P2

In this problem we will explore a real world example to apply machine learning concepts to.

Describe data set and GameHaven

One method we will use to interact with the data is Collaborative Filtering. Collaborative Filtering is a method used to make recommendations to a user based on their preferences and tastes compared with a data set of many users. This method is helpful for GameHaven to recommend board games to their users based on their tastes.

The initial dataset of GameHaven users and their properties is created from a pattern + randomization method. This ensures that data will be predictable but random across the users. Similarly a matrix of game data is created with the same attributes as the users.

From the game and user attribute matrices, a random sparse ratings matrix is generated. The rating is based on how well the user attributes match the game attributes. Additionally to mimic a real world scenario the data is sparse so there are very few initial ratings.

The collaborative filtering system demonstrated here is done in the following way:

1: Cluster users based on their characteristics

First use SVD to determine a good rank for the user matrix, and use that value for the number of clusters.

2: Create similarity matrix between users in the same cluster

Collaborative filtering uses a similarity function. The goal of a similarity function is to measure how similarity between vectors **x** and **y.** In our example we are using cosine similarity.

A similarity matrix **S** can be constructed from **X** where  where .

3: User similarity matrix to fill in missing data in the ratings matrix

The ratings matrix maps the ratings of different games by users, so its size is the number of games by number of users. **R(I,j)** is the rating that user **i** has for game **j.** As mentioned before, most of **R** is zeros since most users do not have ratings for most games. To estimate the ratings for a user based on the given data, we do the following:

1. Extract the ratings of all the players in the same cluster as the given user **R1 = R(user in cluster 1).**
2. Fill in ratings matrix using 3 different collaborative filtering methods as follows:
   1. Unweighted cluster average
   2. Weighted cluster average
   3. Unweighted full average
   4. Weighted full average
3. Compare result, error and timing across 4 methods.

4: Sort ratings matrix and provide user top 10 recommended games

Questions:

1. What is the best number of clusters to represent this data set?
2. Implement the similarity\_matrix.m function
3. What is the highest recommended game for user # 15 using each different method?
4. Compare the timing of each method. Which method would be best for scaling?