DOI: http://doi.org/10.32792/utq.jceps.09.01.13

Apply Hybrid Recommender System Using Genetic Algorithms And Singular Value Decomposition (SVD)

***Jawad S. Hameed*[1]*, Nasshat J. Mohammed*[1]**

[1]*Middle Technical University, Technical College Of Management, Baghdad, Iraq.*

Abstract:

Due to the increasing development in using modern technology for different purposes, recommendation systems have become essential in providing solutions to all users. Recommendation systems can be used for various applications or on the internet to purchase products or to surf the web for scholarly articles, to watch videos and other activities. Many statistical and mathematical suggestions have been made to make different systems capable of giving accurate recommendations that can be of benefit to users in various fields. One of these suggestions is Singular Value Decomposition (SVD) which is the most popular method in previous research. In addition to that, (SVD) has two important factors in its performance, which are, (K , λ) and choosing these two factors plays an important role in the accuracy of recommendation systems. Choosing these two factors was a heavily discussed topic in previous academic research. In this paper, the use of artificial intelligence method Genetic Algorithm (GA) to design a hybrid system with the Singular Value Decomposition (SVD) to specify the best values of (K , λ), and hence acquiring an accurate recommendation system with better speed. After several experiments, the results indicated that the hybrid system better system according to time. The hybrid system (GA) is also better than the system uses (SVD) in accuracy and time.

Keywords: Recommender systems, Genetic Algorithm, Singular Value Decomposition.

1.1 INTRODUCTION

Recommender systems are defined in [1] as software systems in which “people provide recommendations as inputs, which the system then aggregates and directs to appropriate recipients.” Today, the term includes a wider spectrum of systems describing any system that provides individualization of the recommendation results and leads to a procedure that helps users in a personalized way to interesting or useful objects in a large space of possible options. Recommender systems form an important research area because of the abundance of their potential practical applications Recommender systems apply data analysis techniques to the problem of helping customers to find which products they would like to purchase especially on the internet. These systems are rapidly becoming a crucial tool in E-commerce on the Web. The tremendous growth of customers and products poses two main challenges for recommender systems. The first challenge is to improve the quality of the recommendations for the customers. Making good recommendations, increases the customers desire to purchase products, whereas making bad recommendations may result losing customers. Another challenge is to improve the scalability of the recommendation algorithms. These algorithms are able to respond tens of millions of recommendation requests in real-time. In order to make a system scalable, the response time for the requests should be reduced. However, if the algorithm spends less time for recommendation, the quality of the recommendation decreases. Actually, from this perspective these two challenges are in conflict. For this reason, it is important to consider both of them simultaneously for the proposed solutions. Every recommendation system follows a specific process while making recommendations. Systems use the users 'profiles and the information about items or products as the inputs and produce recommendations. In other words, a recommendation system consists of background data. The information that the system has before the recommendation process begins, input data, the information that user must communicate to the system in order to generate a recommendation, and an algorithm that combines background and input data to arrive at its recommendations. Recommendation techniques can be grouped into five as collaborative, content-based, demographic, utility-based, and knowledge-based [2]. Moreover, some hybrid solutions at this point, SVD has an important property that makes it interesting for recommender systems SVD provides the best low-rank linear approximation of the original matrix and the low-rank approximation of the original matrix is better than the original matrix itself [3, 4]. Filtering out of the small singular values can be introduced as removing noise data in the matrix. [4-6].

2. SINGULAR VALUE DECOMPOSITION (SVD) [6, 7]

SVD is a well-known matrix factorization technique that factors a $m \times r$ matrix R into three matrices as the following:

$$R = U \times S \times V'$$

Where, U and V are two orthogonal matrices of size $m \times r$ and $n \times r$ respectively; r is the rank of the matrix R. S is a diagonal matrix of size $r \times r$ having all singular values of matrix R as its diagonal entries all the entries of matrix S are positive and stored in decreasing order of their magnitude. The matrices obtained by performing SVD are particularly useful for our application because of the property that SVD provides the best lower rank approximations of the original matrix R, in terms of Fresenius norm. It is possible to reduce the $r ´ r$ matrix S to have only k largest diagonal values to obtain a matrix S_k, $k < r$. If the matrices U and V are reduced accordingly, then the reconstructed matrix $R_k = U_k S_k V_k^u$ is the closest rank-k matrix to R. In other words, $R_k$minimizes the Frobenius norm $\|R - R_k\|$ over all rank-k matrices. We use SVD in recommender systems to perform two different tasks: First, use SVD to capture latent relationships between customers and products that allow us to compute the predicted likeliness of a certain product by a customer. Second, we use SVD to produce a *low-dimensional* representation of the original customer-product space and then compute neighborhood in the reduced space. Then used that to generate a list of *top-N* product recommendations for customers. The following is a description of our experiments.

2.1 Recommendation Using SVD

The goal of collaborative filtering –based (CF) recommendation algorithms is to suggest new products or to predict the utility of a product for a customer, based on the customer's previous behavior and other similar customers 'opinions. However, these systems have some problems like sparsity, scalability, and synonymy. The weakness of (CF) algorithms for large, sparse databases led the researchers to alternative ways. In order to remove noise data from a large and sparse database, some dimensionality reduction techniques are proposed[3-5] .Alternatrnatin Least Squares(ALS), which is a dimensionality reduction technique that used in information retrieval (IR) widely used technique to reduce the dimensionality of user-item ratings matrix. ALS, which uses singular value decomposition (SVD) as its underlying dimension reduction algorithm, maps nicely into the collaborative filtering recommender algorithm challenge[3]. SVD-based recommendation algorithms produce high quality recommendations, but has to undergo computationally very expensive matrix factorization steps [3].

2.2 Sensitivity of Number of Dimensions k ,λ

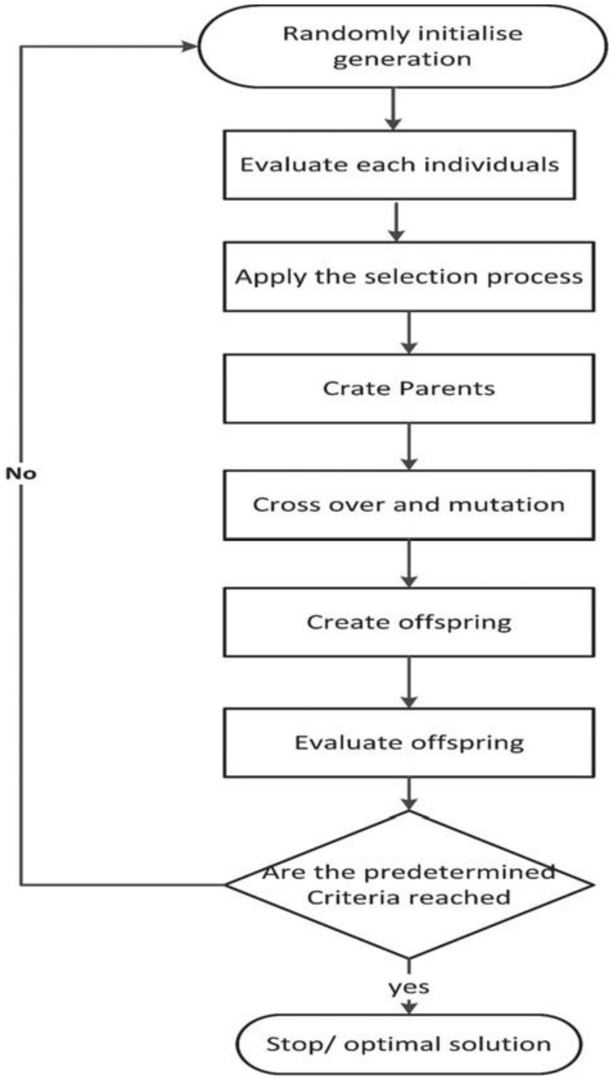
The optimal choice of the value k,λ is critical to high quality prediction generation. We are interested in a value of k,λ that is large enough to capture all the important structures in the matrix yet small enough to avoid overfitting errors. We experimentally find a good value of k,λ by trying several different values.

Where researchers have concluded [3,6,8-10] and others that the optimal choice of k,λ in algorithms **Singular Value Decomposition** is critical to high quality prediction generation They try to get the best values in the course of implementing this algorithm by using the principle of attempt and error for values that reduce the error level .

3. Genetic Algorithms

Genetic algorithms (GA) are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string structures with a structured yet randomized information exchange to form a search algorithm with some of the innovative flair of human search[8]. The flow chart of this algorithm is shown in figure *(1.1).*

First of all. An initial population of potential solutions (individuals) is generated randomly. A selection procedure based on a fitness function enables to choose the individuals candidate for reproduction. The reproduction consists in recombining two individuals by the crossover operator, possibly followed by a mutation of the offspring. Therefore, from the initial population a new generation is obtained . From this new generation, a second new generation is produced by the same process and so on. The stop criterion is normally based on the number of generations.



3.1 Encoding

The heart of every GA is an solution described by its chromosomes which will be later passed down to their offspring. It is common to use a binary string to represent each solution in GA where each bit indicates whether the solution possess that specific characteristic or not and to use numbers to indicate the strength of a feature in an solution[9, 10]. Using numbers have an advantage over using simple bits to indicate presence of a feature because it allows us to define an solution as stronger than others by using more than just presence of the feature. New features can easily be added by introducing new property and assigning a value, and using zero to indicate the solution does not have that feature. we represent a possible solution by describing the strengths of each feature. There are 6 features available and the higher a value, the stronger that feature is (491347).

3.2 Fitness function

The fitness function is a function to rate a given individual and assign a score to it. An individual that shows a lot of promise to be closer to ideal solution we are looking for will have a higher score than others. The fitness function is very critical in GA because they decide whether a solution is good or not and a poorly designed function will lead to less than ideal solutions in the end. Fitness algorithms can vary from being utility function or simple mathematical functions such as sum and average are applied [11, 12]. Utility functions measures the usefulness of something to a user rather than simply plugging into an equation.

3.3 Selection and Reproduction

In GA, the fitness function is applied to each and every solution. The assigned fitness value is used to select an individual probabilistically where individuals with higher fitness value have higher chance of being selected [13, 14] in a process called tournament selection. The algorithms used to select individuals also give individuals with lower fitness score a fighting chance to ensure their characteristics are passed on as well to create diversity. If we have two solutions which we consider parent solutions that are described using two vectors (4 9 1 3 4 8) and (4 9 1 3 4 8), we can make a child of those solutions such as (4 9 1 3 4 8). In this example, the new solution inherits characteristics of both solutions. A particular characteristic from one parent is probabilistically selected over the other one.

3.4 Mutation

Earlier in reproduction, we saw that offspring takes it value from the parents. However, this is not very useful to us when finding the optimal solution. Therefore mutation is the primary way of obtain new solutions to work with in GA. All the solutions in GA are provided with the same probability of mutating with no favoritism . The mutation changes a value in the offspring to be something different than from both of its parents, introducing new possibilities. For example, if we have two solutions (4 9 1 3 4 8) and (4 9 1 3 4 8) which we consider to be the parents, child can be [4 9 1 3 4 8]. In this case, the child not only inherited characteristic of the parents, but the third characteristic is mutated to be different than both parents. This approach has been met with criticism because they exists a chance good solutions will become bad. The solutions that are bad will become worse with the same mutation but the act of assigning the mutation to all also ends up degrading the quality of the good solutions[11,12]. An approach to this includes reducing the mutation done to favorable solutions with higher fitness scores.

4. DATASET

Use in research and data by researchers Bi, Qu [15] Where the data is made up of fields (User ID, Item ID and Rating) . Where the number of users (650), and the number of items (660) a record of 21450 records.

5. Methods

A hybrid system design consists of algorithm (GA) to find values (K, lambda) and (SVD) Figure (1.2) the flow chart for GA implementation with SVD for hybrid system operation to obtain the best values (K, λ) to increase the performance of the recommendation system.

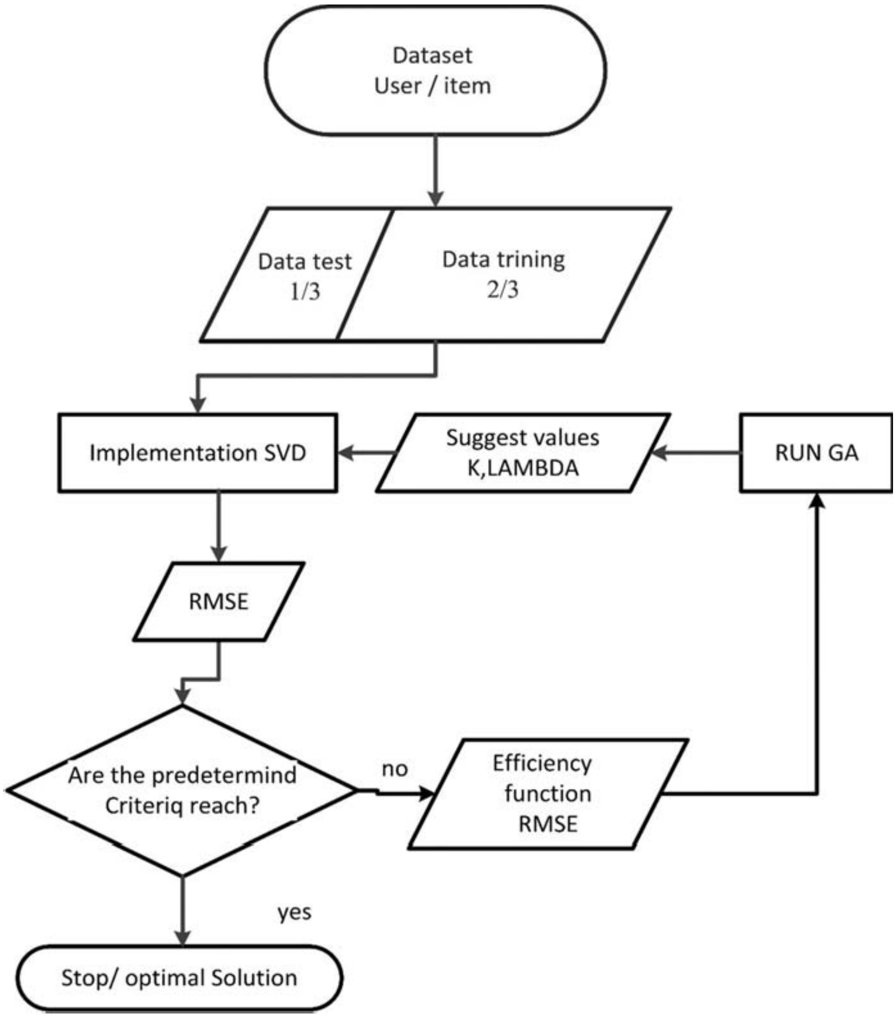
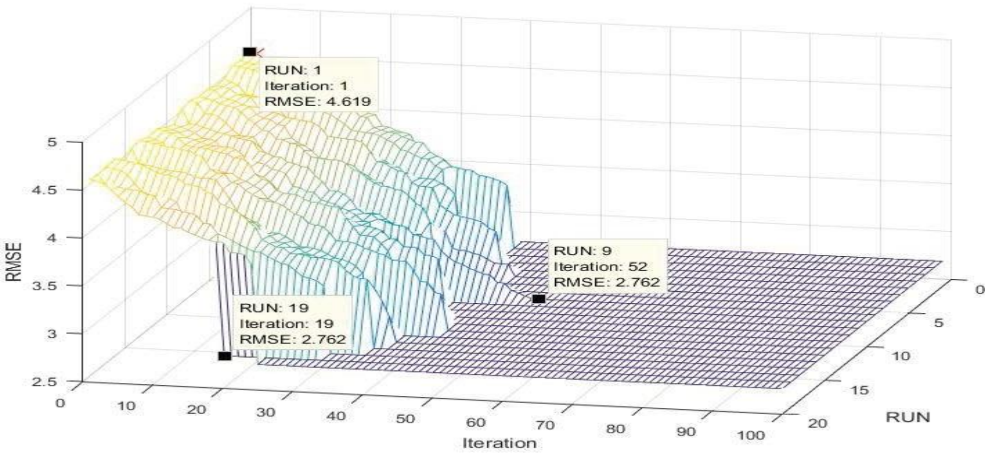


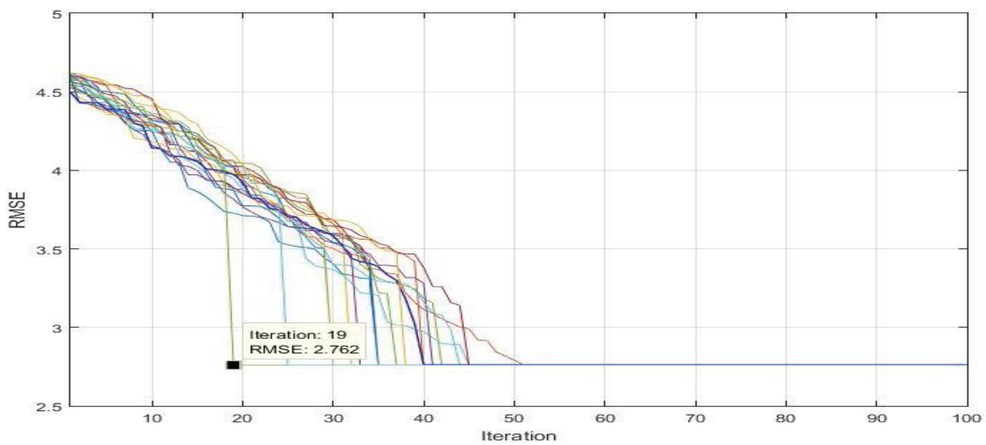
Figure no (1 .2) Outline for the implementation of a hybrid algorithm for the recommendation system

5.1 Results and discussion

Experiment executed (20 times) with 100 iteration at one time, where the data were divided into training data and test data. Usually measured using the statistical criteria such as Root Mean Squared Errors (RMSE). Which is the difference between the predicted and actual outputs, the figure demonstrates the results of the experiments. The figures (1.3) and (1.4) they represent the performance of the system to find the best values (RMSE). where the value indicates K=3 , $\lambda = 1$,best result for less root mean square error Stability this value at iteration (52) for the end of the experiment. Where is prove the best value arrive at hybrid system (RMSE =2.7620) .Which shows that the hybrid system has improved in speed. Moreover, the accuracy of the implementation of the recommendation. On one of the methods of artificial intelligence (GA). In finding values$(K,\ \lambda)$. His method proved that the genetic algorithm was used (GA) has speeded up finding optimal values to achieve better recommendation accuracy. This is consistent with the required benefit of the recommendation system in producing high-resolution recommendations more quickly. This approach does not require the creation of all values $(K,\ \lambda)$.It only find the values that are Made the most accurate recommendations and less than singular value decomposition (SVD), the table (1,1) shows represents performance of the methods.



Fieger no (1.3) The best value (RMSE) was obtained for the hybrid system of recommendation



Fieger no (1.4) The value of RMSE for different values from (K ,Lambdai) when applying SVD-GA hybrid recommendation system

Table no 1.1Represents the performance of the methods used in the recommendation systems

Methods	λ	K	RMSE	TIME (hours)
SVD	12	٣	3.7927	72.26
SVD – GA	1	٣	2.7620	24.35

Choose your value(K, λ) for the method (SVD) According to research published by researchers Bi, Qu[15] And value (K, λ) to find by using (SVD-GA) .

6. Conclusion

The proper choice of values (k, Lambda) has an important role in the algorithm SVD Successful selection of these values results in improved recommendation accuracy. Artificial Intelligence algorithms used to help speed up access to the best values achieve the lowest error rate and thus obtain accuracy in the recommendations suggested. In general, the use of artificial intelligence methods with statistical and mathematical methods in the construction of the recommendation systems leads to the production of hybrid systems, which can give better performance indicators through a more precise and standard recommendation.

References:

[1] P. Resnick and H. R. Varian, "Recommender systems," *Communications of the ACM*, vol. 40, pp. 56-58, 1997.
[2] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, pp. 331-370, 2002.
[3] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Incremental singular value decomposition algorithms for highly scalable recommender systems," in *Fifth International Conference on Computer and Information Science*, 2002, pp. 27-28.
[4] M. W. Berry, S. T. Dumais, and G. W. O'Brien, "Using linear algebra for intelligent information retrieval," *SIAM review*, vol. 37, pp. 573-595, 1995.
[5] D. Gupta, M. Digiovanni, H. Narita, and K. Goldberg, "Jester 2.0 (poster abstract): evaluation of an new linear time collaborative filtering algorithm," in *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 291-292.
[6] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, "Application of dimensionality reduction in recommender system-a case study," Minnesota Univ Minneapolis Dept of Computer Science2000.
[7] Y. Koren, R. Bell, and C. Volinsky, "Matrix factorization techniques for recommender systems," *Computer*, vol. 42, 2009.
[8] D. E. Goldberg, "Genetic algorithms in search, optimization, and machine learning/David E," *Goldberg.–[USA]: Addison-Wesley*, 1989.
[9] M. Mitchell, *An introduction to genetic algorithms*: MIT press, 1998.
[10] N. F. McPhee, R. Poli, and W. B. Langdon, "Field guide to genetic programming," 2008.
[11] W. Fan, E. A. Fox, P. Pathak, and H. Wu, "The effects of fitness functions on genetic programming- based ranking discovery for Web search," *Journal of the Association for Information Science and Technology*, vol. 55, pp. 628-636, 2004.
[12] M. Srinivas and L. M. Patnaik, "Genetic algorithms: A survey," *computer*, vol. 27, pp. 17-26, 1994.
[13] J. T. Alander, "Indexed bibliography of most active researchers of genetic algorithms," *Artificial intelligence*, vol. 960, p. 961, 1993.
[14] R. Poli, W. B. Langdon, N. F. McPhee, and J. R. Koza, *A field guide to genetic programming*: Lulu. com, 2008.
[15] X. Bi, A. Qu, J. Wang, and X. Shen, "A Group-Specific Recommender System," *Journal of the American Statistical Association*, vol. 112, pp. 1344-1353, 2017.