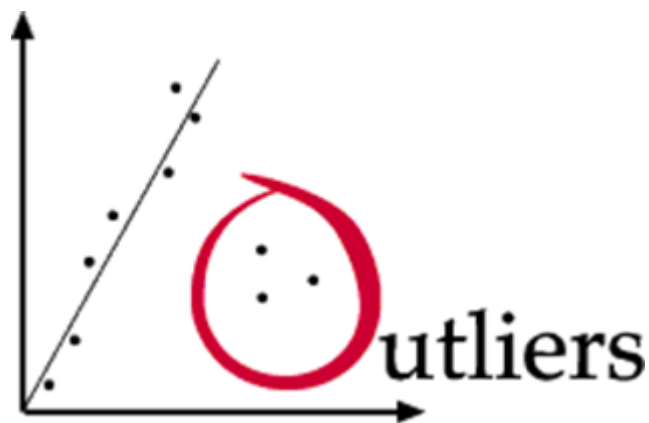


Détection des outliers au sein de données à haute dimensions



Membres du groupe :

M. Mehdi CHAOUNI – M. Maxime BRIENS – M. Valentin GODE

M1 SID

Tuteurs de projet:

Mme Wahiba BAHOUN - M. Riad MOKADEM

Historique du document

Date	Numéro de Version	Chapitre concerné	Cause modification
01/02/2017	V 1.0	Tous	Création du rapport
03/02/2017	V 1.1	Chapitre 6	Début de la recherche bibliographique
06/02/2017	V 1.2	Chapitre 7	Récupération des données
04/03/2017	V 1.3	Chapitre 8	Implémentation des méthodes
20/03/2017	V1.4	Tous	Fin du rapport

Sommaire

Historique du document	2
Sommaire	3
Remerciements	5
1 Introduction.....	6
1.1 Objet.....	6
1.2 But du document.....	6
1.3 Domaine d'application	6
1.4 Présentation du document.....	7
2 Document et vocabulaire de base.....	8
2.1 Document applicable.....	8
2.2 Document de référence	8
3 Terminologie	8
3.1 Termes relatifs aux statistiques.....	8
3.2 Termes relatifs au stockage des données	9
3.3 Planification.....	10
4 Équipe de développement et tâche	11
4.1 Ressources humaines	11
4.2 Planification.....	11
4.3 Planning prévisionnel	12
4.4 SADT	13
5 Démarche et outils de développement.....	16
5.1 Démarche utilisée.....	16
5.2 Outils utilisés	16
6 État de l'art des méthodes de détection des outliers	17
6.1 Méthodes de détection	17
6.2 Zoom sur certaines méthodes, implémentées.....	18
7 Données.....	21
7.1 Récupération et stockage des données	21
7.2 Mise en forme et insertions des documents.....	22
7.3 Rapide descriptif de nos variables.....	23
7.4 Mise en forme de nos jeux de données et ouverture dans python.....	23
7.5 Création de la collection « navigator » dans MongoDB et insertion des documents.....	24
7.6 Récupération et stockage des méthodes :	24
7.7 Intégration des documents au fil de l'eau.....	26

8	Méthodologie	31
8.1	Gestion des valeurs manquantes	31
8.2	ACP	33
8.3	DBSCAN	33
8.4	MCD (Robust Covariance)	36
8.5	SVM One Class.....	38
8.6	Arbre de décision	40
9	Tableau de bord	41
9.1	Description des outliers.....	43
9.2	Comparaison des résultats de chaque méthode.....	48
10	Démarche et assurance qualité.....	50
11	Bilan	58
11.1	Bilan client	58
11.2	Bilan Fournisseur	58
12	Difficultés rencontrées	59
13	Conclusion	60
	Annexe 1 : Bibliographie	61
	Annexe 2 : Transformation des données via l'ETL	62
	Annexe 3 : QQplot Robust.....	63
	Annexe 4 : QQplot WT10G	65

Remerciements

A étoffer, compléter, reformuler

Nous tenons tout d'abord à remercier Madame Josiane MOTHE, qui a tenu un rôle d'encadrante au cours de ce projet, mais qui nous a aussi guidés, tout au long de ce projet très intéressant. Nous la remercions également pour le sujet très intéressant qu'elle nous a proposé.

Nous remercions également Madame Wahiba BAHOUN pour les conseils avisés et l'encadrement qu'elle nous a prodigué tout au long de ce projet. Merci également à Monsieur Riad MOKADEM pour sa présence et son encadrement.

1 Introduction

1.1 Objet

Ce projet est réalisé dans le cadre de notre formation de 1^{ère} année de Master en Statistique et Informatique Décisionnelle (SID). Notre sujet permet de mettre en application certaines notions et méthodes vues en cours de machine learning, ou statistiques ainsi qu'en génie logiciel, dans le cadre d'une application concrète. De plus, et c'est notamment en cela que notre sujet est intéressant, ce projet est l'occasion de nous familiariser avec de nouvelles méthodes que nous n'avons pas vues en cours. Nous nous sommes ainsi renseignés sur certaines méthodes de détection de données aberrantes, ainsi que de stockages de données de type « NoSQL ».

Nous axons ce sujet vers les données à haute dimensions, car nous connaissons déjà certains moyens de mettre en avant des individus aberrants (en effectuant des tests d'hypothèses par exemple). Cependant nous souhaitons nous intéresser, dans ce sujet, à des méthodes plus performantes, applicables sur des bases de données relativement volumineuses, tant en nombre d'individus qu'en nombre de dimensions.

1.2 But du document

Les objectifs de ce projet sont les suivants :

- Faire un état de l'art des méthodes de détection des outliers (données aberrantes)
- Stocker les données fournies
- Implémenter les méthodes de détection retenues
- Stocker les résultats obtenus pour chaque méthode implémentée
- Réaliser un tableau de bord pour illustrer les résultats obtenus

1.3 Domaine d'application

Ce projet est destiné dans un premier temps à Mme Josiane MOTHE, qui a d'ailleurs été à l'initiative de ce sujet pour le projet. Cela lui permettra d'avoir un état de l'art des méthodes connues permettant la mise en lumière d'outliers, ainsi que quelques informations, voire exemples, sur leur implémentation. Dans un second temps, ce sujet nous servira dans le cadre du projet « Tableau de Bord », supervisé par MmeWahibaBAHSOUN et Mr Riad MOKADEM.

1.4 Présentation du document

Le chapitre intitulé « Documents et vocabulaire de base » a pour objectif de définir la problématique qui nous a guidé tout au long du déroulement de ce projet. Il permettra également de préciser quels sont les différents documents utilisés au cours du projet, ou encore la terminologie qui sera employée tout au long de ce rapport.

Le chapitre « Equipe de développement et tâches » permet de présenter l'équipe déployée sur ce projet, ainsi que la répartition des rôles au sein de cette équipe.

Les différents outils utilisés, ainsi que les méthodes mises en œuvre dans le projet sont présentés dans le chapitre intitulé « Méthodes et outils de développements ».

Le chapitre « Démarche de développement » a pour objectif de décrire la démarche mise en place afin de mener ce projet à terme.

Dans le chapitre « Démarche et assurance qualité » sont répertoriées les différentes revues qui ont pu être effectuées au cours du projet.

Le chapitre « Gestion de configuration » a pour objectif de dresser un bilan sur le déroulement global du projet. Il décrit également les différentes difficultés rencontrées.

Les chapitres « Conclusion », « Bibliographie » et « Glossaire » traitent respectivement les points suivants :

- Conclusion globale du projet
- Supports utilisés pour mener à bien le projet
- Récapitulatif du vocabulaire employé dans le rapport

2 Document et vocabulaire de base

2.1 Document applicable

Ce projet a pour but d'effectuer un état de l'art des méthodes de détection d'outliers. Le terme « outlier » définit tous les individus aberrant, remarquable, potentiellement anormal ou fruit d'une erreur, quelle qu'elle soit. Pour mener à bien cette mission, il nous faut stocker les données (qui nous ont été fournies), afin d'implémenter certaines méthodes de détection d'outliers à partir de ces données. L'objectif final étant la restitution des résultats obtenus sous la forme d'un tableau de bord.

Pour atteindre cet objectif, le projet est scindé en plusieurs parties :

- Recherche bibliographique sur les méthodes de détection existantes
- Récupération et stockage des données fournies par le client
- Choix de quelques méthodes à mettre en œuvre
- Implémentation des méthodes retenues
- Stockage des résultats obtenus selon chacune des méthodes implémentées
- Présentation et comparaison des résultats obtenus, sous forme de tableau de bord

2.2 Document de référence

Pour la bonne réalisation du projet, nous nous sommes basés sur certains documents existants, notamment parmi les documents suivants :

- Cours de Statistiques (statistiques descriptives, algorithmes de machine learning et modélisation statistique)
- Cours de Génie Logiciel

3 Terminologie

3.1 Termes relatifs aux statistiques

- **Clustering** : Regroupement de plusieurs données identiques ou similaires en une classe, où cluster.
- **Données aberrantes, outliers** : Les outliers sont les signes d'erreurs potentielles dans un jeu de données. Dans certains cas, les rechercher peut permettre de se concentrer sur des données pouvant être issues de problèmes quelconques (panne, performance, etc., selon les données). Mettre en avant les outliers peut permettre de comprendre l'origine de ces problèmes potentiels.

- **Apprentissage supervisé** : Il s'agit d'une technique ayant pour but de produire automatiquement des règles (de prédiction, de clustering etc.) à partir d'un échantillon d'apprentissage.
- **ACP** : Acronyme de « Analyse en Composantes Principales ». Il s'agit d'un procédé statistique ayant pour but de conserver un maximum d'information, en un minimum de variables.
- **DBSCAN** : Acronyme de l'anglais « Density-Based Spatial Clustering of Applications with Noise ». Il s'agit d'un algorithme de partitionnement de données, basé sur la densité, ayant la spécificité de réserver une classe pour le bruit, correspondant (dans notre cas) aux outliers potentiels.
- **MCD** : Acronyme de l'anglais « Minimum Covariance Determinant ». Il s'agit d'un nom qualifiant la méthode aussi appelée Robust Covariance.
- **SVM** : Acronyme de l'anglais « Support Vector Machines ». Il s'agit d'une méthode d'apprentissage supervisée, utilisée pour la détection de la nouveauté, à partir d'un échantillon d'apprentissage.
- **QQ-plot** : Diagramme Quantile-Quantile, ayant pour objectif d'évaluer la pertinence de l'ajustement d'une distribution donnée à un modèle théorique (par exemple pour tester la normalité d'une variable).
- **Validation croisée** : Technique permettant de savoir à quel point les résultats d'une analyse statistique peut se généraliser sur un jeu de données indépendant. Cette technique permet aussi d'obtenir le résultat, de la méthode correspondante, pour chaque individu, plutôt qu'uniquement pour les individus testés. Il existe plusieurs méthodes de validation croisée (cross-validation en anglais), comme notamment, l'une des plus communes, la méthode « ten-fold ».

3.2 Termes relatifs au stockage des données

- **NoSQL** : Abréviation de l'anglais « Not Only SQL ». Ce terme désigne les bases de données qui ne sont pas fondées sur l'architecture classique des bases de données relationnelles, mais fonctionnant sur le principe d'une architecture machine en clusters.
- **Collection** : Une collection permet de stocker tous les documents. Elle peut être considérée comme l'équivalent NoSQL des tables des bases de données relationnelles.
- **Documents** : Ils sont comparables aux enregistrements des bases de données relationnelles. Cependant, contrairement à celles-ci, la structure des champs d'un enregistrement peut être différente d'un enregistrement à un autre au sein d'une même collection.
- **Transformations** : Ensemble d'étapes nécessaire pour réaliser l'intégration des données.
- **Job** : Tâche permettant d'exécuter automatiquement, selon une période, les transformations réalisées.

- ***MongoDB*** : Base de données « NoSQL » permettant de manipuler des objets structurés sans schéma prédéterminé. Les clés peuvent ainsi être ajoutées à tout moment, sans nécessiter de reconfiguration de la base.
- ***ETL*** : Acronyme de l'anglais « Extract Transform and Load ». Il s'agit d'un outil souvent utilisé en entreprise, permettant d'automatiser le processus d'intégration de données.

3.3 Planification

- ***GANTT*** : Diagramme utilisé pour faire des plannings. Il permet, dans le cadre d'un projet, d'illustrer les différentes tâches, les délais et les ressources humaines nécessaires.
- ***SADT*** : Acronyme de l'anglais « Specified Analysis and Design Technique ». Il s'agit d'une démarche de développement permettant de voir, à différents niveaux, les différentes étapes de la réalisation du projet.

4 Équipe de développement et tâche

4.1 Ressources humaines

Nom de la personne	Rôle dans le projet
Chaouni Mehdi	Chef de projet
Briens Maxime	Responsable de développement
Godé Valentin	Responsable rédaction et données

Mehdi CHAOUNI en tant que chef de projet, a pour rôle de coordonner le bon déroulement du projet en attribuant les tâches à chacun des membres, et en veillant à ce que les délais soient respectés. Il a également pour rôle de rechercher les méthodes existantes d'extraction d'outliers. Il réalise également la recherche de la documentation susceptible de pouvoir participer à répondre à la problématique. Le choix des méthodes à implémenter sera ensuite débattu entre tous les membres du groupe.

Maxime BRIENS est le responsable de développement. Son rôle est d'implémenter les méthodes qui sont retenues, et de les mettre en application sur les données qui nous ont été fournies (relatives aux performances de moteurs de recherches). Il a également pour tâche de prétraiter ces données, contenant de nombreuses valeurs manquantes, afin de rendre les algorithmes opérationnels.

Valentin GODÉ est le responsable de rédaction des revues et des rapports, ainsi que des données. Il a pour rôle de créer une base de données appropriée aux données, permettant le stockage des données nettoyées, ainsi que le stockage des résultats des différentes méthodes sur les données.

4.2 Planification

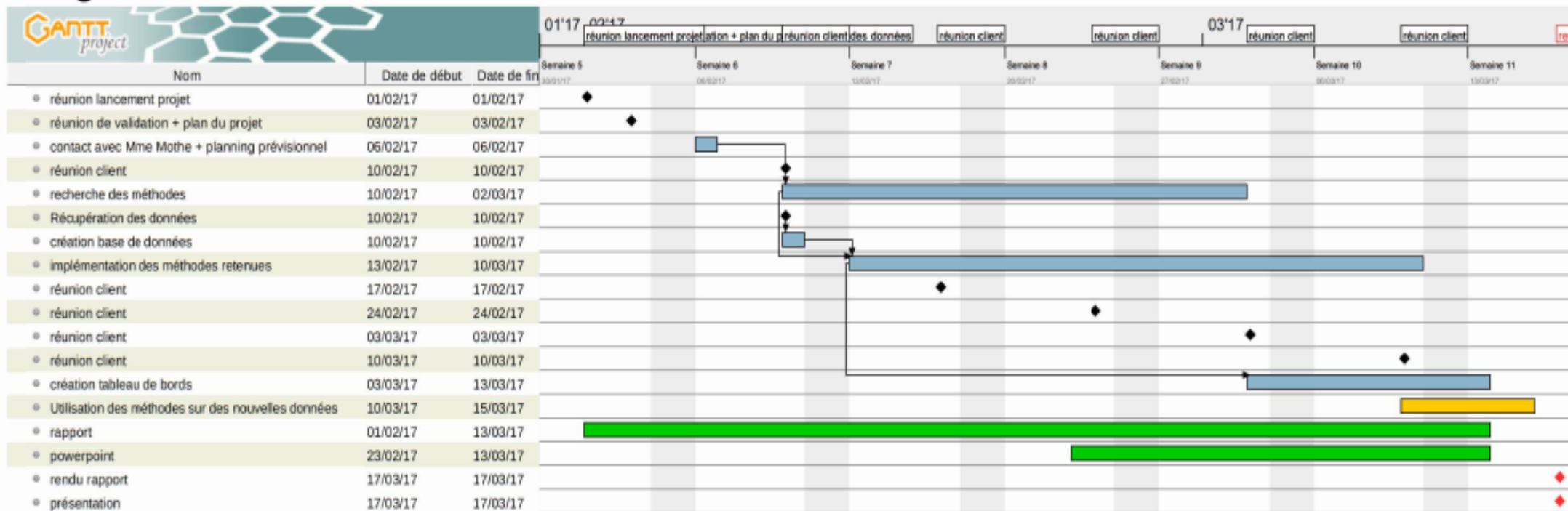
Pour le bon déroulement du projet, nous avons réalisé un diagramme de GANTT prévisionnel, nous permettant d'identifier et de séparer les différentes tâches de notre projet. Cette étape est cruciale pour l'organisation de notre projet : cela nous permet de voir comment il s'inscrit dans le temps, et nous permet d'avoir un premier aperçu sur la répartition des rôles.

Nous avons ainsi identifié les tâches suivantes :

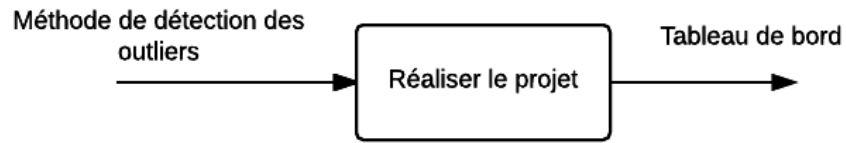
- Recherche de méthodes permettant de détecter des outliers
- Récupération, nettoyage et stockage des données dans la BD
- Implémentation des méthodes sélectionnées
- Interprétation des résultats, et comparaisons entre les résultats issus des différentes méthodes

4.3 Planning prévisionnel

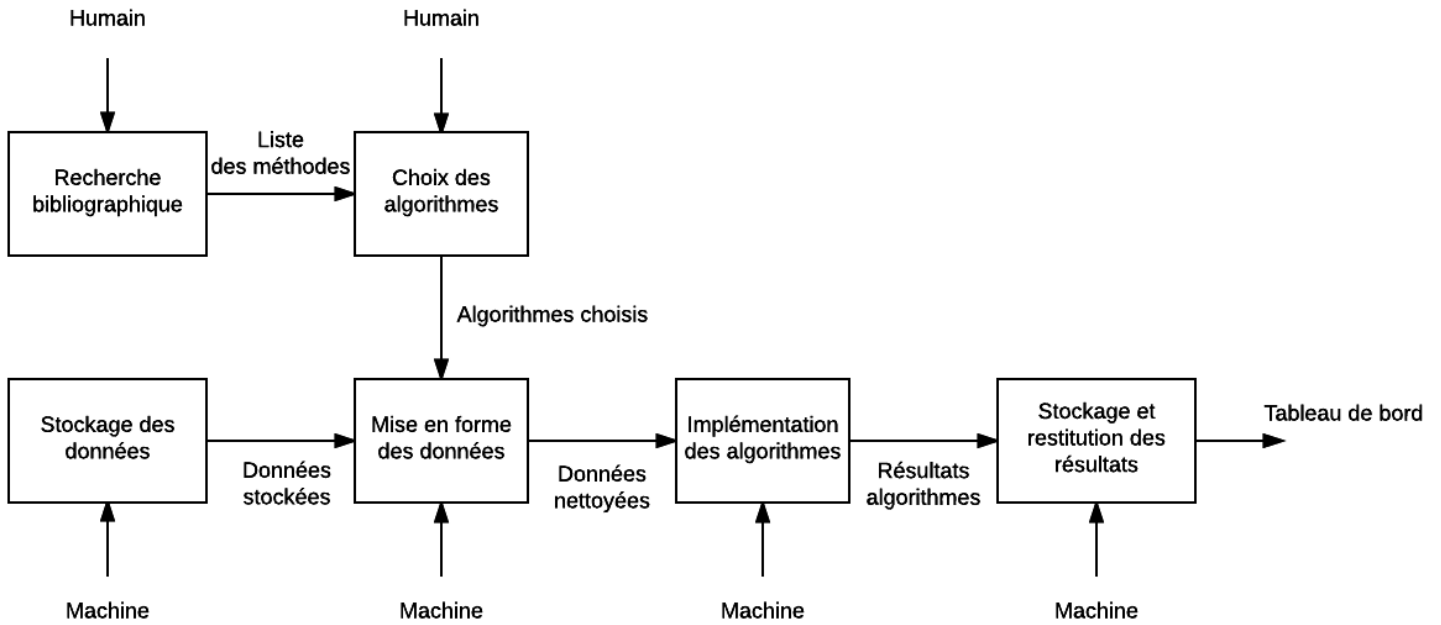
Diagramme de Gantt



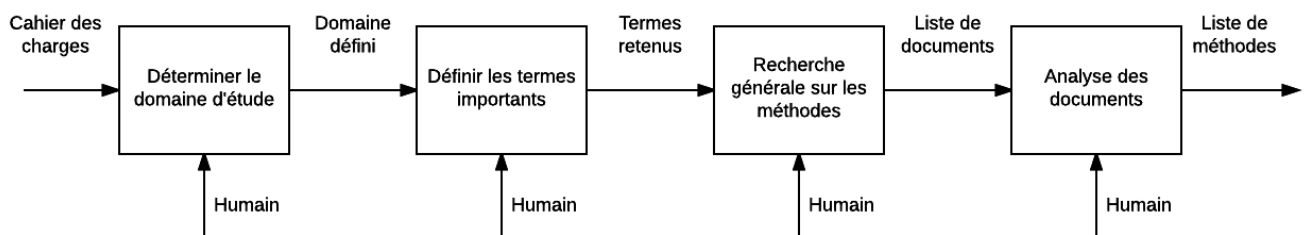
4.4 SADT



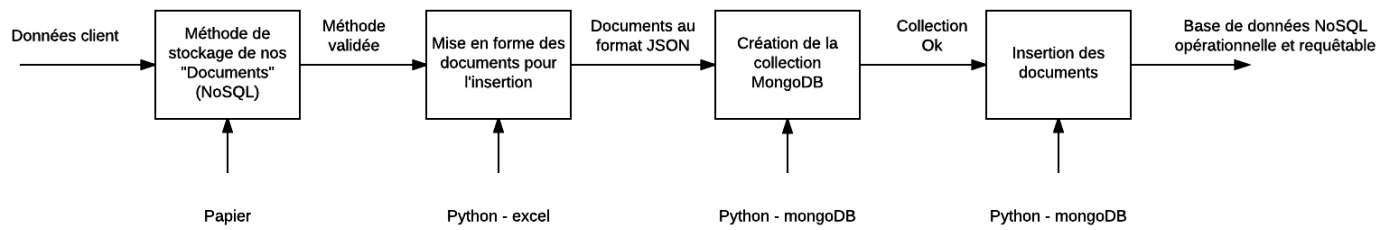
A-0 : Tableau de Bord



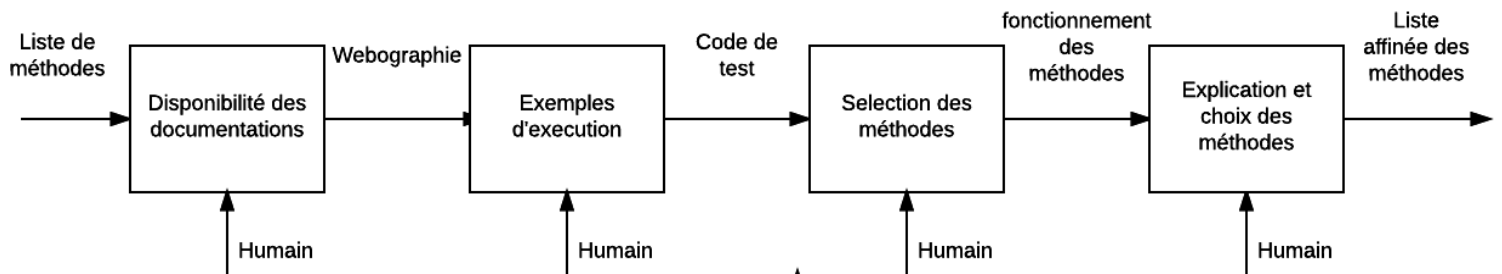
A0 : Réaliser le tableau de bord



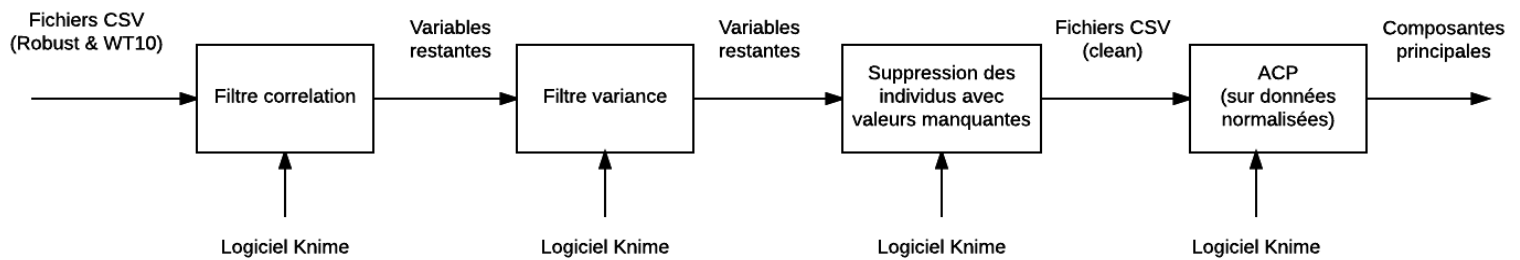
A0-1 : Recherche bibliographique



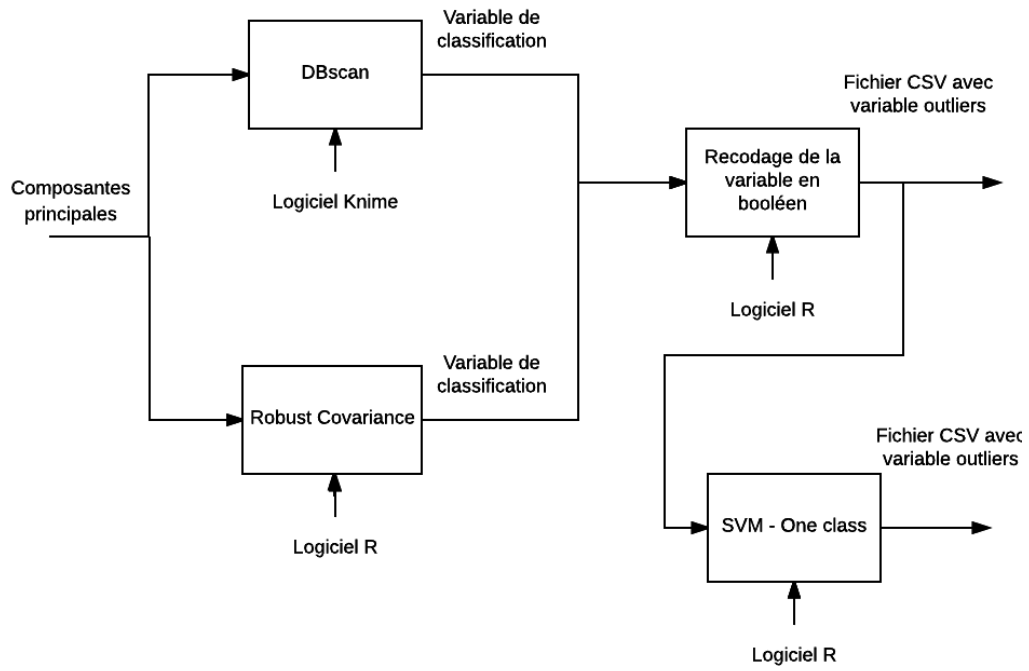
A0-2 : Stockage des données



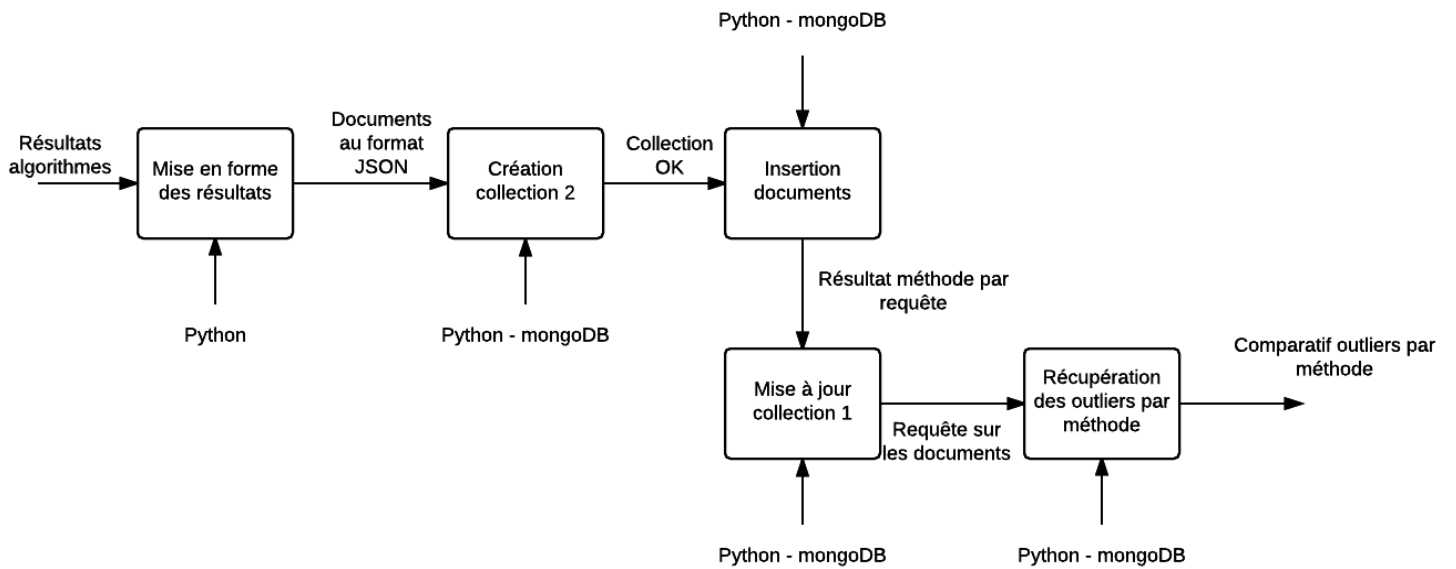
A0-3 : Choix des algorithmes



A0-4 : Mise en forme des données



A0-5 : Implémentation des algorithmes



A0-6 : Stockage et restitution des résultats

5 Démarche et outils de développement

5.1 Démarche utilisée

La démarche utilisée dans le cadre de ce projet est l'intégration continue. Le schéma suivant illustre cette démarche appliquée au cadre de notre projet :

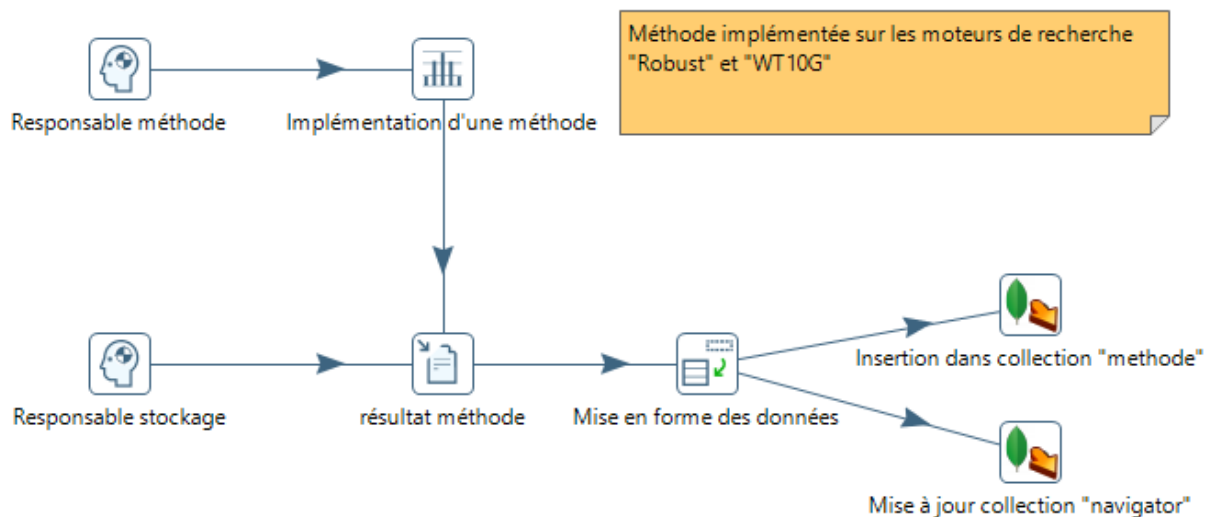


Figure 1 : Démarche intégration continue méthode / mongoDB

5.2 Outils utilisés

Les outils utilisés pour mener à bien ce projet sont :

- MongoDB pour le stockage de nos documents
- Python pour la mise en forme des données, l'insertion et l'interrogation des documents dans mongoDB
- KNIME et R pour l'implémentation des méthodes statistiques, et pour la productions des graphiques.

6 État de l'art des méthodes de détection des outliers

Dans le but de définir une liste de méthodes potentiellement capable de répondre à notre problématique de mise en lumière d'outliers au sein de données à hautes dimensions, nous avons suivi une méthodologie afin de récupérer les documents les plus pertinents. La méthodologie se décompose en 4 étapes :

- Définition du domaine d'étude
- Rassembler les informations relatives aux méthodes
- Établir une bibliographie
- Classement des méthodes à étudier

6.1 Méthodes de détection

Suite à l'application de cette méthodologie, il s'agit d'établir une liste de méthodes connues permettant de détecter des données aberrantes dans un jeu de données.

Test Tau de Thompson modifié : Cette méthode est utilisée pour déterminer si une valeur est considérée comme aberrante.

L'algorithme fonctionne de la manière suivante : On calcule l'écart-type ainsi que la moyenne de la série, puis on détermine le seuil de rejet. Une seule donnée aberrante est trouvée, puis retirée, à la fois. L'écart-type et la moyenne de la série sont recalculés pour chaque itération, jusqu'à ce qu'il n'y ait plus de données aberrantes.

Grubbs test : Le test de Grubbs est utilisé pour détecter les données aberrantes en fonction de la dispersion de moyennes. Pour cela, il faut comparer les valeurs absolues des écarts réduits. Si la valeur obtenue est supérieure à la valeur critique, alors la valeur est considérée comme étant une donnée aberrante et doit être retirée. Ce test ne fonctionne que sur une seule valeur et sur une variable en même temps. Il faut donc répéter l'algorithme à chaque fois que l'on détecte une donnée aberrante, et pour chaque variable. De plus, le nombre exact de données aberrantes doit être connu.

Tietjen-moore test : Ce test est similaire au test de Grubbs. Il diffère par le fait qu'il puisse gérer plusieurs valeurs aberrantes sur une variable. Il faut donc répéter l'algorithme pour chaque variable. De plus, le nombre exact de données aberrantes doit être connu.

GeneralizedExtreme Studentized Deviate test : Ce test est une version améliorée des deux tests cités ci-dessus. Il fonctionne de la même manière et permet de détecter à la fois si une variable possède une ou plusieurs données aberrantes. Contrairement aux deux précédents tests, celui-ci ne nécessite pas le nombre exact de données aberrantes mais seulement le nombre maximal.

Méthodes d'Ensemble Learning et de Boosting : Le but des méthodes d'ensemble learning est de combiner la prédiction d'une multitude d'estimateurs construits par des algorithmes d'apprentissage dans le but d'améliorer la robustesse d'un seul estimateur (moins affecté par les données aberrantes). Les estimateurs peuvent être générés de façon séquentielle ou parallèle. Pour obtenir un bon ensemble, les bases d'apprentissages doivent être le plus précis possible (ex : application de méthodes de validation croisée). Les méthodes de boosting découlent des méthodes d'ensemble et fonctionnent de la manière suivante : les estimateurs sont construits de façon séquentielle et sont utilisés pour réduire le biais des estimateurs combinés, le but étant d'associer plusieurs petits modèles pour créer un ensemble performant. (Exemple de méthodes : Sequential ensemble learning, AdaBoost, etc.).

Replicator Neural Network : Cette méthode est une variation du modèle de régression usuel. Le RNN tente de reproduire les modèles d'entrée, dans la sortie. Sur l'échantillon d'apprentissage, le poids de RNN est ajusté pour minimiser l'erreur quadratique moyenne. Cela a pour conséquence que le modèle commun a plus de chance d'être bien reproduit par le RNN d'apprentissage, ce qui implique une reproduction plus faible du nombre de données aberrantes.

Note : certaines méthodes nous semblaient intéressantes mais n'ont pas été traitées (par exemple Isolation forest, K-means clustering, Birch, Robustnessregression), car elles étaient trop compliquées à prendre en main et implémenter, pas assez détaillées, ou encore pas adaptées à notre cas.

6.2 Zoom sur certaines méthodes, implémentées

Par souci de simplicité, de temps, et d'aise vis-à-vis des données mises à disposition, il était impossible, dans le cadre de ce projet, d'implémenter chacune de ces méthodes.

Ces 3 méthodes décrites ci-dessous ont été implémentées sur notre jeu de données.

One Class SVM : La méthode SVM est une méthode de classification. Cependant, notre problématique ne consiste pas à effectuer une classification, mais nous utiliserons une dérivée de cette méthode : la SVM à une classe.

Cette méthode est une méthode d'apprentissage supervisé. Elle a pour but de détecter la nouveauté : elle se base sur un échantillon d'apprentissage, et forme une classe de l'échantillon test correspondant composée d'individus similaires à ceux de l'échantillon d'apprentissage. Les individus ne faisant pas parti de la classe sont des individus « nouveaux ». Pour appliquer cette méthode à notre contexte de détection d'outliers, il s'agit de renseigner un échantillon d'apprentissage composé uniquement d'inliers (sans outliers). Il faut ensuite tester le modèle obtenu, sur les données restantes. L'évaluation du modèle est effectuée via une matrice de confusion.

Ne disposant pas de données annotées (outliers ou non) à l'origine, il nous faut implémenter une méthode différente de détection d'outliers en amont du one-class SVM, et considérer les résultats obtenus comme étant une vérité générale, sur laquelle la méthode SVM peut se baser.

Afin d'obtenir un résultat (outlier ou non) relatif à chaque individu, il nous faut mettre en application une méthode de validation croisée. N'étant pas adaptée à notre contexte, nous avons dû définir une variante. Nous verrons pourquoi, et comment, dans la partie relative à l'implémentation des méthodes.

DBSCAN : Il s'agit d'un algorithme qui partitionne des données en plusieurs clusters en fonction de leur densité estimée, le nombre de cluster n'est donc pas à spécifier. Cependant, le nombre minimum de point se trouvant dans un rayon de distance « epsilon » doit être renseignée afin de pouvoir définir un cluster. Une partition est réservée pour les données aberrantes qui seront ensuite représentées sur un graphique afin de mieux les distinguer des autres données partitionnées. Un individu est considéré comme outlier si sa distance (la méthode de calcul de la distance peut être définie (Mahalanobis, Euclidienne, etc.)) au cluster le plus proche dépasse un certain seuil spécifié.

Dans notre cas, DBSCAN a été utilisé en amont de la méthode One class SVM. Nous nous sommes ensuite basés sur le résultat obtenu comme étant « vérité absolue » puisque nous ne possédons pas d'échantillon d'apprentissage annoté.

MCD (Robust Covariance) : Cette méthode a pour hypothèse de validité de s'appliquer sur des données suivant une loi normale. Il s'agit d'un estimateur très robuste (c'est-à-dire qu'il n'est pas dépendant des valeurs influentes, potentiellement aberrantes). L'estimateur MCD est déterminé par un sous-ensemble de points (de taille h) qui minimise le déterminant de la matrice de variance-covariance sur tous les sous-ensembles de taille h . Il calcule notamment les distances robustes, ainsi que les distances de Mahalanobis. A l'issue de l'implémentation

de la méthode, nous nous intéressons au vecteur des poids, et plus la valeur est élevée, plus l'individu a des chances d'être considéré comme un outlier.

7 Données

7.1 Récupération et stockage des données

Pour mener à bien notre étude, nous devons stocker deux types de données :

- Les données relatives aux moteurs de recherche fournies par Mme MOTHE
- Les données relatives aux méthodes utilisées pour faire parler ces données (pour éviter de devoir relancer les algorithmes à chaque fois)

Le cas ci-dessus ne s'adapte pas à un schéma relationnel car la structure de nos individus (requêtes sur les moteurs de recherches) varie. En effet, celles-ci ne sont pas expliquées par les mêmes variables et par le même nombre de variables, cela complique leur stockage dans une base de données relationnelle, qui ne serait pas optimal.

MongoDB semble être une bonne alternative, au vu des données. De par son aspect NoSQL, il va nous permettre de stocker plus facilement et plus proprement nos fichiers (ou documents dans ce cas).

Dans un premier temps, une collection sera créée pour stocker nos données. Les documents qui, bien que structurés d'une manière différente (différents noms et nombres des variables par moteur de recherche), donnent les caractéristiques d'une requête et ne nécessitent donc pas l'utilisation de plusieurs collection.

Les documents se présentent sous cette forme :

```
{
  "_id" : .... (Identifiant automatique)
  "ID_R": "400",
  "TF-IDF": "0.0044",
  ...,
}
```

Dans un second temps, une collection « method » sera créée pour stocker les résultats obtenus par les méthodes implémentées. Les champs relatifs aux résultats obtenus sur nos données seront ajoutés à nos individus, via une mise à jour de nos documents.

- Structure de la collection « method » :

```
{ "Code_methode" : 1,
  "Nom" : "One class SVM", }
```

- Mise à jour de la collection « navigator » :

```
{ "query_id" : 1, ...,
  "Code_methode" : 1,
  "outlier" : "Y/N" }
```

7.2 Mise en forme et insertions des documents

Nous disposons de 4 fichiers csv :

- Robust_BM25_title_perf : Ce fichier contient toutes les informations relatives aux performances (évaluations) du moteur de recherche Robust. Il est composé de 3 variables quantitatives :
 - « topic » : identifiant de la requête
 - « P_10 » : Précision du moteur de recherche via la requête sur les 10 premiers résultats obtenus via la requête
 - « Map » : Nombre de documents corrects retournés par la requête en fonction du nombre de documents à retourner.
- WT10G_BM25_title_perf : Ce fichier contient toutes les informations relatives aux performances (évaluations) du moteur de recherche WT10G (14 variables).
- TREC-ROBUST_BM25_301_700C.query et WT10G_BM25_451-550C.query : Ces fichiers contiennent toutes les informations relatives aux caractéristiques des requêtes.

Ils sont composés respectivement de 217 et 235 variables quantitatives.

7.3 Rapide descriptif de nos variables

La majorité de nos variables ont des noms qui comportent des éléments récurrents. Il s'agit ici de décrire rapidement ce à quoi elles correspondent, ce qu'elles signifient :

- Meronym, synonym, hyponym : Champ provenant de wordnet (relations qui existent entre les mots, liens de synonymies, mots génériques, ...). Il s'agit de calculer, pour chaque requête, le nombre moyen de mots synonymes, hyponymes,... (ou minimum, ou maximum, ou écart-type, ou variance)
- bm25... : Score des documents selon le modèle BM25
- WSINGLEFIELDMODEL... : Autre modèle
- AVGSIZE : Taille moyenne des mots de la requête
- AVGSYNSET : Nombre moyen de synonymes dans chaque mot de la requête
-1 : Concerne le corps du document
-0 : Concerne le titre de nos variables

7.4 Mise en forme de nos jeux de données et ouverture dans python

Afin de respecter la mise en forme de notre modèle de données, un champ relatif au nom du moteur de recherche a été rajouté, dans les 2 fichiers '*.*_perf'.

Nous avons ensuite converti nos fichiers CSV en fichiers Excel, car l'ouverture de fichier Excel via python est plus performante pour ce format.

Voici un exemple d'ouverture et d'attributions clés – valeurs d'un fichier Excel :

```
Robust_BM25_perf = xlrd.open_workbook('C:/Users/Valentin/Desktop/SID_Donnees2017/Robust_BM25_title_perf.xls')
sh = Robust_BM25_perf.sheet_by_name(u'Robust_BM25_title_perf')

Robust_BM25 = []
Robust = {}
for i in range(1, sh.nrows):
    Robust = {"topic": sh.row_values(i)[0],
              "p_10": sh.row_values(i)[1],
              "map": sh.row_values(i)[2],
              "navigator": sh.row_values(i)[3]}
    Robust_BM25.append(Robust)
```

Une fois nos fichiers Excel importés, nous avons formé nos documents de la manière suivante :

- Tri des fichiers par « query_id »
- Insertion des champs relatifs aux variables de performances et des variables caractéristiques dans un seul document (1 ligne par requête contenant toutes les variables disponible) via une « jointure »

Voici un exemple de code effectuant ces étapes :

```
for i in range(len(TREC_Robust_BM25)):
    if TREC_Robust_BM25[i]["query_id"] == Robust_BM25[i]["topic"]:
        TREC_Robust_BM25[i].update(Robust_BM25[i])
```

Une fois nos documents formés, nous les avons enregistrés au format JSON, format obligatoire pour insérer des documents dans MongoDB.

7.5 Création de la collection « navigator » dans MongoDB et insertion des documents

```
#connexion python - mongodb
client = MongoClient()
client = MongoClient('mongodb://localhost:27017/')

#création d'une collection "navigator"
db = client.navigator

#insert robust dans collection navigator dans mongoDB
for i in range(len(Robust)):
    db.navigator.insert(Robust[i], check_keys=False)

#insert Wt10g dans mongoDB
for i in range(len(WT10G)):
    db.navigator.insert(WT10G[i], check_keys=False)

#verification insertion
#250 robust
db.navigator.count({'navigator': 'Robust'})

#100 WT10G
db.navigator.count({'navigator': 'WT10G'})

#interrogation Mongo et récupération des documents WT10G
W = {}
WT10G = []
for doc in db.navigator.find({'navigator': 'WT10G'}):
    W = doc
    WT10G.append(W)

#interrogation Mongo et récupération des documents Robust
R = {}
Robust = []
for doc in db.navigator.find({'navigator': 'Robust'}):
    R = doc
    Robust.append(R)
```

7.6 Récupération et stockage des méthodes :

Afin de stocker les informations relatives aux méthodes, nous avons créé une nouvelle collection « method » dans mongoDB.

Voici un exemple de documents insérés dans cette collection :

```
Method = {"Id_M":1, "Method_name":"DBSCAN"}
```

Ensuite, il nous faut mettre à jour nos documents de la collection « navigator » en fonction des résultats obtenus par la méthode.

Voici un exemple d'informations à rajouter sur nos documents :

query_id	outliers	id_M
451.0	N	1
452.0	N	1
453.0	N	1
454.0	N	1
455.0	N	1
456.0	N	1
457.0	N	1
458.0	N	1
459.0	N	1
460.0	N	1
461.0	N	1
462.0	Y	1

Tableau 1: Résultat algorithme

La mise à jour s'effectue de la manière suivante :

- Importation des fichiers contenant les « outliers » par méthode dans python
- Conversion de ces documents en JSON
- En parallèle, récupération des fichiers JSON insérés dans la collection « navigator »
- Création d'une fonction python permettant de faire la jointure entre des listes de dictionnaires
- Création des nouveaux documents
- Insertion des documents dans la collection

Voici un extrait du code python correspondant :

```
#Importation du fichier excel dans python
WT10G_dbscan = xlrd.open_workbook('C:/Users/Valentin/Desktop/SID_Donnees2017/WT10G_dbscan.xls')
sh = WT10G_dbscan.sheet_by_name(u'Feuille1')

WT10G_dbs = []
W_d = {}
for i in range(1, sh.nrows):
    W_d = {"query_id": sh.row_values(i)[0],
          "outliers": sh.row_values(i)[1],
          "id_M": sh.row_values(i)[2]}
    WT10G_dbs.append(W_d)

#récupération du fichier JSON inséré dans "navigator"
file2 = open('C:/Users/Valentin/Desktop/SID_Donnees2017/WT10.json')
Doc_W = json.load(file2)

#conversion de la clé de jointure en "float"
for i in range(len(WT10G_dbs)):
    WT10G_dbs[i]['query_id'] = float(WT10G_dbs[i]['query_id'])

#fonction permettant de faire la jointure entre nos documents
def merge_lists(l1, l2, key):
    merged = {}
    for item in l1+l2:
        if item[key] in merged:
            merged[item[key]].update(item)
        else:
            merged[item[key]] = item
    return [val for (_, val) in merged.items()]

#Jointure via cette fonction
Doc_W = merge_lists(Doc_W, WT10G_dbs, 'query_id')

#insertion des documents dans "navigator"
for i in range(len(Doc_W)):
    db.navigator.insert(Doc_W[i], check_keys=False)
```

7.7 Intégration des documents au fil de l'eau

Une fois une base de données créée, il est important de pouvoir la maintenir et de l'alimenter automatiquement. Pour cela, selon la méthodologie d'intégration continue décrite précédemment, nous avons via un l'ETL (ExtractTransform and Load) Pentaho Data Integration, réalisé cette intégration de données.


Voici son fonctionnement (interface de l'ETL) :

Dans un premier temps, on récupère le fichier contenant les informations relatives à la méthode implémentée sur nos données.



Figure 2 : Récupération et insertion de la nouvelle méthode

Voici un exemple d'informations à insérer dans la collection « méthode » :



New method

Résultats exécution


Historique | Trace | Statistiques

● \${TransPreview.FirstRows.Label} ○ \$

#	Id_M	Method_name
1	2	SVM

Figure 3 : exemple d'informations « méthode »

Ensuite, nous nous connectons à mongoDB et insérons ces données dans la collection correspondante :



Insert new method

Metrics | Prévisu

MongoDB Output

Step name: Insert new method

Configure connection | Output options | Mongo document fields | Create/drop indexes

Host name(s) or IP address(es): localhost

Port: 27017

Figure 4 : Connection Pentaho - mongoDB

Collection: Get collections

Figure 5 : Récupération de la collection associée

#	Name	Mongo document path	Use field name	JSON
1	Id_M		Y	N
2	Nom_method		Y	N

Figure 6 : Insertion des champs souhaités dans la collection

Une fois la collection « methode » alimentée, nous allons intégrer les résultats de l'algorithme à nos documents.

On commence par récupérer les fichiers contenant les résultats de nos algorithmes (au format .txt) voir *tableau 1* résultat algorithme pour la mise en forme.

Ensuite on attribue le champ correspondant à la valeur récupérée (expression régulière) et on récupère également le nom du fichier (nom du fichier = nom du moteur de recherche).

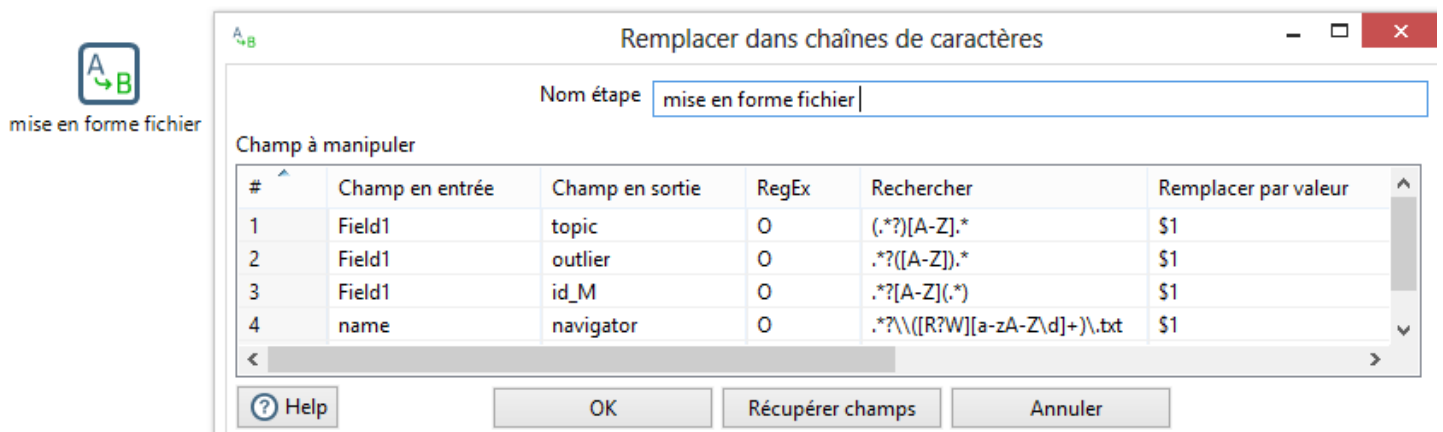
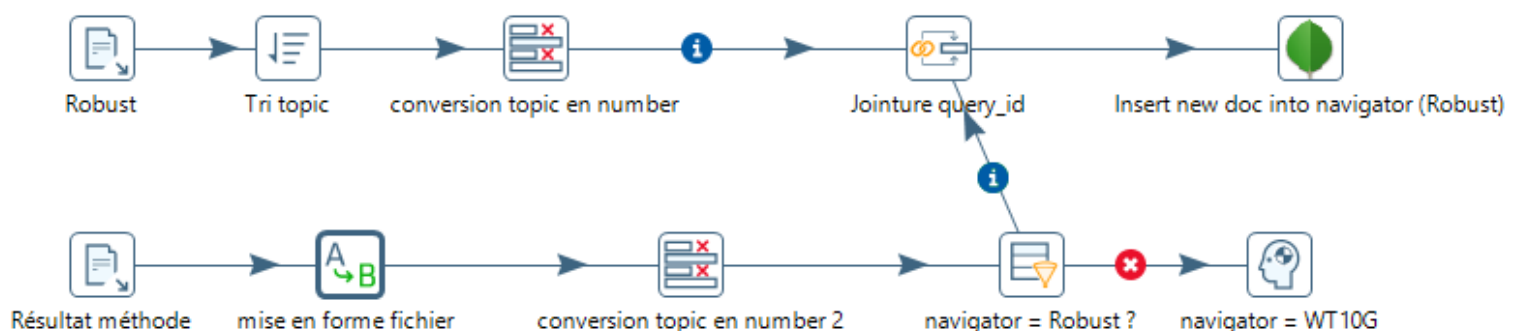


Figure 7: Attribution champs - valeurs et récupération du nom du moteur de recherche

Ensuite on réalise un filtre sur le nom du moteur de recherche pour mettre à jour les documents correspondant.

En parallèle, on récupère les documents mongoDB à mettre à jour, on effectue un tri sur l'identifiant et on réalise une jointure avec les résultats de la méthode implémentée. Une fois la jointure effectuée, on insère les nouveaux documents dans la collection « navigator ».

Voici un exemple de transformation via l'ETL pour les documents relatifs aux moteurs de recherche « robuste » :



Note : La transformation complète réalisée via l'ETL est disponible en annexe.

Une fois l'intégration des données effectuée, on réalise un job qui va nous permettre d'appeler notre transformation à l'arrivée de nouvelles données.

Note : Dans le cas de notre projet, l'implémentation des méthodes ne s'effectue pas par période régulière (par exemple une méthode implémentée par semaine). Celui-ci se lance donc manuellement pour le moment. En revanche, il serait intéressant par la suite, une fois la période d'implémentation d'une méthode connue, de le paramétrer et de l'appeler via un logiciel d'intégration continue comme Jenkins par exemple.

Voici le job réalisé :



Figure 8 : Job exécutant nos transformations

Commentaires :

L'étape « START » permet de lancer l'exécution du job. Comme expliqué précédemment, la planification (cycle de récupération de nos données) n'a pas été réalisée comme nous le montre la capture suivante :

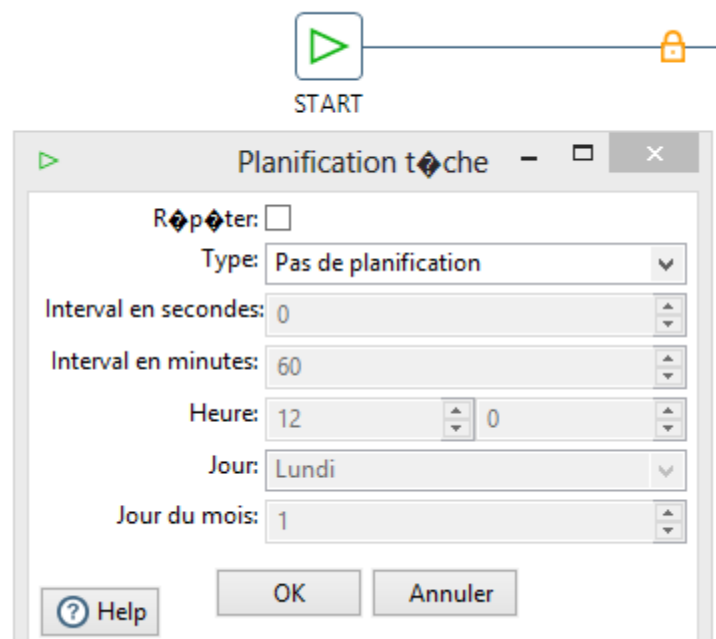


Figure 9 : Lancement du job sans planification

Après cette étape, on exécute la transformation explicitée précédemment :

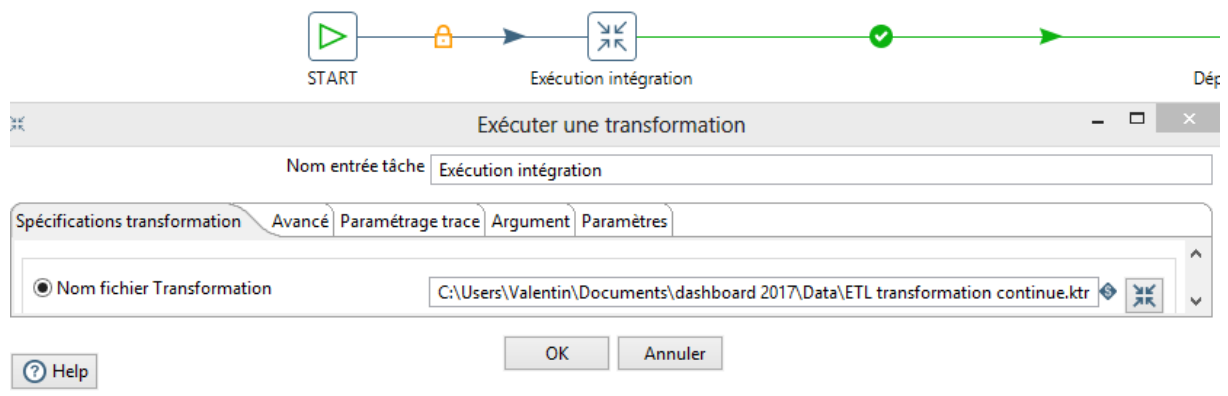


Figure 10 : Appel de la transformation réalisée

Une fois l'intégration exécutée et terminée, on déplace les fichiers .txt utilisée (fichier contenant les résultats de la nouvelle méthode implémentée) dans un autre répertoire. En effet, la transformation a été réalisée dans le but de traiter automatiquement ces fichiers .txt, il nous faut donc les déplacer une fois le traitement terminé. Cela nous permet d'éviter d'insérer des doublons dans notre collection et donc d'avoir des erreurs pour violation de contraintes.

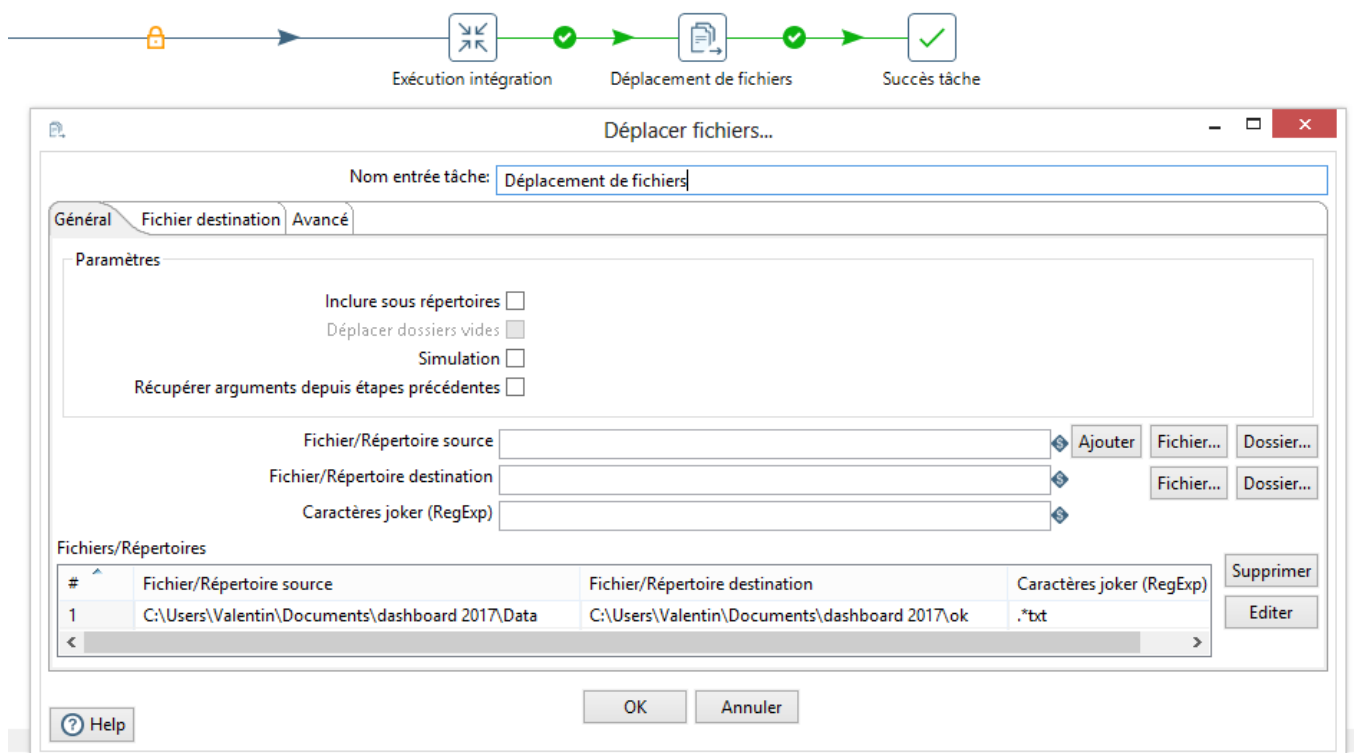


Figure 11 : Changement de répertoire des fichiers traités

Une fois le déplacement des fichiers effectué, l'exécution du job s'est réalisée avec succès.

8 Méthodologie

A partir de MongoDB, nous obtenons deux fichiers, chacun relatif à un moteur de recherche : Robust et WT10G. Ces fichiers sont au format JSON et ensuite converti au format CSV.

Une fois les données importées, nous remarquons qu'ils sont composés d'assez nombreuses valeurs manquantes. Celles-ci nous handicapent car elles compliquent l'implémentation correcte des méthodes de recherche d'outliers.

La première réflexion au sujet de nos données, est de trouver un moyen de contourner ce problème de valeurs manquantes. Bien qu'handicapant dans le cadre des données dont nous disposons, cela nous force à penser à ce problème, qui peut être rencontré dans d'autres contextes, et à rechercher des solutions pour les pallier. Lorsqu'on dispose de données, composées de valeurs manquantes, il est préférable de nettoyer nos données, afin d'utiliser un jeu de données sans valeurs manquantes, pour implémenter les méthodes de mise en lumière d'outliers.

8.1 Gestion des valeurs manquantes

Il est probable que la plupart des valeurs manquantes viennent des mêmes variables. Dans ce cas, plutôt que de supprimer de nombreux individus ayant des valeurs manquantes, il semble préférable de chercher à supprimer quelques variables.

Filtre de corrélation :

Il est fréquent que plusieurs variables soient fortement corrélées entre elles. Lorsque c'est le cas, on peut considérer l'information comme récurrente, et il est alors peu utile de les conserver toutes les deux.

Concernant nos données, c'est bien le cas : plusieurs de nos variables sont corrélées entre elles. En se basant sur la matrice de corrélation de Pearson, nous avons appliqué un filtre permettant de ne conserver qu'une variable sur deux lorsqu'elles sont très corrélées entre elles (seuil fixé à + ou - 0,75). Pour les données Robust, cela nous permet de réduire le nombre de variables de 221 à 112, et pour WT10G, nous passons de 250 à 169, ce qui est considérable.

Filtre de variance

Il peut arriver que certaines variables aient une faible variance. Cela revient à dire que celles-ci sont quasiment constante, il n'est donc pas pertinent de les conserver. En effet, nous souhaitons mettre en avant des outliers potentiels, il est donc inutile de conserver et d'étudier une variable relativement constante.

Au sujet de nos données, on remarque que certaines variables ont effectivement une faible variance. C'est le cas pour Robust, tout comme WT10G. L'application d'un filtre de variance supprimerait les variables ayant une variance inférieure à un seuil fixé. En fixant ce seuil de variance maximale à 2, nous passons, pour Robust, de 112 à 25 variables, et pour WT10G, de 169 à 21 variables.

Filtre de valeurs manquantes

Ayant besoin d'avoir des données sans valeurs manquantes, la solution la plus simple est alors d'utiliser un filtre qui supprime les individus pour lesquels on trouverait des valeurs manquantes.

Si on applique ce filtre sur nos données, après les filtres effectués sur les variables, cela ne supprime qu'un unique individu pour le jeu de données Robust, et 3 pour le jeu de données WT10G, ce qui est relativement faible.

Autres moyens sans suppression de données

Bien que l'application de ces filtres soient justifiés, il faut rester prudent lorsqu'il est question de supprimer de l'information, et ce malgré le bien fondé et la justification de ces suppressions. Si nous le souhaitons, nous pouvons trouver d'autres moyens de nous libérer de ces valeurs manquantes, sans pour autant supprimer des données. Nous pourrions également remplacer les valeurs manquantes, par des valeurs calculées, plutôt que de supprimer de l'information.

Pour ce faire il existe plusieurs méthodes, parmi lesquelles des méthodes de régressions, ou d'interpolation. Cependant, n'ayant pas trouvé de résultats bien plus probants en ayant essayé l'interpolation linéaire, nous avons fait le choix de se satisfaire de ces filtres. De plus, nous considérons que, tout comme il est intéressant de mettre en lumière des individus potentiellement aberrants au sein de jeux de données, il est intéressant de se pencher sur les valeurs manquantes et les moyens existants pour pallier à ce problème.

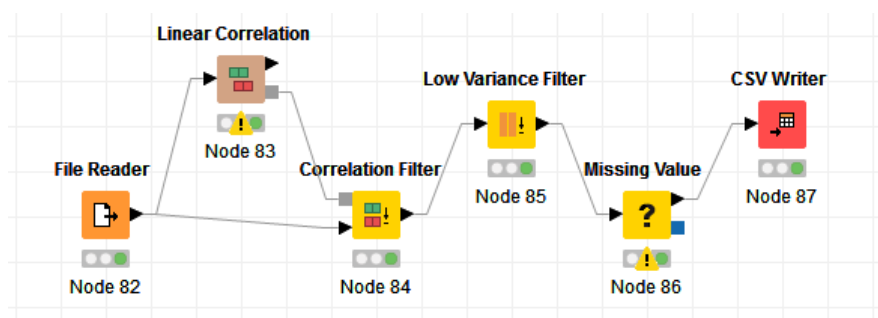


Figure 12 : Nœuds permettant l'application des filtres, dans le logiciel KNIME

8.2 ACP

Une fois que nos données sont débarrassées de toutes leurs valeurs manquantes, et de plus, allégées de plusieurs variables superflues, il est judicieux de réaliser une ACP sur nos données. Cela nous permettrait de synthétiser un maximum d'information en un minimum de variables, dans un souci de simplicité.

Avant de réaliser une ACP, il n'y a aucune raison de donner plus d'importance à une variable plutôt qu'à une autre. Il nous faut donc normaliser les données, avant d'appliquer l'ACP. Elle nous propose, pour le jeu de données Robust, de conserver 66 % d'information en 6 axes, et pour le jeu de données WT10G, de conserver 76 % d'information en 6 axes.

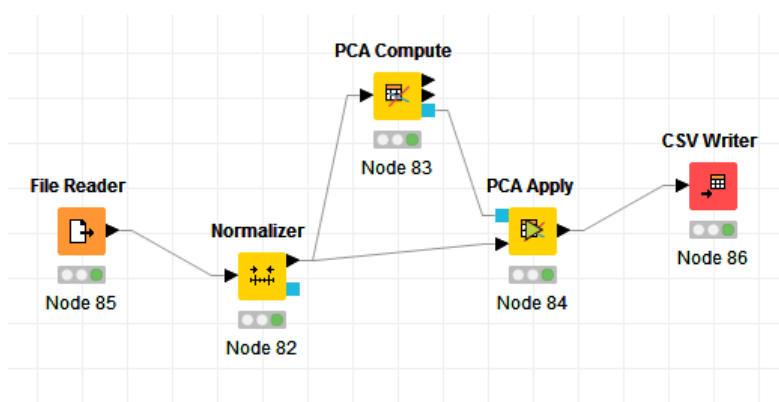


Figure 13 : Nœuds permettant l'application d'une ACP, dans le logiciel KNIME

8.3 DBSCAN

Une fois nos données épurées de leurs valeurs manquantes, et l'information restante synthétisée en quelques variables uniquement (6 pour chacun des jeux de données à notre disposition) nous pouvons passer à l'implémentation de la première méthode de détection d'outliers : DBSCAN.

Comme nous l'avons vu plus haut, DBSCAN est une méthode de classification, à ceci près qu'elle peut aussi considérer des individus comme étant des outliers. Pouvant être basées sur une mesure de distance, quelle qu'elle soit, nous avons remarqué, après plusieurs essais, que la distance de Mahalanobis était la plus appropriée, puisqu'elle considère une distance relative, contrairement à la distance Euclidienne par exemple. Nous calculons donc la matrice des distances de Mahalanobis, à partir desquelles nous nous appuyons pour l'implémentation de la méthode DBSCAN. A l'issue de cette méthode, nous représentons, à l'aide d'un graphique en 3 dimensions, le résultat issu de la méthode DBSCAN.

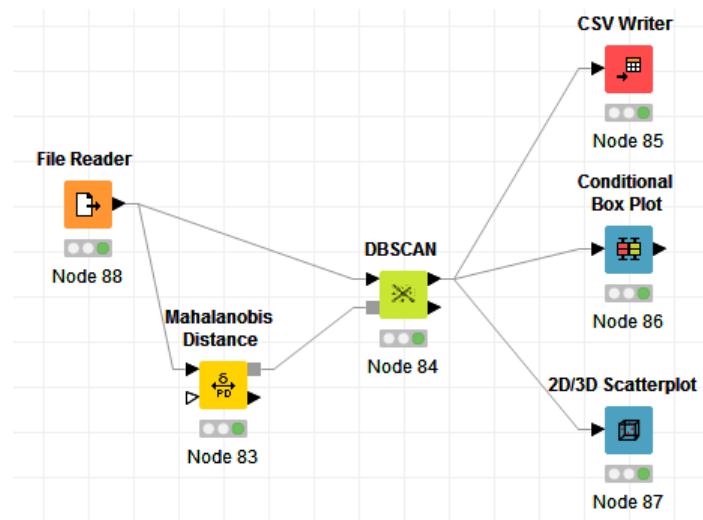


Figure 14 : Nœuds permettant l'application de DBSCAN, dans le logiciel KNIME

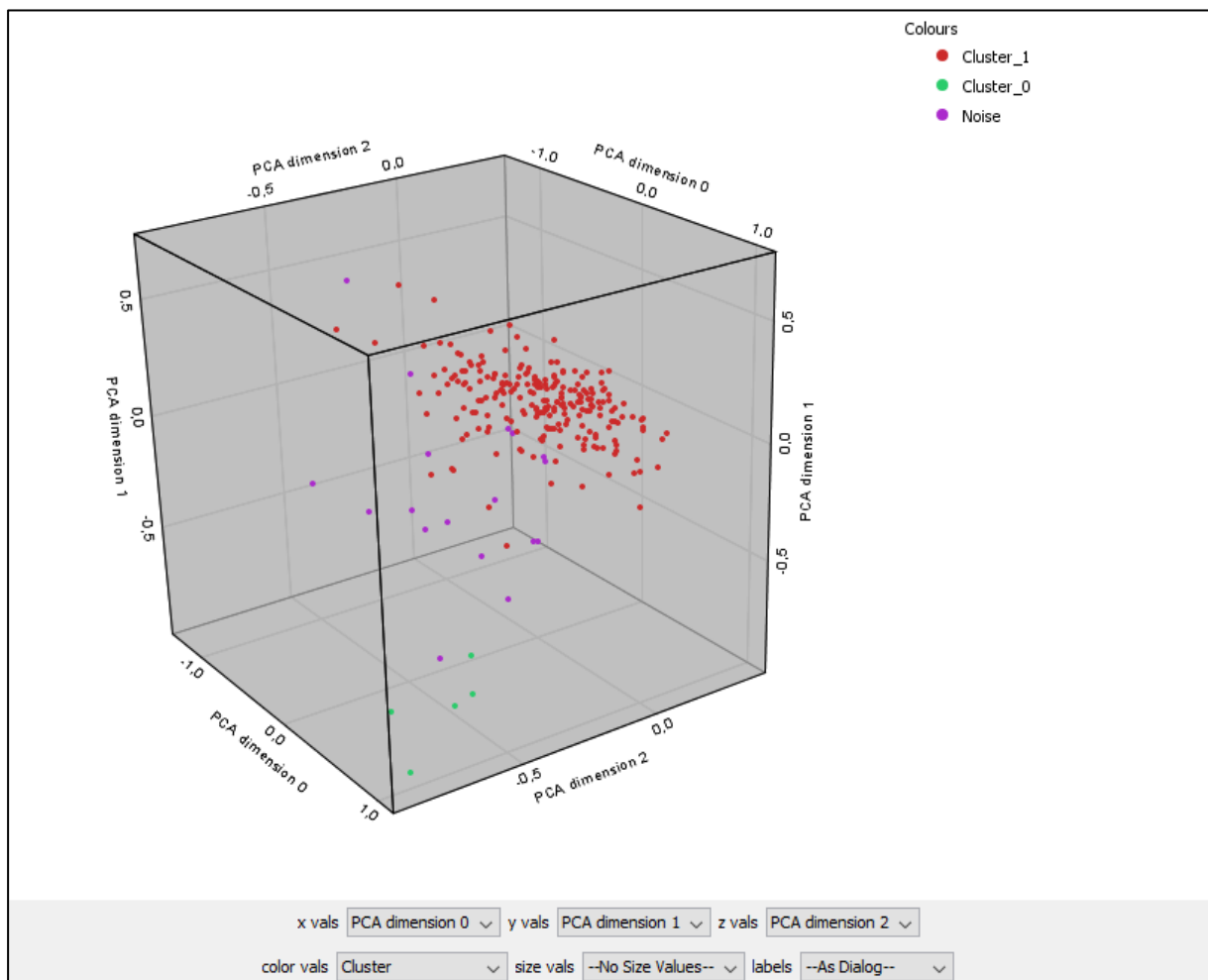


Figure 15 : Graphique en 3 dimensions, illustrant le résultat issu de la méthode DBSCAN sur Robust

On peut voir sur ce graphique que l'utilisation de cette méthode, que les individus non-aberrants sont regroupés essentiellement dans le cluster_1 (en rouge). Les individus aberrants (en violet) sont situés, pour une bonne partie, à une distance assez élevée des individus de ce cluster donc le résultat semble cohérent. En revanche, on pourrait se poser une question sur la pertinence du cluster_0 (en vert). En effet, ce cluster est composé de peu d'individus, situés à une distance assez élevée du cluster_1, et pourrait donc être composé essentiellement d'individus aberrants.

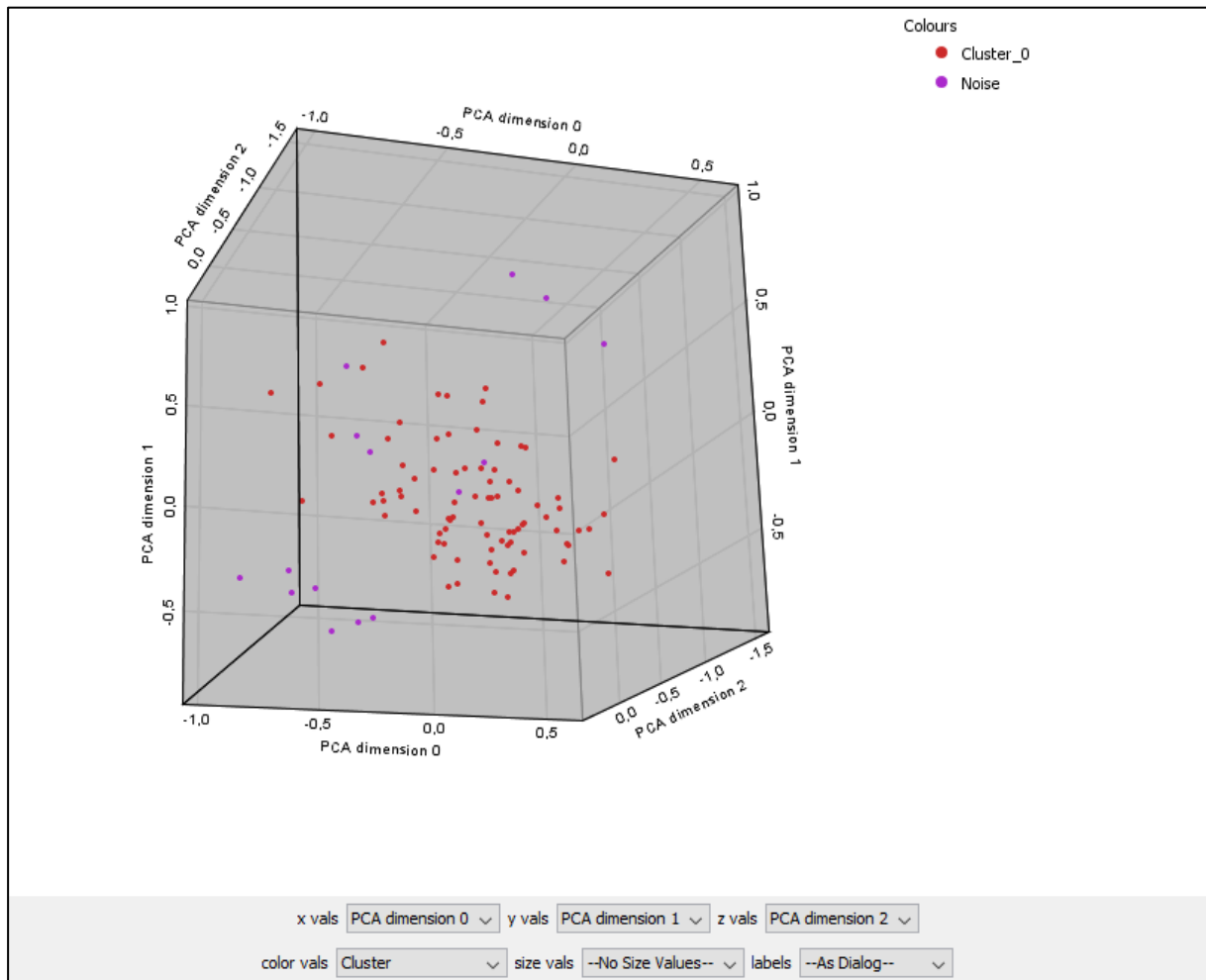


Figure 16 : Graphique en 3 dimensions, illustrant le résultat issu de la méthode DBSCAN sur WT10G

On peut voir sur ce graphique que l'utilisation de cette méthode nous a permis d'identifier de manière convenable (graphiquement) les individus aberrants des autres individus. En effet, on peut voir que les individus aberrants (en violet) sont situés, pour une bonne partie, à une distance assez élevée des autres individus (en rouge).

8.4 MCD (Robust Covariance)

Parallèlement à la méthode DBSCAN, nous avons implémenté la méthode Robust Covariance, aussi appelée MCD (Minimum Covariance Determinant).

Le choix de l'utilisation du logiciel R pour implémenter cette méthode s'explique par le fait qu'il existe plusieurs bibliothèques permettant de faciliter cette implémentation, comme par exemple DetMCD, que nous avons utilisé.

Tout d'abord, il faut savoir que cette méthode est supposée être appliquée sur des données suivant une distribution gaussienne. Avant de procéder à l'implémentation, il nous faut donc vérifier ce postulat, sur nos données.

Nous utiliserons une fonction permettant l'affichage, pour chaque variable en question (en l'occurrence, les axes issus de l'APC), d'un histogramme, permettant d'avoir un aperçu direct sur la distribution de la variable, et d'un QQ-plot, permettant de tester l'hypothèse de normalité.

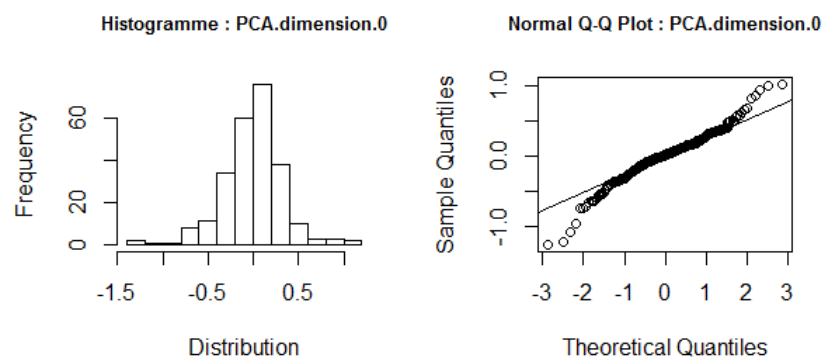


Figure 17 : Graphiques permettant de vérifier la normalité de la première variable de Robust

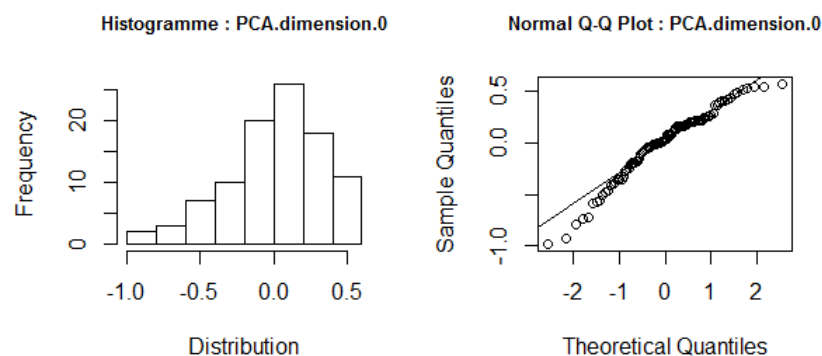


Figure 18 : Graphiques permettant de vérifier la normalité de la première variable de WT10G

Les autres graphiques correspondant à la vérification de la normalité des variables issues de l'ACP sont en Annexe.

Nous remarquons sur ces graphiques, ainsi que sur ceux se trouvant en annexe, que même si elle n'est pas toujours parfaite, l'hypothèse de validité peut être validée pour nos données, qu'il s'agisse de Robust ou de WT10G.

Une fois ces hypothèses de normalités validées, nous pouvons passer à l'implémentation de la méthode MCD.

On utilise alors la fonction DetMCD, issue du package du même nom de R. Avant de pouvoir l'exécuter, il nous faut renseigner un paramètre correspondant à la proportion d'inliers nous souhaitons conserver (et par conséquent la proportion d'outliers). Nous fixons ce paramètre à 0.90, ce qui signifie que 10 % de nos individus seront considérés comme étant des outliers.

Cette fonction nous fournit de nombreux résultats, dont un vecteur composé de valeurs binaires, nous informe si les individus sont qualifiés, ou non, d'outliers. Nous pouvons également créer un graphique, représentant les individus en fonction de la distance robuste, et la distance de Mahalanobis, qui nous permet de visualiser ces outliers.

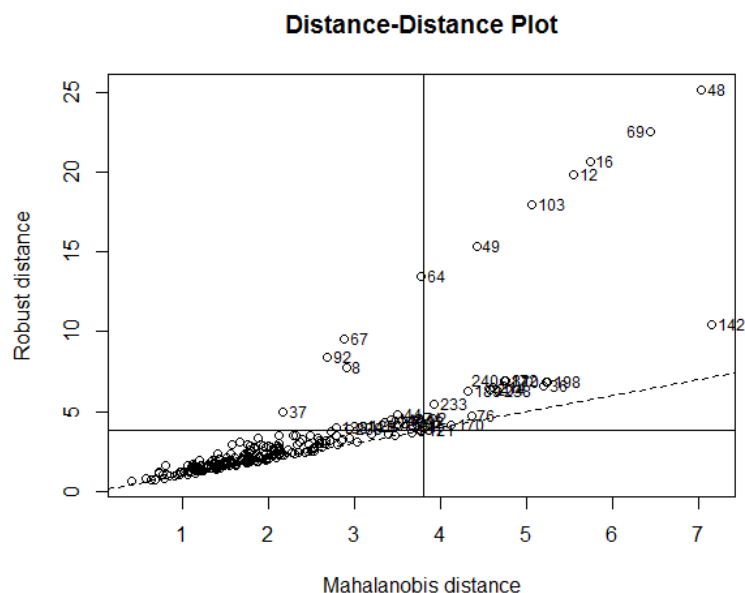


Figure 19 : D-D Plot permettant de visualiser le résultat de MCD à partir de Robust

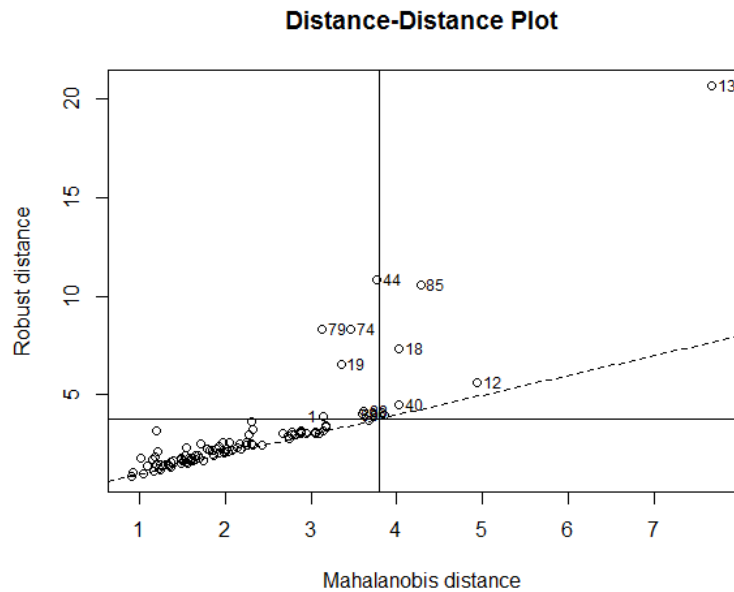


Figure 20 : D-D Plot permettant de visualiser le résultat de MCD à partir de WT10G

Dans ce genre de graphique, les points ont tendance à s'inscrire au niveau de la droite qui passe par l'origine. Les points qui sont loin de cette ligne sont considérés comme étant des outliers, tout comme ceux qui sont à son extrémité.

8.5 SVM One Class

A présent, nous disposons des résultats de deux méthodes différentes permettant de mettre en lumière des outliers. Nous souhaitons également implémenter la méthode SVM one-class. Comme nous l'avons vu dans la partie détaillant les méthodes, pour implémenter cette méthode nous devons disposer d'un échantillon d'apprentissage, composé uniquement d'inliers, afin que tous les individus étant outlier soit remarqué comme étant différent des individus de l'échantillon d'apprentissage. Ne disposant pas d'échantillon annoté, nous devons faire autrement.

Nous allons donc implémenter cette méthode à deux reprises : l'une, en se basant sur les résultats obtenus grâce à DBSCAN, comme étant une vérité (étant donné que nous ne sommes pas en mesure de vérifier le bon fondé de ce résultat), et l'autre en se basant sur les résultats obtenus grâce à Robust Covariance (MCD), comme étant une vérité.

Pour cela, nous avons réalisé plusieurs fonctions à l'aide de R, utilisant les packages e1071, et caret. Ces packages nous permettent d'implémenter la méthode SVM plus facilement, et d'afficher la matrice de confusion permettant d'évaluer le modèle obtenu.

Tout d'abord, nous avons souhaité utiliser un échantillon d'apprentissage, avant de tester le modèle obtenu sur les données restantes : notre échantillon test. L'échantillon étant tiré aléatoirement, nous avons donc bien sûr un résultat différent pour chaque exécution du code.

Nous pouvons par exemple obtenir les matrices de confusion suivantes :

- Jeu de données Robust, basé sur DBSCAN :

		Reference	
Predicted		FALSE	TRUE
FALSE	18	8	
TRUE	0	48	

- Jeu de données WT10G, basé sur DBSCAN :

		Reference	
Predicted		FALSE	TRUE
FALSE		15	7
TRUE		0	13

- Jeu de données Robust, basé sur MCD :

		Reference	
Predicted		FALSE	TRUE
FALSE		18	11
TRUE		0	45

- Jeu de données WT10G, basé sur MCD :

		Reference	
Predicted		FALSE	TRUE
FALSE		15	8
TRUE		0	12

Nous remarquons que ces matrices de confusions sont assez similaires que l'on se base sur DBSCAN, ou sur la Robust Covariance (MCD). Le modèle fonctionne assez bien, et les précisions des modèles sont globalement satisfaisantes.

Ne souhaitant pas établir un modèle prédictif, nous utilisons ensuite une méthode de validation croisée (CF. définitions) permettant d'obtenir un résultat (outlier ou non) pour chaque individu, et non pas uniquement pour les individus faisant parti de l'échantillon de test.

Tout d'abord partis sur la méthode ten-fold, consistant à prendre 10 échantillons, et à alterner l'échantillon test à chaque itération, nous nous sommes rendu compte que cela n'était pas aussi simple dans notre cas. Nous avons besoin d'individus inliers pour composer notre échantillon d'apprentissage, pour implémenter la méthode SVM one-class, ce qui n'est pas possible si on échantillonne en 10 parties tous les individus.

Après réflexion, nous avons pris la décision de réaliser une variante de la validation croisée ten-fold. Cette variante consiste à prendre au total 20, et non plus 10 échantillons de nos individus. Cependant, parmi ces 20 échantillons, 10 ne comprennent que des individus inliers, et les 10 autres que des individus outliers.

Une fois l'échantillonnage effectué, il nous faut composer un échantillon test (composé d'un échantillon parmi les inliers, et un autre parmi les outliers), ainsi qu'un échantillon d'apprentissage (composé de tous les autres individus étant inliers). Cette méthode nous permet de prendre en compte chaque individu une unique fois dans un échantillon test, sans pour autant utiliser d'individus outliers dans l'échantillon d'apprentissage.

8.6 Arbre de décision

Une fois les méthodes implémentées, il est intéressant de voir quels sont les critères qui font qu'un individu soit qualifié d'outlier. Pour ce faire, nous allons utiliser des arbres de décision.

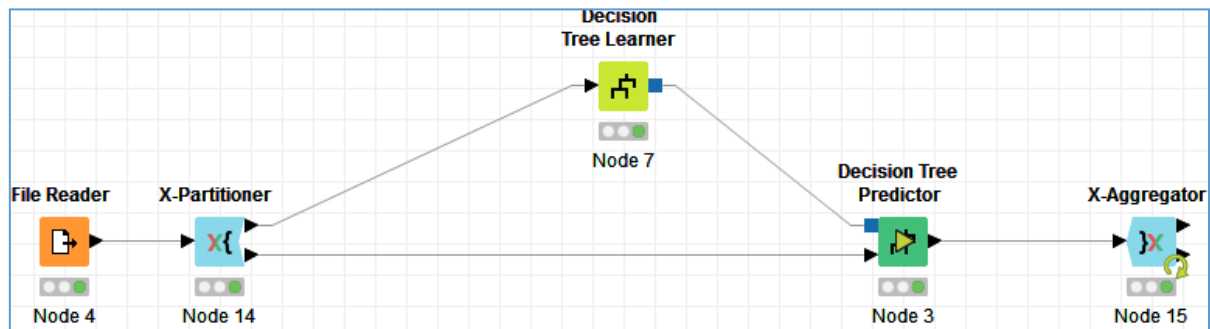


Figure 21 : Méthodologie arbre de décision Robust_DBSCAN

L'étape « File Reader » contient les données des variables retenues via l'ACP réalisée durant l'implémentation des méthodes.

Ensuite, nous avons partitionné nos données de la manière suivante : 90% pour le jeu d'apprentissage et 10% pour le jeu de test.

Puis, nous injectons notre jeu d'apprentissage à l'étape « Decision Tree Learner », étape qui nous permet d'apprendre un modèle. Ensuite, nous testons notre modèle via l'étape « Decision Tree Predictor » qui récupère le résultat de l'étape « Decision Tree Learner » et le jeu de données test.

Notons que pour obtenir l'arbre de décision final, nous avons réalisé une cross-validation (« Etape X-Partitioner »), de la manière suivante : Jeu de données divisé en 9 jeux d'apprentissage et 1 jeu de test (90% apprentissage – 10% test). Chaque jeu de données sert de test et le résultat moyen obtenu sur l'arbre de décision via les jeux de tests constitue le résultat final (arbre final).

9 Tableau de bord

Les résultats ont été obtenus en travaillant sur les 21 variables suivantes, conservées via une ACP et un filtre de variance :

Variance meronyms
BM25_mean
Variance sister-terms
Standard deviationhyponyms
WMODEL:SingleFieldModel(Tf.0)_Q3
WMODEL:SingleFieldModel(DirichletLM.0)_median
WMODEL:SingleFieldModel(Tf.1)_numdocs
WMODEL:SingleFieldModel(DirichletLM.1)_max
Total holonyms
WMODEL:SingleFieldModel(TF_IDF.0)_max
T_avg_idf
Medianhyponyms
AVGSIZE
Minimum hyponyms
AVGSYNSETS
WMODEL:SingleFieldModel(Tf.0)_std
Minimum sister-terms
Q1:SingleFieldModel(DI.1)_std2
Q1:SingleFieldModel(DI.0)_Q3
Medianmeronyms
Minimum synonyms
Q1:SingleFieldModel(DI.0)_min
Averagesynonyms
Q1:SingleFieldModel(DI.0)_std

Voici le résultat obtenu lors de l'exécution de l'algorithme DBSCAN sur nos données :

Navigator	outliers : N	outliers : Y
Robust	92,8%	7,2%
WT10G	84,5%	15,5%

Tableau 2: Résultat DBSCAN par moteur de recherche

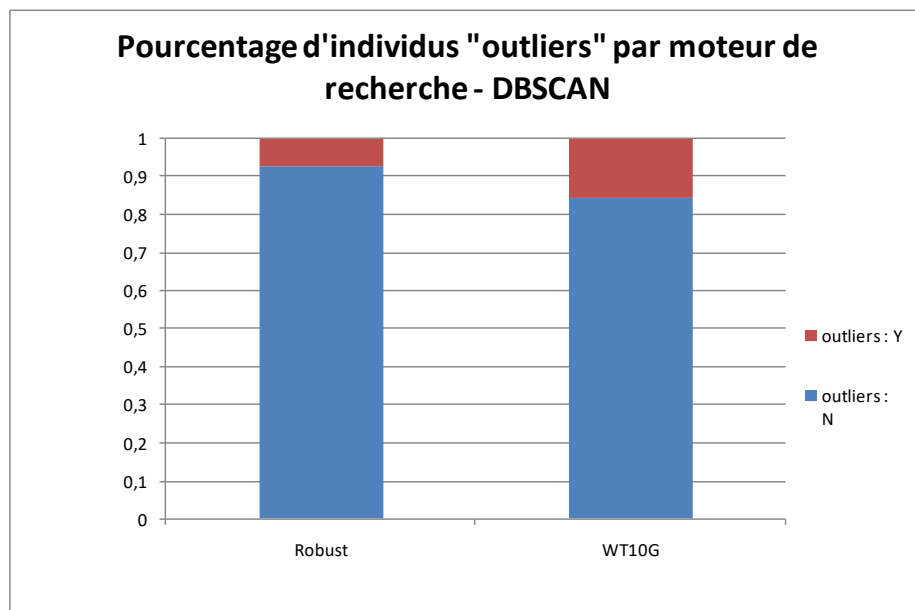


Figure 22 : Visualisation des individus outliers / moteur de recherche

Nous pouvons ainsi voir via ce graphique que la méthode DBSCAN détecte environ deux fois plus d'individus outliers sur le moteur de recherche WT10G que sur le moteur de recherche Robust.

Voici le résultat obtenu lors de l'exécution de l'algorithme Robust Covariance sur nos données :

Navigator	outliers : N	outliers : Y
Robust	91,2%	8,8%
WT10G	91,8%	8,2%

Tableau 3 : Résultat Robust Covariance par moteur de recherche

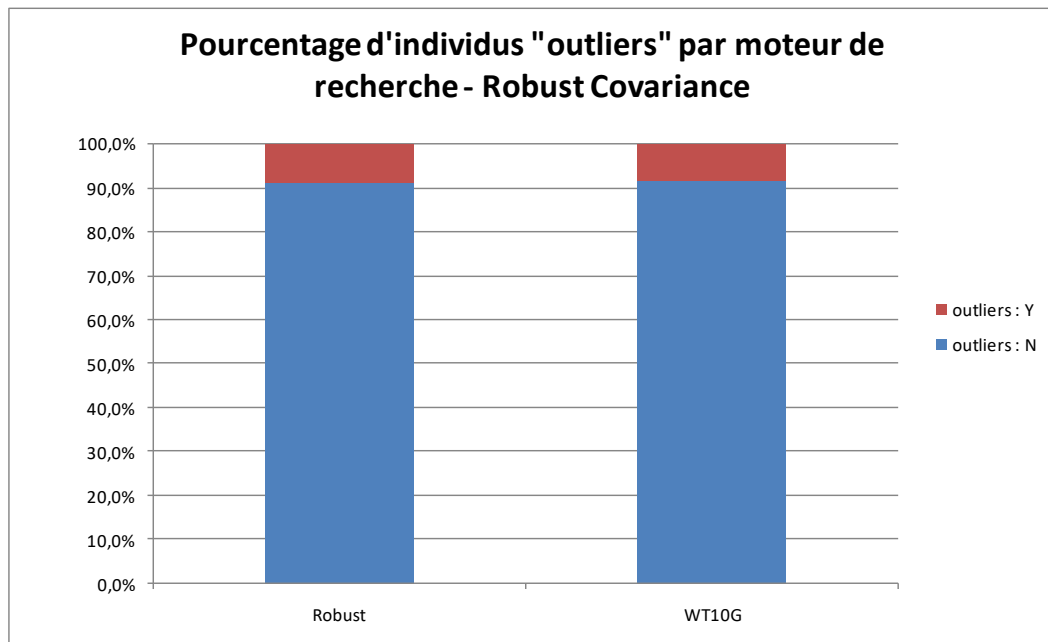


Figure 23 : Visualisation des individus outliers / moteur de recherche

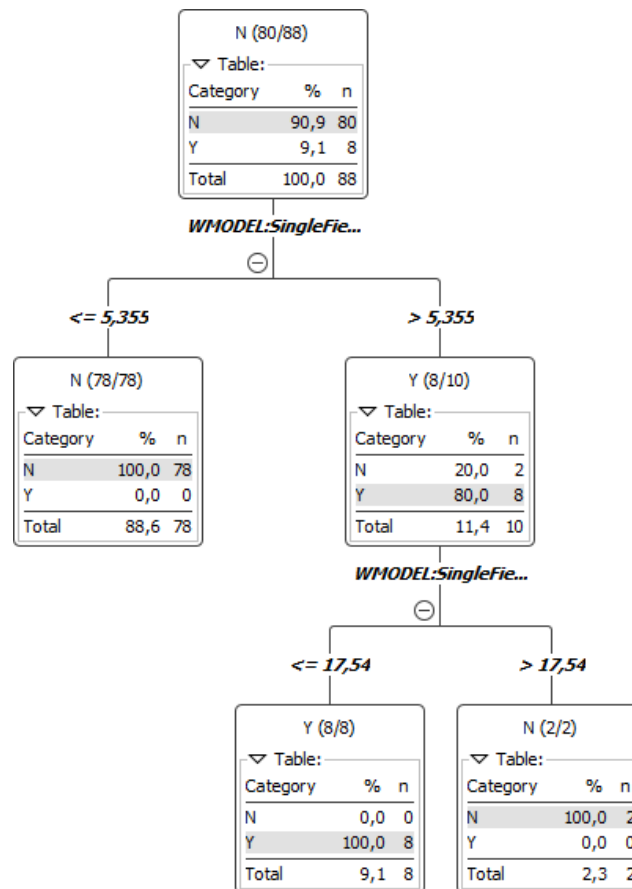
Nous pouvons ainsi voir via ce graphique que la méthode Robust Covariance détecte environ le même nombre d'individu, à savoir respectivement 9% et 8% d'individus « outliers » sur le moteur de recherche Robust et WT10G.

9.1 Description des outliers

Nous allons nous intéresser maintenant à la problématique suivante : Quels sont les critères retenus par la méthode (sur chaque moteur de recherche) pour qualifier un individu d'outliers ou non ?

Nous venons de voir la méthodologie mise en place dans le cadre des arbres de décision. Nous allons à présent voir les résultats de ces arbres de décision.

Robust Covariance : Fichier de données WT10G



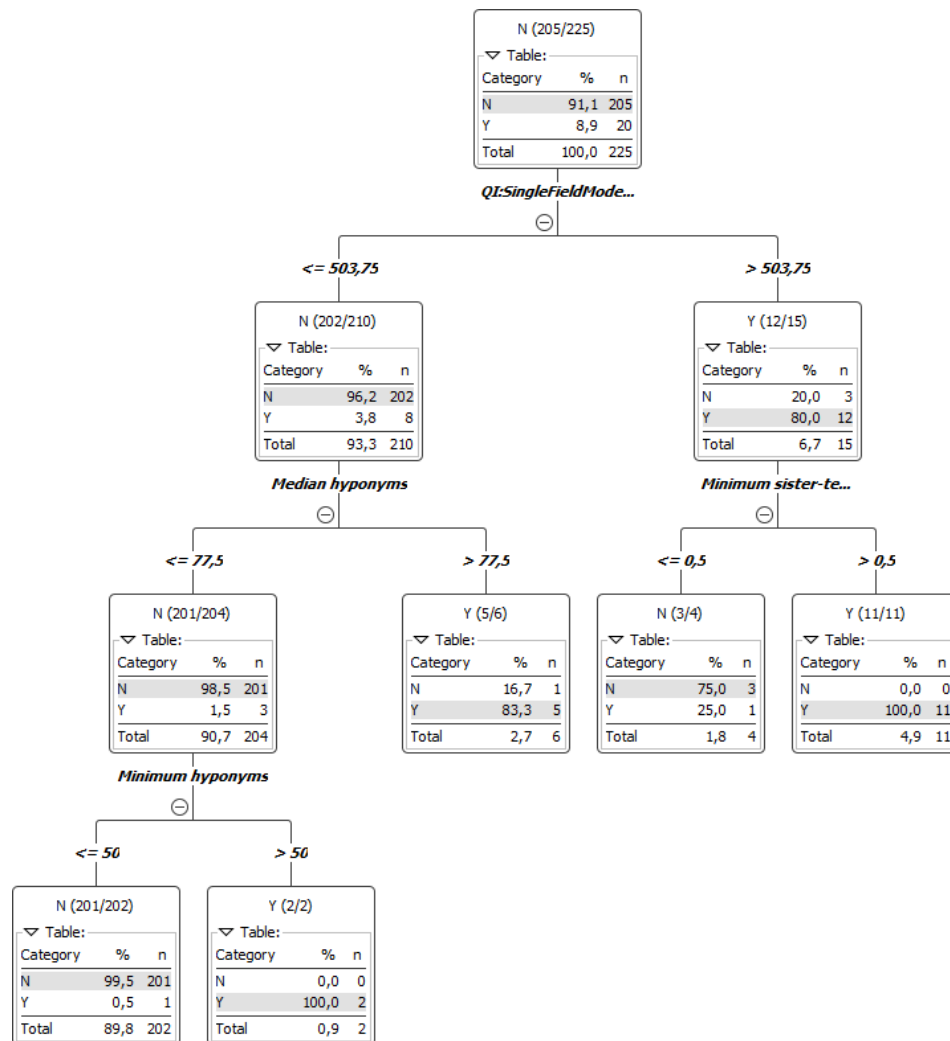
Suite à l'implémentation de la méthode Robust Covariance sur notre jeu de données, nous obtenons l'arbre de décision ci-dessus.

Après l'application de la première variable de découpe « WMODEL:SingleFieldModel(BM25,1)_std », pour un seuil strictement supérieur à 5.355, les individus aberrants sont séparés des autres individus (hormis 2). Les individus aberrants sont ensuite parfaitement séparés des autres individus en appliquant un seuil de 17.54 à la variable « WMODEL:SingleFieldModel(TF_IDF,1)_max ».

Le modèle obtenu permet donc d'expliquer un « outlier » selon 2 critères, à savoir les 2 seuils explicités sur les variables « WMODEL:SingleFieldModel(BM25,1)_std » et « WMODEL:SingleFieldModel(TF_IDF,1)_max ».

Note : En vue du nombre de données assez faible pour réaliser cet arbre de décision, nous avons choisi de ne pas l'élaguer en supprimant la dernière branche. De plus, en vue du faible nombre de données aberrantes à expliquer, il n'était pas pertinent de la supprimer.

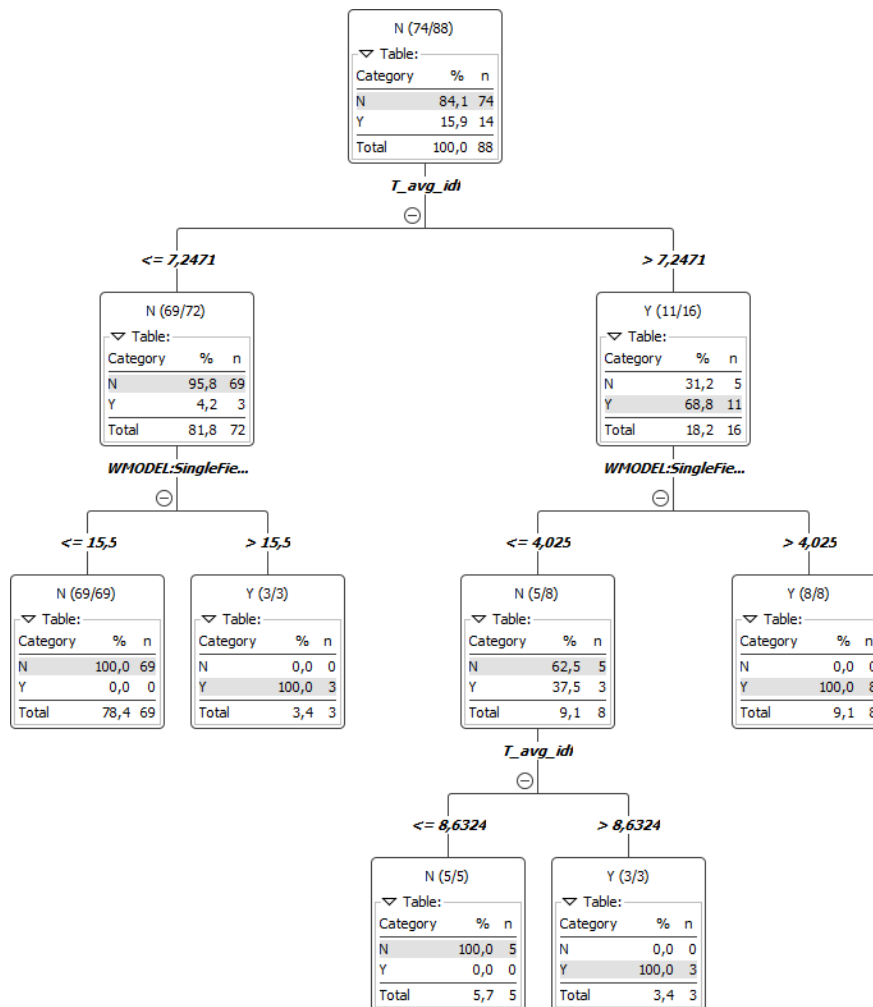
Robust Covariance : Fichier de données Robust



Pour le jeu de données « Robust », nous constatons que 2 critères (seuil > 503.75 pour la première variable et > 0.5 pour la variable « Minimum sister-terms ») nous permettent d'identifier parfaitement 60% de nos individus aberrants (11/20). En partant de l'autre seuil de la première variable (<= 503.75) et en appliquant un seuil > 77.5 pour la variable « MedianHyponyms », on peut identifier de manière assez précise (83.3%), 20% de nos individus pertinents, non identifié via le premier découpage décrit.

Cet arbre nous permet donc d'expliquer pertinemment 80% de nos individus aberrants, en utilisant 4 seuils sur nos variables explicatives.

DBSCAN : Fichier de données WT10G



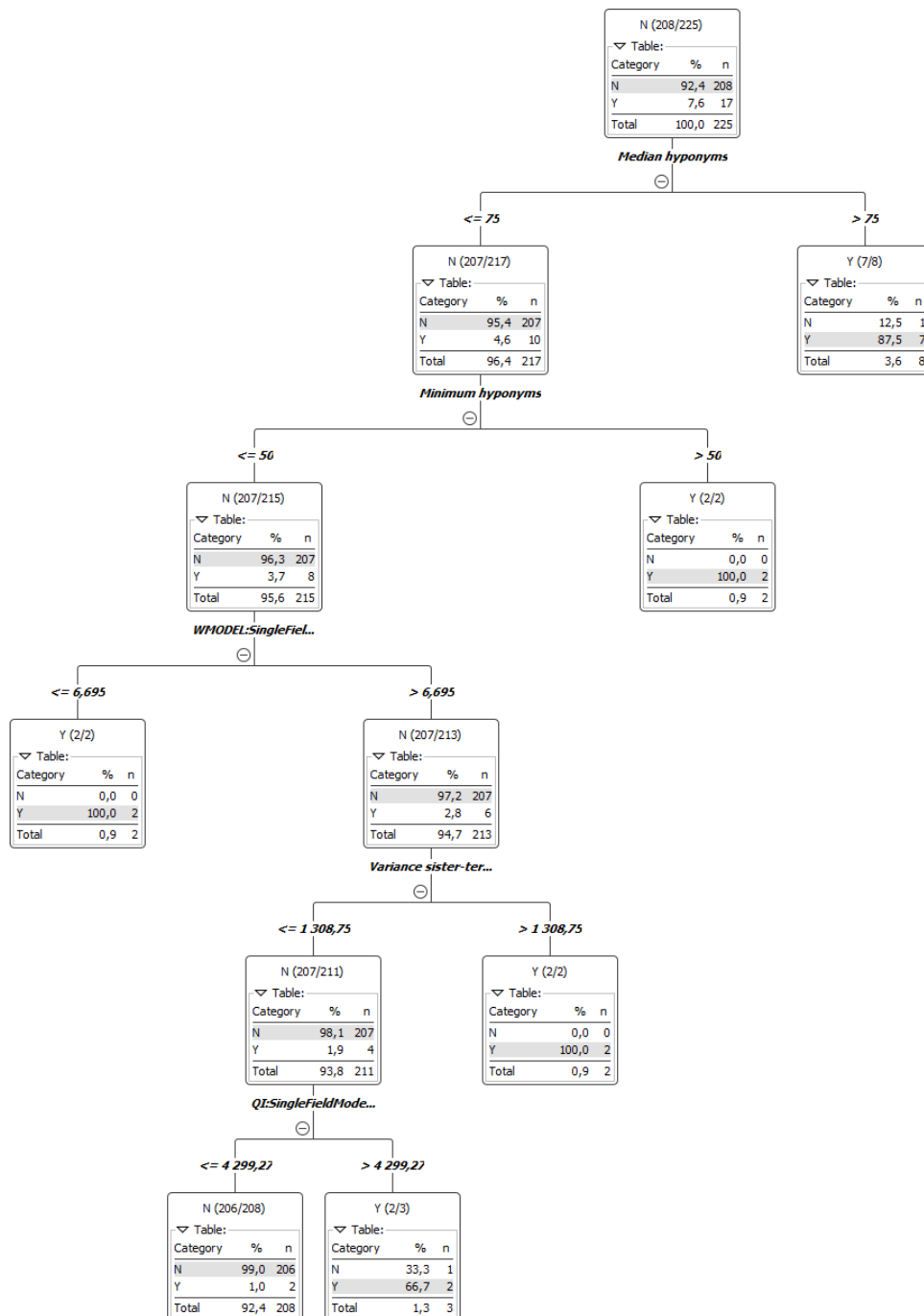
Pour le jeu de données « WT10G », on peut voir qu'une découpe à un seuil > 7.25 pour la variable « T_avg_idf » suivi d'une découpe > 4 pour la variable « $WMODEL:SingleFieldModel(Tf,1)_median$ » nous permet d'identifier pertinemment plus de 60% de nos individus aberrants (8/14).

Un seuil ≤ 7.25 suivi d'un seuil > 15.5 pour la variable « $WMODEL:SingleFieldModel(Tf,1)_std$ » nous permet d'identifier avec précision environ 20% de nos individus aberrants (3/14).

Nous sommes donc en mesure d'identifier assez simplement et pertinemment environ 80% de nos individus aberrants selon des critères assez simples.

Note : En vue du faible nombre d'individus restants pour le critère « T_avg_idf » ≤ 8.63 ou > 8.63 , nous aurions pu élaguer cette branche.

DBSCAN : Fichier de données Robust



Pour le jeu de données « Robust », on peut voir que l'arbre de décision est moins pertinent pour expliquer nos individus aberrants que les 3 autres. En effet, l'arbre est plus complexe (plus profond) mais permet tout de même d'expliquer assez pertinemment, en s'arrêtant au 3^{ème} niveau de l'arbre, environ 65% de nos individus aberrants.

9.2 Comparaison des résultats de chaque méthode

Les graphiques suivants illustrent quels individus ont été détectés comme étant des outliers, par moteur de recherche, et selon la méthode de détection utilisée :

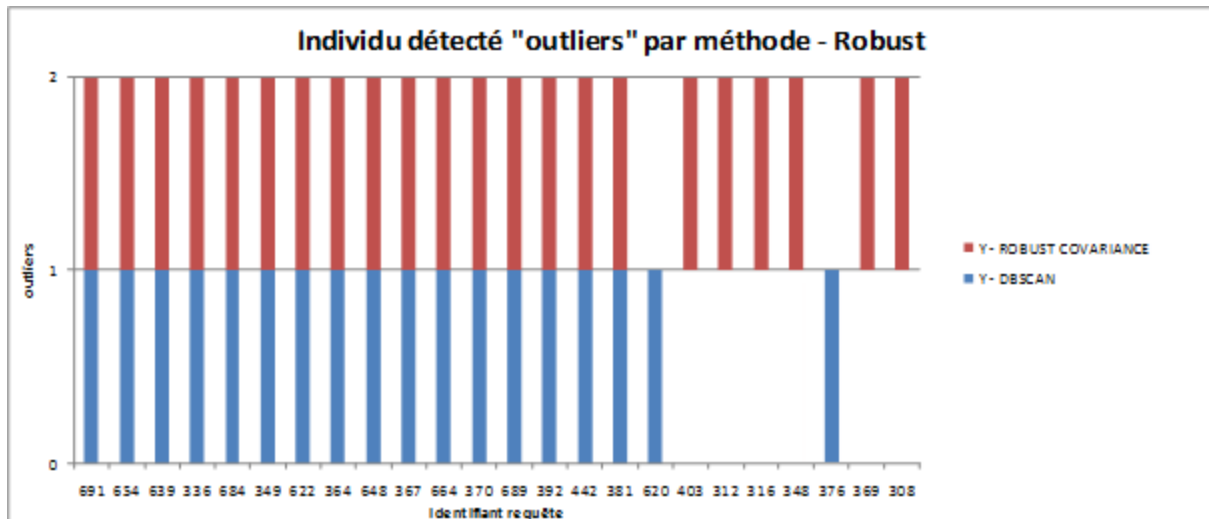


Figure 24 : Outliers détectés par méthode pour le moteur de recherche Robust

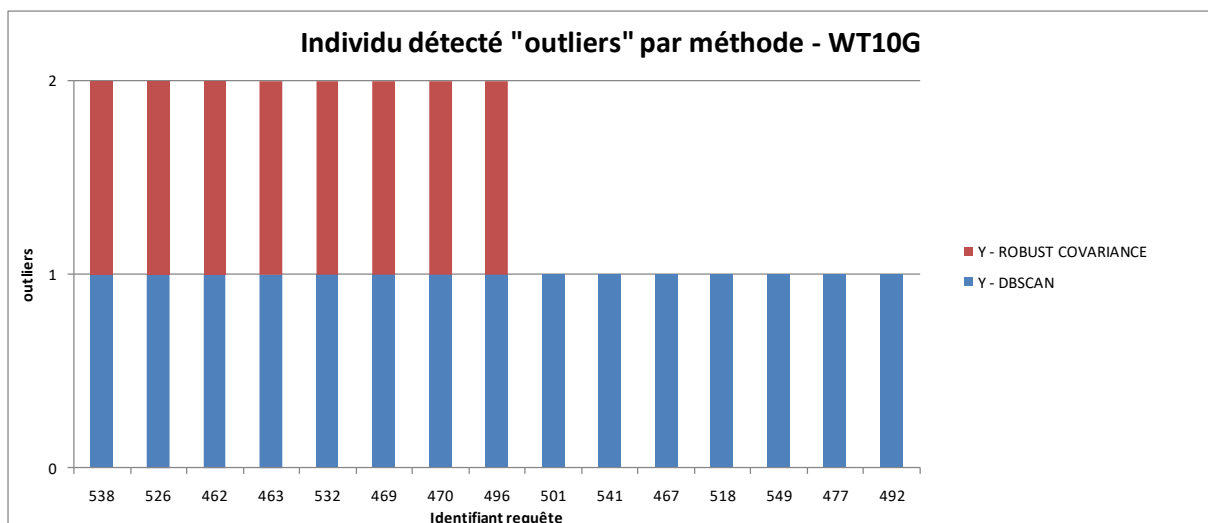


Figure 25 : Outliers détectés par méthode pour le moteur de recherche WT10G

On peut ainsi voir que malgré l'utilisation de méthodes différentes, celles-ci détectent des individus « outliers » communs. Les autres individus détectés comme outlier par une méthode ou par l'autre, dépendent des critères appliqués par la méthode sur les données. Ces critères diffèrent d'une étude de cas à une autre, comme l'illustre les arbres de décisions expliqués précédemment.

Pour cela, nous avons implémenté via le logiciel KNIME un arbre de décision par méthode expliquant les critères de détection de ces données. Voici la méthodologie utilisée :

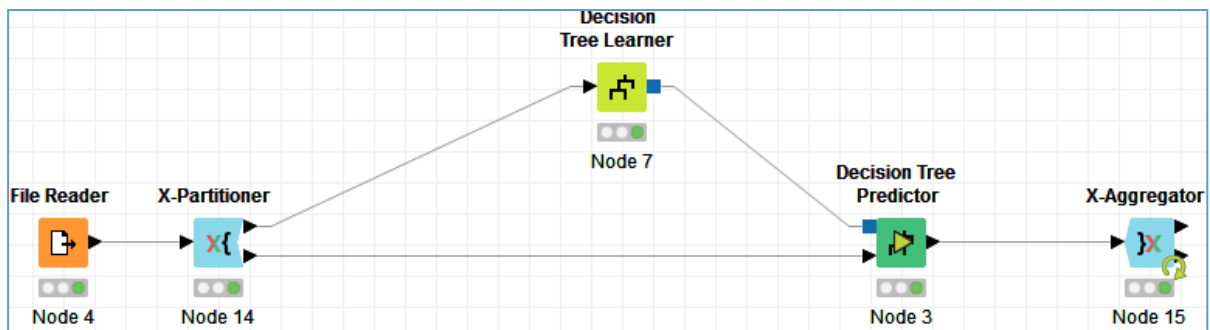


Figure 26 : Méthodologie arbre de décision Robust_DBSCAN

Note : L'ordre des méthodes illustrées ci-dessus a également été réalisée sur le moteur de recherche WT10G.

Commentaires :

L'étape « File Reader » contient les données des variables retenues via l'ACP réalisée durant l'implémentation des méthodes.

Ensuite, nous avons partitionné nos données de la manière suivante : 90% pour le jeu d'apprentissage et 10% pour le jeu de test.

Ensuite, nous injectons notre jeu d'apprentissage à l'étape « Decision Tree Learner », étape qui nous permet d'apprendre un modèle. Ensuite, nous testons notre modèle via l'étape « Decision Tree Predictor » qui récupère le résultat de l'étape « Decision Tree Learner » et le jeu de données test.

Notons que pour obtenir l'arbre de décision final, nous avons réalisé une cross-validation (« Etape X-Partitioner »), de la manière suivante : Jeu de données divisé en 9 jeux d'apprentissage et 1 jeu de test (90% apprentissage – 10% test). Chaque jeu de données sert de test et le résultat moyen obtenu sur l'arbre de décision via les jeux de tests constitue le résultat final (arbre final).

10 Démarche et assurance qualité

Revue 1 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- BRIENS Maxime(Fournisseur)
- BAHSOUN Wahiba (client)
- MOTHE Josiane (client)

Date : 01/02/2017

Lieu : Université Paul Sabatier, IRIT

Sujet:

Durant cette première réunion, nous avons rencontré Mme Josiane MOTHE, qui nous a proposé ce sujet sur la détection des outliers. Nous avons profité de cette réunion pour demander plus de détails concernant l'objectif final de notre travail, ainsi que sur la façon de procéder, afin de remplir les objectifs initiaux, tout en rentrant dans le cadre du projet tableau de bord.

L'idée générale de ce sujet est donc de réaliser un état des méthodes existantes de détection d'outliers, et de leur cadre d'application, ainsi que des domaines d'application de ces méthodes.

Concernant la méthode à suivre, voici comment nous allons procéder afin d'effectuer ces recherches :

1. Tout d'abord, nous devons savoir de quoi nous allons parler : il nous faut nous familiariser avec la problématique, en comprendre ses enjeux.
2. Nous utiliserons la méthode de questionnement quintilien (QQOQCCP), afin de faire le tour de la question, et s'assurer d'un certain niveau de maîtrise du sujet, avant de rentrer dans le vif des recherches. Cette étape devrait également nous permettre de trouver les premières informations sur les méthodes existantes, à partir de banques de données telles que Google Scholar par exemple.
3. Définir une problématique, afin de délimiter le champ de notre étude. Pour ce faire nous pourrions réaliser une carte mentale du travail à réaliser, des recherches à effectuer.

Dans un premier temps, nous avons réfléchi à certaines méthodes dont nous avons connaissance, qui peuvent permettre de mettre en lumière des outliers. Cela consisterait un point d'entrée dans le vif du sujet. Nous avons par exemple évoqué les méthodes d'analyse factorielle des correspondances (AFC), la classification, certains tests d'hypothèses, certaines méthodes de régression.

Sur le point de la gestion de projet, nous avons convenu de se réunir chaque semaine, avec Mme WahibaBAHSOUN et Mme Josiane MOTHE.

Afin de faire coïncider notre sujet au cadre général du projet tableau de bord, nous allons devoir appliquer les méthodes étudiées. Pour ce faire, nous devons disposer de données : nous devrions avoir accès à des données liées à des moteurs de recherche d'information, avec des requêtes, des caractéristiques, et des performances systèmes. Idéalement, nous devrions utiliser plusieurs jeux de données, ce qui nous permettrait d'avoir un aperçu plus concret sur l'utilisation des différentes méthodes, dans différents domaines d'application. Cependant, nous verrons ultérieurement si nous avons le temps de compléter notre étude avec davantage d'applications sur différents jeux de données. Nous préférons tout d'abord nous concentrer sur l'état des lieux des méthodes existantes, en illustrant nos propos sur un jeu de données, avant d'aller plus loin avec plus de jeux de données, et une mise en relief avec les domaines d'application.

La présentation finale aura lieu le 17 mars, et c'est également à cette date que nous devons rendre le rapport. Afin de ne pas prendre de retard, nous allons donc réaliser un GANTT, détaillant le temps prévu sur chaque étape de notre projet.

Revue 2 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime(Fournisseur)
- BAHSOUN Wahiba (client)

Date : 03/02/2017

Lieu : Université Paul Sabatier, Bât U3

Sujet abordé :

Le but de cette réunion avec Mme BAHSOUN était de vérifier, et de valider notre première ébauche de plan de travail et notre compréhension en générale.

Suite à cet entretien nous partons sur le plan suivant :

1. Approche descriptive
 - a. Enumération des méthodes relevées sur internet (domaines d'application, cas d'utilisation, coût...)
 - b. Zoom sur quelques méthodes, type AFC, ANOVA en les utilisant sur des jeux de données
2. Approche apprentissage machine
 - a. Enumération des méthodes relevées sur internet (domaines d'application, cas d'utilisation, coût...)
 - b. Zoom sur quelques méthodes, type SVM, ITREE sur des jeux de données + évaluation de celles-ci
3. Visualisation des résultats obtenus via un ou plusieurs tableaux de bords (1 tableau par méthode illustrée ?)

Revue 3 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime (Fournisseur)
- BAHSOUN Wahiba (client)

Date : 10/02/2017

Lieu : Université Paul Sabatier, Bât U3

Sujet abordé :

Le but de cette réunion avec Mme BAHSOUN était de faire le point avant les vacances, sur l'avancement du projet, ainsi que sur les premiers documents que nous lui avons transmis (planning prévisionnel et plan).

Les données proposées par Mme MOTHE étant confidentielles, nous ne sommes pas libres de faire ce qu'on en souhaite, nous devons faire attention, notamment à la façon dont nous allons les stocker.

Sur notre diagramme de Gantt, nous avons prévu une étape “méthodes”. Cette dénomination n’étant pas suffisamment claire, nous avons décidé de la redéfinir en deux parties, à savoir une première partie recherche, au sujet des méthodes statistiques, contenant l’affichage de données, et une seconde partie concernant la méthodologie, ayant pour but de bien positionner le problème (en utilisant le raisonnement quintilien ou la méthode des 5 P), de bien identifier la cause ou la raison des données aberrantes, c’est-à-dire la cause racine.

Nous devons être vigilants quant-à la subjectivité du terme « coût » : mieux vaut parler de performances, afin d’éviter toutes confusions.

L’étape d’implémentation de la base de données concerne à la fois une définition ainsi que la création de la base.

Nous nous interrogeons quant aux domaines. Afin de répondre à cet aspect du problème, nous aurons besoin d’effectuer d’avantages d’essais, à partir de divers jeux de données. Ne disposant pas d’une telle multitude de jeux de données, ni du temps suffisant pour les traiter, nous prévoyons pour le moment d’aller à l’essentiel : effectuer le travail de recherche concernant les méthodes existantes, c’est-à-dire de faire un état de l’art. Nous prévoyons également d’appliquer ces méthodes sur un jeu de données, relatif aux moteurs de recherche, qui nous a été fourni par Mme MOTHE. Cependant nous ne prévoyons d’étendre cette application à d’avantages de jeux de données qu’en dernier temps, si nous avons suffisamment de temps. Nous devons donc nous assurer, avec Mme MOTHE, que cette façon de procéder ne lui pose pas de problèmes.

Revue 4 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime(Fournisseur)
- MOTHE Josiane (client)

Date : 20/02/2017

Lieu : Université Paul Sabatier, Bât U3

Sujet abordé :

Le but de cette réunion avec Mme MOTHE était de faire le point sur notre avancement après la semaine de vacances.

Nous nous interrogeons au sujet du choix d'un modèle relationnel pour nos données. Nous estimons que ce choix n'était pas idéal, et nous souhaitons savoir si nous pouvions déroger à la directive d'utiliser SQL server.

Toujours au sujet du stockage des données, Mme MOTHE nous a conseillé de stocker les données issues des calculs relatifs aux méthodes de détection que nous souhaitons implémenter. Il est en effet plus optimal, surtout lorsque nous disposons de beaucoup de données, de ne pas répéter les mêmes calculs à chaque exécution des méthodes. Nous allons donc nous pencher sur ce problème.

D'autre part nous devons réfléchir dès maintenant à la façon dont nous comptons visualiser les données. Cela pourrait nous aider pour la sélection des méthodes à exécuter sur nos données.

Nous devons d'ailleurs faire un choix quant-aux méthodes que nous souhaitons mettre en application. Nous nous sommes renseignés sur beaucoup de méthodes différentes, pouvant mettre en avant des outliers, mais nous ne pouvons pas exécuter l'ensemble de ces méthodes sur nos données : nous devons faire un choix, et le justifier.

Pour chacune des méthodes, nous ne devons pas oublier de vérifier les hypothèses de validité, pour chacune des méthodes. Par exemple, la méthode de robust covariance est basée sur des données distribuées normalement.

A l'issue de l'exécution des méthodes, il pourrait être intéressant de voir si les différentes méthodes implémentées indiquent des résultats plutôt similaires ou non. Cela pourrait nous aider à tirer des conclusions quant-à la véracité des données en question, mais aussi pour avoir un regard critique au vu des méthodes utilisées et de leurs résultats.

Revue 5 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime(Fournisseur)
- BAHSOUN Wahiba (client)

Date : 27/02/2017

Lieu : Université Paul Sabatier, Bât U3

Sujet abordé :

Le but de cette réunion avec Mme BAHSOUN était de faire le point au niveau de l'avancement du projet vis à vis de la deadline de la soutenance le 17 mars, ainsi que sur notre requête d'utiliser MongoDB au lieu de SQL Server, comme prévu initialement.

Il semblerait que cela ne pose pas de problèmes, à partir du moment où nous n'oublions pas de justifier notre décision dans notre rapport. Mme BAHSOUN et M. MOKADEM sont d'accord avec cette décision.

Sinon, nous avons fait le point au sujet de la réunion précédente avec Mme MOTHE, et il semblerait qu'il n'y ait aucun problème.

Revue 6 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime(Fournisseur)
- BAHSOUN Wahiba (client)

Date : 03/03/2017

Lieu : Université Paul Sabatier, Bât U3

Sujet abordé :

Le but de cette réunion avec Mme BAHSOUN est de faire un état de l'avancement.

Le travail relatif au stockage des données est terminé.

Modèle défini (relatif aux méthodes), conformément à mongoDB.

Un peu de retard, mais pas d'imprévu, on sait où on va.

Le choix de notre problématique consiste à se restreindre aux données volumineuses. Il nous faut donc définir, et organiser le rapport en fonction.

Revue 7 :

Personnes présentes :

- GODE Valentin (Fournisseur)
- CHAOUNI Mehdi (Fournisseur)
- BRIENS Maxime (Fournisseur)
- BASHOUN Wahiba (client)
- MOTHE Josiane (client)

Date : 13/03/2017

Lieu : Université Paul Sabatier, IRIT

Sujet abordé :

Le but de cette réunion avec Mme MOTHE et Mme BASHOUN était de faire le point sur l'avancement du projet à 1 semaine de la présentation.

Nous avons ainsi discuté des méthodes, posé des questions sur le paramètre du nombre de classes. Il faudra ainsi regarder en changeant le paramètre sur le nombre de classes, si les individus considérés comme outliers sont les mêmes.

En complément des résultats obtenus via les méthodes implémentées, il nous faudra mettre en place une validation croisée, pour avoir une estimation de la fiabilité de notre modèle.

Mme MOTHE nous a ainsi parlé de plusieurs méthodes de validation croisée.

Il y a plusieurs méthodes de cross validation et donc plusieurs façons de sélectionner la méthode adaptée à notre cas (apprentissage & test) :

- Ten-fold cross-validation : Jeu de données divisé en 10, les 9 premiers forment notre jeu d'apprentissage, le 10^{ème} le jeu de test, puis on décale le rôle de chaque jeu de

données lors de chaque itération. Cette méthode est pertinente quand on travaille sur des jeux de données assez volumineux.

- Leave one out : Tous les jeux de données sont exemple d'apprentissage sauf 1, et on décale les rôles de chaque jeu de données pour qu'ils servent tous. Idéal pour peu de données (adapté à notre cas).

Durant cette revue, Mme MOTHE nous a également fait un petit récapitulatif sur la description de nos variables.

Et pour finir, nous avons eu une discussion au sujet du titre de notre rapport. Le titre « données à hautes dimensions », est plus adaptés à notre cas d'étude. En effet, notre jeu de données est en (relative) haute dimension!

11 Bilan

11.1 Bilan client

Du point de vu client, celui-ci récupère un état de l'art de 14 méthodes de détection de données « outliers ». En complément de cette état de l'art, une étude de cas complète (récupération des données, stockage, implémentation d'algorithme et visualisation des résultats) est également fournie. Cette étude illustre donc un exemple concret de détection de données « outliers », via des outils comme MongoDB très utilisés en entreprise en ce moment notamment sur des problématiques liées au Big Data.

11.2 Bilan Fournisseur

Du point de vue fournisseur, le cahier des charges a été respecté. En effet, l'état de l'art réalisé couvre une bonne partie des méthodes existantes pour la détection de données « outliers ». De plus, la partie concernant la détection de ces données sur des données concrètes illustre l'utilisation des méthodes retenues sur un jeu de données avec beaucoup de variables et peu d'individus.

En perspective d'évolution de notre projet, il serait pertinent d'implémenter les méthodes retenues sur :

- Un jeu de données composé de beaucoup d'individus et de peu de variables.
- Un jeu de données composé de beaucoup d'individus et de beaucoup de variables.

12 Difficultés rencontrées

Durant ce projet, nous avons rencontrés plusieurs difficultés, relatives à la structure des données, au stockage, à la compréhension des algorithmes et à l'implémentation de ceux-ci.

Pour la partie donnée, nous avons à notre disposition beaucoup de variables et peu d'individus. De plus les données relatives à chaque moteur de recherche n'avaient pas le même nombre de champs et des variables différentes. Au départ, nous comptions stocker nos données dans une base de données relationnelles qui ne convenaient donc pas à nos besoins comme expliqué dans la partie données. Nous avons donc du assimiler une nouvelle approche, à savoir parler et travailler en termes de « documents », et mettre en place une méthodologie pour le stockage associé.

Pour la partie algorithmes, nous avons rencontrés dans le cadre de la recherche bibliographique, des méthodes de détection parfois complexe, relativement récentes et donc sans documentation, ou trop peu détaillées. On peut par exemple noter la méthode « Replicator Neural Network » qui est renseignée par un seul document sur internet.

Et enfin pour l'implémentation, certaines méthodes retenues nécessitent un échantillon d'apprentissage annoté, qui permettrait en quelques sortes de définir les outliers au sein de nos données. Or nous ne disposons pas de cette information. Nous avons donc du réfléchir à un autre moyen pour malgré tout implémenter la méthode SVM one-class.

De plus certaines méthodes sont plus ou moins efficaces selon la dimensionnalité de nos données, ce qui est problématique dans notre cas en vue du nombre de variables disponible.

Nous avons également eu quelques soucis au sujet des valeurs manquantes. Nous aurions pu trouver des moyens pour les calculer, ou les déduire, mais cela n'était pas plus probant que de supprimer quelques informations, afin de supprimer ces valeurs manquantes. Notons cependant que, tout comme nous faisons ce rapport sur la détection d'outliers, il serait tout aussi intéressant et complexe d'étudier les valeurs manquantes, et les façons de se prémunir de ce problème.

13 Conclusion

Pour conclure, ce projet nous a permis de :

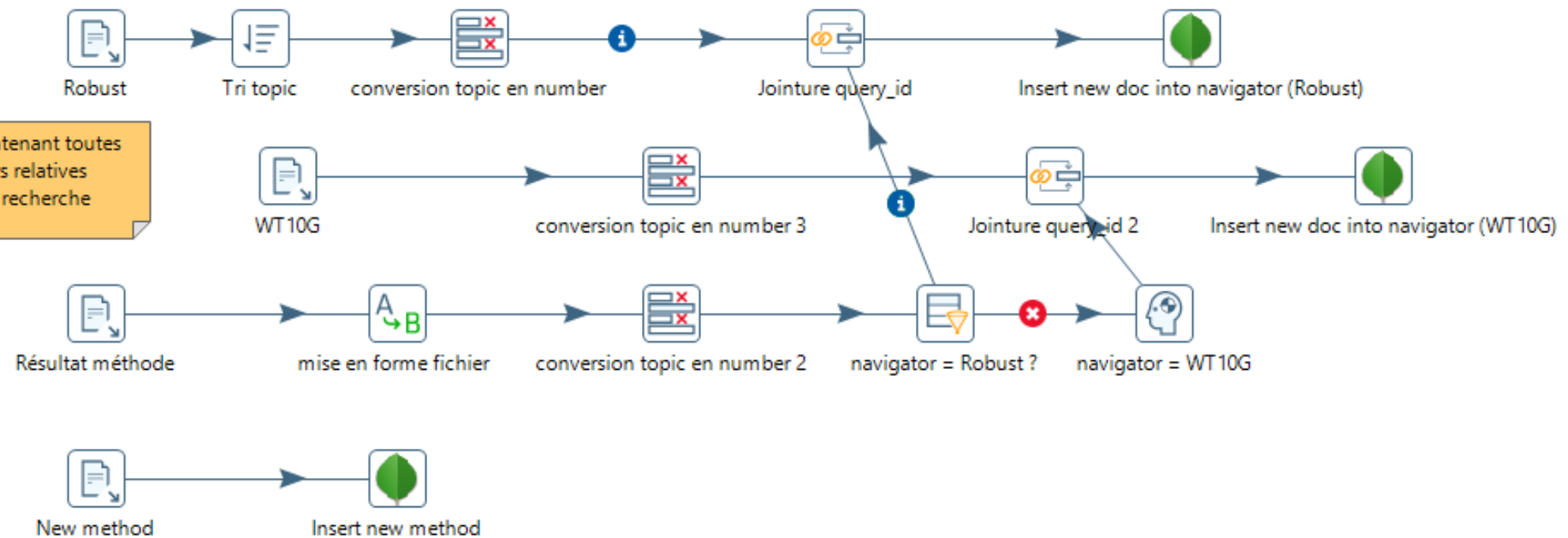
- Découvrir de nouvelles méthodes de détection sur une problématique concrète et très recherchée en ce moment (AIRBUS,...)
- Implémenter ces méthodes sur un jeu de données réel, et complexe
- Utiliser le « NoSQL » pour le stockage de nos données
- Découvrir de nouveaux outils (MongoDB, KNIME, packages R)
- Représenté graphiquement nos résultats

Ce projet constitue donc d'un point de vue général une expérience très intéressante et très enrichissante.

Annexe 1 : Bibliographie

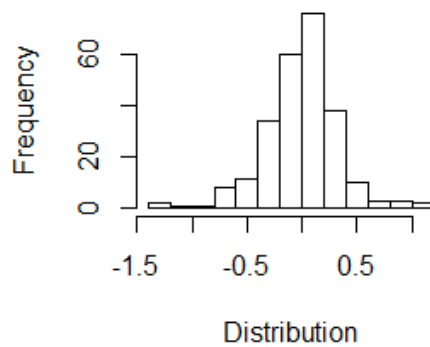
- Modified Thompson Tau test :
<http://www.mne.psu.edu/cimbala/me345/Lectures/Outliers.pdf>
- Ensemble learning :
<https://arxiv.org/pdf/1609.05528v1.pdf>
- Replicator Neural Networks :
<http://togaware.com/papers/dawak02.pdf>
- Grubbs test :
<http://www.itl.nist.gov/div898/handbook/eda/section3/eda35h1.htm>
- Tietjen-moore test :
<http://www.itl.nist.gov/div898/handbook/eda/section3/eda35h2.htm>
- Generalized ESD test :
<http://www.itl.nist.gov/div898/handbook/eda/section3/eda35h3.htm>
- Robust covariance :
<http://scikit-learn.org/stable/modules/covariance.html#robust-covariance-estimation>
- One class SVM :
http://scikit-learn.org/stable/auto_examples/ensemble/plot_isolation_forest.html
- Isolation Forest :
http://scikit-learn.org/stable/modules/outlier_detection.html#isolation-forest
- DBSCAN :
<http://scikit-learn.org/stable/modules/clustering.html#dbscan>
- Birch :
<http://scikit-learn.org/stable/modules/clustering.html#birch>
- Robustness regression :
http://scikit-learn.org/stable/modules/linear_model.html#robustness-regression-outliers-and-modeling-errors
- Replicator Neural Networks :
<https://togaware.com/papers/dawak02.pdf>
- K-Means clustering :
<http://www.pmg.it.usyd.edu.au/outliers.pdf>

Annexe 2 : Transformation des données via l'ETL

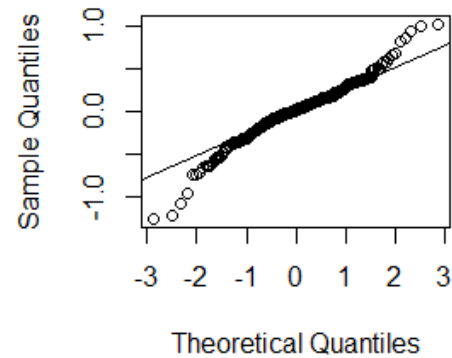


Annexe 3 : QQplot Robust

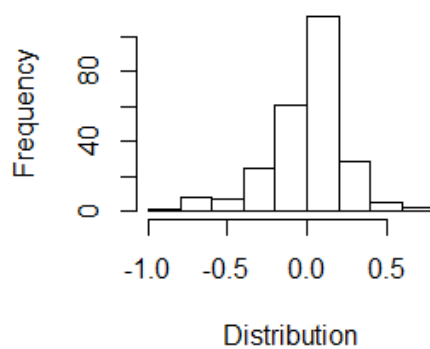
Histogramme : PCA.dimension.0



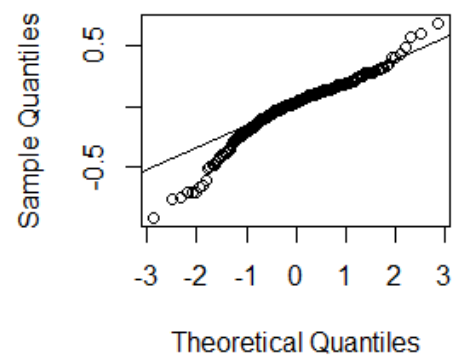
Normal Q-Q Plot : PCA.dimension.0



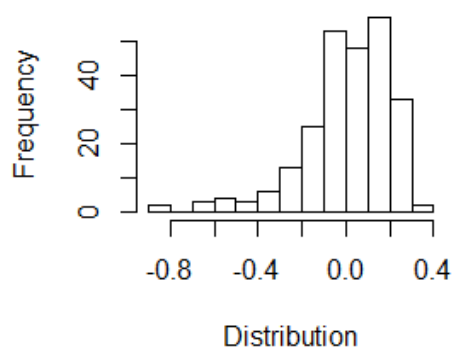
Histogramme : PCA.dimension.1



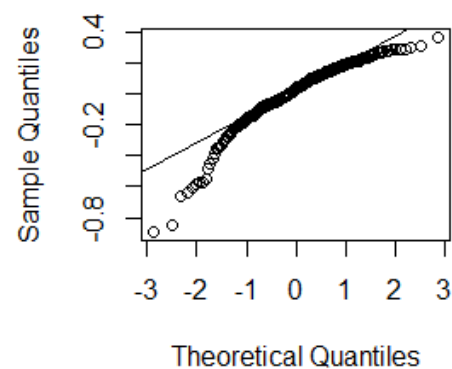
Normal Q-Q Plot : PCA.dimension.1



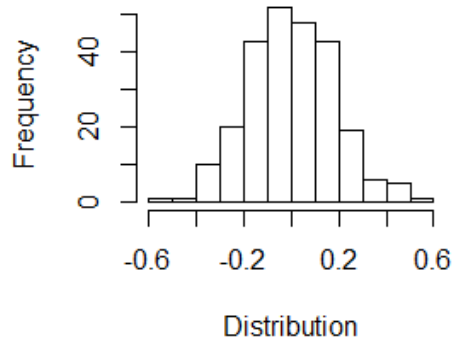
Histogramme : PCA.dimension.2



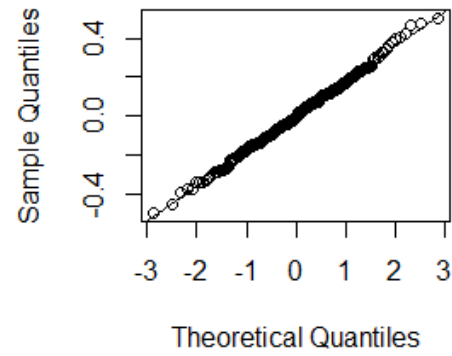
Normal Q-Q Plot : PCA.dimension.2



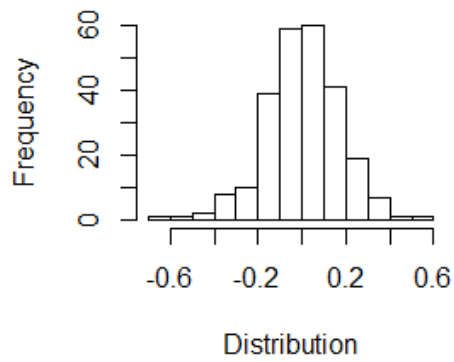
Histogramme : PCA.dimension.3



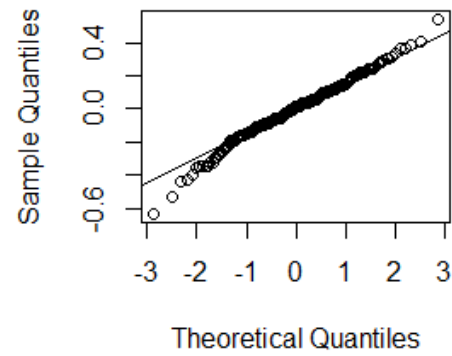
Normal Q-Q Plot : PCA.dimension.3



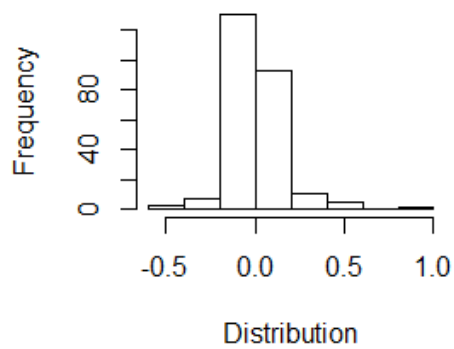
Histogramme : PCA.dimension.4



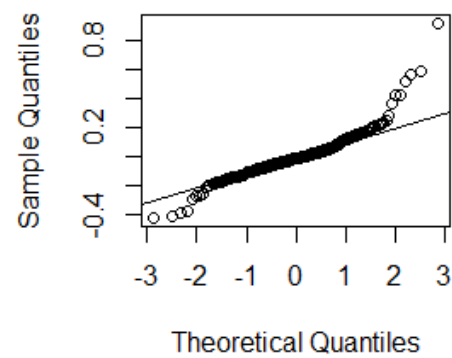
Normal Q-Q Plot : PCA.dimension.4



Histogramme : PCA.dimension.5

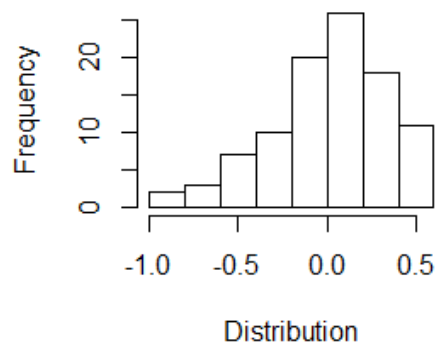


Normal Q-Q Plot : PCA.dimension.5

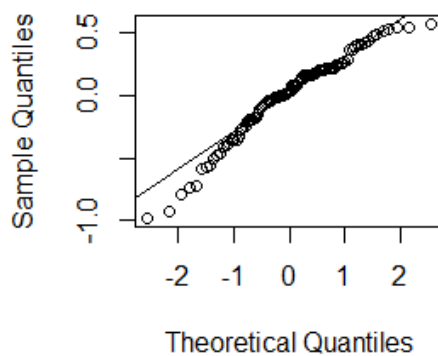


Annexe 4 : QQplot WT10G

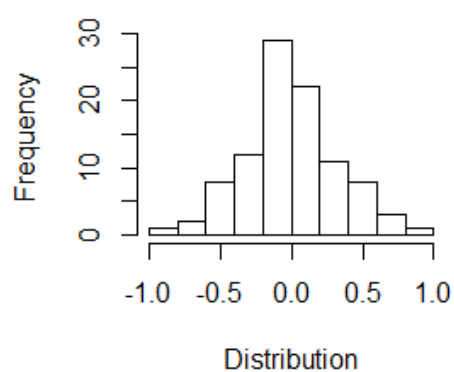
Histogramme : PCA.dimension.0



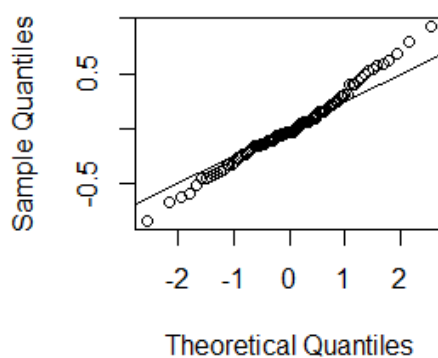
Normal Q-Q Plot : PCA.dimension.0



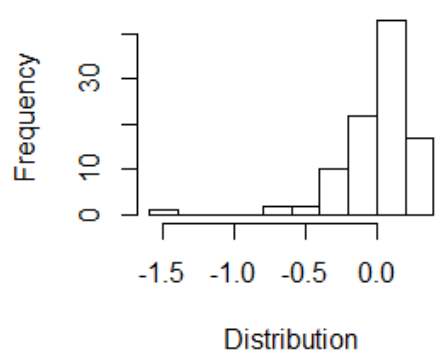
Histogramme : PCA.dimension.1



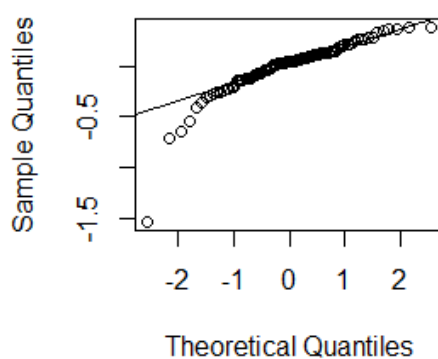
Normal Q-Q Plot : PCA.dimension.1



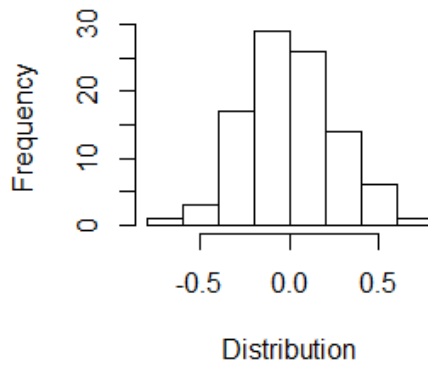
Histogramme : PCA.dimension.2



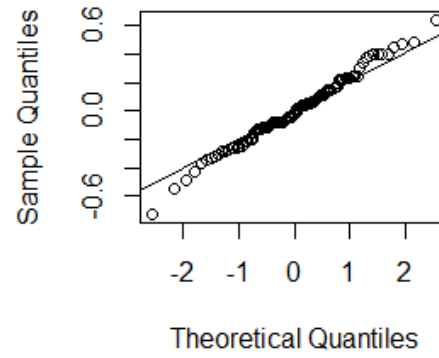
Normal Q-Q Plot : PCA.dimension.2



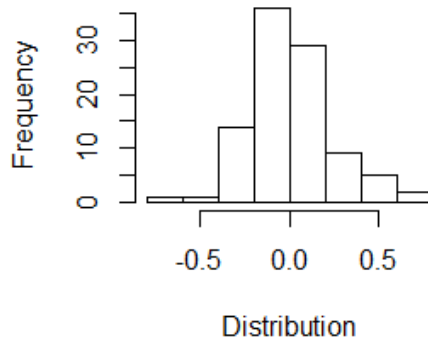
Histogramme : PCA.dimension.3



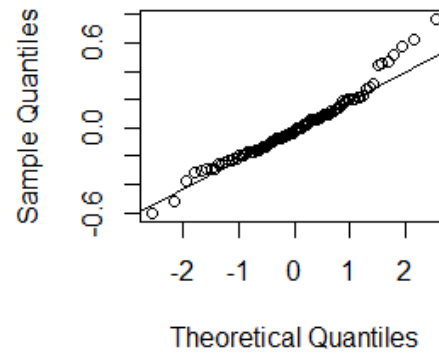
Normal Q-Q Plot : PCA.dimension.3



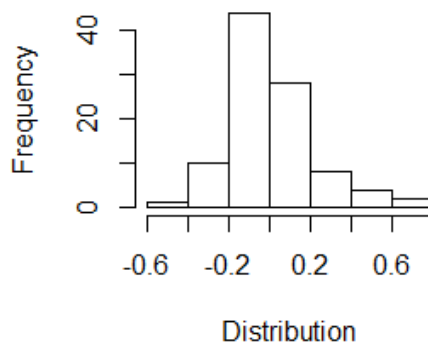
Histogramme : PCA.dimension.4



Normal Q-Q Plot : PCA.dimension.4



Histogramme : PCA.dimension.5



Normal Q-Q Plot : PCA.dimension.5

