# Review: Recommendation System Algorithms

Charvi Bannur
*Dept. of Computer Science*
*PES University*
Bangalore, India
charvibannur@gmail.com

Chaitra Bhat
*Dept. of Computer Science*
*PES University*
Bangalore, India
chaitrabhat084@gmail.com

Nireeksha D Rai
*Dept. of Computer Science*
*PES University*
Bangalore, India
nireekshadrai@gmail.com

***Abstract*—This work is a compilation of state of the art algorithms used in recommendatin systems, followed by explanation and exhausive literature survey.**

***Index Terms*—Recommendation Systems, Apriori Algorithms, Matrix Factorisation, TF-IDF (Term Frequency-Inverse Document Frequency)**

## I. Problem Statement

Extensive Review of state-of-the-art recommendation algorithms and analysis for novel applications i.e Analysis and Study of different algorithms that enable various recommendation systems to function. Further extend the algorithms for other applications.

## II. Motivation

Recommender Systems represent one of the most widespread and impactful applications of predictive machine learning algorithms. It is able to generate more relevant recommendations tailored to the tastes of the user. One of the crucial components behind the working of a product recommendation engine is the recommender algorithm, which we would like to explore.

## III. Related Work

literature review for the following algorithms are neatly tabulated in tables 1, 2, 3.

Table 1. Literature Review for Apriori Algorithm

| Author and Citation | Results and Contribution |
|---|---|
| Mohammed Al-Maolegi, Bassam Arkok [1] | scanning some frequent transactions instead of all of them in order to optimize the time constraint. Time is reduced by 67.38 %. Does not tackle the space complexity problem. |
| Jiao Yabing [2] | Improved association rules detection using pruning of subsets. Increases efficiency in large datasets considerably. Increases the time required. |
| Ning Li, Li Zeng, Qing He and Zhongzhi Shi [3] | framework for processing huge datasets on certain kinds of distributable problems using a large number of computers (nodes). proposed algorithm can scale well and efficiently process large datasets on commodity hardware with 78% scalability |

Table 2. Literature Review for Matrix Factorization

| Author and Citation | Results and Contribution |
|---|---|
| Pavan Manoj Rathod, Rajkumar K Shende [4] | Input Matrix and Time Taken In Seconds By Each of Naive, Strassen's and Proposed Algorithm Comparative Analysis. Proposed Algorithm helps to decrease the complexity of matrix multiplication to a greater extent. |
| Xin Luo et al. [5] | Development of an NMF-based(Non-negative Matrix Factorization) CF(Collaborative Filtering) model. With the single-element-based approach, integrate the Tikhonov regularizing terms to improve the prediction accuracy. |
| Mohamed Hussein Abdi et al. [6] | This work provided a broad overview of available approaches to incorporate contextual information into Collaborative Filtering based Recommender Systems utilizing Matrix Factorization techniques. Matrix/Tensor Factorization models could be optimized by using Stochastic Gradient Descent (SGD), Alternating Least Squares (ALS), or Markov Chain Monte Carlo Inference (MCMC) |

Table 3. Literature Review for TF-IDF

| Author and Citation | Results and Contribution |
|---|---|
| Mir Saman Tajbakhsh et al. [7] | Weighs each document according to two factors: TF-IDF and a semantic similarity measure.The simple TF-IDF weighting schema with cosine similarity has better results in precision and recall measures . |
| Dhongui Wang et al. [8] | Feature vector space is generated in feature selection module and feature vectors are used to train softmax regressor in softmax regression module.Using IG and chi-square selection method, the accuracy increases with the feature number |
| Mohamed Chiny et al. [9] | Set up a system of emission recommendations based on two techniques derived from NLP, which are TF-IDF and Cosine Similarity. Ranking of Countries According to the Number of Emissions Available, ranking of program Categories Proposed by Netflix |

## IV. Our Approach

The code for the implementation is available on Github[1]

### A. Apriori Algorithm: A Unique Approach

The Apriori algorithm is used for mining frequent itemsets and devising association rules from a transactional database to mainly solve the Market Basket Analysis Problem. Retailers use market basket analysis, a data mining technique, to increase sales by better understanding customer purchasing patterns. It entails analysing large data sets, such as purchase history, to identify product groups and products that are likely to be purchased together.

The apriori algorithm is the algorithm used to compute the association rules between objects. It denotes the relationship between two or more objects. In other words, the apriori algorithm is an association rule learning algorithm that analyses whether people who purchased product A also purchased product B. The apriori algorithm's primary goal is to generate the association rule between various objects. The association rule describes the relationship between two or more objects. The Apriori algorithm is also referred to as frequent pattern mining. In general, you run the Apriori algorithm on a database with a large number of transactions.

Some of the important formula used in this algorithm are:

- Support: Support refers to the default popularity of any product. You find the support as a quotient of the division of the number of transactions comprising that product by the total number of transactions.
- Confidence: Customers' confidence refers to the possibility that they purchased both biscuits and chocolates at the same time. To calculate the confidence, divide the number of transactions that include both biscuits and chocolates by the total number of transactions.
- Lift: Consider the above example; lift refers to the increase in the ratio of the sale of chocolates when you sell biscuits. The mathematical equations of lift are given below.

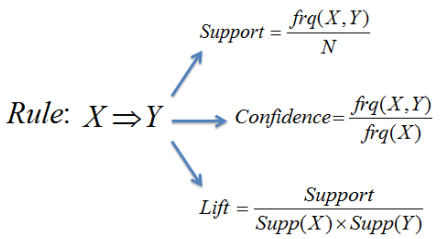


Fig. 1. Important Algorithm for Apriori Algorithm

Methodology to implement the Apriori Algorithm is given below:

- Create a table containing support count of each item present in dataset – Called C1(candidate set). Compare candidate set item's support count with minimum support count(here min_support=2 if support_count of candidate set items is less than min_support then remove those items). This gives us itemset L1.
- Generate candidate set C2 using L1 (this is called join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common.Check all subsets of an itemset are frequent or not and if not frequent remove that itemset.(Example subset ofI1, I2 are I1, I2 they are frequent.Check for each itemset) Now find support count of these itemsets by searching in dataset.
- compare candidate (C2) support count with minimum support count(here min_support=2 if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L2.
- Generate candidate set C3 using L2 (join step). Condition of joining Lk-1 and Lk-1 is that it should have (K-2) elements in common. So here, for L2, first element should match. Check if all subsets of these itemsets are frequent or not and if not, then remove that itemset. Find support count of these remaining itemset by searching in dataset. Compare candidate (C3) support count with minimum support count(if support_count of candidate set item is less than min_support then remove those items) this gives us itemset L3.
- Generate candidate set C4 using L3 (join step). Condition of joining Lk-1 and Lk-1 (K=4) is that, they should have (K-2) elements in common. So here, for L3, first 2 elements (items) should match.Check all subsets of these itemsets are frequent or not. We stop here because no frequent itemsets are found further.
- Thus, we have discovered all the frequent item-sets. Now generation of strong association rule comes into picture. For that we need to calculate confidence of each rule. So if minimum confidence is 50%, then first 3 rules can be considered as strong association rules.

**Stats**

The time complexity is O(2d̂) and space complexity is also O(2d̂) where d is the total number of unique items.

### B. Matrix Factorization Algorithm

Matrix factorization is a technique for generating latent characteristics by multiplying two different types of things. Collaborative filtering is the use of matrix factorization to detect the link between the entities of things and users.With the input of user ratings, forecast how the users would score the remaining products so that users may receive recommendations based on the projection.

Assume we have a ranking table of 5 people and 5 movies, with ratings ranging from 1 to 5, and the matrix is supplied by the table. Because not every user rates every movie, the matrix has many missing values, resulting in a sparse matrix. As a result, any null values not provided by the users are filled with 0 so that the filled values are available for multiplication. For

[1]https://github.com/charvibannur/Advanced-Algorithm-Project

example, two individuals give a high rating to a movie when it is acted by their favourite actor and actress, or when the movie genre is action etc. As a result of the matrix factorization, we may find these latent traits and predict a rating based on the similarity in user preferences and interactions.

The rating matrix may be generated by taking the dot product of the user and item matrices, where the user matrix has the form of k (users) * f (features) and the item matrix has the shape of j(items) * f (features) (features).

- User Matrix - Matrix representing the various users.
- Item Matrix - Matrix representing the score of various items as assigned by the user.
- Rating Matrix - By performing the dot product of the user matrix and item matrix, the rating matrix would be generated.

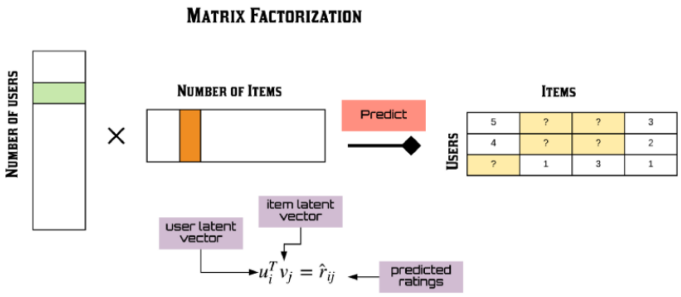


Fig. 2. Matrix Representation Algorithm

The matrix factorization of user and item matrices can be generated when the math cost function RMSE is minimized through matrix factorization. Following the above mathematic concept, gradient descent is one of the methods to minimize RMSE through each iteration. The algorithm learns the embeddings between the users without having to tune the features. Matrix Factorization is the most common technique to find the embeddings or features that make up the interest of a particular user.

In practise, the rating matrix is relatively sparse since each user views movies at a different frequency. The error function RMSE, on the other hand, is only calculated given a non-zero rating. The dot product of the factor matrices would be used to fill in the missing elements in the rating matrix. As a result, we know what unseen movies to recommend to consumers based on the forecast.

To summarize the following steps are followed:

- Initialization of the random, user and item matrix.
- Calculate the Error Function.
- Train and Calculate using Stochastic Gradient Descent.
- Tweak the values in user and item matrix accordingly to get the accurate user and item matrix which can further be used to recommend.

**Stats**

Naive solver: O(mnk)
Efficient algorithm: $O(Ak + mk^2 + nk^2)$. where A is the matrix which has m users and n movies. k is the hidden dimension.

### C. TF-IDF Algorithm

TF-IDF is an algorithm that uses the frequency of words to determine how relevant those words are to a given document. Content based recommenders will use data exclusively about the items. For this we need to have a minimal understanding of the users' preferences, so that we can then recommend new items with similar tags/keywords to those specified (or inferred) by the user.

TF-IDF vectorization involves calculating the TF-IDF score for every word in corpus relative to that document and then putting that information into a vector. Thus each document in corpus would have its own vector, and the vector would have a TF-IDF score for every single word in the entire collection of documents. Once we have these vectors we can apply them to various use cases such as seeing if two documents are similar by comparing their TF-IDF vector using cosine similarity.

$$w_{i,j} = tf_{i,j} \cdot log\left(\frac{N}{df_i}\right)$$

$tf_{i,j}$ = number of occurrences of $i$ in $j$

$df_{i,j}$ = number of documents

$N$ = total number of documents

Fig. 3. Computation of TF-IDF

The TF-IDF algorithm is used to evaluate the importance of words in a textual corpus. The importance is proportional to the number of times the words appear in the document and inversely proportional to the frequency of words appearing in the corpus

- TF represents the frequency of words, indicating the number of times they appear in a corpus.This consists of calculating the number of word appearances out of the total number of words present in the corpus.
- IDF is the measure of the importance of the term in the corpus as a whole. It consists in calculating the logarithm of the inverse of the proportion of documents in the corpus that contain the term
- The weight of TF-IDF is calculated by multiplying the two measures (tfi * idfi), so the greater the weight, the more significant the word concerned in the corpus.

As mentioned already, TF-IDF is a weighting schema for text terms. For a given term in a document, TF-IDF computes the term frequency in the document as well as the number of documents which contain that term, as a criteria to weigh the

term. After calculating the weights of all term of a document, that document is represented as a vector of term-weights.

In content based movie recommendation system tf-idf will help capture the important genres of each movie by giving a higher weight to the less frequent genres.While this approach is more commonly used on a text corpus, it possesses some interesting properties that will be useful in order to obtain a vector representation of the data.

### Stats

The computational complexity of TF-IDF is O(nL log nL), where n is total number of sequences in a dataset, and L the average length of sequences in a dataset.

## V. Conclusion

The study analyses the outcomes of several recommendation algorithms, as well as in-depth understanding of how these algorithms work and forecast user preferences or goods that are likely to be of interest to them. It also covers potential future work, such as the extension of each method to different applications.

## VI. Future Work

**Novel Application: Data Corruption Detection**

- One of the unique approaches that is yet to be explored is the detection of corrupted data in poisoned datasets. We can make use of this algorithm to assess the correctness of values in a categorical dataset.
- Each dimension of the dataset is considered as a unique item and the association between multiple dimensions can be calculated resulting in a unique outlier/anomaly score or threshold to detect if the data was corrupted or not.
- The Apriori Algorithm is typically used to assess how closely two values in a large dataset are related. It is predicated on the idea that values with more disparities between them are less likely to be seen as significant than values with less in common.
- Thus the Apriori Algorithm might be more fit for categorical dataset with more than one dimensions rather than a low dimensional numeric dataset.
- Such an application for this algorithm has not been worked on yet and is something I am working on currently.

**Novel Application: Plagiarism Detection**

- One of the unique approaches that is yet to be explored is the plagiarism detection of text or code. We can make use of this algorithm to assess the correctness of values in a particular code or dataset.
- Each dimension of the dataset is considered as a unique item and the association between multiple dimensions can be used to generate some kind of similarity index which can further determine and compare whether two or more code snippets are similar or not.
- The Algorithm is typically used to assess how closely values are related. It is predicated on the idea that values with more disparities between them are less likely to be seen as significant than values with less in common.
- Such an application for this algorithm has not been worked on yet and is something I am working on currently.

**Novel Application: Datasets from different platforms**

- One of the unique approaches that is yet to be explored is dataset used from different platforms like hotstar, voot etc.
- Different unexplored features of dataset like actors of the movie could also be explored as part of novel application.
- TF-IDF is a statistical measure used to determine the mathematical significance of words in documents
- Such an application for this algorithm has not been worked on yet and is something I am working on currently.

## References


[1] Al-Maolegi, Mohammed, and Bassam Arkok. "An improved Apriori algorithm for association rules." arXiv preprint arXiv:1403.3948 (2014).

[2] Yabing, Jiao. "Research of an improved apriori algorithm in data mining association rules." International Journal of Computer and Communication Engineering 2, no. 1 (2013): 25.

[3] Li, Ning, Li Zeng, Qing He, and Zhongzhi Shi. "Parallel implementation of apriori algorithm based on mapreduce." In 2012 13th ACIS international conference on software engineering, artificial intelligence, networking and parallel/distributed computing, pp. 236-241. IEEE, 2012.

[4] P. M. Rathod and R. K. Shende, "Recommendation System using optimized Matrix Multiplication Algorithm," 2020 IEEE International Symposium on Sustainable Energy, Signal Processing and Cyber Security (iSSSC), 2020, pp. 1-4, doi: 10.1109/iSSSC50941.2020.9358891.

[5] Folasade Olubusola Isinkaye (2021) Matrix Factorization in Recommender Systems: Algorithms, Applications, and Peculiar Challenges, IETE Journal of Research, DOI: 10.1080/03772063.2021.1997357

[6] Abdi, Mohamed Hussein, George Okeyo, and Ronald Waweru Mwangi. "Matrix factorization techniques for context-aware collaborative filtering recommender systems: A survey." (2018).

[7] Tajbakhsh, Mir Saman, and Jamshid Bagherzadeh. "Microblogging hash tag recommendation system based on semantic TF-IDF: Twitter use case." In 2016 IEEE 4th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW), pp. 252-257. IEEE, 2016.

[8] Wang, Donghui, Yanchun Liang, Dong Xu, Xiaoyue Feng, and Renchu Guan. "A content-based recommender system for computer science publications." Knowledge-Based Systems 157 (2018): 1-9.

[9] Chiny, Mohamed, Marouane Chihab, Omar Bencharef, and Younes Chihab. "Netflix Recommendation System based on TF-IDF and Cosine Similarity Algorithms." no. Bml (2022): 15-20.