Mood based Recommder System

Rishneel chand  
*School of IT, Engineering, Mathematics and Physics (STEMP)*  
*University of the South Pacific (USP)*Suva, Fiji  
[chandrishneel23@gmail.com](mailto:chandrishneel23@gmail.com)

Anuraganand Sharma  
*School of IT, Engineering, Mathematics and Physics (STEMP)*  
*University of the South Pacific (USP)*Suva, Fiji  
sharma\_au@usp.ac.fj

Anuraag Singh  
*School of IT, Engineering, Mathematics and Physics (STEMP)*  
*University of the South Pacific (USP)*Suva, Fiji  
[as3085449@gmail.com](mailto:as3085449@gmail.com)

Jeshneel Deo  
*School of IT, Engineering, Mathematics and Physics (STEMP)*  
*University of the South Pacific (USP)*Suva, Fiji  
jashkumar1224@gmail.com

*Abstract*— A recommender system is designed to suit the user’s preference, but very few platforms consider the user’s mood status when recommending a title. Human behavior is very versatile at most and hard to predict, thus this research paper observes how a Mood-based Movie recommender system tries to find a middle ground between the identified gap which is the mood to allow users to get more accurate movies. Many platforms are capable of recommending movies efficiently but there is an uncertainty that the users do not find the recommender systems to be efficient enough from a user’s perspective in terms of recommendations preferred by them. The concept of mood in the research focuses on implementing machine learning algorithms to try and combat the issues the users face. The users are presented with a GUI to select up to three moods to get the results suited for their particular moods. Thus our research reflects a prominence on the Mood-based aspect of the Movie Recommender system.

Keywords—Recommender system, Mood, Machine learning, algorithms.

# **Introduction**

The recent years have evolved for movies and shows as they’ve transitioned their way into digital and online media, people tend to stream and watch movies on the internet which is much simpler and more reliable for users to keep watching newer content on a fast passed scale in the current times of technology. However, with more data available it makes the processing of having satisfactory results much more difficult as users have to navigate certain times through multiple results and searches to get what they are looking for [1]. To combat these problems data scientists have introduced Recommender systems to filter data to the user needs as required.

Recommender systems are generally created to aid us in decision-making when selecting a particular product or service. The recommendation systems are quite relevant in the current times as many services rely on these techniques to generate traffic towards a particular product or to enhance services and customer activity.

The introduction of Machine learning is key as it is a sub-field of artificial intelligence. Machine Learning (ML) uses computers to replicate human intelligence and behavior through learning and allows the computers to recognize and obtain understanding from real-life problems, and further improve performance on tasks based on this new understanding of knowledge [4]. Netflix one of the major leading giants uses machine learning to provide users with their preferred movie titles presented to them. “If subscribers fail to find movies that interest and engage them, they tend to abandon the service. Connecting subscribers to movies that they will love are therefore critical to both the subscribers and the company [5]”.

The complexities of developing an efficient recommender system face challenges as there are many variations and patterns to learn and adapt with machine learning, making the process of choosing machine learning algorithms much more sophisticated.

To improve the recommendation more efficiently using the study of artificial intelligence, “in 2007 Netflix released a dataset to allow data mining experts to further use machine learning to improve the accuracy of recommendations [5]”.

The streaming giant Netflix placed a $1 million award to the first person or team who can design a recommender system to beat the company’s current one (called Cinematch) by increasing the recommendation accuracy by at least 10% [3] which further shows how even major companies such as Netflix are still trying to find a recommender system with better accuracy.

To contribute to the research for RS, this paper focuses on improving the recommender system’s accuracy, by introducing unique criteria such as the user mood, as even though a user might have their favored movie, a simple factor of mood may affect their style of choice further leading towards a different preference style and rating depending on the mood status thus making the accountability of the user mood quite important.

## Content-based Technique

A content-based recommender system is suited to meet the user’s profile based on what the user likes or prefers to view, thus considering a personalized approach when recommending a particular title [2]. The recommender system will work according to the user’s behavior to learn which titles to present to the user, as it compares the genres and ratings of the movie preferred by the user which determines what types of similar recommendations the user is likely to get, thus shaping the recommendation feed similar to the user’s interaction.

## Personalized Mood-based Approach

The identified gap for this research paper is mood, where the analysis will be done on the Movie Recommender system with mood to observe how it performs in terms of accuracy and algorithm efficiency in conjunction with each other. The difficulties arise in RS due to the unpredicted results of users towards the movies, as the mood influences the reaction between the user and the RS which leads to Data inconsistency through the mood as a user might rate their best movie by a low rating on a bad mood creating an inconsistent flow for the RS as the reaction of the user was unprecedented, “Since the recommendation mechanism assumes that all ratings are given consistently, the biased ratings can compromise the recommendation accuracy”[3][2].

# **Problem statement**

People often struggle to select an appropriate movie to their liking as they are overwhelmed with the choices, and decisions and are subjected to keep looking for the right movie for a particular time to enjoy the movie. Many RS out there do the task successfully but not accurately, as the interactions between humans and machine learning are bridged through the data patterns.

# **Mood-based Recommender system**

## Algorithm

The selected algorithm for machine learning utilized in the Mood-based recommender system was the Cosine similarity algorithm which generated similar results of movies watched by the user, as it would look at the similarity in genre, ratings, and other information to find a movie closer suited to the user best. The second algorithm incorporated into the program was the logistic regression which is a classification algorithm*.*

## Mood Scale

The mood was mapped to match certain genres of movies for the user, as the user inputs the mood the program will generate results of recommendations driven towards the moods selected. Users are given the choice to select from up to 1 to 3 moods to add more credibility and accuracy to the recommendations and enable the user to have more freedom of selection rather than being restricted to choosing 1 input. There are 5 positive and negative mood states to choose from making the choice much more versatile for users.

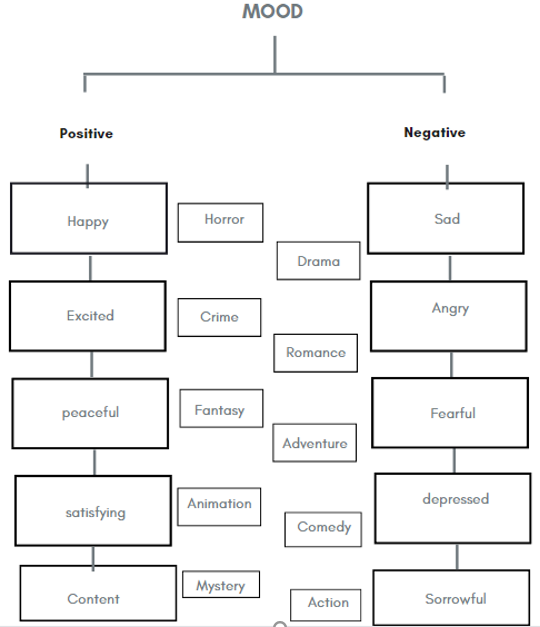
## Mood psychology

Mood can be a bias factor and quite unpredictable for the RS as users may rate or have a positive/negative experience towards the movie titles they are presented with.  Accordingly, a user in bad mood might not be ready to watch sad movies like Me before you although it matches the user’s interest; likewise, a user in stress might tend to watch more light and relaxing movies such as the proposal instead of serious movies that matches their preference [3][5].

“Psychologically, users intend to make a conscious and unconscious selection over entertainment content that serves to maintain positive mood and repair or diminish pain in terms of both intensity and duration [6]”. Users often express and avail themselves of the opposites of what they are experiencing in their emotional states to feel better and overall have a better mood.

Thus we have mapped the mood out in accordance to the psychological aspects and as well as which mood is likely to be associated with which genre the best as example, a fearful state of mood would not appeal to the user to watch a horror movie as we have mentioned in the psychological reasoning above, thus we have mapped an adventurous movie to try and get better results to accommodate the user.

## Mood Mapping chart



*Figure 1 Mood to Genre mapping*

The positive emotion such as happy when selected by the user will pick onto horror results of the movie, excited for crime, peaceful for fantasy, and so forth, whereas the negative mood state upon selection of sad fetches drama titles, angry with romance titles and fearful with adventurous titles and so forth.

The advantage and credibility of the program are as stated earlier that users can select multiple moods as a user wouldn’t just select happy and leave it at that if he/she would want a substantial result, as mood can be expressed in more depth or diverse ways to get more accurate results, thus when a user picks happy, excited and content, he will get a variation of movies that contain horror/crime/mystery aspects to the movie with the highest (reviewed) titles from the dataset.

## Mood Recommender System GUI - TKinter

The RS has a GUI interface created using the python library “tkinter” which contains the Menu for the list of moods for users to select upon the 1st phase of the program. Upon the selection of moods, the program will search the dataset (IMDB and rotten tomatoes)

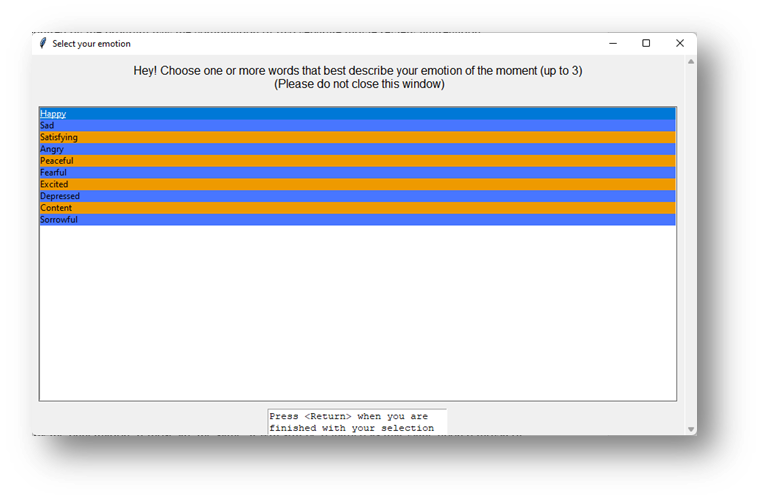
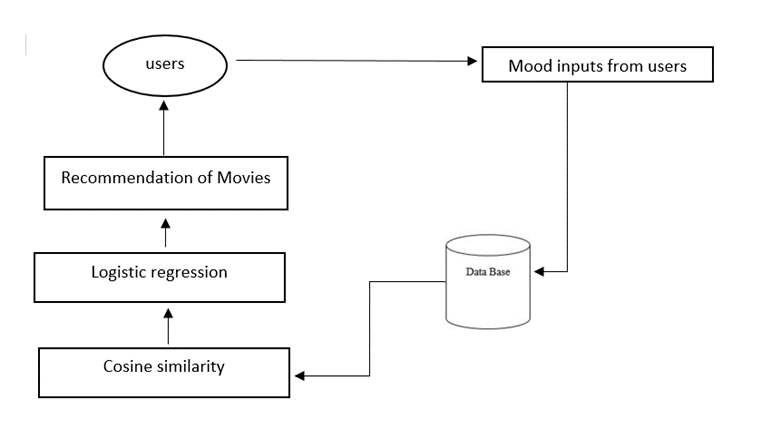


Figure 2 GUI of mood recommender system.

## Mood Recommender Architecture



*Fig 3 Architecture of Movie Recommender system using Mood*

# **Dataset**

## Data Background

The data acquired by the program was the combination of two separate movie review-aggregation database websites which were IMDb and Rotten Tomatoes. These datasets contained the desired attributes for our program such as movie genre, movie rating, movie score, movie review, maturity rating, and movie summary. Furthermore, in terms of data population, these two giants housed more than 10 million titles including their attributes and still counting as new movies are being made. The movie genre mainly was the paramount attribute as the mood was instantaneously calculated or rather derived from it.

## Data Collection Method

The data from IMDb and Rotten Tomatoes were acquired in the program simultaneously by using web crawling and web scraping. Movie scraping and crawling are the automatic and precise options of pulling and extracting information online from the URL such that its HTTP could be browsed, the way humans perceive it, and desired information could be pulled from it [7]. A web crawler was placed in our program to acquire the results or rather the information passed in the URL perimeter including the links to both IMDb and Rotten Tomatoes websites. Similarly, Web scraping, by using the Beautiful Soup library in our python program, was used to pull out or extract specific desired data from those links and saved in the program for further processing and analysis which would give us our results.

## Data Preprocessing

Preprocessing was carried out for the dataset to further refine the data set to eliminate any ambiguities or consistencies from the data and clean it up for accurate data retrieval and analysis.

* The Natural Language Toolkit (NLTK) library was used to preprocess the data utilizing lowercasing, removing punctuation, spelling correction, and removing stop words. First of all, lowercasing was carried out by python for the dataset such as the different variations of words both capitalized and not, will be converted to lowercase values providing the program to easily interpret all the values as the same even though they are capitalized. Similarly, the removing of the punctuations from the data set helps in identifying the value even though they are concatenated or have some sort of punctuation residing. In this manner even though the values are different from each other as being altered by the punctuation, if those are the same, it will still be regarded as that value upon removal of punctuations. Finally, stop words have been added in the program to deliberately remove the inclusion of the common words or the ones specified by the program. Such a technique gives accurate analysis and data retrieval upon request as only specific terms are included in the dataset and the commonly used words as definite articles are removed.
* Secondly, another preprocessing technique that was used was with the help of normalization. This being used in our program will organize the data in terms of removing noise, and complicated and duplicated data to ensure that the dataset is accurate and for easy navigation. This was carried out with the help of *json\_normalise* from the panda library. This refines the dataset by normalizing semi-structured JSON data into a flat table. As seen in Figure 4 below, the unprocessed dictionary data output scraped from the web has a considerable amount of noise, unnecessary punctuation, and stop words. Figure 5 shows the normalized data with the exclusion of the specified unnecessariness as stated above leading to a more structured table.



Figure 4 Unprocessed Data



Figure 5 Normalized and Structured Data

# **V . Methodology**

## Cosine Similarity and Count Vectorizer

Cosine Similarity is a machine learning tool to find out the similarities between two objects or vectors. As seen in Figure 6, this algorithm tends to find the dot product between the two vectors and analyze the angles between them such that by calculating the dot products of the two vectors, the final angle achieved determines the similarity between the two. Adding on, the higher the Cosine Similarity, the closer the vectors and the angles (theta) between them are, which thus determines that the two vectors are similar [8].

Moving on, in conjunction with cosine similarity, a count vectorizer as a vectorizing technique is used on the given value, supposedly words or numbers, to convert textual data into vectors for cosine similarity to give accurate results.

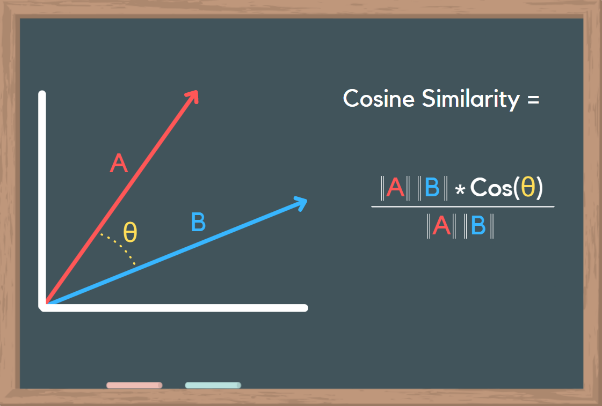


Figure 6 Cosine Similarity Formula and Theta Analysis

To carry out the cosine similarity process in python, the library of cosine similaritywas imported which provides the appropriate methods and analysis for the cosine similarity function by calculating the dot product of the values as seen in Figure 4 above. To begin with, the count vectorizer library was used and imported into the program to convert the text documents into vectors for the cosine similarity to calculate the dot product. As seen in Figures 7 and 8, after normalization as seen in Figure 5, each of the outputs that the scraper fetches and extracts were treated as a parameter in the count vectorizer function and therefore converted to vectors. Each of the entries in the data set preferably ‘movie title’, ‘genre’, ‘maturity rating’, ‘rating’, ‘length’, and ‘summary’ was converted to a specified digit with 0s and 1s which uniquely identifies the vectors for the dataset lines as described above. Mainly the vector for the genre was taken into consideration the most when looking into this specific program as later, the cosine similarity would compare mood and genre using these vectors.

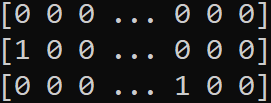


Figure 7 Post-Vectorized digits



Figure 8 Texture data to Vectorized data Conversion

Furthermore, as seen in Figure 9, the below table was fed into the system such that it could indicate which genre the system was aiming for in cosine similarity when choosing a specific mood whereby certain moods had a specific genre-related to them as explained by the research previously. As shown in Figure 10, the moods will be displayed in the GUI to be chosen and will correspond to the specified genre.

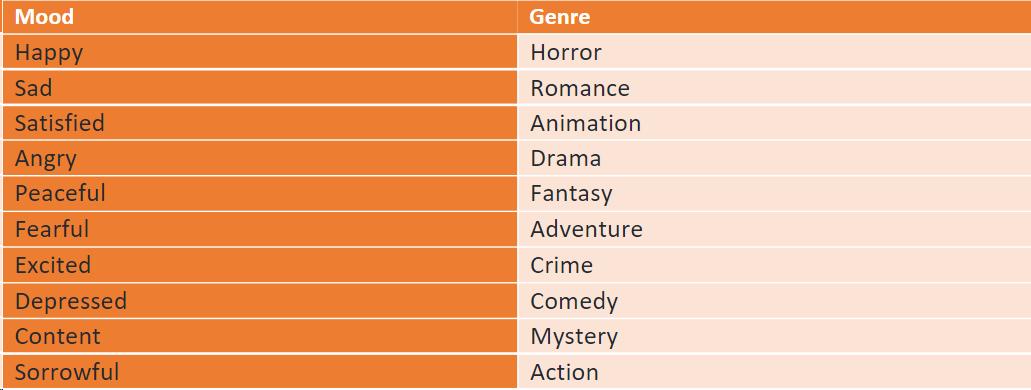


Figure 9 Mood and Genre Comparison

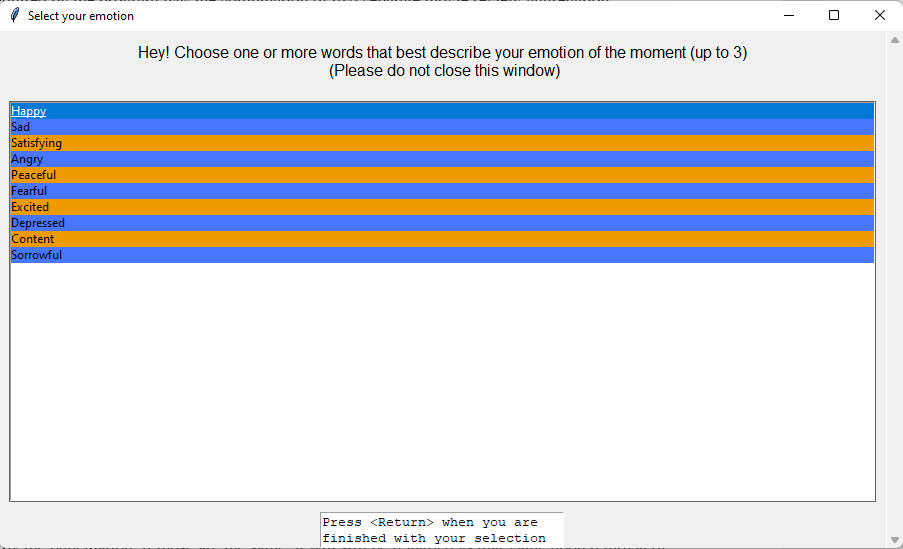


Figure 10 selection of Mood

As shown by Figure 11, the above table will further be applied the same count vectorizer algorithm to get the genre vectors from the mood that will further be put into the cosine similarity dot product to get the value of cosine similarity. Adding on, as seen in Figure 12, the dot products of the genre vector and the data set vector will be calculated to give the value of cosine similarity.



Figure 11 Texture data to Vectorized data Conversion for Genre

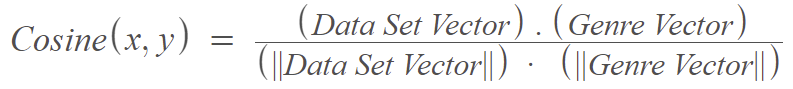


Figure 12 Cosine Similarity

Moving on, Heat maps usually indicate shades of colors and renders data that has a high throughput and higher intensity heat maps conclude to higher accuracy. For the second iteration of the recommender system, a heat map was used to show the intensity of the cosine similarity value amongst the other given data, and the most intense heated element was chosen to give the correct results. We used a 3x3 heat map for the top 3 results for the second iteration of the program. This was utilized by the library seaborn whereby it rendered the heat map and provided the intensities for it.

## Logistic Regression and Rating

Furthermore, Logistic Regression is a supervised machine learning algorithm that is used to analyze the data and also predict and classify continuous data. This uses a sigmoid function to analyze data as well as graph out the log-likelihoods to determine an accurate coefficient for the model [9]. After finding the cosine similarity, logistic regression provided the solution to iron out data that corresponds to the mood but most of all arrange the output or movies with the top ratings such as the movies with good ratings will be recommended first. This was done through the help of a log with Numpy library which gave the output as a natural logarithmic value of x which arranged our highest movie ratings through the number of ratings per review given by people for each movie. This thus approximated whether the movie will be chosen at the top rating spot or below average. The above calculation was done with the formula:

**log(exp(x)) = x**

## Accuracy Analysis (Training & Testing)

Finally, the overall accuracy of the model including the cosine similarity with the logistic regression was determined by training and testing the data. The accuracy of the model decided how good the system was as it gave the percentage of the values that were correctly predicted with the test data by training the model [10]. This was carried out by supervised learning which as stated above through training and testing the data the accuracy could be improved. Using cosine similarity with logistic regression will allow the model to not only force the value or weights to determine output but also learn the model through various supervised learning variations and minimize the differences between actual and predicted labels giving a favorable accuracy of the model [11]. Furthermore, using the library train\_test\_split from sklearn the dot product of the training and testing data were taken as parameters whereby using the split function, the training and testing data were split into 70% and 30% respectively. Also using the accuracy\_score library the accuracy of the model by supervised learning was derived through the function:

accuracy = (dot (a, b)/(norm(a)\*norm(b))) \* 100

# **Analysis and Results**

## Cosine Similarity

Moving on, as explained in section a. of analysis, the vectors of both the dataset and the given mood to genre vectors that were fed in the system was taken into the cosine similarity formula to find out the dot products of these two vectors such that according to the similarity score, it could be found out whether there were any similarities between the two in terms of the dataset having the same genre as the one indicated by the mood to genre table.

As indicated in Figure 8 and Figure 11, the vector units were calculated, and thus as shown in Figure 12, those were fed into the cosine similarity equation. The below output of cosine similarity was calculated when the system was asked to look for movies based on the mood Happy. As seen in Figure 8, the vectors for the mood ‘Happy’ were a bit identical to the vectors of the movie ‘Vikram’, whereby not all of the vectors resided equally with each other but have a significant similarity with the first four as it indicated the same genre vectors. Adding on, that was instantly caught by the cosine similarity equation which computed the dot product and gave a higher cosine similarity score of 0.89897 as seen in Figure 13. This was further run with all of the other movies to find the same computation of similar vectors through genre and the top cosine similarities score was taken into consideration by the model to be recommended that movie upon the chosen mood.

Fig 13. Cosine Similarity value from dot product for Horror

=

= 0.89897

Furthermore, upon deriving the cosine similarity graph as seen in Figure 14, the angle(theta) between the two vectors, genre, and dataset genre as described above was seemingly smaller which indicated that there was a strong similarity between the two vectors. Adding on, a high theta value would have indicated that the two vectors are not similar such that their cosine similarity was lower.

Figure 14 Cosine Similarity Graph



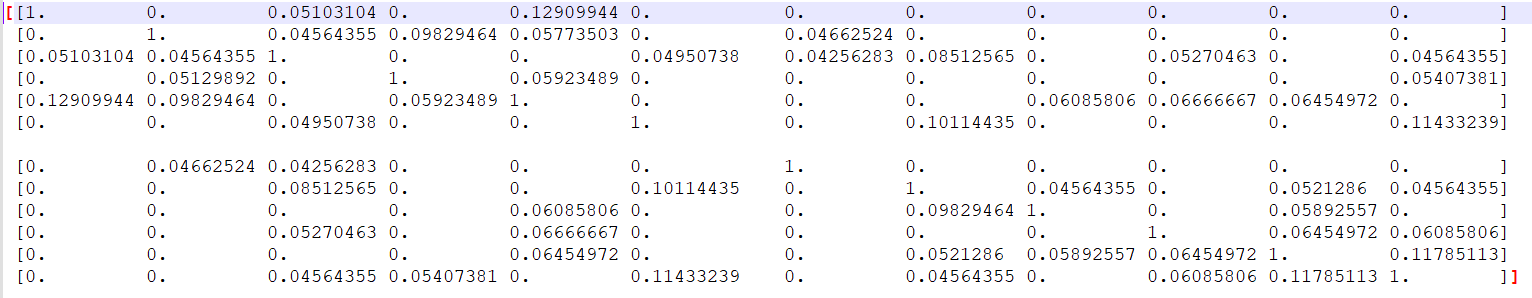


Figure 15 Cosine Similarity output for the Mood and Genre (a portion of it)

Finally, as seen in Figure 15, the similarity score for the mood ‘Happy’ was derived and computed as the dot product with all of the movies and as seen by output whereby there were some scores higher than the rest as those movies’ genres were closer if not identical to the chosen mood’s genres.

## Logistic Regression and Rating

Moving on, as shown in Figure 16, The individual rating intervals were taken and through the process of scraping, the number of interval ratings was identified by the reviews made on a specific and thus the logistic regression value was calculated which was the natural logarithmic values of these parameters. As shown below the highest natural logarithmic value of the rating interval was from 7.5-8.0 and 6.5-7.0 and this suggested that most of the people that reviewed the movie gave it a score of 7.5-8.0 and thus the overall recommendation for this movie would be placed after the movies that have the rating of 8.5-9.0 in the list. Adding on, if the logistic score of another movie had a higher score on the rating 9.5-10.0, that would most certainly be placed at the top of the list, and similarly if the score of a movie was higher on the rating 6.5-7.0 that would have been placed after the 7.5-8.0 rating as the list would be in descending order. Taking the highest logistic score would increase the credibility of the ratings as the highest number of reviewers have given that rating to a movie.

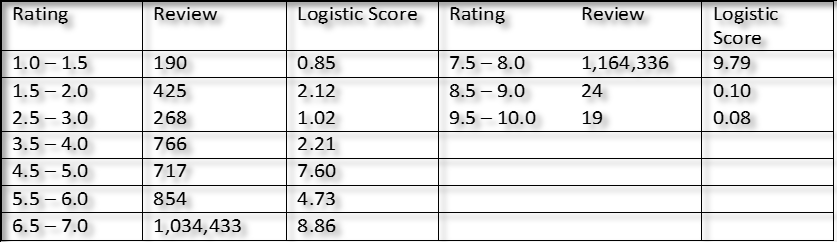


Figure 16 Logistic Score Table

Moving on, as shown in Figure 17, the logistic regression graph accurately plots the ratings vs score graph whereby it can be seen that the highest logistic score was the rating 7.5-8 and as seen by the GUI in Figure 18, it gives the recommended movies based on that highest ratings in descending order.

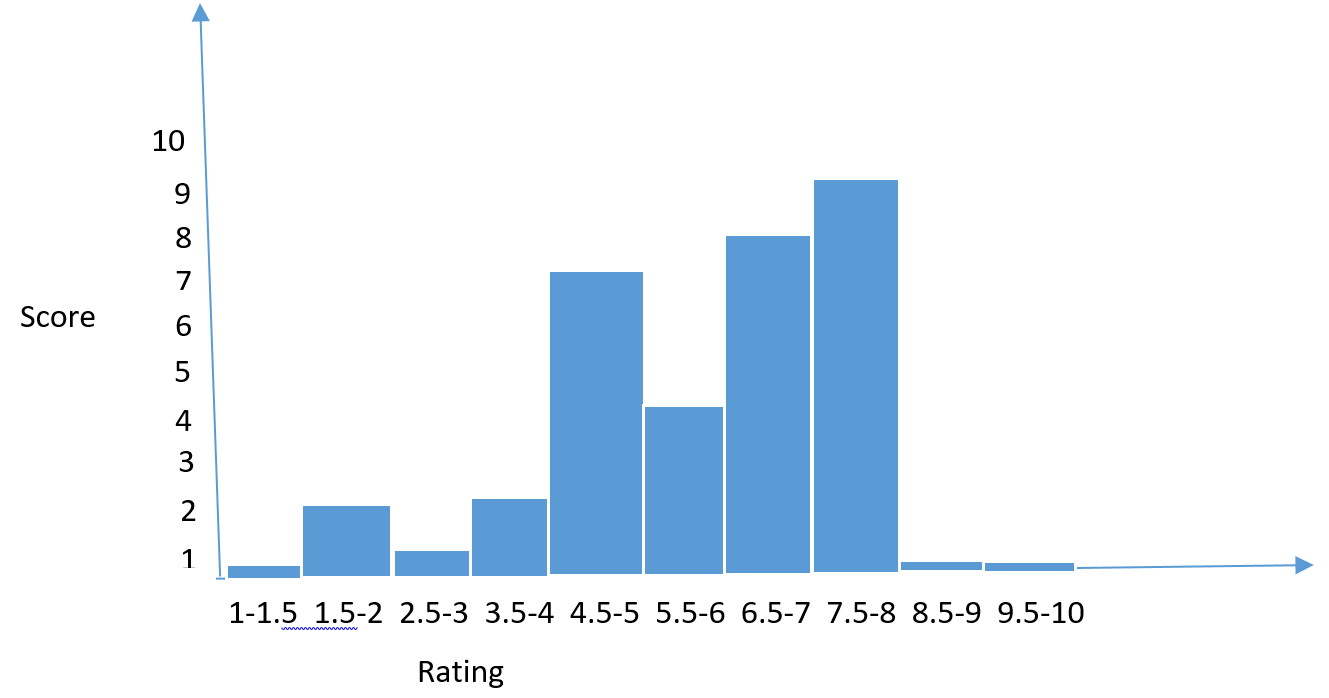


Figure 17 Rating vs Logistic Score Graph

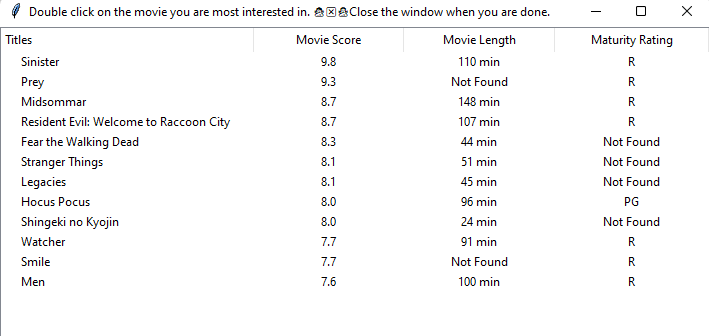


Figure 18 Movie Recommender

## C . Heat map for the second iterative movie recommender

Lastly, in terms of the heat map as explained in the analysis and as seen in Figure 19, the heat map accurately stated by using a 3x3 matrix which of the top three movies were chosen for the second iterative movie recommender. As shown below, amongst the chosen cosine similarity matrix, the top-rated movies were chosen from the logistic function and as seen by the graph, the intensity of the heat map was the highest at 1, 0.555, and 0.123. These values thus indicated the top 3 movies that were to be included in the iterative recommender and as shown by Figure 20, the top 3 movies were being displayed when clicked on a movie in Figure 18, and thus were derived from the cosine similarity as shown in the heat map.

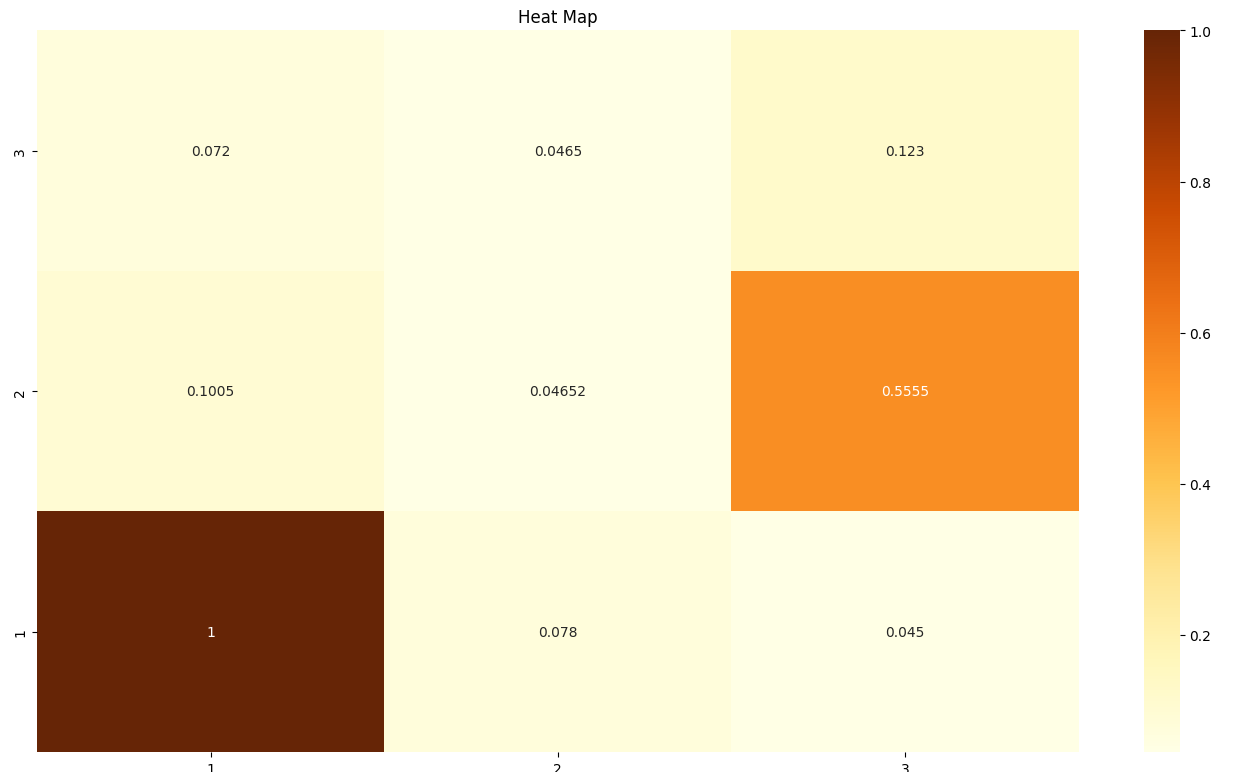


Figure 19 Heat Map

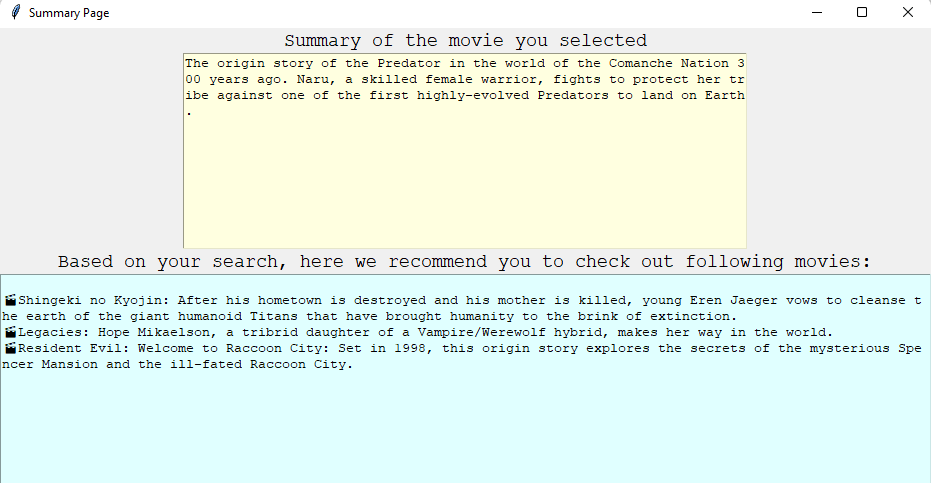


Figure 20 Second Iteration of Recommendation

## Accuracy (Training and Testing)

Finally, as explained in the analysis, the data from the cosine similarity and logistic regression was trained and tested by the principle of supervised learning whereby the dot product was taken to get the accuracy by training 70% of the data and testing it on 30% of the data. As shown in Figure 21, through the 30 iterations that were done, the accuracy was calculated for those iterations and as seen by the table below, the accuracy was above 60% which determined the model to be seemingly better at giving results but later shown to spiked at 90% which lead this model to be more sophisticated and thus more accurate in giving the recommendations.

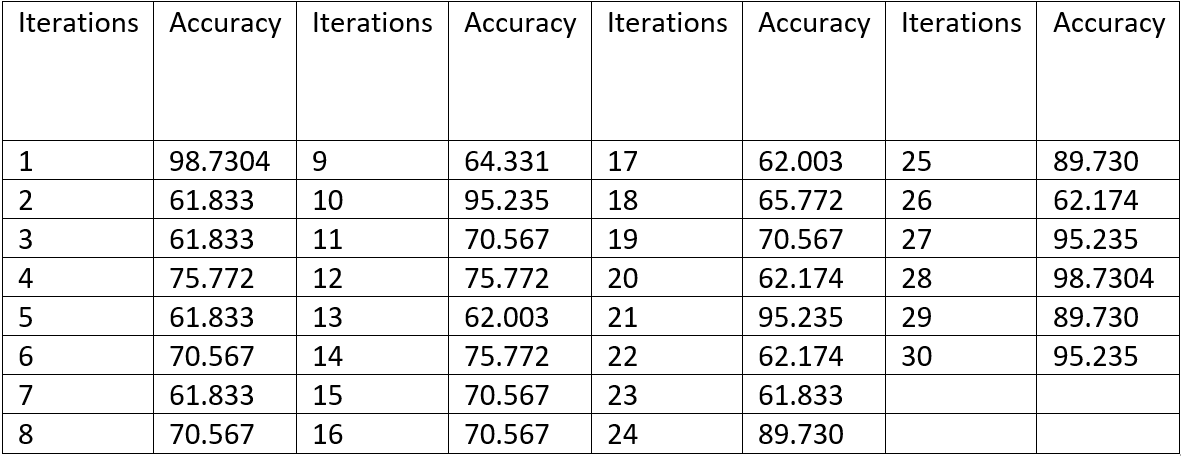


Figure 21 Accuracy of the final model

Moving on, as seen by the graph in Figure 22, as the data was trained and tested upon supervised learning, the model adopted an increase in accuracy after the 18th iteration and spiked the accuracy at the rate of 90%. This thus solidified the idea of supervised learning as upon training the model, it learned the trained data and upon testing, gave a very desirable accuracy.



Figure 22 Graph of Accuracy

# **Future work**

A further improvement could be done on the model in regards to user satisfaction criteria in terms of how well the system recommended the movies based on the user's desired mood. Such a program like this can be implemented by having a rating system input from the users based on how well was it executed in terms of giving the specific and desired movies based on the user choice of mood selected. Furthermore, this rating system can be saved in the dataset and the next time, the recommendations are given to the user, that rating will be taken into consideration in the program to give more appropriate and accurate movies to the user based on their desired mood.

# **Acknowledgment**

We would like to thank Dr. Anurag Sharma for guiding us with our project and giving us essential information and resources in perfecting our program for our report and presentation.

# **Conclusion**

The mood is defined as "a feeling often brought on by a significant event for the person. It often involves a conscious mental state that is aimed towards some object and has a discernible psychological content, physical disturbance of some type, identifiable facial expressions, speech tones, and body language, a state of preparedness for certain actions” [12]. The mood has been incorporated in our program such that it correlates with the genre in the program and using algorithms such as cosine similarity and logistic regression, it will be able to present the recommended movies.

In a nutshell, the aim of successfully recommending movies to the user based on their chosen mood was achieved, and henceforth the analysis gave exceptional results as the model underwent supervised learning. Leading to the proper execution of this system, the problems of having long deliberate hours looking for a movie that suits your mood can be solved with the above-suggested model. The model was successful in recommending movies to the users based on their mood by giving specific genre movies that matched the mood. Furthermore, upon the display of the recommended movies, a second option or rather movies were recommended by using the same algorithm which recommended three more movies by the previous movies watched by the viewers.

# **References**

[1] S. Reddy, S. Nalluri, S. Kunisetti, S. Ashok and B. Venkatesh, "Content-Based Movie Recommendation System Using Genre Correlation", *Smart Intelligent Computing and Applications*, pp. 391-397, 2018. Available: 10.1007/978-981-13-1927-3\_42

[2] S. Sharma, A. Sharma, and Y. Sharma, "Recommender System using Hybrid Approach", 2016.

[3] P. Winoto and T. Tang, "The role of user mood in movie recommendations", Expert Systems with Applications, vol. 37, no. 8, pp. 6086-6092, 2010. Available: 10.1016/j.eswa.2010.02.117.

[4] I. Portugal, P. Alencar, and D. Cowan, "The use of machine learning algorithms in recommender systems: A systematic review", *Expert Systems with Applications*, vol. 97, pp. 205-227, 2018. Available: 10.1016/j.eswa.2017.12.020.

[5] J. Bennett and S. Lanning, "The Netflix Prize", 2007.

[6] M. Gardner, "Mood States and Consumer Behavior: A Critical Review", *Journal of Consumer Research*, vol. 12, no. 3, p. 281, 1985. Available: 10.1086/208516.

[7] Lawson, R. (2015). Web Scraping with Python. Packt Publishing Ltd.

[8] P. Xia, L. Zhang and F. Li, "Learning similarity with cosine similarity ensemble", 2015. [Online]. Available: https://doi.org/10.1016/j.ins.2015.02.024.

[9] A. Thanuja Nishadi, "Predicting Heart Diseases In Logistic Regression Of Machine Learning Algorithms By Python Jupyterlab", *Ijarp.org*, 2019. [Online]. Available: http://www.ijarp.org/published-research-papers/aug2019/Predicting-Heart-Diseases-In-Logistic-Regression-Of-Machine-Learning-Algorithms-By-Python-Jupyterlab.pdf.

[10] V. Nasteski, "An overview of the supervised machine learning methods", 2016. [Online]. Available: https://www.researchgate.net/profile/Vladimir-Nasteski/publication/328146111\_An\_overview\_of\_the\_supervised\_machine\_learning\_methods/links/5c1025194585157ac1bba147/An-overview-of-the-supervised-machine-learning-methods.pdf.

[11] K. Orkphol and W. Yang, "Word Sense Disambiguation Using Cosine Similarity Collaborates with Word2vec and WordNet", *https://www.mdpi.com/1999-5903/11/5/114*, 2019.

[12] Oatley, K, and Jenkins, J.M. Understanding Emotions: Blackwell, 1996.