MOVIE RECOMMENDATION SYSTEM

MOTIVATION

We have ratings file that contains user id, movie id and their respective rating for the movie. We are required to predict the rating that the user might give. The intuition behind these predictions is that, whichever unseen movies are predicted to get higher ratings can be recommended to the specified user.

The very naïve kind of recommendation system would be to simply assign the overall average rating to be the predicted rating for any given user. However this system doesn't take the attributes and tastes of the user into account. Therefore, a recommendation system that can look into other similar users and the taste of the user into account would make better predictions of ratings.

NAÏVE RECOMMENDATION SYSTEM

Firstly lets look at the prediction made by a very naïve system on 100k data set.
In this we will simply take the average of the movie to be predicted over all the available ratings in the training set.
For example:

User Id	Movie ID	Rating
1	10	5

If we have the above test record, we will scan all the rating values for movie number 10 and assign the mean to be the predicted rating for user ID 1.

We will use the following given formula for evaluation of predictions:

$$\text{MAD} = \frac{1}{\sum_i \sum_j r_{ij}} \cdot \sum_i \sum_j r_{ij} \cdot |p_{ij} - t_{ij}|$$

	n=1	n=2	n=3	n=4	n=5	Average
MAD Score	0.78695	0.7972	0.785	0.7803	0.78695	0.79

PART A

In this part, we will go ahead and develop a recommendation system that is more intelligent than the previous one.

IDEA:

Any user is likely to give ratings similar to ratings given by similar user.
So, Instead of taking the mean of ratings over all the users in the training set, we will take the average ratings of users that are more closer to our user.

IMPLEMENTATION

Lets say we need to find the rating by user 1 for Item 10. We have below ratings table of all users:

Movie ID				
User Id	10	20	30	40
1	??	2	3	2
2	0	1	1	1
3	4	2	4	2
4	2	1	5	1

1. We will firstly, shortlist just the user who rated the Movie ID 10.
2. Now, we will find the distance between every user using the movie ratings as features.
3. We will find closest k users that are similar to our user 1.
4. Finally, we will take the average of the ratings given to movie id 10 by k-closest users as the predicted rating for our user 1.
5. Therefore in the above sample table, user ID 2 will be dropped since he didn't rate the movie. Then we find the distance between other users and find the average.

   Following distance metrics were used:
   1. Euclidean
   2. Manhattan
   3. Cosine

Let us look at the results obtained after employing the above approach:

	k = 10					
	n=1	n=2	n=3	n=4	n=5	Avg
Euclidean	0.785	0.78265	0.76875	0.77785	0.7746	0.77777
Manhattan	0.776	0.77185	0.76065	0.7673	0.76655	0.768
Cosine	0.788459	0.797315	0.787745	0.780603	0.797743	0.79037

	k = 50					
	n=1	n=2	n=3	n=4	n=5	Avg
Euclidean	0.784	0.7754	0.76905	0.76915	0.7733	0.774
Manhattan	0.7801	0.7721	0.7653	0.7652	0.76965	0.77
Cosine	0.779889	0.778317	0.779317	0.773604	0.785602	0.77934

	k = 100					
	n=1	n=2	n=3	n=4	n=5	Avg
Euclidean	0.79055	0.7795	0.772	0.77385	0.7771	0.7786
Manhattan	0.78905	0.77845	0.7704	0.773	0.77615	0.77741
Cosine	0.785031	0.782602	0.783602	0.778317	0.791744	0.78425

OBSERVATIONS:

1. The Manhattan distance gave the lowest MAD score. The reason behind this could be that the other functions might highly penalize the difference between distances of two users. This might affect the ratings. We basically need a rough estimation of close users and do not want to highly penalize the differences.
2. Nextly since the dataset is not large enough, the k = 10 gives better results than other k values.
3. We observe a reduction of roughly 5-6% in our MAD score by using this approach instead of the naïve system that we saw before.

PART B

We will devise a recommendation system that is more intelligent than the one we saw in the previous part A.

IDEA:

1. In the previous part, we just considered the ratings given by the close users for a movie. We didn't consider the individual attributes of an user as a measure of closeness.
2. Now apart from ratings as features for the distance function, we will also use his/her age, gender, occupation as features.
3. Also, we will consider the average rating given by the user in question for movies in the similar genre that he had rated previously.
   This means let us say user John has watched batman, superman and spiderman. Now we need to predict the rating he might give for X-man. We know that X-man falls into the SCI-FI genre. So, we will take the average rating given by user john for Sci fi genre movies.
4. Final Formula:
   k- close users = Min k users based on features(ratings for movies, age, gender, occupation)
   Final rating = 0.8 * avg_rating(k-close users) + 0.2 * avg_rating(genre by user john)

   We have allotted a weight of 0.8 and 0.2 to give more importance to what other user rated. This is because ratings significantly depend on the quality of movie as judged by people who have already seen it.

IMPLEMENTATION:

Data Preprocessing:

In order to incorporate the user attributes, I recoded the feature values as below:

1. Age:
   Value = 1: 7-15 years
   Value = 2: 16-25 years
   Value = 3: 26-31 years
   Value = 4: 32 and above years
2. Gender:
   Value = 1: Female
   Value = 2: Male
3. Occupation:

As occupation is a nominal field, I grouped them ordinally based on the professions with high skill to low.

Value =1 : educator,engineer,healthcare,lawyer,programmer,scientist,writer,doctor
Value =2 : artist,Administrator,executive,marketing,entertainment
Value =3 : librarian,salesman,technician
Value =4 : homemaker,none,other,retired

	user id	age	gender	occupation
0	1	2	2	technician
1	2	5	1	other
2	3	2	2	writer
3	4	2	2	technician
4	5	4	1	other

item id	1	2	3	4	5	6	7	8	9	10	...	1662	1663	1664	1666	1669	1672	1673	1677	1678	1679
user id																					
1	5.0	3.0	0.0	3.0	3.0	5.0	4.0	1.0	5.0	3.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	2.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	4.0	3.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Finally using the columns(1:1679, age,gender,occupation) from the above two tables as features, we will calculate the closest K-users based on their distances.
These K-users would help in determining the average ratings for any given movie for a user, say Rating-A
Later we will also find the rating given for the movie by our user for similar genres, say Rating-B

Now we will club both the ratings A and B to give the final rating for the movie.
Final rating = 0.8 * Rating-A + 0.2 * Rating-B

Lets look at the results obtained using this methodology.

	k = 10					
	n=1	n=2	n=3	n=4	n=5	Avg
Manhattan	0.769888	0.755678	0.744388	0.7496608	0.7511504	0.7448
Euclidean	0.777587	0.770792	0.751832	0.757157408	0.7586619	0.752248
Cosine	0.785286	0.770792	0.751832	0.757157408	0.77368491	0.759696

	k = 50					
	n=1	n=2	n=3	n=4	n=5	Avg

Manhattan	0.775793	0.756736	0.749112	0.751111	0.756168	0.757736
Euclidean	0.78355	0.771871	0.756603	0.766133	0.771291	0.772891
Cosine	0.78355	0.764304	0.764094	0.758622	0.771291	0.765313

	k = 100					
	n=1	n=2	n=3	n=4	n=5	Avg
Manhattan	0.780007	0.762303	0.754365	0.757677	0.76146	0.763126
Euclidean	0.787807	0.769926	0.761908	0.772831	0.776689	0.773832
Cosine	0.787807	0.777549	0.761908	0.772831	0.776689	0.770757

OBSERVATION:
We obtain an improved accuracy of ~2-3% by using this method.

RATIONAL BEHIND USING THESE FUNCTIONS:

1. **Age**: Users of similar age group have similar preferences. For example, a kids of age 7-12 years might give higher ratings to Sci-Fi, adventure genres of movies. Whereas, a person of age group might be more interested in romantic and drama genres.
2. **Genders**: It can be a case where specific movie are liked by a same gender. For instance, romantic movies might be more liked by females than males. Whereas males might give higher rating to thriller movies.
3. **Occupation**: The occupation of a person might change his preferences and taste for movies. Say if a person is a programmer, he might watch and give higher ratings to Sci-fi technology-oriented movies like matrix, source code etc than other.

PART C

In this part we will use a bigger data set for our algorithm devised in part B. The data provided with 10 Million records need preprocessing. The smaller 100k data set was cleaner than this.

Firstly, we performed cross validation for breaking the data into 80-20% train & test split.

Distance Function	MAD Score
Euclidean	0.814
Manhattan	0.792
Cosine	0.7985

The Naïve method gave a MAD score of **0.91.** However, our algorithm gives a result of ~0.80. Manhattan and cosine gave a comparable performance.

In conclusion, the MAD score on our 10M dataset provides the evidence that the collaborative filtering technique makes better predictions than simple averaging. This helps in creation of better recommendation systems.

PART D

IMPROVISATIONS:

Following are the improvisations that could further improve the performance of our system.

1. We did not take the zip codes into account. It might be that the users of a particular area might like a movie more than users of other area.
2. We can play around with other distance functions like Pearson's coefficient, Jaccards distance, Minkowski distance.
3. We can also take into consideration how the movies of a particular genre are on an average rated by the users. It might be that adventure movies are rated more by users than the movies in Drama genre.