International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 635-639 www.ijaem.net ISSN: 2395-5252

Music Recommender System Based on Collaborative Filtering

AnuPrabha P S, HarsithaN, Vaishnavi K, Dr.P.Velvadivu and Dr.M.Sujithra

---

Date of Submission: 01-11-2020 Date of Acceptance: 15-11-2020

Abstract: Recently, the building of recommender systems becomes a significant research area that attractive several scientists and researchers across the world. The recommender systems are used in a variety of areas including music, musics, books, news, search queries, and commercial products. Collaborative Filtering algorithm is one of the popular successful techniques of RS, which aims to find users closely similar to the active one in order to recommend items. Collaborative filtering (CF) with alternating least squares (ALS) algorithm is the most imperative techniques which are used for building a music recommendation engine. The ALS algorithm is one of the models of matrix factorization related CF which is con- sidered as the values in the item list of user matrix. As there is a need to perform analysis on the ALS algorithm by selecting different parameters which can even- tually help in building efficient music recommender engine. In this paper, we propose a music recommender system based on ALS using Apache Spark. This research focuses on the selection of parameters of ALS algorithms that can affect the performance of a building robust RS. From the results, a conclusion is drawn according to the selection of parameters of ALS algorithms which can affect the performance of building of a music recommender engine. The model evaluation is done using different metrics such as execution time, root mean squared error (RMSE) of rating prediction, and rank in which the best model was trained. Two best cases are chosen based on best parameters selection from experimental results which can lead to building good prediction rating for a music recommender.

Keywords: Recommender systems, Collaborative filtering, Alternating Least Squares , Apache Spark Big data , Million song dataset

I. INTRODUCTION:

Big data analytics become an important trend for organizations and enterprises that are interesting in providing innovative ideas for enhancing and increasing their business performance and decision-making. Recommender systems are a group of techniques that allow filtering through large samples and information space in order to give suggestion to users when needed. Currently, they are becoming highly popular and utilized in different areas such as musics, research articles, search queries, news, books, social tags, and music. Furthermore, there are other essential recommender systems basically applicable for specialist, collaborators, funny story, restaurant and hotels, dresses, monetary services, life insurance, passion associates which give online dating services and several other social media such as Twitter, LinkedIn, and Facebook.

The main focus of this work is collaborative filtering system. It is well known that collaborative filtering could be described as a procedure whereby automatic prediction (i.e., filtering) about the interests of a user is made by gathering taste or preferences information from many users. The unexpressed assumption of the collaborative filtering approach can be best explained, viz., supposing a person A has similar opinion with person B on a particular issue, the assumption is that person A will be more likely to have the same opinion as person B on a different issue X did the opinion on X of a person chosen randomly[1]. The main tools that we have used here is apache spark and apache hive where we have implemented recommender system using apache spark and using HQL analysis of our dataset is done.

The minimization of the error for the users/music pairs was chosen as the basis for the selection of the two matrices. The alternating least squares algorithm (ALS) which achieves this by randomly filling the user's matrix with values before optimizing the value of the music was used for this purpose. The value of the user's matrix is optimized with the music's matrix being kept constant (Fig. 1). Owing to a fixed set of user factors (i.e., values in the user's matrix), known ratings are employed to find the best values by

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 635-639 www.ijaem.net ISSN: 2395-5252

optimizing the music factors, written on top of the figure. The best user factor with the fixed music factors is sleeted. This paper, reports for the first time, a music recommendation system based on collaborative filtering using apache spark. The performance analysis and evaluation of proposed approach are performed on a Million song dataset.

Dataset:

The dataset chosen is million song dataset. The Million Song Dataset is a freely-available collection of audio features and metadata for a million contemporary popular music tracks. The dataset does not include any audio, only the derived features. The Million Song Dataset was created under a grant from the National Science Foundation. This dataset contain millions of information of users. The dataset contains two sets of data the song data and the user data. The song data consist of artist name, song id, title, release and year. The user data contains song id, user id and the listen counts.

Attributes	Description
Song_id	The unique id assigned to each song.
User_id	The unique id assigned to each user.
Count	The count of each song listened by each user.
Title	The title of the song.
Release	The album with which the song was released.
Artist_name	The singer of the song
Year	The year the song was released.

Data Preprocessing:

Initially the dataset is checked for null values and the null values are replaced by filled using imputation method. Since the dataset contains some categorical values it is converted to numeric values using stringindexer and pipeline modules. The dataset is divided into training and test data in the ratio 75:25 ratio. After the data is preprocessed data is ready for further analysis.

Analysis:

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy. Here hive is used for analyzing data which gives us the following information: Total unique artist, Most popular artist,Top 10 popular track, Song of the year.

Apache spark is a data processing framework that can quickly perform processing tasks on very large data sets, and can also distribute data processing tasks across multiple computers, either on its own or in tandem with other distributed computing tools. These two qualities are key to the worlds of big data and machine learning. Using spark sqlquering is done to know more about data before proceding with the implementations of recommendation system. Analyzing the data we get the following information like songs which are songs played the most and who are all the active users

Recommendation:

Recommender systems aim to predict users' interests and recommend product items that quite likely are interesting for them. They are among the most powerful machine learning systems that online retailers implement in order to drive sales.

Recommendations typically speed up searches and make it easier for users to access content they're interested in, and surprise them with offers they would have never searched for.

By using music recommender system, the music provider can predict and then offer the appropriate songs to their users based on the characteristics of the music that has been heard previously.

Collaborative flitering:

Collaborative Filtering, one the other hand, does not require any information about the items or the user themselves. It recommends the item based on user past experience and behavior. The key idea behind Collaborative Filtering is that similar users share similar interest, people with similar interest tends to like similar items. Hence those items are recommended to similar set of users. For example, if a person A has same opinion as a person B on an issue. Then A is more likely to have B's opinion on different issue. Thus recommendations are made using ALS matrix factorization method.

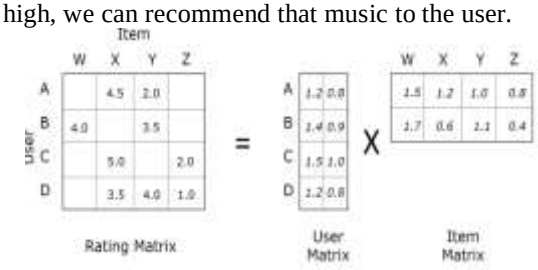
Item-based: measure the similarity between the items that target users rates and interacts with other items.

Matrix factorization methods

When a user gives feed back to a certain music they saw (say they can rate from one to five), this collection of feedback can be represented in a form of a matrix. Where each row represents each users, while each column represents different musics. Obviously the matrix will be sparse since not everyone is going to watch every musics, (we all have different taste when it comes to musics).

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 635-639 www.ijaem.net ISSN: 2395-5252

One strength of matrix factorization is the fact that it can incorporate implicit feedback, information that are not directly given but can be derived by analyzing user behavior. Using this strength we can estimate if a user is going to like a music that (he/she) never saw. And if that estimated rating is high, we can recommend that music to the user.



The above image does an excellent job of summarizing, the core idea behind matrix factorization. Let there be matrix A with dimensionality of (m,n) this matrix can be viewed as a dot product between two matrix with each matrices having dimensions of (m,k) and (k,n).

Just as a side note, the above concept is heavily related to Singular Value Decomposition (SVD). One downside of SVD is the fact that when the original matrix is sparse (incomplete) left and right singular vectors are undefined.

The concept of matrix factorization can be written mathematically to look something like below.

$$\hat{r}_{ui} = q_i^T p_u$$

Then we can create an objective function (that we want to minimize) with respect to q and p, which are (m,k) and (k,n) matrices.

$$\min_{q^*,p^*} \sum_{(u,i)\in\kappa} (r_{ui} - q_i^T p_u)^2 + \lambda(\| q_i \|^2 + \| p_u \|^2)$$

The term on the right is the regularization term, this is added since we do not want our decomposed matrix q and p to over-fit to the original matrix. Since our goal is to generalize the previous ratings in a way that predicts future, unknown ratings, we should not over-fit our model.

Learning Methods

One obvious method to find matrix q and p is the gradient descent method. Since we have the loss function defined, take the partial derivative respect to q and p to optimize those values.

$$q_i \leftarrow q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$$
$$p_u \leftarrow p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$$

By taking partial derivatives, the update rule would look something like above. But the error surface is not convex, we can also take the alternative approach in which we fix q and p alternatively while optimizing for another.

$$\min_{p^*,q^*,b^*} \sum_{(u,i)\in\kappa} (r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda$$
$$(\| p_u \|^2 + \| q_i \|^2 + b_u^2 + b_i^2)$$

$$\hat{r}_{ui}(t) = \mu + b_i(t) + b_u(t) + q_i^T p_u(t)$$

Therefore, standard information retrieval performance measures are frequently used to evaluate recommenderperformance.

Alternating Least Square (ALS)

Alternating Least Square (ALS) is also a matrix factorization algorithm and it runs itself in a parallel fashion. ALS is implemented in Apache Spark ML and built for a larger-scale collaborative filtering problems. ALS is doing a pretty good job at solving scalability and sparseness of the Ratings data, and it's simple and scales well to very large datasets.Most important hyper-parameter in Alternating Least Square (ALS):

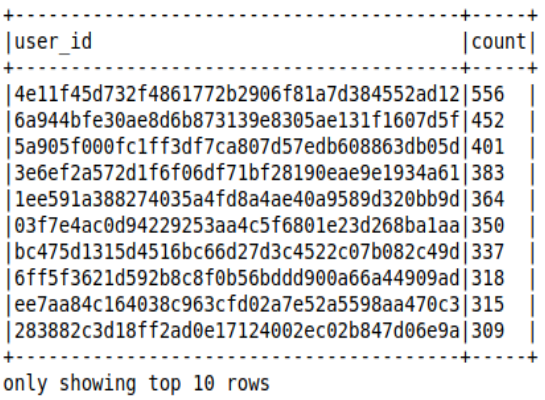
- maxIter: the maximum number of iterations to run (defaults to 10)
- rank: the number of latent factors in the model (defaults to 10)
- regParam: the regularization parameter in ALS (defaults to 1.0)

Steps to be followed to build a recommender system:create dataframe of all distinct songscreating another dataframe which contains already listened songs by active user.Joining both tables on left join. Selecting songs which active user is yet to listen. Adding new column of user_Id of active useer to remaining songs df. making recommendations using ALS recommender model and selecting only top 10 songs.

II. RESULTS AND DISCUSSION:

The fig(i) shows the most repeatedly herd songs from which we know what are all the popular songs based on the users listen counts. This can be used by the streaming apps as a tool to attract users by recommending the popular songs.

International Journal of Advances in Engineering and Management (IJAEM)
Volume 2, Issue 8, pp: 635-639 www.ijaem.net ISSN: 2395-5252

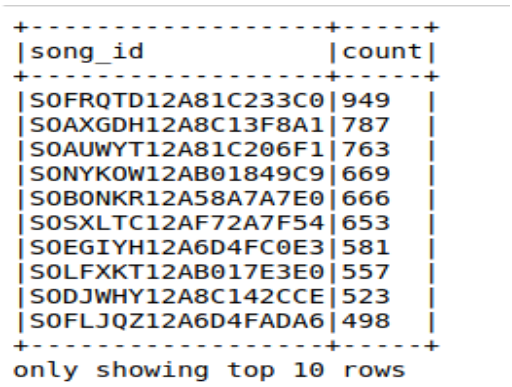


user_id	count
4e11f45d732f4861772b2906f81a7d384552ad12	556
6a944bfe30ae8d6b873139e8305ae131f1607d5f	452
5a905f000fc1ff3df7ca807d57edb608863db05d	401
3e6ef2a572d1f6f06df71bf28190eae9e1934a61	383
1ee591a388274035a4fd8a4ae40a9589d320bb9d	364
03f7e4ac0d94229253aa4c5f6801e23d268ba1aa	350
bc475d1315d4516bc66d27d3c4522c07b082c49d	337
6ff5f3621d592b8c8f0b56bddd900a66a44909ad	318
ee7aa84c164038c963cfd02a7e52a5598aa470c3	315
283882c3d18ff2ad0e17124002ec02b847d06e9a	309

only showing top 10 rows

fig(1)

The fig(2) gives us the information regarding who are all the active users (i.e) which user has listened more number of songs. This will be helpful in knowing the active user of the app which will be useful for the streaming apps to concentrate more on this type of users.

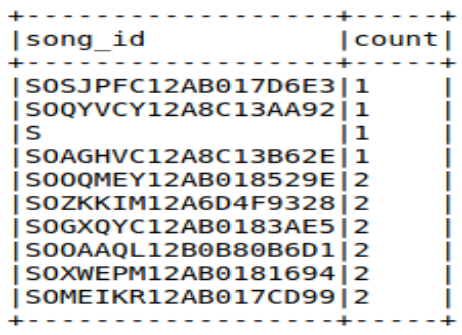


song_id	count
SOFRQTD12A81C233C0	949
SOAXGDH12A8C13F8A1	787
SOAUWYT12A81C206F1	763
SONYKOW12AB01849C9	669
SOBONKR12A58A7A7E0	666
SOSXLTC12AF72A7F54	653
SOEGIYH12A6D4FC0E3	581
SOLFXKT12AB017E3E0	557
SODJWHY12A8C142CCE	523
SOFLJQZ12A6D4FADA6	498

only showing top 10 rows

Fig(2)

Fig(3) displays the users who has listened to least number of songs from which we infer that these are inactive users.

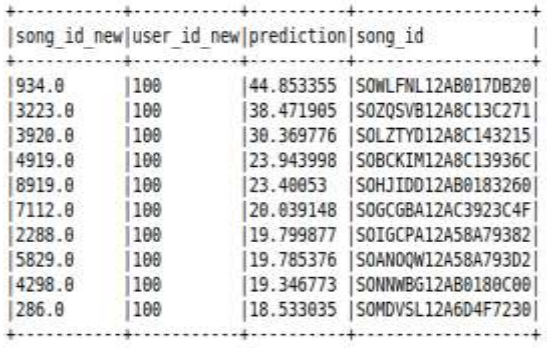


song_id	count
SOSJPFC12AB017D6E3	1
SOQYVCY12A8C13AA92	1
S	1
SOAGHVC12A8C13B62E	1
SOOQMEY12AB018529E	2
SOZKKIM12A6D4F9328	2
SOGXQYC12AB0183AE5	2
SOOAAQL12B0B80B6D1	2
SOXWEPM12AB0181694	2
SOMEIKR12AB017CD99	2

only showing top 10 rows

Fig(3)

Fig(4) is the final output which is obtained by implementing the als algorithm. This figure recommends ten songs to the user of id 100. The recommended songs are the one which are not listened by that user and this recommendation is done based on their interest. Similarly we recommend songs for other users.



song_id_new	user_id_new	prediction	song_id
934.0	100	44.853355	SOWLFNL12AB017DB20
3223.0	100	38.471905	SOZQSVB12A8C13C271
3920.0	100	30.369776	SOLZTYD12A8C143215
4919.0	100	23.943998	SOBCKIM12A8C13936C
8919.0	100	23.40053	SOHJIDD12AB0183260
7112.0	100	20.039148	SOGCGBA12AC3923C4F
2288.0	100	19.799877	SOIGCPA12A58A79382
5829.0	100	19.785376	SOANOQW12A58A793D2
4298.0	100	19.346773	SONNWBG12AB0180C00
286.0	100	18.533035	SOMDVSL12A6D4F7230

Fig(4)

Inference:

Using hive sql basic analysis is done, where we found most popular artist, total unique artist, song of the year, top 10 popular tracks etc . This helps us to develop interests in users.When we consider interest of a particular user, recommendation is very helpful which is done using spark. Songs are recommended for every individual user by item based collaborative filtering which implemented using als algorithm.

III. CONCLUSION:

By doing the analysis and recommendation using hive and spark we conclude that each user is recommended with a unique playlist based on their interest. This also provides the information for the user like wh/at are the most popular songs, who are all the artist, which artist is liked by the people through which he gets to know about the current trend.Music recommender system plays a significant role in identifying a set of music for users based on user interest. Although many move recommendation systems are available for users, these systems have the limitation of not recommending the music efficiently to the existing users. This paper presented a music recommender system based on collaborative filtering using Apache Spark. From the results, the selection of parameters of ALS algorithms can affect the performance of building of a music recommender engine.System evaluation is done using various metricssuch as execution time, RMSE of rating prediction, and rank in which the best Two best cases are chosen based on best parameters selection from experimental results which can lead to

building god prediction rating for a music recommender engine. From these cases, the lowest value of the RMSE is considered the best case for prediction in building music recommendation system. Therefore, the second case is recommended to be used since the value of the RMSE is smaller compared to the value in the first case as well as adopt the second case as the best case, because there is no significant difference in the amount of time execution between the two cases.

REFERENCE:

[1]. (2018). Datajobs.com. Retrieved 22 November2018,from https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf

[2]. Music Genome Project. (2018). En.wikipedia.org.RetrievedNovember2 2018,from https://en.wikipedia.org/wiki/Music_Genome_Project

[3]. Simple Matrix Factorization example on the Millionsong dataset using Pyspark. (2018). Medium.Retrieved22November2018,from https://medium.com/@connectwithghosh/simple-matrix-factorization-example-on-the-musiclens-dataset-using-pyspark-9b7e3f567536\

[4]. Jean-julien Aucouturier and Francois Pachet. Music Similarity Measures: What isthe Use. In Proceedings of the ISMIR, pages 157–163, 2002.

[5]. Luke Barrington, Reid Oda, and G. Lanckriet. Smarter Than Genius? HumanEvaluation of Music Recommender Systems. In 10th International Society forMusic Information Retrieval Conference, number ISMIR, pages 357–362, 2009.

[6]. Kerstin Bischoff, Claudiu S Firan, RalucaPaiu, Wolfgang Nejdl, L S De, Cyril Laurier, and Mohamed Sordo. Music Mood and Theme Classification - A Hybrid Approach.