Movie Recommender System
Using
User Based Cosine Similarity and ALS

Satish Kumar

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 Netflix, YouTube etc. use the previously seen/liked videos by the user to recommend them similar videos. 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 - using Alternating Least Square algorithm or using Cosine similarity approach. In our project we have implemented both these approaches.

Goals

A) We will definitely accomplish

A fully functioning recommender system which can run on both the ALS approach and the cosine similarity approach and a study of performance and accuracy of both these systems.
We have achieved this, the recommender system works.

B) We will likely accomplish.

Implementing both these systems without using the MLlib library in Spark.
We have implemented the cosine similarity and the ALS algorithm without using the MLlib library.

C) We would like to ideally accomplish

Implementing a method where both the above given methodology are used to increase efficiency and to make an interactive User Interface where users can input new movies to the database and have an interactive environment.
We could not achieve this goal due to time constraints.

Challenges

- The biggest challenge was understanding the Cosine filtering Algorithm and implement it on spark RDDs.
- Another challenge was the understanding the working of Collaborative filtering and understanding the Underlying mathematics.

Cool Enhancements

- We have written a code to find the optimal number of neighbors for the neighborhood matrix, which will give us a good RMSE value.
- We used a combination of cartesian matrix and filtering to get user pairs for Cosine similarity Matrix.

Project Description:

Framework Used:

All the algorithms are implemented on SPARK

Data Profile:

The Movielens dataset has the following files:

1) Ratings
2) Movie
3) Links
4) Genome Tags
5) Genome Score

Although there five files in our dataset, but for creating our recommenders system we used rating and movies files.
We have executed our implementation on Movielens file which includes around 100,000 ratings and 1,300 tag applications applied to 9,000 movies by 700 users.

Rating:

	userId	movieId	rating	timestamp
0	1	122	2.0	945544824
1	1	172	1.0	945544871
2	1	1221	5.0	945544788
3	1	1441	4.0	945544871
4	1	1609	3.0	945544824

Movies:

	movieId	title	genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

In our project we implemented collaborative filtering algorithms as this is one of the most commonly used algorithm in the industry. There are two types of collaborating filtering algorithms.

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: It is quite similar to 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 of a number of similarity measures, such as Cosine similarity, Jaccard Similarity.
As mentioned above there are a number of 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:

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

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|_2 \|\mathbf{B}\|_2} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \sqrt{\sum_{i=1}^{n} B_i^2}}$$

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

$$r_{u,i} = \bar{r_u} + k \sum_{u' \in U} simil(u, u')(r_{u',i} - \bar{r_{u'}})$$

$$k = 1 / \sum_{u' \in U} |simil(u, u')|$$

The top 5 predictions are outputted.

Alternating Least Squares:

The Alternating Least Squares algorithm goal is to estimate the complete ratings matrix by the formulae $R \approx X^T Y$. In particular, our aim to minimize the least squares error of the observed ratings.

$$\min_{X,Y} \sum_{r_{ui}\, observed} (r_{ui} - x_u^{\intercal} y_i)^2 + \lambda(\sum_u \|x_u\|^2 + \sum_i \|y_i\|^2)$$

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

Initialize X, Y
repeat
  for $u = 1 \ldots n$ **do**

$$x_u = (\sum_{r_{ui} \in r_{u*}} y_i y_i^{\intercal} + \lambda I_k)^{-1} \sum_{r_{ui} \in r_{u*}} r_{ui} y_i$$

  end for
  for $i = 1 \ldots m$ **do**

$$y_i = (\sum_{r_{ui} \in r_{*i}} x_u x_u^{\intercal} + \lambda I_k)^{-1} \sum_{r_{ui} \in r_{*i}} r_{ui} x_u$$

  end for
until convergence

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

$$RMSErrors = \sqrt{\frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{n}}$$

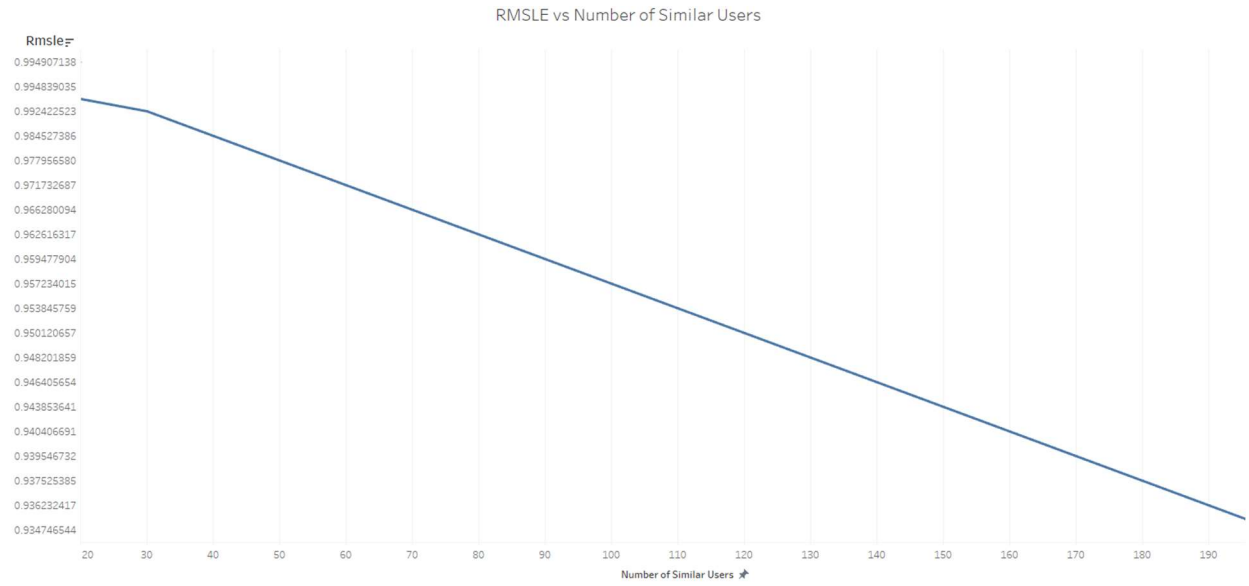
The final prediction rating output is evaluated by the formulae. $R \approx X^T Y$.
We are predicting the movie id for each user id and for the best predicted rating value.

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 seen 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

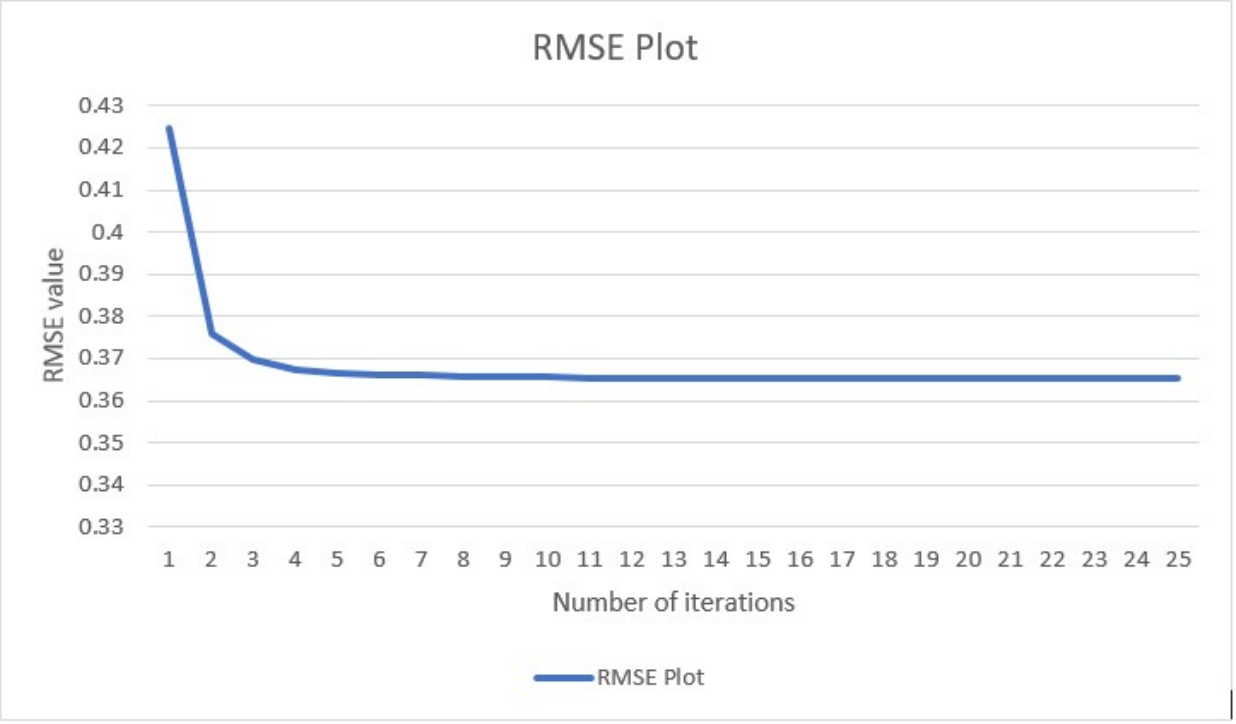


Optimizing the Prediction rating for ALS Implementation:

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.

```
RMSE value after each iterations:  [ 0.42239651  0.37627097  0.37003185  0.36810415  0.36721687  0.36667271
  0.36629295  0.36602049  0.36582558  0.3656856   0.36558337  0.36550702
  0.36544877  0.3654036   0.36536815  0.36534009  0.36531773  0.36529983
  0.36528541  0.36527374  0.36526422  0.36525642  0.36524997  0.3652446
  0.3652401 ]

Avg rmse Value:  0.368584027924
```



Output for Cosine Similarity:

Movies recommended for User ID 3

```
[(3, [u'Hands on a Hard Body (1996)', u'Matewan (1987)', u"Antonia's Line
(Antonia) (1995)", u'Amateur (1994)', u'Love & Human Remains (1993)'])]
```

Output for ALS:

```
The movie predicted for user_id 1: for movie_id 215: Predicted rating is 0.155814
The movie predicted for user_id 2: for movie_id 147: Predicted rating is 3.442854
The movie predicted for user_id 3: for movie_id 403: Predicted rating is 1.654296
The movie predicted for user_id 4: for movie_id 102: Predicted rating is 2.675025
The movie predicted for user_id 5: for movie_id 100: Predicted rating is 2.138578
```

Conclusion and future scope

We have implemented both Cosine similarity and ALS algorithm. We would like to add the functionality of adding a new user to the dataset. Also we would have liked to implement a cleaner user interface where the user can give ratings to different movies and then get a new recommendation based on his new ratings.

References:

1. https://www.youtube.com/watch?v=q97VFt56vRs&list=PLuKhJYywjDe96T2L0-zXFU5Up2jqXlWI9

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

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

4. https://blog.insightdatascience.com/explicit-matrix-factorization-als-sgd-and-all-that-jazz-b00e4d9b21ea

5. https://www.youtube.com/playlist?list=PLLssT5z_DsK-h9vYZkQkYNWcItqhlRJLN

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