Application for Stress Management/Mood-Based Recommendation System

Harshit Chouksey,
San Jose State University
Email: harshit.chouksey@sjsu.edu

Ashish Gaurav,
San Jose State University
Email: ashish.gaurav@sjsu.edu

Sayali Nilangekar,
San Jose State University
Email: sayali.nilangekar@sjsu.edu

Sankalp Tiwari
San Jose State University
Email: sankalpsheetal.tiwari@sjsu.edu

***Abstract*—Today, mental stress is a significant issue for all age groups. This rising stress level contributes to a number of issues such as depression, suicide, and heart attacks. An external stimulus could boost moods or enrich experiences. Music and movies have long been acknowledged as effective tools that greatly impact our emotions and general well-being.**

One of the most significant challenges in the domain of digital wellness is a lack of stress management tools catering to an individual's unique circumstances. People deal with diverse sources of stress in their daily lives, emphasizing the need for tailored and effective stress management solutions. Current tools often fail to address users' varied emotional states, leaving a market need for a more responsive and personalized approach. Existing solutions also fall short since they often offer limited types of recommendations, such as solely music, rather than a comprehensive solution.

This project aims to address the above-mentioned problems by developing a web application where users will fill out questionnaires, allowing the application to predict their stress and mood levels. This data is used by the application to provide customized recommendations for stress relief including movies and music with the help of Random Forest Classifier with Filtering for music recommendations and Content-based filtering using Cosine Similarity for movie recommendations. The outcome will be a reduction in the impact of growing mental stress and elevation of moods in our modern society.

I. INTRODUCTION

In today's world, there are a lot of factors that can impact people's mental health and well-being. This impact of stress on mental well-being is a widespread concern, causing health issues like depression and anxiety. Movies and music have always been one of the preferred ways to relieve tension. In this context, currently, there is a lack of effective personalized solutions to help with managing stress, creating a need for tailored and practical stress management tools. Through this research project, we want to develop an Emotion-based Music and Movie Recommendation System, integrated within a user-friendly web application. We aim to address the necessity and develop a customized tool catering to both stress relief and entertainment needs.

Our approach in this research project is to create a content recommendation system using a mood-based questionnaire, advanced machine learning, and personalized recommendations utilizing the latest tech stack. By getting users to answer a questionnaire, the app predicts mood and stress levels, offering recommendations accordingly. After providing content recommendations, we are also gathering user feedback on whether the recommendations are suitable for the user. The recommendations are then dynamically adjusted accordingly, to make sure they account for any out-of-the-box, user-specific needs. Our use of innovative machine-learning models makes our project stand out, providing a comprehensive and user-friendly experience.

This research brings together machine learning, user experience design, and insights from psychology to create a well-rounded solution. The resulting deliverable is a web app with features like user profiles, sentiment questionnaires, and core machine learning models, ensuring a complete and personalized mood-based entertainment tool. Special attention is given to user feedback on recommendations, real-time mood updates, and strict data privacy measures to prioritize user security and adherence to regulations.

Essentially, this project aims to develop a novel and personalized stress management system using innovative technologies and a user-centric design, making this study a valuable contribution to the integration of technology and mental well-being.

II. PROBLEM STATEMENT

People of all ages are impacted by the growing problem of mental stress, which has dire repercussions including a rise in the rates of anxiety, depression, heart disease, and suicide. Although it is commonly known that movies and music can have a positive impact on our mood and mental health, there are very few products in the digital wellness space that are especially made to address the many and various stress triggers that people encounter. Many of the stress management products on the market take a generic approach, which leaves customers' varied emotional requirements unmet. These solutions usually focus on making recommendations for a certain kind of material, like music, rather than giving a comprehensive stress-reduction plan that includes a variety of activities and entertainment options.

Given these circumstances, a sophisticated approach that can provide individualized, efficient, and complete services while also understanding the complexity of human emotions and stress patterns is desperately needed. This approach should make use of state-of-the-art technology. With the use of a comprehensive questionnaire and sophisticated machine learning algorithms, the proposed online application seeks to close this gap by establishing a system that makes recommendations for movies and music based on the user's mood. This novel method goes beyond the conventional, generalized solutions to offer tailored and adaptive recommendations. The project's goal is to greatly improve mental health management through personalized entertainment by creating this innovative, user-focused, and interactive digital wellness tool. This will be a big step forward in the combination of technology and mental health support.

III. MACHINE LEARNING MODELS

A. Datasets

1) Dataset for Music Recommendation Model: Instead of utilizing a third-party dataset, we built our own dataset for the music recommendation model with the help of the Spotify Web API, a robust service that offers extensive access to user and music library information. We used a function called "Get Track's Audio Features," which lets you extract different musical characteristics and metadata related to every song track. With the use of this feature, we were able to gain a thorough understanding of the audio qualities that help define and distinguish songs.

The dataset has 1167 unique song tracks and includes both categorical and continuous features. We have features such as 'Track_id', 'Name', 'Album', 'Artist', and 'Release_Date' that act as track identifiers and provide a means to explore the dataset. We also included audio features such as 'Acousticness', 'Danceability', 'Energy', 'Instrumentness', 'Liveness', 'Loudness', 'Valence', 'Speechness', and 'Tempo'. These features played an important role in song understanding and development of the recommendation models. Several modifications were implemented to get the dataset in line with our model requirements.

a) Exploratory Data Analysis (EDA): After intensive data cleaning, we performed elaborative descriptive analysis steps like correlation analysis, distribution of audio features, popularity trends, etc. This played a major role in performing feature selection and other preprocessing steps like data parsing, categorization, scaling, and model labeling.

From the correlation analysis, we understood the feature relativity with each other. Features like 'Valence' show a strong positive correlation with 'Energy' and a strong negative correlation with 'Acousticness' indicating that more acoustic tend to be less energetic and songs with high valence tend to have more energy. Features like 'Energy' and 'Danceability' show a strong positive correlation with popularity indicating that more danceable and energetic songs tend to be more popular.

The analysis of the release year that focused on the feature Valence revealed how the emotional content of music has changed over time as determined by Valence. Valence is a

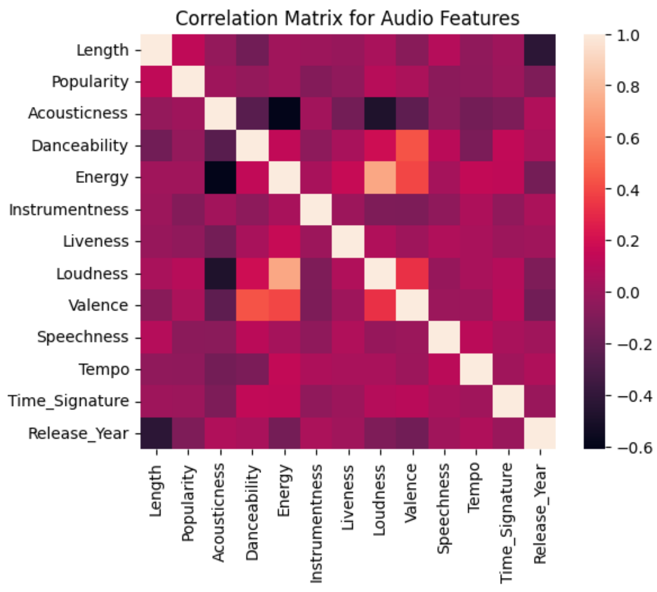


Fig. 1. Correlation Matrix for Audio Features

reflection of the positive musical tone a track delivers. We discovered from the investigation that it has a decreasing tendency, indicating a change towards tones that are more subdued or negative.

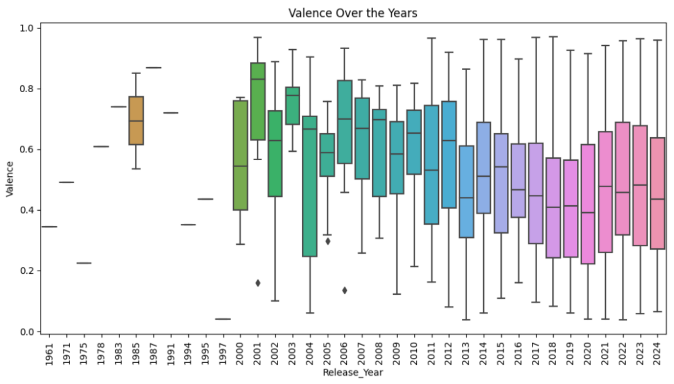


Fig. 2. Valence over the Years Plot

b) Data Preprocessing: For preprocessing, we extracted the release year by converting the 'Release_Date' into a DateTime object. The year was then divided into three time frames ('old, mid, and latest') based on the distribution of release years, which aids in customizing recommendations based on users' time preferences. Based on the EDA we extracted mood labels ('happy', 'sad', 'neutral') from audio features by applying specified thresholds to parameters such as valence, danceability, and energy. These features closely correlate with the emotional content of the music.

```
def assign_mood(row):
    if row['Valence'] > 0.6 and row['
        Energy'] > 0.5 and row['
        Danceability'] > 0.5:
```

```
        return 'happy'
    elif row['Valence'] <
        0.4 and row['Energy'] <
        0.4 and row['Danceability'] <
        0.4:
        return 'sad'
    else:
        return 'neutral'
```

Code Snippet 1. Mood Calculation

2) Dataset for Movie Recommendation Model: The dataset for the movie recommendation system was extracted from a very popular source of movie information known as "The Movie Database" also popular as TMDB. Our dataset consists of 139,072 entries, with each row representing a unique movie.

The features we have extracted are a mix of features essential for the model like the average user rating (vote_average), the total count of votes a movie received (vote_count), popularity, list of genres, and release year. And metadata that would be used when providing the user with the movie information like title, tmdb id, imdb id, and poster path.

a) Data Preprocessing: Our analysis of the dataset reaffirmed that the TMDB data was well-maintained and robust. There were no anomalies, inaccuracies, or missing values across all the critical columns. The missing data in the non-critical fields didn't have any impact on the analytical capabilities and performance of the model.

Some Structural adjustments were made to align the data with the analytical goals. We created a field 'release_era' based on 'release_year' which helped us in providing the user with the appropriate data from the given cinematic period. The categorization was defined as

1. Latest: Movies from 2023 onwards
2. Mid: Movies released between 2010 and 2022
3. Old: Movies older than 2010.

b) Exploratory Data Analysis (EDA):

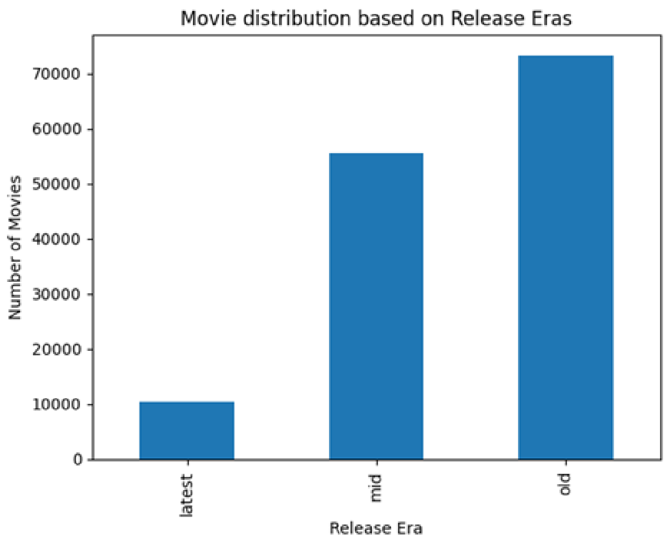


Fig. 3. Movie distribution on Release Era plot

Release Era Analysis: Analysis of the release era showed trends as:

Latest: 10,274 movies were released in or after the year 2023. This category contains the least number of movies because it has the smallest year range. However, it is crucial to understand the current trends and audience preferences.

Mid: 55,463 movies were released between 2010 and 2022. This represents modern cinema and has a wide variety of films.

Old: 73,335 movies are recorded in our dataset that were released before 2010. This is the largest category and contains a diverse range of movies from various decades.

Genre Analysis: Each movie has one or more genres with 19 distinct genres represented in the dataset. The most popular of the genres are drama with 46,992 occurrences, comedy with 33,457 occurrences, and documentary with 27651 occurrences. Each movie is associated with approximately 2.5 genres. This highlights the multidimensional nature of the movies.

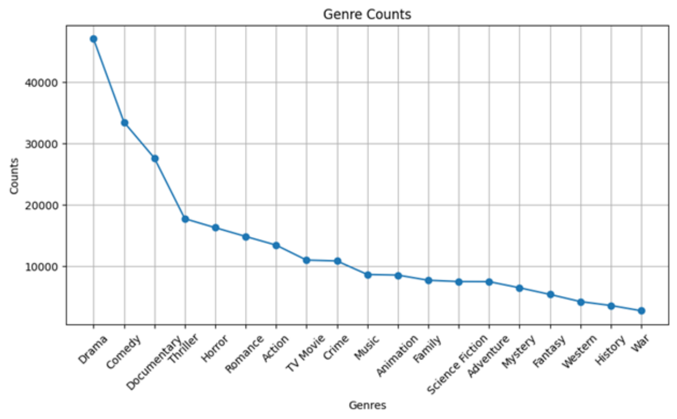


Fig. 4. Plot for Genre Count

Vote Average distribution: The calculated density plot shows that the vote average is approximately normally distributed, with its peak around the vote average of 6. This shows most movies receive average ratings with fewer of them getting high or low ratings.

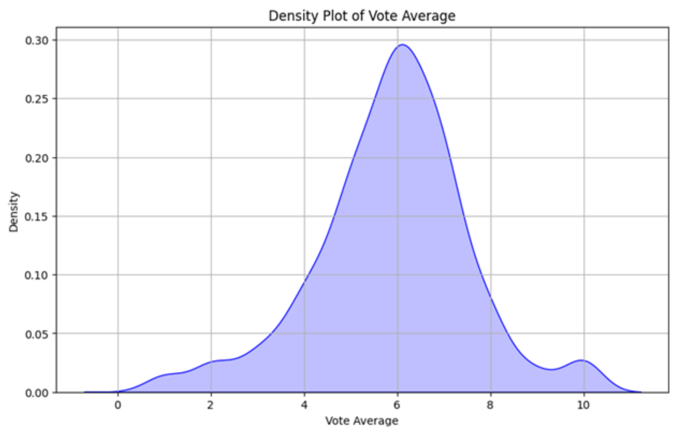


Fig. 5. Density Plot of Vote Average

Correlation Analysis: A Correlation matrix was created with selected features. That is vote_average, vote_count, popularity, release_year. And it showed these notable relations.

Vote count and popularity: There is a moderate positive correlation (0.26), indicating that more popular movies receive more votes.

Popularity and release year: A slight positive correlation (0.04). This shows that newer movies are slightly more popular, possibly due to increased accessibility and marketing.

Vote average and vote count: A weak positive correlation (0.12). This suggests that higher-rated movies might attract more votes. Even though the effect is not very strong.

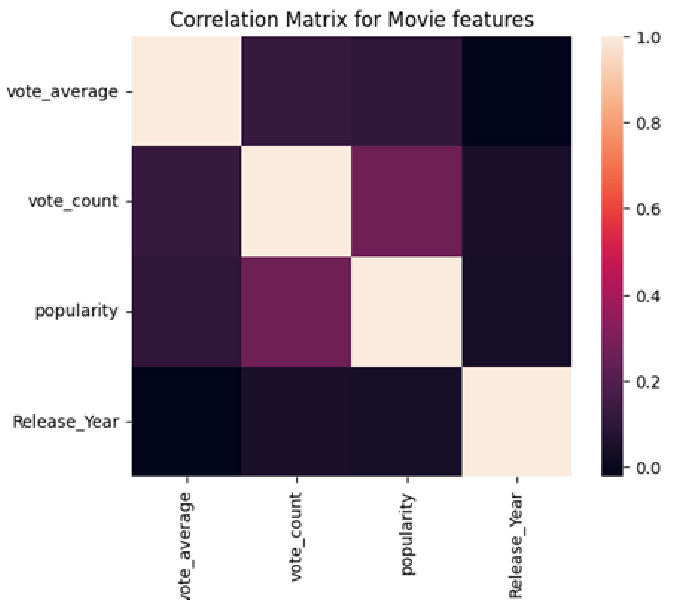


Fig. 6. Correlation matrix for Movie features

B. Music Recommendation Model

1) Comparative Analysis of Models: A music recommendation system's effectiveness, scalability, and satisfaction among users are all strongly impacted by the model that is selected. Three models that we tried for this project are compared in this section: the Multi-output Random Forest Classifier, the k-Nearest Neighbors (kNN), and the Collaborative Filtering model. Each model was assessed according to its performance, scalability, adaptation to user preferences, and suitability for processing music data.

a) k-Nearest Neighbors (kNN): kNN is a simple algorithm where the predictions are based on the closest training examples in the feature space. The simplicity of kNN is useful in applications with well-defined metric spaces and for quick prototyping. On the other hand, its performance deteriorates in high-dimensional spaces that can contain sparse and different music data. In our scenario, kNN is not able to scale well where we have real-time processing of user preferences and a large dataset. The accuracy of the recommendations is low and it is slow compared to other models.

b) Multi-output Random Forest Classifier: To predict numerous target outputs, this classifier uses an ensemble of decision trees, which makes it suitable for handling complex associations and interactions between different musical features. Compared to other models, it offers superior control over overfitting and stable performance, making it an excellent choice for capturing the subtle patterns seen in music data.

Random Forest can handle large datasets and data with high feature dimensionality but it is slow to update and process the real-time inputs and computationally heavy. Overall model accuracy is higher compared to other models but it suffers while handling the processing of real-time user preferences.

c) Collaborative Filtering: This is the most widely used algorithm for modern recommendation systems utilizing user-item interactions to generate personalized recommendations. This model works well as it continuously refines its predictions based on user behavior and adjusts to user preferences over time. It does, however, have issues with the cold start problem, which makes it hard to recommend new users or things with little interaction history. The model accuracy for collaborative filtering is better than kNN but falls behind the Multi-output Random Forest Classifier.

2) Multi-Output Random Forest Classifier with Filtering: For the music recommendation system, we used a combination of Multi-Output Random Forest Classifier and rule-based filtering as our final model. This hybrid approach utilizes the strength of both machine learning and filtering to provide personalized recommendations with higher accuracy.

The Random Forest Classifier employs an ensemble of decision trees to predict several target outputs at the same time, which in our case include categorical labels such as the song's mood and time preference (for example, era of release). It predicts the relevant categories for audio features of the songs and also provides insights into which features are most influential in determining user preferences using its feature importance score. In Random Forest, feature importance is computed by weighing the decrease in node impurity (Gini impurity for classification) against the likelihood of reaching that node. For the feature j, the importance I_j can be expressed as:

$$I_j = \sum_{t \epsilon T_j} p\left(t\right) .\Delta i\left(s_t , t\right)$$

where:

T_j is the set of all the trees where j is getting used, $p\left(t\right)$ is the proportion of samples reaching node t , and $\Delta i\left(s_t , t\right)$ is the reduction in impurity from splitting on feature j at node t.

After classification, a filtering layer is added to further refine the recommendations. Based on user input, this filtering process applies the desired release era preference (e.g., latest, mid, and old eras). The model is able to better fit the user's current preferences by applying these filters. By ensuring that the final recommendations are tailored to the user's needs and circumstances, this step increases the overall model accuracy and user satisfaction. The filtering model applies a set of user-defined rules R based on the user input u like the time

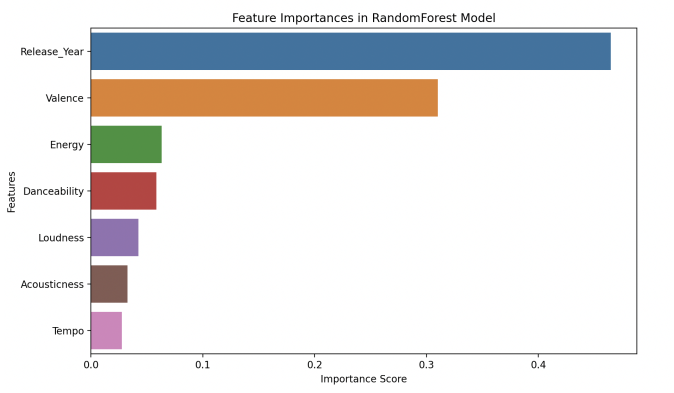


Fig. 7. Feature Importance in Random Forest Model

preference and the mood. The final recommendation set S is derived by applying these rules to the set of predictions P made by the Random Forest model:

$S = \{s\epsilon P : R(u, s)\, is true\}$

where $R(u, s)$ represents the rule that evaluates to true if song s matches the criteria mentioned in u.

This hybrid approach allows for dynamic adaptation to user preferences and can efficiently handle new and diverse datasets, making it particularly effective in the ever-changing domain of music streaming.

a) Importance of Mood Factor in the Model: We differentiated our approach from more traditional recommendation systems by giving the mood component a lot of weight in the development of our final music recommendation system. The model uses audio features including danceability, valence, and energy to categorize tracks into mood groups like "happy," "sad," or "neutral." The model is capable of predicting the emotional tone of songs thanks to its mood classification, which it acquired from RandomForest's strong handling of complex patterns. A filtering system ensures that the music matches not just the listener's taste but also their desired or current emotional context, thereby refining the output by matching song recommendations with user-specified emotional preferences. Our suggestion method is very sensitive to the subtleties of human emotion because of this integration of mood, which greatly personalizes the listening experience.

b) Algorithm: Multi-Output Random Forest Classifier with Filtering: 1. Music dataset created using Spotify API is loaded using pandas which include features like track metadata and audio features. Data cleaning and EDA steps are performed.

2. 'Release_Date' field is used to extract years and categorize the release years into different eras ('latest', 'mid', and 'old') which segments the song based on their release time.

3. 'Mood' label is derived from a tested formula using audio features like 'Valence', 'Energy', and 'Danceability'.

4. Sets of features (X) are defined that are used to train the model and the labels (Y) are extracted that the models will predict. The dataset is split into training and testing sets.

5. Random Forest Classifier is initialized with estimator = 100 and random_state set to 42. This is then used to create a multi-output classifier. The model is trained using the training set.

6. Feature importance scores are extracted and analyzed to understand the features most significantly influencing the prediction results.

7. The trained model is tested on the test dataset which is then passed to the rule-based filter based on the user's mood and time preference inputs.

8. Final song recommendations from the filtered list are uploaded to the web application.

9. end

C. Movie Recommendation Model

1) Comparative Analysis of Models: The usefulness of a recommendation model depends on the compatibility of the predictive model with the use case. To get better accuracy and user satisfaction we tried two machine-learning models for the movie recommendation system. A deep neural network (DNN) with TensorFlow and a content-based filtering approach with cosine similarity.

a) Deep Neural Network (DNN): The DNN model we implemented utilized a multi-input architecture that had separate input layers for user data, movie data, and movie genres. These inputs are then processed through embedding layers; this helps in capturing latent factors and interactions, which are then concatenated and fed through dense layers with nonlinear activation functions to predict user ratings. This architecture enables the model to learn complex patterns from both user behavior and movie characteristics.

The DNN model was chosen since the model can provide excellent personalization using user embedding, Capturing personalized preferences effectively. It is dynamically adaptive and can adapt to the changes in user preferences over time. Additionally, the use of embedding in the dense layers helps in handling large datasets, i.e. our movie data set effectively. That is very useful for real-time recommendation. Though as the data keeps on increasing the model might need significant computational resources due to its complex architecture. DNN is also prone to overfitting.

```
def preprocessing():
  encode_categorical_variables()
  normalize_numerical_features()
  extract_and_binarize_genres()def
      build_dnn_model():
  define_embedding_layers()
  define_input_layer_for_genres()
  compile_model()def train_model():
  fit_model_on_training_data()

def make_recommendations():
  identify_user_preferences()
  prepare_input_for_prediction()
  predict_ratings_for_candidate_movies
      ()
```

```
rank_and_recommend_top_choices()
```

Code Snippet 2. DNN Model

b) Content-Based Filtering: The content-based filtering model is designed to find similarities based on the content features such as genres and metadata. This approach is simpler and straightforward, focusing on the item attributes. By constructing a feature matrix for movies and applying cosine similarity, the system identifies the movies in the preferred genres of the users and those he has previously watched.

The content-based filtering model takes considerably less computational power compared to the deep learning approach. This allows scalability to a large number of items without large overhead in computation. This works well where the features are very well defined.

2) Content-based Filtering with Cosine Similarity: We used a Content-Based Filtering technique with cosine similarity as our final model for the movie recommendation system, enhanced with user-specific genre preferences. Using content attributes as a basis, this model provides tailored recommendations by leveraging the simple computational benefits of similarity measures.

a) Mathematical Foundations: The core of this approach is the cosine similarity calculation, which quantifies how similar two movie vectors are in terms of their content. The vectors represent the movies' features, such as genres encoded in a binary format. The cosine similarity S between two vectors A and B is defined as:

$$S\left(A,B\right) = \frac{A.B}{\vee A \vee \vee B \vee} = \frac{\sum_{i=1}^{n} A_i B_i}{}$$

Where A_i and B_i are components of vectors A and B, respectively, and n is the number of dimensions.

To integrate different scales, such as popularity and user ratings, with similarity scores, normalization is applied using:

$$N\left(x\right) = \frac{x - min\left(x\right)}{max\left(x\right) - min\left(x\right)}$$

This normalization ensures that all features contribute equally to the final recommendation score, which is a composite of weighted similarity, popularity, and ratings:

$F_j = \alpha W_{ij} + \beta N\left(popularity_j\right) + \gamma N\left(rating_j\right)$

Where $\alpha, \beta, \wedge \gamma$ are the coefficients that balance the contributions of each component to the final score.

b) Importance of Mood Factor in the Model: By adapting recommendations to the user's present emotional state, the mood element is incorporated into the content-based filtering using cosine similarity movie recommendation system, greatly increasing its effectiveness. This approach starts by aligning recommendations with predefined genre preferences specific to the user's moods, providing initial direction. It further refines these suggestions by analyzing the user's past movie ratings and the moods associated with these viewings, allowing the system to assign more weight to genres that consistently align more positively with the user during specific emotional states. By ensuring that recommendations are pertinent and appropriate for the context, this layered approach enhances personalization and increases user happiness and engagement. Through adaptively responding to both general and specific user preferences, the system provides precise and effective recommendations, minimizes choice overload, and stays relevant even in the face of changing user preferences. A responsive and user-centric recommendation platform must have mood-aware personalization because it gives a recommendation system a competitive edge by converting it from a static tool into a dynamic service that predicts and adjusts to user demands.

c) Algorithm: Content-Based Filtering with Cosine Similarity: 1. Initialization: A file is used to load the movie dataset. Features in the dataset include average votes, vote counts, popularity scores, movie IDs, titles, and genres. The database contains the user's preferred genres as well as historical watched data.

2. Data Preparation: Movie_era is calculated based on the release year. Genres for each movie are extracted and transformed into lists using the 'ast' module, ready for vectorization.

3. User Mood and Genre Preferences: Preferences are segmented based on user moods which influence genre preferences. These are expanded into a data frame, treating each genre associated with a mood as a separate entry.

4. Genre Encoding: The MultiLabelBinarizer is used to convert genre lists into binary format, creating a feature matrix suitable for calculating cosine similarity.

5. Similarity Calculation: Cosine similarity is computed between the user's preferred genre vector and all movie genre vectors, determining content alignment.

6. Weighted Scoring: Similarity scores are adjusted by genre preference weights, refining the recommendation relevance.

7. Popularity and Voting Adjustment: Movie popularity and voting scores are normalized and combined with weighted similarity scores to enhance the recommendation.

8. Recommendation Generation: Movies already watched are filtered out, and the remaining movies are ranked based on the final scores to select the top recommendations.

9. Output: Recommended movie titles and their genres are displayed, tailored to the user's current mood and preferences.

IV. METHODOLOGY

A. System Design

New users register for an account by providing basic information such as name, email address, password, and date of birth. After registration, users are prompted for recommendation-specific information, including preferred genres and mood indicators. On sign-in, users are always asked about their current mood and preferences using a questionnaire. This questionnaire includes the type of recommendations they want (movies/music), and other preferences that gauge their current interests. After getting the input from the users, the values are passed to the backend from where they are stored in the database and passed to the ML models. The models give the recommendations based on the input which is shared to the dashboard of the application for the user to

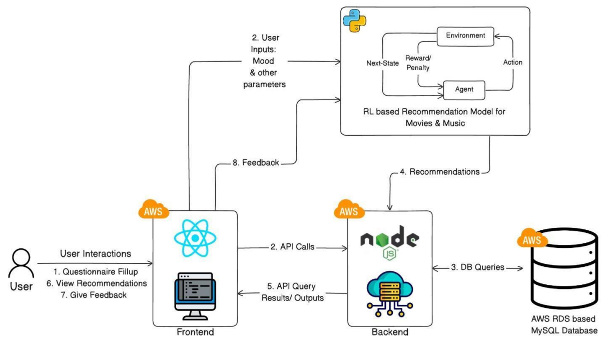


Fig. 8. System Architecture Diagram

view. The users give feedback for the recommendations which is passed to the recommendation models. These models input that feedback to update the recommendation accordingly. The users can update their mood preferences, genres, release era, and other choices at any time and they will get the updated recommendations all the time.

Our web application is designed to provide personalized recommendations for movies and music based on the user's mood and preferences. There are four main components in the architecture of our application: frontend layer, backend layer, database, and ML models. We have carefully selected the technologies for each of these components based on their features and compatibility with our project requirements.

B. Frontend Development

The frontend layer handles user interactions and displays the user interface. This layer is deployed on AWS EC2 servers, ensuring it's both scalable and reliable. We have used standard web technologies for structuring content, styling components, and adding interactivity to the user interface. The frontend application is based on React.JS and uses frameworks like Bootstrap and Material-UI for the UI design to ensure responsiveness, get reusable components, and enhance the visual appeal of the application, making development efficient. The frontend application runs on port 3000 and is compatible with all browsers and screen sizes.

C. Backend Development

The backend layer contains server-side logic for handling API calls to the database and proxy links to the ML models. We have developed the backend using Node.js and JavaScript and it runs on port 3001. REST APIs have been developed to enable seamless communication across different parts of our application. The API calls connect the frontend to the Database which is deployed in AWS RDS. The backend layer is also deployed on separate AWS EC2 servers for scalability and reliability.

D. Database Interaction

For our database, we are using AWS RDS, a relational database service based on MySQL. It stores user profiles and details collected during registration, questionnaire responses, music and movie data, and other relevant information needed for the application.

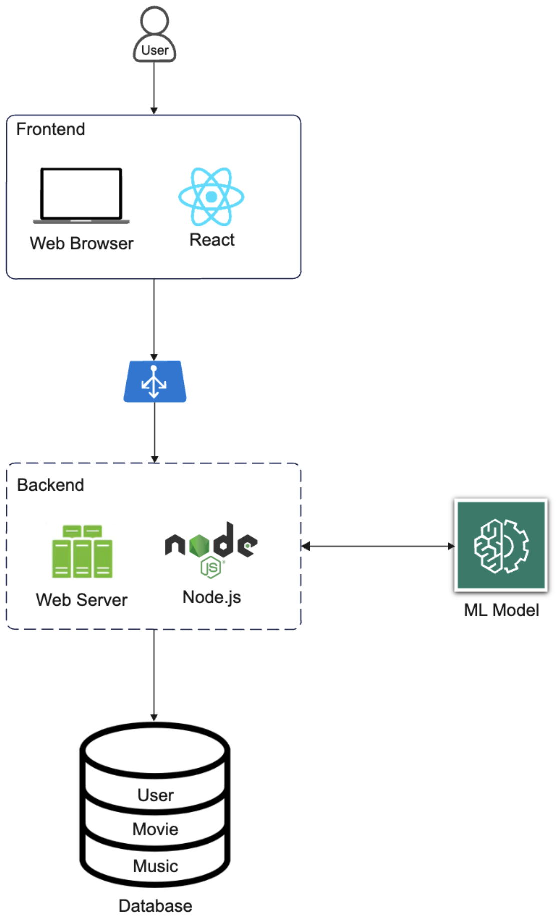


Fig. 9. High-level architecture diagram

E. Machine Learning Models Integration

The machine learning models are important components of the system and are integrated into the web application using the Flask framework and REST APIs. The flask framework acts as a second backend (the first is the node-based backend for the database) which runs separately on a different port (port 5000). It is based on Python and the trained models are imported to the flask to save time and resources. The API end-points facilitate communication between the backend server, the front end, and the recommendation models deployed on the AWS cloud.

F. Deployment and Cloud Infrastructure

Our web application is deployed on AWS cloud infrastructure, leveraging various AWS services for scalability, reliability, and security. We deployed our frontend, node-based backend, and flask-based backend on separate EC2 instances to handle the load and processing smoothly. We also configured the Elastic Load Balancer to spread incoming web traffic among several EC2 instances, guaranteeing high availability and fault tolerance. Auto Scaling groups are also set up to automatically scale the number of EC2 instances as needed, balancing performance and cost-effectiveness.

V. EVALUATION METHODOLOGY AND TESTING

A. Evaluation Methodologies for Music Recommendation Model

Metrics like accuracy, precision, recall, and F1-Score can be considered as the overall model performance metrics for mood-based recommendation systems that utilize a multi-output RandomForestClassifier and Cosine Similarity-based filtering.

*1) **Precision**:* It explains how many of the correctly predicted cases turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives. The importance of Precision is in music or video recommendation systems, e-commerce websites, etc. where wrong results could lead to customer churn, and this could be harmful to the business.

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive}$$

2) Recall: Recall explains how many of the actual positive cases we were able to predict correctly with our model. Recall is a useful metric in cases where a False Negative is of higher concern than a False Positive.

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative}$$

3) F1-Score: F1-Score gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.

$$F1 = 2.\frac{Precision * Recall}{Precision + Recall}$$

4) Confusion Matrix: Confusion Matrix is a performance measurement for machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values.

5) K-Fold Cross Validation: To examine each data point, K-Fold Cross-Validation is a reliable statistical method for assessing the stability and performance of machine learning models. The data set is split into 'K' equally (or nearly equally) sized folds or subsets using this process. Of these folds, 'K-1' is utilized to train the model, and the remaining single fold serves as the test set. This procedure is carried out 'K' times, with each fold acting as the test set precisely once, enabling a thorough evaluation of the complete collection of data. By averaging the model's performance over several subsets, this method helps mitigate the overfitting problem and provides a more impartial and general evaluation of the model's effectiveness.

6) Root Mean Square Error/Mean Square Error (RMSE/MSE): Mean Square Error (MSE) and Root Mean Square Error (RMSE) are important evaluation metrics that are primarily utilized in regression analysis to quantify the discrepancy between values seen from the environment being modeled and values projected by a model. The average squared difference between the estimated and actual values is known as the mean squared error, or MSE for short. The square root of the mean square error, or RMSE, offers an accuracy metric that is sensitive to scale and is reported in the same units as the response variable.

7) A/B Testing: A/B testing, also referred to as split testing, is a potent assessment technique that compares two iterations of a single variable. Usually, it involves comparing a subject's reaction to variant A against variant B and identifying which variant works better. When comparing methods, validating model enhancements, or evaluating modifications to preprocessing stages, A/B testing is an essential tool.

B. Testing and Verification

In the development of our web application, we have made sure to test our code thoroughly to make our application robust and fault-tolerant. This includes unit testing for frontend and backend code, and also model testing which has been covered more thoroughly in the Evaluation Methodology section. We have implemented frontend component unit testing using the Jest testing framework.

1) Unit Testing with Jest: Jest helps create test suites that test individual code components. We aimed for these key aspects:

Code Coverage: This ensures that every smallest individual part of code is tested thoroughly so that any issues can be discovered during development. Through Jest's coverage reports, we achieved 100% code coverage, which shows that all lines of code (statements, functions, branches) were exercised during testing.

Test Execution Time: We made sure to have efficient, simple, and small tests without repetition, to minimize test execution time. For the unit test suites we have built so far, our tests ran for around 2-4 seconds.

2) Cross-Browser Compatibility testing: An important part of the user experience is to make sure that the experience is uniform across different web browsers. To ensure this we included vendor prefixes like -webkit, and -moz in CSS styling, as well as modern standards using HTML5, CSS, and JavaScript standards supported by all browsers. CSS media queries have been utilized to apply styles based on different viewport dimensions. We conducted manual testing on different browsers to make sure that there were no discrepancies in the experience.

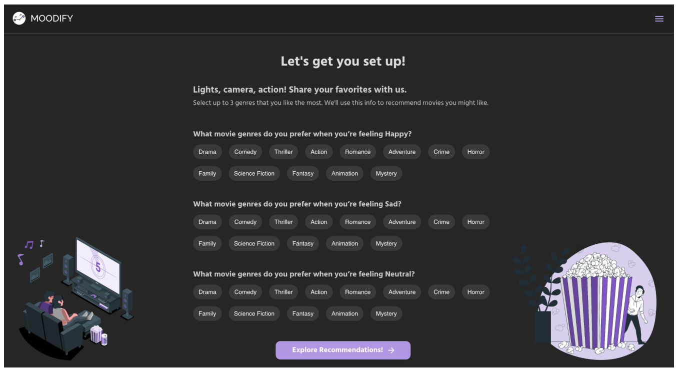
VI. RESULTS

Positive outcomes were obtained from the development and deployment of the stress management application, indicating its efficacy in stress management and mood enhancement via customized entertainment suggestions. Using advanced machine learning algorithms and a succinct mood-based questionnaire, the system was able to reliably anticipate users' current emotional states and propose tailored content such as music, movies, and hobbies.

The project met its primary goal of lowering mental stress and improving mood among users through tailored entertainment choices. The combination of technology and psychological insights to build this stress management application/mood-based recommendation system has demonstrated significant potential for improving mental well-being in today's digital ecosystem.

A. *User Engagement and Experience*

The web application's user-friendly interface enhanced the positive user experience by encouraging regular use and interaction with the content. The simple and elegant user interface allows the users to interact with the application effortlessly. The questionnaire pages allow the users to input their mood-related preferences which is passed to the models in real-time to fetch the updated music and movie recommendations.



Let's get you set up!

Lights, camera, action! Share your favorites with us.

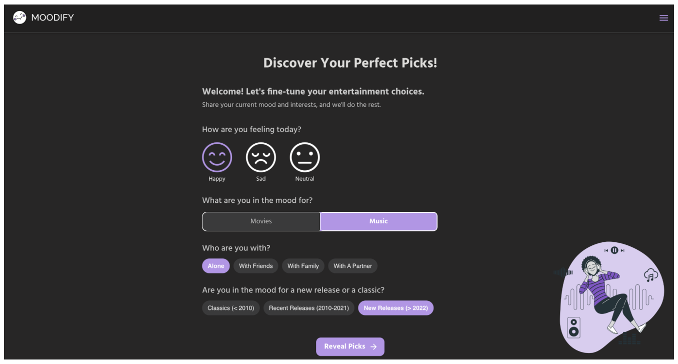
Select up to 3 genres that you like the most. We'll use this info to recommend movies you might like.

What movie genres do you prefer when you're feeling Happy?

What movie genres do you prefer when you're feeling Sad?

What movie genres do you prefer when you're feeling Neutral?

Fig. 10. Sign-up questionnaire



Discover Your Perfect Picks!

Welcome! Let's fine-tune your entertainment choices.

Share your current mood and interests, and we'll do the rest.

How are you feeling today? [✓ Happy]

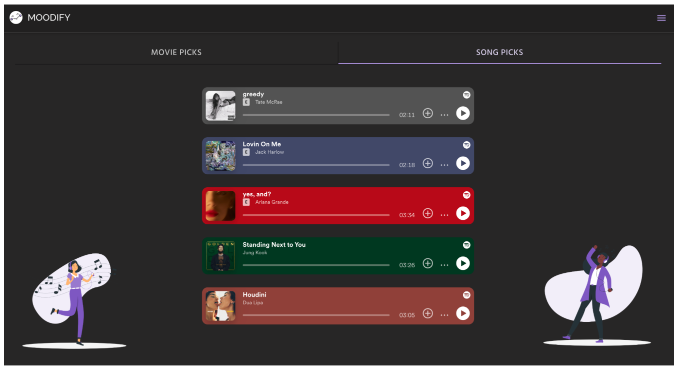
What are you in the mood for? [✓ Music]

Who are you with? [✓ Alone]

Are you in the mood for a new release or a classic? [✓ New Releases (> 2022)]

Fig. 11. Sign-in questionnaire (Mood preference)

Based on the sentiment questionnaire results, the machine learning models used in the recommendation system performed well in mood prediction. The relevancy of the recommendations was significantly high, demonstrating that the models are excellent at understanding and responding to specific user needs and preferences.



greedy — Tate McRae — 02:11

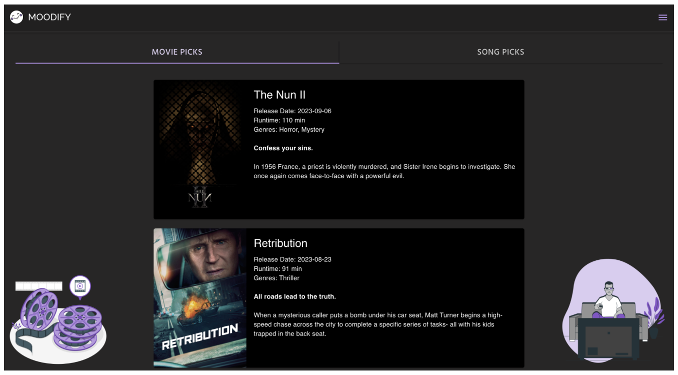
Lovin On Me — Jack Harlow — 02:18

yes, and? — Ariana Grande — 03:34

Standing Next to You — Jung Kook — 03:26

Houdini — Dua Lipa — 03:05

Fig. 12. Music Recommendations



The Nun II

Release Date: 2023-09-06
Runtime: 110 min
Genres: Horror, Mystery

Confess your sins.

In 1956 France, a priest is violently murdered, and Sister Irene begins to investigate. She once again comes face-to-face with a powerful evil.

Retribution

Release Date: 2023-08-23
Runtime: 91 min
Genres: Thriller

All roads lead to the truth.

When a mysterious caller puts a bomb under his car seat, Matt Turner begins a high-speed chase across the city to complete a specific series of tasks- all with his kids trapped in the back seat.

Fig. 13. Movie Recommendations

B. *Performance of Music Recommendation Model*

To evaluate the performance of the music recommendation model: Random Forest Classifier with Filtering we used metrics like precision, recall, F1-Score, and mean CV score to assess its predictive accuracy and its ability to meet user-specific needs through filtering.

We generated the classification report of the Random Forest Classifier focused on 'Mood' as an output label which gave us an idea about model performance in terms of precision, recall, F1-score, and support.

```
Classification Report:
              precision    recall  f1-score   support

       happy       0.96      0.95      0.96        82
     neutral       0.96      0.96      0.96       150
         sad       0.76      0.78      0.77        40

    accuracy                           0.93       272
   macro avg       0.89      0.90      0.89       272
weighted avg       0.93      0.93      0.93       272
```

Fig. 14. Classification Report for Music Recommendation Model

1) Precision: We got a summarized report of the performance of the classification model for the three mood classes: 'happy', 'neutral', and 'sad'. The overall precision of the model is high. Precision explains how many of the correctly predicted cases turned out to be positive. 96% of the instances predicted as 'happy' were actually 'happy'. 96% of the instances predicted as 'happy' were actually 'happy'. 76% of the instances predicted as sad were sad indicating lower reliability in prediction for 'sad' class as compared to classes.

2) Recall: A similar performance level was achieved in terms of recall as well. Recall explains how many of the actual positive cases we were able to predict correctly with our model. The model correctly identified 95% of all the actual 'happy' instances, 96% of all actual 'neutral' instances, and 78% of all actual 'sad' instances.

3) F1-Score: F1-Score gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall. 0.96 F1-score for 'happy' and 'neutral' showing excellent balance between precision and recall. Class 'sad' has a score of 0.77 highlighting poor performance for this class compared to others.

4) Accuracy and Model Average: The model accuracy is 93% indicating that the model performs well overall across all classes. The model has an f1 score of 0.89 without weighting by class size and a score of 0.93 with each class weighted by the number of true instances. This shows the model's effectiveness considering the class imbalance.

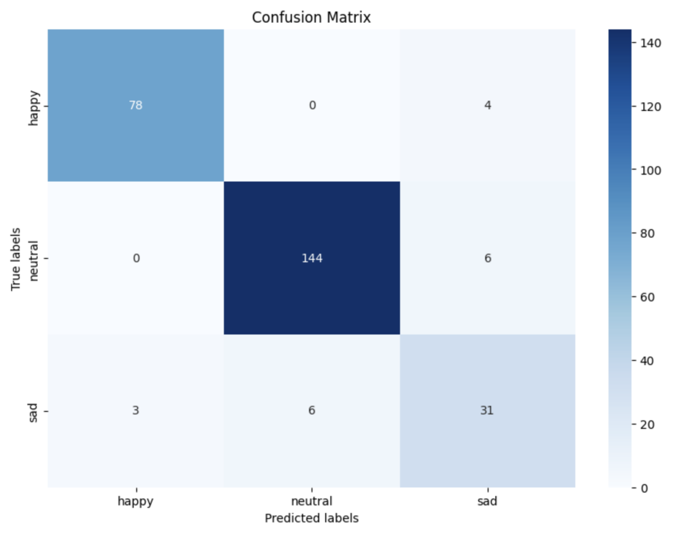


Fig. 15. Confusion matrix

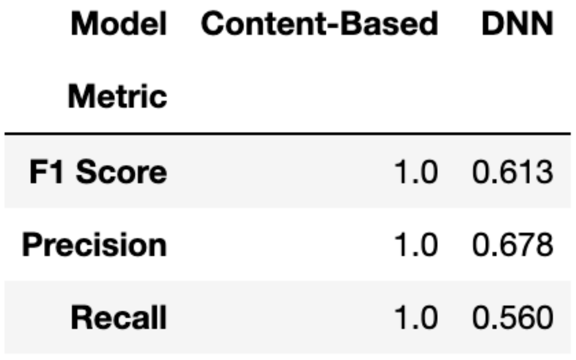
5) Confusion Matrix: The confusion matrix also indicates that the overall accuracy of the model is high, especially for the 'happy', and the 'neutral' predicted classes. The 'sad' class data points are less compared to the other two classes which has caused notable misclassification of 'sad' songs as 'neutral' and 'happy'.

6) K-Fold Cross Validation: We also performed five-fold cross-validation where the dataset is split into five smaller sets, and the model is trained and validated five times, using each fold as a test set once and as part of the training set four times. This helped us understand the model's performance and stability across different subsets of the dataset. We got the following cross-validation scores: 0.91544118, 0.94117647, 0.89338235, 0.90073529, 0.52573529, and a mean CV score of 0.835 giving us a more realistic indication of the model's performance.

C. Performance of Movie Recommendation Model

To assess the effectiveness of the movie recommendation system we implement the A/B testing to analyze the effectiveness of the Deep Neural Network (DNN) model and content-based filtering model. This method allows us to understand which model provides the most accurate and satisfying recommendations based on user-defined genre preferences across different moods.

We focus on three key metrics to measure the quality of recommendations: precision, recall, and F1-score. These metrics are calculated based on how well the recommended genres align with the genres that users have indicated a preference for under various moods.



Model	Content-Based	DNN
Metric		
F1 Score	1.0	0.613
Precision	1.0	0.678
Recall	1.0	0.560

Fig. 16. Content-based model vs DNN

The Content-Based Filtering model performs exceptionally well, with 100% precision, 100% recall, and 100% F1 score. These findings show that every genre this model suggests precisely fits the user's tastes and captures all pertinent genres without excluding any. With a precision of 67.8%, the Deep Neural Network (DNN) model, on the other hand, has less efficacy and suggests that only roughly two-thirds of the proposed genres match user tastes. Its F1-score of 61.3% indicates a poorer performance in striking a balance between precision and recall, and its recall rate of 56.0% shows that it fails to identify a sizable chunk of the genres consumers favor. This comparison demonstrates how much more consistently accurate genre recommendations in line with user preferences can be obtained using the content-based filtering approach.

The main reason the content-based filtering approach performs so well is that the genres it recommends are heavily impacted by the genres that users have indicated as their preferences. The way that this model uses explicit user preferences guarantees that the recommendations are extremely pertinent and customized. Conversely, the DNN model displays poorer precision, recall, and F1 scores, suggesting a mismatch between the genres it recommends and the users' real preferences. This is likely because the DNN model combines a wider range of characteristics and may target less precisely.

These results clearly show that, in this case, content-based filtering provides a more dependable and user-aligned recommendation technique. To better satisfy user expectations and improve their viewing experience, we can use this data to inform future optimization and improvement of our recommendation algorithms.

1) MSE/RMSE: A model-predicted rating is calculated for the content-based filtering using the below formula:

$$Predicted\ Rating = Adjusted\ to\ 10\ scale(\ Normalized\ Similarity\ Score + Normalized\ Popularity + \ Normalized\ Vote\ Average)$$

The actual movie rating was taken from 50 users and MSE and RMSE were calculated, this gave us a MSE of 3.72 and a RMSE of 1.929. This shows that most predictions are close to the actual user rating. With the bigger dataset and user information, and fine-tuning the model there is room for improvement.

VII. Related Work

Mood-based recommendation systems have been widely used and researched for their ability to improve user experience by delivering customized information aligned with their emotions. Significant contributions have been made in the domain of sentiment analysis and mood-based recommendation systems for movies and music that are critical for stress management.

S. Jain et al. (2009) proposed a depression analysis and suicidal ideation detection system that detects the depression level of users using questionnaires [1]. These questionnaires are similar to Patient Health Questionnaire -9 (PHQ-9) and contain questions based on features like age, degree of insomnia, appetite, etc. The answers to these questions and other parameters are fed to various classification algorithms such as Logistic Regression, Decision Tree classifier, and XGBoost algorithm. L. Surace et al. (2017) explored mood recognition systems in real-world settings using Deep Neural Networks (DNN), showcasing this potential [2]. The author uses a combination of DNN with a bottom-up approach and a Bayesian classifier with a top-down approach.

P. Winoto et al. (2010) discussed the impact of user's moods on movie recommendations [3]. The authors studied the effect of different user moods on their movie ratings by conducting research on students during the exam month. From this research, the authors found that the user's ratings for romantic-comedy movies are impacted by their mood, but it had no impact on how they rate the sad movies. T. Elias et al. (2022) explored and analyzed neural networks such as CNN, VGGNet, Inception, MobileNet, and DenseNet for developing a mood-based movie recommendation system [4]. The accuracy of the models concluded that GG16, InceptionV3, and MobileNetV2 have higher performance compared to CNN and DenseNet201 for mood detection. Saraswat et al. (2020) proposed a top-N recommendation model based on emotion analysis of the reviews/comments of the item such as movies using lexical ontology and cosine similarity [5]. The authors utilized an emotion lexicon that categorizes it into six basic emotions. The authors demonstrated the performance advantage of item-based cosine similarity over rating-based cosine similarity which is generally used in collaborative filtering.

The authors in [6], developed a music recommender system using contextual embeddings to extract the context out of the song lyrics. The model suggests the most similar songs to the user by identifying the semantic similarity between the lyrics. The authors state that "this model is different from collaborative-based filtering methods as it does not consider the popular opinion of the user". A. Patel et al. (2018) did a comparative analysis of various music recommendation systems such as 'Graph-based', 'Context-aware Personalization-based', and 'Graph-based Novelty Research on Music Recommendation' [7]. E. Sarin et al. (2021) stated that the current music recommendation systems are "not providing a selection of music based on sentiment and are not adaptive to modern and latest user conditions" [8]. The authors proposed a sentiment-based music recommendation framework named 'SentiSpotMusic'. The authors did data preparation, and feature engineering using TF-IDF (Term Frequency and Inverse Document Frequency) which generated feature vectors for all the songs by assigning values to words of the lyrics.

The current state of the art in the field of recommendation systems is hybrid models that combine Neural Network techniques like DNN with collaborative filtering such as Neural Collaborative Filtering (NCF). In [9], the performance of NCF is compared with traditional collaborative filtering (CF) and it was concluded that NCF is far better than the normal CF in terms of model accuracy and reducing errors. For modeling user-item interaction, NCF combines the non-linearity of the neural network and the linearity of general matrix factorization (GMF). Multi-Layer Perceptron (MLP) is used to learn the interaction function between user and item using TensorFlow and Keras. Four hidden layers with ReLu activation function are used along with the output layer to generate a matrix which is combined with the GMF to provide better recommendations.

VIII. Conclusion and Future Work

In Conclusion, our work has explored something very significant. That is the need for personalized stress relief. We have seen how stress affects the overall health of a person. And the traditional one-size-fits-all solutions aren't adequate. By working on an Emotion-based movie and music recommendation system, we aimed to fulfill the inadequacy. Our tailored suggestions aim to help people manage their stress and find calm in their chaotic lives.

A. Conclusion

Our work has uncovered some exciting findings. Firstly, it is clear how small differences in a person's mood and choices have major implications on their relaxation domain. The cookie-cutter approaches of the past are not the perfect fit for such a delicate matter. By using user-centered advanced machine learning techniques. We have developed a system that truly understands and listens to a person's unique requirements and recommends and responds accordingly. For both music and movies, our system tries to provide a holistic approach to stress relief.

We have also discovered the importance of interdisciplinary collaboration for good solutions. Our system could not be designed solely by machine learning or psychiatric analysis. The current system which considers the complex role of emotions sets us apart from regular solutions.

B. Future Work

As we look ahead, there is still plenty of room for innovation and betterment. The first area we are eager to explore is to refine our machine learning algorithms even further. By fine-tuning the algorithms, we can make sure that the recommendations are tailored to each individual.

There is also a possibility of adding recommendations for other activities aimed at relieving stress. This could include meditation, sports, or any interactive experiences.

But most importantly, we will give precedence to user feedback and review to improve the application. This will make sure to keep the system responsive to the changing needs of the user's preferences. This aligns with our user-centered approach and will help us in making our system personalized and accessible to all.

References

[1] S. Jain, S. P. Narayan, R. K. Dewang, U. Bhartiya, N. Meena, and V. Kumar, "A Machine Learning based Depression Analysis and Suicidal Ideation Detection System using Questionnaires and Twitter," in *2019 IEEE Students Conference on Engineering and Systems (SCES)*, 2019, pp. 1–6.

[2] L. Surace, M. Patacchiola, E. B. Sönmez, W. Spataro, and A. Cangelosi, "Emotion recognition in the wild using deep neural networks and Bayesian classifiers," in *Proceedings of the 19th ACM International Conference on Multimodal Interaction (ICMI '17)*. Association for Computing Machinery, 2017, pp. 593–597.

[3] P. Winoto and T. Y. Tang, "The role of user mood in movie recommendations," *Expert Systems with Applications*, vol. 37, no. 8, pp. 6086–92, 2010.

[4] T. Elias, U. S. Rahman, and K. A. Ahamed, "Movie Recommendation Based on Mood Detection using Deep Learning Approach," in *2022 Second International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT)*, pp. 1–6.

[5] M. Saraswat, S. Chakraverty, and A. Kala, "Analyzing emotion based movie recommender system using fuzzy emotion features"," *International Journal of Information Technology (Singapore. Online)*, vol. 12, no. 2, pp. 467–72, 2020.

[6] V. Gupta, J. S, and S. Kumar, "Songs Recommendation using Context-Based Semantic Similarity between Lyrics," in *2021 IEEE India Council International Subsections Conference (INDISCON)*, 2021, pp. 1–6.

[7] A. Patel and R. Wadhvani, "A Comparative Study of Music Recommendation Systems," in *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, 2018, pp. 1–4.

[8] E. Sarin, S. Vashishtha, S. Megha, and Kaur, "SentiSpotMusic: a music recommendation system based on sentiment analysis," in *2021 4th International Conference on Recent Trends in Computer Science and Technology (ICRTCST)*, 2022, pp. 373–378.

[9] A. Girsang, A. Suganda, J. Wibowo, and Roslynlia, "Comparison between Collaborative Filtering and Neural Collaborative Filtering in Music Recommendation System," *Technology, and Engineering Systems Journal*, vol. 6, no. 1, pp. 1215–1236, 2021.