# Travaux Pratique n°5 :

Système de recommandation de films


Cédric Cognard
cedric.cognard@etu.univ-tours.fr

Paul Van-Elsue
paul.vanelsue@etu.univ-tours.fr

# 1. Introduction

Nous allons chercher à implémenter des systèmes de recommandation de films et à évaluer leur performance. Dans un premier temps nous implémentons un prédicteur random qui détermine aléatoirement le film à recommander puis nous implémentons un prédicteur simple pour vérifier que les résultats soient meilleurs.

Nous devrions ensuite en implémenter d'autres mais par manque de temps nous ne l'avons pas fait.

# 2. Lecture des données

Dans un premier temps, voici le code nous permettant de lire les données depuis le fichier concerné :

```
def readdata(fileParam, nbuser, nbfilm):
  res = np.full((nbuser, nbfilm), -1)
  res = res * -1
  file = open(fileParam, 'rU')
  lines = file.readlines()
  for line in lines:
     lineSplit = line.split("\t")
     userid = int(lineSplit[0]) - 1
     filmid = int(lineSplit[1]) - 1
     score = int(lineSplit[2])
     res[userid, filmid] = score
  return res
```

# 3. Prédicteur aléatoire

Pour le prédicteur aléatoire nous donnons un score aléatoire aux films. Une fois terminé nous affichons le score RMSE pour voir nos résultats.

```
def compute(self):
  scorealea = 1 + (np.random.random() * 4)
  RMSE = rmse(self.VOTE_COUNT, self.data, scorealea)
  print("Random predictor RMSE :", RMSE)
```

Voici le calcul du score RMSE, qui suit la formule indiquée dans l'énoncé :

```
def rmse(C, rui, rui2):
```

```
    return np.sqrt(np.sum(np.square(rui - rui2)) / C)
```

Nous obtenons le score suivant :

```
Random predictor RMSE : 9.81168494771015
```

Le but est maintenant de chercher à minimiser ce score.

# 4. Prédicteur basique

Dans le modèle basique, on suppose que chaque utilisateur note tous les films qu'il a regardé avec un écart à la moyenne qui lui est propre. Aussi, chaque film a un écart à la moyenne qui lui est propre. On obtient alors un prédicateur qui donne des notes attribuées par chaque utilisateur à chaque film.

Dans un premier temps, nous avons donc besoin de pouvoir calculer le score moyen :

```
def average_score(fileParam):
    score = 0
    nb_lines = 0
    file = open(fileParam, 'rU')
    lines = file.readlines()
    for line in lines:
        lineSplit = line.split("\t")
        score += int(lineSplit[2])
        nb_lines += 1
    return score / nb_lines
```

Ensuite, nous appliquons la formule de l'énoncé :

```
def compute(self):
    r = average_score(self.file)
    vu = np.full(self.USER_COUNT, self.MOVIE_COUNT)
    vi = np.full(self.MOVIE_COUNT, self.USER_COUNT)
    bu = (np.sum(np.sum(self.data, axis=1)) / vu) - r
    bi = (np.sum(np.sum(self.data, axis=0)) / vi) - r

    rui = []
    for x in range(self.USER_COUNT):
        rui.append(r + bu[x] + bi)
    rui = np.asarray(rui)
    RMSE = rmse(self.VOTE_COUNT, self.data, rui)
    print("Basic predictor RMSE : ", RMSE)
```

Et voici le score obtenu :

```
Basic predictor RMSE :  12103.167057044953
```

On voit très aisément que le score, censé diminuer, a énormément augmenté. Il y a donc une erreur quelque part et nous n'avons ni réussi à la localiser ni à la résoudre. Nous avons perdu trop de temps dessus et nous aurions du passer à la suite plus tôt.