Netflix Movie Recommendation Using Fuzzy Logic

Hugo D. Calderon-Vilca
Universidad Nacional Mayor de San Marcos
Lima, Perú
https://orcid.org/0000-0002-1177-4947

Valerie A. Namuche Zavala
Universidad Peruana de Ciencias Aplicadas
Lima, Perú
U201716686@upc.edu.pe

Marco A. Herrera Vargas
Universidad Privada de Ciencias Aplicadas
Lima, Perú
U20171B533@upc.edu.pe

***Summary*— A movie as a means of entertainment presents a great variety of characteristics and attributes such as: genre, rating, languages, locations, theme, reviews, special effects, etc. In recent years, with the advent of a large number of streaming services for movies and series, the catalog of options to choose from has increased, which can be a problem for the vast majority of people. In the present research we propose movie recommendation using fuzzy logic. Two databases were used with which we were able to retrieve approximately 1000 movies which are on the Netflix platform. From these movies we took four attributes (Audience Score, Critic Score, Audience Count and Year) with which we have designed 19 fuzzy logic inference rules calculating in this way the probability of movie recommendation, then we filtered according to user data: age, preferred movie length, year of the movie production, genre, and language of the movie, finally with the attributes and data provided by the user se managed to recommend a list of movies. As a result of the experiment, we show that our proposal reaches an accuracy percentage of 83% on the comments made by a set of users who used it.**

I. Introduction

Nowadays, in these times of quarantine and social isolation, people are looking for different ways to keep themselves entertained and pass the time. One of the most common options is to watch movies, which can be easily found in different digital media such as TV channels or streaming services. However, the latter has gained popularity for its variety of movies from which one can choose and watch at any time, as in the case of Netflix.

While there are many movies, this also means a problem when it comes to choosing which movie to watch. For this reason, a large number of filtering algorithms for movie recommendation have emerged, using different approaches which allow to reach a result that best suits the user's interests.

The most commonly used approaches are: filtering according to content [15], collaborative filtering [3], [4], [5], [6], [7], [8] [9], [10], [11], [12], [13], [14], and hybrid filtering [1], [2] which combines the two previous ones. As can be seen, there are many recommendation algorithms focused on collaborative filtering, as these tend to have problems when users are new, or movies have been recently added [15]. For this, additional algorithms are often used to improve results by taking into account such problem.

In the analyzed movie recommendation proposals [1],[2],[3],[4],[5],[6],[7],[8],[9],[10],[11],[13],[15], the age filter is not taken into account, being indifferent about whether it is correct to recommend movies with adult ratings to all types of users without considering their age. On the other hand, the other studies [12],[14] do take into consideration not only the age but also the gender of the user.

The goal of our work is to recommend Netflix movies using fuzzy logic. Our movie recommendation system is based on the attributes: Critic Score, Audience Score, Audience Count, Year, age, preferred movie length, movie production year, movie gender and movie language, which can be selected by the user in order to get recommendations that are more similar to their preferences and needs.

The rest of the document is structured as follows: Section II provides the State of the Art, in which relevant research on movie recommendation is reviewed. Section III describes the contribution of this research. Section IV details the design and implementation of the proposed recommender system. Section V validates the results obtained and discusses them in contrast with other proposals. Finally, Section VI presents the conclusions and future work.

II. State of the Art

A. Recommentarion algorithms with collaborative filtering (CF)

In [3] a comparison of algorithms used after processing the data collected from 18 students who rated 275 movies is performed. The algorithms used were *K-NN baseline item-based, K-NN baseline user-based, K-NN with means item-based, K-NN with means user-based, SVD, SVD++,* and *Co-clustering*; of which the *SVD* algorithm gave the best results.

In [4] a user-based CF is used, in which the *Graph Database* idea is introduced to model and manage data requests in a simpler and more intuitive way. By default, 30 movies are shown with the average rating of the top 50, where the larger the node radius is, the better the recommendation. For movies with different averages, different colors are used to distinguish them, and for the most recommended movie, a yellow border is added to the node; at the same time the thickness of the border represents the recommendation level of a movie.

In contrast, item-item algorithms are difficult to evaluate as it is difficult to know if two items are in fact similar. In [5] a comparison of algorithms used after processing the data obtained from *MovieLens* 20M is made, apart from using additional metadata obtained from *The Open Movie Database (OMDB)*[1] y de *The Movie Database (TMDb)*[2]. The algorithms used were: *CF-Cosine Similarity, CF-Pearson* and *content filtering*, using *TMDb´s similarity algorithm* as a standard of comparison. The results showed that 62% of the evaluated pairs had a total consensus, in which no participant disagreed on their similarity. In the remainder, there was at least one who disagreed. This showed that perceived similarity is definitely not universal; an object that is similar to some may not be similar to others.

However, there may be a high probability that they agree in most cases.

In [6] a latent factor model is applied to overcome the shortcomings that a CF may bring. According to the results obtained by processing the MovieLens 20M dataset, the top 3 *features* are taken for each movie, which represent its latent factor, then if a user gives a good rating to a movie, then it is assumed that he/she likes other movies with similar latent factor. This system only considers the ratings given by the user for the final filter.

In [7] a similarity matrix is created consisting of weights that would come to be the ratings of a user and represents the relationship between the items. Then a modified cosine similarity matrix is generated in which items with similar weights are found. Then we move on to fill in the values of movies not rated by the user. Finally, movies with similar ratings are taken and the "k" with higher ratings are displayed.

In [8] the movies are separated according to their genre, then externally joined with their respective ratings, in this way 3 approaches are obtained for each movie which are user preferences, average rating and consumption ratio. This results in vectors with "n" attributes for each approach, representing the number of genres in the *dataset*. The *K-means Clustering* algorithm is used to separate the different users and compare them in the 3 approaches previously mentioned, in order to choose the best among them. At the same time, a Principal Component Analysis (PCA) is used to reduce the dimension and obtain a better result in the clustering, in this way, the data of the users is not lost after clustering. Finally, the rating is added in the last column, which will then be used by the neural network. A *Pattern Precognition Network* is used to classify the inputs according to the 11 classes of objectives, being these the different classifications that can be given by the user.

In [9] a *Word2Vec* algorithm is applied to process the metadata information of the movies. First a training is performed by taking as input the known user's movie history, a vector of good movies is created in which the movies that have obtained good rating by the user are added, then a pair of movies are randomly taken from this vector as input and output of the *Word2Vec* network for training. Then a prediction is created for the user vector consisting of the movies that have been viewed by the user. Each embedded movie has a weight that reflects the user's preference. Finally, to generate the list of recommendations, an inner product is performed between the vector of movies with the user vector to rank them by each. The "k" movies that obtained the best results are shown.

In [10] they start from a CF, adding an additional filter to improve the results and return more accurate recommendations. *Root mean squared error* (RMSE) is applied with the known ratings and the ratings predicted by the algorithm to evaluate in this way the performance of the algorithm. Then, having computed the ratings of each movie, the similarity between 2 movies is calculated using *Cosine Similarity*, thus obtaining a list of recommended movies. The "k" best performing movies are shown.

In [11] they work with a model-based CF, trained with movie preferences by each known user, to obtain a trend prediction model. It works with a triplet (u, i, r) with known values where: "u" is the user, "i" is the movie and "r" is the rating given by "u" to "i". Using a coordinate system, where the values of the vectors U and I have random coordinated, the positions of all the coordinates are iteratively updated until for each triplet the Euclidean distance between the coordinates of "u" and "I" is equal to "r". Finally, having processed all the known data, for each "u" with an "i" that has not been rated, a "" corresponding to the Euclidean distance between these 2 points is determined, thus compiling a list with the movies that have not yet been rated and displaying the "k" with highest rating.

In [12] a collaborative algorithm and *K-Means Clustering is employed*. It starts by dividing the users into several groups, each group produces an opinion guided by the rating calculation of the items. It is intuited that the history of movies watched by the user is related to their tastes and this will be used to group the users, also taking into account gender and age, in order to get recommendations that are more in line with each user. The *MovieLens dataset* is also used.

In [13] an Item-based Collaborative-Filtering Algorithm is applied. The main steps for the CF algorithm are followed, the most important idea is to multiply the user matrix with similar matrices to obtain a list of recommendations. Then, the distributed item-based algorithm is introduced in a distributed environment, using MapReduce a vector of recommendations is generated and assembled for each user to obtain a list of recommendations for each user, then a filtering is used to obtain the N movies with the highest value obtained in the process. Finally, a recommendation algorithm based on quality is implemented, the K-nearest neighbors algorithm (*KNN Algorithm*) is applied on the list obtained in the last step with respect to the needs obtained with the previous algorithm; thus, arriving at the most appropriate recommendations for the user.

In [14] they explain about the use of natural language recognition with speech recognition for input using google personal assistant applying a neural CF using *tensorflow* via *Google CloudML*, and *PyTorch* via *Python backend interfaces*. In addition, metadata from *The Open Movie Database* (OMDB) and *dataset* from *MovieLens 20M*.

B. Recommendation Algorithms with Content Filtering

In [15] a recommendation system was developed with a simple algorithm, filtering information related to the user's needs and interests (genre, actor, year, score, director) being an important recommendation system due to its economic potential. According to the 5 attributes, the weight of all movies is calculated in relation to the attributes indicated by the user. After having calculated the weights, the *K-means Clustering* is used with k=4, with which a list of movies ordered in descending order according to their weight will be obtained. If the list is less than 20, all the movies obtained otherwise will be shown, a pre-filter is made and the 20 movies with the highest weight are shown, in case they are equal, the preference goes to the one with the highest number of votes.

C. Recommendation Algorithms with Hybrid Filtering (HF)

In [1] they propose to recommend movies based on user preferences and interests and represent a plot for them. The system has 4 main phases: extraction of user interests from

Facebook, content analysis of movie summaries, recommendation phase and plot creation phase. In short, the user data is collected from *Facebook* and the movie *dataset* is taken from IMDb. The data is extracted using Graph API and saved in JSON format indicating which user profile and friends were built. These profiles contain information about the language, age location, movies and celebrities the person likes. We evaluated our system for 50 users and plotted the results in a graph for 10 users who showed extreme variations. Movies that were recommended by content filtering become filtered by CF which makes the system a hybrid recommendation filtering. However, if the user has not shown interest in any actor or movie before then there will be no similarities for him/her. Then there are no movies to show by content and collaborative filtering in the personalized recommender system.

In [2] the proposed solution is to improve the scalability and quality of movie recommendation systems. They use a hybrid approach by unifying Content-Based Filtering and CF, so that it can benefit from both. To compute similarities between different movies in the chosen *dataset* efficiently and in the shortest time and reduce the computation time of the movie recommendation search engine we use cosine similarity measure. *Content-Based Filtering, Collaborative Filtering* (within which *Simple Support Vector Machine Algorithm* and *Adjusted K-Means Algorithm are applied*), *Genetic Algorithm* and *Cosine Similarity Measure are used*. It was demonstrated that a movie recommendation system using the Hybrid approach and genetic algorithms is better and provides better performance than the existing ones based only on one of the two filters in terms of accuracy, quality, scalability, and processing time.

III. NETFLIX MOVIE RECOMMENDATION METHODOLOGY

In our work we employed a movie recommendation system architecture using fuzzy logic, we conducted the experiment in the Python programming language; in addition, we designed a graphical interface developed in the same language for the *backend* using the "*flask*" library and web application with html and JavaScript for the *frontend*.

A. Input data

We used 2 *datasets* obtained from the Kaggle page [19]. The first *dataset* is "*Movies on Netflix, Prime Video, Hulu and Disney+*" and was provided by user Ruchi Bhatia, from which we will take the variables *Critic Score* and *Year*. The second *dataset* is "*rotten-tomatoes-movies-and-critics-datasets*" provided by user Mark, for the remaining variables. In this case, we will only analyze movies found on the Netflix platform. Table I describes in detail the input data: *Critic Score*, *Audience Score*, *Audience Count*, *Year*. In total, 986 movies were obtained for the *dataset*.

TABLE I. INPUT DATA

Attribute	***Description***
Critic Score	Movie rating according to critics.
Audience Score	Movie rating according to the audience.
Audience Count	Number of audience who rated the film.
Year	Year in which the film was released.

B. Data Processing and Normalization

Before proceeding to use the data, we made sure that there is no null data to avoid errors when running the algorithm. For this we considered the following:

- For the "language", the language "English" was used since it is the most common language in the *dataset*.
- For the age rating of the movies, it was assigned as "16+" to avoid cataloging a movie for adults in lower ranges.
- For the description of the movies, an empty string was assigned.
- For the year of release, the year 2000 was used.
- For qualitative data, the mode of the column was assigned.

C. Fuzzy sets and membership functions

a) Critic Score: For this variable we used as references the rating provided by IMDB, which rates movies in the range [0,10], where 0 is bad and 10 is very good. A trapezoidal membership function was used for this variable.

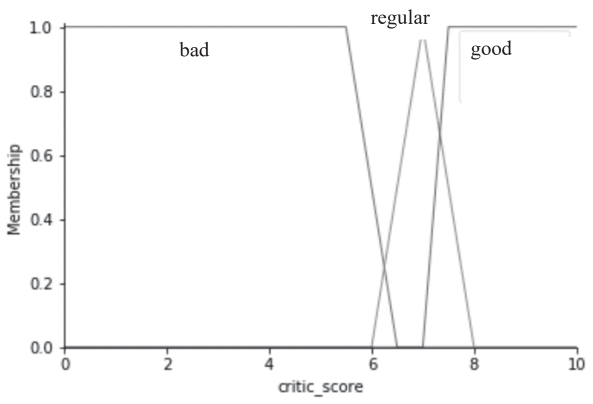


Fig. 1. Fuzzy set of Critic Score and its membership function

b) Audience Score: For this variable, the rating provided by Rotten Tomatoes is used as a reference, using the audience, who rate the films in a range of [0,100]%, where 0% is bad and 100% is very good. A trapezoidal membership function was used for this variable.

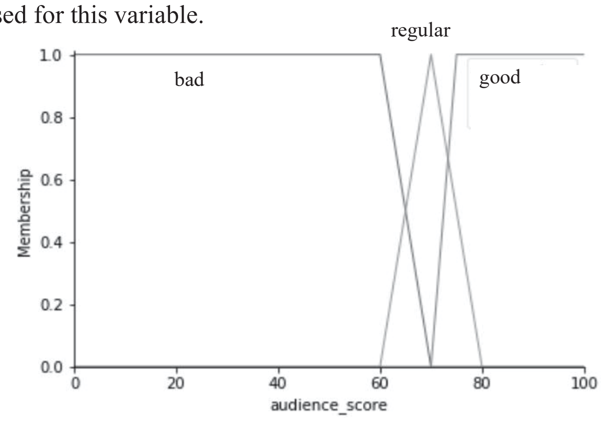


Fig. 2. Fuzzy set of Audience Score and its membership function

c) Audience Count: For this variable we use as a reference the amount of audience that rated the movie from the User Score variable and which was scaled in a range of [0,460].

Where the movies with an audience value in the range of [0,50] are considered as "low", "regular" to those in the range [45,120] and "high" to those in the range [90,460], and for the calculation of this range the average audience was taken into account which was 77. A trapezoidal membership function was used for this variable.

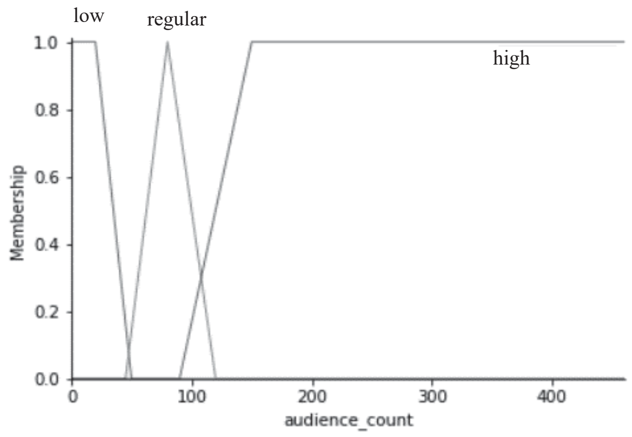


Fig. 3. Fuzzy set of Audience Count and its membership function

d) Year: Year of release of the movie, this variable is taken from the IMDB database. We considered movies to be old if their year of release is before 2000 and recent if they were released after 2000. A trapezoidal membership function was used for this variable.

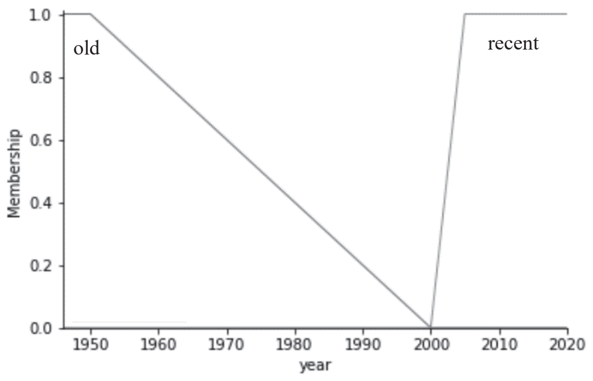


Fig. 4. Fuzzy set of Year and its membership function

D. Inference rules

The rules have been designed combining the variables Critic Score, Audience Score, Audience Count and Year; and as a consequence, the variable Recommendation, which is the one that will determine whether it is recommended or not. We have decided to give more importance to the audience reviews than to the critics, since the latter focus on specific data compared to the audience that mainly consider whether it seemed good or not; for example, movies of the comedy genre do not usually get a good rating from critics.

Table 2 shows the rules, which use at least 2 variables to determine whether to recommend the film. Using as a reference a questionnaire made to a group of people, we were able to determine 19 rules.

For rule 1, 2 and 3 it was determined that, as long as the critics' rating is good and the amount of audience that rated it is low or regular, then the movie is recommended, since the audience's rating can also be affected by their emotions, which can lead to a dishonest rating. For rule 4 it was determined that, if the movie is old and has a poor rating from critics, then it will not be recommended.

For rules 5, 6, 7, 8 and 9 a negative rating by the audience was considered. In 5, 6, 7 and 8 a low or regular rating by critics or a small amount of audience was also considered, so in these cases the movie will not be recommended. However, in 9 the film is recommended since the rating by the critics is very good and the audience is regular.

Rules 10, 11, 12, 13 and 14 start from the premise of a regular rating by the audience. Rules 10 and 13 tell us that if the number of votes was also regular and the firm is recent or the critics' rating was regular, then it will be recommended; on the other hand, rule 11 determines that it is not recommended if the critics' rating is also bad. Apart from that, rule 12 determines that the movie is not recommended if it is also old and has a regular rating from critics. Finally, rule 14 recommends the film if the critics' rating is also very good.

The rest of the rules are premised on a good rating from the audience. Rules 15, 17, 18 and 19 recommend the movies considering a mixed rating from critics, while rule 16 does not recommend the film if the critics rated the film poorly and the audience was low.

TABLE II. RULES DESIGN

N°	*Audience Score*	*Critic Score*	*Audience Count*	*Year*	*Resultado*
1	-	*Very Good*	*Low*	-	*Recommended*
2	-	*Very Good*	*Regular*	*Rece nt*	*Recommended*
3	-	*Mala*	*Low*	-	*Not Recommended*
4	-	*Mala*	-	*Old*	*Not Recommended*
5	*Bad*	-	*High*	-	*Not Recommended*
6	*Bad*	*Bad*	-	-	*Not Recommended*
7	*Bad*	*Regula r*	-	-	*Not Recommended*
8	*Bad*	*Regula r*	*Regular*	-	*Not Recommended*
9	*Bad*	*Very Good*	*Regular*	-	*Recommended*
10	*Regular*	-	*Regular*	*Rece nt*	*Recommended*
11	*Regular*	*Bad*	-	-	*Not Recommended*
12	*Regular*	*Regula r*	-	*Old*	*Not Recommended*
13	*Regular*	*Regula r*	-	*Rece nt*	*Recommended*
14	*Regular*	*Very Good*	-	-	*Recommended*
15	*Very Good*	-	*High*	-	*Recommended*
16	*Very Good*	*Bad*	*Low*	-	*Not Recommended*
17	*Very Good*	*Bad*	*Regular*	-	*Recommended*

18	*Very Good*	*Regula r*	-	-	*Recommended*
19	*Very Good*	*Very Good*	-	-	*Recommended*

E. General recommendation process

Considering that the objective is to calculate the probability that a movie is or is not recommended, according to the data entered by the user: length of the movie, genre of the movie, rating of the movie, original language of the movie and yeah of release of the movie; in this way reducing the number of recommendations and obtaining those that most closely resemble the user's needs, an ethical filter (age of the user) has also been considered, in order to avoid recommending movies whose age rating is not suitable for the user.

Fig. V illustrates the steps involved to obtain the recommendations for the user.

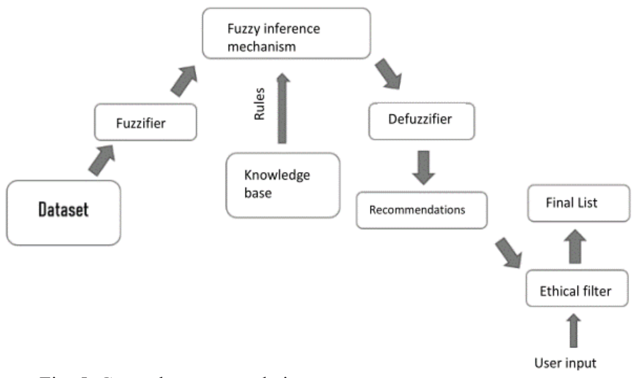


Fig. 5. General recommendation process

With the idea of getting a list of recommendations that meet the user's requirements. When starting the application, all the data are processed in order not to recalculate the probabilities in each user request.

Fuzzy Inference Mechanism: The set if data provided by the user are fuzzy sets subjected to the inference rules seen in Table II, whose purpose is to calculate the probability of recommendation of the movies, and thus be able to use them later in the final filter.

Final Filter: After having processed all the metadata, the user has the possibility to enter values such as movie length, original language, year of release, genre, rating and age; to reduce the recommendations according to these values, the list is filtered according to age.

F. Output data

As output data we have a list that has the "K" highest scores according to its probability of being recommended. Table II represents the output data in detail.

TABLE III. OUTPUT DAT

Attribute	***Description***
Name	Movie title.
Description	Summary of the movie .
Genre	Genre or genres of the movie.
IMDB Score	Rating from IMDB critics.
Rotten Tomatoes Score	Rotten Tomatoes audience rating.
Year	Year of release of the film.
Duration	Duration of the movie.
Rating	Age rating of the movie.
Movie Language	Languages spoken in the movie.

IV. APPLICATION DESIGN

In the present project we have additionally proposed a responsive web application of a movie recommendation system, in which we decided to use Python with the help of the "SKFuzzy" library for the inference rules, "flask" and "flask_cors" for the display, and Html code with JavaScript for the graphical interface.

A. Implementation

For the implementation of the algorithm which gives life to the movie recommendation system, the following steps were followed:

- The data were preprocessed to normalize them according to the methods described in section III.
- The fuzzy inference mechanism was implemented from the fuzzy sets described in Table II.
- The 19 rules were applied to the fuzzy inference mechanism.
- For defuzzification, the Mamdani method was used to obtain a value which determines the recommendation.
- From the complete set of processed data, the age filter is applied with the values entered by the user.

B. Graphical Interface

A minimalist interface was designed where data entry is allowed to display the desired content, this is a web application that consumes services from a Rest API in Python.

The user data input interface, the variables: Critic Score, Audience Score, Audience Count and Year, these variables take values according to Fig. 1,2,3 and 4 respectively, the other data are filters: age, duration of the movie, year of production of the movie, genre and language of the movie.

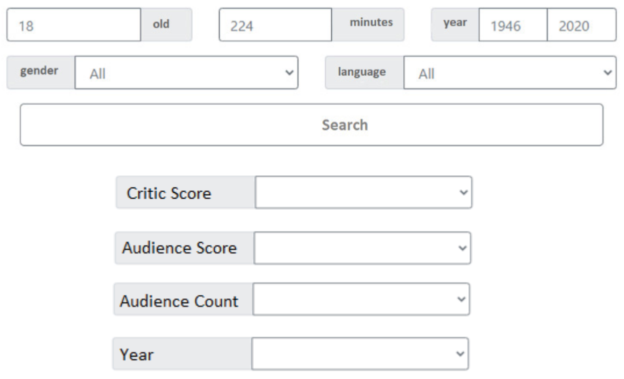
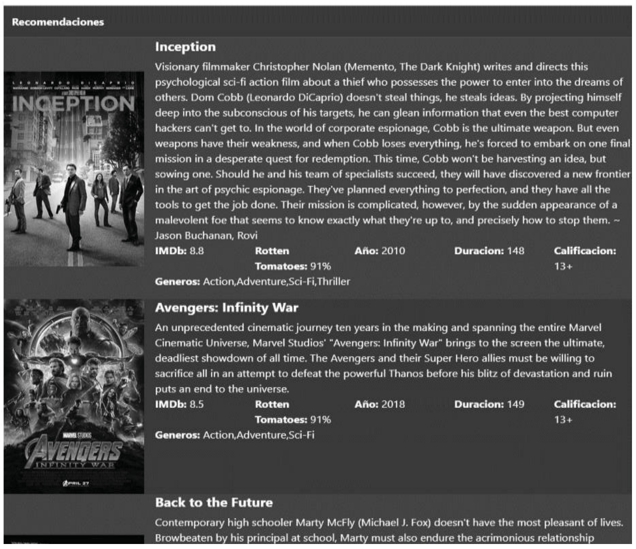


Fig. 6: Graphical data input interface

Fig. VI is the form that the user applies to search for a movie recommendation.


Recomendaciones

Inception

Visionary filmmaker Christopher Nolan (Memento, The Dark Knight) writes and directs this psychological sci-fi action film about a thief who possesses the power to enter into the dreams of others. Dom Cobb (Leonardo DiCaprio) doesn't steal things, he steals ideas. By projecting himself deep into the subconscious of his targets, he can glean information that even the best computer hackers can't get to. In the world of corporate espionage, Cobb is the ultimate weapon. But even weapons have their weakness, and when Cobb loses everything, he's forced to embark on one final mission in a desperate quest for redemption. This time, Cobb won't be harvesting an idea, but sowing one. Should he and his team of specialists succeed, they will have discovered a new frontier in the art of psychic espionage. They've planned everything to perfection, and they have all the tools to get the job done. Their mission is complicated, however, by the sudden appearance of a malevolent foe that seems to know exactly what they're up to, and precisely how to stop them. ~ Jason Buchanan, Rovi

IMDb: 8.8 **Rotten Tomatoes:** 91% **Año:** 2010 **Duracion:** 148 **Calificacion:** 13+

Generos: Action,Adventure,Sci-Fi,Thriller

Avengers: Infinity War

An unprecedented cinematic journey ten years in the making and spanning the entire Marvel Cinematic Universe, Marvel Studios' "Avengers: Infinity War" brings to the screen the ultimate, deadliest showdown of all time. The Avengers and their Super Hero allies must be willing to sacrifice all in an attempt to defeat the powerful Thanos before his blitz of devastation and ruin puts an end to the universe.

IMDb: 8.5 **Rotten Tomatoes:** 91% **Año:** 2018 **Duracion:** 149 **Calificacion:** 13+

Generos: Action,Adventure,Sci-Fi

Back to the Future

Contemporary high schooler Marty McFly (Michael J. Fox) doesn't have the most pleasant of lives. Browbeaten by his principal at school, Marty must also endure the acrimonious relationship

Fig. 7. Graphical data output interface

Fig. VII shows the list of recommended movies according to the recorded data: Critic Score, Audience Score, Audience Count, Year, age, duration of the movie, year range to which the movie production belongs, genre and language of the movie, in the background this recommendation uses the inference rules raised in Table II.

V. Results and discussion

A. Results

A test was performed with 10 people who carried out 10 searches each, being a total of 100 transactions with different input values in each one. After performing these tests, they were asked to comment on the results and the following was obtained:

- 83% of the total transactions had positive comments on 70% of the movies shown.
- 12% showed movies that the users did not know about, but this was due to the fact that they entered rather complicated data.
- The audience rating was quite important as it allowed movies such as comedy or horror to appear as these are not usually rated well by critics.
- The users considered the results shown by the application to be correct, since when analyzing the scores, they were shown in a logical order.

B. Discussion

To recommend movies [2] used Content-Based Filtering, Collaborative Filtering (within which Simple Support Vector Machine Algorithm and Adjusted K-Means Algorithm are applied), Genetic Algorithm and Cosine Similarity Measure. They focused on quality and scalability, comparing it to our work, we used fuzzy logic and managed to recommend movies focusing on variables: Critic Score, Audience Score, Audience Count and Year.

In [8] they manage to recommend movies in three approaches for each movie: user preferences, average rating and consumption ratio; this results in vectors with “n” attributes for each approach, which represent the amount of genres in the dataset, they used the K-means Clustering algorithm. Comparing with our work, instead of preferences we consider age as a filter to recommend, among other variables also year of production of the movie acts as a filter.

Another research [7], to recommend movies, creates a similarity matrix consisting of weights that would come to be the ratings of a user and represents the relationship between the items, then a modified cosine similarity matrix is generated in which the items with similar weights are found, then predict the unknown ratings and, finally, try to fill the values of the movies not rated by the user. In our work we focus on fuzzy logic technique using fuzzy variables Critic Score, Audience Score, Audience Count and Year which allow us to determine the probability of recommendation from that we generate a list for the user always using the age filter.

Other investigations demonstrated the recommendation of movies using collaborative filtering [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14]. However, for collaborative filtering it is required to collect the user profile to submit to the recommendation model. Instead, in our proposal, we only collect age, movie duration, movie production year, genre and language which act as filters, but the important work of recommendation corresponds to the set of rules proposed in Table II which are applied in fuzzy logic.

Conclusions and Future work

With 4 movie attributes we were able to design a fuzzy set which uses each attribute for each element of our model. Five user filters have been added to allow the selection of the recommended movie.

A movie recommendation process was designed, for the inference engine we designed 19 inference rules used in fuzzy logic, which allowed us to reach a more accurate recommendation using the Mamdani model for its defuzzification.

Finally, we propose a responsive web application as a tool for the movie recommendation system implemented in Python and Html with JavaScript.

As future work, it is possible to improve the accuracy by adding more categories in addition to the 4 attributes already chosen and filters. Another new feature that can be added is the possibility to mark movies already watched and rate them for further processing. Finally, more filters can be added for different types of users, for example for people suffering from epilepsy or sensitive to a specific topic.

References

[1] *N. Immaneni, I. Padmanaban, B. Ramasubramanian and R. Sridhar,* “A meta-level hybridization approach to personalized movie recommendation”, *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Udupi, 2017, pp. 2193-2200, doi: 10.1109/ICACCI.2017.8126171.

[2] S. Agrawal and P. Jain, "An improved approach for movie recommendation system", *2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, Palladam, 2017, pp. 336-342, doi: 10.1109/I-SMAC.2017.8058367.

[3] *Mingyue Tang, Hang Xie, and Tiffany Y. Tang. 2018, "*Combining wAMAN and Matrix Factorization to Optimize One-Class Collaborative Filtering and Its Application in an Emotion-Aware Movie Recommendation System*". In Proceedings of the 2018 International Conference on Big Data and Computing (ICBDC '18). Association for Computing Machinery, New York, NY, USA, 108–114. DOI:https://doi.org/10.1145/3220199.3220202.*

[4] *N. Yi, C. Li, X. Feng and M. Shi,* "Design and Implementation of Movie Recommender System Based on Graph Database", *2017 14th Web Information Systems and Applications Conference (WISA)*, Liuzhou, 2017, pp. 132-135, doi: 10.1109/WISA.2017.34.

[5] *Lucas Colucci, Prachi Doshi, Kun-Lin Lee, Jiajie Liang, Yin Lin, Ishan Vashishtha, Jia Zhang, and Alvin Jude. 2016. "*Evaluating Item-Item Similarity Algorithms for Movies". *In Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '16). Association for Computing Machinery, New York, NY, USA, 2141–2147. DOI:https://doi.org/10.1145/2851581.2892362*

[6] *W. Liu, B. Wang and D. Wang*, "Improved Latent Factor Model in Movie Recommendation System", *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, Singapore, 2018, pp. 101-104, doi: 10.1109/ICoIAS.2018.8494074.

[7] *M. K. Kharita, A. Kumar and P. Singh*, "Item-Based Collaborative Filtering in Movie Recommendation in Real time", *2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC)*, Jalandhar, India, 2018, pp. 340-342, doi: 10.1109/ICSCCC.2018.8703362.

[8] *M. Ahmed, M. T. Imtiaz and R. Khan*, "Movie recommendation system using clustering and pattern recognition network", *2018 IEEE 8th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, 2018, pp. 143-147, doi: 10.1109/CCWC.2018.8301695.

[9] *Y. C. Yoon and J. W. Lee*, "Movie Recommendation Using Metadata Based Word2Vec Algorithm", *2018 International Conference on Platform Technology and Service (PlatCon)*, Jeju, 2018, pp. 1-6, doi: 10.1109/PlatCon.2018.8472729.

[10] *Quynh Nhut Nguyen, Nghia Duong-Trung, Dung Ngoc Le Ha, Xuan Son Ha, Tan Tai Phan, Hien Xuan Pham, and Hiep Xuan Huynh. 2020.* "Movie Recommender Systems Made Through Tag Interpolation". *In Proceedings of the 4th International Conference on Machine Learning and Soft Computing (ICMLSC 2020). Association for Computing Machinery, New York, NY, USA, 154–158. DOI:https://doi.org/10.1145/3380688.3380712*

[11] *Harris Papadakis, Nikos Michalakis, Paraskevi Fragopoulou, Costas Panagiotakis, and Athanasios Malamos. 2017. "*Movie SCoRe: Personalized Movie Recommendation on Mobile Devices*". In Proceedings of the 21st Pan-Hellenic Conference on Informatics (PCI 2017). Association for Computing Machinery, New York, NY, USA, Article 46, 1–6. DOI:https://doi.org/10.1145/3139367.31*

[12] *J. Zhang, Y. Wang, Z. Yuan and Q. Jin*, "Personalized real-time movie recommendation system: Practical prototype and evaluation", in *Tsinghua Science and Technology*, vol. 25, no. 2, pp. 180-191, April 2020, doi: 10.26599/TST.2018.9010118.

[13] *Yan Tang, Mingzheng Li, Wangsong Wang, Pengcheng Xuan, and Kun Geng. 2017.* "Quality-Aware Movie Recommendation System on Big Data". *In Proceedings of the Fourth IEEE/ACM International Conference on Big Data Computing, Applications and Technologies (BDCAT '17). Association for Computing Machinery, New York, NY, USA, 273–274. DOI:https://doi.org/10.1145/3148055.3149209*

[14] *Jeffrey Dalton, Victor Ajayi, and Richard Main. 2018.* "Vote Goat: Conversational Movie Recommendation". *In The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval (SIGIR '18). Association for Computing Machinery, New York, NY, USA, 1285–1288. DOI:https://doi.org/10.1145/3209978.3210168*

[15] *M. T. Himel, M. N. Uddin, M. A. Hossain and Y. M. Jang*, "Weight based movie recommendation system using K-means algorithm", *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, Jeju, 2017, pp. 1302-1306, doi: 10.1109/ICTC.2017.8190928.

[16] *Wissal Farsal, Samir Anter, and Mohammed Ramdani. 2018.* "Deep Learning: An Overview". *In Proceedings of the 12th International Conference on Intelligent Systems: Theories and Applications (SITA'18). Association for Computing Machinery, New York, NY, USA, Article 38, 1–6. DOI:https://doi.org/10.1145/3289402.3289538*

[17] *A. Monelli and S. B. Sriramoju*, "An Overview of the Challenges and Applications towards Web Mining", 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), 2018 2nd International Conference on, Palladam, India, 2018, pp. 127-131, doi: 10.1109/I-SMAC.2018.8653669.

[18] *G. Sharma and M. Waghmare*, "Study of Aspect level Sentiment Analysis and Word Embeddings", 2019 IEEE International Conference on Electrical, Computer and Communication Technologies (ICECCT), Coimbatore, India, 2019, pp. 1-6, doi: 10.1109/ICECCT.2019.8869404.

[19] Kaggle: Movies on Netflix, Prime Video, Hulu and Disney+ https://www.kaggle.com/ruchi798/movies-on-netflix-prime-video-hulu-and-disney May 22, 2020

[20] Kaggle: rotten tomato movie reviwes https://www.kaggle.com/yazeidalqahtani/rotten-tomato-movie-reviwe April 6, 2020

[21] Kaggle: IMDb movies extensive dataset https://www.kaggle.com/stefanoleone992/imdb-extensive-dataset?select=IMDb+movies.csv November 24, 2019