ISSN No:-2456-2165

# Travel Companion

Vishal Israni[1], Raaj Raisinghani[2], Varun Rathi[3]
Student, Department of Information Technology
Vivekanand Education Society's Institute of Technology, Mumbai

Charusheela Nehete[4]
Assistant Professor, Department of Information Technology
Vivekanand Education Society's Institute of Technology, Mumbai

**Abstract:- This paper presents a generalized and effective methodology for recommending various entertainment parameters which can be very helpful while traveling using various AI and ML algorithms. A product that recommends the user books, movies, TV shows, songs and places to visit based on their past preferences and the time they have at their disposal. Use of content based, collaborative filtering, hybrid algorithms and demographic based recommender systems to filter the results and recommend them to the user. An additional feature of filtering results based on time constraint is also implemented. Blockchain-based micropayments and other features such as Proof of work and Proof of authority are used for pay per use feature and OAuth will ensure the user's authentication.**

***Keywords:-*** *Blockchain, Machine Learning, Hybrid Filtering, Content-Based Filtering, Collaborative Filtering, Demographic-Based Filtering, Proof of Work, Proof of Authority.*

## I. INTRODUCTION

The current recommendation systems are quite specific in their usage and there is a lack of a good application that suffices all the user's needs of a recommendation system all integrated in one platform.

The portal is aimed at providing the user with entertainment which the user will tend to like. The time constraint feature aids the user in selecting relevant content without fussing over irrelevant recommendations. This application will ensure the social well-being of an individual. Various domains covered are Movies, TV shows, books, Songs and places to visit. The product intends to deliver the user, the most relevant recommendations through a self-explanatory and simplistic Graphical User Interface, along with the liberty to pay only for what the user really wants, secured by the robust blockchain.

### *A. Collaborative Filtering*

Collaborative filtering, also referred to as social filtering, filters information by using the recommendations of other people. It is based on the idea that people who agreed in their evaluation of certain items in the past are likely to agree again in the future. A person who wants to see a movie for example, might ask for recommendations from friends. The recommendations of some friends who have similar interests are trusted more than recommendations from others. This information is used in the decision on which movie to see.[1] The algorithm calculates the similarity between two users or items, and produces a prediction for the user by taking the weighted average of all the ratings. Similarity computation between items or users is an important part of this approach. Multiple measures, such as Pearson correlation and vector cosine based similarity are used for this. The Pearson correlation similarity of two users x, y is defined as

$$\mathrm{simil}(x,y) = \frac{\sum\limits_{i \in I_{xy}} (r_{x,i} - \bar{r_x})(r_{y,i} - \bar{r_y})}{\sqrt{\sum\limits_{i \in I_{xy}} (r_{x,i} - \bar{r_x})^2} \sqrt{\sum\limits_{i \in I_{xy}} (r_{y,i} - \bar{r_y})^2}}$$

Equation 1.1 : Pearson Correlation Similarity

where $I_{xy}$ is the set of items rated by both user *x* and user *y*.[2]

### *B. Content Based Filtering*

Content-based filtering, also referred to as cognitive filtering, recommends items based on a comparison between the content of the items and a user profile. The content of each item is represented as a set of descriptors or terms, typically the words that occur in a document. The user profile is represented with the same terms and built up by analyzing the content of items which have been seen by the user. [3] The algorithm makes use of cosine similarity and finds the relation between the user and the item in consideration.

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\| \|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Equation 1.2 : User and Item Correlation

### *C. Hybrid Recommender Systems*

Both, content based and collaborative filtering have strengths and weaknesses. Four specific problems can be distinguished for collaborative filtering :
Cold start, Sparsity, First rater, popularity bias.

Whereas, Content based filtering faces issues with :
Content description, Over specialization, subjective problem domain [4].

A system that combines content-based filtering and collaborative filtering could take advantage from both the representation of the content as well as the similarities among users. Although there are several ways in which to combine the two techniques a distinction can be made between two basis approaches. A hybrid approach combines the two types of information while it is also possible to use the recommendations of the two filtering techniques independently.[5].

## II. METHODOLOGY

The system takes the user's categorical choice as input and recommends to him/her a list of things that he/she will tend to like. The recommendation list can also be filtered on the basis of time available to the user. I.e if the user wishes to watch a movie but only has 2 hours to spend on, the system will recommend relevant movies of length 2 hours or less. Proposed system architecture :

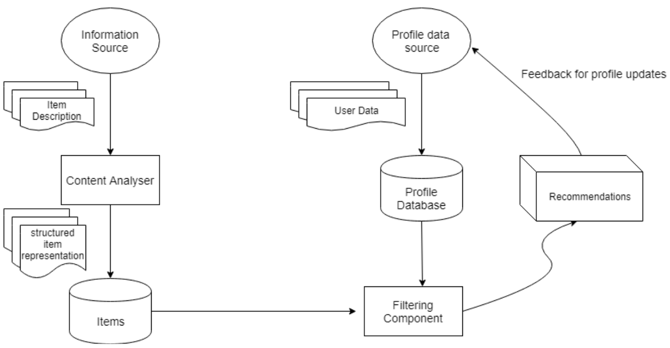


Fig 1:- System Architecture

- *Domains Covered:-*

Movies, TV shows, Songs, Books, Places to visit.

- *Special Section:-*

Surprise Element. Above are the major sections covered in the system.

- *Collecting Data:-*

The initial phase of any new user is basically collection of previously liked entertainment genres and other parameters. Till then, the new users are recommended with certain popular & trending elements.

- *User Preferences:-*

The data collected is then analysed and segregation of data based on what user likes, what similar users like, similarity distance between user and items in the system. And based on all these, algorithms to generate relevant recommendations in desired domains are implemented.

### *A. Song Recommendations*

The user is given three types of songs viz. Generic popular songs recommendation, Song recommendations based on a particular song and songs recommended on the entire profile. The dataset used is the Million Songs Dataset[6]. The schema of one of the files contains the user id, play count and the song id. The other file contains song_id, title, release,artist_name,year and release. These two files form the dataset of the song recommender system.

- *Generic Song Recommender System:-*

This approach is a naive approach and does not yield personalised results. It can be thought of as an all time top songs listened. The generalised approach can be used to recommend songs and overcome the "cold -start" problem [7]. The listen count of all the songs are found out. The total listen count of all the songs are added and a sum is obtained. The percentage of each song's listen count is found with respect to the total listen count.

The sum is sorted and the all the values are normalised between 0 and 1. The top 10 values are then sorted and displayed to the user.

- *Song Recommendation Based On A Particular Song:-*

A database indexer named Solr is used for fast and efficient indexing of the dataset. The more like this query parser of Solr[8] is used to score songs based in their similarity to a particular song. The similarities are based artist_name, release and title. The more like this query parser will then search over the entire dataset and score the tuples based on their "likeness" to the original entries. The top ten entries are displayed as the recommendation to the user.

- *Song Recommendation Based On Entire Profile:-*

Initially, the songs are sorted according to them being the most popular. Unique songs to each of the user and unique user for each of the songs are calculated. The unique users are only found for the songs which the user has listened to. A co-occurrence matrix is created. The dimensions of this co-occurrence matrix are len(user_songs)*len(songs). Then the similarity between all the songs are found out. This similarity is nothing but the number of common users who have listened to both the songs. The similarity measure used is a Jaccard index[9]. A weighted average for all the songs is calculated which is then sorted and the top ten songs are displayed to the user.

### *B. Movie Recommendations*

With such an excel in technology, movies have become an amazing source of entertainment for almost every age group. Recommending movies while travelling, free time will thus be a great action from the user as well as business perspective. To construct an efficient movie recommender system, the dataset from Kaggle[10] is being used The advancement of the construction of a movie recommender in our product line up is as follows *Simple Recommender:-* The Simple Recommender offers generalized recommendations to every user based on movie popularity and genre. The idea behind this recommender is that movies that are more popular and more critically acclaimed will have a higher probability of being liked by the general populace. We sorted our movies based on ratings and popularity and display the top movies of our list. We used the TMDB Ratings to come up with our Top Movies Chart. We used IMDB's weighted rating formula to construct a chart. Mathematically, it is represented as follows:

$$\text{Weighted Rating (WR)} = \left(\frac{v}{v+m}.R\right) + \left(\frac{m}{v+m}.C\right)$$

Equation 2.2.1 : Weighted Rating Formula

where, v is the number of votes for the movie m is the minimum votes required to be listed in the chart R is the average rating of the movie C is the mean vote across the whole report. The algorithm works fine and suggests popular movies with high ratings and most likely to be liked by most of the users. And the algorithm also works great for genre-specific popular movies. That is, when asked for popular movies under a specific genre, it gives appropriate results for an average user. But, the recommender we built in the previous section suffers some severe limitations. For one, it gives the same recommendation to everyone, regardless of the user's personal taste. And to have something which recommends based on user choices and more personalised stuff, we used content based on collaborative filtering.

- *Content Based:-*

For 'The Dark Knight', our system is able to identify it as a Batman film and subsequently recommend other Batman films as its top recommendations. But this is not of much use to most people as it doesn't take into considerations very important features such as cast, crew, director and genre, which determine the rating and the popularity of a movie. Someone who liked The Dark Knight probably likes it more because of Nolan and would hate Batman Forever and every other substandard movie in the Batman Franchise. Therefore, more suggestive metadata like cast, crew, director information etc was also considered When made different considerations, the recommendation list changes towards more relevant results.

- *Collaborative Filtering:-*

The engine that we built is not really personal in that it doesn't capture the personal tastes and biases of a user. Anyone querying our engine for recommendations based on a movie will receive the same recommendations for that movie, regardless of who s/he is. Therefore, we took a step towards Collaborative Filtering to make recommendations to Movie Watchers. Collaborative Filtering is based on the idea that similar-other users' taste can be used to predict how much the current user will like a particular product or service. We used the Surprise library that used extremely powerful algorithms like Singular Value Decomposition (SVD) to minimise RMSE (Root Mean Square Error) and give great recommendations.

- *Hybrid:-*

Advantages of both, collaborative and content-based were exploited to have an efficient recommender engine that provides better and relevant results considering : User's rating for certain movies, Other user's ratings for similar movies to correlate the liking character. The title, overview, cast, rating, and other suggestive metadata about the movie. And after considering all the prominent parameters, and user profile study, the system generates an efficient list of movies relevant to the taste of the user.

*C. Books Recommendation:-*

For recommending books, collaborative filtering is being used where similar users are studies and recommendations are made based on books liked by the group of those similar users amongst themselves.

$$\text{similarity} = \cos(\theta) = \frac{A \cdot B}{\|A\|\|B\|} = \frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} (A_i)^2} \times \sqrt{\sum_{i=1}^{n} (B_i)^2}}$$

Equation 2.3.1 : Formula for Similarity

k-nearest neighbour method is used for clustering and forms the basis for user-based collaborative filtering

The function used for similarity between users is :

$$p_{a,i} = \overline{r}_a + \frac{\sum_{u \in K} (r_{u,i} - \overline{r}_u) \times w_{a,u}}{\sum_{u \in K} w_{a,u}}$$

Equation 2.3.2 : Similarity between Users

Where p(a,i) is the prediction for target user a for item i, w(a,u) is the similarity between users a and u, and K is the neighbourhood of most similar users.

The other approach, i.e. item-based collaborative filtering will use the following formula :

$$p_{a,i} = \frac{\sum_{j \in K} r_{a,j} w_{i,j}}{\sum_{j \in K} |w_{i,j}|}$$

Equation 2.3.3 : Item Collaboration study

where K is the neighbourhood of most similar items rated by an active user a, and w(i,j) is the similarity between items i and j. The knn clustering is used to cluster items. Integrating both implementations, the hybrid system recommends significantly relevant books as per the user's taste.

*D. TV shows Recommendation:-*

Much like movies, TV shows have become a prominent trending entertainment factor. We consider the most significant parameter of genre along with cast. TV shows are also chosen by end users based on their run length. Some users tend not to start a show if its too long in run length. Again, a hybrid system integrating collaborative and content based filtering to generate a list of recommended TV shows clustering similar users and the ratings they have given to various shows is used. Users tend to start watching a TV show based on ratings and popularity in critics world as well as end user ratings. [11]

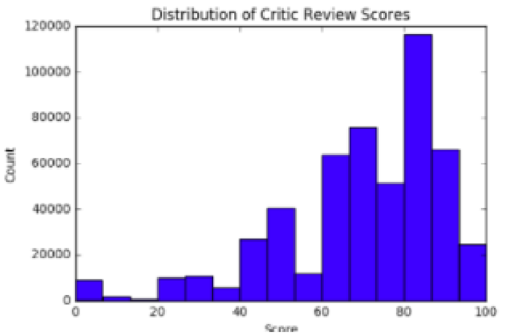


Fig 2:- Critic Review Plot

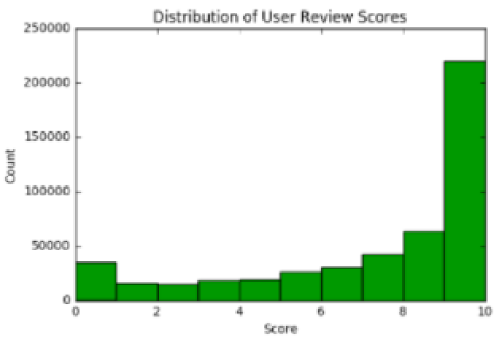


Fig 3:- User Review Plot

Based on these parameters, average ratings are calculated and using similarity measures between similar users and between user - TV shows, recommendation list is generated.

*E. Places to Visit:-*

The user gets a recommendation of places he/she is most probably to enjoy visiting. The system analyses the user profile and studies the type of places user usually visits. Thus if user has visited more beaches in Mumbai, he/she will be recommended with beaches when visited in Bangalore or any other city. The demographic being the proximity to the place and the user's current location. We take the user's geolocation via the IP address of the device[12]. We have used the W3C geolocation API and integrated with the Google Maps API to get the user's location and represent it on the maps. The result of W3C Geolocation API gives 4 location properties, including latitude and longitude (coordinates), altitude (height), and [accuracy of the position gathered], which all depend on the location sources. In some queries, altitude may yield or return no value. The four possible methods for locating the user with the help of this API is:

➢ *GPS (Global Positioning System):-*

This happens for any device which has GPS capabilities. A smartphone with GPS capabilities and set to high accuracy mode will be likely to obtain the location data from this. GPS calculate location information from the satellite signal. It has the highest accuracy; in most Android smartphones, the accuracy can be up to 10 metres.

➢ *Mobile Network Location:-*

Mobile phone tracking is used if a cell phone or wireless modem is used without a GPS chip built in.

➢ *WiFi Positioning System:-*

If WiFi is used indoors, a Wi-Fi positioning system is the likeliest source. Some WiFi spots have location services capabilities.

➢ *IP Address Location*:-

Location is detected based on nearest Public IP Address on a device (which can be a computer, the router it is connected to, or the ISP the router uses). The location depends on the IP information available, but in many cases where the IP is hidden behind Internet Service Provider NAT, the accuracy is only to the level of a city, region or even country.

The user's location is pointed by a red marker whereas the recommended places are denoted by blue markers and are highlighted on the map as well.

*F. Surprise Element:-*

Many a times, when user is bored and psychologically is not able to decide what to do - whether to watch a movie, or listen to some music or binge on a TV show. In such a case, the system simply pops up with some random personalised element (can be a movie, a book, or any such thing) and recommends the user to go for the same.

*G. Time Filtration:-*

This feature is to filter results while recommending entities. The significance being many a times user might have a limited time (say 1.5 hours) and asks for a movie. The system recommending a 3 hour movie would not be an efficient solution. Thus, a time filtration component would actually be useful in such a scenario thereby recommending entities according to the time entered by the user.

*H. Blockchain:-*

A blockchain is a digital, immutable, distributed ledger that chronologically records transactions in near real time. The prerequisite for each subsequent transaction to be added to the ledger is the respective consensus of the network participants (called nodes), thereby creating a continuous mechanism of control over manipulation, errors, & data quality. [13]

There are several reasons to switch to cashless transactions with the help of blockchains. Some of them being : low transaction cost, irrevocable and tamper resistant transactions, highly secure, Fraud minimisation, tracing and auditing by supervisors [14]. The application of such advance

ISSN No:-2456-2165

featurism is to implement micro payments along with transparency and security in transactions. Entertainment these days is costly. End users need to purchase monthly or yearly subscription packages just to watch a few digital programmes. The system aims to provide a pipeline where user can watch recommended digital entities and pay only for what they have used. For example, the user only wishes to watch 2 episodes of any XYZ Tv series. So instead of purchasing entire subscription to watch merely 2 episodes, the user simply pays for what he watches. (The pay per use concept) [15]. The transactions use public keys and private keys of senders and receivers. The transaction is signed by the hash of public key of the sender and is termed as digital signature[16]. This signature is to validate the authenticity of the sender. Further, the proof of concept[17] and proof of authority[18], proof of work[19] features of blockchain verify whether the transaction is valid. This is done by miners[20] of the transactions and receive a reward in the form of cryptocurrency(digital currency used for transaction) for the same. This way, the user has the liberty to pay only for what he/she has used and the transactions are highly secure and transparent as well.

## III. CONCLUSION

All the modules use different filtering models, based on their appropriateness. Demographic filtering was found to be suited for the places to visit model with Straight Line Distance as the admissible heuristic. Similarly, hybrid filtering for movies and a combination of content-based filtering, item-based collaborative filtering and user-based collaborative filtering for books gave the most satisfactory results. Coming to TV shows, hybrid filtering gave efficient results and collaborative filtering proved to be appropriate for recommending songs. The transactions in the end were made highly secure using blockchain features thereby allowing the ‘pay per use’ concept for services leveraged by the user.

## REFERENCES

[1]. recommender-systems.org : available at http://recommender-systems.org/collaborative-filtering/
[2]. wikipedia.com : available at https://en.wikipedia.org/wiki/Collaborative_filtering
[3]. recommender-systems.org : available at http://recommender-systems.org/content-based-filtering/
[4]. slideshare.net available at : https://www.slideshare.net/microlife/recommender-systems-contentbased-and-collaborative-filtering
[5]. recommender-systems.org : available at http://recommender-systems.org/hybrid-recommender-systems/
[6]. Bertin-Mahieux, T., et al. (2011). The million song dataset. 12th International Society for Music Information Retrieval Conference (ISMIR).
[7]. Lika, B., et al. (2014). "Facing the cold start problem in recommender systems." Expert Systems with Applications, vol. 41(4), pp. 2065-2073.
[8]. Class MoreLikeThis, Available at : https://lucene.apache.org/core/7_0_0/queries/org/apache/lucene/queries/mlt/MoreLikeThis.html
[9]. wikipedia.com : Avaialble at : https://en.wikipedia.org/wiki/Jaccard_index
[10]. The Movies Dataset, kaggle.com available at: https://www.kaggle.com/rounakbanik/the-movies-datasetRounak Banik, 2017.
[11]. Metarecommender, available at : https://nycdatascience.com/blog/student-works/capstone/metarecommendr-recommendation-system-video-games-movies-tv-shows/
[12]. Geolocation API Specification, w3.org, https://www.w3.org/TR/geolocation-API/, 8 November 2016.
[13]. Monitor Deloitte: A new game changer for media industry - page 4.
[14]. imarticus.org available at : https://imarticus.org/benefits-of-blockchain-technology/
[15]. https://internetofthingsagenda.techtarget.com/blog/IoT-Agenda/How-blockchain-will-enable-pay-as-you-go-insurance
[16]. wikipedia.org : available at https://en.wikipedia.org/wiki/Digital_signature
[17]. https://en.wikipedia.org/wiki/Proof_of_concept
[18]. https://en.wikipedia.org/wiki/Proof-of-authority
[19]. https://en.wikipedia.org/wiki/Proof-of-work_system
[20]. investopedia.com [online] available at : https://www.investopedia.com/terms/b/bitcoin-mining.asp