## **Project Proposal**

#### • **Group Members:**

- o Alina Rashid (ar6647)
- o Samin Rizwan (sr06989)
- o Ayyan Fawad (af06526)

#### • <u>Title: Project Proposal Ideas</u>

- o Music Recommendation System "Moody"
- o Health care chat-bot "Life-line"

#### **Music Recommendation System: MOODY**

#### • Problem being addressed:

People typically enjoy listening to music that matches their mood. However, music recommendation algorithms generally consider the user's current artist, album, or genre that they listen to and therefore recommend music according to it. We want to cater a solution for when they do not want to listen to their usual music and want to vibe according to their mood. Thus we came up with a solution to that issue.

#### • Solution:

Our music recommendation system (app) focuses its recommendations on the user's mood (in addition to the artist and genre they usually listen to). For example, the user listens to "dance" records more often, so our recommendation system would suggest uplifting or more dance tunes from different artists. Consequently, their moods are boosted, making them love the music more. Additionally, this helps the user experiment with different genres of music based on the type of mood they are in and allows them to expand their musical preferences.

### • Resources to be used (libraries, datasets, etc)

We are going to be using the Numpy library, Scikit learn library and Pandas. And Pygame for the front end.

# • Core part to be developed by yourself (Here you will describe the core AI part on your project that you will develop yourself rather than using some existing library).

Utilizing the recommendation system algorithm that we studied in class is our strategy for the core component of the system. We have categorized songs into these categories based on mood using the Spotify dataset.

['valence', 'old', 'acousticness', 'danceability', 'duration\_ms', 'energy', 'explicit', 'instrumentalness', 'key', 'liveness', 'loudness', 'mode', 'popularity', 'speechiness', 'tempo']

We can cluster the songs using K-means in order to understand how to build a better recommendation system.

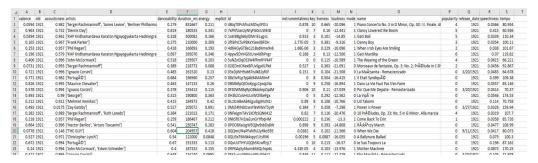
The recommendation algorithm would go about doing something like this:

- Compute the average vector of the audio and metadata features (the mood list) for each song the user has listened
- Find the n-closest data points in the dataset (excluding the points from the songs in the user's listening history) to this average vector.
- Take these n points and recommend the songs corresponding to them

This algorithm follows a common approach that is used in content-based recommender systems.

#### • The dataset we are using is from Kaggle.

https://www.kaggle.com/code/vatsalmavani/music-recommendation-system-using-spotify-dataset/data



#### Health Care chat-bot: Life Line

#### • Problem being addressed:

Even for a small issue we need to consult a doctor by taking an appointment which requires time and cost. Sometimes there is an emergency, in this fast paced world we need quick pre diagnosis to know if the situation is critical enough to even go to the doctor. Thus, our AI app is the solution to that issue.

#### • Solution:

Our healthcare app predicts the disease or illness for the user when they input the symptoms they are facing. This helps in quick diagnose of the illness and not only saves the user's time but also act as a doctor for the early diagnosis of a disease to ensure the treatment can take place on time and lives can be saved It is beneficial for doctors as it provides them a data base and helps them outline even minor issues that may not be as minor as they seem providing them a holistic view on things or guiding them to recommend their patients to some specific specialist.

### • Resources to be used (libraries, datasets, etc)

We are going to be using the Numpy library, Scikit learn library and Pandas and Pygame for the front end.

# • Core part to be developed by yourself (Here you will describe the core AI part on your project that you will develop yourself rather than using some existing library).

Our approach for the core part implementation is going to involve classification algorithms that we have learned in our course.

Gathering the Data: Data preparation is the primary step for any machine learning problem. We will be using a dataset from Kaggle for this problem. This dataset consists of four CSV files for training and for testing.

Cleaning the Data: Cleaning is the most important step in a machine learning project. The quality of our data determines the quality of our machine learning model. So it is always necessary to clean the data before feeding it to the model for training.

Model Building: After gathering and cleaning the data, the data is ready and can be used to train a machine learning model. We will be using this cleaned data to train the Support Vector Classifier, Naive Bayes Classifier algorithm, and Random Forest Classifier. We will be using a confusion matrix to determine the quality of the models.

Inference: After training the three models we will be predicting the disease for the input symptoms by combining the predictions of all three models. This makes our overall prediction more robust and accurate.

#### • The dataset we are using is from Kaggle.

https://www.kaggle.com/code/ronikdedhia/disease-sympton-prediction/data



A A	В	С	D	E	F	G	н	- (	J	K	L	M	N	0	P	Q	
1 Symptom	weight																Т
2 itching		1															
skin_rash		3															
4 nodal_skin_eruptions		4															
5 continuous_sneezing		4															
6 shivering		5															
7 chills		3															
8 joint_pain		3															
stomach_pain		5															
10 acidity		3															
11 ulcers_on_tongue		4															
2 muscle_wasting		3															
13 vomiting		5															
4 burning_micturition		6															
5 spotting_urination		6															
6 fatigue		4															
7 weight_gain		3															
18 anxiety		4															
9 cold_hands_and_feets		5															
0 mood_swings		3															
1 weight_loss		3															
2 restlessness		5															
2 latharm	verity (+)	2															