

Cours PCD – Labo 7 : Recommandation musicale basée sur le contenu

Contexte, données, objectif

- Nous souhaitons construire un système de recommandation musicale : lorsqu'un utilisateur indique qu'il apprécie un morceau de musique, le système lui recommande d'autres morceaux qui pourraient lui plaire également.
- Nous nous plaçons dans la situation où le système n'a pas encore d'informations sur les goûts des utilisateurs (*cold start*), donc il ne peut pas faire du filtrage collaboratif.
- La solution est de faire des recommandations basées sur le contenu : le système propose des morceaux qui sont semblables à un morceau que l'utilisateur a indiqué apprécier.
- **La ressemblance entre morceaux est calculée ici grâce à la distance entre leurs attributs audio.**
- On fournit les données suivantes pour tester la pertinence des attributs :
 - 8 morceaux de musique au format MP3, numérotés de 0 à 7 ;
 - on suppose que l'utilisateur a aimé le morceau 0 ;
 - vérité-terrain : la similarité avec le morceau 0 décroît du morceau 1 au 7 (cela est subjectif, mais on peut supposer que ce sont les préférences d'un utilisateur donné).

L'objectif principal est de **sélectionner les attributs audio les plus adaptés à cette tâche**. Cela demande de répondre à la question : comment sélectionner les attributs audio qui permettent d'obtenir un classement des morceaux qui se rapproche le plus de l'ordre 1, 2, 3, 4, 5, 6, 7 ?

Indications :

- i) classez les morceaux par valeur croissante de *distance* avec le morceau 0 ;
- ii) cette *distance* est donnée par la distance L2 entre les vecteurs d'attributs ;
- iii) deux *classements* seront comparés grâce à la [corrélation de Spearman](#) (ρ).

Consignes

1. Utiliser l'outil [pyAudioAnalysis](#) et sa [documentation](#) mais aussi directement son code source. Cet outil permet d'extraire des attributs à court terme (petites fenêtres) et à moyen terme. Il permet également d'obtenir des attributs globaux (moyenne et écart-type des attributs à moyen terme), et même de traiter ensemble plusieurs morceaux. *Indication* : cherchez dans le code source la fonction traitant un ensemble de morceaux.
2. Organisez votre code en fonctions. Il sera utile par exemple de définir une fonction qui, étant donné un ensemble de noms d'attributs, retourne la proximité du classement fourni par ces attributs avec le classement de référence.
3. Les méthodes de scikit-learn pour la sélection d'attributs n'étant pas utilisables ici, implémentez directement la recherche incrémentale (du type *Sequential Forward Generation* vu en cours).