Analysis of Movies Recommender Algorithm

➢ Introduction:

In this project, we are building a movie recommendation system using item-based collaborative filtering and to some extent content-based taking genre under consideration.

➢ Idea behind our code:

What we are doing here is converting our data into some coordinate form. All movies have some genre type. In total there are 20 genres, so we define an array of 20 integers which can be either 0 or 1. We use this matrix to define the genre of our movies.

We also take ratings into considerations. We calculate the average ratings of each movie and modify our data accordingly to store in the same database using a specific pandas function.

Nearest neighbors algorithm can be used here in an efficient way to solve our problem. The approach used in our algorithm uses both the genre and average rating information.

Suppose user gives the name of the movie he likes, then there is a vector associated with it which is an interpretation of its genre type. We use Cosine measures to calculate the closeness of the movie input with other movies and sort them in ascending order of the cosine measure between two movie vectors.

When we get array of neighbors and respective angles between movies we add the normalized difference of ratings to it and sort the array again. The reason we are doing this is because the sole purpose of recommender system is to recommend movies of similar genres and ratings, not the highest rated movie of that genre.

➢ Analysis of other Algorithms:

1. Taking Euclidean distance instead of cosine measure:
    a. Euclidean distance is unhelpful in higher dimensions because all vectors are almost equidistant to the search query vector.
    b. KNN's performance will suffer from 'The Curse of Dimensionality'.
    c. The common thing of these problems is that when the dimensionality increases the volume of space increases so fast that the available data becomes sparse. In high dimensional data, all objects appear to be sparse and dissimilar in many ways, which prevents common data organization strategies from being efficient.

2. Multiplying cosine measure with difference of ratings instead of adding:
    a. A movie with a totally different genre but almost similar rating will be much higher up in list of recommended movies.

3. Dividing cosine measure with rating of movie:
    a. This will give priority to movies with highest rating not to the movies of ratings similar to that of the searched query.

Team 3:
Raj S. Jagtap - 170030004
Abhinav Kumar - 170010022
Paras Yadav - 170010007
Mehul Saxena - 170010017
Rahul Salunke - 170010002
Ajay Meena - 170010013