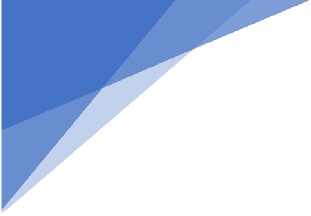
# Product Recommender System Using

User Based Cosine Similarity and ALS

Omkar Bapat, Aabir Datta, Lohitaksh Yogi

2

**Introduction**

Recommender system is a very important tool in marketing. As Ecommerce is becoming more and more popular, every ecommerce website has/wants a recommendation system which targets specific users. Website like Amazon, eBay etc. use the previously seen/carted/purchased products by the user to recommend them similar products. It is very interesting to see how accurate these recommendations are and the underlying motivation of this project was to learn how these systems work and to implement one of our own. There are two major ways the collaborative filtering approach can be performed for filtering products- using Alternating Least Square algorithm or using Cosine similarity approach. In our project we have implemented both these approaches.

## Objectives

1. Implementation of ALS and Cosine Similarity:

The recommender system we have developed can run on both the ALS approach and the cosine similarity approach and we have developed a study of performance and accuracy for both these systems.

1. Implementation of Code without using ML Library.

We have implemented both the algorithms without using the MLlib library in Spark.

1. Increasing efficiency and User Interface

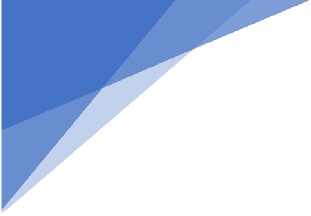
We aim to increase the efficiency of the way our recommender system works and to make an interactive Graphical User Interface where users can input new ratings to the database and hence enable interactive environment.

##### Challenges

* + Calculating the RMSE value was a challenge.
  + Cleaning the Dataset
  + Running the Prediction algorithm over the dataset
  + Actual implementation of Cosine similarity and calculation of scores and ALS algorithm

##### Implementation Details

We have written our code in such a way that it finds the optimal number of neighbors for the neighborhood matrix, which can give us a good RMSE value. We have also used a combination of Cartesian matrix and filtering to get user pairs for Cosine Similarity Matrix

3

Project Description:

###### Framework Used:

We have used SPARK for implementation purposes.

###### Data Profile:

We had emailed **Julian McAuley** for the Amazon product dataset (julian.mcauley@gmail.com).

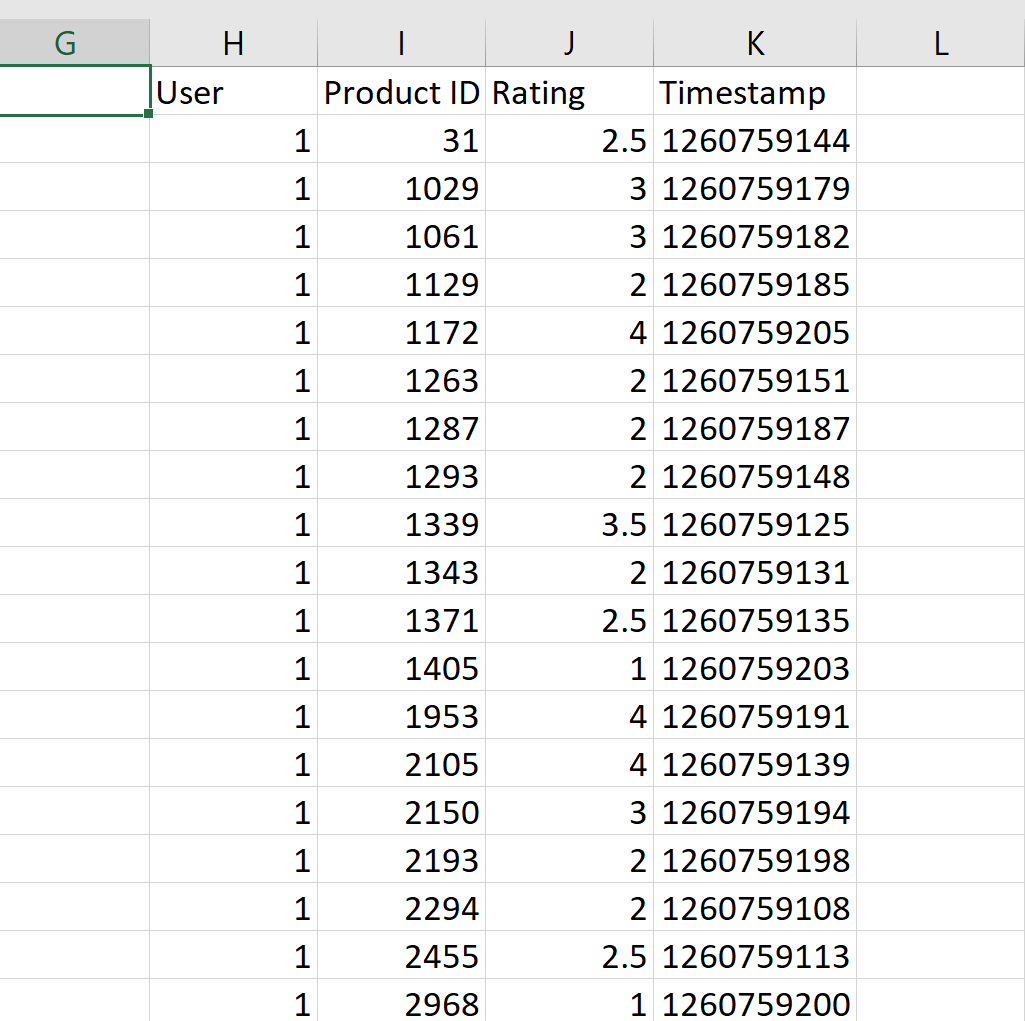
The Product dataset has the following files:

1. ProductId
2. Product

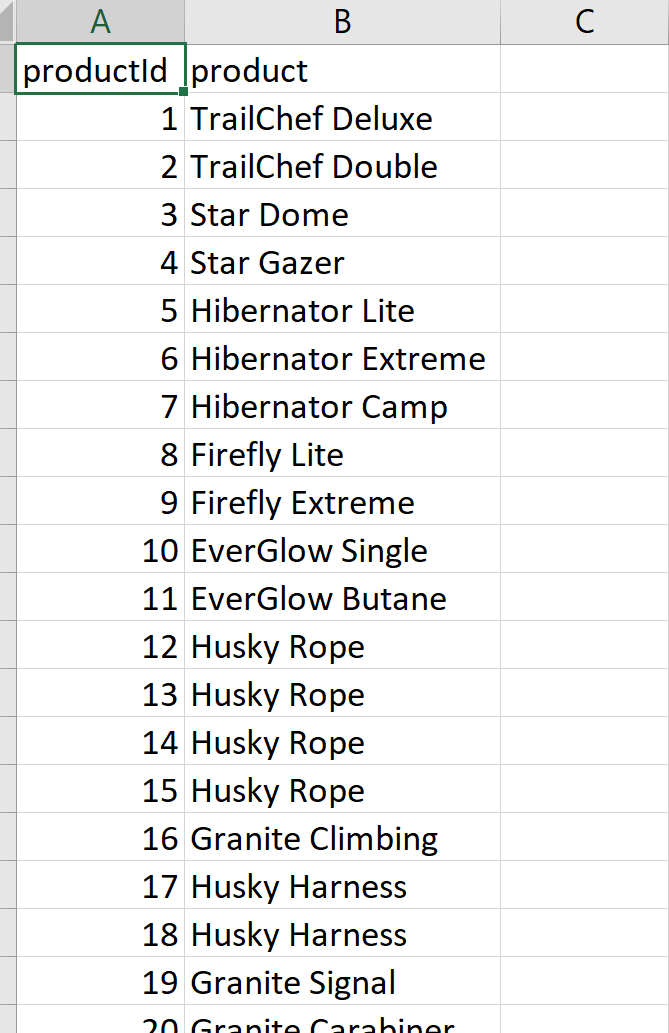
We have used the above two files for making the recommender system work.

We have executed our implementation on Product file which includes around 100,000 ratings applied to 9,000 products by 671 users.

Rating:

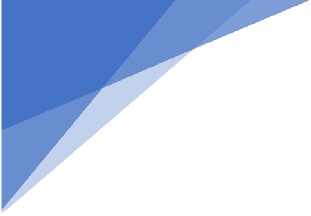


Products:



In our project we implemented collaborative filtering algorithms which is one of the most commonly used algorithm in the industry. We have two types of collaborating filtering algorithms.

4



**User-User Collaborative filtering:** Here we find look alike customer to every customer and offer products which first customer’s look alike has chosen in past. This is done by similarity scores between customers or users.

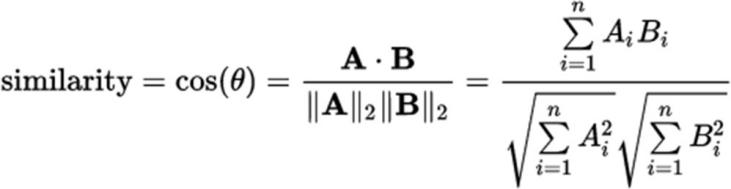
**Item-Item Collaborative filtering:** This is like the previous algorithm, but instead of finding customer look alike, we try finding item look alike.

###### User Based Collaborating Recommenders System using Cosine Similarity:

In the algorithm, the similarities between different users in the dataset are calculated by using one similarity measures, such as Cosine similarity.

As mentioned above there are many different mathematical formulations that can be used to calculate the similarity between two users. In our approach we used cosine similarity for measuring similarity.

**Cosine Similarity Explanation:**

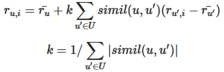
Initially we took rating dataset and calculated the cosine similarity between each of the users.

Further we calculated top k similar users

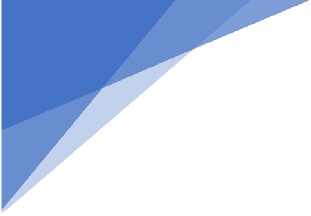
This top K similar users is our neighborhood matrix

The UserID for which we have to predict is given as an argument, and based on his K neighbors we recommend top 5 products

We use aggregate function to calculate the ratings for our User for products which he has not rated. We are also taking into consideration the fact that if other users are stingy/liberal in our calculation.



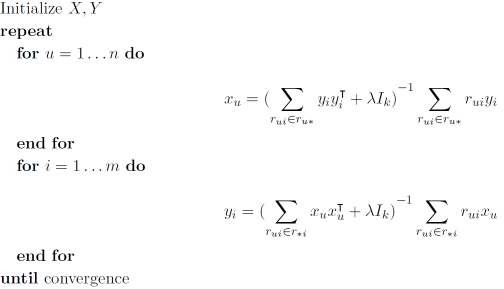
The top 5 predictions are outputted.

5

##### Alternating Least Squares:

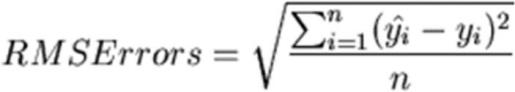
The Alternating Least Squares algorithm goal is to estimate the complete ratings matrix by the formulae . Our aim to minimize the least squares error of the observed ratings.

The ALS algorithm approach is to Fix Y and optimize X and then Fix X and optimize Y until it converges.



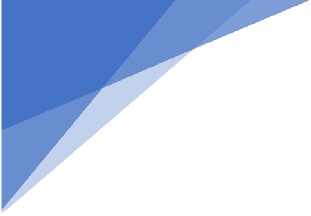
**ALS Implementation**

For ALS implementation we created an initial rating matrix by assigning 0.0 rating for unrated products. In rating matrix rows represents the user id and column represents the product id. We randomly initialize the user matrix (u\*k) and product matrix (m\*k) with factor k. Now we Iterate for evaluating user matrix by fixing product matrix and then evaluating product matrix by fixing the user matrix until it converges. We calculate the RMSE for each iteration to check for convergence and evaluate the Model. Also, we have calculated the average value for RMSE after convergence.



The final prediction rating output is evaluated by the formulae. 

We are predicting the product id for each user id and for the best predicted rating value.

6

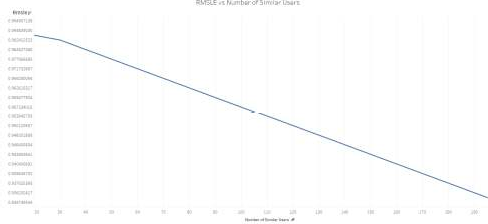
##### Results and Evaluation of Models

###### Optimizing the Number of Neighbors for Cosine Similarity:

We can select the number of neighbors for the Neighborhood matrix, more neighbors give us less error as from the below graph. We run a loop starting at 10 to 200 with a step size of 10 to see how many neighbors gave us the lowest RMSE.

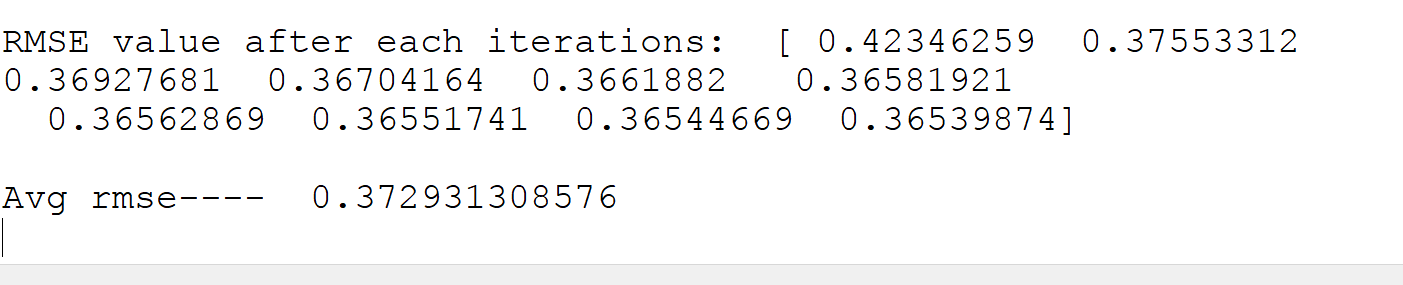
Higher number of neighbors reduces the RMSE but we also have to consider the computational cost.

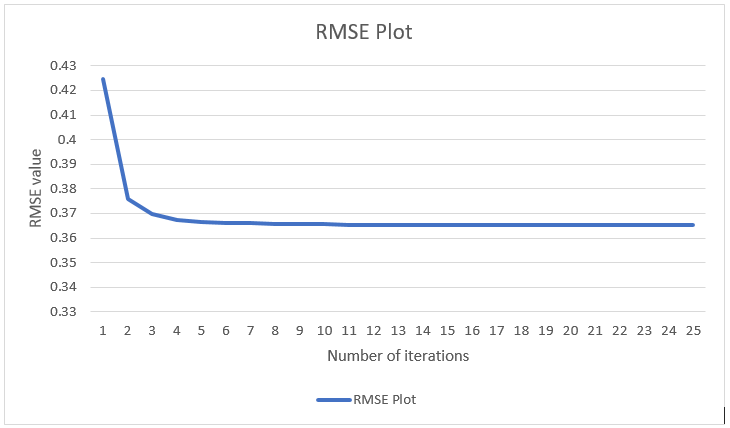
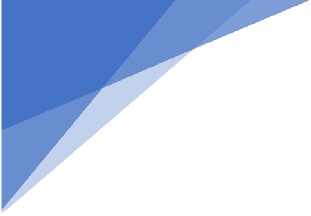
RMSE value of Cosine similarity with 200 Neighbors: 0.9347



###### Optimizing the Prediction rating for ALS Implementation:

Below is the output value for RMSE after each iteration. We have plotted the graph for RMSE value after each iteration for 25 iteration. Also we have calculated the average rmse value after 25 iteration.

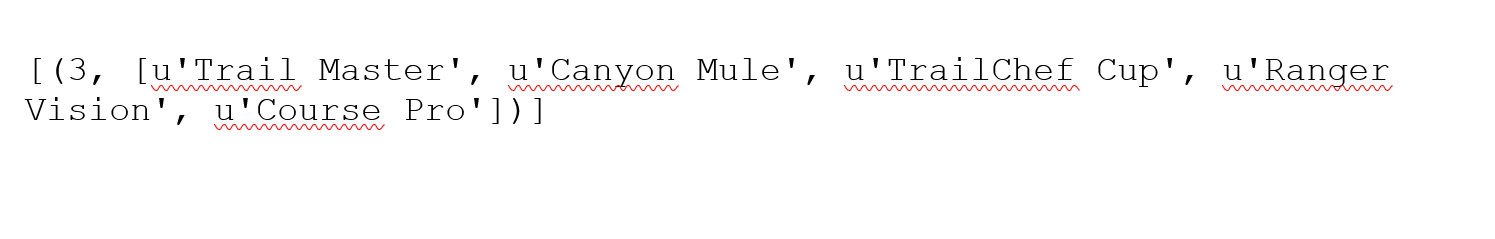




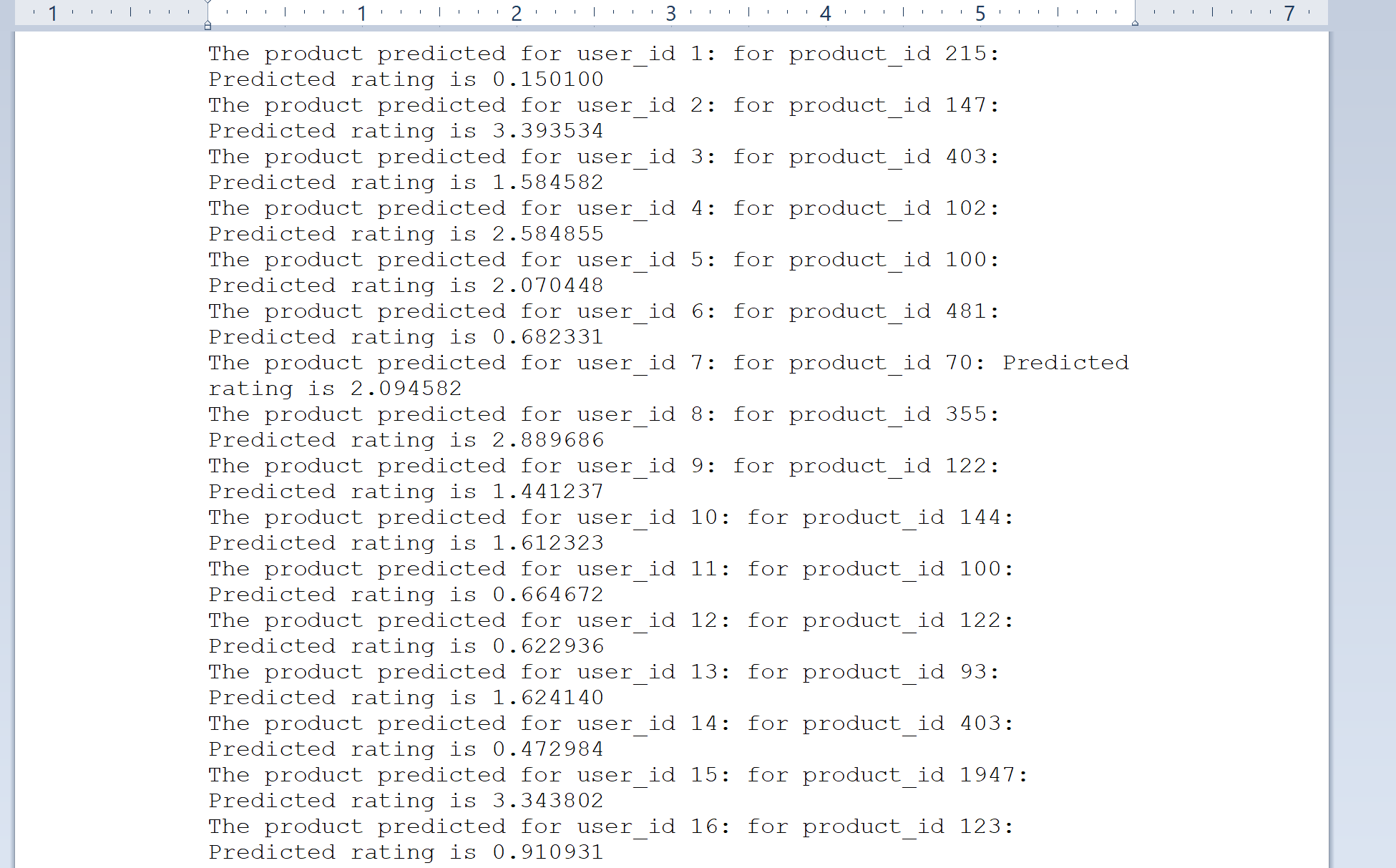
7

##### Output for Cosine Similarity:

Products recommended for User ID 3

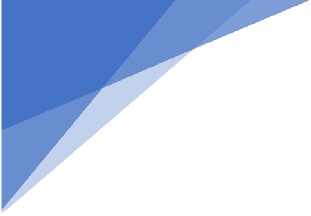


##### Output for ALS:

****

**Conclusion and future scope**

We have successfully implemented two things mentioned in our objectives: a) Cosine similarity b) ALS algorithm. We are planning to implement the functionality of adding a new user to the dataset. Also, as said earlier we have put the user code on GitHub. On contrary, we would like to implement a user interface where the user can give ratings to different products and then get a new recommendation based on his new ratings.

8

**References**

<https://www.packtpub.com/books/content/building-recommendation-engine-spark>

<http://www.infofarm.be/articles/alternating-least-squares-algorithm-recommenderlab>

<https://datasciencemadesimpler.wordpress.com/tag/alternating-least-squares/>

**Individual Tasks**

Omkar Bapat : Project Report, Cosine Similarity Algorithm

Aabir Datta: Data Prepation and exploration, Cosine Similarity Algorithm

Lohitaksh Yogi: Data Preparation for ALS, ALS Algorithm implementation, Evaluation of model