KKBox's Music Recommendation Kaggle Challenge

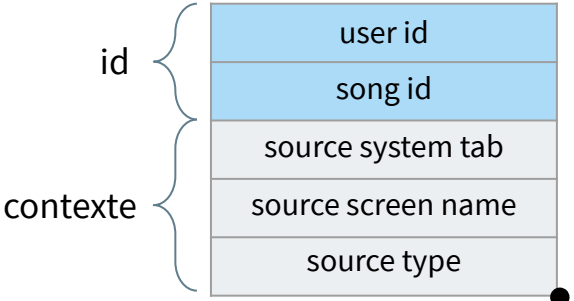
DERVAUX Gautier

MENGHETTI Marina

Plan de la présentation

Introduction	Modèles et évaluations	Conclusion et améliorations
1. Données 2. Objectif	1. Baseline 2. Item-Item 3. CDL (Collaborative Deep Learning) - Sans contexte - Avec contexte	Possibles améliorations futures

Première écoute de la chanson par l'utilisateur



Informations sur les utilisateurs
- id
- ville
- age (possibles erreurs)
- genre
- méthode d'inscription
- date d'inscription
- date d'expiration

1 mois plus tard

Date

Informations sur les chansons
- id
- longueur (ms)
- genre (multiples genre possibles)
- artiste
- compositeur
- parolier
- langue
- nom

On doit prédire une probabilité.
L'utilisateur a-t-il réécouté la chanson pendant le mois ?

Objectif :
Maximiser l'aire sous la courbe ROC (**AUC**)
On fait varier le seuil pour classifier en classe positive ou négative

KKBox : leader asiatique du streaming de musique

Challenge Kaggle (2018)

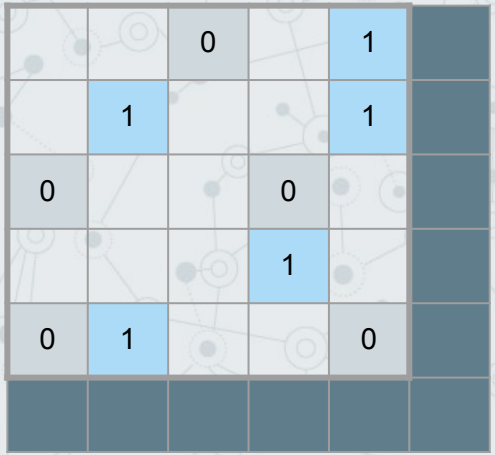
https://www.kaggle.com/c/kkbox-music-recommendation-challenge

Dataset énorme

Matrice de votes

Chansons

Utilisateurs



Environ autant de 0 que de 1 dans les votes
⇒ **Peu de biais**

Entraînement sur 10% des votes
(sinon trop long, mais matrice encore plus sparse)

360 000 chansons ont des votes
2 300 000 chansons en tout

30 000 utilisateurs ont des votes
34 000 utilisateurs en tout

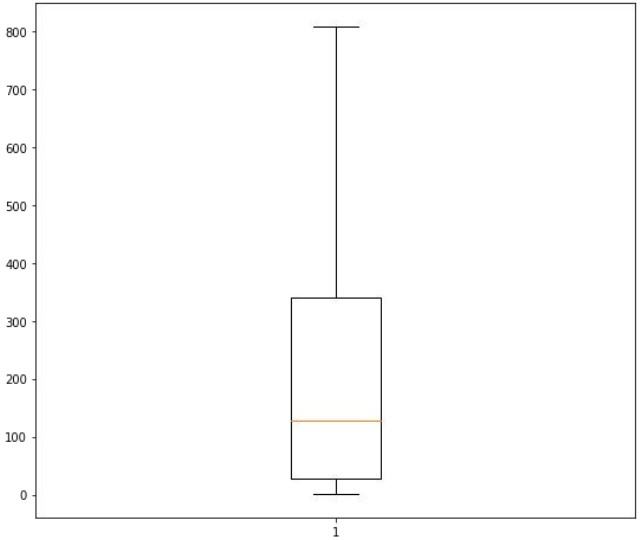
7 400 000 votes en tout

Sparsity de la sous-matrice :
99.93%

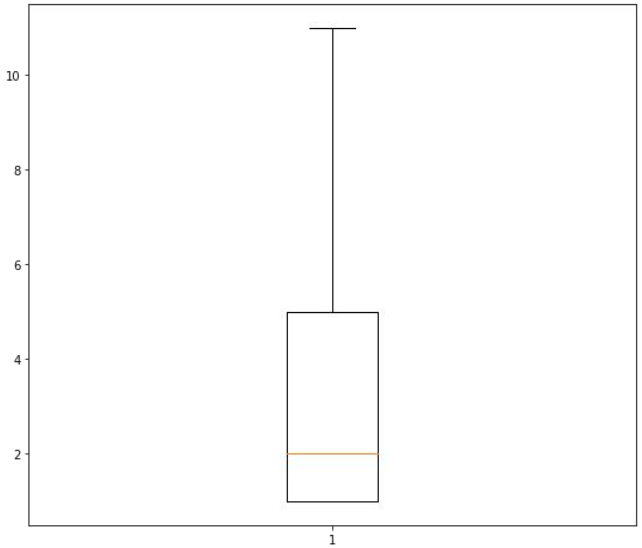
2 500 000 valeurs à prédire en test *(soumission)*

⇒ Utilisation d'un **ensemble de validation** pour nos résultats

Visualisation des données



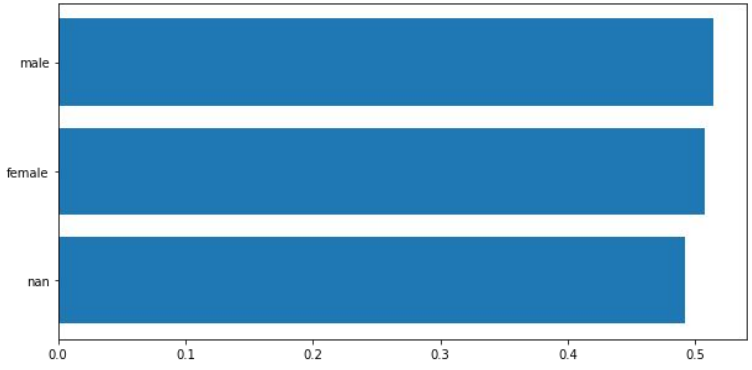
Répartition du nombre de vote par utilisateur



Répartition du nombre de vote par chanson

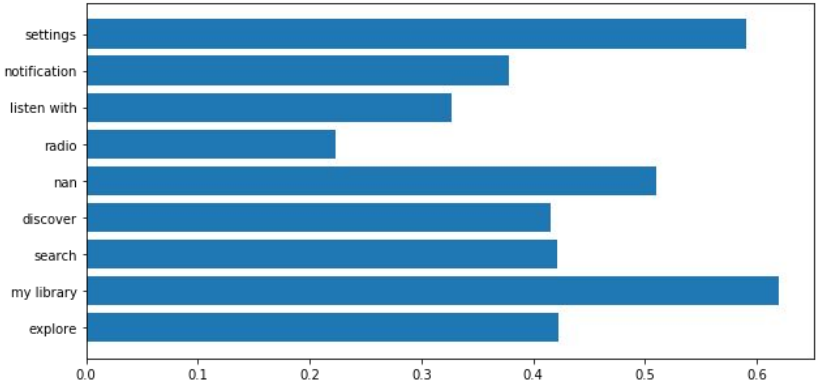
Visualisation des données - Influence des facteurs sur les votes

Influence du genre des utilisateurs



Vote moyen

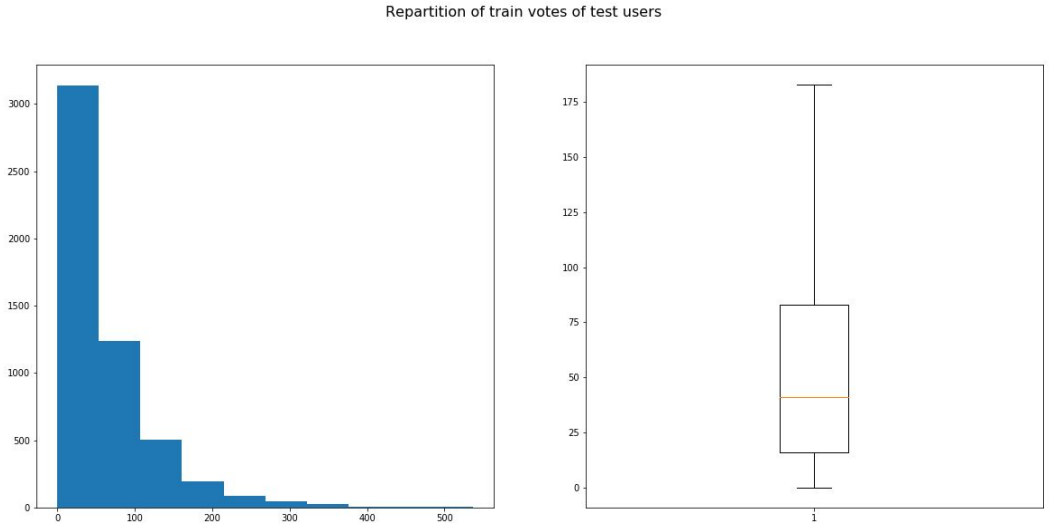
Influence du contexte de première écoute



Vote moyen

Séparation entraînement et validation

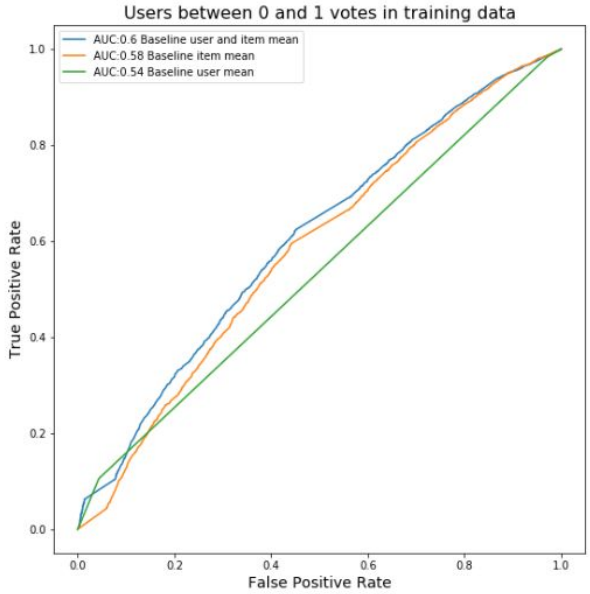
Train | Test

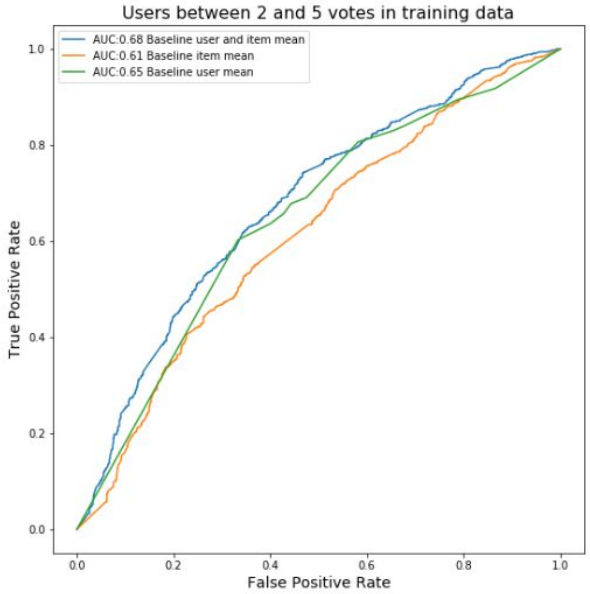


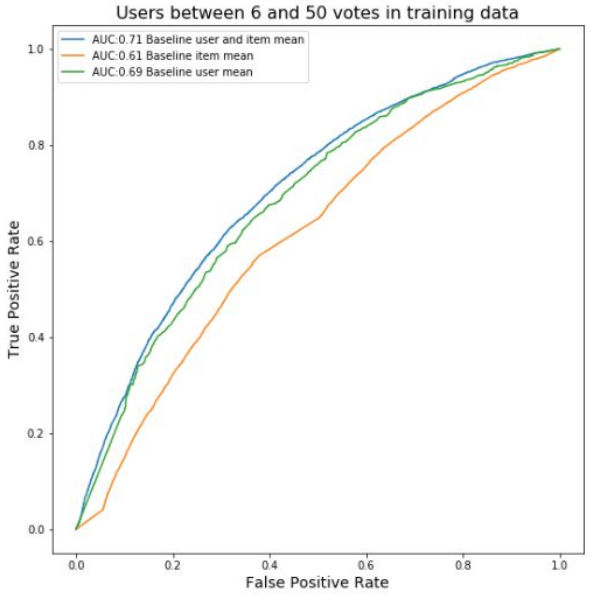
Baseline

Mean(Mean(User) & Mean(Item))
Mean(Item)
Mean(User)

Meilleure baseline
AUC = 0.6968







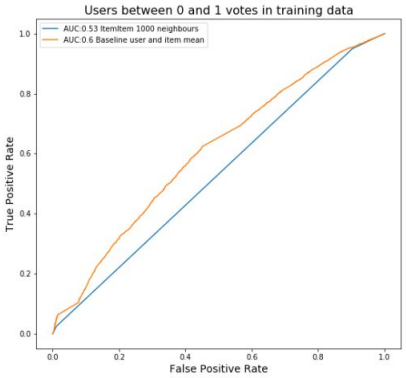
Cold start utilisateur ⟷ Plus de “votes” connus

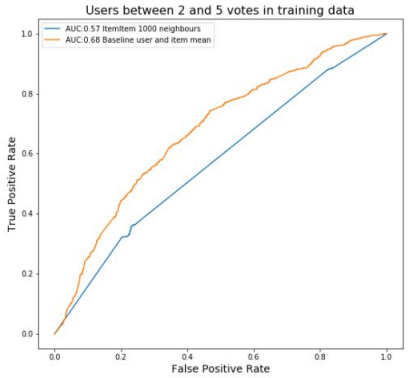
Meilleure Item-Item
AUC = 0.6525

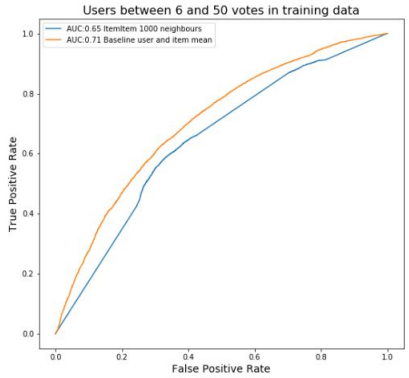
Item-Item (1000 voisins)
Meilleure baseline

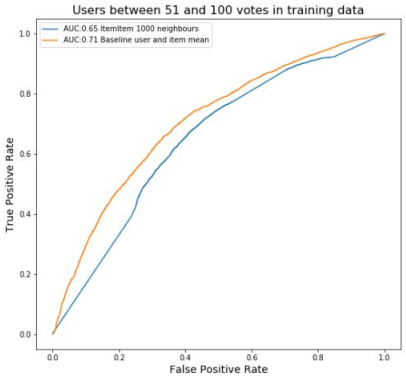
Item-Item

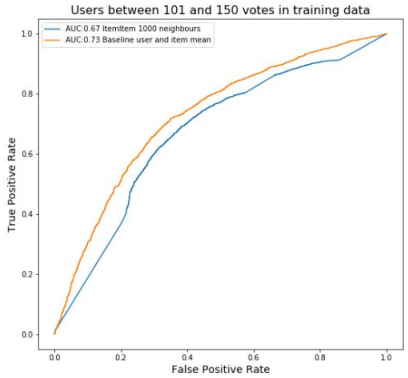
Cold start utilisateur

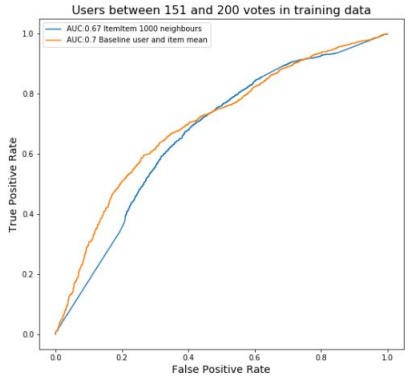






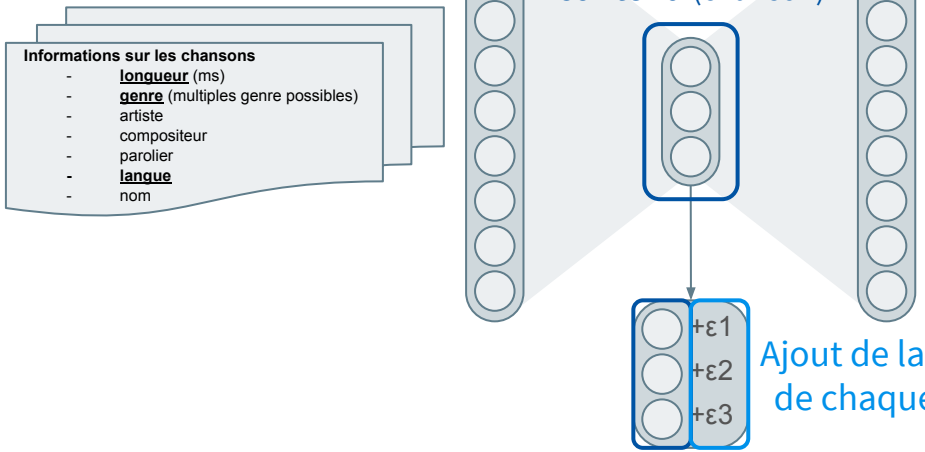






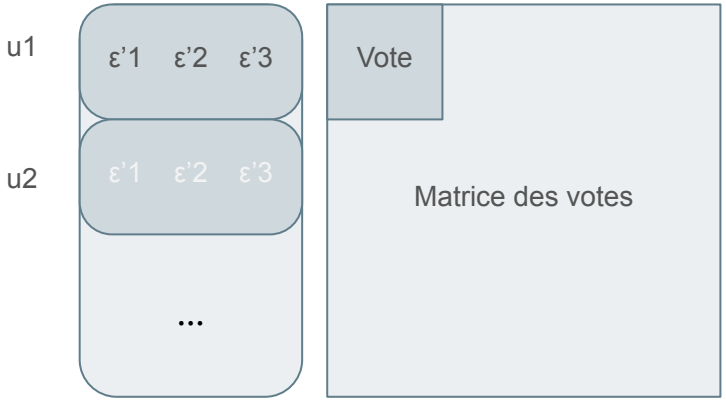
Plus de “votes” connus

CDL - Sans contexte



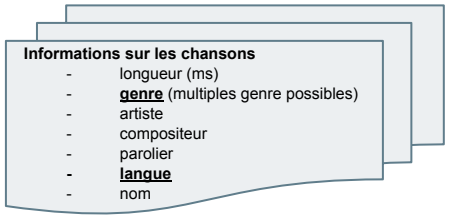
Denoising auto-encoder
⇒ Objectif de reconstruction

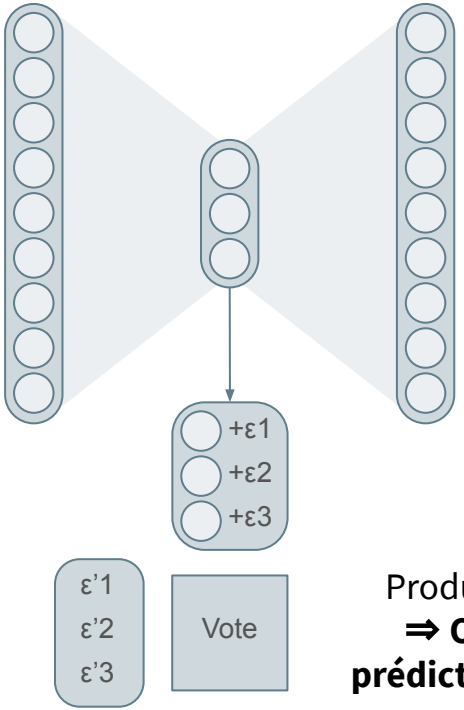
Note :
Les ε sont spécifiques à chaque chanson.
Les ε' sont spécifiques à chaque utilisateur.



Produit matriciel
⇒ Objectif de prédiction des votes

CDL - Sans contexte

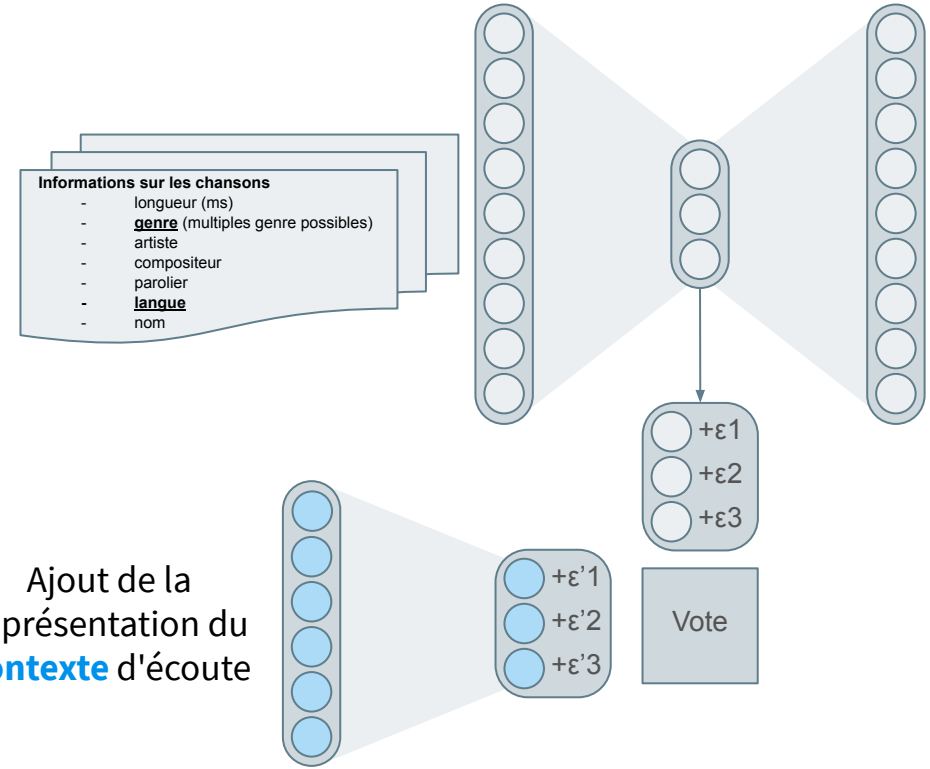




Denoising auto-encoder
⇒ Objectif de reconstruction

Produit matriciel
⇒ Objectif de prédiction des votes

CDL - Avec contexte



source system tab
source screen name
source type

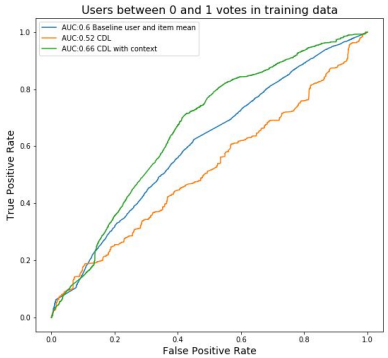
Ajout de la représentation du **contexte** d'écoute

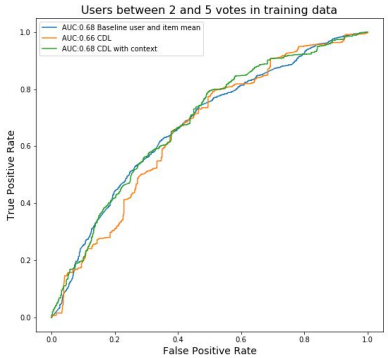
CDL

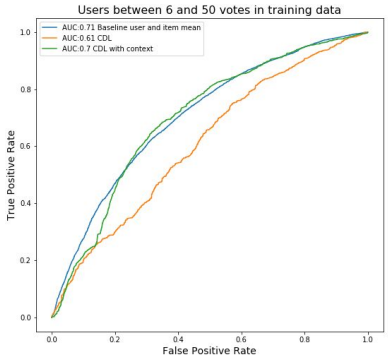
Meilleure baseline
CDL sans contexte
CDL avec contexte

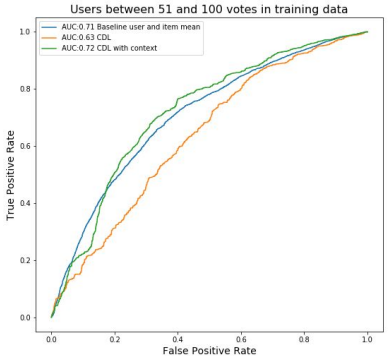
Meilleure CDL
AUC = 0.6979

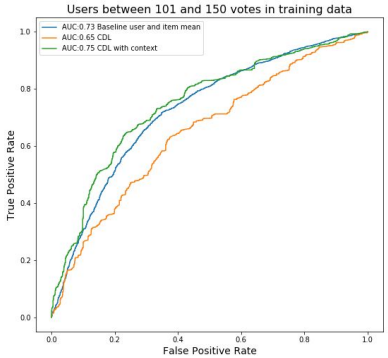
Cold start utilisateur

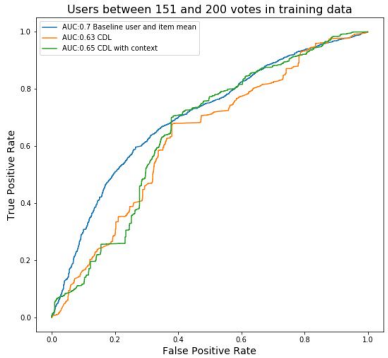












Plus de “votes” connus

Résumé des résultats

Meilleure baseline
AUC = 0.6968

Meilleur Item-Item
AUC = 0.6525

Meilleur CDL (avec contexte)
AUC = 0.6979

Première place compétition
AUC = 0.7478

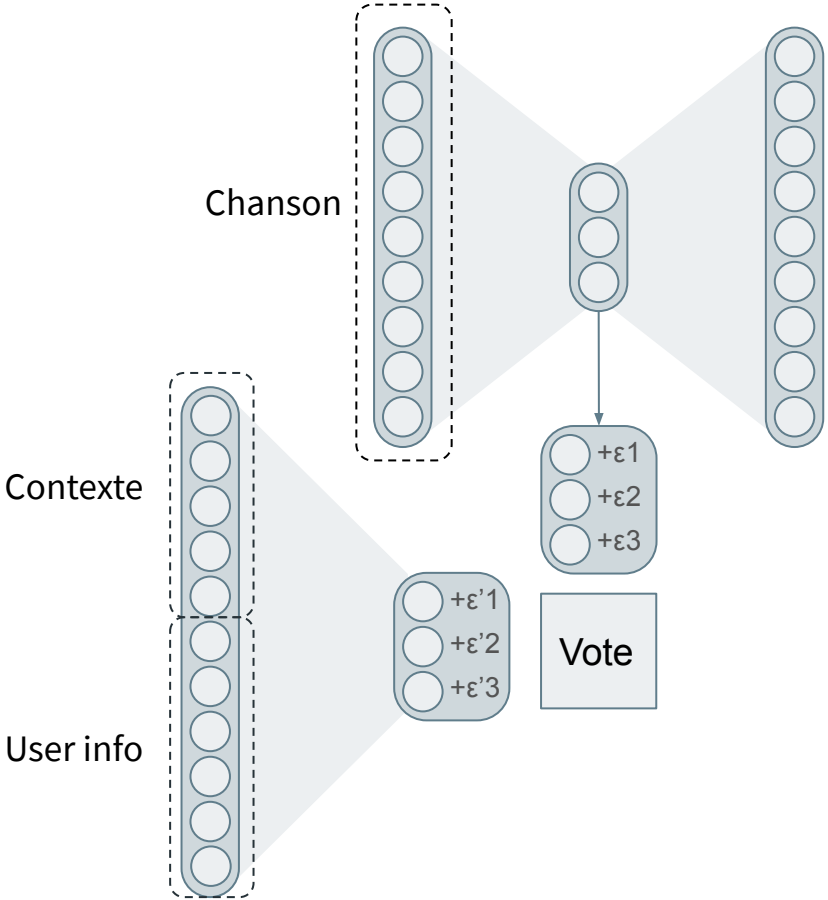
47/1000 teams au-dessus de
AUC = 0.7

- Tenter une soumission sur l'ensemble des données (pour avoir une idée de la position de nos résultats, mais demandera une tonne de RAM et des heures et des heures de calculs)

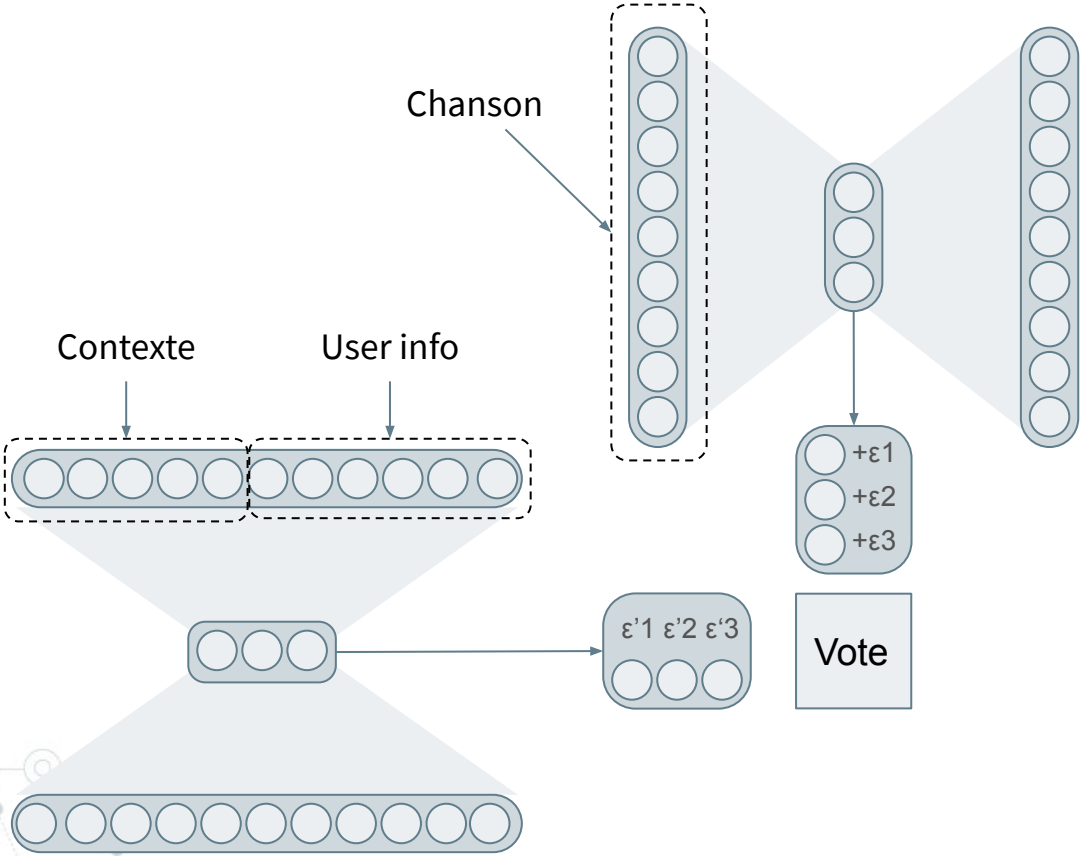
Améliorations possibles *(ou ce que nous aimerions tester si nous avions plus de temps)*

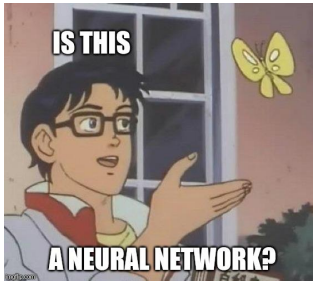
- Améliorer le pre-processing
  - Améliorer le one-hot encoding (notamment sur les genres de musiques avec multi-labels)
  - Vérifier les noms d'artiste, compositeur, parolier et autres... (sans fautes d'orthographe, pas de doublons inutiles dans nos données)
  - Prendre en compte les noms des chansons, etc... (même si les embeddings de noms en chinois sont difficiles à obtenir, et que l'on est pas sûrs que cela ait une grande influence)
- Améliorer l'architecture
  - Tester d'autres prises en compte du contexte (concaténation avec l'utilisateur puis réduction de dimension pour matcher la dimension des facteurs latents, par exemple)
  - Prendre en compte la représentation de l'utilisateur
  - Tenter un auto-encoder pour apprendre l'utilisateur et le contexte
  - Tenter de faire varier d'autres paramètres du CDL (comme le bruit, régularisation, part de l'objectif de reconstruction par rapport aux ratings, le nombre de facteurs latents, de couches, etc…)

Améliorations de l'architecture - Utilisation des infos des utilisateurs



Améliorations de l'architecture - Un 2nd objectif de reconstruction





Références

- Challenge Kaggle KKBox's Music Recommendation : https://www.kaggle.com/c/kkbox-music-recommendation-challenge
- Wang. and Blei. 2011. Collaborative Topic Modeling for Recommending Scientific Articles
- **CDL** : Wang, H., Wang, N., & Yeung, D. Y. (2015, August). Collaborative Deep Learning for recommender systems. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1235-1244)
- Une implémentation du CDL en Keras que nous avons utilisée et modifiée pour les besoins du projet : https://github.com/zoujun123/Keras-CDL/blob/master/

Questions ?