Chellenge Mewo - Apprentissage automatique

Interprétation des prédictions de réseaux de neurones pour la classification multi-label de catalogues musicaux.

Réalisé par

Hamza Benmendil
Jalal Izekki

---

17 décembre 2021

1 Introduction

Mewo est un service de gestion de catalogues musicaux. Le service se base sur un modèle qui permet de récupérer des informations musicales pour la recommandation audio et la classification *multi-label* automatique, celui-ci est basé sur un pipeline d'algorithmes de traitement du signal et de réseaux de neurones profonds. Le modèle produit des prédictions numériques dans l'intervalle $[0, 1]$ pour chaque étiquette sur laquelle il a été entraîné. Cependant, pour l'utilisation finale, une décision doit être prise d'associer ou non une piste audio à une étiquette. Cela peut être fait par un simple processus de *thresholding* (c'est-à-dire de comparer la prédiction numérique à une valeur fixe), mais des algorithmes plus élaborés pourraient être utilisés pour augmenter la qualité de l'étiquetage. Le but de ce projet est de tester différentes approches pour améliorer la précision de la classification *multi-label* des catalogues musicaux. Dans ce rapport, nous présenterons une analyse plus détaillée du sujet, les différentes expériences menées et enfin une analyse des différents résultats obtenus.

2 Analyse du sujet

2.1 Contexte

Récemment, la classification *multi-label* est apparue dans de nombreuses applications. Contrairement à la classification de données traditionnelle, où chaque instance est affectée à une seule étiquette, pour la classification *multi-label*, une instance peut être simultanément pertinente pour plusieurs étiquettes. Par exemple, dans la catégorisation de catalogues musicaux, un morceau peut appartenir à plusieurs rubriques.

Le sujet sur lequel on travaille, consiste à appliquer plusieurs méthodes de classification *multi-label* sur des données issues d'un model basé sur un pipeline d'algorithmes de traitement du signal et de réseaux de neurones profonds. Celui-ci prend en entrée des morceaux musicaux et produit des prédictions numériques dans l'intervalle $[0, 1]$ pour chaque label. Les labels possibles comprennent quelques centaines de tags (`ex. blues-rock, electric-guitar, happy`), regroupés en classes (`Genres, Instruments ou Moods`), partitionnés en catégories (`genres-orchestre, instruments-cuivres, mood-dramatic, ...`). Chaque piste audio de la base de données peut être étiquetée avec une ou plusieurs étiquettes de chaque classe, de sorte à ce que le processus d'étiquetage automatique soit un problème de classification *multi-label*.

2.2 Outils d'évaluation

Afin d'évaluer les prédictions obtenues, celles-ci seront comparées avec les vraies valeurs en comparant la valeur de **weighted F1-score**. Celui-ci peut être interprété

comme une moyenne harmonique de la précision et du rappel. Un score F1 atteint sa meilleure valeur à 1 et le pire score à 0.

3 Démarches entreprises et expérimentations

La classification *multi-label* est utile pour la catégorisation de texte, de la classification multimédia et dans de nombreux autres domaines. Une approche *multi-label* couramment utilisée est la méthode binaire, qui construit une fonction de décision pour chaque étiquette. Pour certaines applications, l'ajustement des seuils dans les fonctions de décision de la méthode binaire améliore significativement les performances.

Les expériences suivantes ont été menées sur les données des catalogues musicaux présentées dans la partie précédente afin d'étudier l'utilité de certaines stratégies de sélection simples.

3.1 Test de plusieurs *thresholds* pour chaque label

Cette méthode consiste à calculer des seuils (*thresholds*) pour lesquels la valeur de `f1_score` des décisions est maximale. En effet, les données fournies sont des probabilités de chaque piste musicale (ligne des données) qu'elle soit étiquetée par un label (colonne des données). Ainsi, le but de cette méthode est de tester un intervalle de seuils entre 0 et 0.99 avec un pas de 0.01, calculer la valeur de `f1_score` des décisions de chaque seuil, et renvoyer le seuil qui a le plus grand score. Ce processus est appliqué sur toutess les colonnes des données jusqu'à construire un tableau des seuils, un seuil de l'indice i correspond à la colonne du même indice. La prédiction se fait en comparant les seuils avec les probabilités des pistes pour chaque label, si la probabilité est inférieure au seuil on considère que la décision de labellisation est 0 pour cette colonne, 1 sinon.

3.2 Utilisation du classificateur OneVsRestClassifier

Dans cette deuxième approche, nous voulons utiliser un classificateur de type régression logistique. Cette méthode étant destinée à la résolution des problèmes de classification *multi-class*, nous avons utilisé une strategie nommée `OneVsRestClassifier` avec la régression logistique comme objet d'estimation. En fait, cette stratégie consiste à ajuster un classificateur par classe. Pour chaque classificateur, la classe est comparée à toutes les autres classes, ce qui signifie qu'on divise la classification *multi-label* en un problème de classification binaire. Puisque chaque classe est représentée par un et un seul classificateur, il est possible d'acquérir des connaissances sur la classe en inspectant son classificateur correspondant. C'est la stratégie la plus couramment utilisée pour la classification *multi-label* et c'est un choix par défaut juste.

3.3 Test de plusieurs autres classificateurs

Pour pouvoir comparer plusieurs algorithmes d'apprentissage automatique pour la classification *multi-label*, nous avons mis en place un algorithme qui itère sur une liste de modèles. Pour chaque modèle, nous utilisons une partie des données de l'entrainement (pour des raisons de complexité de calcul) et appliquons la méthode de `k-fold cross-validation` pour entraîner le modèle et le tester plusieurs fois sur des portions différentes de données à chaque fois, ceci nous permet d'avoir plusieurs valeur de `f1_score`. En calculant la moyenne des scores, nous pourrons classifier les modèles selon leur précision. Ensuite, nous pourrons entraîner le meilleur modèle sur la totalité des données d'entraînement pour pouvoir l'utiliser pour générer les prédictions sur les données de test.

4 Analyse des résultats et bilan

4.1 Evaluation de la méthode de selecion de *thresholds*

Afin d'évaluer l'efficacité de la méthode de selection de *thresholds* qui donnent les meilleurs `f1-score` sur les données d'entrainement. Nous avons suivi l'approche suivante :

- Diviser les données en deux parties : une pour l'entraînement et l'autre pour la validation.
- Pour chaque *tag*, sélectionner la valeur du *threshold* qui donne le meilleur `f1-score` sur la première partie des données
- Utiliser les valeurs de *threshold* obtenues pour effectuer les prédictions sur la deuxième partie des données
- calculer pour chaque *tag* la valeur de `f1-score` en comparant les prédictions obtenues avec les vraies valeurs.

Les résultats obtenus ont été utilisés pour tracer le diagramme à barres de la figure 1. Nous avons également tracer la moyenne des valeurs de `f1-score` pour cette méthode ainsi que pour la méthode de *benchmark* (qui consiste à appliquer un *threshold* fixe de 0.5).

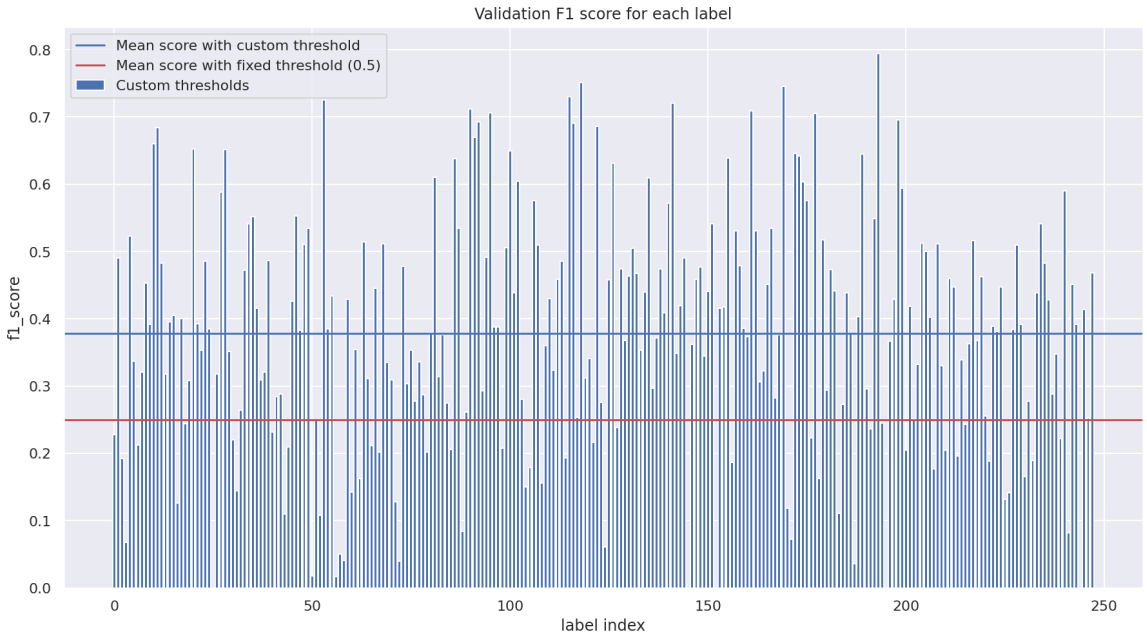

FIGURE 1 – F1-score pour chaque *tag* avec la méthode de la partie 3.1

L'analyse de cette figure montre que la méthode d'adaptation des *thresholds* obtient un score moyen de 0.38 alors que la méthode du *benchmark* obtient un score de 2.5. Il s'agit d'un progrès remarquable, mais qui peut encore être amélioré. En fait, on peut remarquer que pour certains *tags*, la valeur de *threshold* qui donne le meilleur `f1-score` sur les données de test donne une valeur très petite sur les données de validation.

Cette méthode fournit le meilleur score pour les données de test fournies par la platforme challenge data qui est de 0.4515.

4.2 Évaluation de la methode OneVsRest

L'utilisation du classificateur `OneVsRest` avec *l'estimator* `LogisticRegression` donne un score d'environ 0.36 sur le leaderbord. Cette méthode est une méthode de transformation qui transforme le problème *multi-label* en un ensemble de problèmes de classification binaire, qui peuvent ensuite être traités à l'aide de classificateurs à une seule classe. Le problème de cette approche est que le nombre des classificateurs binaires augmente exponentiellement avec le nombre des labels, ce qui implique une complexité exponentionnelle de calcul.

4.3 Bilan

Nous avons utilisé plusieurs approches pour la prise de la décision pour ce problème *multi-label*, que ce soit des algorithmes d'apprentissage automatique de classification ou des approches algorithmiques qui maximisent le `f1_score`. Chaque méthode a ses avantages et ses inconvénients, la majorité des approches de l'apprentissage automatique de classification prennent beaucoup de temps pour entraîner un modèle, ce qui n'est pas toujours efficace. Ainsi, une approche qui pourrait améliorer les résultats et que nous n'avons pas testé est d'utiliser des réseaux de neurones profonds de classification, ou bien des `LSTM`. En plus, nous pourions également utiliser les colonnes représentant les catégories dans l'entraînement de nos modèles, et comparer les résultats des prédictions avec les anciens résultats.