Music Mood prediction based on Spotify's Audio Features

Le projet consiste à vouloir prédire l'humeur musicale d'une chanson. Cette fonctionnalité pourrait permettre d'affiner la pertinence en matière de recommendation de titres à un utilisateur, en se basant sur son humeur.

Les données sont extraites depuis la plateforme de Spotify, via son API. Créer d'abord un compte sur Spotify for Developer, et créer une application pour ce projet (Web API). Retrouver les clés API. "Spotipy" est une librairie Python qui permet de s'authentifier avec ces clés et d'accéder à toutes les données de musique (artistes, albums, titres, mood, ...).

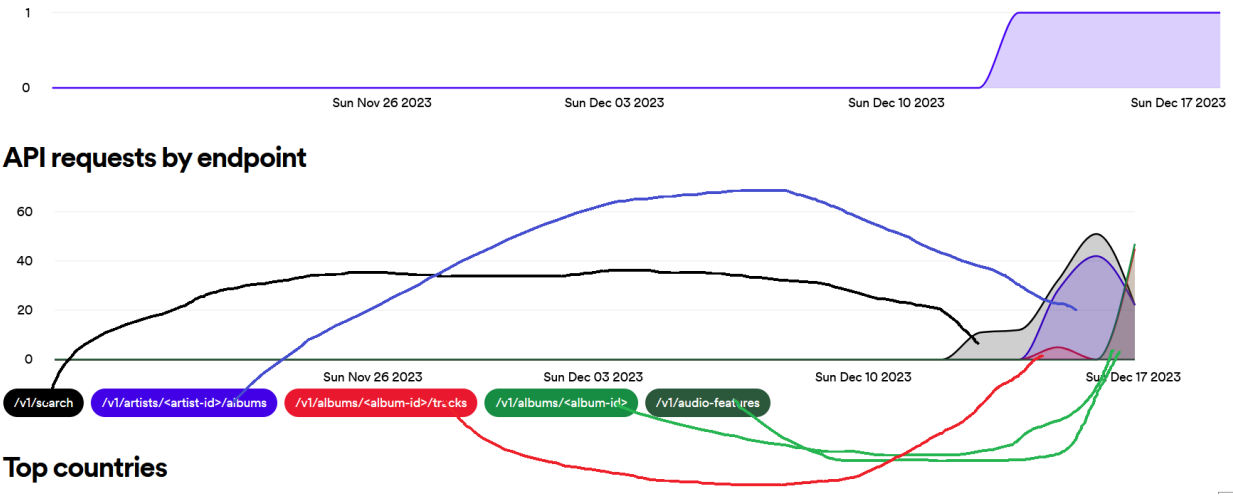
Le code est entièrement rédigé sur Spyder, le programme complet contenant 4 fichiers :

- main.py, qui permet de :
Lancer le code, s'authentifier pour utiliser l'API de Spotify, choisir un artiste, créer un dossier pour l'artiste, appeler les fonctions dans get_data.py pour extraire les titres et leurs informations pour ensuite les stocker sur un fichier CSV généré dans le dossier créé (un dossier au nom de l'artiste). Après avoir copié/collé manuellement les fichiers CSV d'une sélection d'artistes dans un dossier “artists” créé sur le même path, le programme appelle toutes les fonctions par étapes (numérotées de 1) à 5) en prenant soin de les commenter (#) / décommenter au fur et à mesure pour visualiser clairement le processus en tenant compte des fichiers/graphes générés et de ce qui s'affiche sur la console.

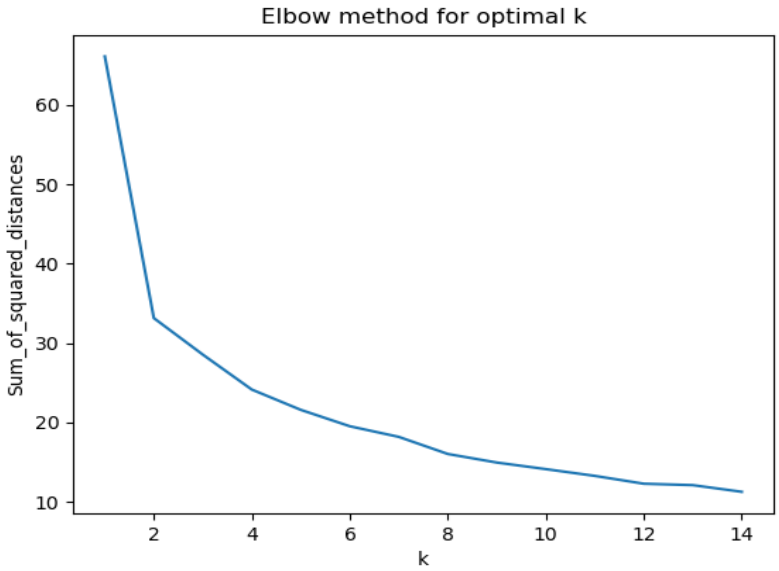
- credentials.yml :
Contient les clés API confidentielles (identifiant, mot de passe et nom d'utilisateur) pour pouvoir utiliser l'API de Spotify.

- get_data.py, qui permet d'extraire les données :
Récolter toutes les données de musique que l'on veut sur un artiste. Procédé par successions d'appels de fonctions et de modules Spotipy :
Nom de l'artiste (main) -> ID de l'artiste (step 1) -> liste d'albums (step 2) --> liste des IDs des titres de ces albums (step 3) --> liste de dictionaries, chaque dictionary contenant les informations et moods de chaque titre. Renvoie un dataframe au main pour créer un CSV “artists_data_labels_NO.csv”.

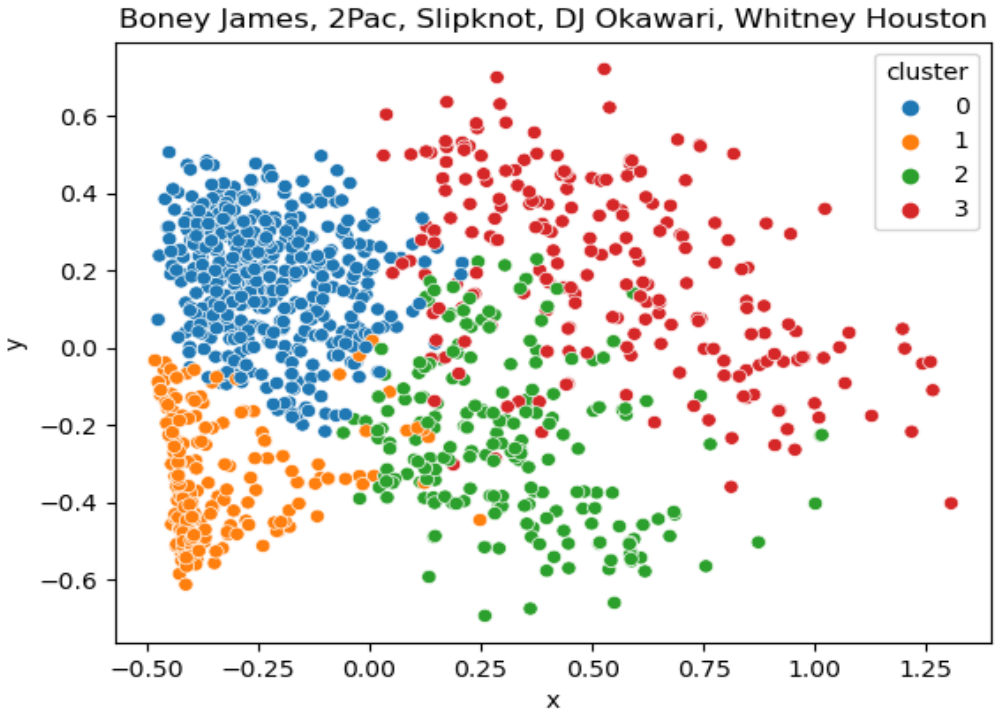
- get_data_API_LIMIT_REACHED.py (non servi lors du lancement du programme) :
C'est une première tentative de programme qui extrait les données voulues, et qui marche, mais on atteint assez rapidement la limite du nombre d'appels API. Le code dans get_data.py contourne ce problème en réduisant le nombre d'appels. get_data_API_LIMIT_REACHED.py n'est pas appelé dans le programme final! On voit ainsi que les appels pour obtenir les audio_features a fait rapidement atteindre cette limite :



- clustering.py, qui permet de :
A partir du fichier CSV généré “artists_data_labels_YES.csv”, ne prendre en compte que les audio features pour pouvoir les séparer en clusters. La méthode d'Elbow est appliquée préalablement afin de trouver le nombre optimal de clusters (ici k=4) grâcé à la génération d'un graphe qui permet de le voir.



Ensuite, puisqu'il y a un grand nombre de features, on fait une dimensionality reduction, qui permettra de réduire l'espace tout en conservant toutes les informations de l'ensemble des features. Ces données seront fittées dans un algorithme de K-Means avec k=4, pour pouvoir visualiser ensuite au travers d'un graphe toutes les chansons séparées en en k clusters. Le code ajoute également une colonne “label” (0, 1, 2, .., k-1) au fichier d'entrée, labellisant chacune des chansons, qui sera bien sûr utile pour le ML.



- classif_model.py, qui permet de couper le dataset en train et test sets, de créer le modèle : un réseau de neurones constitué de 2 couches ayant pour fonction d'activation en dernière couche un Sotfmax car nous avons affaire à une classification de 4 classes dans notre exemple, et de l'entraîner ensuite. Une matrice de confusion est générée. Ce fichier comporte aussi une fonction permettant de tester le modèle prédiction sur une chanson prise au hasard.

Comment utiliser le programme :

L'idée est de lancer le programme par étapes, en commentant/décommentant les fonctions dans le fichier main.
La première étape consiste à récupérer l'ensemble des titres de plusieurs artistes en appelant uniquement 1 (et donc en commentant 2, 3 et 4) et en modifiant artist_name à chaque fois, ainsi, chaque artiste aura un dossier à lui contenant son propre fichier csv. Il faudra alors manuellement récupérer ces fichiers en choisissant ceux que l'on veut intégrer dans la suite du programme en les copiant/collant dans le dossier “artists”. Ne pas oublier de modifier également artists_list.
La deuxième étape consiste à concaténer en un seul fichier csv l'ensemble des données des fichiers csv contenus dans le dossier “artists”.
La troisième étape consiste à clusteriser ces données, et de les labelliser. Cela génère un nouveau fichier csv qui sera utilisé comme dataset pour l'étape du machine learning.
La quatrième étape créé le modèle et s'entraîne sur ce dataset.
La cinquième étape teste le modèle sur une chanson prise au hasard et ne faisant bien entendu pas partie du dataset.

Ce choix du fonctionnement étape par étape a été fait car c'est plus clair et propre, ça évite les confusions, et ça permet de mieux localiser une erreur de process (génération de csv) si il y en a une. Ca permet également de mieux jongler entre différents tests, par exemple en faisant des combinaisons différentes dans le choix des artistes ([metal, rap, classique] ou [metal, disco, classique] etc.), en en gardant plus en moins (1 ou plusieurs artistes), tout cela en ne changeant que très peu les paramètres et tout en gardant des manip simples à faire (comme copier coller dans le dossier artists les données des artistes sélectionnés).

AUTRES FICHIERS :

L'image clustering.png représente les clusters, en se basant sur les audio_features des titres des artistes choisis (ici 5).

artists_data_labels_NO.csv est le fichier intermédiaire contenant les données de plusieurs artistes aux styles musicaux différents, avant le clustering donc sans label.

artists_data_labels_YES.csv est le fichier final contenant les données de plusieurs artistes aux styles musicaux différents, labellisés de 0 à 3 (en 4 clusters), qui servira de dataset pour le machine learning.

requirements.txt comporte tous les packages installés et leur version dans mon système lors de l'élaboration de ce programme.

sp_audio_features.pdf donne des détails sur les audio features.

Whitney Houston.csv est juste un fichier généré par get_data.py pour un seul artiste (pas de labels) donné en guise d'exemple.

Analyse des labels :

Les données des chansons de 5 artistes aux styles bien différents ont été récoltées, transcrites et concaténées dans un même fichier CSV :
Boney James (smooth jazz), 2Pac (rap, hiphop), DJ Okawari (nujazz, hiphop), Slipknot (heavy metal), Whitney Houston (pop).

Ces chansons (1 par ligne) ont ensuite été différenciées en 4 clusters et labellisées [0, 1, 2, 3], en n'intégrant que les features “danceability”, “energy”, etc. dans un algorithme K-Means de la librairie Scikit-Learn. Le choix de k=4 clusters a été déduit par la méthode d'Elbow (présent dans le programme).

A partir d'un comptage des labels pour chaque artiste :

```
artist       label
Boney James  2        120
             1         48
             0         32
             3          0
Name: count, dtype: int64
```

```
artist    label
Slipknot  3        205
          1         12
          2         10
          0          7
Name: count, dtype: int64
```

```
artist           label
Whitney Houston  0        110
                 1        109
                 3          7
                 2          5
Name: count, dtype: int64
```

```
artist  label
2Pac    0        280
        2         18
        3         11
        1          6
Name: count, dtype: int64
```

```
artist      label
DJ Okawari  2        49
            1        36
            0        20
            3         2
Name: count, dtype: int64
```

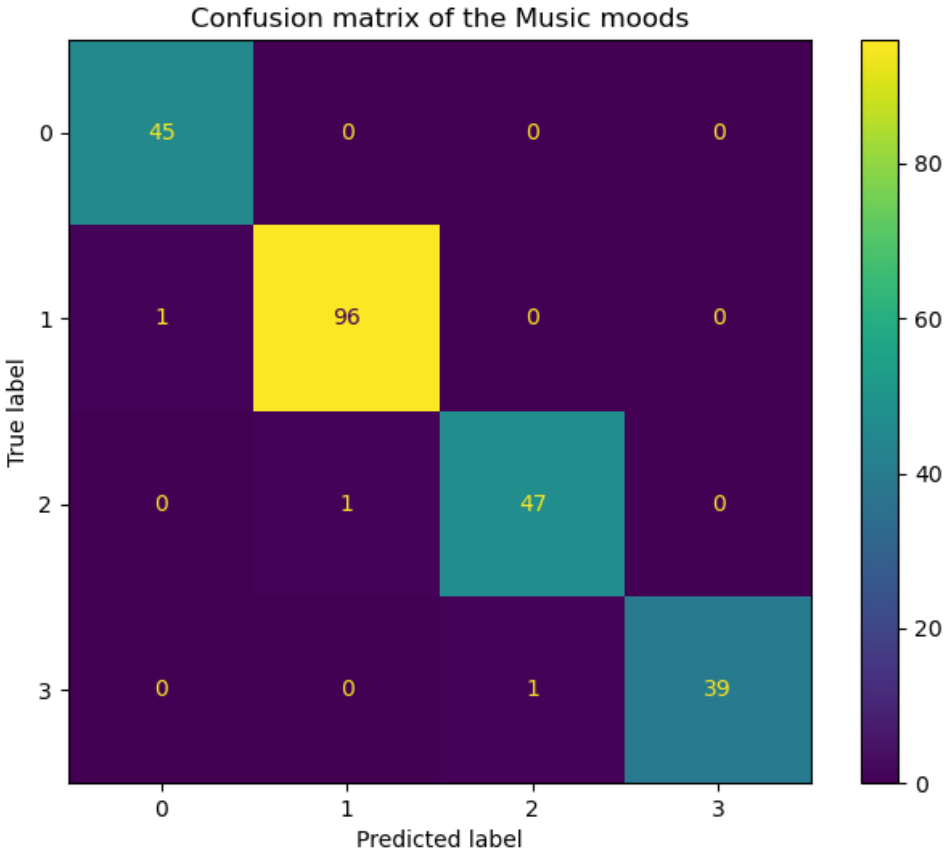
on en déduit certaines prédominances :

Artistes \ Labels	0	1	2	3	Nombre de titres
Boney James	X	X	XX		200
2Pac	XX				315
DJ Okawari	X	XX	XX		107
Slipknot				XX	234
Whitney Houston	XX	XX			231

On pourrait en déduire que (surtout pour les XX) :
Le label **0** pour 2Pac → **paroles, rapidité, positivité, danse**
Le label **1** pour DJ Okawari et Whitney Houston → **tristesse, nostalgie, douceur**
Le label **2** pour Boney James et DJ Okawari → **chill, instrumental**
Le label **3** pour Slipknot → **énergique, bruyant**

Je relève toutefois 2 points douteux :
Le nombre de titres pour chaque artiste n'est pas homogène, ça peut biaiser les résultats en sortie d'algorithme.
Parmis les audio features, certains ont certainement pu être retirés de l'algo : “liveness”, “acouticness” et peut-être “speechiness” n'étaient peut-être pas pertinents pour déterminer l'humeur musicale d'un titre.

La matrice de confusion obtenue à l'issue de l'entraînement semble très satisfaisante :



En prenant une nouvelle chanson au hasard (venant d'un artiste que l'on a pas inclu dans notre training), en l'écoutant au préalable pour avoir une idée du style musical et de son mood, on teste le modèle :
“Spiral” de Nujabes retourne bien (sur la console) un label désignant une musique chill et instrumentale semblable à DJ Okawari, et “Looking up” de Donna Summer retourne un label semblable à celui de Whitney Houston.

Camille – Décembre 2023