**A Cloud based personalized Recommender System**

**Reshma.P 17MIS1009, Nitya Sree.P 17MIS1007, Nimisha Yadav 17MIS1183**

*School of Computing Sciences and Engineering, VIT Chennai, Tamil Nadu, India 600127*

**Abstract:**

Now-a-days, many major e-commerce Websites are using recommendation systems to provide relevantsuggestions to their customers. The recommendations could be based on various parameters, such as items popular on the company‟s Website; user characteristics such as geographical location or other demographic information; or past buying behavior of top customers. In this paper, a book recommendation engine is proposed which uses data mining techniques for recommending books, movies, songs etc The proposed recommender system will give its users‟ the ability to view and search books as well as novels which will be used to draw out conclusions about the stream of a user and the genre of the books liked by that user. The system will analyze the user behavior by using the features of various recommendation techniques such as content based; collaborative and demographic. Thus, in this paper a hybrid recommendation system is proposed which satisfies a user by providing best and efficient recommendations.

**Keywords:** Model-based Collaborative technique, memory-based Collaborative technique, Content Based Technique, Recommendation Engine,User’s Interest, deep learning, matrix factorization

1. **INTRODUCTION**

A Recommendation Engine, in actual definition can be referred to as a system that can run on clustered / non clustered environment taking user online footprint as one of its input set and generating a probable footprint for the user thereby providing its users a prediction closer to reality. Recommendation Engines require a large dataset and a fast computing system that can perform analytics on the same within fraction of seconds. Recommendation Engines, in simpler terms are programs that are data intensive and involve complex pattern matching on a set of predefined parameters and they become efficient with the increase in the size of the content being fed to them. Recommender systems represent user preferences for the purpose of suggesting items to purchase or examine. They have become fundamental applications in electronic commerce and information access, providing suggestions that effectively prune large information spaces so that users are directed toward those items that best meet their needs and preferences. A variety of techniques have been proposed till today for performing recommendations. The techniques such as content-based, collaborative, knowledge-based and demographic are used for recommendations. Sometimes, the features of these techniques are combined in hybrid recommenders to improve the performance of recommendation engine. In the proposed book recommendation engine, books will be displayed according to the readers “preferences in a hierarchical way to categorize readers” interest in different genres, the users “pattern of searching different books and to form an effective set of rules based on that”. “New books will be appropriately presented according to users‟ needs. Based on users “interest and books properties, a book recommendation system will be generating best and efficient book recommendations”. Model-based Collaborative Filtering is based on matrix factorization (MF) which has received greater exposure, mainly as an unsupervised learning method for latent variable decomposition and dimensionality reduction. Matrix factorization is widely used for recommender systems where it can deal better with scalability and sparsity than Memory-based CF

The idea of using deep learning is similar to that of Model-Based Matrix Factorization. In matrix factorization, we decompose our original sparse matrix into product of 2 low rank orthogonal matrices.

1. **Problem Statement:**

The main module in this system is recommender. The registers logs in to the system the user can view the books of different categories and history of books. Thus, in this paper a hybrid recommendation system is proposed which satisfies a user by providing best and efficient books recommendations.

1. **Existing System**

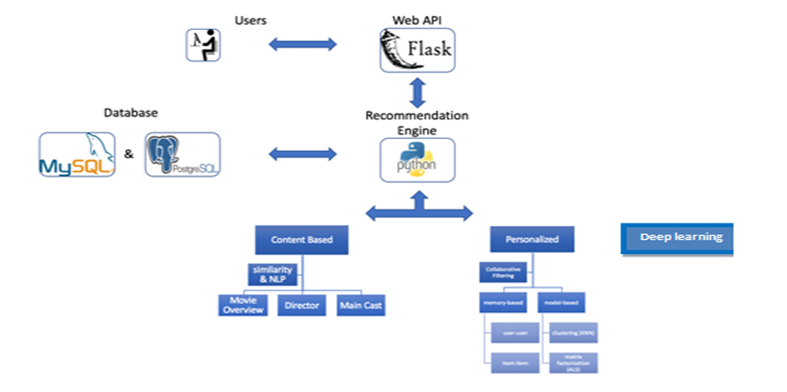
Following are some of the existing book recommendation engines used by the top rated book purchasing websites. The existing engines make use of conventional algorithms for recommendations.

In Content based Recommendation Engine, system generates recommendations from source based on the features associated with products and the user’s information. Content-based recommendation as a user-specific classification problem and learn a classifier for the user's likes and dislikes based on product features.

In Collaborative recommendation engines, suggestions are generated on the basis of ratings given by group of people. It locates peer users with a rating history similar to the current user and generates recommendations for the user.

In Context based Recommendation Engine, system requires the additional data about the context of item consumption like time, mood and behavioral aspects. These data may be used to improve the recommendation compared to what could be performed without this additional source of information.

1. **Proposed System:**

In above figure, the architecture of proposed system is shown. The main module in this system is Recommender system. The registered user logs in to the system. The user can view books of different categories. The user can also rate books as per his/her likings. The rating and searching history of books for each individual is stored in the database. In recommender system module, mainly three techniques are used for recommendations. Collaborative based filtering and content based filtering techniques are performed on the data which is present in user’s history. If null results are generated from these techniques then demographic recommender is used. The results from all the recommender techniques are combined and the set for recommended books is generated.

1. **Experimental setup:**

The input to our system are API requests, which can be classified as online or batch:

* Online requests, which must be handled in real time. Their processing can’t be delayed, because users are waiting for a response. They are also utilized to update the profile for new users and begin to provide them with recommendations.
* The request processor evaluates if a certain API request needs to be run online or it can be batched.
* Batch requests, which may be stored and processed only at given time periods. Now requests processing can be delayed and attended when the system is not at full capacity. These requests are used to upload the initial data from a client and also to update information concerning users with a wide user profile. Each API request generates an HTTP request to a certain end-point where the Request Processor evaluates it and deter-mines whether it must be processed at that moment or it can be delayed until more requests reach the system (for amore optimum processing) or until certain batch process is programmed to be run. Requests can also be classified as update or retrieval
* Retrieval requests just ask the system to return some kind of information, such as a recommendation
* Update requests have the objective to update the pro-file of the source user. When an update request begins to be processed there are two steps that must be taken to produce recommendations for the user. As we wanted to be able to process several types of recommendations (collaborative filtering, content-based, social recommendations), the system had to be general enough to process data in several ways. So we defined those steps in a way that enabled the use of any possible recommender algorithm Update user profile. This can be done by recalculating similarity with other users, re-calculating trust or updating a content-based profile.
* Update user recommendations. This step uses the values obtained from the previous task as input for the recommendation algorithm and produces a new rank of recommendations for the user.

1. **Techniques Used**

Recommendation techniques have a number of possible classifications. The classification is based on the sources of data on which recommendation is based and the use to which that data is put. In general, recommender systems have

1. background data, the information that the system has before the recommendation process begins,
2. input data, the information that user must communicate to the system in order to generate a recommendation, and
3. an algorithm that combines background and input data to arrive at its suggestions.

The recommendation techniques are classified into five types:

1] Collaborative.

2] Content based.

3] Demographic.

4] Utility based.

5] Knowledge

6] matrix factorization

7] deep learning based.

In this paper, for the proposed book recommendation engine collaborative, content-based and

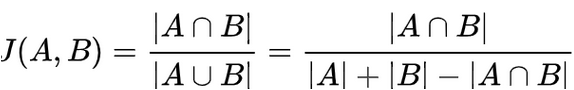
Matrix factory and deep learning are used.

***6.1 Content Based Technique***

This approach relies on creating a plethora of parameters to describe a product “P”. Considering a smart phone as an example the possible parameters could be screen size, image quality, Wi-Fi protocols, brand names, operating systems etc. The larger the parameter set the better and easier it is to match patterns with user profile and his online footprint. The parameters can then be assigned weights and hence a relative priority is set for each of the parameter. All these parameters are then used to create a user profile and each time a prospective user checks out another product, his profile gets updated. Hence we see that the system learns about the user preferences and selection patters by his online footprint. Popular platforms that use such an approach are IMDB and Pandora.

For Content based technique, Locality-sensitive hashing method is used. Locality-sensitive hashing (LSH) is a method of performing probabilistic dimension reduction of high-dimensional data. The basic idea is to hash the input items so that similar items are mapped to the same buckets with high probability. In LSH the goal is to maximize probability of “collision” of similar items. Jaccard similarity is used along with LSH method.

The Math



**6.2** **Context Based Recommender system**

|  |  |  |
| --- | --- | --- |
| **Method** | **Description** | **References** |
| **Hidden Markov Model** | Improved version of a location recommender system by implementing Decision Tree (DT) along with discrete Hidden Markov Model (HMM).Together HMM and DT differentiate between transport modes and reduce noise. | Norma Saiph Savage Maciej Baranski Norma Elva Chavez Tobias Höllerer[34] |
| **Multidimensional approach** | It provides additional contextual information on user and item, and also supports multiple Dimensions, profiling information, and hierarchical aggregation of recommender system. | Gediminas Adomavicius Ramesh Sankaranarayanan Shahana Sen Alexander Tuzhilin [31] |
| **Fuzzy Bayesian Networks** | It gives a recommender system which exploits the fuzzy system, Bayesian Networks in order to get appropriate recommendation with respect to the context. | Han-SaemPark,Ji-Oh YooSung-Bae Cho 34 |
| **Human memory model** | A recommender system is proposed which retrieves relevant preference information from long term memory and uses it in conjunction with the information stored in short term memory. | Sarabjot Singh Anand and BamshadMobasher [32] |
| **Matrix-factorization Predictive Context based Mod** | Distributional-Semantics Prefiltering (DSPF) approach is used to build more precise context aware rating prediction models, by exploiting, in a novel way, the distributional Semantics of contextual conditions. It also shows how DSPF can be improved by using clustering techniques. | Víctor Codina FrancescoRicci LuigiCeccaroni [35] |

***6.3 Model based collaborative:***

Model-based CF Complex patters which are based on training data, are the models (such as data mining algorithms, machine learning) and then intelligent predictions are made for CF tasks for the real world data which are based on learnt models. It intuitive rationale for recommendations. Model disadvantage of model-based CF is that it loses useful information for dimensionality reduction techniques .

Model-based collaborative filtering :

The main drawback of memory-based technique is the requirement of loading a large amount of in-line memory. The problem is serious when rating matrix becomes so huge in situation that there are extremely many persons using system. Computational resource is consumed much and system performance goes down; so system can’t respond user request immediately. Model-based approach intends to solve such problems. There are four common approaches for model-based CF such as clustering, classification, latent model, Markov decision process (MDP), and matrix factorization.

***6.3.1. Clustering CF :***

Clustering CF is based on assumption that users in the same group have the same interest; so they rate items similarly. Therefore users are partitioned into groups called clusters which is defined as a set of similar users. Suppose each user is represented as rating vector denoted ui = (ri1, ri2,…, rin). The dissimilarity measure between two users is the distance between them. We can use Minkowski distance, Euclidian distance or Manhattan distance.

The less distance (u1, u2) is, the more similar u1 and u2 are. Clustering CF includes two steps:

1. Partitioning users into clusters and each cluster always contains rating values. For example, every cluster resulted from k-mean algorithm has a mean which is a rating vector like user vector.

2. The concerned user who needs to be recommended is assigned to concrete cluster and her/his ratings are the same to ratings of such cluster. Of course how to assign a user to right cluster is based on the distance between user and cluster.

So the most important step is how to partition users into clusters. There are many clustering techniques such as k-mean and k-centroid. The most popular clustering algorithm is k-mean algorithm which includes three following steps:

1. It randomly selects k users, each of which initially represents a cluster mean. Of course, we have k cluster means. Each mean is considered as the “representative” of one cluster. There are k clusters.

2. For each user, the distance between it and k cluster means are computed. Such user belongs to the cluster to which it is nearest. In other words, if user ui belong to cluster cv, the distance between ui and mean mv of cluster cv, denoted distance(ui, mv), is minimal over all clusters.

3. After that, the means of all clusters are re-computed. If stopping condition is met then algorithm is terminated, otherwise returning step 2.

This process is repeated until the stopping condition is met. There are two typical terminating conditions (stopping conditions) for k-mean algorithm:

- The k means are not changed. In other words, k clusters are not changed. This condition indicates a perfect clustering task.

- Alternatively, error criterion is less than a pre-defined threshold.

***6.3.2. Matrix Factorization***

In the previous attempt, I have used memory-based collaborative filtering to make movie recommendations from users’ ratings data. I can only try them on a very small data sample (20,000 ratings), and ended up getting pretty high Root Mean Squared Error (bad recommendations). Memory-based collaborative filtering approaches that compute distance relationships between items or users have these two major issues:

It doesn’t scale particularly well to massive datasets, especially for real-time recommendations based on user behavior similarities — which takes a lot of computations.

Ratings matrices may be overfitting to noisy representations of user tastes and preferences. When we use distance based “neighborhood” approaches on raw data, we match to sparse low-level details that we assume represent the user’s preference vector instead of the vector itself.Thus I’d need to apply Dimensionality Reduction technique to derive the tastes and preferences from the raw data, otherwise known as doing low-rank matrix factorization. Why reduce dimensions?

I can discover hidden correlations / features in the raw data.

I can remove redundant and noisy features that are not useful.

I can interpret and visualize the data easier.

I can also access easier data storage and processing.

The Math

Model-based Collaborative Filtering is based on matrix factorization (MF) which has received greater exposure, mainly as an unsupervised learning method for latent variable decomposition and dimensionality reduction. Matrix factorization is widely used for recommender systems where it can deal better with scalability and sparsity than Memory-based CF:

The goal of MF is to learn the latent preferences of users and the latent attributes of items from known ratings (learn features that describe the characteristics of ratings) to then predict the unknown ratings through the dot product of the latent features of users and items.

When you have a very sparse matrix, with a lot of dimensions, by doing matrix factorization, you can restructure the user-item matrix into low-rank structure, and you can represent the matrix by the multiplication of two low-rank matrices, where the rows contain the latent vector.

You fit this matrix to approximate your original matrix, as closely as possible, by multiplying the low-rank matrices together, which fills in the entries missing in the original matrix.

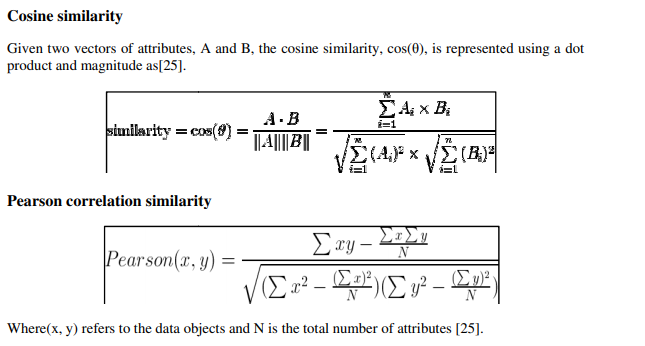
A well-known matrix factorization method is Singular value decomposition (SVD). At a high level, SVD is an algorithm that decomposes a matrix A into the best lower rank (i.e. smaller/simpler) approximation of the original matrix A. Mathematically, it decomposes A into a two unitary matrices and a diagonal matrix:where A is the input data matrix (users’s ratings), U is the left singular vectors (user “features” matrix), Sum is the diagonal matrix of singular values (essentially weights/strengths of each concept), and V^T is the right singular vectors (movie “features” matrix). U and V^T are column orthonormal, and represent different things: U represents how much users “like” each feature and V^T represents how relevant each feature is to each movie.To get the lower rank approximation, I take these matrices and keep only the top k features, which can be thought of as the underlying tastes and preferences vectors.

The Evaluation for loss function



**6.3.3. Memory based collaborative:**

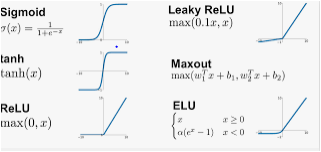
A memory-based CF (nearest-neighbor) approach, mostly called as a form of implementation of the “Word of Mouth” phenomenon (Jin, Chai & Si, 2004) since the entire user database with their preferences are kept in memory. For each prediction computation is performed K. Madadipouya, S. Chelliah - A Literature Review on Recommender Systems Algorithms, Techniques and Evaluations 113 on the whole database. This method could predict a user interests on a specific item based on the rating information of similar user profiles. It reflects where the prediction of a specific item (belonging to a specific user) is done by sorting the row vectors (user profiles) by its dissimilarity toward the user. In this method, more rating by more similar users leads to more rating prediction. Various types of memory-based recommender systems have been developed. Decker and Lenz (2007) stated that Goldberg on 1992 developed certain type of memory-based CF system which is called Tapestry. This approach mostly is used in information retrieval systems. Apart from developments which have been done by researchers, some commercial websites also have developed their own version of memory-based collaborative filtering



* 1. ***Deep Learning***

The idea of using deep learning is similar to that of Model-Based Matrix Factorization. In matrix factorization, we decompose our original sparse matrix into product of 2 low rank orthogonal matrices. For deep learning implementation, we don’t need them to be orthogonal, we want our model to learn the values of embedding matrix itself. The user latent features and movie latent features are looked up from the embedding matrices for specific movie-user combination. These are the input values for further linear and non-linear layers. We can pass this input to multiple relu, linear or sigmoid layers and learn the corresponding weights by any optimization algorithm (Adam, SGD, etc.).

This model performed better than all the approaches I attempted before (content-based, user-item similarity collaborative filtering, SVD). I can certainly improve this model’s performance by making it deeper with more linear and non-linear layers.



1. **Results and discussion:**

We first started off with context based model, then proceeeded with model and memory based collaborative method finally we performed deep learning method. The accuracy of deep learning method was the highest

1. **Conclusion and Future Scope:**

Recommender systems are an extremely potent tool utilized to assist the selection process easier for users. The implemented book recommendation engine is a competent system to recommend for e-users. This recommender system will definitely be a great web application implemented in python language. Such type of web application will be proved beneficial for today‟s high demanding online purchasing websites. This hybrid recommender system is more accurate and efficient as it combines the features of various recommendation techniques. The book recommendation engine will reduce the overhead associated with making the best choices of books among the plenty.

**9. References:**

G. Adomavicius and A. Tuzhilin, “Toward the next generation of recommender systems: A survey of the state-of the-art and possible extensions,” IEEE Trans. Knowl. Data Eng.

G. Linden, B. Smith, and J. York, “Amazon recommendations:Itemto- item collaborative filtering,” IEEE Internet Comput., Feb. 2003.

Michael Hashler, “Recommender Lab: A Framework for Developing and Testing Recommendation Algorithms” Nov. 2011.

R. Bell, Y. Koren, and C. Volinsky, “Modeling relationships at multiple scales to improve accuracy of large recommender systems”KDD '07: Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, New York, NY, USA, 2007, ACM.

O. Celma and P. Herrera, “A new approach to evaluating novel recommendations”, RecSys '08: Proceedings of the 2008 ACM conference on Recommender systems, New York, NY, USA, 2008, ACM.

C. N. Ziegler, S. M. McNee, J. A. Konstan, and G. Lausen, “Improving recommendation lists through topic diversification”: Proceedings of the 14th international conference on World Wide Web, New York, USA, 2005, ACM.

Robin Burke, “Hybrid Recommender Systems: Survey and Experiments”, California State University, Fullerton Department of Information Systems and Decision Sciences.

Dharmendra Pathak, Sandeep Matharia and C. N. S. Murthy, “NOVA: Hybrid Book Recommendation Engine”, IEEE, 2012.

Ahu Sieg, Bamshad Mobasher, Robin Burke, “Ontology-Based Collaborative Recommendation”

**10.Reference Journal links:**

https://ieeexplore.ieee.org/document/8696033

https://ieeexplore.ieee.org/document/6779375

https://ieeexplore.ieee.org/document/7435717

https://ieeexplore.ieee.org/document/8399650

https://ieeexplore.ieee.org/document/8391175