Improving the Accuracy of Recommender Systems Through Annealing

Thesis submitted in partial fulfillment of the requirements for the award of degree of

Master of Engineering

in

Computer Science and Engineering

Submitted By

Shefali Arora

(Roll No. 801432026)

Under the supervision of:

Dr. Shivani Goel

Assistant Professor

COMPUTER SCIENCE AND ENGINEERING DEPARTMENT

THAPAR UNIVERSITY

PATIALA – 147004

June 2016

CERTIFICATE

I hereby certify that the work which is being presented in the thesis entitled, "***Improving the Accuracy of Recommender Systems Through Annealing***," in partial fulfillment of the requirements for the award of degree of Master of Engineering in ***Computer Science and Engineering*** submitted in Computer Science and Engineering Department of Thapar University, Patiala is an authentic record of my own work carried out under the supervision of ***Dr. Shivani Goel*** and refers other researchers' work duly listed in the reference section.

The matter presented in the thesis has not been submitted for award of any other degree of this or any other University.

Signature:

Shefali Arora

Shefali Arora

801432026

ME CSE

This is to certify that the above statement made by the candidate is correct and true to the best of my knowledge.

S Shivani Goel

Dr. Shivani Goel

Assistant Professor,

CSED

Countersigned by

Dr. Maninder Singh

Head

Computer Science and Engineering Department

Thapar University

Patiala

Dr. S.S. Bhatia

Dean(Academic Affairs)

Thapar University

Patiala

ACKNOWLEDGEMENT

I would like to express my gratitude towards my guide, Dr. Shivani Goel, Computer Science & Engineering Department, Thapar University, Patiala, who has guided me throughout the journey. She has always been supportive, guiding me to the right track in the process of writing this thesis report. It is all because of her that I was able to delve into a new field and implement the aspects of my research work in a stipulated amount of time. She cooperated and directed me well to carry forward this task.

I am also thankful to **Dr. S. S. Bhatia**, Dean of Academic Affairs, **Dr. Maninder Singh**, Head of Computer Science & Engineering Department and **Dr. Ashutosh Mishra**, P.G. Coordinator, for providing all the facilities and the right environment for learning.

I would also like to thank my colleagues for extending all kind of help and cooperation during thesis.

Most importantly, I would like to thank my parents, my family members and the almighty for giving me the strength to carry out this research work. This work would not have been possible without their support and for being with me throughout my journey of exploration.

Shefali Arora

801432026

ABSTRACT

Collaborative filtering (CF) is the most popular approach in recommender systems (RS). It makes use of a user item rating matrix and recommends on the basis of preferences and tastes of other users. It faces a number of issues like cold start problem, shilling attack problem and sparse matrix problems. Matrix Factorization (MF) is an efficient approach to get rid of sparse matrix problems. It is a highly reliable and robust technique that helps to predict those ratings to a user for an item that are not yet rated by him. This is done by mapping items and users to a latent space based on a given number of latent features. Minimization in MF is done by either Alternating Least Squares (ALS) method or Stochastic Gradient Descent (SGD) technique.

In this thesis, SGD is used to perform minimization on the matrix factorization function using the concept of singular value decomposition (SVD) to fill in missing entries in the sparse user item rating matrix. Using this base approach, a factor known as learning rate (η) is varied to determine the accuracy and convergence rate of recommender systems. This is done by using simulated annealing, which decrements the value of learning rate in each iteration and provides an optimal solution to minimize error in the system.

In this thesis, five simulated annealing schedules, along with a new proposed annealing schedule have been chosen to discuss the effect of learning rate on the accuracy of a movie recommender system. These annealing schedules are- exponential annealing, inverse scaling logarithmic cooling, linear multiplicative cooling and quadratic multiplicative cooling. Our proposed annealing schedule is named as Square Root Cooling (SRA). The experimental results on Movielens dataset prove that by employing exponential annealing schedule as the learning rate, minimum mean absolute error can be attained for the system at a lower value of learning rate. For higher learning rate values, SRA works the best. Apache Mahout 0.9 is chosen as the platform for the research.

TABLE OF CONTENTS

LIST OF FIGURES

Chapter 1
Introduction

1.1 Recommender System

These days, consumers are overwhelmed with a huge number of choices. Retailers and content providers provide a good selection of products, which are meant to satisfy diverse needs and tastes of customers. In order to gauge user's interests and gain their loyalty, retailers have started making use of recommender systems, which analyze patterns of a user's interest in products so that provide personalized recommendations can be provided according to his taste[1]. E-commerce rulers like **Amazon.com** [2] and **Netflix** are popular websites which use recommender systems. These systems could help to recommend movies, music and books to users. Customers provide their choice of movies as well as the satisfaction level for movies already watched. Thus a huge amount of data is available and is used to recommend more movies of his interest to a particular customer.

Recommender systems could be based on user behavior i.e. a user's ratings in the past provide further recommendations. **Tapestry** was the first such recommender system. Recommendations can also be provided on the basis of demographic features or attributes, like genre in case of a movie. **Pandora.com** is a successful implementation of such a concept where the musical preferences of a listener are captured and music is recommended based on its characteristic features. Recommendations can also be item-based i.e. relationships are calculated among items which are indirectly used to generate recommendations for the user.

1.2 Taxonomy of Recommender Systems

The following features define the basic structure of a recommender system:

1.2.1 Domain

Recommender systems could be used to recommend:

- News
- Products by vendors
- Music playlists or sequences

- Matchmaking
- New items or re-recommend old items(groceries, music)

1.2.2 Level of Personalization

The context of recommendation depends on what the user likes to do . It may be shopping, listening to music or hanging around with other people. For example, Pandora is a music recommendation system which plays songs continuously based on the taste of the user. Wine.com recommends wines based on the opinions of experts.
The personalization level in a recommender could be of the following kinds:

- **Generic/Non Personalized**: Every user receives the same recommendations from the system and there is no rating by fellow users. Landsend.com is an example of a non-personalized recommender which provides generic information to users.
- **Demographic**: These systems match a target group(based on gender ,age etc.) to provide recommendations. For example, recommendations at Brooks Brothers website is on the basis of the demographic attributes of a user.
- **Ephemeral**: These systems provide recommendations based on the current activity of the user. For example, a user who surfs a product on Amazon.com without creating an account, starts getting recommendations similar to that item.
- **Persistent**: Such systems are personalized and look at the long term interests of the users. For example, a user with an account on Amazon.com or CDNow.com would get recommendations based on all his previous activities on these websites.

1.2.3 Privacy and Trustworthiness

The privacy of a recommender system is defined by the following:

- The amount of transparency in the system
- The amount of vulnerability to external manipulation
- The amount of bias in the system. For example, some recommender systems do not sell anything that does not have a high markup.

1.2.4 Interface

The interface of a recommender system is judged by how it deals with taking input and generating outputs. Input could be explicit (a person rating a product) or implicit (looking at a particular product) whereas the output obtained from recommenders constitutes predictions, score, search lists etc.

1.2.5 Recommendation Techniques [3]

As recommender systems have emerged out of the need of information retrieval and persistent usage, it has further given rise to information filtering. In order to implement its core function, they should compare items and predict what is worth rating to the user. The following sections identify different techniques implemented in recommenders to generate predictions.

1.2.5.1 Collaborative filtering based Recommender systems

Collaborative filtering (CF) is an algorithm based on the preferences of users and comparison with tastes of other users. Thus the basic task is to predict recommendations over pairs of users and items[3]. The similarity between two users is calculated based on previous ratings of the users. Thus CF is a popular approach in recommender systems and works on the basis of correlation between a given set of users. Figure 1 shows collaborative filtering in a food recommender system.

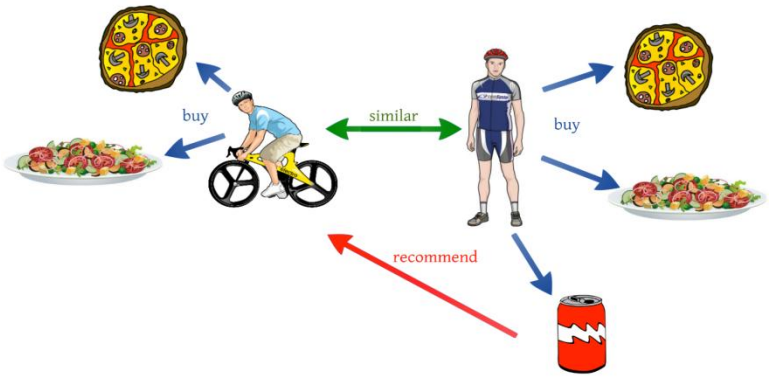


Figure 1. Collaborative filtering to find similar users in a food recommender [4]

This method finds similarities with the help of a user-item rating matrix, which consists of data given by users corresponding to a set of items.

CF can be divided into two categories:

- Memory-based collaborative filtering techniques
- Model-based collaborative filtering techniques

Memory-based collaborative filtering techniques focus on a user's neighborhood to find people who hold interests similar to a particular user. This approach makes use of the nearest neighbor method to find neighbors of a particular user and predict a new item for him. Thus similarity is calculated to find the correlation between users and items, as CF can be user based or item based [5]. Various similarity measures are used to map a correlation value. The similarity measure could return a value between -1 to 1, a value 0 indication least similarity and 1 indicating that the users or items are exactly alike. Some of the similarity measures are:

- **Euclidean Distance Similarity**- It measures the similarity by finding the length of paths connecting two objects. It is useful when data is continuous.

$$E(x,y) = \sqrt{\sum_{i=0}^{n}(x_i - y_i)^2} \quad \text{..... (1)}$$

- **Pearson Correlation Similarity**- It calculates a similarity value between -1 to 1 by finding how large a numerical attribute is in comparison with other values that are high in magnitude. If this tendency is high then value would be close to 1. Given scores x and y of N objects, Pearson correlation coefficient is given as:

$$Pearson(x,y) = \frac{\sum xy - \frac{\sum x \sum y}{N}}{\sqrt{(\sum x^2 - \frac{(\sum x)^2}{N})(\sum y^2 - \frac{(\sum y)^2}{N})}} \quad \text{.....(2)}$$

`

- **Vector Cosine-Based Similarity**- The dot product between the considered items gives the similarity between them. If the cosine of angle is 90^o, then the similarity would be 0.Similarly it would be 1 if items are on the same plane(cosine for an angle 0) .For two items i and j, Vector cosine similarity is given by

$$w_{i,j} = \cos\left(\vec{i}, \vec{j}\right) = \frac{\vec{i} \bullet \vec{j}}{\|\vec{i}\| * \|\vec{j}\|},$$

.....(3)

Thus nearest-neighbors approach is simple and efficient to provide personalized recommendations. User based CF is memory based as it involves a large number of users on e-commerce websites and works on a user item rating matrix.

Model-based collaborative filtering techniques help to make predictions with the help of learned models and patterns based on training and testing data. The complete dataset might be taken as the training set, or it might be split into two parts. Some examples of model based CF approaches are SVD (for numerical ratings) [6] clustering models and classification algorithms. Thus, these models work on the concept of training data and learning to make predictions. The outputs are vectors in a n-dimensional space, on the basis of which similarity is calculated and predictions are made. Item based CF is an example of model based CF approach.

SVD is a powerful model based approach which is used for dimensionality reduction. It is a realization of matrix factorization and helps to provide a low dimensional latent space with latent features. These features are termed as concepts and the strength of each concept is computed. The basic idea is to split user item rating matrix R (q items, p features) into matrices : U(q items, r concepts) ,ʎ (strength of concepts matrix which is singular and always sorted in decreasing order) and V(p features, r concepts). Therefore,

$$R = U\Lambda V^T \qquad \text{..... (4)}$$

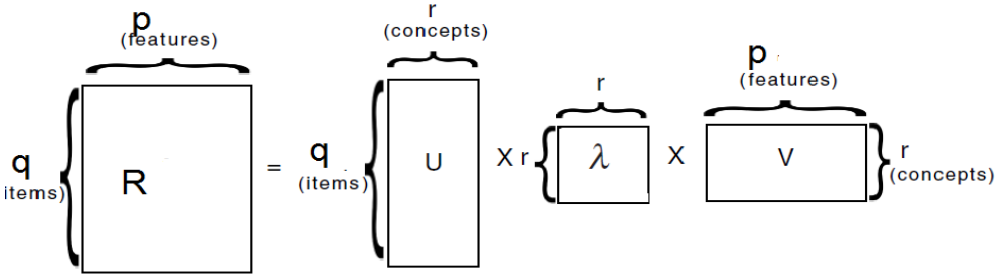


Figure 2. Illustrating SVD in recommender systems

To compute SVD of matrix R, matrices RR^T and R^TR are considered. The columns of U are obtained by calculating the eigenvectors of RR^T and those of V are eigenvectors of R^TR. The values of ʎ are positive square roots of these eigenvectors. Thus SVD is calculated by calculating eigenvectors RR^T and R^TR and further the eigenvectors of the resulting matrices.

Thus SVD can be used to describe latent relations by associating vectors to items and users. Vectors associated to items describe how much latent feature an item has. Those associated with users describe how much interest a user has in each item [7].

1.2.5.2 Content Based Filtering

In content-based filtering, those items are recommended to the user which are similar to the items us The similarity is calculated on the basis of features associated with the item. Figure 3 represents content based filtering on music data[8].

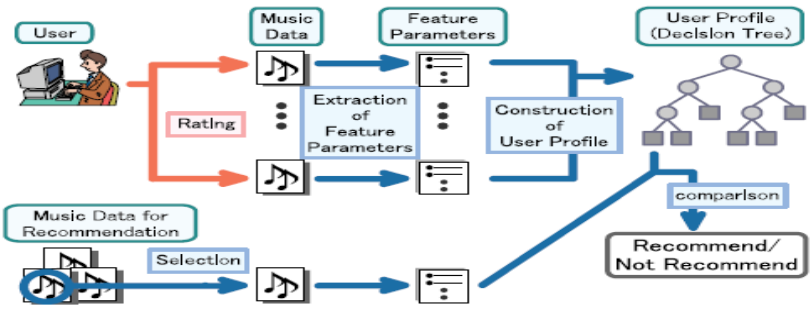


Figure 3. Content-based Music Recommender System [8]

The user rates several songs and the system extracts the features of data already rated. A user profile is created and his preference is represented based on these parameters extracted. The characteristics of target songs are thus compared with the preferences of the user and new recommendations of songs are made, which would be similar to the type of music liked by the user in the past.

1.2.5.3 Demographic Recommender Systems

These systems work on the demographic profile of a user. For example, recommender systems could provide recommendations according to the age of a user. A website may recommend according to the country of a user. These systems are making a mark in the marketing world.

1.2.5.4 Knowledge-based Recommender Systems

These systems are based on a specific domain and if an item fits in that domain, then items would be recommended to a user [9]. The similarity measure is based on how much a user demand matches the generated predictions. The knowledge base could be implemented by gathering user requirements on a particular product and questioning the user. On consulting the knowledge base, the required products are recommended.

1.2.5.5 Community-based Recommender systems

Community based recommenders recommend items on the basis of the tastes of a user's network of friends. It is observed that people tend to prefer the opinions of their friends and thus this combination, along with social networks is giving rise to such systems [10].

Such systems require information about users' relations and the preferences of such connected people. Based o the ratings given by their friends, new recommendations are generated. It is observed that such systems provide better recommendations as compared to the traditional CF techniques.

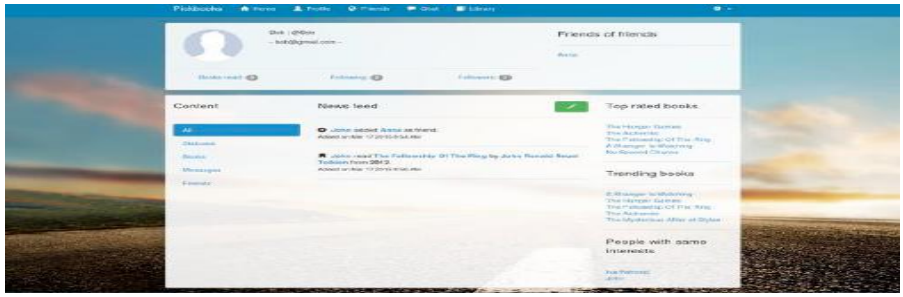


Figure 4. A community driven book recommender system[10]

1.2.5.6 Hybrid recommender systems

These RSs are based on the combination of the above techniques. For example, CF approaches face a number of issues like cold start, sparse matrix problems etc. This might not stop content based approaches to generate predictions. Thus combining two or more techniques, a hybrid system can be created to generate better personalized recommendations [11]. Figure 5 shows a hybrid apparel recommender system.

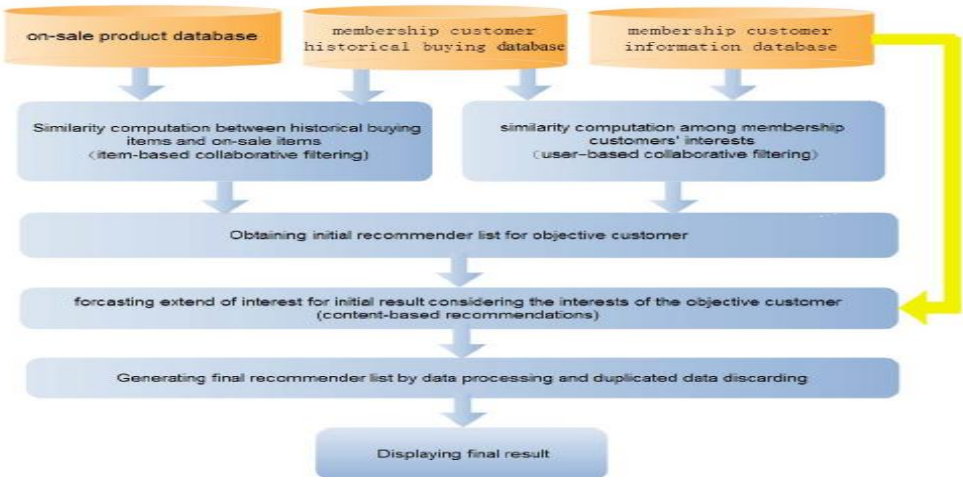


Figure 5. A hybrid recommender for an apparel retail[11]

1.3 Challenges in Recommendation Techniques

Recommender systems have been used extensively for generating recommendations using various recommendation techniques in multifarious application domains and this has brought into notice many challenges. Research areas are emphasizing on solving the issues mentioned below:

1.3.1 Data Sparsity

One of the challenges faced by rating correlation based recommendation systems is sparse user –item rating matrix. This happens when the user-item matrix is sparse and thus recommender systems become inefficient while dealing with nearest neighbor algorithms to calculate similarity. One of the solutions to deal with sparse user-item rating matrix problem is to fill the missing rating with a missing value, which could be the average rating for an item. Autonomous ratings like filterbots could be used to do this job. But this solution might lead to bias in recommendations. An effective method to deal with sparsity is matrix factorization or dimensionality reduction using SVD in which the matrix is projected into a latent space and users as well as items are associated to latent features.

1.3.2 Scalability

A fundamental issue of recommender systems is how to deal with massive datasets and real time data, obtained from users in the form of ratings, reviews etc. Thus scalability is a required feature in recommenders along with accuracy and coverage.

Scalability could be measured by evaluating the speed and resource consumption as the size of a dataset grows.

As there might be a trade off with accuracy, such compromises should be measured. If the accuracy is lower than an algorithm applied on a small dataset, the differences in accuracy over small datasets should be measured. Thus evaluation studies are needed to scale recommender systems to large datasets.

1.3.4 Gray sheep and Black sheep problem

This is another problem which occurs in recommender systems when a user's opinion does not agree or disagree with another user or set of users. Black sheep problem is an opposite case when the user's choice is so different that it is impossible to generate recommendations for him. Thus it is a failure on the part of the recommender system.

1.3.5 Shilling attacks

Recommender systems are vulnerable to shilling attacks or profile injection attacks. Malicious users tend to insert fake profiles in the system so that a certain item gets biased ratings in the user-item rating matrix. Shilling attacks could be classified as push attacks (Showing better rating of one's items) or nuke attacks (Reducing ratings of rival's products)[12]. An instance of generating false recommendations took place in 2001, when Sony Pictures agreed to the allegation that it had used false quotes to promote some newly released films. Thus shilling attacks are mostly used by unscrupulous producers for fun and profit. Shills or users who are influenced to enter the system ask for items in question. Their opinions and reviews thus mislead other users. This poses a threat to the item and cost of other users[13].

Reliable algorithms are needed to deal with shilling attacks, without impacting the accuracy or cost of the recommender system.

1.3.6 Cold-start problem

To recommend something to a user, a recommender system needs to know what the user liked in the past. For a new user, this is a challenging task and leads to the cold start problem. Thus it is important to address the cold start problem by answering demographic questions or rating items as the new user enters or using techniques like clustering [14]. Cold start could be user based or item based. If new user has no

records for generating recommendations then it is new user cold start problem. If a new item has just arrived and is not rated as of yet, it is called cold start problem for the new item.

The rating history and similarity methods can be used to address the cold start problem. Making use of attributes or demographic information can also be used to know the purchase and rating history of the customer. Another approach is to apply technique like clustering so that a system can learn about a new user and address this problem.

1.4 Evaluation Metrics of Recommender Systems

It is necessary to evaluate recommender systems before deployment. Evaluation is required at different stages. At the time of design, evaluation could be done to test the recommendation approach. This could be done by using various algorithms and analyzing the results. Evaluation can be useful if real time users are involved and parameters like nearest neighbors and thresholds are taken into consideration. Accuracy can be gauged with the help of the following metrics, considering dataset ratings from 1 to 5.

The evaluation metrics are divided into three classes: predictive accuracy metrics, classification accuracy metrics and rank accuracy metrics, described as follows:

1.4.1. Predictive Accuracy Metrics

These metrics predict how close the predicted ratings are to the actual ratings. For example, a recommender like Movielens predicts the number of stars a user shall give to the movie. New recommendations are thus generated, based on the fact how close these predictions are to the real number of stars a user give to each movie. Thus metrics that define predictive accuracy find the difference between predicted ratings and true ratings. Some of them are:

Mean absolute error (MAE) is used to measure the average absolute deviation between predicted ratings and user's actual ratings. For a test set T and user-item pair (u,i), MAE for actual rating r_{ui} and predicted rating $\hat{r}_{ui}$ is:

$$MAE = \sqrt{1/T \sum_{(u,i)\in T} |\hat{r}_{ui} - r_{ui}|} \qquad \text{.....(5)}$$

MAE is not useful if a user only views items that are ranked high in the results. Mean absolute error may be less appropriate as errors will only affect the process if a good item is classified into a bad one and vice versa. or vice versa. The advantages of employing MAE are that it is simple and easy.

Root Mean Squared Error*(RMSE)* is a useful measure of accuracy when the emphasis is on large errors.

$$RMSE = \sqrt{1/|T| \sum_{(u,i)\in T} (\hat{r}_{ui} - r_{ui})^2} \quad \text{.....(6)}$$

In comparison with MAE, RMSE is mostly used when large errors are not desired. In RMSE, the difference between predicted and actual observations are squared and then sample is averaged. Then the square root is taken. In MAE, absolute values of predicted and actual ratings are used. Thus accuracy is gauged over continuous variables[1].

1.4.2. Classification Accuracy Metrics

These metrics measure the frequency with which a recommender system makes correct or incorrect decisions. A metrics like precision, recall and F-score have been deployed for this purpose.

Precision and Recall – Precision and recall are important metrics to evaluate a recommender system [15]. For precision and recall, the following measures are often used: True Positives (TP) i.e. the number of instances belonging to a class that actually belong to it; True Negatives (TN) i.e. the number of instances that do not belong to a class and are not classified thus; False Positives(FP) i.e. the number of instances classified into a particular class although they do not belong to it and False Negatives(FN) i.e. instances classified into a particular class but not belonging to it. As per the following measures the accuracy of the system can be defined as the number of classified instances o the total instances.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad \text{.....(7)}$$

Precision is defined as the measure of number of errors made while classifying instances into a particular class. Recall measures how good we are at not leaving out the instances that should have been classified in a particular class.

$$Precision = \frac{TP}{TP+FP} \quad \text{.....(8)}$$

$$Recall = \frac{TP}{TP+FN} \quad \text{.....(9)}$$

It is possible to combine precision and recall into a single unit by introducing a metric F1 measure.

$$F1 = \frac{2*Precision*Recall}{Precision+Recall} \quad \text{.....(10)}$$

It is also possible that it is not required to estimate metrics independently but metrics for different models need to be compared.

ROC or Receiving Operating Characteristic curves are used for this purpose. These curves are a good alternative to precision and recall. ROC measures the extent to which an information filtering system can distinguish between noise and signal . It assumes that a predicted relevance is given to every item. An ROC distribution gives a relation between the positive rate and the negative rate. ROC also works on binary relevance like precision and recall. Items recommended ae either relevant or non-relevant. If all relevant items appear before the irrelevant ones then the curve would be perfect.

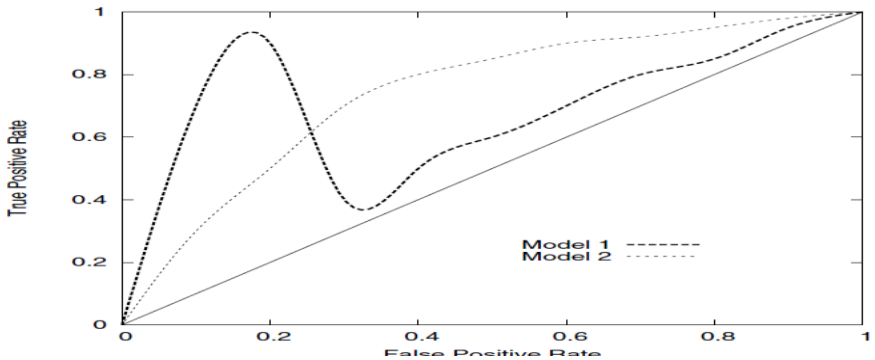


Figure 6. ROC as an evaluation metric

1.5 Structure of the thesis

The remaining thesis is structured based upon chapters as shown below:

Chapter 2: Literature Review. This chapter focuses on the related work that has been done in the field of recommender systems, its techniques and challenges.

Chapter 3: Problem Statement. This chapter throws light on the sparse matrix problem faced in CF and matrix factorization as the effective solution. Further, the problem statement is formulated in which accuracy of matrix factorization is improved by applying simulated annealing and finding mean absolute error.

Chapter 4: Proposed Methodology and Technologies used. In this chapter, the application of existing annealing strategies and proposed strategy is done on the learning rate parameter in matrix factorization. Mahout is used as the platform.

Chapter 5: Introduction to Dataset and Experiment Results. This chapter illustrates the results obtained by working on Movielens dataset . The observations and analysis section succeeded it.

Chapter 6: Conclusion and Future Scope. The whole work presented in thesis is summarized in this chapter and it also contains the scope for future research work in same or different problem domain.

Chapter 2

Literature Review

2.1 Overview of Recommender Systems

Recommender systems are widely used these days to help a user find his desired products from a huge set of options available. Collaborative filtering (CF) is the most popular approach in recommender systems. It makes use of a user item rating matrix and recommends on the basis of preferences and tastes of other users[1]. Content based filtering provides recommendations on the basis of items user previously liked. Table 1 gives an overview of the traditional approaches.

Table 1. Overview of recommender techniques[1].

NAME	DESCRIPTION	ADVANTAGES	DISADVANTAGES
User-based CF	Users who rated the same item have the same taste. Thus it recommends items based on items rated by other similar users.	i) Domain independent ii) Better quality of recommendations iii) Serendipity iv) No content analysis	i) New user problem ii) Scalability iii) Sparsity
Item-based CF	It recommends on the basis of similarity between items. Thus correlation is found.	i) Domain independent ii) Better quality iii) Serendipity iv) No content analysis	i) Cold start problem ii) Popular taste iii) Sparsity
Demographic CF	Attributes of similar users are matched and then recommendations are made	i) Domain independent ii) No cold start problem iii) Serendipity	i) Obtaining metadata information ii) Insufficient information iii) Only popular taste

CF faces a number of issues. Sparsity is one of the major issues in CF algorithms. The user item rating matrix is generally sparse in terms of entries because items are huge in number while number of users who rate them is small. CF cannot deal with this issue. Thus Matrix Factorization (MF) is an efficient approach to get rid of sparse matrix problems [3].It is a highly reliable and robust technique that helps to predict those ratings to a user for an item that are not yet rated by him.

2.2 Matrix Factorization and SVD

Matrix Factorization(MF) maps users and items to a latent space based on a given number of features. These latent features are obtained by characterizing items and users. For example, latent features in case of movies would be genres like comedy/drama/action etc. The latent feature with respect to item would describe the extent to which an item has this feature. With respect to user, these features define the amount of interest a user has in that item. Thus the missing entries are predicted by calculating products of pairs of feature vectors using Singular Value Decomposition (SVD) [7]. SVD is used to reduce the number of dimensions or features i.e. dimensionality reduction to resolve sparse matrix problems. Thus it helps to accept new users without re-computing the whole system model.

SVD has been a better approach as compared to the traditional CF model, as it splits up the rating matrix based on features(as explained in Equation 5).

MF introduces a set of dimensions in a latent space d i.e. the number of latent features in an incomplete matrix R, where each item i is associated with a vector $q_i \in R^d$, which gives the value to which an item possesses a particular dimension or feature. Every user u is associated with a vector $p_u \in R^d$, which shows the amount of interest a user takes in that particular dimension. The dot product $p_u.q_i^T$ gives us the user-item interaction according to a particular feature. The overall task of matrix factorization is achieved by the formula:

$$\min_{p,q} \sum_{(u,i\in K)}(r_{ui} - p_u\, q_i^T) + \Lambda(||q||^2 + ||p||^2 \qquad \text{.....(11)}$$

where K is the index of element in the existing matrix R, r_{ui} is the rating a user gives to an item I , Λ is the regularization parameter which can be obtained by cross validation and is used to avoid overfitting. || || is a Euclidean normalization standard.

Minimization in MF is done by either Alternating Least Squares(ALS) method or Stochastic Gradient Descent (SGD) technique. In ALS method, either of the p_u or qi values is treated as a constant and problem is solved using a quadratic method. SGD

technique is used to perform minimization on this function by iterating through all the ratings and generating a prediction rating $\hat{r}_{ui}$. The prediction error is found as in [16]

$e_{ui}=r_{ui} - \hat{r}_{ui}$ …….(12)

where e_{ui} is the prediction error.

For a given training case r_{ui},

$q_i \leftarrow q_i - \eta(\lambda q_i - e_{ui}p_u)$ ……(13)

$p_u \leftarrow p_u - \eta(\lambda p_u - e_{ui}q_i)$ .....(14)

Further, biases can be introduces when inner dot product is done during SVD i.e $p_u q_i^T$ [3]. A first order approximation of bias involving r_{ui} is:

$b_{ui} = \mu + b_i + b_u$ ……(15)

The bias b_{ui} accounts for the effects on user and item. μ gives the average rating and b_u,b_i give the deviations of user u and item i. Biases can be extended as:

$\hat{r}_{ui} = \mu + b_i + b_u + q_i$ …….(16)

The observed rating is split into global average, item and user bias as well as the interaction between the user and item. For example, we want a first-order estimate a for user's rating of *Titanic*. Say μ is 3.7 stars and *Titanic* is better than an average movie, so it tends to be rated 0.5 stars above average. The user rates it 0.3 stars lower than average. Thus his rating would be 3.9(3.7+0.5-0.3).Thus the system learns by minimizing the function:

$min_{p,q,b} \sum_{(u,i\in K)}(r_{ui} - \mu - b_u - b_i - p_u^T q_i)^2 + \lambda(||p||^2 + ||q||^2 + b_u^2 + b_i^2)$ ....(17)

Paterek has worked on the improvement of SVD by introducing biases in the concept of matrix factorization[17]. He added one parameter c_i for the user and d_i for the movie in the predicting rating.

$\hat{r}_{ui} = c_i + d_j + p_i q^T$ ……...(18)

These biases are trained along with matrices p_u and q_i. Thus it was an improvement in regularized SVD.

$$c_i = c_i + \eta(r_{ui} - \Lambda(c_i + d_j - \text{global\_mean})) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (19)$$

$$d_j = + \eta(r_{ui} - \Lambda(c_i + d_j - \text{global\_mean})) \quad \ldots\ldots\ldots\ldots\ldots\ldots (20)$$

The values were set as : η=0.01, $\Lambda = 0.05$ and global_mean=3.6033

This model has O(QR+PR) parameters, where P is the number of users, q is the number of movies and R is the number of features. Paterek suggested to decrease the number of parameters, instead of fitting p_u for each user separately, it can be shown as a function of a binary vector indicating which movies are rated.

$$\hat{r}_{ui} = c_i + d_j + e_i \sum_{k=1}^{r} q_{ik} \sum_{j \in M} w_{jk} \quad \ldots\ldots\ldots\ldots\ldots\ldots (21)$$

where M is the set of movies rated by the user and $p_{uk} \epsilon e_i \sum_{j \epsilon M} w_{jk}$. He further evaluated the models with the help of RMSE. . Thus each item was linked to two vectors q_i and x_i and user is predicted for a set of ratings R(u) as $b_{ui} + \frac{q_i^T(\sum_{j \in R(u)} x_j)}{\sqrt{|R(u)|}}$.

Koren further worked on the improvement of SVD by introducing the SVD++ model[18].His model helped to introduce accuracy and also added implicit feedback to the already existing SVD model. As described, global weights w_{ij} were assigned and the model was independent of specific users. For the purpose of global optimization, these weights were represented as baseline estimates in this approach. Usually these weights in the neighborhood model are interpolation coefficients that relate unknown ratings to the present ratings. r_{uj}-b_{uj} will be used to multiply these offsets. For two items i and j it is proposed that the estimate can be increased by adding $(r_{uj}$-$b_{uj})w_{ij}$ to the baseline if user rated higher than expected r_{uj}-b_{uj}.

$$\hat{r}_{ui} = b_{ui} + \sum_{j \in R(u)} (r_{uj} - b_j) w_{ij} + \sum_{j \in N(u)} c_{ij} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (22)$$

Thus the bias parameters get optimized like w_{ij} and c_{ij}. Devising such advanced models helps to allow the integration of explicit and implicit user feedback. For example, a movie dataset not only tells about the rating values but also which movies have been rated by the user. This is a kind of implicit feedback, as the preferences can be transformed into a binary matrix: 1 for rated and 0 for not rated. Including such data improves the accuracy of prediction. This has been made use of in previous techniques like conditional RBMs[18] .The incremental models of SVD have also been studied in[19]. Authors introduce the concept of bounded SVD, which confines the ratings in a matrix i.e. bounds are defined beforehand [20]. It was found that this method works better than traditional matrix factorization parameters.

Other improved matrix factorization techniques include improved Minimum Margin Matrix Factorization (MMMF) which aims at introducing offset terms and regularization factors for improving the existing marginal matrix factorization approach[21]. An approximation factor F is introduced (F=UM) where U_{i*} is the user feature matrix and M_{*j} is the item feature matrix.

$$F_{ij} = (U_{i*}, M_{*j}) \quad \text{.....(23)}$$

The regularized loss function is minimized using Froebenius norm ($||U||^2_F = \text{tr } UU^T$) as implemented by Srebro et al.[22]. Thus an optimization problem is obtained as:

$$minimize_{U,M}\ L(F,Y) + \frac{\Lambda_m}{2}||M_F||^2 + \frac{\Lambda_u}{2}\ ||U_F||^2 \quad \text{.....(24)}$$

where Λ_m and Λ_u are the regularization parameters and L(F,Y) gives the difference between F and Y parameters.

Other improvements in MF include latent semantic analysis [23]. It uses an approximation factor to minimize the sum of the square of distances between entries in the original matrix and those calculated as per the approximation factor. Expectation Maximization proposed by authors has also been implemented on MF[24]. Coordinate Descent approach has also been proposed to parallelize and optimize MF[25].

2.3 Impact of Simulated Annealing on Learning Rate

η or learning rate has a great impact on the convergence rate of each iteration in the proposed system as well as the computational cost and complexity. So, it is essential to choose appropriate value of learning rate so that an optimum value is obtained. Numerical methods like SGD are used to learn the parameters of MF. Learning rate set in gradient descent is critical as it determines the number of iterations the process would take to converge and the computational cost required to reach the local optimum as described by the authors in [26]. Due to high complexity of search space, it is often difficult to predict the learning rate of a system. Thus various approaches have been proposed. In [27], initial learning rate was chosen by dividing the previous gradients obtained in the next iterations.

The change in learning rate has been adapted in [28] by integrating three strategies- deterministic step size adaptation, incremental bar delta bar delta and stochastic meta descent. It is observed that for a lower value of learning rate, there is a high prediction accuracy or lower value of Root Mean Squared Error. Thus an adaptive learning rate is chosen by changing the step size and applying the above mentioned techniques to regularize MF.

One of these techniques, deterministic step size adaptation has been implemented in [29] in which dot product is replaced by sum of products as shown:

$$\widehat{r_{ui}} = \sum_{k=1}^{d} p_{u,k}\, q_{i,k}\ (p_{u,0} = q_{i,k} = 1) \qquad \text{.....(25)}$$

Two parameters $d_{u,k}$ and $d_{i,k}$ are calculated to judge the direction of p_u,q_i and r_{ui}.

$$p_{u,k}^{t} = p_{u,k}^{t-1} + \eta d_{u,k}^{t} \quad p_{u,0} = 1 \qquad \text{.....(26)}$$

$$q_{i,k}^{t} = q_{i,k}^{t-1} + \eta d_{i,k}^{t} \quad q_{i,j} = 1 \qquad \text{.....(27)}$$

The adapted learning rate is calculated as follows:

$$\eta_{u,k}^{t} = \begin{matrix} \eta_{u,k}^{t-1}\ x\ \alpha\ if\ d_{u,k}^{t-1}\ x\ d_{u,k}^{t} > 0 \\ \eta_{u,k}^{t-1}\ x\ \beta\ if\ d_{u,k}^{t-1}\ x\ d_{u,k}^{t} < 0 \end{matrix} \qquad \text{.....(28)}$$

$$\eta_{i,k}^{t} = \eta_{i,k}^{t-1} \, x \, \alpha \; if \; d_{i,k}^{t-1} \, x \, d_{i,k}^{t} > 0 \qquad \text{.....(29)}$$
$$\eta_{i,k}^{t-1} \, x \, \beta \; if \; d_{i,k}^{t-1} \, x \, d_{i,k}^{t} < 0$$

Another variation of learning rate with the number of iterations has been done using a linear regression model which helps to reduce the number of iterations upto 50% [30]. Thus matrix factorization is accelerated by using an optimum value of learning rate. As error for p users, q items and S sparse matrix is calculated as $argmin_{p,q} \; err(pq^T, S)$, the model parameters are learned by SGD i.e.

$\odot$=(p,q) is the partial derivative with respect to each parameter. Thus at each step, updations would be treated by the following formula as per this approach:

$$\odot = \odot - \alpha(\eth / \eth \odot err(pq^T, S)) \qquad \text{.....(30)}$$

α is the learning rate after each iteration.

Similar to this approach, simulated annealing can be applied to initial learning rate to predict the output of upcoming iterations. Simulated annealing is an adaptation model which works on a random combination of individuals by choosing an initial energy E_0 which gives the quality of this combination. The combination is better if the energy is lower. A variable called temperature T decrements with time. This change in temperature leads to changes in the random combination. The cost d associated with the change is defined as the difference in energy in the current combination and the energy in the previous modification. If the cost is negative, the current combination has lower energy so it better, else it is not a good change, as proposed in[31]. It also depends on a probability value called acceptance rate. Higher the temperature, higher is the acceptance rate. Acceptance rate for temperature at level k is defined as :

$$t_a = e^{-\frac{d}{T_k}} \qquad \text{.....(31)}$$

Simulated annealing is important for algorithms when convergence towards the best solutions is required. Thus the temperature parameter has been worked on by using various cooling schedules and strategies compared in [32] . The

Considering an initial solution S_1 which would set initial temperature and generate further solutions based on generator functions, the simulated annealing algorithm can be divided into the following steps:

1. Set an initial random solution as the current solution i.e $S_1=S_0$

2. Set initial temperature parameter $T = T_0$

3. Calculate the cost using the function defined.

4. Generate a new solution S_i based on a random neighbor function or annealing schedules (in this case).

5. Compare the new solution S_i with the current solution:

If cost of new solution < cost of old solution : move to the new solution

In this case, new solution S_i is better than the current one, so it is set as the current solution.

6. Repeat the steps until an acceptable solution is found or the maximum number of iterations are reached.

Simulated annealing converges to a set of globally optimal solutions. In [33], the temperature is decreased logarithmically according to equation:

$$T_k = \frac{T_0}{1+\log(1+k)} \quad \text{.....(32)}$$

Generally, an exponential cooling schedule is used function is used, such as

$$T_k = T_0.\alpha^k \ (0.8 <\alpha<0.9) \quad \text{.....(33)}$$

The temperature decrease is made multiplying initial temperature by a factor decreasing exponentially with respect to temperature for each iteration.

By including a cooling speedup , the initial temperature is decreased by multiplying T_0 in inverse proportion to logarithm of iteration k, known as logarithmic multiplicative cooling.

$$T_k = \frac{T_0}{1+ \alpha \log (1+k)} \ (\alpha > 1)$$

……..…….(34)

Another cooling schedule called linear multiplicative cooling decreases temperature in inverse proportion to iteration k[33]:

$$T_k = \frac{T_0}{1+\alpha k} \ \alpha > 0 \qquad \text{…….(35)}$$

Quadratic cooling decreases temperature in inverse proportion to square of cycle k:

$$T_k = \frac{T_0}{1+\alpha k^2} \ \alpha > 0 \qquad \text{…….(36)}$$

Simulated annealing could be applied to learning rate in recommender systems and thus it can be determined with the help of annealing schedules. A technique to improve annealing schedule known as per coordinate schedule has been proposed by introducing the concept of twin learners by Chin et al. [34] .

In [35] it was found that exponential annealing is the best approach for mobile recommendation systems in terms of efficiency and effectiveness. The results were given on the basis of travelling salesman problem and mobile route recommendation problem.

The existing annealing schedules have been applied to learning rate in the following manner:

i) Exponential annealing- This annealing schedule sets the learning rate value based on the following formula:

$$\eta(i) = \eta_0 * \text{pow}(\text{base}, i) \qquad \text{…….(37)}$$

where η_o is the initial learning rate and i is the corresponding iteration. Based on the base of the exponent, the learning rate decays at an exponential rate for each iteration.

ii) Inverse scaling-In this schedule, the annealing schedule α lowers the rate much quickly for the initial iterations and then slows down for the later ones. The annealing rate is fixed by the user.

It sets the learning rate based on the formula:

$$\eta(i) = \frac{\eta_0}{1+\alpha} \qquad \text{…….(38)}$$

iii) Logarithmic cooling- This schedule determines the learning rate as per the formula:

$$\eta(i) = \frac{\eta_0}{1+\beta*\log(i+d)} \qquad \text{.....(39)}$$

where d is constantly set to 1. The bias factor β is set to be greater than 1.This schedule generally leads to a random search in the state space.

iv) Linear Multiplicative Cooling(LMC)- The initial learning rate is decreased by a factor which is inversely proportional to iteration i. The bias factor β is set between 0 to 1.

$$\eta = \frac{\eta_0}{1+\beta*i} \qquad \text{.....(40)}$$

v) Quadratic Multiplicative Cooling(QMC)- The initial learning rate is decreased by a factor inversely proportional to square of iteration i. The bias factor β is set between 0 to 1.

$$\eta = \frac{\eta_0}{1+\beta*i*i} \qquad \text{.....(41)}$$

Thus simulated annealing has a great impact on the learning rate and hence on the accuracy of the recommender system. The given annealing schedules have been applied to the Movielens dataset and results have been validated based on MAE calculations. These have been discussed and analyzed in Chapters 3,4 and 5.

Chapter 3

Problem Statement

3.1 Introduction

User-based Collaborative-filtering (CF) technique is the most frequently used recommender technique due to its simplicity and efficient performance. But it fails to deal with sparse matrix problems that occur in the user-item rating matrix. Matrix Factorization is a highly scalable approach being used in recommender systems these days. Learning rate plays a very important role to determine the accuracy and convergence rate of these recommender systems, but it is difficult to determine its optimum value. In this thesis, a new annealing schedule based on stochastic gradient descent has been chosen as the base approach to discuss the effect of learning rate on the accuracy of a recommender system. It has been compared with existing annealing schedules- exponential annealing, inverse scaling, logarithmic cooling, linear multiplicative cooling and quadratic multiplicative cooling. The experimental results on Movielens dataset prove that by employing exponential annealing schedule as the learning rate, minimum mean absolute error can be attained for the system at a lower value of learning rate. For higher learning rate values, the proposed annealing schedule works best. Apache Mahout 0.9 is chosen as the platform for the research which serves as an efficient and scalable platform for this purpose.

3.2 Objectives

The main focus of this work is to apply simulated annealing techniques to learning rate and predict the mean absolute error in the recommender system. Based on the use of five annealing schedules- exponential annealing, inverse scaling, linear multiplicative cooling, logarithmic cooling and quadratic multiplicative cooling, the accuracy has been calculated by fixing a range of learning rate values. Further, a new annealing schedule has been proposed which gives better accuracy for some values of learning rate. The main objectives achieved through the work are:

- Apache Mahout has been used as the platform to implement SVDRecommender based on Movielens . A Recommender evaluator is used to find

the difference between the actual ratings and the predicted ratings. Thus MAE is evaluated.

- Based on SGD approach in MF, the concept of annealing schedules has been introduced to minimize MAE in CF recommender systems. SGD is a simple and efficient minimization technique used in MF to get rid of sparse matrix problems in tradition recommender systems. This is done by associating latent features with the involved items and users.

- The learning rate parameter in SGD is varied by using five different annealing schedules like exponential annealing, inverse scaling , logarithmic cooling, linear multiplicative cooling and quadratic multiplicative cooling . Based on an initial learning rate and the application of these annealing strategies, the corresponding MAE values are used to predict the accuracy of the system.

- A new annealing schedule is proposed which is compared with the annealing schedules applied to the system. It is observed that proposed annealing schedule outperforms the other annealing strategies for higher values of learning rate. For lower values, exponential annealing works the best.

- Segregation of Movielens is done i.e. for dataset sizes n=10000, 100000 and 1000000, these six annealing schedules are tested for given initial values of learning rate. The results are validated for all the sizes used for the experiments. It is proved that learning rate affected by simulated annealing greatly affects the working and accuracy of our implemented recommender system.

Chapter 4

Proposed Technique and Technologies Used

4.1 Building a Recommender in Apache Mahout

Mahout was developed as a part of Apache Lucene's project, commonly used to perform search, text mining and information retrieval. Mahout provides a flexible engine for CF and matrix factorization in recommender systems. Mahout is an open source platform which provides tools and libraries to build recommender systems[35]. It supports classification, clustering and collaborative filtering algorithms. Mahout helps to provide user based and item based recommendations with the help of neighborhood formation and similarity calculations. It was started as project called 'Taste' and later continued to be used inside Mahout with other Hadoop based code. A Mahout based collaborative engine stores the tastes and preferences of a user and returns estimated preferences for the other items. In this case, based on the previous ratings of movies, Mahout uses the details to recommend other movies that a user might be interested in. Mahout can also be used by recommender systems to scale to huge datasets by using MapReduce i.e. involving parallel processing of datasets, making it a more efficient workbench for implementation.

Many interfaces are used in Mahout to develop a recommender system[36]. The sub package org.apache.mahout.cf.taste.impl holds implementations for these interfaces. Five such primary components or interfaces to build a recommender in Mahout are:

1) DataModel: It is a storage for items ,users and preferences. Data might be drawn from any source, be it JDBC or MySQL. Users and items are identified by using IDs. These IDs must be numeric as it identifies Java long type. A Preference object encapsulates the relation that is present between the users and their preferred items. Mahout also supports FileDataModel interface which is used for very small applications . A so-called Boolean data model is also supported in which users simply express an association for an item or none at all. For example, while users might express a preference from 1 to 5 in the context of a movie recommender site, there might be an association between a user and pages that have been visited.

2) UserSimilarity: It is an interface that defines the similarity between two users. This is an important part of a recommendation engine. These are related to a **Neighborhood** implementation in the system.

3) ItemSimilarity: It is an interface that gives similarity between two items . This is an important part of a recommendation engine. These are related to a **Neighborhood** implementation in the system.

4) UserNeighborhood: It gives an interface for computing a neighborhood of users who are similar so that recommendations can be given as per their preferences.

5) Recommender: It is an interface for provides recommendations .The recommender could be User based or item based. User-based recommenders are conventional style of recommender systems. They involve the following steps: creating a DataModel using a .csv file dataset, implementing UserSimilarity algorithm (say pearsonCorrelationSimilarity) and then creating a UserNeighborhood to recommend items for the user. In an item based recommender, item similarity is calculated instead of user similarity.

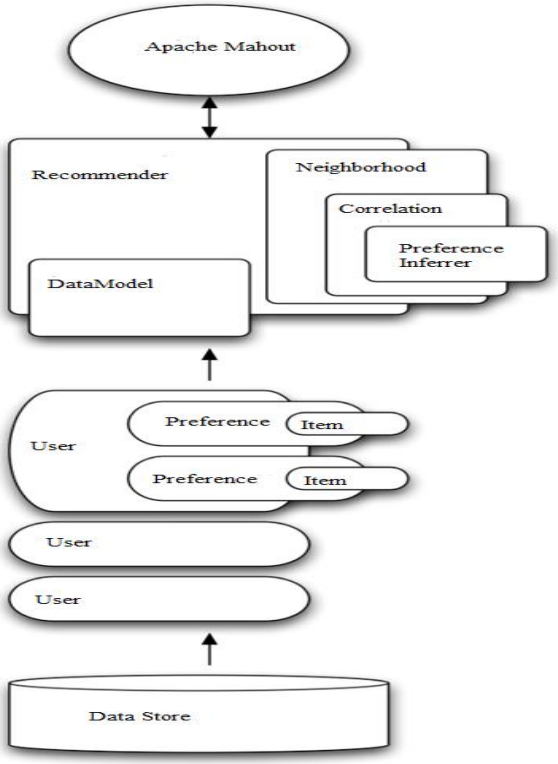


Figure 7. A user based recommender in Mahout

6) SVDRecommender: SVDRecommender is an interface in Mahout which helps to build MF recommender systems. Mahout has implemented matrix factorization based on stochastic gradient descent(SGD) and alternating least squares(ALS)techniques. Both these techniques associate vectors to items and users based latent features. In ALS method, either of the p_u or qi values is treated as a constant and problem is solved using a quadratic method. SGD technique is used to perform minimization on this function by iterating through all the ratings and generating a prediction rating

7) AverageAbsoluteDifferenceRecommenderEvaluator : Mahout provides this interface which computes the average absolute difference between predicted and actual ratings for users. Thus it is an evaluator to predict the MAE in the ratings, helping to gauge the accuracy of the recommender system. The Mahout libraries can also include Hadoop tools to make recommendations on large datasets. Thus a lot of user preferences can be met this way[36]. Scalable and efficient recommenders can be built.

The training data is used to build a recommender system while evaluating it . The recommender estimates the preference of the user. Then it picks the actual preference data and compares how accurate the predicted preference was.Thus while evaluating with the help of different metrics, MAE or RMSE hwelp to interpret the deviation between actual and estimated ratings.

Mahout is an efficient platform to build a recommender as has a wide range of resources for collaborative filtering algorithms,classification,clustering etc. User based, item based, SVD and SlopeOne recommenders can be easily implemented with its help. RMSE and MAE can be easily calculated with the help of its evaluation methods. It is highly scalable and can support parallel processing on large datasets [37].

Installing Mahout in Ubuntu. Ubuntu is used as the operating system to carry out this implementation. Ubuntu is used as it is open source and can host various cloud and server applications. The prerequisites to install Mahout in Ubuntu are:

- Java 1.6 or above
- Maven 2 or above

Once they are installed, user is ready to use Mahout.

4.2 Methodology used

The annealing schedules explained in the previous section are applied on the Movielens dataset, which comprises of 6000 users and 4000 movies with over 100000 ratings.

Table 2. Dataset statistics

	No of users	No of items	No of ratings
Movielens	6000	4000	1000000

4.2.1 Proposed Annealing Schedule(SRA)

Besides working on the existing schedules, a new annealing schedule (SRA) is proposed in which the initial learning rate is decreased by a factor inversely proportional to square root of iteration i. The bias factor β is set to be greater than 1. This gives an improvement in accuracy for the system as compared to the schedules discussed in the previous sections.

$$\eta(i) = \frac{\eta_0}{1+\beta*\mathrm{sqrt}(i)} \qquad \text{.....(42)}$$

Results are discussed in Chapter 5.

4.2.2 Implementation in Mahout

Apache Mahout 0.9 is used as the platform to implement these annealing schedules. It makes use of SVDRecommender and ParallelSGDFactorizer interfaces to implement matrix factorization in the recommender systems. AverageAbsoluteDifferenceRecommenderEvaluator is used to find MAE between the actual and predicted ratings by a user.

SGD is used for minimization in matrix factorization.The learning rate parameter is varied by using six different annealing schedules like exponential annealing, inverse scaling , logarithmic cooling, linear multiplicative cooling , quadratic multiplicative cooling and proposed annealing schedule. Based on an initial learning rate and the application of these annealing strategies, the corresponding MAE values are used to predict the accuracy of the system.

The chosen dataset, Movielens consisting of movie ratings, is divided into different samples, 10000,100000 and 1000000. The value of learning rate is ranged from 0.008 to 0.09. The regularization parameter ʎ and number of features are set to 0.05 and 3 respectively. The annealing rate in case of inverse scaling is set to 0.05. All experiments are carried out on a PC with 4 GB RAM and i5 processor using Java .The proposed methodology is shown in the figure below:

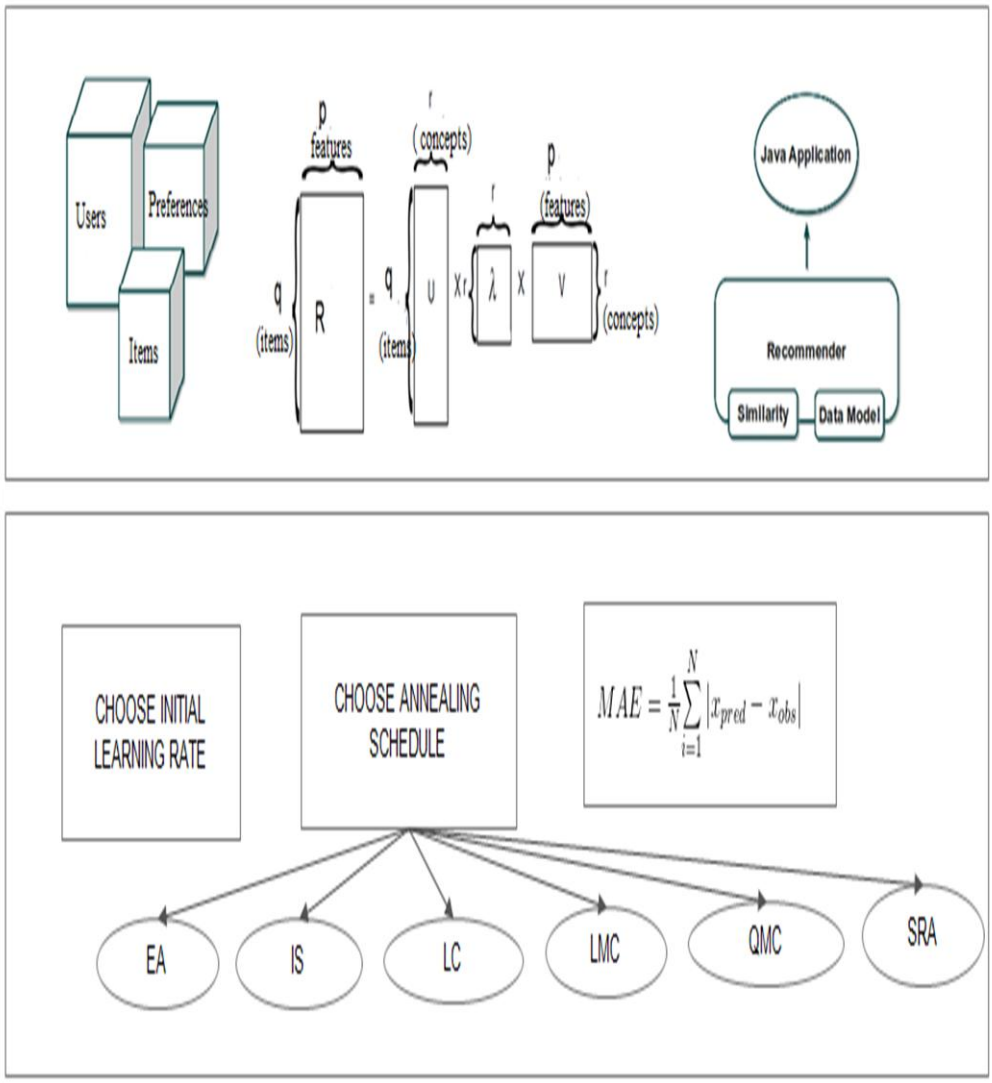


Figure 8 . Applying simulated annealing schedules to recommender system

Finally, the values of MAE are compared for six annealing schedules and it is observed that exponential annealing is the best simulated annealing technique for m lower learning rate values. As the value of learning rate increases, the proposed annealing schedule works best, as it records a better accuracy for the recommender system. The results are validated for all the three sizes used for the experiments. It is proved that learning rate affected by simulated annealing greatly affects the working and accuracy of the implemented recommender system.

4.2.3 Use of Eclipse

Eclipse is an integrated Development Environment (IDE) used to implement codes in Java and other programming languages like C,C++,Ruby etc.

In this thesis, Java is used and Mahout libraries are imported in Eclipse to implement a recommender system. The operating system used is **Ubuntu.** The ratings file in extension .dat are also converted to .csv with the help of Java in this platform.

The steps to import Mahout libraries in Eclipse are as follows:

- Click on New-> Java Project in Eclipse.
- Give a new project name and click on Finish.
- Right click on the new project and add New->package. Name it default.
- Right click on default package and click on Configure->Add External Archives
- Browse through the Mahout jars in /usr/local/hadoop/mahout-0.9-cdh5.4.0/lib and add the mahout jar.
- Once the libraries have been imported, a new project is created in Java and proposed methodology is implemented.

The implementation of methodology and results have been described in detail in Chapter 5.

Chapter 5

Experiment Results

5.1 Introduction to MovieLens Dataset

ml-1M dataset is chosen for the purpose of experiments. It contains 1000209 anonymous ratings of approximately 3900 movies by 6040 users . These ratings go back to as far as the year 2000.

It describes a 5-star rating activity by the users from movie recommender service, Movielens (http://movielens.org), a movie recommendation service. This dataset consists of the following files:

DATA FILES DETAILS OF MOVIELENS ML-1M DATABASE

i) **RATINGS DATA FILE STRUCTURE (ratings.dat)**

This file contains information in the format given below:

userId, movieId, rating, timestamp

The user id is followed by movie id and rating is given on a scale of 1 to 5.Timestamps are recorded in seconds since epoch.

ii) **MOVIES DATA FILE STRUCTURE (movies.csv)**

Movie information is contained in the file `movies.dat`. This file has the following format:

movieId, title, genre

Movie titles are imported https://www.themoviedb.org. Genres are separated by pipes and represented as:

Action| Adventure| Animation| Children's| Comedy| Crime| Documentary| Drama| Fantasy (some of the genres). Errors may exist as movies are entered manually.

iii) USERS DATA FILE STRUCTURE

User information is in the file "users.dat" and is in the following format:

UserID::Gender::Age::Occupation::Zip-code

This file is converted into .csv for experimentation using Java in Eclipse. All demographic information is provided voluntarily by the users and its detailed description is shown below:

- User gender is denoted by M(Male) and F(Female)
- Age is chosen from the ranges:
- 1(Under 18)
- 18(18-24)
- 25(25-34)
- 35(35-44)
- 45(45-49)
- 50(50-55)
- 56(56+)

Occupation of users could range from:

0: "other" or not specified
1: "academic/educator"
2: "artist"
3: "clerical/admin"
4: "college/grad student"
5: "customer service"
6: "doctor/health care"
7: "executive/managerial"
8: "farmer"
11: "lawyer"
12: "programmer"
13: "retired"
14: "sales/marketing"
15: "scientist"
16: "self-employed"
17: "technician/engineer"
18: "tradesman/craftsman"
19: "unemployed"

5.2 Experiment I: Part A

The tools used for the experiment are Eclipse and Apache Mahout. Using a Java code, the first part of the experiment is initiated i.e. the ratings dataset ratings.dat is converted into a csv file so that it is in a useable format to generate recommendations. The following steps are followed:

- Click on New-> Java Project in Eclipse.
- Give a new project name and click on Finish.
- Right click on the new project and add New->package. Name it default.
- Right click on default package and add New->Class. Name the class Convert.
- Using the Java code, convert ratings.dat to ratings.csv.
- Repeat the steps for 10000,100000 and 1000000 movie rating entries

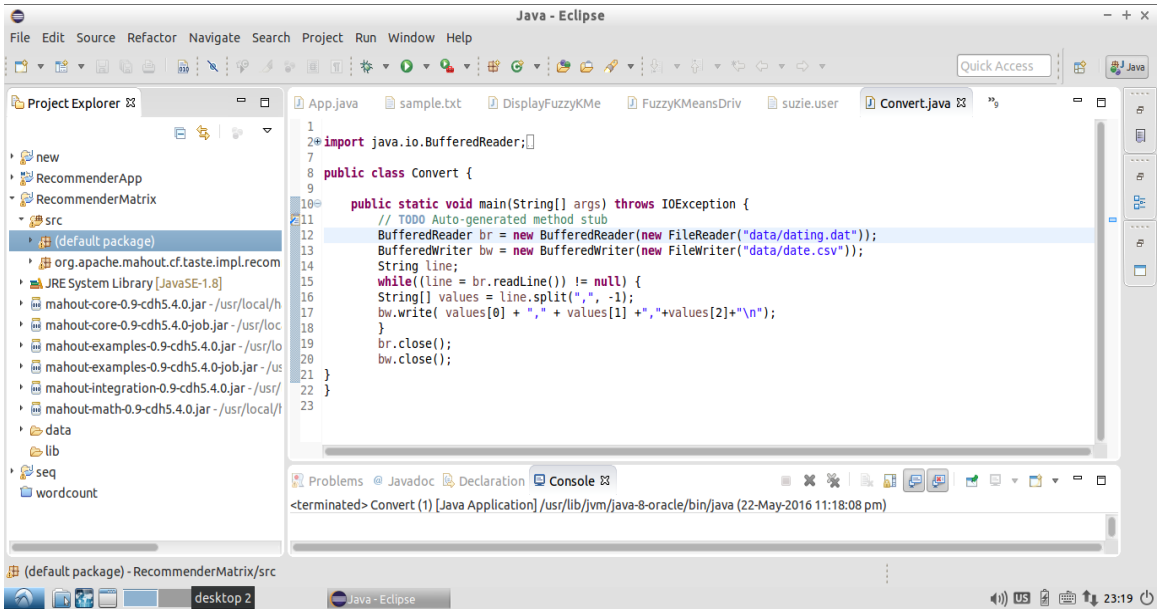


Figure 9. Converting .dat files to .csv files in Eclipse

5.3 Experiment I: PART-B

The next step of the experiment is to build a recommender using Mahout libraries and interfaces. Using the SVDRecommender interface, a movie recommender is built which predicts movies from Movielens dataset. The Movielens dataset is divided into three sizes, n=10000, 100000 and 1000000. The following recommendations are generated from dataset of size 1000000 (named as mvis.csv).

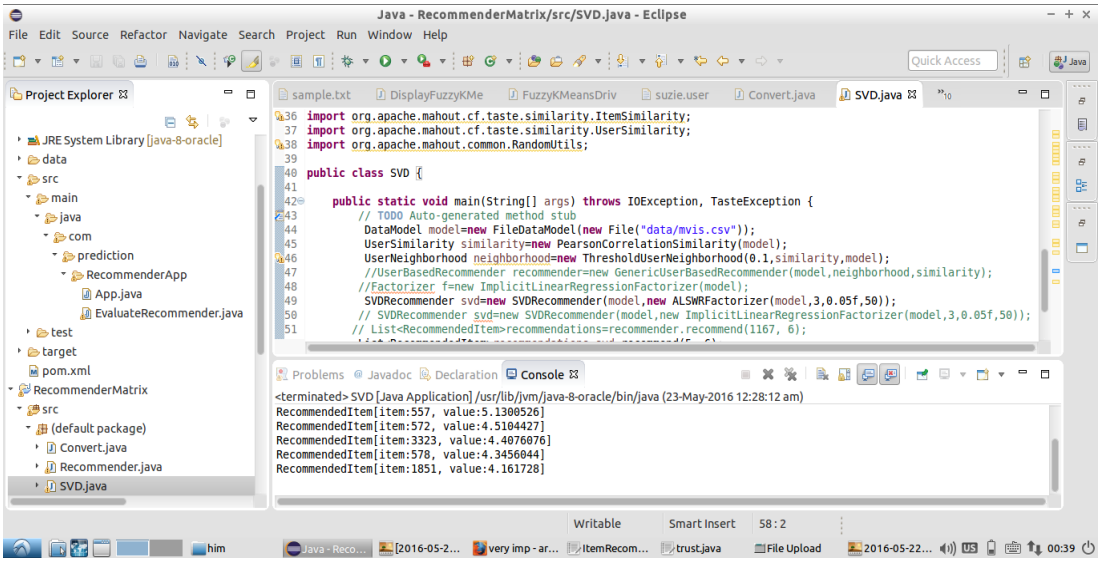


Figure 10 . SVD Recommender in Mahout

5.4 Experiment II: Applying Simulated Annealing

The next step is to apply the six annealing schedules- exponential annealing, inverse scaling, logarithmic cooling, linear multiplicative cooling , quadratic multiplicative cooling and proposed annealing schedule SRA on the Movielens dataset divided into three different sizes. The regularization parameter is set to 0.05, the number of features are set to 3 and the learning rate varies from 0.008 to 0.09.

- For n =1000000, the following table gives the MAE values for the six annealing schedules , given the range of learning rate values.

Table 3. MAE values for annealing schedules (n=10000)

η	EA	IS	LC	LMC	QMC	SRA
0.008	0.70356	0.88184	0.73761	0.743	0.78875	0.74575
0.01	0.7049	0.87482	0.73284	0.73765	0.7789	0.74011
0.015	0.70925	0.85889	0.72572	0.72944	0.76302	0.73138
0.018	0.71131	0.85035	0.72305	0.72638	0.75674	0.7281
0.02	0.71251	0.84509	0.72165	0.72476	0.75335	0.72639
0.03	0.71698	0.82426	0.71541	0.71931	0.74198	0.72067
0.05	0.72719	0.79817	0.70349	0.70683	0.73081	0.70992
0.08	0.72598	0.77684	0.70686	0.70424	0.72322	0.7037

- Table 4 gives statistics for Movielens dataset size 100000.

Table 4. MAE values for annealing schedules (n=100000)

	EA	IS	LC	LMC	QMC	SRA
0.008	0.77356	0.91123	0.8091	0.81496	0.85293	0.8177
0.01	0.77797	0.90732	0.80323	0.80912	0.84533	0.81191
0.015	0.80152	0.89841	0.79294	0.79853	0.83276	0.8013
0.018	0.8131	0.89395	0.78863	0.79395	0.82766	0.79663
0.02	0.81945	0.89057	0.7863	0.79138	0.8248	0.79399
0.03	0.84318	0.87738	0.77849	0.78268	0.81394	0.78483
0.05	0.86419	0.85956	0.77363	0.77546	0.80042	0.77664
0.08	0.87485	0.84369	0.78724	0.7756	0.7877	0.77395

- For n=10000, the statistics are as shown below:

Table 5. MAE values for annealing schedules (n=1000000)

	EA	IS	LC	LMC	QMC	SRA
0.008	0.77356	0.91123	0.8091	0.81496	0.85293	0.8177
0.01	0.77797	0.90732	0.80323	0.80912	0.84533	0.81191
0.015	0.80152	0.89841	0.79294	0.79853	0.83276	0.8013
0.018	0.8131	0.89395	0.78863	0.79395	0.82766	0.79663
0.02	0.81945	0.89752	0.7863	0.79138	0.8248	0.79399
0.03	0.84318	0.87738	0.77894	0.78268	0.81394	0.78483
0.05	0.86419	0.85956	0.77363	0.77546	0.80042	0.77664
0.08	0.87485	0.84369	0.78724	0.7756	0.7877	0.77395

From the experimental results, it is evident that exponential annealing is the best annealing approach at a lower value of learning rate. Proposed annealing schedule SRA beats exponential annealing as the learning rate values get higher. The mean absolute error is minimum for the recommender system in these cases.

At a learning rate value =0.008, exponential annealing gives the best accuracy, with MAE 0.70356(for n=1000000). For a higher value =0.08, SRA(proposed schedule) gives an error of 0.70378(for n=1000000) which is the lowest against all the six

annealing schedules. Further, a new annealing schedule could be designed to suggest the best learning rate to give an improved value of accuracy for the system.

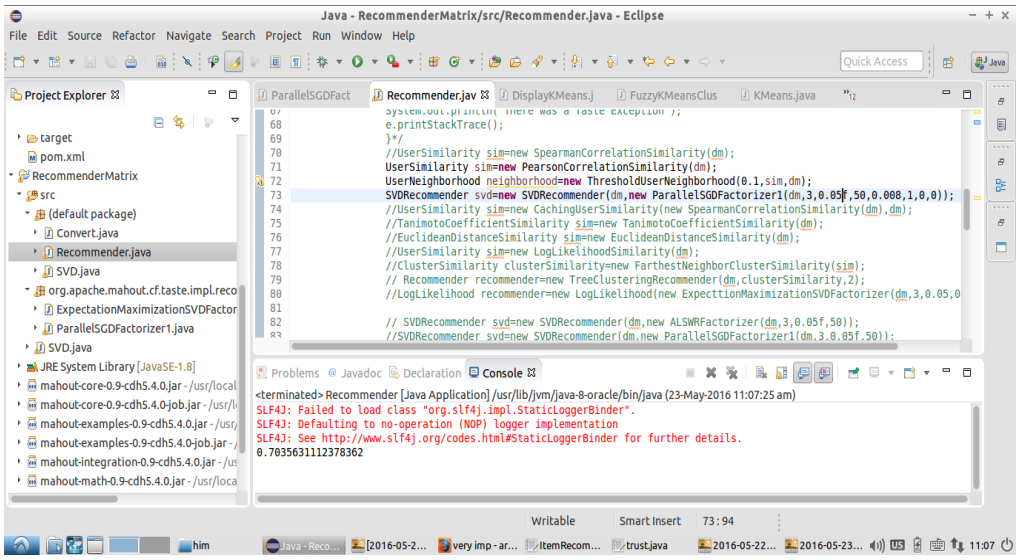


Figure 11 . MAE value for EA schedule (n=1000000)

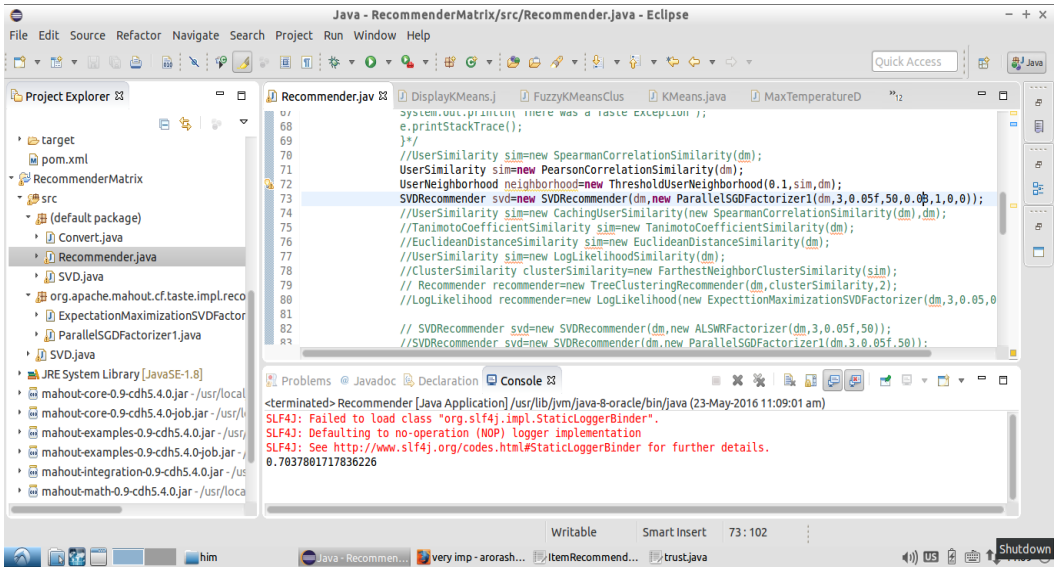


Figure 12. MAE for SRA schedule (n=1000000)

As learning rate increments, the value of error increases for exponential annealing. Thus lower learning rates should be preferred for better learning. By applying the proposed schedule, the value of mean absolute error becomes lower than the other schedules as learning rate increases. Inverse scaling shows a steady decrease in this value as we approach higher learning rates. But still accuracy is poor. Figures 16, 17 and 18 confirm these results for different sizes of the datasets.

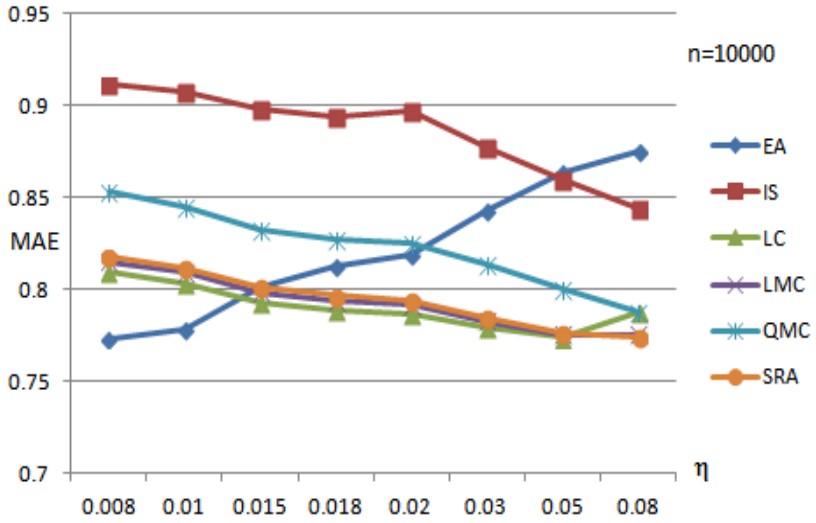


Figure 13 . MAE vs learning rate statistics for n=10000

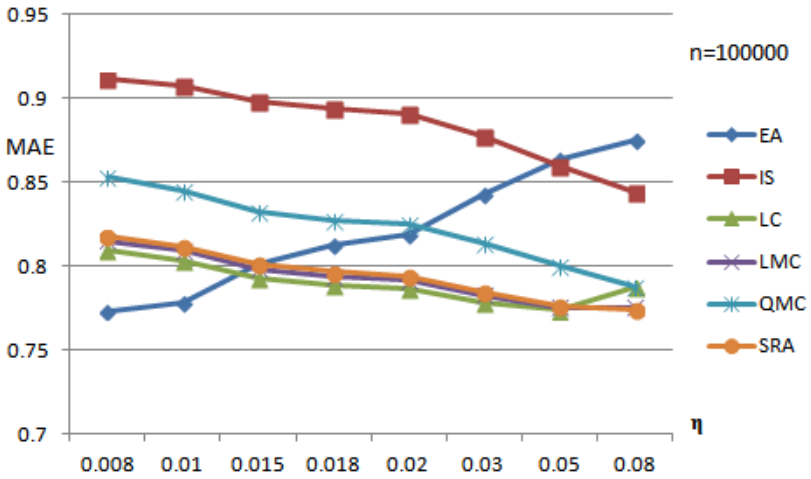


Figure 14 . MAE vs learning rate statistics for n=100000

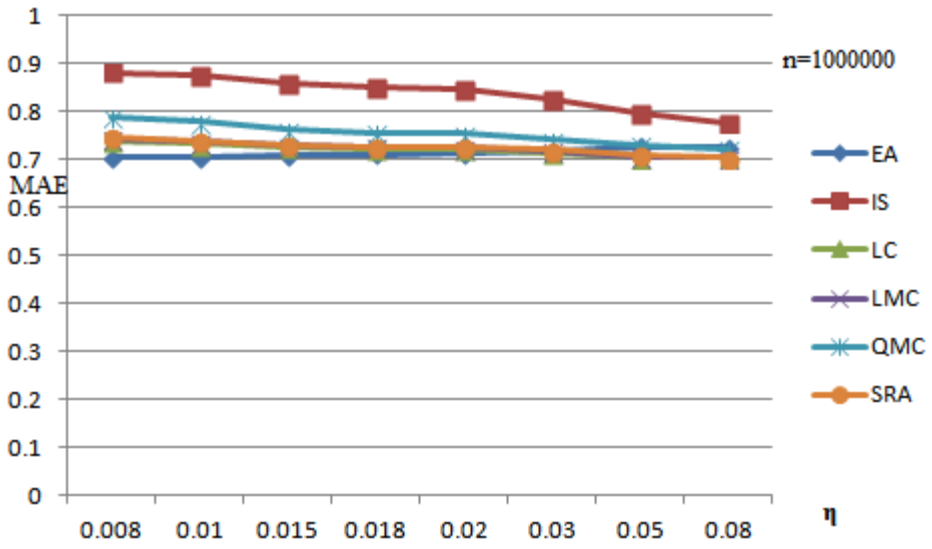


Figure 15 . MAE vs learning rate statistics for n=1000000

5.4 Experiment III: Generating recommendations with improved accuracy

As per the experiments, exponential annealing works best for lower values of learning rate and the SRA annealing schedule works best when learning rate values are higher. Thus, a better quality of recommendations can be generated using these statistics. Figure 19 shows recommendations when learning rate is 0.08 and annealing strategy used is exponential annealing. Similarly, learning rate could be varied to 0.08 and proposed annealing schedule could be used to generate a better quality of recommendations for the user, based on the movie ratings.

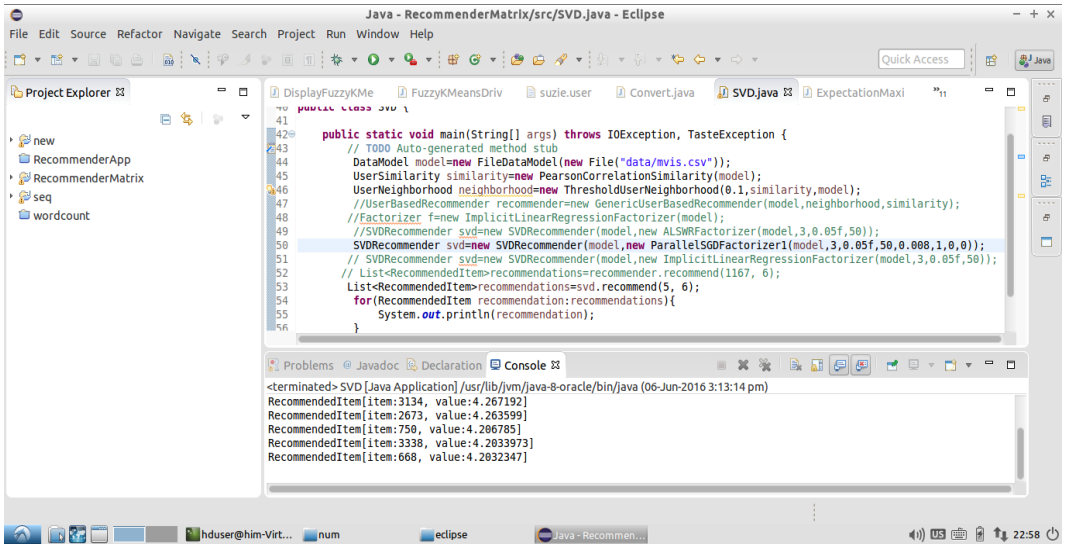


Figure 16 . Recommendations for user 5 with learning rate 0.008(exponential annealing)

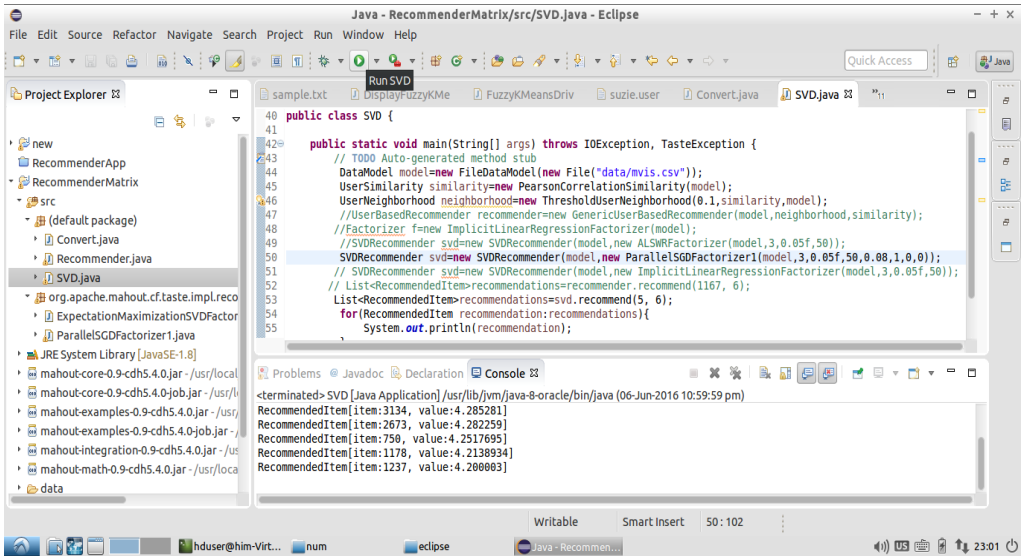


Figure 17 . Recommendations for user 5 with learning rate 0.08(SRA schedule)

Thus, simulated annealing using matrix factorization is implemented and the results are validated using mean absolute error.

These results are concluded in Chapter 6.

Chapter 6
Conclusion and Future Scope

The application of the existing simulated annealing schedules as well as proposed schedule on the recommender system helps to gauge the accuracy of the system with respect to the annealing rate. Thus simulated annealing can help to improve the accuracy of the existing system and better quality of recommendations can be generated. Also, the problem of using a sparse rating matrix in the traditional collaborative filtering systems is solved by using matrix factorization. The proposed methodology works by dividing the dataset into different sizes so that the consistency of the results is proved. This helps to test given strategies on different ranges of the dataset. Further, exponential annealing works best when learning rate is kept lower while the proposed annealing schedule outperforms other schedule when learning rate is kept on the higher side. Thus accuracy is worked upon and sparse matrix problems are reduced to a great extent using these techniques. Mahout acts as an effective platform to carry out these strategies. Further, as similarity of users is taken into account, recommendations can be generated easily based on a neighborhood of users with similar preferences.

As a future scope, more such annealing schedules could be worked out which further improve the accuracy of the recommender systems. Also, learning rate affects the cost and complexity of the system. A new approach could be found which could balance the cost expedited by the recommender system when learning rate is kept high. That means a tradeoff between the accuracy and cost of the recommender system could be done to improve results.

Further, the work could be extended to different datasets and the results could be compared based on these schedules. Simulated annealing could be applied to other domains like handling cold start or shilling attacks in recommender systems. Datasets could be extended to real time as a part of this process. In case of large datasets, Apache Mahout could be leveraged with the abilities of Apache Hadoop so that parallel processing could be done on such datasets. This would make recommender system more efficient and scalable.

REFERENCES

[1] F. Ricci, L. Rokach and B. Shapira, Recommender Systems Handbook, Springer-Verlag New York, Inc. New York, NY, USA, ISBN-038785819997803878581 97.

[2] G. Linden, B. Smith, and J. York, "Item to Item Collaborative Filtering", IEEE Computer Society, Amazon.com Recommendations, 2003, pp. 76-80.

[3] Y. Koren, R. Bell, and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems", Computer, 42(8), 2009, pp. 30-37.

[4] http://www.asis.org/SocialMedia/

[5] A. Bellogín and A. P. Vries, "Understanding similarity metrics in Neighbor based Recommender Systems", ICTIR, ACM, October 2, 2013.

[6] D.K. Bhokde, S. Girase and D. Mukhopadhyay, "An Item Based Collaborative Filtering using Dimensionality Reduction Techniques On Mahout Framework", Computing Research Repository ,**Vol abs/1503.06562**,April 2015.

[7] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Application of dimensionality reduction in recommender systems-a case study", in Proceedings of the ACM WebKDD, 2000, pp. 1-12.

[8] K. Iwahama,Yoshinori Hijikata and Shogo Nishida, "Content-Based Filtering For Music Data", in Proceedings of the 2004 International Symposium on Applications and Internet Workshops(SAINTW'04), 2004.

[9] I. Chun and I. Hong, "The implementation of knowledge-based recommender system for electronic commerce using Java expert system library", *In Proceedings of IEEE International Symposium on Industrial Electronics, 2001*, Pusan, pp. 1766-1770.

[10] I.Petrovic, P. Petrovic and I. Satjudhar, “A Profile and Community Driven Book Recommender System”, MIPRO2015, Opatija, Croatia, 25-29 May 2015, pp. 633-635.

[11] Y. Linangxing and D. Aihua, “Hybrid Product Recommender System For Apparel Retailing Customers”, in WASE International Conference on Information Engineering, 2010, pp. 356-360.

[12] R. Burke, B. Mobasher and R. Bhaumik, “Limited Knowledge Shilling Attacks in Collaborative filtering systems”, in Proceedings of Workshop Intelligent Techniques for Web Personalization (ITWP’05) 2005, pp. 1-8.

[13] K. Lam and J.Riedl, “Shilling Recommenders For Fun and Profit”, WWW2004, New York, USA, ACM, May 17-22, 2004, pp. 393-402.

[14] M. Zhang, J. Tang, X. Zhang, and X. Xue, “Addressing cold start in recommender systems: A semi-supervised co-training algorithm”, in Proceedings of the 37th International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR ’14. New York, NY, USA, ACM, 2014, pp. 73–82.

[15] J. Konstan, G. Terveen and J. Riedl, “Evaluating Collaborative Filtering Recommender Systems”, ACM Transactions on Information Systems, Vol. 22, 5-53, January 2004, pp. 5-53.

[16] G. Takacs, I. Pilaszy, B. Nemeth and D. Tikk, “Investigation of Various Matrix Factorization Methods for Large Recommender Systems”, in Proceedings of IEEE Conference on Data Mining Workshops, 2008, pp. 553-562.

[17] A. Paterek, “Improving regularized single value decomposition for collaborative filtering”, in Proceedings of the 13th ACM SIGKIDD International Conference on Knowledge Discovery and Data Mining, 2007, pp. 39-42.

[18] Y. Koren, “Factorization Meets the Neighborhood:a Multifaceted Collaborative Filtering Model”, KDD’08, Las Vegas, Nevada, USA, August 24-27, 2008, pp. 426-434.

[19] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, “Incremental Singular Value Decomposition Algorithms For Highly Scalable Recommender Systems”, in Proceedings of Fifth International Conference on Computer and Information Science, 2002, pp. 27-28.

[20] B.H. Lee, K. Q. Nguyen and R. Thawonmas, “Bounded SVD:A Matrix Factorization Method With Bound Constraints For Recommender Systems”, in Proceedings of International Conference on Emerging Information Technology and Engineering Solutions, 2015, pp. 23-26.

[21] M. Weimer, A. Karatzoglou and A. Smola, “ Improving maximum margin matrix factorization”, Machine Learning, vol. 72, no. 3, 2008, pp. 263-276.

[22] N. Srebro, J. Rennie and T. Jaakola, “Maximum Margin Matrix factorization”, Advances in Neural Information Processing Systems, vol. 17, 2004, pp.1-8.

[23] T. Hoffman,”Latent semantic models for collaborative filtering”, in ACM Transactions on Information Systems, vol. 22, no. 1, 2004, pp. 89-115.

[24] M. Kurucz, A. Benczur and K. Csalogany, “Methods for large scale SVD with missing values”, in Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery And Data Mining, California, ACM, 2007, pp. 53-56

[25] H. F. Yu, C. J. Hsieh, S. Si and I. Dhillon, “Scalable Coordinate Descent Approaches to Parallel Matrix Factorization for Recommender Systems”, in priceedings of *IEEE 12th International Conference on Data Mining*, Brussels, 2012, pp. 765-774.

[26] DeCoste, “Collaborative prediction using ensembles of maximum margin matrix factorizations”, in Proceedings of the 23rd International Conference on Machine Learning. ICML'06, 2006, New York, USA, ACM, pp. 249-256.

[27] F. Yousefian, A. Nedić, and U. Shanbhag, "On stochastic gradient and subgradient methods with adaptive steplength sequences", Automatica, vol. 48, no. 1, 2012, pp. 56-67.

[28] X. Luo, Y. Xia and Q. Zhu, "Applying the learning rate adaptation to the matrix factorization based collaborative filtering", Knowledge-Based Systems, vol. 37, 2013, pp. 154-164.

[29] L. Almeida, L., T. Langlois and J. Amaral, "Parameter Adaptation in Stochastic Optimization", Online Learning in Neural Networks, Cambridge University Press, 1998, pp. 111-134.

[30] C.S.B. Nobrega and L.B Marinho, "Predicting the Learning Rate of gradient descent for Matrix Factorization", **S**ymposium on Knowledge Discovery,Mining and Leanring,KDMiLe,2013,pp.1-8.

[31] R. Clémente, C. Cruz and C. Nicolle, "A Semantic-based Recommender System Using A Simulated Annealing Algorithm", in Proceedings of the Fourth International Conference on Advances in Semantic Processing, SEMAPRO, 2010, pp.132-137.

[32] J.Phys. A:Math. Gen.31,Printed in the UK PII:S0305-4470(98)86565-7, 1998, pp. 8373-8385.

[33] http://what-when-how.com/artificial-intelligence/a-comparison-of-cooling-schedules-for-simulated-annealing-artificial-intelligence/

[34] W.S. Chin,Y. Zhuang, Y.C. Juan and C.J. Lin , "A Learning rate Schedule For Stochastic Gradient Descent To matrix Factorization", Advances in Knowledge Discovery and Data Mining, Vol. 9077**,** pp. 442-455.

[35] Z. Ye, K. Xiao and Y. Deng, " Investigation of Simulated Annealing Cooling Schedule For Mobile Recommendations", in Proceedings of the IEEE International Conference On Data Mining Workshop(ICDMW), 2015, pp.1078-1084.

[36] G. R. Bamnote and S. S. Agrawal, "Evaluating and implementing collaborative filtering systems using apache mahout," in Proceedings of Computing Communication Control and Automation (ICCUBEA), Feb 2015, pp. 858–862.

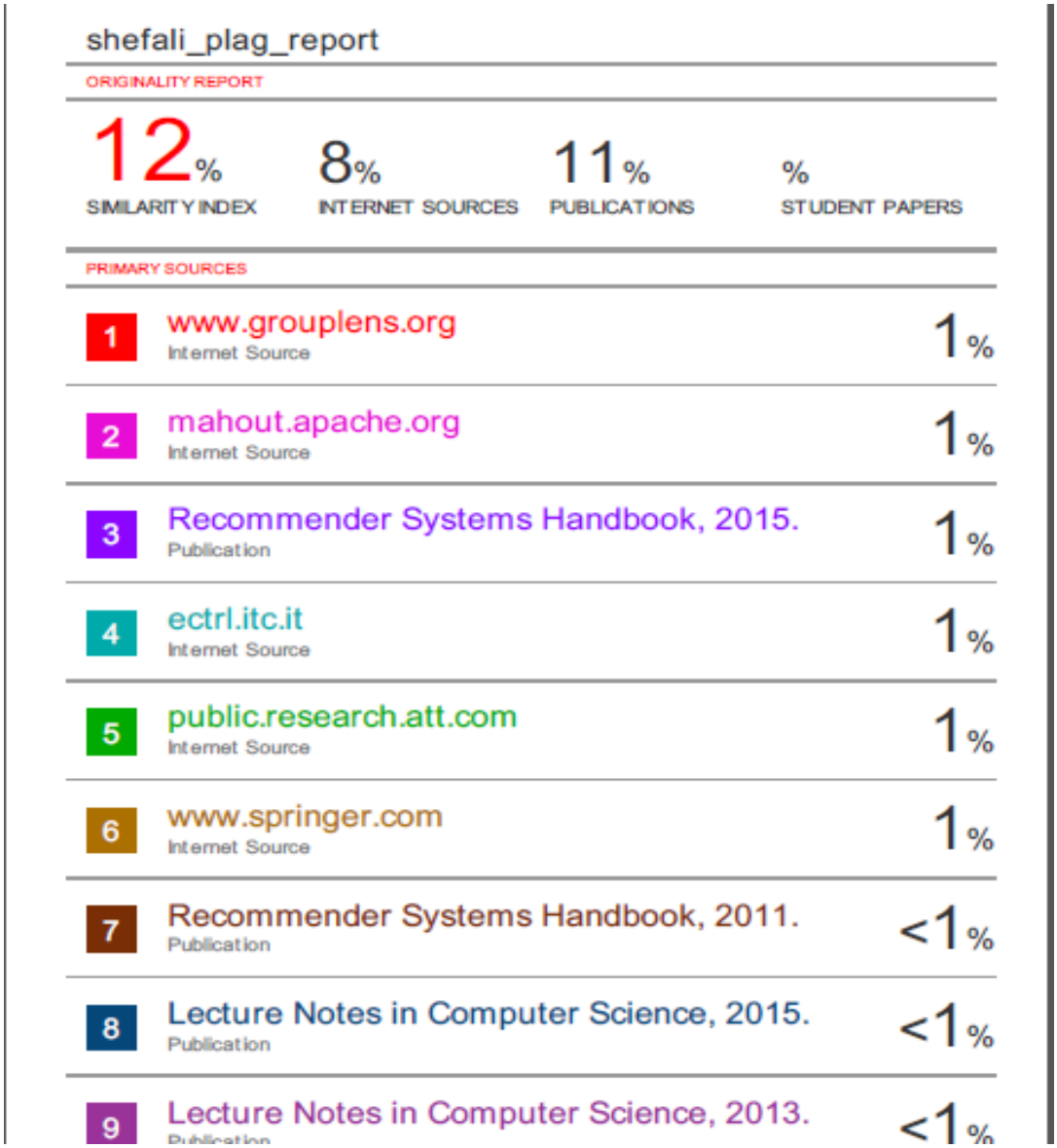
LIST OF PUBLICATIONS

- Shefali Arora, Shivani Goel, "Improving The Accuracy of Recommender Systems Through Annealing," in International Conference on ICT for Sustainable Development (ICT4SD 2016).[Accepted] to be held from Juy 1-2 at Vivanta Hotels and Resorts, Panaji,Goa.
- Shefali Arora, Shivani Goel, "Impact of Learning Rate on matrix Factorization in Recommender Systems," in Machine Learning Journal, Journal No. 10994, Published by Springer US, indexed in Science Citation Index. [Communicated]

VIDEO URL

https://www.youtube.com/channel/UC6pN-bnkoNkJbndrQyZeMJg

PLAGIARISM REPORT



shefali_plag_report

ORIGINALITY REPORT

12%	8%	11%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.grouplens.org Internet Source	1%
2	mahout.apache.org Internet Source	1%
3	Recommender Systems Handbook, 2015. Publication	1%
4	ectrl.itc.it Internet Source	1%
5	public.research.att.com Internet Source	1%
6	www.springer.com Internet Source	1%
7	Recommender Systems Handbook, 2011. Publication	<1%
8	Lecture Notes in Computer Science, 2015. Publication	<1%
9	Lecture Notes in Computer Science, 2013. Publication	<1%