Instructions

Rendre un rapport PDF de 4 pages max + codes sous fic(s).py, pour répondre aux sections 1 et 2. La section 3 permet de préparer le TDM5 à venir, sur le clustering : ne mène à aucun rendu ce jour.

Si vous ré-utilisez du code ou du texte, cela doit être clairement lisible dans votre rendu : il faut citer leur source et leur auteur, avec si possible un lien internet s'il existe, ou des références permettant de le(s) retrouver.

Si vous partagez votre code/rapport publiquement, ou de façon privée, signalez le nous (en première page de votre rapport, en indiquant les canaux de diffusion utilisés). Cela peut vous éviter un 0/20 si nous retrouvons – même partiellement – vos sources dans des rendus du MINF1.

Préalable Ouvrir un onglet dans un navigateur internet vers la documentation de `pandas`, notamment `dataframe`. Revoir les CM1 & CM2 et les TDM1 & TDM2.

1 Prise en main des données Unpopular Spotify Songs (USS)

Ce jeu de données compile des milliers de chansons de spotify, avec des indicateurs (attributs) propres à la musique (rythme, dansabilité, harmonique, etc.), des informations factuelles (titre, auteur, etc.), et des informations de popularité.

1. Récupérer le jeu de données `unpopspotify.csv`[1] sous la forme d'un *data frame* en utilisant `pandas` (cf. TP1). Utiliser les fonctionnalités de `pandas` pour étudier les statistiques descriptives de ce jeu de données afin de répondre aux questions classiques : type & stats de chaque colonne, est-ce qu'il y a des données manquantes, quelle est la distribution de chaque colonne, quelles régularités sont observées (colonne par colonne), etc. Pensez-vous qu'il soit opportun de réaliser une standardisation (et pourquoi ?), etc. ? Est-ce que certaines colonnes sont inutiles pour étudier les régularités ?
2. Déterminer la matrice de corrélation (linéaire) : les colonnes `popularity` et `key` sont-elles (même partiellement) correlées avec toutes les autres ? Autres commentaires sur ce que cette matrice révèle ?

2 Préparation des données et classification

Nous cherchons ici à apprendre un bon classifieur qui, à partir de certaines colonnes pertinentes, permette de modéliser le lien entre la colonne `popularity` (cible y, de 0 à 11) et les autres.

1. Caractériser le problème de classification (supervisée ou pas, si oui nombre de classes et équilibre de distribution de chaque classe $P(C)$)
2. Enlever du `dataframe` les colonnes estimées inutiles pour ce problème de classification (justifier lesquelles et en quoi elles sont considérées inutiles)
3. Si besoin, compléter les colonnes pour qu'il n'y ait pas de valeurs manquantes.
4. Si elle n'a pas été enlevée, utiliser les fonctionnalités de `pandas` pour remplacer la colonne `explicit` par une colonne dont les valeurs sont 0 et 1.
5. Modifier le jeu de données pour standardiser les colonnes dont les valeurs sont réelles, (*cf.* `https://www.geeksforgeeks.org/how-to-standardize-data-in-a-pandas-dataframe/`)
6. Par validation croisée, indiquer les performances d'un classifieur de type *arbre de décision* de profondeur max 5, en termes d'erreur et de temps d'apprentissage.

1. Credits Etienne GCC, `https://www.kaggle.com/datasets/estienneggx/spotify-unpopular-songs/data`

7. Par validation croisée, indiquer les performances d'un classifieur de type *k-plus proches voisins* avec $k = \log(n)$ où n est le nombre d'exemples d'apprentissage, en termes d'erreur et de temps d'apprentissage. Quelle mesure de distance a été utilisée ?
8. Conclure sur les capacités de prédire la popularité d'une musique en fonction des colonnes conservées (à quel point cela relève du hasard, ou au regard des perf. de ces deux seuls algorithmes, est-il possible d'imaginer un modèle plus complexe qui améliore ceux testés, etc. ?)
9. Visualiser l'arbre de décision de profondeur max 5 appris sur toutes les données : quelles sont les 6 colonnes les plus discriminantes ? Quelles sont les colonnes absentes ? Qu'en conclure ?
10. A partir de la réponse à la question précédente, évaluer par validation croisée les performances d'un k-ppv seulement sur les colonnes les plus discriminantes, en choisissant le meilleur k. Quel k est le meilleur ? Remarque-t-on une amélioration wrt ce qui avait été appris sur toutes les colonnes , et si oui de quelle ordre ? Et sinon, pour quelles raisons les plus probables (à imaginer) ?
11. Comment peut-on facilement transformer ce problème de classification en un problème de régression ? Quelle performance donnerait alors les moindres carrés, en terme de coefficient de Pearson ?

3 Classification non-supervisée (préparation TDM5)

Ici, on cherchera à identifier des regroupements de chansons aux caractéristiques similaires, et à étudier ces regroupements au regard des caractères de non-popularité (colonne `popularity`), et au regard des octaves/harmoniques des chansons (colonne `key`).

1. Produire un *dataframe* issu de l'initial, qui enlève les colonnes `popularity` et `key` (tout en conservant ces informations que nous utiliserons plus tard comme supervision), et sans toutes les colonnes enlevées dans la section précédente. Quelles sont les colonnes descriptives restantes ?
2. Nous voulons retrouver les classes de non-popularité à partir des autres colonnes. Combien y en a-t-il ? Quel est le nombre de clusters que nous voulons définir ?
3. Lancer l'apprentissage d'un clustering sur toutes les données, avec autant de clusters que nécessaires pour résoudre ce problème (justifier ce nombre). Quels sont les mesures pour évaluer la qualité du clustering obtenu (1. comme si nous avions accès à l'impopularité de chaque chanson, et 2. comme si nous n'y avions pas accès)

4 Annexe : dataset description

See the source `https://www.kaggle.com/datasets/estienneggx/spotify-unpopular-songs/data`