Rapport TDM6 : Introduction aux sciences des données et à l'intelligence artificielle

Fanani Selma et Amsaf Rim (Groupe 4)

October 18, 2024

1 Analyse des données du jeu Unpopular spotify Songs (USS)

L'objectif de cette partie est d'étudier les caractéristiques du jeu de données **Unpopular Spotify Songs (USS)** en utilisant les fonctionnalités de pandas. Nous avons récupérer le fichier unpopspotify.csv sous la forme d'un DataFrame et analyser les statistiques descriptives.

Après avoir chargé le fichier, nous avons examiné les types de données présents dans chaque colonne. Les résultats montrent que le jeu de données contient une diversité de types, allant de valeurs numériques à des chaînes de caractères et des valeurs booléennes. Les indicateurs musicaux, tels que danceability, energy, loudness, et tempo, sont représentés par des valeurs décimales (float64), tandis que des attributs comme key, mode, duration_ms, et popularity utilisent des valeurs entières (int64). On retrouve également une colonne booléenne (explicit).Enfin, les informations textuelles, telles que le titre de la chanson, le nom de l'artiste et l'identifiant de la piste, sont stockées sous forme de chaînes de caractères (object). Cette diversité de types de données reflète la variété des informations capturées, englobant à la fois des caractéristiques musicales et des données contextuelles sur chaque chanson.

En analysant les statistiques descriptives des colonnes numériques du jeu de données, on peut résumer les principales caractéristiques des variables. Ces statistiques incluent la moyenne, l'écart-type, les valeurs minimales et maximales, ainsi que les quartiles, ce qui donne un aperçu général de la distribution et de la variabilité des données. Par exemple, les indicateurs comme danceability, energy, et valence montrent des moyennes modérées, ce qui reflète des niveaux équilibrés de dansabilité, d'énergie et de positivité dans les chansons. Les valeurs de speechiness et instrumentalness indiquent qu'il y a une présence notable de morceaux avec du texte parlé et moins de passages purement instrumentaux. En ce qui concerne loudness, il est généralement assez bas pour la majorité des morceaux.Enfin, la popularité des chansons est globalement faible, avec une moyenne inférieure à 1, ce qui signifie que ce jeu de données regroupe principalement des titres peu populaires.

En vérifiant la présence de valeurs manquantes dans le jeu de données, nous avons constaté qu'aucune colonne ne contient de valeurs manquantes. Cela indique que le jeu de données est complet et prêt pour des analyses plus approfondies, sans nécessiter de traitement préalable pour combler les lacunes éventuelles.

L'analyse des histogrammes permet de visualiser la distribution des principales variables numériques. Pour la majorité des indicateurs musicaux (danceability, energy, valence), les valeurs sont réparties de manière relativement équilibrée, avec une concentration autour des valeurs moyennes. Par exemple, la plupart des chansons ont un niveau de dansabilité et d'énergie modéré.

Certaines variables comme acousticness et instrumentalness ont beaucoup de valeurs proches de zéro. Ça veut dire que, pour la plupart des morceaux, il y a peu d'éléments vraiment acoustiques ou instrumentaux. Par contre, il y a des pics plus marqués pour liveness et speechiness, ce qui montre que certaines chansons ont un côté plus 'live' ou contiennent plus de paroles parlées.

Enfin, les données sur le tempo et le loudness montrent que la plupart des chansons ont un rythme assez modéré et un volume plutôt bas.

Pour compléter l'analyse, des échantillons aléatoires de données ont été examinés afin de repérer d'éventuelles régularités ou anomalies. Cette approche permet de visualiser la diversité des chansons et de noter certaines tendances ou variations parmi les pistes du jeu de données. Par exemple, il a été observé que certains morceaux présentent des niveaux très bas de loudness, comme une piste mesurée à -25,37, ce qui suggère un volume inférieur par rapport à la moyenne. De plus, les valeurs de instrumentalness varient considérablement, allant de morceaux entièrement instrumentaux à d'autres sans aucun élément instrumental.

Enfin, la distribution de la popularité révèle que la majorité des chansons sont peu populaires, avec certaines affichant même une popularité de zéro.

2 Préparation des données et classification supervisée

Nous allons maintenant aborder la classification supervisée de notre jeu de données. Une classification est considérée comme supervisée lorsque notre modèle s'entraîne sur un ensemble de données étiquetées. Dans notre cas, cela est particulièrement pertinent, car notre jeu de données comprend une colonne *popularity*, qui servira d'étiquette pour nos classes.

Tout d'abord, nous allons déterminer le nombre de classes présentes dans notre jeu de données. Pour ce faire, nous afficherons le nombre de valeurs uniques de la colonne *popularity*. Nous constatons qu'il y a 10 classes distinctes.

Ensuite, nous allons examiner le pourcentage de chaque classe. Il est évident que notre jeu de données est très déséquilibré, car la classe 0 représente à elle seule environ 64% de l'ensemble des données. Cette répartition inégale pourrait poser des problèmes lors de l'entraînement de notre modèle.

Avant de commencer l'apprentissage de notre modèle, il est essentiel d'examiner les différentes colonnes de notre DataFrame et de retirer celles qui ne seront pas utiles pour la création de notre modèle. Pour ce faire, nous procédons comme suit :

Tout d'abord, nous éliminons les colonnes non informatives, telles que *track-name*, qui n'apportent pas de valeur ajoutée à notre analyse. Ensuite, nous transformons la colonne *explicit*, qui est de type booléen, en valeurs entières (0 ou 1).

Enfin, nous calculons la corrélation entre la colonne *popularity* et chacune des variables restantes. Cette étape nous permettra d'identifier les relations pertinentes et de conserver uniquement les caractéristiques les plus significatives pour l'apprentissage.

Après avoir calculé la corrélation entre chacune de nos variables et *popularity*, nous choisissons de conserver dans notre DataFrame uniquement les variables dont la valeur de corrélation, en valeur absolue, est supérieure à 0,05. Notre DataFrame contient donc actuellement les variables suivantes : *danceability*, *energy*, *loudness*, *acousticness*, *instrumentalness*, *valence*, *explicit*.

En examinant les colonnes que nous avons conservées, nous constatons qu'aucune d'entre elles ne présente de valeurs manquantes. Par conséquent, nous n'ajoutons rien à notre jeu de données pour le moment.

La prochaine étape consiste à standardiser nos données avant de réaliser notre premier modèle d'apprentissage. Pour ce faire, nous retirons la colonne *popularity* de notre jeu de données, car c'est la seule colonne à valeurs entières qui ne nécessite pas de standardisation. Ensuite, nous standardisons les autres colonnes dont les valeurs sont de type Int, puis nous réintroduisons la colonne *popularity* que nous avions précédemment supprimée.

Maintenant que nous avons un jeu de données suffisamment correct pour réaliser une classification supervisée, nous choisissons d'apprendre un arbre de décision d'une profondeur de 5 en utilisant la

validation croisée. En calculant l'erreur moyenne, nous obtenons une valeur de 0,38, ce qui signifie que notre arbre a prédit environ 40% de nos données de test de manière incorrecte. Ces résultats étaient à peu près attendus, compte tenu du grand déséquilibre de nos données. Il sera donc essentiel d'essayer de rééquilibrer notre jeu de données.

Avant de passer au rééquilibrage de nos données, notre première approche consistera à examiner les variables qui ont le plus contribué à la création de notre arbre. Pour ce faire, nous affichons l'arbre généré grâce à `sklearn.tree`, puis, en utilisant la méthode `feature_importance_`, nous présentons les six colonnes les plus discriminantes ainsi que celles qui n'apparaissent pas. Nous obtenons les résultats suivants :

Les six colonnes les plus discriminantes sont :

- `danceability` : 0.282790
- `loudness` : 0.256919
- `acousticness` : 0.156109
- `instrumentalness` : 0.145394
- `energy` : 0.088099
- `valence` : 0.070689

Les colonnes absentes sont : `['explicit']`.

Nous décidons donc de supprimer la colonne `['explicit']` ainsi que les deux dernières colonnes, `energy` et `valence`, qui ne sont pas suffisamment discriminantes. En effet, nous avons testé notre arbre de décision avec et sans ces données, et nous obtenons le même score d'accuracy.

En utilisant la validation croisée, nous obtenons un score d'accuracy de 0,62, une valeur qui reste acceptable mais insuffisante, car proche de l'aléatoire. Nous décidons donc de tester l'ajout de données artificielles à notre jeu de données en utilisant la méthode SMOTE pour le rééquilibrer et voir si cela améliorera notre score.

En faisant varier le nombre de données pour chaque classe et en observant celui qui produit le meilleur modèle, nous remarquons qu'en fixant le nombre de données à 500 (données originales + données artificielles) pour les classes de 2 à 9, nous obtenons une accuracy de 0,59. Ce résultat n'est pas meilleur que celui obtenu précédemment. Par conséquent, nous décidons de conserver le premier modèle pour le moment.

Avant de tester l'efficacité d'un classifieur *k-nearest neighbors* sur ce jeu de données, nous optons pour une dernière approche qui consiste à effectuer une recherche par grille (*grid search*) afin d'essayer différents hyperparamètres susceptibles d'améliorer les performances de notre arbre de décision. Comme dans le TDM précédent, nous choisissons la grille d'hyperparamètres suivante :

- `criterion` : {'entropy', 'gini'}
- `max_depth` : `range(1, 21)`
- `max_leaf_nodes` : `range(10, 221, 20)`

Cette approche nous permet d'explorer plusieurs combinaisons de paramètres et de déterminer celle qui maximise la performance du modèle. Cependant, en affichant les 5 meilleurs modèles, nous obtenons un score d'accuracy de 0,55, ce qui est inférieur à celui obtenu lors de notre premier modèle.

Passons maintenant au classifieur *k-nearest neighbors* (k-NN). Nous allons évaluer les performances de ce classifieur par validation croisée, en nous concentrant uniquement sur les colonnes les plus discriminantes (celles que nous avons gardées pour notre modèle final) et en choisissant la valeur la plus optimale de k.

Nous avons choisi d'effectuer une recherche par grille (*grid search*) en faisant varier k de 1 à 20. Étant donné que certaines classes de notre jeu de données contiennent peu d'exemples, nous avons décidé de ne pas prendre de valeurs de k trop grandes. La meilleure valeur de k retournée est $k = 20$, avec une précision (*accuracy*) de 0.62.

Nous avons également testé notre modèle sur les données augmentées avec SMOTE, où nous obtenons un score de 0.69. Cependant, dans ce cas, la meilleure valeur de k est $k = 1$, ce qui n'est pas idéal, car cela peut souvent mener à du surapprentissage (*overfitting*).

$$\text{Meilleur k sans SMOTE} = 20, \quad \text{Accuracy} = 0.62$$
$$\text{Meilleur k avec SMOTE} = 1, \quad \text{Accuracy} = 0.69$$

3 Classification non-supervisée

L'objectif de cette section est de regrouper les chansons en fonction de leurs caractéristiques communes grâce à une méthode de classification non-supervisée. Par la suite, nous évaluerons ces regroupements en les comparant à leur popularité. La qualité des clusters sera mesurée à l'aide du Rand Index en prenant la popularité comme référence, ainsi que via des mesures classiques d'évaluation de clustering.

Afin d'appliquer la classification non-supervisée, il a été nécessaire de préparer les données en supprimant la colonne popularity, que nous utiliserons ultérieurement pour l'évaluation. De plus, toutes les colonnes jugées non pertinentes ou déjà supprimées lors des étapes précédentes ont été retirées. Après cette étape de filtrage, les colonnes descriptives restantes pour l'analyse incluent **danceability, loudness, acousticness, instrumentalness.**

Nous avons identifié 10 valeurs uniques dans la colonne **popularity**, ce qui signifie que nous allons définir 10 clusters pour refléter ces différentes classes de popularité.

Pour évaluer la qualité des regroupements réalisés par la méthode de classification non-supervisée, nous avons répété les étapes d'expérimentation dix fois, en garantissant des divisions différentes à chaque itération grâce à l'utilisation de la variable random_state.

À chaque répétition, nous avons d'abord découpé le jeu de données en un jeu d'apprentissage représentant 70 % des données et un jeu de test de 30 %. Ensuite, nous avons appliqué l'algorithme K-Means pour déterminer 10 clusters, comme établi précédemment. Pour chaque élément du jeu de test, nous avons déterminé le cluster auquel il appartenait en utilisant le modèle entraîné sur le jeu d'apprentissage.

Pour évaluer la qualité des clusters, nous avons calculé le score de Davies-Bouldin, qui mesure la compacité et la séparation des clusters, ainsi que la silhouette moyenne, qui évalue la cohésion et la séparation des clusters. Enfin, nous avons calculé le Rand Index pour comparer les clusters prédits avec les vrais clusters, en utilisant la colonne popularity convertie en valeurs binaires. Après avoir effectué ces calculs pour les dix répétitions, nous avons obtenu des moyennes pour chaque métrique, avec un score de Davies-Bouldin de **1.13**, une silhouette moyenne de **0.26**, et un Rand Index de **0.0008**.

Les résultats de notre analyse de clustering montrent des performances variées. Le score de Davies-Bouldin, qui s'élève à 1.13, indique que les clusters ne sont pas bien séparés, suggérant un chevauchement significatif entre les groupes de chansons. De plus, la silhouette moyenne est de 0.26, ce qui est relativement bas et suggère que les chansons au sein des clusters manquent de cohésion, indiquant que certains points pourraient être mal classés ou que les groupes ne sont pas suffisamment denses. Enfin, le Rand Index, qui atteint seulement 0.0008, souligne que les clusters obtenus ne correspondent pas aux classes de popularité, ce qui témoigne d'une inefficacité de notre méthode de clustering pour regrouper les chansons selon leur popularité.

Afin de mieux comprendre les raisons de ces résultats, nous avons également visualisé la distribution des données en utilisant un graphique 3D des caractéristiques descriptives. Cette visualisation a été réalisée en appliquant une réduction de dimensionnalité sur les données, permettant d'examiner comment les chansons se répartissent dans un espace réduit. Le graphique montre clairement que

les chansons, représentées par des points colorés selon leur popularité, sont dispersées de manière dense et mélangée, sans regroupement distinct. Cette absence de séparation claire indique que les caractéristiques actuelles ne permettent pas de distinguer efficacement les chansons selon leur popularité, ce qui explique pourquoi l'algorithme K-Means a du mal à former des clusters cohérents.

References

[1] Imbalanced-learn Documentation. *Imblearn User Guide*. https://imbalanced-learn.org/stable/

[2] Vous retrouverez le code directement en suivant ce lien. https://github.com/SelmaFanani/TDM6