VINCENT GAGNON – GAGV25059800

Baccalauréat en Informatique

Rapport Projet Final

Travail d’apprentissage machine

présenté à

Julien Maitre

dans le cadre du cours

8INF436 Forage de données

Module d’informatique et de mathématique

Université du Québec à Chicoutimi

Le 4 Mai 2021

## Description de l’ensemble de données

L’ensemble de données choisi contient 42 305 chansons provenant de Spotify. Chaque chanson possède 22 attributs dont le titre et le genre. Nous chercherons à classer les chansons selon leur genre et donc entraînerons notre modèle pour qu’il puisse prédire le genre d’une chanson selon les caractéristiques de celle-ci. Voici les attributs de chaque instance :

### Danceability

Plus la valeur est élevée, plus il est facile de danser sur la chanson.

### Energy

Plus la valeur est élevée, plus la chanson est énergétique.

### Key

La clé de la chanson (Do, Ré, Mi, …). Prend les valeurs 0 à 11 qui correspondent aux douze notes que peut prendre la clé.

### Loudness

L’intensité sonore relative de la chanson en décibels.

### Mode

Valeur binaire représentant s’il s’agis d’une chanson commençant par une progression d’accords majeurs (1) ou non (0).

### Speechiness

Plus la valeur est élevée, plus la chanson contient des paroles.

### Acousticness

Plus la valeur est élevée, plus la chanson est acoustique

### Instrumentalness

Plus la valeur est proche de 1, plus la chanson est instrumentale.

### Liveness

Plus la valeur est élevée, plus il y a de chances que la chanson ait été enregistrée avec un public ou dans un concert.

### Valence

Plus la valeur est élevée, plus la chanson sonne positive.

### Tempo

Le tempo de la chanson, en battements par minute.

### Type

Type de contenu sur Spotify, ne prend qu’une seule valeur et n’est donc pas un attribut utile.

### Id

Identifiant unique de la chanson sur Spotify.

### Uri

Lien Spotify vers la chanson.

### Track\_href

Lien vers la chanson sur l’api de Spotify.

### Analysis\_url

Lien de l’analyse de la chanson sur l’api de Spotify.

### Duration\_ms

Durée de la chanson en millisecondes.

### Time\_signature

Mesure de la chanson.

### Genre

Genre de la chanson, prend 15 valeurs distinctes : Underground Rap (5875), Dark Trap (4578), Hiphop (3028), Trance (2999), Trap (2987), Techhouse (2975), DnB (2966), Psytrance (2961), Techno (2956), Hardstyle (2936), RnB (2099), Trap Metal (1956), Rap (1848), Emo (1680) et Pop (461).

### Song\_name

Nom de la chanson.

### Unnamed : 0

Numéro de la chanson dont le champ song\_name est null.

### Title

Titre des chansons dont le champ song\_name est null.

En voyant que l’ensemble de données ne contient que 42 305 chansons et 15 genres, il est facile de constater que celui-ci est loin d’être l’ensemble complet de toutes les chansons sur Spotify.

## Méthodologie

La méthodologie utilisée pour classifier les instances selon le genre sera la suivante. En premier lieu nous allons effectuer un clustering sur les instances sans regarder les labels. De cette manière nous essayerons de regrouper les instances semblables dans les mêmes clusters. Après avoir mis chaque instance dans un cluster, nous allons séparer les données en ensemble de test et d’entrainement de façon qu’il y ait 30% de données de test et 70% de données d’entrainement. Ensuite, on applique le modèle supervisé DecisionTreeClassifier de sklearn afin d’entrainer le modèle, cependant les labels utilisés sont le cluster dans lequel chaque instance a été grouper. Après l’entrainement on utilise les données de test afin de classer les instances restantes dans des clusters.

Ensuite pour noter les performances de notre modèle, on regardera la classe dominante dans chaque cluster et cette classe sera choisi comme classe vrai pour ce cluster. On utilise ensuite le ratio de classe vrai et de classe fausse dans chaque cluster afin de mesurer différentes métriques de performance du modèle.

Les seuls paramètres que nous allons modifier afin de trouver le meilleur modèle sont l’algorithme de clustering et ses hyperparamètres. Par exemple, si l’on utilisait l’algorithme kmeans, on pourrait aussi modifier les paramètres dictant le nombre de clusters, le nombre d’initialisations, le nombre maximal d’itérations et la tolérance relative entre deux clusters.

On entraine et test chaque modèle trois fois avec trois seed de génération de nombres aléatoires différentes et on prend la moyenne des résultats comme valeur absolue de la configuration.

Les métriques de performance que l’on regarde sont les suivantes : la précision moyenne de tous les clusters avec un poids selon le nombre d’instances dans le cluster (Plus d’instances sont dans le cluster, plus la précision de ce cluster sera importante), le score Kappa, la précision moyenne avec un poids selon le nombre d’instances correctement classifiées dans le cluster, le recall avec poids selon le nombre d’instances correctement classifiées dans le cluster, le score F1 avec poids selon le nombre d’instances correctement classifiées dans le cluster, le coefficient de corrélation de Matthews et on pourra aussi regarder la matrice de confusion.

## Hypothèses

Dans chacun des autres travaux dans lequel j’ai fait des hypothèses j’ai toujours surestimé les performances de mes modèles, et donc cette fois je vais essayer de diminuer mes attentes envers ce modèle. Je vais prédire que mon modèle aura pour finir une précision moyenne de 50%, cela semble faible mais il ne faut pas oublier qu’on doit prédire la bonne classe sur 15 classes, ce qui est beaucoup plus difficile que de prédire vrai ou faux.

Je crois également que le meilleur algorithme de clustering sera Kmeans car c’est celui avec lequel j’ai vu les meilleurs résultats en général dans mon expérience. Je pourrais même m’avancer à dire que la précision moyenne la plus élevé sans avoir trop de clusters sera atteinte avec un nombre de clusters un peu plus élevé que le nombre de classes, disons 19.

## Vérification et pré-traitement des données

L’ensemble de données contient des valeurs manquantes pour l’attribut song\_name, Unnamed : 0 et Title. Cependant, les instances donc la valeur est manquante pour song\_name, l’attribut Title est présent. On peut donc dire que l’ensemble de données ne contient pas de valeurs manquantes.

En regardant les diagrammes à boite des valeurs des différentes caractéristiques on s’aperçoit que certaines d’entre elles contiennent des valeurs extrêmes. En revanche, aucune d’entre elles ne sont des valeurs aberrantes et donc nous n’avons pas d’instances uniques à supprimer concernant les valeurs aberrantes.

Plusieurs attributs sont inutiles pour l’usage dont nous voulons en faire. Entre autres, tout ce qui a rapport aux titres des chansons et les identifiants et Uri Spotify, nous allons donc supprimer ces colonnes de l’ensemble de données. Nous supprimons aussi la colonne type car elle n’a qu’une seule valeur qui n’est pas significative.

En regardant le nombre d’instance par genre de l’ensemble de données, on constate que celui-ci est débalancé, On retrouve 5875 instances de Underground Rap contre seulement 461 instances de pop dans l’ensemble de données. Je vais donc balancer l’ensemble de données artificiellement en faisant de l’oversampling. C’est-à-dire que l’on regarde le nombre d’instances dans la classe dominante, soit 5875 dans notre cas, et on fait des copies des instances des autres classes dans l’ensemble de données jusqu’à ce que le nombre d’instances dans chaque classe soit égal à 5875. Notre ensemble de données passe donc de 42 305 à 88 125 instances.

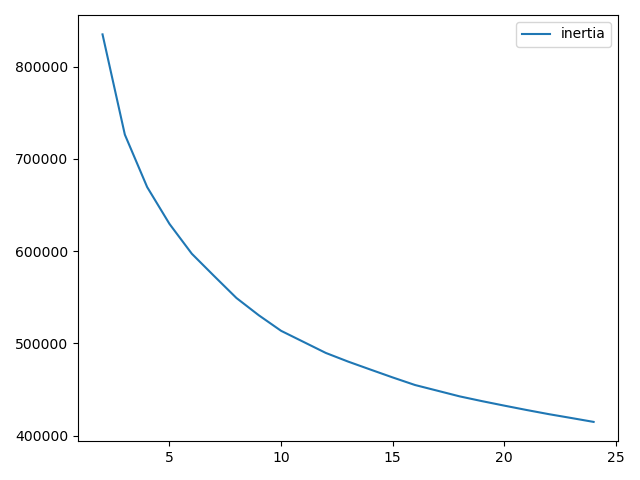
Après avoir supprimer ces colonnes, on retrouve encore des valeurs catégoriques dans le dataset, entre autres les attributs key, mode et time\_signature. Je vais donc encoder ces attributs en utilisant le One Hot Encoding afin de les encodeurs. Cela consiste à créer une nouvelle colonne pour chaque catégorie d’un attribut et pour chaque instance on met un 1 dans la bonne colonne correspondant à la catégorie de l’attribut. En effectuant cela pour les colonnes key et time\_signature on obtient seulement des 0 et des 1 pour les attributs catégoriques du dataset.

On encode les caractéristiques catégoriques de cette façon car on ne veut pas débalancer les algorithmes que nous utiliserons plus tard dans ce travail, par exemple si on avait encodé les attributs en remplaçant les catégories par des chiffres, ceux-ci auraient pu débalancer l’algorithme car les catégories avec des nombres plus élevé auraient eu un plus grand poids que ceux ayant un nombre moins élevé. En mettant des 1 et des 0 on évite ce problème.

Après avoir encoder les attributs catégoriques, on standardise les valeurs des attributs numériques afin qu’une colonne n’ait pas des valeurs qui dominent les autres colonnes.

## Regroupement des instances

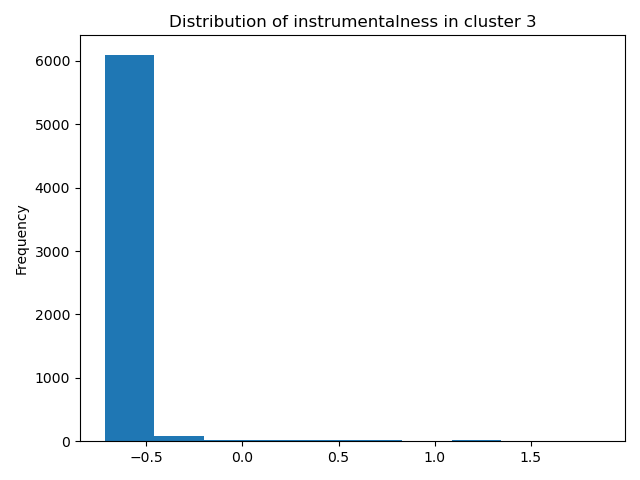
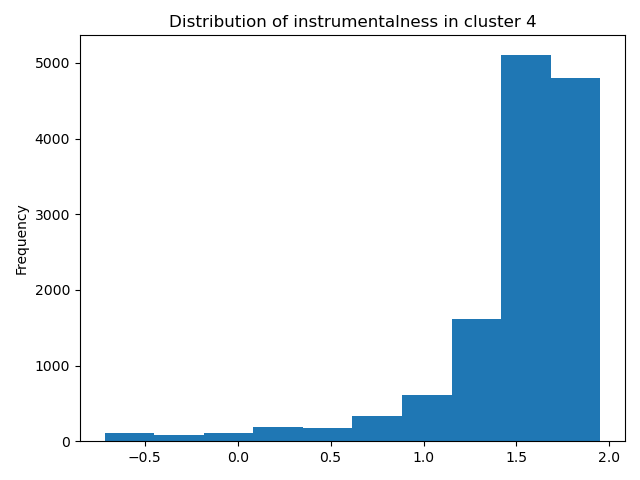
Pour commencer le regroupement des instances, j’utilise l’algorithme de clustering Kmeans avec un nombre d’initialisations de 10, le nombre maximal d’itérations de 300 et une valeur tol de 0.0001.

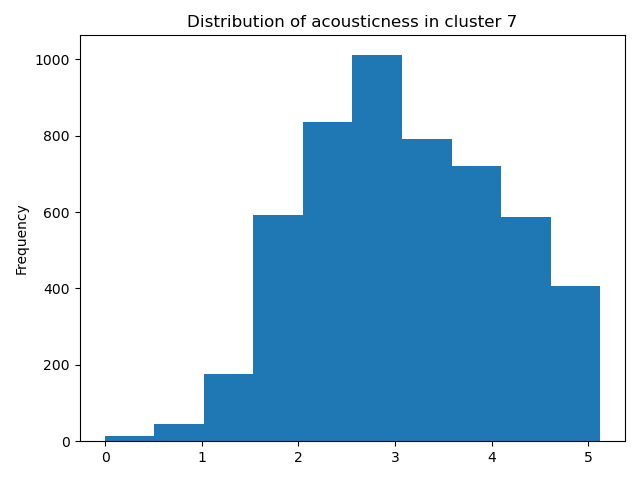
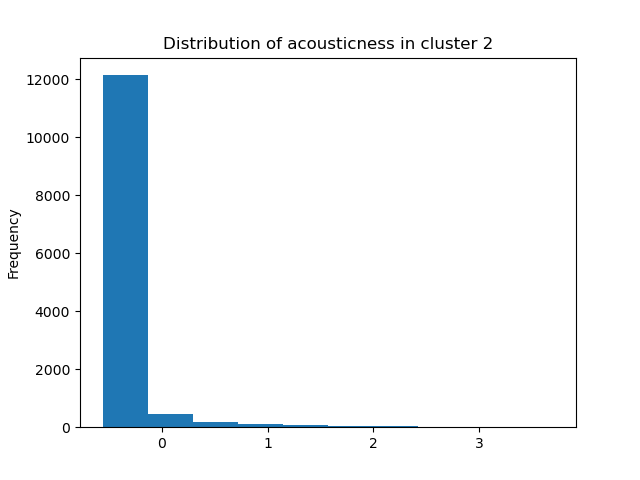
Je commence par tracer le graphique de l’inertie par rapport au nombre de clusters afin de trouver un juste milieu entre ces deux valeurs. Le voici :

On voit que la pente de la courbe est à pic au début et donc qu’augmenter le nombre de cluster d’un à deux a un grand impact sur l’inertie intra-clusters. Cependant, la pente de la courbe devient de moins en moins à pic à mesure que l’on ajoute des clusters, on doit donc trouver le point où le nombre de cluster n’est pas trop élevé, car cela complique la classification, mais on doit tout de même avoir assez de cluster pour que la précision de l’algorithme soit assez bonne.

Je décide donc de choisir un nombre de cluster égal à 15 comme première valeur pour tester la classification de l’étape suivante. J’ai choisi cette valeur car elle est égale au nombre de classes et que l’inertie semble devenir plus stagnante après 15 clusters.

J’ai fait tracer un graphique de la distribution d’attributs selon deux groupes différents. Voici ceux de l’instrumentalness :



On voit que les instances sont bien séparées selon cet attribut. Cependant ce n’est pas toujours le cas pour tous les attributs, entre autres pour ce qui est de l’acousticness.

On voit que le cluster 7 contient des instances dont l’attribut acousticness est très varié contrairement au cluster 2 dont cet attribut est faible sur toutes les instances. Au total on retrouve 2 clusters dont cet attribut est très varié comme dans le cluster 7 alors que dans le reste des ces clusters, cet attribut est faible. Ce comportement semble normal car les genres de musique que l’on retrouve dans l’ensemble de données sont soit de la musique électronique ou du rap, et dans ce type de musique les chansons sont rarement acoustiques. L’algorithme de clustering regroupe donc les musique les plus acoustiques, même si elles ne le sont pas toutes, dans les mêmes clusters.

## Classification et performances

Après avoir regroupé les instances dans des clusters, on entraîne un modèle supervisé avec comme labels les clusters dans lequel les instances ont été classifiées.

Le code pour l’entrainement du modèle de classification et la majorité des métriques de performance de celui-ci a été fourni par Julien Maître, celui-ci implémente l’entrainement d’un modèle supervisé qui utilise l’algorithme de classification DecisionTreeClassifier de sklearn, qui est seulement un arbre de décision ordinaire.

Après avoir ajouter les clusters comme des labels aux instances, j’entraine celle-ci à l’aide du code fourni et j’interprète les résultats.

Voici les résultats obtenus pour la première et la meilleure classification pour l’algorithme de clustering Kmeans :

|  |  |  |
| --- | --- | --- |
|  | Premier essai | Meilleur essai |
| N\_clusters | 15 | 20 |
| N\_init | 10 | 10 |
| Max\_iter | 300 | 30 |
| Tol | 0,0001 | 0,0001 |
| Précision moyenne | 34,07 | 38,47 |
| Score kappa | 0,2935 | 0,3407 |
| Précision | 0,2803 | 0,3687 |
| Recall | 0,3407 | 0,3847 |
| Score F1 | 0,2949 | 0,3601 |
| Coefficient de corrélation de Matthews | 0,2998 | 0,3454 |

Premièrement, j’ai augmenté le nombre clusters à 20, car on gagne environ 4,5% sur toutes les métriques en ajoutant seulement 5 clusters. J’ai essayé avec 23 clusters et le gain était d’à peine 1%, je trouve donc que 20 clusters semble être un nombre raisonnable. Augmenter le nombre de clusters diminue l’inertie et a tendance à augmenter la précision, d’où l’augmentation de celle-ci.

J’ai aussi diminué le nombre d’itérations de 300 à 30. Bien que ce paramètre ait beaucoup moins d’impact que le nombre clusters, on retrouve tout de même une légère augmentation en passant de 300 à 30. Je suppose que diminuer le nombre d’itérations diminue les chances que l’algorithme de clustering tombe dans un minimum ou maximum local d’où les meilleures performances mais ce n’est qu’une hypothèse.

Après avoir trouvé les paramètres que je trouve les meilleurs pour le clustering avec Kmeans, je suis passé à l’algorithme de clustering hiérarchique. Voici le meilleur résultat :

|  |  |
| --- | --- |
| N\_clusters | 20 |
| Précision moyenne | 36,13 |
| Score kappa | 0,3156 |
| Précision | 0,3411 |
| Recall | 0,3613 |
| Score F1 | 0,3320 |
| Coefficient de corrélation de Matthews | 0,3205 |

Ces résultats sont similaires à l’algorithme Kmeans mais un peu moins bon d’environ 2%.

Pour finir j’ai essayé l’algorithme de clustering DBSCAN, voici les résultats :

|  |  |
| --- | --- |
| eps | 1,43 |
| Min\_samples | 5 |
| Précision moyenne | 19,46 |
| Score kappa | 0,1367 |
| Précision | 0,6473 |
| Recall | 0,1946 |
| Score F1 | 0,2001 |
| Coefficient de corrélation de Matthews | 0,1701 |

Cet algorithme a été de loin le plus difficile à balancer et même finalement je n’ai pas réussi. On voit dans les résultats que la précision moyenne est de 19,46% alors que la précision est de 64,73%. Cela s’explique par le fait que la plupart des instances sont grouper dans le même cluster, on peut le voir en allant voir le nombre d’instances par clusters. Étant donné que la précision moyenne prend le poids en fonction du nombre d’instances total et que la précision prend le poids en fonction du nombre d’instances bien classifiées, ces deux valeurs sont différentes et la précision ne montre pas la réalité de ce modèle.

J’ai essayé beaucoup de paramètres différents par cet algorithme mais en vain, chaque essai menant à soit trop d’instances se retrouvant dans aucun groupe ou soit trop d’instances dans le même groupe, finalement DBSCAN n’a pas été utile dans cet ensemble de données.

## Conclusion de l’étude

Pour conclure l’étude, les résultats obtenus n’ont pas été très bon, avec la meilleure précision de 36,87% et un recall de 38,47% à l’aide de l’algorithme de clustering Kmeans.

Je valide donc mon hypothèse que cet algorithme de clustering soit le meilleur pour cet ensemble de données, en revanche j’ai encore surestimé la précision du modèle dans mon hypothèse, j’avais estimé une précision de 50% alors qu’au final on obtient 37%.

Ces valeurs de précision et de rappel ne semblent pas être élevé mais, comme je le mentionnais dans mon hypothèse, ici il n’est pas seulement question de deux classes, mais de 15, il est donc beaucoup plus difficile de bien classifier les instances et donc la précision du modèle est probablement plus élevé que ce que l’on croit.

Une division des données différentes aurait pu donner d’autres résultats et peut-être améliorer le modèle mais je n’ai pas pris la peine de le tester.

## Conclusion Générale

Pour terminer le rapport, j’ai trouvé intéressant ce que l’on peut faire de manière semi-supervisée. J’avais lu la théorie mais le mettre en pratique m’as beaucoup aidé à comprendre comment cela fonctionnait réellement.

Je dirais que mon modèle n’a pas performé aussi bien que je l’aurais souhaité mais il a tout de même réussi à bien classifier une bonne partie de l’ensemble de données test.

Avec plus de temps j’aurais également pu trouver un ensemble de paramètres qui fonctionne pour l’algorithme de clustering DBSCAN. J’ai trouvé que certain choix de paramètres prenait beaucoup de temps à exécuter et cela m’a ralenti considérablement dans les essais de différents paramètres. J’aurais peut-être dû diminuer le nombre d’instances dans mon ensemble de données.