UFR des Sciences et Techniques
Master 1 GIL Université de Rouen

Rapport du Projet de Fouilles de données

A la demande de Mme Soualmia

AHOUNOU Folabi Thierry & AYADI Hanane

2012-2013

Introduction

Dans le cadre du module de fouilles de données nous avions à une application interfacée, en Java, implantant l'algorithme Close. Le but de cet algorithme est d'extraire les règles exactes et approximatives liées à un ensemble d'items.

Pour implanter l'algorithme on a découpé notre code en quatre grandes parties :

- Préparation des données
- Appel à l'algorithme Close
- La génération des règles exactes
- La génération des règles approximatives

Ces quatre grandes parties seront développées dans la suite de notre rapport.

Analyse (Diagramme de classe)

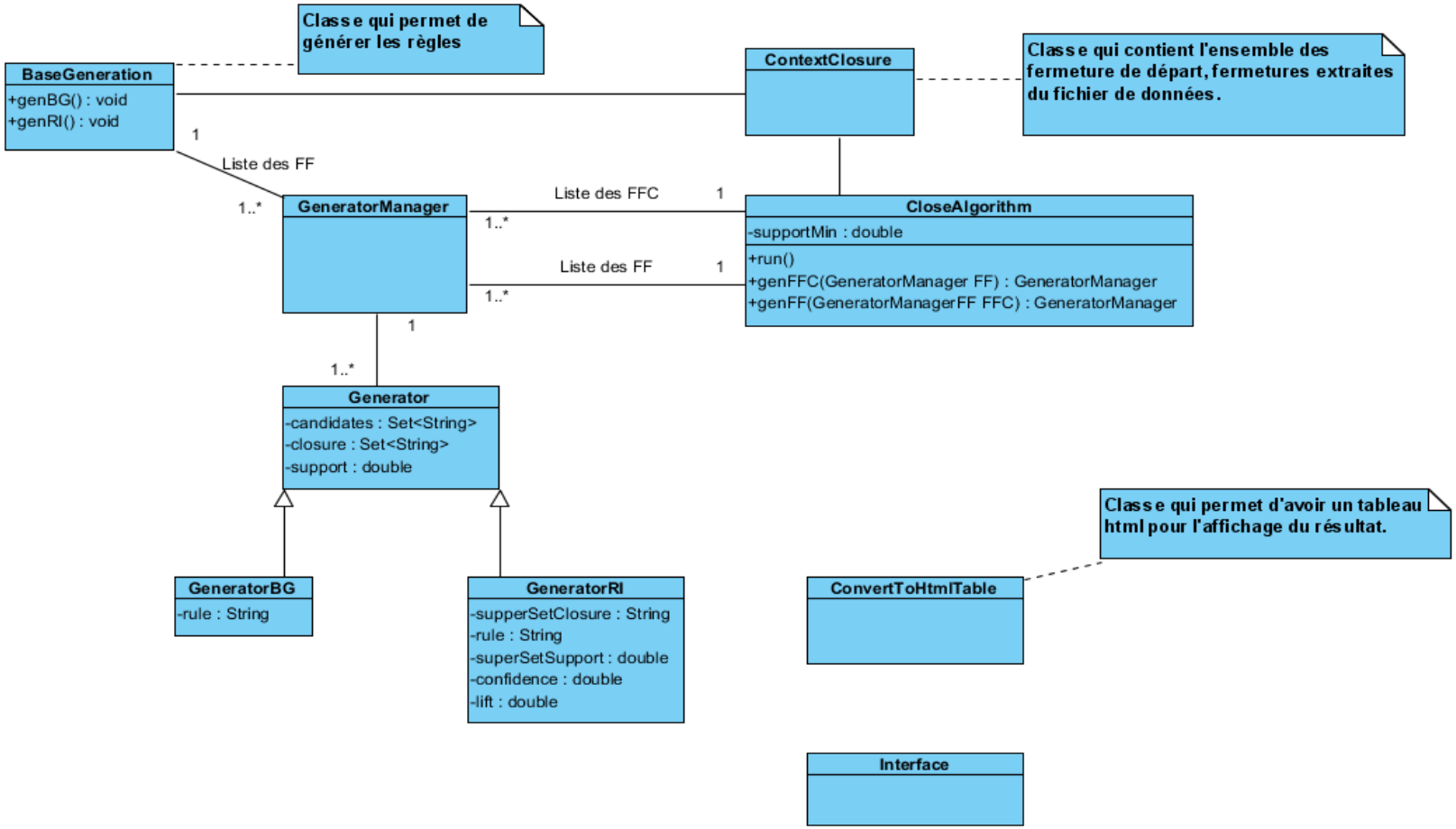

Figure 1 : Diagramme de classe de l'applicatoin

Préparation des données

Cette phase qui est la toute première phase du projet consiste à extraire les données nécessaires d'un fichier. Les données sont étalées sur plusieurs lignes. Chaque ligne du fichier contient un ensemble de générateurs séparés par le caractère **« | »**.

La méthode utilisée pour extraire les données est de lire chaque ligne du fichier et de casser cette dernière en plusieurs parties en utilisant comme délimiteur le caractère « | ». De là on récupère l'ensemble des générateurs et on les stocke dans une liste.

Algorithme Close

On retrouve dans cet algorithme trois fonctions principales :

- La fonction ***run*** qui permet de lancer l'algorithme.
- La fonction ***genFFC*** qui permet de générer le FFC suivant.
- Et la fonction ***genFF*** qui permet de générer le FF suivant.

L'implémentation de la fonction *run* commence par un premier calcul de $\boldsymbol{FFC_1}$ et $\boldsymbol{FF_1}$. Cette première étape est faite manuellement puisque FFC_1 est généré à partir du contenu de fichier de données. Il suffit donc après de supprimer de FFC_1 les générateurs dont la fermeture est inférieure à la fermeture minimale.

Avec une boucle on génère les $\boldsymbol{FFC_k}$ et $\boldsymbol{FF_k}$ suivants à l'aide des méthodes *genFFC* et *genFF*.

Cette boucle s'arrête lorsqu'il n'y plus de générateurs avec un support inférieur au support minimal.

La fonction genFFC

L'ensemble des (k+1)-générateurs candidats (utilisés durant l'itération suivante) est construit, en joignant les k-générateurs fréquents de l'ensemble FF_k comme suit :

1. Les (k+1)-générateurs candidats sont créés en joignant les k-générateurs de FF_k qui possèdent les mêmes k-1 premiers items.
2. Les (k+1)-générateurs candidats dont on sait qu'ils sont soit infréquents, soit non minimaux sont ensuite supprimés. Ces générateurs sont identifiés par l'absence d'un de leurs sous-ensembles de taille k parmi les k-générateurs fréquents de FF_k.
3. La troisième phase permet de supprimer parmi ces générateurs ceux dont la fermeture a déjà été calculée. Un tel générateur est identifié car il est inclus dans la fermeture d'un k-générateur fréquent de FF_k dont il est un sur-ensemble.

La fonction genFF

Une fois qu'on a calculé le FFC_k suivant le calcul de FF_k s'avère facile. En effet il faut supprimer de FFC_k les générateurs dont le support est inférieur au support minimal.

Génération des règles

Génération des règles exactes

La génération des règles exactes fut très simple. La seule chose qu'on a eu à faire est d'isoler de la fermeture du générateur courant les candidats du présent générateur. Cependant il est important de noter que quand le générateur a les mêmes candidats que sa fermeture, le générateur n'a donc pas de règles et on n'affiche pas non plus son support.

Génération des règles approximatives

Comparé à la génération des règles exactes, la génération des règles approximatives est plus complexe mais reste tout à fait abordable. En effet ici apparaissent les notions de sur-ensemble fermé, de confiance et de lift.

Pour obtenir le sur-ensemble fermé d'un générateur on récupère la fermeture du générateur et ensuite on stocke dans une liste l'ensemble des fermetures de la fermeture du générateur.

Le calcul de la confiance correspond à la division du support du sur-ensemble fermé sur le support du générateur courant. Quant au lift, sur cet exemple (a->bc), il correspond au support de (abc) sur le support de (a) multiplié par le support de (bc).

Il est aussi important de noter ici que si un générateur n'a pas de sur-ensemble fermé, on affiche que le générateur et sa fermeture.

Manuel Utilisateur

1. Charger un fichier de données avec le bouton ***« Choose »***
2. Renseigner un support avec les boutons ***« + »*** et ***« - »*** ou écrire directement dans la zone de texte. Seuls les boutons standards d'un pavé numérique sont autorisés.

   Note : Si un fichier n'est pas choisi, il est impossible de choisir un support et donc de lancer l'algorithme.

3. Pour finir il faut cliquer sur le bouton « Start » pour lancer l'algorithme.

A droite au centre, se trouve la trace d'exécution de l'algorithme et les règles.

En bas du bouton « Start » se trouve une liste des traces d'exécution de l'algorithme. Quand on clique sur un élément de la liste l'affichage est mise à jour dans la partie centrale de l'interface. Il est aussi possible d'enregistrer le résultat dans un fichier texte en cliquant sur la souris gauche puis sur l'élément ***« Save Buffer »***.