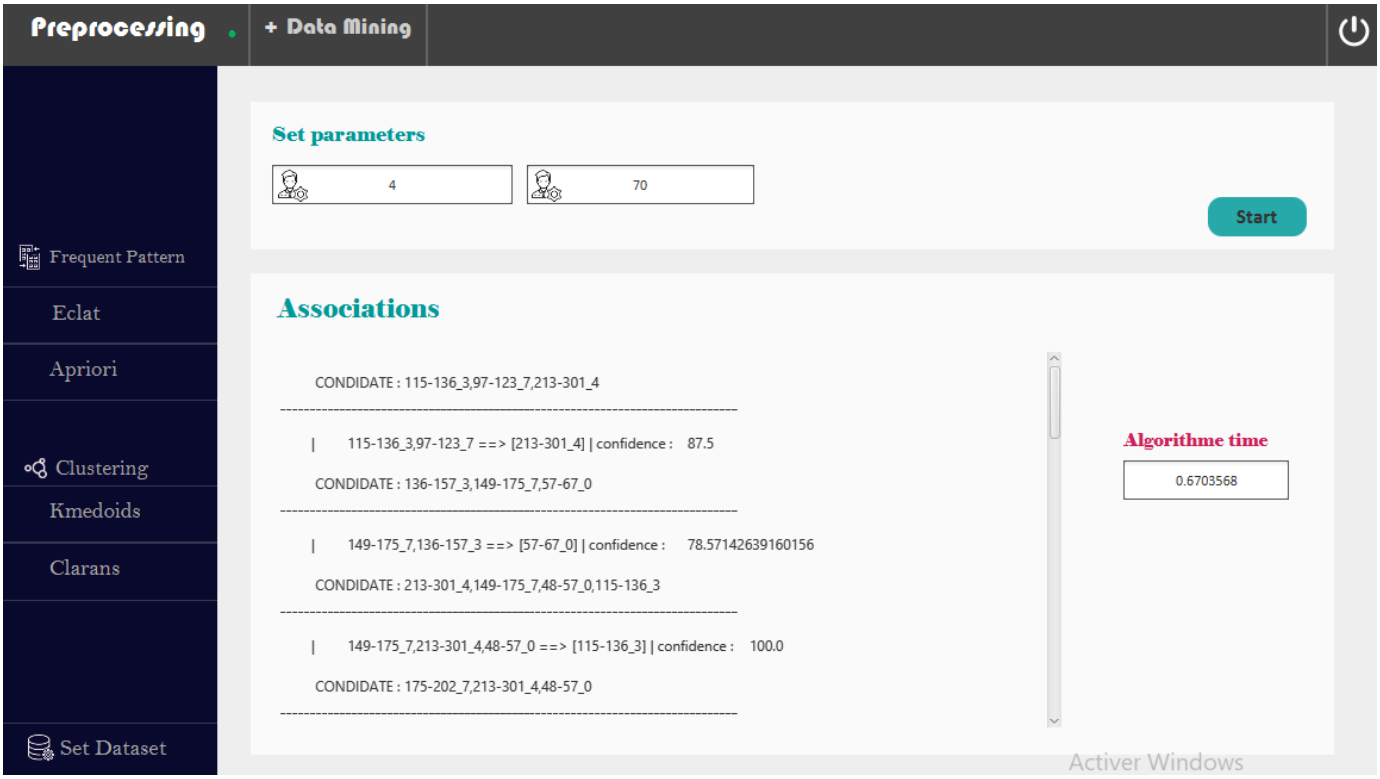
- ✓ L : ensemble des éléments fréquents

Début

1. Générer c1 le sous ensemble des éléments fréquents de taille k tel que k=1 à partir de D tel que la clé et l'élément et la valeur représente la fréquence de cette élément.
2. Générer L1 qui contient les éléments de c1 tell que la fréquence est supérieur à supp_min
3. Ajouter L1 à la structure L
4. Générer Ck le sous ensemble des éléments fréquents de taille k+1 à partir de Lk-1 et retourner à la base donnée D pour chaque nouveau élément afin de calculer ca fréquence.
5. Générer Lk qui contient les éléments de Ck dont leurs fréquences dépasse la support_min
6. Ajouter Lk à la structure L
7. Si aucun sous ensemble nouveau a été générer alors retourner L, Sinon aller à 4
8. Extraire les règles d'associations pour tous les candidats de L. Nous générons tous la combinaison possible de la même manière de génération des ensembles fréquents.
9. Calculer le confidence de chaque règle et filtrer a la fin les règles dont leurs confidence est superieur a confidence_min.

FIN

Résultat



2.1.2 Eclat

Algorithme

Entré

- ✓ D : Base de donné de transaction
- ✓ Support_min : le support minimum
- ✓ Confidence_min: la confidence minimale

Sortie

- ✓ L : ensemble des éléments fréquents

Début

1. Générer c1 le sous ensemble des éléments fréquents de taille k tel que k=1 à partir de D tel que la clé et l'élément et la valeur représente la liste de tout les transactions ou cette élément appartient
2. Générer L1 qui contient les éléments de c1 tell que la fréquence est supérieur à supp_min
3. Ajouter L1 à la structure L

4. Générer Ck le sous ensemble des éléments fréquents de taille k+1 à partir de Lk-1 et calculer la fréquence de chaque nouveaux élément par l'intersection entre les liste de transactions de ces éléments composants sans revenir à la base de transaction D.

5. Générer Lk qui contient les éléments de Ck dont leurs fréquences dépasse la support_min

6. Ajouter Lk à la structure L

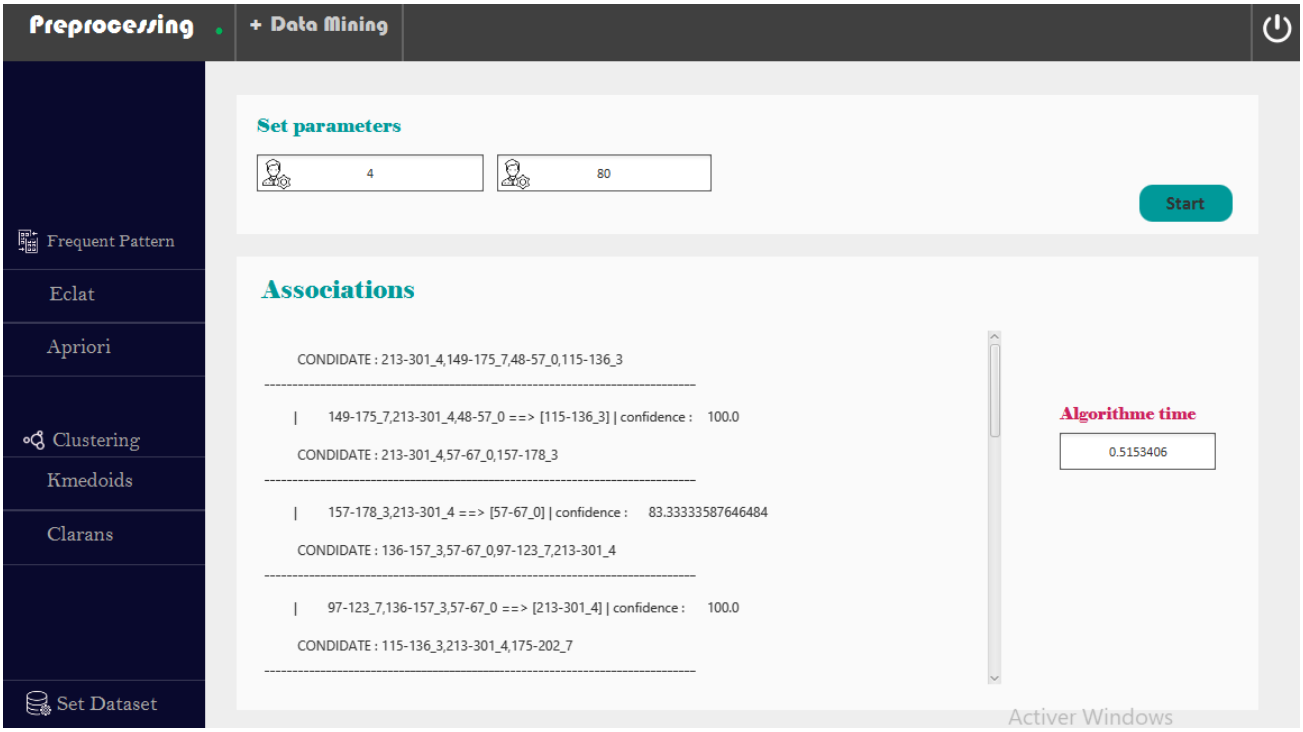
7. Si aucun sous ensemble nouveau a été générer alors retourner L, Sinon aller à 4

8. Extraire les règles d'associations pour tous les candidats de L. Nous générons tous la combinaison possible de la même manière de génération des ensembles fréquents.

9. Calculer le confidence de chaque règle et filtrer a la fin les règles dont leurs confidence est superieur à confidence_min.

FIN

Résultat



2.2 Algorithme de partitionnement

a. K-medoids

Algorithme

Entré

- ✓ D : Dataset
- ✓ K : Nombre de clusters

Sortie

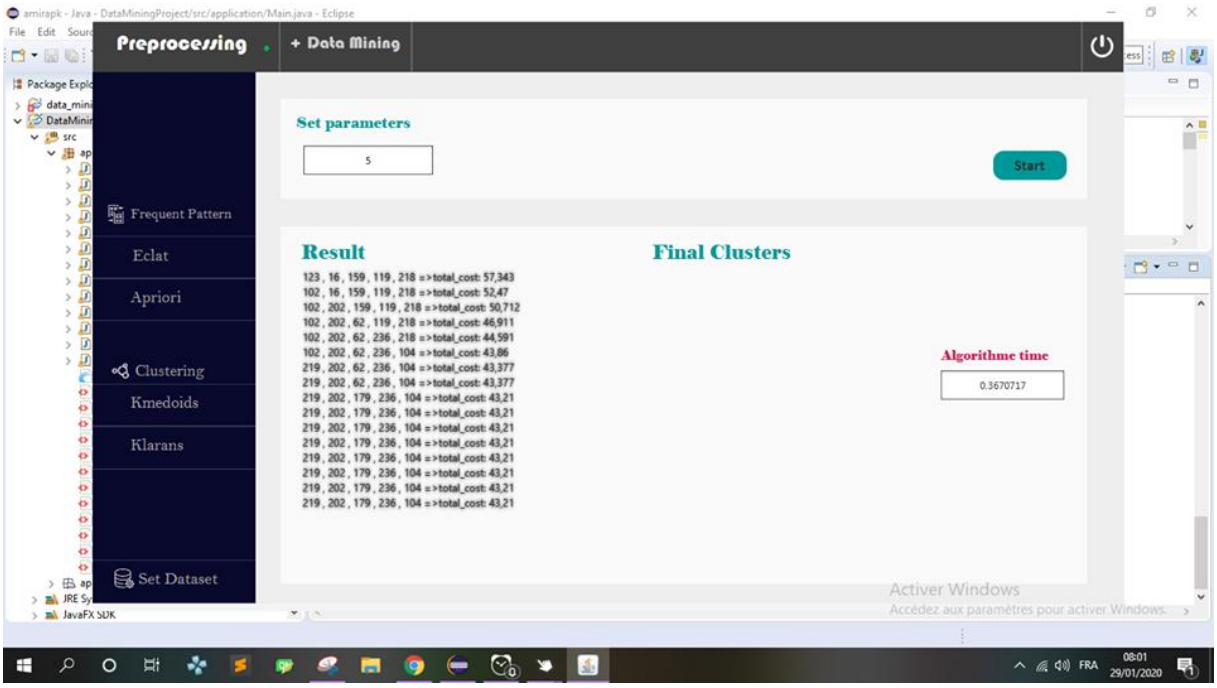
- ✓ Les K cluster

Début

1. Choisir k instance aléatoire comme centroide
2.
    - Calculer la distance entre chaque instance non centroide et les k centroide
    - Mettre l'instance dans cluster dont la distance du centroide est minimal
3. Calculer la moyenne de chaque cluster qui représente les nouveaux centroide
4. Répéter les étapes 2 et 3 jusqu'à ce que les clusters reste inchangé

FIN

Résultat



b. Clarans

Algorithme

Entré

- ✓ D : Dataset
- ✓ K : Nombre de clusters

Sortie

- ✓ Les K cluster

Début

1. Paramètres d'entrée numlocal et maxneighbors. Initialiser i à 1 et mincost à un grand nombre.
2. Définissez le courant sur un nœud arbitraire dans Gn; k.
3. Réglez j sur 1.
4. Considérons un voisin aléatoire S de courant, et basé sur sur 5, calculez le différentiel de coût des deux nœuds.
5. Si S a un coût inférieur, réglez le courant sur S et accédez à Étape 3.
6. Sinon, incrémentez j de 1. Si j maxneighbors, allez à l'étape 4.

7. Sinon, lorsque j> maxneighbors, comparer le coût de courant avec mincost. Si le premier est inférieur à mincost, définissez mincost sur le coût du courant et définissez bestnode au courant.

8. Incrémenter i de 1. Si i> numlocal, sortir bestnode et arrêtez. Sinon, passez à l'étape

FIN

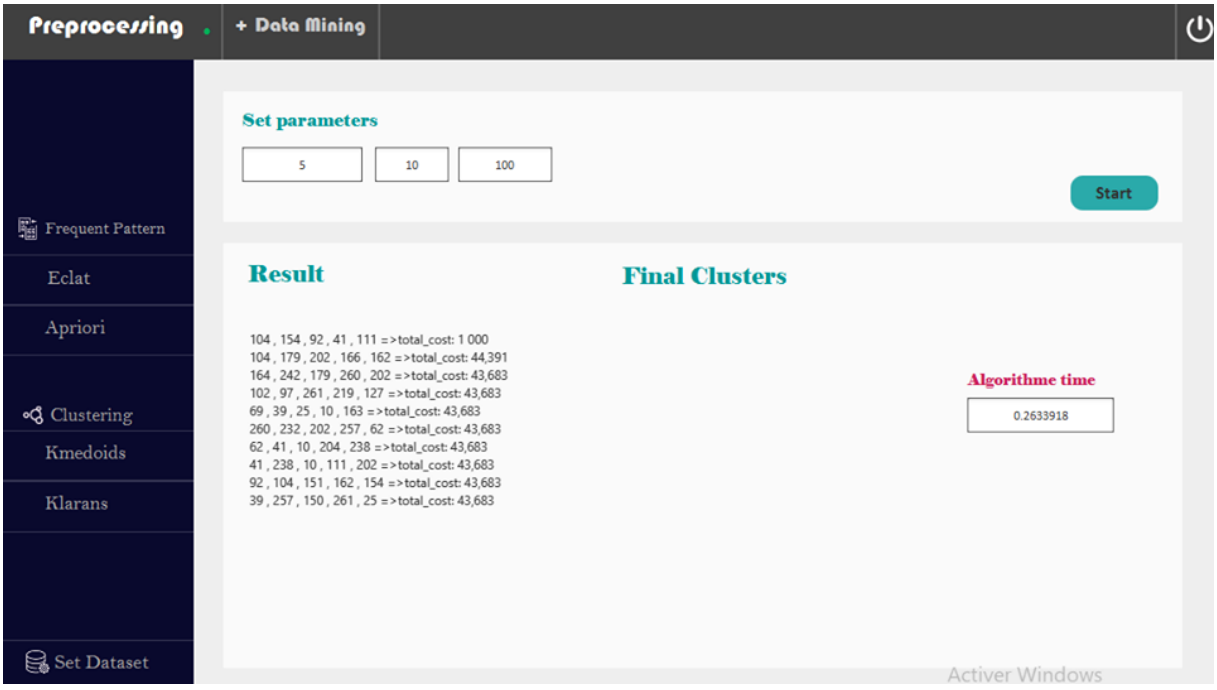
Les étapes 3 à 6 ci-dessus recherchent des nœuds avec progressivement plus bas les coûts. Mais, si le nœud actuel a déjà été comparé avec le nombre maximum de voisins du nœud (spécifié par maxneighbors) et est toujours du coût le plus bas, le nœud actuel est déclaré comme un minimum «local». Ensuite, dans Étape 7, le coût de ce minimum local est comparé au coût le plus bas obtenu jusqu'à présent. Le moindre des deux coûts ci-dessus est stocké dans mincost. L'algorithme CLARANS se répète ensuite rechercher d'autres minima locaux, jusqu'à ce que nombre d'entre eux été trouvé.

Comme indiqué ci-dessus, CLARANS a deux paramètres:

- ✓ Nombre maximum de voisins examinés (maxneighbors)
- ✓ Nombre de minima locaux obtenus (numlocal).

Plus la valeur de maxneighbors est élevée, plus CLARANS est proche de PAM, et plus la recherche de minima locaux est longue. Mais, la qualité d'un tel minimum local est plus élevée et moins locale des minima doivent être obtenus.

Résultat



3. Etude Comparative entre les Algorithmes

a. Comparaison entre Eclat et Apriori

- ✓ A-priori est un algorithme puissant, il permet parfaitement d'extraire et d'identifier les éléments en relation dans un DataSet et qui apparaissent fréquemment ensemble ce qui nous permet d'établir des règles à partir des données de notre DataSet. Mais son inconvénient principale c'est qu'il parcourt tout le Dataset pour chaque nouveau élément a fin de calculer sa fréquence.

- ✓ Eclat contrairement a Apriori il parcourt le Dataset une seul fois pour générer le sous ensemble des éléments de taille 1.Ensuite pour trouver la liste des transactions pour les nouveaux éléments, il fait l'intersection des ensembles des transactions des composantes de cet élément.

- ✓ Pour Apriori et Eclat, le nombre de règles extraites et leur niveau de fiabilité dépend du niveau de confiance et du support fixé par l'utilisateur. Il est donc préférable de fixer un niveau de confiance élevé et un support élevé dans le cas des domaines sensibles (médecine par exemple) pour que les règles soient fiables.
- ✓ En ne se basant que sur les fréquences d'apparition, il est possible de sortir des règles qui n'ont pas une grande fiabilité sémantique car parfois il existe seulement des coïncidences (deux éléments apparaissent plusieurs fois ensemble sans pour autant être en relation).

b. Comparaison entre K-medoids et clarans

K-medoidss palie au problème de précision du K-means. Les coûts sont pris en compte dans le changement des centres, ce qui fait qu'on ne change de centre que pour un meilleur centre. Donc, on obtient un meilleur clustering. Aussi, les centres sont des membres du DataSet donc notre base de comparaison est un élément du DataSet, ce qui donne un clustering plus fiable.

Clarans combine entre les avantages des deux algorithmes de classification précédents (K-means et K-medoidss) le premier en rapidité et le deuxième en qualité du clustering.

4. Conclusion

Nous avons, lors de ce TP pu atteindre nos objectifs fixés au départ :

Pour la partie de Prétraitement

- ✓ Extraire et afficher du contenu et la description de Dataset « Heart_Statlog ».
- ✓ Calculer les mesures de tendance centrale et déduire les symétries ou asymétries des attributs
- ✓ Afficher de la boite à moustache pour chaque attribut et déduire les données aberrantes.
- ✓ Afficher des histogrammes pour chaque attribut et en déduire les valeurs les plus fréquentes.
- ✓ Afficher les diagrammes de dispersions et en déduire les corrélations.

La première partie nous a permis de manipuler et de comprendre les caractéristiques des attributs de Dataset. On a pu voir aussi que l'exploration est une phase très importante, parce que les données sont parfois sales, inconsistants et incomplètes.Dans le cas de Dataset « Heart_statlog » le traitement était simple car nous avons 13 attributs seulement et pas de valeurs manquantes dans les attributs ce qui a simplifier le processus.

Pour la partie d'applications des outils de Datamining

- ✓ Implémentation de A-PRIORI ECLAT
- ✓ Appliquer A-PRIORI et ECLAT
- ✓ Extraire les règles d'associations les plus fortes.
- ✓ Implémentation de K-MEDOIDS et CLARNS
- ✓ Appliquer K-MEDOIDS et CLARNS
- ✓ Extraire les clusters les plus forts.
- ✓ Application des algorithmes sur HEART_Stat
- ✓ Comparaison entre les 4 algorithmes.
- ✓ Afficher les résultats sur l'interface