
# BDA

## Rapport : Recommending Music and the Audioscrobbler Data Set

Rapport
Version 1.0, 12.06.2020

Samuel Torche
Ayrton Dumas
Marco Mattei
Groupe B

# Table des matières

# 1. Introduction

Dans le cadre du cours de Big Data Analytics, nous allons réalisé un mini-projet afin de mettre en pratique les principes vus durant le cours. Notre choix s'est porté sur le dataset Audioscrobbler et sur un mini-projet centré sur la recommandation de musiques.

# 2. Description du dataset

Le dataset utilisé est appelé AudioScrobbler. Ce dataset contient les artistes musicaux écoutés par plus de 140'000 utilisateurs. Il s'agit d'une archive compressée contenant plusieurs fichiers texte. L'archive fait 135 MB compressée et 500 MB une fois décompressée.

## User artist data

Le fichier user_artist_data.txt contient les utilisateurs et les artistes qu'ils écoutent.Chaque ligne correspond contient l'id de l'utilisateur, l'id de l'artiste et le nombre de fois qu'il a écouté une chanson de cet artiste, donc pour chaque artiste écouté par un utilisateur, il y aura une nouvelle ligne. Il y a 140'000 utilisateurs qui ont écouté en tout 24'296'858 artistes. Ce fichier contient donc 3 colonnes: userid, artistid, playcount

## Artist data

Le fichier artist_data.txt contient les mappings artistes id -> nom de l'artiste. Ce fichier contient donc 2 colonnes: artistid, artist_name. Il y a 1'848'707 artistes dans ce fichier.

## Artist alias

Le fichier artist_alias.txt contient les noms d'artiste qui sont souvent mal orthographiés ou qui sont sujets à variante (Depeche Mode mal orthographié en Depeche Mood par exemple). Ce fichier contient donc 2 colonnes: badid, goodid. Il y a 193'027 alias.

# 3. Questions d'analyse

1. Peut-on obtenir des résultats intéressants en utilisant des techniques de Market Basket Analysis telles que les règles d'association ?
2. Peut-on déduire les genres des musiques à l'aide d'un clustering des utilisateurs et leurs écoutes ?
3. Est-il possible de recommander non pas des artistes mais d'autres utilisateurs qui partagent les même goût avec ALS ?

Etant donné que la question 3 n'était pas possible à répondre (cf. 5. Algorithmes appliqués), nous l'avons adaptée avec la suivante:

4. Est-il possible de recommander des utilisateurs qui ont des goûts similaires à un certain utilisateur en utilisant du clustering ?

# 4. Description des features / pre-processing afin d'extraires des features additionnelles

## Travail déjà effectué dans le livre

Dans le chapitre donné du livre [1], TODO

## Market Basket Analysis preprocessing

Afin d'appliquer des techniques de Market Basket Analysis, les données doivent être transformées. En effet, actuellement les données se trouvent sous cette forme:

```
+-------+-------+-----+
|   user| artist|count|
+-------+-------+-----+
|1000002|      1|   55|
|1000002|1000006|   33|
|1000002|1000007|    8|
|1000002|1000009|  144|
|1000002|1000010|  314|
+-------+-------+-----+
```

On peut voir qu'un utilisateur est présent sur plusieurs lignes, une ligne pour chaque artiste qu'il a écouté. Afin de pouvoir exécuter les algorithmes de Market Basket Analysis (voir chapitre 5. Algorithmes appliqués pour plus d'informations), les données doivent être transformé en format transactionnel, c'est à dire ainsi:

```
+-------+--------------------+
|   user|               items|
+-------+--------------------+
|1000002|[1, 18, 28, 30, 5...|
|1000019|[1000010, 1000028...|
+-------+--------------------+
```

On aperçoit à présent qu'un seul utilisateur est présent par ligne, et que tous les artistes qu'il a écouté ont été regroupé sur une seule ligne, dans une colonne nommé "items" de type *Array*. Pour faire l'analogie avec un exemple plus parlant, on peut dire, qu'avant, notre dataset contenait une ligne pour chaque item que notre client avait acheté, donc une ligne pour clientA + pain, une deuxième ligne pour clientA + beurre. Maintenant ces lignes ont été regroupé par client donc clientA + pain + beurre.
Pour obtenir ce résultat, il a fallu:

- Grouper le dataframe par user
- Pivoter le champs artiste en tant que colonne
- Grouper toutes ces colonnes d'artistes un une seule colonne "items" de type *Array*

# 5. Algorithmes appliqués

## Market Basket Analysis

L'algorithme utilisé pour miner les frequent items sets est FPGrowth. Il existe deux implémentations de cet algorithme dans la librairie Scala: org.apache.spark.ml.fpm.FPGrowth et org.apache.spark.mllib.fpm.FPGrowth. La première alternative a été utilisé car il était plus aisé de performer les différentes étapes de preprocessing décrites au chapitre 4. L'avantage de l'algorithme de FPGrowth par rapport à l'algorithme de apriori vu en cours de Data Management au semestre passé est qu'il ne génère pas les candidats explicitement, ce qui fait qu'il est largement plus adapté aux grands sets de données, ce qui est notre cas dans ce projet. "*It is found out that the FP-Growth algorithm outperforms the Apriori and ECLAT algorithms for all databases in terms of runtime and usage of memory*." Sinthuja et al. [3].
FPGrowth nous permet donc de connaître les items sets fréquents, ensuite, à partir des ces items sets fréquents, on peut en déduire les règles d'association.

## Clustering des utilisateurs

Notre idée pour déterminer les genres de musique et de clusteriser les utilisateurs; un utilisateur écoute principalement qu'un seul type de musique. Nous avons donc sélectionné le top 3 des écoute de chaque utilisateur et effectué une clusterisation.

## Recommandation d'utilisateurs avec ALS

En premier lieu, nous voulions donc utiliser ALS pour recommander des utilisateurs à un certain utilisateur, en fonction des artistes qu'ils écoutent. C'est-à-dire qu'on veut suggérer à un utilisateur A des utilisateurs qui écoutent des artistes similaires à l'utilisateur A.
Après un certain nombre de recherches, nous nous sommes rendus compte qu'ALS n'était pas adapté à ce genre de problème. En effet, ALS fonctionne sur le principe "user-item" et le problème indiqué ici est plus un problème "user-user" et malgré nos recherches nous n'avons pas trouvé un moyen de répondre à la question avec ALS.
Nous avons donc décidé de tenter de répondre à la question avec du clustering, comme expliqué au chapitre suivant.

## Recommandation d'utilisateurs avec du clustering

Notre idée pour recommander des utilisateurs avec du clustering a été de clusteriser tout le set de données. C'est-à-dire que c'est un cluster 3D avec les axes suivants :

- X : Utilisateur
- Y : Artiste
- Z : Nb d'écoutes

Ceci permet de trouver des clusters d'utilisateurs qui ont écoutent beaucoup de fois les mêmes artistes.
Cependant, cette méthode contient un défaut. En effet, nous nous sommes rendus compte qu'il aurait fallu normaliser les axes car actuellement les utilisateurs et les artistes sont des valeurs catégoriques ce qui signifie que leur valeur sur l'axe n'a pas réellement de "sens" pour le cluster.

Effectivement, un utilisateur A qui écoute les mêmes artistes qu'un utilisateur B peut se trouver à une très grande distance de ce dernier à cause de son placement dans le graphe. Il s'agit donc d'un problème relativement complexe.

# 6. Optimisations réalisées

## Market Basket Analysis

Les paramètres à optimiser sont les suivants:

- Le support minimal: le support correspond au pourcentage de transactions qui contiennent la totalité des items d'un item set

$$Support = \frac{frq(X,Y)}{N}$$

- La confidence minimale: la confidence minimale correspond au pourcentage de transactions qui contiennent la totalité des items d'un item set sur le pourcentage qui contiennent la partie gauche de l'item set (l'antécédent)

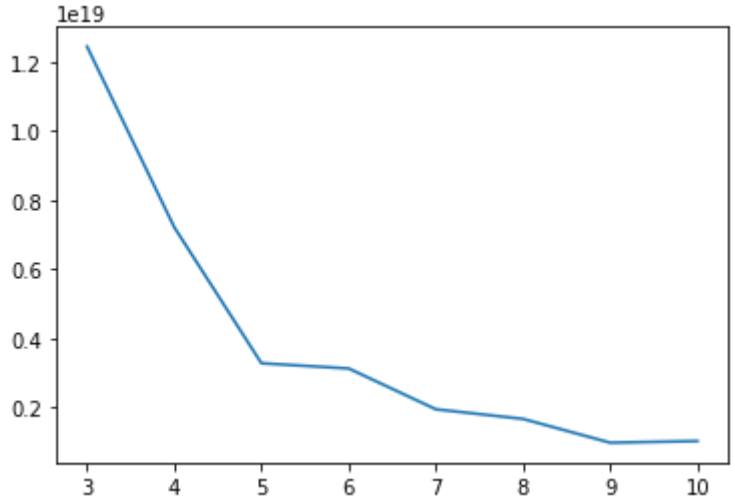
$$Confidence = \frac{frq(X,Y)}{frq(X)}$$

Nous avons choisi les valeurs suivantes pour ces paramètres:

- Support minimal: 0.1 => le dataset étant très grand et contenant énormément d'artistes différents (environ 2 millions), pour seulement 140'000 utilisateurs donc 140'000 transactions, il est nécessaire d'avoir une valeur très basse pour ce paramètre. Si on met une valeur trop haute, on obtient tout simplement 0 règles d'association. Le fait de mettre une valeur trop grande rendrait les règles d'association biaisé envers les artistes très populaire au détriment des artistes un peu moins connu et écouté.
  Rien qu'en passant le support de 0.1 à 0.2, sans tenir compte de la confidence (on met le minimum à 0), on passe de 1470 règles à 0 règles
- Confidence minimale: 0.4 => avec le support minimum mis à 0.1, on passe de 1470 règles à 1277 règles. Pour la confidence, le dénominateur étant les transactions contenant "X", on peut se permettre de mettre une valeur plus élevée que pour le support.
  En passant la confidence à 0.5, on passe à 821 règles comparé au 1277 obtenu avec 0.4, ce qui correspond à une perte trop importante.

## Méthode du coude

Dans un système de clustering, le nombre de groupe à créer est inconnu. Dans notre cas, nous ne savons pas combien de genres de musique existent. Nous avons utilisé la méthode du coude déduire le nombre de genre idéales. Dans la figure suivante, il est possible de voir ce "coude" lorsque **K** vaut 5, 5 genre de musiques.




# 7. Tests et évaluations

## Market Basket Analysis

Malheureusement, il n'a pas été possible de travailler avec l'intégralité des données. En effet, l'utilisation des 25 millions d'écoutes utilisateurs faisait crasher l'application avec une erreur org.apache.thrift.transport.TTransportException qui correspond à un problème dû à un problème de mémoire. Une solution proposée sur internet indiquait d'augmenter la mémoire allouée à l'interpréteur spark dans la configuration de zeppelin [1], conseil également dispensé par les collègues Yann-Ivain Beffa et Jonathan Donzallaz sur le Teams du cours. Nous avons augmenté cette mémoire de 1g à 12g, mais le problème subsistait. Nous avons donc pris la décision arbitraire de ne garder que 5 millions des 24 millions d'écoutes utilisateurs. Cela fait que nous avons uniquement 21'117 transactions (donc utilisateurs), sur les 140'000 de base.
Comme expliqué dans le chapitre 6 Optimisations réalisées, le système final possède 1277 règles d'association.
L'évaluation des règles d'association est une tâche ardue. En effet, Kaliappan Vanitha et al. indiquent que "*However, mining association rules often results in a very large number of found rules, leaving the analyst with the task to go through all the rules and discover interesting ones.*" [2].
Une approche manuelle est donc à faire, et forcément l'évaluation est sujette au biais de l'évaluateur (est-ce qu'il trouve que cette règles est pertinente ?). Nous n'avons bien sûr pas le temps d'analyser ces 1277 règles, et nos connaissances musicales sont plutôt limitées, surtout quand on sait que le set contient 2 millions d'artistes, il est donc inimaginable que nous connaissions ces 2 millions d'artistes.

[1] https://stackoverflow.com/questions/36835122/org-apache-thrift-transport-ttransportexception-error-while-reading-large-json-f

```
[352,1177,979,0.8645339652448657]
Beck, Coldplay,  -> Radiohead
[352,1307,979,0.843316144387148]
Beck, The White Stripes,  -> Radiohead
[1001646,1177,979,0.8378378378378378]
The Smashing Pumpkins, Coldplay,  -> Radiohead
[352,3327,979,0.8362779740871613]
Beck, Weezer,  -> Radiohead
[1001646,1307,979,0.8283378746594006]
The Smashing Pumpkins, The White Stripes,  -> Radiohead
[1001646,1275996,979,0.8282633808240277]
The Smashing Pumpkins, R.E.M.,  -> Radiohead
[234,1000113,979,0.823134328358209]
Pixies, The Beatles,  -> Radiohead
[1001779,1000113,979,0.8190219484020023]
Modest Mouse, The Beatles,  -> Radiohead
[352,1000113,979,0.8182748039549949]
Beck, The Beatles,  -> Radiohead
```

Ci-dessus les 9 règles d'association ayant le plus de confidence. La première ligne contient les ids des artistes et la confidence, la 2ème ligne le nom des artistes. Sur ces 9 règles, le groupe "Radiohead" est toujours le conséquent de la règle. "Radiohead" est un groupe anglais de rock des années 80. En s'intéressant à la 8ème règle (Modest Mouse, The Beatles -> Radiohead), tous les groupes de la règle font de la musique rock et datent des années 60-90, ce qui fait du sens. C'est le cas de la plupart de ces 9 règles qui concernent des artistes rock en général.
Une manière d'évaluer serait de regarder toutes les règles contenant un artiste apprécié par un des étudiants, ainsi il serait plus apte à juger des recommandations en fonction de ses goûts.
L'artiste choisi est R.E.M:

```
[1275996,4267,979,0.7412831241283124]
R.E.M., Green Day,  -> Radiohead
[1275996,1274,976,0.737450462351387]
R.E.M., Red Hot Chili Peppers,  -> Nirvana
[1275996,1177,1205,0.7180659915060438]
R.E.M., Coldplay,  -> U2
[1275996,976,1274,0.7150176112712135]
R.E.M., Nirvana,  -> Red Hot Chili Peppers
```

Les règles font du sens, l'étudiant en question apprécie les artistes recommandés.

# Clustering

KMeans offre une méthode de calculs de coût permettant d'évaluer la position des centroids, nous l'avons appliqué à plusieurs k pour évaluer lequel était le plus performant.

```
K=1 Within Set Sum of Squared Errors = 1.2445173257535345E19
K=2 Within Set Sum of Squared Errors = 7.2132628707644774E18
K=3 Within Set Sum of Squared Errors = 3.2724949687905096E18
K=4 Within Set Sum of Squared Errors = 3.1214615696898002E18
```

Une évaluation des résultats .. TODO

# 8. Résultats obtenus

## Market Basket Analysis

Nous pouvons répondre à notre question: Peut-on obtenir des résultats intéressants en utilisant des techniques de Market Basket Analysis telles que les règles d'association ?
La réponse est oui. Le système mis en place actuellement possède certaines limites, mais il permet de valider l'idée. Les chapitres 7 Tests et évaluations et 9 Améliorations futures décrivent les résultats obtenus, leur pertinence, ainsi que leur limitations.

???????? Demo dispo à : https://www.gnoosic.com/

# 9. Améliorations futures

De manière générale, le dataset aurait besoin d'un énorme nettoyage. En effet, dans le fichier des artistes, bien des artistes sont mal orthographiés, certaines fois il y a l'artiste et le nom d'une de ses chansons ensemble. Le fichier des alias est insuffisant pour pouvoir nettoyer correctement ce fichier. Certains artistes sont présent plus de 50 fois dans le fichier des artistes. Certaines fois cela fait du sens, par exemple lors du featuring de plusieurs artistes.

## Market Basket Analysis

Le système possède 1277 règles d'association. Au départ, nous pensions que c'était beaucoup, mais en fait cela est peu. En effet, 1277 règles ne permettent pas de prendre en comptes les artistes les moins bien connus, il faudrait réduire encore le support minimum jusqu'à obtenir plus de règles, 10'000 règles serait une bonne base. En cherchant des artistes moins connus, il est impossible de trouver une règle parmis ces 1277 règles.

# 10. Conclusion

# 11. Références

[1] Sandy Ryza, Uri Laserson, Sean Owen, Josh Wills, Advanced Analytics with Spark, O'Reilly, May 2017
[2] Kaliappan Vanitha, R. Vijaya Santhi, Evaluating the performance of association rule mining algorithms, 2011
[3] Sinthuja, M. & Puviarasan, N. & Aruna, P.. (2017). Evaluating the Performance of Association Rule Mining Algorithms. World Applied Sciences Journal. 35. 43-53., 10.5829/idosi.wasj.2017.43.53.