International Journal of Recent Engineering Research and Development (IJRERD)
ISSN: 2455-8761
www.ijrerd.com || Volume 05 – Issue 05 || May 2020 || PP. 14-20

Movie Recommendation Analysis using Alternating Least Square Apache Spark

Yosua Alvin Adi Soetrisno[1], Enda Wista Sinuraya[2], Eko Handoyo[3], Ajub Ajulian[4], Karnoto[5], Sudjadi[6], Tejo Sukmadi[7], and Imam Santoso[8]
[1,2,3,4,5,6,7,8]Diponegoro University, Department of Electrical Engineering,
Jl. Prof. H. Sudarto, Semarang, Indonesia

Abstract: This research is conducted to test alternating least square algorithm on a medium data set. Alternating least square is algorithm to search the recommendation from other user based on other user rating. In case of movie recommendation, there are set of rating taken from IMDB resources. The problem that exist is that the data is growing in sparsity and volume. Clustering algorithm has been used to make a user preference grouping. Preference of the user is dynamically changing and the change is affecting the structure of user relation according to ranking and movie preferred. Alternating least square is selected as algorithm for collaborative filtering. We don't use cosine similarity because our dataset don't contain movie genre. This research also use Spark as resilient dataset engine that could speed up the aggregation process of analytics. The result is tested by comparing explicit parameter and implicit parameter from rating. Implicit parameter update the preference based on the positive and negative review. Measurement of ALS performance is using RMSE. RMSE in implicit parameter which is 3.03 give a greater error that explicit parameter which is 0.71.RMSE must be compared with top twenty recommendation for adjust the recommendation not based only on error rating criteria. The iteration of ALS need a lot of memory, and increasing in few iteration did not make a very significance change, so we used only 10 maximum iteration.

Keywords: alternating least square, recommendation system, Spark, big data, Hadoop, performance test

1. Introduction

Movie is one of interesting topic that growth today. All movie is developed based on the market of the user. User analytic has to be developed to gain the partially graph of market analytics. Movie is growth by viewer review in social media or in IMDB rating as the basic survey rating on some movie critic. IMDB database is including all information of the movie, like the title, the director, the actor, the synopses and rating. Database of the IMDB is growingly fast and could become big data. Big data analysis is helping to collect the useful information between the reviewer relations to gain the successful movie recommendation. Movie recommendation is briefly used in movie's streaming service company like Netflix, to suggest the movie's hobbyist to subscribe a lot of interesting film gained from movie recommendation. Netflix gained 75% subscriber only based on movie recommendation[1]. Recommendation engine could be developed from content based filtering, which is collaborative filtering and hybrid. Collaborative filtering build recommendation based on another user rating which have same preference with some user. Collaborative filtering could be used as crowd based knowledge. The main problem of collaborative filtering is that the data is growing big as the number of movie reviewer. Matrix of the user that could be used in collaborative filtering could end with error because the sparsity between each genre is not very balanced. Based on some research, we could know that alternating least square is used to predict rating and build personal recommendation based on big data. Because of the big data, we could use parallel technique for processing for distributing task on some part of dataset but with aggregating comprehensive result. In this research is using library from Apache Spark which is "MLLib". "MLLib" is special library for machine learning algorithm including alternating least square. Another predecessor research is build collaborative filtering with kNN algorithm or using SVM with some matrix factorization. Problem that exist is that Root Mean Square Error still high, but there is another research that test that alternating least square is better [2]. This research is early research to gain recommendation preference based on big data map reduce mechanism. Modification of algorithm that could breakdown the movie genre is not done in this research.

The aim of the research is to test the explicit and implicit feature feedback. RMSE value in explicit parameter is fewer than implicit but could not directly represent best recommendation scenario, so this research want to test, if there is hidden feature that was not considered before. This research also does a comparison in maximum iteration trial of ALS algorithm, are maximum iteration affect the accuracy. This research also tests the performance of resilient distributed dataset based on Apache Spark that combined with HDFS. Data that used is read from HDFS.

2. Related Work

Research conducted by Suganeshwari [3] uses time adaptive collaborative filtering because movie is having difference preference in some period only. Alternating least square is combined with lazy collaborative filtering with dynamic neighborhood (LCFDN). LCFDN adopts time with tow defined feature which is alfa and delta. Alfa is used to truncate the neighborhood selection in similarity evaluation based on time stamp. Beta prunes the recent transaction uses from the large past purchase history available [4]. Another research using personalization sentiment analysis [5]. Sentiment analysis is gained from movie's viewer comment. This comment could describe the rating more clearly. Random forest is used as the algorithm for gaining and training the sentiment. Sentiment could increasing probability of user preference. Difference from Suganeshwari [3], [4], and Govind [5], this research only focus on gaining top twenty result of movie recommendation and do the validation with comparing to other movie's viewer that has same preference.

There is another research done by Indah [1] which using cosine similarity to measure the closeness of each genre based on the alternate least square recommendation. Cosine similarity is decreasing error by RMSE parameter. This methodology could generate the high precision because filter the current recommendation with closeness calculation. Another research which using movie rating data was conducted by Weston [6]. Weston implemented a k-nearest neighbor and several matrix factorization. Stochastic gradient is used to update the model based on numerical result.

Jung-bin [7] is developed ALS with Hadoop Yarn platform. ALS algorithm is outperformed movie attribute content based (MACB) algorithm. The performance of 20 K dataset is same with 100 K dataset and the Spark standalone without Apache Yarn is giving a better result. Cervantes research [8] on the other side is comparing map reduce with GraphLab methodology for mapping data. GraphLab performed better when RMSE was considered, but there is an issue with shared memory. In this research, we consider to use map reduce because there is no major issue according to shared memory.

Contribution in this research is to provide a preliminary research and tuning solution for handling big data with specific algorithm which is ALS. Performance analytic is based on comparison of internal parameter of the ALS rather than testing with specific genre or with specific clustering technique which is controlling the rating.

3. Methodology

3.1 Recommendation System

Recommendation system is build based on the interest of user who is make rating on some preference based on the context [8]. Figure 1 show the process done in the recommendation system. There is three way to develop recommendation engine which are content based recommendation, collaborative recommendation, and hybrid approach. Content based preference is based on previous history rating of the user. Collaborative recommendation makes recommendation based on another user rating which has same preference. Hybrid approach is using content based as the performance comparator for collaborative filtering.

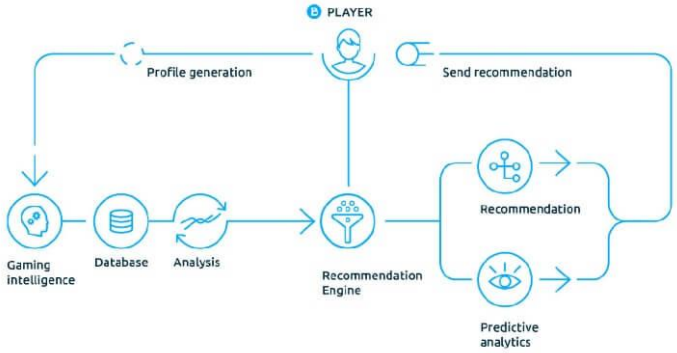

Figure 1: Recommendation Engine

3.2 Alternating Least Square

Alternating Least Squares is an algorithm built form collaborative filtering paradigm. Input of ASL is matrix factorization of user preference with a lot of variation. The algorithm computes matrix factorization R = U x V which contain movie ID and rating in this scenario. Figure 2 show matrix factorization in movie recommendation scenario. The lost function is defined in squared error, where the task of the algorithm is also minimize the root mean square error between predicted value of some movie rating based on rating of another

user. This is the matrix factorization formula for user, movie id, and rating. Alternating least square is has high cost on complexity which is $O(d^{-1}[Nr + (m + n)r^2]+r^3)$where m is the number of movie rating, and n is the number of user.

$$(\hat{U}, \hat{V}) = \underset{U,V}{argmin} \sum i,j \in R(_{r_{ij}} - v_i^T u_j)^2 \quad (1)$$

Item

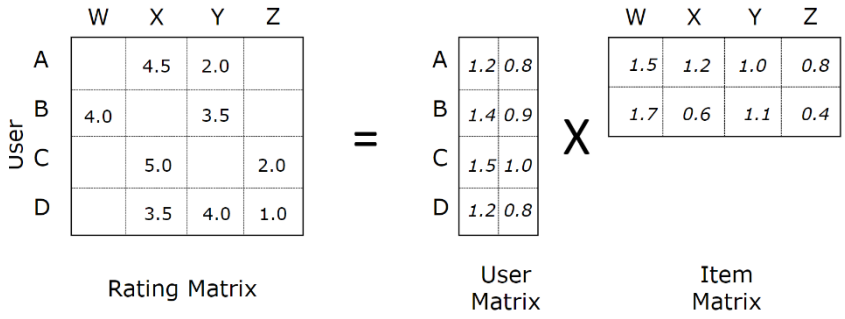

Figure 2: Matrix Factorization on Recommendation Engine

3.3 Alternating Least Square on Map Reduce and Spark

Map reduce uses four task of modelling process. Each item in dataset is labelled as (u,j,r), u is the user, j is the label of the movie, and r is corresponding rating from user to some movie. In the U update step, the matrix V formed first is used and sent to all cluster in RDD. RDD is used as parallel information storage which make the process of mapping and reduce become parallel. In RDD, context management is created, to make the process become parallel, iteration on model built could be done until the data become convergence. Training rating R is used to compute user matrix U, which is including input as lambda parameter λ to regularization, number of latent factors [8]. Figure 3 represents map reduce with ALS algorithm that implemented in Apache Spark. The process of V and U update is same but uses different matrix for input. Input in V update is item matrix and in U update is user matrix. After U and V update is successful doing mapping, there is a fitting process of training model into the ALS algorithm input. After ALS algorithm is doing the prediction, regression evaluator is doing summarization of all RMSE distributed in all user into one RMSE average value. The parallel ALS with Weighted Regularization is explained step by step in algorithm 1.

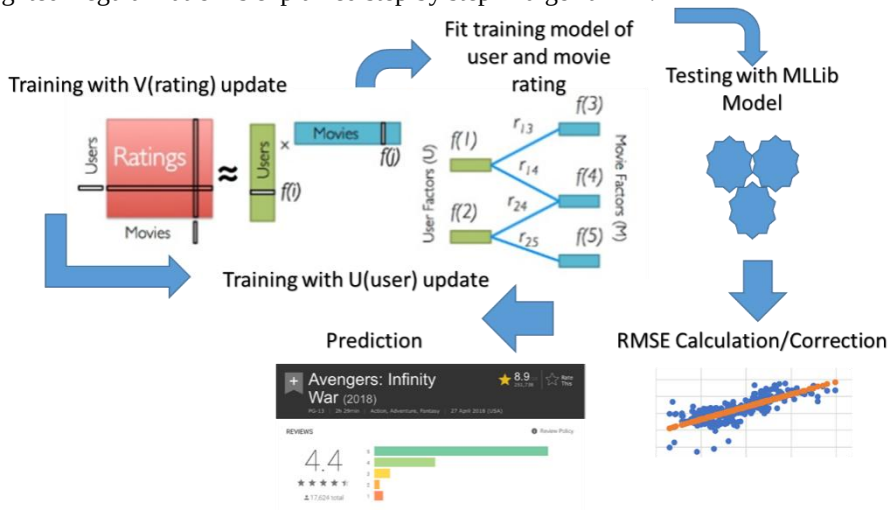

Figure 3: Matrix Factorization on Recommendation Engine

Algorithm 1 Alternating Least Square (ALS) algorithm

1: Initialize V with random values between 0 and 1 with specific formula
2: Hold V constants, and solve U by minimizing the objective function
3: Hold U constants, and solve M by minimizing the objective function

4: **Repeat** step 2 and 3 until objective function converge

Formula for step 1 is shown in equation 2, which is the general form of linear regression with lambda regularization (λ), to avoid over fits its penalty.

$$f(\hat{U}, \hat{V}) = \sum_{i,j \in I}(r_{ij} - u_i^T v_j)^2 + \lambda(\sum_i n_{ui} ||u_i||^2 + \sum_j n_{vj} ||v_j||^2) \quad (2)$$

3.4 Movie Lens Dataset

Movie Lens is free dataset for big data analytic especially in movie recommendation. Movie Lens is developed under Group Lens research group which is also developing recommender system. The dataset is used the collaboration of 100000 rating from 913 user and 1682 movie item. The data structure is user ID, movie ID, rating, and timestamp. Timestamp is not used because in this research did not do a time series analytic of correlation from periodically rating.

3.5 Map Reduce and Apache Spark Configuration

Setup of Map Reduce is using single machine with single worker node and data node. Hadoop version 2.7 used for admit rating data from HDFS. ALS was taken from Spark MLLib library. Apache Spark that used is version 2.4. Python version 2.7 is used for Pyspark library interfacing API and used spark submit functionality for submitting the movie recommendation.

4. Result and Evaluation

This result is gained from experimental setting of ALS parameter. Table 1 shows the parameter configuration from 4 scenario. In Table 1 for first experimental setting the maximum iteration as the latent factor have been increased from 5 to 10 and the implicit and explicit parameter are varied. From the result gained, RMSE is different from implicit and explicit parameter. This is because the implicit consider feedback as the positive and negative rating. In implicit there is no high or low ranking but can have both positive and negative rating. If the rating is from 1 to 5 then the positive rating give 2.5 to 5 but negative give below 2.5. This is could be happen because not every people is movie admirer, if there is a people with narrow movie experience could give a wrong rating judgement. The RMSE is higher in implicit parameter because of this scenario. Interesting fact is that maximum iteration is decreasing RMSE in explicit case but increasing RMSE in implicit case. In the term of execution time maximum iteration is give lower execution time because

Table 1: Experimental Setting 1 of ALS Parameter

Scenario	1	2	3	4
Implicit/Explicit	Explicit	Explicit	Implicit	Implicit
Max Iteration	5	10	5	10
Lambda	0.01	0.01	0.01	0.01
RMSE	0.712	0.695	3.047	3.051
Execution Time	141.73	96.98	120.66	100.32

Table 2 shows the modified parameter configuration which is to test the influence of increasing lambda with variation of explicit or implicit state and variation of max iteration. There is interesting fact that RMSE is error because lambda is increased in implicit state. This shows that implicit state need the lowest lambda that could be accepted by the system. Compared with Table 1, increasing lambda is increasing RMSE which is not very good. Lambda makes max iteration and RMSE is inversely proportional, because higher iteration give lower RMSE.

Table 2: Experimental Setting 2 of ALS Parameter

Scenario	1	2	3	4
Implicit/Explicit	Explicit	Explicit	Implicit	Implicit
Max Iteration	5	10	5	10
Lambda	0.05	0.05	0.05	0.05
RMSE	0.729	0.714	Error	Error
Execution Time	164.12	118.34	NaN	NaN

Table 3 shows the maximum iteration that could take by ALS. After 25 iteration, ALS submit become error. The lambda is setto minimum 0.01 and the RMSE hit the lowest value. In the term of RMSE, the best

state is explicit, the best max iteration is 20 and the best lambda is 0.01. If considering implicit parameter the best setting is rather different with max iteration of 5 and the best lambda is 0.01. Implicit parameter acts differently and the result of recommendation is also different compared in the Table 4.

Table 3: Experimental Setting 3 of ALS Parameter

Scenario	1	2
Implicit/Explicit	Explicit	Explicit
Max Iteration	20	25
Lambda	0.01	0.01
RMSE	0.6918	Error
Execution Time	120.62	NaN

Table 4 show the difference between implicit and explicit. The rating between implicit and explicit is different because explicit accept high and low rating between 1 – 5 and implicit act the rating like positive and negative, more positive is higher than 1 and more negative is lower than 1. There is interesting result that the same movie title recommendation between explicit and implicit is only two movie. Although the RMSE is better in explicit but the movie title is very different in implicit and explicit, so need further analysis that check the implicit pattern according to movie preference.

Table 4: Experimental Setting 3 of ALS Parameter

Explicit	Implicit
(u'Secrets & Lies (1996)', 4.884361267089844)	(u'Trainspotting (1996)', 1.2299593687057495)
(u'Close Shave, A (1995)', 4.847671985626221)	(u'Twelve Monkeys (1995)', 1.178810954093933)
(u'Pulp Fiction (1994)', 4.838813304901123)	(u'Fargo (1996)', 1.1516880989074707)
(u'Wrong Trousers, The (1993)', 4.793168067932129)	(u'Clockwork Orange, A (1971)', 1.1455484628677368)
(u'Chasing Amy (1997)', 4.790938377380371)	(u'Star Wars (1977)', 1.1166439056396484)
(u'As Good As It Gets (1997)', 4.782371520996094)	(u'Return of the Jedi (1983)', 1.1098368167877197)
(u'Casablanca (1942)', 4.773658752441406)	(u'Willy Wonka and the Chocolate Factory (1971)', 1.108720064163208)
(u'Godfather, The (1972)', 4.73976469039917)	(u'Four Weddings and a Funeral (1994)', 1.1074471473693848)
(u'Chasing Amy (1997)', 4.732972621917725)	(u'Much Ado About Nothing (1993)', 1.0832960605621338)
(u'Cinema Paradiso (1988)', 4.711854457855225)	(u'Star Trek: First Contact (1996)', 1.0815389156341553)
(u'Citizen Kane (1941)', 4.693789005279541)	(u'Toy Story (1995)', 1.075878620147705)
(u'Hoop Dreams (1994)', 4.67500638961792)	(u'Brazil (1985)', 1.0718410015106201)
(u'Boot, Das (1981)', 4.662115097045898)	(u'Heathers (1989)', 1.066953420639038)
(u'Sense and Sensibility (1995)', 4.656736373901367)	(u'Monty Python and the Holy Grail (1974)', 1.0606353282928467)
(u"Schindler's List (1993)", 4.648641586303711)	(u'Princess Bride, The (1987)', 1.0588643550872803)
(u'Good Will Hunting (1997)', 4.641825199127197)	(u'Blade Runner (1982)', 1.0566431283950806)
(u'Star Wars (1977)', 4.636653900146484)	(u'Clerks (1994)', 1.0473716259002686)
(u'Nightmare Before Christmas, The (1993)', 4.606769561767578)	(u'Pulp Fiction (1994)', 1.0448020696640015)
(u'Dr. Strangelove or: How I Learned to Stop Worrying and Love the Bomb (1963)', 4.5991902351379395)	(u'Nightmare Before Christmas, The (1993)', 1.0376540422439575)
(u'Full Monty, The (1997)', 4.569430828094482)	(u'Independence Day (ID4) (1996)', 1.0315202474594116)

5. Conclusion

This research can concluded some conclusions after test the ALS parameter. Experiment setup was run over Movie Lens dataset and Apache Spark environment with regaining data from HDFS. RMSE between implicit and explicit parameter were different. Explicit parameter give better result than implicit but the rating suggestion must be rearranged and researched later because has very different point of view. Best maximum iteration parameter is 5 for implicit and 20 for explicit. Best lambda parameter is 0.01 for overall. The performance of RDD was tested only on execution time which are along 90 – 140 second for 10000 dataset. From this research need further analysis to make relation between movie and suggestion becoming more related. Further analysis could be done by double check in the suggestion and cross validation between implicit and explicit parameter.

References

[1] I. SurvyanaWahyudi, A. Affandi, and M. Hariadi, "Recommender engine using cosine similarity based on alternating least square-weight regularization," in *2017 15th International Conference on Quality in Research (QiR) : International Symposium on Electrical and Computer Engineering*, Nusa Dua, Jul. 2017, pp. 256–261, doi: 10.1109/QIR.2017.8168492.

[2] I. S. Wahyudi, "Big data analytic untuk pembuatan rekomendasi koleksi film personal menggunakan Mlib. Apache Spark," *Berk. Ilmu Perpust. Dan Inf.*, vol. 14, no. 1, p. 11, Jun. 2018, doi: 10.22146/bip.32208.

[3] "Gopalswamy and Mohamed - Time Adaptive Collaborative Filtering for Movie Re.pdf." .

[4] S. G., S. I. S.P., and G. Li, "Lazy collaborative filtering with dynamic neighborhoods," *Inf. Discov. Deliv.*, vol. 46, no. 2, pp. 95–109, May 2018, doi: 10.1108/IDD-02-2018-0007.

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

[6] W. Barger and S. Rudy, "Recommender Systems for Movie Rating Data," p. 8.

[7] L. Barolli, F.-Y. Leu, T. Enokido, and H.-C. Chen, Eds., *Advances on Broadband and Wireless Computing, Communication and Applications: Proceedings of the 13th International Conference on Broadband and Wireless Computing, Communication and Applications (BWCCA-2018)*, vol. 25. Cham: Springer International Publishing, 2019.

[8] E. V. V. Cervantes, L. V. C. Quispe, and J. E. O. Luna, "Performance of Alternating Least Squares in a distributed approach using GraphLab and MapReduce," p. 7.

Author Profile

Yosua Alvin Adi Soetrisno was a lecturer from Diponegoro University. Yosua was born on Semarang, 13th October 1990. Yosua got his first bachelor degree from Diponegoro University majoring computer networking. Yosua got his master degree from Gadjah Mada University majoring software engineering. Yosua has a brief experience in programming of computer services, especially in IoT web-based environment.

Enda Wista Sinuraya was a lecturer from Diponegoro University. Enda was born on Sumatera Utara, 21th January 1980. Enda got his first bachelor degree from Sumatera Utara University majoring control system. Enda got his master degree from University of Indonesia majoring computer engineering. Enda has a brief experience in microcontroller and IoT services design.

Eko Handoyo was a lecturer from Diponegoro University. Eko was born on Salatiga, 8th June 1975. Eko got his first bachelor degree from Brawijaya University majoring information technology. Eko got his master degree from Bandung Institute of Technology majoring computer engineering. Eko has a brief experience in algorithm design and computer system.

Ajub Ajulian was a lecturer from Diponegoro University. Ajub was born on Semarang, 19th July 1971. Ajub got his first bachelor degree from Diponegoro University majoring telecommunication enginering. Ajub got his master degree from Gadjah Mada Univeristy majoring telecommunication and traffic engineering. Ajub has a brief experience in traffic planning and simulation.

Karnoto was a senior lecturer from Diponegoro University. Karnoto was born on Purworejo, 9th July 1969. Karnoto got his first bachelor degree from Diponegoro University majoring power system. Karnoto got his master degree from Gadjah Mada University majoring power system planning and analysis. Karnoto has a brief experience in electrical planning and installment on building.

Sudjadi was a senior lecturer from Diponegoro University. Sudjadi was born on Salatiga, 19th June 1959. Sudjadi got his first bachelor degree from Bandung Institue of Technology majoring electrical engineering. Sudjadi got his master degree from Gadjah Mada University majoring microcontroller and control system. Sudjadi has a brief experience in microprocessor and computer system design.

Tejo Sukmadi was a senior lecturer and associate professor from Diponegoro University. Tejo was born on Solo, 17th November 1961. Tejo got his first bachelor degree from Gadjah Mada University majoring electrical power. Tejoalso got his master degree from Gadjah Mada University majoring electrical machinery. Tejo has a brief view in electrical power and has a strong experience in electrical machinery and transformator.

Imam Santoso was a senior lecturer from Diponegoro University. Imam was born on 3rd December 1970. Imam got his first bachelor degree from Diponegoro University majoring telecommunication system. Imam got his master degree from Gadjah Mada University majoring telecommunication engineering especially on machine learning and image processing.