How Many Stars Will I Give To That Food Item?
(Extended To Food Recommendation System)

By Harshil Shah & Naman Shah

December 2, 2015

Contents

1 Problem Statement

Amazon is the biggest e-commerce website available in the world of Internet. It maintains huge dataset of user's buying in all section - movie, music, fine food items etcetera. Giving prediction regarding what user may think about the specific item (or in other words - how many starts is the user going to give to the item) is challenging thing. The system, we have developed, gives prediction of the ratings user will give to the specific item. As far as this project is concerned, we have focused on fine food items dataset only. We are having big dataset containing more than 550,000 fine food items' rating given by more than 250,000 users. Dataset is taken from SNAP(Stanford Network Analysis Project : See Link Here).

Each rating set of the food item, in the dataset, contains 8 different attributes. Out of all available attributes only 4 attributes - food item's ID (Amazon Standard Identification Number), user's ID, ratings given by user & usefulness of the rating - are important. Every feature is self descriptive except usefulness of rating. Usefulness of rating implies the fraction of users who found the review helpful. Our developed system is Big Data problem as it required scanning and analysis of whole dataset to give prediction of rating of user for specific food item. Furthermore, as we achieved good accuracy in prediction (nearly 80%), we extended our software to recommendation model too. System can be very useful for e-commerce fine food items dealing companies for suggesting items to users on per user's past buying list basis. Using same approach, system can be used for recommendation of any product as system is not product centric.

2 Strategy To Solution (Back-End)

Basically, for any recommendation system, there are two approaches we can follow.[1]

1. Content-based filtering - This method emphasizes on items. It tries to filter items which are similar to what user liked in the past.

2. Collaborative filtering - Unlike content-based filtering, this method emphasizes on users' behaviours, activities or preferences and tries to predict what user will like based on their similarity with other users. Key advantage of this approach is, it relies on users' behaviour rather than items which helps in accurate recommendation and prediction.

For this project, we have followed collaborative filtering to make system independent from product centric behaviour. System provides below functionality.

2.1 Generation Of User-Product Ratings Dataset

System combines all products rated by a user. Generated dataset is used to fulfill upcoming functionality.

2.2 Generation Of Average Rating Of Products

$Helpfulness$ of review was used to calculate average rating of product. System follows below formula to compute ratings.

$$\text{Avg Rating}_p = \frac{\sum_1^n (rating * helpfulness)}{n} \tag{1}$$

Where,
Avg Rating$_p$ = average rating for product p
n = user count who have rated product p

Here helpfulness will work as weight of the rating.

2.3 Generation Of User-User Similarity Matrix

Collaborative filtering approach requires some metric for user similarity. Here we are implementing user similarity matrix. There are many methods for finding distance between two users like euclidean distance, cosine distance, etcetere. System is measuring distance between users base on single property, ratings given to product. For single property, cosine distance gives more meaningful results than euclidean distance. System measures cosine distance in following manner,

$$dist_p(u1, u2) = cos(\text{absolute difference in ratings to product p given by users u1 and u2} * 22.5) \tag{2}$$

Where,
$dist_p(u1, u2)$: distance between user $u1$ and $u2$ for product p

We are multiplying difference of ratings with 22.5 as we want to map this difference in range of 0($cos 90^\circ$) to 1($cos 0^\circ$). Maximum difference of ratings is 4, so 90/4 = 22.5

To construct user similarity matrix, we are measuring cosine distance according to equation[2] for each possible pair of users for common products rated and then summing up all distance measures for each pair of users and that will be similarity index for that pair of users. For example, user $u1$ and user $u2$ have rated product p, 3 and 4 respectively. According to distance measure method used, cosine distance will be $cos(1 * 22.5)^\circ$ = 0.9238 for product p. We are adding all distances for common products between user $u1$ and user $u2$ and that will be similarity index between those users. System will provide top most 25 similar users only associated with each user.

2.4 User-Product Rating Prediction Method

To predict what specific user will rate to specific product, System will use user-user similarity matrix and user-product ratings dataset. System will find out which users from similarity matrix, for associated user, have rated for that product. We will consider two things, similarity index of that users and rating given by that user and apply following formula. Here similarity index will work as weight of rating.

$$\text{Predicted Rating} = \frac{\sum_{i=1}^{n}(rating_i * \text{similarity index}_i)}{\sum_{i=1}^{n}(\text{similarity index}_i)} \tag{3}$$

Where,
n = users count from similarity matrix who have rated for specific product
rating_i = rating given by user *i*
$\text{similarity index}_i$ = similarity index of user *i*

2.5 Tasks

2.5.1 Prediction

System will be trained on training dataset and will compute all the functionality mentioned in above sections. Prediction of rating for combination of User-Product from testing dataset will be computed according to methodology mentioned in section 2.4. Some users are suffering from cold start problem. For cold start users, prediction regarding User-Product combination is not possible using collaborative filtering. For such combinations, prediction of product will be simply taken as average rating given to product (output regarding is mentioned in section 2.2). Result generated by prediction functionality will further be used to test accuracy of system.

2.5.2 Recommendation

As accuracy of system is very high in prediction, we have extended our software to recommendation system. In this task, system will first find candidate product set for each user. Candidate product set comprise by all products for which any of the similar users have rated but given user has not. Then system will predict rating for each product in candidate set by applying prediction method discussed in section 2.4. Intermediate output dataset of User-Product predicted rating dataset will be used further to support front-end of system. System will then produced top most 10 items from predicted dataset to recommend to associated users. Dataset generated - User-Recommended Products - will also be further used in front-end of software.

3 Functionality Provided To Users (Front-End)

According to functionality provided by our back-end infrastructure of our software (described in section 2.5.2), two basic data-dumps are generated - (1) User-Product Predicted Ratings & (2) User-Recommended Products Dataset. A simple front-end structure was developed for convenience of user. Front-end does not directly communicate with back-end structure, rather it will use only two datasets to full-fill user's requirement. Back-end structure will be run periodically to keep both dataset up-to-date.

As system was giving very higher accuracy in prediction of ratings user will give to any food item, we have extended software to food recommendation system. So front-end will provide two powerful functionality to client - (1) Recommend Me Good Products & (2) Predict Rate For Me.

Both functions will take user's ID as input. Recommendation system will directly recommend the products for which user may give higher ratings (threshold = 4.0). For simplicity 10 best possible items for users will be recommended. Dataset, recommendation functionality will use is User-Recommended Products Dataset. Now for prediction functionality, system will further ask for the item ID for which user wants to know how many stars will he/she actually give. System will use User-Product Predicted Ratings Dataset to support this functionality. Both dataset will be dumped from back-end infrastructure periodically. Datasets will be cached in system's RAM (as combined memory of both dataset is nearly 750MB only) to speed up look-up process. Front-end may look like as below.

```
System Is Contacting BackEnd Structre
Downloading Files From Cluster
Read Files
Files Have Been Cached

   Enter UserName(ID):
   A1PUT35XF7KVD5

   1. Recommend Me
   2. Predict Rate For Me

   Selection: 1
   Recommended Prodcuts For A1PUT35XF7KVD5

   List: A1PUT35XF7KVD5%%
   B0025UC2DQ,B001GINOQC,B001P3NU4O,B001P3NU4E,B003SBU2VA,B000W7WQX0,B000H228UQ,
   B001GINOP8,B00024G8WI,B0000T15J6,

   Enter UserName(ID):
   ARQU1PP9XRO4K

   1. Recommend Me
   2. Predict Rate For Me

   Selection: 2

   Enter Product ID:
   B004U43ZO0

   Rating: ARQU1PP9XRO4K%#@B004U43ZO0%%
   4.0

   Enter UserName(ID):
```

4 Testing The System

System testing will be done in traditional way only. System will partition whole dataset randomly into 80:20 ratio where 80% will be training dataset and 20% will be testing dataset. User-User Similarity Matrix, User-Product Ratings and Product-Average Rating will be generated based on training dataset. Functionality to generate such matrices are mentioned in sections - 2.1, 2.2 & 2.3. Now according to methodology mentioned in section 2.5.1, prediction of rating for each combination of user-product from testing dataset will be generated. As testing dataset's actual rating will already be known beforehand (supervised learning), system can be tested by measuring average of absolute deviation of predicted outcome from actual outcome (MAD - Mean Absolute Deviation). Furthermore, accuracy of system is measured according to percentage of variation in result.

$$Accuracy = 100 - \frac{\sum_{i=1}^{N} \frac{\text{Absolute Deviation}_i * 100}{\text{Maximum Possible Deviation}}}{N}$$

Where:
N = total number of predicted ratings
Maximum Possible Deviation = 4
Absolute Deviation$_i$ = For rating i, absolute deviation(predicted outcome - actual outcome)

5 Results & Evaluation

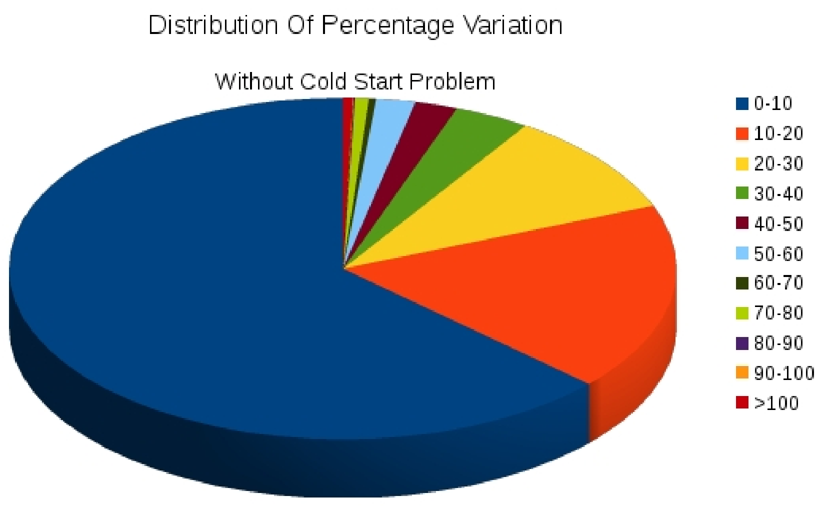
Evaluation was done as mentioned in above section 4. Evaluation has been done on both type of users - suffering from cold start problem & normal users. Combined results were also derived.

5.1 For Normal Users

Accuracy regarding normal users was nearly 89% (to be precise 88.78017%). Furthermore, average of absolute deviation (MAD) was only 0.4488. Below table 1 represents that nearly 63% of records gives deviation less than 10%. Nearly 90% of records gives deviation less than 30%. Only 0.5% of records gives worst performance(¿=100%). Pie chart following below table depicts graphical representation of distribution of deviations.

Range Of Variation	Total Count	Percentage Of Count Total Count
0-10	28867	63.1842756145
10-20%	8129	17.792807582
20-30%	4487	9.8211745135
30-40%	1668	3.6509291483
40-50%	939	2.0552892508
50-60%	870	1.9042616061
60-70%	152	0.3326985795
70-80%	312	0.6829076105
80-90%	29	0.0634753869
90-100%	19	0.0415873224
>=100	215	0.4705933854
Total	**45687**	-

Table 1: Analysis Results - For Non Cold Start Users

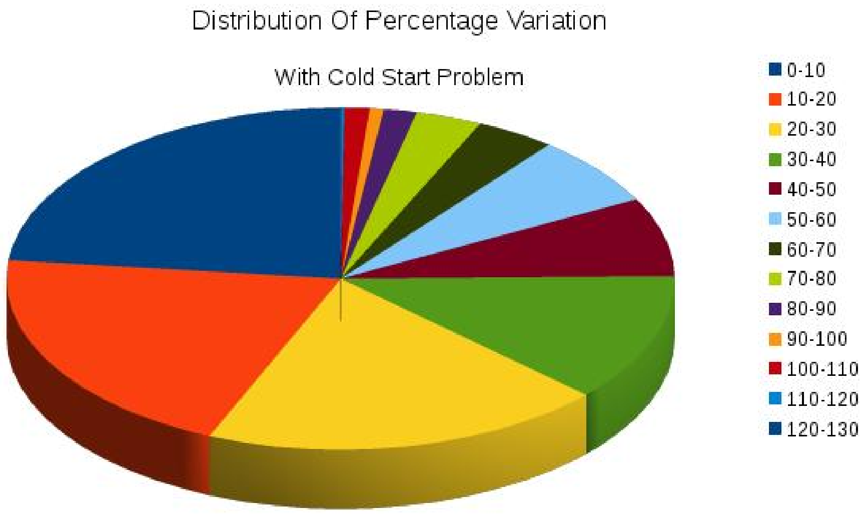


5.2 For Users Suffering From Cold Start Problem

Accuracy regarding such users was nearly 72% (to be precise 72.0463%). Furthermore, average of absolute deviation (MAD) was 1.1181428. Below table 2 represents that only 23% of records gives deviation less than 10%. 80% of records gives deviation less than 50%. 1.5% of records gives worst performance(¿=100%). Pie chart following below table depicts graphical representation of distribution of deviations.

Range Of Variation	Total Count	Percentage Of Count Total Count
0-10	14359	*23.2881377923*
10-20%	12501	*20.274741315*
20-30%	12049	*19.5416653151*
30-40%	7473	*12.1200817412*
40-50%	4593	7.4491550164
50-60%	4077	6.6122806448
60-70%	2362	3.8308086542
70-80%	1979	3.2096402738
80-90%	977	1.5845470174
90-100%	421	0.6827986636
$>$**100**	867	1.4061435661
Total	**61648**	

Table 2: Analysis Results - For Cold Start Users

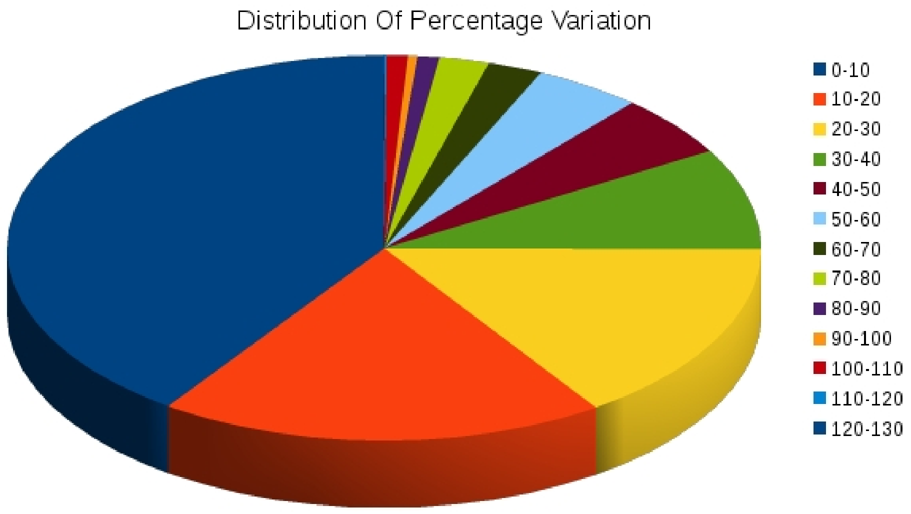


5.3 Combined Results

Accuracy for combined dataset was nearly 80% (to be precise 79.16827%). Furthermore, average of absolute deviation (MAD) was 0.83325833. Below table 3 represents that 40% of records gives deviation less than 10%. 90% of records gives deviation less than 50%. Only 1% of records gives worst performance(¿=100%). Pie chart following below table depicts graphical representation of distribution of deviations.

Range Of Variation	Total Count	Percentage Of Count To Total Count
0-10	43123	*40.2856795867*
10-20%	20556	*19.2034976598*
20-30%	16494	*15.4087609652*
30-40%	9106	*8.5068617285*
40-50%	5523	5.1596087554
50-60%	4936	4.6112310006
60-70%	2505	2.3401810487
70-80%	2286	2.1355903702
80-90%	1002	0.9360724195
90-100%	440	0.411049765
>100	1072	1.0014667003
Total	**107043**	-

Table 3: Analysis Results



Before developing system, accuracy of system was predicted to be only 40-50%. But results totally suggest good system implementation. Furthermore, for normal users, accuracy is very high compare to users facing cold start problem. But number of cold start problem users are bit higher than number of normal users, accuracy was reduced. Overall performance, even though, was good.

6 Contribution

Developer	**Harshil Shah**	**Naman Shah**
Job Task 1	User-User Similarity	User-Product, Product - Average Rating Dataset
Job Task 2	Recommendation System Without Using U-U Similarity Index	Recommendation System (Old) Without Using U-U Similarity Index
Job Task 3	Prediction Of Ratings	Evaluation Of Recommendation System (Old), Construction Of Candidate Products Associated With Each Users
Job Task 4	Recommendation Using Prediction	Evaluation Of Recommend Model

Table 4: Contribution Of Each Developer

References

[1] Recommendation System - Wikipedia
`en.wikipedia.org/wiki/Recommender_system`