UNVERSITÉ NATIONALE DU VIETNAM À HO CHI MINH
UNIVERSITÉ DES SCIENCES
FACULTÉ DE TECHNOLOGIE DE L'INFORMATION

NGUYEN MAI NGOC LINH - 1552007
HOA MINH LUAN - 1552008

RECHERCHE DE L'ALGORITHME DE REGROUPEMENT SUR L'OUTIL APACHE SPARK POUR LA SEGMENTATION DU MARCHÉ

MÉMOIRE DE FIN D'ÉTUDE

À HO CHI MINH VILLE, 2019

UNVERSITÉ NATIONALE DU VIETNAM À HO CHI MINH
UNIVERSITÉ DES SCIENCES
FACULTÉ DE TECHNOLOGIE DE L'INFORMATION

NGUYEN MAI NGOC LINH - 1552007
HOA MINH LUAN - 1552008

RECHERCHE DE L'ALGORITHME DE REGROUPEMENT SUR L'OUTIL APACHE SPARK POUR LA SEGMENTATION DU MARCHÉ

MÉMOIRE DE FIN D'ÉTUDE

SOUS LA DIRECTION DE

LE THI NHAN

À HO CHI MINH VILLE, 2019

REMERCIEMENTS

Nous tenons tout d'abord à remercier Mme LE Thi Nhan, notre encadrante de mémoire de fin d'étude, qui nous a aidé à diriger notre travail et nous a donné la patience et le courage tout au long de ce mémoire. Grâce à ses précieux conseils et ses encouragements, nous avons finit bien notre travail et ce mémoire.

Nous adressons également nos remerciements à tous les membres du jury qui ont examiné notre travail et nous ont proposé des remarques constructives.

Ensuite, nos remerciements vont à M. TRUONG Dinh Vu, pour ses efforts et ses aides à la relecture et à la correction de notre mémoire.

Enfin, nous voudrions remercier à notre famille et nos amis qui nous ont soutenu et encouragé afin de réaliser ce travail.

PLAN DETAILLE

Nom de mémoire :

Recherche de l'algorithme de regroupement sur l'outil Apache Spark pour la segmentation du marché

Encadrante du mémoire : Mme. LE Thi Nhan

Temps de réalisation : 07/01/2019 – 30/06/2019

Étudiants :

NGUYEN Mai Ngoc Linh – 1552007

HOA Minh Luan - 1552008

Type du mémoire : Rechercher

Description

Aujourd'hui, le marketing porte toujours un rôle important, parce qu'il est un moyen pour que les entreprises présentent efficacement leurs nouveaux produits et services aux consommateurs, ainsi, augmentent les revenus et les profits. L'une des méthodes de marketing les plus populaires est la segmentation du marché. La méthode traditionnelle de la segmentation consiste à l'analyse des spécialistes d'économique afin de déterminer le segment approprié pour chaque groupe de client. Toutefois, cette méthode prend beaucoup de temps, car notre capacité de travail sur une grande volumétrie de données est limitée. Dans ce mémoire, nous automatisons le processus de segmentation du marché par l'algorithme K-means sur l'ensemble de données Amadeus Entertainment sur la plate-

forme Apache Spark. Enfin, nous extrayons les informations exprimant les caractères spécifiques des clients dans chaque groupe.

Objectif

- Rechercher la théorique de la segmentation du marché
- Implémenter l'algorithme K-means Baseline sur Apache Spark et extraire les caractéristiques du cluster
- Expérimenter et évaluer l'efficacité de l'algorithme

Outils

- Langage, bibliothèque : Python, PySpark
- Environnement : Jupyter Notebook
- Plate-forme : Apache Spark
- Slide, rapport : MS Word, MS Powerpoint

Épreuves

- Créer l'ensemble d'entrainement à partir de l'ensemble de données brut qui consiste à beaucoup tables enregistrant les informations de commande des clients
- Développer sur Apache Spark – la plate-forme de Big data, il est nécessaire un concept différent de la programmation
- Assurer la qualité du clustering de l'algorithme K-means Baseline

Contenu

Ce projet comprend cinq grandes parties :

- Rechercher la théorique de la segmentation du marché et de la plate-forme Apache Spark
- Rechercher les algorithmes de clustering

- Implémenter K-means Baseline sur Spark
- Extraire les caractéristiques de chaque groupe
- Utiliser les méthodes différentes d'évaluation pour comparer l'efficacité de K-means Baseline à K-means fournissant par la bibliothèque de Spark

Plan de mise en œuvre

- 05/01/2019 – 15/02/2019 : Rechercher la théorique de la segmentation du marché et le fondement d'Apache Spark
- 16/02/2019 – 15/03/2019 : Rechercher les algorithmes de regroupement et l'ensemble de données approprié
- 16/03/2019 – 05/04/2019 : Analyser et prétraiter l'ensemble de données
- 06/04/2019 – 03/05/2019 : Exécuter K-means de la bibliothèque MLlib avec le nombre de cluster de 2 à 1000 pour trouver K approprié
- 04/05/2019 – 24/05/2019 : Implémenter K-means Baseline sur Apache Spark
- 25/05/2019 – 31/05/2019 : Visualiser les résultats et découvrir les caractéristiques de chaque cluster
- 01/06/2019 – 15/06/2019 : Expérimenter et évaluer la qualité de l'algorithme
- 15/06/2019 – 21/06/2019 : Écrire le rapport
- 22/06/2019 – 29/06/2019 : Modifier le rapport
- 30/06/2019 : Compléter

Confirmation d'encadrante	**Les étudiants**

TABLE DE MATIÈRE

TABLE DES FIGURES

TABLE DES TABLEAUX

RÉSUMÉ

La segmentation du marché est une méthode de marketing utilisé pour diviser le marché en petites parties (les segments) en basant de quelques critères spécifiques. Cette méthode aide les entreprises à comprendre les caractéristiques de chaque segment pour élaborer des stratégies appropriées de marketing, ainsi augmenter les revenus de l'entreprise. Il existe de nombreuses méthodes pour segmenter le marché. L'une des méthodes la plus commun est de diviser un ensemble de clients par leur histoire d'achat. Dans ce mémoire, nous analysons l'ensemble de données Amadeus Entertainment et réalisons la segmentation du marché en fonction de l'histoire des transactions des clients. En particulier, nous appliquons l'algorithme K-means sur la plate-forme Apache Spark pour automatiser le processus de la segmentation du marché. Nous implémentons et exécutons l'algorithme K-means avec les nombres différents des clusters et évaluons la qualité du regroupement à chaque exécution. Enfin, nous avons décidé de diviser l'ensemble de clients en 66 groupes car après la division, chaque groupe a ses propres caractéristiques et l'indice de Purity de toutes les données gagne 0,5759110764624777.

1. INTRODUCTION

Dans ce chapitre, nous présentons les méthodes de marketing, les objectifs principaux et enfin l'organisation du mémoire.

1.1. Introduction du marketing

Le marketing est une tâche importante pour toutes les entreprises. C'est un moyen pour que les entreprises présentent ses produits et ses services aux consommateurs. Par conséquent, l'efficacité du marketing influence directement aux résultats de commerces des entreprises. Il existe de nombreuses méthodes différentes de marketing, surtout le marketing de masse et la segmentation du marché sont deux méthodes les plus populaires. Selon des différents objectifs de marketing, les entreprises choisissent la méthode appropriée.

Le marketing de masse est une méthode destinant à tous les gens dans le marché. La forme la plus commun de cette méthode est la publicité dans les journaux, à la télévision ou dans les médias pour que tout le monde puisse facilement approcher. En utilisant cette façon, les entreprises souhaitent les informations de marketing peuvent s'approcher autant de monde que possible pour identifier leur marque et présenter leurs nouveaux produits ou leurs promotions. Par exemple, en 2003, l'entreprise Ford Motor a proposé une stratégie publicitaire pour les camions F-150. En particulier, elle a fait de la publicité sur tous les moyens de médias en même temps. Donc tous les téléspectateurs doivent regarder la publicité de F-150. Cette stratégie a remporté un grand succès [1].

Le marketing de masse donne aux entreprises beaucoup des avantages, mais les inconvénients existent également. D'une part, il aide les entreprises à apporter les informations des produits à un grand nombre de clients. En particulier, cette méthode est efficace lorsque l'entreprise voudrait orienter leurs produits à tout le monde. Toutefois, les produits doivent avoir des caractéristiques attirant beaucoup de gens (ex : le prix, la promotion, …) afin de limiter les utilisateurs à acheter les produits des rivaux [2].

La segmentation du marché est inversée avec le marketing de masse. C'est-à-dire, au lieu d'espérer que les produits gagneraient autant de clients que possible, les entreprises divisent leurs clients en petites groupes et déploient la stratégie de marketing pour chaque groupe de

clients. Par exemple, pour quelques détaillants, leurs clients peuvent être divisés par les rangs différents (cuivre, argent, or, etc.) basant au total de l'argent qu'ils ont payé. Ensuite, pour chaque rang, on a des façons différentes de marketing, par exemple il y a une meilleure réduction pour le rang plus haut.

Pour segmenter le marché, les entreprises ont besoin d'aide des spécialistes de marketing, elles recherchent l'ensemble de données et le but de marketing pour sélectionner des critères appropriés de la segmentation. Il y a de nombreux critères pour la segmentation de la clientèle, mais les plus courants sont les critères par géographie (le pays, la ville, la densité de population, la température, la langue, etc.), par informations personnelles (le sexe, l'âge, l'éducation, la profession, etc.), par psychologie (la personnalité, l'habitude, le style de vie, etc.) ou par comportement (la fidélité, l'histoire des achats des clients, etc.). Pour chaque critère, il y a des méthodes différentes de la segmentation. Par exemple, si la segmentation du marché est basée sur le comportement, en particulier l'histoire des achats des clients, les spécialistes résument toutes les informations de l'histoire des transactions des clients, puis, elles groupent les clients qui s'intéressent à un certain groupe de produits au même cluster.

La segmentation du marché aide les entreprises à connaître bien les demandes des clients dans chaque groupe. Alors, ils peuvent modifier les produits pour concorder avec les besoins des acheteurs. De plus, cette méthode aide également les entreprises à répartir rationnellement leurs budgets de marketing. Néanmoins, la division du marché et les stratégies différentes pour chaque segment augmentent le budget de marketing des entreprises. Notamment, les critères des segments et les stratégies de marketing de chaque groupe changent constamment parce que le changement rapide du marché. Ainsi, le prix du produit doit augmenter pour compenser le budget de marketing [3].

Pour surmonter les faiblesses de la méthode de segmentation du marché, ce mémoire souhaite automatiser le processus de la division du marché. Grâce à la vitesse de calcul supérieure et à la capacité de travailler en Big data, le processus de la segmentation du marché sera raccourci et la qualité du segment sera améliorée. Depuis, on peut minimiser le budget de marketing.

La segmentation automatique du marché peut être réalisée en utilisant des algorithmes de clustering dans l'apprentissage automatique. Ces algorithmes de clustering sont des

algorithmes utilisés pour diviser un ensemble de données en petits groupes en basant des caractéristiques des points de données. Après du regroupement, les caractéristiques des points d'un même cluster seront similaires, mais seront différents aux points des autres clusters. Ces algorithmes sont divisés en deux groupes principaux : le clustering hiérarchique (Agglomerative, Devisive) et le clustering non hiérarchique (Single-pass, Relocation, Nearest neighbor, etc, …) [4].

Dans ce mémoire, nous segmentons du marché en analysant du comportement de la clientèle, spécifiquement, en fonction de l'histoire des achats de la clientèle chez Amadeus Entertainment. Pour chaque client, nous calculons la somme de produits qu'il a acheté pour chaque type de produit. Ensuite, nous procédons à la mise en cluster automatique en utilisant l'algorithme K-means (l'algorithme appartient au groupe de Relocation dans le cluster non hiérarchique) sur la plate-forme Apache Spark. Le nombre de clusters est le nombre de segments du marché et les caractéristiques de chaque cluster sont les caractéristiques des clients dans ce cluster.

1.2. Objectifs du mémoire

Ce projet se compose donc de quatre parties :

- Rechercher la théorique et les méthodes de segmentation du marché
- Rechercher et implémenter l'algorithme K-means Baseline sur Apache Spark
- Appliquer l'algorithme pour regrouper et extraire les caractéristiques de chaque cluster
- Expérimenter pour évaluer l'efficacité de l'algorithme sur les différentes mesures et entre l'algorithme K-means de Spark avec K-means Baseline.

1.3. Organisation du mémoire

Le mémoire est divisé en 7 chapitres suivants (y compris ce chapitre « Introduction »)

- **Chapitre 2 : Travaux connexes** - nous présentons les travaux connexes
- **Chapitre 3 : Big data et Apache Spark** - nous présentons le Big data, la méthode MapReduce, l'architecture et les caractéristiques principaux d'Apache Spark

- **Chapitre 4 : Algorithme K-means** - nous présentons l'algorithme K-means, les algorithmes de clustering sur Big data et les méthodes d'évaluation de la qualité du regroupement
- **Chapitre 5 : Mise en œuvre de mémoire** - nous présentons les étapes pour mettre en œuvre du mémoire et la méthode de la réalisation dans chaque étape
- **Chapitre 6 : Expérimentation** - nous présentons les objectifs, les méthodes de réalisation et les résultats obtenu. De plus, nous décrivons les caractéristiques des clusters reçu
- **Chapitre 7 : Conclusion** - nous présentons le résultat obtenu, les inconvénients et l'orientation de développement.

2. TRAVAUX CONNEXES

Jusqu'aujourd'hui, il y a de nombreuses études ayant appliqué des algorithmes de clustering afin de résoudre du problème de la segmentation du marché. Dans ce mémoire, nous présentons brièvement quelques études utilisant l'algorithme de clustering et les caractéristiques de l'ensemble de données utilisé.

Au début de 2019, Grzegorz Maciejewski et ses collèges [6] ont mené une étude pour déterminer si les éléments de l'environnement auraient été des caractéristiques de l'ensemble de données, avec les facteurs comportementaux du client, pour segmenter le marché de café. D'abord, ils ont fait une enquête de 30 questions et ont obtenu les données de 800 clients. Ensuite, ils ont utilisé EFA (Exploratory Factor Analysis) pour réduire la dimension et créer 5 nouveaux caractères. Après de l'application de plusieurs algorithmes de clustering pour regrouper sur le nouvel ensemble de données, ils ont choisi l'algorithme de clustering non hiérarchique pour segmenter le marché. Le résultat obtenu 6 clusters, et à chaque cluster correspondant, ils extraient facilement leurs caractéristiques lorsque 70 à 80% des clients du cluster se sont intéressés à un même problème.

Dans une autre étude, Matthias Carnein et Heike Trautmann [7] ont présenté l'algorithme Stream Clustering pour résoudre le problème de la segmentation du marché. Ils ont utilisé les données d'un détaillant de meubles avec plus de 1.7 millions de transactions de 500,000 clients de juin 2014 à décembre 2017. Les informations de transaction incluent le taux de retour des produits achetés, la moyenne de la quantité de produits dans une commande, la date du dernier achat. Après de la mise en cluster, ils ont utilisé l'indice Silhouette pour évaluer la qualité du regroupement et finalement, ils ont analysé les clusters afin de déterminer leurs caractéristiques.

En 2013, Kishana R. Kashwan [8] et ses collègues ont proposé d'utiliser l'algorithme K-means pour classifier des clients. Après de la présentation de la segmentation du marché et de l'algorithme K-means, ils ont appliqué cet algorithme avec quatre clusters pour segmenter les clients sur l'un ensemble de données d'un supermarché vendant des produits ménagers. L'ensemble de données est le résultat de l'enquête de 2138 clients avec les évaluations de 1 point (en désaccord) à 5 points (tout à fait d'accord) pour 15 questions. Après du

regroupement, ils ont utilisé Analysis of Variance (ANOVA) pour évaluer les résultats du regroupement. Enfin, grâce aux valeurs statistiques, ils ont extrait les caractéristiques de chaque cluster.

De plus, en 2018, Shreya Tripathi [9] et ses collègues ont également proposé d'appliquer l'algorithme de clustering pour segmenter le marché. Dans l'article, ils proposent deux algorithmes de clustering, K-means et Hierarchical Clustering, ainsi que les avantages et les inconvénients de chaque algorithme. Pour comparer les deux algorithmes, ils ont utilisé un ensemble de données de centre commercial comprenant des informations sur le nom, l'âge, le sexe, le revenu et l'indice des achats de 200 clients. Enfin, ils ont conclu qu'avec cet ensemble de données, K-means donne au meilleur résultat.

Les informations sur les ensembles de données et les algorithmes utilisés par chaque étude sont résumés dans le Tableau 2-1 ci-dessous :

Tableau 2-1 : Résumé des informations des recherches

Auteurs	**Ensemble de donnée**	**Algorithme**
Grzegorz Maciejewski et ses collèges	Le résultat du sondage auprès de 800 clients pour 30 questions	EFA, Hierarchical Clustering, K-means
Matthias Carnein et Heike Trautmann	1.7 millions de transactions de 500,000 clients	Stream Clustering
Kishana R. Kashwan et ses collèges	Le résultat de l'enquête auprès de 2138 clients pour 15 questions	K-means
Shreya Tripathi et ses collèges	Les informations personnelles de 200 clients	K-means, Hierarchical Clustering

L’utilisation de l’apprentissage automatique pour regrouper les consommateurs dans l’analyse et la segmentation du marché aide les entreprises à économiser le budget pour le marketing. Toutefois, aujourd’hui, la volumétrie de données augmente rapidement. Par exemple, le nombre total de journaux de Facebook par jour était de 60 TB ou l'indice de Google Web dépassait 10 PB[1], avec un seul ordinateur, il n’est pas possible de stocker ou traiter ces données. Donc, on a besoin de diviser et distribuer cet ensemble de données aux les différentes machines, traiter sur chaque sous-machine et finalement, rassembler leurs résultats. On souhaite que le résultat de clustering s'exerçant sur nombreuse machines soit presque équivalent au résultat du regroupement sur une machine.

En 2015, Btissam Zerhari et ses collèges [10] a utilisé “3V” (Volumn, Velocity, Variety) pour déterminer le Big data. En même temps, ils ont proposé quelques méthodes différentes pour le regroupement sur le Big data (Figure 2-1) et ont montré également les avantages et les inconvénients de chaque méthode.

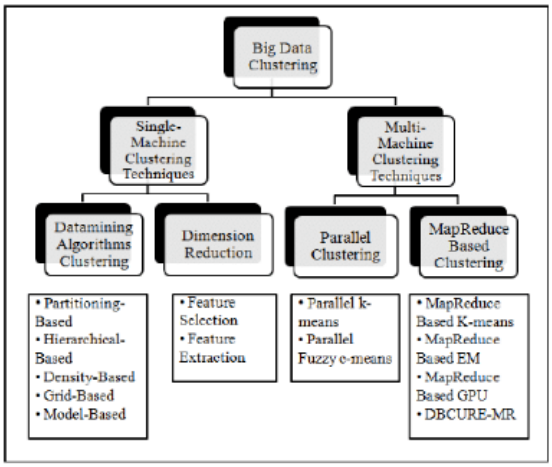

Figure 2-1 : Les méthodes du regroupement sur le Big Data (via [10])

En raison des résultats positifs des auteurs ci-dessus, nous sommes fondées à appliquer les algorithmes de regroupement pour segmenter le marché.

[1] Cours « Introduction to Spark » - edX.com

Dans ce mémoire, bien que l'ensemble de données pour l'analyse et l'exécution d'algorithme ne soit pas très grand (environ 35 000 lignes x 65 colonnes), nous voulons orienter à la capacité de regrouper du modèle sur le Big data, donc, nous avons décidé d'installer l'algorithme K-means sur la plate-forme Apache Spark (une plate-forme supportant du traitement le Big data) en basant de l'idée de MapReduce Based K-means proposant par les auteurs ci-dessus. L'ensemble de données et l'algorithme d'installation seront décrits dans les chapitres suivants.

3. BIG DATA ET APACHE SPARK

Dans ce chapitre, nous présentons d'abord la définition et les caractéristiques du Big data, ensuite la méthode MapReduce, enfin la plate-forme Apache Spark.

3.1. Big data

3.1.1. Introduction

Le Big data est un ensemble de données dont taille est très grande et dont structure est très compliquée. Nous ne pouvons pas donc le traiter par les l'outil de gestion de base de données ou par les applications traditionnelles [11].

Le Big data peut apparaître de nombreuses sources. Les principales sources sont les données des entreprises, les données des activités d'applications Web et mobiles, les données de sites de réseaux sociaux ou les données assemblant et publiant par le gouvernement. Ils ont une très grande taille, une création très rapide et des nombreuses structures.

3.1.2. Caractéristique

Les caractéristiques générales de la source de données ci-dessus sont également trois caractéristiques de base du Big data : la taille (volume), la vitesse (velocity) et la variété (variety) - Figure 3-1

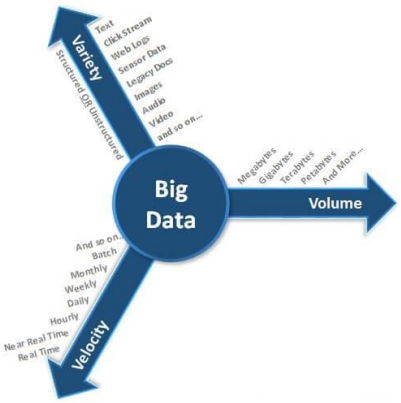

Figure 3-1 : "3V" du Big data [11]

Premièrement, la caractéristique la plus évidente du Big data est sa taille. Travailler avec la grande volumétrie de données signifie qu'on doit traiter les données de dizaines de GO à

nombreuses PO. Avec la vitesse de développement actuel, ce nombre augmentera encore plus (par exemple, les données de Google en 2014 ont atteint environ 15 000 PO[2]). Avec un ensemble de données à dizaines GO, un ordinateur avec une configuration solide peut les manipuler, mais cela prend certainement beaucoup de temps. De plus, lorsque la taille des données augmente trop, il est impossible de les traiter et de les analyser avec un seul ordinateur. La taille de l'ensemble de données influence au matériel et aux méthodes de traitement, donc, on a besoin de réfléchir avant travailler avec le Big data.

Deuxième caractère du Big data est la vitesse de la création des données. En basant des différents problèmes, la demande de temps de traitement des données sont aussi différents. Par exemple, dans le système de pilote automatique de l'avion, il est nécessaire de gérer une grande volumétrie de données pendant très peu de temps afin de répondre à la situation actuelle de l'avion. Donc, le système doit manipuler des données extrêmement rapides. L'exemple ci-dessus montre l'importance de la vitesse entre la création et le traitement des données. Ainsi, elle devient une caractéristique du Big data.

La dernière caractéristique du Big Data est la variété, qui crée les différences des formats des ensembles de données. Il s'est généralement composé en deux types principaux : les données structurées et non structurées. Pour chaque type, nous devons appliquer les différentes méthodes de stockage et de traitement. Spécialement, l'augmentation rapide et la variété de données non structurées (texte, images, audio, etc.) rend le stockage et le traitement plus difficiles. Donc, elle devient la troisième « V » à côté de deux caractéristiques ci-dessus.

3.2. MapReduce

MapReduce est un modèle plus efficace et rapide pour traiter des données. Ce modèle consiste à deux fonctions Map() et Reduce(). D'abord, Map() est un processus de la transformation de la structure des données à (clé, valeur). Ensuite, Reduce() combine cet ensemble des (clé, valeur) dans étape Map() afin de créer un nouveau ensemble des (clé, valeur) mais la taille de cet ensemble est inférieure. De plus, MapReduce ajoute des étapes entre Map() et Reduce()

[2] https://followthedata.wordpress.com/2014/06/24/data-size-estimates/

comme : Diviser (split), combiner (combine), mélanger (shuffle) et trier (sort) pour former un flux d'activité complet (Figure 3-2).

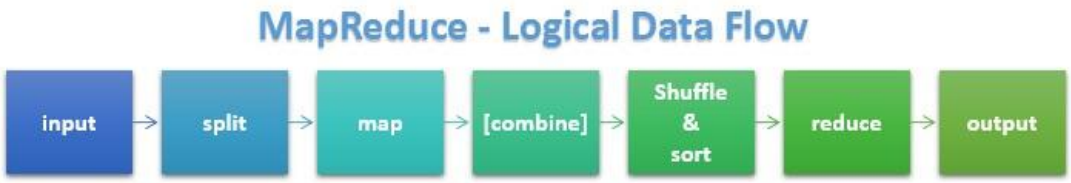

Figure 3-2 : Le flux d'activité de MapReduce[3]

L'exemple le plus simple du modèle MapReduce est le problème « Word Count » (Figure 3-3)

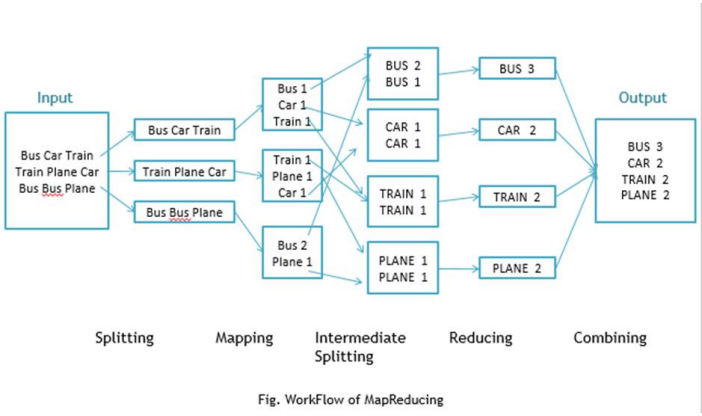

Figure 3-3 : L'exemple de « Word Cout »[4]

Le but du problème est de compter le nombre d'occurrences de mots dans un paragraphe. Par exemple, on a le paragraphe suivant : "Bus Car Train Train Plane Car Bus Bus Plane", et on a besoin de compter le nombre d'occurrences de « Bus », « Car », « Train » et « Plane ». Tout d'abord, imaginez que ce texte soit très volumineux et ne puisse pas être stocké sur une seule machine, donc, on a besoin de diviser (splitting) le texte et le distribuer aux sous-machines en vue de traiter. Pour chaque sous-machine, la première étape est Map(), qui permet de découper le texte en mots sous la forme de (clé, valeur) contenant le mot apparu et la valeur 1 (pour dire que le mot a apparu une fois). Ensuite, les sous-machines réorganiseront le nouvel ensemble de données en assemblant les mêmes clés (splitting). Puis, l'étape Reduce() fait la

[3] https://www.mssqltips.com/sqlservertip/3222/big-data-basics--part-5--introduction-to-mapreduce/
[4] https://dzone.com/articles/word-count-hello-word-program-in-mapreduce

somme des values pour chaque mot ayant même clé dans le but de trouver le nombre total d'occurrences de ces mots dans le texte. Enfin, le modèle assemblera et donnera le résultat final.

3.3. Apache Spark

Dans cette partie, nous présentons le concept et les composants d'Apache Spark, RDD, DAG et le concept de la programmation sur Spark.

3.3.1. Introduction

Spark est un framework open source afin de travailler avec le Big data. La différence plus frappante de Spark est qu'il ne lit les données sur HDD/SSD qu'une fois, puis il traite et convertit les données en RAM, enfin il écrit dans la mémoire externe. En raison du transfert des données entre CPU et RAM est très rapide, donc, Spark exprime une vitesse idéale.

Par ailleurs, Apache Spark fournit aux utilisateurs un écosystème dans le but de travailler dans nombreux domaines différents (Figure 3-4)

Figure 3-4 : L'écosystème de Spark[5]

Pour chaque bibliothèque, Spark supporte 4 langages de programmation courants dans le domain du traitement de données : Scala, Java, Python et R. Dans les bibliothèques ci-dessus, nous nous intéressons particulièrement à MLlib, une bibliothèque fournissant des algorithmes d'apprentissage sur Spark comme : la régression, le clustering, la classification, etc. De plus,

[5] https://www.edureka.co/blog/spark-architecture/

MLlib fournit des fonctions statistiques, des fonctions de l'extraction les caractéristiques de données et supporte au stockage des modèles formés.

L'architecture d'Apache Spark est développé en le mécanisme de master - worker (Figure 3-5).

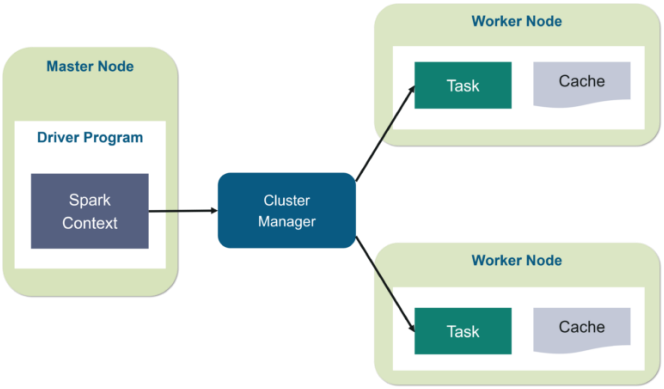

Figure 3-5 : L'architecture d'Apache Spark[6]

Le master contient le programme à exécuter (Driver Program). Pour l'exécuter, on doit créer le Spark Context, il est comme la porte pour gérer tous les traitements de Spark. La tâche sera envoyée au Cluster Manager par Spark Context. Ensuite, le Cluster Manager divisera la tâche en des nombreux parties petites et les distribuera aux workers. Enfin, la tâche des workers sera effectuée et les résultats seront renvoyés au Spark Context.

3.3.2. RDD

Pour la définition, RDD signifie Resilient Distributed Dataset, une structure de données de Spark. Lorsqu'un ensemble de données est inséré dans RDD, il sera décomposé en plusieurs parties, puis, ces parties seront stockées dans la mémoire des workers. RDD fournit deux opérations : les transformations et les actions. Les transformations transforment ce RDD en un autre RDD, et les actions calculent, traitent des données dans RDD et renvoient le résultat au Spark Context (Figure 3-6).

[6] https://www.edureka.co/blog/spark-architecture/

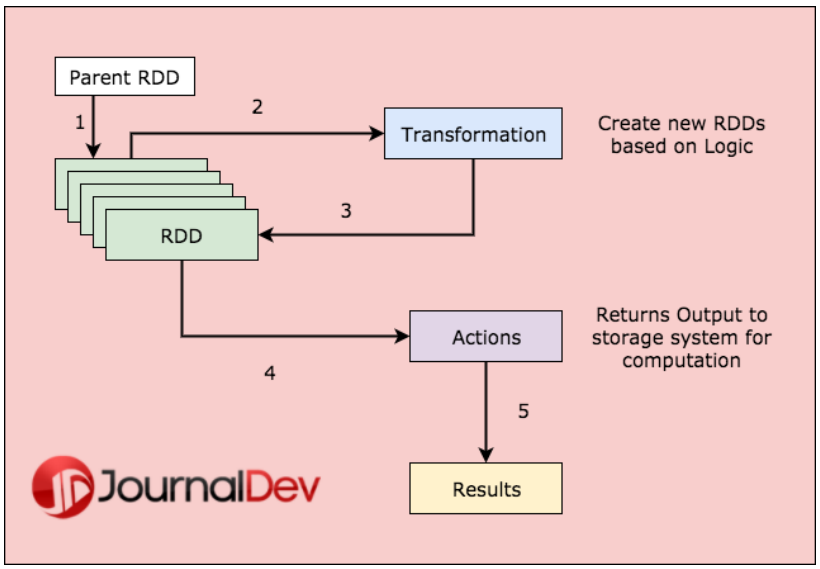

Figure 3-6 : Les transformations et les actions dans RDD[7]

Selon la Figure 3-6, la transformation convertira les données sur les RDD et renverra les résultats avec l'action. Par exemple, pour calculer la moyenne d'un tableau d'entiers sur RDD, nous pouvons le faire comme le code ci-dessous :

```
In [1]:
1  # charger fichier txt au RDD
2  listNumbers = sc.textFile('numbers.txt')
3  # transformation: déviser données dans RDD listNumbers au liste de numéros
4  rdd_1 = listNumbers.flatMap(lambda x: x.split(',')[:])
5  # transformation: créer liste de tuple avec format: [(valeur, 1)]
6  rdd_2 = rdd_1.map(lambda number: (int(number), 1))
7  # action: créer tuple (sum(valeur), sum(1))
8  rdd_3 = rdd_2.reduce(lambda n1, n2: (n1[0]+n2[0], n1[1]+n2[1]))
9  # sum(valeur)/sum(1)
10 mean = rdd_3[0]/rdd_3[1]
11 # resultat
12 mean

Out[1]: 52.07692307692308
```

Tout d'abord, la séquence de nombres du fichier *numbers.txt* sera chargée au RDD *listNumbers* sous forme de chaîne de caractères (ligne 2). Ensuite, *listNumbers* subdivise cette chaîne en un tableau de *string* éléments et l'enregistre dans *rdd_1* (ligne 4). Ensuite, *rdd_1* convertit ces éléments en *int* et mappe chaque élément sous la forme *(valeur, 1)* et enregistre dans *rdd_2* (ligne 6). Puis, *rdd_2* fait la somme de tous les éléments et enregistre dans *rdd_3* (ligne 8). Maintenant, *rdd_3* est une variable de la forme *(somme(valeur), somme(1))*. Enfin, pour calculer la moyenne, nous fenons le premier élément de *rdd_3* divisée par le deuxième

[7] https://www.journaldev.com/20261/apache-spark

élément (ligne 10) et imprimer le résultat (ligne 12). Grâce à l'exemple ci-dessus, nous pouvons voir que Spark effectue une série des transformations avec l'intention de convertir les données de *listNumbers* en *rdd_1* puis *rdd_2* et enfin exécuter une action pour renvoyer le résultat enregistré à *rdd_3*.

3.3.3. DAG

DAG signifie Direct Acyclic Graph, l'une des deux l'abstraction importantes de Spark, consiste à une série de calculs des données sur RDD. Dans DAG, chaque nœud est une partition RDD, le bord est les transformations de données. Les deux propriétés Direct et Acyclic de Graph disent que le graphe ne peut pas revenir à la partition RDD passée récemment (Acyclic) et chaque transformation dans le DAG doit être une conversion directe de l'état des données (Direct). Le code suivant décrit le problème WordCount respectant DAG :

```
1  # charger fichier txt au RDD
2  text = sc.textFile('word-count-text.txt', 4)
3  # transformation :déviser le document dans RDD text au liste de mots
4  rdd_1 = text.flatMap(lambda x: x.split())
5  # transformation: créer liste de tuple avec format: [(mot, 1)]
6  rdd_2 = rdd_1.map(lambda x: (x,1))
7  # transformation: combiner les tuples ayant même clé
8  rdd_3 = rdd_2.reduceByKey(lambda x1, x2: x1+x2)
9  # action: convertir RDD de resultat au liste
10 rdd_3.collect()
```

Tout d'abord, le texte sera chargé dans le RDD *text* (ligne 2). Ensuite, le *text* coupera les mots de ce paragraphe (ligne 4) et les enregistrera dans *rdd_1*. Ensuite, *rdd_1* convertira chaque élément sous la forme *(mot, 1)* et sauvegardera dans *rdd_2* (ligne 6). Maintenant, *rdd_2* enregistre une liste d'éléments sous la forme *(mot, 1)* et le "mot" qui joue le rôle de clés. Ensuite, *rdd_2* combinera les mêmes clés et enregistrera les résultats dans *rdd_3* (ligne 8). Enfin, nous obtenons le résultat de *rdd_3* à la ligne 10.

Dans l'exemple ci-dessus, les nœuds sont : *text, rdd_1, rdd_2, rdd_3* et les bords sont les transformations : *flatMap, map, reductionByKey*. De plus, les bords transforment l'ancien RDD en un nouveau RDD, sans revenir au travail sur les anciens RDD (Acyclic). Enfin, après chaque transformation, la structure de l'ensemble de données est changée (Direct).

3.3.4. Le concept de programmation sur Spark

La programmation dans le RDD de Spark demande les programmeurs de penser différemment que la programmation traditionnelle. En particulier, dans les problèmes de calcul, la solution doit consister à un ensemble de calculs commutatifs et combinables afin que Spark puisse diviser ces calculs pour les workers puis combiner les résultats de workers afin de donner le résultat final. Par exemple, le problème du calcul du moyenne d'une séquence d'entiers peut être résolu simplement en faisant la somme de ce nombre et en le divisant par la longueur de la séquence, comme suit :

```
In [1]:
1  # charger string from fichier txt
2  listNumbers = sc.textFile('numbers.txt')
3  # déviser string au liste de numéros
4  arr = listNumbers.collect()[0].split(',')
5  # convertir les élements dans liste arr à int
6  arr = [int(x) for x in arr]
7  # initialiser variable sum = 0
8  sum = 0
9  # calculer sum des numéros dans liste arr
10 for i in range(len(arr)):
11     sum += arr[i]
12 # caculer mean par déviser sum par nombre des numéros
13 result = sum/len(arr)
14 result

Out[1]: 52.07692307692308
```

Premièrement, nous chargeons la séquence de nombres à partir du fichier *numbers.txt* (ligne 2) et divisons cette séquence en éléments, puis les enregistrons dans la liste *arr* (ligne 4). Maintenant, les éléments dans *arr* ont un type de données *string*, nous devons convertir leur type en *integer* (ligne 6). Ensuite, nous calculons la valeur moyenne en calculant la somme de ces éléments (de ligne 8 à 11) et puis, divisons la somme par le nombre d'éléments de la séquence (ligne 13) pour le résultat final.

Cependant, cette méthode ne permet pas de calculer des grandes séquences de nombres qui ne peuvent pas être enregistrés sur une machine. Nous devons donc modifier la pensée d'implémentation comme la méthode dans l'exemple de la partie 3.3.2.

4. ALGORITHME K-MEANS

Dans ce chapitre, nous présentons la théorique de l'algorithme K-means, l'idée de l'installation sur Spark, les critères de l'évaluation de la qualité de l'algorithme et quelques discussions.

4.1. K-means

4.1.1. Introduction

K-means est un algorithme d'apprentissage non supervisé (Unsupervised Learning), il n'apprend que sur les caractéristiques de l'ensemble de données, mais on ne connait pas le label des données d'entrée. En particulier, avec le nombre de clusters (K) et en supposant que chaque point de données appartient un seul groupe, l'algorithme divise l'ensemble de données en K groupes telle que les données de chaque groupe ont des attributs similaires. Selon les différents ensembles de données, il existe des méthodes différentes pour déterminer les points de proximité, la distance Euclidienne est la méthode la plus populaire. Dans ce chapitre, nous présentons deux autres mesures de distance, Cosine et Pearson.

4.1.2. Analyses mathématiques

Soit la matrice de données $\boldsymbol{X}$ composés N individus :

$$\boldsymbol{X} = \{x_1, x_2, \dots, x_n\} \in \mathbb{R}^{d \times N} \text{ et } K \text{ est le nombre de clusters } (K < N)$$

Le but de la méthode K-means est détermine un partitionnement de ces individus en K groupes, ainsi :

- Chaque individu x_i est affecté à exactement un groupe. Les affectations sont décrites par la matrice $A = \{a_1, a_2, \dots, a_n\} \in \{0,1\}^{N \times K}$, avec a_i est le vecteur d'étiquette de x_i telle que :

$$a_{ij} = \begin{cases} 1 \; si \; x \; appartient \; au \; group \; j \\ 0 \; sinon \end{cases} \qquad (4.1)$$

Par exemple, $a_i = [1,0,0, \dots ,0]$ si x_i appartient au cluster 1

- Groupe $j^{ième}$ est décrit par un vecteur m_j appartenant au même espace avec $\boldsymbol{X}$, appelé centroïde :

$$M = \{m_1, m_2, \dots, m_j, \dots, m_k\} \forall m_j, 1 < j < k: m_j \in R^d$$

4.1.2.1. Fonction distance moyenne

Suppose qu'un point de données x_i appartient au cluster k, on a $a_{ik} = 1, a_{ij} = 0 \; \forall \; j \neq k$, la distance de x_i au centroïde m_k de cluster k est $dist(x_i, m_j)$:

$$dist(x_i, m_j) = a_{ik} dist(x_i, m_j) = \sum_{j=1}^{k} a_{ij} dist(x_i, m_j) \qquad (4.2)$$

Donc, la distance moyenne entre tous les points de données et leurs centroïdes est :

$$\mathcal{L}(A, M) = \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} a_{ij} dist(x_i, m_j) \qquad (4.3)$$

L'application de l'algorithme K-means permet de diviser l'ensemble de données en clusters. Après de la mise en cluster, la distance entre un point de données et leur centroïde sera inférieure que autres centroïdes. Ainsi, la distance moyenne de tous les points de données et leurs centroïdes atteindra la valeur minimale ou la fonction (4.3) atteindra la valeur minimale. Donc, pour trouver le centroïde et le label correspondante de chaque point de données, nous devons résoudre le problème de trouver A, M de manière à ce que la valeur de (4.3) atteigne la valeur minimale :

$$A, M = \underset{A,M}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} a_{ij} dist(x_i, m_j)$$

avec $$a_{ij} \in \{0,1\} \; \forall i, j \; ; \; \sum_{j=1}^{K} a_{ij} = 1 \; \forall i \qquad (4.4)$$

4.1.2.2. Trouver la valeur minimale de la fonction de distance moyenne

Pour trouver la valeur minimale de la fonction de distance moyenne, nous devons résoudre le problème (4.4). Cependant, comme il existe 2 variables, il est difficile de trouver le point minimal. Donc, on doit résoudre alternatif A et M lorsqu'un de deux variables est fixée jusqu'à ce que la fonction converge. En particulier, on a besoin de résoudre deux problèmes suivants :

Fixer M, retrouver A

L'objectif : *Suppose qu'on ait les centroïdes, on doit rechercher des vecteurs d'étiquette afin que la fonction de distance moyenne atteigne la plus petite valeur.*

Lorsque les centroïdes est fixés, on a besoin de rechercher des vecteurs de label de toutes les données en recherchant le vecteur de label pour chaque point de données x_i :

$$a_i = \underset{a_i}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} \sum_{j=1}^{K} a_{ij} dist(x_i, m_j)$$

avec $$a_{ij} \in \{0,1\}\ \forall i,j\ ;\ \sum_{j=1}^{K} a_{ij} = 1\ \forall i \qquad (4.5)$$

Comme un seul élément de a_i a valeur de 1, donc, le problème de (4.5) est de rechercher le centroïde étant plus près de x_i :

$$j = \underset{j}{\operatorname{argmin}}\ dist(x_i, m_j) \qquad (4.6)$$

$dist(x_i, m_j)$ est la distance de x_i au centroïde m_j, on peut démontrer que chaque point x_i appartient au cluster dont le centroïde est le plus proche. Donc, il est possible de retrouver le vecteur label de chaque point de données.

Fixer A, retrouver M

L'objectif : *Suppose qu'on connaisse le label de chaque point, on a besoin de rechercher le nouveau centroïde viser à minimiser la fonction de distance moyenne*.

On a besoin de résoudre le problème ci-dessous :

$$m_j = \underset{m_j}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} a_{ij} dist(x_i, m_j) \qquad (4.7)$$

Au problème (4.7), selon la mesure, on a des méthodes différentes pour chercher m_j. Donc, on résoudra ce problème dans la partie (4.1.2.4), après la recherche des mesures communs.

4.1.2.3. Les mesures de distance populaires

Lorsqu'on applique K-means, on a besoin de faire attention aux mesures de distance parce qu'ils ont des significations et des objectifs de l'utilisation différents. Par exemple, on utilise souvent la distance Euclidienne pour calculer la distance entre deux points différents, de plus, on utilise la distance Cosine ou Pearson au lieu de la distance Euclidienne avec l'intention de calculer la similarité entre deux vecteurs à n dimensions. Le Tableau 4-1 résume la signification de la formule et l'objectif de l'utilisation de chaque mesure de distance.

Tableau 4-1 : Résumer les informations de 3 mesures de distance

La mesure de distance	**Le formule**	**L'objectif**
Euclidean Distance	$dist_{Euclidean}(p,q) = \|p-q\|_2^2$	Calculer la distance entre p et q
Cosine Distance	$dist_{Cosine}(p,q) = 1 - \frac{pq}{\|p\|\|q\|}$ p, q sont les vecteurs	Calculer la similarité entre p et q
Pearson Distance	$dist_{Pearson}(p,q) = 1 - \frac{(p-\bar{p})(q-\bar{q})}{\|p-\bar{p}\|\|q-\bar{q}\|}$ p, q sont les vecteurs $\bar{p}, \bar{q}$ sont des vecteurs moyens de p, q	Calculer la similarité linéaire entre deux vecteurs p et q

4.1.2.4. La formule pour mettre à jour le nouveau centroïde

Pour chaque mesure, on doit utiliser différentes méthodes afin de résoudre le problème (4.7).

Distance Euclidienne

Avec

$$dist_{Euclidean}(p,q) = \|p-q\|_2^2 \tag{4.8}$$

(4.7) devient

$$m_j = \underset{m_j}{\operatorname{argmin}} \frac{1}{N}\sum_{i=1}^{N} a_{ij}\left\|x_i - m_j\right\|_2^2 \tag{4.9}$$

Afin d'identifier m_j, on peut chercher la racine de la fonction en calculant d'abord le dérivé, que l'on note $\nabla m_j \mathcal{L}$, puis on résout l'équation $\nabla m_j \mathcal{L} = 0$:

$$\nabla m_j \mathcal{L}(m_j) = \nabla m_j \left(\frac{1}{N}\sum_{i=1}^{N} a_{ij}\left\|x_i - m_j\right\|_2^2\right)$$

$$= \nabla m_j \left(\frac{1}{N}\sum_{i=1}^{N} a_{ij}\left(x_i^2 - 2x_i m_j + m_j^2\right)\right) \tag{4.10}$$

$$= \frac{2}{N}\sum_{i=1}^{N} a_{ij}\left(-2x_i m_j + 2m_j\right) = 0 \tag{4.11}$$

$$\Leftrightarrow m_j \sum_{i=1}^{N} a_{ij} = \sum_{i=1}^{N} a_{ij} x_i \tag{4.12}$$

$$\Leftrightarrow m_j = \frac{\sum_{i=1}^{N} a_{ij} x_i}{\sum_{i=1}^{N} a_{ij}} \tag{4.13}$$

D'après (4.13), le nouveau centroïde du cluster est la moyenne (mean) de tous les points dans le cluster.

Distance Cosine

Avec

$$dist_{Cosine}(p,q) = 1 - \frac{pq}{\|p\|\|q\|} \tag{4.14}$$

(4.7) devient

$$m_j = \underset{m_j}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} a_{ij} \left(1 - \frac{x_i m_j}{\|x_i\| \|m_j\|}\right) \tag{4.15}$$

On examine la fonction :

$$\mathcal{L}(m_j) = \frac{1}{N} \sum_{i=1}^{N} a_{ij} \left(1 - \frac{x_i m_j}{\|x_i\| \|m_j\|}\right)$$

$$= \frac{1}{N} \left(\sum_{i=1}^{N} a_{ij} - \sum_{i=1}^{N} a_{ij} \frac{x_i}{\|x_i\|} \frac{m_j}{\|m_j\|}\right) \tag{4.16}$$

Propose

$$u = \sum_{i=1}^{N} a_{ij} \frac{x_i}{\|x_i\|} \tag{4.17}$$

(4.16) devient

$$\mathcal{L}(m_j) = \frac{1}{N} \left(\sum_{i=1}^{N} a_{ij} - \frac{u \cdot m_j}{\|m_j\|}\right) \tag{4.18}$$

En appliquant l'inégalité de Cauchy - Schwarz à u et m_j on a :

$$u \cdot m_j \leq \|u\| \|m_j\| \tag{4.19}$$

$$\Leftrightarrow \frac{u \cdot m_j}{\|m_j\|} \leq \|u\| \tag{4.20}$$

D'après (4.18), (4.20)

$$\Rightarrow \mathcal{L}(m_j) \geq \frac{1}{N} \left(\sum_{i=1}^{N} a_{ij} - \|u\|\right) \tag{4.21}$$

Donc

$$\min \mathcal{L}(m_j) = \frac{1}{N} \left(\sum_{i=1}^{N} a_{ij} - \|u\|\right) \tag{4.22}$$

Selon l'inégalité de Cauchy-Schwarz, l'équation de (4.20) apparaît lorsque le vecteur m_j et le vecteur u sont dans la même direction :

$$m_j = ku \; \forall \; k \tag{4.23}$$

On choisit $k = \frac{1}{\sum_{i=1}^{N} a_{ij}}$, d'après (4.17), (4.23) on montre que :

$$m_j = \frac{\sum_{i=1}^{N} a_{ij} \frac{x_i}{\|x_i\|}}{\sum_{i=1}^{N} a_{ij}} \tag{4.24}$$

Ainsi, avec la distance Cosine, le nouveau centroïde du cluster est la moyenne de tous les points dans le cluster, après de la normalisation au vecteur unitaire.

Distance Pearson

Avec

$$dist_{Pearson}(p,q) = 1 - \frac{(p-\bar{p})(q-\bar{q})}{\|p-\bar{p}\|\|q-\bar{q}\|} \tag{4.25}$$

(4.7) devient

$$m_j = \underset{m_j}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} a_{ij} \left[1 - \frac{(x_i-\bar{x_\iota})(m_j-\overline{m_J})}{\|x_i-\bar{x_\iota}\|\|m_j-\overline{m_J}\|}\right] \tag{4.26}$$

On examine la fonction :

$$\mathcal{L}(m_j) = \frac{1}{N} \sum_{i=1}^{N} a_{ij} \left[1 - \frac{(x_i-\bar{x_\iota})(m_j-\overline{m_J})}{\|x_i-\bar{x_\iota}\|\|m_j-\overline{m_J}\|}\right]$$

$$= \frac{1}{N} \left[\sum_{i=1}^{N} a_{ij} - \sum_{i=1}^{N} a_{ij} \frac{(x_i-\bar{x_\iota})}{\|x_i-\bar{x_\iota}\|} \frac{(m_j-\overline{m_J})}{\|m_j-\overline{m_J}\|}\right] \tag{4.27}$$

Propose

$$u = \sum_{i=1}^{N} a_{ij} \frac{(x_i-\bar{x_\iota})}{\|x_i-\bar{x_\iota}\|} \tag{4.28}$$

$$v = m_j - \overline{m_J} \tag{4.29}$$

(4.27) devient

$$\mathcal{L}(m_j) = \frac{1}{N} \left(\sum_{i=1}^{N} a_{ij} - \frac{u \cdot v}{\|v\|}\right) \tag{4.30}$$

En appliquant l'inégalité de Cauchy - Schwarz à u et v, on a :

$$u \cdot v \leq \|u\|\|v\| \tag{4.31}$$

$$\Leftrightarrow \frac{u \cdot v}{\|v\|} \leq \|u\| \tag{4.32}$$

D'après (4.30), (4.32), on peut montrer que :

$$\Rightarrow \mathcal{L}(m_j) \geq \frac{1}{N}\left(\sum_{i=1}^{N} a_{ij} - \|u\|\right) \tag{4.33}$$

Donc

$$\min \mathcal{L}(m_j) = \frac{1}{N}\left(\sum_{i=1}^{N} a_{ij} - \|u\|\right) \tag{4.34}$$

Selon l'inégalité de Cauchy-Schwarz, l'équation de (4.32) apparaît lorsque le vecteur v et le vecteur u sont dans la même direction :

$$v = ku \; \forall \, k \tag{4.35}$$

On choisit $k = \frac{1}{\sum_{i=1}^{N} a_{ij}}$, d'après (4.29), (4.35), on a :

$$m_j - \overline{m_j} = \frac{1}{\sum_{i=1}^{N} a_{ij}} u \tag{4.36}$$

$$\Leftrightarrow \begin{cases} m_{j1} - \frac{m_{j1} + m_{j2} + \ldots + m_{jd}}{d} = \frac{1}{\sum_{i=1}^{N} a_{ij}} u_1 \\ \vdots \\ m_{jd} - \frac{m_{j1} + m_{j2} + \ldots + m_{jd}}{d} = \frac{1}{\sum_{i=1}^{N} a_{ij}} u_d \end{cases} \tag{4.37}$$

(4.37) est un système d'équations linéaires ayant une infinité de solutions. Donc, on a besoin de rechercher seulement une solution pour (4.36). D'abord, on examine :

$$u = \sum_{i=1}^{N} a_{ij} \frac{(x_i - \bar{x}_i)}{\|x_i - \bar{x}_i\|} \tag{4.28}$$

u est un vecteur de d dimensions, donc, on peut décrire par :

$$\begin{cases} u_1 = \sum_{i=1}^{N} a_{ij} \frac{(x_{i1} - \overline{x_{\iota 1}})}{\|x_i - \bar{x_\iota}\|} \\ \vdots \\ u_d = \sum_{i=1}^{N} a_{ij} \frac{(x_{id} - \overline{x_{\iota d}})}{\|x_i - \bar{x_\iota}\|} \end{cases} \quad (4.38)$$

$$\Longrightarrow \sum_{k=1}^{d} u_k = \sum_{i=1}^{N} a_{ij} \frac{(\sum_{k=1}^{d} x_{ik} - \sum_{k=1}^{d} \overline{x_{\iota k}})}{\|x_i - \bar{x_\iota}\|} \quad (4.39)$$

$$\Leftrightarrow \sum_{k=1}^{d} u_k = \sum_{i=1}^{N} a_{ij} \frac{(\sum_{k=1}^{d} x_{ik} - \frac{\sum_{k=1}^{d} x_{ik}}{d} d)}{\|x_i - \bar{x_\iota}\|} \quad (4.40)$$

$$\Leftrightarrow \sum_{k=1}^{d} u_k = \sum_{i=1}^{N} a_{ij} \frac{(\sum_{k=1}^{d} x_{ik} - \sum_{k=1}^{d} x_{ik})}{\|x_i - \bar{x_\iota}\|} \quad (4.41)$$

$$\Leftrightarrow \sum_{k=1}^{d} u_k = 0 \quad (4.42)$$

$$\Leftrightarrow \sum_{k=1}^{d} \frac{u_k}{d} = 0 \quad (4.43)$$

$$\Leftrightarrow \bar{u} = \vec{0} \quad (4.44)$$

D'après (4.36), (4.44), on a :

$$m_j - \overline{m_j} = \frac{1}{\sum_{i=1}^{N} a_{ij}} u - \frac{1}{\sum_{i=1}^{N} a_{ij}} \bar{u} \quad (4.45)$$

avec

$$m_j = \frac{1}{\sum_{i=1}^{N} a_{ij}} u \quad (4.46)$$

(4.45) devient

$$\frac{1}{\sum_{i=1}^{N} a_{ij}} u - \frac{1}{\sum_{i=1}^{N} a_{ij}} \bar{u} = \frac{1}{\sum_{i=1}^{N} a_{ij}} u - \frac{1}{\sum_{i=1}^{N} a_{ij}} \bar{u} \quad (4.47)$$

$$\Leftrightarrow 0 = 0 \text{ (toujours vrai)} \quad (4.48)$$

Donc

$$m_j = \frac{1}{\sum_{i=1}^{N} a_{ij}} u \quad (4.46)$$

Est une solution de (4.36). D'après (4.36), (4.46), on a :

$$m_j = \frac{\sum_{i=1}^{N} a_{ij} \frac{(x_i - \overline{x_l})}{\|x_i - \overline{x_l}\|}}{\sum_{i=1}^{N} a_{ij}} \quad (4.49)$$

Ainsi, le nouveau centroïde du cluster est la moyenne de tous des points dans le cluster, après de normalisation de ces points à la moyenne nulle et à l'écart de type d'unité.

La formule permettant de trouver des centroïdes correspondant à chaque mesure est résumée au Tableau 4-2 ci-dessous.

Tableau 4-2 : Les formules pour déterminer le centroïde de chaque mesure

La mesure de distance	**Le Formule**
Euclidean Distance	$m_j = \frac{\sum_{i=1}^{N} a_{ij} x_i}{\sum_{i=1}^{N} a_{ij}}$
Cosine Distance	$m_j = \frac{\sum_{i=1}^{N} a_{ij} \frac{x_i}{\|x_i\|}}{\sum_{i=1}^{N} a_{ij}}$
Pearson Distance	$m_j = \frac{\sum_{i=1}^{N} a_{ij} \frac{(x_i - \overline{x_l})}{\|x_i - \overline{x_l}\|}}{\sum_{i=1}^{N} a_{ij}}$

4.1.3. Résume de l'algorithme

D'après des analyses mathématiques, on peut résumer l'algorithme K-means comme suivant :

- Entrée : Matrice de données $\boldsymbol{X} \in \mathbb{R}^{d \times N}$ et nombre de clusters $K < N$
- Sortie : Matrice de centroïde $\boldsymbol{M} \in \mathbb{R}^{d \times N}$ et matrice de label $\boldsymbol{Y} \in \mathbb{R}^{N \times K}$
  - Choisir aléatoirement K points dans le training set $\boldsymbol{X}$ comme les centroïdes initiaux
  - Répéter
  - Assigner chaque point au cluster dont il est le plus proche au son centroïde

- Recalculer le centre de chaque cluster et modifier le centroïde
- Jusqu'à convergence (les centroïdes ne changent plus lors des itérations)

Pour comprendre le flux de K-means, on peut le résumer en visualisant au diagramme de workflow ci-dessous (Figure 4-1).

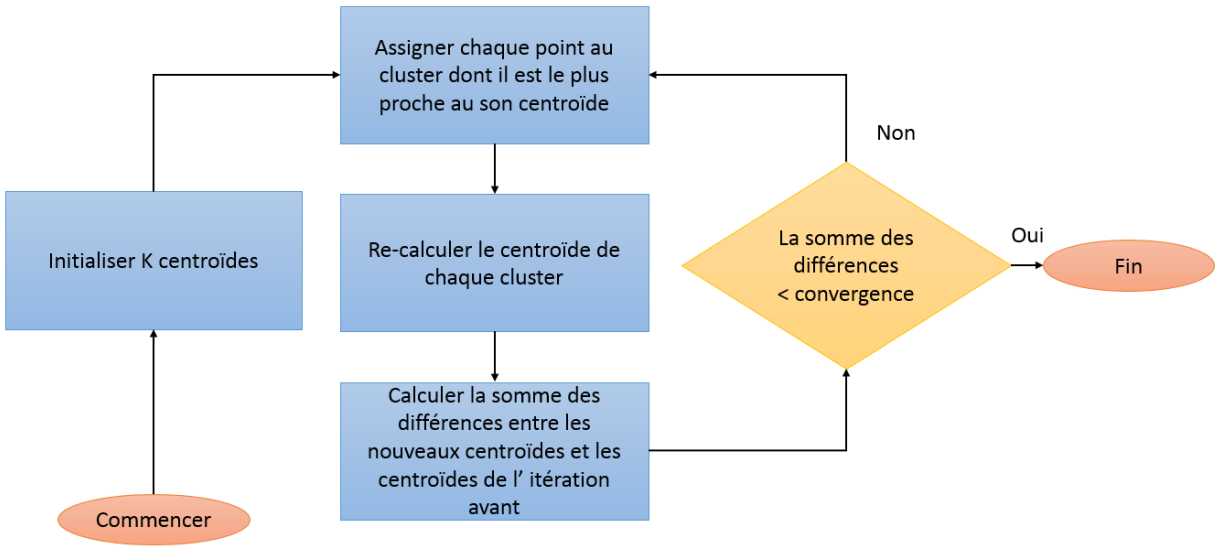

Figure 4-1 : Le flux d'activité de K-means

Quand on recherche les nouveaux centroïdes, s'il y a un cluster qui ne contient aucun point, le dénominateur de (4.13), (4.24) et (4.49) est égal à zéro, la division est irréalisable. Donc, K points dans le training set $\boldsymbol{X}$ étant choisis comme les centroïdes initiaux viser à assurer que chaque cluster contient au moins un point.

4.2. K-means en basant MapReduce

4.2.1. K-means sur Big data

L'algorithme K-means regroupe les points de données en fonction de la distance (Distance Euclidienne) ou la similarité (Distance Cosine, Distance Pearson) entre les points du training set. Donc, il doit avoir un ensemble de données complet pour les rassembler efficacement. Mais, il y a des grands ensembles de données de sorte qu'on ne puisse pas de stocker ou de

traitement sur en seul ordinateur (via la partie 3.1). Lorsque les données sont divisées afin de stocker sur les machines différentes, est-ce qu'on peut exécuter K-means sur chaque sous-machine et puis rassemble leurs résultats (Figure 4-2) ?

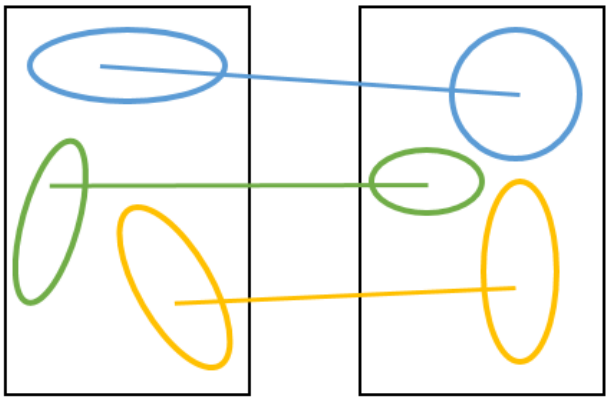

Figure 4-2 : Le processus d'assemblage des résultats de regroupement sur les sous-machines

D'après de Figure 4-2, on peut voir que les clusters bleus dans la machine 1 et 2 n'ont aucune relation car ils ont des centroïdes différents. Donc, on ne peut pas appliquer K-means sur les sous-machines et enfin combiner les clusters dans ces machines.

Le problème ci-dessus nécessite d'autres algorithmes viser à fonctionner bien sur la plate-forme Big Data

4.2.2. Algorithmes de clustering sur Big data

Pour répondre aux besoins de travailler avec Big Data, nous présentons trois algorithmes de clustering communs : BFR, CURE et K-means *en basant* MapReduce.

4.2.2.1. Algorithme BFR

BFR signifie Bradley-Fayyad-Reina, le nom de trois scientifiques qui ont inventé cet algorithme. Le BFR a été développé sur la base de l'algorithme K-means en supposant que les points de données sont distribués autour du centre du cluster dans l'espace euclidien et que les clusters ont la forme ellipse aligné sur l'axe. L'algorithme charge chaque partie de l'ensemble de données à la mémoire, les traite et enregistre les valeurs statistiques et les

données nécessaires, puis procède de la même manière avec le reste des données. BFR place les données dans mémoire en trois types suivants :

- Discard set : ensemble de points sont suffisamment proches du centroïde. Ces points seront résumés par des valeurs statistiques et ne seront plus conservés dans la mémoire.
- Compression set : ensemble de points situe près des autres, mais ne situe pas à proximité des aucun centroïde. Ces points seront résumés, mais ne seront pas affectés aux clusters.
- Retained set : les points sporadiques attendent d'être affecté à la compression set.

Ces trois ensembles de données peuvent être visualisés en Figure 4-3 ci-dessous :

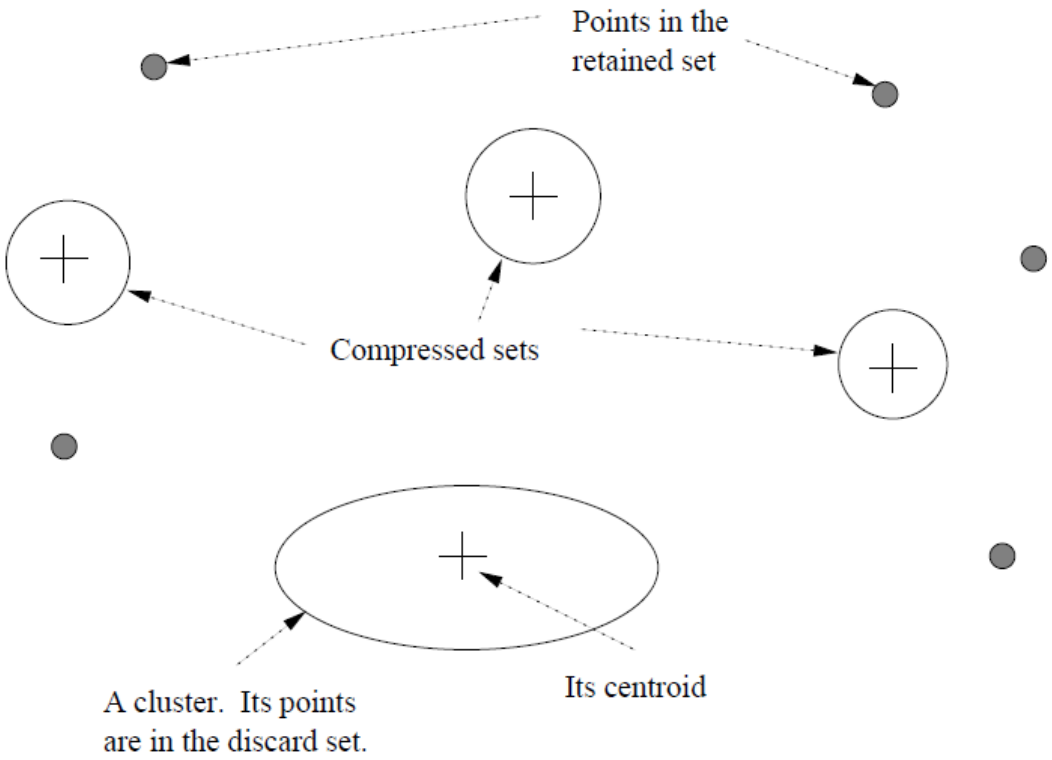

Figure 4-3 : Trois sets de données dans BFR [12]

Pour Discard set et Compression set, les valeurs doivent être résumées et enregistrées :

- Nombre N de points de données dans le set.
- Le vecteur SUM avec le $i^{ème}$ composant stocke la somme des points de données dans la $i^{ème}$ dimension.

- Le vecteur $SUMQ$, est similaire à SUM mais enregistre la somme des carrés de points de données dans chaque dimension.

L'algorithme BFR effectue un regroupement en chargeant chaque partie de l'ensemble de données dans la mémoire. Lors du premier chargement, BFR prend aléatoirement K points de l'ensemble de données chercher à l'initialisation des centroïdes. Ensuite, les points sont suffisamment proches (*) de centroïde seront affectés à ce cluster et Discard set du cluster. Puis, les points de données restants et les points de Retained set dans la charge précédente seront regroupés localement par un algorithme de clustering (par exemple, K-means). Après de la mise en cluster, les points de données dans les clusters sont attribués à Compression set, les points sporadiques sont attribués à Retained set. Ensuite, nous mettons à jour les statistiques dans Discard Set. Enfin, nous considérons la combinaison de Compression Set (**). Toutefois, s'il s'agit du dernier chargement, nous combinerons tous les Compression set et affecterons les points de Retained set au cluster ayant le centroïde le plus proche.

- (*) : La distance de Mahalanobis entre le point de données et le centroïde est inférieure à un certain seuil
- (**) : Calculer la variance après la combinaison de 2 Compression sets. Si la variance est inférieure à un certain seuil, nous pouvons les combiner.

4.2.2.2. Algorithme CURE

CURE (Clustering Using REpresentative) est également un algorithme de clustering pouvant fonctionner sur plusieurs machines. L'algorithme fonctionne également sur l'espace euclidien et permet de créer les clusters en n'importe quelle forme. Bien que BFR, K-means, etc. utilise le centroïde, CURE utilise un ensemble de points pour représenter un cluster. Cette représentation aide CURE à diviser l'ensemble d'entraînement en les clusters de toute forme.

Le processus de regroupement de CURE passe par 2 phases. Première phase, on charge une partie de l'ensemble d'entraînement dans la mémoire. Ensuite, on utilise un algorithme de clustering hiérarchique en vue de regrouper ces données. Après la mise en cluster, dans chaque cluster, on choisit un ensemble de points représentatifs tels que ces points situeraient plus éloignés que possible. Ensuite, avec des points dans le groupe de représentants, on déplace ces points de 20% du côté du centroïde. Enfin, à la deuxième phase, il suffit de parcourir

chaque point de données restant dans l'ensemble d'entraînement et d'affecter ces points au cluster ayant le point représentant le plus proche.

4.2.2.3. K-means en basant MapReduce

K-means en basant MapReduce est similaire à K-means traditionnel, mais les étapes de calculer sont transformé pour pouvoir fonctionner sur le Big data, il est basé sur le concept de MapReduce. Maintenant, nous présentons l'algorithme K-means Baseline proposant par les spécialistes d'Apache Spark. Il résout deux problèmes : assigner des points de données au cluster ayant le centroïde le plus proche et mettre à jour le nouveau centroïde du cluster.

Premièrement, nous choisissons aléatoirement K points dans l'ensemble de données comme des centroïdes initiaux des clusters. Ensuite, nous envoyons cet ensemble des centroïdes à tous les sous-machines. Sur les sous-machines, nous regroupons les points de données en basant de l'ensemble des centroïdes envoyés. Après de la mise en cluster, nous allons effectuer Map(), chaque point de données est représenté sous forme (clé, valeur), avec *clé* est le cluster dont ce point appartient, et *valeur* se compose de set (p, 1), avec *p* est le point de donnés, 1 est le nombre d'élément *p* dans le cluster. Ensuite, nous procédons Reduce() en fonction de *clé* de la liste que nous venons de créer. En particulier, pour les éléments avec la même *clé* (même cluster), la fonction effectuera la somme de la valeur (p, 1) selon la formule correspondant à chaque mesure de distance. Ainsi, après l'étape Reduce(), l'ensemble de données a été réduit à l'un ensemble de K éléments sous la forme $(clusterID, (\sum transformed(p),\ cluster\_size))$. À ce stade, nous pouvons trouver le nouveau centroïde en réalisant Map(): pour chaque $clusterID$, le nouveau centroïde est $(\sum transformed(p)/cluster\_size)$. Enfin, s'il y a un nouvel ensemble de centroïdes, nous calculons la distance totale entre les nouveaux et les actuels centroïdes. Si cette valeur est inférieure au niveau de convergence, nous allons arrêter l'algorithme, sinon nous allons répéter le processus ci-dessus.

En bref, le regroupement des points de données de K-means Baseline est effectuée localement sur chaque machine en fonction de l'ensemble de centroïde reçus. Ensuite, l'ensemble de données passe par les étapes Map(), Reduce() afin de créer des nouveaux ensembles de centroïdes. La structure de l'ensemble de données à travers chaque étape est décrite ci-dessous (Figure 4-4).

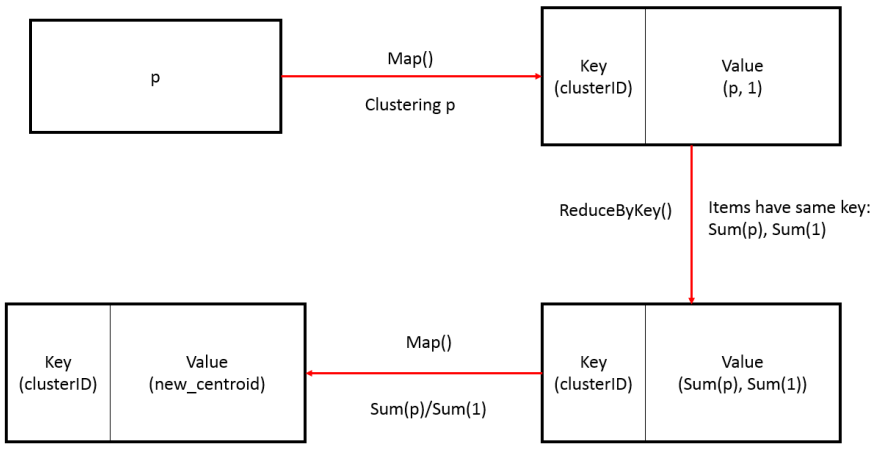

Figure 4-4 : Le flux de la transformation de données dans K-means Based MapReduce

4.3. Les méthodes d'évaluation de la qualité du regroupement

L'évaluation de la qualité du regroupement est un problème compliqué, car l'algorithme de clustering est une méthode d'apprentissage non supervisée. L'ensemble de données n'a pas de label, donc on ne peut pas vérifier des preuves vraies ou fausses des résultats. Cependant, il y a des nombreux articles proposant des méthodes pour évaluer la qualité du regroupement. Ces méthodes sont divisées en 2 groupes principaux : l'évaluation interne et l'évaluation externe.

4.3.1. Les méthodes de l'évaluation interne

La méthode de l'évaluation interne est utilisée pour évaluer les résultats du regroupement en basant uniquement sur la structure des clusters, il n'intéresse pas aux informations externes. Par conséquent, ces méthodes évaluent la qualité en fonction de l'objectif général du clustering, elles ne basent pas sur les caractéristiques de l'ensemble de données ou l'efficacité de la résolution du problème.

L'objectif de l'algorithme du regroupement est d'assigner les points de données en groupes afin que les points d'un même groupe soient suffisamment proches et que la distance entre les groupes soit très éloignée. Cet objectif permet que les points de données d'un même groupe aient des caractéristiques similaires et soient différents des points de données d'autres groupes. Les méthodes d'évaluation interne basent sur deux facteurs : la distance entre les points d'un

cluster et la distance entre les clusters. Dans ce mémoire, on introduit deux méthodes populaires : Sum of Squared Errors (SSE) et la distance Silhouette [13].

4.3.1.1. Sum of Squared Errors (SSE)

Il s'agit d'une méthode d'évaluation de la qualité des points dans un même cluster en calculant la somme des carrés des distances entre les points de données et leur centroïde. Sa formule :

$$SSE(X, A, M) = \sum_{j=1}^{K} \sum_{i=1}^{N} \left\| a_{ij} x_i - m_j \right\|^2 \quad (4.50)$$

Avec :

- $X \in \mathbb{R}^{d \times N}$: la matrice d'entrainement
- $A \in \mathbb{R}^{N \times K}$: la matrice d'entrainement
- $M \in \mathbb{R}^{d \times N}$: la matrice des centroïdes

Selon cette méthode, l'indice SSE de l'ensemble de données sera élevé si la distance entre les points du cluster avec le centroïde est grande. L'objectif de l'algorithme de clustering est que la distance des points d'un même cluster doit être petite. On peut donc conclure que l'indice de SSE est plus petit, la qualité du regroupement est meilleure.

Toutefois, cette méthode a deux problèmes. Premièrement, la formule de calcul de SSE ne convient que pour la distance Euclidien, car elle calcule la somme de carré de la distance entre les points et le centre du cluster. Si l'on utilise la distance Cosine ou la distance Pearson, SSE ne sera pas évalué correctement. On propose donc une méthode d'évaluation similaire à SSE mais, elle adapte à toutes les mesures de distance : Cost. Le Cost d'un résultat du regroupement est calculé en faisant la somme de la distance totale entre les points et le centroïde des clusters. Pourtant, cette distance sera calculée en fonction des mesures différentes lors de l'exécution de l'algorithme K-means. On peut exprimer cette méthode comme :

$$Cost(X, A, M) = \sum_{j=1}^{K} \sum_{i=1}^{N} a_{ij} dist(x_i, m_j) \quad (4.51)$$

avec X, A, M similaires à (4.50). Le deuxième problème existe à la fois en SSE et en Cost, ces deux méthodes évaluent uniquement en fonction de la distance entre les points d'un cluster

mais ignorent la distance entre les clusters. Ce problème est résolu dans la méthode d'évaluation prochaine : la distance Silhouette.

4.3.1.2. Distance Silhouette

La distance Silhouette est une méthode d'évaluation plus complète que le SSE et le Cost parce qu'elle concerne à la fois la similarité des points de données dans un cluster et la différence entre les clusters. Donc, il nécessite des étapes de calcul plus compliqué. Afin de calculer la distance Silhouette de toutes les données ($S(X)$), on doit calculer la distance Silhouette de chaque point de données du cluster ($s(i)$), puis calculer la moyenne :

$$S(X) = \frac{\sum_{i=1}^{N} s(i)}{N} \tag{4.52}$$

avec :

$$s(i) = \frac{q(i)-p(i)}{\max\{p(i),q(i)\}} \tag{4.53}$$

avec :

$$p(i) = \frac{\sum_{j \in C_i, j \neq i} d(i,j)}{|C_i|-1} \tag{4.54}$$

est la distance moyenne du point de données i à tous les autres points dans son cluster

$$q(i) = \min_{i \neq j} \frac{1}{|C_j|} \sum_{j \in C_j} d(i,j) \tag{4.55}$$

est la distance minimale du point i à tous les points des autres groupes.

En particulier

- C_i : le cluster contient du point de données i
- $|C_i|$: le nombre de points dans le cluster C_i
- C_j : le cluster contient du point de données j
- $|C_j|$: le nombre de points dans le clusterC_j
- $d(i,j)$: la distance entre 2 points de données i et j

D'après (4.53), $s(i) \in [-1,1]$, avec $s(i)$ s'approche de 1, il exprime $p(i) \ll q(i)$, ou la distance moyenne entre les points dans C_i est plus petite que la distance de i au cluster le plus proche. Cela indique que le point de données i est assigné correctement. Inversement, si $s(i)$ s'approche de -1, cela indique un résultat de regroupement incorrect, car la distance moyenne entre i et les points dans C_i est supérieure à la distance de i et au groupe le plus proche. On peut donc conclure qu'un grand $S(X)$ signifie une bonne qualité du regroupement et inversement.

4.3.2. Les méthodes de l'évaluation externe

Les méthodes de l'évaluation externe ne s'intéressent pas aux informations internes du cluster, mais, elles se concentrent des facteurs externes tels que les informations obtenues à partir de clusters de données (caractéristiques du cluster) ou niveau de résolution de problèmes. Il existe de nombreuses méthodes d'évaluation externe, dans lesquelles il y a 2 indices populaires pour évaluer la qualité des informations obtenues à partir de groupe de données. : l'Entropy et le Purity [14][15]. Dans cette partie, on suppose que la caractéristique d'un cluster est la caractéristique ayant la plus grande valeur de ce cluster.

4.3.2.1. L'indice Entropy

L'Entropy est une mesure déterminant l'incertitude d'une variable aléatoire, cette incertitude est liée à la probabilité d'occurrence de cette variable [16]. L'indice d'Entropy élevé signifie que la probabilité d'occurrence d'événements est presque égale. Inversement, si l'indice Entropy est faible, chaque événement aura une probabilité d'occurrence différente.

Dans le problème du clustering, on souhaite que chaque cluster ait une caractéristique spéciale, c'est-à-dire, lorsqu'on faire les statistiques en basant les caractéristiques des points de donnée, il doit exister une caractéristique supérieure aux autres. Cette supériorité rendra la probabilité d'occurrence d'événements ne soient plus uniforme et l'indice d'Entropy sera faible. Au contraire, si l'indice d'Entropy est élevé, le cluster ne présente pas de caractéristiques remarquables. Ainsi, on peut conclure qu'un faible indice d'Entropy présente de bon résultat du regroupement, et inversement.

Par exemple, le Figure 4-5 recense les caractéristiques du cluster après le regroupement en 6 clusters. Dans chaque cluster, on peut voir qu'il existe une colonne avec une valeur supérieure,

il s'agit de la caractéristique du cluster. De plus, le troisième cluster a l'indice d'Entropie est inférieur au quatrième parce que la caractéristique du troisième cluster est « Sports » ayant la valeur est supérieure aux autres caractéristiques dans ce cluster, alors que la caractéristique « Financial » du quatrième n'est pas remarquable.

Cluster	Entertainment	Financial	Foreign	Metro	National	Sports	Entropy	Purity
1	3	5	40	506	96	27	1.2270	0.7474
2	4	7	280	29	39	2	1.1472	0.7756
3	1	1	1	7	4	671	0.1813	0.9796
4	10	162	3	119	73	2	1.7487	0.4390
5	331	22	5	70	13	23	1.3976	0.7134
6	5	358	12	212	48	13	1.5523	0.5525
Total	354	555	341	943	273	738	1.1450	0.7203

Figure 4-5 : L'exemple des résultats statistiques en fonction des caractéristiques des données [15]

Viser à calculer l'Entropy de tous les clusters, on doit d'abord calculer l'Entropy de chaque cluster. L'indice d'Entropy e_j de la cluster j est calculé selon la formule :

$$e_j = -\sum_{i=1}^{d} p_{ij} \log_2 p_{ij} \tag{4.56}$$

d est le nombre de caractéristiques du cluster (qui est également la dimension du vecteur de caractéristique des données), et p_{ij} est la probabilité qu'un élément du cluster j présente la caractéristique i et est calculée selon la formule :

$$p_{ij} = \frac{m_{ij}}{m_j} \tag{4.57}$$

m_{ij} est le nombre de la caractéristique i dans le cluster j et m_j est le nombre totale de toutes les caractéristique du cluster j. L'indice d'Entropy de tous les clusters sera calculée selon la formule :

$$e = \sum_{j=1}^{K} \frac{m_j}{m} e_j \tag{4.58}$$

avec

- K : le nombre de clusters
- m_j : la taille de cluster j
- m : la taille de l'ensemble de données

4.3.2.2. L'indice Purity

Dans la méthode d'évaluation externe, l'indice de Purity est également utilisé afin de comparer la corrélation entre la valeur élevée et les autres valeurs dans le cluster. Afin de trouver facilement la caractéristique du groupe, on a besoin de caractéristique plus élevé de ce groupe. Si la valeur de cette fonctionnalité est remarquable, il est plus facile en vue de la découvrir. Et, l'indice de Purity est la méthode pour évaluer la « supériorité » de cette caractéristique.

Par exemple, selon Figure 4-5, on peut facilement reconnaître que « Sports » est la caractéristique du troisième cluster car sa valeur est supérieure aux autres caractéristiques du groupe, donc, son Purity est très élevée. En revanche, la caractéristique du quatrième cluster est « Financial », mais sa valeur est similaire aux valeurs d'autres caractéristiques, donc, Son indice de Purity est très faible.

Le Purity de tous les clusters est calculée selon la formule :

$$purity = \sum_{j=1}^{K} \frac{m_j}{m} purity_j \tag{4.59}$$

avec $purity_j$ est l'indice Purity de cluster j et est calculée par

$$purity_j = \max(p_{ij}) \tag{4.60}$$

4.4. Discussions

D'après d'avoir étudié l'algorithme K-means, on constate qu'outre les avantages, K-means a aussi des problèmes à améliorer. Premièrement, parlons des avantages de K-means, on peut voir que cet algorithme a une idée simple et facile à installer. De plus, l'idée de regroupement de K-means peut être facilement modifiée pour exécuter parallèlement si l'ensemble de données est trop volumineux. En outre, il est possible d'appliquer un certain nombre de

mesures de distance différentes de sorte que K-means puisse traiter sur plusieurs ensembles de données ayant des caractéristiques distinctes. Cependant, K-means a encore certaines des inconvénients. Premièrement, l'algorithme nécessite d'indiquer le nombre de clusters spécifiques. Deuxièmement, le résultat de K-means dépend des K points d'initialisation aléatoires.

Dans les inconvénients ci-dessus, la sélection du nombre approprié de cluster pose le plus de difficultés, car la détermination de K est la première condition pour que l'algorithme fonctionne. Une solution de Robert L. Thorndike en 1953 consiste à appliquer la méthode du « coude ». Cette méthode est basée sur la méthode d'évaluation interne (4.3.1). Plus précisément, on va exécuter K-means en plusieurs fois avec les valeurs différentes de K. À chaque exécution, on enregistre le SSE. Ensuite, on représente l'ensemble de données (K, SSE) sur le graphique. On constate qu'avec la petite valeur, l'augmentation de K fera diminuer considérablement le SSE, mais jusqu'à une valeur certaine de K, l'augmentation de K diminue négligeable le SSE. Cette K est la valeur K appropriée pour l'ensemble d'entrainement (Figure 4-6).

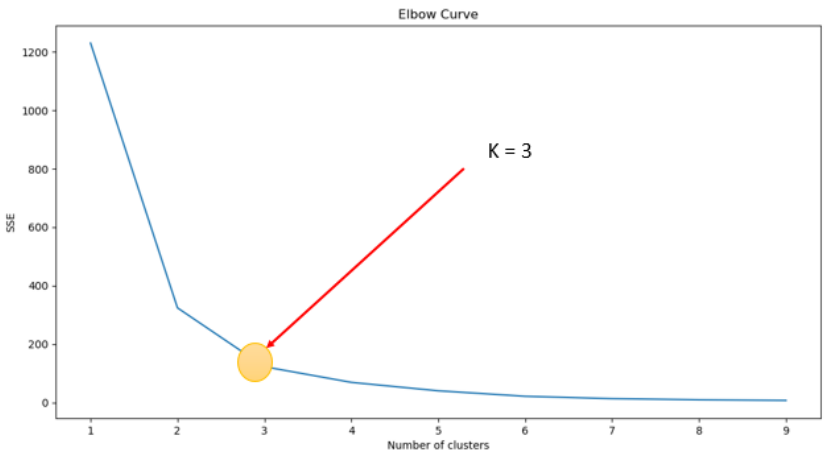

Figure 4-6 : La méthode « coude »

Du côté de SSE, on peut utiliser les autres indices d'évaluation, tels que le Cost ou la distance Silhouette avec la même manière. Cependant, lors de l'application de la méthode « coude » à la distance Silhouette (S), On choisit K à la position tel qu'en augmentant K, S n'augmente plus (la forme du "coude" sera l'inverse lors de l'application pour le SSE ou le $Cost$).

5. MISE EN ŒUVRE DE THESE

La thèse est réalisée en cinq étapes (Figure 5-1) :

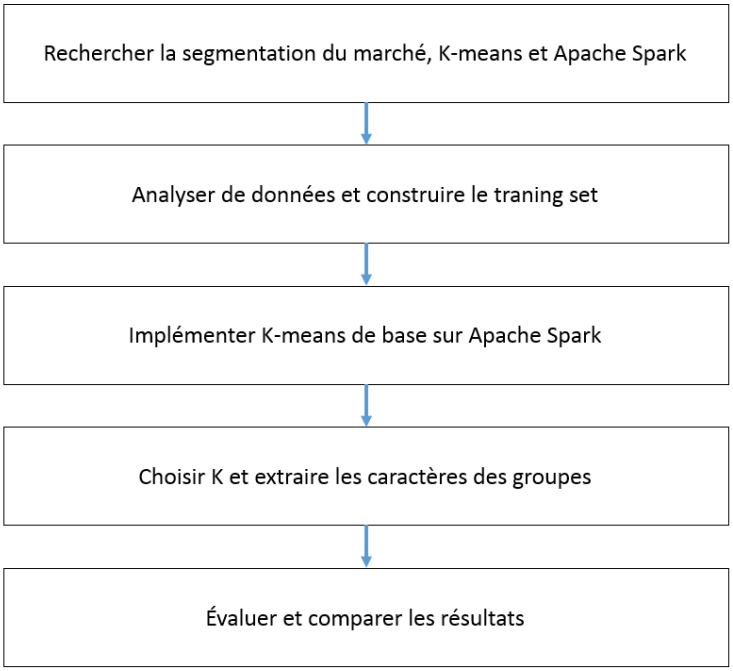

Figure 5-1 : Les étapes de la réalisation de la thèse

5.1. L'étape 1 : Rechercher la segmentation du marché, l'algorithme K-means et la plate-forme du Big data Apache Spark

Dans cette étape, on a recherché l'objectif, les avantages du marketing pour les entreprises. En fait, on a constaté qu'outre la segmentation du marché, les entreprises appliquent également des autres méthodes du marketing, lesquelles la méthode la plus commun est le marketing de masse. On a donc l'appris afin de comparer ses avantages et ses inconvénients à segmentation du marché.

Après la recherche du concept et de l'objectif de la segmentation du marché, on a recherché les manières dont les entreprises s'appliquent avec l'intention de mettre en œuvre cette stratégie. En particulier, la méthode principale repose sur l'historique des achats des clients. À partir de l'historique des transactions de tous les clients, ils sélectionneront des critères spécifiques et diviseront la liste de clients en des groupes. Selon cette méthode, les clients du même groupe auront des similitudes liant aux critères sélectionnés pour la subdivision. Enfin,

en fonction de chaque groupe de clients, les entreprises devront donner des stratégies de marketing distinctes étant correspondant aux caractéristiques du groupe. Après de la découverte, on a constaté que le problème de la division des clients en basant de leurs caractéristiques est similaire au regroupement de données dans le domaine de la Computer Science.

Clustering est une méthode d'apprentissage non supervisé (unsupervised learning). Bien qu'on ne connaisse pas le label des données d'entrée avant, mais, on peut les traiter par leur structure et leur caractéristique. La tâche de l'algorithme de clustering consiste à diviser l'ensemble de données X en petits groupes (cluster) en fonction de l'association entre les données du groupe les données du même cluster ont les mêmes propriétés, mais les caractéristiques entre ces groupes doivent être plus différentes que possible. Ceci est similaire au processus de segmentation du marché, les entreprises ont besoin également d'analyser et de diviser l'ensemble de données de client en groupes différentes. Sur la base d'un certain critère, les clients ayant des caractéristiques similaires seront placés dans même groupe, depuis, on a les stratégies marketing appropriées pour chaque groupe d'utilisateurs. Pour la visualisation de l'ensemble de clients avec chaque client est un point du graphique (Figure 5-2)

Figure 5-2 : L'ensemble de client avant mise en cluster

La segmentation du marché est le processus de création de bordures viser à diviser l'ensemble de clients en groupes plus petits (Figure 5-3). L'algorithme de clustering est l'une des méthodes permettant de résoudre à cet objectif [5]. Par conséquent, le problème de

segmentation du marché peut convertir en problème de regroupement en fonction de l'histoire d'achat des clients.

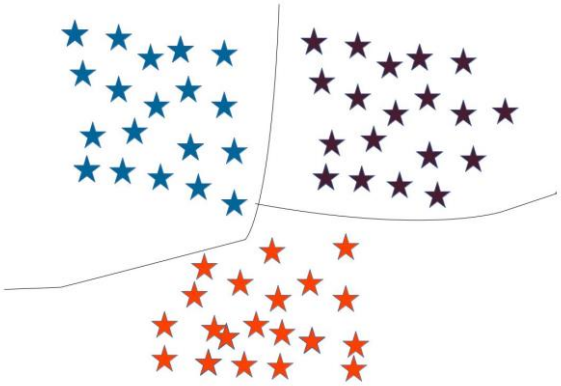

Figure 5-3 : L'ensemble de client après mise en cluster

Nous souhaitons de résoudre da segmentation du marché sur le Big data. Nous avons recherché les framework qui nous supporte à résoudre ce problème. Finalement, nous avons décidé d'utiliser Apache Spark pour implémenter l'algorithme. Nous avons exploré le concept et les caractéristiques de Spark, ainsi que leurs abstractions importantes tels que RDD et DAG. De plus, nous savons également que Spark fournit la bibliothèque MLlib afin de résoudre les problèmes d'apprentissage automatique. Donc, nous avons étudié spécifiquement MLlib et son algorithme K-means. En particulier, nous avons appris à utiliser PySpark dans le but de mettre les données du fichier .csv au dataframe de Spark, puis nous avons exécuté la fonction d'entrainement de K-means et avons analysé le résultat renvoyé.

Nous avons commencé à apprendre les algorithmes de clustering. Dans ce groupe d'algorithmes, K-means est l'algorithme le plus fondamental et le plus populaire. Nous avons donc décidé d'étudier l'algorithme K-means de manière à résoudre notre problème. Nous avons approché le K-means en analysant sa mathématique, nous avons démontré que le regroupement et la mise à jour de centroïde du cluster sont le processus de résolution de la recherche minimale pour la fonction de distance moyenne de K-means. De plus, nous avons appris que K-means supporte aux nombreuses mesures différentes, mais la plupart des documents se concentrent uniquement sur la distance Euclidien. Donc, nous devons trouver la formule dans l'intention de mette à jour le nouveau centroïde de deux autres mesures : la distance Cosine et la distance Pearson. Après d'avoir compris l'analyse mathématique et les

étapes de l'algorithme, nous avons essayé d'implémenter l'algorithme en langage Python à l'aide de la bibliothèque numpy. Puis, nous créons un ensemble d'entrainement pour exécuter K-means dans un espace à 2 dimensions et visualisons afin de vérifier les résultats du regroupement.

Après d'avoir compris K-means, nous souhaitons l'appliquer sur le Big data. Cependant, nous réalisons que l'algorithme K-means normal ne peut pas être appliqué en vue de résoudre le problème de clustering sur des données volumineuses. Donc, nous avons étudié les autres algorithmes pouvant fonctionner sur la plate-forme de Big data comme BFR, CURE et K-means Baseline (chapitre 4.2). En recherchant simultanée des trois algorithmes, nous montrons que K-means Baseline peut être implémenter plus facile que deux autres algorithmes. De plus, il base également sur le K-means traditionnel, l'algorithme que nous avons étudié bien. Ainsi, nous avons décidé d'utiliser l'algorithme K-means Baseline pour l'entrainement.

5.2. L'étape 2 : Analyse de données et construis le training set

À partir de l'ensemble de données d'Amadeus Entertainment sous la forme d'une base de données SQLServer, on transforme les tables de la base de données en fichiers .csv et recherche la signification de chaque fichier .csv dans 3 dossiers : Jade, Jupyter et WebTower. Cet ensemble de données enregistre les informations des clients, des produits, des commandes, des magasins et des informations supplémentaires telles que les taux de change quotidiens. Les trois dossiers contiennent un total de 34 fichiers .csv, chaque fichier contient des informations différentes. Après d'avoir les compris, on utilise les fichiers .csv suivants avec l'intention de créer l'ensemble d'entraînement :

- customer.csv (customer_number, account_number, customer_type, name, gender, email_address, date_of_birth, address1, address2, address3, address4, city, state, zipcode, country, phone_number, occupation, household_income, date_registered, status, permission, preferred_channel1, preferred_channel2, interest1, interest2, interest3, created, last_updated): y compris 54,158 lignes, chaque ligne contient les informations d'un client. Toutefois, les informations personnelles telles que l'adresse,

le courrier électronique et la carrière ont été modifiées afin de protéger la confidentialité des clients.

- order_header.csv (order_id, order_date, customer_number, store_number, currency, created, last_updated) : y compris 44,425 lignes, chaque ligne contient des informations générales sur une commande telles que la date de la transaction, le client, le code du magasin, ...
- order_detail.csv (order_id, line_no, product_code, qty, price, unit_cost) : y compris 63778 lignes. Chaque ligne contient des informations sur un produit acheté par une commande. Par conséquent, si la commande a 2 produits, il y aura 2 lignes avec la même "order_id" et "line_no" qui seront respectivement 1 et 2. Ce fichier et order_header.csv fournissent des informations complètes sur la commande.
- product_category.csv (product_category, description) : compose de 3 lignes contenant le nom et la description des 3 catégories de produits : "Music", "Audio books" et "Films"
- product_type.csv (product_type_code, product_type, product_category) : inclut 65 lignes enregistrant des informations sur 65 types de produits avec sa catégorie.
- product.csv (product_code, name, description, title, artist_code, product_type_code, format, unit_price, unit_cost, status, created, last_updated) : compose 2,458 lignes, chaque ligne stocke les informations d'un produit, chaque produit appartient à un type de produit stockant dans "product_type.csv"

À partir des fichiers .csv ci-dessus, nous avons construit le fichier .csv pour créer l’ensemble d’entrainement (Figure 5-4).

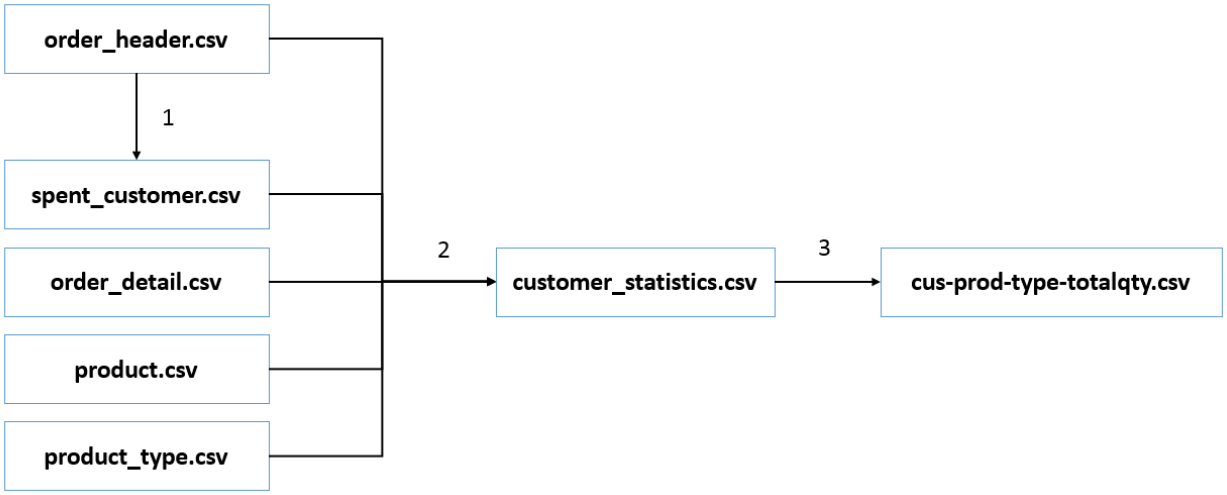

Figure 5-4 : La création d'un ensemble d'entrainement

Nous utilisons le SQL Query pour créer le fichier spent_customer.csv (customer_number) contenant le code des clients dont leurs transactions sont stockées dans order_header.csv. Ensuite, nous continuons à agréger les informations de order_header.csv, order_detail.csv, spent_customer.csv, product.csv et product_type.csv en vue de générer le fichier customer_statistics.csv (customer_number, product_type_code, total_qty, top_qty, avg_qty, total_times, total_spent, avg_spent) : pour chaque client, il y a 65 lignes correspondant à 65 types de produits dans product_type.csv. Chaque ligne contient les informations de transaction de ce client pour le type de produit correspondant (par exemple, la ligne (10000, AA, ...) contient les informations sur la transaction du client ayant le code "10000" avec les produits de type "AA"). Les informations stockées sur chaque ligne incluent : le nombre total de produits achetés, le nombre le plus élevé de produits achetés en une fois, le moyen de produits achetés, le nombre total d'achats, le montant total et le montant moyen des transactions. Enfin, avec les données de customer_statistics.csv, on utilise PySpark dans le but des agréger dans un ensemble de données cus-prod-type-totalqty.csv : chaque ligne contient les informations de transaction d'un client, et chaque colonne représente le nombre total de produits que le client a achetés correspondant au type de produits de cette colonne. Leur structure est visualisée à la Figure 5-5 ci-dessous :

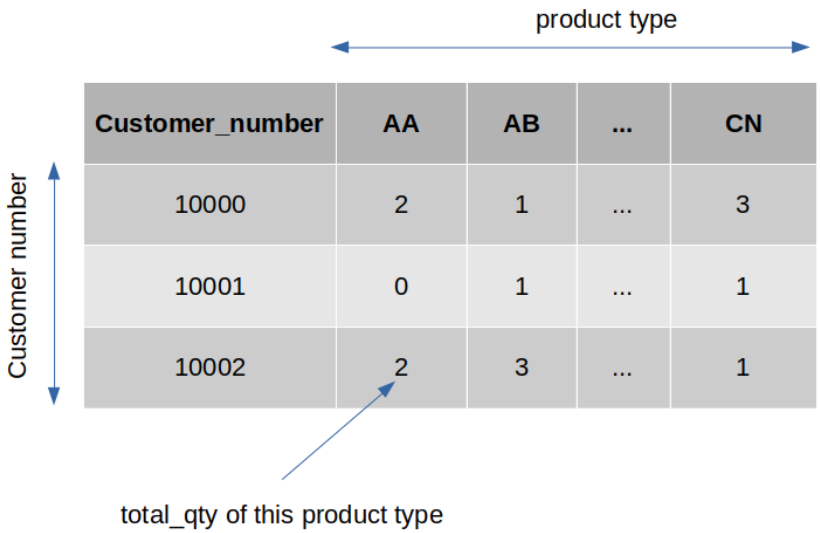

Figure 5-5 : La structure de cus-prod-type-totalqty.csv

5.3. L'étape 3 : Implémenter K-means sur Apache Spark

Dans cette étape, nous étudions l'algorithme K-means fournissant par MLlib pour savoir les fonctions à installer, ainsi que les paramètres d'entrée et les résultats renvoyés. Après, nous avons implémenter l'algorithme sur Spark avec les fonctions du calcul de la qualité de regroupement.

L'algorithme K-means Baseline peut être résumé comme :

1. Choisir aléatoirement K points chercher à l'initialisation des centroïdes du cluster.
2. Commencer la boucle
3. Assigner chaque point de données p au cluster ayant le centroïde le plus proche.
4. Dans chaque cluster, calculer la somme de valeur des points de données et le nombre d'éléments
5. Dans chaque cluster, affecter le nouveau centroïde du cluster en divisant la valeur totale par le nombre d'éléments.
6. Calculez la somme de distance entre le nouveau centroïde et l'ancien centroïde du cluster.
7. Si la distance totale dans l'étape 6 est supérieure à la valeur de convergence, revenir à l'étape 3. Sinon, arrêter l'algorithme.

Liste des fonctions :

- *cosine_similarity(vector1, vector2)* : Calculer la valeur de Cosine similarity de 2 vecteurs
- *pearson_similarity(vector1, vector2)* : Calculer la valeur de pearson similarity de 2 vector
- *closestPoint(p, centers, distanceMeasure)* : Trouver le cluster dont le centroïde est le plus proche du point p en basant de chaque mesure.
- *error(p, kPoints, distanceMeasure)* : renvoyer la distance entre le point p et le centroïde du cluster le plus proche en fonction de chaque mesure
- *MyKmeans()* : Contenir les fonctions et les paramètres pour prétraiter et entraîner l'ensemble de données en l'algorithme K-means en basant de MapReduce
    - *K* : le nombre de cluster
    - *tol* : la valeur de convergence (K-means s'arrête lorsque la distance totale entre l'ancien et le nouveau centroïde du cluster est inférieure à cette valeur)
    - *maxIter* : le nombre maximum d'itérations de l'algorithme
    - *seed* : la graine est utilisée viser à choisir aléatoirement K points étant les centroïdes des cluster.
    - *distanceMeasure* : la mesure (la valeur défaut est la distance Cosin - 'cosinus', nous supportons également 2 autres mesures : la distance euclidien - 'euclidienne' et la distance Pearson - 'pearson')
    - Le fonction *transform(df, feaureStartIndex=1)* : recevoir un dataframe et renvoyer l'ensemble d'entrainement à partir de *featureStartIndex*^{ème} colonne de ce dataframe.
    - Le fonction *initKPoints(data)* : initialiser K centroïdes
    - Le fonction *train(df, featureStartIndex=1)* : entraîner le dataframe d'entrée et renvoyer l'ensemble des K centroïdes
    - Le fonction *predict(df, kPoints, featureStartIndex=1)* : identifier les étiquettes pour l'ensemble d'entrainement
    - Le fonction *summarize(result_df)* : à partir du résultat du fonction *Predict*(), synthétiser les informations (la taille de l'ensemble d'entrainement, la taille de

chaque cluster, les valeurs des caractéristiques des données de chaque cluster) afin de supporter aux fonctions de l'évaluation de la qualité du clustering.

- Le fonction *computeCost(df, kPoints)* : calculer l'indice *Cost*
- Le fonction *computeEntropy(count, sum_count, sum_set)* : calculer l'indice *Entropy*
- Le fonction *computePurity(count, sum_count, sum_set)* : calculer l'indice *Purity*

5.4. L'étape 4 : Sélectionner le numéro de cluster approprié pour K-means et extraire la fonctionnalité de cluster.

Après de l'étape 4, nous avons cus-prod-type-totalqty.csv, l'ensemble d'entraînement. Cependant, on ne sait pas de nombre de clusters choisir approprié, on utilise donc la méthode du "coude" pour le choisir. En particulier, on a exécuté la méthode K-means avec la distance Cosine (car la similarité des points de données n'est exprimée pas par leur position, donc on ne peut pas utiliser la distance Euclidien) et la valeur K comprises de 2 à 1000, puis on a calculé le Cost, la distance Silhouette afin de trouver la valeur K appropriée. On peut visualiser le résultat en le graphe. L'axe de x est la valeur de K, le Cost est la ligne verte et la distance Silhouette est la ligne rouge (Figure 5-6)

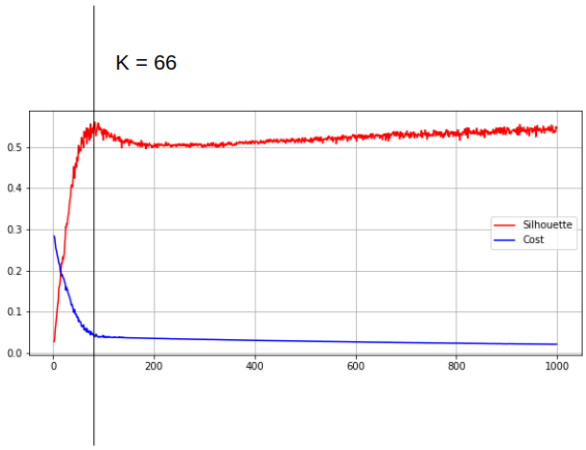

Figure 5-6 : L'utilisation de la méthode "coude" pour sélectionner K

Lorsqu'on a sélectionné le nombre approprié, on a exécuté K-means avec K = 66. Ensuite, pour chaque cluster, on a synthétisé les informations afin d'extraire les caractéristiques. Le résultat est représenté par un graphique à colonnes empilées, chaque colonne contient des informations à partir d'un cluster et chaque élément de la colonne représente le nombre total

de produits achetés par les clients de ce cluster. Ainsi, avec 66 clusters, nous aurons 66 colonnes (étiquetées de 0 à 65) et chaque colonne contiendra 65 éléments correspondant à 65 types de produits (Figure 5-7)

Figure 5-7 : Le résultat de l'algorithme de clustering

On peut constater que chaque colonne ne compose que 1 à 2 couleurs montrées clairement sur 65, ce qui indique que les clients de chaque cluster ne s'intéressent qu'à un ou deux types de produits. C'est également la caractéristique de cluster. Depuis, on peut donner des stratégies appropriées pour ces groups telles que donner des nouveaux produits ou donner des promotions dans les types de produit que les clients s'intéressent. En outre, il existe des groupes de clients qui dépensent moins que les autres groupes (par exemple, les groupes 7, 9, 19, 21, etc.). Pour ces groupes, les entreprises doivent également définir des stratégies de marketing spécifique afin de stimuler les besoins commerciaux des clients. En revanche, les clients des groupes 20, 38 et 45 dépensent beaucoup pour des produits. Donc, les entreprises doivent donner les stratégies dans le but de conserver les clients de ces groupes.

Ainsi, le problème de segmentation du marché basé sur l'algorithme de clustering a été résolu. En fait, cette solution comporte quelques inconvénients. En particulier, le modèle dans le mémoire segmente uniquement les clients en basant de l'histoire de leurs transactions, mais en réalité, pour segmenter le marché, les spécialistes de marketing doivent examiner des nombreux critères différents de sorte que les segments atteignent la meilleure qualité. Donc, le mémoire n'a proposé qu'une méthode simple afin de segmenter le marché en basant d'un algorithme de clustering de l'apprentissage automatique, mais viser à l'appliquer à la réalité, le modèle doit améliorer de nombreux autres facteurs.

5.5. L'étape 5 : Évaluer et comparer les résultats

Dans cette étape, nous avons évalué la qualité du regroupement K-means Baseline en basant des méthodes d'évaluation interne (Cost) et des méthodes d'évaluation externe (Entropy, Purity). En particulier, lorsqu'on entraîne l'ensemble de données avec le nombre K = 66 en K-means Baseline, l'indice Cost est 6928,852498069664, l'indice Entropy est 2.1554520746635935 et l'indice Purity est 0.5759110764624777. D'autre part, nous avons comparé la qualité du clustering en utilisant la distance Cosine entre K-means Baseline et K-means founissant par la bibliothèque MLlib de Spark, et la qualité du regroupement entre la distance Cosine et la distance Pearson sur K-means Baseline (chapitre 6).

6. EXPERIMENTATION

Dans ce chapitre, on effectue deux expérimentations : comparer la qualité de notre algorithme avec K-means fournissant par MLlib d'Apache Spark et comparer la qualité de regroupement des deux mesures de distance : la distance Cosine et la distance Pearson. Les expérimentations reposent sur des méthodes d'évaluation de la qualité du regroupement dans l'intention de déterminer qu'avec le même ensemble de données et le même nombre de clusters, lequel algorithmes et laquelle mesure donnera de meilleurs résultats. L'ensemble de données utilisé pour l'expérimentation est cus-prod-type-totalqty.csv mentionnant en 5.4 et 5.5.

6.1. L'expérimentation 1 : Comparer la qualité de K-means Baseline avec K-means fournissant par MLlib d'Apache Spark

L'objectif de cette expérimentation est de comparer la qualité de clustering de l'algorithme K-means Baseline avec K-means fournissant par la bibliothèque MLlib d'Apache Spark. On évalue la qualité du clustering en basant du Cost - méthode d'évaluation interne et de l'Entropy, du Purity – les méthodes d'évaluation externe (chapitre 4.3). Pour faire la comparaison, on entraîne l'ensemble de données cus-prod-type-totalqty.csv avec deux algorithmes K-means avec la distance Cosine, mêmee valeur K et autres paramètres. La valeur de K comprise de 2 à 100, à chaque entraînement, on a enregistré les valeurs de Cost, d'Entropy et de Purity des deux algorithmes dans trois fichiers : cost-stats-top100.csv, entropy-stats- top100.csv et purity-stats-top100.csv. Enfin, on représente les indices ci-dessus sur le graphique afin de comparer la qualité de clustering des deux algorithmes (Figure 6-1).

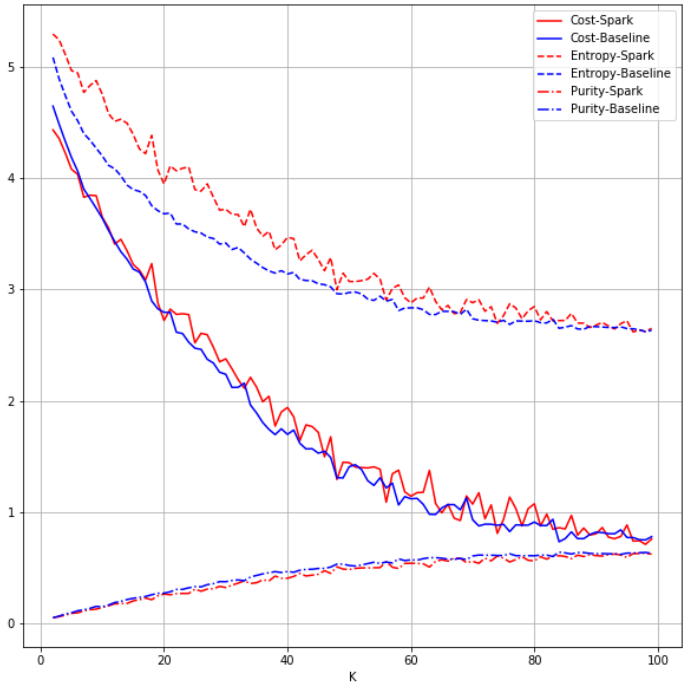

Figure 6-1 : La qualité du regroupement entre K-mean de Spark et K-mean Baseline

D'après la Figure 6-1, on peut voir que les deux algorithmes donnent une qualité similaire, mais l'algorithme K-means Baseline est un peu meilleur. Plus précisément, l'indice d'Entropy de la K-means Baseline est inférieur à la K-means de Spark avec la valeur K de 2 à 50. D'autre part, avec K > 50, l'algorithmes K-means Baseline ont une entropie inférieure à K-means de Spark, mais la différence négligeable. Ensuite, comme l'indice Entropy, les indices de Cost et de Purity montrent que les résultats du clustering de K-means Baseline sont meilleurs que K-means de Spark, mais, la différence entre les deux algorithmes n'est pas grande. Ainsi, le résultat d'expérimentation 1 garantit la qualité de regroupement de l'algorithme K-means Baseline en comparant à l'algorithme K-means fournissant par MLlib de Spark.

6.2. L'expérimentation 2 : Comparer la qualité de regroupement entre la distance Cosine et la distance Pearson.

D'après (4.1.2.3), on présente trois mesures communes dans K-means : la distance Euclidienne, la distance Cosine et la distance Pearson. Avec la distance Euclidienne, il s'agit d'une mesure basée sur la position des points de données. Il n'est donc pas approprié de mesurer la similarité entre deux points de données. Elle n'est donc pas adaptable à notre ensemble d'entrainement. Donc, deux mesures restantes sont appropriées car elles mesurent

la similarité de chaque composante dans le vecteur de points. Dans cette expérimentation, on compare la qualité de regroupement de K-means Baseline en utilisant de ces mesures. Comme l'indice Cost est calculé différemment pour chaque mesure, on ne l'utilise pas afin d'évaluer la qualité du regroupement. Donc, on utilise les méthodes d'évaluation externe, Entropy et Purity, pour évaluer la qualité du regroupement des deux mesures ci-dessus. En raison d'avoir l'indice d'Entropy et de Purity de la distance Cosine dans l'expérimentation 1, on a exécuté seulement notre algorithme avec la distance Pearson et la valeur K de 2 à 100. À chaque entraînement, on a enregistré les valeurs d'Entropy et de Purity sur le fichier mykmeans-pearson-top100.csv. Enfin, on utilise les données du nouveau fichier et du fichier entropy-stats-top100.csv et de purity-stats-top100.csv dans la première expérimentation en vue de visualiser les indices Entropy, Purity des deux mesures Cosine et Pearson (Figure 6-2).

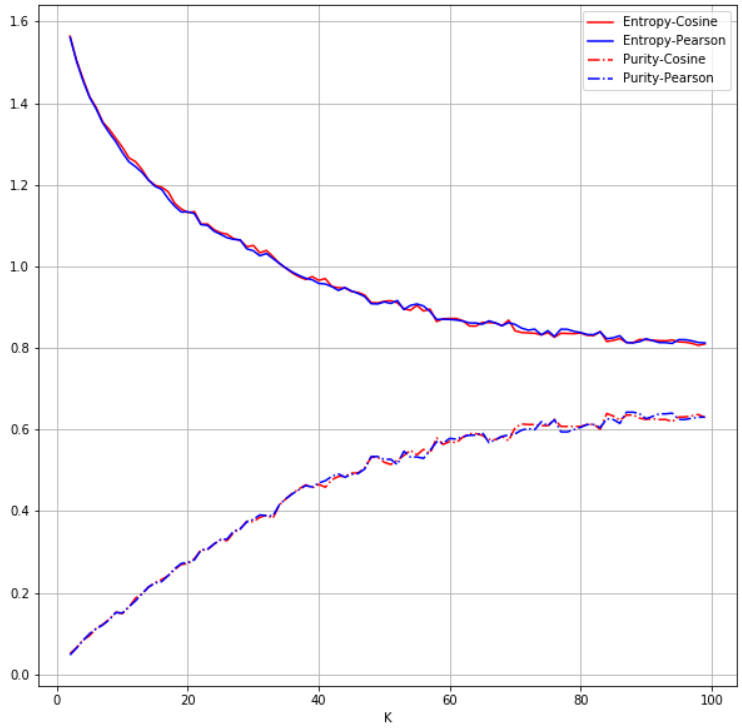

Figure 6-2 : La qualité du regroupement entre deux mesures : Cosine et Pearson

Selon la Figure 6-2, on peut facilement montrer que la qualité du clustering entre ces deux mesures est presque égale. Donc, on a choisi la distance Cosine car leur formule de calcul est plus simple que la distance Pearson, ainsi, on peut économiser le temps d'entrainement.

7. CONCLUSION

Au cours de la mise en œuvre du mémoire, nous avons recherché le concept de segmentation du marché ainsi que les méthodes courantes pour segmenter le marché. En même temps, nous avons également étudié l'algorithme de clustering K-means et la plate-forme du Big data Apache Spark. En outre, nous avons également installé avec succès l'algorithme K-means Baseline sur Spark et l'avons utilisé afin de regrouper l'ensemble de données contenant des informations de l'histoire des achats des clients d'Amadeus Entertainment. Après d'utilisation de K-means Baseline dans le but de mettre en cluster et extraire le caractère de chaque cluster, nous avons résolu le problème de la segmentation du marché avec l'algorithme de clustering K-means sur la plate-forme Apache Spark. Chaque groupe a des caractéristiques claires lorsque l'indice de Purity atteint 0,5759110764624777. Parallèlement, avec deux expérimentations, nous avons également démontré l'efficacité de l'algorithme K-means Baseline et la raison d'utiliser la distance Cosine lors de l'exécution de K-means.

Le mémoire a résolu avec succès le problème de l'automatisation de la segmentation du marché grâce à l'algorithme de clustering de l'apprentissage automatique. De plus, l'installation d'algorithmes sur Spark nous aide à entraîner des grands ensembles de données. Cependant, le mémoire comporte encore de nombreux des inconvénients. Premièrement, bien qu'il existe de nombreux algorithmes de clustering, nous n'utilisons qu'un seul algorithme K-means, il n'est donc pas possible de comparer la qualité du regroupement de K-means aux autres. En outre, la comparaison des résultats de regroupement en basant uniquement sur un seul ensemble d'entrainement donne également des résultats moins précis. Deuxièmement, le mémoire base d'un critère unique, l'histoire des transactions des clients, pour segmenter le marché. En fait, nous devons appliquer simultanément plusieurs critères (informations personnelles, psychologie, géographie, etc.) de sorte que les résultats de la segmentation soient plus efficaces.

Pour surmonter ces limitations, nous allons segmenter le marché en fonction de nombreux autres algorithmes de clustering tels que K Nearest Neighbor, Hierarchical Clustering, etc ... afin de trouver l'algorithme le plus efficace. Dans le même temps, nous comparerons sur de nombreux ensembles d'entraînement différents pour obtenir des résultats de comparaison plus

corrects. En outre, nous intégrerons d'autres caractéristiques aux clients dans l'ensemble d'entraînement afin d'accroître l'efficacité du problème de la segmentation du marché.

Grâce au mémoire, nous avons acquis de nombreuses nouvelles connaissances et compétences. Avant, nous n'avons aucune connaissance d'apprentissage automatique et de données volumineuses, mais, maintenant nous avons appris de nombreuses nouveaux méthodes et algorithmes. Spécialement, c'est le concept de la programmation parallèle, pour pouvoir travailler sur les données volumineuses. Nous savons également mener des expérimentations viser à vérifier les résultats, savons appliquer le l'apprentissage automatique à la résolution de problèmes pratiques et connaissons le processus de mise en œuvre d'un sujet de recherche. Le mémoire nous facilite de travailler avec les données réelles et d'améliorer des compétences en analyse de données. De plus, la thèse nous aide également à savoir de résoudre des problèmes pratiques sous le point de vue d'un Data Scientist. En outre, au cours de la réalisation, nous avons également acquis une certaine quantité de connaissances dans le domaine du marketing. Nous avons également appris à rédiger une thèse de recherche et avons amélioré les techniques de recherche et de lecture d'autres articles de recherche. Les connaissances, les expériences et les compétences à partir de la thèse constituent un atout dans l'intention de devenir un Data Scientist.

BIBLIOGRAPHIES

[1] Efraim Turban, David King, Jae Kyu Lee, Ting-Peng Liang, Deborrah C. Turban, « ElectronicCommerce A Managerial and Social Networks Perspective », 8th Edition, p.415.

[2] « Mass Marketing Definition Explanation with Examples », le 14 mai 2018, « https://www.marketingtutor.net/mass-marketing/ ».

[3] Smriti Chand, « Advantages and Disadvantages of Market Segmentation », « http://www.yourarticlelibrary.com/marketing/marketing-management/advantages-and-disadvantages-of-market-segmentation/27955 ».

[4] G. M. Downs and J. M. Barnard, « Hierarchical and non-Hierarchical Clustering », le 1er Decembre 1995, Daylight EUROMUG meeting, GlaxoWellcome, Stevenage UK.

[5] Kimberly Coffey, « k-means Clustering for Customer Segmentation: A Practical Exam », 13 août 2016, « http://www.kimberlycoffey.com/blog/2016/8/k-means-clustering-for-customer-segmentation ».

[6] Grzegorz Maciejewski, Sylwia Mokrysz et Łukasz Wróblewski, « Segmentation of Coffee Consumers Using Sustainable Values: Cluster Analysis on the Polish Coffee Market », Sustainability 2019, 11, 613.

[7] Matthias Carnein et Heike Trautmann, « Customer Segmentation Based on Transactional Data Using Stream Clustering », University of Münster, Germany.

[8] Kishana R. Kashwan, Member, IACSIT, and C. M. Velu, « Customer Segmentation Using Clustering and Data Mining Techniques », International Journal of Computer Theory and Engineering, Vol. 5, No. 6, December 2013.

[9] Shreya Tripathi, Aditya Bhardwaj, Poovammal E, « Approaches to Clustering in Customer Segmentation », International Journal of Engineering &Technology, 7 (3.12) (2018) 802 -807.

[10] Btissam Zerhari, Ayoub Ait Lahcen, Salma Mouline1, « Big Data Clustering: Algorithms and Challenges ».

[11] A. Sherin, S. Uma, K.Saranya and M. Saranya Vani, « Survey On Big Data Mining Platforms, Algorithms And Challenges ». International Journal of Computer Science & Engineering Technology, Vol. 5 No, 2014

[12] Jure Leskovec, Anand Rajaraman, Jeffrey D. Ullman, « Mining of Massive Datasets », p.259.

[13] Tippaya Thinsungnoena, Nuntawut Kaoungkub, Pongsakorn Durongdumronchaib, Kittisak Kerdprasopb, Nittaya Kerdprasopb, « The Clustering Validity with Silhouette and Sum of Squared Errors », Proceedings of the 3rd International Conference on Industrial Application Engineering 2015.

[14] Ying Zhao and George Karypis, « Criterion Functions for Document Clustering Experiments and Analysis », February 21, 2002, chapitre 4.2.

[15] « Cluster Validation », « http://www.cs.kent.edu/~jin/DM08/ClusterValidation.pdf », p22

[16] Jorge R.Vergara and Pablo A.Estesvez, « A-review-of-feature-selection-methods-based-on-mutual-information », Neural Comput & Applic, Vol.24, 2014, pp175-186.