TP 1-2: Initiation à KNIME et règles d'associations

Cette séance de TP a pour but de se familiariser avec KNIME, une plateforme d'algorithmes de fouille et visualisation de données écrite en Java, ainsi que d'expérimenter les algorithmes de génération d'itemsets fréquents et de règles d'associations.

1 Présentation de Knime

KNIME (prononcer [naille-mh]) est un outil gratuit de création de workflows de fouille de données. Vous trouverez plusieurs tutoriels sur le site web `https://www.knime.org/`.

L'interface se décompose en 5 parties principales comme on le voit sur la figure ci-dessous.

- La partie 1 correspond au navigateur dans les workflows déjà créés.
- La partie 2 est un navigateur dans les "nœuds" de KNIME, à savoir les briques prédéfinies que l'on peut utiliser pour créer un workflow. Il suffit de prendre un nœud et de le faire glisser dans la partie 3.
- La partie 3 est l'éditeur de workflow, dans lequel on peut ajouter des nœuds et tirer des liens entre ces nœuds.
- La partie 4 décrit chacun des nœuds notamment les types d'entrée et de sortie du nœud et ses options de configuration.
- Enfin la partie 5 est la console de KNIME, qui informe des erreurs qui peuvent se révéler lors de l'exécution du workflow.

Dans la section suivante, nous voyons comment créer un workflow simple en utilisant le jeu de données `iris.csv` qui décrit des fleurs (longueur et largeur de pétale et sépale).

2 Création d'un premier workflow

Vous allez créer un premier workflow avec le jeu de données "iris" afin de découvrir les nœuds les plus communs pour la fouille et la visualisation de données.

Tâche 2.1 *Créer un nouveau workflow KNIME avec la command* `File / New...` *ou l'icône en haut à gauche. Nommer-le "iris".*

La méthode générale à suivre pour réaliser les tâches suivantes consiste à successivement :

1. sélectionner un nœud dans `Node repository`,
2. l'ajouter au workflow pas glisser-déposer (on peut créer plusieurs instances du même nœud),
3. le relier aux nœuds existants si besoin via les "ports" (entrées/sorties) des nœuds,
4. le configurer avec la commande `Configure...` du menu contextuel,

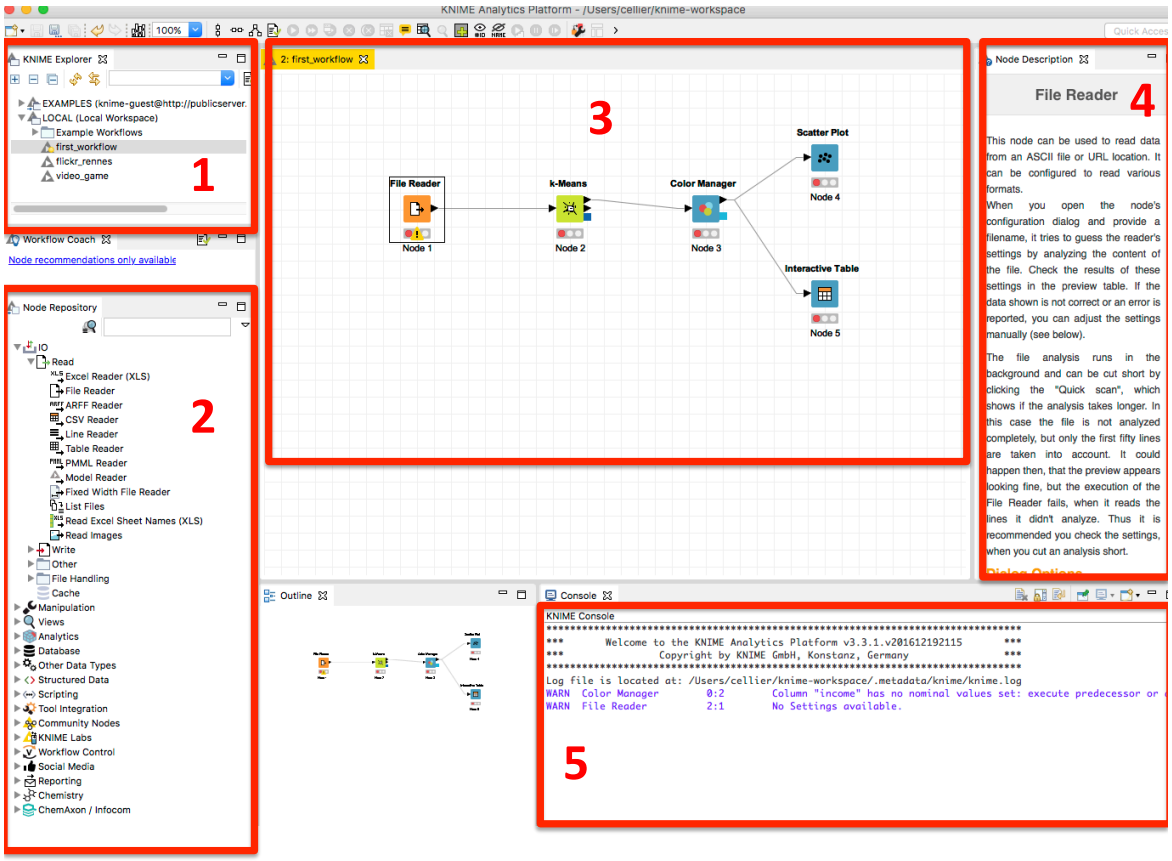

FIGURE 1 – Interface de KNIME

5. l'éxécuter avec la commande `Execute` du menu contextuel ou la touche F7,
6. et, si défini, visualiser le résultat avec les commandes dont le nom commence par `View:` ou placées en bas du menu contextuel.

Le fait de configurer, éxécuter et visualiser les nœuds au fur et à mesure permet de vérifier que le workflow est conforme à ce qu'on attend et de détecter les erreurs rapidement.

Il existe deux moyens de chercher un nœud : (a) en explorant l'arborescence de nœuds qui les classe par catégorie, (b) en entrant des mots-clés. Les deux moyens peuvent être combinés. En sélectionnant un nœud on peut voir sa description dans la partie droite de l'interface.

On dispose d'un fichier CSV `iris.csv` contenant le jeu de données "iris". Il contient la description de 150 spécimens d'iris (fleurs) de 3 espèces différentes. Chaque description est composée de 4 attributs numériques (longueur et largeur des sépales et des pétales), et d'un 5ème attribut qui est la classe de cet exemple (c'est-à-dire l'espèce d'iris auquel il appartient).

Tâche 2.2 *Trouver un nœud permettant de lire le fichier* `iris.csv` *et suivre les étapes ci-dessus.*

Indices : chercher parmi les nœuds d'entrées-sorties (IO). Vérifier que le fichier a été correctement chargé avec la commande de visualisation `File table`.

Avant d'appliquer des algorithmes de fouille de données il est utile de se familiariser avec les données par des statistiques et visualisations simples.

Tâche 2.3 *Appliquer le nœud* Statistics *au tableau contenant les données "iris". Noter la distinction entre attributs numériques et attributs nominaux.*

Lorsque les données appartiennent à plusieurs classes, il est utile d'associer une couleur à chaque classe afin d'améliorer les visualisations de ces données.

Tâche 2.4 *Utiliser le nœud* Color Manager *et appliquer le au tableau de données pour associer une couleur à chaque classe. La sortie de ce nœud est un tableau "coloré".*

Tâche 2.5 *Appliquer les nœuds* Scatter Plot Matrix *et* Scatter Plot *du répertoire de nœuds* Views *au tableau coloré pour observer les corrélations entre les classes et les combinaisons de deux attributs numériques. Jouer avec les vues produites.*

Question 2.6 *Quelle combinaison de deux attributs (largeur/longueur des sépales/pétales) sépare le mieux les 3 classes ? Pouvez-vous énoncer des critères permettant de prédire la classe d'un iris à partir des valeurs de ces deux attributs ?*

Bien que ce soit au programme de FST plutôt que de FSY, nous allons apprendre un arbre de décision à partir des données pour découvrir automatiquement des critères permettant de prédire la classe d'un iris.

Tâche 2.7 *Appliquer le nœud* Decision Tree Learner *au tableau coloré pour produire un arbre de décision. Comparer les critères trouvés aux vôtres.*

3 Un premier exemple de règles d'association

Il s'agit ici de construire un workflow pour la recherche de règles d'associations sur les données du golf.

Tâche 3.1 *Créer un nouveau workflow et charger le fichier* `weather.nominal.arff` *comme donnée d'entrée. Il s'agit d'une version déjà discrétisée des données du golf.*

Tâche 3.2 *Ajouter des nœuds pour obtenir des statistiques et visualisations des données. Ne pas hésiter à faire des essais avec les nœuds du répertoire* Views.

Tâche 3.3 *Trouver le nœud permettant le calcul des règles d'association et tenter de l'appliquer au fichier de données. Quel est le problème ?*

Comme le plus souvent en fouille de données, il est nécessaire de préparer les données en fonction de l'algorithme à appliquer. Ici, le nœud en question a besoin d'une colonne de type *BitVector*, où chaque bit correspond à un item. Pour cela, il y a le nœud *Create Bit Vector* dans le même répertoire que le nœud calculant les règles d'association. Mais avant de l'appliquer, il faut déjà propositionaliser les données, c'est-à-dire se ramener à des 0/1 comme valeurs.

Tâche 3.4 *Utiliser le nœud* One to Many *pour effectuer la propositionalisation du tableau de données. Vérifier le résultat obtenu. Est-ce correct ? Quelle est la transformation appliquée aux données ?*

Tâche 3.5 *Appliquer maintenant le nœud* Create Bit Vector *aux données propositionalisées. Regarder le résultat et expliquer le.*

Tâche 3.6 *Appliquer maintenant le calcul des règles d'associations et jouer avec les options de configurations. Trouver le moyen de fixer le support minimum et la confiance minimale. Trouver le moyen de ne générer que les itemsets fréquents, pas les règles.*

Suite à la propositionalisation les valeurs sont devenues des noms de colonnes, ce qui peut entraîner des confusions. Par exemple, l'item `yes` concerne-t-il la présence de vent ou le fait d'aller jouer au golf ?

Tâche 3.7 *Trouver le moyen de renommer certaines colonnes dans les données de façon à rendre les itemsets et règles plus lisibles. Indice : KNIME permet d'insérer des nœuds partout, pas seulement en fin de chaine !*

Si l'objectif est de prédire si on peut jouer ou pas en fonction des autres critères, alors on peut se contenter des règles dont la conséquence est une valeur de l'attribut `play`.

Tâche 3.8 *Trouver le moyen de filtrer la liste des règles pour ne garder que celle dont la conséquence est une valeur de* `play`. *Indice : le mot-clé est "filtrer".*

4 Un deuxième exemple de règles d'associations

Le fichier `adult.csv` contient des données extraites d'un recensement de la population américaine. Le but de ces données est initialement de prédire si quelqu'un gagne plus de 50.000 dollars par an. On va transformer un peu les données avant d'extraire les règles d'association.

Tâche 4.1 *Créer un nouveau workflow et charger le fichier* `adult.csv` *comme données d'entrée.*

Les nœuds de transformation de données se trouvent pour la plupart dans le répertoire *Manipulation*. On y trouve des nœuds pour filtrer les colonnes ou les lignes, pour appliquer des transformations aux valeurs d'une colonne, pour fusionner des colonnes ou des lignes, ou au contraire pour les éclater.

Les données comportent souvent des attributs inutiles : numéro de dossier, nom, date de saisie... Il est possible d'en supprimer à la main, à condition de connaître le domaine. On peut aussi lancer un algorithme de fouille de données et regarder les attributs qui ont été utilisés : soit ceux-ci sont pertinents et il est important de les consever, soit ils sont tellement liés à la classe qu'à eux seuls ils emportent la décision (pensez à un attribut qui serait la copie de la classe).

Tâche 4.2 *Montrer à l'aide des statistiques des attributs, qu'on peut ignorer l'attribut* `fnlwgt`*. Supprimer-le !*

Supprimer également les attributs `capital-gain` *et* `capital-loss`*.*

Certains algorithmes ont besoin d'attributs discrets pour fonctionner (ex., règles d'associations, analyse de concepts), d'autres n'acceptent que des attributs continus (ex., réseaux de neurones, plus proches voisins). D'autres encore acceptent indiféremment des attributs des deux types.

Question 4.3 *Quels sont les attributs restants qui ont besoin d'être discrétisés ?*

Tâche 4.4 *Trouver et appliquer un nœud pour discrétiser ces attributs. Indice : le mot-clé est* bin*. Configurer le nœud pour découper l'ensemble des valeurs en 3 intervalles. Consulter et expérimenter avec les différentes options de configuration.*

Maintenant que les données sont discrétisées, il est possible de rechercher les règles d'association comme à la section précédente.

Tâche 4.5 *Compléter le workflow de façon à produire un ensemble de règles d'association. Jouer avec les différentes options de façon à trouver un ensemble de règle "intéressant".*

Tâche 4.6 *Sélectionner les règles de classification, c'est-à-dire avec les valeurs de l'attribut* `class`*. Adapter à nouveau les options et trier les règles pour placer les règles les plus importantes en haut du tableau.*

Question 4.7 *Quelles conclusions tirez-vous des résultats obtenus ?*