Recommender System With Sentiment Analysis

Raktim Bijoypuri IIT2018125
Semester VI, Department of Information Technology,
Indian Institute of Information Technology, Allahabad, Prayagraj.

Abstract

The aim of this study is to develop a scalable recommender system using not only user ratings but also user reviews. Various techniques like ALS matrix factorization, weighted normalisation of ratings are used for this purpose. The proposed model has been compared with state of the art methods. This recommender system will be helpful for big data applications. As the data generated by users reviews and ratings is large in size, distributed computing helps to process the data in an efficient manner for storage and fast computation purposes that leads to better recommendations. In this study, our proposed model has improved performance compared to state of the art models by using NLP, sentiment analysis along with additional techniques mentioned above for better performance and accuracy.

Keywords

Distributed systems , Recommender systems, Big data, Sentiment analysis.

1. INTRODUCTION

With increasing usages of e-commerce, huge data is being generated every day. Big data has been an important field in data mining. This data is so huge that our systems cannot handle these computations. Various methods are being developed to manage this data. That is why study of big data is so important nowadays. Big data methods are very helpful when we are dealing with such large data. Users generate various types of data by their activities. They give ratings to products and also state their opinion about a particular product by writing reviews. This information can be wisely used and is very helpful from a data mining perspective. Companies can learn a lot about user preferences by using this data. The data have many features and fields divided into various categories. This will give us more data as well as a concrete base of the opinion of the user about the product. For clients, a recommendation system is a well built gadget for furnishing user related content and known to be a very powerful tool. For searching suitable content fast and easily the suggestions are provided to assist the client and to manage with the data overload. In content based recommendation item

profiles are generated for each of the items liked by the user, these item-profiles are then matched with other items to estimate whether the user will like it or not. In view of the running of user-generated content and online information collaborative filtering is presumed to be the most broadly deployed and popular technique in the recommender system. With efficient use of

these techniques, users can get better recommendations while browsing for products and services.
In this paper, we present a scalable distributed recommender system. We suggest the architecture using collaborative filtering and sentiment analysis with ALS algorithm parallelised on distributed systems. This paper is organized as follows: section 2 describes the relevant study done on existing state of the art methods of recommendation systems. Section 3 describes the dataset we have used. Section 4 is experimental setup. Section 5 states the methodology we adopted while section 6 states the experimental results we obtained from this research. Section 7 includes conclusion and discusses future works.

2. LITERATURE REVIEW

Yeliz Yengi et al. proposed sentiment analysis instead of user rating to devise a recommender system for big data applications[1]. **Hadoop distributed file system (HDFS)** is used to store a big amount of data on a file system. Using the **MapReduce approach**, batch learning is done to build the model. **Lexicon based sentiment analysis** were to detain the matter of subject,For generating a positive or negative lebel ,the judgement in text need to process.The recommender system predicts the rating and preference that a user gives to an item. **Collaborative filtering is used for better understanding of the sentiment.** Traditional star rating has more **RMSE** than quality of different sentiment rating .

In the paper by Shigang Hu et al, they proposed a recommendation methodology which is devised from **fine-grained feature sentiment analysis and reviewer credibility analysis**[2]. The **CISER** model contains mainly **5 modules** which are **recommendation module,extraction feature candidate, client interest, user feature, assignment of sentiment.** Solely rating oriented analysis is not sufficient as many other factors affect the user decisions like brand loyalty, product prejustice. With the help of feature sentiment value we classify the product recommendation.Encourage model by them help to score weighted sentiment credibility of customer to sale and purchase.However content and percentage security of user is done by quantifying review utility.The model suggested by them also uses sentiment which are credibility weighted for purchase recommendations scoring of user preferred features. With this technique, they obtained a better mean **average precision of 93% for CISER** than recent techniques.

A movie recommendation framework based on sentiment analysis and a hybrid recommendation model which is proposed by name Yibo Wang [3]. Then **sentiment analysis** is employed to optimize the list. Sentiment analysis is also employed in this system. According to them, **content-based approaches and collaborative filtering are balancing each other and can compensate for the shortcomings of each other which provide stability and accuracy.** For this purpose, they used **Douban movie data** which is divided into user data, movie data, and review data. User data and movie data is used for collaborative filtering method, while review data is used in content-based method. A hybrid recommendation method is used with sentiment-Based Recommendation Module which includes **Chinese Word**

Segmentation and sentiment analysis. Recommendations are ranked according to the scores that are derived from the calculation of the content-based recommendation and collaborative filtering method. The combination of content based and collaborative filtering method with sentiment analysis makes the model perform better with an **F1 score of 0.771** than traditional model with content-based recommendation and collaborative filtering with **F1 score of 0.582.**

In the paper, **Deep Learning Sentiment Analysis For Recommendations In Social Applications**[4], different techniques of sentiment analysis are explained in detail and the ML algorithms are explained to do the same. We read about the **Collaborative Filtering(CF), Content based Recommender System,** and **Hybrid Recommender System.** These are some of the techniques used to identify and analyse the sentiments of people. There are other techniques also, such as Cross-Domain Recommendation System and Constraint based Recommendation System but nothing much is written about them in this paper. Furthermore, we read about the ML Algorithms that can be used to do sentiment analysis. Several of them are explained in detail including **Deep, dense Networks, Recurrent Neural Networks(RNN), Recursive Neural Networks(RNN) and Convolutional Neural Networks(CNN).** As per conclusion, RNN(Recursive Neural Networks) are best suited for sentiment analysis and based on the tree representation of a sentence, RNN is constructed and can understand the relationships between words in a sentence more adequately well suited for sentiment labels training and exhibits better performance.

In the paper, **Novel Recommender Systems Using Personalized Sentiment Mining**[5], they made a model for sentiment analysis and movie recommendation. Various methods are analysed such as Unigram, Bigram, Trigram etc. for sentiment mining. Experiments were performed using popular algorithms such as **Random Forests, Bernoulli Naive Bayes**, etc. The **yelp** dataset is used for sentiment analysis. It has:

1. 2.7M Reviews
2. 649K Tips
3. 687K Users
4. 86K Businesses
5. 4.2M Social Edges

For the Recommender system, **Movielens dataset** and **jester** dataset is used. Sentiment Analysis results are as follows. For Bernoulli Bayes, RMSE is **1.4404** and for Random Forest is **0.7577** which is the highest and the lowest respectively. Same results for Bigram, Trigram, Bi-Trigram Bayes and Naive Bayes are **0.87128, 1.2496, 0.8962** and **0.8505** respectively.

For the Recommender system, RMSE of **0.821835** and **0.9721** is achieved in the 20M and 10M dataset of jester respectively. Sentiments are added to make the recommendations more personalised. Random forest achieved the best result as it can be seen from the table.

In the paper, **Detecting user preferences using Sentiment Analysis in Smart Cities,** they propose a model which uses the user comments and reviews on the **TripAdvisor** Social Network[6]. These reviews are used as a dataset for sentiment mining. The training data includes the reviews of 100 tourists of different nationalities and ages about various attractions within 6 months (from January 2018 to June 2018). Those tourist's comments in the first trip after June, 2018 are included in the testing data. Sentiment analysis is conducted using SentiWordNet 3.0. The method proposed in the paper extracts tourists' preferences which is based on their reviews and sentiment analysis along with performing text mining. Noun clustering is done on the reviews. Afterwards, a score is given to each of the clusters based on its word frequency and sentiment analysis. The cluster with the highest score shows the tourists' preferences. The whole process is divided into following 4 stages:

- **Preprocessing** : Parts of speech tagging, stop word elimination, noun extraction and stemming
- **Creating Similarity Graph** : creating noun matrix, normalising it and creating graph based on this matrix
- **Clustering** : transferring sentences to related clusters after removing redundant edges
- **Extracting Preferences** : scoring clusters and extracting user preferences.

The proposed method achieved high precision with an accuracy of **93.05%.**

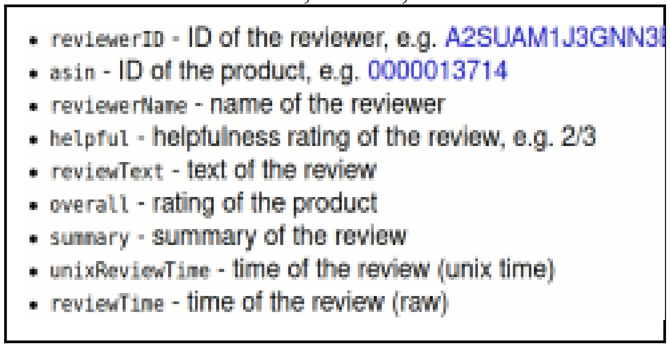
In the paper, **Movie Recommendation System Using Sentiment Analysis From Microblogging Data**[7], they have proposed a Hybrid Recommendation system that uses movie review tweets and other user reviews to understand the current trends, public sentiments and the user response of the movie. In short, it collects data about the movie by various means mentioned here. A user-rated movie database is needed, where ratings for relevant movies are present, and another dataset needed is the user tweets from Twitter. **The MovieTweetings** database is used in this paper because it provides up-to-date movie ratings so that it contains more realistic data for sentiment analysis. **The MovieTweetings** database contains the movies with published years from 1894 to 2017. Due to the lack of tweets for old movies, they only considered a subset of the database which contained only those movies for which tweets are also available and these are the movies released in or after 2014. The final selected database consisted of 292,863 ratings by 51,081 users on 6209 different movies. At first, tweets are preprocessed and then words are matched with sentiments. **Hybrid Recommendation System** is a combination of collaborative filtering and content-based filtering. The results of this Hybrid model are compared with another Pure Hybrid model(PH model) which is also a combination of collaborative filtering and content-based filtering. It is found that the model proposed in this paper achieves a better precision value in both cases with 2.54 for Top-5 and 4.97 for Top-10 in comparison with the PH Model. Thus, it is concluded that this method will suggest at least two recommended movies out of five and five recommended movies out of ten.

Noemi Mauro et al.[10] focused on a set of helpfulness determinants. They selected some parameters like length, rating, coherence and polarity. They didn't consider the review date or review age as it is a partial indicator. They used a subset of Yelp 2019 dataset for their study. They categorized the variables as dependent like review helpfulness and independent like length, context, etc. Review polarity and helpfulness was correlated with ratings for better predictions.

Sumaia Mohammed Al-ghuribi et al.[11] suggested that review helpfulness can help and be used in recommendation systems for better predictions. More is the rating, more is the quality score assigned to that review. Using these features along with the polarity of users, they created user profiles. These user profiles are then used to calculate the product score which is used to recommend a product to a particular user. They used data from TripAdvisor's website to evaluate their model.

3. DATASET DESCRIPTION

For this project, Amazon Review Dataset (2018) [9] have been utilized. It is an updated version of the Amazon Review Data released in 2014. This dataset includes more and newer reviews along with metadata related to product like color, product type, technical details and product images taken by the users. It also includes the 2014 dataset's reviews, product metadata and links. The dataset is also divided into smaller subsets for training purposes like K-cores is arranged such that each of the remaining users and items have k reviews each along with Ratings only dataset which includes no metadata or review but only (item, user, rating, timestamp) tuples. The total number of reviews is 233.1 million (34 GB). This dataset contains various categories like - Musical Instruments, Books, Amazon Instant Video, Digital Music etc.



- reviewerID - ID of the reviewer, e.g. A2SUAM1J3GNN3
- asin - ID of the product, e.g. 0000013714
- reviewerName - name of the reviewer
- helpful - helpfulness rating of the review, e.g. 2/3
- reviewText - text of the review
- overall - rating of the product
- summary - summary of the review
- unixReviewTime - time of the review (unix time)
- reviewTime - time of the review (raw)

Fig 1 : Dataset description (from jmaculey)

4. EXPERIMENTAL SETUP

We have used Google Colab and Apache spark(3.1.1), python 3.7 to carry out our experiment. Colab provides us with a single GPU cluster. It is a NVIDIA K80 GPU, 12 GB RAM, 0.82 GHz clock speed, 4.1 TFlops computation.

5. PROPOSED METHODOLOGY

The proposed methodology contains three sections that have been discussed below:

The dataset is providing us the data in a json format. So at first step we will be converting the json data into a pandas dataframe. The data have many features and fields divided into various categories. Among them the fields namely - 1) reviewText, 2) Overall (rating), 3) votes are what we are interested in.

Most of the models and researches done on this dataset and other similar ones only include the overall (rating) into the consideration of recommendation. In our case apart from overall (rating) we have also taken the reviews given by users into consideration. This will give us more data as well as a concrete base of the opinion of the user about the product. The up/down vote counts are used to assign a weight into the review to gain more accurate results.

5.1 Distributed Computing

On a popular e-commerce website like Amazon thousands of users leave billions of reviews, ratings etc important information about various products and it keeps growing with the time. To recommend a product we need to take these enormous amounts of data into consideration. Only then our model will predict with a good accuracy.

To keep up with this increasing amount of data handling we can do these on a single machine with enhanced hardware and power. However as the dataset we are dealing with is quite large, a single machine with enhanced hardware might not be sufficient. We may not be able to scale it properly as the workload increases. One efficient solution to overcome this problem is distributed computing.

Distributed computing[15] allows us to share the information with different computers. In this paradigm a larger task is broken into several small parts and distributed among a set of different computers. These computers then execute the task separately. Finally the results are combined. In this way a larger task can be done in a faster and efficient manner. It will be more fault tolerant than single node operations and we can scale our system with the increasing workload to meet the requirements. To implement distributed computing we have to use algorithms that can be distributed over separate nodes. Apache Spark will be used in obtaining this parallelism.

5.2 Sentiment analysis

Sentiment analysis[14] converts the review texts into a numerical value on a certain scale made by us. This will give us more numerical data upon which we can train our models and get more accurate results.

To convert the review text into a numerical value we need to analyze them and figure out the underlying sentiment of the user about the product. We will be doing sentiment analysis of the product reviews. It will require Natural Language Processing (NLP). This will be done in the following steps.

Step1:- The review texts will be converted into a spark data frame which will be divided into separate nodes.

Step2:- In each node the text data will be preprocessed. The stop words, punctuations will be removed.

Step3:- After that a word2vec model will be created from the remaining text. A word2vec model is actually a matrix representation of the text. It takes the text as an input and returns a sparse matrix, each column representing each word. If a word is present in a review, then the corresponding row of that column is set to true or 1.

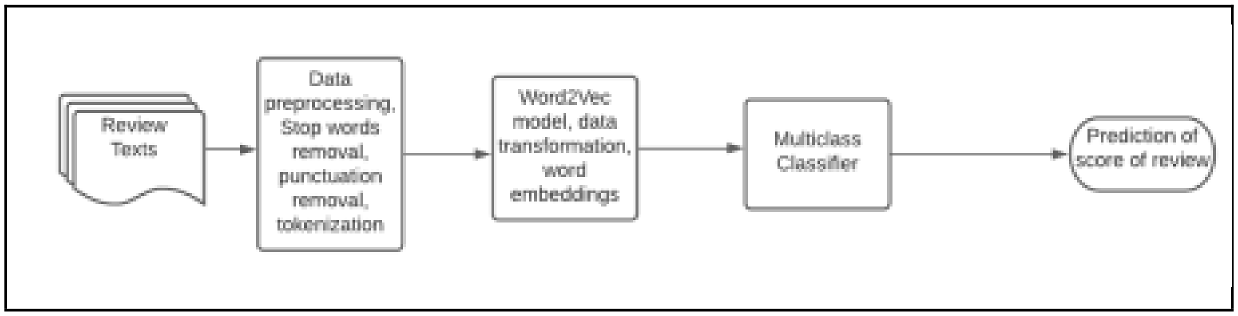

Review Texts
Data preprocessing, Stop words removal, punctuation removal, tokenization
Word2Vec model, data transformation, word embeddings
Multiclass Classifier
Prediction of score of review

Fig 2: Sentiment analysis

Step4:- The resultant matrix from step 2 will be very sparse and therefore difficult to use. To help with that, embedding spaces is needed. Embedding spaces or word embeddings will make semantically similar words to be near each other within the embedding space which will allow the NLP model to share information across various words of a similar nature.

After this a multiclass classifier will be used on this data to give each review a score on a scale from very good to very bad. The whole dataset will be converted into a rating matrix and it will be sent to the next steps of our model.

5.3 Recommendation algorithm

For making recommendations 2 popular algorithms are commonly used -

1) Content-based
2) Collaborative filtering

5.3.1 Content based Filtering:

Content based filtering[16] generates item profiles for each of the items liked by the user, these item-profiles are then matched with other items to estimate whether the user will like it or not. This method does not use other information that the system may have of other similar users and fails to generalise because it rarely gives recommendations outside of the matched item-profile(overspecialisation).

5.3.2 Collaborative Filtering:

Collaborative filtering[13], incorporate the ratings given by other users to find similar users and use their ratings and the products they have seen to estimate ratings for new unseen products.

While collaborative filtering has its own set of problems such as cold-start or recommending popular items or the fact that the rating matrix is very sparse, it is still very widely used and proven to give good enough recommendations so we will be using this method for our project.

Collaborative filtering challenges:

(a) **Cold start:** This problem occurs when users have not rated any products yet. Ratings are not available for this user. Also new products arriving do not have ratings associated with them. These factors result in null values being fed into the item-user interaction matrix and collaborative filtering may not give accurate results.

(b) **Data sparsity:** This problem is associated with lack of information. If very few items in the dataset are rated, the item-user matrix gets filled with more null values. This can lead the collaborative filtering technique to make weak predictions due to lack of neighbors. It might also result in recommending a set of products only.

We wanted to build a system capable of working with large datasets so we decide to use Matrix factorization with ALS(Alternating least squares), this method has been shown to work well with large datasets because it can be parallelised efficiently with only about 5-10% overhead of communication costs due to parallelisation.

5.4 Matrix Factorization using ALS:

Formally, given a system of X users and Y movies we want to factorize the given rating matrix R of order X*Y, into 2 matrices of order X*K and K*Y respectively, here K represents latent-features. Although latent-features don't have any physical meaning after factorization, we can think of them as representations of factors such as is the movie a comedy? Does it contain action? etc. In this way it is similar to a hidden layer in neural networks.

There can be multiple latent-features which tie people together, **the number of latent-features is a hyper-parameter subject to experimentation.**

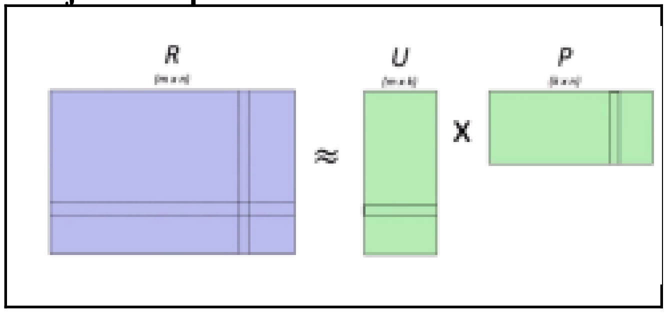


Fig 3 : Matrix factorisation (*source*)

Matrix factorization can also help us to save a lost of space for example for a rating matrix of size X*Y if we factorize it into X*K and K*Y then the memory required for storing these 2 matrices is O(K*(X+Y)).Usually K is small so the amount of memory required reduces greatly.

As discussed above we want to reduce the mean-square error for all the ratings, this can be mathematically expressed as(all equations are taken from [8]) -

$$\mathcal{L}^{emp}(R,U,M) = \frac{1}{n}\sum_{(i,j)\in I}\mathcal{L}^2(r_{ij},\mathbf{u_i},\mathbf{m_j}),$$

Note that the square error is calculated only for the points (i,j) for which a rating exists, further to prevent overfitting we regularize the above error using **regularisation.**

$$f(U,M) = \sum_{(i,j)\in I}(r_{ij}-\mathbf{u}_i^T\mathbf{m}_j)^2 + \lambda\left(\sum_i n_{u_i}\|\mathbf{u}_i\|^2 + \sum_j n_{m_j}\|\mathbf{m}_j\|^2\right)$$

To minimise the mean-square error we use the ALS Algorithm.

5.4.1 ALS Algorithm[12]

In this algorithm we first assume the matrices U, M randomly. Then we alternately try to minimize the mean-square error by first fixing the matrix ‘U’ and minimising for ‘M’ and vice-versa (hence the name **Alternating**).

For example if we first keep the matrix ‘M’ as fixed then we want to compute the matrix ‘U’ such as the error is minimised, for doing this we use the following formula (derived by setting the derivative of mean-square error function equals to zero with respect to user matrix U) -

$$\mathbf{u}_i = A_i^{-1}V_i, \quad \forall i$$

Where the matrices �� and are calculated as

������

$$A_i = M_{I_i}M_{I_i}^T + \lambda n_{u_i}E$$

$$V_i = M_{I_i}\ R^T(i, I_i)$$

Here

�� = Set of movies rated by the user “i”
��

�� = row “i” of the user matrix U.
��

�� Sub-matrix M with only columns from ����= ����

��= Identity matrix of order K*K(latent features)

��(��, �� = ith row of rating matrix with only columns from .

��) ����

Similar calculations are done for updating M when U is kept fixed, this alternating least squares method continues till a specified number of iterations or till the given mean-square error threshold is reached.

5.4.2 Parallel ALS -

We want to parallelise the updation of U and M, for this we divide the rating matrix in 2 different ways first by distributing a group of rows and another by distributing groups of columns.

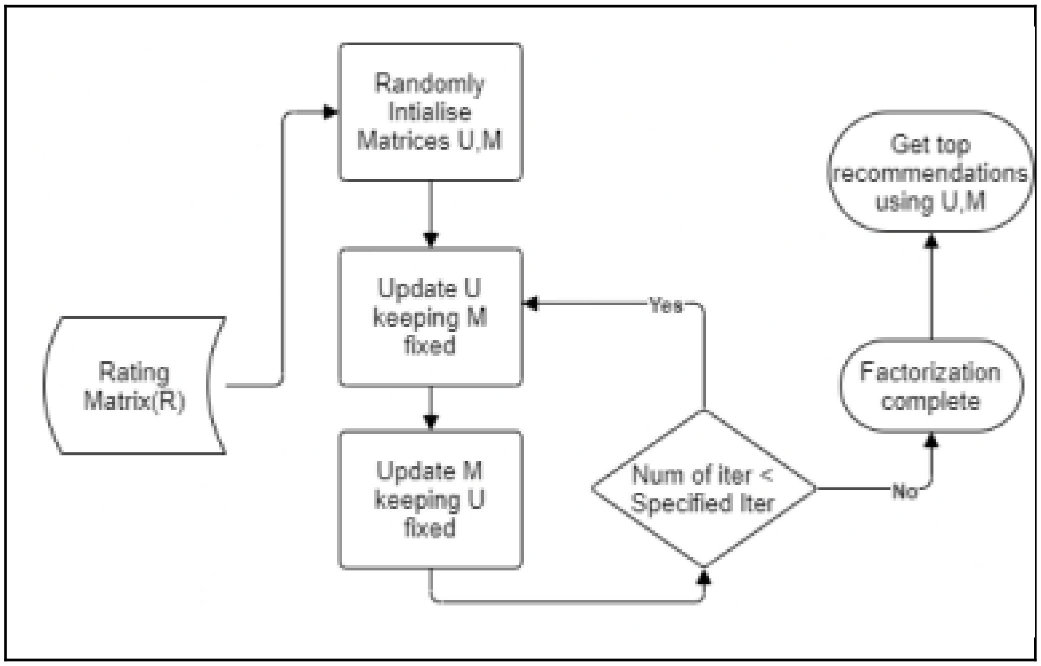

Randomly Intialise Matrices U,M
Update U keeping M fixed
Update M keeping U fixed
Rating Matrix(R)
Num of iter < Specified Iter
Yes
No
Factorization complete
Get top recommendations using U,M

Fig 4 : ALS flowchart

When we want to update M, each of the processes have their own private copy of U and the local copy of R distributed by columns. Each of the processors only requires the above 2 parameters for calculating columns of M corresponding to the small local-copy of R that it has which ensures that work is efficiently divided among all the processors.

Once each the processors finish calculating their portion of the matrix M, they gather all their results by broadcasting the information to each other, this is done only after M is calculated completely so the performance overhead because of this is not too large (roughly around 5-10%

of the total time), also the number of ALS iterations is not too huge to impact performance. Similar approach is also used for calculating U, while keeping M fixed, the only difference being that this time the rating matrix R is distributed by groups of rows and not columns.

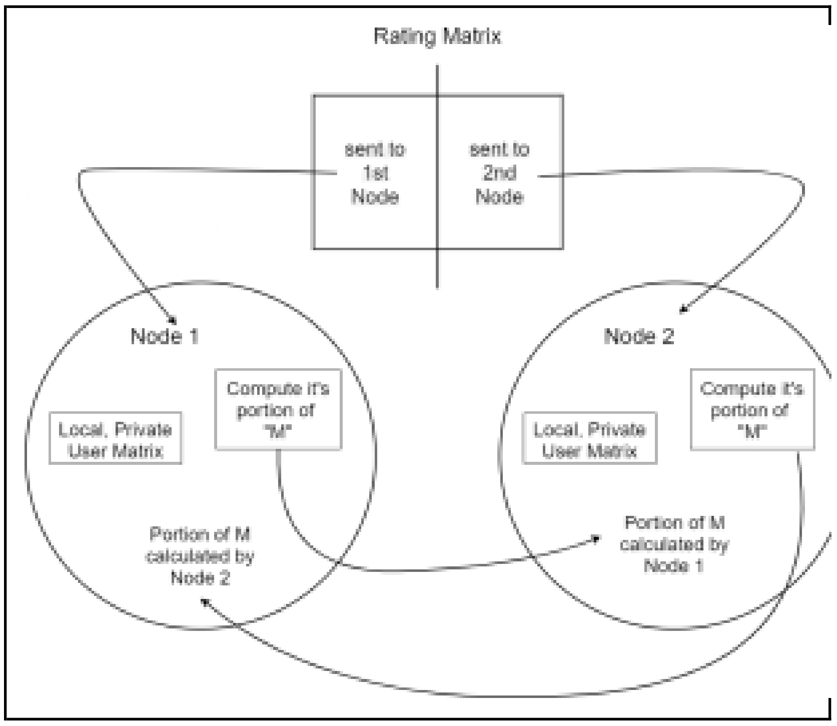

Rating Matrix
sent to 1st Node
sent to 2nd Node
Node 1
Local, Private User Matrix
Compute it's portion of "M"
Portion of M calculated by Node 2
Node 2
Local, Private User Matrix
Compute it's portion of "M"
Portion of M calculated by Node 1

Fig 5 : Updating M, keeping U fixed

5. 5 Weighted Normalisation of Ratings (WNR):

We often see an upvote system in many applications. Websites arrange their reviews according to the helpfulness score of the review. Highest voted reviews are arranged at top. These votes play an important role for users in deciding whether the review is helpful or not. It helps users to determine the reliability of reviews.

Users tend to believe reviews having highest upvotes and their decisions also get affected by these. So we decided to take into consideration the helpfulness of the reviews. Normalising these ratings resulted in a change of range from 1-5 to 1-6. Following formula is used for normalising the ratings:

��������������������_������������ =

��������_������������ * 0. 8 + ��������_������������� *

0. 2 + ��������������_������������ Where, *overall_effect* is

calculated as:

$$overall\_effect = \frac{1 + pos}{pos + neg}$$

for *pos* > *neg*

else

$$overall\_effect = \frac{1 - neg}{pos + neg}$$

Where,

pos = number of people who found that review helpful

neg = number of people who found that review not helpful

Rather than using the original ratings, these normalised ratings are used in the model which **improved the accuracy** as compared to conventional ratings.

6. EXPERIMENT AND RESULT DISCUSSION

6.1 Sentiment Analysis:

Our experiment process begins with getting the rating equivalent of the reviews. From the original dataset we take only the reviews column. After that a pipeline is created. This pipeline contains stages that convert our text input of reviews into a more refined sparse matrix of numbers which is easier for a classifier to work with. At the first stage of the pipeline we tokenize the reviewText followed by removal of the stop words. We convert the resultant data into a word vector model where each word is represented as a vector of numbers. TF-IDF method is used in doing so. Finally a logistic regression classifier is applied on this converted data which gives us a numerical value in range of 1 to 5. This rating equivalent of reviews is then fed to our next phase i.e, the Recommendation algorithm.

As our experimental dataset we chose Musical Instruments, a category which has a comparatively smaller amount of user reviews and ratings. This is suitable for our hardware setup.

We obtained best accuracy of 61.38% for sentiment analysis part using hyper-parameters(maxIter=20, regParam=0.3 tuned using cross validation)

6.2 ALS:

6.2.1 Hyperparameter Tuning :

While factoring the rating matrix we need to pass several hyperparameters such as rank,maxIter. For this we use cross validation to obtain the best set of parameters. For our dataset we obtained the following results -

```
Rank: 50

MaxIter: 20

RegParam: 0.1
```

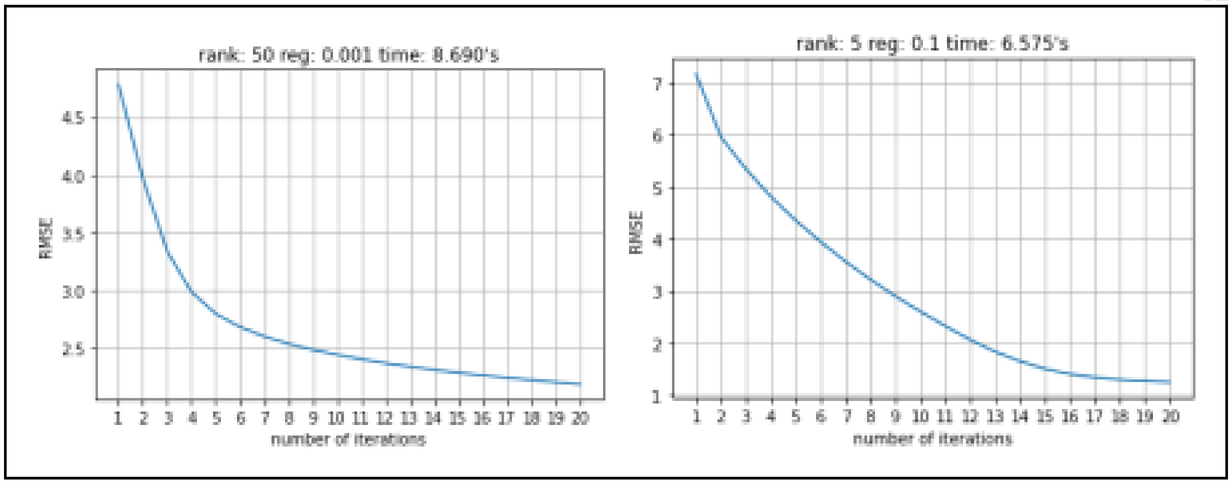


Fig 6: RMSE vs Iterations

Note that, changing these hyper-parameters results in changes to which value RMSE converges to and at what rate, as shown below.

6.2.2 RMSE Results :

We observed reduction in RMSE on normalised rating, however it must be remarked that this does not necessarily result in a better recommendation system because we didn't better our result on the same rating matrix but on the normalised rating matrix using WNS etc mentioned earlier.

The results are presented below on some of the categories we evaluated our model on -

Category	**RMSE on original Rating**	**RMSE on normalised rating**
Musical Instruments	1.100	0.97
Amazon Instant Video	1.10	0.91
Digital Music	1.017	0.85
Tools and home improvement	1.16	0.97

Table 2: RMSE Results

6.2.3 Comparison with state of the Art Methods:

We compare our results with result of the following paper on Digital Music dataset -

Paper RMSE	
Sentiment Analysis + review based Item and User profiling ,R. K. Chaurasiya and U. Sahu[17]	0.65

Table 3: Comparison

Our implementation has an RMSE of 0.85 which shows **our model was not able to achieve better results in comparison with the state of the art model[17] which uses item based collaborative filtering .** However ALS has been shown to perform better on large datasets and has good scalability so a higher RMSE does not necessarily mean it's bad.

6.2.4 Scalability :

We Measure the time taken by our model for loading/processing the dataset and factorisation of the rating matrix using ALS to find out how the model behaves on increasing the number of reviews

S.No	Size(MB)	Number of reviews	Time(in seconds)
1.	7.4	10,261	22
2.	28.1	37,126	49
3.	89	64,706	113
4.	112.4	134,476	171

Table 4 : Scalability Analysis

Plotting the results in the above table we observe that time does not strictly increase linearly with number of reviews instead the slope of the line increases with the number of reviews, but still the time taken does not increase exponentially which is a good sign as we want it to perform well on even larger datasets.

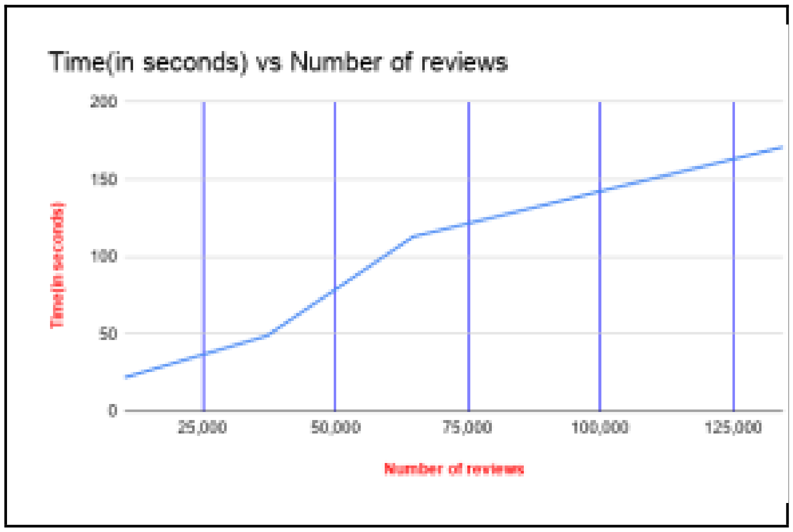


Fig 7: Scalability analysis

7. CONCLUSION

Recommendation systems have been very useful both for big data applications and users point of view. They provide better and personalised products based on user's data which eventually saves the time and efforts of users. As users tend to go with products and services recommended to them, big data applications also get benefitted. For better recommendation, quality of data is useful along with the performance of large data. In this study, we assessed the performances of state of the art recommendation systems and proposed a better model. We performed sentiment analysis along with collaborative filtering. Distributed computing is used with the help of Apache Spark for better handling of big data. Additionally, we have implemented our own weighted normalization of rating to improve the data quality which produced better results. ALS Matrix factorization can be used for the recommendation part to make a scalable system capable of handling large datasets. This paper shows that with the help of user rating and reviews with appropriate data processing a scalable recommendation system can be built with better performance and efficiency.

8. REFERENCES

[1] Yengi, Yeliz & Omurca, Sevinc. (2016). Distributed Recommender Systems with Sentiment Analysis. European Journal of Science and Technology. 4. 51-57.

[2] S. Hu, A. Kumar, F. Al-Turjman, S. Gupta, S. Seth and Shubham, "Reviewer Credibility and Sentiment Analysis Based User Profile Modelling for Online Product Recommendation," in IEEE Access, vol. 8, pp. 26172-26189, 2020, doi: 10.1109/ACCESS.2020.2971087.

[3] Yibo Wang, Mingming Wang, Wei Xu, "A Sentiment-Enhanced Hybrid Recommender System for Movie Recommendation: A Big Data Analytics Framework", Wireless Communications and Mobile Computing, vol. 2018, Article ID 8263704, 9 pages, 2018.

[4] Kandasamy, Devipriya & Dhandayudam, Prabha & Pirya, V & Sengan, Sudhakar. (2020). Deep Learning Sentiment Analysis For Recommendations In Social Applications. International Journal of Scientific & Technology Research. 9. 3812-3815.

[5] B. S. S. Govind, R. Tene and K. L. Saideep, "Novel Recommender Systems Using Personalized Sentiment Mining," 2018 IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), Bangalore, 2018, pp. 1-5, doi: 10.1109/CONECCT.2018.8482394.

[6] Z. Abbasi-Moud, H. Vahdat-Nejad and W. Mansoor, "Detecting Tourist's Preferences by Sentiment Analysis in Smart Cities," 2019 IEEE Global Conference on Internet of Things (GCIoT), Dubai, United Arab Emirates, 2019, pp. 1-4, doi: 10.1109/GCIoT47977.2019.9058397.

[7] S. Kumar, K. De and P. P. Roy, "Movie Recommendation System Using Sentiment Analysis From Microblogging Data," in IEEE Transactions on Computational Social Systems, vol. 7, no. 4, pp. 915-923, Aug. 2020, doi: 10.1109/TCSS.2020.2993585.

[8] Large-scale Parallel Collaborative Filtering for the Netflix Prize, Yunhong Zhou, Dennis Wilkinson, Robert Schreiber and Rong Pan

[9] Justifying recommendations using distantly-labeled reviews and fine-grained aspects, Jianmo Ni, Jiacheng Li, Julian McAuleyEmpirical Methods in Natural Language Processing (EMNLP), 2019.

[10] Mauro, Noemi & Ardissono, Liliana & Petrone, Giovanna. (2020). User and Item-aware Estimation of Review Helpfulness.

[11] S. M. Al-Ghuribi and S. A. Mohd Noah, "Multi-Criteria Review-Based Recommender System–The State of the Art," in IEEE Access, vol. 7, pp. 169446-169468, 2019, doi: 10.1109/ACCESS.2019.2954861.

[12] Cichocki, Andrzej, Rafal Zdunek, and Shun-ichi Amari. "Hierarchical ALS algorithms for nonnegative matrix and 3D tensor factorization." In International Conference on Independent Component Analysis and Signal Separation, pp. 169-176. Springer, Berlin, Heidelberg, 2007.

[13] Herlocker, Jonathan L., Joseph A. Konstan, and John Riedl. "Explaining collaborative filtering recommendations." In Proceedings of the 2000 ACM conference on Computer supported cooperative work, pp. 241-250. 2000.

[14] Liu, Bing. "Sentiment analysis and subjectivity." Handbook of natural language processing 2, no. 2010 (2010): 627-666.

[15] Thain, Douglas, Todd Tannenbaum, and Miron Livny. "Distributed computing in practice: the Condor experience." Concurrency and computation: practice and experience 17, no. 2‐4 (2005): 323-356.

[16] Pazzani, Michael J. "A framework for collaborative, content-based and demographic filtering." Artificial intelligence review 13, no. 5 (1999): 393-408.

[17] R. K. Chaurasiya and U. Sahu, "Improving Performance of Product Recommendations Using User Reviews," 2018 3rd International Conference and Workshops on Recent Advances and Innovations in Engineering (ICRAIE), 2018, pp. 1-4, doi: 10.1109/ICRAIE.2018.8710414.

[18] L. Chen, R. Li, Y. Liu, R. Zhang and D. M. Woodbridge, "Machine learning-based product recommendation using Apache Spark," 2017 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computed, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), 2017, pp. 1-6, doi: 10.1109/UIC-ATC.2017.8397470.